



Universidad de las Ciencias Informáticas

Facultad 6

Extensión de graficación v2.0 para PostgreSQL

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



Autora: Dalilys Martínez Gutiérrez

Tutores: MsC. Yudisney Vazquez Ortiz

MsC. Anthony Rafael Sotolongo León

La Habana, junio de 2013

Año 55 de la Revolución

DECLARACIÓN DE AUTORÍA

Por este medio declaro que soy la única autora de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI) a hacer uso del mismo en su beneficio.

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

Dalilys Martínez Gutiérrez

Tesista

MsC. Yudisney Vazquez Ortíz

Tutora

MsC. Anthony R. Sotolongo León

Tutor

DATOS DE CONTACTOS

Tutora:

MsC. Yudisney Vazquez Ortíz

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: yvazquezo@uci.cu

Tutor:

MsC. Anthony Rafael Sotolongo León

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: asotolongo@uci.cu

AGRADECIMIENTOS

A mi mamá por estar siempre pendiente de mí, por cuidarme, apoyarme y ofrecerme lo mejor en cada momento.

A mi familia: mi hermana, mi abuela, mi papá, mis tías, primas y primos, mi suegra, a todos por su apoyo y cariño.

A mi novio por estar siempre a mi lado, soportarme y apoyarme incondicionalmente.

A mis tutores por su tiempo y dedicación.

A todos los amigos con los que he compartido estos 5 años.

DEDICATORIA

A mi mamá por todo su esfuerzo, dedicación y cariño.

RESUMEN

Existen variadas herramientas informáticas útiles y de calidad para la graficación de datos, que permiten la creación de gráficos de gran variedad y facilitan el manejo de la información. En esta investigación se desarrolla la extensión *pgr-graphic v2.0* para la generación de gráficos desde el gestor PostgreSQL. El objetivo de incorporar esta extensión al gestor es el de poder representar gráficamente datos contenidos en una base de datos PostgreSQL, permitiendo al gestor además de guardar y gestionar gran cantidad de datos, brindar la posibilidad de análisis y salidas gráficas, prescindiendo de servicios externos. Para el desarrollo de la extensión se realizó el análisis y selección de la metodología, tecnologías y herramientas. Se identificaron las funcionalidades que debe cumplir, se realizó el diseño y a partir esto se implementó la solución. Para comprobar el correcto funcionamiento se realizaron pruebas de aceptación arrojando resultados satisfactorios.

Palabras claves: graficación de datos, graficación desde PostgreSQL, extensión de PostgreSQL

ÍNDICE DE CONTENIDOS

INTRODUCCIÓN	1
CAPÍTULO 1: HERRAMIENTAS PARA LA GRAFICACIÓN DE INFORMACIÓN	5
1.1. Almacenamiento de la información	5
1.1.1. Sistemas de gestión de bases de datos	6
1.2. Presentación de la información	7
1.2.1. Gráficos de barras o columnas	9
1.2.2. Gráficos de líneas	11
1.2.3. Gráficos circulares	12
1.2.4. Gráficos de áreas	14
1.2.5. Gráficos de dispersión	14
1.2.6. Gráficos de caja-bigotes	15
1.2.7. Gráficos de superficie	15
1.2.8. Gráficos mixtos	16
1.2.9. Gráficos a implementar en pgr-graphic v2.0	16
1.3. pgr-graphic v1.0	16
1.4. Metodología de desarrollo de software	18
1.5. Tecnologías a emplear para el desarrollo de la solución	19
1.5.1. PostgreSQL 9.1	19
1.5.2. R	20
1.5.3. PL/R	21
1.5.4. Python	21
Conclusiones del capítulo	21

CAPÍTULO 2: CARACTERÍSTICAS Y DISEÑO DE LA SOLUCIÓN	23
2.1. Modelo de dominio	23
2.2. Descripción del sistema propuesto	25
2.2.1. Historias de Usuario	25
2.2.2. Lista de reserva del producto	30
2.2.3. Plan de iteraciones	32
2.3. Modelo de diseño	33
2.3.1. Tarjetas CRC	33
Conclusiones del capítulo	38
CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN	39
3.1. Tareas de ingeniería	39
3.2. Estándar de codificación	41
3.3. Implementación de las historias de usuario	47
3.4. Pruebas	47
3.4.2. Diseño de casos de prueba y registro de no conformidades	49
3.4.3 Resultados de las pruebas de aceptación	62
Conclusiones del capítulo	63
CONCLUSIONES GENERALES	64
RECOMENDACIONES	65
REFERENCIAS BIBLIOGRÁFICAS	66

ÍNDICE DE TABLAS

Tabla 1: Subtipos de gráficos de barras	10
Tabla 2: Subtipos de gráficos de líneas	12
Tabla 3: Subtipos de gráficos circulares	13
Tabla 4: Subtipos de gráficos de áreas	14
Tabla 5: Subtipos de gráficos de dispersión	15
Tabla 6: Gráficos a incorporar a la extensión pgr-graphic v2.0	16
Tabla 7: Descripción de los conceptos del diagrama de dominio	24
Tabla 8: Historia de usuario Generar gráfico de barras	25
Tabla 9: Historia de usuario Generar gráfico de líneas	26
Tabla 10: Historia de usuario Generar gráfico de circular	27
Tabla 11: Historia de usuario Generar gráfico de dispersión	28
Tabla 12: Historia de usuario Generar gráfico de caja	28
Tabla 13: Historia de usuario Salvar gráficos en la base de datos	29
Tabla 14: Lista de reserva del producto	30
Tabla 15: Plan de iteraciones	33
Tabla 16: Tarjeta CRC Generar gráfico de barras simples	34
Tabla 17: Tarjeta CRC Generar gráfico de barras agrupadas	35
Tabla 18: Tarjeta CRC Generar gráfico de barras apiladas	35
Tabla 19: Tarjeta CRC Generar gráfico de histograma	35
Tabla 20: Tarjeta CRC Generar gráfico de líneas	36
Tabla 21: Tarjeta CRC Generar gráfico circular	36
Tabla 22: Tarjeta CRC Generar gráfico de caja	36

Tabla 23: Tarjeta CRC Generar gráfico de dispersión _____	37
Tabla 24: Tarjeta CRC Salvar imágenes en la base de datos _____	37
Tabla 25: Tarea de ingeniería Generar gráfico de barras simples _____	39
Tabla 26: Tarea de ingeniería Generar gráfico de barras agrupadas _____	40
Tabla 27: Tarea de ingeniería Generar gráfico de histograma _____	40
Tabla 28: Tarea de ingeniería Generar gráfico de barras apiladas _____	41
Tabla 29: Secciones para probar la historia de usuario Generar gráfico de barras _____	50
Tabla 30: Descripción de variables de la historia de usuario Generar gráficos de barras _____	52
Tabla 31: Listado de relaciones _____	54
Tabla 32: Matriz de datos de la historia de usuario Generar gráficos de barras _____	56
Tabla 33: No conformidades detectadas en la aplicación de los casos de pruebas _____	62

ÍNDICE DE FIGURAS

Figura 1: Porcentaje de preferencia de un tipo específico de gráfico por los usuarios	9
Figura 2: Proceso de graficación con el empleo de una solución externa	17
Figura 3: Proceso de graficación con pgr-graphic	18
Figura 4: Modelo de dominio	24
Figura 5: Ejemplo correcto de ruptura de línea	42
Figura 6: Ejemplo correcto de comentario	43
Figura 7: Ejemplo correcto de declaración	43
Figura 8: Ejemplo correcto de declaración de función	43
Figura 9: Ejemplo de sentencia simple	44
Figura 10: Ejemplo correcto de sentencia WHILE	44
Figura 11: Ejemplo correcto de sentencias IF, IF-ELSE, IF ELSE-IF ELSE	45
Figura 12: Ejemplo correcto de aplicación de espacios en blanco	45
Figura 13: Ejemplo correcto de aplicación de espacios en blanco	46
Figura 14: Ejemplo correcto de convención de funciones	46
Figura 15: Ejemplo correcto de convención de nombres de objetos o variables	46
Figura 16: Estructura de pgr-graphic v2.0	47
Figura 17: Consulta definida por el usuario ejecutada en el pgAdmin	60
Figura 18: Respuesta del sistema al ejecutar la consulta correctamente	60
Figura 19: Inserción de la imagen en la tabla de la base de datos	61
Figura 20: Gráfico generado como resultado de la consulta	61
Figura 21: Resultados de las pruebas de aceptación al final de cada iteración	63

INTRODUCCIÓN

El gradual desarrollo alcanzado por las Tecnologías de la Información y las Comunicaciones (TIC) facilita la creación, distribución y manipulación de grandes volúmenes de información. El volumen de datos generado cada año crece de forma exponencial en todos los sectores. Tal es la velocidad con la que aumenta, que el 90% de la información existente ha sido generada en los últimos dos años. (Dominguez, Oscar, 2012)

Según el informe de 2011 de *McKinsey Global Institute*¹ sobre el aumento del volumen de información a escala mundial, en 2010 se generaron y almacenaron más de 6 500 *petabytes* de datos (aproximadamente 6 500 millones de *gigabytes*). Además, dicho informe prevé un crecimiento de un 40% anual, con Norteamérica y Europa a la cabeza. (Dominguez, Oscar, 2012)

Dicha situación deriva en que los sistemas informáticos sean cada vez más empleados en todas las esferas de la sociedad. Estos emplean, generalmente, software específico para el soporte de la información que manipulan; tal es el caso de los Sistemas de Gestión de Bases de Datos (SGBD), software que permite administrar los datos contenidos en las bases de datos del sistema y que, posteriormente, se convertirán en información; permitiendo a los usuarios procesar, describir, manipular y recuperar los datos almacenados en una base de datos. (Ansejo, 2006)

La forma en que se muestra la información a usuarios finales en los sistemas informáticos es fundamental para que el sistema sea calificado de eficaz y, que el usuario logre un alto nivel de satisfacción. En la actualidad la exigencia de los usuarios ha contribuido a que el nivel de calidad en la presentación de datos aumente. Con el fin de cumplir con las expectativas de estos, una de las tendencias más destacada es la graficación de la información, facilitando que los datos sean bien presentados, sin necesidad de acudir al texto para entender o interpretar la información mostrada.

Existe una gran variedad de herramientas informáticas útiles y de calidad para la graficación de la información, ejemplo de estas son:

¹ Firma de consultoría de gestión global, asesora de confianza de las empresas líderes en el mundo, los gobiernos y las instituciones. (Mckinsey&Company, 2012)

- Los programas de hojas de cálculo, que incluyen una funcionalidad para elaborar gráficos de forma sencilla, en los que sólo se deben seleccionar los datos de la hoja de cálculo y elegir el tipo de gráfico que mejor se adapte al objetivo, entre los que destacan *Microsoft Excel* y *OpenOffice.org Calc*.
- Sitios de Internet, que ofrecen servicios de creación de gráficos que no sólo son más llamativas que las elaboradas con una hoja de cálculo, sino que ofrecen también, animación e interactividad, entre estos se encuentran *Chartle*, *ChartGO*, *Charts* y *Chartle.net*.
- Potentes herramientas de tratamiento de datos y análisis estadístico, que presentan una interfaz gráfica de usuario amigable y algunas permiten a través de lenguajes de programación y comandos personalizar sus opciones, entre estas se encuentran *SPSS*, *S-PLUS*, *MINITAB*, *MATLAB* y *R*.

Estas soluciones brindan la posibilidad de crear gráficos de gran variedad y facilidad en el manejo de bases de datos.

Los proyectos de la Universidad de las Ciencias Informáticas (UCI) en aras de ofrecer soluciones más completas, han ido generando un interés creciente por el estudio y uso de estas herramientas, que permiten la graficación de la información generada y gestionada en sus sistemas. La selección de una u otra depende de las necesidades del proyecto y de la compatibilidad de las tecnologías que se emplean con las de la herramienta seleccionada. Existe el inconveniente de que estas herramientas de graficación son externas a las aplicaciones desarrolladas en los proyectos. Por lo que es necesario enviarles como parámetros los datos a graficar o la fuente de donde obtenerlos, haciendo el proceso de graficación más complejo.

A raíz de esto, el departamento de PostgreSQL del Centro de Tecnologías de Gestión de Datos (DATEC), dedicado a potenciar el empleo y desarrollo del sistema de gestión de bases de datos PostgreSQL, por ser este un gestor de código abierto con una amplia gama de funcionalidades del más alto nivel, implementó una extensión para la versión 9.1 del gestor PostgreSQL llamada *pgr-graphic*, que permite la graficación de los datos desde el gestor. Aun cuando esta solución permite la graficación desde el gestor prescindiendo de servicios externos a las aplicaciones desarrolladas por los proyectos, tiene una serie de insuficiencias que impiden que sea una herramienta competitiva, entre ellas destacan:

- Las pocas opciones de tipos de gráficos que ofrece, limitando a los proyectos productivos al empleo de una gama reducida de gráficos en sus soluciones.
- Las pocas opciones de configuración de las gráficas, afectando la calidad de la información mostrada.
- La poca documentación sobre la generación de gráficas, dificultando el empleo de la solución por usuarios con poco dominio de la tecnología.

Estos elementos limitan el empleo de la extensión desarrollada por el departamento, por lo que surge el siguiente **problema a resolver**: las escasas opciones de gráficos que brinda la extensión *pgr-graphic*, dificultan la graficación desde PostgreSQL y por ende, su empleo en los proyectos productivos de la Universidad.

Para dar solución al problema antes planteado se define como **objeto de estudio**: la representación de la información mediante gráficos; enmarcados en el **campo de acción**: la generación de gráficos en PostgreSQL, debido a que se define como **objetivo general**: desarrollar una nueva versión de la extensión *pgr-graphic* que corrija las limitaciones de la anterior, para facilitar su empleo en los proyectos productivos de la Universidad.

A partir del objetivo general, se derivan los siguientes **objetivos específicos**:

- Definir los tipos de gráficos a implementar, así como las tecnologías a utilizar para el desarrollo de la solución.
- Definir los requisitos y el diseño de la extensión de graficación *pgr-graphic v2.0*.
- Implementar la extensión de graficación *pgr-graphic v2.0*.
- Validar la extensión desarrollada mediante la realización de pruebas.

Para cumplir con los objetivos planteados se llevaron a cabo las siguientes **tareas de la investigación**:

- Análisis de las tecnologías existentes para el almacenamiento de la información.
- Análisis de los medios existentes para la presentación de la información.
- Caracterización de la extensión *pgr-graphic*.
- Selección de las tecnologías a utilizar para el desarrollo de la solución.

- Levantamiento de requisitos de la extensión *pgr-graphic* v2.0.
- Desarrollo de los artefactos correspondientes a la metodología seleccionada.
- Planificación de las iteraciones necesarias para la implementación de *pgr-graphic* v2.0.
- Implementación de las nuevas características *pgr-graphic* v2.0.
- Diseño de los casos de pruebas para los tipos de gráficos implementados.
- Realización de pruebas de caja negra para validar los tipos de gráficos implementados.

El informe de la investigación está organizado en tres capítulos:

- Capítulo 1. Herramientas para la graficación de información: en este capítulo se abordan temas como el almacenamiento y presentación de la información, aludiendo a algunas de las herramientas y técnicas más empleadas para lograr éxito en ambos, como son las bases de datos, los sistemas de gestión de bases de datos y los gráficos estadísticos. Se muestra además en el capítulo, una caracterización de la extensión *pgr-graphic* y de las herramientas y tecnologías a emplear en la implementación de una versión superior a esta.
- Capítulo 2. Características y diseño de la solución: en este capítulo se expone el diseño del sistema propuesto, como parte de este, se presenta un modelo de dominio con el fin de proporcionar un mejor entendimiento de los principales conceptos del negocio y capturar con mayor grado de detalle lo necesario para comprender el sistema, una descripción de las historias de usuario definidas, las tarjetas CRC, los requisitos tanto funcionales como no funcionales a tener en cuenta en su desarrollo, agrupados en la lista de reserva del producto, así como un plan de iteraciones que contempla el tiempo de implementación del sistema.
- Capítulo 3. Implementación y pruebas de la solución: en este capítulo se exponen detalles de las fases de implementación y pruebas de la extensión, se relacionan las tareas asignadas a cada historia de usuario para desarrollar la implementación, el estándar de codificación utilizado, así como una descripción de las pruebas realizadas a la extensión.

CAPÍTULO 1

HERRAMIENTAS PARA LA GRAFICACIÓN DE INFORMACIÓN

Actualmente la información aumenta a ritmo acelerado, existen gracias a la tecnología más canales para su transmisión, facilidad de registro y almacenamiento, sin embargo, en ocasiones deja de ser óptima porque no es presentada apropiadamente. En este capítulo se abordan temas como el almacenamiento y presentación de la información, aludiendo a algunas de las herramientas y técnicas más empleadas para alcanzar el éxito en ambos, como son las bases de datos, los sistemas de gestión de bases de datos y los gráficos estadísticos. Se muestra también en este capítulo una caracterización de la extensión *pgr-graphic* y, de las herramientas y tecnologías a emplear en la implementación de una versión superior a esta.

1.1. Almacenamiento de la información

En la actualidad se maneja un volumen de datos cada vez mayor que crece de manera exponencial impulsado por el desarrollo de las Tecnologías de la Información y las Comunicaciones. Dado el volumen de datos generados, existe la necesidad de contar con tecnología capaz de almacenar, procesar y analizar dicha información de manera rápida y eficiente, con el menor costo posible.

Una de las herramientas más empleadas para el almacenamiento de la información son las bases de datos, el concepto de base de datos surgió a fines de la década del 60 como propuesta de solución a un conjunto de problemas técnicos y administrativos presentados en el manejo de archivos. A medida que los sistemas de información se volvían más complejos y se extendían a nuevas áreas de la operación de una empresa o institución, las dificultades en mantener su funcionamiento y costo bajo control se acentuaban.

Evitar la redundancia de los datos almacenados resultaba complejo y se obstaculizaba el acceso rápido a la información por parte de los interesados, debido a que no existía una estructura adecuada para el almacenamiento. A raíz de esto, las tecnologías de bases de datos proponían que toda la información, apropiadamente organizada y codificada, se colocara en un gran reservorio llamado *base de datos*. (Mendelzon, 2000)

Desde entonces el estudio de las bases de datos ha sido extenso, existiendo diversas definiciones del término ofrecida por especialistas del tema.

Gary Hansen en su libro *Diseño y administración de bases de datos*, la define como "...una colección de elementos de datos interrelacionados que pueden procesarse por uno o más sistemas de aplicación". (Hansen James, 2010)

Olga Capote, en *Introducción a las bases de datos: el modelo relacional*, las caracteriza como el "fondo común de información almacenada en una computadora para que cualquier persona o programa autorizado pueda acceder a ella, independientemente de su procedencia y del uso que se haga". (Capote Pons, 2005)

Alejandro Díaz especifica que "es un almacén de datos relacionados con diferentes modos de organización,... representa algunos aspectos del mundo real, aquellos que le interesan al usuario. Y que almacena datos con un propósito específico. Con la palabra "datos" se hace referencia a hechos conocidos que pueden registrarse, como son números telefónicos, direcciones, nombres". (Díaz Gutierrez, 2012)

Se asume para la investigación que una base de datos es un conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso. La utilización de las bases de datos ofrece como ventaja consistencia, integridad, seguridad, flexibilidad y rapidez al obtener datos.

1.1.1. Sistemas de gestión de bases de datos

Los sistemas de gestión de bases de datos son empleados para la manipulación de las bases de datos.

Mendelzon en su libro *Introducción a las bases de datos relacionales* lo define como un sistema de software que permite la definición de bases de datos; así como la determinación de las estructuras de datos necesarias para el almacenamiento y consulta de los datos. Con un grupo de funciones provistas para la consulta y actualización de los datos, el mantenimiento de esquemas y el manejo de transacciones. Son una herramienta que les permite a varios usuarios acceder a los datos al mismo tiempo; garantizan la confidencialidad, calidad, seguridad e integridad de los datos, así como un acceso fácil y eficiente a los mismos. (Mendelzon, 2000)

Olga Capote, en *Introducción a las bases de datos: el modelo relacional*, especifica que es el “conjunto de elementos software con capacidad para definir, mantener y utilizar una base de datos”. (Capote Pons, 2005)

Elmasri Navathe lo define como “un conjunto de programas que permite a los usuarios crear y mantener una base de datos. Es un software de propósito general que facilita el proceso de definir, construir y manipular bases de datos de diversas aplicaciones”. (Navathe, 2005)

Mercedes Marques defiende la idea de que es “una aplicación que permite a los usuarios definir, crear y mantener la base de datos, y proporciona acceso controlado a la misma”. (Marques, 2001)

De manera general, los sistemas de gestión de bases de datos son un conjunto de programas que crean, administran y mantienen toda la información de una base de datos; garantizando la integridad, confidencialidad y seguridad de las bases de datos, sirviendo como vía de comunicación entre el usuario y las aplicaciones que utilizan estas bases de datos.

1.2. Presentación de la información

Uno de los retos actuales, derivados del elevado volumen de información que circula mundialmente, consiste en cómo mostrarla de forma más atractiva, siendo la presentación de la información clave en el logro de una correcta interpretación de la misma.

Los gráficos son medios popularizados y, a menudo, los más convenientes, para presentar datos; son llamativos, facilitan a los usuarios la visualización de comparaciones, tramas y tendencias de los datos y, constituyen por sí mismos una poderosa herramienta para el análisis; siendo en ocasiones el medio más efectivo, no sólo para describir y resumir la información, sino también para analizarla.

*William Playfair*², pionero de la estadística gráfica, estableció los siguientes principios: (Playfair, 1801)

- El método gráfico es una forma de simplificar lo tedioso y lo complejo.
- Las personas ocupadas necesitan alguna clase de ayuda visual.
- Un gráfico es más accesible que una tabla.
- El método gráfico es concordante con los ojos.

² *William Playfair* (1759-1823), escocés ingeniero y economista, fundador de los métodos gráficos de estadísticas.

- El método gráfico ayuda al cerebro, ya que permite entender y memorizar mejor.

Se dice que un gráfico es la representación de datos, generalmente numéricos, mediante líneas, superficies o símbolos, para ver la relación que esos datos guardan entre sí. También puede ser un conjunto de puntos, que se plasman en coordenadas cartesianas, y sirven para analizar el comportamiento de un proceso, o un conjunto de elementos o signos que permitan la interpretación de un fenómeno. (Scribd, 2012)

Arnold Hoffman lo define como la “representación mediante un dibujo o por otros medios análogos de un hecho dado u observado”. (Hofmann, 2010)

Otra definición dada por *Hoffman* es que los gráficos son una “representación en imagen de datos numéricos o de relaciones de orden, que sean de naturaleza espacial, temporal o abstracta, especialmente casuales”. (Hofmann, 2010)

Una denominación de gráfico o gráfica pudiera ser, la representación de datos generalmente numéricos, mediante recursos gráficos (líneas, vectores, superficies o símbolos), combinando la utilización de sombreado, colores, puntos, símbolos, números, texto y un sistema de referencia, que muestre visualmente la relación matemática o correlación estadística que guardan entre sí.

Existen una gran variedad de gráficos. El motivo de que existan tantos tipos diferentes no es solamente estético, cada uno de ellos está especialmente diseñado para representar los datos de una manera distinta. Por lo tanto, si se desea ganar visibilidad y alcance con el contenido mostrado, obtener la máxima eficacia al crear gráficos y presentar los datos de la mejor manera posible, es necesario tener en cuenta que cada tipo de gráfico está destinado para una labor específica.

La compañía *Microsoft* publica que los gráficos más generados en la herramienta de hoja de cálculo *Excel* de su suite ofimática son los de barras o columnas, líneas, circulares, dispersión. (Microsoft, 2013)

La empresa *excellentias.com* al realizar una encuesta a usuarios de *Microsoft Excel*, sobre los tipos de gráficos estadísticos que más se generan en la herramienta obtuvo resultados similares a los que publica *Microsoft*. (*excellentias*, 2013)

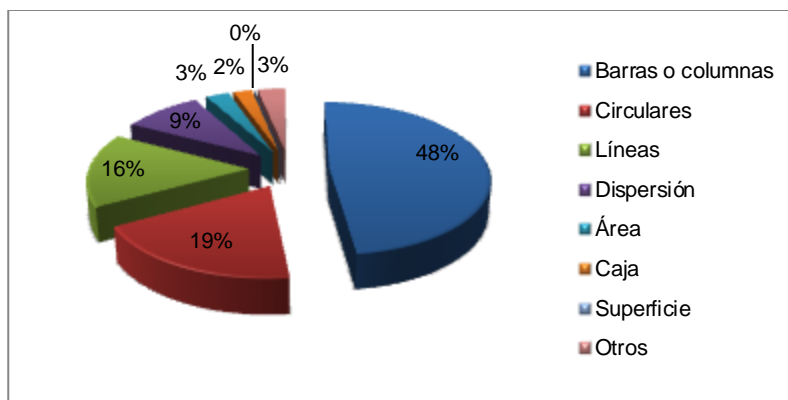


Figura 1: Porcentaje de preferencia de un tipo específico de gráfico por los usuarios (excellentias, 2013)

Teniendo en cuenta estos resultados se realiza un estudio de estos tipos de gráficos como posibles propuestas a incluir en la solución.

1.2.1. Gráficos de barras o columnas

Un gráfico de barras es aquella representación bidimensional en que los objetos gráficos elementales son un conjunto de rectángulos dispuestos paralelamente, de manera que la extensión de los mismos es proporcional a la magnitud que se quiere representar. Las barras o rectángulos pueden aparecer horizontal o verticalmente (en el caso de que aparezcan vertical se denomina gráfico de columnas). (Infovis, 2012)

Un gráfico de barras consta al menos de: (Infovis, 2012)

- Un eje cuantitativo con una escala lineal (puede contener valores negativos), que sirve de referencia a la magnitud de la variable en cuestión.
- Un eje nominal o secuencial, en el que se disponen los elementos de la secuencia.
- Un conjunto de rectángulos, cuya extensión paralela al eje cuantitativo es proporcional a la magnitud de los elementos representados en el eje nominal.

Subtipos de gráficos de barras

Es recomendable emplear este tipo de gráfico estadístico para variables nominales y secuenciales (especialmente las temporales), no siendo muy apropiado para representar datos cuantitativos. Es útil

para mostrar cambios de datos en un período de tiempo o para realizar comparaciones entre elementos. (Infovis, 2012)

Existe una gran variedad de estos, cada uno de ellos especialmente diseñado para representar los datos de una manera distinta. La tabla 1 muestra los subtipos más empleados.

Tabla 1: Subtipos de gráficos de barras

No	Subtipo	Descripción
1	Simple	Contiene solamente una serie de datos
2	Agrupados	Contiene varias series de datos; cada serie se representa mediante un conjunto de rectángulos que tienen igual color o textura; en cada secuencia los rectángulos suelen estar juntos formando un grupo, dejándose un espacio entre grupos
3	Solapado	Gráfico de barras agrupadas, en el que los elementos de un grupo en vez de aparecer adosados se solapan parcialmente
4	Apilados	También llamado segmentado o extendido, es similar al agrupado pero cada uno de los segmentos en que está dividida la barra pertenece a una serie de datos diferente
5	Enlazado o conectado	Al añadir líneas que enlacen los lugares donde se cambia de segmento se llama apilado enlazado o conectado
6	Cien por cien	Gráfico apilado en el que la altura del total cubre todo el eje cuantitativo de forma que lo que muestran los segmentos es el porcentaje con que contribuyen al total, que representa el 100%

7	Flotantes, bidireccionales o aparejados	La línea de valor cero actúa como separador de dos gráficos de barra que comparten el 0 como línea de base pero en el que cada uno muestra su barra en dirección contraria
8	Pictóricos	Las barras están constituidas por una serie de símbolos que representan la naturaleza de los datos
9	De Rangos	La extensión máxima y mínima de las barras indican los rangos superior e inferior de validez de los datos considerados
10	Histogramas	Despliega la variabilidad dentro de un proceso; toma datos variables (tales como alturas, pesos, densidades, tiempo, temperaturas, etc.) y despliega su distribución

1.2.2. Gráficos de líneas

Un gráfico de líneas es aquella representación gráfica que permite mostrar datos continuos a lo largo del tiempo, expresados con respecto a una escala común. Los gráficos de líneas muestran los datos en forma de puntos y todos los puntos de la misma serie se unen mediante una línea. Cada valor aparece representado por un punto que es la intersección entre los datos del eje horizontal y los del eje vertical. (educacion.es, 2012)

Un gráfico de línea consta al menos de: (educacion.es, 2012)

- Valores: los valores de un parámetro determinan el alto de las líneas de ese parámetro en el eje Y. Las etiquetas de los valores aparecen en el eje Y. Cada parámetro consta de un conjunto de valores que aparecen como una línea separada.
- Grupos de series: las series se muestran como etiquetas en el eje X. En los gráficos de líneas, las series suelen estar relacionadas con el tiempo.
- Grupos de parámetros: los parámetros se muestran como líneas separadas en la gráfica. Cada parámetro también se presenta en la leyenda de la gráfica.

Subtipos de gráficos de líneas

Los gráficos de líneas son una buena solución para representar gráficamente datos numéricos. Resultan especialmente útiles para expresar los cambios que se producen en los valores entre las distintas categorías de datos. Son idóneos cuando las etiquetas de categorías son texto y representan valores espaciados de manera uniforme. (educacion.es, 2012)

Existe una gran variedad de estos, el uso de cada uno de estos depende del tipo de información y la forma en que se desee mostrar la misma. La tabla 2 muestra los subtipos más empleados.

Tabla 2: Subtipos de gráficos de líneas

No	Subtipo	Descripción
1	Líneas suavizadas	Presenta los parámetros como líneas separadas; las líneas son curvas, en lugar de rectas; el valor determina el alto de cada columna
2	Líneas con o sin marcadores	Muestra tendencias a lo largo del tiempo o categorías ordenadas, especialmente cuando hay varios puntos de datos y el orden en que se presentan es importante
3	Líneas apiladas con o sin marcadores	Muestra la tendencia de la contribución de cada valor a lo largo del tiempo o categorías ordenadas
4	Líneas 100% apiladas con o sin marcadores	Muestra la tendencia del porcentaje con que cada valor contribuye a lo largo del tiempo o categorías ordenadas

1.2.3. Gráficos circulares

Círculo que se divide (o rebana) desde su punto central, donde cada rebanada representa la frecuencia proporcional de determinada categoría de una variable nominal. (Infovis, 2012)

También son conocidos como gráficos de pastel o gráficas de 360 grados, el número de elementos comparados dentro de un gráfico circular puede ser de más de 5, y los segmentos se ordenan de mayor a menor, iniciando con el más amplio a partir de las 12, como en un reloj. Una manera fácil de identificar los segmentos es sombreando de claro a oscuro, donde el de mayor tamaño es el más claro y el de menor tamaño el más oscuro, el empleo de tonalidades o colores facilita la diferenciación de los porcentajes o proporciones. (Infovis, 2012)

Subtipos de gráficos circulares

Los gráficos circulares se utilizan en aquellos casos donde interesa no sólo mostrar el número de veces que se da una característica o atributo de manera tabular, sino más bien de manera gráfica, de tal forma que se pueda visualizar mejor la proporción en que aparece esa característica respecto del total. Además es posible seleccionar un tipo de gráfico circular específico según la forma en que se desee presentar la información. (Infovis, 2012)

La tabla siguiente muestra los subtipos más empleados.

Tabla 3: Subtipos de gráficos circulares

No	Subtipo	Descripción
1	Circular	Muestra la contribución de cada valor al total
2	Circular o de barras	Muestra los gráficos circulares con valores definidos por el usuario que se extraen del gráfico circular principal y se combinan en un gráfico circular secundario o en un gráfico de barras apiladas. Este tipo resulta útil cuando se quiere que los sectores más pequeños del gráfico circular principal se distingan con mayor facilidad
3	Circular seccionado	Muestra la contribución de cada valor al total destacando los valores individuales

1.2.4. Gráficos de áreas

Los gráficos de áreas muestran las series como un conjunto de puntos conectados por una línea, con un área rellena por debajo de la línea. Los valores se representan por el alto de los puntos con relación al eje Y. Las etiquetas de las categorías aparecen en el eje X. Los gráficos de áreas suelen utilizarse para comparar valores a lo largo del tiempo. (Infovis, 2012)

Subtipos de gráficos de áreas

Los gráficos de área resultan especialmente útiles para expresar los cambios que se producen en los valores entre las distintas categorías de datos. (microsoft, 2012)

La tabla siguiente muestra los más empleados.

Tabla 4: Subtipos de gráficos de áreas

No	Subtipo	Descripción
1	Áreas apiladas	Muestra la tendencia de la contribución de cada valor a lo largo del tiempo u otros datos de categorías
2	Áreas 100% apiladas	Muestra la tendencia del porcentaje con que cada valor contribuye a lo largo del tiempo u otros datos de categorías

1.2.5. Gráficos de dispersión

Un gráfico de dispersión o gráficos XY muestra una serie como un conjunto de puntos. Los valores se representan mediante la posición de los puntos en el gráfico. Las categorías se representan mediante distintos marcadores en el gráfico. (Estadística, 2010)

Subtipos de gráficos de dispersión

Los gráficos de dispersión se usan normalmente para mostrar y comparar valores numéricos, como datos científicos, estadísticos y de ingeniería. Facilitan comparar grandes cantidades de puntos de datos sin tener en cuenta el tiempo. Cuantos más datos se incluyan, mejores comparaciones se podrán realizar. (Estadística, 2010)

Tabla 5: Subtipos de gráficos de dispersión

No	Tipo	Descripción
1	Burbuja	Muestra la diferencia entre dos valores de un punto de datos basándose en el tamaño de la burbuja; cuanto mayor sea la burbuja, mayor será la diferencia entre los dos valores
2	Dispersión sólo con marcadores	Compara pares de valores, ideal si se tienen muchos puntos de datos y cuando las líneas de conexión dificultarían la lectura de datos.
3	Dispersión con líneas suavizadas	Muestra una curva suavizada que conecta los puntos de datos. Las líneas suavizadas pueden mostrarse con o sin marcadores
4	Dispersión con líneas rectas	Muestra líneas de conexión rectas entre los puntos de datos. Las líneas rectas pueden mostrarse con o sin marcadores

1.2.6. Gráficos de caja-bigotes

Los diagramas de Caja-Bigotes (*boxplots o box and whiskers*) son una presentación visual que describe varias características importantes, al mismo tiempo, tales como la dispersión y simetría. Para su realización se representan los tres cuartiles y los valores mínimo y máximo de los datos, sobre un rectángulo, alineado horizontal o verticalmente. (Estadística, 2010)

1.2.7. Gráficos de superficie

Un gráfico de superficie es útil cuando se buscan combinaciones óptimas entre dos conjuntos de datos. Como en un mapa topográfico, los colores y las tramas indican áreas que están en el mismo rango de valores. (Estadística, 2010)

Puede utilizar un gráfico de superficie cuando ambas categorías y series de datos sean valores numéricos.

1.2.8. Gráficos mixtos

Los gráficos mixtos representan dos o más series de datos, cada una con un tipo diferente de gráfico. Son gráficos más vistosos y se usan para resaltar las diferencias entre las series. Pueden presentarse en dos o en tres dimensiones. (Fuentes, 2012)

1.2.9. Gráficos a implementar en pgr-graphic v2.0

Luego de estudiar los distintos tipos de gráficos más empleados, sus características y aplicaciones, teniendo en cuenta las funciones con que cuenta el paquete base de graficación de R, con el cual se desarrolló la versión anterior de la extensión, se deciden incorporar a la extensión los gráficos y subtipos mostrados en la tabla siguiente.

Tabla 6: Gráficos a incorporar a la extensión pgr-graphic v2.0

No	Tipo	Subtipos
1	Barras o columnas	Simples, Agrupadas, Apiladas, Histogramas
2	Líneas	Líneas con o sin marcadores
3	Circulares	Circular
4	Dispersión	Dispersión sólo con marcadores
5	Caja-Bigotes	Caja

1.3. pgr-graphic v1.0

Tradicionalmente los sistemas de gestión de bases de datos, han sido diseñados para guardar y gestionar gran cantidad de datos, por ello las posibilidades de análisis y salidas gráficas se han delegado mayormente en otro software, como hojas de cálculo y paquetes estadísticos.

pgr-graphic es una extensión desarrollada por el departamento PostgreSQL para la versión 9.1 del gestor, que permite crear a partir de datos almacenados en una base de datos, utilizando las funciones del

lenguaje R mediante el lenguaje procedural PL/R, gráficos circulares simples, de dispersión y de barras simples.

Con el empleo de *pgr-graphic* el proceso de graficación se modifica; generalmente cuando se desean graficar datos almacenados en una base de datos, se hace necesario que el usuario solicite los datos desde una aplicación, la cual envía la solicitud a la base de datos, respondiendo esta con la información adecuada, luego la aplicación envía los datos recibidos hacia un componente de graficación y este genera el gráfico correspondiente que luego se muestra en la aplicación. La figura 2 muestra gráficamente este proceso.

Al incluirse la extensión *pgr-graphic* en el gestor, el proceso de graficación se simplifica, como muestra la figura 3, el usuario solicita los datos desde una aplicación, la cual envía la solicitud a la base de datos, respondiendo esta con la información adecuada, entonces en la propia base de datos se genera el gráfico correspondiente, el cual es generado en forma de imagen y almacenado en el directorio de trabajo del motor gráfico R, evitándose la importación-exportación y, por ende, agilizándose el proceso de graficación.



Figura 2: Proceso de graficación con el empleo de una solución externa



Figura 3: Proceso de graficación con pgr-graphic

La versión 1.0 de *pgr-graphic* ofrece entre sus opciones gráficos circulares simples, de dispersión y de barras simples, por lo que:

- Las pocas opciones de tipos de gráficos que ofrece limita a los usuarios que lo utilicen al empleo de una gama reducida de gráficos en sus soluciones.
- Las pocas opciones de configuración de las gráficas afecta la calidad de la información mostrada.
- La poca documentación sobre la generación de gráficas dificulta el empleo de la solución por usuarios con poco dominio de la tecnología.

Todas estas deficiencias evidencian la necesidad de implementar una versión superior, que permita suplir tales necesidades en función de que los usuarios que los empleen en los entornos reales de producción, obtengan de él el mayor provecho posible.

1.4. Metodología de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y un soporte documental para el desarrollo de productos software. Indicando paso a paso las actividades a realizar para lograr el producto informático deseado, indicando qué personas deben participar en el desarrollo de las actividades y qué papel deben jugar. Además detallan la información necesaria para realizar una actividad y qué se debe producir como resultado de su ejecución. (Pressman, 2005)

Programación Extrema o *Extreme Programming*, conocida comúnmente como XP, es una metodología de desarrollo de software enmarcada dentro de las metodologías ágiles de desarrollo. Constituye la metodología más utilizada dentro del grupo de las ágiles. Su objetivo principal es asegurar la producción de software con buena calidad y cubriendo las necesidades y requisitos del usuario. (Beck, 2000)

En la presente investigación se decide utilizar la metodología XP teniendo en cuenta que:

- Se utiliza para la realización de proyectos a corto plazo.
- Fomenta el desarrollo de equipos pequeños: Para solucionar el problema de la investigación planteado solo se cuenta con una persona para todo el ciclo de desarrollo del software.
- Consiste en una programación rápida o extrema: esta metodología centra la atención del equipo en desarrollar las funcionalidades y no en una profunda documentación del proceso de desarrollo.
- El cliente forma parte del equipo de desarrollo: para el desarrollo de la extensión el cliente estará presente y disponible todo el tiempo para el equipo. Gran parte del éxito de la utilización de esta metodología se debe a que es el cliente quien conduce constantemente el trabajo hacia lo que aportará mayor valor de negocio y, los programadores pueden resolver de manera inmediata cualquier duda asociada.

1.5. Tecnologías a emplear para el desarrollo de la solución

1.5.1. PostgreSQL 9.1

PostgreSQL es un sistema de gestión de bases de datos relacional, orientado a objetos y de código abierto, publicado bajo la licencia BSD³. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre y/o apoyada por organizaciones comerciales.

La estabilidad, potencia, robustez, facilidad de administración e implementación de estándares se han tenido en cuenta durante el desarrollo de todas sus versiones. PostgreSQL funciona muy bien con

³ Del inglés *Berkeley Software Distribution*, permite el uso del código fuente. (Esteve, Marta , 2010)

grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema. (PostgreSQL, 2013)

PostgreSQL proporciona un gran número de características, entre ellas la de contar con soporte para lenguajes procedurales internos y embebidos. En el caso de una función o procedimiento definido en un lenguaje procedural, la base de datos no tiene un conocimiento implícito sobre cómo interpretar el código fuente de las funciones. El manejador en sí es una función de un lenguaje de programación compilada en forma de objeto compartido, y cargado cuando es necesario. (PostgreSQL, 2013)

Otra de las características que proporciona el gestor PostgreSQL es la de poder crear en él extensiones. Una extensión incluye típicamente múltiples objetos de SQL, por ejemplo, un nuevo tipo de datos, nuevos operadores y funcionalidades que no aparecen de forma nativa en este. Es útil recoger todos estos objetos en un solo paquete, simplificando la gestión de bases de datos. Para definir una extensión se necesitan, al menos, un archivo de comandos que contiene los comandos SQL para crear los objetos de la extensión y, un archivo de control que especifica algunas propiedades básicas de la propia extensión. En el sitio *pgxn.org* se gestionan dichas extensiones, existiendo disímiles tipos que van desde tipos de datos, funciones, diccionarios y otros. (PostgreSQL, 2013)

1.5.2. R

Programa estadístico y lenguaje de programación, uno de los más flexibles, potentes y profesionales que existen actualmente para el manejo de datos, gráficas y cálculo. En particular, está desarrollado y mantenido como un gran proyecto colaborativo de los más prestigiosos estadísticos actuales de diversos países y disciplinas. Cuenta, además, con la ventaja de ser multiplataforma, gratuito y de descarga e instalación sencilla. Permite leer datos en una gran variedad de formatos estándares, implementa algoritmos modernos y robustos, además, un número importante de paquetes están continuamente siendo desarrollados y puestos a disposición en Internet para su instalación, lo que convierte a R en un sistema integrado de herramientas para el análisis de datos. (R, 2013)

El lenguaje R dispone de varias funciones preparadas para la representación gráfica de datos. Estas funciones se dividen en dos grandes grupos: (R, 2013)

- Gráficos de alto nivel: crean un nuevo gráfico en la ventana de gráficos.
- Gráficos de bajo nivel: permiten añadir líneas, puntos y etiquetas a un gráfico ya existente.

La calidad visual de un gráfico en R es superior a la de muchos paquetes estadísticos, permite que estos se puedan personalizar y exportar en una amplia gama de formato, las salidas de procesos que ofrece son concisas y dejan al usuario la opción de solicitar un mayor nivel de detalle, por otra parte, la transparencia en la construcción de R permite un mayor control del proceso de generación de conocimiento por parte de los usuarios; estos aspectos hacen de R un excelente programa estadístico. (R, 2013)

1.5.3. PL/R

PL/R es un lenguaje procedural catalogado de desconfianza de PostgreSQL. Admite escribir funciones en PostgreSQL y funciones de agregado en el lenguaje estadístico R, permitiendo agregar funciones que no existen de forma nativa en PostgreSQL. Posibilita el acceso a bases de datos PostgreSQL desde el mismo sin necesidad de un conector especial, pues utiliza las SPI (*Server Programming Interface*) que provee el gestor escritas en C. (Conway, 2012)

Como es catalogado un lenguaje de desconfianza se debe tener cuidado con su utilización, pues se pueden acceder a recursos del servidor fuera de la base de datos. Manejando esto con sutileza puede ser una herramienta poderosa pues permite utilizar todas las funciones que provee R para realizar análisis estadísticos y graficación. Posee compatibilidades con los tipos de datos PostgreSQL. Los requisitos básicos para comenzar a trabajar con PL/R son, tener instalado PostgreSQL, tener PL/R instalado, y crear una base de datos en PostgreSQL donde se añade el nuevo lenguaje procedural. (Conway, 2012)

1.5.4. Python

Python es un potente lenguaje de programación. Cuenta con estructuras eficientes de datos de alto nivel y un enfoque simple pero eficaz a la programación orientada a objetos. La elegante sintaxis de *Python* y el tipado dinámico, junto con su naturaleza interpretada, lo convierten en un lenguaje ideal para scripting y desarrollo rápido de aplicaciones en muchas áreas de la mayoría de las plataformas. (Python, 2013)

El intérprete de *Python* y la extensa biblioteca estándar están disponibles gratuitamente en formato fuente o binario para todas las plataformas y puede ser distribuido libremente. (Python, 2013)

Conclusiones del capítulo

En el capítulo fueron abordados temas como el almacenamiento y presentación de la información, se realizó un estudio de herramientas y técnicas empleadas para ello, como son las bases de datos, los

sistemas gestores de bases de datos y los gráficos estadísticos, se realizó también una caracterización de la extensión *pgr-graphic*, la cual brinda un grupo escaso de opciones gráficas para la graficación desde el gestor PostgreSQL, evidenciándose la necesidad de realizar una nueva implementación que supla sus deficiencias. Para ello, se definió como metodología de desarrollo de software XP. Se emplearán además PostgreSQL 9.1 como gestor de base de datos, el paquete estadístico R 2.14 como motor gráfico, los lenguajes PL/R 8.3.0.13 para agregar funciones que no existen de forma nativa en PostgreSQL y, *Python* 2.4 para almacenar los gráficos generados en la base de datos; lo que permitirá generar gráficos desde el gestor tomando como motor de gráficos a R.

CAPÍTULO 2

CARACTERÍSTICAS Y DISEÑO DE LA SOLUCIÓN

El diseño de un producto o sistema es fundamental para un desarrollo exitoso del mismo, en este se desarrollan, revisan y documentan los requisitos del producto y los detalles procedimentales de su implementación. En este capítulo se presenta el diseño del producto a desarrollar, como parte de este, se presenta un modelo de dominio con el fin de proporcionar un mejor entendimiento de los principales conceptos del negocio, una descripción de las historias de usuario definidas, los requisitos tanto funcionales como no funcionales a tener en cuenta en su desarrollo, agrupados en la lista de reserva del producto, las tarjetas CRC, así como un plan de iteraciones que contempla el tiempo de implementación del sistema.

2.1. Modelo de dominio

A pesar de que la metodología de desarrollo de software XP no precisa una técnica específica para definir el negocio, con el fin de proporcionar un mejor entendimiento de los principales conceptos que se manejan en este, se decide realizar un modelo de dominio, el cual se describe mediante diagramas de UML, específicamente mediante diagramas de clases.

Según Gloria Lucía Gómez Giraldo, el modelo de dominio es "... la representación visual de los conceptos u objetos del mundo real en un dominio de interés... agrupa los conceptos de un dominio. Es el mecanismo fundamental para comprender el dominio del problema y para establecer conceptos comunes". (Giraldo Gómez, 2010)

El modelo de dominio tiene como objetivo principal ayudar a comprender los conceptos con los que se trabajan y utilizan los usuarios y, con los que se deberá trabajar la aplicación.

El modelo de dominio que será tomado como punto de partida para el diseño del sistema se muestra en la figura siguiente.

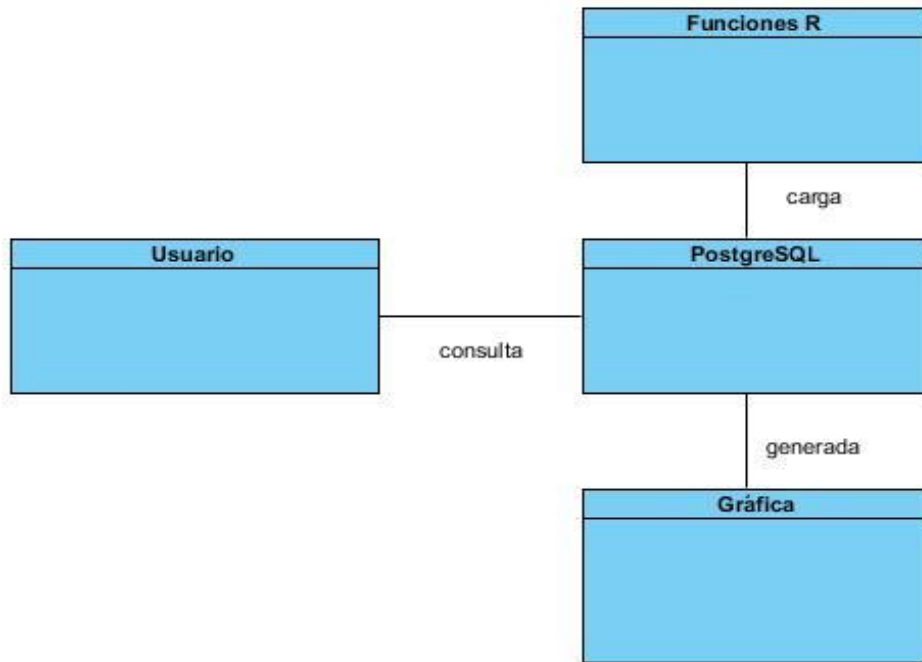


Figura 4: Modelo de dominio

Tabla 7: Descripción de los conceptos del diagrama de dominio

No	Concepto	Descripción
1	Usuario	Persona que interactúa con el sistema de gestión de bases de datos
2	PostgreSQL	Servidor de bases de datos
3	Funciones R	Funciones implementadas en el paquete estadístico R y que son cargadas desde el gestor para ejecutar consultas
4	Gráfica	Resultado obtenido luego de ejecutar las consultas realizadas

2.2. Descripción del sistema propuesto

Se propone la implementación de una extensión de graficación para el gestor PostgreSQL que permitirá a los usuarios, a través de consultas, representar la información contenida en las bases de datos mediante gráficas de barras simples, agrupadas, apiladas, histogramas, líneas, circulares, de caja y de dispersión. Las gráficas serán generadas como imagen y almacenadas en el directorio de trabajo del motor gráfico R y, a petición del usuario, podrán ser almacenadas en la base de datos.

2.2.1. Historias de Usuario

El primer paso de cualquier proyecto que siga la metodología XP es definir las historias de usuario (HU) con el cliente. Las historias de usuario “son descripciones cortas y escritas en el lenguaje del usuario sin terminología técnica, que proporcionan los detalles sobre la estimación del riesgo y cuánto tiempo tardará su implementación”. Estas: (Mesa Reyes, y otros, 2011)

Constan de 3 o 4 líneas escritas por el cliente sin profundizar en detalles técnicos; sin describir posibles algoritmos para su implementación.

- Son usadas para estimar tiempos de desarrollo de la aplicación que describen.
- Son usadas en la fase de pruebas, para verificar si el programa cumple con lo que especifica la historia de usuario.
- Para implementarlas se requiere de encuentros entre cliente y desarrolladores, para concretar y detallar lo que tienen que hacer.
- El tiempo de desarrollo ideal de cada historia es entre 1 y 3 semanas.

Para definir las historias de usuario se utiliza una planilla, que contiene todos los datos necesarios para desarrollar la funcionalidad descrita.

En las tablas siguientes se detallan las 6 historias de usuario definidas para el desarrollo de la propuesta de solución de la investigación.

Tabla 8: Historia de usuario Generar gráfico de barras

Historia de usuario

Número: 1	Nombre: Generar gráfico de barras	
Cantidad de modificaciones: 0		
Usuario: Dalilys Martínez Gutiérrez	Iteración asignada: 1	
Prioridad en negocio: Muy Alta	Puntos estimados: 3	
Riesgo en desarrollo: Alto	Puntos reales: 3	
Descripción: Permite al usuario mediante consultas SQL generar a partir de los datos contenidos en una base de datos, gráficos de barras simples, agrupadas, apiladas e histogramas		
Observaciones: -		
Prototipo de interfaz:		

Tabla 9: Historia de usuario Generar gráfico de líneas

Historia de usuario		
Número: 2	Nombre: Generar gráfico de líneas	
Cantidad de modificaciones: 0		
Usuario: Dalilys Martínez Gutiérrez	Iteración asignada: 1	
Prioridad en negocio: Muy Alta	Puntos estimados: 1	

Riesgo en desarrollo: Alto	Puntos reales: 1
Descripción: Permite al usuario, mediante consultas SQL, generar a partir de los datos contenidos en una base de datos gráficos de líneas	
Observaciones: -	
Prototipo de interfaz:	

Tabla 10: Historia de usuario Generar gráfico de circular

Historia de usuario	
Número: 3	Nombre: Generar gráfico circular
Cantidad de modificaciones: 0	
Usuario: Dalilys Martínez Gutiérrez	Iteración asignada: 1
Prioridad en negocio: Muy Alta	Puntos estimados: 1
Riesgo en desarrollo: Alto	Puntos reales: 1
Descripción: Permite al usuario, mediante consultas SQL, generar a partir de los datos contenidos en una base de datos gráficos circulares	
Observaciones: -	
Prototipo de interfaz:	

Tabla 11: Historia de usuario Generar gráfico de dispersión

Historia de usuario	
Número: 4	Nombre: Generar gráfico de dispersión
Cantidad de modificaciones: 0	
Usuario: Dalilys Martínez Gutiérrez	Iteración asignada: 1
Prioridad en negocio: Muy Alta	Puntos estimados: 1
Riesgo en desarrollo: Alto	Puntos reales: 1
Descripción: Permite al usuario, mediante consultas SQL, generar a partir de los datos contenidos en una base de datos gráficos de dispersión	
Observaciones: -	
Prototipo de interfaz:	

Tabla 12: Historia de usuario Generar gráfico de caja

Historia de usuario	
Número: 5	Nombre: Generar gráfico de caja
Cantidad de modificaciones: 0	
Usuario: Dalilys Martínez Gutiérrez	Iteración asignada: 1

Prioridad en negocio: Muy Alta	Puntos estimados: 1
Riesgo en desarrollo: Alto	Puntos reales: 1
Descripción: Permite al usuario, mediante consultas SQL, generar a partir de los datos contenidos en una base de datos gráficos de caja	
Observaciones: -	
Prototipo de interfaz:	

Tabla 13: Historia de usuario Salvar gráficos en la base de datos

Historia de usuario	
Número: 6	Nombre: Salvar gráficos en la base de datos
Cantidad de modificaciones: 0	
Usuario: Dalilys Martínez Gutiérrez	Iteración asignada: 2
Prioridad en negocio: Alta	Puntos estimados: 3
Riesgo en desarrollo: Alto	Puntos reales: 3
Descripción: Los gráficos generados son insertados en una tabla de la base datos	
Observaciones: -	

Prototipo de interfaz:

2.2.2. Lista de reserva del producto

La etapa de definición de requisitos es una tarea de suma importancia a la hora de desarrollar un sistema. Esta consiste en generar una definición clara y precisa de los aspectos más relevantes del producto. La lista de reserva del producto agrupa los requisitos funcionales organizados por prioridad según la complejidad en el negocio y los requisitos no funcionales con una breve descripción de cada uno de ellos.

Según *Ivar Jacobson, James Rumbaugh y Grady Booch*, precursores de la metodología de desarrollo RUP, “...los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir”. (Jacobson, y otros, 2000)

Estos mismos autores definieron los requisitos no funcionales como aquellos que “especifican propiedades del sistema, como restricciones del entorno o de la implementación, rendimiento, facilidad de mantenimiento, extensibilidad y fiabilidad”. (Jacobson, y otros, 2000)

Cuanto más alto sea el grado de cumplimiento y aceptación por el cliente de los requisitos, más alta será la calidad del sistema desarrollado.

En la tabla siguiente se muestran los requisitos funcionales y no funcionales definidos para su implementación en la propuesta de solución, derivados de las historias de usuario acordadas con el cliente.

Tabla 14: Lista de reserva del producto

Ítem	Descripción	Estimación	Estimado por
Requisitos funcionales			
Prioridad: Muy Alta			
1	Generar gráficos de barras simples	0.5	Analista

2	Generar gráficos de barras agrupadas	1	Analista
3	Generar gráficos de barras apiladas	1	Analista
4	Generar gráficos de histogramas	0.5	Analista
5	Generar gráficos de líneas	1	Analista
6	Generar gráficos circulares	1	Analista
7	Generar gráficos de caja	1	Analista
8	Generar gráficos de dispersión	1	Analista
Prioridad: Alta			
9	Salvar imágenes en la base de datos	3	Analista
Requisitos no funcionales			
Software			

10	<p>Sistema Operativo: GNU/Linux preferentemente Ubuntu GNU/Linux 12.4</p> <p>Se requiere tener instalado:</p> <ul style="list-style-type: none"> - postgresql-9.1 o superior - r-base - postgresql-9.1-plr - postgresql-server-dev-9.1 - python-psycpg2 <p>Los nombres de los paquetes coinciden con los pertenecientes a <i>Ubuntu</i> 12.4, estos pueden variar según el sistema operativo sobre el que se trabaje y la versión de PostgreSQL que sea instalada</p>		
Hardware			
11	<p>El ordenador debe tener como mínimo:</p> <ul style="list-style-type: none"> - Microprocesador con una velocidad de 500 Mhz - Capacidad libre en disco duro de 5 GB - 512 MB de RAM 		

2.2.3. Plan de iteraciones

Después de tener definidas las historias de usuario, se hace necesario crear un plan de iteraciones, (en inglés *release planning*), un plan de iteraciones es una planificación donde los desarrolladores y clientes establecen los tiempos de implementación ideales de las historias de usuario, la prioridad de estas y cuáles serán implementadas en cada versión del programa. Al comienzo de cada iteración los clientes

deben seleccionar las historias de usuario definidas en el plan de iteración, que serán implementadas. (Beck, 2000)

La implementación del sistema propuesto tendrá una duración de 10 semanas, tiempo en que se realizarán dos iteraciones. La tabla que se muestra a continuación presenta el plan propuesto.

Tabla 15: Plan de iteraciones

Iteración	Descripción de la iteración	Orden de la HU a implementar	Duración total
1	En esta iteración se implementarán las historias de usuario que tengan la prioridad en el negocio MUY ALTA	1, 2, 3, 4, 5	7 semanas
2	En esta iteración se implementarán las historias de usuario que tengan la prioridad en el negocio ALTA	6	3 semanas

2.3. Modelo de diseño

Para el diseño de aplicaciones informáticas, la metodología XP no requiere la presentación del sistema mediante diagramas de clases utilizando notación UML. En su lugar se usan otras técnicas como las tarjetas CRC (del inglés *Class-Responsibility-Collaboration*).

2.3.1. Tarjetas CRC

El uso de las tarjetas CRC permite al programador centrarse en el desarrollo orientado a objetos, rompiendo con la clásica programación procedural.

Para la implementación de las 6 historias de usuario definidas para dar cumplimiento a los requisitos del cliente, se identificaron 9 tarjetas CRC:

- Generar gráfico de barras simples.
- Generar gráfico de barras agrupadas.

- Generar gráfico de barras apiladas.
- Generar gráfico de histograma.
- Generar gráfico de líneas.
- Generar gráfico circular.
- Generar gráfico de caja.
- Generar gráficos de dispersión.
- Salvar imágenes en la base de datos.

Las tarjetas que se muestran en las tablas siguientes representan objetos; la clase a la que pertenece cada objeto aparece escrita en la parte superior de la tarjeta, en la columna a la izquierda se muestran las responsabilidades u objetivos que debe cumplir el objeto y a la derecha, las clases que colaboran.

En todas las tarjetas, las colaboraciones están compuestas por dos tipos de funciones gráficas que provee el paquete estadístico R; funciones de alto y bajo nivel. Las de alto nivel son funciones generales que construyen gráficos, y las de bajo nivel son funciones que permiten añadir elementos a los gráficos generados.

Tabla 16: Tarjeta CRC Generar gráfico de barras simples

Tarjeta CRC	
Clase: Generar gráfico de barras simples	
Responsabilidades	Colaboraciones
Generar gráfico de barras simples	barplot() legend() insertar()

Tabla 17: Tarjeta CRC Generar gráfico de barras agrupadas

Tarjeta CRC	
Clase: Generar gráfico de barras agrupadas	
Responsabilidades	Colaboraciones
Generar gráfico de barras agrupadas	barplot() legend() insertar()

Tabla 18: Tarjeta CRC Generar gráfico de barras apiladas

Tarjeta CRC	
Clase: Generar gráfico de barras apiladas	
Responsabilidades	Colaboraciones
Generar gráfico de barras apiladas	barplot() legend() insertar()

Tabla 19: Tarjeta CRC Generar gráfico de histograma

Tarjeta CRC	
Clase: Generar gráfico de histograma	
Responsabilidades	Colaboraciones
Generar gráfico de histograma	hist()

	legend() insertar()
--	------------------------

Tabla 20: Tarjeta CRC Generar gráfico de líneas

Tarjeta CRC	
Clase: Generar gráfico de líneas	
Responsabilidades	Colaboraciones
Generar gráfico de líneas	plot() legend() insertar()

Tabla 21: Tarjeta CRC Generar gráfico circular

Tarjeta CRC	
Clase: Generar gráfico circular	
Responsabilidades	Colaboraciones
Generar gráfico circular	pie() legend() insertar()

Tabla 22: Tarjeta CRC Generar gráfico de caja

Tarjeta CRC	
Clase: Generar gráfico de caja	

Responsabilidades	Colaboraciones
Generar gráfico de caja	boxplot() legend() insertar()

Tabla 23: Tarjeta CRC Generar gráfico de dispersión

Tarjeta CRC	
Clase: Generar gráfico de dispersión	
Responsabilidades	Colaboraciones
Generar gráfico de dispersión	dotchar() legend() insertar()

Tabla 24: Tarjeta CRC Salvar imágenes en la base de datos

Tarjeta CRC	
Clase: Salvar imágenes en la base de datos	
Responsabilidades	Colaboraciones

Insertar imágenes en la base de datos	barras_simples() barras_agrupadas() barras_apiladas() histogramas() pie() dispersión() líneas() cajas()
---------------------------------------	--

Conclusiones del capítulo

En este capítulo se plasmaron los artefactos generados con el empleo de la metodología XP, que contribuirán al éxito del desarrollo de la nueva versión de la extensión *pgr-graphic*. Se confeccionó un modelo de dominio para comprender el funcionamiento del negocio. Fueron identificadas 6 historias de usuario desglosadas en 9 requisitos funcionales agrupados en la lista de reserva del producto. Se confeccionaron 9 tarjetas CRC para la implementación de las historias de usuario definidas. Se elaboró además un plan de iteraciones, con el que se concluye que la duración de la implementación de la extensión tendrá una duración de 10 semanas.

CAPÍTULO 3

IMPLEMENTACIÓN Y PRUEBAS DE LA SOLUCIÓN

La implementación y pruebas de un producto informático son fundamentales en su proceso de desarrollo. Como resultado de estas fases se obtiene un producto funcional para entregar a los clientes, con el que debe garantizarse el cumplimiento de los requisitos estipulados con la calidad requerida. En este capítulo se exponen las tareas de ingeniería definidas para cada historia de usuario con vistas a su implementación. Se presenta además el estándar de codificación utilizado, así como una descripción de las pruebas realizadas a la solución y los resultados de su ejecución.

3.1. Tareas de ingeniería

Entre los artefactos que se generan durante la fase de implementación en el desarrollo de software con la metodología XP se encuentran las tareas de ingeniería. Estas surgen de la división de las historias de usuario en tareas más sencillas, para planificar el trabajo desde un punto de vista técnico. Las tablas siguientes muestran las definidas para la historia de usuario Generar gráfico de barras, el resto se pueden consultar en el expediente de proyecto.

Tabla 25: Tarea de ingeniería Generar gráfico de barras simples

Tarea de ingeniería	
Número tarea: 1	Número de historia de usuario: 1
Nombre tarea: Generar gráfico de barras simples	
Tipo de tarea : Desarrollo	Puntos estimados: 1
Fecha inicio: 04/02/2013	Fecha fin: 08/02/2013
Programador responsable: Dalilys Martínez Gutiérrez	

Descripción: El sistema debe permitir al usuario mediante una consulta, generar un gráfico de barras simples a partir de datos contenidos en una base de datos, el usuario debe especificar los datos a graficar en los parámetros de la consulta ejecutada.

Tabla 26: Tarea de ingeniería Generar gráfico de barras agrupadas

Tarea de ingeniería	
Número tarea: 2	Número de historia de usuario: 1
Nombre tarea: Generar gráfico de barras agrupadas	
Tipo de tarea : Desarrollo	Puntos estimados: 1
Fecha inicio: 11/02/2013	Fecha fin: 15/02/2013
Programador responsable: Dalilys Martínez Gutiérrez	
Descripción: El sistema debe permitir al usuario mediante una consulta, generar un gráfico de barras agrupadas a partir de datos contenidos en una base de datos, el usuario debe especificar los datos a graficar en los parámetros de la consulta ejecutada.	

Tabla 27: Tarea de ingeniería Generar gráfico de histograma

Tarea de ingeniería	
Número tarea: 3	Número de historia de usuario: 1
Nombre tarea: Generar gráfico de histograma	
Tipo de tarea : Desarrollo	Puntos estimados: 1
Fecha inicio: 18/02/2013	Fecha fin: 22/02/2013

Programador responsable: Dalilys Martínez Gutiérrez
Descripción: El sistema debe permitir al usuario mediante una consulta, generar un histograma a partir de datos contenidos en una base de datos, el usuario debe especificar los datos a graficar en los parámetros de la consulta ejecutada.

Tabla 28: Tarea de ingeniería Generar gráfico de barras apiladas

Tarea de ingeniería	
Número tarea: 4	Número de historia de usuario: 1
Nombre tarea: Generar gráfico de barras apiladas	
Tipo de tarea : Desarrollo	Puntos estimados: 1
Fecha inicio: 04/03/2013	Fecha fin: 08/03/2013
Programador responsable: Dalilys Martínez Gutiérrez	
Descripción: El sistema debe permitir al usuario mediante una consulta, generar un gráfico de barras apiladas a partir de datos contenidos en una base de datos, el usuario debe especificar los datos a graficar en los parámetros de la consulta ejecutada.	

3.2. Estándar de codificación

Una de las buenas prácticas de la metodología XP define el empleo de los estándares de programación. De esta manera se logra obtener un código uniforme, como si el sistema fuera programado por una sola persona. Al programar en un lenguaje específico, se deben seguir reglas que permitan que cualquier persona que se desempeñe como codificador de dicho lenguaje pueda interpretar de manera eficiente la escritura del código. Estas reglas o convenciones de codificación son importantes para los programadores por varias razones. (Beck, 2000)

- El 80% del costo del tiempo de vida de una pieza de software es dedicado al mantenimiento.

- Casi nunca ningún software es mantenido durante toda su vida por su autor original.
- Las convenciones de código mejoran la lectura del software, permitiendo a los ingenieros entender mejor el nuevo código.

La metodología XP promueve la programación basada en estándares, de manera que sea fácilmente entendible por todo el equipo y que facilite la recodificación. A continuación se presenta cada punto a cumplir del estándar utilizado y un ejemplo de su aplicación.

3.2.1. Indentación

La indentación del código permite que este sea más legible al ganar en organización. Se deben garantizar los siguientes elementos:

- Longitud de la línea: líneas de menos de 80 caracteres.
- Rompiendo líneas: cuando una expresión tenga más de 80 caracteres, romperla de acuerdo con los siguientes principios:
 - Romper después de una coma.
 - Alinear la nueva línea con el comienzo de la expresión al mismo nivel de la línea anterior.

La siguiente figura muestra un ejemplo de ruptura de línea correcta.

```
barplot(as.matrix(resultado), xlim=NULL, ylim=yrange, beside=TRUE, main=nombre,  
        ylab=etiquetay, xlab=etiquetax, col=rainbow(length(as.matrix(resultado)))  
        names.arg=as.matrix(resultado))
```

Figura 5: Ejemplo correcto de ruptura de línea

3.2.2. Comentarios

Es conveniente comentar información que pueda ser leída tiempo después por desarrolladores (incluido el mismo que programó), que necesitan entender qué fue lo que se hizo en el fragmento de código. Los comentarios deben ser escritos de forma clara en una sola línea. La figura siguiente muestra un ejemplo.

esta función genera un gráfico de barras simples\$

```
CREATE OR REPLACE FUNCTION Pgr_Graphic2.Barras_Simples(nombre text, etiquetay text, etiquetax text
consulta text, guardar character)
```

Figura 6: Ejemplo correcto de comentario

3.2.3. Declaraciones

Las declaraciones realizadas deben cumplir con las siguientes normas:

- Poner una declaración por línea, ya que facilita los comentarios.
- Inicializar las variables locales donde sean declaradas.

```
yrange<-range (0,resultado)
resultado<-pg.spi.exec(consulta)
```

Figura 7: Ejemplo correcto de declaración

- Garantizar en las declaraciones de funciones que:
 - No existan espacios en blanco entre el nombre de la función y el paréntesis que abre su lista de parámetros.
 - El cuerpo de la función se delimite con el comentario \$BODY\$.
 - Las llamadas a funciones se separen con una línea en blanco.

La siguiente figura muestra un ejemplo de declaración de una función.

```
CREATE OR REPLACE FUNCTION Pgr_Graphic.Barras_Simples(nombre text, etiquetay text, etiquetax
consulta text, guardar character)
RETURNS text AS
$BODY$
    barplot()
    legend()
$BODY$
LANGUAGE plr VOLATILE
```

Figura 8: Ejemplo correcto de declaración de función

3.2.4. Sentencias

Las sentencias realizadas deben cumplir con las siguientes normas:

- Cada línea debe contener solo una sentencia simple, como muestra la figura 9.
- En las sentencias WHILE debe haber un espacio en blanco entre la palabra reservada WHILE y el paréntesis de la condición, así como entre el paréntesis que cierra dicha condición y las llaves de la sentencia. Las sentencias a ejecutar dentro de esta deben aparecer en la línea siguiente, cumpliendo las reglas de indentación según las normas definidas en el epígrafe 3.2.1, como muestra la figura 10.
- En las sentencias IF-ELSE debe haber siempre un espacio en blanco entre la palabra reservada IF y el paréntesis de la condición, entre el paréntesis que cierra dicha condición y las llaves de la sentencia, así como entre la palabra reservada ELSE y las llaves que le anteceden y prosiguen. Las sentencias a ejecutar deben aparecer en la línea siguiente, cumpliendo las reglas de indentación según las normas definidas en el epígrafe 3.2.1, como muestra la figura 11.

```
par (xpd = TRUE, mar = c (8.5,4,4,4))  
resultado<-pg.spi.exec (consulta)
```

Figura 9: Ejemplo de sentencia simple

```
while (condicion) {  
    sentencias  
}
```

Figura 10: Ejemplo correcto de sentencia WHILE

```

if (condición) {
    sentencias
}
if (condición) {
    sentencias
} else {
    sentencias
}
if (condicion) {
    sentencia
} else if (condicion) {
    sentencia
} else{
    sentencia
}

```

Figura 11: Ejemplo correcto de sentencias IF, IF-ELSE, IF ELSE-IF ELSE

3.2.5. Espacios en blanco

Las líneas en blanco mejoran la facilidad de lectura separando secciones de código que están lógicamente relacionadas. Se debe usar siempre una línea en blanco en las siguientes circunstancias:

- Entre funciones.
- Entre las distintas secciones lógicas de una función para facilitar la lectura.

La figura siguiente muestra un ejemplo de empleo de líneas en blanco entre las secciones de la función.

```

CREATE OR REPLACE FUNCTION Pgr_Graphic.Barras_Simples()
    RETURNS text AS
    $BODY$
        barplot()
        legend()
    $BODY$
LANGUAGE plr VOLATILE

```

Figura 12: Ejemplo correcto de aplicación de espacios en blanco

Se deben usar espacios en blanco en las siguientes circunstancias:

- Una palabra clave del lenguaje seguida por un paréntesis debe separarse por un espacio.
- Debe aparecer un espacio en blanco después de cada coma en las listas de argumentos.

La figura siguiente muestra un ejemplo de aplicación de espacios en blanco.

```
png (paste (nombre, "png", sep="."))
par (xpd = TRUE, mar = c (8.5,4,4,4))
```

Figura 13: Ejemplo correcto de aplicación de espacios en blanco

3.2.6. Convenciones de nombres

Para la implementación deben emplearse nombres que den idea de lo que realiza determinada función o lo que almacena determinada variable u objeto, con el fin de que el código sea legible. Se deben cumplir con las siguientes normas para los nombres:

- Los nombres de las funciones deben ser sugerentes, cuando son compuestos tendrán la primera letra de cada palabra que lo forma en mayúsculas y se utilizarán guiones bajos (_) para separarlas. Se debe intentar mantener los nombres de las clases simples y descriptivas, usar palabras completas, así como evitar acrónimos y abreviaturas. La figura siguiente muestra ejemplos.

```
CREATE OR REPLACE FUNCTION Pgr_Graphic.Barras_Simples()
CREATE OR REPLACE FUNCTION Pgr_Graphic.Barras_Apiladas()
```

Figura 14: Ejemplo correcto de convención de funciones

- Los nombres de los objetos o variables deben ser cortos pero con significado. Se distingue entre letras mayúsculas y minúsculas, de tal manera que x y X se hacen alusión a objetos diferentes. La figura siguiente muestra ejemplos.

```
yrange<-range (0,resultado)
Yrange<-length ((as.matrix(resultado)))/2
```

Figura 15: Ejemplo correcto de convención de nombres de objetos o variables

3.3. Implementación de las historias de usuario

Como resultado de las tareas de ingeniería derivadas de las historias de usuario definidas se obtuvo la extensión *pgr-graphic* en su versión 2.0 para PostgreSQL 9.1. La extensión está compuesta por 9 funciones, las que permiten generar variedad de gráficos y una tabla en la cual serán almacenadas las imágenes generadas, *pgr-graphic* está estructurada como muestra la figura 16. Para su instalación y uso se debe consultar el manual de usuario.

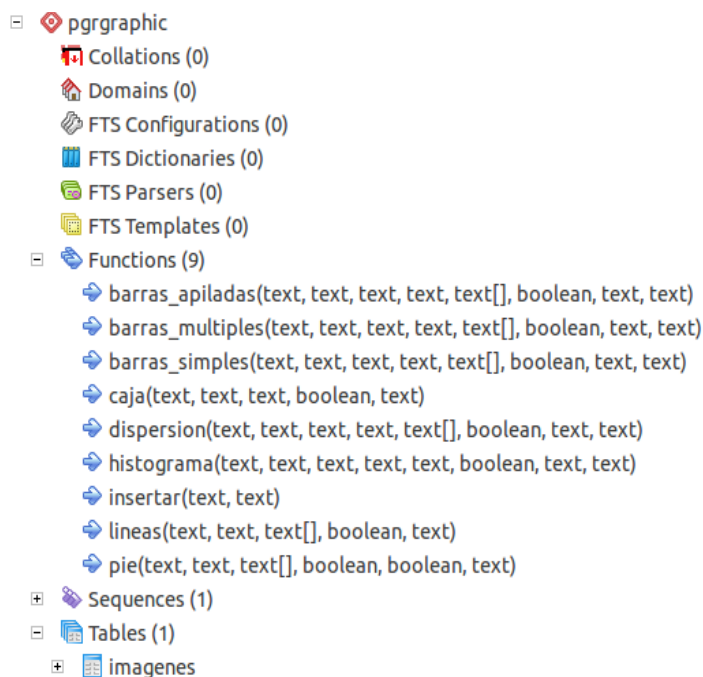


Figura 16: Estructura de *pgr-graphic* v2.0

3.4. Pruebas

El uso de cualquier producto de software tiene que estar justificado por las ventajas que ofrece. Sin embargo, antes de empezar a usarlo es muy difícil determinar si sus ventajas realmente justifican su empleo. El único modo adecuado para determinar el estado de la calidad de un producto es el proceso de pruebas. El mejor instrumento para esta determinación es la llamada prueba de aceptación.

3.4.1. Pruebas de aceptación

En esta prueba se evalúa el grado de calidad del producto con relación a todos los aspectos relevantes para que el uso de este se justifique. Estas pruebas las realiza el cliente. Son básicamente pruebas funcionales sobre el sistema completo y, buscan una cobertura de la especificación de requisitos y del manual de usuario. (Pressman, 2005)

Para determinar la calidad de la extensión desarrollada se aplicarán pruebas de caja negra.

Pruebas de caja negra

Las pruebas de caja negra, también denominadas pruebas de comportamiento, se llevan a cabo sobre la interfaz del producto, obviando el comportamiento interno y la estructura del programa. Al aplicar pruebas de caja negra se derivan un conjunto de casos de pruebas que involucran condiciones de entrada que utilizan todos los requisitos funcionales de un programa. Los casos de prueba de la caja negra pretenden demostrar que: (Pressman, 2005)

- Las funciones del software son operativas.
- La entrada se acepta de forma correcta.
- Se produce una salida correcta.
- La integridad de la información externa se mantiene.

Las pruebas de caja negra pretenden encontrar los tipos de errores siguientes: (Pressman, 2005)

- Funciones incorrecta o ausentes.
- Errores en la interfaz.
- Errores en estructuras de datos o en accesos a bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y de terminación.

Para confeccionar los casos de prueba de caja negra existen distintos criterios, el empleado en este proceso de prueba será el de partición de equivalencia.

Partición de equivalencia

La partición de equivalencia es un método de pruebas de caja negra que divide el campo de entrada de un programa en clases de datos, de los que se pueden derivar casos de prueba. La partición equivalente se dirige a una definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar.

Este método intenta dividir el dominio de entrada de un programa en un número finito de clases de equivalencia. De tal modo que se pueda asumir razonablemente que una prueba realizada con un valor representativo de cada clase es equivalente a una prueba realizada con cualquier otro valor de dicha clase. Esto quiere decir que si el caso de prueba correspondiente a una clase de equivalencia detecta un error, el resto de los casos de prueba de dicha clase de equivalencia deben detectar el mismo error. Y viceversa, si un caso de prueba no ha detectado ningún error, es de esperar que ninguno de los casos de prueba correspondientes a la misma clase de equivalencia encuentre ningún error. (Pressman, 2005)

El objetivo principal de la fase de pruebas es encontrar la mayor cantidad de errores y defectos posibles, por lo que es conveniente que las pruebas de aceptación sean realizadas por el propio usuario final y no por el desarrollador del producto. Con ese fin serán aplicadas a la extensión desarrollada pruebas alfa.

Prueba alfa

Se lleva a cabo, por un cliente en el lugar de desarrollo. Se usa el software de forma natural con el desarrollador como observador del usuario. Las pruebas alfa se llevan a cabo en un entorno controlado. Para que tengan validez, se debe primero crear un ambiente con las mismas condiciones que se encontrarán en las instalaciones del cliente. Una vez logrado esto, se procede a realizar las pruebas y a documentar los resultados. (Pressman, 2005)

3.4.2. Diseño de casos de prueba y registro de no conformidades

Un caso de prueba se diseña según las funcionalidades descritas en las historias de usuario. Este diseño se elabora previamente a realizar las pruebas funcionales a la aplicación. Se parte de la descripción de las historias de usuario del sistema como apoyo para las revisiones. Cada planilla de caso de prueba recoge la especificación de una historia de usuario, dividido en secciones y escenarios, detallando las funcionalidades descritas en estas y describiendo cada variable que recoge la historia de usuario en

cuestión, además, quedan plasmadas las revisiones realizadas al caso de prueba, así como un registro de todo aquello que no se corresponde con la calidad del producto.

En el proceso de pruebas en cuestión se hace uso de los casos de prueba de caja negra. Partiendo de la descripción de las historias de usuario del sistema, como apoyo para las revisiones, se diseñó un caso de prueba asociado a cada historia de usuario (ver expediente de proyecto). Para detallar la historia de usuario se utiliza una tabla, donde se desglosan estas en secciones y a su vez en escenarios para hacer más fructífera la ejecución de las pruebas. Esta tabla contiene los campos:

- Nombre de la sección: se especifica el nombre de la sección [SC 1: Nombre de la sección].
- Escenarios de la sección: se especifican los escenarios de cada sección [EC 1.1: Nombre del Escenario].
- Descripción de la funcionalidad: se describe brevemente la funcionalidad del escenario.

El siguiente es un ejemplo donde se detalla la historia de usuario Generar gráfico de barras.

Tabla 29: Secciones para probar la historia de usuario Generar gráfico de barras

Nombre de sección	Escenarios de la sección	Descripción de la funcionalidad
SC1: Generar gráfico de barras simples	EC1.1 Generar gráfico de barras simples con datos correctos	El usuario realiza una consulta que recibe por parámetros los datos necesarios para generar un gráfico de barras simples. El sistema genera el gráfico y muestra un mensaje indicando que el gráfico ha sido generado
	EC1.2 Generar gráfico de barras simples con datos incorrectos	El usuario realiza una consulta que recibe por parámetros los datos de forma incorrecta para generar un gráfico de barras simples. El sistema muestra un mensaje indicando que ha ocurrido un error
SC2: Generar gráfico	EC2.1 Generar gráfico	El usuario realiza una consulta que recibe por

de barras agrupadas	de barras agrupadas con datos correctos	parámetros los datos necesarios para generar un gráfico de barras agrupadas. El sistema genera el gráfico y muestra un mensaje indicando que el gráfico ha sido generado
	EC2.2 Generar gráfico de barras agrupadas con datos incorrectos	El usuario realiza una consulta que recibe por parámetros los datos de forma incorrecta para generar un gráfico de barras agrupadas. El sistema muestra un mensaje indicando que ha ocurrido un error
SC3: Generar gráfico de barras apiladas	EC3.1 Generar gráfico de barras apiladas con datos correctos	El usuario realiza una consulta que recibe por parámetros los datos necesarios para generar un gráfico de barras apiladas. El sistema genera el gráfico y muestra un mensaje indicando que el gráfico ha sido generado
	EC3.2 Generar gráfico de barras apiladas con datos incorrectos	El usuario realiza una consulta que recibe por parámetros los datos de forma incorrecta para generar un gráfico de barras apiladas. El sistema muestra un mensaje indicando que ha ocurrido un error
SC4: Generar histogramas	EC4.1 Generar histogramas con datos correctos	El usuario realiza una consulta que recibe por parámetros los datos necesarios para generar un histograma. El sistema genera el gráfico y muestra un mensaje indicando que el gráfico ha sido generado
	EC4.2 Generar histogramas con datos incorrectos	El usuario realiza una consulta que recibe por parámetros los datos de forma incorrecta para generar un histograma. El sistema muestra un mensaje indicando que ha ocurrido un error

A partir de esta descripción se detallan las variables asociadas a la historia de usuario. Esta descripción está compuesta por los campos:

- No: se enumeraron todos los campos o variables, descritos en el caso de prueba.
- Nombre de campo: se especificó el nombre del campo de entrada.
- Clasificación: se especificó la clasificación según el componente de diseño utilizado, ejemplo: texto, consulta o arreglo.
- Nulo: se especificó si el campo puede ser nulo o no, para ello solo se puso Sí o No.
- Descripción: se describieron brevemente los datos que debían introducirse.

En la siguiente tabla se muestra un ejemplo de la descripción de variables de la historia de usuario Generar gráficos de barras.

Tabla 30: Descripción de variables de la historia de usuario Generar gráficos de barras

No.	Nombre del campo	Clasificación	Nulo	Descripción
1	titulo_principal	texto	Sí	Indica el título principal de la gráfica
2	marquilla_ejex	texto	Sí	Indica el título para el eje x
3	marquilla_ejey	texto	Sí	Indica el título para el eje y
4	datos	consulta	No	Consulta con la cual se obtienen los datos que se desean graficar
5	texto_leyenda	arreglo	Sí	Texto que se colocará en la leyenda del gráfico
6	guardar	boolean	No	Indica si el usuario desea o no guardar la imagen generada en la base de datos

7	tipo	texto	No	Indica la preferencia del usuario al confeccionar la gráfica (barra,columna)
8	formato	texto	No	Indica el formato en que se desea generar y almacenar la gráfica. (png, jpeg)

Esta descripción permitió que se realizara una matriz de datos, donde se evaluó y probó la validez de cada uno de los datos introducidos en la extensión, específicamente en la sección que se estuvo probando. Utilizando un juego de datos válidos e inválidos se identificó el empleo de la técnica de partición de equivalencia. Esta matriz de datos contiene los siguientes aspectos:

- Id del escenario: se especifica el id del escenario [EC1, EC2,..., ECn]
- Escenario: se especifica el nombre del escenario.
- Variables [1, 2,..., n]: se especifica el nombre de la variable o el número, según la tabla de descripción de variables, y en su celda correspondiente se indicó el valor del dato [V (Válido), I (Inválido), N/A (No Aplica)].
- Respuesta del sistema: se escribe el resultado que se esperaba al realizar la prueba.
- Resultado de la prueba: se escribe el resultado que se obtuvo al realizar la prueba.
- Flujo central: se expone el flujo central de la historia de usuario al que se le estaba diseñando el caso de prueba.

Para la ejecución de los casos de prueba se empleó la base de datos de tamaño normal de *Dell Store 2*⁴, disponible en el proyecto “Colección de bases de datos de ejemplos para PostgreSQL”⁵, diseñada para la versión 8.1 o superior del gestor; la cual tiene la estructura mostrada en la tabla siguiente.

⁴ Disponible en <http://linux.dell.com/dvdstore/>

⁵ Disponible en <http://pgfoundry.org/projects/dbsamples/>

Tabla 31: Listado de relaciones

Esquema	Nombre	Tipo	Total de registros
<i>public</i>	<i>categories</i>	tabla	16
<i>public</i>	<i>cust_hist</i>	tabla	60350
<i>public</i>	<i>customers</i>	tabla	20000
<i>public</i>	<i>inventory</i>	tabla	10000
<i>public</i>	<i>orderlines</i>	tabla	60350
<i>public</i>	<i>orders</i>	tabla	12000
<i>public</i>	<i>products</i>	tabla	10000
<i>public</i>	<i>reorder</i>	tabla	0

Los resultados de las pruebas que no fueron satisfactorios pasaron a ser no conformidades y se plasmaron en el registro de defectos y dificultades detectados que se encuentra en la parte final de cada diseño de caso de prueba con los siguientes campos:

- Elemento: se especifica el nombre del elemento.
- No: se especifica el número de la no conformidad.
- No conformidad: se describe la no conformidad.
- Aspecto correspondiente: se especifica el aspecto correspondiente a la no conformidad.
- Etapa de detección: se especifica la etapa de detección del error.
- Clasificación: se pone una (S), en caso de que la no conformidad estuviera clasificada como significativa, (NS), en caso de que la no conformidad estuviera clasificada como no significativa y una (R), en caso de que la no conformidad solo fuera una recomendación

- Estado NC: se coloca el estado de la no conformidad y la fecha, cada vez que se revisa se deja el estado anterior y se coloca el nuevo con la fecha en que se revisó [RA: Resuelta, PD: Pendiente, NP: No Procede].
- Respuesta del equipo de desarrollo: se comienza a llenar a partir de la segunda iteración, y es responsabilidad del equipo de desarrollo, quien especifica si se ha resuelto o no la no conformidad y en caso de no proceder se explica la causa.

En la tabla 31 se muestra un ejemplo de la matriz de datos para la historia de usuario Generar gráficos de barras, de la SC 1 Generar gráfico de barras simples, con los juegos de datos tomados de la base de datos de prueba seleccionada. Para ganar en claridad en la tabla se emplean las siguientes notaciones para las consultas:

* **[SELECT COUNT (products.prod_id) AS cantidad FROM products JOIN categories**

ON categories.category=products.category

GROUP BY categories.categoryname, categories.category

ORDER BY categories.category]

** **[SELECT array_agg(categoryname::text) FROM categories]**

*** **[COUNT(products.prod_id) AS cantidad FROM products JOIN categories**

ON categories.category=products.category

GROUP BY categories.categoryname, categories.category

ORDER BY categories.category]

**** **[array_agg(categoryname::text) FROM categories]**

Tabla 32: Matriz de datos de la historia de usuario Generar gráficos de barras

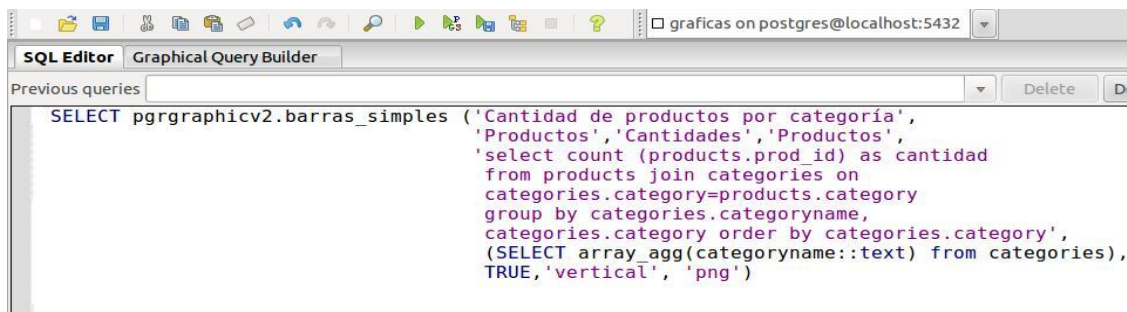
Id del escenario	Escenario	Variables (enumeradas según tabla 30)								Respuesta del sistema	Flujo central
		1	2	3	4	5	6	7	8		
EC1.1	Generar gráfico de barras simples insertando datos correctos	V	V	V	V	V	V	V	V	El sistema genera un gráfico de barras simples El sistema emite un mensaje mostrando que se ha generado la gráfica correctamente	1. El usuario realiza una consulta para generar un gráfico 2. El sistema genera el gráfico y emite un mensaje indicando que ha sido generado
		Cantidad de productos por categoría	Cantidades	Productos	*	**	true	barra	png		

EC1.2	Generar gráfico de barras simples insertando datos incorrectos	I	V	V	V	V	V	V	V	El sistema muestra un mensaje indicando un error	<p>1. El usuario realiza una consulta para generar el gráfico</p> <p>2. El sistema emite un mensaje indicando que ha ocurrido un error</p>
		Cantidad de productos por categoría	Cantidades	Productos	*	**	true	barra	png		
		V	I	V	V	V	V	V	V		
		Cantidad de productos por categoría	Cantidades	Productos	*	**	true	barra	png		
		V	V	I	V	V	V	V	V		
		Cantidad de productos por categoría	Cantidades	Productos	*	**	true	barra	png		
		V	V	V	I	V	V	V	V		
		Cantidad de productos	Cantidades	Productos	***	**	true	barra	png		

		por categoría								
		V	V	V	V	I	V	V	V	
		Cantidad de productos por categoría	Cantidades	Productos	*	****	true	barra	png	
		V	V	V	V	V	I	V	V	
		Cantidad de productos por categoría	Cantidades	Productos	*	**	falso	barra	png	
		V	V	V	V	V	V	I	V	
		Cantidad de productos por categoría	Cantidades	Productos	*	**	true	barras	png	
		V	V	V	V	V	V	V	I	

		Cantidad de productos por categoría	Cantidades	Productos	*	**	true	barra	bmp		
--	--	-------------------------------------	------------	-----------	---	----	------	-------	-----	--	--

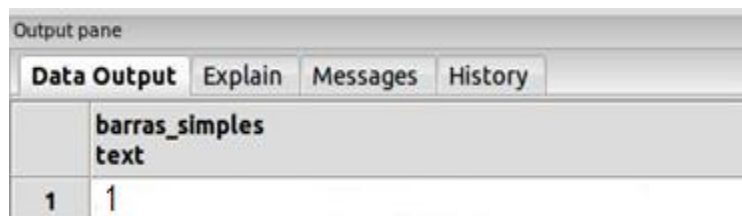
Como resultado de la ejecución del escenario EC1.1 mostrado en la tabla 29, se obtuvieron los resultados mostrados en las pantallas siguientes.



```
SELECT pgrgraphicv2.barras_simples ('Cantidad de productos por categoría',
'Productos','Cantidades','Productos',
'select count (products.prod_id) as cantidad
from products join categories on
categories.category=products.category
group by categories.categoryname,
categories.category order by categories.category',
(SELECT array_agg(categoryname::text) from categories),
TRUE,'vertical', 'png')
```

Figura 17: Consulta definida por el usuario ejecutada en el pgAdmin

La consulta que se realice para generar un gráfico en el *pgr-graphic* puede ejecutarse en cualquier cliente de administración del gestor, siempre que se conecte a la base de datos de la propuesta de solución. En el ejemplo mostrado en la figura 16 se empleó el *PgAdmin* por ser la herramienta definida para emplear en el desarrollo de la solución.



Output pane			
Data Output	Explain	Messages	History
	barras_simples text		
1	1		

Figura 18: Respuesta del sistema al ejecutar la consulta correctamente

Dependiendo del resultado de la ejecución de la consulta definida, el sistema muestra un pequeño mensaje indicando el mismo. Ver sección: Respuestas del sistema en el manual de usuario.

	id [PK] serial	nombre character varying(50)	imagen bytea
1	1	barrasimple.png	<binary data>
2	2	barrasmultiples.png	<binary data>
3	3	prueba.png	<binary data>
4	4	barrassimples.png	<binary data>
5	5	pie.png	<binary data>
6	6	simples.png	<binary data>
7	7	apiladas.png	<binary data>
8	8	Cantidad_de_productos_por_categoria.png	<binary data>
*			

Figura 19: Inserción de la imagen en la tabla de la base de datos

Cantidad de productos por categoría

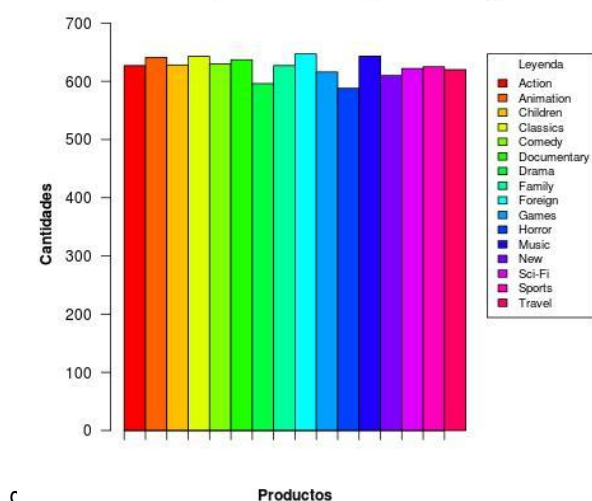


Figura 20: Gráfico generado como resultado de la consulta

Como resultado de la aplicación de las pruebas, se detectaron un conjunto de no conformidades. Se muestran en la tabla siguiente las asociadas a la historia de usuario Generar gráfico de barras, específicamente el escenario Generar gráfico de barras simples.

Tabla 33: No conformidades detectadas en la aplicación del caso de pruebas Generar gráfico de barras, específicamente el escenario Generar gráfico de barras simples

No.	NC	Aspecto asociado	Etapas de detección	Clasificación	Estado NC	Respuesta
1	Acepta en el campo titulo_principal datos de tipo NULL, generándose la gráfica sin título	Generar gráficos de barras simples	Pruebas	S	9/05/13 PD 9/05/13 RA	Corregido

La plantilla de No Conformidades recoge además los errores que son detectados durante la revisión de la documentación del sistema. Se elabora un documento por cada revisión que se haga y se controlan a través de versiones según se vayan eliminando los errores, hasta que finalmente se hayan erradicado todos los defectos que posea el elemento que se prueba. Además de estar plasmado en la planilla de diseño de casos de prueba, estas no conformidades se van registrando en un documento aparte para luego enviarlo al equipo de desarrolladores. De igual manera, aun trabajando paralelamente con los casos de prueba, el documento emitido a los desarrolladores es independiente, porque los casos de prueba son para consumo y único uso de los probadores.

3.4.3 Resultados de las pruebas de aceptación

La presentación de los resultados de las pruebas de aceptación es importante, *Beck* y *Kent* recomiendan la exhibición de los resultados que se obtienen al ejecutar las pruebas de aceptación, generando reportes y gráficas que desplieguen los porcentajes de efectividad obtenidos. Estos índices permiten evaluar si el equipo de desarrollo está realizando un buen trabajo o no. (Beck, 2000)

En esta investigación se diseñaron un total de 6 casos de pruebas, se realizó una primera iteración de pruebas en la cual fueron probados los casos de pruebas correspondientes a las historias de usuario de

prioridad Muy Alta, detectándose un total de 7 no conformidades. En una segunda iteración fueron probados los casos de pruebas de prioridad Alta, detectándose dos no conformidades. Se realizó una tercera iteración donde fueron probados nuevamente el total de casos de pruebas diseñados para asegurar la calidad del producto, en la cual no se detectaron no conformidades. La siguiente gráfica muestra el resultado de la aplicación de las pruebas.

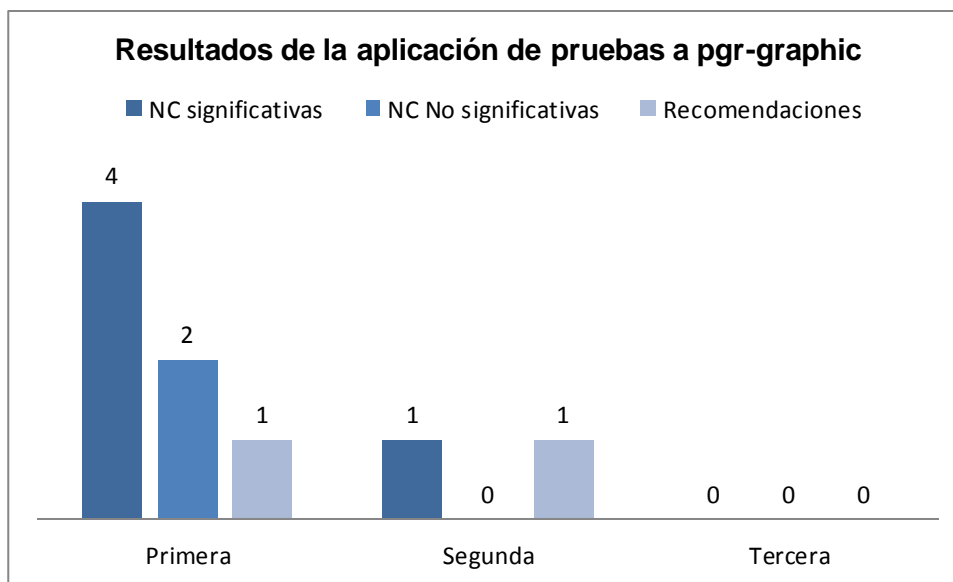


Figura 21: Resultados de las pruebas de aceptación al final de cada iteración

Conclusiones del capítulo

En este capítulo se definieron 9 tareas de ingeniería correspondientes a las historias de usuario identificadas. Se definió el estándar de codificación, se exponen los resultados de las pruebas de aceptación realizadas a la extensión *pgr-graphic* v2.0 mediante la técnica de caja negra, específicamente mediante el método de partición de equivalencia. Se realizaron tres iteraciones de pruebas en las cuales se probaron un total de 6 casos de prueba, detectándose 5 no conformidades significativas, 2 no significativas y 2 recomendaciones, las cuales fueron resueltas demostrando que la extensión desarrollada cumple con las expectativas del cliente.

CONCLUSIONES GENERALES

Como resultado de la investigación se obtuvo la segunda versión de la extensión de graficación para PostgreSQL *pgr-graphic*, arribándose a las siguientes conclusiones:

- Los tipos de gráficos implementados en la extensión fueron los de barras, líneas, circulares, dispersión y de caja-bigotes, al contar el paquete básico de graficación de R con funcionalidades para generarlos.
- Se utilizó como metodología de desarrollo de software XP, PostgreSQL 9.1 como gestor de base de datos, el paquete estadístico R 2.14 como motor gráfico, los lenguajes PL/R 8.3.10.13 y *Python* 2.7 y, PL/R para agregar funciones que no existen de forma nativa en el gestor.
- Se identificaron 6 historias de usuario desglosadas en 9 requisitos funcionales, de las que se derivaron 9 tarjetas CRC.
- Se definieron 9 tareas de ingeniería derivadas de las historias de usuario, empleándose un estándar de codificación para su implementación en 10 semanas.
- La validación de la extensión se realizó mediante pruebas de aceptación, específicamente mediante pruebas de caja negra empleando el método de partición de equivalencia.
- Se ejecutaron 6 casos de prueba en 3 iteraciones; detectándose 5 no conformidades significativas, 3 no significativas y 2 recomendaciones, las cuales fueron resueltas satisfactoriamente.

Por todo lo anteriormente expuesto se evidencia el cumplimiento de los objetivos propuestos.

RECOMENDACIONES

Independientemente de haberse logrado los objetivos propuestos, se recomienda:

- Emplear la extensión *pgr-graphic* v2.0 en entornos reales de producción.
- Continuar el desarrollo de la extensión empleando además del paquete básico el paquete *ggplot2* perteneciente al paquete estadístico R.

REFERENCIAS BIBLIOGRÁFICAS

- Giraldo Gómez, Gloria Lucia. 2010.** Ingeniería de software clase 5, Actores y sus roles. Modelo de Dominio. Escuela de sistemas, universidad nacional de Colombia, sede Merlin. Merlin, Colombia : s.n., 2010.
- Python, Software Foundation. 2013.** python. python. [En línea] 2013. <http://www.python.org/>.
- Ansejo, Jorge Sanchez. 2006.** Sistemas Gestores de Base de datos. 2006.
- Beck, Kent. 2000.** Extreme Programing. Addison-Wesley : Massachusetts, 2000. 0201616416.
- Capote Pons, Olga. 2005.** Introducción a las bases de datos: el modelo relacional. 2005.
- Conway, Joe. 2012.** PL/R. PL/R. [En línea] 2012. <http://www.joeconway.com/plr/>.
- Diaz Gutierrez, Alejandro. 2012.** 2012.
- Dominguez, Oscar. 2012.** Baquia. Baquia. [En línea] 4 de junio de 2012. [Citado el: 15 de octubre de 2012.] <http://www.baquia.com/blogs/cloud/posts/2012-06-04-big-data-como-gestionar-volumenes-de-informacion-de-mas-de-mil-millones-de-gbs>.
- educacion.es. 2012.** educacion.es. educacion.es. [En línea] noviembre de 2012. [Citado el: 10 de noviembre de 2012.] http://www.ite.educacion.es/formacion/materiales/180/cd/m4_10/grficos_de_lineas.html.
- Estadística. 2010.** Estadística Educativa. Estadística Educativa. [En línea] noviembre de 2010. [Citado el: 10 de noviembre de 2012.] http://estadisticaeducativaunefm.blogspot.com/2010/05/graficos-de-dispersion_09.html.
- Esteve, Marta . 2010.** Software Libre. Software Libre. [En línea] 21 de octubre de 2010. [Citado el: 3 de febrero de 2013.] <http://softwarefreedom.wordpress.com/2010/10/21/licencias-bsd/>.
- excellentias. 2013.** excellentias. excellentias. [En línea] 20 de mayo de 2013. [Citado el: 20 de mayo de 2013.] www.excellentias.com.
- Fuentes, Luis Nolberto. 2012.** Gráficos estadísticos. Gráficos estadísticos. [En línea] 2012. [Citado el: 14 de enero de 2013.] http://www.rmm.cl/index_sub.php?id_contenido=12179&id_seccion=8060&id_portal=1548.
- Hansen James, Gary W. 2010.** Diseño y administracion de base de datos. 2010.
- Hofmann, Arnold. 2010.** Los graficos en la gestión. 2010.

Infovis. 2012. Infovis. Infovis. [En línea] noviembre de 2012.]
<http://www.infovis.net/printMag.php?num=157&lang=1>.

Jacobson, Ivar, Booch , Grady y Rumbaugh , James. 2000. El proceso unificado del software. 2000. págs. 107, Cap 6, Epígrafe 6.3.

Marques, Mercedes. 2001. Apuntes de Ficheros y Bases de datos. 2001.

Mckinsey&Company. 2012. Mckinsey&Company. [En línea] 2012. [Citado el: 10 de octubre de 2012.]
http://www.mckinsey.com/local_language_information.

Mendelzon. 2000. Introduccion a las Bases de Datos Relacionales . 2000.

Mesa Reyes, Yunior y Vásquez Ortiz, Yudisney. 2011. Sitemas de Bases de Datos. Espacio de comunicación e intercambio para la comunidad técnica cubana de postgreSQL. PostgreSQL. Ciudad de la Habana, Cuba : s.n., 2011. pág. 6. 1994-1536.

Microsoft. 2013. Microsoft. Microsoft. [En línea] Microsoft, mayo de 2013. [Citado el: 22 de mayo de 2013.] <http://mac2.microsoft.com/help/office/14/es-es/word/item/dfecbba6-a4cf-46b2-a4cd-a60f30868cde>.

Navathe, Elmasri. 2005. Sistemas de bases de datos ,conceptos fundamentales. 2005.

Playfair, William . 1801. The Commercial and Political Atlas. 1801. 3ra edición.

PostgreSQL. 2013. PostgreSQL. [En línea] marzo de 2013.
<http://www.postgresql.org/docs/9.0/static/plpython.html>.

2013. PostgreSQL. PostgreSQL. [En línea] Comunidad Técnica Cubana, 10 de mayo de 2013. [Citado el: 10 de mayo de 2013.] <http://postgresql.uci.cu/>.

Pressman, Roger S. 2005. Ingeniería el software. Un enfoque práctico. 2005.

R. 2013. R. R. [En línea] Institute for Statistics and Mathematics of WU (Wirtschaftsuniversität Wien, 10 de mayo de 2013. [Citado el: 10 de mayo de 2013.] <http://www.r-project.org/>.

Scribd. 2012. Scribd. Scribd. [En línea] noviembre de 2012.
<http://www.scribd.com/doc/11854611/Graficacion>.

BIBLIOGRAFÍA

Ansejo, Jorge Sanchez. 2006. Sistemas Gestores de Base de datos. 2006.

Beck, Kent. 2000. Extreme Programing. Addison-Wesley : Massachusetts, 2000. 0201616416.

Capote Pons, Olga. 2005. Introducción a las bases de datos: el modelo relacional. 2005.

Conway, Joe. 2012. PL/R. PL/R. [En línea] 2012. <http://www.joeconway.com/plr/>.

Diaz Gutierrez, Alejandro. 2012. 2012.

Dominguez, Oscar. 2012. Baquia. Baquia. [En línea] 4 de junio de 2012. [Citado el: 15 de octubre de 2012.] <http://www.baquia.com/blogs/cloud/posts/2012-06-04-big-data-como-gestionar-volumenes-de-informacion-de-mas-de-mil-millones-de-gbs>.

educacion.es. 2012. educacion.es. educacion.es. [En línea] noviembre de 2012. [Citado el: 10 de noviembre de 2012.] http://www.ite.educacion.es/formacion/materiales/180/cd/m4_10/grficos_de_lineas.html.

Estadística. 2010. Estadística Educativa. Estadística Educativa. [En línea] noviembre de 2010. [Citado el: 10 de noviembre de 2012.] http://estadisticaeducativaunefm.blogspot.com/2010/05/graficos-de-dispersion_09.html.

Esteve, Marta . 2010. Software Libre. Software Libre. [En línea] 21 de octubre de 2010. [Citado el: 3 de febrero de 2013.] <http://softwarefreedom.wordpress.com/2010/10/21/licencias-bsd/>.

excellentias. 2013. excellentias. excellentias. [En línea] 20 de mayo de 2013. [Citado el: 20 de mayo de 2013.] www.excellentias.com.

Fuentes, Luis Nolberto. 2012. Gráficos estadísticos. Gráficos estadísticos. [En línea] 2012. [Citado el: 14 de enero de 2013.] http://www.rmm.cl/index_sub.php?id_contenido=12179&id_seccion=8060&id_portal=1548.

Giraldo Gómez, Gloria Lucia. 2010. Ingeniería de software clase 5, Actores y sus roles. Modelo de Dominio. Escuela de sistemas, universidad nacional de Colombia, sede Merlin. Merlin, Colombia : s.n., 2010.

Hansen James, Gary W. 2010. Diseño y administracion de base de datos. 2010.

Hofmann, Arnold. 2010. Los graficos en la gestión. 2010.

Infovis. 2012. Infovis. Infovis. [En línea] noviembre de 2012.]
<http://www.infovis.net/printMag.php?num=157&lang=1>.

Jacobson, Ivar, Booch , Grady y Rumbaugh , James. 2000. El proceso unificado del software. 2000.
págs. 107, Cap 6, Epígrafe 6.3.

Marques, Mercedes. 2001. Apuntes de Ficheros y Bases de datos. 2001.

Mckinsey&Company. 2012. Mckinsey&Company. [En línea] 2012. [Citado el: 10 de octubre de 2012.]
http://www.mckinsey.com/local_language_information.

Mendelzon. 2000. Introduccion a las Bases de Datos Relacionales . 2000.

Mesa Reyes, Yunior y Vásquez Ortiz, Yudisney. 2011. Sitemas de Bases de Datos. Espacio de comunicación e intercambio para la comunidad técnica cubana de postgresQL. PostgreSQL. Ciudad de la Habana, Cuba : s.n., 2011. pág. 6. 1994-1536.

Microsoft. 2013. Microsoft. Microsoft. [En línea] Microsoft, mayo de 2013. [Citado el: 22 de mayo de 2013.] <http://mac2.microsoft.com/help/office/14/es-es/word/item/dfecbba6-a4cf-46b2-a4cd-a60f30868cde>.

Navathe, Elmasri. 2005. Sistemas de bases de datos ,conceptos fundamentales. 2005.

Python, Software Foundation. 2013. python. python. [En línea] 2013. <http://www.python.org/>.

Playfair, William . 1801. The Commercial and Political Atlas. 1801. 3ra edición.

PostgreSQL. 2013. PostgreSQL. [En línea] marzo de 2013.
<http://www.postgresql.org/docs/9.0/static/plpython.html>.

2013. PostgreSQL. PostgreSQL. [En línea] Comunidad Técnica Cubana, 10 de mayo de 2013. [Citado el: 10 de mayo de 2013.] <http://postgresql.uci.cu/>.

Pressman, Roger S. 2005. Ingeniería el software. Un enfoque práctico. 2005.

R. 2013. R. R. [En línea] Institute for Statistics and Mathematics of WU (Wirtschaftsuniversität Wien, 10 de mayo de 2013. [Citado el: 10 de mayo de 2013.] <http://www.r-project.org/>.