Universidad de las Ciencias Informáticas

Facultad 5

Centro de Consultoría y Desarrollo de Arquitecturas Empresariales

Título: *Extensión de la herramienta LibreOffice para generar reportes dinámicos.*

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autora: *Yamila González González*

Tutora: *Ing. Alena M. Santiesteban García*

La Habana, junio de 2014

"Año 56 de la Revolución

"Obtener el triunfo con sacrificio es el mejor pretexto para no perder lo alcanzado."

José Martí

AGRADECIMIENTOS

A Fidel y a la Revolución por darme la maravillosa oportunidad de haber podido estudiar en esta grandiosa universidad.

A la persona más importante de mi vida y a la que más quiero, mi mamá Vicky, por ser madre y padre a la vez, por impulsarme y darme las fuerzas para llegar lejos sin escatimar sacrificios.

A mi papá por la ayuda.

A mi amiga Annelys por estar a mi lado en las buenas y en las malas, por las noches en vela, por sacarme de los apuros, por toda la ayuda y el apoyo que me ha dado, si no fuera por ella no hubiese pasado 5to año.

A mis amigos del aula que me han acompañado durante el transcurso de estos inolvidables años, en las noches de estudio, en la tensión de las pruebas y en las fiestas.

A Yudenia por ser mi hermana mayor, por ayudarme tanto y cuidarme siempre.

A todos los profesores que de alguna forma influyeron en mi formación como ingeniera, en especial a Belkis y a Pedro.

A mi tutora Alena por brindarme su ayuda incondicional en cada momento que lo necesité, así como a todos mis compañeros de laboratorio que en ocasiones molesté para aclarar dudas.

A todos los que han sido mis compañeros de apartamento.

A los que son mis amigos cercanos, Luilly, Adriana, Yaima, Yaumara, Karel, Chavelis, Carlos Alberto, Haymeé, Laura, Etián.

Al Chamo por ayudarme y dedicarme su tiempo.

DEDICATORIA

Dedico este trabajo de diploma, producto de un gran esfuerzo, a mi mamá porque de no ser por ella no estaría aquí y no fuese la persona que soy hoy.

A mi familia, en especial a mis abuelos por el amor y el apoyo que me han dado.

Yamila González

DECLARACIÓN DE AUTORÍA

Declaro ser autora de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

Yamila González González

_____

Firma del Autor

Ing. Alena M. Santiesteban García

_____

Firma del Tutor

DATOS DE CONTACTO

Autora:
Yamila González González
Universidad de las Ciencias Informáticas.
La Habana, Cuba.
Correo: ygonzalezg@estudiantes.uci.cu

Tutora:
Ing. Alena María Santiesteban García
Universidad de las Ciencias Informáticas.
La Habana, Cuba.
Correo: alena@uci.cu

RESUMEN

El uso de los reportes de base de datos ha tenido un auge significativo en los últimos años. Los reportes son utilizados por muchas empresas para mostrar información de las bases de datos y para ayudar en la gestión de proyectos y en la toma de decisiones. Existen varias herramientas dedicadas a la generación y manipulación de los reportes que cada vez se necesitan más y se hacen más complejos. Debido a ello es fundamental mejorar las opciones para generar reportes y satisfacer la demanda. LibreOffice, suite ofimática libre, provee la funcionalidad de generar reportes pero no para todas sus versiones.

La presente investigación titulada “Extensión de la herramienta LibreOffice para generar reportes dinámicos” propone el desarrollo de una extensión para generar reportes dinámicos en la herramienta LibreOffice con el propósito de ampliar las opciones de los usuarios que utilizan esta alternativa.

Esta extensión libre y compatible con todas las versiones de LibreOffice permite generar reportes a través de esta herramienta en múltiples formatos. Ofrece ventajas como: su utilización sin estar conectado a la red, el uso de varias tablas en un mismo reporte y la conexión a PostgreSQL sin tener que usar el componente LibreOffice Base.

La extensión fue desarrollada empleando un grupo de técnicas, tecnologías y metodología, las cuales son explicadas con más detalles en el presente documento.

Palabras clave: Extensión, LibreOffice, Reporte.

ABSTRACT

The use of database reports has had a significant boom in recent years. The reports are used by many companies to display information from the database and to assist in project management and decision-making. There are several tools dedicated for the generation and manipulation of the reports that are required increasingly more and become more complex. As a result is crucial to improve the options for generating reports and meet demand. LibreOffice, a free office suite provides functionality to generate reports but not for all its versions.

This research titled "LibreOffice extension to generate dynamic reports" proposes the development of an extension to generate dynamic reports in LibreOffice tool in order to expand the options of users who use this alternative.

This free extension and compatible with all versions of LibreOffice can generate reports through this tool in multiple formats. Offers advantages such as its use without being connected to the network, the use of multiple tables in one report and PostgreSQL connection without having to use LibreOffice Base component.

The extension was developed using a set of techniques, technologies and methodology which are explained in more detail herein.

Keywords: Extension, LibreOffice, Report.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

La informática constituye uno de los logros más grande alcanzado por el hombre y desde su surgimiento ha cambiado la vida de la sociedad gracias a los avances tecnológicos. El auge en las Tecnologías de la Informática y las Comunicaciones (TIC) y la rapidez con que fluye la información a nivel mundial favorece que se considere la misma como uno de los principales activos de cualquier organización.

Actualmente las empresas manejan gran volumen de información, por lo que necesitan almacenarlas en bases de datos. Para que luego puedan ser gestionadas a través de aplicaciones profesionales y mostradas a través de los llamados informes o reportes. Estos organizan y exhiben la información contenida en las bases de datos, aplicando un formato determinado a los mismos, para ser mostrado a través de un diseño atractivo y fácil de interpretar por los usuarios en las organizaciones (Brito Rodríguez, y otros, 2013).

Existen fuentes bibliográficas que definen el concepto de reportes de forma diferente al de informes, a diferencia de otros criterios que enfocan al concepto de reportes como informes, "…que son una herramienta de comunicación sumamente útil para transmitir los resultados económicos, sociales y ambientales de la organización a los distintos grupos de interés" (Cáliz, 2007).

Los informes o reportes son generados a partir de herramientas que se encargan de la gestión y manipulación de los mismos. Estas permiten estructurar y resumir datos relevantes guardados o generados por la misma aplicación de tal manera que se vuelvan útiles para los fines de los usuarios (Brito Rodríguez, y otros, 2013).

Para visualizar información y resultados, se utilizan reportes, diseñados a través de herramientas conocidas como generadores de reportes. Los generadores de reportes utilizan una especie de lenguaje transparente para el usuario, por medio del cual este realiza consultas a la base de datos y se obtiene información en forma de reporte. De esta manera se facilita la identificación de nuevas oportunidades de negocio o servicios.

Cuba valiéndose del potencial profesional con que cuenta, ha creado organismos y empresas para solventar la necesidad creciente que existe de desarrollar productos informáticos y de manejar información, ya sea para la comercialización y la demanda nacional o internacional. Sobre este tema el general de ejército Raúl Castro Ruz expresó:
"El desarrollo tecnológico combinado con el gran avance de las ciencias informáticas, ha propiciado que la complejidad de los proyectos de desarrollo de software se haya incrementado considerablemente" (Castro Ruz, 2011).

De esta manera la Universidad de Ciencias Informáticas (UCI) constituye una de las principales entidades productoras de software del país. Su modelo de producción está basado en Centros de Desarrollo especializados en áreas temáticas. En estos centros se vinculan la producción, la investigación y la formación de los estudiantes en profesionales de

la informática de alto nivel. Ejemplo de ello es el Centro de Consultoría y Desarrollo de Arquitecturas Empresariales (CDAE) que brinda servicios de consultoría a organizaciones.

Dicho centro cuenta con un Departamento de Gestión de Proyectos, su objetivo es facilitar el trabajo de los especialistas y profesores de los diferentes proyectos de la universidad. En este departamento se generan los reportes desde las tablas o consultas de una base de datos para la obtención resumida de los datos, que ayuda en la toma de decisiones en los diferentes proyectos de cada centro.

Una vía de fácil acceso para generar reportes en la UCI es la herramienta LibreOffice, conjunto completo de herramientas de ofimática disponibles libremente, que presenta un conjunto de extensiones que pueden instalarse en la herramienta para añadir una funcionalidad nueva. La extensión que le permite generar reportes es “Diseñador de Informes”, que se utiliza para crear informes elegantes o complejos de bases de datos desde el componente LibreOffice Base.

Actualmente existen deficiencias al generar los reportes, pues con la extensión existente se pueden hacer llamadas únicamente a una tabla o consulta y los reportes requieren información de varias tablas. Esta extensión solamente se puede utilizar en versiones de LibreOffice específicas, y en los repositorios de la universidad no está actualizada. También existen sistemas operativos como Nova que no cuentan con el componente Base, quien permite la conexión con PostgreSQL para la llamada de datos. Esto trae consigo dificultades a los especialistas en el proceso de apoyo a la toma de decisiones en la planificación de los proyectos en los centros.

Por lo antes expuesto surge como **problema de la investigación**:
Las deficiencias en las funcionalidades de la extensión diseñador de informes en la herramienta LibreOffice afecta la generación de reportes dinámicos.

Se define como **objeto de estudio**: extensión diseñador de informes en la herramienta LibreOffice.

El **objetivo general** consiste en implementar una extensión a la herramienta LibreOffice con conexión a base de datos PostgreSQL que permita generar reportes dinámicos.

Para lograr un mejor desarrollo de la investigación se han trazado los siguientes **objetivos específicos**:

- Realizar el Marco Teórico de la investigación.
- Realizar el análisis y diseño de una extensión para la generación de reportes dinámicos en LibreOffice en conexión con base de datos generada en PostgreSQL.
- Realizar la implementación de una extensión para la generación de reportes dinámicos en LibreOffice en conexión con base de datos generada en PostgreSQL.
- Realizar pruebas funcionales a la extensión implementada.

El objeto de la investigación está enmarcado en el **campo de acción:** herramienta LibreOffice con conexión base de datos PostgreSQL.

Con vistas al alcance de los objetivos propuestos se hace necesario realizar las siguientes **tareas de investigación**:

1) Realizar un estudio del estado del arte sobre los procesos que están involucrados en el desarrollo de extensiones para Libre Office sobre el lenguaje Python para la conexión a bases de datos en PostgreSQL.
2) Implementar una extensión de la herramienta Libre Office para la generación de reportes dinámicos.
3) Realizar en LibreOffice el diseño de un reporte permitiendo conectarse, a través de la extensión propuesta, a diferentes funciones mediante una base de datos en PostgreSQL.

En esta investigación se plantea la siguiente **idea a defender**: Con la herramienta propuesta será posible la generación de los reportes dinámicos en LibreOffice conectado con base de datos PostgreSQL.

El **aporte práctico de la investigación** se basa en la implantación de la extensión de reportes dinámicos en la herramienta LibreOffice. Esta posibilitará la generación de los reportes de bases de datos PostgreSQL, permitiendo a su vez que dichos reportes contengan la información solicitada, que luego podría ser usada en el análisis y la toma de decisiones en los proyectos dentro de los centros. También se espera que la documentación técnica generada sirva de guía para el uso de esta extensión de aplicaciones en el centro y la universidad.

Para el desarrollo de la investigación se utilizaron diferentes métodos científicos como los métodos teóricos y empíricos que se muestran a continuación:

Métodos teóricos:

1. Histórico-Lógico: Empleado para establecer un estudio de los diferentes comportamientos y características específicas que brinda cada una de las herramientas para generar reportes, en la determinación y selección de la propuesta específica de las herramientas a utilizar.

2. Analítico-Sintético: Permite el estudio de diferentes bibliografías con el fin de sintetizar, acotar las búsquedas, establecer cuáles son las fuentes fiables de información, y llegar a conclusiones aceptadas y bien balanceadas sobre el tema en cuestión.

3. Modelación: Permite establecer esquemas, modelar ideas de forma gráfica, además de posibilitar la utilización de diagramas para expresar información de manera organizada. En el proceso de análisis y diseño el modelaje constituye una herramienta indispensable para el entendimiento de las diferentes etapas del desarrollo del software.

Método empírico:

1. Entrevista: Utilizado para el proceso de entendimiento del negocio y los requisitos que el sistema deberá satisfacer, además del proceso de familiarización con el personal capacitado sobre los temas a tratar.

Estructura de la tesis:

Para una mejor comprensión del contenido la investigación se ha estructurado en tres capítulos, conclusiones generales y bibliografía utilizada, además de un glosario de términos, donde se explican en detalles los términos técnicos y poco precisos que han sido utilizados en la elaboración del documento, y los anexos que complementan el trabajo realizado.

- **Capítulo I: Fundamentación Teórica.**
  En este capítulo se realiza un estudio sobre los sistemas que están involucrados en la generación de reportes, así como de las extensiones de Libre Office. También se hace la selección de herramientas y tecnologías para el proceso de desarrollo de la extensión.

- **Capítulo II: Análisis y Diseño de la extensión.**
  En este capítulo se define todo lo referente a las funcionalidades que debe realizar el sistema, principalmente en las descripciones generales del funcionamiento del mismo. Además se elaboran una lista de requisitos funcionales y no funcionales que se deben tener en cuenta para la realización de la extensión.

- **Capítulo III: Implementación y pruebas de la extensión.**
  En este capítulo se realiza todo lo relacionado con los flujos de trabajo de implementación y prueba. Se presenta el modelo de implementación y se realizan pruebas a la extensión, para comprobar su correcto funcionamiento mediante los casos de prueba.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

En el presente capítulo se esclarecen los conceptos y definiciones necesarias para llevar a cabo el desarrollo de una extensión de LibreOffice. Se brinda un estudio sobre los sistemas para generar reportes dinámicos, tanto en la UCI como en el mundo; además de las características y funcionalidades de las extensiones más utilizadas para la generación de reportes. También se definen las tecnologías a utilizar en la construcción de una extensión.

El modo eficiente de desarrollar el proceso de gestión de informes es un tema que cobra vital importancia en la actualidad, el diseño de sistemas de reportes que esté conformado por un conjunto de herramientas con una arquitectura flexible, modular y robusta en el contexto actual, conlleva al éxito de la organización donde se aplique, una vez que perfecciona los procesos de presentación y uso de la información que en esta se maneja (Abreu Medina, y otros, 2012).

1.1 Reportes

En el ámbito de la informática, los reportes son informes que organizan y exhiben la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. De esta forma, confiere una mayor utilidad a los datos. Los reportes tienen diversos niveles de complejidad, desde una lista o enumeración hasta gráficos mucho más desarrollados. Según el programa informático y la base de datos en cuestión, los reportes permiten la creación de etiquetas y la elaboración de facturas, entre otras tareas.

De la misma forma, gracias a los reportes cualquier persona puede proceder a realizar un resumen de datos o clasificar estos en grupos determinados. Estos son obtenidos a partir de generadores de reportes.

Los generadores de reportes son herramientas que permiten a los usuarios acceder de forma simple y rápida a los datos o bases de datos guardados en forma de reportes. Estas herramientas ofrecen información en un formato particular y realizan varias operaciones a partir de los datos que se encuentran almacenados en una base de datos. Además permiten realizar consultas a las bases de datos obteniendo así la información en forma de reportes con un mejor dominio en el diseño y la visualización de los datos obtenidos (García Suárez del Villar, y otros, 2013).

Los reportes son de vital importancia por tener la tarea de procesar y mostrar datos, además de ser objetos que muestran información en un formato particular. Estos ofrecen la ventaja de poder realizar ciertas operaciones de utilidad para los usuarios, como imprimirlos, enviarlos por correo, guardarlos a un archivo, entre otras.

1.2 Generadores de reportes a nivel mundial

Son muchos los sistemas que actualmente facilitan el proceso de generación de reportes, que se están haciendo cada vez más importantes en el mercado mundial, en la medida en que las compañías demandan flexibilidad para ver detalle sobre cómo se está invirtiendo el presupuesto.

Estos sistemas permiten a los equipos de trabajo tomar decisiones, además de administrar, seguir y generar reportes fácilmente sobre sus actividades.
Es importante para crear y publicar reportes determinar qué información con su origen de datos se necesita mostrar en el reporte, teniendo en cuenta que al usar el reporte se debe tener acceso a los datos. Luego elegir la herramienta de creación de reportes para posteriormente crearlos y guardarlos.

Entre las principales herramientas generadoras de reportes que más se utilizan en el mercado se encuentran: Pentaho Reporting, Eclipse Birt y Jasper Reports (Brito Rodríguez, y otros, 2013).

1.2.1 Pentaho Reporting

Pentaho Reporting es una herramienta integrada para generar informes incluida en la suite Pentaho BI, conjunto de programas libres para generar inteligencia empresarial (*Business Intelligence*), fundado por Richard Daley y dirigido por Quentin Gallivan. Esta herramienta consiste en un motor de presentación, capaz de generar informes programáticos sobre la base de un archivo de definición XML.

La unidad de reportes de Pentaho (Pentaho Reporting) permite a las organizaciones acceder, dar formato y distribuir fácilmente la información a empleados, clientes y asociados. Pentaho provee acceso a fuentes de datos relacionales, OLAP o basadas en XML, además de ofrecer varios formatos de salida como PDF, HTML, Excel o hasta texto plano. También permite llevar esta información a los usuarios finales vía web, e-mail, portales corporativos o aplicaciones propias.

Tiene características como:

- Proporciona funcionalidad crítica para usuarios finales como acceso vía web, informes parametrizados, y otros.
- Proporciona ventajas a especialistas en informes como acceso a fuentes de datos heterogéneos, capacidad de integración en aplicaciones o portales, definición modular de informes (distinción entre presentación y consulta).
- Diseño de informes flexible debido su entorno de diseño gráfico, capacidad de uso de templates, acceso a datos relacionales, OLAP y XML.
- Multiplataforma (tanto a nivel de cliente como servidor): mac, linux/unix y Windows (Pentaho Corporation, 2005-2014).

1.2.2 Eclipse Birt

Desarrollado por la Eclipse Foundation, BIRT es un sistema de presentación de informes de código abierto basado en Eclipse para aplicaciones web, especialmente los basados en Java y Java EE.

BIRT tiene dos componentes principales: un diseñador de informes visuales dentro de Eclipse IDE para crear informes BIRT, y un componente de rutina para generar informes que pueden ser puestos en uso en cualquier entorno Java (*Java environment*). El proyecto BIRT también incluye un motor de gráficos que está integrado en el diseñador de informes y además puede ser usado por separado para incluir gráficas en una aplicación.

Los diseños de informes BIRT se hacen en XML y pueden acceder a cierto número de fuentes de datos diferentes incluyendo SQL *databases*, XML y Servicios Web.

Las principales características son:

- Está programado en Java.
- Multiplataforma.
- Licencia *Eclipse Public License* (The Eclipse Foundation, 2013).

1.2.3 Jasper Reports

Es una herramienta de creación de informes desarrollado por JasperSoft que tiene la habilidad de entregar contenido enriquecido al monitor, a la impresora o a ficheros PDF, HTML, XLS, CSV y XML. Su principal propósito es crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible. Está bajo *GNU Lesser General Public License*, por lo que es Software libre.

Posee características como:

- Está escrito completamente en Java y puede ser usado en gran variedad de aplicaciones de este tipo, incluyendo J2EE o aplicaciones web, para generar contenido dinámico.
- Es una biblioteca que puede ser embebida en cualquier aplicación Java. Sus funciones incluyen:
    - Scriptlets, que pueden acompañar a la definición del informe, y pueden ser invocados en cualquier momento para realizar un procesamiento adicional. El scriptlet se basa en Java, y tiene muchos hooks (ganchos) que se pueden invocar antes o después de las etapas de la generación de informes, como el Informe, Página, Columna o Grupo.
    - Sub-informes

Para usuarios con requisitos más sofisticados de gestión, los informes diseñados para JasperReports pueden ser fácilmente importados a JasperServer que es el servidor de informes interactivos (Jaspersoft Corporation, 2000-2014).

Estas herramientas proveen buenas soluciones a las necesidades de inteligencia de negocios de las distintas organizaciones. Por ser tecnologías multiplataformas ofrecen alternativas aprovechables en la generación de reportes. Están enfocadas a cubrir la demanda de creación de informes de cualquier tipo, además de que permiten a los desarrolladores de aplicaciones diseñar e integrar los reportes.

Además dependen de una serie de requisitos para su correcto funcionamiento que no son favorables para su uso por usuarios poco experimentados con respecto a otras herramientas. Por ejemplo, en Eclipse Birt el desarrollador debe trabajar con el Eclipse ya que este es el IDE de programación necesario para diseñar reportes con esta herramienta, Jasper Reports necesita que exista la Máquina Virtual de Java en el entorno donde se ejecute, y Pentaho Reporting que presenta falta de documentación fiable y un mayor coste de desarrollo.

Otra de las herramientas que presenta funcionalidades para la generación de reportes es LibreOffice, una suite ofimática de alta productividad y calidad profesional.

1.2.4 LibreOffice

Es un conjunto completo de herramientas de ofimática disponibles libremente, desarrollado por The Document Foundation.

LibreOffice incluye los componentes:

- Procesador de texto (LibreOffice Writer)
- Hoja de Cálculo (LibreOffice Calc)
- Presentaciones en Diapositivas (LibreOffice Impress)
- Programa de diseño de gráficos vectoriales (LibreOffice Draw)
- Gestor de bases de datos (LibreOffice Base)
- Editor de fórmulas matemáticas (LibreOffice Math) (Sanz, y otros, 2012)

De estos componentes el utilizado para el diseño de los reportes es Base. El mismo es una completa base de datos de escritorio, diseñada para satisfacer las necesidades de una amplia gama de usuarios, para todo tipo de usos. Proporciona asistentes para ayudar a los usuarios que son nuevos en el diseño de bases de datos (o simplemente nuevos en base) para crear tablas, consultas, formularios e informes (The Document Foundation, 2014).

Para los componentes de LibreOffice existen extensiones que agregan algún tipo de funcionalidad adicional a la suite, ya sea para un uso particular (Writer, Calc, Base,...), o para todas las aplicaciones.

Las extensiones son plug-ins[1] de software que se instalan como extras al LibreOffice estándar que se descarga, que permiten mejorar las funcionalidades originales de la aplicación. Una extensión es un paquete que puede instalarse en LibreOffice para añadir una funcionalidad nueva.

Diseñador de Informes es una extensión de LibreOffice que mejora las opciones para trabajar con reportes, añadiendo nuevas funcionalidades y ofreciendo la posibilidad de crear informes sin utilizar el asistente.

La extensión Diseñador de Informes**,** de Sun Microsystems. Open source, se utiliza para crear atractivos y complejos reportes de bases de datos desde el componente Base. El flexible editor de informes le permite definir grupo y encabezados y pies de página, y puede incorporar campos calculados para producir informes complejos. Al usarlo con Base puede diseñar informes para HSQL, Oracle y cualquier base de datos. Puede exportar sus informes en formato PDF u ODF (*OpenDocument format*), o enviarlos como adjuntos de un correo electrónico. (The Document Foundation, 2014)

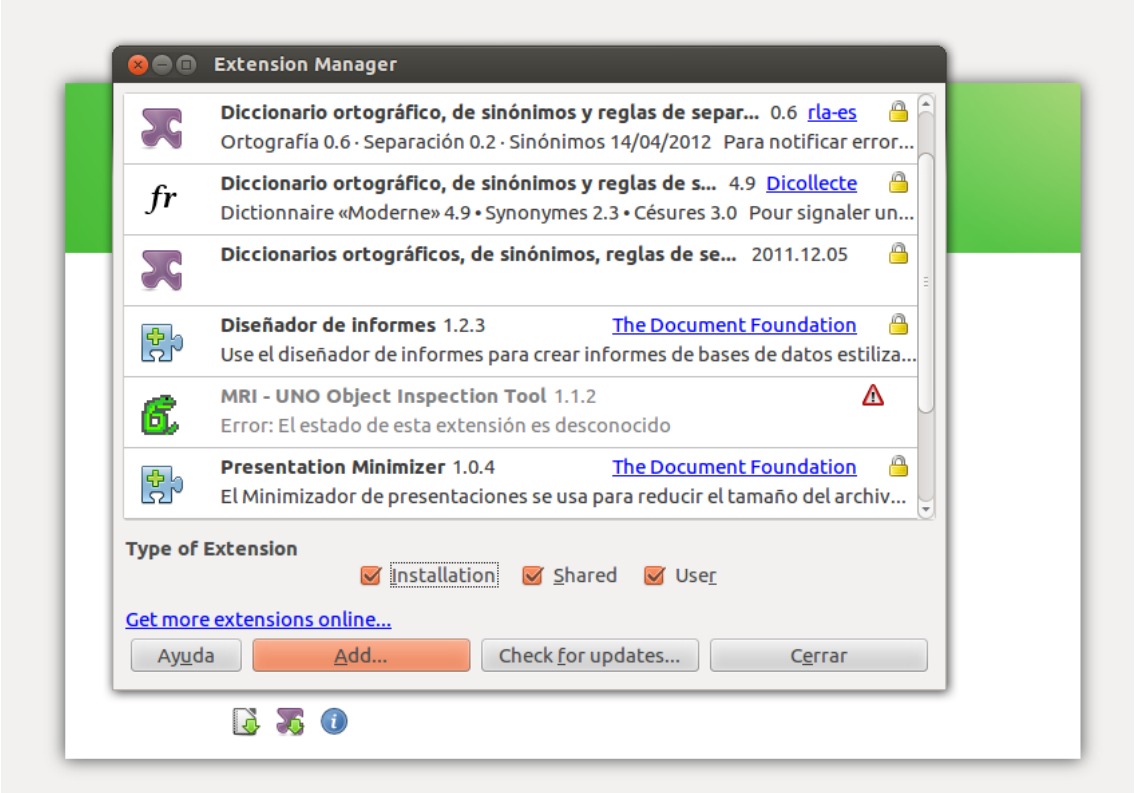


Fig. 1 Administrador de extensiones de LibreOffice

Un reporte en LibreOffice es un documento en el cual se muestran los datos organizados con formato y orden. Los reportes son documentos que se utilizan para presentar los datos de una forma más amigable. Es decir, su objetivo es recuperar

[1] Plugin: Plugin o plug-in -en inglés "enchufar"-, también conocido como addin, add-in, addon o add-on) es una aplicación informática que interactúa con otra aplicación para aportarle una función o utilidad específica.

información de la base de datos, igual que las consultas, pero con un formato que facilita su presentación e impresión en papel.

Los reportes proporcionan de forma útil la información que se encuentra en la base de datos, de forma similar a las consultas. Los reportes se generan desde las tablas o consultas de la base de datos. Pueden contener todos los campos de la tabla o consulta, o solamente un grupo seleccionado de campos. Los reportes pueden ser estáticos o dinámicos. Los estáticos contienen los datos de los campos seleccionados en el momento de generar el informe mientras que los dinámicos se pueden actualizar para que muestren los últimos datos (Sanz, y otros, 2012).

1.3 Generadores de reportes a nivel nacional

En Cuba, el modo eficiente de desarrollar el proceso de gestión de reportes es un tema que cobra vital importancia en la actualidad, el diseño de sistemas de reportes que esté conformado por un conjunto de herramientas con una arquitectura flexible, modular y robusta en el contexto actual, conlleva al éxito de la organización donde se aplique. A continuación se analizan varios sistemas de este tipo.

1.3.1 VERSAT Sarasola

Se distingue por ser el primer sistema de contabilidad cubano certificado, según las nuevas normativas establecidas por los Ministerios de Finanzas y Precios y de la Informática y las Comunicaciones, para este tipo de Software. Su principal creador, Miguel Cabrera González, es un contador profesional, Licenciado en Economía.

Es un sistema económico integrado constituido por 10 módulos o subsistemas que incluyen configuración y seguridad, contabilidad general y de gastos, costos y procesos, finanzas y caja. Además, en el proceso intervienen activos fijos, planificación y presupuestos, control de inventarios, pago de salario (nómina), facturación y generador de reportes.

El generador de reportes, también llamado “Complementos del Versat”, es la herramienta que permite obtener información sin mayores complejidades de uso, expresando sus resultados en una gama de funciones tipo y predeterminadas. Interactúa con Microsoft Excel, utilizando sus facilidades para el diseño de información. La concepción de estas funciones está ligada a los distintos subsistemas del Versat Sarasola.

Características del VERSAT Sarasola:

- Herramienta para la planificación económica, el control y el análisis de gestión.
- Diseñado para su empleo en cualquier tipo de entidad empresarial o presupuestada.
- Permite llevar el control y registro contable individual de todos los hechos económicos que se originan en las estructuras internas de las entidades, así

como exponer el estado financiero y toda la información económica y contable en este universo.

- Se estructura en un grupo de subsistemas en los cuales se procesan y contabilizan los documentos primarios, donde se anotan los movimientos, los recursos materiales, laborales y financieros que se utilizan en una entidad.
- Se logra establecer un proceso de interacción usuario-sistema.
- Rapidez y fiabilidad, a partir de la configuración del proceso de contabilización de los documentos primarios y de las propias posibilidades de trabajo contenidas en cada subsistema (Sosa Porteiro, y otros, 2005).

1.3.2 Generador Dinámico de Reportes (GDR)

Desarrollada en la UCI en el 2010 por el Centro de Tecnologías de Gestión de Datos (DATEC), es una aplicación web para la gestión de la información de cualquier empresa o institución, mediante el cual muestra el cumplimiento de las metas establecidas y con ello contribuir a la toma de decisiones. Posibilita diseñar reportes tabulares (con gráficos incluidos), tabla pivote y cruzada desde los gestores de Bases de Datos: SQL Server, SQLite, Oracle, MySQL y PostgreSQL. Dicha herramienta permite a los usuarios abstraerse de los conocimientos relacionados con los gestores de bases de datos y generar reportes en varios formatos con gran variedad de opciones en su diseño, marcando una diferencia entre los reportes tradicionales y los reportes dinámicos. GDR es uno de sus módulos de inteligencia de negocio de la Plataforma de Ayuda para la Toma de Decisiones y Soluciones Integrales (PATDSI).

GDR presenta características como:

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

La última versión estable de este sistema cubre el ciclo básico de la generación de reportes y soluciona el problema de obtener los diferentes informes en los sistemas de gestión de la información que se desarrollan en cualquier entorno empresarial, incluyendo la UCI. Es una herramienta web que permite a sus clientes consultar los gestores de bases de datos de sus organizaciones y generar reportes con la información que se manejan en sus negocios (Brito Rodríguez, y otros, 2013).

En estas aplicaciones de gestión los reportes cumplen un papel muy importante, entregan información en un formato particular y permiten realizar ciertas operaciones. Se destaca la generación de reportes de forma dinámica y efectiva cumpliendo con las solicitudes del cliente. Con estos sistemas se contribuye al ahorro monetario del país y a la elaboración de software similar o superior. Además, estas herramientas presentan

desventajas para su uso, en el GDR se dificulta el trabajo con grandes cantidades de volúmenes de datos; el VERSAT Sarasola no presenta una versión para sistemas operativos libres.

1.4 Metodología de desarrollo

Al desarrollar un producto de software surge la necesidad de utilizar un conjunto de procedimientos, técnicas, herramientas y soporte documental, por lo que es de gran importancia tener en cuenta la metodología de desarrollo a utilizar. Es necesario llevar una metodología apropiada con el objetivo de lograr la satisfacción tanto del cliente como del proveedor del producto y servicios.

La comparación y/o clasificación de metodologías no es una tarea sencilla debido a la diversidad de propuestas y diferencias en el grado de detalle, información disponible y alcance de cada una de ellas. A grandes rasgos, considerando su filosofía de desarrollo, aquellas metodologías con mayor énfasis en la planificación y control del proyecto, en especificación precisa de requisitos y modelado, reciben el apelativo de Metodologías Tradicionales (también denominadas Metodologías Pesadas o Peso Pesado). Otras metodologías denominadas Metodologías Ágiles, están más orientadas a la generación de código con ciclos muy cortos de desarrollo, se dirigen a equipos de desarrollo pequeños, hacen especial hincapié en aspectos humanos asociados al trabajo en equipo e involucran activamente al cliente en el proceso (Tangient LLC, 2014).

Las Metodologías Ágiles dan mayor valor al individuo, a la colaboración con el cliente y al desarrollo incremental del software con iteraciones muy cortas. Este enfoque muestra su efectividad en proyectos con requisitos muy cambiantes y cuando se exige reducir drásticamente los tiempos de desarrollo pero manteniendo una alta calidad (Canós, y otros, 2005).

Estas metodologías están orientadas al resultado del producto y no a la documentación; exige que el proceso sea adaptable, permitiendo realizar cambios de último momento (Pressman, 2011).

Para seleccionar una metodología ágil se analizaron dos metodologías ágiles que más fuerza están cobrando en la UCI: XP y Scrum.

1.4.1 Metodología Programación Extrema / Extreme Programming (XP)

XP es probablemente la metodología ágil más conocida; está centrada en la colaboración, la creación temprana y rápida de software y una serie de prácticas útiles en el desarrollo de software. Se funda en cuatro valores: comunicación, simplicidad, retroalimentación y coraje, que incluyen también 12 prácticas fundamentales como la programación por pares, refactorización constante y un desarrollo orientado a las pruebas, entre otras (Fernández Escribano, 2002).

Esta metodología está centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo. XP se basa en realimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. Se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico.

XP tiene ventajas como:

- Programación organizada.
- Menor taza de errores.
- Satisfacción del programador.

Los roles de XP son:

- Programador. El programador escribe las pruebas unitarias y produce el código del sistema.
- Cliente. Escribe las historias de usuario y las pruebas funcionales para validar su implementación. Además, asigna la prioridad a las historias de usuario y decide cuáles se implementan en cada iteración centrándose en aportar mayor valor al negocio.
- Encargado de pruebas (*Tester*). Ayuda al cliente a escribir las pruebas funcionales. Ejecuta las pruebas regularmente, difunde los resultados en el equipo y es responsable de las herramientas de soporte para pruebas.
- Encargado de seguimiento (*Tracker*). Proporciona realimentación al equipo. Verifica el grado de acierto entre las estimaciones realizadas y el tiempo real dedicado, para mejorar futuras estimaciones. Realiza el seguimiento del progreso de cada iteración.
- Entrenador (*Coach*). Es responsable del proceso global. Debe proveer guías al equipo de forma que se apliquen las prácticas XP y se siga el proceso correctamente.
- Consultor. Es un miembro externo del equipo con un conocimiento específico en algún tema necesario para el proyecto, en el que puedan surgir problemas.
- Gestor (*Big boss*). Es el vínculo entre clientes y programadores, ayuda a que el equipo trabaje efectivamente creando las condiciones adecuadas. Su labor esencial es de coordinación.

El ciclo de desarrollo consiste (a grandes rasgos) en los siguientes pasos:

1. El cliente define el valor de negocio a implementar.
2. El programador estima el esfuerzo necesario para su implementación.
3. El cliente selecciona qué construir, de acuerdo con sus prioridades y las restricciones de tiempo.
4. El programador construye ese valor de negocio.

5. Vuelve al paso 1.

En todas las iteraciones de este ciclo tanto el cliente como el programador aprenden. No se debe presionar al programador a realizar más trabajo que el estimado, ya que se perderá calidad en el software o no se cumplirán los plazos. De la misma forma el cliente tiene la obligación de manejar el ámbito de entrega del producto, para asegurarse que el sistema tenga el mayor valor de negocio posible con cada iteración.

En la actualidad existen una serie de libros asociados a cada una de las metodologías ágiles, pero es XP la metodología que resalta por contar con la mayor cantidad de información disponible y es con diferencia la más popular (Canós, y otros, 2005).

1.4.2 Metodología Scrum

Scrum es un proceso en el que se aplican buenas prácticas para trabajar colaborativamente, en equipo, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

También se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto (Albaladejo, 2014)**.**

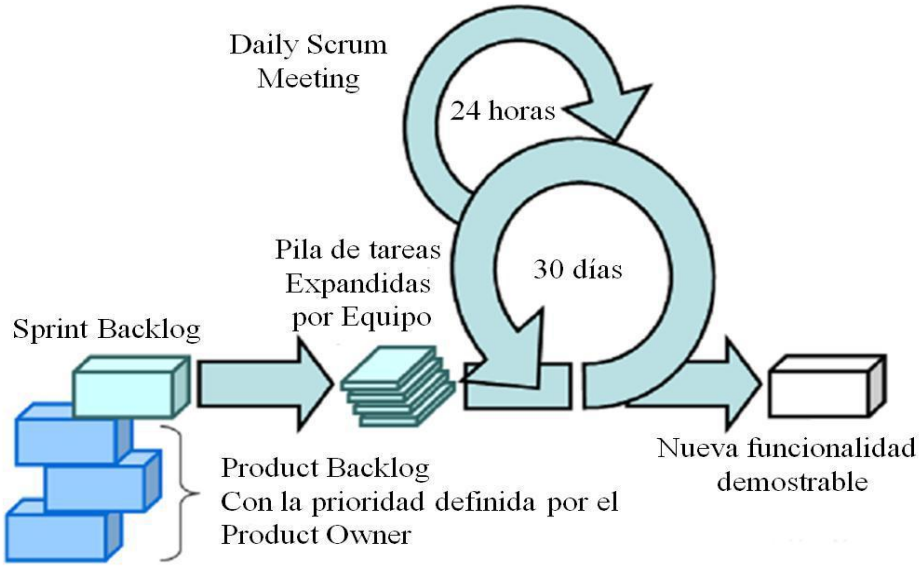


Fig. 2 Proceso de desarrollo de Scrum

El círculo inferior representa una iteración del desarrollo de las actividades que ocurren una tras otra. El producto de cada iteración es un incremento en el desarrollo. El círculo superior representa la reunión diaria que ocurre durante la iteración, en la cual los miembros individualmente del grupo conocen, inspeccionan las actividades y hacen los cambios apropiados. Como resultado de la iteración queda una lista de requerimientos. Este ciclo se repite durante todo el proyecto.

Scrum ofrece beneficios tales como:

- Gestión regular de las expectativas del cliente: El cliente establece sus expectativas indicando el valor que le aporta cada requisito del proyecto y cuando espera que esté completado.
- Resultados anticipados ("*time to market*"): El cliente puede empezar a utilizar los resultados más importantes del proyecto antes de que esté finalizado por completo.
- Flexibilidad y adaptación: De manera regular el cliente redirige el proyecto en función de sus nuevas prioridades, de los cambios en el mercado, de los requisitos completados que le permiten entender mejor el producto, de la velocidad real de desarrollo, entre otros.
- Mitigación de riesgos: Desde la primera iteración el equipo tiene que gestionar los problemas que pueden aparecer en una entrega del proyecto. Al hacer patentes estos riesgos, es posible iniciar su mitigación de manera anticipada.
- Productividad y calidad: De manera regular el equipo va mejorando y simplificando su forma de trabajar.
- Alineamiento entre cliente y equipo: Los resultados y esfuerzos del proyecto se miden en forma de objetivos y requisitos entregados al negocio. Todos los

participantes en el proyecto conocen cuál es el objetivo a conseguir. El producto se enriquece con las aportaciones de todos (Albaladejo, 2014).

Teniendo en cuenta lo antes expuesto se decide asumir como metodología de desarrollo Scrum, en consideración con las políticas del Departamento de Gestión de Proyectos para el desarrollo de software.

Además de las ventajas que presenta Scrum, permite al departamento entregar un producto funcional al finalizar cada iteración, posibilita ajustar la funcionalidad en base a necesidades de negocio del cliente, se tiene un seguimiento a diario y semanal de los proyectos, se obtiene un alcance acotado y viable. Se realiza una entrega quincenal o mensual de resultados (los requisitos más prioritarios en ese momento, ya completados).

Se debe tener en cuenta que el éxito de un proyecto no depende de una metodología específica, esta debe ser adaptada al contexto del proyecto para ofrecer una solución en dependencia a sus características. Las metodologías ágiles se destacan por su sencillez, tanto en su aprendizaje como en su aplicación, reduciéndose así los costos de implantación en un equipo de desarrollo.

1.5 Herramientas y tecnologías a utilizar en el desarrollo de la extensión

1.5.1 Herramienta de modelado: Visual Paradigm

Visual Paradigm es una herramienta CASE: Ingeniería de Software Asistida por Computación. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación. Es compatible con los equipos de desarrollo de software en la captura de requisitos, análisis de casos de uso, ingeniería de código, modelado de la clase, el modelado de datos, y otros.

Soporta las últimas versiones del Lenguaje de Modelado Unificado (UML) y la Notación y Modelado de Procesos de negocios, además de un generador de mapeo de objetos-relacionales para los lenguajes de programación Java .NET y PHP. Se integra con las siguientes herramientas de java: Eclipse, WebSphere, JBuilder, NetBeans IDE, Oracle JDeveloper, BEA Weblogic.

Se caracteriza por:

- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo
- Disponibilidad de múltiples versiones, con diferentes especificaciones.
- Licencia: gratuita y comercial.
- Soporta aplicaciones Web.
- Las imágenes y reportes generados, no son de muy buena calidad.
- Varios idiomas.

- Generación de código para Java y exportación como HTML.
- Fácil de instalar y actualizar.
- Compatibilidad entre ediciones (Pressman, 2011).

Se selecciona esta herramientas por estar disponible para múltiples plataformas(Windows, Linux), posee un diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad y además usa un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.

1.5.2 Entorno de desarrollo integrado: Eclipse

Eclipse es un entorno de desarrollo integrado (IDE), de Código abierto y Multiplataforma. Mayoritariamente se utiliza para desarrollar lo que se conoce como "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Es una potente y completa plataforma de Programación, desarrollo y compilación de elementos tan variados como sitios web, programas en C++ o aplicaciones Java. En este entorno de desarrollo integrado se pueden encontrar todas las herramientas y funciones necesarias para el trabajo, recogidas además en una atractiva interfaz que lo hace fácil y agradable de usar.

Ventajas en la utilización de Eclipse:

1. El entorno de desarrollo integrado (IDE) de Eclipse emplea módulos (en inglés plug-in) para proporcionar toda su funcionalidad al frente de la Plataforma de Cliente rico, a diferencia de otros entornos monolíticos donde las funcionalidades están todas incluidas, las necesite el usuario o no.
2. Este mecanismo de módulos es una plataforma ligera para componentes de software. Adicionalmente a permitirle a Eclipse extenderse usando otros lenguajes de programación como son C/C++ y Python, permite a Eclipse trabajar con lenguajes para procesado de texto como LaTeX, aplicaciones en red como Telnet y Sistema de gestión de base de datos.
3. La arquitectura plug-in permite escribir cualquier extensión deseada en el ambiente, como sería Gestión de la configuración. Se provee soporte para Java y CVS en el SDK de Eclipse.
4. La definición que da el proyecto Eclipse acerca de su Software es: "una especie de herramienta universal - un IDE abierto y extensible para todo y nada en particular" (The Eclipse Foundation, 2013).

Este IDE fue seleccionado principalmente por su potente editor de código, la interfaz amigable que presenta y la existencia en el ciberespacio de una gran cantidad de plugins que hacen del IDE una herramienta fuerte, además de su arquitectura basada en plugins. Entre otras de sus características tenemos que es un software libre y multiplataforma.

1.5.3 Sistema Gestor de bases de datos: PostgreSQL

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Es el sistema de gestión de bases de datos de código abierto más potente del mercado. PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando.

Entre las ventajas que presenta se encuentran las siguientes:

- Ahorros considerables en costos de operación.
- Estabilidad y confiabilidad.
- Soporte multiplataforma.
- Diseñado para ambientes de alto volumen.
- Herramientas gráficas de diseño y administración de BD.
- Permite la gestión de diferentes usuarios.

Algunas de las herramientas más conocidas de administración para el SGBD PostgreSQL son:

- EMS SQL Manager.
- PgAdmin.
- PgAccess (Martínez, 2010).

Este sistema fue seleccionado por su capacidad de ser un potente, robusto y estable gestor de BD, disponible para los principales sistemas operativos. Además se puede utilizar o modificar de forma gratuita. También por su seguridad en términos generales, integridad en las bases de datos, además de que posee características importantes para el trabajo con bases de datos como los disparadores, vistas y funciones.

1.5.4 Herramienta para la administración gráfica de PostgreSQL: pgAdmin3

La herramienta pgAdmin3 es una aplicación de diseño y manejo de bases de datos para su uso con PostgreSQL. Este software fue diseñado para responder a las necesidades de todos los usuarios, desde la escritura de simples consultas SQL a la elaboración de bases de datos complejas. La interfaz gráfica es compatible con todas las características de PostgreSQL y facilita la administración. La aplicación también incluye un editor de la sintaxis SQL, un editor de código del lado del servidor, un agente para la programación de tareas «SQL/batch/shell», soporte para el motor de replicación Slony-I y mucho más. La conexión del servidor se puede realizar mediante TCP/IP o Unix Domain Sockets (en plataformas *nix), y puede ser cifrado mediante SSL por seguridad. No se requieren controladores adicionales para comunicarse con la base de datos del servidor.

Sus características son:

- En pgAdmin3 se puede ver y trabajar con casi todos los objetos de la base de datos, examinar sus propiedades y realizar tareas administrativas.
- Una característica interesante de pgAdmin3 es que, cada vez que se realiza alguna modificación en un objeto, escribe la(s) sentencia(s) SQL correspondiente(s), lo que hace que, además de una herramienta muy útil, sea a la vez didáctica. También incorpora funcionalidades para realizar consultas, examinar su ejecución (como el comando explain) y trabajar con los datos (The pgAdmin Devlopment Team, 2002-2011).

Esta herramienta fue seleccionada ya que presenta una serie de ventajas, permite conectarse a bases de datos PostgreSQL que estén ejecutándose en cualquier plataforma, facilita la gestión y administración de bases de datos ya sea mediante instrucciones SQL o con ayuda de un entorno gráfico, además permite acceder a todas las funcionalidades de la base de datos; consulta, manipulación y gestión de datos.

1.5.5 Lenguaje Unificado de Modelado/ Unified Modeling Language (UML)

Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es importante resaltar que UML es un "lenguaje" para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional), pero no especifica en sí mismo qué metodología o proceso usar.

UML cuenta con varios tipos de diagramas, los cuales muestran diferentes aspectos de las entidades representadas (Object Manegement Group, Inc., 1997-2014).

1.5.6 Lenguaje de Programación

Un lenguaje de programación es un lenguaje artificial que se utiliza para expresar programas de ordenador.
Cada ordenador, según su diseño, “entiende” un cierto conjunto de instrucciones elementales (lenguaje de máquina). No obstante, para facilitar la tarea del programador, se dispone también de lenguajes de alto nivel más fáciles de manejar y que no dependen del diseño específico de cada ordenador (Rodríguez Sala, 2003).

El lenguaje de programación utilizado en la presente investigación es Python. Este lenguaje es interpretado o de script, con tipado dinámico, multiplataforma y orientado a

objetos. Tiene muchas de las características de los lenguajes compilados, por lo que se podría decir que es semi interpretado. Su sintaxis simple, clara y sencilla; el tipado dinámico, el gestor de memoria, la gran cantidad de librerías disponibles y la potencia del lenguaje, entre otros, hacen que desarrollar una aplicación en este lenguaje sea sencillo y rápido.

Su sintaxis es sencilla y cercana al lenguaje natural que los programas elaborados en Python parecen pseudocódigo. Por este motivo se trata además de uno de los mejores lenguajes para comenzar a programar. No es adecuado para la programación de bajo nivel o para aplicaciones en las que el rendimiento sea crítico.

Este leguaje ofrece características como:

- Permite dividir el programa en módulos reutilizables desde otros programas Python. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas. También hay módulos incluidos que proporcionan E/S de ficheros, llamadas al sistema, Sockets y hasta interfaces a GUI como Tk, GTK, Qt entre otros.
- Se utiliza como lenguaje de programación interpretado, lo que ahorra un tiempo considerable en el desarrollo del programa, pues no es necesario compilar ni enlazar. El intérprete se puede utilizar de modo interactivo, lo que facilita experimentar con características del lenguaje, escribir programas desechables o probar funciones durante el desarrollo del programa (González Duque, 2010).

1.6 Consideraciones Parciales

Cada vez se hace más necesaria la muestra de los datos de forma resumida, por lo que el uso de los reportes aumenta y también su complejidad. En la UCI existen plataformas en las que es necesario generar reportes, ya sea para comprobar el estado de un proyecto o para visualizar elementos de cualquier base de datos. A partir del estudio realizado se llegó a la conclusión de que la metodología a utilizar es Scrum, debido a que es una metodología apropiada para proyectos pequeños y de requisitos cambiantes. Como lenguaje de modelado se seleccionó UML y como herramienta CASE el Visual Paradigm, puesto a que es una herramienta potente y multiplataforma. Python fue el lenguaje de programación seleccionado y se emplea eclipse como IDE.

CAPÍTULO 2: ANÁLISIS Y DISEÑO DE LA EXTENSIÓN

En este capítulo se describe la propuesta de solución para el sistema, se realiza el diseño de la extensión y se muestran los requisitos funcionales y no funcionales que se deben tener en cuenta para su implementación. Además se exponen los artefactos necesarios que fueron creados con la guía de la metodología Scrum a lo largo del desarrollo de la extensión. Se identifican las funcionalidades que debe tener una extensión para facilitar la integración y configuración de cada una de estas características.

2.1 Propuesta de solución

Se propone realizar una extensión de la herramienta LibreOffice que permita generar reportes dinámicos a partir de su integración con la herramienta. La misma brindará nuevas funcionalidades para realizar reportes, permitiendo a los usuarios hacerlo de una forma confiable, amigable y sencilla, teniendo como fuente de datos una base de datos PostgreSQL.

2.2 Modelo de Dominio

El Modelo de Dominio o Conceptual se utiliza para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema. Se toma como el punto de partida para el diseño del sistema. Cuando se realiza la programación orientada a objetos, el funcionamiento interno del software va a imitar en alguna medida a la realidad, por lo que el mapa de conceptos del modelo de dominio constituye una primera versión del sistema (Conferencia #2. Fase de Inicio, Modelo del Negocio, ISW 1.).

A continuación se muestran los eventos que suceden en el entorno donde estará el sistema, relacionados en un diagrama UML. Este modelo comprende y describe los conceptos más importantes dentro del contexto del sistema.

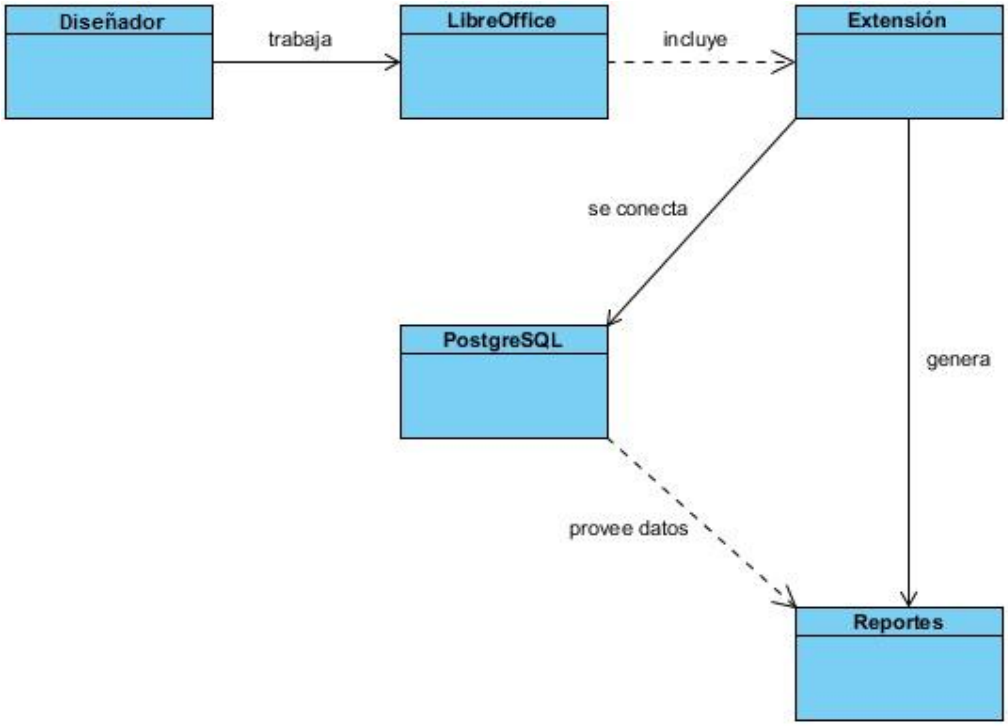


Fig. 3 Modelo de Dominio

En la figura 3 se muestra el proceso que se realiza para la generación de reportes a través de la herramienta LibreOffice, además de las clases que relacionadas entre sí conforman el modelo. El diseñador trabaja en la herramienta, donde se incluye la extensión para luego generar reportes. Es importante destacar que el proceso de generación de un reporte está asociado a un origen de datos, por lo que es necesario conectarse a una base de datos alojada en PostgreSQL. La conexión a la base de datos se realiza desde la extensión, de ahí se obtendrán los datos a mostrar en el reporte.

2.2.1 Definición de conceptos del Modelo de Dominio

Diseñador: son los diseñadores de reportes responsables de utilizar la herramienta "LibreOffice".

LibreOffice: es un conjunto completo de herramientas de ofimática que permite, a través de la funcionalidad que se añade, crear diversos reportes con datos procedentes de una base de datos.

Extensión: representa la extensión, contiene las funcionalidades que permite a partir de una base de datos generar reportes dinámicos.

PostgreSQL: es la herramienta encargada de la gestión de la base de datos.

Reporte: es el informe que organiza y exhibe de forma organizada la información contenida en la base de datos.

2.3 Actores del sistema

Un actor es un usuario del sistema, este desempeña el trabajo que es de valor para el negocio. Para este sistema el diseñador es el responsable de realizar las actividades que serán automatizadas en el futuro sistema.

Tabla. 1 Actores del sistema

Nombre del Actor	Descripción
Diseñador	Es el encargado de realizar los diseños de los reportes, para generarlos de forma dinámica con cada una de los elementos del diseño anteriormente predefinido.

2.4 Pila de producto (*Product Backlog*)

El *product backlog* es un documento de alto nivel para todo el proyecto. Contiene descripciones genéricas de todos los requerimientos, funcionalidades deseables, entre otros, priorizadas según su retorno sobre la inversión. Es el qué va a ser construido. Es abierto y cualquiera puede modificarlo. Contiene estimaciones tanto del valor para el negocio como el esfuerzo de desarrollo requerido. Esta estimación ayuda a ajustar la línea temporal y de manera limitada, la prioridad de las diferentes tareas (Pressman, 2011).

2.4.1 Requisitos Funcionales

Definen qué es lo que el sistema debe hacer, para lo cual se identifican las funcionalidades requeridas y las restricciones que se imponen. Además son condiciones o capacidades que necesita un usuario para resolver un problema o lograr un objetivo. Son capacidades o condiciones que el sistema debe cumplir. Los requerimientos funcionales se mantienen invariables sin importar con que propiedades o cualidades se relacionen (Conferencia #3. Flujo de trabajo de Requerimientos.).

Tabla. 2 Requisitos Funcionales

Requisitos Funcionales		
ID	**Nombre**	**Descripción**
1	Crear reporte.	El sistema debe permitir crear un reporte.
2	Especificar parámetros de conexión.	El sistema debe permitir especificar los parámetros de conexión a la base de datos PostgreSQL.
3	Seleccionar tablas de la base de datos.	El sistema debe permitir seleccionar las tablas procedentes de la fuente establecida con anterioridad.
4	Seleccionar campos de las tablas.	El sistema debe permitir añadir los campos necesarios a mostrar en el reporte, procedentes de las tablas seleccionadas.
5	Introducir valores de	El sistema debe permitir introducir los valores de salida

	salida.	del reporte que el usuario necesite.
6	Aplicar propiedades al reporte.	El sistema debe permitir aplicar propiedades para configurar el aspecto del reporte así como de los datos.
7	Generar reporte dinámico.	El sistema debe permitir generar el reporte con los datos seleccionados durante su diseño, datos que se irán actualizando cada vez que se genere el reporte.

2.4.2 Definición de las historias de usuario

Son técnicas utilizadas para especificar los requisitos del software. Se trata de tarjetas en las cuales el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (Canós, y otros, 2005).

Tabla. 3 Historia: Crear reporte

Historia de Usuario	
Número: 1	**Usuario:** Diseñador
Nombre historia: Crear reporte	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 3	**Iteración asignada:** 1
Descripción: Se permitirá la creación de un reporte.	
Observaciones: Para crear un reporte se necesita tener añadida la extensión y activada la funcionalidad.	

Tabla. 4 Historia: Especificar parámetros de conexión

Historia de Usuario	
Número: 2	**Usuario:** Diseñador
Nombre historia: Especificar parámetros de conexión	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Alto
Puntos estimados: 2	**Iteración asignada:** 1
Descripción: Se establecerán los valores de la conexión a la base de datos: nombre de la base de datos, dirección ip del servidor, puerto, usuario y contraseña.	
Observaciones: Es necesario tener alguna base de datos PostgreSQL para el acceso a los datos.	
Prototipo de interfaz:	

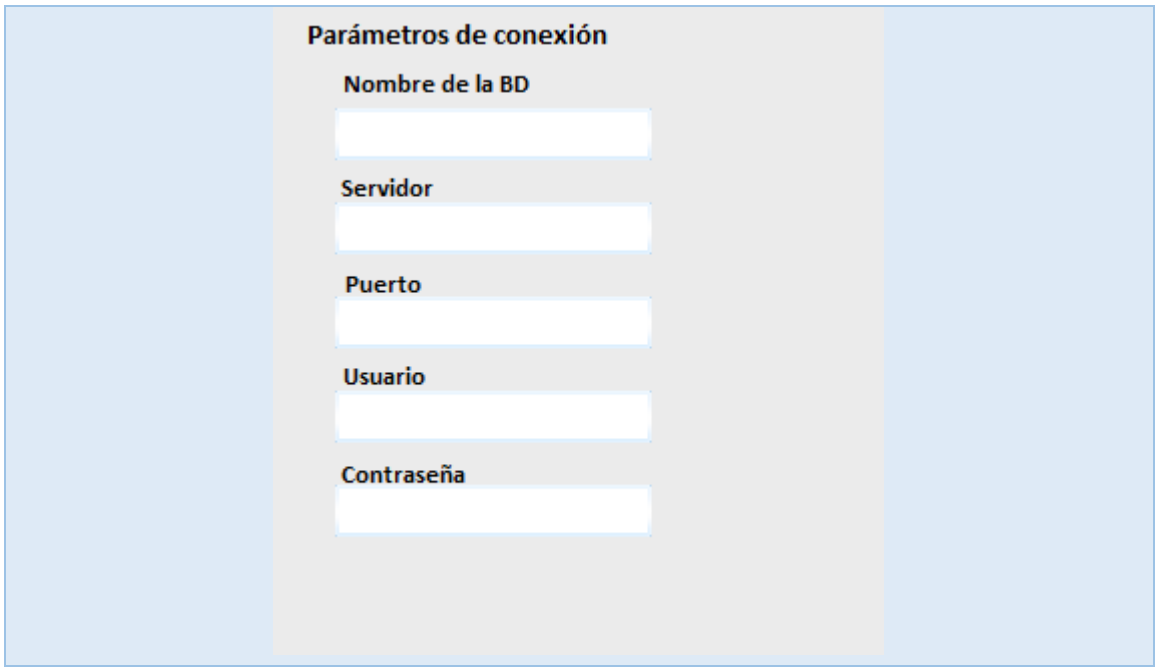


Tabla. 5 Historia: Seleccionar tablas de la base de datos

Historia de Usuario	
Número: 3	**Usuario:** Diseñador
Nombre historia: Seleccionar tablas de la base de datos	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Descripción: Se elegirán las tablas de la base de datos especificada de la que se quiere mostrar campos en el reporte.	
Observaciones: Es necesario que para seleccionar las tablas se especifique con anterioridad la fuente de donde provienen los datos.	

Tabla. 6 Historia: Seleccionar los campos de las tablas.

Historia de Usuario	
Número: 4	**Usuario:** Diseñador
Nombre historia: Seleccionar campos de las tablas	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** Medio
Puntos estimados: 3	**Iteración asignada:** 1
Descripción: Se elegirán los campos correspondientes al diseño del reporte, estos campos son llamados de la base de datos y se actualizan en el reporte cada vez que este es generado según sus valores en las tablas.	
Observaciones:	

Es necesario que para añadir los campos se seleccione con anterioridad la tabla de donde provienen los datos.

Tabla. 7 Historia: Introducir valores de salida

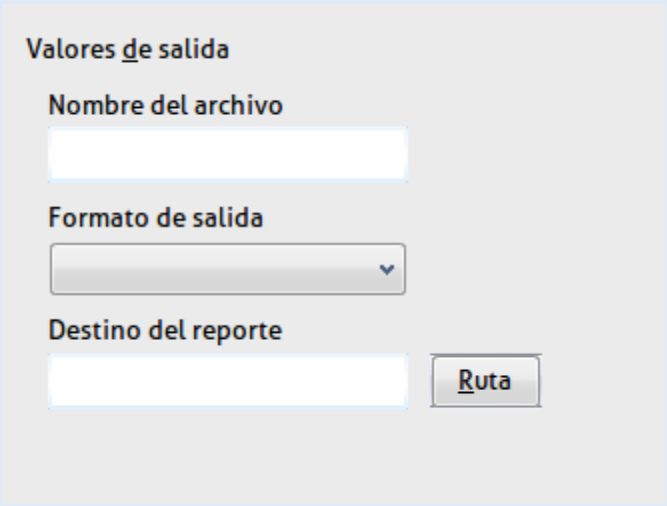
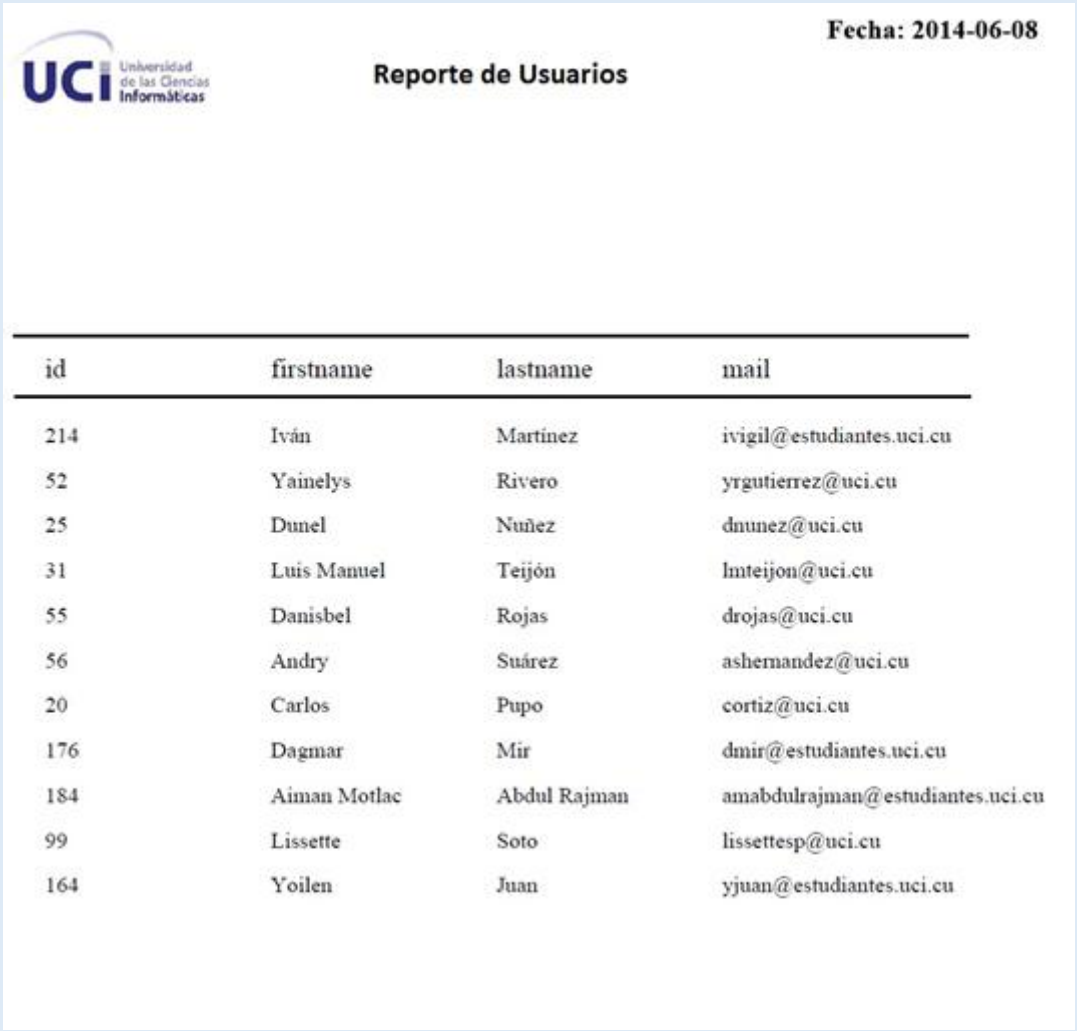
Historia de Usuario	
Número: 5	**Usuario:** Diseñador
Nombre historia: Introducir valores de salida	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Descripción: Se establecerán los valores de cómo se quiere que sea la salida del reporte: nombre del archivo, el formato de salida (ods o pdf), y la ruta de destino del reporte.	
Observaciones: Es necesario tener alguna base de datos de PostgreSQL para el acceso a los datos.	
Prototipo de interfaz: 	

Tabla. 8 Historia: Aplicar propiedades al reporte

Historia de Usuario	
Número: 6	**Usuario:** Diseñador
Nombre historia: Aplicar propiedades al reporte	
Prioridad en negocio: Media	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Descripción: Se le aplicará un formato adecuado a los elementos del reporte para que sea más amigable y agradable para la vista del usuario.	
Observaciones:	

Tabla. 9 Historia: Generar reporte dinámico

Historia de Usuario	
Número: 7	**Usuario:** Diseñador
Nombre historia: Generar reporte dinámico	
Prioridad en negocio: Baja	**Riesgo en desarrollo:** Medio
Puntos estimados: 2	**Iteración asignada:** 1
Descripción: Se generará un reporte con los datos seleccionados durante el diseño de este.	
Observaciones:	
Prototipo de interfaz:	



2.5 Pila de tareas del Sprint (*Sprint Backlog*)

El *sprint backlog* es un documento detallado donde se describe el *cómo* el equipo va a implementar los requisitos durante el siguiente sprint. Las tareas se dividen en *horas* con ninguna tarea de duración superior a 16 horas. Las tareas en el *sprint backlog* nunca son asignadas, son tomadas por los miembros del equipo del modo que les parezca oportuno (Pressman, 2011).

Tabla. 10 Pila de Sprint I

Pila de Sprint I (Base de Datos)			
Backlog Tarea	**Estado**	**Estimado**	**Real**
Conexión a la Base de Datos			
Introducir parámetros de conexión	100%	2	2
Conectar base de datos con PostgreSQL	100%	4	5

Tabla. 11 Pila de Sprint II

Pila de Sprint II (Construcción del reporte)			
Backlog Tarea	**Estado**	**Estimado**	**Real**
Creación, diseño y generación del reporte			
Seleccionar tablas	100%	5	4
Seleccionar campos	100%	10	10
Definir nombre del archivo	100%	10	15
Especificar formato	100%	15	15
Elegir destino	100%	5	5
Añadir texto de encabezado	100%	5	5
Definir formato de letra	100%	5	5
Generar reporte	100%	10	15

Tabla. 12 Pila de Sprint III

Pila de Sprint III (Validación)			
Backlog Tarea	**Estado**	**Estimado**	**Real**
Pruebas de caja negra	100%	15	15

2.6 Requisitos No Funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. Los requerimientos no funcionales forman una parte significativa de la especificación. Son importantes para que clientes y usuarios puedan valorar las características no funcionales del producto, pues si se conoce que el mismo cumple con la toda la funcionalidad requerida, las propiedades no funcionales, como cuán usable, seguro, conveniente y agradable, pueden marcar la diferencia entre un producto bien aceptado y uno con poca aceptación (Conferencia #3. Flujo de trabajo de Requerimientos.).

Se definen los siguientes requisitos no funcionales asumiendo los de la herramienta “LibreOffice”, ya que la extensión por sí sola no cumple funcionalidad:

Requisitos de Software

- La herramienta “LibreOffice”.
- Linux kernel versión 2.6.18 o superior
- Paquete python-psycopg2.
- Paquete python-pygresql.

Requisitos de Hardware

- PC compatible con Pentium (Pentium III, Athlon o sistema más reciente recomendado).
- 256 MB de RAM (512 MB de RAM recomendado o superior a esta).
- Hasta 1.55Gb de espacio disponible en disco duro.

Restricciones del diseño y la implementación

- Se hace uso de la herramienta “LibreOffice” en su versión 4.0.2 y IDE Eclipse 4.2
- El lenguaje de programación usado para la implementación es Python.

Requisitos de Usabilidad

- Facilidad de uso por parte de los usuarios: La interfaz debe ser lo más descriptiva posible, permitiendo que las operaciones a realizar por los usuarios estén bien descritas, de manera que se puedan entender claramente.
- La interfaz debe tener mensajes contextuales asociados a los objetos.

Requisitos de Soporte

- Proveer un manual de usuario.

Requisitos de Portabilidad

- La extensión una vez integrada a la herramienta LibreOffice, podrá ser instalada y disponer de la misma en diferentes versiones y sistemas operativos libres.

2.7 Diagrama de clases del diseño

El diagrama de clases del diseño muestra las propiedades y funcionalidades de cada clase en el lenguaje de desarrollo y tecnologías seleccionados. Expresa la colaboración y responsabilidades de cada clase entorno al sistema que conforman y muestra cómo quedaría implementada toda la aplicación en términos lógicos.

A continuación se muestra el diagrama de clases del diseño de la extensión a implementar. En dicho diagrama se muestran las clases que va a contener la extensión así como las relaciones entre ellas. También se especifican los atributos y los métodos de cada una de las clases.

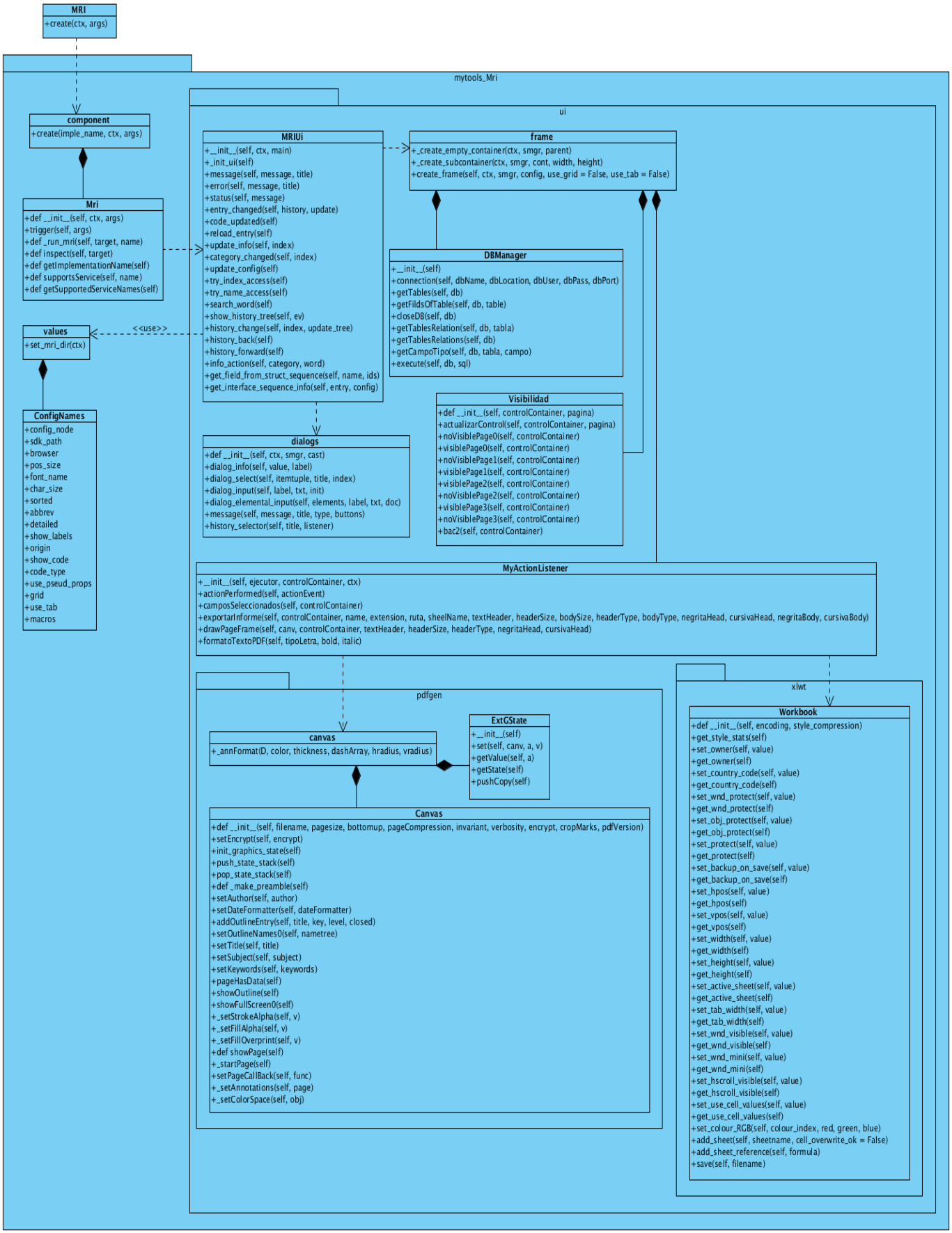


Fig. 4 Diagrama de clases del diseño.

2.7.1 Descripción de las clases del diseño

MRI: Clase principal para iniciar el funcionamiento del sistema.

component: Se encarga de crear los componentes del sistema. Se encuentra compuesta por la clase Mri.

Mri: Clase encargada de inicializar los componentes del sistema.

MRIUi: Clase encargada de crear las interfaces de usuario y la información a trabajar por el sistema.

values: Se encarga de manejar las configuraciones del sistema. Compuesto por la clase ConfigNames.

ConfigNames: Clase que contiene las configuraciones por defecto del sistema.

dialogs: Se encarga de manejar las interfaces del sistema.

frame: Es donde se definen los componentes de las interfaces. Compuesto por las clases DBManager, Visibilidad y MyActionListener.

DBManager: Clase encargada de manejar el acceso a datos.

Visibilidad: Clase encargada manejar la visibilidad de los diferentes componentes del sistema.

MyActionListener: Clase encargada de manejar los eventos del sistema.

canvas: Se encarga de inicializar las funcionalidades para la generación de reportes en formato pdf. Compuesto por las clases Canvas y ExtGState.

Canvas: Clase que contiene las funcionalidades para la generación de reportes en formato pdf.

ExtGState: Clase encargada de manejar los estados de los informes.

Workbook: Clase que contiene las funcionalidades para la generación de reportes en formato ods.

2.8 Patrones Utilizados

A lo largo del proceso de diseño y desarrollo, los atributos de calidad juegan un papel importante, pues en base a estos se generan las decisiones de diseño y argumentos que los justifican. Dado que la arquitectura de software inhibe o facilita los atributos de calidad, resulta de particular interés analizar la influencia de ciertos elementos de diseño utilizados para la definición de la misma, determinando sus características. Estos elementos de diseño son los estilos arquitectónicos y los patrones de diseño. (Camacho, y otros, 2004)

Para la implementación y funcionalidad de la extensión deben implementarse estilos arquitectónicos como los que a continuación se describen, así como los patrones de diseño

GRASP (*General Responsibility Assignment Software Patterns*) y GOF (*Gang Of Four*) con el fin de agilizar el proceso de implementación al desarrollador a través de la transformación de modelos.

2.8.1 Descripción de la arquitectura

Generalmente, en el proceso de desarrollo de una aplicación, suelen emplearse varios estilos arquitectónicos, lo que permite dar una mejor solución por el uso de las ventajas que proporcionan cada uno de ellos. En el desarrollo de la extensión se utiliza el estilo arquitectónico Arquitectura en 3 Capas.
El estilo arquitectural en capas se basa en una distribución jerárquica de los roles y las responsabilidades para proporcionar una división efectiva de los problemas a resolver. Los roles indican el tipo y la forma de la interacción con otras capas y las responsabilidades la funcionalidad que implementan. La mayor ventaja de este patrón de arquitectura es que en el desarrollo se lleva a cabo en varios niveles, o capas, y en el caso de que exista algún error o la necesidad de algún cambio obligatorio, solo es necesario cambiar el nivel en cuestión, sin afectar el correcto funcionamiento del resto del sistema (Pressman, 2011).

La selección de este estilo se basa en la necesidad de dividir las capas de acuerdo con su responsabilidad. En la figura 4 se muestra cómo se definieron 3 capas:

- La capa de Presentación, LibreOffice, es la que se le muestra al usuario, con el objetivo de presentarle la aplicación y gestionar la información manejada. La principal característica que tiene que poseer, es que debe ser amigable y lograr que el usuario se sienta motivado para emplear la aplicación.
- La capa de Lógica del negocio es la encargada de manejar todas las reglas que tienen que cumplirse con el objetivo de manejar las funcionalidades del sistema.
- La capa de Acceso a datos es la encargada de contener las entidades necesarias para el trabajo del sistema, con el objetivo de comunicar a la lógica del con la base de datos.

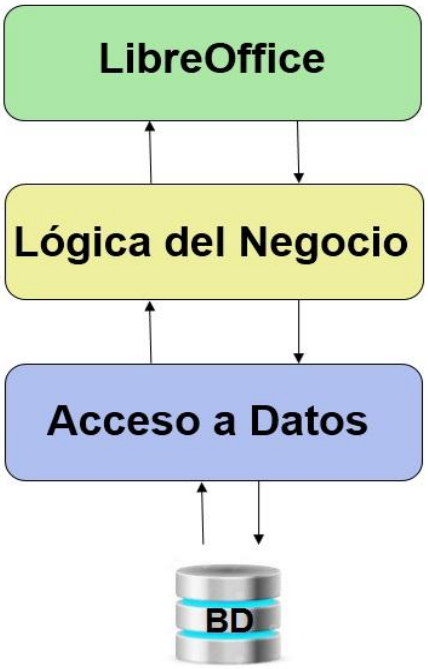


Fig. 5 Arquitectura N-Capas (3-Capas)

2.8.2 Patrones GRASP

Los patrones GRASP representan los principios básicos de la asignación de responsabilidades a objetos, expresados en forma de patrones. GRASP es el acrónimo para *General Responsibility Assignment Software Patterns*, del español Patrones Generales de Software para Asignar Responsabilidades (Larman, 1998).

Para el desarrollo de la extensión se utilizan los siguientes patrones.

Experto: Asigna una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad. El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clases sencillas y más cohesivas que son más fáciles de comprender y mantener (Larman, 1998).

En la presente investigación se utiliza para implementar la responsabilidad de realizar las acciones directamente con la base de datos, en la clase *MyActionListener* para asignar responsabilidades a la clase *DBManager*. Con la utilización de este patrón se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide.

Creador: Define la clase que debe tener la responsabilidad de crear una nueva instancia de otra clase. El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos. El propósito fundamental de este patrón es encontrar un creador que se debe conectar con el objeto producido en cualquier evento (Larman, 1998).

En la presente investigación se utiliza para la comunicación entre los presentadores y las interfaces, los presentadores contienen los datos de inicialización necesarios para construir las vistas, por lo que los presentadores son los creadores de las vistas y utiliza sus objetos para trabajarlos. La clase *MRIUi* es la encargada de este proceso.

Bajo acoplamiento: El acoplamiento mide el grado en que una clase está conectada a otra, tiene conocimiento de otra o, de alguna manera, depende de otra. El bajo acoplamiento permite crear clases más independientes, más reutilizables, lo que implica mayor productividad.

En la presente investigación se utiliza con el objetivo de evitar una gran cantidad de conexiones entre las diferentes clases, permitiendo principalmente la independencia de cada uno para reducir el impacto en los futuros cambios que se realicen.

Alta cohesión: La cohesión es la medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. El patrón Alta Cohesión es la meta principal que ha de tenerse en cuenta en cada momento en todas las decisiones de diseño. Es un patrón evaluativo que el desarrollador aplica al valorar sus decisiones de diseño. Una clase de alta cohesión posee un número relativamente pequeño, con una importante funcionalidad relacionada y poco trabajo que hacer (Larman, 1998).

En la presente investigación se utiliza para minimizar las responsabilidades que presentan las clases, por lo que las mismas solo presentarán la información necesaria para su trabajo, evitando los métodos y atributos que pueden sobrecargarlas y dificulte su entendimiento y reutilización.

2.8.3 Patrones GOF

Los patrones GoF (*Gang Of Four*) son patrones de diseño clasificados según su propósito en creacionales, estructurales y de comportamiento.

Singleton (Instancia única): Es de tipo creacional. Está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. (Awesome Inc, 2013)

En la presente investigación se utiliza cuando debe haber únicamente una instancia de una clase y debe ser claro su acceso para los clientes. El patrón Singleton o Solitario es implementado por la clase DBManager.

2.9 Consideraciones Parciales

Se definieron los conceptos más importantes para el desarrollo de la aplicación, además de la especificación de los requisitos funcionales y no funcionales para el correcto funcionamiento del sistema. Se llegó a la conclusión de que con la identificación del actor y la realización de las historias de usuario se obtiene un mayor entendimiento de los requisitos funcionales previamente establecidos, así como la pila de tareas. Se seleccionaron los patrones empleados para el proceso de desarrollo de la extensión como una solución a un problema de diseño.

CAPÍTULO 3: IMPLEMENTACIÓN Y PRUEBAS DE LA EXTENSIÓN

En este capítulo se describe el plan de liberaciones, y además se muestra el modelo de implementación del sistema constituido por dos elementos fundamentales, que ponen en práctica el diseño de la extensión, diagrama de componentes y diagrama de despliegue. Se definen los estándares de codificación a utilizar en la implementación. Se describen las pruebas a realizar, con el objetivo de probar las funcionalidades de la extensión en diferentes casos.

3.1 Plan de liberaciones

Una vez elaboradas las historias de usuario y estimado el esfuerzo de los desarrolladores para la realización de las mismas, es necesario elaborar la planificación de las etapas de implementación de la extensión. El plan contiene las historias de usuario por iteraciones, definiendo cuáles serán desarrolladas en cada iteración del proceso de implementación. La implementación de la extensión se realiza en tres iteraciones:

- Iteración 1.

En esta primera iteración se le da cumplimiento a las historias de usuario de prioridad alta (1, 2, 3, 4), las que constituyen la estructura básica de la extensión.

- Iteración 2.

En esta segunda iteración se le da cumplimiento a las historias de usuario de prioridad media (5, 6). Estas le permiten al usuario definir valores que le brindarán al sistema un alto nivel de aceptación.

- Iteración 3.

En esta tercera y última iteración se le da cumplimiento a la historia de usuario número 8. Con la implementación de esta se termina con las funcionalidades de la extensión propuesta.

Tabla. 13 Plan de liberaciones

Iteración	Descripción de la Iteración	Orden de la HU a Implementar	Duración total de la iteración
1	Desarrollo de las historias de usuario de prioridad alta.	1, 2, 3, 4	8 semanas
2	Desarrollo de las historias de usuario de prioridad media.	5, 6	6 semanas
3	Desarrollo de las historias de usuario de prioridad baja.	7	1 semana

3.2 Modelo de implementación

El Modelo de Implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes podemos encontrar datos, archivos, ejecutables, código fuente y los directorios. Fundamentalmente, se describe la relación que existe desde los paquetes y clases del modelo de diseño a subsistemas y componentes físicos.

Un diagrama de implementación muestra:

- Las dependencias entre las partes de código del sistema (diagramas de componentes).
- La estructura del sistema en ejecución (diagrama de despliegue) (Daniele, 2007).

3.2.1 Diagrama de componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones, muestran las opciones de realización incluyendo código fuente, binario y ejecutable

Los componentes representan todos los tipos de elementos software que entran en la fabricación de aplicaciones informáticas Pueden ser simples archivos, paquetes, bibliotecas cargadas dinámicamente, entre otros.

Los diferentes tipos de componentes son:

- Executable: Especifica un componente que se puede ejecutar en un nodo.
- Library: Especifica una biblioteca de objetos estática o dinámica.
- Table: Especifica un componente que representa una tabla de una base de datos.
- File: Especifica un componente que representa un documento que contiene código fuente o datos.
- Document: Especifica un componente que representa un documento (Scribd Inc, 2014).

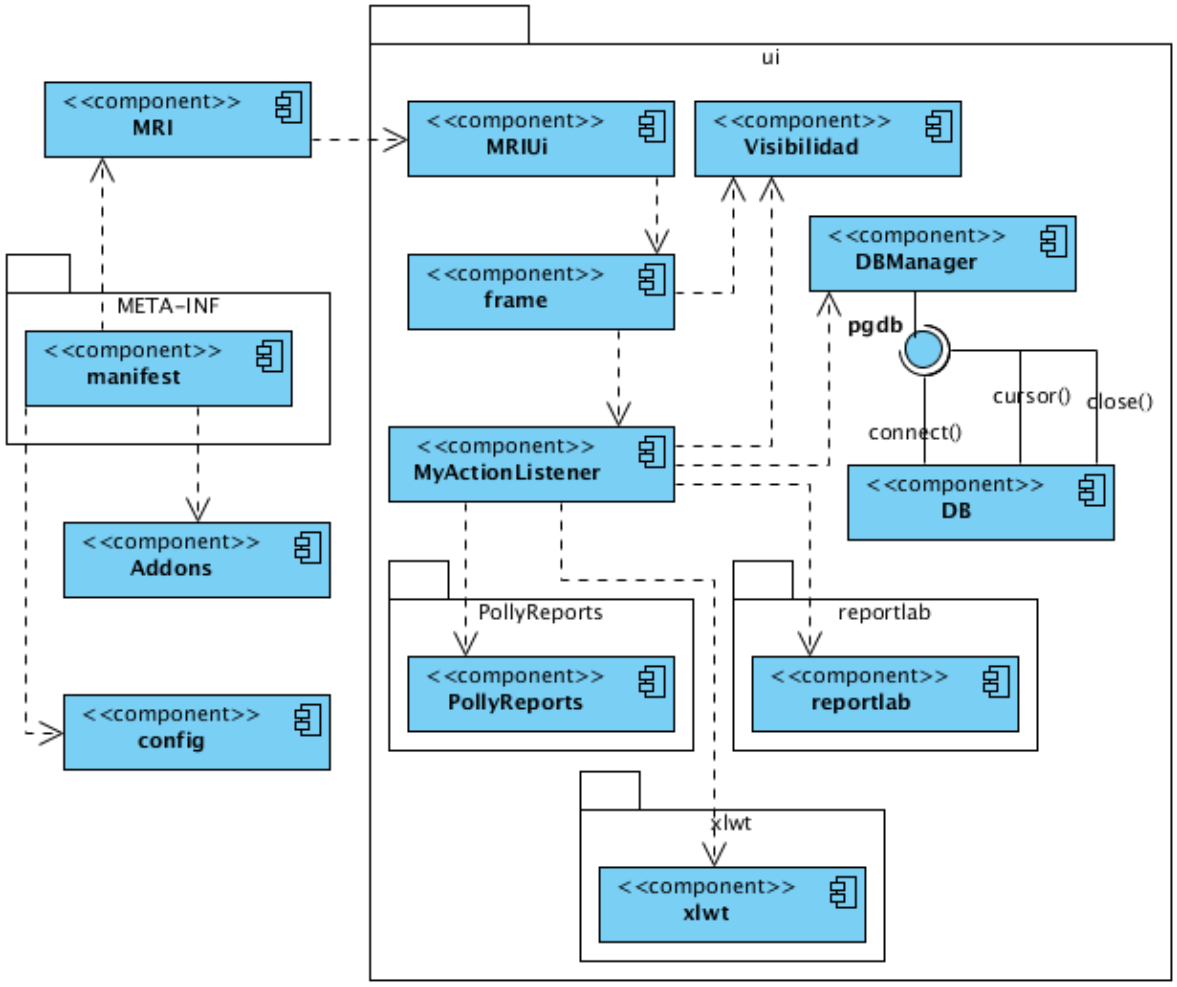


Fig. 6 Diagrama de componentes

El diagrama de componentes presenta cada uno de los componentes asociados a la construcción de la extensión para la herramienta de ofimática LibreOffice, así como la relación de dependencia entre los componentes que la integran.

3.2.2 Diagrama de despliegue

El Diagrama de Despliegue es un diagrama que se utiliza para modelar el hardware utilizado en las implementaciones de sistemas y las relaciones entre sus componentes.

- Permiten modelar la disposición física o topología de un sistema.
- Muestra el hardware usado y los componentes instalados en el hardware.
- Muestra las conexiones físicas entre el hardware y las relaciones entre componentes (Scribd Inc, 2014).

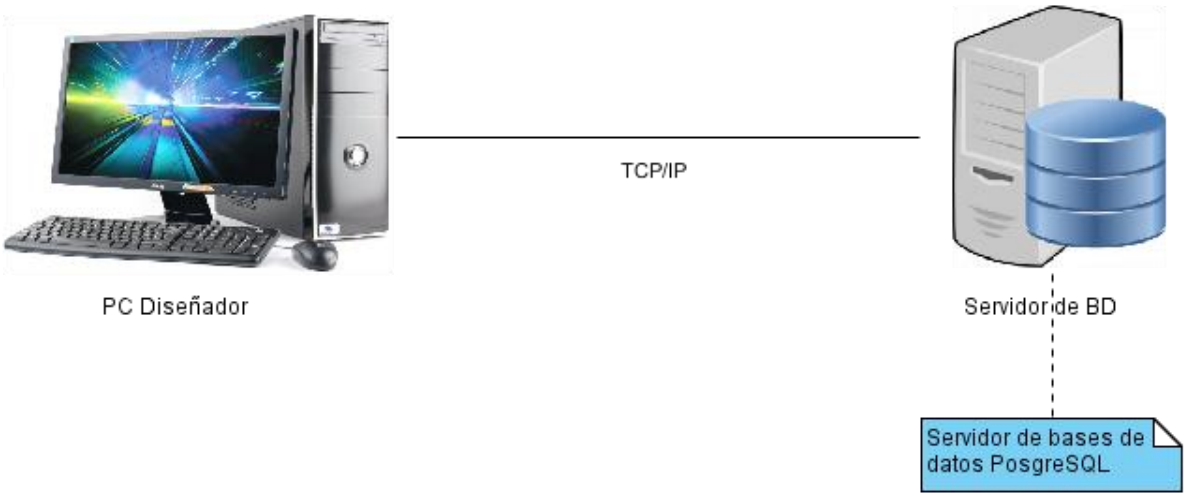


Fig. 7 Diagrama de despliegue

3.3 Estándares de codificación

3.3.1 Formateo del código

Indentación: Usa 4 espacios por cada nivel de indentación.

Tabuladores o espacios: La forma más popular de indentar en Python es utilizar sólo espacios. No se recomienda mezclar tabuladores y espacios.

Tamaño máximo de línea: Limita todas las líneas a un máximo de 79 caracteres.

Líneas en blanco: Separa las funciones no anidadas y las definiciones de clases con dos líneas en blanco. Las definiciones de métodos dentro de una misma clase se separan con una línea en blanco (van Rossum, 2007).

Ejemplo:

```
class EntryBase(object):
    def __init__(self, mri, target, inspected=None, type_info=None):
        self.__dict__["target"] = target
        self.__dict__["inspected"] = inspected
        self.__dict__["type"] = type_info
        self.__dict__["mri"] = mri
        self.__dict__["code_entry"] = None

    def _convert_to_tuple(self, value):
        if isinstance(value, list) or isinstance(value, tuple):
            a = []
            for i in value:
                a.append(self._convert_to_tuple(i))
            return tuple(a)
        else:
            if isinstance(value, Entry):
                return value.get_target()
            return value
```

Fig. 8 Ejemplo de formateo del código

3.3.2 Imports

Los imports se colocan siempre en la parte superior del archivo, justo después de cualquier comentario o cadena de documentación del módulo, y antes de las variables globales y las constantes del módulo (van Rossum, 2007).

```
#  limitations under the License.

import mytools_Mri.tools
import mytools_Mri.values


class ModuleEntry(object):
```

Fig. 9 Ejemplo de imports

3.3.3 Comentarios

Comentarios de bloque:

- Los comentarios de bloque generalmente se aplican al código que se encuentra a continuación, y se indentan al mismo nivel que dicho código. Cada línea de un comentario de bloque comienza con un # y un único espacio (a menos que haya texto indentado dentro del comentario).
- Los párrafos dentro de un comentario de bloque se separan por una línea conteniendo un único #.

Comentarios en línea:

- Utiliza comentarios en línea de forma moderada.
- Un comentario en línea es un comentario que se encuentra en la misma línea que una sentencia. Los comentarios en línea deberían separarse por al menos dos espacios de la sentencia que comentan. Deberían comenzar con un # y un espacio (van Rossum, 2007).

3.3.4 Convenciones de nombres

Estilos de nombrado:

- minusculas_con_guiones_bajos.
- PalabrasEnMayusculas (*CapWords* o *CamelCase*, llamado así por el parecido de las mayúsculas con las jorobas de los camellos). Algunas veces también se llaman *StudlyCaps*.
- capitalizacionMezclada (se diferencia de PalabrasEnMayusculas porque la primera letra está en minúsculas).

Nombres de paquetes y módulos:

- Los módulos tienen nombres cortos formados en su totalidad por letras minúsculas. Se utilizan guiones bajos en el nombre del módulo si mejora la legibilidad.
- Los paquetes Python también tienen nombres cortos formados de letras minúsculas.

Nombres de clases:

- Los nombres de clases usan la convención *CapWords*.

Nombres de funciones:

- Los nombres de funciones están en letras minúsculas, con palabras separadas mediante guiones bajos según sea necesario para mejorar la legibilidad.

Argumentos de funciones y métodos:

- Se usa siempre *'self'* como primer argumento de los métodos de instancia.

Nombres de métodos y variables de instancia:

- Se utilizan las reglas de los nombres de funciones: minúsculas con palabras separadas por guiones bajos cuando es necesario para mejorar la legibilidad. Se usan guiones bajos al inicio sólo para métodos no públicos y variables de instancia (van Rossum, 2007).

3.4 Creación e integración de la extensión con LibreOffice

Para la creación e integración de la extensión se necesita crear una carpeta con el nombre que se desee, dicho nombre no debe contener espacios ni símbolos o signos. En esta carpeta se almacenan todos los archivos que se requieren para crear la extensión. A continuación se exponen una serie de pasos a seguir para lograr crear la extensión e integrarla a la herramienta:

1. Implementación del servicio/interfaz en Python: La extensión incorpora un código creado en lenguaje Python, donde se implementan las funcionalidades principales de la extensión.

2. Crear *description.xml*: El archivo *description.xml* contiene la declaración de los elementos que se mostrarán al usuario al instalar la extensión en el administrador de extensiones.

3. Crear *Addons.xcu*: En este archivo se hace la declaración de los componentes del perfil de usuario, las barras de herramientas y menús, y se realizan las modificaciones pertinentes.
4. Crear *manifest.xml*: En *manifest.xml* se indica dónde están almacenados todos esos archivos declarativos y las librerías de código, y de qué tipo de elemento se trata en cada caso. Debe estar contenido en una carpeta llamada *META-INF* de forma obligatoria. El administrador de extensiones busca ese archivo en esa carpeta, y si no lo encuentra, se queda colgado o no instala correctamente la extensión.

5. Empaquetar todo en zip y nombrarlo como Reportes.oxt: La extensión en realidad es un archivo comprimido en formato *.zip* que contiene todos los archivos, imágenes, carpetas. Se comprimen todos los elementos y se cambia el *.zip* por *.oxt*.

6. Instalar con *unopkg /usr/lib/libreoffice/program/unopkg add Reportes.oxt*, ó abriendo la extensión con LibreOffice.

3.5 Pruebas de software

La prueba del software es un elemento crítico para la garantía de calidad del software y representa una revisión final de las especificaciones, del diseño y de la codificación. Las pruebas son un proceso de ejecución de un programa con la intención de descubrir errores.
Los objetivos principales de realizar una prueba son:

- Detectar un error.
- Tener un buen caso de prueba, es decir que tenga más probabilidad de mostrar un error no descubierto antes.
- Descubrir un error no descubierto antes (éxito de la prueba).

3.5.1 Pruebas funcionales

Las pruebas funcionales**,** se basan en la funcionalidad de la extensión, tienen como objetivo asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. El método que utiliza este tipo de prueba es el método de Caja Negra.

Método de prueba

El método de prueba a utilizar es el método de Caja Negra. Se realiza con el fin de asegurar que el producto es operativo, se centra en los requisitos funcionales del software y permite obtener entradas que prueben todos los requisitos funcionales del programa. Para aplicar estas pruebas se tienen en cuenta varias técnicas como son: métodos de prueba basados en grafos, partición equivalente, análisis de valores límites, prueba de comparación, de ellos se seleccionó la técnica de partición equivalente. Esta técnica divide el dominio de entrada de un programa en clases de datos y su diseño de casos de prueba se basa en la evaluación de las clases de equivalencia.

3.5.2 Casos de prueba

Tabla. 14 Caso de Prueba: Crear reporte.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP1	El usuario debe crear un reporte	El sistema muestra la ventana del generador de reportes.	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en el menú. 4. Seleccionar “Generador de reportes”.

Tabla. 15 Caso de Prueba: Especificar parámetros de conexión.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP2	El usuario debe entrar los parámetros de conexión.	El sistema muestra la ventana de introducir los parámetros de conexión	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en el menú. 4. Seleccionar “Generador de reportes”.

Tabla. 16 Caso de Prueba: Seleccionar tablas de la base de datos.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP3	El usuario debe seleccionar tablas de la base de datos a mostrar en el reporte.	El sistema muestra la ventana de seleccionar campos.	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en el menú. 4. Seleccionar “Generador de reportes”. 5. Accionar el botón “Siguiente” en la parte inferior de la ventana.

Tabla. 17 Caso de Prueba: Seleccionar campos de las tablas.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP4	El usuario debe seleccionar los campos de las tablas que va a mostrar en el reporte.	El sistema muestra la ventana de seleccionar campos.	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en

			el menú. 4. Seleccionar “Generador de reportes”. 5. Accionar el botón “Siguiente” en la parte inferior de la ventana.

Tabla. 18 Caso de Prueba: Introducir valores de salida.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP5	El usuario debe introducir los valores de salida del reporte.	El sistema muestra la ventana de introducir valores de salida.	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en el menú. 4. Seleccionar “Generador de reportes”. 5. Accionar el botón “Siguiente” en la parte inferior de la ventana. 6. Accionar el botón “Siguiente” en la parte inferior de la ventana.

Tabla. 19 Caso de Prueba: Aplicar propiedades al reporte.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP6	El usuario debe aplicar propiedades al reporte.	El sistema muestra la ventana de aplicar propiedades al reporte.	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en el menú. 4. Seleccionar “Generador de reportes”.

			5. Accionar el botón “Siguiente” en la parte inferior de la ventana. 6. Accionar el botón “Siguiente” en la parte inferior de la ventana. 7. Accionar el botón “Siguiente” en la parte inferior de la ventana.

Tabla. 20 Caso de Prueba: Generar reporte dinámico.

Escenario de prueba	Descripción	Respuesta del sistema	Flujo central
EP7	El usuario debe generar un reporte dinámico.	El sistema muestra la ventana de aplicar propiedades al reporte.	1. Abrir LibreOffice. 2. Acceder a “Herramientas” en la barra de menú. 3. Seleccionar “Complementos”, última opción que aparece en el menú. 4. Seleccionar “Generador de reportes”. 5. Accionar el botón “Siguiente” en la parte inferior de la ventana. 6. Accionar el botón “Siguiente” en la parte inferior de la ventana. 7. Accionar el botón “Siguiente” en la parte inferior de la ventana. 8. Accionar el botón “Siguiente” en la parte inferior de la ventana. 9. Accionar el botón “Finalizar” en la parte inferior de la ventana.

3.5.3 Pruebas de rendimiento

Las pruebas de rendimiento son las pruebas que se realizan, desde una perspectiva, para determinar lo rápido que realiza una tarea un sistema en condiciones particulares de trabajo. También puede servir para validar y verificar otros atributos de la calidad del sistema, tales como la escalabilidad, fiabilidad y uso de los recursos. Las pruebas de rendimiento son un subconjunto de la ingeniería de pruebas, una práctica informática que se esfuerza por mejorar el rendimiento, englobándose en el diseño y la arquitectura de un sistema, antes incluso del esfuerzo inicial de la codificación (Globe Testing, 2011).

Esta prueba se realizó con el objetivo de satisfacer al cliente con respecto al tiempo que demora la extensión para generar reportes. Para ello se tuvieron en cuenta distintas versiones de LibreOffice (Anexo 3 y Anexo 4), varios reportes (Anexo 1 y Anexo 2) y dos sistemas operativos. Además se hicieron las modificaciones pertinentes a la extensión para permitir su correcto funcionamiento.

Tabla. 21 Prueba para evaluar el rendimiento

Sistema operativo	Versión de LibreOffice	Reporte	Tiempo de generación	Modificaciones
Ubuntu 12.04	3.5.7.2	Listado de usuarios.	2370 ms	
		Listado de proyectos	2110 ms	
	4.0.0.3	Listado de usuarios.	2348 ms	
		Listado de proyectos	2105 ms	
Nova Ligero 2013	3.5.7.2	Listado de usuarios.	3220 ms	• Cambiar ruta de almacenamiento. • Cambiar puntos por comas en el tamaño de letra.
		Listado de proyectos	3150 ms	
	4.0.0.3	Listado de usuarios.	2073 ms	
		Listado de proyectos	2069 ms	

3.5.4 Resultados de la pruebas

Las pruebas funcionales realizadas, utilizando el método de caja negra, se aplicaron principalmente a las interfaces del sistema apoyado en los casos de prueba basados. Estas dieron como resultado el satisfactorio cumplimiento de los requisitos definidos. Se realizaron

tres iteraciones de pruebas, donde se detectaron siete no conformidades en la primera iteración, hasta quedar resueltas en la tercera iteración.

A continuación se muestra el resultado:

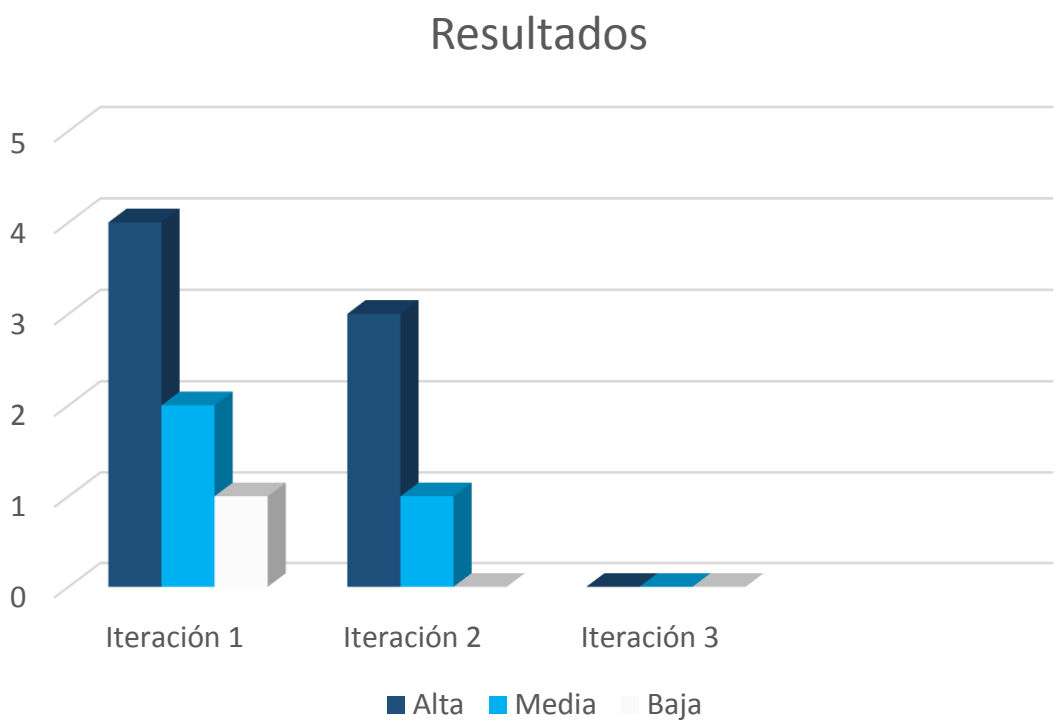


Fig. 10 Resultados de las pruebas de caja negra.

También se realizaron pruebas de rendimiento donde después de obtenidas las respuestas de la extensión se puede decir que la misma tiene un rendimiento adecuado en los escenarios de prueba. Los tiempos de generación de reportes pueden variar de acuerdo al sistema operativo, a la versión de LibreOffice y a la cantidad de datos que requiera el reporte a generar.

Además se realizaron las pruebas de aceptación en conjunto con el cliente para comprobar que el sistema cumple con los requisitos definidos, obteniéndose una buena aceptación por parte del cliente.

3.6 Consideraciones Parciales

En el presente capítulo se mostraron los elementos correspondientes a la implementación y las pruebas de la extensión. Se realizó el modelo de implementación con el propósito de mostrar los componentes del sistema y sus relaciones, a través del diagrama de componentes y diagrama de despliegue. Además se realizaron pruebas para validar el software, comprobando que los requisitos fueron implementados correctamente, a través del método de caja negra. Se llegó a la conclusión de que la extensión funciona correctamente y cumple con los requisitos definidos previamente.

CONCLUSIONES

El desarrollo de la solución cumple con los objetivos planteados en el inicio de la investigación obteniéndose una extensión de la herramienta LibreOffice para generar reportes dinámicos en conexión con base de datos de PostgreSQL.

- El estudio de herramientas generadoras de reportes permitió identificar características y funcionalidades necesarias para el desarrollo de la extensión.
- El estudio de la metodología ofreció una guía durante el proceso de desarrollo, obteniéndose así la Pila de Producto, la Pila de tareas del Sprint, y generando las Historias de Usuario.
- El desarrollo de la extensión permitió obtener una aplicación libre y completamente funcional para cualquier versión de LibreOffice, que genera reportes en múltiples formatos.
- Las pruebas realizadas permitieron comprobar el correcto funcionamiento de la extensión, a través de las pruebas de caja negra realizadas mediante los casos de prueba.

RECOMENDACIONES

Con el propósito de mejorar la propuesta realizada en este trabajo, se sugiere:

- Definir nuevos requerimientos para adicionar funcionalidades a la extensión desarrollada que permitan generar reportes integradores para las diferentes áreas de conocimiento.
- Realizar una capacitación a los especialistas generales que utilizarán la extensión.

REFERENCIAS BIBLIOGRÁFICAS

Abreu Medina, Aldis Joan, Lezcano Ramos, Miguel y Hernández Hernández, Yasmany. 2012. *Generador dinámico de reportes.* La Habana : Serie Científica de la Universidad de las Ciencias Informáticas, 2012.

Albaladejo, Xavier. 2014. Proyectos Ágiles.org. [En línea] 2014. http://www.proyectosagiles.org/beneficios-de-scrum.

Albaladejo, Xavier. 2013. Proyectos Ágiles.org. [En línea] 2013. http://www.proyectosagiles.org/que-es-scrum.

Almeira, Adriana y Perez Cavenago, Vanina. 2007. *Arquitectura de Software: Estilos y Patrones.* Argentina : s.n., 2007.

Awesome Inc. 2013. Design Patterns with UML. [En línea] 2013. http://design-patterns-with-uml.blogspot.com.ar/2013/02/singleton-pattern.html.

Brito Rodríguez, Julio César, y otros. 2013. *Módulo diseñador de modelos para el generador dinámico de reportes.* La Habana : Serie Científica de la Universidad de las Ciencias Informáticas, 2013.

Cáliz, Doris. 2007. *Generador de Reportes para Postgres.* Quito : s.n., 2007. Tesis de grado.

Camacho, Erika, Cardeso, Fabio y Nuñez, Gabriel. 2004. *Arquitecturas de Software - Guía de estudio.* 2004.

Canós, José H., Letelier, Patricio y Penadés, Ma Carmen. 2005. *Métodologías Ágiles en el Desarrollo de Software.* Valencia : DSIC -Universidad Politécnica de Valencia, 2005.

Castro Ruz, Raúl. 2011. *Decreto Ley 281.* s.l. : Gaceta Oficial de la República de Cuba, 2011. Gaceta Oficial de la República de Cuba.

Conferencia #2. Fase de Inicio, Modelo del Negocio, ISW 1. . s.l. : E.V.A. UCI, I. D. S.

Conferencia #3. Flujo de trabajo de Requerimientos. s.l. : E.V.A. UCI, I. D. S. .

Daniele, Marcela. 2007. *Teoría 11: El Arte de Modelar UML.* 2007.

Fernández Escribano, Gerardo. 2002. *Introducción a Extreme Programming: Ingeniería del Software II.* 2002.

García Suárez del Villar, Claudia, Brito Rodríguez, Julio César y Brizuela Figueredo, Clara Elena. 2013. *Herramienta para importar reportes.* La Habana : Serie Científica de la Universidad de las Ciencias Informáticas, 2013.

Globe Testing. 2011. globe TESTING. [En línea] 2011. http://www.globetesting.com/pruebas-de-rendimiento/.

González Duque, Raúl. 2010. *Python para todos.* 2010.

Ingeniería Revista de la Universidad de Costa Rica. **2012.** 2, Costa Rica : UNIVERSIDAD DE COSTA RICA, 2012, Vol. 22.

Jaspersoft Corporation. 2000-2014. JASPERSOFT COMMUNITY. *THE INTELLIGENCE INSIDE.* [En línea] 2000-2014. http://community.jaspersoft.com/project/jasperreports-library.

Larman, Craig. 1998. *Applying UML and Patterns. Prentice Hall.* 1998.

2014. LibreOffice. *The Document Foundation.* [En línea] 2014. https://es.libreoffice.org/caracteristicas/.

Martínez, Rafael. 2010. PostgreSQL-es. *Portal en español sobre PostgreSQL.* [En línea] 2010. http://www.postgresql.org.es/sobre_postgresql.

Módulo diseñador de modelos para el generador dinámico de reportes. **2013.** La Habana : Serie Científica de la Universidad de las Ciencias Informáticas, 2013.

Object Manegement Group, Inc. 1997-2014. OMG Object Management Group. [En línea] 1997-2014. http://www.omg.org/gettingstarted/what_is_uml.htm.

Pentaho Corporation. 2005-2014. pentaho. [En línea] 2005-2014. http://community.pentaho.com/projects/reporting/.

2013. PostgreSQL-es. [En línea] 2013. http://www.postgresql.org.es/sobre_postgresql....

Pressman, Roger. *El Proceso Unificado de Software.*

Pressman, Roger S. 2011. *Ingeniería del Software: Un Enfoque Práctico.* 2011.

Rodríguez Sala, Jesús Javier. 2003. *Introducción a la programación. Teoría y práctica.* San Vicente : Editorial Club Universitario, 2003. I.S.B.N.:84-8454-274-2.

Sanz, Juan C. y Guzmán Soriano, Jorge A. 2012. *Primeros Pasos con LibreOffice 3.3.* 2012.

Scribd Inc. 2014. Scribd. [En línea] 2014. http://es.scribd.com/doc/7884665/Arquitectura-de-Software-II-Diagrama-de-Componentes-y-Despliegue.

Sosa Porteiro, Marisel y Cobo Morales, Pedro H. 2005. REVISTA BETSIME. *LA REVISTA DEL EMPRESARIO CUBANO.* [En línea] 2005. http://www.betsime.disaic.cu/secciones/eco_enemar_07.htm.

Tangient LLC. 2014. Procesos de Software. [En línea] 2014. http://procesosdesoftware.wikispaces.com/METODOLOGIAS+PARA+DESARROLLO+DE+SOFTWARE.

The Document Foundation. 2014. LibreOffice. [En línea] 2014. http://es.libreoffice.org/caracteristicas/base/.

—. 2014. LibreOffice. [En línea] 2014. http://es.libreoffice.org/caracteristicas/extensiones/.

The Eclipse Foundation. 2013. eclipse. [En línea] 2013. http://www.eclipse.org/birt/phoenix/intro/.

—. 2013. eclipse marketplace. [En línea] 2013. http://marketplace.eclipse.org/.

The pgAdmin Devlopment Team. 2002-2011. *pg Admin III Documentation.* 2002-2011.

van Rossum, Guido. 2007. Guía de estilo del código Python. *Guía de Estilo para Código C.* 2007.

ANEXOS

Anexo 1: Reporte Listado de Usuarios

Fecha: 09/06/2014

Listado de Usuarios

Centro: Centro de Consultoría y Desarrollo de Arquitecturas Empresariales

Proyecto: Departamento de Gestión de Proyectos

id	firstname	lastname	mail
212	Alena	Santiesteban	alena@uci.cu
207	Yamila	González	ygonzalezg@estudiantes.uci.cu
216	José Alejandro	Lugo	jalugo@uci.cu
185	Marielis	Izquierdo	mmatias@uci.cu
1	Pedro	Piñero	pppyob@uci.cu
225	Iliana	Pérez	iperez@uci.cu
231	Rosel	Sosa	rsosag@uci.cu
215	Karina Mileisis	Torres	mileisis@uci.cu
209	Reinaldo	machado	reinaldomp@uci.cu
244	Yenisleydis	Rodríguez	yenirm@uci.cu
91	Antonio	Marrero Palomino	amarrero@uci.cu
251	Oniel	González	ojardines@estudiantes.uci.cu
54	Jorge	Infante	jorgeio@uci.cu
198	Isuel	Méndez	isuelm@uci.cu

Anexo 2: Reporte Listado de Proyectos

Fecha: 09/06/2014

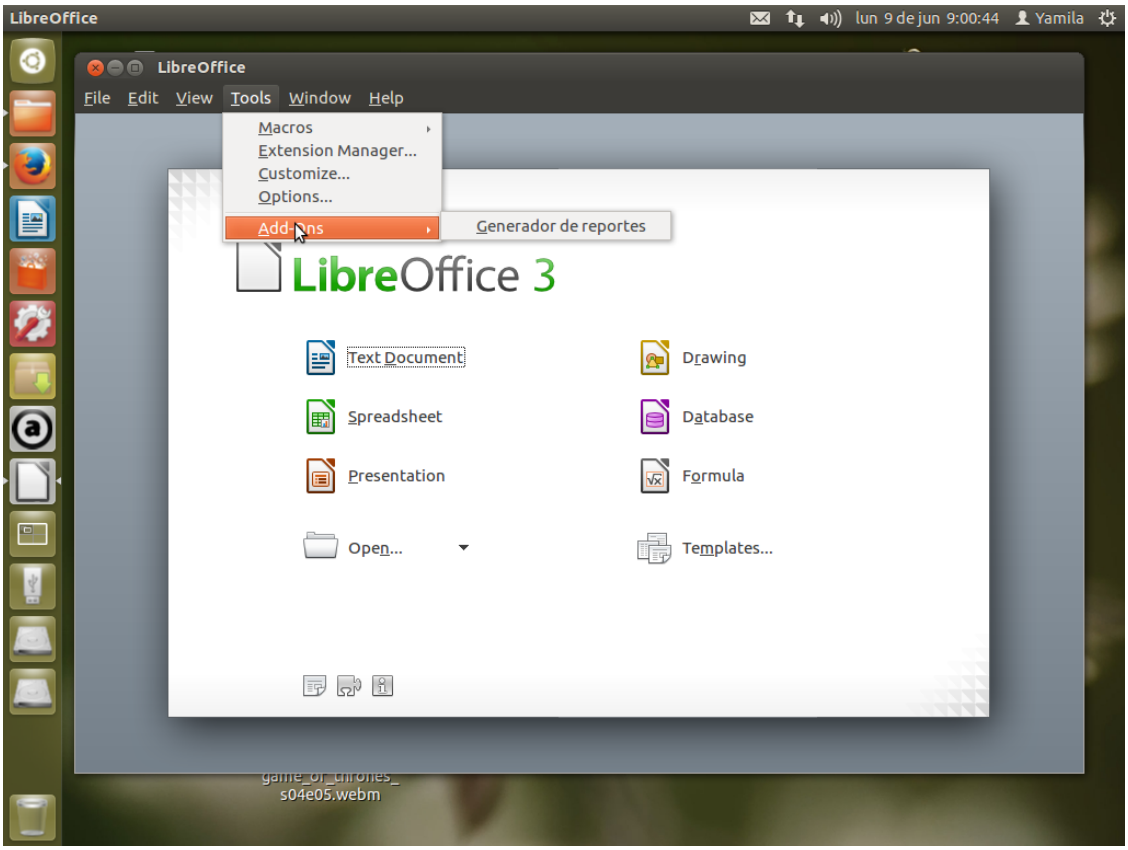
Listado de Proyectos

Centro: CDAE

Proyecto: Subproyecto gestión de datos

id	name
60	Subproyecto Gestión GESPRO
164	Módulo Economía y Planificación
163	Módulo Importaciones
162	Módulo Exportación y Sustitución de Importaciones
160	Módulo Intranet
153	Personalización GESPRO para RED Centros UCI
45	Replicador de Datos Reko v2.0
139	Subproyecto de Liberación
161	Módulo Colaboración Económica
25	Consejo de Dirección
135	Subproyecto gestión de datos
100	Modelación de Servicios en SOAML y MDA
136	Replicador de Datos REKO v3.0
112	Proyecto Solución de Software UDDI. AGN

Anexo 2: Extensión en LibreOffice 3.5.7.2



Anexo 3: Extensión en LibreOffice 4.0.0.3