Universidad de las Ciencias Informáticas

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

Título: Gestión de reportes para la Plataforma PTARTV

Autor: Reinaldo Pérez Castillo

Tutor: Ing. Yandy León Núñez

La Habana, junio 20 de 2011

"Año 53 de la Revolución"

"Si fuéramos capaces de unirnos… Qué hermoso y cercano sería el futuro"

Che

Dedicatoria

A mis padres por ser mis primeros maestros y hacer de mí la persona que soy… Por el amor infinito y la confianza que siempre me han dado.

A mi familia y los amigos de toda la vida.

Agradecimientos

A la Revolución Cubana por darme la maravillosa posibilidad de ser un profesional.

Agradezco a mi madre, por apoyarme, aconsejarme y ser mi principal inspiración en el estudio.

A mi tío Jorge por ser tan especial conmigo.

A mis amigos de toda la vida Osniel, Samuel, Saúl y Roilan por su apoyo durante estos años, Gracias mis hermanos.

A mi tutor Yandy, que en todo momento me ha apoyado, además de aportar gran parte de sus conocimientos para que este trabajo llegara a ser posible. Gracias por tu ayuda y sacrificio.

A los miembros del proyecto PTARTV por todo su apoyo.

A mis compañeros y amigos que durante estos cinco años, que siempre me brindaron su apoyo incondicional en cada obstáculo que se presentara. A todos ustedes muchas gracias.

A los profesores que me ayudaron durante estos años a formarme como profesional y brindarme su mano amiga. Gracias a todos…

Declaración de autoría

Declaro que soy el único autor del trabajo titulado:

Gestión de reportes para la Plataforma de Transmisión Abierta para Radio y Televisión. Autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales del mismo, con carácter exclusivo. Para que así conste firmo la presente a los ___ días del mes de _____del año _____.

_____	_____
Reinaldo Pérez Castillo	Ing. Yandy León Núñez
[Autor]	[Tutor]

Resumen

La información es poder y actualmente está formando parte inseparable de cualquier sistema informático. Los generadores de reportes son herramientas complementarias de los sistemas de información. Utiliza una especie de lenguaje transparente para el usuario por medio del cual este realiza consultas a la base de datos y obtiene información de ella en forma de reporte.

En el presente trabajo se muestran las bases de la implementación del subsistema de Reporte de la Plataforma de Transmisión Abierta para Radio y Televisión (PTARTV). Dicha plataforma tiene como principal objetivo la distribución de contenidos audiovisuales haciendo uso de los sistemas de transmisión de señales y empleándose la tecnología streaming, generándose un gran flujo de información.

Debido a este problema se realiza este trabajo, consistente en la implementación del subsistema de Reportes que permita controlar y organizar eficientemente la información dentro de la plataforma, además la generación de reportes que sean de ayuda en la toma de decisiones y en la realización de informes, con el fin de insertarla en el mercado como un producto de alta calidad.

Este documento recoge los resultados de la investigación realizada en el estudio y análisis de tecnologías, técnicas de programación y herramientas necesarias para el desarrollo del subsistema. Se describen las funcionalidades de las aplicaciones mostrándose a través del modelo de implementación y los diagramas de componente y despliegue.

Obteniéndose como resultado un Componente para la Gestión de Reportes compatible con aplicaciones desarrolladas en entorno de escritorio dentro de la Plataforma de Transmisión Abierta para Radio y Televisión encargada de visualizar, graficar y exportar los reportes desarrollados con anterioridad en el Generador Dinámico de Reportes.

Palabras claves:
Información, plataforma, reporte, componente

Figuras y Tablas

Índice de Figuras

Índice de Tablas

Índice

Introducción

Desde sus inicios, el hombre ha tenido la necesidad de comunicarse con los demás, de expresar sus pensamientos e ideas. Así también desde tiempos remotos se reconoce en el ser humano la necesidad de buscar, saber y obtener información ya sea creada, enunciada o almacenada.

Desde tiempos antiguos se le da gran importancia a la información, llegando al punto de que en las guerras antiguas la información era vital, por lo que empiezan a desarrollarse algoritmos de encriptación para mantenerla segura y así pudiera viajar sin ser interceptada por personas que pudieran beneficiarse de ella. Si se desea maximizar la utilidad que posee la información, se debe manejar de forma correcta y eficiente.

Aunque la información se encuentre formando parte del trabajo diario, se debe saber que ésta no es gratis, debido a que un uso indebido de la misma puede incurrir en una pérdida parcial o total de la información almacenada y las entidades se verían obligadas a dedicar tiempo y esfuerzo de trabajo para su recuperación derivando en pérdidas millonarias. El uso de la información es estrictamente estratégico para posicionar de forma ventajosa la empresa dentro de un negocio.

En la actualidad los Sistemas de Información (SI) y las Tecnologías de la Información y la Comunicación (TIC) han cambiado la forma en que operan las organizaciones actuales. A través de su uso se logran importantes mejoras, pues automatizan los procesos operativos, suministran una plataforma de información necesaria para la toma de decisiones y lo más importante, su implantación logra ventajas competitivas o reducen la competencia con los rivales.

Actualmente se habla mucho de la importancia que tiene la información dentro de una organización, catalogándola inclusive en muchos casos como uno de los principales activos. Sin embargo existen limitaciones que hacen imposible el cumplimiento de este importante objetivo, en muchas de ellas ya sea por desconocimiento tecnológico o falta de recursos. Un sistema de información lo conforman un conjunto de componentes capaces de realizar operaciones de procesamiento con datos para generar reportes.

En todo sistema de información es fundamental contar con herramientas complementarias cuya función sea la de crear reportes a partir de la información almacenada en una base de datos. En este sentido, los generadores de reportes son herramientas adicionales a los sistemas de información.

Los generadores de reportes tienen definido un mecanismo para realizar consultas a la base de datos y obtener información de estas en forma de reporte. A diferencia de las consultas tradicionales, con los gestores de reportes se tiene mayor control sobre el proyecto que tendrá la salida de la información, es decir, el usuario puede definir el aspecto que tendrá el reporte que desea generar.

En la actualidad existen a nivel mundial generadores de reportes comerciales como Crystal Reports, ReporView y Stimulsoft Reports Designer Web. El problema con estos es que su costo es demasiado elevado, y por esta razón no pueden ser explotados por la mayoría de los usuarios que necesitan una herramienta de este tipo.

Cuba no ha estado ajena al acelerado desarrollo de la informática y las comunicaciones en las últimas décadas, así como el gran auge que han tenido las TIC. Debido a esto se creó en septiembre del 2002 la Universidad de las Ciencias Informáticas (UCI), la cual es la primera universidad surgida al calor de la batalla de ideas, su objetivo fundamental es la formación integral de profesionales de la informática comprometidos con la revolución para llevar a cabo una informatización en la sociedad cubana. La UCI constituye un pilar primordial en el desarrollo de software tanto a nivel nacional como internacional.

La UCI cuenta con varias facultades destinadas a la realización de software en distintas ramas. Formando parte de la Facultad 6 se encuentran los centros de desarrollo DATEC[1] y GEYSED[2]. La Plataforma de Transmisión Abierta para Radio y Televisión (PTARTV) perteneciente al departamento de Señales Digitales del centro de desarrollo GEYSED, es un proyecto de investigación y desarrollo enfocado en la realización de un producto de calidad para mejorar las transmisiones tanto televisivas como radiales y automatizar los procesos que se llevan a cabo dentro de una televisora.

[1] **DATEC:** Centro de Tecnologías de Gestión de Datos

[2] **GEYSED:** Centro de desarrollo Geoinformática y Señales Digitales

En estos momentos el principal objetivo de la plataforma es lograr mejorar las trasmisiones que son realizadas en la UCI por la Dirección de Televisión Universitaria (DTU). Actualmente todos los receptores en la universidad reciben la señal televisiva de forma analógica y lo que se concibe con esta plataforma es lograr llegar a la televisión ip dentro de la universidad y la unificación de todos los servicios prestados en un mismo producto. La misma cuenta con 10 subsistemas: Transferencia, Gestión de Medias, Web, Programación, Transmisión, Radial, Seguridad, Monitoreo, Equipamiento y Producción.

Durante la realización y la puesta en ejecución de estos subsistemas y después de un número de entrevistas realizadas al personal de la DTU, se encontraron un conjunto de deficiencias significativas en el manejo de la información las cuales influyen de manera directa en la toma de decisiones dentro de la televisora. Durante las transmisiones y el monitoreo de las señales televisivas no se tiene control de las medias que causaron error para ser reprogramadas en futuras programaciones, ni se tiene un control estricto sobre la cantidad de errores que se producen en cada uno de los canales de la plataforma por un mes, semana o día determinado.

En la conversión de las medias que se realiza en el subsistema de Transferencia no se tiene un registro de los errores producidos ni de la cantidad de medias convertidas durante el mes, así como el tiempo total de conversión por espacios (documentales, series, películas, etc.). En el subsistema de Producción no se lleva una trazabilidad de las tareas cumplidas durante una semana o mes de un determinado trabajador.

En el subsistema de Gestión de Medias se debe llevar un control de las medias subidas al servidor con éxito así como las que causaron error. Además en el subsistema Radial no se controlan todas las interrupciones, así como el usuario que estaba trasmitiendo a la hora de dicha interrupción.

Con el objetivo de darle solución a lo antes planteado es necesaria la incorporación de un subsistema para la gestión de reportes que sea de ayuda para darle solución a estos errores, y además represente una fuente viable y de apoyo para la toma de decisiones dentro de la PTARTV.

Esta solución repercutirá indudablemente en un ahorro de tiempo para el personal que haga uso de estos subsistemas en la plataforma. Los reportes o informes serán generados a través de una herramienta que

brindará la posibilidad de exportar los mismos en formato .pdf. Por tanto, se define para la investigación el siguiente problema a resolver:

¿Cómo dar seguimiento al funcionamiento de la plataforma PTARTV para apoyar la toma de decisiones?

Se propone como objetivo general implementar un subsistema que gestione los reportes en la plataforma PTARTV y constituya un apoyo para la toma de decisiones.

En este trabajo es objeto de estudio los procesos para la gestión y generación de reportes y el campo de acción se centra en los procedimientos y técnicas para la generación de reportes en la plataforma PTARTV.

Se propone como idea a defender la implementación de un subsistema que gestione los reportes en la plataforma PTARTV y constituya una herramienta de apoyo para el control y la toma de decisiones.

Para dar cumplimiento al objetivo propuesto se definen las siguientes tareas de la investigación:

1. Describir la evolución histórico-lógica de los procedimientos y técnicas para la generación de reportes.
2. Fundamentar la utilización del Generador Dinámico de Reportes del Departamento DATEC como base para el desarrollo.
3. Caracterizar las herramientas y tecnologías de desarrollo a utilizar en la construcción del software.
4. Desarrollar artefactos y documentación según la metodología de desarrollo seleccionada.
5. Implementar las funcionalidades del subsistema de Reportes.
6. Validar la propuesta de solución.

Obteniéndose como posibles resultados:

1. Documento de informe de la investigación.

2. Documentación y artefactos generados según la metodología para rol de Implementador.

- Modelo de implementación

- Diagrama de despliegue
- Diagrama de componentes.
- Manual de usuario.

3. Subsistema de Reportes para la PTARTV.

Para obtener los conocimientos necesarios que hagan posible el cumplimiento del objetivo trazado en el trabajo, se lleva a cabo una investigación en la que se utilizan algunos de los métodos científicos existentes, tanto teóricos como empíricos.

Los **Métodos Teóricos** utilizados para dar cumplimiento a las tareas previstas son:

- **Analítico-Sintético:** Permite analizar la información y dividir la investigación en partes durante su desarrollo y luego sintetizarla a través de las relaciones existentes, obteniendo ideas claras y concisas.
- **Histórico-Lógico:** Ayuda a caracterizar las soluciones existentes entendiendo su evolución y facilitando aprovechar puntos en común y conceptos teóricos que sean de relevancia.
- **Modelación:** En el desarrollo de la investigación se utiliza este método para realizar modelos que ofrecen la posibilidad de crear abstracciones. Se pone en práctica en el trabajo al crear el modelo de implementación.

Los **Métodos Empíricos** permiten extraerse de los fenómenos, analizado la información que se necesita a través de observaciones, entrevista o encuestas. Por consiguiente se aplicaron en el seguimiento de la presente investigación los siguientes:

- **Observación:** Facilita conocer el panorama real de una situación mediante la percepción directa. Con la utilización de este método se determinaron los rasgos imprescindibles en el desarrollo del subsistema de Reportes de la Plataforma de Transmisión Abierta para Radio y Televisión.
- **Entrevista:** Se le realiza al personal de la DTU para definir los reportes necesarios en cada uno de los departamentos de esta. Para esto se ha seleccionado una población de 15 personas entre los cuales se encuentran trabajadores y directivos de la dirección de televisión universitaria (DTU). Para el desarrollo de las entrevistas se toma como muestra 7 trabajadores y 2 directivos. La

muestra representa el 60 por ciento de la población la cual fue seleccionada de forma intencional empleando técnica de muestreo no probabilístico. (Ver anexo 1)

Capítulo 1. Fundamentación teórica

Introducción

En este capítulo se abordan una serie de conceptos que giran alrededor del objetivo general de la investigación así como el problema planteado. Se realiza una breve descripción de los conceptos asociados a la comprensión del problema. Realizándose un análisis de algunas soluciones existentes a nivel internacional y nacional que son de interés por tener estrechos lazos con el alcance de esta investigación.

1.1. Conceptos asociados al dominio del problema

Para lograr una mejor comprensión del dominio del problema de la presente investigación, se realizará una descripción de los principales conceptos.

1.1.1. Plataforma de Transmisión Abierta para Radio y Televisión.

PTARTV es un producto destinado a automatizar los procesos desarrollados dentro de una televisora, posee funcionalidades que facilitan la programación, la gestión de las medias, el monitoreo de las transmisiones televisivas y radiales, la transferencia y el entorno web de la aplicación. Este producto permitirá experimentar un salto considerable en la calidad de las transmisiones televisivas y radiales que son realizadas por la dirección de televisión de la UCI.

La Plataforma PTARTV en cuanto a sus tecnologías y lenguajes de desarrollo está dividida en dos grupos, los subsistemas desarrollados sobre tecnología web y los desarrollados para computadoras personales. El primero de estos grupos hace uso del lenguaje de programación PHP con el framework de desarrollo Symfony y la librería JavaScript Ext Js, mientras que el segundo fue implementado con C++ y el framework de desarrollo QT Creator.

1.1.2. Reporte

El reporte es aquel documento que se utilizará cuando se quiera informar o dar noticia acerca de una determinada cuestión. Un reporte es un documento generado por un sistema, que presenta datos

relevantes previamente guardados ya sea en un sistema gestor de base datos u otro, de tal manera que se vuelvan útiles para los fines de una entidad. (1)

A diferencia de un formulario, los datos dentro de un reporte no pueden ser manipulados o modificados directamente, sino que tienen que ser afectados en alguna otra parte del sistema para que se reflejen los cambios una vez que el reporte sea generado nuevamente. (2)

Un reporte es generado dinámicamente, cuando cada vez que es llamado o invocado desde el sistema, el reporte actualiza la información a los datos más recientes disponibles. (2)

1.2. Análisis de otras soluciones existentes

En el mundo del software es común encontrarse con aplicaciones similares, derivadas unas de otras, donde las diferencias solo responden a necesidades muy específicas de los clientes. Muchas de estas soluciones son hechas a la medida, enfocadas a objetivos específicos, cumpliendo con los requerimientos de los usuarios. A continuación se analizan algunos de los procesos para la gestión y generación de reportes más significativos existentes tanto a nivel internacional como nacional, los cuales son de interés por tener estrechos lazos con el alcance de esta investigación.

1.2.1. Ámbito internacional

- **Crystal Reports**

Crystal Reports es la solución de elaboración de informes más usada en el mundo, con más de 8 millones de copias vendidas. (3)

Es una aplicación de inteligencia empresarial utilizada para diseñar y generar informes desde una amplia gama de fuentes de datos (bases de datos). Varias aplicaciones, como Microsoft Visual Studio, incluyen una versión OEM[3] de Crystal Reports como una herramienta de propósito para generar los informes y reportes. Crystal Reports se convirtió en el escritor de informes estándar cuando Microsoft lo liberó con Visual Basic. (4)

[3]**OEM:** (Original Equipment Manufacture) en español "fabricante de equipamiento original" empresa fabricante de productos que luego son comprados por otra empresa y vendidos bajo la marca de la empresa compradora.

Características

- Transforma rápidamente cualquier fuente de datos en contenido interactivo.
- Integra estrechamente capacidades de diseño, modificación y visualización en aplicaciones .Net y Java.
- Permite a los usuarios finales acceder e interactuar con los reportes a través de portales web, dispositivos móviles y documentos de Microsoft Office.

El más completo acceso a datos: Crystal Reports provee más opciones de conectividad a datos que cualquier otra herramienta. Incluye más de 30 drivers para acceso a bases de datos relacionales (Incluyendo sistemas Oracle, IBM DB2 y Microsoft SQL Server). También puede acceder a datos personalizados a través de JavaBeans para una conectividad más flexible. (5)

Diseño integral y opciones de formato: La tecnología flexible de diseño de Crystal Reports provee control completo sobre el acceso y la presentación de los datos en los reportes. Posibilita elegir entre más de 100 opciones de formato, incluyendo parámetros, mapas, tablas cruzadas, gráficos e hipervínculos, para incrementar el impacto de sus reportes. También se incluyen más de 160 formulas, funciones y operadores para un control completo de la presentación de los datos. (5)

Productividad: El nuevo Repositorio Crystal le permite almacenar elementos clave de los reportes de su organización, tales como objetos de texto, imágenes, sentencias SQL y funciones personalizadas. Gracias a este repositorio central, se pueden reutilizar estos objetos en múltiples reportes. Este almacén centralizado de objetos permite minimizar los esfuerzos de mantenimiento de sus reportes y al mismo tiempo ser más productivo en el diseño de reportes nuevos. (5)

Ventajas

- Permite exportar los informes a diferentes formatos (Crystal Reports (rpt), Microsoft Excel, html, Microsoft Word, PDF, Rich Text Format, text).
- Permite el uso de diferentes bases de datos.
- Se puede utilizar cualquier lenguaje, por ejemplo SQL Reportin Service.
- Personaliza los informes en tiempo de ejecución.

- Gran capacidad de visualización y análisis de datos. (6)

Entre los puntos que provocan que Crystal Reports no sea una solución factible a utilizar por la Plataforma de Transmisión Abierta para Radio y Televisión a pesar de ser la solución de elaboración de informes más usada a nivel mundial están, que cada entidad que vaya a hacer uso de esta aplicación para realizar diseños y generar reportes debe pagar por utilizarlo debido a que es un software privativo.

- **ReporView**

Los controles ReportViewer se utilizan para alojar los informes creados en Microsoft Visual Studio. Hay dos versiones del control: el control de servidor web ReporView para páginas ASP.NET y el control de Windows Forms ReporView que puede utilizarse en aplicaciones de Windows Forms. Visual Studio determinará qué versión debe utilizarse según el tipo de proyecto que use para incrustar el control. (7)

Características

Las dos versiones del control ReportViewer admiten las siguientes características:

- Áreas de vista para mostrar un informe, una barra de herramientas y un mapa del documento. La barra de herramientas es configurable y proporciona características en tiempo de ejecución para admitir la exploración en un informe de varias páginas, búsqueda, impresión y exportación. Se mostrará un mapa del documento si agrega uno explícitamente al informe.
- Propiedades que le permitirán configurar el modo de procesamiento, las áreas de vista y la barra de herramientas.
- Compatibilidad con los modos de procesamiento local y remoto para controlar dónde y cómo se procesan los informes. El modo de procesamiento local recupera y combina un conjunto de datos existente en un diseño de informe y representa el informe mediante la funcionalidad de procesamiento interna al control. El procesamiento remoto representa un informe publicado en un servidor de informes de Microsoft SQL Server Reporting Services.
- Interfaces de programación que permiten personalizar, configurar e interactuar con el control mediante código, así como cambiar los orígenes de datos que utiliza ReportViewer en tiempo de ejecución. (7)

Entre los puntos que provocan que ReporView no sea una solución factible a utilizar, es al igual que Crystal Reports para hacer uso de esta aplicación se debe estar sujeto a Microsoft Visual Studio que es propiedad de Microsoft y debe pagar por concepto de licencias para poder utilizarlo.

- **JasperReports**

JasperReports es una herramienta de creación de informes de código abierto y uno de los más populares motores de informes. Está escrito completamente en Java y es capaz de utilizar los datos procedentes de cualquier tipo de fuente de datos y presentar los documentos con precisión de píxel que se puede ver, imprimir o exportar en una variedad de formatos de documentos incluyendo HTML, PDF, Excel, OpenOffice y Word. (8)

Puede ser usado en gran variedad de aplicaciones de Java, incluyendo J2EE o aplicaciones web, para generar contenido dinámico. Su propósito principal es ayudar a crear documentos de tipo páginas, preparados para imprimir en una forma simple y flexible. Es una poderosa y flexible solución de código abierto para la generación y gestión de informes. (5)

El principal objetivo de este proyecto es facilitar la construcción de documentos con contenido dinámico y su visualización en diferentes formatos (PDF, HTML y XML). También existe la posibilidad de exportar los informes a formato PDF, a una hoja de cálculo de Excel o documento Word. (9)

Agrega dinamismo a los reportes en JasperReports con DynamicJasper:

DynamicJasper (DJ) es un proyecto open-source que permite al desarrollador crear rápidamente una gran variedad de reportes a través de una intuitiva API[4] escrita en Java. Esta permite definir programáticamente las columnas, grupos, totales, gráficos (charts), sub-reportes. La API maneja todo lo relacionado con la diagramación y posicionamiento de los elementos del reporte haciendo el proceso de diseño fácil y automático. (9)

DynamicJasper tiene como objetivo abarcar el 99% de los reportes que se basan en columnas como así también los que tienen grupos (cortes de control). La API permite agregar variables en las cabeceras y pie de las columnas y grupos con operaciones tales como *suma*, *contar*, etc. Se puede definir en tiempo de

[4]**API:** (Application Programming Interface) en español “interfaz de programación de aplicaciones”

ejecución el orden de aparición de las columnas, los grupos, las variables, los estilos, sub-reportes, etc. (9)

JasperReports a pesar de ser una herramienta open-source y estar desarrollada en software libre, no cumple con los requisitos de tecnologías de desarrollo definidas por los arquitectos de la Plataforma de Transmisión Abierta para Radio y Televisión debido que se encuentra desarrollado en Java, por lo que no es una alternativa viable a utilizar.

1.2.2. Ámbito nacional

- **Dynamicreport**

Dynamicreport es un proyecto comunitario rectorado por la línea JEE del Centro de Consultoría y Desarrollo de Arquitecturas Empresariales (CDAE), su principal objetivo es crear un reporteador dinámico que pueda abstraer al programador de la fuente de datos. Está basado en el funcionamiento de JasperReport y DynamicJasper. El proyecto es web, y puede ser utilizado de forma independiente o como un componente del framework Dalas. Soporta como acceso a datos JDBC, Hibernate y XML. Permite generar reportes en archivos pdf, csv, xls entre otros. (10)

Características

- Tipo de aplicación: Aplicación web
- Lenguaje base: Java
- Framework bases desarrollo: Spring
- Gestor de base de datos: PostgreSQL
- Sistema operativo: Multiplataforma: soportada en múltiples sistemas operativos
- Herramientas de modelación: Visual Paradigm
- Entorno de desarrollo: Eclipse (10)

Dynamicreport al igual que JasperReports se encuentra desarrollado en Java, por lo que no es una alternativa a utilizar para la realización del módulo de gestión de reportes para la Plataforma de Transmisión Abierta para Radio y Televisión.

- **Generador Dinámico de Reportes Dinámicos (PATDSI GDR)**

Es una aplicación web que tiene como objetivo generar reportes de forma rápida, interactiva y con una amplia gama de alternativas para los usuarios. La extensión en su uso puede estandarizar la generación de reportes en diferentes aplicaciones independientemente del Sistema Gestor de Base de Datos que utilicen ya sea MySQL, Oracle o PostgreSQL. (11)

El Generador Dinámico de Reportes es una herramienta multiplataforma, que da cobertura al ciclo de vida completo de un reporte: diseño, visualización, almacenamiento, eliminación, etc. Está formado por 4 módulos: Diseñador de Modelos, Diseñador de Reportes, Diseñador de Consultas y Visor de Reportes. (12)

Es un proyecto desarrollado en el centro DATEC, varios son los clientes que actualmente hacen uso del producto, entre ellos los centros CEIGE[5] y Calisoft de la UCI, entre otros, al tiempo que se encuentra en 11 de las soluciones convenidas como parte de la X Comisión Mixta Cuba-Venezuela. (12)

El Generador permite a los usuarios, entre otras opciones, abstraerse a los conocimientos relacionados con los gestores de base de datos, agilizar la toma de decisiones y generar reportes en varios formatos como pdf y html y con gran variedad de opciones en su diseño, marcando una diferencia entre los reportes tradicionales y los reportes dinámicos, objetos de este producto.

El Generador Dinámico de Reportes (PATDSI GDR) Es una aplicación web desarrollada con PHP haciendo uso del framework de desarrollo Symfony y la librería JavaScript Ext Js, este es uno de los principales aspectos que se tuvieron en cuenta para seleccionar esta herramienta como base para el desarrollo de este trabajo, puesto que coincide con los requisitos de tecnologías de desarrollo definidas en la PTARTV y no se debe incurrir en tiempos adicionales en capacitación del personal. Además es una aplicación desarrollada en software libre y en cuanto a la realización de reportes la más usada y centralizada a nivel de la comunidad UCI.

Dicha aplicación permite como ya se había planteado dar cobertura al ciclo de vida completo de un reporte, también permite generar los reportes haciendo consultas a los datos almacenados desde la

[5] **CEIGE:** Sistema de Gestión de Incidencias.

misma aplicación haciendo uso de las tablas de una base de datos antes cargada, esto es de gran utilidad a la hora de generar reportes.

La principal desventaja de su utilización es que al estar desarrollado en entorno web es factible su utilización en los subsistemas web dentro de la PTARTV, no siendo así para los subsistemas desarrollados en entorno de escritorio que para hacer uso del mismo necesitan abrir el reporteador desde un navegador web.

1.3. Conclusiones parciales

En este capítulo se ha realizado una valoración de las soluciones existentes tanto a nivel mundial como nacional y se expone una fundamentación de la selección del Generador Dinámico de Reportes (PATDSI GDR) como base para la generación de reportes dentro de la PTARTV. Una selección racional de estas soluciones permitirá obtener un sistema de alta calidad.

Capítulo 2. Tendencias y tecnologías actuales

Introducción

En este capítulo se hace una breve referencia de las técnicas y lenguajes actuales de programación, así como las tendencias tecnológicas, además de las metodologías de desarrollo de software y el patrón arquitectónico a utilizarse durante la realización del presente trabajo de diploma.

2.1. Paradigmas de programación.

Un paradigma de la programación puede definirse como un conjunto de ideas que indican o define una forma de programar adoptada por una comunidad de programadores. A través de ellos se pueden obtener diversas formas de "ver" y "pensar" un programa antes de escribirlo. Estas ideas han evolucionado a la par de los lenguajes de programación. Un paradigma de programación está delimitado en el tiempo en cuanto a aceptación y uso, ya que nuevos paradigmas aportan nuevas o mejores soluciones que la sustituyen parcial o totalmente.

Los paradigmas de programación más comunes son:

- Imperativo o por procedimientos.
- Lógico
- Funcional
- Estructurado
- Orientado a objetos

2.1.1. Programación Orientada a Objetos (POO)

Es un estilo de programación en que cada programa es visto como un objeto, se forma por una serie de componentes, que cooperan para realizar las acciones de la aplicación completa. La programación orientada a objetos expresa un programa como un conjunto de estos objetos, que colaboran entre ellos para realizar tareas. Esto permite hacer los programas y módulos más fáciles de escribir, mantener y reutilizar.

La programación orientada a objetos introduce conceptos, que superan y amplían los ya hasta el momento conocidos. Entre otros se destacan los siguientes:

Objeto: entidad provista de un conjunto de propiedades o atributos (datos) y de comportamiento o funcionalidad (métodos).
Clase: definiciones de las propiedades y comportamiento de un tipo de objeto concreto. La instanciación es la lectura de estas definiciones y la creación de un objeto a partir de ellas.
Herencia: las clases no están aisladas, sino que se relacionan entre sí, formando una jerarquía de clasificación. Se pueden crear nuevos objetos a partir de otros, los cuales heredan las propiedades y el comportamiento de todas las clases a las que pertenecen
Abstracción*:* Es un término externo al objeto, que controla la forma en que es visto por los demás.
Polimorfismo: Habilidad de llamar de la misma forma a dos métodos o procedimientos diferentes. El programa decide qué método utilizar según los parámetros que recibe.

Existen numerosos lenguajes de programación orientados a objetos dentro de los cuales se destacan los siguientes:

- C++
- C#
- Delphi
- Java
- Perl
- PHP (a partir de su versión 5)
- Python
- VB.NET

2.1.2. Ventajas de la Programación Orientada a Objetos.

Aunque el paradigma de POO no es infalible y tiene desventajas estás son superadas por las ventajas que brinda. Por tal razón serán estás últimas el centro de atención.

- La POO constituye una evolución de otros paradigmas incorporando nuevos conceptos que le otorgan gran relevancia.
- Incorpora en su entorno de ejecución mecanismos tales como el polimorfismo y el envío de mensajes entre objetos.
- Fomenta la reutilización y extensión del código siendo de gran ayuda para los implementadores.
- Permite la creación de programas más complejos y agiliza su desarrollo.
- Relaciona el software con el mundo real y permite su evolución.
- Facilita el mantenimiento de los sistemas y el trabajo en equipo.

2.2. Lenguaje de programación

Un lenguaje de programación es un idioma artificial, compuesto por un conjunto de símbolos, reglas semánticas y sintácticas que permiten la comunicación entre una persona y el ordenador.

Pueden clasificarse en dos grupos principales:

- Lenguajes de bajo nivel: Se acercan al funcionamiento del computador. El lenguaje representativo es el código máquina. Las instrucciones en este lenguaje están formadas por cadenas binarias (0 y 1). Puede citarse el lenguaje ensamblador el cual trabaja directamente con los registros de la computadora.

- Lenguajes de alto nivel: Son fáciles de aprender porque están formados por elementos del lenguajes y al usarlos puede dar la sensación de que las computadoras los comprenden.

2.2.1. Web

2.2.1.1. Lenguajes de programación del lado del cliente

Un lenguaje del lado cliente es totalmente independiente del servidor. En el caso de una aplicación web esto permite que la página pueda ser albergada en cualquier sitio. Pero la página no se verá bien si el ordenador cliente no tiene instalado la versión correcta del navegador y los plugins[6] adecuados.

[6] Programas que se relacionan con un sistema con el objetivo de añadirle funcionalidades.

- **HTML**

Es un lenguaje sencillo, permite definir documentos de hipertexto a base de ciertas etiquetas que marcan partes del documento dándoles una estructura o jerarquía. Presenta el texto de una manera estructurada y agradable, con enlaces (hyperlinks) que conducen a otros documentos o fuentes de información relacionadas, y con inserciones multimedia (gráficos, sonido, video). El lugar donde se encuentra esta información puede ser el mismo documento o cualquier otro lugar de Internet. (13)

- **JavaScript**

JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. (14)

2.2.1.2. Lenguajes de programación del lado del servidor

La Programación del lado del servidor es una tecnología que consiste en el procesamiento de una petición de un usuario mediante el reconocimiento, ejecución e interpretación de un script en el servidor para generar páginas HTML dinámicamente como respuesta, las cuales son comprensibles para el cliente. Un lenguaje que se ejecuta en el lado del servidor es independiente del cliente por lo que es mucho menos rígido respecto al cambio de un navegador a otro o respecto a las versiones del mismo.

- **PHP**

PHP es un robusto lenguaje de programación del lado del servidor. Fue diseñado por Rasmus Lerdorf en 1994 como un CGI escrito en C que permitía la interpretación de un número limitado de comandos. Dada la aceptación del primer PHP y de manera adicional, su creador diseñó un sistema para procesar formularios al que le atribuyó el nombre de FI (Form Interpreter) y el conjunto de estas dos herramientas, sería la primera versión compacta del lenguaje. (15)

Dado que es un lenguaje abierto dando la posibilidad de modificar el código fuente y añadir nuevas funcionalidades ha tenido una rápida evolución y desarrollo. Actualmente se encuentra en su versión PHP5. (16)

Una de sus grandes potencialidades es su soporte para una gran cantidad de bases de datos. De las cuales se pueden mencionar InterBase, MySQL, Oracle y PostgreSQL, entre otras. PHP también ofrece la integración con varias bibliotecas externas, que dan al desarrollador la posibilidad de realizar cualquier tarea, desde generar documentos en pdf (Portable Document Format) hasta analizar código XML. (16)

2.2.2. Escritorio

- **C++**

El comité para el estándar ANSI[7] fue formado en 1983 con el objetivo de crear un lenguaje uniforme a partir del C original, desarrollado por Kernighan y Ritchie en 1972. El lenguaje C++ se comenzó a desarrollar en 1980. Su autor fue B. Stroustrup. Al comienzo era una extensión del lenguaje C que fue denominada C con clases.

En la actualidad, el C++ es un lenguaje versátil, potente y general. Su éxito entre los programadores profesionales le ha llevado a ocupar el primer puesto como herramienta de desarrollo de aplicaciones. El C++ mantiene las ventajas del C en cuanto a riqueza de operadores y expresiones, flexibilidad, concisión y eficiencia. Además, ha eliminado algunas de las dificultades y limitaciones del C original. La evolución de C++ ha continuado con la aparición de Java, un lenguaje creado simplificando algunas cosas de C++ y añadiendo otras, que se utiliza para realizar aplicaciones en Internet. (17)

Ventajas:

- Lenguaje de programación orientado a objetos.
- Lenguaje muy didáctico, gracias a este lenguaje se puede aprender muchos otros lenguajes con gran facilidad, como C#, Java, Visual Basic, PHP, entre otros.
- Es muy potente en lo que se refiere a creación de sistemas complejos, un lenguaje muy robusto.
- Es un lenguaje muy empleado, existen muchos tutoriales en línea, libros, códigos fuentes abiertos, existe amplia bibliografía sobre el tema.

[7] American National Standards Institute, del español Instituto Nacional Americano de Estándares

Desventajas:

- Manejo de punteros y memoria respecto a ello. Claro, esta también es una gran ventaja porque permite un mejor control de la memoria y una buena administración de recursos de computadora, pero la inexperiencia de los desarrolladores o la pérdida de costumbre con este tipo de variables.
- No es recomendable para desarrollo de páginas web.
- Existen muchos entornos de programación para C++y no existen estándares para ello. De manera que puedes encontrar C++ para Unix/Linux, C++ para Windows. Además, en cada SO encuentras diferentes IDEs de desarrollo, y también encuentras IDEs para desarrollo de aplicaciones Qt para Unix/Linux, Borland C++ Builder y Visual Studio C++ para Windows.

2.3. Framework de desarrollo

Un Framework (“Marco de Trabajo” o “Marco de Desarrollo”) es un conjunto de librerías y componentes de probada solvencia, junto con una documentación y metodología de uso, que permite diseñar, construir e implantar aplicaciones corporativas de forma más uniforme, rápida, y con mayor calidad. (18)
Un framework simplifica el desarrollo de una aplicación mediante la automatización de algunos de los patrones utilizados para resolver las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al desarrollador a crear código más legible y más fácil de mantener. Por último, un framework facilita la programación de aplicaciones, ya que encapsula operaciones complejas en instrucciones sencillas. (18)

2.3.1. Symfony

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones web. En primer lugar separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de un proyecto web complejo. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación web. (18)

Características

- Fácil de instalar y configurar.
- Independiente del sistema gestor de bases de datos.
- Sencillo de usar en la mayoría de los casos, pero lo suficientemente flexible como para adaptarse a los casos más complejos.
- Fácil de extender, lo que permite su integración con librerías desarrolladas por terceros.

2.3.2. Qt

Qt es una biblioteca multiplataforma para el desarrollo de interfaces gráficas de usuario y también para el desarrollo de programas sin interfaz gráfica como herramientas de la consola. Qt es utilizada en entorno de escritorio para sistemas operativos como GNU/Linux o Windows, entre otros. Qt utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser utilizado en varios otros lenguajes de programación a través de bindings.

Presenta un amplio apoyo. El API de la biblioteca cuenta con métodos y ejemplos muy buenos de cómo hacer uso del framework, así como uso de XML. Es producido por la división de software Qt de Nokia, es desarrollado bajo licencia LGPL y liberado como software libre y de código abierto.

2.3.3. Qt Creator

Qt Creator es un IDE (Entorno de Desarrollo Integrado) creado por Trolltech para el desarrollo de aplicaciones con las bibliotecas Qt. Está diseñado para el trabajo con bibliotecas Qt.

Entre las principales características de Qt Creator se pueden mencionar las siguientes:

- Resaltado de sintaxis y completado de código
- Control de código estático y consejos de estilo a medida que se escribe
- Apoyo a la refactorización de código fuente
- Ayuda sensible al contexto
- Coincidencia de paréntesis y los modos de selección de paréntesis
- Capacidades de edición avanzada

Además de las características antes mencionadas se debe resaltar que las bibliotecas Qt y el IDE Qt Creator son multiplataforma y las aplicaciones que se apoyan en ellas tienen buena respuesta y un consumo de recursos aceptable.

2.4. Sistema Gestor de Base de Datos

Un Sistema de Gestión de Base de Datos (SGBD) es un conjunto de programas que permite a los usuarios crear y mantener una Base de Datos (BD), por lo tanto, el SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular la BD para diversas aplicaciones. Pueden ser de propósito general o específico. (19)

Actualmente existe una amplia gama de SGBD con características propias, no obstante todos deben tener en cuenta los siguientes aspectos de manera general: abstracción de la información, la independencia de los datos, redundancia mínima, consistencia en los datos, seguridad, integridad, respaldo y recuperación, tiempo de respuesta y control de concurrencia. (19)

2.4.1. PostgreSQL 8.4

PostgreSQL es un potente motor de bases de datos, que tiene prestaciones y funcionalidades equivalentes a muchos gestores de bases de datos comerciales. Es un sistema de gestión de base de datos relacional orientada a objetos y libre.

Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, altruista, libre, apoyada por organizaciones comerciales. Dicha comunidad es denominada el PGDG (PostgreSQL Global Development Group). Es más completo que MySQL ya que permite métodos almacenados, restricciones de integridad, vistas, etc. aunque en las últimas versiones de MySQL se han hecho grandes avances en ese sentido. (20)

La utilización de PostgresSQL como SGBD base para almacenar toda la información que se genera en la PTARTV, a pesar que el Generador Dinámico de Reportes estandariza la generación de reportes independientemente del Sistema Gestor de Base de Datos que sea usado, el subsistema de Reportes va a estar haciendo uso de la información que se encuentra almacenada en la base de datos de la PTARTV.

De forma general PostgresSQL proporciona la elaboración de un sistema con gran robustez y alto nivel de escalabilidad.

2.5. Servidor web

El servidor web es un programa que corre sobre el servidor que escucha las peticiones HTTP que le llegan y las satisface. Dependiendo del tipo de la petición, el servidor web buscará una página web o bien ejecutará un programa en el servidor. De cualquier modo, siempre devolverá algún tipo de resultado HTML al cliente o navegador que realizó la petición.

A la hora de afrontar un proyecto web, uno de los puntos más importantes a tener en cuenta es el servidor donde se va a alojar el trabajo. Pues es un ordenador conectado a la red 24 horas al día, 7 días a la semana y accesible por cualquier persona que conozca de su existencia, bien sea a través de una dirección ip o un dominio de internet.

2.5.1. Apache

Es un servidor web HTTP de código abierto y multiplataforma, presenta una elevada robustez y estabilidad por lo que hacen que cada vez millones de servidores reiteren su confianza en este programa. (21)

La historia de Apache se remonta a 1995, donde empieza el proyecto del grupo Apache, el cual está basado en el servidor Apache httpd de la aplicación original de NCSA (National Center for Supercomputing Applications). El desarrollo de esta aplicación original se estancó por algún tiempo tras la marcha de Rob McCool por lo que varios webmaster siguieron creando sus parches para sus servidores web, fue ahí cuando formaron el grupo Apache. (21)

La licencia Apache es descendiente de la licencias BSD (Berkeley Software Distribution), no es GPL. Esta licencia te permite hacer lo que quieras con el código fuente siempre que reconozcas el trabajo. (21)

Ventajas:

- Corre en una multitud de Sistemas Operativos, lo que lo hace prácticamente universal.
- Es una tecnología gratuita y de código fuente abierto. El hecho de ser gratuita es importante pero no tanto como que se trate de código fuente abierto. Esto le da una transparencia a este software

de manera que si se quiere ver que es lo que se está instalando como servidor, se puede saber, sin ningún secreto, sin ninguna puerta trasera.

- Apache es un servidor altamente configurable de diseño modular. Es muy sencillo ampliar las capacidades del servidor web Apache.
- Apache permite personalizar la respuesta ante los posibles errores que se puedan dar en el servidor. Es posible configurar Apache para que ejecute un determinado script cuando ocurra un error en concreto. (21)

El servidor web Apache entre los puntos más significativos por los cuales fue seleccionado se encuentran su gran uso a nivel internacional, su integridad y robustez además de las enormes posibilidades que brinda al estar implementado sobre software libre; por lo antes planteado resulta la opción más factible para el Generador Dinámico de Reportes y señalar además que la PTARTV alberga sus subsistemas web también en este servidor.

2.6. Patrón Arquitectónico

Cada patrón describe un problema que ocurre una y otra vez en un mismo entorno, para describir después el núcleo de la solución a ese problema, de tal manera que esa solución pueda ser usada más de un millón de veces sin hacerlo siquiera en dos ocasiones de la misma forma.

2.6.1. Modelo en Capas

El patrón de arquitectura en capas genera una organización con cierta jerarquía permitiendo que cada capa provea a la superior los servicios que requiere. Aunque pueden implementarse sistemas con dos, tres y n capas el ejemplo más representativo lo constituye el sistema con arquitectura de tres capas.

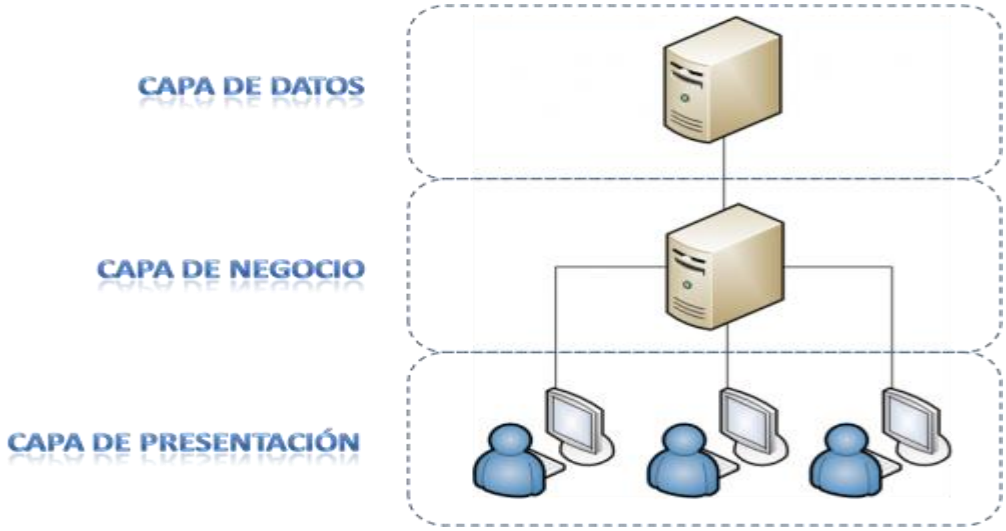


Figura 1: Representación de un sistema con arquitectura de tres capas. (22)

Las tres capas planteadas son:

- Capa de Presentación.

Está habilitada para el manejo de las interfaces e interacción con los usuarios. Es la cara del sistema y debe ser funcional para lograr una buena comunicación entre el sistema y los agentes externos.

- Capa de Negocio (Capa Lógica).

La forman los elementos que dentro del software se encargan de la automatización de los procesos de negocio realizado por los usuarios. Se comunica con la capa de presentación, para recibir las solicitudes y presentar los resultados, y con la capa de datos, para solicitar información al gestor de base de datos, almacenarla o recuperar datos de él.

- La capa de Datos.

Reúne las capacidades que se ocupan del manejo de los datos. Puede estar formada por uno o más gestores de bases de datos. Recibe las órdenes de almacenamiento o recuperación de la información desde la capa de negocio.

2.6.2. Modelo Vista Controlador

El patrón de diseño Modelo Vista Controlador o MVC describe una forma, muy utilizada en el web, da organización al código de una aplicación separando los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Se trata de realizar un diseño que desacople la vista del modelo, con la finalidad de mejorar la reusabilidad. De esta forma las modificaciones en las vistas impactan en menor medida en la lógica de negocio o de datos.

- **Modelo:** Componente encargado del acceso a datos
- **Vista:** Define la interfaz de usuario, HTML+CSS, enviados en el navegador
- **Controlador:** Responde a eventos y modifica la vista y el modelo

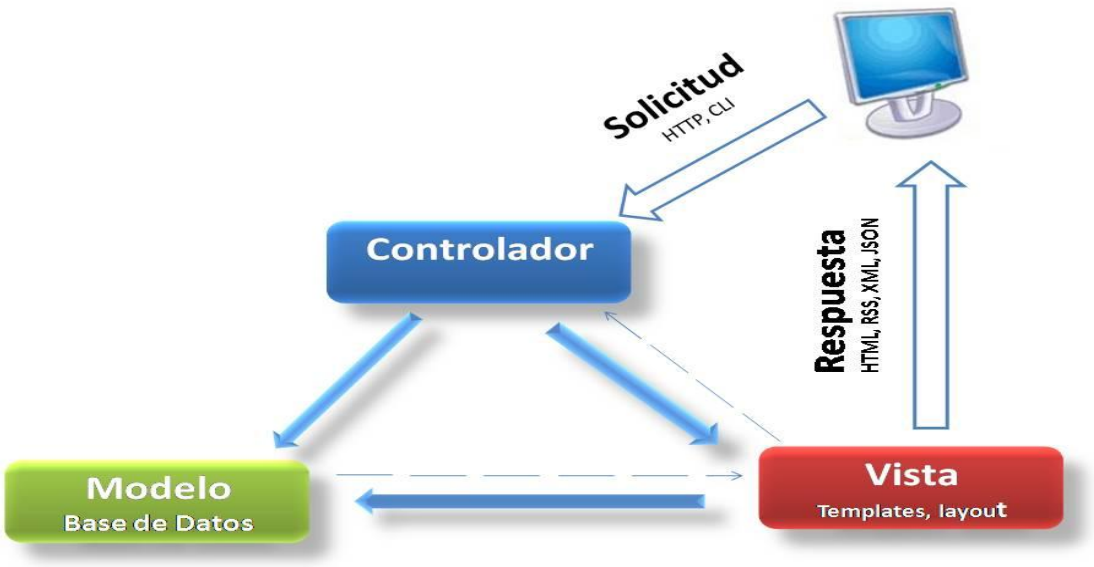


Figura 2: Representación del patrón arquitectónico Modelo Vista Controlador.

2.6.3. Fundamentación del patrón arquitectónico seleccionado

En los diseños de sistemas actuales a nivel mundial se suele usar las arquitecturas multinivel o programación por capas. Con el uso del modelo de tres capas se logra en cierta medida obtener una mejor organización durante la realización del software.

En dichas arquitecturas a cada nivel se le confía una misión concreta, lo que permite el diseño de arquitecturas escalables (que pueden ser ampliadas con facilidad en caso de ser necesario).

2.7. Metodología de Desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos y técnicas que ayudan a la documentación para el desarrollo de productos de software. Es como un libro de recetas de cocina, en el que se van indicando paso a paso todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben de tener. (24)

2.7.1. Proceso Unificado de Desarrollo (Rational Unified Process RUP)

El Proceso Unificado de Desarrollo tiene como fundamento la experiencia adquirida en el uso de la tecnología orientada a objetos y el desarrollo de software en una variedad de industrias encabezada por la compañía Rational, donde resaltan figuras como Grady Booch, James Rumbaugh e Ivar Jacobson. Es un proceso de desarrollo de software configurable que se adapta a través de los proyectos variados en tamaños y complejidad. Ha demostrado a través de los años su gran efectividad por lo que las grandes consultoras suelen recomendarlo incluso para proyectos delicados de misión crítica. (23)

Lo que sustenta el proceso de desarrollo de software son: el proyecto, las personas, el producto y el proceso, existe una estrecha relación entre ellas. Es conocido como las cuatro P en el desarrollo del software. Los aspectos más importantes que se definen en este Proceso Unificado son tres: es iterativo e incremental, dirigido por casos de uso y centrado en la arquitectura. (23)

RUP está enfocado a cualquier tipo de proyecto, es válido decir que RUP posee cuatro fases que representan el ciclo de desarrollo del software: inicio, elaboración, construcción y transición, con hitos

definidos en cada una de ellas; estas deben satisfacer la construcción de un grupo de artefactos (productos tangibles) para cumplir sus objetivos o hitos. A continuación se muestra la imagen de los flujos de trabajo que define RUP para cada unas de las fases.

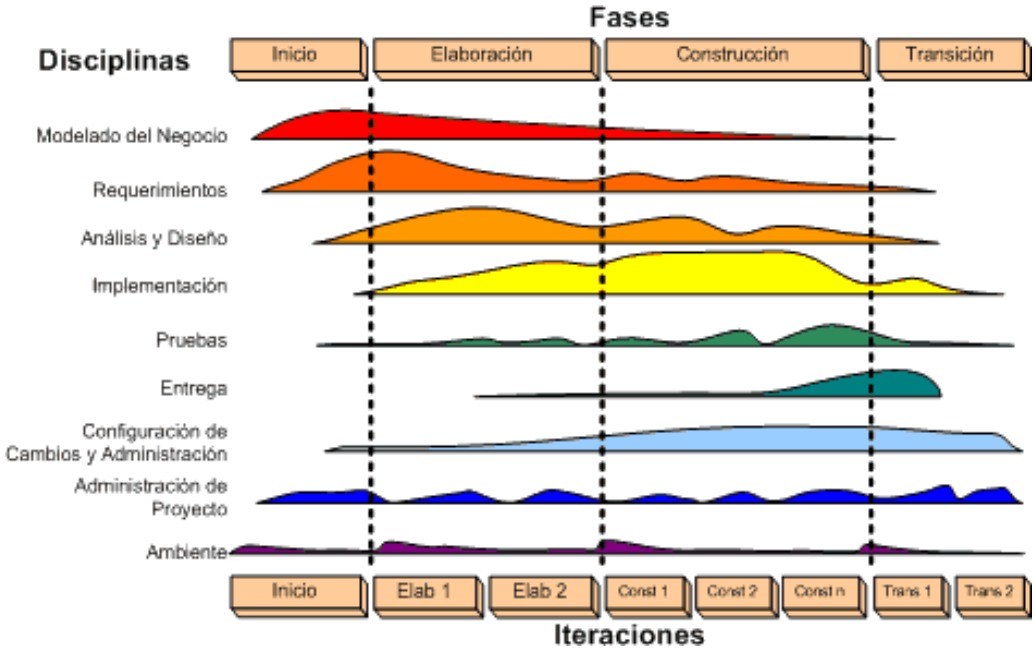


Figura 3: Diagrama de flujos de trabajos de RUP. (24)

Se hará énfasis en las actividades que son responsabilidad del rol de implementador en flujo trabajo implementación, ellas son:

- Implementar elementos de diseño
- Implementar los subsistemas de implementación
- Ejecutar pruebas de desarrollador
- Implementar las prueba de desarrollador
- Corregir la Implementación (25)

RUP utiliza Unified Model Language (UML) como el lenguaje de modelado para la representación de los diagramas. A continuación se describen las características de dicho lenguaje.

2.8. Lenguaje Unificado de Modelado (Unified Model Language UML)

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software que ayuda a capturar la idea de un sistema para comunicarla posteriormente a quien está involucrado en su proceso de desarrollo; esto se lleva a cabo mediante un conjunto de símbolos y diagramas. Cada diagrama tiene fines distintos dentro del proceso de desarrollo. (26)

UML es el lenguaje de modelado de sistemas de software más popular en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Además de lo anterior, ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. Es sin lugar a dudas una herramienta muy poderosa cuando se intenta crear un sistema complejo y se hace necesaria la comprensión general del mismo por parte de todas las personas involucradas.

2.9. Herramientas CASE para la modelación UML

Las herramientas CASE[8] tienen como fin automatizar los aspectos claves de todo el proceso de desarrollo de un sistema. De acuerdo con Kendall la ingeniería de sistemas asistida por ordenador es la aplicación de tecnología informática a las actividades, las técnicas y las metodologías propias de desarrollo, su objetivo es acelerar el proceso para el que han sido diseñadas, en el caso de CASE para automatizar o apoyar una o más fases del ciclo de vida del desarrollo de sistemas. El uso de Herramientas CASE para la modelación UML proporciona mayor rapidez y entendimiento en el desarrollo de software. Este tipo de aplicaciones se han convertido en ayuda y apoyo imprescindible para los desarrolladores.

2.9.1. Visual Paradigm

Visual Paradigm para UML es una herramienta UML profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El software de modelado UML ayuda a una más rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código

[8] Computer Aided Software Engineering, del español Ingeniería de Software Asistida por Ordenador

desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML. (27)

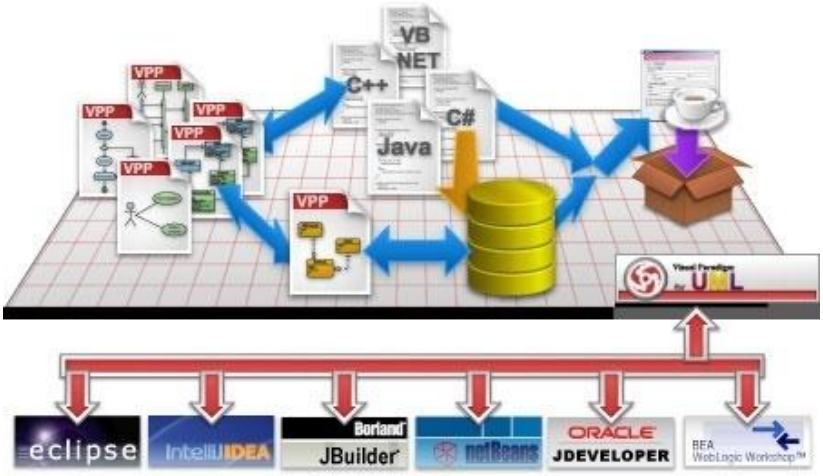


Figura 4: Descripción de Visual Paradigm para UML (28)

Entre sus características figuran:

- El uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Entorno de creación de diagramas para UML 2.0.
- Diseño centrado en casos de uso y enfocado al negocio que generan un software de mayor calidad.
- Modelo y código que permanece sincronizado en todo el ciclo de desarrollo.
- Disponibilidad de integrarse en los principales IDEs.

2.9.2. Rational Rose Enterprise

Rational Rose Enterprise constituye una herramienta poderosa para el modelado visual durante el ciclo de vida de un software. Rose propone cuatro tipos de modelo para la realización del diseño de un sistema, utilizando para ello una vista estática y otra dinámica de los modelos del sistema, uno lógico y otro físico. Permite crear y refinar estas vistas creando de esta forma un modelo completo que representa el dominio del problema y el sistema de software.

Rational Rose Enterprise utiliza un proceso de desarrollo iterativo controlado, donde el desarrollo se lleva a cabo en una secuencia de iteraciones. Ofrece la posibilidad de generar código a partir de modelos en Ada, ANSI C++, C++, CORBA, Java/J2EE, Visual C++ y Visual Basic. Proporciona un lenguaje común de modelado para el equipo que facilita la creación de software de calidad más rápidamente. (29)

Entre las características fundamentales del Rational Rose Enterprise se pueden citar:

- Cubre todo el ciclo de vida de un proyecto.
- Concepción y formalización del modelo.
- Construcción de los componentes.
- Transición a los usuarios.
- Certificación de las distintas fases

2.9.3. Fundamentación de la Herramienta CASE seleccionada.

La utilización de la herramienta CASE Rational Rose Enterprise seria una solución viable por su gran cantidad de funcionalidades y su fácil uso para cualquier usuario pero una de las políticas de la PTARTV es su desarrollo basado en software libre, por lo que se impone el uso de la herramienta CASE Visual Paradigm la cual cumple con este término a diferencia de Rational Rose Enterprise. Apoyándose también en el alto número de ventajas proporcionadas por Visual Paradigm entre las cuales se encuentran su fácil uso, es amigable, genera código y posee editor de figuras.

2.10. Sistema Operativo

Un Sistema Operativo es el software encargado de ejercer el control y coordinar el uso del hardware entre diferentes programas de aplicación y los diferentes usuarios. Es un administrador de los recursos de hardware del sistema. En una definición informal es un sistema que consiste en ofrecer una distribución ordenada y controlada de los procesadores, memorias y dispositivos de E/S entre los diversos programas que compiten por ellos. (30)

El sistema operativo es el programa (o software) más importante de un ordenador. Para que funcionen los otros programas, cada ordenador para su uso debe tener un sistema operativo. (31)

Figura 5: Sistemas Operativos más utilizados a nivel mundial. (32)

2.10.1. Ubuntu Linux

Ubuntu es una distribución GNU/Linux que ofrece un Sistema Operativo predominantemente enfocado a ordenadores de escritorio aunque también proporciona soporte para servidores. Basada en Debian GNU/Linux, Ubuntu concentra su objetivo en la facilidad de uso, la libertad de uso, los lanzamientos regulares (cada 6 meses) y la facilidad en la instalación. (33)

El proyecto Ubuntu está totalmente basado en los principios del desarrollo de software de código abierto; se anima a que la gente use, mejore y distribuya software de código abierto. (33)

Desde sus inicios la PTARTV se propuso la meta de desarrollarse completamente en software libre haciendo uso del sistema operativo Ubuntu Linux**.** Ubuntu se convierte según las necesidades de este trabajo en una solución robusta. Actualmente el sistema operativo se encuentra en su versión 11.4.

2.11. Conclusiones parciales

En este capítulo se ha realizado una fundamentación de las tendencias tecnológicas, técnicas y lenguajes de programación, metodologías de desarrollo y patrones arquitectónicos. Una selección racional de los elementos descritos anteriormente permitirá obtener un sistema de mayor calidad. Se propone utilizar para desarrollar la solución la utilización del lenguaje de programación C++ y el framework de desarrollo Qt. También se expuso una evaluación de las herramientas y tecnologías empleadas en la construcción del

Generador Dinámico de Reporte del departamento DATEC. Además se realizó una evaluación de las metodologías de desarrollo arrojando como conclusión que la mejor opción era la utilización de RUP.

Capítulo3. Descripción de la solución propuesta

Introducción

En este capítulo se desarrollará una breve descripción de la solución propuesta para la realización del subsistema de Reportes de la PTARTV, además de una representación de los artefactos y documentación a realizarse según la metodología de desarrollo seleccionada entre ellos los diagramas de componente y despliegue así como el modelo de implementación. También se analizaran los estándares de codificación utilizados en la implementación.

3.1. Descripción de las aplicaciones

3.1.1. Generador Dinámico de Reportes (GDR)

El Generador Dinámico de Reportes es una aplicación web desarrollada con el framework de desarrollo Symfony, el cual estructura la aplicación por módulos. Quedando el GDR conformados por 5 módulos principales visor de reportes, diseñador de modelos, diseñador de reportes, diseñador de consulta y administrador de reportes los cueles serán analizados a continuación; haciendo énfasis en sus principales características y funcionalidades.

El GDR se utilizará para la confección de los reportes en cada uno los subsistemas dentro de la plataforma PTARTV, durante la elaboración de los mismos es necesario tener presente que estos serán registrados en el formato que se explica a continuación: nombre del reporte guion bajo y nombre del subsistema al cual pertenece.

Ejemplos validos: Reporte de las interrupciones televisivas_transmisión, Reporte de las medias eliminadas del servidor_medias, Reporte de las interrupciones_radial. Ejemplos incorrectos: reporte de las interrupciones radiales, Reporte de las medias eliminadas del servidor.

- **Módulo diseñador de modelos**

La principal función de este módulo es adicionar y configurar la base de datos a utilizarse para la realización de los reportes.

Está compuesto por dos componentes fundamentales:

- ✓ **Seleccionar un origen de datos** donde se adicionan, modifican o eliminan los orígenes de datos.
- ✓ **Modelos Existentes** donde se muestran los modelos anteriormente creados así como los elementos que los componen (Consultas, Rutinas, Tablas, Vistas).

- **Módulo diseñador de consultas**

Con la utilización de este módulo se podrán generar las consultas a la base de datos a utilizarse posteriormente en la confección de los reportes. También proporciona la posibilidad de visualizar los resultados de la consulta.

Está estructurado por dos componentes fundamentales:

- ✓ **Área de desarrollo** es el área sobre la cual se diseñará la consulta, la cual consta de dos pestañas: **Diseño** donde se establecen las consultas visualmente y **Editor de Consultas** en la cual se verifica la sintaxis de la consulta realizada.
- ✓ **Explorador de consultas** muestra los modelos existentes y el árbol de construcción de la consulta que se diseñe.

- **Módulo diseñador de reportes**

Haciendo uso de este módulo se pueden diseñar los reportes. El área de trabajo está estructurada por tres componentes fundamentales:

- ✓ **Paleta de componentes** muestra todos los componentes y las propiedades del componente que esté utilizando.
- ✓ **Área de diseño** es donde se encuentra cada una de los espacios que tiene la estructura de un documento y soporta la utilización de los componentes en cada uno de los espacios.

- ✓ **Inspector** presenta en forma de árbol la estructura del reporte o documento a realizar; en este componente se selecciona además la fuente de datos con que se va a trabajar en el diseño del reporte.

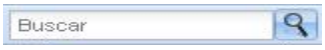
- ➢ **Módulo visor de reportes**

Este es el módulo encargado de presentarle al usuario todos los reportes existentes, los cuales han sido desarrollados y guardados con anterioridad en el modulo diseñador de reportes.

Se encuentra dividido en dos paneles:

- ✓ **Reportes existentes** donde se muestran los reportes generados con anterioridad, el mismo se encuentra en la parte izquierda.
- ✓ **Visor** donde se visualiza el reporte seleccionado.

Entre sus principales funcionalidades resaltan:

- ✓ **Búsqueda de un reporte** en el cual se escribe el nombre o las iníciales para que se vayan filtrando aquellos reportes que cumplan con el criterio de búsqueda. 

- ✓ **Filtrado de un reporte** se puede filtrar los reportes por distintos campos ya sean parámetros o campos, también se pueden adicionar nuevos filtros.
- ✓ **Exportar un reporte** permite exportar los reportes en diferentes formatos (PDF, EXCEL, HTML, CSV).
- ✓ **Imprimir un reporte** brinda la posibilidad de imprimir los reportes, especificando la cantidad de copias.

- ➢ **Módulo administrador de reportes**

La responsabilidad de este modulo es administrar las plantillas, los reportes y los modelos anteriormente creados. Entiéndase por administrar las acciones de buscar, eliminar, modificar y renombrar.

3.1.2. Componente para la Gestión de reportes desde aplicaciones de escritorio (PTARTV_report).

Este componente le proporciona a los subsistemas desarrollados en entornos de escritorio en la PTARTV la posibilidad de visualizar los reportes desde sus aplicaciones sin la necesidad de tener que hacer uso del Generador Dinámico de Reportes. Para su implementación se ha dividido en 4 casos de uso fundamentales, gestionar, visualizar, exportar y graficar reportes. A continuación se realizará una descripción de estos casos de uso para un mejor entendimiento de los mismos.

- **Gestionar Reportes**

El principal objetivo de este caso de uso es la gestión de los reportes como su nombre indica, entiéndase por gestión (cargar y eliminar los reportes). Para cargar los reportes es necesario especifica un subsistema y posteriormente se comparan los nombre de los reportes generados a través del GDR y guardados en su base de datos. Con todos los reportes que coincidan con el nombre del subsistema proporcionado se crea una lista, almacenándose todos los datos de los reportes: nombre, id, XML del reporte (proporciona todos los datos del reporte como son la cantidad y los nombres de las columnas del reporte), XML de la consulta (brinda la consulta del reporte) y gráfica (nos dice si el reporte es una gráfica o no). En el caso de eliminar un reporte se pasa un su nombre y es eliminado de la lista.

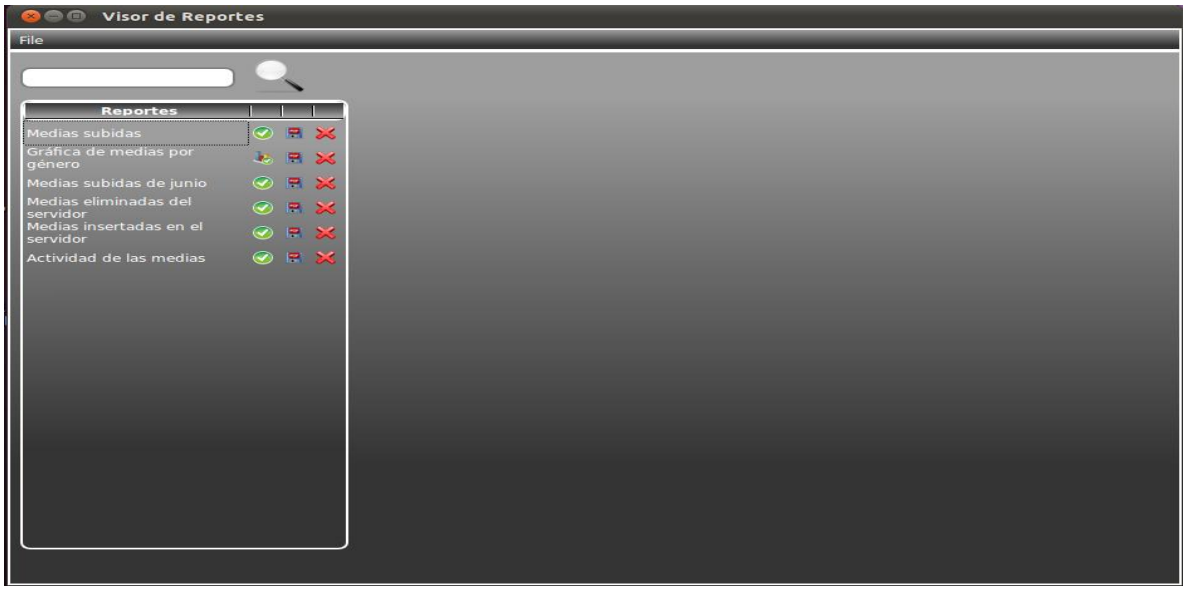


Figura 6: Visualización de la lista de reportes.

- **Visualizar Reportes**

Este caso de uso es el encargado de obtener los datos de los reportes para mostrárselos a los usuarios. Se debe pasar un reporte y posteriormente se lee el XML del mismo y se devuelve una lista con los nombres de las columnas, después se ejecuta el XML de la consulta y se devuelve una query. A continuación con los datos devueltos por estos dos métodos se puede proceder a la visualización del reporte ya sea en una tabla u otro componente.

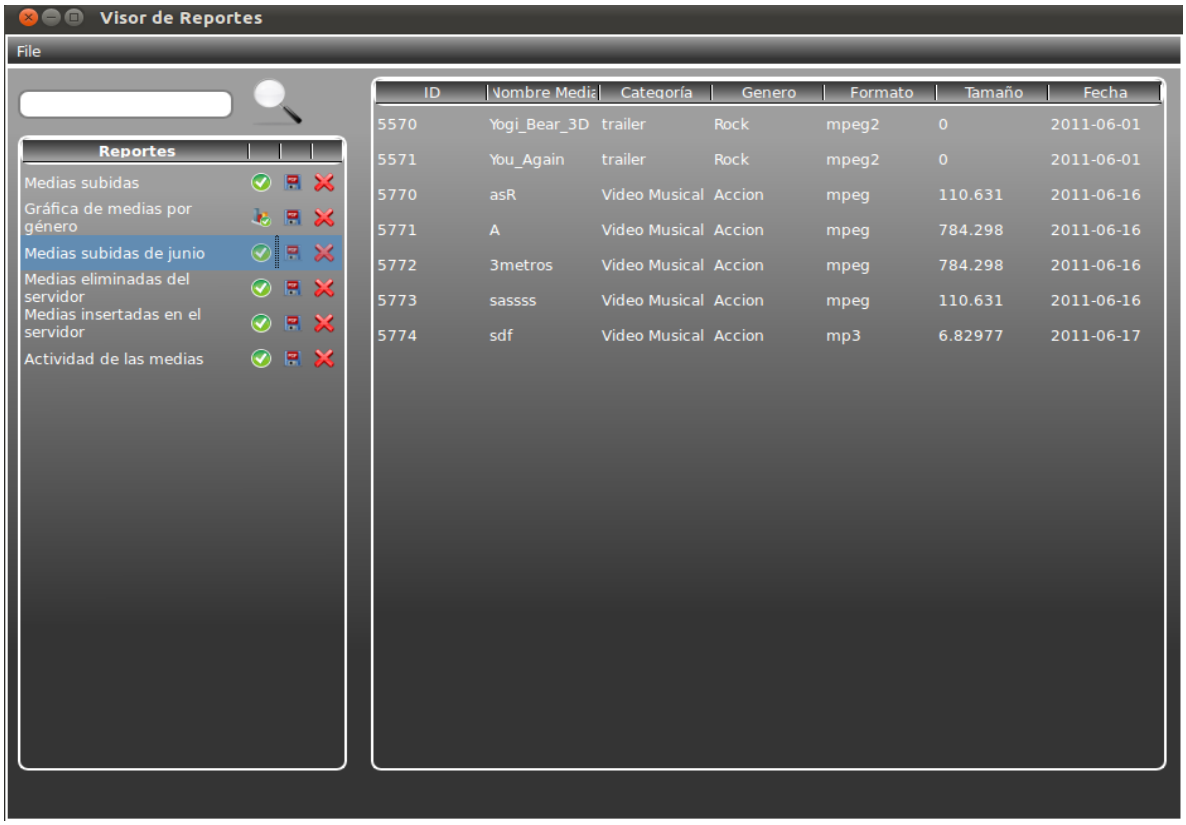


Figura 7: Visualización de un reporte.

- **Exportar Reportes**

Este caso de uso permite exportar los reportes a formato .pdf. Dado el id de un reporte se realiza una petición a través de una API que proporciona el Generador Dinámico de Reportes, la cual devuelve un pdf con los datos del reporte.

- **Graficar Reportes**

En este caso de uso se brinda la posibilidad de poder mostrar los datos de los reportes de forma gráfica. Primero se pregunta si el reporte es una gráfica y posteriormente se leen los datos de los XML del reporte dado de igual forma que para visualizarlo y se crea una gráfica con los datos recuperados. También se brinda la opción de nuevamente realizar una petición al servidor para volver a graficar el reporte de ocurrir cambios en la base de datos.

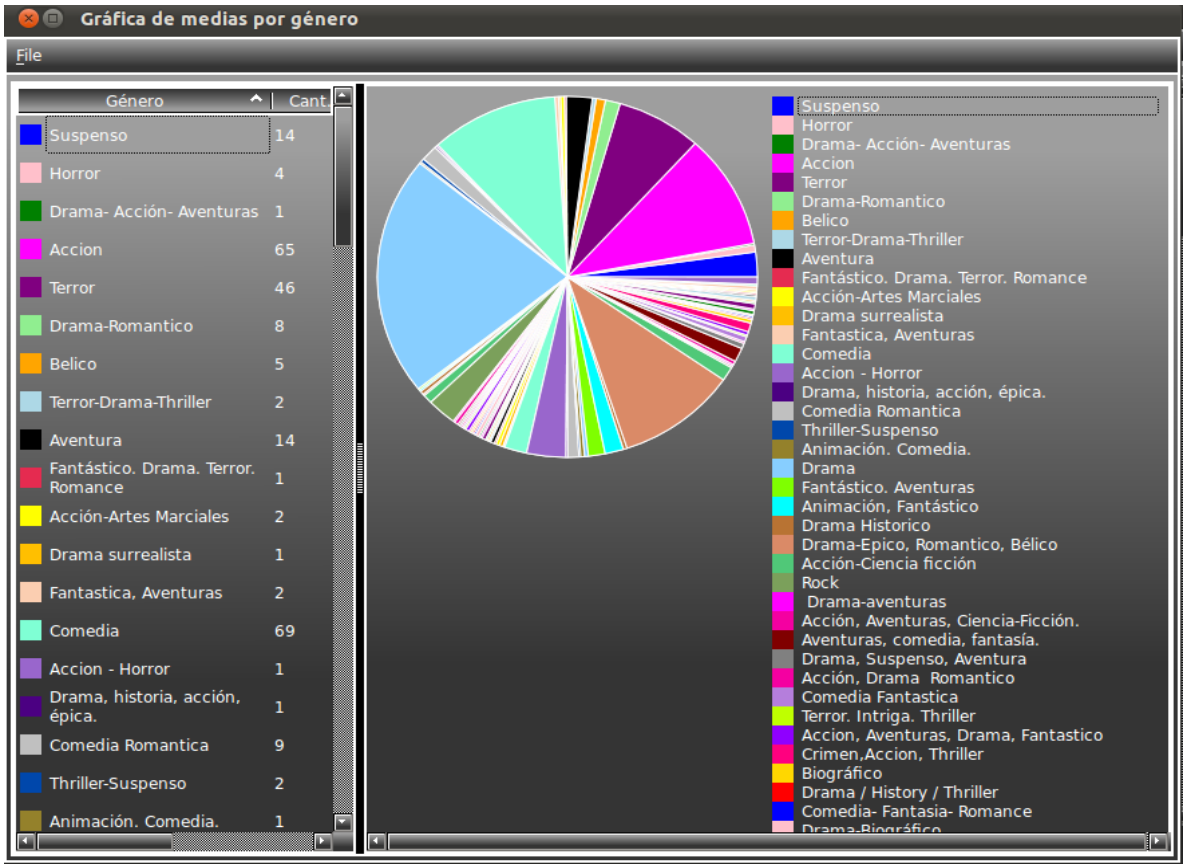


Figura 8: Representación gráfica de un reporte.

3.2. Modelo de Implementación

El modelo de implementación está comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Entre los componentes se pueden

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. Este artefacto describe cómo se implementan los componentes, congregándolos en subsistemas organizados en capas y jerarquías, y señala las dependencias entre éstos. (33)

Durante la selección del patrón arquitectónico se decidió seguir una arquitectura de tres capas. La implementación de la aplicación, se ha ceñido a dicha arquitectura manteniendo la independencia entre los componentes de cada capa y realizando únicamente llamadas descendentes sin saltar niveles.

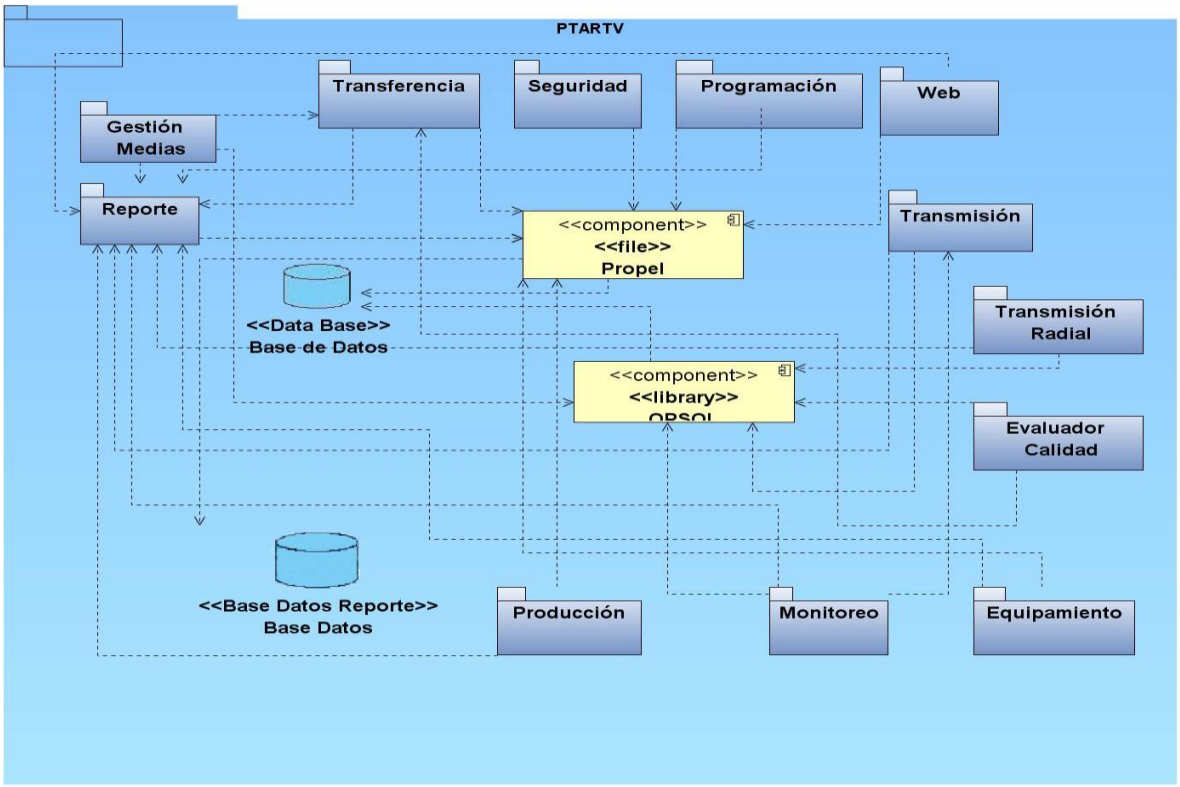


Figura 9: Modelo de implementación

3.3. Diagrama de Componentes

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Estos diagramas 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, bibliotecas cargadas dinámicamente, entre otros. Las relaciones de dependencia se utilizan en los diagramas de componentes para indicar que un componente utiliza los servicios ofrecidos por otro componente.

Los principales estereotipos estándares que presentan los componentes dentro de un modelo son:

- <<executable>> Representa programas que se ejecutan en un nodo
- <<file>> Son ficheros de datos o código fuente
- <<library>> Modelan librerías estáticas o dinámicas
- <<table>> Constituyen tablas de bases de datos
- <<document>> Simbolizan documentos

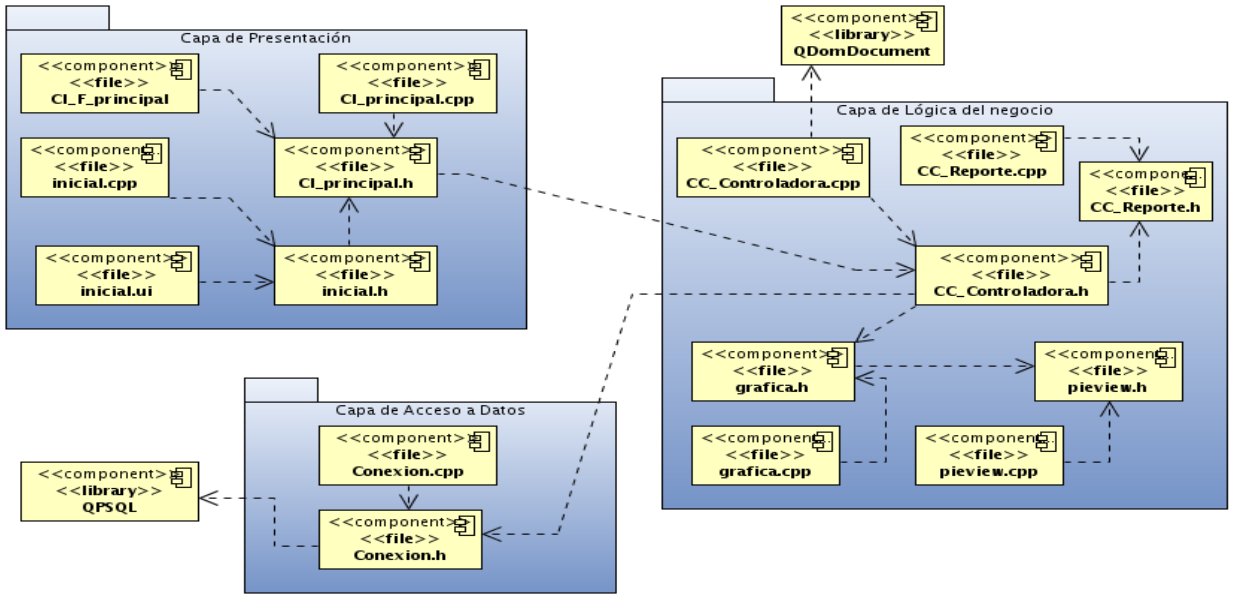


Figura 10: Diagrama de componentes para el subsistema de Reporte

3.4. Diagramas de Despliegue

El Diagrama de Despliegue es un tipo de diagrama del Lenguaje Unificado de Modelado que modela la arquitectura en tiempo de ejecución de un sistema, muestra la configuración de los elementos de hardware (nodos) y cómo los elementos y artefactos del software se relacionan.

Un nodo es un elemento de hardware o software. Se representa en forma de caja en tres dimensiones.

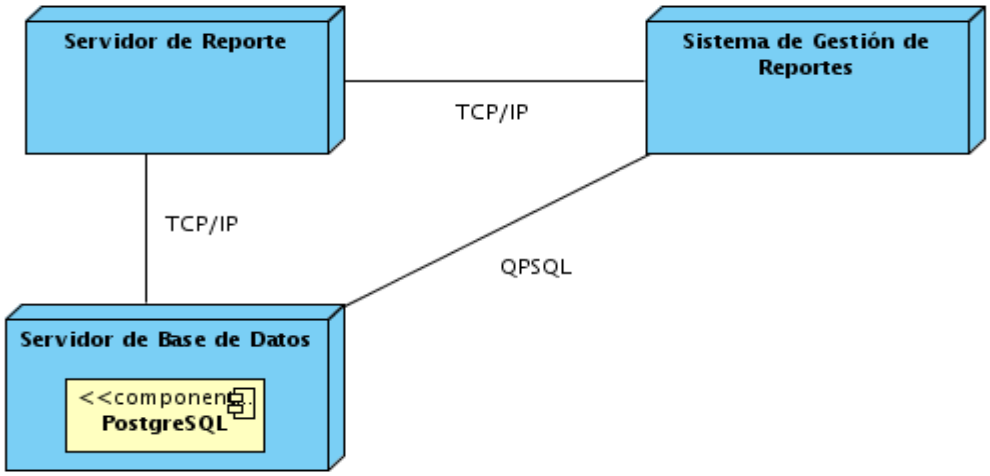


Figura 11: Diagrama de Despliegue para el subsistema de Reporte.

En el nodo Servidor de Base de Datos se van a encontrar el Sistema de Gestión de Base de Datos (SGBD) PostgresSQL donde van a estar tanto la base de datos de la plataforma PTARTV como la del Generador Dinámico de Reportes.

El nodo Servidor de Reportes será donde estará instalado el Generador Dinámico de Reportes el cual a su ves estará conectado con el servidor de base de datos a través de una conexión TCP/IP

En el nodo Sistema de Gestión de Reportes se encuentra el componente desarrollado en Qt para la visualización de reportes desde las aplicaciones programadas en entorno de escritorio dentro de la plataforma PTARTV, la cual obtiene los datos del servidor haciendo uso del driver QPSQL y se conecta con el servidor de reporte a través de TCP/IP para la generación de cada uno de los reportes.

3.5. Estándares de Codificación

Las convenciones o estándares de codificación son pautas de programación que no están enfocadas a la lógica del programa, sino a su estructura y apariencia física para facilitar la lectura, comprensión y mantenimiento del código. A continuación se presentan algunas de estas convenciones para la programación en lenguaje C++.

3.5.1 Comentarios

Cada programa deberá comenzar con un comentario que incluya:

- Autor
- Fecha
- Objetivo, o problema que resuelve el programa
- Fecha de creación y bitácora de versiones con las dos últimas fechas de modificación
- Algoritmo

Cada función debe tener un encabezado que contenga:

- Objetivo de la función y no descripción del procedimiento.
- Comentarios de apoyo a variables, llamadas a función o inclusión de archivos que no sean obvios al proceso.
- Explicación de uso de argumentos (parámetros) no obvios.
- Explicación de uso de valores devueltos (de retorno).

3.5.2 Nombres de identificadores

Se considera como identificador a los nombres de variables (arreglos, matrices, apuntadores), funciones, así como cualquier tipo de dato definido por el usuario (estructura, clase). Dichos identificadores deberán seguir las siguientes normas, además de las definidas por el propio lenguaje.

- Deberán tener un nombre significativo para que por su simple lectura, pueda conocerse su función, sin tener que consultar manuales o hacer demasiados comentarios.
- Para nombres que se usen con frecuencia o para términos largos, se recomienda usar abreviaturas estándar para que éstos tengan una longitud razonable. Si usa abreviaturas deben manejar la misma lógica en todo el programa.
- Evitar identificadores que comiencen con uno o dos caracteres de subrayado para evitar que se confundan con los que el compilador selecciona.
- Cada identificador de función, variable o procedimiento deberá ser precedido por la abreviación del tipo de dato de que es la variable, o si se trata de una función o procedimiento del tipo de dato que regresa.

3.5.2.1 Identificadores de variables

Comenzarán siempre con la primera letra minúscula correspondiente a su tipo de dato.

Para distinguir palabras dentro del nombre deberá emplearse una letra mayúscula o un guión bajo (_), sin mezclar ambas formas en un mismo programa.

Ejemplos:

temperaturaDeVapor

temperatura_de_vapor

3.5.2.2 Identificadores de punteros (apuntadores)

Su nombre deberá comenzar con la letra p.

Ejemplo: pAlumno

Dónde pAlumno es un puntero que podrá tener la dirección del lugar donde se almacena información de un alumno.

Tipo de dato Abreviatura

integer I, float f, double d, Arreglo ar, char c, enumeración e, estructura st

constantes TODAS LAS LETRAS DEL IDENTIFICADOR CON MAYÚSCULA.

punteros p

3.5.2.3 Identificadores de variables dimensionadas (arreglos, matrices)

Su nombre deberá comenzar con las letras ar.

Ejemplo: arAlumnos

Donde *arAlumnos* es un arreglo de datos para guardar información de alumnos.

3.5.2.4 Identificadores de datos constantes

Serán declaradas en letras mayúsculas.

Ejemplo: const IVA = 0.15;

3.5.2.5 Identificadores de funciones

La primera letra deberá ser mayúscula.

Ejemplo: void vFuncion ( );

3.5.2.6 Identificadores de tipos definidos por el usuario

La primera letra será mayúscula.

Ejemplo: class Clase o struct Estructura

3.5.3 Organización Visual del Programa

3.5.3.1 Generales

- No manejar en los programas más de una instrucción por línea.
- Declarar las variables en líneas separadas
- Añadir comentarios descriptivos junto a cada declaración de variables, si es necesario.

3.5.3.2 Sangrías

- Las sangrías tendrán una longitud de tres espacios.
- Para las llaves que definen el cuerpo de una función, sangre un nivel.

```
Ejemplo: void Funcion ( )

{

//Instrucciones de la función

}
```

- Sangre las instrucciones del cuerpo de cada estructura de control.

```
Ejemplo: for (int x = 0; x < 5; x++)

{

//Instrucciones a ejecutar

}
```

- Trate de evitar codificar más de tres niveles de sangrado.

3.5.3.3 Líneas y espacios en blanco

- Insertar una línea en blanco antes y después de una declaración de datos que aparezca entre instrucciones ejecutables.

```
a = b + c;

// línea en blanco
```

```
int f; //declaración entre instrucciones

// línea en blanco

f = a;
```

- Las declaraciones de datos dentro de una función, deberán ir al inicio y separadas de las instrucciones ejecutables de la función por medio de una línea en blanco.
- Deben incluirse espacios en ambos lados de los operadores binarios.

  Ejemplo:

```
y = 50 + 15 - x;
```

- Es posible distribuir una instrucción grande sobre varias líneas. Si lo hace, seleccione puntos de ruptura que tengan sentido, como después de una coma en el caso de una lista, o después de un operador en el caso de una expresión larga.
- Sangre todas las líneas subsecuentes.

  Ejemplo:

```
cout << “Ejemplo de ruptura de una instrucción en varias”

<<” línea de comandos”;
```

- Los operadores unarios (++, --, etc.) deben ponerse junto a su operando, sin espacios intermedios.
- Antes y después de cada estructura de control deberá poner una línea en blanco.

3.5.3.4 Paréntesis

Para hacer más clara una expresión, es aceptable agregarle paréntesis innecesarios. Dichos paréntesis se llaman paréntesis redundantes.

3.6 Validación de la solución propuesta

Las pruebas son de vital importancia, ya que aseguran la calidad del software, para que el mismo llegue a manos de los clientes cumpliendo todas sus funcionalidades y quedando lo más libre posible de fallas. Cada prueba tiene su estrategia y propósito. La ausencia de defectos no puede ser asegurada a través de las pruebas, sino que solo se puede demostrar que existen errores en el software. El proceso de pruebas cuenta con actividades bien definidas, comienza con la planificación de las pruebas, luego la ejecución, el control y por último la evaluación de las mismas. La creación de extensos y monótonos casos de pruebas ha sido remplazada al transcurrir el tiempo por procesos automatizados.

Tipos de pruebas:

- ✓ **Unitarias:** consisten en comprobaciones (manuales o automatizadas) realizadas para verificar que el código correspondiente funcione de acuerdo con los requisitos del sistema.
- ✓ **Integración:** se realizan una vez que se han aprobado las pruebas unitarias. Consisten en realizar pruebas para verificar que un gran conjunto de partes de software funcionan en juntos.
- ✓ **Funcionales:** A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra, ya que no enfocan su atención a como se generan las respuestas del sistema, básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas.
- ✓ **Rendimiento:** son realizadas para demostrar que el sistema cumple con los criterios de rendimiento, permiten medir que partes del sistema tiene un mal funcionamiento. Se dividen en pruebas de carga, estrés y estabilidad.

- Carga: se realiza generalmente para observar el comportamiento de una aplicación bajo una cantidad de peticiones esperada. Esta carga puede ser el número esperado de usuarios concurrentes utilizando la aplicación y que realizan un número específico de transacciones durante el tiempo que dura la carga. Esta prueba puede mostrar los tiempos de respuesta de todas las transacciones importantes de la aplicación, también se monitorizan, entonces esta prueba puede mostrar el cuello de botella en la aplicación.

- Estrés: se utiliza normalmente para romper la aplicación. Se va doblando el número de usuarios que se agregan a la aplicación y se ejecuta una prueba de carga hasta que se rompe. Este tipo de prueba se realiza para determinar la solidez de la aplicación en los momentos de carga extrema y ayuda a los administradores para determinar si la aplicación rendirá lo suficiente en caso de que la carga real supere a la carga esperada.
- Estabilidad: se hace para determinar si la aplicación puede aguantar una carga esperada continuada. Generalmente esta prueba se realiza para determinar si hay alguna fuga de memoria en la aplicación.

✓ **Aceptación:** el objetivo de la prueba es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento.

A continuación se aborda todo lo referente a las herramientas y elementos empleados en la validación del subsistema de Reporte de la PTARTV con el objetivo de garantizar el correcto funcionamiento del mismo, esta investigación se apoyará en las pruebas unitarias y funcionales.

3.6.1 Pruebas unitarias y funcionales

Los sistemas que han pasado por pruebas unitarias tienen un menor tiempo de pruebas funcionales, este comportamiento es obvio, ya que las pruebas unitarias permiten encontrar los errores más evidentes y fáciles de corregir, en la etapa de pruebas funcionales el sistema debería estar bastante estable y con muy pocos errores críticos. Si un sistema llega a la etapa de pruebas funcionales con demasiados errores críticos y/o bloqueantes, se debería devolver el sistema a la etapa de pruebas unitarias ya que resulta muy poco productivo realizar pruebas funcionales con sistemas inestables, el avance es demasiado lento y no se podrá apoyar mucho en la resolución de los errores ya que en esta etapa solo se centra la atención en las entradas y salidas, y no en la lógica intermedia. (34)

Se denominan pruebas funcionales o Functional Testing a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido

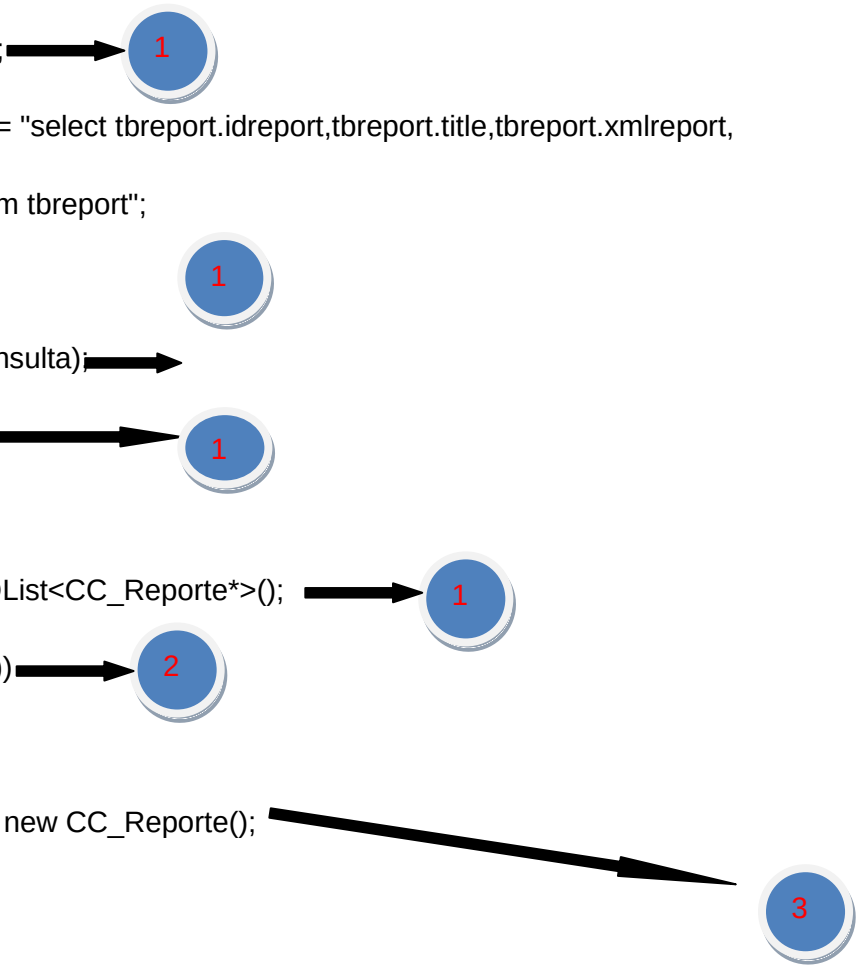
creados, es común que este tipo de pruebas sean desarrolladas con apoyo de algunos usuarios finales, esta etapa suele ser la última etapa de pruebas. (34)

Las pruebas funcionales en la mayoría de los casos son realizadas manualmente, también es posible automatizar este tipo de pruebas utilizando. La automatización de pruebas puede resultar compleja y solo se recomiendan en algunas funcionalidades específicas, por ejemplo en las pantallas que tendrán mayor uso, generalmente pantallas de ingreso de datos.

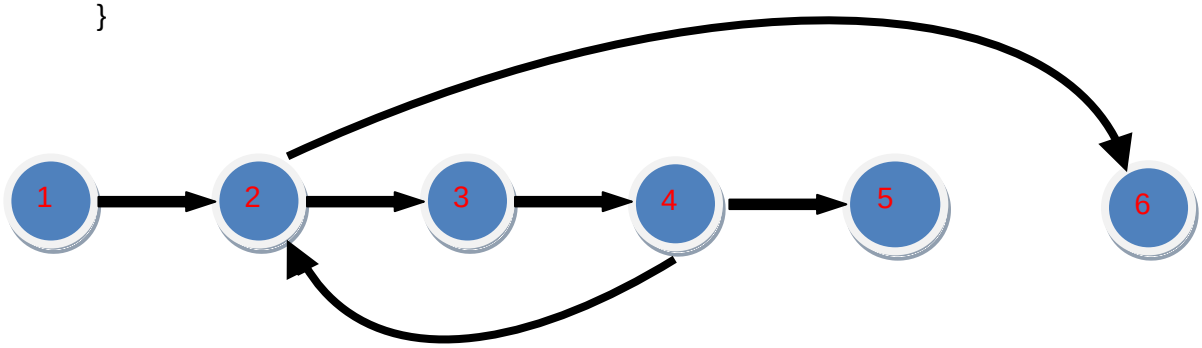
3.6.1.1 Pruebas unitarias

Caso de uso Gestionar Reportes

```
void CC_Controladora::cargar_Reportes(QString subsistema)

{

    QSqlQuery query;

    QString consulta = "select tbreport.idreport,tbreport.title,tbreport.xmlreport,

    tbreport.query from tbreport";

    query.prepare(consulta);

    query.exec();

    lista_reportes = QList<CC_Reporte*>();

    while(query.next())

    {

    reporte_prueba = new CC_Reporte();
```



```
reporte_prueba->set_Id(query.record().value(0).toString());
reporte_prueba->set_Nombre(query.record().value(1).toString());
reporte_prueba->set_Xml(query.record().value(2).toString());
reporte_prueba->set_Query(query.record().value(3).toString());
if(reporte_prueba->get_Nombre().contains(subsistema,Qt::CaseInsensitive))
    {
        lista_reportes.append(reporte_prueba);
    }
}
}
```



Complejidad ciclomática

V (G)= Número de Aristas - Número de Nodos + 2

V (G)= 6- 6+2= 2

Caminos básicos

CB1: 1 2 6

CB2: 1 2 3 4 2 6

CB3: 1 2 3 4 5 2 6

Caso de prueba para el camino básico

CB2: 1 2 3 4 2 6

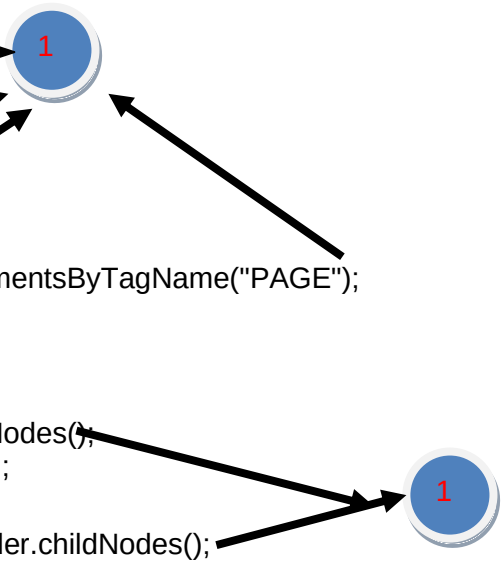
Entrada: Nombre del subsistema.

Resultado Esperado: Se crea una lista con todos los reportes del subsistema.

Condiciones: Se realiza esta tarea antes de realizar las demás.

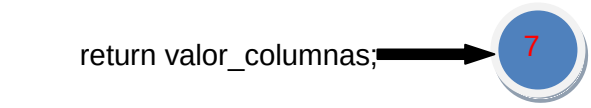
Caso de uso Ver Reportes

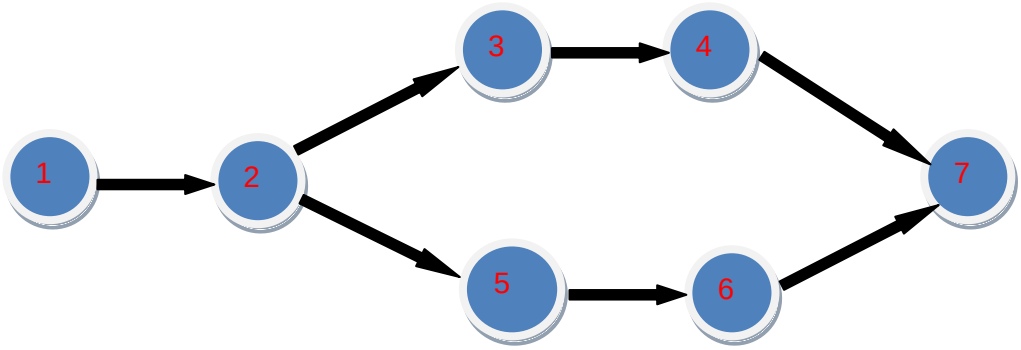
```
QStringList CC_Controladora::xml_read(CC_Reporte *reporte)

{

    QString xml = reporte->get_xml();

    QDomDocument document_xml;

    document_xml.setContent(xml);

    QDomNodeList page = document_xml.elementsByTagName("PAGE");

    QDomNode nodo_page = page.item(0);

   QDomNodeList header = nodo_page.childNodes();
    QDomNode nodo_header = header.item(0);

    QDomNodeList header_band = nodo_header.childNodes();

    QDomNode nodo_header_band = header_band.item(0);
```



```
QDomNodeList header_band_col = nodo_header_band.childNodes();

QDomNodeList img = document_xml.elementsByTagName("IMG");

QStringList valor_columnas;

  if(img.isEmpty())
  {
     for( uint i=0;i<header_band_col.length();i++)
        {
            QDomNode nodo_header_band_col = header_band_col.item(i);
            valor_columnas.append(nodo_header_band_col.toElement().text());
        }
  }
    else
      {
        for(uint i=1;i<header_band_col.length();i++)
            {
               QDomNode nodo_header_band_col = header_band_col.item(i);
                valor_columnas.append(nodo_header_band_col.toElement().text());
            }
```

```
        }
    return valor_columnas;
}
```





Complejidad ciclomática

V (G)= Número de Aristas - Número de Nodos + 2

V(G)= 7- 7+2= 2

Caminos básicos

CB1: 1 2 3 4 7

CB2: 1 2 3 5 6 7

Caso de prueba para el camino básico

CB2: 1 2 3 4 7

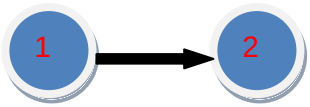
Entrada: Un reporte de la lista cargada.

Resultado Esperado: Una lista con los datos del reporte.

Condiciones: Se realiza después de cargar los reportes.

Caso de uso Exportar Reportes

```
QUrl CC_Controladora::url_exportar(CC_Reporte *reporte)

{

    QString dir_puerto = "localhost:5800";

    QString datos_url = "http://"+dir_puerto+"/report_generator.php/api/exportReport?id="

+  reporte->get_id();

    QUrl url =  QUrl(datos_url);

    return url;

}
```



Complejidad ciclomática

V (G)= Número de Aristas - Número de Nodos + 2

V(G)= 2- 1+2= 3

Caminos básicos

CB1: 1 2

Caso de prueba para el camino básico

CB2: 1 2 3 4 2 6

Entrada: Un reporte de la lista cargada.

Resultado Esperado: Una url con la dirección del reporte.

Condiciones: Se realiza después de cargar y visualizar los reportes.

3.6.1.2 Pruebas funcionales

Descripción de variables para el caso de uso Gestionar reporte				
No	Nombre	Clasificación	Datos Válidos	Datos Inválidos
1	subsistema	Campo de texto (alfanumérico)	Monitoreo, Transmisión, Radial, Medias	Cualquier otra combinación de letras y números hasta 255 caracteres.

Tabla 1: Descripción de las variables utilizadas para las pruebas funcionales

Clases Válidas	Clases Inválidas	Respuesta Esperada	Respuesta de la prueba	Observaciones
Cargar Reportes		Se genera una lista con todos los reportes existentes en la base de datos del subsistema.	Satisfactorio	
Eliminar Reportes		Se eliminan los reportes seleccionados	Satisfactorio	

		de la lista.		

Tabla 2: Resultados de las pruebas funcionales aplicadas al caso de uso Gestionar Reportes.

3.7 Conclusiones parciales

En este capítulo se prestó atención a las características relacionadas con las pruebas para la fase de implementación. Para lograr que el subsistema de Reportes sea lo más robusto posible es vital someterlo a las pruebas pertinentes que confirmen las funcionalidades previstas.

Las pruebas que se le realizaron al software permitieron detectar todas aquellas deficiencias que afectaban el buen desempeño de la aplicación.

Al aplicarle las pruebas unitarias a la aplicación, se demostró que, aunque los procedimientos realizan las tareas de forma satisfactoria, sí presentan cierta demora en la ejecución.

Las pruebas funcionales aplicadas, arrojaron resultados que se tomarán como no conformidades y que serán analizadas en una nueva iteración del proceso de desarrollo de software según la metodología empleada.

Conclusiones generales

Una vez finalizado el proceso de desarrollo se ha podido cumplir a cabalidad con los objetivos propuestos inicialmente de construir una herramienta para la gestión de los reportes dentro de la PTARTV. Permitiendo la generación de reportes de manera sencilla haciendo uso del GDR y su integración con las aplicaciones desarrolladas en entorno de escritorio, dando la posibilidad de exportar esta información a formato pdf.

El estudio efectuado acerca de las principales metodologías de desarrollo de software existentes y las ventajas de cada una permitió concluir que RUP es la metodología que más se ajusta a las necesidades ya que posee un ciclo de desarrollo completo dándole cobertura a todas las etapas.

Visual Paradigm proporcionó un soporte asistido al proceso de ingeniería de software bastante bueno permitiendo la generación de los diferentes diagramas y modelos que brindaron visión adecuada que permitió la posterior implementación del sistema.

El sistema construido cumple con la especificación de requerimientos iníciales y además cumple con los estándares propios de aplicaciones informáticas como son (interfaces amigables e intuitivas, comprensibles, confiable, portable, robusto y escalable).

La elección de C++ como lenguaje y Qt Creator como IDE de desarrollo fue un acierto considerando la estabilidad y confiabilidad del producto final obtenido y cero costo de licenciamiento.

Recomendaciones

La información que maneja cualquier organización debe ser considerada como su principal activo, por lo cual una herramienta de reportes como la que se ha desarrollado en el presente trabajo debe estar funcionando dentro de toda organización poniendo a disposición de los usuarios de manera controlada dicha información.

Se recomienda que cualquier usuario que vaya a hacer uso del GDR se capacite antes de iniciar la manipulación de la información y de los reportes. Con este objetivo se incluyen anexos como el manual de instalación y el manual de usuario que describe con detalles la forma de utilizar la herramienta. Adicionalmente se recomienda asignar la responsabilidad de llevar a cabo el mantenimiento y diseño de los reportes a uno o varios usuarios con el perfil técnico al menos básico que conozcan las funcionalidades brindadas por el GDR.

Se recomienda utilizar la herramienta CASE como Visual Paradigm durante el proceso de diseño de una aplicación y utilizar dicha herramienta en su verdadera dimensión.

Si se requiere un mantenimiento futuro de la aplicación, se recomienda contar con personal con conocimientos en bases de datos PostgresSQL, programación en symfony y C++.

Se recomienda la investigación en otras temáticas como la generación de reportes gráficos en forma de barras como parte de futuras actualizaciones del sistema de generación de reportes para la PTARTV.

Referencias bibliográficas

1. definicionabc. [En línea] [Citado el: 10 de 11 de 2010.] http://www.definicionabc.com/comunicacion/reporte.php..

2. Fundamentos de Investigacion. [En línea] [Citado el: 30 de 11 de 2010.] http://almtzhers81fi.blogspot.es.

3. danysoft. [En línea] [Citado el: 24 de 11 de 2010.] http://www.danysoft.com/bol/crystal10.htm.

4. codigoadicto. [En línea] [Citado el: 15 de 11 de 2010.] http://www.codigoadicto.com/software-crystal-reports-14-beta-1-para-visual-studio-2010.

5. territorioscuola. [En línea] [Citado el: 28 de 11 de 2010.] http://www.territorioscuola.com/software/index_es.php?title=JasperReports.

6. crystalsolutions. [En línea] [Citado el: 24 de 11 de 2010.] http://www.crystalsolutions.com.ar/productos/crystalreports.html.

7. MSDN-microsoft. [En línea] [Citado el: 24 de 11 de 2010.] http://msdn.microsoft.com/es-es/library/ms251771%28VS.80%29.aspx.

8. jasperforge. [En línea] [Citado el: 28 de 11 de 2010.] http://jasperforge.org/projects/jasperreports.

9. javahispano. [En línea] [Citado el: 20 de 11 de 2010.] http://www.javahispano.org/contenidos/es/agrega_dinamismo_a_los_reportes_en_jasperreports_con_dynamicjasper_11/.

10. comunidades. [En línea] [Citado el: 25 de 11 de 2010.] http://comunidades.uci.cu/projects/dynamicreport.

11. **desarrollo, Equipo de.** *PATSI GDR.* Habana, Cuba : s.n., 2010. Manual de usuario.

12. FESI. [En línea] [Citado el: 22 de 01 de 2011.] http://fesi.uci.cu/centros/datec/generador-de-reportes-din%C3%A1micos-patdsi-grd.

13. xpps. [En línea] [Citado el: 25 de 01 de 2011.] http://www.xpps.net/contenido_xppsnet/areatec/HMTL.pdf.

14. librosweb. [En línea] [Citado el: 25 de 01 de 2011.] http://www.librosweb.es/javascript/capitulo1.html.

15. desarrolloweb. [En línea] [Citado el: 20 de 01 de 2011.] http://www.desarrolloweb.com/articulos/436.php.

16. maestros del web. [En línea] [Citado el: 20 de 01 de 2011.] http://www.maestrosdelweb.com/editorial/phpintro.

17. **García de Jalón, Javier, Ignacio Rodríguez, Jose y María Sarriegui, Jose.** mat21. [En línea] [Citado el: 10 de 02 de 2011.] http://mat21.etsii.upm.es/ayudainf/aprendainf/Cpp/manualcpp.pdf.

18. librosweb. [En línea] [Citado el: 20 de 01 de 2011.] http://www.librosweb.es/symfony/capitulo1/symfony_en_pocas_palabras.html.

19. **Gómez Ballester, Eva y Martínez Barco, patricio.** alu. [En línea] [Citado el: 02 de 02 de 2011.] http://www.alu.ua.es/j/jmr36/Conectate/Base%20Datos/Apuntes2006.pdf.

20. guia-ubuntu. [En línea] [Citado el: 20 de 01 de 2011.] http://www.guia-ubuntu.org/index.php?title=PostgreSQL.

21. linux.ciberaula. [En línea] [Citado el: 20 de 01 de 1011.] http://linux.ciberaula.com/articulo/linux_apache_intro.

22. modelo tres capas. [En línea] [Citado el: 20 de 01 de 2011.] http://uhpa.wordpress.com/2009/02/01/pxcapas-2/.

23. mitecnologico. [En línea] [Citado el: 25 de 02 de 2011.] http://www.mitecnologico.com/Main/ElModeloProcesoUnificado.

24. RUP. [En línea] [Citado el: 15 de 02 de 2011.] http://utopicainformatica.blogspot.com/2011/04/bandas-temporales.html.

25. fing. [En línea] [Citado el: 02 de 02 de 2011.] http://www.fing.edu.uy/inco/cursos/ingsoft/pis/proceso/MUM/roles/imp.htm.

26. **Wiley, Jonh.** *sofware Engineering standards.* s.l. : IEEE Press, 2001. 829-1983.

27. free download manager. [En línea] [Citado el: 05 de 02 de 2011.] http://www.freedownloadmanager.org/es/downloads/Paradigma_Visual_para_UML_%28M%C3%8D%29_14720_p/.

28. visual paradigm. [En línea] [Citado el: 26 de 02 de 2011.] http://www.downloadsource.net/2868/Visual-Paradigm-for-UML-Community-Edition/.

29. rational. [En línea] [Citado el: 03 de 02 de 2011.] http://www.rational.com.ar/herramientas/roseenterprise.html.

30. euram. [En línea] [Citado el: 02 de 02 de 2011.] http://www.euram.com.ni/pverdes/verdes_informatica/informatica_al_dia/que_es_un_so_144.htm.

31. mas adelante. [En línea] [Citado el: 02 de 02 de 2011.] http://www.masadelante.com/faqs/sistema-operativo.

32. sistemas operativos. [En línea] [Citado el: 02 de 02 de 2011.] http://ciberprensa.com/fin-de-los-sistemas-operativos/.

33. ubuntu. [En línea] [Citado el: 20 de 02 de 2011.] http://doc.ubuntu-es.org/Sobre_Ubuntu-es.org/Sobre_Ubuntu.

34. Calidad y software. [En línea] [Citado el: 02 de 02 de 2011.] http://www.calidadysoftware.com/testing/pruebas_funcionales.php.

Anexos

ANEXO 1

Entrevista a trabajadores y directivos de la DTU

Compañero(a):

Con motivo de la realización del trabajo de diploma para el desarrollo de un sistema para la Gestión de reportes en la plataforma PTARTV, el autor del mismo se encuentra realizando esta entrevista con personal de todas las áreas de trabajos de la DTU, en aras de conocer algunos criterios relevantes sobre el tema y de obtener información importante proveniente de expertos, la cual será de ayuda para la posterior implementación.

A continuación le presentamos una serie de preguntas seleccionadas por el autor del trabajo.

1. ¿Cuáles son los reportes que necesitan sean realizados en su área de trabajo?
2. Si al final del mes necesita realizar algún tipo de informe a sus superiores. De ser preciso menciónelos.

Glosario de términos

GNU/Linux: GNU con Linux es la denominación defendida por Richard Stallman y otros para el sistema operativo que utiliza el kernel Linux en conjunto con las aplicaciones de sistema creadas por el proyecto GNU.

Distribución GNU/Linux: Es un conjunto de aplicaciones reunidas que permiten brindar mejoras para instalar fácilmente un sistema GNU/Linux. En general, se destacan por las herramientas para configuración y sistemas de paquetes de software a instalar.

Aplicación: Es una clase de programa informático creado para facilitar al usuario un determinado tipo de trabajo. Esto lo caracteriza frente a otros programas como los sistemas operativos, las utilidades y los lenguajes de programación.

API: Conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.

XML: Extensible Markup Language (lenguaje extensible de marcado), Formato estándar para el intercambio de datos basado en archivos de texto plano con una estructura de tags.

Navegador: Aplicación capaz de interpretar las órdenes recibidas en forma de código HTML fundamentalmente y convertirlas en las páginas que son el resultado de dicha orden, permitiendo así la comunicación con el servidor.

UCI: Universidad de las Ciencias Informáticas.

PTARTV: Plataforma de trasmisión abierta para radio y televisión.

Transmisión Abierta: Señal de libre acceso por la cual no es necesario pagar para consumirla.