Universidad de las Ciencias Informáticas

Facultad 5

"Componente para la configuración de Reportes Web"

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

Autor(es): Dayanis Pacheco Márquez.

Sandra Rangel Garcés.

Tutores: Ing. Raudi Agdel Bacallao Sánchez.

Ing. Ernesto Leyva Barrero

Ciudad de la Habana, 2011

DECLARACIÓN DE AUTORÍA.

Por este medio se declara que somos los únicos autores de este trabajo y se autoriza a la Universidad de las Ciencias Informáticas (UCI) para que haga el uso que estime pertinente con este trabajo.

Para que así conste se firma la presente a los __ días del mes de_____ del 2011.

_____	_____
Firma del Autor	Firma del Autor
(Dayanis Pacheco Márquez)	(Sandra Rangel Garcés)
_____	_____
Firma del Tutor	Firma del Cotutor
(Ing. Raudi Agdel Bacallao Sánchez.)	(Ing. Ernesto Leyva Barrero)

DATOS DE CONTACTO.

Tutor: Ing. Raudi Agdel Bacallao Sánchez

Edad: 27 años.

Ciudadanía: cubana.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ingeniero en Ciencias Informáticas.

Categoría Docente: Profesor Adiestrado.

E-mail: rabacallao@uci.cu.

Graduado de Ingeniero en Ciencias Informáticas y profesor de la UCI, con 3 años de experiencia en el desarrollo de software.

Tutor: Ing. Ernesto Leyva Barrero

Edad: 25 años.

Ciudadanía: cubana.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ingeniero en Ciencias Informáticas.

Categoría Docente: Profesor Adiestrado.

E-mail: ebarrero@uci.cu.

Graduado de Ingeniero en Ciencias Informáticas y profesor de la UCI, con 2 años de experiencia en el desarrollo de software.

AGRADECIMIENTOS

Dayanis:

Dicen que madre es una sola, yo no lo creo así. Por mucho que quiera disfrazar estas palabras para que reflejen todo mi agradecimiento, todo intento resulta insatisfactorio. Le agradezco con toda mi alma, a mi madre, que me trajo al mundo y me enseñó los primeros pasos de mi vida; que tanto me quiere y me ha soportado con mis resabios. A ella que podrá ver en su hija, 22 años después, la realización de lo que pudo llegar a ser, a ella por estar tan orgullosa de mí, por ser la luz que me guía por el buen camino. A mi abuelita, que la verdad, mi acervo cultural no me permite plasmar la dimensión de todo lo que esa viejita significa para mí. A ella, mi otra madre, la que más se ha preocupado por mí desde siempre, la que me crió desde que era una bebé, le dedico este resultado. A mi hermanita Leyanis que siempre que me sentía mal estaba a mi lado, aunque por ser pequeña siempre tenía el don para hacerme reír. A mis primas y tías por confiar en mí. A todos mis amigos, y prefiero pecar antes de dejar a alguien fuera, Yoana, Libeidy, Liusba, Liudmila, Irenna y sobre todo a Sandra que además de ser mi compañera de tesis es mi amiga, les agradezco por permitirme ser su amiga y por ser todos para mí, parte de mi familia. Y un parte especial para aquellos amigos que tanto nos ayudaron en la confección de este documento, Yuri, Dariel, Samada (ojalá no olvide a nadie). A nuestros tutores Ernesto y Raudi por su valiosa ayuda y por tanto defendernos y a Yoel, que sintió las bragas de esta tesis junto a nosotras.

Sandra:

A Dios por todas las personas y momentos lindos que ha puesto en mi vida. A mi abuelita de Zulueta por todo el amor y la crianza que me diste, se que desde el cielo estás orgullosa de mi y de lo que he logrado hasta el día de hoy. A mi mamita por ser mi mejor amiga, por conocerme más que yo misma, por estar siempre en los momentos difíciles y fáciles de mi vida, por no dejarme caer nunca y estar siempre ahí para mí, por ser la mejor mamita del mundo. A mi papito por sus consejos, su cariño y todo su amor, por ser un ejemplo para mi toda mi vida. A mi abuela de aquí de la Habana por ser mi tercera madre, por sus consejos, su amor, su preocupación y estar siempre que la he necesitado. A mis hermanos José Carlos, Daniel y Alex David por ser partes de mi vida, por todo su cariño y por ser los mejores hermanitos del mundo. A mi hermanita Daniela, aunque eres muy pequeñita ya estás dando alegrías a mi vida. A toda mi familia que de una forma u otra me han ayudado a llegar a donde estoy, a mis abuelos, tías, tíos y primos, y muy especial a mis primas Tata, Daine, Lisbe, Leydi y Anaelis que son como mis hermana y a mi madrina por ser la mejor del mundo. A Jorgito por su preocupación, por querer tanto a mi mamá y ser como un padre para mí. A Yune por hacer a mi papá cada día feliz. A mi novio por todo su cariño, amor, comprensión y amistad, sus consejos y también sus regaños, por ayudarme tanto y soportarme con mis defectos y mis virtudes, sin ti no hubiera sido posible, eres lo mejor que me ha pasado en la vida. A mi suegra por su preocupación, su cariño y sus consejos. A mis amigas y amigos que han recorrido parte de su vida junto a mí, a Yadira, Greysi, Lilia, Lismary, Marinetsy, Yalaina, Lazi, Yanet, Yeila y Carlos en Zulueta, a Libeydis, Liusba, Yoana, Irenna, Liudmila, Aliuska, Dayana, Mailen, La Cecy, Annia, Sindy, Mailín, Alexis y Adriel aquí en la universidad. A Dayanis por ser mi mejor amiga en estos 5 años, por su ayuda, por sus consejos, y también por todos los pleitos que hemos tenido. A nuestros tutores Raudi y Ernesto por ser nuestra guía en esta investigación. A los amigos que aportaron tiempo y conocimientos en este trabajo Samada, Pimienta, Ubalquis, Riolvis, Roberto, Yuri, Dariel, Yoandry. A Ismelys y a Osniel por tener paciencia, ayudarme y ser tan buenos amigos para mí en este último año. A mis viejos y nuevos amigos de la UCI, por acompañarme en todo este difícil recorrido y haberme permitido estar junto a ellos, a las muchachitas del futbol tanto a las de la facultad como a las de la universidad y en especial a Lulu y las Yaris, al profe de futbol Ariel por enseñarme no solo a jugar mejor al futbol sino también a ser mejor cada día. A los profesores que he tenido en todo el transcurso de mi vida, los buenos y los no tan buenos, porque siempre me enseñaron algo importante. A Fidel y la Revolución, por haber creado esta universidad de excelencia.

RESUMEN.

Los sistemas SCADA, son sistemas de Supervisión, Control y Adquisición de Datos como lo indican sus siglas en ingles, y en su mayoría cuentan con mecanismos para la generación de informes. Este proceso de generación de informes ha constituido una tarea primordial y crítica en muchas industrias, ya que se dedican esfuerzos considerables para realizar operaciones básicas como son agrupaciones, saltos de página, y conseguir que todo salga correctamente impreso.

Es por esto que surgió la necesidad de desarrollar un módulo para la configuración y generación de reportes, que permitiera diseñar y generar reportes, que fuera lo suficientemente flexible para extraer y procesar de forma automática la información almacenada en las bases de datos históricas. Este módulo es una aplicación escritorio creado con la biblioteca gráfica GTK (GIMP - General Image Manipulation Program Toolkit) el cual se encuentra empotrado dentro del Editor de Configuración de Despliegues o HMI, lo que trae consigo su no portabilidad, que no se interactúe desde diferentes estaciones de trabajo con la aplicación y no poder ser accedido desde un navegador Web.

En aras de dar solución a este problema se realiza este trabajo de diploma cuyo principal objetivo es implementar un diseñador de reportes en la Web, sobre plataforma de software libre, donde este se desarrolle fuera del entorno de trabajo del HMI, que pueda ser accedido desde un navegador web y ser diseñado de acuerdo a las necesidades del cliente.

Siguiendo esta alta meta se obtuvo como resultado una aplicación Web que solventa todas las dificultades anteriormente mencionadas y que cumple con los objetivos trazados al inicio de su confección.

PALABRAS CLAVE

Clases, diseño, informe, reporte, SCADA, software, Web, XML.

TABLA DE CONTENIDOS.

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

INTRODUCCIÓN

Hoy en día las industrias no pueden ignorar el grave problema que implica desarrollar y adaptar un software al ritmo que imponen los negocios. La globalización y fusión de empresas, el crecimiento de Internet, entre otros factores, han acentuado aún más estos problemas llevando el software desarrollado, que normalmente había sido creado para una plataforma específica, a un ambiente distribuido heterogéneo. Esto involucra, en consecuencia, la necesidad de considerar una amplia gama de aspectos como son la integración de datos heterogéneos, la interacción entre diversos sistemas, los distintos sistemas operativos, las tecnologías web, cuestiones de escalabilidad y rendimiento, por citar algunos de ellos.

Históricamente el desarrollo de la industria informática en Cuba, la necesidad de la inmersión del país en el mercado internacional digital y el fortalecimiento de la colaboración económica, son firmes y sólidas causas de desarrollo e investigación del sector informático en Cuba. En estos momentos se está dedicando gran parte de esta fuerza para el fomento del progreso de la economía nacional y venezolana. Esto se ha encausado debido a los fuertes lazos de amistad entre ambos gobiernos, que se hicieron oficiales en el Salón Ayacucho del Palacio de Miraflores, el 30 de octubre de 2000.

Producto a las relaciones Cuba-Venezuela, y a los proyectos existentes con la Universidad de las Ciencias Informáticas y a los precedentes que fueron surgiendo en muchas empresas y universidades del país con respecto al desarrollo de SCADAs, que son sistemas de Supervisión, Control y Adquisición de Datos como lo indican sus siglas en inglés, se asume el proyecto “SCADA Nacional”, conformando un equipo multidisciplinario de estudiantes, profesores y especialistas automáticos de todo el país, para el desempeño de esta importante tarea que exigía sus componentes desarrollados en software libre, lo cual era un mundo nuevo por conocer.

Un sistema SCADA es una aplicación software especialmente diseñada para funcionar sobre ordenadores en el control de producción, proporcionando comunicación con los dispositivos de campo y controlando el proceso de forma automática desde la pantalla del operador. Además, provee a diversos usuarios, toda la información que se genera en el proceso productivo, control de calidad, supervisión y mantenimiento. [17]

Procesos, Gestión de Archivos y de Datos, Comunicación e Interfaz Gráfica del Operador (HMI) entre otros. El módulo HMI proporciona al operador las funciones de control y supervisión de la planta, este proceso se representa mediante sinópticos gráficos. Estos sinópticos son

almacenados en la computadora consola de operación y generados desde un editor gráfico que puede estar incorporado al SCADA o importados desde otra aplicación durante la configuración del sistema.

Los sistemas SCADA en su mayoría cuentan con mecanismos para la generación de informes. Este proceso de generación de informes ha constituido una tarea primordial y crítica en muchas industrias, ya que se dedican esfuerzos considerables para realizar operaciones básicas como son agrupaciones, saltos de página, y conseguir que todo salga correctamente impreso.

Mediante lo antes mencionado surgió la necesidad de desarrollar un módulo para la configuración y generación de reportes en el SCADA Nacional que permitiera diseñar y generar reportes, que fuera lo suficientemente flexible para extraer y procesar de forma automática la información almacenada en las bases de datos históricas. Este módulo cuenta con dos herramientas fundamentales un Editor de Reportes para el diseño o configuración de las plantillas de informes que luego el Generador de Reportes genera los informes a diversos formatos como PDF, HTML o CSV.

El Editor de Reportes es una aplicación de escritorio creado con la biblioteca gráfica GTK (GIMP - General Image Manipulation Program Toolkit) que se encuentra empotrado dentro del Editor de Configuración de Despliegues o HMI, lo que trae consigo su no portabilidad, que no se interactúe desde diferentes estaciones de trabajo, para esto debe ser instalado el módulo en la estación directa donde se vaya a desarrollar cualquier operación y además de no poder ser accedido desde un navegador web.

De aquí la necesidad de desarrollar un componente para la configuración de reportes utilizando las tecnologías del lado del cliente para el desarrollo de un Editor de Reportes Web, que solvente las dificultades antes expuestas, donde este se desarrolle fuera del entorno de trabajo del Editor de Configuración de Despliegues, que pueda ser accedido desde cualquier estación de trabajo mediante los diferentes navegadores web, pueda ser diseñado de acuerdo a las necesidades del cliente, que el consumo de recursos del lado del cliente sean más económicos y factibles para el desempeño de trabajo del mismo. Que sus funcionalidades conlleven a su correcto uso y que pueda ser utilizado en un SCADA u otra aplicación que lo necesite.

Para dar solución a la situación problemática existente se plantea el siguiente **problema científico**: ¿Cómo dotar al sistema SCADA de un Editor de Reportes que se desarrolle fuera del Editor de Configuración de Despliegue y que permita ser accedido desde un navegador Web?

El **objeto de estudio** en la presente investigación es la configuración o diseño de un Editor de Reportes dentro del cual se define como **campo de acción** la elaboración y configuración de un Editor de reportes basado en tecnologías Web.

El **objetivo** del trabajo es implementar el módulo de configuración o diseño de Reportes Web usando herramientas y tecnologías bajo preceptos de software libre.

La **idea a defender** que se plantea es la siguiente:
Con el desarrollo de un módulo para la configuración o diseño de reportes en la web, se podrá acceder desde un navegador web e interactuar con el mismo desde diferentes estaciones de trabajo.

De acuerdo al objetivo expuesto se definen las siguientes **tareas investigativas** para lograr un mayor nivel de detalle en la investigación:

- Realizar un estudio del arte y elaborar una fundamentación teórica sobre los editores de reportes en la web.
- Realizar un estudio y selección de tecnologías adecuadas para el desarrollo del editor de reportes web.
- Realizar el diseño de la interfaz de usuario de forma tal que la aplicación tenga similitud con la versión de escritorio.
- Implementar el módulo de configuración o diseño de reportes web siguiendo la arquitectura del módulo de escritorio.
- Diseñar las pruebas de interfaz del módulo de configuración o diseño de reportes web.
- Validar el módulo de configuración o diseño de reportes web mediante pruebas de interfaz.

Esperando obtener como **resultados** los enunciados a continuación:

- Un módulo para el diseño o configuración de reportes web que reúna las características y funcionalidades necesarias para su correcto uso y funcionamiento en un SCADA u otra aplicación que lo necesite.
- Un software que cumpla con los objetivos y requerimientos para los cuales fue implementado efectuando el diseño y ejecución de las pruebas necesarias.

Métodos de Investigación:

En la realización de este trabajo se utilizaron diferentes métodos científicos para estudiar las características del objeto de investigación:

Métodos Teóricos:

- **Analítico – Sintético:** Se utiliza para procesar teorías y la información obtenida por vía empírica, permite la extracción de los elementos más importantes que se relacionan con los módulos de configuración y diseño de reportes.

- **Análisis histórico – lógico:** Para conocer, con mayor profundidad, los antecedentes y las tendencias actuales referidas a los módulos de configuración y diseño de reportes.

Métodos Empíricos:

- **Observación:** Se utiliza para observar los editores ya existentes, para así recopilar los puntos más importantes que sean necesarios en la aplicación para obtener un producto final con la calidad necesaria.

El presente trabajo de diploma está estructurado en tres capítulos:

En el **Capítulo 1** se realiza un esbozo teórico centrado en los generadores de informes, editores de reportes en la Web y las principales tendencias y tecnologías actuales que sirven de apoyo a todo el trabajo desarrollado.

En el **Capítulo 2** se analiza la arquitectura propuesta y se explican las principales funcionalidades.

El **Capítulo 3** se dedica a las pruebas realizadas para garantizar la integridad de la aplicación desarrollada.

CAPÍTULO 1

FUNDAMENTACIÓN TEÓRICA

En este capítulo se hace alusión a algunos conceptos y características generales de los generadores de reportes y la presencia en alguno de ellos de diseñadores de informes. Además se aborda acerca de los Reportes en la Web de los cuales se mencionan las características más importantes y la necesidad de integrarlos con la aplicación actual, se exponen algunos diseñadores de Reportes en la web con el objetivo de obtener de ellos las principales tecnologías que usan para su desarrollo y correcto funcionamiento dependiendo de las necesidades del cliente, teniendo en cuenta también las tendencias fundamentales en el desarrollo del software.

1.1 Generalidades sobre los generadores de Reportes.

Los generadores de reportes se componen fundamentalmente de dos elementos básicos, un diseñador o editor de informes y un motor de generación.

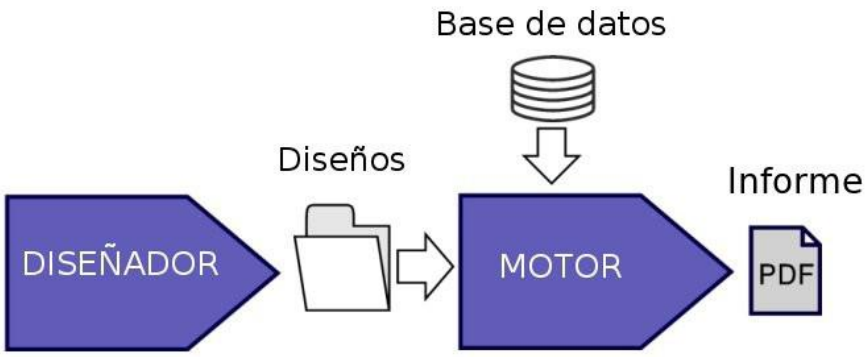


Fig. 1: Estructura general de un generador de reportes. Esquema de funcionamiento.

Como muestra la figura, las principales tareas que debe realizar un generador de reportes son las siguientes:

- Configurar el diseño y la apariencia que van a tener los reportes.
- Obtener los datos necesarios y elaborar el informe final de forma ordenada con la apariencia previamente configurada o diseñada.

Es por ello, la importancia de una herramienta capaz de llevar a cabo el diseño o edición de los reportes, que permita la configuración de plantillas deseadas con un formato determinado, de manera tal que el generador posteriormente las utilice para volcar los datos proveniente de las bases de datos y lograr generar el reporte deseado con la apariencia definida en la edición obtenida de la plantilla. Se conoce que varios generadores de informes presentan solo el componente de generación o motor generador, y la manera que tienen para diferenciar la apariencia visual de un reporte a otro es modificando las plantillas, a las cuales les definen una estructura y formato por defecto. Esta técnica es muy usada pero un poco engorrosa, puesto que el diseño del reporte final no se ve como se desea.

1.2 Diseño de informes.

El diseño de un reporte o informe no es más que una plantilla que modela y encierra las características con las cuales el usuario quiere que se genere el reporte deseado. De esta manera una vez que se insertan todos los componentes en tiempo de diseño, en las distintas secciones: Encabezado de Reporte, Encabezado de Página, Detalles, Pie de Página y Pie de Reporte, persistirán en la plantilla estas características y componentes para que a la hora de generarlos, el reporte se vea cómo mismo se diseñó pero con los datos volcados en su interior. Es válido aclarar que los diseños de reportes que se deseen crear deben tener la apariencia adecuada para un sistema SCADA u otra aplicación, por lo cual será necesario definir y especificar toda una serie de propiedades y mecanismos que permitan crear plantillas que puedan ser usadas desde el punto de vista funcional en un SCADA u otra aplicación que lo necesite.

1.3 Los Reportes en la Web.

La necesidad de un sistema de Reportes se vive como primordial en todos los niveles de acceso a la información de una empresa. Es fundamental, tanto en el caso de un operador que precisa conocer el avance de determinado proceso, como para obtener el material de gestión que permita conducir el destino del negocio.

A partir de los avances tecnológicos en el desarrollo de reportes se agrega un nuevo tipo de presentación de información denominada *Reportes Web* donde los usuarios tienen la posibilidad de

especificar una salida personalizada para las consultas dentro de la propia herramienta. Estos Reportes Web, introducen a la presentación de los datos devueltos por las consultas, todas las ventajas de un diseño HTML, acorde a las necesidades del usuario además de facilidades para la impresión de la información.

1.3.1 Inconvenientes de los Reportes en la Web.

Si se pretende ver los reportes vía Web, se debe tener en cuenta una serie de inconvenientes que suelen presentarse al intentar obtener información de un sistema, como:

- Que la base de datos no esté disponible vía Web o es muy complicado extraer datos para exportarlos al esquema Web.
- La seguridad es un punto siempre débil y crea una barrera al momento de hacer disponible la información a través de la Web.
- La información deberá estar disponible en dos formatos que sean totalmente diferentes.
- Se tendrá que permitir el acceso a diferentes perfiles y redes.

Debe ser considerado que para enfrentar un camino de soluciones a la problemática antes expuesta, hay que pensar en los Reportes Web como un sistema que integre la aplicación actual y proyecte el crecimiento buscado, contando con una plataforma ágil que permita implementar soluciones al alcance de los recursos disponibles.

1.3.2 Ventajas de los Reportes en la Web.

Las soluciones Web son capaces de ofrecer magníficas ventajas que facilitan el trabajo de muchos de los equipos de desarrolladores que siguen los avances tecnológicos desde muy cerca, las cuales son utilizadas de acuerdo a las necesidades de cada uno de estos equipos, como son:

- Distribución eficiente e inmediata de información a los usuarios de la red que tengan el servicio.
- Reducción de tiempos para la toma de decisiones.
- Ahorro de tiempo para obtener reportes.
- Ahorro en los costos de insumos de impresión de documentos.
- Prácticamente no existe la necesidad de capacitar a los usuarios para obtener reportes.

De esta forma se hace necesario juntar las ideas y crear las bases para la implementación de un componente para la configuración de Reportes en la Web, teniendo en cuenta las necesidades del cliente, el sistema debe ser lo más sencillo y amigable posible y a su vez de una manera robusta y fiable, debe permitir la visualización del reporte final siendo utilizadas las tecnologías evaluadas para el desarrollo del mismo.

1.4 Editores de Reportes en la Web.

Como parte del estudio realizado de los diferentes Editores de Reportes en la Web y que a su vez permitan la edición de los mismos, se muestra a continuación un resumen de los más importantes en cuanto a las funcionalidades básicas y la distribución de los componentes. De dichas herramientas se hizo un análisis con el objetivo de obtener algunas ideas y conceptos importantes.

1.4.1 RBTData.

Es una herramienta de diseño de reportes y consultas, basada en web, amigable y gratuita, con una aplicación cliente DHTML y un soporte a Java Enterprise. La aplicación Diseñador está hecha para proveer una interfaz poderosa, para usuarios no técnicos, respaldada por la tecnología *Java 2 Enterprise Edition* (J2EE) escalable. La aplicación Administrador provee una instalación de seguridad basada en roles y control de acceso a usuarios o grupos para las tablas y columnas de la base de datos. Nombres de usuarios pueden ser asignados hacia las tablas y columnas de la base de datos para mostrar todas las áreas de la aplicación Diseñador, para mejorar las consultas y diseño de reportes para usuarios no técnicos. En el Diseñador, los usuarios son presentados con una simple vista de tipo árbol, para la selección de columna y tabuladores para filtrar y diseñar salidas HTML. [15]

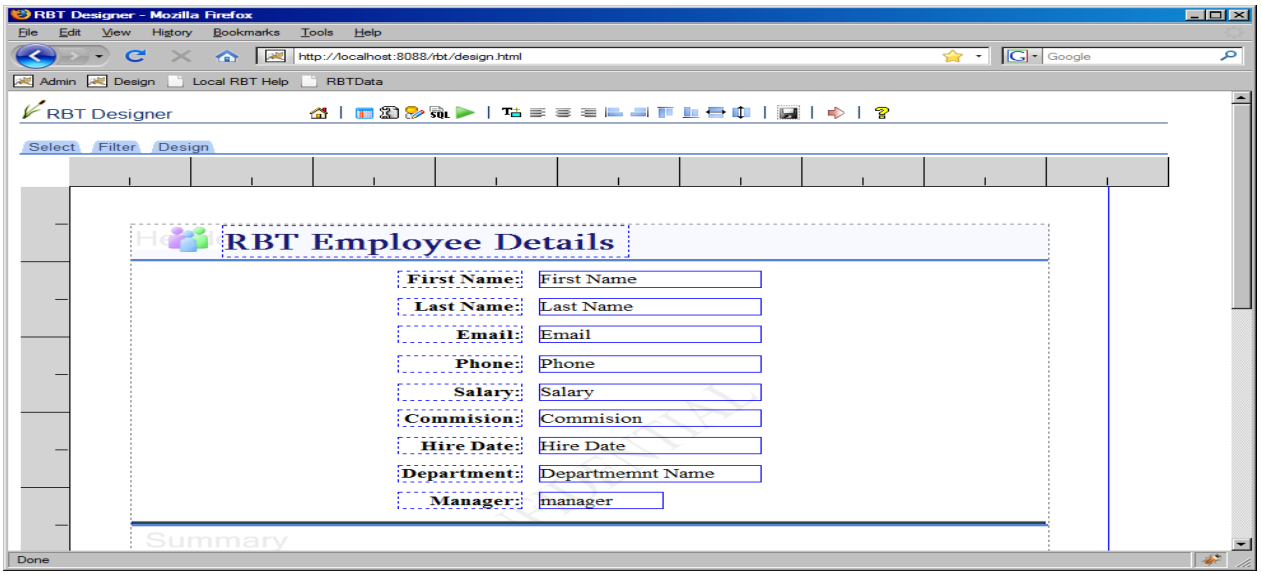


Fig. 2: Ambiente de Edición de RBTData

1.4.2 Stimulsoft.

Es el primer diseñador de reportes que permite editar completamente Reportes en la Web. No hay necesidad de instalar el framework .NET, componentes ActiveX u otros complementos adicionales especiales en la computadora del cliente, todo lo que necesita es un explorador Web. Stimulsoft Reports Designer Web es una interfaz moderna y útil, con funcionalidad enriquecida y gran velocidad de trabajo, es decir, no demora en exportar el diseño antes previsto. Stimulsoft tiene una interfaz estilo Ribbon (cinta), expone todas las funciones de un programa en un solo lugar, completamente funcional, constructor visual de componentes, tiene la capacidad para deshacer - rehacer, trabajar con el portapapeles, edición tabulada, amplificación (zooming) y muchas otras funciones útiles para el desarrollo establecido. Este diseñador no tiene necesidad de agregar los manejadores de eventos para guardar o cargar en la base de dato. [15]

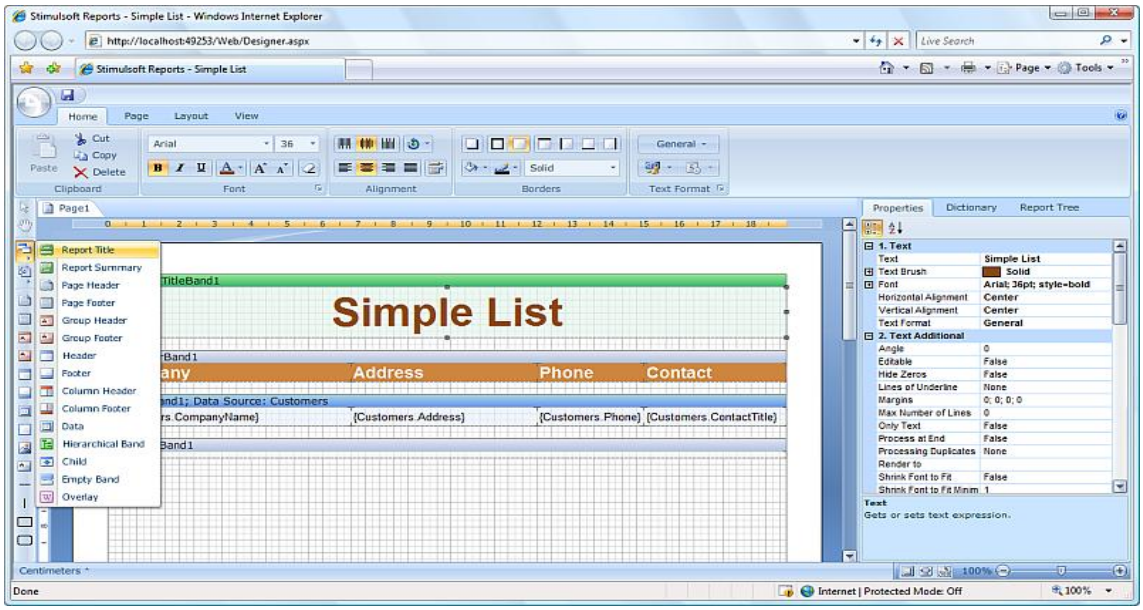


Fig. 3: Pantalla de la aplicación de Stimulsoft Reports Designer Web.

1.5 Principales Tendencias y Tecnologías.

Es una realidad de estos días las tendencias de muchas empresas, instituciones y personas civiles, que dedican tiempo a pensar en el uso o no de algunas tecnologías, con el objetivo de enfocarse y encaminarse por una de ellas, la que creen que cumple de forma soberana y equitativa los conceptos más importantes para los usuarios, como son libre distribución, código abierto, entre otros. Es por ello que cada día crece el uso de software libre a nivel mundial, pues en la medida de que nos damos cuenta que los software realizados sobre plataformas propietarias, se ven abarrotados de licencias, no muestran su código fuente, y el proceso de mantenimiento o cambio de versiones es realizado puramente por la compañía propietaria y en todos estos procesos se invierten grandes sumas de dinero para poder adquirir y mantener un producto determinado.

El hecho de desarrollar un sistema sobre plataforma libre es primeramente garantía de que el producto final pueda ser modificado por quien lo desee y de esta manera poder hacerle las adaptaciones o mejoras que este estime conveniente.

1.5.1 Software libre.

Es todo software cuyo "código fuente", es conocido y además puede ser difundido libremente. Otro aspecto importante es que el autor de dicho software lo puede publicar y permitirle a cualquier persona o institución la modificación o el simple uso de su obra. Existen dos formas de distribuir el

software, una es bajo las llamadas licencias sin restricciones de tipo BSD y la otra es bajo el uso de la GPL de la *Free Software Foundation* que plantea que se puede modificar el software pero sí y sólo si, el nuevo producto mantiene las mismas condiciones de uso y licencia del original y a su vez siga estando libremente al alcance de los que lo deseen.

1.5.2 Metodologías de Desarrollo de Software.

Las metodologías de desarrollo de software definen cómo trabajar eficientemente evitando las problemáticas que conllevan al fracaso de muchos proyectos. El objetivo fundamental de una metodología es aumentar la calidad del software a producir en todas las fases de desarrollo del mismo, haciendo énfasis en la calidad y menor tiempo de construcción del software o lo que es lo mismo “producir lo esperado en el tiempo esperado y con el coste esperado” (MOLPECERES). Las metodologías de desarrollo de software se dividen en dos grupos, las llamadas “pesadas” y las que se conocen como “ágiles”, como sus nombres lo indican ambos grupos tienen marcadas diferencias y la razón del uso o no de alguna de ellas está dada por la medida que el equipo de desarrollo del software determina el gran tamaño del producto o la simplicidad del mismo.

1.5.2.1 Metodologías “Pesadas”.

Estas metodologías en el transcurso de los años han demostrado ser eficientes y muy efectivas en la realización o desarrollo de grandes proyectos de software. En las mismas se lleva bien claro la definición del proceso en su totalidad en cuanto a roles, actividades, artefactos y casos de uso. Aunque muchas personas plantean que son muy exigentes con la documentación, otras piensan que es la manera de que el producto que va creciendo tenga la calidad y la documentación necesaria.

1.5.2.1.1 *El Proceso Unificado de Rational (RUP):*

Como sus siglas en inglés lo indican su nombre es Rational Unified Process(RUP). Esta es, de las metodologías existentes, una de las más generales y completas. Su surgimiento se remonta a aproximadamente treinta años atrás, lo cual da una medida de lo bien estructurada y refinada que está. Fue creada por James Jacobson y se puede decir que unifica los mejores elementos de las metodologías anteriores. Además está preparada para desarrollar grandes y complejos proyectos. Está enfocado al uso del paradigma orientado a objetos y utiliza UML como su lenguaje de representación visual. Dentro de sus principales características están, que es guiado por casos de

usos, iterativo e incremental y centra su arquitectura al esqueleto del producto como tal. También es válido destacar que se divide en cuatro fases importantes Inicio, Elaboración, Construcción y Transición.

1.5.2.2 Metodologías "Ágiles".

Las metodologías ágiles brindan facilidades y soluciones que resultan en muchos casos ser las ideales para una gran cantidad de proyectos. Estas metodologías se caracterizan por su sencillez, tanto en aprendizaje como en la aplicación de las mismas, reduciéndose así los costos en tiempo y recursos para que un equipo determinado la aplique. Es por ello que han revolucionado su uso por parte de numerosos grupos de desarrollo de software, pero hay que tener presente una serie de inconvenientes y restricciones para su aplicación, y una de ellas es que están dirigidas a equipos medianos o pequeños, donde el equipo se desarrolle en un ambiente que permita la comunicación y colaboración entre todos sus miembros durante todo el tiempo, cualquier inconveniente que presente del cliente o cualquier resistencia que emita el equipo de desarrollo puede llevar el proceso al fracaso, es por ello que el clima de trabajo, la colaboración y la relación equipo-cliente son claves para el éxito del producto final.

1.5.2.2.1 Programación Extrema (XP).

La metodología XP (en español Programación Extrema) es una de las metodologías ágiles más usadas en la actualidad. XP intenta reducir la complejidad por medio de un trabajo orientado directamente al objetivo, basado en las relaciones interpersonales y la velocidad de reacción. Es de vital importancia en su uso y aplicación que exista en todo momento un representante o miembro de la parte del cliente junto al equipo de trabajo con el objetivo de responder a cualquier duda o aclaración que necesite cualquier miembro y de esta forma no perder tiempo, además de que el software vaya creciendo justo como lo necesita y desea el cliente. Además esta metodología basa su implantación y desarrollo en las llamadas User Stories (en español, historias escritas) por el cliente en las que describe escenarios sobre el funcionamiento del sistema y que no sólo están limitados a la interfaz de usuario, sino que también pueden describir modelos, dominio, etc.

1.5.2.2.2 OpenUp.

Esta metodología es un proceso mínimo y suficiente, solo el contenido fundamental y necesario es incluido. Por lo que no provee reglas para todos los elementos que se manejan en un proyecto pero tiene los componentes básicos que pueden servir de base a procesos específicos. La mayoría de

los elementos de *OpenUp* están declarados para fomentar el intercambio de información entre los equipos de desarrollo y mantener un entendimiento compartido del proyecto, sus objetivos, alcance y avances.

Está organizado en dos dimensiones diferentes el método y el proceso. El contenido del método es donde los roles, tareas, artefactos y reglas son definidos, sin tener en cuenta como son utilizados en el ciclo de vida del proyecto. El proceso es donde los elementos del método son aplicados de forma ordenada en el tiempo de desarrollo. Estructura el ciclo de vida de un proyecto en cuatro fases: concepción, elaboración, construcción y transición. Este ciclo provee a los interesados un mecanismo de supervisión y dirección para controlar los fundamentos del proyecto, su ámbito, la exposición a los riesgos, el aumento de valor y otros aspectos.

1.5.3 UML como lenguaje de modelado.

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) es el sistema de software más conocido y utilizado en la actualidad; está respaldado por el *Object Management Group (*OMG*)*. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un "plano" del sistema llamado modelo, incluyendo aspectos conceptuales tales como procesos de negocios y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables.

Es válido destacar que UML es un "lenguaje" para especificar, no para describir métodos o procesos. Se utiliza para definir un sistema de software, detallar los artefactos en el sistema, documentar y construir. Se puede aplicar de diversas formas para dar soporte a una metodología de desarrollo de software tal como RUP, pero no especifica por si solo qué metodología o proceso utilizar.

UML no puede compararse con la programación estructurada, ya que no es programación, solo se diagrama la realidad de una utilización en un requerimiento. Mientras que, programación estructurada, es una forma de programar como lo es la orientación a objetos, sin embargo, la orientación a objetos viene siendo un complemento perfecto de UML, pero no por eso se toma UML sólo para lenguajes orientados a objetos.

1.5.4 Arquitectura de tres capas.

La programación por capas es un estilo de programación donde el objetivo primordial es la separación de la lógica de negocios de la lógica del diseño; un ejemplo básico de esto consiste en separar la capa de datos de la capa de presentación al usuario.

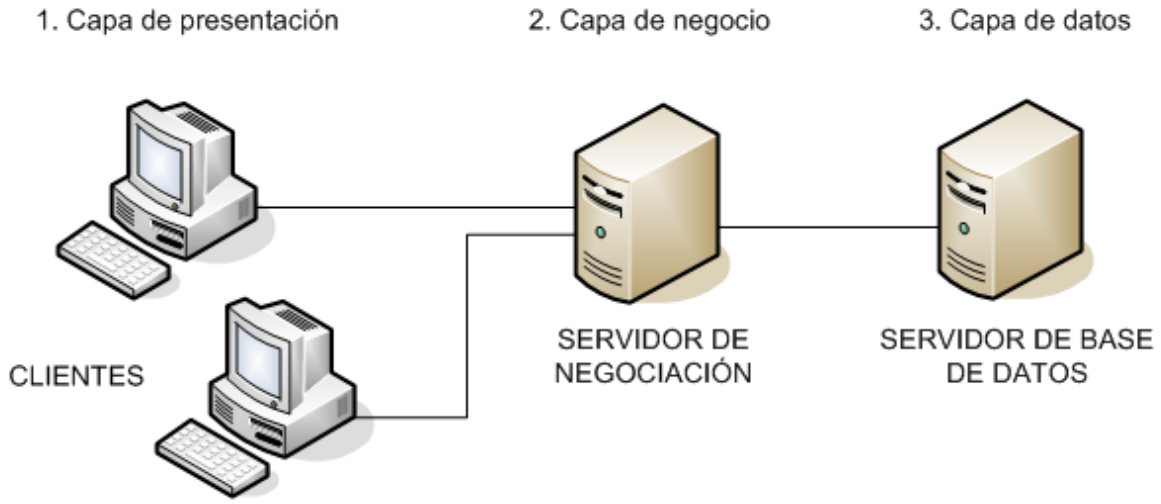


Fig. 4: Esquema Arquitectura en capas

La ventaja principal de este estilo es que el desarrollo se puede llevar a cabo en varios niveles y, en caso que sobrevenga algún cambio, sólo se ataca al nivel requerido sin tener que revisar entre código mezclado. Un buen ejemplo de este método de programación sería el modelo de interconexión de sistemas abiertos.

Además, permite distribuir el trabajo de creación de una aplicación por niveles; de este modo, cada grupo de trabajo está totalmente abstraído del resto de los niveles, de forma que basta con conocer la API que existe entre niveles.

En el diseño de sistemas informáticos actuales se suelen usar las arquitecturas multinivel o programación por capas. A cada nivel se le confía una misión simple, lo que permite el diseño de arquitecturas escalables que pueden ampliarse con facilidad en caso de que las necesidades aumenten. Este diseño es el más utilizado actualmente.

Todas las capas pueden residir en un único ordenador, aunque lo más usual es que haya una multitud de ordenadores donde reside la capa de presentación. Las capas de negocio y de datos pueden residir en el mismo ordenador, y si el crecimiento de las necesidades lo aconseja se pueden separar en dos o más ordenadores. Si el tamaño o complejidad de la base de datos aumenta, se puede separar en varios ordenadores los cuales recibirán las peticiones del ordenador donde se encuentre la capa de negocio, realizando solicitudes a una única base de datos. En

sistemas muy complejos se llega a tener una serie de ordenadores sobre los cuales corre la capa de negocio, y otra serie de ordenadores sobre los cuales corre la base de datos.

1.5.5 Patrón arquitectónico Modelo Vista Controlador (MVC).

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón de llamada y retorno MVC, se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos a la página. El modelo es el Sistema de Gestión de Base de Datos y la lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista.

1.6 Tecnologías del lado del cliente.

Se realizará un estudio de los diferentes framework existentes que proveen funcionalidades genéricas que pueden ser utilizadas para desarrollar aplicaciones de manera rápida, fácil, modular y sencilla, ahorrándose tiempo y esfuerzo; como es el caso del Editor. Ya que se necesita rectificar las deficiencias antes señaladas **(Introducción pág.2 párrafo 3)** para realizar un editor de Reportes sobre la Web que cumpla con todos los requisitos que necesita un SCADA. Por tal motivo se necesita hacer un análisis profundo de las tendencias y tecnologías actuales, con potencialidades para desarrollar ambientes gráficos eficientes.

Cada framework tiene sus características que los diferencian de los demás. Podemos destacar los siguientes:

1.6.1 jQuery.

1.6.1.1 Descripción

jQuery fue presentada el 14 de enero de 2006 en el Bar Camp NYC, creada inicialmente por John Resig. Es una biblioteca o framework de JavaScript, que permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol Document Object Model (DOM), manejar eventos, desarrollar animaciones y agregar interacción con la tecnología AJAX a páginas Web.

Es de software libre y código abierto, posee un doble licenciamiento bajo la licencia del Instituto Tecnológico de Massachusetts (MIT) y la Licencia Pública General de GNU (LGPL)v2, permitiendo

su uso en proyectos libres y privativos. Al igual que otras bibliotecas, ofrece una serie de funcionalidades basadas en JavaScript que de otra forma requerirían de mucho más código, es decir, con las funciones propias de esta biblioteca se logran grandes resultados en menos tiempo y espacio.

1.6.1.2 Características

Este framework cuenta con una amplia gama de características que lo hacen más vistoso y ameno a los ojos del cliente, como son:

1. Selección de elementos DOM.
2. Maneja eventos.
3. Manipulación de la hoja de estilos CSS.
4. Efectos y animaciones.
5. Animaciones personalizadas.
6. Interacción con AJAX.
7. Soporta extensiones.
8. Obtiene información del navegador, opera con objetos y vectores, cuenta con funciones como trim() (elimina los espacios en blanco del principio y final de una cadena de caracteres).
9. Compatible con los navegadores Mozilla Firefox 2.0+, Internet Explorer 6+, Safari3+, Opera 9+ y Google Chrome 1+ 5.

1.6.2 Prototype.

1.6.2.1 Descripción.

Fue creado por Sam *Stephenson* para el desarrollo sencillo y dinámico de páginas Web. Es un framework escrito en JavaScript que implementa las técnicas AJAX y su potencial es aprovechado al máximo cuando se desarrolla con *Ruby On Rails*. Prototype simplifica parte del trabajo cuando se pretende desarrollar páginas altamente interactivas.

Se ha convertido en poco tiempo en una referencia básica de AJAX y es la base de muchos otros framework y bibliotecas relacionadas como script.aculo.us. Las primeras versiones de Prototype no incluían ningún tipo de documentación, lo que dificultaba su uso y provocaba que la mayoría de los usuarios desconocieran su verdadero potencial.

1.6.2.2 Características.

- **Desarrollo sencillo de aplicaciones AJAX**: Además de las llamadas sencillas, este módulo también proporciona una forma más amena de trabajar con el código JavaScript enviado por el servidor. También proporciona clases para facilitar las tareas de polling.

- **Extensión de DOM:** Añade varios métodos a los elementos retornados con la función $().
- **Utilidades JavaScript Object Notation (JSON):** Es una alternativa más ligera que las llamadas XML realizadas con AJAX.

1.6.3 Extjs.

1.6.3.1 Descripción.

Extjs empezó siendo un conjunto de bibliotecas y extensiones para Yahoo User Inteface (YUI). Con el tiempo se convirtió en un framework independiente y a principios de 2007 se creó una compañía para comercializar y dar soporte del framework Ext. De esta forma tiene dos tipos de licencias, LGPL y comercial. Tiene componentes para crear y manipular *DataGrids* gozando de cierta ventaja sobre otros frameworks. Es posible crear “ventanas” con barras de herramientas y menú con estilo de aplicaciones de escritorio, diálogos modales y eventos.

1.6.3.2 Características.

- **Ext.Element:** Representa un elemento del árbol DOM. Muchas de las funciones de manipulación de los elementos tienen un parámetro opcional que permite realizar el cambio mediante un efecto de animación, el cual puede ser un dato booleano o un objeto que incluye las opciones de la animación.
- **Ext.BorderLayout:** Representa un diseño común para ser usado en aplicaciones de escritorio.
- **Ext.DomHelper:** Utilidad para trabajar plantillas o DOM. Soporta el uso de DOM o fragmentos de HTML de forma transparente.
- **Ext.TabPanel:** Un ligero contenedor de tabs.
- **Ext.Updatemanager:** Proporciona soporte para actualización AJAX de los objetos *Element*.

1.6.3.3 Justificación.

Dispone de un conjunto de componentes *(widgets)* para incluir dentro de una aplicación web, como:

- Cuadros y áreas de texto.
- Campos para fechas.
- Campos numéricos.
- Combos.
- Radiobuttons y checkboxs.
- Editor HTML.
- Elementos de datos con modos de sólo lectura, datos ordenables, columnas que se pueden bloquear y arrastrar, etc.
- Árbol de datos.
- Pestañas.
- Barra de herramientas.
- Menús al estilo de Windows.
- Paneles divisibles en secciones.
- Sliders.

Varios de estos componentes están capacitados para comunicarse con el servidor usando AJAX. Contiene numerosas funcionalidades que permiten añadir interactividad a las páginas HTML, como: Cuadros de diálogo, *quicktips* para mostrar mensajes de validación e información sobre campos individuales.

1.6.4 Dojo.

1.6.4.1 Descripción.

Es un framework que contiene APIs (Application Programming Interface) y widgets (controles) para facilitar el desarrollo de aplicaciones Web que utilicen tecnología AJAX. Contiene un sistema de empaquetado inteligente que puede ser utilizado para enriquecer sitios web con varias características que trabajan a través de la mayoría de los navegadores, tales como: Menúes, Tabs, Tooltips y tablas ordenables. Dojo resuelve asuntos de usabilidad comunes como pueden ser la navegación y detección del navegador, soportar cambios de URL para luego regresar a ellas (bookmarking), y la habilidad de degradar cuando AJAX/JavaScript no es completamente soportado en el cliente.

1.6.4.2 Características.

- **Multiple Points Of Entry:** Es un concepto fundamental en el diseño de Dojo. Este término significa que los usuarios pueden empezar a utilizar Dojo al nivel que ellos deseen.

- **Independencia del interpretador:** Asegurar soporte para el mayor número de plataformas.

1.6.5 Mootools.

1.6.5.1 Descripción.

Liviano, modular y orientado a objetos, la meta es ser un intermediario para los desarrolladores ayudándolos a crear código JavaScript en una manera elegante, flexible y eficiente. Contiene un gran número de componentes, pero no todos necesitan ser cargados en cada aplicación. Consta de un Core, que es una colección de bibliotecas que el resto de sus componentes necesitan, Class, que es la biblioteca básica.

También provee de unos componentes que enriquecen a los objetos nativos de JavaScript, para agregar compatibilidad y simplificación de código. Fx es una API avanzado de efectos para animar elementos. Algunas aplicaciones que utiliza MooTools son: ape, bing, joomla, vimeo, palm, Nintendo, phpMyAdmin y netvibes.

1.6.5.2 Características.

- **Core:** Funciones principales, basado en prototype.
- **Class:** Permite la creación de clases reutilizables.
- **Native:** Simplifica el trabajo y definición de arrays, funciones, tipos de datos (numéricos y cadena) y elementos DOM.
- **Element:** Funciones adicionales para trabajar con elementos DOM (selección, filtros, formularios, eventos, dimensiones).
- **Windows:** funciones específicas para el objeto Windows (DomReady y cambio de dimensiones).
- **Remote:** Simplifica el uso de AJAX, Cookies, Json.
- **Effects:** Diferentes efectos de animación. Es parte de otro proyecto moo.fx (una biblioteca de efectos gráficos realmente ligera, 30Kb, compatible tanto con mootolols como con prototype).
- **Drag:** Funciones de drag and drop.
- **Plugins**: 10 utilidades adicionales (hash, scroller, sortables, tips, etc.)

1.6.6 Matriz de decisión.

Determinar cuál framework es mejor es prácticamente imposible. Es importante recordar que estos framework son herramientas que ayudan a realizar diferentes tareas, todos basados en el mismo lenguaje, *JavaScript*. Lo correcto es seleccionar uno u otro dependiendo de la utilidad y la capacidad de saber cómo utilizarlo. La eficiencia de estos está muy atada al navegador que utiliza el usuario final y los aportes de estos a las necesidades de los desarrolladores.

1.6.6.1 Selección de los criterios de evaluación para los framework.

A continuación se exponen una serie de criterios que deben ser valorados para la selección de los frameworks que cumplen con estos criterios y después seleccionar el mejor para los objetivos propuestos:

1. **Compatibilidad:** Agregan la posibilidad de escribir código JavaScript totalmente compatible con todos los navegadores y motores JavaScript más utilizados. Esto aumenta la portabilidad y elimina la incompatibilidad entre navegadores y sus motores intérpretes JavaScript.
2. **Comunicación asíncrona (Ajax):** Usando este acercamiento, es fácil utilizar XMLHttpRequest para manejar y manipular los datos en los elementos de un sitio, aumentando la interactividad y experiencia del usuario.
3. **DOM:** Maximizan la capacidad de agregar, editar, cambiar, eliminar elementos de manera dinámica agregando bibliotecas que facilitan usar DOM.
4. **Validación de Formularios:** Permiten validar campos dentro de uno o varios formularios. Esto, desde el punto de vista del desarrollador, simplifica y reduce el código para procesar dichos formularios, ya que los datos llegan previamente validados, reduciendo los errores de tipos de datos.
5. **Efectos visuales:** Utilizando la manipulación de los elementos, se pueden crear efectos visuales y animaciones. Entre los que se encuentran: aparecer y desaparecer, redimensionamiento, mover, entre otros.
6. **Almacenamiento Client-side:** Facilita funciones para leer y escribir cookies. Proveen una abstracción de almacenamiento que permite a las aplicaciones web guardar datos del lado del cliente, persistente y de manera segura.
7. **Manejo JSON:** Incrementa al máximo el manejo de datos que pueden ser utilizados para presentar informaciones de manera dinámica y en tiempo de ejecución.

8. **Manejo de Eventos:** Permite reaccionar de una forma u otra dependiendo de las acciones del usuario.
9. **Recibidores de Datos:** Permiten utilizar diferentes formatos de datos como XML, HTML, Texto, JSON, ATOM, entre otros.
10. **Arrastra y Suelta:** Conocido como Drag and Drop. Es una funcionalidad que brinda la posibilidad de arrastrar elementos dentro de una misma página que interactúe con el resto de los elementos.

1.6.6.2 Importancia.

Criterios	Importancia
• Compatibilidad	15
• Comunicación asíncrona (Ajax)	10
• DOM	10
• Validación de Formularios	10
• Efectos visuales	15
• Almacenamiento Client-side	5
• Manejo JSON	10
• Manejo de Eventos	10
• Recibidores de Datos	15
• Drag and Drop	10

Tabla 1: Importancia de criterios de selección de framework.

1.6.6.3 Selección de las principales tecnologías a utilizar.

Teniendo en cuenta los requerimientos de la aplicación se selecciona Extjs ya que dispone de un conjunto de componentes *(widgets)* para incluir dentro de una aplicación Web, como cuadro y aéreas de texto, campos para fecha y numéricos, combos, radiobuttons y editor HTML; componentes que se necesitan para el desarrollo del Editor de Reportes en la Web utilizando las ventajas que facilita este framework.

1.7 Tecnologías del lado del servidor

Se realizará un estudio de los lenguajes con más potencialidades para el desarrollo de un Editor de Reportes en la Web, ya que en el editor existente se han encontrado algunas deficiencias **(Introducción pág.2 párrafo 3)**, de aquí la necesidad de desarrollar un componente para la configuración de reportes utilizando las tecnologías del lado del servidor acordes para la realización del editor. Por tal motivo se necesita realizar un análisis profundo de las tendencias y tecnologías actuales, con potencialidades para desarrollar ambientes gráficos eficientes y que elimine las deficiencias presentadas en el Editor de Escritorio.

1.7.1 Hypertext Preprocessor (PHP)

1.7.1.1 Descripción

PHP es un lenguaje script interpretado en el lado del servidor, utilizado para la generación de páginas Web dinámicas, embebidas en páginas HTML y ejecutadas en el servidor. No necesita ser compilado para ejecutarse, para su funcionamiento necesita tener instalado Apache o Internet Information Server (IIS) con las bibliotecas de PHP. La mayor parte de su sintaxis ha sido tomada de C, Java y Perl con algunas características específicas. Los archivos cuentan con la extensión (*.php*).

Es un poderoso lenguaje e intérprete, ya sea incluido como parte de un servidor web en forma de módulo o ejecutado como un binario Gráficos de interfaz de la computadora (CGI) separado, es capaz de acceder a archivos, ejecutar comandos y abrir conexiones de red en el servidor. Estas propiedades hacen que cualquier evento que sea ejecutado en un servidor web sea inseguro por naturaleza.

1.7.1.2 Características

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

- **Soporte para una gran cantidad de bases de datos:** MySQL, PostgreSQL, Oracle, Microsoft SQL Server, Sybase SQL, Informix, entre otras.
- **Integración con varias bibliotecas externas:** Permite generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.
- **Ofrece una solución simple y universal:** Para las páginas dinámicas de la Web de fácil programación.

- **Fácil:** Es más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- **Soportado por una gran comunidad de desarrolladores:** Como producto de código abierto, PHP goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.
- **El código se pone al día**: Continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.
- **Todo el trabajo lo realiza el servidor y no delega al cliente**: Puede ser más ineficiente a medida que las solicitudes aumentan.
- **Legibilidad del código:** Puede verse afectada al mezclar sentencias HTML y PHP.
- **Programación orientada a objetos:** Es aún muy deficiente para aplicaciones grandes.

1.7.1.3 Justificación

- Soporta en cierta medida la orientación a objeto. Clases y herencia.
- Capacidad de expandir su potencial utilizando módulos.
- Posee documentación en su página oficial la cual incluye descripción y ejemplos de cada una de sus funciones.
- Es libre, por lo que se presenta como una alternativa de fácil acceso para todos.
- Incluye gran cantidad de funciones.
- No requiere definición de tipos de variables ni manejo detallado del bajo nivel.

Es un lenguaje que se puede utilizar tanto en sistemas simples como complejos, es fácil y rápido, pero a la vez potente y muy amplio, es multiplataforma, libre, interpretado, fácil de cambiar, es independiente de plataforma pues existe un módulo de PHP para casi cualquier servidor web, posee conexión con la mayoría de los gestores de base de datos que se utilizan en la actualidad como son: PostgreSQL, MySQL, Oracle y Microsoft SQL Server. Es un software de código abierto, por lo que se presenta como una alternativa de fácil acceso para todos.

1.7.2 Active Server Pages (ASP)

1.7.2.1 Descripción

Es una tecnología del lado de servidor desarrollada por Microsoft para el desarrollo de sitios web dinámicos. Las páginas web desarrolladas bajo este lenguaje necesitan tener instalado *Internet Information Server (IIS)*. No necesita ser compilado para ejecutarse. Existen varios lenguajes que se pueden utilizar para crear páginas ASP, el más utilizado es VBScript, nativo de Microsoft. ASP, también en Perl y Jscript (no JavaScript). El código ASP puede ser insertado junto con el código HTML. Los archivos cuentan con la extensión (*asp*).

1.7.2.2 Características

- Usa Visual Basic Script, siendo fácil para los usuarios.
- Comunicación óptima con SQL Server.
- Soporta el lenguaje JScript (Javascript de Microsoft).
- Código desorganizado.
- Se necesita escribir mucho código para realizar funciones sencillas.
- Tecnología propietaria.
- Hospedaje de sitios web costosos.

1.7.3 Python.

1.7.3.1 Descripción

Python es un lenguaje de programación interpretado, creado por Guido van Rossum en el año 1991. En la actualidad se desarrolla como un proyecto de código abierto, administrado por la Python Software Foundation. Es considerado como la "oposición leal" a Perl, lenguaje con el cual mantiene una rivalidad amistosa. Los usuarios de Python consideran a éste mucho más limpio y elegante para programar. Permite dividir el programa en módulos reutilizables desde otros programas. Viene con una gran colección de módulos estándar que se pueden utilizar como base de los programas. También hay módulos incluidos que proporcionan E/S de ficheros, llamadas al sistema, sockets y hasta interfaces a GUI (interfaz gráfica con el usuario) como GTK, Qt entre otros.

1.7.3.2 Características

- Es compatible con cualquier sistema o plataforma y existen versiones disponibles en muchos sistemas informáticos distintos. Originalmente se desarrolló para Unix y su compatibilidad está dada para cualquier sistema en el que exista un intérprete programado para él.
- Es un lenguaje de programación multiparadigma dando la posibilidad a los programadores de adoptar un estilo particular de programación, permite varios estilos: programación orientada a objetos, programación estructurada y programación funcional. Otros paradigmas están soportados mediante el uso de extensiones.
- Resolución dinámica de nombres; es decir, lo que enlaza un método y un nombre de variable durante la ejecución del programa (también llamado ligadura dinámica de métodos).
- Dispone de sintaxis extremadamente simple, es un lenguaje minimalista haciendo innecesario escribir caracteres como (;, {,},`\n`), etc. Existen muchas funciones incorporadas en el propio lenguaje, para el tratamiento de strings, números, archivos, etc. Además, posee muchas bibliotecas que podemos importar en los programas para tratar temas específicos como la programación de ventanas o sistemas en red. Su sencillez permite una alta velocidad de desarrollo.
- Soporta objetos y estructuras de alto nivel como cadenas, listas, diccionarios y proporciona varios niveles para organizar el código: funciones, clases, módulos y paquetes

  .

1.7.4 Java Server Pages (JSP)

1.7.4.1 Descripción

Es un lenguaje para la creación de sitios web dinámicos, orientado a desarrollar páginas web en Java. Es un lenguaje multiplataforma, creado para ejecutarse del lado del servidor. Comparte ventajas similares a las de ASP.NET. Posee un motor de páginas basado en los servlets de Java. Para su funcionamiento se necesita tener instalado un servidor Tomcat.

1.7.4.2 Características

- Código separado de la lógica del programa.
- Las páginas son compiladas en la primera petición.

- Permite separar la parte dinámica de la estática en las páginas web.
- Los archivos se encuentran con la extensión (jsp).
- El código JSP puede ser incrustado en código HTML.
- Ejecución rápida del servlets.
- Crear páginas del lado del servidor.
- Multiplataforma.
- Código bien estructurado.
- Integridad con los módulos de Java.
- La parte dinámica está escrita en Java.
- Permite la utilización de servlets.
- Complejidad de aprendizaje.

1.7.5 Matriz de decisión.

1.7.5.1 Selección de los criterios de evaluación para las tecnologías del lado del servidor.

A continuación se exponen una serie de criterios que deben ser valorados para la selección de las tecnologías que cumplen con estos criterios y seleccionar la más vinculada a los objetivos propuestos:

1. **Facilidad de implementación:** Pretende evaluar cuán difícil será implementar la solución con las tecnologías seleccionadas partiendo de las habilidades del equipo de desarrollo y de la documentación disponible respecto a dicha tecnología.
2. **Consumo CPU:** Que la tecnología seleccionada, de uso eficiente de la unidad de procesamiento en los algoritmos matemáticos.
3. **Consumo de Memoria:** Es determinante el uso eficiente de la memoria para cada tecnología, para no sobrecargar el sistema.
4. **Serialización:** Facilidad de hacer persistente la información en los mecanismos de salvar y cargar la información.
5. **Integración con otros lenguajes y/o bibliotecas gráficas:** Capacidades de interpolaridad con otros lenguajes y frameworks que permitan hacer más robusta la solución.
6. **Integración con bibliotecas para realizar pruebas de unidad:** Capacidad de las tecnologías para integrarse con herramientas en la realización de pruebas automatizadas.

7. **Portabilidad:** Que las tecnologías sean multiplataforma.
8. **Software libre:** Necesidad de desarrollar siempre con tecnologías libres.
9. **Cálculo de algoritmos recursivos:**
10. Tamaño del código.

1.7.5.2 Importancia.

Criterios	Importancia
1. Facilidad de implementación	15
2. Consumo CPU	15
3. Consumo de Memoria	15
4. Serialización	10
5. Integración con otros lenguajes y/o bibliotecas gráficas	15
6. Integración con bibliotecas para realizar pruebas de unidad	10
7. Portabilidad	10
8. Software libre	10
1. Cálculo de algoritmos recursivos	5
2. Tamaño de código	10

Tabla 2: Importancia de criterios de selección de tecnologías de lado del servidor.

1.7.5.3 Selección de las principales tecnologías a utilizar.

Teniendo en cuenta los requerimientos de la aplicación se selecciona PHP como mejor tecnología para este tipo de aplicación ya que es un lenguaje que se puede utilizar tanto en sistemas simples como complejos, es fácil y rápido, pero a la vez potente y muy amplio, es multiplataforma, libre, interpretado, fácil de cambiar, independiente de plataforma pues existe un módulo de PHP para casi cualquier servidor Web, posee conexión con la mayoría de los gestores de base de datos que se utilizan en la actualidad como son: PostgreSQL, MySQL, Oracle y Microsoft SQL Server. Es un software de código abierto, por lo que se presenta como una alternativa de fácil acceso para todos.

Entonces podemos determinar que PHP consta de:

- Soporte para una gran cantidad de bases de datos: MySQL, PostgreSQL, Oracle, MS SQL Server, Sybase mSQL, Informix, entre otras.
- Integración con varias bibliotecas externas, permitiendo generar documentos en PDF (documentos de Acrobat Reader) hasta analizar código XML.
- Ofrece una solución simple y universal para el paginado dinámico de la Web de fácil programación.
- Perceptiblemente más fácil de mantener y poner al día que el código desarrollado en otros lenguajes.
- Soportado por una gran comunidad de desarrolladores, como producto de código abierto, goza de la ayuda de un gran grupo de programadores, permitiendo que los fallos de funcionamiento se encuentren y reparen rápidamente.
- El código se pone al día continuamente con mejoras y extensiones de lenguaje para ampliar las capacidades de PHP.
- Con PHP se puede hacer cualquier cosa que se pueda realizar con un script CGI, como el procesamiento de información en formularios, foros de discusión, manipulación de cookies y páginas dinámicas.

1.8 Tendencias para el desarrollo.

En la actualidad existen varias tendencias a la hora de enfrentar el desarrollo de productos software de este tipo. Una de ellas es la de incorporar o integrar la solución que se desea realizar a herramientas ofimáticas. Otra es la de reutilizar alguna de las herramientas generadoras de informes existentes, realizando las modificaciones pertinentes para llevarlo a las necesidades del cliente, lo cual trae como consecuencia que se disponga de suficiente tiempo en el estudio de la solución planteada para comenzar a incorporarle las nuevas funcionalidades, sin saber si el resultado que se obtenga será un software estable. Además pudiéndose desarrollar la aplicación desde cero, partiendo solo de los componentes básicos que pongan a disposición las bibliotecas que se decidan emplear para el proyecto. De esta forma se dispondría de todo el código desarrollado y a la vez de toda la documentación, lo cual le daría homogeneidad a la solución final. Esta última parece ser la vía más trabajosa, pero es la que realmente da la medida de desarrollar un producto en su totalidad, además será más sencillo adaptarlo a las condiciones que sean necesarias, así como a la hora de intercomunicar la aplicación con otros subsistemas del SCADA sería aun más fácil, puesto que conocer el diseño total de una aplicación es la ruta al éxito en la comunicación con terceros.

1.9 Propuesta

Según todo lo antes expuesto se procede a realizar una propuesta de solución para esta investigación, la cual consiste en desarrollar un editor de Reportes Web usando una alternativa autónoma, donde no se pierdan las ideas y características de otras aplicaciones similares de las estudiadas y que incluya además las nuevas funcionalidades que necesita el SCADA u otra aplicación. El IDE que se utilizará para el desarrollo será Zend Studio.7, ya que permite agilizar el desarrollo de aplicaciones web y simplificar proyectos complejos. Como biblioteca gráfica se utilizará el framework Extjs ya que disponen de un conjunto de componentes para incluir dentro de una aplicación web como: cuadros y aéreas de texto, campo para fechas, editor HTML, paneles divisibles en secciones, barra de herramientas; entre otras. Se aplicará como metodología de desarrollo OpenUp ya que es un proceso mínimo y suficiente, solo el contenido fundamental y necesario es incluido y como patrón arquitectónico el Modelo Vista Controlador, en especial su variante llamado Modelo – Presentación, ya que este permite tener separada las responsabilidades en dos dominios.

1.10 Conclusiones

En este capítulo se realiza un esbozo de las principales características de los generadores de reportes, se hace un análisis de las tecnologías, conceptos, herramientas existentes y tendencias actuales a utilizar en la realización de esta investigación con la finalidad de proponerse una solución al problema investigativo planteado.

CAPÍTULO 2

2. CONSTRUCCIÓN DE LA SOLUCIÓN PROPUESTA.

En este capítulo se presentan las principales clases y funcionalidades del diseñador, relacionadas con los conceptos fundamentales que abarca, separándolos en escenarios en los cuales se explica detalladamente y se muestran los distintos diagramas del diseño. Además se evalúan los parámetros para una correcta implementación de los componentes. Para ello se describen las principales funcionalidades que deben cumplir, centrando el trabajo en las propiedades que deben manejar los mismos.

2.1 Interacción del módulo Diseñador de Reportes.

El funcionamiento del sistema a desarrollar puede dividirse en dos:

1. Por un lado se tiene un diseñador de informes, que proporciona al usuario una forma sencilla de diseñar una plantilla o diseño de informe. En esta plantilla el usuario tendrá la posibilidad de introducir imágenes, etiquetas de texto, componentes gráficos, y cajas de texto para indicar el origen de los datos procedentes de distintas bases de datos.

2. Por otro lado un motor de generación, que se encargará de extraer los datos de una o varias bases de datos, después enlazará estos datos con el diseño antes realizado, y finalmente generará un documento en base al diseño de informe obtenido en el diseñador. El documento resultante es un archivo que incluye todo lo citado anteriormente (diseño más datos), el cual se puede visualizar en pantalla o despachar a distintos medios, como la impresora o el disco duro en forma de archivo con un determinado formato para ser accedido mediante los servicios que se establezcan.

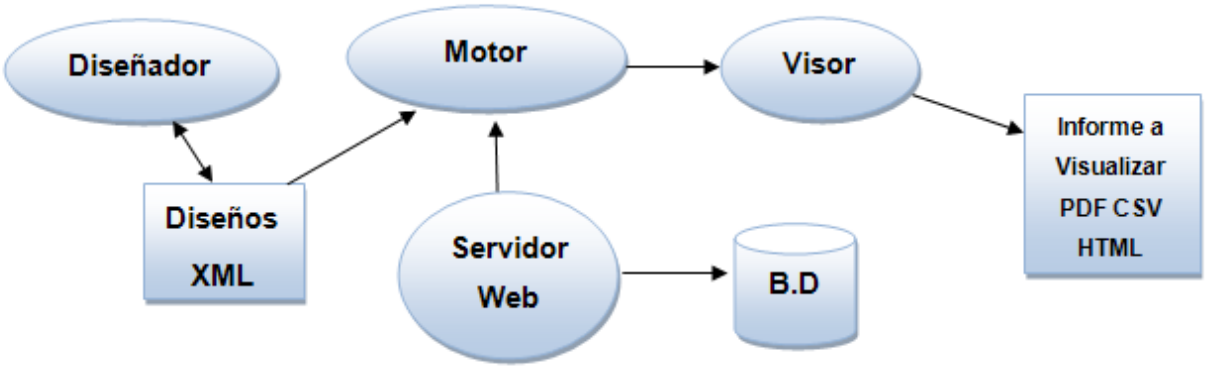


Fig. 5: Esquema funcional del sistema propuesto.

Como se puede apreciar en este esquema los diseños persistirán en un formato estándar (XML), para ser tomados lo mismo por el motor de generación para generar el informe ante solicitudes provenientes del SCADA, que por el diseñador para ser usados como diseños pre elaborados.

2.2 Patrones.

"Los patrones en general, y en particular los de arquitectura, son un intento de formalizar la comunicación y el reúso de la experiencia del diseño, describen los problemas recurrentes que surgen en determinados contextos y esquemas de soluciones probados. Son una herramienta para los no expertos, ofreciendo un paso más hacia la Ingeniería de Software (**ISW**); ya que permite que personas comunes hagan cosas que antes requerían virtuosismo (…) Los patrones no se proponen descubrir ni expresar nuevos principios de la ISW, todo lo contrario, intenta codificar el conocimiento, las expresiones y los principios ya existentes: cuanto más trillados y generalizados, tanto mejor. "[4]

Por la similitud que existe entre el sistema a desarrollar y una aplicación de gestión se propone una arquitectura en Modelo Vista Controlador, la cual brinda una gran flexibilidad a la aplicación dado el bajo acoplamiento funcional que se logra y proporcionando que la aplicación se adapte de forma fácil a cualquier cambio existente(adiciones, modificaciones o eliminaciones).

2.2.1 Arquitectura Propuesta.

La arquitectura propuesta persigue que el producto pueda dar respuesta a los requisitos de comprensibilidad, portabilidad, extensibilidad y eficiencia de forma satisfactoria. Para lograr estos

fines se propone la aplicación del patrón arquitectónico Modelo Vista Controlador, y en especial una variante del mismo, conocida como Modelo – Presentación ya que este proporciona un avance metodológico importante que empieza a permitir la unión de lo mejor de dos mundos: la ubicuidad y sencillez de la web y la interactividad del escritorio.

El uso de esta variante permite tener separadas las responsabilidades dentro de la aplicación en dos dominios, uno sería el llamado Presentación, que abarca la lógica correspondiente a la interacción con el usuario y el otro, llamado Modelo, en el cual se maneja la lógica perteneciente al negocio.

2.2.1.1 Presentación.

En el dominio o capa de Presentación es donde se implementa la interfaz de usuario y las clases que las gestionan. Estas se encargan de enlazar los eventos de la interfaz con los manejadores que a su vez invocarán las funcionalidades en el modelo para su encuesta y modificación.

2.2.1.2 Modelo.

Dentro del modelo se identifican un conjunto de clases que se corresponden con los conceptos internos de la aplicación. Estos se asocian principalmente a estados de la configuración del informe o reporte y son independientes de cómo son representados al usuario. También se incluyen otro conjunto de clases las cuales tienen como finalidad la trasformación de los estados de las distintas entidades.

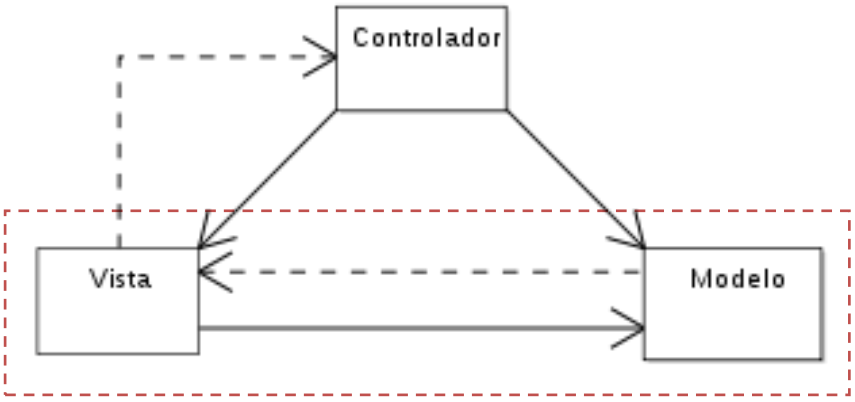


Fig. 6: Esquema funcional del Modelo - Presentación.

2.3 Funcionamiento del diseñador y relación entre los dos dominios.

El objetivo principal del editor es brindar al usuario un ambiente amigable para la creación y diseño de las plantillas de reportes que desee, para lograrlo se ha implementado un conjunto de funcionalidades que debe poseer el sistema.

Según la arquitectura planteada, estas funcionalidades estarán separadas en las clases que pertenecen al modelo y las que forman parte del dominio de Presentación.

A continuación se presenta la explicación de las principales funcionalidades del Editor de Reportes y la interacción entre los dos dominios, para ello se muestran los diagramas de clases del diseño y de interacción de dichas funcionalidades con una breve explicación de las clases más importantes para el desarrollo de los principales escenarios.

2.3.1 Crear Reporte.

Para crear un reporte la aplicación le brinda al usuario la interfaz para realizar su petición ya que este es un sistema independiente que no se encuentra empotrado dentro del Editor de Configuración de Despliegues. Esta interfaz es un botón “Nuevo” ubicado en la barra de herramienta superior, que al hacer clic encima de este muestra una ventana emergente donde pide garantizar el identificador del reporte. Al hacer clic en “Aceptar” las funcionalidades de la biblioteca gráfica se encargarán de gestionar los eventos correspondientes. Estos enlazarán la presentación con los respectivos manejadores y los mismos se encargarán de ejecutar una función de una clase controladora cuyo nombre en este caso es *Configurador,* que luego de haber creado el proyecto llamaría a otra clase controladora para que muestre las propiedades del reporte con los parámetros por defecto del reporte creado, permitiendo al usuario modificar los que desee. Esta clase controladora es la que se encarga de actualizar los datos en el modelo a la vez que se dé la opción de salvar las propiedades del reporte en cuestión. Además con la creación del reporte se muestran todos los componentes por defectos que tiene el editor.

Fig. 7: Diagrama de Secuencia Crear Reporte.

En el siguiente diagrama se muestran las clases encargadas de dar solución a la creación de un reporte.

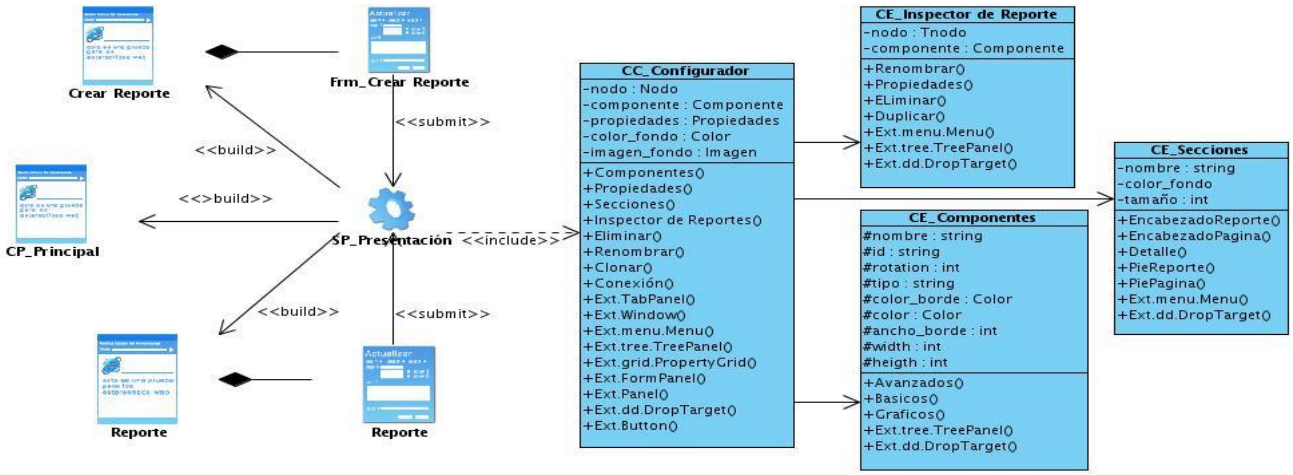


Fig. 8: Diagrama de Clases Crear Reporte.

2.3.1.1 Descripción de las principales clases.

A continuación se presenta la descripción de algunas de las clases involucradas en el escenario anterior, para facilitar la interpretación de las funcionalidades de la aplicación. En ellas se especifican los elementos necesarios para la creación de un reporte, como son los conceptos componentes, secciones e inspector de reportes.

En este proceso de Crear Reporte se toma como la acción del usuario en ejecutar solamente la funcionalidad, donde el dominio Modelo es quien se encarga de manejar el negocio para mostrar las vistas que se generan paralelamente en este escenario relacionándose con la capa de Presentación, como son las siguientes:

La clase *Configurador* es una de las que conforman el dominio Modelo, la cual engloba una serie de funciones específicas y fundamentales para la ejecución de la acción Crear Reporte. Además de ser una clase controladora que se encarga de dar solución a las principales funcionalidades del sistema. Tabla 11: Configurar Reporte.

La clase InspectorReportes representa el reporte como tal, dentro de una de las vistas de interfaz de usuario siendo manejada por el dominio Presentación. La misma lo visualiza en forma de nodo teniendo adjunto a él una serie de funcionalidades como propiedades, eliminar, duplicar y renombrar. Tabla 13: “InspectorReportes”.

Otra de las clases inmersas en este escenario son Componentes y Secciones, las cuales van a ser descritas más adelante con sus respectivas funcionalidades, ya que requieren mayor énfasis en otro de los escenarios que brinda el sistema.

Una vez creado el reporte o informe se está en la etapa más importante de esta investigación, la de facilitar al usuario configurar su documento como lo desee, estableciendo una serie de propiedades y estilos en cuanto a su necesidad de trabajo.

2.3.2 Configurar Reporte.

Para configurar un reporte se debe haber creado un informe siguiendo todos los pasos anteriormente descritos, haciendo cumplir con las funcionalidades de la biblioteca gráfica que se encargará de gestionar los eventos correspondientes. La aplicación le brinda al usuario la interfaz para realizar su petición. Esta interfaz es una ventana desplegable nombrada “Propiedades Reporte” que aparece al hacer clic derecho encima del reporte creado, ubicado en el “Inspector de Reporte” y en la barra de herramientas superior como un botón de igual nombre, saliendo una ventana emergente pidiendo garantizar las propiedades del diseño. Estos enlazarán la presentación con los respectivos manejadores y los mismos se encargarán de ejecutar una función de una clase controladora cuyo nombre en este caso es *Configurador,* que una vez establecido los parámetros de diseño, el usuario puede definir los componentes que desee en las secciones correspondientes, creadas mediante una clase controladora *Base_SVG,* mostrándose paralelamente las propiedades

de cada uno de los componentes y actualizándose el "Inspector de Reporte" con todos los componentes adjuntos a él.

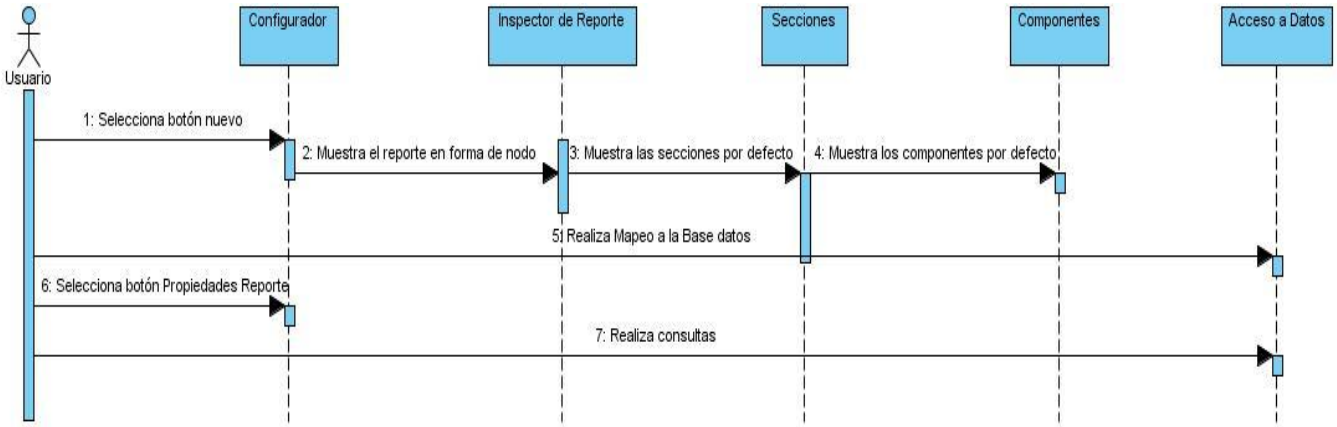


Fig. 9: Diagrama de Secuencia Configurar Reporte.

En el siguiente diagrama se muestran las clases encargadas de dar solución a la configuración de un reporte.

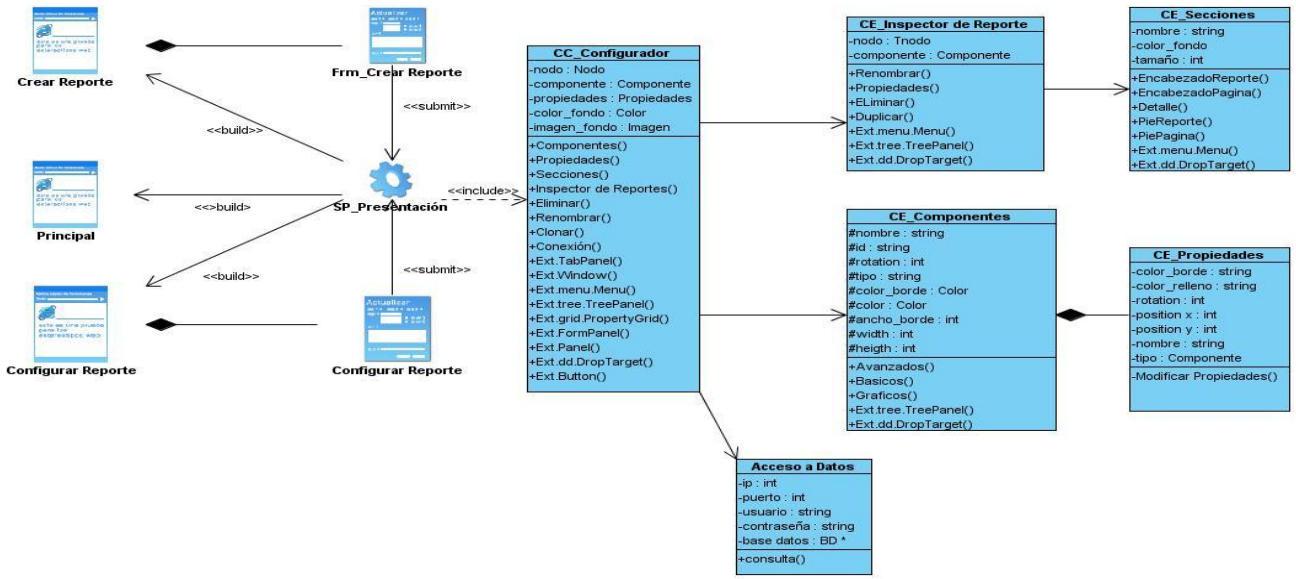


Fig. 10. Diagrama de Clases Configurar Reporte.

A continuación se presentan a modo de descripción las propiedades generales de un reporte en su creación, con la respectiva explicación de forma práctica de cada vista presentada. Dentro de la ventana general de las propiedades del reporte se encuentran varias sub ventanas, de las cuales se presentarán solo la “General”, “Configuración de la Página”, “Consulta” y “Secciones”.

2.3.2.1 Propiedades generales del diseño.

Cuando se crea un diseño es necesario establecer un conjunto de propiedades generales, las cuales definirán características esenciales del mismo. Si estas propiedades no son especificadas por el usuario se establecerán un conjunto de valores por defecto. La clase que maneja esta ventana de propiedades generales del reporte es la anteriormente descrita “Configurador”. Como se muestra en la Fig. 11, dentro de estas propiedades se encuentra la paleta General donde aparece el nombre, el título y la descripción del diseño que se crea.

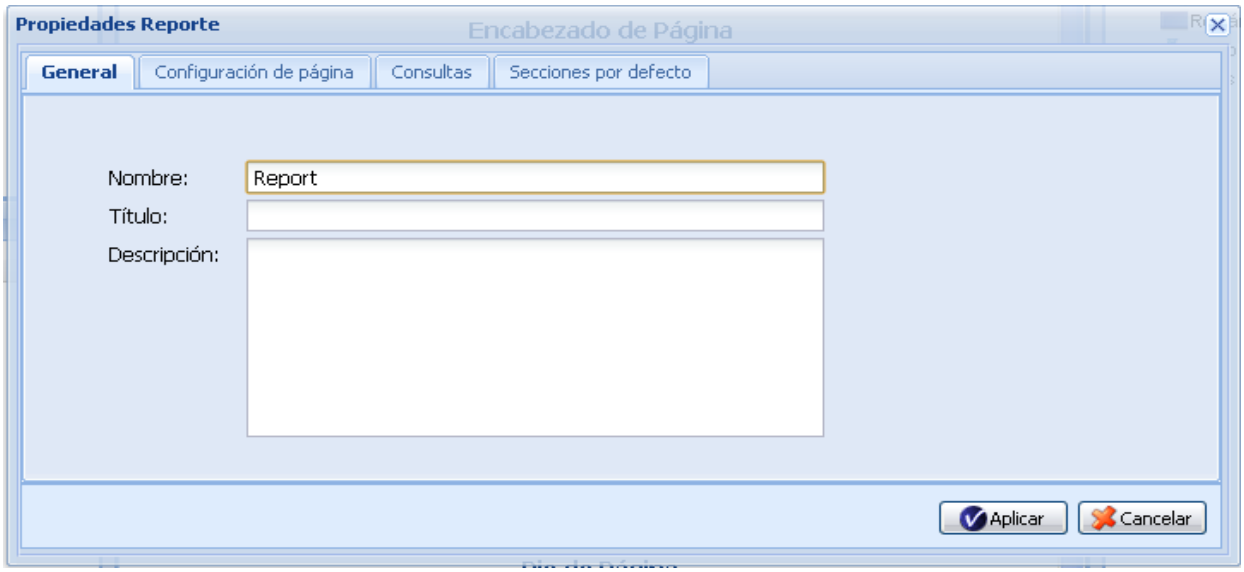


Fig. 11: Propiedades generales de un diseño.

2.3.2.2 Propiedades de las páginas.

Al generarse el reporte, este se desplegará sobre hojas con dimensiones estándar o personalizadas por el diseñador. En cualquiera de los casos será necesario en el momento de diseño especificar las características de estas páginas. Este proceso se realizará a través de la interfaz mostrada en la Fig. 12. Como se puede apreciar, deben ser especificadas las dimensiones de las páginas. Para

facilitar este trabajo, se pueden elegir los formatos de hoja mediante la lista desplegable que aparece en el cuadro “Tamaño de hoja”.

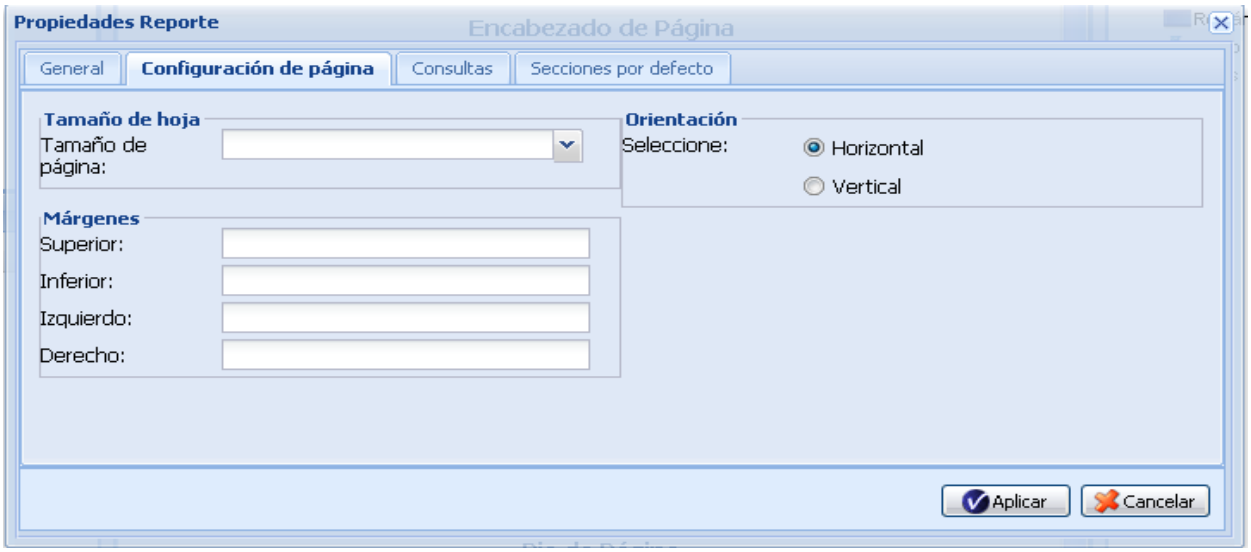


Fig. 12: Propiedades de las páginas del reporte.

Además de especificar las dimensiones de las hojas se podrán especificar los márgenes laterales, así como superior e inferior en las casillas correspondientes. Esto provocará que en el momento de generar el reporte su despliegue se limite al área comprendida entre estos márgenes. Por último se podrá definir la orientación que se le dará a las hojas para desplegar sobre ellas el reporte.

A continuación se presenta la descripción de la clase Propiedades Reporte que es la encargada de dar solución a las necesidades de diseño del cliente. Tabla 14: “Propiedades Reporte”.

2.3.2.3 Consultas.

Cuando se trata el término consultas, se refiere al manejo que se le da a los datos de una o varias base datos, de las cuales se seleccionará la que se quiere que se muestre en el detalle del reporte generado, o lo que es lo mismo, cada detalle tendrá asociada una consulta para volcar los datos en los campos de datos insertados en el reporte. Por lo que se hizo necesario establecer un mapeo a la base datos para la obtención de estos datos, que a la hora de generación se verán volcados en el interior del diseño de informe.

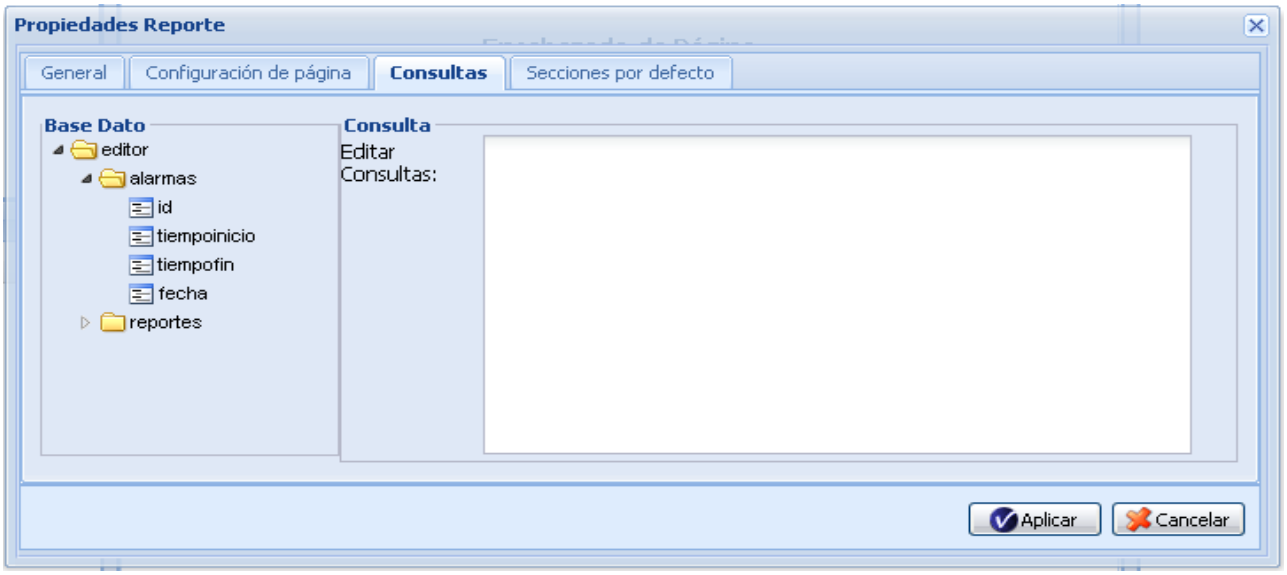


Fig. 13: Configuración de Consultas

2.3.2.3.1 Mapeo a Base Datos.

El mapeo es una técnica de programación para convertir datos entre los sistemas que utilizan un lenguaje de programación orientado a objetos y el utilizado en una base de datos relacional. En la práctica esto crea una base de datos orientada a objetos virtual sobre la base de datos relacional. Posibilitando el uso de las características propias de la orientación a objetos (básicamente herencia y polimorfismo). Hay paquetes comerciales y de uso libre disponibles que desarrollan el mapeo relacional de objetos, aunque algunos programadores prefieren crear sus propias herramientas.

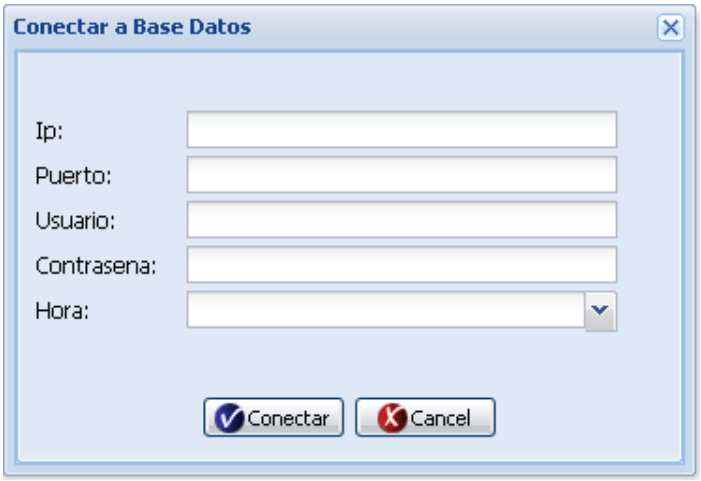


Fig. 14: Conexión a Base Datos

2.3.2.4 Secciones.

Cuando se realiza un diseño una de las tareas más creativas es la especificación del comportamiento que tendrá el despliegue de los datos sobre el área disponible. Para controlar este comportamiento, en el instante de diseño se maneja el concepto de sección. Una sección es un área acotada que define el espacio donde se desplegarán los datos que se obtengan en el momento de generar el reporte. La Fig. 15 muestra como el área de diseño aparece dividida en 5 fragmentos verticales separadas por líneas de color azul, estas son las referidas secciones. Existen varios tipos de secciones cuya peculiaridad fundamental es su comportamiento en el momento de generación.

Fig. 15: Secciones básicas de un diseño.

En este esquema se puede apreciar la división del diseño en cinco secciones, el uso típico de las mismas es el siguiente:

- **Encabezado del reporte:** Es una sección que sólo se muestra en la parte superior de la primera página, normalmente se usa para especificar el título del reporte que se generará.
- **Encabezado de página:** Sección que se muestra en la parte superior de todas las hojas, normalmente empleada para colocar generalidades referidas al reporte, que permitan identificar la pertenencia de una hoja a un determinado reporte, como nombre del informe, fecha de generación, etc.

- **Detalle:** En esta sección normalmente se despliegan los datos que provienen de la ejecución de consultas. En el instante de generación, esta sección se repetirá tantas veces como sea necesario para desplegar todo el flujo de datos que recibe, usando para ello el área libre que queda entre la zona de encabezados y pies, dando lugar a la generación de nuevas páginas.
- **Pie de página:** Sección que se muestra en la parte inferior de todas las hojas, normalmente empleada para colocar números de página.
- **Pie de reporte:** Es una sección que sólo se muestra en la parte inferior de la última página del reporte. Normalmente es usada para poner fin a un reporte. Suele además desplegarse en esta sección información que consolida o resume todos los datos desplegados en el reporte, típicamente totales, medias y demás funciones estadísticas.

La descripción anterior se corresponde con el comportamiento típico de estas secciones y posteriormente dentro del escenario “Insertar Componente” se describe Sección como entidad del modelo. Mediante la interfaz de edición de las secciones, mostradas en la figura 16, se puede especificar otro comportamiento mediante la selección de las opciones que aparecen en la parte izquierda de esta ventana. De esta se puede inferir que en un diseño se podrá o no incluir encabezados, pie de reporte, y pie de páginas. En el caso de los encabezados y pie de páginas, se podrá especificar si queremos que se muestre sólo en la primera y la última página del reporte generado o si se desea que aparezca a lo largo de todo el reporte.

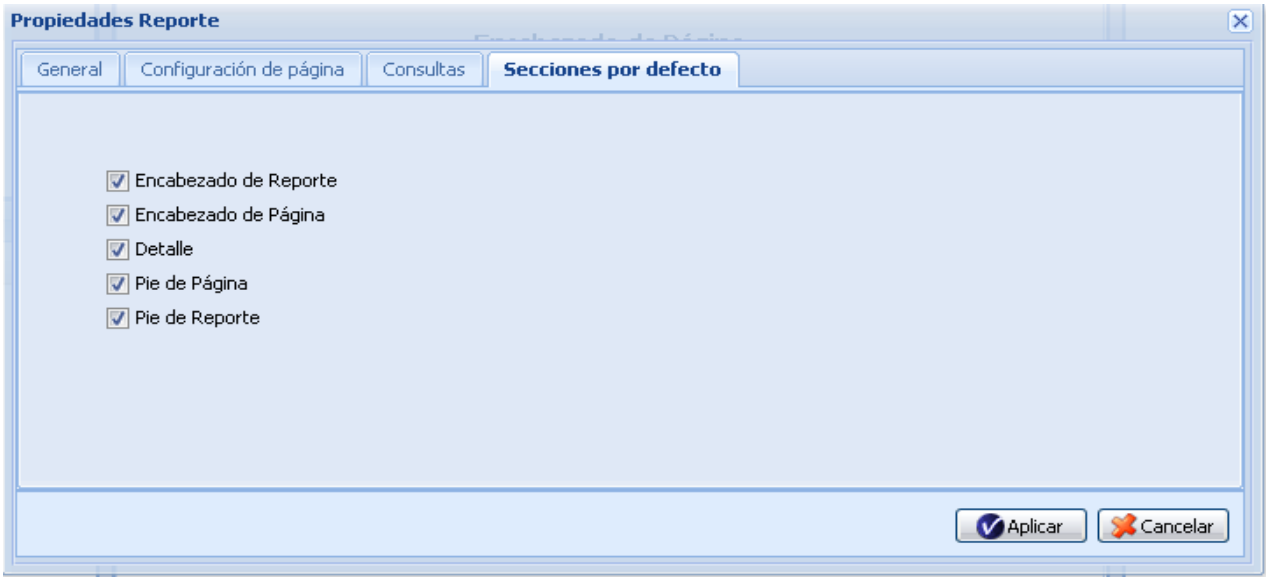


Fig. 16: Edición de Secciones.

Esta interfaz cuenta con una lista de todas las secciones de tipo “detalle” existentes en el diseño, que por su complejidad en esta versión solo concentra la aparición de un solo detalle por cada reporte. A continuación se muestra parte de la descripción de la clase correspondiente a la vista de secciones en la ventana de Selección de Secciones.

La clase Secciones es donde se definen las secciones por defecto que va a tener el diseño de informe, teniendo la posibilidad el usuario de especificar las que desee mediante las Propiedades del diseño. Tabla 15: “Secciones”.

Como se ha podido observar la primera parte de la creación de un diseño de reporte consiste en definirle una serie de características o propiedades que son de vital importancia para la visualización del reporte deseado. Una vez definidas estas características se lleva a cabo otro proceso fundamental, el llamado “Diseñar Reporte”.

Se debe tener en cuenta que cuando se habla de diseñar, no solo es la acción de darle el mejor estilo de trabajo al informe en el momento de diseño; sino también, modificar el estilo de alguno ya realizado con anterioridad, mediante otra de las funcionalidades requeridas de la aplicación, la cual será descrita en su escenario correspondiente.

2.3.3 Diseñar Reporte.

Para diseñar un reporte se debió haber creado y configurado un informe siguiendo todos los pasos anteriormente descritos, haciendo cumplir con las funcionalidades de la biblioteca gráfica que se encargará de gestionar los eventos correspondientes cumpliendo con las características que desee el cliente para su informe. Posteriormente el usuario selecciona de la paleta de componentes el que desea insertar activándose el “Inspector de reporte” quien contiene el tipo de componente y se invoca una instancia de la clase “*Componentes_base_SVG* “para que envíe al administrador del modelo llamado “*Configurador*” la petición de fabricar el objeto deseado. Una vez que es seleccionado el componente se inserta en el área de edición, creándolo a partir de la biblioteca perteneciente al modelo “Raphaeljs” mostrando el componente en la sección seleccionada y sus propiedades en la vista de igual nombre, actualizándose la capa de presentación “Índex”. Este es otro de los ejemplos donde se pone de manifiesto de forma práctica la relación entre los dominios.

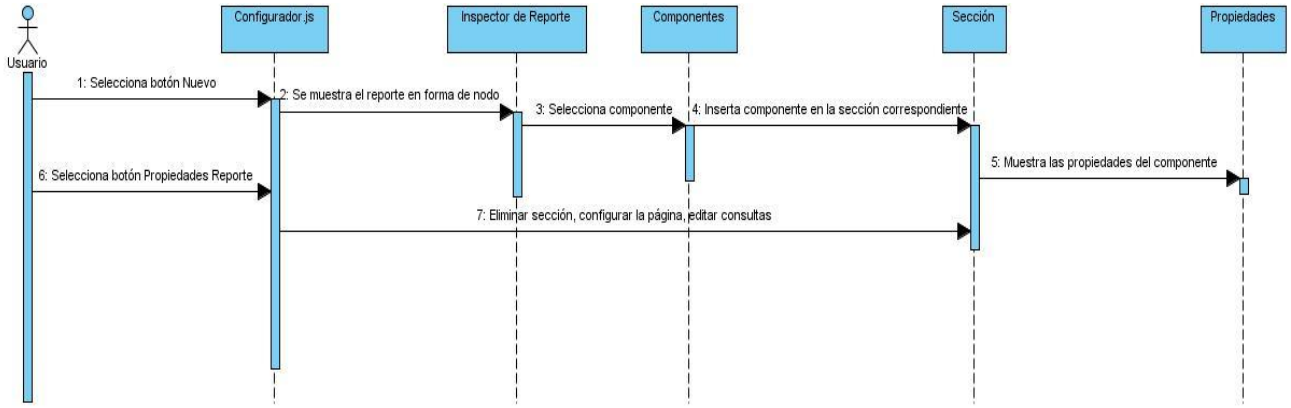


Fig. 17: Diagrama de Secuencia Diseñar Reporte.

En el siguiente diagrama se muestran las principales clases que se encargan de dar solución al diseño de informe:

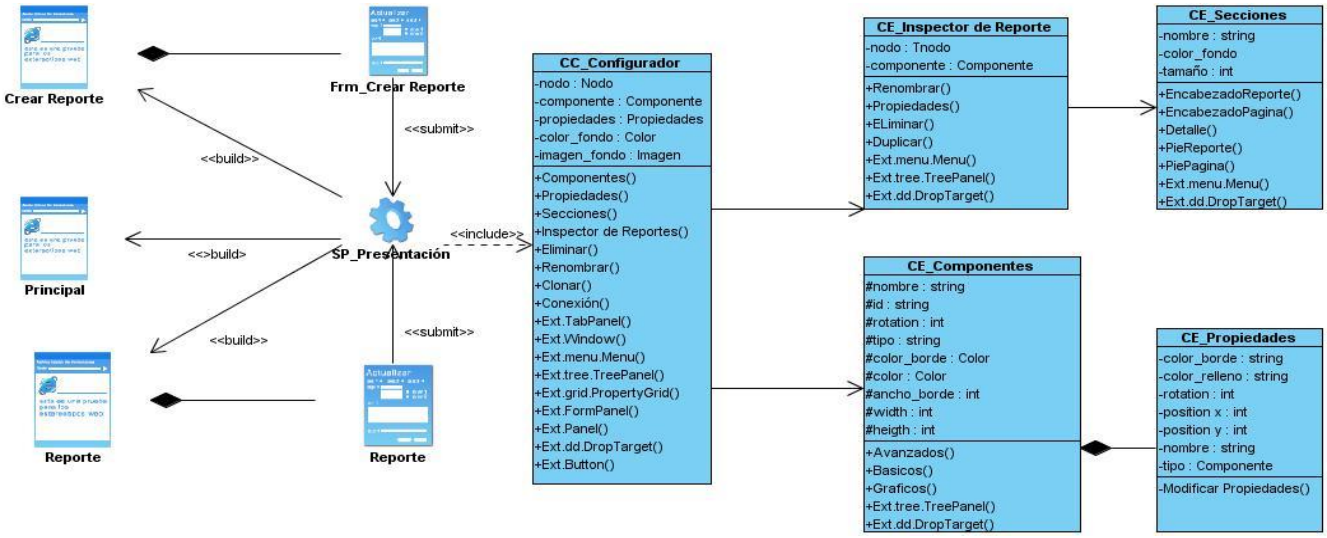


Fig. 18. Diagrama de Clases Diseñar Reporte

Se debe puntualizar en este escenario la utilización de una biblioteca de JavaScript llamada “Raphaeljs” la cual permite el manejo de elementos Gráficos Vectoriales Escalables (SVG), en la aplicación los componentes son realizados con esta variante.

2.3.3.1 Biblioteca Raphaeljs.

Es una pequeña biblioteca que simplifica el trabajo con gráficos vectoriales, utiliza las recomendaciones de SVG como base para la creación de gráficos. Por lo que en cada objeto se crea además un objeto DOM, para que pueda conectar los controles de eventos JavaScript o

modificarlos más adelante. Teniendo como objetivo principal proporcionar un adaptador para el arte de dibujo vectorial compatible con varios navegadores como son hasta el momento Firefox 3.0 +, Safari 3.0 +, Chrome 5.0, Opera 9.5 + e Internet Explorer 6.0 o superior

2.3.3.1.1 Componentes SVG.

Es un formato de gráficos para la web, el cual define un lenguaje basado en XML para la construcción de gráficos vectoriales de dos dimensiones (2d), con multitud de efectos y características avanzadas, como patrones de relleno y gradientes, no pierde calidad si se hace zoom o si se redimensiona y la más importante es que tiene integración con otras tecnologías como es XML; formato estándar en que el Diseñador de este sistema va a exportar el documento o informe. Además permite la utilización de tres tipos de objetos gráficos, un grupo que incluyen algunas figures geométricas como apoyo visual al diseño y el texto, otro los gráficos y por último aquellos que su información será obtenida de la base datos . Todos componentes que se utilizaran en el desarrollo del Editor.

Por la importancia que tienen los objetos gráficos en esta investigación, a continuación se presenta el diagrama de clases correspondiente a la jerarquía de componentes.

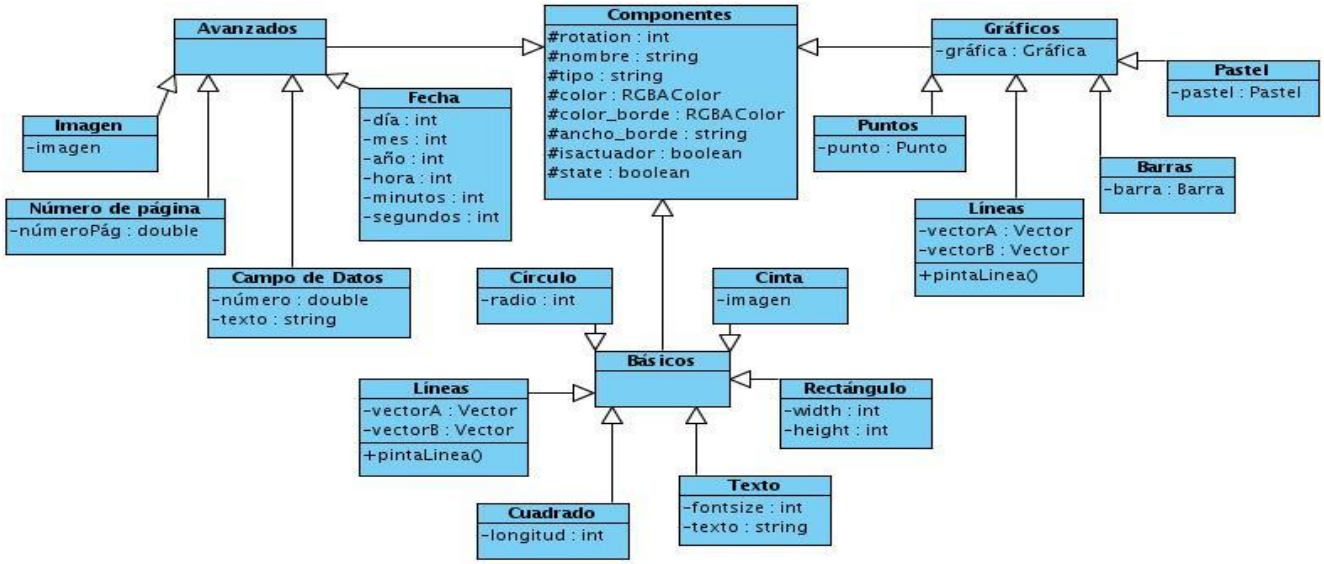


Fig. 19: Jerarquía de Componentes.

En el diagrama de clases anterior, figura 19, se observa que existen tres clases herederas de Componentes_Base_SVG, estos son los grupos de componentes en los que va estar divididos el

Diseñador: Básicos, clase base que la conforman Círculo, Cuadrado, Línea, Texto, Cinta y Rectángulo, los Avanzados de los que forman parte el Campo de datos, Fecha, Número de página e Imagen y los Gráficos que incluyen las gráficas de Pastel, Barra, Punto y Líneas. A continuación se muestra la descripción de alguna de las clases de esta jerarquía por la similitud entre ellas. Tabla 16: "Componentes_Base_SVG".

La clase Círculo es la entidad que se encarga de modelar el componente círculo. Tabla 17: Componente "Círculo".

La clase Cuadrado es la entidad que se encarga de modelar el componente cuadrado. Tabla 18: Componente "Cuadrado".

La clase Línea es la entidad que se encarga de modelar el componente línea. Tabla 19: Componente "Línea".

La clase Rectángulo es la entidad que se encarga de modelar el componente rectángulo. Tabla 20: Componente "Rectángulo".

Entre otras de las clases mostradas en la jerarquía de componente se encuentra la clase Avanzados la cual es base de todos los objetos gráficos que se pintan y visualiza en forma de Texto, dígase Campo de datos y Número de página, por lo que se mostrará solamente la descripción de la clase texto que es uno de los componentes básicos, pero por la importancia del mismo y su similitud con estos dos componentes se describe a continuación. Tabla 21: Componente "Texto".

La clase Fecha es la entidad que modela el componente fecha, la misma basa su comportamiento en extraer del ordenador la fecha actual y mostrarla en una cadena de caracteres en el diseño. Tabla 22: Componente "Fecha".

La clase Imagen es entidad que se encarga de cargar imágenes que identifican la Institución o sea imprescindible para el usuario en el desarrollo de su diseño. Tabla 23: Componente "Imagen".

Dentro de las clases que heredan de Gráficas se encuentran las siguientes:

La clase Gráfica de Barra es la entidad que se encarga de modelar el componente gráfico de barra, donde está definida una data por defecto y se le da la posibilidad al usuario de insertar la que desee. Al igual que las clases Gráfica de Punto, Gráfica de Pastel y Gráfica de Líneas. Tabla 24: Componente "Gráfica de Barra".

Ya en esta etapa se logró la creación, configuración y diseño del informe o reporte en cuanto a las necesidades del usuario, por lo que este se encuentra en el momento de hacer cumplir con otra de las funcionalidades más importantes de este sistema; la de Exportar el documento en una dirección local para su posterior generación.

2.3.4 Exportar Reporte.

Para exportar un reporte se debió haber creado un informe siguiendo todos los pasos anteriormente descritos, haciendo cumplir con las funcionalidades de la biblioteca gráfica que se encargará de gestionar los eventos correspondientes. La aplicación le brinda al usuario la interfaz para realizar su petición, esta es un botón ubicado en la barra de herramienta superior que al hacer clic encima del botón “Exportar” muestra una ventana emergente, donde pide garantizar el destino que recibirá el reporte. Estos enlazarán la presentación con los respectivos manejadores y los mismos se encargarán de ejecutar una clase controladora cuyo nombre es ExportarXml.

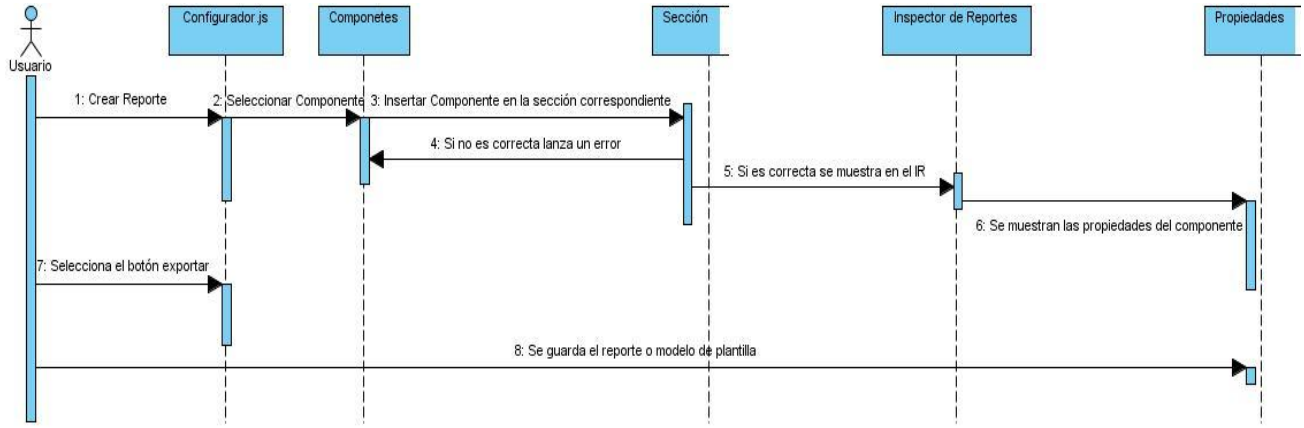


Fig. 20: Diagrama de Secuencia Exportar Reporte.

En el siguiente diagrama se muestran las clases encargadas de dar solución a Exportar un reporte.

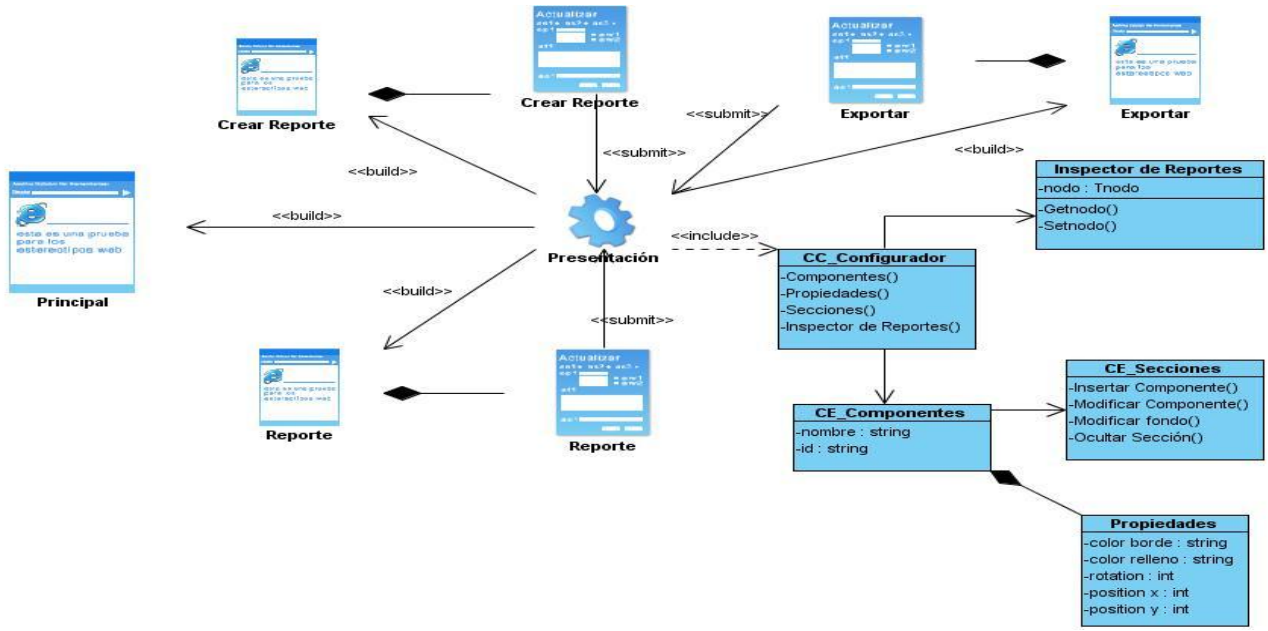


Fig. 21: Diagrama de Clases Exportar Reporte.

2.3.4.1 Descripción de las principales clases.

A continuación se presenta la descripción de la clase ExportarXml del escenario anterior para facilitar la interpretación de esta funcionalidad de la aplicación. No es necesario describir las otras clases involucradas, ya que fueron explicadas anteriormente en el escenario Crear reporte.

Tabla 3: "ExportarXml".

Nombre	ExportarXml
Tipo	Controladora
Atributo	$file_name; report_name; fondo; componente
Operaciones	
Nombre	Ext.Button
Descripción	Crea botón donde se realiza la funcionalidad Exportar Reporte.

2.3.5 Importar Reporte.

Esta funcionalidad consiste en cargar un reporte anteriormente realizado, facilitándole al usuario la posibilidad de realizar algún cambio pertinente. La aplicación le brinda al usuario la interfaz para realizar su petición, esta es un botón ubicado en la barra de herramienta superior que al hacer clic encima del botón “Importar” sale una ventana emergente, donde pide garantizar el destino donde se encuentra ubicado el reporte. Estos enlazarán la presentación con los respectivos manejadores y los mismos se encargarán de ejecutar una clase controladora cuyo nombre es ImportarXml, haciendo cumplir con las funcionalidades de la biblioteca gráfica que se encargará de gestionar los eventos correspondientes.

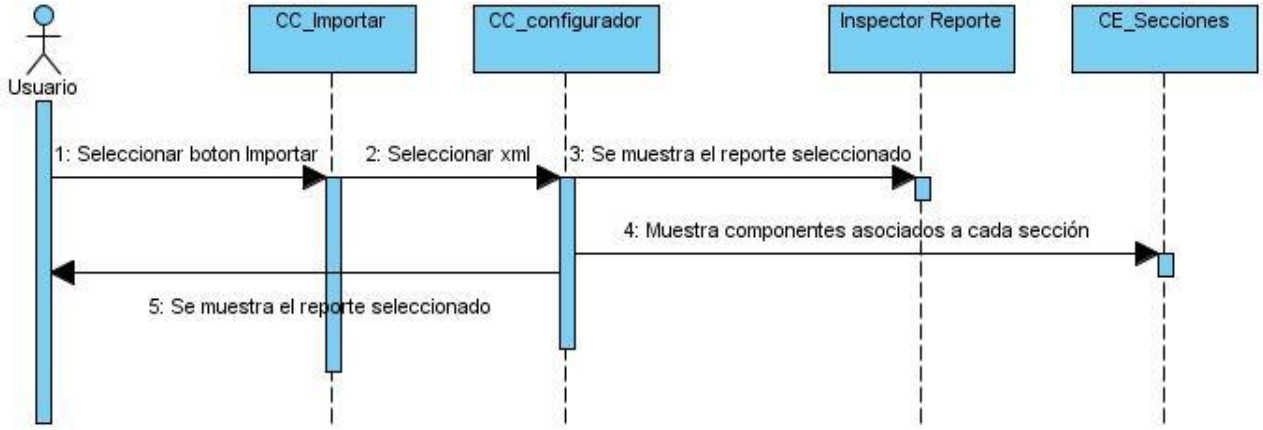


Fig. 22: Diagrama de Secuencia Importar Reporte.

En el siguiente diagrama se muestran las clases encargadas de dar solución a Importar un reporte.

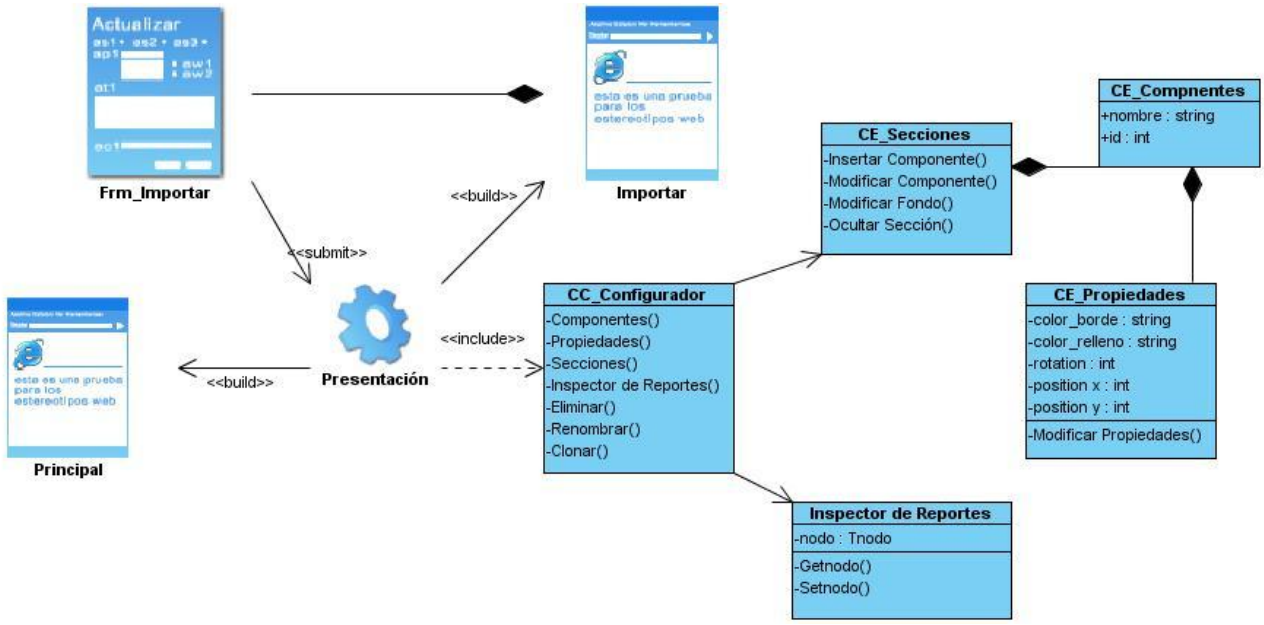


Fig. 23: Diagrama de Clases Importar Reporte.

2.3.5.1 Descripción de las principales clases.

A continuación se presenta la descripción de la clase ImportarXml, no es necesaria la descripción del resto de las clases porque ya fueron explicadas en escenarios anteriores.

Tabla 4: "ImportarXml".

Nombre	ImportarXml
Tipo	Controladora
Atributo	$dir_file; $xmlcargado;
Operaciones	
Nombre	Ext.FormPanel()
Descripción	Para importar un Reporte.
Nombre	Ext.Button
Descripción	Crea botón donde se realiza la funcionalidad de Importar Reporte.

2.4 Funcionalidades y propiedades de los componentes.

2.4.1 Componentes visuales para un diseño.

Como se refirió anteriormente para la elaboración de diseños se dispone de una paleta de componentes gráficos. Mediante una correcta disposición de estos componentes sobre las secciones del diseño, se puede definir la apariencia visual de los reportes. Cada componente gráfico tiene su propio conjunto de propiedades, mediante las cuales se define su comportamiento al momento de generar un reporte. A dichas propiedades se puede acceder mediante la ventana que aparece del lado lateral izquierdo inferior de la interfaz cuando se halla insertado algún componente en el área de diseño.

No es necesaria la realización del diagrama de clases y de interacción de dicha acción, ya que la funcionalidad Diseñar Reporte anteriormente descrita incluye esta parte de mostrar las propiedades de los componentes, ya que estas son visualizadas en el mismo momento que el usuario inserta el componente en la sección deseada.

A continuación se describen las propiedades de los componentes visuales que se podrán emplear en la elaboración de un diseño y se muestran los inspectores de propiedades de cada uno de ellos.

2.4.1.1 Rectángulo, Círculo y Cuadrado.

Estos componentes son tres de las figuras básicas con las que cuenta el diseñador, los cuales se utilizan para darle un mayor sentido y organización al reporte final, para mejorar la apariencia del mismo, además pueden usarse como marco para resaltar alguna información o para conformar tablas en el caso del rectángulo y el cuadrado. A continuación se presentan las siguientes propiedades que los definen:

- Nombre: Nombre que identificará el componente en el diseño, es un nombre para diferenciar las distintas instancias de este componente.

- Posición X: Coordenada en el eje horizontal correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.

- Posición Y: Coordenada en el eje vertical correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.

- Largo: Extensión horizontal del componente.
- Ancho: Extensión vertical del componente.
- Color del borde: Color que tomará la línea que bordea el componente.
- Color de fondo: Color que tomará el fondo del componente.
- Ancho de borde: Grosor que tomará la línea que bordeará el componente.
- Rotación: Movimiento que se le dará al componente en cualquier sentido.

Propiedades

Nombre	Valor
nombre	Circulo0
x	39
y	39
radio	45
color	
color_borde	#000
ancho_borde	3
rotation	0

Propiedades

Nombre	Valor
nombre	Rectangulo0
x	3
y	3
width	184
height	84
color	
color_borde	#000
ancho_borde	0
rotation	0

Propiedades

Nombre	Valor
nombre	Cuadrado0
x	3
y	3
longitud	84
color	
color_borde	#000
ancho_borde	3
rotation	0

Fig. 24: Interfaces de las propiedades de los componentes Círculo, Rectángulo y Cuadrado.

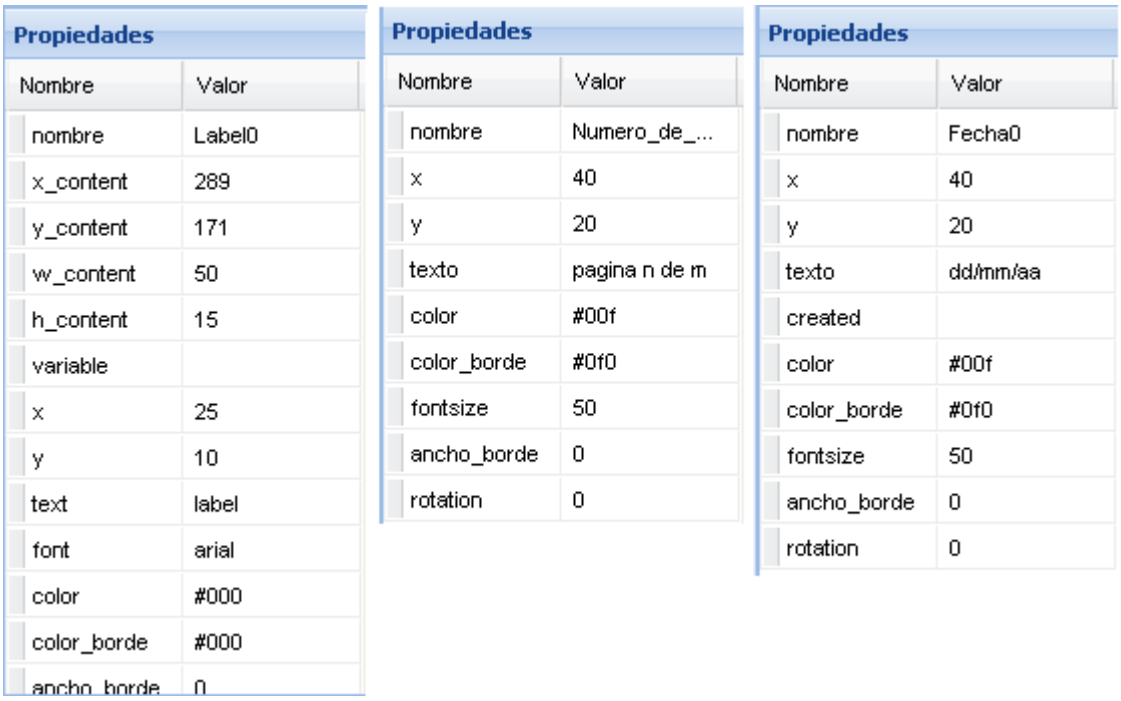
2.4.1.2 Campo de datos, Fecha, Texto y Número de página.

Estos cuatro componentes a pesar de no formar parte de la misma herencia presentan características similares y son muy importantes en el diseño de informes.

El Texto es fundamental pues es donde se plasman todos los datos escribibles del diseño. Por su parte el Campo de Datos facilita la obtención de alguna información en específica de la Base de Datos referente a la búsqueda invocada, el componente Fecha facilita la obtención del día, mes y año cargados del ordenado, teniendo además la posibilidad de brindar al usuario poner la fecha que desee de forma manual; por otra parte se encuentra el Número de página al cual se le define en el

diseño el número con que quiere el usuario que comience el enumerado de las páginas del reporte y este se va incrementando en tiempo de generación una unidad por cada página que tenga el reporte final. Es válido aclarar que estos tres últimos componentes en el diseño no cargan los datos, solo en ejecución, a la hora de generar el reporte es donde se actualizan los datos diseñados anteriormente. Las propiedades que deberán ser establecidas para estos componentes son las siguientes:

- Nombre: Nombre que identificará el componente en el diseño, es un nombre para diferenciar las distintas instancias de este componente.
- Posición X: Coordenada en el eje horizontal correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.
- Posición Y: Coordenada en el eje vertical correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.
- Texto: Donde se podrá especificar lo que quieras escribir en el diseño.
- Color de Letra: Paleta donde especificas el color que le quieras establecer al texto introducido en el diseño.
- Color de Borde: Color que tomará la línea que bordea el texto.
- Tipo de letra: Donde se establece una familia de tipo de letras para especificar la que desees.
- Tamaño de letra: Tamaño que tomará la propiedad Texto.
- Rotación: Movimiento que se le dará al componente en cualquier sentido.



Propiedades

Nombre	Valor
nombre	Label0
x_content	289
y_content	171
w_content	50
h_content	15
variable	
x	25
y	10
text	label
font	arial
color	#000
color_borde	#000
ancho_borde	0

Propiedades

Nombre	Valor
nombre	Numero_de_...
x	40
y	20
texto	pagina n de m
color	#00f
color_borde	#0f0
fontsize	50
ancho_borde	0
rotation	0

Propiedades

Nombre	Valor
nombre	Fecha0
x	40
y	20
texto	dd/mm/aa
created	
color	#00f
color_borde	#0f0
fontsize	50
ancho_borde	0
rotation	0

Fig. 25: Interfaces de las propiedades de los componentes Texto, Número de Página y Fecha.

2.4.1.3 Imagen.

En los reportes suelen usarse imágenes que identifican la institución o que sencillamente contribuyen estéticamente a la calidad final del reporte. Para ello el diseñador proporciona un componente mediante el cual se puede insertar en el diseño imágenes almacenadas en distintos formatos estándares, como PNG, JPG y SVG. Este componente, a diferencia de los restantes, se carga desde un directorio local y se dibuja con las mismas características del archivo que está siendo cargado. A continuación se presentan las siguientes propiedades que lo define:

- Nombre: Nombre que identificará el componente en el diseño, es un nombre para diferenciar las distintas instancias de este componente.

- Posición X: Coordenada en el eje horizontal correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.

- Posición Y: Coordenada en el eje vertical correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.

- URL: Dirección donde se encuentra la imagen que desee insertar.

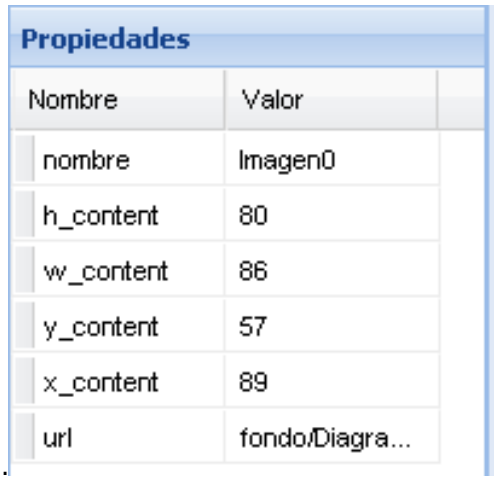


Fig. 26: Interfaz para establecer las propiedades de la Imagen.

2.4.1.4 Línea.

El uso de este componente permite insertar líneas en un diseño. Comúnmente en los reportes las líneas son usadas como separadores. A continuación se presentan sus propiedades:

- Nombre: Nombre que identificará el componente en el diseño, es un nombre para diferenciar las distintas instancias de este componente.
- Posición X: Coordenada en el eje horizontal correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.
- Posición Y: Coordenada en el eje vertical correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.
- Largo: Extensión horizontal del componente.
- Ancho: Extensión vertical del componente.
- Color: Color que tomará el componente.
- Ancho de borde: Grosor que tomará la línea que bordeará el componente.
- Rotación: Movimiento que se le dará al componente en cualquier sentido.

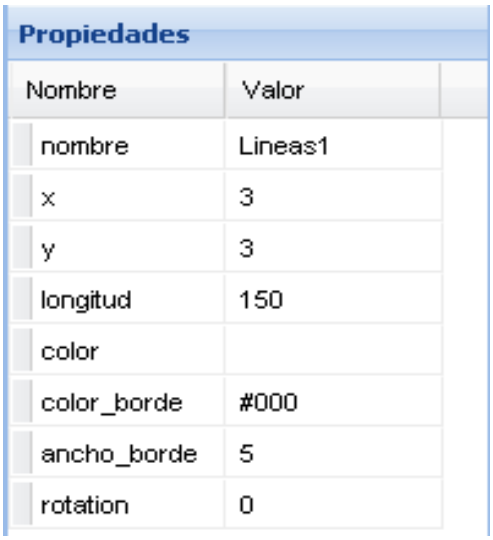


Fig. 27: Interfaz para establecer las propiedades de la Línea.

2.4.1.5 Graficas de Barra, Pastel, Líneas y Puntos.

El uso de estos componentes le permite al usuario organizar su información con un estilo más representativo de los datos obtenidos de las diferentes bases de datos. Donde se debe puntualizar que en este caso solo se muestran las diferentes opciones de cómo puede ser visualizada la información. A continuación se presentan sus propiedades:

- Nombre: Nombre que identificará el componente en el diseño, es un nombre para diferenciar las distintas instancias de este componente.

- Posición X: Coordenada en el eje horizontal correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.

- Posición Y: Coordenada en el eje vertical correspondiente a la posición real que tiene el componente dentro de la sección en la que está dibujado.

- Value: Data que el usuario puede poner manual sin necesidad de obtener los datos de una base datos.

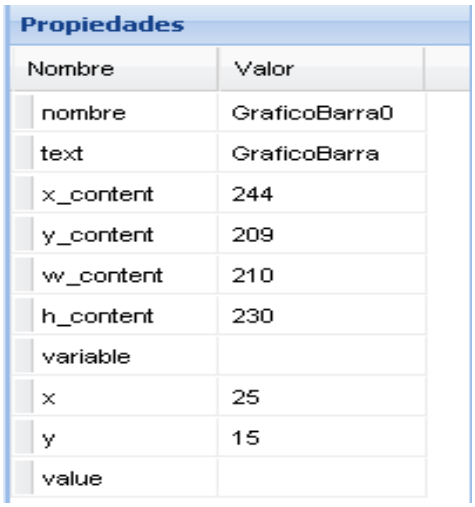


Fig. 28: Interfaz para establecer las propiedades de la Gráfica de Barras

En todo sistema de edición o configuración es necesaria la existencia de funcionalidades básicas que faciliten el diseño. Entre las más conocidas está duplicar, eliminar, renombrar, traer al frente y enviar al fondo. Todas las mencionadas han sido implementadas en esta versión. Se presentan las distintas clases relacionadas con estas acciones, así como su respectiva descripción y explicación para lograr un mayor entendimiento de las mismas.

2.4.2 Funcionalidades en el diseño.

Es válido aclarar que estas funcionalidades no son clases del sistema, sino funciones de diseño que forman parte de la clase controladora *Configurador* del dominio Modelo, donde las descripciones de las tablas van a ser de sus respectivas funciones.

2.4.2.1 Duplicar.

Esta opción es muy cómoda, pues logra hacer una copia fiel del componente previamente seleccionado. En este caso los elementos duplicados se dibujan con una breve variación en las coordenadas de los originales.

Tabla 5: Función "Duplicar".

Nombre	Duplicar
Tipo	action
Atributo	**Tipo**

Operaciones	
Nombre	getNodeById ()
Descripción	Almacena en un arreglo el identificador del componente seleccionado.
Nombre	getSelectionModel ()
Descripción	Mediante la operación getNodeById () esta se encarga de obtener las propiedades del componente en diferencia de las coordenadas de posición.

2.4.2.2 Eliminar.

Esta funcionalidad se encarga de chequear el elemento seleccionado y sacarlo de la lista de objetos gráficos; y luego enviar señal de repintar escena ahora sin este elemento.

Tabla 6: Función "Eliminar".

Nombre	Eliminar
Tipo	action
Atributo	**Tipo**
Operaciones	
Nombre	getNodeById ()
Descripción	Almacena en un arreglo el identificador del componente seleccionado.
Nombre	removeChild ()
Descripción	Elimina el componte seleccionado anteriormente.
Nombre	setActiveItem ()
Descripción	Se repinta la escena ya sin los objetos eliminados.

2.4.2.3 Renombrar.

Esta operación le brinda la posibilidad a el usuario de cambiarle el nombre antes declarado al componente, sin necesidad de eliminarlo para poder establecer el nombre que realmente desee.

Tabla 7: Función "Renombrar".

Nombre	Renombrar
Tipo	action
Atributo	**Tipo**
Operaciones	
Nombre	getNodeById ()
Descripción	Almacena en un arreglo el identificador del componente seleccionado.
Nombre	getSelectedNode ()
Descripción	Captura el elemento seleccionado anteriormente.
Nombre	setText ()
Descripción	Cambia el nombre del elemento.

2.4.2.4 Traer al frente y enviar al fondo.

De la forma en que se dibujan los componentes en este trabajo, resulta sencillo darle solución a esta importante funcionalidad, y es precisamente debido a que a medida que se inserta un nuevo componente en el área de diseño, este se ubica detrás de la posición que ocupa el objeto anterior en la lista de componentes de esa sección.

2.5 Conclusiones.

Es de vital importancia para todo sistema SCADA la presencia de un diseñador de informes, haciendo más cómodo y entendible el proceso de toma de decisiones y la organización de la información. Además de ser sistemas independientes muy sencillos de usar, de fácil instalación y puesta en funcionamiento. En este capítulo se ha realizado la descripción de la mayoría de los

elementos implementados, se han seleccionado las principales funcionalidades y se han descrito detalladamente las clases más significativas que intervienen en cada uno de los escenarios propuestos.

CAPÍTULO 3

3. VALIDACIÓN DE LA SOLUCIÓN.

INTRODUCCIÓN

Todo sistema de software debe ser probado antes de su entrega o puesta en funcionamiento, a este proceso o fase se le denomina prueba o testeo de la aplicación. Es una de las etapas más importante en la vida del software y en ocasiones es a la que menos recursos se le asigna. Por lo que es una mala práctica que conlleva en muchos casos al fracaso del producto o la no satisfacción del cliente. No siempre las pruebas garantizan que el software esté totalmente en cuanto aspectos funcionales, pero si destacan numerosas fallas que el desarrollador no encontró y que ahora pueden ser corregidas y vueltas a probar. En el siguiente capítulo se presentan las pruebas para validar el correcto funcionamiento del sistema.

3.1 Prueba del Sistema.

El proceso de pruebas se enfoca sobre la lógica interna del software y las funciones externas. Es un proceso de ejecución de un programa con la intención de descubrir uno o varios errores. Un buen Caso de Pruebas es aquel que tiene alta probabilidad de mostrar un error no descubierto hasta entonces.

La prueba es el proceso de ejercitar un programa bajo condiciones específicas cuyos resultados deben ser registrados y luego analizados. Es posible hacerlas a distintos niveles, tales como unidad, integración y aceptación.

Por lo que cualquier producto de ingeniería puede ser probado mediante una de estas técnicas:

- Conociendo la funcionalidad específica para la cual fue diseñado el producto, se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa. **Pruebas de Caja Negra**.

- Conociendo el funcionamiento del producto se pueden desarrollar pruebas que aseguren que "todas las piezas encajen ", o sea, que la operación interna se ajusta a las especificaciones y que todo los componentes internos se han comprobado de forma adecuada. **Pruebas de Caja Blanca**.

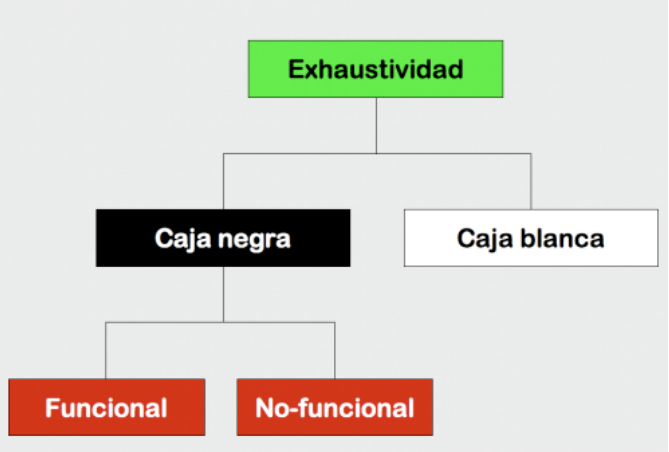


Fig. 29: Técnicas de Pruebas.

3.2 Plan de prueba.

Para la correcta realización de las pruebas, se contó con una serie de recursos que facilitaron el trabajo de ejecutar cada Caso de Prueba sobre la herramienta:

- **Recursos Físicos:** Las computadoras utilizadas para el desarrollo de las pruebas contaron con 1.0 GB de memoria RAM, microprocesador Intel Core2Duo E4500 con velocidad de 2.20 GHz, motherboard Intel y una capacidad en disco duro de 160 GB, red alambica.

- **Recursos Lógicos:** Sistema Operativo en el cual se desarrollaron las pruebas fue Windows XP SP2.

3.3 Diseño de los Casos de Pruebas.

Al sistema desarrollado se le realizaron Pruebas de Caja Negra, la cual se centra fundamentalmente en las funcionalidades descritas en el capítulo anterior.

Estas pruebas permiten obtener un conjunto de condiciones de entrada que ejercitan completamente todos los requisitos funcionales de un programa. En ellas se ignora la estructura de control, concentrándose en los requisitos funcionales de la herramienta.

Los casos de prueba escritos incluyen una descripción de la funcionalidad que se probará, la cuál es tomada ya sea de los requisitos funcionales o de los casos de uso, y la preparación requerida para asegurarse que la prueba pueda ser dirigida.

Tabla 8: Casos de Prueba Crear Reporte

Condición de entrada	**Casos válidos**	**Casos no válidos**
El usuario debe introducir el nombre del reporte.	Que el usuario introduzca el nombre del reporte.	Que el usuario no introduzca el parámetro definido.
Funcionalidad:	Crear Reporte.	
Caso de Prueba:	Crear Reporte.	
Entradas:		
Nombre: Reporte.		
Resultado:	El sistema crea el reporte y actualiza el Inspector de Reportes (Ver Pantalla 1: Crear Reporte.).	
Condiciones:	Los datos deben estar correctamente validados.	
Funcionalidad:	Crear Reporte.	
Caso de Prueba	Crear Reporte.	
Entradas		
Nombre:		
Resultado	El sistema muestra un mensaje de error (Ver Pantalla 2: Crear Reporte.).	
Condiciones	Todos los campos deben estar llenos y validados correctamente.	

Tabla 9: Caso de Prueba Exportar Reporte.

Condición de entrada	Casos válidos	Casos no válidos
El usuario debe tener insertado en el área de diseño al menos un componente.	Que el usuario haya insertado al menos un componente.	Que el usuario no haya insertado ningún componente.
Funcionalidad:	Exportar Reporte.	
Caso de Prueba:	Exportar Reporte.	
Entradas:		
Componentes.		
Resultado:	El sistema guarda el Reporte en el destino previsto (Ver Pantalla 3: Exportar Reporte.).	
Condiciones:	El usuario debe especificar la dirección destino del reporte.	
Funcionalidad:	Exportar Reporte.	
Caso de Prueba	Exportar Reporte.	
Entradas		
Componte.		
Resultado	El sistema no habilita la opción de Exportar reporte (Ver Pantalla 4: Exportar Reporte.).	
Condiciones	Debe insertar al menos un componente en el área de diseño.	

Tabla 10: Importar Reporte

Condición de entrada	Casos válidos	Casos no válidos
El usuario debe especificar la dirección de donde se cargará el reporte.	Que el usuario especifique la dirección de donde se cargará el	Que el usuario no especifique la dirección.

	reporte.	
Funcionalidad:	Importar Reporte	
Caso de Prueba:	Importar Reporte	
Entradas:		
Dirección: C:\Documents and Settings\Administrador\Mis documentos\Descargas		
Resultado:	El sistema carga el Reporte del destino previsto (Ver Pantalla 5: Importar Reporte.).	
Condiciones:	El usuario debe especificar la dirección de donde se cargará el reporte.	
Funcionalidad:	Importar Reporte	
Caso de Prueba	Importar Reporte	
Entradas		
Dirección: “ ”		
Resultado	El sistema muestra un mensaje de error que debe especificar la dirección origen (Ver Pantalla 6: Importar Reporte.).	
Condiciones	Debe especificarse la dirección origen del reporte.	

Tabla 11: Configurar Reporte.

Condición de entrada	Casos válidos	Casos no válidos
El usuario debe introducir el IP y puerto.	Que el usuario introduzca IP y puerto.	Que el usuario no introduzca alguno de los parámetros.
El usuario debe introducir el usuario, contraseña y base dato a conectarse.	Que el usuario introduzca el user, contraseña y base dato a conectarse.	Que el usuario no introduzca alguno de los datos.
Funcionalidad:	Configurar Reporte.	

Caso de Prueba:	Configurar Reporte.
Entradas:	
IP: 10.35. 19.19 Puerto: 5800 Usuario: root Contraseña: root Base Dato: editor	
Resultado:	El sistema se conecta a la base datos para la realización del mapeo. (Ver Pantalla 7: Configurar Reporte.).
Condiciones:	Los datos deben estar correctamente validados.
Funcionalidad:	Configurar Reporte.
Caso de Prueba	Configurar Reporte.
Entradas	
IP: **Puerto:** Usuario: root Contraseña: root Base Dato: editor	
Resultado	El sistema muestra un mensaje de error (Ver Pantalla 8: Configurar Reporte.).
Condiciones	Todos los campos deben estar llenos y validados correctamente.
Funcionalidad	Configurar Reporte.
Caso de Prueba	Configurar Reporte.
Entradas	

IP: 10.35. 19.19 Puerto: 5800 **Usuario:** **Contraseña:** **Hora:**	
Resultado	El sistema muestra un mensaje de error (Ver Pantalla 9: Configurar Reporte.).
Condiciones	Todos los campos deben estar llenos y validados correctamente.

3.4 Resumen del Proceso de Pruebas.

Para el correcto desarrollo del proceso de pruebas fueron definidas un total de 4 Casos de Pruebas, realizándose 4 iteraciones, no haciéndose necesario la puesta en prueba del escenario Diseñar Reporte, ya que no presenta posibles casos donde el usuario pueda tener tendencia a cometer errores. Utilizándose el método de Caja Negra para probar las principales funcionalidades con las que debe cumplir la aplicación, detectándose un total de 8 No Conformidades. Estos errores tenían una repercusión negativa sobre el funcionamiento del sistema y el Mapeo a Base Datos. Con la erradicación de los mismos se pudo mejorar y optimizar gradualmente el funcionamiento de la aplicación.

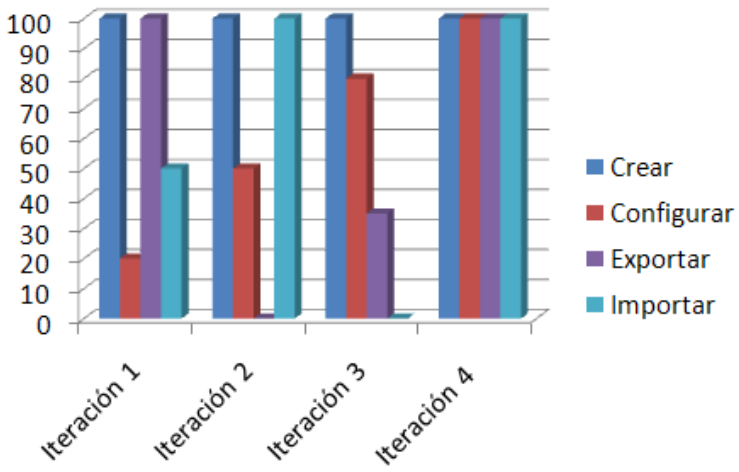


Fig. 30: Resultado de pruebas.

3.5 Conclusiones.

En este capítulo se elaboraron y ejecutaron pruebas al sistema, arrojando resultados significativos para el perfeccionamiento del Editor de Reportes Web, resultados que se optimizaron tras el proceso de pruebas efectuado.

CONCLUSIONES GENERALES.

A partir de la investigación realizada para el proceso de desarrollo de la aplicación se arribaron a las siguientes conclusiones:

- Como respuesta al problema originado, se obtiene un Editor de Reportes Web dando cumplimiento al objetivo de la investigación, el que servirá para el diseño y configuración de las plantillas de informe dependiendo de las necesidades del usuario.

- El paradigma de software libre facilita en gran medida la reutilización de código y un desarrollo acelerado de aplicaciones.

- El uso del patrón arquitectónico MVC y específicamente de la variante utilizada Modelo – Presentación, resulta muy conveniente para el desarrollo de un diseñador de informes para un SCADA u otra aplicación que la necesite, brindando la facilidad de resistir al cambio de los requisitos.

  Se diseño y ejecutó un conjunto de pruebas de interfaz que posibilitaron la validación de la solución.

RECOMENDACIONES.

- Incorporar nuevas funcionalidades al editor de acuerdo a las necesidades que se vayan presentando.

- Desarrollar un motor de generación de reportes en la web para comprobar la multiplicidad de escenarios de los informes diseñados.

- Evaluar la posibilidad del uso del editor, en otros sistemas en el plano nacional.

REFERENCIAS BIBLIOGRAFICAS

[1]. **Zuluaga, Carlos.** Scrpit de Pruebas. *Scrpit de Pruebas.* [Online] WikiDot.com, 9 12, 2010. [Cited: 4 23, 2011.] http://carloszuluaga.wikidot.com/pruebascarga:creacion-script-prueba.

[2]. **Valdés, Damian Pérez.** Maestros del Web. *Maestros del Web.* [Online] Worpress, 2009. [Cited: 1 26, 2011.] http://www.maestrosdelweb.com/editorial/editores-web-que-facilitan-tu-trabajo/.

[3]. **Torrijos, Ricard Lou.** Introducción a la Tecnología JavaServer Faces. *Introducción a la Tecnología JavaServer Faces.* [Online] 2010. [Cited: 11 24, 2010.] http://www.programacion.com/articulo/introduccion_a_la_tecnologia_javaserver_faces_233/9 .

[4]. **Rodriguez, Gilberto.** Patrones de Arquitectura. *Patrones de Arquitectura.* [Online] 2010. [Cited: 12 8, 2010.] http://www.neleste.com/modelo-vista-controlador/.

[5]. **Pos, Alfredo.** Framework Darly. *Framework Darly.* [Online] 2009. [Cited: 10 23, 2010.] http://blog.creonfx.com/javascript/mootools-vs-jquery-vs-prototype-vs-yui-vs-dojo-comparison-revised.

[6]. **Ochoa, Sergio.** *Introducción a los Patrones (Diseño y Arquitectura).* 2010.

[7]. **Mendive, Rafael.** *Componentes Graficos Vctoriales.* [Documento] España : s.n., 2009.

[8]. **Mendez, Natxo.** Desarrollo Web. *Desarrollo Web.* [Online] 2009. [Cited: 11 25, 2010.] http://www.desarrolloweb.com/articulos/832.php.

[9]. **Jacobson, James.** *El proceso Unificado de Desarrollo de Software.* Habana : s.n., 2008.

[10]. **Gutierrez, Javier J.** *Framework Web.* España : s.n., 2009.

[11]. **Dürsteler, Juan C.** La revista digital de InfoVis.net. *La revista digital de InfoVis.net.* [Online] 2000 - 2011. [Cited: marzo 23, 2011.] http://www.infovis.net/printMag.php?num=177&lang=1.

[12]. **Cardenas, Roberto.** *PDVZA Guardian del ALBA Selección de tecnologías .* habana : s.n., 2010.

[13]. **Alvarez, Miguel Angel.** Desarrollo Web. *Desarrollo Web.* [Online] 2009. [Cited: 11 23, 2010.] http://www.desarrolloweb.com/articulos/393.php.

[14]. **sourceforge.net.** Visión Futura. *Visión Futura.* [Online] 2009. [Cited: 10 23, 2010.] http://datavision.sourceforge.net/.

[15]. **Free Dawnload manager.** Sitio de Descargas de Software. *Sitio de Descargas de Software.* [Online] 2004 - 2010. [Cited: noviembre 19, 2010.] http://www.freedownloadmanager.org/es/downloads/dise%C3%B1ador_reportes_webgratis

[16]. Sitio de Descargas de Software. *Sitio de Descargas de Software.* [Online] 2004 - 2010. [Cited: noviembre 19, 2010.] http://www.freedownloadmanager.org/es/downloads/dise%C3%B1ador_reportes_webgratis

[17]. **Barrero, Esnesto Leyva.** *Implementación del módulo de diseño de reportes para el SCADA Guardián del Alba.* Habana : s.n., 2009.

[18]. **Amigo, Yoandry Martínez Rodríguez y Ernesto Pérez. "***Sistema para el Análisis Cuantitativo de los Riesgos de los proyectos productivos del Centro de Desarrollo de Informática Industrial*. Habana : s.n., 2010.

GLOSARIO DE TERMINOS.

XML (e***X***tensible ***M***arkup ***L***anguage, 'lenguaje de marcas extensible'): Metalenguaje extensible de etiquetas desarrollado por el World Wide Web Consortium (W3C). Permite definir la gramática de lenguajes específicos, por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG y MathML.

Framework: Estructura de soporte definida en la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, un framework puede incluir soporte de programas, bibliotecas y un lenguaje interpretado entre otros software para ayudar a desarrollar y unir los diferentes componentes de un proyecto.

Informe o reporte: Documento contentivo de información personalizada y útil para el destinatario del mismo.

LGPL: Son las siglas de Lesser General Public License o Library General Public License. Esta licencia permite el enlace dinámico de aplicaciones libres a aplicaciones no libres.

GPL: Son las siglas de General Public License, Licencia Pública General, definida por la Fundación para el Software Libre (FSF) para proteger los derechos de copia del software libre.

HTML (***H***yper***T***ext ***M***arkup ***L***anguage, lenguaje de marcas hipertextuales): Lenguaje de marcación diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas web.

Generador de informes o reportes: Herramienta que permite generar informes, también conocidos como informes a partir de datos primarios.

Diseño de informe: Apariencia final con la cual será generada el informe.

CSV (en inglés *comma-separated values*): Son un tipo de documento sencillo para representar datos en forma de tabla

SCADA: Es una aplicación software especialmente diseñada para funcionar sobre ordenadores en el control de producción, proporcionando comunicación con los dispositivos de campo y controlando el proceso de forma automática desde la pantalla del operador. Además, provee a diversos usuarios, toda la información que se genera en el proceso productivo, control de calidad, supervisión y mantenimiento.

GTK: Es una biblioteca multiplataforma del equipo GTK+, la cual contiene los objetos y funciones para crear la interfaz gráfica de usuario. Maneja widgets como ventanas, botones, menús, etiquetas, deslizadores, pestañas,

SVG: Es un formato de gráficos para la web, el cual define un lenguaje basado en XML para la construcción de gráficos vectoriales de dos dimensiones (2d)

Informe persistente: Informe que una vez generado debe almacenarse en medios de almacenamiento de forma permanente en el tiempo.

Informe temporal: Informe generado con carácter temporal, razón por la cual no es necesario su almacenamiento permanente en el tiempo.

Multiplataforma: Es un término utilizado frecuentemente en informática para indicar la capacidad o características de poder funcionar o mantener una interoperabilidad de forma similar en diferentes sistemas operativos o plataformas.

ANEXOS

Tabla 12: "Configurador".

Nombre:	Configurador
Tipo	Controladora(Modelo)
Atributo	Nodo, componente, propiedades, color_fondo, imagen_fondo,
Operaciones	
Nombre	InspectorReportes(nodo)
Descripción	Es donde se crea el reporte en forma de nodo.
Nombre	Secciones(nombre, color_fondo, tamaño)
Descripción	Es el espacio de trabajo donde se crearán y modificarán los componentes relacionados con la sección.
Nombre	Componentes(nombre, id)
Descripción	Son los componentes que se necesitarán para la realización del reporte, estos se crearán y modificarán en las secciones.
Nombre	Ext.TabPanel()
Descripción	Trata las regiones de especificación de las Propiedades Generales del Reporte, páginas, consultas y secciones.
Nombre	Ext.Window()
Descripción	Crea la estación de trabajo Propiedades Reporte, donde se realizan las consultas, configura la página y tratamiento de secciones.
Nombre	Ext.menu.Menu()
Descripción	Crea la ventana Propiedades Reporte.
Nombre	Ext.tree.TreePanel()

Descripción	Crea el nodo en el Inspector de Reportes y trata el árbol de los componentes.
Nombre	Ext.grid.PropertyGrid()
Descripción	Es donde se crean las propiedades de los componentes.
Nombre	Ext.FormPanel()
Descripción	Para importar un Reporte.
Nombre	Ext.Panel()
Descripción	Crea las vistas de interfaz de usuario.
Nombre	Ext.dd.DropTarget()
Descripción	Arrastra el componente en las secciones correspondientes, muestra las propiedades de este en el panel Propiedades y se muestra en el Inspector de Reportes.
Nombre	Ext.Button
Descripción	Crea botón donde se realizan las funcionalidades Importar y Exportar Reporte.

Tabla 13: "InspectorReportes".

Nombre:	InspectorReportes
Tipo	Entidad(Interfaz)
Atributo	Nodo, componente
Operaciones	
Nombre	Renombrar()
Descripción	Cambia el nombre del reporte antes creado y del componente que se quiera modificar.
Nombre	Propiedades()

Descripción	Se muestran las propiedades generales de un diseño: consultas, páginas y tratamiento de secciones.
Nombre	Eliminar()
Descripción	Elimina el Reporte o un componente.
Nombre	Duplicar()
Descripción	Crea un nuevo componente similar al seleccionado, con las mismas características.
Nombre	Ext.menu.Menu()
Descripción	Es la función generadora del Reporte, colocándose este en el Inspector de Reportes en forma de nodo. Renombra, elimina y dupliica un componente, y muestra las propiedades de un diseño.
Nombre	Ext.tree.TreePanel()
Descripción	Crea el nodo en el Inspector de Reportes.
Nombre	Ext.dd.DropTarget()
Descripción	Se muestra el componente en el Inspector de Reportes.

Tabla 14: "Propiedades Reporte".

Nombre	Propiedades
Tipo	Entidad
Atributo	Nombre; título; descripción; Margen(superior, inferior, izquierdo, derecho); Orientación(vertical, horizontal); tamaño hoja; consulta; sección.
Operaciones	
Nombre	Ext.TabPanel()
Descripción	Trata las regiones de especificación de las Propiedades Generales del Reporte,

	páginas, consultas y secciones.
Nombre	Ext.Window()
Descripción	Crea la estación de trabajo Propiedades Reporte, donde se realizan las consultas, configura la página y tratamiento de secciones.

Tabla 15: "Secciones".

Nombre	Secciones
Tipo	Entidad(Interfaz)
Atributo	nombre, color_fondo, tamaño
Operaciones	
Nombre	Encabezado de Reporte(Título)
Descripción	Se usa para especificar el título del reporte que se generará.
Nombre	Encabezado de Página(nombre, fecha)
Descripción	Empleada para colocar generalidades referidas al reporte que permitan identificar la pertenencia de una hoja a un determinado reporte como nombre del reporte, fecha de generación.
Nombre	Detalle
Descripción	Se crean y modifican los diferentes componentes relacionados con esta sección.
Nombre	Pie de Página(númPag)
Descripción	Empleada para colocar el número de la página.
Nombre	Pie de Reporte()
Descripción	Se usa para dar fin a un reporte.
Nombre	Ext.menu.Menu()

Descripción	Es donde se crean las secciones de trabajo para diseñar el reporte.
Nombre	Ext.dd.DropTarget()
Descripción	Arrastra el componente en las secciones correspondientes

Tabla 16: "Componentes_Base_SVG".

Nombre	Componentes
Tipo	Controladora
Atributo	nombre, tipo, color_borde, color, ancho_borde, width, heigth, rotation
Operaciones	
Nombre	Avanzados(Campo Calculado, Campo de Datos, Fecha, Imagen, Número Página, Texto)
Descripción	Componentes avanzados para colocar en las secciones.
Nombre	Básicos(Círculo, Cuadrado, Línea, Rectángulo)
Descripción	Componentes básicos para colocar en las secciones.
Nombre	Gráfico(Barra, Puntos, Pastel, Lineas)
Descripción	Componentes gráficos para colocar en la sección detalle.
Nombre	Ext.tree.TreePanel()
Descripción	Trata el árbol de los componentes.
Nombre	Ext.dd.DropTarget()
Descripción	Arrastra el componente en las secciones correspondientes.

Tabla 17: Componente "Círculo".

Nombre	Círculo

Tipo	Entidad
Atributos	Nombre; Position_x; Position_y; color; color_fondo; radio; ancho_borde; isactuador; state; rotation; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades del círculo.
Nombre	create()
Descripción	Crea el componente círculo.
Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente círculo.

Tabla 18: Componente "Cuadrado".

Nombre	Cuadrado
Tipo	Entidad
Atributos	Nombre; color; Position_x; Position_y; color_fondo; longitud; ancho_borde; isactuador; state; rotation; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades del cuadrado.
Nombre	create()
Descripción	Crea el componente cuadrado.

Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente cuadrado.

Tabla 19: Componente "Línea".

Nombre	Línea
Tipo	Entidad
Atributo	Nombre; color; Position_x; Position_y; color_fondo; longitud; ancho_borde; isactuador; state; rotation; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades de la línea.
Nombre	create()
Descripción	Crea el componente línea.
Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente línea.

Tabla 20: Componente "Rectángulo".

Nombre	Rectángulo
Tipo	Entidad

Atributo	Nombre; color; Position_x; Position_y; color_fondo; width; height; ancho_borde; isactuador; state; rotation; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades del rectángulo.
Nombre	create()
Descripción	Crea el componente rectángulo.
Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente rectángulo.

Tabla 21: Componente "Texto".

Nombre	Texto
Tipo	Entidad
Atributo	Nombre; color; Position_x; Position_y; color_fondo; text; fontfamily; fontsize; letterspacing; align; ancho_borde; isactuador; state; rotation; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades del texto.
Nombre	create()
Descripción	Crea el componente texto.
Nombre	recize()

Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente texto.

Tabla 22: Componente "Fecha".

Nombre	Fecha
Tipo	Entidad
Atributo	Nombre; color; Position_x; Position_y; text; isactuador; state; rotation; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades del Fecha.
Nombre	create()
Descripción	Crea el componente Fecha.
Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente fecha.

Tabla 23: Componente "Imagen".

Nombre	Imagen
Tipo	Entidad
Atributo	Nombre; Position_x; Position_y; state; rotation; tipo; url;

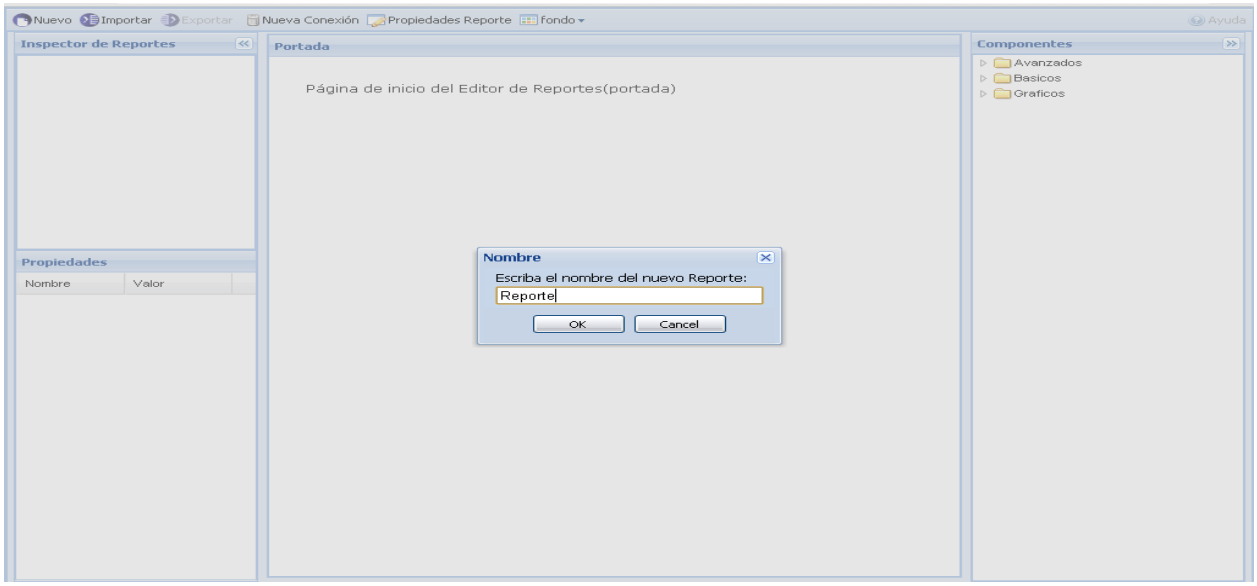
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades de la imagen
Nombre	create()
Descripción	Crea el componente imagen.
Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()
Descripción	Para exportar el componente imagen.

Tabla 24: Componente "Gráfica de Barra".

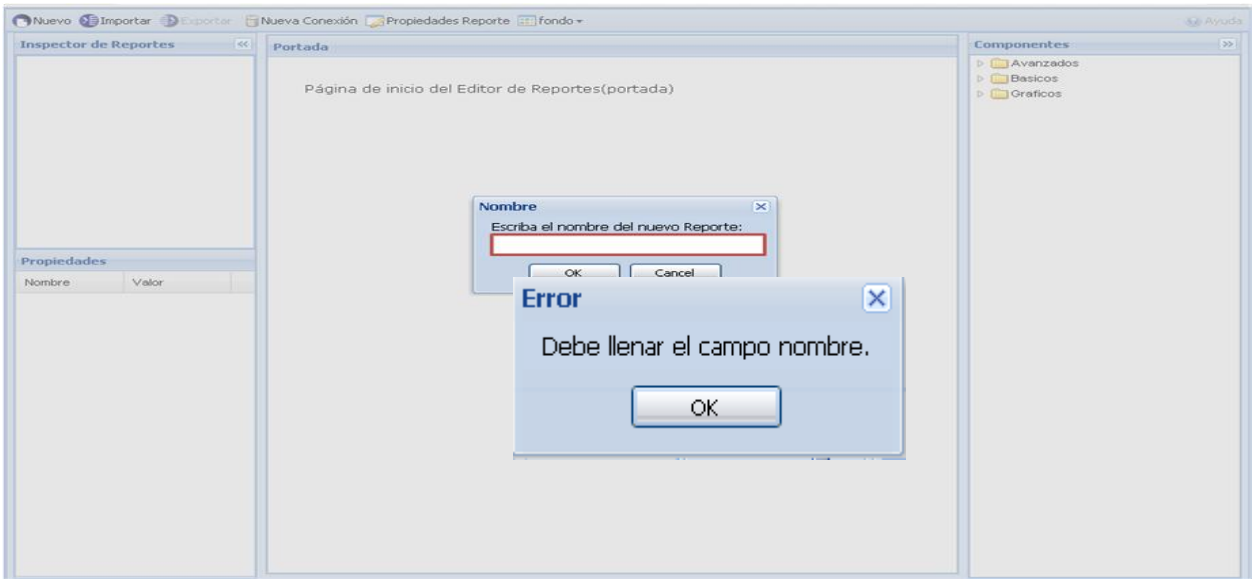
Nombre	Gráfica de Barra
Tipo	Entidad
Atributo	Nombre; Position_x; Position_y; isactuador; state; data; tipo;
Operaciones	
Nombre	Propiedades()
Descripción	Muestra y modifica las propiedades de la Gráfica de Barra.
Nombre	create()
Descripción	Crea el componente Gráfica de Barra.
Nombre	recize()
Descripción	Para redimensionar el componente.
Nombre	exportarXml()

Descripción	Para exportar el componente Gráfica de Barra.

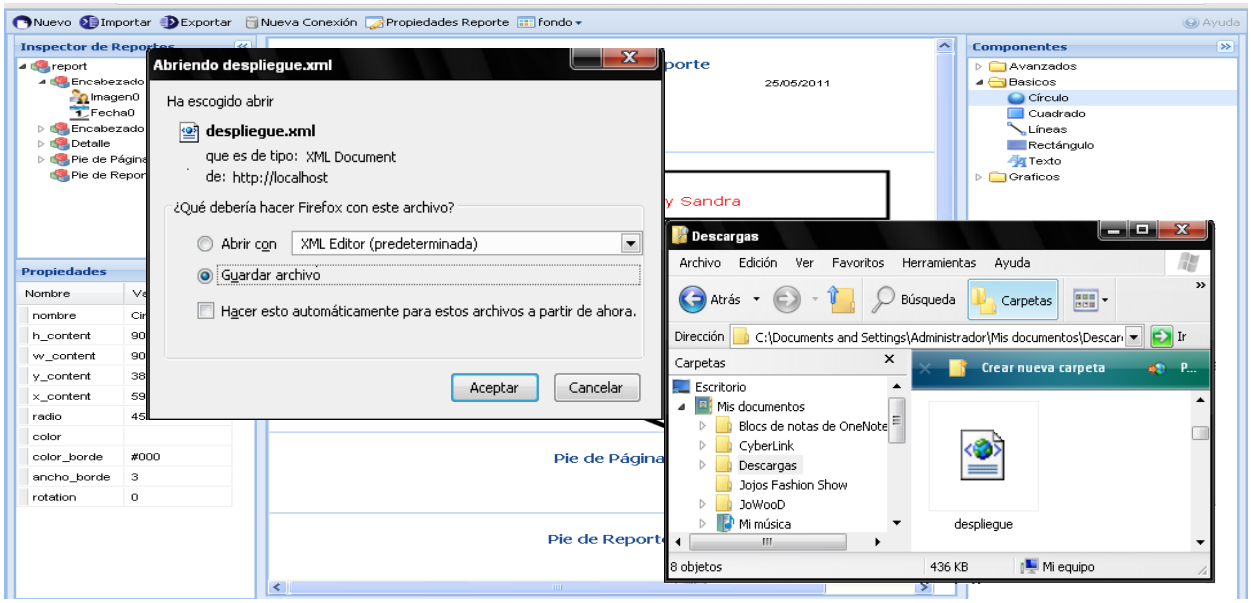
Pantalla 1: Crear Reporte.



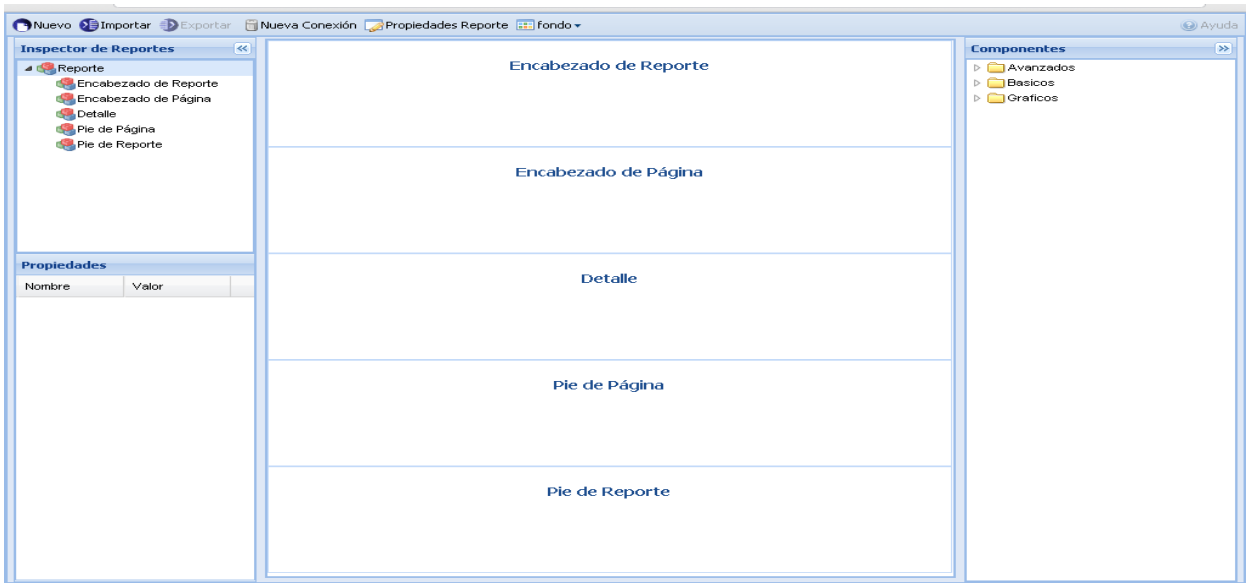
Pantalla 2: Crear Reporte.



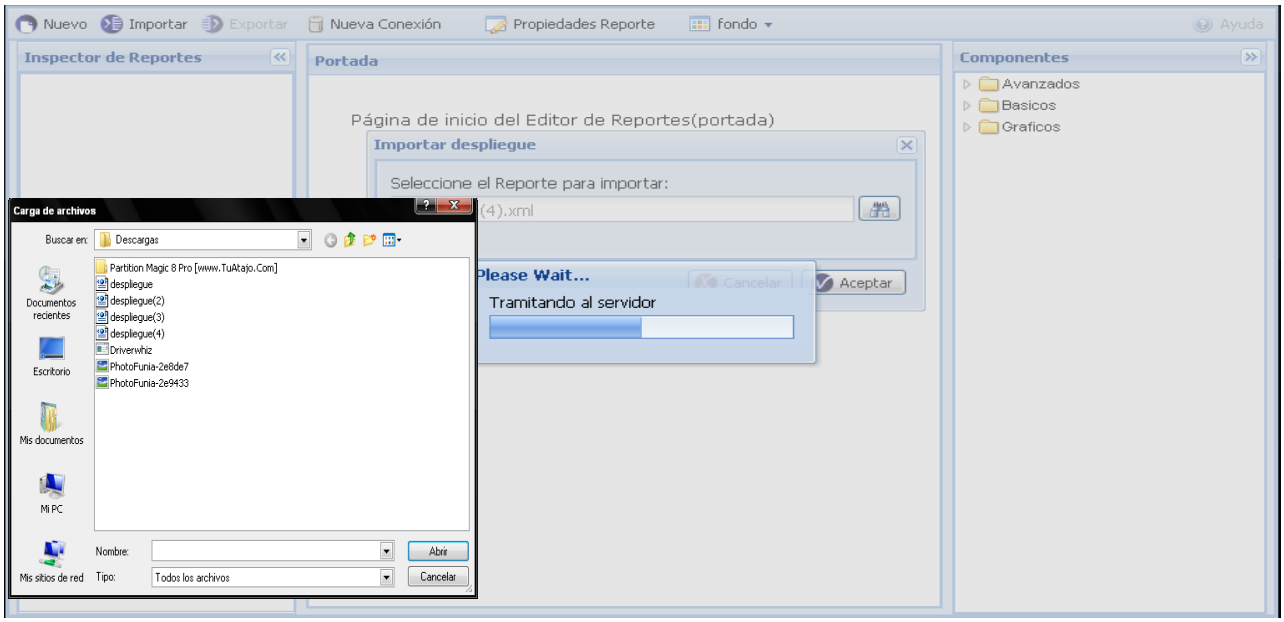
Pantalla 3: Exportar Reporte.



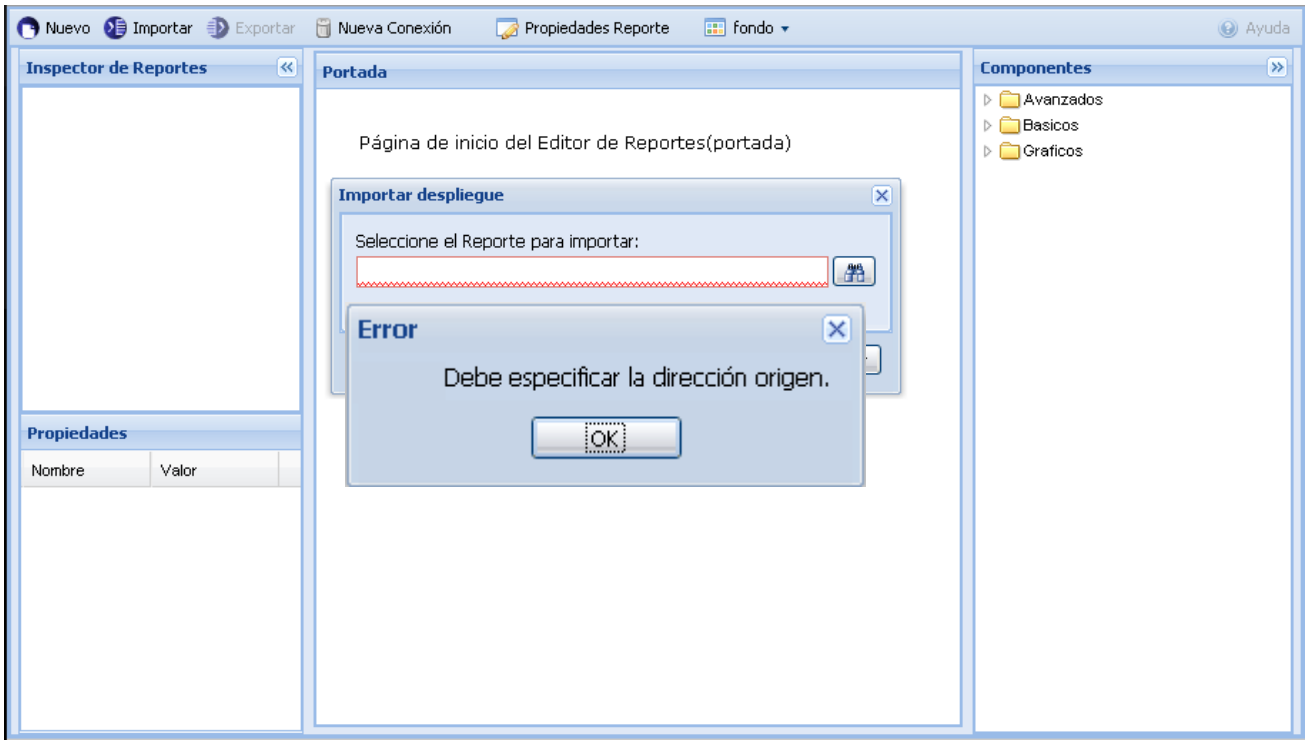
Pantalla 4: Exportar Reporte.



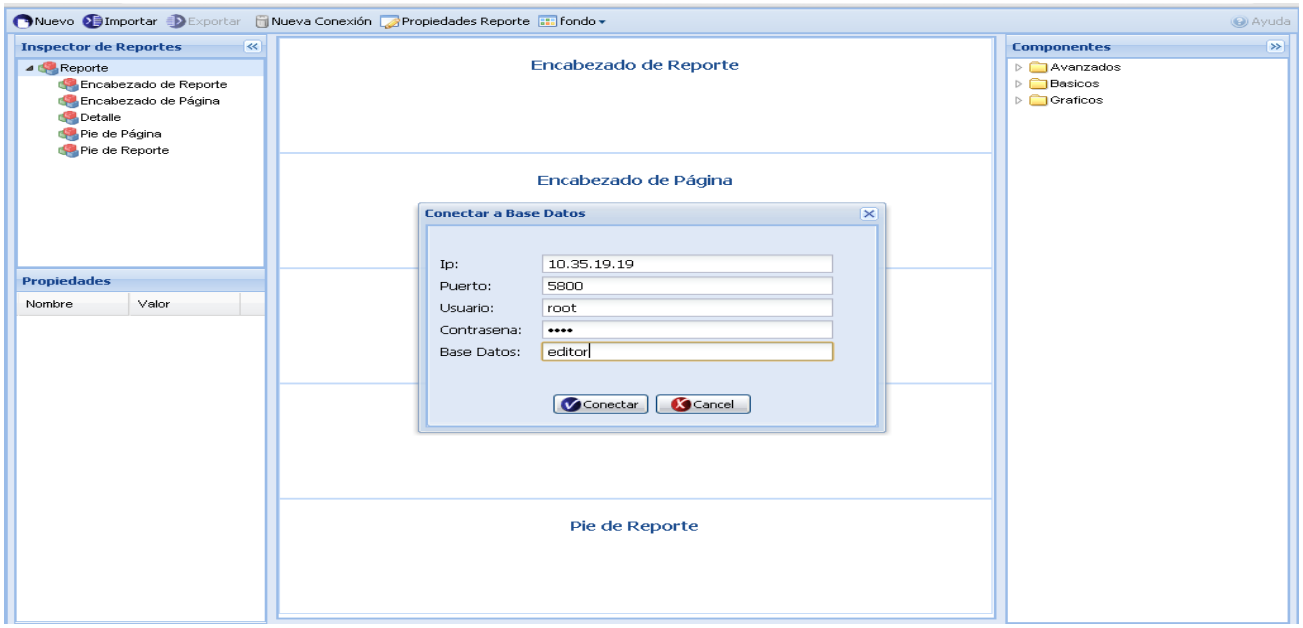
Pantalla 5: Importar Reporte.



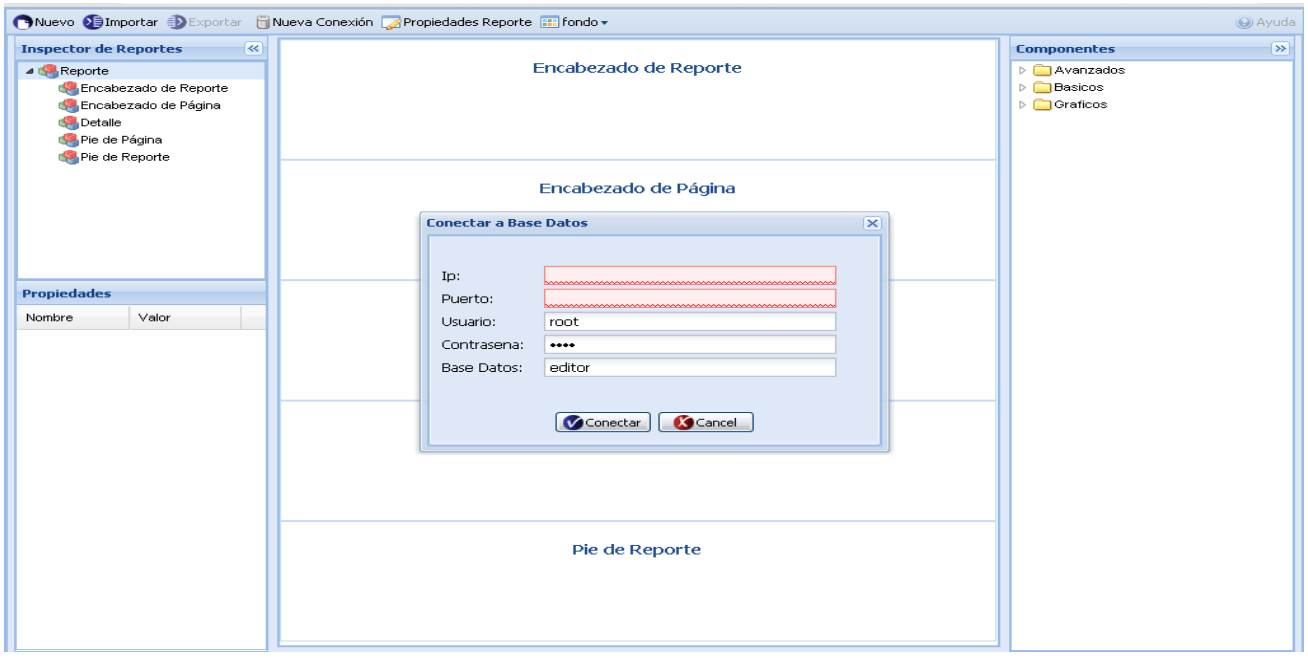
Pantalla 6: Importar Reporte.



Pantalla 7: Configurar Reporte.



Pantalla 8: Configurar Reporte.



Pantalla 9: Configurar Reporte.