



UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS
FACULTAD 5

**TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE
INGENIERO EN CIENCIAS INFORMÁTICAS.**

Título:

**Módulo de reportes para el Sistema Integral de
Confiability Operacional.**

Autor: Alberto Arístides Puentes González

Tutores: Ing. Yuniel Sardiñas García

Ing. Reinier Chávez La Rosa

La Habana Cuba, Junio 2012.

“Año 54 de la Revolución.”

DECLARACIÓN DE AUTORÍA

Declaración de autoría

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

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

Alberto Arístides Puentes González

Firma del Autor

Yuniel Sardiñas García

Firma del Tutor

Reinier Chávez La Rosa

Firma del Tutor

Datos de contacto

Tutor: Ing. Yuniel Sardiñas García.

Edad: 26

Ciudadanía: cubana.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ingeniero en Ciencias Informática.

E-mail: yunielsg@uci.cu

Tutor: Ing. Reinier Chávez La Rosa.

Edad: 29

Ciudadanía: cubana.

Institución: Universidad de las Ciencias Informáticas (UCI).

Título: Ingeniero en Ciencias Informática.

Categoría Docente: Instructor.

E-mail: rchavez@uci.cu

Dedicatoria

Dedico este trabajo de diploma a las personas más importantes en mi vida que son mis padres, especialmente a mi madre María Elena que tiene todo mi amor y dedicación. A mi padre José Arístides, a mi hermana Tatiana, a mi madrina y a su hija Jessica, a Lizy, a todos mis primos que son como hermanos y a mi abuela Amparo Santana que fue más que una madre. Para ustedes con todo el amor del mundo va el fruto de mi esfuerzo durante todos estos años.

Agradecimientos

A mi madre

Quisiera agradecer especialmente a mi madre por estar siempre a mi lado y darme su apoyo e incondicionalidad, todo mi amor y el mayor de los agradecimientos está contigo.

A mi padre

Agradecer a mi padre por haber formado mi carácter y personalidad, por estar presente en todo momento para apoyarme e indicarme el camino correcto.

A mis tutores

Gracias por haber confiado en mí para la realización de este trabajo y por ayudarme hasta el último momento, sin ustedes no hubiese podido completar esta difícil tarea.

Agradecer a toda mi familia que siempre me ha apoyado en los momentos difíciles, a mi madrina que estuvo presente como otra madre en la ausencia de mis padres y a su hija Jessica. A todos mis compañeros que de una forma u otra me han ayudado durante estos cinco años. A todos los profesores que han puesto en mí su granito de arena para llegar a donde estoy hoy. Para todos ustedes mi mayor agradecimiento y respeto.

Resumen

Los reportes son documentos que presentan de manera estructurada y resumida datos relevantes a los usuarios finales. Su correcta gestión en las empresas, por el personal autorizado, trae consigo un aumento significativo en la calidad de la toma de decisiones. En el Centro de Informática Industrial de la Universidad de las Ciencias Informáticas se desarrolla actualmente el proyecto Sistema Integral de Confiabilidad Operacional (SIC); este proyecto no cuenta con una herramienta capaz de generar reportes para los análisis de confiabilidad realizados a los activos de una empresa determinada. La presente investigación expone el diseño e implementación de un módulo de reportes capaz de satisfacer las necesidades del SIC, para lo cual se analizaron un conjunto de *softwares* actuales que posibilitan la generación de reportes con aspecto profesional, teniendo como premisa el uso de tecnologías libres.

Como resultado del presente trabajo se obtuvo un módulo de reportes que posibilita al SIC la obtención y presentación de la información procedente de los análisis de confiabilidad de forma estructurada y resumida, cumpliéndose de esta forma el objetivo propuesto.

Palabras clave: análisis de confiabilidad, activos, reportes, toma de decisiones.

Índice

INTRODUCCIÓN	1
CAPÍTULO 1 FUNDAMENTACIÓN TEÓRICA.	4
1.1 INTRODUCCIÓN.....	4
1.2 CONFIABILIDAD OPERACIONAL.....	4
1.2.1 <i>Aplicación de la Confiabilidad Operacional.</i>	5
1.3 REPORTE.	6
1.3.1 <i>Reportes de Análisis de Confiabilidad.</i>	7
1.3.2 <i>Tipos de reportes.</i>	7
1.4 MÓDULOS DE REPORTE PARA SISTEMAS CONFIABLES.	9
1.5 HERRAMIENTAS PARA GENERAR REPORTE.	11
1.6 METODOLOGÍA DE DESARROLLO Y HERRAMIENTA CASE.....	13
1.6.1 <i>RUP y Visual Paradigm.</i>	14
1.7 DESCRIPCIÓN DE LAS HERRAMIENTAS.	15
1.7.1 <i>Java y Groovy.</i>	16
1.7.2 <i>Grails y Google Web Toolkit (GWT).</i>	17
1.7.3 <i>PostgreSQL.</i>	18
1.8 CONCLUSIONES PARCIALES.	19
CAPÍTULO 2 CARACTERÍSTICAS DEL SISTEMA.	20
2.1 INTRODUCCIÓN.....	20
2.2 ESPECIFICACIÓN DE LOS REPORTE A REALIZAR.	20
2.3 COMPRENSIÓN DEL CONTEXTO DEL SISTEMA.....	20
2.3.1 <i>Modelo de Dominio.</i>	21
2.3.2 <i>Especificación de los requisitos.</i>	23
2.4 DESCRIPCIÓN DE LOS ACTORES DEL SISTEMA.	24
2.5 CASOS DE USO DEL SISTEMA.	25
2.5.1 <i>Diagrama de casos de uso del sistema.</i>	25
2.5.2 <i>Descripción textual de los casos de uso.</i>	26
2.6 CONCLUSIONES PARCIALES.	30
CAPÍTULO 3 ANÁLISIS Y DISEÑO DEL SISTEMA.	31
3.1 INTRODUCCIÓN.....	31
3.2 DESCRIPCIÓN DE LA ARQUITECTURA.	31
3.3 MODELO DE ANÁLISIS.	34
3.4 DISEÑO DEL SISTEMA.	34
2.4.1 <i>Diagrama de clases del paquete "View".</i>	35
2.4.2 <i>Diagrama de clases del paquete "Controller".</i>	37
2.4.3 <i>Diagrama de clases del paquete "Model".</i>	39
3.5 MODELO DE DESPLIEGUE.	41
3.6 CONCLUSIONES PARCIALES.	42
CAPÍTULO 4 IMPLEMENTACIÓN Y PRUEBA.	43
4.1 INTRODUCCIÓN.....	43
4.2 IMPLEMENTACIÓN DEL SISTEMA.	43
4.2.1 <i>Estilo de codificación.</i>	43
4.2.2 <i>Diagrama de Componentes.</i>	45
4.2.3 <i>Funcionamiento de JasperReports.</i>	47
4.2.4 <i>Secciones del XML de configuración del reporte.</i>	48
4.3 PRUEBAS DE SOFTWARE.	49

4.3.1 Diseño de casos de prueba.	50
4.3.2 Resultados de las pruebas.	55
4.4 VALIDACIÓN Y VERIFICACIÓN POR ESPECIALISTAS.	56
4.5 CONCLUSIONES PARCIALES.	57
CONCLUSIONES.	58
RECOMENDACIONES.	59
REFERENCIAS BIBLIOGRÁFICAS.	60
BIBLIOGRAFÍA.	61
GLOSARIO DE TÉRMINOS.	63
ANEXOS.	66
ANEXO 1: DIAGRAMA DE CLASES DEL ANÁLISIS DEL CASO DE USO “EXPORTAR REPORTE”	66
ANEXO 2: DIAGRAMA DE COLABORACIÓN DEL CASO DE USO “EXPORTAR REPORTE”.	67
ANEXO 3: DIAGRAMA DE SECUENCIA DEL CASO DE USO “EXPORTAR REPORTE”.	68
ANEXO 4: DIAGRAMA DE CLASES DEL ANÁLISIS DEL CASO DE USO “IMPRIMIR REPORTE”	69
ANEXO 5: DIAGRAMA DE COLABORACIÓN DEL CASO DE USO “IMPRIMIR REPORTE”.	70
ANEXO6: DIAGRAMA DE SECUENCIA DEL CASO DE USO “IMPRIMIR REPORTE”	71
ANEXO 7: REPORTE DE FALLAS FUNCIONALES DE UN GRUPO DE EQUIPOS DETERMINADO.	72
ANEXO 8: REPORTE DE LAS ESTRATEGIAS DE MANTENIMIENTO APLICADAS A UN GRUPO DE EQUIPOS DETERMINADO.	73
ANEXO 9: REPORTE DE LAS ACTIVIDADES DE MANTENIMIENTO ASOCIADAS A UN GRUPO DE EQUIPOS DETERMINADO.	74
ANEXO 10: DISEÑO DE CASOS DE PRUEBAS PARA EL CASO DE USO EXPORTAR REPORTE.	75
ANEXO 11: DISEÑO DE CASOS DE PRUEBAS PARA EL CASO DE USO IMPRIMIR REPORTE.	76

Índice de tablas

TABLA 1: TIPOS DE REPORTE POR CATEGORÍAS.	20
TABLA 2: DESCRIPCIÓN DE LOS ACTORES.....	25
TABLA 3: DESCRIPCIÓN DEL CASO DE USO “EXPORTAR REPORTE”.	26
TABLA 4: DESCRIPCIÓN DEL CASO DE USO “IMPRIMIR REPORTE”.....	28
TABLA 5: DESCRIPCIÓN DE LAS CLASES DEL PAQUETE “MVP”.....	37
TABLA 6: DESCRIPCIÓN DE CLASES DEL PAQUETE “CONTROLLER”.....	38
TABLA 7: DESCRIPCIÓN DE CLASES DEL PAQUETE “MODEL”.....	40
TABLA 8: DISEÑO DE CASOS DE PRUEBA DEL CASO DE USO “EXPORTAR REPORTE”.	50
TABLA 9: CASOS DE PRUEBAS APLICADOS AL CASO DE USO “EXPORTAR REPORTE”.	52
TABLA 10: DISEÑO DE CASOS DE PRUEBA DEL CASO DE USO “IMPRIMIR REPORTE”.	53
TABLA 11: CASOS DE PRUEBA APLICADOS AL CASO DE USO “IMPRIMIR REPORTE”.	54

Índice de figuras

FIGURA 1: <i>REPORTE EN FORMA DE LISTADO.</i>	8
FIGURA 2: <i>REPORTE CON DRILL DOWN.</i>	8
FIGURA 3: <i>REPORTE GRÁFICO (BARRAS).</i>	9
FIGURA 4: <i>FASES E ITERACIONES DE LA METODOLOGÍA RUP.</i>	14
FIGURA 5: <i>ARQUITECTURA DE GRAILS.</i>	17
FIGURA 6: <i>MODELO DE DOMINIO.</i>	22
FIGURA 7: <i>DIAGRAMA DE CASOS DE USO DEL SISTEMA.</i>	26
FIGURA 8: <i>PATRÓN MODELO VISTA CONTROLADOR (MVC).</i>	32
FIGURA 9: <i>MODELO VISTA PRESENTADOR (MVP).</i>	33
FIGURA 10: <i>REPRESENTACIÓN DE LOS PAQUETES DE DISEÑO.</i>	35
FIGURA 11: <i>REPRESENTACIÓN DEL PAQUETE DE DISEÑO “VIEW”.</i>	35
FIGURA 12: <i>DIAGRAMA DE CLASES CORRESPONDIENTE AL PAQUETE “MVP”.</i>	36
FIGURA 13: <i>DIAGRAMA DE CLASES DEL PAQUETE “CONTROLLER”.</i>	38
FIGURA 14: <i>DIAGRAMA DE CLASES DEL PAQUETE “MODEL”.</i>	39
FIGURA 15: <i>DIAGRAMA DE DESPLIEGUE DEL MÓDULO DE REPORTES.</i>	41
FIGURA 16: <i>DIAGRAMA DE COMPONENTES DEL MÓDULO DE REPORTES.</i>	46
FIGURA 17: <i>FUNCIONAMIENTO DE JASPERREPORTS.</i>	48
FIGURA 18: <i>SECCIONES DE LOS REPORTES.</i>	48
FIGURA 19: <i>GRÁFICA DE NO CONFORMIDADES POR CASOS DE USO.</i>	55
FIGURA 20: <i>DIAGRAMA DE CLASES DEL ANÁLISIS DEL CASO DE USO “EXPORTAR REPORTE”.</i>	66
FIGURA 21: <i>DIAGRAMA DE COLABORACIÓN DEL CASO DE USO “EXPORTAR REPORTE”.</i>	67
FIGURA 22: <i>DIAGRAMA DE SECUENCIA DEL CASO DE USO “EXPORTAR REPORTE”.</i>	68
FIGURA 23: <i>DIAGRAMA DE SECUENCIA DEL CASO DE USO “IMPRIMIR REPORTE”.</i>	69
FIGURA 24: <i>DIAGRAMA DE COLABORACIÓN DEL CASO DE USO “IMPRIMIR REPORTE”.</i>	70
FIGURA 25: <i>DIAGRAMA DE SECUENCIA DEL CASO DE USO “IMPRIMIR REPORTE”.</i>	71
FIGURA 26: <i>REPORTE DE FALLAS FUNCIONALES DE UN GRUPO DE EQUIPOS DETERMINADO.</i>	72
FIGURA 27: <i>REPORTE DE LAS ESTRATEGIAS DE MANTENIMIENTO APLICADAS A UN GRUPO DE EQUIPOS DETERMINADO.</i>	73
FIGURA 28: <i>REPORTE DE LAS ACTIVIDADES DE MANTENIMIENTO ASOCIADAS A UN GRUPO DE EQUIPOS DETERMINADO.</i>	74
FIGURA 29: <i>DISEÑO DE CASOS DE PRUEBAS PARA EL CASO DE USO EXPORTAR REPORTE.</i>	75
FIGURA 30: <i>DISEÑO DE CASOS DE PRUEBAS PARA EL CASO DE USO IMPRIMIR REPORTE.</i>	76

Introducción

En los últimos tiempos el mantenimiento dentro de la industria moderna ha sufrido un conjunto de cambios a nivel tecnológico, económico y social. Estas transformaciones vienen dadas por la competitividad actual del mercado internacional, lo que ha generado que la gestión de activos basada en Ingeniería de la Confiabilidad Operacional constituya una vía efectiva para las empresas en el logro de los planes trazados y los retos constantes a que son sometidas día a día.

El notable aumento de la productividad en función de la automatización de los procesos de producción y comercialización genera una constante e incremental demanda de *software*. Además, el desarrollo de aplicaciones para la implantación de mejores prácticas de operación, mantenimiento y control de cambios también ha ocupado un lugar de gran importancia en la industria de *software*. El proceso de toma de decisiones basado en una correcta información es uno de los factores claves para cualquier empresa; aprovechar los recursos que ofrece la tecnología para proveer información clara y confiable constituye un elemento diferenciador del cual depende en gran medida el éxito de la organización.

Con el desarrollo de programas informáticos, especializados en la realización de un gran número de cálculos complejos que permiten la toma de decisiones óptimas, los análisis de confiabilidad se hacen simples y pueden ser adoptados como parte del proceso diario del mantenimiento de los activos de una empresa.

Cuba no se encuentra exenta del proceso de modernización de la sociedad y cuenta hoy día con la empresa comercializadora de soluciones informáticas ALBET Ingeniería y Sistemas S. A. La Universidad de las Ciencias Informáticas (UCI), principal productora de programas informáticos en Cuba, es una de las entidades que comercializa sus productos a través de ALBET y cuenta actualmente con 7 facultades enfocadas al desarrollo de *software* en distintas esferas. Específicamente en la facultad 5 se encuentra el Centro de Informática Industrial (CEDIN) dentro del cual se desarrolla el proyecto "Sistema Integral de Confiabilidad Operacional" (SIC).

Este proyecto tiene sus bases en análisis estadísticos y de condición para mantener la confiabilidad de los activos, permitiéndole a la empresa operar durante un largo período de tiempo sin perder su función. Actualmente existen dificultades en la interpretación de estos

análisis y con ello en el proceso de toma de decisiones, debido a que los datos almacenados no se obtienen de forma estructurada y resumida, provocando que no se muestre de forma asequible el comportamiento de los activos y las acciones que sobre ellos se realizan.

Partiendo de la **problemática** planteada surge como **problema científico** la siguiente interrogante: ¿Cómo facilitar la obtención y presentación de la información procedente de los análisis de confiabilidad?

Para darle respuesta a esta pregunta la presente investigación tiene como **Objeto de Estudio** los sistemas generadores de reportes informáticos.

El **Objetivo General** de este trabajo es: Desarrollar una herramienta, usando tecnologías libres, que permita la obtención y presentación de la información procedente de los análisis de confiabilidad; teniendo como **Campo de Acción**: Los sistemas generadores de reportes informáticos para análisis de confiabilidad.

Para darle cumplimiento al objetivo planteado se proponen una serie de **Tareas de Investigación** que se relacionan a continuación:

- Determinación de las tendencias actuales de los sistemas de confiabilidad y los reportes que en estos se realizan para lograr una mayor familiarización con el entorno de la problemática.
- Selección de las herramientas libres convenientes para dar cumplimiento a los requerimientos del cliente.
- Confección del modelo conceptual para dividir, comprender y representar los conceptos relacionados con el dominio del sistema.
- Identificación y descripción de los requerimientos del sistema.
- Representación y descripción de los Casos de Uso del Sistema (CUS) para documentar el comportamiento del sistema desde el punto de vista del usuario.
- Diseño e implementación de un módulo de reportes para el Sistema Integral de Confiabilidad Operacional.
- Realización de las pruebas necesarias para validar la solución obtenida.

En la realización de este trabajo se hizo uso de los métodos teóricos y empíricos

pertenecientes a los métodos científicos de investigación para el estudio de los sistemas generadores de reportes.

Métodos Teóricos:

- Con el objetivo de extraer las características que distinguen los sistemas generadores de reportes informáticos y seleccionar los elementos más importantes y que están más relacionados con el objeto de estudio planteado se utilizó el método **Analítico-sintético**.
- Con el objetivo de conocer sobre la evolución de los sistemas generadores de reportes informáticos y sus tendencias actuales, así como los principales hitos que surgen en el desarrollo de un sistema como este se utilizó el método **Análisis histórico lógico**.
- Con el objetivo de permitir la representación de las ideas concebidas en un *software* de este tipo se utilizó el método de **Modelación**.

Métodos Empíricos:

- Con el objetivo de seleccionar la información necesaria en la investigación a partir del estudio de documentos y diferentes bibliografías se utilizó el método **Revisión de la documentación**.

El contenido del documento está compuesto por resumen, introducción, cuatro capítulos de contenido, conclusiones, recomendaciones, referencias bibliográficas, bibliografía consultada, glosario de términos y anexos.

El primer capítulo, Fundamentación Teórica, abarca los conceptos fundamentales de Confiabilidad Operacional y reportes, así como los sistemas generadores de reportes y herramientas necesarias para el desarrollo del módulo. El segundo capítulo, Características del Sistema, expone el negocio del sistema, la especificación de requisitos y los casos de uso determinados. En el tercer capítulo, Análisis y Diseño del Sistema, se diseña un sistema de clases teniendo en cuenta las técnicas y normas de la Programación Orientada a Objetos (POO) para luego ser implementadas en el cuarto y último capítulo. En este se obtiene un prototipo funcional del módulo y se realizan las pruebas necesarias para lograr la calidad requerida del producto final.

Capítulo 1 Fundamentación Teórica.

1.1 Introducción.

En el presente capítulo se abordan temas relacionados con la Confiabilidad Operacional y los reportes emitidos a partir de esta práctica argumentando la importancia que estos alcanzan a nivel empresarial. Además, se presenta un estudio de las tecnologías actuales para la realización de reportes empresariales con carácter profesional, así como una descripción de las herramientas previamente seleccionadas para el proyecto Sistema Integral de Confiabilidad Operacional, las cuales se integrarán en el presente módulo.

1.2 Confiabilidad Operacional.

La Confiabilidad Operacional es la capacidad que posee la empresa a través de los procesos, las tecnologías y las personas para cumplir con su propósito dentro de los límites del diseño y de las condiciones operacionales. (1)

El objetivo principal de la misma es propiciarle a las industrias soluciones que permitan un incremento en la capacidad productiva y el rendimiento financiero sobre sus activos sin efectuar nuevas inversiones en activos adicionales. Para ello considera una serie de procesos de mejora continua que incorporan en forma sistémica herramientas de diagnóstico, metodologías de análisis y nuevas tecnologías para la gestión, planeación, ejecución y control asociados con la producción, el abastecimiento y el mantenimiento industrial.

Estos procesos para lograr un mejoramiento continuo y de largo plazo se rigen por un conjunto de parámetros operativos que constituyen las bases de la Confiabilidad Operacional, los cuales son: (1)

- Confiabilidad del equipamiento: integra las estrategias empleadas, la efectividad del mantenimiento y el incremento del tiempo medio entre fallos de los activos.
- Confiabilidad de los procesos: incluye la operación confiable dentro de los parámetros de diseño, así como la correcta interpretación de los procesos y los procedimientos.
- Confiabilidad humana: comprende el grado de compromiso y sentido de pertenencia, así como el nivel de conocimientos y habilidades requeridos.
- Mantenibilidad: garantiza la continuidad de la confiabilidad inherente y la disminución

del tiempo medio para reparar los activos.

Estos parámetros deben estar presentes en los programas de mejora de la confiabilidad operacional y no se deben atender de forma aislada, puesto que ignorar uno de ellos puede comprometer los beneficios a largo plazo que de esta se derivan. La confiabilidad del equipamiento y la mantenibilidad constituyen los parámetros operativos dentro del contexto del módulo a desarrollar de mayor importancia, ya que están directamente relacionados con la operación y el mantenimiento en el ciclo de vida de los activos, aspectos que se emplearán para la generación de reportes en el SIC.

1.2.1 Aplicación de la Confiabilidad Operacional.

La Confiabilidad Operacional constituye un elemento de gran importancia en el mantenimiento industrial teniendo una fuerte aplicación en los casos relacionados con: (2)

- Elaboración de planes y programas de mantenimiento e inspección de equipos estáticos y dinámicos.
- Solución de los problemas recurrentes en equipos e instalaciones que afectan los costos y la efectividad de las operaciones.
- Determinación de las tareas que permiten minimizar riesgos en los procesos, equipos, instalaciones y medio ambiente.
- Establecer el alcance y frecuencia óptima de paradas de plantas.
- Establecer procedimientos operacionales y prácticas de trabajo seguro.

Estas aplicaciones son de gran relevancia para las industrias ya que permiten minimizar la ocurrencia y recurrencia de fallos, entender el “Por qué” de estos fallos y prevenirlos eficazmente maximizando el desempeño del sistema y usando eficientemente los recursos. Para esto se necesitan herramientas que permitan el análisis de las fallas esperadas en un período de tiempo futuro y las consecuentes correcciones a realizar para prevenir o al menos mitigar estas. La generación de reportes constituye una de las principales funcionalidades en estas herramientas para garantizar una interpretación correcta de estos análisis.

1.3 Reportes.

Un reporte es un documento, una noticia o un informe que brinda información con algún propósito y que presenta de manera estructurada y/o resumida datos relevantes acerca de uno o varios temas en específico. (3)

En el campo de la informática, los reportes son informes que organizan y exhiben la información contenida en una fuente de datos. El ciclo de vida de los reportes está formado por la definición del informe y la representación de estos. Su definición es un archivo que contiene toda la estructura, especificando la forma de conexión con la fuente de datos, las consultas utilizadas para recuperar estos datos, expresiones, parámetros, imágenes, tablas, cuadros de textos y cualquier otro elemento de diseño que pueda incluirse. Su representación es la especificación de un formato estándar para posibilitar la visualización de los datos representados en su definición. Un informe básico debe especificar qué datos necesita, cómo desea organizarlos en la página y cómo desea que los usuarios vean el informe.

Los reportes son documentos que se utilizan cuando se quiere informar de una cuestión determinada, por lo que generalmente agrupan datos de acuerdo a un interés específico, estos datos no pueden ser accedidos para modificarlos directamente y pueden hacer uso de varios elementos (gráficos, tablas, entre otros) para su representación. Dentro de los elementos que no deben faltar en estos se encuentran el título, el nombre del autor y la fecha de realización ya que constituyen elementos distintivos de gran importancia para el uso de la información que presentan. Los reportes se utilizan como instrumento de gestión propiciando información actualizada sobre un determinado tema, la cual puede utilizarse para la futura planificación y toma de decisiones. Además, actúan como un medio eficaz de comunicación dentro de las organizaciones proporcionando información de utilidad a sus empleados para la solución de problemas existentes.

Los reportes pueden clasificarse en 3 categorías comunes a nivel institucional. Los reportes periódicos, los cuales se producen a intervalos de tiempos regulares por necesidad de la institución, ejemplo de estos son los reportes de ventas mensuales de una tienda. Los reportes de excepción, estos indican acontecimientos inusuales en la institución generándose de forma automática por una herramienta destinada para ello. Por último y no menos importante los reportes a solicitud, los cuales son realizados por petición expresa con el objetivo de analizar, comprobar o medir un conjunto de parámetros determinado, esta categoría es la que se tendrá

en cuenta para realizar los reportes procedentes de los análisis de confiabilidad ya que estos se efectúan a solicitud de los ingenieros de confiabilidad.

1.3.1 Reportes de Análisis de Confiabilidad.

El análisis de confiabilidad, disponibilidad y mantenibilidad de sistemas de producción durante las diferentes fases de un proyecto proporciona reportes que permiten planear, organizar, controlar y comunicar situaciones presentes en los diferentes escenarios analizados permitiendo responder al instante preguntas como: (4)

¿Qué tan productivo será el proceso?

¿Qué mejoras se pueden introducir?

¿Qué trabajos quedan por hacer?

¿Cuáles son las actividades críticas?

¿Dónde concentrar los esfuerzos y recursos?

Estos reportes constituyen un instrumento muy útil para mejorar la confiabilidad y seguridad de los activos de una empresa creando un medio de apoyo, normalmente a través de *software*, al proceso de reportar, clasificar, analizar fallas y planear las acciones correctivas en respuesta a estas fallas. Estos tienen como propósito recolectar en forma sistémica datos de fallas, análisis y gestión de las acciones correctivas con el objetivo de apoyar la toma de decisiones oportuna a nivel empresarial. Teniendo en cuenta la información que proveen estos reportes y la importancia que alcanzan a nivel industrial se hace necesario la búsqueda de la mejor solución para mostrar la información a los ingenieros de confiabilidad según las especificaciones en el SIC.

1.3.2 Tipos de reportes.

Existen numerosas formas para mostrar información en un reporte, el uso de cada una de estas está en dependencia de los datos que se quieran mostrar y el objetivo que se quiera alcanzar.

A continuación se muestran los tipos de reportes más usados a nivel empresarial. La figura 1 presenta la estructura que poseen los reportes en forma de listados.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Consulta de Proveedores

Id	Proveedor	Teléfono	Fax	Tipo de Proveedor	Activo	Borrar
8	Zapaterías Deportivas, S.A. de C.V.			Gasto	Si	
5	HMS: Human Metric System			Compra de Producto	No	
11	Alejandro Corrales Gómez			Gasto	Si	
12	Seguridad Industrial ONIX, S.A. de C.V.			Gasto	Si	

Figura 1: Reporte en forma de listado.

En la figura 1 se muestra un reporte en forma de listado, estos son los reportes más sencillos de realizar, consisten en un despliegue de información en forma de listado acerca de elementos deseados (productos, clientes, proveedores). Estos se usan principalmente con el objetivo de presentar cierta cantidad de datos sin un alto nivel de análisis.

La figura 2 muestra como se estructuran los reportes con *Drill Down* para mostrar información.

Categoría	Cantidad	Total
Terrenos	8.00	11,390,322.48
Permisos	16.00	35,676.83
Estudios y Diseño	22.00	90,800.00
Maquinaria	1,329.50	430,106.20
Mano de Obra	189,477.98	4,753,096.29
Materiales	1,553,322.04	5,986,075.17
Sub Categoría	Cantidad	Total
Albañilería	171,730.44	2,483,571.69
Plomería	247,517.91	251,078.60
Electricidad	63,313.70	295,000.00
Acabados	1,070,091.99	2,455,749.88
Jardín	667.00	36,675.00
Producto	Cantidad	Total
Pasto San Agustín	650.00	15,275.00
Tierra Negra 14m	5.00	13,000.00
Encino Siempre Verde 2"	12.00	8,400.00
		36,675.00
Pintura, Plafones e Imper	1.00	464,000.00
		5,986,075.17
Otros Gastos	113.76	17,595.93
		22,703,672.90

Page Size: 10 | Prev 1 Next | Displaying page 1 in 1, items 1 to 7 of 7.

Figura 2: Reporte con *Drill Down*.

En la figura 2 se muestra un reporte con *Drill Down*, estos reportes se usan para ir filtrando información por categorías previamente definidas y así tener una mayor organización de la información. Permiten el desglose de la información en función de la búsqueda por lo que están orientados a la web. Se usan principalmente cuando la cantidad de información a representar es amplia y puede agruparse por categorías.

La figura 3 muestra como se estructuran los reportes gráficos de barra para presentar la información.

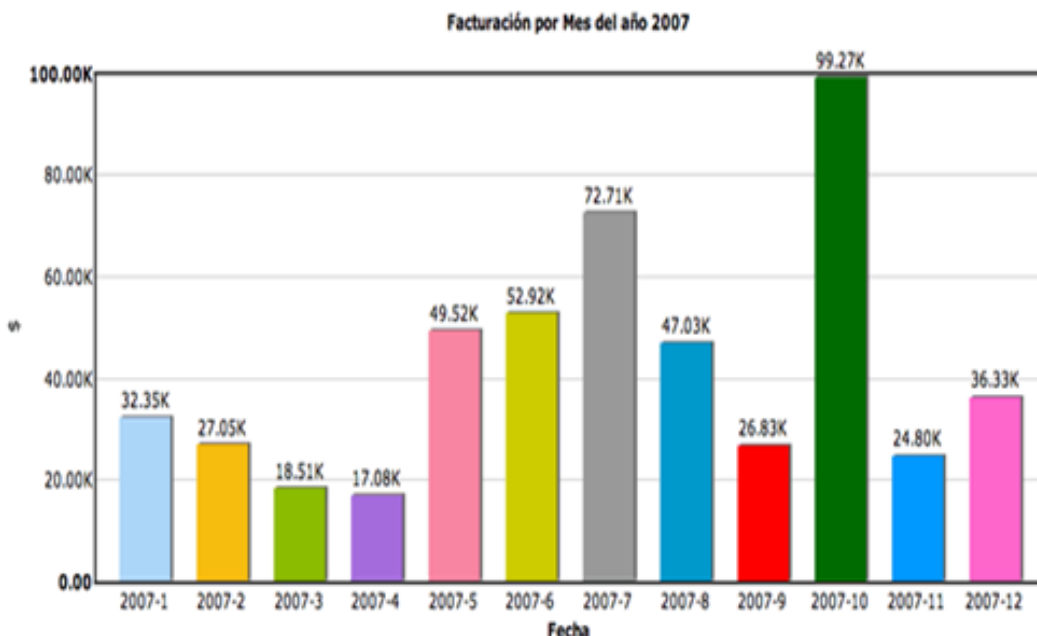


Figura 3: Reporte gráfico (Barras).

En la figura 3 se muestra un reporte gráfico, estos se centran en brindar información rápida y visual al usuario, este puede visualizar la información en gráficas lineales, de barra y de pastel. Se utilizan para observar comportamientos de estado como ventas, compras, utilidades, comparativo de proyectos y otros.

Los tipos de reportes a tener en cuenta para presentar la información procedente de los análisis de confiabilidad serán los reportes en forma de listados combinados con los reportes gráficos, estos brindan una solución factible para el proceso de toma de decisiones facilitando la mayor cantidad de información posible de forma resumida y estructurada. De esta forma se proporcionará a los ingenieros de confiabilidad un conjunto de meta-información de gran relevancia además de proporcionar los datos necesarios.

1.4 Módulos de reportes para sistemas confiables.

La presentación de la información para sistemas confiables constituye un aspecto fundamental para la realización de análisis de comportamiento y el proceso de toma de decisiones, no se concibe un sistema confiable que no presente un módulo para la realización de reportes.

La Solución Empresarial para Sistemas de Potencia de Electricidad (ETAP) se encarga de la

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

confiabilidad de sistemas de potencia a partir de los cálculos de índices de coste de energía y análisis de sensibilidad. ETAP ofrece una *suite* de programas totalmente integrados para la supervisión, simulación y optimización continua de sistemas eléctricos. Dentro de las herramientas que brinda se encuentra un módulo de reportes para la advertencia de temperaturas límites, ajustes óptimos, análisis de sensibilidad e índices de confiabilidad a partir de gráficas de todo tipo, permitiendo exportar los reportes realizados a formatos reconocibles por *softwares* de edición lo que proporciona un alto grado de configuración. Este módulo está previsto principalmente para instalaciones eléctricas de alto impacto y no para ser aplicado a cualquier industria, además forma parte de un producto privativo por el que hay que pagar altos precios para su adquisición.

La *suite* Business Reliability & Integrity Management System (BRIMS) está diseñada con base en estándares de la industria y enfocada a la evaluación, valoración y administración de riesgos asociados a la integridad y confiabilidad de los activos del negocio. Esta *suite* cuenta con la herramienta TComm la cual permite incluir la información y características de los diferentes activos de una industria para el control y seguimiento de preguntas de ingeniería, no conformidades e instrucciones de cambio para gestionar mejor el proyecto o el activo en operación. Para cumplir con su propósito TComm cuenta con varios módulos, entre ellos se destaca un módulo que permite la generación de reportes gerenciales como archivos en diferentes formatos. El uso de esta herramienta está limitado por cuestiones económicas, ya que la empresa Tecnicontrol S.A. es propietaria de esta y cobra grandes sumas por su implantación.

Otro sistema encargado de la realización de reportes dentro del concepto de confiabilidad y mantenibilidad es el Módulo de Reportes de Detenciones del *software* SAM. Este módulo se encarga de ingresar las detenciones ocurridas y sus causas para cada uno de los activos de las áreas de producción, recolectando valiosa información que permite a los departamentos de confiabilidad analizar a fondo como mejorar la disponibilidad las plantas. El módulo SAM es desarrollado por la empresa SPM Ingeniería en Mantenimiento, esta se encargada de implantarlo y darle soporte cobrando muy altos precios por ello.

Son muchos los sistemas que cuentan con módulos de reportes para presentar información determinada de forma estructurada y resumida. La reutilización de estos en la mayoría de los casos es bien complicada porque generalmente forman parte de *softwares* concebidos para un

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

negocio específico con propietarios que cobran grandes sumas de dinero por su utilización. Debido a esto se hace necesario el desarrollo de un módulo completamente nuevo que permita la generación de reportes ajustado al negocio existente.

1.5 Herramientas para generar reportes.

Los generadores de reportes son herramientas que permiten explotar la información contenida en fuentes de datos mediante la elaboración de informes que cubran las necesidades del cliente. La importancia de la realización de reportes está dada por la organización y presentación a los diversos públicos de interés de la empresa de toda la información relacionada con los ámbitos sociales, ambientales y económicos de la misma. Esto hace que la utilización de herramientas para la generación de reportes constituya una necesidad de toda empresa moderna. Actualmente existen múltiples herramientas que permiten la generación de informes que son de gran utilidad a nivel empresarial.

DataVision es una de estas herramientas para generar reportes desarrollada en Java donde los reportes pueden ser ejecutados, visualizados e impresos desde la aplicación o exportados como HyperText Markup Language (HTML), Extensible Markup Language (XML), Portable Document Format (PDF) y Microsoft Excel (XLS). Los reportes pueden ser diseñados desde una interfaz gráfica y generados desde cualquier base de datos que posea el manejador para la interfaz de programación de aplicaciones JDBC (Java Database Connectivity). (5)

Business Intelligence and Reporting Tools (BIRT) es un sistema de reporte para aplicaciones web basado en Eclipse, especialmente en Java y Java 2 Enterprise Edition (J2EE). Posee dos componentes principales, un diseñador de informes basado en Eclipse y un componente de tiempo de ejecución, los cuales se pueden agregar al servidor de aplicaciones. Ofrece un motor gráfico que le permite agregar gráficas a las aplicaciones de forma personalizada. (6)

DataVision y Business Intelligence and Reporting Tools poseen dos componentes principales, un diseñador de informes y un componente de tiempo de ejecución para generar informes que pueden ser integrados en cualquier entorno Java. Estas herramientas que cuentan con diseñadores gráficos integrados deben ser empaquetadas con múltiples componentes, incluyendo un conjunto de funcionalidades en forma de *plug-ins* que no serán utilizadas en la presente aplicación lo que aumenta notablemente el tamaño de esta, aspecto por el cual se decide obviarlas en la selección.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Crystal Reports es un producto de alta tecnología para la creación e integración de reportes con información proveniente de múltiples fuentes de datos. Provee numerables opciones de conectividad a datos incluyendo más de 30 *drivers* para acceso a bases de datos relacionales y fuentes de datos XML. La tecnología flexible de diseño de Crystal Reports provee control completo sobre el acceso y la presentación de los datos en los reportes. Esta herramienta habilita la visualización e interacción con los reportes a través de una amplia variedad de dispositivos y entornos. Los visores web avanzados habilitan a los usuarios finales para realizar búsquedas dentro de los datos de un reporte y exportarlas posteriormente a Microsoft Excel, Word y páginas HTML con el vínculo dinámico al reporte original. Adicionalmente, el reporte completo puede ser exportado a una variedad de formatos incluyendo XML, PDF, HTML y Microsoft Excel. (7)

Active Reports 6 es un componente de informes para aplicaciones windows y formularios web. Entre sus principales características figuran la personalización, alto rendimiento, alta calidad y prestaciones multilingües. Admite exportaciones de datos a todos los formatos habituales como HTML, PDF, XML y Microsoft Excel. Active Reports 6 incluye un diseñador de informes fácil de usar y una potente Interfaz de Programación de Aplicaciones (API, por sus siglas en inglés). (8)

Crystal Reports y Active Reports 6 constituyen potentes herramientas para la generación de reportes basadas en .NET *framework*, estas cuentan con licencias privativas y no son compatibles con las tecnologías definidas para la implementación del SIC por lo que no cumplen con las premisas fundamentales para el desarrollo de la aplicación.

JasperReports es una poderosa herramienta de código abierto para generar reportes que tiene la habilidad de presentar contenido en la pantalla, prepararlo para impresión o exportarlo a los archivos PDF, HTML, XLS, XML, OpenOffice y Word. Está implementado completamente en Java y se puede utilizar en aplicaciones J2EE o en la web para generar contenido dinámico. No requiere de un servidor de reportes externo y puede ser fácilmente vinculado con el diseñador de reportes visual IReport. Es extensible a cualquier fuente de datos y en un mismo reporte puede hacer uso de más de una de estas. (9)

JasperReports es una de las herramientas que brinda JasperSoft Business Intelligence Suite de un conjunto de herramientas muy poderosas que permiten a las organizaciones generar información basada en sus datos de administración para la evaluación y toma diaria de decisiones en forma dinámica y *on-line*. Es el motor que ejecuta los reportes y uno de los de

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

mayor utilización en el mundo de código abierto integrándose en aplicaciones web y de escritorio bajo código Java o Groovy, los cuales constituyen los lenguajes de programación especificados para el desarrollo del SIC. Cuenta con la herramienta IReport, totalmente independiente para el diseño de sus reportes, por lo que no interfiere en el tamaño del producto final y posibilita la personalización de los reportes con un alto grado de profesionalidad.

Este motor de reportes brinda amplias funciones de diseño posibilitando mostrar tablas con doble entrada (*croostabs*), gráficos, *widgets* y subinformes para diseño de alta complejidad, proporcionando una salida con precisión de píxeles orientado a la página, la web o la impresión. Posee una arquitectura escalable ya que las consultas de control hacen uso de los recursos del sistema, el visualizador de reportes optimiza el consumo de memoria y el tamaño del reporte es ilimitado. Su implementación cuenta con un alto grado de flexibilidad permitiendo la conexión con bases de datos relacionales, Hibernate, Enterprise JavaBeans (EJB) y XML.

Los aspectos antes expuestos y la perfecta integración con las herramientas definidas para el desarrollo del proyecto SIC son la base para la selección de JasperReports como herramienta central para el desarrollo de la aplicación.

1.6 Metodología de desarrollo y herramienta CASE.

Una metodología de desarrollo de *software* es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar determinado producto informático. Puede seguir uno o varios modelos de ciclo de vida, indicando la forma de obtener los distintos productos parciales y finales.

Principalmente se encarga de elaborar estrategias de desarrollo que promuevan prácticas adaptativas en vez de predictivas; centradas en las personas o los equipos y en una comunicación intensiva, la cual requiere la implicación directa del cliente.

Dentro de las metodologías más comunes para el proceso de desarrollo de *software* se encuentran:

- Proceso Unificado de Desarrollo de Software (RUP).
- Programación Extrema (XP).
- Scrum.

Durante el proceso de desarrollo se hace uso de diversas herramientas dentro de las que se

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

encuentran las encargadas de automatizar los aspectos claves de este, conocidas como herramientas de Ingeniería Asistida por Computadora (CASE, por sus siglas en inglés). Estas brindan principalmente un fuerte apoyo a la documentación a través de diagramas establecidos por la metodología.

1.6.1 RUP y Visual Paradigm.

RUP es una metodología de desarrollo de *software* que está basada en componentes e interfaces bien definidas. Es un conjunto de actividades necesarias para convertir los requisitos del usuario en funcionalidades del sistema. Está caracterizado por tres elementos fundamentales que son: dirigido por casos de uso, iterativo e incremental y centrado en la arquitectura.

Esta metodología permite una forma disciplinada de asignar tareas y responsabilidades de acuerdo a quién hace qué, cuándo y cómo. Utiliza el Lenguaje Unificado de Modelado (UML, por sus siglas en inglés) como lenguaje de modelado, el cual se encuentra orientado a objetos y genera una gran cantidad de documentación, elemento importante para los equipos de desarrollo cuyo personal varía constantemente y en proyectos con una gran duración. A continuación se muestra la figura 4 donde se evidencia como RUP establece el proceso de desarrollo de *software*:

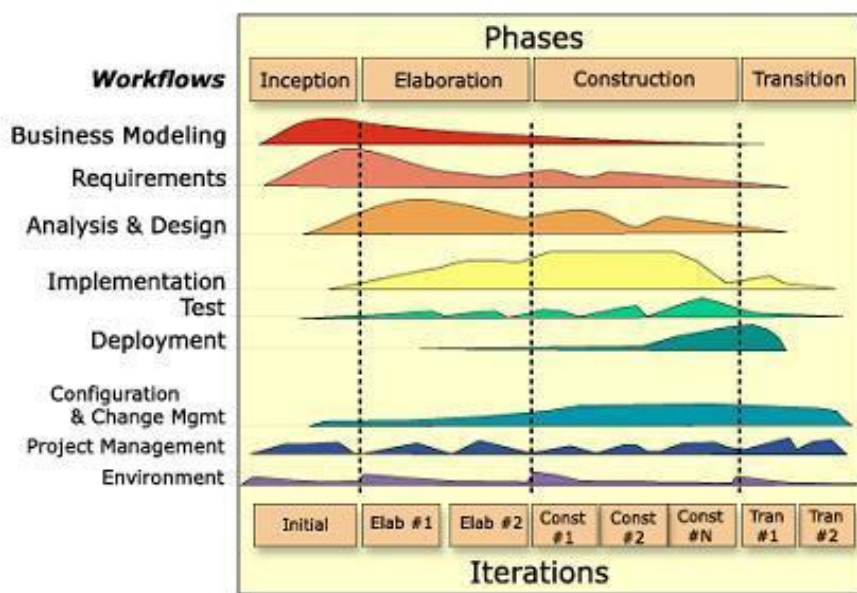


Figura 4: Fases e Iteraciones de la Metodología RUP.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

La figura 4 muestra como RUP organiza las tareas en 5 flujos de trabajos fundamentales: modelamiento del negocio, requisitos, análisis y diseño, implementación y prueba. Además de estos, cuenta con 4 flujos destinados al soporte que incluyen la configuración y administración de cambio, administración del proyecto, ambiente y distribución, los que se van desarrollando a lo largo de las 4 fases que esta metodología posee: inicio, elaboración, construcción y transición.

Esta metodología es la que se decide usar en el proyecto ya que brinda una gran cantidad de documentación, lo cual como bien se explica en las características es un elemento importante para aquel equipo de desarrollo cuyo personal es muy cambiante, como es el caso del proyecto SIC. Además de esto el cliente no forma parte del equipo de desarrollo e interesa por tanto generar la mayor documentación posible permitiendo una mayor flexibilidad y posibilidad de cambios.

Como herramienta CASE se decide usar en el proyecto Visual Paradigm, la cual está diseñada para ingenieros de *software*, analistas de sistemas, arquitectos de sistemas y otros interesados en el diseño de *software* orientado a objetos ya que soporta el ciclo de vida completo del desarrollo: análisis y diseño, implementación, pruebas y despliegue. Esta herramienta de modelado UML posibilita una rápida construcción de aplicaciones con características significativas que la hacen una de las herramientas más usadas ya que es multiplataforma, posibilita la generación de código y de documentación, además permite crear todos los tipos de diagramas de clases que se derivan del uso de la metodología RUP. Su diseño es centrado en casos de uso y usa un lenguaje estándar común para el equipo de desarrollo.

1.7 Descripción de las herramientas.

La identificación de las herramientas y tecnologías adecuadas en el proceso de desarrollo de *software* es una de las acciones a tener en cuenta para minimizar su complejidad y lograr productos con calidad, que sean rápidos, fiables, fáciles de usar, legibles, modulares y estructurados. Debido a que la solución de la aplicación está prevista para entornos web y bajo el paradigma del *software* libre, se decide para el proyecto SIC usar Groovy y Java como lenguajes de programación haciendo uso de los *frameworks* de desarrollo Grails y Google Web Toolkit, PostgreSQL como gestor de base de datos y Visual Paradigm como herramienta CASE.

1.7.1 Java y Groovy.

Java es un lenguaje de programación de uso general, su sintaxis y semántica son suficientemente completas para poder abarcar programas de todo tipo. Es una de las plataformas que cuenta con mayor acogida para la construcción de aplicaciones multicapas. Su Arquitectura J2EE ofrece conceptos básicos de componentes, herramientas y ambientes de desarrollo que permiten generar programas o páginas dinámicas desplegadas para los distintos ámbitos de la programación. Su principal característica es la capacidad de ejecutarse en cualquier máquina y sobre cualquier sistema operativo o arquitectura, manteniendo las facilidades básicas del lenguaje. (10)

Este lenguaje permite fácilmente el desarrollo tanto de arquitecturas cliente-servidor como de aplicaciones distribuidas, consistentes en crear aplicaciones capaces de conectarse a varios ordenadores y ejecutar tareas en estos simultáneamente. Aunque también otros lenguajes de programación permiten crear aplicaciones de este tipo, Java incorpora en su propia API estas funcionalidades.

En el ámbito web, Java da la capacidad de desplazar el control de la interactividad de los servidores hacia las máquinas de los usuarios, permitiendo la utilización de distintos componentes que son interpretados por los navegadores web y en los casos correspondientes por la Máquina Virtual de Java (JVM, por sus siglas en inglés). Otra característica propia es el acceso a herramientas de desarrollo de licencia libre que permite a los usuarios un fácil acceso a esta tecnología. Ejemplo de ello son los *softwares* de reporte que permiten la formación, integración y explotación de informes de forma rápida y sencilla, brindando soluciones efectivas y elegantes de forma simple para los usuarios de todos los niveles.

Groovy es un lenguaje de programación orientado a objetos que puede usarse como lenguaje de *scripting* dentro de la plataforma Java. Cuenta con la Groovy Development Kit (GDK) que se extiende de la API de Java con la adición de nuevos métodos y funciones que le brinda una mayor robustez. El *bytecode* generado en el proceso de compilación es totalmente compatible con la JVM, por lo que puede usarse directamente en cualquier aplicación de este tipo. La mayor parte de código escrito en Java es totalmente válido en Groovy lo que permite mezclar y combinar estos con el mínimo esfuerzo haciendo uso de las potencialidades de ambos. (11)

El principal aporte de Groovy es que reduce grandemente el tiempo de desarrollo de *software*

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

ya que su sintaxis elimina la complejidad presente en Java. Además de todo lo mencionado anteriormente cuenta con el potente *framework* Grails para el desarrollo de aplicaciones web.

1.7.2 Grails y Google Web Toolkit (GWT).

Grails es un novedoso *framework* web para la plataforma Java que se basa en el lenguaje dinámico Groovy. Con el objetivo de disminuir el tiempo de desarrollo y reutilizar los mejores componentes arquitectónicos de la plataforma J2EE, Grails hace uso de la estrategia Conventions Over Configurations (Convención sobre la Configuración) brindando la opción de seguir un conjunto de pasos definidos para establecer la arquitectura del sistema. La composición arquitectónica está dividida en tres capas: la capa web, contiene la lógica de presentación; la capa de negocio, la cual controla la lógica de negocio mediante una capa intermedia; la capa de persistencia, que opera dentro de los bordes de la capa de negocio y es controlada también por una capa intermedia para el manejo de la persistencia. (11)

Posee una arquitectura orientada a *plug-ins* donde la filosofía es la utilización del componente más capaz para la realización de las tareas, la figura 5 muestra su arquitectura:

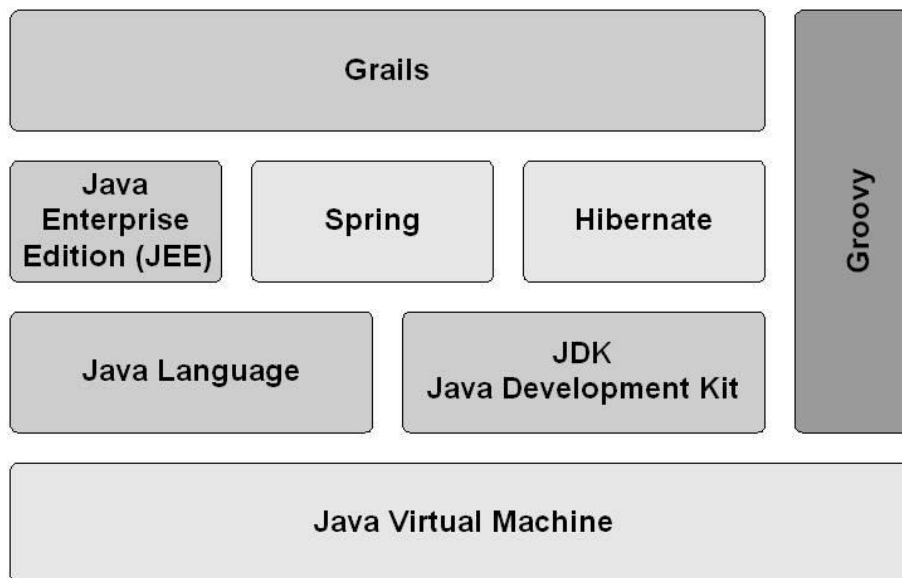


Figura 5: *Arquitectura de Grails.*

La figura 5 presenta como la arquitectura de Grails está basada sobre fuertes pilares como Spring para los flujos de trabajo e inyección de dependencias, Hibernate para la persistencia y para la creación de propiedades y métodos dinámicos en los objetos de la aplicación los lenguajes Java y Groovy. Además de estos cuenta con Sitemesh para la composición de las

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

vistas y Apache Ant para la gestión del proceso de desarrollo.

Grails ofrece un camino de migración desde entornos Java menos productivos monitorizándose y depurándose con potentes herramientas. Permite la creación de aplicaciones más rápidas en comparación con el típico *framework* de Java ya que se necesita menos código para el desarrollo de estas. Mediante el uso de Lenguajes de Dominio Específicos (DSL, por sus siglas en inglés), la versatilidad de Groovy y un ecosistema de *plug-ins* que mejora la productividad en un conjunto cada vez más amplio de escenarios, este *framework* se ha hecho popular y constituye hoy un motor de cambio en el espacio Java.

Google Web Toolkit (GWT) es un *framework* de código abierto bajo la licencia Apache 2.0 que permite el desarrollo de complejas aplicaciones web basadas en AJAX usando el lenguaje de programación Java; compila y traduce todo el código Java en JavaScript y HTML compatible con cualquier navegador web. Dispone de una colección de componentes de interfaz de usuario (*widgets*) dinámicas y reusables, permite la administración del historial de navegación, la depuración de código en tiempo real con los avanzados sistemas de *debugging* y manipulación de excepciones incluidos en los Entornos de Desarrollo Integrado (IDE, por sus siglas en inglés) como Eclipse. (12)

GWT establece la comunicación con el servidor a través de la Invocación de Procedimientos Remotos (RPC, por sus siglas en inglés), estos permiten fácilmente al cliente enviar y recibir objetos de Java sobre el Protocolo de Transferencia de Hipertexto (HTTP, por sus siglas en inglés). Además, brinda la oportunidad de mover toda la lógica de la interfaz de usuario al cliente lo que mejora el funcionamiento de la aplicación, reduce el ancho de banda usado y la carga del servidor.

1.7.3 PostgreSQL.

PostgreSQL es un gestor de base de datos objeto relacional de código abierto, que como muchos proyectos de este tipo no es manejado por una sola compañía sino que es dirigido por una comunidad de desarrolladores y organizaciones comerciales las cuales trabajan en su desarrollo.

Es un sistema objeto relacional puesto que incluye características propias de la orientación a objetos, proporcionando soporte para los lenguajes muy populares como PHP, C, C++, Python y otros. Incorpora extensiones para datos espaciales y minería de datos así como funciones

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

para el manejo de fechas y reglas. Utiliza un modelo cliente-servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema aprovechando la potencia de sistemas multiprocesadores. PostgreSQL permite la gestión de diferentes usuarios y los permisos asignados a cada uno de estos contando con soporte para el Protocolo de Capa de Conexión Segura (SSL, por sus siglas en inglés).

1.8 Conclusiones parciales.

Los módulos de reportes existentes para sistemas confiables constituyen soluciones enmarcadas en negocios específicos que impiden su reutilización en el SIC.

El estudio de los reportes para análisis de confiabilidad y de las diferentes formas de realizar estos permitió identificar la estructura adecuada de los reportes a generar.

El estudio de las principales herramientas que permiten la generación de reportes posibilitó la selección del motor JasperReports como el más adecuado para la realización de los reportes.

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Capítulo 2 Características del Sistema.

2.1 Introducción.

En el presente capítulo se adquiere una visión práctica del módulo a desarrollar a partir de la descripción del negocio en cuestión. En el mismo, se expondrán los requisitos funcionales y no funcionales previstos para dar una correcta solución a las necesidades de los clientes. Además, se determinarán los casos de uso correspondientes y se describirán los procesos de las principales funcionalidades del sistema.

2.2 Especificación de los reportes a realizar.

Los tipos de reportes previstos se encuentran divididos por categorías, a continuación se muestra la tabla 1 con la descripción de cada una de estas y sus reportes correspondientes.

Tabla 1: *Tipos de reporte por categorías.*

Categorías	Tipos de Reportes	Descripción
Modo de falla, falla funcional y función.	Modos de Fallas.	Se realiza un reporte en función de la cantidad de funciones, modos de fallas o fallas funcionales de un grupo de equipos según el tipo de reporte.
	Fallas Funcionales.	
	Funciones.	
Actividad de Mantenimiento.	Actividades de Mantenimiento.	Se realiza el reporte por la cantidad de actividades de mantenimiento que presenta cada equipo de un grupo de equipos.
Estrategia de Mantenimiento.	Estrategias de Mantenimiento.	Se realiza el reporte por la cantidad de estrategias de mantenimiento que presenta cada equipo de un grupo de equipos.

La tabla 1 muestra los diferentes tipos de reportes que se deben realizar con la implementación del presente módulo, cada reporte pertenece a una categoría específica y en función de esta se realizará el diseño del mismo.

2.3 Comprensión del contexto del sistema.

El resultado final de un proyecto de *software* es un producto que toma forma a lo largo del desarrollo del proyecto. La calidad del producto final está estrechamente ligada a la calidad de

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

su proceso de desarrollo entre otros aspectos.

Cualquier metodología de análisis y diseño para el desarrollo de sistemas tiene como punto de partida la captura de requisitos obtenidos por los analistas en interacción con los usuarios, que más tarde serán analizados y plasmados en herramientas propias de cada metodología, de manera que cubran las expectativas de los usuarios y que se ajusten a las tendencias actuales de desarrollo de aplicaciones.

La obtención de requerimientos es un paso muy importante para el posterior desarrollo de las siguientes etapas, pues un error en estas fases iniciales puede dar al traste con un sistema que no cumpla las expectativas de los usuarios y difícilmente aporte valor agregado al negocio para el que debe ser concebido. Por todo esto se hace necesaria la obtención por parte de los analistas de un modelado del negocio y modelado conceptual que se ajuste a las necesidades de los clientes.

2.3.1 Modelo de Dominio.

El modelado del dominio es una variante simplificada del modelado del negocio, el cual solo se centra en las “cosas”, es decir, en las clases del dominio o entidades del negocio que necesitan usar los trabajadores. Este se describe mediante diagramas de UML (especialmente mediante diagramas de clases) representando clases conceptuales del mundo real, no componentes de *software*. Las clases del dominio aparecen en tres formas típicas: (13)

- Objetos del negocio que representan cosas que se manipulan en este.
- Objetos del mundo real y conceptos de los que el sistema debe hacer un seguimiento.
- Sucesos que ocurrirán o han ocurrido.

Estas clases se usan al diseñar la interfaz de usuario y para sugerir clases internas al sistema en desarrollo durante el análisis.

Debido al poco conocimiento sobre el negocio de la realización de los reportes para el SIC y las experiencias adquiridas en otros proyectos en el desarrollo de módulos similares se decide realizar un modelado de dominio. La figura 6 que se muestra a continuación representa el modelo de dominio de la solución propuesta.

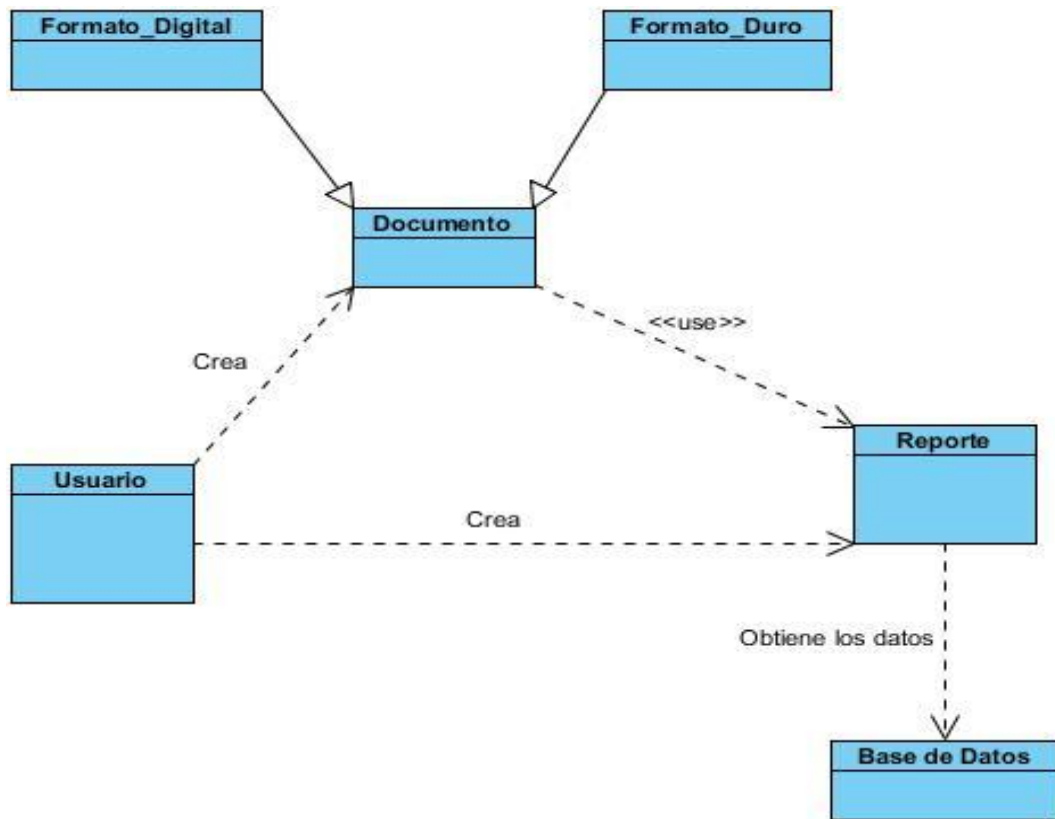


Figura 6: Modelo de dominio.

En la figura 6 se representan a través de clases conceptuales los principales conceptos que serán manejados para alcanzar los objetivos trazados para este proyecto. Estas son:

Usuario: Persona autorizada a realizar reportes en la institución.

Documento: Archivo creado por el usuario que contendrá el reporte.

Formato_Duro: Documento creado por el usuario con el objetivo de ser impreso.

Formato_Digital: Documento creado por el usuario con el objetivo de ser guardado en el disco duro o mostrado por pantalla.

Reporte: Informe creado para los análisis de confiabilidad, disponibilidad y mantenibilidad.

Base de Datos: Fuentes de datos utilizada por el reporte para dar valor a sus campos.

2.3.2 Especificación de los requisitos.

Requisitos funcionales.

Los requisitos funcionales son características que el sistema debe tener, describen la interacción entre el sistema y su ambiente independientemente de su implementación, el ambiente incluye al usuario y cualquier otro sistema externo que interactúa con el sistema.

El módulo de reportes para el SIC debe ser capaz de:

RF1: Generar informes parametrizables con gráficos de barra y de pastel.

Este requisito se refiere a la parametrización del informe por parte del usuario respecto a los datos que se quieran obtener de la fuente de datos y su forma de representación.

RF 2: Exportar los informes a PDF y HTML.

Este requisito viene dado por la necesidad mostrar y archivar los datos obtenidos en formatos digitales estándares.

RF 3: Imprimir los informes generados.

Este requisito expresa la necesidad de tener los informes en formato duro y no solo digital.

Requisitos no funcionales.

Los requerimientos no funcionales constituyen una restricción que el sistema debe satisfacer para ser aceptada por el cliente, describen aspectos del sistema que son visibles por el usuario que no incluyen una relación directa con el comportamiento funcional del sistema. Incluyen restricciones como el tiempo de respuesta (desempeño), la precisión, recursos consumidos, seguridad, entre otros.

El módulo de reportes para el SIC debe satisfacer los RnF que se presentan a continuación:

Requisitos de Software.

RnF 1:

- Máquina Virtual de Java.
- Base de datos PostgreSQL.

- Navegador Web.

Requisitos de Hardware.

RnF 2:

- Memoria RAM 1Giga o superior.
- Microprocesador de 2.0 GHz o superior.

Requisitos de diseño e implementación.

RnF 3: Empleo del mismo conjunto de herramientas que utilizarán los demás módulos del proyecto, garantizando la escalabilidad y compatibilidad con estos.

RnF 4: Licencia bajo código abierto: el sistema debe cumplir con los lineamientos necesarios para la producción de *software* libre.

RnF 5: Sistema susceptibles a ampliaciones: el módulo debe ser diseñado e implementado lo más sencillo posible, aplicando para su desarrollo las metodologías y estándares que para ello sean necesarios.

RnF 6: Sistema multiplataforma: el módulo debe ejecutarse en diferentes plataformas de hardware y *software*, es decir, debe ser portable a varios sistemas operativos como Windows y Linux.

Requisitos de usabilidad.

RnF 7: La herramienta debe poseer un procedimiento sencillo de uso, garantizando que quienes la utilicen se sientan familiarizados y cómodos con la misma.

2.4 Descripción de los actores del sistema.

Un actor del sistema es un conjunto de roles que los usuarios de casos de uso desempeñan cuando interactúan con estos, representan cualquier elemento externo que colabora con el sistema. Cuando se ha logrado determinar los actores del sistema se tiene identificado el entorno externo de este, la tabla 2 que se presenta a continuación muestra el actor del sistema del módulo en cuestión:

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Tabla 2: Descripción de los actores.

Actor	Descripción
Ingeniero de Confiabilidad	Es el encargado de realizar los reportes a partir del Repositorio de Información Unificada (RIU).

La tabla 2 muestra como se define un solo actor del sistema para interactuar con el mismo, siendo los ingenieros de confiabilidad los representantes de este actor y encargados de la realización de los reportes.

2.5 Casos de uso del sistema.

El objetivo principal de los requisitos es desarrollar un modelo del sistema que se va a crear, la utilización de los casos de uso es la forma adecuada para crear este modelo. Esto es debido a que los requisitos funcionales se estructuran de forma natural mediante casos de uso, y a que la mayoría de los otros requisitos no funcionales son específicos de un solo caso de uso.

Cada forma en que los actores usan el sistema se representa mediante estos, los cuales son “fragmentos” de funcionalidad que el sistema ofrece para aportar un resultado de valor para sus actores. Un caso de uso especifica una secuencia de acciones que el sistema puede llevar a cabo interactuando con sus actores, incluyendo alternativas dentro de la secuencia.

2.5.1 Diagrama de casos de uso del sistema.

El diagrama de casos de uso del sistema muestra el comportamiento del sistema a partir de la comunicación de los actores con los casos de uso del sistema. La figura 7 mostrada a continuación representa el diagrama de casos de uso del sistema para el módulo de reportes.

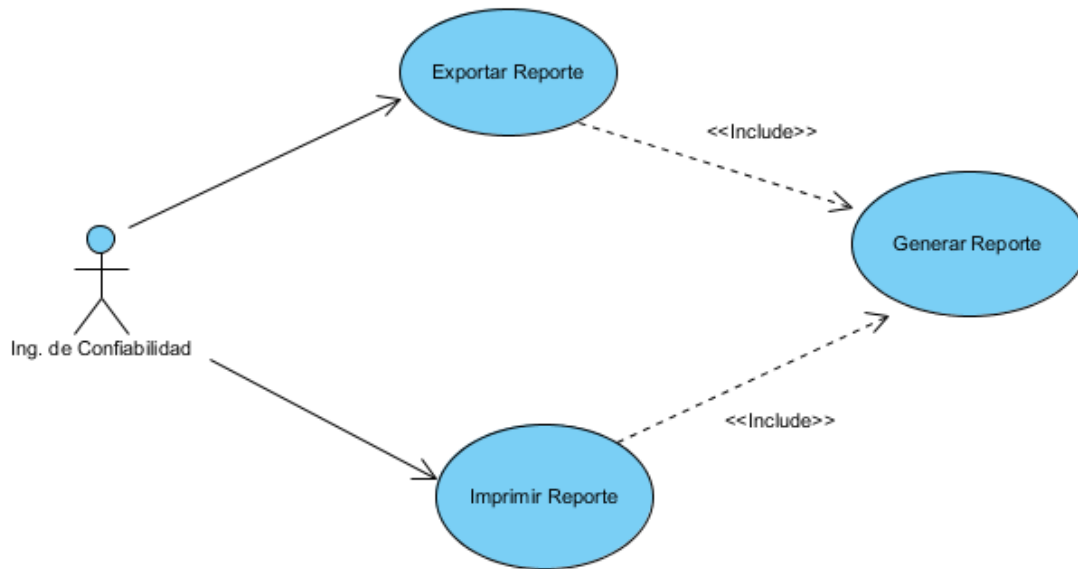


Figura 7: Diagrama de casos de uso del sistema.

En la figura 7 se muestra como el ingeniero de confiabilidad interactúa con el sistema a partir de los casos de uso “Exportar Reporte” e “Imprimir Reporte” para la realización de las funcionalidades establecidas. El caso de uso “Exportar Reporte” se encarga de dar un formato estándar escogido por el ingeniero de confiabilidad al reporte realizado, mientras que el caso de uso “Imprimir Reporte” es el encargado de solicitar al sistema operativo los recursos necesarios para imprimir el reporte. El caso de uso “Generar Reporte” es inicializado por los casos de uso mencionados anteriormente y se encarga de generar el reporte con los datos necesarios, para ello se realizan una serie de pasos que serán explicados más adelante.

2.5.2 Descripción textual de los casos de uso.

La descripción de los casos de uso tiene como objetivo principal detallar cada uno de estos describiendo su flujo de sucesos en detalles, incluyendo como comienzan, terminan e interactúan con los actores. Las tablas 3 y 4 siguientes muestran la descripción de los casos de uso “Exportar Reportes” e “Imprimir Reportes” respectivamente, los cuales tienen implícito el caso de uso “Generar Reportes”.

Tabla 3: Descripción del caso de uso “Exportar Reporte”.

Objetivo	El actor tiene como objetivo exportar los reportes generados.
Actores	Ingeniero de confiabilidad

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

Resumen	Este caso de uso le permite al actor generar el reporte y exportar el mismo al formato deseado.	
Complejidad	Alta.	
Prioridad	Crítico.	
Precondiciones	El ingeniero de confiabilidad debe estar autenticado. Debe existir conexión con la base de datos.	
Postcondiciones	Si el caso de uso finalizó correctamente se logró parametrizar, generar y exportar el reporte, en caso contrario se muestran los errores correspondientes.	
Flujo de eventos		
Flujo básico		
Actor	Sistema	
1. Selecciona del menú la opción Reportes.	2. Muestra un conjunto de categorías en las que se clasifican los reportes.	
3. Selecciona una de las categorías de reportes disponibles.	4. Muestra un grupo de campos necesarios para crear y configurar el reporte para la categoría escogida así como las opciones de exportar o imprimir el reporte.	
5. Selecciona los campos necesarios así como el formato a exportar.	6. Parsea el archivo de configuración correspondiente al reporte seleccionado. (Ver flujo alternativo No 1). 7. Valida la conexión con la fuente de datos (Ver flujo alternativo 2). 8. Parametriza el reporte configurado por el actor. 9. Construye el reporte con los valores obtenidos de la fuente de datos (Ver flujo alternativo 3). 10. Muestra el reporte en el formato especificado. 11. Termina el caso de uso.	
Flujos alternos		
Nº 1. Campos vacíos		
Actor	Sistema	
	1. Muestra una notificación señalando que faltan datos por introducir.	
2. Retorna al paso 5.		
Nº 2. Error de conexión con la fuente de datos		
Actor	Sistema	

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

		1. Muestra un mensaje de error en una ventana predeterminada por el sistema, indicándole al actor "Error al conectarse con la fuente de datos".
2. Presiona el botón OK.		3. Retorna al flujo básico.
Nº 3. Fuente de datos sin valores		
Actor		Sistema
		1. Muestra un mensaje en una ventana de error predeterminada por el sistema "Error al intentar construir el reporte, fuente de datos sin valores a mostrar".
		2. Vuelve al flujo básico.
Relaciones	CU Incluidos	Generar Reporte
	CU Extendidos	
Requisitos funcionales	RF 1, RF2	

Tabla 4: Descripción del caso de uso "Imprimir Reporte".

Objetivo	El actor tiene como objetivo imprimir los reportes generados.
Actores	Ingeniero de confiabilidad
Resumen	Este caso de uso le permite al actor generar el reporte e imprimir el mismo.
Complejidad	Alta.
Prioridad	Crítico.
Precondiciones	El ingeniero de confiabilidad debe estar autenticado. Debe existir conexión con la base de datos. Debe existir conexión con la impresora.
Postcondiciones	Si el caso de uso finalizó correctamente se logró parametrizar, generar e imprimir el reporte, en caso contrario se muestran los errores correspondientes.
Flujo de eventos	
Flujo básico	
Actor	Sistema

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

1. Selecciona del menú la opción Reportes.	2. Muestra un conjunto de categorías en las que se clasifican los reportes.
3. Selecciona una de las categorías de reportes disponibles.	4. Muestra un grupo de campos necesarios para crear y configurar el reporte para la categoría escogida así como las opciones de exportar o imprimir el reporte.
5. Selecciona los campos necesarios así como la opción de imprimir el reporte.	6. Parsea el archivo de configuración correspondiente al reporte seleccionado. (Ver flujo alternativo No 1). 7. Parametriza el reporte configurado por el actor. 8. Valida la conexión con la fuente de datos (Ver flujo alternativo 2). 9. Construye el reporte con los valores obtenidos de la fuente de datos (Ver flujo alternativo 3). 10. Envía el reporte a la impresora predeterminada por el Sistema Operativo. (Ver flujo alternativo 4). 11. Termina el caso de uso.
Flujos alternos	
Nº 1. Campos vacíos	
Actor	Sistema
	1. Muestra una notificación señalando que faltan datos por introducir.
2. Retorna al paso 5.	
Nº 2. Error de conexión con la fuente de datos	
Actor	Sistema
	1. Muestra un mensaje de error en una ventana predeterminada por el sistema, indicándole al actor "Error al conectarse con la fuente de datos".
2. Presiona el botón OK.	3. Retorna al flujo básico.
Nº 3. Fuente de datos sin valores	
Actor	Sistema
	1. Muestra un mensaje en una ventana de error predeterminada por el

CAPÍTULO 2: CARACTERÍSTICAS DEL SISTEMA

	sistema “Error al intentar construir el reporte, fuente de datos sin valores a mostrar”.	
	2. Vuelve al flujo básico.	
Nº 4. No hay impresora conectada a la máquina		
Actor		
Sistema		
	1. Muestra un mensaje en una ventana de error predeterminada por el sistema “No se encontró impresora conectada a la computadora”.	
	2. Vuelve al flujo básico.	
Relaciones	CU Incluidos	Generar Reporte
	CU Extendidos	
Requisitos funcionales	RF 1, RF3	

En las tablas 3 y 4 se puede observar el proceso establecido para exportar e imprimir un reporte respectivamente, este proceso está compuesto por flujos básicos y alternos que especifican la interacción entre el usuario y el sistema explicando el comportamiento del mismo.

2.6 Conclusiones parciales.

En este capítulo han sido sentadas las bases para el desarrollo del sistema a partir del entendimiento del proceso de negocio y la definición de los requerimientos funcionales y no funcionales del mismo.

La definición de los casos de uso y su consecuente descripción constituyen los primeros pasos para entender como el *software* funcionará sin entrar en detalles de diseño o implementación.

Capítulo 3 Análisis y Diseño del Sistema.

3.1 Introducción.

En el presente capítulo se hará una descripción de la arquitectura del sistema a partir del uso de patrones arquitectónicos y de diseño transmitiendo las decisiones arquitectónicas más importantes. Se expondrá además el diagrama de clases del análisis y diseño del sistema como representación de la estructura interna del mismo teniendo como base las técnicas de la programación orientada a objetos (POO).

3.2 Descripción de la arquitectura.

La aplicación de una arquitectura de *software* es el proceso de definición de una solución estructurada que cumple con todos los requisitos técnicos y operativos, además de la optimización de los atributos comunes de calidad tales como rendimiento, seguridad y manejabilidad. Se trata de una serie de decisiones basadas en una amplia gama de factores, y cada una de estas puede tener un impacto considerable en la calidad, rendimiento, mantenimiento, y el éxito global de la aplicación. (13)

En el libro *Software Architecture in Practice (2nd edition)*, Bass, Clements, y Kazman definen la arquitectura como:

“La estructura o las estructuras del sistema, que comprende elementos de *software*, las propiedades externamente visibles de esos elementos, y las relaciones entre ellos.”

La arquitectura de un *software* se define a partir de patrones arquitectónicos que posibilitan la solución del problema en disímiles escenarios de forma general. Estos patrones definen la estructura básica del sistema, pudiendo estar relacionado con otros patrones y representando una plantilla de construcción que provee un conjunto de subsistemas aportando las normas para su construcción. Dentro de los patrones arquitectónicos más utilizados en aplicaciones cliente-servidor para la gestión de la información a nivel empresarial tenemos:

- Modelo Vista Controlador (MVC).
- Arquitectura en capas.
- Arquitectura Orientada a Objetos.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

- Arquitectura Basada en Componentes.
- Arquitectura Orientada a Servicios.

Grails propone el patrón Modelo Vista Controlador (MVC) el cual facilita la funcionalidad, mantenibilidad y escalabilidad del sistema de forma sencilla dividiendo las aplicaciones en tres niveles de abstracción (Modelo, Vista y Controlador). La figura 8 que se muestra a continuación representa el patrón MVC.

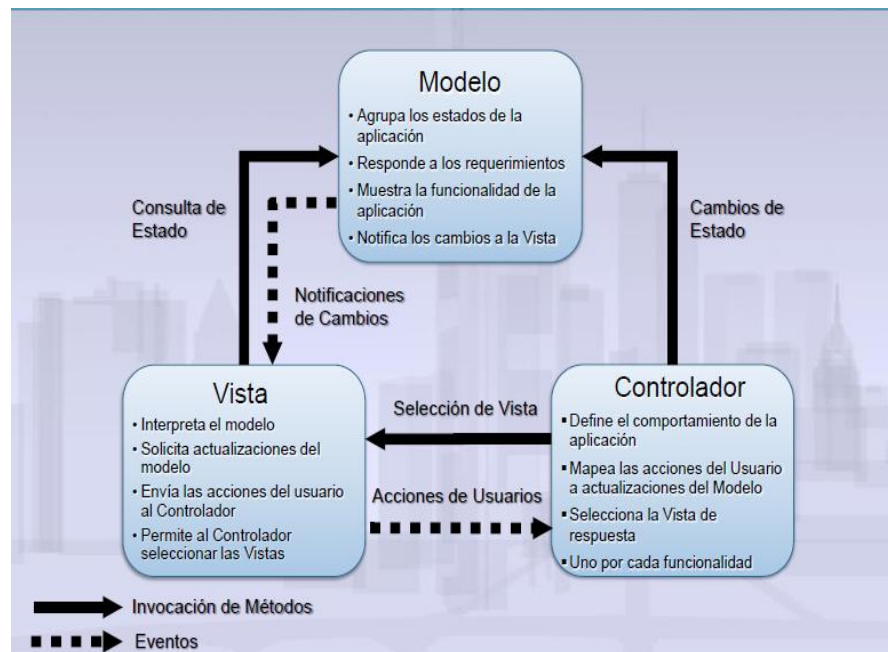


Figura 8: Patrón Modelo Vista Controlador (MVC).

En la figura 8 se muestra como se establece la relación entre los niveles de abstracción del patrón MVC. El modelo representa la lógica de negocios, es el encargado de acceder de forma directa a los datos actuando como intermediario con la base de datos, este es independiente de cualquier representación de salida o comportamiento de entrada. La vista es la encargada de mostrar la información al usuario de forma gráfica y legible, pueden existir múltiples vistas del modelo y cada una de estas debe tener asociado un controlador. El controlador es el intermediario entre la vista y el modelo, es quien controla las interacciones del usuario solicitando los datos al modelo y entregándolos a la vista para que esta lo presente al usuario. La forma en la que cada una de estas capas se estructura para cumplir su funcionalidad es independiente y hace uso de los patrones de diseños necesarios para ello.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

En la capa de las vistas se implementará el patrón de diseño Modelo Vista Presentación (MVP) bajo el *framework* Google Web Toolkit, este constituye una de las mejores soluciones de diseño, previendo una fácil integración, para aplicaciones de gran tamaño donde se trabaja de forma modular. A continuación se muestra la figura 9 que representa la estructura que propone el patrón MVP.

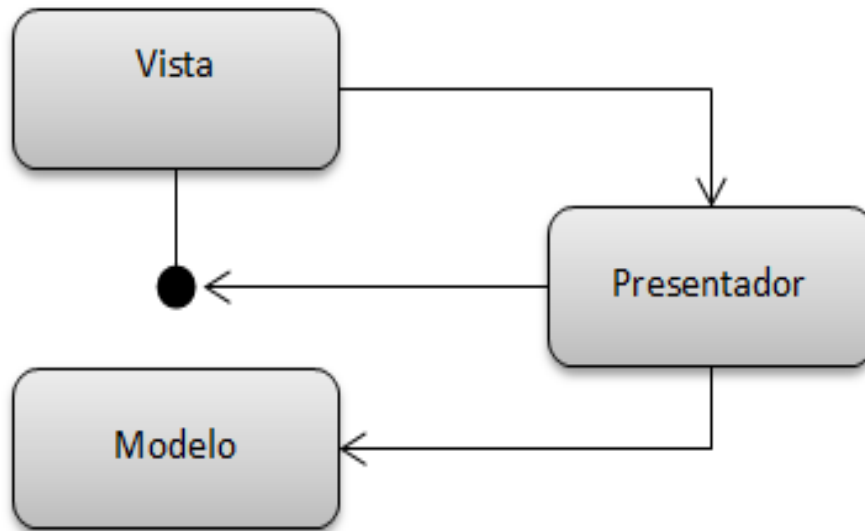


Figura 9: Modelo Vista Presentador (MVP).

El patrón Modelo Vista Presentación es un derivado del Modelo Vista Controlador destinado a proporcionar una limpia separación entre la vista, el modelo y el controlador como se muestra en la figura 9. El principal cambio radica en que la vista y el modelo están físicamente separados y no tienen conocimiento íntimo uno del otro. La vista se encarga de manejar los aspectos visuales manteniendo una referencia a su presentador al cual le delega la responsabilidad del manejo de eventos. El presentador contiene la lógica para responder a los eventos utilizando el modelo para responder a estos y es el responsable de manipular el estado de las vistas mediante una interfaz de referencia a estas. El modelo está compuesto por los objetos que conocen y manejan los datos dentro de la aplicación representando las clases que conforman el modelo de negocio.

En aplicaciones desarrolladas bajo el *framework* GWT el patrón MVP es uno de los más utilizados por los desarrolladores para la creación de aplicaciones eficientes, amigables y dinámicas en plataformas como Microsoft.NET, Android y Windows. Si bien existen varios patrones arquitectónicos para elegir, el MVP funciona mejor para aplicaciones desarrolladas

sobre GWT, este permite desacoplar el desarrollo de manera que varios desarrolladores puedan trabajar de forma simultánea y minimiza el uso de los casos de prueba para GWT (GWTTTestCase), que se basan en la presencia de un navegador. El principal beneficio que aporta este patrón es la separación de las funcionalidades en componentes que lógicamente tienen sentido, en este caso GWT tiene un enfoque claro y es que las vistas sean lo más simples posible para minimizar las dependencias de GWTTTestCase y reducir el tiempo en las pruebas de funcionamiento.

3.3 Modelo de análisis.

El modelo de análisis ayuda a refinar la arquitectura y permite razonar sobre los aspectos internos del sistema, refinando y estructurando los requerimientos que se describieron en la captura de requisitos para una mejor comprensión, preparación y mantenimiento de estos. No tiene en cuenta aquellos elementos relacionados con el diseño, pues no toma en consideración las tecnologías a utilizar en el momento del desarrollo.

Los diagramas de clases del análisis constituyen artefactos del modelo de análisis, donde las clases y sus objetos participan en varias realizaciones de casos de uso centrándose en el tratamiento de los requisitos funcionales y posponiendo los no funcionales para el diseño e implementación. Los diagramas de clases pertenecientes a los casos de uso “Exportar Reporte” e “Imprimir Reporte” así como los restantes diagramas (secuencia y colaboración) que de estos se devienen se pueden consultar en los [Anexos](#).

3.4 Diseño del Sistema.

La función del sistema es generar reportes a la medida para la correcta toma de decisiones por parte de los ingenieros de confiabilidad. Para lograr este objetivo el diseño del sistema se dividió en cuatro paquetes para un análisis más detallado basado en los patrones arquitectónicos Modelo Vista Controlador (MVC) y Modelo Vista Presentador (MVP) a utilizar en la presente aplicación. A continuación se presenta la figura 10 en la cual se evidencia la estructuración y relación de estos paquetes.

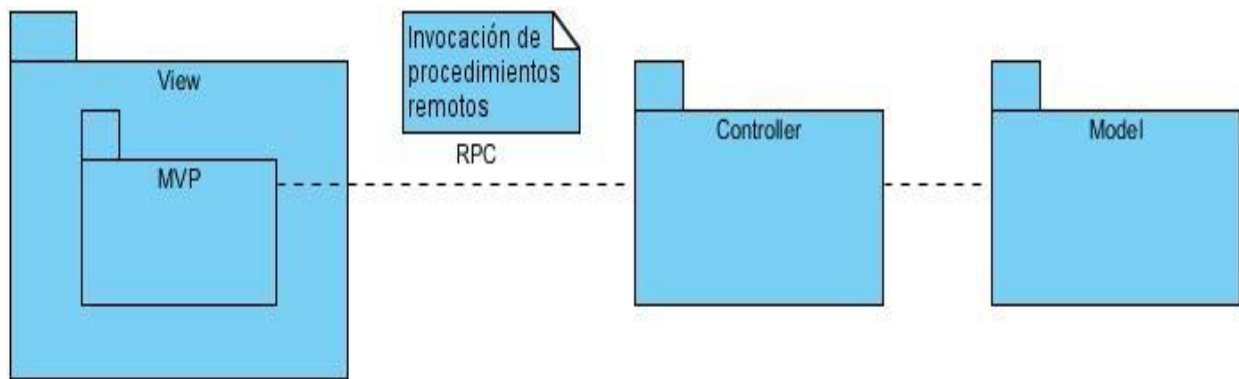


Figura 10: Representación de los paquetes de diseño.

La estructura de los paquetes presentes en la figura 10 se expondrá en los epígrafes siguientes para un mejor entendimiento del funcionamiento de cada uno de estos estableciendo siempre la relación existente entre ellos.

2.4.1 Diagrama de clases del paquete “View”.

El paquete “View” mostrado en la figura 10 contiene las páginas proporcionadas por Grails para presentar la información al usuario final y al paquete MVP, el que proporciona los datos a estas páginas en forma de script traduciendo todo el código Java. La figura 11 que se muestra a continuación constituye la estructuración interna de este paquete.

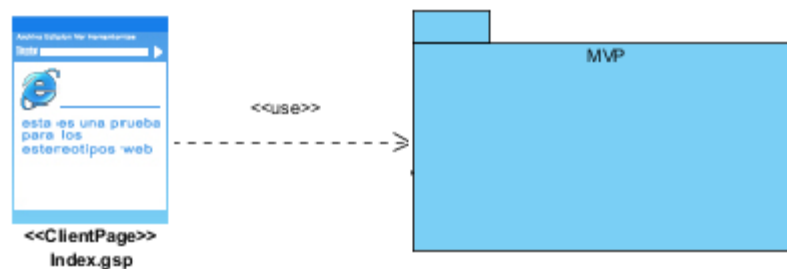


Figura 11: Representación del paquete de diseño “View”.

La figura 11 muestra como se produce la integración de la página proporcionada por el *framework* Grails y el paquete “MVP” creado con el *framework* Google Web Toolkit en el paquete “View”. El paquete “MVP” contiene las clases de Java que implementan el patrón Modelo Vista Presentador empleando el *framework* GWT. La figura 12 mostrada a continuación constituye la estructuración interna de este paquete.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

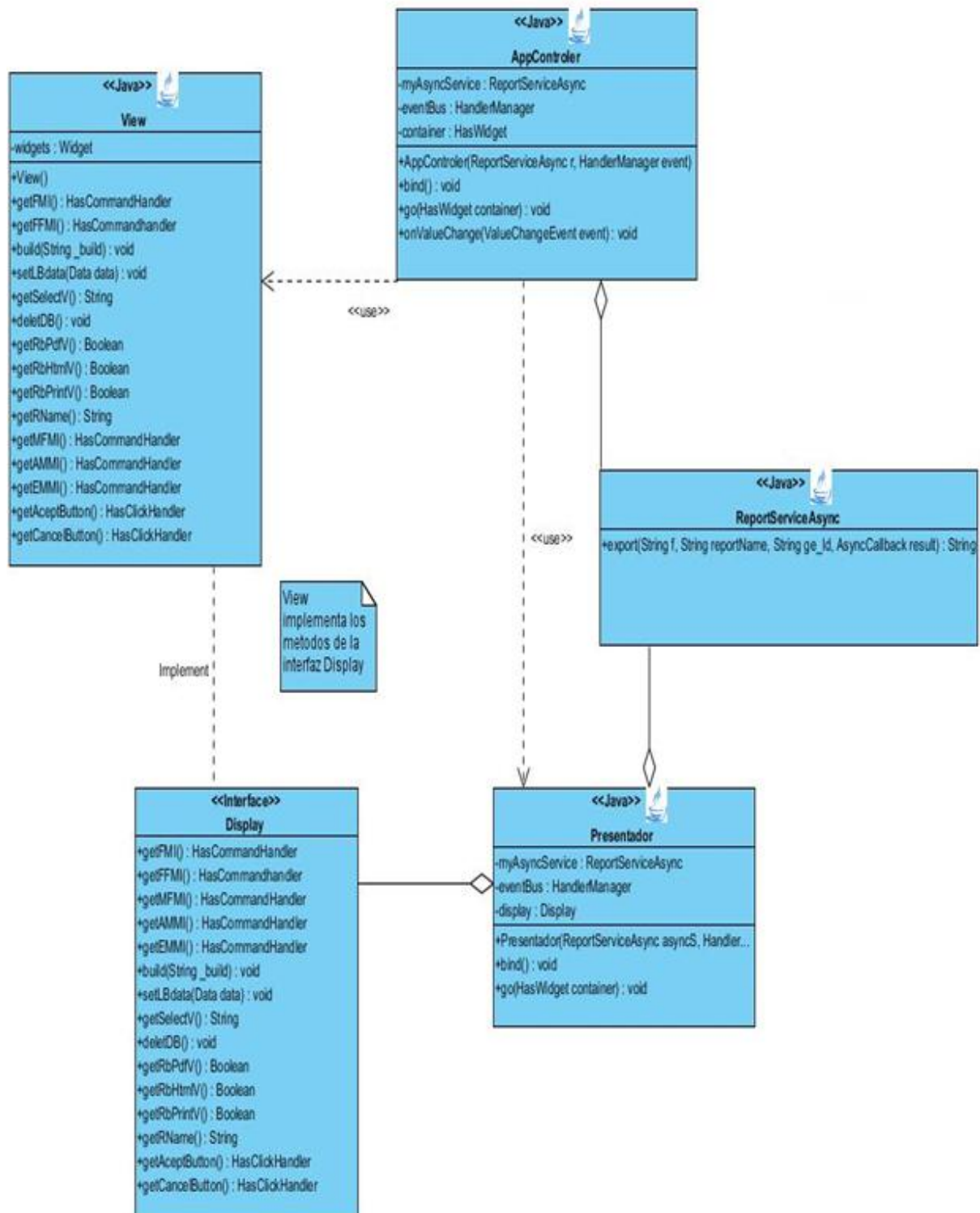


Figura 12: Diagrama de clases correspondiente al paquete "MVP".

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

En la tabla 5 se presenta la descripción de cada una de las clases representadas en la figura 12.

Tabla 5: Descripción de las clases del paquete “MVP”.

<i>Descripción de las clases del paquete “MVP”</i>	
<i>Nombre de la clase</i>	<i>Descripción</i>
View	Clase contenedora de todos los <i>widgets</i> proporcionados por el <i>framework</i> Google Web Toolkit.
Display	Interfaz implementada por la clase View para la captura de las solicitudes.
Presenter	Clase encargada de presentar los datos a la clase “View” y ejecutar las solicitudes a través de la interfaz Display y de una instancia de la clase ReportServiceAsync.
AppControler	Clase controladora de los eventos lanzados por la clase Presenter para el cambio de vistas.
ReportServiceAsync	Clase encargada de la comunicación entre el cliente y el servidor a través de la Invocación de Procedimientos Remotos (RPC).

La tabla 5 muestra las clases presentes en el paquete “MVP” las cuales se encargan del manejo de la interfaz de la aplicación, se describe además el funcionamiento de cada una de estas para establecer la comunicación con el servidor y mostrar la información requerida.

2.4.2 Diagrama de clases del paquete “Controller”.

El paquete “Controller” mostrado en la figura 10 contiene las clases de tipo *service* que brinda Grails para manejar la lógica de la aplicación además de otras clases auxiliares para el manejo de archivos en el servidor. En este paquete se encuentra la clase que constituye el punto de comunicación con el cliente a partir de su exposición a este. La figura 13 mostrada a continuación constituye la estructuración interna de este paquete.

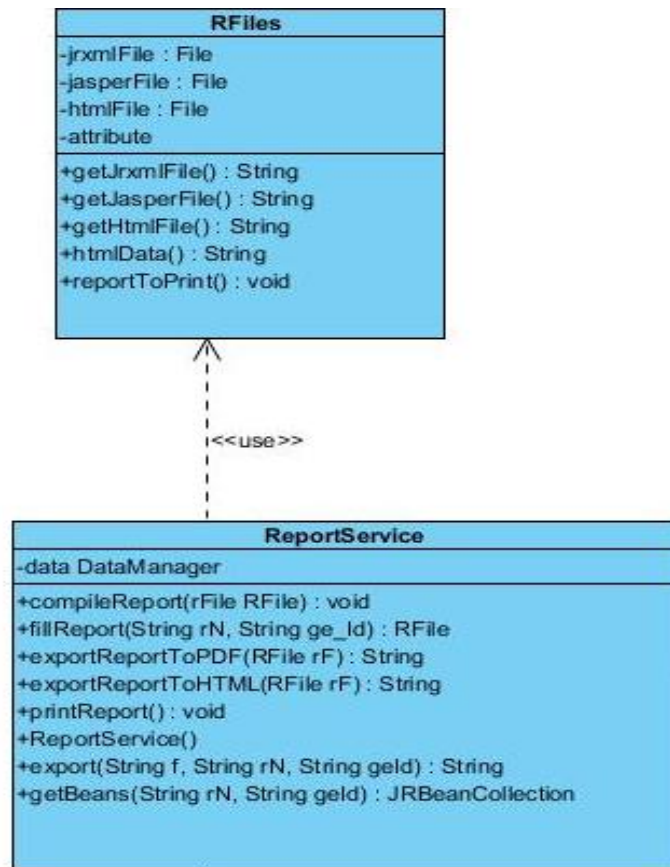


Figura 13: Diagrama de clases del paquete “Controller”.

En la tabla 6 se presenta la descripción de cada una de las clases representadas en la figura 13.

Tabla 6: Descripción de clases del paquete “Controller”.

Descripción de las clases del paquete “Controller”	
Nombre de la clase	Descripción
ReportService	Clase encargada de la realización de los reportes y punto de encuentro con la parte cliente a partir de su exposición a este.
RFile	Clase encargada del manejo de los archivos vinculados a los reportes y de los propios reportes.

La tabla 6 muestra las clases presentes en el paquete “Controller”, las cuales se encargan del manejo de la lógica de la aplicación, además describe el funcionamiento de cada una de estas clases.

2.4.3 Diagrama de clases del paquete “Model”.

El paquete “Model” mostrado en la figura 10 cuenta con las clases proporcionadas por Grails Object Relational Mapping (GORM) que permiten la comunicación con la base de datos y otras clases que permiten el manejo de estas a partir de herramientas del propio Object Relational Mapping (ORM). La figura 14 que se muestra a continuación constituye la estructuración interna de este paquete.

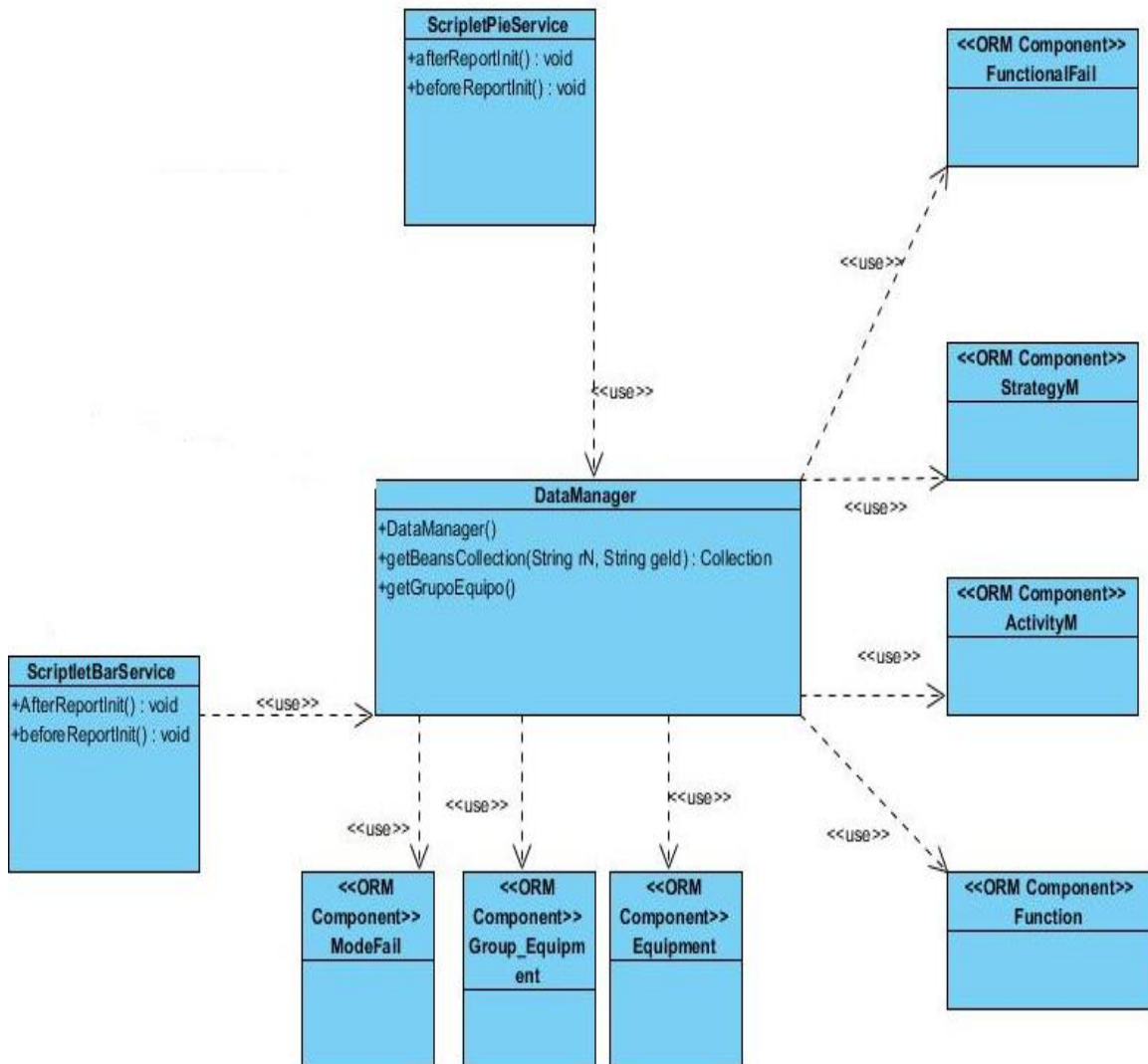


Figura 14: Diagrama de clases del paquete “Model”.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

La tabla 7 presenta la descripción de cada una de las clases representadas en la figura 14.

Tabla 7: Descripción de clases del paquete “Model”.

Descripción de las clases del paquete “Model”	
Nombre de la clase	Descripción
DataManager	Clase encargada de la obtención de datos a partir de la herramienta Hibernate Query Language (HQL).
ScriptletBarService	Clase encargada de introducir y extraer datos al reporte para crear los gráficos de barras.
ScriptletPieService	Clase encargada de introducir y extraer datos al reporte para crear los gráficos de pastel.
GroupEquipment	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen un grupo de equipo.
Equipment	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen un equipo.
FunctionalFail	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen una falla funcional de un equipo.
FailMode	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen un modo de falla de un equipo.
Function	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen una función de un equipo.
StrategyM	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen una estrategia de mantenimiento para un equipo.
ActivityM	Clase que proporciona el ORM de Grails para mapear las tablas de la base de datos. Contiene los elementos que componen una actividad de mantenimiento para un equipo.

CAPÍTULO 3: ANÁLISIS Y DISEÑO DEL SISTEMA

La tabla 7 muestra las clases presentes en el paquete “Model”, las cuales se encargan del acceso a los datos en la aplicación, además describe el funcionamiento de cada una de estas clases.

3.5 Modelo de Despliegue.

El modelo de despliegue tiene como objetivo recolectar toda la configuración de los elementos que intervienen en el despliegue del *software* y las conexiones entre estos elementos. Este modelo es utilizado por la mayoría de los integrantes de un equipo de desarrollo; los arquitectos de *software* lo usan para capturar y entender el entorno de ejecución física del sistema y su distribución, los diseñadores de *software* y de bases de datos para asimilar la distribución del proceso y los datos en el sistema y el gestor de proyectos en la estimación de costes. El diagrama de despliegue del presente módulo muestra la ubicación física y relación de los distintos nodos que componen el sistema como se presenta en la figura 15.

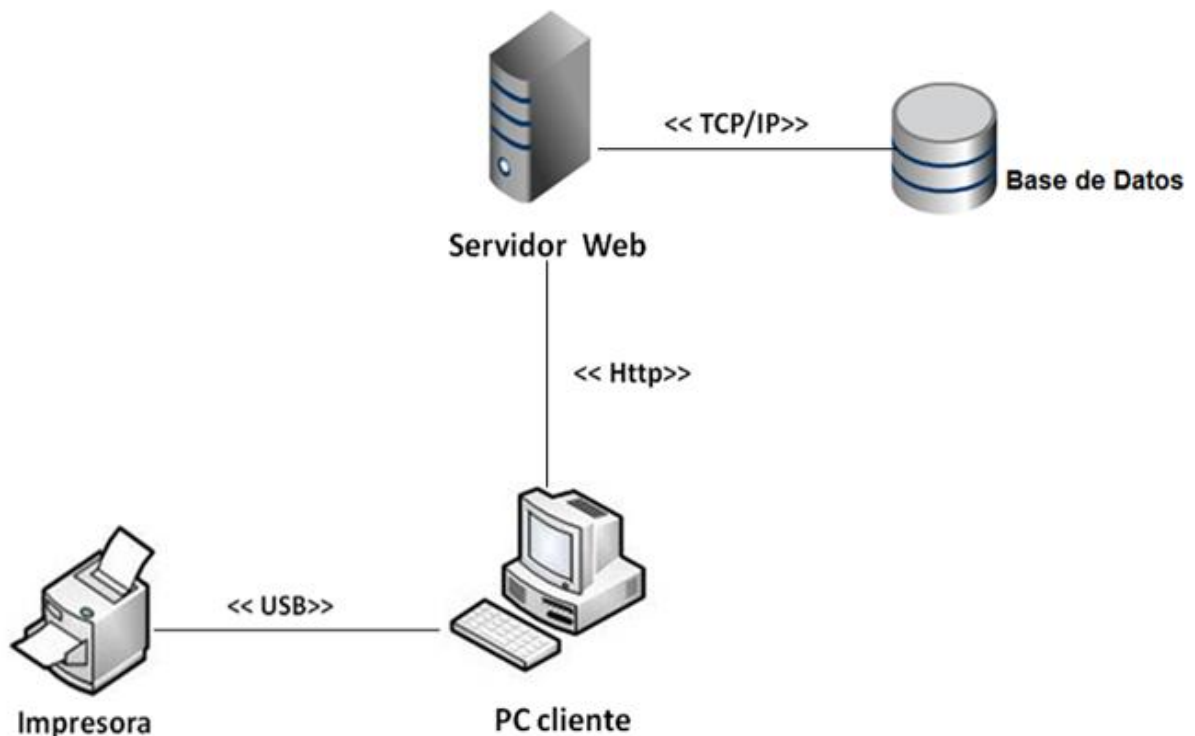


Figura 15: Diagrama de despliegue del módulo de reportes.

La figura 15 muestra la disposición de las instancias de componentes (Impresora, Servidor Web, Base de Datos y Computadora) de ejecución en instancias de nodos conectados por

enlaces de comunicación. En esta se describe la arquitectura física del sistema durante la ejecución en términos de procesadores, dispositivos y componentes de *software*.

3.6 Conclusiones parciales.

La estructuración por paquetes del módulo contribuye a una mejor comprensión del funcionamiento de los patrones arquitectónicos MVC y MVP definidos para el desarrollo de este.

Los patrones arquitectónicos utilizados proporcionan una perfecta integración entre los *frameworks* Grails y GWT, dotando al módulo de un alto grado de flexibilidad y escalabilidad.

La descripción de cada una de las futuras clases a implementar y sus relaciones son la base para una implementación rápida y eficiente.

Capítulo 4 Implementación y Prueba.

4.1 Introducción.

El presente capítulo completará el proceso de desarrollo del módulo en cuestión a partir de la implementación de las clases del diseño y la realización de las pruebas necesarias para la validación de la solución propuesta. En el mismo se definirá el estilo de la codificación y se explicarán elementos importantes relacionados con la generación de los reportes previendo el mantenimiento y utilización del sistema por terceros.

4.2 Implementación del Sistema.

4.2.1 Estilo de codificación.

Los estilos de codificación son convenciones o pautas que se establecen en las organizaciones para el desarrollo de productos informáticos. No se encuentran enfocados a la lógica del programa sino a la estructura y apariencia de este, evitando erróneas interpretaciones en el momento de dar mantenimiento al sistema y estableciendo como operar con la base de código existente.

Los siguientes estilos fueron definidos para el desarrollo del módulo de reportes teniendo en cuenta que el idioma a utilizar será el inglés:

Clases.

- Los nombres de las clases deben comenzar con mayúscula y el resto será en minúscula exceptuando los nombres compuestos, estos harán uso de mayúsculas en el carácter inicial de cada palabra. Ejemplos: ReportService, ScriptletBarService, Display, etc.
- Los nombres de los atributos de las clases e instancias de estas deben ser con minúscula, en caso de ser nombres compuestos el inicio de cada palabra compuesta será con mayúscula. Ejemplos: file, reportFile, reportFileName, etc.
- Las funciones deben llevar por nombre verbos que muestren la acción que se realiza y el formato coincide con el de los atributos. Ejemplos: getReportName, export, getFile, etc.
- Los nombres empleados para las clases, atributos, funciones y objetos deben estar

acordes con el propósito de los mismos sin excederse en el tamaño de estos.

Variables y constantes.

- Los nombres de las variables tendrán el mismo formato que los nombres de los atributos de clases.
- Los nombres de las variables booleanas deben comenzar con “is”. Ejemplo: isPieChar.
- Concatenar prefijos de cálculos (min, max, sum) a las variables que almacenan el producto de estos. Ejemplo: minValue.
- Los nombres de constantes deben ser en mayúscula, inicializados con la palabra “CONST” y separados por el carácter “_” (underscore). Ejemplo: CONST_REPORT_FIELD.
- Los nombres utilizados deben permitir que se reconozca el propósito de las variables o constantes declaradas.
- Emplear las variables con un solo propósito.

Documentación del código fuente (Comentarios).

- Comentar el código en el momento en que se está programando de forma explicativa paso por paso.
- Especificar el objetivo de las clases en la parte superior de su implementación.
- Documentar situaciones complejas que tributen a la lógica de la aplicación.
- Documentar cambios significativos en la aplicación con una breve justificación.

Tratamiento de errores.

- Los errores se pueden manejar mediante mecanismos de excepciones o retorno de valores, haciendo uso de uno solo de estos mecanismos en cada clase.
- La validación de campos tanto en el cliente como en el servidor debe apoyar este tratamiento de errores para que las rutinas puedan seguir su ejecución y el programa no

colapse.

Codificación.

- Alinear secciones de código y las llaves que abren y cierran estas secciones.
- Utilizar espacios antes y después de los operadores del lenguaje.
- Utilizar una sola sentencia de código por línea.
- Emplear constantes en lugar de números y cadenas de caracteres.
- Mantener el encapsulamiento evitando el uso de variables públicas.
- Emplear líneas en blanco para separar pasos lógicos y la organización del código.
- Evitar prácticas que incrementen la complejidad del sistema y mantener la modularidad de este.

4.2.2 Diagrama de Componentes.

El diagrama de componentes muestra como los elementos del modelo de diseño se implementan en términos de componentes como ficheros de código fuente y ejecutables. Además, se encarga de organizar los componentes de acuerdo con los mecanismos de estructuración y modularización disponibles en el entorno de implementación.

Este diagrama describe los elementos físicos del sistema y sus relaciones. Se utiliza para mostrar las dependencias de compilación de los ficheros de código, relaciones de derivación entre ficheros de código fuente y ficheros que son resultados de la compilación, dependencias entre elementos de implementación y los correspondientes elementos de diseños que son implementados. (13)

Teniendo en cuenta la arquitectura Modelo Vista Controlador que brinda soporte a la aplicación, se muestra en la figura 16 el diagrama de componentes perteneciente al módulo de reportes.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

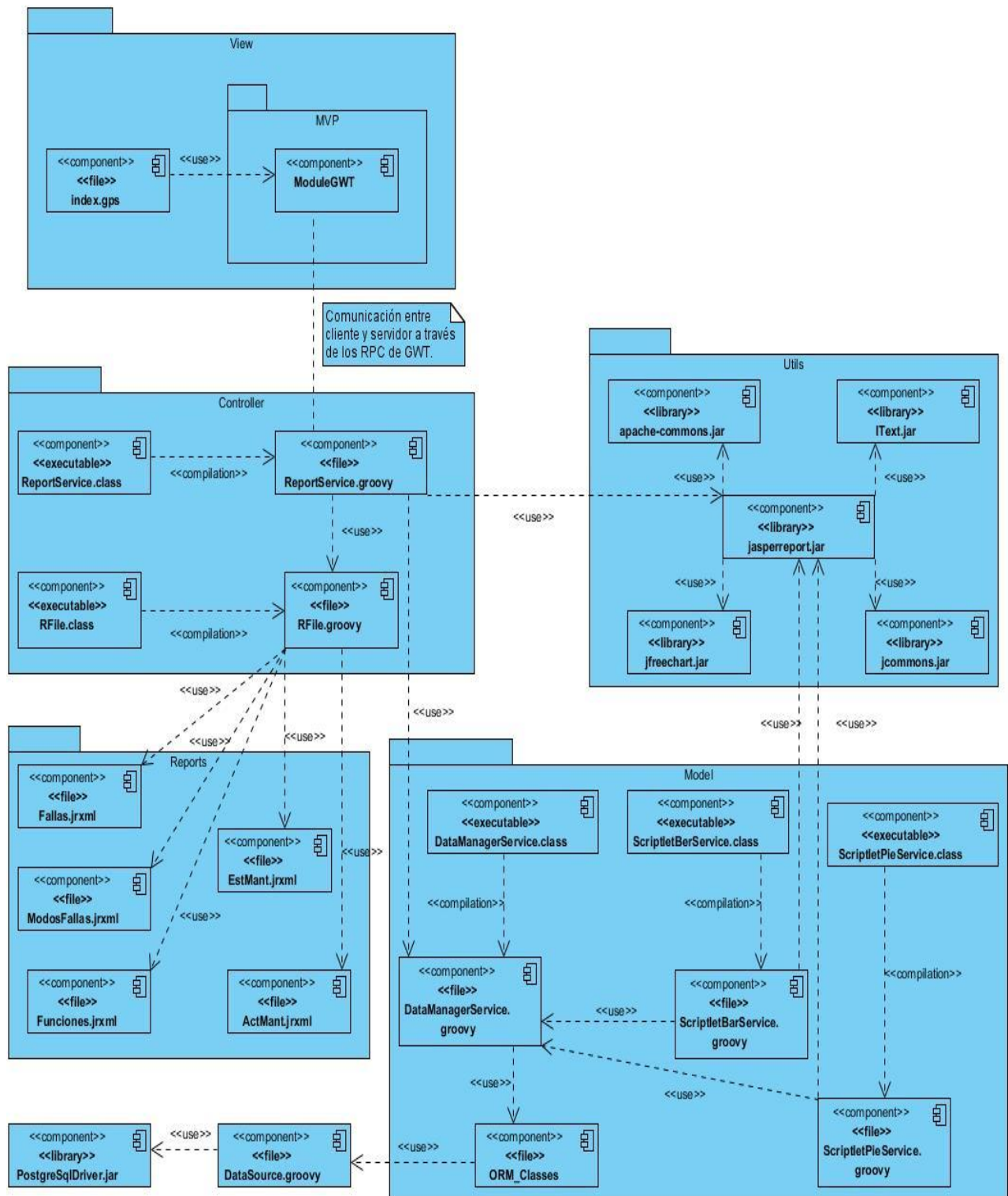


Figura 16: Diagrama de componentes del módulo de reportes.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

La figura 16 muestra una estructuración por paquetes de los componentes del sistema y sus relaciones. Los componentes mostrados con estereotipo <<executable>> representan el código fuente compilado en un ejecutable o binario que materializa las funcionalidades y responsabilidades del sistema, y los mostrados con estereotipo <<file>> representan los archivos de código fuente.

El paquete “Reports” contiene los archivos XML que utiliza la biblioteca JasperReports para la configuración de sus reportes, la estructura de estos archivos se analizará más adelante.

El paquete “Utils” contiene la biblioteca JasperReports usada para la generación de reportes, para ello se apoya en algunas bibliotecas que proporciona Apache Commons como Apache Commons Digester, Apache Commons Collections, Apache Commons Loggings y Apache Commons BeanUtils. Estas bibliotecas proporcionan una serie de funcionalidades de uso común para complementar y aumentar el marco de la interfaz Collection de Java, la inicialización de objetos Java desde un XML y el envío de datos a los archivos de registro, aspectos necesarios para la generación de los reportes. IText es una biblioteca de código abierto que le propicia la creación y manipulación de los archivos PDF por lo que se hace necesaria para el sistema que tiene como uno de sus requisitos exportar los reportes a este formato. Para el manejo de las gráficas se utilizó JfreeChart, biblioteca de fácil integración con JasperReports que permite la creación de gráficos con aspecto profesional.

El componente “DataSource.groovy” es el archivo de configuración que proporciona Grails para la conexión con la base de datos a través del manejador JDBC a utilizar (componente “PostgreSQLDriver.jar”).

4.2.3 Funcionamiento de JasperReports.

JasperReports trabaja en forma similar a compiladores e intérpretes. A partir de un diseño del reporte codificado en un XML de acuerdo con las etiquetas y atributos definidos por jasperreports.dtd (parte de JasperReports), se define completamente el reporte describiendo donde colocar texto, imágenes, líneas, rectángulos, cómo adquirir los datos y cómo realizar ciertos cálculos para mostrar totales. La presente figura 17 muestra el proceso de generación de un reporte haciendo uso de esta herramienta.

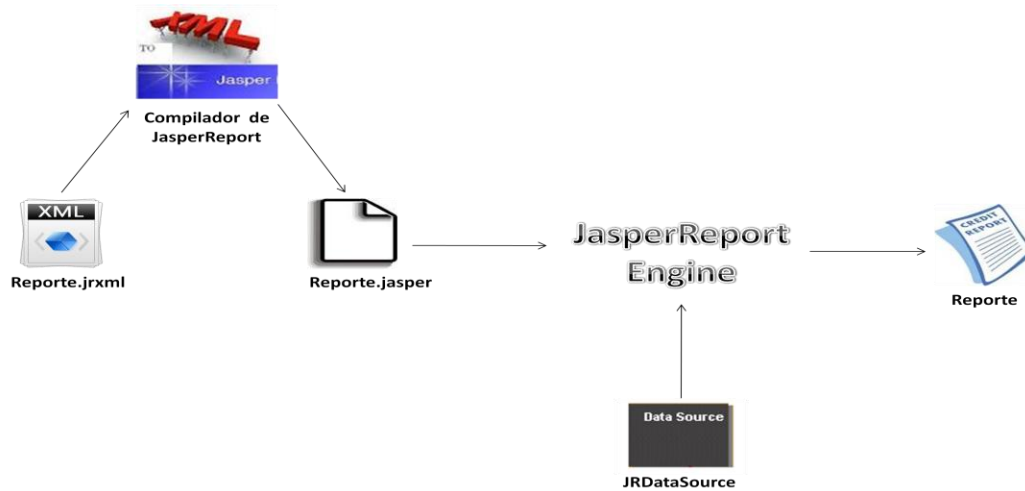


Figura 17: *Funcionamiento de JasperReports.*

La figura 17 muestra como el archivo fuente XML debe ser compilado y los datos requeridos obtenidos para que el motor de JasperReports genere un reporte real. Una vez generados los reportes estos pueden ser exportados a una serie de formatos mencionados en capítulos anteriores.

4.2.4 Secciones del XML de configuración del reporte.

El archivo XML que utiliza JasperReports para la configuración y diseño de los reportes cuenta con un conjunto de secciones que definen su estructura como muestra la figura 18.

Title
pageHeader
columnHeader
detail
columnFooter
pageFooter
lastPageFooter
summary

Figura 18: *Secciones de los reportes.*

En la figura 18 se observan las diferentes secciones que componen un reporte, las cuales se

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

describen a continuación:

Title: El título del reporte debe escribirse en esta sección y solo se mostrará una vez independientemente de las páginas que tenga el informe.

PageHeader: Esta sección es la cabecera de la página, se repite cada vez que se pinta una página nueva.

ColumnHeader: Esta sección es la cabecera de las columnas, en ella se muestran los nombres de los campos que se van a presentar.

Detail: Sección encargada de desplegar los valores correspondientes a los nombres de los campos definidos en la sección anterior, estos datos pueden obtenerse mediante consultas SQL a una base de datos.

ColumnFooter: Pie de la columna. Su comportamiento es análogo a ColumnHeader.

PageFooter: Pie de página, se repite una vez por página. Su comportamiento es análogo a PageHeader.

Summary: Se repite una vez por informe en la última página del mismo. Su comportamiento es análogo a Title.

En cada una de estas secciones JasperReports permite el trabajo dinámico de datos a partir de campos ($\$F$ {nombre_campo}), variables ($\$V$ {nombre_variable}) y parámetros ($\$P$ {nombre_parámetro}). Los parámetros se usan para definir la aparición o no de textos, o para mostrar algún valor concreto que no se pasa como campo. Los campos son aquellos datos que se recuperen de la base de datos y se pasen en un objeto que entienda JasperReports. Las variables se utilizan para hacer cálculos dentro del informe, como por ejemplo calcular una suma de campos o un paginado.

4.3 Pruebas de software.

El desarrollo del *software* implica una serie de actividades de producción en las que las posibilidades de que aparezca la falla humana son comunes. Los errores pueden empezar a darse desde el primer momento del proceso en el que los objetivos pueden estar especificados de forma errónea, así como en los posteriores pasos del diseño y desarrollo.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

La prueba de *software* es un elemento crítico para contrarrestar los errores cometidos y garantizar la calidad del mismo, esta constituye una actividad en la cual un sistema o componente es ejecutado bajo condiciones o requerimientos especificados, los resultados son observados y registrados, y una evaluación es hecha de algún aspecto del sistema o componente. (13)

Con el objetivo de garantizar la calidad del presente módulo se realizaron pruebas de funcionalidad y fiabilidad para la validación de casos de uso, resistencia a fallos y respuestas bajo condiciones anormales (servicios, *hardware* y recursos compartidos no disponible). Estas pruebas se basan en el método de caja negra, el cual se lleva a cabo sobre la interfaz del *software*, para demostrar que las funciones de este son operativas, que la entrada se acepta de forma adecuada y se produce un resultado correcto.

El método de caja negra emplea diferentes técnicas para el desarrollo de las pruebas, se utilizará la Técnica de la Partición de Equivalencia ya que es una de las más efectivas pues permite examinar los valores válidos e inválidos de las entradas existentes en el *software*, descubriendo de forma inmediata errores que, de otro modo, requerirían la ejecución de muchos casos antes de detectar el error genérico.

4.3.1 Diseño de casos de prueba.

La tabla 8 presentada a continuación muestra el diseño de casos de prueba del caso de uso “Exportar Reporte”.

CU Exportar Reporte

1 Descripción general: Este caso de uso le permite al actor generar el reporte y exportar el mismo al formato deseado.

2 Condiciones de ejecución: El usuario debe estar autenticado.

Debe existir conexión con la base de datos.

3 Secciones

Tabla 8: *Diseño de casos de prueba del caso de uso “Exportar Reporte”.*

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
----------------------	--------------------------	---------------------------------	---------------

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

SC.1 Exportar reporte.	EC 1.1 Exportar reporte seleccionado.	Se exporta el reporte seleccionado por el usuario.	<ol style="list-style-type: none">1. El sistema muestra la pestaña reporte.2. El usuario presiona el botón izquierdo del <i>mouse</i> sobre la pestaña reporte y se despliegan las categorías:<ul style="list-style-type: none">• Modos de fallas, fallas funcionales y fallas.• Estrategias de mantenimiento.• Actividades de mantenimiento.3. El usuario selecciona una categoría y el sistema muestra una ventana con:<ul style="list-style-type: none">• Grupo de equipos.• PDF.• HTML.• Imprimir.• Fecha inicial.• Fecha final.• Aceptar.• Cancelar.4. El usuario llena los campos desplegados en la ventana seleccionando el formato deseado.5. El usuario presiona el botón Aceptar.6. El sistema crea el reporte seleccionado y da la posibilidad al
------------------------	---------------------------------------	----------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

			usuario de guardarlo en el directorio deseado.
	EC 1.2 No exportar el reporte.	No se exporta el reporte seleccionado por el usuario.	1. Una vez abierta la ventana en el paso 3 del escenario exportar reporte el usuario puede oprimir el botón Cancelar. El sistema cierra la ventana emergente y no se exporta el reporte.

En la tabla 8 se muestran los escenarios que componen el diseño de casos de prueba para el caso de uso exportar reporte. La tabla 9 mostrada a continuación establece varios casos de pruebas para estos escenarios.

Tabla 9: Casos de pruebas aplicados al caso de uso “Exportar Reporte”.

Escenario de la sección	Grupo Equipo	PDF	HTML	Imprimir	F.I	F.F	Respuesta del sistema	Resultado de la prueba
<i>Exportar reporte.</i>	V	V	-----	N/A	2012 06 01	2012 06 21	El sistema descarga el reporte realizado y da la opción al usuario de guardarlo en la dirección deseada.	Se obtiene un reporte con formato PDF.
<i>Exportar reporte.</i>		-----	V	N/A	2012 06 12	2012 07 01	El sistema muestra el mensaje de error correspondiente.	No se exporta el reporte.
<i>Exportar reporte.</i>	V	V	-----	N/A	2012 07 12	2012 06 21	El sistema muestra el mensaje de error correspondiente.	No se exporta el reporte.

La tabla 9 muestra varios casos de pruebas realizados al caso de uso exportar reporte presentando los resultados obtenidos y las respuestas del sistema para cada una de estas

pruebas.

En la tabla 10 presentada a continuación se muestra el diseño de casos de prueba para el caso de uso “Imprimir Reporte”.

CU Imprimir Reporte

1 Descripción general: Este caso de uso le permite al actor generar el reporte e imprimir el mismo.

2 Condiciones de ejecución: El usuario debe estar autenticado.

Debe existir conexión con la base de datos.

Debe existir conexión con la impresora.

3 Secciones

Tabla 10: *Diseño de casos de prueba del caso de uso “Imprimir Reporte”.*

Nombre de la sección	Escenarios de la sección	Descripción de la funcionalidad	Flujo central
SC.1 Imprimir reporte.	EC 1.1 Imprimir reporte seleccionado.	Se imprime el reporte seleccionado por el usuario.	<ol style="list-style-type: none"> 1. El sistema muestra la pestaña reporte. 2. El usuario presiona el botón izquierdo del <i>mouse</i> sobre la pestaña reporte y se despliegan las categorías: <ul style="list-style-type: none"> • Modos de fallas, fallas funcionales y fallas. • Estrategias de mantenimiento. • Actividades de mantenimiento. 3. El usuario selecciona una categoría y el sistema muestra una ventana con: <ul style="list-style-type: none"> • Grupo de equipos.

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

			<ul style="list-style-type: none"> • PDF. • HTML. • Imprimir. • Fecha inicial. • Fecha final. • Aceptar. • Cancelar. <p>4. El usuario llena los campos desplegados en la ventana seleccionando el <i>radiobutton</i> (Imprimir).</p> <p>5. El usuario presiona el botón Aceptar.</p> <p>6. El sistema crea el reporte seleccionado y solicita la impresora para imprimir el mismo.</p>
	EC 1.2 No imprimir el reporte.	No se imprime el reporte seleccionado por el usuario.	1. Una vez abierta la ventana en el paso 3 del escenario imprimir reporte el usuario puede oprimir el botón Cancelar. El sistema cierra la ventana emergente y no se imprime el reporte.

En la tabla 10 se muestran los escenarios que componen el diseño de casos de prueba para el caso de uso imprimir reporte. La tabla 11 mostrada a continuación establece varios casos de pruebas para estos escenarios.

Tabla 11: *Casos de prueba aplicados al caso de uso "Imprimir Reporte".*

Escenario de la sección	Grupo Equipo	PDF	HTML	Imprimir	F.I	F.F	Respuesta del sistema	Resultado de la prueba
-------------------------	--------------	-----	------	----------	-----	-----	-----------------------	------------------------

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

<i>Exportar reporte.</i>	V	N/A	N/A	V	2012 06 01	2012 06 21	El sistema solicita la impresora predeterminada para imprimir el reporte.	Se imprime el reporte seleccionado.
<i>Exportar reporte.</i>		N/A	N/A	V	2012 06 12	2012 07 01	El sistema muestra el mensaje de error correspondiente.	No se imprime el reporte.
<i>Exportar reporte.</i>	V	N/A	N/A		2012 07 12	2012 06 21	El sistema muestra el mensaje de error correspondiente.	No se imprime el reporte.

La tabla 11 muestra varios casos de pruebas realizados al caso de uso imprimir reporte presentando los resultados obtenidos y las respuestas del sistema para cada una de estas pruebas.

4.3.2 Resultados de las pruebas.

Las pruebas de funcionabilidad y fiabilidad realizadas a partir del método de caja negra presentaron un total de 3 no conformidades en la primera iteración como se muestra en la figura 19.

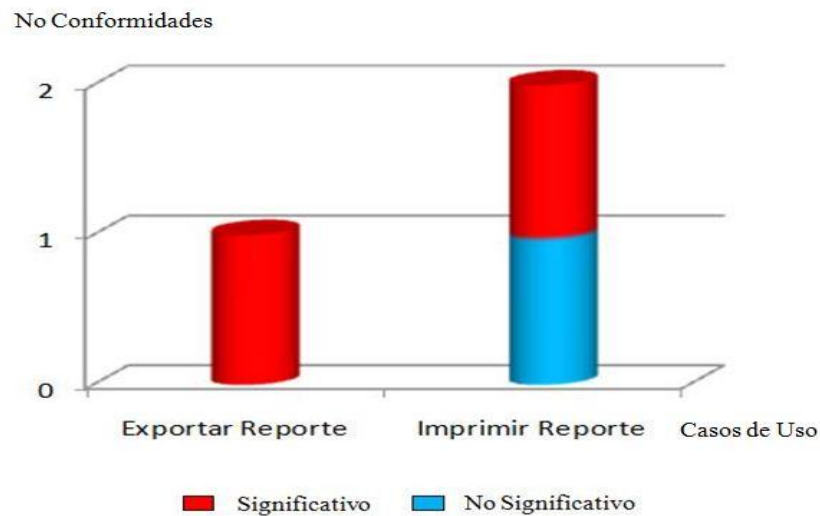


Figura 19: Gráfica de No Conformidades por casos de uso.

La figura 19 muestra como en el caso de uso "Imprimir Reporte" se detectaron 2 no

CAPÍTULO 4: IMPLEMENTACIÓN Y PRUEBA

conformidades, una no significativa y otra significativa la cual podría comprometer el correcto funcionamiento del módulo. Por otra parte, en el caso de uso “Exportar Reporte” se encontró solo una no conformidad significativa. Las no conformidades significativas encontradas en ambos casos de uso estuvieron vinculadas al proceso de conexión con la base de datos, mientras que la no significativa estuvo dirigida principalmente al mejoramiento de la comunicación entre el usuario y la aplicación.

Una vez finalizada la primera iteración se estableció un plazo acorde con la complejidad de cada no conformidad encontrada para dar solución a las mismas e iniciar la posterior iteración. Para el término de esta iteración las no conformidades detectadas anteriormente estaban corregidas y no se encontraron nuevas fallas.

4.4 Validación y verificación por especialistas.

La validación y verificación constituyen un conjunto de procesos de comprobación y análisis que aseguran que el software cumple los requisitos funcionales y no funcionales de su especificación. Estas cuentan con un ciclo de vida completo iniciándose por las revisiones de los requerimientos y del diseño, luego se hacen inspecciones de código hasta la prueba del producto.

Con el objetivo de validar y verificar la solución propuesta por un personal capacitado, después de ser sometida a pruebas de funcionabilidad y fiabilidad por el desarrollador, el módulo de reportes para el SIC fue sometido a pruebas internas en el departamento de Integración y Despliegue de la facultad 5. Para la aplicación de estas pruebas se definieron dos diseños de casos de prueba asociados a los casos de uso Exportar Reporte e Imprimir Reporte, estos cuentan con una estructura específica determinada por el departamento y pueden ser consultadas en los [Anexos](#). Los diseños realizados presentan distintos escenarios en los que el usuario puede encontrarse al interactuar con el sistema estableciendo una lógica de acción y reacción.

Para una primera iteración de pruebas no se obtuvo no conformidades, lo cual está respaldado por el proceso similar de pruebas que se habían aplicado, siendo estos los resultados queda liberado el producto por la facultad y preparado para ser pasado por el Centro Nacional de Calidad de Software (CALISOFT). Como constancia de lo anteriormente expuesto se anexa a este trabajo el documento de Entrega de Servicio de Pruebas.

4.5 Conclusiones parciales.

Con el desarrollo de este capítulo se implementaron todas las clases de diseño obteniéndose un producto completamente funcional, el cual cumple con todos los requisitos funcionales definidos por el cliente.

La realización de pruebas al *software* constituye una fase fundamental en el proceso de desarrollo para garantizar la calidad del producto final, siempre que estas se realicen correctamente podrán detectarse y corregirse fallas que influyen en su calidad.

Conclusiones.

La culminación del Módulo de Reportes para el Sistema Integral de Confiabilidad (SIC) permite arribar a las siguientes conclusiones:

- Las herramientas de acceso a datos que proporciona el *framework* Grails simplifican en gran medida la generación de reportes parametrizables con la biblioteca JasperReports, constituyendo la integración de estas tecnologías una factible solución para el módulo desarrollado.
- La solución obtenida brinda al SIC un módulo capaz de realizar reportes personalizados procedentes de los análisis de confiabilidad, los cuales permiten mostrar la información de forma estructurada y resumida cumpliéndose de esta forma el objetivo propuesto.
- Las pruebas realizadas al *software* permitieron corregir fallas internas que no se habían encontrado durante el proceso de desarrollo, proporcionando una mayor calidad al producto y fiabilidad al cliente.

Recomendaciones.

A partir de la experiencia adquirida en el desarrollo del presente trabajo se proponen las siguientes recomendaciones:

- Utilizar IReport como herramienta para el diseño de las plantillas de los reportes.
- Brindar al usuario una mayor personalización de los reportes sin que esto influya en la pérdida de los datos necesarios.

Referencias bibliográficas.

1. *¿Qué es Confiabilidad Operacional?* **DURÁN, José Bernardo.** 2, s.l. : Revista Club Mantenimiento, 2000.
2. **HUERTA, Rosendo.** Confiabilidad Operacional: Técnicas y Herramientas de Aplicación. [Online] 2004 . [Cited: octubre 7, 2011.] <http://confiabilidad.net/articulos/la-cultura-de-la-confiabilidad-operacional/>
3. Definición de. *Definición de reporte.* [Online] 2008. [Cited: octubre 10, 2011.] <http://www.definicion.de/reportes>.
4. Scribd. *Análisis de Confiabilidad en Sistemas Productivos.* [Online] Scribd, 2009. [Cited: octubre 20, 2011.] <http://www.scribd.com/doc/Analisis-Confiabilidad>.
5. **Menard, Jim.** DataVision.[Online] Sourceforge.net, julio 15, 2008. [Cited: octubre 15, 2011.] <http://datavision.sourceforge.net/>.
6. Eclipse. *BIRT.* [Online] The Eclipse Foundation. [Cited: octubre 15, 2011.] <http://www.eclipse.org/birt/phoenix/>.
7. Crystal Solutions. *Crystal Reports.* [Online] [Cited: noviembre 12, 2011.] <http://www.crysol.net/esp/soluciones/producto/bi/reporting.php>.
8. **ComponentSource.** *Active Report 6.* [Online] 1996. [Cited: noviembre 15, 2011.] <http://www.componentsource.com/index.html>.
9. JasperForge.org. *JasperReports.* [Online] 2000. [Cited: octubre 20, 2011.] <http://jasperforge.org/projects/jasperreports>
10. **Campione, Mary .Walrath, Kathy y Huml, Alison.** *Java Tutorial, Third Edition, A Short Course on the Basics.* s.l. : Addison-Wesley , 2008. 0-201-70393-9.
11. **Abdul-Jawad, Bashar.** *Groovy and Grails Recipes.* United States of America : s.n., s.l.: Apress, 2009. 978-1-4302-1600-1.
12. **Smeets, Bram Boness, Uri y Bankras, Roald.** *Beginning Google Web From Novice to Professional.* United States of America : s.n., s.l.: Apress, 2008. 978-1-4302-1031-3.
13. **Jacobson, Ivan, Booch, Grady y Rumbaugh, James.** 2000. *El Proceso Unificado de Desarrollo de Software.* Madrid: Pearson Educacion, S.A., 2000. ISBN 84-7829-036-2.

Bibliografía.

1. **Arata, Adolfo. 2009.** *Ingeniería y Gestión de la confiabilidad operacional en las plantas industriales*. Chile: Ril Editores, 2009.
2. **Amendola, Luís José. 2002.** “*Modelos Mixtos de confiabilidad*”. s.l.: Datastream. www.mantenimientomundial.com.
3. **Abdul-Jawad, Bashar.** *Groovy and Grails Recipes*. United States of America : s.n., 2009. 978-1-4302-1600-1.
4. **Campione, Mary; Walrath, Kathy y Huml, Alison.** *Java Tutorial, Third Edition, A Short Course on the Basics*. s.l. : Addison-Wesley , 2008. 0-201-70393-9.
5. **Drake, José M. y López, Patricia.** Ctr UC. *Ingeniería de Software: Validación y verificación*. [En línea] 2010 [Citado el: 22 de junio de 2012.] http://www.ctr.unican.es/asignaturas/Ingenieria_Software_4_F/Doc/M7_09_VerificacionValidacion-2011.pdf.
6. **EVA.** Colectivo de Asignatura, Fac 9. *Unidad Didáctica # 3:Diseño, Arquitectura e Integración de Sistema*.
7. **ETAP.** *Solución Empresarial para Sistemas de Potencia de Electricidad*. [En línea] 2008. [Citado el: 20 de junio de 2012.] http://software-gg.com/etap_es.htm.
8. **Glen, Smith y Ledbrook, Peter.** *Grails in Action*. United States of America : s.n., s.l.: Manning Publications Co., 2009.
9. **Hanson, Robert y Tacy, Adam.** *GWT in action: Easy Ajax with the Google Web Toolkit*. s.l.: Manning Publications Co., 2007.
10. **Judd, Christopher M.; Faisal, Joseph y Shingler, James.** *Beginning Groovy and Grails from Novice to Professional*. United States of America : s.n., 2008. 978-1-4302-1045-0.
11. **Konig, Dierk.** *Groovy in Actions*. New York, NY 10021 : Manning Publications Co., 2007. 1-932394-84-2.
12. **Pérez Jaramillo, Carlos Mario.** *¿Qué significa verdaderamente confiabilidad?* VI congreso panamericano de Ingeniería de Mantenimiento, 2004, México-D.F.
13. **Ramsdale, Chris.** *Google Developers Relations - Large scale application development and MVP*. [Online] 2010. <https://developers.google.com/web-toolkit/articles/mvp-architecture>.
14. **Smeets, Bram; Boness, Uri y Bankras, Roald.** *Beginning Google Web From Novice to Professional*. United States of America : s.n., s.l.: Apress, 2008. 978-1-4302-1031-3.

15. **SPM Ingeniería en Mantenimiento.** *Módulo de reportes (SAM)*. [En línea] 2005. [Citado el: 20 de junio de 2012.] <http://www.spm-ing.com/sam/modulo-de-reporte-de-detenciones/>.
16. **Tecnicontrol.** *Soluciones Integrales Confiables*. [En línea] 2008. [Citado el: 20 de junio de 2012.] <http://portal.tc.com.co/tecnicontrol/software-tecnico>.

Glosario de términos.

-A-

Activos: conjunto de bienes tangibles o intangibles de los que es titular una empresa.

Análisis de Confiabilidad: ejecución de diversos tipos de exámenes a productos o sistemas para determinar cuan confiable es el producto o sistema en cuestión.

API: acrónimo de “Application Programming Interface”. Interfaz de programación de aplicaciones que puede ser utilizada por otro software como una capa de abstracción.

AJAX: acrónimo de “Asynchronous JavaScript And XML”. Conjunto de tecnologías que permite crear aplicaciones web interactivas.

-B-

Bytecode: conjunto de instrucciones diseñadas para ser ejecutadas por un intérprete de software.

-C-

Crosstab: tabla que muestra la distribución conjunta de dos o más variables simultáneamente.

CVS: acrónimo de Current Version System. Aplicación informática que posibilita el control de versiones.

-D-

Debugging: proceso de identificación y corrección de errores de programación.

-E-

EJB: acrónimo de “Enterprise JavaBeans”. Tecnología que conforma la arquitectura de componentes en el servidor para la plataforma J2EE.

-F-

Framework: marco de trabajo que constituye una estructura conceptual y tecnológica

compuesta por componentes de softwares que permiten el desarrollo rápido de aplicaciones.

-H-

Hibernate: *framework* de persistencia de código abierto para la plataforma Java.

-J-

JDBC: acrónimo de “Java Database Connectivity”. API que permite la ejecución de operaciones sobre base de datos desde el lenguaje Java.

J2EE: acrónimo de “Java 2 Enterprise Edition”. Conjunto de tecnologías que define el estándar para el desarrollo de aplicaciones empresariales de varios niveles.

-O-

Objeto-relacional: Término que se utiliza para describir a base de datos que han evolucionado desde el modelo relacional hacia otro más amplia que incorpora conceptos del paradigma orientado a objetos.

-P-

POJO: acrónimo de “Plain Old Java Object”. Objetos simples de Java que no sigue las convenciones de marcos de trabajo determinados.

Plug-in: módulo de hardware o software que añade funcionalidades a un sistema más grande.

Parametrizar: modificación dinámica de datos que garantiza la actualización e interacción de variables.

Parsear: proceso de análisis sintáctico de un archivo (Ejemplo: XML).

-R-

RPC: acrónimo de “Remote Procedure Call”. Mecanismo basado en los Servlets de Java para proveer acceso a los recursos del servidor.

-S-

Spring: *framework* de código abierto para la plataforma Java.

-W-

Widget: término genérico de una GUI que brinda una interfaz a los usuarios para interactuar con la aplicación y el sistema operativo.

Anexos.

Anexo 1: *Diagrama de clases del análisis del caso de uso "Exportar Reporte".*

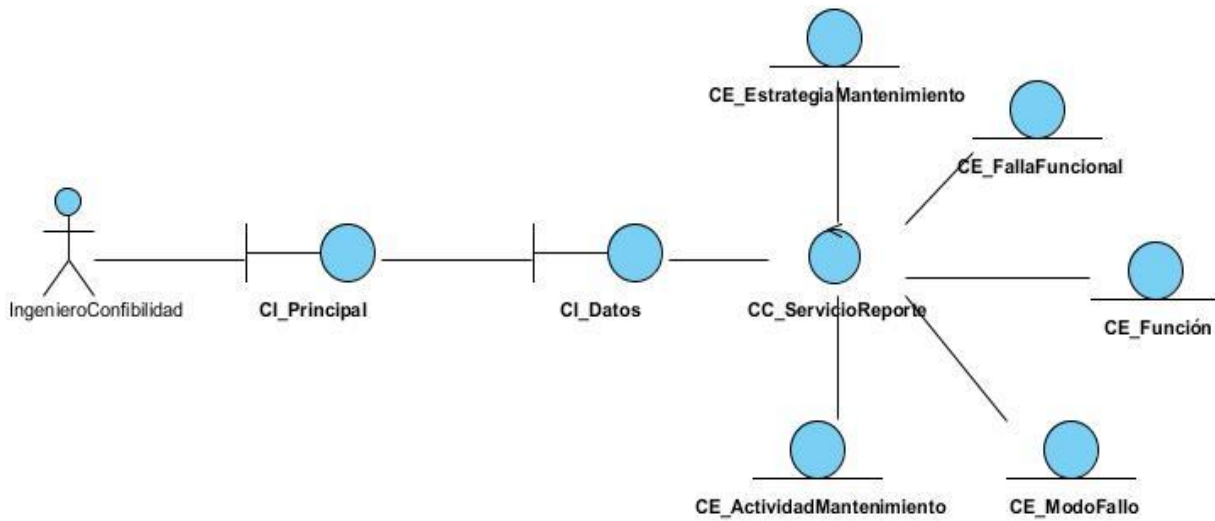


Figura 20: *Diagrama de clases del análisis del caso de uso "Exportar Reporte".*

Anexo 2: Diagrama de colaboración del caso de uso “Exportar Reporte”.

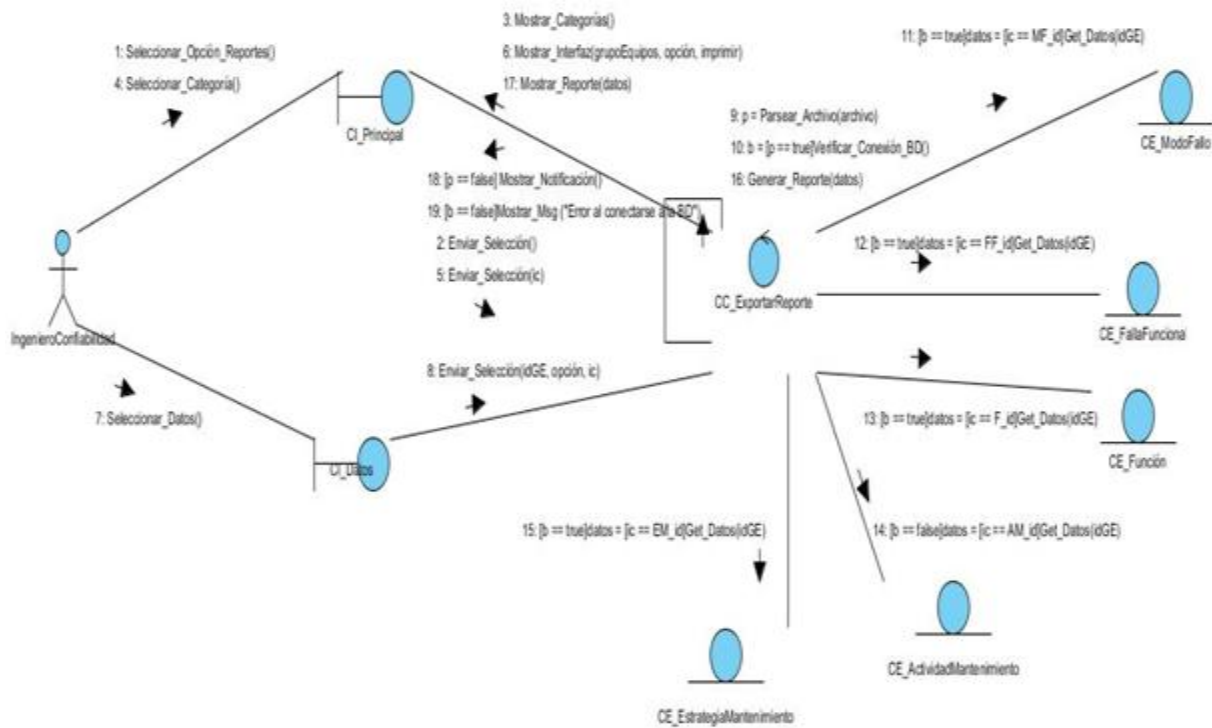


Figura 21: Diagrama de colaboración del caso de uso “Exportar Reporte”.

Anexo 3: **Diagrama de secuencia del caso de uso “Exportar Reporte”.**

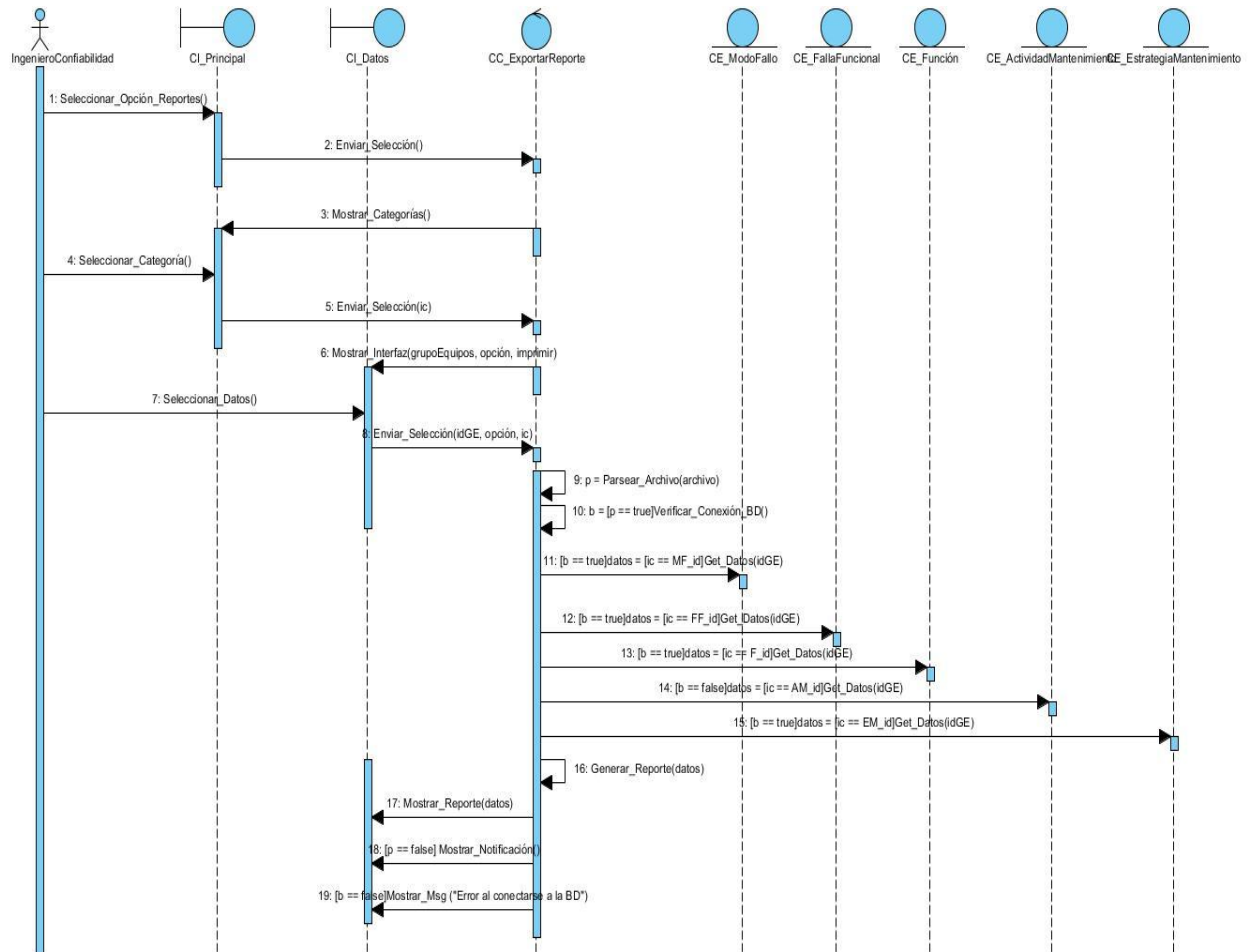


Figura 22: Diagrama de secuencia del caso de uso “Exportar Reporte”.

Anexo 4: **Diagrama de clases del análisis del caso de uso "Imprimir Reporte".**

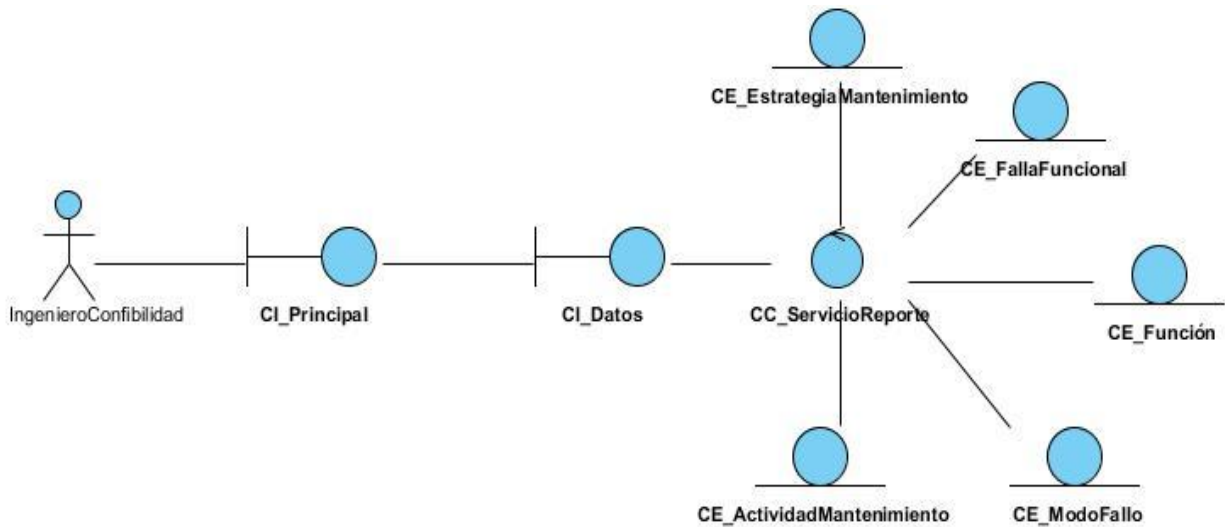


Figura 23: Diagrama de secuencia del caso de uso "Imprimir Reporte".

Anexo 5: **Diagrama de colaboración del caso de uso “Imprimir Reporte”.**

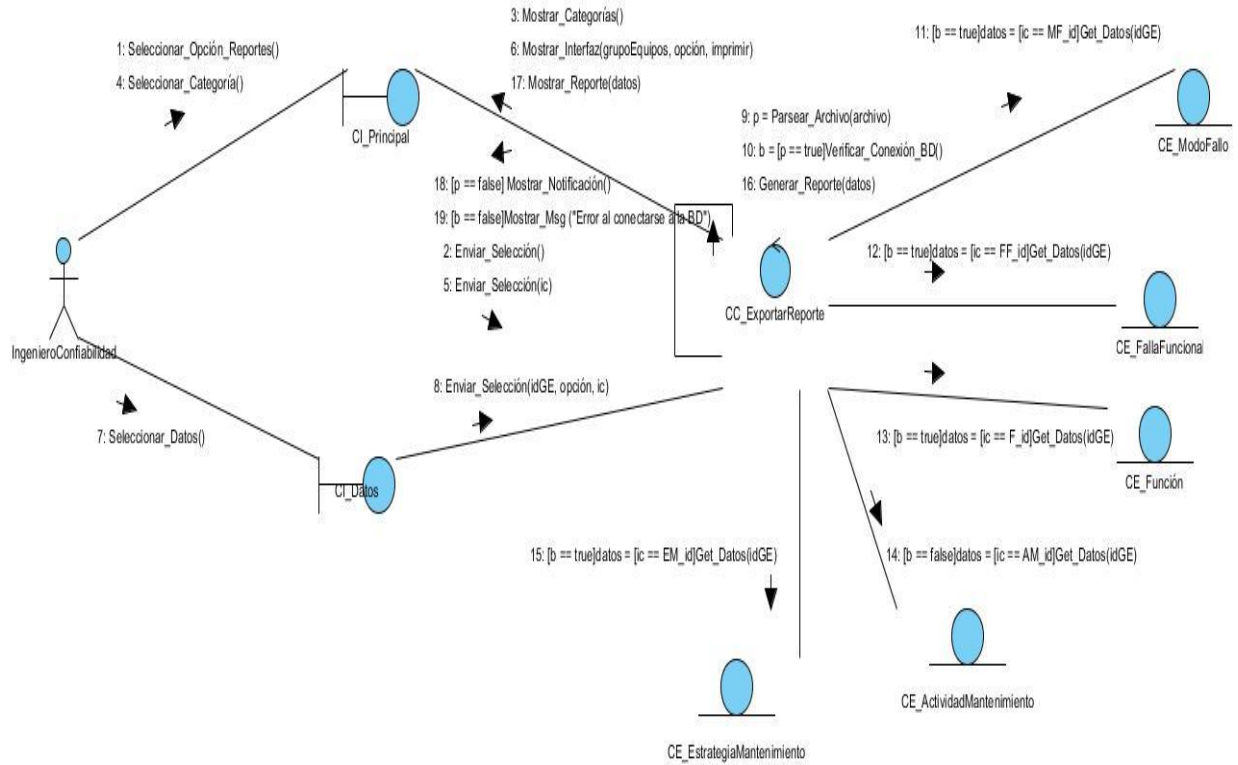


Figura 24: Diagrama de colaboración del caso de uso “Imprimir Reporte”.

Anexo6: Diagrama de secuencia del caso de uso "Imprimir Reporte".

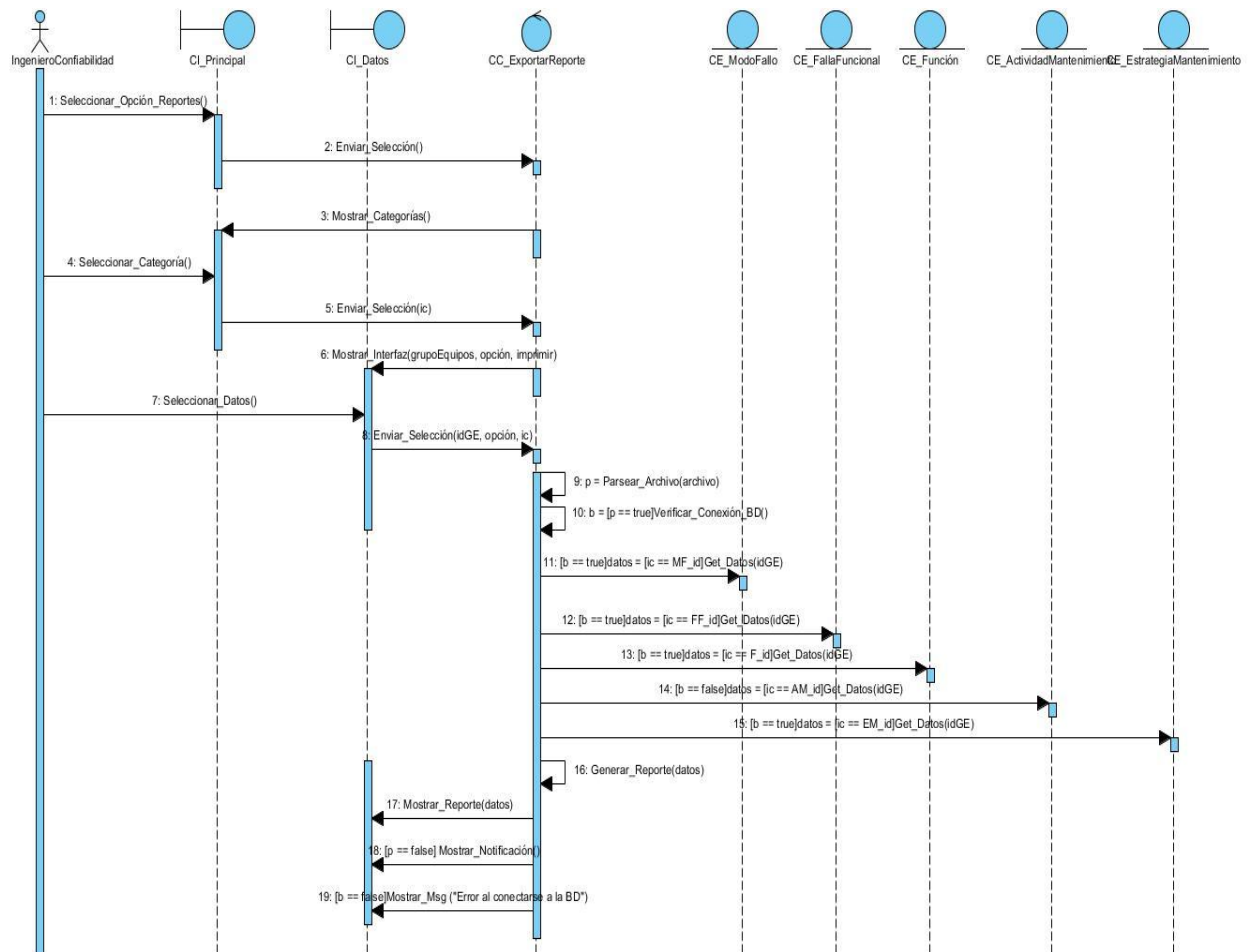



Figura 25: Diagrama de secuencia del caso de uso "Imprimir Reporte".

Anexo 7: Reporte de Fallas Funcionales de un Grupo de Equipos determinado.



Sistema Integral de Confiabilidad Operacional
14 06 2012

Reporte de las Fallas Funcionales asociadas al Grupo de Equipos GE_ID_001

ID	Nombre de la Falla Funcional	Tipo de Falla Funcional
1. E_ID_01		
FF_ID_01	Nombre_FF_01	Tipo_FF_01
FF_ID_02	Nombre_FF_02	Tipo_FF_02
FF_ID_05	Nombre_FF_05	Tipo_FF_05
Cantidad de Fallas Funcionales : 3		
2. E_ID_03		
FF_ID_08	Nombre_FF_08	Tipo_FF_08
FF_ID_03	Nombre_FF_03	Tipo_FF_03
Cantidad de Fallas Funcionales : 2		
3. E_ID_04		
FF_ID_04	Nombre_FF_04	Tipo_FF_04
FF_ID_06	Nombre_FF_06	Tipo_FF_06
FF_ID_07	Nombre_FF_07	Tipo_FF_07
Cantidad de Fallas Funcionales : 3		

Ing. Yuniel Sardiña Garcia

Página 1

Figura 26: Reporte de Fallas Funcionales de un Grupo de Equipos determinado.

Anexo 8: Reporte de las Estrategias de Mantenimiento aplicadas a un Grupo de Equipos determinado.

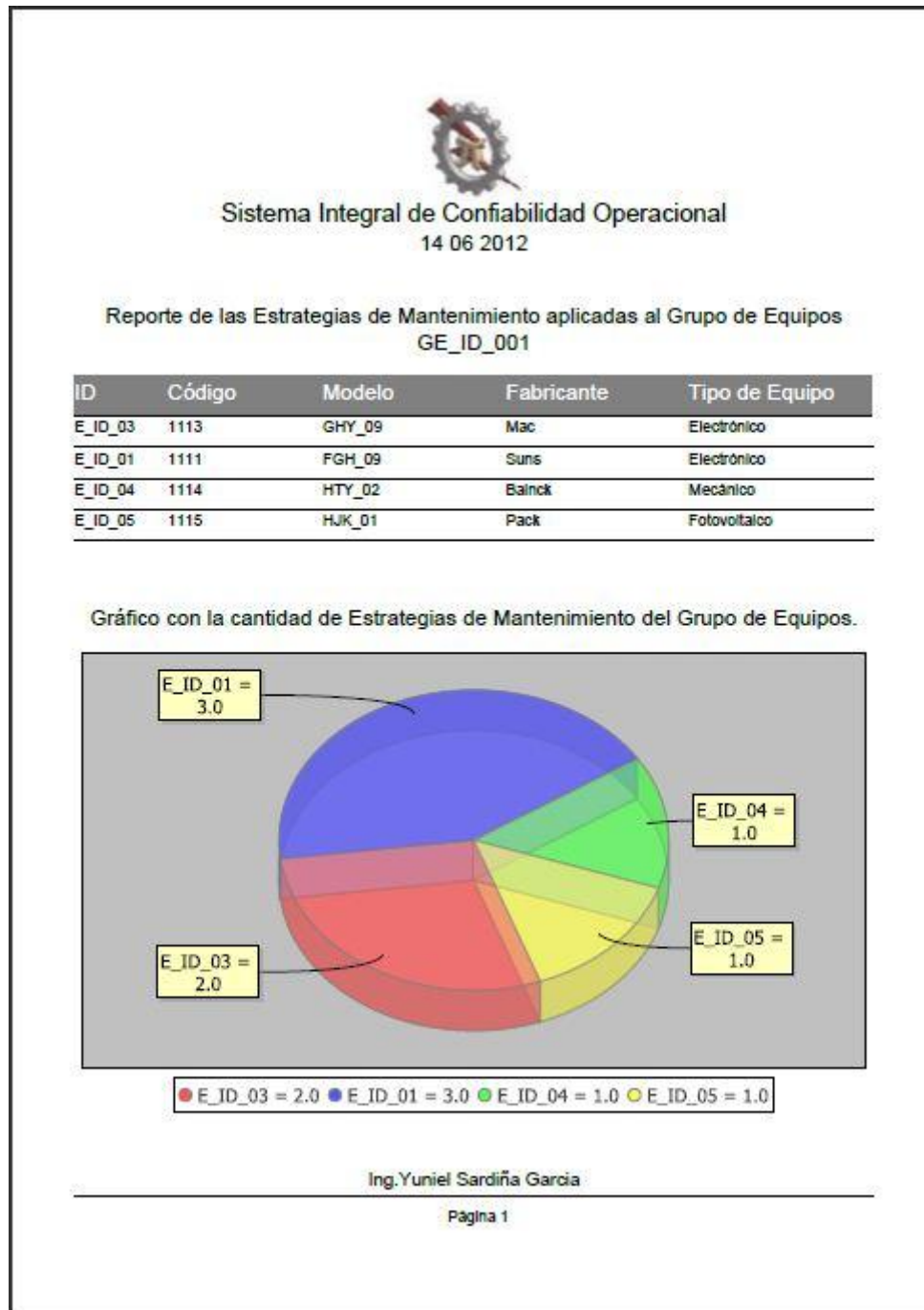


Figura 27: Reporte de las Estrategias de Mantenimiento aplicadas a un Grupo de Equipos determinado.

Anexo 9: Reporte de las Actividades de Mantenimiento asociadas a un Grupo de Equipos determinado.

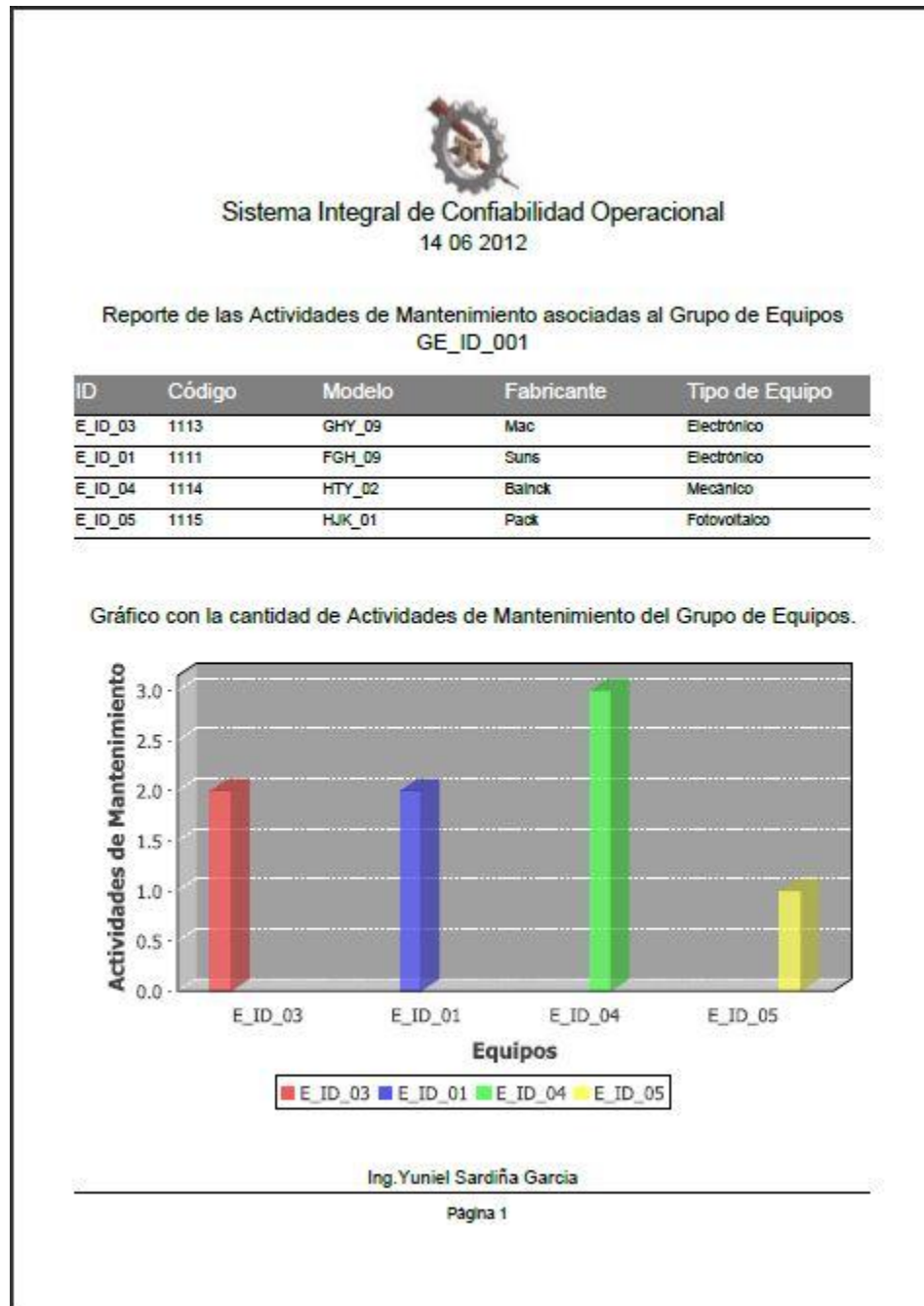


Figura 28: Reporte de las Actividades de Mantenimiento asociadas a un Grupo de Equipos determinado.

Anexo 10: **Diseño de casos de pruebas para el caso de uso Exportar Reporte.**

Descripción general									
Este caso de uso le permite al actor generar el reporte y exportar el mismo al formato deseado.									
Condiciones de ejecución									
El usuario debe estar autenticado.									
Existe conexión con la base de datos.									
SC <Exportar Reporte>									
Escenario	Descripción	GE	PDF	HTML	Imprimir	F.I	F.F	Respuesta del sistema	Flujo central
EC 1.1 Exportar reporte seleccionado.	1. El usuario presiona el botón izquierdo del mouse sobre la pestaña reporte. 3. El usuario selecciona una categoría. 5. El usuario llena los campos desplegados en la ventana seleccionando el formato deseado (PDF o HTML) y presiona el botón Aceptar.	V GE_ID_01	V Seleccionar radiobutton.	I	N/A	V 2012 June 01	V 2012 June 23	2. El sistema despliega las categorías: • Modos de fallas, fallas funcionales y fallas. • Estrategias de mantenimiento. • Actividades de mantenimiento. 4. El sistema muestra una ventana con: • Grupo de equipos • PDF • HTML • Imprimir • Fecha inicial • Fecha final • Aceptar • Cancelar 6. El sistema crea el reporte seleccionado y da la posibilidad al usuario de guardarlo en el directorio deseado.	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo y exportarlo al formato escogido.
EC 1.2 No exportar reporte seleccionado.	1. El usuario presiona el botón izquierdo del mouse sobre la pestaña reporte. 3. El usuario selecciona una categoría. 5. El usuario presiona el botón Cancelar.	V GE_ID_01	V Seleccionar radiobutton.	V Seleccionar radiobutton.	N/A	V 2012 June 01	V 2012 June 23	2. El sistema despliega las categorías: • Modos de fallas, fallas funcionales y fallas. • Estrategias de mantenimiento. • Actividades de mantenimiento. 4. El sistema muestra una ventana con: • Grupo de equipos • PDF • HTML • Imprimir • Fecha inicial • Fecha final • Aceptar • Cancelar 6. El sistema retorna a la página principal.	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo y exportarlo al formato escogido.
EC 1.3 Exportar reporte seleccionado dejando el campo GE vacío.	1. Ejecutar los pasos 1,2,3 y 4 del escenario 1.1 y presionar el botón Aceptar dejando el GE vacío.	I			N/A	V 2012 June 01	V 2012 June 23	2. El sistema muestra el mensaje de error correspondiente "Seleccione el Grupo de Equipos".	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo y exportarlo al formato escogido.
EC 1.4 Exportar reporte seleccionado dejando el formato vacío.	1. Ejecutar los pasos 1,2,3 y 4 del escenario 1.1 y presionar el botón Aceptar dejando el formato vacío.	V GE_ID_01	I		N/A	V 2012 June 01	V 2012 June 23	2. El sistema muestra el mensaje de error correspondiente "Seleccione el formato del reporte".	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo y exportarlo al formato escogido.
EC 1.5 Exportar reporte seleccionado escogiendo mal las fechas.	1. Ejecutar los pasos 1,2,3 y 4 del escenario 1.1 y presionar el botón Aceptar escogiendo mal las fechas.	V GE_ID_01			N/A	I 2012 June 6	I 2012 June 1	2. El sistema muestra el mensaje de error correspondiente "Fechas mal seleccionadas".	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo y exportarlo al formato escogido.

Figura 29: **Diseño de casos de pruebas para el caso de uso Exportar Reporte.**

Anexo 11: *Diseño de casos de pruebas para el caso de uso Imprimir Reporte.*

Descripción general									
Este caso de uso le permite al actor genera									
Condiciones de ejecución									
El usuario debe estar autenticado.									
Existe conexión con la base de datos.									
Existe conexión con la impresora.									
SC <Imprimir Reporte>									
Escenario	Descripción	GE	PDF	HTML	Imprimir	F.I	F.F	Respuesta del sistema	Flujo central
EC 1.1 Imprimir reporte seleccionado.	1. El usuario presiona el botón izquierdo del mouse sobre la pestaña reporte. 3. El usuario selecciona una categoría. 5. El usuario llena los campos desplegados en la ventana seleccionando el radiobutton (Imprimir) y presiona el botón Aceptar.	V GE_ID_01	N/A	N/A	V Seleccionar radiobutton.	V 2012 June 01	V 2012 June 23	2. El sistema despliega las categorías: • Modos de fallas, fallas funcionales y fallas. • Estrategias de mantenimiento. • Actividades de mantenimiento. 4. El sistema muestra una ventana con: • Grupo de equipos • PDF • HTML • Imprimir • Fecha inicial • Fecha final • Aceptar • Cancelar 6. El sistema crea el reporte seleccionado e imprime el mismo en la impresora predeterminada.	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo e imprimirlo en la impresora predeterminada.
EC 1.2 No Imprimir reporte seleccionado.	1. El usuario presiona el botón izquierdo del mouse sobre la pestaña reporte. 3. El usuario selecciona una categoría. 5. El usuario presiona el botón Cancelar.	V GE_ID_01	N/A	N/A	V Seleccionar radiobutton.	V 2012 June 01	V 2012 June 23	2. El sistema despliega las categorías: • Modos de fallas, fallas funcionales y fallas. • Estrategias de mantenimiento. • Actividades de mantenimiento. 4. El sistema muestra una ventana con: • Grupo de equipos • PDF • HTML • Imprimir • Fecha inicial • Fecha final • Aceptar • Cancelar 6. El sistema retorna a la página principal.	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo e imprimirlo en la impresora predeterminada.
EC 1.3 Imprimir reporte seleccionado dejando el campo GE vacío.	1. Ejecutar los pasos 1,2,3 y 4 del escenario 1.1 y presionar el botón Aceptar dejando el GE vacío.	I	N/A	N/A	V Seleccionar radiobutton.	V 2012 June 01	V 2012 June 23	2. El sistema muestra el mensaje de error correspondiente "Seleccione el Grupo de Equipos".	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo e imprimirlo en la impresora predeterminada.
EC 1.4 Imprimir reporte seleccionado dejando la opción Imprimir sin seleccionar.	1. Ejecutar los pasos 1,2,3 y 4 del escenario 1.1 y presionar el botón Aceptar dejando la opción Imprimir sin seleccionar.	V GE_ID_01	N/A	N/A	I	V 2012 June 01	V 2012 June 23	2. El sistema muestra el mensaje de error correspondiente "Seleccione el formato del reporte".	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo e imprimirlo en la impresora predeterminada.
EC 1.5 Imprimir reporte seleccionado escogiendo mal las fechas.	1. Ejecutar los pasos 1,2,3 y 4 del escenario 1.1 y presionar el botón Aceptar escogiendo mal las fechas.	V GE_ID_01			N/A	I 2012 June 6	I 2012 June 1	2. El sistema muestra el mensaje de error correspondiente "Fechas mal seleccionadas".	Seleccionar la pestaña reporte y escoger uno de los reportes existentes para crearlo e imprimirlo en la impresora predeterminada.

Figura 30: *Diseño de casos de pruebas para el caso de uso Imprimir Reporte.*