Universidad de las Ciencias Informáticas

Facultad 2

Herramienta informática de generación de reportes dinámicos basados en Apache Solr para el Sistema para repositorios digitales REPXOS 3.0.

Trabajo de Diploma para optar por el Título de

Ingeniero en Ciencias Informáticas

Autores: Lázaro Javier Pérez Moleiro

Aramis Alejandro Sánchez Gutierrez

Tutores: Ing. Gleidis Yuriannis Rosabal Espinosa

Ing. Deylert Pérez Rivera

La Habana, junio de 2015 “Año 57 de la Revolución”

"El pensamiento estadístico será algún día tan necesario para el ciudadano competente como la habilidad de leer y escribir"

Herbert George Wells

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

Para que así conste firmamos la presente a los 18 días del mes de junio del año 2015.

Autores:

Aramis Alejandro Sánchez Gutierrez	Lázaro Javier Pérez Moleiro

Tutores:

Ing. Gleidis Yuriannis Rosabal Espinosa	Ing. Deylert Pérez Rivera

DATOS DE CONTACTO

Tutores:

Ing. Gleidis Yuriannis Rosabal Espinosa: Graduada en el año 2010 como Ingeniera en Ciencias Informáticas, en la Universidad de las Ciencias Informáticas. Se desempeña como especialista B en Ciencias Informáticas en el Departamento de Aplicaciones del Centro CIGED ocupando el rol de analista del proyecto Repositorio Institucional. Correo electrónico: gyrosabal@uci.cu.

Ing. Deylert Pérez Rivera: graduado en el año 2013 como Ingeniero en Ciencias Informáticas. Se desempeña como recién graduado en adiestramiento en el Centro CIGED de la Facultad 2, ocupando el rol de desarrollador en el proyecto Repositorio Institucional.
Correo electrónico: drivera@uci.cu.

DEDICATORIA

A mi abuelo Ramón…

A quien yo llamaba cariñosamente "Guateque", que aunque no se encuentre físicamente entre nosotros, soñó siempre con este momento e hizo todo lo posible para que sucediera.
A mis mamas Eliza y Elsita que continuaron el sueño de mi abuelo con el mismo ímpetu.

Aramis

A mis padres...

porque me dieron la vida, porque siempre creyeron en mí, por haber contado con su apoyo incondicional para cumplir mi sueño y haber estado ahí siempre que los necesite.
A ustedes les debo quien soy.

Lachy

AGRADECIMIENTOS

A mis abuelos Ramón y Eliza por educarme y hacer de mí quien soy, les debo todo.

A mi mamá Elsita por darme la vida y siempre trasmitirme con un inmenso amor esa confianza eterna que tiene en mí y a mi padrastro Jurniurt por apoyarme en todos los momentos.

A mi papá por enseñarme más allá de los conocimientos de la escuela y demostrarme que siempre se puede más.

A mi hermanito David que sigue mis pasos, obligándome a crecer como estudiante con la tarea de poner la varilla bien alta ya que él piensa que soy el hermano mayor de Big Hero.

A mi bisabuela Esther que con su cariño, amor y preocupación siempre ha demostrado interés en mi carrera.

A mi abuelo Alejandro, mis tías Mily y Emni por toda la ayuda que me han dado y por su preocupación en todo momento.

A mis tías Damaris, Mariester, mi prima Suyen por estar siempre pendientes de mi carrera y ayudarme en todo lo posible.

A mis amigos y hermanos aquí en la universidad Amigo, Lachy, Héctor y Alberto que han atravesado junto a mí por problemas, secretos y fiestas.

A Claudia Jiménez que me ayudo en mi vida académica de todas las maneras posibles, a ti Claudia gracias.

A mis amigos Alexis, el calvo, Bexi, Aguilera, Yuliet, Cristina, Zahyli, Jefrey, el luisma, Maidevis, Jairo y la gente de Artemisa por aguantar mis pesadeces con alegría y paciencia.

A mi tutora Gleidis por aguantar mi presencia a cualquier hora del día y ayudarme como una amiga más. A todos aquellos profesores desde la primaria que aportaron a mi formación como estudiante.

A todos aquellos tanto dentro como fuera de la universidad que han demostrado interés en mi carrera, muchas gracias.

Aramis

A mis padres por la paciencia de tantos años, por los regaños y el amor que siempre me han dado, por ser mis guías en la vida, este es mi regalo, los quiero con todo mi corazón.

A mi hermano por ser un ejemplo como hombre y padre, porque a pesar de la distancia siempre he tenido su apoyo.
A mis tías Adriana e Ivon que siempre estuvieron a mi lado velando por mí, moviendo cielo y tierra.
A la que fue mi familia durante muchos años Mabel, Yiye y Maybel , aunque hoy no estén aquí, este también es resultado de su sacrificio.

A mis abuelos que ya no están hoy con nosotros, especialmente a mi abuela Ada que siempre estuvo cerca malcriándome, los llevo en mi corazón.

A mami ceney que siempre ha rezado por mí, y mi tío nene que tanto me ha ayudado.
A toda mi familia, mis tíos, tías, mis primos y mis sobrinos, porque siempre han estado ahí y son la mejor familia del mundo, los quiero a todos.

A los hermanos con los que tuve la dicha de vivir durante 5 años, Héctor y Alberto ustedes han estado en las buenas y malas, cuando otros no podían estar por la distancia, ustedes me soportaron día a día, y yo a ustedes y lo que nos queda...

A mis hermanos habaneros Fiax y Moro gracias por estar ahí, por brindarme siempre su apoyo, de ustedes he aprendido lo mejor.

A mi compañero de tesis y hermano, especialmente por la paciencia a lo largo de este largo trabajo, sé que no fue fácil, y por demostrarme tu amistad cuando la cosa se puso fea, tú sabes de lo que hablo.

A mis hermanos de pinar Maybel, Fiki y el Pollo.
A todos mis amigos, gracias por sus consejos y por ensenarme el valor de la amistad.

A mis compañeros de aula y a mis compañeros de la casa.
A todos los profesores que contribuyeron a mi formación desde el primer día en esta universidad, especialmente a nuestra tutora.

Lachy

Resumen

El Centro de Informatización de la Gestión Documental CIGED, es uno de los centros de desarrollo de software de la Universidad de las Ciencias Informáticas. Actualmente se encuentra desarrollando el Sistema para repositorios digitales REPXOS 3.0 basado en Dspace 4.2. Este último es una herramienta para la administración de colecciones digitales. El sistema cuenta con un módulo de reportes propio de la herramienta DSpace 4.2, que brinda reportes estáticos y básicos, predefinidos en la implementación del sistema. Esto provoca una dependencia total del equipo de desarrollo, que para posibles reformas de un nuevo reporte estadístico, requeriría ejecutar transformaciones en el código del sistema, consumiendo tiempo y recursos a la entidad. Para darle solución a esta problemática se plantea desarrollar una herramienta para el sistema REPXOS 3.0 que permita generar los reportes de forma que puedan ser transformados según las necesidades de una institución. En función de darle cumplimiento al objetivo se realizó un análisis de los sistemas generadores de reportes existentes. Además se definieron un conjunto de tecnologías, herramientas y lenguajes, que guiado por la metodología RUP, permitieron llevar a cabo la construcción del sistema. Se implementaron todas las funcionalidades propuestas y se validaron mediante pruebas de software. Las pruebas arrojaron resultados satisfactorios, lo cual demuestra la calidad de la solución de la propuesta. Como resultado del trabajo se obtuvo una herramienta que realiza reportes de forma dinámica y responde a las necesidades estadísticas del REPXOS 3.0, sin necesidad de realizar cambios en su código fuente.

Palabras claves: DSpace, Reportes dinámicos, Repositorio Digital**,** Solr

TABLA DE CONTENIDOS

INTRODUCCIÓN 1

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR..... 5

1.1. CONCEPTOS FUNDAMENTALES..... 5

1.2. SISTEMAS GENERADORES DE REPORTES DINÁMICOS 8

1.3. METODOLOGÍAS, TECNOLOGÍAS, HERRAMIENTAS Y LENGUAJES UTILIZADOS 13

CAPÍTULO 2. CARACTERÍSTICAS DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR 24

2.1. DESCRIPCIÓN DE LA HERRAMIENTA PROPUESTA..... 24

2.2. MODELO DE DOMINIO 25

2.3. ESPECIFICACIÓN DE REQUISITOS DE SOFTWARE..... 27

2.4. DEFINICIÓN DE LOS ACTORES 32

2.5. DIAGRAMA DE CASOS DE USO DE SISTEMA..... 32

CAPÍTULO 3. ARQUITECTURA Y DISEÑO DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR 39

3.1. MODELO DE DISEÑO..... 39

3.2. MODELO DE DATOS 41

3.3. MODELO ARQUITECTÓNICO..... 44

3.4. PATRONES DE DISEÑO UTILIZADOS EN LA HERRAMIENTA SISTEMA DE REPORTES 46

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR..... 49

4.1. MODELO DE IMPLEMENTACIÓN 49

4.2. ESTÁNDARES DE CODIFICACIÓN 53

4.3. PRUEBAS DE SOFTWARE..... 53

4.4. RESULTADOS DE LAS PRUEBAS APLICADAS A LA HERRAMIENTA SISTEMA DE REPORTES 58

CONCLUSIONES 62

RECOMENDACIONES 63

REFERENCIAS BIBLIOGRÁFICAS 64

GLOSARIO DE TÉRMINOS..... 67

Índice de Tablas

Tabla 1 Requisitos Funcionales de la Herramienta informática para la gestión de reportes dinámicos basados en consulta Solr (fuente: elaboración propia) 28
Tabla 2 Diagrama de CU Gestionar Usuarios (fuente: elaboración propia)..... 34
Tabla 3 Diagrama de CU Gestionar Modelos De Reportes (fuente: elaboración propia) 35
Tabla 4 Diagrama de CU Autenticar Usuario (fuente: elaboración propia)..... 35
Tabla 5 Diagrama de CU Buscar Modelo de Reporte (fuente: elaboración propia)..... 36
Tabla 6 Diagrama de CU Buscar Reporte (fuente: elaboración propia) 36
Tabla 7 Diagrama de CU Gestionar Servidores (fuente: elaboración propia)..... 37
Tabla 8 Diagrama de CU Exportar a PDF (fuente: elaboración propia) 37
Tabla 9 Diagrama de CU Gestionar Reportes (fuente: elaboración propia) 38
Tabla 10 Caso de Prueba de Caja negra Autenticar Usuario (fuente: elaboración propia) 57
Tabla 12 Resumen de no conformidades del proceso de liberación interno (fuente: Registro de Revisiones de Inconsistencias) 59

Índice de Figuras

Figura 1 Modelo de Dominio de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. (Fuente: elaboración propia) 26
Figura 2 Diagrama de Caso de Usos del Sistema (fuente: elaboración propia). 33
Figura 3 Diagrama de clases del diseño CU Gestionar Modelo de Reporte (fuente: elaboración propia) 40
Figura 4 Diagrama de colaboración crear modelo de reporte (fuente: elaboración propia) 41
Figura 5 Modelo Físico de Datos del sistema Herramienta informática para la gestión de reportes dinámicos basado en consulta solr. (Fuente: elaboración propia) 42
Figura 6 Diagrama de Paquetes del Sistema (fuente: elaboración propia) 43
Figura 7 Diagrama de Arquitectura N capas utilizada por Grails (fuente: tomado de (14)) 45
Figura 8 Diagrama de Componentes de la Capa de Presentación (fuente: elaboración propia) 50
Figura 9 Diagrama de Componentes Capa de Control (fuente: elaboración propia) 51
Figura 10 Diagrama de Componentes Capa de Lógica de Negocio (fuente: elaboración propia) .. 51
Figura 11 Diagrama de Despliegue del Sistema (fuente: elaboración propia) 52
Figura 12 Modelo del proceso de prueba de software. (Fuente: elaboración propia) 54
Figura 13 Prueba de Carga a la funcionalidad crear modelo de reporte con 60 usuarios. (Fuente: elaborado con (36)) 60
Figura 14 Prueba de Carga a la funcionalidad crear reporte con 60 usuarios. (Fuente: elaborado con (36)) 60
Figura 15 Prueba de Carga a la funcionalidad crear modelo de reporte con 200 usuarios. (Fuente: elaborado con (36)) 60
Figura 16 Prueba de Carga a la funcionalidad crear reporte con 200 usuarios. (Fuente: elaborado con (36)) 61

INTRODUCCIÓN

En la actualidad, la estadística se ha convertido en un método efectivo para describir con exactitud los valores de los datos económicos, políticos, sociales, psicológicos, biológicos y físicos. Se utiliza como herramienta para relacionar y analizar dichos datos teniendo una injerencia directa en cuestiones sociales. Según el manual de estadísticas del profesor David Ruiz Muñoz y la recopilación de conceptos de estadísticas de Minguez y Chacón se puede definir la estadística como una ciencia formal y una herramienta que estudia el uso y los análisis provenientes de una muestra representativa de datos (1). Está relacionada con el estudio de procesos cuyo resultado es más o menos imprescindible y con la finalidad de obtener conclusiones para tomar decisiones razonables de acuerdo con tales observaciones.

En una organización las estadísticas constituyen una valiosa fuente de información, debido a su importante apoyo en el proceso de toma de decisiones. Contribuyen a agilizar el proceso de control y planificación, y permiten detectar nuevas oportunidades de negocios o servicios. Una forma eficiente de estudiar las estadísticas es mediante los reportes. Estos permiten visualizar análisis y resultados, facilitando la identificación de patrones o la clasificación de datos. Los reportes estadísticos organizan y exhiben la información contenida en las bases de datos, aplicando un formato determinado a los mismos, para después mostrarlos por medio de un diseño atractivo y fácil de interpretar por los usuarios (1).

Para la realización de reportes en la actualidad se destacan como herramientas informáticas los sistemas generadores de reportes. Estos utilizan una especie de lenguaje transparente para el usuario, por medio del cual este realiza consultas a la base de datos y se obtiene información en forma de reporte. Entre estos sistemas marcan la diferencia aquellos que lo realizan de forma dinámica. Los sistemas generadores de reportes dinámicos ofrecen un mayor control sobre la información permitiendo al usuario definir el aspecto y el contenido deseado de acuerdo a sus necesidades (2).

En la informatización de una empresa, los reportes estadísticos generados de forma dinámica pueden brindar eficiencia y eficacia en la gestión de información relevante. Esto produce un incremento en la capacidad de organización de la empresa**,** que influye de forma significativa en los procesos productivos y de toma de decisiones.

El Centro de Informatización de la Gestión Documental CIGED, es uno de los centros de desarrollo de software de la Universidad de las Ciencias Informáticas. Actualmente se encuentra desarrollando el Sistema para repositorios digitales REPXOS 3.0 que está basado en DSpace 4.2. Dspace es un software de código abierto que provee herramientas para la administración de colecciones digitales. REPXOS 3.0 cuenta con un módulo de reportes propio de la herramienta DSpace 4.2, que brinda reportes estáticos y básicos que son predefinidos cuando se implementa el sistema. Esta situación provoca que una vez instalado el repositorio en una institución, la consulta para obtener las estadísticas siempre estará predeterminada en los reportes. Teniendo en cuenta esta condición, ningún directivo o interesado podrá adquirir el conocimiento de otro estado del repositorio que desee analizar. Además, no podrán ser atentidas nuevas necesidades de estadísticas sobre el uso del mismo.

Esto provocaría una dependencia total del equipo de desarrollo, que para posibles reformas de un nuevo reporte estadístico, requeriría ejecutar transformaciones en el código fuente del sistema, consumiendo tiempo y recursos a la entidad. El vínculo dependiente a los desarrolladores se debe a la incapacidad del sistema de permitir al administrador configurar el tipo de reporte estadístico que quiere obtener, con los datos detallados y apariencia que desea mostrar, de manera que facilite el proceso de toma de decisiones con respecto a la administración de contenidos del sistema.

De acuerdo a la situación problemática planteada anteriormente se define el siguiente **problema a resolver**: ¿Cómo generar los reportes de forma que puedan ser transformados según las necesidades de una institución?

Dicho problema se enmarca en el **objeto de estudio** proceso de obtención de reportes en los sistemas generadores de reportes dinámicos, enmarcándose en el **campo de acción** proceso de generación de reportes.

La investigación tiene como **objetivo general**: Desarrollar una herramienta para el sistema REPXOS 3.0 que permita generar los reportes de forma que puedan ser transformados según las necesidades de una institución.

Con el propósito de dar cumplimiento al objetivo planteado se definen un conjunto de **tareas de investigación**:

1. Caracterización de los sistemas informáticos para la creación de reportes estadísticos dinámicos para los repositorios digitales.

2. Descripción de las herramientas necesarias para la implementación de la aplicación.
3. Generación de los artefactos definidos por la metodología seleccionada.
4. Implementación de la aplicación para la creación de reportes estadísticos dinámicos para los repositorios digitales.
5. Realización de las pruebas para validar el correcto funcionamiento del sistema.

El desarrollo de la herramienta informática para la generación de reportes dinámicos basados en Apache Solr para el sistema para repositorios digitales REPXOS 3.0, denominada Sistema de Reportes, propicia a los usuarios interesados en las estadísticas de REPXOS 3.0, conocer de forma rápida y organizada la información asociada a los contenidos de administración del sistema. A través de los reportes estadísticos podrá analizar los comportamientos de los usuarios y su interactividad con la documentación disponible en el repositorio, brindando la oportunidad de cambiar o enfatizar en los procesos más críticos para el usuario final.

Por el carácter independiente de la herramienta puede ser aprovechado para otros sistemas, como los sistemas de negocios de empresas comercializadoras, productoras o de cualquier ámbito en general, siempre que sus estadísticas y datos se encuentren indexados en Solr. Esto posibilitaría junto al estudio de los procedimientos de los clientes, a través de las estadísticas reportadas, definir estrategias de negocios con soporte real e indagar en las tendencias del mercado.

Los métodos científicos utilizados en esta investigación se describen a continuación:

Métodos Teóricos:

Estos métodos permiten estudiar las características del problema que no son observables directamente.

Analítico sintético: Este método que posee una base objetiva de la realidad se ha usado durante la confección del diseño teórico de la investigación. Se realizó un análisis sobre los sistemas generadores de reportes, las diferentes tecnologías y herramientas; dividiendo su estudio en partes para un mejor entendimiento y luego realizar una síntesis específica y concreta de estos elementos.

Modelación: para confeccionar los diagramas generados, a partir de la metodología propuesta, que ayudan a la comprensión de los procesos a desarrollar como parte de la investigación.

Estructura del contenido

El presente trabajo se estructura en introducción, cuatro capítulos, conclusiones, recomendaciones, referencias bibliográficas y glosario de términos.

Capítulo 1. Fundamentación teórica de la Herramienta informática para la generación de reportes dinámicos basados en Apache Solr: agrupa los resultados del estudio de los sistemas generadores de reportes; sus características, así como su importancia en los sistemas de información. Se hace además, un análisis del estado del arte de sistemas generadores de reportes. También se describen las herramientas y tecnologías usadas para el desarrollo del sistema, así como la metodología de desarrollo y el lenguaje de modelado a utilizar.

Capítulo 2. Características de la Herramienta informática para la generación de reportes dinámicos basados en Apache Solr: describe la herramienta propuesta y se realiza el modelo de dominio definiendo los elementos manejados en el sistema junto a las relaciones establecidas entre estos. Se especifican los requerimientos funcionales y no funcionales de la aplicación. Además se representa el diagrama de casos de uso del sistema unido a las descripciones de los casos de uso del mismo.

Capítulo 3. Arquitectura y diseño de la Herramienta informática para la generación de reportes dinámicos basados en Apache Solr: formula el diseño de la herramienta informática para la generación de reportes dinámicos basados en Apache Solr, a partir de los diagramas de clases del diseño y los diagramas de colaboración del diseño. Se describe detalladamente los elementos pertenecientes a estos diagramas. Se muestra una representación de la base de datos a través del modelo físico de datos. Se definen el estilo arquitectónico y los patrones de diseños utilizados para la implementación de la herramienta.

Capítulo 4. Implementación y pruebas de la Herramienta informática para la generación de reportes dinámicos basados en Apache Solr: describe la fase de implementación y pruebas, para ello se realizan los diagramas de componentes y despliegue. Además se determinan los tipos de prueba que se realizan y las técnicas utilizadas, así como los casos de pruebas que se le aplican al sistema con el objetivo de comprobar los errores que pueda tener y comprobar que la propuesta de solución cumple con todas sus especificaciones.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR.

El presente capitulo contiene los resultados del estudio de los sistemas generadores de reportes; sus características, así como su importancia en los sistemas de información. Se hace además, un análisis del estado del arte de sistemas generadores de reportes. También se describen las herramientas y tecnologías usadas para el desarrollo del sistema, así como la metodología de desarrollo y el lenguaje de modelado a utilizar.

1.1. Conceptos fundamentales.

1.1.1. Estadísticas

Muchos han sido los autores que a través de la historia han dado su definición de estadística. En el manual de estadísticas escrito por David Ruiz Muñoz en el 2004 se define como “la ciencia que trata de la recopilación, organización, presentación, análisis e interpretación de datos numéricos con el fin de realizar una toma de decisiones más efectiva” (1). Para Chacón esta se define como "la ciencia que tiene por objeto el estudio cuantitativo de los colectivos"; otros la definen como la expresión cuantitativa del conocimiento dispuesta en forma adecuada para el escrutinio y análisis. La más aceptada, sin embargo, es la de Minguez, que define la Estadística como "La ciencia que tiene por objeto aplicar las leyes de la cantidad a los hechos sociales para medir su intensidad, deducir las leyes que los rigen y hacer su predicción próxima". (3)

Tomando en consideración los conceptos citados se pudiera concluir que la estadística es la ciencia encargada de la recopilación e interpretación de los datos de una muestra. Entre varias de sus definiciones se expone como objetivo ayudar en el proceso de toma de decisiones o explicar circunstancias de algún fenómeno o estudio.

Importancia de la estadística

Kendall y Buckland (citados por Glas, Stanley; 1980) aseguran que “la importancia que tiene está relacionada con el área o áreas en las que se puede aplicar, debido a que está presente en todas las áreas del saber” (3):

- En las ciencias naturales: se emplea con profusión en la descripción de modelos termodinámicos complejos (mecánica estadística), en física cuántica, en mecánica de fluidos o en la teoría cinética de los gases, entre otros muchos campos.
- En las ciencias sociales y económicas: es un pilar básico del desarrollo de la demografía y la sociología aplicada.
- En economía: suministra los valores que ayudan a descubrir interrelaciones entre múltiples parámetros macro y microeconómicos.
- En las ciencias médicas: permite establecer pautas sobre la evolución de las enfermedades, los enfermos y el grado de eficacia de un medicamento, etcétera.

Se pudiera concluir acerca de la estadística para la informática, pero más bien su verdadera importancia es la estadística con la informática. La informática permite a la estadística llevarla a lo más profundo de las investigaciones trayendo consigo resultados más precisos y en el menor tiempo posible, además que lleva las estadísticas a todas las demás áreas de las ciencias convirtiéndola en una herramienta informática para la deducción y el descubrimiento científico.

Estadísticas del DSpace

Las estadísticas del sistema REPXOS 3.0 que está implementado sobre la herramienta DSpace 4.2, son obtenidas mediante consultas Solr. Solr es un motor de búsqueda de código abierto basado en la biblioteca Java, y es usado en DSpace para lograr las funcionalidades estadísticas y búsquedas. Solr a su vez ofrece una interfaz de administración que permite hacer consultas contra los índices y analizarlas. Las estadísticas de uso y descarga, son obtenidas a partir de /Solr/statistics. Para obtener las estadísticas del sistema REPXOS 3.0 se crean consultas Solr a través de facetas definidas para especificar qué se va a adquirir de los datos indexados. Las facetas son consideradas parámetros para buscar por el motor de búsqueda Solr, en estas se define el formato en que se quieren obtener los datos, teniendo como opciones xml, json, ruby y php. Las consultas a través de las facetas permiten extraer la información necesaria y exacta para la realización de reportes.

DSpace 4.2 es un software de código abierto que provee herramientas para la administración de colecciones digitales y comúnmente es usada como solución de repositorio institucional. Soporta una gran variedad de datos, incluyendo libros, tesis, fotografías, videos, datos de investigación y

otras formas de contenido. DSpace cuenta con algunas características de gran importancia que permiten y facilitan su uso en una gran cantidad de instituciones. Entre ellas se encuentra su distribución bajo Licencia *Berkeley Software Distribution* (BSD, por sus siglas en inglés); que es multiplataforma, por lo que se adapta a gran número de Sistemas Operativos; es basado en tecnología web y es un sistema adaptable. (4)

1.1.2. Reporte

Un reporte es una noticia o informe que brinda información con algún propósito. En el ámbito de la informática es un informe que organiza y exhibe la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios. A través de los reportes se refleja el comportamiento de los diferentes componentes que integran cualquier sistema de información de una empresa o institución, permitiendo el reconocimiento, control y monitoreo de las diversas problemáticas existentes en la misma. De esta forma se apoya en la correcta toma de decisiones (5). Los reportes en la informática como informes de cierta actividad o proceso, posibilitan presentar las estadísticas de cualquier ámbito o entorno. La manera fácil para descifrar o analizar problemas descritos en diversos medios permite indagar en respuestas y soluciones.

Reportes dinámicos

En la actualidad la forma de generar los diferentes tipos de reportes se restringe a las facilidades que presten las herramientas propias de las bases de datos o herramientas creadas con este fin. Se presenta como problema, que el soporte para la adaptación de los reportes a las necesidades de los usuarios sea solo por personal técnico capacitado, lo que implica gastos a las empresas en tiempo y costos (6). Además, en toda organización existe demasiada información, la cual necesita ser procesada y manipulada para que el personal de toma de decisiones pueda abstraer los datos que necesitan en el momento que lo requieran. Esta es la causa por la que necesitarían un reportador eficiente y fácil de manejar, sin necesidad de un mayor conocimiento técnico, es decir un reportador dinámico (5).

En resumen, un reportador dinámico es una herramienta informática, que basada en las estadísticas que desea el usuario, es capaz de transformar o cambiar la estructura del reporte o la consulta de los datos a obtener según las necesidades que tiene el usuario.

1.1.3. Sistemas Generadores de Reportes

En todo sistema de información es fundamental contar con una herramienta adicional cuyo objetivo sea la de generar reportes. Los sistemas generadores de reportes tienen en las bases de datos su principal fuente de alimentación, pues a partir de la información almacenada en la misma se realizan consultas para obtener información en forma de reporte (6).

A diferencia de las consultas tradicionales, con los generadores de reportes se puede definir el diseño y la forma en que la información será visualizada. Este se compone por un diseñador de reportes y por un motor de generación de reportes. El primero se encarga de brindar las herramientas para diseñar la apariencia del informe. El segundo accede a la fuente de datos, obtiene los necesarios y los introduce en el diseño de la plantilla del reporte con la que luego se puede realizar ciertas operaciones como: imprimir, enviar por correo electrónico o guardar en un archivo. Los sistemas de generación de reportes incluyen además varias funcionalidades que permiten la introducción de gráficos en los reportes y la definición del formato final en el que son generados (6).

Los sistemas generadores de reportes permiten una mayor manipulación de la información que se debe mostrar. Brindan la posibilidad de exponer los datos de mayor relevancia para la institución, mediante una interfaz amigable en forma de reporte. Estos sistemas utilizan modos más expresivos como esquemas y dibujos de fácil comprensión para cualquier usuario. Siempre buscando que el proceso de toma de decisiones sea cada vez más sencillo de llevar, proporcionándoles una herramienta viable a los interesados.

1.2. Sistemas Generadores de reportes dinámicos

A nivel nacional e internacional existen diversos sistemas generadores de reportes dinámicos, entre los cuales se encuentran algunos como los descritos a continuación.

1.2.1. Sistemas Generadores de reportes dinámicos a nivel internacional.

Gestion22 Reporting

Es una herramienta que permite la creación de los Reportes Dinámicos con el formato y presentación que se desee, encargándose automáticamente de todas las tareas de actualización, clasificación y distribución sin intervención humana (7).

Beneficios:

- Actualización automática de sus Reportes en Excel.
- Información precisa y oportuna para la toma de decisiones.
- Resguarda el nivel de calidad, actualización y seguridad de su información.
- Reducción de errores automatizando tareas de manipulación humana de información.

Características

- Acceso a cualquier tipo de Sistema y Bases de Datos, tanto locales como remotas: el módulo se encargará automáticamente de todas las tareas de actualización, clasificación y distribución. De esta manera, cada persona podrá acceder a los reportes que realmente necesita y solo a ellos. Todo esto sin el menor costo de manipulación humana, con una total seguridad de acceso, manteniendo todo en orden y resguardando el nivel de calidad de la información.
- Seguridad de la Información: los reportes dinámicos están protegidos dentro de un repositorio. Solo el usuario que está autorizado, podrá modificar el formato de los reportes, pero nunca sus datos.

Ventajas de la herramienta

- Configuración personal: brinda información exclusiva para cada persona o rol.
- Posee múltiples esquemas de clasificación.
- Permite la búsqueda, agregación y transferencia automática de los datos necesarios para generar los reportes, sin importar donde se encuentren.

Desventajas para la solución

- Es una herramienta propietaria.
- No permite la utilización de consultas Solr al índice de datos indexados del DSpace.

Generador de reportes Ad-Hoc.

Denominado Generador de Reportes Dinámicos (GRD), el cual le proporciona al usuario un mecanismo adicional para el diseño y ejecución de reportes según sus necesidades. Con esta herramienta se pueden diseñar reportes parametrizables o estáticos, los cuales podrán ser ejecutados por los usuarios del sistema a los cuales se les asigne. (2)

Características

- Parametrización de las diferentes consultas sobre la cual se van a basar los reportes.
- Administrador de parámetros de reportes.
- Generación de reportes dinámicos.

Desventajas para la solución

- Es una herramienta propietaria.
- No permite la utilización de consultas Solr al índice de datos indexados del DSpace.

Software Gestor de Reportes

Permite efectuar informes y generar consultas desde los sistemas de CAS-CHILE (empresa chilena de producción de software de gestión pública) por medio de la utilización de cualquier tipo de base de datos independiente de su estructura y diseño. Esta particularidad otorga notables ventajas con respecto a otros sistemas que se encuentran desarrollados sólo para una base de datos determinada. Esta aplicación cuenta con dos secciones, la primera corresponde a la ventana de selección de base datos con la cual se trabajará y la segunda, donde se diseñará el informe. En esta última se encuentran propiedades y herramientas para seleccionar campos, insertar textos e imágenes. (8)

Características técnicas

- Ambiente de Ejecución: Sistema operativo Windows XP o superior.
- Modalidad de Ejecución: Escritorio.
- Lenguaje de Ejecución: Visual Basic 6.0.
- Motor de la Base de Datos: SQL Server, desde 2005 o superior.

Desventajas para la solución

- Es una herramienta propietaria.
- No permite la utilización de consultas Solr al índice de datos indexados del DSpace.

1.2.2. Sistemas Generadores de Reportes Dinámicos en Cuba

GeReport: Sistema de Gestión de Reportes Dinámicos.

GeReport es una herramienta destinada al diseño, generación y configuración de los reportes relacionados con los datos históricos almacenados en una fuente de datos. Además, luego de contar con toda la información del reporte es posible exportarlo como imagen y en los formatos HTML, PDF y Excel. Para la interacción con las aplicaciones externas, el sistema implementa un servicio que expone los metadatos de los reportes para poder utilizarlos sin restricciones de lenguajes y plataformas. Se utiliza en la Universidad de Cienfuegos por analistas y programadores del Grupo de Estudios y Desarrollo de Ingeniería y Sistemas, perteneciente a la Facultad de Ingeniería. (6)

Características

- Gestión de los reportes.
- Diseño, generación y configuración del reporte.
- Importación de reportes desde sitios externos.
- Exportación de reportes a los formatos HTML, PDF, Excel y a otros como PNG o JPG.

Desventajas para la solución

- No permite la utilización de consultas Solr al índice de datos indexados del DSpace.

1.2.3. Sistemas Generadores de Reportes en la Universidad de las Ciencia Informáticas

Generador Dinámico de Reportes (GDR)

El Generador Dinámico de Reportes del Centro de Tecnologías de Gestión de Datos (DATEC) de la UCI es una herramienta multiplataforma desarrollada con tecnologías web. Este sistema permite la creación y edición de informes/reportes extrayendo datos de una amplia gama de gestores de bases de datos, entre los que se encuentran PostgreSQL, MySQL, Oracle, SQLite y MS-SQL. Los reportes son creados basado en el siguiente estándar: encabezado del documento, encabezado de página, cuerpo, pie del documento y pie de página. (9)

Características

- Permite cargar en una vista estándar los modelos de bases de datos con los que se desea trabajar a través de una interfaz amigable.
- Personaliza en dependencia de las necesidades del informe deseado.
- Brinda la posibilidad de agregar gráficas e imágenes que muestren una representación clara de los datos a valorar.
- Cuenta con un diseñador de consultas que permite crear, editar y ejecutar consultas SQL de forma gráfica.

Desventajas para la solución

- No permite la utilización de consultas Solr al índice de datos indexados del DSpace.

1.2.4. Análisis de los sistemas generadores de reportes dinámicos en la actualidad

El estudio de los sistemas generadores de reportes, permitió afirmar que ninguno cumple con las necesidades de reportes estadísticos del Repositorio Institucional REPXOS 3.0. Este sistema está desarrollado sobre la plataforma DSpace que utiliza un servidor Solr para indexar los log de uso de la aplicación, permitiendo el acceso a sus estadísticas mediante consultas Solr. Los sistemas generadores de reportes dinámicos analizados poseen muchas de las características y requisitos que debe cumplir el sistema que dará solución a los objetivos propuestos. Algunas características de utilidad que resaltan son la parametrización de las diferentes consultas sobre la cual se van a basar los reportes, la gestión, diseño y configuración de los reportes y su exportación a formato físico PDF. El carácter propietario de varios de estos sistemas impide aprovechar las funcionalidades y características que estos poseen mediante su utilización por una vía económica para la UCI.

Para el uso de sistemas propietarios la UCI deberá consumir recursos que no están destinados para este fin. Además que ninguno de estos sistemas independientemente de la licencia que poseen, o su fin comercial, trabajan mediante consultas Solr o interactúan con el servidor Apache Solr para la realización de reportes. Todos son sistemas que realizan consultas a base de datos relacionales, mediante consultas SQL. Asimismo evidenció la necesidad de desarrollar un nuevo generador de reportes dinámicos estadísticos, basado en Apache Solr y que permita la obtención de los reportes estadísticos del REPXOS 3.0.

1.3. Metodologías, tecnologías, herramientas y lenguajes utilizados

El generador de reportes dinámicos basado en Apache Solr fue desarrollado utilizando tecnologías, metodologías, herramientas y lenguajes que se analizaron e investigaron para dar solución a los objetivos propuestos. A continuación se explican las razones de selección de varios de ellos.

1.3.1. Metodologías de Desarrollo de Software

OpenUp

OpenUp es una metodología de Proceso Unificado que aplica enfoques iterativos e incrementales dentro de un ciclo de vida estructurado. Utiliza una filosofía ágil que se enfoca en la naturaleza de colaboración para el desarrollo de software. Se basa en RUP (Rational Unified Process) y contiene el conjunto mínimo de prácticas que ayudan a un equipo de desarrollo de software a realizar un producto de alta calidad, de una forma eficiente. Este tipo de metodología es un proceso mínimo y suficiente, lo que significa que solo el contenido fundamental y necesario es incluido Por lo tanto no provee lineamientos para todos los elementos que se manejan en un proyecto, pero tiene los componentes básicos que pueden servir de base a procesos específicos. La mayoría de los elementos de esta metodología están declarados para fomentar el intercambio de información entre los equipos de desarrollo. Mantiene un entendimiento compartido del proyecto, sus objetivos, alcance y avances. (10)

Ventajas:

- Es una metodología ágil.
- Se puede adaptar con otros procesos.
- Permite disminuir las posibilidades de riesgo.
- Permite descubrir errores tempranos a través de ciclos iterativos.

Desventajas:

- A veces omite contenido que puede ser de interés en el proyecto.
- Se espera que cubra un amplio sistema de necesidades en los proyectos en un plazo muy corto.

Crystal

Crystal da vital importancia a las personas que componen el equipo de un proyecto, y por tanto sus puntos de estudio son: (11)

- Aspecto humano del equipo
- Tamaño de un equipo (número de componentes)
- Comunicación entre los componentes
- Distintas políticas a seguir
- Espacio físico de trabajo.

Crystal aconseja que el tamaño del equipo sea reducido. La mejora de la comunicación entre los miembros del equipo del proyecto y el mismo lugar de trabajo disminuye el coste de la comunicación.

Ventajas

- Es apropiada para entornos ligeros
- Al estar diseñada para el cambio, experimenta reducción de costo.
- Presenta una planificación más transparente para los clientes.
- Se definen en cada iteración cuales son los objetivos de la siguiente.
- Permite tener una muy útil realimentación de los usuarios.

Desventajas

- Delimita el alcance del proyecto con el cliente.

RUP

Proceso de desarrollo Unificado (RUP por sus siglas en inglés) es un modelo de software que permite el desarrollo de software a gran escala. Este mediante un proceso continuo de pruebas y retroalimentación, garantiza el cumplimiento de ciertos estándares de calidad. El proceso de desarrollo constituye un marco metodológico que define en términos de metas estratégicas, objetivos, actividades y artefactos (documentación) requerido en cada fase de desarrollo. Esto permite enfocar esfuerzo de los recursos humanos en términos de habilidades, competencias y capacidades a asumir roles específicos con responsabilidades bien definidas. RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización. (12)

Características

RUP junto con el Lenguaje Unificado de Modelado UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos. Usa un enfoque iterativo (mini-proyectos) que es una secuencia de pasos incrementales (versiones) (13).Las características esenciales de la metodología RUP son tres: dirigida por casos de uso, iterativos e incrementales y centrados en la arquitectura.

Ventajas:

- Un proceso de software hecho a la medida para ser publicado y hacerlo accesible para todo el equipo del proyecto.
- Un proceso de software configurable, para satisfacer necesidades específicas de un proyecto.
- Una definición común del proceso que puede ser compartida por todo el equipo de desarrollo, ayudando a asegurar una comunicación clara y sin ambigüedades entre los miembros del equipo.
- Ofrece a cada usuario, un filtrado personalizado de la definición del proceso publicado, acorde con su rol dentro del proyecto.

Desventajas

- Método pesado.
- En proyectos pequeños, es posible que no se puedan cubrir los costos de dedicación del equipo de profesionales necesarios.

Análisis de los resultados de las metodologías

A partir del estudio de las metodologías, se seleccionó una que permitiera llegar al producto o software requerido con la documentación técnica que lo soporte. A pesar de que el equipo de desarrollo consta de solo 2 personas RUP es la metodología seleccionada para dar solución a los objetivos propuestos. Esto se debe a que puede ser adaptable a las necesidades del proyecto, unifica todo el equipo de desarrollo de software y optimiza su comunicación. RUP provee a cada miembro de una aproximación al desarrollo de software con una base de conocimiento de acuerdo a las necesidades específicas del proyecto, además hace énfasis en la documentación precisa. Los

autores del presente trabajo conocían la metodología seleccionada, evitando gastos de tiempo en su aprendizaje. El proyecto Repositorio Institucional donde se desarrolla la herramienta Sistema de Reportes, emplea esta metodología. Sí se decidiera realizar mejoras al sistema utilizando otro equipo de desarrollo, no se produciría pérdida de tiempo y recursos como al aprender una metodología diferente.

1.3.2. Lenguaje de Programación

Groovy

Groovy es un lenguaje de programación orientado a objetos implementado sobre la plataforma Java. Tiene características similares a Python, Ruby, Perl y Smalltalk. Usa una sintaxis muy parecida a Java, comparte el mismo modelo de objetos, de hilos y de seguridad. Desde Groovy se puede acceder directamente a todas las API existentes en Java. El bytecode generado en el proceso de compilación es totalmente compatible con el generado por el lenguaje Java para la Máquina Virtual de Java (JVM por sus siglas en ingles), por tanto puede usarse directamente en cualquier aplicación Java. Todo lo anterior unido a que la mayor parte de código escrito en Java es totalmente válido en Groovy hacen que este lenguaje sea de muy fácil adopción para programadores Java; la curva de aprendizaje se reduce mucho en comparación con otros lenguajes que generan bytecode para la JVM, tales como Jython o JRuby. (14)

Características

- La lectura de archivos en Groovy es igual o más fácil que la lectura de archivos en Java.
- Tiene soporte nativo para manejo de colecciones.
- Posee soporte nativo para expresiones regulares y rangos.
- Permite la sobrecarga de operadores.
- Tiene soporte nativo de lenguajes de marcado, como XML.

Desventajas

- Se le considera menos conveniente para depurar que otros lenguajes, sus mensajes de error pueden ser crípticos.

Groovy es el lenguaje escogido para dar solución al desarrollo del sistema generador de reportes dinámicos basados en Apache Solr. Reúne características de Java, Ruby, Python y SmallTalk en un

solo lenguaje, convirtiéndolo en un lenguaje sencillo de usar. El cambio de lenguaje para los desarrolladores es relativamente pequeño, lo cual permite crear aplicaciones de forma inmediata. Groovy es un lenguaje dinámico, fácil de aprender en poco tiempo si se tienen previos conocimientos de programación orientada a objetos y debido a que el código es mucho más expresivo, hay menos posibilidades de errores en el código. Agrupa parte de las mejores características de los demás lenguajes, erradicando muchas de sus desventajas. La elección de este lenguaje permitió ganar en tiempo y recursos empleados para su conocimiento.

1.3.3. Marco de Trabajo: Grails

Grails es un marco de trabajo para aplicaciones web libre desarrollado sobre el lenguaje de programación Groovy (el cual a su vez se basa en la plataforma Java). Grails pretende ser un marco de trabajo altamente productivo siguiendo paradigmas tales como convención sobre configuración o no te repitas, proporcionando un entorno de desarrollo estandarizado y ocultando gran parte de los detalles de configuración al programador. Grails está construido sobre cinco fuertes pilares; el lenguaje Groovy para la creación de propiedades y métodos dinámicos en los objetos de la aplicación; Spring para los flujos de trabajo e inyección de dependencias; Hibernate para la persistencia; SiteMesh para la composición de la vista y Ant para la gestión del proceso de desarrollo. En general, Grails toma las mejores prácticas de cada uno de los frameworks para formar un marco de trabajo estable, robusto, sencillo de usar y de fácil mantenimiento. (15)

1.3.4. Entorno de Desarrollo Integrado (IDE por sus siglas en Ingles): Eclipse V 4.2

Eclipse es el entorno de desarrollo definido para desarrollar el sistema. Permite la refactorización y la actualización/instalación automática de código. Es completamente neutral a la plataforma y al lenguaje. Soporta diversos lenguajes como: Java, C/C++, Python, Eiffel, PHP, Ruby, y C#. La plataforma Eclipse, cuando se combina con las herramientas de desarrollo Java (JDT por sus siglas en ingles), ofrece muchas de las características que cabría esperar de un IDE (Entorno Integrado de Desarrollo) de calidad comercial: editor con sintaxis coloreada, compilación incremental, un depurador que tiene en cuenta los hilos a nivel fuente, un navegador de clases, un controlador de ficheros/proyectos, e interfaces para control estándar de código fuente.

Eclipse también incluye varias características únicas, como la refactorización de código, la actualización/instalación automática de código (mediante *Update Manager*), una lista de tareas, integración con la herramienta de construcción de Jakarta: Ant. Añade soporte para un nuevo tipo de editor, una vista, o un lenguaje de programación. Es fácil de usar, con el API y los bloques de construcción que proporciona. (14)

1.3.5. Sistema Gestor de Base de Datos: PostgreSQL V 9.1

PostgreSQL es un sistema de gestión de bases de datos objeto-relacional, distribuido bajo licencia BSD y con su código fuente disponible libremente. Esta herramienta utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema. Un fallo en uno de los procesos no afectará el resto y el sistema continuará funcionando (15).Sus características técnicas la hacen una de las bases de datos más robustas y potentes del mercado. La estabilidad, potencia, robustez, facilidad de administración e implementación de estándares han sido las características que más se han tenido en cuenta durante su desarrollo. PostgreSQL funciona muy bien con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema (16).

Características principales

- Soporta distintos tipos de datos: además del soporte para los tipos base, también soporta datos de tipo fecha, monetarios, elementos gráficos, datos sobre redes, cadenas de bits, etc. También permite la creación de tipos propios.
- Incluye herencia entre tablas, por lo que a este gestor de bases de datos se le incluye entre los gestores objeto-relacionales.
- Tiene completa documentación.
- Posee Licencia BSD.
- Multiplataforma: está disponible en casi cualquier Unix (34 plataformas en la última versión estable).
- Diseñado para ambientes de alto volumen: usa una estrategia de almacenamiento de filas llamada Acceso concurrente multiversión (MVCC por sus siglas en inglés) para conseguir una mucha mejor respuesta en ambientes de grandes volúmenes.

1.3.6. Contenedor de aplicaciones web: Apache Tomcat V 7.0

Tomcat es un servidor web. En sus inicios existió la percepción de que el uso de Tomcat de forma autónoma era solo recomendable para entornos de desarrollo y entornos con requisitos mínimos de velocidad y gestión de transacciones. Hoy en día ya no existe esa percepción y Tomcat es usado como servidor web autónomo en entornos con alto nivel de tráfico y alta disponibilidad. Dado que Tomcat fue escrito en Java, funciona en cualquier sistema operativo que disponga de la máquina virtual Java. (17)

Tomcat es sencillo de configurar para empezar a funcionar, y dispone de abundante documentación para su aprendizaje y soporte como servidor de aplicaciones web.

1.3.7. Lenguaje de Modelado Unificado (UML)

Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. UML(Lenguaje de Modelado Unificado) ofrece un estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio, funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y compuestos reciclados. Se utiliza para definir un sistema, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en el desarrollo de software debido a su gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como el Proceso Unificado Racional o RUP), pero no específica en sí mismo qué metodología o proceso usar. (18)

1.3.8. Herramienta CASE para el modelado: Visual Paradigm V 8.0

Es una herramienta de modelado que emplea UML, diseñada para apoyar a los arquitectos de sistema, desarrolladores y diseñadores para acelerar el proceso de análisis y diseño de aplicaciones complejas. También ofrece un diseño centrado en casos de uso y enfocado al negocio que genera un software de mayor calidad; el uso de un lenguaje estándar común para todo el equipo de desarrollo que facilita la comunicación; un modelo y código que permanece sincronizado en todo el ciclo de desarrollo. Se encuentra disponible en múltiples plataformas y permite integrarse en los principales IDE. (19)

Además, soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite dibujar todos los tipos de diagramas de clases, generar código desde diagramas y generar documentación. Es una aplicación que posee todas las soluciones para emplear la metodología RUP, seleccionada para el proceso de desarrollo de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr.

1.3.9. Tecnologías

Bootstrap V 3.0

Es un framework desarrollado por Twitter que simplifica el proceso de creación de diseños web combinando CSS y JavaScript. La mayor ventaja es que se puede crear interfaces que se adapten a los distintos navegadores con la propiedad denominada diseño adaptativo. Los diseños creados con Bootstrap son simples, limpios e intuitivos, dándole agilidad a la hora de cargar y adaptarse a otros dispositivos (20).

Características

- Se integra perfectamente con las principales librerías Javascript, por ejemplo Jquery.
- Ofrece un diseño sólido usando LESS y estándares como CSS3 y HTML5.
- Es un framework ligero que se integra de forma limpia en el proyecto actual.
- Dispone de distintos diseños predefinidos con estructuras fijas a 940 píxeles de distintas columnas o diseños fluidos.
- Todas las clases CSS están contenidos en un solo archivo y permite una mejor personalización mediante mejoras, herencias y especificidad (21).

Bootstrap en su versión 3.0 permitió agilizar el diseño de las vistas de la herramienta. Esto se debe a la utilización de componentes y servicios creados por la comunidad web, tales como: HTML5, CSS3, jquery UI, LESS entre otros. Además de ser una herramienta sencilla y ágil en la construcción de interfaces.

CCS V 3.0

Hoja de estilo en cascada o CSS (*Cascading Style Sheets,* por sus siglas en inglés) es un lenguaje de hojas de estilos creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos y su presentación y es imprescindible para crear páginas web complejas (22).

Dada la utilización de Bootstrap 3.0 en la aplicación, la utilización de CSS en su versión 3 es inherente a este, asimismo presenta características propias de la versión, algunas presentadas a continuación.

Características

- Atributo gradiente de colores en borde: posibilidad de definir un gradiente de color en el borde de los elementos, en un atributo no estándar de Firefox.
- Bordes redondeados: define la curvatura del borde del elemento con el atributo border-radius.
- Múltiples imágenes de fondo: consigue que un elemento de la página tenga varias imágenes de fondo a la vez, con CSS básico y con características propias de la versión 3.0.
- Crea sombras con el atributo box-shadow, aplica sombras a los elementos de la página, sin usar imágenes y javascript de forma extra.

HTML V 5.0

El Lenguaje de Marcado de Hipertexto HTML (*HyperText Markup Language,* por sus siglas en inglés) es una colección de estándares para el diseño y desarrollo de páginas web. Esta colección representa la manera en que se presenta la información en el explorador de internet y la manera de interactuar con ella. (23)

HTML 5 permite una mayor interacción entre las páginas web y contenido media, así como una mayor facilidad a la hora de codificar el diseño básico. La versión 5 permite un diseño más común de las páginas web alrededor del mundo, para llegar a un estándar de etiquetas que realicen las mismas tareas de manera más rápida y eficiente. Tiene un nuevo diseño para páginas web, reflejado en las etiquetas <header>, <footer>, <nav>, <section>, <article> las cuales están destinadas a remplazar la necesidad de tener un para cada parte de la página, y en cambio, tener etiquetas específicas para ello. HTML 5 es empleada en el presente trabajo por la utilización de Bootstrap 3.0.

Jquery V 1.8

Es un framework Javascript que ofrece una infraestructura para la creación de aplicaciones complejas del lado del cliente. Con jquery se obtiene ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax. Jquery permite simplificar la manera de interactuar con los documentos HTML, manipular el árbol DOM (*Document Object Model*, por sus siglas en ingles), manejar eventos y desarrollar animaciones. (24)

Características

- Selección de elementos DOM.
- Interactividad y modificaciones del árbol DOM, incluyendo soporte para CSS 1-3.
- Permite la realización de eventos.
- Permite manipulación de la hoja de estilos CSS.
- Permite la realización de efectos y animaciones.
- Proporciona animaciones personalizadas.
- Utilidades varias como obtener información del navegador, operar con objetos y funciones para rutinas comunes.

Jquery permitió ahorrar tiempo y recursos en el desarrollo del software. Posee complementos con funcionalidades de uso común en las vistas de los sistemas web, lo hacen un framework flexible y rápido para el desarrollo web. Además los errores son resueltos rápidamente apoyándose en una comunidad de soporte.

Javascript V 1.2

JavaScript es un lenguaje interpretado, utilizado en la construcción de páginas Web, con una sintaxis muy semejante a Java y a C. Al contrario de Java, no se trata de un lenguaje orientado a objetos propiamente dicho, sino que está basado en prototipos, pues las nuevas clases se generan clonando las clases base (prototipos) y extendiendo su funcionalidad.

JavaScript es un lenguaje que diferencia entre mayúsculas y minúsculas. Se puede encerrar las expresiones con una serie de caracteres especiales. Estos caracteres se denominan operadores y sirven tanto para encerrar expresiones como para realizar trabajos con ellas, como operaciones

matemáticas o de texto. JavaScript como lenguaje es usado por el framework Jquery versión 1.8 en el proyecto.

Características

- Imperativo y estructurado: compatible con gran parte de la estructura de programación del lenguaje C.
- Tipado dinámico: como en la mayoría de lenguajes de scripting, el tipo está asociado al valor, no a la variable.
- Objetual: está formado casi en su totalidad por objetos.
- Evaluación en tiempo de ejecución: incluye la función “eval” que permite evaluar expresiones como cadenas en tiempo de ejecución
- Prototípico: usa prototipos en vez de clases para el uso de herencia.

Conclusiones del Capítulo

El estudio de los conceptos asociados a las estadísticas y reportes permitió conocer sus principales características y beneficios. La caracterización de sistemas generadores de reportes dinámicos en la actualidad, proporcionó un conjunto de conocimientos acerca de sus rasgos y propiedades fundamentales, los cuales no satisfacen todas las necesidades y condiciones requeridas. La descripción de las herramientas, lenguajes y tecnologías definidos por los autores permitió la familiarización de los elementos del ambiente de desarrollo, además de adquirir los conocimientos esenciales para poder utilizarlos en la construcción de la propuesta de solución.

CAPÍTULO 2. **CARACTERÍSTICAS DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR**

En el presente capítulo se describe la herramienta propuesta y se realiza el modelo de dominio definiendo los elementos manejados en el sistema junto a las relaciones establecidas entre estos. Se especifican los requerimientos funcionales y no funcionales de la aplicación. Además se representa el diagrama de casos de uso del sistema unido a las descripciones de los casos de uso del mismo.

2.1. Descripción de la herramienta propuesta

Actualmente el sistema REPXOS permite la creación de reportes estáticos, lo cual provoca que una vez instalado el repositorio en una institución, la consulta para obtener las estadísticas siempre estará predeterminada en los reportes. Es por ello que se hace necesario desarrollar una herramienta para el sistema REPXOS que provea un mecanismo de generación de reportes dinámicos y que a su vez se integre a la arquitectura base del DSpace. Este último hace uso de un servidor Solr que permite la búsqueda y la obtención de estadísticas, a partir del procesado de los logs del sistema.

La herramienta que se propone, Sistema de Reportes, aunque se integre a DSpace funciona como un sistema independiente, por lo que se requiere la gestión de la seguridad (control los usuarios y su acceso a la aplicación). Para garantizar este requerimiento cada persona tiene su usuario y contraseña, y en dependencia de los permisos que tenga asignados, podrá acceder a las funcionalidades del sistema. La generación de reportes es la parte principal, por lo que permitirá la obtención de reportes a partir de las estadísticas que brinda DSpace. Para poder generar los reportes se hace necesaria la creación de modelos que serán predefinidos por el administrador del sistema. En estos modelos se crean consultas Solr donde el administrador indica los campos y parámetros que son necesarios para la construcción de dicha consulta, en dependencia del contenido del reporte que desee obtener. Uno de los elementos fundamentales para construir la consulta es el servidor Solr al que se desee conectar, por ello el sistema brinda la posibilidad de su gestión; pudiendo en cualquier momento añadirlos, consultarlos, editarlos o eliminarlos.

Para la obtención de reportes los usuarios del sistema utilizarán los modelos predefinidos anteriormente, llenando los campos obligatorios establecidos en dicho modelo. Una vez creados los reportes estos podrán ser consultados o eliminados. De manera general el sistema resuelve la problemática presentada y está en correspondencia con los requisitos del cliente. Permite la creación de reportes dinámicos utilizando el motor de búsqueda Solr, que se encuentra en DSpace.

2.2. Modelo de Dominio

El Proceso de desarrollo Unificado (RUP, por sus siglas en inglés) con el fin de esclarecer el entorno del problema define en su primera fase de desarrollo la realización de un Modelo de Casos de Uso del Negocio. Cuando los procesos del negocio no se encuentran bien identificados, RUP propone realizar un Modelo de Dominio. El Modelo de Dominio tiene como objetivo comprender y describir solamente las clases más importantes dentro del contexto en el cual se desempeña el software, con el propósito de sentar las bases del entendimiento del desarrollo y no para definirlo completamente (18).

El sistema no tiene los procesos de negocio bien definidos, lo que ha propicia un modelado del dominio en lugar del negocio. El objetivo del modelado del dominio es capturar, comprender y describir las clases más importantes dentro del contexto del sistema. El Modelo del Dominio, junto al Diccionario de Clases del Dominio, ayuda a los usuarios, clientes, desarrolladores y otros interesados a utilizar un vocabulario común. En la figura 1 se muestra el Diagrama de Clases del Dominio. Por su parte, el Diccionario de Clases del Dominio describe textualmente las clases identificadas durante el modelado del dominio del problema. Este diccionario sirve como glosario de términos y se muestra a continuación:

- Administrador: Hace referencia a la persona que posee el rol de administrador, este tiene todos los permisos para realizar cualquier cambio dentro del sistema Herramienta informática para la gestión de reportes dinámicos basado en consulta solr.
- Modelo de Reporte: representa un modelo que posee un conjunto de parámetros previamente establecidos por el administrador que constituyen la guía para realizar el reporte.

- Reporte: tipo de contenido que representa un informe que organiza y exhibe la información necesitada por el usuario. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar.
- Servidor: hace referencia a todos los servidores a los cuales se conecta el sistema Herramienta informática para la gestión de reportes dinámicos basado en consulta solr, poseen un conjunto de datos necesarios para representarlos.
- Usuario: representa a todas las personas autenticadas en el sistema y que acceden al sistema con el fin de crear o acceder a ver los reportes.

Modelo de Dominio

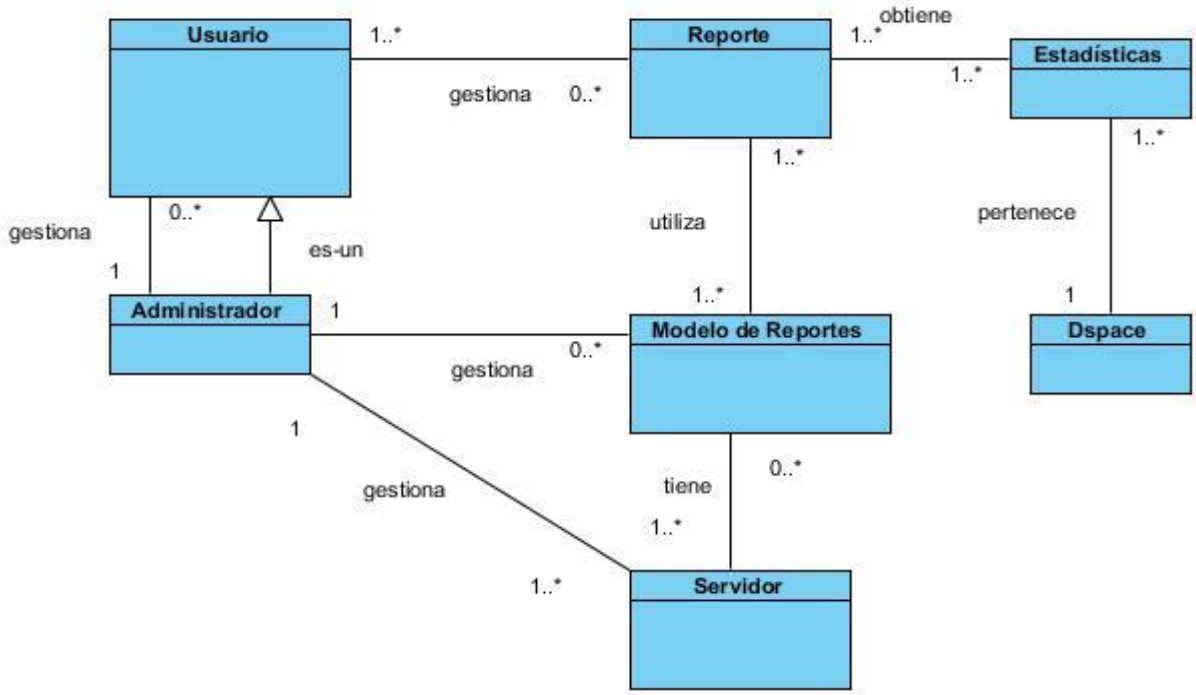


Figura 1 Modelo de Dominio de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. (Fuente: elaboración propia)

2.2.1. Descripción del modelo de dominio

El administrador al acceder al sistema puede gestionar todos los elementos: modelo de reportes, reportes, servidores y usuarios. Todos estos pueden ser creados, modificados y eliminados, exceptuando la modificación de los reportes, debido a que la opción de modificar de los reportes solapa la operación de crearlos. La modificación de un reporte es la creación de otro. Los modelos de reportes se realizan mediante la interfaz, insertando el nombre del modelo y los campos que se requieran para las estadísticas solicitadas. Luego se guarda el modelo de reporte en la base de Datos. El usuario solo tendrá los permisos de visualizar los modelos de reportes previamente confeccionados por el administrador. El usuario puede seleccionar a su conveniencia el modelo deseado y así elaborar el reporte. El usuario deberá llenar los campos requeridos para la obtención del reporte, teniendo como permiso además de su gestión, la opción de visualizarlo. El reporte quedará guardado en la base de datos permaneciendo accesible para otros usuarios, brindando la opción de exportarlo a formato PDF. Para la obtención de las estadísticas de los reportes, cada reporte realizado será una consulta Solr. Esta será enviada al servidor seleccionado al cual se debe conectar el sistema. En este caso las estadísticas que consumirá la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr, será las del REPXOS 3.0 implementado sobre el DSpace 4.2.Por lo tanto el servidor escogido será el de DSpace.

2.3. Especificación de requisitos de software

La ingeniería de requisitos ayuda a los ingenieros de software a entender el problema en cuya solución trabajarán. Incluyen el conjunto de tareas que conducen a comprender cuál será el impacto del software sobre el negocio, que es lo que el cliente quiere y como interactuarán los usuarios finales con él software (36).

Los requisitos se clasifican en dos tipos; los requisitos funcionales y los requisitos no funcionales. Los requisitos funcionales describen lo que el sistema debe hacer. Mientras que los requisitos no funcionales son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento (25) .

2.3.1. Especificación de los requisitos funcionales

Los requerimientos funcionales (RF) son capacidades o condiciones que el sistema debe cumplir, define qué es lo que el sistema debe hacer, así como las funciones que será capaz de realizar. Describen las transformaciones que el sistema realiza sobre las entradas, para producir salidas (25). En la tabla 1 se muestran a continuación los requisitos funcionales del sistema.

Tabla 1 Requisitos Funcionales de la Herramienta informática para la gestión de reportes dinámicos basados en consulta Solr (fuente: elaboración propia)

Nº	Nombre	Descripción	Nivel de Acceso
RF1	Autenticar usuario	El usuario debe identificarse para acceder al sistema, introduce usuario y contraseña.	Administrativo
RF2	Crear usuario	Se adiciona un nuevo usuario en el sistema.	Administrativo
RF3	Modificar usuario	Se actualiza el usuario creado.	Administrativo
RF4	Eliminar usuario	Se elimina el usuario creado.	Administrativo
RF5	Buscar usuario	Se busca el usuario por el nombre de usuario.	Administrativo
RF6	Crear Modelo de Reporte.	Se crea un Modelo de Reporte.	Administrativo
RF7	Modificar un Modelo de Reporte.	Se modifica un Modelo de Reporte.	Administrativo
RF8	Eliminar un Modelo de Reporte.	Se elimina un Modelo de Reporte.	Administrativo

RF9	Insertar Servidor.	Se agrega un servidor	Administrativo
RF10	Modificar Servidor.	Se actualiza un usuario.	Administrativo
RF11	Eliminar Servidor.	Se elimina un servidor.	Administrativo
RF12	Crear reporte	Se crea un reporte a partir de un modelo de Reporte.	Público
RF13	Listar reportes	Se muestra una lista de los Reportes.	Público
RF14	Eliminar reporte	Se elimina el reporte creado.	Administrativo
RF15	Mostrar Reporte.	Se muestra un reporte seleccionado, previamente creado.	Público
RF16	Buscar Reporte	Se busca el Reporte por el título de Reporte.	Público
RF17	Buscar Modelo de Reporte	Busca el Modelo de Reporte por el título del Modelo de Reporte.	Público
RF18	Listar Usuarios	Se muestra una lista con todos los usuarios.	Administrativo
RF19	Listar Modelos de Reportes	Se muestra una lista con todos los modelos de reportes.	Público
RF 20	Listar Servidores	Se muestra una lista con todos los servidores.	Administrativo
RF21	Exportar reporte a formato físico PDF.	Se exporta a formato físico PDF un reporte seleccionado.	Público

2.3.2. Especificación de los requisitos no funcionales

Los requerimientos no funcionales (RNF) son propiedades o cualidades que el producto debe tener, debe pensarse en propiedades que hacen al producto atractivo, usable, rápido o confiable. Son aquellos requerimientos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de este, como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento (25).

Usabilidad

El sistema proporciona una interfaz sencilla, con facilidad de uso para usuarios con conocimientos mínimos de informática. Los grupos de botones y vínculos están organizados por funcionalidad, con el objetivo de facilitar al usuario la interacción con el sistema. Los mensajes para interactuar con los usuarios y los de error son lo suficientemente informativos, en idioma español y no revelan información interna. Los colores utilizados en el desarrollo de Sistema de Reportes son visualmente agradables y sigue la estética del sistema REPXOS 3.0.

Tiene mecanismos de búsqueda de usuarios, modelos de reporte y reportes. El menú y títulos de páginas se encuentran claros y bien definidos. Los iconos son sugerentes, siguiendo los diseños habituales. Se requieren menos de 3 clics para llegar a cualquier contenido en el sistema. La consistencia está dada debido a que los botones y funciones se muestran de igual forma en todas las vistas. Las secciones persistentes están visibles en el mismo lugar en las páginas necesarias.

Seguridad

El sistema debe garantizar la protección de los datos almacenados. Para ello se establecerán dos roles (administrador y usuario) garantizando que cada persona tenga acceso solamente a las funcionalidades según los permisos de su rol. Se aplicará en la autenticación de usuario la encriptación de la contraseña.

Confiabilidad

La base de datos será actualizada en tiempo real en caso de actualización, modificación o inserción de datos. Se utilizarán mecanismos de validación de los datos para que la información sea guardada de forma correcta y completa.

Soporte

Se documentará el sistema con un manual de usuario con el objetivo de explicar su uso.

Restricciones de diseño

Se utilizará para la construcción del sistema el lenguaje de programación Groovy y herramientas que se distribuyan bajo licencias libres. Las librerías a usar no deben ser propietarias.

Interfaces de usuario

Según el uso del framework de diseño Bootstrap versión 3.0, para acceder a Sistema de Reportes, debe usarse una versión del navegador Mozilla/Firefox igual o superior a la 27.0 o Internet Explorer igual o superior a la 9.0. Específicamente a partir de la versión 8.0 de Internet Explorer no soporta distintas funcionalidades CSS 3. No se garantiza su correcto funcionamiento en otros navegadores.

Interfaces Hardware

El hardware donde se instalará el sistema debe poseer al menos una interfaz de red cuya velocidad de transferencia iguale o supere los 100 Mbps.

Interfaces Software

El sistema debe integrarse con los siguientes productos de software en estas versiones:

PostgreSQL 8.4: es un sistema de administración de base de datos relacional. Almacena de forma persistente la información necesaria para el funcionamiento del sistema.

Apache Tomcat 7.0: es un servidor HTTP en el cual estará publicado el sistema.

Requerimientos del Hardware

Para la implementación del sistema el ambiente de desarrollo que se utilizó tenía las siguientes propiedades:

- Tarjeta de red.
- 4 GB de memoria RAM.
- Microprocesador Intel Celeron de 2.16 GHz.

- 5 GB de espacio libre en el disco duro.

En un ambiente informático con prestaciones inferiores no se garantiza su correcto funcionamiento.

Requerimientos de Software

Clientes.

Navegador Web que cumpla con los estándares de la World Wide Web Consortium (W3C).

Servidores.

- Servidor de Bases de Datos PostgreSQL versión 8.4.
- Contenedor Web Apache Tomcat versión 7.0.
- Máquina Virtual de Java (JRE Por sus siglas en inglés) 7.0.

2.4. Definición de los actores

Actores del sistema.

Un actor es una agrupación uniforme de personas, sistemas o máquinas externas al sistema que se está desarrollando, se relaciona con éste ya que solicita sus funcionalidades. En Sistema de Reportes interactúan dos actores, los cuales se definen a continuación:

Administrador**:** Establece todos los permisos sobre el sistema.

Usuario**:** Establece los criterios de obtención de las Reportes.

2.5. Diagrama de Casos de Uso de Sistema

El diagrama de casos de uso documenta el comportamiento de un sistema desde el punto de vista del usuario. Por lo tanto, los casos de uso determinan los requisitos funcionales del sistema, es decir, representan las funciones que un sistema puede ejecutar (10). Su ventaja principal es la facilidad para interpretarlos, lo que hace que sean especialmente útiles en la comunicación con el cliente. Observar en la figura 2 el diagrama de casos de uso del sistema.

Definición de los casos de uso del sistema

- Gestionar Usuarios.

- Buscar Usuario.
- Gestionar Modelos de Reportes.
- Buscar Modelos de Reportes.
- Gestionar Reportes.
- Buscar Reporte.
- Gestionar Servidores.
- Autenticar Usuario.
- Exportar a Formato PDF.

Diagrama de casos de Uso

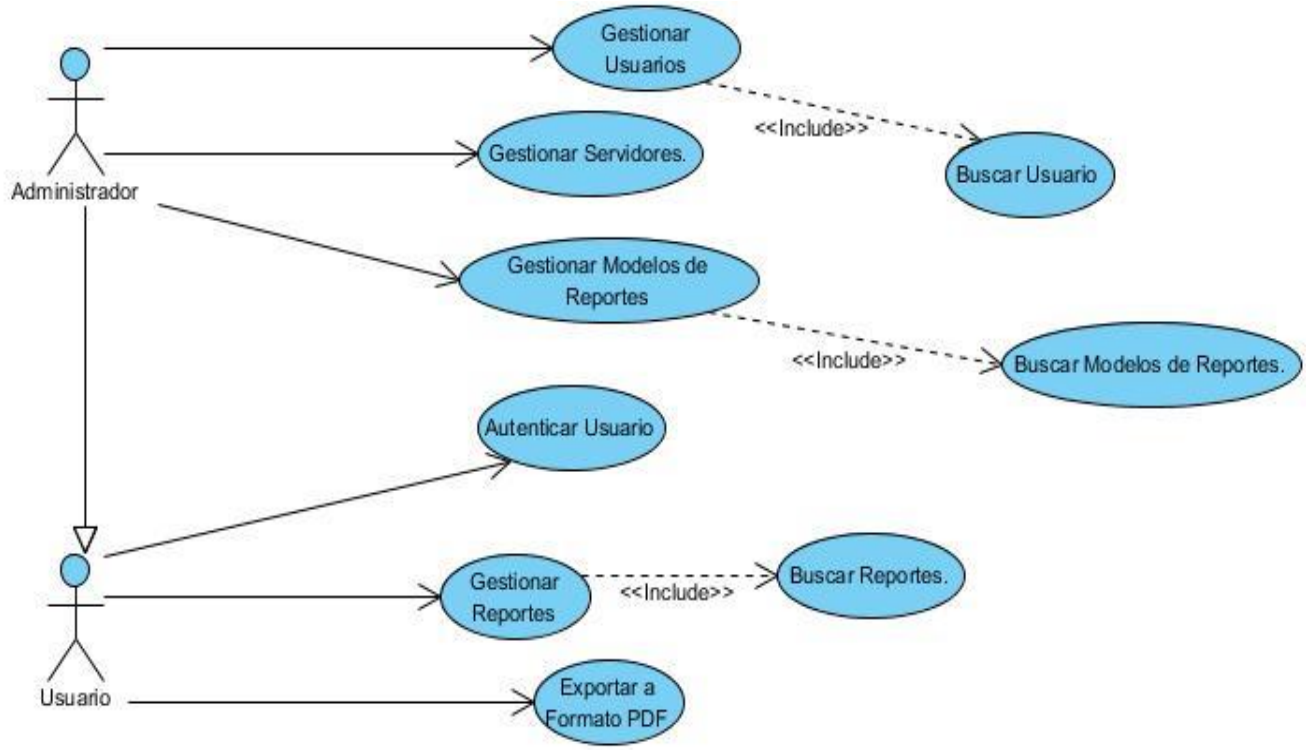


Figura 2 Diagrama de Caso de Usos del Sistema (fuente: elaboración propia).

2.5.1. Descripción del Caso de Uso del Sistema

A continuación, se muestran el resumen de las descripciones correspondiente a los casos de uso identificados anteriormente, para más información consultar el Anexo 1.

Tabla 2 Diagrama de CU Buscar Usuario (fuente: elaboración propia)

Objetivo	Buscar un usuario en la base de datos.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el administrador escoge buscar un usuario en la lista, dado su nombre de usuario para modificarlo o eliminarlo.
Complejidad	Baja
Prioridad	Alta
Precondiciones	El usuario debe estar registrado.
Postcondiciones	Se muestran todos los datos del usuario.

Tabla 2 Diagrama de CU Gestionar Usuarios (fuente: elaboración propia)

Objetivo	Crear los datos de los usuarios, pudiendo en cualquier momento mostrar dichos datos, modificarlos e incluso eliminarlos.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el administrador crea o lista los usuarios teniendo la opción de modificarlos o eliminarlos.
Complejidad	Baja
Prioridad	Alta
Precondiciones	El administrador esta autenticado en el sistema.
Postcondiciones	Usuario creado, modificado o eliminado de la base datos.

Tabla 3 Diagrama de CU Gestionar Modelos De Reportes (fuente: elaboración propia)

Objetivo	Crear en el sistema los Modelos de Reportes, pudiendo en cualquier momento modificarlos o eliminarlos.
Actores	Administrador: (Inicia) Crea, modifica y elimina los Modelo de Reportes en el sistema.
Resumen	El caso de uso inicia cuando el administrador crea o lista los modelos de Reporte, teniendo la opción de modificarlos, eliminarlos y mostrarlos.
Complejidad	Alta
Prioridad	Alta
Precondiciones	Estar el administrador autenticado en el sistema.
Postcondiciones	Es creado, modificado o eliminado el Modelo de Reporte.

Tabla 4 Diagrama de CU Autenticar Usuario (fuente: elaboración propia)

Objetivo	Autenticar usuario en el sistema.
Actores	Usuario
Resumen	El usuario se autentica en el sistema mediante su nombre de usuario y contraseña.
Complejidad	Baja
Prioridad	Alta
Precondiciones	El usuario debe estar registrado en el sistema.

Postcondiciones	El usuario accede al sistema.

Tabla 5 Diagrama de CU Buscar Modelo de Reporte (fuente: elaboración propia)

Objetivo	Buscar un modelo de reporte en la base de datos.
Actores	Administrador
Resumen	El caso de uso se inicia cuando el administrador escoge buscar un modelo de reporte dado su nombre para modificarlo o eliminarlo.
Complejidad	Baja
Prioridad	Alta
Precondiciones	El modelo de reporte debe estar registrado.
Postcondiciones	Se muestran todos los datos del modelo de reporte.

Tabla 6 Diagrama de CU Buscar Reporte (fuente: elaboración propia)

Objetivo	Buscar un reporte en la base de datos.
Actores	Usuario.
Resumen	El caso de uso inicia cuando el usuario escoge buscar un reporte dado su nombre para verlo o eliminarlo.
Complejidad	Baja
Prioridad	Media
Precondiciones	El reporte debe estar creado.

Postcondiciones	Se muestran todos los datos del reporte.

Tabla 7 Diagrama de CU Gestionar Servidores (fuente: elaboración propia)

Objetivo	Gestionar los Servidores a los cuales se va a conectar el sistema, pudiendo en cualquier momento Crear dichos servidores, modificarlos y eliminarlos.
Actores	Administrador
Resumen	El caso de uso inicia cuando el administrador crea un servidor para poder conectar el sistema a este.
Complejidad	Baja
Prioridad	Alta
Precondiciones	Tener los servidores reales a los cuales puede conectarse el sistema.
Postcondiciones	Servidor creado en la base de datos.

Tabla 8 Diagrama de CU Exportar a PDF (fuente: elaboración propia)

Objetivo	Exporta a formato a PDF los reportes que desee el usuario.
Actores	Usuario
Resumen	El usuario después de Mostrar su reporte elige la opción exportar a formato PDF
Complejidad	Baja
Prioridad	Media
Precondiciones	Debe haber un Reporte creado y mostrado.

Postcondiciones	El reporte es mostrado en formato PDF.

Tabla 9 Diagrama de CU Gestionar Reportes (fuente: elaboración propia)

Objetivo	Gestionar los datos de los reportes, pudiendo en cualquier momento crearlos, mostrarlos o eliminarlos.
Actores	Usuario, Administrador
Resumen	EL caso de uso inicia cuando el usuario crea o lista los reportes teniendo la opción de mostrarlos y el administrador de eliminarlos.
Complejidad	Baja
Prioridad	Alta
Precondiciones	Debe estar registrado al menos un modelo de Reporte.
Postcondiciones	Reporte creado, mostrado o eliminado.

Conclusiones del capítulo

Se realizó la descripción de la herramienta propuesta, aportando una posible solución a la problemática planteada. Se especificó el modelo de dominio que permitió definir los conceptos fundamentales del negocio. De esta forma se obtuvo una visión más clara y óptima del entorno sobre el cual se desarrolla el problema a resolver. Al identificar las características con las que debe contar el reportador se obtuvieron los requisitos funcionales y los requisitos no funcionales. Se modeló el diagrama de casos de uso del sistema posibilitando un mejor entendimiento del proceso. Con las especificaciones de los casos de uso del sistema se establecen los flujos básicos de estos y se logra una descripción detallada para una mejor comprensión del sistema que se va a implementar.

CAPÍTULO 3. ARQUITECTURA Y DISEÑO DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR

En el presente capítulo se formula el diseño de la herramienta informática para la generación de reportes dinámicos basados en Apache Solr, a partir de los diagramas de clases del diseño y los diagramas de colaboración del diseño. Se describe detalladamente los elementos pertenecientes a estos diagramas. Se muestra una representación de la base de datos a través del modelo físico de datos. Se definen el estilo arquitectónico y los patrones de diseños utilizados para la implementación de la herramienta.

3.1. Modelo de Diseño

Diagramas de clases del diseño

Es una representación más concreta que el diagrama de clases del análisis, donde se representa la parte estática del sistema y las clases con sus relaciones, junto con otras restricciones relacionadas con el entorno de implementación (26). Cada clase del diseño representa una abstracción con una correspondencia en la implementación. Este modelo brinda una completa idea de lo que es el software y constituye un punto de partida para las actividades de implementación.

En la Figura 3 se representa el diagrama de clases del diseño para el caso de uso Gestionar Modelo de reporte. Solo se muestra el diagrama de este CU porque es uno de los de mayor grado de incidencia tiene sobre el proyecto. Esto se debe, a que a partir de los modelos de reportes es posible realizar los reportes de tanta importancia para el proceso de toma de decisiones. En el diagrama de clase de diseño Gestionar Modelo de Reporte, se muestra la relación y dependencia entre las distintas clases y librerías. La página servidora guarda relación con todo el proceso de gestión de los modelos de reportes, estando vinculada a la clase controladora Modelo, la página principal, las páginas clientes, y a los formularios para la entrada de datos del usuario.

La clase controladora Modelo posee una dependencia con la clase entidad Modelo, que a su vez depende de la clase Servidor. El servidor tiene la dirección y el puerto al que se debe conectar el sistema para obtener las estadísticas y el modelo tiene los parámetros que va a definir la consulta para la confección del reporte. Todos los formularios utilizan las librerías javascript, bootstrap y

jquery. El administrador puede crear, modificar y eliminar en el CU Gestionar Modelo de Reporte que se apoya en el CU incluido buscar Modelo de Reporte. En el Anexo 1 se encuentra la descripción del caso de uso representado en el diagrama de clase de diseño Gestionar Modelo de Reporte.

A continuación, se muestra un ejemplo del diagrama de clase del diseño Gestionar Modelo de Reporte. Para consultar el resto ver Anexo 2.

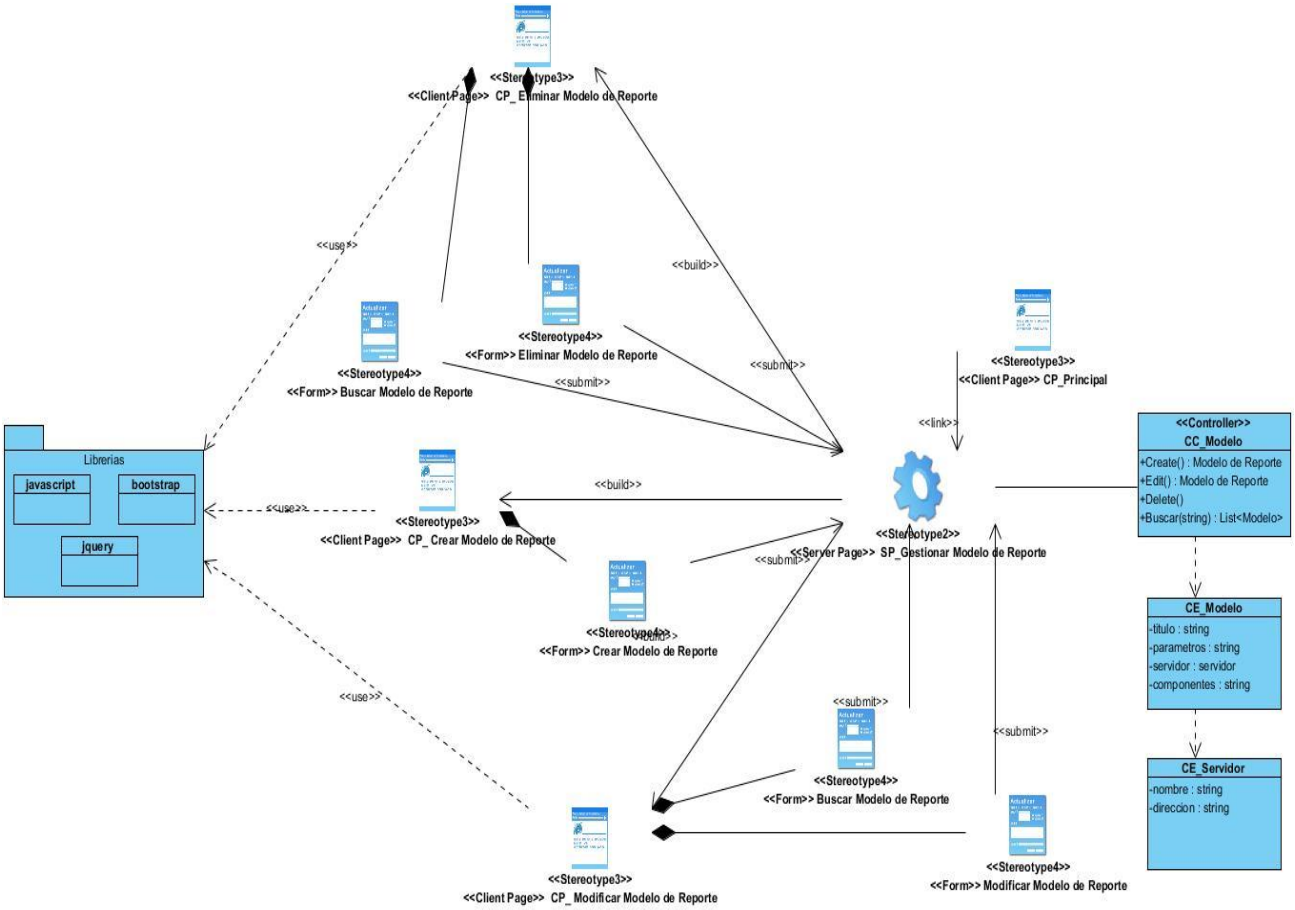


Figura 3 Diagrama de clases del diseño CU Gestionar Modelo de Reporte (fuente: elaboración propia)

Diagramas de Colaboración

El diagrama de colaboración es una forma alternativa al diagrama de secuencia de mostrar un escenario. En este tipo de diagrama se muestran las interacciones entre objetos. Estas son organizadas en torno a los objetos y los enlaces entre ellos, además de añadir mensajes. (25)

A continuación se presenta un ejemplo de diagramas de colaboración, para consultar el resto ver Anexo 3.

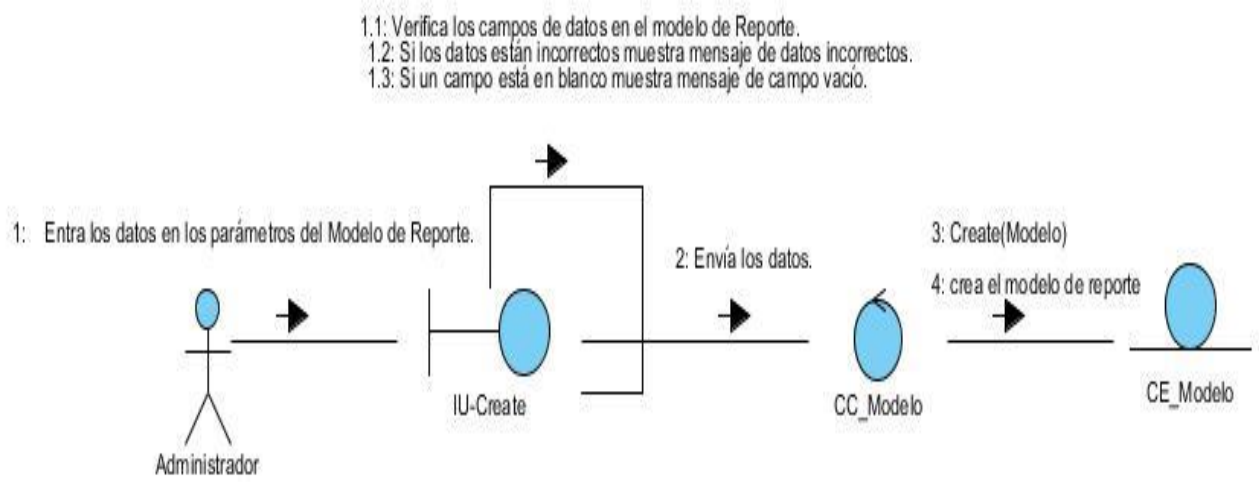


Figura 4 Diagrama de colaboración crear modelo de reporte (fuente: elaboración propia)

El diagrama de colaboración crear modelo de reporte inicia cuando el administrador selecciona la opción crear modelo, del menú principal. La interfaz de usuario Create representa el Modelo de Reporte a confeccionar. En la interfaz se llenan los campos de texto título y parámetros mientras que el servidor es seleccionado de una lista. Al llenarse los campos título y parámetros con los valores correspondientes o si se escoge algún servidor, son validados los datos desde la misma interfaz. Luego una vez validados los datos estos son enviados a través de la clase controladora Modelo. Esta ejecuta la operación Create con la información del modelo y crea un nuevo modelo de reporte almacenándolo en la base de datos. El modelo de reporte se representa en el diagrama de colaboración por la clase entidad Modelo.

3.2. Modelo de datos

Se puede decir que un Modelo de Datos (MD) es un conjunto de conceptos que permiten describir, a distintos niveles de abstracción, la estructura de una base de datos. Además, describe su tipo y la forma en que se relacionan. Es factible pensar que un modelo de datos permite describir los elementos de la realidad que intervienen en un problema dado y la forma en que se relacionan esos elementos entre sí (27).

El modelo físico de datos que se representa en la figura 5 puede considerarse una descripción de la implementación de la base de datos de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. Se definen su estructura de almacenamiento así como la descripción de sus relaciones base.

A continuación se representa en la figura 5 el Modelo Físico de Datos

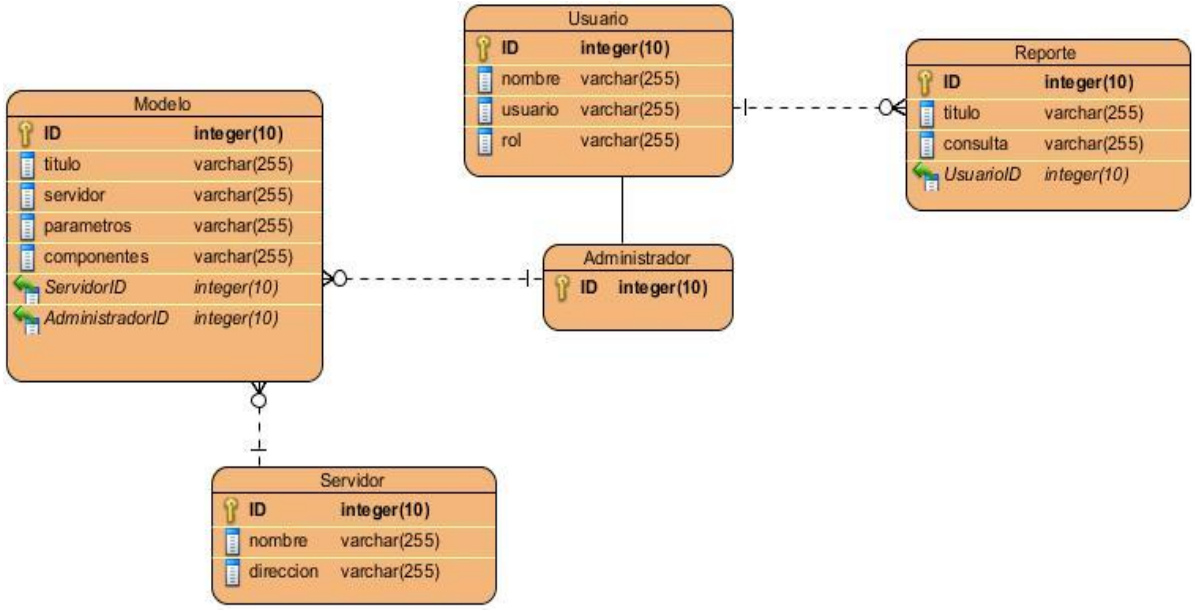


Figura 5 Modelo Físico de Datos del sistema Herramienta informática para la gestión de reportes dinámicos basado en consulta solr. (Fuente: elaboración propia)

En el modelo físico se representan mediante tablas relacionales las entidades del sistema. Estas representan las clases del dominio y las relaciones en la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. Un modelo está compuesto por título, parámetros, componentes y servidor, este último representa el servidor Solr al que debe conectarse el sistema para obtener las estadísticas; por tanto modelo se relaciona con la entidad servidor que se estructura de nombre y dirección. Los modelos son gestionados por el administrador, esto provoca que la entidad modelo posea también las llaves foráneas ServidorID y AdministradorID. Administrador hereda de usuario constituido por nombre, usuario y el atributo rol que determina los permisos del usuario en el sistema. El usuario gestiona los reportes que se comprende de título y consulta, por lo que reporte también incluye en sus atributos la llave foránea UsuarioID, guardando la relación del usuario que lo creó.

Diagrama de paquetes

Un diagrama de paquetes muestra cómo un sistema está dividido en agrupaciones lógicas mostrando las dependencias entre esas agrupaciones. Los paquetes son construcciones UML que permiten organizar los elementos del modelo en grupos, haciendo que los diagramas UML sean más simples y fácil de entender. Dado que normalmente un paquete está considerado como un directorio, los diagramas de paquetes suministran una descomposición de la jerarquía lógica de un sistema. Los paquetes están normalmente organizados para maximizar la coherencia interna dentro de cada paquete y minimizar el acoplamiento externo entre los paquetes. Los paquetes son buenos elementos de gestión. Cada paquete puede asignarse a un individuo o a un equipo, y las dependencias entre ellos pueden indicar el orden de desarrollo requerido (27).

A continuación se muestra del diagrama de paquetes de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr.

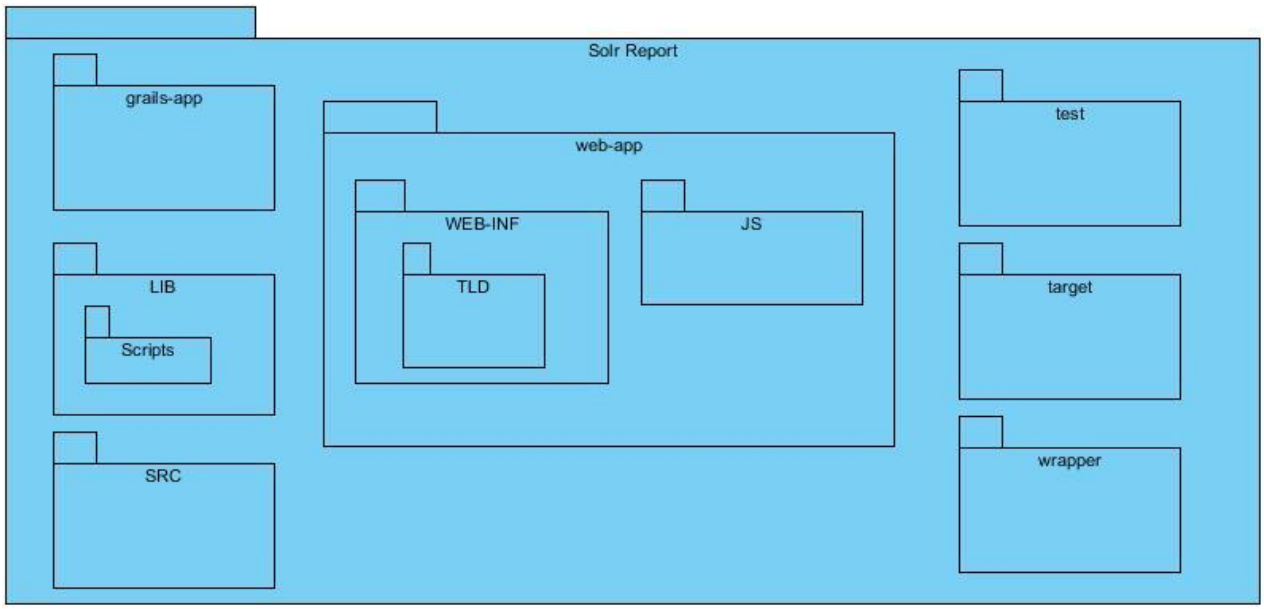


Figura 6 Diagrama de Paquetes del Sistema (fuente: elaboración propia)

El diagrama de paquetes que se muestra en la figura 5 es un esquema lógico de la estructura de la aplicación. La división en carpetas del sistema, representa la distribución en paquetes de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. Aplicación que mantiene la composición de archivos y carpetas de programas que se desarrollan sobre el framework Grails.

3.3. Modelo arquitectónico

Arquitectura

El diseño de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr está basado en la Arquitectura ofrecida por el framework Grails. Está basada en la arquitectura N capas divididas en 3 capas:

· Capa de lógica de negocios.

· Capa de control.

· Capa de presentación.

Capa de lógica de negocios

El núcleo de una aplicación que utiliza una arquitectura de 3 capas es la lógica de negocios. En ella se modelan las entidades y se realizan las tareas necesarias para su manipulación, tal como su creación, almacenamiento y actualización. Dentro del entorno de Grails, esta capa está representada por las clases de dominio, sus métodos relacionados con la persistencia y el acceso a la base de datos. En algunas aplicaciones más robustas, se incluye la capa de servicios, la cual controla el acceso de cualquier programa a la lógica de negocios, facilitando el mantenimiento y promoviendo la reutilización de código. (15)

Capa de control

Las aplicaciones web necesitan una forma de comunicación entre la lógica de negocios y el resultado final mostrado al usuario. En Grails, la capa de control es la encargada de lograr la interactividad entre las clases de dominio (o los servicios) y la capa de presentación a través de clases controladoras. La capa de control tiene la habilidad de recibir peticiones HTTP, procesarlas y emitir un resultado utilizando el mismo protocolo. El contenido puede abarcar desde simples caracteres hasta complejos archivos multimedia.

Capa de presentación

En una aplicación web, el resultado final mostrado al usuario generalmente se representa por medio de interfaces graficas de usuario (GUI, por sus siglas en inglés) en un navegador web. La capa de presentación en Grails viene representada por la tecnología Páginas Servidoras de Groovy (GSP,

por sus siglas en inglés). La capa de control entrega un modelo que un archivo GSP puede manipular mediante código Groovy. Asimismo, la capa de presentación se representa por medio de archivos XML, texto en formato JSON, PDF, e inclusive audio y video.

La Figura 5 muestra un diagrama de la Arquitectura de Grails a través de la interacción entre la capa de lógica de negocios, la capa de control y la capa de presentación. Las flechas indican el sentido en que fluye la comunicación entre capas .Cada capa tiene conocimiento únicamente de la capa inmediata inferior. Esto quiere decir, por ejemplo, que los servicios solo tienen acceso a las clases de dominio las cuales a su vez solo pueden acceder a la base de datos. La base de datos no tiene conocimiento de las capas que acceden a ella.

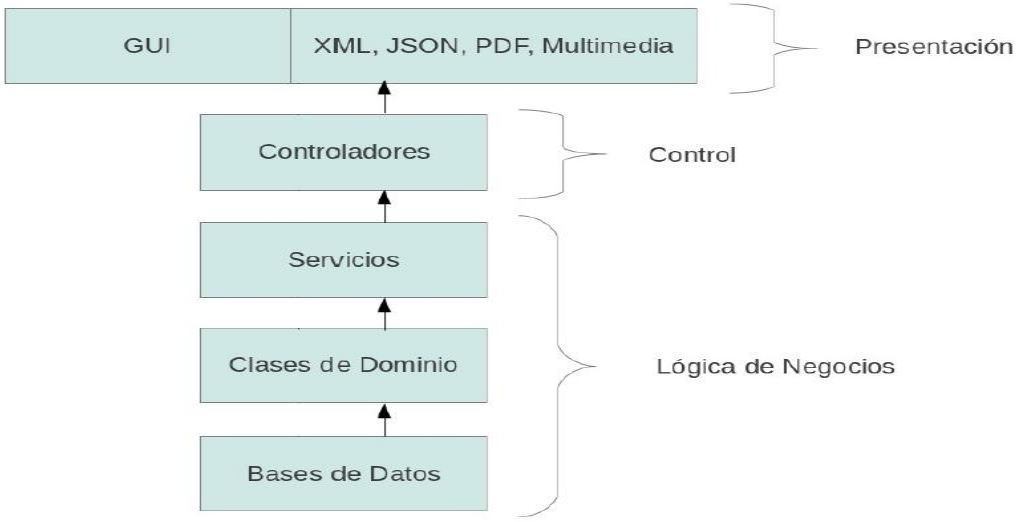


Figura 7 Diagrama de Arquitectura N capas utilizada por Grails (fuente: tomado de (14))

Patrón arquitectónico

Se utilizará en el desarrollo del presente trabajo como patrón arquitectónico el modelo Modelo-Vista-Controlador (MVC). Este patrón establece que los componentes de un sistema de software deben organizarse en tres capas según su misión:

Modelo o capa de datos: contiene los componentes que representa y gestionan los datos manejados por la aplicación. En el caso más típico, los objetos encargados de leer y escribir en la base de datos.

Vista o capa de presentación: los componentes de esta capa son responsables de mostrarle al usuario el estado actual del modelo de datos, y presentarles las distintas acciones disponibles.

Capa de control: contendrá los componentes que reciben las órdenes del usuario, gestionan la aplicación de la lógica de negocio sobre el modelo de datos, y determinan que vista debe mostrarse a continuación. (28)

Descrito anteriormente, en la arquitectura N capas del framework Grails se define muy bien el patrón Modelo-Vista-Controlador. La capa de presentación constituye la vista, la capa de control el controlador y en la capa de lógica de Negocio se encuentra el modelo representando las entidades del dominio. Por lo tanto cualquier aplicación que se desarrolle en el framework Grails, estará implementando el patrón Modelo-Vista-Controlador.

La utilización del patrón MVC permite gracias a la separación de las capas y su nivel de jerarquía un proyecto más escalable. Esto indica la habilidad del proyecto de adaptarse o de crecer continuamente de manera fluida sin perder calidad en los servicios ofrecidos. Este patrón arquitectónico proporciona a los desarrolladores mayor sencillez para agregar múltiples representaciones de datos y la reutilización de componentes. La independencia en el funcionamiento del sistema posibilita un fácil mantenimiento en caso de errores.

3.4. Patrones de diseño utilizados en la herramienta Sistema de Reportes

Un patrón de diseño se define como una solución a un problema que se usa repetidamente en contextos similares, se encarga de identificar, abstraer y nombrar los aspectos elementales de una estructura de diseño, donde los componentes, son las clases y objetos, y sus mecanismos de interacción son mensajes. Ayudan a elegir diseños alternativos que hacen un sistema reutilizable y evitan alternativas que comprometan la reutilización. (18)

Para el desarrollo del trabajo se utilizaron los siguientes patrones de diseño:

- Experto en Información

Es el principio básico de asignación de responsabilidades. Pertenece a los patrones generales de software para asignación de responsabilidades (GRASP, por sus siglas en inglés). Indica, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento) (18).

En el sistema este patrón se pone de manifiesto en el controlador ReporteController, este es el encargado de la creación de la consulta y la realización de la petición al servidor Solr para la obtención de cada reporte. De igual forma en el sistema cada controlador tiene asignada la responsabilidad y conoce toda la información necesaria para la gestión de cada objeto o implementación de cada método correspondiente a la entidad a la que pertenece.

- Creador

El patrón creador identifica quién debe ser el responsable de la creación (o instanciación) de nuevos objetos o clases. Sugiere encontrar clases de objetos que estén vinculadas para hacerlas responsables de la creación de los objetos, manteniendo así un bajo acoplamiento. La nueva instancia deberá ser creada por la clase que tiene la información necesaria para realizar la creación del objeto, o que agregue o contenga instancias del objeto. En el sistema este patrón se pone de manifiesto en los controladores, pues cada uno de ellos está encargado de la creación de nuevas instancias de las clases entidad a la que están asociados.

- Controlador

Un controlador es un objeto responsable del manejo de los eventos del sistema, que no pertenece a la interfaz del usuario. El controlador recibe la solicitud del servicio desde la capa de presentación y coordina su realización delegando a otros objetos. Él uso de este patrón en el sistema esta evidenciado en los controladores existentes que son los encargados de manejar los eventos asociados a las clases entidad a la que pertenecen. En el proyecto el controlador ReporteController.groovy es el encargado del manejo de la entidad Reporte y las vistas asociadas a la misma.

- Bajo acoplamiento

Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De esta forma en caso de producirse una modificación en alguna de ellas, se tiene la mínima repercusión posible en el resto de las clases, potenciando la reutilización, y disminuyendo la dependencia entre estas. En el sistema las clases fueron implementadas con la menor relación posible entre ellas para reducir el impacto de los cambios que puedan realizarse.

- Alta Cohesión

La cohesión es la medida en la que una clase se dedica a realizar solo la tarea para la cual fue creada, de cuan relacionadas y enfocadas están las responsabilidades en ella, delegando las tareas complementarias a otros componentes. Una clase debe de hacer lo que respecta a su entidad, y no hacer acciones que involucren a otra clase o entidad. En el sistema cada controlador tiene definido acciones que respectan únicamente a la entidad vinculada, evitando el trabajo excesivo del mismo.

- DAO

Un objeto de acceso a datos o en inglés Data Access Object (DAO) da resumen y encapsula todos los accesos a la fuente de datos suministrando una interfaz común entre esta y la aplicación. El DAO maneja la conexión con la fuente de datos para obtener y almacenar datos. Este patrón implementa el mecanismo de acceso requerido para trabajar con la fuente de datos. Oculta completamente la fuente de datos de los detalles de la ejecución. Debido a que la interfaz expuesta por el objeto de acceso a datos a los clientes no cambia cuando ocurren cambios en la implementación de las fuentes de datos subyacente, este patrón le permite adaptarse a diferentes sistemas de almacenamiento sin que ello afecte a sus clientes o lo componentes del negocio. (18) Este patrón es usado en el sistema por el framework Hibernate para el acceso a datos.

Conclusiones del capítulo

En este capítulo se llevó a cabo todo el proceso relacionado con el diseño y la arquitectura en general de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. Se representó la lógica del negocio mediante los diagramas de colaboración y de clases del diseño, donde se brinda una visión general del sistema, que constituye parte importante dentro de la etapa de elaboración del software. Se realizó una breve descripción de los patrones de diseño y arquitectónicos utilizados en el desarrollo del sistema. Todo esto tributa que al concluir este capítulo estén creadas las condiciones para el comienzo de la etapa de implementación del sistema.

CAPÍTULO 4. IMPLEMENTACIÓN Y PRUEBA DE LA HERRAMIENTA INFORMÁTICA PARA LA GENERACIÓN DE REPORTES DINÁMICOS BASADOS EN APACHE SOLR

En el presente capítulo se describe cómo se ha desarrollado las etapas de implementación y prueba de la propuesta de solución. Para ello se realiza el modelo de implementación del sistema, separados en los diagramas de componentes y despliegue. Además se determinan los tipos de prueba que se realizan y las técnicas utilizadas, así como los casos de pruebas que se le aplican al sistema con el objetivo de comprobar los errores que pueda tener y comprobar que la propuesta de solución cumple con todas sus especificaciones.

4.1. Modelo de Implementación

El modelo de implementación es comprendido por un conjunto de componentes y subsistemas que constituyen la composición física de la implementación del sistema. Fundamentalmente, se describe la relación que existe entre los paquetes y clases del modelo de diseño, como subsistemas y componentes físicos. (29)

Un diagrama de implementación muestra:

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

4.1.1. Diagrama de componentes

Los diagramas de componentes muestran la estructura de los componentes, incluyendo los clasificadores que especifican los componentes y los artefactos que los implementan. También se pueden utilizar para mostrar la estructura de alto nivel del modelo de implementación en términos de subsistemas de implementación, y las relaciones entre los elementos de implementación (27). En las figuras 7,8 y 9 se muestran tres diagramas de componentes del sistema. Estas representan las distintas capas en las que está dividida la aplicación.

En la capa de presentación se encuentran las vistas del sistemas, representadas por los componentes basados en la tecnología Página Servidora de Groovy (GSP, por sus siglas en inglés). Como componentes están las entidades del modelo representadas en las vistas y los archivos gsp (vistas del sistema) que son manipulados mediante código Groovy por los primeros. Otro

componente son las librerías utilizadas en las vistas del sistema, propias del framework Grails y que trabajan con los archivos gsp directamente.

En la capa de control se encuentran los componentes clases controladoras y las librerias que utilizan estas clases. Las librerias que se representan son las utlizadas directamente por las controladoras del negocio y otras propias de la parte de seguridad del framework spring que implementa grails. Estas últimas usadas por la controladoras de acceso al sistema.

En la capa de lógica de negocios los componentes presentes son las clases del dominio y las librerias de acceso y persistencia de datos. Las librerias presentes son implementadas por el framework Hibernate para el acceso y mapeo de las clases. Otra librería presente es la empleada por las clases de seguridad del dominio, que define roles y permiso en el sistema.

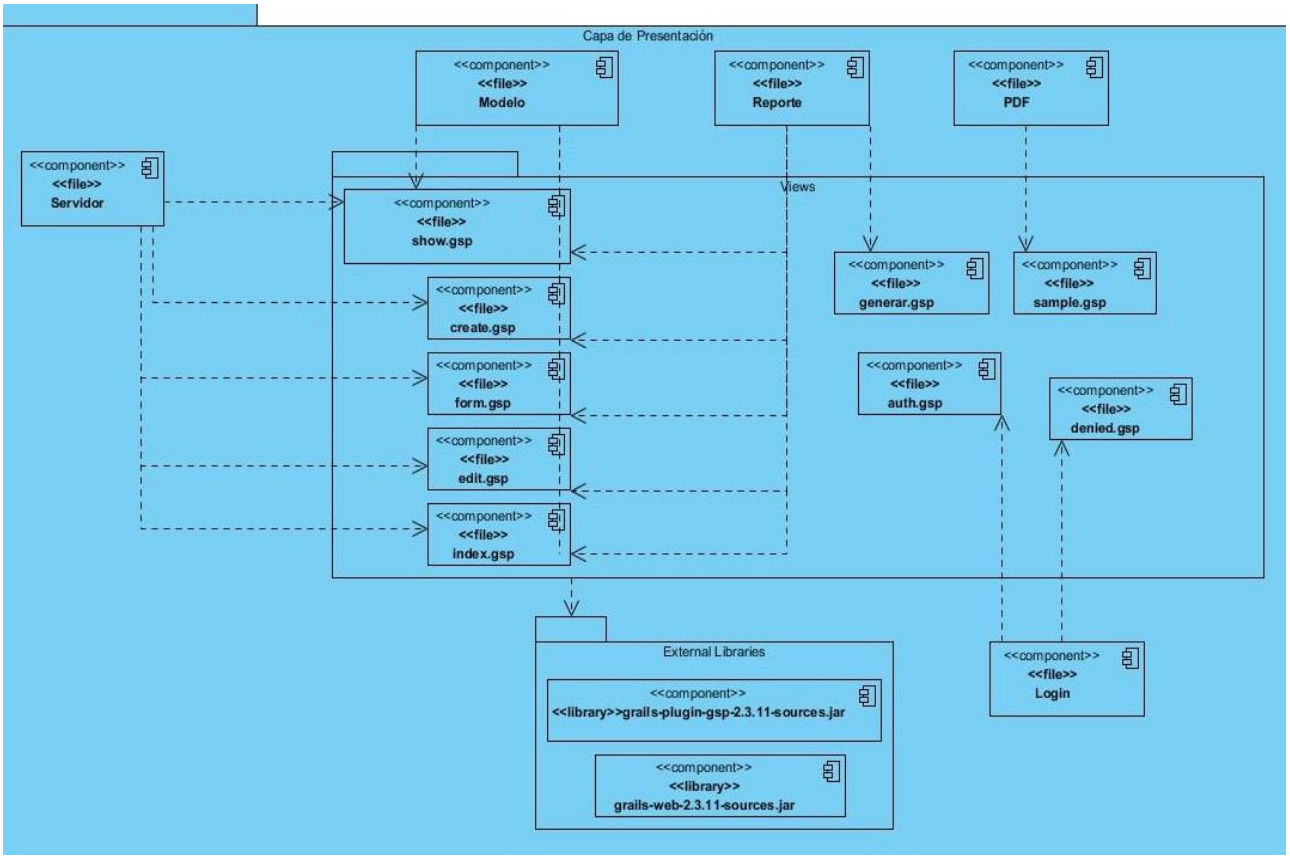


Figura 8 Diagrama de Componentes de la Capa de Presentación (fuente: elaboración propia)

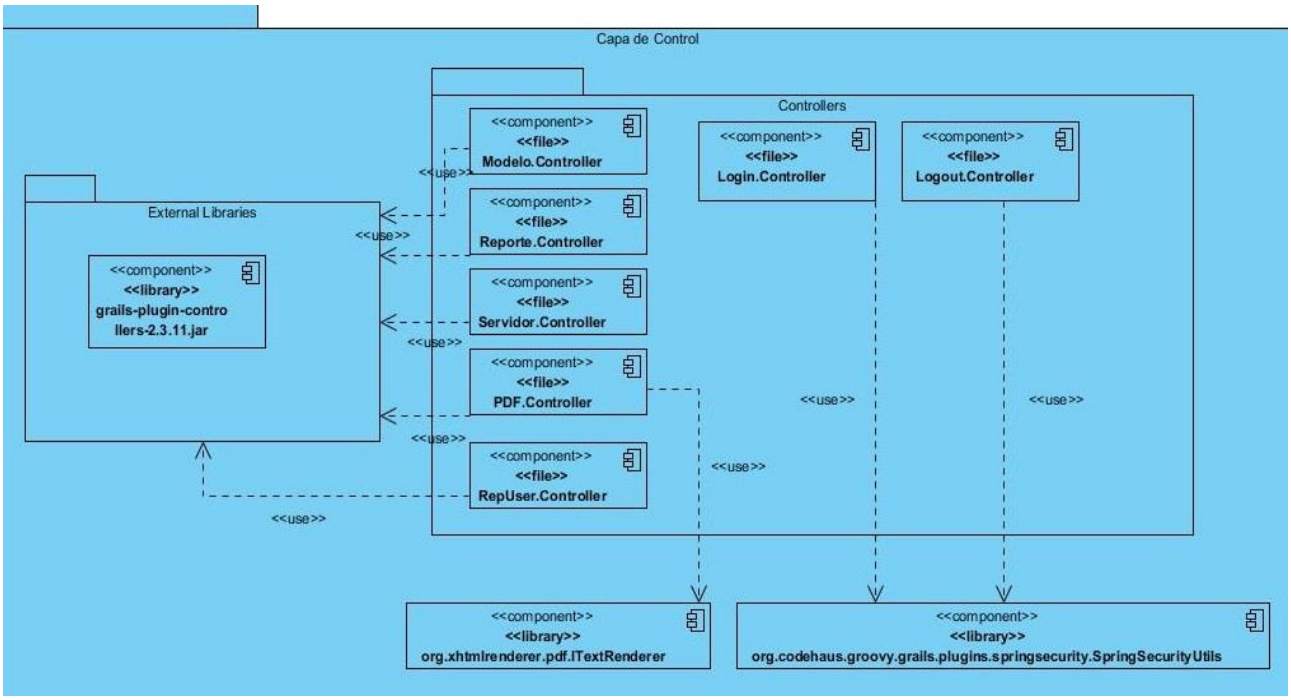


Figura 9 Diagrama de Componentes Capa de Control (fuente: elaboración propia)

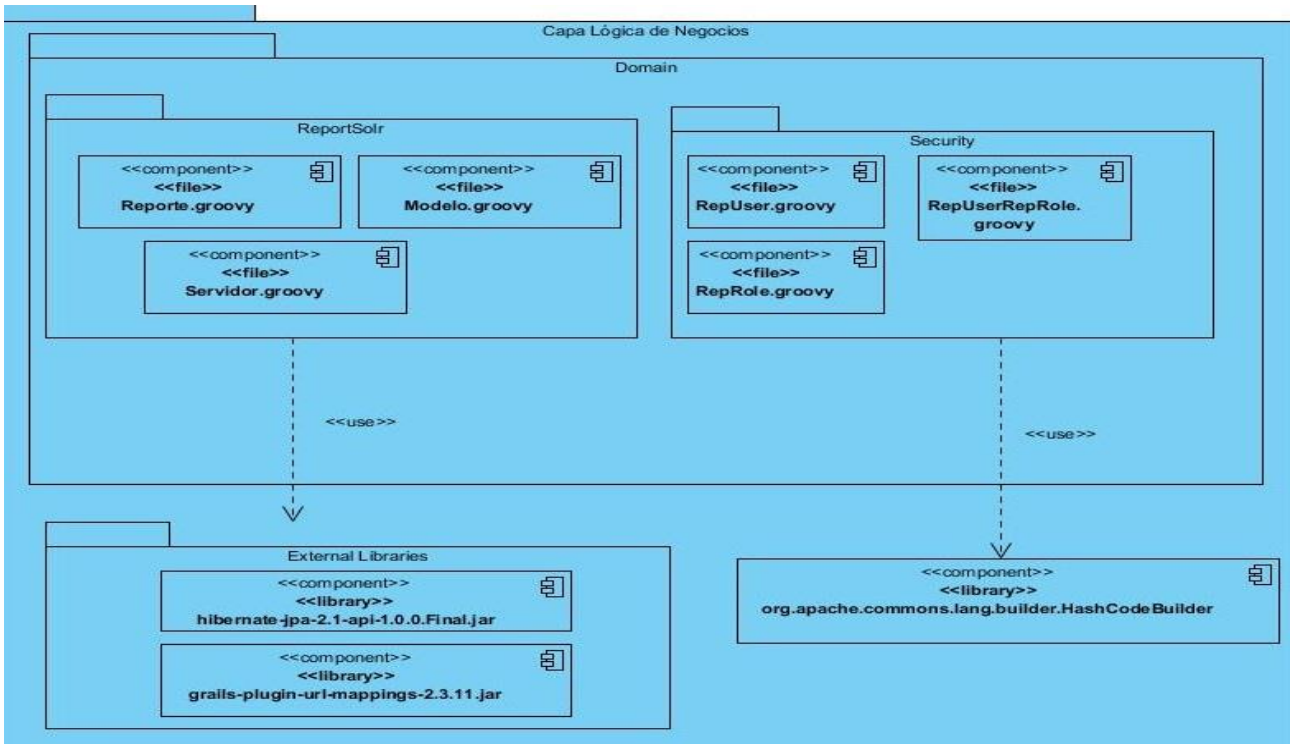


Figura 10 Diagrama de Componentes Capa de Lógica de Negocio (fuente: elaboración propia)

4.1.2. Diagrama de Despliegue

El diagrama de despliegue contiene los nodos que forman la topología hardware sobre la que se ejecuta el sistema, el software necesario para su funcionamiento y los protocolos de comunicación. Al sistema se puede acceder desde un cliente utilizando un navegador web que cumpla con los estándares definidos por la W3C. Este navegador envía o recibe información desde el sistema que se encuentra alojado junto al servidor Apache Solr en el contenedor web Apache Tomcat.

A continuación en la Figura se muestra el diagrama de despliegue correspondiente a la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr.

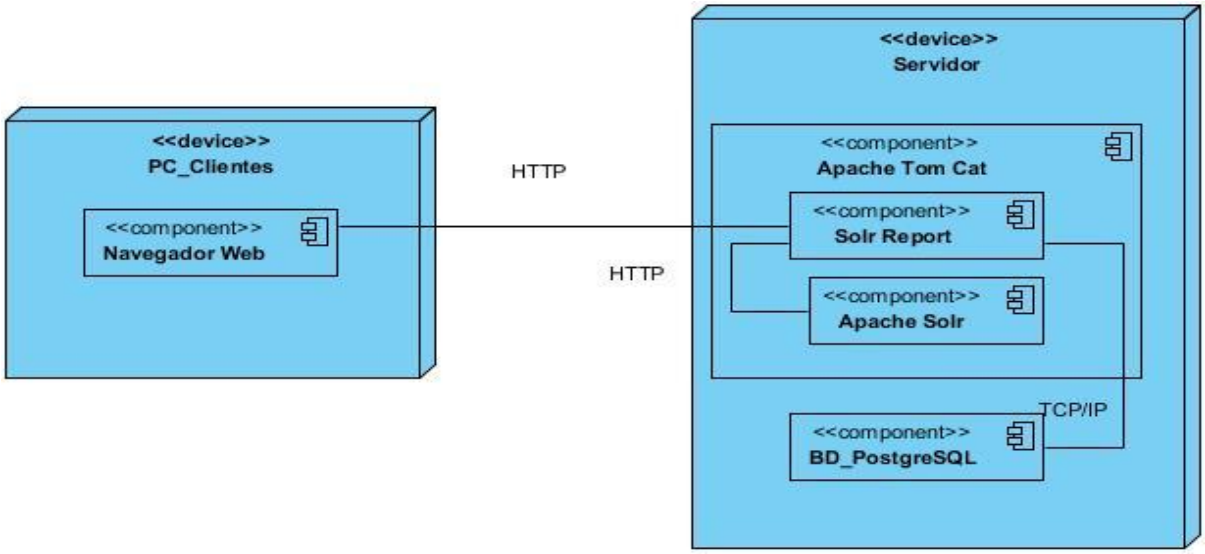


Figura 11 Diagrama de Despliegue del Sistema (fuente: elaboración propia)

Descripción de los Protocolos de Comunicación

Se utilizaron la familia de protocolos de comunicación TCP/IP y HTTP debido a que la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr se desarrolla sobre tecnología Web siguiendo los estándares para la misma.

<<HTTP>>: El protocolo de transferencia de hipertexto (HTTP, *HyperText Transfer Protocol)* es el protocolo usado en cada transacción de la Web (WWW). HTTP define la sintaxis y la semántica que utilizan los elementos software de la arquitectura web (cliente, servidor, proxy) para comunicarse.

<<TCP/IP>>: La familia de protocolos de Internet es un conjunto de protocolos de red en la que se basa Internet y que permite la transmisión de datos entre redes de computadoras. Denominada en

muchas ocasiones conjunto de protocolos TCP/IP en referencia a los dos protocolos más importantes que la componen: Protocolo de Control de Transmisión (TCP) y Protocolo de Internet (IP), que fueron los dos primeros en definirse, y que son los más utilizados de la familia. El TCP/IP es la base de Internet, y sirve para enlazar computadoras que utilizan diferentes sistemas operativos sobre redes de área local (LAN) y área extensa (WAN).

4.2. Estándares de codificación

Al programar en un lenguaje en específico se deben seguir reglas. Estas permitirán que cualquier persona que se desempeñe como codificador de dicho lenguaje, pueda interpretar de manera eficiente la escritura del código. Con el objetivo de mejorar la legibilidad del código del Sistema de Reportes, facilitar el mantenimiento y permitir su compresión rápida por parte de cualquier desarrollador, se siguieron los siguientes estándares de codificación.

- Utilizar una mezcla entre la nomenclatura tradicional en inglés y el castellano para el nombre de clases y métodos, siendo lo más descriptivo posible. Ejemplo: createmodelo, showreporte.
- Los nombres de los métodos, variables y atributos se escribirán en minúscula. Ejemplo: título, insert.
- Los nombres de las clases comenzaran con letra inicial mayúscula. Ejemplo: Reporte.
- Las instancias de entidades se conformaran por el nombre de la entidad seguido de la palabra Instance. Ejemplo: ReporteInstance.
- Comentar los métodos que puedan resultar de difícil comprensión.

4.3. Pruebas de software

La prueba de software es una tarea crucial y a la vez muy desafiante dentro del proceso de desarrollo de software. La prueba permite encontrar errores y problemas del software contra la especificación del mismo y cumple un rol fundamental en el aseguramiento de la calidad del producto. (30)

La prueba de software es un elemento crítico para la garantía de la calidad del software y representa una revisión final de las especificaciones del diseño y de la codificación. Las pruebas son básicamente un conjunto de actividades que pueden ser implementadas en cualquier momento dentro del desarrollo de software (31).

A continuación se muestra en la figura un modelo general del proceso de prueba de software.

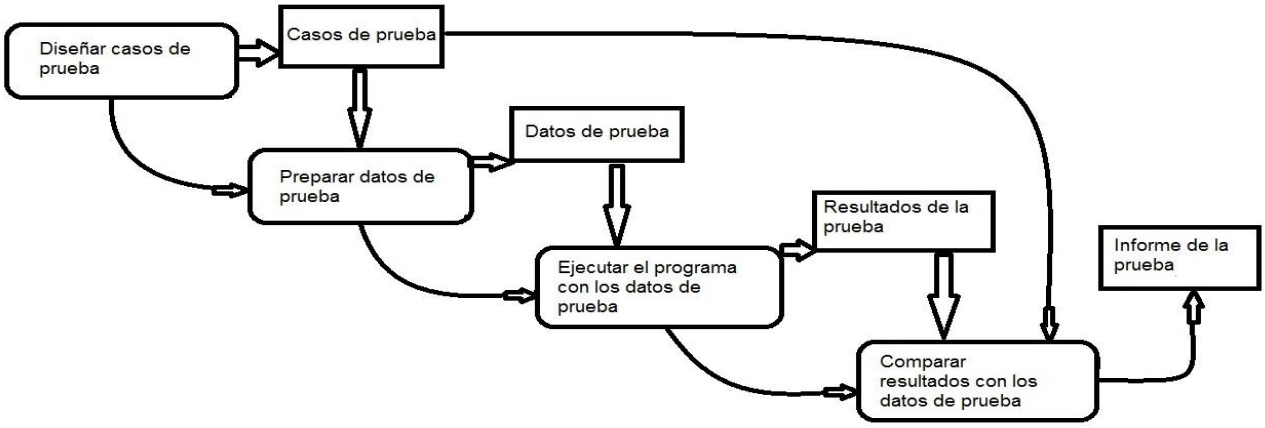


Figura 12 Modelo del proceso de prueba de software. (Fuente: elaboración propia)

En la figura 11, se representa como se puede ejecutar un proceso de pruebas de software. Los casos de prueba son especificaciones donde se define una entrada conocida y una salida esperada por el sistema, junto con el escenario de lo que se prueba. Los datos de prueba son las entradas válidas y no válidas para probar el sistema y en ocasiones pueden generarse automáticamente. La salida de las pruebas sólo puede ser pronosticada por personas que entienden el funcionamiento del sistema, así como sus respuestas en distintos contextos. El informe registra las no conformidades obtenidas de las pruebas.

4.3.1. Estrategia de Prueba

Una estrategia de prueba del software integra los métodos de diseño de casos de pruebas del software en una serie bien planeada de pasos, que desembocará en la eficaz construcción del software. La estrategia proporciona un mapa que describe los pasos que se darán como parte de la prueba. Esto permite conocer cuánto esfuerzo, tiempo y recursos consumirán un enfoque estratégico para la prueba del software (32).

La estrategia de prueba debe indicar los niveles de pruebas (ciclos) que se aplican y la intensidad o profundidad a aplicar para cada nivel de pruebas definido. Se definen los criterios de entrada y salida para cada ciclo de pruebas a ejecutar.

4.3.2. Niveles de Prueba

Existen entre las pruebas de software disimiles niveles de pruebas entre ellos se encuentran las pruebas de integración, pruebas de sistema y pruebas de aceptación. En cada uno de estos niveles

de prueba, se podrán ejecutar diferentes tipos de prueba tales como: pruebas funcionales, no funcionales y de arquitectura.

Pruebas de Sistema: consiste en la ejecución de actividades de prueba en donde se debe verificar que la funcionalidad total de un sistema fue implementada de acuerdo a los documentos de especificación definidos en el proyecto. Los casos de prueba a diseñar en este nivel de pruebas, deben cubrir los aspectos funcionales y no funcionales del sistema.

Prueba de integración: es el nivel de pruebas posterior a las pruebas modulares de los componentes de un sistema. Se centra principalmente en probar la comunicación entre los componentes de un mismo sistema, comunicación entre sistemas o entre hardware y software. Consisten en la comprobación de que elementos del software que interactúan entre sí, funcionan de manera correcta.

Pruebas de Aceptación: estas pruebas las realiza el cliente. Son básicamente pruebas funcionales, sobre el sistema completo, y buscan una cobertura de la especificación de requisitos y del manual del usuario. Estas pruebas no se realizan durante el desarrollo, pues sería impresentable al cliente; sino que se realizan sobre el producto terminado e integrado o pudiera ser una versión del producto pactada previamente con el cliente.

Según la estrategia de prueba de la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr se desarrollaron los niveles de pruebas de sistema y de aceptación. En las pruebas de sistema se cubrieron los aspectos funcionales y no funcionales, mediante pruebas de funcionalidad y de rendimiento.

4.3.3. Pruebas de Funcionalidad

El servicio de pruebas funcionales se centra en comprobar que los sistemas desarrollados funcionan acorde a las especificaciones funcionales y requisitos del cliente. Este servicio ayuda a su organización a detectar los posibles defectos derivados de errores en la fase de programación (33).

Prueba Funcional

El objetivo es asegurar el trabajo apropiado de los requisitos funcionales, incluyendo la navegación, entrada de datos, procesamiento y obtención de resultados. Las metas son:

• Verificar el procesamiento, recuperación e implementación adecuada de las reglas del negocio.

• Verificar la apropiada aceptación de datos.

Método de Prueba

Cualquier producto de ingeniería se puede probar de dos formas:

• **Pruebas de caja negra:** Realizar pruebas de forma que se compruebe que cada función es operativa.

• **Pruebas de caja blanca:** Desarrollar pruebas de forma que se asegure que la operación interna se ajusta a las especificaciones, y que todos los componentes internos se han probado de forma adecuada.

La prueba de caja negra será realizada a la Herramienta informática para la gestión de reportes dinámicos basado en consulta Solr. Los casos de prueba pretenden demostrar que las funciones del software son operativas, que la entrada se acepta de forma adecuada y que se produce una salida correcta.

Prueba de caja negra

Las pruebas de caja negra se llevan a cabo sobre la interfaz del software, obviando el comportamiento interno y la estructura del programa. Los casos de prueba de la caja negra pretenden demostrar que las funciones del software son operativas, la entrada se acepta de forma correcta, se produce una salida correcta y la integridad de la información externa se mantiene. (34)

Las técnicas de caja negra que pueden aplicarse son:

- Prueba de partición equivalente.
- Prueba de análisis de valores límites.
- Grafo de Causa-Efecto

Para las pruebas de caja negra del sistema se utilizara la técnica de prueba de partición equivalente.

Prueba de partición equivalente

Este método de prueba de caja negra divide el dominio de entrada de un programa en clases de datos, a partir de las cuales deriva los casos de prueba. Cada una de las clases de equivalencia representa a un conjunto de estados válidos o inválidos para las condiciones de entrada.

Caso de Prueba: los casos de pruebas serían entonces un conjunto de entradas con datos de prueba, unas condiciones de ejecución, y unos resultados esperados cuyo propósito es identificar y comunicar las condiciones que se llevarán a cabo en la prueba. Los Casos de la Prueba son necesarios para verificar la aplicación exitosa y aceptable de los requisitos del producto (casos de uso). (35)

A continuación se muestra un caso de prueba de caja negra aplicado al CU Autenticar Usuario.

Tabla 10 Caso de Prueba de Caja negra Autenticar Usuario (fuente: elaboración propia)

Escenario	Descripción	Respuesta del sistema	Flujo central
EC 1.1 Autenticar usuario satisfactoriamente.	Se debe autenticar un usuario en el sistema, con los datos correctos.	Se autentica el usuario en el sistema, dándole sus respectivos permisos según su rol y muestra la página principal del sistema.	1-Introducir el nombre de usuario y la contraseña. 2-Presionar el botón Iniciar Sesión.
EC 1.2 Introducir datos incorrectos en los campos de autenticación.	El usuario introduce el nombre de usuario o la contraseña incorrecta en la ventana de autenticación.	Muestra el mensaje "Usuario o contraseña incorrecta".	1- Introducir el nombre de usuario o contraseña incorrectos. 2-Presionar el botón Iniciar Sesión.
EC 1.3 Autenticar usuario con los campos vacíos.	Se debe autenticar un usuario al sistema, con los campos vacíos.	Muestra un mensaje debajo del campo que está vacío: “campo vacío”.	1-No introducir el nombre de usuario o contraseña. 2-Presionar el botón Iniciar Sesión.

No Conformidades: La plantilla de no conformidades recoge los errores que son detectados durante la revisión de la documentación del sistema. Se elabora un documento por cada revisión que se haga y se controlan a través de versiones según se vayan eliminando los errores, hasta que finalmente se hayan erradicado todos los defectos que posea el elemento que se prueba. Además de estar inmerso en la planilla de diseño de casos de prueba, estas no conformidades se van registrando en un documento aparte para luego enviarlo al equipo de desarrolladores.

4.3.4. Pruebas de Rendimiento

Las pruebas de rendimiento consisten en someter al sistema a altas cargas de trabajo mediante la simulación de la actividad de los usuarios reales en el sistema. Esta simulación debe ser lo más fidedigna posible para alcanzar resultados relevantes con conclusiones acertadas y precisas. La simulación se lleva a cabo utilizando las funcionalidades al alcance de los usuarios, realizando una selección entre aquellas más usadas, que demanden más recursos o tiempo para ejecutarse o sean lo suficientemente interesantes para poner a prueba las capacidades del sistema. (34)

Pruebas de Carga: Esta prueba es usada para validar y valorar la aceptabilidad de los límites operacionales de un sistema bajo carga de trabajo variable, mientras el sistema bajo prueba permanece constante. La variación en carga es simular a la carga de trabajo promedio y con picos que ocurren dentro de tolerancias operacionales normales.

Para realizar las pruebas de carga se utilizó la aplicación Apache JMeter que es una herramienta de carga diseñada para realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web. Está desarrollado para diferentes tipos de test; permitiendo diseñar tanto sencillos teses que soliciten simples páginas web, como complejas secuencias de requisiciones que permitan evaluar el comportamiento de una aplicación o como la capacidad de carga máxima que pueda tener una aplicación en un servidor. Apache JMeter en la comprobación de los recursos del sistema, permite una mayor efectividad en el proceso y en la fiabilidad de los resultados (36).

4.4. Resultados de las pruebas aplicadas a la herramienta Sistema de Reportes

En la siguiente tabla se expone un resumen de los resultados de las No Conformidades obtenidos para contabilizar el total de No Conformidades de la herramienta Sistema de Reportes, distinguiendo de ellas cuáles proceden, cuáles no proceden y las que ya fueron resueltas.

Tabla 11 Resumen de no conformidades del proceso de liberación interno (fuente: Registro de Revisiones de Inconsistencias)

No. de Iteraciones	Total de no conformidades	No procede	Resueltas
1ra iteración	5	0	5
2da iteración	1	0	1
3ra iteración	0	0	0

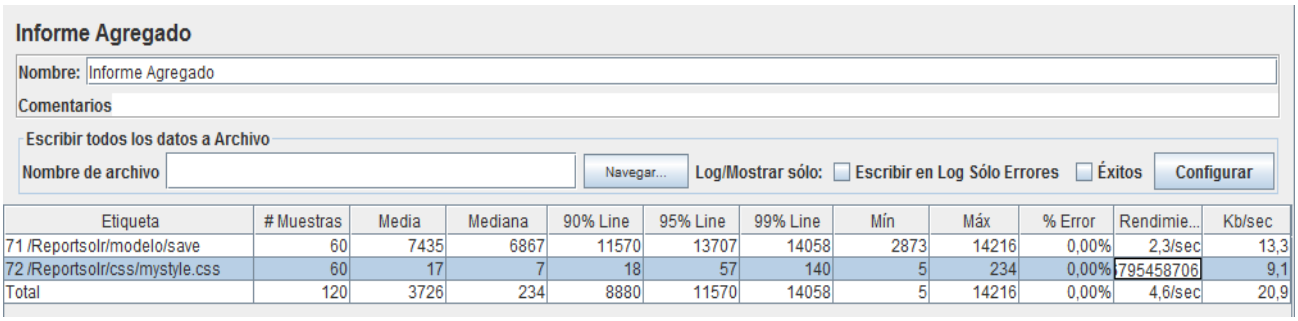
A continuación se exponen los resultados de las pruebas de carga aplicadas a la herramienta Sistema de Reportes, utilizando la aplicación Apache JMeter.

El **Informe agregado** crea una fila por cada petición en el test. Se analizaron los siguientes aspectos:

- Etiqueta: El nombre de la muestra (conjunto de muestras).
- # Muestras: El número de muestras para cada URL.
- Media: El tiempo medio transcurrido para un conjunto de resultados.
- Mín: El mínimo tiempo transcurrido para las muestras de la URL dada.
- Máx: El máximo tiempo transcurrido para las muestras de la URL dada.
- % Error: Porcentaje de las peticiones con errores.
- Rendimiento: Rendimiento medido en base a peticiones por segundo/minuto/hora.
- Kb/sec: Rendimiento medido en Kilobytes por segundo.

Simulación del 30% de usuarios concurrentes conectados de 200 posibles

Para la primera prueba, se introducen 60 hilos (número de usuarios a simular) y 1 segundo de periodo de tiempo.



Informe Agregado

Nombre: Informe Agregado

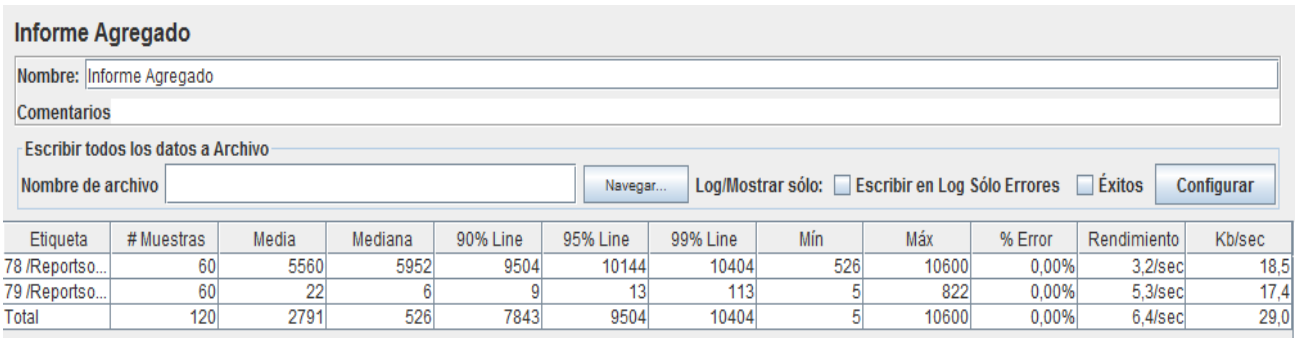
Comentarios

Escribir todos los datos a Archivo

Nombre de archivo | Navegar... | Log/Mostrar sólo: | Escribir en Log Sólo Errores | Éxitos | Configurar

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Mín	Máx	% Error	Rendimie...	Kb/sec
71 /Reportsolr/modelo/save	60	7435	6867	11570	13707	14058	2873	14216	0,00%	2,3/sec	13,3
72 /Reportsolr/css/mystyle.css	60	17	7	18	57	140	5	234	0,00%	795458706	9,1
Total	120	3726	234	8880	11570	14058	5	14216	0,00%	4,6/sec	20,9

Figura 13 Prueba de Carga a la funcionalidad crear modelo de reporte con 60 usuarios. (Fuente: elaborado con (36))



Informe Agregado

Nombre: Informe Agregado

Comentarios

Escribir todos los datos a Archivo

Nombre de archivo | Navegar... | Log/Mostrar sólo: | Escribir en Log Sólo Errores | Éxitos | Configurar

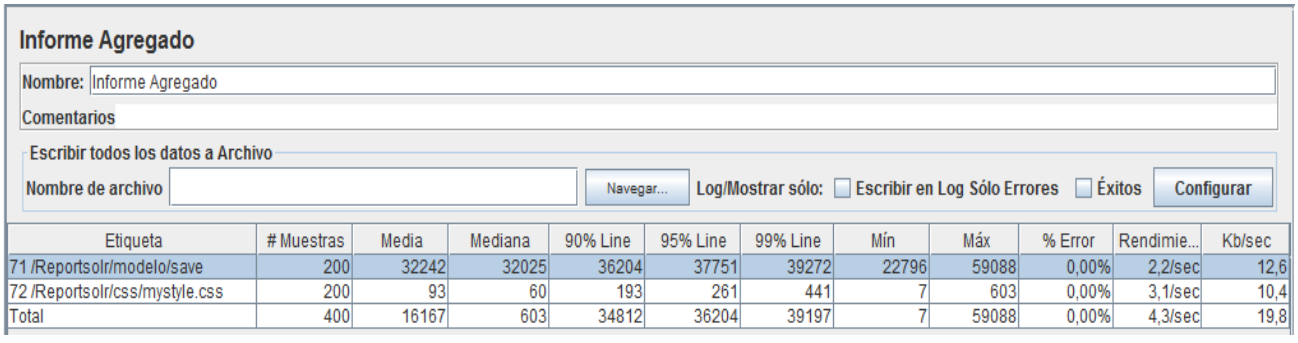
Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Mín	Máx	% Error	Rendimiento	Kb/sec
78 /Reportso...	60	5560	5952	9504	10144	10404	526	10600	0,00%	3,2/sec	18,5
79 /Reportso...	60	22	6	9	13	113	5	822	0,00%	5,3/sec	17,4
Total	120	2791	526	7843	9504	10404	5	10600	0,00%	6,4/sec	29,0

Figura 14 Prueba de Carga a la funcionalidad crear reporte con 60 usuarios. (Fuente: elaborado con (36))

Para las funcionalidades crear modelo de reporte y crear reporte con 60 usuarios conectados concurrentemente se obtiene como resultado que el sistema mantiene su normal funcionamiento con respuestas de 4.6 y 6.4 segundos en las peticiones respectivamente.

Simulación de los 200 usuarios conectados concurrentemente

Para la segunda prueba, se introducen 200 hilos y 1 segundo de periodo de tiempo.



Informe Agregado

Nombre: Informe Agregado

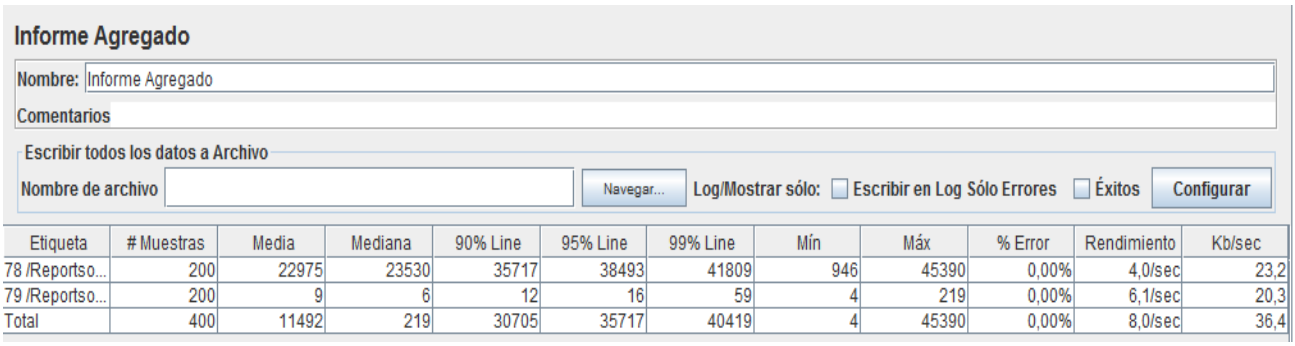
Comentarios

Escribir todos los datos a Archivo

Nombre de archivo | Navegar... | Log/Mostrar sólo: | Escribir en Log Sólo Errores | Éxitos | Configurar

Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Mín	Máx	% Error	Rendimie...	Kb/sec
71 /Reportsolr/modelo/save	200	32242	32025	36204	37751	39272	22796	59088	0,00%	2,2/sec	12,6
72 /Reportsolr/css/mystyle.css	200	93	60	193	261	441	7	603	0,00%	3,1/sec	10,4
Total	400	16167	603	34812	36204	39197	7	59088	0,00%	4,3/sec	19,8

Figura 15 Prueba de Carga a la funcionalidad crear modelo de reporte con 200 usuarios. (Fuente: elaborado con (36))



Etiqueta	# Muestras	Media	Mediana	90% Line	95% Line	99% Line	Mín	Máx	% Error	Rendimiento	Kb/sec
78 /Reportso...	200	22975	23530	35717	38493	41809	946	45390	0,00%	4,0/sec	23,2
79 /Reportso...	200	9	6	12	16	59	4	219	0,00%	6,1/sec	20,3
Total	400	11492	219	30705	35717	40419	4	45390	0,00%	8,0/sec	36,4

Figura 16 Prueba de Carga a la funcionalidad crear reporte con 200 usuarios. (Fuente: elaborado con (36))

Para las funcionalidades crear modelo de reporte y crear reporte con 200 usuarios conectados concurrentemente se obtienen como resultado que el sistema mantiene su normal funcionamiento con respuestas de 4.3 y 8.0 segundos en las peticiones respectivamente.

Según el autor Jakob Nielsen 10 segundos es el tiempo máximo que se puede mantener la atención del usuario sin que se distraiga o inicie otra tarea cualquiera. (37) Los tiempos de respuesta de las pruebas de carga realizadas a las funcionalidades crear modelo de reporte y crear reporte son menores a estos 10 segundos, aunque la funcionalidad crear reporte para 200 usuarios conectados concurrentemente se acerca en gran medida al límite de espera de un usuario. Estas pruebas garantizan un correcto funcionamiento del sistema para un servidor con las prestaciones del ambiente de desarrollo donde se implementó el sistema, descrito en el capítulo 2 en los requisitos no funcionales.

Conclusiones del capítulo

A partir de los resultados obtenidos en este capítulo se logró implementar la herramienta Sistema de Reportes. Los diagrama de componentes y de despliegue permitieron analizar la forma en que se distribuyen los diferentes componentes de la aplicación y como se han de comunicar. La utilización de un estándar de codificación permitió homogenizar el código generado y posibilitó que la herramienta sea entendible por otros programadores. Las pruebas de sistema realizadas permitieron la verificación de funcionalidades y la validación de parte de los requisitos no funcionales del sistema.

CONCLUSIONES

Para dar cumplimiento a los objetivos específicos de este trabajo y a la problemática planteada, se realizaron con éxito las tareas trazadas en el comienzo del mismo.

- El análisis de las herramientas vinculadas al campo de acción evidenció que las mismas no satisfacen la problemática descrita.
- A partir del análisis de los reportes de los principales sistemas generadores de reportes, se determinaron requisitos adicionales que la Herramienta informática para la generación de reportes dinámicos basados en Apache Solr debería cumplir.
- El uso de la metodología RUP permitió documentar todo el ciclo de desarrollo de la Herramienta informática para la gestión de reportes dinámicos basados en consulta Solr.
- Se obtuvo una Herramienta informática para la gestión de reportes dinámicos basada en consulta Solr capaz de satisfacer las necesidades estadísticas del sistema REPXOS 3.0, sin necesidad de hacer cambios en el código fuente de la herramienta.
- Las pruebas realizadas permitieron la verificación de las funcionalidades y la validación de los límites operacionales del sistema.

RECOMENDACIONES

Teniendo en cuenta el estudio realizado durante todo el proceso de desarrollo de la herramienta y en aras de enriquecer la solución, los autores recomiendan:

- ✓ Crear un componente para cargar los metadatos previamente almacenados en la base de datos de REPXOS, con el objetivo de evitar la entrada de datos incompatibles o incorrectos en los parámetros establecidos, necesarios para la creación de las consultas.

REFERENCIAS BIBLIOGRÁFICAS

1. MUÑOZ, D. R. Manual de Estadística. [En línea] 2004. [Citado el: 7 de 11 de 2014.] http://www.eumed.net/cursecon/libreria/drm/drm-estad.pdf..

2. Amerikati. Generador de reportes. [En línea] 2012. http://www.amerikati.com/?page_id=364.

3. Silva, Erika Josefina Sulbaran. *La estadística.* Bolivar : Universidad Bolivariana de Venezuela, 2011.

4. **. About Dspace. DSPACE. [En línea] 2015. [Citado el: 20 de febrero de 2015.] http://www.dspace.org/introducing.**

5. Definición De. [En línea] 2012. [Citado el: 8 de 11 de 2014.] http://definicion.de/reporte.

6. INFANTE PRÓMETA, Y. H. H. *Sistema de Gestión de Reportes Dinámicos.* 2009.

7. REPORTES DINÁMICOS « GESTIÓN 22. REPORTES DINÁMICOS « GESTIÓN 22. [En línea] 2011. http://gestion22.cl/reportes-dinamicos-2/.

8. Repositorio Digital EPN. Repositorio Digital EPN. [En línea] 2010. [Citado el: 14 de diciembre de 2014.] http://bibdigital.epn.edu.ec/handle/15000/2615.

9. UCI. Módulo diseñador de modelos para el generador dinámico de reportes |. [En línea] http://publicaciones.uci.cu/index.php/SC/article/view/1258/711.

10. Metodología open up ágil y tradicional. Metodología open up ágil y tradicional. [En línea] http://es.slideshare.net/carmeloh2/metodologa-open-up-39321348.

11. Metodologias Crystal. Metodologias Crystal. [En línea] http://es.scribd.com/doc/107099160/Metodologias-Crystal#scrib.

12. Rational Software Corporation. *Rational Unified Process.* s.l. : Version 2003.06.00.65., 2003.

13. IVAN JACOBSON, G. B., JAMES RUMBAUGH. *El Proceso Unificado de Desarrollo de Software.* 2000.

14. Groovy . Groovy . [En línea] 2010. http://groovy.codehaus.org/.

15. GRANADOS, P.L.S.C. ALEJANDRO GARCÍA. *Desarrollo Ágil de Aplicaciones Web con Grails Framework.* Hidalgo,Mexico : Universidad Autónoma del Estado de Hidalgo, 2012. ISBN.

16. PostgreSQL: Características, limitaciones y ventajas. PostgreSQL: Características, limitaciones y ventajas. [En línea] 2009. http://postgresql-dbms.blogspot.com/p/limitaciones-puntos-de-recuperacion.html.

17. Apachefoundation. Apachefoundation. [En línea] 2009. http://apachefoundation.wikispaces.com.

18. Larman, Craig. *Uml y Patrones.* s.l. : 2da Edicion, 2003.

19. Visual Paradigm. Visual Paradigm. [En línea] 5 de abril de 2009. http://www.visual-paradigm.com/product/vpuml/modeleredition.jsp.

20. Kuvaja, P., Similä, J., Krzanik, L., Bicego, A., Koch, G. *Software Process Assessment and Improvement. The BOOTSTRAP Approach.* Oxford, UK, and Cambridge : MA , 1994.

21. genbeta. genbetadev. [En línea] 2015. http://www.genbetadev.com/frameworks/bootstrap.

22. Pérez, Javier Eguíluz. *Introducción a CSS.* Madrid : s.n., 2009.

23. Universidad Nacional ,Ibagué, Tolima, Colombia. Tecnologia innovadora. [En línea] http://tecnologiainnovadoraunad.blogspot.com/2012/05/para-que-sirve-el-html5.html.

24. Dreamtech Press. *jQuery in Action.* s.l. : B Bibeault, Y Kats , 2008 .

25. R.S, PRESSMAN. *Ingeniería de Software. Un enfoque práctico.* 2005.

26. Castellano, Diseño web y desarrollo web. Programación en. Diseño web y desarrollo web. Programación en Castellano. *Diseño web y desarrollo web. Programación en Castellano.* [Online] 2013. http://www.programacion.com/articulo/eclipse_--_i_-_historia_y_toma_de_contacto_288.

27. Cambridge University Press. *The Elements of UML 2.0 Style.* 2005. ISBN: 0-521-61678-6.

28. Brito, Nacho. *Manual Grails.* 2009.

29. SlidesShare. [En línea] [Citado el: 20 de 03 de 2015.] http://www.slideshare.net/techmi/curso-uml-25-diagramas-de-implementacion.

30. Barrientos, Pablo Andrés. *Enfoque para pruebas de unidad basado en la generación aleatoria de objetos.* Buenos Aires : Publicaciones SEDICI, 2014. ISBN 10915/34969.

31. Cem Kaner, J.D., Ph.D. Context-Driven-Testing. [En línea] [Citado el: 30 de 4 de 2015.] http://context-driven-testing.com/.

32. Mollom. DataPrix. [En línea] [Citado el: 29 de 04 de 2015.] http://www.dataprix.com/blogs/canzion23/estrategias-pruebas-software.

33. Globe Testing. globetesting. [En línea] http://www.globetesting.com/pruebas-funcionales/.

34. Mtorres. *Tecnicas de Prueba.*

35. Jorrín, Ing. Violena Hernández Aguilar y Ing. Michael González. *Proceso de pruebas de caja negra basado en la descripción de casos de uso. .* s.l. : UCl : s.n.

36. Sociedad Informática del Gobierno Vasco. Apache JMeter. Manual de usuario. [En línea] [Citado el: 1 de junio de 2015.]

37. Nielsen, Jakob. *Usability Engineering.* San Francisco : s.n., 1993. ISBN 0-12-518406-9.

38. Jasper soft Corporation. JasperForge.org. [En línea] 21 de 2 de 2009. http://jasperforge.org/website/jasperreportswebsite/trunk/highlights.html?groupid=252.

39. Arias, Diego Alberto. Servicio de nomenclátor utilizando el motor de búsqueda Solr. Caso práctico en la provincia de Lugo. [En línea] 2012. http://www.ign.es/resources/jiide2012/jueves/manana/Colombia/6.comunicacion-jiide-2012-diegoalberto.arias.pdf.

40. Metodologías de Desarrollo de Software: Características SCRUM. . Metodologías de Desarrollo de Software: Características SCRUM. . [En línea] http://desarrollodefw.blogspot.com/2012/10/caracteristicas-scrum.html.

41. Burbeck, Steve. "Application programming in Smalltalk-80: How to use ModelView-Controller (MVC)". University of Illinois in Urbana-Champaign, Smalltalk . [En línea] [Citado el: 4 de febrero de 2015.] http://st-www.cs.uiuc.edu/users/smarch/st-docs/mvc.html..

42. Sun Developer Network. Sun Developer Network. [En línea] 25 de abril de 2009. http://java.sun.com/blueprints/corej2eepatterns/Patterns/DataAccessObject.html.

43. Sommerville, Ian. *Softweare Engineering.* 2006. ISBN 84 7829 074 5.

44. Kuvaja, P., Similä, J., Krzanik, L., Bicego, A., Koch, G., and Saukkonen. *S., Software Process Assessment and Improvement. The BOOTSTRAP Approach, Blackwell Business.* Oxford, UK, and Cambridge : Cambridge, MA 1994.

45. Flanagan, D. *JavaScript. La Guía Definitiva.* Madrid : s.n., 2007 . ISBN: 978-84-415-2202-2 84-415-2202-2.

46. librosweb. [En línea] 2015. http://librosweb.es/libro/bootstrap_3/capitulo_1/compatibilidad_con_los_navegadores.html.

GLOSARIO DE TÉRMINOS

AJAX: acrónimo de *Asynchronous JavaScript and XML* (JavaScript asíncrono y XML), es una técnica de desarrollo web para crear aplicaciones interactivas. Estas aplicaciones se ejecutan en el cliente, es decir, en el navegador de los usuarios mientras se mantiene la comunicación asíncrona con el servidor en segundo plano.

Bytecode: es un código intermedio más abstracto que el código máquina. Habitualmente es tratado como un archivo binario que contiene un programa ejecutable similar a un módulo objeto, que es un archivo binario producido por el compilador cuyo contenido es el código objeto o código máquina.

CASE*:* ingeniería de software asistida por computadora en español, son diversas aplicaciones informáticas destinadas a aumentar la productividad en el desarrollo de software reduciendo el costo de las mismas en términos de tiempo. Estas herramientas pueden ayudar en todos los aspectos del ciclo de vida de desarrollo del software.

Críptico: que no es comprensible para la mayoría de las personas porque está hecho para ser entendido por unos pocos.

DOM: *Document Object Model* ('Modelo de Objetos del Documento') es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML.

Fidedigno: qué es digno de ser creído o que merece crédito.

Framework: marco de trabajo en español, es un entorno o ambiente de trabajo para desarrollo; dependiendo del lenguaje normalmente integra componentes que facilitan el desarrollo de aplicaciones como el soporte de programa, bibliotecas y plantillas.

MVCC (Acceso concurrente multiversión): permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo commit.

LESS: extiende CSS para dotarlo de un comportamiento dinámico a través de variables, operaciones y funciones.

Scripting: Un lenguaje scripting es un tipo de lenguaje de programación que es generalmente interpretado.