



Universidad de las Ciencias Informáticas

FACULTAD 10

Título: “Módulo de Generación Automática de Boletines para el Sistema de Monitoreo y Análisis de Noticias (SIMAN)”.

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

Autores: Frangel Moreno Díaz.

Tutores: Ing. Eduardo M. Macías Sotolongo

Ing. Adniel Quintana

Ciudad de La Habana, Junio de 2010

“Año 52 de la Revolución”

Declaración de Autoría

Declaración de Autoría.

Declaro que soy el único autor de este trabajo y autorizo a la de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

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

Frangel Moreno Díaz

Ing. Eduardo M. Macías Sotolongo.

Ing. Adniel Quintana Muñoz.

Resumen

La investigación recoge todo el proceso de desarrollo del módulo para la generación automática de boletines en el Sistema de Monitoreo y Análisis de Noticias (SIMAN).

Para cumplir con los propósitos de la investigación se realizó un estudio de los principales sistemas existentes tanto a nivel nacional como internacional, relacionados con la generación de boletines. Se utilizó Java como lenguaje de programación, como metodología de desarrollo el Proceso Racional Unificado (RUP) junto al Lenguaje Unificado de Modelado (UML); como herramienta de modelado Visual Paradigm y NetBeans como plataforma de desarrollo. Se documentaron y generaron los diferentes artefactos necesarios para la construcción de la aplicación como establece la metodología utilizada.

El módulo para la generación automática de boletines del Sistema de Monitoreo y Análisis de Noticias, logrará que la vigilancia tecnológica y política en el proyecto GIDI sea más eficiente.

La herramienta le da la posibilidad al Editor de generar de forma muy rápida un boletín con las noticias que son descargadas por el Sistema de Monitoreo y Análisis de Noticias. Le permite realizar un filtrado de noticias según el tema que se quiera guardar en el boletín, establecer un orden de las mismas y aplicarle un diseño en específico. Obtener un fichero con la información en formatos como: pdf, html, rtf, doc, xsl entre otros.

Palabras Claves: Generación de Boletines, Monitoreo de noticias, Generación de Reportes.

Índice

INTRODUCCIÓN	5
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA Y ESTADO DEL ARTE.	8
1.1 <i>Conceptos relacionados con el tema.</i>	8
1.2 <i>Antecedentes.</i>	9
1.3 <i>Entorno de desarrollo NetBeans.</i>	10
1.4 <i>Metodologías de Desarrollo de Software.</i>	11
1.5 <i>DB4O.</i>	16
1.6 <i>Herramienta y framework escogido para diseñar las plantillas de los reportes.</i>	17
1.7 <i>Fundamentación de la solución propuesta.</i>	21
1.8 <i>Conclusiones.</i>	22
CAPÍTULO 2. CARACTERÍSTICAS DEL SISTEMA.	23
2.1 <i>Requerimientos Funcionales.</i>	23
2.2 <i>Requerimientos No Funcionales.</i>	24
2.3 <i>Descripción de la Arquitectura. Fundamentación.</i>	25
2.4 <i>Análisis de posibles implementaciones, componentes o módulos ya existentes.</i>	26
2.5 <i>Modelo de casos de uso.</i>	28
2.7 <i>Vista de Despliegue.</i>	31
2.8 <i>Conclusiones.</i>	32
CAPÍTULO 3. DESCRIPCIÓN Y ANÁLISIS DE LA SOLUCIÓN PROPUESTA. PRUEBAS.	34
3.1. <i>Modelo de Diseño.</i>	34
3.2. <i>Diagrama de clases del Diseño.</i>	34
3.3. <i>Descripción de las Clases.</i>	35
□ <i>Tabla 15. Descripción de la clase CC_HiloAcoplarNoticias. (Ver Anexo 12)</i>	39
□ <i>Tabla 16. Descripción de la clase CC_HiloBoletines. (Ver Anexo 13)</i>	39
□ <i>Tabla 19. Descripción de la clase CC_HiloNoticiasDescripción. (Ver Anexo 17)</i>	39
□ <i>Tabla 20. Descripción de la clase CC_HiloNoticiasFecha. (Ver Anexo 18)</i>	39
3.4. <i>Diagrama de Interacción.</i>	39
3.5 <i>Diseño de Boletines con IReport.</i>	41
3.6. <i>Diagrama de Componentes.</i>	43
3.7 <i>Trabajo con DB4O y Lucene.</i>	44
3.8 <i>Prueba.</i>	47
3.9 <i>Conclusiones.</i>	55
CONCLUSIONES.	56
RECOMENDACIONES.	57
REFERENCIAS BIBLIOGRÁFICAS.	58
BIBLIOGRAFÍA.	60

ANEXOS	61
<i>Anexo 1. Descripción Caso de Uso Buscar noticia.</i>	<i>61</i>
<i>Anexo 2. Descripción Caso de Uso Organizar Noticia.</i>	<i>62</i>
<i>Anexo 3. Descripción Caso de Uso Establecer campos.....</i>	<i>62</i>
<i>Anexo 4. Descripción Caso de Uso Editar y guardar noticias.....</i>	<i>63</i>
<i>Anexo 5. Descripción Caso de Uso Escoger plantilla.....</i>	<i>64</i>
<i>Anexo 6. Descripción Caso de Uso Exportar boletín.....</i>	<i>65</i>
<i>Anexo 7. Descripción Caso de Uso Guardar boletín en Base de Datos.....</i>	<i>66</i>
<i>Anexo 8. Descripción Caso de Uso Cargar boletín.....</i>	<i>66</i>
<i>Anexo 9. Descripción Caso de Uso Eliminar boletín.....</i>	<i>67</i>
<i>Anexo 10. Descripción Caso de Uso Adicionar noticia.....</i>	<i>68</i>
<i>Anexo 11. Descripción de la clase CC_ControladorVisual.....</i>	<i>69</i>
<i>Anexo 12. Descripción de la clase CC_HiloAcoplarNoticias.....</i>	<i>71</i>
<i>Anexo 13. Descripción de la clase CC_HiloBoletines.....</i>	<i>72</i>
<i>Anexo 14. Descripción de la clase CC_HiloGenerarDiseno.....</i>	<i>72</i>
<i>Anexo 15. Descripción de la clase CC_HiloGenerarCarga.....</i>	<i>73</i>
<i>Anexo 16. Descripción de la clase CC_HiloNoticiasClaves.....</i>	<i>73</i>
<i>Anexo 17. Descripción de la clase CC_HiloNoticiasDescripción.....</i>	<i>74</i>
<i>Anexo 18. Descripción de la clase CC_HiloNoticiasFecha.....</i>	<i>74</i>
<i>Anexo 19. Descripción de la clase CC_HiloNoticiasMarcadas.....</i>	<i>75</i>
<i>Anexo 20. Descripción de la clase CC_HiloNoticiasTítulo.....</i>	<i>75</i>
<i>Anexo 21. Diagrama de Colaboración Buscar noticia.....</i>	<i>76</i>
<i>Anexo 22. Diagrama de Colaboración Organizar noticia.....</i>	<i>76</i>
<i>Anexo 23. Diagrama de Colaboración Establecer campos.....</i>	<i>76</i>
<i>Anexo 24. Diagrama de Colaboración Editar y guardar noticias.....</i>	<i>77</i>
<i>Anexo 25. Diagrama de Colaboración Escoger plantilla.....</i>	<i>77</i>
<i>Anexo 26. Diagrama de Colaboración Exportar boletín.....</i>	<i>77</i>
<i>Anexo 27. Diagrama de Colaboración Guardar boletín en Base de Datos.....</i>	<i>77</i>
<i>Anexo 28. Diagrama de Colaboración Cargar boletín.....</i>	<i>78</i>
<i>Anexo 29. Diagrama de Colaboración Eliminar boletín.....</i>	<i>78</i>
<i>Anexo 30. Diagrama de Colaboración Adicionar Noticia.....</i>	<i>78</i>
GLOSARIO DE TÉRMINOS	79

Introducción

En la actualidad Internet viene alcanzando un desarrollo elevado, por lo cual un mayor número de Empresas se dedican a crear sitios web para darse a conocer ante el mundo. Con el fin de competir con las demás empresas ya sea en cuanto a publicidad, o por el hecho de alcanzar un mayor número de clientes, se toman medidas para aumentar los recursos de información; uno de los recursos empleados son los boletines informativos, el cual debe ser breve y enviado periódicamente.

Por otra parte, que con el elevado desarrollo que van teniendo los Sistemas informáticos, se va haciendo cada vez más necesario que tengan una funcionalidad para generar reportes, según lo que se quiera mostrar del mismo, esto ayuda al usuario a conocer de forma detallada los eventos y funciones que realiza el software. Ejemplo de ello tenemos, los sistemas antivirus, que nos puede brindar reportes, ya sea con el resultado de una búsqueda de virus en la PC, o una lista de los virus que más hayan afectado en un mes determinado. Todo esto permite que esa información pueda guardarse y hasta imprimirse para diferentes actividades.

Se han creado en el Mundo, módulos para integrarlos a Sistemas informáticos, sin importar el tema que se quiera abordar, los cuales permiten especificar el tamaño de la fuente, las imágenes que va a contener, e incluso hay algunos que posibilitan filtrar el contenido para que el boletín esté enfocado en un tema específico.

En Cuba igual que en la Universidad de las Ciencias Informáticas se han desarrollado herramientas informáticas que contienen un módulo de generación de reportes, pero no se conoce ninguna que se integre a un Sistema de Monitoreo y Análisis de Noticias, y que se integre con un software de escritorio.

En la Universidad de las Ciencias Informáticas en el año 2006 fue creado el Proyecto Operación Verdad, con el objetivo de acercar a nuestro país a todo lo que acontece en el mundo, con este fin se creó el Grupo de Vigilancia Tecnológica y Política, el cual monitorea las noticias que son publicadas en Internet.

Los integrantes del Grupo de Vigilancia Tecnológica y Política se organizan en brigadas de trabajo, donde en cada turno deben buscar y descargar manualmente las noticias de Internet, una vez descargadas las noticias, se envían al editor que realiza la actividad de conformar el boletín de forma manual. Todo esto trae consigo que se cometan errores, entre los que encontramos: que se repitan las noticias, el texto no cumpla con el tamaño ni la fuente que requiere y haya un retraso en la entrega de los informes, haciendo el trabajo muy engorroso e ineficiente.

En el curso 2007-2008 en el sub-proyecto GIDI (Grupo de Investigación y Desarrollo sobre Internet) se realizó un Trabajo de Diploma titulado Sistema de Vigilancia Tecnológica para el proyecto Operación Verdad: Análisis y

Diseño del módulo “Edición y Generación de Informes” (Socarras, 2008). El cual realizó todo el análisis y diseño del módulo de Generación de Informes.

Posteriormente en el curso 2008-2009 por medio de un Trabajo de Diploma titulado Implementación del Módulo de Búsqueda del Sistema de Monitoreo y Análisis de Noticias, fue creado el software SIMAN (Sistema de Monitoreo y Análisis de Noticias) (Muñoz, 2009) el cual hoy en día se encuentra funcionando dentro del proyecto, posibilitando que el trabajo del Grupo de Vigilancia Tecnológica y Política sea más eficiente. Pero SIMAN solo cuenta con el módulo de Búsqueda de Noticias, y se hace necesario adicionarle un módulo, capaz de generar de forma automática un boletín informativo con la información deseada y con un diseño específico.

A partir de la situación problemática existente se plantea el siguiente **problema científico**:

¿Cómo generar de forma automática los boletines con las noticias que son descargadas por SIMAN?

Como **objeto de estudio** se tienen los procesos de generación automática de boletines y como **campo de acción** el proceso de generación automática de boletines en el Sistema de Monitoreo y Análisis de Noticias.

Se ha trazado el siguiente **Objetivo General** de la investigación: Desarrollar un módulo de generación automática de boletines para el Sistema de Monitoreo y Análisis de Noticias.

Idea a defender: con la creación de un módulo para la generación automática de boletines del Sistema de Monitoreo y Análisis de Noticias, la vigilancia tecnológica y política en el proyecto GIDI será más eficiente.

Para el desarrollo de la investigación se proponen los siguientes **objetivos específicos**:

1. Implementar un módulo para la generación automática de boletines.
2. Realizar pruebas al módulo de generación automática de boletines.

Para darle respuesta a los objetivos anteriores es necesaria la realización de las siguientes **tareas investigativas**:

1. Analizar la tecnología a utilizar.
2. Analizar el diseño del sistema y valorar cambios.
3. Definir el estándar de codificación a utilizar.

4. Diseñar plantillas de boletines.
5. Desarrollar los artefactos establecidos para el flujo de trabajo Implementación.
6. Implementar el componente.

Este trabajo propone como solución a las dificultades analizadas, realizar un software que permita generar de forma automática un boletín y almacenarla en una base de datos para su posterior análisis y utilización.

El presente documento se encuentra estructurado en introducción, tres capítulos, conclusiones, recomendaciones y anexos. En el **Capítulo 1** se presenta la fundamentación teórica acerca de los sistemas generadores de boletines, que ayudan a agilizar el desarrollo de un software, haciendo un acercamiento al ámbito nacional y de la UCI en general y las posibles soluciones que se le dan a estos problemas. Además de conceptos importantes para la comprensión del trabajo.

En el **Capítulo 2** se presenta el levantamiento de requisitos, se hace una descripción de la arquitectura, se hace un refinamiento los casos de uso definidos por la tesis de análisis y diseño y se muestran los artefactos correspondientes.

En el **Capítulo 3** se presenta la Descripción, Análisis y Prueba de la solución propuesta. Describe como fue implementada la aplicación a través del diagrama de componentes. Recoge las pruebas realizadas a la aplicación con el objetivo de detectar errores y defectos.

Capítulo 1. Fundamentación teórica y estado del arte.

En este capítulo se valoran los aspectos teóricos que fundamentan la implementación del módulo propuesto. Se explican y justifican las tecnologías y herramientas usadas para su desarrollo. Así como las técnicas de programación empleadas para solucionar diversas situaciones. También se da una panorámica de las herramientas existentes para la generación de boletines.

1.1 Conceptos relacionados con el tema.

Boletín: Publicación destinada a tratar de asuntos científicos, artísticos, históricos o literarios, generalmente publicada por alguna corporación (Real Academia Española¹).

Boletín informativo: Un boletín informativo es una publicación distribuida de forma regular, generalmente centrada en un tema principal que es del interés de sus suscriptores. Muchos boletines son publicados por clubes, sociedades, asociaciones y negocios, particularmente compañías, para proporcionar información de interés a sus miembros o empleados. Algunos boletines informativos son creados con ánimo de lucro y se venden directamente a sus suscriptores.

PDF: El formato PDF (Portable Document Format) es el estándar de hecho en la distribución de documentos electrónicos en todo el mundo. El formato PDF es un formato de archivo universal que preserva todas las fuentes, el formateo, los colores y los gráficos de cualquier documento fuente, sin que importe la aplicación ni la plataforma usadas en su creación. Los archivos PDF son compactos y pueden compartirse, visualizarse, navegarse e imprimirse exactamente como pretenda cualquiera. (Sociedad Española de Cardiología , 1997).

XML: (Extensible Markup Language) parecido al HTML de las páginas web, es el lenguaje en el que están escritos los ficheros Jasper que son generados por IReport. Creado por la W3C World Wide Web Consortium). Se trata de un metalenguaje que ordena, estructura y describe los documentos de las páginas web, permitiendo una descripción más minuciosa. En un principio, no rivalizarán HTML y XML, se complementarán el uno al otro anidándose ambas gramáticas. Este Lenguaje de marcas extensible (XML) es una versión abreviada del SGML (Standard Generalized Markup Language). (González, 2008).

Características del lenguaje XML. (González, 2008)

- Es una arquitectura más abierta y extensible. No se necesitan versiones para que puedan funcionar en futuros navegadores. Los identificadores pueden crearse de manera simple y ser adaptados en el acto en internet/intranet por medio de un validador de documentos.
- Integración de los datos de las fuentes más dispares. Se podrá hacer el intercambio de documentos entre las aplicaciones tanto en el propio PC como en una red local o extensa.
- Datos compuestos de múltiples aplicaciones. La extensibilidad y flexibilidad de este lenguaje permitirá agrupar una variedad amplia de aplicaciones, desde páginas web hasta bases de datos.

1.2 Antecedentes.

Realizando una investigación profunda sobre la existencia en el mundo de aplicaciones informáticas, que tuvieran incorporadas un módulo para la generación de reportes, se encontraron las siguientes:

1.2.1 Generación Multilingüe de Boletines Meteorológicos (MultiMeteo)

Es un sistema que genera boletines meteorológicos en diferentes idiomas llamado MultiMeteo, fue realizado en el marco del proyecto europeo del mismo nombre, cuyos miembros usuarios son:

Météo France, Instituto Nacional de Meteorología (España), Institut Royal Météorologique (Bélgica), Zentralanstalt für Meteorologie und Geodynamik (Austria).

Su objetivo es generar boletines en varias lenguas (inglés, español, francés, alemán y neerlandés) y varios estilos a partir de los formatos de datos de los institutos europeos.

El sistema toma como entrada una base numérica de datos de predicción, obtenida a partir de modelos matemáticos, y produce textos que contienen predicciones meteorológicas derivada de dicha base de datos. Usa un motor de generación que contiene dos módulos, el de planificación y el de realización. (Karine Chevreau, 2000)

1.2.2 Publish-iT

Es una especie de procesador de textos, pero especialmente diseñado para crear boletines de noticias

Esta utilidad combina herramientas de un procesador, con capacidad también para trabajar con gráficos, tablas y funciones propias de una aplicación de maquetación y publicación online.

Es capaz de manejar al mismo tiempo diversos diseños y varias páginas en un mismo documento, e incluye una amplísima variedad de utilidades de edición de textos y gráficos.

El programa incorpora plantilla de diseño para facilitarte el trabajo, así como un completo tutorial. (Santos., 2006)

1.2.3 Sistema de Investigación e Información Policial (SIIPOL)

Sistema perteneciente a la facultad 2, implementado en J2EE, contiene un módulo para la generación de reportes totalmente funcional para aplicaciones Web y no para Desktop.

Análisis de los antecedentes.

Se determinó que las herramientas encontradas en la investigación no cumplen con las características que requiere el módulo de generación de boletines que se quiere integrar con SIMAN. El mismo debe permitir:

Escoger una plantilla para cada boletín que se quiera generar.

Diseñar las plantillas independientes del sistema.

Integrarse con el motor de base de datos DB4O y realizar las consultas necesarias.

Por tanto, se hace necesario implementar un módulo de generación automática de boletines para integrarlo con SIMAN y no utilizar los existentes.

1.3 Entorno de desarrollo NetBeans.

El NetBeans es un entorno de desarrollo integrado (IDE), basado en estándares, escrito en el lenguaje de programación Java. El proyecto NetBeans consiste en un IDE de código abierto y una plataforma de aplicaciones, que puede ser utilizado como un marco genérico para construir cualquier tipo de aplicación. El enfoque de NetBeans 6.8 es la mejora de la productividad a través de un editor más inteligente, más rápido, y la integración de todos los productos en un IDE. (Sitio Oficial del Netbeans)

El NetBeans 6.8 amplía el soporte más allá de Java y C/C++ al ofrecer una serie de funcionalidades para los lenguajes de programación dinámicos Ruby y Java Script, así como soporte para el último Java Standards, manteniendo, a su vez, la facilidad de uso y alta productividad de NetBeans.

Además esta versión incluye una mejor edición del código, capacidades de navegación e inspección, historia local, soporte integrado para Subversion (un controlador de versiones), y mayores capacidades de personalización.

1.4 Metodologías de Desarrollo de Software.

Las metodologías de desarrollo de software describen los pasos que se deben seguir para la producción de un determinado producto informático, a través de patrones, que surgieron gracias a la experiencia en un determinado campo del desarrollo de software, como es el caso de los patrones de diseño, los cuales sugieren una serie de pasos a la hora de realizar un diseño determinado.

Metodologías Estructuradas

Estas metodologías fueron creadas en los años 70 según surgió la programación estructurada, por lo que es potencialmente efectivo en proyectos que utilizan como lenguajes de desarrollo los de tercera y cuarta generación. Ejemplos de estas metodologías son: MERISE2 (Francia), MÉTRICA3 (España), SSADM4 (Reino Unido), Gane & Sarson5, Ward & Mellor6, Yourdon & DeMarco7 e Information Engineering8. La metodología estructurada está basada en la representación de las funciones que debe realizar el sistema y los datos que fluyen entre ellas. (AGUT, 2008)

Metodologías tradicionales

Al inicio el desarrollo de software era artesanal en su totalidad. La ausencia de procesos formales, lineamientos claros, determinaron que se importara la concepción y fundamentos de metodologías existentes en otras áreas, y adaptarlas al desarrollo de software. Esta nueva etapa de adaptación contenía el desarrollo dividido en etapas de manera secuencial, que de algo mejoraba la necesidad latente en el campo del software. Entre las principales metodologías tradicionales existen los tan conocidos RUP y MSF entre otros, que centran su atención en llevar una documentación exhaustiva de todo el proyecto y en cumplir con un plan de proyecto, definido en la fase inicial del desarrollo del proyecto.

Metodologías ágiles

Estas metodologías se basan en dos aspectos puntuales, el retrasar las decisiones y la planificación adaptativa; permitiendo potenciar aún más el desarrollo de software a gran escala. Ya que tienen un desarrollo de software incremental, es decir, se van entregando pequeñas porciones de software en períodos de tiempo cortos. Además de que los desarrolladores y clientes trabajan juntos, permitiendo de esta forma una cercana comunicación entre ellos, logrando controlar de una forma efectiva cualquier cambio que se produzca en el proceso de desarrollo del software. Y por sobre todo es muy sencillo pues es muy fácil de aprender, de modificar y muy bien documentado.

Dentro de estas metodologías se encuentran el Extreme Programming (XP), el SCRUM, incluso el RUP podría llegar a ser ágil realizándole una configuración adecuada dado por el especial énfasis que presenta en cuanto a su adaptación a las condiciones del proyecto, Dynamic Systems Development Method (DSDM), entre otros.

Metodologías más Usadas

Las metodologías más usadas en la actualidad dados por las potencialidades y las capacidades de adaptabilidad que tienen son el XP, el RUP, el SCRUM, MSF y la más reciente la OMT.

Rational Unified Process (RUP)

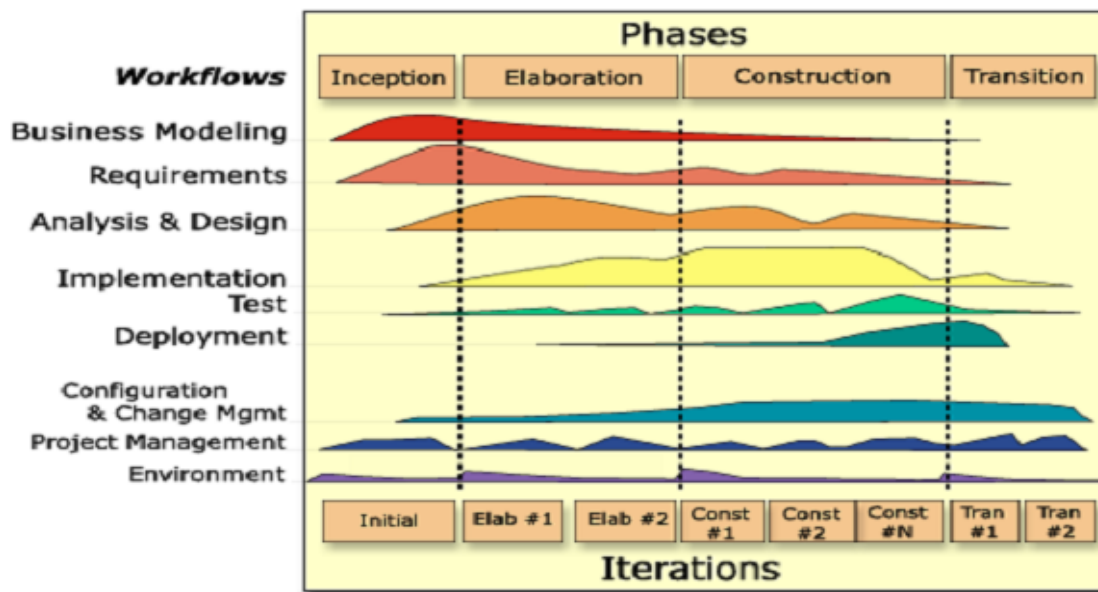
El RUP (Rational Unified Process) especifica un framework para el desarrollo de un proyecto, en particular un proyecto de desarrollo de software, definiendo: etapas, actividades a realizar por un equipo de desarrollo, secuencia y lógica necesaria para obtener el producto final. (RIESCO, 2004)

RUP es un proceso formal: Provee un acercamiento disciplinado para asignar tareas y responsabilidades dentro de una organización de desarrollo. Su objetivo es asegurar la producción de software de alta calidad que satisfaga los requerimientos de los usuarios finales (respetando cronograma y presupuesto). Fue desarrollado por Rational Software, y está integrado con toda la suite Rational de herramientas. Puede ser adaptado y extendido para satisfacer las necesidades de la organización que lo adopte. Es guiado por casos de uso y centrado en la arquitectura, y utiliza UML como lenguaje de notación.

El RUP y los productos similares, como Object-Oriented Software Process (OOSP), y el OPEN Process, son herramientas de la ingeniería del software que combinan los aspectos procesales de desarrollo (como fases definidas, técnicas, y prácticas) con otros componentes de desarrollo (como documentos, modelos, manuales, código fuente, etc.) dentro de un framework unificado.

Está formado por cuatro fases y 9 flujos de trabajo. Cada una de estas etapas o fases es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes. Además de ser guiado por casos de uso, centrado en la arquitectura y se caracteriza por ser iterativo e incremental.

Imagen 1. Esquema de Trabajo de RUP.



Esquema de Trabajo RUP

Frases de RUP

Las cuatro fases del ciclo de vida son:

- Concepción (Inception)
- Elaboración (Elaboration)
- Construcción (Construction)
- Transición (transition)

Flujos de Trabajo

Los flujos de trabajo son nueve, de ellos:

Seis son de ingeniería

1. Modelamiento del Negocio: que es donde se entiende las necesidades del negocio.
2. Requerimientos: Traslado de las necesidades del negocio a un sistema automatizado.

3. Análisis y Diseño: Trasladando los requerimientos dentro de la arquitectura de software.
4. Implementación: Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.
5. Pruebas: Asegurándose que el comportamiento requerido es el correcto y que todo el solicitado está presente.
6. Instalación: Produce release del producto y realiza actividades (empaquete, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.

Y tres son de soporte.

7. Administración del proyecto: Involucra actividades con las que se busca producir un producto que satisfaga las necesidades de los clientes.
8. Administración de configuración y cambios: Describe cómo controlar los elementos producidos por todos los integrantes del equipo de proyecto en cuanto a: utilización/actualización concurrente de elementos, control de versiones, etc.
9. Ambiente: Contiene actividades que describen los procesos y herramientas que soportarán el equipo de trabajo del proyecto; así como el procedimiento para implementar el proceso en una organización.

Ventajas

- Evaluación en cada fase que permite cambios de objetivos
- Funciona bien en proyectos de innovación.
- Es sencillo, ya que sigue los pasos intuitivos necesarios a la hora de desarrollar el software.
- Seguimiento detallado en cada una de las fases.

Desventajas

- La evaluación de riesgos es compleja
- Excesiva flexibilidad para algunos proyectos
- Se corre el riesgo de poner al cliente en una situación que puede ser muy incómoda para él.
- El cliente deberá ser capaz de describir y entender a un gran nivel de detalle para poder acordar un alcance del proyecto con él.

Extreme Programming (XP)

Es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los desarrolladores, y propiciando un buen clima de trabajo.

XP se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y coraje para enfrentar los cambios. XP se define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes, y donde existe un alto riesgo técnico. (LETELIER)

Los defensores de XP consideran que los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos. Creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

Las características fundamentales del método son:

- ✓ Desarrollo iterativo e incremental: pequeñas mejoras, unas tras otras.
- ✓ Pruebas unitarias continuas, frecuentemente repetidas y automatizadas, incluyendo pruebas de regresión. Se aconseja escribir el código de la prueba antes de la codificación.
- ✓ Programación por parejas: se recomienda que las tareas de desarrollo se lleven a cabo por dos personas en un mismo puesto. Se supone una mayor calidad del código escrito de esta manera, ya que el código es revisado y discutido mientras se escribe.

Ventajas

- ✓ Apropiado para entornos volátiles.
- ✓ Estar preparados para el cambio, significa reducir su coste.

- ✓ Planificación más transparente para los clientes, conocen las fechas de entrega de funcionalidades vitales para su negocio.

Desventajas

- ✓ Delimitar el alcance del proyecto con el cliente. Para mitigar esta desventaja se plantea definir un alcance a alto nivel basado en la experiencia.

Resultado del estudio de las metodologías

Después de un detallado estudio de las metodologías anteriores se decidió continuar utilizando RUP definido anteriormente en la tesis de Sistema de Vigilancia Tecnológica para el proyecto Operación Verdad: Análisis y Diseño del módulo “Edición y Generación de Informes” (Socarras, 2008), para el refinamiento de algunos elementos necesarios, en la implementación del Módulo de Generación Automática de Boletines por dos razones. Una es que el módulo a implementar pertenece al Sistema de Monitoreo y Análisis de Noticias (SIMAN) el cual también utilizó RUP. La segunda razón es que a pesar de que desde la realización de la tesis de Análisis y Diseño del módulo “Edición y Generación de Informes” han pasado dos años y con ello han surgido nuevas metodologías de desarrollo, RUP aun sigue teniendo un protagonismo importante a la hora de desarrollar un software por su robustez y su estricta documentación.

1.5 DB4O.

Db4o (Data Base for Object, por sus siglas en inglés) es una base de datos orientada a objetos de código abierto para Java y .NET. Se afirma ser la única base de datos que es nativa para ambos lenguajes de programación, de Java y .Net. (WEEKLY, 2007)

Actualmente este producto funciona como una librería. La mayor clave del éxito que está teniendo este motor de base de datos frente a otros competidores que han desarrollado tecnologías similares, es que se ha optado por la licencia dual GPL/comercial. Es decir, si se quiere desarrollar software libre con esta librería, su uso no conlleva ningún coste por licencia. DB4O reduce el tiempo y costo de desarrollo y provee un desempeño superior. Almacena incluso las estructuras de objetos más complejas con una sola línea de código.

Dentro de sus principales características se encuentran:

- ✓ **Alto rendimiento:** Ofreciendo notables ventajas con sistemas que utilizan objetos anidados o compuestos, o en donde existen referencias cruzadas, herencia o interacciones ricas entre los objetos.
- ✓ Por su **bajo consumo de recursos**, es especialmente apta para dispositivos móviles y entornos Clientes/Servidor, aunque no necesariamente limitada sólo a ellos.
- ✓ **Doble licencia:** GPL (Open Source) y Comercial (que incluye soporte).

1.6 Herramienta y framework escogido para diseñar las plantillas de los reportes.

Existen en la actualidad varias herramientas que nos permiten diseñar de forma visual plantillas de reportes con una gran facilidad, se integran con los IDEs de desarrollo agilizando la implementación de un módulo para la generación de reportes, dependiendo de las características de cada herramienta el equipo de desarrollo determina cual va a utilizar, por lo cual se realiza una investigación de las principales existentes hasta la actualidad, para ver cual es la más idónea en el desarrollo del módulo para generar boletines que se quiere integrar con SIMAN.

1.6.1 Report Sharp-Shooter. (Anónimo1)

Report Sharp-Shooter es un componente de generación de informes que ayuda a crear informes flexibles y con muchas funcionalidades, y a integrarlos fácilmente en aplicaciones Web y de Escritorio. Soporta una variedad de formatos de exportación estándares de la industria, incluyendo PDF, HTML, Excel, Word, RTF, etc.

La generación de informes es muy personalizable debido al soporte de C# y VB.Scripting .NET. No hay necesidad de aprender lenguajes script adicionales. Esto garantiza al desarrollador el acceso total a todas las posibilidades de .NET Framework, tales como importar cualquier espacio de nombres del proyecto, variables locales, procedimientos, etc.

Los informes y sub informes parametrizados proveen la oportunidad de especificar el nivel de detalle necesario para propósitos particulares y crea una jerarquía complicada de informes. Report Sharp-Shooter ofrece a los desarrolladores una manera rápida y efectiva de crear e integrar informes interactivos y altamente formateados que satisfacen los requerimientos del usuario final.

Características claves de Report Sharp-Shooter.

- Compatible con los más populares. NET IDEs.

- Soporta WinForms y ASP.NET WebForms.
- El producto está escrito en C # y contiene sólo el código administrado.
- Formato de archivo de informe basado en XML es fácil de compartir a través de Internet.

1.6.2 Crystal Reports. (Microsoft, 2005)

Crystal Reports ha formado parte de Visual Studio desde 1993, y ahora es el estándar de creación de informes de Visual Studio 2005. Se incluye en todas las copias de Visual Studio 2005 y se integra directamente en el entorno de desarrollo.

Crystal Reports para Visual Studio 2005 incorpora la posibilidad de crear contenido interactivo con calidad de presentación al entorno de Windows. Con Crystal Reports para Visual Studio 2005, puede crear informes complejos y profesionales en un programa basado en GUI. Después puede conectar el informe a casi todos los orígenes de base de datos, así como a datos proxy, como un conjunto de resultados (por ejemplo, un ADO.NET DataSet). Los asistentes del diseñador de GUI le permiten establecer fácilmente los criterios de formato, agrupamiento y gráficos, etc.

Crystal Reports para Visual Studio 2005 se suministra con un SDK ampliado. Puede utilizarlo para interactuar con el informe mediante programación en tiempo de ejecución, usando uno de los cuatro modelos de objetos posibles:

- CrystalReportViewer, el modelo de objetos más sencillo.
- ReportDocument, el modelo de objetos más completo.
- ReportClientDocument, el modelo de objetos más completo.

Los informes se pueden relacionar con el proyecto de Visual Studio 2005 de muchas formas:

- Incruste los informes directamente en el proyecto.
- Obtenga acceso a ellos externamente, desde un directorio de archivos.
- Obtenga acceso a ellos como servicio Web de informes desde un servidor remoto.

1.6.3 iReport. (Herrera, 2005)

La herramienta iReport es un constructor / diseñador de informes visual, poderoso, intuitivo y fácil de usar para JasperReport escrito en Java. Este instrumento permite que los usuarios corrijan visualmente informes complejos con cartas, imágenes y subinformes.

IReport está además integrado con JFreeChart, una de la biblioteca gráficas de código abierto más difundida para Java. Los datos para imprimir pueden ser recuperados por varios caminos incluso múltiples uniones JDBC, TableModels, JavaBeans y XML. Es una herramienta visual que sirve para generar ficheros XML que se puedan utilizar con la herramienta de generación de informes JasperReports.

Características claves de IReport.

- 100% escrito en Java y además Open Source y gratuito.
- Maneja el 98% de las etiquetas de JasperReports.
- Permite diseñar con sus propias herramientas: rectángulos, líneas, elipses, campos de los textfields, cartas, y subreportes.
- Soporta internacionalización nativamente.
- Browser de la estructura del documento.
- Recopilador y exportador integrados.
- Soporta JDBC.

Resultado del estudio de las herramientas existentes.

Estas son las herramientas para generar plantillas de reportes de forma visual, más distribuidas de las que existen en la actualidad, de las cuales se optó por utilizar IReport para desarrollar el módulo a integrar con SIMAN, por ser esta la más idónea, ya que:

1. Es la herramienta que domina el equipo de desarrollo.
2. Está escrito el 100% en el lenguaje Java, es Open Source y gratuito.
3. Maneja el 98% de las etiquetas de JasperReports, la cual es la librería para generar reportes que se utiliza en el desarrollo del módulo.
4. Con la creación de un plugin de iReport para NetBeans, se logró hacer que el trabajo con esta herramienta sea más cómodo para el propio equipo de desarrollo.

Además que Report Sharp-Shooter y Crystal Reports presentan inconvenientes como por ejemplo:

1. Ninguno de los dos está implementado en JAVA, que es el mismo lenguaje en el que se va a desarrollar el módulo para SIMAN, lo cual no se garantiza que no traería conflictos de integración entre el lenguaje y la plantilla creada.
2. Son aplicaciones bajo la licencia de software privativo, lo cual imposibilita que se puedan utilizar libremente y tener acceso a todas sus funcionalidades.

Por lo tanto, se hizo necesario implementar un módulo de generación de boletines que utilizara las plantillas de iReport, para integrarlo con SIMAN.

Un elemento importante es escoger, una herramienta para integrar con el lenguaje de desarrollo, y utilizar las plantillas diseñadas en iReport. Entre las que se encuentran con más demanda hoy en día:

Frameworks JasperReport: JasperReport es una poderosa herramienta libre de código abierto, se compone de un conjunto de librerías java para facilitar la generación de informes en aplicaciones tanto web como de escritorio. Permite incluir en los reportes textos, imágenes, gráficos y hasta genera imágenes de fondo para utilizarlo como marcas de agua con el propósito de identificar el reporte o simplemente por motivos de seguridad.

Los reportes se pueden dividir en secciones opcionales que son: título del reporte, el encabezado de página, una sección para los detalles del reporte, el pie de página y una sección de resumen que aparece al final del reporte. Pueden generar sub-reportes que permiten la creación de reportes dentro de otro reporte lo que facilita bastante el diseño.

Los reportes generados pueden ser exportados a una multitud de formatos como PDF, XLS, RTF, HTML, XML, TXT, jasper viewer, CVS (valores separados por coma) y texto plano.

Los reportes son capaces de presentar los datos de manera textual o a través de gráficos: no sólo son capaces de mostrar los datos que le son pasados sino que pueden generar o calcular con esos datos otros datos de forma dinámica y mostrarlos.

Sostiene la generación de gráficos estadísticos a través de la librería JFreeChart. En caso de querer generar datos directamente desde la aplicación, los gráficos pueden crearse independientemente como imágenes o generados por datos de entrada, incluso utilizando una de las numerosas librerías de código libre, disponibles para la

creación de gráficos. La imagen producida será mostrada usando un componente de imagen generado a través de la librería JFreeChart.

JasperReport es la herramienta que va a permitir que el Módulo de Generación Automática pueda utilizar las plantillas que fueron diseñadas con iReport.

1.7 Fundamentación de la solución propuesta.

Luego del análisis y el estudio realizado en este capítulo se propone implementar un módulo para el Sistema de Monitoreo y Análisis de Noticias (SIMAN) que le permita a dicho sistema brindarle al cliente la posibilidad de generar de forma automática un boletín con las noticias que este decida que van a estar en el mismo.

Por lo que para la realización del módulo de manera más eficiente y que cumpla con los requerimientos propuestos por el proyecto, se proponen las siguientes

Herramientas:

- Herramienta Case para el modelamiento: Visual Paradigm 6.4 Enterprise Edition.
- Entorno de desarrollo: NetBeans 6.8.
- Lenguaje de Programación: JAVA.
- Herramienta para diseñar plantillas: iReport para el diseño de las plantillas de los boletines.
- Sistema gestor de base de datos: DB4O.
- Framework para la generación de reportes: JasperReport.

Tecnologías:

- Lenguaje de Modelado: UML.
- Metodología de desarrollo de software.

El módulo se modelará en la herramienta case de modelación Visual Paradigm, mediante el lenguaje de modelado UML y bajo la metodología de desarrollo de software RUP. Se implementará en el lenguaje de programación Java aprovechando las potencialidades del NetBeans 6.8 como entorno de desarrollo integrado y a la vez explotar las funcionalidades de iReport como diseñador visual de plantillas y JasperReport como framework para la generación de reportes. Y por último DB4O como sistema gestor de base de datos.

1.8 Conclusiones.

En este capítulo se analizaron las características principales de algunos programas, sistemas y herramientas que realizan la generación de boletines, se ha dado una panorámica de los existentes actualmente en Cuba y el mundo. Se fundamentó la utilización del lenguaje y la metodología de desarrollo de software a utilizar en el desarrollo de la propuesta de la solución, así como algunos conceptos fundamentales a tener en cuenta.

Capítulo 2. Características del sistema.

En este capítulo se presentan los elementos de arquitectura utilizados para llevar a cabo el desarrollo del Módulo de Generación Automática de Boletines del Sistema de Monitoreo y Análisis de Noticias. Abordando lo requerimientos no funcionales imprescindibles para que el sistema funcione correctamente y con buen rendimiento, se fundamenta la arquitectura a utilizar, también se realiza un análisis crítico de posibles implementaciones y componentes existentes que se reutilizaron, así como las estrategias de integración del mismo, aspectos de seguridad, la vista de despliegue del sistema y por último pero no menos importante el estándar de codificación a utilizar.

2.1 Requerimientos Funcionales.

En el trabajo de diploma Sistema de Vigilancia Tecnológica para el proyecto Operación Verdad: Análisis y Diseño del módulo “Edición y Generación de Informes” (Socarras, 2008) se determinaron los requisitos funcionales del Módulo de Generación automática de Boletines, los cuales debido a que han pasado dos cursos y las necesidades del proyecto Operación Verdad han cambiado hasta la fecha, se ha hecho un refinamiento de los requisitos funcionales los cuales quedan listados de la siguiente forma:

- R1. Marcar noticias.
- R2. Buscar noticias.
- R3. Ordenar noticias.
- R4. Establecer campos.
- R5. Editar y guardar noticias.
- R6. Escoger plantilla.
- R7. Exportar boletín.
- R8. Guardar boletín.
- R9. Cargar boletín.
- R10. Eliminar boletín.
- R11. Adicionar noticia.

2.2 Requerimientos No Funcionales.

Los requisitos o requerimientos no funcionales son propiedades o cualidades que el producto debe tener. Debe pensarse en estas propiedades como las características que hacen al producto atractivo, usable, rápido o confiable. No se refieren a funciones específicas que proporciona el sistema. Son restricciones de los servicios o funciones ofrecidas por el sistema y surgen de las necesidades del usuario.

Por el hecho de que se llamen —no funcionales— no quiere decir que no aporten a la funcionalidad del sistema que se esté desarrollando ya que sirven como complemento a los requisitos funcionales que debe cumplir el sistema. Un ejemplo de lo anteriormente planteado sería que un sistema cualquiera podría tener como requisito funcional que toda la información con la que se trabaje en el mismo se debe guardar en una base de datos, y como requisito no funcional que las salvadas deben realizarse en un intervalo de tiempo determinado.

De no cumplir con ese requisito no funcional, a pesar de estar cumpliendo con la funcionalidad, el sistema no estaría funcionando correctamente y por ende no cumplir con las necesidades del cliente o del usuario. Además, los requerimientos no funcionales pueden marcar la diferencia entre productos que cumplan con todas las funcionalidades requeridas. Ya que son los que pueden llegar a definir el grado de aceptación de un producto determinado como pueden ser: la fácil manipulación del sistema por parte del usuario, que el usuario pueda realizar una operación determinada por más de una vía, en fin son los que le dan un rostro al producto a mostrar.

A continuación se detallan los requerimientos no funcionales del Módulo de Generación Automática de Boletines:

Requerimientos de apariencia o interfaz externa

- El módulo debe tener un ambiente amigable y entendible para los usuarios finales, de forma tal que no les sea muy complicado realizar las operaciones sobre el mismo.
- Identificación y utilización de los mismos colores y formatos utilizados en SIMAN para lograr una mejor apariencia.

Requerimientos de usabilidad

- El módulo podrá ser usado por cualquier persona que tenga conocimientos básicos en el manejo de la computadora y esté interesada en trabajar en la aplicación.

- Debe brindarse comodidad a la hora de acceder a las diferentes funcionalidades que brinda el módulo mediante teclas de acceso rápido, la navegabilidad no debe ser muy compleja, todas las funcionalidades deben ser rápidamente accesibles por el usuario.

Requerimientos de rendimiento

- Se debe garantizar que la respuesta a solicitudes de los usuarios del sistema se realice en un breve período de tiempo para la toma de decisiones.
- Se debe garantizar que la generación del boletín se realice en un breve período de tiempo.

2.3 Descripción de la Arquitectura. Fundamentación.

La arquitectura de software, tiene que ver con el diseño y la implementación de estructuras de software de alto nivel. Es el resultado de ensamblar un cierto número de elementos arquitectónicos de forma adecuada para satisfacer la mayor funcionalidad y requerimientos de desempeño de un sistema, así como requerimientos no funcionales, como la confiabilidad, escalabilidad, portabilidad, y disponibilidad.

Philippe Kruchten.

La arquitectura del software es una disciplina que a pesar de conocerse desde los años 60 por grupos de personalidades en la informática como Dijkstra. No se hace popular hasta 1990 con la crisis del software, que azotó en ese año. Entre otros factores, debido a que se realizaba la producción de software sin un patrón o una buena práctica a seguir. Como consecuencia, el resultado final después de grandes inversiones y empleo de un tiempo considerable, se obtenían productos altamente acoplados y en muchos casos con muy alta cohesión. Lo que provocaba que si se detectaba un error en algunos de los módulos desarrollados paralelamente, la corrección del mismo resultaba una tarea titánica. En muchos casos significaba rehacer gran parte del trabajo, porque existía una dependencia total o parcial entre los módulos. Por esta razón es que alcanzan gran auge la arquitectura en capas, la de cliente/servidor, la modelo vista controlador (MVC), etc.

Modelo Vista Controlador (MVC)

Modelo Vista Controlador es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos. El patrón MVC se ve frecuentemente en aplicaciones web, donde la vista es la página HTML y el código que provee de datos dinámicos

a la página, el modelo es el Sistema de Gestión de Base de Datos y la Lógica de negocio, y el controlador es el responsable de recibir los eventos de entrada desde la vista. (Anónimo4)

El patrón de diseño software MVC establece que la arquitectura se divide en tres partes fundamentales:

Modelo: Esta es la representación específica de la información con la cual el sistema opera. La lógica de datos asegura la integridad de estos y permite derivar nuevos datos.

Vista: Este presenta el modelo en un formato adecuado para interactuar, usualmente la interfaz de usuario.

Controlador: Este responde a eventos, usualmente acciones del usuario e invoca cambios en el modelo y probablemente en la vista.

La principal ventaja de esta separación reside en la facilidad para realizar cambios en la aplicación puesto que:

- Cuando se realiza un cambio de bases de datos, programación o interfaz de usuario solo se trabaja con uno de los componentes sin que afecte al resto de los componentes.
- Se puede modificar uno de los componentes sin conocer cómo funcionan los otros.

En el caso del Módulo de Generación Automática de Boletines para el Sistema de Monitoreo de Análisis de Noticias (SIMAN) se aplica el patrón de arquitectura Modelo Vista Controlador (MVC) que ya analizamos anteriormente, por lo que contiene las clases persistentes en el paquete llamado Modelo, las clases controladoras en otro paquete llamado Controlador y por último las clases correspondientes a la interfaz de usuario en el paquete llamado Vista. Destacar que como el módulo debe contar con un fuerte trabajo de componentes visuales se utilizaron dos clases controladoras para dividir el control de los elementos visuales y el control de la lógica del negocio.

2.4 Análisis de posibles implementaciones, componentes o módulos ya existentes.

En la actualidad la reusabilidad (uso de componentes desarrollados con anterioridad), se ha convertido en una práctica a seguir por casi todos los desarrolladores del mundo. Ya que se logra reducir el tiempo de desarrollo al máximo, se construyen aplicaciones más robustas y eficientes. Además de que se puede

implementar en paralelo y reducir las formaciones de “cuello de botella” (término asignado en la ingeniería de software, al proceso de esperar a que se implemente o desarrolle una funcionalidad determinada para comenzar a desarrollar otra) en la producción.

En el Módulo de Generación Automática de Boletines se emplearon componentes ya desarrollados antes, estos son los pertenecientes al:

Paquete Swing de JAVA: Es una biblioteca gráfica para Java. Incluye widgets para interfaz gráfica de usuario tales como cajas de texto, botones, desplegados y tablas.

Ventajas de Swing:

- El diseño en Java puro posee menos limitaciones de plataforma.
- El desarrollo de componentes Swing es más activo.
- Los componentes de Swing soportan más características.

Además, se hizo uso de librerías externas al IDE de desarrollo, que sirvieron para optimizar el tiempo de desarrollo del mismo.

Estas librerías se detallan a continuación:

- **JCalendar:** JCalendar es un bean de Java que consiste en un calendario que permite seleccionar una fecha. (Jide, 2007)
- **Lucene:** Lucene es una librería escalable de alto rendimiento para la recuperación de información, que permite agregar capacidades de indexación y búsqueda a las aplicaciones. Puede indexar y buscar cualquier dato que pueda ser convertido a formato textual. Se trata de una API de indexación y búsqueda con simplicidad en su utilización que requiere básicamente que el usuario aprenda el uso de sus clases. (Figuerola, 2007)

Lucene está implementado originalmente en el lenguaje de programación Java, y es miembro de la familia Apache Jakarta, siendo un proyecto de código abierto gratuito, muy maduro y fácil de utilizar. Durante muchos años ha sido la librería gratuita más utilizada bajo Java para recuperación de información. Lucene, siendo una librería, no es una aplicación finalizada lista para ser utilizada, como lo pudiera ser un programa buscador de archivos o un buscador de sitios Web. Sino más bien es

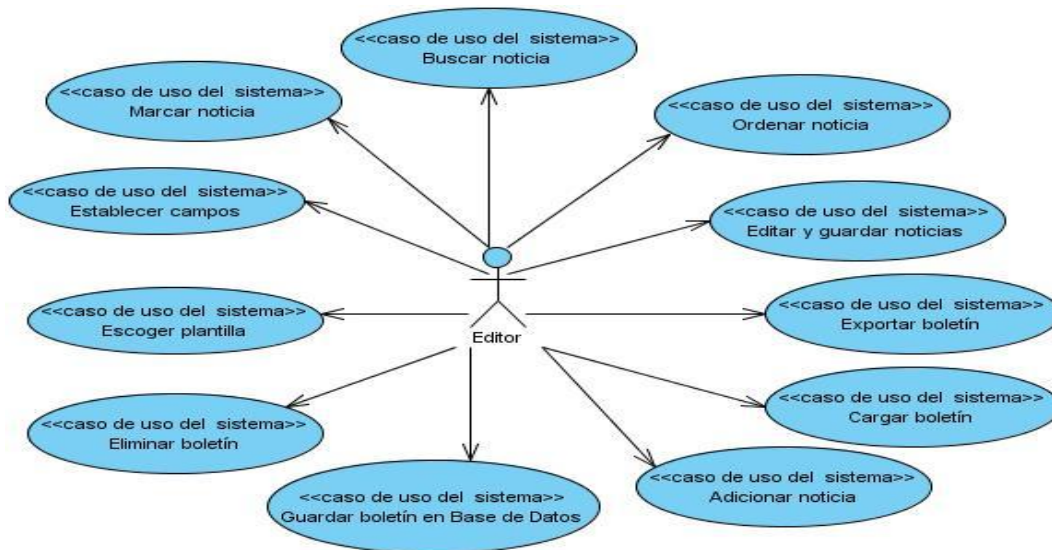
una librería de software utilizada como herramienta para incorporar capacidades de búsqueda a las aplicaciones.

2.5 Modelo de casos de uso.

Debido ha que se han redefinido los requisitos funcionales establecidos en el trabajo de diploma Sistema de Vigilancia Tecnológica para el proyecto Operación Verdad: Análisis y Diseño del módulo “Edición y Generación de Informes” (Socarras, 2008), ya que las actividades que realiza el grupo de Vigilancia Tecnológica y Política han sufrido cambios desde la fecha de creación de ese trabajo hasta la actualidad, se va a mostrar nuevamente el modelo de casos de uso del sistema, y finalmente se van a describir los casos de uso por separado para una mejor comprensión.

El editor es el único actor que interactúa con el módulo, iniciando todos los casos de uso del sistema. En la figura se observan los casos definidos para representar el flujo de eventos que permiten obtener un resultado de valor para los editores durante el diseño del boletín.

Imagen 5. Diagrama de casos de uso del sistema.



Actor presente en el sistema:

Tabla 1: Actor del sistema

Actor	Descripción
-------	-------------

Editor	Es el encargado de realizar todas las actividades en el Módulo de Generación Automática de Boletines, es quien interactúa con todos los casos de uso del sistema.
--------	---

2.5.1 Descripción de los casos uso del sistema.

Describe cómo se lleva a cabo y se ejecuta un caso de uso determinado, en términos de las clases del análisis y de sus objetos en interacción.

Antes de describir cada caso de uso del sistema, se explica a continuación como el Editor llega a interactuar con el Módulo de Generación Automática de Boletines, para lo cual sigue los siguientes pasos:

1. Se registra en el Sistema de Monitoreo y Análisis de Noticias (SIMAN) mediante un usuario y una contraseña que se encuentran en una Base de Datos DB4O, en el caso de la contraseña se guarda encriptada con el algoritmo de encriptación DES. En caso de que la Base de Datos no exista, el editor tiene la posibilidad de crear una.
2. Ya registrado, el primer módulo que muestra el sistema es el Módulo de Búsqueda y Descarga de noticias, el cual busca y descarga Noticias mediante canales de sindicación, y la guarda en la Base de Datos donde mismo se guardan los datos para registrarse en el sistema.
3. El sistema tiene un segundo módulo llamado Módulo de Generación Automática de Boletines el cual es accesible por el Editor, donde interactúa con el mismo mediante los casos de usos, que se describen a continuación.

- **Tabla 2: Descripción Caso de Uso Marcar noticia.**

CU-1	Marcar noticia
Actores:	Editor

Propósito:	Escoger las primeras noticias candidatas, que va a contener el boletín en el Módulo Búsqueda y Descarga de SIMAN.	
<u>Resumen:</u> El Editor escoge en el Módulo Búsqueda y Descarga de SIMAN noticias que van a ser las primeras candidatas, a formar parte del boletín que se desea diseñar.		
Referencias:	R1	
Precondiciones:	Debe existir al menos una noticia en la Base de Datos que pertenezca al Editor logueado en el sistema en ese momento.	
<u>Acción del actor</u>		<u>Respuesta del sistema</u>
1. El editor selecciona las noticias en el Módulo de Búsqueda y Descarga de SIMAN.		2. El Sistema va mostrando las noticias que son escogidas por el Editor. 3. El Sistema guarda las noticias mascadas en una lista que se encuentra en memoria.

- **Tabla 3: Descripción Caso de Uso Buscar noticia.** ([Ver Anexo 1](#))
- **Tabla 4: Descripción Caso de Uso Organizar Noticia.** ([Ver Anexo 2](#))
- **Tabla 5: Descripción Caso de Uso Establecer**

campos. ([Ver Anexo 3](#))

- **Tabla 6: Descripción Caso de Uso Editar y guardar noticias.** ([Ver Anexo 4](#))
- **Tabla 7: Descripción Caso de Uso Escoger plantilla.** ([Ver Anexo 5](#))
- **Tabla 8: Descripción Caso de Uso Exportar boletín.** ([Ver Anexo 6](#))
- **Tabla 9: Descripción Caso de Uso Guardar boletín en Base de Datos.** ([Ver Anexo 7](#))
- **Tabla 10: Descripción Caso de Uso Cargar boletín.** ([Ver Anexo 8](#))
- **Tabla 11: Descripción Caso de Uso Eliminar boletín.** ([Ver Anexo 9](#))
- **Tabla 12: Caso de Uso Adicionar noticia.** ([Ver Anexo 10](#))

2.7 Vista de Despliegue.

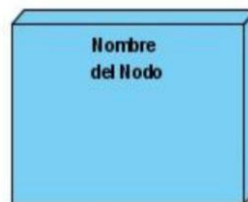
Para el desarrollo de una aplicación informática cualquiera, el arquitecto de software debe prestarle especial atención a la configuración de hardware en la cual se desplegará el sistema, identificando nodos procesadores (computadoras), dispositivos y protocolos; lo que en su conjunto conforman el modelo de despliegue, el cual define la arquitectura física del sistema por medio de nodos interconectados. Estos nodos son elementos hardware sobre los cuales pueden ejecutarse los elementos software y para representar esta información se emplea el elemento de UML denominado Diagrama de Despliegue.

Un diagrama de despliegue muestra la configuración de los procesos en tiempo de corrida y los componentes que se encuentran en los nodos. Los nodos son elementos físicos, que existen en tiempo de ejecución y representan un recurso computacional que generalmente tienen alguna memoria y, a menudo, capacidad de procesamiento. Los nodos sirven para modelar la topología del hardware sobre el que se ejecuta el sistema.

Un nodo representa normalmente un procesador o un dispositivo sobre el que se pueden desplegar los componentes. Un nodo debe tener un nombre asignado que lo distinga del resto de nodos.

Y se representan de la siguiente forma:

Imagen 2. Nodo de un Diagrama de Despliegue.



Mientras que los componentes pertenecen al mundo físico, es decir, representan un bloque de construcción al modelar aspectos físicos de un sistema. Una característica básica de un componente es que:

Debe definir una abstracción precisa con una interfaz bien definida, y permitiendo reemplazar fácilmente los componentes más viejos con otros más nuevos y compatibles.

Existen básicamente tres tipos de componentes:

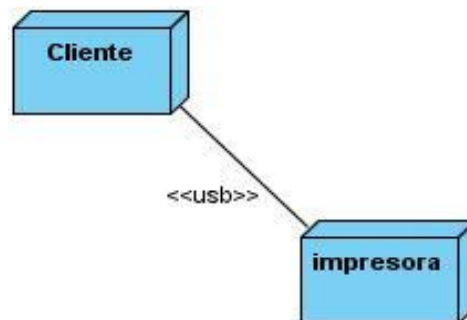
Componentes de despliegue: componentes necesarios y suficientes para formar un sistema ejecutable, como pueden ser las bibliotecas dinámicas (DLLs) y ejecutables (EXEs).

Componentes producto del trabajo: estos componentes son básicamente productos que quedan al final del proceso de desarrollo. Consisten en archivos de código fuente y de datos a partir de los cuales se crean los componentes de despliegue.

Componentes de ejecución: son componentes que se crean como consecuencia de un sistema en ejecución. Es el caso de un objeto que se instancia (o se crea) a partir de una DLL. (GRAU, 2009)

En el caso del Módulo de Generación Automática de SIMAN se cuenta con una PC cliente, en la cual estará corriendo el sistema (SIMAN), en esta PC cliente se encuentra la base de datos, y es opcional tener una impresora para imprimir los boletines generados por este módulo en caso que se quieran llevar a formato duro. A continuación se muestra el Diagrama de Despliegue correspondiente al Módulo de Generación Automática de Boletines de SIMAN:

Imagen 3. Diagrama de Despliegue.



2.8 Conclusiones.

En este capítulo se presentaron los elementos de arquitectura utilizados para llevar a cabo el desarrollo del Módulo de Generación Automática de Boletines del Sistema de Monitoreo y Análisis de Noticias. Abordando los requerimientos no funcionales imprescindibles para que el sistema funcione correctamente y con buen rendimiento, se fundamentó la arquitectura a utilizar, también se realizó un análisis crítico de

posibles implementaciones y componentes existentes que se reutilizaron, así como las estrategias de integración del mismo, aspectos de seguridad, la vista de despliegue del sistema y por último pero no menos importante el estándar de codificación a utilizar.

Capítulo 3. Descripción y Análisis de la solución propuesta. Pruebas.

En el presente capítulo se expone la descripción y se analiza la solución propuesta para desarrollar la aplicación modelándose los artefactos necesarios que contribuyen a la implementación del sistema. Se refinan los principales diagramas de clases y secuencia del modelo de análisis y diseño. Se explica con detalles la forma en que desarrollarse las plantillas con IReport. Además, se explican los dos tipos de pruebas de calidad que se le efectuaron al módulo.

3.1. Modelo de Diseño.

La fase de diseño (y los modelos UML resultantes) expande y detalla los modelos de análisis tomando en cuenta todas las implicaciones y restricciones técnicas. El propósito del diseño es especificar una solución que trabaje y pueda ser fácilmente convertida en código fuente y construir una arquitectura simple y fácilmente extensible.

En el diseño se modela el sistema y se le da forma (incluida la arquitectura) para que soporte todos los requisitos y las restricciones que se le suponen. La entrada principal para el diseño es el resultado del análisis, o sea, el modelo de análisis, que proporciona una comprensión detallada de los requisitos. Además de plantear una estructura del sistema la cual debe mantenerse lo más fielmente posible cuando se le dé forma al sistema.

3.2. Diagrama de clases del Diseño.

Clases del Diseño: Una clase de diseño es una abstracción de una clase o construcción en la implementación del sistema.

Diagrama de Clases del Diseño: Una clase de diseño es una construcción similar en la implementación del sistema. Los diagramas de clases de diseño exponen un conjunto de interfaces, colaboraciones y sus relaciones. Se utilizan para modelar la vista de diseño estática de un sistema.

Al igual que los demás diagramas, los diagramas de clases pueden contener notas y restricciones. Los diagramas de clases también pueden contener paquetes o subsistemas, los cuales se usan para agrupar los elementos de un modelo en partes más grandes.

Dadas las características particulares del sistema por ser una aplicación de escritorio se mostrará el diagrama de clases de una manera general agrupando la interrelación de todas clases. Además de representar todos los cambios llevados a cabo en el diagrama de clases debido al refinamiento que se le realizó al diseño del sistema.

Imagen 6. Diagrama de Clases del Diseño General.



La clase CE_Noticia pertenece al Módulo de Búsqueda y Descarga de Noticias de SIMAN. La cual se usó porque cumplía con los requisitos del Módulo de Generación Automática de Boletines. Se refleja en el diagrama de clases del diseño general cual es el uso que se le da a esta clase.

3.3. Descripción de las Clases.

La descripción de las clases da una idea más específica de cómo estarán constituidas las clases. A continuación se muestra la descripción de la clase CC_Controlador.

Tabla 13. Descripción de la clase CC_Controlador.

Nombre: CC_Controlador	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
cont	CC_Controlador
buscador	CC_Buscador
lista_noticias	List<CE_Noticia>
copia_lista_noticias	List<CE_Noticia>
boletín	CE_Boletin
fecha1	Calendar
fecha2	Calendar
palabra	String
dirección	String
texto	String
pasos	int
noticia	CE_Noticia
virar	boolean
títulos	List<String>

lista_boletines	List<CE_Boletin>
lista_boletines_eliminar	List<CE_Boletin>
estado_guardar	int
Para cada responsabilidad	
Nombre:	Singleton()
Descripción:	Devuelve en objeto de la clase controladora solo si este no se ha creado antes, lo que permite que solo se cree una sola instancia de la clase CC_Controlador
Nombre:	AdicionarNoticia(CE_Noticia e)
Descripción:	Inserta un objeto de la clase CE_Noticia en la lista de noticias que se encuentra en la clases CC_Controlador
Nombre:	getDescripcionNoticia(int fila,JTable tabla, JEditorPane editor)
Descripción:	Permite tomar el índice que se selecciona de la JTable y editar el texto del JeditorPane.
Nombre:	buscarPorParametros(String título, String autor, String fecha)
Descripción:	Busca por parámetros una noticia en una lista.
Nombre:	EliminarRepetidos(List<CE_Noticia> lista)
Descripción:	Permite eliminar los elementos repetidos de la lista que se pasa por parámetro.
Nombre:	DevuelveNoticia(String ptitulo)
Descripción:	Permite buscar en la base de datos la noticia dado el titulo de la misma.

Nombre:	NoticiasMarcadas()
Descripción:	Busca las noticias marcadas en la base de datos.
Nombre:	NoticiaPorFecha(Calendar u,Calendar d)
Descripción:	Dado dos fechas devuelve procedente de la base de datos las noticias que se hayan guardado entre ese rango de fechas.
Nombre:	DameNoticiasQueCoincidanTitulo(String pvalorBuscar)
Descripción:	Devuelve dado un valor las noticias que lo contengan en su título.
Nombre:	DameNoticiasQueCoincidanDescrip(String pvalorBuscar)
Descripción:	Devuelve dado un valor las noticias que lo contengan en su descripción.
Nombre:	DameNoticiasQueCoincidanClaves(String pvalorBuscar)
Descripción:	Devuelve dado un valor las noticias que lo contengan en sus palabras claves.
Nombre:	AplicarBusquedas(JTable tabla)
Descripción:	Inicia las búsquedas que se hayan habilitado.
Nombre:	AcoplarNoticias()
Descripción:	Pasa los datos de la lista de noticias a un String.
Nombre:	isRepetido(CE_Noticia n)
Descripción:	Verifica que una noticia no se repita en la lista de noticias.
Nombre:	EliminarBoletín(CE_Boletin boletín)

Descripción:	Elimina el boletín pasado por parámetro de la base de datos.
Nombre:	Clasificar(JList listaBoletines, int estado)
Descripción:	Muestra los boletines en la lista según la clasificación que puede ser por nombres, autor, o fecha.
Nombre:	EliminarTodos()
Descripción:	Elimina todos los boletines de la Base de Datos.

- **Tabla 14. Descripción de la clase CC_ControladorVisual.** ([Ver Anexo 11](#))
- **Tabla 15. Descripción de la clase CC_HiloAcoplarNoticias.** ([Ver Anexo 12](#))
- **Tabla 16. Descripción de la clase CC_HiloBoletines.** ([Ver Anexo 13](#))
- **Tabla 17. Descripción de la clase CC_HiloGenerarDiseno.** ([Ver Anexo 14](#))
- **Tabla 18. Descripción de la clase CC_HiloGenerarCarga.** ([Ver Anexo 15](#))
- **Tabla 14. Descripción de la clase CC_HiloNoticiasClaves.** ([Ver Anexo 16](#))
- **Tabla 19. Descripción de la clase CC_HiloNoticiasDescripción.** ([Ver Anexo 17](#))
- **Tabla 20. Descripción de la clase CC_HiloNoticiasFecha.** ([Ver Anexo 18](#))
- **Tabla 21. Descripción de la clase HiloNoticiasMarcadas.** ([Ver Anexo 19](#))
- **Tabla 22. Descripción de la clase CC_HiloNoticiasTítulo.** ([Ver Anexo 20](#))

3.4. Diagrama de Interacción.

Un diagrama de interacción consiste en un conjunto de objetos y sus relaciones, incluyendo los mensajes que se pueden enviar entre ellos. Pueden utilizarse para visualizar, especificar, construir y

documentar la dinámica de una sociedad particular de objetos, o se pueden utilizar para modelar un flujo de control particular de un caso de uso. Los diagramas de interacción no son sólo importantes para modelar los aspectos dinámicos de un sistema, sino también para construir sistemas ejecutables por medio de ingeniería directa e inversa. Los diagramas de colaboración y los diagramas de secuencia son diagramas de interacción.

Un diagrama de colaboración en las versiones de UML 1.x es esencialmente un diagrama que muestra interacciones organizadas alrededor de los roles. A diferencia de los diagramas de secuencia, los diagramas de comunicación muestran explícitamente las relaciones de los roles. Por otra parte, un diagrama de comunicación no muestra el tiempo como una dimensión aparte, por lo que resulta necesario etiquetar con números de secuencia tanto la secuencia de mensajes como los hilos concurrentes.

- Muestra cómo las instancias específicas de las clases trabajan juntas para conseguir un objetivo común.
- Implementa las asociaciones del diagrama de clases mediante el paso de mensajes de un objeto a otro. Dicha implementación es llamada "enlace".

A continuación se muestran los diagramas de colaboración del sistema.

Diagrama de Colaboración 1. Marcar noticia

Int_Pantalla y CC_Sistema pertenecen al módulo de búsqueda de SIMAN ya que las noticias se seleccionan en el para ser mostradas en el módulo de generación de boletines

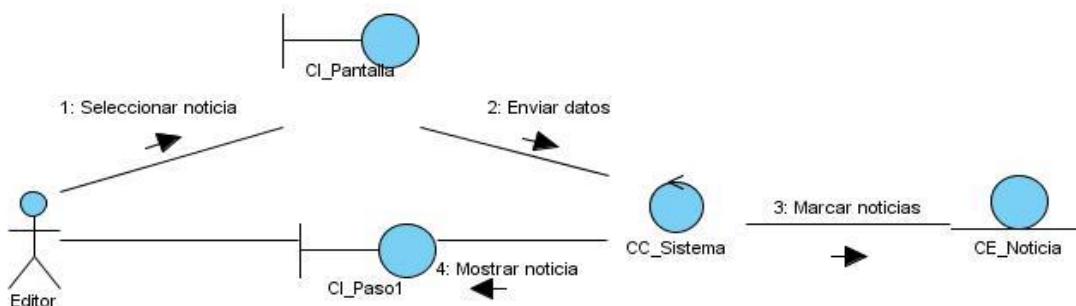


Diagrama de Colaboración 2. Buscar noticia. [\(Ver Anexo 21\)](#)

Diagrama de Colaboración 3. Organizar noticia. [\(Ver Anexo 22\)](#)

Diagrama de Colaboración 4. Establecer campos. [\(Ver Anexo 23\)](#)

Diagrama de Colaboración 5. Editar y guardar noticias. [\(Ver Anexo 24\)](#)

Diagrama de Colaboración 6. Escoger plantilla. [\(Ver Anexo 25\)](#)

Diagrama de Colaboración 7. Exportar boletín. [\(Ver Anexo 26\)](#)

Diagrama de Colaboración 8. Guardar boletín en Base de Datos. [\(Ver Anexo 27\)](#)

Diagrama de Colaboración 9. Cargar boletín. [\(Ver Anexo 28\)](#)

Diagrama de Colaboración 10. Eliminar boletín. [\(Ver Anexo 29\)](#)

Diagrama de Colaboración 11. Adicionar noticia. [\(Ver Anexo 30\)](#)

3.5 Diseño de Boletines con IReport.

El Módulo de Generación Automática de Boletines utiliza plantillas para aplicarle un estilo específico a cada boletín que es generado por el usuario, estas plantillas son diseñadas en IReport lo que permite un rápido modelamiento y diseño de las mismas. Se estableció un estándar para el diseño de las plantillas, lo que garantizará que las mismas puedan ser cargadas sin dar como resultado ningún error. El estándar de diseño se basa en aspectos como el nombre del directorio, donde se va a encontrar las plantillas, el número de imágenes que va a contener, con nombres, extensiones y tamaños específicos, y los nombres y las extensiones de las plantillas.

Explicación de los aspectos de diseño.

- ✓ El directorio principal debe de llamarse “Plantillas”, el mismo es el que va a contener otros directorios que van a ser de las plantillas.
- ✓ El reporte principal debe llamarse Boletín y el subreporte debe llamarse “Noticia, y la extensión de los dos es”.jasper”.
- ✓ Las imágenes que debe tener cada plantilla son:
 - Presentacion.jpg (595x842)
 - Fondo.jpg (595x842)

Las carpetas de las plantillas deben estar a un nivel inmediato dentro de la carpeta principal “Plantillas” y los archivos requeridos deben estar dentro de ellas. Los directorios de las plantillas se pueden llamar como lo desee el que hace la plantilla.

La plantilla Boletín tiene 4 fields:

- título: de tipo String, es quien va a mostrar el título del boletín en la página principal.
- autor: de tipo String, es quien va a mostrar el nombre del autor del boletín en la página principal.
- descripción: de tipo String, es quien va a mostrar la descripción del boletín en la página principal.
- listaNoticias: de tipo `net.sf.jasperreports.engine.JRDataSource` es quien va a contener la lista de noticias que se va a mostrar en el boletín.

Además, tiene 3 parámetros:

- presentación: de tipo String, es el parámetro que se localiza en la propiedad Image Expression de la imagen Presentación.jpg, que se ubica en la página principal del boletín, al que por programación, se le pasa la ubicación de la imagen.
- fondo: de tipo String, es el parámetro que se localiza en la propiedad Image Expression de la imagen Fondo.jpg, que se ubica en las páginas del boletín, al que por programación, se le pasa la ubicación de la imagen.
- noticias: de tipo String, es el parámetro el que por programación se le pasa la ubicación del subreporte Noticia.jasper.

La plantilla Noticia tiene 2 fields:

- id: de tipo Integer, el cual es utilizado por el reporte Boletín, para ir enumerando las noticias.
- texto: de tipo String, el cual va a mostrar todo el texto, quien va a contener todas las noticias.

Imagen 7. Métodos programados para generar los boletines.

```
55
56     JasperReport reporte = (JasperReport) JRLoader.loadObject(CC_Controlador.Singleton().getDireccion()+
57     "Boletín.jasper");
58
59     Map subreporte = new HashMap();
60
61     subreporte.put("noticia",CC_Controlador.Singleton().getDireccion()+"Noticia.jasper");
62     subreporte.put("presentacion",CC_Controlador.Singleton().getDireccion()+"Presentación.jpg");
63     subreporte.put("fondo",CC_Controlador.Singleton().getDireccion()+"Fondo.jpg");
64
65     JasperPrint jasperPrint = JasperFillManager.fillReport(reporte, subreporte,
66     new JRBeanCollectionDataSource(listaBoletas));
67
```

Como se muestra en la figura en la línea 56 se carga la plantilla Boletín.jasper, luego en la 59 se crea un HashMap el cual va a almacenar todos los parámetros de la plantilla principal, en las líneas 61,62,63 se cargan el sub-reporter Noticia.jasper, y se llenan los parámetros de la Imagen Presentación.jpg y Fondo.jpg respectivamente. Más adelante en la 65 se crean el objeto principal para crear reportes, pasando por parámetro el reporte principal, el HashMap y la lista de noticias.

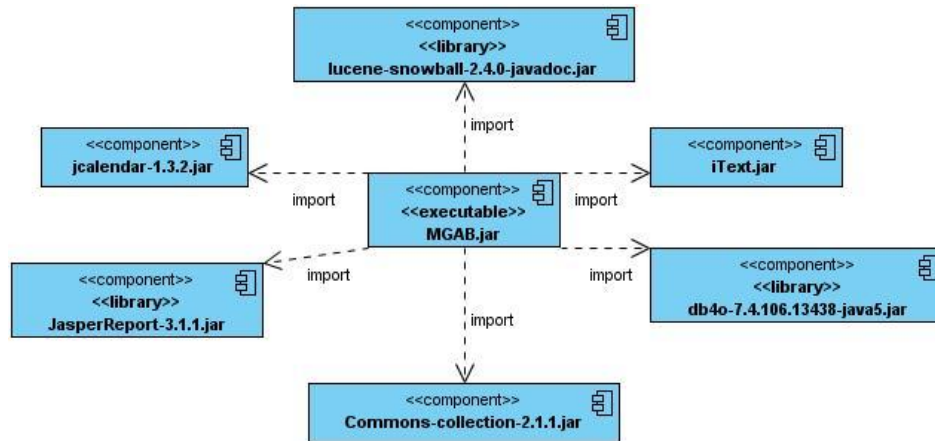
Todos estos aspectos deben cumplirse para tener una satisfactoria generación del boletín, porque en caso de no hacerse así deja de generarse, y si lo llegase a hacer le faltará algún componente, lo que traerá que los usuarios que lean el mismo no entenderán algunos aspectos de importancia.

3.6. Diagrama de Componentes.

Un diagrama de componentes muestra las organizaciones y dependencias lógicas entre componentes software, sean éstos componentes de código fuente, binarios o ejecutables. Desde el punto de vista del diagrama de componentes se tienen en consideración los requisitos relacionados con la facilidad de desarrollo, la gestión del software, la reutilización, y las restricciones impuestas por los lenguajes de programación y las herramientas utilizadas en el desarrollo. Los elementos de modelado dentro de un diagrama de componentes serán componentes y paquetes.

Para el desarrollo del Módulo de Generación Automática de Boletines del Sistema de Monitoreo y Análisis de Noticias se emplearon varias librerías las cuales son consideradas componentes del sistema, en conjunto con el ejecutable general del mismo. Por lo que el diagrama de componentes quedaría de la siguiente forma:

Imagen 8. Diagrama de componentes del Módulo.



3.7 Trabajo con DB4O y Lucene.

El trabajo con la base de datos orientada a objetos DB4O fue muy sencillo ya que solo se realizaron dos consultas, una que devolvía todas las noticias marcadas y la otra que devolvía todas las noticias que se encontraban en el rango de dos fechas. El primero se utilizó en el paso 1 del diseño de boletines y el 2do en el filtro por fechas.

A continuación se muestran las consultas:

Imagen 9. Primera consulta realizada.

```
1454 public void getListamarcadas() {
1455     List<CE_Noticia> lista_vacia = new ArrayList<CE_Noticia>();
1456     CC_Controlador.Singleton().setLista_noticias(lista_vacia);
1457
1458     contenedor.query(new Predicate<CE_Noticia>() {
1459
1460         @Override
1461         public boolean match(CE_Noticia noticia) {
1462
1463             if(noticia.getMarcada()) {
1464                 CC_Controlador.Singleton().AdicionarNoticia(noticia);
1465                 System.out.print(CC_Controlador.Singleton().getListamarcadas().size());
1466             }
1467
1468             return noticia.getMarcada();
1469         }
1470     });
1471 }
1472
1473 }
```

Se ve en la línea 1455 como se crea una lista de noticias en la cual se van a guardar todas las noticias. Luego en la línea 1456 se vacía la lista de noticias de la clase CC_Controlador, pasándole la nueva lista creada en la línea anterior. Luego desde la línea 1458 a la 1471 se encuentra la consulta DB4O, que va a consistir en devolver todas las noticias que estén en la base de datos que cumplan con la condición de estar marcadas, las cuales se van adicionando por cada iteración a la lista de la clase CC_Controlador.

Imagen 10. Segunda consulta realizada.

```
1475 public List<CE_Noticia> getListaPorFecha(Calendar u, Calendar d) {
1476
1477     List<CE_Noticia> noticias = new ArrayList<CE_Noticia>();
1478     final Calendar e = u;
1479     final Calendar f = d;
1480
1481     ObjectSet resultado = contenedor.query(new Predicate<CE_Noticia>() {
1482
1483         @Override
1484         public boolean match(CE_Noticia noticia) {
1485
1486             Calendar antes = CC_Convertir.Singleton().StringACalendar(noticia.GetFechaAdicionBD());
1487             return (noticia.GetFechaAdicionBD() != null && (antes.after(e) && antes.before(f)));
1488         }
1489     });
1490
1491     while(resultado.hasNext())
1492     {
1493         noticias.add((CE_Noticia)resultado.next());
1494     }
1495
1496     return noticias;
1497 }
```

En esta consulta se devuelve una lista de noticias. Se ve en la línea 1477 como se crea una lista de noticias en la cual se van a guardar todas las noticias. En las líneas 1478 y 1479 se crean dos objetos de Calendar los cuales van a tomar los valores de las fechas que se pasan por parámetros. Luego desde la línea 1481 a la 1489 se encuentra la consulta DB4O, que va a consistir en devolver todas las noticias que estén en la base de datos que cumplan con la condición de estar en el rango de las dos fechas que se pasan por parámetros. Más adelante en la línea 1491 hay un while el cual va a llenar la lista creada anteriormente con las noticias que se devuelve en la consulta, y finalmente se retorna la lista en la línea 1496.

El trabajo con la librería Lucene solo consistió en utilizar los siguientes métodos pertenecientes a SIMAN:

Imagen 11. Primer método redefinido.

```
49 public void Indexar(List<CE_Noticia> pnoticias)
50 throws Exception
51
52 Analyzer analizador = new SnowballAnalyzer("Spanish", stopWords.GetStopWords());
53 IndexWriter writer = new IndexWriter(directorioIndexacion, analizador, true,
54 IndexWriter.MaxFieldLength.UNLIMITED);
55 for(int i = 0; i < pnoticias.size(); i++)
56 {
57     Document documento = new Document();
58     String titulo = pnoticias.get(i).GetTitulo();
59     String descripcion = pnoticias.get(i).GetDescripcion();
60     String palabrasClaves = pnoticias.get(i).GetPalabrasClaves();
61     String fechaPublicacion = pnoticias.get(i).GetFechaPublicacion();
62     String autor = pnoticias.get(i).GetAutor();
63     String url = pnoticias.get(i).GetURL();
64
65
66     documento.add(new Field("titulo", titulo, Field.Store.YES, Field.Index.ANALYZED));
67     documento.add(new Field("descripcion", descripcion, Field.Store.YES, Field.Index.ANALYZED));
68     if(palabrasClaves != null)
69     {
70         documento.add(new Field("palabrasClaves", palabrasClaves, Field.Store.YES, Field.Index.ANALYZED));
71     }
72     documento.add(new Field("fechaPublicacion", fechaPublicacion, Field.Store.YES, Field.Index.ANALYZED));
73     documento.add(new Field("autor", autor, Field.Store.YES, Field.Index.ANALYZED));
74     documento.add(new Field("url", url, Field.Store.YES, Field.Index.ANALYZED));
75     writer.addDocument(documento);
76 }
77 writer.optimize();
78 writer.close();
79 }
```

Este método de halla explicado en el trabajo de diploma Implementación del Módulo de Búsqueda del Sistema de Monitoreo y Análisis de Noticias (Muñoz, 2009), pero se muestra nuevamente porque se le hizo una redefinición, la cual consiste en que se agregaron los campos: palabrasClaves, fechaPublicacion, autor y URL, en las líneas 60, 61, 62, y 63 respectivamente.

Imagen 12. Segundo método redefinido.

```
46 public List<CE_Noticia> RealizarBusqueda()
47 throws Exception(
48     List<CE_Noticia> noticias = new ArrayList<CE_Noticia>();
49     String [] camposBuscar = new String[6];
50     camposBuscar[0] = "titulo";
51     camposBuscar[1] = "descripcion";
52     camposBuscar[2] = "palabrasClaves";
53     camposBuscar[3] = "fechaPublicacion";
54     camposBuscar[4] = "autor";
55     camposBuscar[5] = "url";
56     IndexSearcher buscador = new IndexSearcher(directorioBuscar);
57     Analyzer analizador = new SnowballAnalyzer("Spanish", stopWords.GetStopWords());
58     BooleanClause.Occur[] banderas = {BooleanClause.Occur.SHOULD, BooleanClause.Occur.SHOULD,
59     BooleanClause.Occur.SHOULD, BooleanClause.Occur.SHOULD, BooleanClause.Occur.SHOULD,
60     BooleanClause.Occur.SHOULD};
61     Query consulta = MultiFieldQueryParser.parse(parametro, camposBuscar, banderas, analizador);
62     TopDocs documentos = buscador.search(consulta, null, 100);
63     ScoreDoc [] hits = documentos.scoreDocs;
64     for(int i = 0; i < hits.length; i++){
65         CE_Noticia noticiaAuxiliar = new CE_Noticia();
66         int docId = hits[i].doc;
67         Document documento = buscador.doc(docId);
68         noticiaAuxiliar.SetTitulo(documento.get("titulo"));
69         noticiaAuxiliar.SetDescripcion(documento.get("descripcion"));
70         noticiaAuxiliar.SetPalabrasClaves(documento.get("palabrasClaves"));
71         noticiaAuxiliar.SetFechaPublicacion(documento.get("fechaPublicacion"));
72         noticiaAuxiliar.SetAutor(documento.get("autor"));
73         noticiaAuxiliar.SetURL(documento.get("url"));
74         noticias.add(noticiaAuxiliar);
75     }
76     buscador.close();
77     return noticias;
78 }
```

Este método de halla explicado en el trabajo de diploma Implementación del Módulo de Búsqueda del Sistema de Monitoreo y Análisis de Noticias (Muñoz, 2009), pero se muestra nuevamente porque se le hizo una redefinición, la cual consiste en que se agregaron los campos: palabrasClaves, fechaPublicacion, autor y URL, en las líneas 52, 53, 54, y 55 respectivamente.

Los métodos anteriores fueron utilizados en los filtros por palabras aprovechando el potencial de búsqueda que posee Lucene.

3.8 Prueba.

El aseguramiento de la calidad del software se ha convertido en una necesidad prioritaria y en una tarea vital en el desarrollo de cualquier sistema informático, por la importancia que tiene que garantizar el tiempo de desarrollo, el costo y la calidad del producto. Por esta razón es necesario establecer un conjunto de pruebas que garanticen que el software cumpla con los requisitos especificados y que no presente errores.

Para determinar la calidad de un producto de software se deben efectuar medidas y desarrollar actividades que permitan comprobar el grado de cumplimiento de las especificaciones iniciales del sistema, y es aquí donde las pruebas de software desempeñan un papel fundamental. Estas verifican el desarrollo que va alcanzando el producto durante todas sus etapas, identificando posibles fallos de implementación, calidad, o usabilidad de un programa.

Según Roger Pressman, en su libro Un Enfoque Práctico, las pruebas del software son 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.¹

Una de las características típicas durante el desarrollo de un software basado en el ciclo de vida de RUP es la realización de controles periódicos. Estos controles pretenden evaluar la calidad de los productos generados para poder detectar posibles defectos cuanto antes. Todo sistema o aplicación, independientemente de estas revisiones, debe ser probado mediante su ejecución, controlado antes de ser entregado al cliente. Estas ejecuciones o ensayos de funcionamiento, posteriores a la terminación del código del software, se denominan habitualmente pruebas.

¹ [Citado el: 2 de mayo de 2010.] <http://www.informaticamilenium.com.mx/Paginas/espanol/sitioweb.htm>.

La IEEE (Standard Glossary of Software Engineering Terminology) plantea que la prueba es la actividad en la cual un sistema o componente es ejecutado bajo condiciones específicas, se observan y almacenan los resultados y se realiza una evaluación de algún aspecto del sistema o componente. Según RUP las pruebas son una disciplina en el proceso de ingeniería de software cuyo objetivo es integrar y poner a prueba el sistema.

Estos son algunos de los principales conceptos de prueba manejados actualmente en el mundo del software. Todos ellos apuntan a la significativa importancia de este proceso durante todo el desarrollo de un producto. Se considera que la prueba de software es una etapa imprescindible durante todo el proceso de desarrollo, pues una vez que se genera código fuente, el software debe ser probado para descubrir y corregir el máximo número de errores posibles antes de su entrega al cliente.

3.8.1 Pruebas aplicadas.

Las pruebas aplicadas al software fueron las Unitarias. La misma tiene fundamentalmente dos enfoques: Pruebas de Caja Blanca (o Pruebas Estructurales) y las Pruebas de Caja Negra (o Pruebas Funcionales).

Las **pruebas de Caja Blanca** suelen ser llamadas estructurales o de cobertura lógica. En ellas se pretende investigar sobre la estructura interna del código, exceptuando detalles referidos a datos de entrada o salida, para probar la lógica del programa desde el punto de vista algorítmico. Realizan un seguimiento del código fuente según se va ejecutando los casos de prueba, determinándose de manera concreta las instrucciones, bloques, etc. que han sido ejecutados por los casos de prueba. Es por ello que se considera a la prueba de Caja Blanca como uno de los tipos de pruebas más importantes que se le aplican a los software, logrando como resultado que disminuya en un gran por ciento el número de errores existentes en los sistemas y por ende una mayor calidad y confiabilidad.

Existen varios métodos de prueba de Caja Blanca, entre los que se encuentra: Prueba de camino básico que es la que se empleó para verificar que el software no presentara errores. Es una técnica que permite al diseñador de casos de prueba obtener una medida de la complejidad lógica de un diseño procedimental y usar esa medida

como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba obtenidos del conjunto básico garantizan que durante la prueba se ejecute por lo menos una vez cada sentencia del programa.²

Las **pruebas de Caja Negra** demostrarán que las funciones del software son operativas, que la entrada se acepta de forma adecuada, que se produce una salida correcta, y que se mantiene la integridad de la información externa. Estas pruebas se centran en los requisitos funcionales, se llevan a cabo sobre la interfaz del software y son completamente indiferentes al comportamiento interno y la estructura del programa. La prueba de Caja Negra no es una alternativa a las técnicas de prueba de la Caja Blanca, sino un enfoque complementario que intenta descubrir diferentes tipos de errores a los encontrados en los métodos de la Caja Blanca.

Las Pruebas de la Caja Negra intentan encontrar errores de: funciones incorrectas o inexistentes, errores relativos a las interfaces, errores en estructuras de datos o en accesos a bases de datos externas, errores debidos al rendimiento, errores de inicialización o terminación.

Dentro de los métodos de Caja Negra se encuentra la partición equivalente, el cual fue utilizado para la detectar los de errores de la aplicación. Esta técnica divide el dominio de entrada de un programa en clases de datos de los que se pueden derivar casos de prueba. El diseño de estos casos de prueba para la partición equivalente se basa en la evaluación de las clases de equivalencia.

El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada. Regularmente, una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición lógica.

A menudo, se plantea que las pruebas a los software nunca terminan, simplemente se transfiere del desarrollador al cliente. Cada vez que el cliente usa el programa está llevando a cabo una prueba. Aplicando el diseño de casos de pruebas al software en cuestión se puede conseguir una prueba más completa y descubrir y corregir el mayor número de errores antes de que comiencen las “*pruebas del cliente*”.

² [Citado el: 2 de mayo de 2010.] <http://www.informaticamilenium.com.mx/Paginas/espanol/sitioweb.htm>.

3.8.2 Casos de prueba.

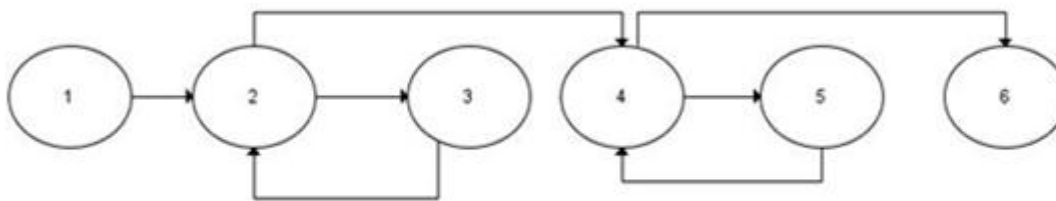
Los casos de pruebas tienen como objetivo comprobar que el producto que se encuentra en desarrollo, satisface realmente las peticiones del usuario final, tal y como se describe en la especificación de los requerimientos y los casos de uso.

Los casos de pruebas no son más que 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.

3.8.2.1 Casos de prueba de Caja Blanca.

A continuación se presentan algunos de los casos de prueba de Caja Blanca realizados a la aplicación.

```
public void actualizarListaTodas (List<CE_Boletin> lista1, List<CE_Boletin> lista2 ,JList uno, JList dos)
```



Complejidad ciclomática.
 $V(G) = 3.$

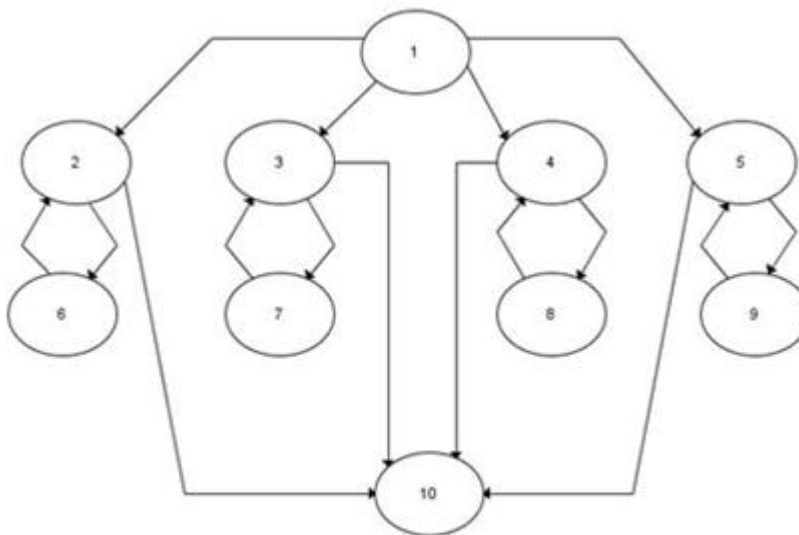
Caminos Independientes.

1-2-4-6.

1-2-3-2-4-6.

1-2-4-5-4-6.

Grafo de flujo del Método ActualizarListaTodas



```
public void Clasificar(JList listaBoletines, int estado)
```

Complejidad ciclomática.
 $V(G) = 8.$

Caminos Independientes.

1-2-3-4-10.

1-2-6-2-3-4-5-10.

1-2-3-7-3-4-5-10.

1-2-6-2-3-7-3-4-5-10.

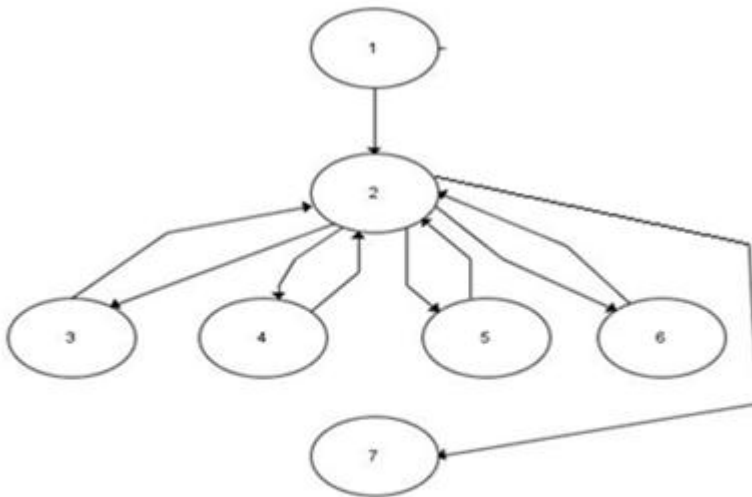
1-2-3-4-8-4-5-10.

1-2-6-2-3-4-8-4-5-10.

1-2-3-7-3-4-8-4-5-10.

1-2-3-7-3-4-8-4-5-9-5-10.

Grafo de flujo del Método Clasificar

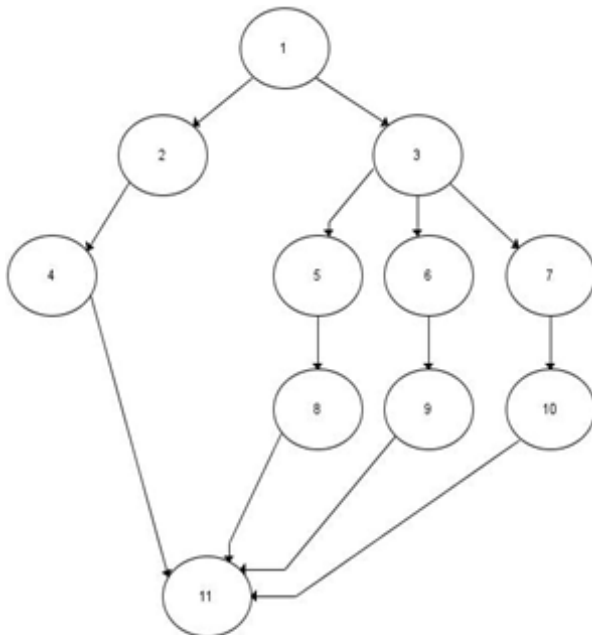


`public void AcoplarNoticias()`

Complejidad ciclomática.
 $V(G) = 5.$

Caminos Independientes.
1-2-7.
1-2-3-2-7.
1-2-4-2-7.
1-3-2-4-2-7.
1-2-5-2-7.

Grafo de flujo del Método AcoplarNoticia



`public void AplicarBusquedas (JTable tabla)`

Complejidad ciclomática.
 $V(G) = 4.$

Caminos Independientes.
1-2-4-11.
1-3-5-8-11.
1-3-6-9-11.
1-3-7-10-11.

Grafo de flujo del Método AplicarBusqueda

Luego de tener elaborados los Grafos de Flujos y los caminos a recorrer, se prepararon los casos de prueba que forzaron la ejecución de cada uno de los caminos. Se escogieron los datos de entradas de forma que todas las condiciones y caminos de cada método se ejecutara con el fin de reflejar los problemas que presentaba la aplicación. En el mismo se compararon los resultados con los esperados.

Tabla 23. Errores encontrados en cada tipo de prueba.

Prueba	Errores Encontrados
Caja Blanca	3
Caja Negra	7

Tabla 24. Prioridad de los errores encontrados.

Prioridad	Cantidad
Alta	3
Media.	2
Baja.	5

3.8.2.2 Casos de prueba de Caja Negra.

A continuación se muestran tanto las clases válidas como inválidas identificadas de algunos de los casos de usos escogidos para la realización de las pruebas de Caja Negra.

Descripción

Este caso de prueba consiste en que el usuario introduce criterios de búsquedas, para encontrar noticias en la Base de Datos.

Flujo Central.

El usuario solicita buscar noticias en la Base de Datos, donde el sistema muestra una pantalla con los campos necesarios para introducir los criterios de búsquedas que pueden ser: si una noticia se encuentra entre dos fechas escogidas por el usuario y si una noticia contiene palabras escogidas por el usuario en el título, en la descripción o en sus palabras claves.

El usuario introduce los datos especificados y el sistema en caso de que exista algún error notifica una respuesta en correspondencia con el error introducido de lo contrario se activan las opciones que el sistema brinda.

Tabla 25. Clases de Equivalencia del CU Buscar noticias.

Condición de entrada	Clases válidas	Clase inválidas
fechas	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> • (1) A – Z • (2) 0-9 • (3) Cumplen con el patrón Ej: May 4, 2010 • (4) Espacios 	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> • (5) extraños tales como: ¡ “ \$ % & / () = ¿ ? Ç * ^ “ _ : ; ; a @
palabras en el título, en la descripción y en las palabras claves	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> • (4) A – Z • (5) Números • (6) Espacios • (7) Comienzan con Espacios • (8) Incluye todo tipo de caracteres extraños tales como: ¡ “ · \$ % & / () = ¿ ? Ç * ^ “ _ : ; ; a 	

Descripción

Este caso de prueba consiste en que el usuario introduce los valores de los campos del boletín y marca los campos de las noticias.

Flujo Central.

El usuario solicita introducir los datos del boletín y marcar los datos de las noticias que quiere que se muestren en el mismo, donde el sistema muestra una pantalla con los campos necesarios para introducir el Título del Documento, el Autor del Documento, marcar si se van a mostrar el Título, Autor, Fecha de publicación, Descripción y URL de las noticias en el boletín.

El usuario introduce los datos especificados y el sistema en caso de que exista algún error notifica una respuesta en correspondencia con el error introducido de lo contrario se activan las opciones que el sistema brinda.

Tabla 26. Clases de Equivalencia del CU Establecer campos.

Condición de entrada	Clases válidas	Clase inválidas
Título del Documento	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> (1) A – Z (2) 0-9 (3) Espacios 	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> (4) extraños tales como: ¡ “ \$ % & / () = ¿ ? Ç * ^ “ _ : ; a @
Autor del Documento	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> (1) A – Z (2) 0-9 (3) Espacios 	Cadenas de caracteres que incluyen: <ul style="list-style-type: none"> (4) extraños tales como: ¡ “ \$ % & / () = ¿ ? Ç * ^ “ _ : ; a @

La ejecución de los Casos de Prueba de Caja Negra reveló principalmente algunos errores en la implementación de la interfaz visual. La mayoría debido a descuidos o a errores en la lógica de programación pero en general los resultados fueron satisfactorios. Finalmente, se corrigieron dichos errores y se comprobó nuevamente el resultado.

3.8.3 Análisis de los defectos encontrados.

Las pruebas realizadas al software permitieron verificar que el mismo cumplía con todo los requisitos funcionales especificados e identificar cuáles fueron las principales deficiencias detectadas. Los Casos de Pruebas que se definieron anteriormente fueron ejecutados satisfactoriamente por lo que se puede plantear de forma general que el sistema sólo contenía algunas fallas de validación, errores ortográficos y de tratamiento de errores. Los distintos errores encontrados que a simple vista no hubieran sido fáciles identificar se produjeron por las siguientes causas:

descuido, falta de comprensión del problema a solucionar y falta de revisión durante la codificación. Los errores fueron clasificados según la prioridad (alta, media, baja), los cuales fueron corregidos en la siguiente tabla.

3.9 Conclusiones.

En este capítulo se realizó una modelación y representación del sistema en términos del diseño, para cada uno de los casos de uso del sistema y por escenarios de los mismos. Lo que permitió lograr una mejor comprensión de las funcionalidades de la solución propuesta. También se identificaron las clases interfaces, de control y de entidad de este flujo de trabajo. Se obtuvieron el diagrama de clases del diseño y los diagramas de secuencia. Se logró dar una explicación breve de los aspectos a tener en cuenta a la hora diseñar una plantilla de boletín. Mediante los casos de prueba se validó el software, verificando el correcto funcionamiento del sistema, comprobándose que no existían errores en las funciones operativas. Posibilitó la obtención de las principales deficiencias que presentaba el software.

Conclusiones.

Con el desarrollo del módulo de Generación Automática de Boletines para el Sistema de Monitoreo y Análisis de Noticias (SIMAN), se cumplió con el objetivo general propuesto. Obteniendo como resultado un módulo integrado a SIMAN, capaz de crear y gestionar boletines informativos; dándole seguimiento al trabajo de diploma titulado Sistema de Vigilancia Tecnológica para el proyecto Operación Verdad: Análisis y Diseño del módulo “Edición y Generación de Informes” (Socarras, 2008). Donde se realizó el análisis y el diseño de dicho módulo, redefiniéndose algunos elementos necesarios para su perfeccionamiento. Logrando con la integración, convertir a SIMAN en un software más funcional, que permite elevar la calidad del trabajo en el Grupo de Investigación y Desarrollo de la Internet. El módulo de Generación Automática de Boletines influirá en la eficiencia del trabajo de los integrantes del Grupo de Monitoreo Tecnológico y Político, de GIDI.

Recomendaciones.

Con el objetivo de lograr mejoras en el funcionamiento del Módulo de Generación Automática de Boletines se recomienda al equipo de Desarrollo de proyecto GIDI:

- Profundizar en el estudio de la herramienta iReport, con vista de establecer posibles cambios o modificaciones a las plantillas de los boletines, para lograr mayor aceptación del usuario.
- Continuar con el proceso de desarrollo del módulo con el fin de incorporarle nuevas funcionalidades, y mejorar las existentes.
 - ✓ Se recomienda al Equipo de Desarrollo del proyecto GIDI incorporar al módulo la funcionalidad de envío por correo electrónico de los boletines, para lograr una mayor interactividad entre los editores y los usuarios.
 - ✓ Se recomienda al Equipo de Desarrollo del proyecto GIDI incorporar al módulo la funcionalidad que permita generar los boletines con un índice de contenidos, para que en los reportes la información sea más fácil de localizar.

Referencias bibliográficas

Referencias bibliográficas.

Socarras Peláez, Arelis y Alfonso Nole, Pedro L. 2008. Sistema de Vigilancia Tecnológica para el proyecto Operación Verdad: Análisis y Diseño del módulo “Edición y Generación de Informes”. Ciudad de la Habana : s.n., 2008.

Muñoz Quintana, Adniel. 2009. Implementación del Módulo de Búsqueda del Sistema . Ciudad de la Habana : s.n., 2009.

Sociedad Española de Cardiología. 1997. doyma. [Online] 1997. [Cited: Abril 10, 2010.].

González, S. F. 2008. fesabid98.florida-uni. [Online] 2008. [Cited: Enero 25, 2010.] http://fesabid98.florida-uni.es/Comunicaciones/f_santamaria/f_santamaria.htm

Anónimo1. freedownloadmanager. [Online] [Cited: Marzo 9, 2010.] http://www.freedownloadmanager.org/es/downloads/Tirador_de_primera_de_Informe_43879_p/.

Microsoft. 2005. msdn.microsoft. [Online] 2005. [Cited: Abril 16, 2010.] [http://msdn.microsoft.com/es-es/library/ms225593\(VS.80\).aspx](http://msdn.microsoft.com/es-es/library/ms225593(VS.80).aspx).

Herrera, Cristain. 2005. adictosaltrabajo. [Online] Abril 29, 2005. [Cited: Enero 16, 2010.] <http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=IReport>.

Real Academia Española1. buscon.rae. [Online] [Cited: Mayo 2, 2010.] http://buscon.rae.es/drael/SrvltConsulta?TIPO_BUS=3&LEMA=boletin.

Anónimo2. Dsecripción de JAVA 2 Platform Micro Edition, pág 2. [Online] [Cited: Abril 12, 2010.] <http://bieec.epn.edu.ec:8180/dspace/bitstream/123456789/1147/5/T%2011010%20CAP%202.p>.

Anónimo3. PROGRAMACIONENJAVA. [Online] [Cited: Enero 11, 2010.] <http://www.programacionenjava.com/blog/2008/06/10/tutorial/tutorial-caracteristicas-de-java>

AGUT, M. R. 2008. Especificación de Requisitos Software según el estándar de IEEE 830. [Online] Marzo 12, 2008. [Cited: Febrero 23, 2010.] http://www3.uji.es/~coltell/Docs/IngSw_Apuntos/ISW_2004_D08.pdf.

Sitio Oficial del Netbeans. NETBEANS. [Online] [Cited: Mayo 12, 2010.] <http://www.netbeans.org/community/releases/60/>.

ISSI, G. 2003. Metodologías Ágiles en el Desarrollo de Software. [Online] 2003. [Cited: Abril 10, 2010.] <http://issi.dsic.upv.es/archives/f-1069167248521/actas.pdf#page=9>.

ROMERO, A. 2007. Gestión de riesgos con CMMI, RUP e ISO en ingeniería de software minero. [Online] 2007. [Cited: Abril 14, 2010.] http://www.scielo.org.pe/scielo.php?pid=S1561-08882007000100005&script=sci_arttext&lng=es.

Referencias bibliográficas

- LETELIER, P.** Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). [Online] [Cited: Abril 15, 2010.] <http://www.cyta.com.ar/ta0502/v5n2a1.htm>.
- RIESCO, D. 2004.** Un Workflow que Automatice los Procesos de Negocios del Proceso Unificado. [Online] 2004. [Cited: Abril 13, 2010.] <http://users.dsic.upv.es/~dromero/papers/wicc2004.pdf>.
- WEEKLY, C. 2007.** Hot skills: cut costs and coding time with Database for Objects.(Career Moves). [Online] 2007. [Cited: Abril 16, 2010.] //find.galegroup.com/itx/retrieve.do?contentSet=IAC- Documents&resultListType=RESULT_LIST&qrySerId=Locale%28es%2C%29%3AFQE%3D%28KE%2CNone%2C4%29DB4O%3AAnd%3ALQE%3D%28AC%2CNone%2C8%29fulltext%24&sgHitCountType=None&inPS=true.
- Anónimo4.** tics.org.ar. [Online] [Cited: Mayo 11, 2010.] http://www.tics.org.ar/index.php?option=com_content&view=article&id=170:modelo-vista-controlador&catid=26:area-desarrollo&Itemid=44.
- Jide. 2007.** Proiektutaldea-ren Weblog. [Online] 2007. [Cited: Abril 23, 2010.] <http://proiektutaldea.wordpress.com/2008/06/07/jcalendar/>.
- Figuerola, Carlos G. 2007.** reina.usal.es. [Online] Julio 2007. [Cited: Abril 14, 2010.] <http://reina.usal.es/papers/pazmadrid2007librerias.pdf>.
- ALCAIDE, P. J. 2008.** Convenciones de código Java. [Online] 2008. [Cited: Marzo 7, 2010.] <http://jpereza.nom.es/convenciones-de-codigo-java>.
- GRAU, XAVIER FERRÉ. 2009.** Desarrollo Orientado a Objetos con UML. [Online] 2009. [Cited: Abril 16, 2010.] <http://www.willydev.net/descargas/Articulos/General/umiTotal.pdf>
- Chevreau Karine, Choch José, Moya José A. García, Alonzo Margarita.** Generación Multilingüe de Boletines Meteorológicos. [Online] 2000. [Cited: Noviembre 12, 2009.] <http://www.sepln.org/revistaSEPLN/revista/25/25-Pag51.pdf>.
- Equipo de Soft-tonic. 2005.** Soft-tonic. [Online] 2005. [Cited: Enero 12, 2010.] <http://newsmerge.softtonic.com>.
- Santos., Elena. 2006.** Soft-tonic. [Online] 2006. [Cited: Enero 13, 2010.] <http://publish-it.softonic.com/>.

Bibliografía

Flanagan, David. 2002. Java in a nutshell: a desktop quick reference. 2002.

James Gosling, Bill Joy. 2000. The Java language specification. 2000.

Smart, John. 2008. Java Power Tools. 2008.

Stefan Edlich, Jim Paterson, Henrik Hörning, Reidar Hörning. 2006. The definitive guide to db4o. 2006.

Teodor Danciu, Lucian Chirita. 2007. The Definitive Guide to JasperReports. 2007.

Toffoli, Giulio. 2007. The Definitive Guide to IReport. 2007.

ANEXOS

Anexo 1. Descripción Caso de Uso Buscar noticia.

CU-2	Buscar noticia	
Actores:	Editor	
Propósito:	Buscar noticias en la Base de Datos para agregarlas en el boletín.	
<p><u>Resumen:</u> Este caso de uso es iniciado en dos ocasiones por el editor, el primer caso es cuando él toma la opción de diseñar un boletín, en el cual busca las noticias que desea, ya puede ser bajo el criterio de rango entre fechas, o coincidencia de una palabra en el título, en la descripción o en las palabras claves de las noticias. El resultado de la búsqueda son noticias que cumplen con los criterios de búsqueda, las cuales se suman a las lista de noticias del boletín que se va a diseñar. El segundo caso es cuando toma la opción de gestionar boletines, y realiza la misma búsqueda para agregar nuevas noticias a un boletín que encuentran en la Base de Datos.</p>		
Referencias:	R2	
Precondiciones:	Debe existir al menos una noticia en la Base de Datos que pertenezca al Editor logueado en el sistema en ese momento.	
<u>Acción del actor</u>		<u>Respuesta del sistema</u>
1. El Editor selecciona la opción Diseñar boletín. 3. El Editor despliega el filtro con el que desee buscar e inserta los criterios de búsqueda.		2. El Sistema un panel con dos filtros. 4. El Sistema realiza la búsqueda y la muestra en una lista.

Anexo 2. Descripción Caso de Uso Organizar Noticia.

CU-3	Organizar noticia	
Actores:	Editor	
Propósito:	Establecerle un orden a las noticias que se encuentran listas para ser agregadas al boletín que se desea diseñar.	
<u>Resumen:</u> El editor organiza las noticias que se encuentran en la lista de noticias, que van a ser agregadas al boletín que se desea diseñar. El editor puede establecer el orden por título, por autor, por fecha o manualmente.		
Referencias:	R3	
Precondiciones:	Debe existir alguna noticia para agregar al boletín.	
<u>Acción del actor</u>		<u>Respuesta del sistema</u>
1. El editor organiza las noticias que van a ser agregadas al boletín.		2. El Sistema va mostrando las noticias ordenadas.

Anexo 3. Descripción Caso de Uso Establecer campos.

CU-4	Establecer campos	
Actores:	Editor	
Propósito:	Introducir el nombre y autor el boletín que se va a diseñar, así como definir los campos de las noticias que se desea mostrar en el boletín.	

<u>Resumen:</u> El editor titula el boletín y se nombra creador del mismo, además selecciona los campos de noticias que va a contener el boletín.	
Referencias:	R4
Precondiciones:	
<u>Acción del actor</u>	<u>Respuesta del sistema</u>
1. El Editor titula el boletín y se nombra creador del mismo.	2. El Sistema muestra el nombre y el autor del boletín, además de los campos de las noticias escogidos por el Editor.

Anexo 4. Descripción Caso de Uso Editar y guardar noticias.

CU-5	Editar y guardar noticias
Actores:	Editor
Propósito:	Editar las noticias que se van incorporar al boletín y guardar los cambios en la Base de Datos.
<u>Resumen:</u> Este caso de uso es iniciado en dos ocasiones por el editor, el primer caso es cuando él toma la opción de diseñar un boletín, edita las noticias que se van a insertar en el boletín y guarda los cambios automáticamente, El segundo caso es cuando toma la opción de gestionar boletines, y edita las noticias de los boletines que le pertenecen, y los cambios se guardan antes de generar el boletín solo si acepta la pregunta se le hace el Sistema.	
Referencias:	R5
Precondiciones:	Debe existir al menos una noticia para editar.

<u>Acción del actor</u>	<u>Respuesta del sistema</u>
1. El Editor edita las noticias que se van a agregar al boletín.	2. En caso de que se esté diseñando un boletín nuevo, el Sistema guarda los cambios automáticamente. 3. En caso de que se esté gestionando un boletín de la Base de Datos, el Sistema le pregunta al Editor si desea guardar los cambios.

Anexo 5. Descripción Caso de Uso Escoger plantilla.

CU-6	Escoger plantilla
Actores:	Editor
Propósito:	Escoger una plantilla para aplicarle un diseño al boletín que se desea generar.
<u>Resumen:</u> El editor selecciona la plantilla que define el diseño que va a tener el boletín.	
Referencias:	R6
Precondiciones:	Debe existir al menos una plantilla en el directorio "Plantillas", donde el módulo busca las existentes.
<u>Acción del actor</u>	<u>Respuesta del sistema</u>
1. El Editor selecciona mostrar plantillas. 3. El Editor escoge la plantilla deseada.	2. El Sistema muestra las plantillas existentes. 4. El Sistema aplica el diseño

	escogido al boletín cuando es generado.
--	---

Anexo 6. Descripción Caso de Uso Exportar boletín.

CU-7	Exportar boletín
Actores:	Editor
Propósito:	Exportar el boletín a formatos como pdf, rtf, html, y doc.
Resumen: El editor exporta el boletín a distintos formatos.	
Referencias:	R7, R8
Precondiciones:	Se debe haber escogido antes una plantilla.
<u>Acción del actor</u>	<u>Respuesta del sistema</u>
1. El Editor manda a exportar el boletín. 3. El Editor escoge si desea o no guardar el boletín en la Base de Datos. 5. El editor escoge donde es que va a guardar el boletín en la PC;	2. El Sistema pregunta si se quiere guardar el boletín en la Base de Datos. (R8) 4. El Sistema muestra una vista previa de cómo va a quedar el boletín, con la posibilidad de escoger el formato y el lugar de la PC donde se quiere guardar. 4. El Sistema aplica el diseño escogido al boletín y lo exporta el formato deseado por el Editor.

Anexo 7. Descripción Caso de Uso Guardar boletín en Base de Datos.

CU-8	Guardar boletín en Base de Datos	
Actores:	Editor	
Propósito:	Guardar el boletín en la Base de Datos.	
<u>Resumen:</u> El Editor guarda el boletín en la Base de Datos.		
Referencias:	R8	
Precondiciones:		
	<u>Acción del actor</u>	<u>Respuesta del sistema</u>
	1. El Editor selecciona guardar el boletín en la Base de Datos.	2. El Sistema guarda el boletín en la Base de Datos.

Anexo 8. Descripción Caso de Uso Cargar boletín.

CU-9	Cargar boletín	
Actores:	Editor	
Propósito:	Cargar los boletines que se encuentran en la Base de Datos para ser editados, exportados o eliminados. (R10)	
<u>Resumen:</u> El editor carga el boletín que se halla guardado en la Base de Datos, donde puede editar sus noticias, guardar los cambios y exportarlo.		

Referencias:	R9, R10	
Precondiciones:		
<u>Acción del actor</u>	<u>Respuesta del sistema</u>	
1. El Editor selecciona cargar los boletines que se encuentran en la Base de Datos.	2. El Sistema muestra los boletines que se encuentran en la Base de Datos en dos grupos, los que le pertenecen al Editor logueado en ese momento y los de otros editores.	

Anexo 9. Descripción Caso de Uso Eliminar boletín.

CU-10	Eliminar boletín	
Actores:	Editor	
Propósito:	Eliminar los boletines.	
<u>Resumen:</u> El editor escoge el boletín o los boletines que desea eliminar de la Base de Datos y los elimina.		
Referencias:	R10	
Precondiciones:	Que el Editor logueado haya creado algún boletín.	
<u>Acción del actor</u>	<u>Respuesta del sistema</u>	
1. El Editor selecciona el boletín o boletines que desea eliminar y los elimina.	2. El Sistema muestra un mensaje que dice que el boletín o boletines fueron eliminados correctamente.	

Anexo 10. Descripción Caso de Uso Adicionar noticia.

CU-11	Adicionar noticia	
Actores:	Editor	
Propósito:	Adicionar nuevas noticias a los boletines que se encuentran en la Base de Datos.	
Resumen: El Editor adiciona nuevas noticias a uno de los boletines que cargo desde la Base de Datos.		
Referencias:	R11, R2	
Precondiciones:	Que exista al menos un boletín creado por el Editor logueado en ese instante, y que el Sistema encuentre al menos una noticia.	
<u>Acción del actor</u>		<u>Respuesta del sistema</u>
<ol style="list-style-type: none"> 1. El Editor busca las noticias. 3. El Editor escoge las noticias que va a incorporar al boletín escogido. 5. El editor escoge los campos y acepta. 		<ol style="list-style-type: none"> 2. El Sistema muestra todas las noticias encontradas. 4. El sistema muestra una ventana para que el Editor escoja los campos de las noticias que desee se muestren en el boletín. 6. Incorpora todo lo solicitado por el Editor en las noticias del boletín seleccionado.

Anexo 11. Descripción de la clase CC_ControladorVisual.

Nombre: CC_ControladorVisual	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
controlVisual	CC_ControladorVisual
búsquedaFecha	boolean
busquedaPalabras	boolean
fila1	int
fila2	int
título	boolean
autor	boolean
fecha	boolean
descripción	boolean
url	boolean
titulo_presentación	String
autor_presentación	String
enTítulo	boolean
enDescripción	boolean

enClaves	boolean
tablaOrdenada	boolean
reinicia	boolean
boletín	CE_Boletin
posición	int
editados	boolean
Para cada responsabilidad	
Nombre:	Singleton()
Descripción:	Devuelve en objeto de la clase controladora solo si este no se ha creado antes, lo que permite que solo se cree una sola instancia de la clase CC_ControladorVisual.
Nombre:	MostrarLasNoticiasSiguyentes(JTable tabla,List<CE_Noticia> lista)
Descripción:	Muestra lista, en tabla pasados por parámetro.
Nombre:	MostrarLasNoticias(JTable tabla,CE_Noticia noticia)
Descripción:	Muestra noticia, en tabla pasados por parámetro.
Nombre:	MostrarenLista(List<CE_Boletin> lista, JList lista1)
Descripción:	Muestra lista en lista1 pasados por parámetro.
Nombre:	OrdenarManualmente(final JTable tabla, final JPanel p)
Descripción:	Ordena la tabla manualmente, intercambiando noticia por noticia

Nombre:	EstablecerCursor(JPanel p)
Descripción:	Cambia el cursor.
Nombre:	Re_EstablecerCursor(JPanel p)
Descripción:	Devuelve el cursor a su forma por defecto.
Nombre:	OrdenarTabla(JTable tabla)
Descripción:	Ordena la tabla por título, autor y fecha.
Nombre:	ConfigurarTabla(JTable tabla, int col1, int col2, int col3)
Descripción:	Establece un tamaño de las columnas de tabla, determinado por los valores que se pasan por parámetro.
Nombre:	DespuesDeOrdenada(JTable tabla)
Descripción:	Guarda los datos de tabla después que esta se termina de ordenar.
Nombre:	MostrarBarra(boolean estado)
Descripción:	Muestra o no una JProgressBar según el estado que se pasa por parámetro.

Anexo 12. Descripción de la clase CC_HiloAcoplarNoticias.

Nombre: CC_HiloAcoplarNoticias	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
panel	JEditorPane

Para cada responsabilidad	
Nombre:	run()
Descripción:	Toma los atributos de cada elemento de la lista de noticia y los concatena guardándolo en un String.

Anexo 13. Descripción de la clase CC_HiloBoletines.

Nombre: CC_HiloBoletines	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
lista	JList
boletines	List<CE_Boletin>
Para cada responsabilidad	
Nombre:	run()
Descripción:	Busca todos los boletines existentes en la Base de Datos.

Anexo 14. Descripción de la clase CC_HiloGenerarDiseno.

Nombre: CC_HiloGenerarDiseno	
Tipo de Clase: Clase Controladora	
Atributo	Tipo

Para cada responsabilidad	
Nombre:	run()
Descripción:	Genera los boletines.

Anexo 15. Descripción de la clase CC_ HiloGenerarCarga.

Nombre: CC_HiloGenerarCarga	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
Para cada responsabilidad	
Nombre:	run()
Descripción:	Genera los boletines cargados.

Anexo 16. Descripción de la clase CC_ HiloNoticiasClaves.

Nombre: CC_HiloNoticiasClaves	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
tabla	JTable
Para cada responsabilidad	
Nombre:	run()

Descripción:	Busca en el fichero de la librería Lucene las noticias que contengan ciertas palabras claves, y las muestra en tabla.
---------------------	---

Anexo 17. Descripción de la clase CC_ HiloNoticiasDescripción.

Nombre: CC_HiloNoticiasDescripción	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
tabla	JTable
Para cada responsabilidad	
Nombre:	run()
Descripción:	Busca en el fichero de la librería Lucene las noticias que contengan ciertas palabras en sus descripciones, y las muestra en tabla.

Anexo 18. Descripción de la clase CC_ HiloNoticiasFecha.

Nombre: CC_HiloNoticiasFecha	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
tabla	JTable
Para cada responsabilidad	
Nombre:	run()

Descripción:	Busca en la Base de Datos las noticias que se hallan publicado entre una fecha y otra, escogidas por el editor, y las muestra en tabla.
---------------------	---

Anexo 19. Descripción de la clase CC_HiloNoticiasMarcadas.

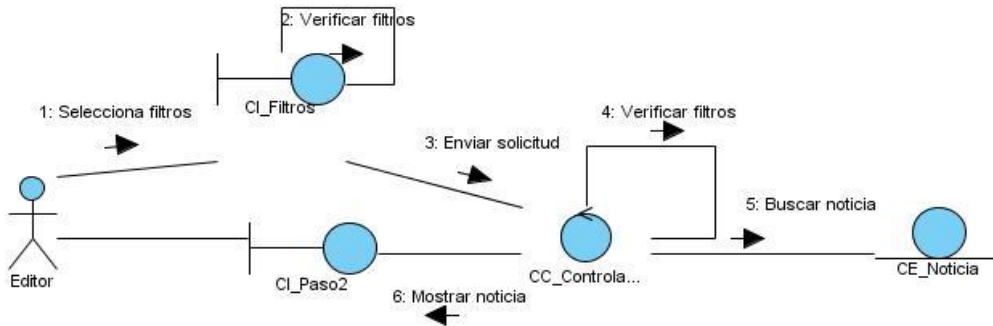
Nombre: CC_HiloNoticiasMarcadas	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
tabla	JTable
Para cada responsabilidad	
Nombre:	run()
Descripción:	Busca en la Base de Datos las noticias que se hallan marcadas, y las muestra en tabla.

Anexo 20. Descripción de la clase CC_HiloNoticiasTítulo.

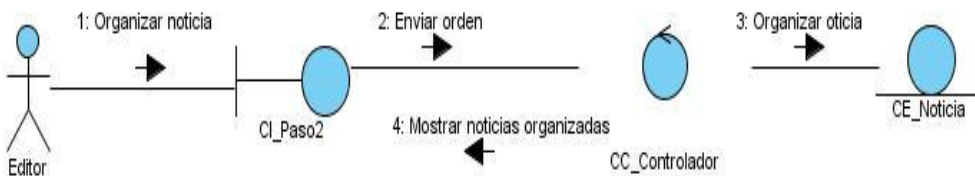
Nombre: CC_HiloNoticiasTítulo	
Tipo de Clase: Clase Controladora	
Atributo	Tipo
tabla	JTable
Para cada responsabilidad	
Nombre:	run()

Descripción:	Busca en el fichero de la librería Lucene las noticias que contengan ciertas palabras en sus títulos, y las muestra en tabla.
---------------------	---

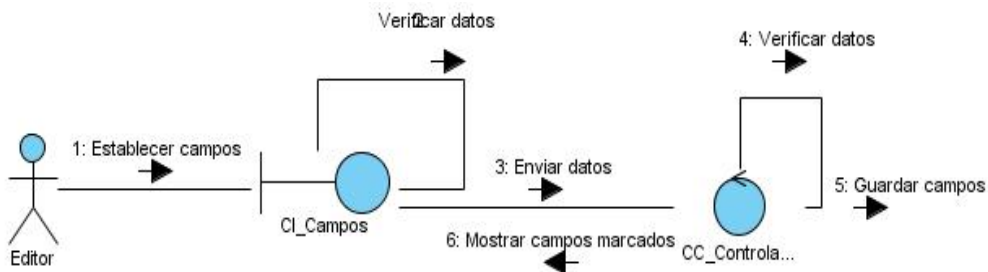
Anexo 21. Diagrama de Colaboración Buscar noticia.



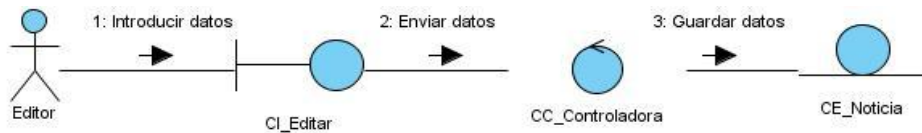
Anexo 22. Diagrama de Colaboración Organizar noticia.



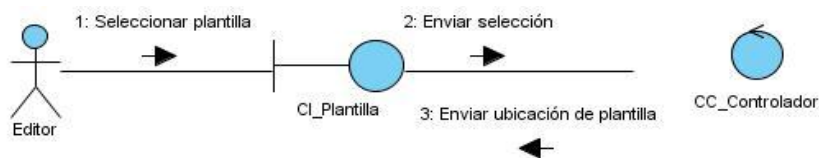
Anexo 23. Diagrama de Colaboración Establecer campos.



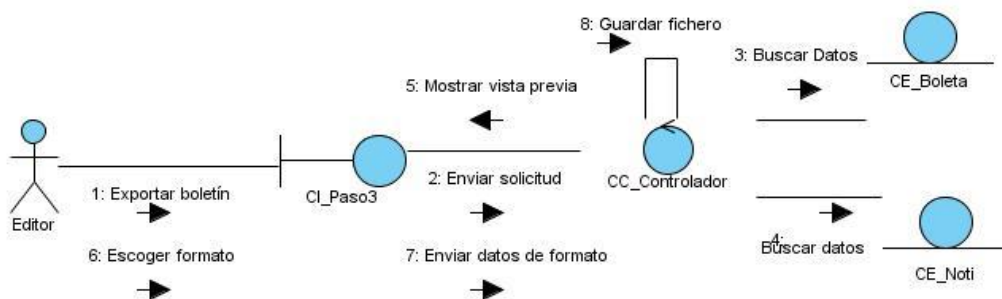
Anexo 24. Diagrama de Colaboración Editar y guardar noticias.



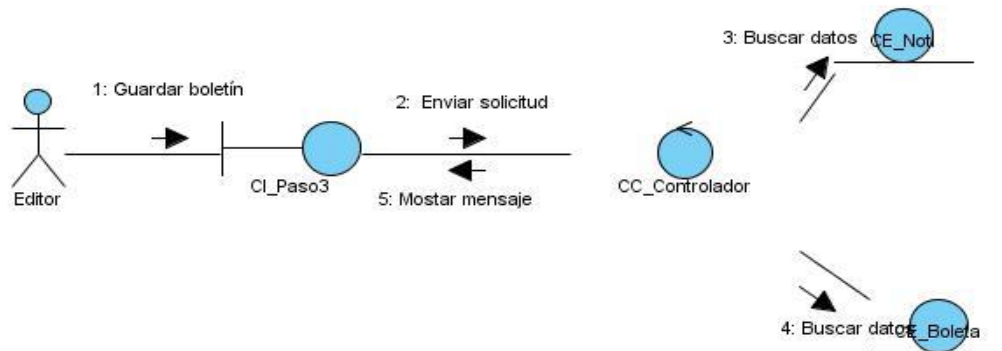
Anexo 25. Diagrama de Colaboración Escoger plantilla.



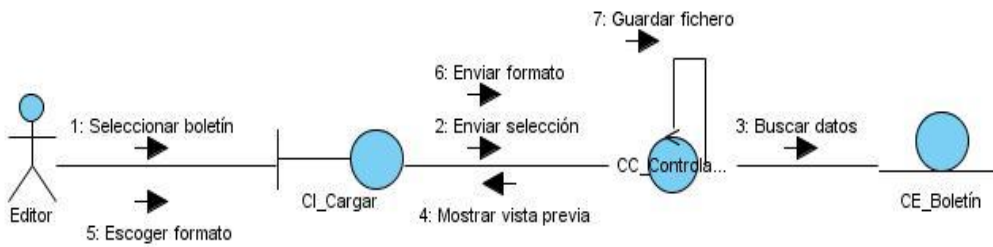
Anexo 26. Diagrama de Colaboración Exportar boletín.



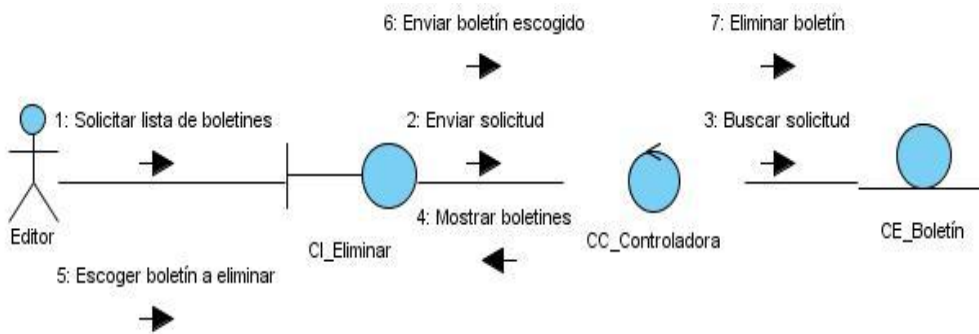
Anexo 27. Diagrama de Colaboración Guardar boletín en Base de Datos.



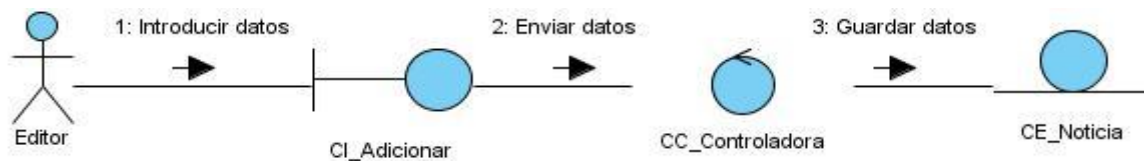
Anexo 28. Diagrama de Colaboración Cargar boletín.



Anexo 29. Diagrama de Colaboración Eliminar boletín.



Anexo 30. Diagrama de Colaboración Adicionar Noticia.



Glosario de Términos

Internet: conjunto de ordenadores, o servidores, conectados en una red de redes mundial, que comparten un mismo protocolo de comunicación, y que prestan servicio a los ordenadores que se conectan a esa red.

Bean: Un Bean es un componente software que tiene la particularidad de ser reutilizable y así evitar la tediosa tarea de programar los distintos componentes uno a uno.

Index: Proviene de la palabra en inglés index que dentro de una de sus acepciones significa ponerle un índice a algo según WordReference.com.

FTP: (sigla en inglés de File Transfer Protocol - Protocolo de Transferencia de Archivos) en informática, es un protocolo de red para la transferencia de archivos entre sistemas conectados a una red TCP, basado en la arquitectura cliente-servidor.

Herramientas: Funciones que ofrece un programa a través de una barra con íconos, que representan los distintos recursos del software para realizar una tarea determinada.

HTML: Lenguaje utilizado para armar documentos de hipertexto, como las páginas que se visualizan en la World Wide Web.

Interfaz: Conexión e interacción entre elementos de hardware, programas y usuarios, es decir como la plataforma o medio de comunicación entre usuario o programa.. Las interfaces de usuario son todos los dispositivos de ingreso de datos (teclado, mouse) y los recursos gráficos de los sistemas operativos (cajas de diálogo, menús descolgables, íconos, etc.) que hacen posible la interacción entre la PC y el operador.

JAVA: es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems a principios de los años 90. Inicialmente Java fue creado por un grupo de investigadores de Sun Microsystems como un lenguaje de programación multipropósito y que funcionara no sólo en computadoras, sino en cualquier dispositivo electrónico que necesite de programación.

Lenguaje de Programación: Código utilizado para la creación de programas. Lo utilizan los programadores (generalmente especializados en un lenguaje en particular) para la creación de diversas aplicaciones.

PC: Computadora personal dotada de memoria y procesamiento, capaz de ejecutar aplicaciones comerciales, procesamiento de textos, generación de gráficos, programas de entretenimiento, multimedia, etc. En la actualidad, la gran mayoría de las computadoras de hogares y oficinas son PC.

Glosario de Términos

Requerimientos: es una necesidad documentada sobre el contenido, forma o funcionalidad de un producto o servicio. Se usa en un sentido formal en la ingeniería de sistemas o la ingeniería de software. Otro (Comprender todas las tareas relacionadas con la determinación de las necesidades o de las condiciones a satisfacer).

RunTime: Se denomina Tiempo de ejecución (Runtime en inglés) al intervalo de tiempo en el que un programa de computadora se ejecuta en un sistema operativo.

Sindicación: Sistema que recoge los titulares de las noticias o historias publicadas en los weblog, casi todos los weblogs incorporan enlaces de texto, o bien, los famosos botones naranjas con las abreviaturas XML, RSS, RDF, o ATOM para que el lector copie la dirección web que traen conocida como feed y la inserten en un lector/agregador de noticias o feeds que avisará de las actualizaciones de los weblogs favoritos evitando acudir a la página web de forma que no se tenga que consultar las novedades que hay en cada página individual, al centralizar las novedades de las páginas seleccionadas en un único lugar de consulta.

Sitio web: Colección de páginas web dotada de una dirección web única.

Sistema: Grupo de componentes relacionados que interactúan para realizar una tarea determinada. En el campo computacional, un sistema informático se compone de una CPU, el sistema operativo y los dispositivos periféricos.

Software: Conjunto de programas, documentos, procesamientos y rutinas asociadas con la operación de un sistema de computadoras, es decir, la parte intangible o lógica de una computadora.

UML: Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.

Usuario: Persona que utiliza la computadora para sistematizar tareas, guardar información, conectarse con otras personas para compartir información, etc.

Multithread: Un lenguaje que soporta múltiples threads (hilos en español) es un lenguaje que puede ejecutar varios procedimientos en paralelo.