Universidad de las Ciencias Informáticas

Facultad 1

Subsistema de alertas para el buscador Orión

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas.

Autora:

Raisa Díaz Aleman

Tutores:

Ing. José Gabriel Espinosa Ramírez

Ing. Leiny Amel Pons Flores

La Habana, julio de 2016

"El conocimiento nos hace responsables"

Ernesto Guevara de la Serna

Declaración de Autoría

Yo, Raisa Díaz Aleman, declaro que soy la única autora del trabajo de diploma titulado: “Subsistema de alertas para el buscador Orión”, y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los __ días del mes de _____ del año 2016.

_____

Firma del autor

Raisa Díaz Aleman

Los abajo firmantes certificamos que el presente trabajo ha sido revisado según acuerdo de la dirección del Centro de Ideoinformática (CIDI) y el mismo cumple los requisitos que debe tener un trabajo de esta envergadura.

_____	_____
Firma del tutor	Firma del tutor
Ing. José Gabriel Espinosa Ramírez	Ing. Leiny Amel Pons Flores

Dedicatoria

Dedico mi trabajo de diploma a mis padres María Luisa y Domingo porque ellos han dado razón a mi vida. Por brindarme sus consejos, su apoyo incondicional y su paciencia, todo lo que hoy soy es gracias a ustedes. Me siento muy orgullosa de que sean mis padres y espero que ustedes también estén muy orgullosos de mi.

También dedico este logro a mi sobrina Sandra María y la que aún viene en camino Sofía, espero que algún día puedan superarme y ser mejores estudiantes que yo.

Agradecimientos

Quiero agradecerle a mis padres por todo lo que han luchado porque yo llegara hoy aquí y sobre todo por siempre confiar en mi, a mi hermano Raynel por ser el mejor hermano del mundo y siempre estar ahí cuando lo necesité, a mis abuelas, tíos y primos que aunque no están aquí hoy sé que están muy preocupados por mi. A mi novio Roberto quiero agradecerle por darme la oportunidad de demostrarle que sí podía ser una mejor persona, por haberme tenido mucha paciencia estos meses cuando el estrés de la tesis me daba por cogerla con él. A mi cuñada Tania que a pesar de tener la misma edad que yo siempre me han servido de mucho todos sus consejos.

A mis tutores José Gabriel y Leyni, siempre voy a estar eternamente agradecida porque sin ustedes nada de esto hubiera sido posible, gracias por sacarme del fondo cuando otras personas luchaban por hundirme.

A Lisbet y a Betsy por haber sido mis confidentes, mis mejores amigas, por siempre poder contar con ustedes y espero que se miren en mi espejo para que el día que estén en mi lugar no lo pases tan mal como yo.

A mi mejor amigo Alejandro, tú fuiste mi mayor apoyo cuando solo tenía ganas de rendirme, tú me diste el ánimo para seguir adelante. Siempre me dijistes vamos a salir de esto y aquí estamos. Quiero que sepas que eres muy importante para mi.

A Wilbia y Aimet por estar siempre al pendiente de mi durante estos 8 años que llevamos juntas. A Laura y a Baby que a pesar de ser mis amigas de menos tiempo viví momentos maravillosos junto a ustedes.

A mis compañeros de aula por haberme soportado durante todo este tiempo qué sé que no fue para nada fácil. A René, Darío, Oberman, Yojahny, Luis Ernesto, Amaury, Rubén, Danilo y Arián. Gracias por los buenos momentos y siempre poder contar con ustedes.

A José Carlos, Ramón, Andy y Jorge Javier por darme mucha confianza y estar ahí en los buenos y malos momentos.

Resumen

El presente trabajo consiste en el desarrollo de un subsistema de alertas para el buscador Orión desarrollado en el centro de Ideinformática CIDI de la Universidad de las Ciencias Informáticas (UCI). Para ello, se realizó un estudio de sistemas de alertas para buscadores. Teniendo en cuenta el análisis realizado, se obtuvieron características que se tuvieron en cuenta para su desarrollo. Para la implementación de la propuesta de solución, guiado por la metodología AUP, se seleccionaron como principales tecnologías: el marco de trabajo Symfony2 para la programación en PHP, RabbitMQ 3.6 para el trabajo con mensajes y NetBeans 7.2 como entorno de desarrollo integrado. Para las pruebas de seguridad y rendimiento se utilizaron las herramientas Acunetix 8.0 y Apache JMeter 2.9 respectivamente. El subsistema de alertas para el buscador Orión permite a los usuarios realizar solicitudes de alertas y ser notificados automáticamente al correo electrónico sobre nuevos contenidos que coincidan con los criterios de búsqueda introducidos por los usuarios.

Palabras clave: alertas**,** buscador, correo electrónico, notificados, Orión.

Contenido

Introducción

Con los avances tecnológicos de la informática, la demanda de información tiende a crecer notablemente en Internet[1] (Colimba, 2013). En muchos casos, esta se encuentra dispersa y poco estructurada, lo que hace muy engorroso el proceso de recuperación de información útil en la red. Los Sistemas de Recuperación de Información (SRI), constituyen una alternativa para resolver este tipo de problemas (Olvera, 2015). Los SRI son sistemas capaces de localizar cualquier contenido existente en la Web. En este sentido, se destacan los directorios temáticos, los motores de búsqueda o buscadores y los metabuscadores (Pinto, 2015).

Los motores de búsqueda, también conocidos como buscadores de Internet, son programas que permiten localizar coincidencias entre la información que existe en la red y la que demanda un usuario. Funcionan rastreando la red de forma periódica y como resultado de la búsqueda se obtiene aquellos recursos web que se ajustan a los criterios de búsqueda introducidos por un usuario (Pastor, 2016). La relevancia de los motores de búsqueda radica en la imposibilidad de catalogar con presición y en su totalidad la inmensidad de los contenidos publicados en la red, de forma manual para su posterior recuperación bajo demanda. La automatización de los procesos de rastreo, análisis, indexado y recuperación de contenidos en la Web mediante el desarrollo de motores de búsqueda se ha constituido en un pilar para la evolución de Internet (Yofu, 2015), al reducir significativamente el tiempo para acceder o descubrir un recurso en la red. Además los motores de búsqueda son plataformas sobre las cuales desplegar servicios de valor agregado que solucionen las necesidades de información del usuario (Pastor, 2016).

Entre los motores de búsqueda más destacados en Internet se encuentran Google, con un 62.74% del mercado en el año 2015, Bing y Yahoo con 8.70% y 7.79% respectivamente y Baidú con un 18.68% (Suárez, 2016). Los mismos permiten realizar búsquedas en catálogos, de imágenes, videos, música, noticias y libros (Pinto, 2015). En el ámbito nacional también se han llevado a cabo la implementación de algunos buscadores como Lupa, 2x3 y la plataforma de contenidos unificados para la búsqueda avanzada C.U.B.A, algunos de ellos encontrándose en deshuso actualmente (Díaz,

[1] Internet: Conjunto descentralizado de redes de comunicación que se conectan entre si y utilizan la familia de protocolos TCP/IP, garantizando que las redes físicas heterogéneas que la componen funcionen como una red lógica única, de alcance mundial.

2015). La Universidad de las Ciencias Informáticas (UCI) y específicamente el centro de Ideoinformática (CIDI) cuenta con un equipo de desarrollo que se encarga de poner en práctica y brindar soporte al buscador Orión, el mismo cuenta con facilidades para realizar búsquedas sobre páginas web, imágenes y documentos.

Es común que los usuarios de los motores de búsqueda, tengan determinadas necesidades de información que deben ser satisfechas de forma períodica, mediante la ejecución de búsquedas en la Web sobre un tópico específico. Para estos usuarios contar con un servicio que realice estas búsquedas de forma automática y les notifique los resultados a intervalos de tiempo predefinidos es una característica altamente valorada.

Como parte de los servicios ofrecidos por las plataformas de los motores de búsqueda de Google, Bing, Yahoo! y Baidú se cuenta, entre otras facilidades, con un sistema de alertas que se encarga de avisar al usuario a través del correo electrónico acerca de los resultados obtenidos para los criterios de búsquedas de su interés (Mena, 2016). De esta manera automáticamente se notifica al usuario, períodicamente, cuando el nuevo contenido de las noticias, web, bitácoras (blogs), videos y grupos de discusión coincide con un conjunto de términos de búsqueda seleccionados por el usuario (Mena, 2016). La emisión de alertas períodicas sobre los contenidos encontrados, que satisfacen las necesidades de información de los usuarios, es una característica que contribuye a una mejor experiencia del usuario final y aporta un valor agregado al motor de búsqueda como una herramienta de recuperación de información.

Dada la dinámica de Internet en la actualidad que se caracteriza por la constante generación de nuevos contenidos en diversas áreas de interés, los usuarios de la red fijan líneas temáticas sobre las cuales necesitan mantenerse actualizados (Pinto, 2015). Los usuarios del motor de búsqueda Orión que tengan necesidades de información, que deban ser atendidas de forma regular en el tiempo, solo pueden solventarlas mediante la realización de las búsquedas necesarias directamente del buscador, en cada momento. La reiteración en la ejecución de estas búsquedas puede constituir un gasto del fondo de tiempo, de los recursos de los que disponga el usuario e inducir errores en los criterios de búsquedas introducidos, lo cual puede distorsionar los resultados obtenidos. Los elementos anteriormente expuestos atentan contra la fidelidad y el interés del usuario por el uso del motor de búsqueda Orión.

Tomando en cuenta la problemática planteada anteriormente surge como **problema a resolver**: ¿Cómo mantener informado a los usuarios a través del correo electrónico sobre nuevos contenidos en la Web que sean de su interés?

Por tanto, el presente problema se enmarca en el **objeto de estudio**: sistema de alertas para buscadores, delimitado por el **campo de acción**: subsistema de alertas para el buscador Orión.

Para dar solución al problema existente se define como **objetivo general**: Desarrollar un subsistema de alertas que notifique al usuario a través del correo electrónico sobre nuevos contenidos en la Web que sean de su interés.

Planteando como **objetivos específicos**:

- ✓ Describir el estado del arte de los sistemas de alertas para buscadores y de las tecnologías utilizadas o candidatas a utilizar para el desarrollo del subsistema de alertas para el buscador Orión.
- ✓ Describir los requisitos de software, garantizando así un hilo conductor en el desarrollo del subsistema de alertas para el buscador Orión.
- ✓ Desarrollar el subsistema de alertas integrado con el buscador Orión.
- ✓ Evaluar el subsistema de alertas para el buscador Orión desarrollado a través de pruebas de software acordes a los requisitos de la misma.

Como **idea a defender** se plantea que: el desarrollo de un subsistema de alertas que se integre al buscador Orión posibilitará a los usuarios recibir una notificación cuando el motor de búsqueda detecte nuevos contenidos en la Web que sean de su interés.

Para la solución que se propone es necesario el uso de **métodos científicos**. A continuación, se detallan los métodos teóricos y empíricos utilizados:

Métodos teóricos:

Histórico-lógico: Este método permite realizar una investigación que da inicio con los orígenes de los sistemas de alertas para buscadores y las tendencias que actualmente existen.

Analítico-Sintético: Este método permite el análisis de documentos y teorías acerca de los sistemas de alertas para buscadores, permitiendo analizar los lenguajes, tecnologías y herramientas a utilizar en el desarrollo de este subsistema.

Métodos empíricos:

Modelación: Utilizado en la representación, mediante el uso de diagramas, de las características del subsistema, relaciones entre objetos y componentes del subsistema propuesto.

Observación: Se hace un estudio minucioso de trabajos realizados que se encuentran estrechamente vinculados con la problemática a la que se hace referencia.

Entrevista: Entrevista informal a profesores y especialistas en la rama que poseen conocimientos acerca del buscador Orión, especialmente a aquellos que se encuentran actualmente trabajando en ello.

Como **posible resultado** se obtendrá un subsistema de alertas que mediante la utilización del buscador Orión notifica al usuario a través del correo electrónico sobre nuevos contenidos en la Web que sean de su interés.

El documento está estructurado en tres capítulos:

Capítulo 1. Fundamentación teórica del subsistema de alertas para el buscador Orión.

En este capítulo se presentan los elementos teóricos que sirven de base a la investigación del problema planteado, realizando un estudio del estado del arte de sistemas similares al que se desea implementar. Se realiza un análisis de los lenguajes de programación, herramientas, tecnologías y metodología de desarrollo de software a usar en el desarrollo de este subsistema de alertas.

Capítulo 2. Características y diseño del subsistema de alertas para el buscador Orión.

En este capítulo se realiza el análisis y diseño del subsistema de alertas para el buscador Orión. Para

ello, se hace uso de los artefactos que propone la metodología AUP, así como la representación de los principales procesos mediante casos de uso y diagramas de clases del diseño. Se describen además los requisitos funcionales y no funcionales, la arquitectura y los patrones de diseño a usar en la realización del subsistema.

Capítulo 3. Implementación y pruebas realizadas al subsistema de alertas para el buscador Orión.

En este capítulo se describe la etapa de implementación. Se elaboran y documentan las pruebas realizadas a la solución propuesta para demostrar su correcto funcionamiento. Se realiza un análisis de los resultados del subsistema en un entorno real, con el objetivo de asegurar la calidad y eficiencia del software.

Capítulo 1. Fundamentación teórica del subsistema de alertas para el buscador Orión

1.1 Introducción

En el presente capítulo se presentan los elementos teóricos que sirven de base a la investigación del problema planteado. Se realiza un estudio sobre sistemas similares, permitiendo analizar las características de los mismos, así como las tendencias de estos en la actualidad. Se describen además las principales metodologías de desarrollo de software, lenguajes de programación, herramientas y tecnologías, definiéndose los que serán utilizados para todo el proceso de modelado e implementación del subsistema.

1.2 Sistemas similares existentes.

Existen en la actualidad herramientas de búsqueda que permiten obtener una notificación a través del correo electrónico de nuevas noticias que coinciden con el criterio de búsqueda introducido por el usuario (Pastor, 2016).

1.2.1 Buscadores internacionales.

A continuación, se darán a conocer algunos detalles correspondientes a buscadores internacionales que permiten el servicio descrito anteriormente.

Yahoo!.

Motor de búsqueda desarrollado por la compañía Yahoo! que hizo su aparición pública en el mercado de valores de Nueva York el 12 de abril de 1996 (Requena, 2014).

Entre los servicios que hoy Yahoo! ofrece se encuentran: Yahoo! Correo, Yahoo! Grupos, Yahoo! Juegos, Yahoo! Compras, Yahoo! Subastas, además ofrece Yahoo! Alertas (Esteffan, 2015), un servicio de alertas de noticias por correo electrónico que exige el registro del usuario.

Este sistema de alertas cuenta con tres opciones:

- ✓ Alerta de palabras clave (*keyword alert*): envía a una dirección de correo electrónico vínculos con las informaciones que tengan relación con la palabra o palabras clave que el usuario ha introducido previamente.
- ✓ Alerta de Noticias de última hora (*breaking news alert*): remite al correo electrónico las noticias de última hora.
- ✓ Boletines (*news bulletins*): facilita también por correo electrónico, boletines de noticias relacionados con un área temática concreta, como tecnología, salud y negocios.

Es importante señalar que las notificaciones que ofrece este motor de búsqueda solo pueden ser enviadas a una cuenta de correo de Yahoo!.

Google.

Motor de búsqueda desarrollado por la compañía Google, estrenado en Internet el 27 de septiembre 1998. El motor de búsqueda Google integra servicios como el correo electrónico llamado *Gmail,* sus servicios de mapas *Google Maps* y *Google Earth*, el sitio web de videos *YouTube* y otras utilidades web como el sistema de alertas *Google Alerts* (Martínez, 2014).

Este sistema de alertas permite recibir notificaciones que pueden ser enviadas por correo electrónico o como un hipervínculo a usuarios que no necesariamente tienen una cuenta en *Gmail* (Mena, 2016). Se debe destacar que alertas de Google solo ofrece contenidos del propio buscador Google.

Las alertas se envían sólo si el nuevo contenido coincide con los términos de búsqueda seleccionados por el usuario. Las notificaciones que ofrece este motor de búsqueda provienen solo de sitios de noticias, lo que significa que se ignoran páginas personales, bitácoras (*blogs*), entre otros sitios (Mena, 2016).

1.2.2 Buscadores nacionales.

Buscador 2x3

El primer buscador cubano en Internet para los sitios con dominio .cu fue presentado el 15 de febrero de 2007 por Roberto del Puerto, director de la Oficina para la Informatización, en la XII Convención y

Expo Internacional Informática 2007.

La característica esencial de 2x3, es que su acción de búsqueda se desarrolla en el dominio .cu. Con ello, se facilita considerablemente el hallazgo de información generada en el país, cuya exploración hasta el momento dependía de los buscadores particulares de cada sitio. Además, cuenta con varios servicios de búsqueda especializada como el de imágenes, prensa, discursos de Fidel, archivo (Word y PDF), multimedia y más (tiempo, noticias, diccionarios) (Díaz, 2015).

C.U.B.A

Plataforma de contenidos unificados para búsqueda avanzada que opera con el motor de búsqueda Orión, desarrollado en la UCI y administrada por los Joven Club de Computación y Electrónica.

El creciente número de sitios web bajo el dominio .cu, ha generado una serie de productos y aplicaciones nacionales en línea que se encuentran dispersas. Así, la intención de la plataforma C.U.B.A es unificarlas bajo un solo portal para facilitar a los usuarios el acceso a sus diversas prestaciones. El sitio es, en síntesis, un buscador que permite recuperar información publicada solo en el dominio .cu.

Esta plataforma, cuenta con un diseño que se adapta a los diferentes terminales (ordenadores, tabletas, teléfonos móviles). Utiliza, además, tecnologías libres lo cual garantiza soberanía tecnológica a la nación. En esta primera etapa de su lanzamiento, la plataforma enlaza a otros servicios, como la cartelera cultural La Papeleta, la plataforma de bitácoras (*blogs*) Reflejos, la enciclopedia Ecured, el sitio noticioso Cubadebate y el Andariego, así como una aplicación de mapas interactivos desarrollado por la Empresa de Cartografía y Soluciones Geomáticas (Díaz, 2015).

Resultados del estudio de los sistemas similares existentes.

Luego de realizar un estudio a los motores de búsqueda, tanto en el ámbito nacional como internacional, se llegó a las siguientes conclusiones:

1. Los buscadores internacionales presentan características que se tuvieron en cuenta para la implementación del subsistema, por ejemplo: el estilo y formato de los formularios, así como las categorías de búsqueda que puede introducir el usuario para realizar su solicitud de alerta.

2. En el caso de los buscadores nacionales estudiados no se puede tomar ninguna característica, ya que no poseen un mecanismo de alertas que permita notificar al usuario a través del correo electrónico sobre nuevos contenidos encontrados en la Web.

3. Desarrollar un subsistema de alertas para el buscador Orión que responda a las necesidades del centro CIDI.

1.3 Metodología de desarrollo de software.

Una metodología es un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software (Gómez, 2016). Actualmente existen varias metodologías, cada una de ellas con características particulares que las hacen diferenciarse, son clasificadas en dos grupos: ágiles y tradicionales. En este epígrafe se estudian diferentes metodologías con el objetivo de seleccionar la que guiará el proceso de desarrollo del subsistema de alertas para el buscador Orión.

Las metodologías tradicionales imponen una disciplina de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente (Uñoja, 2015). Entre las metodologías tradicionales o pesadas se encuentran:

RUP (*Rational Unified Procces*).

RUP es un proceso que define claramente quien, cómo, cuándo y qué debe hacerse; este aporta artefactos como los casos de uso, que describen los requerimientos, además de permitir la ejecución iterativa del proyecto y del control de riesgos (Eucaris, 2015).

Las características principales del proceso son:

- ✓ Guiado por los casos de uso: Representan los requerimientos base para el desarrollo del sistema, constituyen el punto de partida para las tareas de análisis y diseño y son la fuente para que el equipo de pruebas construya los casos de pruebas.
- ✓ Centrado en arquitectura: Esta característica en un proceso de desarrollo permite que el sistema se cree a medida que se obtienen o que se desarrollan y maduran sus componentes.

- ✓ Iterativo e Incremental: RUP divide el proceso en cuatro fases. Resulta muy práctico dividir el trabajo en piezas o miniproyectos.

Las metodologías ágiles defienden la idea de que el proceso de desarrollo del software, se centra en el software y no en la documentación alrededor de este (Uñoja, 2015). Entre las metodologías ágiles se encuentran:

XP (*Extreme Programming*).

La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto (Villamil, 2015).

Las características principales del proceso son:

- ✓ Desarrollo iterativo e incremental: pequeñas mejoras.
- ✓ Pruebas unitarias: se basa en las pruebas realizadas a los principales procesos, de tal manera que, se puedan realizar pruebas de las fallas que pudieran ocurrir.
- ✓ Refactorización del código: reescribir ciertas partes del código para aumentar su legibilidad y mantenibilidad, pero sin modificar su comportamiento.
- ✓ Programación en pares: la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo.
- ✓ Propiedad del código compartida: promueve que todo el personal pueda corregir y extender cualquier parte del proyecto.
- ✓ Simplicidad en el código: se podrán añadir funcionalidades si es necesario.

Las etapas que se recomienda seguir en esta metodología son: planeación, diseño, codificación y prueba (Villamil, 2015).

AUP *(Agile Unified Process).*

El Proceso Unificado Ágil es una versión simplificada de RUP. Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles. AUP se preocupa especialmente de la gestión de riesgos (Aquino, 2015).

El proceso AUP establece un modelo más simple que el que aparece en RUP por lo que reúne en una única disciplina: modelado de negocio, requisitos y análisis y diseño. El resto de las disciplinas (implementación, pruebas, despliegue, gestión de configuración, gestión y entorno) coinciden con las restantes de RUP (Aquino, 2015).

Las características principales del proceso son:

- ✓ Versión simplificada de la metodología RUP.
- ✓ La atención se centra en las actividades que se ve que son esenciales para el de desarrollo, no todas las actividades que suceden forman parte del proyecto.
- ✓ Es de fácil acomodo a través de cualquier herramienta de edición de HTML.

Al igual que en RUP, en AUP se establecen cuatro fases que transcurren de manera consecutiva: concepción, elaboración, construcción, transición (Morán, 2016).

Fundamentación de la metodología de desarrollo de software utilizada:

Para el desarrollo de la propuesta de solución se decide utilizar la metodología AUP, la cual ofrece un amplio marco de buenas prácticas en la fase de construcción de software en búsqueda de la optimización, promoviendo medidas como la ejecución de pruebas en paralelo con la programación. Además, cuenta con actividades de carácter iterativo e incremental favoreciendo al logro de un producto software en menor tiempo y bajo una comunicación horizontal en el tratamiento de cambios, característica esencial ya que el equipo de desarrolladores está reunido directamente con el cliente para conocer sus necesidades. Por otra parte, todo se describe concisamente utilizando poca documentación, la atención se centra en las actividades que en realidad lo requieren, no en todo el proyecto, permitiendo así adaptar el producto que se obtiene de manera que satisfaga las necesidades del cliente. Se tuvo en cuenta que esta metodología es la que guía el proceso de desarrollo de software en el centro CIDI.

1.4 Lenguaje de modelado.

El lenguaje de modelado de objetos es un conjunto estandarizado de símbolos para modelar un diseño de software orientado a objetos y es esencial en la construcción de un software para comunicar la estructura de un sistema complejo y comprender mejor lo que se está construyendo (Silva, 2015).

Para el desarrollo de este subsistema se decidió usar el Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) respaldado por el OMG (*Object Management Group*), por su posibilidad de visualizar, especificar, construir y documentar un sistema. UML tiene como característica que es independiente del lenguaje de implementación, de tal forma que los diseños realizados usando UML se pueden implementar en cualquier lenguaje que soporte las posibilidades de UML , principalmente lenguajes orientados a objetos.

Este lenguaje estandariza 9 tipos de diagramas para representar gráficamente un sistema desde distintos puntos de vista, estos son: diagramas de clases, objetos, casos de uso, secuencia, colaboración, estados, actividades, componentes y desarrollo (Silva, 2015).

1.5 Herramienta para el modelado.

Las herramientas CASE (Ingeniería de Software Asistida por Computadoras, por sus siglas en inglés *Computer Aided Software Engineering*) son herramientas informáticas que intentan proporcionar ayuda automatizada a las actividades del proceso de desarrollo de software. Entre ellas: diseño de proyectos, implementación de parte del código automáticamente y detección de errores (Campechano, 2015). A continuación, se realiza un estudio a diferentes herramientas CASE, seleccionando la más apropiada para el modelado del subsistema de alertas para el buscador Orión.

Visual Paradigm.

Visual Paradigm es una herramienta CASE del tipo de sofware *freeware*, basado en el lenguaje UML. Visual Paradigm ha sido concebida para soportar el ciclo de vida completo del proceso de desarrollo del software a través de la representación de todo tipo de diagramas. Ofrece un diseño centrado en casos de uso y enfocado al negocio, de manera que genera un software de mayor calidad. Presenta capacidades de ingeniería directa e inversa. Está disponible en múltiples plataformas como Windows

o Linux. Es capaz de generar la documentación del proyecto en diferentes formatos como web o pdf (Pacheco, 2015).

Rational Rose.

Es una herramienta CASE del tipo de sofware propietario, basada en el lenguaje UML. Permite crear los diagramas que se van generando durante el proceso de ingeniería en el desarrollo del software, excepto los diagramas de implementación. Rational Rose necesita de mucha memoria para poder ser manejado de forma rápida y eficiente, está disponible solo para Windows y su entorno gráfico no es muy amigable (Pacheco, 2015).

Enterprise Architect

Es una herramienta CASE del tipo de sofware propietario para modelado UML. Cuenta con un buen soporte para modelar negocios, software y sistemas. Puede ser empleada en múltiples plataformas (Pacheco, 2015). Esta herramienta no tiene una interfaz con manejo intuitivo y los diagramas que permite realizar no se encuentran en la vista.

Fundamentación de la herramienta CASE utilizada:

El análisis realizado a las diferentes herramientas CASE arrojó como resultado que será Visual Paradigm en su versión 8.0 la herramienta a utilizar para el modelado del subsistema debido a que su licencia es gratuita para el uso de la comunidad de desarrolladores.

Su estabilidad de ejecución en diferentes sistemas operativos. Esta característica es muy importante pues por ejemplo el Rational Rose, es una herramienta muy recomendada, pero tiene como desventaja que obliga al usuario a desarrollar en máquinas con el sistema operativo Windows. Se tuvo en cuenta además, que permite la generación de bases de datos, permitiendo la transformación de diagramas de entidad-relación en tablas de base de datos.

1.6 Lenguaje de programación.

Un lenguaje de programación es un lenguaje formal diseñado para realizar procesos que pueden ser llevados a cabo por máquinas como las computadoras. Está formado por un conjunto de símbolos y

reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones (Pérez, 2016).

Para el desarrollo de la aplicación que dará solución al problema inicialmente planteado en la presente investigación, es necesario tener lenguajes de programación del lado del cliente.

Dichos lenguajes son interpretados por el propio navegador, mostrando al usuario la información proveniente del servidor con una estructura y jerarquía, facilitando una mayor comprensión de dicha información (Gómez, 2015).

1.6.1 Lenguajes de programación del lado del cliente.

AJAX (*Asynchronous JavaScript And XML*).

Se trata de un conjunto de tecnologías, entre las que se encuentran XHTML[2], DOM[3], XML[4], JSON[5]; con las que se logra el desarrollo de aplicaciones web dinámicas e interactivas, evitando el envío de peticiones continuas al servidor y la recarga constante de la página (Arenas, 2015). Esta tecnología pierde la capacidad para hacer cosas que se hacía con webs tradicionales como usar los botones de avance y retroceso del navegador o añadir una página a favoritos. La indexación para los motores de búsqueda también se ve dificultada, con lo cual los sitios web pueden perder visibilidad en los buscadores (Rodríguez, 2016).

HTML *(HyperText Markup Language).*

Es un estándar para la elaboración de páginas web, utilizado para crear documentos que muestren una estructura de hipertexto. HTML permite, además, crear documentos multimedia, es decir, que contengan información más allá de lo simplemente textual, como por ejemplo: imágenes, videos y

[2] XHTML: Lenguaje de Marcado de Hipertexto Extensible de su significado en inglés *eXtensible HyperText Markup Language*.

[3] DOM: Acrónimo de *Document Object Model*, es un estándar definido por la W3C, que modela la estructura de los documentos HTML como objetos y que permite ser modificada utilizando JavaScript u otro lenguaje de programación.

[4] XML: *Extensible Markup Language*- Lenguaje de etiquetado extensible.

[5] JSON: JavaScript *Object Notation* - Notación de Objetos de JavaScript.

sonidos. Es un lenguaje ligero, al ser más sencillo y simple el código, lo que permite que las páginas escritas en este lenguaje carguen más rápido en el navegador (Ruiz, 2014).

Existen numerosas aplicaciones y editores de páginas web que generan el código automáticamente, por lo que no es necesario ser un experto informático para hacer páginas basadas en este lenguaje. Puede ser abierto y modificado con cualquier editor de texto. Es independiente de la plataforma utilizada y se basa fundamentalmente en el uso de etiquetas estructurales y semánticas (Gómez, 2015).

CSS *(Cascading Style Sheets).*

Lenguaje que describe la presentación de los documentos estructurados en hojas de estilo para diferentes métodos de interpretación (Munguía, 2015). Este lenguaje es utilizado para modificar la apariencia de una página web de una forma más sencilla, agilizando su actualización y permitiendo a los desarrolladores controlar el estilo y formato de sus documentos (Gómez, 2015). Los estilos se definen en un archivo .css que se carga junto con la página HTML, permitiendo centralizar los estilos de todo el sitio web.

JavaScript.

Lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios (Gómez, 2015).

Este lenguaje posee varias características, entre ellas se puede mencionar que es un lenguaje basado en acciones, gran parte de la programación en este lenguaje está centrada en describir objetos, escribir funciones que respondan a movimientos del mouse, aperturas, utilización de teclas y cargas de páginas (Munguía, 2015). Uno de los aportes más significativos de JavaScript como lenguaje, son todas las librerías que han sido codificadas y que permiten agregarles mayor

dinamismo, estética, funcionalidad y usabilidad a los sitios web; entre ellas se destacan: jQuery[6] y Node.js[7].

Applets de Java.

Se trata de pequeños programas hechos en Java, que se transfieren con las páginas web y que el navegador ejecuta en el espacio de la página. Los *applets* son más difíciles de programar que los guiones (*scripts*) en JavaScript y requerirán unos conocimientos básicos o medios del lenguaje Java (Vaquero, 2014).

Como desventajas en relación con JavaScript cabe señalar que los *applets* son más lentos de procesar y que tienen espacio muy delimitado en la página donde se ejecutan, es decir, no se mezclan con todos los componentes de la página ni tienen acceso a ellos. Con la proliferación de HTML5, el uso de *applets* ha declinado y, a pesar de los esfuerzos de Oracle, es una tecnología que tiende a desaparecer (Castro, 2016).

Visual Basic Script.

Es un lenguaje de programación de guiones (*scripts*) del lado del cliente, pero sólo compatible con Internet Explorer. Es por ello por lo que su utilización está desaconsejada a favor de JavaScript. Está basado en Visual Basic[8], un popular lenguaje para crear aplicaciones Windows. Presenta como inconveniente que la ejecución se ralentiza, al ser necesaria la interpretación línea a línea cada vez que se introduce una nueva línea de código (Gómez, 2015).

También se necesitan lenguajes de programación del lado del servidor. Estos son reconocidos, ejecutados e interpretados por el propio servidor e independientes del cliente. Son utilizados en el procesamiento de las peticiones de los usuarios para generar páginas web dinámicamente como respuesta (Gómez, 2015).

[6] jQuery: Librería más conocida de Javascript por su potencia y facilidad de uso.

[7] Node.js: Lenguaje de propósito general que usa el motor de Javascript para la ejecución de los programas.

[8] Visual Basic: Lenguaje de programación dirigido por eventos.

1.6.2 Lenguajes de programación del lado del servidor.

PHP (*Hypertext Preprocessor*).

Es un lenguaje interpretado de alto nivel embebido en páginas HTML, enfocado principalmente a la programación de guiones (*scripts*) del lado del servidor, por lo que se puede recopilar datos de formularios, generar páginas con contenidos dinámicos, o enviar y recibir *cookies* (Munguía, 2015). PHP puede ser desplegado en la mayoría de los servidores web y en casi todos los sistemas operativos y plataformas sin costo alguno, es rápido y cuenta con una gran librería de funciones y mucha documentación (Arenas, 2015).

Java

Es un lenguaje de programación de propósito general, concurrente y orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible. Su intención es permitir que los desarrolladores de aplicaciones escriban el programa una vez y lo ejecuten en cualquier dispositivo (Steven, 2012).

Al igual que en PHP tiene una gran comunidad de programadores, pero su curva de aprendizaje es muy pesada, ya que este lenguaje es muy complejo, lo cual no lo hace justificable si el desarrollo es medianamente simple. La implementación de esta tecnología es más costosa por lo tanto es más escasa la oferta de alojamiento web (*hosting*) para la misma. Una de las críticas habituales a Java es que en ocasiones escribir un programa sencillo ocupa muchas líneas de código (Munguía, 2015).

Python

Es un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma. Python tiene una curva de aprendizaje muy suave y también implementa una gran cantidad de bibliotecas como sucede en PHP (Roldán, 2013).

Este lenguaje tiene como desventajas que no hay muchos programadores en esta tecnología, además de la documentación que es muy escasa en español y en inglés. Implementar esta tecnología en web es muy compleja, por lo tanto, los *hostings* que soportan Python son todavía más escasos que en

Java (Munguía, 2015).

ASP (*Active Server Pages*).

Tecnología desarrollada por Microsoft para la creación de páginas dinámicas del lado del servidor, lo que significa que cuando el usuario solicita un documento ASP, las instrucciones de programación dentro del script son ejecutadas para enviar al navegador únicamente el código HTML resultante. ASP se escribe en la misma página web, utilizando el lenguaje Visual Basic Script o Jscript (JavaScript de Microsoft).

Este lenguaje tiene como desventajas que es un código propietario de Microsoft, el consumo de recursos de ASP es importante, por lo que se requieren servidores de mayor capacidad. Es un lenguaje difícil de leer e interpretar, necesitando escribir más código para hacer lo mismo que con otros lenguajes como el PHP (Arenas, 2015).

Fundamentación de los lenguajes de programación utilizados:

El análisis realizado a los diferentes lenguajes de programación arrojó como resultado que serán los que se muestran a continuación los utilizados en el desarrollo del subsistema de alertas para el buscador Orión.

HTML

Se escogió HTML5 como lenguaje de programación del lado del cliente por ser un lenguaje universal que todos los navegadores traen embebido por su gran sencillez. Se tuvo en cuenta que es un lenguaje de código abierto y la facilidad con que se pueden actualizar los contenidos. Además, permite la creación de estilos en formato CCS e incorpora nuevas capacidades JavaScript.

El código es más simple lo que permite hacer páginas más ligeras que se cargan más rápidamente favoreciendo la usabilidad y la indexación en buscadores.

JavaScript

Se escogió además JavaScript como lenguaje de programación del lado del cliente por ser un lenguaje de programación script que se utiliza fundamentalmente para crear páginas web dinámicas.

Es un lenguaje interpretado, por lo que no es necesario compilar los programas para ejecutarlos. Está basado directamente en objetos y guiado por eventos. Puede utilizarse conjuntamente y muy fácilmente con el lenguaje HTML.

CSS

Se escogió también CSS3 como lenguaje de programación del lado del cliente ya que permite, una vez creados los contenidos mediante el lenguaje HTML/XHTML, definir la manera en la que se va a mostrar la presentación de los documentos. CSS3 es la mejor forma de separar los contenidos y su presentación y es imprescindible para la creación de páginas web complejas. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes.

PHP

Se escogió PHP5 como lenguaje de programación del lado del servidor por ser un lenguaje de alto nivel e interpretado, utilizado en su mayoría para el procesamiento dinámico de información en la Web. Dicha elección se basa fundamentalmente en las características y múltiples ventajas que proporciona; teniendo en cuenta que es un lenguaje completamente orientado al desarrollo de aplicaciones web dinámicas con acceso a información almacenada en una base de datos. Además, es uno de los lenguajes de programación web que más documentación posee, dispone de un sitio oficial que brinda toda la documentación que los desarrolladores necesiten, y cuenta con una gran comunidad de desarrollo. Lo soportan la mayoría de las plataformas de alojamiento web y está en continuo desarrollo.

1.7 Marco de trabajo (*framework*).

En el desarrollo de software, un *framework* es una estructura conceptual y tecnológica de soporte definida, normalmente, con artefactos o módulos de software concretos. Los objetivos principales que persigue son: acelerar el proceso de desarrollo, reutilizar código ya existente y promover buenas prácticas de desarrollo como el uso de patrones (Alcalde, 2012). A continuación, se realiza un estudio de marcos de trabajo para el desarrollo de software, con el objetivo de escoger el más conveniente para la implementación del subsistema de alertas para el buscador Orión.

Symfony2.

Symfony2 es un marco de trabajo PHP pensado y diseñado para optimizar el desarrollo de aplicaciones web, basado en el patrón Modelo Vista Controlador. Este marco de trabajo separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación web (Esaú, 2015). Es compatible con la mayoría de los Sistemas Gestores de Bases de Datos (SGBD). Se puede ejecutar tanto en plataformas Linux como en plataformas Windows. Requiere para su uso de PHP 5.3 o superior. Incorpora las mejores ideas del resto de los marcos de trabajo, tanto de los que están desarrollados en PHP como los que no (Boyarynoy, 2013).

Los usuarios normales pueden iniciar rápidamente el desarrollo mediante el uso de una distribución Symfony2, que proporciona un esqueleto de proyecto con parámetros por defecto.

Zend framework.

Es un marco de trabajo de código abierto de aplicaciones web, orientado a objetos e implementado en PHP 5, desarrollado con el objetivo de simplificar el desarrollo web. Permite a los desarrolladores reutilizar componentes en sus aplicaciones (Esaú, 2015).

Es necesario comprender algunos patrones de diseño y programación orientada a objetos para utilizar todo el potencial de Zend Framework. Y eso no sólo requiere tiempo dedicado al aprendizaje, sino también la experiencia de trabajar con él (Rosa, 2015). Zend Framework presenta como desventaja que es una herramienta pesada que gasta mucha memoria, además de ser muy pobre la comunidad de desarrolladores, por lo que podría ser difícil encontrar un buen manual para realizar un proceso o aclarar una duda. La estructura de archivos de Zend no está definida por sí misma, cualquier desarrollador puede modificarla, esto indica que no sería compatible con otra aplicación desarrollada con Zend (Rivera, 2016).

CodeIgniter.

Es un marco de trabajo para aplicaciones web en PHP de código abierto, basado en el paradigma de programación Modelo Vista Controlador. Entre sus características fundamentales se encuentran su excelente rendimiento y su amplia documentación (Esaú, 2015)**.**

CodeIgniter no tiene un orden por defecto para las vistas, no tiene sistema de plantillas y tampoco módulos. Además, señalar que los controladores no cargan por defecto las listas (Rosa, 2015).

Fundamentación del marco de trabajo utilizado:

Luego de realizar un estudio a los diferentes marcos de trabajo se define Symfony en su versión 2.8 para la implementación del subsistema de alertas, por ser uno de los marcos de trabajo de PHP con mejor rendimiento y hacer uso de los lenguajes HTML 5 y CCS 3. Symfony2 posee características que lo hacen único con respecto a otros *frameworks*, como el soporte HTTP, el contenedor de inyección de dependencias, el uso de plantillas *twigs*, el sistema de paquetes (*bundles*) y su fácil integración con cualquier SGBD, tanto para esquemas relacionales como no relacionales. Existe abundante documentación en diferentes idiomas, fundamentalmente en inglés, español y francés.

1.8 *Framework* de CSS y JavaScript

Se realiza además, el estudio de marcos de trabajo para CSS y JavaScript permitiendo decidir el más idóneo para el desarrollo del subsistema de alertas para el buscador Orión.

Boostrap.

Es una colección de herramientas de software libre para la creación de sitios y aplicaciones web. Contiene plantillas de diseño basadas en HTML y CSS con tipografías, formularios, botones, gráficos, barras de navegación y demás componentes de interfaz, así como librerías JavaScript y Less[9] (Mur, 2015). Boostrap permite la adaptación de la interfaz dependiendo del tamaño del dispositivo en el que se visualice sin que el usuario tenga que hacer nada, esto se denomina *Responsive Web Design*[10]. La mayor ventaja que presenta este marco de trabajo es que se pueden crear interfaces que se adapten a los distintos navegadores como Chrome[11], Firefox[12], Safari[13] y Opera[14], con tabletas y teléfonos inteligentes.

[9] Less: Lenguaje de hojas de estilo dinámico, que permite introducir funciones y variables en las CSS.

[10] *Responsive Web Design*: Es la técnica que permite crear sitios adaptables a las condiciones del ordenador o dispositivo desde donde se van a acceder, sobre todo en lo que tiene relación con la pantalla del sistema donde se están visualizando.

[11] Chrome: Navegador web desarrollado por Google.

[12] Firefox: Navegador web libre y de código abierto desarrollado para Microsoft Windows, Android, OS X y GNU/Linux.

[13] Safari: Navegador web de código cerrado desarrollado por *Apple Inc*.

[14] Opera: Navegador web creado por la empresa noruega *Opera Software*.

Foundation 4.

Es un marco de trabajo que consta de herramientas útiles para la toma de respuesta, primero los sitios web móviles, construido con HTML, CSS y jQuery, utiliza tecnologías y prácticas modernas (Mur, 2015).

Foundation 4 se utiliza para aplicaciones móviles proporcionando tiempos de carga mínimos para aquellos usuarios que naveguen desde sus móviles. Presenta como limitación que se debe tener conocimiento en los estándares de aplicaciones móviles.

Backbone.js.

Permite al desarrollador estructurar la aplicación, delegando ciertas labores a elementos específicos con el fin de hacer aplicaciones escalables, robustas y que brinden al usuario una gran experiencia de uso. Este marco de trabajo permite la programación atendiendo al paradigma Modelo Vista Controlador y ayuda a pintar interfaces de usuario en la página (Vereau, 2015). Presenta como desventaja que la curva de aprendizaje es lenta si no hay conocimientos previos en jQuery.

Fundamentación del *framework* de CSS y JavaScript utilizado:

Como *framework* de CCS y JavaScript se utiliza Boostrap en su versión 3.0 por su compatibilidad con todos los navegadores habituales. Ofrece una serie de plantillas CSS y ficheros JavaScript que permiten integrar el marco de trabajo de forma sencilla y potente en proyectos web, además cuenta con una documentación extensa y detallada, en la cual se encuentran ejemplos fáciles para el uso de los componentes y el diseño web con Boostrap.

1.9 Entorno de Desarrollo Integrado (IDE).

Un IDE, traducido del inglés *Integrated Development Environment* es un entorno de programación que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica. Proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como: C++[15], PHP, Python, Java, C#[16] (Rouse, 2016). A continuación, se realiza un estudio de algunos IDE,

[15] C++: lenguaje de programación estructurado y orientado a objetos.

con el objetivo de escoger el más conveniente para el desarrollo del subsistema de alertas para el buscador Orión.

Eclipse.

Fue desarrollado por OTI por sus siglas en inglés *Object Technology International* como un proyecto de IBM[17] Canadá. Eclipse es un entorno de desarrollo integrado, de código abierto y multiplataforma para desarrollar proyectos. En Eclipse se pueden usar diferentes lenguajes de programación como: Java, C++ y PHP, aunque pretende ser un entorno versátil soportando varios lenguajes de programación, es con el lenguaje Java con el que mejor se integra y con el que ha ganado su popularidad (Quintero, 2012). Eclipse consume muchos recursos del equipo, por lo que se debe tener una buena tarjeta gráfica para su uso. No ofrece la posibilidad de crear todos los diagramas UML y es un IDE de baja calidad.

NetBeans.

Entorno de desarrollo de código abierto para que los programadores puedan escribir, compilar, depurar y ejecutar programas. Está escrito en Java, pero da soporte a otras tecnologías como: PHP, C/C++, HTML5. Este, además de ser gratuito y sin restricciones de uso, posee versiones para los distintos sistemas operativos del mercado, convirtiéndolo en una alternativa con grandes ventajas para los desarrolladores (Jiménez, 2014).

Geany.

Es un editor de texto pequeño y ligero con características básicas de IDE y utiliza bibliotecas GTK para su funcionamiento. Está disponible para distintos sistemas operativos, como GNU/Linux y Microsoft Windows. Es distribuido como software libre bajo la Licencia Pública General de GNU. Entre sus características se encuentran su compatibilidad con la mayoría de lenguajes como C, Java, PHP, Python, Perl y Pascal (Cherencio, 2011). Este editor tarda en guardar los archivos y le faltan herramientas para la corrección de código.

---

[16] C#: lenguaje de programación independiente diseñado para generar programas sobre la plataforma .NET.

[17]IBM: Empresa multinacional estadounidense de tecnología y consultoría que fabrica y comercializa hardware y software para computadoras.

Fundamentación del IDE utilizado:

El análisis realizado a los diferentes IDE arrojó como resultado que será NetBeans en su versión 7.2 el entorno de programación usado por sus grandes posibilidades de ser extendido, permitiendo a los desarrolladores agregar continuamente nuevas funcionalidades. Además, posee un buen editor de código que cuenta con comprobaciones sintácticas y semánticas, permitiendo el completamiento de código, así como el depurado de errores. NetBeans propone un esqueleto para organizar el código fuente, el editor conjuntamente integra los lenguajes como HTML, JS y CSS.

Una característica esencial a tener en cuenta para su selección es que se integra con el marco de trabajo Symfony2 usado en este trabajo del cual posee algunas bibliotecas y *plugins* que el programador puede agregarle a su aplicación.

1.10 Sistema Gestor de Base de Datos (SGBD).

Un SGBD consiste en una colección de datos interrelacionados y un conjunto de programas para acceder a los mismos. Permite el almacenamiento, manipulación y consulta de datos pertenecientes a una base de datos organizada en uno o varios ficheros (Ramírez, 2012). A continuación, se realiza un estudio de SGBD con el objetivo de escoger el más conveniente para el trabajo con los datos del subsistema de alertas para el buscador Orión.

PostgreSQL.

PostgreSQL es un SGBD orientadas a objetos muy conocido y usado en entornos de software libre porque cumple los estándares SQL92 y SQL99. PostgreSQL puede funcionar en plataformas Unix y también en Windows. Durante su desarrollo se han tenido en cuenta características como la estabilidad, potencia, robustez, facilidad de administración e implementación de estándares (Bolton, 2015). Este SGBD es fácil de vulnerar sin protección adecuada y presenta una reducida cantidad de tipos de datos, además consume más recursos y es más lento que MySQL.

MySQL.

Este SGBD es muy conocido y ampliamente usado por su simplicidad y notable rendimiento. Aunque carece de algunas características avanzadas disponibles en otros SGBD del mercado, es una opción

atractiva tanto para aplicaciones comerciales como de entretenimiento. Esto y su libre distribución en Internet bajo licencia GPL le otorgan como beneficios adicionales, (no menos importantes), contar con un alto grado de estabilidad y un rápido desarrollo (Bolton, 2015). Está disponible en gran cantidad de plataformas y sistemas. Un gran porcentaje de las utilidades de MySQL no están documentadas y se utiliza en aplicaciones web no muy complejas que requieren muchos usuarios.

MongoDB.

MongoDB (de la palabra en inglés "*humongous*" que significa enorme) es un sistema de bases de datos no relacionales, multiplataforma e inspirada en el tipo de bases de datos documental y clave/valor. Está liberada bajo licencia de software libre, específicamente GNU AGPL 3.0[18]. MongoDB es un sistema de código abierto y escrito en C++, orientado al almacenamiento de datos en documentos al estilo JSON con esquemas dinámicos, que ofrecen potencia y simplicidad, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida (Serrano, 2014). Permite que los datos sean distribuidos en distintos servidores, balanceando la carga entre ellos.

Se destaca por conservar los índices de todos los atributos y hacer mucho más flexible la agregación y procesamiento de datos. Los datos en MongoDB tienen un esquema flexible, por lo que, a diferencia de las bases de datos relacionales, no se debe determinar y declarar el esquema de una tabla antes de introducir los datos. Esta flexibilidad facilita el mapeo de documentos a una entidad o un objeto.

Fundamentación del SGBD utilizado:

El análisis realizado a los diferentes SGBD arrojó como resultados que será MongoDB 2.6 el usado para la implementación del subsistema. Se selecciona este por lograr un mejor manejo de la información frente a un volumen considerable de datos, pues se utilizará para el almacenamiento de una gran cantidad de criterios de búsqueda realizados por los usuarios. MongoDB ofrece un gran rendimiento a la hora de insertar datos mucho mayores que otros sistemas de base de datos tradicionales.

[18]GNU AGPL 3.0: licencia *copyleft* derivada de la Licencia Pública General de GNU.

Permite una fácil integración con el lenguaje de programación seleccionado, además de que el marco de trabajo a utilizar, Symfony2, posee soporte para MongoDB. Además, se tuvo en cuenta para su elección que es este SGBD es el usado en CIDI para el desarrollo del buscador Orión y teniendo en cuenta que el subsistema de alertas debe integrase a dicho buscador se escoge el mismo SGBD, en este caso MongoDB 2.6.

1.11 Servidor de aplicación.

La piedra angular de cualquier sitio o portal web es, con toda seguridad, su servidor web; el software encargado de atender las peticiones de los clientes y ejecutar las páginas web solicitadas (Tusa, 2012). Un servidor de aplicaciones generalmente gestiona la mayor parte de las funciones de lógica de negocio y de acceso a los datos de la aplicación. A continuación, se caracterizan algunos servidores de aplicación.

Apache.

Apache es un poderoso servidor web, cuyo nombre proviene de la frase inglesa "*a patchy server*" y es completamente libre. Una de las ventajas de Apache, es que es un servidor web multiplataforma, es decir, puede trabajar con diferentes sistemas operativos y mantener su excelente rendimiento. Este estupendo servidor es utilizado principalmente, para realizar servicio a páginas web, ya sean estáticas o dinámicas y una de sus principales características es que se integra a la perfección con otras aplicaciones (Balkhi, 2013).

Este servidor posee limitaciones en las versiones que no son de la familia "*Server*" y formatos de configuración no estándar. No cuenta con una buena administración y es lento en el arranque debido a la sobrecarga que la resolución representa para el cargador (Wajser, 2015).

***Internet Information Server* (IIS).**

IIS es un servidor web que permite publicar información en una intranet o en Internet. Este servidor web se apoya sobre el protocolo de transferencia de hipertexto HTTP, para transmitir la información y comunicarse con los clientes. Es el servidor web sobre el que se ejecutan las páginas ASP.NET (Salas, 2013). Este servidor no es multiplataforma por lo que funciona solamente sobre el sistema operativo Windows, además ocupa mucho más espacio en disco que otros servidores.

NGINX.

Es un servidor web, de código abierto y desarrollado casi en su totalidad en lenguaje C, lo que le proporciona un alto rendimiento aprovechando al máximo los recursos del sistema. Incluye también servicios de correo electrónico con acceso al *Internet Message Protocol* (IMAP) y al servidor *Post Office Protocol* (POP).

NGINX actúa como la capa entre el cliente y la aplicación web. Aquí se pone en marcha el almacenamiento en caché de cualquier sitio al que NGINX está actuando como un proxy inverso. Entre sus principales características se encuentra que posee servidores virtuales basados en nombre y/o en dirección IP, soporte para autenticación y es compatible con IPv6[19] (Wajser, 2015). Procesa una gran cantidad (miles de decenas) de conexiones simultáneas en un proceso compacto y con varios núcleos de CPU de las cuales sólo tienes el número correspondiente de procesos de NGINX para escalar realmente bien.

Fundamentación del servidor de aplicación utilizado:

Se seleccionó NGINX en su versión 2.0 como servidor web y de aplicaciones por sus principales características que se asocian con el rendimiento, la escalabilidad y la eficiencia de costes. Es mucho más ligero que Apache, por lo tanto, realiza menos funciones. Es un servidor de archivos estáticos, índices y auto indexado, permitiendo el balance de carga y la tolerancia a fallos.

1.12 Sistemas para el intercambio de procesos de forma asíncrona.

Cuando se desarrolla una aplicación y esta comienza a crecer, a menudo se necesita interconectar distintos componentes. En estos casos se utiliza un *middleware*[20] que permita comunicar las distintas piezas. A continuación, se realiza un estudio sobre diferentes sistemas que permiten el intercambio entre procesos de forma asíncrona.

RabbitMQ.

[19] IPv6: Protocolo de Internet versión 6.

[20] *Middleware*: Lógica de intercambio de información entre aplicaciones. Software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware y/o sistemas operativos.

RabbitMQ es una herramienta de encolamiento de mensajes o gestor de colas, muy potente y escalable, desarrollado con Erlang[21]. Este sistema permite escribir mensajes con un solo proceso, llamado productor, y luego vuelve a leer desde la cola con otro proceso, llamado consumidor (Johansson, 2015).

Los mensajes en colas se almacenan de forma independiente el uno del otro. Cada mensaje contiene todo lo que se necesita para ser procesado.

Hay algunos conceptos que hay que tener en cuenta si se desea trabajar con RabbitMQ (ver anexo 1):

- ✓ Intercambiar (e*xchange*): Es el punto de entrada de un mensaje. Pueden ser directo (*direct*) si entregan un mensaje en una cola, despliegue en abanico (*fanout*) si se entregan copias del mensaje a todas las colas o tema (*topic*) si se entregan copias del mensaje sólo a algunas colas.
- ✓ Cola (*queue*): Es el punto de lectura de un mensaje. Pueden persistentes si almacenan los mensajes para sobrevivir a un reinicio de RabbitMQ. También pueden ser exclusivas, si sólo un consumidor puede estar conectado a la vez.
- ✓ Encuadernaciones (*bindings*): Son reglas que indican cómo llegar de un intercambio a las colas asociadas.
- ✓ Enrutamiento clave (r*outing key*): Filtro asociado a un *binding* que permite seleccionar sólo algunos mensajes para dicho *binding*.
- ✓ Productor (*producer*): Programa que escribe en un intercambio.
- ✓ Consumidor (*consumer*): Programa que escucha en una cola.
- ✓ AMQP: Protocolo de comunicaciones utilizado por RabbitMQ. Lo usa tanto el Productor como el Consumidor.
- ✓ Virtual host o vhost: Un entorno aislado, con sus propios grupos de usuarios.
- ✓ Mnesia: La base de datos interna de RabbitMQ.

Ventajas del uso de RabbitMQ.

[21]Erlang: Lenguaje de programación orientado a la concurrencia diseñado para realizar aplicaciones distribuidas y de funcionamiento ininterrumpido.

- ✓ Escalabilidad: proceso de un consumidor que lee un mensaje de una cola, pero no hay nada que limitar a un consumidor. Por lo que podría, por ejemplo, procesar la cola usando tantos consumidores como sea necesario.
- ✓ La agrupación: varios servidores RabbitMQ en una red local pueden ser agrupados juntos, formando un único agente lógico.
- ✓ Colas de alta disponibilidad: las colas se pueden reflejar a través de varias máquinas en un clúster, asegurando que incluso en caso de fallo de hardware sus mensajes son seguros.
- ✓ Las aplicaciones más sensibles: algunas acciones requieren mucho tiempo como se ha mencionado, como el envío de un correo electrónico de registro o la creación de miniaturas. Se puede evitar un retraso para el usuario, delegando la tarea costosa a un consumidor especializado que lo procesará de forma asíncrona.

Kafka

Kafka es un sistema de mensajería de código abierto desarrollado para solventar las limitaciones de los sistemas de mensajería. Está escrito en Java y ahora bajo el proyecto Apache. Almacena los mensajes en archivos planos y los consumidores piden mensajes en función de un desplazamiento.

Kafka tiene como objetivo unificar el procesamiento de análisis *offline* y *onlin*e, así como proporcionar la capacidad de partición del consumo en tiempo real de un *cluster* de máquinas. Es un sistema de mensajería persistente, escalable, replicado, tolerante a fallos, capaz de atender cientos de megas por segundo de lecturas y escrituras provenientes de miles de clientes. Esta disponibles para los lenguajes de programación Scala, Phyton y Java (Lorite, 2015).

Fundamentación del sistema para el intercambio de procesos de forma asíncrona utilizado:

Se seleccionó RabbitMQ en su versión 3.6 como herramienta para el encolamiento de mensajes por responder rápidamente a las solicitudes, en lugar de ser forzados a realizar procedimientos de recursos pesados en el acto. También se tuvo en cuenta para su elección que se puede distribuir un mensaje a varios destinatarios disminuyendo el consumo y logrando un equilibrio de cargas entre los trabajadores.

La disponibilidad es otro elemento a tener en cuenta ya que en caso de que una parte del subsistema falle los mensajes no se perderán, estos seguirán en la cola, al mismo tiempo la cola podrá seguir recibiendo mensajes para el procesamiento posterior a la recuperación del subsistema.

RabbitMQ presenta notificación de recibo de mensajes que utiliza Mnesia para mantener el estado de entrega del mensaje mientras que Kafka no presenta esta característica. Es simple de implementar y posee una buena documentación.

Además, el marco de trabajo utilizado para el desarrollo del subsistema Symfony2 posee un *bundle* llamado RabbitMqBundle que incorpora mensajería utilizando la biblioteca de php-amqplib, en el caso de Kafka no está disponible para el lenguaje PHP.

1.13 Otras herramientas utilizadas.

Para el desarrollo del subsistema de alertas se hace necesario utilizar algunas herramientas que no fue necesario realizar un estudio para seleccionar la más convenientes, ya que son herramientas definidas por el centro CIDI y por la Universidad. A continuación, se presentan algunas características de cada una de estas herramientas.

Indexador Solr.

Es una plataforma de búsquedas basada en *Apache Lucene*, que funciona como un "servidor de búsquedas". Sus principales características incluyen búsquedas de texto completo, resaltado de resultados y manejo de documentos (como Word y PDF). Solr es escalable, permitiendo realizar búsquedas distribuidas y replicación de índices, así como la integración de bases de datos (Sironi, 2013). Solr utiliza la biblioteca de *Java Lucene* para la búsqueda e indexación de texto completo y tiene APIs que hacen que sea fácil de utilizar desde cualquier lenguaje de programación. Presenta una configuración externa de Solr, permitiendo adaptarse a casi cualquier tipo de aplicación sin necesidad de codificación en Java.

Teniendo en cuenta que el buscador Orión hace uso de Solr 4.10.3 para el indexado de documentos y que el subsistema de alertas permite realizar solicitudes de alertas mediante este buscador se decide usar Solr como Indexador de documentos.

Apache Jmeter

Es una herramienta de código abierto diseñada para pruebas de carga de comportamientos funcionales y la medición del rendimiento. Prueba la resistencia y analiza el rendimiento en diferentes tipos de carga. Para la realización de las pruebas de carga y estrés al subsistema de alertas se utiliza esta herramienta en su versión 2.9 (Vicente, 2014).

Acunetix Web Vulnerability Scanner

Es la herramienta utilizada en la Universidad para la detección de vulnerabilidades en sitios y aplicaciones web. Esta herramienta posee un componente que facilita la realización de pruebas a formularios y a áreas protegidas por contraseña. Permite realizar varias peticiones simultáneamente, siendo capaz de explorar cientos de páginas sin interrupciones. Para la realización de las pruebas de seguridad al subsistema de alertas se utiliza esta herramienta en su versión 8.0 (Ortega, 2014).

1.14 Conclusiones Parciales

En este capítulo se han abordado los elementos correspondientes a la fundamentación teórica del subsistema de alertas, arribando a las siguientes conclusiones:

- ✓ Los antecedentes de los sistemas de alertas existentes en lo nacional e internacional, permitió determinar que no existe un sistema de alertas que satisfaga las necesidades del centro CIDI para el envío continuo de noticias.
- ✓ El estudio de las herramientas y tecnologías permitió definir la base tecnológica a utilizar en el desarrollo del subsistema de alertas para el buscador Orión.

Capítulo 2. Características y diseño del subsistema de alertas para el buscador Orión

2.1 Introducción

En el presente capítulo se diseña la propuesta de solución para el subsistema de alertas, especificando los requerimientos funcionales y no funcionales. Se describe dicho subsistema a partir de los casos de uso y la descripción de los mismos. Se establecen además los patrones de diseño evidenciando su utilización en la solución propuesta.

2.2 Flujo actual de los procesos.

El subsistema que se propone desarrollar permitirá notificar a los usuarios a través del correo electrónico sobre nuevos contenidos encontrados en la Web que sean de su interés.

El usuario primeramente debe registrarse en el subsistema, introduciendo los datos necesarios como son: nombre de usuario, contraseña, dirección de correo electrónico donde recibirá la notificación, nombre completo, sexo y fecha de nacimiento. Luego, este usuario previamente autenticado realiza una solicitud de alerta, proponiendo el criterio de búsqueda de su interés. Estas solicitudes se almacenan en la base de datos de MongoDB. Mientras, hay un comando ejecutándose constantemente y encuestando a la base de datos para obtener las solicitudes que no han sido atendidas o aquellas que nuevamente deben ser consideradas, por haber vencido su lapso temporal desde su última atención. Estas solicitudes que deben ser analizadas se ponen en una cola de RabbitMQ, para ser procesadas de forma asíncrona[22]. Los procesos consumidores o *workers* se encuentran esperando por la recepción de mensajes, desde la cola de RabbitMQ. Una vez el proceso consumidor ha recibido un mensaje que contiene el identificador de una solicitud, encuesta a la base de datos de MongoDB para obtener esta solicitud, de la cual extrae el criterio de búsqueda y los datos del usuario que la realizó. Con el criterio de búsqueda y utilizando los servicios previamente declarados en el buscador Orión se prepara y ejecuta la consulta sobre los datos indexados en el servidor de Solr. Una vez se han obtenido los resultados de la consulta los mismos son plasmados en el cuerpo de un correo electrónico que es enviado a la cuenta de correo definida en los datos del usuario que solicitó la solicitud.

[22] Asíncrona: Proceso de comunicación entre emisor y receptor que no tiene un intervalo de tiempo constante entre cada evento (Araujo, 2015).

2.3 Especificación de los requerimientos del software.

Con el objetivo de establecer los servicios que el subsistema de alertas debería proveer y las restricciones bajo las cuales debería operar y ser desarrollado, se definen los requisitos funcionales y no funcionales.

2.3.1 Requisitos Funcionales (RF).

Los RF no son más que las condiciones o capacidades que el sistema debe cumplir (Pfleeger, 2001). A continuación, se relacionan los requisitos funcionales que debe cumplir el subsistema de alertas para el buscador Orión. Para una descripción más detallada (ver anexo 2).

Tabla 1: Requisitos funcionales

Nº	Nombre	Prioridad
RF1	Solicitar Alerta	Baja
RF2	Generar Alerta	Alta
RF3	Modificar Alerta	Media
RF4	Buscar Alerta	Media
RF5	Eliminar Alerta	Media
RF6	Mostrar recomendaciones de alertas	Media
RF7	Listar recomendaciones de alertas.	Media
RF8	Editar recomendaciones de alertas.	Media
RF9	Eliminar recomendaciones de alertas.	Media
RF10	Buscar recomendaciones de alertas.	Media

RF11	Crear recomendaciones de alertas.	Media
RF12	Generar solicitud de alerta basada en recomendación de alerta.	Media

2.3.2 Requisitos no funcionales (RnF).

Un RnF es un requisito de software que describe no lo que el software hará, sino como lo hará (Dorfman, y otros, 1990). A continuación, se relacionan los requisitos no funcionales que debe cumplir el subsistema de alertas para el buscador Orión.

Tabla 2: Requisitos no funcionales

Usabilidad	
RnF 1	El subsistema debe poseer una interfaz de manejo cómoda, que posibilite a los usuarios sin experiencia una rápida adaptación.
Software	
RnF 2	Las estaciones de trabajo (dispositivo cliente) utilizan los navegadores web Mozilla Firefox 2.0 o superior e Internet Explorer 9.0.
RnF 3	Se requiere la instalación de un servidor web (recomendándose, NGINX en su versión 2.0) y PHP 5.5 o superior para el correcto funcionamiento de la aplicación web.
RnF 4	Para la indexación de documentos se deberá utilizar la herramienta Solr.
Hardware	
RnF 5	Los servidores donde estarán desplegados el componente de rastreo, el de indexación y la aplicación web deben tener como recursos mínimos

	de hardware: 4 GB de RAM, un microprocesador Core i3 con una velocidad 2.10 GHz y un disco duro con una capacidad mínima de 500 GB.
Rendimiento	
RnF 6	El subsistema debe responder en un máximo de 15 segundos las solicitudes de los usuarios.
RnF 7	El sistema debe soportar un mínimo de 100 peticiones.
Soporte	
RnF 8	El soporte del subsistema se debe gestionar mediante el Centro de Soporte de la UCI.
Mantenimiento y soporte	
RnF 9	El subsistema funcionará en varios sistemas operativos como Windows y Linux.
Interfaz	
RnF 10	Para la realización del diseño se requiere homogeneidad con la identidad del motor de búsqueda Orión.
RnF 11	Se requiere una interfaz visualmente agradable y que posibilite al usuario una navegabilidad intuitiva por las funcionalidades del subsistema.
Seguridad	
RnF 12	El subsistema debe contar con un conjunto de roles que hacen más segura la aplicación restringiendo los permisos de cada usuario.

2.4 Casos de usos del sistema.

Los casos de uso (CU) son un conjunto de escenarios que identifican una línea de utilización para el sistema que va a ser construido y que facilitan una descripción de cómo el sistema se usará (Hernández, 2016). En este epígrafe se definen los actores, se modelan los diagramas de casos de usos del sistema (DCUS) y se realiza la especificación de los CU que se deben tener en cuenta para el desarrollo del subsistema de alertas.

Definición de los actores del sistema.

Tabla 3: Definición de los actores del sistema.

Actor	Descripción
Usuario	Gestiona las alertas sobre los contenidos indexados en el buscador.
Administrador	Gestiona los componentes alertas.
Cron	Emite señales temporales para el inicio de eventos del sistema.

Diagrama de casos de usos del sistema.

Un DCUS muestra la relación entre los casos de uso y los actores del sistema (Hernández, 2016). Para satisfacer los requisitos que debe cumplir el subsistema, se representa en la siguiente imagen la relación existente entre los actores y los casos de uso que deberán interactuar con el subsistema de alertas.

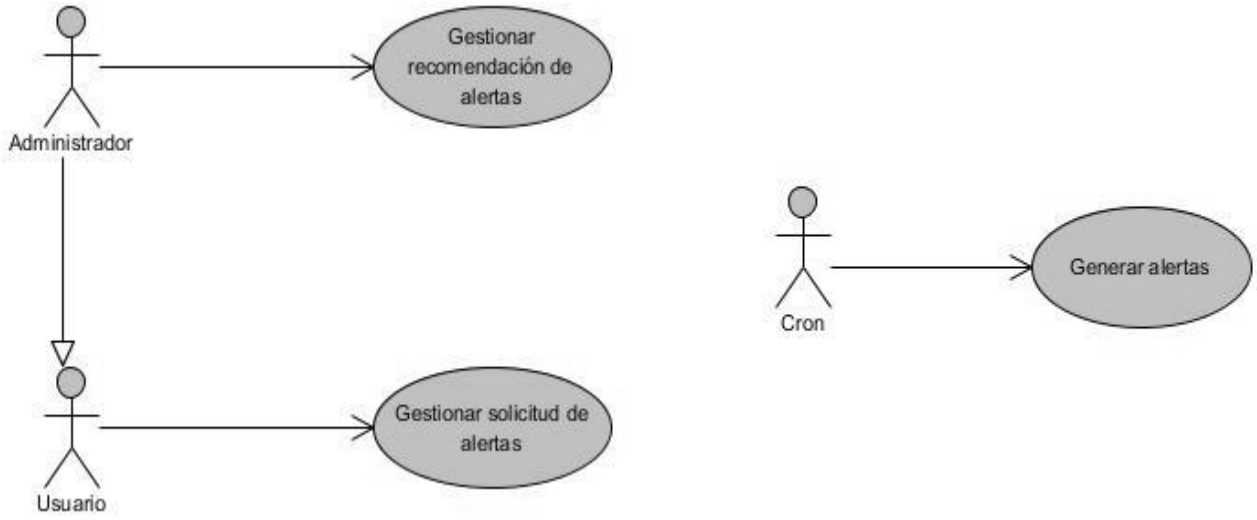

Ilustración 1: Diagrama de casos de uso del sistema.

Descripción de Casos de Usos.

La descripción de los CU son los pasos que sigue el actor para interactuar con el sistema (Rodríguez, 2015). A continuación, se muestra la descripción del CU Generar alerta y para las descripciones correspondientes al resto de los CU (ver anexo 3):

Generar Alerta

Tabla 4: Descripción CU generar alerta

Objetivo	Generar y enviar por correo electrónico automáticamente la alerta seleccionada
Actores	Cron
Resumen	El susbsietma permite generar y enviar por correo electrónico automáticamente la alerta solicitada por el usuario.
Complejidad	Media

Prioridad	Media	
Precondiciones	El usuario debe estar autenticado en el sistema	
Pos condiciones	El usuario recibe correctamente la alerta seleccionada	
Flujo de eventos		
Flujo básico: Generar Alerta		
	Actor	**Sistema**
1	Accede a la vista “Seleccionar Alerta”.	
2		Muestra la vista “Seleccionar Alerta”.
3	Selecciona la Alerta deseada	
4		Valida los datos Envia automáticamente por correo elctrónico la alerta seleccionada Muestra el siguiente mensaje “La acción ha sido satisfactoria.”
Flujos alternos : El usuario presiona la opción Cancelar		
	Actor	**Sistema**
1	Selecciona la opción “**Cancelar”.**	
2		Regresa a la interfaz anterior y muestra el mensaje de información: “La acción ha sido cancelada.”.
Relaciones	CU incluido	-

	CU extendidos	-
Requisitos no funcionales		
Asuntos pendientes		
Prototipo elemental de interfaz gráfica de usuario		
No aplica		

2.5 Arquitectura de la solución propuesta.

La arquitectura de software se refiere a “las estructuras de un sistema, compuestas de elementos con propiedades visibles de forma externa y las relaciones que existen entre ellos” (Len, y otros, 1999).

Teniendo en cuenta el sistema propuesto, se sugiere la arquitectura que se muestra en la siguiente imagen:

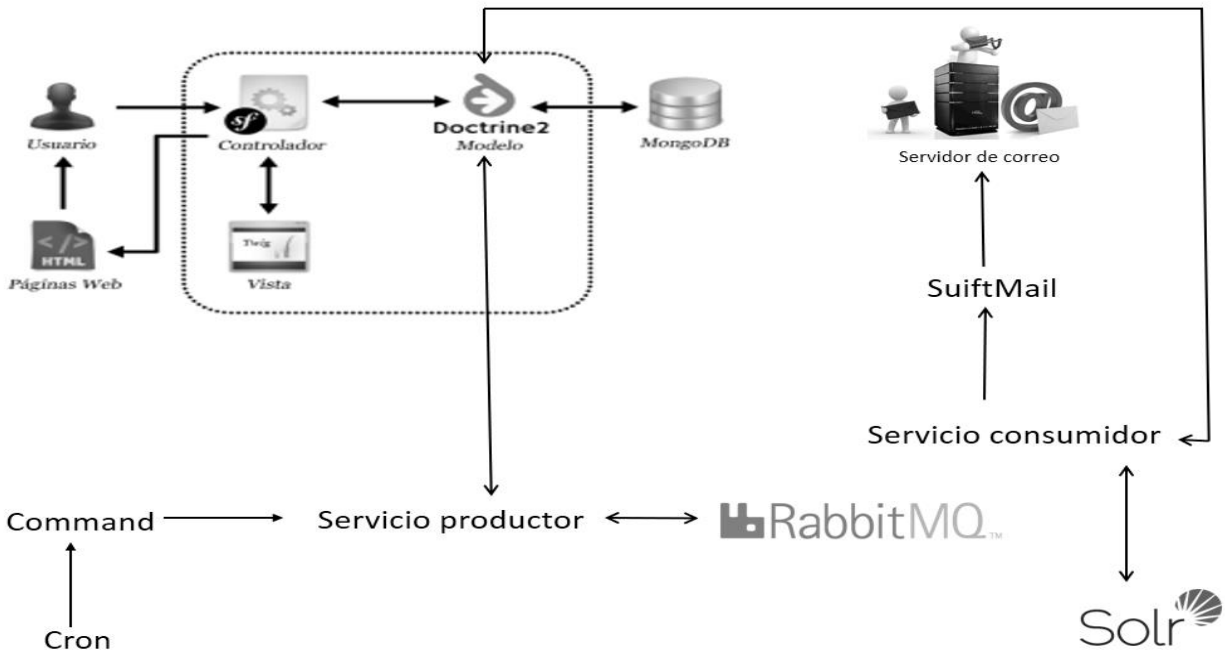

Ilustración 2: Arquitectura del sistema propuesto

Como se muestra en la imagen anterior, el sistema que se propone desarrollar está constituido por diferentes componentes, los cuáles serán explicados a continuación:

RabbitMQ: es el encragado de manejar los mensajes a través de la biblioteca php-amqplib del *bundle* RabbitMqBundle que posee Symfony2.

Solr: se encarga del indexado de documentos.

Cron: se encarga de emitir señales temporales para el inicio de eventos del sistema.

Command: es un comando que se ejecuta constantemente encargado de encuestar a la BD a través del servicio productor.

Servicio productor: encargado de poner en la cola de RabbitMQ las solicitudes no atendidas.

Servicio consumidor: encargado de sacar de la cola de RabbitMQ las solicitudes que deben ser enviadas por correo electrónico.

SwiftMail: es una librería de Symfony encargada del trabajo con mensajes por correo electrónico.

Servidor de correo: encargado de recibir las alertas que serán notificadas a la dirección de correo introducida por el usuario.

2.6 Descripción de estilos arquitectónicos y/o patrones de diseño.

A continuación, se realiza un análisis de los estilos arquitectónicos y patrones de diseño utilizados en el subsistema de alertas para el buscador Orión.

2.6.1 Patrón arquitectónico.

Modelo Vista Controlador (MVC).

La arquitectura MVC separa la lógica de negocio y la presentación, por lo que se consigue un mantenimiento más sencillo de las aplicaciones (Pantoja, 2008).

Para el desarrollo del subsistema se decide emplear el patrón arquitectónico MVC, ya que el marco de trabajo Symfony2 brinda ciertas facilidades que permiten la construcción de aplicaciones web basadas en la estructura y funcionamiento que propone este patrón.

En la figura que se muestra a continuación, se representa la estructura que propone el patrón MVC teniendo en cuenta los componentes que ofrece el marco de trabajo Symfony2 para la implementación de aplicaciones.

El principio más importante de la arquitectura MVC es la separación del código del programa en tres capas, dependiendo de su naturaleza. La lógica relacionada con los datos se incluye en el modelo, el código de la presentación en la vista y la lógica de la aplicación en el controlador.

- ✓ Modelo: Es el componente que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. Este componente se evidencia en la clase **OrionAlertRequest.php** ya que se encarga de trabajar con la representación real de los datos.
- ✓ Vista: Es el componente que maneja la presentación visual de los datos contenidos por el modelo. Genera una representación visual del modelo y muestra los datos al usuario. Este componente se

evidencia en la clase _**form.html.twig** ya que se encarga de proveer la interacción con el usuario para visualizar las interfaces que permiten realizar solicitudes de alertas, así como mostrar los datos e informar de errores.

- ✓ Controlador: Es el componente que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del modelo o por alteraciones de la vista. Este componente se evidencia en la clase **OrionAlertRequestController.php** ya que contiene el código necesario para responder a las acciones que se solicitan en el subsistema, como solicitar una alerta y enviar una notificación a través del correo electrónico.

2.6.2 Patrones de diseño.

Un patrón de diseño es una descripción de un problema y su solución, que recibe un nombre y que puede emplearse en otros contextos. También puede ser visto como una pareja de problema/solución con una sugerencia sobre la manera de utilizarlo en situaciones nuevas.

Patrones GRASP (*General Reponsability Assignament Software Pattern*):

En diseño orientado a objetos, GRASP son patrones generales de software para asignación de responsabilidades, aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

- ✓ Patrón controlador: Es el encargado de definir las estructuras para los patrones experto y creador. Tiene como objetivo fundamental administrar diferentes eventos del subsistema, así como definir los métodos para su operación.

  Este patrón se pone de manifiesto en todo el sistema debido a que cada uno de los eventos generados por el usuario son atendidos por el archivo **routing.yml** que es el encargado de redirigir a una clase controladora que realice las operaciones solicitadas, por ejemplo, la clase **OrionAlertRequestController.php**. Esta tiene la responsabilidad de controlar los eventos asociados a realizar una solicitud de alerta.

  A continuación, se muestra un ejemplo de este patrón en el subsistema de alertas para el buscador Orión:

```
orion_alert_delete_alert_request:
    path:     /delete_alert_request/{id}
    defaults: { _controller: ORIONAlertBundle:OrionAlertRequest:delete }
    methods: [DELETE]

orion_alert_edit_alert_request:
    path:     /edit_alert_request/{id}
    defaults: { _controller: ORIONAlertBundle:OrionAlertRequest:edit }
    methods: [GET]

orion_alert_update_alert_request:
    path:     /update_alert_request/{id}
    defaults: { _controller: ORIONAlertBundle:OrionAlertRequest:update }
    methods: [POST]

orion_alert_list_alert_request:
    path:    /list_alert_request
    defaults: { _controller: ORIONAlertBundle:OrionAlertRequest:list }
    methods: [GET]
```

Ilustración 3: Patrón controlador

- ✓ Patrón bajo acoplamiento: Es la idea de tener las clases lo menos ligadas entre sí, de tal forma que, en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases. Este patrón se ve reflejado en el subsistema ya que en Symfony2 las clases controladoras heredan de la clase **Controller.php** alcanzando de esta manera un bajo acoplamiento.
  A continuación, se muestra un ejemplo de este patrón en el subsistema de alertas para el buscador Orión:

```
class DefaultController extends Controller {

    public function indexAction($name) {

        $message = \Swift_Message::newInstance()->
                setSubject('Hello Email')
                ->setFrom('jgespinosa@uci.cu')
                ->setTo('raleman@estudiantes.uci.cu')
                ->setBody('Este es un correo', 'text/plain');
        var_dump($this->get('mailer')->send($message));


       return $this->render('ORIONAlertBundle:Default:index.html.twig', array('name' => $name));
    }

}
```

Ilustración 4: Patrón bajo acoplamiento

- ✓ Patrón alta cohesión: La información que almacena una clase debe de ser coherente y debe estar (en la medida de lo posible) relacionada con la clase. Se puede observar en el subsistema, puesto que, cada clase controladora maneja solamente las responsabilidades correspondientes a las entidades con las que se relaciona, además para cada vista existe una página controladora encargada de manejar sus solicitudes.

  A continuación, se muestra un ejemplo de este patrón en el subsistema de alertas para el buscador Orión:

```
public function createAction(Request $request) {
    $config = $this->loadConfig();
    $type = $this->getForm();
    $itemClass = $config["entity"];
    $item = new $itemClass();
    $form = $this->createForm($type, $item);

    $form->handleRequest($request);


    if ($form->isValid()) {
        $manager = $this->getManager();
        $manager->create($item);
        $this->get("session")->getFlashBag()->add("success", $this->getCreateSuccesMessage());
        return $this->redirect($this->generateUrl($config['route']['default']));
    } else {
        $this->get("session")->getFlashBag()->add("error", $this->getCreateFailMessage());
    }

    $params = array(
        'form' => $form->createView(),
        'item' => $item,
        'config' => $config,
    );

    return $this->render($config['view']['create'], $params);
}
```

Ilustración 5: Patrón alta cohesión

Patrones GOF (*Gang-Of-Four*):

Estos patrones describen la organización en que los objetos y las clases están estructurados, para interactuar unos con otros.

- ✓ Patrón decorador: añade dinámicamente nuevas responsabilidades a un objeto, proporcionando una alternativa flexible a la herencia para extender la funcionalidad. Provee una alternativa muy flexible para agregar funcionalidad a una clase.

  En Symfony2 este patrón es fácilmente visible, ya que, la vista se separa por niveles, en hasta 3 niveles, una plantilla base y varias plantillas que heredan de esta. Normalmente, la plantilla base es global en toda la aplicación y contiene el código HTML que es común en la mayoría de las páginas.

A continuación, se muestra un ejemplo de este patrón en el subsistema de alertas para el buscador Orión:

```
{% extends "::base.html.twig" %}

{% block body %}

    {% if error %}
        <div>{{ error.message }}</div>
    {% endif %}

    <form action="{{ path('form_login_check') }}" method="post">
        <label for="username">Username:</label>
        <input type="text" id="username" name="_username" value="{{ last_username }}" />

        <label for="password">Password:</label>
        <input type="password" id="password" name="_password" />

        <input type="hidden" name="_target_path" value="" />

        <input type="submit" name="login" />
    </form>

{% endblock %}
```

Ilustración 6: Patrón decorador

2.7 Diagramas de clases.

Los diagramas de clases describen los tipos de objetos de un sistema, así como las relaciones que pueden existir entre ellos (Peñalvo, y otros, 2000). A continuación, se representa el diagrama de clases con estereotipos web correspondiente al subsistema de alertas para el buscador Orión.

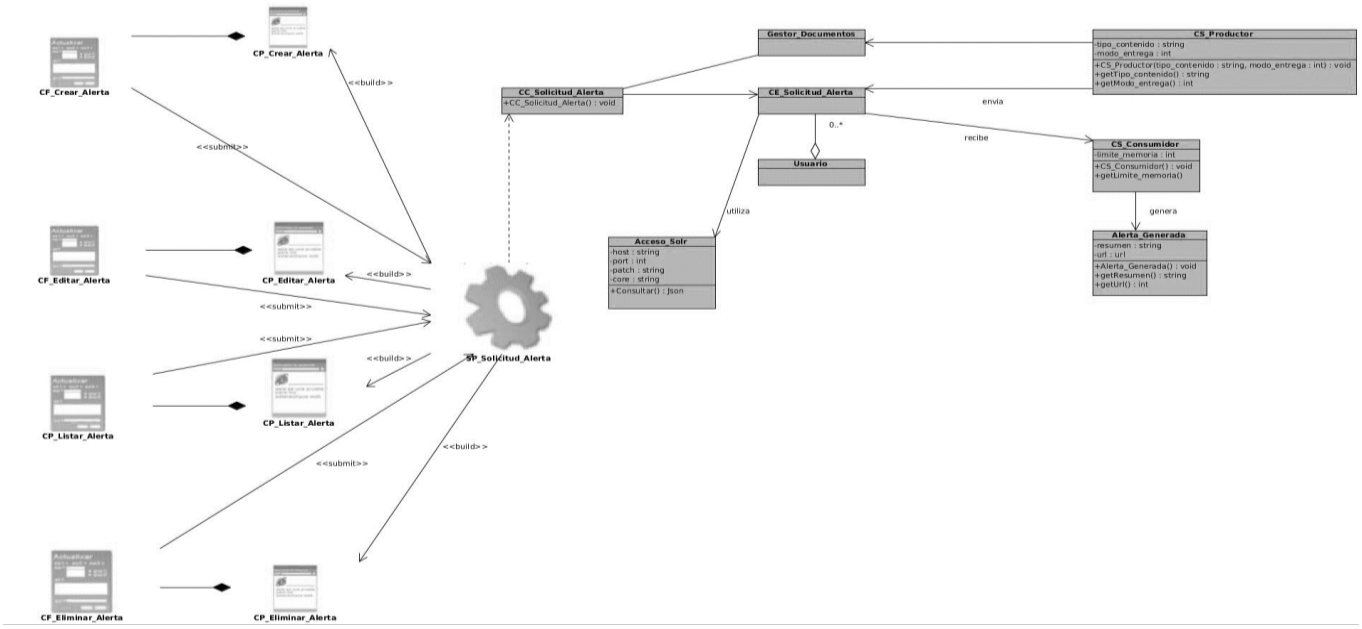

Ilustración 7: Diagrama de clases

2.8 Modelo de datos (físico).

Conjunto de herramientas conceptuales para describir la representación de la información en términos de datos. Los modelos de datos comprenden aspectos relacionados con: estructuras y tipos de datos, operaciones y restricciones (Pantaleón, 2010).

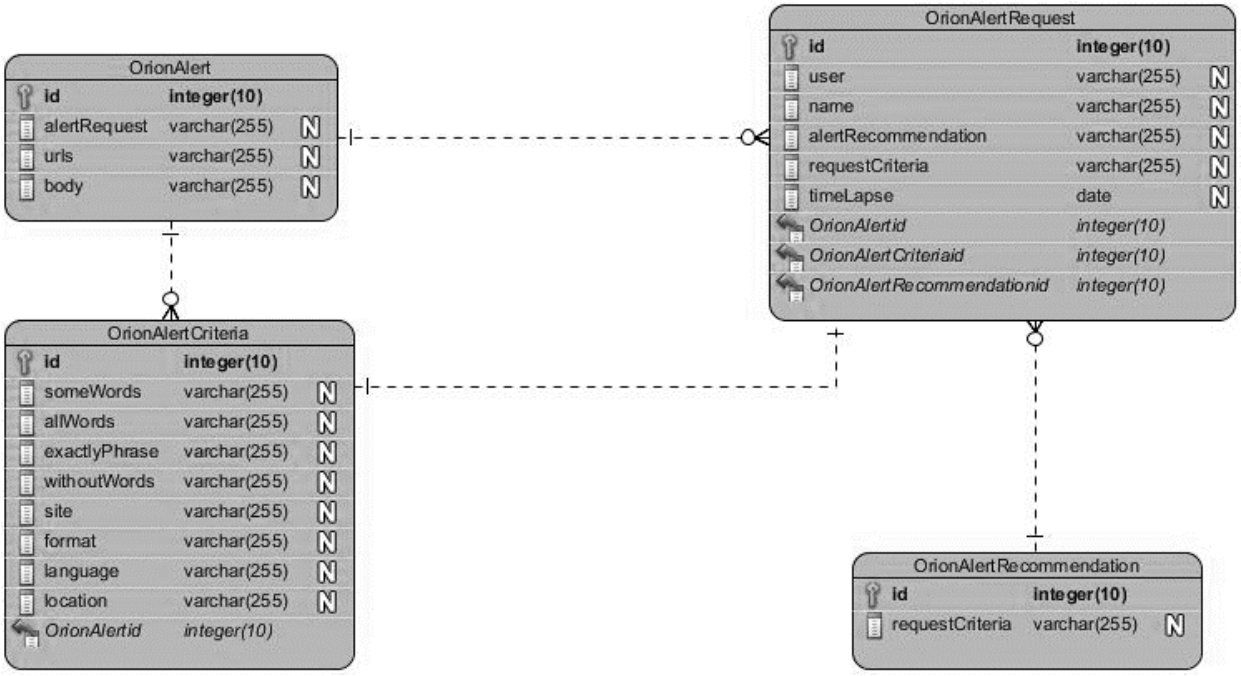

Ilustración 8: Modelo de datos

2.9 Modelo de despliegue.

Consiste en un diagrama estructurado que muestra la arquitectura del sistema desde el punto de vista de distribución de los artefactos del software en los destinos de despliegue (Sarmiento, 2013). A continuación, se muestra el diagrama de despliegue propuesto para el subsistema de alertas.

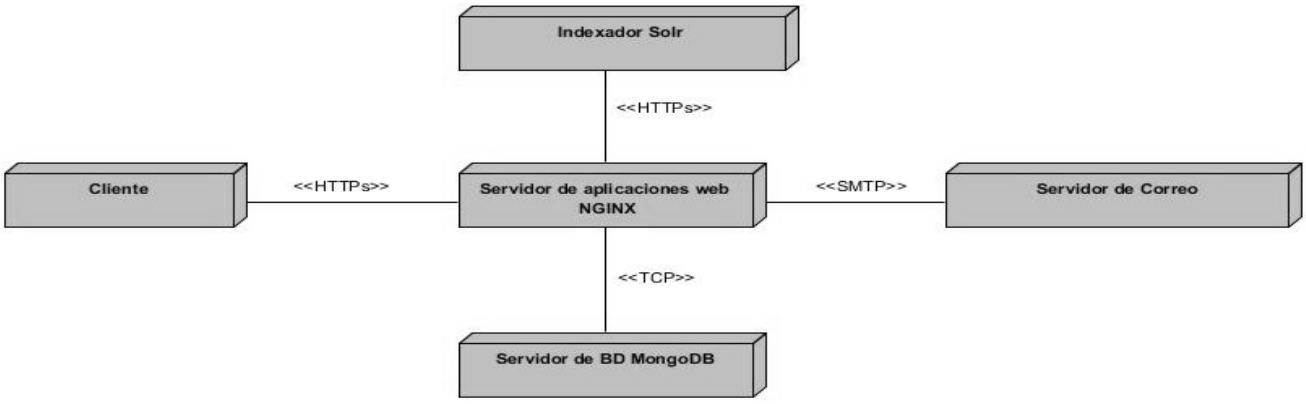

Ilustración 9: Diagrama de despliegue

Para una mejor comprensión del diagrama mostrado, se describen a continuación cada uno de los artefactos que intervienen en el mismo:

- ✓ **Cliente:** Se comunica con el servidor web donde se encuentra funcionando el buscador de la red cubana mediante el protocolo HTTPs.
- ✓ **Servidor de aplicaciones web**: es donde se encuentra el funcionando el buscador. El servidor de aplicaciones web es el que permite que el cliente interactúe y tenga acceso a la aplicación.
- ✓ **Indexador:** es que se encarga de realizar la búsqueda de la información que el usuario desea ser notificado.
- ✓ **Servidor de base de datos:** es donde se guardan los datos correspondientes a las solicitudes realizdas por el usuario.
- ✓ **Servidor de correo:** es la herramienta encargada de recibir las notificaciones que recibirá el usuario.

2.10 Conclusiones Parciales

En el presente capítulo se definieron los elementos de análisis y diseño para el desarrollo del subsistema de alertas , arribando a las siguientes conclusiones:

- ✓ Se identificaron los requerimientos funcionales y no funcionales permitiendo definir las características, condiciones y funcionalidades que debe cumplir del subsistema de alertas.
- ✓ La definición de la arquitectura y los patrones de diseño a utilizar, permitieron establecer las bases para fomentar la reutilización y las buenas prácticas de programación entre los desarrolladores durante la fase de implementación, así como disminuir el impacto de los cambios futuros en el código fuente.
- ✓ La elaboración del diagrama de despliegue permitió identificar la disposición física de los componentes del sistema de alertas.

Capítulo 3. Implementación y pruebas realizadas al subsistema de alertas para el buscador Orión

3.1 Introducción.

El presente capítulo tiene como objetivo describir la etapa de implementación y pruebas, generando los artefactos pertenecientes a las mismas, como es el caso del diagrama de componentes, los estándares de codificación y los casos de prueba. Posteriormente se describen los valores empleados en la ejecución de las pruebas y los resultados obtenidos.

3.2 Diagrama de componentes.

Un diagrama de componentes representa la separación de un sistema de software en componentes físicos, por ejemplo: archivos, cabeceras, módulos y paquetes. Además, muestra las dependencias entre estos componentes (Diestra, 2015).

Como se muestra en la siguiente imagen, los componentes físicos del paquete Aplicación web están divididos en seis subpaquetes: Vistas, Controladores, Formularios, Servicios, Modelos y Configuraciones.

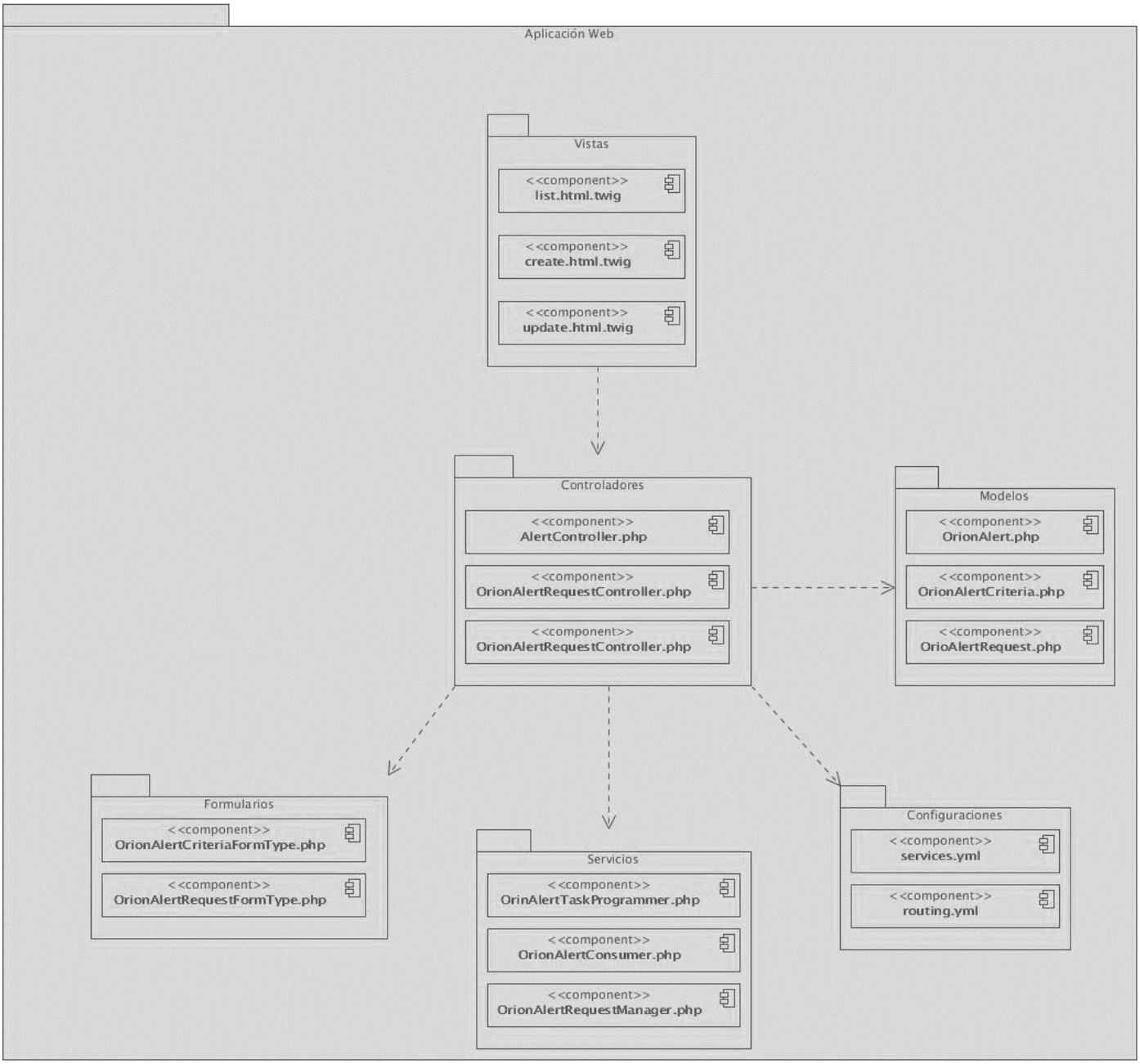

Ilustración 10: Diagrama de componentes para el paquete aplicación web.

- ✓ En el paquete Controladores se incluyen los componentes **DefaultController.php**, **OrionAlertController.php** y **OrionAlertRequestController.php** las cuales se encargan de manejar las acciones que puede hacer un usuario, los formularios y las vistas correspondientes.

- ✓ En el paquete Configuración se encuentran los ficheros de configuración correspondientes a la aplicación web:

  routing.yml: Contiene las rutas que utilizará el subsistema.

  service.yml: Contiene las configuraciones necesarias del subsistema.

- ✓ En el paquete Vistas están presentes las vistas que muestra el subsistema:

  create.html.twig: Muestra el formulario en el que el usuario puede crear una solicitud de alerta.

  update.html.twig: Muestra el formulario en el que el usuario puede modificar una solicitud de alerta.

  list.html.twig: Muestra todas las alertas realizadas por un usuario.

- ✓ En el paquete Modelos se incluyen:

  OrionAlert.php: Contiene las clases que describen o representan los datos de la notificación que será enviada por correo electrónico contenidos en la base de datos.

  OrionRequestAlert.php: Contiene clases que describen o representan los datos de las alertas solicitadas por un usuario.

  OrionCriteriaAlert.php: Contiene clases que describen o representan los criterios de búsqueda para realizar una solicitud de alerta.

3.3 Estándares de codificación.

La forma de escribir código es propia de cada programador y completamente diferente a la forma de cualquier otro. Se definen estándares de codificación porque un estilo de programación homogéneo en un proyecto permite que todos los participantes lo puedan escribir y entender en menos tiempo. Además, permite que el código en consecuencia sea mantenible (Acosta, 2015).

En el caso del subsistema de alertas para el buscador Orión se utiliza el estándar definido por la comunidad de desarrollo de Symfony. A continuación, se muestran algunos ejemplos de los estándares

utilizados.

Estructura.

- ✓ Añade un solo espacio después de cada delimitador coma.

Ejemplo:

```
return $this->render('ORIONAlertBundle:Default:index.html.twig', array('name' => $name));
```

Ilustración 11: Un solo espacio después de cada delimitador coma.

- ✓ Añade un solo espacio alrededor de los operadores.

Ejemplo:

```
$this->criteria = $this->request->getRequestCriteria();
```

Ilustración 12: Un solo espacio alrededor de los operadores.

Convenciones de nomenclatura.

- ✓ Utiliza mayúsculas intercaladas en nombres de variable, función, método o argumentos.

Ejemplo:

```
public function indexAction($name)
```

Ilustración 13: Mayúsculas intercaladas en nombres de variable, función, método o argumentos.

- ✓ Utiliza espacios de nombres para todas las clases.

Ejemplo:

```
namespace ORION\Bundle\AlertBundle\Service;
```

Ilustración 14: Espacios de nombres para todas las clases.

3.4 Pruebas de software.

Las aplicaciones web son creadas, desarrolladas e implementadas por seres humanos y por ende en cualquiera de sus etapas de creación se puede presentar una equivocación. Por tal motivo, se hace necesario luego de concluido un proyecto, verificar que se cumplan con las especificaciones planteadas desde un inicio por el analista o el propio cliente (Paz, 2015). Esto, permitirá eliminar los posibles errores que se hayan cometido en cualquier etapa del desarrollo de software.

Las pruebas de software son básicamente un conjunto de actividades dentro del desarrollo de software y dependiendo del tipo de pruebas, estas actividades podrán ser implementadas en cualquier momento del proceso de desarrollo (Fiestas, 2014).

Para comprobar que la solución implementada es válida y que cumple con los requisitos acordados con el cliente, se propone la siguiente estrategia de pruebas:

Tabla 5: Estrategia de pruebas.

Niveles de prueba	Tipo de prueba	Método	Técnica
Pruebas de integración	Funcional	Caja negra	Incremental
Pruebas de sistema	Rendimiento	Caja negra	Automática
	Funcional	Caja negra	Partición equivalente
	Seguridad	Caja negra	Automática

3.4.1 Pruebas funcionales.

Las pruebas funcionales se basan en los requerimientos entregados para el desarrollo de un software.

Éstas se realizan mediante casos de prueba cuyo fin es validar que el software cumple con el nivel de calidad requerido para entrar en producción (León, 2015).

Para el desarrollo del subsistema de alertas se decidió utilizar la técnica de caja negra, partición equivalente. Esta, divide el campo de entrada en clases de datos que tienden a ejercitar determinadas funciones del software. Se dirige a la definición de casos de prueba que descubran clases de errores, reduciendo así el número total de casos de prueba que hay que desarrollar (Acosta, 2016).

Se definieron casos de pruebas para los casos de usos, a continuación se dan a conocer algunos fragmentos de las pruebas que se realizaron al CU Gestionar alerta.

Tabla 6: Diseño de casos de prueba "Solicitar alerta"

Escenario	Descripción	V1	V2	V3	V4	V5	V6	V7
EC 1.1 Insertar datos de forma correcta.	Mediante este escenario se realiza una solicitud de alerta de forma correcta.	V Cuba	N/A	N/A	N/A	N/A	V .pdf	V español
		V UCI	N/A	N/A	N/A	V https://dragones.uci.cu/	N/A	N/A
EC 1.2 Insertar datos incorrecto	Mediante este escenario se realiza una	N/A	N/A	N/A	N/A	N/A	V .pdf	V español

s.	solicitud de alerta de forma incorrecta	V UCI	N/A	N/A	N/A	I raisa	V .pdf	V español

La tabla mostrada ateriormente evidencia la respuesta correcta del sistema para cada uno de los juegos de datos de entrada. A continuación, se muestran las variables empleadas para el caso de prueba gestionar alerta.

Tabla 7: Variables para el CU "Gestionar alerta"

Nº	Variables	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	V1	Algunas palabras	Campo de texto	No	Debe ser una frase o palabra coherente. (Obligatorio)
2	V2	Todas las palabras	Campo de texto	Si	Debe ser una frase o palabra coherente. (Opcional)
3	V3	Frase exacta	Campo de texto	Si	Debe ser una frase o palabra coherente. (Opcional)
4	V4	Sin palabras	Campo de texto	Si	Debe ser una frase o palabra coherente. (Opcional)

5	V5	Sitio	Campo de texto	Si	Debe ser una dirección de un sitio web válida. (Opcional)
6	V6	Formato	Campo de selección	No	Debe ser seleccionado una de las opciones que ofrece el buscador. (Opcional)
7	V7	Idioma	Campo de selección	No	Debe ser seleccionado una de las opciones que ofrece el buscador. (Opcional)
8	V8	Ubicación	Campo de selección	No	Debe ser seleccionado una de las opciones que ofrece el buscador. (Opcional)
9	V9	Lapso de tiempo	Campo de selección	No	Debe ser seleccionado una de las opciones que ofrece el buscador. (Opcional)

A continuación, se muestra la relación de no conformidades por cada iteración efectuada durante la realización de las pruebas funcionales a los requisitos implementados.

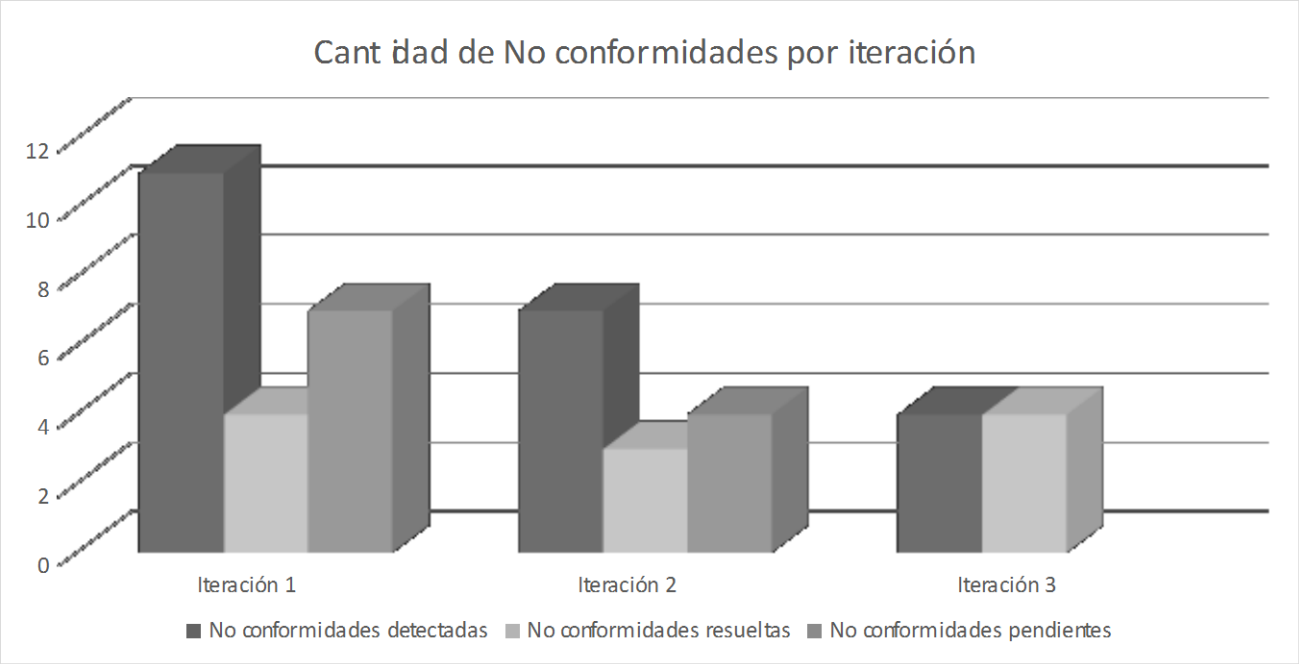

Ilustración 15: Relación de No conformidades por iteración

En la figura mostrada anteriormente se puede observar que se detectaron un total de 11 no conformidades en la primera iteración, de las cuales 4 fueron corregidas, quedaron 7 pendientes ya que se corrigieron solamente las no conformidades correspondientes a funcionalidades no implementadas y mensajes incorrectos. En la segunda iteración se encontraron 7 no conformidades pendientes de la iteración anterior, de las cuales 3 fueron resueltas, atendiendo a que solo se daría solución a las no conformidades referentes a formatos incorrectos. En el caso de la tercera iteración se detectaron 4 no conformidades pendientes de la iteración 2 las cuales fueron resueltas satisfactoriamente.

Como se menciona anteriormente, las no conformidades pueden ser agrupadas atendiendo a diferentes tipos de errores. La cantidad de no conformidades detectadas por cada tipo de error quedan representadas mediante una gráfica (ver anexo 4).

3.4.2 Pruebas de integración.

Las pruebas de integración verifican la interacción entre dos o más programas probados unitariamente, validando que cada componente funciona correctamente cuando actúa en conjunto con los demás (Correa, y otros, 2015).

Una vez realizadas las pruebas funcionales a cada componente interno de manera independiente, se verificó que las funcionalidades implementadas se corresponden de acuerdo a los requisitos funcionales y no funcionales establecidos. De este modo, se pudo comprobar el correcto funcionamiento de los componentes.

Mediante la ejecución de las pruebas de integración se logró verificar la correcta integración de los componentes del subsistema de alertas entre sí y con el motor de búsqueda Orión.

3.4.3 Pruebas de seguridad.

Las pruebas de seguridad consisten en revisar las aplicaciones en búsqueda de vulnerabilidades. Una vez ejecutadas las pruebas de seguridad es posible detectar las vulnerabilidades a las que se ve expuesto el sistema, así como tomar medidas para la disminución de amenazas de ataques (Torets, 2016). Para ello, luego de desarrollada la aplicación se utiliza la herramienta Acunetix Web Vulnerability Scanner 8.0, caracterizada en la sección 1.13.

Mediante estas pruebas, se obtuvieron los siguientes resultados:

Tabla 8: Resultado de las pruebas de seguridad.

Categorías de vulnerabilidades	Cantidad
Formulario HTML sin protección CSRF[23]	16
Transición insegura de HTTP a HTTPS	1

[23] CSRF: falsificación de petición en sitios cruzados, del inglés *Cross-site request forgery*. Es un tipo de *exploit* malicioso de un sitio web en el que comandos no autorizados son transmitidos por un usuario en el cual el sitio web confía.

Credenciales de usuario en texto sin cifrar	9
Habilitado el método *OPTIONS*	1
Total	**27**

Formulario HTML sin protección CSRF: Esta vulnerabilidad permite a un atacante forzar a los usuarios de una aplicación web para ejecutar acciones de la elección del atacante. Un ataque exitoso puede comprometer los datos del usuario final y sus operaciones en el caso de un usuario normal. Si el usuario final apuntado es la cuenta de administrador, esto puede poner en peligro toda la aplicación web (Lázaro, 2015).

Transición insegura de HTTP a HTTPS: Esta vulnerabilidad permite a un atacante reemplazar el destino del formulario.

Credenciales de usuario en texto sin cifrar: Esta vulnerabilidad permite que un tercero pueda ser capaz de leer las credenciales de usuario mediante la interceptación de una conexión HTTP sin cifrar.

Habilitado el método *OPTIONS*: Provee la lista de métodos HTTP soportados por el servidor web, lo cual expone información sensible que puede ayudar a un atacante a realizar ataques avanzados.

Después de analizar los resultados obtenidos en las pruebas, se procedió a corregir las deficiencias encontradas. Para ello se llevaron a cabo un conjunto de acciones en el código fuente de la aplicación, que permitió reforzar la seguridad del subsistema de alertas para el buscador Orión. A continuación, se exponen algunas de las acciones realizadas:

- Inclusión de campos con identificadores únicos en los formularios HTML, para prevenir los ataques CSRF.
- Intercambio de datos sensibles entre cliente y servidor mediante un canal de comunicación encriptado utilizando el protocolo criptográfico SSL.
- Desactivación de las opciones de métodos HTTP en el servidor web.

3.4.4 Pruebas de carga y estrés.

Mediante las pruebas de estrés es posible indentificar la capacidad de respuesta de un sistema bajo condiciones de carga extrema, representadas por una alta concurrencia de usuarios o procesos (Torets, 2016).

Mediante la ejecución de las pruebas de carga es posible identificar la capacidad de recuperación de un sistema cuando es sometido a cargas variables tanto de usuarios como de procesos, determinando el tiempo de respuesta de todas las transacciones críticas del sistema (Torets, 2016).

Para el subsistema de alertas es preciso realizar dichas pruebas, pues resulta necesario comprobar el rendimiento del sistema soportando una cantidad máxima de usuarios que realicen solicitudes de alertas y su comportamiento al aumentar esta carga con los mismos recursos disponibles.

Para la realización de las pruebas de carga y estrés se utilizó la herramienta Apache JMeter en su versión 2.9, caracterizada en la sección 1.13, definiéndose las características del entorno en que fueron realizadas: 1 PC cliente, con un procesador Intel(R) Core(TM) i3-4160 y 4GB de RAM.

Luego de definido el hardware se configuran los parámetros del Apache Jmeter logrando un ambiente de simulación con un total de 500 y 1500 usuarios conectados concurrentemente, para ello se realiza una solicitud de alerta al subsistema.

En la figura que se muestra a continuación se puede observar los resultados que arrojó la herramienta.

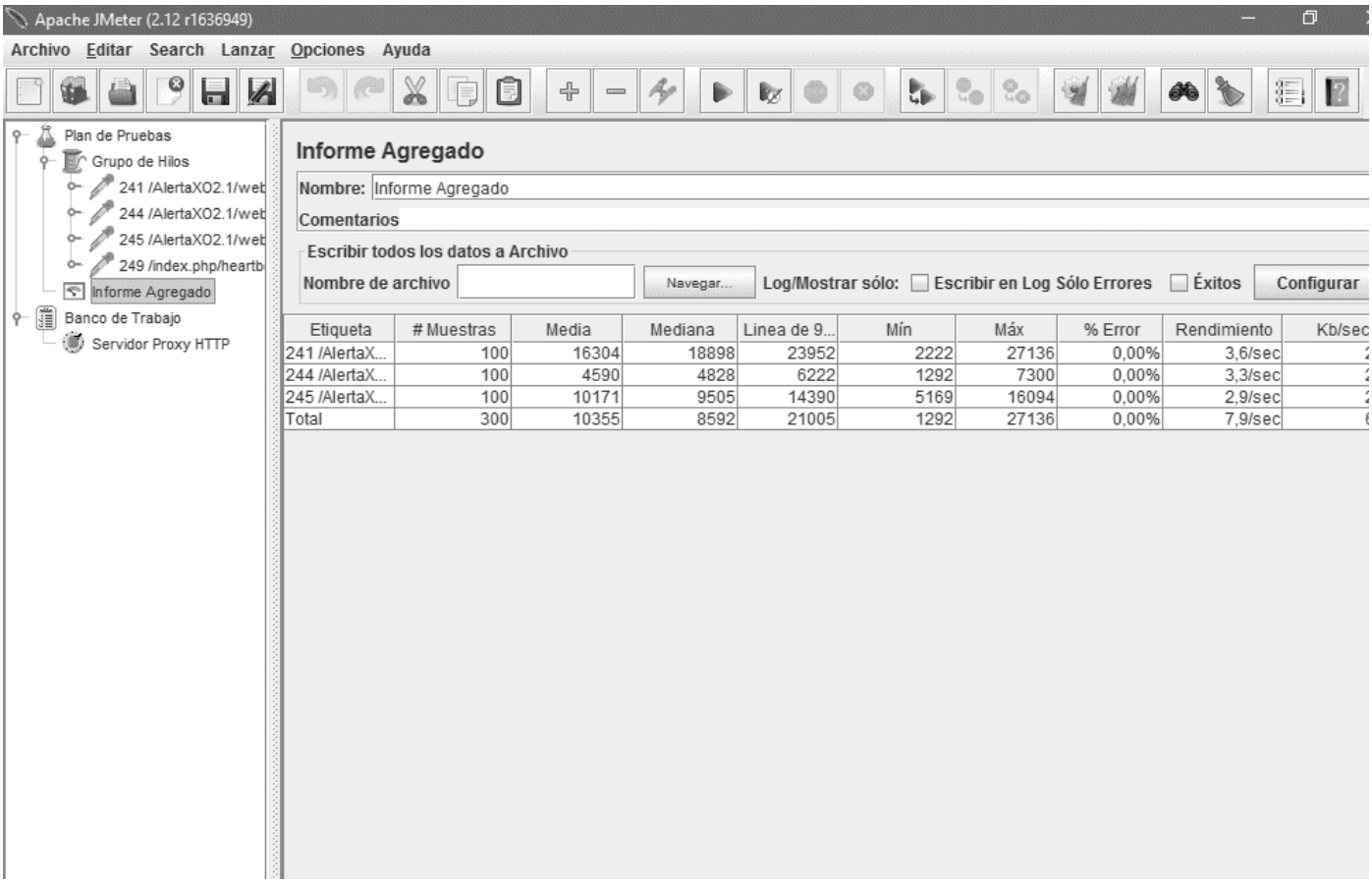

Ilustración 16: Resultados de la prueba de carga y estrés.

Las pruebas realizadas muestran que el sistema es capaz de responder a 300 peticiones de usuarios conectados simultáneamente en un tiempo promedio de 10.355 segundos con un 0.0% de error. Esto demuestra que el subsistema de alertas para el buscador Orión fue capaz de responder correctamente las peticiones realizadas en un tiempo aceptable.

3.5 Conclusiones parciales

En este capítulo se han abordado los elementos correspondientes a la implementación y validación del subsistema de alertas para el buscador Orión, arribando a las siguientes conclusiones:

- ✓ La elaboración de los diagramas de componentes permitió una mejor comprensión de la estructura de los componentes del subsistema implementado.

- ✓ La definición de los estándares de código a utilizar en la implementación de la solución permitió a otros integrantes del equipo de desarrollo entender con mayor facilidad el código desarrollado.
- ✓ Se comprobó el correcto funcionamiento del subsistema de alertas a través de la realización de pruebas de software.

Conclusiones Generales

En el presente trabajo se ha llevado a cabo un proceso de desarrollo de software completo, dividido en flujos de trabajos e iteraciones, con el objetivo de lograr un subsistema de alertas que se integre al motor de búsqueda Orión.

- ✓ Con el estudio de los sistemas de alertas, se definió el marco teórico conceptual de la investigación, que permitió identificar la situación problemática y las bases para analizar, diseñar e implementar el subsistema de alertas para el buscador Orión.
- ✓ El análisis de las diferentes metodologías, herramientas y tecnologías posibilitó la selección adecuada de la base tecnológica para el desarrollo del sistema.
- ✓ Se obtuvieron los artefactos necesarios para el desarrollo del sistema, aplicando AUP como metodología de desarrollo de software y UML como lenguaje de modelado.
- ✓ Se obtuvo un subsistema de alertas que se integra al motor de búsqueda Orión permitiendo notificar a los usuarios a través del correo electrónico cuando nuevos contenidos de su interés aparezcan en la Web.
- ✓ La evaluación de las pruebas de software realizadas permitió corregir los errores detectados en el subsistema desarrollado.

Recomendaciones

- ✓ Incorporar una funcionalidad al subsistema de alertas que permita a un usuario redireccionar las notificaciones a otro correo electrónico.
- ✓ Agregar al subsistema de alertas una funcionalidad que permita a los usuarios realizar solicitudes de alertas sin autenticación.

Referencias Bibliográficas

Acosta, García Vidal Orlando. 2016. *Fabulari, aplicación web de videoconferencia para la Plataforma Web Interactiva del Sistema de Transmisión de Canales Virtuales.* Cuba : Ediciones Futuro, 2016.

Acosta, Juan Carlos. 2015. Estándares de codificación para el desarrollo de software. [En línea] 09 de agosto de 2015. [Citado el: 28 de mayo de 2016.] http://documents.mx/documents/estandar-de-codificacion-para-el-desarrollo-de-softwaredocx.html.

Alcalde, Alejandro. 2012. *Los 11 Mejores Frameworks gratuitos para Aplicaciones Web.* Granada : s.n., 2012.

Aquino, Franck. 2015. Metodología AUP. [En línea] 6 de agosto de 2015. [Citado el: 8 de febrero de 2016.] http://documents.mx/documents/metodologia-aup.html.

Araujo, Luis. 2015. *Comunicación asíncrona.* 2015.

Arenas, Jairo. 2015. Ventajas y desventajas de los lenguajes de programación web. [En línea] 11 de agosto de 2015. [Citado el: 28 de octubre de 2016.] http://documents.mx/documents/ventajas-y-desventajas-de-los-lenguajes-d-eprogramacion-web.html.

Balkhi, Syed. 2013. *What is: Apache.* 2013.

Bolton, David. 2015. Why I Choose PostgreSQL Over MySQL/MariaDB. [En línea] 19 de marzo de 2015. [Citado el: 15 de abril de 2016.] http://insights.dice.com/2015/03/19/why-i-choose-postgresql-over-mysqlmariadb/.

Campechano, Aura. 2015. Uso de herramientas CASE para el desarrollo de bases de datos. [En línea] abril de 2015. [Citado el: 10 de abril de 2016.]

Castro, Luis. 2016. ¿Qué es un Applet en Java? [En línea] 2016. [Citado el: 15 de junio de 2016.] http://aprenderinternet.about.com/od/Glosario/g/Applet-En-Java.htm.

Cherencio, R. Guillermo. 2011. *Trabajando con Librerías Estáticas en el Entorno de Desarrollo Geany.* 2011.

Colimba, Emma. 2013. *Avances Tecnológicos Informáticos.* 2013.

Correa, Eduardo, Arteaga, Diana y Mendoza, Mauricio . 2015. Pruebas de integración. [En línea] 2015. [Citado el: 20 de mayo de 2016.] http://www.findingsqa.com.co/es/node/20.

Díaz, Sabdiel Batista. 2015. *Presentan el (otro más) primer buscador cubano en internet.* Cuba : s.n., 2015.

Diestra, Carlos Castillo. 2015. *Diagrama de Componentes.* Trujillo : s.n., 2015.

Elizalde, Erika. 2016. Los buscadores más populares de Internet. [En línea] 2016. [Citado el: 22 de octubre de 2015.] http://buscadores.about.com/od/conceptosbasicos/tp/Los-Buscadores-M-As-Populares-De-Internet.htm.

Esaú, Antonio. 2015. Los 10 Frameworks PHP que solicitan las empresas. [En línea] 28 de septiembre de 2015. [Citado el: 12 de abril de 2016.] https://openwebinars.net/los-10-mejores-frameworks-php-que-solicitan-las-empresas/.

Esteffan, Gabriel. 2015. Los 5 mejores servicios de Yahoo! a 20 años de su debut. [En línea] 02 de marzo de 2015. [Citado el: 03 de noviembre de 2015.] http://www.t13.cl/noticia/entretencion/tecnologia/los-5-mejores-servicios-de-yahoo-a-20-anos-de-su-debut.

Eucaris, Yurilay. 2015. Metodología RUP. [En línea] 19 de octubre de 2015. [Citado el: 10 de noviembre de 2015.] http://documents.tips/documents/metodologia-ruppdf.html.

Flaquet, David Diaz. 2016. Nuevo buscador Cubano. [En línea] 2016. [Citado el: 10 de junio de 2016.] http://tutoriales.cubava.cu/2015/07/14/nuevo-buscador-cubano/.

Gómez, Nestor. 2015. HTML - CSS - XML - JavaScript: Los principales estándares web del lado del cliente. [En línea] 13 de septiembre de 2015. [Citado el: 10 de abril de 2016.] http://www.pixelesybytes.com/2015/09/html-css-xml-javascript.html.

Gómez, Rodrigo. 2016. *Metodologías para el desarrollo ágil del software.* 2016.

Hernández, Lionel R. Baquero. 2016. *Diagramas de Casos de Uso.* 2016.

Johansson, Lovisa. 2015. Part 1: RabbitMQ for beginners - What is RabbitMQ? [En línea] 18 de mayo de 2015. [Citado el: 17 de junio de 2016.] https://www.cloudamqp.com/blog/2015-05-18-part1-rabbitmq-for-beginners-what-is-rabbitmq.html.

Lázaro, Diego. 2015. Ataques CSRF: Cross-Site Request Forgery en PHP. [En línea] 13 de noviembre de 2015. [Citado el: 25 de mayo de 2016.] https://diego.com.es/ataques-csrf-cross-site-request-forgery-en-php.

León, Marcela. 2015. ¿Por qué son importantes las pruebas funcionales? [En línea] 30 de septiembre de 2015. [Citado el: 10 de mayo de 2016.] http://www.kibernum.com/noticias/por-que-son-importantes-las-pruebas-funcionales-2/.

Lorite, Juan Francisco Adame. 2015. Apache Kafka. [En línea] 15 de julio de 2015. [Citado el: mayo de 17 de 2016.] https://bytesandchips.net/2015/07/15/apache-kafka/.

Martínez, Miguel Angel Muñoz. 2014. *Servicios Google.* España : s.n., 2014.

Mena, David. 2016. Monitoriza tu presencia online con las alertas de Google. [En línea] 2016. [Citado el: 24 de octubre de 2015.] http://wanaleads.com/monitoriza-tu-presencia-online-con-las-alertas-de-google/.

Morán, José Luis Osorio. 2016. Metodología ágil de desarrollo de software. [En línea] 21 de enero de 2016. [Citado el: 18 de junio de 2016.] https://www.clubensayos.com/Tecnolog%C3%ADa/Metodolog%C3%ADa-%C3%A1gil-de-desarrollo-de-software/3101157.html.

Munguía, Enrique. 2015. ¿Qué lenguaje de programación debo elegir? [En línea] 07 de diciembre de 2015. [Citado el: 08 de marzo de 2016.] http://www.enrique7mc.com/2015/12/que-lenguaje-de-programacion-debo-elegir/.

Mur, Fontanals Jesús. 2015. *Mis 5 Frameworks CSS preferidos.* 2015.

Olvera, María Dolores. 2015. *Sistemas de recuperación de información en la Web: Nuevos servicios de búsqueda y recuperación.* Granada : s.n., 2015.

Ortega, Michael. 2014. Acunetix Web Vulnerability Scanner. [En línea] 18 de noviembre de 2014. [Citado el: 10 de junio de 2016.] https://hakin9.org/acunetix-web-vulnerability-scanner/.

Pacheco, María Isabel. 2015. Herramientas CASE para el proceso de desarrollo de sofware. [En línea] 2015 de julio de 2015. [Citado el: 15 de marzo de 2016.] http://documents.mx/documents/herramientas-case-para-el-proceso-de-desarrollo-de-software.html.

Pastor, José Juan Castelló. 2016. *Motores de búsqueda y derechos de autor: infracción y responsabilidad.* s.l. : Aranzadi, 2016.

Paz, Julián Andrés Mera. 2015. La Importancia del proceso de pruebas de Calidad de Software en la Formación de los Ingenieros de Sistemas. [En línea] 21 de octubre de 2015. [Citado el: 21 de mayo de 2016.] http://www.ucc.edu.co/prensa/2015/Paginas/la-importancia-del-proceso-de-pruebas-de-calidad-de-software-en-la-formacion-de-los-ingenieros-de-sistemas.aspx.

Pérez, Gustavo. 2016. ¿Qué lenguaje de programación aprender? [En línea] marzo de 18 de 2016. [Citado el: 10 de junio de 2016.] http://noticias.universia.es/ciencia-tecnologia/noticia/2016/03/18/1137432/lenguaje-programacion-aprender.html.

Pinto, Maria. 2015. Búsqueda y Recuperación de Información. [En línea] 13 de diciembre de 2015. [Citado el: 5 de abril de 2016.] http://www.mariapinto.es/e-coms/busqueda-y-recuperacion-de-informacion/.

Quintero, Jose. 2012. *Eclipse como IDE para PHP.* 2012.

Ramírez, Luz. 2012. *Sistemas Gestores de Bases de Datos.* 2012.

Requena, Enny. 2014. Motores de busqueda. [En línea] 16 de noviembre de 2014. [Citado el: 03 de noviembre de 2015.] http://motordebusquedas.com/2014/11/buscador-yahoo.html.

Rivera, Navarrete Ricardo. 2016. *Zend Framework 2 vs Symfony2.* Perú : s.n., 2016.

Rodríguez, Alex. 2016. Ajax: ejemplos efectos y uso en desarrollos web con HTML, CSS… Ventajas e inconvenientes. [En línea] 2016. [Citado el: 10 de abril de 2016.] http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=920:ajax-ejemplos-

efectos-y-uso-en-desarrollos-web-con-html-css-ventajas-e-inconvenientes-cu01205f&catid=83:tutorial-basico-programador-web-ajax-desde-cero&Itemid=212.

Rodríguez, Javier Jesús Gutiérrez. 2015. *Diagramas UML de casos de uso y de requisitos.* Sevilla : s.n., 2015.

Roldán, Carlos santana. 2013. *¿Qué es Python? .* Estados Unidos : s.n., 2013.

Rosa, Ramos Manuel. 2015. *Los 10 mejores frameworks de PHP para el 2015.* 2015.

Rouse, Margaret. 2016. integrated development environment (IDE). [En línea] 2016. [Citado el: 8 de abril de 2016.] http://searchsoftwarequality.techtarget.com/definition/integrated-development-environment.

Ruiz, Alfredo. 2014. ¿Que es HTML y para que sirve? [En línea] 2014. [Citado el: 28 de marzo de 2016.] http://www.acercadehtml.com/manual-html/que-es-html.html.

Salas, Carlos. 2013. Internet Information Server (IIS) 8. [En línea] 25 de junio de 2013. [Citado el: 15 de mayo de 2016.] http://www.expertosensistemas.com/internet-information-server-iis-8/.

Silva, Nerio. 2015. *Unified Modeling Language .* 2015.

Sironi, Giorgio. 2013. *Apache Solr: una introducción .* 2013.

Steven, Perry J. 2012. *Introducción a la programación Java, parte 1: Conceptos básicos del lenguaje Java.* 2012.

Suárez, Luis. 2016. Buscadores Web. [En línea] 2016. [Citado el: 26 de mayo de 2016.] http://buscadores-web.com/ranking-buscadores-internet/.

Torets, Jose Manuel López. 2016. V&V Quality. [En línea] 2016. [Citado el: 18 de mayo de 2016.] http://vyvquality.com/pruebas-seguridad/.

Tusa, Stalin. 2012. *Servidor de aplicaciones.* 2012.

Uñoja, Roger Humberto. 2015. *Metodologías de desarrollo tradicionales vs ágiles.* 2015.

Vaquero, Miguel. 2014. ¿Qué es un Applet? Lenguaje de programación Java. [En línea] 2014. [Citado el: 28 de octubre de 2016.] http://www.deciencias.net/simulaciones/paginas/appletjava.htm.

Vereau, César. 2015. *AngularJS vs BackboneJS, Jquery, ReactJS y otros.* 2015.

Vicente, Jose María Toribio. 2014. *Pruebas de Rendimiento y Funcionales Web.* 2014.

Villamil, Juan. 2015. Metodologías de desarrollo de sofware. [En línea] 4 de noviembre de 2015. [Citado el: 9 de diciembre de 2015.] http://jupaviro.blogspot.com/2015/11/metodologias-de-desarrollo-de-software.html.

Wajser, Damián. 2015. Combate entre servidores web: NGINX vs Apache. [En línea] 18 de agosto de 2015. [Citado el: 18 de mayo de 2016.] http://latamdigital.softtek.co/combate-entre-servidores-web-nginx-vs-apache.

Yofu, Juan David Cañas. 2015. *El objetivo de los buscadores.* 2015.