Universidad de las Ciencias Informáticas

Facultad 1

Título: “Sistema de monitoreo y análisis estadístico V 2.0”

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

Autor: Alfredo Luis Santisteban Céspedes.

Tutores: Lic. Katiuska Cedeño Espinosa.

Ing. Frank Emilio Hernández Sánchez.

La Habana

“Año 57 de la Revolución”

Declaración de autoría

Declaro que soy el único autor del presente trabajo y autorizo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de este, con carácter exclusivo.

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

Alfredo Luis Santisteban Céspedes

_____

Firma del autor

Frank Emilio Hernández Sánchez

_____

Firma del tutor

Katiuska Cedeño Espinosa

_____

Firma del tutor

Agradecimientos

Le agradezco especialmente a mi mamá por darme siempre su amor y su apoyo incondicional, por estar siempre presente para mí, por sus consejos, por sus chistes que siempre me alegran la vida, por enseñarme a luchar por lo que quiero y por soportarme, que a veces es una tarea difícil.

A mi papá, por todo su amor, por hablarme con firmeza cuando hizo falta, por preocuparse tanto y por sus buenos consejos como padre y como amigo.

A mi prima Lucre, que ha sido como otra madre más, siempre corriendo y luchando conmigo.

A mi abuelita María, por ser tan especial, a mi tía Adriana por quererme tanto y consentirme, a mi tía Gladys, a mi hermanita linda Rosibel, a mi hermano Leo, mi tía Mariela, mi abuela Mirian, mi primo Robert, a Elizabeth y al resto de la familia.

A mis amigos: Siré, Manuel Alejandro, José Luis, Adrián, Mario, Leonardo e Iraldo, gracias por estar presentes en las buenas y en las malas, por sus buenos y no tan buenos consejos.

A todos mis colegas y conocidos.

A mis tutores Katy y Frank, por tenerme tanta paciencia, por aconsejarme y apoyarme.

Gracias a todos por formar parte de mi vida.

Alfredo Luis Santisteban Céspedes

Dedicatoria

Le dedico este trabajo a mi mamá por ser mi ejemplo en la vida.

Te AMO mami.

Resumen

En la actualidad, la existencia de sistemas de monitoreo y análisis posibilita realizar estudios de tendencia que garantizan una mejor toma de decisiones de gobiernos, empresas e instituciones. En la Universidad de las Ciencias Informáticas fue creada una herramienta “SIMAE” versión 1.0, que realiza este proceso para sitios *web*; sin embargo presenta algunas deficiencias que obstaculizan la obtención y el procesamiento de datos. Por esta razón se propone la creación de una nueva versión de dicha herramienta, utilizando como *framework Symfony* 2.5, lenguaje de programación en el servidor (PHP) y en el cliente (*JQuery*), *PostgreSQL* 9.2 como gestor base de datos y *OpenUP* como metodología para guiar el proceso de desarrollo de *software*. Este sistema posibilitará la recopilación de datos de forma manual y automática; además una gestión avanzada de los sitios *web* y las fuentes de información correspondientes a las publicaciones. Tendrá en cuenta, en la recopilación de datos, elementos como el idioma y las imágenes. Facilitará la realización de reportes y gráficas que a admitan el entrecruzamiento de información. Permitirá el envío de notificaciones mediante correo electrónico a directivos o responsables de un área en cuestión, así como hacer una adecuada gestión de las trazas de los usuarios en el sistema. Obteniendo como resultado final una solución que permitirá mayor eficiencia en los procesos de monitoreo y análisis de información. Validándose la calidad del sistema a partir de la aplicación de pruebas funcionales, de seguridad y de carga y estrés.

Palabras clave: análisis de información, análisis estadístico, recopilación de datos, sistema de monitoreo.

Índice

Declaración de autoría I
Agradecimientos II
Dedicatoria III
Resumen IV
Índice de tablas VII
Índice de ilustraciones VIII
Introducción 1
Capítulo 1: Fundamentación Teórica 6
1.1 Marco teórico conceptual 6
1.1.1 Información 6
1.1.2 Análisis de información 7
1.1.3 Gestión de la información 7
1.1.4 Sistema de gestión de información 8
1.1.5 Publicaciones *web* 9
1.1.6 Gestores bibliográficos 9
1.2 Herramientas homólogas 10
1.3 *Framework* 15
1.4 Lenguajes de programación y de marcado 17
1.4.1 PHP 17
1.4.2 HTML 18
1.4.3 CSS 19
1.4.4 *JQuery* 19
1.5 Entorno de desarrollo integrado 19
1.6 Sistema gestor de base de datos 21
1.7 Lenguaje unificado de modelado 22
1.8 Herramienta de modelado CASE 22
1.9 Metodología de desarrollo 23
1.10 Servidor *Web* 24

1.11 Bibliotecas para generar gráficos 25
1.12 Bibliotecas para exportar a PDF 26
1.13 Conclusiones parciales 27
Capítulo 2: Análisis y diseño del sistema 28
2.1 Modelo del dominio 28
2.2 Requisitos de *software* 29
2.2.1 Requisitos funcionales 30
2.2.2 Requisitos no funcionales 32
2.3 Modelo de casos de uso del sistema 33
2.4 Patrón arquitectónico Modelo-Vista-Controlador (MVC) 40
2.5 Patrones de diseño 41
2.6 Modelo de datos 46
2.7 Diagrama de clases de diseño 47
2.8 Modelo de despliegue 49
2.9 Conclusiones parciales 49
Capítulo 3: Implementación y prueba 51
3.1 Modelo de implementación 51
3.1.1 Diagrama de componentes 51
3.1.2 Estándares de codificación 52
3.2 Pruebas de *software* 56
3.2.1 Pruebas funcionales 56
3.2.2 Pruebas de seguridad 58
3.2.3 Pruebas de carga y estrés 59
3.3 Conclusiones parciales 61
Conclusiones generales 63
Recomendaciones 64
Referencias bibliográficas 65
Bibliografía 69
Glosario de términos 71

Índice de tablas

Tabla 1 Especificación de requisitos funcionales 30
Tabla 2 Descripción de los actores del sistema 35
Tabla 3 Descripción CU: gestionar publicación manual 36

Índice de ilustraciones

Ilustración 1 Diagrama de modelo del dominio 29
Ilustración 2 Diagrama de CU del sistema 35
Ilustración 3 Patrón MVC en symfony 40
Ilustración 4 Patrón experto 42
Ilustración 5 Patrón alta cohesión 43
Ilustración 6 Patrón controlador 44
Ilustración 7 Patrón controlador 44
Ilustración 8 Patrón singleton 45
Ilustración 9 Patrón decorator 45
Ilustración 10 Patrón unidad de trabajo 46
Ilustración 11 Diseño de la base de datos 47
Ilustración 12 DCD CU 4 Gestionar publicación manual 48
Ilustración 13 Diagrama de despliegue 49
Ilustración 14 Diagrama de componentes CU 4 Gestionar publicación manual 52
Ilustración 15 Identación, llaves de apertura y cierre, y tamaño de las líneas 53
Ilustración 16 Convención de nomenclatura para variable 53
Ilustración 17 Convención de nomenclatura para clase simple 53
Ilustración 18 Convención de nomenclatura para clase compuesta 54
Ilustración 19 Convención de nomenclatura para función 54
Ilustración 20 Convención de nomenclatura para fichero 54
Ilustración 21 Estructuras de control 55
Ilustración 22 Documentación de clase 55
Ilustración 23 Documentación de método 55
Ilustración 24 Gráfico estadístico de las no conformidades obtenidas en las pruebas funcionales 58
Ilustración 25 Gráfico estadístico de las vulnerabilidades obtenidas en las pruebas de seguridad 58
Ilustración 26 Resultado de la aplicación JMeter para 50 usuarios 60
Ilustración 27 Resultado de la aplicación JMeter para 100 usuarios 61

Introducción

Con el desarrollo de la sociedad moderna el manejo de información ha llegado a ser un factor importante para el progreso y éxito de cualquier entidad, organismo o empresa. Toda la información que se almacena constituye un recurso clave para dichas organizaciones. De ahí que uno de los desafíos en la actualidad sea la gestión de la misma, así como la garantía de que se realice de forma sencilla y eficiente.

El análisis de sitios *web* que permita la evaluación de indicadores sensibles como usabilidad, rendimiento y navegación, deviene como otro de los retos de los sistemas informáticos de hoy día, pues posibilita materializar dichos indicadores en reportes que constituyen una pieza fundamental para apoyar la toma de decisiones acertadas en cualquier organismo o entidad.

Actualmente existen herramientas informáticas que realizan un análisis estadístico de los sitios *web*, estas arrojan datos como: enlaces, velocidad de carga, palabras clave, posicionamiento en buscadores y visitas en tiempo real. Dichos datos permiten formular recomendaciones para mejorar el funcionamiento y calidad de los sitios *web*, permitiendo además realizar un estudio detallado de algunas de las tendencias que circulan en la red.

Las publicaciones, ya sean libros, revistas, periódicos o boletines digitales, constituyen un porciento considerable de la información mostrada en la mayoría de los sitios *web*, las que, independientemente del tema que aborden, pueden generar gran influencia en los individuos que las consulten, alcanzando un impacto social importante. De ahí la necesidad de la extracción de algunos de sus elementos, para posterior análisis e interpretación.

La Revolución Cubana por su desempeño a escala global, por la política interna y externa que ha manifestado desde sus inicios y por su conflicto histórico con el gobierno norteamericano, ha sido punto de mira de la opinión pública a nivel internacional. A ello va aparejado el interés del país por conocer las tendencias a nivel nacional e internacional de temas relacionados con tecnología, estudios científicos, salud, educación, sociales, intercambio económico y de cooperación entre estados, así como otros que resulten importantes para su desarrollo como nación; además de que circulan diariamente por la red un sin fin de publicaciones relacionadas con Cuba. Por tales motivos se hace necesaria la existencia de un sistema que permita conocer dichas tendencias a través del monitoreo y análisis posterior de publicaciones, para posibilitar la extracción de diferentes datos como son: título, vínculo, sitios, redactores, cantidad de visitas y

comentarios, fuentes, áreas y temáticas correspondientes a la publicación, así como la fecha y el texto del artículo; los cuales brindarán los elementos necesarios para hacer una correcta interpretación, por lo que podrán ser utilizados para apoyar la toma de decisiones.

Internacionalmente existen herramientas que realizan análisis y monitoreo de sitios *web*, con diferentes fines, como *Woorank* que se utiliza para analizar aspectos relacionados con el tráfico y nivel de calidad de un sitio al igual que *Google Analytics* que realiza funciones similares, pero de forma gratuita o *twXplorer* que permite realizar análisis de las principales tendencias que circulan en la red social *Twitter*. A nivel nacional está el Sistema para el Análisis de la Información Mediática Internacional (SAIMI) el cual tiene como objetivo monitorear y analizar los medios de prensa internacionales, para apoyar la toma de decisiones políticas del país.

Actualmente en la Universidad de las Ciencias Informáticas (UCI) hay dos sistemas que realizan el monitoreo y análisis de sitios *web*: el Sistema de Almacenamiento, Clasificación y Análisis de Noticias (SACAN) versión 1.0 cuyo objetivo es monitorear, recopilar y analizar las noticias que son publicadas en *internet* y el Sistema de Monitoreo y Análisis Estadístico (SIMAE) versión 1.0 que permite almacenar la información referente a las publicaciones de sitios *web* y *blogs* internos de la UCI, para realizar un estudio de tendencias sobre los principales temas tratados a través de la red universitaria.

El departamento de Ingeniería Social Universitaria (ISU), perteneciente al Centro de Ideoinformática (CIDI), es el encargado de realizar esos análisis haciendo uso de la herramienta SIMAE, en la cual la recopilación de información de las publicaciones se realiza de forma manual, llegando a ser un trabajo engorroso dada la cantidad de información que se maneja. No se puede hacer una gestión adecuada para los sitios que se quieran monitorear, así como de las fuentes que son utilizadas en las publicaciones. Además existen ciertos parámetros necesarios como el idioma y las imágenes que no se pueden obtener. El diseño y la arquitectura del sistema no están en correspondencia con las necesidades actuales del departamento. A lo que se suma que no se consigue obtener algunos reportes que son importantes para el análisis a través del entrecruzamiento de los diferentes parámetros que se recopilan. Existen problemas para gestionar el seguimiento de las publicaciones. También es necesario que se le informe mediante correo electrónico a los directivos o responsables de un área en cuestión cuando se realice una alerta.

Por otra parte se tiene dificultad para manipular algunos parámetros de administración como: configurar la autenticación según se quiera, ya sea local y/o utilización de LDAP, así como poder configurar los diferentes

parámetros de esta última; poder configurar a través de una interfaz la conexión con las bases de datos que utiliza el sistema. Otra de las problemáticas actuales está referida a la adecuada gestión de las trazas de los usuarios en el sistema.

Por tanto se plantea como **problema de investigación**: ¿Cómo mejorar el proceso de recopilación, análisis de información de las publicaciones y la administración del sistema informático SIMAE v1.0?

Para dar solución al problema en cuestión se define como **objeto de estudio** los sistemas de análisis de información. Restringiendo el **campo de acción** a los sistemas de análisis de información en sitios *web*.

Para dar respuesta al problema a resolver se plantea como **objetivo general**: desarrollar un sistema en su segunda versión que permita mejorar el proceso de recopilación, análisis de información de las publicaciones y la administración del sistema informático SIMAE v1.0.

Se formula la siguiente **idea a defender**: la creación de un sistema que mejore el proceso de recopilación, análisis de información de las publicaciones y la administración del sistema informático SIMAE v1.0, permitirá una mayor eficiencia en el trabajo que realiza el departamento ISU.

Tareas de la investigación:

1. Caracterización de los procesos relacionados con la recopilación y análisis de información de publicaciones en sitios *web*.
2. Identificación de las distintas soluciones existentes que realicen el proceso de recopilación y análisis de información de publicaciones en sitios *web*.
3. Comparación de las distintas soluciones estudiadas para obtener las bases necesarias para mejorar la herramienta SIMAE.
4. Identificación de las funcionalidades que serán mejoradas en el sistema SIMAE v2.0.
5. Definición de la metodología, tecnologías, herramientas y lenguajes a utilizar en el proceso de desarrollo de *software*.
6. Realización del análisis, diseño e implementación de las mejoras al sistema SIMAE v2.0.
7. Realización de pruebas para validar el sistema.

Posibles resultados: una herramienta informática capaz de hacer eficiente el proceso de recopilación y análisis estadístico de información, así como la documentación técnica correspondiente al desarrollo del sistema.

Métodos de investigación científica

Con el fin de resolver y dar cumplimiento al objetivo general y las tareas propuestas se utilizaron los siguientes métodos de investigación:

Métodos teóricos

- Analítico - sintético: en la investigación se utilizó este método para realizar un análisis teórico e identificar los principales conceptos a incluir en la fundamentación teórica, permitiendo extraer elementos importantes relacionados con el objeto de estudio. Además posibilitó identificar las tecnologías, herramientas, lenguajes y la metodología a utilizar en el desarrollo de la solución propuesta.
- Modelación: este método sirvió de apoyo en la realización de los artefactos correspondientes al análisis, diseño e implementación de las funcionalidades propuestas.

Métodos empíricos

- Entrevista: este método se evidenció en la entrevista realizada a las partes involucradas en los procesos fundamentales del MINREX, a fin de obtener información detallada que permitiera el entendimiento general de los procesos (ver anexo 1).

El presente trabajo se encuentra estructurado de la siguiente forma: resumen, introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía, glosario de términos y anexos.

Capítulo 1. Fundamentación teórica: se incluye el estado del arte del tema tratado en la presente investigación y se establece una comparación entre metodologías, tecnologías, herramientas y lenguajes para definir las que serán utilizadas en el desarrollo de la solución propuesta.

Capítulo 2. Análisis y diseño del sistema: en este capítulo se especifican los artefactos elaborados que forman parte de la metodología de desarrollo de *software* propuesta. Se realiza el modelado del negocio

con el objetivo de esclarecer el contexto de la solución. Se describen los requisitos funcionales y no funcionales del sistema, los casos de uso y se realiza el diseño del sistema.

Capítulo 3. Implementación y prueba: en este capítulo se realiza la implementación del sistema cumpliendo con los planes trazados en el análisis y diseño, además se realizaron las pruebas funcionales, de seguridad y de carga y estrés sobre la aplicación para validar su funcionamiento.

Capítulo 1: Fundamentación Teórica

Este capítulo incluye un estudio sobre varias definiciones relacionadas con el proceso de recopilación y análisis de información de sitios *web* que permite un mejor entendimiento del negocio. Se hace una investigación sobre herramientas que realizan monitoreo y análisis estadístico de sitios *web* cuyas características pudieran servir de aporte para el desarrollo de la solución propuesta. Se realiza la descripción y comparación de distintas tecnologías, metodologías, herramientas y lenguajes de programación, marcado y modelado para identificar los que serán utilizados en el presente trabajo.

1.1 Marco teórico conceptual

1.1.1 Información

Uno de los términos utilizado a diario para conocer acerca de sucesos, lugares, personas, trabajos, es el de información, ya sea para aumentar los conocimientos acerca de algo o para reducir el nivel de incertidumbre cuando se quiere tomar una decisión. Dado su uso constante es fundamental conocer su definición.

Algunos autores o fuentes en *internet* proponen varias definiciones:

1. Información: es un conjunto de datos con un significado, o sea, que reduce la incertidumbre o que aumenta el conocimiento de algo. En verdad, la información es un mensaje con significado en un determinado contexto, disponible para uso inmediato y que proporciona orientación a las acciones por el hecho de reducir el margen de incertidumbre con respecto a las decisiones (1).

2. Información: comprende los datos y conocimientos que se usan en la toma de decisiones (2).

3. Información: consiste en datos seleccionados y ordenados con un propósito específico (3).

4. El Diccionario de la Real Academia Española, la define de las siguientes maneras: (Del lat. informatĭo, -ōnis). 1. f. Acción y efecto de informar. 2. f. Oficina donde se informa sobre algo. 3. f. Averiguación jurídica y legal de un hecho o delito. 4. f. Pruebas que se hacen de la calidad y circunstancias necesarias en una persona para un empleo u honor. U. m. en pl. 5. f. Comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada. 6. f. Conocimientos así comunicados o adquiridos (4).

A partir del análisis de las definiciones propuestas anteriormente se plantea la siguiente definición:

La información es un conjunto de datos, clasificados y ordenados que poseen un significado y permiten ampliar los conocimientos y disminuir el grado de incertidumbre para la toma de decisiones, pero su abundancia implica que no toda sea correcta, por tal motivo es necesario constatar que la misma sea de calidad, así se garantiza que las decisiones que se tomen a partir de ella sean acertadas.

1.1.2 Análisis de información

El análisis de información es una forma de investigación, cuyo objetivo es la captación, evaluación, selección y síntesis de los mensajes subyacentes en el contenido de los documentos, a partir del análisis de sus significados, a la luz de un problema determinado. Así, contribuye a apoyar la toma de decisiones, al cambio en el curso de las acciones y de las estrategias. Es el instrumento por excelencia de la gestión de la información (5).

Adquiere una relevancia extraordinaria, pues su realización exitosa y eficiente genera una mejor utilización del conocimiento disponible en aras de acelerar el proceso de su implementación. Confluye en el propósito de crear vías para hacer llegar la información al usuario que la requiere. Posibilita la descripción y representación del documento, a partir de la reproducción en síntesis del documento real, utiliza taxonomías para procesar la información y vocabularios que recrean el entorno lingüístico, repercute en la calidad de los productos y servicios de información, eleva la capacidad de recuperación y reduce la incertidumbre (5).

El análisis de información se centra en el análisis de contenido en un contexto específico, se remite directamente al autor, posibilita la recuperación de la información. Está condicionada por la calificación, inteligencia, creatividad y conocimientos del tema que posea el analista, que posibilite el uso, relación y manejo de conceptos, así como la habilidad para ubicar en un contexto y establecer los nexos necesarios entre la información procesada y el conocimiento disponible para la solución de un problema. Nace como respuesta a la necesidad de una metodología científica para tratar rigurosamente los documentos y la información existente en diferentes contextos (5).

1.1.3 Gestión de la información

La gestión de la información en el presente siglo se impone como una actividad sumamente importante de la ciencia de la información. Se trata de una revolución basada en ella, porque los avances tecnológicos actuales permiten procesarla, almacenarla, recuperarla y comunicarla en cualquiera de sus formas (voz, textos, imágenes) sin importar la distancia, el tiempo o volumen.

Esta no es más que el proceso de organizar, evaluar, presentar, comparar los datos en un determinado contexto, controlando su calidad, de manera que sea veraz, oportuna, significativa, exacta, útil y que esté disponible en el momento que se le necesite. Se encamina al manejo de la información, documentos, metodologías, informes, publicaciones, soportes y flujos en función de los objetivos estratégicos de una organización (6). Dentro de sus principales procesos están: la identificación de las necesidades de información, la adquisición de las fuentes informativas, su organización y almacenamiento, el desarrollo de productos y servicios, su distribución y uso, que son también la base de la creación del conocimiento durante la existencia productiva de la organización (7).

El desarrollo acelerado de la tecnología, acompañado de la renovadora industria del *software* y la incorporación de coherentes sistemas para la gestión de información, proponen novedosas soluciones para potenciar valores a los denominados recursos intangibles, mejorar estrategias de administración y elevar niveles de eficiencia y eficacia, pues en el contexto actual parece indiscutible que el éxito de una empresa no dependa únicamente de cómo se manejen los activos materiales, sino también de la gestión de la información.

1.1.4 Sistema de gestión de información

La importancia que tiene el recurso información es tal que las organizaciones deben ser consideradas como sistemas de información. Es frecuente confundirlo con la tecnología que lo soporta. Estas tecnologías han supuesto una auténtica revolución en la capacidad de manejo de sus recursos, permitiendo un rápido y eficiente proceso de adquisición, enriquecimiento y acceso a la misma, aunque nunca hay que olvidar que un sistema de gestión de información (SGI) va más allá de las propias herramientas utilizadas.

Estos SGI pueden definirse como un conjunto de funciones o componentes interrelacionados que forman un todo, obtienen, procesan, almacenan y distribuyen la información, manipulando los datos y consiguiendo, para una organización o empresa, la búsqueda de mejores vías para la dirección y control correspondiente de sus procesos, apoyando la toma de las decisiones en el desempeño de las funciones, de acuerdo a sus estrategias propias (8).

La creciente producción de información y el volumen que puede alcanzar en los SGI trae consigo que se pierda tiempo y esfuerzo en su búsqueda y análisis, lo que puede provocar demoras y errores en la toma de decisiones. Es por ello que muchos de los sistemas actuales analizan la información mediante el uso de herramientas o módulos estadísticos que agilizan el proceso.

1.1.5 Publicaciones *web*

Anteriormente la información estaba contenida en documentos impresos en bibliotecas y en un número variable de copias, pero con el aumento de la cantidad de información que exige un espacio cada vez mayor para su almacenamiento y la necesidad de una difusión más rápida como resultado del llamado proceso de globalización o internacionalización del conocimiento, la tendencia actual es la sustitución de las publicaciones impresas por las de formato electrónico, a pesar de que el avance que están sufriendo las tecnologías de la información y las comunicaciones provocan un incremento notable en la distribución de la información que en ocasiones es de forma incontrolable, sin una estructura informática diseñada específicamente para estos fines (9).

Las publicaciones *web* contienen los elementos básicos de sus predecesoras las publicaciones impresas como la presentación, estructura y organización de la información. Sin embargo con la utilización de las nuevas tecnologías para el almacenamiento y tratamiento de la información, se puede obtener un producto superior en cuanto a calidad y que cumpla con sus funciones de manera efectiva.

Existen diferentes formas de publicaciones electrónicas como: las revistas digitales, periódicos y boletines electrónicos, libros y colecciones de ellas llamadas comúnmente bibliotecas electrónicas, las cuales brindan varias ventajas:

- La preservación de documentos raros y frágiles sin prohibir el acceso a quienes deseen consultarlos.
- Fácil transmisión mediante redes telemáticas.
- Acceso de muchos usuarios simultáneamente.
- Solución al problema de espacio físico para el almacenamiento.
- Los costos de edición y de distribución se reducen considerablemente al utilizar los medios electrónicos para la transmisión de la información.

1.1.6 Gestores bibliográficos

Los gestores bibliográficos son herramientas que recogen las referencias bibliográficas de las bases de datos de investigación (catálogos de bibliotecas, índices, bases de datos de revistas científicas), y permiten organizar las citas y la bibliografía para un proyecto. Además ayudan en la realización de trabajos de investigación permitiendo la creación automática de citas, bibliografía y notas al pie (10).

Estos contribuyen a simplificar sustancialmente las dificultades presentes en el desarrollo de actividades académicas e investigativas, dadas por el desconocimiento por parte de los usuarios de las normas y herramientas existentes para realizar las citas empleadas en la elaboración de trabajos escritos, las imprecisiones al citar documentos consultados o la acumulación desorganizada de documentos descargados de bases de datos.

Sus funciones principales son la recolección, edición y organización de referencias bibliográficas, así como importar y exportar dichas referencias. También pueden crear citas y bibliografías automáticamente desde un procesador de texto y realizar búsquedas en bases de datos científicas desde su propia interfaz.

Algunas de estas aplicaciones son: *EndNoteWeb* que es gestor de referencias integrado en *Web of Science* de ISI, *Mendeley* que es un gestor de referencias gratuito, de escritorio y *web*, que permite el trabajo académico colaborativo en línea y *Zotero* que es una extensión libre para *firefox*, permite recolectar, administrar y citar trabajos de investigación de todo tipo de orígenes del navegador, y además organizar documentos en colecciones y subcolecciones.

1.2 Herramientas homólogas

Este epígrafe muestra un estudio de herramientas homólogas al sistema en desarrollo, que realizan análisis estadísticos de sitios *web*, gestión de publicaciones y análisis de tendencias en redes sociales. De estas se analizarán algunas de sus principales funciones para determinar cuáles realizan mayor aporte al presente trabajo y las principales desventajas que representa su utilización.

Woorank

Se trata de un método novedoso en línea para analizar y evaluar diferentes aspectos relacionados con el tráfico y el nivel de un sitio *web*. Es una herramienta multifacética ideal para cualquier *webmaster*. El sistema ofrece referencias sobre más de cincuenta criterios diferentes de cada página solicitada. Todos los datos que arroja representan información útil para mantener un sitio *web* en el mejor estado. Además, el lenguaje que presenta es básico permitiendo que cualquier tipo de sujeto con mínimos conocimientos informáticos pueda utilizarlo sin dificultades. Entre las funciones más importantes que presenta se destacan las siguientes (11):

- Análisis estimativo -con gran nivel de aproximación a la realidad- sobre el tráfico de un sitio *web* o una página de cualquier tipo.

- Medición general del espacio en *Alexa.*
- Cálculo sobre el índice de páginas indexadas.
- Permite determinar las palabras clave que más se utilizaron en un sitio *web.*

Ventajas de *Woorank* frente a otros servicios de analítica *web* (11):

- Revisión del sitio *web*: posee una interfaz gráfica muy sencilla y muestra los datos de forma intuitiva.
- Integración de datos para un análisis completo: permite integrar datos de cuentas de *Google Analytics* y *Facebook*, con tal de proveer gráficas para analizar el progreso del negocio.

Esta herramienta presenta la desventaja de que el servicio no es gratuito solo permite un análisis gratis de prueba inicial, por lo que no es viable para la Universidad, además no responde a las necesidades del departamento ISU debido a que está más centrada en realizar un análisis sobre indicadores de funcionamiento y calidad de un sitio *web*. Sin embargo algunas de las funcionalidades que realiza sirvieron de base en el desarrollo de la nueva versión de SIMAE como la generación de reportes y gráficas que permite determinar cuáles datos serán mostrados, así como la posibilidad de entrecruzarlos. También se tuvo en cuenta el análisis de palabras clave que realiza, el cual facilita la identificación de tendencias en la información recopilada.

Google Analytics

Es una herramienta de *google* que ofrece un servicio de analítica digital. Es una de las soluciones más eficaces y presenta la gran ventaja de ser gratuita. Permite realizar un análisis de visitas en tiempo real, información de procedencia, comportamiento de los visitantes a través de mapas de calor, contenido más popular, variaciones en gustos, influencia de medios sociales, estadísticas de conversiones, múltiples usuarios, reportes en PDF. Entre sus principales funciones se encuentran (12):

- Herramientas de análisis: se basa en una plataforma de informes, potente y fácil de usar que permite decidir qué datos desea ver y personalizar.
- Analítica de contenido: muestra a través de los informes de contenido los resultados de los análisis realizados a sitios *web*.
- Representación gráfica: genera informes que poseen distintos recursos de representaciones gráficas para reflejar los valores de las tablas.

Una de sus principales desventajas radica en que los datos obtenidos del análisis se almacenan en servidores de *google* y aunque ofrezca una política de confidencialidad estricta para la protección de dichos datos, es más recomendable guardarlos en un servidor propio. Además realiza muestreo en grandes cantidades de datos para procesar una menor información y aunque para datos genéricos no sea un problema, a la hora de extraer datos concretos pueden existir variaciones significativas en los resultados obtenidos. A partir de estas desventajas y que la herramienta no satisface las necesidades existentes en el departamento ISU, se determinó que no es factible su utilización. No obstante algunas funcionalidades que realiza como la representación gráfica y la generación de informes sobre los datos obtenidos en el análisis aportaron ideas para el desarrollo del nuevo sistema.

twXplorer

Es una herramienta gratuita a la cual se puede acceder registrándose con una cuenta de *twitter*. Es útil para identificar tendencias, ya que trabaja con un buscador de términos que permite identificar las últimas 500 menciones a esa palabra. Los resultados muestran los *retweets* recientes, enlaces, *hashtags*, usuarios, entre otros elementos.

Permite realizar consultas de búsquedas en un idioma determinado sobre algún término específico, muestra los resultados en un tiempo relativamente rápido e incluye además gráficos de barras para mostrar las palabras más populares que aparecen. Brinda también los *hashtags* y los enlaces más populares que contengan un término específico. Otra manera de realizar búsquedas es a través de listas que ha creado *twitter* en correspondencia con el término que se ha introducido en la herramienta. Puede además guardar un conjunto de resultados que se muestran por fecha y hora y se agrupan por consultas o por listas en dependencia de como obtuvo los resultados (13).

Sistema para el Análisis de la Información Mediática Internacional

El Sistema para el Análisis de la Información Mediática Internacional (SAIMI) versión 1.0 fue desarrollado por el Ministerio de Relaciones Exteriores (MINREX) con el objetivo de realizar monitoreo y análisis de noticias de los principales medios de prensa a nivel internacional; sin embargo el trabajo con esta herramienta está mayormente enfocado al proceso de análisis de información que apoya la toma de decisiones políticas del país. Permite evaluar la cobertura que tienen los medios de prensa internacionales

sobre el tema Cuba u otro tema de especial seguimiento. Realizan monitoreo de *blogs* y televisoras a través de sus sitios *web*, pues también marcan tendencias desde el punto de vista político.

Almacena todas las publicaciones recopiladas, ya sea de Cuba u otro tema, lo que permite establecer comparaciones entre la cobertura que le dieron distintos medios a un tema en diferentes años; permite realizar entrecruzamiento entre todos los parámetros que se recopilan: fuente, medio, autor, los nomencladores que gestiona al sistema. Se generan 3 o 4 partes diarios, uno semanal, mensual, trimestral, semestral y anual sobre la información en los medios de prensa internacionales referente a Cuba u otro tema específico.

Sin embargo no se recopilan los comentarios procedentes de las publicaciones, enfocándose mayormente en las líneas editoriales de los medios internacionales. No se monitorea los medios de prensa nacionales. No permite configurar cómo se generan los reportes ni establecer un orden entre las publicaciones recopiladas. Utiliza RSS pero solo como apoyo para guiar la búsqueda. Además la recopilación de información se realiza de forma manual.

Se puede acceder mediante los roles de monitoreador: inserta publicaciones procedentes de medios de prensa internacionales, todo referente al tema Cuba o temas de prioridad para el MINREX; el analista: clasifica las publicaciones insertadas por el monitoreador, además puede eliminar las publicaciones que no serán clasificadas, genera los partes, las gráficas, las tablas estadísticas y los informes de análisis; el administrador: además de realizar las funciones del monitoreador y el analista, gestiona los distintos nomencladores del sistema, gestiona los usuarios y consulta las trazas generadas; y el usuario: que accede al sistema solamente para consultar las noticias.

SAIMI es un sistema bastante completo con respecto a los procesos de monitoreo y análisis de sitios *web*, aunque no realiza algunas funcionalidades que son necesarias para el departamento ISU como la gestión de los comentarios, del seguimiento de las publicaciones o de las alertas. Sin embargo no se puede tener acceso ni a la herramienta ni al código fuente, por lo que solo se estudiaron sus funcionalidades para determinar cuáles podrían ser utilizadas en el desarrollo de la solución propuesta como la gestión de los nomencladores y el entrecruzamiento de los parámetros en los reportes y gráficas generadas.

Sistema de Almacenamiento, Clasificación y Análisis de Noticias

El Sistema de Almacenamiento, Clasificación y Análisis de Noticias (SACAN) versión 1.0 fue desarrollado por el Departamento de Soluciones Informáticas para *Internet* (SINI) perteneciente a la UCI con el objetivo de monitorear, recopilar y analizar las noticias que son publicadas en *internet*. Este sistema es actualmente utilizado por el Departamento de Operaciones *Web* y Análisis de Información (DOWAI), la mesa redonda y la UJC Nacional.

Entre sus principales funcionalidades está la clasificación y gestión de noticias, la gestión de alertas en el sistema, gestión de comentarios, realizar búsquedas avanzadas, imprimir y exportar graficas sobre análisis estadísticos generados. Permite también generar boletines de noticias en formato PDF y reportes de noticias en hojas de cálculo, así como el envío y recepción de mensajes a los usuarios. Además se pueden gestionar varios elementos como: temáticas, autores, fuentes y tipos de fuentes, descriptores, idioma, usuarios y permisos.

Cuenta con una interfaz gráfica amigable y sencilla para el usuario, es ejecutable tanto en *windows* como *linux*. Fue desarrollado con PHP5 haciendo uso de la herramienta *NetBeans* en su versión 6.8. El *framework* utilizado para su desarrollo fue *Symfony* 1.

Sin embargo el departamento ISU presenta varias necesidades que SACAN no satisface como: la realización de un estudio más detallado de las tendencias que circulan en la red, debido a que las estadísticas que se obtienen de él son lineales pues no permite el entrecruzamiento de varios parámetros para su posterior análisis, además no gestiona los comentarios que vienen con las publicaciones, ni los sitios *web* de donde provienen dichas publicaciones. Tampoco permite obtener las noticias automáticamente, este proceso debe realizarse de forma manual. Por tales motivos no es factible utilizar este sistema en el departamento.

Sistema de Monitoreo y Análisis Estadístico

El Sistema de Monitoreo y Análisis Estadístico (SIMAE) versión 1.0 fue desarrollado por CIDI con el objetivo de monitorear y analizar las publicaciones en los sitios *web* y *blogs* internos de la UCI. El sistema es actualmente utilizado por el departamento de ingeniería social universitaria para realizar un estudio de las diferentes tendencias que circulan en la red universitaria. Permite realizar de forma manual la gestión de las publicaciones que son recopiladas, así como de los comentarios, sitios *web* y las fuentes correspondientes a las publicaciones. Posibilita darle seguimiento a las publicaciones, gestionar las alertas que se generan

sobre publicaciones o comentarios, generar reportes, gráficas y realizar análisis estadístico sobre la información recopilada. Por otra parte gestiona las personas relacionadas con las publicaciones, las que pueden ser redactores o los directivos a los que se les envían las alertas, además de gestionar los nomencladores, los usuarios que acceden al sistema y las trazas.

Este sistema no satisface totalmente las necesidades actuales que tiene el departamento ISU, por tal razón se propuso la creación de una segunda versión en la cual se tendrá en cuenta algunas de las funcionalidades que realizaba la versión anterior como la gestión manual de las publicaciones y la gestión de los comentarios y además se mejoraron funcionalidades como la gestión de los sitios *web*, de las fuentes, los redactores, los directivos y los nomencladores. A partir de la nueva versión los reportes (boletines, gráficas y análisis estadísticos) permitirán entrecruzar los distintos parámetros recopilados y se podrá gestionar correctamente las trazas generadas.

1.3 *Framework*

Un *framework* orientado a objeto es una arquitectura de *software* reusable, que provee tanto código como diseño. Este representa una aplicación parcial de *software*, tanto en diseño como en código, para solucionar problemas de un dominio dado. Se compone de un conjunto de clases y de flujos de control y de esta manera, brinda una estructura precisa para definir nuevas aplicaciones que resuelven problemas dentro de un dominio dado (14).

Symfony 2.5

Es un *framework* diseñado para optimizar el desarrollo de las aplicaciones *web* debido a sus características. Separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación *web*. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación *web* compleja. Automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación (15).

Está desarrollado completamente con PHP 5. Es compatible con gestores de bases de datos como *MySQL*, *PostgreSQL*, *Oracle* y *SQL Server* de *Microsoft*. Es estable, productivo y documentado, difundido en la actualidad para la construcción de aplicaciones *web*. Es multiplataforma (15).

Symfony 2.5 ha sido ideado para exprimir al límite todas las nuevas características de PHP 5.3 y por eso es uno de los *frameworks* PHP con mejor rendimiento. Su arquitectura interna está completamente

desacoplada, lo que permite reemplazar o eliminar fácilmente aquellas partes que no encajan en un proyecto (16).

Symfony 2.5 es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones. Dentro de sus principales características están (17):

- Emplea el tradicional patrón modelo – vista – controlador (MVC) para separar las distintas partes que forman una aplicación *web*.
- Escalable: es infinitamente escalable si se dispone de los recursos necesarios.
- Probado: ha sido probado con éxito durante varios años en aplicaciones de gran escala, así como en sitios medianos y pequeños.
- Soporte: *symfony* 2.5 sigue una política de tipo LTS (*long term support* o soporte a largo plazo). Las versiones estables se mantienen durante 3 años sin cambios pero con una continua corrección de los errores conocidos.
- Licencia: utiliza una licencia MIT (*Massachusetts Institute of Technology* o Instituto de Tecnología de Massachusetts), con la que se pueden hacer aplicaciones *web* comerciales, gratuitas y/o de *software* libre.
- Código: desde su primera versión ha sido creado para PHP 5, desechando la versión PHP 4.
- Seguro: se puede controlar hasta el último acceso a la información e incluye por defecto protección contra ataques XSS y CSRF.
- Documentado: se trata del *framework* PHP mejor documentado, ya que se han publicado varios libros y además, están completamente traducidos al español.
- Calidad: su código fuente incluye más de 8.000 pruebas unitarias y funcionales.
- Internacionalización: se pueden crear aplicaciones en varios idiomas. La internacionalización está integrada en el *framework*, sigue los estándares XLIFF, es completa.

CodeIgniter

Es un *framework* para desarrollo de aplicaciones, un conjunto de herramientas para construir sitios *web* usando PHP. Su objetivo es permitir desarrollar proyectos mucho más rápido que lo que podría hacer si escribiera el código desde cero, proveyendo un rico conjunto de bibliotecas para tareas comunes, así como una interfaz sencilla y una estructura lógica para acceder a esas bibliotecas. Permite minimizar la cantidad de código necesaria para una tarea dada (18).

Se encuentra bajo una licencia open source *Apache*/BSD-*style*, por lo que puede ser utilizado libremente. Sigue el modelo de la estructura MVC, lo que permite una buena separación entre la lógica y presentación. Algunas de sus características más atractivas son la rapidez en el tiempo de ejecución, dispone de un conjunto bastante amplio de bibliotecas, además de una interfaz gráfica simple y una estructura lógica sencilla que permite acceder a esas bibliotecas. Posee un entorne sencillo y su núcleo solo necesita de algunas de estas bibliotecas para funcionar correctamente. Brinda buena seguridad y filtro de XSS. Sin embargo no contiene un sistema de plantillas, ni tiene una plantilla en general. Además de que no presenta módulos y no hay un mapeo de objetos a bases de datos (ORM por sus siglas en inglés *Object Relational Mapping*).

Selección de *framework* a utilizar

Después de analizar las diferentes características de estos *framework*s se llega a la conclusión de emplear *Symfony* 2.5 ya que proporciona una gran cantidad de herramientas que simplifican el desarrollo de aplicaciones *web*, con conocimientos básicos de la programación moderna como: la programación orientada a objetos (OOP por sus siglas en inglés *Object Oriented Programming*), el ORM y el desarrollo rápido de aplicaciones (RAD por sus siglas en inglés *Rapid Application Development*) se puede entender fácilmente, además el conjunto de características que posee, antes descritas, facilitan el proceso de desarrollo de una aplicación con calidad.

1.4 Lenguajes de programación y de marcado

1.4.1 PHP

PHP (*Hypertext Preprocessor*) es un lenguaje de programación interpretado que inicialmente se diseñó para la creación de páginas *web* dinámicas. Se usa principalmente en interpretación del lado del servidor. Es de código abierto e independiente de plataforma, con una gran biblioteca de funciones, rápido y con mucha documentación.

Ventajas:

- Permite la interacción con muchos motores de bases de datos como: *MySQL*, *Oracle* y *PostgreSQL*.
- Es de código abierto, soportado además por una gran comunidad de desarrolladores lo que lo hace cada vez más potente.
- Biblioteca nativa de funciones amplia e incluida.

- Es un lenguaje multiplataforma.
- PHP generalmente es utilizado como módulo de *Apache*, lo que lo hace veloz.
- Posee una amplia documentación en su página *web* oficial.
- Permite las técnicas de la programación orientada a objetos.
- Es capaz de leer y manipular datos desde diversas fuentes, incluyendo datos ingresados desde formularios HTML.

La versión PHP 5 incluye un conjunto de mejoras como en el soporte para la OOP con PHP *Data Object*, mejoras en el rendimiento, mejor soporte para *MySQL* con extensión completamente reescrita. También mejora el soporte XML (*XPath*, DOM), así como soporte nativo para *SQLite* e integrado para SOAP (*Simple Object Access Protocol*). Además destaca por características como la seguridad ya que provee diferentes niveles de la misma y que pueden ser configurados fácilmente. Cuenta con una estabilidad impresionante gracias a la gran comunidad de desarrolladores y usuarios que posee, pues utiliza su propio sistema de administración de recursos y dispone de un sofisticado método de manejo de variables (19).

1.4.2 HTML

El lenguaje de marcación HTML (*Hypertext Markup Language*) surge como una propuesta para crear la estructura básica de páginas *web*, organizar su contenido y compartir información. HTML 5 no es simplemente una nueva versión del lenguaje de marcación HTML, sino una agrupación de diversas especificaciones concernientes al desarrollo *web*. Es decir, HTML 5 no se limita sólo a crear nuevas etiquetas, atributos y eliminar aquellas marcas que están en desuso o se utilizan inadecuadamente, sino que va mucho más allá: es considerado el producto de la combinación de las tecnologías HTML, CSS y *JavaScript* que actúan como una unidad organizada bajo la especificación de HTML 5, además se presenta con características renovadas como es el caso de la estructura, que se perfecciona estableciendo qué es cada sección, eliminando el uso abusivo de la etiqueta div haciéndola más coherente y fácil de entender (20).

Es el lenguaje de marcado que hace que el proceso de codificación sea más fácil. Muchas de las características permiten a los usuarios ejecutar contenidos complejos en plataformas de baja potencia. Entre ellas se incluyen mejoras en los formularios, la sintaxis <video>, <audio>, integración de contenido de gráficos vectoriales. Estas nuevas características aseguran un buen flujo de contenidos multimedia y diseño gráfico en la *web*, incluso sin ningún tipo de *plugins*.

1.4.3 CSS

CSS son las siglas de *Cascading Style Sheets*, en español hojas de estilo en cascada. Estas poseen grandes ventajas ya que permiten a los desarrolladores *web* controlar el estilo y el formato de múltiples páginas *web* al mismo tiempo, es una tecnología que permite crear páginas *web* de una manera más exacta. Gracias a CSS los desarrolladores se involucran en los resultados finales de la página, pudiendo incluir márgenes, tipos de letra, fondos, colores e incluso definiendo los estilos en un archivo externo; así, si se quiere cambiar alguno de estos, automáticamente se actualizan todas las páginas vinculadas al sitio (21).

CSS 3 no solo pretende reducir el uso de código *JavaScript* y de estandarizar funciones populares, sino que incorpora nuevos mecanismos que además de encargarse del diseño y estilo también lo hacen de la forma y movimiento de los elementos que se muestran en las páginas. Contiene módulos que permiten a la tecnología definir aspectos de funcionalidad dentro de un conjunto, como los selectores de media, colores, svg (Gráficos Vectoriales Redimensionables). Esta modularidad garantiza mayor facilidad y mejoramiento en el manejo de aspectos específicos del CSS que si se realizara una especificación masiva que incluya todas sus características. Además introduce nuevos estilos que pretenden un mejor control sobre la presentación de las modernas interfaces (20).

1.4.4 *JQuery*

JQuery es un marco de trabajo rápido, poderoso y fácil de utilizar que permite a los desarrolladores y diseñadores *web* agregar elementos dinámicos e interactivos a sus sitios. Simplifica la manera de interactuar con los documentos HTML, así como la manipulación del árbol DOM, manejar eventos, desarrollar animaciones y agregar interacción con la técnica *AJAX.* Además cuenta con un gran número de *plugins* que permiten extender sus funcionalidades (22).

1.5 Entorno de desarrollo integrado

Un Entorno de Desarrollo Integrado (IDE) es un conjunto de herramientas para el programador que incluye un buen editor de código, administrador de proyectos y archivos, enlace transparente a compiladores e integración con sistemas controladores de versiones o repositorios.

NetBeans IDE 8.0

Es un entorno de desarrollo integrado de código abierto, distribución gratuita, sin restricciones de uso y apoyado además por una gran comunidad de desarrolladores. Aunque está escrito en *Java* puede ser usado

con otros lenguajes de programación como *Java*, C/C++, *JavaScript*, PHP y *Phyton*. Permite el desarrollo de aplicaciones de escritorio, *web* y para móviles. Es una herramienta para programadores pensada para escribir, compilar, depurar y ejecutar programas (23).

Características (23):

- Incluye el control de versiones y compilación avanzada.
- Es multiplataforma, está disponible para GNU/*Linux*, *Windows*, *Mac OS X* y Solaris.
- Separa el diseño del *software* de la implementación con modelado UML.
- Soporte para *Java*, C, C++, XML, lenguajes HTML, RMI, JSP, CORBA, JINI, JDBC y tecnologías *Servlet*.
- Creación visual de componentes gráficos.
- Dispone de soporte para la creación de interfaces gráficas de forma visual, agregando y alineando el espacio de trabajo.

PhpStorm

Es un IDE muy potente que proporciona un excelente soporte para PHP (incluyendo las últimas versiones de idiomas y marcos), HTML, *JavaScript*, CSS, *Sass, Less* y *CoffeScript*. Su editor de PHP entiende perfectamente su estructura y soporta las versiones 5.3, 5.4, 5.5 y 5.6. Entre otras características proporciona codificación con código de finalización sensible al contexto inteligente, comprobación de errores al instante o mezcla de lenguajes (24).

La nueva versión de *PhpStorm* soporta los CCMMSS Drupal y *WordPress*. Lleva a cabo un análisis de la calidad del código realizando cientos de inspecciones que se encargan de verificar su código a medida que escribe. Tiene un entorno de desarrollo óptimo que permite realizar tareas rutinarias desde el propio IDE, gracias a la integración de sistemas de control de versiones, el apoyo a la implementación remota, bases de datos *SQL*, herramientas de línea de comando y *Composer*. Posee gran experiencia multiplataforma, brindando buen rendimiento y estabilidad ya sea en *Windows*, *Mac OS X* o *Linux* con actualizaciones frecuentes.

Selección del IDE de desarrollo

Analizando las características de ambos IIDDEE se puede concluir que son óptimos, además presentan varias similitudes en cuanto a sus características como el control de versiones, que son multiplataforma, poseen soporte para varios lenguajes de programación. Pero la característica que define su utilización es que *NetBeans* es de código abierto, con distribución gratuita por lo que no posee restricciones de uso.

1.6 Sistema gestor de base de datos

Los sistemas de gestión de bases de datos, son aplicaciones que permiten a los usuarios definir, crear y mantener la base de datos y proporciona un acceso controlado a la misma (25).

MySQL

Es un sistema de gestión de base de datos relacional, multihilos y multiusuario. Se ofrece bajo licencia GNU GPL, es de código abierto, su arquitectura lo hace rápido y fácil de adaptar. Consume pocos recursos y es usado tanto para aplicaciones sencillas como complejas, brinda gran facilidad de instalación y configuración, es multiplataforma. Sin embargo un porcentaje importante de las utilidades de *MySQL* no están documentadas y es poco intuitivo (26).

PostgreSQL

Es un sistema de gestión de base de datos objeto-relacional, distribuido bajo licencia BSD (*berkeley software distribution* o distribución de *software* berkeley) y con su código fuente disponible libremente. Utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema, un fallo en uno de los procesos no afectará el resto. Funciona correctamente con grandes cantidades de datos y una alta concurrencia de usuarios accediendo a la vez al sistema.

Entre las características más importantes y soportadas por *PostgreSQL* está que es una base de datos 100% ACID. Mantiene buena velocidad de respuesta tanto en bases de datos pequeñas como grandes. Ha sido diseñado para tener un mantenimiento y ajuste menor que los productos de proveedores comerciales, conservando todas las características, estabilidad y rendimiento, lo que garantiza ahorros considerables. Soporta los tipos de datos, cláusulas, funciones y comandos de tipo estándar *SQL*92/*SQL*99 y extendidos propios de *PostgreSQL*. Puede operar sobre distintas plataformas, incluyendo *Linux*, *Windows*, *Unix*, Solaris y *Mac OS X*. Presenta buen sistema de seguridad mediante la gestión de usuarios, grupos de usuarios y contraseñas y tiene gran capacidad de almacenamiento. Es capaz de ajustarse al número de CPU y a la

cantidad de memoria disponible de forma óptima, soportando una mayor cantidad de peticiones simultáneas a la base de datos de forma correcta (27).

Selección del gestor de base de datos a utilizar

Se seleccionó *PostgreSQL* v9.4, debido a que es multiplataforma, confiable, estable, con gran escalabilidad, control de concurrencia y funcionalidades que lo destacan como uno de los SGBD más potentes en la actualidad. Además cuenta con abundante soporte en línea pues cuenta con foros oficiales, y la comunidad de usuarios es grande. Es más práctico hacer uso de *PostgreSQL* como gestor de base de datos para la realización de la aplicación.

1.7 Lenguaje unificado de modelado

El lenguaje unificado de modelado (LUM o UML, por sus siglas en inglés, *Unified Modeling Language*) es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Ofrece un modo estándar para describir un "plano" del sistema (modelo), incluyendo aspectos conceptuales tales como procesos de negocio y funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables (28). UML está implícito en la metodología utilizada, además brinda un conjunto de herramientas gráficas que permiten especificar, ilustrar, documentar y construir un sistema de *software* brindando todo lo necesario para modelar la solución propuesta.

1.8 Herramienta de modelado CASE

Son aplicaciones informáticas que tienen como objetivo aumentar la productividad y facilitar la gestión de las fases del proceso de desarrollo de *software*. Permiten mejorar la planificación de un proyecto y la calidad del producto. Además de ahorrar tiempo y costo de desarrollo y mantenimiento de los sistemas informáticos (29).

Visual Paradigm

Como herramienta CASE (*Computer Aided Software Engineering* o Ingeniería de *Software* Asistida por Computadora) se propone Visual Paradigm que es una poderosa herramienta de modelación visual. La misma utiliza UML para el modelado permitiendo la creación de diagramas en un ambiente totalmente visual. Esta herramienta ayuda a una rápida construcción de la aplicación con alta calidad y a un menor costo. Cuenta con excelente documentación (30).

Visual Paradigm ofrece (30):

- Entorno de creación de diagramas para UML.
- Diseño centrado en casos de uso y enfocado al negocio que generan un *software* de mayor calidad.
- Uso de un lenguaje estándar común a todo el equipo de desarrollo que facilita la comunicación.
- Capacidades de ingeniería directa e inversa.
- Modelo y código que permanecen sincronizados en todo el ciclo de desarrollo.
- Disponibilidad de múltiples versiones, para cada necesidad.
- Disponibilidad de integrarse en los principales IIDDEE.
- Disponibilidad en múltiples plataformas.

1.9 Metodología de desarrollo

OpenUP

El Proceso Unificado Abierto (*OpenUP*) es un proceso unificado y ágil que involucra un conjunto mínimo de prácticas que ayudan a los equipos de trabajo a ser más efectivos en el desarrollo de sistemas de *software* u otros sistemas de ingeniería (31).

Adopta un enfoque pragmático y su filosofía ágil se centra en la naturaleza colaborativa de desarrollo de *software*. Es una metodología agnóstica, ejecuta procesos de baja formalidad que pueden ser usados tal cual o ampliarse para hacer frente a una amplia variedad de proyectos. Es un proceso de desarrollo de *software* mínimamente suficiente dado que incluye solo el contenido fundamental no teniendo que lidiar con temas tales como el tamaño del equipo, el cumplimiento, seguridad, orientación tecnológica, entre otros. Sin embargo *OpenUP* manifiesta por completo el proceso de construir un sistema. Para atender las necesidades que no están cubiertas en su contenido *OpenUP* es extensible a ser utilizado como base sobre la cual se pueden añadir o adaptarse a contenido de otro proceso que sea necesario. Se pueden resaltar dentro de sus características más importantes las siguientes (31):

- Desarrollo incremental.
- Uso de casos de uso y escenarios.
- Manejo de riesgos.
- Diseño basado en la arquitectura.

OpenUP es un proceso completo, flexible y corto, fomenta el uso de técnicas ágiles y principios, mientras que tiene un ciclo de vida estructurado y probado que hace referencia en la continua entrega de *software* de calidad (31).

***Extreme Programing* (XP)**

XP o Programación Extrema es una metodología que se centra en el desarrollo de *software* y en un conjunto de reglas que giran alrededor de las necesidades del cliente, con el objetivo de lograr un producto de buena calidad con una reducción considerable de tiempo. Básicamente se encarga de potenciar las relaciones interpersonales como un factor clave para el éxito, por lo que opera directamente con el cliente. Uno de los elementos a tener en cuenta en esta metodología es definir un estándar en el tipo de codificación, debido a la necesidad de poner en práctica la programación en pares, lo que conlleva a que los programadores tengan bien definido un estilo común de programación. Las pruebas constituyen una acción más que necesaria en cada iteración, ya que permiten que se prevean errores a medida que se programe. El fin de XP es generar versiones de un sistema, tan pequeñas como sea posible, pero que proporcionen un valor adicional claro desde la perspectiva del negocio. Utiliza historias de usuarios para cubrir la falta de casos de uso (32).

XP presenta un proceso iterativo e incremental y su ciclo de vida consta de cuatro fases: planificación, diseño, desarrollo o codificación y prueba.

Selección de la metodología a utilizar

Para guiar el trabajo se escoge la metodología *OpenUP* porque tiene un enfoque centrado en el cliente, con iteraciones de corta duración que permiten la detección temprana de errores. Es un proceso iterativo e incremental que tiene las características de ser mínimo, completo y extensible. Se genera solo la documentación verdaderamente necesaria. Se valora el aporte del cliente y otras personas interesadas en el resultado.

1.10 Servidor *Web*

Apache es un servidor HTTP (*Hypertext Transfer Protocol*). Es un *software* libre y de código abierto para las plataformas *Windows*, *Mac OS X* y *UNIX* (GNU, BSD). Posee un gran desempeño y una sólida robustez, gracias a su constante desarrollo. Provee una buena base para la seguridad del sistema, debido a los módulos de autentificación, autorización y control de acceso al servidor *web*, además el nuevo módulo para

la versión 2.0 que permite poder utilizar SSL/TLS en el servidor para mayor seguridad en la transmisión de datos. Soporta la versión 1.1 del protocolo HTTP. El soporte es abundante al igual que la documentación. La extensibilidad por módulos lo hace flexible y fácil de usar, así como de configurar, además de que trabaja en conjunto con gran cantidad de lenguajes de programación interpretados como PHP, *Perl*, soporte con CGI (*Common Gateway Interface*), *Java* y JSP (*Java Server Pages*) (33).

1.11 Bibliotecas para generar gráficos

HighCharts

Es una biblioteca desarrollada en *JavaScript* que ofrece gráficos intuitivos e interactivos. Permite la creación de una gran variedad de estos, entre los que se destacan los de línea, spline, área, areaspline, columna, barra, pastel y los tipos de gráfico de dispersión. Es compatible con cualquier navegador que soporte *JavaScript*. Una de sus grandes ventajas es que funciona lo mismo con *Windows*, *Linux*, *Mac*, *Android*, iOS, *Windows Phone*, *Firefox*, *Chrome*, *Safari*, *Internet Explorer*, en una PC, en un *smartphone* o en una tableta (34).

Permite la creación de gráficos interactivos así como la creación de gráficos en tiempo real obteniendo los datos en forma constante desde una base de datos u otra fuente de información desde el servidor. Posibilita la combinación de varios tipos de gráficos, la rotación de texto, ampliar las diferentes áreas del gráfico, modificar el aspecto, visualizar porcentajes, gráficos con valores negativos entre otros. Su principal desventaja es que aunque ofrece una licencia gratis para uso no comercial sí hay que pagarla para un sitio comercial.

Chart.js

Chart.js es una biblioteca *JavaScript* que permite generar distintos tipos de gráficas. Es posible representar datos usando 6 tipos de gráficas diferentes, totalmente personalizables y animadas. Funciona con HTML5 por lo que soporta la mayoría de los navegadores modernos. No depende de bibliotecas externas y es muy ligero. *Chart.js* es una biblioteca de código abierto bajo licencia MIT y dispone de muy buena documentación con ejemplos de uso completos. Además es compatible con dispositivos móviles como celulares y tabletas proporcionando una correcta escala de granularidad para ellos. Es modular lo que permite utilización solo de los tipos de gráficos necesarios (35).

Flot

Es una biblioteca gráfica para *JQuery* que brinda gráficos atractivos, una sintaxis intuitiva. Además provee soporte para eventos y la posibilidad de ser extendida a través de *plugins*. *Flot* acepta múltiples series de datos. Presenta además algunos códigos contenedores para activar *Excanvas* con los que *Flashcanvas* es compatible. Admite todos los navegadores que soportan la etiqueta *canvas* HTML 5. Es compatible con los navegadores: *Internet Explorer* 6+, *Chrome*, *Firefox* 2+, *Safari* 3+ y *Opera* 9.5+. Sin embargo tiene problemas para medir las dimensiones de la etiqueta y para medir las dimensiones de marcador de posición (36).

Selección de la biblioteca para generar gráficos

Se definió utilizar *HighCharts* pues brinda mayores ventajas con respecto a las otras, ya que muestra una alta calidad en los gráficos, así como la combinación de estos; posibilita también el uso de graficas interactivas, presenta gran variedad de gráficos y además es gratuita.

1.12 Bibliotecas para exportar a PDF

TCPDF

Es una clase de *software* libre para el lenguaje de programación *web* PHP en sus versiones 4 y 5, la cual permite crear ficheros PDF dinámicamente. Posee un buen rendimiento que no se ve afectado por la complejidad o extensión de los documentos, es capaz de generarlos en un tiempo mínimo. Esta no necesita bibliotecas externas para las funciones básicas. Incluye soporte para página en formato ISO y soporte de UTF-8 *Unicode* e idiomas de derecha a izquierda. Permite interpretación de HTML. Contiene métodos para la creación de código de barras. Además brinda soporte de fuentes *truetypeunicode, opentypeunicode, opentype, truetype, type1* y CID-0 *fonts* (37).

Incluye métodos para la creación de cabeceras y pies para las páginas. Permite soporte de imágenes, colores, compresión de páginas. Incluye gráficos y métodos de transformación. Facilita el encabezado de página automático y la gestión de pie de página. También garantiza la encriptación de documento de hasta 256 *bits* y certificados de firma digital (37).

FPDF

Es una clase escrita en PHP que permite generar documentos PDF directamente. No necesita de ninguna extensión para PHP (excepto zlib para activar la compresión y la biblioteca de gráficos GD para soporte a

gif) y funciona con PHP4 (≥ 4.3.10) y PHP5. Entre sus principales ventajas esta la utilización de funciones de alto nivel. Además permite hacer elección de la unidad de medida, formato de página y márgenes. Posibilita la gestión de cabeceras y pies de páginas, así como el salto de página automático, y salto de línea y justificación del texto automáticos. También permite la admisión de imágenes (jpeg, png y gif) y de fuentes *truetype, type* 1 y codificación. La desventaja en cuanto a rendimiento es muy razonable y se adecúa a la mayoría de los casos, la velocidad de generación de un documento, especialmente cuando estos son complejos o extensos no es la mejor (38).

Selección de la biblioteca para exportar a PDF

Ambas bibliotecas presentan características similares en cuanto a las funcionalidades que brindan, los lenguajes de programación que soportan. Sin embargo el peso de la selección de la biblioteca que será utilizada en el desarrollo del sistema recae en el rendimiento para generar documentos complejos o extensos, que favorece a TCPDF.

1.13 Conclusiones parciales

El estudio de las definiciones asociadas al campo de acción posibilitó un mejor entendimiento de la investigación, adquiriendo y fomentando los conocimientos necesarios para el desarrollo de la propuesta de solución. La identificación de las principales características funcionales, ventajas y desventajas de herramientas que realizan monitoreo y análisis de sitios *web* permitió guiar el desarrollo de la investigación. El estudio de las metodologías, tecnologías y herramientas de desarrollo de *software* permitió definir la utilización de la siguiente base tecnológica: *OpenUP* como metodología de desarrollo; *Symfony* 2.5 como *framework*; *PostgreSQL* 9.4 como sistema gestor de base de datos; PHP 5, *JQuery* y CSS 3 como lenguajes de programación; HTML 5 como lenguaje de marcado; UML como lenguaje de modelado; las herramientas *NetBeans* 8.0 y *Visual Paradigm* 8.0; como servidor *web Apache* 2.0 y las bibliotecas *HighCharts* y TCPDF para generar gráficas y exportar a PDF.

Capítulo 2: Análisis y diseño del sistema

El análisis y diseño constituyen dos etapas importantes en el desarrollo de un *software*, pues es necesario definir los procesos que intervienen en este para lograr un mejor entendimiento del sistema entre clientes y desarrolladores. El objetivo de este capítulo radica en especificar brevemente los artefactos construidos que forman parte de la metodología de desarrollo *OpenUP* propuesta en el capítulo anterior. Se realiza el modelado del negocio con el objetivo de esclarecer el contexto de la solución. Se especifican los requisitos funcionales y no funcionales del sistema, así como los casos de uso y las tareas de ingeniería asociadas.

2.1 Modelo del dominio

Un modelo del dominio es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés (39). Para la presente investigación, se decide realizar este artefacto, debido a que no es posible identificar con claridad los procesos y actores que realizan las actividades asociadas al negocio. Su correcta definición contribuirá a lograr una mejor comunicación entre usuarios y desarrolladores, mediante el establecimiento de un vocabulario común que permita entender el funcionamiento del sistema.

La ilustración 1 muestra el diagrama de modelo del dominio en el cual se representan las clases conceptuales y las relaciones entre ellas: las clases “publicación”, “comentario” y “sitio *web”* están relacionadas de igual modo con la clase “alerta” ya que una publicación, un comentario o un sitio *web* pueden recibir varias o ninguna alerta y una misma alerta puede estar dirigida a una publicación, a un comentario o ambos o puede estar dirigida a un sitio *web*. La clase “publicación” tiene una relación con la clase “seguimiento” debido a que se le puede dar seguimiento a una o varias publicaciones pero una publicación solo puede tener un seguimiento. También posee una relación con “comentario” pues una publicación puede tener uno, varios o ningún comentario relacionado y un comentario está asociado a una sola publicación; se relaciona además con “sitio *web”* a causa de que una publicación puede pertenecer a uno o varios sitios *web* y un sitio *web* puede contener una o varias publicaciones; de igual modo está relacionada con “nomenclador” ya que una publicación puede tener uno o varios nomencladores asociados y un nomenclador puede pertenecer a una o varias publicaciones; y finalmente está relacionada con las clases “redactor” y “fuente” debido a que una publicación puede tener uno, varios o ningún redactor o fuente asociada y un redactor o una fuente pueden estar relacionados con una o varias publicaciones.

Por otro lado están las clases “comentario” y “alerta” que tienen una relación con la clase “nomenclador” pues un comentario o un alerta contiene únicamente un nomenclador y el mismo nomenclador puede

pertenecer a una o varias alertas o comentarios. La clase “comentario” también está relacionada con la clase “redactor” de forma tal que un comentario le corresponde a un único redactor y un redactor puede tener varios comentarios. La clase “alerta” por su parte está relacionada con la clase “directivo” debido a que una alerta se puede enviar a uno o varios directivos y un directivo puede tener una o varias alertas asociadas. Las clases “publicación”, “comentario” y “alerta” representan textos, las clases “fuente” y “sitio *web*” medios y las clases “directivo” y “redactor” personas, todas forman parte del sistema de información. La clase “traza” representa las acciones de los usuarios en el sistema y la clase “usuario” sistema representa propiamente los usuarios que acceden al sistema.

Diagrama de modelo del dominio

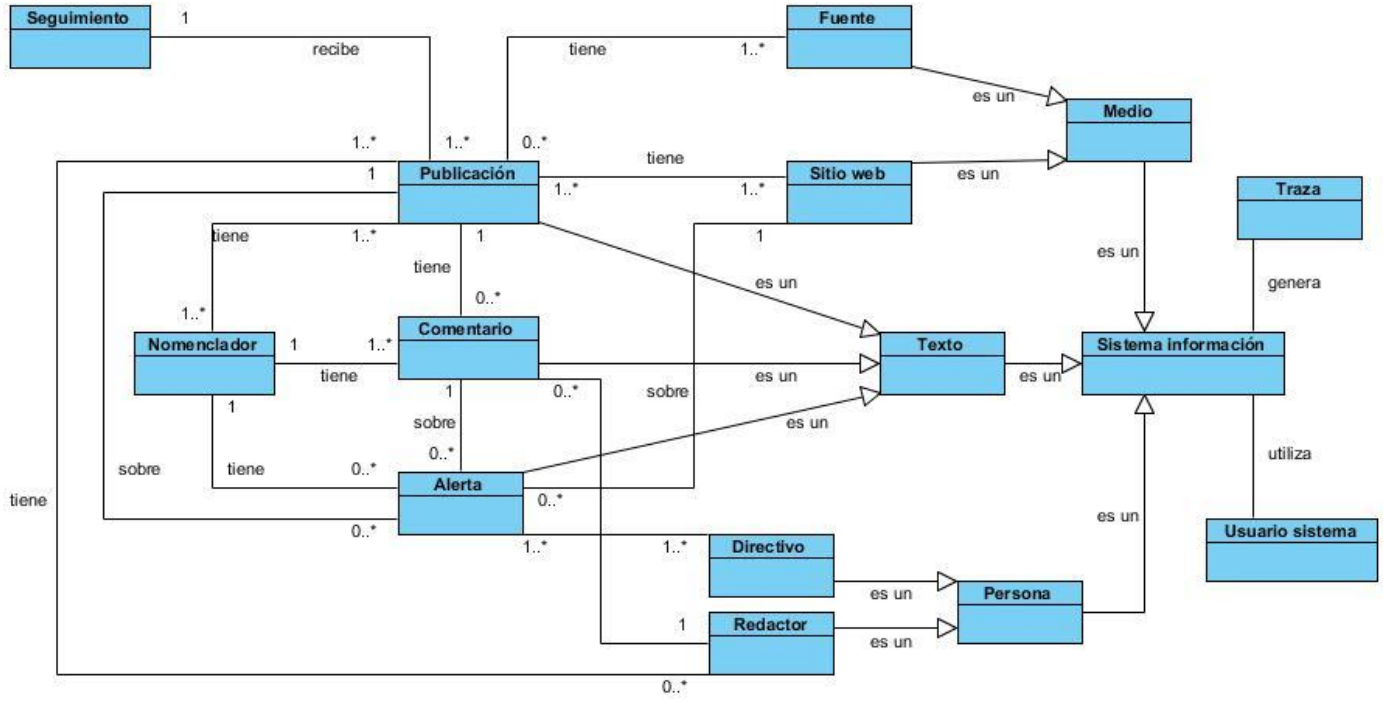


Ilustración 1 Diagrama de modelo del dominio

2.2 Requisitos de *software*

Los requerimientos para un sistema o requisitos de *software* son la descripción de los servicios proporcionados por el sistema y sus restricciones operativas. Estos requerimientos reflejan las necesidades de los clientes de un sistema que ayude a resolver algún problema (40).

2.2.1 Requisitos funcionales

Los requerimientos o requisitos funcionales (RF) son declaraciones de los servicios que debe proporcionar el sistema, de la manera en que éste debe reaccionar ante entradas particulares y de cómo se debe comportar en situaciones particulares (40). A continuación se muestra en la tabla 1 los requisitos funcionales de prioridad alta, el resto se adhiere al anexo 2:

Tabla 1 Especificación de requisitos funcionales

Nº	Nombre	Descripción	Prioridad para el cliente	Complejidad
RF1	Autenticar usuario	Permite la autenticación del usuario con sus parámetros de conexión: usuario y contraseña, para que pueda consumir los demás servicios del sistema en correspondencia con los permisos que tiene asignados por su rol.	Alta	Alta
RF2	Gestionar publicación semiautomáticamente	El sistema permite al usuario en dependencia de los permisos asignados realizar varias acciones en correspondencia con una publicación, listar o crear.	Alta	Alta
RF2.1	Clasificar publicación semiautomática	Permite al usuario crear una nueva publicación de forma semiautomática, por lo que solo tendrá que definir algunos parámetros correspondientes a una publicación, el resto se mostrará automáticamente.	Alta	Media

RF2.2	Listar publicación semiautomática	Permite mostrar un listado de las publicaciones obtenidas de una base de datos externa.	Alta	Baja
RF2.3	Buscar publicación semiautomática	Permite buscar una publicación por el título y por el sitio *web.*	Alta	Media
RF2.4	No procesar una publicación semiautomática.	Permite seleccionar de la lista de publicaciones obtenidas de la base de datos externa la que no se procesará por el sistema.	Alta	Alta
RF3	Actualizar publicación semiautomática	Permite actualizar las publicaciones introducidas al sistema de forma semiautomática.	Alta	Alta
RF4.1	Crear publicación manual	Permite al usuario crear una nueva publicación insertando todos los parámetros correspondientes a la misma de forma manual.	Alta	Alta
RF4.2	Listar publicación manual	Permite al usuario mostrar un listado de las publicaciones creadas en el sistema.	Alta	Baja
RF4.3	Buscar publicación	Brinda la posibilidad de insertar un criterio de búsqueda para obtener la publicación deseada.	Alta	Medio
RF4.4	Modificar publicación	Permite modificar algunos parámetros correspondientes a alguna publicación creada previamente.	Alta	Alta
RF4.5	Eliminar publicación	Permite seleccionar de un listado la publicación que se desee eliminar.	Alta	Baja
RF6.1	Crear comentario	Permite hacerle un comentario a una publicación previamente creada.	Alta	Media

RF6.2	Listar comentario	Permite al usuario mostrar un listado de los comentarios creados en el sistema.	Alta	Baja
RF6.3	Buscar comentario	Permite insertar en el sistema algún criterio de búsqueda para obtener un comentario.	Alta	Media
RF6.4	Modificar comentario	Permite modificar alguno de los elementos correspondientes a un comentario creado	Alta	Media
RF6.5	Eliminar comentario	Permite eliminar un comentario de una publicación.	Alta	Baja
RF7	Clasificar comentario semiautomático	Permite clasificar los comentarios correspondientes a las publicaciones que se introduzcan en el sistema de forma semiautomática.	Alta	Alta
RF12.1	Crear alerta	Permite generar una alerta a una determinada publicación, un comentario o un sitio *web*.	Alta	Media
RF12.2	Listar alerta	Permite al usuario mostrar un listado de las alertas creadas en el sistema.	Alta	Media
RF12.3	Buscar alerta	El sistema permite buscar una alerta creada previamente.	Alta	Baja
RF12.4	Modificar alerta	Permite modificar algún parámetro correspondiente a una alerta creada.	Alta	Media
RF12.5	Eliminar alerta	Permite eliminar determinada alerta hecha a una publicación o un comentario.	Alta	Media

2.2.2 Requisitos no funcionales

Los requerimientos o requisitos no funcionales (RNF) son restricciones de los servicios o funciones ofrecidas por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad (40).

Software

- RNF1: en las máquinas clientes debe estar instalado el sistema operativo GNU/*Linux* preferentemente *Ubuntu* GNU/*Linux* 12.04 o *Windows* XP o versiones superiores de ambos.
- RNF2: en los servidores debe estar instalado el sistema operativo GNU/*Linux* preferentemente *Ubuntu* GNU/*Linux* 12.04 o *Windows* XP o versiones superiores, servidor de base de datos *PostgreSQL* 9.4 en adelante y servidor *web Apache* 2.2 o superior.

Hardware

- RNF3: procesador *Intel Pentium Processor* G2020T (3MB cache, 2.50 GHz) o AMD similar o superiores para el servidor y para el cliente.
- RNF4: 2 *gigabytes* (GB) de memoria RAM (*Random Access Memory*, por sus siglas en inglés) o superior para el servidor y 512 *megabytes* (MB) para el cliente o superior (recomendado 1024 MB).
- RNF5: 80 GB de espacio en disco duro o superior.

Seguridad

- RNF9: un usuario no autenticado no podrá hacer uso de las funcionalidades de los módulos.

Usabilidad

- RNF10: el sistema podrá ser usado por cualquier persona que tenga conocimientos básicos en el manejo de la computadora y de un ambiente *web* en sentido general.
- RNF11: los mensajes, textos y formularios expresarán claramente la información que se le desea mostrar al usuario.
- RNF12: el sistema debe permitir el acceso al menú general desde cualquiera de sus páginas.

Rendimiento

- RNF13: se garantizará que la respuesta a peticiones de los usuarios del sistema sea en un período de tiempo breve (en segundos), al igual que la velocidad de procesamiento de la información.

2.3 Modelo de casos de uso del sistema

Un modelo de caso de uso describe lo que hace un sistema sin especificar como lo hace, es un modelo lógico del sistema. Refleja la vista de un sistema desde la perspectiva de un usuario fuera del sistema (41).

En esta sección se identifican los actores y casos de uso del sistema, quedando determinado el diagrama de casos de uso.

La ilustración 2 muestra el diagrama de casos de uso del sistema el cual representa las relaciones entre los actores y los casos de uso. Se definió un actor genérico “usuario” que inicia el caso de uso autenticar usuario, del cual heredan el resto de los actores del sistema. El actor “editor” inicia los casos de uso correspondientes a la gestión y clasificación de comentarios, clasificar y actualizar publicaciones de forma semiautomática, gestionar publicaciones manuales y listar las publicaciones existentes. El actor “gestor” realiza la gestión de las fuentes, los redactores, los directivos y las alertas, además genera todos los reportes, las gráficas y los análisis estadísticos y puede exportarlos a PDF o a imagen en el caso de las gráficas. El actor “periodista” es el encargado de generar al boletín avanzado y puede exportarlo a PDF. Se creó un actor genérico “gestor-editor” del cual heredan el editor y el gestor, el mismo realiza la gestión del seguimiento que se le hace a las publicaciones.

Por otra parte está el actor “administrador” el cual hereda las funcionalidades del editor, el gestor, el periodista, gestor-editor y del usuario, además realiza la gestión de los sitios *web*, los nomencladores y de los usuarios, puede consultar las trazas generadas y administrar las configuraciones del sistema como las variables globales y el modo de autenticación.

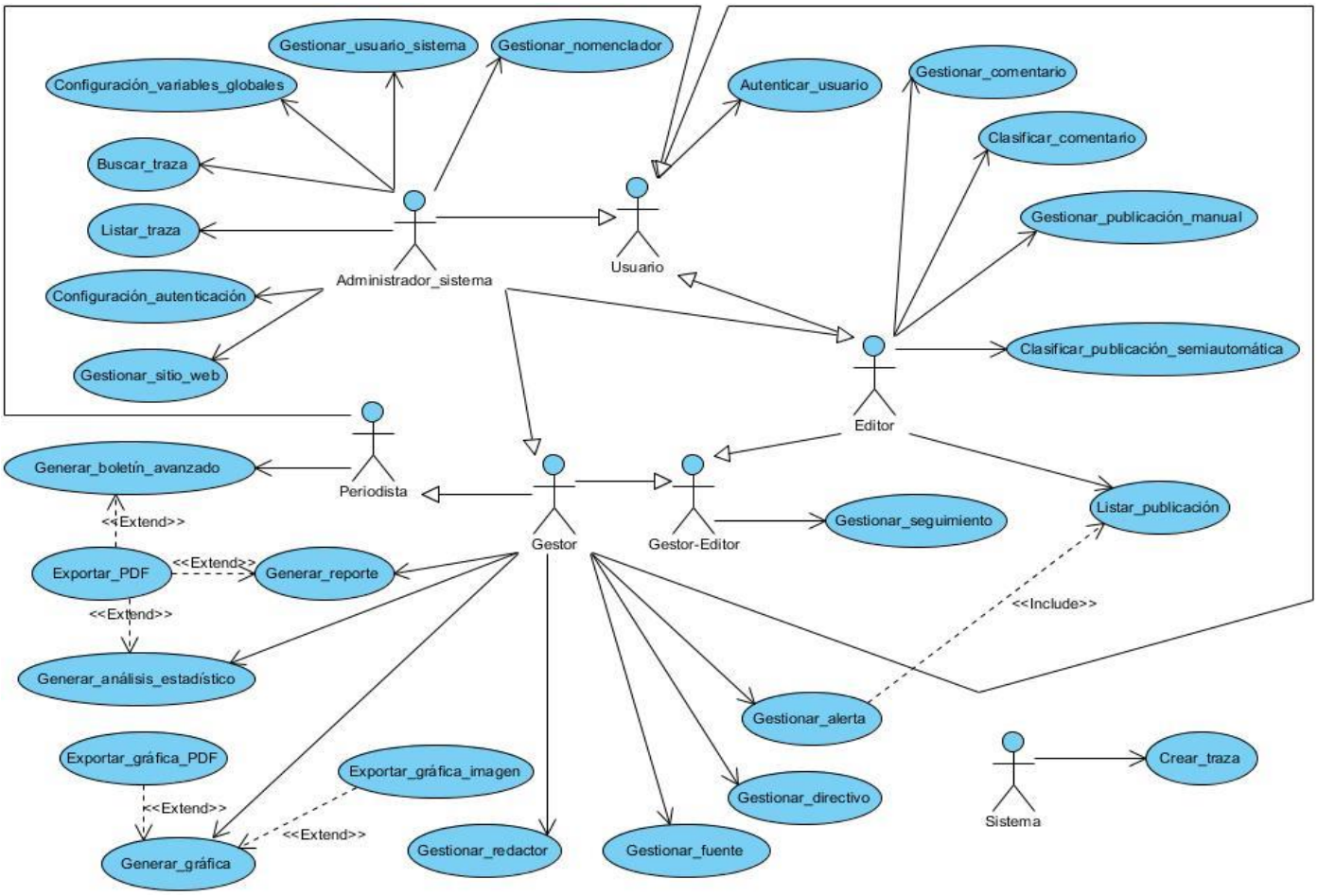


Ilustración 2 Diagrama de CU del sistema

Definición de los actores del sistema

Los actores del sistema intercambian información con él. Pueden representar el rol que juegan una o varias personas, un equipo o un sistema automatizado. A continuación se muestran los actores y la justificación que tienen en el sistema.

Tabla 2 Descripción de los actores del sistema

Actor	Objetivos

Editor	Gestionar publicaciones y comentarios. Solo puede editar y eliminar los suyos propios. Gestionar seguimiento.
Gestor	Gestionar las fuentes, directivo, redactor, alertas y seguimiento en el sistema. Generar gráfica, reporte y análisis estadístico.
Periodista	Gestionar boletín avanzado.
Administrador	Administrar todas las funcionalidades del sistema.
Sistema	Crear trazas de sistema.

Especificación de casos de uso

Los casos de uso (CU) son fragmentos de funcionalidades que agrupan los requisitos que debe cumplir el sistema. A continuación se muestra la especificación de uno de los CU del sistema; en el anexo 3 se adhieren las especificaciones del resto de los CU.

Tabla 3 Descripción CU: gestionar publicación manual

Objetivo	El sistema permita la gestión de una publicación de forma manual.
Actores	Administrador, editor (Inicia)
Resumen	El caso de uso inicia cuando el actor selecciona alguna de las opciones que le permiten gestionar una publicación. El sistema muestra un listado del cual se seleccionan la publicaciones para realizar un conjunto de acciones como: crear, buscar, mostrar, modificar y eliminar, en caso de crear o modificar el sistema permite introducir un conjunto de datos asociados a la misma, en el caso de buscar se introduce un criterio de búsqueda y se muestra un listado de publicaciones que coincidan con dicho criterio, mostrar permite ver los detalles de una publicación y en caso de eliminar se selecciona la publicación y se elimina. Para el actor editor el sistema solo permite modificar y eliminar las introducidas al sistema por él mismo. El sistema guarda los cambios realizados y termina el caso de uso.
Complejidad	Alta

Prioridad	Crítico
Precondiciones	El usuario ha sido autenticado correctamente en el sistema.
Postcondiciones	Los datos introducidos o modificados por el usuario deben quedar registrados en el sistema, listados, mostrados o eliminados si es ese el caso.

Flujo de eventos	
Flujo básico: Gestionar publicación manual	
Actor	**Sistema**
1. Selecciona la opción “Publicación” en el menú “Navegación” de la barra de menús.	2. Muestra un listado de las publicaciones creadas. Permite realizar varias acciones: - Crear una publicación: Ver sección 1: crear publicación. - Buscar una publicación. Ver sección 2: buscar publicación. - Modificar una publicación. Ver sección 3: modificar publicación. - Eliminar una publicación. Ver sección 4: eliminar publicación. - Mostrar los detalles de una publicación. Ver sección 5: mostrar detalles.
	Termina el caso de uso.
Sección 1: Crear publicación	
Flujo básico	
Actor	**Sistema**
1. Selecciona la opción crear publicación.	2. Muestra una vista para que el usuario inserte los datos correspondientes a una publicación: - título - fecha de publicación - cantidad de visitas - vínculo

	- redactor - fuente - dimensión - área - temática - idioma - estilo - sitio *web* - cuerpo Muestras las opciones: “Aceptar y continuar”, “Guardar” y “Cancelar”.
3. Ingresa los datos requeridos y selecciona la opción “Guardar”.	4. Guarda los cambios e inserta la publicación en el sistema.
Flujos alternos	
Nº 3 Los datos son incorrectos	
Actor	**Sistema**
	4. Muestra un mensaje de error con los datos que son incorrectos. Ir paso 3.
Nº 3 Faltan campos por completar	
Actor	**Sistema**
	4. Muestra un mensaje de error con los campos que faltan por completar. Ir paso 3.
Sección 2: Buscar publicación	
Flujo básico	
Actor	**Sistema**
1. Ingresa los datos del criterio de búsqueda.	2. Muestra una tabla con los resultados que coincidan con dicho criterio. Permite realizar varias acciones con estos resultados:

	- Modificar una publicación. Ver sección 3: modificar publicación. - Eliminar una publicación. Ver sección 4: eliminar publicación.
Sección 3: Modificar publicación	
Flujo básico	
Actor	**Sistema**
1. Selecciona la opción modificar publicación.	2. Muestra una vista que permite modificar los datos correspondientes a una publicación.
3. Ingresa los datos.	4. Guarda los cambios.
Flujos alternos	
Nº 3 Los datos son incorrectos	
Actor	**Sistema**
	4. Muestra un mensaje de error con los datos que son incorrectos. Ir paso 3.
Nº 3. Faltan campos por completar	
Actor	**Sistema**
	4. Muestra un mensaje de error con los campos que faltan por completar. Ir paso 3.
Sección 4: Eliminar publicación	
Flujo básico	
Actor	**Sistema**
1. Selecciona la opción eliminar una publicación.	2. Muestra una ventana para confirmar su selección con las opciones aceptar y cancelar.
3. Selecciona la opción “Aceptar”.	4. Elimina la publicación y guarda los cambios.
Flujos alternos	
Nº 3 Selecciona la opción “Cancelar”	
Actor	**Sistema**
	4. No realiza ninguna acción.

Sección 5: Mostrar detalles		
Flujo básico		
Actor		**Sistema**
1. Selecciona la opción para mostrar detalles.		2. Muestra una ventana con los datos correspondientes a una publicación.
3. Selecciona la opción “Cerrar”.		4. Cierra la ventana.
Relaciones	**CU incluidos**	
	CU extendidos	
Requisitos no funcionales		
Asuntos pendientes		

2.4 Patrón arquitectónico Modelo-Vista-Controlador (MVC)

La herramienta SIMAE v2.0 se desarrolla haciendo uso del *framework symfony*, utilizando la lógica que implementa el patrón arquitectónico de llamada y retorno conocido como Modelo Vista Controlador (MVC), representado en la ilustración 3.

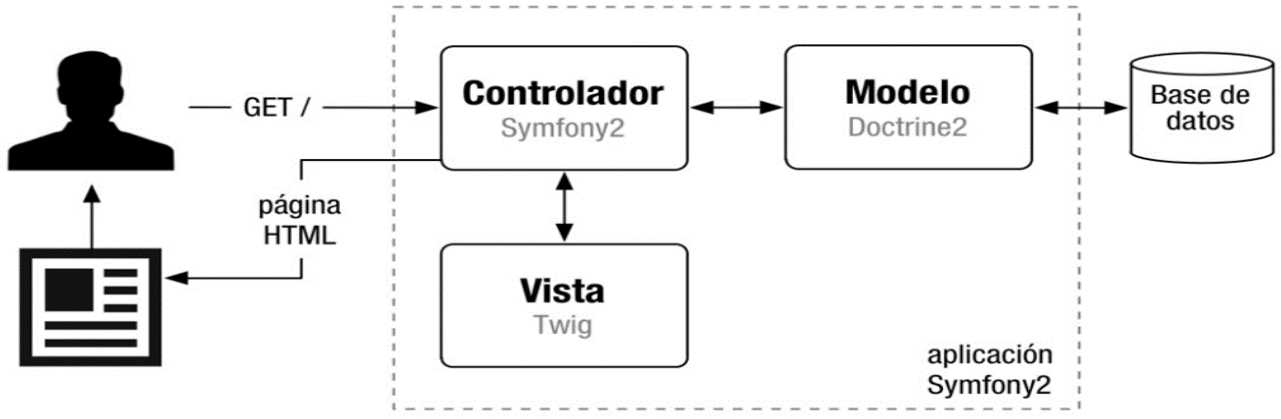


Ilustración 3 Patrón MVC en *symfony*

Es un patrón de arquitectura de *software* encargado de separar la lógica de negocio de la interfaz del usuario, facilita la funcionalidad, mantenibilidad y escalabilidad del sistema, de forma simple y sencilla (42). Divide una aplicación interactiva en tres áreas: procesamiento, salida y entrada. Utiliza las siguientes abstracciones (43):

- Modelo (*Model*): encapsula los datos y las funcionalidades. El modelo es independiente de cualquier representación de salida y/o comportamiento de entrada.
- Vista (*View*): muestra la información al usuario. Obtiene los datos del modelo. Pueden existir múltiples vistas del modelo. Cada vista tiene asociado un componente controlador.
- Controlador (*Controller*): reciben las entradas, usualmente como eventos que codifican los movimientos o pulsación de botones del ratón, pulsaciones de teclas, entre otros. Los eventos son traducidos a solicitudes de servicio para el modelo o la vista. El usuario interactúa con el sistema a través de los controladores.

En *symfony* la estructura de la base de datos es mapeada a clases del sistema mediante un ORM. En los *bundles* ComunBundle, QuaeroBundle y SeguridadBundle se utilizan algunas clases del modelo generadas por *doctrine*, algunos ejemplos de estas son: Publicacion.php, Comentario.php, Alerta.php, Sitiow.php, PublicacionQuaero.php, ComentarioQuaero.php, Usuario.php y Traza.php, las cuales permiten el acceso a los datos de las tablas correspondientes a estas clases.

La vista es la encargada de originar las páginas que son mostradas al usuario. Está representada por ficheros escritos en *twig,* que es un motor de plantillas que se encarga de construir las páginas HTML con las que interactúa el usuario. Estos archivos están ubicados en el directorio *Resources/Views* de cada *bundle* del sistema.

El nivel controlador está constituido por varios componentes:

- El controlador frontal, que es el único punto de entrada a la aplicación, el cual carga la configuración y determina la acción a ejecutarse.
- Las clases controladoras, que contienen la lógica de la aplicación, verifican la integridad de las peticiones y preparan los datos requeridos para actualizar la vista. En el sistema estas se encuentran en los directorios *Controller* de cada uno de los *bundles.*
- Los objetos *request*, *response* y *session*, dan acceso a los parámetros de la petición, las cabeceras de las respuestas y a los datos persistentes del usuario respectivamente.

2.5 Patrones de diseño

Estos patrones brindan una solución a problemas comunes que pueden ser encontrados durante el diseño, perfeccionando los componentes de un sistema de *software* y sus relaciones. Para estructurar el diseño del

sistema se utilizaron patrones que incluye el *framework* de desarrollo en su arquitectura, debido a que su utilización proporciona ventajas significativas pues no es necesaria su implementación ya que lo hace por defecto.

Patrones GRASP:

Los patrones GRASP (patrones generales de software para asignación de responsabilidades o *general responsibility assignment software patterns* por sus siglas en inglés) describen los principios fundamentales del diseño para la asignación de responsabilidades a objetos, expresados en forma de patrones (44). A continuación se hace una breve descripción de los mismos:

- **Experto:** su uso se encuentra enmarcado fuertemente en el mapeo y abstracción de la base de datos, se puede reflejar siempre que se generan de forma automática ambas capas del modelo. Es uno de los patrones más utilizados, debido a que *Doctrine*, genera las clases de acceso a datos asignándoles las responsabilidades de ejecutar todas las funcionalidades comunes de la entidad que representa y de la cual posee información. En la ilustración 4 se manifiesta este patrón debido a que *Doctrine* permite persistir objetos completos a la base de datos y recuperar objetos completos desde esta, sin tener en cuenta el sistema gestor de bases de datos empleado.

```
251
252     //Notificar alerta
253     public function registrarAlertaAction($id_publicacion, Request $request) {
254         $em = $this->getDoctrine()->getManager();
255         $alertas = $em->getRepository('PublicacionComunBundle:tbdpublicacion')->obtenerAlertasDadoId($id_publicacion);
            $publicacion = $em->getRepository('PublicacionComunBundle:tbdpublicacion')->find($id_publicacion);
257         $alerta = new tbdalerta($publicacion);
258         $form = $this->createForm(new \Publicacion\ComunBundle\Form\tbdalertaType(), $alerta);
259
```

Ilustración 4 Patrón experto

- **Alta cohesión y bajo acoplamiento:** estos patrones vienen implementados en el propio *framework Symfony*2. Las entidades implementadas reflejan la alta cohesión pues la información que almacena cada una de estas clases es coherente. El bajo acoplamiento es garantizado pues las clases se encuentran lo menos ligadas posibles entre sí, lo cual potencia la reutilización y disminuye la dependencia entre estas. La alta cohesión es una de las principales características de *Symfony*, es la organización del trabajo en cuanto a la estructura del proyecto, lo que permite crear y trabajar con clases con una alta cohesión. Por ejemplo, la clase “PublicacionController” contiene varias

funcionalidades en las que cada una posee un propósito único, no desempeñado por el resto de los elementos, siendo estas funcionalidades las encargadas de controlar la lógica de la gestión de las publicaciones. Esto hace posible que el *software* sea flexible a cambios sustanciales con efecto mínimo, lo que garantiza la alta cohesión.

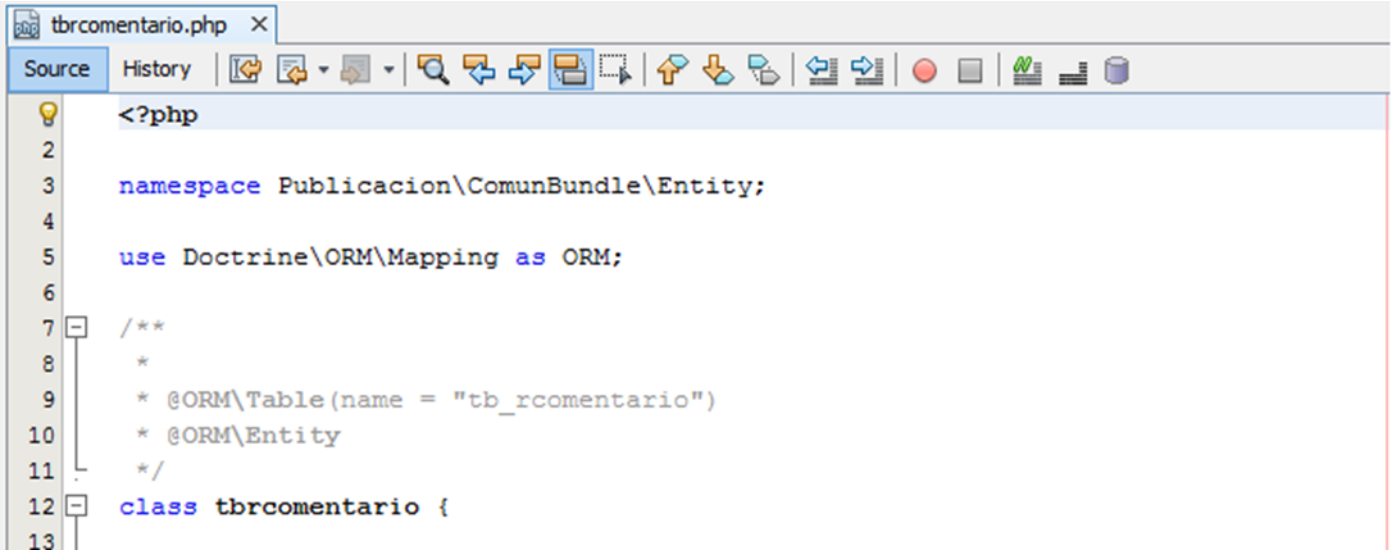


Ilustración 5 Patrón alta cohesión

- **Controlador:** este patrón se pone de manifiesto en los controladores de la aplicación que son los encargados de recibir y ejecutar la petición y enviar la respuesta correspondiente a la vista. Se basa en la existencia de un intermediario entre las páginas clientes y el algoritmo que responde a las peticiones realizadas por estas. La existencia del controlador frontal (ilustración 6), es el ejemplo básico que evidencia de forma clara su utilización. Este maneja las peticiones del usuario, la seguridad, carga la configuración de la aplicación, siendo único para la aplicación. En busca de aminorar la carga que posee se cuenta con las acciones que contienen las especificaciones de cada página.

  Cada petición es manejada por un único controlador frontal, ejemplo app_dev.php el cual es responsable de arrancar la aplicación.

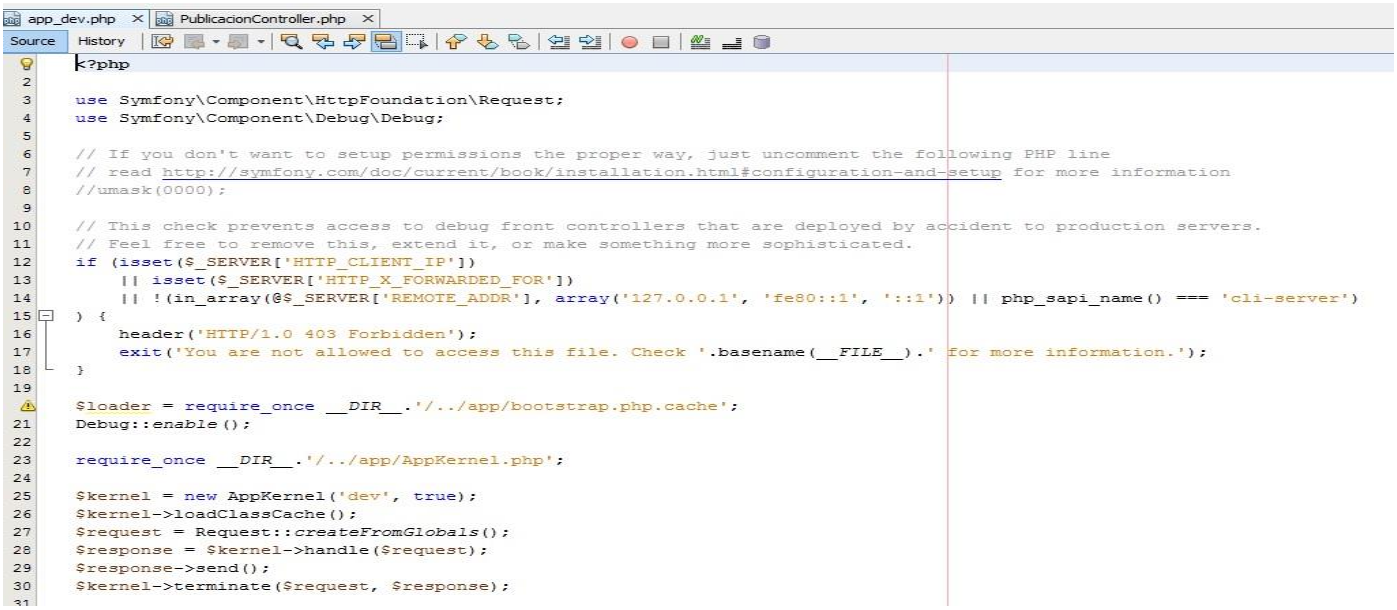


Ilustración 6 Patrón controlador

El enrutador lee la información de la petición, encuentra una ruta que coincida con esa información y luego ejecuta el controlador y el código dentro del mismo (ilustración 7).

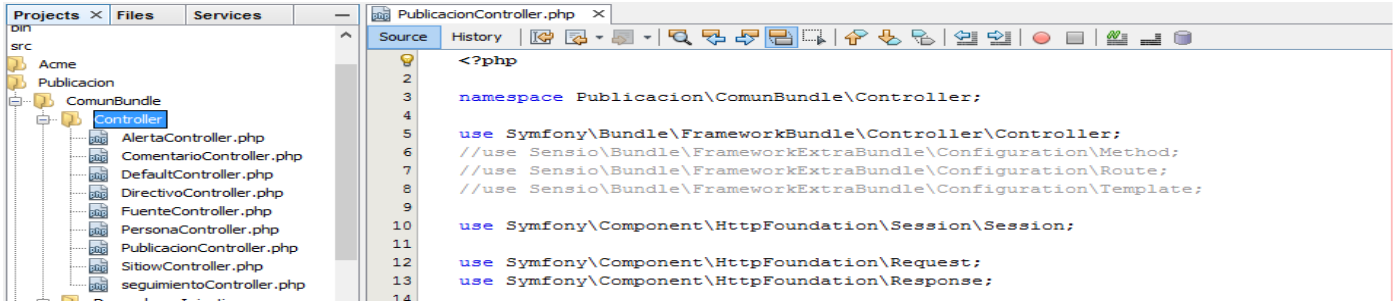


Ilustración 7 Patrón controlador

Patrones GOF

Existen tres grandes categorías de patrones GOF (*Gang of Four*), basadas en su propósito: creacionales, estructurales y de comportamiento.

Creacionales:

- ***Singleton* (Instancia única):** lo implementa *Symfony* por defecto pues todos sus controladores constituyen un *singleton*. Garantiza la existencia de una única instancia para una clase y la creación de un mecanismo de acceso global a dicha instancia (ilustración 8).

```
112     public function modificarPublicacionAction($id, Request $request) {
            $em = $this->getDoctrine()->getManager();
114         $publicacion = $em->getRepository('PublicacionComunBundle:tbdpublicacion')->find($id);
115         $cuerpo = $em->getRepository('PublicacionComunBundle:tbdpublicacioncuerpo')->find($id);
116
```

Ilustración 8 Patrón singleton

- ***Abstract factory* (Fábrica abstracta):** permite trabajar con objetos de distintas familias de manera que estas últimas no se mezclen entre sí y haciendo transparente el tipo de familia concreta que se esté usando. Cuando el *framework* necesita crear un nuevo objeto para una petición, busca en la definición de la factoría el nombre de la clase que se debe utilizar para esta tarea.

Estructurales:

- ***Decorator* (Envoltorio):** añade funcionalidad a una clase, dinámicamente. El archivo base.layout.html.twig, que también se denomina plantilla global, almacena el código HTML que es común a todas las páginas de la aplicación, para que no se repita en cada una de ellas. El contenido de la plantilla se integra en el *layout*, o si se mira desde el otro punto de vista, el *layout* decora la plantilla. La ilustración 9 muestra la plantilla hija listar.html.twig que extiende del diseño base (base.layout.html.twig) y reemplaza cualquiera de sus bloques.

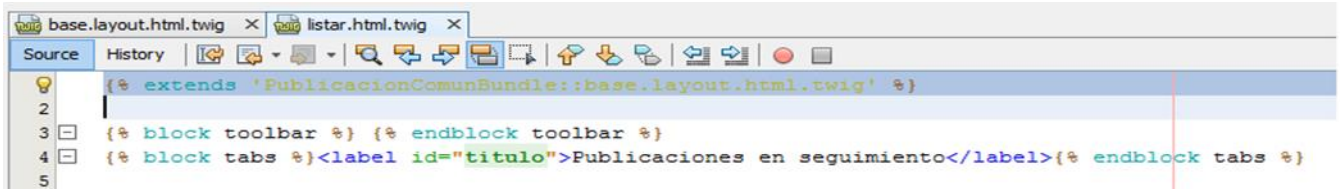


Ilustración 9 Patrón decorator

Comportamiento:

- **Unidad de trabajo:** permite que diferentes objetos puedan ejecutar la misma acción sin necesidad de repetir su declaración e implementación. En la ilustración 10 se evidencia este patrón ya que

Doctrine es consciente de todas sus entidades gestionadas cuando se llama al método *flush* (línea 102), calcula el conjunto de cambios y ejecuta la(s) consulta(s) más eficiente(s) posible(s). Por ejemplo si persiste un total de cinco objetos fuente, y posteriormente se llama al método *flush*, *Doctrine* creará una sola declaración preparada y la volverá a utilizar en cada inserción.

```
79      public function registrarPublicacionAction(Request $request) {
80          $em = $this->getDoctrine()->getManager();
81          $publicacion = new tbdpublicacion();
82          $fuentes = $em->getRepository('PublicacionComunBundle:tbnfuente')->findAll();
83          $form = $this->createForm(new tbdpublicacionType(), $publicacion);
84
85          if($request->getMethod() == 'POST'){
86              $all_post = $request->request->all();
87              $form->handleRequest($request);
88              $mensaje = $all_post['cuerpo_pub']; // Mensaje del cuerpo de la publicacion
89
90              $em->persist($publicacion);
91              $cuerpo = new tbdpublicacioncuerpo($publicacion, $mensaje);
92              $em->persist($cuerpo); //Registro el cuerpo y la publicacion en cascada
93
94              if(isset($all_post['fuentes'])){
95                  $fuentes = $all_post['fuentes'];
96                  foreach ($fuentes as $fuente) {
97                      $fuente = $em->getRepository('PublicacionComunBundle:tbnfuente')->find($fuente);
98                      $pubicacion_fuente = new tbrpublicacionfuente($publicacion, $fuente);
99                      $em->persist($pubicacion_fuente);
100                 }
101             }
102             $em->flush();
```

Ilustración 10 Patrón unidad de trabajo

2.6 Modelo de datos

El modelo de datos se utiliza para describir la estructura lógica y física de la información persistente que puede ser gestionada por el sistema. La ilustración 11 presenta el diseño de las tablas necesarias para el desarrollo de las funcionalidades que formarán parte de la base de datos después de identificar las clases persistentes y realizar las transformaciones correspondientes.

Se muestra la tabla “publicación” que almacena los datos correspondientes a una publicación, está relacionada con las tablas comentario, alerta, sitio *web*, fuente, seguimiento, dimensión, área de la publicación, temática, estilo, idioma, cuerpo de la publicación, palabras clave y con la tabla redactor a través de la tabla persona.

La tabla “comentario” contiene los datos de los comentarios realizados a una publicación, está relacionada con las tablas redactor y tipo de comentario. La de “alerta” brinda los datos de una alerta generada en el sistema a una publicación, un comentario, ambos o a un sitio *web*, esta tiene una relación con la tabla causa

y con la tabla directivo a través de la tabla área uci. Y finalmente la de “notificaciones”, que es una tabla de apoyo para el envío de los mensajes de alertas generados en el sistema.

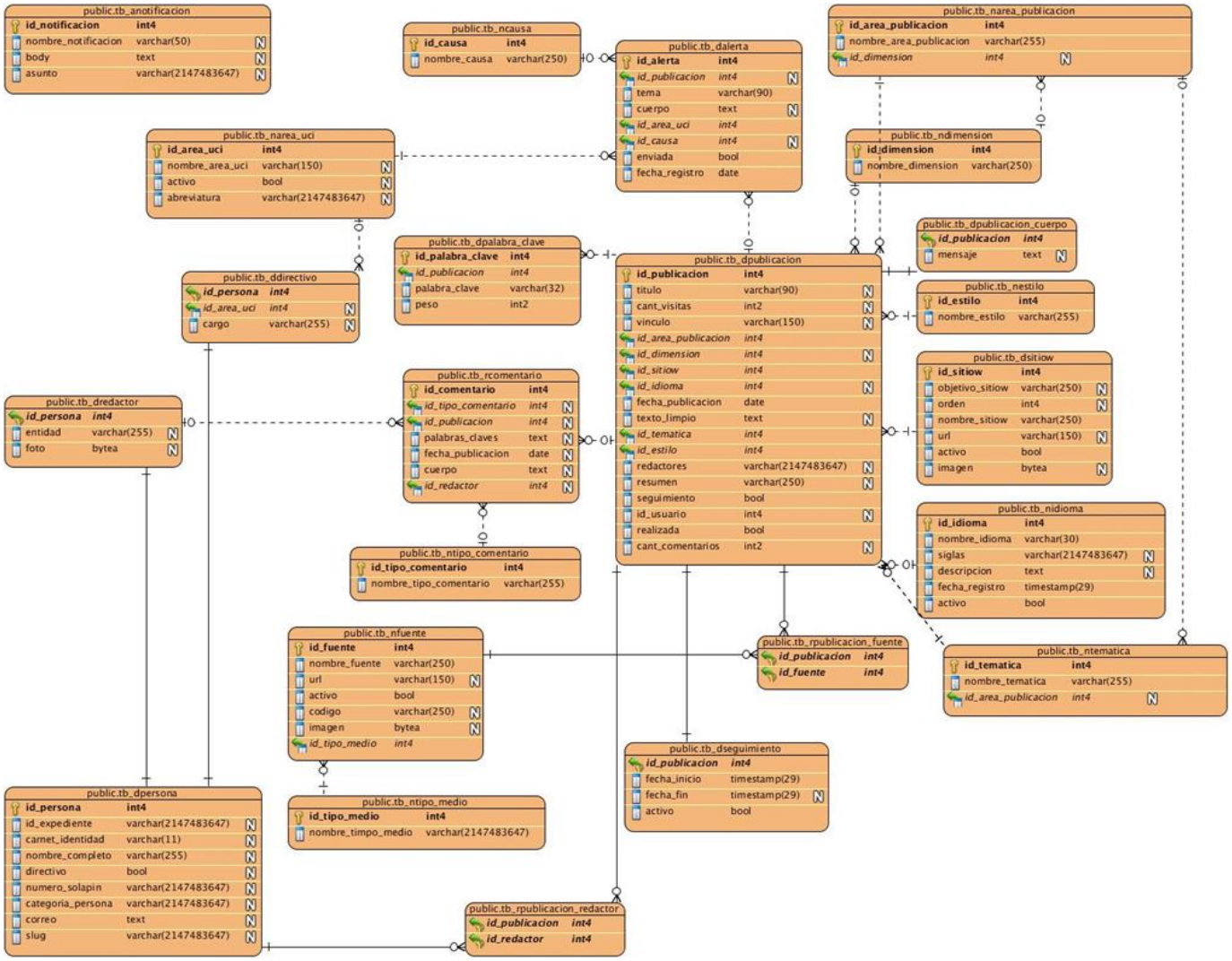


Ilustración 11 Diseño de la base de datos

2.7 Diagrama de clases de diseño

El diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces de *software* de una aplicación. Entre la información general se encuentran: las clases, atributos y sus relaciones, interfaces con sus operaciones y constantes, métodos, información acerca de los tipos de

atributos, navegabilidad y dependencias (39). Los diagramas de clases de diseño de la solución propuesta se basan en la implementación del patrón MVC que realiza *Symfony*. La ilustración 12 muestra el DCD correspondiente al CU gestionar publicación manual; el resto se adhieren en el anexo 4.

Las distintas *server page* como “SP_CrearPublicación”, “SP_ListarPublicacion”, “SP_BuscarPublicacion”, “SP_ModificarPublicacion”, “SP_MostrarPublicacion” y “SP_EliminarPublicacion” construyen las *client page* que tienen asociadas. Estas últimas son las encargadas de mostrar la información a los usuarios, pueden tener asociados formularios para recoger los datos introducidos. Dichos datos son enviados a las *server page*, las cuales están relacionadas con la clase controladora “PublicaciónController” que contiene los métodos y atributos que permiten manejar las acciones que realizan los usuarios. Esta a su vez está relacionada con la clase de acceso a dato que posibilita obtener la información contenida en la base de datos.

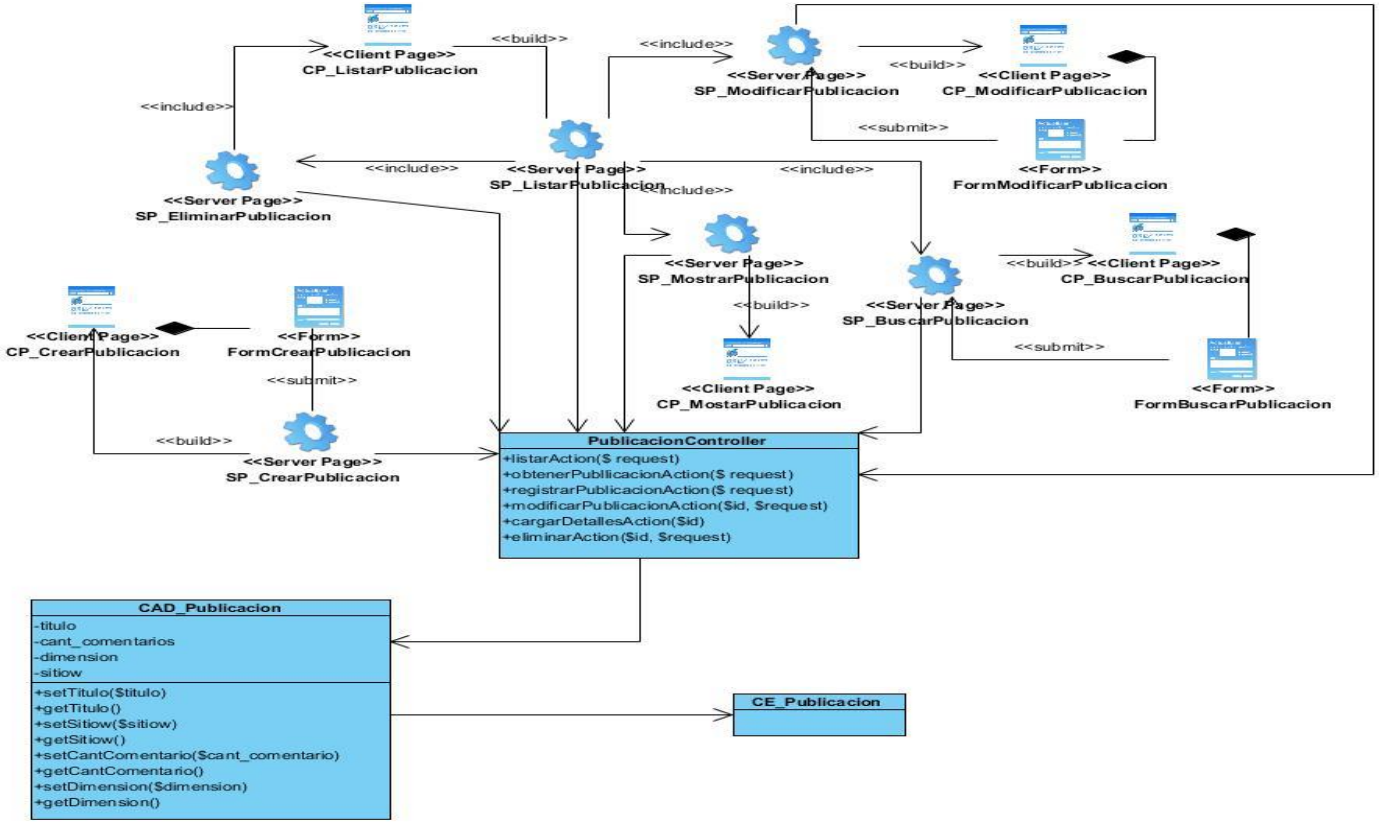


Ilustración 12 DCD CU 4 Gestionar publicación manual

2.8 Modelo de despliegue

El modelo de despliegue está constituido por un diagrama que es utilizado para representar la topología de un sistema, la estructura de sus elementos de *hardware* y *software*. El mismo representa nodos y las relaciones entre ellos; los nodos son conectados por enlaces de red. Según Craig Larman en su libro “UML y Patrones”, un diagrama de despliegue muestra la forma en que se configuran las instancias de los componentes y los procesos en tiempo de ejecución, en las instancias de los nodos de proceso (39). En la ilustración 13 se muestra el diagrama de despliegue correspondiente a la propuesta de solución.

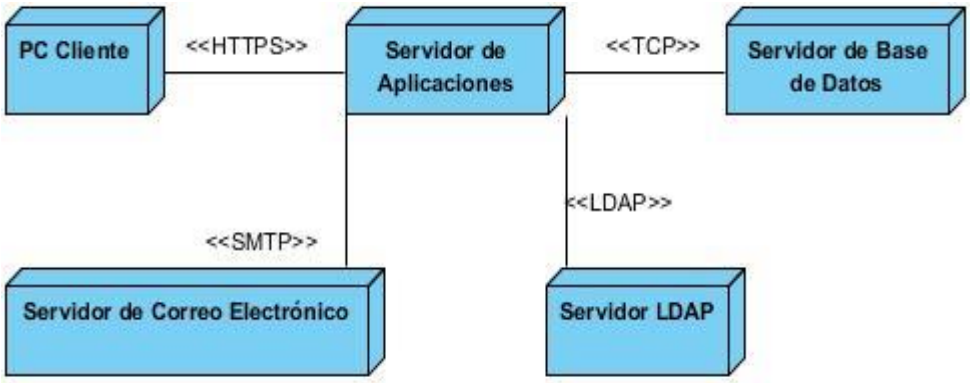


Ilustración 13 Diagrama de despliegue

El nodo PC Cliente de la figura representa los usuarios que se conectan al sistema realizando peticiones al nodo Servidor de aplicaciones mediante el protocolo https. El segundo nodo de izquierda a derecha posee el componente servidor que representa el servidor *web Apache* en el que se montará el sistema, el mismo recibe peticiones del cliente y realizan peticiones al servidor de base de datos mediante el protocolo tcp. El nodo que contiene el componente servidor de base de datos, que representa *PostgreSQL* que almacena la información del sistema. El servidor de aplicaciones se conecta al servidor ldap para realizar la autenticación de los usuarios del mismo modo para obtener datos de los usuarios, según lo requieran los procesos. Además es necesaria la comunicación entre el servidor de aplicaciones y un servidor de correo electrónico mediante el uso del protocolo smtp para el envío de correos electrónicos desde el sistema.

2.9 Conclusiones parciales

Se identificaron los principales conceptos asociados a la solución, reflejados en el modelo de dominio que posibilitó definir los requisitos funcionales y no funcionales que sustentan la propuesta de solución,

representados a través de los CU del sistema. Con la descripción realizada sobre los patrones de diseño se pudieron estructurar los diagramas de clases del diseño como entrada principal para la implementación. Además el desarrollo del diagrama de despliegue, permitió describir el sistema en términos de componentes en el próximo flujo de trabajo.

Capítulo 3: Implementación y prueba

Este capítulo muestra el periodo de implementación y pruebas partiendo de los resultados obtenidos durante el análisis y diseño de la solución. Para ello, se muestra la organización del sistema en términos de componentes, así como la dependencia con los nodos físicos a partir de los que funcionará la aplicación, permitiendo que los componentes desarrollados cumplan con los requisitos establecidos y puedan ser integrados a un sistema ejecutable. Posteriormente se aplicarán distintas pruebas para validar la calidad del sistema.

3.1 Modelo de implementación

El modelo de implementación es una colección de componentes y los subsistemas que los contienen. Estos componentes incluyen: ficheros de código fuente y otros tipos de ficheros necesarios para la implantación y despliegue del sistema. Describe también, cómo se organizan los componentes de acuerdo con los mecanismos de estructuración disponibles en el entorno de implementación y en el lenguaje o lenguajes de programación utilizados, y cómo dependen los componentes unos de otros (45).

3.1.1 Diagrama de componentes

Los diagramas de componentes son usados para estructurar el modelo de implementación en término de subsistemas de implementación y mostrar las relaciones entre los elementos de implementación. De esta forma, se modela la vista estática de un sistema, estableciendo la organización y las dependencias entre un conjunto de componentes de *software*, sean estos código fuente, paquetes, bibliotecas, binarios o ejecutables (28).

A continuación se muestra en la ilustración 14 el diagrama de componentes referente al CU gestionar publicación manual. El resto se adhieren en el anexo 5.

El componente *symfony* en su propia estructura es el encargado de relacionar el modelo, la vista y el controlador. El modelo comprende las clases “AreaPublicacion.php”, “Dimension.php”, “Tematica.php”, “Idioma.php”, “Publicación.php”, “PublicaciónRedactor.php”, “PublicacionFuente.php”, “Estilo.php”, “PublicacionCuerpo.php” y “PalabraClave.php” que permitirán el acceso a la base de datos a través del componente *doctrine*.

La vista incluye las plantillas “listar.html.twig”, “modificar_publicacion.html.twig”, “detalles.html.twig” y “registrar_publicacion.html.twig” que posibilitarán mostrar la información al usuario, para ello se apoyará en los componentes *jquery*, *javascript* y CSS.

El controlador contiene el componente correspondiente a la clase “PublicacionController.php” que maneja la lógica de la aplicación para este CU, verifica la integridad de los datos y los muestra a través de la vista.

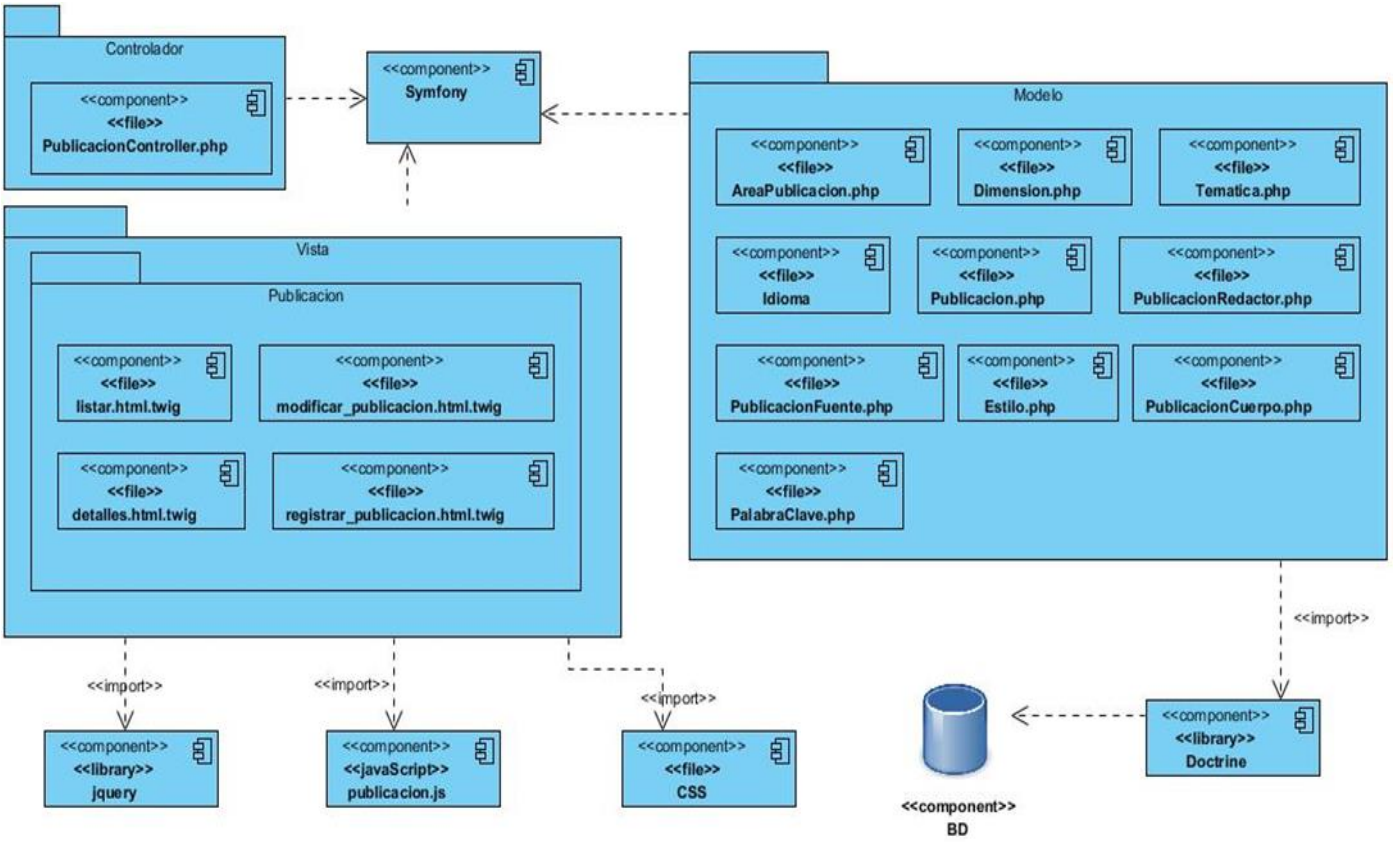


Ilustración 14 Diagrama de componentes CU 4 Gestionar publicación manual

3.1.2 Estándares de codificación

Los estándares de codificación son reglamentos de la programación que están enfocados a la estructura y apariencia física del programa con el objetivo de facilitar la lectura, comprensión y mantenimiento del código. Un estándar de codificación comprende todos los aspectos de la generación de código. Los programadores deben implementar un estándar de forma prudente que tienda siempre a lo práctico. La legibilidad del código

fuente influye directamente en el entendimiento que puedan tener los desarrolladores del equipo de trabajo, pues todo *software* tiene que someterse constantemente a mantenimiento y mejora de sus funcionalidades.

Los estándares utilizados en la codificación fueron los siguientes:

Identación, llaves de apertura y cierre, y tamaño de las líneas: usar una identación sin tabulaciones, con un equivalente a 4 espacios. El uso de las llaves "{}" será en una nueva línea. La longitud de las líneas de código es aproximadamente de 75-80 caracteres para mantener la legibilidad del código. Ver ilustración 15.

```
11      public function listarAction()
12      {
13          return $this->render('PublicacionComunBundle:comentario:listar.html.twig');
14      }
```

Ilustración 15 Identación, llaves de apertura y cierre, y tamaño de las líneas

Convención de nomenclatura:

- **Variables:** se rigen por la nomenclatura *underscoar*. Siempre debe ser todo en minúscula, con guión bajo "_" para separar palabras en caso de nombres compuestos. Ver ilustración 16.

```
118         $fuentes = $em->getRepository('PublicacionComunBundle:tbnfuente')->findAll();
119         $fuentes_asociadas = array();
```

Ilustración 16 Convención de nomenclatura para variable

- **Clases:** se rigen por la nomenclatura *camelCase*. Siempre comienzan con mayúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Ver ilustración 17 y 18.

```
12      class Alerta
13      {...}
138
```

Ilustración 17 Convención de nomenclatura para clase simple

```
10      class AlertaController extends Controller
11      {...}
68
```

Ilustración 18 Convención de nomenclatura para clase compuesta

- **Funciones:** se rigen por la nomenclatura *camelCase*. Siempre comienzan con minúscula y en caso de nombres compuestos la primera letra de cada palabra comienza con mayúscula. Los parámetros son separados por espacio luego de la coma que los separa. Ver ilustración 19.

```
102         public function modificarDirectivoAction($id, Request $request)
103         {...}
135
```

Ilustración 19 Convención de nomenclatura para función

Ficheros: todo siempre en minúscula y en caso de nombres compuestos se usa el guión bajo. Ver ilustración 20.

Ilustración 20 Convención de nomenclatura para fichero

Estructuras de control: las estructuras de control incluyen *if, for, foreach, while, switch*. Entre las estructuras de control y los paréntesis debe de existir un espacio. Se recomienda utilizar siempre llaves de apertura y cierre, incluso en situaciones en las que técnicamente son opcionales. Con esto se aumenta la legibilidad del código y se disminuye la probabilidad de errores lógicos. Si las condiciones son muy largas que sobrepasan el tamaño de la línea, estas se dividen en varias líneas, en el mejor de los casos cuando la condición es muy extensa, se puede dividir en variables y compararlas dentro de la estructura de control. Ver ilustración 21.

```
145             if (!empty($persona)) {
146                 foreach ($persona as $objP) {
147                     if ($objP->getId() != $datos['id_persona']) {
148                         $bandera = true;
149                     }
150                 }
151             }
152             else {
153                 $bandera = false;
154             }
```

Ilustración 21 Estructuras de control

Documentación: todos los archivos deben de tener la documentación asociada. Para esto debe de cumplir con el siguiente bloque al principio de cada clase. Ver ilustración 22 y 23.

```
3   /**
4    * Clase controladora Fuente.
5    * Esta clase se encarga de la logica asociada a las fuentes.
6    * @package SIMAE V 2.0
7    * @category Controller
8    * @author Alfredo L. Santisteban Céspedes <alsantisteban@estudiantes.uci.cu>
9    */
10  namespace Publicacion\ComunBundle\Controller;
11
12  use Symfony\Bundle\FrameworkBundle\Controller\Controller;
13  use Symfony\Component\HttpFoundation\Request;
14  use Publicacion\ComunBundle\Entity\tbnfuente;
15  use Publicacion\ComunBundle\Form\tbnfuenteType;
16  use Symfony\Component\HttpFoundation\Response;
17
18  class FuenteController extends Controller
```

Ilustración 22 Documentación de clase

```
29      /**
30       * Obtiene las fuentes
31       * @author Alfredo L. Santisteban Céspedes <alsantisteban@estudiantes.uci.cu>
32       * @return array()
33       */
34      public function obtenerFuentesAction(Request $request)
35      {...}
```

Ilustración 23 Documentación de método

Buenas prácticas: los valores booleanos y nulos siempre se escriben con mayúscula, para facilitar la legibilidad del código. Se debe usar un "*enter*" antes de las estructuras de control y definición de las funciones.

3.2 Pruebas de *software*

Las pruebas de *software*, se encargan de verificar la calidad de un sistema o componente que es ejecutado bajo condiciones o requisitos especificados previamente. Tienen el objetivo de encontrar y documentar errores, garantizando que el producto final funcione como fue diseñado y efectúe correctamente los requerimientos identificados. Entre las pruebas a realizar se encuentran las pruebas funcionales, de seguridad y de carga y estrés.

3.2.1 Pruebas funcionales

Dentro de las pruebas funcionales se hicieron pruebas de caja negra, cuyo objetivo es verificar desde la interfaz de usuario que dada una determinada entrada el sistema responda como está dispuesto en su concepción.

Condiciones de ejecución: el sistema SIMAE v2.0 debe estar disponible, un usuario con el rol de administrador debe estar autenticado.

Resultados de las pruebas funcionales

Se aplicaron los casos de prueba (ver anexo 6) correspondientes a las funcionalidades del sistema SIMAE v2.0, para ello se realizaron tres iteraciones y se obtuvo un total de 170 no conformidades que fueron resueltas en su totalidad. De las no conformidades, 29 fueron de no correspondencia con la documentación pues en la ejecución de varias funcionalidades no se obtenían los resultados esperados; 102 correspondientes a campos que no validaban si los datos insertados podían ser nulos o si estaban escritos correctamente; 2 relacionadas con los mensajes que debe mostrar el sistema por la ocurrencia de errores; 20 debido a errores con el diseño del sistema; 2 error funcionales de la aplicación y el resto correspondían a nombres de menús, botones o campos con errores ortográficos o de redacción.

En la primera iteración se obtuvo un total de 152 no conformidades de las cuales se resolvieron 115, quedando pendientes 37. Se detectó que ninguna gráfica que genera la aplicación mostraba los datos correctos, de igual modo los reportes y los análisis estadísticos no mostraban los resultados esperados. No se validaba completar campos de carácter obligatorio cuando se creaba o modificaba una publicación como:

la fecha de publicación, la cantidad de visitas y el vínculo, un comentario (el tipo de comentario), una alerta (la causa), un sitio *web* (el orden), un usuario (el nombre completo), un directivo (el área) o un nomenclador (en caso de ser de tipo área: dimensión y de tipo temática: el área).

Los campos correspondientes al correo, vínculo, URL y la fecha no validaban que estuvieran escritos correctamente. El sistema aceptaba insertar el mismo redactor y el mismo directivo varias veces. También se detectó que se podían insertar fechas posteriores a la actual. En esta iteración quedó pendiente que elementos visuales del sistema como campos para la entrada de datos, botones y etiquetas estaban distorsionados o desorganizados, corregir los errores funcionales referentes a la generación de reportes y que el sistema no permitía adjuntar una imagen correctamente. Las gráficas correspondientes a las fuentes más utilizadas y cantidad de alertas no retornaban los datos correctos. Los reportes de seguimiento, de exportar palabras claves de publicaciones, de exportar palabras claves de comentarios no estaban funcionales, de igual modo los análisis estadísticos

En la segunda iteración se obtuvieron 17 no conformidades además de las 37 pendientes de la iteración anterior, se resolvieron 40 y quedaron pendientes 14. Se detectó que cuando se insertaba o modificaba una publicación o un comentario el sistema guardaba correctamente los cambios pero mostraba un mensaje señalando la ocurrencia de un error. Se percibió la existencia de errores ortográficos y gramaticales en varias etiquetas, menús y submenús. En esta iteración quedó pendiente que el sistema no permitía configurar correctamente el modo de autenticación. También se constató que no se validaba el tamaño máximo permitido para un archivo adjunto correspondiente a la imagen de un sitio *web*, el cual no debe exceder de 2 MB. Los reportes generados no mostraban los resultados esperados y corregir el formato de algunos campos.

Se realizó una tercera iteración en la cual se detectaron 15 no conformidades, incluidas las 14 que habían quedado pendientes de la iteración anterior. Se comprobó que el sistema no estaba generando correctamente las trazas. Todas las no conformidades fueron resultas validando la calidad del sistema con respecto a las funcionalidades que realiza.

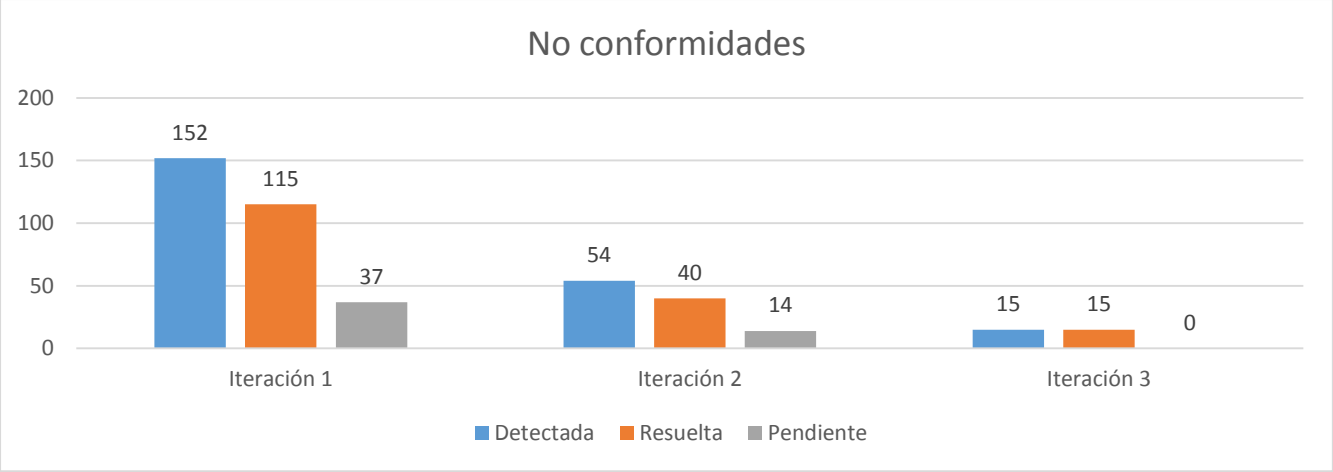


Ilustración 24 Gráfico estadístico de las no conformidades obtenidas en las pruebas funcionales

3.2.2 Pruebas de seguridad

En un sistema informático la seguridad es un aspecto fundamental pues evita que los datos puedan ser manipulados por personas sin previa autorización y comprometiendo su integridad. Por tal motivo es necesario realizar pruebas de seguridad de forma que se comprueben los niveles de acceso de los usuarios a las distintas funcionalidades del sistema. Para realizar las pruebas de seguridad se utilizó la aplicación *Websecurify* en su versión 0.8, que permite encontrar vulnerabilidades en sitios *web*.

Resultados de las pruebas

La siguiente gráfica se muestra los resultados obtenidos:

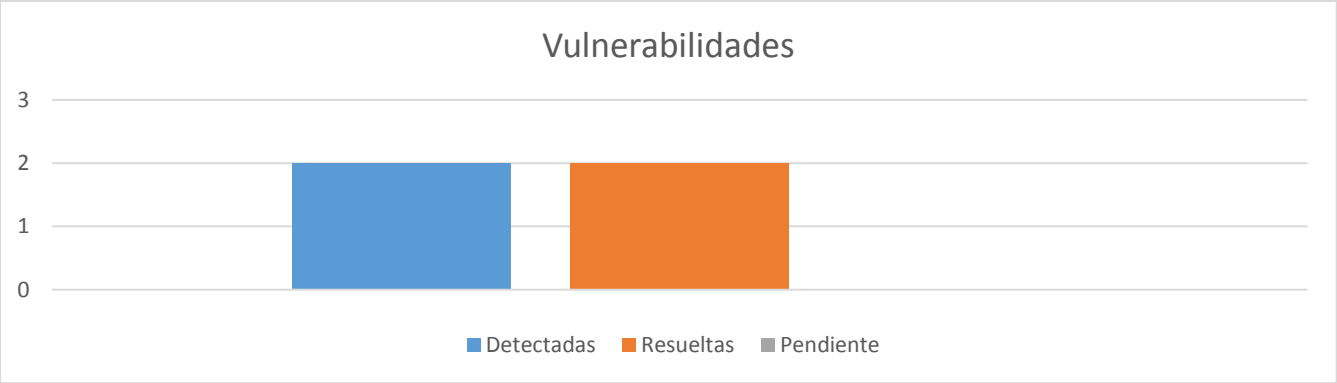


Ilustración 25 Gráfico estadístico de las vulnerabilidades obtenidas en las pruebas de seguridad

Se detectó que el sistema es vulnerable a un acceso no autorizado, el cual consiste en que un usuario malicioso puede llamar a cualquier controlador de la aplicación mediante la ruta /_fragment pasando un *hash* inválido en la url (o incluso eliminándolo). Esto posibilita que los sistemas de seguridad que validan *hash* no se ejecuten y por tanto se permita el acceso no autorizado. Además se descubrió que si un atacante envía una petición con un *host* formado por una cadena extensa, podría producirse un ataque de tipo denegación de servicio.

También se validó que cada rol tuviera los permisos asignados correctamente, o sea, los usuarios solo pueden realizar modificaciones en los elementos correspondientes al rol al que pertenecen. Un usuario que no pertenezca al rol de administrador no puede modificar sus permisos ni los de otros usuarios.

Por otra parte se comprobó que al cerrar el navegador o cerrar la sesión de usuario en la aplicación, no se pueda acceder sin autenticarse previamente a la sesión que se encontraba abierta. Se probó en distintos navegadores y puestos de trabajo que al cerrar el navegador y volverlo a abrir el sistema no permite acceder automáticamente a la sesión que estaba abierta antes de cerrar.

Al culminar las pruebas se solucionaron las vulnerabilidades encontradas evitando que puedan ser utilizadas por un atacante para comprometer el sistema. Se comprobó que el sistema no muestra ningún mensaje en el cual se brinda información que pueda ser utilizada para intentar acceder de forma no autorizada. Se validó que el sistema cumple con la seguridad necesaria para proteger la información contenida.

3.2.3 Pruebas de carga y estrés

Estas pruebas son realizadas con el objetivo de observar el comportamiento del sistema al recibir un número determinado de peticiones, de ellas se obtienen los tiempos de respuesta a las distintas peticiones. Además evalúa cómo el sistema responde bajo condiciones atípicas. Para estas pruebas se empleó la aplicación *JMeter* en su versión 2.3.1, que permite realizar pruebas de carga, medición de rendimiento y resistencia.

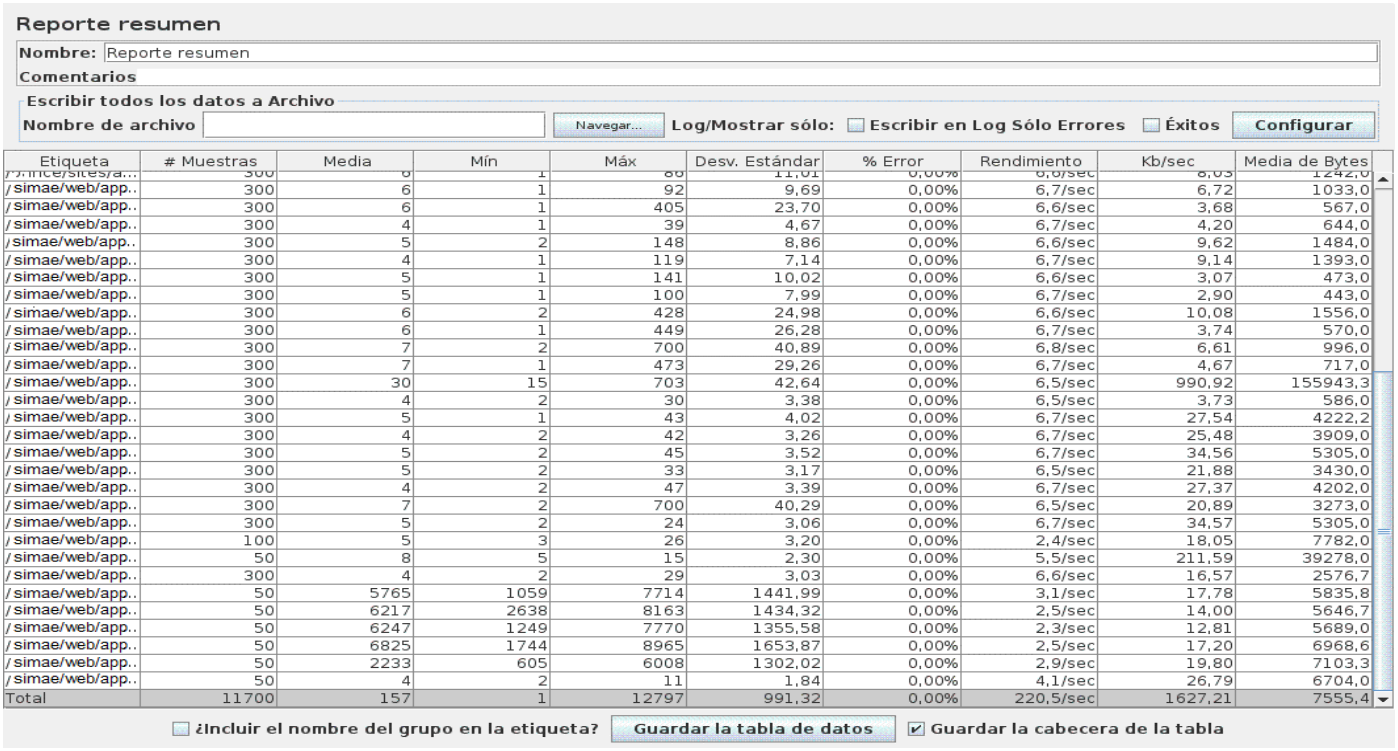
El sistema sobre el cual se realizaron las pruebas cumple con las siguientes especificaciones:

Procesador: *Intel Pentium Processor* G2020T (3MB cache, 2.50 GHz)

Memoria RAM: 2 GB

Resultados de las pruebas de carga y estrés

Las pruebas fueron realizadas con 50 y 100 hilos simulando la cantidad de usuarios realizando peticiones concurrentes al sistema. La siguiente tabla constituye el reporte generado por la herramienta *JMeter* para una muestra de 50 usuarios conectados, con un período de subida de 1 segundo:



Reporte resumen

Nombre: Reporte resumen

Comentarios

Escribir todos los datos a Archivo

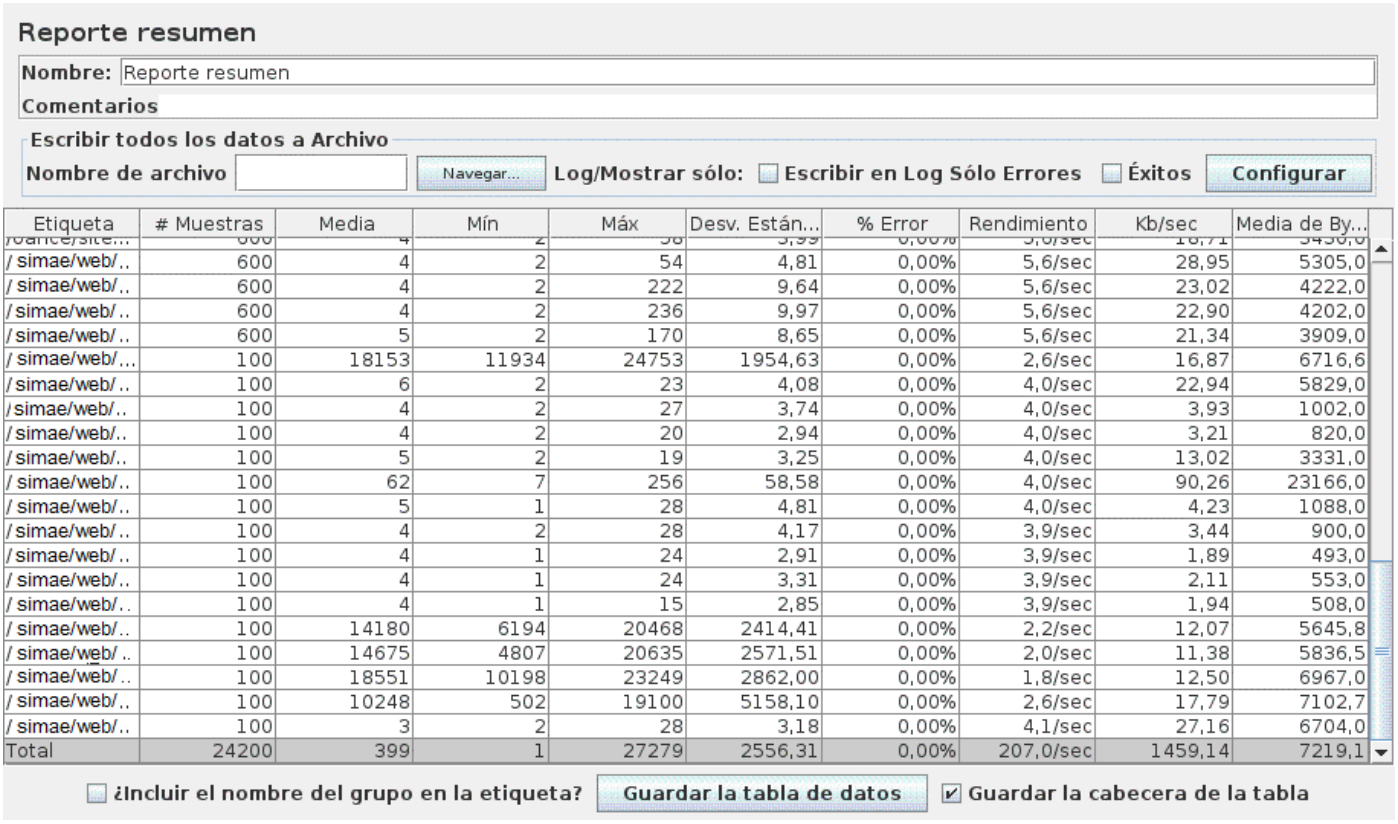
Nombre de archivo | Navegar... | Log/Mostrar sólo: Escribir en Log Sólo Errores | Éxitos | Configurar

Etiqueta	# Muestras	Media	Mín	Máx	Desv. Estándar	% Error	Rendimiento	Kb/sec	Media de Bytes
/simae/web/app..	300	6	1	92	9,69	0,00%	6,7/sec	6,72	1033,0
/simae/web/app..	300	6	1	405	23,70	0,00%	6,6/sec	3,68	567,0
/simae/web/app..	300	4	1	39	4,67	0,00%	6,7/sec	4,20	644,0
/simae/web/app..	300	5	2	148	8,86	0,00%	6,6/sec	9,62	1484,0
/simae/web/app..	300	4	1	119	7,14	0,00%	6,7/sec	9,14	1393,0
/simae/web/app..	300	5	1	141	10,02	0,00%	6,6/sec	3,07	473,0
/simae/web/app..	300	5	1	100	7,99	0,00%	6,7/sec	2,90	443,0
/simae/web/app..	300	6	2	428	24,98	0,00%	6,6/sec	10,08	1556,0
/simae/web/app..	300	6	1	449	26,28	0,00%	6,7/sec	3,74	570,0
/simae/web/app..	300	7	2	700	40,89	0,00%	6,8/sec	6,61	996,0
/simae/web/app..	300	7	1	473	29,26	0,00%	6,7/sec	4,67	717,0
/simae/web/app..	300	30	15	703	42,64	0,00%	6,5/sec	990,92	155943,3
/simae/web/app..	300	4	2	30	3,38	0,00%	6,5/sec	3,73	586,0
/simae/web/app..	300	5	1	43	4,02	0,00%	6,7/sec	27,54	4222,2
/simae/web/app..	300	4	2	42	3,26	0,00%	6,7/sec	25,48	3909,0
/simae/web/app..	300	5	2	45	3,52	0,00%	6,7/sec	34,56	5305,0
/simae/web/app..	300	5	2	33	3,17	0,00%	6,5/sec	21,88	3430,0
/simae/web/app..	300	4	2	47	3,39	0,00%	6,7/sec	27,37	4202,0
/simae/web/app..	300	7	2	700	40,29	0,00%	6,5/sec	20,89	3273,0
/simae/web/app..	300	5	2	24	3,06	0,00%	6,7/sec	34,57	5305,0
/simae/web/app..	100	5	3	26	3,20	0,00%	2,4/sec	18,05	7782,0
/simae/web/app..	50	8	5	15	2,30	0,00%	5,5/sec	211,59	39278,0
/simae/web/app..	300	4	2	29	3,03	0,00%	6,6/sec	16,57	2576,7
/simae/web/app..	50	5765	1059	7714	1441,99	0,00%	3,1/sec	17,78	5835,8
/simae/web/app..	50	6217	2638	8163	1434,32	0,00%	2,5/sec	14,00	5646,7
/simae/web/app..	50	6247	1249	7770	1355,58	0,00%	2,3/sec	12,81	5689,0
/simae/web/app..	50	6825	1744	8965	1653,87	0,00%	2,5/sec	17,20	6968,6
/simae/web/app..	50	2233	605	6008	1302,02	0,00%	2,9/sec	19,80	7103,3
/simae/web/app..	50	4	2	11	1,84	0,00%	4,1/sec	26,79	6704,0
Total	11700	157	1	12797	991,32	0,00%	220,5/sec	1627,21	7555,4

¿Incluir el nombre del grupo en la etiqueta? | Guardar la tabla de datos | Guardar la cabecera de la tabla

Ilustración 26 Resultado de la aplicación JMeter para 50 usuarios

La siguiente tabla constituye el reporte generado para una muestra de 100 usuarios, con un período de subida de 1 segundo:



Etiqueta	# Muestras	Media	Mín	Máx	Desv. Están...	% Error	Rendimiento	Kb/sec	Media de By...
/simae/web/..	600	4	2	54	4,81	0,00%	5,6/sec	28,95	5305,0
/simae/web/..	600	4	2	222	9,64	0,00%	5,6/sec	23,02	4222,0
/simae/web/..	600	4	2	236	9,97	0,00%	5,6/sec	22,90	4202,0
/simae/web/..	600	5	2	170	8,65	0,00%	5,6/sec	21,34	3909,0
/simae/web/...	100	18153	11934	24753	1954,63	0,00%	2,6/sec	16,87	6716,6
/simae/web/..	100	6	2	23	4,08	0,00%	4,0/sec	22,94	5829,0
/simae/web/..	100	4	2	27	3,74	0,00%	4,0/sec	3,93	1002,0
/simae/web/..	100	4	2	20	2,94	0,00%	4,0/sec	3,21	820,0
/simae/web/..	100	5	2	19	3,25	0,00%	4,0/sec	13,02	3331,0
/simae/web/..	100	62	7	256	58,58	0,00%	4,0/sec	90,26	23166,0
/simae/web/..	100	5	1	28	4,81	0,00%	4,0/sec	4,23	1088,0
/simae/web/..	100	4	2	28	4,17	0,00%	3,9/sec	3,44	900,0
/simae/web/..	100	4	1	24	2,91	0,00%	3,9/sec	1,89	493,0
/simae/web/..	100	4	1	24	3,31	0,00%	3,9/sec	2,11	553,0
/simae/web/..	100	4	1	15	2,85	0,00%	3,9/sec	1,94	508,0
/simae/web/..	100	14180	6194	20468	2414,41	0,00%	2,2/sec	12,07	5645,8
/simae/web/..	100	14675	4807	20635	2571,51	0,00%	2,0/sec	11,38	5836,5
/simae/web/..	100	18551	10198	23249	2862,00	0,00%	1,8/sec	12,50	6967,0
/simae/web/..	100	10248	502	19100	5158,10	0,00%	2,6/sec	17,79	7102,7
/simae/web/..	100	3	2	28	3,18	0,00%	4,1/sec	27,16	6704,0
Total	24200	399	1	27279	2556,31	0,00%	207,0/sec	1459,14	7219,1

Ilustración 27 Resultado de la aplicación JMeter para 100 usuarios

Para el caso de 50 usuarios conectados realizando peticiones se realizaron un total de 11 700 peticiones con un rendimiento de 220.5 peticiones por segundo; para 100 usuarios se realizaron un total 24 200 peticiones con un rendimiento de 207.0 peticiones por segundo. En ambos casos la ocurrencia de errores fue nula para cada petición realizada. Los resultados obtenidos en las pruebas realizadas permitieron constatar que el sistema es capaz de responder de forma correcta ante situaciones atípicas de carga y estrés.

3.3 Conclusiones parciales

El empleo de los estándares de codificación permitió mantener una estructura y organización en el desarrollo de la aplicación. La utilización de casos de prueba permitió garantizar que las funcionalidades estuvieran correctamente implementadas y que respondieran de la forma esperada. Las pruebas de seguridad y de carga y estrés permitieron certificar que la aplicación no evidencia vulnerabilidades que puedan ser

aprovechadas por terceros para uso no deseado y que la solución es capaz de responder correctamente ante situaciones atípicas en las que se realizan un gran número de peticiones en un período corto de tiempo.

Conclusiones generales

- El estudio de conceptos relacionados con la recopilación y análisis de información de publicaciones en sitios *web*, facilitó una mejor comprensión de los procesos que realiza el sistema.
- La investigación concerniente a las herramientas que realizan monitoreo y análisis de información en sitios *web*, permitió seleccionar las principales características que sirvieron de base para la incorporación de nuevas funcionalidades y mejoras en el sistema SIMAE v2.0.
- El estudio profundo de la metodología, las tecnologías y las herramientas permitió guiar correctamente el proceso de desarrollo de *software*.
- El análisis y diseño de los procesos garantizaron la correcta implementación de las funcionalidades del sistema SIMAE v2.0.
- La aplicación de pruebas (funcionales, de seguridad, carga y estrés) permitieron validar el funcionamiento y calidad del sistema en los procesos de recopilación y análisis de información.

Recomendaciones

- Incorporar al sistema SIMAE la clasificación automática de contenidos.
- Actualizar el *framework* utilizado a la versión 2.7.
- Utilizar la documentación del presente trabajo como material de estudio en futuras investigaciones.
- Fomentar la utilización del sistema SIMAE v2.0 en empresas e instituciones.

Referencias bibliográficas

1. **Chiavenato, Idalberto.** *Introducción a la Teoría General de la Administración.* s.l. : McGraw-Hill Interamericana, 2006.

2. **Ferrell, O. C. y Hirt, Geoffrey.** *Introducción a los Negocios en un Mundo Cambiante.* s.l. : McGraw-Hill Interamericana, 2004.

3. **Czinkota, Michael y Kotabe, Masaaki.** *Administración de Mercadotecnia .* s.l. : International Thomson Editores, 2001.

4. Información: Diccionario de la Real Academia Española. *Diccionario de la Real Academia Española.* [En línea] 2015. http://buscon.rae.es/draeI/SrvltConsulta?TIPO_BUS=3&LEMA=informaci%C3%B3n.

5. **Dulzaides Iglesias, María Elinor y Molina Gómez, Ana María.** Scielo: Análisis documental y de información: dos componentes de un mismo proceso. *Scielo.* [En línea] 2004. [Citado el: 18 de Mayo de 2015.] http://scielo.sld.cu/scielo.php?pid=S1024-94352004000200011&script=sci_arttext. ISSN 1024-9435.

6. **Vidal Ledo, María Josefina y Araña Pérez, Ana Bárbara.** Educación Médica Superior. [En línea] 15 de Marzo de 2012. www.ems.sld.cu/index.php/ems/article/view/56/46.

7. **Rojas Mesa, Yuniet.** *ACIMED.* [En línea] 2006. http://scielo.sld.cu/scielo.php?script=sci_arttext&pid=S1024-94352006000100002&lng=es&nrm=iso. ISSN 1024-9435.

8. **Lara Mas, Elvin y Herrera Almaguer, Dayana.** *Sistema de Gestión Académica para la Vice Dirección de Docencia e Investigaciones del Policlínico – Docente Ernesto Che Guevara.* 2010.

9. **Travieso Aguilar, Mayelín.** Las Publicaciones electrónicas: una revolución en el siglo XXI. *Acimed.* [En línea] 25 de 1 de 2003. http://www.bvs.sld.cu/revistas/aci/vol11_2_03/aci010203.htm.

10. Gestores bibliográficos: Sistema de Bibliotecas de la Universidad de Antioquia. *Sistema de Bibliotecas de la Universidad de Antioquia.* [En línea] 2011.

11. WooRank. *WooRank.* [En línea] 2015. https://www.woorank.com/es/.

12. Funciones: Google Analytics. *Google Analytics.* [En línea] http://www.google.com/intl/es_ALL/analytics/features/index.html.

13. About: twXplorer. *twXplorer.* [En línea] 2013. http://twxplorer.knightlab.com/about/.

14. *FAIA: Framework para la enseñanza de agente en IA.* **Roa, Jorge, Gutiérrez, Milagro y Stegmayer, Georgina.** 7/8, Santa Fe, Argentina : s.n., 2008. 1699-4574.

15. **Potencier, Fabien y Zaninotto, Francois.** Libros*web*: Symfony, la guia definitiva. *Librosweb.* [En línea] 2015. http://libros*web*.es/libro/symfony_1_1/capitulo_1/symfony_en_pocas_palabras.html.

16. **Eguiluz, Javier.** *Desarrollo web ágil con Symfony 2.* 2013.

17. Noticias: Symfony.es. *Symfony.es.* [En línea] 2015. http://symfony.es/noticias/2008/04/28/10-razones-por-las-que-tu-empresa-deberia-utilizar-symfony.

18. CodeIgniter User Guide Version 2.2.0: CodeIgniter. *CodeIgniter.* [En línea] 2014. https://ellislab.com/codeigniter/user-guide/overview/at_a_glance.html.

19. **Vázquez Mariño, Carlos.** *Programación en PHP 5. Nivel Básico.* Ferrol : s.n., 2008.

20. **Gauchat, Juan Diego.** *El gran libro de HTML5, CSS3 y Javascript.* España : MARCOMBO, 2012. ISBN eBook: 978-84-267-1782-5.

21. **Raquel Quispillo, Mariana y Monserrat, Paulina.** *Análisis Cualitativo y Cuantitativo de Herramientas de Entorno Visual para desarrollo web en PHP aplicado a la EPEC.* Riobamba,Ecuador : s.n., 2009.

22. **Álvarez, Miguel Angel.** Manual de JQuery: Desarrollo*web*. *Desarrolloweb.* [En línea] 2015. http://www.desarrollo*web*.com/manuales/manual-jquery.html.

23. NetBeans IDE Features: NetBeans. *NetBeans.* [En línea] 2015. https://netbeans.org/features/index.html.

24. PhpStorm Features: PhpStorm. *PhpStorm.* [En línea] 2015. https://www.jetbrains.com/phpstorm/features/.

25. **Gil, Fidel, Albrigo, Javier y Do Rosario , Javier.** *Sistemas de Gestión de Base de Datos SGBD/DBMS.* Valencia : s.n., 2005.

26. About MySQL: MySQL. *MySQL.* [En línea] 2015. http://www.mysql.com/about.

27. About: PostgreSQL. *PostgreSQL.* [En línea] 2015. http://www.postgresql.org/about/.

28. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* Madrid : Addison Wesley, 2000. 8478290370.

29. *Evaluación comparativa de herramientas CASE para UML desde el punto de vista notacional.* **Génova Fuster, Gonzalo, Fuentes Torres, José Miguel y Valiente Vázquez, María Cruz.** 181, España : s.n., 2006. 0211-2124.

30. **Sierra, María.** Visual Paradigm: Scribd. *Scribd.* [En línea] 2015.

31. Introduction to OpenUP: OpenUP. *OpenUP.* [En línea] 2011. http://epf.eclipse.org/wikis/openup/.

32. **Letelier, Patricio y Penadés, Carmen.** *Métodologías ágiles para el desarrollo de software: Extreme .* Valencia : s.n., 2006.

33. Overview of new features in Apache 2.2: The Apache Software Foundation. *The Apache Software Foundation.* [En línea] 2015. http://httpd.apache.org/docs/2.2/new_features_2_0.html.

34. Highcharts product: Highcharts. *Highcharts.* [En línea] 2015. http://www.highcharts.com/products/highcharts.

35. Chart.js. *Chart.js.* [En línea] 2015. http://www.chartjs.org.

36. GitHub. *GitHub.* [En línea] 2015. https://github.com/flot/flot/blob/master/README.md.

37. TCPDF. *TCPDF.* [En línea] 2013. http://www.tcpdf.org.

38. FPDF Library. *FPDF Library.* [En línea] 2015. http://www.fpdf.org.

39. **Larman, Craig.** *UML y Patrones. Segunda Edición.* s.l. : Prentice Hall.

40. **Sommerville, Ian.** *Ingeniería del software. Septima edición.* Madrid : PEARSON EDUCACIÓN, 2005. ISBN: 84-7829-074-5.

41. **Kendall, Kenneth E. y Kendall, Julie E.** *Análisis y diseño de sistemas. Sexta edición.* México : PEARSON EDUCACIÓN , 2005. ISBN: 970-26-0577-6.

42. **Bahit, Eugenia.** *POO y MVC en PHP.* 2011.

43. **Welicki, León.** Patrones y antipatrones: una Introducción. *Microsoft Developer Network.* [En línea] 2015. http://msdn.microsoft.com/es-es/library/bb972251.aspx#M19.

44. **Olivares Rojas, Juan Carlos.** *Patrones de Diseño.* México: Instituto Tecnológico de México : s.n., 2009.

45. **Pressman, R. S.** *Software Engineering, a practitioner's approach.* 2007.

Bibliografía

1. **Álvarez, Miguel Angel.** Manual de JQuery: Desarrollo*web*. *Desarrolloweb.* [En línea] 2015. http://www.desarrollo*web*.com/manuales/manual-*JQuery*.html.

2. **Bahit, Eugenia.** *POO y MVC en PHP*. 2011.

3. **Eguiluz, Javier.** *Desarrollo web ágil con Symfony 2*. 2013.

4. **Gauchat, Juan Diego.** *El gran libro de HTML5, CSS3 y JavaScript.* España : MARCOMBO, 2012. ISBN eBook: 978-84-267-1782-5.

5. **Gil, Fidel, Albrigo, Javier y Do Rosario, Javier.** *Sistemas de Gestión de Base de Datos SGBD/DBMS.* Valencia : s.n., 2005.

6. **Kendall, Kenneth E. y Kendall, Julie E.** *Análisis y diseño de sistemas. Sexta edición.* México : PEARSON EDUCACIÓN , 2005. ISBN: 970-26-0577-6.

7. **Larman, Craig.** *UML y Patrones. Segunda Edición.* s.l. : Prentice Hall.

8. **Letelier, Patricio y Penadés, Carmen.** *Metodologías ágiles para el desarrollo de software: Extreme.* Valencia : s.n., 2006.

9. **Maestras, Juan Pavón.** *Estructura de las Aplicaciones Orientadas a Objetos. El patrón Modelo Vista.* Madrid: Universidad Complutense de Madrid Dpto. Ingeniería del Software e Inteligencia Artificial : s.n.

10. **Olivares Rojas, Juan Carlos.** *Patrones de Diseño.* México: Instituto Tecnológico de México : s.n., 2009.

11. **Potencier, Fabien.** *Symfony, la guía definitiva*. 2008.

12. **Pressman, R. S.** *Software Engineering, a practitioner's approach.* 2007.

13. **Ruiz, Javier Heredia.** *Comparación y tendencias entre metodologías ágiles y formales.* s.l. : Ceige, 2011. ISSN | RNPS.

14. **Rumbaugh, James, Jacobson, Ivar y Booch, Grady.** *El Lenguaje Unificado de Modelado. Manual de Referencia.* Madrid : Addison Wesley, 2000. 8478290370.

15. **Sommerville, Ian.** *Ingeniería del software. Septima edición.* Madrid : PEARSON EDUCACIÓN, 2005. ISBN: 84-7829-074-5.

16. **Vázquez Mariño, Carlos.** *Programación en PHP 5. Nivel Básico.* Ferrol : s.n., 2008.

Glosario de términos

ACID: En bases de datos se denomina ACID a un conjunto de características necesarias para que una serie de instrucciones puedan ser consideradas como una transacción.

Alexa: Alexa *Internet*, Inc. es una subsidiaria de la compañía Amazon.com con base en California. Es conocida por operar el sitio *web* alexa.com que provee información acerca de la cantidad de visitas que recibe un sitio *web* y los clasifica en un *ranking*.

API: La interfaz de programación de aplicaciones, abreviada como API por sus siglas en inglés (*Application Programming Interface*), es el conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.

Bundle: Un *bundle* es un concepto similar al de plugins en aplicaciones informáticas, incluye tanto las funcionalidades básicas de la plataforma como el código escrito para una aplicación.

Canvas: (lienzo en español) es un elemento HTML incorporado en HTML5 que permite la generación de gráficos dinámicamente por medio del *Scripting*. Permite generar gráficos estáticos y animaciones.

Composer: es un gestor de paquetes a nivel de aplicación para el lenguaje de programación PHP que ofrece un formato estándar para la gestión de dependencias de *software* PHP y bibliotecas necesarias.

CSRF: (del inglés *Cross-site request forgery* o falsificación de petición en sitios cruzados) 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. Esta vulnerabilidad es conocida también por otros nombres como XSRF, enlace hostil, ataque de un click, cabalgamiento de sesión, y ataque automático.

***Doctrine*:** es un mapeador de objetos-relacional (ORM) escrito en PHP que proporciona una capa de persistencia para objetos PHP. Es una capa de abstracción que se sitúa justo encima de un SGBD (sistema de gestión de bases de datos).

DOM o *Document Object Model*: ('Modelo de Objetos del Documento' o 'Modelo en Objetos para la Representación de Documentos') es esencialmente una interfaz de programación de aplicaciones (API) que proporciona un conjunto estándar de objetos para representar documentos HTML y XML, un modelo estándar sobre cómo pueden combinarse dichos objetos, y una interfaz estándar para acceder a ellos y manipularlos.

Engineering-CASE: es un tipo de ingeniería de *software* en la que se intenta aumentar la eficacia de sus procesos, al soportar la realización de las tareas con el uso de tecnologías.

***Firefox*:** es un navegador *web* libre y de código abierto desarrollado para *Microsoft Windows*, *Android*, OS X y GNU/*Linux* coordinado por la Corporación Mozilla y la Fundación Mozilla.

***Google*:** es una empresa multinacional estadounidense especializada en servicios y productos relacionados con *software*, i*nternet*, dispositivos electrónicos y otras tecnologías, cuyo producto principal es un motor de búsqueda.

***Hashtags*:** es una etiqueta de metadatos precedida por una almohadilla o numeral (#) con el fin de que, tanto el sistema como el usuario, la identifiquen de forma rápida. Se usa en servicios *web* tales como *Twitter*, *FriendFeed*, *Facebook*, *Google+*, *Instagram* o en mensajería basada en protocolos IRC para señalar un tema sobre el que gira cierta conversación.

Indexada: es el nombre que se le da al proceso de recolectar y almacenar páginas *web* por parte de un buscador de *internet*.

***Plugins*:** Es aquella aplicación que, en un programa informático, añade una funcionalidad adicional o una nueva característica al *software*. Se le conoce además como complemento.

RAD o desarrollo rápido de aplicaciones: (acrónimo en inglés de *Rapid Application Development*) es un proceso de desarrollo de *software*. El método comprende el desarrollo interactivo, la construcción de prototipos y el uso de utilidades CASE (ingeniería asistida por computadora). Tradicionalmente, el desarrollo rápido de aplicaciones tiende a englobar también la usabilidad, utilidad y la rapidez de ejecución.

***Retweets*:** Los propios usuarios de *Twitter* inventaron algo que se dio a conocer como “hacer *retweet*”.O sea que, consiste en decir mediante tu usuario lo que ha comentado otro usuario, a partir de que se considere interesante.

SOAP: (siglas de *Simple Object Access Protocol*) es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML.

***Software*:** Según la Real Academia Española (RAE), es un conjunto de programas, instrucciones y reglas informáticas que permiten ejecutar distintas tareas en una computadora. Se considera que el *software* es el equipamiento lógico e intangible de un ordenador.

SVG (Gráficos Vectoriales Redimensionables): son una especificación para describir gráficos vectoriales bidimensionales, tanto estáticos como animados (estos últimos con ayuda de SMIL), en formato XML.

***Twitter*:** un término inglés que puede traducirse como "gorjear" o "trinar", es el nombre de una red de *microblogging* (es una variante de los *blogs*) que permite escribir y leer mensajes en *internet* que no superen los 140 caracteres.

***Webmaster*:** Es conocido también como arquitecto *web*, desarrollador *web*, autor del sitio digital, administrador del sitio digital. Es la persona responsable de mantenimiento y/o programación de un sitio *web*.

***Web* of Science de ISI:** es un servicio en línea de información científica, integrado en ISI *Web of Knowledge* (WoK). Facilita el acceso a un conjunto de bases de datos en las que aparecen citas de artículos de revistas científicas, libros y otros tipos de material impreso que abarcan todos los campos del conocimiento académico.

XLIFF: (XML *Localization Interchange File Format*) es un formato basado en XML creado para estandarizar localización.

XSS: (del inglés *Cross-site Scripting*) es un tipo de inseguridad informática o agujero de seguridad típico de las aplicaciones *Web*, que permite a una tercera parte inyectar en páginas *web* visitadas por el usuario código *JavaScript* o en otro lenguaje *Script* similar (ej: *VBScript*), evitando medidas de control como la Política del mismo origen. Este tipo de vulnerabilidad se conoce en español con el nombre de Secuencias de órdenes en sitios cruzados.