Universidad de las Ciencias Informáticas

Facultad 1

“Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper”

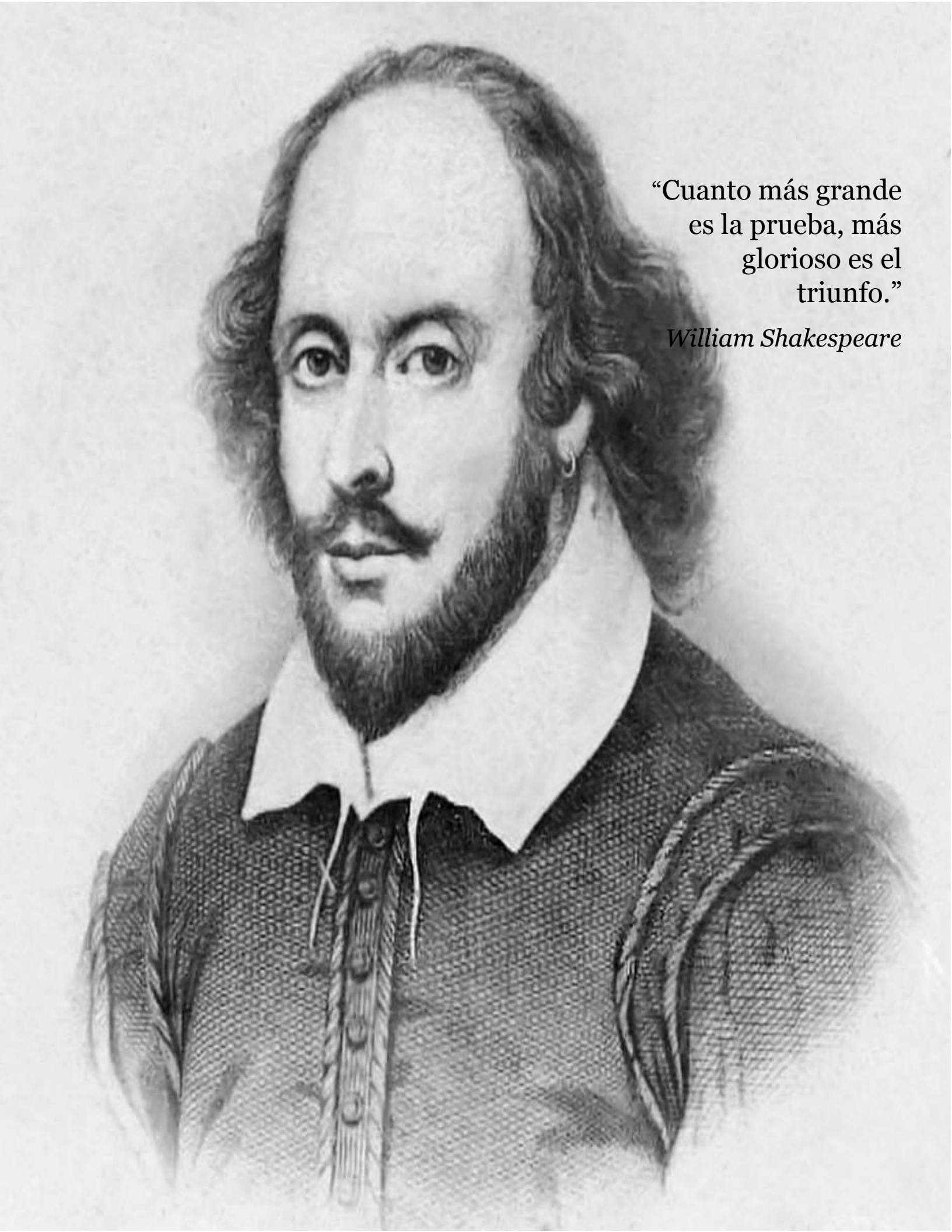
Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autora: Laura Gregores Nápoles

Tutores: Ing. Yojahny Sánchez Marrero

MSc. Graciela González Pérez

La Habana, junio 2018



"Cuanto más grande es la prueba, más glorioso es el triunfo."

William Shakespeare

Declaración de autoría

Declaro por este medio que yo, Laura Gregores Nápoles, con carnet de identidad 95092029534, soy la autora principal del trabajo titulado **"Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper"** y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Declaro que todo lo anteriormente expuesto se ajusta a la verdad, y asumo la responsabilidad moral y jurídica que se derive de este juramento profesional.

Para que así conste, firmo la presente declaración de autoría en La Habana a los ___ días del mes de ___ del año 2018.

Autor:

_____

Laura Gregores Nápoles

Tutores:

_____ _____

Ing: Yojahny Chávez Marrero MSc: Graciela González Pérez

A mis dos tesoros más grandes, mi madre

y mi abuela.

Agradecimientos:

A mis tutores, los consejos de la profe Graciela y a Yojahny por no dejarme sola y apoyarme en todo el proceso.

Al ingeniero Rubén Reynaldo Bonachea, por brindarme su apoyo en el momento más crítico de este proceso y a la profe Yordanka, gracias por su macabra estrategia de hacerme exponer frente a sus compañeros de laboratorios para combatir mi miedo escénico. A Yasmani, que en el momento final me apoyó y estuvo al tanto de los detalles, y cuando más a flor de piel estaban los nervios me dio muchísima confianza.

A las niñas del apartamento por estos años de convivencia.

A mis amistades del grupo y del año en general, quienes me han acompañado en esta travesía de 5 años, que sin dudas marcó mi vida para siempre.

A Grettel y Ada, que me demostraron que no se necesitan años para sentirnos esas amigas que llegan a ser hermanas, que han asumido la difícil tarea de soportarme, pero sobre todo que han creído en mí siempre, incluso cuando la confianza en mí misma ha fallado.

A mi tío Iván, mi gordito hermoso, por preocuparse siempre por mí y quererme tanto.

A mi abuelito Pedro que ya no está entre nosotros, no pudo esperar a ver este momento, pero donde quiera que esté sé que me ve y está orgulloso.

A Mima (mi madrina), quien ha estado cerca siempre velando por mí, queriéndome y apoyándome.

A mi tía Tamara, quien ha sido el típico ejemplo de que no solo los lazos de sangre hacen la familia, a pesar de la distancia siempre ha estado al pendiente.

A mis amigos Reinier y los dos Daniel, a Manoli, gracias bicho por estar siempre para mí y al pendiente.

Al Wisho, un amigo especial, una de las mejores personas que he conocido. Hay lazos que las millas no pueden romper, y no importa la distancia, de alguna manera hoy estás aquí.

A mi novio Yasmany, quien llegó a mi vida en medio de esta difícil etapa, pero ha estado a mi lado haciendo suyos también los tragos más amargos. Gracias por esa dedicación, por esas horas de charla y apoyo emocional, por la confianza en mí, este logro también es tuyo.

A mi abuela, quien ha jugado el papel de segunda madre en mi vida, gracias por esos resabios y esa "matraquilla" de –aprovecha el tiempo, que ya queda poco- que, aunque me desesperaron, me dieron el empujoncito hasta aquí.

A mi madre y mejor amiga, quien ha hecho de mi lo que hoy soy y me ha apoyado en cada decisión, gracias por tanta confianza y dedicación. Gracias por vivir cada pedacito de este camino conmigo de alegrías, amigos, exámenes con buenos resultados, pero los que no también, que fueron las que más dolores de cabeza te ocasionaron. Este logro también es tuyo, por lo que además de una estomatóloga excepcional se puede decir que también te estás graduando de ingeniera en ciencias informáticas junto conmigo.

Resumen:

Xilema Smart Keeper es un sistema de filtrado desarrollado en la Universidad de las Ciencias Informáticas para el monitoreo y control del tráfico web. Entre sus funcionalidades se encuentra generar reportes estadísticos a partir de la información tomada de los registros del *proxy* Squid. Los reportes generados por Xilema Smart Keeper no poseen una interfaz amigable, carecen de gráficas y tablas para una mejor comprensión del usuario administrador, así como requiere de profundos conocimientos de la estructura de base de datos del sistema para el análisis de la información.

La presente investigación tiene como objetivo desarrollar un módulo que permita generar de un modo comprensible e intuitivo los reportes estadísticos y facilitar el manejo de información, así como el análisis y la toma de decisiones. Se incluye un estudio de los sistemas generadores de reportes estadísticos de los eventos de la navegación en Internet más utilizados en la actualidad. El desarrollo estuvo guiado por la metodología AUP-UCI y se seleccionó como tecnologías PHP como lenguaje de programación, el marco de trabajo Symfony y como entorno integrado de desarrollo Netbeans. A partir de la aplicación de las pruebas de *software* de tipo funcional, integración y carga y estrés, se verificó el correcto funcionamiento y calidad del módulo.

Palabras clave: reportes estadísticos, navegación, logs, *proxy* Squid, Xilema Smart Keeper.

Índice de contenidos.

Índice de Tablas

Índice de Figuras

Introducción

Internet se ha convertido en un fenómeno tecnológico que impacta y transforma la cultura, la economía y la vida de quienes acceden a él. Su expansión, enmarcada en un contexto más general de desarrollo de tecnologías de información y de comunicación (TIC) está modificando la estructura y la dinámica del espacio mediático en las sociedades que está presente. En poco tiempo, ha adquirido una enorme importancia como espacio de circulación y contraste de información, discusión y debate público.

Sin dudas Internet ofrece infinitas posibilidades, pero también conlleva grandes riesgos. Todas las funcionalidades de Internet pueden implicar algún riesgo, al igual que ocurre en las actividades que realizamos en el "mundo físico". El estudio plasmado en (Ponce de León Leiva, 2016) destaca como principales riesgos: el acceso a información peligrosa e inmoral, la dispersión y pérdida de tiempo, la recepción de mensajes de propagandas que envían empresas de todo el mundo, que en ocasiones resultan de naturaleza sexual o proponen oscuros negocios, las adicciones a juegos, redes sociales y compras compulsivas y el acceso de menores a sitios inapropiados y sin autorización paterna.

Esta realidad junto con la necesidad de monitorear la navegación de usuarios, ha llevado al mercado sistemas de filtrado de accesos a Internet, para garantizar un uso seguro del Internet a los usuarios. Los sistemas de filtrado son aplicaciones que permiten bloquear el contenido no deseado de páginas Web, restringen el acceso a páginas de entretenimiento, compras, también bloquean los mensajeros instantáneos (programas para chatear o comunicarse con los amigos o la familia en línea), deniegan y evitan los programas de descarga.

Este tipo de sistemas operan a través del bloqueo de direcciones, controlando horas de acceso, utilizando listas de acceso o no acceso, asignando diferentes perfiles en diferentes días y horas (trabajo, tiempo libre, etc.), regulando qué servicios se pueden utilizar en cada momento y por cada usuario. Los sistemas de filtrado permiten además generar reportes estadísticos para una mejor compresión del consumo de estos contenidos y ayuda a la toma de decisiones.

La generación de reportes estadísticos en un sistema de filtrado se realiza a través de aplicaciones o subsistemas cuya funcionalidad es generar diversas estadísticas partiendo de la información rescatada en los logs de navegación, estadísticas basadas en la información factible que se obtiene a partir de los datos

almacenados. Tales reportes pueden ser las mayores descargas realizadas, sitios visitados, fallas de autenticación, entre otros (Dspace, 2017).

En la Universidad de las Ciencias Informáticas (UCI) desde 2005 se desarrolla un *software* de filtrado de contenidos de Internet denominado Xilema Smart Keeper. Para mostrar estadísticas de los eventos de navegación, actualmente, el sistema no genera gráficas, siendo este en ocasiones el medio más efectivo no sólo para describir y resumir la información, sino también para analizarla según la fuente (Hernández, 2017). Esta situación requiere por parte del administrador amplios conocimientos de la estructura de la base de datos del sistema para la interpretación de los datos y la toma de decisiones.

Xilema Smart Keeper se desarrolló basado en tecnologías que en la actualidad son obsoletas algunas y otras se les ha dejado de dar soporte. En el presente año se pretende llevar el sistema a su versión 3.0, por tanto, se hace necesario el desarrollo de un módulo para la generación de reportes estadísticos de los eventos realizados durante la navegación en Internet, que muestre los resultados mediante gráficas, utilizando tecnologías más actualizadas.

A partir de los planteamientos anteriores se establece el siguiente **problema de investigación:** ¿Cómo contribuir al proceso de generación de reportes estadísticos del sistema Xilema Smart Keeper?

Por lo que se formula la siguiente **idea a defender:** El desarrollo de un módulo que ejecute el proceso de generación de reportes estadísticos del sistema Xilema Smart Keeper mediante gráficas, facilitará el análisis de datos y toma de decisiones.

Se define como **objeto de estudio** el proceso de generación de reportes estadísticos y como **campo de acción**: el proceso de generación de reportes estadísticos mediante gráficas en el sistema Xilema Smart Keeper. **El objetivo general** derivado del problema de investigación es el siguiente: desarrollar un módulo que permita mejorar el proceso de generación de reportes estadísticos del sistema Xilema Smart Keeper. De esta forma se derivan los siguientes objetivos específicos:

- Sistematizar los referentes teóricos fundamentales que sustentan la investigación relacionados con el desarrollo de herramientas para la generación de reportes estadísticos en los *software* de filtrado de contenidos de Internet.
- Desarrollar el módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper.

- Validar las funcionalidades del módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper.

Se pretende obtener como **posibles resultados** el marco teórico referencial sobre la generación de reportes estadísticos en los *software* de filtrado de contenidos de Internet y un módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper.

Para lograr el cumplimiento de las tareas se utilizaron los siguientes métodos científicos:

Métodos teóricos:

Histórico-lógico: se llevó a cabo para comprender el funcionamiento de los filtros de contenidos de Internet y las herramientas para generar reportes estadísticos.

Analítico-sintético: permitió descomponer en diversas partes el conocimiento, durante la investigación sobre los mecanismos de generación de reportes, para un posterior estudio y arribo de conclusiones según los resultados.

Modelación: Se utiliza para realizar el modelado del sistema informático a través de los artefactos correspondientes a las fases: Modelo del Negocio, Requisitos, Análisis y Diseño e Implementación.

Métodos empíricos:

Observación: Posibilita obtener conocimiento acerca del funcionamiento de los sistemas existentes en la actualidad para la generación de reportes estadísticos.

Encuesta: Se utilizó para la recopilación de información en la validación de la idea a defender, por el método Criterio de Expertos.

La presente investigación está estructurada por: introducción, tres capítulos, conclusiones generales, recomendaciones y la bibliografía. La estructura de los capítulos obedece a la lógica que se describe a continuación:

Primer capítulo: "Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper, funcionalidades y tecnologías para su desarrollo", aborda un estudio del tema relacionado con el objeto de estudio definido. Se analiza la existencia de sistemas similares en el ámbito nacional e internacional que

puedan servir de apoyo. Se exponen los lenguajes, herramientas y metodologías a emplear durante el desarrollo de la propuesta de solución.

Segundo capítulo: "Análisis y diseño del módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper", se profundiza en el problema a resolver a través de su descripción. También se describe una propuesta de solución, así como los diagramas que se generan en la medida en que se detallan las funcionalidades a implementar. Se define la arquitectura del módulo con la elección adecuada de los patrones de diseño. Además, se exponen las ideas relacionadas con la validación de los datos, el tratamiento de errores y la seguridad, basándose en los mecanismos que provee el *framework* Symfony. También se describen algunos de los artefactos que son generados durante el diseño de la solución.

Tercer capítulo: "Implementación y validación del módulo generador de reportes estadísticos para sistema de filtrado Xilema Smart Keeper", se exponen los elementos fundamentales del desarrollo del módulo. Se valida la solución propuesta, se presentan las pruebas realizadas a la aplicación para comprobar su correcto funcionamiento, así como un análisis de los resultados obtenidos a partir de las pruebas realizadas.

Capítulo 1: Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper, funcionalidades y tecnologías para su desarrollo.

Un sistema generador de reportes estadísticos de la navegación es una poderosa herramienta diseñada para el manejo de datos de usuarios finales, como pueden ser los administradores de un sistema de *software*. Es importante la comprensión y uso correcto de dicha herramienta para arribar a una correcta toma de decisiones, por tanto, debe tratarse de una interfaz amistosa e intuitiva. Para obtener un amplio conocimiento de su funcionamiento, se hace necesario realizar un profundo estudio del arte entorno a los sistemas generadores de reportes estadísticos para la navegación en Internet, así como una correcta selección e integración de las tecnologías a emplear en el desarrollo de la propuesta de solución.

1.1 Elementos conceptuales

Reporte: Un reporte es un informe o noticia que transmite una información, puede ser de carácter digital, audiovisual, impreso, etc. Es la conclusión de una investigación previa y suele ir acompañado por gráficos, diagramas, tablas de contenido o notas al pie de página en caso de ser un documento (dle.rae.es, 2018).

En el ámbito de la informática**,** los reportes son informes que organizan y exhiben la información contenida en una base de datos**.** Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios.

Estadística: La estadística es una ciencia que utiliza datos numéricos para obtener inferencias basadas en el cálculo de probabilidades.

Como resultado de la aplicación de un algoritmo estadístico a un grupo de datos, permiten la toma de decisiones dentro del ámbito gubernamental, pero también en el mundo de los negocios y el comercio. Hoy puede decirse que la recopilación y la interpretación de los datos obtenidos en un estudio es tarea de la estadística, considerada como una rama de la matemática (upcommons, 2017).

Proxy**:** Es un sistema de *software* o equipo de cómputo dedicado a actuar como intermediario entre un dispositivo de punto final, como un ordenador y otro servidor al cual el usuario solicita determinado servicio. Por ejemplo, si una máquina A solicita un recurso a C, lo hará mediante una petición a B, que a su vez trasladará la petición a C; de esta forma C no sabrá que la petición procedió originalmente de A

(ver figura 1). Esta situación estratégica de punto intermedio le permite ofrecer diversas funcionalidades: control de acceso, registro del tráfico, restricción a determinados tipos de tráfico, mejora de rendimiento y anonimato de la comunicación (arxiv.org, 2013).

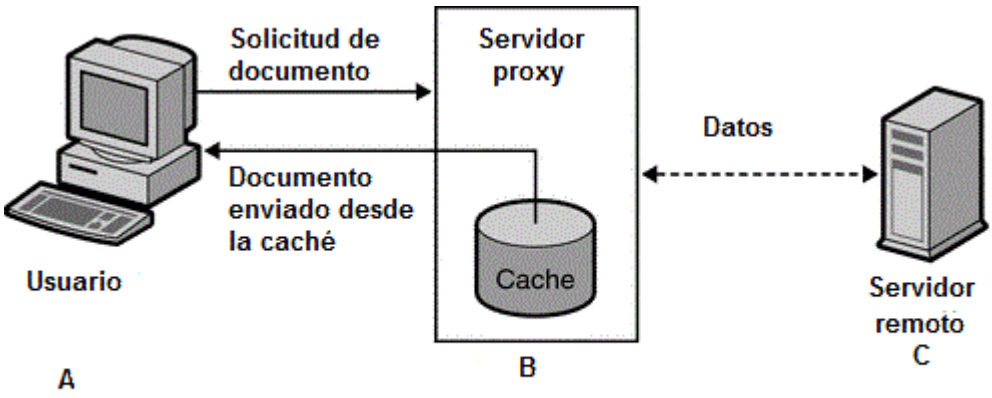


Figura 1. Flujo de trabajo en un *proxy*

Los servidores *proxy* permiten proteger y mejorar el acceso a las páginas web al conservarlas en la caché. De este modo cuando un navegador envía una petición para acceder a un sitio, previamente almacenado en la caché, la respuesta y el tiempo de visualización es más rápido. Los servidores *proxy* aumentan también la seguridad, ya que pueden filtrar cierto contenido web y programas maliciosos (hindawi, 2013).

Archivos logs: Los logs son archivos de texto. Estos ficheros registran todos los procesos que han sido definidos como relevantes por el programador de la aplicación. Por ejemplo, en el caso de los archivos logs de una base de datos se registran todos los cambios de aquellas transacciones completadas exitosamente. Así, en caso de que un fallo del sistema elimine información de la base de datos, el log será la clave para la restauración completa de la base de datos correspondiente. (Digital Guide, 2016)

Dependiendo de su programación, los ficheros log se generan automáticamente. Sin embargo, si se cuenta con los conocimientos necesarios, también será posible crear archivos de registro propios. En general, la línea de un log contiene:

- **Evento** recopilado (p. ej., inicio de programa)
- **Marca de tiempo**, que le asigna fecha y hora

Por lo general, la marca de tiempo se genera primero, con el fin de reflejar la secuencia cronológica de los eventos.

1.2 Sistemas generadores de reportes

Todo servidor de Internet cuenta con un log que registra cada visita de los usuarios, esta información contiene la dirección IP[1], sistema operativo, navegador que se utiliza, el tiempo de permanencia y otros datos que generalmente son empleados con fines estadísticos. Los registros oficiales de eventos son manipulados por administradores de red, haciéndoles posible obtener a través de los sistemas generadores de reportes estadísticos, detalles sobre el qué, quién, dónde, cuándo y por qué un evento ocurre para una aplicación o dispositivo (Digital Guide, 2016).

Existen muchos sistemas generadores de reportes de la navegación. Algunos difieren en cuanto a funcionalidades, configuración, estándares de log que soporta, mientras que otros ofrecen soluciones similares. Con el objetivo de solucionar el problema de investigación definido, se realizó un estudio de los más utilizados en la actualidad a nivel internacional y de las propuestas de solución que se han dado en nuestro país para controlar el uso de Internet en el ámbito empresarial, enfatizando en las características y procesos de interés para el desarrollo de la investigación.

1.2.1 Ámbito Internacional

En la presente sección se muestran los sistemas generadores de reportes de carácter internacional más utilizados por administradores de red en la actualidad.

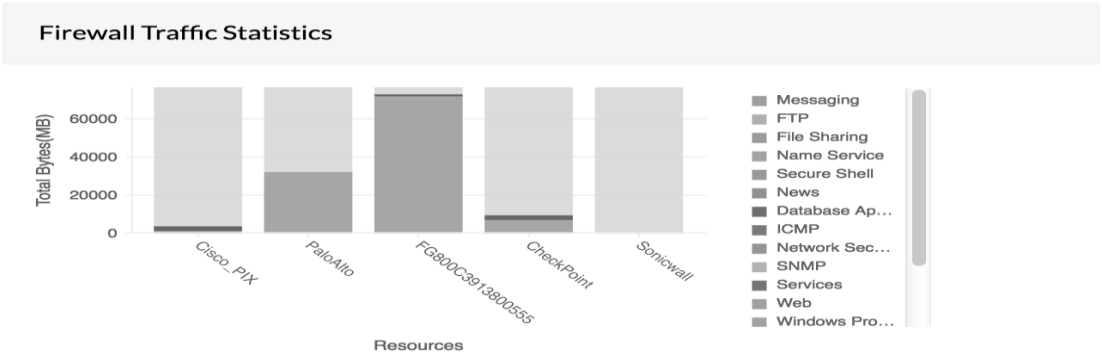
Firewall Analyzer

Firewall Analyzer es un *software* de código abierto para el análisis de registros y administración de configuración para la seguridad en la red de dispositivos. Este producto de análisis de registros de *firewall* se utiliza para monitorear y analizar el estado de seguridad, centralizar la administración y gestionar los cambios. Así mismo, permite monitorear el uso de Internet por parte de los empleados, analizar el ancho de banda, planificar las necesidades futuras, generar informes de auditoría de cumplimiento y muchas otras tareas (Manage Engine, 2017) .

[1] Siglas en inglés de Protocolo Internet

Firewall Analyzer es compatible con prácticamente todos los fabricantes, incluyendo los *firewall* de código abierto, además de los dispositivos comerciales (Check Point, Cisco, Juniper, Fortinet, Snort, Squid Project, SonicWALL, Cyberoam, etc.), IDS/IPS[2], las VPN[3], *Proxies* y dispositivos de seguridad relacionados según (Manage Engine, 2017).

Las estadísticas generadas por el subsistema generador de reportes estadísticos de Firewall Analyzer están orientadas a responsables de seguridad, administradores de la red, y proveedores de servicios gestionados de seguridad, que necesitan recopilar, archivar, analizar los registros de seguridad de los dispositivos y generar informes forenses de forma centralizada.



Device Name	Sent	Received	Total
Cisco_PIX	3.65 GB	28.07 MB	3.67 GB
PaloAlto	31.44 GB	0 MB	31.44 GB
FG800C3913800555	4.98 GB	66.18 GB	71.16 GB
CheckPoint	0 MB	9.32 GB	9.32 GB
Sonicwall	16.52 MB	20.15 MB	36.67 MB

Imagen 1. Estadísticas del tráfico de navegación desde diferentes dispositivos según Firewall Analyzer

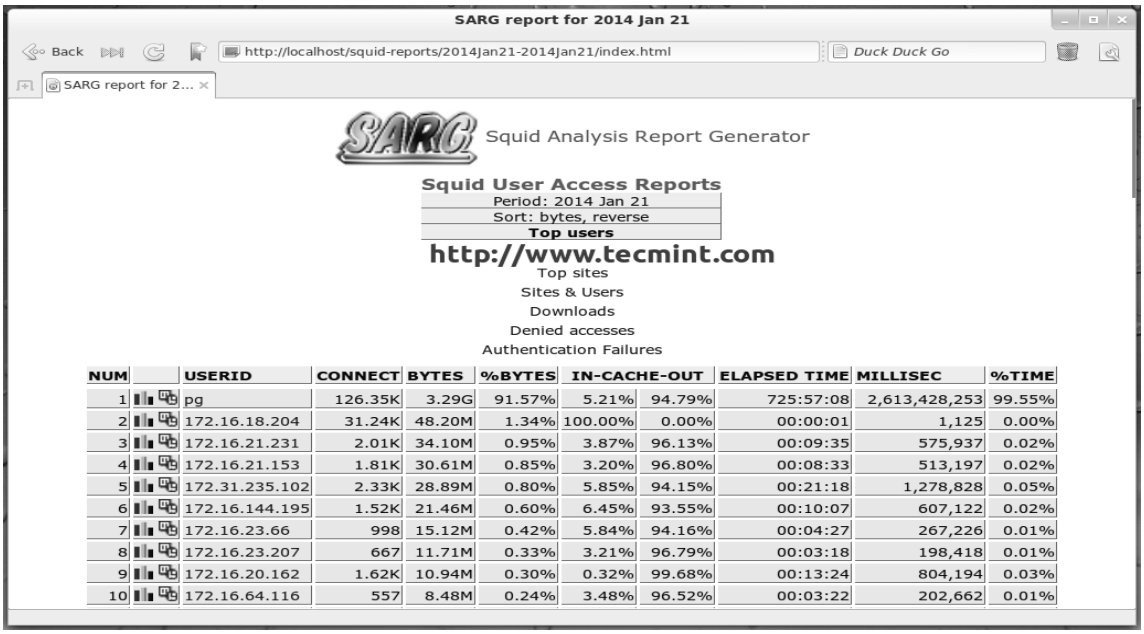
A partir de los informes de registro de tráfico del análisis de *firewall*, los administradores de red supervisarán el uso razonable del ancho de banda para fines comerciales y planificarán los requisitos futuros de capacidad de ancho de banda.

[2] Siglas en inglés de Sistema de Detección de Intrusos / IPS Sistema de Prevención de Intrusos
[3] Siglas en inglés de Red Privada Virtual

Los reportes estadísticos generados en Firewall Analyzer, a pesar de constituir una solución muy favorable tratándose de procesamiento de logs, generación de reportes a través de gráficos, compatibilidad con Squid y otras tecnologías, no es conveniente integrarlo al sistema como solución debido a su empleo de un sistema de base de datos relacional, lo cual no se ajusta a la idea de la propuesta de solución de utilizar un motor de búsqueda para la ejecución más rápida y organizada de consultas y manejo de altos volúmenes de datos.

SARG

SARG, según sus siglas en inglés *Squid Analisys Resport Generetor*, es una herramienta de código abierto que te permite ver con detalle la actividad de todos los equipos y/o usuarios dentro de la red de área local (LAN), registrada en la bitácora del *proxy*, provee mucha información acerca de las actividades de usuarios de Squid: direcciones IP, sitios de acceso uso de ancho de banda total, tiempo transcurrido, descargas, sitios web denegados de acceso, informes diarios, informes semanales e informes mensuales (ver imagen 3).



NUM		USERID	CONNECT	BYTES	%BYTES	IN-CACHE-OUT		ELAPSED TIME	MILLISEC	%TIME
1		pg	126.35K	3.29G	91.57%	5.21%	94.79%	725:57:08	2,613,428,253	99.55%
2		172.16.18.204	31.24K	48.20M	1.34%	100.00%	0.00%	00:00:01	1,125	0.00%
3		172.16.21.231	2.01K	34.10M	0.95%	3.87%	96.13%	00:09:35	575,937	0.02%
4		172.16.21.153	1.81K	30.61M	0.85%	3.20%	96.80%	00:08:33	513,197	0.02%
5		172.31.235.102	2.33K	28.89M	0.80%	5.85%	94.15%	00:21:18	1,278,828	0.05%
6		172.16.144.195	1.52K	21.46M	0.60%	6.45%	93.55%	00:10:07	607,122	0.02%
7		172.16.23.66	998	15.12M	0.42%	5.84%	94.16%	00:04:27	267,226	0.01%
8		172.16.23.207	667	11.71M	0.33%	3.21%	96.79%	00:03:18	198,418	0.01%
9		172.16.20.162	1.62K	10.94M	0.30%	0.32%	99.68%	00:13:24	804,194	0.03%
10		172.16.64.116	557	8.48M	0.24%	3.48%	96.52%	00:03:22	202,662	0.01%

Imagen 2. Estadísticas del tráfico de navegación en SARG

Genera estadísticas en formato html usando como datos los logs de Squid, de toda la navegación realizada a través del *proxy* en un intervalo de tiempo (Ravi Salve, 2016).

SARG es compatible en los siguientes sistemas operativos AIX, BSDI, Digital Unix, FreeBSD, HP-UX, IRIX, GNU/Linux, Mac OS X, NetBSD, NeXTStep, OpenBSD, SCO Unix, SunOS/Solaris, Windows.

Es una herramienta muy útil para ver cuánto ancho de banda de Internet utilizan las máquinas individuales en la red y puede ver a qué sitios web acceden los usuarios de la red, sin embargo, para generar los reportes lo hace a través de tablas, no muestra gráficos y carece de una interfaz amigable, por tanto, no es una alternativa viable para la propuesta de solución.

Sawmill

Herramienta de carácter jerárquico optimizado para reportes web, en su proceso es capaz de soportar 818 formatos de logs. Además de procesar logs de formatos combinados y ampliados, genera estadísticas de ellos, presentación de informes y análisis de acontecimientos. Exporta los datos hacia una base de datos MySQL, Microsoft SQL Server, Oracle, o a la suya propia. Genera y agrega informes de filtrado de forma dinámica a través de una interfaz web. Puede realizar análisis de registros en cualquier plataforma incluyendo Windows, Linux FreeBSD, OpenBSD, Mac OS, Solaris y Unix (sawmill, 2017).

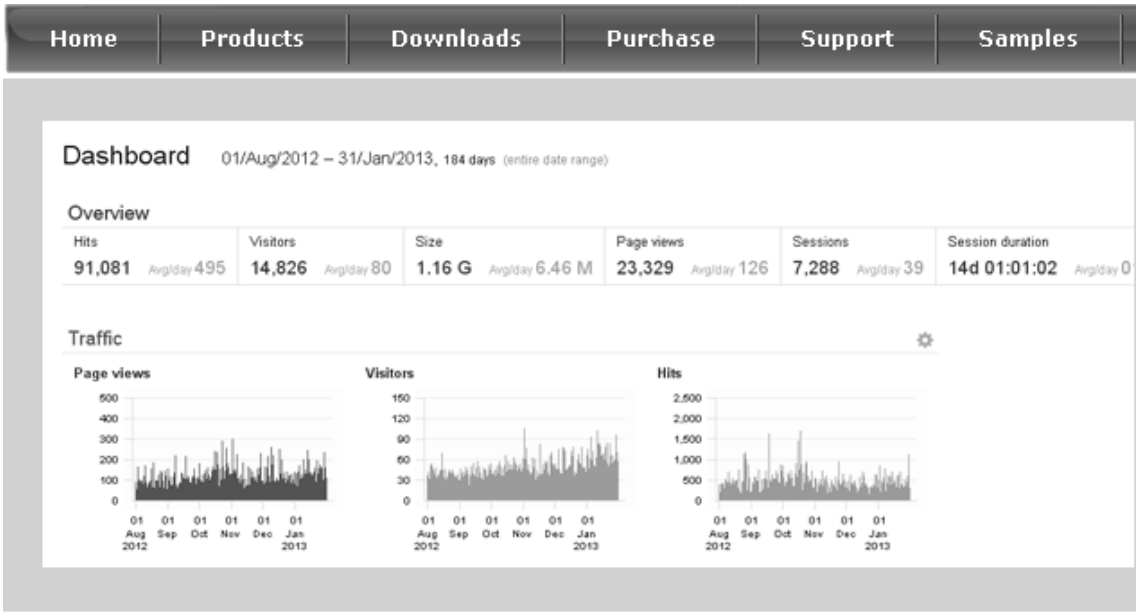


Imagen 3. Reportes del tráfico entre 1/8/2012 y 31/1/2013

Sawmill resulta un *software* altamente configurable, de gran facilidad de uso y que permite procesar gran variedad de reportes. Es un sistema muy dinámico a la hora de filtrar la información procesada de los

registros, lo cual posibilita la extracción de cierta información (ver figura 3). Cuenta con una interfaz amigable, y muestra las estadísticas en forma de gráficas. Su última versión se encuentra disponible en repositorio desde noviembre de 2016.

Es importante resaltar que Sawmill posee una estructura y características de interés para el desarrollo de la presente propuesta de solución, pero se encuentra bajo licencia privativa, lo cual hace altamente costoso su actualización y mantenimiento. Además, no se hace factible tomarlo como solución, pues el modelado de datos también lo ejecuta a través de un sistema de base de datos relacional.

Kibana

Kibana es una herramienta de exploración y visualización de datos de código abierto que se utiliza para el análisis de log y series de tiempo, monitoreo de aplicaciones y casos de uso de inteligencia operacional. Los usuarios pueden crear gráficos de barra, línea y dispersión, o gráficos circulares y mapas (ver imagen 4). Kibana no solo cuenta con estas funcionalidades, es un gran sistema, permite visualizar datos geoespaciales, describe consultas, transformaciones y visualizaciones con expresiones potentes y fáciles de aprender, explora anomalías, etc. (Takase Wataru, Nakamura Tomoaki, 2017)

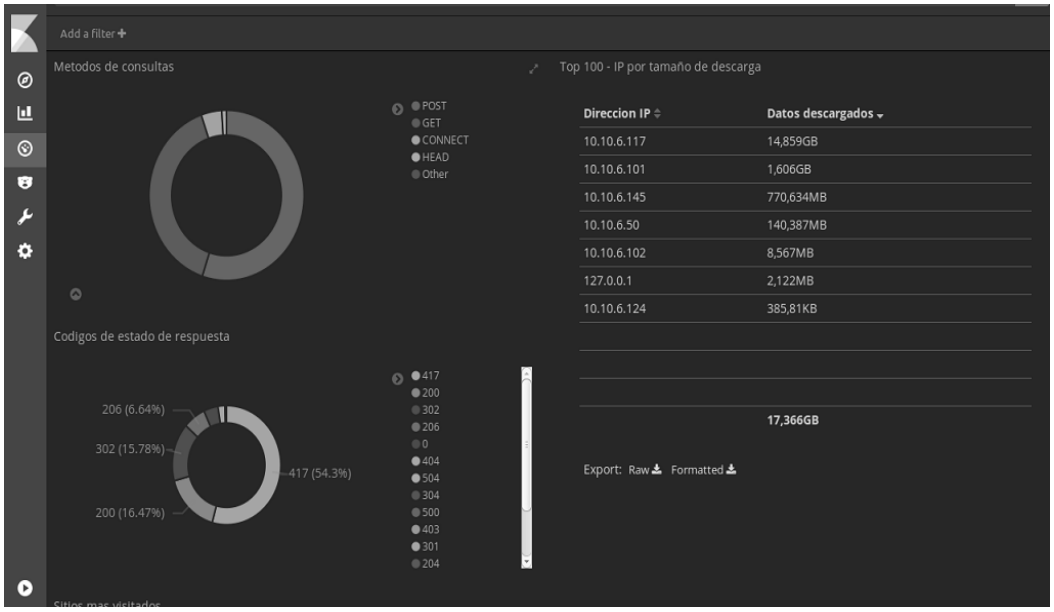


Imagen 4 Reportes en porcentaje de los códigos de respuesta, métodos de consulta y tabla descendente de IP por tamaño de descargas en Kibana.

Kibana ofrece una estrecha integración con Elasticsearch, que la convierte en la opción predeterminada para visualizar los datos almacenados, lo cual es muy conveniente si se fuera a tomar como solución. A pesar de ser un sistema configurable, no es posible agregar o prescindir de funcionalidades de acuerdo a las particularidades de los requisitos, por lo cual no es factible considerar este sistema como una solución.

1.2.2 Ámbito Nacional

En Cuba centros como Segurmática, la Universidad de Ciencias Informáticas, el Instituto Central de Investigaciones Digitales (ICID) y la CUJAE se han dado a la tarea de desarrollar *software* para el control del tráfico de la navegación en Internet, debido a que los sistemas de índole internacional anteriormente estudiados constituyen una alternativa, pero no siempre se adaptan a las necesidades particulares de estas instituciones. En la presente sección se realiza un estudio de los sistemas desarrollados en Cuba, analizadores de logs, los cuales cuentan con funcionalidades y características de interés para el desarrollo de la actual propuesta de solución.

AiresProxy

Sistema desarrollado en la Universidad de las Ciencias Informáticas que emite reportes estadísticos partiendo del análisis de logs de un servidor *proxy*. Apoyándose en técnicas de minería de datos, estadísticas e inteligencia artificial, este *software* posibilita tener datos reales sobre los eventos en la red tanto para estudiantes como para trabajadores de manera individual. Procesa y almacena en una base de datos la información relacionada con los logs del servidor *proxy* Squid. AiresProxy muestra reportes tales como: dominios por IP, por fecha, URL[4] , total por fecha y por IP indistintamente, reporte general, mostrando además el consumo real y el consumo UCI, además de contar con una interfaz amigable (Mutuberría, y otros, 2009).

Esta herramienta permite a los usuarios el conocimiento de sus trazas por Internet, pero no cuenta con reportes especiales para administradores de redes. La mayor desventaja de su uso se encuentra en que ocupa gran espacio del disco según la cantidad de usuarios que se encuentren navegando. Por lo anteriormente planteado y que está basado en el trabajo con base de datos relacional no es conveniente su integración a Xilema Smart Keeper, mas su estudio resulta de gran interés y apoyo para el desarrollo

[4] Siglas en inglés de Localizador de recursos uniforme

de la propuesta de solución, considerando tomar en cuenta algunas funcionalidades y características en general.

Isaweb

Sistema web desarrollado en el Instituto Central de Investigación Digital (ICID), permite a un usuario autenticado visualizar la información de su navegación. Este *software* analiza los ficheros logs generados por servidores *Proxy* IsaServer. Únicamente la información relevante para los usuarios es extraída de estos ficheros y transformada en una base de datos desarrollada en SQL Server. En Isaweb toda la información necesaria es generada por otras aplicaciones en diversos formatos y para transformarla a base de datos relacionales en SQL Server, se crearon aplicaciones de consola que se ejecutan como tareas de Windows en los servidores de Internet y de datos (Palenzuela, 2006).

La información es almacenada en trece tablas relacionadas con un trimestre, donde cada tabla contiene información de una semana a partir del momento en que se pone a funcionar el sistema. La información que se desea ver puede ser filtrada por varios criterios: departamentos, usuarios, palabra clave, fecha, dirección IP, etc. El rango de días en que se quiere realizar la consulta es de carácter obligatorio, debido al gran volumen de información que puede tomar la base de datos (Palenzuela, 2006).

Aparentemente Isaweb brinda una solución adaptable a los requisitos del sistema a desarrollar, es eficiente, utiliza el protocolo LDAP para autenticación de usuarios, cuenta con especificación de filtros para generar consultas e integración con base de datos que facilita en procesamiento de grandes volúmenes de información. Pero como se mencionó anteriormente está diseñado para el servidor IsaServer y no está preparado para analizar ficheros logs de un servidor *Proxy* Squid, que es el que utiliza Smart Keeper, por dicho motivo no se toma como solución.

1.3 Metodología de desarrollo de s*oftware*

En la actualidad existen una gran cantidad de metodologías para el desarrollo de *software*, separadas en dos grandes grupos; las metodologías tradicionales o pesadas y las metodologías ágiles. Las metodologías tradicionales se basan en las buenas prácticas dentro de la ingeniería del *software*, siguiendo un marco de disciplina estricto y un riguroso proceso de aplicación. Las metodologías ágiles, en cambio, representan una solución a los problemas que requieren una respuesta rápida en un ambiente

flexible y con cambios constantes, haciendo caso omiso de la documentación rigurosa y los métodos formales (Esteban Gabriel Maida y otros, 2015).

El Proceso Unificado Ágil de Scott Ambler o Agile Unified Process (AUP) en inglés es una versión simplificada del Rational Unified Process (RUP). Este describe de una manera simple y fácil de entender la forma de desarrollar aplicaciones de *software* de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP (Sánchez, 2015).

El AUP aplica técnicas ágiles incluyendo:

- Desarrollo Dirigido por Pruebas (*test driven development* - TDD en inglés)
- Modelado ágil
- Gestión de cambios ágil
- Refactorización de Base de Datos para mejorar la productividad

Para el satisfactorio cumplimiento de la propuesta de solución se siguió el ciclo de fases AUP:

1. Inicio: El objetivo de esta fase es obtener una comprensión común cliente-equipo de desarrollo del alcance del nuevo sistema y definir una o varias arquitecturas candidatas para el mismo.
2. Elaboración: El objetivo es que el equipo de desarrollo profundice en la comprensión de los requisitos del sistema y en validar la arquitectura.
3. Construcción: Durante la fase de construcción el sistema es desarrollado y probado al completo en el ambiente de desarrollo.
4. Transición: El sistema se lleva a los entornos de preproducción donde se somete a pruebas de validación y aceptación y finalmente se despliega en los sistemas de producción.

Al no existir una metodología de *software* universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable. Se decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI (Sánchez, 2015).

Con esta adaptación se logra estandarizar el proceso de desarrollo de *software*, dando cumplimiento además a las buenas prácticas que define CMMI-DEV v1.3[5]. Se logra hablar un lenguaje común en cuanto

[5] (Capability Maturity Model Integration). Conjunto de modelos de buenas prácticas, para la mejora de los procesos generado a partir de la Arquitectura y Marco1 de CMMI V1.3.

a fases, disciplinas, roles y productos de trabajos. Se redujo a 1 la cantidad de metodologías que se usaban y de más de 20 roles en total que se definían se redujeron a 11, ya que lo que antes era XP, OPEN-UP, RUP, BPM, DAC, KIMBAL, SXP, SCRUM y NOVA OPEN UP, ahora se encuentra recogido en la variación AUP-UCI (Sánchez, 2015). El sistema Xilema Smart Keeper, al cual debe estar integrado el módulo generador de reportes estadísticos de navegación, fue desarrollado bajo esta metodología. Por lo anteriormente expuesto, se decide desarrollar la presente propuesta de solución guiado por la metodología AUP-UCI, considerada la más óptima y adecuada.

1.4 Herramientas y tecnologías

Para el desarrollo del sistema se realizó un estudio de tecnologías con el objetivo de seleccionar las herramientas más adecuadas a utilizar. Siendo el sistema generador de reportes estadísticos un módulo a integrar dentro de Xilema Smart Keeper se tuvieron en cuenta algunas tecnologías empleadas para el desarrollo del mismo, lo que garantizaría la total compatibilidad.

1.4.1 Lenguaje y herramienta de modelado

El lenguaje de modelado es cualquier lenguaje de computadora gráfico o textual que provee el diseño y la construcción de estructuras y modelos siguiendo un conjunto sistemático de reglas y marcos, se usa principalmente en el campo de la ciencia de la computación y la ingeniería para diseñar modelos de *software*, sistemas, dispositivos y equipos nuevos (Techopedia, 2017). Para llevar a cabo la presente propuesta de solución se decidió emplear el Lenguaje Unificado de Modelado (UML por sus siglas en inglés) versión 2.5 utilizada para modelar requisitos, arquitectura, diseño detallado y generación de código de *software.* Las herramientas de modelado UML ofrecen editores gráficos para el desarrollo de modelos UML, generan *software* a partir de modelos UML, crean modelos UML a partir del *software* y respaldan el desarrollo de modelos colaborativos (Koivulahti-Ojala, 2017).

Las herramientas de modelado permiten crear un simulacro del sistema, a bajo costo y con poca probabilidad de riesgo, discutir cambios y correcciones de los requerimientos del usuario, además verificar que se comprenda correctamente el ambiente y quede documentado para que los diseñadores y programadores puedan construir el sistema. Dentro de las principales herramientas de modelado se encuentran las CASE[6]. Visual Paradigm para UML está concebida para soportar el ciclo de vida completo

[6] Siglas en inglés de Ingeniería de *Software* Asistida por Computadora.

del proceso de desarrollo del *software* a través de la representación de diagramas. Se caracteriza por la disponibilidad en múltiples plataformas, por su capacidad de ingeniería directa e inversa, el modelo y código que permanece sincronizado en todo el ciclo de desarrollo, su diseño está centrado en casos de uso y enfocado al negocio que generan un *software* de mayor calidad, genera informes, es fácil de instalar y actualizar, soporta aplicaciones Web (Pressman, 2002), lo cual es imprescindible para la presente propuesta de solución. Por lo anteriormente planteado se ha decidido utilizar esta herramienta en su versión 8.0.

1.4.2 *Framework*

Framework o marco de trabajo es una estructura conceptual y tecnológica de asistencia definida, normalmente, con artefactos o módulos concretos de *software*, que puede servir de base para la organización y desarrollo de *software*, puede incluir bibliotecas y lenguaje interpretado (Riehle, 2000). Es decir, funciona como una base predefinida a la cual agregándole ciertas particularidades conforma la aplicación que se desee implementar.

Symfony es un *framework* PHP[7] de *software* libre que permite crear aplicaciones y sitios web rápidos y seguros de forma profesional. Su código, y el de todos los componentes y librerías que incluye, se publican bajo la licencia MIT (Massachusetts Institute of Technology) de *software* libre. Está basado en el patrón MVC (Modelo-Vista-Controlador). Los componentes de Symfony son muy útiles y se encuentran probados, proyectos tan gigantescos como Drupal 8 están construidos con ellos (2017). Estudiadas sus características y teniendo en cuenta que Xilema Smart Keeper emplea este *framework* en su desarrollo, para la implementación del sistema generador de reportes se decidió usar Symfony en su versión 3.4.

1.4.3 Lenguaje de programación del lado del servidor

Un lenguaje de programación es un código de comunicación con el ordenador que permite crear programas que realicen cualquier tipo de proceso ejecutable. Un lenguaje de programación incluye instrucciones, funciones aritmético lógicas, símbolos, o caracteres que representan datos, palabras y un conjunto de normas (Suárez, 2007). En otras palabras, es un conjunto de reglas semánticas y sintácticas empleadas para la codificación de instrucciones de un programa o algoritmo de programación. Uno de los lenguajes de programación del lado del servidor más populares en la actualidad es PHP, el cual es un

[7] Hipertext Pre-procesor

lenguaje de código abierto especialmente adecuado para aplicaciones web dinámicas. Posee alta capacidad de expandir su potencial empelando módulos y extensiones, permite el manejo de excepciones y la aplicación de técnicas de programación orientada a objetos (PHP, 2017). PHP cuenta con una amplia documentación en diversos idiomas, por lo que se hace fácil su estudio y aprendizaje. Para la implementación del módulo se emplea el lenguaje PHP en su versión 7.0.

1.4.4 Entorno Integrado de Desarrollo

NetBeans es un entorno para el desarrollo de *software* con un código abierto. Es compatible con los siguientes lenguajes de programación: Java, C, C + +, PHP, Python, JavaScript, etc. NetBeans contiene las funciones de refactorización, perfilado, auto-completado, resaltado de sintaxis coloreada y define las plantillas de código. También NetBeans proporciona a un desarrollador de las herramientas necesarias para crear aplicaciones profesionales, empresariales y móviles (NetBeans, 2018).

1.4.5 Servidor Web

Para el desarrollo de la propuesta de solución se establecerá el sistema en el servidor web Apache 2.0. Apache es un servidor web HTTP de código abierto desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la *Apache Software Foundation* dentro del proyecto HTTP Server (httpd). Es altamente configurable, presenta bases de datos de autenticación y negociado de contenido y es compatible con el lenguaje de programación PHP. Apache es flexible, rápido y posee la característica de ampliar considerablemente sus capacidades pues existe un repositorio extenso completamente gratuito de extensiones y módulos.

1.4.6 Squid

Es un *proxy* de almacenamiento en caché para la Web compatible con HTTP, HTTPS, FTP y más. Reduce el ancho de banda y mejora los tiempos de respuesta mediante el almacenamiento en caché y la reutilización de páginas web solicitadas con frecuencia. Squid tiene amplios controles de acceso y es un excelente acelerador de servidores. Es libre y se ejecuta en la mayoría de los sistemas operativos disponibles, incluido Windows, y tiene licencia bajo GNU GPL (squid-cache.org, 2013).

Según el sitio oficial de la herramienta (squid-cache.org, 2013), entre las principales características de Squid se encuentran:

- Jerarquías de caché: Diversos servidores trabajan conjuntamente atendiendo las peticiones. Un navegador solicita siempre las páginas a un solo *proxy* y si este no tiene la página en su caché consulta a sus hermanos, que a su vez también podrían consultar con sus padres.
- Caché transparente: Squid se puede configurar para ser usado como proxy transparente empleando un *firewall* que intercepte y redirija las conexiones sin configuración por parte del cliente, e incluso sin que el propio usuario conozca de su existencia.
- Control de acceso: Ofrece la posibilidad de establecer reglas de control de acceso. Esto permite establecer políticas de acceso en forma centralizada, simplificando la administración de una red.
- Gestión de tráfico: Permite categorizar el tráfico y limitarlo de manera individual o agrupada para conseguir un mejor aprovechamiento del ancho de banda disponible en la conexión a Internet.

1.4.7 Elasticsearch

Elasticsearch es un servidor de búsqueda publicado como código abierto bajo la licencia Apache. Permite indexar y analizar en tiempo real grandes cantidades de datos de manera distribuida. Va un paso más allá de la búsqueda por texto gracias a un DSL[8] y un API[9] para búsquedas más complicadas. Provee un motor de búsqueda de texto completo, distribuido y con capacidad de multi-tenencia con una interfaz web RESTful[10] con documentos JSON[11] (Schmitt, y otros, 2014). Tiene una gran velocidad de respuesta. El que sea distribuido hace de Elasticsearch un sistema fácilmente escalable y adaptable a las distintas situaciones y ofrece diversas librerías cliente desde las cuales manejar los nodos/clusters/índices. Está desarrollado en Java por lo que es sumamente compatible con casi todas las plataformas. Para el desarrollo de la propuesta de solución se hace uso del servidor de búsqueda Elasticsearch en su versión 5.5.0 en lugar de una base de datos relacional, tal elección se basa en la rapidez con que puede ejecutar una consulta, tratándose de datos indexados y no de tablas.

[8] *Digital Subscriber Line:* Familia de tecnologías que proporcionan el acceso a Internet mediante la transmisión de datos digitales a través de los cables de una red telefónica local
[9] *Application Programming Interface*: Conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro *software* como una capa de abstracción.
[10] Hace referencia a un servicio web que implementa la arquitectura REST (Representational State Transfer)
[11] *JavaScript Object Notation:* Formato ligero de intercambio de datos basado en un subconjunto del Lenguaje de Programación JavaScript

1.4.8 Logstash

Logstash es una herramienta para la administración de logs. Se puede utilizar para recolectar, parsear y guardar los logs para futuras búsquedas. Esta herramienta basa su funcionamiento en la integración de entradas, códecs[12], filtros y salidas. Las entradas son las fuentes de datos que se usarán posteriormente; los códecs convierten un formato de entrada en otro que Logstash acepte, y éste último formato en otro de salida (Elder, 2017). Logstash es altamente compatible con Elasticsearch, por lo cual se determinó emplearlo para extraer la información de los logs de Squid y enviarlos al servidor de búsqueda en su versión 5.5.0.

1.4.9 JMeter

JMeter es un *software* de Apache desarrollado en Java utilizado como herramienta diseñada para cubrir pruebas de carga, funcional, rendimiento, regresión, etc. JMeter soporta aserciones para asegurar que los datos recibidos son correctos, por lo que es una herramienta de realización de pruebas automáticas. puede ser usado como una herramienta de pruebas unitarias para conexiones de bases de datos con JDBC, FTP, LDAP, Servicios web, JMS, HTTP y conexiones TCP genéricas. JMeter puede también ser configurado como un monitor, aunque es comúnmente considerado una solución ad-hoc respecto de soluciones avanzadas de monitoreo (Sai Matam y otros, 2016).

Las características de Apache JMeter incluyen:
Capacidad para cargar y probar el rendimiento de diferentes aplicaciones / servidores / tipos de protocolos:

- Web: HTTP, HTTPS (Java, NodeJS, PHP, ASP.NET, ...)
- Servicios web SOAP / REST
- FTP
- Base de datos a través de JDBC
- LDAP
- *Middleware* orientado a mensajes (MOM) a través de JMS
- Correo: SMTP (S), POP3 (S) e IMAP (S)
- Comandos nativos o scripts de Shell

[12] Programa o dispositivo hardware capaz de codificar o decodificar una señal o flujo de datos digitales

- TCP
- Objetos de Java

Apache JMeter se puede usar para probar el rendimiento tanto en recursos dinámicos como estáticos, aplicaciones web dinámicas. Además, es utilizado para simular una gran carga en un servidor, grupo de servidores, red u objeto para probar su fortaleza o analizar el rendimiento general bajo diferentes tipos de carga (Apache Software Fundation, 2017). Para efectuar las pruebas de rendimiento luego de desarrollada la propuesta de solución y con el objetivo de obtener resultados satisfactorios y confiables se empleará JMeter en su versión 2.12.

Conclusiones parciales

- A partir del análisis de los preceptos teóricos abordados en el presente capítulo se arrojan las siguientes conclusiones parciales:
- El estudio de soluciones existente arrojó como resultado que diversos softwares desarrollados para la generación de reportes estadísticos resultan adecuados, pero se trata de aplicaciones independientes a Xilema Smart Keeper e integrar alguna de ellas al sistema implicaría dependencia y el empleo de más recursos como servidores, por lo cual, no se tomó ninguna como solución a la problemática planteada.
- El estudio sobre herramientas, tecnologías y metodologías permitió realizar una selección de las más adecuadas para definir los componentes bases para el desarrollo de la propuesta de solución, teniendo en cuenta que contaran con un soporte multiplataforma y basados en software libre.

Capítulo 2: Análisis y diseño del módulo generador de reportes estadísticos del sistema Xilema Smart Keeper.

En el presente capítulo se describe el proceso llevado a cabo durante la fase de análisis y diseño; en el cual se definen los Requisitos Funcionales (RF) y No Funcionales (RNF), fase importante durante el desarrollo de la propuesta de solución, pues el éxito del *software* parte de una correcta definición de los requisitos. Se especifica la estructura física de la solución, describiéndose las principales clases del sistema y de los componentes de la solución existente. Como parte del proceso se generan los diferentes diagramas y artefactos correspondientes para dar cumplimiento a los objetivos trazados.

2.1 Descripción de la propuesta de solución.

En el capítulo anterior se reflejó la necesidad de contar con un sistema generador de reportes estadísticos para Xilema Smart Keeper que permita mostrar los principales reportes existentes de una forma más comprensible. En la actualidad, el generador de reportes utilizado por el sistema no muestra sus resultados mediante gráficas, por tanto, requiere de un mayor esfuerzo el análisis e interpretación de la información.

La propuesta de solución a la problemática planteada consiste en elaborar una aplicación con las siguientes características:

- Capaz de generar gráficos con los datos almacenados Elasticsearch y que se relacionan con los logs de Squid (códigos de estado de respuesta sitios más visitados, usuarios por tamaño de descarga, dominios accedidos desde un IP, horarios y días de mayor navegación, tipos de contenido descargados y métodos de conexión).
- Capaz de generar, en tiempo real, tablas y gráficos que representen datos tomados de los registros del *proxy* Squid.
- La interfaz web es la encargada de mostrar las gráficas y tablas creadas, así como de proporcionar la opción de filtrar dichas gráficas en un rango de tiempo deseado por el administrador (última hora, último día, última semana o último mes).
- Para almacenar los datos de Squid en el motor de búsqueda se hace a través de Logstash quien analiza los logs de navegación, enviándolos luego a Elasticsearch.

2.2 Requisitos de la propuesta de solución

El objetivo principal de la Especificación de Requisitos del Sistema (ERS) es servir como *medio de* comunicación entre clientes, usuarios, ingenieros de requisitos y desarrolladores. En la ERS deben recogerse tanto las necesidades de clientes y usuarios *(*necesidades del negocio*, también conocidas como* requisitos de usuario*,* requisitos de cliente*,* necesidades de usuario*, etc.)* como los requisitos que debe cumplir el sistema *software* a desarrollar para satisfacer dichas necesidades *(*requisitos del producto*, también conocidos como* requisitos de sistema *o* requisitos *software*) (Marco de desarrollo de la Junta de Andalucía, 2018).

2.2.1 Requisitos Funcionales

Según (Portillo, 2013) los requisitos funcionales son declaraciones de los servicios que proveerá el sistema, de la manera en que éste reaccionará a entradas particulares. En algunos casos, también declaran explícitamente lo que el sistema no debe hacer. En la siguiente tabla se muestran los requisitos funcionales del módulo generador de reportes estadísticos para Xilema Smart Keeper:

Tabla 1: Requisitos funcionales del módulo.

Requisitos	Descripción	Prioridad
RF1 Generar reporte de estado de respuesta.	El sistema genera reporte en grafica circular sobre el estado de los sitios visitados basándose en los códigos de respuesta.	Alta
RF2 Generar reporte de los sitios más visitados.	El sistema genera reporte en forma de gráfica circular de los sitios más visitados.	Alta
RF3 Generar reporte de todos los sitios visitados.	El sistema genera reporte en forma de tabla todos los sitios visitados y la cantidad de visitas.	Alta
RF4 Generar reporte de los usuarios con mayor	El sistema genera reporte en forma de gráfica circular de los 10 usuarios con	Alta

tamaño de descarga.	tamaño de descargas.	
RF5 Generar reporte de los usuarios por tamaño de descarga.	El sistema genera reporte en forma de tabla de todos usuarios por tamaño de descargas y el tamaño total de descargas, de manera descendente.	Alta
RF6 Generar reporte de los IP con mayor tamaño de descarga.	El sistema genera reporte en forma de gráfica circular de los 10 IP con tamaño de descargas.	Alta
RF7 Generar reporte de los IP por tamaño de descarga.	El sistema genera reporte en forma de tabla de todos IP por tamaño de descargas y el tamaño total de descargas, de manera descendente.	Alta
RF8 Generar reporte de los dominios accedidos desde un IP.	El sistema genera reporte de todos los dominios visitados por un mismo IP.	Alta
RF9 Generar reporte de los dominios accedidos por un usuario.	El sistema genera reporte de todos los dominios visitados por un mismo usuario.	Alta
RF10 Generar reporte de los usuarios que han accedido a un dominio.	El sistema genera reporte de todos los usuarios que han visitado un dominio.	Alta
RF11 Generar reporte de los IP que han accedido a un dominio.	El sistema genera reporte de todos los IP que han visitado un dominio.	Alta
RF12 Generar reporte de los horarios de mayor	El sistema genera reporte en graficas de barras de los horarios de mayor navegación.	Alta

navegación.		
RF13 Generar reporte de navegación por día (por peticiones).	El sistema genera reporte en graficas de barras de los días de mayor navegación basado en las peticiones.	Alta
RF14 Generar reporte de navegación por día (por consumo).	El sistema genera reporte en graficas de barras de los días de mayor navegación basado en el consumo.	
RF15 Generar reporte de los tipos de contenido más descargados.	El sistema genera reporte en grafica circular de los tipos de contenido con mayor cantidad de descargas.	Alta
RF16 Generar reporte sobre los métodos de conexión.	El sistema genera un reporte estadístico en gráfica circular de los diferentes métodos de conexión.	Alta
RF17 Exportar los reportes.	El sistema exporta los reportes en varios formatos.	Media
RF18 Imprimir los reportes.	El sistema imprime los reportes	Baja
RF19 Seleccionar rango de fecha.	El sistema permite seleccionar el rango de fecha de los reportes.	Media

2.2.2 Requisitos no funcionales

Según (Fuentes López, 2011) los requisitos funcionales son aquellos que no se refieren directamente a las funciones específicas que entrega el sistema, sino a las propiedades emergentes de éste como la fiabilidad, la respuesta en el tiempo y la capacidad de almacenamiento. De forma alternativa, definen las restricciones del sistema como la capacidad de los dispositivos de entrada/salida y la representación de

datos que se utiliza en la interfaz del sistema. A partir de un análisis se establecieron 8 requisitos nos funcionales, los cuales quedan relacionados a continuación:

- Hardware

  RnF 1**:** El ordenador donde se ejecute el módulo debe tener una memoria *RAM* de 2GB o superior y un procesador a una velocidad 1.7 GHz o superior.

  RnF 2: Los servidores para el almacenamiento de la base de datos deben contar con un disco duro de más de 250 GB.

- *Software*

  RnF 3: El ordenador donde se ejecute el módulo debe tener instalada la distribución de *GNU/Linux Ubuntu* 14.04 o superior.

  RnF 4: El ordenador donde se ejecute el módulo requiere de las librerías *ssh*, *openssl* y un servidor de base de datos Elasticsearsh.

- Apariencia o interfaz externa

  RnF 5: La interfaz del sistema debe ser intuitiva y sencilla, debe contar con un diseño formal y claro teniendo en cuenta el fin con el que se desarrolla la aplicación.

- Usabilidad

  RnF 6: El sistema debe permitir el acceso y trabajo a administradores de red que posean conocimientos básicos en el manejo de bases de datos y tecnologías en general. Se podrá acceder a la información deseada, con pocos pasos, en el menor tiempo posible y con la facilidad requerida.

- Rendimiento

  RnF 7: Debe ser intuitivo, los requisitos funcionales son agrupados por responsabilidad, para permitir a los administradores de red de poca experiencia, hacer uso del mismo.

- Legalidad

  RnF 8: Las tecnologías seleccionadas para el desarrollo del sistema están basadas principalmente en licencias libres.

2.3 Historias de Usuarios

En su escenario 4 para la disciplina Requisitos, la metodología AUP-UCI, genera como artefacto a las Historias de Usuario (HU) (Sánchez, 2015). Las historias de usuario son descripciones, siempre muy cortas y esquemáticas, que resumen la necesidad concreta de un usuario al utilizar un producto o servicio, así como la solución que la satisface. Su función principal es identificar problemas percibidos, proponer soluciones y estimar el esfuerzo que requieren implementar las ideas propuestas (Villamizar, y otros, 2016).

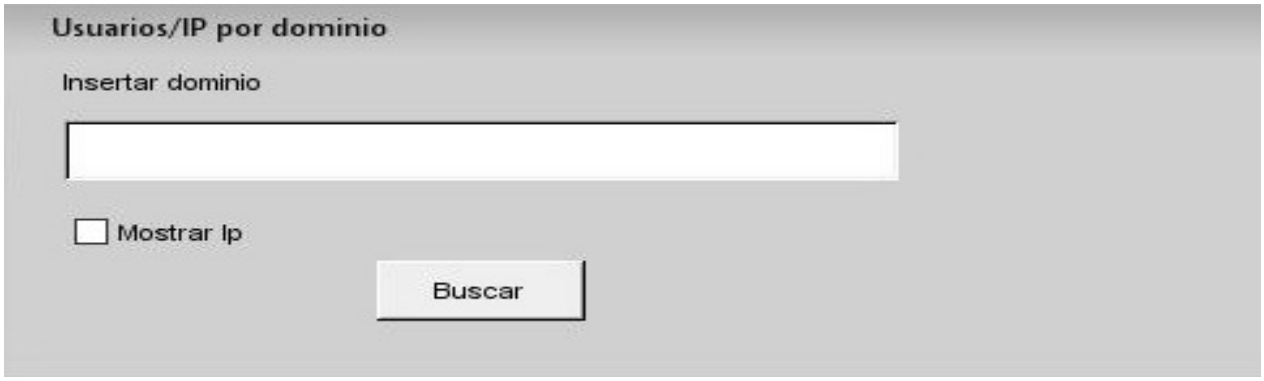
Durante el desarrollo de la propuesta de solución fueron generadas 19 HU, de las cuales 2 se muestran a continuación, la HU_1 y la HU_8 pertenecientes a los RF 1 y 8 respectivamente:

Tabla 2. HU_1 Generar Reportes de estado de respuesta.

Historia de Usuario	
Numero: HU_1	**Nombre:** Generar reporte de estados de respuesta
Programador responsable: Laura Gregores Nápoles	**Iteración asignada:** 1
Prioridad: Alta	
Tiempo estimado: 8 horas	**Tiempo real:** 8 horas
Descripción: El usuario puede visualizar a través de un gráfico de pastel los estados de respuesta más comunes y el por ciento con respecto al total que representa cada uno.	
Observaciones:	
Prototipo de interfaz: No aplica	

Tabla 3 HU_9. Generar Reportes de estado de respuesta.

Historia de Usuario	
Numero: HU_9	**Nombre:** Generar reporte de los dominios accedidos desde un usuario.
Programador responsable: Laura Gregores Nápoles	**Iteración asignada:** 1
Prioridad: Alta	

Tiempo estimado: 8 horas	**Tiempo real:** 8 horas
Descripción: El usuario administrador introduce un usuario en el formulario y puede visualizar reporte de todos los dominios visitados por dicho usuario.	
Observaciones: Si se introduce correctamente el usuario el sistema mostrará una tabla con los dominios visitados por el mismo. Si el usuario introducido en el formulario no existe en el registro el sistema muestra un mensaje de 0 coincidencias encontradas. Si se introducen los datos incorrectamente el sistema mostrará un mensaje de error y señalará en rojo el campo cuya validación es incorrecta.	
Prototipo de interfaz: 	

2.4 Arquitectura de *Software*

La arquitectura de *software* es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un *software*, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores del *software* compartir una misma línea de trabajo y cubrir todos los objetivos y restricciones de la aplicación. Es considerada el nivel más alto en el diseño de la arquitectura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del *software* (García Acevedo , 2015).

Modelo – Vista – Controlador

El flujo d trabajo de Symfony está basado en el patrón arquitectónico Modelo – Vista – Controlador, proporcionando a las aplicaciones un desarrollo rápido y sencillo. Los niveles que conforman este patrón son:

Modelo: Contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia. El modelo está representado en los archivos del directorio: *Entity.*

Vista: También llamada interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste. Las vistas del módulo están agrupadas en el directorio: *app/Resources/view.*

Controlador: Actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno. Las clases controladoras del módulo están situadas en el directorio: *Controller.*

En el controlador se encuentran todas las acciones necesarias para la realización de los reportes, el mismo contienen la lógica de la aplicación. Estas acciones utilizan el modelo para consultar los datos necesarios y enviarlos a la vista, que es la encargada de originar las páginas que son mostradas como resultado de las acciones. En el modelo se encuentra la clase relacionada con los logs del servidor *proxy* Squid. El controlador es la capa intermediaria entre la Vista y el Modelo, encargado de recibir todas las acciones del módulo y convertirlas en resultados.

2.5 Patrones de diseño

Según define (Pressman, 2010) un patrón de diseño está caracterizado como una regla de tres partes, la cual expresa una relación entre cierto contexto, un problema y una solución. Para el diseño de *software,* el contexto permite comprender el ambiente en el que reside el problema y cuál sería la solución apropiada. Un conjunto de requerimientos, incluidas limitaciones y restricciones, actúan como sistemas de fuerzas que influyen en la manera en la que puede interpretarse el problema en este contexto y en cómo podría aplicarse con eficacia la solución. Durante la implementación del módulo generador de reportes, se estarán empleando los siguientes patrones de diseño:

2.5.1 Patrones Generales de *Software* para la Asignación de Responsabilidades (GRASP)

Se denominan "Patrones GRASP" por sus siglas en inglés: *General Responsibility Assignment Software Patterns* o Patrones Generales de *Software* para Asignación de Responsabilidades, en español y según (Cortés, y otros, 2014). Para el desarrollo del módulo, dentro de esta familia, se emplearon los siguientes patrones:

Experto: En el módulo se evidencia el uso del patrón en el momento de realizar los reportes mediante la clase ***ReporteController*** la encargada de realizar todos los reportes.

Alta Cohesión: La cohesión es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Las mismas poseen un número relativamente pequeño de responsabilidades, las clases realizan solo las funciones para las cuales fueron creadas, esto conlleva a un bajo acoplamiento y fomenta la reutilización. En el módulo, este patrón se manifiesta en la clase controladora ***ReporteController*** colabora y delega responsabilidades a la clase ***DominioType*** para la creación de formularios.

Bajo Acoplamiento: Es una medida de la fuerza en las conexiones de una clase con otras, la recurrencia y las conexiones a ella. Una clase con bajo acoplamiento no depende de muchas clases. Su utilización se evidencia en que las clases controladoras del sistema no se relacionan entre sí, lo que disminuye las dependencias entre las mismas, en el módulo se evidencia en la clase ***ReporteController***.

Controlador: Un Controlador es un objeto de interfaz no destinado al usuario, se encarga de manejar un evento del sistema a una clase que represente el sistema global. En el módulo, los eventos generados por el usuario son redirigidos a una clase controladora que realiza las operaciones solicitadas.

2.5.2 Patrones Gang of Four (GOF)

Según (Craig, 2003) los patrones GOF describen las formas comunes en que diferentes tipos de objetos, pueden ser organizados para trabajar unos con otros. Tratan la relación entre clases y la formación de estructuras de mayor complejidad. Además, permiten crear grupos de objetos para ayudar a realizar tareas complejas. Los utilizados en el desarrollo del módulo fueron:

Estructurales

***Decorator*:** En el módulo se evidencia en la clase *base.html.twig* y en la clase *layout.html.twig,* las cuales representan la estructura de diseño del sistema y de ellas heredan el resto de las páginas del módulo

Otros

Registry: Es un patrón útil para la programación orientada a objetos. Permite compartir datos y objetos en la aplicación de manera sencilla, sin la necesidad de conservar numerosos parámetros o el empleo de variables globales. En el módulo se aplica en la clase de configuración (*config.yml*) que es la encargada

de guardar las variables globales del módulo, como *drivers* de conexión a Elasticsearch y rutas bases del proyecto.

2.6 Diagramas de clases del diseño

Un Diagrama de Clases de Diseño (DCD) muestra la especificación para las clases de una aplicación, incluyendo sus asociaciones y atributos, la navegabilidad, métodos, dependencias y las interfaces con sus respectivas operaciones y constantes (UNAD, 2018). Las DCD muestra definiciones de entidades *software* más que conceptos del mundo real. En la presente investigación se generaron un total de 18 DCD, de los cuales se exponen 2, correspondientes a HU_1 y HU_2:

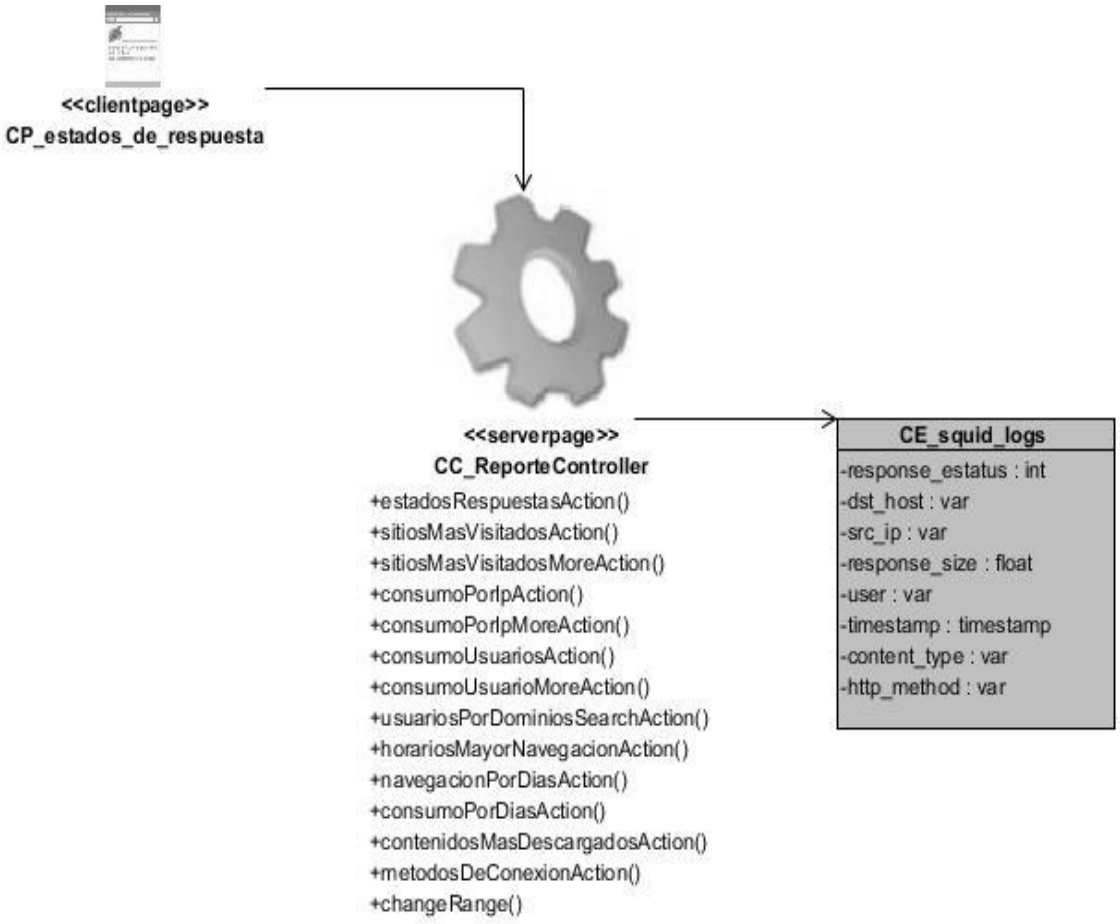


Figura 2. Diagrama de clases del diseño HU_1 Reporte de estados de respuesta.

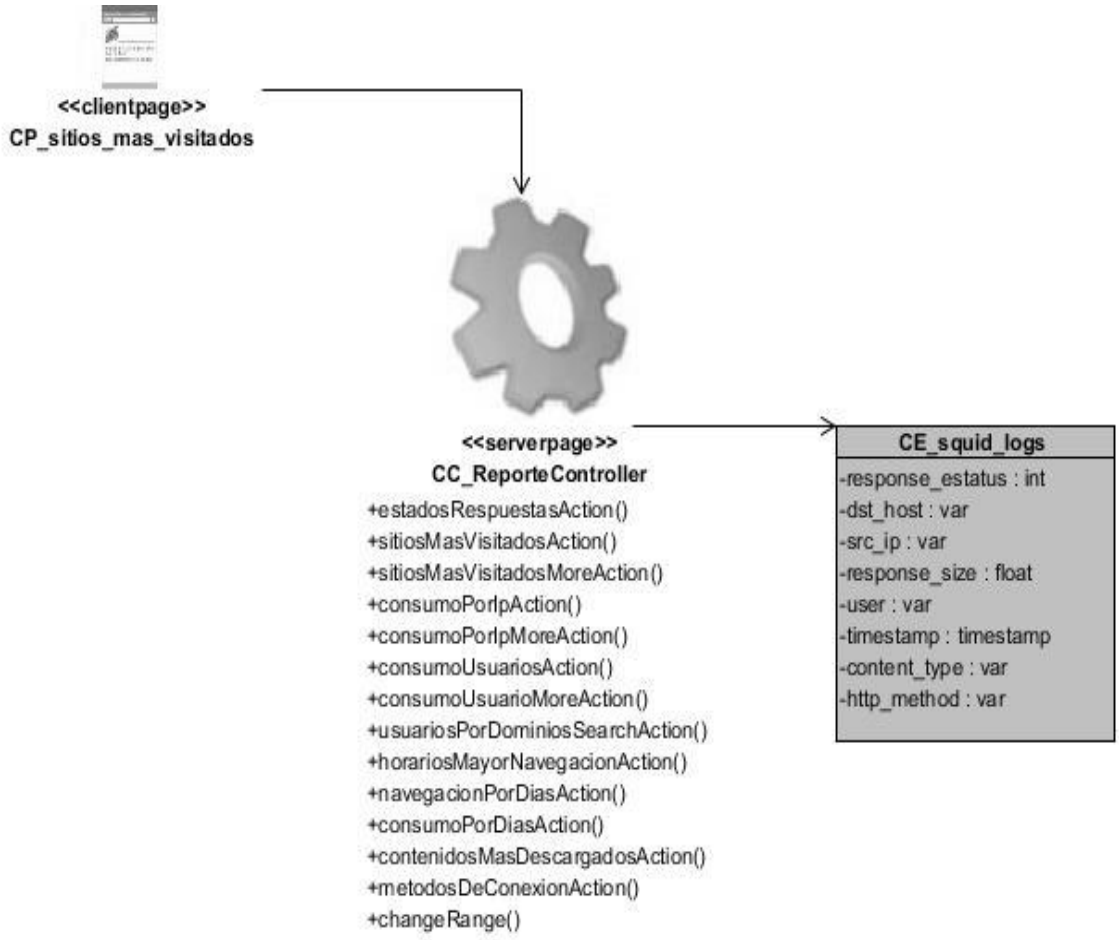


Figura 3. Diagrama de clases del diseño HU_2 Reportes de sitios más visitados.

2.7 Diagramas de secuencia

Un diagrama de secuencia es una forma de diagrama de interacción que muestra los objetos como líneas de vida a lo largo de la página y con sus interacciones en el tiempo representadas como mensajes dibujados como flechas desde la línea de vida origen hasta la línea de vida destino. Los diagramas de secuencia son buenos para mostrar qué objetos se comunican con qué otros objetos y qué mensajes generan esas comunicaciones (Rodríguez , y otros, 2013). A continuación, se muestran dos de los 15 generados en la presente investigación, relacionados con los RF 2 y RF 3 respectivamente

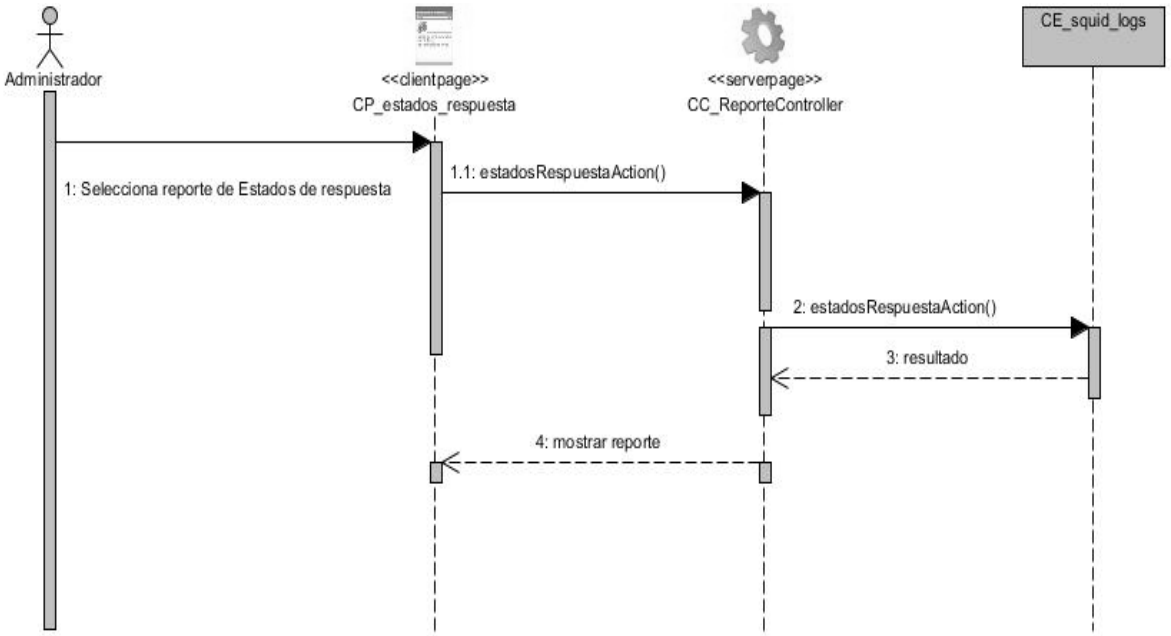


Figura 4. Diagrama de secuencia HU_1 Generar reporte de Estados de respuesta.

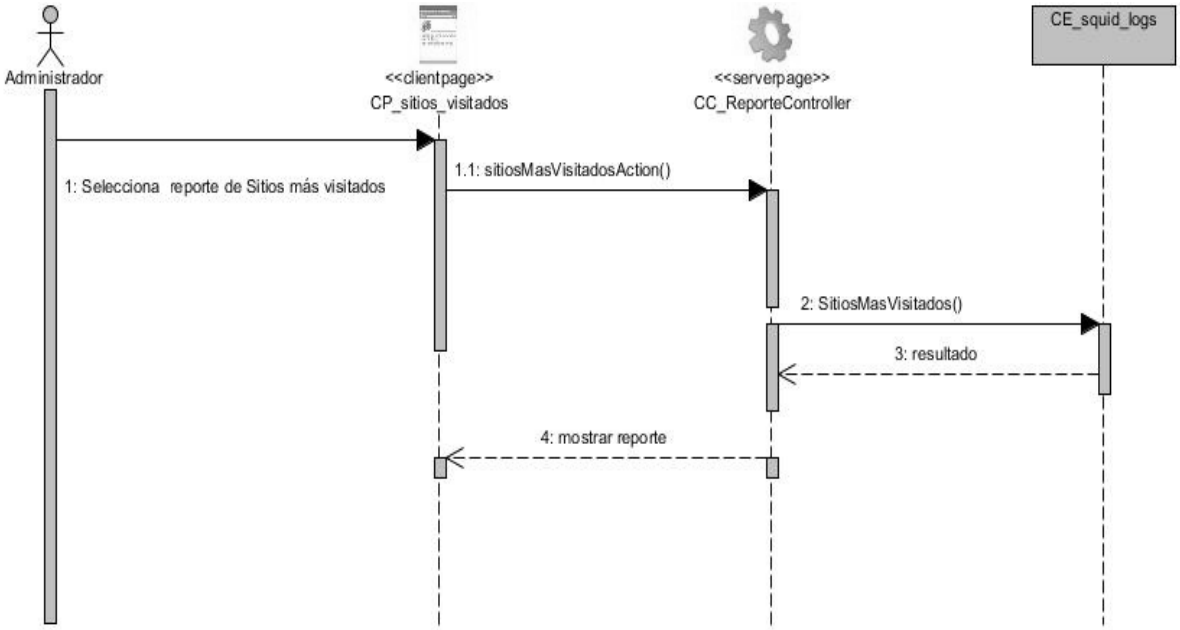


Figura 5. Diagrama de secuencia HU_2 Generar Reportes de Sitios más visitados

2.8 Modelo de datos

Según el concepto plasmado en (Fuentes González, 2015) un modelo datos muestra la estructura lógica de la base de datos, incluidas las relaciones y limitaciones que determinan cómo se almacenan los datos y cómo se accede a ellos. Los modelos de bases de datos individuales se diseñan en base a las reglas y los conceptos de cualquier modelo de datos más amplio que los diseñadores adopten. La mayoría de los modelos de datos se pueden representar por medio de un diagrama de base de datos acompañante. Como puede observarse en la figura, el modelo de datos generado para la propuesta de solución de la presente investigación, está estructurado en una tabla.

Squid-logs: Almacena la información que brindan los logs de Squid, la cual posteriormente es macheada por Logstash y enviados a la base de datos Elasticsearch, donde está contenida esta tabla. De los logs de Squid, en esta tabla se almacenan, estado de respuesta (response_status), dominio (dst_host), dirección ip (src_ip), user(usuario), tiempo en que se realizó la respuesta (timestamp), tamaño de respuesta (response_size), tipo de contenido (content_type), método de conexión (http_method).

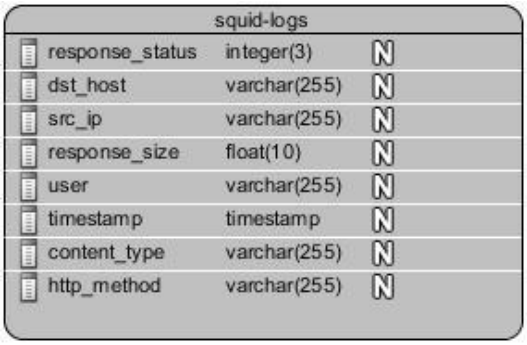


Figura 6. Modelo de datos de la propuesta de solución.

2.9 Diagrama de despliegue

Los Diagramas de Despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación (UNAD, 2018). La figura 9 representa el diagrama de despliegue propuesto para el módulo.

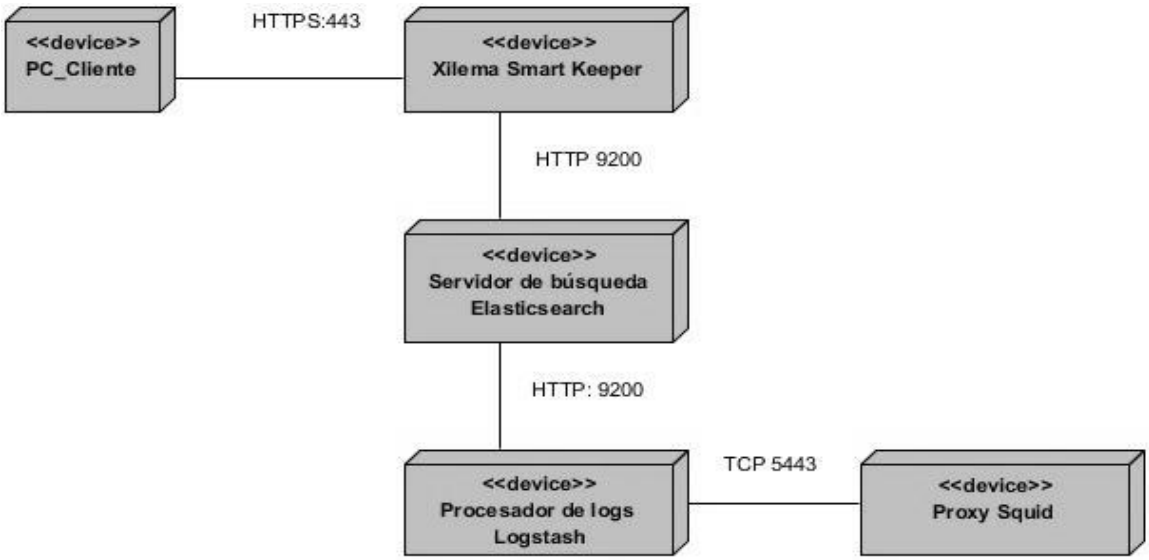


Figura 7. Modelo de despliegue.

En el nodo PC_Cliente se originan las peticiones al sistema Xilema Smart Keeper ubicado en el servidor de aplicaciones web a través del protocolo HTTPS. El sistema mantiene comunicación constante con el servidor de búsqueda Elasticsearch, el cual contiene los datos necesarios para generar los reportes, esta comunicación se realiza a través del protocolo HTTP, puerto 9200. El procesador de logs Logstash toma los logs del *proxy* Squid a través del protocolo TCP puerto 5443 y los envía a Elasticsearch por el protocolo HTTP, puerto 9200.

Conclusiones parciales

- Finalizado el flujo de análisis y diseño de la propuesta de solución se arriba a las siguientes conclusiones:
- La identificación de las características y responsabilidades con las debe cumplir el sistema propuesto permitió la definición de los requisitos de software.
- El adecuado análisis de la descripción de la propuesta de solución, modelo conceptual, así como el apoyo en los artefactos generados, proporcionó una satisfactoria implementación de una primera versión de los requisitos identificados.

- La identificación de los patrones de diseño y estilo arquitectónico proporcionaron la implementación de una aplicación sólida. A través de los diagramas de clase y secuencia se estableció una visión del funcionamiento físico y lógico del módulo.

Capítulo 3: Implementación y validación del módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper.

En el presente capítulo se describe la etapa de implementación y codificación del módulo, se materializan en el componente la arquitectura, patrones de diseño establecidos y artefactos generados en la fase de análisis y diseño. Así como la ejecución de pruebas posteriores al desarrollo cuyo objetivo es asegurar que la solución propuesta funcione en consecuencia con los requisitos definidos.

3.1 Diagrama de componentes:

Los diagramas de componentes describen los elementos físicos del sistema y sus relaciones. Muestran las opciones de realización incluyendo código fuente, binario y ejecutable. Los componentes representan todos los tipos de elementos *software* que entran en la fabricación de aplicaciones informáticas. Pueden ser simples archivos, paquetes de Ada[13], bibliotecas cargadas dinámicamente, etc (Microsoft, 2018). Según la arquitectura basada en el *framework* Symfony, la figura 9 muestra el diagrama de componentes del módulo generador de reportes de Xilema Smart Keeper, en la Tabla 4 se describen los componentes del diagrama:

Tabla 4. Descripción de los componentes del Diagrama de Componentes.

Componentes					Descripción
Xilema Smart Keeper	Módulo generador de reportes estadísticos	Modelo	SquidLogs.php		Clase entidad que almacena los datos de los logs del *proxy* Squid.
		Controlador	ReporteController.php		Clase controladora que contiene las acciones relacionadas con la generación de los reportes.
			Formularios	DominioType.php	Clase responsable de construir los formularios relacionados con los reportes por dominio y por

[13] Ada: Lenguaje imperativo, descendiente de Pascal.

					usuario.
		Vista	consumo_dias.html.twig		Muestra reporte en gráfica de barras de los días de mayor consumo.
			navegacion_dias.html.twig		Muestra reporte en gráfica de barras de los días de mayor navegación atendiendo a la cantidad de peticiones.
			consumo_ip.html.twig		Muestra reporte en gráfica de barra de los 10 IP de mayor consumo.
			consumo_ip_mas.html.twig		Muestra reporte en forma de tabla del consumo por IP.
			consumo_usuarios.html.twig		Muestra reporte en gráfica circular de los 10 usuarios de mayor consumo,
			consumo_usuarios_mas.html.twig		Muestra reporte en forma de tabla del consumo por usuario.
			contenidos_mas_descargados.html.twig		Muestra reporte en gráfica circular de la cantidad de peticiones agrupados por tipo de contenido descargado.

			estados_respuestas.html.twig	Muestra reporte en gráfica circular de la cantidad de peticiones agrupados por estados de respuesta.
			horarios_mayor_navegacion.html.twig	Muestra reporte en gráfica de barras de los horarios de mayor navegación atendiendo a la cantidad de peticiones.
			metodos_de_conexion.html.twig	Muestra reporte en gráfica circular de la cantidad de conexiones agrupadas por métodos de conexión
			sitios_visitados.html.twig	Muestra reporte en gráfica circular de los 10 sitios más visitados atendiendo a la cantidad de peticiones.
			sitios_visitados_mas.html.twig	Muestra reporte en forma de tabla de la cantidad de peticiones realizadas agrupadas por dominio.
			usuarios_por_dominio.html.twig	Muestra un formulario para obtener qué usuarios o IP han visitado un determinado dominio.
			dominios_por_usuario.html.twig	Muestra un formulario para obtener los dominios accedidos desde un determinado usuario o IP.

			base.html.twig	Plantilla base, contiene los CSS y JavaScript, de la cual heredan todas las demás plantillas del módulo.
			layout.html.twig	Plantilla que contiene toda la estructura de diseño del módulo.
	config.yml			Contiene las configuraciones generales del sistema Xilema Smart Keeper.

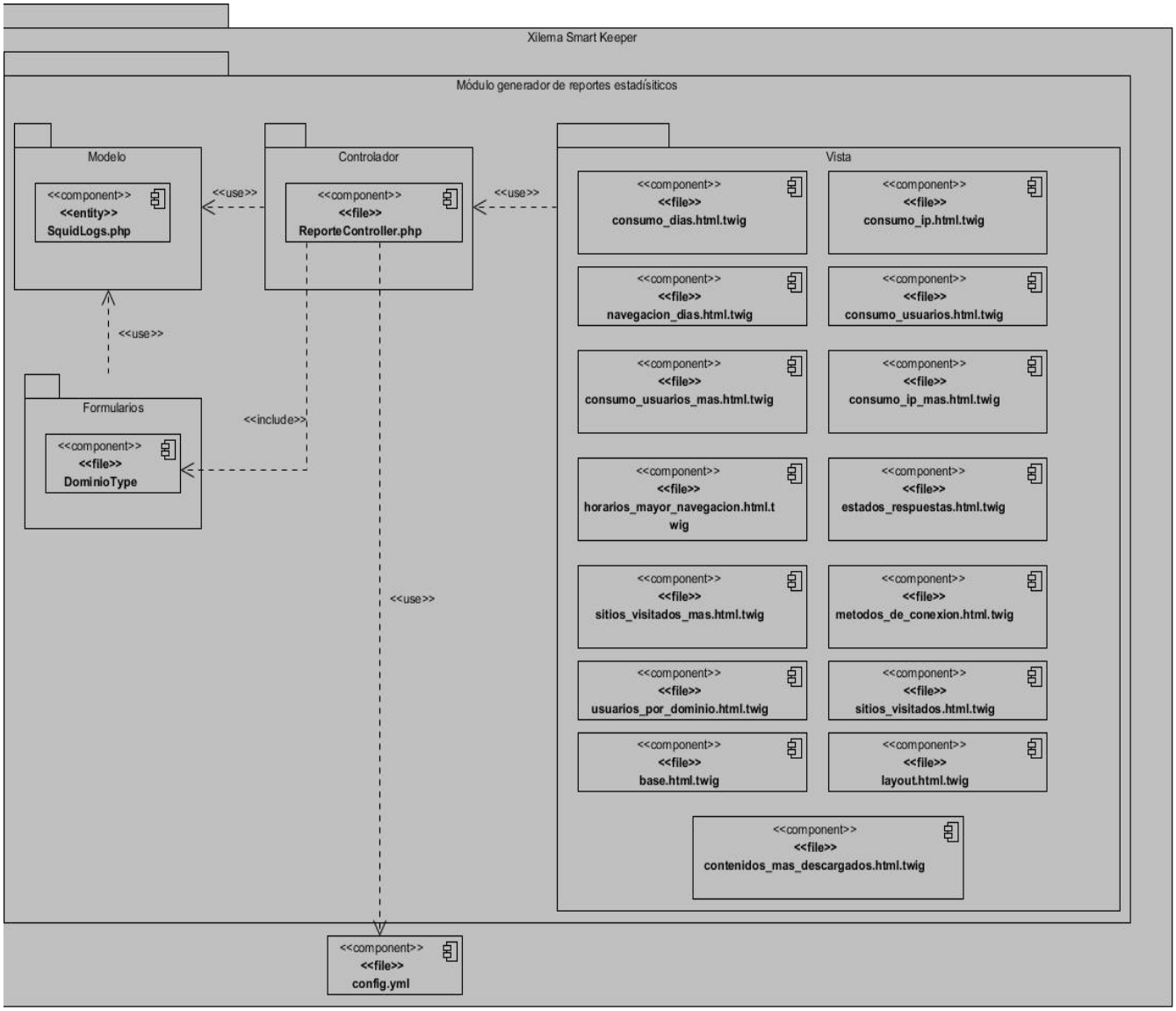


Figura 8. Diagrama de componentes

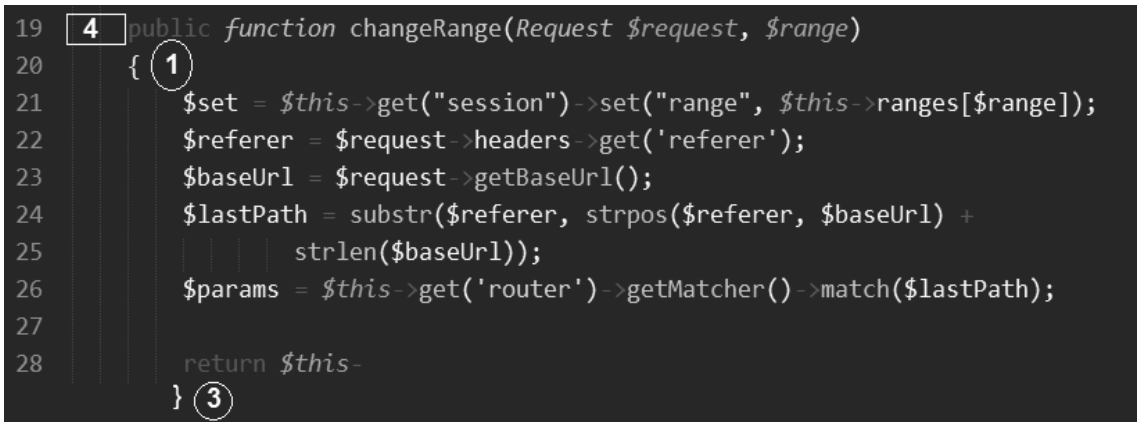
3.2 Estándares de codificación

Los estándares de codificación son un conjunto de reglas no formales, que ayudan con la creación de código para hacerlo más legible y fácil de entender. Constituyen una forma de normalizar la programación de forma tal que, al trabajar en un proyecto, cualquier persona involucrada en el mismo tenga acceso y comprenda el código. A continuación de definen los estándares de codificación utilizados durante la

implementación del módulo, basándose en los utilizados por el lenguaje *PHP* para el marco de trabajo *Symfony* (PHP-FIG, 2018).

Organización del código

1. La apertura de llaves para clases y funciones deben estar en la próxima línea después de su declaración.
2. La apertura de llaves en las estructuras de control debe estar en la misma línea.
3. El cierre de llaves para clases, funciones y estructuras de control deben estar en la próxima línea después del bloque de código.
4. La identación se realiza con 4 espacios y no con tabulación.




```
19    public function changeRange(Request $request, $range)
20    {
21        $set = $this->get("session")->set("range", $this->ranges[$range]);
22        $referer = $request->headers->get('referer');
23        $baseUrl = $request->getBaseUrl();
24        $lastPath = substr($referer, strpos($referer, $baseUrl) +
25                strlen($baseUrl));
26        $params = $this->get('router')->getMatcher()->match($lastPath);
27
28        return $this-
          }
```

Líneas

5. Se deberá agregar una línea en blanco delante de la sentencia *return*.
6. Se puede agregar líneas en blanco para mejorar la legibilidad e indicar bloques de códigos relacionados.
7. Debe haber una declaración por línea.

```
1    public function navegacionPorDiasAction(Request $request)
2    {
3    $res = \AppBundle\ElasticSearchManager\ElasticMg::navegacionPorDias($this->hosts,
     $this->get("session")->get("range", "now-1d"), 10);
4
5    return $this->render('default/navegacion_dias.html.twig', array("data" => $res));
6    }
```

Codificación

8. Los archivos *PHP* deben usar solo las etiquetas de apertura *<?php* omitiendo las etiquetas de cierre *?>*
9. Utilizar la codificación *UTF-8.*

Espacios en blanco en expresiones y sentencias

10. Se debe utilizar un espacio en blanco después de la palabra clave en las estructuras de control.
11. No deben existir espacios en la apertura y cierre de paréntesis.
12. Debe usarse un espacio en blanco entre los operadores lógicos, aritméticos, de comparación y asignación

```
1    if ($request->isMethod('POST')) {
2        $form->handleRequest($request);
3        $data = $form->getData();
4        if ($data["ip"]) {
5            $res =
             \AppBundle\ElasticSearchManager\ElasticMg::ipPorDominiosSearchAction($this->
             hosts, $this->get("session")->get("range", "now-1d"), $data["domain"]);
6
```

Convenciones de Nombramientos

13. Se debe usar el estilo de escritura *camelCase* para la declaración de variables y no guiones bajos para separar palabras.
14. Se debe usar el estilo de escritura *StudlyCaps* para los nombres de clases.

```
4  class ReporteController extends Controller
5  {                              14
6      /**
7       * @Route("/change/{range}", name="change_range")
8       */
9      >redirect($this->generateUrl($params['_route']));
10     }
11     public function changeRange(Request $request, $range)
12     {
13         $set = $this->get("session")->set("range", $this->ranges[$range]);
14         $referer = $request->headers->get('referer');
15         $baseUrl = $request->getBaseUrl();
16 13      $lastPath = substr($referer, strpos($referer, $baseUrl) +
17                     strlen($baseUrl));
18         $params = $this->get('router')->getMatcher()->match($lastPath);
19
20         return $this-
```

3.3 Validación de la propuesta de solución

La fase de validación es el proceso de evaluar la calidad del *software*, con el objetivo de detectar posibles errores y corregirlos enfocándose en la lógica interna y los requerimientos especificados según (ISO/IEC/IEEE International Standard, 2017). A continuación, se presentan los resultados de las pruebas realizadas al generador de reportes estadísticos del sistema Xilema Smart Keeper, con el objetivo de garantizar su calidad y correcto funcionamiento.

3.3.1 Pruebas funcionales

Un aspecto crucial en el control de calidad del desarrollo de *software* son las pruebas y, dentro de estas, las pruebas funcionales, en las cuales se hace una verificación dinámica del comportamiento de un sistema. Las pruebas funcionales están basadas en la ejecución, revisión y retroalimentación de las funcionalidades previamente diseñadas en el *software*. Se centran en comprobar que el sistema desarrollado funcione acorde a las especificaciones funcionales y requisitos del cliente, permitiendo detectar los defectos derivados de errores en la fase de implementación (González Palacio, 2014). Para este tipo de prueba, el autor selecciona la técnica de Caja negra, según (Pressman, 2010), la prueba de caja negra se refiere a las pruebas que se llevan a cabo en la interfaz del *software*. Una prueba de caja negra examina algunos aspectos fundamentales de un sistema con poca preocupación por la estructura lógica interna del *software.* Las pruebas se aplican sobre el módulo empleando un determinado conjunto

de datos de entrada y observando las salidas que se producen para determinar si la función se está desempeñando correctamente por el sistema bajo prueba (Garrido Tejero, 2016). Para aplicar las pruebas funcionales al módulo se diseñaron 19 Casos de Prueba (CP) que corresponden a los requisitos funcionales de prioridad alta. A continuación, se muestran 2 CP correspondientes a los RF3 y RF11 respectivamente.

Tabla 5. Descripción de las variables para el Caso de Prueba 1.

Nº	**Nombre del campo**	**Clasificación**	**Valor Nulo**	**Descripción**
1	Rango de fecha	Campo de selección	No	Puede seleccionar uno de los siguientes valores: Última hora, Último día, Última semana, Últimos 6 meses, Último año

Tabla 6. Caso de Prueba del RF3.

Caso de Prueba 2: SC RF3_Generar reporte de sitios más visitados				
Escenario	**Descripción**	**1**	**Respuesta del sistema**	**Flujo central**
EC 2.1 Mostrar reporte de los 10 sitios más visitados de forma correcta.	Mostrar reporte en forma de gráfica de pastel de los 10 sitios más visitados.	V Últimos 6 meses	Muestra una gráfica de pastel con los 10 sitios más visitados y la cantidad de visitas	1. En el menú superior derecho el usuario selecciona el intervalo de tiempo en el cual desea realizar el reporte. 2. En el menú lateral del sistema el usuario accede a la opción Sitios más visitados.
EC 2.2 Mostrar reporte de los 10	El módulo no genera	V	Muestra un mensaje de error "No se	

sitios más visitados en un intervalo de tiempo del cual no se tengan datos.	reporte.	Última hora	encontraron datos para el intervalo seleccionado".	3. El sistema muestra el reporte en forma de grafica de pastel de los 10 sitios más visitados en el intervalo seleccionado y la cantidad de visitas.

Tabla 7. Descripción de las variables para el Caso de Prueba 2.

Nº	Nombre del campo	Clasificación	Valor Nulo	Descripción
1	Rango de fecha	Campo de selección	No	Puede seleccionar uno de los siguientes valores: Última hora, Último día, Última semana, Últimos 6 meses, Último año
2	Dominio	Campo de texto	No	Campo de texto que permite todos los caracteres.
3	Mostrar IP	Campo booleano	Si	Puede ser seleccionado o no.

Tabla 8. Caso de Prueba del RF11.

Caso de Prueba 2: SC RF11_Generar reporte de usuarios por dominio.						
Escenario	**Descripción**	**1**	**2**	**3**	**Respuesta del Sistema**	**Flujo Central**
EC 1.1 Mostrar usuarios por dominio de forma	El módulo muestra el reporte de los usuarios que han	V Últimos 6 meses	V Cubadebate.cu	V Falso	Muestra una tabla con los usuarios que han accedido al	1. En el menú superior derecho el usuario selecciona el intervalo de tiempo en el cual desea

correcta.	accedido a un determinado dominio.				dominio así como la cantidad de visitas.	realizar el reporte. 2. En el menú lateral del sistema el usuario accede a la opción Usuarios por dominio. 3. El usuario en el formulario mostrado introduce el dominio. 4. El usuario ejecuta el botón Generar reporte 5. El sistema muestra el reporte de los usuarios que han accedido a ese dominio.
EC 1.2 Mostrar usuarios por dominio con campo vacío	El módulo no genera reporte.	V Últimos 6 meses	I	V Falso	Muestra un mensaje de error "El campo no puede estar vacío".	
EC 1.3 Mostrar usuarios por dominio sin visitas.	El módulo no genera reporte.	V Últimos 6 meses	V midominio.com	V Falso	Muestra un mensaje de error "No se encontraron resultados para el domino buscado"	

Resultado de las pruebas funcionales

En la primera iteración de las pruebas funcionales se obtuvo un total de 12 no conformidades, divididas en 4 de funcionalidad y 8 de ortografía, quedando corregidas las 12 no conformidades. En una segunda iteración surgieron otras 2 de funcionalidad, las cuales fueron satisfactoriamente solucionadas. En la tercera iteración no se detectó ninguna no conformidad.

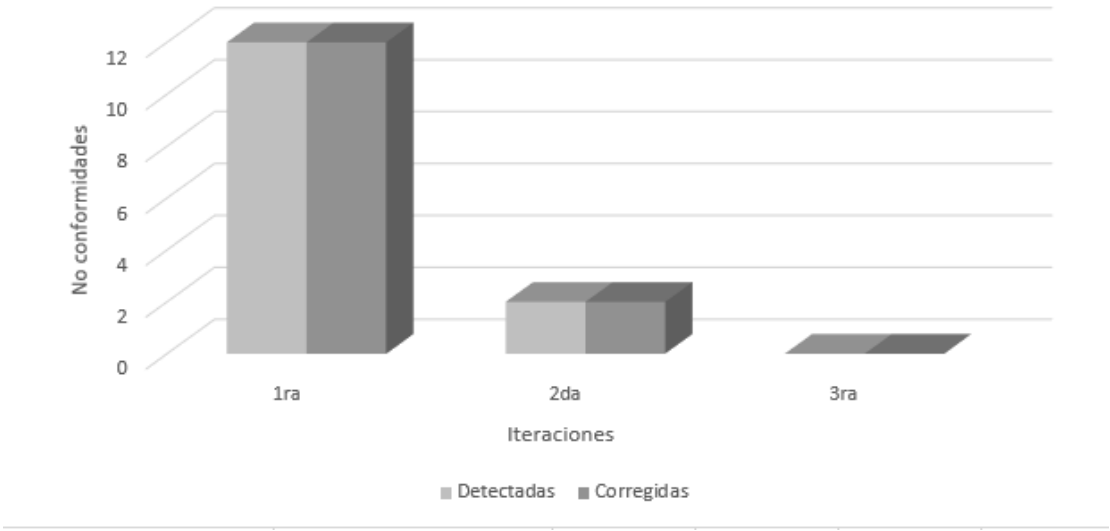


Figura 9. Resultados de las pruebas funcionales.

Las no conformidades funcionales, están relacionadas con la respuesta del sistema ante peticiones del usuario. Las estadísticas mostradas no estaban en correspondencia con los datos reales del intervalo seleccionado, la presencia de esta no conformidad se debe a que el controlador no enviaba el intervalo correcto. Además, no se mostraba mensajes de error en caso de no existir información del rango fecha seleccionado, para ello se incluyeron dichos mensajes en el código fuente.

3.3.2 Pruebas de integración

Las pruebas de integración se realizan con el objetivo de verificar el correcto ensamblaje entre los distintos componentes una vez que han sido probados unitariamente con el fin de comprobar que interactúan correctamente a través de sus interfaces, tanto internas como externas, cubren la funcionalidad establecida y se ajustan a los requisitos no funcionales especificados en las verificaciones correspondientes (Gómez Rodríguez, 2015).

Una vez realizadas las pruebas funcionales a cada componente interno de manera independiente, y verificado que las funcionalidades implementadas se corresponden de acuerdo a los requisitos funcionales y no funcionales establecidos; se pudo comprobar el correcto funcionamiento de los componentes mediante el estudio del flujo de datos entre ellos. Posterior a estas pruebas, se hace necesaria la

realización de pruebas de integración, con la finalidad de validar la compatibilidad y el funcionamiento de las interfaces que comunican las diferentes partes que componen la solución.

Algunas de las acciones que se llevaron a cabo para la realización de estas pruebas fueron:

- Comprobación del funcionamiento del enlace entre el sistema Xilema Smart Keeper y el Módulo generador de reportes estadísticos.
- Verificación de la conexión entre el módulo y el motor de búsqueda Elasticsearch.
- Validación de las rutas de acceso, en el *routing* principal del sistema Xilema Smart Keeper.

Mediante la ejecución de las pruebas de integración se logró verificar la operación conjunta de cada uno de los componentes Módulo generador de reportes estadísticos, junto a los componentes que forman el sistema Xilema Smart Keeper y la adecuada conexión del sistema con el motor de búsqueda Elasticsearch.

3.3.3 Pruebas de carga y estrés

Las pruebas de carga y estrés son una clase de pruebas que se implementan y se ejecutan para caracterizar y evaluar las características relacionadas con el rendimiento del destino de la prueba, como los perfiles de tiempo, el flujo de ejecución, los tiempos de respuesta y la fiabilidad y los límites operativos (Mera Paz, 2016).

En el diseño del plan de pruebas de rendimiento para el módulo, se tuvo en cuenta, las diferentes acciones que el usuario administrador puede realizar para visualizar los principales reportes generador por el módulo. En una muestra de 1000 peticiones, la herramienta JMeter generó el resultado mostrado en la tabla 9.

Tabla 9. Resultados de las pruebas de rendimiento mediante JMeter.

Petición	Cantidad de Peticiones	Tiempo medio de respuesta(ms)	Tasa de error(%)
Generar reporte de estados de respuesta.	1000	85	0
Generar reporte de consumo por	1000	92	0

IP.			
Generar reportes de contenidos más descargados.	1000	92	0
Generar reporte de horarios de mayor navegación.	1000	91	0
Generar reporte de navegación por día, basada en consumo.	1000	94	0
Total	5000	91	0

Como se puede observar en el análisis del resumen arrojado por la herramienta JMeter, para un total de 1000 peticiones que se le realizaron al módulo por un usuario administrador, se alcanzó un rendimiento de 5000 peticiones en 454 milisegundos, con un porcentaje de 0% de errores para cada petición realizada. Como se puede apreciar en la tabla 9, la media de respuesta por cada petición es de 91 milisegundo. Según (Nielsen, 1993) 0,1 segundo es aproximadamente el límite para que el usuario sienta que el sistema está reaccionando instantáneamente, lo que significa que no es necesaria ninguna retroalimentación especial, excepto para mostrar el resultado, por lo que es posible afirmar que el Módulo generador de reportes estadísticos responde correctamente ante situaciones de carga y estrés en función de la muestra utilizada.

El ambiente de prueba es un parámetro que ejerce cierta influencia sobre el tiempo de respuesta del sistema, puede elementos que pueden provocar cierto retraso en la generación de la carga en las pruebas e interferencia en la obtención de los resultados. El despliegue fue simulado en una máquina con Ubuntu 16.04 como sistema operativo de 64 bit con 6 GB de RAM y 4 procesadores, a una velocidad de 1.7 GHz.

3.3.4 Aplicación del Criterio de Expertos.

Para evaluar la fiabilidad del módulo desarrollado se emplea el Criterio de Expertos mediante el método Delphi. Según (Torrado y otros, 2016) el método Delphi es una técnica de recogida de información que permite obtener la opinión de un grupo de expertos a través de la consulta reiterada. Esta técnica, de carácter cualitativo, es recomendable cuando no se dispone de información suficiente para la toma de decisiones o es necesario, para nuestra investigación, recoger opiniones consensuadas y representativas de un colectivo de especialistas.

Durante aplicación del Método Delphi se emplearán los siguientes pasos:

- Identificación y selección de posibles expertos.
- Aplicación de encuestas a expertos.
- Valoración de la información obtenida.

Identificación y selección de posibles expertos

Se identificaron 4 expertos, quienes son especialistas desarrolladores del sistema AiresProxy y primeras versiones de Xilema Smart Keeper. Para valorar la experiencia con que cuentan, se elabora y aplica un cuestionario de autoevaluación del nivel de información acerca del presente tema (Ver anexo). A continuación, se presentan los resultados de cada uno de los expertos:

Tabla 10. Resultado de los expertos.

Expertos	Nivel de conocimiento o información del tema										Kc
	1	2	3	4	5	6	7	8	9	10	
Rubén Reynaldo Bonachea	8	9	9	10					x		0.9
Miguel Ángel Chávez Alfonso	8	8	9	10					x		0.85
Goar Espinosa Marrero	9	8	7	9				x			0.825
Evelyn Labrada Oduardo	7	8	10	9				x			
Eddy Fonseca Lahens	8	8	8	9				x			0.825

Para calcular el Coeficiente de Conocimiento o Información (Kc) del experto se utiliza la siguiente ecuación:

$$Kc = n(0{,}1)$$

Siendo *n* el nivel de conocimiento del experto. Son necesarios, además, una serie de factores para determinar el coeficiente de argumentación del experto, los cuales se relacionan en la tabla siguiente:

Tabla 11. Factores para determinar el coeficiente de argumentación del experto.

Fuentes de Argumentación	Alto	Medio	Bajo
Análisis teóricos realizados por usted	0.3	0.2	0.1
Experiencia obtenida	0.5	0.4	0.2
Trabajos de autores nacionales	0.05	0.05	0.05
Trabajos de autores extranjeros	0.05	0.05	0.05
Su conocimiento del estado del problema en el extranjero	0.05	0.05	0.05
Su Intuición	0.05	0.05	0.05

El Coeficiente de Argumentación (Ka) del experto se calcula con la ecuación:

$$Ka = \sum_{i=1}^{6} n_i$$

Siendo n_i es el valor obtenido en la fuente de argumentación *i* (de 1 hasta 6). Al aplicar la ecuación se obtienen los siguientes resultados:

- Rubén Reynaldo Bonachea $Ka = 1$
- Miguel Ángel Chávez Alfonso $Ka = 1$
- Goar Espinosa Marrero $Ka = 0.8$
- Evelyn Labrada Oduardo $Ka = 0.8$
- Eddy Fonseca Lahens $Ka = 0.8$

Con los Coeficientes de Conocimiento (Kc) y Argumentación (Ka) se puede obtener finalmente el Coeficiente de Competencia (K) para seleccionar los expertos a trabajar en la investigación. Este es calculado con la ecuación:

$$K = 0{,}5(Kc + Ka)$$

Los resultados obtenidos se muestran en la siguiente tabla:

Tabla 12. Coeficiente de competencia de los expertos.

Expertos	K	Valoración
Rubén Reynaldo Bonachea	0.95	Alto
Miguel Ángel Chávez Alfonso	0.88	Alto
Goar Espinosa Marrero	0.81	Alto
Evelyn Labrada Oduardo	0.81	Alto
Eddy Fonseca Lahens	0.81	Alto

Para la valoración de los resultados obtenidos se tuvo en cuenta que:

- K es alto si $0{,}8 < K < 1$
- K es medio si $0{,}5 < K < 0{,}8$
- K es bajo si $K < 0{,}5$

Luego de seleccionar los expertos que contaran con un nivel de experiencia y dominio del tema alto, el panel de experto quedó conformado de la siguiente manera:

Tabla 13. Panel de expertos seleccionados.

Nombre y Apellidos	Entidad	Años de Experiencia
Rubén Reynaldo Bonachea	Centro de Ideoinformática (UCI)	6
Miguel Ángel Chávez Alfonso	Centro de Ideoinformática (UCI)	8
Goar Espinosa Marrero	Universidad de las Ciencias Informáticas	6
Evelyn Labrada Oduardo	Centro de Ideoinformática (UCI)	5
Eddy Fonseca Lahens	Dirección de proyectos especiales	7

El panel de expertos seleccionado para trabajar en la validación idea a defender en la presente investigación es sometido a una encuesta (Ver anexo 4) con el objetivo de evaluar el Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper. En la tabla se muestran los resultados emitidos a partir de las siguientes sentencias:

1. Fiabilidad de las estadísticas mostradas
2. Relevancia de la información mostrada en los reportes
3. Rapidez en la respuesta de peticiones.
4. Presentación de una interfaz amigable e intuitiva para el usuario.

Tabla 14. Resultado de las encuestas realizadas a los expertos

Sentencias	Categorías Evaluativas				Total
	MA	A	PA	I	
1	0	5	0	0	5
2	1	4	0	0	5
3	3	2	0	0	5
4	2	3	0	0	5

Para el procesar y analizar de la información obtenida se clasifican las respuestas en las siguientes categorías evaluativas: muy adecuado (MA), adecuado (A), poco adecuado (PA), inadecuado (I).

Luego de obtener los resultados de los criterios de expertos, se realizan los siguientes pasos hasta llegar a la conclusión de valoración de cada sentencia

- Obtención de la tabla de frecuencia acumulada.
- Obtención de la tabla de frecuencia relativa acumulativa.
- Asignación del valor de la imagen que corresponde a cada frecuencia relativa acumulativa, por la inversa de la curva normal (Tabla Z de la distribución normal) y obtención de puntos mediante el cálculo N-P para el promedio relativo.

Tabla 15. Frecuencia acumulada de los datos primarios obtenidos

Sentencias	Categorías Evaluativas

	MA	A	PA	I
1	0	5	5	5
2	1	5	5	5
3	3	5	5	5
4	2	5	5	5

Tabla 16. Frecuencia relativa acumulativa de los datos primarios obtenidos

Sentencias	Categorías Evaluativas	
	MA	A
1	0	1
2	0.2	1
3	0.6	1
4	0.4	1

Tabla 17. Imagen de la frecuencia relativa acumulativa de los datos primarios obtenidos

Sentencias	Categorías Evaluativas		Suma	Promedio	Promedio relativo
	MA	A			
1	3.49	3.49	6.98	3.49	-1.28
2	-0.75	3.49	2.74	1.37	0.83
3	0.27	3.49	3.76	1.88	0.96
4	-0.25	3.49	3.24	1.62	0.78
Puntos de Corte	0.69	3.49	17.65		

Luego de analizar los promedios relativos, se observa que 3 de ellos exceden el punto de corte de la categoría MA y 1 se encuentra por debajo, lo que quiere decir que los expertos, con un nivel de concordancia de 75% clasifican con el nivel de Adecuadas las sentencias evaluadas, apoyando la idea a defender planteada, lo cual evidencia la calidad del Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper.

Conclusiones parciales

Una vez finalizada la fase de implementación y validación es posible concluir que:

- La estrategia de pruebas trazadas permitió solucionar los errores cometidos durante el desarrollo
- del módulo, garantizando la calidad del módulo.
- El Criterio de Expertos, demostró, con una concordancia de 75% de los expertos, el módulo desarrollado es adecuado y cumple con los requisitos establecidos

Conclusiones

De manera general se puede concluir sobre la presente investigación:

- El estudio del arte de los *software* generadores de reportes estadísticos de la navegación en Internet, permitió demostrar que los ya existente a nivel nacional e internacional no satisfacen las necesidades requeridas para el sistema Xilema Smart Keeper.
- La elaboración de los artefactos propuestos por la metodología de desarrollo y el levantamiento de requisitos permitieron una mayor comprensión del módulo a desarrollar, así como la identificación de los procesos y características del mismo.
- Con la realización de las pruebas de *software* se solucionaron los errores cometidos durante el desarrollo del módulo, garantizando así una solución funcional, integrable y con la calidad requerida.
- Se demostró a través de la aplicación del Criterio de Expertos, con una concordancia de 75% que el Módulo generador de reportes estadísticos para el sistema Xilema Smart Keeper es adecuado, cumple con los requisitos establecidos y a través del empleo de gráficas para mostrar sus resultados facilita al usuario administrador el análisis de los datos.

Recomendaciones

Para el desarrollo de futuras investigaciones relacionadas con la presente se recomienda:

- Adicionar una funcionalidad que permita tomar los datos automáticamente de Squid por Logstash.
- Adicionar otros tipos de reportes estadísticos de navegación.
- Permitir en el caso de las gráficas de barra y circular, obtener información detallada cuando se seleccione un punto en específico. Ejemplos: En el caso de los estados de respuesta, si se selecciona el punto que representa un estado determinado, permita ver los accesos con dicha respuesta. En el caso de la navegación por día, al seleccionar el punto que representa un día determinado brinde información de todos los dominios pedidos en esa fecha.

Bibliografía

ESTEBAN, GABRIEL, MAIDA y otros. *Metodologías de desarrollo de software.* Pontificia Universidad Católica Argentina. [en línea] 2015. [Fecha de consulta: 15 de diciembre de 2017.]. Disponible en: http://bibliotecadigital.uca.edu.ar/greenstone/cgi-bin/library.cgi?a=d&c=tesis&d=metodologias-desarrollo-software

NetBeans. [en línea] 2017. [Fecha de consulta: 17 de diciembre de 2017]. Disponible en: https://netbeans.org

MERA, PAZ , JULIÁN . *Análisis del proceso de pruebas de calidad de software.* 20, Colombia : Universidad Cooperativa de Colombia. 25 de julio de 2016, Vol. 12. 2357-6014

Apache Software Fundation. 2017. *Apache JMeter™.* [en línea] 2017. [Fecha de consulta: 22 de diciembre de 2017.]. Disponible en: http://jmeter.apache.org/

arxiv.org. 2013. Cornell University Library. [en línea] 2013. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: https://arxiv.org/abs/1312.5429

awstats. 2016. AWStats official web site. *AWStats logfile analyzer 7.6 Documentation.* [en línea] 2016. [Fecha de consulta: 11 de noviembre de 2017.]. Disponible en: http://www.awstats.org/docs/index.html

CORTÉS, GLORIA y CASALLAS, RUBBY. Introducción a los patrones de Software. s.l. : Universidad de los Andes, 2014

CRAIG, LARMAN. UML y patrones: una introducción al análisis y diseño orientado a objetos y al proceso unificado. Madrid : Pearson Educación, 2003. 84-205-3438-2

Digital Guide. 1&1 Digital Guide. *Ficheros log: Toda la información de registro en un archivo.* [en línea] 2016. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: https://www.1and1.es/digitalguide/online-marketing/analisis-web/el-log-el-archivo-de-registro-de-procesos-informaticos/

Dspace. Repositorio Dspace. [en línea] 2017. [Fecha de consulta: 11 de noviembre de 2017.] http://www.dspace.espol.edu.ec/xmlui/handle/123456789/41170

TORRADO, FONSECA, MERCES y REGUANT, ÁLVAREZ, MERCEDES. *El método Delphi***.** 87-102, Barcelona : REIRE, 2016, Vol. 9

ELDER, ROBERT. *The Logstash book: log management made easy.* Montana : s.n., 2017. 978-1975795238

Elmanytas. Página web de elmanytas. [en línea] 2014. [Fecha de consulta: 8 de julio de 2014]. Disponible en: http://elmanytas.es/?q=node/318

EMRE, ERTURK, ANGEL RAJAN. *Web Vulnerability Scanners: A Case Study.* Cornell University Library. [en línea] 2017. [Fecha de consulta: 20 de diciembre de 2017.]. Disponible en: https://arxiv.org/abs/1706.08017

FUENTES, GONZÁLEZ, MÓNICA CECILIA. *Introducción al modelado de datos.* Universidad Autónoma del Estado de México. [en línea] 2015. [Fecha de consulta: 2 de marzo de 2018.]. Disponible en: http://ri.uaemex.mx/handle/20.500.11799/34276

FUENTES, LÓPEZ, MARÍA DEL CARMEN. Notas de clase: Análisis de requerimientos. Tlalpan : Universidad Autónoma Metropolitana, 2011

GARCÍA, ACEVEDO , CARLOS ALBERTO. Guía Pedagógica Arquitectura de software. s.l. : Unidad Académica Profesional Tianguistenco , 2015

GARRIDO, TEJERO, ANTONIO. *Pruebas de caja negra. Técnica de Partición equivalente.* Valencia : Universidad Politécnica de Valencia, 2016

GitHub. *Git 2.8 has been released.* [en línea] 2016. [Fecha de consulta: 20 de noviembre de 2017.]. Disponible en: https://github.com/blog/2131-git-2-8-has-been-released

GitHub. *dradis-acunetix.* [en línea] 2017. [Fecha de consulta: 20 de diciembre de 2017.]. Disponible en: https://github.com/dradis/dradis-acunetix

GÓMEZ, RODRÍGUEZ, NURIA. Las pruebas de Integración como proceso de calidad del software. *Universidad de Carlos III.* [en línea] 2015. [Fecha de consulta: 28 de abril de 2018.]. Disponible en: hdl.handle.net/10016/25775

GONZÁLEZ, PALACIO, LILIANA. *Método para generar casos de prueba funcional en el desarrollo de software.* Colombia : Universidad de Medellín, 2014. 15 Sup.1

HERNÁNDEZ MENDOZA, SANDRA LUZ, *La estadísitica.* Universidad Autónoma del Estado de Hidalgo [en línea] 2018. [Fecha de consulta: 18 de mayo de 2018.]. Disponible en https://repository.uaeh.edu.mx/revistas/index.php/icea/article/view/2711

Hindawi. International Journal of Digital Multimedia Broadcasting. [en línea] 2013. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: http://dx.doi.org/10.1155/2013/471683

Dle.rae.es. Real Academia Española. [en línea] 2018. [Fecha de consulta: 10 de enero de 2018.]. Disponible en: http://dle.rae.es/?id=W3HQJvs

ISO/IEC/IEEE International Standard. *Systems and software engineering-Vocabulary.* New York : ISO/IEC/IEEE, 2017. 24765

KNIGHT-DAVIS, Stacey. *Using AWStats to Analyze Logs from EZProxy and from the Public OPAC Logs.* The Keep, Eastern Illinois University. [en línea] 2017. [Fecha de consulta: 11 de noviembre de 2017.]. Disponible en: http://thekeep.eiu.edu/lib_fac/228/

KOIVULAHTI-OJALA, MERVI. *On UML modeling tool evaluation, use and training.* University of Jyvaskyla. [en línea] 2017. [Fecha de consulta: 20 de noviembre de 2017.]. Disponible en: https://jyx.jyu.fi/dspace/handle/123456789/56043

Manage Engine. *Firewal Analizer.* [en línea] 2017. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: https://www.manageengine.com/products/firewall/squid-proxy-report.html

Manage Engine. *Análisis de Registros de Firewall.* [en línea] 2017. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: http://www.manageengine.mx/firewall/firewall-analyzer-vision-general.html

Manage Engine. *Firewall Analyzer.* [en línea] 2017. [Fecha de consulta: 15 de diciembre de 2017.]. Disponible en: https://www.manageengine.com/products/firewall/fortigate-firewall-analyzer.html

Manage Engine. *Firewall Analysis: Analyze Security, Traffic Logs and Configurations, Policies, Rules.* [en línea] 2017. [Fecha de consulta: 18 de diciembre de 2017.]. Disponible en: https://www.manageengine.com/products/firewall/firewall-analysis.html

Marco de desarrollo de la Junta de Andalucía. *Metodología para la elicitación de requisitos.* [en línea] 2018. [Fecha de consulta: 18 de febrero de 2018.]. Disponible en: http://www.juntadeandalucia.es/servicios/madeja/contenido/recurso/407

Microsoft. *Diagramas de componentes de UML: Referencia.* 2018

MUTUBERRÍA, CARLOS JAVIER PÉREZ y VÁZQUEZ, LEÓN, JORGE LUIS. Analizador de Registros de Servidores Proxy. La Habana, Cuba : Universidad de las CIencias Informáticas, 2009

NIELSEN, JAKOB. Usability Engineering. 1993. 7-111-14792-8

OUTLAW, ROBERT. Microsoft. *Visual Studio.* [en línea] 2017. [Fecha de consulta: 20 de noviembre de 2017.]. Disponible en: https://www.visualstudio.com/es/learn/what-is-version-control

BARRERA, PALENZUELA, OTNIEL. XIV Forum de Ciencia y Técnica. *ISAWeb Monitoreo de Tráfico en Internet.* La Habana, Cuba : ICID, 2006

PONCE DE LEÓN, LEIVA, ROSSANA y otros. *Participción de la familia en la sociedad virtual: Conocimiento sobre uso y riesgos de internet.* Buenos Aires : Suplemento Signos EAD, 2016

PHP. *¿Qué es PHP? - Manual.* [en línea] 2017. [Fecha de consulta: 22 de noviembre de 2017.]. Disponible en: php.net/manual/es/intro-whatis.php

PHP-FIG. 2018. PHP Framework Interop Group. [en línea] 2018. [Fecha de consulta: 19 de noviembre de 2017]. Disponible en: https://www.php-fig.org/psr/

PHPStorm. PHPStorm. *Enjoy Productive PHP.* [en línea] 2017. [Fecha de consulta: 28 de noviembre de 2017.]. Disponible en: https://www.jetbrains.com/phpstorm/

PORTILLO, LENIS ROSSI WONG. Un modelo de Razonamiento Basado en Casos para la Captación de Requisitos en el desarrollo de proyectos de software. San Marcos : Universidad Nacional Mayor de San Marcos, 2013

PRESSMAN, ROGER S. Ingeniería de Software, un enfoque práctico. s.l. : McGraw-Hill Companies, 2002. ISBN: 8448132149

PRESSMAN, ROGER S. . Ingeniería de Software. Un enfoque práctico. México D.F : s.n., 2010. 978-607-15-0314-5

PRESSMAN, ROGER S. *Ingeniería de Software. Un enfoque práctico.* México D.F : The Mc Graw Hill, 2010. 978-0-07-337597-7

Quality. Quality, performance, security and process solutions. [en línea] 2016. [Fecha de consulta: 15 de noviembre de 2017]. Disponible en: http://vyvquality.com/pruebas-seguridad/

RAVI, SALVE. Tecmint, Linux Howto's Guide . *SARG – Squid Analysis Report Generator and Internet Bandwidth Monitoring Tool.* [en línea] 2016. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: https://www.tecmint.com/sarg-squid-analysis-report-generator-and-internet-bandwidth-monitoring-tool/

RIEHLE, DIRK. Framework Design: A Role Modeling Approach. s.l. : Swiss Federal Institute of Technology, 2000

RODRÍGUEZ , DARÍO y GARCÍA-MARTÍNEZ, RAMÓN . Elementos de Análisis y Diseño para Espacios Virtuales para la Formación de Investigadores. *Revista Latinoamericana de Ingeniería de Software.* [en línea] 2013. [Fecha de consulta: 1 de marzo de 2018.]. Disponible en: http://revistas.unla.edu.ar/software/article/view/95. 2314-2642

MATAM, SAI y otros. *Pro Apache JMeter Web Application Performance Testing.* BTH Blekinge Institute of Technology. [en línea] 2016. [Fecha de consulta: 20 de diciembre de 2017.]. Disponible en: http://bth.diva-portal.org/smash/record.jsf?pid=diva2%3A951918&dswid=-2161

SCOTTS, AMUEL. *Five reasons to upgrade to Kibana 4.* The Server Side.COM. [en línea] 2015. [Fecha de consulta: 16 de diciembre de 2017.]. Disponible en: http://www.theserverside.com/discussions/thread/80828.html

SÁNCHEZ, RODRIGUÉZ, TAMARA. Metodología de desarrollo para la Actividad productiva de la UCI v1.2. La Habana, Cuba : s.n., 2015

Sawmill. Sawmill Universal Log File Analysis and Reporting. [en línea] 2017. [Fecha de consulta: 11 de noviembre de 2017.]. Disponible en: http://www.sawmill.net/index.html

SCHMITT, CHRISTIAN y DUPMEIER, CLEMENS. *Storing and Analyzing Bibliographic Metadata with ElasticSearch.* Institut für Angewandte Informatik. [en línea] 2014. [Fecha de consulta: 8 de febrero de 2018.]. Disponible en: https://publikationen.bibliothek.kit.edu/1000041752

Segurmática 2017. Segurmática. *AAInternet Analizador de Acceso a Internet.* [en línea] 2017. [Fecha de consulta: 11 de noviembre de 2017.]. Disponible en: http://www.segurmatica.cu/aaInternetprod

SlideShare 2017. SlideShare. [en línea] 2017. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: https://es.slideshare.net/katival95/el-reporte-71892650

CHAVEZ, SONIA MARIA y otros. *Análise dos riscos e efeitos nocivos do uso da Internet.* Universidade Tecnológica Federal do Paraná. [en línea] 2015. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: http://repositorio.utfpr.edu.br/jspui/handle/1/1987

Squid-cache.org. Squid: Optimising Web Delivery. [en línea] 2013. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: http://www.squid-cache.org/

SUÁREZ, RAMÓN CARLOS. *Tecnologías de la Información y la Comunicación: Introducción a los sistemas de información y de telecomunicación.* s.l. : IdeasPropias, 2007

Symfony.es. [en línea] 2017. [Fecha de consulta: 21 de noviembre de 2017.]. Disponible en: http://symfony.es/pagina/que-es-symfony

WATARU, TAKASE, TOMOAKI, NAKAMURA. *A solution for secure use of Kibana and Elasticsearch in multi-user environment.* Cornell University Library. [en línea] 2017. [Fecha de consulta: 16 de diciembre de 2017.]. Disponible en: https://arxiv.org/abs/1706.10040

Techopedia. *Modeling Language.* [en línea] 2017. [Fecha de consulta: 20 de noviembre de 2017.]. Disponible en: https://www.techopedia.com/definition/20810/modeling-language

UNAD. Universidad Nacional Abierta y a Distancia. [en línea] 2018. [Fecha de consulta: 28 de febrero de 2018.]. Disponible en: http://stadium.unad.edu.co/ovas/10596_9836/diagrama_de_clases_de_diseo.html

Dragramas de Despliegue. Universidad Nacional Abierta y a Distancia. [en línea] 2018. [Fecha de consulta: 4 de marzo de 2018.]. Disponible en: http://stadium.unad.edu.co/ovas/10596_9836/diagramas_de_despliegue.html

Upcommons. Universitat Politécnica de Catalunya. [en línea] 2017. [Fecha de consulta: 10 de noviembre de 2017.]. Disponible en: http://upcommons.upc.edu/handle/2099/14430

VILLAMIZAR, KATERINE, TABARES, GARCÍA, JHON JAIRO y ZAPATA, JARAMILLO, CARLOS MARIO. *Mejora de historias de usuario y casos de prueba de metodologías ágiles.* Tecnológico de Antioquia. Institución universitaria. [en línea] 2016. [Fecha de consulta: 1 de marzo de 2018.]. Disponible en: http://ojs.tdea.edu.co/index.php/cuadernoactiva/article/view/246. 2027-8101