



Universidad de las Ciencias
Informáticas

Facultad 5.

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

Título:

Sistema para la identificación de violaciones de Seguridad Informática en el uso de mensajería instantánea, correo electrónico y navegación por Internet en la UCI.

Autor: Manuel Reina Aguilera.

Tutores: Msc. Ramón Alexander Anglada Martínez.

Lic. Luis Gabriel Viciado.

La Habana, Junio 2015

DECLARACIÓN DE AUTORÍA.

Declaro ser autor de la presente tesis y concedo a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año 2015.

Manuel Reina Aguilera

Msc. Ramón Alexander Anglada Martínez

Lic. Luis Gabriel Vicedo

AGRADECIMIENTOS.

Agradezco:

A la Revolución Cubana, que me ha permitido llegar hasta aquí hoy.

A Fidel, como guía y luz que me compromete a ser el relevo que se necesita.

A mi madre, como la best mother del mundo con todo el significado que tiene para mí.

A mi familia, mi papá, mis hermanos, abuelos, tíos, sobrinos, primos, la familia que conozco y la que no, la que está presente, y la que ya no está físicamente.

A las madres de mi vida, Mami, Rosi, mis profes Maria Caridad, Amalita, mis tías, las grandes madres de la UCI: Aleida, Mayra, Miriam, Yunelsy, Natalia, Andrea, Zaida y las profesoras.

A los padres de mi vida, Papi, mi tíos, Jorge Luis y los de la UCI: Villar, San Anglada.

A mis hermanos UCI, Hector, el noble, Jochi, el más Alpha, Reni, el inmortal. Todos: invencibles.

A mi amor, Gracias por existir, por estar ahí cuando lo necesito, eres única.

A mis tutores, Viciado por nunca abandonarme, y Anglada por darme la confianza final.

A los profes del tribunal, Villa, Yaser, Yadira, Valcarcel, Susej, Parra, Prevot, por sus consejos sabios.

A mis profesores, los que sin importar cuán grande o pequeña, dejaron una huella en mí.

A Seguridad Informática, por brindarme su apoyo, y consolidar lo que aprendí en el CDAE.

A mi Piquete, los infladores, los gavilanes, los fashion, los amigos en todo momento.

A los Alpha-Cun-Cun, Agrupación de personas fanáticas al agua, las fiestas y las piscinas.

A la FEU, que desde el primer día hasta el último me hizo dirigir y pasar muchos buenos momentos con la satisfacción del deber cumplido y de conocer gente maravillosa, nunca la olvidaré.

A la UCI, por superar mis expectativas y porque siga haciendo mucho en poco tiempo.

A las amistades, esas que nunca olvidaré a pesar del tiempo y la distancia que nos separe.

A la vida, ya que ni un tiro, ni ningún accidente, han impedido que esté aquí hoy, con 23 añitos.

A todos los que de una manera u otra han hecho posible este trabajo y me brindan su cariño y su presencia a diario, en esta tesis y en cada momento de mi vida: **GRACIAS!!!**

DEDICATORIA.

Aunque el tiempo pase, recordaré este momento especial, porque con este documento me hice Ingeniero en Ciencias Informáticas, un logro enorme en mi vida que se lo dedico a todas las personas que me han permitido llegar hasta aquí, y en especial:

A la Revolución Cubana.

A Fidel, Raúl, Ramón, Fernando, Antonio, René y Gerardo.

A mi madre y a todos mis familiares.

A los 23 años de estudio en mi vida.

A las personas que me salvaron la vida.

A los que confiaron en mí.

A ti que lees esta tesis hoy.

A todos...

PENSAMIENTO.

“En la Vida uno es capaz de lograr todo lo que se proponga.”

Frase del profesor Jorge Luis Rodríguez Gonzáles dos días antes de su muerte en el año 2007.

(Director de la Unidad 1 del IPVCE Vladimir Ilich Lenin.)

RESUMEN.

En la universidad existe demora cuando se acumulan denuncias de robo o uso indebido de credenciales de acceso por parte de los usuarios. Las herramientas actuales de análisis de trazas, no arrojan resultados que puedan resolver e identificar las posibles violaciones de Seguridad Informática relacionadas a credenciales de acceso en el uso de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet. El presente trabajo brinda la solución a este problema, obteniendo la información de acceso a dichos servicios desde una base de datos relacional y analizando los comportamientos anómalos de los usuarios, tales como las direcciones Ip no frecuentes, o condiciones de uso simultáneo de las cuentas. Con el uso del framework Symfony, se obtuvo como resultado una aplicación Web capaz de informar a los especialistas de Seguridad Informática sobre las posibles violaciones relacionadas con el robo y facilitación de credenciales de acceso en el uso de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet.

Palabras Claves: credenciales de acceso, Seguridad Informática, servicios telemáticos, Symfony2, Trazas.

INDICE GENERAL.

INTRODUCCION.....	9
CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA.....	14
1.1 Introducción.....	14
1.2 Principales conceptos asociados a la investigación.....	14
1.2.1 Seguridad Informática.....	14
1.2.2 Trazas.....	15
1.2.3 Credenciales de acceso.....	16
1.2.4 Servicios telemáticos.....	16
1.3 Herramientas para el análisis de trazas de los servicios de mensajería instantánea, correo electrónico y navegación por Internet.....	18
1.3.1 Sawmill.....	18
1.3.2 Analysis report generator (SARG).....	19
1.3.3 W3Perl.....	19
1.3.4 AiresProxy.....	19
1.3.5 Consideraciones del estudio.....	20
1.4 Tecnologías de desarrollo de software.....	20
1.4.1 Sistema gestor de bases de datos.....	21
1.4.2 Lenguajes de programación.....	21
1.4.3 Framework de desarrollo.....	23
1.4.4 Servidor Web.....	25
1.4.5 Metodología de desarrollo de software.....	25
1.4.6 Herramienta de prueba. JMeter.....	31
1.5 Conclusiones parciales.....	32
CAPÍTULO 2. DESCRIPCIÓN DE LA SOLUCIÓN.....	33
2.1 Introducción.....	33
2.2 Propuesta de solución.....	33
2.3 Especificación de los requisitos del sistema.....	34
2.3.1 Requisitos no funcionales.....	34
2.3.2 Involucrados en el sistema.....	37
2.3.3 Historias de usuario.....	37
2.4 Estimación y planificación.....	46

2.4.1 Estimación de esfuerzo por historias de usuarios	47
2.4.2 Plan de iteraciones.	47
2.4.3 Plan de entrega	50
2.5 Arquitectura de software.....	51
2.6 Patrones de diseño.	53
2.6.1 Patrones de asignación de responsabilidad (GRASP).....	53
2.7 Tarjetas Cargo o Clase, Responsabilidad y Colaboración (CRC).	54
2.8 Conclusiones parciales.....	58
CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA.	59
3.1 Introducción.....	59
3.2 Fase de implementación.....	59
3.2.1 Iteración 1.....	59
3.2.2 Iteración 2.....	61
3.2.3 Iteración 3.....	62
3.3 Pruebas.....	63
3.3.1 Plan de pruebas.	63
3.3.2 Pruebas de aceptación.	64
3.3.3 Pruebas de rendimiento.	67
3.4 Conclusiones parciales.....	69
CONCLUSIONES.....	70
RECOMENDACIONES.....	70
ANEXOS	71
BIBLIOGRAFÍA.....	71
REFERENCIAS BIBLIOGRÁFICAS.....	74

INTRODUCCION.

En la actualidad, las denominadas tecnologías de la información y las comunicaciones (TIC) ocupan un lugar central en la sociedad y en la economía de principios de siglo. Este elemento permite que surja una convergencia tecnológica entre la electrónica, la informática y las infraestructuras de telecomunicaciones. La asociación de estas tres tecnologías da lugar a una concepción del proceso de la información, en el que las comunicaciones abren nuevos horizontes y paradigmas en el desarrollo de la sociedad. (1).

Por consiguiente, el crecimiento que ha tenido el empleo de las redes informáticas y servicios telemáticos en los últimos años a nivel internacional y nacional, ha propiciado que sea más complejo el control del uso eficiente y seguro de los mismos, lo cual varía en dependencia del lugar y las condiciones que sean analizadas.

La Universidad de las Ciencias Informáticas (UCI) se ha convertido desde el punto de vista tecnológico, en un sitio de referencia en el uso de los servicios informáticos a nivel nacional. La misma, cuenta con una red de 32 megabits de ancho de banda, con la que interactúan alrededor de 6500 usuarios, soportado por un centro de datos que aloja los servidores y sistemas de almacenamiento donde se ejecutan las aplicaciones y se procesan los datos que permiten el funcionamiento la universidad.

Toda esta infraestructura de la UCI es supervisada por la Dirección de Seguridad Informática (DSI), que cumple como principal misión, garantizar la disponibilidad, confiabilidad e integridad de los servidores y los servicios telemáticos brindados a toda la comunidad universitaria, en apoyo a las actividades en la docencia, la investigación, la producción de software y la Extensión Universitaria. (2).

De igual forma, el análisis periódico de las bitácoras y registros de auditoría¹, así como la gestión de los incidentes de Seguridad Informática (3), constituyen elementos claves para apoyar la misión de la DSI.

En la actualidad esta dirección, diariamente, y de manera automática, almacena las trazas de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet, en una base de datos relacional con el objetivo de, a través de aplicaciones o

¹ Registra la aparición de cualquier evento auditado. El registro incluye información, como quién realizó la acción, qué archivos fueron afectados, qué acción se intentó realizar y dónde y cuándo ocurrió la acción que fue auditada.

herramientas informáticas, mostrar estos datos para ayudar en la toma de decisiones.

En estos momentos, las herramientas que utiliza la DSI para obtener información a partir de las trazas obtenidas, no ofrecen elementos relacionados a determinadas violaciones de Seguridad Informática que cometen los usuarios, específicamente las relacionadas a robo y facilitación de credenciales de acceso. Estas se encuentran reflejadas en las trazas a partir de determinadas características en el uso de los servicios, como por ejemplo la utilización de los servicios telemáticos desde direcciones Ip no frecuentes, o el uso de manera simultánea de un mismo usuario desde diferentes direcciones Ip en un mismo intervalo de tiempo.

Se aprecia, del lado de los usuarios, que en varias ocasiones facilitan sus credenciales de acceso o sufren el robo de las mismas, constituyendo así, una de las principales causas de sus denuncias. Los mismos, al ver violentada su privacidad y sus respectivas cuentas requieren los servicios de la DSI, para lo cual, en momentos de una cantidad elevada de denuncias, se produce una demora para dar respuesta a las violaciones cometidas.

Por tanto, la **situación problemática** es la siguiente:

- Existe demora en la respuesta cuando se realiza una denuncia de posible robo de credenciales de acceso por un usuario a la dirección de Seguridad Informática.
- Elevado esfuerzo y tiempo empleado por el especialista de Seguridad Informática para determinar posibles infractores en el uso de la mensajería instantánea, el correo electrónico y la navegación por Internet.
- No se ofrece información por ninguna herramienta de las empleadas por la DSI, relacionada a las violaciones que cometen los usuarios de los servicios, específicamente las violaciones relacionadas a robo y facilitación de credenciales de acceso, lo cual se refleja en las trazas a partir de determinadas características.
- No se generan reportes que contribuyan a informar a los especialistas de seguridad sobre posibles violaciones de este tipo.

Esta situación conlleva a plantear el siguiente **problema científico**: ¿Cómo identificar violaciones de Seguridad Informática relacionadas a las credenciales de acceso en el uso de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet para apoyar en la toma de decisiones de los especialistas de Seguridad Informática?

Luego, el **objeto de estudio** es: Trazas generadas por los servicios telemáticos mensajería instantánea, correo electrónico y navegación por Internet.

Y para dar solución al problema científico se propone como **objetivo general** de la investigación: Desarrollar una herramienta informática que muestre, toda la información relacionada a las posibles violaciones de Seguridad Informática de robo y facilitación de credenciales de acceso en el uso de la mensajería instantánea, el correo electrónico y la navegación por Internet, para mejorar la toma de decisiones de los especialistas de la DSI.

Todo lo anterior delimita como **campo de acción**: El almacén de trazas de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet de la UCI.

Para darle cumplimiento al objetivo, se proponen las siguientes **tareas de investigación**:

1. Elaboración del marco teórico de la investigación, identificando tendencias para la realización de sistemas de reportes de servicios telemáticos.
2. Diseño de la aplicación informática según el objetivo propuesto.
3. Implementación de la aplicación informática que integre las características necesarias para realizar una solución de este tipo.
4. Validación de la propuesta de solución desarrollada.

Para dar cumplimiento a los objetivos de la investigación se emplearán los siguientes métodos científicos.

Métodos teóricos:

- **Análisis y síntesis:** Se utiliza para el análisis de las herramientas, tecnologías, lenguajes y la metodología de desarrollo seleccionadas, identificando las características que las distinguen y se refleja la valoración del autor sobre las mismas.
- **Histórico - Lógico:** Permite, la descripción de los hechos explicándolos a partir de la lógica de su desarrollo. En la concepción de la investigación, permite conocer desde los orígenes de la detección de violaciones de Seguridad Informática en las trazas de los servicios telemáticos, hasta el instante actual. Analizando la evolución cronológica del objeto y campo de acción que se estudia, y cómo han crecido sus funcionalidades se evidencia el uso de este método en la investigación.

➤ **Modelación:** El modelo científico es un instrumento de la investigación, de carácter material o teórico, que permite reproducir el fenómeno que se está estudiando. Es una reproducción simplificada de la realidad. Está empleado en la representación del módulo a utilizar en la solución propuesta.

Métodos empíricos:

➤ **Búsqueda Documental:** Se realiza con el fin de realizar un estudio profundo de la bibliografía empleada en el estudio de las metodologías, herramientas y tecnologías para elaborar el marco teórico de la investigación.

➤ **Consulta de expertos:** Mediante el método Delphi fue empleada, validando la investigación mediante el cuestionamiento a los expertos a partir de preguntas sobre la aplicación.

Aportes prácticos esperados del trabajo:

Agilizar el proceso de atención a los incidentes de Seguridad Informática relacionados con el robo de credenciales de acceso a los servicios de mensajería instantánea, correo electrónico y navegación por Internet.

Estructura del documento:

El presente documento está estructurado en tres capítulos como se describe a continuación:

Capítulo 1: Fundamentación teórica.

Se realiza un estudio de algunas soluciones de software existentes relacionadas con sistemas de reportes del uso de servicios telemáticos y al finalizar se hace referencia a la metodología de desarrollo de software y herramientas utilizadas durante el desarrollo de la investigación.

Capítulo 2: Descripción de la solución.

Se hace alusión a las características del sistema, donde se plasman los requisitos definidos, se realiza la descripción de las historias de usuarios, se explica cómo está estructurada la arquitectura de la aplicación que se implementará, los patrones de diseño que se utilizarán y se exponen las características de la solución propuesta.

Capítulo 3: Implementación y pruebas.

En este tercer y último capítulo se definen las tareas de ingeniería y se implementa el

sistema. Incluye ejemplos de código de las diferentes clases que explican el funcionamiento de los componentes y se concluye realizando pruebas para comprobar si la aplicación cumple sus objetivos, mostrando los resultados obtenidos.

Además esta tesis posee conclusiones, recomendaciones, anexos, bibliografía y referencias bibliográficas.

CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción.

El proceso de desarrollo del software tiene en cuenta las tecnologías y herramientas empleadas para la creación de productos de alta calidad. Es por ello, que es requerido que se defina las herramientas que son necesarias y cuáles se adaptan adecuadamente a las necesidades de solución.

En este capítulo se abordarán los antecedentes de los sistemas de reportes del uso de los servicios telemáticos y se realizará un estudio acerca de los sistemas similares que existen en la actualidad, obteniendo las tendencias que rigen el desarrollo de estas aplicaciones.

1.2 Principales conceptos asociados a la investigación.

A continuación, se pretende explicar un grupo de conceptos fundamentales para alcanzar un correcto entendimiento y comprensión de la investigación.

1.2.1 Seguridad Informática.

Según Jesús Rodea, seguridad es un estado de cualquier sistema (informático o no) que nos indica que ese sistema está libre de peligro, daño o riesgo. Se entiende como peligro o daño todo aquello que pueda afectar su funcionamiento directo o los resultados que se obtienen del mismo. El concepto de seguridad en la informática es utópico porque no existe un sistema totalmente seguro. Para que un sistema se pueda definir como seguro debe tener estas cuatro características:

- **Integridad:** La información sólo puede ser modificada por quien está autorizado.
- **Confidencialidad:** La información sólo debe ser legible para los autorizados.
- **Disponibilidad:** Debe estar disponible cuando se necesita.
- **Irrefutabilidad:** (No-Rechazo o No Repudio) Que no se pueda negar la autoría.

Dependiendo de las fuentes de amenaza, la seguridad puede dividirse en seguridad lógica y seguridad física. En estos momentos la Seguridad Informática es un tema de dominio obligado por cualquier usuario de la Internet, para no permitir que su información sea robada. (4).

Similar definición, pero con otros argumentos proporciona George Beekman al referir que la seguridad es un tema muy importante para cualquier empresa, esté o no conectada a una red pública. No solamente es importante, sino que también puede llegar a ser compleja. (5).

Por último, es válido el criterio de Jeffrey I. Whitten, cuya definición hace hincapié en el hecho de garantizar que los recursos informáticos de una compañía estén disponibles para cumplir sus propósitos, es decir, que no estén dañados o alterados por circunstancias o factores externos, es una definición útil para conocer lo que implica el concepto de Seguridad Informática.

En términos generales, la seguridad puede entenderse como aquellas reglas técnicas y/o actividades destinadas a prevenir, proteger y resguardar lo que es considerado como susceptible de robo, pérdida o daño. (6).

Se puede apreciar que la información es el elemento principal a proteger, resguardar y recuperar dentro de los diferentes tipos de redes para garantizar la seguridad de las mismas.

1.2.2 Trazas.

Teniendo claridad en cuanto al significado de la Seguridad Informática, también es clave en el desarrollo de la investigación, conocer el concepto de trazas, con el cual se comprenderá la forma en que se almacena la información del uso de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet de la universidad.

Traza, o log, como se le conoce a estos archivos, es un registro oficial de eventos durante un período de tiempo en particular. Es usado para registrar datos o información sobre quién, qué, cuándo, dónde y por qué un evento ocurre para un dispositivo en particular o aplicación. A nivel mundial la mayoría de las trazas se almacenan en texto plano, es decir, entendibles por cualquier persona, lo que permite que cualquier otro dispositivo o aplicación pueda hacer uso de éstos. (7).

En el **Anexo 1** se mostrará un ejemplo de las trazas que se analizan, lo cual permite apreciar las propiedades y la información que muestra cada una de ellas.

De manera específica, las trazas de la universidad están compuestas por elementos comunes, que permiten la obtención y trabajo con la información que ellas generan. Los parámetros fundamentales que almacenan son: año, mes, tipo de log, y nombre del log.

Todos estos elementos demuestran con claridad a qué traza se refiere cada uno de los registros almacenados en los servidores de la universidad.

1.2.3 Credenciales de acceso.

Para acceder a los servicios, es clave la utilización de credenciales de acceso, garantizando la generación de trazas a partir de una identificación previa. Este concepto es importante para poder entender las violaciones que se cometan.

Una credencial de acceso, es aquel valor personal, que permite identificar al dueño de la misma, con un código, clave o mensaje encriptado único, es una forma de autenticación que utiliza información secreta para controlar el acceso hacia algún recurso. Debe mantenerse en secreto ante aquellos a quien no se les permite el acceso. A aquellos que desean acceder a la información se les solicita esta credencial; si conocen o no la conocen, se concede o se niega el acceso a la información según sea el caso. (8).

En la UCI, es utilizada como credenciales de acceso un usuario y una clave de acceso o contraseña, con las cuales, los usuarios acceden a todos los servicios de la universidad.

1.2.4 Servicios telemáticos.

En el mundo actual, donde la informática y las comunicaciones tienen un despegue impresionante y cada vez es mayor el flujo de datos e información en la red de redes, se hace imprescindible conocer los conceptos fundamentales de servicios telemáticos, para obtener una idea clara de los mismos.

Se define, como servicio telemático, a aquellos que permiten y facilitan el trabajo en el campo científico y tecnológico, englobando el estudio, diseño, gestión y aplicación de las redes y servicios de comunicaciones, para el transporte, almacenamiento y procesado de cualquier tipo de información (datos, voz, vídeo, etc.), incluyendo el análisis y diseño de tecnologías y sistemas de conmutación. (9).

El conocimiento de estos conceptos, permite conocer y clasificar correctamente los servicios informáticos de la universidad, de los cuales, se concentrará el estudio sobre los tres servicios que más utiliza la comunidad universitaria: la mensajería instantánea, el correo electrónico y la navegación por Internet.

➤ Servicio de mensajería instantánea.

El servicio mensajería instantánea está formado por una serie de herramientas que permiten disponer de una forma de comunicación alternativa a la mensajería vocal y el

correo electrónico. Normalmente todos los servicios de mensajería instantánea soportan una serie de características que los hacen interesantes no sólo en los entornos sociales sino en los entornos colaborativos de trabajo.

El servidor de mensajería instantánea posee un dominio y permite que cualquier usuario pueda conectarse al mismo mediante los distintos programas que existen para los protocolos de mensajería instantánea.

Los datos de acceso más comunes son: servidor, nombre de usuario y clave de acceso. Estos dos últimos parámetros coinciden con los empleados en el servidor de correo electrónico de la universidad.

La UCI cuenta con un servicio de mensajería instantánea, que permite tener intercambio en tiempo real con todos los usuarios registrados del sistema. Por las características del tipo de servicio, es incluso posible intercambiar con usuarios que estén registrados en otros servidores cubanos. (10)

➤ **Servicio de correo electrónico.**

El servicio de correo electrónico, constituye una revolución en el mundo tecnológico actual, pues cada día son más los mensajes de este tipo que se envían y reciben en el mundo, con el aumento de las tecnologías. El usuario de Internet puede enviar y recibir mensajes de cualquier otro usuario de la red y puede enviar mensajes a otros sistemas de correo que estén conectados a Internet.

El correo electrónico no sólo significa mensajes personales, sino cualquier cosa que se pueda almacenar en un archivo se puede enviar, como por ejemplo un programa fuente o una foto escaneada.

En la UCI, se utiliza el correo electrónico con el uso de credenciales de acceso, y a partir del cual se generan trazas al igual que en los demás servicios. (11).

➤ **Servicio de navegación por Internet.**

Los servicios de Internet en el mundo varían en dependencia de los proveedores de dichos servicios, así como de la naturaleza del hardware, el contenido disponible desde el sitio y otras variables. Los servidores de Internet varían su funcionamiento en dependencia de la carga de trabajo que estos presenten. (12).

Internet consiste en millares de redes interconectadas dirigidas por proveedores de servicios, compañías individuales, universidades, gobiernos y otras personas. Es una red de redes que necesita operar en el mundo como si se tratase de una sola cosa.

En el caso de la universidad, el servicio de Internet es controlado mediante un sistema de cuotas bien estructurado para aprovechar las capacidades tecnológicas de la institución, tomando en cuenta el ancho de banda de 32 megabits con el que se cuenta y la cantidad de usuarios que disfrutan de este servicio, con el principal objetivo de hacer un uso racional y controlado del canal de Internet.

El uso de Internet se gestiona además con el empleo de un proxy², que es capaz de autorizar las peticiones y enviarlas a los servidores reales, o denegarlas y devolverlas a quien las solicitó, en dependencia de la validez de las credenciales de acceso.

1.3 Herramientas para el análisis de trazas de los servicios de mensajería instantánea, correo electrónico y navegación por Internet.

Las herramientas de análisis de trazas, tienen sus antecedentes desde que surgieron los sistemas informáticos, y se han desarrollado a la par de las nuevas tecnologías, posibilitando un mayor impacto en cuanto a la información que los mismos recogen.

En la búsqueda de sistemas o herramientas para analizar las trazas y obtener información de las violaciones en el uso de los servicios de mensajería instantánea, correo e Internet en determinado lugar, se pudieron encontrar algunas herramientas, de las cuales se analizarán sus características fundamentales, como precedente a la propuesta de solución de esta investigación.

1.3.1 Sawmill.

En primer lugar, y como ejemplo más completo, se encuentra el Sawmill, que genera estadísticas a partir de ficheros log. (13).

Se encuentra especialmente diseñado para analizar logs de acceso a Internet, pero puede procesar otros tipos de logs. Se ejecuta como un programa CGI³ en un servidor Web, y publica un intuitivo interfaz gráfico de usuario, que puede ser usado desde cualquier navegador, para configurar y ejecutarlo, o para ver estadísticas de páginas.

Las estadísticas son jerárquicas, atractivas, y llenas de enlaces que facilitan la

² Programa que permite o niega el acceso a una aplicación determinada entre dos redes. (14).

³ Programa de Interfaz de entrada común, por sus siglas en inglés CGI: Common Gateway Interface.

navegación. El programa incluye una completa documentación y ofrece una gran cantidad de opciones, incluyendo una base de datos persistente, control sobre la apariencia de las páginas de estadísticas, y opciones de filtrado sobre los logs.

1.3.2 Analysis report generator (SARG).

SARG es una herramienta que permite conocer en detalle la actividad de todos los usuarios y/o equipos dentro de la red, registrada en los logs del proxy. Fue desarrollada con el lenguaje de programación C y se distribuye bajo la licencia GPLv2⁴.

Esta licencia asegura la libertad de distribuir copias de software libre, de que se reciba el código fuente o se pueda conseguir, de que se puede cambiar el software o usar fragmentos de él en nuevos programas libres (15).

Provee mucha información sobre las actividades de los usuarios de Squid: tiempo, bytes, sitios y otros.

1.3.3 W3Perl.

W3Perl es un analizador de log, generalmente usado sobre software libre, aunque se puede usar en cualquier sistema operativo. Diseñado especialmente para obtener toda la información de los archivos de registros personales o para aquellos profesionales que necesiten una herramienta con la que tener estos archivos para solucionar problemas con los ordenadores de sus clientes.

Este programa analiza todos los logs del servidor y hace todos los informes necesarios sobre el estado y uso del servidor permitiendo solucionar problemas para el mejor funcionamiento de este. (16).

Los scripts generados permiten tener todos estos informes en formato HTML y pdf, de manera que puedan ser guardados en el equipo como historial, impresos o enviados por correo electrónico, siendo muy cómodo el trabajo con los mismos.

1.3.4 AiresProxy.

AiresProxy es un analizador de ficheros de logs que brinda a los usuarios la información referente a sus registros de navegación en forma de reportes. Para ello utiliza un motor de segmentación de registros, otro de mantenimiento a la base de datos y por último un módulo de análisis y presentación de datos. El sistema posibilita actualizar la cuota de

⁴ Licencia Pública General de GNU. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios.

navegación de los usuarios en caso de que la institución establezca cuotas para acceder a redes externas. (17).

Posee un programa de mantenimiento auxiliar encargado de eliminar de la base de datos los registros que no son de interés del usuario o los administradores de red.

Es válido resaltar que este analizador fue desarrollado por la UCI y se encuentra desplegado en la institución, presentando problemas de rendimiento y arquitectura en la actualidad.

1.3.5 Consideraciones del estudio.

El estudio de los sistemas analizadores de log existentes en la actualidad ha permitido concluir que en el caso de Sawmill, al ser un software propietario, su uso se ve limitado por la posibilidad de adquirir sus licencias. En la totalidad de las herramientas se aprecia una elevada complejidad cuando se intenta adaptar sus características al entorno de la UCI.

Esta característica, unido a que no ofrecen información relacionada a posibles violaciones de Seguridad Informática en el uso de los servicios de mensajería instantánea, correo e Internet de la universidad, permiten optar por una nueva solución adaptada a la arquitectura y las nuevas tecnologías existentes en la DSI.

1.4 Tecnologías de desarrollo de software.

Se define como tecnología de desarrollo de software a un conjunto integrado de notaciones, herramientas y métodos, que permiten el desarrollo de un producto software en un contexto organizativo dado. Una tecnología de software es la base que posibilita el desarrollo de sistemas de software (18) capaces de brindar una solución adecuada a una situación problemática, desde el punto de vista técnico.

A continuación se abordará sobre el estudio, selección, y fundamentación de las tecnologías de desarrollo de software empleadas en la investigación, haciendo énfasis en el gestor de bases de datos empleado, los lenguajes de programación, frameworks de desarrollo, los servidores Web, así como la metodología de desarrollo de software adecuada y el empleo de diferentes herramientas para la validación y el correcto funcionamiento del software propuesto.

1.4.1 Sistema gestor de bases de datos.

Toda la información que se procese con los scripts que interactúan con los logs de los usuarios, debe ser procesada y guardada en una base de datos, la cual constituye el punto de partida para obtener las consultas y los métodos que informen sobre las violaciones a la Seguridad Informática, para ello se utiliza un gestor de base de datos, los cuales son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

Se compone de un lenguaje de definición de datos, de un lenguaje de manipulación de datos y de un lenguaje de consulta. Tienen como propósito manejar de manera clara, sencilla y ordenada un conjunto de información.

Debido a que en los servidores de la DSI, los datos se encuentran almacenados en PostgreSQL, se analizarán las particularidades de este sistema a utilizar.

➤ **PostgreSQL.**

Es un servidor de bases de datos relacional libre. Sus principales características son: llaves ajenas o llaves foráneas, disparadores, vistas, integridad transaccional, acceso concurrente multiversión (no se bloquean las tablas, ni siquiera las filas, cuando un proceso escribe), capacidad de albergar programas en el servidor en varios lenguajes. Herencia de tablas, tipos de datos y operaciones geométricas. (19).

Por sus funcionalidades y la utilización del mismo en la DSI, será el empleado en la realización de la aplicación.

1.4.2 Lenguajes de programación.

Un lenguaje de programación es una poderosa herramienta utilizada para controlar el comportamiento de una máquina, particularmente una computadora. Consiste en un conjunto de símbolos y reglas sintácticas y semánticas que definen su estructura y el significado de sus elementos y expresiones.

Un lenguaje de programación, según expertos es “una técnica estándar de comunicación que permite expresar las instrucciones que han de ser ejecutadas en una computadora” (20). Pueden usarse para crear programas que controlen el comportamiento físico y lógico de una máquina, para expresar algoritmos con precisión, o como modo de comunicación humana (21).

Para el desarrollo de aplicaciones Web, existen lenguajes como Python, Java y especialmente uno que nació con el propósito de facilitar el diseño de páginas Web de carácter dinámico y hacer a la par programas potentes: PHP Hypertext Pre-Processor.

A continuación se describirá las principales características de cada uno de ellos, con la consiguiente argumentación de la selección para la solución propuesta.

➤ **Python**

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible.

Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional.

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos (22).

➤ **Java**

Java es un lenguaje de programación de alto nivel desarrollado por Sun MicroSystem a principios de la década del 90. Su sintaxis es muy parecida a la de C y C++ ya que su desarrollo fue inspirado en los mismos, siempre buscando eliminar errores que suelen inducirse en lo que respecta a la manipulación de memoria y punteros (23).

Los programas en Java generalmente son compilados a un lenguaje intermedio llamado bytecode, que luego es interpretado por una máquina virtual (VM por sus siglas en inglés). Esta última sirve como una plataforma de abstracción entre el hardware y el lenguaje. Señalar que la compilación a código máquina también es posible, brindando mayor eficiencia en la ejecución del programa pero limita su característica multiplataforma (24).

➤ **PHP**

PHP es un lenguaje interpretado en el lado del servidor que se caracteriza por su potencia, versatilidad, robustez y modularidad. Los programas escritos en PHP son embebidos directamente en el código HTML y ejecutados por el servidor Web a través de un intérprete antes de transferir al cliente que lo ha solicitado un resultado en forma de código HTML puro. Al ser un lenguaje que sigue las corrientes open source, son totalmente accesibles de forma gratuita en la red. Es un lenguaje multiplataforma, que

puede funcionar sobre la mayoría de los servidores Web y brinda soporte a más de 20 tipos de base de datos (22).

El lenguaje PHP se encuentra instalado en más de 20 millones de sitios Web y en un millón de servidores. Es también el módulo Apache más popular entre las computadoras que utilizan Apache como servidor Web.

El gran parecido que posee PHP con los lenguajes más comunes de programación estructurada, como C y Perl, permiten a la mayoría de los programadores crear aplicaciones complejas con una curva de aprendizaje muy corta. También les permite involucrarse con aplicaciones de contenido dinámico sin tener que aprender todo un nuevo grupo de funciones.

➤ **Análisis de la selección del lenguaje de programación.**

A partir del análisis de diferentes lenguajes de programación, y teniendo en cuenta las bondades que brinda para el trabajo con aplicaciones Web, se decide escoger a PHP como lenguaje que servirá para la programación de la solución propuesta.

Su elección, basada en su utilización para aplicaciones Web, la interoperabilidad que permite, el rendimiento que se pone de manifiesto en la rapidez de los programas, y fundamentalmente la experiencia adquirida en la DSI con su empleo, determinan que el lenguaje PHP es el más adecuado para lograr la solución deseada.

1.4.3 Framework de desarrollo.

Un framework, en el idioma utilizado por los desarrolladores de software, “es una estructura de soporte definido, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado para ayudar a desarrollar y unir los diferentes componentes de un software” (25).

En resumen permite, simplificar el desarrollo de las aplicaciones mediante la automatización de muchas de las tareas comunes. Además, un framework proporciona estructura al código fuente, forzando al programador a crear código más legible y más fácil de mantener.

A continuación se expondrá sobre el framework de desarrollo Symfony, el cual es muy utilizado para la programación en PHP, y permite el trabajo con aplicaciones Web profesionales, siendo una herramienta útil para los desarrolladores.

➤ **Symfony.**

Symfony sigue la estela de Rails para simplificar al máximo el desarrollo de aplicaciones Web profesionales con PHP, utilizando las mejores prácticas y los patrones de diseño más importantes. Symfony incorpora muchas de las ideas del RAD (“desarrollo rápido de aplicaciones”) para conseguir que la programación de las aplicaciones sea lo más productiva y correcta posible.

Symfony es un completo framework diseñado para optimizar, gracias a sus características, el desarrollo de las aplicaciones Web. Para empezar, separa la lógica de negocio, la lógica de servidor y la presentación de la aplicación Web. Proporciona varias herramientas y clases encaminadas a reducir el tiempo de desarrollo de una aplicación Web compleja. Además, automatiza las tareas más comunes, permitiendo al desarrollador dedicarse por completo a los aspectos específicos de cada aplicación. El resultado de todas estas ventajas es que no se debe reinventar la rueda cada vez que se crea una nueva aplicación Web.

Está desarrollado completamente con PHP y ha sido probado con éxito en sitios como Yahoo! Answers, delicious, DailyMotion y muchos otros sitios Web de primer nivel. Symfony es compatible con la mayoría de gestores de bases de datos, como MySQL, PostgreSQL, Oracle y SQL Server de Microsoft. Se puede ejecutar tanto en plataformas *nix (Unix, Linux, etc.) como en plataformas Windows. (26)

Constituye un framework para construir aplicaciones Web con PHP. En otras palabras, Symfony es un enorme conjunto de herramientas y utilidades que simplifican el desarrollo de las aplicaciones Web.

Symfony emplea el tradicional patrón de diseño MVC (modelo-vista-controlador) para separar las distintas partes que forman una aplicación Web. El modelo representa la información con la que trabaja la aplicación y se encarga de acceder a los datos.

La vista transforma la información obtenida por el modelo en las páginas Web a las que acceden los usuarios. El controlador es el encargado de coordinar todos los demás elementos y transformar las peticiones del usuario en operaciones sobre el modelo y la vista.

De manera general, se decidió emplear como framework de desarrollo a Symfony, en su versión 2.6.9. Fundamentalmente avalado en la experiencia que se brinda del trabajo en

el Departamento de Seguridad Informática, los beneficios que plantea al programador el empleo del mismo y por las bondades que posee en la confección de aplicaciones Web.

1.4.4 Servidor Web.

Es un programa que gestiona cualquier aplicación en el lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas con el cliente generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente.

El servidor Web se ejecuta en un ordenador manteniéndose a la espera de peticiones por parte de un cliente (un navegador Web) y que responde a estas peticiones, mediante una página Web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error (27).

Luego de haber seleccionado a PHP como lenguaje de programación, y a Symfony como framework de desarrollo, se decide utilizar el servidor Web Apache para permitir el tratamiento de las peticiones de los clientes.

➤ **Apache.**

Apache es desarrollado, en un entorno abierto, participativo y publicado bajo la licencia Apache versión 2, por miembros de la Apache Software Foundation y voluntarios independientes. El proyecto tiene la intención de ser una colaboración de los mejores desarrolladores de su clase de todo el mundo.

Apache puede funcionar como servidor Web por sí mismo. Al principio de su desarrollo existió la percepción de que su utilización de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con mínimos requisitos de velocidad y gestión de transacciones. Actualmente ya no existe esa percepción y Apache es usado como servidor Web independiente en entornos con alto nivel de tráfico y alta disponibilidad (28).

Se decide utilizar Apache en su versión 7, pues el mismo posee una licencia completamente libre, facilitando la obtención de nuevas versiones y por satisfacer los requerimientos del equipo de desarrollo de la DSI.

1.4.5 Metodología de desarrollo de software.

Una metodología de desarrollo de software se define como: “un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software. La metodología indica cómo hay que obtener los distintos productos parciales y finales.” (29).

➤ **Selección del enfoque de la metodología. Modelo de Barry Boehm y Richard Turner.**

La existencia de varios modelos de desarrollo de software, agrupados primitivamente en dos categorías: ágiles o robustas, y la necesidad real de abordar el desarrollo de soluciones informáticas a partir de una selección adecuada de una de estas metodologías constituye un elemento primario a decidir cuando el equipo de desarrollo se enfrenta a un problema de un cliente. El modelo propuesto por Barry Boehm y Richard Turner trata de decidir qué tipo de metodología a usar para determinar este elemento de decisión.

El mismo propone un grafo de decisión a partir de cuantificar mediante escalas ajustadas 5 variables. Estas variables son:

Personal: Cuantifica para su evaluación la composición de la estructura del grupo de desarrollo del sistema en cuanto a capacidad, experiencia y posibilidades para enfrentar la tarea. En este caso se cuenta con un solo desarrollador con las posibilidades de enfrentar la tarea y con conocimiento de programación Web para implementar la solución.

Criticidad: Evalúa y cuantifica en porcentaje o en valor absoluto cuán crítico puede ser el fallo del sistema en cuanto a pérdidas de vidas humanas, valores materiales y recursos. El fallo del sistema puede conllevar a un mayor esfuerzo y tiempo por los especialistas de Seguridad Informática para dar solución a posibles violaciones en el uso de los servicios telemáticos estudiados. Además, del correcto funcionamiento dependerá la detección de los posibles infractores.

Tamaño: Esta evaluación está dirigida a cuantificar el tamaño del grupo de desarrollo del sistema y en consecuencia en qué medida se requiere de una mayor comunicación entre sus integrantes y la necesidad de documentar el papel y acciones de cada uno. El tamaño es pequeño, ya que el grupo de desarrollo lo compone un estudiante, que interactúa con los especialistas de Seguridad Informática para poder dar la solución requerida, cumpliendo con toda la documentación necesaria.

Dinamismo: Expresa la capacidad del equipo de desarrollo para enfrentar el porcentaje de modificaciones del catálogo de requisitos en unidades de tiempo (mes). Lo cual se considera elevado, ya que se requiere que las modificaciones se realicen de manera rápida y efectiva, en un corto período de tiempo.

Cultura: Cuantifica la capacidad del equipo de desarrollo para adaptarse a cambios críticos del ambiente de desarrollo. En la aplicación, se puede apreciar este indicador en

los cambios ocurridos respecto a la idea inicial, que han permitido una solución útil a la dirección de Seguridad Informática de la UCI.

La siguiente tabla, muestra, de manera general, cómo se comportan los factores ante metodologías ágiles y robustas.

Factor	Discriminadores ágiles	Discriminadores robustos
Tamaño	Dependencia y escalabilidad limitada por el porcentaje alto de conocimiento tácito. Apropiado para equipos y productos pequeños.	Escalabilidad y conocimiento explícito. Apropiado para productos y equipos grandes. Duro de mantener en pequeños proyectos.
Criticidad	La simplicidad en la documentación y el diseño dificulta los planes de prueba. No aconsejado para sistemas con niveles de criticidad altos. (IEEE 1012).	Rigor de requisitos y diseño adecuados para procesos de prueba, verificación y validación. Duros de gestionar en proyectos de escasa criticidad.
Dinamismo	Factorizar desde un diseño básico hasta el producto final es un método ideal para entornos dinámicos e innovadores, pero muy caro por el “re-trabajo” para entornos estables o conocidos.	En sistemas estables y conocidos, partir de requisitos completos y diseños detallados permite trazar y seguir un plan completo y “hacerlo bien a la primera”.
Personal	Los métodos de trabajo ágiles requieren una masa crítica de técnicos con niveles de experiencia medios-altos, capaces de comprender y adaptar los métodos y las técnicas empleadas.	Aunque es aconsejable contar con personas expertas en las fases de definición del proyecto, luego pueden ejecutarse con menor masa crítica de expertos.

Cultura	Más apropiado para culturas de “empoderamiento” responsabilidad y horquilla de decisión y libertad personal.	Más apropiado en culturas en las que las personas se sienten seguras con un marco de tareas y responsabilidades bien definido.
----------------	--	--

Tabla 1: Discriminadores ágiles y robustos.

La organización de estas variables en un grafo estrella donde la convergencia es hacia el desarrollo de metodologías de desarrollo ágiles y la divergencia en al menos una de las variables conlleva a la sugerencia de empleo de metodologías más pesadas.

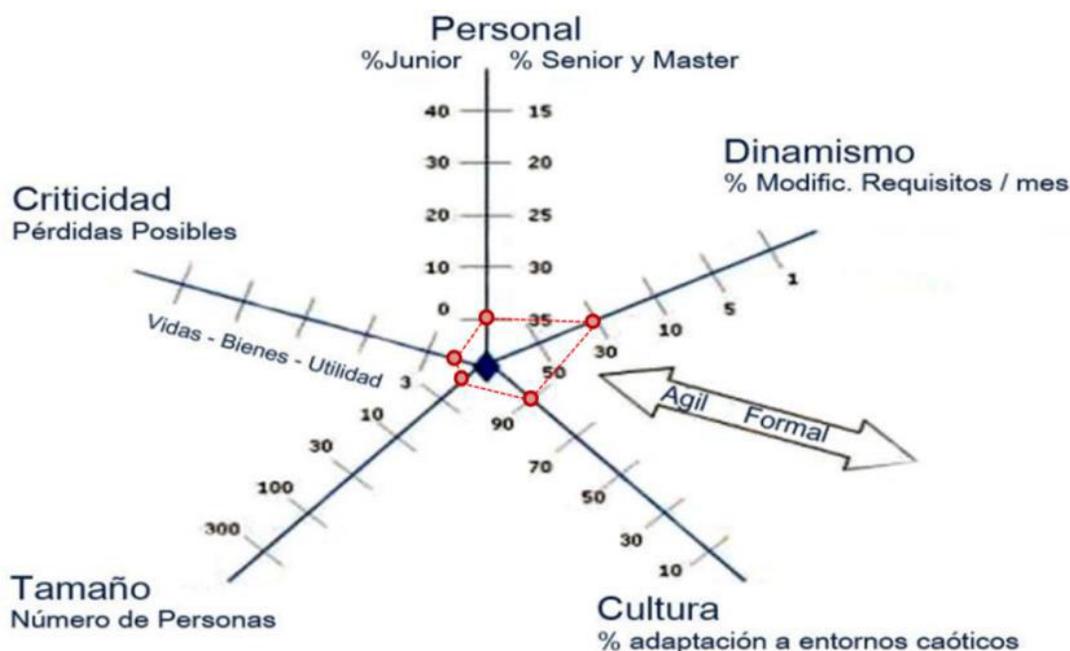


Figura 1: Modelo de Barry Boehm y Richard Turner con las características del SISTEMA-DSI⁵ (30).

Todos estos elementos apreciados con la ayuda de la estrella de la **Figura 1** y de la tabla comparativa, permiten decidir el enfoque de desarrollo a utilizar, lo cual ayudará a la selección de la metodología de desarrollo adecuada dentro del enfoque ágil seleccionado según los criterios apreciados.

⁵ Siglas que harán referencia en este documento al Sistema para la identificación de violaciones de Seguridad Informática en el uso de mensajería instantánea, correo electrónico y navegación por Internet en la UCI.

➤ **Selección de la metodología de desarrollo.**

Para la selección de la metodología de desarrollo a utilizar fueron consideradas dos de las más utilizadas, que funcionan en el desarrollo de software de una forma ágil. Estas son Scrum (S) y la Programación Extrema (XP).

➤ **SCRUM.**

Scrum es un proceso de desarrollo de software iterativo y creciente utilizado frecuentemente en entornos basados en el desarrollo ágil de software.

Aunque SCRUM estaba enfocado a la gestión de procesos de desarrollo de software, puede ser utilizado en equipos de mantenimiento de software, o en una aproximación de gestión de programas.

Está basado en un proceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de software. Está especialmente indicado para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en:

- El desarrollo de software se realiza mediante iteraciones, denominadas “sprints”, con una duración de 1 a 4 semanas. El resultado de cada “sprint” es un incremento ejecutable que se muestra al cliente.
- Reuniones a lo largo del proyecto. Éstas son las verdaderas protagonistas, especialmente la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración (31).

➤ **XP.**

La Programación Extrema es una metodología ágil concebida e implementada para dirigir las necesidades específicas del desarrollo de software conducido por equipos pequeños. Esta le da poder a los desarrolladores para responder con confianza a los requerimientos cambiantes del consumidor final.

Se caracteriza además por fomentar la comunicación desarrollador-cliente desde el primer día. Es considerada ligera, por la poca cantidad de documentación que la misma requiere, flexible por la forma en que se pueden hacer modificaciones al planteamiento inicial del problema de acuerdo a las nuevas necesidades que vayan apareciendo en la solución del problema, predecible, por la forma en que se van ejecutando sus etapas de manera escalonada y de bajo riesgo por la alta probabilidad de éxito en su utilización con el

consiguiente valor para procesos de desarrollo de corto tiempo. Sin embargo, pese a estas propiedades, no se considera para nada menos científica.

Otras ventajas que no pueden pasar por alto son los pocos requerimientos de documentación y planificación siendo las historias de usuarios los principales artefactos que se generan, así como la exigencia de tener siempre el cliente disponible para el desarrollo, implicando una mejor correspondencia entre el producto y la necesidad del negocio (31).

➤ **Análisis de la selección de la metodología de desarrollo.**

Teniendo en cuenta las características de las metodologías de desarrollo, se pudo encontrar un grupo de semejanzas y diferencias entre las mismas.

Como semejanza se aprecia que ambas metodologías de desarrollo se basan en los valores del manifiesto ágil, utilizan las historias de usuario, se realizan continuamente entregas al cliente en cortos períodos de tiempo, y poseen las reuniones repentinas entre los miembros del equipo.

Como diferencias se muestra que al finalizar un sprint, las tareas que se han realizado en las cuales se ha mostrado conformidad ya no se vuelven a tocar en ningún momento en el caso de SCRUM, mientras que en XP las tareas que se van terminando en las diferentes entregas al cliente son susceptibles a modificaciones durante el transcurso de todo el proyecto, incluso después de que funcione correctamente.

En SCRUM se puede modificar el orden de prioridad para el desarrollo de las tareas, mientras en XP se sigue el orden establecido por el cliente sin modificaciones.

Por último SCRUM es una metodología más basada en la administración del proyecto, mientras XP se centra más en la propia programación o creación del producto.

Después de analizadas sus principales similitudes y desventajas se determina seleccionar la metodología XP. Esto se fundamenta en la última diferencia mencionada, que permite centrar más el uso de la metodología hacia la creación del software, y no a la gestión o administración del proyecto.

La metodología seleccionada resalta por estar orientada a una entrega rápida de resultados, permitiendo la creación de un sistema funcional y que cumpla con los requisitos establecidos por el cliente.

1.4.6 Herramienta de prueba. JMeter.

Para el desarrollo de las pruebas de rendimiento a la aplicación Web, además de utilizar las ventajas que ofrece el framework Symfony para mostrar el rendimiento mediante su profiler⁶, existe el Apache JMeter, una herramienta de código abierto desarrollada en Java y diseñada para evaluar el rendimiento y comportamiento funcional de un sistema. Desarrollado por THE APACHE SOFTWARE FOUNDATION, la primera versión (v1.0) data de marzo del 2001.

Originalmente el Apache JMeter fue diseñado para realizar pruebas de estrés sobre aplicaciones Web (pruebas Web clásicas). Sin embargo hoy en día su arquitectura ha evolucionado, ahora no sólo puede llevar a cabo pruebas en componentes típicos de Internet (HTTP), sino también puede realizar pruebas sobre Bases de Datos, scripts Perl, servlets, objetos java, servidores FTP y prácticamente cualquier medio de los que se pueden encontrar en la red (32).

Esta herramienta incluye una interfaz gráfica de usuario que facilita el diseño de las pruebas. Esta interfaz gráfica además de aportar un entorno cómodo de trabajo, también permite guardar y alterar tanto los test desarrollados como los componentes que lo integran. Gracias a esto se pueden reutilizar las pruebas o módulos de las mismas en el desarrollo de nuevas pruebas. (33).

⁶ Symfony profiler: Brinda al desarrollador información actualizada por cada request sobre elementos útiles para desarrollador como son: parámetros del request, sentencias SQL ejecutadas, tiempos transcurridos en las respuestas, datos de sesión, entre otros.

1.5 Conclusiones parciales.

Se ha realizado un estudio de varias soluciones de software existentes relacionadas con sistemas de análisis de trazas, determinando que es muy compleja la incorporación de nuevas funcionalidades adaptables al entorno UCI. Además, el hecho de que Sawmill sea un software propietario, unido a que junto al resto de las herramientas no ofrecen información relacionada a posibles violaciones de Seguridad Informática en el uso de los servicios de mensajería instantánea, correo e Internet de la universidad, concluyen en la utilización de una nueva aplicación, capaz de ajustarse a las nuevas tecnologías y a la arquitectura actual de la DSI.

La investigación de las tecnologías de desarrollo de software existentes ha determinado el uso del lenguaje de programación PHP, del framework de desarrollo Web Symfony y el sistema gestor de base de datos PostgreSQL. Todo guiado por la metodología de desarrollo ágil XP.

CAPÍTULO 2. DESCRIPCIÓN DE LA SOLUCIÓN.

2.1 Introducción.

En este capítulo se abordarán los temas relacionados con la fase de exploración y planificación de la metodología de desarrollo XP, así como las características con que va a contar el sistema a implementar y su metáfora de arquitectura, además se tratarán los principales artefactos generados y la planificación del tiempo y el esfuerzo de las fases posteriores.

2.2 Propuesta de solución.

Una vez realizado el estudio del estado del arte se describe la solución propuesta. La misma, parte como premisa de que la DSI almacena las trazas en un servidor central que organiza la información por cada tipo de servicio. Estos registros son llevados luego de manera automática a una base de datos relacional para una mejor comprensión, y sobre este trabajo previo realizado en la DSI de la universidad, se comienza a implementar el sistema.

Partiendo del escenario actual antes descrito, se propone, una aplicación Web a la que podrán acceder los especialistas del departamento de Seguridad Informática de la UCI, que muestre las posibles violaciones de la Seguridad Informática en el uso de la mensajería instantánea, el correo electrónico y la navegación por Internet en la universidad. (**Figura 2**).

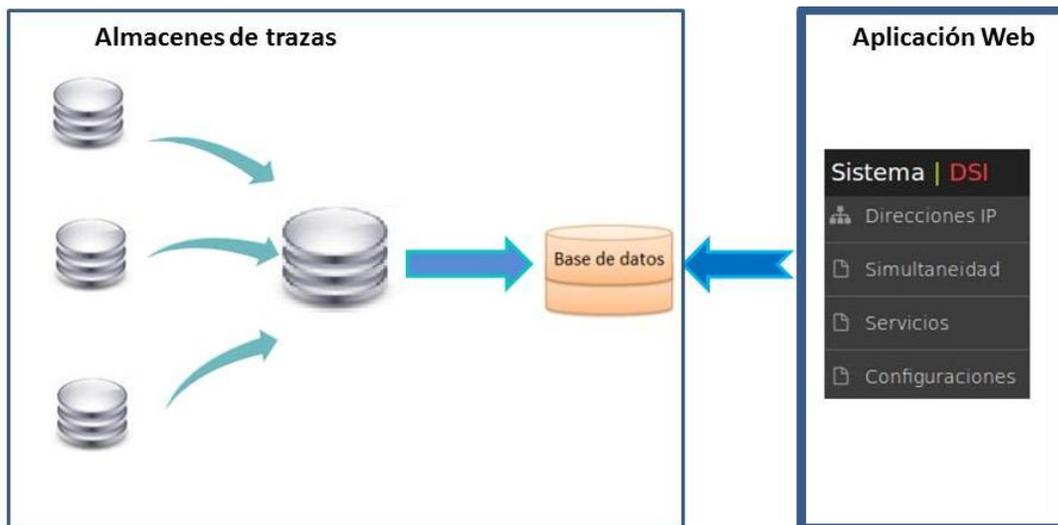


Figura 2: Propuesta de Solución.

Es importante resaltar como el sistema propuesto se va a centrar en diferentes situaciones claves como son:

- El acceso de los usuarios a los diferentes servicios.
- Identificación de las direcciones IP no comunes por las cuales se ha utilizado cualquiera de los servicios mencionados para todos los usuarios de la universidad.
- La ocurrencia de simultaneidad de uso de los servicios mencionados para todos los usuarios de la universidad.
- Notificar por correo electrónico de manera configurable la navegación por IP no comunes y la simultaneidad de uso de los servicios.
- Exportar a hoja de cálculo la información obtenida relacionada a las direcciones IP no comunes y la simultaneidad de acceso a los servicios.

Es importante acotar que de toda la información disponible, centrará la atención en la referente al uso de Internet, mensajería instantánea y correo de la universidad, para favorecer el análisis de los especialistas de Seguridad Informática.

2.3 Especificación de los requisitos del sistema.

Los requisitos de una aplicación son las cualidades que todo sistema debe poseer y determinan la arquitectura de cualquier software a desarrollar. (34). Para la realización de la aplicación propuesta se han identificado diferentes funcionalidades o requisitos según el empleo de la metodología de desarrollo de software seleccionada.

2.3.1 Requisitos no funcionales.

Se definen como requisitos no funcionales (RNF) a las cualidades o atributos globales del sistema que establecen restricciones sobre el producto desarrollado, el proceso de desarrollo o externas.

No están generalmente relacionados con la funcionalidad del sistema, y definen un grupo de características que el mismo debe cumplir.

➤ RNF#1 Apariencia o interfaz.

1. Poseer un banner con la información de los usuarios y el logo del sistema.
2. Tener un menú lateral con las opciones para interactuar en dependencia de los roles asignados.

3. Se utiliza una plantilla común para todas las interfaces con navegabilidad entre todas las páginas a visitar.
4. Las vistas tendrán 4 colores predominantes: gris oscuro, verde, gris claro y blanco, similares a los colores del departamento de Seguridad Informática.
5. El tipo de letra será “Helvética” con un tamaño de 12 píxeles. En los enunciados y el menú principal, se aumentará o resaltará para destacar este contenido.
6. Las páginas de la aplicación no tendrán imágenes, sólo se emplearán estas en el caso que el especialista desee mostrar alguna información en la página principal.
7. En las vistas, los resultados se obtendrán a través de tablas y formularios, los cuales deben brindar al usuario la información sobre los datos a introducir, facilitando la inserción de los mismos.
8. La página principal tendrá información que servirá de guía al usuario para trabajar con la aplicación mientras realice cualquier operación.
9. El logo estará ubicado en la parte superior izquierda del sitio, exponiendo su significado en la parte superior de la página.

➤ **RNF#2 Usabilidad**

1. La aplicación propuesta será usada por los especialistas de Seguridad Informática, quienes brindarán acceso a los usuarios de la universidad según los privilegios que consideren pertinentes.
2. El usuario tendrá acceso a un manual de uso que poseerá una descripción indicando de forma explícita las funcionalidades del sistema.
3. La interfaz facilita al usuario la interacción con el sistema, brindando la posibilidad de mensajes, para facilitar el trabajo con el sistema y para notificar e informar cualquier reporte del mismo.

➤ **RNF#3 Rendimiento**

1. Se mantendrá almacenados los datos de la aplicación en caché para que no sean solicitados de nuevo al servidor.
2. Las consultas a la base de datos se mantendrán en una caché global para poder ser utilizada por todos los usuarios y favorecer la velocidad del sistema.

3. La propuesta de solución debe ser rápida con un tiempo de respuesta que no deberá exceder de 200 milisegundos para las consultas comunes en la aplicación, ni los 2 segundos para consultas complejas.

➤ **RNF#4 Portabilidad**

1. La herramienta desarrollada sobre la Web debe ser multi-plataforma. Será utilizada preferentemente bajo cualquier distribución de Linux, y también puede ser visitado desde Windows o cualquier otro sistema operativo que se desee.
2. El servidor Web y el servidor de base de datos deberán conformar una arquitectura distribuida con los servidores en máquinas diferentes garantizado un mejor funcionamiento de los mismos, evitando sobrecargas de responsabilidades y facilitando el mantenimiento.

➤ **RNF#5 Seguridad**

1. Disponibilidad: La aplicación deberá estar disponible los 7 días de la semana las 24 horas del día.
2. Acceso a datos: El acceso o cualquier manipulación del sistema, debe estar sometido a un proceso de autenticación del usuario donde será especificado el usuario y la contraseña.
3. Validación del acceso a datos: Las contraseñas que servirán de credenciales de acceso al sistema serán codificadas para evitar violaciones de la seguridad, mediante el algoritmo sha512⁷.

➤ **RNF#6 Software.**

1. Requisitos mínimos para el cliente:
 - 1.1. Debe tener instalado un navegador Web para acceder a la aplicación.
 - 1.2. Se recomienda Linux como sistema operativo, aunque la aplicación es multiplataforma.
2. Requisitos mínimos para el servidor:
 - 2.1. Se requiere del gestor de base de datos PostgreSQL 9.4.

⁷ Algoritmo de encriptación de contraseñas utilizado por Symfony2 empleado en el sistema propuesto. Hace que las contraseñas se codifiquen 5.000 veces seguidas con el algoritmo sha512 utilizando en cada pasada el resultado de la anterior y codificando el resultado final (conocido como hash) con Base64. (35).

2.2. Servidor Web Apache.

2.3. PHP en su versión 5.

2.4. Se deben utilizar estas versiones o superiores para evitar incompatibilidades con versiones anteriores.

2.5. Se puede ejecutar sobre el sistema operativo Windows o Linux, aunque se recomienda cualquier versión de software libre.

➤ **RNF#7 Hardware.**

1. El servidor debe tener al menos 4GB de RAM.
2. El procesador debe tener una velocidad igual o mayor a 3.0 GHz.
3. Se debe poseer una tarjeta de red en el equipo.
4. Se requieren 40 GB mínimo de capacidad de disco duro.

2.3.2 Involucrados en el sistema.

Se definen como involucrados en el sistema todos aquellos que realizan una función o interactúan con él de una forma u otra.

Personas Relacionadas con el Sistema	Justificación
Usuario	Es la persona que se encarga de revisar los informes de los registros emitidos sobre su persona, puede revisar el resto, pero no modificar ninguna información de configuración.
Administrador	Es el encargado de revisar el correcto funcionamiento y velar por la seguridad del sistema. Gestiona el acceso y las acciones de los usuarios dentro del sistema y establece los parámetros de configuración.

Tabla 2: Involucrados en el sistema.

2.3.3 Historias de usuario.

Las Historias de usuario son la forma en que se especifican en XP las funcionalidades del sistema, realizándose una por cada característica principal del sistema; tienen el mismo

propósito que los casos de uso en las metodologías de desarrollo de software pesadas, aunque no son lo mismo ya que son escritas por los propios clientes desde su perspectiva del sistema, por lo que serán descripciones cortas y escritas en el lenguaje del usuario. (36).

Para definir las historias de usuario es utilizada la siguiente planilla, que contiene todos los datos necesarios para desarrollar la funcionalidad descrita.

Historia de Usuario	
Número: <i>Número de la HU, incremental en el tiempo.</i>	Nombre: El nombre de la HU, sirve para identificarla fácilmente entre los desarrolladores y los clientes.
Usuario: <i>El usuario del sistema que utiliza o protagoniza la historia.</i>	Iteración asignada: <i>La iteración a la que corresponde</i>
Prioridad en Negocio: <i>Qué tan importante es para el cliente.</i>	Puntos Estimados: <i>Qué tan difícil es para el desarrollador.</i>
Riesgo en desarrollo: <i>Qué tan importante es para la culminación del software.</i>	Puntos Reales: <i>Qué tan difícil fue en realidad para el desarrollador.</i>
Descripción: La descripción de la historia, detallando las operaciones del usuario y opcionalmente las respuestas del sistema.	
Observaciones: Algunas observaciones de interés, como glosario, información sobre usuarios, etc.	

Tabla 3: Planilla de las Historias de Usuario.

Cada historia de usuario fue asignada a una iteración, proceso el cual se discutirá con más detalle en la próxima sección. A continuación se muestran las **13** historias de usuario definidas para la elaboración del SISTEMA-DSI.

Historia de Usuario	
Número: 1	Nombre: Autenticar usuario.
Usuario: Todos	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 0,2
Riesgo en desarrollo: Bajo	Puntos Reales: 0,2
Descripción: El usuario debe ingresar su usuario y contraseña al sistema, si las credenciales de acceso coinciden, el usuario accederá a la información a la cual deba acceder en dependencia del rol que se le fue asignado, de lo contrario, se le mostrará un mensaje de error, pidiéndole que revise si expresó bien sus credenciales de acceso.	
Observaciones:	

Tabla 4: Historia de Usuario #1.

Historia de Usuario	
Número: 2	Nombre: Insertar usuario.
Usuario: Administrador	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 0,3
Riesgo en desarrollo: Bajo	Puntos Reales: 0,2
Descripción: El administrador del sistema debe ingresar los datos del usuario que desee agregar al sistema, una vez ingresado, el nombre, los apellido, la contraseña, y los datos del usuario, debe insertarlo en la base de datos, pasando a formar parte de la lista de usuarios del sistema que se mostrará al efectuarse un registro correcto y mostrando un mensaje de inserción exitosa. En caso de error, se le mostrará un mensaje de error, pidiéndole que revise si introdujo correctamente todos los datos solicitados.	
Observaciones:	

Tabla 5: Historia de Usuario #2.

Historia de Usuario	
Número: 3	Nombre: Actualizar usuario.
Usuario: Administrador	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 0,2
Riesgo en desarrollo: Bajo	Puntos Reales: 0,2
<p>Descripción: El administrador del sistema debe actualizar los datos del usuario que desee agregar al sistema, para ello ingresa el nombre, los apellido, la contraseña, y los datos del usuario. Además, podrá mostrar los datos una vez actualizados. Luego de la gestión se mostrará la lista del usuario modificado con un mensaje de actualización exitosa. En caso de error, se mostrará un mensaje, pidiéndole que revise si introdujo correctamente todos los datos solicitados en la actualización.</p>	
Observaciones:	

Tabla 6: Historia de Usuario #3.

Historia de Usuario	
Número: 4	Nombre: Eliminar usuario.
Usuario: Administrador	Iteración Asignada: 2
Prioridad en Negocio: Media	Puntos Estimados: 0,2
Riesgo en desarrollo: Bajo	Puntos Reales: 0,2
<p>Descripción: El administrador del sistema debe eliminar los datos del usuario que desee del sistema, para ello selecciona la opción editar usuario, y una vez dentro mostrado el usuario, procede a eliminarlo. Luego de la eliminación se mostrará la lista de los usuarios sin el usuario eliminado con un mensaje de eliminación exitosa. En caso de error, se mostrará un mensaje, pidiéndole que revise si puede proceder a eliminar el usuario.</p>	
Observaciones:	

Tabla 7: Historia de Usuario #4.

Historia de Usuario	
Número: 5	Nombre: Mostrar uso de servicios telemáticos.
Usuario: Todos	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 1
Riesgo en desarrollo: Medio	Puntos Reales: 0,8
<p>Descripción: El usuario podrá llenar un grupo de parámetros como el nombre del usuario, un rango de fecha, y el servicio específico que ha utilizado, y a partir de estos datos se mostrarán los registros de dichos usuarios. En caso de error, se mostrará un mensaje, pidiéndole que revise si introdujo correctamente todos los datos solicitados y en caso de no encontrarse información, se mostrará la tabla sin elementos indicando revisar nuevamente los parámetros que desea visualizar y un mensaje de texto solicitando que revise los parámetros de búsqueda.</p>	
Observaciones:	

Tabla 8: Historia de Usuario #5.

Historia de Usuario	
Número: 6	Nombre: Filtrar uso de Servicios.
Usuario: Todos	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 1,2
Riesgo en desarrollo: Medio	Puntos Reales: 1,4
<p>Descripción: El usuario podrá seleccionar y modificar el criterio de búsqueda en cuanto al uso de los servicios telemáticos, pudiendo filtrar todos los datos de los servicios a partir de si desea buscar por el nombre del usuario, un rango de fecha o el servicio específico utilizado. Se mostrarán los registros a partir del criterio de búsqueda utilizado. En caso de</p>	

error, se mostrará un mensaje, pidiéndole que revise si introdujo correctamente los datos solicitados y en caso de no encontrarse información, se mostrará la tabla sin elementos indicando revisar nuevamente los parámetros que desea visualizar y un mensaje de texto solicitando que revise los parámetros de búsqueda.

Observaciones:

Tabla 9: Historia de Usuario #6.

Historia de Usuario	
Número: 7	Nombre: Mostrar Ip Comunes
Usuario: Todos	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 1,6
Riesgo en desarrollo: Medio	Puntos Reales: 1,8
<p>Descripción: El usuario podrá seleccionar un nombre de usuario, un servicio y un rango de fecha en específico, y a partir de estos datos se podrán mostrar las direcciones Ip más frecuentes que visita el mismo, mostrando una cantidad definida en la configuración del sistema, y emitiendo la cantidad de conexiones establecidas por el usuario desde esas direcciones Ip más frecuentes. En caso de error, se mostrará un mensaje, pidiéndole que revise si introdujo correctamente los datos solicitados y en caso de no encontrarse información, se mostrará la tabla sin elementos indicando revisar nuevamente los parámetros que desea visualizar y un mensaje de texto solicitando que revise los parámetros de búsqueda.</p>	
<p>Observaciones: Por defecto se consideran Ip comunes aquellas 5 direcciones Ip a las cuales se ha conectado una mayor cantidad de veces el usuario.</p>	

Tabla 10: Historia de Usuario #7.

Historia de Usuario	
Número: 8	Nombre: Mostrar Ip No Comunes

Usuario: Todos	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 1,4
Riesgo en desarrollo: Medio	Puntos Reales: 1,4
<p>Descripción: El usuario podrá seleccionar un nombre de usuario, un servicio y un rango de fecha en específico, y a partir de estos datos se podrán mostrar las direcciones Ip menos frecuentes de las cuales se ha conectado el usuario, mostrando una cantidad definida en la configuración del sistema, y emitiendo la cantidad de conexiones establecidas por el usuario desde esas direcciones Ip poco frecuentes. En caso de error, se mostrará un mensaje, pidiéndole que revise si introdujo correctamente los datos solicitados y en caso de no encontrarse información, se mostrará la tabla sin elementos y un mensaje de texto solicitando que revise los parámetros de búsqueda.</p>	
<p>Observaciones: Por defecto se consideran Ip no comunes aquellas 5 direcciones Ip a las cuales se ha conectado una menor cantidad de veces el usuario.</p>	

Tabla 11: Historia de Usuario #8.

Historia de Usuario	
Número: 9	Nombre: Mostrar Simultaneidad de uso.
Usuario: Todos	Iteración Asignada: 1
Prioridad en Negocio: Alta	Puntos Estimados: 3
Riesgo en desarrollo: Alta	Puntos Reales: 3,2
<p>Descripción: El usuario podrá seleccionar un nombre de usuario, un servicio y un rango de fecha en específico, y a partir de estos datos se ordenarán por el usuario dado y desde la fecha de inicio, hasta la fecha final, los datos de todas las conexiones del usuario ordenadas ascendentemente por el tiempo de conexión, lo cual permitirá definir una conexión simultánea, cuando se detecte dos direcciones Ip consecutivas en un intervalo de tiempo menor al definido por parámetro. Se mostrará el usuario, la dirección Ip actual, las direcciones Ip con las cuales se determinó la simultaneidad, y el intervalo de tiempo de</p>	

la misma. En caso de error, se mostrará un mensaje, pidiéndole que revise si introdujo correctamente los datos solicitados y en caso de no encontrarse información, se mostrará la tabla sin elementos indicando revisar nuevamente los parámetros que desea visualizar y un mensaje de texto solicitando que revise los parámetros de búsqueda.

Observaciones: Por defecto se considera Conexión Simultánea a la conexión que se realiza con un intervalo de tiempo menor a 60 segundos, o 1 minuto desde diferentes direcciones Ip.

Tabla 12: Historia de Usuario #9.

Historia de Usuario	
Número: 10	Nombre: Configurar Ip Comunes
Usuario: Administrador	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0,4
Riesgo en desarrollo: Bajo	Puntos Reales: 0,6
Descripción: El administrador podrá seleccionar la cantidad de Ip a considerar como frecuente para cada usuario, cifra que va a determinar los datos que se mostrarán en la vista de Ip Comunes para cada usuario. En caso de no configurarse, la cantidad de direcciones por defecto será 5. Se mostrará un mensaje ante cualquier cambio en esta configuración.	
Observaciones:	

Tabla 13: Historia de Usuario #10.

Historia de Usuario	
Número: 11	Nombre: Configurar Simultaneidad
Usuario: Administrador	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 0,4

Riesgo en desarrollo: Bajo	Puntos Reales: 0,4
Descripción: El administrador podrá seleccionar la cantidad de tiempo que se tendrá en cuenta para considerar que existen conexiones simultáneas entre dos direcciones Ip distintas. Por defecto el tiempo a considerar será de 1 minuto y en caso de no configurarse, esta será la configuración. Se mostrará un mensaje ante cualquier cambio en esta configuración.	
Observaciones:	

Tabla 14: Historia de Usuario #11.

Historia de Usuario	
Número: 12	Nombre: Enviar Correo de Notificación
Usuario: Todos	Iteración Asignada: 3
Prioridad en Negocio: Baja	Puntos Estimados: 1
Riesgo en desarrollo: Bajo	Puntos Reales: 1
Descripción: El usuario podrá enviar un correo de notificación, en donde pueda informar sobre los reportes generados en cuanto al uso de los servicios telemáticos. Se pedirá el usuario de correo, la contraseña y los datos de la persona que desea enviar la notificación, además del correo al cual se desea enviar la notificación y se enviará la misma. Mostrando un mensaje de confirmación en caso positivo y uno de error si no se pudo enviar el correo.	
Observaciones:	

Tabla 15: Historia de Usuario #12.

Historia de Usuario	
Número: 13	Nombre: Generar excel de informe
Usuario: Todos	Iteración Asignada: 3

Prioridad en Negocio: Baja	Puntos Estimados: 0,8
Riesgo en desarrollo: Bajo	Puntos Reales: 1
Descripción: El usuario podrá generar un documento excel de informe, en donde pueda registrar todos los valores sobre los reportes generados en cuanto al uso de los servicios telemáticos. Se dará la opción de abrir el archivo, guardarlo, o modificarlo según las necesidades del usuario y se permitirá filtrar los datos del excel generado para un mejor análisis.	
Observaciones:	

Tabla 16: Historia de Usuario #13.

2.4 Estimación y planificación.

Durante la planificación del proyecto el cliente establece la prioridad de cada historia de usuario, y correspondientemente, los programadores realizan una estimación del esfuerzo necesario de cada una de ellas. Se toman acuerdos sobre el contenido de la primera entrega y se determina un cronograma en conjunto con el cliente. Una entrega debería obtenerse en no más de tres meses. Esta estimación dura unos 3 días.

Las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. Las historias generalmente valen de 1 a 3 puntos.

Por otra parte, el equipo de desarrollo mantiene un registro de la “velocidad” de desarrollo, establecida en puntos por iteración, basándose principalmente en la suma de puntos correspondientes a las historias de usuario que fueron terminadas en la última iteración.

La planificación se puede realizar basándose en el tiempo o el alcance. La velocidad del proyecto es utilizada para establecer cuántas historias se pueden implementar antes de una fecha determinada o cuánto tiempo tomará implementar un conjunto de historias. Según alcance del sistema, se divide la suma de puntos de las historias de usuario seleccionadas entre la velocidad del proyecto, obteniendo el número de iteraciones necesarias para su implementación (37).

2.4.1 Estimación de esfuerzo por historias de usuarios

Para el desarrollo de la aplicación propuesta se realizó una estimación de esfuerzo por cada una de las historias de usuario identificadas, los resultados se muestran a continuación:

Historia de Usuario	Puntos de estimación (semanas)
Autenticar usuario.	0.2
Insertar usuario.	0.3
Actualizar usuario.	0.2
Eliminar usuario.	0.2
Mostrar uso de servicios telemáticos.	1
Filtrar uso de Servicios.	1.2
Mostrar Ip Comunes.	1.6
Mostrar Ip No Comunes.	1.4
Mostrar Simultaneidad de uso.	3
Configurar Ip Comunes.	0.4
Configurar Simultaneidad.	0.4
Enviar Correo de Notificación.	1
Generar excel de informe.	0.8

Tabla 17: Estimación del esfuerzo por HU.

2.4.2 Plan de iteraciones.

Después de ser identificadas las historias de usuario y estimado el esfuerzo dedicado a la realización de cada una de estas historias de usuario se procede a la realización de la planificación de la etapa de implementación de la solución. Para un mejor desempeño a la hora del desarrollo se ha establecido una división de la implementación en 3 iteraciones:

➤ **Iteración 1:**

La iteración 1 será la encargada de la implementación de las historias de usuario con mayor prioridad en el sistema. Al finalizar esta iteración se contará con la primera versión de prueba del SISTEMA-DSI para el desarrollo, ejercitación y realización de monitoreos simples. Como resultado de esta, el sistema se pondrá en producción para evaluar su desempeño.

En esta iteración se implementarán las historias de usuario que por la importancia que tienen para el cliente, tienen prioridad con respecto a las otras. Al finalizar esta iteración se contará con las funcionalidades descritas en las historias de usuario 5, 6, 7, 8 y 9 referentes a las funcionalidades fundamentales del sistema, sobre todo, la detección de los Ip no comunes de determinados usuarios, el reporte de la simultaneidad de uso en las redes de la universidad y el listado de los servicios telemáticos que se han utilizado. Esta versión del sistema tiene como objetivo mostrarle al cliente cómo va quedando la aplicación, para comprobar el grado de aceptación que tiene el producto.

➤ **Iteración 2:**

La iteración 2 tendrá como principal objetivo la implementación de las HU con prioridad media.

El objetivo de esta iteración es la implementación de las funcionalidades de prioridad alta, media y baja que no fueron tratadas en la primera iteración. Al término de esta, se tendrán implementadas las funcionalidades reflejadas en las historias de usuario 1, 2, 3 y 4 referentes a la autenticación, y gestión de los usuarios del sistema. Al igual que en la primera iteración, al término de esta se contará con una versión de prueba del producto, versión número 2 que tendrá un objetivo similar a la anterior iteración.

➤ **Iteración 3:**

En esta iteración se implementarán las funcionalidades restantes, con la culminación de la misma estarán desarrolladas las peticiones del cliente descritas en las historias de usuario 10, 11, 12 y 13, referentes a determinar los parámetros de configuración de simultaneidad e Ip no comunes, así como la posibilidad de generar un excel con el informe de los reportes visualizados en la aplicación, y la notificación por correo de los mismos.

La iteración 3 tendrá como tarea la implementación de las HU restantes, que son las encargadas de complementar el resultado de las iteraciones anteriores. Al terminar esta,

se contará con una versión 2.0 del producto final que sustituirá al primero para probar su funcionamiento.

Plan de duración de las iteraciones:

Iteraciones	Orden de las Historias de usuario a implementar	Cantidad de tiempo de trabajo
1	Mostrar uso de servicios telemáticos.	8.2 semanas
	Filtrar uso de Servicios.	
	Mostrar Ip Comunes.	
	Mostrar Ip No Comunes.	
	Mostrar Simultaneidad de uso.	
2	Autenticar usuario	0.9 semanas
	Insertar usuario.	
	Actualizar usuario.	
	Eliminar usuario.	
3	Configurar Ip Comunes.	2.6 semanas
	Configurar Simultaneidad.	
	Enviar Correo de Notificación.	
	Generar excel de informe.	
Total de Semanas		11.7

Tabla 18: Plan de Duración de las Estimaciones.

2.4.3 Plan de entrega

El plan de entrega se estructuró en torno a 3 hitos fundamentales, donde cada liberación se correspondió con una iteración de desarrollo:

Entregables	Entregas		
	20/05/2014	01/06/2014	15/06/2014
SISTEMA-DSI	v1.0 beta	v1.1 beta	v2.0 final

Tabla 19: Plan de entregas de la aplicación.

Cada entrega ocurre normalmente luego de varias iteraciones, en el transcurso de las cuales se cumplen las historias de usuario correspondientes a cada entrega. A su vez, cada historia de usuario se compone de varias tareas, además de existir tareas que no soportan ninguna historia en particular, sino que tienen como objetivo un mejor funcionamiento de la aplicación como un todo.

Especialmente la primera iteración es atípica según lo que dicta la metodología XP en lo referente a la duración de las iteraciones. En la planificación, la iteración 2 y la 3 constan de alrededor de 3 semanas para su desarrollo, precedidas de una semana de exploración para cada una. Sin embargo la primera iteración se prolongó por más de 2 meses de forma deliberada, hasta que todas las historias de usuario que conformasen la aplicación con las funcionalidades necesarias estuviesen implementadas.

Se tomó esta decisión dado que no tenía sentido liberar el software que no estaba completo. Sin embargo cada historia de usuario fue probada a medida que se terminaba, lo cual mitiga el riesgo corrido al retardar la liberación. Una vez lograda la primera liberación (20/05/2014) se hizo con la absoluta certeza de que todos los componentes desarrollados funcionaban correctamente, gracias a la estrategia de pruebas constantes.

La planificación de las entregas también se realizó atendiendo a un criterio particular: la completitud de la versión. Debido a las condiciones particulares de desarrollo de la aplicación, se hizo necesario definir estados en los cuales la aplicación pudiera congelarse un tiempo indefinido de ser necesario. Cada uno de estos estados o versiones puede mantenerse en producción de forma autosuficiente por un intervalo de tiempo ilimitado, lo cual sirve de mecanismo de contingencia por si es necesario detener el desarrollo durante semanas o incluso meses. Estos estados son las entregas.

2.5 Arquitectura de software.

En el desarrollo de la aplicación se utilizó el patrón MVC como modelo para la arquitectura del software. Esta selección se encuentra basada en la utilización de este patrón de diseño en múltiples aplicaciones Web y en el empleo del framework de desarrollo Symfony, que tiene esta arquitectura dentro de sus características. El patrón MVC contiene como capas lógicas principales: Capa de Presentación o Vistas, Capa de Control o Controlador y Capa de Datos o Modelo como se puede apreciar en la **Figura 3**, las mismas interactúan con el usuario y con una base de datos relacional indistintamente.

MVC es aplicable al desarrollo de cualquier aplicación independientemente del lenguaje de programación elegido (38). Es muy útil en la implementación de una aplicación Web, permitiendo la separación de los datos, la interfaz de usuario y la lógica de control en tres componentes distintos:

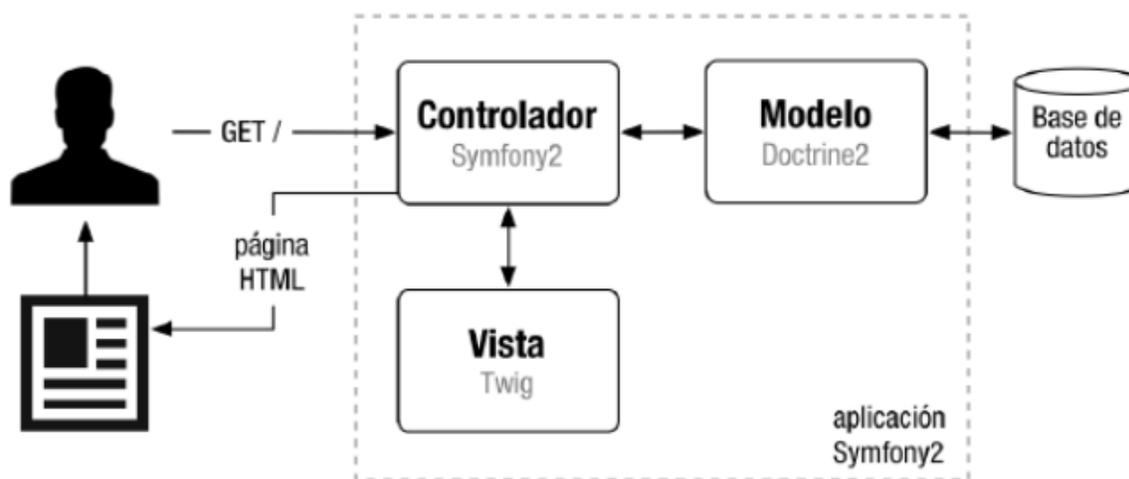


Figura 3: Arquitectura propuesta.

La capa de Datos o Modelo (Figura 4) contiene los componentes que representan y gestionan los datos manejados por la aplicación. En el caso más típico, los objetos encargados de leer y escribir en la base de datos. En la aplicación la capa de datos la compone la base de datos relacional que agrupa todos los elementos de los usuarios, servicios, direcciones Ip, fechas, tiempo, conexiones y datos agrupados todos en un único modelo de datos, el cual es accedido desde las clases entidades de la aplicación, que simulan las entidades que se aprecian en la base de datos, usando las ventajas del doctrine que provee Symfony2.

La capa Presentación o Vistas (Figura 4) es la responsable de mostrar al usuario el

estado del sistema y las acciones que tiene a su disposición. El estado del sistema pueden ser páginas HTML, documentos en formato PDF, hojas de cálculo, entre otras.

Al tener la capa vista separada de la capa controladora se pueden realizar cambios en esta sin tener que tocar nada más que una parte delimitada de código. Se puede apreciar con el uso del framework de desarrollo, una abstracción en la parte visual, que permitirá mostrar los datos de una base de datos, sin tener que entrar en el código de la misma, pero mostrando una interfaz cumpliendo los parámetros de diseño establecidos.

Las vistas se mostrarán a través del uso de las clases twig que proporciona Symfony que permite la visualización de toda la interfaz programada en la aplicación

La capa de Control o Controladores (Figura 4) es la encargada de la unión de las capas vista y modelo, esta capa es la que escucha los cambios en la vista y se los envía al modelo, el cual le regresa los datos a la vista. El framework utilizado propone una abstracción entre las capas modelo y vista, siendo la capa controladora la encargada de establecer la comunicación entre ellas, todo implementado con la ayuda de Symfony, que es capaz a la vez de controlar, procesar y almacenar toda la información recopilada en la base de datos con la información de las trazas de Internet de la universidad, con el apoyo de las clases controladoras que reciben la información de los formularios y procesa la misma interactuando con las entidades de la base de datos, para posteriormente enviar el resultado a la vista o interfaz de la aplicación.

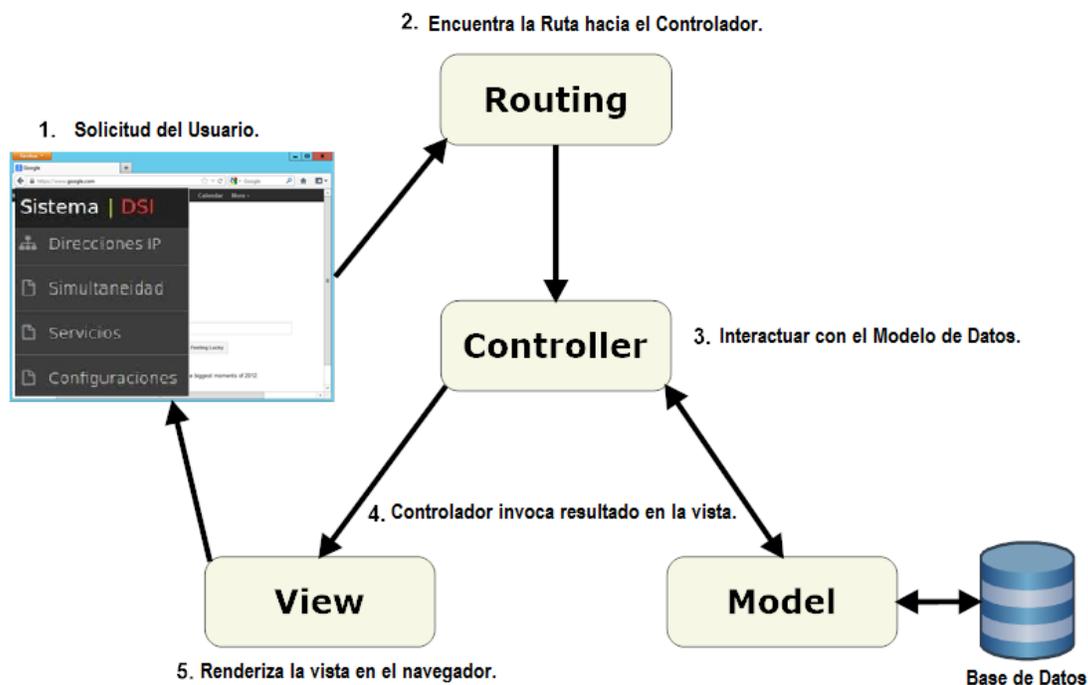


Figura 4: Modelo Vista Controlador en la aplicación.

2.6 Patrones de diseño.

Un patrón es un par problema/solución con nombre que se puede aplicar en nuevos contextos, con consejos acerca de cómo aplicarlo en nuevas situaciones y discusiones sobre sus compromisos. Un patrón es una descripción de un problema bien conocido que suele incluir: descripción, escenario de uso, solución concreta, las consecuencias de utilizar este patrón, ejemplos de implementación y la lista de patrones relacionados (40).

Los patrones de diseño ayuda a como se debe estructurar las clases y objetos para guiar todo el procesos de creación de software. Componen soluciones concretas a problemas que se presentan durante el diseño de una aplicación.

Entre los patrones de diseño más utilizados se encuentran los patrones GRASP, los cuales serán argumentados a continuación.

2.6.1 Patrones de asignación de responsabilidad (GRASP)

➤ Alta cohesión:

La cohesión es una medida de cuan relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme.

Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo. No conviene este tipo de clases pues presentan los siguientes problemas:

- Son difíciles de comprender, reutilizar y conservar.
- Las clases con baja cohesión a menudo representan un alto grado de abstracción o han asumido responsabilidades que deberían haber delegado a otros objetos (39).

En la aplicación, este patrón se evidencia, en la clase que implementa los métodos Actions, la cual está formada por varias funcionalidades que están estrechamente relacionadas, siendo la misma la responsable de definir las acciones para las plantillas y colaborar con otras para realizar diferentes operaciones, instanciar objetos y acceder a las propiedades, lo cual es muy necesario en las funcionalidades para listar servicios, mostrar lp comunes y simultaneidad de uso fundamentalmente.

➤ Bajo acoplamiento:

Asigna una responsabilidad para mantener bajo acoplamiento. El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y con que recurre a ellas. Una clase con bajo (o débil) acoplamiento no depende de muchas otras (39).

A lo largo del proceso de desarrollo se ha puesto en práctica dicho patrón al determinar que la clase `loguserservice` hereda únicamente de `loguserserviceaction` para alcanzar un bajo acoplamiento de clases. Las clases que implementan la lógica del negocio y de acceso a datos se encuentran en el modelo, las cuales no tienen asociaciones con las de la vista o el controlador, lo que proporciona que la dependencia en este caso sea baja.

➤ **Experto:**

Asigna una responsabilidad al experto en información: la clase que cuenta con la información necesaria para cumplir la responsabilidad (41).

El patrón experto es uno de los que más se utiliza cuando se trabaja con Symfony. Se evidencia claramente con el uso de la librería Propel para mapear la Base de Datos. Esta librería se utiliza para realizar su capa de abstracción en el modelo, para encapsular toda la lógica de los datos y generar las clases con todas las funcionalidades comunes de las entidades, las clases de abstracción de datos (Clases Peer del Modelo) poseen un grupo de funcionalidades que están relacionadas directamente con la entidad que representan y contienen la información necesaria de la tabla que representan.

➤ **Creador:**

Este patrón plantea asignarle a la clase B la responsabilidad de crear una instancia de la clase A (41). Dicho patrón se evidencia en la clase `Actions`, donde se encuentran las acciones definidas para el sistema. En dichas acciones se crean los objetos de las clases que representan las entidades, lo que evidencia que la clase `Actions` es “creador” de dichas entidades. Ejemplos de algunas funciones utilizadas en la clase `Actions` son: `filtradodeservicio()`, `ipcomunes()`, `simultaneidad()`.

2.7 Tarjetas Cargo o Clase, Responsabilidad y Colaboración (CRC).

Para guiar el diseño, la metodología XP usa las tarjetas Cargo o Clase, Responsabilidad y Colaboración (CRC). Esta técnica se usa para guiar el sistema a través del análisis donde las clases se examinan, se filtran y se refinan en base a sus responsabilidades con respecto al sistema. (42).

El nombre de la clase se coloca a modo de título en la tarjeta, las responsabilidades se colocan a la izquierda, y las clases que se implican en cada responsabilidad a la derecha, en la misma línea que su requerimiento correspondiente. Las tarjetas determinan el comportamiento de cada actividad. A continuación se muestran **8** de las tarjetas CRC definidas en el desarrollo de la solución:

➤ **Plantilla de la Tarjeta CRC.**

Tarjeta CRC	
Nombre de la clase:	
Responsabilidad:	Colaboración:

Tabla 20: Plantilla de Tarjeta CRC.

Tarjeta CRC 1	
Nombre de la clase: LogUserServiceController	
Responsabilidad	Colaboración
getId getUsername getClientAddress getEventDate getEventTime getServiceId	LogUserServiceRepository, Usuario, Service.

Tabla 21: Tarjeta CRC # 1.

Tarjeta CRC 2	
Nombre de la clase: listarUserService	
Responsabilidad	Colaboración

filtradoServicio servicio dadoUsuarioLog	LogUserServiceController, LogUserServiceRepository, DefaultController.
--	--

Tabla 22: Tarjeta CRC # 2.

Tarjeta CRC 3	
Nombre de la clase: ipComunes	
Responsabilidad	Colaboración
filtradoServicio IpComunes IpNoComunes	LogUserServiceController, LogUserServiceRepository, ipComunType.

Tabla 23: Tarjeta CRC # 3.

Tarjeta CRC 4	
Nombre de la clase: Simultaneidad	
Responsabilidad	Colaboración
filtradoServicio IpComunes LogUserService	LogUserServiceController, LogUserServiceRepository, simultaneidadType

Tabla 24: Tarjeta CRC # 4.

Tarjeta CRC 5
Nombre de la clase: generarReporte

Responsabilidad	Colaboración
filtradoServicio IpComunes Simultaneidad	LogUserServiceController, FakeController, LiuggioExcelBundle.

Tabla 25: Tarjeta CRC # 5.

Tarjeta CRC 6	
Nombre de la clase: notificarCorreo	
Responsabilidad	Colaboración
generarReporte filtradoServicio IpComunes Simultaneidad	LogUserServiceController, FakeController, MailerInterface.

Tabla 26: Tarjeta CRC # 6.

Tarjeta CRC 7	
Nombre de la clase: Usuario	
Responsabilidad	Colaboración
new index show edit	LogUserServiceController, Login, Logout.

Tabla 27: Tarjeta CRC # 7.

Tarjeta CRC 8

Nombre de la clase: DefaultController	
Responsabilidad	Colaboración
LogUserServiceController, UsuarioController	Service, Usuario, UsuarioRepository.

Tabla 28: Tarjeta CRC # 8.

2.8 Conclusiones parciales.

En el presente capítulo se definió la arquitectura de la aplicación a desarrollar y fueron identificadas las principales funcionalidades a desarrollar, las cuales quedaron plasmadas en las historias de usuario. Para una mejor organización del software se realizó el plan de iteraciones por el cual se regirá el proceso de desarrollo del mismo.

Se concluye, además, que todos los artefactos generados por la metodología en esta etapa guiarán de forma efectiva el capítulo de implementación y prueba.

CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBA.

3.1 Introducción.

En este capítulo se abordarán dos de las fases más importantes en el ciclo de vida de cualquier software: Implementación y Prueba, en esta fase se materializará el software mediante su codificación y se realizarán las pruebas necesarias para comprobar si fueron cumplidos los objetivos trazados durante la concepción del SISTEMA-DSI.

3.2 Fase de implementación.

Como parte de la metodología ágil escogida durante el inicio de cada iteración se revisa el plan de iteraciones y se expresan las tareas de programación, donde a cada una de ellas le es asignada el programador responsable, estas tareas son descritas en un lenguaje técnico que no tienen por qué ser entendible para el cliente. (43). Como parte de la planificación realizada en el capítulo anterior se detallan a continuación las iteraciones y las historias de usuario implementadas en cada iteración.

3.2.1 Iteración 1.

En esta iteración se implementaron las historias de usuarios referentes a las principales funcionalidades de la aplicación Web con el objetivo de poner en producción el primer prototipo funcional de la solución.

Tarea de Ingeniería	
Número Tarea: 1	Número Historia de Usuario: 6
Nombre Tarea: Mostrar Ip Comunes.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0,6
Fecha Inicio: 01 de mayo de 2015	Fecha Fin: 02 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se diseña la interfaz de la funcionalidad Mostrar Ip comunes, que devuelva todos los parámetros vistos en la HU. Debe tener en la misma los parámetros usuario, servicio y rango de fechas y devolverá el usuario, los Ip comunes y la cantidad de conexiones a cada Ip.	

Tabla 29: Tarea de Ingeniería # 1, HU # 6.

Tarea de Ingeniería	
Número Tarea: 2	Número Historia de Usuario: 6
Nombre Tarea: Implementar la funcionalidad Mostrar Ip comunes.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 1,0
Fecha Inicio: 02 de mayo de 2015	Fecha Fin: 03 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se codifica la funcionalidad Mostrar Ip comunes que permite, una vez introducidos los datos, poder determinar, por cantidad de conexiones a una misma dirección Ip, cuales son consideradas comunes y cuáles no. Se busca todas las direcciones Ip que cumplan con los parámetros indicados por el usuario.	

Tabla 30: Tarea de Ingeniería # 2, HU # 6.

Tarea de Ingeniería	
Número Tarea: 1	Número Historia de Usuario: 8
Nombre Tarea: Diseño de la interfaz Mostrar Simultaneidad de uso.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 1
Fecha Inicio: 03 de mayo de 2015	Fecha Fin: 04 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se devuelve todos los elementos vistos en la HU. Debe tener en la misma los parámetros usuario, servicio y rango de fechas y devolverá el usuario, los Ip simultáneos, la fecha, la hora y la diferencia de tiempo entre las conexiones simultáneas.	

Tabla 31: Tarea de Ingeniería # 1, HU # 8.

Tarea de Ingeniería	
Número Tarea: 2	Número Historia de Usuario: 8
Nombre Tarea: Implementar la funcionalidad Mostrar Simultaneidad de uso.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 2
Fecha Inicio: 04 de mayo de 2015	Fecha Fin: 05 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se codifica la funcionalidad Mostrar Simultaneidad de uso, que permite, realizar consultas que permitan establecer similitudes en cuanto a patrones comunes. Para ello, se obtienen los parámetros introducidos por el usuario, se filtran los registros, ordenan por tiempo, comparan por Ip y por tiempo, y luego se devuelve el resultado esperado.	

Tabla 32: Tarea de Ingeniería # 2, HU # 8.

3.2.2 Iteración 2.

En esta iteración se implementaron las historias de usuarios referentes a la segunda iteración del desarrollo del Sistema para el Monitoreo del uso de los servicios telemáticos.

Tarea de Ingeniería	
Número Tarea: 1	Número Historia de Usuario: 1
Nombre Tarea: Diseño de la interfaz Autenticar usuario.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0,1
Fecha Inicio: 12 de mayo de 2015	Fecha Fin: 13 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se diseña la interfaz de la funcionalidad Autenticar usuario que permite introducir los datos para poder identificar al usuario que acceda al sistema. Debe tener las credenciales de acceso: usuario y contraseña y una opción para recordar las mismas en el sistema.	

Tabla 33: Tarea de Ingeniería # 1, HU # 1.

Tarea de Ingeniería	
Número Tarea: 2	Número Historia de Usuario: 1
Nombre Tarea: Implementar la funcionalidad Autenticar Usuario.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0,1
Fecha Inicio: 13 de mayo de 2015	Fecha Fin: 14 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se codifica la funcionalidad Autenticar usuario, que permite autenticar en el sistema a los usuarios. Para ello se consulta en la Base de Datos si los datos introducidos son correctos.	

Tabla 34: Tarea de Ingeniería # 2, HU # 1.

3.2.3 Iteración 3.

En esta iteración se implementaron las historias de usuarios referentes a la tercera iteración del desarrollo del Sistema para el Monitoreo del uso de los servicios telemáticos.

Tarea de Ingeniería	
Número Tarea: 1	Número Historia de Usuario: 9
Nombre Tarea: Diseño de la interfaz Configurar Ip Comunes.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0,1
Fecha Inicio: 18 de mayo de 2015	Fecha Fin: 19 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se diseña la interfaz de la funcionalidad Configurar Ip Comunes que permite modificar la cantidad de Ip comunes que son considerados por la DSI. Por defecto este valor es	

5. Se debe mostrar un mensaje con el tipo de Configuración a realizar, y se debe dar la opción de cambiar esta configuración, validando que sea un valor correcto, o sea, un número.

Tabla 35: Tarea de Ingeniería # 1, HU # 9.

Tarea de Ingeniería	
Número Tarea: 2	Número Historia de Usuario: 9
Nombre Tarea: Implementar la funcionalidad Configurar Ip Comunes.	
Tipo de Tarea: Desarrollo	Puntos Estimados: 0,3
Fecha Inicio: 19 de mayo de 2015	Fecha Fin: 20 de mayo de 2015
Programador Responsable: Manuel Reina Aguilera	
Descripción: Se codifica la funcionalidad Configurar Ip Comunes, que muestra y modifica los valores de esta configuración que sirve para el método IpComunes. Para ello se obtiene el valor introducido por el usuario y se guarda en la base de datos, este valor luego es obtenido en los métodos IpComunes.	

Tabla 36: Tarea de Ingeniería # 2, HU # 9.

3.3 Pruebas.

Las pruebas de software tienen como objetivos detectar errores no encontrados hasta el momento en la aplicación. Se puede hablar entonces del éxito de las pruebas siempre y cuando se hallen errores. Con las pruebas se puede además observar hasta qué punto el software parece funcionar en concordancia con las HU descritas. Comprobar el funcionamiento de programas puede convincentemente demostrar la presencia de errores, pero nunca puede demostrar su ausencia (44).

Para comprobar el correcto funcionamiento del sistema, se realizará un plan de pruebas, que permitirá emplear los métodos de prueba de aceptación o pruebas funcionales y las pruebas de carga y estrés.

3.3.1 Plan de pruebas.

Los objetivos fundamentales para diseñar un plan de pruebas radican en la obtención de las configuraciones óptimas para el despliegue de la aplicación, la validación del estimado

de carga y la concurrencia que se espera soporte la misma en condiciones extremas. Por otra parte, permiten la detección y localización de posibles errores existentes en el software.

Las pruebas fueron diseñadas para probar la estabilidad y concurrencia soportada por la aplicación, por encima de la velocidad de respuesta; debido a que se considera que la estabilidad y fiabilidad de la aplicación posee mayor importancia en situaciones extremas (evaluación de soluciones, crawler⁸, alta concurrencia y realización de concursos) que la rapidez en los tiempos de respuesta.

Para la realización de las pruebas se utilizó un entorno de hardware y software consistente en:

- Laptop Dell con procesador Core i-3 ~ 2.20 GHz y 4 GB de RAM.
- Sistema operativo Ubuntu en su versión 14.10.
- Entorno de ejecución java (JRE) en su versión 1.7.
- Servidor Web Apache.
- Servidor de base de datos PostgreSQL en su versión 9.4.

Antes de iniciar la prueba se identificaron 3 páginas fundamentales que recogen el mayor número de peticiones realizadas al servidor debido a que manejan grandes volúmenes de datos y son solicitadas con mayor frecuencia por los usuarios. El plan de pruebas fue diseñado para simular un escenario que supere la concurrencia real estimada para el sistema. Las pruebas de rendimiento simularán una concurrencia de 350 usuarios realizando 1 solicitud de monitoreo en intervalos de 1 segundo de forma concurrente y las de aceptación validarán las historias de usuario identificadas como funcionalidades del sistema.

3.3.2 Pruebas de aceptación.

Dentro de las pruebas de aceptación se utilizarán las pruebas de caja negra que se crean a partir de las HU y son realizadas por el cliente y los usuarios finales de la aplicación. En ellas serán probadas las funcionalidades exigidas por el cliente. Luego de haber superado las pruebas de aceptación podrá considerarse que la aplicación es apta para el uso y despliegue.

⁸ Un crawler o araña web es un programa que inspecciona las páginas del World Wide Web de forma metódica y automatizada.

Para eliminar la influencia de conflictos de intereses, y para que sea lo más objetiva posible, la prueba de aceptación nunca debería ser responsabilidad de los ingenieros de software que han desarrollado el producto. Para la preparación, la ejecución y la evaluación de la prueba de aceptación ni siquiera hacen falta conocimientos informáticos. Sin embargo, un conocimiento amplio de métodos y técnicas de prueba y de la gestión de la calidad en general facilita esta labor (45).

Las pruebas de aceptación se llevarán a cabo de la forma siguiente:

1. Se redactarán los casos de prueba teniendo en cuenta el orden de las HU y los niveles de prioridad.
2. Se hará la planificación con el cliente de cuándo y cuáles pruebas serán llevadas a cabo.
3. Se reunirán los miembros del proyecto seleccionados para realizar las pruebas.
4. Se completará cada uno de los campos de la tabla de pruebas de aceptación con el resultado.

La persona adecuada (o el equipo adecuado) para llevar a cabo la prueba de aceptación dispone de estos conocimientos y además es capaz de interpretar los requerimientos especificados por los futuros usuarios del sistema del software en cuestión (45).

➤ **Casos de prueba.**

Caso de pruebas de aceptación	
Código Caso de Prueba: 01	Nombre Historia de Usuario: Autenticar usuario.
Nombre de la persona que realiza la prueba: Manuel Reina Aguilera	
Descripción de la Prueba: Un usuario accede al formulario de autenticación (<a href="http://<servidor>/sistema/login">http://<servidor>/sistema/login) donde llena los datos personales requeridos para dicho proceso cumpliendo con todos los requisitos propuestos para cada uno de los campos.	
Condiciones de ejecución: El usuario no debe estar autenticado. Tiene que existir el usuario registrado en la base de datos.	

<p>Entradas:</p> <ul style="list-style-type: none"> - Nombre de usuario. - Contraseña.
<p>Resultado esperado: Se autentica un usuario al sistema y se permite el acceso sistema de dicho usuario.</p>
<p>Evaluación: Satisfactorio</p>

Tabla 37: Caso de Prueba # 01.

En la siguiente prueba se evaluará el proceso de registro de un usuario en el sistema, cuando dicho usuario omite el llenado de algunos campos obligatorios, provee una credencial de acceso que no cumple con el formato adecuado, o cualquier otro error.

Caso de pruebas de aceptación	
Código Caso de Prueba: 02	Nombre Historia de Usuario: Autenticar usuario.
Nombre de la persona que realiza la prueba: Manuel Reina Aguilera	
Descripción de la Prueba: Un usuario accede al formulario de autenticación (<a href="http://<servidor>/sistema/login">http://<servidor>/sistema/login) donde llena los datos personales requeridos para dicho proceso de forma parcial o con errores.	
Condiciones de ejecución: El usuario no debe estar autenticado. Tiene que existir el usuario registrado en la base de datos.	
<p>Entradas:</p> <ul style="list-style-type: none"> - Nombre de usuario. - Contraseña 	
Resultado esperado: El sistema alerta al usuario de los errores cometidos a la hora de completar el formulario indicando para cada campo un mensaje de error.	

Evaluación: Satisfactorio

Tabla 38: Caso de Prueba # 02.

3.3.3 Pruebas de rendimiento.

Este tipo de pruebas, están dirigidas a evaluar la conformidad de un sistema o componente con requerimientos de desempeño específicos. Normalmente esto se lleva a cabo usando una herramienta de prueba automática para simular un gran número de usuarios, carga y volumen de información y para monitorear el desempeño del hardware (46). La herramienta seleccionada para llevar a cabo las pruebas de rendimiento fue JMeter.

➤ **Resultados.**

Al culminar la prueba se puede concluir que la aplicación se encuentra lista para soportar una concurrencia muy superior a la que estará sometida; dígase 350 usuarios concurrentes realizando cada uno 3 peticiones a las 3 páginas que mayor consumo de recursos posee la aplicación. A continuación se mostrarán los resultados de las pruebas arrojados por el software JMeter a través de gráficas.

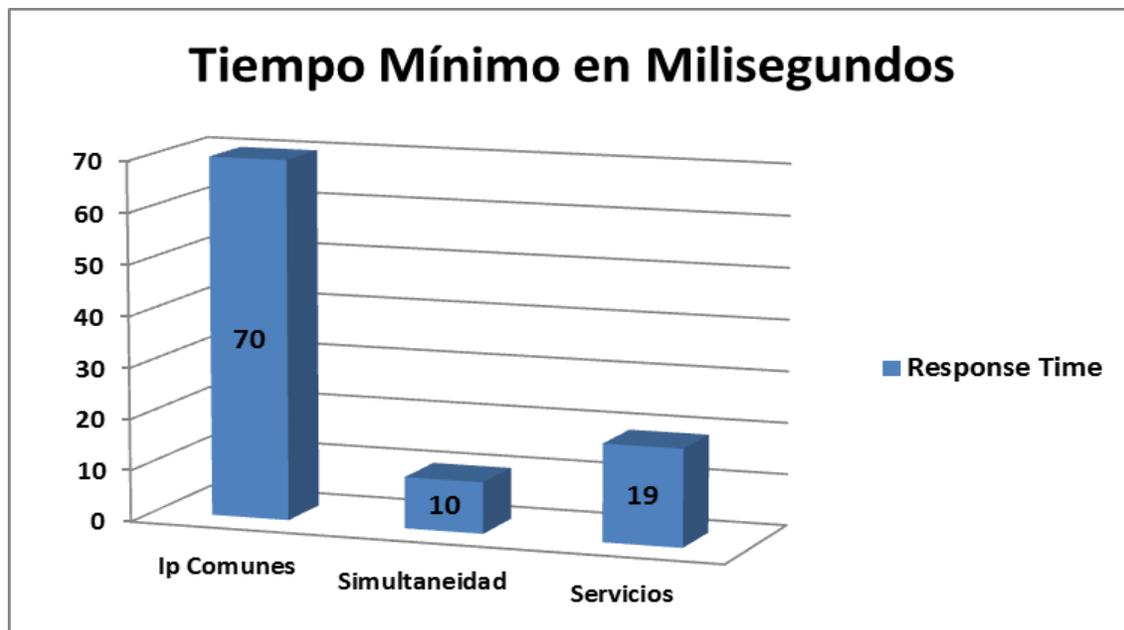


Figura 5: Tiempo mínimo de respuesta de una petición.

La gráfica anterior muestra el tiempo mínimo tomado por una petición para cada una de las páginas; nótese que la petición más rápida tardó solo 10 milisegundos en ser atendida

sobre la simultaneidad de uso de un servicio. Es importante destacar que estos casos extremos son difíciles de repetir en un ambiente donde el sistema se use por varios usuarios simultáneos.

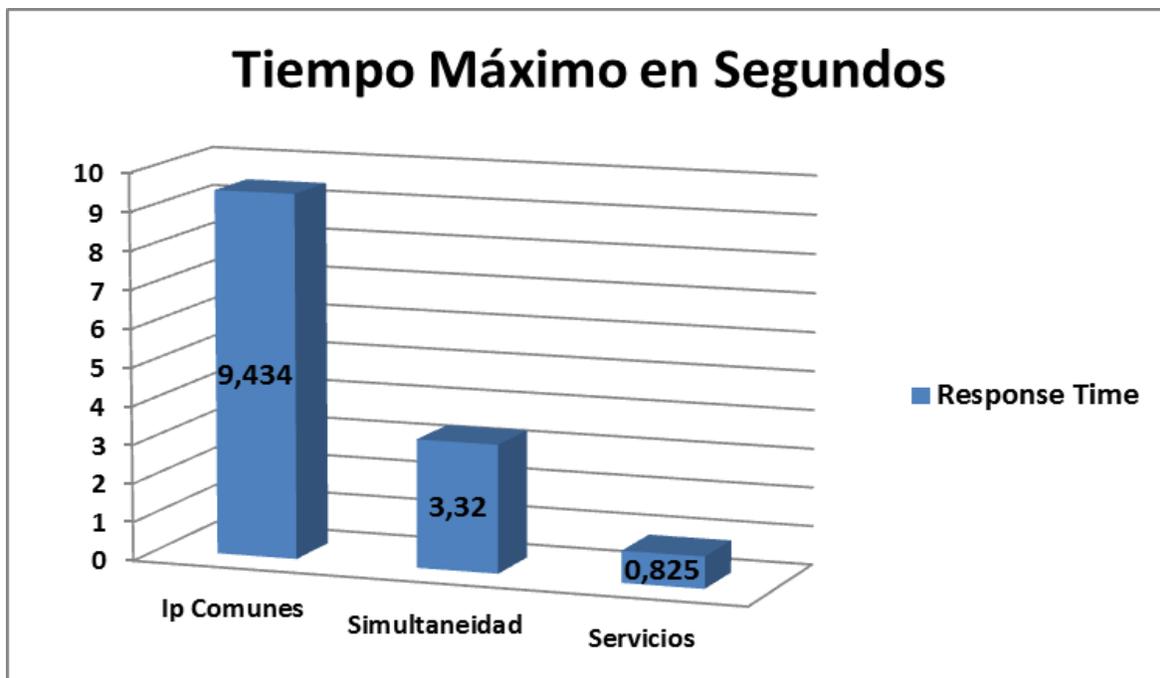


Figura 6: Tiempo máximo de respuesta de una petición.

La imagen muestra los tiempos máximos de respuesta de las peticiones realizadas a cada una de las páginas escogidas para la prueba; siendo el tiempo más alto de respuesta de 9,4 segundos. Teniendo en cuenta que las condiciones de hardware utilizadas para realizar las pruebas no son las óptimas y que la prueba simula un ambiente con mayor demanda que la mayor estimada para el sistema en un entorno real, aun en los casos más extremos (realización de Ip Comunes y Simultaneidad) el tiempo máximo resultante es considerablemente más bajo del esperado y permite tomar medidas para evitar que el sistema colapse ante esta situación.

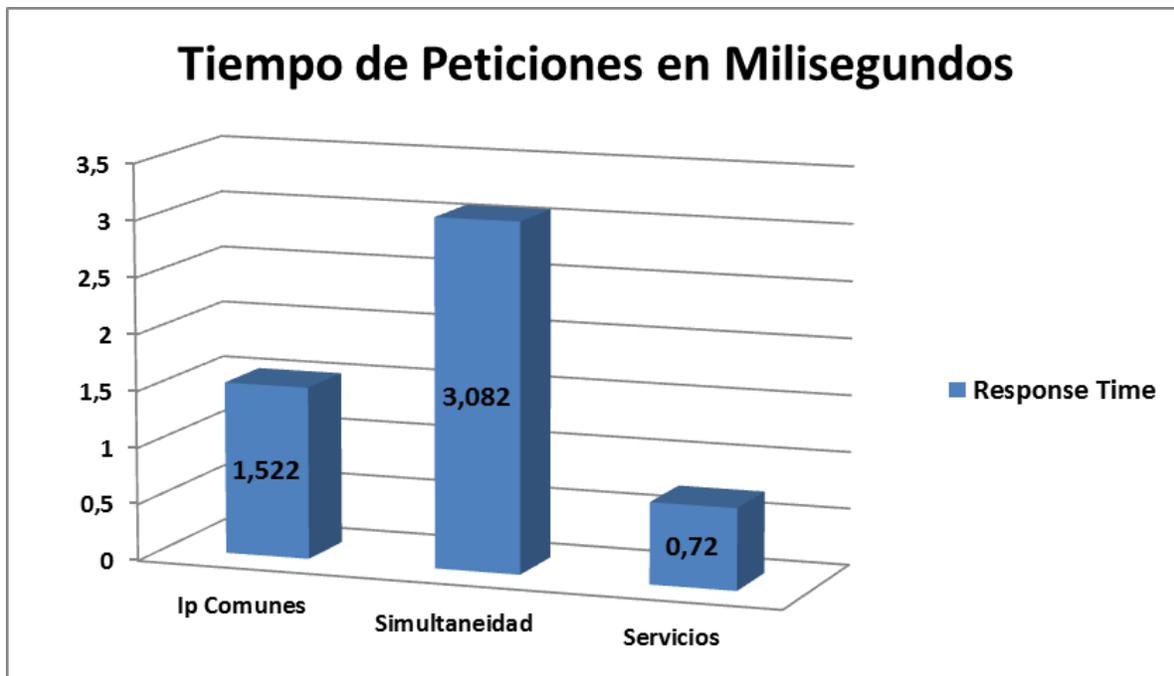


Figura 7: Línea del 90% de las peticiones.

La imagen indica el tiempo de respuesta del 90% de las peticiones para cada una de las páginas solicitadas en la prueba. La página más lenta resulta ser “simultaneidad” lo cual es esperado debido a que en dicha página se manejan grandes volúmenes de información.

3.4 Conclusiones parciales.

En el presente capítulo se abordaron las tareas de ingeniería, las cuales permitieron describir las tareas realizadas para implementar las funcionalidades, luego de la etapa de implementación se dio paso las pruebas, con estas fueron validados los requisitos trazados.

Se considera que el uso del framework Symfony, como base para el desarrollo de la solución, agilizó el proceso de implementación de la misma; además de permitir la obtención de un sistema modular que facilita la integración de nuevas funcionalidades mediante módulos independientes. Asimismo, la utilización del patrón arquitectónico Modelo-Vista-Controlador permitió la distribución de los recursos.

CONCLUSIONES

Luego del desarrollo de la investigación, se pudo arribar a las siguientes conclusiones:

- Los sistemas actuales para el análisis de trazas no permiten la identificación de posibles violaciones de Seguridad Informática relacionadas a credenciales de acceso en el uso de los servicios telemáticos de mensajería instantánea, correo electrónico y navegación por Internet en la universidad.
- La metodología de desarrollo de software XP permitió el desarrollo de un proyecto en poco tiempo como se necesitaba, y junto a las herramientas de desarrollo utilizadas, apoyó la creación de la solución propuesta.
- La herramienta obtenida minimiza el tiempo que emplea la DSI en solucionar incidentes de robo y facilitación de credenciales de acceso.

RECOMENDACIONES

El desarrollo de este trabajo de diploma plantea la siguiente recomendación:

- Realizar un estudio de otros tipos de trazas almacenados en los servidores de la universidad, que permitan la incorporación de otros servicios a este análisis de posibles violaciones a la Seguridad Informática relacionadas a las credenciales de acceso en el uso de la mensajería instantánea, el correo electrónico y la navegación por Internet.

ANEXOS

BIBLIOGRAFÍA.

1. Jacobson, Ivar, Booch, Grady. El proceso Unificado de Desarrollo de Software. Madrid: Addison Wesley, 2000.
2. Ruiz, Francisco y López, Patricia. Bienvenido a OpenCourseWare - OCW Universidad de Cantabria. [En línea] [Citado el: 27 de Abril de 2014.] <http://ocw.unican.es/enseanzas-tecnicas/ingenieria-del-software-i/materiales-de-clase-1/is1-t02-trans.pdf>.
3. Inicio | Facultad de Ingeniería. [En línea] [Citado el: 27 de Abril de 2014.] <http://www.fing.edu.uy/inco/cursos/ingsoft/pis/proceso/MUM/images/fases.jpg>.
4. Palacio, Juan. Flexibilidad con SCRUM. [En línea] 2008. [Citado el: 15 de Mayo de 2014.] www.navegapolis.net/content/view/694/.
5. Manifiesto por el Desarrollo Ágil de Software. [En línea] [Citado el: 15 de Mayo de 2014.] <http://agilemanifesto.org/>.
6. Somerville, Ian. Requisitos del Software. [En línea] 2004. [Citado el: 2 de abril de 2013.] <http://lsi.ugr.es/~ig1/docis/requeintro.pdf>.
7. Modelado de datos: Fundamentos de diseño de bases de datos. [En línea] [Citado el: 27 de Abril de 2014.] <http://elvex.ugr.es/idbis/db/docs/intro/C%20Modelado%20de%20datos.pdf>.
8. RAMÍREZ MONTAÑEZ, Jorge Enrique. Análisis, evaluación de riesgos y asesoramiento de la seguridad informática en el área de redes y sistemas de la Alcaldía de Pamplona Norte de Santander. 2015.
9. Blog Académico. Analizadores de Log [en línea]. 2015 [fecha de consulta: 19 de marzo del 2015]. Disponible en: https://marlisq.wordpress.com/administracion_web/analizador-de-logs/
10. LUCERO, Sandra Elizabeth Cerón; VILLARREAL, Gabriel Arturo Lomas; MONTALVÁN, Ángela Olivia Yanza. INGENIERO EN SISTEMAS COMPUTACIONALES. 2005.

11. Colaboradores de Wikipedia. Log (registro) [en línea]. Wikipedia, La enciclopedia libre, 2015 [fecha de consulta: 29 de marzo del 2015]. Disponible en <[http://es.wikipedia.org/w/index.php?title=Log_\(registro\)&oldid=79942381](http://es.wikipedia.org/w/index.php?title=Log_(registro)&oldid=79942381)>.
12. Blog Académico. Analizadores de Log [en línea]. 2015 [fecha de consulta: 19 de marzo del 2015]. Disponible en: https://marlisq.wordpress.com/administracion_web/analizador-de-logs/.
13. PATNAIK, Rachna; MISHRA, Mukesh Kumar. Role of Content Management Software (CMS) in libraries for information dissemination. En Emerging Trends and Technologies in Libraries and Information Services (ETTLIS), 2015 4th International Symposium on. IEEE, 2015. p. 117-121.
14. DOMOSŁAWSKA, Beata; SZYMAŃSKA, Zyta. "Polska Bibliografia Literacka" on-line. Tworzenie bazy, ograniczenia, innowacje i dalszy rozwój. 2009.
15. WHITEHOUSE, Kamin. The design of calamari: an ad-hoc localization system for sensor networks. University of California at Berkeley, 2002.
16. AMBRE, Amruta; SHEKOKAR, Narendra. Insider Threat Detection Using Log Analysis and Event Correlation. Procedia Computer Science, 2015, vol. 45, p. 436-445.
17. KAIZER, Andrew J.; GUPTA, Minaxi. \ sim Open Resolvers: Understanding the Origins of Anomalous Open DNS Resolvers. En Passive and Active Measurement. Springer International Publishing, 2015. p. 3-14.
18. GHOSH, Partha; MANDAL, Abhay Kumar; KUMAR, Rupesh. An Efficient Cloud Network Intrusion Detection System. En Information Systems Design and Intelligent Applications. Springer India, 2015. p. 91-99.
19. ANDERSCH, Adrienn, et al. US Hardwood Sawmill Log Procurement Practices. BioResources, 2015, vol. 10, no 1, p. 1224-1244.
20. HURD, Steven A.; STAMP, Jason E.; CHAVEZ, Adrian R. OPSAID Initial Design and Testing Report. Report, Sandia National Laboratories, 2007
21. Oracle Corporation. [En línea] 2015. [Citado el: 18 de abril de 2015.] <http://www.oracle.com>.
22. CANETTI, Ran; GOLDREICH, Oded; HALEVI, Shai. The random oracle methodology, revisited. Journal of the ACM (JACM), 2004, vol. 51, no 4, p. 557-594.

23. PostgreSQL. [En línea] 2015. [Citado el: 18 de abril de 2015.] <http://www.postgresql.org>.
24. XU, Mei. Distribution Curve of Three-dimensional Space of Using Frequency Based on the Interface of Java3D and MySQL Database. En 2015 International Conference on Automation, Mechanical Control and Computational Engineering. Atlantis Press, 2015.

REFERENCIAS BIBLIOGRÁFICAS.

1. ALFONSO MORENO, Adriana, et al. Responsabilidades legales derivadas de internet y del comercio electrónico [recurso electrónico]. 2013.
2. Portal de Seguridad Informática. [En línea] 2015. <http://seguridad.uci.cu/?q=mision>.
3. Portal de Seguridad Informática. [En línea] 2015. <http://seguridad.uci.cu/?q=mision>.
4. Beekman G. Seguridad informática. E.U.A.: Mc Graw Hill.p.29, 13, 46. (1999).
5. Fielding, R.; Gettys, J.; Mogul, J.; Frystyk, H.; Masinter, L.; Leach, P.; Berners-Lee, T. Hypertext Transfer Protocol — HTTP/1.1. June 1999.
6. Colaboradores de Wikipedia. Log (registro) [en línea]. Wikipedia, La enciclopedia libre. [Fecha de consulta: 29 de marzo del 2015]. Disponible en <[http://es.wikipedia.org/w/index.php?title=Log_\(registro\)&oldid=79942381](http://es.wikipedia.org/w/index.php?title=Log_(registro)&oldid=79942381)>, 2015.
7. GONZÁLEZ-CASTAÑO, Idis, et al. Una metodología para la detección de anomalías en una red a partir de las trazas de ejecución de los servicios.
8. PÉREZ FERNÁNDEZ, Carmen, et al. Integración LDAP con el Servicio PAPI. 2009.
9. MARTÍNEZ, Ramón A. Anglada; HERNÁNDEZ, Alain Abel Garófalo. PLATAFORMA DE GESTIÓN DE SERVICIOS TELEMÁTICOS EN GNU/LINUX. Revista Telem@tica, 2013, vol. 11, no 3, p. 46-57.
10. Portal de Seguridad Informática. [En línea] 2015. <http://seguridad.uci.cu/mensajeria-instantanea>.
11. Portal de Seguridad Informática. [En línea] 2015. <http://seguridad.uci.cu/content/algunos-cambios-en-la-configuracion-del-correo-zimbra>
12. MANTILLA, Susana Pérez; LÓPEZ, Aryam Gutiérrez; HERNÁNDEZ, Alain A. Garófalo. HERRAMIENTA PARA EL ANÁLISIS DE LA PERTINENCIA DE LA NAVEGACIÓN POR INTERNET. Revista Telem@tica, 2013, vol. 11, no 3, p. 11-20.
13. AN, Yuntao. Coarse-resolution CT scanning for sawmill logs sorting and grading. 2013.
14. BOLDYREVA, Alexandra; PALACIO, Adriana; WARINSCHI, Bogdan. Secure proxy signature schemes for delegation of signing rights. Journal of Cryptology, 2012, vol. 25, no 1, p. 57-115.

15. DILLEY, John; ARLITT, Martin; PERRET, Stephane. Enhancement and validation of Squid's cache replacement policy. HP Laboratories Technical Report HPL, 1999, no 69.
16. VOGELMAN, Julie. Determining Web Usability Through an Analysis of Server Logs. 2001. Tesis Doctoral. University of Virginia.
17. SANTANA, Tamara Betancourt, et al. Analizador de registros proxy para auditores Proxy Log Analyzer for auditors.
18. LEÓN, Gonzalo Serrano, Ingeniería de Sistemas de Software. 2011, 4ta edición, p. 6, 220.
19. RAY, Kisor; BAG, Sourav; SARKAR, Saumen. Easy and Fast Design and Implementation of PostgreSQL based image handling application. arXiv preprint arXiv: 1503.05294, 2015.
20. EduMéxico.net. [En línea] [Citado el: 21 de Abril de 2015.] http://edumexico.net/secundaria/plan%20de%20estudios/computacion2/apuntes_sec9.htm.
21. Wikispaces. [En línea] [Citado el: 21 de Abril de 2015.] [https://www.wikispaces.com/file/view/Lenguajes formales.docx](https://www.wikispaces.com/file/view/Lenguajes+formales.docx).
22. González Duque, Raúl; DSpace at Universia: Home. [En línea] [Citado el: 21 de Abril de 2014.] http://dspace.universia.net/bitstream/2024/919/1/Python_para_todos.pdf.
23. Deitel, Paul J y Deitel, Harvey M. Cómo programar en Java. México: s.n., 2008. 978-970-26-1190-5.
24. <http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/introduccion/virtual.htm>. La Máquina Virtual Java. [En línea] [Citado el: 25 de Abril de 2015.] <http://www.sc.ehu.es/sbweb/fisica/cursoJava/fundamentos/introduccion/virtual.htm>.
25. LCC: Departamento de Lenguajes y Ciencias de la Computación. [En línea] [Citado el: 27 de Abril de 2015.] <http://www.lcc.uma.es/~av/Publicaciones/02/MarcoConceptualJISBD02.pdf>.
26. OLANREWAJU, Rashidah F.; ISLAM, Thouhedul; ALI, N. An Empirical Study of the Evolution of PHP MVC Framework. En Advanced Computer and Communication Engineering Technology. Springer International Publishing, 2015. p. 399-410.

27. Servidor web - Wikipedia, la enciclopedia libre. [En línea] [Citado el: 21 de Abril de 2014.] http://es.wikipedia.org/wiki/Servidor_Web.
28. Apache Tomcat - Welcome! [En línea] [Citado el: 21 de Abril de 2015.] <http://tomcat.apache.org/>.
29. Reyes González, Yunia, González Bravo, Lisbet M. y Ruiz Constanten, Yadira. Sitio web del ISPJAE. [En línea] [Citado el: 27 de Abril de 2015.] <http://ccia.cujae.edu.cu/index.php/siia/siia2008/paper/viewFile/1240/298>.
30. [En línea] [Citado el: 8 de Mayo de 2015.] http://1.bp.blogspot.com/-pC7wKR2aGUI/T43Db6PXiml/AAAAAAAAACM/LJ5re9L_zA8/s1600/agil.bmp.
31. METODOLOGIAS_TRADICIONALES_VS._METODOLOGIAS_AGILES. [En línea] [Citado el: 27 de Abril de 2015.] http://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_1_Procesos/Metodologias/METODOLOGIAS_TRADICIONALES_VS._METODOLOGIAS_AGILES.pdf.
32. Apache JMeter - Apache JMeter. [En línea] [Citado el: 4 de Mayo de 2014.] <https://jmeter.apache.org/>.
33. JMeter. Manual de usuario v1.2.pdf. [En línea] [Citado el: 28 de Abril de 2014.] <http://www.ejie.net/documentos/Herramientas/JMeter.%20Manual%20de%20usuario%20v1.2.pdf>.
34. Delgado, H., Losavio, F., & Matteo, A. (2013, October). Goal oriented techniques and methods: Goal refinement and levels of abstraction. In Computing Conference (CLEI), 2013 XXXIX Latin American (pp. 1-12). IEEE.
35. CHRISTENSEN, David. Re: dm-crypt/LUKS Rendimiento. 2014.
36. LETELIER, Patricio. Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). 2006.
37. Letelier, Patricio y Penadés, M^a Carmen. Metodologías ágiles para el desarrollo de software: extreme Programming (XP).
38. Dpto. Lenguajes y Sistemas Informáticos. Universidad de Alicante. [En línea] 29 de Febrero de 2012. [Citado el: 5 de Mayo de 2015.] <http://rua.ua.es/dspace/bitstream/10045/21018/1/transp.pdf>.

39. Larman, Craig. UML y Patrones. Introducción al análisis y diseño orientado a objetos. 1999.
40. Welicki, L. E., & Aguilar, L. J. (2014). Meta-especificación y catalogación de patrones de software con lenguajes de dominio específico y modelos de objetos adaptativos: una vía para la gestión del conocimiento en la ingeniería del software.
41. ROJAS, MC Juan Carlos Olivares. Patrones de Diseño.
42. ESCRIBANO, Gerardo Fernández. Introducción a Extreme Programming. Introducción a Extreme Programming, 2002.
43. MOLPECERES, Alberto. Procesos de desarrollo: RUP, XP y FDD. 2002.
44. Wybe Dijkstra, Edsger. Sobre la crueldad de verdaderamente enseñar ciencias de la computación. [En línea] [Citado el: 4 de Mayo de 2015.] http://www.smaldone.com.ar/documentos/ewd/sobre_la_crueldad.html.
45. La prueba de aceptación es la prueba más importante para los productos software. [En línea] [Citado el: 28 de Abril de 2015.] <http://pruebasdesoftware.com/pruebadeaceptacion.htm>.
46. Pruebas de Rendimiento de Software. [En línea] [Citado el: 28 de Abril de 2014.] <http://www.greensqa.com/portal/soluciones/calidad-de-productos/55-pruebas-de-rendimiento-de-software>.

Anexo 1: Muestra de formato de las trazas de Internet, mensajería instantánea y Correo en la UCI.

➤ **Trazas de Internet.**

1430465943.634 235 10.8.44.105 TCP_MISS/200 943 POST http://ocsp.digicert.com/mreina DIRECT/72.21.91.29 application/ocsp-response
1430466059.822 117735 10.8.44.105 TCP_MISS/200 6697 CONNECT self-repair.mozilla.org:443 mreina DIRECT/54.192.81.35 –
1430489602.750 7759 10.8.111.241 TCP_MISS/200 4563 CONNECT chat.facebook.com:443 mreina DIRECT/173.252.122.5 –

➤ **Trazas de mensajería instantánea.**

2015.05.01 00:39:34 [jabber.uci.cu] Login by mreina from 10.8.44.105 (secure)
2015.05.01 00:39:36 Packet sent to unreachable address <iq to="mreina@jabber.uci.cu/149bdbfd" type="get" id="105" from="Irdguezglez@jabber.uci.cu/Gajim"><query xmlns="http://jabber.org/protocol/disco#info"/></iq>
2015.05.01 19:14:16 [jabber.uci.cu] Logout by mreina from 10.8.44.105 (secure)

➤ **Trazas de Correo.**

2015-05-01 00:40:02,522 INFO [qtp1545357780-39316:http://127.0.0.1:8080/service/soap/AuthRequest]
[name=mreina@estudiantes.uci.cu;oip=10.8.44.105;ua=zclient/8.0.5_GA_5839;] security - cmd=Auth; account=mreina@estudiantes.uci.cu; protocol=soap;
2015-05-01 01:45:24,033 INFO [qtp1160981642-2773703:http://10.0.0.81:8080/service/soap/NoOpRequest]
[name=mreina@estudiantes.uci.cu;ip=10.0.0.82;ua=ZCS/8.0.5_GA_5839;via=10.8.44.105 (ZimbraWebClient - FF37 (Win)/8.0.5_GA_5839);] soap - NoOpRequest elapsed=5