Facultad 2

Herramienta para el procesamiento de peticiones de la navegación web.

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

Autores: Yosiel Quesada Fonseca
Yuliet Hernández Gainza

Tutores: Ing. Jorge Armando Túñez González
Ing. Luis Manuel Gonçalves Torres

La Habana, 2017

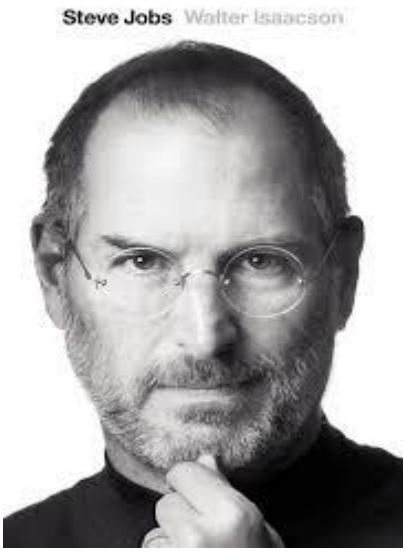
"Año 59 de la Revolución"

Declaración de autoría

Declaramos ser autores del presente trabajo de diploma y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

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

_____	_____
Yosiel Quesada Fonseca	Yuliet Hernández Gainza
Autor	Autor
_____	_____
Jorge Armando Túñez González	Luis Manuel Gonçalves Torres
Tutor	Tutor



"Pienso que, si haces algo y resulta ser una buena idea, entonces debes hacer otras cosas increíbles, no lo pienses mucho tiempo. Sólo descubre qué es lo que sigue."

Steve Jobs

Agradecimientos

Yuliet

A mi madre, por haberme dado la vida, por haber confiado siempre en mí y darme las fuerzas para seguir mis estudios.

A mi papi, que lo es todo para mí, que hoy no está presente aquí por circunstancias de la vida, pero sé que cuando escuche la noticia de que ya soy "ingeniería en ciencias informáticas: se va a sentir muy orgulloso de mi, ¨de su Barby¨, como me dice cariñosamente.

A mi hermana, que me ha dado los dos regalos más lindos de mi vida mis sobrinos Brayan y Aaron.

A mis abuelos maternos y paternos, en especial a mi abuela materna Luisa, por malcriarme tanto.

A los abuelos de mi hermana, que son como mis abuelos también, en especial a Benito porque siempre confió en mí y sabía que algún día llegaría a graduarme y ser una "científica" como me dice él.

A mi tío Chacho por haberme ayudado siempre en todo y nunca tener un "no" como respuesta para mí. Por aconsejarme en los momentos buenos y malos de mi vida.

A mi tía Yisry por malcriarme tanto.

A mis tíos: Juan Luis, Maite, Tony, Odalis, Neni, Mercedes.

A Yaima por escuchar y darme consejos siempre que lo he necesitado.

A todos mis primos, en especial a Dianelis y Alain.

A mis dos amigas incondicionales Adaluz y Olga por estar siempre a mi lado en todo momento.

Agradecimientos

A mis tutores.

A todos los profesores que he tenido durante la carrera.

A todos mis compañeros de aula, en especial a las nenas de mi apartamento: Sandra, Yeny, Daynela,Y anira,Yenlis y a Yadian que no me podía faltar.

A la familia, que ellos saben quiénes son, por haberme invitado a algunas de sus fiestas y por las que no me invitaron también.

A todos los amigos que he conocido en mi transcurso por la universidad.

A mi compañero de tesis, porque sí.

Y por último a mí.

Yosiel

-agradecerle a mi madre que es lo que más quiero en la vida por toda la paciencia y la fe que depositó en mí.

-agradecerle a mi padre que descansé en paz que siempre lo tuve presente cuando no tenía a nadie al lado.

-agradecerle a mi hermano y a su esposa por haberme ayudado en cada momento que los necesité.

-agradecerle a mi padrastro por haberme dado todos esos consejos sobre la vida y haber confiado en mí.

-agradecerle a mi abuela y a mi tío por su ayuda y consejos.

-agradecerle a mi novia por haber llegado a mi vida y aguantar mis pesadeces en este momento tan difícil.

Agradecimientos

-agradecerles a mis hermanos de universidad (incluyendo a juan miguel) por todos estos años buenos que he pasado juntos a ellos.

-agradecerle a toda la familia que no he mencionado que me apoyaron de una manera u otra.

-agradecerles a todas mis amistades del aula y de la universidad.

-agradecerles a los tutores.

-por último agradecerle a mi compañera de tesis por haberme ayudado en los cinco años de carrera y así poder lograr uno de mis sueños.

Dedicatoria

A toda mi familia, especialmente a mis padres Niurka y Carlos por haberme apoyado en todos los momentos de mi vida.

Yuliet Hernández Gainza

A toda mi familia por haberme apoyado en todo el transcurso de mi formación como profesional.

Yosiel Quesada Fonseca

Resumen

La Universidad de las Ciencias Informáticas es una institución universitaria que cuenta con una red interna, que contiene un gran número de computadoras conectadas a través de la misma, para fines docentes, productivos e investigativos. Los usuarios que están conectados en la red pueden acceder a la información publicada en internet, a través de las peticiones que son realizadas por el navegador web al servidor. El presente trabajo investigativo tiene como precedente la necesidad de acceder a todos los contenidos empotrados en los sitios web permitidos, según las políticas establecidas por la Dirección de Seguridad Informática de la institución, pero algunos de estos contenidos representan publicaciones referentes a sitios web a los que no está permitido el acceso. Para ello se obtiene una solución informática capaz de interceptar, modificar y procesar las peticiones de la navegación web, por lo que se obtiene una herramienta capaz de permitir que los usuarios puedan visualizar todos los contenidos empotrados en los sitios web confiables.

En la propuesta de solución se utiliza como metodología de desarrollo de software *Extreme Programming*. Para el desarrollo de la herramienta se utilizó como tecnologías, el servidor proxy Squid, JavaScript como lenguaje de programación, WebStorm como IDE de desarrollo y WebExtensions como sistema de desarrollo de complementos.

Palabras claves: contenidos empotrados, navegador web, peticiones, servidor, sitios web permitidos.

Índice de Contenidos

Índice de Figuras

Índice de Tablas

Introducción

La seguridad informática se orienta a la protección de la infraestructura computacional y todo lo relacionado con ésta (incluyendo la información contenida).(García *et al;* 2016) Así mismo, la seguridad de la información es un factor primordial en cualquier institución, donde la incorrecta manipulación de los datos puede ocasionar daños económicos y sociales considerables. Por tanto, se requiere que se realice una correcta protección de los recursos informáticos valiosos de la organización, tales como la información, el hardware y el software, a través de la adopción de políticas de seguridad que involucran las personas, los recursos y los procesos.

En el mundo conectado de hoy día, se depende del papel que la tecnología desempeña virtualmente en casi todos los aspectos de nuestra vida; desde las operaciones bancarias móviles, hasta la seguridad de los sistemas más críticos. Con el mayor uso de la tecnología también aumenta el volumen y sofisticación de las amenazas".(Organización de los Estados Americanos *et al*; 2014)

El internet constituye un avance de gran impacto para la sociedad, específicamente para los centros educativos, que hacen uso de este para permitirle a los usuarios el acceso a la información que se encuentra publicada en la red.

Cuba, a pesar de ser un país subdesarrollado, no ha quedado exenta del avance de la tecnología. En los últimos años uno de los principales objetivos del estado cubano ha sido el desarrollo de la industria del software, no solo con el fin de desarrollar sistemas para la informatización de la sociedad cubana, sino también con el objetivo de insertarse en el mercado internacional.(Cornelio *et al*; 2014)

En aras de impulsar este desarrollo, en septiembre del 2002 se creó la Universidad de las Ciencias Informáticas (en lo adelante UCI) como un programa de la Revolución, que tiene como tarea fundamental formar profesionales comprometidos con su Patria y altamente calificados en la rama de la Informática. Producir aplicaciones y servicios informáticos, a partir de la vinculación estudio-trabajo como modelo de formación. Servir de soporte a la industria cubana de la informática.

Algunos de los servicios brindados por la UCI son: correo electrónico, mensajería instantánea, multimedia, la navegación nacional y de internet. Este último brinda a los usuarios la posibilidad de acceder a sitios web para la búsqueda de información.

Introducción

En la UCI el proceso de navegación por internet se lleva a cabo a través de mecanismos que involucran herramientas que se encargan del filtrado del contenido web, para un mejor funcionamiento del mismo.

Cuando un usuario desde un navegador hace una petición, las misma es atendida por el programa Redirect[1], el cual consulta la cuota en la base de datos, si la petición es de los sitios que no necesita cuota, le permite la navegación al usuario. En el caso de que sí necesite cuota, la petición es capturada por el E2guardian[2], que verifica que la petición corresponda a un sitio que no tiene ninguna ponderación (sitio permitido), se lo envía al servidor Squid[3], y este se encarga de permitir el acceso. En caso contrario el E2guardian directamente le deniega el acceso al usuario.

En la UCI, según las políticas de seguridad informática, los sitios web se encuentran definidos por categorías (sitios confiables, sitios permitidos y sitios no permitidos). Los confiables son de especial interés para la institución por la información que contienen. Los permitidos y no permitidos son aquellos que su consulta está autorizada o no respectivamente, dependiendo de la información que brindan.

Actualmente, cuando un usuario accede a un sitio web confiable, el mismo puede contener contenidos empotrados[4] referentes a información de materiales educativos, de carácter científico-técnico, de ocio y entretenimiento, que pueden estar almacenados localmente en sitios web no permitidos. Sin embargo, no está autorizado el acceso a algunos de estos contenidos, debido a la forma en que se autorizan y deniegan las peticiones en el servidor Squid.

Por otra parte, actualmente, cuando se realizan las peticiones al servidor Squid, en el mismo es posible identificar cuál es el dominio principal de cada petición, para de esta manera permitirles a los usuarios acceder a todos los contenidos empotrados en los sitios web confiables.

Teniendo en cuenta la problemática anteriormente expuesta se propone como **problema de investigación**: ¿Cómo procesar las peticiones de la navegación web en la UCI para permitir a los usuarios acceder a todos los contenidos empotrados en los sitios web confiables?

[1] **Redirect:** herramienta desarrollada en la Universidad de las Ciencias Informáticas.
[2] **E2guardian:** herramienta para el filtrado de contenido web.
[3] **Squid:** servidor proxy-caché.
[4] **contenidos empotrados:** información que se encuentran adjunta dentro de los sitios web.

Por lo que el **objeto de estudio** de la investigación se enmarca en las herramientas que procesan las peticiones de la navegación web.

Teniendo en cuenta el problema de investigación se define como **objetivo general**: desarrollar una herramienta que permita procesar las peticiones de la navegación web en la UCI para permitir a los usuarios acceder a todos los contenidos empotrados en los sitios web confiables.

Definiéndose **campo de acción:** el procesamiento de las peticiones de la navegación web en la UCI.

Para dar cumplimiento al objetivo propuesto, se definen las siguientes **tareas de investigación:**

1. Análisis de las herramientas que permiten procesar las peticiones de la navegación web, para establecer similitudes con la investigación en curso.
2. Análisis de las herramientas y metodologías necesarias a utilizar en el desarrollo del sistema, para lograr una aplicación funcional como solución al problema de investigación.
3. Identificación de funcionalidades para detallar las características que debe cumplir la Herramienta para el procesamiento de peticiones de la navegación web.
4. Estudio de los tipos de prueba que propone la metodología escogida, para verificar la correcta implementación de la herramienta.

La realización del presente trabajo se apoya en el uso de los siguientes métodos de investigación:

Métodos Teóricos:

- ✓ Analítico-Sintético: Se utilizó este método para realizar el estudio teórico de la investigación, facilitando el análisis de la documentación referente a las herramientas que permiten procesar las peticiones de la navegación web y extraer de estas los elementos más importantes.
- ✓ Inductivo-Deductivo: Este método se empleó para inferir conocimientos lógicos, como las conclusiones parciales y generales a partir del estudio de la información resumida sobre el procesamiento de las peticiones de la navegación web.

Métodos Empíricos:

- ✓ Medición: Este método permite que la herramienta obtenida como resultado de la investigación pueda ser medida mediante las pruebas que propone la metodología escogida, para garantizar el correcto funcionamiento de la misma.
- ✓ Entrevista: Este método permite la comunicación con el personal especializado en el tema de la investigación, para definir los problemas existentes actualmente en la UCI, a los que se le dará solución posteriormente con la implementación de la solución propuesta.

Para una mejor comprensión del presente trabajo, se estructurará de manera coherente y organizada en tres capítulos, los cuales se describen a continuación.

Capítulo 1. "Fundamentación Teórica": Se abordan los elementos teóricos más importantes asociados a la investigación en curso. Incluye el estudio de estado del arte de las herramientas para el procesamiento de las peticiones de navegación web. Se explica la metodología que guiará el proceso de desarrollo de software. Mediante un análisis bibliográfico se determinan las tecnologías, herramientas y lenguajes de programación a utilizar para implementar la solución propuesta.

Capítulo 2. "Exploración, Planificación y Diseño": Se realiza una descripción del sistema propuesto. Se detallan los artefactos generados durante la fase de exploración, planificación y diseño que propone la metodología escogida, tales como: las historias de usuario (HU), el plan de iteraciones, el plan de entregas, la arquitectura, los patrones de diseño y las tarjetas Clase-Responsabilidad- Colaboración (CRC). Se describe el impacto social del presente trabajo.

Capítulo 3. "Implementación y Pruebas": Se describen los estándares de codificación utilizados en la implementación de la herramienta. Se realiza una descripción de los principales artefactos a generar por la metodología seleccionada, como son: las tareas de ingeniería por cada HU identificada y las pruebas empleadas para validar la solución propuesta.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En este capítulo se plasma una conceptualización de los elementos fundamentales para una mejor comprensión de la investigación en curso. Se realiza un proceso investigativo de los sistemas existentes, afines al procesamiento de las peticiones de la navegación web, para evaluar sus ventajas e insuficiencias. Se describen las herramientas, tecnologías y metodologías a utilizar para el desarrollo del sistema.

1.2 Conceptos fundamentales

1.2.1 Petición

Una petición es un requerimiento o solicitud que le hace un cliente a un servidor. El servidor web es el que se encarga de mantener *online* (en línea) un sitio web. Un cliente (el visitante a través de su navegador) solicita que dicho servidor le dé una página web determinada. Esa solicitud es una petición HTTP (*Hypertext Transfer Protocol*). (Leandro Alegsa, 2010)

1.2.2 Navegador web

Un navegador web es una aplicación de software que permite a los usuarios de internet acceder, navegar y buscar información, servicios o productos a nivel mundial. Los navegadores web interpretan enlaces de hipertexto que permiten leer documentos formateados en HTML (*Hyper Text Markup Language*), JavaScript y AJAX de tal manera que puedan ser vistos en la pantalla del computador.(Cavsi, 2007)

1.2.3 Complemento web

Los complementos para los navegadores añaden funciones que facilitan el desarrollo de las páginas web, como consultar las propiedades CSS (Hoja de Estilo en Cascada) que tiene un elemento de la página web, o que permiten la utilización de algún servicio en internet, como la validación del código HTML o CSS de una página, con una única pulsación del ratón.(Sergio Luján Mora, 2017)

1.2.4 Servidor web

Servidor que se dedica a prestar servicios relacionados a la WWW (*World Wide Web*), para que un sitio web esté disponible en internet.(Leandro Alegsa, 2010)

1.2.5 Contenidos empotrados

Los contenidos empotrados representan la información que se encuentran adjunta dentro de los sitios web.(Lionel Pairuna, 2017) Por ejemplo, los videos de la página de YouTube que se encuentran agregados en la página de CubaDebate.

1.2.6 Sitio web

Un sitio web es una estructura de información y/o comunicación generada en el nuevo ámbito o espacio de comunicación (Internet), creado por la aplicación de las tecnologías de la información (tecnologías de creación, mantenimiento y desarrollo de los sitios web), que posee dos elementos fundamentales (acciones de los sujetos y contenidos) y en donde se plantean un conjunto de prestaciones que los usuarios que visitan dicho web pueden ejercitar para satisfacer una o varias necesidades que posean.(Jaime Alonso, 2008)

1.3 Herramientas para el procesamiento de peticiones de navegación web.

1.3.1 Lightbeam para Firefox

Firefox es un navegador web libre y de código abierto, en cuyo desarrollo puede colaborar cualquier usuario que lo desee. Es descendiente de Mozilla *Application Suite* y es desarrollado por la Fundación Mozilla.

Lightbeam es un complemento para Firefox que, a través de visualizaciones interactivas, muestra los sitios de origen y de terceros que interactúan con el usuario en la Web. Mientras se navega, Lightbeam muestra la web actual en toda su dimensión, incluyendo aquellas partes no tan claras para el usuario medio. (Mozilla, 2016)

Crea un registro de eventos para cada sitio que se visita y cada sitio de terceros que esté almacenado localmente en el navegador, también muestra gráficos visuales de estos eventos para resaltar las interacciones entre los sitios que se visitan a propósito y los de terceros. Lightbeam continuará añadiendo información al gráfico mientras se esté navegando por la web. Se puede detener en cualquier momento desactivando el complemento o desinstalándolo. (Mozilla, 2016)

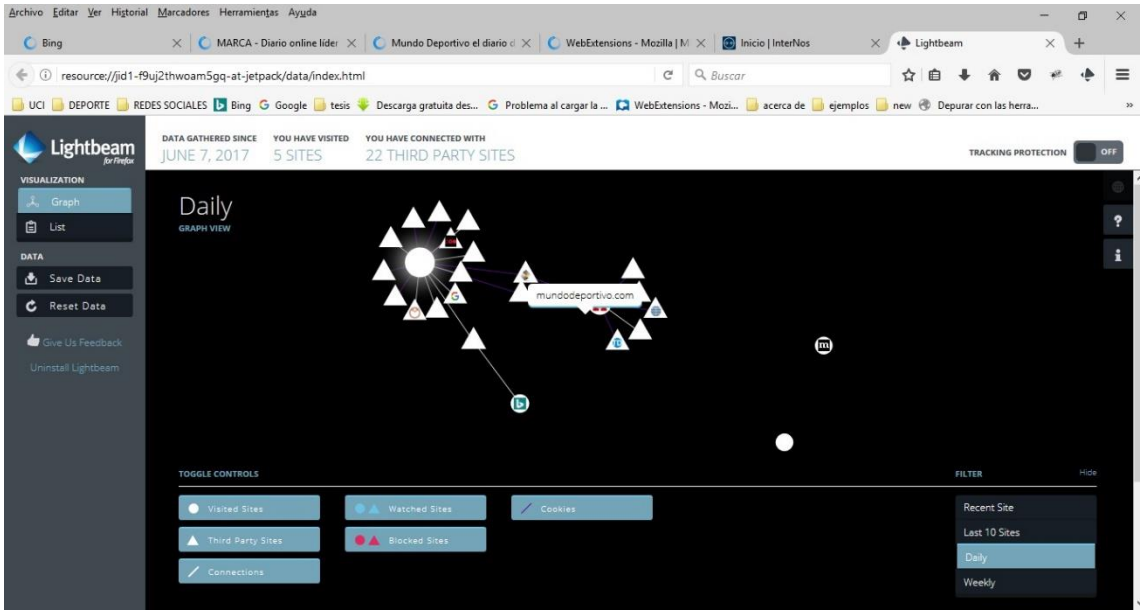


Figura 1. Representación gráfica de los sitios de origen y de terceros en Ligthbeam

Lightbeam es la continuación de Collusion, un complemento experimental, que ahora incluye nuevas características y una interfaz renovada. Dispone de tres presentaciones principales:

- **la vista de tráfico**, que muestra la información de los sitios que se han visitado y sus conexiones
- **la vista de reloj**, que muestra la actividad de las últimas 24 horas en un gráfico circular que permite identificar patrones de comportamiento por horas
- **el listado**, que permite profundizar más en la información de cada sitio.

El principal inconveniente para el uso de esta herramienta, es que está diseñada para un único navegador web (Firefox) y no permite la comunicación con un servidor para permitir o denegar determinados dominios.

1.3.2 Smart Keeper

Smart Keeper es un filtro de contenido web que permite aplicar la política de uso aceptable de internet de una institución para regular el acceso de los usuarios a la red de redes. Está basado en el uso e integración del servidor proxy cache Squid, el servidor web Apache2, el servidor de base de datos PostgreSQL y otros componentes libres(«Manual de usuario Smart Keeper», 2014).

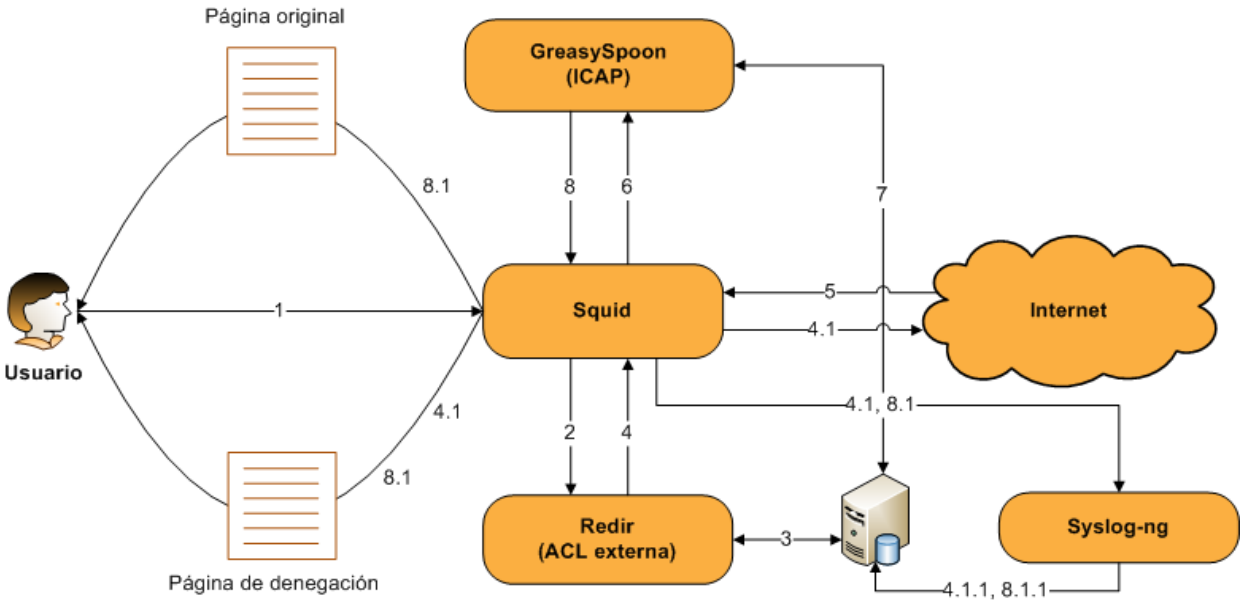


Figura 2. Funcionamiento de Smart Keeper

1. Squid recibe la petición realizada por el usuario.
2. Squid envía la petición al programa Redir.
3. Redir consulta en la base de datos del sistema las configuraciones que afectan la navegación del usuario y decide si ha de permitir o denegar la petición.
4. Redir comunica a Squid el resultado de la decisión. Si la respuesta es:

4.1 Denegar, Squid reenvía al usuario a la Página de Denegación terminando el proceso.

4.1 Permitir, Squid puede obtener la respuesta de la petición.

5. Squid obtiene de Internet (o la caché) la respuesta de la petición.
6. Squid envía la respuesta de la petición al servidor ICAP (protocolo de adaptación de contenidos de internet) GreasySpoon.
7. GreasySpoon consulta en la base de datos del sistema las configuraciones que afectan la navegación del usuario y decide si ha de permitir o denegar la respuesta a la petición.
8. GreasySpoon comunica a Squid el resultado de la decisión. Si la respuesta es:

8.1 Denegar, Squid reenvía al usuario a la Página de Denegación terminando el proceso.

8.1 Permitir, Squid entrega al usuario la respuesta de la petición terminando el proceso.

El principal inconveniente para el uso de esta herramienta es que de la forma en que llegan las peticiones al servidor, no es posible conocer los sitios de origen y de terceros a los que accedió un usuario respectivamente, esto trae consigo que el proceso de

aceptar o denegar las peticiones en el servidor, no se realice de la forma deseada según la problemática planteada.

Smart Keeper es usado en uno de los centros de producción de la UCI, pero en el Nodo Central no se utiliza porque es considerado una mala personalización del servidor Squid, ya que las mismas reglas que están definidas en Squid, se pueden administrar en el Smart Keeper a través de una interfaz que no resulta muy amigable para los administradores de red.

1.3.3 E2guardian

E2guardian es un filtro de contenido web de código abierto, que filtra el contenido real de las páginas basadas en muchos métodos, incluyendo la coincidencia de frases, el encabezado de la solicitud y el filtrado de URL (*Uniform Resource Locator*), entre otros. (Frederic Bourgeois, 2013)

Características de E2Guardian:

- Se puede configurar de tal manera que es posible tener varias configuraciones de filtro, para proporcionar grados de filtrado web a diferentes grupos de usuario.
- Soporte de autenticación de resumen básica, de IP (*Internet Protocol*) y de DNS.
- Análisis y manipulación de cabeceras.
- Soporte para descarga y escaneado de archivos grandes (más de 2 GB).
- Dominios y URL de la lista blanca y de la negra.
- Es posible negar expresiones en URL, contenido corporal y encabezados.
- Reemplazo de la expresión regular de URL para que sea posible forzar la búsqueda segura en los motores de búsqueda.
- Exploración profunda de URL para detectar URL en URL, es; decir, bloquear imágenes de Google.
- Bloqueo avanzado de anuncios.
- Bloqueo SSL (*Secure Sockets Layer*), para denegar proxys anónimos SSL.

E2guardian tiene un mecanismo de filtrado que tiene una serie de ficheros, donde se le pueden definir reglas, ya sean estáticas o por expresiones regulares, esta última es la más común.

El funcionamiento del mecanismo que realiza para el filtrado del contenido web, está dado a partir de que se recibe la petición y la filtra, luego aparecen los mensajes de denegación definiendo por qué no se puede acceder a la página o la información de la página solicitada.

Actualmente esta herramienta es utilizada en la UCI por los especialistas de Seguridad Informática y su código está disponible en GitHub. No se va a utilizar en la solución propuesta, porque es más factible que las reglas de filtrado se implementen directamente en el servidor Squid.

1.3.4 Dansguardian

Dansguardian es un filtro de contenido web Open Source que actualmente se ejecuta en Linux, FreeBSD, OpenBSD, NetBSD, Mac OS X, HP-UX y Solaris. Filtra el contenido real de las páginas basándose en muchos métodos, como la concordancia de frases, el filtrado de fotos y el filtrado de URL. No filtra puramente en una lista prohibida de sitios. (Daniel Barron, 2011)
No se va a utilizar en la solución propuesta, porque es más factible que las reglas de filtrado se implementen directamente en el servidor Squid.

Luego del análisis de todas las herramientas mencionadas anteriormente, es posible arribar a las siguientes conclusiones:

- Lightbeam no es posible utilizarlo en la solución propuesta porque no permite la comunicación con un servidor para permitir o denegar determinados dominios. Sin embargo, sirvió de base en el desarrollo de un complemento para interceptar las peticiones y modificarlas.
- Smart Keeper no permite conocer los sitios de origen y de terceros a los que accedió un usuario respectivamente, esto trae consigo que el proceso de aceptar o denegar las peticiones en el servidor, no se realice de la forma deseada según la problemática planteada.
- Smart Keeper es usado en uno de los centros de producción de la UCI, pero en el Nodo Central no se utiliza porque es considerado una mala personalización del servidor Squid, ya que las mismas reglas que están definidas en Squid, se pueden administrar en el Smart Keeper a través de una interfaz que no resulta muy amigable para los administradores de red.
- E2guardian y Dansguardian no se va a utilizar en la solución propuesta, porque es más factible que las reglas de filtrado se implementen directamente en el servidor Squid.

1.4 Metodología de desarrollo de software

Las tendencias modernas en el desarrollo de software apuntan hacia el uso de metodologías más flexibles con un enfoque simple, donde el cliente está presente en todo el proceso de avance, estas son las metodologías ágiles.

Principios de las metodologías ágiles. (Ian Sommerville, 2009)

- ✓ Participación del cliente: Los clientes deben estar fuertemente implicados en todo el proceso de desarrollo. Su papel es proporcionar y priorizar nuevos requerimientos del sistema y evaluar las iteraciones del sistema.
- ✓ Entrega incremental: El software se desarrolla en incrementos, donde el cliente especifica los requerimientos a incluir en cada incremento.
- ✓ Personas, no procesos: Se deben reconocer y explotar las habilidades del equipo de desarrollo. Se les debe dejar desarrollar sus propias formas de trabajar, sin procesos formales, a los miembros del equipo.
- ✓ Aceptar el cambio: Contar con que los requerimientos del sistema cambian, por lo que el sistema se diseña para dar cabida a estos cambios.
- ✓ Mantener la simplicidad: Se deben centrar en la simplicidad tanto en el software a desarrollar como en el proceso de desarrollo. Donde sea posible, se trabaja activamente para eliminar la complejidad del sistema.

La Programación Extrema (XP, por sus siglas en inglés), es una metodología de desarrollo de software ágil. El nombre fue acuñado por Kent Beck debido a que el enfoque fue desarrollado utilizando buenas prácticas reconocidas, como el desarrollo iterativo y con la participación del cliente. (Ian Sommerville, 2009)

Se decidió utilizar XP debido a que se adapta al tipo de proyecto a desarrollar, las condiciones de trabajo y las prácticas utilizadas. A continuación, se explican las razones de la selección de esta metodología:

- El proyecto es pequeño (2 personas). Unas de las características de XP, es que está ideada para equipos pequeños y muy integrados (2-12 personas).
- Los requisitos del sistema pueden cambiar. Uno de los principios básicos de XP, es que es capaz de adaptarse a los cambios de requisitos.
- El cliente forma parte del equipo de desarrollo. La aplicación de XP está orientada a quien produce y usa software, por lo que el cliente participa muy activamente.
- El riesgo de desarrollo es elevado debido al corto tiempo de entrega planteado y a los continuos cambios de requerimientos. XP está diseñada para mitigar los riesgos en proyectos con estas características.
- Se basa en que el funcionamiento del software es más importante que la documentación exhaustiva.

1.5 Tecnologías y herramientas de desarrollo

Los programas de software usan muchos lenguajes y tecnologías diferentes, con las herramientas típicamente creadas para tecnologías específicas. El desarrollo de software puede ser una actividad compleja y larga, por lo que las herramientas disponibles pueden reducir el estrés y aumentar el desempeño tanto de desarrolladores como de las aplicaciones resultantes.(Sue Smith, 2016)

A continuación, se describen las tecnologías y herramientas seleccionadas para el desarrollo del presente trabajo.

1.5.1 Tecnologías y herramientas del lado del servidor

1.5.1.1 Servidor Squid

Squid es uno de los aceleradores de contenido más antiguos, utilizado por miles de sitios web de todo el mundo para facilitar la carga en sus servidores. El contenido frecuentemente visto es almacenado en caché por Squid y se sirve al cliente final con sólo una fracción de la carga del servidor de aplicaciones que se necesita normalmente.(Dawson *et al;* 2013)

Squid es un servidor web proxy-caché con licencia GPL (*General Public License*), cuyo objetivo es funcionar como proxy de la red y también como zona caché para almacenar páginas web.(Dawson *et al;* 2013)

Este servidor permite la configuración de las Listas de Control de Acceso (ACL, por sus siglas en inglés), para restringir el acceso a los contenidos definidos y sirve como zona de caché para acelerar el acceso a las páginas web, es decir; de la navegación en internet. Squid también funciona como caché, significa que está guardando copia de los datos obtenidos de otras peticiones y de esa forma acelera el acceso a estos datos si se producen peticiones similares. Sólo se accederá de nuevo a las páginas originales cuando se detecte que se han producido modificaciones.(Dawson *et al;* 2013)

1.5.2 Tecnologías y herramientas del lado del cliente.

1.5.2.1 JavaScript

JavaScript es un lenguaje de *scripting* multiplataforma y orientado a objetos. Es un lenguaje pequeño y liviano. Dentro de un ambiente de host, JavaScript puede conectarse a los objetos de su ambiente y proporcionar control programático sobre ellos.(Mozilla Developer Network, 2016)

JavaScript contiene una librería estándar de objetos, tales como *Array*, *Date*, y *Math*, y un conjunto central de elementos del lenguaje, tales como operadores, estructuras de

control, y sentencias. El núcleo de JavaScript puede extenderse para varios propósitos, complementándolo con objetos adicionales.(Mozilla Developer Network, 2016)

1.5.2.2 WebExtensions

WebExtensions es un sistema para desarrollar complementos de navegador. En gran medida, el sistema es compatible con la Interfaz de Programación de Aplicaciones (API, por sus siglas en inglés) de extensión admitida por Google, Chrome y Opera. Las extensiones escritas para estos navegadores se ejecutan en la mayoría de los casos en Firefox o Microsoft Edge con sólo unos pocos cambios. La API también es totalmente compatible con multiprocesos Firefox.(Mozilla Developer Network, 2017)

1.5.2.3 WebStorm 2016.3.4

WebStom es un IDE (***I****ntegrated Development Environment*) de desarrollo que proporciona soporte de primera clase para JavaScript, Node.js, HTML y CSS, así como para sus sucesores modernos. Sirve como ayuda para escribir mejor el código gracias a la finalización del código inteligente, detección de errores en marcha, potente navegación y refactorización. La integración con herramientas que hacen posible minimizar el uso de la línea de comandos. Utiliza un depurador de gran alcance para JavaScript y Node.js.(JetBrains, 2017)

1.6 Conclusiones del capítulo

El estudio de los referentes teóricos permitió el entendimiento de los conceptos esenciales relacionados con las peticiones de navegación web. Además, se realizó un estudio de las herramientas afines existentes a nivel nacional e internacional. Esto permitió llegar a la conclusión de que existen características específicas de cada herramienta, que dificultan su empleo en el desarrollo de la solución propuesta, pero también algunas facilitan el entendimiento del funcionamiento del sistema para la futura implementación del mismo. La metodología de software seleccionada permitió guiar el proceso de desarrollo de la aplicación. Las tecnologías y herramientas de desarrollo, propuestas para la solución del problema de investigación, facilitarán la gestión de los datos y la construcción de una aplicación de forma rápida que sea entendible y sencilla para el usuario final.

Capítulo 2: Exploración, Planificación y Diseño

2.1 Introducción

En el presente capítulo se realiza una descripción del sistema propuesto. Se definen artefactos generados por la metodología XP dentro de las fases de Exploración, Planificación y Diseño, tales como: las HU, el plan de iteraciones, el plan de entregas, la arquitectura, los patrones de diseño y las tarjetas CRC.

2.2 Descripción del sistema propuesto

Se propone como solución informática el desarrollo de un complemento para el navegador web Firefox, utilizando WebExtensions. Este permite interceptar las peticiones enviadas desde el lado cliente y una vez obtenidas se realiza una modificación de las mismas, específicamente en sus cabeceras.

Se realiza una identificación de peticiones por pestañas. Una vez que se identifica que la petición en cuestión es de una pestaña correspondiente, se procede a modificar la cabecera de esta petición, adicionando un nuevo campo denominado **dominio principal** y modificando el *referer*[5]. Como resultado se obtiene un conjunto de peticiones modificadas, las cuales contienen en el campo **dominio principal** y en el *referer* la URL que dio inicio a esta petición, junto con los demás campos correspondientes a cada petición.

El navegador con el complemento integrado, le envía la petición al servidor Squid. Luego el mismo se encarga de analizar la cabecera de la petición enviada y determina si permitir o denegar la petición basándose en el campo referer que contiene la URL que dio inicio a la petición. Los dominios permitidos y no permitidos están configurados localmente en el servidor.

Luego de este análisis, el servidor Squid le envía la respuesta de la petición al navegador y el cliente podrá tener acceso a todos los contenidos empotrados en los sitios web confiables. Estos contenidos se encuentran localmente almacenados en sitios web no permitidos.

A continuación, se muestran tres capturas de la solución propuesta:

[5] **referer:** cabecera que contiene la dirección de la página web anterior desde la cual se solicitó la página actual.

- Ejemplo de la visualización de los contenidos empotrados en un sitio web confiable:

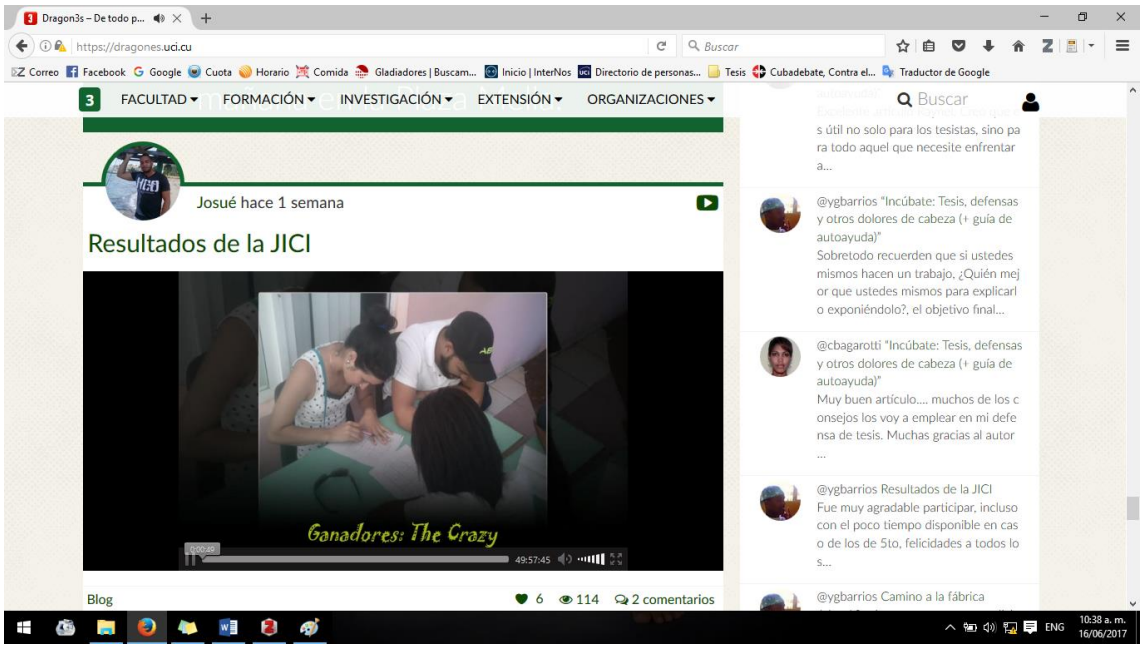


Figura 3. Sitio web Dragones

- Ejemplo de denegación de contenidos empotrados en un sitio web permitido:

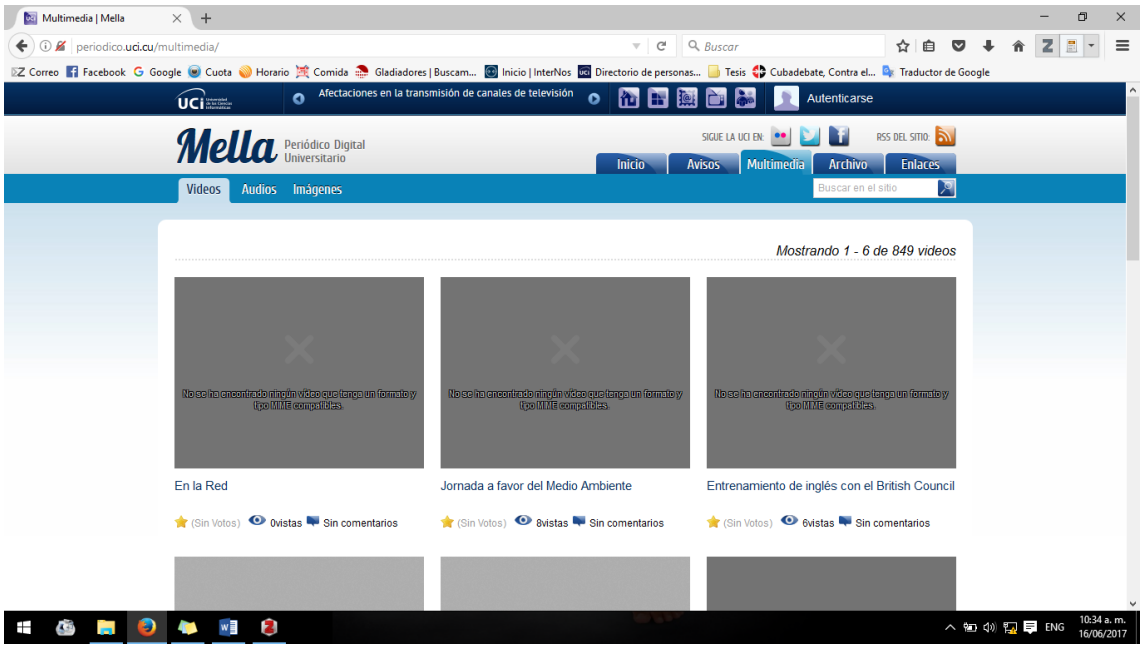


Figura 4. Sitio web Periódico

- Ejemplo de denegación de sitio no permitido:

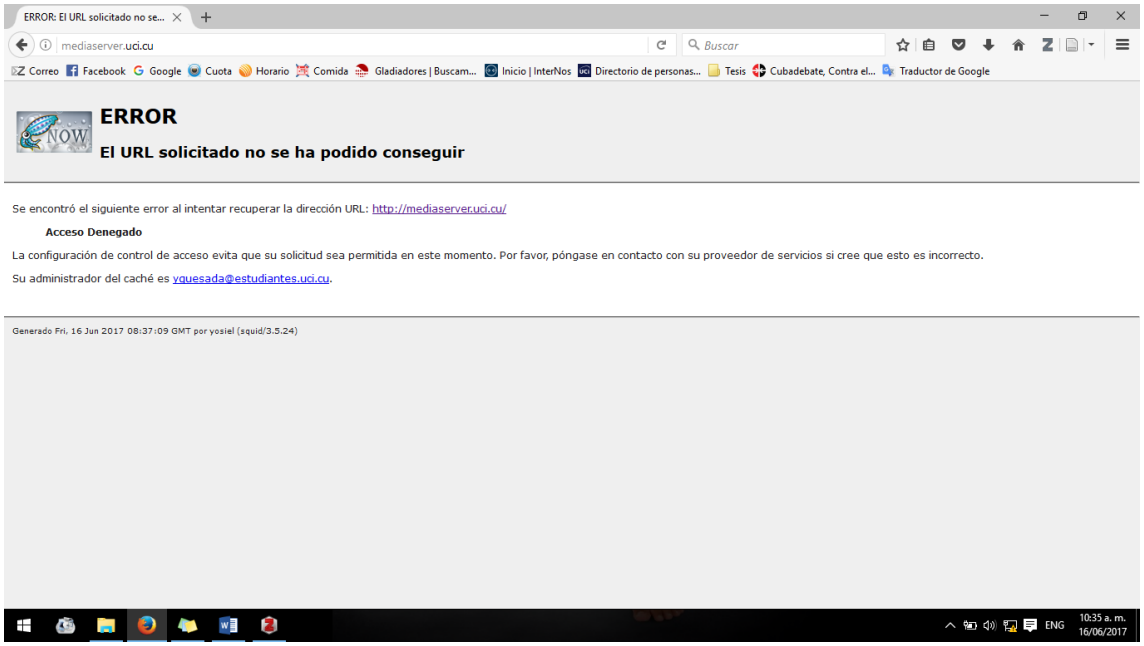


Figura 5 Sitio web Media server

Además, se añade el campo dominio principal de cada petición al registro (log) generado por el servidor proxy, para posteriormente poder organizar las trazas jerárquicamente en otras herramientas.

A continuación, se representa el funcionamiento del sistema propuesto:

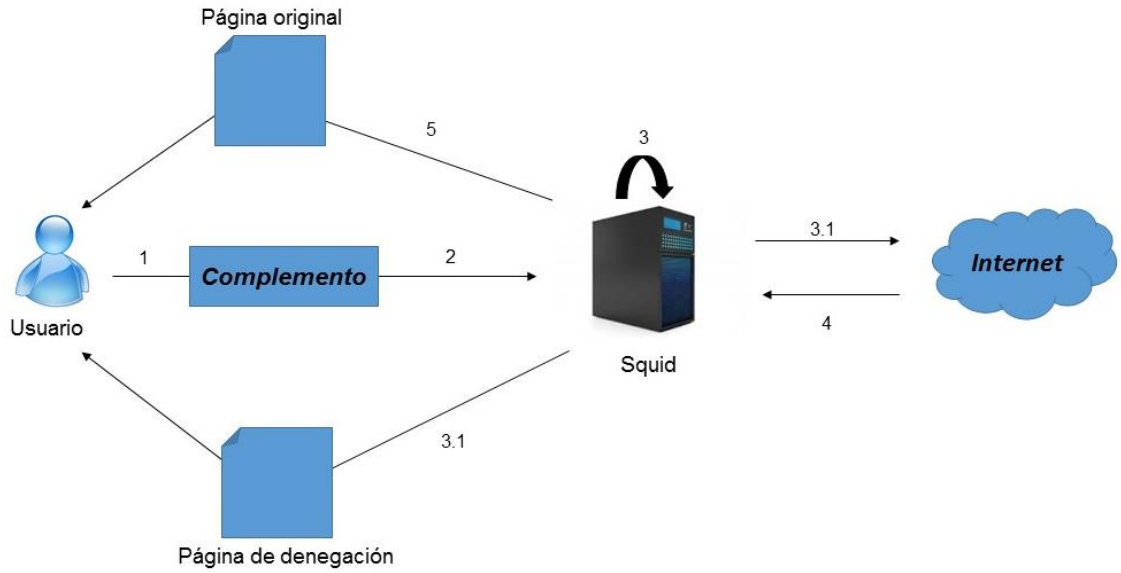


Figura 6. Funcionamiento del sistema propuesto

1- El complemento intercepta la petición realizada por el usuario.

2- El complemento envía la petición modificada al servidor Squid.

3- Squid analiza la petición. Si la respuesta es:

3.1 Permitir, Squid envía a Internet la petición.

3.1 Denegar, Squid reenvía al usuario la Página de Denegación terminando el proceso.

4- Squid obtiene de Internet (o la caché) la respuesta de la petición.

5- Squid entrega al usuario la respuesta de la petición terminando el proceso.

La carga de una página web comprende decenas de peticiones al servidor y respuestas de este cuyo esquema básico es el siguiente:

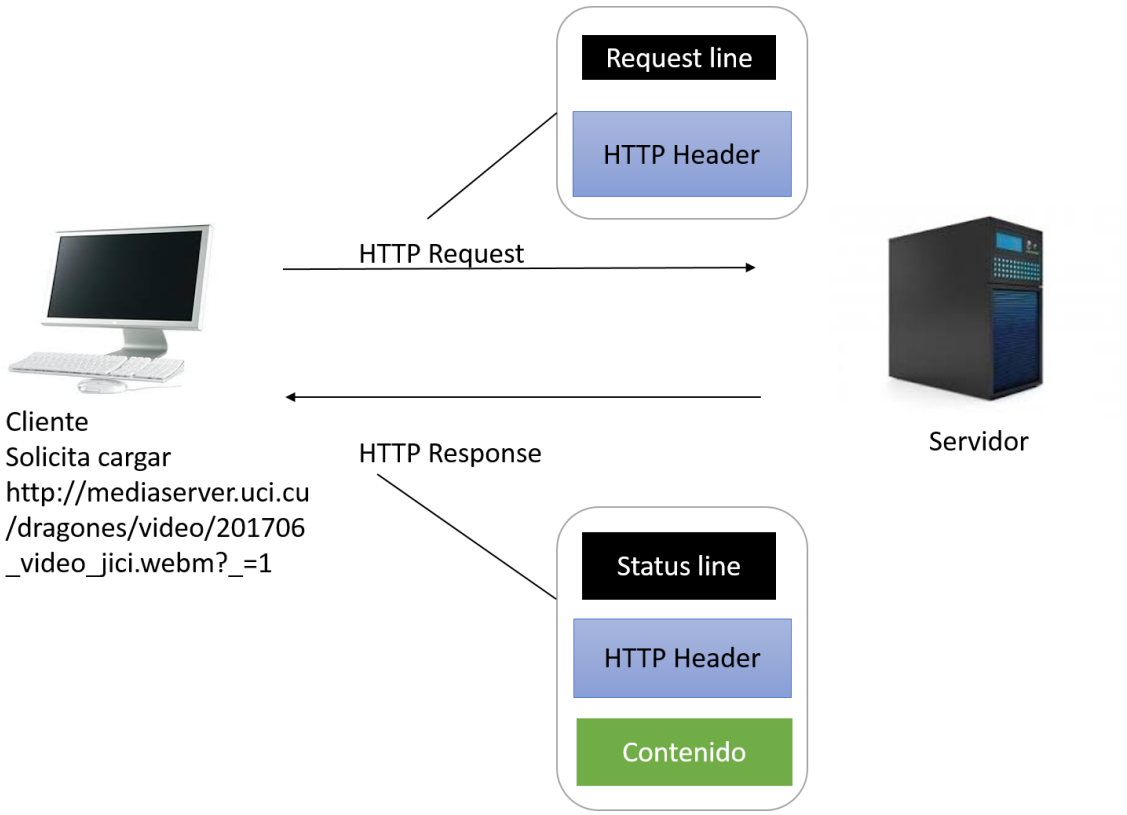


Figura 7. Esquema protocolo HTTP

2.2.1 Cabecera de peticiones (*Request*)

La petición consta de(Anónimo, 2017):

- *Request line* (línea de petición): contiene información básica de la petición. Está formada por tres campos (método, ruta y protocolo).

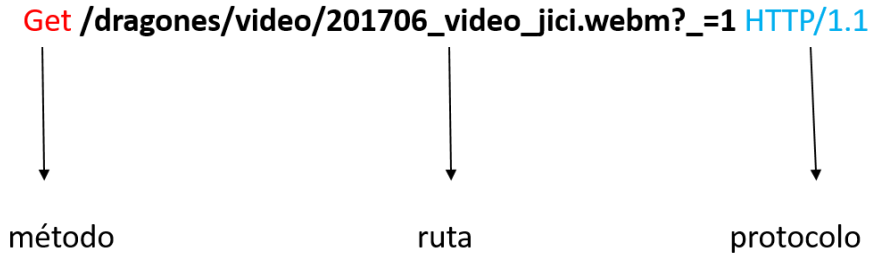


Figura 8. Request line

- *Header* (cabecera): está compuesta por los campos que se describen a continuación:
  - ✓ *Host:* dirección a la que se quiere acceder.
  - ✓ User-Agent: información sobre el navegador.
  - ✓ *Accept:* indica que los tipos de medios son aceptables para la respuesta.
  - ✓ *Accept-Language:* idiomas que el cliente prefiere para la respuesta.
  - ✓ Accept-Encoding: mecanismo de codificación que se ha aplicado al cuerpo del mensaje y, por lo tanto, qué mecanismo de decodificación debe utilizarse para obtener la información.
  - ✓ *Cookie:* contiene pequeñas cantidades de datos que se envían entre un emisor y un receptor. En el caso de Internet el emisor sería el servidor donde está alojada la página web y el receptor es el navegador que se utiliza para visitar cualquier página web.
  - ✓ *Referer:* permite al cliente especificar la dirección de los recursos desde la que obtiene la respuesta.
  - ✓ *Range:* indica la parte del archivo que el servidor debe devolver.
  - ✓ *Connection:* controla si la conexión de red permanece abierta después de que finaliza la transacción actual.

A continuación, se muestra un ejemplo de los campos de la cabecera de una petición luego de ser modificada por el complemento:

- ✓ *Host:* “mediaserver.uci.cu” (dirección a la que se quiere acceder)
- ✓ User-Agent: “Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:51.0) Gecko/20100101 Firefox/51.0”
- ✓ *Accept:* “video/webm,video/ogg,video/*;q=0.9,application/ogg;q=0.7,audio/*;q=0.6,*/*;q=0.5”
- ✓ *Accept-Language:* “es-ES,es;q=0.8,en-US;q=0.5,en;q=0.3”

- ✓ *Accept*-Encoding
- ✓ *Cookie:* “ga=GA1.2.218592864.1493748597”
- ✓ *Referer:* “dragones.uci.cu”
- ✓ *Range:* “bytes=2938916”
- ✓ *Connection:* “keep-alive”
- ✓ Dominio principal: “dragones.uci.cu”

2.2.2 Parámetros de configuración en el servidor Squid

Configuración de las ACL para permitir el acceso a la web:

- Creación de las ACL para el permitir el acceso a la web:
  acl [nombre_lista] src[6] [componentes_lista] (sintaxis de una ACL src)
  acl localnet src 10.8.16.46 (servidor)
  acl localnet src 10.8.16.107 (cliente)
- Creación de las reglas de control de acceso para permitir el acceso a la web:
  http_access [deny / allow] [lista_control_acceso] (sintaxis de la regla de control de acceso que se le aplica a las ACL)
  http_access allow localnet (permite al cliente el acceso a la web)

Configuración de las ACL para controlar el acceso a la web:

- Creación de las ACL para controlar el acceso a la web:
  acl [nombre_lista] dstdomain [componentes_lista] (sintaxis de una ACL dstdomain)
  acl denegada dstdomain mediaserver.uci.cu (se define el sitio no permitido)
  acl myreferer referer_regex -i gladiadores.uci.cu dragones.uci.cu (se define por donde filtrar el acceso a los sitios web confiables)
- Creación de las reglas para controlar el acceso a la web:
  http_access allow myreferer (permite al cliente el acceso a los sitios confiables)
  http_access deny denegada (deniega al cliente el acceso al sitio no permitido)

Configuración del puerto por el que atiende las peticiones:

http_port 10.8.16.46:3128 (proxy)
http_port 10.8.16.46:8080 (puerto)

Configuración del registro log:

[6] **src:** tipo de ACL.

cache_access_log /var/log/squid/access.log (dirección donde se encuentra almacenado el registro *log*)

logformat squidmime %ts.%03tu %6tr %>a %Ss/%03Hs %<st %rm %ru %un %Sh/%<A %mt [%>h] [%<h] (formato del *log*)

2.2.3 Configuración del navegador web Firefox

A continuación, se muestran los pasos a seguir para la configuración del proxy en el navegador web Firefox:

Paso #1: Herramientas -> Opciones

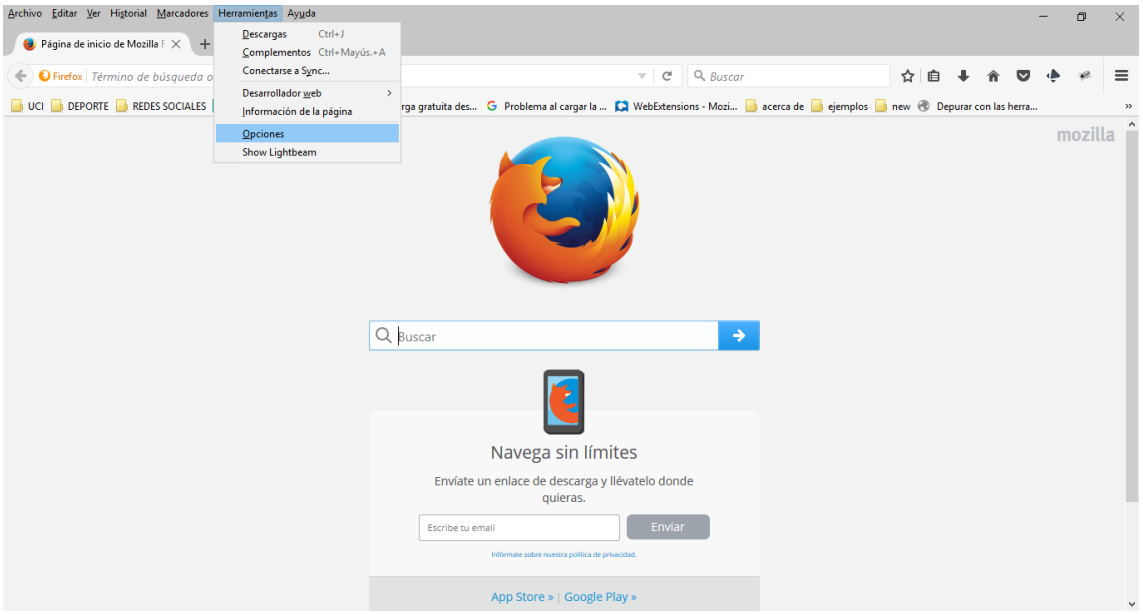


Figura 9. Paso #1

Paso #2: Avanzado -> Red -> Configuración

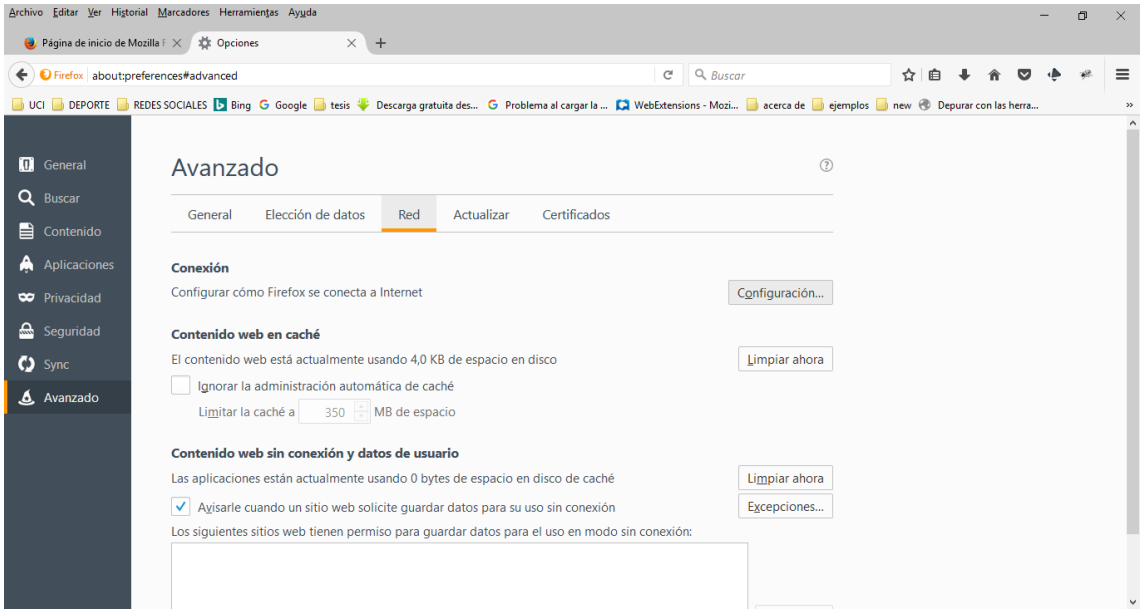


Figura 10. Paso #2

Paso #3: configuración manual del proxy

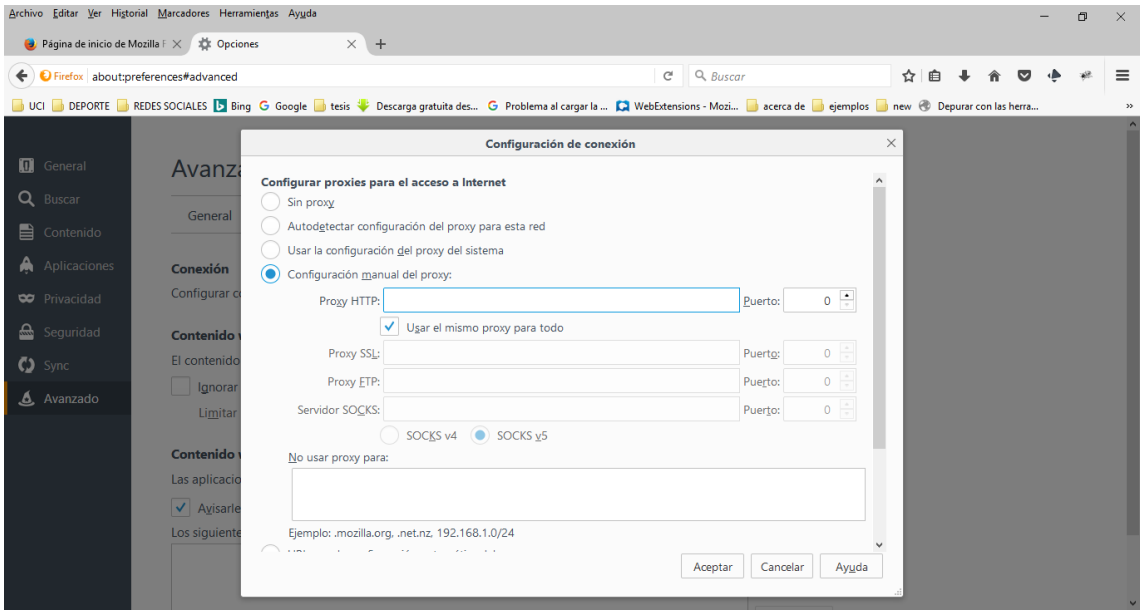


Figura 11. Paso #3

2.3 Características no funcionales del sistema

Las características no funcionales del sistema son restricciones de los servicios o funciones ofrecidos por el sistema. Incluyen restricciones de tiempo, sobre el proceso de desarrollo y estándares. Los requerimientos no funcionales a menudo se aplican al sistema en su totalidad. Normalmente a penas se aplican a características o servicios individuales del sistema. (Ian Sommerville, 2009)

2.3.1 Usabilidad

- El sistema debe proporcionar mensajes de error que propicien información para el usuario final.

2.3.2 Software

- Para el funcionamiento de la herramienta, las máquinas clientes deben tener instalado el complemento, que funciona en navegadores web Firefox, etc.
- Para el funcionamiento de la herramienta debe estar configurado el navegador para hacer las peticiones al servidor Squid previamente configurado.

2.3.3 Soporte

- La versión del navegador web Firefox, debe ser a partir de la 45 para poder instalar el complemento.

2.4 Exploración

La metodología de desarrollo XP comienza con la fase de Exploración. Durante esta se realiza el proceso de identificación de las HU, así como la familiarización de los equipos de trabajo con las tecnologías y herramientas seleccionadas para la construcción del proyecto.(Beck *et al*; 2005)

2.4.1 Historias de Usuario (HU)

Las HU son la forma en que se especifican en XP los requisitos del sistema. Estas se escriben desde la perspectiva del cliente, aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas.(Beck *et al;* 2005) Durante la fase de exploración se identificaron las siguientes HU:

- ✓ Interceptar y modificar peticiones
- ✓ Configurar servidor Squid
- ✓ Añadir dominio principal al registro log

Las HU del presente proyecto fueron elaboradas siguiendo el ejemplo que presenta Kent Beck en su libro “Extreme Programing Explained”, quedando definido a continuación algunos de los elementos a los que se hace referencia:

# de la Historia de Usuario	
Nombre	
Prioridad(alta/media/baja)	Riesgo en desarrollo(alto/medio/bajo)
Iteración asignada	Puntos estimados
Programador responsable	
Descripción	

Observaciones

Tabla 1. Formato de HU

Los parámetros definidos en cada HU se describen a continuación:

- ✓ Nombre: para identificar la HU.
- ✓ Prioridad:
  - Alta: para las funcionalidades principales del sistema, o que forman parte esencial de la arquitectura del mismo.
  - Media: para las funcionalidades de importancia y con valor para el usuario, pero que no se consideran imprescindibles para el funcionamiento del sistema.
  - Baja: para las funcionalidades que sería deseable tener y estas podrían incluirse en caso de que hubiese recursos para ello.
- ✓ Riesgo en desarrollo:
  - Alto: en caso de presentar algún error de implementación, pueden traer consigo la inoperatividad del sistema.
  - Medio: en caso de presentar errores retrasan la entrega de la versión.
  - Bajo: en caso de presentar errores estos son tratados con facilidad, sin que traigan consigo prejuicios para el desarrollo del proyecto.
- ✓ Iteración asignada: número de la iteración a la que ha sido asignada.
- ✓ Puntos estimados: tiempo estimado de desarrollo para completar las HU. Un punto estimado equivale a una semana de trabajo (5 días laborables).
- ✓ Programador responsable: programador encargado de desarrollar la HU.
- ✓ Descripción: es donde se realiza una breve descripción de la funcionalidad que se quiere implementar.
- ✓ Observaciones: detalles importantes a tener en cuenta para desarrollar la HU correctamente, es; decir, señalamientos o advertencias del sistema.

A continuación, se muestran las HU definidas por el cliente en conjunto con el equipo de desarrollo, para la investigación en curso:

Historia de Usuario # 1	
Nombre: Interceptar y modificar peticiones	
Prioridad: alta	Riesgo en desarrollo: medio
Iteración asignada: 1	Puntos estimados: 3
Programador responsable: Yosiel Quesada Fonseca y Yuliet Hernández Gainza	

Descripción: se captura la petición realizada por el cliente y luego se modifica la cabecera de la misma para añadir el dominio principal (URL en la barra de direcciones) a la que pertenece la petición.
Observaciones

Tabla 2. HU: Interceptar y Modificar peticiones

Historia de Usuario # 2	
Nombre: Configurar servidor Squid	
Prioridad: alta	Riesgo en desarrollo: medio
Iteración asignada: 2	Puntos estimados: 3
Programador responsable: Yosiel Quesada Fonseca y Yuliet Hernández Gainza	
Descripción: se realiza el proceso de configuración del servidor Squid, a partir de sus principales funcionalidades.	
Observaciones:	

Tabla 3. HU: Configurar servidor Squid

Historia de Usuario # 3	
Nombre: Añadir dominio principal al registro log	
Prioridad: media	Riesgo en desarrollo: medio
Iteración asignada: 3	Puntos estimados: 2
Programador responsable: Yosiel Quesada Fonseca y Yuliet Hernández Gainza	
Descripción: se añade el dominio principal de cada petición al registro (log) generado por el servidor proxy Squid.	
Observaciones	

Tabla 4. HU: Añadir dominio principal al registro log

2.5 Planificación

Durante la fase de Planificación se realiza una estimación del esfuerzo que costará implementar cada HU. Este se expresa utilizado como medida el punto. Un punto se considera como una semana ideal de trabajo, donde los miembros de los equipos de desarrollo trabajan el tiempo planeado sin ningún tipo de interrupción. Esta estimación incluye todo el esfuerzo asociado a la implementación de la HU, que incluye: las pruebas unitarias, la integración y refactorización del código, y la preparación y ejecución de las pruebas de aceptación.(Beck *et al;* 2005)

2.5.1 Estimación de esfuerzo por HU

Para el desarrollo de la herramienta propuesta se generó una estimación de esfuerzo por cada HU definida, arribando a los resultados que se muestran a continuación:

Historias de Usuario	Puntos Estimados
Interceptar y Modificar peticiones	3
Configurar servidor Squid	3
Añadir dominio principal al registro log	2

Tabla 5. Estimación de esfuerzo por HU

2.5.2 Plan de Iteraciones

Una vez identificadas y descritas las HU y estimado el esfuerzo dedicado a la realización de cada una de ellas, se procede a la planificación de la etapa de implementación del proyecto. Se decide realizar 3 iteraciones, las cuales se describen a continuación.

- ✓ **Iteración 1:** durante esta iteración se realizará la HU de mayor prioridad para el negocio (Interceptar y Modificar peticiones), por lo que se logrará interceptar y luego modificar la cabecera de las peticiones. Al final de esta iteración se tendrá una primera versión del sistema, la cual será mostrada al cliente.
- ✓ **Iteración 2:** el objetivo de esta iteración es la implementación de la HU (Configurar servidor Squid), la que permitirá aceptar o denegar las peticiones según los dominios que se encuentran configurados en el servidor. Como resultado se mostrará al cliente una versión más completa de las funcionalidades del sistema.
- ✓ **Iteración 3:** durante esta iteración se implementará la HU (Añadir dominio principal al registro log). Al finalizar la misma se contará con la versión 1.0 del producto final y el mismo podrá ser probado por el cliente.

A modo de resumen se muestra a continuación, la tabla# 6 que refleja la duración de cada iteración, así como el orden en que serán implementadas cada HU.

Iteración	Historia de Usuario	Duración
1	Interceptar y Modificar peticiones	3 semanas
2	Configurar servidor Squid	3 semanas
3	Añadir dominio principal al registro log	2 semanas

Tabla 6. Duración de cada iteración

2.5.3 Plan de Entregas

A continuación, se muestra el plan de entregas diseñado para la fase de implementación que comienza el día 1 de marzo de 2017. Como producto del mismo se harán entregas

incrementales del sistema al finalizar cada iteración, en la fecha aproximada que se indica en la tabla # 7.

Iteración	Fecha de entrega
1	21/03/2017
2	11/04/2017
3	25/04/2017

Tabla 7. Plan de entregas

2.6 Diseño

La metodología de desarrollo de software XP sugiere la elaboración de diseños sencillos para lograr un fácil entendimiento en la fase de desarrollo, permitiendo la reducción del tiempo de elaboración de la tarea asignada al desarrollador. Se debe tener en cuenta que un diseño complejo siempre tarda más en desarrollarse que uno simple, y que siempre es más fácil añadir complejidad a un diseño simple que quitarla de uno complejo.(Beck *et al;* 2005)

2.6.1 Arquitectura de software

La arquitectura de software de un programa o de un sistema computacional está definida por la estructura, comprendida por los elementos de software, las propiedades visibles de esos elementos y las relaciones entre ellos.(Ian Sommerville, 2009)

Para el desarrollo del presente trabajo se utilizó la arquitectura Cliente-Servidor, en la cual el cliente y el servidor son objetos separados de un punto de vista lógico y que se comunican a través de una red de comunicaciones para realizar una o varias tareas de forma conjunta. El cliente hace una petición de un servicio y el servidor recibe y procesa la petición y devuelve la respuesta al cliente.

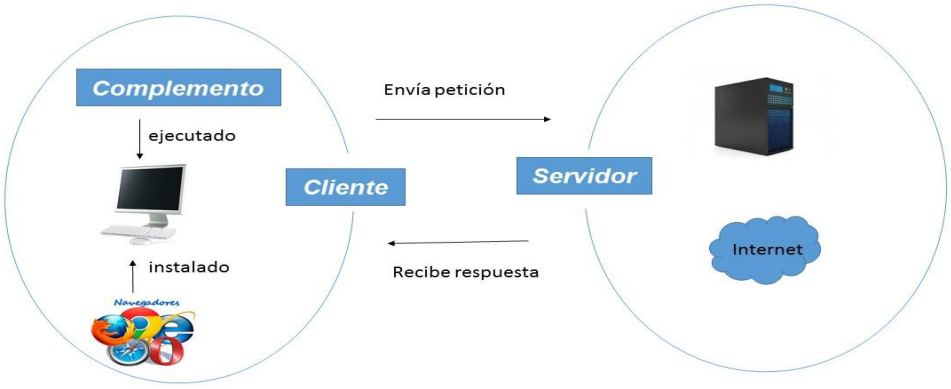


Figura 12. Definición de la arquitectura Cliente-Servidor

2.6.2 Patrones de diseño

Los patrones de diseño son soluciones para problemas típicos y recurrentes que se pueden encontrar a la hora de desarrollar una aplicación.(rubenfa, 2014) Se puede decir que se está en presencia de un patrón de diseño si la forma de solucionar un problema determinado se puede extraer, explicar y reutilizar en múltiples ámbitos.

Para diseñar la herramienta se hizo uso de Patrones Generales de Software para Asignar Responsabilidades (GRASP). Estos son guías o principios que sirven para asignar responsabilidades a las clases.(Maia Kord, 2011) A continuación, se describen algunos de los patrones GRASP utilizados:

- ✓ Experto: se manifiesta con el uso de clases que poseen responsabilidades específicas de acuerdo con la información que contienen y manejan. Se evidencia en la funcionalidad: function getRequests(e).
- ✓ Alta cohesión: se manifiesta a través de lo relacionadas que están las responsabilidades de una clase, es; decir, una clase con responsabilidades altamente relacionadas y que no lleva a cabo gran cantidad de trabajo: Se evidencia en las funcionalidades: createTab, updateTab, eliminarTab, fullInfoTab y onError.
- ✓ Bajo acoplamiento: el acoplamiento indica qué tan fuertemente está conectada una clase con otra, es; decir, qué tanto influye una clase sobre otra. Una clase con bajo acoplamiento no depende de otras clases. Se evidencia en las funcionalidades: createTab, eliminarTab, onError.

- ✓ Controlador: se manifiesta con el uso de clases responsables del manejo de los eventos del sistema, el controlador recibe la solicitud y coordina su realización delegando responsabilidades a otros objetos. Se evidencia en la funcionalidad: fullInfoTab.

2.6.3 Tarjetas CRC

En la fase de Diseño, la metodología XP no requiere la representación del sistema mediante diagramas de clases utilizando notación Lenguaje Unificado de Modelado (UML). En su lugar se usan otras técnicas como las tarjetas CRC como una extensión informal a UML. La técnica de las tarjetas CRC se puede usar para guiar el sistema a través de análisis guiados por la responsabilidad.(Cohn, 2004)

A continuación, se muestran las tarjetas CRC definidas para el presente trabajo:

Tarjeta CRC #1	
Clase: getRequests	
Responsabilidad	Colaboración
Es la que se encarga de interceptar y modificar las peticiones.	✓ fullInfoTab ✓ onError

Tabla 8. Tarjeta CRC#1

Tarjeta CRC #2	
Clase: fullInfoTab	
Responsabilidad	Colaboración
Es la que se encarga de mantener una información actualizada de las pestañas que se encuentran en la ventana sin tener que estar activas.	getRequests

Tabla 9. Tarjeta CRC#2

Tarjeta CRC #3	
Clase: onError	
Responsabilidad	Colaboración
Es la que se encarga de lanzar un error en caso de que ocurra un error con las pestañas.	getRequests

Tabla 10. Tarjeta CRC#3

Tarjeta CRC #4

Clase: createTab	
Responsabilidad	Colaboración
Esta funcionalidad es lanzada cuando se crea una nueva pestaña. Este método se encarga de actualizar la lista de pestañas, lo cual posteriormente servirá para verificar el dominio principal.	getRequests

Tabla 11. Tarjeta CRC#4

Tarjeta CRC #5	
Clase: updateTab	
Responsabilidad	Colaboración
Esta funcionalidad es lanzada cuando una pestaña se actualiza. En este método se actualiza la lista de pestañas, con el objetivo de actualizar la pestaña que está siendo actualizada, y con ello el dominio en cuestión.	getRequests

Tabla 12. Tarjeta CRC#5

Tarjeta CRC #6	
Clase: eliminarTab	
Responsabilidad	Colaboración
Es la que se encarga de actualizar la lista de pestañas, si se elimina una pestaña, se elimina de la lista de pestañas. Para perder toda referencia a la pestaña que emite el dominio principal.	getRequests

Tabla 13. Tarjeta CRC#6

2.7 Impacto social de la investigación

El presente trabajo investigativo tiene un impacto social que está basado en los siguientes planteamientos:

- En el lineamiento 131 de la Política Económica y Social del Partido y la Revolución plantea que: “Sostener y desarrollar los resultados alcanzados en el campo de la biotecnología, la producción médico-farmacéutica, la industria del software y el proceso de informatización de la sociedad, las ciencias básicas,

las ciencias naturales, los estudios y el empleo de las fuentes de energía renovables, las tecnologías sociales y educativas, la transferencia tecnológica industrial, la producción de equipos de tecnología avanzada, la nanotecnología y los servicios científicos y tecnológicos de alto valor agregado".

- En la UCI específicamente, apoya al proceso de toma de decisiones, en caso de posibles auditorías y cuando un usuario comete una falta grave que corresponda a seguridad informática. También en la resolución de control interno que se realiza en el nodo.
- La herramienta desarrollada les permite a los usuarios de la UCI, acceder a todos los contenidos empotrados en los sitios web confiables.

2.8 Conclusiones del capítulo

Con la realización de este capítulo fue posible especificar los resultados obtenidos luego de aplicar las fases de Exploración, Planificación y Diseño que propone la metodología XP. Luego de definidas las HU, la construcción del plan de iteraciones permitió conocer cuáles de ellas implementar en cada iteración y el orden de prioridad de cada una. Además, la confección del plan de entregas permitió conocer la fecha estimada en que se entregará una primera versión del producto al cliente.

Se definieron los patrones de diseño usados con el objetivo de lograr una mayor organización en los elementos que conforman la aplicación, así como la definición de las tarjetas CRC permitió identificar las clases principales y las relaciones entre ellas, posibilitando una reducción del acoplamiento y un aumento de la reutilización de la herramienta.

Capítulo 3: Implementación y Pruebas

3.1 Introducción

La metodología XP define que la implementación del software debe realizarse de forma iterativa, para obtener al final de cada iteración un producto funcional que debe ser probado y mostrado al cliente. En el presente capítulo se desarrollan las fases de Implementación y Pruebas que propone la metodología seleccionada. Se describen las tres iteraciones llevadas a cabo, definiendo en las mismas las tareas de ingeniería generadas por cada HU, así como las pruebas unitarias y de aceptación aplicadas al software. También se describen los estándares de codificación utilizados en la implementación del sistema propuesto para la investigación en curso.

3.2 Estándares de Codificación

Un estándar de codificación comprende todos los aspectos de la generación de código.(Carlos Benítez, 2012)

Al comenzar un proyecto de software, se debe establecer un estándar de codificación para asegurarse de que todos los programadores del proyecto trabajen de forma coordinada. Para lograr este objetivo se utilizó la guía de estilo para archivos JavaScript. A continuación, se mencionan algunos de los utilizados en la implementación del sistema propuesto en la presente investigación:

- ✓ Los nombres de las variables deben ser cortos y significativos.
- ✓ Declarar siempre las variables usando let.
- ✓ Se debe declarar cada variable en su propia línea.
- ✓ Los nombres de los métodos deben ser verbos o palabras que identifiquen de manera general el objetivo del método.
- ✓ Los nombres de los métodos no pueden contener espacios ni caracteres especiales, sólo son permitidas las letras de la “a” a la “z” y los números del 0 al 9.

3.3 Desarrollo por iteraciones

Durante el transcurso de las iteraciones se realiza la implementación de las HU seleccionadas para cada una de estas. Al inicio de las mismas, se lleva a cabo una revisión del plan de iteraciones y se modifica de ser necesario. Como parte de este plan, se descomponen las HU en tareas de ingeniería. Estas tareas son para el uso de los programadores, pueden escribirse utilizando un lenguaje técnico y no necesariamente deben ser entendibles para el cliente.(Beck *et al;* 2005)

En el presente trabajo, de acuerdo con la planificación realizada se llevaron a cabo tres iteraciones de desarrollo, obteniendo al finalizarlas un producto listo para su despliegue.

3.4 Implementación

En esta fase las HU se descomponen en tareas de programación o ingeniería, que a la vez son convertidas en código fuente. Se tiene en cuenta el desarrollo de las iteraciones según el plan de iteraciones definido, la entrega de la herramienta en iteraciones pequeñas y un diseño simple y poco redundante del código con las funcionalidades necesarias.

3.5 Pruebas

Las pruebas son fundamentales en XP, se ha desarrollado un enfoque en el que se reduce la probabilidad de producir nuevos incrementos del sistema que introduzcan errores en el software existente.(Ian Sommerville, 2009)

Las características claves de las pruebas en XP son:

- Desarrollo previamente probado.
- Desarrollo de pruebas incremental por escenarios.
- Participación del usuario en del desarrollo de las pruebas y en la validación.

Durante el desarrollo de cada una de las iteraciones del presente trabajo se realizaron pruebas de caja blanca (pruebas unitarias) y de caja negra (pruebas de aceptación). Las mismas fueron evaluadas en cada una de las iteraciones, obteniéndose un resultado final que se muestra en los próximos epígrafes.

3.5.1 Pruebas Unitarias

Las pruebas unitarias deben ser escritas antes que los métodos. Su implementación y ejecución deben consumir el menor tiempo posible.(Echeverry *et al;* 2007)

Las pruebas unitarias son una forma de probar el correcto funcionamiento del código. Sirven para asegurar que cada una de las funcionalidades de sistema proceda como se espera y son diseñadas por el programador.

3.5.2 Pruebas de Aceptación

Las pruebas de aceptación se deben diseñar en base a las HU.(Echeverry *et al*; 2007)

El objetivo de estas pruebas es validar que el sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema determinar el criterio de aceptación,

desde el punto de vista de la funcionalidad y el rendimiento. Las pruebas unitarias se realizaron manualmente.

3.6 Iteración 1

Durante esta iteración se desarrolló la HU #1 (Interceptar y Modificar peticiones) y al final de esta se logró una primera versión del sistema.

3.6.1 Tareas de Ingeniería

3.6.1.1 HU- Interceptar y Modificar peticiones

Tarea de Ingeniería # 1	
Nombre: Interceptar peticiones	
HU: Interceptar y Modificar peticiones	
Tipo de Tarea : Desarrollo	Puntos estimados: 1.6
Fecha de inicio: 1/03/2017	Fecha de fin: 10/03/2017
Descripción: Es el procedimiento donde se capturan las peticiones realizadas del lado del cliente.	

Tabla 14. Tarea de Ingeniería #1: Interceptar peticiones

Tarea de Ingeniería # 2	
Nombre: Modificar peticiones	
HU: Interceptar y Modificar peticiones	
Tipo de Tarea : Desarrollo	Puntos estimados: 1.4
Fecha de inicio: 13/04/2017	Fecha de fin: 21/04/2017
Descripción: Es el procedimiento donde a las peticiones interceptadas se les modifica la cabecera y se les agrega un nuevo campo denominado dominio principal.	

Tabla 15. Tarea de Ingeniería #2: Modificar peticiones

3.6.2 Pruebas unitarias

En esta primera iteración se realizaron 6 pruebas unitarias, en las cuales se obtuvo el resultado esperado, es decir; que los métodos probados funcionaron correctamente.

A continuación, se muestran ejemplos de las pruebas unitarias realizadas:

```
function testRequests() {
    console.log("Unitary test to all tab requests made....");

    let count = 0;
    for (let request of test_request) {
        let flag = false;
        for (let header of request.requestHeaders) {
            if (header.name === "dominio_principal") {
                flag = true;

            }
        }
        if (flag) {
            console.log("True");
            count++;
        }

    }
    if (count === array_request.length)
        console.log("OK ");

    test_request = [];
}
```

Figura 13. Prueba unitaria testRequests()

```
function testTabsOnUpdate(tabs) {
    console.log("Unitary test for tabs....");

    if (tabs.length === tabs_id.length) {
        console.log("Stored tabs: " + tabs_id.length + " Browsers tab: " + tabs.length+": OK");
    } else
        console.error("There is not relationship between the tabs ")
}
```

Figura 14. Prueba unitaria testTabsOnUpdate(tabs)

3.6.3 Resultados de las pruebas unitarias

Para la ejecución de las pruebas unitarias fue necesario seguir una serie de pasos:

1- Abrir una pestaña en el navegador.
2- Se escribe en la pestaña del navegador la dirección (correo.estudiantes.uci.cu).
3- Recargar la pestaña.

A continuación, se muestra el resultado de las pruebas unitarias realizadas:

- La segunda y la tercera línea hacen referencia a la primera prueba unitaria, la cual muestra todas las peticiones almacenadas en el arreglo e indica que todas las peticiones han sido modificadas.

- La quinta línea indica la cantidad de peticiones que se están realizando en la pestaña activa y verifica que han sido modificadas.
- La última línea indica la cantidad de pestañas activas en una o varias instancias del navegador.

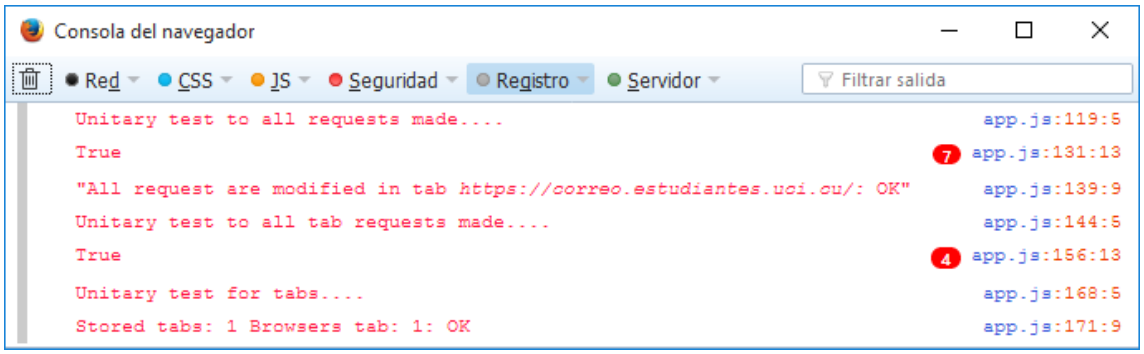


Figura 15. Resultado de las pruebas unitarias

3.7 Iteración 2

En esta segunda iteración se desarrolló la HU #2 (Configurar servidor Squid) y como resultado se obtuvo una versión más completa del producto final para el cliente.

3.7.1 Tareas de Ingeniería

3.7.1.1 HU- Configurar servidor Squid

Tarea de Ingeniería # 3	
Nombre: Permitir el acceso web	
HU: Configurar servidor Squid	
Tipo de Tarea : Desarrollo	Puntos estimados: 1
Fecha de inicio: 22/03/02017	Fecha de fin: 28/03/02017
Descripción: Es el procedimiento donde se le da acceso a los IP(Internet Protocol) de las máquinas clientes, mediante la configuración de las ACL.	

Tabla 16. Tarea de Ingeniería #3: Permitir el acceso web

Tarea de Ingeniería # 4	
Nombre: Controlar el acceso web	
HU: Configurar servidor Squid	
Tipo de Tarea : Desarrollo	Puntos estimados: 1
Fecha de inicio: 29/03/2017	Fecha de fin: 4/04/2017
Descripción: Es el procedimiento donde se controla el acceso web, a través de las ACL, definiendo los dominios permitidos y denegados respectivamente.	

Tabla 17. Tarea de Ingeniería #4: Controlar el acceso web

Tarea de Ingeniería # 5	
Nombre: Registrar logs	
HU: Configurar servidor Squid	
Tipo de Tarea : Desarrollo	Puntos estimados: 1
Fecha de inicio: 5/04/2017	Fecha de fin: 11/04/2017
Descripción: Es el procedimiento donde se almacenan los logs de las peticiones realizadas por el cliente.	

Tabla 18. Tarea de Ingeniería #5: Registrar logs

3.7.2 Pruebas de Aceptación

Para realizar las pruebas de aceptación correctamente se confeccionaron casos de pruebas para cada una de ellas. A continuación, se describen los casos de prueba correspondientes a la HU #2 Configurar servidor Squid.

Caso de Prueba de Aceptación
Nombre: Acceso a sitio web permitido.
HU: Configurar servidor Squid.
Descripción: Se intenta acceder a un sitio web permitido y carga en el navegador la URL solicitada.
Condiciones de ejecución: ✓ Tener instalado el navegador web Firefox. ✓ Instalación de complemento en el navegador web Firefox. ✓ Configurar el navegador para hacer las peticiones al servidor Squid previamente configurado.
Pasos de ejecución: Se escribe en el navegador la URL de un sitio web .
Resultado esperado: Se carga en el navegador la página de la URL solicitada.
Evaluación de la prueba: Satisfactoria

Tabla 19. Caso de Prueba de Aceptación. Acceso a sitio web *permitido*

Caso de Prueba de Aceptación
Nombre: Acceso a sitio web no permitido.
HU: Configurar servidor Squid.
Descripción: Se intenta acceder a un sitio web no permitido y no carga la URL solicitada, mostrando un mensaje de denegación en el navegador.
Condiciones de ejecución: ✓ Navegador web Firefox. ✓ Instalación de complemento en el navegador web Firefox.

✓ Configurar el navegador para hacer las peticiones al servidor Squid previamente configurado.
Pasos de ejecución: Se escribe en el navegador la URL de un sitio web.
Resultado esperado: Se muestra en el navegador un mensaje, indicado que no es posible acceder a la página de la URL solicitada.
Evaluación de la prueba: Satisfactoria.

Tabla 20. Caso de Prueba de Aceptación. Acceso a sitio web *no permitido*

Caso de Prueba de Aceptación
Nombre: Acceso a los contenidos empotrados.
HU: Configurar servidor Squid.
Descripción: Se intenta acceder a los contenidos empotrados dentro de un sitio web permitido y se visualizan estos contenidos.
Condiciones de ejecución: ✓ Navegador web Firefox. ✓ Instalación de complemento en el navegador web Firefox. ✓ Configurar el navegador para hacer las peticiones al servidor Squid previamente configurado.
Pasos de ejecución: Se escribe en el navegador la URL de un sitio web permitido.
Resultado esperado: Se carga en el navegador la página de la URL solicitada y se visualizan los contenidos empotrados dentro de la misma.
Evaluación de la prueba: Satisfactoria.

Tabla 21. Caso de Prueba de Aceptación. Acceso a los contenidos empotrados

En esta iteración se realizaron 6 pruebas unitarias y 3 pruebas de aceptación a la HU #2 Configurar servidor Squid.

3.8 Iteración 3

En la tercera iteración se desarrolló la HU #3 de prioridad media (Añadir dominio principal al registro log), donde se implementaron las funcionalidades relacionadas con el proceso de añadir el dominio principal de cada petición al registro log generado por el servidor proxy Squid. Al finalizar la misma se contará con la versión 1.0 del sistema y podrá se mostrado al cliente y probado por el mismo.

3.8.1 Tareas de Ingeniería

3.8.1.1 HU- Añadir dominio principal al registro log

Tarea de Ingeniería # 6
Nombre: Añadir dominio principal

HU: Añadir dominio principal al registro log	
Tipo de Tarea : Desarrollo	Puntos estimados: 2
Fecha de inicio: 12/04/2017	Fecha de fin: 25/04/2017
Descripción: Es el procedimiento donde se le añade el dominio principal al registro log generado por el servidor Squid.	

Tabla 22. Tarea de Ingeniería #6. Añadir dominio principal

3.8.2 Pruebas de Aceptación

A continuación, se describe el caso de prueba de aceptación correspondiente a la HU #3 Añadir dominio principal al registro log.

Caso de Prueba de Aceptación
Nombre: Anadir dominio principal.
HU: Añadir dominio principal al registro log.
Descripción: Luego de intentar acceder a la página web, se almacena en el archivo access_log del servidor Squid la petición con el dominio principal añadido.
Condiciones de ejecución: ✓ Navegador web Firefox. ✓ Instalación de complemento en el navegador web Firefox. ✓ Configurar el navegador para hacer las peticiones al servidor Squid previamente configurado.
Pasos de ejecución: Se escribe en el navegador la URL de un sitio web confiable.
Resultado esperado: Se muestra en el archivo access_log del servidor Squid la petición con el dominio principal añadido.
Evaluación de la prueba: Satisfactoria.

Tabla 23. Caso de Prueba de Aceptación. Añadir dominio principal

En esta iteración se realizaron 6 pruebas unitarias, las 3 pruebas de aceptación de la iteración anterior y 1 prueba de aceptación a la HU #3 Añadir dominio principal al registro log, sumando un total de 6 pruebas unitarias y 4 pruebas de aceptación.

A continuación, se muestra un gráfico que resume las pruebas unitarias y de aceptación realizadas en cada una de las iteraciones:

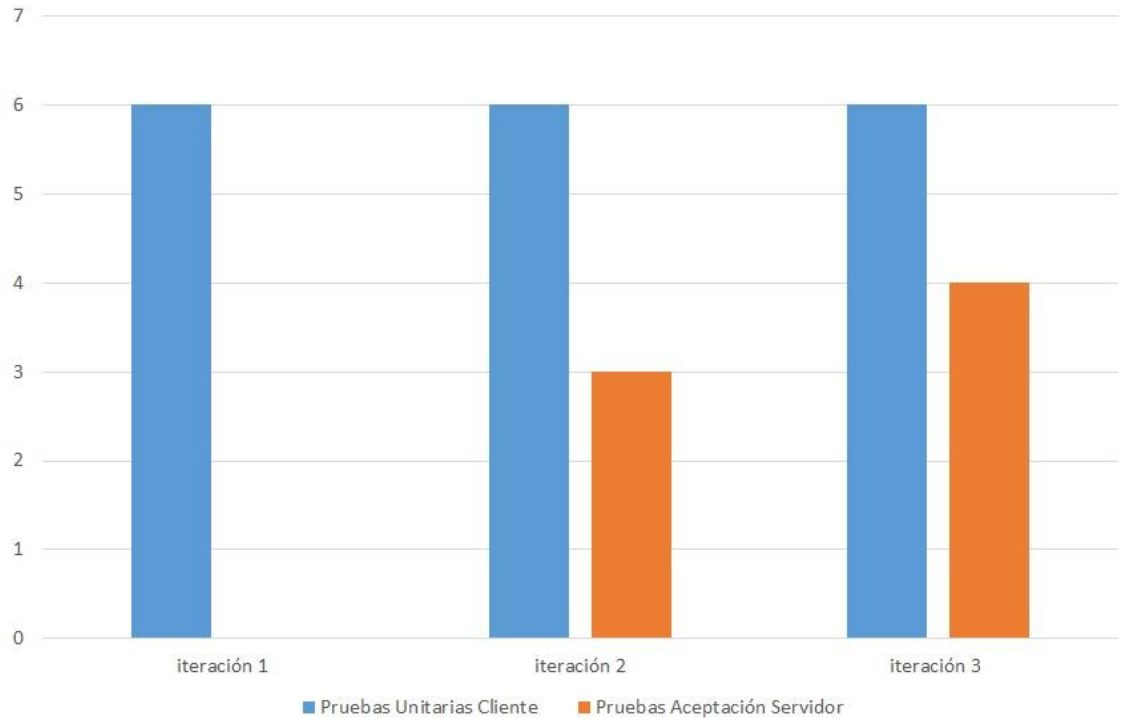


Figura 16. Resumen de las pruebas unitarias y de aceptación

3.9 Conclusiones del capítulo

En este capítulo se llevaron a cabo las fases de Implementación y Pruebas, definidas por la metodología XP. Se realizaron las tareas de ingeniería que dieron solución a todas las HU definidas, logrando una mayor rapidez y organización en el desarrollo del sistema. Se realizaron las pruebas unitarias y de aceptación, con resultados satisfactorios, contribuyendo estas a elevar la calidad del producto final, validando cada una de las funcionalidades propuestas y evitando así que el software llegue con defectos al cliente.

Conclusiones Generales

El desarrollo de la presente investigación permitió el desarrollo de la Herramienta para el procesamiento de las peticiones de la navegación web. Finalmente se obtuvieron las siguientes conclusiones:

- ✓ El estudio de las principales tecnologías y herramientas que permiten el procesamiento de las peticiones de la navegación web, garantizó la base metodológica sobre los elementos importantes que caracterizan el procesamiento de las peticiones.
- ✓ La selección de la metodología de desarrollo de software XP, permitió obtener un modelo de guía para la implementación del sistema, a partir de los artefactos generados por la misma, así como la culminación en tiempo a partir del cronograma planificado para cumplir con el objetivo de la investigación.
- ✓ La validación de la solución propuesta haciendo uso de las pruebas unitarias y de aceptación, permitió obtener una herramienta estable y correcta, lo cual garantiza que se cumpla satisfactoriamente el objetivo trazado en el presente trabajo de diploma, dando solución a la problemática planteada por el cliente.

Recomendaciones

Debido a los resultados de la investigación realizada y los conocimientos adquiridos durante la ejecución de este trabajo de diploma, se recomienda continuar con el desarrollo de la herramienta propuesta, teniendo en cuenta las siguientes recomendaciones:

- ✓ Asegurar que en el complemento los usuarios no sean capaces de modificar las cabeceras que este define.
- ✓ Mostrar el nuevo campo (dominio principal) que se añade actualmente al registro log, en los sistemas utilizados actualmente, como SRNI+, u otros.
- ✓ Desplegar la solución en la UCI, para comprobar su utilidad con el acceso a internet.

Acrónimos

UCI: Universidad de las Ciencias Informáticas

XP: *Extreme Programming* (Programación Extrema)

HTML: *Hyper Text Markup Language* (Lenguaje de Marcado de Hipertexto)

IDE: ***I****ntegrated Development Environment* (Entorno de Desarrollo Integrado)

API: Interfaz de Programación de Aplicaciones

CCS: Hoja de Estilo en Cascada

DNS: *Domain Name Server* (Servidor de Nombre de Dominio)

CRC: Clase Responsabilidad Colaboración

HTTP: *Hypertext Tranfer Protocol* (Protocolo de Transferencia de Hipertexto)

UML: Lenguaje Unificado de Modelado

SSL: *Secure Sockets Layer* (Capa Segura de Zócalos)

URL: *Uniform Resource Locator* (Localizador Uniforme de Servicios)

ACL: Listas de Control de Acceso

HU: Historia de Usuario

GRASP: Patrones Generales de Software para Asignar Responsabilidades

ICAP: Protocolo de Adaptación de Contenidos de Internet

IP: *Internet Protocol* (Protocolo de Internet)

WWW: World Wide *Web* (Red Mundial)

CTS: Ciencia-Tecnología-Sociedad

Referencias bibliográficas

ALEX DAWSON y ADRIAN CHADD, 2013. squid : Optimising Web Delivery. [en línea]. [Consulta: 27 abril 2017]. Disponible en: http://www.squid-cache.org/.

ANÓNIMO, 2017. HTTP Request / Response - Conceptos de Programacion. [en línea]. [Consulta: 16 junio 2017]. Disponible en: https://sites.google.com/site/conceptoprogramacion/request-response.

CARLOS BENÍTEZ, 2012. Guía de estilo. Nomenclatura en archivos Javascript | EtnasSoft. [en línea]. [Consulta: 14 junio 2017]. Disponible en: http://www.etnassoft.com/2012/10/23/guia-de-estilo-nomenclatura-en-archivos-javascript/.

CAVSI, 2007. ¿Qué es un Navegador Web? - Concepto y Definición. *Foro Tecnico* [en línea]. [Consulta: 12 abril 2017]. Disponible en: http://www.cavsi.com/preguntasrespuestas/que-es-un-navegador-web/.

COHN, M., 2004. *User Stories Applied: For Agile Software Development* [en línea]. ilustrada. S.l.: Addison-Wesley Professional. [Consulta: 2 marzo 2017]. ISBN 978-0-321-20568-1. Disponible en: https://books.google.ca/books/about/User_Stories_Applied.html?id=SvIwuX4SVigC.

DANIEL BARRON, 2011. DansGuardian - True Web Content Filtering for All. [en línea]. [Consulta: 23 marzo 2017]. Disponible en: http://dansguardian.org/.

FREDERIC BOURGEOIS, 2013. Web filtering solution. [en línea]. [Consulta: 23 marzo 2017]. Disponible en: http://e2guardian.org/cms/index.php.

GONZALO GARCÍA PIERRAT y MARÍA JOSEFINA VIDAL LEDO, 2016. La informática y la seguridad. Un tema de importancia para el directivo. *2016* [en línea], vol. 22. [Consulta: 7 noviembre 2016]. ISSN 1996-3521. Disponible en: http://www.revinfodir.sld.cu/index.php/infodir/article/view/177.

IAN SOMMERVILLE, 2009. *Software engineering / Ian Sommerville. — 9th ed.* 9th. S.l.: s.n. ISBN 978-0-13-703515-1.

JAIME ALONSO, 2008. El sitio web como unidad básica de información y comunicación. Aproximación teórica: definición y elementos constitutivos. , no. 5, pp. 226-247.

JETBRAINS, 2017. WebStorm: The Smartest JavaScript IDE. *JetBrains* [en línea]. [Consulta: 9 marzo 2017]. Disponible en: https://www.jetbrains.com/webstorm/.

KENT BECK y CYNTHIA ANDRES, 2005. *Extreme Programming Explained: Embrace Change*. S.l.: Addison-Wesley. ISBN 0-321-27865-8.

LEANDRO ALEGSA, 2010a. Definición de petición (computación). [en línea]. [Consulta: 28 marzo 2017]. Disponible en: http://www.alegsa.com.ar/Dic/peticion.php.

LEANDRO ALEGSA, 2010b. Definicion de Servidor web. *Diccionario de Informática y Teconlogía* [en línea]. [Consulta: 13 junio 2017]. Disponible en: http://www.alegsa.com.ar/Dic/servidor_web.php.

LIONEL PAIRUNA, 2017. Información sobre web / ¿ Qué es y para que sirve un sitio web ? *Code Dimension, Diseño Web Profesional en Salta, Argentina* [en línea]. [Consulta: 19 junio 2017]. Disponible en: http://www.codedimension.com.ar/noticias-sobre-tecnologia/noticias/-que-es-y-para-que-sirve-un-sitio-web-/1.

LUIS MIGUEL ECHEVERRY TOBÓN y LUZ ELENA DELGADO CARMONA, 2007. Caso práctico de la metodología ágil XP al desarrollo de software. . UNIVERSIDAD TECNOLÓGICA DE PEREIRA:

MAIA KORD, 2011. Patrones GRASP. Patrones GoF. Diferencia entre GRASP y GoF - TuxNots. [en línea]. [Consulta: 3 marzo 2017]. Disponible en: http://tuxnots.com.ar/materias-de-la-facu/metodologia-de-sistemas/patronesgrasppatronesgofdiferenciaentregraspygof.

Manual de usuario Smart Keeper, 2014. marzo 2014. S.l.: s.n.

MOZILLA, 2016. Lightbeam para Firefox. *Mozilla* [en línea]. [Consulta: 9 noviembre 2016]. Disponible en: https://www.mozilla.org/es-ES/lightbeam/.

MOZILLA DEVELOPER NETWORK, 2016. Introducción. *Mozilla Developer Network* [en línea]. [Consulta: 9 marzo 2017]. Disponible en: https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Introducci%C3%B3n.

MOZILLA DEVELOPER NETWORK, 2017. WebExtensions. *Mozilla Developer Network* [en línea]. [Consulta: 9 marzo 2017]. Disponible en: https://developer.mozilla.org/en-US/docs/Mozilla/Add-ons/WebExtensions.

OMAR MAR CORNELIO, LISSETE GONZALEZ GALLO, BÁRBARA BRON FONSECA y YUSIMIL DAVILA FONSECA, 2014. Aplicación informática para la gestión de recursos a través del directorio activo de la Universidad de las Ciencias Informáticas. , no. 45, pp. 10.

ORGANIZACIÓN DE LOS ESTADOS AMERICANOS, SYMANTEC, AMERIPOL, LACNIC, MICROSOFT, LA CORPORACIÓN DE INTERNET PARA LA ASIGNACION DE NOMBRES Y NUMEROS y ANTI-PHISHING WORKING GROUP, 2014. Tendencias de Seguridad Cibernética en América Latina y el Caribe. [en línea]. Symantec Corporation: [Consulta: 7 noviembre 2016]. Disponible en: https://www.symantec.com/es/mx/page.jsp?id=cybersecurity-trends.

RUBENFA, 2014. Patrones de diseño: qué son y por qué debes usarlos. *Genbeta Dev* [en línea]. [Consulta: 3 marzo 2017]. Disponible en: https://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.

SERGIO LUJÁN MORA, 2017. Accesibilidad Web: Complementos para navegadores. [en línea]. [Consulta: 12 abril 2017]. Disponible en: http://accesibilidadweb.dlsi.ua.es/?menu=hr-complementos.

SUE SMITH, 2016. Herramientas usadas en el desarrollo de software | eHow en Español. [en línea]. [Consulta: 9 noviembre 2016]. Disponible en: http://www.ehowenespanol.com/herramientas-usadas-desarrollo-software-info_271852/.

Bibliografía

ALLEN DOWNEY, JEFFREY ELKNER y CHRIS MEYERS, 2002. *Aprende a Pensar Como un Programador en Python* [en línea]. S.l.: s.n. ISBN 0-9716775-0-6. Disponible en: http://www.thinkpython.com.

BOLINAGA, E.L., 2013. Descubre quién rastrea tus datos y tu actividad en línea. *Firefoxmanía* [en línea]. [Consulta: 7 marzo 2017]. Disponible en: https://firefoxmania.uci.cu/descubre-quien-rastrea-tus-datos-y-tu-actividad-en-linea/.

DJANGO SOFTWARE FOUNDATION, 2015. The Web framework for perfectionists with deadlines | Django. [en línea]. [Consulta: 7 marzo 2017]. Disponible en: https://www.djangoproject.com/.

DUSSAN CLAVIJO y CIRO ANTONIO, 2006. Políticas de Seguridad Informática. *Entramado*, vol. 2, no. num 1, pp. 8. ISSN 1900-3803.

Firewall - Maravento. [en línea], 2017. [Consulta: 19 junio 2017]. Disponible en: http://www.maravento.com/2013/03/firewall.html.

FRANCISCO CRUZ ARGUDO, 2008. RedIRIS - Seguridad en Redes y Sistemas. [en línea]. [Consulta: 7 marzo 2017]. Disponible en: http://www.rediris.es/difusion/publicaciones/boletin/35/enfoque2.html.

HENRIK KNIBERG, 2007. *Scrum y XP desde las trincheras Como hacemos Scrum*. Estados Unidos de América: InfoQ.com. ISBN 978-1-4303-2264-1.

JACK WALLEN, 2015. Filter Content on Your Home Network with E2guardian | Linux.com | The source for Linux information. [en línea]. [Consulta: 19 junio 2017]. Disponible en: https://www.google.com/search?q=Filter+Content+on+Your+Home+Network+with+E2guardian+_+Linux.com+_+The+source+for+Linux+information.html&ie=utf-8&oe=utf-8&client=firefox-b-ab.

JOEL BARRIOS DUEÑAS, 2016. Configuracion del servidor Proxy Squid. [en línea]. S.l.: Disponible en: http://www.linuxparatodos.net/.

JOEL BARRIOS DUEÑAS, 2017. Configuracion de Squid_Opciones basicas- Alcanc Libre. [en línea]. [Consulta: 23 marzo 2017]. Disponible en: http://www.alcancelibre.org/staticpages/index.php/19-0-como-squid-general.

KREŠIMIR NESEK, 2015. Manually Installing E2guardian to pfSense. [en línea]. [Consulta: 19 junio 2017]. Disponible en: http://knes1.github.io/blog/2015/2015-07-18-manually-installing-e2guardian-to-pfsense.html.

LUIS CASTRO, 2017. ¿Qué es un complemento o extensión? [en línea]. [Consulta: 18 mayo 2017]. Disponible en: http://aprenderinternet.about.com/od/Glosario/g/Que-Es-Un-Complemento.htm.

LUIS FERNANDEZ, 2007. Una revisión sistemática de la adaptación del proceso

software. *Revista Española de Innovación Calidad e Ingeniería del Software*, vol. 3, no. 2, pp. 21-38.

Manual de Squid (uso, instalación y configuración) X Arael25 - Foros de programación informática - Comunidad de programación. [en línea], 2009. [Consulta: 19 junio 2017]. Disponible en: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwjqvLGv1srUAhXHnRoKHW9vDokQFgghMAA&url=https%3A%2F%2Fwww.foro.lospillaos.es%2Fmanual-de-squid-uso-instalacion-y-configuracion-x-arael25-vt5766.html&usg=AFQjCNF-vTI4Pju4awc2vEqY5GsbLhSkuQ&cad=rja.

MARELY DEL ROSARIO CRUZ FELIPE, REINIER MARTÍNEZ GÓMEZ y CARLOS M. HIERREZUELO PEREZ, 2013. ANÁLISIS DE TRÁFICO EN LA RED UCI MEDIANTE LA SIMULACIÓN. *Revista Telem@tica*, vol. 12, no. No 1, pp. 14. ISSN 1729-3804.

MICHEL MIRANDA CAIRO, OSMANY VALDÉS PUGA, IVÁN PÉREZ MALLEA, RENIER PORTELLES COBAS y RAÚL SÁNCHEZ ZEQUEIRA, 2016. Metodología para la Implementación de la Gestión Automatizada de Controles de Seguridad Informática. *Revista Cubana de Ciencias Informaticas*, vol. 10, no. num 2, pp. 14. ISSN 1994-1536.

MICROSOFT CORPORATION, 2016. Microsoft ISA Server: ¿Qué es ISA Server 2006? [en línea]. [Consulta: 7 marzo 2017]. Disponible en: https://www.microsoft.com/spain/isaserver/prodinfo/whatis.mspx.

MIGUEL ANGEL ALVAREZ, 2009. Manual de jQuery. *DesarrolloWeb.com* [en línea]. [Consulta: 23 marzo 2017]. Disponible en: http://www.desarrolloweb.com/manuales/manual-jquery.html.

OSWALDO CASTILLO, DANIEL FIGUEROA y HÉCTOR SEVILLA, 2013. Programación Extrema. [en línea]. S.l.: [Consulta: 18 abril 2017]. Disponible en: http://programacionextrema.tripod.com/index.htm.

PyAr - Python Argentina. [en línea], 2016. [Consulta: 7 marzo 2017]. Disponible en: http://www.python.org.ar/.

ROLANDO ALFREDO HERNÁNDEZ LEÓN y SAYDA COELLO GONZÁLEZ, 2011. *El Proceso De Inevstigación Científica*. Cuidad de la Habana: Editorial Universitaria del Ministerio de Educación Superior. ISBN 978-959-16- 1307-3.

SERGIO LOPEZ, 2013. PROXY SQUID PARA WINDOWS SERVER – Blog de soluciones.si. [en línea]. [Consulta: 19 junio 2017]. Disponible en: https://www.soluciones.si/blog/2013/04/19/configuracion-de-proxy-squid-sistema-windows/.

SurfControl Web Filter. [en línea], 2016. [Consulta: 7 marzo 2017]. Disponible en: http://www.e-projectgroup.cl/Partners/SurfControl/surfcontrol_webfilter.htm.

YURISLEIDY HERNANDEZ MOYA, DOVIER ANTONIO RIPOLL MENDEZ, LUIS

ENRIQUE SANCHEZ ARCE, KIUVER KADDIEL IBAÑEZ CASTRO y KAREL ANTONIO VERDECIA ORTIZ, 2012. Técnicas de Inteligencia Artificial en el filtro de contenido web Smart Keeper para la clasificación de información. *Revista Cubana de Ciencias Informáticas*, pp. 11. ISSN 1994-1536.

Anexo

Entrevista a Omar Pimentel Alfonso, especialista de la Dirección de Servicios Telemáticos.

Breve explicación de la solución propuesta:

Se propone como solución informática el desarrollo de un complemento para los navegadores web, utilizando WebExtensions. Este permite interceptar las peticiones enviadas desde el lado cliente y una vez obtenidas se realiza una modificación de las mismas, específicamente en sus cabeceras.

El navegador con el complemento integrado, le envía la petición al servidor Squid. Luego el mismo se encarga de analizar la cabecera de la petición enviada y determina si permitir o denegar la petición basándose en el campo referer que contiene la URL que dio inicio a la petición. También se añade el campo dominio principal de cada petición al registro (log) generado por el servidor proxy, para posteriormente poder organizar las trazas jerárquicamente en otras herramientas.

1. ¿Cuándo un sitio web es considerado confiable o no?
2. De forma general, ¿cómo funciona el proceso de navegación por internet en la UCI?
3. ¿La herramienta Smart Keeper es usada actualmente en la UCI?
4. ¿Cuál es la herramienta que se usa para el filtrado de contenido web actualmente en la UCI, E2guardian o Dansguardian?
5. De forma general, ¿cómo es el funcionamiento de esa herramienta?
6. Algunos de los métodos que utilizan para filtrar el contenido real de las páginas son:

- la coincidencia de frases
- encabezado de la solicitud
- filtrado URL
- filtrado de PICS

¿Cómo funcionan esos métodos?

7. La herramienta E2guardian, ¿cómo realiza el análisis y la manipulación de las cabeceras de cada petición?
8. ¿Cuál es la diferencia entre el E2guardian y Dansguardian?
9. ¿Sería factible usar el E2guardian para el desarrollo de nuestra propuesta de solución?