UNIVERSIDAD CENTRAL DE VENEZUELA
FACULTAD DE CIENCIAS
ESCUELA DE COMPUTACIÓN

Trabajo Especial de Grado: Desarrollo de una herramienta para el monitoreo de eventos y actividades de usuarios en aplicaciones Web

Trabajo Especial de Grado presentado ante la ilustre Universidad Central de Venezuela por el

Br. Ricardo C. Hergueta F.
CI 18.031.704

Tutor:
Prof. Andrés Castro

Noviembre 2008

Acta

Quienes suscriben, miembros del jurado designado por el Consejo de Escuela de Computación, para examinar el Trabajo Especial de Grado titulado "Desarrollo de una herramienta para el monitoreo de eventos y actividades de usuarios en aplicaciones Web" y presentado por el Br. Ricardo Carpentier Hergueta Fermín, de Cédula de Identidad 18.031.704 a los fines de optar al título de **Licenciado en Computación**, dejamos constancia de lo siguiente:

Leído como fue dicho trabajo, por cada uno de los miembros del jurado, se fijó el día 10 de noviembre de 2008, a las 3:00 pm., para que el autor lo defendiera en forma pública, lo que este hizo en la Sala PB III de la Escuela de Computación, mediante una presentación oral de su contenido, luego de lo cual respondió a las preguntas formuladas. Finalizada la defensa pública del Trabajo Especial de Grado, el jurado decidió aprobar con la nota de puntos.

En fe de lo cual se levanta la presente Acta, en Caracas el día 10 de noviembre de 2008.

Dejando también constancia de que actuó como coordinador del jurado en Prof. tutor Andrés Castro.

_____

Prof. Andrés Castro
Tutor

_____ _____

Prof. Nora Montaño
(Jurado)

Prof. Sergio Rivas
(Jurado)

Dedicatoria y agradecimientos

A Dios, por todas las bendiciones que me ha concedido.

A Marisol y a Carlos, mis padres, por su amor y apoyo incondicional.

A mi tía Solmar y mi primo Mikel, por haberme hecho la vida más fácil a lo largo de la carrera.

A mi tía Nela y mis primos Edwin y Tavo, por estar allí cuando los necesité.

A Jossie, por su perseverancia y por su adoptarme de manera oportuna.

A Andrés, mi Tutor, por asesorarme y ayudarme contra todo pronóstico.

A los profesores Nora Montaño y Sergio Rivas, por su respaldo y consejos para la culminación exitosa de esta investigación.

Al Ing. Hugo Aponte, por haber hecho posible que una idea se convirtiera en una investigación seria e importante.

A mis amigos y compañeros de estudios: Yurbelis, George (K1), Carlos (Guillermo), Gustavo, Jhonny, Joselyn y Martha; por todas las experiencias compartidas durante la carrera.

Resumen

La finalidad del presente Trabajo Especial de Grado consiste en el desarrollo de una herramienta con la capacidad de: realizar el monitoreo de eventos y acciones de usuarios, durante su interacción con aplicaciones y sitios Web, y generar información que pueda ser utilizada posteriormente, con múltiples propósitos. Para ello, se aplicó una adaptación del modelo de desarrollo en cascada, bajo una metodología Ad hoc, la cual se realizó en cinco fases: descripción de requerimientos, análisis, diseño, implementación y, finalmente, la fase de pruebas de la herramienta desarrollada. Como resultado final se obtuvo una herramienta de fácil integración, con la capacidad de realizar el proceso de grabación de eventos y acciones relevantes de los usuarios, llevar a cabo reproducciones automáticas de dichas grabaciones y generar reportes con información de utilidad acerca de las actividades registradas por los usuarios de cualquier aplicación Web.

Palabras clave

Monitoreo de usuarios, reproducción automática, eventos, grabación, JavaScript, Page Tagging, Document Object Model (DOM).

Índice de contenidos

Índice de figuras y tablas

Introducción

Las aplicaciones Web, a medida que pasa el tiempo, van ganando mayor importancia y popularidad debido al rápido avance de la tecnología y a las facilidades que estas ofrecen, tales como la practicidad de los navegadores Web para el acceso de los usuarios y la rapidez para su actualización.

Como consecuencia directa de esta popularidad, las aplicaciones Web de hoy día, independientemente de su finalidad, tienden a ser más complejas y utilizadas por un gran número de usuarios, con características diferentes.

Para los responsables de las aplicaciones Web, resulta de gran importancia conocer las actividades que los usuarios llevan a cabo dentro de dichas aplicaciones, debido a que con esa información se pueden realizar una series de tareas y tomas de decisiones que van desde conocer las funcionalidades o servicios de interés para los usuarios, hasta detectar errores en la aplicación.

En la actualidad, existen diversos mecanismos para el monitoreo de eventos y actividades de los usuarios, ya sea realizado de forma manual (con todas las limitaciones que el factor humano pueda acarrear) o mediante el uso de software y diversos dispositivos de grabación (tomando en cuenta la dificultad y costos asociados de uso e integración, así como también la dependencia tecnológica).

Por estas razones, se plantea como objetivo de la investigación, desarrollar una herramienta de fácil integración, para monitorear eventos y actividades de los usuarios en aplicaciones Web, utilizando tecnologías comunes y disponibles para el desarrollo Web. Empezando por definir los eventos dentro de las aplicaciones a ser monitoreados, desarrollar la herramienta con su procedimiento de uso respectivo y, finalmente, evaluar la herramienta para comprobar sus funcionalidades.

Con dicha investigación se busca brindar un nuevo aporte y ofrecer una alternativa diferente a lo ya existente.

El documento se encuentra estructurado de la siguiente manera:

En el Capítulo I se describe el Problema de Investigación. Tratando los puntos de: planteamiento del problema, solución propuesta, objetivos de la investigación, importancia y justificación, alcance de la investigación

El capítulo II presenta el marco conceptual donde se muestran las bases en las que se fundamenta el desarrollo de este trabajo. Se describen algunos de los aspectos relevantes de las aplicaciones Web, arquitectura cliente/servidor, protocolo HTTP, usabilidad, necesidad del monitoreo de eventos y la técnica de Page Tagging, además de algunas herramientas de automatización Web existentes en la actualidad y las tecnologías, tanto del lado del cliente como del servidor, utilizadas para el desarrollo de la herramienta propuesta así como también la metodología de desarrollo a emplear.

El capítulo III esta constituido por el marco aplicativo. En el cual se describe el conjunto de etapas realizadas para el desarrollo de la herramienta, comenzando con la recopilación de requerimientos: funcionales y no funcionales; análisis del problema formalizado a través del modelo de: casos de uso, objeto del dominio y de análisis; diseño de la aplicación modelado a través del: diagrama de clases, diagrama de base de datos, WAE (Web Application Extension) y diagrama de secuencia; y, finalmente, se especifica cómo ha sido la implementación de la solución y las pruebas realizadas para comprobar su correcto funcionamiento.

Por último se presentan las conclusiones y recomendaciones a las cuales condujo el desarrollo de este trabajo.

Capítulo I Problema de investigación

1.1 Planteamiento del Problema

Las aplicaciones Web de hoy día tienden a ser complejas y utilizadas por un gran número de usuarios, con diferentes características.

El conocimiento acerca del comportamiento de los usuarios y las actividades generadas por medio de la interacción con una aplicación Web, resulta un recurso valioso y de gran utilidad para los desarrolladores y responsables de la misma, ya que el registro de las actividades que realizan puede servir como información para ser utilizada con múltiples propósitos, desde realizar pruebas y detectar errores en las aplicaciones, hasta categorizar a los usuarios y conocer sus intereses, capacidades y limitaciones.

El monitoreo y registro de eventos y actividades de los usuarios, en aplicaciones Web, resulta una tarea compleja ya sea que se realice de forma manual, requiriendo la intervención de personas capacitadas, propenso al error humano y tiempos prolongados; o por medio de una herramienta de software, considerando dificultades en integración, implementación y dependencia tecnológica.

La automatización, como proceso, puede ser utilizada para complementar o sustituir procesos manuales, permitiendo reducir los tiempos y mejorar el rendimiento, en algunas áreas.

Teniendo conocimiento de lo anteriormente expuesto, en el contexto de las aplicaciones Web, surge la necesidad de desarrollar una herramienta con la cual monitorear y registrar las actividades de los usuarios, que pueda integrarse fácilmente con aplicaciones existentes y permita generar información útil para los desarrolladores y demás personas involucradas, con la finalidad de brindar un nuevo aporte y ofrecer una alternativa diferente a la ya existente.

1.2 Solución Propuesta

Una vez descrito el problema de investigación, se propone en este Trabajo Especial de grado el desarrollo de una herramienta para monitorear eventos y actividades de usuarios en aplicaciones Web la cual, aprovechándose de la existencia de tecnologías comunes y disponibles en el desarrollo Web y de procesos como la automatización, tenga la capacidad de generar información que resulte de utilidad para las personas involucradas con el desarrollo y funcionamiento de las aplicaciones.

Para iniciar el proceso de desarrollo de la herramienta, se realizó una revisión bibliográfica exhaustiva para determinar qué aspectos de las aplicaciones Web van a ser tomados en cuenta. De igual manera se llevaron a cabo una serie de pruebas de conceptos para verificar la factibilidad en la implementación de algunas de las ideas pensadas para la aplicación.

1.3 Objetivos

En esta sección se definen el objetivo general y objetivos específicos que deben ser cumplidos en esta investigación.

1.3.1 Objetivo General

Desarrollar una herramienta de fácil integración, para monitorear eventos y actividades de usuarios en aplicaciones Web, utilizando tecnologías comunes y disponibles en el desarrollo Web.

1.3.2 Objetivos Específicos

- Definir el conjunto de actividades y eventos a ser monitoreados y registrados.

- Describir el mecanismo para reproducir actividades y eventos realizados por un usuario en un momento determinado.

- Analizar, diseñar e implementar la herramienta para el monitoreo de eventos.

- Diseñar el procedimiento para la utilización de la herramienta propuesta.

- Evaluar la herramienta desarrollada para comprobar su efectividad.

1.4 Importancia y Justificación

Las aplicaciones Web, con el tiempo han obtenido mayor importancia y popularidad debido al rápido avance de la tecnología y a las facilidades que éstas ofrecen, tales como la practicidad de los navegadores Web para el acceso de los usuarios y la rapidez para su actualización.

Como consecuencia directa de esta popularidad, las aplicaciones Web de hoy día son más complejas y utilizadas por un gran número de usuarios con diferentes características.

Mediante el monitoreo y registro de actividades que los diferentes usuarios realizan en una aplicación, se puede observar y analizar su comportamiento con la finalidad de conocer sus intereses, capacidades, limitaciones y otros factores, con respecto al uso de la aplicación. Además se pueden visualizar las secuencias de navegación dentro de la aplicación, para realizar los cambios que faciliten el acceso a funcionalidades frecuentemente utilizadas y examinar caminos, sin usar o poco frecuentes, para comprender la razón por la cual los usuarios los evitan.

De igual manera, las actividades registradas pueden servir como información de entrada para realizar pruebas posteriores de usabilidad, seguridad, rendimiento, entre otras; con la finalidad de detectar fallas o comportamientos inesperados y para llevar a cabo la depuración de errores mediante la repetición de acciones previamente registradas.

De allí la importancia que tiene este Trabajo Especial de Grado al presentar el desarrollo de una herramienta para realizar el monitoreo de eventos y actividades de usuarios en aplicaciones Web para generar información de utilidad, tomando en cuenta aspectos tales como facilidad de integración y automatización para facilitar el proceso y lograr un ahorro tanto de tiempo como de recursos.

1.5 Alcance

El alcance de esta investigación es generar una herramienta de monitoreo de eventos y actividades de los usuarios en aplicaciones Web que resulte sencilla de integrar.

La herramienta a ser elaborada debe estar en la capacidad de monitorear algunos eventos y acciones generadas por el usuario, peticiones AJAX y otros sucesos que sean considerados relevantes de manera automática. De igual manera, debe poder realizar la reproducción o replica de eventos previamente registrados en aplicaciones Web y generar diversos reportes a ser presentados, tomando en cuenta la información registrada.

La herramienta debe estar en la capacidad de funcionar de manera correcta en algunos de los navegadores populares de la actualidad: Internet Explorer 6 y 7, y Firefox 2 y 3.

Capítulo II Marco Conceptual

2.1 Aplicaciones Web

[1] Las aplicaciones Web son aquellas aplicaciones accedidas a través de un navegador Web por medio de una de una red. También pueden ser vistas como un software de computadora codificado en un lenguaje soportado por navegadores (tales como HTML, Javascript, etc.) el cual depende de un navegador Web común para desplegar el ejecutable de la aplicación.

La facilidad para actualizar y mantener aplicaciones Web sin distribuir e instalar software en miles de potenciales clientes es otra razón de su popularidad. Es importante mencionar que una página Web puede contener elementos que permiten una comunicación activa entre el usuario y la información lo cual permite que el usuario acceda a ella de modo interactivo, gracias a que la página responderá a cada una de sus acciones.

Aunque muchas variaciones son posibles, una aplicación Web está comúnmente estructurada en tres capas (Ver Figura #1). En su forma más común, el navegador Web es la primera capa, un motor usando alguna tecnología Web dinámica (ejemplo: CGI, PHP, Java Servlets o ASP) es la capa del medio, y una base de datos como última capa. El navegador Web envía peticiones a la capa media, que la entrega valiéndose de consultas y actualizaciones a la base de datos generando una interfaz de usuario.

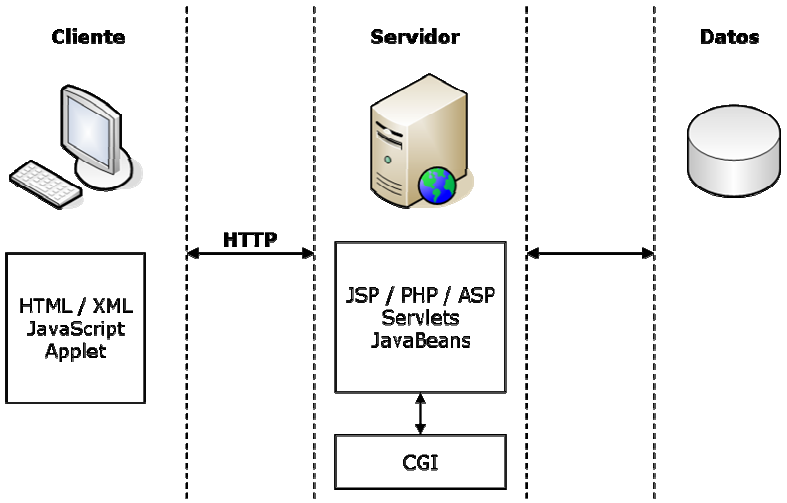

Figura #1: Estructura general de las aplicaciones Web

En tiempos recientes se ha usado la estrategia de generalizar esta arquitectura mediante la adición de piezas de hardware que permiten balancear la carga de los servidores Web y de aplicación.

Es importante resaltar que es extremadamente difícil llegar a niveles de perfección, por lo cual siempre se requiere la continua evaluación y un proceso de mejoras de cada una de las cosas que nos rodean y la Web es una de ellas. Es por ello que se debe planificar la calidad de las aplicaciones durante todo el ciclo de vida, lo que garantizara una detección y corrección de errores de manera oportuna.

2.2 Modelo cliente / servidor

Es un modelo de sistema en el que éste se organiza como un conjunto de servicios y servidores asociados, más unos clientes que acceden y usan los servicios. Los principales componentes del modelo son [2]:

- Un conjunto de servidores que ofrecen servicios a otros subsistemas.

- Un conjunto de clientes que llaman a los servicios ofrecidos por los servidores.

- Una red que permite a los clientes acceder a estos servicios. Esto no es estricto ya que los clientes y los servidores podrían ejecutarse en la misma máquina. En la práctica, sin embargo, la mayoría de los sistemas cliente-servidor se implementan como sistemas distribuidos.

Los clientes pueden conocer los nombres de los servidores disponibles y los servicios que éstos proporcionan. Sin embargo, los servidores no necesitan conocer la identidad de los clientes o cuantos clientes tienen. Los clientes acceden a los servicios proporcionados por un servidor a través de llamadas a procedimientos remotos usando un protocolo de petición-respuesta tal como el protocolo HTTP. Básicamente, un cliente realiza una petición a un servidor y espera hasta que recibe una respuesta.

Una de las variantes del modelo Cliente/Servidor más utilizadas en la actualidad es la de Tres Capas, la cual se muestra en la Figura #2. La arquitectura Cliente/Servidor de tres capas, se constituye de un código de presentación, de procesamiento de datos y de almacenamiento de datos [3]

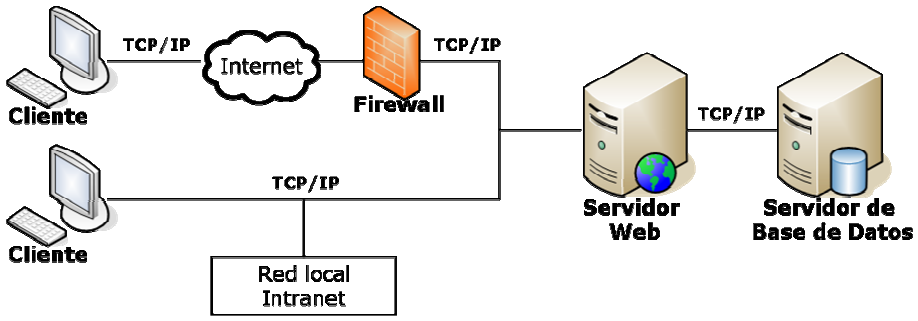

Figura #2: Arquitectura cliente/servidor de tres capas

Esta arquitectura se basa en el paradigma de ubicar el código de presentación, de procesamiento y de almacenamiento en servidores diferentes con el fin de separar en módulos, el trabajo. En términos generales, la capa de presentación proporciona la interfaz necesaria para presentar información y reunir datos. La capa de procesamiento responde a peticiones del usuario para ejecutar una tarea en específico, interactuando con los datos que están almacenados. La capa de almacenamiento representa las fuentes de datos finales y está formada por

uno o más gestores de bases de datos que realizan todo el almacenamiento y reciben solicitudes de recuperación de información desde la capa de negocio.

2.3 HyperText Transfer Protocol (HTTP)

Es un protocolo a nivel de aplicación para sistemas de información multimedia distribuidos. En un protocolo no orientado a estado que puede ser utilizado, entre muchas otros propósitos, para manejar ficheros HTML. Entre sus principales propiedades encontramos las siguientes [4]:

- Un esquema de direccionamiento comprensible: utilizando el Universal Resource Identifier (URI), para localizar sitios (URL) o nombres (URN) sobre los que hay que aplicar un método. La forma general de una URL es: servicio://host/ fichero.extensión

- Arquitectura cliente servidor: HTTP se asienta en el paradigma solicitud / respuesta. La comunicación se asienta sobre TCP/IP. El puerto por defecto es el 80, pero pueden ser utilizados otros puertos.

- Es un protocolo sin conexión ni estado: luego de que el servidor ha respondido la petición del cliente, la conexión entre ambos se rompe. Además no se guarda memoria de contexto de la conexión actual para conexiones posteriores.

- Está abierto para nuevos tipos de datos: utiliza tipos MIME (Multipart Internet Mail Extension) para la determinación de los tipos de datos que transporta. Cuando un servidor HTTP transmite información de regreso al cliente incluye una cabecera que le indica al cliente sobre los tipos de datos que componen el documento. De la gestión de esos datos se encargan las utilidades que tenga el cliente (visor de imágenes, de vídeo, etc.)

2.4 Usabilidad

Es un atributo de calidad que indica que tan fáciles de usar son las interfaces de usuario. La palabra usabilidad también se refiere a los métodos para mejorar la "facilidad de uso" durante el proceso de diseño. La usabilidad es definida por cinco componentes de calidad [5]:

- Intuición (Learnability): indica que tan fácil resulta para los usuarios cumplir con las tareas básicas la primera vez que se enfrentan a la interfaz.

- Eficiencia: indica que tan rápido los usuarios realizan las tareas, una vez que conocen la interfaz.

- Memorización: indica la rapidez con la cual los usuarios retoman el dominio en el manejo de la interfaz luego de un periodo de no haberla utilizado.

- Errores: indica la cantidad de errores cometidos por un usuario, que tan severos son esos errores y que tan fácil se puede recuperar de ellos.

- Satisfacción: indica el nivel de agrado del usuario ante la interfaz.

2.4.1 Importancia de la usabilidad

En la Web, la usabilidad es una condición necesaria para la supervivencia. Si un sitio Web es difícil de usar, los usuarios dejan de utilizarlo. Si la página principal falla en indicar claramente lo que el sitio ofrece y las actividades que pueden ser realizadas, los usuarios dejan de utilizarlo. Si un usuario no sabe que hacer durante la navegación en un sitio Web, abandona dicho sitio. Si la información de un sitio Web es difícil de leer o no responde las preguntas del usuario, este se retira. Hay gran cantidad de sitios Web disponibles, la competencia es alta, retirarse es la primera acción del usuario cuando encuentra dificultades.

En el caso de las intranets, la usabilidad es un asunto de productividad de los empleados. El tiempo innecesario empleado en la intranet o indicando instrucciones ocasiona pérdidas tanto de tiempo como de recursos. [5]

2.4.2 Pruebas de usuario

Existen muchas maneras de estudiar la usabilidad. Una de las maneras mas básicas consiste en las pruebas de usuario la cual posee tres componentes:

- Obtener un grupo de usuarios representativos.
- Solicitar al grupo de usuarios que realicen tareas representativas.
- Monitorear las actividades de los usuarios, donde logran sus objetivos y donde ocurren fallos.

Durante estas es importante someter a prueba a cada usuario de manera individual y dejarlos resolver los problemas que se presenten por sus propios medios. Cualquier intervención externa contaminaría los resultados de las pruebas. Para evaluar la interacción de manera correcta se debe monitorear u observar a los usuarios mientras estos realizan actividades con la interfaz. [5]

Existen diversos mecanismos para recolectar información durante estas pruebas. Uno de ellos consiste en la observación y anotación por parte de expertos, otra en el uso de grabación de audio y video para el registro de la información y, finalmente, el uso de software especializado para el monitoreo y registro las actividades realizadas por los usuarios (escritura, eventos de ratón, entre otros).

En cada etapa de diseño, la interfaz debe ser refinada iterativamente, y una versión mejorada puede ser sometida a prueba. Es importante corregir hasta la más mínima falla, desde errores de escritura hasta distribución de los elementos de la interfaz, ya que influencian en las decisiones de los usuarios. [6]

2.4.3 Caracterización de los usuarios

Las interfaces deben enfrentar una amplia gama de experiencias y expectativas de parte de los usuarios. Para adaptarse a esta variabilidad un método consiste en categorizar los usuarios del sistema en grupos describiendo y modelando el comportamiento de cada uno de ellos, y luego incorporar esta información al proceso de diseño de la interfaz. [7]

La categorización de usuarios se basa en características similares de comportamiento. Se entiende por características aquellas cualidades o rasgos que son medibles, difieren entre los usuarios, pueden modelarse y reflejan el comportamiento del usuario interactuando con el sistema.

El "Task Load Index" desarrollado por la NASA contiene seis dimensiones a considerar para categorizar a los usuarios: demanda mental (por ejemplo memorización), demanda temporal, realización de una tarea, esfuerzo y frustración.

Una vez determinada la categorización se seleccionan usuarios de cada grupo y se observa su comportamiento en relación a una tarea. Los resultados deben mostrar similitudes dentro de un mismo grupo y diferencias con los demás.

A través del análisis de tareas se intenta descubrir el funcionamiento cognitivo del usuario que la realiza. Para ello se describe la ejecución de una serie de tareas tal como son percibidas por el usuario. El análisis se basa entonces en las representaciones mentales a las que el usuario recurre cuando ejecuta una determinada tarea: los procedimientos que usa y las condiciones necesarias para aplicarlos.

Una vez que la tarea del usuario ha sido descrita deben proponerse especificaciones para el diseño de la interfaz basadas en la información obtenida.

En este marco el diseño de la interfaz orientada al usuario constaría de dos etapas. En primer lugar, a un nivel conceptual, deben reconocerse las funciones necesarias, establecerse la secuencia de estas funciones y definir el flujo de

interacción entre usuario y computadora. En segundo lugar, a un nivel de percepción, se involucra la presentación visual al usuario, el diseño de cada pantalla y las partes que la constituyen. Se parte de cuatro preguntas básicas:

- ¿Quién es el usuario? – perfil basado en edad, sexo, actividad ocupacional, educación.

- ¿Dónde? – variable que consideraría varios aspectos, por un lado el contexto institucional donde el usuario actúa, y por otro lado cuestiones relativas a la ubicuidad y portabilidad.

- ¿Cómo? – se analizaría comportamiento, estrategias, procesos cognitivos, experiencia.

- ¿Para hacer qué? – atiende al tema de la funcionalidad.

Una primera caracterización de los usuarios se puede realizar en base a su experiencia en el uso de computadoras:

- Novatos: no están familiarizados en el uso de computadoras en general, pero sí acceden a puntos de información públicos, por ejemplo recorridos virtuales en shoppings, cajeros automáticos, etc.

- Especialistas en computación: conocen de computación pero no de aplicaciones específicas.

- Especialistas en aplicaciones: conocen la temática y la información, pero no los sistemas de computación.

- Especialistas por analogía: experiencias en el uso de sistemas de computación similares, por ejemplo uso de buscadores de Internet.

Esta variable se ocuparía de la interacción humano-computador a un nivel meramente operativo.

Otra tipología también frecuente se basa en la actividad ocupacional del usuario y en el tipo de información que requiere para desempeñarla:

- Niños
- Jóvenes
- Adultos
- Amas de casa
- Obrero
- Profesional

Esta tipología es común en muchas interfaces de Internet, por ejemplo el servidor de correo electrónico de Hotmail. [7]

2.5 Evaluación durante uso activo del sistema

Independientemente de lo bien diseñado y probado a fondo que un sistema se encuentre, su uso activo requiere la atención constante del personal relacionado. Todos los involucrados en brindar soporte a la comunidad de usuarios pueden contribuir a mejoras en el sistema que provean inclusive mayores niveles de servicio. No se puede complacer a todos los usuarios siempre, pero un esfuerzo serio será recompensado por el aprecio y la gratitud de la comunidad e usuarios. [6]

2.5.1 Registro de datos durante el uso activo de los usuarios

La arquitectura de software debería hacer fácil para los administradores del sistema recolectar información relacionada con los patrones de uso del sistema,

velocidad en el desempeño de los usuarios o frecuencias de peticiones de ayuda en línea. El registro de información provee una guía en la adquisición de nuevo hardware, cambios en procedimientos operativos, mejora en el entrenamiento, planes de expansión del sistema, entre otros.

Por ejemplo, si la frecuencia de cada mensaje de error es recopilada, entonces el error de más frecuencia de ocurrencia es el candidato para ser atendido. Sin la información específica registrada, el personal encargado del mantenimiento del sistema no tiene manera de conocer cual de las cientos de posibilidades de situaciones de error es la de mayor problema para los usuarios. De igual manera, se deben examinar los mensajes que nunca aparecen, para verificar la existencia de algún error a nivel de código o si es que los usuarios están evitando el uso de alguna facilidad.

Si el registro de información se encuentra activo para cada acción, cada pantalla de ayuda, cada registro en la base de datos, entonces se pueden realizar cambios en la interfaz que faciliten al usuario el acceso a las funcionalidades de uso frecuente. De igual manera se deben examinar las funcionalidades sin uso o de acceso poco frecuente para comprender las razones por las cuales los usuarios están evitando estas características.

Un gran beneficio de utilizar el registro de datos durante el uso frecuente es la guía que este provee para optimizar el desempeño del sistema y reducir los costos para todos los participantes. [6]

2.6 Factores que afectan la prueba de aplicaciones Web

A continuación se describen, de manera breve, algunos de los factores que tienen influencia en la realización de pruebas de aplicaciones Web: [8]

- Numerosos caminos de uso dentro de la aplicación son posibles: debido al diseño y la naturaleza de las aplicaciones Web es posible que diferentes usuarios

sigan diferentes caminos y todas las combinaciones y permutaciones deben ser probadas para garantizar el funcionamiento correcto.

- Personas con diferentes perfiles y habilidades técnicas puede usar la aplicación: no todas las aplicaciones son de uso intuitivo para todas las personas. Las personas poseen diferentes perfiles y algunas pueden encontrar la aplicación difícil de utilizar. Esto afecta el diseño de las aplicaciones y deben efectuarse pruebas de usabilidad que permitan cubrir este aspecto.

- Las aplicaciones Web evolucionan más rápido que las basadas en tecnologías tradicionales de software y poseen requerimientos de mantenimiento frecuentes: las características de su ciclo de vida incluyen cambios y actualizaciones constantes de contenido, interfaces y funcionalidades ofrecidas a los usuarios.

- Las aplicaciones Web poseen características que no se encuentran presentes en sistemas cliente-servidor y distribuidos: estas incluyen control de sesiones, cookies y la característica del protocolo HTTP de no mantener el estado (luego de entregar una página al cliente cierra la conexión).

- Los usuarios finales pueden acceder la aplicación desde diferentes tipos de Navegadores y localidades geográficas de manera concurrente: deben efectuarse pruebas con los distintos navegadores soportados, debido al hecho de que los navegadores muestran una misma página de manera diferente y también poseen diferentes niveles de soporte para lenguajes de scripting del lado del cliente como Javascript.

- Inclusive en navegadores similares, la aplicación puede desplegarse de manera diferente basándose en diferentes resoluciones de pantalla y configuración del Hardware y Software.

- Velocidades de la red: son importantes las pruebas de rendimiento ya que redes lentas y otros factores pueden causar que los componentes en una página Web sean cargados con retraso y se pueden generar errores.

- Seguridad: es un factor crucial a ser evaluado en caso de que la aplicación capture información privada o importante.

- Plataforma tecnológica donde la aplicación Web es construida: esta crea diferentes fuerzas y debilidades. Diversas herramientas de automatización de pruebas y paquetes están disponibles para diferentes plataformas. Esto puede tener influencia en la estrategia de prueba y en la manera como la ejecución de las pruebas es llevada a cabo.

2.7 Automatización en la Web

La automatización, en el contexto de la informática, se puede definir como el conjunto de métodos que sirven para realizar tareas repetitivas en un computador. Algunos métodos para la automatización de tareas son la programación simple, las macros, los intérpretes y las bombas lógicas. También existen programas específicos dedicados a la automatización.

A nivel de Web, la automatización consiste en reproducir, de forma automática, la interacción de los usuarios con las aplicaciones Web con la finalidad de agilizar y compartir tareas comunes y/o tediosas. Las interacciones de los usuarios pueden ser registradas, mediante grabación de las acciones que realizan, para luego ser reproducidas mediante scripts de automatización u otro mecanismo. Algunos retos que pueden ser identificados, en la automatización Web, son los siguientes [9]:

- **Lidiar con el cambio:** muchas aplicaciones poseen enlaces que son definidos por su posición y el contexto de una página. Las herramientas de automatización no siempre pueden acceder a URLs fijas para acceder a las páginas: La URL de asociada a un enlace puede cambiar entre visitas a una página y una misma URL puede llevar a páginas distintas en diferentes oportunidades. De igual manera muchos sitios Web utilizan campos ocultos en formularios que contienen valores aleatorios que expiran con el tiempo con diversos propósitos. Esto debe ser tomado en cuenta por las herramientas de automatización Web.

- **Razonar acerca del estado:** HTTP es un protocolo sin estado, sin embargo el uso de cookies permite a los servidores almacenar el estado entre peticiones http. Las páginas mostradas a los usuarios pueden depender del contexto de las cookies almacenadas en la computadora del usuario. El RFC 2068 define que las peticiones HTTP GET, HEAD, PUT y DELETE son idempotentes, lo que implica que el efecto secundario de múltiples peticiones idénticas son los mismos para peticiones únicas. Sin embargo no existen estándares para describir los efectos secundarios de otros métodos de petición. A diferencia de un humano que comprende la información en una página Web, una herramienta de automatización no posee información acerca de los efectos secundarios que pueden causar la ejecución de acciones en la página.

- **Superar la opacidad de la información:** un humano puede determinar el éxito o no de una autenticación en una página, para un sistema de automatización detectar condiciones similares a esta no es un asunto trivial y reaccionar ante ellas resulta todo un reto. El contenido de una página se puede encontrar en un formato imposible de comprender por una herramienta de automatización, incluyendo imágenes y mapas de imágenes, contenido generado del lado del cliente, plugins y Applets de Java.

- **Parametrización de scripts de automatización Web:** el poder de una herramienta de automatización Web aumenta si sus scripts soportan parámetros. Su habilidad puede ser incrementada si puede distinguir información utilizada para indicar lo que el usuario desea e identificar al usuario ante la aplicación.

2.7.1 iMacros

iMacros es una herramienta para la extracción de datos, automatización y pruebas en la Web desarrollada por iOpus [10]. En la actualidad posee dos versiones: una comercial soportadas solo por sistemas Windows (Vista, XP,

2000/2003/2008) y otra con funcionalidades básicas como complemento para el navegador Firefox [11].

Esta herramienta fue diseñada para automatizar las tareas más repetitivas de la Web. cualquier actividad puede grabarse y ser reproducida en cualquier momento: llenado de formularios, recordar contraseñas, crear un notificador de webmail, descargar información de otros sitios y más. Las macros desarrolladas pueden ser para uso propio o pueden ser compartidas con otros.

A continuación se describen de manera breve algunos ejemplos de los usos que se le puede dar a iMacros [10].

- **Automatización Web:** permite grabar y reproducir trabajos repetitivos. Esta en la capacidad de llenar formularios y automatizar la descarga y subida de texto, imágenes, archivos y páginas Web. Se puede importar y exportar los datos utilizando archivos CSV y XML, bases de datos o cualquier otra fuente para aplicaciones Web. Incluye soporte para manejo de PDF, capturas de pantalla, simular diferentes user agents y conectarse a servidores Proxy.

- **Extracción de datos:** esta en la capacidad de encontrar texto exacto (precios y descripciones de productos, etc.) e imágenes de sitios Web.

- **Pruebas de la Web:** puede ser utilizada para realizar pruebas funcionales, de desempeño y regresión. iMacros dice ser la única herramienta capaz de automatizar pruebas en los navegadores Internet Explorer y Firefox. También dice ser la única herramienta para realizar pruebas de applets en Java, Flash o Silverlight. El comando STOPWATCH incorporado captura los tiempos de respuesta de la página por cada paso del proceso. Incluye soporte para muchos elementos AJAX.

- **Llenado de formularios y administrador de contraseñas:** elimina la tediosa repetición de verificar los mismos sitios todos los días, recordando contraseñas y llenando formularios. Esta en la capacidad de llenar automáticamente formularios que se extienden por varias páginas y toda la información es

almacenada en archivos de texto plano de fácil lectura y edición. Las contraseñas son almacenadas de manera segura con una encriptación AES de 256 bits.

- **Como componente de software:** añade automatización a las aplicaciones en minutos en vez de semanas o meses. Es una tecnología que tiene la garantía que ha sido probada y depurada por más de cinco años con más de 500.000 instalaciones.

En la Figura #3 se presenta la interfaz de iMacros como complemento para el navegador Firefox y posteriormente, en la Figura #4, se muestra el ejemplo de la edición de una macro.

Figura #3: Ejemplo de la interfaz de iMacros como complemento para Firefox

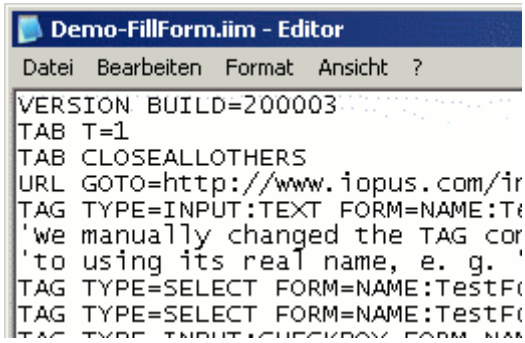

Figura #4: Ejemplo de edición de una macro en iMacros

2.7.1.1 Discusión de iMacros

iMacros es una herramienta muy poderosa que se encuentra disponible, ya sea en su versión comercial paga o como complemento del navegador Firefox. Dicha herramienta, aunque se encuentre basada en grabación y automatización, posee diferencias de enfoque considerables con la herramienta desarrollada en esta investigación.

Entre las principales diferencias podemos encontrar que iMacros funciona como aplicación de escritorio y como complemento de navegadores, la herramienta propuesta funciona como aplicación Web con integración a nivel del cliente por medio de la técnica de page tagging.

Otra diferencia a ser tomada en cuenta es que iMacros se orienta a nivel de manejo de scripts y macros multipropósito, con su sintaxis específica, mientras que la herramienta desarrollada se orienta a la grabación de sesiones de navegación, para brindar información de interés, sin la interacción directa de los usuarios en estos procesos.

Finalmente iMacros, requiere la instalación de un software, a nivel local, para ser utilizada y la idea de la herramienta propuesta es poder realizar las grabación de acciones y eventos a través de la integración con un navegador sin la necesidad de elementos adicionales.

2.7.2 Web Replay 2

Es una herramienta de pruebas automáticas para aplicaciones Web realizada por Emmanuel Kartmann. Ayuda a detectar fallos y regresiones en aplicaciones Web reproduciendo escenarios para someter a prueba la aplicación. Para utilizar esta herramienta, se debe construir un archivo JavaScript de escenario, con un formato particular, que contiene una función que describe el conjunto de acciones que va a llevar a cabo la herramienta.

Se encuentra Basado en Internet Explorer 6.0, comportamientos DHTML y JavaScript. Utiliza un FRAMESET con comportamiento DHTML añadido al evento de carga de la ventana principal para ejecutar código JavaScript que reproduce un escenario. Es implementado sólo a nivel de script: JavaScript ejecutado en Explorer del lado del cliente y JavaScript ejecutado en páginas ASP del lado del servidor. [12]

Características relevantes:

- Reproduce escenarios de un archivo JavaScript.
- Brinda soporte a cada característica que permita Internet Explorer.
- Soporta formularios, campos, enlaces HTML y cualquier acción que JavaScript pueda realizar sobre cualquier elemento.
- Detecta errores HTTP en la aplicación.
- Reproduce escenarios solo para servidores en el mismo dominio.
- No realiza grabaciones automáticas de los escenarios, deben ser escritos en forma manual.

En la Figura #5 se presenta un ejemplo del código que debe contener la función que debe definirse para construir un escenario en la herramienta Web Replay 2.

```
function WebReplayScenario()
{
  switch (gintState)
  {
    case 0:
      // Abre una página Web en localhost sólamente debido a
      // limitaciones de cross-site
"http://perso.wanadoo.fr/replay.grasse/freeware/WebReplay2/WebReplay2/WebReplaySDK.htm#WebReplay_Navigate" target=_blank>WebReplay_Navigate(
        "http://localhost/WebReplay2/WebReplay2Scenario1_step1.asp");
      break;

    case 1:
      // Escribe texto en un campo determinado
"http://perso.wanadoo.fr/replay.grasse/freeware/WebReplay2/WebReplay2/WebReplaySDK.htm#WebReplay_SimulateTextInput"
target=_blank>WebReplay_SimulateTextInput("Text1", "NewValue1");
      break;

    default:
      // Automáticamente se sale del escenario
"http://perso.wanadoo.fr/replay.grasse/freeware/WebReplay2/WebReplay2/WebReplaySDK.htm#WebReplay_SetStateNext"
target=_blank>WebReplay_SetStateNext(-1);
      break;
  }
}
```

Figura #5: Ejemplo de la función a definir para el escenario en Web Replay 2

2.8 Page Tagging

Es una técnica de integración de aplicaciones Web a nivel de cliente, generalmente consiste un pequeño fragmento de código JavaScript ubicado en las páginas del sitio Web. Cada vez que una página es solicitada o se genera algún evento relevante, se ejecuta el código del script y se envía información recopilada de esa página a un servidor remoto. La información capturada y almacenada en el servidor remoto puede ser procesada para monitorear las actividades realizadas en el sitio Web. La transmisión de datos se realiza, típicamente a través de la solicitud de una imagen (pixel) con varios parámetros de interés anexos al query string. [13] [14]

La Figura #6 representa, de manera general, el proceso llevado a cabo para realizar la recopilación de información utilizando la técnica de Page Tagging.

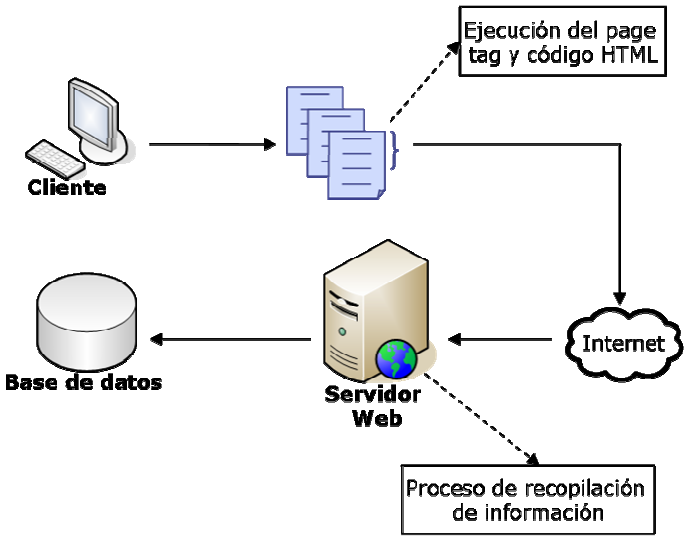

Figura #6: Recopilación de información utilizando Page Tagging

Ventajas

- Reducción de los datos a tratar y almacenar. Los tags capturan la información tan sólo de las páginas que se quieren monitorear.

- El código del script se ejecuta cada vez que la página se carga, por lo que hay mecanismos para vencer o inhibir el caché del browser o de servidores proxies que pudieran estar presentes.

- Se pueden registrar eventos que no se encuentran relacionados con peticiones al servidor Web: interacciones con Flash, llenado parcial y borrado de formularios, eventos del lado del cliente como onclick, onkeyup, entre otros.

- Los tags se pueden integrar fácil y rápidamente, con sólo insertar unas pocas líneas de código en las páginas que se vayan a monitorear.

- Facilidad de integrar nuevos y varios portales: independiente de la plataforma de la aplicación y en general se puede manejar la diversidad de plataformas del usuario (navegador y sistema operativo).

Desventajas

- Esfuerzo en la implementación: Los tags requieren tiempo o software para incluir el script en cada página que se quiere monitorizar. Todas las páginas que serán sujetas a monitoreo, deben contener el tag, lo cual puede requerir de cierto tiempo y esfuerzo en la implementación. No obstante, típicamente los tags se incluyen en plantillas o a través de server-side includes, por lo que el esfuerzo en la implementación puede ser reducido.

- Hay que se cuidadoso en cuanto a la manera en que se coloca el tag para evitar que fallas en la carga o ejecución de éste, afecten la carga de la página. Sin embargo, esto es crítico para aplicaciones como adservers o herramientas para reportes de tráfico, cuyo tag se ejecuta para todos los usuarios. En el caso de la aplicación de pruebas, ésta solo se habilitaría cuando se requiera realizar pruebas.

- Códigos de error: La mayoría de los sitios requieren configuraciones adicionales para permitir a una aplicación de medición de tráfico, recolectar los códigos de error.

- Desactivación de JavaScript: Es posible que JavaScript se encuentre desactivado en los navegadores de los clientes o que se limite el nivel de seguridad. Se estima que un pequeño porcentaje de usuarios desactiva esta característica. Sin embargo, muchas aplicaciones que utilizan esta técnica de recopilación de datos de tráfico, solo utilizan JavaScript para monitorear a los visitantes únicos y establecer cookies o mostrar publicidad, pero la aplicación principal del (website) pudiera seguir funcionando correctamente. Por otra parte, en el caso de la aplicación de pruebas, ésta solo se habilitaría cuando se requiera realizar pruebas.

2.9 Herramientas Tecnológicas para el desarrollo de Aplicaciones Web

En esta sección se describen una serie de tecnologías, tanto del lado del cliente como del servidor, que son comúnmente utilizadas en el desarrollo de aplicaciones Web.

2.9.1 Tecnologías del lado del Cliente

A continuación se presentan las tecnologías del lado del cliente: JavaScript y AJAX, explicando algunas de sus características más relevantes.

2.9.1.1 JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas Web dinámicas.

Una página Web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al usuario, AJAX, validación de formularios, entre otros.

Técnicamente, JavaScript es un lenguaje de programación interpretado, lo cual significa que no requiere ser compilado para su ejecución. Los programas escritos en JavaScript pueden ser ejecutados en prácticamente cualquier navegador Web sin necesidad de procesos intermedios. Aunque no posee ninguna relación directa con el lenguaje Java, posee una sintaxis bastante similar. [15]

El HTML DOM es una recomendación estándar que especifica la manera como se deben acceder y manipular documentos HTML [16], la cual los distintos navegadores implementan con algunas modificaciones originando cierta incompatibilidad entre ellos.

Varias metodologías de desarrollo utilizan JavaScript para interactuar con el DOM, entre ella se encuentran AJAX y DHTML.

La integración de JavaScript con HTML es muy flexible y existen al menos tres maneras de incluir código JavaScript en las páginas Web [13], como a continuación se mencionan:

- Incluir JavaScript en el mismo documento HTML.
- Definir JavaScript en un archivo externo.
- Incluir JavaScript en los elementos XHTML.

2.9.1.2 AJAX

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript asíncrono y XML), es una técnica de desarrollo Web para crear aplicaciones interactivas o RIA (Rich Internet Applications). Éstas se ejecutan en el cliente, es decir, en el navegador de los usuarios y mantiene comunicación asíncrona con el servidor en segundo plano. De esta forma es posible realizar cambios sobre la misma página sin necesidad de recargarla. Esto significa aumentar la interactividad, velocidad y usabilidad en la aplicación.

AJAX es una combinación de varias tecnologías existentes (ver Figura #7):

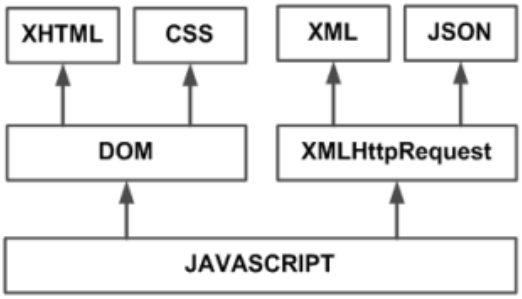

Figura #7: Tecnologías agrupadas bajo el concepto de AJAX

- XHTML (o HTML) y hojas de estilos en cascada (CSS) para el diseño que acompaña a la información.

- Document Object Model (DOM) accedido con un lenguaje de scripting por parte del usuario, especialmente implementaciones ECMAScript como JavaScript y JScript, para mostrar e interactuar dinámicamente con la información presentada.

- El objeto XMLHttpRequest para intercambiar datos asincrónicamente con el servidor Web.

- XML es el formato usado comúnmente para la transferencia de vuelta al servidor, aunque cualquier formato puede funcionar, incluyendo HTML previamente formateado, texto plano, JSON entre otros.

Como el DHTML, AJAX no constituye una tecnología en sí, sino que es un término que engloba a un grupo de éstas que trabajan conjuntamente. [17]

2.9.2 Tecnologías del lado del Servidor Web

En este punto se desarrollan algunos de los aspectos más relevantes del lenguaje de programación PHP y el manejador de bases de datos MySQL.

2.9.2.1 PHP

PHP (acrónimo recursivo que significa PHP Hypertext Pre-processor) es un lenguaje interpretado de propósito general ampliamente usado y que está diseñado especialmente para desarrollo Web y puede ser embebido dentro de código HTML. Generalmente se ejecuta en un servidor Web, tomando el código en PHP como su entrada y creando páginas Web como salida.

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.

Cuando el cliente hace una petición al servidor para que le envíe una página Web, el servidor ejecuta el intérprete de PHP. Éste procesa el script solicitado que generará el contenido de manera dinámica (por ejemplo obteniendo información de una base de datos). El resultado es enviado por el intérprete al servidor, quien a su vez se lo envía al cliente. Mediante extensiones es también posible la generación de archivos PDF, Flash, así como imágenes en diferentes formatos.

PHP maneja el paradigma orientado a objetos permitiendo la definición de clases con su constructor, atributos y métodos; manejando conceptos como herencia, encapsulamiento y modularidad. La utilización de este enfoque permite expresar a las aplicaciones como un conjunto de objetos que colaboran entre ellos para realizar tareas, permitiendo que los programas y módulos sean más fáciles de escribir, mantener y reutilizar.

Permite la conexión a diferentes tipos de servidores de bases de datos tales como MySQL, Postgres, Oracle, ODBC, DB2, Microsoft SQL Server, Firebird y SQLite.

PHP también tiene la capacidad de ser ejecutado en la mayoría de los sistemas operativos, tales como UNIX (y de ese tipo, como Linux o Mac OS X) y Windows, y puede interactuar con los servidores de Web más populares ya que existe en versión CGI, módulo para Apache, e ISAPI. [18]

2.9.2.2 Bases de datos: MySQL

MySQL es un sistema manejador de bases de datos relacional (SMBDR), multihilo y multiusuario. MySQL AB es una compañía comercial de software libre, que distribuye y soporta MySQL, y además lo desarrolla bajo un esquema de licencia dual. Por un lado se tiene la licencia GNU GPL (General Public License GNU) y por otro, aquellas empresas que desean utilizar este sistema en productos privados, pueden comprar la licencia que les permita su uso. [19]

Actualmente MySQL es uno de los sistemas de manejadores de bases de datos más popular del mundo. Su popularidad de uso en aplicación Web está muy

ligada a PHP, que a menudo aparece en combinación con MySQL, sin embargo existen varias librerías que permiten comunicarse con aplicaciones escritas en diversos lenguajes de programación.

Interioridades y portabilidad

- Escrito en C y en C++
- Funciona en diferentes plataformas. Funciona en plataformas como AIX, BSD, FreeBSD, HPUX, GNU/Linux, Mac OS X, NetBSD, Windows 95/98/NT/2000/XP/Vista y otras versiones de Windows, entre otros.
- Uso completo de multi-hilos mediante hilos del kernel. Pueden usarse fácilmente multiple CPUs si están disponibles.
- Múltiples motores de almacenamiento (MyISAM, Merge, InnoDB, Memory/heap, entre otros), permitiendo al usuario escoger la que sea más adecuada para cada tabla de la base de datos.

Tipos de datos

- Diversos tipos de datos: enteros con/sin signo de 1, 2, 3, 4, y 8 bytes de longitud, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET, ENUM, y tipos espaciales OpenGIS.
- Registros de longitud fija y longitud variable.

Sentencias y funciones

- Soporte completo para operadores y funciones en las cláusulas de consultas SELECT y WHERE.
- Soporte completo para las cláusulas SQL GROUP BY y ORDER BY.
- Puede mezclar tablas de distintas bases de datos en la misma consulta.

Seguridad

- Un sistema de privilegios y contraseñas que es muy flexible y seguro, y que permite verificación basada en el host. Las contraseñas son seguras porque todo el tráfico de contraseñas está encriptado cuando se conecta con un servidor.

2.10 Metodología de desarrollo

Para la construcción de la herramienta monitoreo de eventos y actividades de los usuarios en aplicaciones Web será utilizada una metodología Ad-hoc, aplicando un modelo de desarrollo en cascada o ciclo de vida del software, llevando a cabo una serie de pasos en cada etapa. La figura #8 presenta un esquema de las etapas del modelo en cascada a ser aplicado.

De igual manera serán utilizados Unified Modeling Language (UML) Y Web Application Extension for UML (WAE) como formalismo para modelar y documentar las distintas vistas de la aplicación.

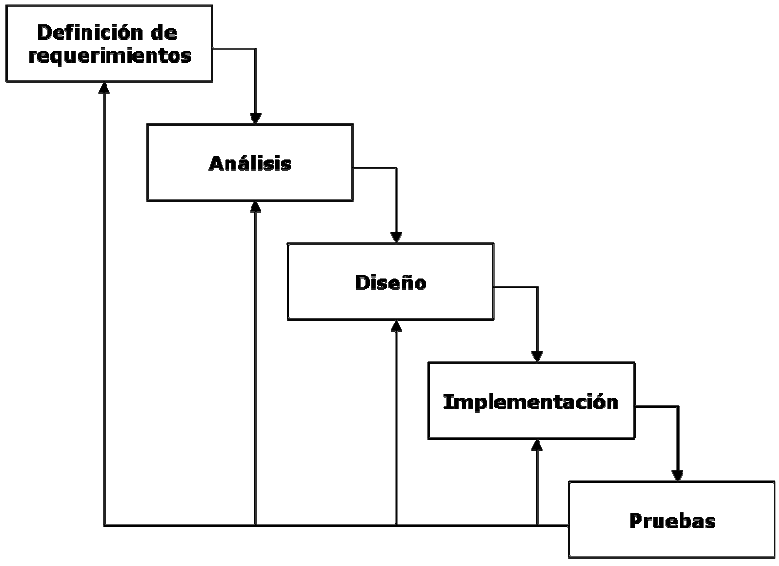

Figura #8: Etapas del modelo en cascada a ser aplicado

- **Definición de requerimientos:** En esta etapa se establecerán los servicios del sistema, el alcance y los objetivos. Se estudian las alternativas de solución para utilizar una o varias de ellas.

- **Análisis:** Una vez que se han recopilado todos los requerimientos, objetivos y alcances, se procederá a la elaboración de un conjunto de especificaciones que describan la funcionalidad del sistema.

- **Diseño:** El diseño de la aplicación contemplará los aspectos físicos del sistema, considerando las características tecnológicas del entorno específico. Identifica y describe las abstracciones fundamentales del sistema software y sus relaciones.

- **Implementación:** A partir de las especificaciones del diseño, debe obtenerse una versión totalmente funcional del sistema. Debe ajustarse el proceso de codificación a las condiciones determinadas en las etapas de análisis y diseño.

- **Pruebas (Funcionamiento y mantenimiento):** Una vez obtenida una versión final resultante en la etapa de implementación, es necesario realizar las pruebas la aplicación en un ambiente real de producción, con la finalidad de medir tolerancia a fallas y optimizar los procesos que así lo requieran.

El resultado de cada fase consiste en uno o más documentos aprobados. La siguiente fase no debe comenzar hasta que la fase previa haya finalizado. En la práctica estas etapas se superponen y proporcionan información unas a las otras. Durante el diseño se identifican problemas en los requerimientos; durante el diseño del código se encuentra problemas y así sucesivamente. El proceso de desarrollo de software no es un modelo lineal, sino que implica una serie de iteraciones de las actividades de desarrollo. Durante la etapa final del ciclo de vida, el software se pone en funcionamiento. Se descubren errores y omisiones de los requerimientos originales del software. Los errores de programación y diseño emergen y se identifica la necesidad de una nueva funcionalidad. Por lo tanto el sistema debe evolucionar para mantenerse útil. Hacer estos cambios puede implicar repetir etapas previas del proceso.

Las ventajas del modelo en cascada son que la documentación se produce en cada fase y esta tiene relación con otros modelos de procesos de ingeniería. Su principal problema es su inflexibilidad para dividir el proyecto en diferentes etapas. Se deben hacer compromisos en etapas iniciales, lo que hace difícil responder a los cambios en los requerimientos iniciales. El modelo en cascada solo es recomendable utilizarlo cuando los requerimientos sean bien comprendidos y sea improbable que cambien radicalmente durante el desarrollo del sistema. [2]

Capítulo III Marco Aplicativo

En este capítulo se presenta una adaptación del proceso de desarrollo de software al caso particular de estudio, basado en el modelo en cascada, dentro del contexto de una metodología Ad-hoc, para la construcción de la herramienta de monitoreo de actividades y eventos de usuarios en aplicaciones Web. En este sentido, se describen cada una de las fases del método utilizado para el desarrollo la herramienta.

Fue aplicado un modelo de desarrollo en cascada ya que es sencillo de implementar y resulta apropiado para sistemas pequeños donde los requerimientos iniciales son claros y no poseen tendencia de cambiar radicalmente a lo largo del tiempo, caso similar al de la herramienta propuesta.

Se utilizó una metodología Ad-hoc ya que permite adaptar y personalizar el modelo de desarrollo aplicado de acuerdo con las características específicas de la aplicación a construir.

3.1 Fase I: Definición de requerimientos

En esta fase se definen los requerimientos, funcionales y no funcionales, que deben ser tomados en cuenta para el desarrollo de la herramienta propuesta por esta investigación.

3.1.1 Requerimientos funcionales

Se define como principal requerimiento funcional, a ser considerado en el desarrollo de la herramienta de monitoreo propuesta, realizar la grabación de eventos y acciones de usuarios en aplicaciones Web.

Para cumplir con es se deben llevar a cabo las siguientes funcionalidades:

- Capturar y registrar eventos relevantes generados dentro de la aplicación sometida a prueba de manera automática. Del conjunto total de eventos que pueden ser generados, se han seleccionado, como de interés para su captura y registro, los siguientes:
    a. Envío y limpieza de formularios.
    b. Click del ratón sobre algún enlace o elemento de interés dentro de la página.
    c. Eventos relacionados teclado, tales como presión y combinación de teclas.
    d. Errores generados en la aplicación a nivel de Scripts.
    e. Peticiones AJAX.
    f. Carga de imágenes.
    g. Estado de enlaces en la página.

- Reproducir automáticamente acciones registradas de un usuario en la aplicación Web.

- Generar reportes de utilidad a partir de grabaciones realizadas.

3.1.2 Requerimientos no funcionales

La herramienta o aplicación a desarrollar, contempla dos interfaces diferentes de acuerdo con el actor con el cual interactúe. La herramienta será utilizada tanto por actores humanos como por aquellas aplicaciones Web sometidas a prueba.

El módulo de la herramienta que administra la interfaz encargada de la interacción con las aplicaciones Web sometidas a prueba debe cumplir con lo siguiente:

- Una adaptación del mecanismo de page tags, utilizando JavaScript como lenguaje para la integración entre las aplicaciones Web sometidas a prueba y la

herramienta a desarrollar buscando simpleza en el proceso de comunicación e independencia de plataforma tecnológica de las aplicaciones involucradas.

El módulo encargado de la interfaz con los usuarios humanos, deberá cumplir con los siguientes requerimientos no funcionales:

- Utilización de tecnologías que cumplan con recomendaciones propuestas por la W3C.

- Utilización de PHP como tecnología del lado del servidor.

3.2 Fase II: Análisis

Se enfoca en el análisis desde la perspectiva del problema, con la finalidad de modelar el proceso que se desea automatizar de manera objetiva, permitiendo posteriormente realizar el diseño de la solución. En esta etapa se define una especificación del sistema y se elaboran los modelos de casos de uso y objeto del dominio y del análisis.

3.2.1 Modelo de Casos de Uso

Especifica la funcionalidad y el comportamiento del sistema mediante la interacción de otros usuarios y/o sistemas.

La Figura #9 presenta el modelo de casos de uso elaborado para modelar las funcionalidades y el comportamiento de la herramienta de monitoreo de eventos propuesta por esta investigación.

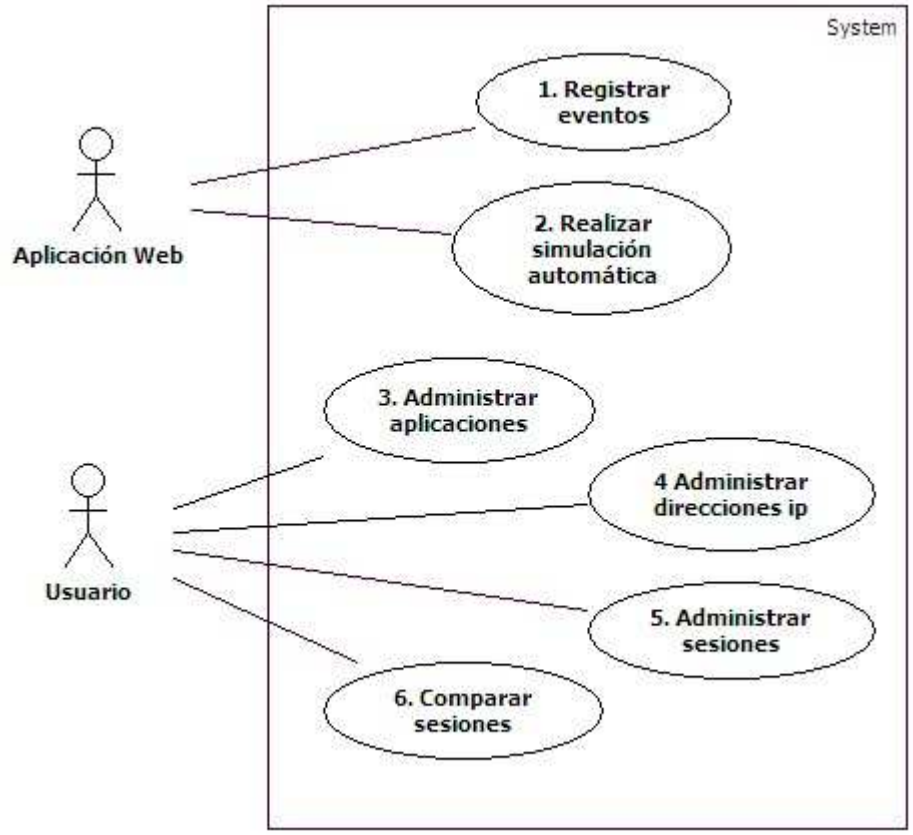

Figura #9: Modelo de Casos de Uso

Identificación de los actores:

- **Aplicación Web:** aquella aplicación en la cual deba ser monitoreado los eventos de los usuarios que interactúen con ella.

- **Usuario:** aquella persona que puede ingresar a la herramienta para observar y analizar la información y los resultados de las de las grabaciones realizadas.

3.2.1.2 Descripción de los Casos de Uso

A continuación se describen los casos de uso del modelo presentado previamente.

Nombre:	Registrar eventos
Valor:	1
Resumen:	La herramienta debe estar en la capacidad de registrar diferentes eventos para permitir la posterior reproducción automática de acciones
Dependencias:	Este caso de uso no posee dependencias.
Actor:	Aplicación Web.
Precondición:	Para que puedan ser registrados los eventos, la aplicación Web que envía los datos debe estar autorizada para dicha acción dentro de la herramienta y debe incluir el script que provee la herramienta en sus páginas.
Descripción:	Se deben registrar: eventos de envío y borrados de formularios, click a elementos relevantes, errores generados a nivel de script, errores en la carga de imágenes, eventos de teclado, peticiones AJAX, algunos tiempos de respuesta, disponibilidad de enlaces (internos y externos por ahora).
Secuencia de pasos:	➢ La aplicación Web genera un evento que amerite ser registrado. ➢ El evento es capturado por el script incluido. ➢ Se recopila la información asociada al evento y se envía a la herramienta. ➢ La información enviada es registrada.
Poscondición:	Los eventos son registrados en la base de datos para su posterior uso.

Tabla #1: Descripción caso de uso Registrar eventos

Nombre:	Realizar simulación automática
Valor:	2
Resumen:	La herramienta debe estar en capacidad de generar de forma dinámica el código que reproduzca una serie de acciones, grabadas previamente, en la aplicación Web sometida a prueba.
Dependencias:	Este caso de uso no posee dependencias.
Actor:	Aplicación Web.
Precondición:	La aplicación Web debe estar registrada en la herramienta e incluir el script provisto por esta.
Descripción:	El usuario indica los parámetros y el conjunto de acciones que formarán parte de la simulación. La herramienta obtiene la información correspondiente y genera el código que va ser ejecutado por la aplicación Web para la reproducción automática de las acciones que conforman la simulación.
Secuencia de pasos:	➢ A través de la aplicación Web, son indicados los parámetros y el conjunto de acciones a ser simuladas. ➢ Los parámetros e información indicados son enviados a la herramienta. ➢ La herramienta genera el código correspondiente al conjunto de acciones a ser automatizadas durante la simulación.
Poscondición:	El código encargado de realizar la simulación automática es generado por la herramienta, listo para ser ejecutado por la aplicación.

Tabla #2: Descripción caso de uso Realizar simulación automática

Nombre:	Administrar aplicaciones
Valor:	3
Resumen:	El usuario esta en la capacidad de administrar el conjunto de aplicaciones Web, de las cuales se registra información de sus usuarios, asociadas a la herramienta
Dependencias:	Este caso de uso no posee dependencias
Actor:	Usuario
Precondición:	El usuario debe estar autenticado en la herramienta
Descripción:	El usuario puede registrar nuevas aplicaciones y eliminar las ya existentes en la herramienta.
Secuencia de pasos:	➢ Se le muestra al usuario una interfaz que contiene una serie de campos correspondientes a la información de la nueva aplicación a registrar y una lista de las aplicaciones existentes. ➢ El usuario llena los campos y confirma el registro de una nueva aplicación o decide eliminar alguna de las ya existentes.
Poscondición:	Este caso de uso no posee poscondición.

Tabla #3: Descripción caso de uso Registrar aplicaciones

Nombre:	Administrar direcciones ip
Valor:	4
Resumen:	El usuario debe estar en la capacidad de administrar la información de las diferentes direcciones ip asociadas a las aplicaciones Web registradas en la herramienta.
Dependencias:	Este caso de uso no posee dependencias
Actor:	Usuario
Precondición:	El usuario debe estar autenticado en la herramienta
Descripción:	El usuario puede asociar nuevas direcciones ip a las aplicaciones registradas en la herramienta y eliminar las direcciones ya existentes.
Secuencia de pasos:	➢ Se le muestra al usuario una interfaz que contiene un listado con las diferentes direcciones ip asociadas a una aplicación en particular y la opción de eliminarlas o crear una nueva a través de un formulario. ➢ El usuario llena los campos y confirma la asociación de una nueva dirección ip a la aplicación o decide eliminar alguna de las direcciones ya existentes.
Poscondición:	Este caso de uso no posee poscondición.

Tabla #4: Descripción caso de uso Administrar sesiones

Nombre:	Administrar sesiones
Valor:	5
Resumen:	El usuario debe estar en la capacidad de administrar la información de las diferentes sesiones almacenada en la herramienta.
Dependencias:	Este caso de uso no posee dependencias
Actor:	Usuario
Precondición:	El usuario debe estar autenticado en la herramienta
Descripción:	El usuario puede ver y eliminar la información correspondiente a las sesiones de grabación.
Secuencia de pasos:	➢ Se le muestra al usuario una interfaz con las diferentes sesiones existentes. ➢ El usuario puede observar el contenido de las sesiones almacenadas y eliminarlas si así lo desea.
Poscondición:	Este caso de uso no posee poscondición.

Tabla #5: Descripción caso de uso Administrar sesiones

Nombre:	Comparar sesiones
Valor:	6
Resumen:	El usuario puede observar y analizar los resultados grabaciones de eventos y acciones de manera organizada
Dependencias:	Este caso de uso no posee dependencias
Actor:	Usuario
Precondición:	El usuario debe estar autenticado en la herramienta
Descripción:	Le son presentados al usuario los resultados de las grabaciones realizadas por la herramienta con la finalidad de que pueda analizar y comparar diferentes sesiones en distintos escenarios para llegar a conclusiones útiles.
Secuencia de pasos:	➢ Se le muestra al usuario una interfaz con las sesiones existentes de las distintas aplicaciones Web sometidas a prueba. ➢ El usuario selecciona dos sesiones que desea ver y comparar entre si. ➢ Se despliega el contenido de ambas sesiones estableciendo similitudes y diferencias entre ellas.
Poscondición:	Los resultados de las de las grabaciones son presentados al usuario para su observación y análisis.

Tabla #6: Descripción caso de uso Comparar sesiones

3.2.2 Modelo Objeto del Dominio (MOD)

Contiene los objetos pertenecientes al dominio de la aplicación, con sus respectivas descripciones y relaciones. La Figura #10 presenta el modelo objeto del dominio elaborado para esta investigación.

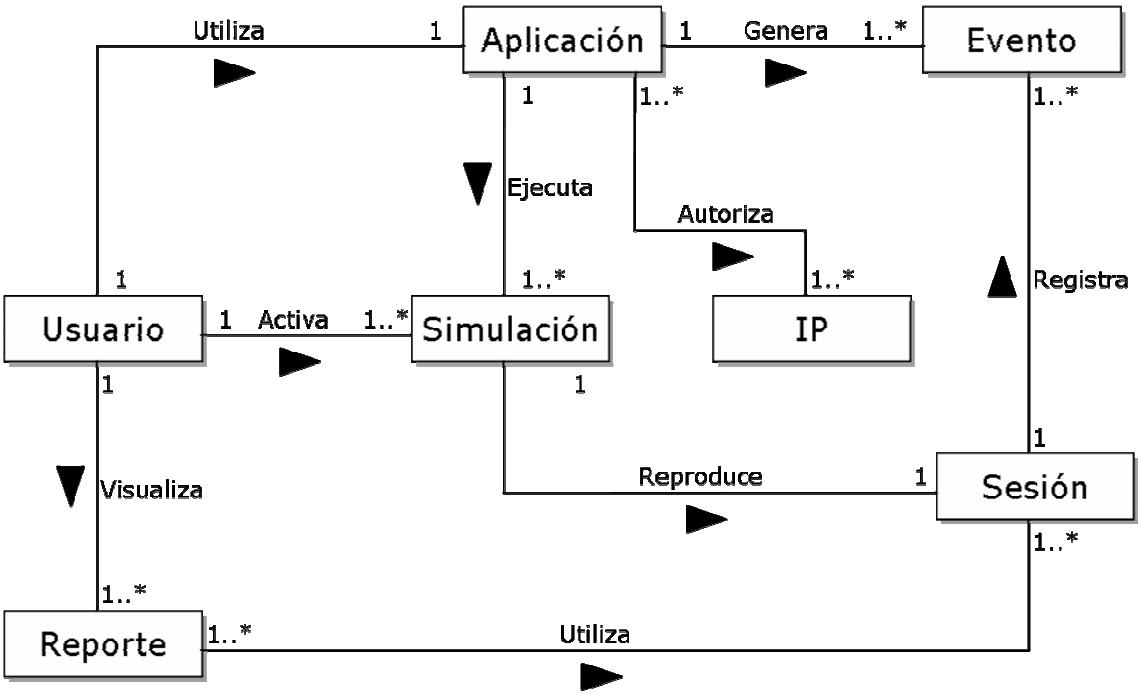

Figura #10: Modelo de objeto del dominio

Los objetos identificados en el dominio son:

- **Aplicación:** objeto que representa la aplicación Web de la cual van a ser registrados eventos. Genera eventos ante la interacción con un usuario o a través de la ejecución de una simulación.

- **IP:** dirección IP asociada a una aplicación, la cual autoriza el acceso a las funcionalidades de la herramienta.

- **Usuario:** representa a la persona que se encuentra en la capacidad de de efectuar interactuar con la aplicación Web para generar eventos, activar la

simulación en la aplicación Web y visualizar los resultados de grabaciones previamente realizadas.

- **Simulación:** representa la reproducción automática, en la aplicación, de los eventos registrados en una sesión determinada.

- **Evento:** acción de un usuario o comportamiento relevante que pueda ser generado y detectado en la aplicación.

- **Sesión:** agrupación de un conjunto de eventos bajo el criterio de sesión de navegación. Es utilizado para la generación de reportes y en las simulaciones.

- **Reporte:** presentación útil de los resultados de las grabaciones realizadas cuya información es generada a partir de sesiones.

3.2.3 Modelo Objeto del Análisis (MOA)

Genera una visualización de los objetos que conforman el espacio de información de la aplicación a desarrollar. Clasifica los objetos como interfaz, control o entidad. A continuación se muestran los diferentes MOA desarrollados que modelan cada una de las funcionalidades principales de la herramienta propuesta.

3.2.3.1 MOA registrar eventos (ver Figura #11)

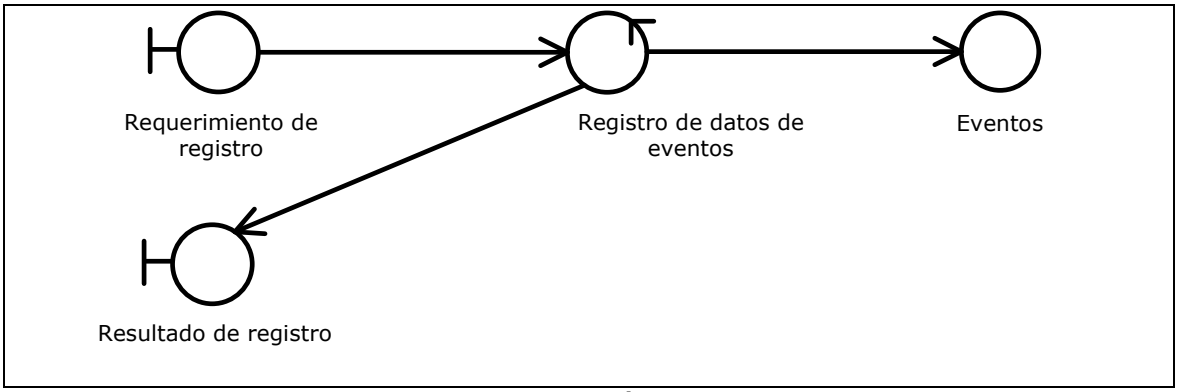

Figura #11: Modelo objeto del análisis para registrar eventos

3.2.3.2 MOA realizar simulación automática (ver Figura #12)

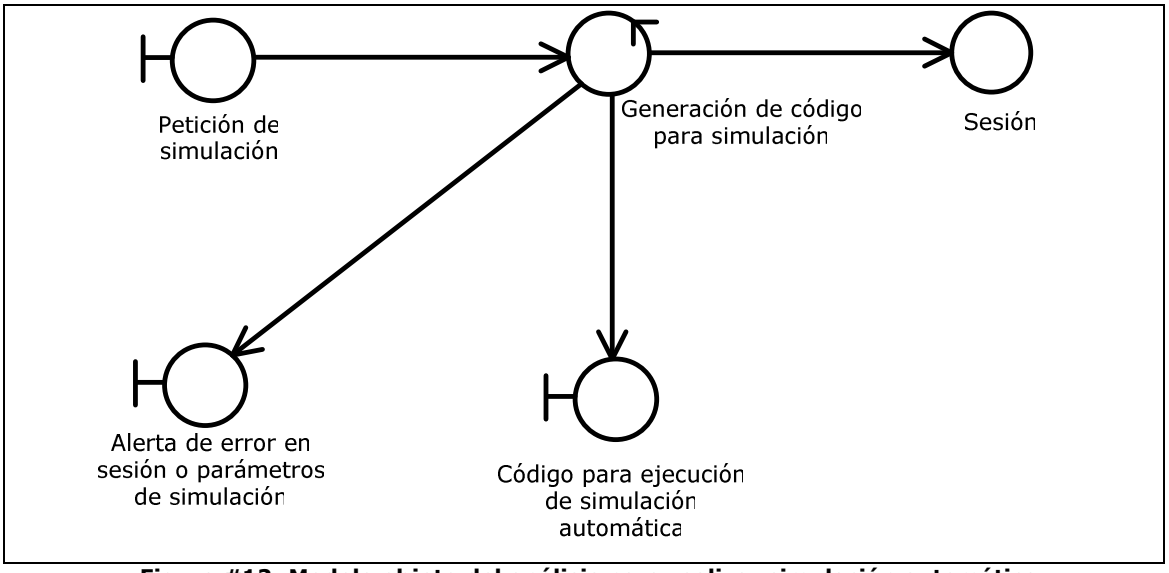

Figura #12: Modelo objeto del análisis para realizar simulación automática

3.2.3.3 MOA administrar aplicaciones (ver Figura #13)

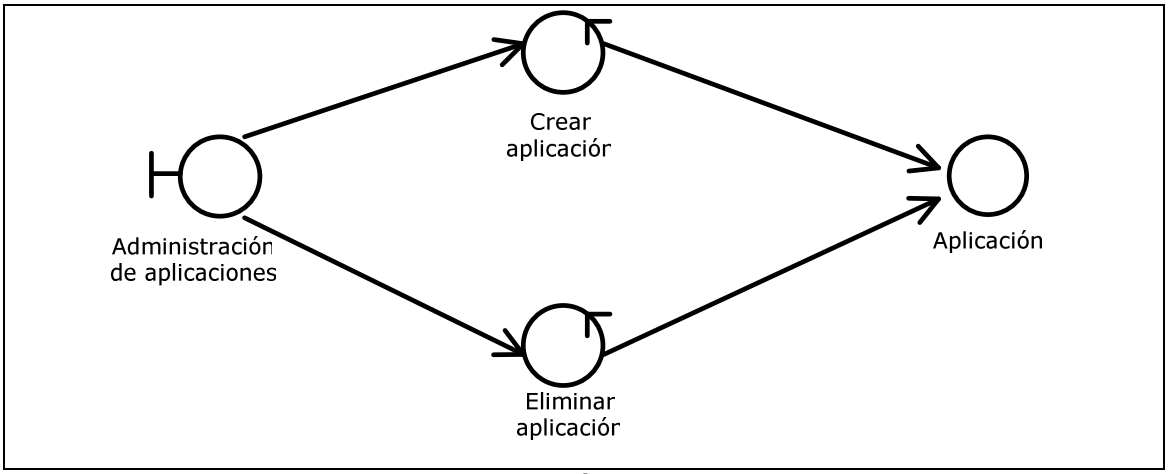

Figura #13: Modelo objeto del análisis para administrar aplicaciones

3.2.3.4 MOA administrar direcciones IP (ver Figura #14)

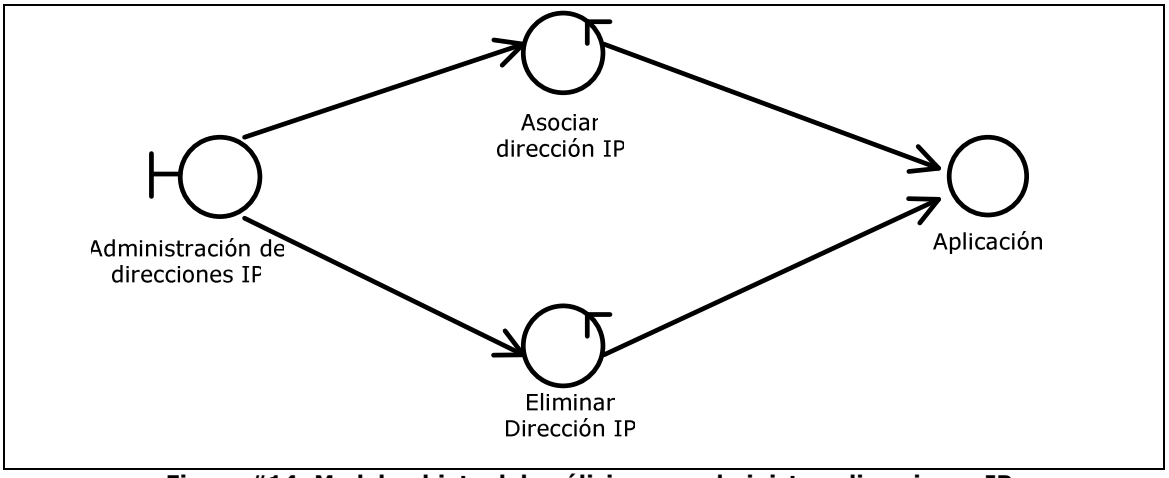

Figura #14: Modelo objeto del análisis para administrar direcciones IP

3.2.3.5 MOA administrar sesiones (ver Figura #15)

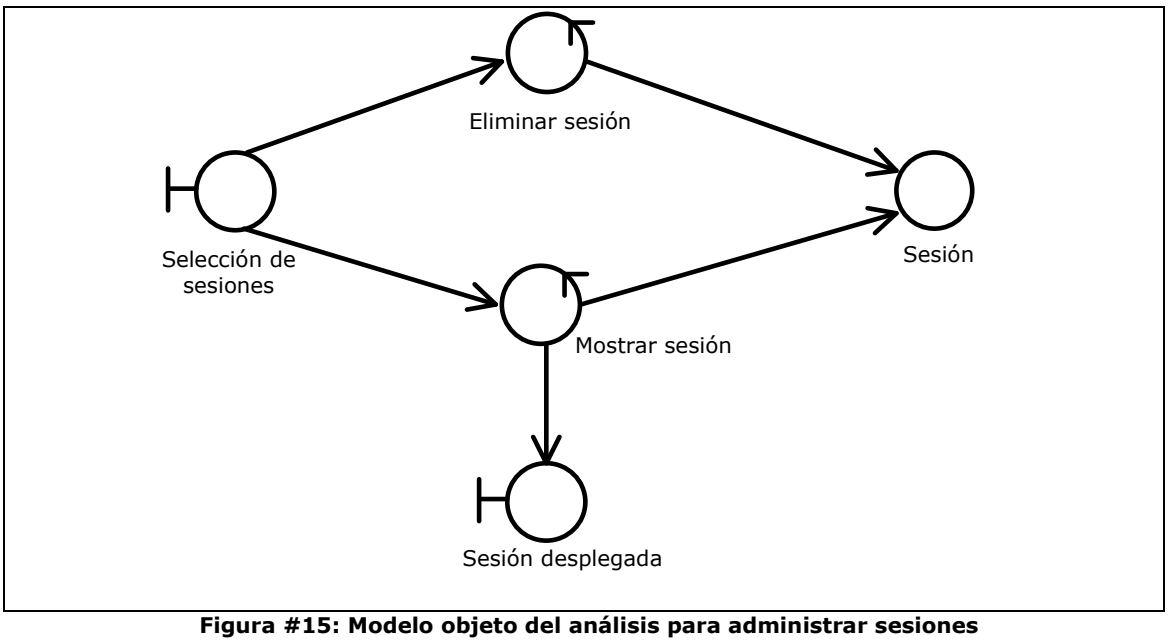

Figura #15: Modelo objeto del análisis para administrar sesiones

3.2.3.6 MOA comparar sesiones (ver Figura #16)

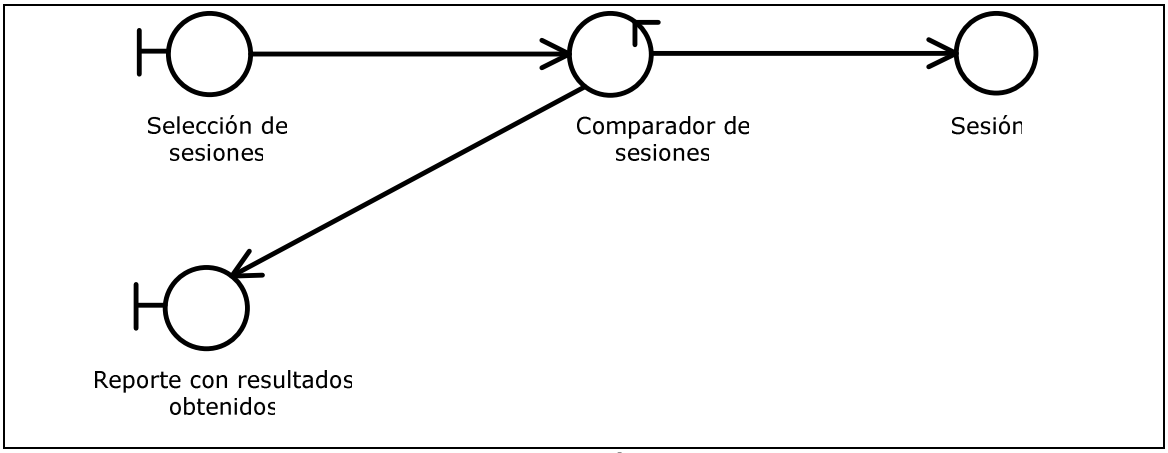

Figura #16: Modelo objeto del análisis para comparar sesiones

3.3 Fase III: Diseño

La fase de diseño se encuentra orientada al dominio de la solución y tiene como entrada los artefactos productos de la fase de análisis y la arquitectura donde se va a desarrollar la aplicación. Su objetivo consiste en diseñar lo relacionado con la implementación refinando los artefactos de análisis y definiendo los elementos de software que conforman la solución. En esta fase se realizan el diagrama de clases persistentes, diagrama de base de datos, Web Application Extension for UML (WAE) y un diagrama de secuencia describiendo el comportamiento de la aplicación en lo referente al registro de eventos.

3.3.1 Diagrama de clases persistentes

Permite modelar conceptos relacionados con el dominio de la aplicación, identificando las clases que se encuentran en el sistema así como también sus respectivas relaciones estructurales y de herencia. Una clase describe un conjunto de objetos que comparte los mismos atributos, operaciones y relaciones.

A continuación se presenta la Figura #17 con el diagrama de clases persistentes elaborado.

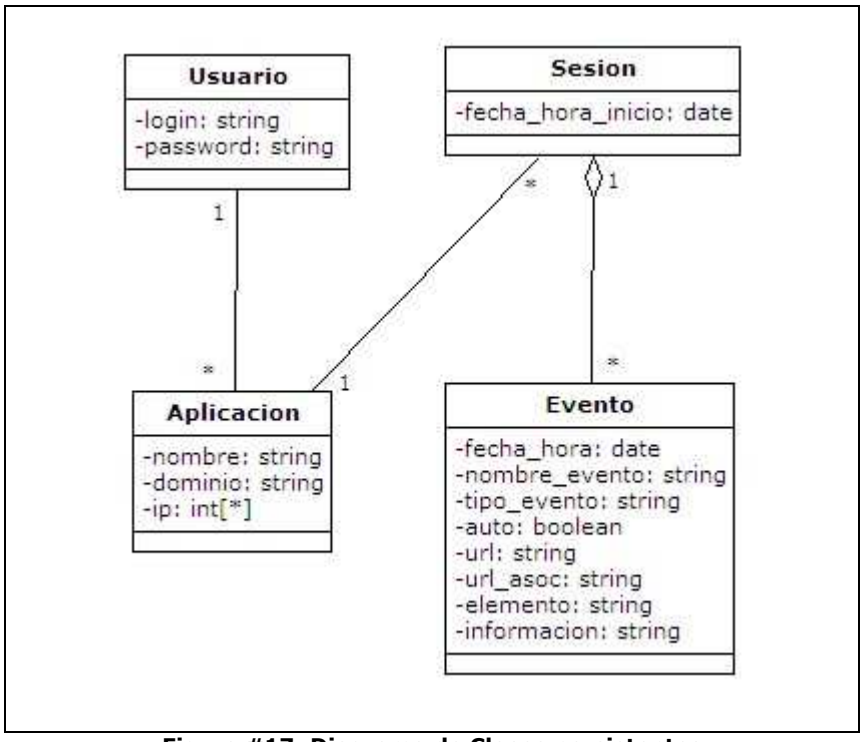

Figura #17: Diagrama de Clases persistentes

3.3.2 Diagrama de base de datos

Permite visualizar la estructura de la base de datos que va a ser utilizada por la aplicación desarrollada. Este diagrama es elaborado tomando en cuenta las clases, objetos y relaciones definidos previamente así como también procesos como la normalización.

El diagrama de la base de datos a ser usada en la implementación de la herramienta propuesta por esta investigación puede ser visto en la Figura #18.

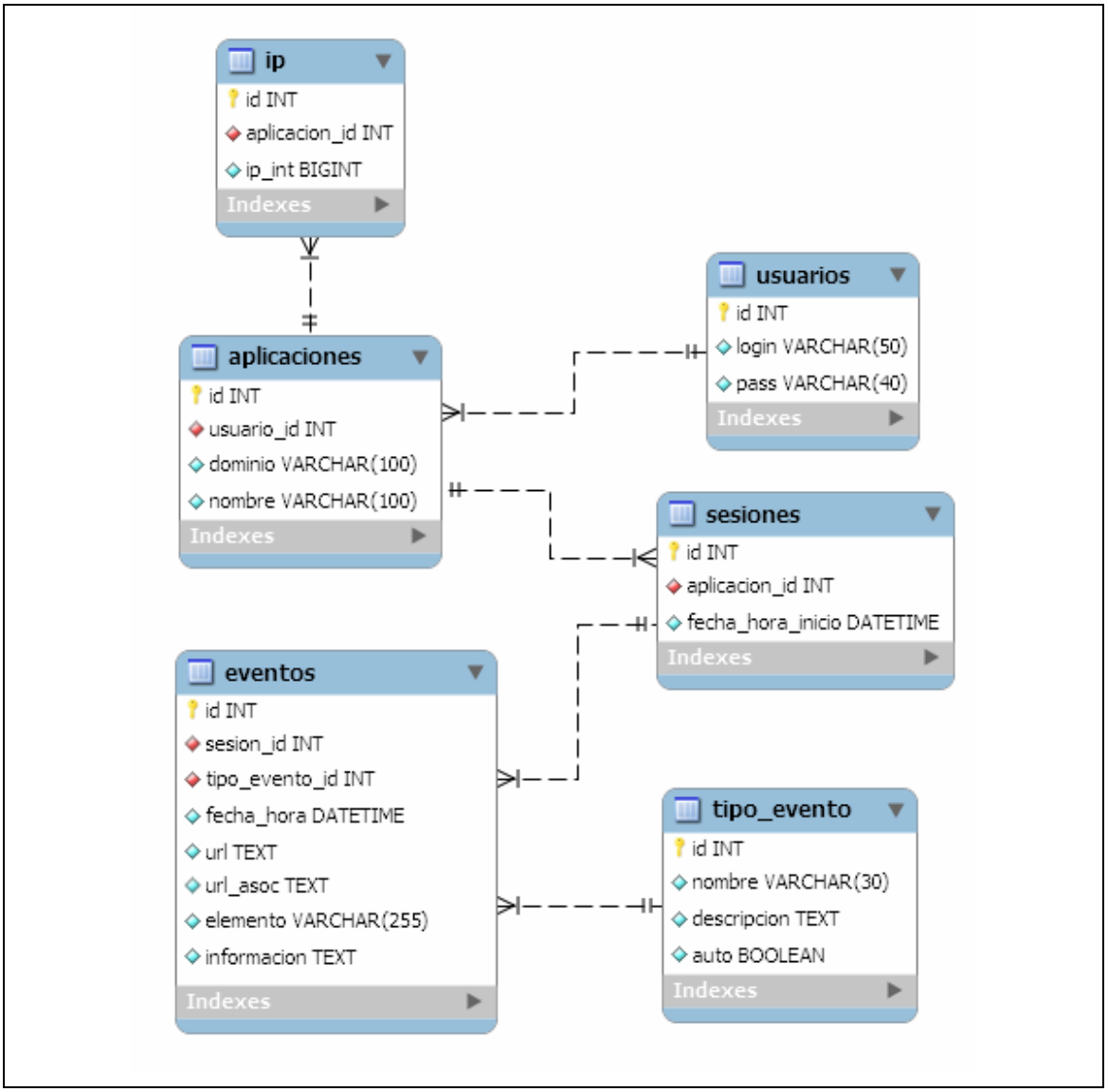

Figura #18: Diagrama de Base de Datos

3.3.3 Web Application Extension (WAE)

Es una extensión de la notación UML con semánticas y restricciones que permite modelar elementos de arquitectura específicos de la Web, aplicaciones como parte de un sistema completo y la lógica de negocio que debe estar reflejada en la aplicación. A continuación, pueden ser observados el conjunto de diagramas WAE que modelan la aplicación o herramienta propuesta por esta investigación.

3.3.3.1 WAE registro de eventos y simulación automática

(ver Figura #19)

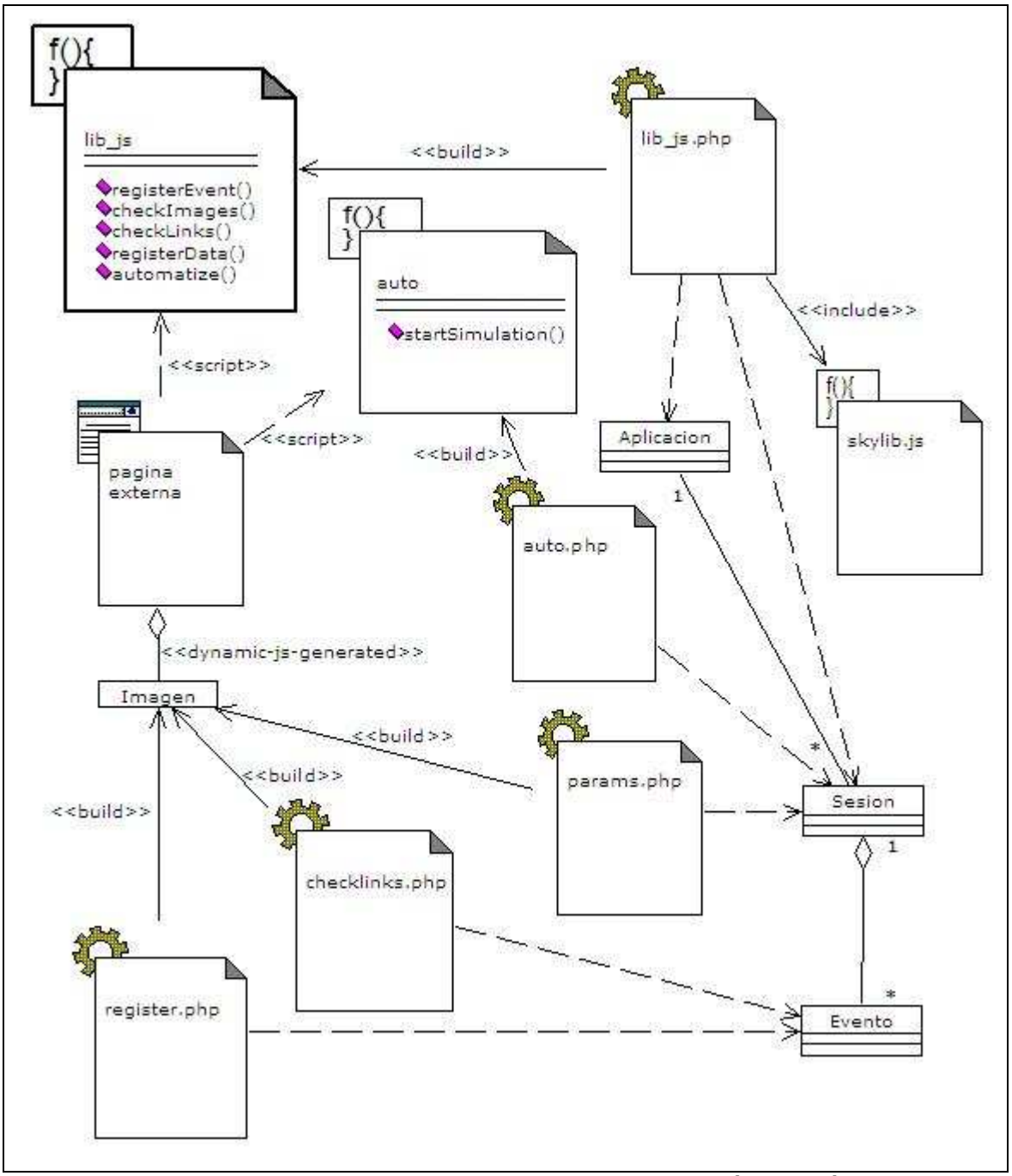

Figura #19: Diagrama WAE registro de eventos y simulación automática

3.3.3.2 WAE administrar aplicaciones (ver Figura #20)

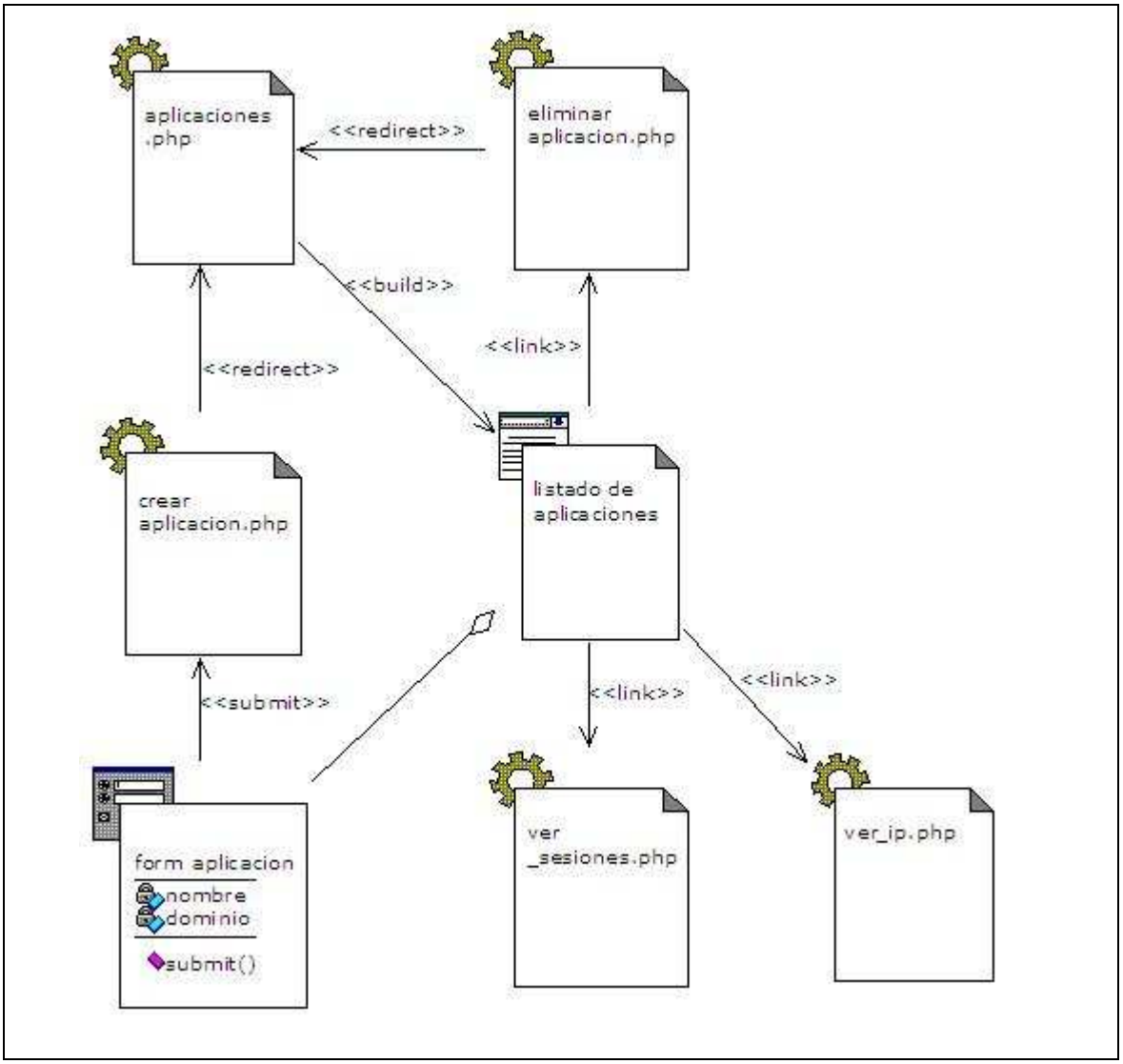

Figura #20: Diagrama WAE administrar aplicaciones

3.3.3.3 WAE administrar direcciones IP (ver Figura #21)

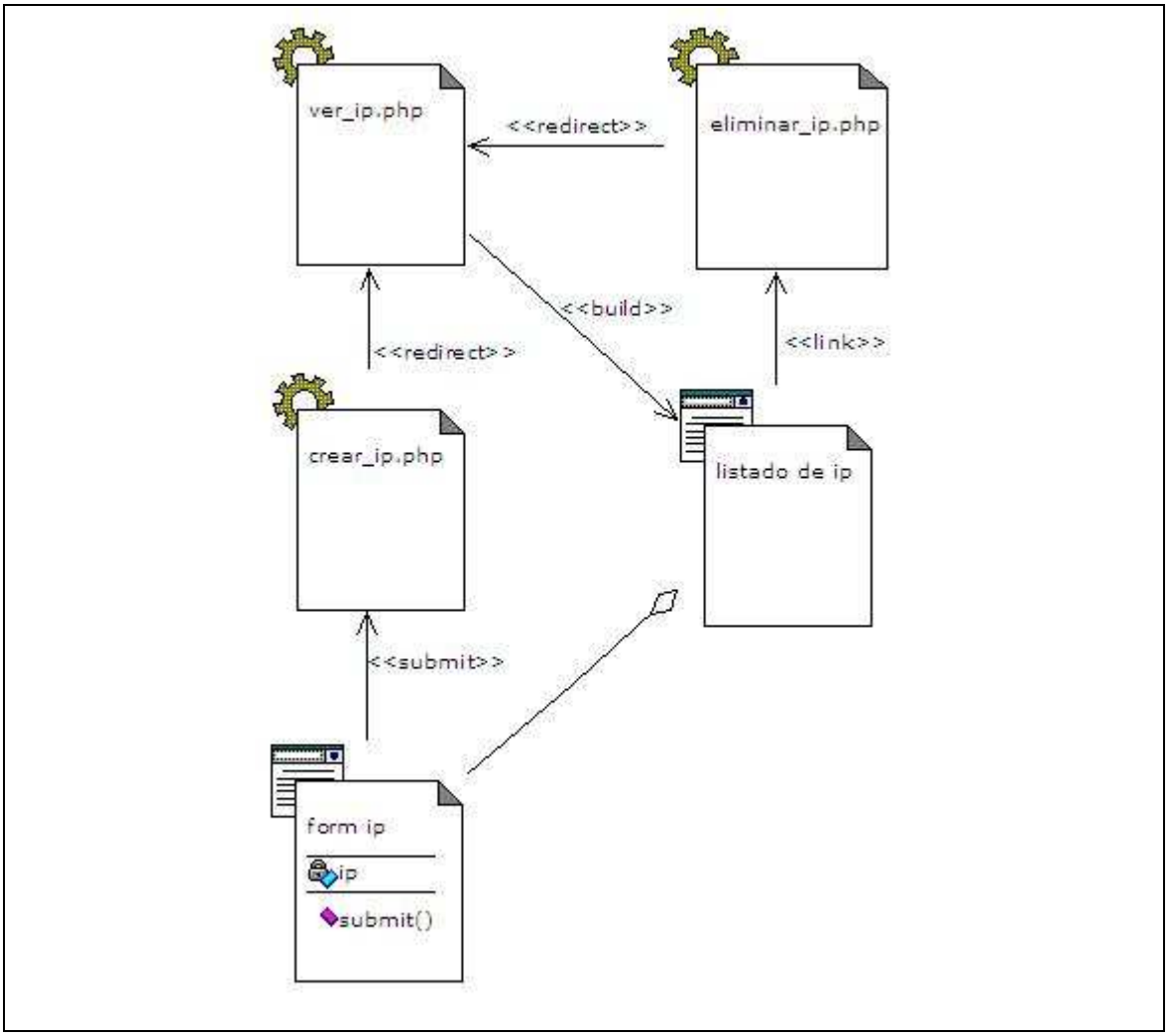

Figura #21: Diagrama WAE administrar direcciones IP

3.3.3.4 WAE administrar sesiones (ver Figura #22)

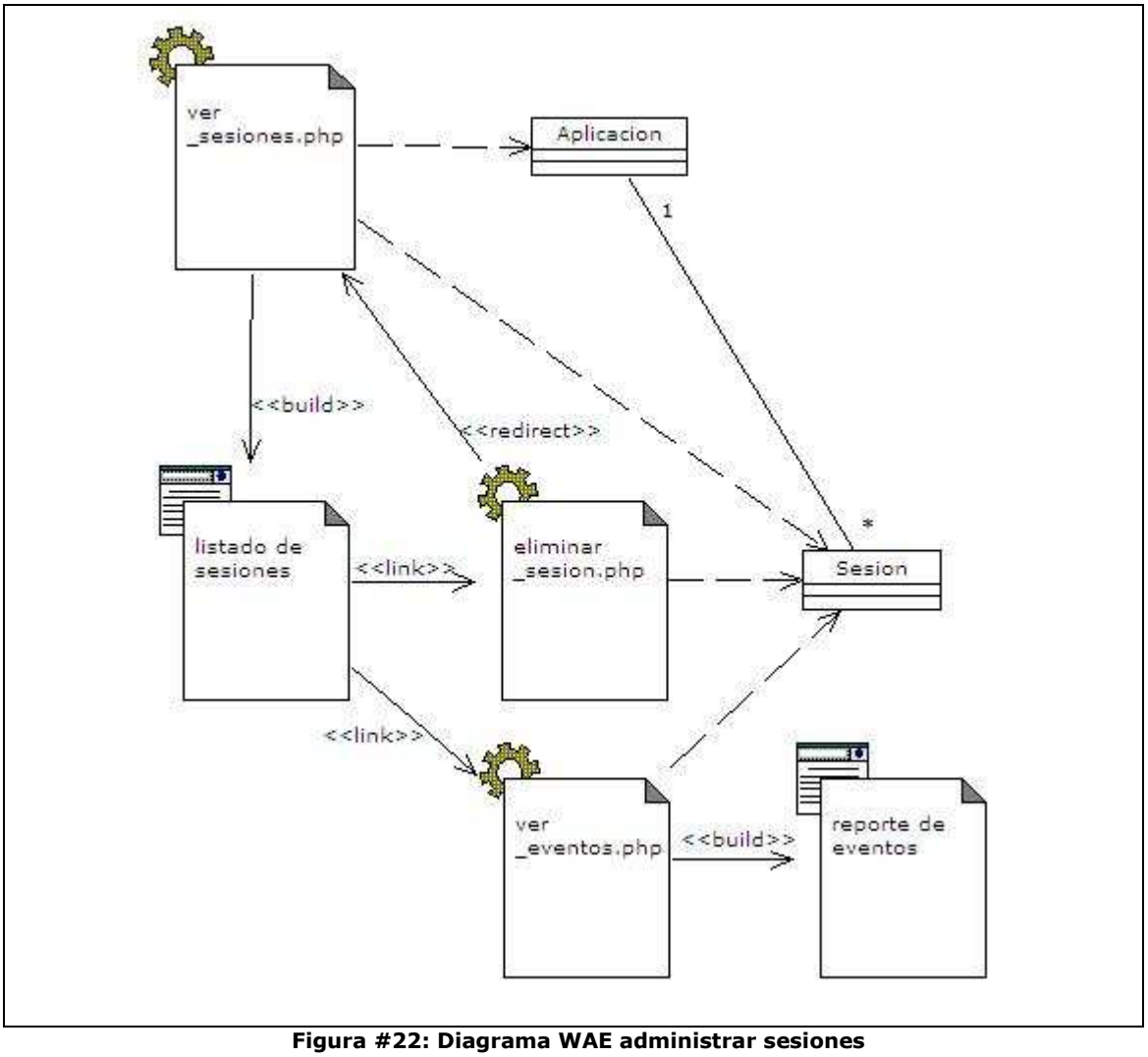

Figura #22: Diagrama WAE administrar sesiones

3.3.3.5 WAE comparar sesiones (ver Figura #23)

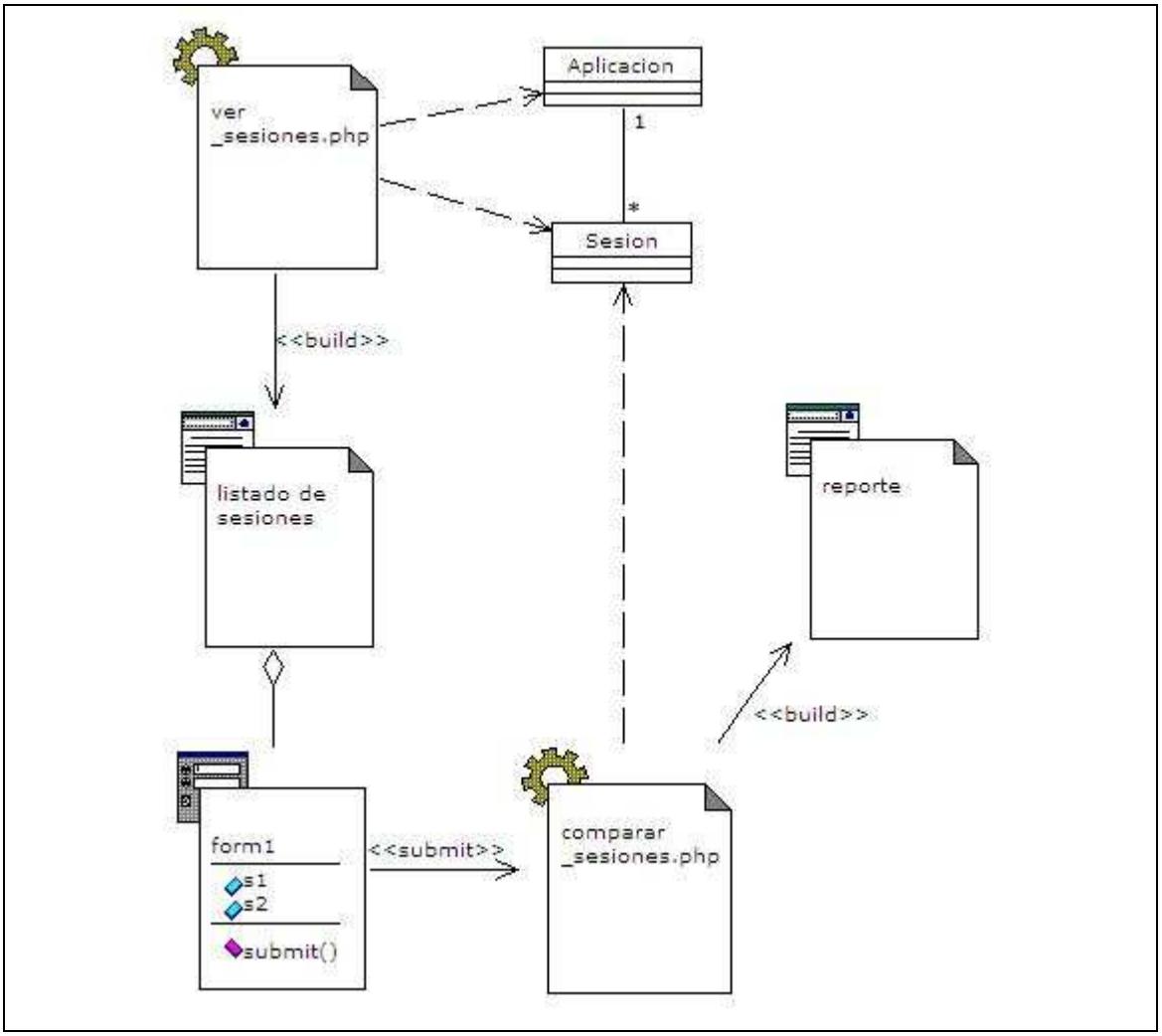

Figura #23: Diagrama WAE comparar sesiones

3.3.4 Diagrama de secuencia

Permite generar una perspectiva cronológica de las interacciones que se presentan en el sistema entre los objetos presentes en un escenario y el conjunto de mensajes intercambiados entre ellos para llevar a cabo la funcionalidad descrita por el escenario. La Figura #24 presenta el diagrama de secuencia del proceso de registro de eventos por parte de la herramienta propuesta.

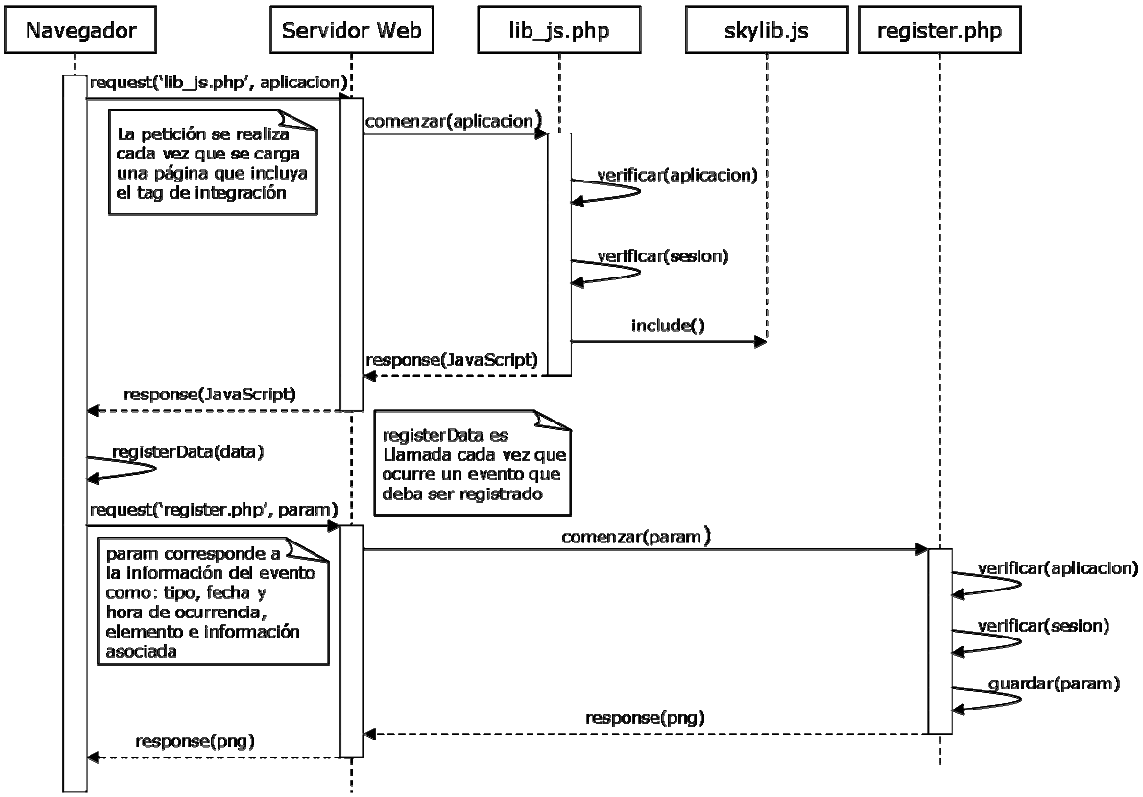

Figura #24: Diagrama de secuencia para el registro de eventos

3.4 Fase IV: Implementación de la solución

La implementación de la herramienta a desarrollar está basada en la técnica de page tagging, para la captura y registro de eventos, usando como tecnología de desarrollo Web PHP, JavaScript y AJAX; y como manejador de base de datos MySQL 5.0.

3.4.1 Justificación del uso de Page Tagging

Los tags se pueden integrar fácil y rápidamente, insertando unas pocas líneas de código en las páginas que se vayan a monitorear.

La integración es independiente de la plataforma de las aplicaciones: no importa si las aplicaciones se encuentran en plataformas distintas e incluso no es necesario conocer la plataforma, debido a que la integración es del lado del cliente.

El page tagging ha sido un estándar al integrar aplicaciones como servidores de publicidad (ad servers) y sistemas de monitoreo de tráfico.

Generalmente, consiste en recopilar información por medio de código del lado del cliente, JavaScript por ejemplo, y se envía al servidor con una petición de una imagen. La petición de una imagen fluye fácilmente por el navegador y no afecta el tiempo de carga de la página.

Es una técnica independiente del navegador del usuario, a diferencia de otras técnicas como los Add-ons y plugins.

También resulta un mecanismo apropiado para capturar las acciones de los usuarios, debido a que se pueden registrar eventos que no se encuentran relacionados con peticiones directas al servidor Web: interacciones con Flash, llenado parcial y borrado de formularios, eventos del lado del cliente como click del ratón, presión y combinación de teclas, entre otros.

3.4.2 Justificación del uso de PHP

PHP es un lenguaje de creación de scripts que corre en una gran cantidad de plataformas y servidores utilizando el mismo código fuente. Como en todos los sistemas se utiliza el mismo código base, los scripts pueden ser ejecutados de manera independiente al sistema operativo.

Además utiliza el paradigma orientado a objetos, permitiendo que los programas y módulos sean más fáciles de escribir, mantener y reutilizar.

PHP es software libre, por consecuencia, no es necesario el pago de licencias para su utilización, adicionalmente existe una amplia documentación en línea de este lenguaje.

3.4.3 Justificación del uso de JavaScript

JavaScript permite crear un mecanismo de integración entre el navegador del cliente de una aplicación Web y la herramienta para realizar pruebas, a través de la inclusión de código (page tags) dentro de las páginas de la aplicación que será sometida a prueba.

JavaScript ofrece un mecanismo de integración que no afecta el desempeño de los servidores donde la aplicación Web sometida a monitoreo, ya que la herramienta puede estar ubicada en otros servidores.

JavaScript se encuentra disponible en la mayoría de los navegadores actuales y ofrece mecanismos para capturar y generar eventos, con su información asociada, a través de las interacciones de los usuarios con las aplicaciones Web y fácil acceso al Document Object Model (DOM).

3.4.4 Justificación del uso de MySQL 5.0

MySQL, en la actualidad, es un sistema manejador de bases de datos muy popular, con documentación extensa y de fácil integración con php y otros lenguajes. También es compatible con múltiples plataformas y sistemas.

MySQL es escalable, y posee múltiples tipos de datos disponibles además de permitir el uso de funciones y sentencias complejas incluyendo el uso de procedimientos almacenados (Store Procedures) para delegar funciones sobre el sistema manejador de base de datos, disminuyendo la cantidad de actividad que deben manejar los scripts del lado del servidor Web.

MySQL es software libre, por consecuencia, no es necesario el pago de licencias para su utilización.

3.4.5 Tag JavaScript: mecanismo de integración con aplicaciones Web

La técnica de "page tagging" fue utilizada como mecanismo de integración entre las aplicaciones Web y la herramienta de pruebas automáticas.

El tag utilizado puede ser visto en la Figura #25:

```
<script type = "text/javascript"
        src = "http://<dominio_herramienta>/skylib/tag/lib_js.php?id=<id_aplicación>">
</script>
```

Figura #25: Tag de integración con aplicaciones Web

Donde <dominio_herramienta> es el dominio donde se encuentre alojada la herramienta de pruebas (no el dominio de las aplicaciones a ser probadas) y <id_aplicación> es el identificador único generado por la herramienta que corresponde a cada aplicación Web registrada en ella.

El script realiza una serie de funciones descritas a continuación:

- Descarga un archivo con código JavaScript (lib_js.php) que contiene todas las funciones encargadas de monitorear la ocurrencia de eventos, peticiones AJAX y el resto de la información que registra la herramienta. En este punto se valida la aplicación Web con la herramienta tomando en cuenta el identificador (<id_aplicación>) y la dirección ip desde donde se realiza la petición del script. Luego se crea la sesión de grabación correspondiente. En caso de que se indique automatización, este script genera el tag que descarga el código correspondiente a ser ejecutado en la simulación.

- Una vez descargado el código, este contiene un evento asociado a la carga de la página que cuando ocurre, se registran las funciones monitoras de eventos con los elementos correspondientes dentro de la página (Formularios, imágenes, etc.) y que también define el comportamiento ante peticiones AJAX.

- Este script contiene una función registerData() que es llamada cada vez que ocurra y deba ser registrado un evento dentro de la página. Dicha función forma un QueryString que será anexado a una petición que se realizará a la herramienta a través de la solicitud de una imagen. Este QueryString contiene la información asociada a un evento, que será procesada por la herramienta, con la finalidad de registrar este evento.

- Finalmente, si la automatización fue indicada, se incluye y ejecuta el código correspondiente a la sesión que desea ser automatizada.

La petición, o contenido de la función registerData(), que permite enviar la información asociada al evento puede ser observada en la Figura #26:

```
function registerData(evt, url, id, info)
{
	if(_GRABAR) // Se graba si y solo si esta la opcion activa
	{
		param = "evento=" + evt + "&fecha=" + gDate() + "&url_asoc=" + escape(url) +
			"&elemento=" + id + "&info=" + info + "&nocache=" + Math.random();
		imgRequest = new Image(1,1);
		imgRequest.src = "http://" + _HOST + "/skylib/tag/register.php?" + param;
		// Si es un submit damos cierto tiempo para que se complete el request
		if(evt == 2) sleep(500);
	}
}
```

Figura #26: Función registerData para el registro de eventos

Es importante destacar que tanto el código que genera el Tag como la imagen utilizada para realizar la petición de registro pueden ser capturados por el caché del navegador. Esta situación debe ser solucionada, en el caso particular de la imagen, ya que podría evitar que sea llevado a cabo el registro de eventos al no realizar la petición a la herramienta sino tomar la imagen del caché. La solución implementada consiste en agregar un parámetro adicional en la url de la petición de la imagen que sea un número aleatorio para que la petición sea diferente cada vez y evitar el posible caché.

3.4.6 Script de generación de código JavaScript: lib_js.php

El tag de integración incluido en las páginas de las aplicaciones Web ejecuta este script el cual se encarga de generar e incluir código JavaScript para realizar la grabación y automatización de eventos. El contenido del script es mostrado a continuación en la Figura #27.

```
<?php

include_once("../class/aplicacion.php");
session_start(); // iniciamos la sesion
$valido = true;// Indica si se genera o no el código de la el código JavaScript
if(!isset($_SESSION["aplicacion"]))// Si no se encuentra validada la aplicación
{
```

```
        // Si el id de la aplicación es pasada por parametro se puede proseguir
        if(isset($_GET["id"]))
        {
                if(get_magic_quotes_gpc()) $_GET["id"] = stripslashes($_GET["id"]);
                // Creamos un nuevo objeto aplicación
                $app = new aplicacion($_GET["id"], "", "");
                // Verificamos que la aplicación exista en la BD
                // y la IP de la petición sea una autorizada por dicha aplicación
                if($app -> existe($_GET["id"]) && $app -> existe_ip($_SERVER["REMOTE_ADDR"]))
                {
                        $_SESSION["aplicacion"] = $_GET["id"];// Validamos la aplicación
                }
                else
                {
                        $valido = false;// Error: no se genera código
                }
        }
}
// Indicamos que el tipo de contenido a devolver es código JavaScript
header("content-type: text/javascript");

if($valido)// Si la aplicación fue validada por la herramienta
{
//Si no existen en sesión los parámetros de comportamiento de la herramienta se inicializan
if(!isset($_SESSION["grabar"])) $_SESSION["grabar"] = 0;
if(!isset($_SESSION["auto"])) $_SESSION ["auto"] = 0;
if(!isset($_SESSION["id_auto"])) $_SESSION ["id_auto"] = 0;
if(!isset($_SESSION["tiempo"])) $_SESSION ["tiempo"] = 0;
?>
// Parámetros a ser tomados en cuenta en el comportamiento de la herramienta
// Si se graban o no los eventos
var _GRABAR =  <?=$_SESSION ["grabar"]?>;
// Si se va a realizar una simulación automática
var _AUTO =  <?=$_SESSION ["auto"]?>;
// El id de la sesion en caso de automatizar
var _IDAUTO = <?=$_SESSION ["id_auto"]?>;
// El tiempo de espera entre las acciones de simulación
var _TIEMPO =  <?=$_SESSION ["tiempo"]?>;
// El dominio donde se encuentra alojada la herramienta
var _HOST = "<?=$_SERVER['HTTP_HOST']?>";
```

```
<?php
        // Se incluye el código JavaScript con las funciones para la grabación y ejecución automática
        include_once("skylib.js");
}
?>
```

Figura #27: Script para la generación del código del tag de integración

3.4.7 Script de registro de información: register.php

Es el script encargado de manejar las peticiones de registro de eventos realizadas por las aplicaciones y retornar una imagen. La Figura #28 muestra el código del script.

```
<?php
include("../class/evento.php");
session_start();
// Sólo se registra en caso de que la aplicación se encuentre validada y la sesión para grabar exista
if(isset($_SESSION["aplicacion"]) && isset($_SESSION["sesion"]))
{
        // Para que la implementacion sea independiente del get_magic_quotes_gpc en caso de que exista lo revertimos
        if(get_magic_quotes_gpc())
        {
                $_GET["evento"] = stripslashes($_GET["evento"]);
                $_GET["fecha"] = stripslashes($_GET["fecha"]);
                $_GET["url_asoc"] = stripslashes($_GET["url_asoc"]);
                $_GET["elemento"] = stripslashes($_GET["elemento"]);
                $_GET["info"] = stripslashes($_GET["info"]);
        }
        // Creamos un objeto evento temporal para almacenar los datos enviados
        $evt = new evento("", $_GET["evento"], 0, $_GET["fecha"], $_SERVER["HTTP_REFERER"], $_GET["url_asoc"], $_GET["elemento"], $_GET["info"]);
        // Guardamos el evento en la bd tomando en cuenta la sesión a la cual pertenece
        $evt -> guardar($_SESSION["sesion"]);
}
// Se indica que el tipo de contenido a generar es una imagen
header("Content-type: image/png");
// Se crea y genera la imagen
$im = @imagecreatetruecolor(1, 1) or die("No se puede crear la imagen");
```

```
imagepng($im);
?>
```

Figura #28: Script para el registro de eventos

3.4.8 Mecanismo de activación de grabación / automatización

Es importante destacar que las funciones del tag incluido en las aplicaciones Web no se deben encontrar disponibles para todos los usuarios que hagan uso de ellas, sino sólo por aquellos destinados a realizar las pruebas. Entonces un mecanismo de activación debe ser provisto para indicarle esta situación.

El mecanismo desarrollado consiste en agregar parámetros adicionales a la url de la aplicación Web al momento de realizar la prueba con el siguiente formato:

skyrec=<on/off>&skyauto=<on/off>:<id_sesion>:<t_auto>

Donde skyrec, con valores "on" u "off", indica el hecho de registrar o no los eventos ocurridos ya sean generados por el usuario o automáticos, skyauto, con valores "on" u "off", indica si se va a realizar la simulación automática de alguna sesión, <id_sesion> indica el identificador provisto por la herramienta correspondiente a la sesión que se desea automatizar y <t_auto> indica el tiempo de espera en ms. entre cada acción a automatizar. Por ejemplo, si por medio del navegador se indica:

http://www.ejemplo.com/index.html?skyrec=on&skyauto=on:15:200

Esto significa que estamos monitoreando la aplicación www.ejemplo.com (Asumiendo que esta aplicación ya tiene incluido el tag de integración en la página actual index.html), sí se desea realizar la grabación de eventos, sí se desea automatizar y se va a reproducir la sesión con identificador 15 y el tiempo de espera entre cada acción será de 200 ms.

Estos parámetros son capturados por una función perteneciente al código del Tag generado y de acuerdo con sus valores se actualizan los parámetros de

grabación / automatización que utiliza la herramienta. El código de dicha función es presentado en la Figura #29.

```
// Activa / desactiva la grabación / simulación
function setParam()
{
	// Se obtienen los parametros de la url
	rec = getParamUrl("skyrec");
	auto = getParamUrl("skyauto");
	param = "";
	if(rec != "")
	{
		// Se arma el querystring
		param += "rec=" + rec;
		// Se actualizan los parámetros globales a nivel de JavaScript
		_GRABAR = (rec=="on")?1:0;
	}
	if(auto != "") // Si se especifica el tiempo de grabación
	{
		auto = auto.split(":");
		sim = auto[0];
		sesion_id = auto[1];
		time = auto[2]
		// Se actualizan los parámetros globales a nivel de JavaScript
		_AUTO = (sim=="on")?1:0;
		_IDAUTO = parseInt(sesion_id);
		_TIEMPO = parseInt(time);
		// Se arma el querystring
		if(param != "") param += "&";
		param += "auto=" + sim + "&id=" + sesion_id + "&time=" + time;
	}
	// Si existe alguno de los 2 parámetros se hace la petición al servidor indicando actualizar
	if(param != "")
	{
		param += "&nocache=" + Math.random();
		imgRequest = new Image(1,1);
		imgRequest.src = "http://" + _HOST + "/skylib/tag/params.php?" + param;
	}
}
```

Figura #29: Función setParam para el ajuste de grabación / simulación

3.4.9 Script de actualización de parámetros: params.php

Este script actualiza los parámetros de comportamiento referentes a grabación y simulación de la herramienta que se encuentran alojados en sesión, de acuerdo con el mecanismo indicado anteriormente. La Figura #30 muestra el código del script params.php

```
<?php
include("../class/sesion.php");
session_start();
if(isset($_SESSION["aplicacion"]))// Si la aplicación se encuentra validada
{
	if(isset($_GET["rec"]))// Si existe el parámetro de grabación
	{
		switch($_GET["rec"])
		{
			case "on": $_SESSION["grabar"] = 1; break;
			case "off": $_SESSION["grabar"] = 0; break;
		}
		// Si se activa la opcion de grabar, se crea una sesión nueva en caso que no exista
		if($_SESSION["grabar"] == 1 && !isset($_SESSION["sesion"]))
		{
			$s = new sesion("","");
			$_SESSION["sesion"] = $s -> nueva($_SESSION["aplicacion"]);
		}
		// Si la grabar esta desactivado, se destruye la sesión actual en caso que exista
		if($_SESSION["grabar"] == 0 && isset($_SESSION["sesion"]))
		{
			unset($_SESSION["sesion"]);
		}
	}
	if(isset($_GET["auto"]))// Si existe el parámetro de simulación
	{
		switch($_GET["auto"])
		{
			case "on": $_SESSION["auto"] = 1; break;
			case "off":
				$_SESSION["auto"] = 0;
				// Eliminamos las variables relacionadas con la automatización
				if(isset($_SESSION["auto_s"])) unset($_SESSION["auto_s"]);
```

```
                                if(isset($_SESSION["auto_i"])) unset($_SESSION["auto_i"]);
                        break;
                }
                $_SESSION["id_auto"] = $_GET["id"];
                $_SESSION["tiempo"] = $_GET["time"];
        }
}
?>
```

Figura #30: Script para el ajuste de grabación / simulación

3.4.10 Funciones relevantes para el manejo de eventos

Dentro del código JavaScript generado existen numerosas funciones relacionadas con la detección, registro y automatización de eventos. A continuación se exponen algunas de las más importantes.

La detección de eventos es realizada mediante observadores (listeners) de eventos asociados a los elementos del HTML de las páginas, los cuales tienen una diferente implementación dependiendo del navegador. La función, que asocia un observador a un elemento, es presentada a continuación en la Figura #31.

```
function addEvent(elm, evType, func, useCapture)
{
        if(elm.addEventListener)
        {
                elm.addEventListener(evType, func, useCapture); return true;
        }
        else if(elm.attachEvent)
        {
                var r = elm.attachEvent('on' + evType, func); return r;
        }
        else
        {
                elm['on' + evType] = func;
        }
}
```

Figura #31: Función addEvent para el monitoreo de eventos

De manera particular, addEvent es utilizada para asociar una función de inicialización ante la ocurrencia del evento de carga de la página. Dicha función de inicialización añade observadores a los elementos pertinentes, ejecuta funciones para comprobar imágenes y enlaces de la página, define las funciones para el monitoreo de las peticiones AJAX e inicializa la automatización. La Figura #32, presentada en breve, contiene el código de la función.

```
function initialize()
{
	// Se registra la carga de la pagina actual
	registerData(12, "", "", "");
	addEvent(document, 'click', registerClick, false);
	// El evento submit y reset aplicado al document no funciona en IE, debe ser asociado
formulario por formulario
	if(window.event)// Explorer
	{
		formu = document.forms;
		for(i = 0; i < formu.length; i++)
		{
			addEvent(formu[i], 'submit', registerSubmit, false);
			addEvent(formu[i], 'reset', registerReset, false);
		}
	}
	else
	{
		addEvent(document, 'submit', registerSubmit, false);
		addEvent(document, 'reset', registerReset, false);
	}
	// Se asocia el evento 'keyup' para detectar los eventos de teclado
	addEvent(document, 'keyup', registerKey, false);
	// Se verifica la carga de todas las imagenes del documento
	checkImages();
	// Se verifican los enlaces de todo el documento
	checkLinks();
	// Si existe el objeto XMLHttprequest con funciones de monitoreo
	if(typeof(XMLHttpRequest.onopen) != 'undefined')
	{
		// Definición de la función de monitoreo de la llamada a open() del objeto
		XMLHttpRequest.onopen = function(sMethod, sUrl, bAsync)
		{
```

```
                registerData(8, sUrl, "", sMethod + " Asinc=" + bAsync);
        }
        // Definición de la función de monitoreo de la llamada a send() del objeto
        XMLHttpRequest.onsend = function(vData)
        {
                registerData(9, "", "", vData);
        }
        // Definición de la función de monitoreo del cambio de estado de la llamada
        XMLHttpRequest.onreadystatechange = function()
        {
                if(this.readyState == 4)
                        registerData(10, "", "", this.status +" "+ this.statusText);
        }
        // Definición de la función de monitoreo de la llamada a abort() del objeto
        XMLHttpRequest.onabort = function()
        {
                registerData(11, "", "", "");
        }
    }
    if(_AUTO) // Si la automatización se encuentra activa
    {
        startSimulation(); // Se comienza la simulación de acciones
    }
}
```

Figura #32: Función initialize para activar el código generado

Para automatizar un evento, fue elaborada una función (ver Figura #33) que recibe una cadena con formato preestablecido que contiene la información del evento y realiza el conjunto de acciones necesarias para simularlo utilizando fireEvent (Internet Explorer) y dispatchEvent (Firefox).

```
function automatize(line)
{
    // Separamos el string q contiene la instruccion a automatizar
    line = line.split(_S);
    // Se realiza una accion dependiendo de los valores que tome el arreglo de acuerdo al formato
establecido
    switch(line[0])
    {
        case "click": // ["click", elemento]
            // Obtenemos el elemento
```

```
                        var fireOnThis = document.getElementById(line[1]);
                        if(document.createEvent)// Firefox
                        {
                                var evObj = document.createEvent('MouseEvents');
                                evObj.initEvent(line[0],true, true);
                                fireOnThis.dispatchEvent(evObj);
                        }
                        else if( document.createEventObject )// Explorer
                        {
                                fireOnThis.fireEvent("on" + line[0]);
                        }
                        // Si es un enlace tradicional ... simulamos la acción por defecto(No se
consigue disparando el eventos)
                        if(fireOnThis.href)
                        {
                                // Si esta definido un target particular como _blank
                                if(fireOnThis.target)        window.open(node.href,node.target);
                                else location.href = fireOnThis.href;// Sino, es un enlace tradicional...
en la misma página
                        }
                break;
                case "submit": // ["submit", "elemento_id", n_campos, "tipo_campo1" // "info 1", ... ,
"tipo_campo n" / "info n" ]
                        // Obtenemos el elemento
                        var fireOnThis = document.getElementById(line[1]);
                        for(i = 0; i < parseInt(line[2]); i++)
                        {
                                //alert(line[i+3]);
                                campo = line[i+3].split(" // ");
                                switch(campo[0])
                                {
                                        case "text":
                                        case "textarea":
                                        case "select-one":
                                        case "password":
                                        case "hidden":
                                                fireOnThis.elements[i].value = campo[1];
                                        break;
                                        case "checkbox":
                                        case "radio":
                                                // Si se encuentran  seleccionadas (checked ==
true)
```

```
                                                if(parseInt(campo[1]))
fireOnThis.elements[i].checked = true;
                                        }
                                }
                                if(document.createEvent)// Firefox
                                {
                                        var evObj = document.createEvent('HTMLEvents');
                                        evObj.initEvent(line[0],true, true);
                                        fireOnThis.dispatchEvent(evObj);
                                }
                                else if( document.createEventObject )// Explorer
                                {
                                        fireOnThis.fireEvent("on" + line[0]);
                                        fireOnThis.submit();
                                }
                        break;
                        case "reset":// ["reset", elemento]
                                var fireOnThis = document.getElementById(line[1]);
                                if(document.createEvent)// Firefox
                                {
                                        var evObj = document.createEvent('HTMLEvents');
                                        evObj.initEvent(line[0],true, true);
                                        fireOnThis.dispatchEvent(evObj);
                                }
                                else if( document.createEventObject )// Explorer
                                {
                                        fireOnThis.fireEvent("on" + line[0]);
                                }
                        break;
                        case "keydown":// ["keydown", keycode, ctrl, alt, shift]
                        case "keyup":// ["keyup", keycode, ctrl, alt, shift]
                                var fireOnThis = document;// Seria recomendable determinar el elemento ...
                                if(document.createEvent)
                                {
                                        var evObj = document.createEvent('KeyboardEvent');
                                        evObj.initKeyEvent(line[0], true, true, null, parseInt(line[2]),
parseInt(line[3]), parseInt(line[4]), false, parseInt(line[1]), 0);
                                        document.body.dispatchEvent(evObj);
                                }
                                else if( document.createEventObject )
                                {
                                        var evObj = document.createEventObject();
```

```
                                    evObj.keyCode = parseInt(line[1]);
                                    evObj.ctrlKey = parseInt(line[2]);
                                    evObj.altKey = parseInt(line[3]);
                                    evObj.shiftKey = parseInt(line[4]);
                                    document.body.fireEvent("on" + line[0],evObj);
                                }
                        break;
                }
}
```

Figura #33: Función automatize para la simulación de un evento

Para finalizar este punto, es importante destacar las instrucciones que se encuentran al final del código generado, entre las cuales se incluye el código que contiene la simulación. La Figura #34 presenta estas instrucciones.

```
// Se determinan los parametros skyrec y skyauto y se toman las acciones correspondientes en caso de
que existan
setParam();
// Si se va a automatizar, se incluye el script que contiene el código de la automatización
if(_AUTO)
{
        document.write('<scr' + 'ipt type="text/javascript" src="http://'+ _HOST
+'/skylib/tag/auto.php?s=' + _IDAUTO + '&t=' + _TIEMPO + '&nocache=' + Math.random() + '"></scr'
+ 'ipt>');
}
window.onerror = registerError; // Para detectar y registrar errores a nivel de script
addEvent(window, 'load', initialize, false);// Añadir la función initialize al evento de carga de la página
```

Figura #34: Código relevante para el comportamiento de la herramienta

3.4.11 Script para generar el código de simulación: auto.php

Este script (ver Figura #35) recibe como parámetros la sesión a reproducir y el tiempo de espera entre cada acción que forme parte de dicha simulación para generar el código JavaScript con la función startSimulation que contiene una llamada a la función automatize, expuesta anteriormente, por cada instruccion que forman parte de la sesión a ser simulada.

```
<?php
include("../class/sesion.php");
session_start();
// Indicamos que el contenido generado será código JavaScript
header("content-type: text/javascript");
if(isset($_GET["s"]) && isset($_GET["t"]))
{
	// Si la sesión a automatizar no existe o se desea cargar una con otro id ...
	if(!isset($_SESSION["auto_s"]) || $_SESSION["auto_s"] -> get_id() != $_GET["s"])
	{
		$_SESSION["auto_s"] = new sesion("","");
		// Se intentan cargar los datos de la sesión que corresponde
		if(!$_SESSION["auto_s"] -> cargar($_GET["s"]))
		{
			// Si no exista la sesion u ocurrió algun error la eliminamos
			unset($_SESSION["auto_s"]);
			// Eliminamos el indice que recorre la sesion también en caso de que exista
			if(isset($_SESSION["auto_i"])) unset($_SESSION["auto_i"]);
		}
		else
		{
			// Como es una nueva sesion se empieza a recorrer desde el primer evento
			$_SESSION["auto_i"] = 0;
		}
	}
	if(isset($_SESSION["auto_s"]))// Si ya existe (se cargo de manera exitosa)
	{
		// Separamos la url de la peticion de los parametros
		$url = explode("?", $_SERVER['HTTP_REFERER']);
		$urlRequest = $url[0];
		echo "function startSimulation(){";// Comienzo código JavaScript
		// Se recorren los eventos de la sesión que corresponden a la página actual
		for(; $_SESSION["auto_i"] < $_SESSION["auto_s"] -> n_eventos();
$_SESSION["auto_i"]++)
		{
			$evt = $_SESSION["auto_s"] -> get_evento($_SESSION["auto_i"]);
			$param = "";
			// Separamos la url del evento actual de los parametros
			$url = explode("?",$evt -> get_url());
			$urlEvt = $url[0];
			// Si la url del evento actual es diferente a la url de la petición
			if($urlRequest != $urlEvt)
```

```
                        {
                                break;// no se generan mas eventos para esa página
                        }
                        // Tomamos en cuenta solo los eventos q son automatizables
                        if($evt -> get_auto())
                        {
                                // Armamos un string diferente de acuerdo al nombre de cada evento
                                switch($evt -> get_nombre())
                                {
                                        case "click":
                                                $param = "click _// ".$evt -> get_elemento();
                                                break;
                                        case "submit":
                                                // Separamos el string que contiene los campos del
formulario
                                                $campos = explode(" _// \n", $evt ->
get_informacion());
                                                $tam = count($campos)-1;
                                                $param = "submit _// ".$evt -> get_elemento()."
_// ".$tam;
                                                for($j = 0; $j < $tam; $j++)
                                                {
                                                        // Separamos los elementos que forman el
campo actual y tomamos solo los necesarios para la automatización
                                                        $campo_actual = explode(" _// ",
$campos[$j]);
                                                        $param .= " _// ".$campo_actual[0]." //
".$campo_actual[1];
                                                }
                                                break;
                                        case "reset":
                                                $param = "reset _// ".$evt -> get_elemento();
                                                break;
                                        case "keyup":
                                                $param = "keyup _// ".$evt -> get_informacion();
                                                break;
                                }
                                // Acomodar el salto de linea de las textarea
                                $param = str_replace(array("\r\n", "\n"), "\\n", $param);
                                //Código para automatizar el evento actual
                                echo "automatize('".$param."');";
                                echo "sleep(".$_GET["t"].");";
```

```
                        }
                }
                echo "}"; //Fin startSimulacion
        }
        else
        {
                // En caso de que la sesión que desea ser simulada no exista en la bd
                echo("function startSimulation(){alert('Error en la sesión / sesion no existente');}");
        }
}
?>
```

Figura #35: Script para generar el código de simulación de una sesión

3.5 Fase V: Pruebas

En esta fase la herramienta o aplicación desarrollada es sometida a una serie de pruebas, con la finalidad comprobar su correcto funcionamiento, detectando posibles comportamientos inesperados o errores que puedan presentarse. A continuación se presentan las pruebas realizadas.

3.5.1 Prueba de autenticación de usuarios en la herramienta

El acceso a la sección administrativa de la herramienta de monitoreo de eventos desarrollada debe ser protegido de usuarios indeseados, por lo que se implementó un sistema de autenticación para accesar a dicha sección.

Al momento de realizar la prueba la aplicación contaba sólo con un usuario "admin" autorizado con contraseña "123". La información del usuario almacenada nivel de base de datos puede verse en la Figura #36. Importante destacar que la herramienta almacena y trabaja con las contraseñas encriptadas a través sha1 (el store procedure existente insertar_usuario realiza esta encriptación al momento de crear un nuevo usuario).

Figura #36: Datos de usuario autorizado para acceso a sección administrativa

Se intenta acceder a la aplicación con datos de usuario inválidos o inexistentes para comprobar la respuesta del a aplicación (ver Figura #37).

Figura #37: Formulario de ingreso a la herramienta: datos inválidos

La respuesta obtenida puede observarse en la Figura #38 la cual consiste en mostrar el formulario de ingreso nuevamente pero con un mensaje informativo indicando que el intentó de autenticación previa ha fallado por algún motivo (En este caso datos erróneos).

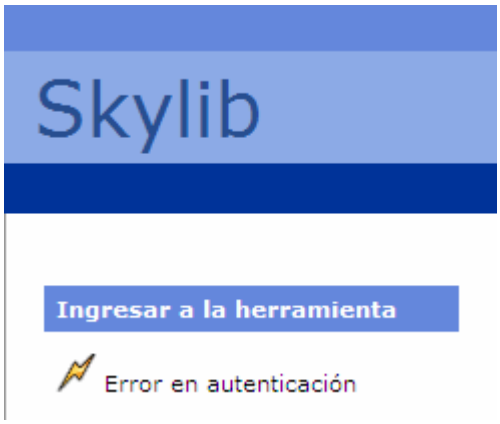

Figura #38: Respuesta de autenticación ante datos inválidos

3.5.2 Prueba de administración de aplicaciones

Una vez el que un usuario se encuentre autenticado en la herramienta, de manera satisfactoria, tiene la posibilidad de crear nuevas aplicaciones y asociar direcciones IP a las aplicaciones creadas con el fin de autorizar el registro de eventos desde esas direcciones. La Figura #39 muestra la interfaz para la creación de nuevas aplicaciones.

Figura #39: Interfaz de creación de aplicaciones

Se prueba enviando el formulario de creación de nueva aplicación con campos incompletos y se genera un mensaje de alerta (ver Figura #40) indicando que todos los campos deben ser rellenados. Lo cual es correcto.

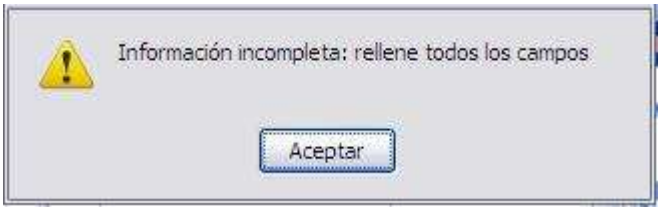

Figura #40: Mensaje de alerta información incompleta: crear aplicación

Si se llenan los datos completos del formulario y se envía, se muestra la interfaz de aplicaciones con la nueva aplicación creada junto con un mensaje indicando el éxito de la creación. La Figura #41 muestra dicha interfaz para la creación de una aplicación llamada "ejemplo" cuyo dominio es "localhost".

Figura #41: Interfaz ver aplicaciones: aplicación creada

Si se ejecuta el enlace "eliminar" asociado a una aplicación creada se solicita la confirmación de la eliminación, mediante confirm() de JavaScript y en caso afirmativo la aplicación es eliminada y se muestra la interfaz con el mensaje

correspondiente y los cambios realizados. En la Figura #42 se muestra la interfaz mencionada luego de eliminar la aplicación creada anteriormente.

Figura #42: Interfaz ver aplicaciones: aplicación eliminada

3.5.3 Prueba de administración de direcciones IP

Recordando que la herramienta de monitoreo autoriza a las aplicaciones Web a realizar peticiones de registro de eventos solamente desde direcciones IP autorizadas, se debe comprobar el correcto funcionamiento mecanismo que permite autorizar y desautorizar (que se traduce en crear y eliminar) direcciones IP para las aplicaciones registradas. La interfaz que permite visualizar las direcciones IP asociadas a una aplicación Web en particular y autorizar nuevas direcciones se puede ver en la Figura #43, en este caso para la aplicación con dominio "localhost".

Figura #43: Interfaz de direcciones IP. Aplicación localhost

Si el campo de la nueva dirección IP se encuentra vacío y se pretender crear una nueva IP, se muestra un mensaje de alerta a nivel de JavaScript (ver Figura #44) indicando que el campo debe ser rellenado para poder procesar la petición de creación. El despliegue y funcionamiento de este mensaje de alerta es correcto.

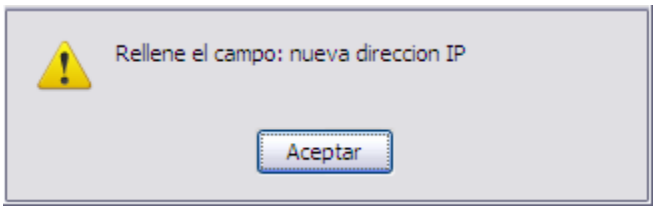

Figura #44: Mensaje de alerta al crear nueva IP: campo vacío

Si el campo no se encuentra vacío pero el texto en él posee un formato no acorde con una dirección IP válida (que contenga letras, espacios u otros caracteres especiales inválidos) la creación no se lleva a cabo y se muestra el mensaje correspondiente de error de formato de la dirección. La Figura #45 muestra este mensaje.

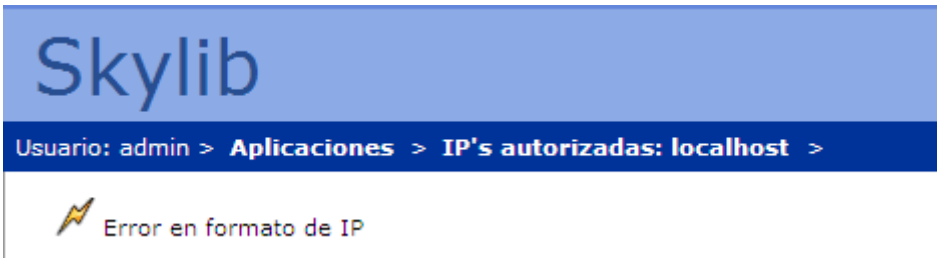

Figura #45: Error en formato de dirección IP

Si se desea crear una nueva dirección IP, el formato es el correcto, y se presiona el botón "Crear". La nueva dirección es creada y asociada a la aplicación y se muestra el mensaje informativo correspondiente. La Figura #46 muestra la interfaz luego de haber creado y asociado la IP "127.0.0.1" a la aplicación con dominio "lolcahost".

Figura #46: Dirección IP creada de manera exitosa

Si se ejecuta el enlace "eliminar", asociado a una dirección IP creada, dicha dirección debe eliminarse y dejar de estar autorizada por la aplicación a la que

pertenecía. La Figura #47 muestra el resultado de eliminar una dirección IP creada con anterioridad.

Figura #47: Dirección IP eliminada

3.5.4 Prueba de autenticación de aplicaciones Web en la herramienta

Una aplicación Web, para poder monitorear los eventos de sus usuarios, debe estar registrada previamente en la herramienta y la solicitud de registro debe provenir de direcciones IP autorizadas por esa aplicación. El código del tag debe ser generado o no dependiendo de estos dos factores. Para someter a prueba la generación del JavaScript, en primer lugar se añade código adicional que se genera en caso de que la aplicación no se encuentre registrada o la IP no sea autorizada. La Figura #48 muestra el código adicional añadido al script lib_js.php.

```
echo "// Este comentario es visible cuando la aplicación no existe\n";
echo "// o la petición proviene de una IP no autorizada";
```

Figura #48: Código adicional añadido a lib_js.php para pruebas.

Luego creamos una aplicación de ejemplo sin ninguna IP autorizada a través de la sección administrativa de la herramienta. En la Figura #49 se puede ver la aplicación creada cuyo ID es 3, por lo que el Tag que debe ser colocado en las páginas para incluir el script de monitoreo y simulación de eventos tendría el siguiente formato: http://localhost/skylib/tag/lib_js.php?id=3 (durante las pruebas la herramienta se encuentra alojada localmente).

Figura #49: Aplicación de ejemplo creada para pruebas

Realizar invocaciones directas a lib_js.php desde navegador equivale a una petición de una aplicación que incluya el Tag en sus páginas. Por lo que se van a realizar dos peticiones diferentes, una de ellas con un id de una aplicación no registrada en la herramienta y otra con el id de la aplicación registrada sin direcciones IP autorizadas. En ambos casos el código del script no debe ser generado ya que no cumple con las condiciones explicadas anteriormente y en su lugar se debe mostrar el código adicional añadido a lib-js.php. La Figura #50 muestra el resultado obtenido de ambas peticiones que resulta el resultado esperado.

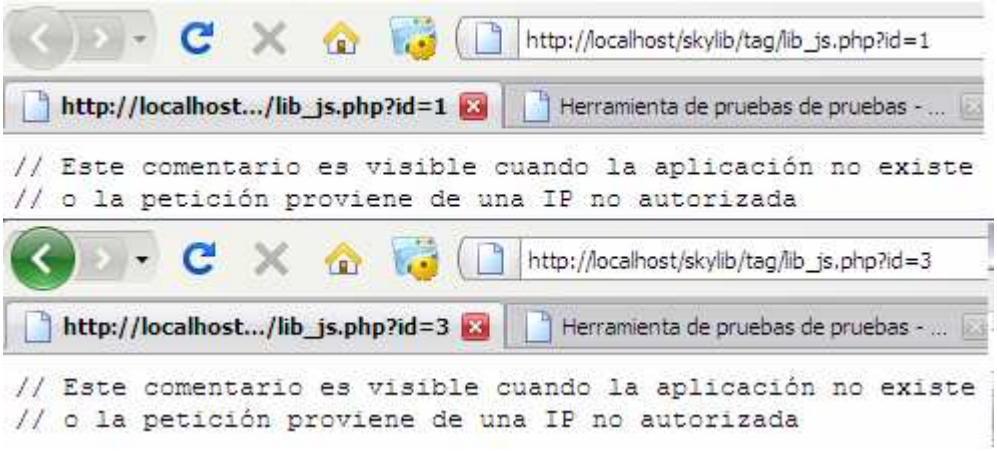

Figura #50: Figura #53: código no generado

Como siguiente paso, se autoriza la dirección IP, desde donde hicimos las peticiones anteriores, para la aplicación de ejemplo (ver Figura #51) y se realiza nuevamente la petición con el ID de la aplicación existente, y ahora autorizada, cuyo resultado puede verse en la Figura #52. El código es generado ya que las condiciones establecidas se cumplen, el resultado es correcto.

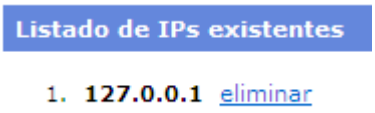

Figura #51: IP autorizada para propósito de pruebas

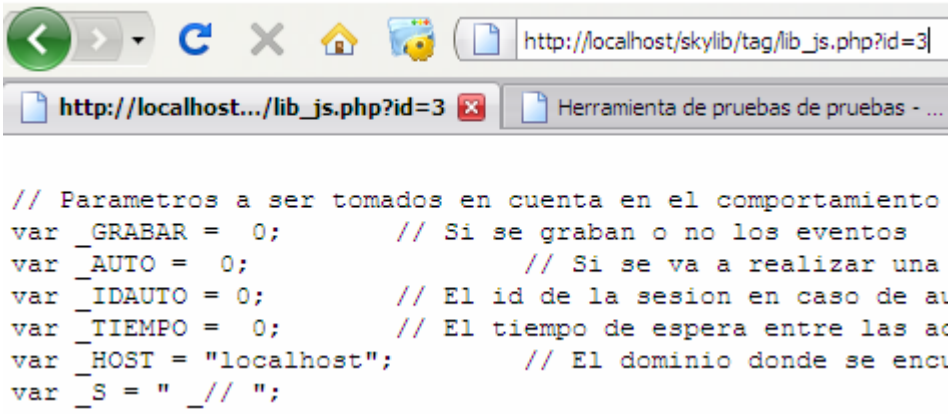

Figura #52: Resultado de petición: código generado

3.5.5 Prueba de grabación

El objetivo de esta prueba consiste en verificar el registro correcto de los eventos que fueron definidos como de interés para ser monitoreados por la herramienta.

Para realizar la prueba de grabación se construyeron algunas páginas sencillas con la siguiente descripción:

- **enlaces.html:** contiene una serie de enlaces cuyo resultado se carga en un iframe dentro de la página: un enlace externo (http://www.google.com), un enlace interno a otra página que si existe (ok.html) y un enlace interno a una

página inexistente (no_existe.html). Contiene enlaces para dirigirse a formulario.html y ajax.html.

- **formulario.html:** contiene tres imágenes (dos existen y la otra no) y un formulario, con varios campos de diferentes tipos, cuyo action va dirigido a un script PHP simple que redirecciona a otra página. Contiene enlaces para dirigirse a enlaces.html y ajax.html.

- **ajax.html:** contiene tres botones, uno realiza una petición AJAX por GET, otro por POST y el último realiza una petición por GET a una página inexistente. Contiene enlaces para dirigirse a formulario.html y enlaces.html.

El Tag de integración se incluye en las páginas mencionadas anteriormente tomando en cuenta una aplicación creada y una IP autorizada. La secuencia de acciones para realizar la prueba es la siguiente:

- **enlaces.html:** se comienza la grabación, click en el enlace OK, presionar combinación Shift+O, presionar tecla K, click en enlace hacia formulario el cual redirecciona a formulario.html.

- **formulario.html:** reset del formulario, se llenan los campos y se envía el formulario cuyo action redirecciona a la página ajax.html.

- **ajax.html:** click sobre el botón para realizar petición AJAX POST, click sobre el botón para realizar la petición sin respuesta (no response) y finalmente se detiene la grabación.

El resultado de la grabación realizada por la herramienta, de la secuencia de eventos mencionada anteriormente, puede apreciarse en la Figura #53.

1. 03-11-08 11:33:10 - **pagina cargada**
   url: 'http://localhost/ejemplo/enlaces.html?skyrec=on'
2. 03-11-08 11:33:11 - **enlace**
   Estado: 200 OK - tipo: 'text/css'
   url: 'http://localhost/ejemplo/style.css'
3. 03-11-08 11:33:11 - **enlace**
   Estado: 200 OK - tipo: 'text/javascript'
   url: 'http://localhost/skylib/tag/lib_js.php?id=1'
4. 03-11-08 11:33:11 - **enlace**
   Estado: 302 Found - tipo: ''
   url: 'http://www.google.com/'
5. 03-11-08 11:33:11 - **enlace**
   Estado: 200 OK - tipo: 'text/html'
   url: 'http://localhost/ejemplo/ok.html'
6. 03-11-08 11:33:11 - **enlace**
   Estado: 404 Not Found - tipo: 'text/html; charset=iso-8859-1'
   url: 'http://localhost/ejemplo/no_existe.html'
7. 03-11-08 11:33:11 - **enlace**
   Estado: 200 OK - tipo: 'text/html'
   url: 'http://localhost/ejemplo/formulario.html'
8. 03-11-08 11:33:11 - **enlace**
   Estado: 200 OK - tipo: 'text/html'
   url: 'http://localhost/ejemplo/ajax.html'
9. 03-11-08 11:33:30 - **click**
   ID: 'enlace ok' - Tipo: A
   href: 'http://localhost/ejemplo/ok.html'
10. 03-11-08 11:33:38 - **keyup**
    shift + O
11. 03-11-08 11:33:41 - **keyup**
    K
12. 03-11-08 11:33:52 - **click**
    ID: 'Ir a form' - Tipo: A
    href: 'http://localhost/ejemplo/formulario.html'
13. 03-11-08 11:33:52 - **enlace**
    Estado: 200 OK - tipo: 'text/css'
    url: 'http://localhost/ejemplo/style.css'
14. 03-11-08 11:33:52 - **enlace**
    Estado: 200 OK - tipo: 'text/javascript'
    url: 'http://localhost/skylib/tag/lib_js.php?id=1'
15. 03-11-08 11:33:52 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/enlaces.html'
16. 03-11-08 11:33:52 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/ajax.html'
17. 03-11-08 11:33:52 - **pagina cargada**
    url: 'http://localhost/ejemplo/formulario.html'
18. 03-11-08 11:33:52 - **carga de imagen**
    alt: 'foto1'
    src: 'http://localhost/ejemplo/nutria.jpg'
19. 03-11-08 11:33:52 - **error en imagen**
    alt: 'foto2'
    src: 'http://localhost/ejemplo/noexiste.jpg'
20. 03-11-08 11:33:52 - **carga de imagen**
    alt: 'foto3'
    src: 'http://localhost/ejemplo/claymore.jpg'
21. 03-11-08 11:33:58 - **reset**
    ID: 'form1'
    action: 'http://localhost/ejemplo/procesa_form.php'
22. 03-11-08 11:34:17 - **submit**
    ID: 'form1'
    action: 'http://localhost/ejemplo/procesa_form.php'
23. 03-11-08 11:34:18 - **pagina cargada**
    url: 'http://localhost/ejemplo/ajax.html'
24. 03-11-08 11:34:18 - **enlace**
    Estado: 200 OK - tipo: 'text/css'
    url: 'http://localhost/ejemplo/style.css'
25. 03-11-08 11:34:18 - **enlace**
    Estado: 200 OK - tipo: 'text/javascript'
    url: 'http://localhost/skylib/tag/lib_js.php?id=1'
26. 03-11-08 11:34:18 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/enlaces.html'
27. 03-11-08 11:34:18 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/formulario.html'
28. 03-11-08 11:34:33 - **ajax open**
    Información: POST Asinc=true
    url: 'ejemploajax3.html'
29. 03-11-08 11:34:33 - **ajax send**
    parámetros: 't1=Campo1'
30. 03-11-08 11:34:33 - **click**
    ID: 'ajax_post' - Tipo: INPUT
    href: ''
31. 03-11-08 11:34:33 - **ajax readystatechange 4**
    Estado 200 OK
32. 03-11-08 11:34:37 - **ajax open**
    Información: GET Asinc=true
    url: 'no_existe.html'
33. 03-11-08 11:34:37 - **click**
    ID: 'ajax_noresponse' - Tipo: INPUT
    href: ''
34. 03-11-08 11:34:37 - **ajax send**
    parámetros: 'null'
35. 03-11-08 11:34:37 - **ajax readystatechange 4**
    Estado 404 Not Found

Figura #53: Eventos registrados en prueba de grabación

3.5.6 Prueba de simulación

El objetivo de esta prueba consiste en verificar el correcto funcionamiento de la reproducción automatizada de eventos de sesiones grabadas previamente. Para realizar esta prueba serán utilizadas las mismas páginas y será reproducida o simulada la misma secuencia de eventos pertenecientes a la sesión registrada en la prueba de grabación anteriormente descrita. En la Figura #54 se pueden apreciar las sesiones registradas al momento de realizar la simulación, en este caso solamente se encuentra la sesión que corresponde al conjunto de acciones realizadas en la prueba de grabación con ID 1.

Figura #54: Sesiones registradas al momento de la simulación

Para realizar esta prueba se indicará a la herramienta, siguiendo el procedimiento establecido, realizar la grabación de eventos y la reproducción automática de la sesión registrada que corresponde a la prueba de grabación (Cuyo ID, en este caso, es 1). Los parámetros añadidos a la url para activar la grabación y simulación de esta prueba quedan de la siguiente manera: skyrec=on&skyauto=on:1:500, indicando que se activa la grabación y se van a reproducir los eventos de la sesión con identificador 1 y el tiempo de espera entre cada acción es de 500 ms.

Los resultados obtenidos pueden ser apreciados en la Figura #55. Si comparamos la sesión nueva generada a partir de la reproducción de la sesión 1 con la original se puede apreciar que son idénticas en cuanto a los eventos que la conforman con la única diferencia de que la automatizada incluye el script auto.php que, como ya se explicó en algún punto, es el encargado de generar las instrucciones que forman parte de una simulación. Esta comparación lleva a concluir que la reproducción automática de sesiones funciona correctamente para una aplicación sin alteraciones considerables (Cambios de contenido y elementos).

1. 03-11-08 13:11:04 - **pagina cargada**
   url: 'http://localhost/ejemplo/enlaces.html?skyrec=on&skyauto=on:1:500'
2. 03-11-08 13:11:04 - **enlace**
   Estado: 200 OK - tipo: 'text/css'
   url: 'http://localhost/ejemplo/style.css'
3. 03-11-08 13:11:04 - **enlace**
   Estado: 200 OK - tipo: 'text/javascript'
   url: 'http://localhost/skylib/tag/lib_js.php?id=1'
4. 03-11-08 13:11:04 - **enlace**
   Estado: 200 OK - tipo: 'text/javascript'
   url: 'http://localhost/skylib/tag/auto.php?s=1&t=500&
   nocache=0.7182262658325766'
5. 03-11-08 13:11:05 - **enlace**
   Estado: 302 Found - tipo: ''
   url: 'http://www.google.com/'
6. 03-11-08 13:11:05 - **enlace**
   Estado: 200 OK - tipo: 'text/html'
   url: 'http://localhost/ejemplo/ok.html'
7. 03-11-08 13:11:05 - **enlace**
   Estado: 404 Not Found - tipo: 'text/html; charset=iso-8859-1'
   url: 'http://localhost/ejemplo/no_existe.html'
8. 03-11-08 13:11:05 - **enlace**
   Estado: 200 OK - tipo: 'text/html'
   url: 'http://localhost/ejemplo/formulario.html'
9. 03-11-08 13:11:05 - **enlace**
   Estado: 200 OK - tipo: 'text/html'
   url: 'http://localhost/ejemplo/ajax.html'
10. 03-11-08 13:11:05 - **click**
    ID: 'enlace ok' - Tipo: A
    href: 'http://localhost/ejemplo/ok.html'
11. 03-11-08 13:11:06 - **keyup**
    shift + O
12. 03-11-08 13:11:06 - **keyup**
    K
13. 03-11-08 13:11:07 - **click**
    ID: 'Ir a form' - Tipo: A
    href: 'http://localhost/ejemplo/formulario.html'

---

14. 03-11-08 13:11:07 - **pagina cargada**
    url: 'http://localhost/ejemplo/formulario.html'
15. 03-11-08 13:11:08 - **enlace**
    Estado: 200 OK - tipo: 'text/css'
    url: 'http://localhost/ejemplo/style.css'
16. 03-11-08 13:11:08 - **enlace**
    Estado: 200 OK - tipo: 'text/javascript'
    url: 'http://localhost/skylib/tag/lib_js.php?id=1'
17. 03-11-08 13:11:08 - **enlace**
    Estado: 200 OK - tipo: 'text/javascript'
    url: 'http://localhost/skylib/tag/auto.php?s=1&t=500&
    nocache=0.5787446962560953'
18. 03-11-08 13:11:08 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/enlaces.html'
19. 03-11-08 13:11:08 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/ajax.html'
20. 03-11-08 13:11:08 - **carga de imagen**
    alt: 'foto1'
    src: 'http://localhost/ejemplo/nutria.jpg'
21. 03-11-08 13:11:08 - **error en imagen**
    alt: 'foto2'
    src: 'http://localhost/ejemplo/noexiste.jpg'
22. 03-11-08 13:11:08 - **carga de imagen**
    alt: 'foto3'
    src: 'http://localhost/ejemplo/claymore.jpg'
23. 03-11-08 13:11:09 - **reset**
    ID: 'form1'
    action: 'http://localhost/ejemplo/procesa_form.php'
24. 03-11-08 13:11:12 - **submit**
    ID: 'form1'
    action: 'http://localhost/ejemplo/procesa_form.php'

---

25. 03-11-08 13:11:13 - **pagina cargada**
    url: 'http://localhost/ejemplo/ajax.html'
26. 03-11-08 13:11:13 - **enlace**
    Estado: 200 OK - tipo: 'text/css'
    url: 'http://localhost/ejemplo/style.css'
27. 03-11-08 13:11:13 - **enlace**
    Estado: 200 OK - tipo: 'text/javascript'
    url: 'http://localhost/skylib/tag/lib_js.php?id=1'
28. 03-11-08 13:11:13 - **enlace**
    Estado: 200 OK - tipo: 'text/javascript'
    url: 'http://localhost/skylib/tag/auto.php?s=1&t=500&
    nocache=0.35769671512868506'
29. 03-11-08 13:11:14 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/enlaces.html'
30. 03-11-08 13:11:14 - **enlace**
    Estado: 200 OK - tipo: 'text/html'
    url: 'http://localhost/ejemplo/formulario.html'
31. 03-11-08 13:11:13 - **ajax open**
    Información: POST Asinc=true
    url: 'ejemploajax3.html'
32. 03-11-08 13:11:13 - **ajax send**
    parámetros: 't1=Campo1'
33. 03-11-08 13:11:13 - **click**
    ID: 'ajax_post' - Tipo: INPUT
    href: ''
34. 03-11-08 13:11:14 - **ajax send**
    parámetros: 'null'
35. 03-11-08 13:11:14 - **ajax open**
    Información: GET Asinc=true
    url: 'no_existe.html'
36. 03-11-08 13:11:14 - **click**
    ID: 'ajax_noresponse' - Tipo: INPUT
    href: ''
37. 03-11-08 13:11:14 - **ajax readystatechange 4**
    Estado 200 OK
38. 03-11-08 13:11:14 - **ajax readystatechange 4**
    Estado 404 Not Found

Figura #55: Eventos registrados durante la reproducción de una sesión

Una validación importante que se debió realizar es la que ocurre ante el intento de reproducción de una sesión no existente. La Figura #56 muestra el mensaje de alerta generado por la herramienta cuando se intenta reproducir la sesión con ID 5 la cual no existe.

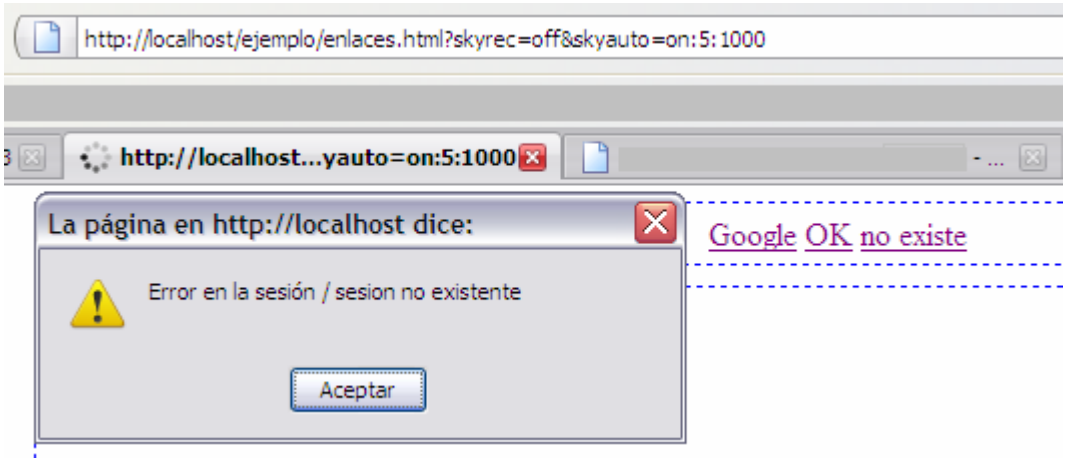

Figura #56: Alerta ante intento de reproducción de sesión no registrada

Conclusiones

El objetivo del presente Trabajo Especial de Grado fue cumplido, el cual consistió en el desarrollo de una herramienta de fácil integración para realizar monitoreo de eventos y acciones de los usuarios en aplicaciones Web. Se logró obtener un producto que cumple con los requerimientos planteados al comienzo de este trabajo y que genera un impacto mínimo en las aplicaciones Web involucradas.

Es de gran importancia destacar la versatilidad alcanzada mediante la integración del lado del cliente, por medio de JavaScript, entre la herramienta de monitoreo y las aplicaciones Web. Al utilizar este mecanismo se logra facilidad, rapidez e independencia de la plataforma tecnológica en el proceso de integración, cumpliendo con las expectativas en la investigación.

JavaScript resultó ser una herramienta poderosa de mucha utilidad en el desarrollo de la aplicación. Mediante este lenguaje, común en los navegadores de la actualidad, se pudo realizar el proceso de integración (Usando Page tagging) y la captura y automatización de eventos.

El código elaborado de la herramienta tuvo que ser reescrito y adaptado para que funcionara de manera correcta en diferentes navegadores, debido a las diversas implementaciones que estos poseen de JavaScript.

Para poder llevar a cabo el proceso de automatización de sesiones registradas, se tuvieron que definir los elementos que forman parte de esta simulación y la forma como éstos fueron construidos y ejecutados, de manera tal que resultara tanto útil como simple para el usuario que utilice la herramienta.

El seguimiento de las pautas establecidas en la metodología de desarrollo propuesta contribuyó, en gran medida, con el cumplimiento de los objetivos de la investigación. Se debe tomar en cuenta que dicha metodología posee etapas, con alcances bien delimitados, los cuales permitieron organizar el proceso de desarrollo de la herramienta, pudiendo llevar así un completo control de las tareas realizadas,

rigiendo el conjunto de pasos que deben ser llevados a cabo para realizar una aplicación de este tipo.

Esta herramienta realiza monitoreo sobre una serie de aspectos de las aplicaciones Web tales como acciones de los usuarios, peticiones AJAX, carga de imágenes entre otros, permitiendo generar información de utilidad, que pueda ser utilizada como entrada para múltiples propósitos entre los que se pueden mencionar:

- Conocer el comportamiento y poder categorizar los usuarios con la posibilidad de determinar sus intereses, habilidades y limitaciones.

- Realizar pruebas de usabilidad utilizando las acciones registradas como mecanismo de observación de los usuarios.

- Detectar posibles errores, comportamientos y funcionalidades de interés para los usuarios que la aplicación pueda poseer

- Depurar errores replicando las acciones, registradas previamente, que produjeron dichos errores.

El desarrollo de este Trabajo Especial de Grado ofrece un nuevo aporte y contiene un considerable valor académico y profesional en el área de la tecnología Internet, al involucrar conceptos relacionados con el área monitoreo de eventos, automatización, Page Tagging y aplicaciones Web.

Recomendaciones

Se recomienda, en investigaciones posteriores, ampliar el conjunto y la variedad de información en los reportes que son generados con esta herramienta, orientada hasta ahora a la comparación entre sesiones.

La herramienta actualmente registra un conjunto pequeño de eventos generados por el usuario y cierta información como la disponibilidad de enlaces y carga de imágenes. Se recomienda, con el fin de mejorar la información que provee la herramienta, ampliar el conjunto de eventos e información que esta puede registrar (interacciones con Flash, por ejemplo), explotando las ventajas de JavaScript y otras tecnologías disponibles.

Actualmente, la herramienta maneja el acceso a los elementos asociados a los diferentes eventos por medio del atributo "id" del HTML. Si dicho atributo no se encuentra presente la herramienta puede no funcionar de manera correcta al realizar acciones como la simulación automatizada. Se recomienda desarrollar mecanismos alternos para identificar y acceder a los elementos mencionados.

El tipo de archivo tanto del script de integración como de la imagen solicitada para realizar el registro de eventos posee la extensión .php en la actualidad. Se recomienda configurar el servidor Web, donde se aloja la herramienta, para que maneje los archivos de otras extensiones (.js, .gif) de manera tal que permitan ejecutar acciones del lado del servidor y eliminen la necesidad de extensiones no naturales para los archivos mencionados. Esto con la finalidad de evitar ciertos errores indeseados de compatibilidad que algunas configuraciones puedan ocasionar.

Adicionalmente se recomienda crear un sistema de manejo de usuarios con privilegios de acceso determinados, para que la herramienta pueda ser vista como un servicio, alojado en una sola localidad y utilizada para monitorear las acciones de las diferentes aplicaciones pertenecientes a diversos usuarios independientes unos de otros.

Referencias Bibliográficas

[1] Wikipedia. *Web Application*. [Documento en línea] Disponible: http://en.wikipedia.org/wiki/Web_application [Consulta: julio 13, 2008]

[2] Sommerville, I. (2005) *Ingeniería del Software.* Séptima edición. Madrid: PEARSON EDUCACIÓN, S.A.

[3] Antelo, E. (2004) *Diseño de aplicaciones en Tres Capas*. [Documento en línea] Disponible: http://www.geocities.com/trescapas/TresCapas.htm [Consulta: octubre 31, 2008]

[4] Vegas, J. (2002) *Creación de un Portal Web Docente* [Documento en línea] Disponible: http://www.infor.uva.es/~jvegas/cursos/buendia/pordocente/ [Consulta: octubre 31, 2008]

[5] Nielsen, J. (2003) Usability 101: Introduction to Usability [Documento en línea] Disponible: http://www.useit.com/alertbox/20030825.html [Consulta: octubre 31, 2008]

[6] Shneiderman, B. (1998) *Designing the User Interface. Strategies for Effective human-Computer Interaction*. Tercera edición. ADDISON-WESLEY.

[7] www.crnti.edu.uy/05trabajos/interface/Arano-Rodriguez-modelado.doc

[8] Exforsys Inc. *Challenges in Testing Web Based Applications*. [Documento en línea] Disponible: http://www.exforsys.com/tutorials/testing/challenges-in-testing-web-based-applications.html [Consulta: agosto 5, 2008]

[9] www.users.cs.umn.edu/~safonov/publications/webnet2001/WebAutomationChallenges.doc

[10] iOpus Inc. *iMacros*. [Documento en línea] Disponible:

http://www.iopus.com/imacros/ [Consulta: agosto 27, 2008]

[11] iOpus Inc. *iMacros for Firefox*. [Documento en línea] Disponible: https://addons.mozilla.org/en-US/firefox/addon/3863 [Consulta: agosto 27, 2008]

[12] Kartmann E. (2005) *Web Replay 2 – Automated Web Testing*. [Documento en línea] Disponible: http://www.codeproject.com/KB/applications/WebReplay 2.aspx?display=Print [Consulta: agosto 27, 2008]

[13] sclANALYTICS. *Should I Use Page Tags, Log Files or a Hybrid Solution?* [Documento en línea] Disponible: http://www.sclanalytics.com/info/page-tags-or-log-files [Consulta: agosto 27, 2008]

[14] Stratigent. *Data Collection – Single Methodologies*. [Documento en línea] Disponible: http://www.stratigent.com/web-sight-newsletter/web-analytics-newsletter-archive/data-collection-single/default.html [Consulta: agosto 27, 2008]

[15] Eguiluz, J. *Introducción a Javascript*. [Documento en línea] Disponible: http://www.librosweb.es/javascript/ [Consulta: julio 13, 2008]

[16] W3 Schools. *HTML DOM Tutorial*. [Documento en línea] Disponible: http://www.w3schools.com/htmldom/ [Consulta: agosto 7, 2008]

[17] Wikipedia. *AJAX*. [Documento en línea] Disponible: http://es.wikipedia.org/wiki/AJAX [Consulta: agosto 7, 2008]

[18] Wikipedia. *PHP*. [Documento en línea] Disponible: http://es.wikipedia.org/wiki/PHP [Consulta: agosto 7, 2008]

[19] Wikipedia. *MySQL*. [Documento en línea] Disponible: http://es.wikipedia.org/wiki/Mysql [Consulta: agosto 7, 2008]

[20] MySQL. *MySQL 5.0 Reference Manual.* [Documento en línea] Disponible: http://dev.mysql.com/doc/refman/5.0/es/index.html [Consulta: agosto 7, 2008]