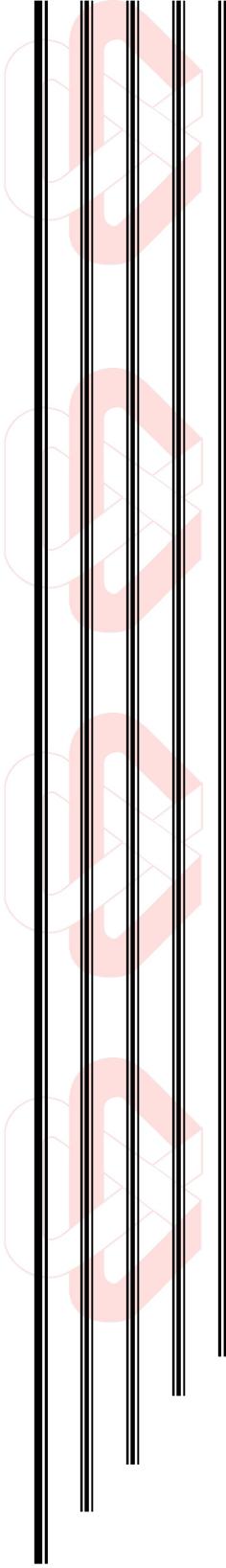


Centro de Investigación en Matemáticas, A.C.

CIMAT



Comparativa de Herramientas para aplicar pruebas de estrés y desempeño para aplicaciones web

REPORTE TÉCNICO

Que para obtener el grado de

Maestro en Ingeniería de Software

P r e s e n t a

Diego Alberto Falcón Cordero

Director(a) de Reporte Técnico

José Guadalupe Hernández Reveles

Zacatecas, Zacatecas., 22 de julio de 2011

Agradecimientos

A mis padres, por su amor y apoyo incondicional, por el ejemplo que siempre nos han dado a mi hermano y a mí.

A mi familia y amigos, por su apoyo y sus palabras de aliento.

A José Guadalupe Hernández, por su orientación y apoyo en la realización del presente reporte, y la obtención de los artículos utilizados en el mismo.

AL Dr. Cuauhtémoc Lemus, por la amistad y el apoyo que nos ha brindado durante estos dos años.

A Juan Gabriel Hernández, por su ayuda en la obtención de artículos para la realización de este reporte.

A mis compañeros de generación, Paty, Ericka, Enrique y Jesús, por las experiencias vividas durante la maestría, gracias por brindarme su amistad.

A las personas que laboran y estudian en CIMAT Unidad Zacatecas, gracias por su amistad y apoyo.

Índice

I.	Introducción.....	1
I.1.	Descripción del Tema Seleccionado	1
I.2.	Objetivo	1
II.	Estado del Arte.....	2
II.1.	Evolución de las Pruebas de Software.....	2
II.1.1.	Pruebas de Software 1.0.....	2
II.1.2.	Pruebas de Software 2.0.....	2
II.1.3.	Pruebas de Software 3.0.....	3
II.2.	Evolución de las Pruebas de Software en Base a Procesos	3
II.2.1.	Modelos de Mejora para el Proceso de Pruebas	3
II.2.1.1.	Test Process Improvement (TPI).....	3
II.2.1.2.	Test Maturity Model integration (TMMi).....	4
II.3.	Pruebas en Aplicaciones Web.....	5
II.3.1.	Pruebas de Desempeño	5
II.3.1.1.	Prueba de Carga	6
II.3.1.2.	Prueba de Estrés	6
II.3.1.3.	Pruebas de Durabilidad/Resistencia	7
II.3.1.4.	Conceptos relacionados a Pruebas de Desempeño.....	7
II.4.	Planeación de la Capacidad	8
II.4.1.	Proceso de Planeación de la Capacidad.....	8
II.4.1.1.	Entender la Arquitectura de Servicio.....	9
II.4.1.2.	Caracterizar la carga de trabajo.....	10
II.4.1.3.	Obtener los parámetros del modelo	10
II.4.1.4.	Pronosticar la evolución de la carga de trabajo	11
II.4.1.5.	Desarrollar el modelo de desempeño.....	11
II.4.1.6.	Calibrar y Validar los Modelos.....	11
II.4.1.7.	Predecir el desempeño del servicio	11
II.4.1.8.	Analizar los intercambios entre costo y desempeño.....	12
II.5.	Retorno de la Inversión	12
II.6.	Herramientas para la realización de pruebas de Desempeño/Estrés	12

Comparativo de herramientas para aplicar pruebas de estrés y desempeño para aplicaciones Web

II.6.1.	WAPT (Web Application Testing)	13
II.6.1.1.	Características de WAPT	13
II.6.2.	Jakarta Jmeter	14
II.6.2.1.	Características de Jmeter	15
III.	Análisis Comparativo.....	16
III.1.	Análisis Comparativo de Herramientas	16
III.1.1.	Características a Medir en el Análisis Comparativo	16
III.1.2.	Realización de las Pruebas	17
III.1.2.1.	Resultados del Análisis del Sistema PREP	18
III.1.2.2.	Prueba con WAPT	21
III.1.2.2.1.	Generar un Escenario de Prueba	22
III.1.2.2.2.	Grabación de un Perfil de Usuario.....	23
III.1.2.2.3.	Método de Prueba de WAPT	23
III.1.2.2.4.	Resultados en Tiempo Real	24
III.1.2.2.5.	Reportes/Gráficos/Registros de Salida	26
III.1.2.3.	Pruebas con JMeter	28
III.1.2.3.1.	Creación de un Grupo de hilos	29
III.1.2.3.2.	Configuración de un Proxy	29
III.1.2.3.3.	Elementos Listener	31
III.1.2.3.4.	Método de Prueba de JMeter	32
III.1.2.3.5.	Resultados en Tiempo Real	32
III.1.2.3.6.	Reportes/Graficas/Registros de Salida.....	34
III.1.3.	Resumen de Análisis Comparativo	34
III.1.4.	Resultados del Análisis Comparativo.....	35
III.1.5.	Resultados de las Pruebas.....	36
IV.	Conclusiones	37
IV.1.	Trabajo Futuro.....	38
V.	Bibliografía y Referencias	39

Índice de Figuras

Figura 1- Modelo TPI	4
Figura 2 – Proceso de Planeación de la Capacidad	9
Figura 3 – Primer Prueba, Gráfica de Desempeño.....	19
Figura 4 – Ultima Prueba – Grafica de Desempeño.....	20
Figura 5 – Primer Prueba, Grafica de Errores.....	20
Figura 6 – Ultima Prueba, Grafica de Errores	21
Figura 7 – Interfaz Grafica WAPT	21
Figura 8 – Asistente de Configuración para un Escenario de Pruebas.....	22
Figura 9 – Navegador WAPT	23
Figura 10 – WAPT, Grafica de Desempeño.....	26
Figura 11 – WAPT, Reporte de Prueba.....	27
Figura 12 – Interfaz Grafica JMeter	28
Figura 13 – Jmeter, Configuración de un Grupo de Hilos	29
Figura 14 – Jmeter, Configuración de Servidor proxy.....	30
Figura 15 – Configuración de Proxy, Mozilla Firefox.....	31
Figura 16 – Jmeter, Gráfica de Desempeño	33
Figura 17 – Jmeter, Log de Resultados.....	33

Índice de Tablas

Tabla 1 – Conceptos Relacionados a Desempeño	7
Tabla 2 – Primeras Pruebas con JMeter.....	18
Tabla 3 – Pruebas Finales con JMeter.....	19
Tabla 4 – Variables para Monitoreo de Pruebas de WAPT	24
Tabla 5 – Jmeter, Listeners.....	32
Tabla 6 – Comparativo de Herramientas	34

I. Introducción

I.1. Descripción del Tema Seleccionado

El objetivo del presente reporte es utilizar un conjunto de herramientas para realizar pruebas de desempeño y estrés a una aplicación web, de tal forma que se pueda establecer una guía para implementar esta práctica para nuevas aplicaciones.

El documento se divide en tres partes.

- En la primer parte del documento se describen conceptos básicos sobre pruebas en el ámbito de desarrollo de software, en específico aplicaciones web. Algunos estándares existentes que ayudan a medir la calidad del proceso de pruebas, y algunas herramientas que sirven para la automatización de pruebas.
- La segunda parte contará con una tabla comparativa de dos herramientas que sirven para la automatización de pruebas de desempeño en aplicaciones web. Esto con la finalidad de determinar cómo se pueden complementar para la realización de pruebas de desempeño y obtener el mejor resultado posible.

I.2. Objetivo

Describir y comparar a través de un caso de estudio el funcionamiento de las herramientas como Jmeter y WAPT aplicando pruebas de estrés y desempeño a aplicaciones web.

- Descubrir el número de usuarios concurrentes que el servidor puede soportar.
- Probar si el servidor es capaz de atender 3000 peticiones en un periodo de 15 minutos.

II. Estado del Arte

II.1. Evolución de las Pruebas de Software

De acuerdo a la **Real Academia Española**, una prueba se puede definir de la siguiente forma: “Ensayo o experimento que se hace de algo, para saber cómo resultará en su forma definitiva.”

Para el desarrollo de software, una prueba consta de los procesos que permiten verificar y revelar la calidad de un producto de software. Se utilizan para identificar posibles fallos de implementación, calidad o usabilidad en un programa.

II.1.1. Pruebas de Software 1.0

En esta etapa, las pruebas eran consideradas como una tarea complementaria y eran realizadas por personal poco calificado, el cual tomaba esta tarea como un punto más dentro del proceso de desarrollo de software.

Las pruebas aplicadas en este tiempo eran pocas, así como los métodos para lograr la automatización de pruebas. Dichas herramientas eran caras, complejas y de difícil uso, además de la ineeficacia que se tenía para abordar las necesidades de productividad, no existía una división o parte del equipo que se encargara administrativamente de las pruebas de software (**Navarro 2010**).

II.1.2. Pruebas de Software 2.0

Se reconoce que las pruebas de software forman parte importante del desarrollo, se establece una moda y todos comienzan a realizar pruebas en serio. El problema se presenta para localizar desde el punto de vista de la organización y la administración, lo referente al presupuesto, dirección y toma de decisiones.

El número de herramientas para pruebas se incrementa en esta etapa. Existían herramientas que mostraban ser más una distracción para el encargado de la realización de las pruebas, ya que gastaba más tiempo tratando de seleccionar la herramienta adecuada, así que se descuidaban los objetivos, arquitectura y dirección de las pruebas. La comprensión y participación de la dirección ejecutiva eran rudimentarias en esta etapa (**Navarro 2010**).

II.1.3. Pruebas de Software 3.0

Considero que actualmente no encontramos en una nueva etapa para las pruebas de software, donde las organizaciones ya cuentan con áreas dedicadas a las pruebas, cuentan con sus propios métodos y procesos, además, existen bastantes herramientas que permiten la realización de pruebas de manera automática.

II.2. Evolución de las Pruebas de Software en Base a Procesos

En ocasiones, las pruebas son consideradas como un proceso caro y sin control, toman demasiado tiempo, sus costos se elevan mucho más de lo planeado, y no ofrecen una buena concepción de la calidad del proceso mismo. Asimismo, la calidad de la información del sistema y los riesgos de negocio son difíciles de determinar (**Navarro 2010**).

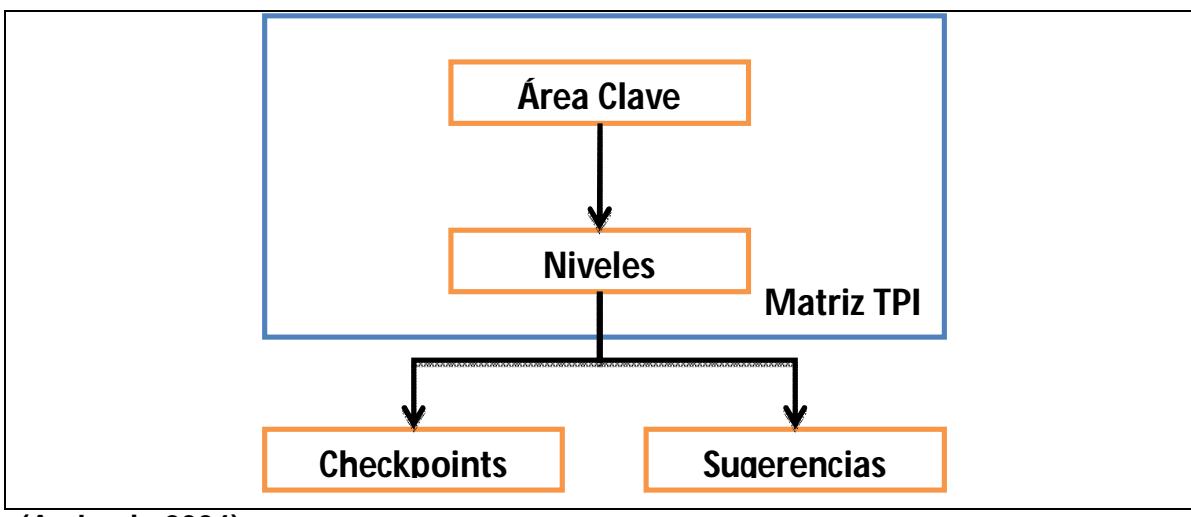
II.2.1. Modelos de Mejora para el Proceso de Pruebas

En la industria se han desarrollado modelos para el desarrollo de software (Espiral, RAD, etc.), que han ayudado a llevar de una manera más ordenada y documentada dicho desarrollo, de tal manera que las pruebas de software no pueden quedar fuera. Y gracias a la importancia que este proceso ha tomado en la tarea de desarrollo, se han desarrollado modelos que ayudan a la ejecución de pruebas de manera más ordenada y consistente. Entre estos modelos destacan el modelo de Mejora de Procesos de Prueba (TPI), y el Modelo de Integración y Madurez de las Pruebas (TMMI).

II.2.1.1. Test Process Improvement (TPI)

TPI trata de ver la fase de pruebas desde varios puntos de vista, los cuales son llamados áreas Clave, cada una dividida en niveles de madurez (de A hasta D). No todas son igual de importantes, y puede que existan algunas dependencias entre áreas, en la siguiente figura se muestra la estructura básica de los elementos que conforman el modelo TPI (**Andersin 2004**).

Figura 1- Modelo TPI



(Andersin 2004)

Cada proceso necesita cubrir ciertas áreas para poder demostrar que está bien definido, estas Áreas Clave, son las base para la mejora del proceso de pruebas, TPI consta de 20 Áreas para determinar la madurez del proceso de pruebas.

Este modelo contiene un Área llamada herramienta de pruebas, la cual esta encargada de la automatización de las pruebas del software, esto para lograr una mejora del proceso, esta mejora a través de la atomización se puede reflejar de varias maneras, como un menor consumo de recursos o tiempo, mejora en la cobertura de las pruebas, motivación del personal, etc. (**Navarro, 2010**).

II.2.1.2. Test Maturity Model integration (TMMi)

Creado por la TMMi Foundation, como un modelo de apoyo para CMMI (Capability Maturity Model integration), para el soporte de actividades de pruebas y la mejora del proceso de pruebas para disciplinas como ingeniería de sistemas e ingeniería de software.

TMMi provee un marco de trabajo para ser usado como modelo de referencia durante la mejora del proceso de pruebas. Y de acuerdo a una arquitectura para la mejora del proceso de pruebas, contiene niveles, a través de los que la organización debe pasar mientras su proceso de pruebas evoluciona.

TMMi cuenta con cinco niveles: Inicial, Administrado, Definido, Medido y optimizado, y cada uno de estos niveles está provisto por áreas de proceso, que

deberán cumplirse para poder obtener el grado deseado (**TMMi Foundation 2010**).

La utilización de herramientas de pruebas puede ser apoyada por el área clave Optimización del Proceso de Pruebas, ya que en ella se debe identificar nuevas tecnologías de pruebas, y determinar si son apropiadas para la implementación dentro de la organización.

II.3. Pruebas en Aplicaciones Web

Las pruebas para aplicaciones de software convencionales se han definido a través de los años con una serie de métodos, cada uno implementado con una estrategia apropiada, lamentablemente te este tipo de métodos son inútiles para probar aplicaciones web, ya que estas varían su funcionalidad, presentación y potenciales usuarios, de esta manera, tenemos que las aplicaciones web son dinámicas por naturaleza y requieren una metodología fuerte para su prueba. Las pruebas a este tipo de aplicaciones involucran probar la funcionalidad como en un sistema convencional, la capa de presentación y el desempeño. Durante estas pruebas se pueden presentar muchas situaciones que se deben registrar (**Subraya, Subrahmanyam 2000**).

II.3.1. Pruebas de Desempeño

El desempeño de un sistema, es una cualidad difícil de entender, ya que es afectada por cada aspecto del diseño, código y ambiente de ejecución. Y puede llegar a causar desde pequeños retrasos, hasta el abandono del desarrollo de un sistema (**Woodside, Franks, Petriu, 2007**).

Por pruebas de desempeño, nos referimos a todas las actividades centradas en la evaluación de cómo se espera que sea el comportamiento del sistema, en la perspectiva del cliente, esto se traduce en términos de salidas, tiempo de estímulo-respuesta, o una combinación de ambos (**Weyuker, Volokos, 2000**).

A menudo, los principales problemas que se reportan de un sistema una vez que se ha liberado, se refieren al degradado del desempeño del sistema, ya que aunque el sistema pasó por una serie extensiva de pruebas para cumplir con su funcionalidad deseada, nunca fue probado para cumplir el desempeño deseado (**Weyuker, Volokos, 2000**).

A pesar del gran impacto que tiene el desempeño de un sistema en su éxito/fallo, los equipos de proyectos de software en el ámbito industrial, utilizan pocos o nulos recursos y tiempo en la realización de pruebas de desempeño, en comparación con los recursos utilizados en probar la funcionalidad del sistema. Los problemas encontrados al realizar pruebas de desempeño difieren bastante de los problemas relacionados a la funcionalidad (**Weyuker, Volokos, 2000**).

Según Subraya, Subrahmanyam (2000), Las pruebas de desempeño se realizan grabando acciones que un usuario real haría, después dichas acciones son replicadas en el sistema de una manera automatizada y controlada. De acuerdo al objetivo que se quiere llegar, existen tres variaciones a la prueba de desempeño:

II.3.1.1. Prueba de Carga

Provee una vista de cómo trabaja internamente el sitio web en una situación rutinaria. Se modela el comportamiento del usuario en un mundo real. El script de prueba debe imitar el comportamiento que los usuarios realizan comúnmente, esto incluye tiempos de retraso (think time delay) y tasas de llegada acordes al mundo real. El comportamiento de la aplicación dependerá del tipo de sistema a probar. En algunas aplicaciones la prueba podrá correr por un periodo corto de tiempo, dependiendo de la funcionalidad que se esté probando.

La gran parte de las acciones en la web se completan en segundos, así que una prueba que dure de 10 a 30 segundos es adecuada para simular un estado de carga estable, ejecutar la funcionalidad deseada y obtener un conjunto de resultados grande (**Subraya et.al. 2000**).

II.3.1.2. Prueba de Estrés

Pruebas enfocadas en determinar o validar las características de desempeño de un sistema expuesto a condiciones más allá de las anticipadas para producción (**Toledo et al., 2008**).

Probar un sistema enfatizando aspectos como la robustez, disponibilidad, manejo de errores, etc. Bajo una carga pesada, el objetivo de estas pruebas es de asegurar que el sistema no colapsara cuando disponga de pocos recursos ó exista una gran concurrencia de usuarios (**Wikipedia**).

Los sistemas bajo cargas pesadas de información muestran grandes variaciones en su desempeño, lo que contribuye a obtener resultados estadísticos imprecisos (**Woodside et al., 2007**).

II.3.1.3. Pruebas de Durabilidad/Resistencia

Es una prueba de larga duración, ya sea de carga o estrés. En lugar de un periodo de ejecución de 10 minutos, estas pruebas se ejecutan por horas, o incluso días. Con estas pruebas se revelaran problemas como fugas pequeñas en la memoria, acumulación de transacciones sin compromiso en la base de datos en un buffer. Este tipo de pruebas son requeridas para de gran importancia que deben sostenerse por periodos largos de tiempo. Es difícil realizar estas pruebas en el lugar de producción del sistema, pero pueden realizarse como parte de las pruebas de aceptación (**Subraya et.al. 2000**).

II.3.1.4. Conceptos relacionados a Pruebas de Desempeño

Una vez realizadas las pruebas, hay que tomar en cuenta los resultados arrojados para realizar un análisis de resultados, dichos resultados están directamente relacionados a estas pruebas, y algunos de los más destacados son:

Tabla 1 – Conceptos Relacionados a Desempeño

Termino	Definición
Tiempo de Respuesta	Variable que permite medir el tiempo que tarda la respuesta de un objeto que ha sido probado (Subraya et.al. 2000).
Tiempo de Procesamiento	Permite la medición del tiempo que tarda el servidor en procesar una petición hecha.
Sesión	Espacio tiempo donde un usuario ejecuta las acciones programadas mediante la herramienta de pruebas (Subraya et.al. 2000).
Errores de Red	Errores producidos por cualquier tipo de fallo en la red.
Errores Http	Errores producidos por el servidor donde se aloja la aplicación web.
Timeout	Una vez que se cumple el tiempo máximo configurado para atender una petición, el servidor mostrara un mensaje de error en vez de seguir atendiendo la petición hecha.

II.4. Planeación de la Capacidad

Es muy común en organizaciones que manejan Tecnologías de Información, donde los problemas relacionados al desempeño del sistema son corregidos una vez que el sistema se ha puesto en marcha y algún usuario identifica y reporta dicho error. En un mundo perfecto, los administradores prepararían por anticipado la planeación de capacidad, y así evitar problemas de desempeño como cuellos de botella (**Rich, Hill, 2010**).

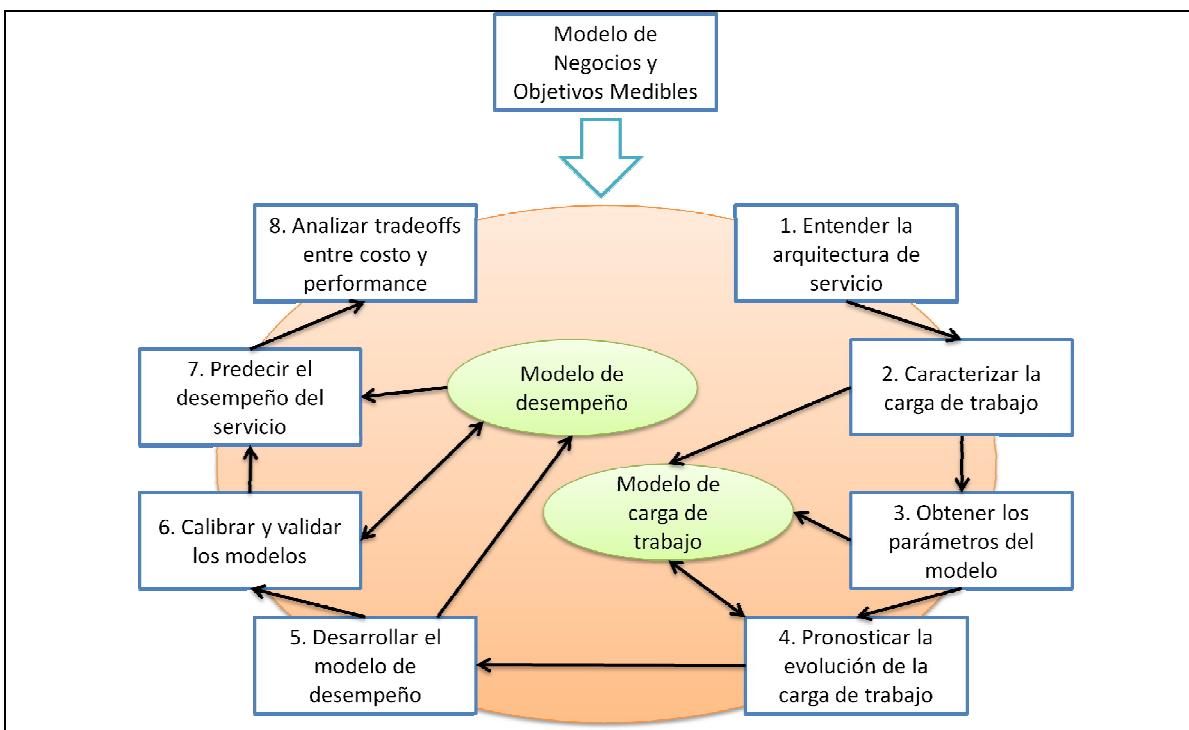
El objetivo de la planeación de la capacidad es proveer niveles de servicio satisfactorios a los usuarios de un sistema, de una forma efectiva y económica (**Rich et. al, 2010**).

Cabe mencionar que esto debe ser parte de la planeación del mismo sistema, de tal manera que las pruebas de desempeño sirvan para corroborar que se cumple con el desempeño deseado, o determinar ajustes para lograr dichos requerimientos.

II.4.1. Proceso de Planeación de la Capacidad

La planeación para un servicio web requiere el seguimiento de una serie de pasos de forma sistemática, dicho proceso comienza con el modelo de negocios y sus objetivos medibles, de los cuales se conforman los objetivos de nivel de servicio (**Almeida, Menascé, 2002**).

Figura 2 – Proceso de Planeación de la Capacidad



(Almeida et. al, 2002)

II.4.1.1. Entender la Arquitectura de Servicio

Lo primer paso de un análisis cuantitativo implica obtener un profundo entendimiento de la arquitectura de servicio. Lo que conlleva a contestar las siguientes preguntas:

- ¿Cuáles son los requerimientos del modelo de negocios del sistema?
- ¿Cuál es la configuración del sitio en cuestión de servidores y conectividad interna?
- ¿Internamente, con cuantas capas cuenta el sitio?
- ¿En qué tipo de servidor(es) corre el sitio?
- ¿Qué tipo de software utiliza cada servidor?
- ¿Cuán escalable y confiable es la arquitectura?

Una vez contestadas estas preguntas, este paso producirá una descripción sistemática del ambiente de red, sus componentes y servicios (**Almeida et. al, 2002**).

II.4.1.2. Caracterizar la carga de trabajo

Las sesiones, son los bloques de construcción de las cargas de trabajo en los negocios electrónicos, implican una secuencia de peticiones para ejecutar funciones del negocio. Un cliente realizará las peticiones durante una sola visita al sitio dependiendo de la funcionalidad del mismo.

Cada petición ejecutada en el sitio utiliza los recursos del mismo de manera diferente. Algunos servicios pueden utilizar grandes cantidades de tiempo para procesamiento por parte del servidor, y algunas otras se pueden concentrar en el servidor de base de datos. El comportamiento puede diferir de un cliente a otro.

El entender el comportamiento del cliente es necesario para obtener los objetivos de negocio, de la misma forma lo es para dimensionar el tamaño de los recursos del sitio. Para lo cual es necesario la caracterización, en forma de patrones navegación, además de la caracterización de la frecuencia en que cada tipo de sesión será iniciada, para indicar la intensidad de la carga de trabajo (**Almeida et. al, 2002**).

II.4.1.3. Obtener los parámetros del modelo

Consiste en recolectar los parámetros para el modelo de carga de trabajo, también se debe monitorear y medir el desempeño del servicio web, de forma que se pueda garantizar la calidad de servicio y prevenir problemas.

Una medida de desempeño puede provenir de diferentes puntos de referencia, y es necesario escoger cuidadosamente las medidas de desempeño que se monitorearan.

La información recolectada deberá ayudar a responder lo siguiente:

- ¿Cuántos clientes visitan el sitio en un día?
- ¿Cuál son los índices pico y promedio de tráfico del sitio?
- ¿Qué recurso es el más solicitado por las peticiones dentro de la infraestructura del TI?

(Almeida et. al, 2002)

II.4.1.4. Pronosticar la evolución de la carga de trabajo

Se debe proveer respuesta a preguntas como:

- ¿Cuánto variara el número de clientes durante los próximos seis meses?
- ¿Cuántos usuarios en paralelo podrán tener acceso a los servicios los próximos seis meses?

(Almeida et. al, 2002)

II.4.1.5. Desarrollar el modelo de desempeño

Se utilizaran técnicas cuantitativas y modelos analíticos basados en teoría de colas para desarrollar modelos de desempeño, dichos modelos podrán predecir el desempeño después de algún cambio en la carga de trabajo o la arquitectura del sitio (Almeida et. al, 2002).

II.4.1.6. Calibrar y Validar los Modelos

El modelo de desempeño es válido si las métricas de desempeño calculadas por el modelo concuerdan con las mediciones actuales del sistema, dentro de un margen aceptable de error. Para el proceso es aceptable una variación de 10% al 30% (Almeida et. al, 2002).

II.4.1.7. Predecir el desempeño del servicio

La predicción es la clave de la planeación de la capacidad, ya que se debe determinar como el servicio reaccionara cuando los niveles de carga cambien, o al desarrollar nuevos modelos de negocio. Esto necesita de modelos predictivos, así que se debe predecir el desempeño del servicio bajo distintos escenarios (Almeida et. al, 2002).

II.4.1.8. Analizar los intercambios entre costo y desempeño

Se realiza un análisis de varias arquitecturas, para determinar cuál es la mejor de acuerdo a su costo-efectividad. Se deberán considerar escenarios futuros de carga de trabajo, costos del sitio, calidad del servicio percibida por el cliente, al final esto servirá para indicar que acciones garantizaran que los servicios de TI lograran obtener sus objetivos de negocio (**Almeida et. al, 2002**).

II.5. Retorno de la Inversión

El término se refiere a una medida de desempeño, en la cual se mide la efectividad que tiene una inversión, y es calculada tomando la ganancia de la inversión, y dividiéndola sobre el monto invertido, esta medida es expresada en porcentaje (**Investopedia**).

SEI, a través de un estudio sobre los beneficios de utilizar procesos de mejora basados en CMM, obtuvieron como resultado, reportes que indican un retorno de la inversión en un rango entre 400% y 880%, con un retorno promedio de 500% (**Butler, 1995**)

En lo que corresponde al caso de estudio, no se han realizado este cálculo, ya que no se cuentan con los datos necesarios.

II.6. Herramientas para la realización de pruebas de Desempeño/Estrés

Es importante destacar, que en el desarrollo de software, se puede ejecutar una prueba, sin importar el tipo, tanto de manera manual como automática.

En una prueba de tipo manual se hace indispensable el usuario (tester), el cual se encargara del desarrollo e la prueba, sin el apoyo de alguna herramienta de automatización. En este tipo de pruebas el tester deberá contar con ciertas cualidades, tales como paciencia, ser observador y creativo entre otras (**Navarro, 2010**).

Las pruebas manuales se enfocan generalmente a la funcionalidad del producto, su usabilidad y la interfaz grafica.

Para la realización de pruebas automatizadas se hace uso de una herramienta de automatización, la cual se encargara de la ejecución de las pruebas, el establecimiento de las condiciones de la prueba y la comparación de los resultados obtenidos.

Hoy en día el mercado ofrece una gran variedad de herramientas de este tipo, tanto comerciales como libres, las cuales son denominadas Computer Aided Software Testing (CAST).

Las herramientas elegidas para esta comparación son WAPT y Jakarta Jmeter, la primera de estas fue seleccionada en base a una recomendación, a la cual siguió una investigación para poder observar las características que ofrece y determinar su viabilidad, la segunda se eligió en base a que se necesitaban corroborar los datos de salida que ofrecería la primera.

II.6.1. WAPT (Web Application Testing)

Herramienta para realización pruebas de carga/estrés de una manera fácil, capaz de encontrar cuellos de botella en el rendimiento de una aplicación web, de acuerdo a un conjunto de configuraciones que pueden variar. Permite la interacción de distintos tipos de usuarios dentro de una misma prueba, lo cual es muy útil en el caso que la aplicación tenga diferentes niveles de acceso (permisos) de acuerdo al usuario que la utilice, finalmente entrega un reporte detallado sobre las pruebas realizadas (**SoftLogica**).

II.6.1.1. Características de WAPT

WAPT provee una simulación precisa de carga a un servidor web creada por usuarios reales. Cada usuario virtual que participa en las pruebas es independiente de los demás. Podrá tener sus propias cookies, datos y demás parámetros. Retrasos aleatorios entre páginas, así como la velocidad de conexión, lo que permitirá crear una carga de trabajo realista contra el servidor. WAPT provee la sustitución de la cabecera proxy para emular la actividad de múltiples usuarios que trabajan en diferentes equipos.

Esta herramienta tiene excelentes habilidades para generar datos en tiempo real. Los valores de los parámetros requeridos y URLs se pueden calcular de varias formas. Inclusive pueden ser determinadas por la respuesta del servidor a una petición previa.

Comparativo de herramientas para aplicar pruebas de estrés y desempeño para aplicaciones Web

WAPT soporta todas las características de seguridad referentes a https/ssl. Se podrán grabar y reproducir las peticiones a contenido seguro. También soporta pruebas a sitios protegidos con sistemas de autorización básicos.

Se pueden definir como cambiara el número de usuarios virtuales y otros parámetros durante la prueba. WAPT se puede configurar para la realización de varias iteraciones de una secuencia de pruebas. Se puede especificar la duración de la prueba, y el número de hits a realizar. Se puede limitar el nivel de carga a un número definido de hits por segundo.

Los resultados de las pruebas pueden ser representados por medio de graficas descriptivas y reportes, que de una manera detallada entregan la información sobre el desempeño del sitio web durante las condiciones de carga especificadas. Los reportes se pueden guardar en un formato compatible con Microsoft Excel.

WAPT creara un registro completo de las actividades realizadas por los usuarios virtuales, incluyendo todas las peticiones y respuestas realizadas, se puede configurar par que el registro solo contenga las respuestas de error.

Uno se puede familiarizar en cuestión de minutos con las características básicas de WAPT. La creación de pruebas es muy sencilla y se basa en registrar las acciones mientras el usuario navega el sitio web en Internet Explorer. Un vez terminada la grabación se puede editar la secuencia a las necesidades de la prueba.

Otras características de WAPT son:

- Soporte de todo tipo de servidores proxy: HTTP(S), SOCKS4(5).
- Las pruebas pueden ser calendarizadas para correr en un tiempo específico.
- Capacidad de agregar cadenas personalizadas a las cabeceras http.
- Habilidad para probar múltiples servidores al mismo tiempo.
- Soporte de redirección, incluso a otro servidor.

(Vail, 2005)

II.6.2. Jakarta Jmeter

Aplicación 100% Java, de código libre, diseñada para realizar pruebas al comportamiento y medir el performance de una aplicación web, es capaz de simular grandes cargas al servidor y su funcionalidad es expandible por medio de plugins (**Apache Foundation**).

II.6.2.1. Características de Jmeter

Algunas de las características que incluye Jmeter son:

- Capacidad de hacer pruebas de carga y desempeño en diferentes tipos de servidores, como Web(HTTP, HTTPS), SOAP, Bases de Datos JDBC, LDAP, JMS, Servidores de Correo (POP3 e IMAP).
- Portabilidad.
- Un marco de trabajo multi hilos que permite el sampleo de muchos hilos de manera simultánea de diferentes funciones realizadas por grupos de hilos separados.
- Una Interfaz Grafica diseñada cuidadosamente que permite una operación rápida y sincronización más precisa.
- Almacenamiento y Análisis/reproducción de los resultados sin contar con una conexión.
- Altamente extensible:
 - Muchas estadistas de prueba pueden ser elegidas para su sincronización.
 - Plugins de análisis de datos y visualización permiten extensibilidad y personalización.
 - Las funciones pueden ser usadas para proveer entradas dinámicas a una prueba, o proveer manipulación de datos.

(Apache Foundation).

III. Análisis Comparativo

III.1. Análisis Comparativo de Herramientas

Las aplicaciones web de hoy en día no solo necesitan ofrecer funcionalidad y operatividad, también necesitan tener niveles óptimos de calidad del servicio (QoS), los cuales se relacionan directamente al tiempo de respuesta y la disponibilidad. Gracias a las pruebas de rendimiento es posible encontrar los fallos de diseño arquitectónico, facilitando la corrección de estos y de esta manera mejorar las características y el rendimiento de la aplicación (**Navarro 2010**).

En este documento se compararan 2 herramientas de software que están diseñadas para la realización de pruebas de rendimiento en aplicaciones web, Esta comparativa servirá para estudiar factores como la usabilidad, interfaz, y datos de salida que ofrece cada una de las herramientas.

La comparación de estas herramientas se realizo en un caso de estudio, en el cual se midió el performance de una aplicación web, así como se obtuvo el número de usuarios simultáneos que la aplicación soportara, de acuerdo al ambiente en el cual se alojo dicha aplicación.

III.1.1. Características a Medir en el Análisis Comparativo

Para la comparación de las herramientas se tomaran en cuenta los once factores que a continuación se listan:

- Fabricante: Nombre de la empresa que desarrollo la herramienta, así como el sitio web en el cual se puede obtener.
- Tipo de Licencia: La licencia delimita muchas de las capacidades de la herramienta.
- Plataforma: Sistemas Operativos en los que puede funcionar la herramienta.
- Manejo de Perfiles de Usuario: Por lo general este tipo de herramientas permiten la utilización de varios perfiles de usuario, que delimitaran las acciones que pueden realizar en las pruebas.
- Proxy Http: Esta funcionalidad hace conexión con el navegador web, con el fin de grabar los scripts de pruebas.

- Escalabilidad de usuarios: El número de usuarios que la herramienta maneja se puede incrementar de acuerdo a los recursos del equipo en la que se instale.
- Distribución de carga: La herramienta es capaz de realizar pruebas desde múltiples equipos, asignando un porcentaje de la carga total a cada uno.
- Interfaz de Usuario: Como su nombre lo indica, es el medio por el cual el usuario podrá interactuar con la herramienta.
- Facilidad de Uso: En este punto se tomara en cuenta la facilidad con la cual se puede realizar la grabación de un script de pruebas.
- Resultados en tiempo real: La capacidad de ver el estado de las pruebas
- Reportes de Salida: Tipo de reportes que la herramienta ofrece al terminar sus pruebas.

En cuanto a las pruebas realizadas al sistema se monitorearan las siguientes variables:

- No. De Usuarios: Representa el número de usuarios virtuales que interactúan con el sistema (**Softlogika**).
- Tiempo de Respuesta: tiempo en el cual el servidor carga el contenido leible (sin incluir imágenes y estilos) de una página solicitada (**Softlogika**).
- % de Errores HTTP: El porcentaje de errores http (del servidor) del total de hits (elementos cargados por un a solicitud) (**Softlogika**).

III.1.2. Realización de las Pruebas

Las pruebas de performance se realizaron sobre una aplicación Web en desarrollo, llamada “Sistema PREP”, su objetivo es el de poder registrar encuestas de salida y resultados obtenidos sobre un ejercicio electoral en México, además de que esta información pueda ser consultada por los usuarios que tengan los permisos suficientes para ello.

Los objetivos de la prueba son los siguientes:

- Obtener el número de usuarios concurrentes que el sistema soporta en el ambiente de pruebas actual.
- Descubrir si el sistema es capaz de atender 3000 peticiones en un periodo de 15 minutos.

Primero que nada se debió establecer las acciones que el usuario deberá hacer en la prueba, por lo cual se establecieron 2 tipos de usuarios para la prueba, el primero, que tendrá la tarea de ingresar en el sistema, depositar los resultados de la elección de la casilla a la cual se le asigna, y finalmente terminar su sesión, el segundo, un administrador, el cual una vez que ingrese al sistema, solo navegará a través de las diferentes páginas a las cuales tiene permisos de acceso, finalmente también deberá cerrar su sesión.

Las pruebas en WAPT utilizaron ambos perfiles de usuario, con una carga de cero a 900 usuarios, 450 de cada perfil, en un periodo de 15 minutos. A su vez, en JMeter solo se utilizó el perfil que se encarga del depósito de los resultados, con una carga de cero a 900 usuarios en el mismo periodo de 15 minutos.

III.1.2.1. Resultados del Análisis del Sistema PREP

Durante las primeras pruebas, solo se contaba con la herramienta JMeter, pero se logró detectar, que el servidor virtual en el cual se alojaba la aplicación web no estaba configurado para un servicio dedicado, ya que compartía sus recursos, con otros equipos virtuales, así que el servidor trabajaba con los recursos que tenía disponibles. Gracias a esta situación, el desempeño de la aplicación variaba dependiendo del uso que se le daban a todos los equipos virtuales alojados junto al servidor.

Tabla 2 – Primeras Pruebas con JMeter

Usuarios	Muestras	TR Media	TR Min	TR Max	% Error
500	4000	83ms	34ms	3536ms	0
1000	8000	90ms	33ms	3501ms	0
2000	16000	1850ms	34ms	33444ms	0.01

Una vez informado el problema, se hicieron los cambios adecuados, para que el servidor tuviese recursos fijos, de esta forma el desempeño no variaría, no importando si se estuviesen usando otros equipos virtuales instalados en el mismo equipo físico.

Al tener el servidor “estable”, y además ya se contaba con la licencia de la herramienta WAPT, se procedió a la realización de las pruebas una vez más, ahora con ambas herramientas, con lo que se pudo determinar que el servidor

Comparativo de herramientas para aplicar pruebas de estrés y desempeño para aplicaciones Web

podría atender 2500 peticiones en un periodo de 15 minutos, sin mostrar error alguno.

Tabla 3 – Pruebas Finales con JMeter

Usuarios	Muestras	TR Media	TR Min	TR Max	% Error
2500	25000	233ms	25ms	4957ms	0
2750	27500	7340ms	29ms	328024ms	0.00450909
3000	30000	10527ms	29ms	617997ms	0.06793333

Los resultados obtenidos durante dos de las pruebas que se realizaron con WAPT, demuestran una mejoría en los tiempos de respuesta y procesado de las peticiones, cambio obtenido por la configuración de los tiempos de timeout del servidor (se redujo a 15 segundos), además de algunos cambios realizados de acuerdo a recomendaciones para optimizar la carga de la aplicación, finalmente se instaló en el servidor el paquete Apache PHP Cache, el cual sirve para optimizar el desempeño de aplicaciones PHP.

Figura 3 – Primer Prueba, Gráfica de Desempeño

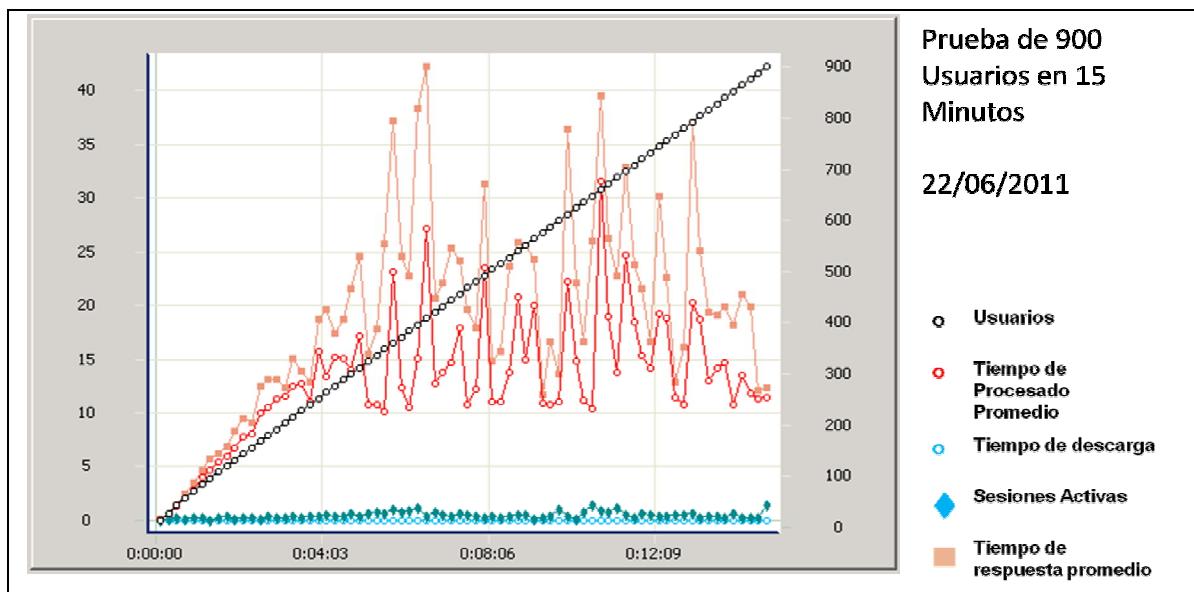


Figura 4 – Ultima Prueba – Grafica de Desempeño

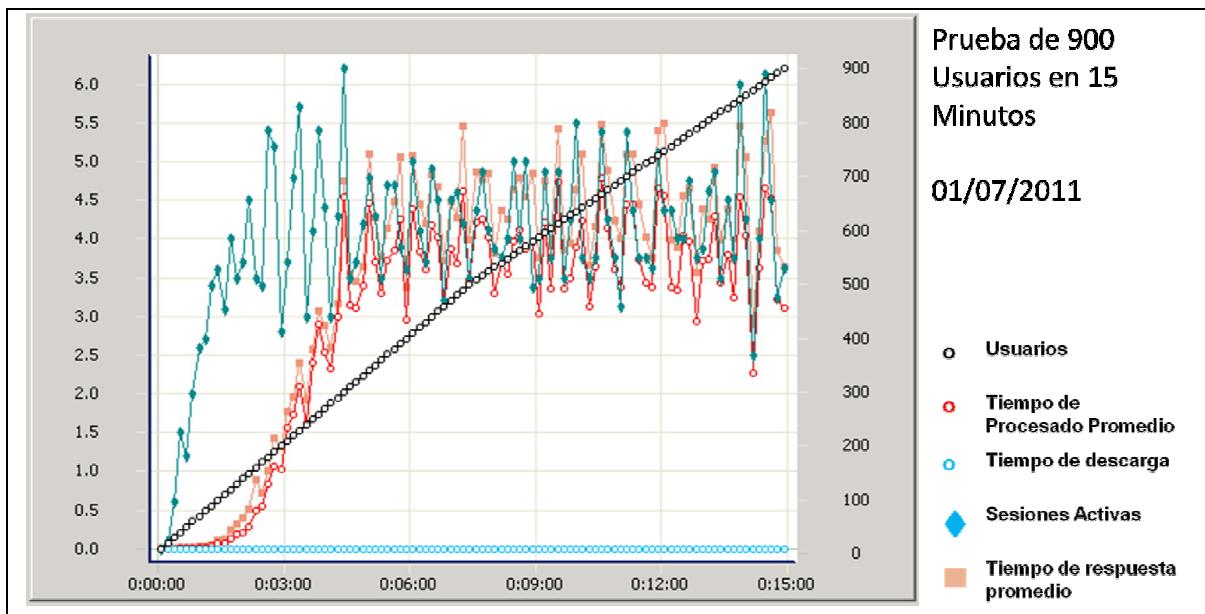


Figura 5 – Primer Prueba, Grafica de Errores

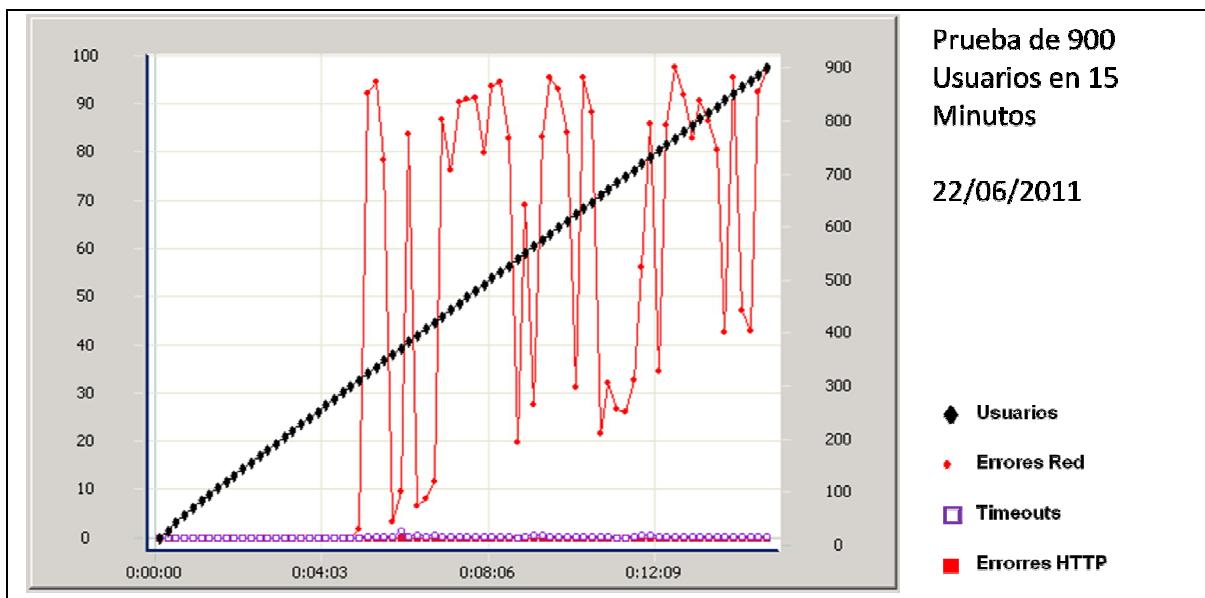
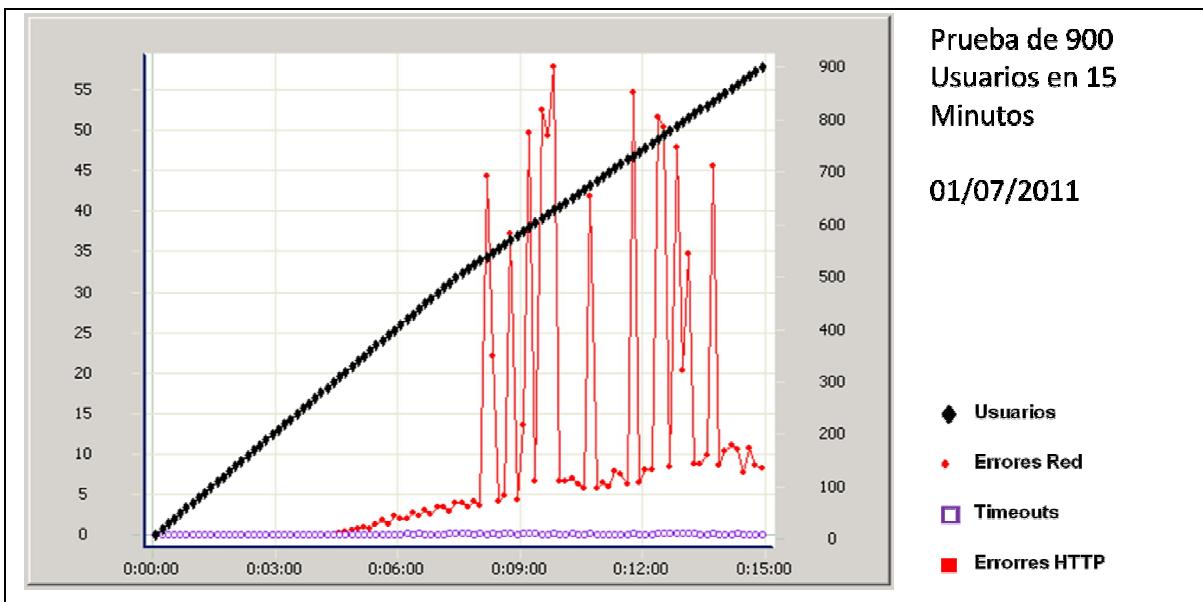
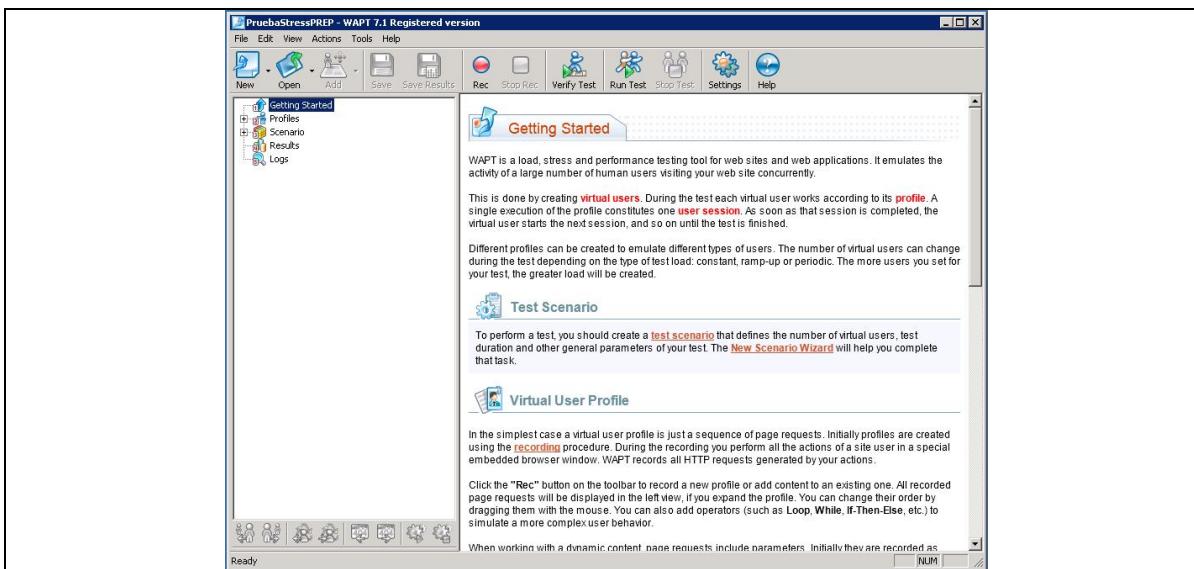


Figura 6 – Ultima Prueba, Grafica de Errores



III.1.2.2. Prueba con WAPT

Figura 7 – Interfaz Grafica WAPT

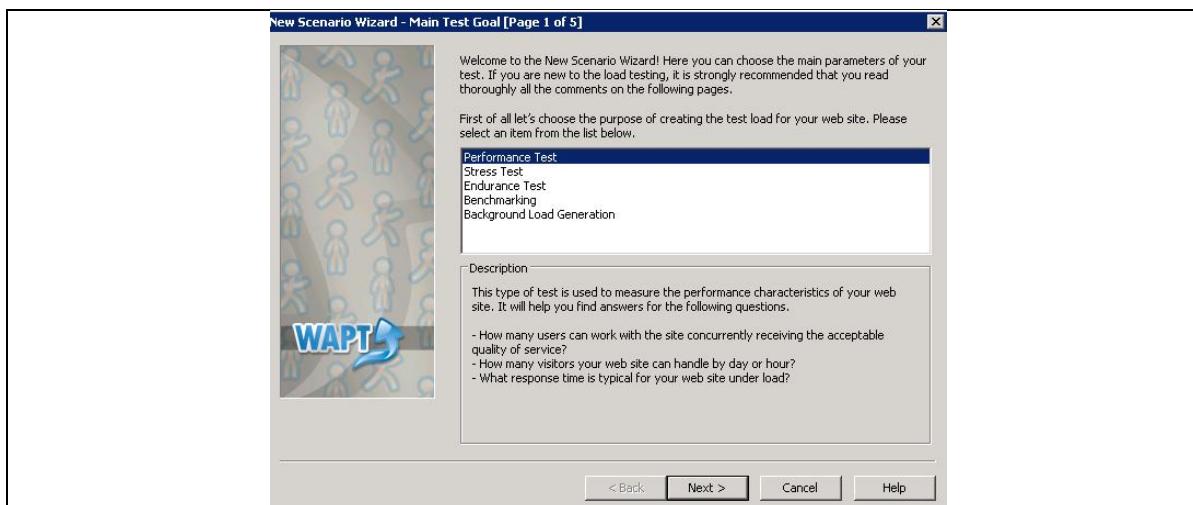


III.1.2.2.1. Generar un Escenario de Prueba

En primer lugar, veremos la interfaz con la que cuenta esta herramienta, muchas de las acciones básicas para la realización de pruebas se encuentran a la vista, como los botones para comenzar a grabar y parar la grabación de un script, así como empezar y parar las pruebas.

Para poder establecer un escenario de pruebas, WAPT cuenta con un asistente, que paso a paso, ayuda la configuración de la misma, facilitando la tarea de creación de las pruebas.

Figura 8 – Asistente de Configuración para un Escenario de Pruebas



En la primera de las opciones del asistente elegiremos el tipo de pruebas que WAPT realizara, para lo cual seleccionaremos “stress test”, una vez que avancemos, en el siguiente paso nos pedirá el número de usuarios virtuales iniciales y finales de la prueba, así como el tiempo entre el incremento de número de usuarios, y cuantos usuarios se incrementaran.

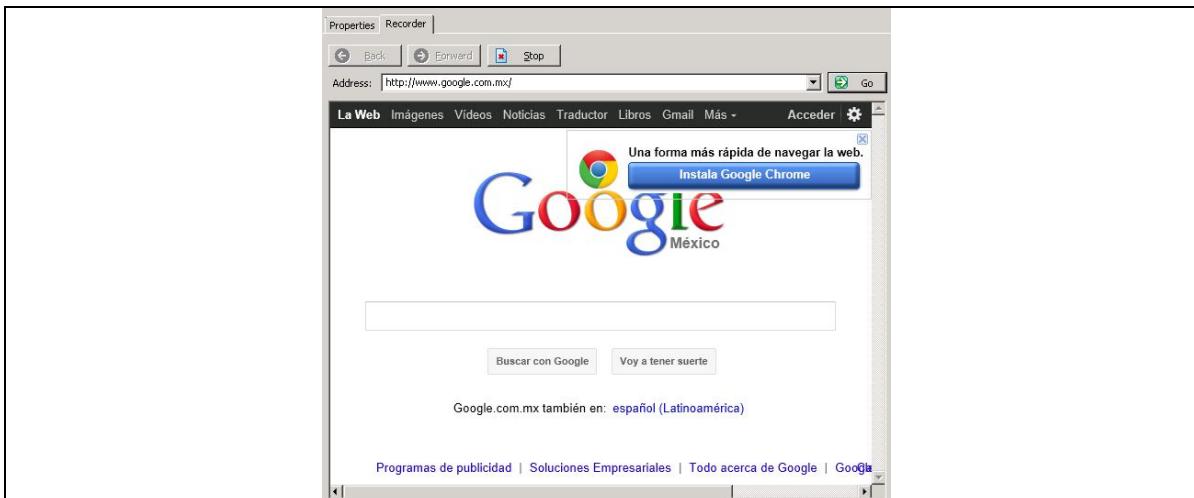
En la tercer pantalla de opciones del asistente nos pedirá el tiempo que durara la prueba, o en su caso, el numero de sesiones que se deberán cumplir para que la prueba termine, en la siguiente pantalla pedirá el numero de columnas que contendrá el reporte a generar, esto sirve para mostrar las estadísticas en secciones de tiempo, mientras más columnas, estas secciones serán más cortas.

Finalmente el asistente te dará la opción de comenzar a grabar los perfiles de usuario, en caso contrario, uno puede realizar la grabación de los perfiles en el momento que uno desee.

III.1.2.2.2. Grabación de un Perfil de Usuario

Para la grabación de un nuevo perfil de pruebas, la herramienta cuenta con un navegador web embebido (Internet Explorer) dentro de la misma aplicación, con lo cual no es necesaria la utilización de herramientas extra para la grabación de los scripts de prueba que utilizaran los perfiles de usuario creados.

Figura 9 – Navegador WAPT



Una vez grabados todos los scripts de prueba, es necesario validarlos, WAPT se encarga de hacer esta acción, de tal manera que recorre y ejecuta las acciones previamente grabadas en el(los) script(s) de prueba de el(los) perfil(es) seleccionado(s).

III.1.2.2.3. Método de Prueba de WAPT

Al iniciar la prueba, WAPT comenzara con los usuarios virtuales indicados durante la configuración del escenario, e irá incrementando el número de usuarios virtuales en los intervalos de tiempo marcados, durante la prueba, si un usuario termina de ejecutar el script, comenzara de nuevo, de tal forma que todos los usuarios trabajen de manera concurrente durante la prueba.

III.1.2.2.4. Resultados en Tiempo Real

WAPT ofrece una serie de datos en tiempo de ejecución de la prueba, así como graficas sobre el desempeño, ancho de banda y errores de la aplicación, estas graficas son personalizables, ya que se puede elegir que datos queremos mostrar. Algunos de los datos que WAPT muestra en sus graficas son las siguientes:

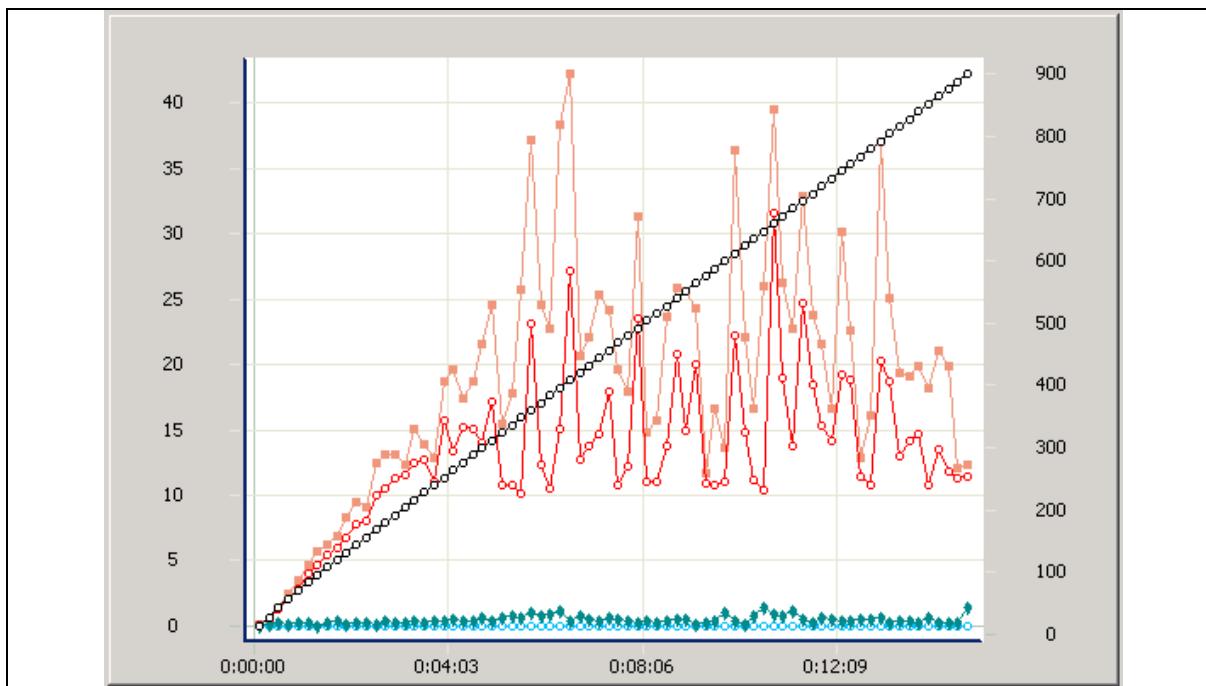
Tabla 4 – Variables para Monitoreo de Pruebas de WAPT

Termino	Definición
Tiempo de respuesta promedio	Tiempo promedio (entre todos los usuarios) que tarda la aplicación en responder las peticiones realizadas sobre una página, sin contar los elementos que contiene la misma
Tiempo de respuesta promedio (incluyendo elementos de la pagina)	Tiempo promedio (entre todos los usuarios) que tarda la aplicación en responder las peticiones realizadas sobre una página, contando los elementos que la componen
Tiempo de procesado promedio	Muestra el valor promedio (entre todos los usuarios) de tiempo de proceso de una petición, sin contar los elementos de la pagina
Tiempo de bajada promedio	Tiempo promedio (entre todos los usuarios) que tarda en bajar una página, sin contar sus elementos, WAPT mide el tiempo que tarda en cargar el contenido leible de la pagina web, como su estructura (marcos, tablas) y el texto en html
Sesiones por segundo	El numero de sesiones ejecutadas en un momento específico de la prueba
Paginas por segundo	El número de páginas que se ejecutan en un momento específico de la prueba.
Hits por segundo	El número de hits ejecutados en un momento específico de la prueba, un hit es catalogado como una llamada a cualquier otro elemento dentro de una página.

Usuarios Activos	El número de usuarios virtuales participando en el test.
Kilobits enviados	El número de Kb que son enviados por segundo hacia el servidor
Kilobits recibidos	El número de Kb que son recibidos por segundo hacia el servidor
Kilobits enviados por usuario	El número de Kb que son enviados por segundo hacia el servidor por un solo usuario virtual
Kilobits recibidos por usuario	El número de Kb que son recibidos por segundo hacia el servidor por un solo usuario virtual
% de Errores http	El porcentaje de respuestas con error HTTP (errores del servidor) en base al número total de hits
% de Errores de red	El porcentaje de respuestas con error de red en base al número total de hits.
% de Timeouts	El porcentaje de respuestas timeout en base al numero total de hits, un timeout se logra cuando el servidor agota su tiempo para poder atender una petición sin éxito.
% de errores totales	El porcentaje de respuestas de error (cualquiera de los anteriores) en base al número total de hits.

(SoftLogica).

Figura 10 – WAPT, Grafica de Desempeño



III.1.2.2.5. Reportes/Gráficos/Registros de Salida

La herramienta, una vez terminada la prueba generara y desplegará un reporte en formato html, además, se puede configurar para que este mismo reporte sea mandado vía correo electrónico a una dirección otorgada.

Este reporte está distribuido de la siguiente forma:

- Parámetros de ejecución de la prueba
- Resumen
- Informe de Usuarios Activos
- Informe de Sesiones Exitosas/Fallidas
- Informe de Paginas Exitosas/Fallidas
- Informe de Hits Exitosos/Fallidos
- Informe de sesiones exitosas por segundo
- Informe de paginas exitosas por segundo
- Informe de Hits exitosos por segundo
- Tiempos de respuesta incluyendo elementos de pagina
- Informe de Kbytes Enviados
- Informe de Kbytes recibidos

Comparativo de herramientas para aplicar pruebas de estrés y desempeño para aplicaciones Web

- Informe de Velocidad de Envió (Kbits/segundo)
- Informe de Velocidad de Recepción (Kbits/segundo)
- Informe de Velocidad de Envió por usuario virtual (Kbits/segundo)
- Informe de Velocidad de Recepción por usuario virtual (Kbits/segundo)
- Informe sobre paginas fallidas
- Informe sobre Hits fallidos
- Informe sobre códigos de respuesta obtenidos
- % de errores HTTP
- % de errores de red
- % de timeouts
- % de errores totales
- Informe sobre utilización de recursos por parte de WAPT
- Informe de URLs visitadas por los perfiles incluidos en la prueba
- Grafica de Errores de Desempeño
- Grafica de Ancho de Banda
- Grafica de Tiempos de Respuesta

Figura 11 – WAPT, Reporte de Prueba

The screenshot shows the WAPT 7.1 - Report interface. On the left, there's a sidebar with links for Reports (Test execution parameters, Summary, Performance Data, Response Time, Bandwidth Usage, Errors, WAPT utilization, URLs), Graphs (Overall performance, Errors, Average bandwidth, Average response time), and a summary table.

Test execution parameters:

- Test status: finished
- Test started at: 22/06/2011 01:01:05 p.m.
- Test finished at: 22/06/2011 01:16:05 p.m.
- Scenario name: PruebaStressPREP.wps
- Test run comment:
- Test executed by: Administrador
- Test executed on: SLK-WIN-TESTER
- Test duration: 0:15:00
- Virtual users: 1 - 900

Summary

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
RegVotos	206	42403	1983	42403	16887	42403	7300	63675	21.3(41.9)
Admin	191	70417	2194	70417	13889	70417	5778	52613	17.2(27.6)

Number of active users

Profile	0:00:00-0:01:30	0:01:30-0:03:00	0:03:00-0:04:30	0:04:30-0:06:00	0:06:00-0:07:30	0:07:30-0:09:00	0:09:00-0:10:30	0:10:30-0:12:00	0:12:00-0:13:30	0:13:30-0:15:00
RegVotos	45	90	135	180	225	270	315	360	405	450
Admin	45	90	135	180	225	270	315	360	405	450
Total	90	180	270	360	450	540	630	720	810	900

Successful sessions (Failed sessions)

Profile	0:00:00-0:01:30	0:01:30-0:03:00	0:03:00-0:04:30	0:04:30-0:06:00	0:06:00-0:07:30	0:07:30-0:09:00	0:09:00-0:10:30	0:10:30-0:12:00	0:12:00-0:13:30	0:13:30-0:15:00	Total
RegVotos	5(0)	7(0)	11(0)	23(2521)	35(1576)	16(7429)	20(6103)	52(3239)	20(12863)	17(9582)	206(42403)
Admin	8(0)	13(0)	16(0)	33(8018)	26(5642)	18(15651)	17(10117)	18(5964)	18(13057)	24(11968)	191(70417)
Total	13(0)	20(0)	27(0)	56(10539)	61(7218)	34(23080)	37(16220)	70(8293)	38(25920)	41(21550)	397(112820)

WAPT puede guardar los resultados en un formato nativo de la misma herramienta para poder consultar posteriormente.

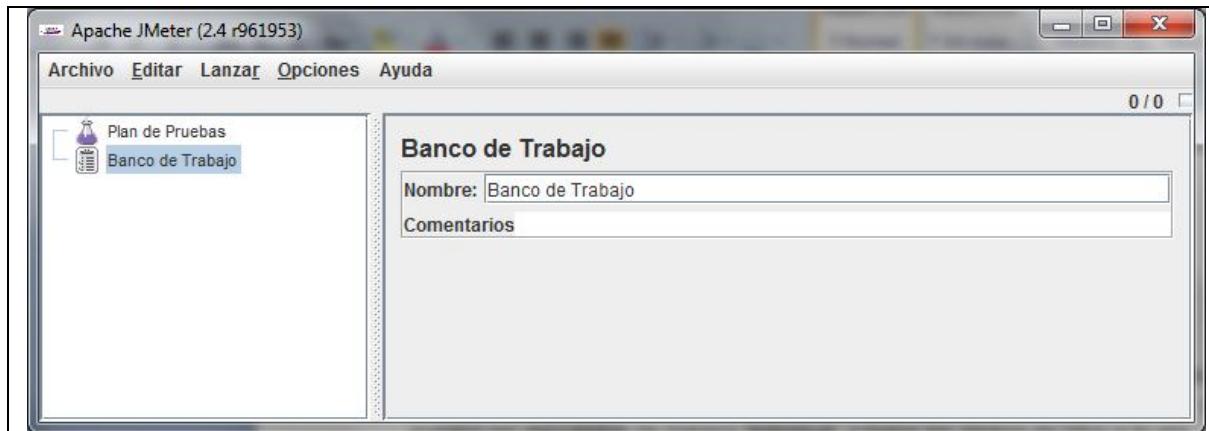
Además de esto, en la interfaz de la herramienta se pueden consultar los gráficos que fue generando la prueba, de tal manera que se pueden escalar para mostrar un mayor o menor detalle de los datos que acontecieron durante el tiempo que duro la prueba, permitiendo de la misma forma, la exportación de estas graficas en archivos de imagen.

Otra cosa que puede ser revisada una vez que se termina la prueba son los registros (logs), en los cuales se muestra de manera detallada, las acciones realizadas por cada uno de los usuarios virtuales, de esta manera podremos ver cuántas veces recorrió el script cada uno de los usuarios virtuales, que páginas pudo visitar con éxito, y en cuales falló. Estos registros pueden exportarse en archivos con extensión .log.

III.1.2.3. Pruebas con JMeter

Para el caso de JMeter, no es necesario establecer un escenario de pruebas, todos los elementos que se agreguen quedaran en un plan de pruebas, y pueden ser ejecutados de manera individual, o todos los grupos de hilos a la vez.

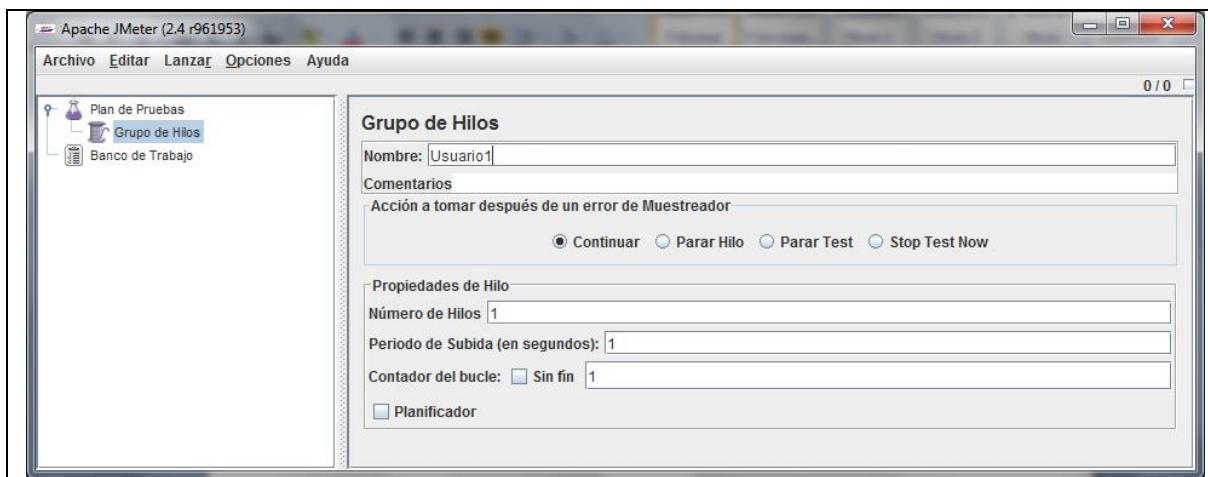
Figura 12 – Interfaz Grafica JMeter



III.1.2.3.1. Creación de un Grupo de hilos

Un grupo de hilos de JMeter es el equivalente a un perfil de usuario en WAPT, en el se grabara la secuencia de acciones que el usuario virtual deberá de realizar una vez iniciada la prueba. JMeter permite que se agreguen los grupos de hilos necesarios para la realización de una prueba.

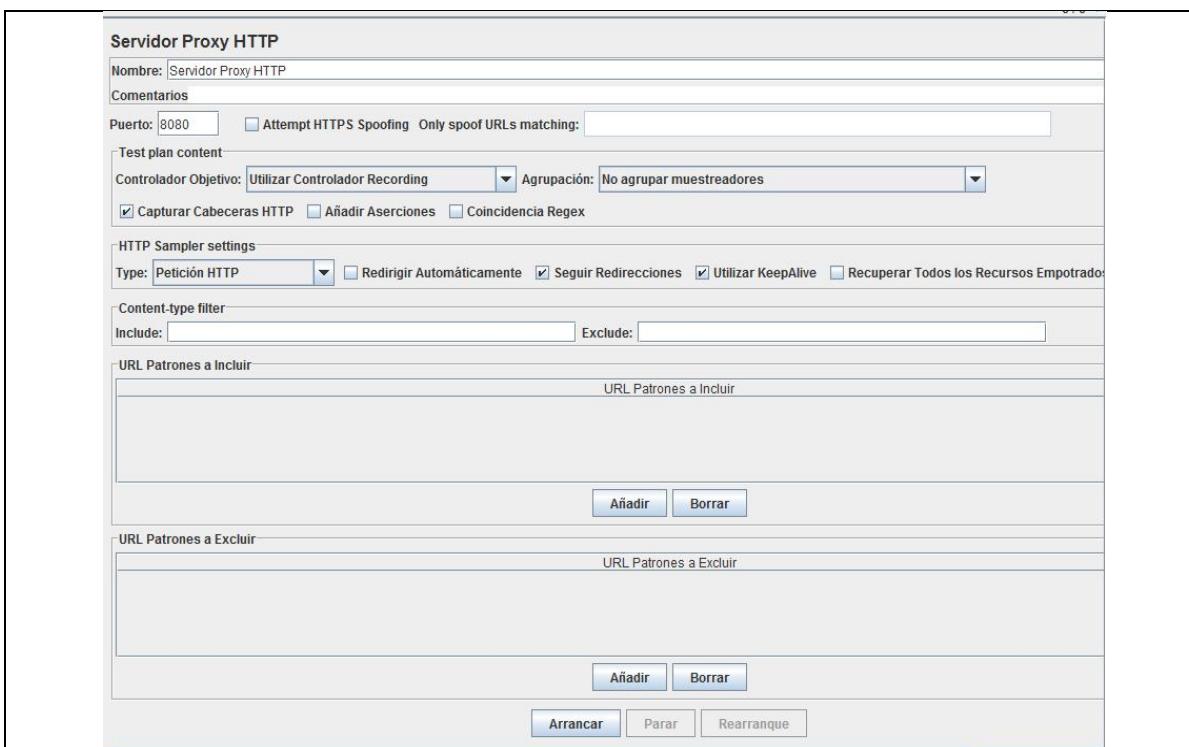
Figura 13 – Jmeter, Configuración de un Grupo de Hilos



III.1.2.3.2. Configuración de un Proxy

Una vez que se han creado uno o más grupos de hilos, se tendrá que hacer la grabación del script de pruebas para cada uno, para poder realizar esta acción, es necesario configurar previamente un proxy, el cual grabara las acciones de navegación que pasen a través de él en el grupo de hilos que se le indique, dicho proxy se tendrá que agregar dentro del banco de trabajo.

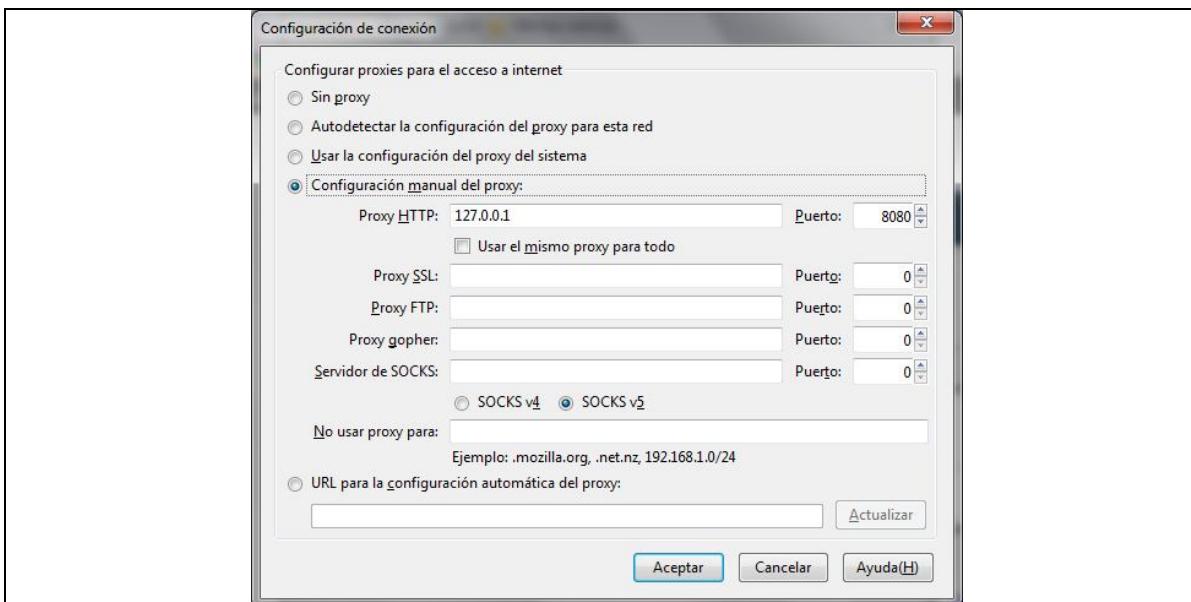
Figura 14 – Jmeter, Configuración de Servidor proxy



Dentro de la configuración, se necesita establecer por fuerza, cual es el puerto por el cual se va a comunicar este proxy, asi como, el grupo de hilos en el cual se grabaran las acciones, entre la configuración opcional se encuentran los patrones a incluir y excluir, en los cuales se pueden establecer los formatos que se desea o no, formen parte de las acciones, dentro de la prueba realizada con el sistema prep, se excluyeron llamadas a los objetos del tipo .jpeg, .jpg, .png, .gif, .css, .js y .ico.

Una vez establecida la configuración del proxy, es necesario configurar algún navegador (Internet Explorer, Firefox, Chrome), para que pueda navegar a través de este.

Figura 15 – Configuración de Proxy, Mozilla Firefox



Ahora, todas las peticiones que pasen a través del proxy, serán grabadas, en el grupo de hilos indicado dentro de la configuración del mismo. Al finalizar solo bastara con desactivar el proxy para que deje de grabar las peticiones realizadas en el navegador.

III.1.2.3.3. Elementos Listener

Estos son los elementos que se encargaran de vigilar el estado de las pruebas, además llevaran el registro de las respuestas que van apareciendo a través de la prueba, estos elementos puedes integrarse tanto al plan de pruebas, con la finalidad de evaluar el conjunto de grupos de hilos, o en un grupo de hilos, de tal forma que solo se encargaran de la recolección de datos del grupo al que pertenece.

Entre los elementos de este tipo que se utilizaron para las pruebas están:

Tabla 5 – Jmeter, Listeners

Grafico de Resultados	Área donde se desplegaran de manera visual algunos datos sobre los tiempos de respuesta de todas las muestras (peticiones)
Summary Report	Resumen general de la prueba
Árbol de Resultados	Es el equivalente a los registros (logs) de WAPT, en el que se hace un informe de las respuestas a cada una de las peticiones.

III.1.2.3.4. Método de Prueba de JMeter

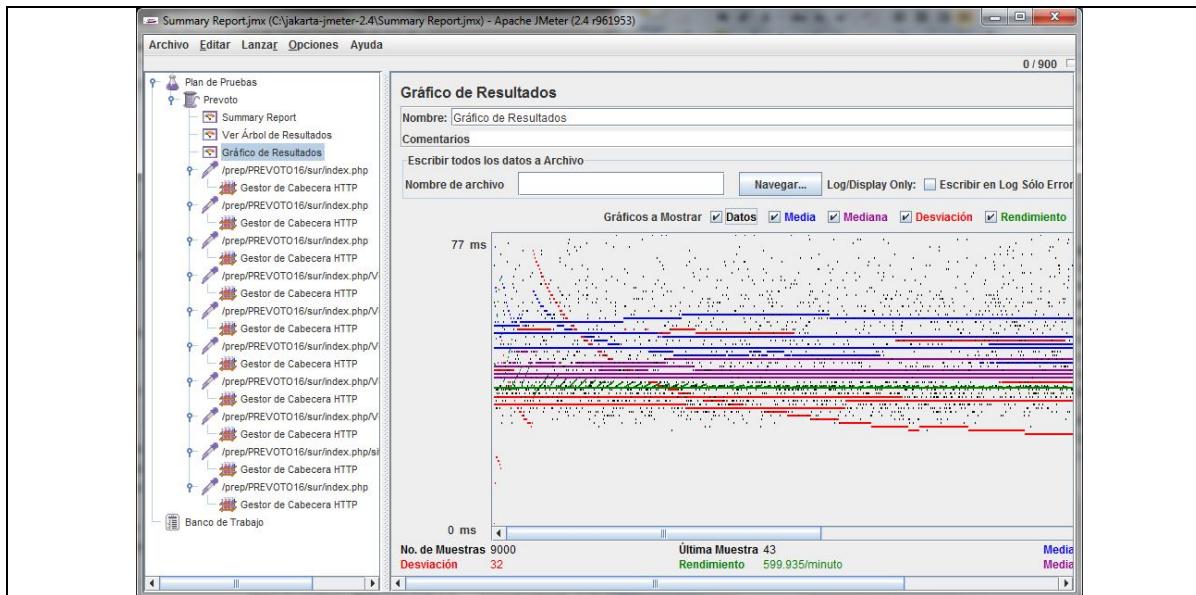
La forma en que se configuraron los usuarios virtuales dentro de JMeter logra que una vez que un usuario virtual termine la ejecución del script de pruebas, deje el sistema, así que cada usuario virtual solo intervendrá una vez en la prueba.

III.1.2.3.5. Resultados en Tiempo Real

Al igual que WAPT, JMeter tiene la capacidad de ir mostrando datos de la prueba en tiempo de ejecución, los datos que podrá mostrar dependerán de los listeners que se utilicen, en cuanto a las graficas, estas muestran los tiempos de respuesta para todas las muestras (peticiones) que pasen en la prueba, el inconveniente con estas graficas radica que esta utiliza un área fija, por lo cual si la prueba es algo extensa, si la grafica llega a su límite del eje horizontal, los nuevos resultados comenzaran a aparecer al principio de esta área, y si logra llegar una vez más al límite vertical, volverá a mostrar resultados al principio de este eje, de esta manera se mezclaran todos los datos, haciendo difícil la tarea de interpretar esta grafica.

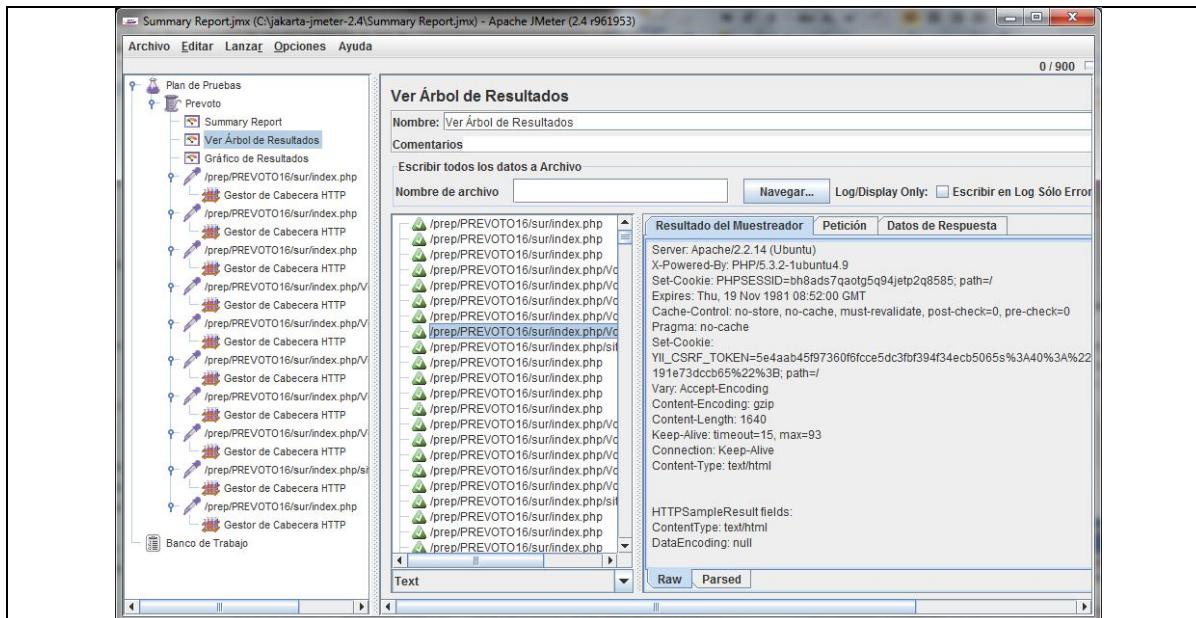
Comparativo de herramientas para aplicar pruebas de estrés y desempeño para aplicaciones Web

Figura 16 – Jmeter, Gráfica de Desempeño



A diferencia de WAPT, JMeter deja consultar el log de muestras en tiempo de ejecución de prueba, JMeter no organiza estas muestras de acuerdo a que usuario virtual la invoca, sino que solamente la lista en el orden que cada una de las peticiones fueron realizadas.

Figura 17 – Jmeter, Log de Resultados



III.1.2.3.6. Reportes/Graficas/Registros de Salida

JMeter no genera reporte alguno, las únicas formas de extraer la información, es guardando pantallas, en el caso de las graficas, para los listeners, la herramienta tiene una opción para guardar registros en un archivo de valores separados por coma (*.CSV), en el cual se volcara la información correspondiente al listener que hayamos configurado.

III.1.3. Resumen de Análisis Comparativo

Tabla 6 – Comparativo de Herramientas

Factor	WAPT	Jakarta Jmeter
Fabricante	Grupo Apache (http://jakarta.apache.org/jmeter)	SoftLogica (http://www.loadtestingtool.com)
Licencia	Open Source - Freeware	450 y 900 usdls
Plataforma	Windows, Linux, Solaris	Windows
Perfiles de Usuario	✓	✓
Proxy HTTP	✓	✓
Escalabilidad	✓	✓
Distribución de Carga	✗	Solo versión PRO
GUI	✓	✓
Facilidad de Uso	✗	✓
Resultados en tiempo real	✓	✓
Reportes de Salida	✗	✓

III.1.4. Resultados del Análisis Comparativo

Como podemos ver, en muchos de los factores de comparación, ambas herramientas cumplen con el requisito, pero es necesario aclarar los puntos en los cuales una herramienta es mejor que la otra.

Jakarta Jmeter está desarrollado 100% en código java, por lo cual se puede utilizar en plataformas como Windows, Linux y Solaris, mientras que WAPT solamente funciona para el Sistema Operativo Linux.

En cuanto a la distribución de cargas de las pruebas, esta funcionalidad solo está disponible en la versión PRO de la herramienta WAPT, con la cual en una maquina central se establecerá la configuración de las pruebas, en cuya configuración se establecerá el porcentaje de carga de las peticiones que realizará cada uno de los equipos que estarán a cargo de la realización de las pruebas.

El proceso de grabación de un script de pruebas difiere de una herramienta a otra, WAPT cuenta con un explorador (Internet Explorer) embebido dentro de la herramienta, con lo cual solo es necesario oprimir un botón para comenzar a grabar los pasos de los que constará la prueba para un perfil de usuario, mientras que en Jmeter es necesaria la configuración e inicialización de un proxy http, ya que no cuenta con un explorador dentro de su interfaz, después se tendrá que configurar un explorador de internet para que navegue a través de este proxy, así que cada movimiento que se realice dentro del explorador, será tomado en cuenta para el script de pruebas.

Además de esto, en JMeter es necesaria ordenar la limpieza de los elementos del tipo oyente (listeners), ya que si no se realiza la limpieza de estos, si se realiza una prueba, los resultados de una prueba anterior se mezclarán con los de la nueva.

Finalmente, en cuanto a los reportes de salida, WAPT por default mostrara un reporte en formato HTML, en el cual mostrara el resumen de las pruebas, además cuenta con graficas en cuanto al desempeño de la aplicación y el servidor, el ancho de banda, y los errores ocurridos durante la prueba, dando la opción de que este reporte pueda ser enviado a un destinatario de correo electrónico.

III.1.5. Resultados de las Pruebas

En resumen, se lograron los siguientes resultados al finalizar las pruebas con el sistema PREP:

- Se logró bajar el tiempo máximo de respuesta de >30 a 5 segundos, que de acuerdo a Alejandro García, líder de la empresa que desarrolló el sistema PREP, es un muy buen tiempo de respuesta, ya que la aplicación no será abierta a todo público y funcionará solo para quien tenga comprado el servicio.
- El número de peticiones soportadas por la aplicación en un periodo de 15 minutos es de 2500.
- El sistema soporta hasta 300 usuarios concurrentes.

IV. Conclusiones

Las pruebas de software son una forma para medir y asegurar los niveles de calidad de un producto de software, permiten la detección de errores que surgen durante el proceso de desarrollo del producto.

Hoy en día existe gran cantidad de productos de software, entre los cuales se incluyen las aplicaciones web, las empresas/organizaciones desarrolladoras se esmeran por lograr altos índices de calidad en sus productos, con lo cual aseguran la competitividad de los mismos, y la satisfacción del usuario final.

Asimismo se está subestimando el valor que conlleva el desempeño del sistema en operación, se escatima más tiempo para planear los diseños de funcionalidad, y muy pocas veces se hace un esfuerzo para satisfacer ciertos niveles de desempeño. Y solo se le da su importancia una vez que el cliente reporta algún error.

Las deben de ser llevadas de la mano con una planeación de capacidad, ya que primero que nada ayudara a corroborar o refutar los índices que en la planeación surjan, y de esta manera verificar o realizar cambios al modelo. Y cumplir con los requerimientos de desempeño que el cliente pide.

En mi experiencia, fue muy benéfico el uso de ambas herramientas para el análisis del sistema, ya que los datos de salida ofrecidos, se corroboraban, y en algunos de los casos se complementan.

En cuanto a la cobertura en un nivel gráfico, WAPT sobrepasa a JMeter (sin modificaciones), ya que gracias a los gráficos que ofrece, se tiene una mejor idea de cómo se comporta el sistema.

Para finalizar puedo decir que el desempeño del sistema fue evaluado exitosamente, se lograron obtener los datos deseados por el cliente.

IV.1. Trabajo Futuro

Evaluación del desempeño del sistema, implementando múltiples servidores y distribuyendo la atención de peticiones entre estos (distribución de carga).

Buscar una configuración de JMeter que logre ofrecer los mismos resultados, por lo menos gráficamente, de WAPT.

Medir el Retorno de la Inversión del Sistema PREP.

Ayudar con la implementación de algún modelo/proceso de planeación de la capacidad a la empresa, ya que actualmente no cuentan con ninguno.

V. Bibliografía y Referencias

1. Almeida Virgilio A.F., Menascé Daniel A. (Agosto 2002) Capacity planning: An Essential Tool for Managing Web Services. Documento Electrónico del sitio IEEE Xplore, ultima consulta 23 de Agosto de 2011.
http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=1046642&abstractAccess=no&userType=inst
2. Andersin Jari (Octubre 2004) TPI – a model for test Process Improvement. Documento Electrónico del sitio Tietojenkäsittelytieteen laitos, consultado el 8 de Julio de 2011. <http://www.cs.helsinki.fi/u/paakki/Andersin.pdf>
3. Apache Software Foundation (S.F) Jmeter – Apache Jmeter. Ultima consulta el 12 de Julio de 2011. <http://jakarta.apache.org/jmeter>
4. Butler Kelley L. (Julio 1995) The Economic Benefits of Software Process Improvement. Documento electrónico de la pagina del personal del Stevens Institute of Technology, ultima consulta 18 de agosto de 2011.
http://personal.stevens.edu/~aschron/MIS-620_LockStep/Articles/Butler_The_Economic_Benefits_of_SPI.pdf
5. Daneva, Maya (1995). Software benchmark design and use. Extracto del Libro Re-Engineering the Enterprise (pg. 20-41) de Browne y David O'Sullivan (Oct 31, 1995), Documento Electrónico del sitio Google books. Ultima consulta 16 de Julio de 2011.
<http://books.google.com.mx/books?hl=es&lr=&id=oUVZ9uCbbggC&oi=fnd&pg=PA20&dq=softwarebenchmark+&ots=31a7Op8wjf&sig=y4HXgEweh7Hhxr7X-s-FCAvc0Aw#v=onepage&q&f=false>
6. Navarro, Juan Oliver (2010). Estado del Arte de Métodos, Tipos de Testing y Herramientas para Aplicar Pruebas de Rendimiento. Documento Electrónico del sitio web scribd, ultima consulta 13 de Julio de 2011.
<http://es.scribd.com/doc/37584872/ESTADO-DEL-ARTE-DE-METODOS-TIPOS-DE-TESTING-Y-HERRAMIENTAS-PARA-APLICAR-PRUEBAS-DE-RENDIMIENTO>.
7. Palmer, Jonathan W. (Junio 2002) Information Systems Research (pag. 151-167) Documento electrónico, consultado el 5 de Julio.
8. Real Academia Española (S.F) Sitio de la Real Academia Española, consultada el 12 de julio de 2011. <http://www.rae.es/rae.html>
9. Return of Investment (ROI) – Investopedia (SF) Sitio Investopedia, ultima consulta 18 de agosto de 2011.
<http://www.investopedia.com/terms/r/returnoninvestment.asp#axzz1VOTJaDrC>

Comparativo de herramientas para aplicar pruebas de estrés y desempeño para aplicaciones Web

10. Rich Joe, Hill Jon (2010) How to Do Capacity Planning. Documento Electrónico de la página TeamQuest. Ultima consulta 20 de Agosto de 2011. <http://www.teamquest.com/pdfs/whitepaper/tqwp23.pdf>
11. SoftLogica (S.F.) WAPT – Web Application Load, Stress and Performance Test. Ultima consulta el 13 de Julio de 2011. <http://www.loadtestingtool.com>
12. Stress Testing - Wikipedia (S.F.) Sitio Web Wikipedia. Ultima consulta el día 11 de Agosto de 2011. http://en.wikipedia.org/wiki/Stress_testing.
13. Subraya, B.M y Subrahmany, S.V. (2000). Object driven Testing of Web Applications. Documento Electrónico, consultado el 7 de Julio de 2011 de IEEE. ID de Documento:0-7695-0825-1/00
14. TMMi Foundation (2010) Test Maturity Model integration (TMMi) Version 3.1 Erik Van Veenendaal. Documento electrónico del sitio TMMi Foundation el día 8 de Julio de 2011. <http://www.tmmifoundation.org/downloads/tmmi/TMMi%20Framework.pdf>
15. Toledo Federico, López Horacio, Reina Matias, Vazquez Gustavo, Uvarow Simon de, Greisin Edgardo (2008) Metodología para Pruebas de Desempeño. Documento electrónico de la pagina web de la Facultad de Ingeniería de la Universidad de la Republica – Uruguay. Ultima consulta el día 11 de Agosto de 2011. <http://www.fing.edu.uy/inco/pedeciba/bibliote/reptec/TR0820.pdf>
16. Vail, Cordell (2005) Stress, Load, Volume, Performance, Benchmark and Base Line Testing Tool Evaluation an Comparison. Documento electrónico del sitio web CiteSeerX. Ultima consulta 16 de Julio de 2011. <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.130.6847&rep=re p1&type=pdf>
17. Weyuker Elaine J., Vokolos Filippos I. (Diciembre 2000) Experience with Performance Testing of Software Systems: Issues, an Approach, and Case Study. Documento Electronico de la página IEEE Xplore, ultima consulta 15 de Agosto de 2011. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=888628&abstractAccess=no&userType=inst
18. Woodside Murray, Franks Greg, Petriu Dorina C. (2007) The Future of Software Performance Engineering. Documento Electrónico de la página web Portal ACM, ultima consulta el día 11 de Agosto de 2011. <http://portal.acm.org/citation.cfm?id=1254717>