

# Comparación de las características de algunas herramientas de software para pruebas de carga

## Feature comparison among some software tools for load testing

Carlos Mario Zapata J.<sup>1</sup> Ph.D. & Crhistian de Jesús Cardona Velásquez. M.Sc. (c)

1. Grupo en Lenguajes Computacionales, Escuela de Sistemas, Universidad Nacional de Colombia, sede Medellín

2. Maestría en Data Mining, Universidad de Buenos Aires, Argentina

{cmzapata, cdcardon}@unal.edu.co

Recibido para revisión 13 de abril de 2011, aceptado 28 de junio de 2011, versión final 01 de junio de 2011

**Resumen**— Las pruebas de los productos de software procuran altos estándares de calidad para lograr la satisfacción del cliente. La utilización de herramientas de pruebas de carga y el aseguramiento de la fiabilidad y eficiencia de los productos de software marcan las pautas actuales en la construcción de aplicaciones *Web*. Estas herramientas simulan conexiones simultáneas de usuarios virtuales y permiten encontrar los puntos de quiebre de los aplicativos, revelando problemas de arquitectura o configuración. En este artículo se comparan algunas herramientas que permiten realizar las pruebas de carga, sus características y las ventajas competitivas. Se detecta una tendencia marcada hacia aplicaciones integrales que permitan realizar pruebas de carga, ejecutar monitoreos globales y aplicar automatización de pruebas funcionales en un mismo entorno.

**Palabras clave**— Pruebas, prueba de carga, concurrencia, rendimiento.

**Abstract**— Software testing tools try to guarantee high-quality standards of applications, in order to achieve customer satisfaction. There are some trends in using load testing tools for reliability and efficiency assurance of Web applications. These tools detect breaking points and architecture & configuration problems by simulating synchronous connections of virtual users. In this paper, we compare some software tools for load testing, their features, and their competitive advantages. As a result, we find a trend towards unified applications for load testing, global monitoring, and functional testing automation.

**Keywords**— Test, Load test, Turnout, Performance.

### I. INTRODUCCIÓN

Las pruebas de carga llevan al límite una aplicación *Web*, mediante la emulación de determinado número de usuarios con diversos periodos de conexión simultánea y distintas funcionalidades específicas en ejecución. Estas pruebas se relacionan directamente con las pruebas de rendimiento, debido

a que éste se afecta a medida que la aplicación de software llega a su límite. Por ello, el monitoreo constante y los sistemas que apoyan los aplicativos *Web* toman relevancia al realizar este tipo de pruebas.

Las aplicaciones *Web* actuales, además de factores de funcionalidad y operatividad, exigen, más que nunca, niveles de calidad del servicio (QoS) óptimos [1] [2], directamente relacionados con los tiempos de respuestas y la disponibilidad del servicio. Es por esto que las pruebas de carga tienen una gran relevancia, ya que permiten encontrar el límite óptimo del funcionamiento de una aplicación *Web* determinada para detectar fallos de diseño de tipo arquitectónico. De esta manera, se facilita la realización de ajustes en las aplicaciones para mejorar su rendimiento.

Las pruebas de carga, también, se pueden realizar sobre servicios basados en protocolos de Internet y servicios especializados, como: FTP, Autenticaciones LDAP, *Web Services* (SOAP), bases de datos, *etc.* Estas funcionalidades, unidas con algunos otros factores propios de herramientas para la realización de pruebas de carga, hacen que una aplicación especializada en dichas pruebas se use de forma más conveniente en algunos sistemas o ambientes que en otros.

En este artículo, se compararan 19 herramientas de software para realizar pruebas de carga a aplicaciones basadas en protocolos de Internet y software, con arquitecturas cliente servidor, buscando las características más representativas de las herramientas y analizando la tendencia del desarrollo de las mismas en cuanto a funcionalidad se refiere. Para la comparación, se identifican grupos de aplicaciones con características similares, permitiendo seleccionar alguna de ellas según el ambiente y las características de lo que se quiere probar.

El artículo se estructura así: en la siguiente sección, se describen los elementos conceptuales relevantes en el tema de las pruebas de carga; posteriormente, se listan y explican las

características a tener en cuenta al realizar el estudio comparativo; seguidamente, se definen tres categorías de herramientas de pruebas y se detalla cada herramienta de cada categoría con sus características específicas; a continuación, se realizan los análisis de los resultados de las pruebas comparativas; finalmente, se presentan las conclusiones y el trabajo futuro.

## II. ANÁLISIS TEÓRICO

Las pruebas de carga, también conocidas en algunos casos como pruebas de *stress* o *test load*, tienen por objeto emular la conexión a un aplicativo *Web* de determinado número de usuarios, con el fin de medir la reacción de éste y del sistema donde reside, cuando la concurrencia alcanza niveles específicos [1] [2] [3].

Una herramienta de carga (*software tool for load testing*) opera enviando continuamente peticiones a un sitio *Web* y parando por periodos de tiempos programables, para comenzar de nuevo con el envío de peticiones continuas, concurrentes y escalables, tanto como el sistema y el software de prueba lo permitan (Véase la Figura 1). Cada ingreso al aplicativo *Web* se denomina usuario virtual y permite [1]:

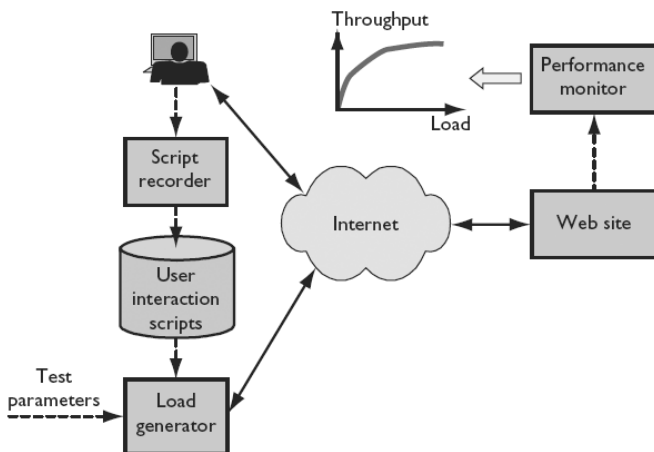


Figura 1. Proceso de las pruebas de carga [1]

- Obtener resultados similares a los que se logran con usuarios reales conectados en forma concurrente.
- Obtener respuestas negativas por ingresos no concluidos, abandonos de sesión y rechazo de peticiones por tiempo excesivo de respuesta.

### 2.1. Diferencia entre pruebas de carga, pruebas de rendimiento y pruebas de estrés.

Generalmente, se tiende a confundir estos tres tipos de pruebas, debido a que, en muchas ocasiones, se realizan paralelamente. Los objetivos de dichas pruebas permiten encontrar sus diferencias y entender por qué en algunos casos se intercambian al mencionarse:

a. Pruebas de rendimiento (*performance testing*): Su objetivo principal es desarrollar estrategias eficaces para mejorar el rendimiento del sistema. En las pruebas de rendimiento se recopila y analiza información mediante un proceso de medición, en el que se recogen datos para predecir cuándo los niveles de carga agotarán los recursos del sistema [2] [18].

b. Pruebas de carga (*load testing*): Evalúan el rendimiento del sistema con una carga predefinida. La prueba de carga mide cuánto se tarda un sistema para realizar diversas tareas y funciones del programa bajo condiciones normales o predefinidas. Debido a que el objetivo de las pruebas de carga es determinar si el rendimiento del sistema satisface los requisitos no funcionales de carga, es pertinente determinar, antes de comenzar las pruebas, la configuración mínima y máxima y los niveles de actividad [3] [4].

c. Pruebas de estrés (*stress testing*): Evalúan el comportamiento de los sistemas, cuando se llevan más allá de sus límites operacionales (que pueden ser muy superiores a los requisitos no funcionales). Se evalúan las respuestas del sistema y de la aplicación a periodos de mayor volumen de actividad, que superen las limitaciones del sistema. El objetivo principal de las pruebas de estrés es determinar si un sistema se bloquea o se recupera en dichas condiciones. Las pruebas de estrés se deben diseñar para llevar los límites de los recursos del sistema, hasta exponer los puntos débiles de la aplicación [3] [4] [5].

### 2.2. Relación entre las pruebas de carga y el rendimiento del sistema.

El uso de pruebas de carga, para predecir el rendimiento de un aplicativo *Web*, es muy útil en sistemas con requisitos de carga altos. Son factores básicos de los análisis realizados en las pruebas de carga [1]:

$N_{VU}$ : Número de usuarios virtuales concurrentes, es decir, el número de usuarios que se emulan para usar el sistema al mismo tiempo.

$Z$ : Tiempo entre las peticiones realizadas al servicio o sistema probado. Así, entre cada petición realizada, emulando los usuarios virtuales concurrentes, se debe tener un tiempo entre cada una de ellas para simular la realidad de accesos en ambientes productivos. Generalmente se toman segundos, dado que lo que se espera es que cada segundo cierto número de usuarios virtuales consuman el servicio específico a la vez.

$R$ : Tiempo promedio de respuesta por cada petición del servicio o sistema probado. Constituye la medida de respuesta de servicio o sistema probado y, dependiendo de si esta respuesta es rápida o lenta, este tiempo puede permitir una concurrencia alta o baja, respectivamente.

$X_O$ : Tiempo promedio de peticiones por segundo realizadas al servicio o sistema probado.

Se usa, entonces, la ley del tiempo de respuesta [6] [7]

$$R = \frac{N_{vu}}{X_o} - Z$$

En general, el gráfico de tiempos de respuesta *versus* respuestas recibidas o usuarios virtuales concurrentes (Véase la Figura 2) es el estadístico más diciente en una prueba de carga, pues relaciona la carga con el rendimiento de la aplicación y permite el estudio de *R*.

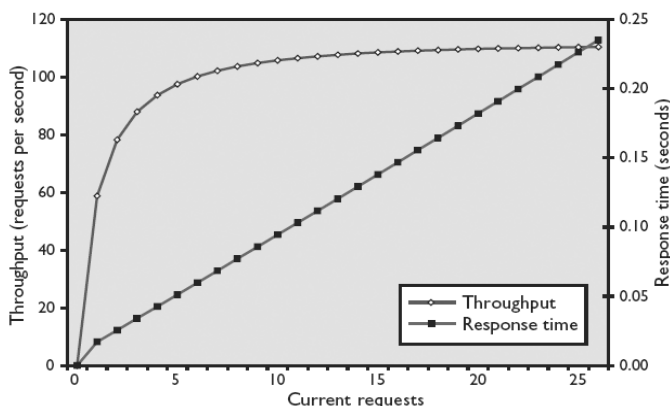


Figura 2. Gráfica de carga vs. rendimiento [1].

Para las herramientas a evaluar, algunas de las características relevantes en las pruebas de carga se relacionan con el comportamiento de los siguientes tiempos [8]: consulta DNS, conexión TCP, obtención de paquetes, redirección, obtención de página base y obtención de contenido.

Otros conceptos utilizados en el ámbito de las pruebas de carga, que son relevantes a la hora de realizar un análisis de resultados y se relacionan, directamente, con las características a medir mediante las herramientas de pruebas de carga, son:

- Hits: Solicitud que hace un servicio, aplicación o explorador *Web* a un servidor *Web*. Así, por ejemplo, cada imagen de una página *Web* genera *hits* [9] [10].
- Punto de quiebre: Cantidad máxima de usuarios concurrentes que la aplicación puede soportar antes de no entregar una respuesta efectiva.
- UVC (Usuarios Virtuales Concurrentes): Cantidad de usuarios virtuales que se configuran para generar la concurrencia establecida en la prueba de carga.
- Sesión: Tiempo en que el usuario virtual ejecuta la acción programada mediante la herramienta de carga.
- MTR (Máximo Tiempo de Respuesta): Variable que permite medir el tiempo máximo de respuesta que arroja algún objeto o elemento probado con la herramienta.
- Throughput [3]: Respuesta del servidor de aplicaciones a la petición que envía el generador de carga. Incluye lo relacionado con la capacidad de transmisión de un medio

de comunicación en cualquier momento y se suele medir en *bits* por segundo (bps).

- **Response time** (Tiempo de respuesta): Tiempo transcurrido entre la petición y la respuesta de un servicio cliente-servidor [11].

Recientemente, las compañías de software descubrieron que la clave, a largo plazo, es lograr una ventaja competitiva en el mercado, mediante la satisfacción del cliente empleando la fiabilidad y la eficiencia, que son características de un sistema de software con calidad según la norma ISO 9126 [12] y, además, mediante la rapidez y tiempos de respuesta competitivos presentes en sus aplicativos [13] [14]. Por esta razón, las pruebas de carga siempre están presentes en los planes de prueba de sus aplicaciones y, como consecuencia, a mediano plazo, se puede observar una reducción en los costos de mantenimiento [15].

Al iniciar el ciclo de mantenimiento de una aplicación de software, se debe incluir en el plan una prueba de carga en condiciones de uso reales y en un ambiente de pruebas similar al que se utilizará en la fase de producción [16]. La realización de pruebas de carga, a menudo, acarrea altos costos a las pruebas de las aplicaciones. Estos costos se pueden justificar con beneficios, al aplicar acciones correctivas durante la prueba que potencialicen la fiabilidad del producto de software [17]. Además de las aplicaciones *Web* [2] [18], las pruebas de carga se realizan a software de teléfonos móviles [19], software de manejo de hardware especializado [20] y circuitos electrónicos específicos (pruebas de carga realizadas mediante modelos estadísticos [14]). De hecho, el ámbito de pruebas de carga a aplicaciones de software se nutrió, inicialmente, con las experiencias adquiridas por otras áreas de la ingeniería sobre otros objetos o construcciones.

Al evaluar posibles herramientas para realizar pruebas de carga, surgen conceptos y consideraciones para realizar una escogencia correcta, de acuerdo con el ambiente y las necesidades organizacionales y funcionales de la prueba.

### 2.3. Características a medir en el estudio comparativo

Para realizar la comparación, se proponen 35 características (entre factores y funcionalidades) relacionados con cada una de las herramientas para realizar pruebas de carga [11]. A cada característica se le asigna un peso subjetivo, pero tomando en consideración la recomendación del Proceso Jerárquico Analítico (AHP, por sus siglas en inglés) [21], que es un enfoque genérico para la solución de problemas en toma de decisiones complejas multicriterio. El peso de cada característica se asigna con su intensidad de importancia, tomando como referencia los siguientes valores, que representan la relevancia de la característica para realizar una posible selección de las herramientas [21]:

- {1} Igual importancia: Dos actividades contribuyen igualmente al objetivo.
- {3} Importancia moderada: La experiencia y el juicio favorecen ligeramente una sobre otra.

- {5} Importancia fuerte: La experiencia y el juicio favorecen fuertemente una sobre otra.
- {7} Importancia muy fuerte: Se favorece fuertemente una actividad y su predominio se demuestra en la práctica.
- {9} Importancia absoluta: La importancia de una sobre otra se afirma en su máximo orden posible.

Los valores {2}, {4}, {6} y {8} son valores intermedios y se usan para representar compromiso entre las prioridades listadas. Las características a evaluar se describen seguidamente. El número entre llaves es la intensidad de importancia que se asignará en cada caso.

1. *Dirección url del fabricante*: sitio *Web* en el cual se encuentran los manuales, explicaciones y archivos instaladores de la herramienta {1}.
2. *Tipo de licencia y precio*: el tipo de licencia delimita la forma de uso y se relaciona, directamente, con el precio ya que, dependiendo de las capacidades o límites del aplicativo, los precios suben o bajan. Generalmente, en este tipo de aplicaciones el máximo de usuarios concurrentes limita el precio {2}.
3. *Acceso al código fuente*: posibilidad de acceder al código fuente de la herramienta y realizar modificaciones en ella. Esta característica depende del tipo de licencia que posea la herramienta {2}.
4. *Plataforma*: sistemas operativos que soporta o el lenguaje de programación que se usa para la construcción de la herramienta y que puede permitir portabilidad entre sistemas, como en el caso de las herramientas programadas en Java, que son portables entre sistemas Windows, Linux, UNIX y Solaris, entre otros {2}.
5. *Multiplataforma*: Característica que describe qué tan portable es la herramienta {8}.
6. *Requisitos de hardware*: los necesarios para que la herramienta se ejecute correctamente. Se refiere al tipo de procesador, memoria RAM y disco duro {2}.
7. *Idiomas soportados*: el hecho de que las herramientas sean multilingües, les da la capacidad de ser fácilmente adaptables sin importar la cultura en la cual se utilicen. Por ello, esta es una de las características deseables en las herramientas a analizar {6}.
8. *Manejo de perfiles de usuarios virtuales*: en las aplicaciones *Web* se suelen manejar grupos de usuarios con características y perfiles distintos. La posibilidad de emular los tipos de usuarios mediante usuarios virtuales, para lograr agrupaciones que correspondan a la realidad de un ambiente operativo, facilita el diseño de las pruebas {7}.
9. *Uso de variables*: algunas herramientas permiten agregar contadores o variables que almacenan el número de veces que se carga un segmento de código específico durante la prueba, con el fin de construir gráficas personalizadas {5}.
10. *Soporte IP Spoofing [2]*: permite a la herramienta de pruebas asignar una IP a un usuario virtual para acercar la prueba a la realidad de las peticiones concurrentes de usuarios conectados en distintas estaciones de trabajo {5}.
11. *Visualización en tiempo real*: permite visualizar los resultados de las pruebas, las gráficas y las tablas de datos mientras la prueba se ejecuta {9}.
12. *Pruebas programadas*: permite programar el inicio, las paradas y el fin de un plan de pruebas específico {9}.
13. *Proxy HTTP*: para que las pruebas se realicen de forma automática, en vez de generar el *script* con los pasos que se deben seguir para la prueba, se utiliza esta funcionalidad que hace interfaz con el navegador *Web* normal y graba las acciones que se realizan en él, generando las instrucciones o *scripts* necesarios para realizar la prueba específica. En el caso de HTTP, se monitorea todo el flujo y se graban las peticiones y respuestas no codificadas {9}.
14. *Proxy HTTPS*: tiene el mismo objetivo y filosofía del *proxy* HTTP, sólo que éste graba de forma especial las secuencias, debido a que debe codificar y decodificar las peticiones y respuestas y usa constantemente las *key* y certificados propios de SSL dependiendo de la versión y del cifrado utilizado (128, 512 bits, etc.). Muchas herramientas no permiten la grabación de páginas encriptadas con HTTPS, debido a su complejidad, lo que genera ventajas comparativas para las que sí lo hacen {9}.
15. *Scripting [2]*: al realizar el diseño de las pruebas, la forma en que éstas se programan o estructuran tiene mucho que ver con la fácil manipulación que se le puede dar a la misma. Se suele tener un lenguaje que define el fabricante de la herramienta para diseñar las órdenes de petición, espera, generación de flujos de navegación, etc. En otros casos, algunas herramientas cuentan con lenguajes más conocidos, que permiten realizar el diseño de las pruebas de forma más amplia y fácil, con la ventaja de que no se requiere aprender un nuevo lenguaje programático para generarlas {9}.
16. *Controladores Lógicos*: permiten asignar un valor a determinada variable de la prueba, si se da cierto evento lógico. Entre ellos, se encuentran las instrucciones. *if*, *while*, *do while*, y *for* {9}.
17. *Número de informes nativos*: mientras más informes predeterminados tenga, se facilita la interpretación de resultados y el tiempo de la prueba se reduce {5}.
18. *Diseño de informes personalizados*: las herramientas traen consigo, por defecto, informes específicos para el análisis de los resultados de las pruebas. Sin embargo, algunas permiten la manipulación de variables propias en las



pruebas y la construcción de gráficas personalizadas para un mejor análisis {5}.

19. *Protocolos*: elementos de comunicación para capturar, manipular y emular peticiones. Entre ellos, los más comunes son HTTP 1.0 / 1.1 / HTTPS (SSL). Es uno de los requisitos básicos de las herramientas y se considera una ventaja comparativa el hecho de que posea la funcionalidad de proxy HTTP o HTTPS {9}.
20. *Monitoreo de bases de datos*: funcionalidades propias de la aplicación, que se conectan directamente con el motor de base de datos que utiliza la aplicación *Web* probada y permite visualizar su comportamiento en términos de rendimiento y uso de recursos, a medida que la prueba se ejecuta {7}.
21. *Monitoreo Sistema Operativo*: funcionalidades propias de la aplicación, que se conectan directamente con el sistema operativo y muestran, mediante gráficas y tablas, el comportamiento de los recursos internos, a medida que se ejecuta la prueba {7}.
22. *Monitoreo Web server [11]*: funcionalidad que permite conectarse a un servidor de aplicaciones y monitorear el rendimiento y la memoria utilizada, entre otros factores, con el fin de visualizar su comportamiento durante un periodo definido {7}.
23. *Pruebas LDAP (Lightweight Directory Access Protocol)*: que se realizan directamente sobre un servicio tipo LDAP y permiten encontrar la capacidad o límite de servicio de usuarios concurrentes autenticándose en el mismo {5}.
24. *Pruebas FTP*: que se realizan sobre un servicio de transferencia de archivos FTP y permiten encontrar el límite de servicio de usuarios concurrentes autenticándose y enviando o recibiendo información al mismo tiempo {5}.
25. *Pruebas de Web services [10]*: los servicios *Web*, que funcionan bajo el lenguaje SOAP, constituyen, hoy en día, una de las tendencias más populares para la construcción de aplicaciones *Web*. Una herramienta que permite pruebas de carga a dichos servicios es apta no sólo para enviar peticiones e instanciar el servicio congruente con SOAP, sino también para recibir mensajes e interactuar completamente con dicho servicio {9}.
26. *Pruebas Bases de datos*: mediante el envío de consultas, ejecución de procesos almacenados y generación de vistas temporales, estudian el rendimiento y la capacidad de carga de la base de datos en cuestión {7}.
27. *Pruebas mail-Server*: o pruebas a servidores de correo, mediante peticiones simultáneas de recepción y envío de correos {5}.
28. *Manejador de Cookies*: las *cookies* son archivos temporales que manejan información de sesión para agregar funcionalidades específicas a los aplicativos *Web*. Es muy importante cuando se realizan pruebas de carga con flujos de navegación específicos en aplicativos que usan las *cookies*, pues su carencia provoca errores de denegación de servicio que no aportan a la medición ningún dato objetivo {9}.
29. *Administración remota*: implica la posibilidad de realizar la configuración y todas las funcionalidades propias de una prueba a distancia, mediante una conexión a un puerto específico desde un ambiente cliente compatible con la herramienta. Esto, es útil en casos en que las pruebas se deben llevar a cabo en ambientes controlados de producción o en casos en que se utilicen pruebas en segmentos múltiples o en ambientes distribuidos, en donde se deben monitorear varias instancias de la herramienta a la vez {9}.
30. *Temporizadores*: permiten programar tareas de inicio de carga de hilos concurrentes, teniendo en cuenta un patrón específico, en cuya función se obtiene una variable temporal que dispara el inicio del evento {9}.
31. *Pruebas en segmentos múltiples*: se refiere a la posibilidad de realizar una prueba desde distintos puntos de una arquitectura de red, ya sea aplicando el mismo plan de pruebas o uno distinto. Se usa tanto para evaluar el comportamiento de la aplicación *Web* probada bajo la carga desde distintos nodos de una red, como para potenciar más la carga de trabajo, ya que, en algunos casos, el ambiente en el cual se ejecuta la herramienta para realizar la prueba se puede sobrecargar y arrojar resultados incoherentes {7}.
32. *Pruebas funcionales*: en herramientas muy especializadas, permiten inicialmente realizar pruebas funcionales basadas en requisitos, generalmente programables por medio de *scripts*. En algunas, se pueden realizar pruebas de carga basadas en las pruebas funcionales programadas, permitiendo un ahorro de tiempo y un desempeño más realista {9}.
33. *Posibilidad de extensiones*: permite incrementar las funcionalidades de la herramienta mediante adiciones, *scripts* o subprogramas {8}.
34. *Emulación de velocidad de conexión de usuarios*: suministra mayor realismo a las pruebas {9}.
35. *Escalabilidad de usuarios*: permite generar un número de usuarios virtuales, de acuerdo con los recursos específicos de la máquina en la cual se ejecuta la prueba. Este factor depende del número, tamaño y complejidad del *script* de prueba {9}.

### III. ANÁLISIS DE RESULTADOS

Para comparar las herramientas para pruebas de carga, se crearon 3 niveles, que toman en cuenta las características que posee cada herramienta y, por ende, su funcionalidad general (Véanse las tablas 1, 2 y 3, que contienen el listado detallado de características por herramienta):

- Bajo: corresponde a las herramientas con menor número de características presentes en su estructura funcional.
- Medio: corresponde a las herramientas con un número de características presentes en su estructura funcional aceptable y que se consideran normales o estándar.
- Avanzado: corresponde a las herramientas que, además de tener el mayor número de funcionalidades propias de pruebas de carga, poseen características adicionales que las hace poseedoras de ventajas comparativas muy altas frente a las demás.

Para estas tablas, se conserva la nomenclatura empleada para la intensidad de importancia. En la característica se coloca la intensidad que sirve de base y en las herramientas se coloca la intensidad asignada, según la característica. En el proceso de análisis de las 19 herramientas se evidenciaron los siguientes detalles:

- Las herramientas que componen el nivel bajo tienen entre el 0% y el 35% de las funcionalidades evaluadas

positivamente, las de nivel medio entre el 35% y el 46% y las de nivel avanzado entre el 46% y el 63%.

- Sólo el 21% de las herramientas evaluadas tiene la posibilidad de realizar pruebas de funcionalidad y la mayoría pertenece al nivel avanzado, lo cual podría indicar que su ausencia o presencia es uno de los factores diferenciadores de las herramientas de carga.
- El 21% de las herramientas evaluadas es *open source* (código libre), lo cual permite utilizar gratuitamente todas sus funcionalidades y modificar el código según se necesite. En muchos casos, las herramientas hacen parte de empresas que aportan código y toman funcionalidades de la comunidad. Además, estas empresas venden algunas funcionalidades mucho más avanzadas bajo licencia de protección a los derechos de autor, modalidad muy común en la actualidad y que permite que el desarrollo de productos de diversas funcionalidades avance ostensiblemente.

**Tabla 1.** Herramientas de nivel bajo (parte 1/2)

#	Característica	vPerformer	StressTester	LoadManager	Visual Studio Team System 2008 Test Load Agent	Websver Stress Tool	TestComplete 6	Jblitz
1	Dirección url del fabri-cante {1}	www.verisium.com {1}	www.stresstester.net {1}	www.alvicom.hu {1}	www.microsoft.com {1}	www.paessler.com {1}	www.automatedqa.com {1}	www.clanproductions.com {1}
2	Tipo licencia y precio {2}	\$S995.00 (50 UV), US\$2.995.00 (100 UV) y más de 200 UV necesita contacto con el fabricante {2}.	Licencia con precios pactados por contacto con distribuidor. Permite evaluar demo previa suscripción al sitio {0}.	Producto licenciado. Precio directamente dado por contacto {0}.	Prueba de 90 días. Licencia acoplada al valor de la licencia Visual Studio Team System 2008 - Aproximadamente desde US\$5190 {2}	Desde \$249.95 {2}	US\$1999 – Licencia completa {2}	Licenciado. Precios mediante contacto directo con distribuidor {0}
3	Acceso al código fuente {2}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}
4	Plataforma {2}	Windows XP/2000/2003/NT {2}	Windows, Linux y Unix {2}	Linux, Windows, HP Unix {2}	Microsoft Windows 98/Me/NT/2000/XP/2003 {2}	Windows 98/ME/NT/2000/XP/2003/Vista {2}	Windows 98/ME/NT/2000/XP/2003/Vista/NT {2}	Cualquier plataforma que soporte Java 2 Standard Edition 1.4.2 o posterior {2}
5	Multiplataforma {8}	No {0}	Si {8}	Si {8}	No {0}	No {0}	No {0}	Si {8}
6	Requisitos de hardware {2}	3 sistemas independientes pero no especificadas sus características {2}	No especificados {0}	No especificados {0}	2.0-GHz CPU, 512 MB RAM, 8-GB de Disco Duro {2}	Procesador 1 Ghz – RAM 512 MB {2}	Intel Pentium 4 3 GHz – RAM 1 GB {2}	No especificado {0}
7	Idiomas soportados {6}	Inglés {0}	Inglés {0}	Inglés {0}	Inglés {0}	Inglés {0}	Inglés {0}	Inglés {0}
8	Manejo de perfiles de usuarios virtuales {7}	Si {7}	Si {7}	No especificado {0}	Si {7}	Si (Limitados) {7}	Si {7}	Si {7}
9	Uso de variables {5}	No {0}	No {0}	Si {5}	No {0}	Si {5}	Si {5}	No {0}
10	Soporte IP Spoofing {5}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}
11	Visualización en tiempo real {9}	Si {9}	Si {9}	No especificado {0}	Si {9}	Si {9}	Si {9}	Si {9}
12	Pruebas programadas {9}	No {0}	No {0}	No especificado {0}	No {0}	Si {9}	Si {9}	Si {9}
13	Proxy http {9}	Si {9}	Si {9}	No especificado {0}	No {0}	Si {9}	Si {9}	No {0}

Comparación de las características de algunas herramientas de software para pruebas de carga  
– Zapata & Cardona

#	Característica	vPerformer	StressTester	LoadManager	Visual Studio Team System 2008 Test Load Agent	Websserver Stress Tool	TestComplete 6	Jblitz
14	Proxy https {9}	No {0}	Si {9}	No especificado {0}	No {0}	No {0}	Si {9}	No {0}
15	Scripting	Si	Si	Si		Si	Si	Si
16	Controladores Lógicos {9}	No {0}	No {0}	No especificado {0}	No {0}	Si {9}	Si {9}	No {0}
17	Número de informes nativos {5}	10 {5}	Entre 4 y 10 {5}	No especificado {0}	No especificado {0}	10 {5}	+8 {5}	7 {5}
18	Diseño de informes personalizados {5}	Si {5}	Si {5}	No especificado {0}	Si {5}	No {0}	No {0}	No {0}
19	Protocolos {9}	http {9}	Http-https {9}	http-https {9}	http-https {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}
20	Monitoreo de Bases de datos {7}	Si (5 tipos) {7}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}
21	Monitoreo Sistema Operativo {7}	Si (2 tipos) {7}	No {0}	No especificado {0}	Si {7}	Si (1) {7}	No {0}	No {0}
22	Monitoreo Web Server {7}	Si (11 tipos) {7}	Si {7}	No especificado {0}	Si {7}	Si (5) {7}	No {0}	No {0}
23	Pruebas LDAP {5}	No {0}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}
24	Pruebas FTP {5}	No {0}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}

Tabla 1. Herramientas de nivel bajo (parte 2/2)

#	Característica	vPerformer	StressTester	LoadManager	Visual Studio Team System 2008 Test Load Agent	Websserver Stress Tool	TestComplete 6	Jblitz
1	Pruebas Web Services {9}	No {0}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}
2	Pruebas Bases de datos {7}	No {0}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}
3	Pruebas Mail-Server {5}	No {0}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}
4	Manejador de Cookies {9}	Si {9}	Si {9}	No especificado {0}	Si {9}	Si {9}	Si {9}	Si {9}
5	Administración remota {9}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}
6	Temporizadores {9}	No {0}	Si {9}	No especificado {0}	No {0}	Si (2) {9}	Si {9}	Si {9}
7	Pruebas en segmentos múltiples {7}	Si {7}	No {0}	No especificado {0}	No {0}	No {0}	Si {7}	No {0}
8	Pruebas funcionales {9}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}	Si {9}
9	Posibilidades de extensión {8}	No {0}	No {0}	No especificado {0}	No {0}	No {0}	No {0}	No {0}
10	Emulación de velocidad de conexión de usuarios {9}	Si {9}	No {0}	No especificado {0}	No {0}	Si {9}	No {0}	No {0}
11	Escalabilidad de usuarios {9}	No especificado {0}	No especificado {0}	No especificado {0}	Depende del número de UV {9}	No especificado {0}	No especificado {0}	No especificado {0}
12	TOTALES {234}	{108}	{98}	{34}	{69}	{119}	{114}	{88}





Comparación de las características de algunas herramientas de software para pruebas de carga  
– Zapata & Cardona

#	Nombre	OPENSTA	PROXY SNIFFER	Qtest 5.0	Web performance load tester	Web performance load tester	WEBLOAD
18	Administración remota {9}	No {0}	Si {9}	No {0}	No {0}	No {0}	No {0}
19	Temporizadores {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si (5) {9}
20	Pruebas en segmentos múltiples {7}	Si {9}	Si {9}	No {0}	No {0}	No {0}	No {0}
21	Pruebas funcionales {9}	No {0}	No {0}	No {0}	No {0}	No {0}	No {0}
22	Posibilidades de extensión {8}	Si {8}	No {0}	Si {8}	No {0}	No {0}	Si {8}
23	Emulación de velocidad de conexión de usuarios {9}	No {0}	Si {9}	No {0}	Si {9}	Si {9}	Si {9}
24	Escalabilidad de usuarios {9}	No Especificado {0}	No Especificado {0}	No Especificado {0}	No Especificado {0}	No Especificado {0}	No Especificado {0}
25	TOTALES {234}	{109}	{135}	{137}	{114}	{114}	{111}

Tabla 3. Herramientas de nivel avanzado (Parte 1/2)

#	Nombre	MINQ - PureLoad	Jmeter	QEngine	TestMaker	NeoLoad	APPPERFECT TEST SUITE STUDIO
1	Dirección url del fabricante {1}	www.minq.se {1}	jakarta.apache.org/jmeter {1}	www.adventnet.com {1}	www.pushtotest.com {1}	www.neotys.com {1}	www.appperfect.com {1}
2	Tipo licencia y precio {2}	Edición Web, \$8,990 para Usuarios virtuales ilimitados Edición empresarial inicial para 50 usuarios virtuales US \$9,990 {2}	Freeware – Opensource {2}	De US\$795 a \$12,995 dependiendo del tipo de licencia y el número máximo de usuarios virtuales {2}	Freeware – Opensource {2}	De € 742 a € 32,063 dependiendo de la licencia más los módulos adicionales que están entre € 2,142 y € 3,245 {2}	Toda la suite con las funcionalidades totales desde US\$2980. Por componente el precio varía {2}
3	Acceso al código fuente {2}	No {0}	Si {2}	No {0}	Si {2}	No {0}	No {0}
4	Plataforma {2}	Windows NT/200/XP (Server Editions), Linux Redhat, Solaris/SPARC 8 y plataformas con Java 2 Standard Edition 1.4.2 {2}	Cualquier plataforma que soporte Java 2 Standard Edition 1.4.2 o posterior {2}	Linux y Windows {2}	Windows 2000/XP/2003/Vista, Linux, UNIX y Mac OS X {2}	Windows 2000/XP/2003/Vista Linux RedHat y Mandriva, Solaris 10 {2}	Windows 2000/XP/2003, Linux x86, Solaris 8/9, Mac OS X {2}
5	Multiplataforma {8}	Si {8}	Si {8}	Si {8}	Si {8}	Si {8}	Si {8}
6	Requisitos de hardware {2}	RAM 512 MB 20 MB HD {2}	No especificados {0}	No especificados {0}	No especificados {0}	185 MB HD RAM 150 MB {2}	No especificados {0}
7	Idiomas soportados {6}	Inglés {0}	Español, Inglés, Japonés, Noruego Alemán, Francés, Chino {6}	Inglés {0}	Inglés {0}	Inglés, Francés {6}	Inglés {0}
8	Manejo de perfiles de usuarios virtuales {7}	Si {7}	Si {7}	Si {7}	Si {7}	Si {7}	Si {7}
9	Uso de variables {5}	Si {5}	Si {5}	Si {5}	Si {5}	Si {5}	Si {5}
10	Soporte IP Spoofing {5}	No {0}	Si {5}	Si {5}	No {0}	Si {5}	Si {5}
11	Visualización en tiempo real {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}
12	Pruebas programadas {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}
13	Proxy http {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}
14	Proxy https {9}	Si {9}	No {0}	Si {9}	Si {9}	Si {9}	Si {9}
15	Scripting {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}
16	Controladores Lógicos {9}	Si {9}	Si {9}	No {0}	Si {9}	Si {9}	Si {9}

#	Nombre	MINQ - PureLoad	Jmeter	QEngine	TestMaker	NeoLoad	APPPERFECT TEST SUITE STUDIO
17	Número de informes nativos {5}	+8 {5}	13 {5}	16 {5}	+10 {5}	11 + monitores {5}	+15 {5}
18	Diseño de informes personalizados {5}	No {0}	Si {5}	No {0}	No {0}	No {0}	Si {5}
19	Protocolos {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}	Http 1.0-1.1-https {9}
20	Monitoreo de Bases de datos {7}	No {0}	No {0}	Si (3) {7}	No {0}	Si (5) {7}	Si (6) {7}
21	Monitoreo Sistema Operativo {7}	No {0}	No {0}	Si (3) {7}	Si (1) {7}	Si (5) {7}	Si (5) {7}
22	Monitoreo Web Server {7}	No {0}	Si (1) {7}	No {0}	No {0}	Si (12) {7}	Si (10) {7}
23	Pruebas LDAP {5}	Si {5}	Si {5}	No {0}	No {0}	No {0}	No {0}
24	Pruebas FTP {5}	Si {5}	Si {5}	No {0}	No {0}	No {0}	No {0}
25	Pruebas Web Services {9}	Si {9}	Si {9}	Si {9}	No {0}	No {0}	Si {9}
26	Pruebas Bases de datos {7}	Si {7}	Si {7}	No {0}	No {0}	No {0}	No {0}
27	Pruebas Mail Server {5}	Si {5}	Si {5}	No {0}	No {0}	No {0}	No {0}
28	Manejador de Cookies {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}	Si {9}
29	Administración remota {9}	No {0}	Si {9}	Si {9}	No {0}	No {0}	Si {9}
30	Temporizadores {9}	Si {9}	Si {9}	Si (5) {9}	Si {9}	Si (3) {9}	Si {9}
31	Pruebas en segmentos múltiples {7}	Si {7}	Si {7}	Si {7}	Si {7}	No {0}	Si {7}

Tabla 3. Herramientas de nivel avanzado (Parte 2/2)

#	Nombre	MINQ - PureLoad	Jmeter	QEngine	TestMaker	NeoLoad	APPPERFECT TEST SUITE STUDIO
1	Pruebas funcionales {9}	No {0}	No {0}	Si {9}	Si {9}	No {0}	Si {9}
2	Posibilidades de extensión {8}	Si {8}	Si {8}	Si {8}	Si {8}	Si {8}	Si {8}
3	Emulación de velocidad de conexión de usuarios {9}	No {0}	No {0}	Si {9}	Si {9}	Si {9}	Si {9}
4	Escalabilidad de usuarios {9}	No especificada {0}	No especificada {0}	500 Mb por 250 UV {9}	No especificada {0}	No especificada {0}	No especificada {0}
5	TOTALES {234}	{144}	{161}	{168}	{138}	{141}	{175}

- El 89% de las herramientas sólo soportan el idioma inglés lo cual puede causar una baja tasa de utilización en regiones en las cuales este idioma no es muy popular y puede explicar el hecho de que herramientas como *Jmeter*, que tienen soporte para varios lenguajes, sean muy populares en la comunidad latina.
- El 68% de las herramientas son multiplataforma, lo que indica una tendencia del mercado hacia la promoción de herramientas de este tipo. Esta cifra coincide con el hecho de que el 47% de dichas herramientas permite realizar pruebas en múltiples segmentos y, en algunos casos, llevan a realizar pruebas en sistemas o plataformas de distinto tipo.
- El 47% de las herramientas evaluadas permite la utilización de extensiones que, generalmente, son componentes adicionales de monitoreo. Es por esto que dicha característica

se convierte en un elemento diferenciador en el mercado de las herramientas de carga.

- El número promedio de reportes nativos de las herramientas evaluadas es de 12 y demuestra que uno de los puntos fuertes de una herramienta de este tipo se apoya en la posibilidad de visualizar y evaluar los resultados de las pruebas correctamente.
- El hecho de que el sólo el 5% de las herramientas tenga una descripción de la escalabilidad de los usuarios deja varios puntos muertos en la planeación de las pruebas, ya que no se pueden estimar los requisitos de infraestructura necesarios. Una posible explicación es que son muchos los factores que pueden influir en las pruebas, lo que hace sumamente difícil, para las empresas desarrolladoras de las herramientas evaluadas, el cálculo de la escalabilidad de la infraestructura

a usar con determinados usuarios virtuales concurrentes.

- La unidad de comparación definida, denominada intensidad de importancia, permite señalar a *Webserver Stress Tool* como la mejor de las herramientas de nivel bajo, con una intensidad incluso superior a varias de las herramientas de nivel medio.
- En el nivel medio, dos herramientas se disputan de cerca el primer lugar: *Qtest 5.0* y *Proxy Sniffer*, pero esta vez con puntajes levemente inferiores al menor puntaje del nivel avanzado.
- Finalmente, en el nivel avanzado la unidad de comparación permitió escoger a *APPPERFECT TEST SUITE STUDIO* como la mejor herramienta, seguida por *QEngine* y *Jmeter*.
- Se debe anotar que los resultados obtenidos en este artículo se basan en apreciaciones subjetivas de las diferentes características evaluadas, pero pueden suministrar algunos indicios de la naturaleza de las características que serían deseables para herramientas de pruebas de carga.

#### IV. CONCLUSIONES

En términos reales, el factor más relevante a la hora de realizar el análisis de rendimiento de un aplicativo sometido a una prueba de carga es el índice de respuesta efectiva del aplicativo *Web* ante peticiones de usuarios virtuales. En los últimos años, la industria del software asumió el reto comercial de vender productos con altos estándares de fiabilidad y eficiencia, por lo que las pruebas de carga y rendimiento se volvieron obligatorias en todos los planes de prueba de los proyectos de software. Esta necesidad se evidencia en la cantidad y calidad de productos de este tipo disponibles en el mercado. La selección adecuada de la herramienta de carga parte del conocimiento que se tenga de las herramientas disponibles. Por ello, un trabajo de comparación como el que se aborda en este artículo es de suma importancia para evaluar las características del entorno y la aplicación.

La tendencia actual, en el mercado de las herramientas de pruebas de carga, se orienta al suministro de soluciones integrales, que permitan la realización de pruebas de carga, posibiliten el monitoreo de servicios y sistemas y automaticen las pruebas funcionales. Además, la portabilidad de dichas herramientas para la realización de pruebas distribuidas hace parte de las características que se están imponiendo en este tipo de herramientas.

#### V. TRABAJO FUTURO

La posibilidad de conocer un conjunto determinado de herramientas para la realización de pruebas de carga, abre un camino claro hacia la aplicación de éstas en un caso

de prueba sobre una herramienta *Web* real, con el fin de evaluar las características ya mencionadas y su usabilidad en ambientes reales de producción. La complejidad, las limitaciones funcionales y las características de automatización en herramientas para pruebas de carga, conforman el objetivo primordial a la hora de utilizar las herramientas incluidas en este estudio comparativo, o un subconjunto de ellas, sobre un caso de prueba bajo las mismas condiciones.

Una línea adicional de trabajo futuro consiste en la ampliación de este estudio con más herramientas de este tipo, con el fin de establecer si las que se seleccionaron como “mejores” en cada uno de los niveles continúan siendo las de mejores características. Un estudio tal debería abordar, también, la definición de parámetros mucho más precisos y objetivos para la intensidad de importancia de las características a estudiar en cada herramienta, aunque este estudio procuró involucrar conocimiento experto en esta escala.

#### REFERENCIAS

- [1] D. Menasce. “Load Testing of Web Sites”. IEEE Internet Computing. Vol. 6 N° 4, pp. 70-74. 2002.
- [2] T M. J.D., B. Prashant, B. Scott, R. Dennis. Performance Testing Guidance for Web Applications, patterns & practices. Microsoft Corporation. p.221. 2007.
- [3] M. Ian. The Art of Application Performance Testing. Editorial O’Reilly. Primera edición. Estados Unidos. p. 160. 2009.
- [4] H. Nguyen. “Testing Applications on the Web”. New York: John Wiley & Sons, Inc, p. 421. 2001.
- [5] L. Gullo and R. Davis. “Accelerated Stress Testing to Detect Probabilistic Software Failures”. Proceedings of the 50<sup>th</sup> Annual Reliability & Maintainability Symposium, RAMS, Los Angeles, pp. 249-255. 2004.
- [6] P. Denning and J. Buzen. “The Operational Analysis of Queuing Network Models”. ACM Computing Surveys. Vol. 10 N° 3, pp. 225-261, 1978.
- [7] A. Menascé and V.A.F. Almeida. “Capacity Planning for Web Services: Metrics, Models, and Methods”. Prentice Hall. Upper Saddle River, N.J., Estados Unidos. 2002.
- [8] V. Beltran, J. Guitart, D. Carrera, J. Torres, E. Ayguade and J. Labarta. “Performance Impact of Using SSL on Dynamic Web applications”. Proceedings of XV Jornadas de Paralelismo, Almeria. 2004.
- [9] Saerens, M.; Fouss, F. “HITS is principal components analysis”. Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence. Campiegne, pp. 782-785. 2005.
- [10] L. Henry H. Software Performance Adn Scalability, A quantitative Approach. John Wiley & Sons, Inc. Hoboken, New Jersey, Estados Unidos de America. p. 401. 2009.
- [11] B.M. Subraya. Integrated approach to Web Performance Testing: a practitioner’s Guide. Infosys Technologies Limited. IRM Press. Estados Unidos de America. p.387. 2006.
- [12] ISO/IEC. “Software engineering—Product quality—Part 1: Quality model”. International Standard ISO/IEC 9126-1:2001(E). 2001.
- [13] W. Ehrlich, S. Lee, and R. Molisani. “Applying Reliability Measurement: A Case Study”. IEEE Software. Vol. 7 N° 2, pp. 56-64. 1990.

- [14] M. Elbert, C. Mpagazehe, and T. Weyant. "Stress testing and reliability". Proceedings of Southcon/94, Orlando, pp. 357-362. 1994.
- [15] M. Hecht. "Use of importance sampling and related techniques to measure very high reliability software". Proceedings of the IEEE Aerospace Conference, Big Sky, pp. 533-546. 2000.
- [16] D. Draheim, J. Grundy, J. Hosking, C. Lutteroth, and G. Weber. "Realistic Load Testing of Web applications". Proceedings of the Conference on Software Maintenance and Reengineering (CSMR'06), Bari, pp. 57-70. 2006.
- [17] H. Chan. "A Formulation to Optimize stress Testing". Proceedings of the 44<sup>th</sup> Electronic Components and Technology Conference, Washington, pp. 1020-1027. 1994.
- [18] M. Villalba, L. Fernández, J. Escribano y P. Lara. "Un estudio sobre rendimiento Web", Memorias de IADIS ibero-americana, Alcalá de Henares. 2004.
- [19] M. Aiman. "Stress Test on J2ME Compatible Mobile Device". Innovations in Information Technology, Dubai, pp. 1-5. 2006.
- [20] H. A. Chan, University of Cape Town, Cape Town. "Accelerated Stress Testing for Both Hardware and Software". Proceedings of the 50<sup>th</sup> Annual Reliability & Maintainability Symposium, RAMS, Los Angeles, pp. 346-351. 2004.
- [21] T.L. Saaty. "The Analytic Hierarchy Process". McGraw-Hill. New York, Estados Unidos. 1980.