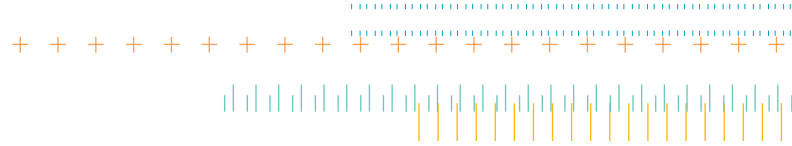




# Modelado

de patrones de comportamiento



# Modelado de patrones de comportamiento de usuarios www de segunda generación

Raúl Peña-Ortiz - Innovation Department - Intelligent Software Components S.A

Julio Sahuquillo, Ana Pont y José A. Gil - Departamento de Informática de Sistemas y Computadores de la Universidad Politécnica de Valencia

**La comprensión de las características de la carga que generan los usuarios en la web es un aspecto de crucial importancia cuando se diseña o se provee servicios en web. La mayoría de las técnicas para la caracterización de la carga actual de la web presentan limitaciones cuando tratan de representar las características dinámicas del comportamiento de los usuarios. Esto implica que la mayor parte de los generadores de carga actuales están modelando estas características de un modo simple e inapropiado. Este trabajo se centra en el comportamiento dinámico de los usuarios de la web y la forma de caracterizarlo. Nuestros esfuerzos se dirigen hacia el desarrollo de un generador de carga dinámica para web que tenga en cuenta el comportamiento actual de los clientes.**

En la actualidad los contenidos dinámicos y los servicios ofrecidos por la web suponen una segunda generación [22] en la que se pueden distinguir cambios importantes tanto en el comportamiento de los clientes como en la arquitectura e infraestructura.

Las sesiones de navegación que comienzan buscando un recurso dinámico a través de un sitio especializado [21] son un ejemplo clásico del comportamiento dinámico de los clientes. Este comportamiento no ha sido, hasta el momento, modelado con precisión. Algunos autores han realizado un importante esfuerzo para caracterizar el comportamiento de los usuarios web e incluso establecer perfiles característicos que los definan [20, 30, 18, 28]. Desafortunadamente, en las aplicaciones web actuales el comportamiento del usuario cambia continuamente pasando de un modo a otro, lo que no ha sido tenido en cuenta hasta el momento. La creciente complejidad de las aplicaciones y servicios para web, el correcto diseño de éstas así como de la infraestructura que las sustenta, requiere disponer de modelos precisos de la carga que estos sistemas van a soportar [17, 15].

En general, los puntos críticos en la caracterización de la carga web actual son: i) el modelado del comportamiento dinámico del usuario, ii) el modelado de *role* de cada usuario y iii) los continuos cambios del comportamiento de los usuarios en las aplicaciones web.

En trabajos previos [25, 27, 26], hemos especificado las bases para el desarrollo de un generador de carga denominado GUERNICA capaz de representar tanto el comportamiento dinámico de los usuarios en la web como los continuos en los patrones de navegación que éste realiza. Para ello, se ha propuesto el concepto de navegación para definir el comportamiento del usuario cuando interactúa en la web y el concepto de carga de prueba para especificar los cambios en el comportamiento de los usuarios así como representar distintos usuarios simultáneos.

En este trabajo se realiza una revisión de los generadores de carga más representativos, analizando las principales características, ventajas y limitaciones. Posteriormente, se presentan las principales características de GUERNICA y se ilustra su funcionamiento con algunos ejemplos.

## Motivación

El desarrollo de aplicaciones y servicios para la web hace necesario disponer de estudios de caracterización de la carga que permitan reproducir el comportamiento de los usuarios [17], tanto con el objetivo de evaluar prestaciones del sistema como en términos de fidelización de clientes en aplicaciones.

Los modelos de carga son abstracciones de la carga real, que intentan reproducir su comportamiento evitando aquellos aspectos que tienen escasa incidencia en un estudio particular. Un modelo es suficientemente preciso cuando reproduce el comportamiento del usuario asegurando que la aplicación web funciona de la misma manera que lo haría con usuarios reales. Los modelos de carga se clasifican en dos grandes grupos: basados en traza y generadores de carga.

Las trazas contienen la secuencia de peticiones HTTP y órdenes recibidas por una aplicación web durante un cierto período de tiempo. Estas trazas se obtienen bajo unas condiciones dadas; velocidad de proceso del servidor, ancho de banda de la red, espacio de almacenamiento en cache, etc. Lógicamente, si alguno de estos parámetros cambia la traza obtenida podría ser diferente. El principal desafío de estos modelos es conseguir una buena representatividad [17] de entornos que exhiban una gran variación. Por tanto, estos modelos son poco adecuados para modelar los cambios en el comportamiento de los usuarios, por lo que en este trabajo se centra en los generadores. Los generadores de carga son productos software diseñados e implementados para generar peticiones HTTP similares a las reales. Para poder representar distintos escenarios, los generadores pueden configurar una serie de parámetros de entrada que afectan a las principales características de la carga que deben producir. Por ello, estas herramientas son flexibles y adecuadas para llevar a cabo estudios de ajuste y planificación de carga.

Algunos estudios [19], [31] y [17] confirman la dificultad de generar peticiones representativas cuando se trata de modelar las características de un sitio web dinámico y como éstas afectan a los usuarios.

## Trabajo relacionado

En esta sección analizamos las principales características de un subconjunto representativo de generadores de carga. Los generados han sido seleccionados teniendo en cuenta tres criterios principales: i) su representatividad tanto en el mundo comercial como académico, ii) su naturaleza, por ejemplo, dirigidos por traza o modelos matemáticos y, iii) los principales objetivos de su diseño, por ejemplo, dirigidos a comparar prestaciones de servidores, ajuste de servidores proxy, etc.

En la Tabla 1 se resumen las principales características de cada generador. *Arquitectura distribuida* hace referencia a la capacidad del generador para distribuir los procesos de generación de carga entre los diferentes nodos, emulando usuarios trabajando en máquinas distintas. *Arquitectura matemática* se refiere a la capacidad de utilizar modelos matemáticos. *Arquitectura orientada a negocios* indica la capacidad para modelar la estructura de la lógica de negocios de la aplicación web. *Parametrización de clientes* considera la habilidad para parametrizar el comportamiento de los usuarios. Algunos generadores organizan la carga en categorías o *Tipos de carga*, donde cada uno de ellos modela un perfil de usuario dado. Otros generadores definen los tests funcionales para modelar la lógica de negocios o de aplicaciones *Test de negocios*. Las simulaciones generalmente se ejecutan en entornos LAN y la mayoría de los simuladores actuales no pueden modelar las diferencias entre estas redes y las de área amplia (WAN) LAN y WAN, donde las aplicaciones normalmente residen. La principal característica en la que estamos interesados es un superconjunto de las características previas, ya

Característica/ Capacidad	Webload	Webstone	SPECweb99	TPC-W	S-clients	SURGE	JMeter	Polygraph	Webjamma	Deluge	Mercury	Hammer	PFester	Siege	Httpperf	Autobench
Arq. distribuida	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
Arq. matemática	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
Arq. negocios	▲	◆	◆	◆	◆	◆	▲	▲	◆	◆	▲	◆	◆	◆	◆	◆
Parametr. clientes	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	▲	◆	◆	◆	◆
Tipos de carga	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
Test de negocios	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
LAN y WAN	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
Multiplat.	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	▲	◆	▲	◆	◆
Comp. dinámico	▲	◆	◆	◆	◆	◆	▲	◆	◆	◆	▲	◆	◆	◆	◆	◆
Facilidad de uso	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
Informes	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆
Código abierto	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆	◆

◆ 100% ▲ Parcialmente ◇ 0%

Tabla 1: Características de generadores de carga actuales

que incluye la habilidad para modelar el comportamiento dinámico de usuarios como sus cambios continuos. En la Tabla 1 también pueden verse otras características como la *Facilidad de uso*, la capacidad de ofrecer Informes de prestaciones, y si el generador ofrece *Código Abierto*.

A continuación se identifican las principales características de los simuladores atendiendo a cuatro categorías:

### 1. Generadores desarrollados para la investigación académica.

Un importante grupo de generadores que incorporan una importante componente matemática se propuso con el objetivo de modelar la carga asociada al tráfico en internet. Por ej., SURGE [16], cuyo principal objetivo era medir las prestaciones del servidor variando su carga, o Web Polygraph [29], que simula clientes y servidores tanto a través de modelos matemáticos como componentes reales. Una característica de los generadores de esta categoría es su capacidad de experimentar nuevas arquitecturas en la generación de carga, por ej., el S-Clients [14], que divide el proceso de generación de clientes HTTP en dos subprocesos: uno para establecer la conexión y otro para obtener el contenido. En este grupo encontramos también el Webjamma [13].

### 2. Generadores comerciales actuales.

Los generadores mencionados en este punto son los líderes del mercado en este momento. Como productos comerciales que son, destacan por su facilidad de uso, capacidad de generar informes, parametrización de clientes y arquitectura orientada a negocios, como por ej., WEBLOAD [7, 33, 8, 9, 10], comercializado por Rad-View, JMeter, de Apache Project [3] y MERCURY Load Runner [4]. Estas tres soluciones son las únicas que incluyen aproximaciones para modelar el comportamiento dinámico de los usuarios.

### 3. Generadores comerciales históricos.

Se trata de generadores que por sus características han influido en las tendencias actuales, por ej., WEBSTONE [5], SPECweb99 [12] y TPC-W [32]. Los tres fueron referentes en su momento; por ej., WEBSTONE basa su operación en unidades de ejecución de clientes distribuidos en varias máquinas; SPECweb99 distribuye la carga entre varios clientes gestionados a través de un cliente central. Por su parte TPC-W está orientado hacia transacciones de comercio electrónico proveyendo modelos business-client (B2C) y business-business (B2B).

4. **Herramientas y scripts.** Muchas herramientas se han desarrollado para llevar a cabo tests de stress. En general, son productos de código abierto y amplia utilización; por ej., DELUGE [1], HAMMERHEAD [2], PTESTER [6], SIEGE [11], HTTPERF [24] or AUTOBENCH [23].

## GUERNICA

Las limitaciones de los generadores mencionados nos motivaron a diseñar e implementar un nuevo generador de carga con un doble objetivo. Primero, incluir las principales capacidades de los simuladores actuales y, segundo, añadir la capacidad de modelar adecuadamente el comportamiento dinámico de los usuarios. GUERNICA (Universal Generator of Dynamic Workload under WWW Platforms) [25, 27, 26] es el resultado de la colaboración entre el grupo de investigación de Arquitectura y Prestaciones de la Web de la Universidad Politécnica de Valencia y la empresa Intelligent Software Components (iSOCCO).

### Navegación y carga de prueba

El concepto de *Navegación*, lo introdujimos en [25, 27] y es la principal característica en el diseño de GUERNICA. De manera informal se confirma que la navegación define el comportamiento del usuario mientras interactúa con la web. Formalmente se define la navegación  $N$  como una secuencia  $S$  de  $n$  URLs de peticiones HTTP, donde:

$$S = \{url_1, url_2, \dots, url_n\} \forall_i = 2..n : url_i \text{ depende del contenido de } url_k \text{ para } k < i$$

Este concepto ha sido posteriormente ampliado con el de *carga de prueba* [26] de manera que nos permite modelar los continuos cambios en el papel representado por el usuario. Informalmente podemos definir este último como un conjunto de navegaciones que pueden ser ejecutadas en paralelo por uno o más hilos de ejecución (representando diferentes usuarios) durante el proceso de simulación del comportamiento del usuario. Se ha establecido una relación entre sus navegaciones de manera que se elige la siguiente navegación en función de las previas. Formalmente definimos la carga de prueba  $T(C, \varphi: C \times C \rightarrow N, I)$  donde  $C = \{n_1, n_2, \dots, n_k\}$  con  $n_i \in N$ .  $C$  es el conjunto de navegaciones.  $\varphi: C \times C \rightarrow N$ , es la relación que escoge la siguiente navegación a ejecutar, e  $I$  es el número de usuarios que se pueden simular en paralelo. La implementación de la carga de prueba está basada en un autómata, cuyos nodos representan navegaciones y sus arcos ponderados indican las transiciones entre las distintas navegaciones y la probabilidad asociada a cada una de ellas. Las navegaciones modelan el comportamiento dinámico de los usuarios mientras que las transiciones modelan los continuos cambios de su comportamiento.

### Características GUERNICA

En su actual versión, las características y capacidades de GUERNICA son las siguientes:

- Utilizando archivos XML define las navegaciones y sus parámetros.
- Utilizando archivos XML define las cargas de prueba con sus parámetros.

- Genera estadísticas a partir de la ejecución de las cargas de prueba, que también se pueden tener formato XML.
- Modela el comportamiento dinámico de los usuarios. Por lo tanto, cada navegación evoluciona hacia otra secuencia de peticiones HTTP en función de navegaciones previas, tal y como lo realiza un usuario real. En la actual versión el comportamiento dinámico solo depende del contenido de peticiones previas y no de su meta-información.
- Representa el tiempo de pensar del usuario, es decir, cuando el contenido de una petición HTTP llega, el generador incluye un retardo que puede ser tanto una constante como estar basado en una distribución Gaussiana.
- Ejecuta navegaciones concurrentemente utilizando ejecución multihilo.

Finalmente la actual versión de GUERNICA incluye seis de las once características previamente escritas. De manera parcial, incluye las características restantes, estando prevista su inclusión total en futuras versiones. La Tabla 2 resume estas características.

Característica/Capacidad	GUERNICA
Arquitectura distribuida	◆
Arquitectura matemática	▲
Arquitectura negocios	▲
Parametrización de clientes	◆
Tipos de carga	◆
Test de negocios	▲
LAN y WAN	▲
Multiplat.	◆
Comportamiento dinámico	◆
Facilidad de uso	◆
Informes	◆
Código abierto	▲

◆ 100%   ▲ Parcialmente

Tabla 2: Características del generador GUERNICA

### Caso de carga dinámica

En esta sección analizamos una carga de prueba que un usuario podría generar cuando navega, tanto con un comportamiento de tipo *surfer* o *buscador*. Los buscadores son los usuarios que comienzan sus navegaciones a través de una cuestión en un robot buscador como por ejemplo Google. Los surfers son aquellos usuarios que prefieren navegar utilizando sus propios hipervínculos directamente [28].

La Figura 1 representa un autómata que refleja estas dos clases de comportamiento, y como un usuario puede evolucionar de un comportamiento a otro. Inicialmente, el usuario adopta un comportamiento de tipo buscador o surfer, con una probabilidad del 75% y el 25% respectivamente. En el caso de los usuarios buscadores, tras ejecutar su navegación existe una probabilidad del 50% de que hayan encontrado el contenido deseado (acabando el proceso) o no (cambiando su comportamiento a tipo surfer). Un usuario de tipo surfer, tras ejecutar su navegación, encuentra el contenido deseado con una probabilidad del 80% (acabando el proceso) y con un 20% de probabilidad accede a un contenido con el que no está totalmente satisfecho (cambiando su comportamiento a tipo buscador).

Para explicar estos dos comportamientos utilizamos un ejemplo de navegación en el que se busca información relativa a las Jornadas Científico-Técnicas en Servicios Web 2005, celebradas en Granada. La Figura 1 presenta un autómata que muestra como un usuario busca la citada información mediante: i) búsquedas de la frase *jsweb granada* en un buscador, e.g. Google: *google\_jsweb*, o ii) navegando a través del site del Primer Congreso Español de Informática (CEDI): *cedi\_jsweb*. Estas dos navegaciones se ejecutan asociadas a los comportamientos *buscador* y *surfer* respectivamente. El listado 1 muestra el fichero XML en el que se define el caso de test asociado al autómata.

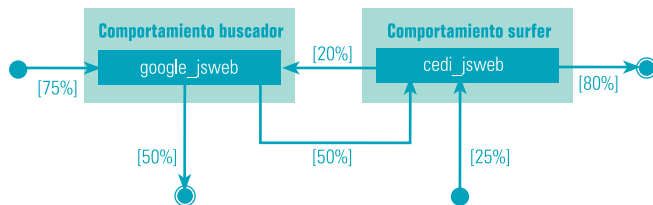


Figura 1: Cambios en el comportamiento del usuario: de buscador surfer y viceversa

Listing 1: Caso de test para la búsqueda de información sobre JSWEB 2005

```
<?xml version="1.0" encoding="UTF-8"?>
<WorkloadTest id="test_searcher_vs_surfer">
  <UsersNumber>1</UsersNumber>
  <NavigationGraph>
    <InitialNavigations>
      <InitialNavigation
        id="google_jsweb" probability="0.75"/>
      <InitialNavigation
        id="cedi_jsweb" probability="0.25"/>
    </InitialNavigations>
    <NavigationTransitions>
      <NavigationTransition
        from="google_jsweb" to="cedi_jsweb"
        probability="0.50"/>
      <NavigationTransition
        from="cedi_jsweb" to="google_jsweb"
        probability="0.20"/>
    </NavigationTransitions>
  </NavigationGraph>
</WorkloadTest>
```

La figura 2 muestra el grafo de navegación asociado al comportamiento buscador. En ella, podemos ver que lo primero que se hace es acceder a la página principal de Google (i.e., *www.google.es*), posteriormente el usuario espera un tiempo representado por una distribución Gaussiana de media 3500ms y desviación 1500ms (tiempo de pensar del usuario). Después, el usuario tiene dos opciones representadas por dos ramas en el grafo; si el buscador proporciona resultados para la frase *granada jsweb* (rama de la izquierda) el usuario accede al site de las jornadas (primer site devuelto); en otro caso, el usuario finaliza el proceso (rama de la derecha). Si el usuario accede al site de las jornadas, un nuevo tiempo de reflexión de 5000ms, en este caso dado por una distribución constante, aparece antes de finalizar la navegación (punto negro). Se ha usado un lenguaje de modelado basado en tags XML para definir las navegaciones y los casos de test. El listado 2 se corresponde al fichero XML que define la navegación de la figura 2.

Listing 2: Navegación basada en búsqueda Google de JSWEB

```
<?xml version="1.0" encoding="UTF-8"?>
<Navigation id="google_jsweb">
  <InputData>
    <Param name="phrase" value="granada jsweb"/>
  </InputData>
  <ExecutionCode>
    <PCA-Plugin name="google.pca.xml"/>
  </ExecutionCode>
  <StatisticsConfiguration>
```

```
<StatisticAttribute name="NavigationTime"/>
<StatisticAttribute name="ExecutionTime"/>
<StatisticAttribute name="HttpRoute">
  <StatisticAttribute name="URL"/>
  <StatisticAttribute name="HttpMethod"/>
  <StatisticAttribute name="Stablished"/>
  <StatisticAttribute name="StablishmentTime"/>
  <StatisticAttribute name="TransferTime"/>
  <StatisticAttribute name="ThinkUserTime"/>
  <StatisticAttribute name="ContentSize"/>
</StatisticAttribute>
</StatisticsConfiguration>
</Navigation>
```

En el listado se define una navegación en la que las palabras claves *granada jsweb* (phrase) son parámetros de entrada para buscarse en Google. Un conjunto de estadísticas es obtenido; por ej., el tiempo total de ejecución de GUERNICA (ExecutionTime), el tiempo total de navegación, y para cada petición HTTP el método GET/POST (HttpMethod), el tiempo de conexión (StablishmentTime), el tiempo de transferencia, el tiempo de reflexión, el tamaño del contenido, la URL accedida, y el éxito a la hora de establecer la conexión (Stablished). PCA-Plugin es el fichero XML que define, usando el lenguaje de scripting PCA, el comportamiento del usuario. Más detalles sobre el motor de navegación pueden encontrarse en [25, 27].

Listing 3: Resultados del Caso de test para la búsqueda de información sobre JSWEB 2005

```
<?xml version="1.0" encoding="UTF-8"?>
<WorkloadTestStatistics
  id="test_searcher_vs_surfer-generator-1"
  testId="test_searcher_vs_surfer">
  <UserNavigations>
    <NavigationStatistic navigationId="google_jsweb">
      <NavigationTime>20807</NavigationTime>
      <ExecutionTime>21421</ExecutionTime>
      <HttpRoute>
        <HttpRouteElement>
          <URL>http://www.google.es</URL>
          <HttpMethod>GET</HttpMethod>
          <Stablished>true</Stablished>
          <StablishmentTime>141</StablishmentTime>
          ...
        </HttpRouteElement>
        ...
      </HttpRouteElement>
      <NavigationStatistic navigationId="cedi_jsweb">
        <NavigationTime>14813</NavigationTime>
        <ExecutionTime>14828</ExecutionTime>
        <HttpRoute>
          <HttpRouteElement>
            <URL>http://www.w3c.es/Eventos/
              ServiciosWeb/</URL>
            ...
            <TransferTime>456</TransferTime>
            <ThinkUserTime>5000</ThinkUserTime>
            <ContentSize>11934</ContentSize>
          </HttpRouteElement>
        </HttpRoute>
      </NavigationStatistic>
    </UserNavigations>
  </WorkloadTestStatistics>
```

Una vez que se ejecuta el caso de test, se obtiene la traza de las navegaciones realizadas y sus peticiones HTTP, cada una con resultados diferentes. El listado 3 muestra parte de una salida para el ejemplo ilustrado.

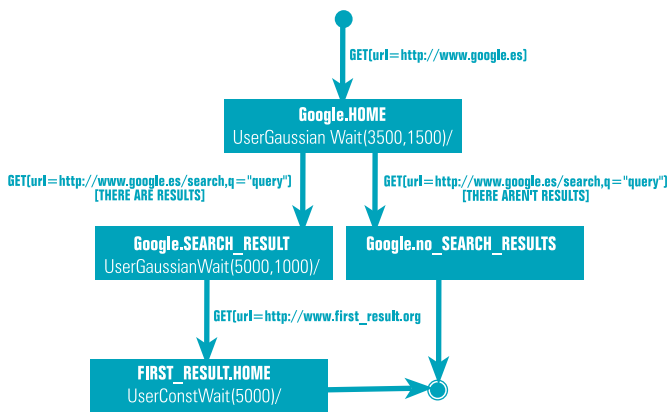


Figura 2: Search in Google navigation

## Conclusiones y Trabajo Futuro

La evolución de la segunda generación WWW ha introducido nuevas características de las aplicaciones y los usuarios han pasado a ser clientes en potencia. Esta nueva generación de la web presenta un elevada cantidad de contenido dinámico y personalizado que condiciona el comportamiento de la navegación de los usuarios. Por ello, es necesario disponer de nuevos modelos de carga capaces de representar este comportamiento.

**En este trabajo hemos presentado nuestra propuesta de generador, GUERNICA, basado en el concepto de navegación y carga de prueba, el cuál permite modelar el comportamiento dinámico de los usuarios y los continuos cambios en los patrones de navegación de éstos.**

En este trabajo hemos presentado nuestra propuesta de generador, GUERNICA, basado en el concepto de navegación y carga de prueba, el cuál permite modelar el comportamiento dinámico de los usuarios y los continuos cambios en los patrones de navegación de éstos. Además, la actual versión reduce el tiempo de ejecución experimental mediante el soporte de ejecución multihilo y permite simular distintos usuarios navegando al mismo tiempo. La navegación permite seleccionar caminos alternativos, emulando decisiones humanas. Más aún, permite seleccionar caminos dependiendo de: i) las condiciones del entorno real, por ejemplo, el tiempo de transferencia o el de establecimiento de la conexión, ii) las características del objeto, por ejemplo, su tamaño, y iii) las características del usuario, por ejemplo, su tiempo de pensar. Nuestro generador también incorpora distribuciones estadísticas que pueden parametrizarse para ajustarse al comportamiento del usuario; por ejemplo, distribución Gaussiana o constante; además, ofrece tanto estadísticas como resultados gráficos. En este trabajo también se ha realizado un análisis exhaustivo de las principales características, capacidades y limitaciones de los generadores de carga más representativos del momento.

## Agradecimientos

Los autores quieren agradecer a Emilio Gil, Javier Sevilla, Tomás Couso, Francisco J. García, José Miguel Pardo, Salvador Martín, Carlos Pardo y José Antonio Cruz de iSOCO, Ester Palacios y Marga Nácher del Instituto Tecnológico de Informática, Bernardo de la Ossa e Ingrid J. Niño de la Universidad Politécnica de Valencia, su valiosa ayuda y soporte en el Proyecto GUERNICA. Este trabajo ha sido parcialmente financiado por el Gobierno Español (CICYT TIC2001-1374-C03-02 y FIT-340000-2004-236), por el IMPIVA (IMIDTD/2004/92, IMIDTD/2005/15) y por el Ministerio de Educación y Ciencia y el FEDER (TSI 2005-07876-C03-01).

### Referencias Bibliográficas

- [1] Deluge. <http://deluge.sourceforge.net>.
- [2] Hammerhead 2. <http://hammerhead.sourceforge.net>.
- [3] Jakarta apache.jmeter. <http://jakarta.apache.org/jmeter/index.html>.
- [4] Mercury loadrunner. <http://www.mercury.com/>.
- [5] Mindcraft. webstone. <http://www.mindcraft.com/webstone>.
- [6] Ptester. <http://freshmeat.net/projects/ptester>.
- [7] Radview software. webload. <http://www.radview.com>.
- [8] Recording WebLOAD 5.0 Load Testing. Quick start guide.
- [9] Recording WebLOAD 5.0 Programming Guide. v 5.0.
- [10] Recording WebLOAD Agendas. v 5.0.
- [11] Siege, <http://www.joedog.org/siege/index.php>.
- [11] Siege, <http://www.joedog.org/siege/index.php>.
- [12] Specweb99. <http://www.specbench.org/osg/web99>.
- [13] Virginia tech's network research group. webjamma. <http://www.cs.vt.edu/chitra/tools.html>.
- [14] Gaurav Banga and Peter Druschel. Measuring the capacity of a web server under realistic loads. World Wide Web, 1999.
- [15] Paul Barford, Azer Bestavros, Adam Bradley, and Mark Crovella. Changes in web client access patterns: Characteristics and caching implications. Technical report, Boston University, 1998.
- [16] Paul Barford and Mark Crovella. An architecture for a www workload generator. In World Wide Web Consortium Workshop on Workload Characterization, 1997.
- [17] Paul Barford and Mark Crovella. Generating representative web workloads for network and server performance evaluation. In ACM SIGMETRICS, 1998.
- [18] Lara D. Catledge and James E. Pitkow. Characterizing browsing strategies in the world-wide web. In 3rd Inter. WWW conference on Tech., tools and applications, 1995.
- [19] Sally Floyd and Vern Paxson. Difficulties in simulating the internet. IEEE/ACM Transactions on Networking, 2001.
- [20] Enrique Frías-Martínez and Vijay Karamcheti. A prediction model for user access sequences. In 4th WEBKDD, 2002.
- [21] Abdulla Ghaleb. Analysis and Modeling of World Wide Web Traffic. PhD thesis, 1998.
- [22] Juan Antonio Lacort, Ana Pont, José Antonio Gil, and Julio Sahuquillo. A comprehensive web workload characterization. In 2nd Inter. HET-NET, 2004.
- [23] Julian T.J Midgley. <http://www.xenoclast.org/autobench>.
- [24] David Mosberger and Tai Jin. Httpref. <http://www.hpl.hp.com>.
- [25] Raúl Peña-Ortiz, Julio Sahuquillo, Ana Pont, and José Antonio Gil. Modeling users' dynamic behavior in web application environments. In 2nd Inter. HET-NET, 2004.
- [26] Raúl Peña-Ortiz, Julio Sahuquillo, Ana Pont, and José Antonio Gil. Modeling continuous changes of the user's web dynamic behavior in the WWW. In 5th WOSP, 2005.
- [27] Raúl Peña-Ortiz, Julio Sahuquillo, Ana Pont, and José Antonio Gil. Modeling users' dynamic behavior in e-business environments using navigations. IJEB, 2005.
- [28] Peter Pirolli and James E. Pitkow. Distributions of surfers' paths through the world wide web: Empirical characterizations. World Wide Web, 1999.
- [29] Alex Rousskov, Duane Wessels, and Glenn Chisholm. The \_rst ircache web cache bake-off. Technical report, National Laboratory for Applied Network Research, 1999.
- [30] Weisong Shi, Randy Wright, Eli Collins, and Vijay Karamcheti. Workload characterization of a personalized web site and its implications for dynamic content caching. In 7th Inter. WCW, 2002.
- [31] Weisong Shi, Randy Wright, Eli Collins, and Vijay Karamcheti. Modeling object characteristics of dynamic web content. Journal of Parallel and Distributed Computing, 2003.
- [32] Wayne D. Smith. Tpc-w: Benchmarking an ecommerce solution. Technical report, Intel Corporation, 2000.
- [33] RadView Software. Webload 6.0 the webload difference. Technical report, 2003.