

UN ESTUDIO SOBRE RENDIMIENTO WEB

Nombre del Autor

Afiliación

Dirección

E-mail

Nombre del Autor

Afiliación

Dirección

E-mail

Nombre del Autor

Afiliación

Dirección

E-mail

Nombre del Autor

Afiliación

Dirección

E-mail

RESUMEN

Pasado el tiempo en el que la presencia en Internet suponía una ventaja competitiva para una organización, en la actualidad la preocupación de las organizaciones se centra en competir a través de la creación de sitios Web de calidad. El usuario, por su parte, considera como una de los principales motivos de elección de un sitio Web el valor que debe pagar, tanto en tiempo como en coste de la conexión. Esto hace plantearse a las organizaciones el rendimiento de sus sitios Web como una cuestión fundamental para evitar la “fuga” de sus usuarios.

En los últimos tiempos se han desarrollado diferentes trabajos sobre el rendimiento Web según distintos puntos de vista. En este artículo se van a resumir algunos de estos trabajos para dar al lector una visión amplia sobre los puntos que se han tratado o se siguen tratando en la actualidad. Sin embargo, aunque existen muchas recomendaciones acerca de lo que un desarrollador puede controlar para evitar problemas de rendimiento en fases tempranas del proyecto, éstos se basan la mayoría de las veces en la experiencia y no aportan datos objetivos y extrapolables a otros entornos.

En este trabajo se tratará la necesidad de trabajar en los aspectos de la configuración de aplicaciones, con el fin de proporcionar pautas a seguir por los gestores de las aplicaciones.

PALABRAS CLAVES

Rendimiento, Web, Aplicaciones

1. INTRODUCCIÓN

El crecimiento del software de aplicaciones ha impulsado el aumento de la importancia que se le otorga a la calidad del software. Esta importancia se hace aún más patente cuando lo aplicamos al software Web teniendo en cuenta que un rendimiento insuficiente en un sitio Web es la principal causa de abandono de éste por otro que ofrezca una mayor velocidad de acceso. En un estudio dirigido por la Georgia Teach University en 1997, más del 80% de los encuestados encontró la velocidad como el principal problema de Internet [Wilson, 1997].

Según el estándar IEEE Std 1061-1998 [IEEE Std 1061], se define calidad del software como el grado en el que el software posee una combinación deseada de características previamente definidas. No podemos

saber si la calidad es satisfactoria si no podemos medirla. Para poder cuantificar la calidad software se definen métricas. Según las características especificadas, así como, las subcaracterísticas en las que las mismas se dividan, surgen diferentes modelos de calidad. Los modelos de McCall[Fenton, 1999] y Boehm[Boehm et al, 1978] describen la calidad en un enfoque de descomposición descendente. En el modelo de MacCall se definen una serie de atributos o características llamados *factores* que se clasifican según su uso(operación, revisión o transición). Son factores, por ejemplo, la usabilidad, la integridad o la eficiencia. Cada factor está compuesto de *criterios* de nivel inferior que son más fáciles de entender y de medir que los factores. Por ejemplo, son criterios del factor eficiencia la *eficiencia de almacenamiento* y la *eficiencia de ejecución*". Esta noción de división de factores en criterios ha sido implementada en el estándar ISO 9126[ISO/IEC 9126]. El modelo de Boehm es similar al de McCall aunque reduce el número de atributos de once, que se utilizan en el modelo de McCall, a siete. Otros modelos para representar características, subcaracterísticas, así como, el modelo de descomposición subyacente son los estándares IEEE 1061[IEEE Std 1061] e ISO 9126[ISO/IEC 9126]. Ambos trabajos coinciden significativamente en su contenido. En lo que respecta al estándar ISO 9126, describe la calidad software a partir de 6 características generales: funcionalidad, disponibilidad, usabilidad, eficiencia, facilidad de mantenimiento y portabilidad. Dentro del marco conceptual de la ISO la eficiencia, en concreto, se define como el conjunto de cualidades que influye en la relación entre el nivel de rendimiento del software y la cantidad de recursos usados bajo condiciones establecidas donde los recursos son otro software, hardware o bien servicios(operación, mantenimiento, etc.).Por tanto, podemos distinguir dos partes bien diferenciadas o "subcaracterísticas": el comportamiento en relación al tiempo y el comportamiento en relación a los recursos. Otro modelo de calidad más reciente y que nos interesa especialmente por estar centrado en la calidad Web, es la denominada Metodología de evaluación de calidad de sitios Web (*Web-site Quality Evaluation Method*, Web-site QEM)[Olsina et al, 2002] que aporta una visión de calidad de un sitio Web según las necesidades de cierto perfil de usuario y aporta un enfoque cuantitativo completo para evaluar sitios Web que incluye modelos, métodos, procedimientos, criterios y herramientas.

El objetivo de este trabajo será ofrecer soluciones a quienes desarrollan aplicaciones dentro de los factores que éstos puedan controlar. A lo largo del trabajo, veremos primero un resumen de los trabajos existentes en este campo en la sección 2, a continuación en la sección 3 nos centraremos en las soluciones que hay en la actualidad para el desarrollo de aplicaciones y, por último, se darán unas conclusiones acerca de la necesidad de buscar soluciones para la configuración de aplicaciones Web.

2. EL PROCESO DE OBTENCIÓN DE UNA PÁGINA WEB

Un indicador útil para medir el rendimiento de un sitio web es el tiempo total transcurrido para la obtención de una página web también denominado latencia.

Podemos separar el tiempo de obtención de una página web en diferentes componentes de forma que podamos aislar cuellos de botellas y planificar los cambios apropiados.

En la figura 1 se puede apreciar una línea del tiempo con las diferentes fases por las que pasa la petición de una página web hasta su obtención[Loosley, 2000].

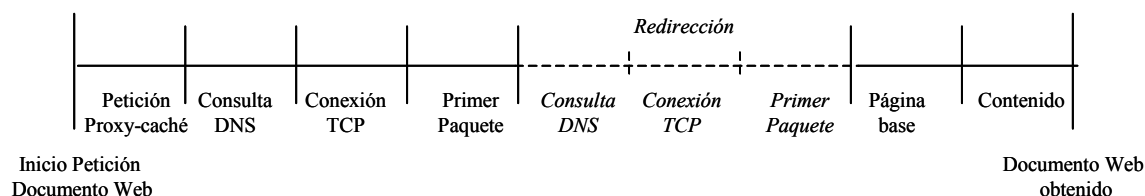


Figura 1

Tal como indica la Figura 1, en un primer paso el cliente que inicia la comunicación se pone en contacto con su servidor proxy-caché, en caso de que la arquitectura de la red que le provee de conexión a Internet así lo requiera, para que sea éste quien le proporcione el documento requerido, ya sea porque lo tiene almacenado en su caché como resultado de una petición del mismo documento, o ya sea conectándose por él

mismo al servidor proveedor del contenido. En un segundo paso, bien el servidor proxy-caché o bien el cliente en caso de disponer de salida a Internet sin necesidad de proxy-caché, se conectará al servidor DNS local para que éste resuelva el nombre de servidor indicado en la petición HTTP a su dirección IP. El tiempo de consulta DNS es el tiempo requerido para que el servidor DNS local resuelva el nombre de host que de no tenerlo almacenado en caché deberá resolver por medio de peticiones iterativas comenzando por el servidor DNS de más alto nivel en la jerarquía o servidor raíz. Una vez obtenida la dirección IP se iniciará la conexión TCP con el servidor de destino a través de un proceso conocido como “saludo a tres vías” en el que el emisor propone al receptor el inicio de una conexión enviándole su número de secuencia inicial que identificará su conexión de forma unívoca, cuando el receptor recibe el intento de conexión, en caso de estar en condiciones de aceptar la conexión, responderá al emisor con un acuse de recibo y su número de secuencia inicial y el emisor al recibirlo acusará recibo del mensaje y enviará la petición HTTP considerándose iniciada la comunicación a partir de ese momento. El tiempo de conexión TCP mide el tiempo que tarda en completarse el saludo a tres vías siendo, además, un excelente indicador del tiempo de ida y vuelta de un paquete(RTT) al realizarse en el nivel de transporte y al ser paquetes relativamente pequeños puesto que no llevan datos. El tiempo de envío del primer paquete es el tiempo que el cliente espera entre el HTTP GET request y la recepción del primer paquete enviado por el servidor web de destino. El motivo de separar este primer paquete del resto es que éste es un buen indicador del retardo causado por el servidor web o tiempo que el servidor tarda en procesar la petición consultando los servidores necesarios para ello: servidores de aplicación web, bases de datos, etc. En una siguiente etapa el tiempo de redirección mide el tiempo empleado cuando el primer paquete recibido indica una redirección a otro sitio Web(por ejemplo un mensaje HTTP 302). Si dicha redirección apunta a un servidor diferente, será necesario de nuevo realizar una consulta DNS y una conexión TCP. Puede que haya más de una redirección antes de contactar con el servidor web que devolverá el documento web. El tiempo de obtención de la página base contempla el tiempo transcurrido hasta que el cliente obtiene el código HTML. Por último, el tiempo de obtención del contenido identifica el tiempo requerido para obtener los objetos embebidos en el documento Web tales como imágenes o applets).

A continuación se citan los trabajos existentes hasta el momento en relación al rendimiento de cada una de las fases mencionadas.

2.1 Uso de servidor Proxy-caché

Una de las técnicas más extendidas para mejorar el rendimiento de Internet es el uso de servidores proxy-caché que actúan como una caché de documentos Web para un conjunto de clientes HTTP. En estos casos, el servidor proxy-caché actúa como intermediario solicitando él mismo la petición al servidor Web y recibiendo el resultado de ésta para reenviarlo más tarde al cliente que lo solicitó. Cuando un cliente envía una petición, el servidor proxy-caché comprueba si tiene el documento almacenado en su caché y, si es así, si es una copia “fresca” del documento, en caso afirmativo, devuelve la copia que tiene almacenada del documento Web al cliente, en caso contrario, reenvía la petición al servidor Web encargado de proveer el contenido solicitado quien, una vez procesada la petición, devolverá el contenido al servidor proxy quien, además de reenviar el documento al cliente, lo almacena en su caché para subsiguientes peticiones del mismo documento.

El uso de proxy-caché reduce la latencia percibida por el usuario a la hora de obtener un documento Web, además de reducir el tráfico en la red y la carga en los servidores del proveedor de contenidos. Al usar servidor proxy-caché la latencia depende de si la conexión se hace a través de módem o de LAN consiguiendo una ventaja mínima por el uso de proxy-caché en el caso de conexiones a través de módem[Almeida, 1998]. De hecho, en [Feldmann, 1999] se muestra una mejora de la latencia de un 8% en el caso de conexiones por módem frente a un 38% en clientes conectados a través de una LAN donde el ancho de banda es superior.

Se han realizado diversos estudios para mejorar el rendimiento en servidores proxy-caché, en la mayoría de los casos dirigidos a mejorar el hit ratio o número de objetos almacenados en caché con respecto al total de objetos solicitados. El hit ratio depende del número de peticiones realizadas y del tamaño de la caché. Un alto hit ratio se traduce en ahorro de ancho de banda para proveedores de Internet. En [Almeida, 1998] se presenta una arquitectura de servidores proxy distribuida que incrementa la disponibilidad del servidor, proporciona escalabilidad y balanceo de carga. Otros estudios se han dirigido hacia la cooperación entre servidores proxy[Wolman et al, 1999, Kubiawicz et al, 2000]. En [Wolman et al, 1999] se demuestran los beneficios de proxy-caché en pequeñas organizaciones. Los estudios más actuales, en lo que a cooperación se

refiere, se dirigen hacia la cooperación de servidores proxy-caché basándose en redes peer-to-peer. Algunos ejemplos son Oceanstore[Kubiatowicz et al, 2000, Eaton, 2002], Pastry[Rowstron1 et al, 2001] o Squirrel[Lyer et al, 2002] entre otros.

2.2 Tiempo de consulta DNS.

El primer paso para la obtención de un documento Web consiste en una consulta al servidor DNS con el fin de obtener la dirección IP asociada al nombre de host requerido.

DNS es un componente crítico en la operación de las aplicaciones en Internet. Según el tráfico recogido por los enlaces de Internet del MIT Laboratory for Computer Science y el Korea Advanced Institute of Science and Technology[Jung et al, 2002] el 23% de las consultas DNS en Internet no recibe respuesta, dado que las consultas son retransmitidos durante un tiempo predefinido, dichas consultas implican que más del 50% del tráfico de Internet consista en paquetes DNS.

Existen trabajos sobre la optimización del valor TTL, que dicta el tiempo que un elemento, una vez devuelto, permanece válido en caché, así como sobre, la influencia del tamaño de la caché en las medidas [Jung et al, 2002, Jung et al, 2003].

2.3 Tiempo de conexión TCP

Para la transferencia de documentos web en Internet se utiliza el protocolo de nivel de Aplicación HTTP, HyperText Transport Protocol[Berners-Lee et al, 1995] sobre el protocolo de transporte confiable de Internet TCP[Postel, 1981]. TCP debido a su naturaleza de protocolo orientado a conexión y a sus funcionalidades de control de congestión y confiabilidad a través de la retransmisión de paquetes no confirmados perjudica de forma notable al tiempo de bajada de una página Web.

Dado que TCP es un protocolo confiable, espera confirmaciones de recepción de los mensajes enviados. Para ello, usa un temporizador de retransmisión según el cuál, pasado un tiempo preestablecido sin recibir el acuse de recibo de un mensaje, éste es retransmitido. Es importante que el valor del temporizador sea adecuado, ya que, valores altos de éste pueden retrasar la transmisión de los datos al esperar más tiempo del necesario para reenviar un paquete perdido, mientras que valores bajos pueden provocar retransmisiones innecesarias sobrecargando la red y generando congestión en ella.

El establecimiento de conexión en TCP se basa en un procedimiento conocido como “saludo a tres vías”, en el cuál, emisor y receptor deben enviarse un total de 3 paquetes para considerar establecida la conexión. Por otra parte, para obtener una página completa es necesario obtener el contenido HTML de la página en una conexión/desconexión por saludo a tres vías y después realizar otra conexión/desconexión a tres vías por cada uno de los objetos embebidos en la página Web. Para mejorar la eficiencia de HTTP sobre TCP, se utilizan las conexiones persistentes, en las cuáles, la conexión TCP se mantiene activa hasta que la página Web completa se ha transmitido siempre que los objetos del documento se encuentren todos en el mismo servidor, así como, la página base. Las conexiones persistentes forman parte de la especificación de HTTP/1.1[Fielding et. Al 1997], por tanto, debemos asegurarnos que nuestro servidor Web ejecuta HTTP/1.1, aunque de nada sirve si el cliente que realiza la petición lo hace en HTTP/1.0, ya que en ese caso, será esta versión de protocolo y no la actual la que se utilice.

Otras mejoras de la eficiencia de HTTP sobre TCP son el caché de conexiones y el uso de tuberías. El uso de conexiones persistentes evita tener que realizar múltiples peticiones HTTP para bajar un página Web. En [Nielsen et al, 1997] se sugiere el uso de TCP transaccional, T/TCP, para el tráfico Web. T/TCP cachea la información de configuración de la conexión TCP, de forma que, subsiguientes conexiones TCP evitan la conexión por saludo a tres vías. Por otra parte, el uso de tuberías consiste en enviar todas las peticiones sin esperar respuesta[<http://www.weblogic.com/>].

2.4 Tiempo de obtención del primer paquete

Generalmente el primer paquete contiene un código de respuesta HTTP 200 en el que se identifica el servidor Web y las características del documento Web requerido. Por tanto, se puede equiparar al retardo del Servidor, es decir, al tiempo que el servidor necesita para procesar la petición del cliente, tanto si se trata de

localizar un documento Web estático como si consiste en generar un documento dinámico utilizando para ello servidores de aplicación, bases de datos, etc.

En el retardo del servidor Web influye tanto el diseño e implementación de la aplicación Web como la infraestructura y configuración de éste. Con el aumento de uso de páginas Web dinámicas, la mayoría de los sitios Web utilizan una arquitectura de tres capas, en la cuál, el servidor Web es el responsable de obtener la petición y enviar la respuesta al cliente, el servidor de aplicaciones realiza el procesamiento requerido por la lógica de negocio y el servidor de base de datos se encarga de almacenar y devolver los datos cuando el servidor de aplicaciones así lo requiere. Esta complejidad, aunque beneficiosa para el diseño de la aplicación, dificulta la gestión del rendimiento.

Los servidores Web tradicionalmente han dado poca información acerca del tiempo que les lleva realizar una petición. Por ejemplo, reportan el tiempo que les lleva manejar una petición en segundos y no en milisegundos. Sin embargo, actualmente existen herramientas que proporcionan información en tiempo real sobre el tiempo de respuesta de los servidores, la cantidad de recursos que está utilizando el servidor, etc. El mecanismo más popular para la gestión centralizada de datos referentes al rendimiento de servidores es el Simple Network Management Protocol(SNMP)[Harrington et al, 2002] . La mayoría de los proveedores de software incorporan compatibilidad con SNMP. Es el caso, por ejemplo, de los servidores Web de Apache, Microsoft Internet Information Server o IPlanet Enterprise Edition.

Con SNMP es posible comprobar si el servidor tiene suficiente pool de procesos o hilos para manejar su carga actual o comprobar el uso de memoria. Usando SNMP es posible regularmente obtener información acerca del estado del servidor. En la actualidad existen muchos productos que, basándose en SNMP, proporcionan información según las métricas seleccionadas acerca del rendimiento. En la actualidad dichas herramientas se basan en la optimización de la experiencia de un usuario final de la aplicación Web. Por tanto, además de monitorizar la infraestructura interna , cuando se trata de aplicaciones Web será necesario monitorizar el tiempo de respuesta que el usuario percibe. En este sentido, la monitorización del rendimiento se lleva a cabo a través de Internet. Diferentes compañías distribuyen puntos de monitorización por todo el mundo para cada cierto tiempo medir el tiempo que tarda desde cada uno de los puntos obtener cierta página o páginas Web previamente acordadas, o ciertas transacciones Web. Como en muchos casos las aplicaciones son alojadas en Centros de Procesos de Datos de proveedores, generalmente el mismo que provee del acceso a Internet, la organización responsable de la aplicación no tendrá el control sobre todos los elementos requeridos. Este tipo de aplicaciones pueden ayudar a introducir control a través de la formalización de acuerdos de nivel de servicio. De esta forma, las organizaciones podrán recibir informes de una organización neutral para utilizarlos en las revisiones de cumplimiento de niveles de servicio acordadas. Muchas compañías se han especializado en la monitorización de aplicaciones Web, tales como SiteAngel de BMC Software y Tivoli de IBM entre otras, muchas veces para completar sus ya comercializados productos de gestión de rendimiento.

Aparte de la monitorización de la aplicación en producción, existen artículos[Jagielski, 2002, Levitt, 1999] así como “guías” sobre configuraciones que pueden mejorar el rendimiento de nuestra aplicación según el servidor Web que estemos utilizando. Es el caso de Apache[Diao et al, 2003], <http://httpd.apache.org/> , Internet Information Server(<http://www.microsoft.com/>) o IPlanet(<http://docs.sun.com/>)

Por supuesto, no todo el retardo está relacionado con el servidor HTTP; éste podría estar esperando respuesta del servidor de aplicaciones. En la monitorización de servidores de aplicación es posible, aparte de utilizar SNMP al igual que en el servidor Web, introducir traps SNMP en el código que nos proporcione información útil de control en varios estadios del proceso de la aplicación.

Diferentes factores afectan al rendimiento del servidor de aplicaciones. En algunos casos el propio proveedor del servidor de aplicaciones proporciona herramientas para la monitorización del rendimiento del servidor. Es el caso de IBM WebSphere 5.0, por ejemplo, para el que existe una herramienta, Performance Advisors[Norton et al, 2003] que permite analizar el rendimiento del servidor de aplicaciones y proporcionar recomendaciones de configuración.

Pero además de influencias de elementos propios de la transmisión de información, existen factores de ruido de considerable magnitud que hacen que el rendimiento de los servidores pueda sufrir interferencias ajenas al de la propia aplicación. Un ejemplo destacado podría ser el tiempo de servicio del sistema de archivos, en el que interviene el sistema operativo del servidor y el propio hardware (velocidad y tamaño de la unidad de disco). En [Escribano et al., 2001] se contempla que el tiempo de respuesta de un documento HTML que incluya una imagen, es distinto si la imagen y el fichero HTML se encuentran en el mismo

directorio o no. Además, en esta diferencia influye significativamente la “distancia” entre ambos directorios. Por supuesto, el grado de fragmentación de un disco, tendrá una influencia sobre el rendimiento del mismo inversamente proporcional. Estos factores externos son –en términos relativos- más importantes cuanto más pequeños sean los archivos a servir, ya que suelen ser valores de tiempo constantes muy cercanos a procesos mecánicos (movimiento de cabezas lectoras de un disco, tiempo de latencia del mismo...) o a algoritmos de planificación del sistema operativo. Ahora bien, en procesos sencillos (como: servicio de documentos HTML, pequeños desarrollos de script de servidor, gestión de cookies...) estos pequeños documentos pueden llegar a representar una parte importante del total de ficheros de una aplicación web.

Los servidores de aplicación utilizan también muchos sockets para la comunicación entre los diferentes componentes tanto el servidor Web como el de base de datos pero también a otros servidores tales como servidores LDAP. En este sentido es más eficiente el uso de conexiones persistentes a tener que abrir y cerrar una conexión cada vez, pero la mejor solución es el uso de un pool de conexiones que son un conjunto predefinido de conexiones que se mantiene abierto durante todo el tiempo que el servidor está activo, de forma que, cuando se necesita una conexión no es necesario abrirla ya que puede utilizarse una de las conexiones abiertas del pool. Por otra parte, el uso de conexiones seguras(SSL) supone también un problema de rendimiento, por lo cuál, deben utilizarse sólo cuando sea necesario.

Otro punto a tener en cuenta es la necesidad de autenticación y autorización (A&A). Al tener varios servidores, no sólo es necesaria en su caso la autenticación y autorización para el usuario en el servidor Web sino también la de éste último cuando requiere acceso al servidor de aplicaciones o éste al servidor de base de datos. En este caso, el uso de servidores de autenticación centralizados, por ejemplo, servidores LDAP[Zeilenga 2002] Para conseguir un buen diseño y, sobretodo, un diseño seguro el servidor LDAP deberá estar protegido detrás de la DMZ y la comunicación entre los servidores Web o de aplicación con el servidor LDAP debería realizarse mediante SSL. Este diseño aunque deseable desde el punto de vista de la seguridad se traducirá en un aumento de la latencia.

Como hemos visto, el diseño de la infraestructura interna también es importante. Ya que diferentes servidores necesitan conectarse entre sí, es necesario que estas conexiones se realicen de forma eficiente. También es deseable el hecho de balancear la carga entre diferentes servidores tanto a nivel de servidor Web como de Aplicación. Añadir redundancia en cada nivel minimiza el impacto de fallos de componentes. Elegir un balanceador de carga que, además, proporcione tolerancia a fallos para dos o más servidores Web y seleccionar una aplicación que admita balanceo de carga, como es el caso de BEA Weblogic(<http://www.weblogic.com/>) pueden evitar paradas del sitio Web no planificadas. Otra medida a nivel de arquitectura que permite distribuir la carga es construir diferentes sitios Web, uno para cada línea de negocio, por ejemplo. Una configuración típica es el disponer de un sitio Web diferente para usuarios registrados y usuarios no registrados u ocasionales.

A nivel de configuración del servidor de aplicación, existen diversas recomendaciones de fabricante. Por ejemplo, en el caso de BEA Weblogic(<http://www.weblogic.com/>), se proporciona una guía con medidas para mejorar el rendimiento del servidor de aplicaciones, tanto a nivel de sistema operativo, hardware, etc, como a nivel de parámetros de configuración de la aplicación. También incorpora un software, Performance Pack, dependiente de la plataforma que mejora el rendimiento de la entrada/salida.

Pero, ¿qué configuraciones puede un desarrollador o un gestor de aplicaciones tener en cuenta para un mejor rendimiento del servidor Web o del servidor de Aplicaciones? Los proveedores de software hacen recomendaciones en sus manuales de documentación acerca de configuraciones óptimas como hemos visto, también existen guías de buenas prácticas pero éstas no basan en experimentos que las respalden.

Existen, sin embargo intentos de formalizar resultados al respecto. Un grupo de investigación de la Universidad de Carleton en California(<http://www.sce.carleton.ca>) introducen los patrones orientados a rendimiento(Performance-oriented patters, POPs). Los POPs son elementos de diseño estructural o de arquitectura que a menudo se repiten en los diseño software, de forma análoga a los patrones de diseño software , que no son más que soluciones parciales a ciertos problemas de diseño de forma que puedan ser reutilizados. El objetivo de los POPs es crear un repertorio de patrones útiles que podamos reconocer en cualquier diseño, de forma que, pueden guiar nuestro conocimiento de sus problemas de rendimiento.

Otro intento de relacionar la configuración del sistema con la latencia del documento Web son las mediciones que se comentaron anteriormente[Escribano et al., 2001] sobre la influencia del lugar en el que se encuentran el documento Web base y los objetos de éste.

2.5 Tiempo de redirección

Si el mensaje de respuesta del primer paquete es un HTTP 302 o si el HTML incluye un etiqueta de redirección el tiempo necesario para obtener un documento Web aumenta significativamente al tener que empezar de nuevo, es decir, realizar una nueva consulta DNS, el correspondiente inicio de conexión y obtener el primer paquete. Por tanto, deben evitarse las redirecciones en páginas Web en la medida de lo posible o, al menos, utilizarse como mecanismos temporales.

2.6 Tiempo de obtención de la página base

Una vez enviado el primer paquete, el servidor continua enviando paquetes con el HTML del documento. Este proceso se realiza en modo mono-tarea, es decir, con un único thread o hilo, por tanto, el número de bytes HTML tiene una influencia directa sobre el tiempo total de obtención de un documento Web.

2.7 Tiempo de obtención del contenido

Si observamos el camino que sigue una petición desde el cliente o proxy de su red al servidor una vez resuelta la consulta DNS, éste consiste en enlaces y routers, por tanto, otro de los aspectos a considerar son el medio físico de los enlaces y los protocolos de transmisión de datos utilizados que limitarán el máximo ancho de banda del enlace y el tiempo de ida y vuelta del paquete(RTT- Round Trip Time), así como, los enrutadores. Si un enrutador tiene un búfer o una velocidad de procesamiento insuficiente entonces descartará paquetes. La pérdida de paquetes aumenta significativamente el RTT.

En relación a la optimización de rendimiento de enrutadores existen muchos trabajos[Kuhns et al, 2002, Pappu et al, 2003, Baboescu et al 2003] que intentan aumentar la eficiencia de los algoritmos de enrutamiento utilizados minimizando el uso de memoria y procesador de los enrutadores, así como, el tráfico en la red.

En lo que al diseño de la página web se refiere existen listas de buenas prácticas para mejorar el rendimiento[Levitt, 1999, Zhi, 2001, Amerson et al, 2000] IBM utiliza una herramienta interna[Amerson et al, 2000] que proporcionan datos sobre el rendimiento de un sitio Web, además de comprobar si cumple un conjunto predefinido de buenas prácticas para optimizar el rendimiento. En [Zhi, 2001] se insiste en la importancia del número de paquetes más que del tamaño de página. Dado que en Internet toda la información viaja en paquetes, el tiempo de transferencia de dichos paquetes dependerá del número de paquetes, así como, del tamaño de los mismos. Por tanto, una página será más eficiente en su diseño cuanto más pequeño sea su tamaño y cuantas menos peticiones haya que realizar para obtenerla, lo que se traduce en un número pequeño de objetos embebidos por página. Como consecuencia, el tamaño de los documentos Web debe entenderse en términos del número de paquetes que requieren para su transmisión. Dado que el tamaño de paquete viene limitado en Internet por el MTU(Maximum Transfer Unit) que por defecto es 1500 bytes y dado que las cabeceras IP y TCP son 20 bytes cada una, nos encontramos con 1460 bytes disponibles para datos.

3. RENDIMIENTO WEB DESDE EL PUNTO DE VISTA DEL GESTOR DE APLICACIONES

Según lo visto en la sección 2, la inmensa mayoría de las soluciones para mejorar el rendimiento de las aplicaciones Web se basan en la premisa de un sistema en fase de Producción, a través de la monitorización, ya sea de la infraestructura interna de dichas aplicaciones o ya sea a través de Internet para obtener la percepción del usuario al realizar transacciones típicas previamente configuradas.

Sin embargo, los costes de cambios una vez la aplicación ha pasado a la fase de producción implican, en la mayoría de los casos, costes elevados e incluso en muchos casos los cambios son sino imposibles muy difíciles de llevar a cabo. Además, en el caso de aplicaciones Web, existe un coste adicional que es el de perder al usuario al pasar la aplicación a producción y ésta no presentar un rendimiento óptimo. Todo ello, sugiere la necesidad de soluciones para fases tempranas del ciclo de vida en las que los costes son inferiores.

Además, es necesario que dichas soluciones no impliquen costes elevados ni en tiempo ni en dinero ya que en caso contrario no serán útiles y, en consecuencia, no deben basarse en técnicas de predicción de rendimiento tales como la simulación o el análisis.

Como ya se mencionó en la sección 2.4[ISO/IEC 9126] existen recomendaciones de fabricantes y guías de buenas prácticas [Jagielski, 2002], [Gomolski, 2001], [Norton et al, 2003], [<http://www.weblogic.com/>],[50] pero éstas no se basan en resultados objetivos sino que, más bien, se derivan de la experiencia subjetiva e influida por aspectos como la tecnología utilizada o la arquitectura.

En lo que se refiere a intentos de búsqueda de soluciones relacionados con la configuración de aplicaciones Web, nos encontramos con los patrones de rendimiento (<http://www.sce.carleton.ca>) definidos en la sección 2.4 que tratan de ayudar a elegir la arquitectura adecuada según el punto de vista del rendimiento de forma análoga a la reutilización de buenas soluciones de diseño que se realiza con los patrones software . Los conceptos están basados en el hecho de identificar recursos y retardos que pueden limitar el rendimiento y aplicar patrones de diseño orientados a rendimiento y patrones de arquitectura, de forma que, se consiga el nivel de rendimiento deseado. Otro trabajo en este sentido es el citado en la sección 2.4, en el que se prueban diferentes configuraciones, en este caso relacionadas con el lugar que ocupa el archivo, para comprobar su influencia en el tiempo de obtención de un documento Web.

Se hace patente, por tanto, la necesidad de trabajar en soluciones para los gestores de aplicaciones que puedan aplicar en fases iniciales del proyecto para finalmente obtener una aplicación Web con un rendimiento óptimo al pasar ésta a producción, y no optimizándola a posteriori cuando muchos potenciales clientes ya no volverán al sitio Web.

4. CONCLUSIÓN

Según lo visto en el presente trabajo existen en la actualidad numerosos trabajos acerca del rendimiento Web aunque faltan soluciones para que los desarrolladores o administradores de aplicaciones Web puedan aplicar a sus configuraciones con el fin de no tener que esperar a realizar cambios referentes a tecnología, arquitectura, configuraciones, etc, al momento en que la aplicación se encuentra en Producción donde los costes son superiores.

REFERENCIAS

- [Almeida, 1998] Almeida J, Cao P, 1998. Measuring proxy performance with the Wisconsin Proxy Benchmark. *Proceedings of the Third International Web Caching Workshop*, Manchester, UK. <http://www.cs.wisc.edu/~cao/publications.html>.
- [Amerson et al, 2000] M. Amerson, G. Fisher, L. Hsiung, L. Krueger, and N. Mill ,2000. Design for performance. Analysis os download times for pages elements suggest ways to optimize. *IBM High-Volume Web Site Team*. <http://www-4.ibm.com/software/developer/library/perform/index.html>.
- [Berners-Lee et al, 1995] T. Berners-Lee, R. Fielding, H. Frystyk, 1995. "Hypertext transfer Protocol --HTTP/1.0". RFC 1945, *Internet Request for comment*.
- [Baboescu et al 2003] F. Baboescu, S. Singh, and G. Varghese, 2003. Packet classification for core routers: Is there an alternative to cams? *Proceedings of IEEE Infocom 2003*.
- [Boehm et al, 1978] Boehm, B.W., Kaspar, J.R. y otros, 1978. Characteristics of Software Quality, *TRW Series of Software Technology*.
- [Chankhunthod et al, 1996] A. Chankhunthod, P.B. Danzig, C.Neerdaels, M.F. Schwartz, and K.J. Worrell, 1996. A hierarchical Internet object cache. *In Proceedings of the USENIX 1996 Annual Technical Conference*, San Diego, California.
- [Diao et al, 2003] Y. Diao, J. L. Hellerstein, S. Parekh, J. P. Bigus, 2003. Managing Web Server performance with AutoTune agents. *IBM Systems Journal*.
- [Eaton, 2002] Eaton,P. R., 2002. Caching the Web with OceanStore. . *U.C. Berkeley Master's Report, Technical Report UCB/CSD-02-1212*.
- [Escribano et al., 2001] Escribano, J.J, Fernández, L., García M. J., 2001. Predicción del Tiempo de Carga en el Diseño de Páginas Web. *Mundo Internet 2001*.

- [Feldmann, 1999] A. Feldmann, R. Caceres, F. Douglis, G. Glass, and M. Rabinovich, 1999. Performance of web proxy caching in heterogeneous bandwidth environments. *In Proc. of IEEE INFOCOM '99*.
- [Fenton, 1999] Fenton, N., 1999. http://nick.dcs.qmul.ac.uk/~norman/papers/qa_metrics_article/section_3_metrics.html
- [Fielding et al 1997] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, 1997. "Hypertext transfer Protocol -- HTTP/1.1". RFC 2068, *Internet Request for comment*.
- [Gomolski, 2001] Gomolski, Barb, 2001. E-BUSINESS MATTERS: Top 10 recommendations on building scalable, high-performance sites. *InfoWorld*.
- [Grieser, 2001] T. Grieser, 2001. Web Performance Challenge. *Software Magazine*.
- [Harrington et al, 2002] D. Harrington, R. Presuhn, B. Wijnen, 2002. An Architecture for Describing Simple Network Management Protocol (SNMP) Management Frameworks. *RFC 3411, Internet Request for Comments*.
- [IEEE 1061] IEEE Std 1061-1992, IEEE Standard for a Software Quality Metrics Methodology. *IEEE Computer Society Press*.
- [ISO/IEC 9126] International Standard, ISO/IEC 9126-1991. Information technology – Software product evaluation – Quality characteristics and guidelines for their use.
- [Jagielski, 2002] J. Jagielski, 2002. What you get is what you see: keeping an eye on Web performance. *New Architect*.
- [Jung et al, 2002] Jung, J. , Sit, E. , Balakrishnan, H. , Morris, R., 2002. DNS performance and the effectiveness of caching, *ACM SIGCOMM Computer Communication Review*, v.32 n.1.
- [Jung et al, 2003] Jung, J., Berger, A. W., Balakrishnan, H., 2003. Modeling TTL-based Internet Caches. *Proc. IEEE Infocom*, San Francisco, CA.
- [Kubiatowicz et al, 2000] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, s. Rhea, H. Weatherspoon, W. Weimer, C. Wells, B. Zhao, 2000. OceanStore: An Architecture for Global-Scale Persistent Storage. *Proceedings of the Ninth International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)*.
- [Kuhns et al, 2002] Kuhns, F., Dehart, J., Kantawala, A. Keller, R. Lockwood, J. Pappu, P. Richard, D. Taylor, D. Parwatikar, J. Spitznagel, E. Turner, J. Wong, K., 2002. Design and evaluation of a high performance dynamically extensible router. *Proceedings of the DARPA Active Networks Conference and Exposition*.
- [Levitt, 1999] Levitt, J., 1999. Built to scale – Behind the Friendly Face of Many Web Sites Is A Sophisticated System For Maintaining Peak Performance. *Information week*.
- [Loosley, 2000] Loosley, C., 2000. E-commerce Response Time: A referente Model. *Proc. CMG'2000 International conference*.
- [Lyer et al, 2002] Lyer, S., Rowstron, A., Druschel, P., 2002. Squirrel: A decentralized peer-to-peer web cache. *Principles of Distributed Computing (PODC 2002)*, Monterey, CA.
- [Nielsen et al, 1997] H. F. Nielsen, J. Gettys, A. Baird-Smith, E. Prud'hommeaux, H. W. Lie, and C. Lilley, 1997. "Network performance effects of HTTP/1.1, CSS1 and PNG". *NOTE-pipelng-970207*, <http://www.w3.org/pub/WWW/Protocols/HTTP/Performance/Pipeline.html>.
- [Norton et al, 2003] Norton, C., Rangaswamy, S., 2003. Tuning WebSphere 5.0 using the performance advisors: getting the most from your Web Site. *WebSphere Developer's Journal*.
- [Norton et al, 2003] C. Norton, S. Rangaswamy, 2003. Tuning WebSphere 5.0 using the performance advisors: getting the most from your Web site. *WebSphere Developer's Journal*.
- [Olsina et al, 2002] Olsina, L.; Rossi, 2002. Measuring Web Application Quality with WebQEM, *In IEEE Multimedia Magazine*, Vol. 9, N° 4, pp. 20-29.
- [Pappu et al, 2003] Pappu, P., Parwatikar, J., Turner, J. S., Wong, K., 2003. Distributed Queuing in Scalable High Performance Routers. *Proceedings of IEEE Infocom*.
- [Postel, 1981] J. Postel, 1981. Transmission Control Protocol. RFC 0793, *Internet Request for comment*.
- [Rowstron et al, 2001] A. Rowstron, P. Druschel, 2001. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. *IFIP/ACM International Conference on Distributed Systems Platforms (Middleware)*, Heidelberg, Germany, pages 329-350.
- [Wolman et al, 1999] Alec Wolman, M. Voelker, Nitin Sharma, Neal Cardwell, Anna Karlin, Henry M. Levy, 1999. On the scale and performance of cooperative Web proxy caching. *Proceedings of the seventeenth ACM symposium on Operating systems principles*, p.16-31, Charleston, South Carolina, United States.
- [Zhi, 2001] Zhi, J., 2001. Web Page Design and Download Time. *CMG Journal of Computer Resource Management*, no. 102 (2001): 40-55.
- [Zeilenga, 2002] K. Zeilenga, 2002. Named Subordinate References in Lightweight Directory Access Protocol (LDAP) Directories. *RFC 3296, Request for comments*.