

Cherokee Web server

Alvaro López Ortega

Madrid

España

alvaro@alobbs.com

Cherokee es un proyecto que implementa una librería para poder dotar a toda clase de aplicaciones de servicios web de una forma fácil y rápida. En su desarrollo se realiza un esfuerzo especial en mantener un core reducido - de forma que se pueda utilizar en sistemas empotrados - e implementar todas las funcionalidades como módulos cargables en tiempo de ejecución. De igual forma, la alta eficiencia y una arquitectura lo suficientemente flexible como para poder escalar a servidores SMP son características de Cherokee que pueden suponer un paso adelante respecto a los servidores web libres existentes.

1. Introducción

Desde hace años, la tecnología que rodea la web tiene una importancia grandísima. Alrededor de esta tecnología se han producido grandes batallas; en unos casos por pura y sana competencia con productos que implementan una funcionalidad similar y en otros por el interés de romper los estándares. Sin ninguna duda, el vencedor indiscutible de esta batalla es Apache, un servidor web libre, que a día de hoy utiliza casi el 65% de los servidores en Internet.

Cherokee es otro servidor web. Se trata de un proyecto que desarrolla una nueva implementación de este tipo de aplicación, con una serie de características y funcionalidades concretas. El fin último del proyecto no es clonar ni imitar las funcionalidades de Apache, trabajo que sería sumamente complejo e innecesario, sino hacer un servidor con unas características de las que Apache carece debido a su diseño original.

Las funcionalidades de Apache son muchas y a estas hay que sumarle las que aportan los módulos y extensiones. El resultado es el que conocemos, Apache es un servidor web que puede hacer casi cualquier cosa con una flexibilidad y velocidad razonables. Ahora bien, todas estas características positivas tienen un coste: que en algunos casos se trata de un servidor lento, que no fue diseñado como un servidor para ser empotrado, etc.

En el diseño de Cherokee se han tenido en cuenta todos estos puntos en los que creíamos que Apache flaqueaba. Se ha puesto especial interés en los tres puntos siguientes: velocidad, flexibilidad y capacidad de ser empotrado.

La velocidad es un punto básico en Cherokee. En el último benchmark realizado hasta el momento (<http://www.alobbs.com/news/104>), Cherokee fue cinco veces más rápido que Apache. Este es un punto importante a tener en cuenta para la elección del software de un sitio web con mucho tráfico (por ejemplo, un servidor de banners). Posiblemente, gracias al cambio de software, se aplaza la necesidad de actualización del hardware del servidor.

El segundo de los puntos en los que se ha hecho especial énfasis en el diseño de Cherokee es la flexibilidad. En este respecto Cherokee dispone de un sistema para la carga dinámica de módulos (como el soporte `mod_*` de Apache)

tanto para handlers como encoders y sistema de logging. Siempre que se han añadido nuevas funcionalidades, se ha hecho de la forma más modular posible: a día de hoy, prácticamente cualquier nueva característica será implementada como un módulo cargable que Cherokee utilizará o no dependiendo de la configuración de este.

La última característica que se cuida con gran interés en el desarrollo de Cherokee es la capacidad para ser empotrado dentro de otras aplicaciones. Todo el código con la lógica del servidor se encuentra en una librería dinámica que puede utilizar cualquier aplicación. El API de esta librería es muy sencillo; básicamente permite crear, configurar y ejecutar de diferentes formas objetos "servidor". Existe una aplicación (<http://freshmeat.net/projects/gnome-cheroke>) para GNOME2 que implementa un servidor web personal (<http://cvs.gnome.org/bonsai/rview.cgi?cvsroot=/cvs/gnome&dir=gnome-network>) basándose en la librería de Cherokee.

2. Cherokee para los usuarios

A día de hoy, Cherokee tiene tanto ventajas como inconvenientes para los usuarios. Hay que tener en cuenta que se trata de un software que actualmente sigue en desarrollo y que hay funcionalidades no terminadas sobre las que se continua trabajando. Por otro lado, se trata de un proyecto con más de dos años de vida y presenta algunas grandes ventajas sobre otras opciones.

El principal inconveniente que tiene Cherokee hoy en día desde el punto de vista de un usuario final es la necesidad de un par de módulos importantes para servir contenidos dinámicos: PHP y Python.

Nota: Este artículo está escrito en Septiembre del 2003, en estos momentos ya hay trabajo realizado en la implementación de ambos módulos. Es probable que para la fecha de la celebración del congreso al que va dirigido el paper (III Jornadas Andaluzas de Software Libre) ambos módulos estén terminados, probados y funcionando.

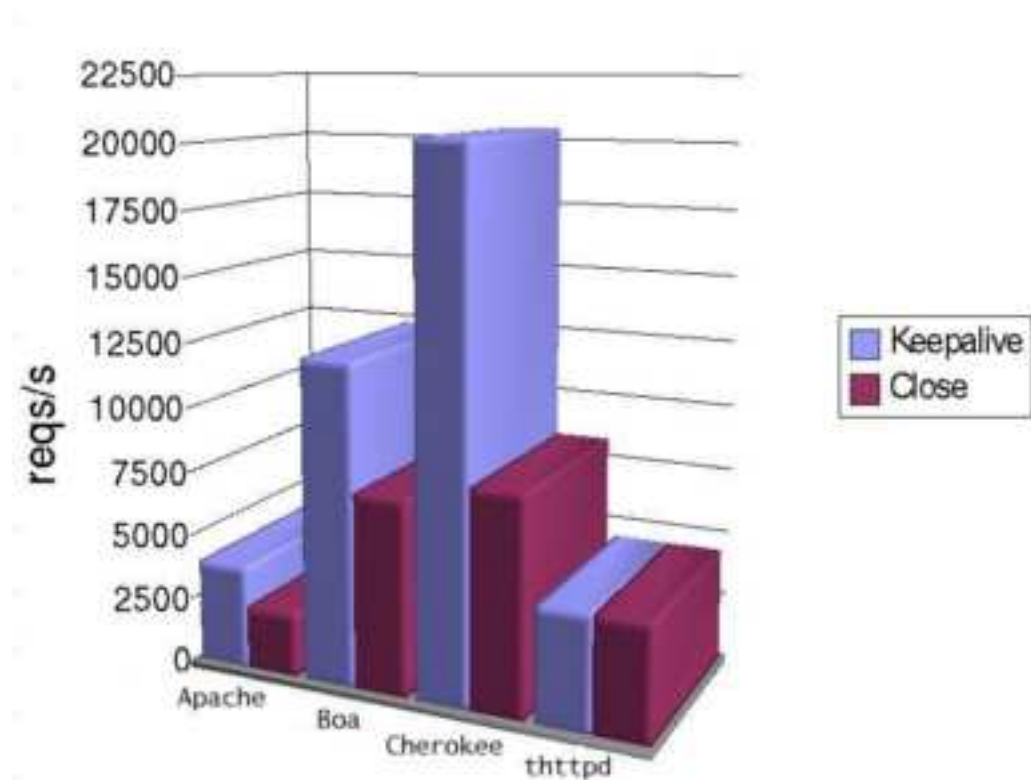
Por otro lado, también existen ventajas para los usuarios finales. La velocidad para servir las páginas se plantea como la principal de ellas; crea menos carga en máquinas pequeñas o desde otro punto de vista, admite más peticiones por segundo en máquinas que funcionen al límite.

Otra ventaja a tener en cuenta es que al tratarse de un servidor pensado para ser empotrado, es posible desarrollar sencillas interfaces gráficas con las que recubrirlo para que el usuario final trabaje con una herramienta mucho más amigable y no con ficheros de texto con muchas sentencias que es posible que no entienda. GNOME-Network utiliza actualmente libcherokee para implementar su aplicación de servidor web personal.

Por último, Cherokee al igual que Apache, pero al contrario que al gran mayoría de servidores web libres, escala a servidor SMP y máquinas con hyperthreading. La implementación del servidor tiene una visión doble: por una parte realiza servicio de peticiones mediante time-slices, pero por la otra es capaz de manejar más de un thread y en cada uno de ellos, de nuevo, volver a procesar conexiones mediante compartición de tiempo. De esta forma Cherokee escala a máquinas con más de un procesador y al mismo tiempo mantiene el rendimiento de los servidores basados en select() o poll().

Respecto a la implementación de time-slice de Cherokee, soporta poll(), epoll() (GNU/Linux 2.6.0 o superior) y emulación de poll() mediante select() para los sistemas en los que no existe función poll() en el sistema (MacOS X, por ejemplo).

Figura 1. Benchmark, Cherokee 0.4.3



3. Características principales

La librería de cherokee, libcherokee, implementa las características básicas de un servidor web, permitiendo cargar desde módulos muchas otras. Esta arquitectura modular ha sido elegida para permitir cargar y ejecutar únicamente las partes y funcionalidades que sean necesarias en cada caso concreto. De esta forma, se ahorran recursos, se aumenta la seguridad (menos código en ejecución implica menos posibilidad de existir un bug en él) y se disminuye ligeramente la carga del servidor web.

Hay tres grandes grupos de módulos cargables: handlers, encoders, validators. Los handlers son manejadores de peticiones. Cuando el servidor procesa una petición, decide que clase de manejador debe de utilizar para responder a dicha petición. Dependiendo de el módulo, la respuesta será una u otra. Cherokee incorpora el concepto de asociación de manejadores a directorios, de forma que el usuario puede definir que manejador desea utilizar en cada uno de los directorios servidos por web. Actualmente, se distribuyen los siguientes manejadores dentro del paquete principal de Cherokee:

1. file: Sirve ficheros al cliente
2. dirlist: Construye una página con lista de los ficheros contenidos en un directorio
3. redir: Redirecciona peticiones
4. nn: Basado en el algoritmo de "Near Neighbors" atiende las peticiones recibidas con una respuesta simple positiva
5. gnomevfs: Utiliza las librería de GNOME-VFS para atender las peticiones, de forma que es posible que sea el servidor web el que exporte ficheros localizados en ubicaciones accesibles bajo otros protocolos: NNTP,

FTP, HTTP (en este caso trabajaría de proxy), etc.

Los encoders por su parte, son módulos que implementan una funcionalidad de conversión de la información que se va a enviar a los clientes. Actualmente, el encoder más útil es el de GZip. Este módulo comprime la información que se sirve antes de enviarla a los clientes, ahorrando ancho de banda y acelerando la transmisión.

Los validadores son los módulos que implementan posibles formas de validar a los usuarios. En el momento de escribir este artículo (Septiembre 2003) se encuentran en desarrollo. Cherokee implementa módulos para validar con LDAP, PAM y httpasswd.

Todos los módulos son configurables en tiempo de ejecución, normalmente mediante cadenas de texto que procesa libcherokee.

4. Cherokee para el desarrollador

Desde el punto de vista de un desarrollador, Cherokee proporciona la librería (libcherokee.so) con la que dotar de las aplicaciones de servicios web. Para que un programa proporcione esta clase de servicios, lo único que tiene que hacer es linkar junto con esta librería y añadir un par de fragmentos de código adicionales.

1. La instanciación de un nuevo objeto servidor y la configuración de este.
2. El manejador (handler) o pasarela para que la aplicación exporte los datos que desee por medio de la interface web.

La implementación de un nuevo manejador de peticiones no es una tarea larga; simplemente hay que implementar 5 métodos virtuales (Aunque Cherokee está escrito por completo en C, utilizamos conceptos de programación orientada a objetos ya que existe un grandísimo parecido entre estos conceptos y la implementación en Cherokee): new(), init(), free(), step() y add_headers().

De igual forma, es posible implementar nuevos encoders y validadores implementando otro pequeño número de métodos. Para consultar los nombres y los parámetros de estos métodos, se puede consultar la documentación de libcherokee o directamente los ficheros de cabeceras en el código fuente: handler.h, encoder.h o validator.h.

5. Roadmap

Existe un roadmap (<http://www.alobbs.com/news/124>) publicado con los hitos cercanos en el desarrollo de cherokee. Al margen del desarrollo del core de Cherokee se está realizando la implementación de dos manejadores de especial importancia en el proyecto:

1. PHP4: De forma que todo el software disponible escrito en PHP se pueda utilizar bajo Cherokee.
2. Python: El objetivo de este manejador es en un primer momento, la implementación de server scripting basado en Python, y en un segundo paso la implementación de un conector para WebWare (<http://webware.sourceforge.net>).

6. Conclusiones

Cherokee tiene dos facetas. En primer lugar la de servidor web, en la que se centra en la eficiencia y la flexibilidad; implementa las funcionalidades deseables en un servidor web moderno (http 1.1, gran modularización, compresión gzip, etc).

En su segundo enfoque proporciona un framework con el que dotar de funcionalidades web a las aplicaciones de una forma rápida y sencilla, de forma que cualquier aplicación pueda incorporar servicios Web.