REDES DE COMUNICACIONES II

Práctica 1 - Servidor Web

Volver a: Prácticas

1 Introducción

1.3 Desarrollo de la práctica

Depuración

Cada vez que un estudiante de la EPS decide depurar su programa utilizando printfs() y otras chapuzas similares, Bill Gates mata a un gatito. Tenlo en cuenta, por favor, por los gatitos del mundo. En serio, depurar un programa a base de este tipo de funciones es, a parte de incómodo, mucho más lento e incorrecto, impropio de un programador serio.

Existen diversas alternativas:

- **Eclipse**, o cualquier otro IDE. Estos entornos disponen de un depurador integrado, fácil y sencillo de utilizar.
- **gdb**, un depurador que puede usarse desde la línea de comandos. Es ligero y viene instalado por defecto en casi cualquier distribución Linux. Más información sobre cómo utilizarlo aquí.

Otro aspecto esencial es que tu programa no tengas fugas de memoria, dado que va a ser programado en C. Para encontrarlas y solucionarlas, puedes (debes) utilizar Valgrind.

Orden de implementación

Es importante ordenar adecuadamente el desarrollo de las distintas características que debe cumplir la práctica para no retrasarte innecesariamente:

- Por ejemplo, como el servidor debe ejecutarse en forma de demonio, y ésto implica que la salida debe estar redirigida a un fichero de log y se trabaja con un proceso diferente al padre, depurar un programa en este modo es muy incómodo. Por eso, un consejo es desarrollar primera la función de demonización, comprobar que funciona y luego comentarla para desarrollar la práctica con un solo hilo de ejecución.
- El mismo comentario aplica a la concurrencia del servidor: puede resultar más efectivo actuar de la siguiente manera:
  - Elegir primero el esquema de servidor concurrente a desarrollar (con un proceso o thread para cade petición, con un pool de los mismos, etc.), implementarlo y comprobar que funciona correctamente sin ninguna funcionalidad HTTP, simplemente devolviendo una cadena predeterminada. Así se puede probar fácilmente el rendimiento del servidor y su estabilidad ante cargas muy altas.
  - A continuación, cuando se esté seguro de que esta parte está acabada, comentar las funciones necesarias para convertirlo de nuevo en un servidor iterativo (con un solo hilo de ejecución, sin threads ni procesos).
  - Así podrás centrarte ahora en desarrollar toda la funcionalidad de recepción, parseo y respuesta de peticiones HTTP, "aislando" esta parte del resto.
  - Cuando esté acabada, volver a activar la concurrencia y demonización.

Como ves, se trata de atacar el desarrollo por "partes", aislando las distintas características, y facilitar así el desarrollo.

Volver a: Prácticas