



Instituto Politécnico Nacional

ESCUELA SUPERIOR DE CÓMPUTO

Servidor Web HTTP Multihilo con Balanceo de Carga Simulado

Materia: Aplicaciones para
Comunicaciones en Red

Profesor: Axel Ernesto Moreno Cervantes

Grupo: 6CM1

Integrantes:

Jiménez Rodríguez Alejandro Martín
Núñez Ramírez Valery Aylin



Servidor Web HTTP Multihilo con Balanceo de Carga Simulado

Introducción

El presente proyecto consiste en el diseño e implementación de una aplicación servidor HTTP desarrollada en lenguaje Java. El objetivo principal es comprender el funcionamiento del protocolo HTTP, la gestión de conexiones mediante Sockets y la implementación de concurrencia a través de un Pool de Hilos (Thread Pool).

A diferencia de un servidor básico, esta implementación introduce una lógica de "balanceo de carga" o redireccionamiento: cuando la capacidad del pool definida por el usuario llega a su límite, el servidor principal redirige automáticamente las nuevas peticiones a un servidor espejo. Además, el servidor es capaz de identificar y servir diferentes tipos de recursos (MIME) y responder a los métodos fundamentales de la web: GET, POST, PUT y DELETE.

Desarrollo

Arquitectura del Sistema

El sistema se divide en dos componentes principales:

1. Servidor Web (Java): Gestiona los Sockets de escucha, el Pool de hilos y la lógica de respuesta HTTP.
2. Cliente (Navegador/Postman/cURL): Realiza las peticiones de recursos.

Implementación del Pool y Redirección

Se utilizó la interfaz ExecutorService con un FixedThreadPool. El flujo de control es el siguiente:

- Al iniciar, el usuario define la capacidad total del Pool.
- Por cada conexión entrante, se verifica el estado de hilos activos mediante ThreadPoolExecutor.getActiveCount().
- Si los hilos activos igualan el tamaño del pool, el servidor responde con un código HTTP 302 (Found) apuntando a la dirección del servidor espejo (puerto 8001), evitando la saturación del servidor principal.

Manejo de Métodos HTTP y Tipos MIME

Dentro de la clase interna Manejador (que implementa Runnable), se analiza la cadena de texto de la petición para extraer el método y el recurso.

- GET: Recupera archivos del disco duro (HTML, imágenes JPG, documentos PDF y archivos de texto plano).
- POST/PUT/DELETE: Se implementó una respuesta simulada que confirma la recepción del método y el recurso afectado, devolviendo un cuerpo HTML generado dinámicamente.
- MIME: Se configuró la cabecera Content-Type basándose en la extensión del archivo, asegurando que el navegador interprete correctamente el contenido.

Pruebas Realizadas

- Pruebas de Concurrencia: Se limitó el pool a 1 hilo y se realizaron múltiples peticiones simultáneas desde el navegador, verificando el redireccionamiento exitoso al puerto 8001.
- Pruebas de Interfaz: Se creó un archivo `index.html` con JavaScript (`fetch`) para ejecutar métodos que el navegador no realiza de forma nativa (PUT y DELETE) y visualizar los estilos CSS aplicados.

Conclusión

Alejandro Martin Jimenez Rodriguez

La realización de este proyecto permitió profundizar en la capa de aplicación del modelo OSI. Se demostró que la implementación de un Pool de Conexiones es vital para la estabilidad de un sistema, ya que previene el agotamiento de recursos del sistema operativo al limitar la creación indiscriminada de hilos.

Asimismo, se comprendió la importancia de las cabeceras HTTP; sin un manejo correcto de los tipos MIME y códigos de estado (200, 302, 404), la comunicación entre cliente y servidor sería errática. En conclusión, el servidor desarrollado cumple con los estándares de robustez y escalabilidad básica necesarios para entender arquitecturas web modernas.

Valery Aylin Nuñez Ramirez

Con ayuda de esta práctica se pudo profundizar sobre la comunicación que existe entre un cliente(navegador) y un servidor. De igual forma, pudimos recordar el uso de promesas en la parte del frontend dentro de nuestro java script correspondiente a la parte del Cliente. Así mismo, comprendimos la importancia del content-type que permite el envío y visualización de archivos de diferentes formatos.

Finalmente, el uso de un pool de hilos para la asignación de tareas fue un concepto nuevo y de mucha relevancia para servidores en la actualidad, pues no enseñó a establecer un sistema de balanceo de carga.