Práctica de la UT4 - Implementación de un Cliente y Servidor FTP Simplificados.

UT4. Generación de Servicios en Red
Programación de Servicios y Procesos

Febrero de 2025

Índice

1. Objetivos

1. Comprender y aplicar los conceptos de programación de aplicaciones en red.
2. Desarrollar habilidades en la implementación de comunicaciones cliente-servidor utilizando sockets en Java.
3. Implementar un protocolo de aplicación basado en FTP simplificado.
4. Gestionar sesiones de usuario, transferencia de archivos y comandos de control en un servidor de archivos.

2. Descripción

En esta práctica, se implementará una versión reducida del protocolo FTP. El servidor deberá aceptar conexiones de clientes que dispongan de credenciales válidas y permitir la navegación por directorios, la transferencia de archivos y la terminación de la sesión de manera ordenada.

3. Comandos Implementados

El protocolo simplificado soportará los siguientes comandos:

- `USER <nombre>` - Iniciar sesión con un nombre de usuario.
- `PASS <contraseña>` - Proporcionar la contraseña del usuario.
- `LIST` - Listar los archivos en el directorio actual del servidor.
- `PWD` - Mostrar el directorio actual en el servidor.
- `CWD <directorio>` - Cambiar al directorio especificado.
- `GET <archivo>` - Descargar un archivo del servidor.
- `PUT <archivo>` - Subir un archivo al servidor.
- `DELE <archivo>` - Eliminar un archivo en el servidor.
- `QUIT` - Cerrar la conexión de manera ordenada.

4. Guía de Desarrollo

Para facilitar la implementación, se recomienda seguir el siguiente flujo de trabajo:

1. Creación del Servidor FTP

- Implementar una aplicación en Java que escuche en un puerto determinado.
- Aceptar conexiones de clientes y manejar cada cliente en un hilo independiente.
- Gestionar sesiones de usuario con USER y PASS.
- Implementar el reconocimiento (*parsing*) de comandos para interpretar las solicitudes del cliente.
- Manejar operaciones sobre archivos y directorios.

2. Implementación del Cliente FTP

   - Desarrollar un cliente en Java que se conecte al servidor.
   - Enviar comandos al servidor y recibir respuestas adecuadamente.
   - Implementar la descarga y subida de archivos.
   - Manejar la desconexión de forma ordenada.

3. Manejo del reconocimiento y ejecución de comandos:

   - Para facilitar el reconocimiento y ejecución de los comandos, se recomienda implementar un diseño basado en objetos:
   - Una clase `CommandParser` que identifique y traduzca los comandos recibidos.
   - Una interfaz `FTPCommand`[1] con un método `execute()`.
   - Clases concretas que implementen `FTPCommand` para cada operación (por ejemplo, `UserCommand`, `ListCommand`, `GetCommand`, etc.).
   - Un `CommandExecutor` que asigne la ejecución del comando correcto según la entrada del usuario.

4. Juegos de pruebas requeridos

   a) El desarrollo de esta práctica requerirá la creación y ejecución de un conjunto de juegos de pruebas (véase la sección siguiente).

   b) Cada una de las pruebas deberá documentarse como se indica en la sección siguiente.

5. Juegos de Pruebas Requeridos

Para garantizar que la implementación del cliente y del servidor FTP es completamente funcional y robusta, es necesario desarrollar y documentar un conjunto de pruebas que validen todos los aspectos críticos del sistema. Las pruebas deberán cubrir:

[1] Si bien existen clases que ayudan al manejo de clientes y comandos de diversos protocolos (por ejemplo, en *Apache Commons Net*), aquí se trata de que implementes las clases objeto de este desarrollo por tu cuenta para así adquirir un conocimiento más profundo del funcionamiento tanto del protocolo como de los aspectos necesarios para poder ofrecer estos servicios y utilizarlos por parte de un cliente.

- **Pruebas de Conexión:** Validar que el cliente puede conectarse y desconectarse del servidor adecuadamente utilizando los comandos `USER`, `PASS` y `QUIT`.
- **Pruebas de Navegación:** Confirmar que los comandos `LIST`, `PWD` y `CWD` operan correctamente para listar contenidos, mostrar el directorio actual y cambiar de directorio.
- **Pruebas de Transferencia de Archivos:** Verificar que los comandos `GET` y `PUT` manejan la transferencia de archivos de manera efectiva, incluyendo tanto archivos pequeños como de tamaño considerable.
- **Pruebas de Eliminación de Archivos:** Probar el comando `DELE` para asegurar que los archivos se eliminan adecuadamente del servidor.
- **Pruebas de Manejo de Errores:** Incluir pruebas para manejar intentos de conexión con credenciales incorrectas, solicitudes de archivos inexistentes y comandos mal formados.

Cada juego de pruebas debe ser documentado con:

1. Descripción del objetivo de la prueba.
2. Pasos para ejecutar la prueba.
3. Resultados esperados y cómo verificarlos.
4. Observaciones sobre los resultados obtenidos.

6. Criterios de Evaluación

- Funcionalidad (50 %): Implementación correcta de los comandos y gestión de la comunicación cliente-servidor.
- Robustez (20 %): Manejo adecuado de errores, como autenticación incorrecta o archivos inexistentes.
- Juegos de pruebas (20 %). Creación, ejecución y documentación de juegos de pruebas. Se valorará su exhaustividad y adecuación de dichas pruebas.
- Documentación y estructura del código (10 %): Claridad en los comentarios y organización del código.

7. Entrega

La entrega de la práctica consistirá en:

- El código fuente del cliente y del servidor,
- Un informe breve explicando el diseño de la implementación y

- Juegos de pruebas (ficheros de prueba, forma de ejecución, resultados e informe razonado sobre las pruebas).