RELAZIONE

Introduzione

Il compito consiste nella realizzazione di un'applicazione client-server che permetta il trasferimento di file tra un client e un server, con funzionalità di lettura, scrittura e listing dei file. L'applicazione è composta da due programmi principali: **myFTserver** (server) e **myFTclient** (client). L'obiettivo del server è gestire più connessioni concorrenti dai client e garantire una gestione corretta delle richieste di lettura e scrittura dei file. Il client si occupa di inviare e ricevere file dal server, nonché di ottenere informazioni sui file presenti.

Obiettivo

L'obiettivo del progetto è sviluppare un'applicazione robusta che permetta di trasferire file tra un client e un server in modo efficiente, gestendo correttamente concorrenza, errori di rete e di sistema, e il corretto funzionamento del server in presenza di richieste multiple. Le operazioni richieste sono:

1. Scrittura di un file dal client al server.
2. Lettura di un file dal server al client.
3. Lista dei file presenti in una directory del server.
4. Gestione degli errori, come richieste di file non esistenti, spazio esaurito, interruzione della connessione, parametri di input errati, etc.

Architettura

L'applicazione client-server è organizzata come segue:

- **Server (`myFTserver`)**:
    - Si avvia con il comando `myFTserver -a server_address -p server_port -d ft_root_directory`.
    - Mette in ascolto il server su un indirizzo IP e una porta specificata.
    - Crea e gestisce una directory `ft_root_directory` per il salvataggio dei file ricevuti o la lettura di quelli richiesti.
    - Gestisce concorrentemente più client tramite threading o thread pool, assicurando che le richieste di scrittura sui file vengano eseguite in modo sicuro (lock sui file, gestione delle eccezioni).
    - Gestisce errori come la mancanza di spazio su disco, errori di connessione, e richieste malformate.
- **Client (`myFTclient`)**:
    - Si avvia con diversi comandi a seconda dell'operazione da eseguire:
        - **Scrittura**: `myFTclient -w -a server_address -p port -f local_path/filename_local -o remote_path/filename_remote`.
        - **Lettura**: `myFTclient -r -a server_address -p port -f remote_path/filename_remote -o local_path/filename_local`.
        - **Listing**: `myFTclient -l -a server_address -p port -f remote_path/`.

    - Gestisce errori come file non trovato, spazio insufficiente, interruzione della connessione e parametri di input errati.

Gestione della Concorrenza

Poiché il server deve gestire richieste concorrenti da più client, è necessario implementare un sistema di gestione della concorrenza. Una soluzione comune in questi casi è l'uso di **thread pool**, che consente di gestire un numero fisso di thread per l'elaborazione delle richieste. Ogni richiesta di lettura o scrittura può essere gestita da un thread separato, riducendo il carico e aumentando l'efficienza del server. I file devono essere protetti da accessi simultanei tramite l'utilizzo di meccanismi di **lock**, in modo che le richieste concorrenti di scrittura non compromettano l'integrità dei dati.

Gestione degli Errori

Il sistema deve essere in grado di gestire vari tipi di errori, come ad esempio:

- **File non esistente**: quando un client richiede un file che non esiste sul server, deve essere restituito un errore appropriato.
- **Spazio su disco esaurito**: se il server o il client non hanno spazio sufficiente per completare l'operazione, deve essere emesso un errore.
- **Connessione interrotta**: se la connessione tra client e server viene interrotta durante il trasferimento di un file, il sistema deve rilevarlo e gestirlo in modo adeguato.
- **Parametri errati**: se uno dei parametri di invocazione è errato (ad esempio, il percorso del file non esiste), deve essere segnalato all'utente con un messaggio chiaro.

Test e Verifica

Per verificare che il sistema funzioni correttamente, sono stati effettuati diversi test:

- **Test di scrittura e lettura**: per garantire che i file vengano correttamente inviati dal client al server e viceversa.
- **Test di concorrenza**: per verificare che più client possano interagire con il server contemporaneamente senza compromettere l'integrità dei file.
- **Test di gestione degli errori**: per simulare errori di rete, mancanza di spazio su disco e parametri errati, verificando che vengano gestiti correttamente dal sistema.

Conclusioni

In questo progetto, è stato sviluppato un sistema client-server che consente il trasferimento sicuro di file tra client e server, gestendo in modo efficiente le richieste concorrenti e gli errori. Il server è in grado di accettare richieste simultanee, garantendo che i file vengano trasferiti senza conflitti grazie alla gestione della concorrenza. Inoltre, il sistema è in grado di gestire diversi tipi di errori, assicurando un'esperienza utente robusta e priva di interruzioni.