Progetto di Reti

Michael Pierantoni
Matricola:0000974519
michael.pierantoni2@studio.unibo.it

26 luglio 2022

Sommario

Il progetto tratta la progettazione in Python di un server e di un client che si possono connettere ed attraverso il protocollo UDP (Transport Level) e il client seguendo alcuni comandi dettati dal protocollo di comunicazione può: richiedere la lista dei file disponibili sul server, scaricare un file dal server e caricare dei file sul server

Indice

1 Descrizione del progetto

1.1 Scopo

Lo scopo de progetto è quello di progettare ed implementare in linguaggio Python, un'applicazione client-server per il trasferimento di file che impieghi il servizio di rete senza connessione (socket tipo SOCK_DGRAM, ovvero UDP come protocollo di strato di trasporto).

1.2 Requisiti del software

- Connessione client-server senza autenticazione;
- La visualizzazione sul client dei file disponibili sul server;
- Il download di un file dal server;
- L'upload di un file sul server;

1.3 Funzionalità del server

- Connessione client-server senza autenticazione;
- La visualizzazione sul client dei file disponibili sul server;
- Il download di un file dal server;
- L'upload di un file sul server;

1.4 Funzionalità del client

- L'invio del messaggio list per richiedere la lista dei nomi dei file disponibili;
- L'invio del messaggio get per ottenere un file
- La ricezione di un file richiesta tramite il messaggio di get o la gestione dell'eventuale errore
- L'invio del messaggio put per effettuare l'upload di un file sul server e la ricezione del messaggio di risposta con l'esito dell'operazione.

2 Progettazione

2.1 Funzionalità List

il client invia il comando List al server che risponde con la lista dei file disponibili

2.2 Funzionalità Get

il client invia il comando Get al server con il nome del file che vuole scaricare, il server controlla se il file è disponibile e se non lo trova, comunica di non aver trovato il file, se trova il file, invia il nome del file, il peso del file (utilizzo nel controllo finale) ed infine inizia ad inviare il file.

2.3 Funzionalità Put

il client controlla se il file è presente, invia il comando Put al server con il nome del file che vuole caricare, poi invia il peso del file e poi inizia a caricare il file, il server inizia a ricevere il file e alla fine controlla se il peso corrrisponde, infine comunica al client se il trrasferimento è andato a buon fine o meno

2.4 Funzionalità Close

Digitare il comando close sul client per chiudere la connessione e spegnere il client stesso.

3 Architettura del progetto

3.1 Struttura delle directory

Il server e il client sono stati divisi in due cartelle omonime ed in ogni cartella troviamo il file di codice .py e una sottocartella Files con le risorse disponibili per quel dispositivo

3.2 Server

Il server crea un socket con indirizzo '127.0.0.1' (localhost) sulla porta 10000. il server invia e riceve i pacchetti dei vari file su due buffer di dimensione pari a 32 Kb e tra l'invio di un pacchetto e il successivo, avviene un time-out di 0.002 sec. Il server mostra nella propria console alcuni messaggi per visualizzare il proprio stato, cosa sta facendo e quali messaggi ha ricevuto dal client.

3.3 Client

Il client si collega al server con indirizzo '127.0.0.1' (localhost) sulla porta 10000. il client invia e riceve i pacchetti dei vari file su due buffer di dimensione pari a 32 Kb e tra l'invio di un pacchetto e il successivo, avviene un time-out di 0.002 sec. Il client mostra nella propria console alcuni messaggi per visualizzare il proprio stato, cosa sta facendo e quali messaggi ha ricevuto dal client.

4 Manuale

Avviare "UDPServer.py" e "UDPClient.py".

4.1 LIST

list

Comando utilizzato dal client per richiedere la lista dei file disponibili al server.

4.2 GET

get <filename>

Comando utilizzato dal client per scaricare un file dal server, il nome dev'essere lo stesso del file presente sul server e si può far la richiesta di un solo file alla volta.

4.3 PUT

put <filename>

Comando utilizzato dal client per caricare un file sul server, il nome dev'essere lo stesso del file presente sul client e si può far caricare solo un file alla volta.

4.4 CLOSE

close

Comando utilizzato dal client per chiudere il collegamento e terminare il client stesso.