Architettura Client-Server

Manuele Graziani
manuele.graziani@fermielearning.it
Istituto Tecnico Tecnologico Enrico Fermi
Frascati, Italia

1 INTRODUCTION

Questa applicazione di rete ha l'obiettivo di riuscire a far comunicare più processi client contemporaneamente con lo stesso server, questo è garantito tramite l'ausilio dei thread, implementati nella libreria `pthread`.

I client hanno il compito di inviare un messaggio di tipo `char` che viene ricevuto e letto dal server, dopo questa ricezione del messaggio, il server invia una risposta, per poi procedere alla chiusura della comunicazione.

2 ARCHITETTURA

Il server per gestire e garantire il servizio alle varie richieste da parte dei client, necessità di avere un socket avente il compito di rimanere costantemente in ascolto delle varie richieste; una volta accettate è compito di un altro socket, allocato a tempo di runtime, di gestire la comunicazione tra i due interlocutori assegnandolo a un thread.

Lo scopo del programma è di riuscire a gestire una quantità variabile di richieste da parte dei client, di conseguenza ho la necessità di memorizzare in modo dinamico i vari id dei thread, permettendomi successivamente di poter effettuare su di loro le `join`.

Per riuscire in tale scopo ho implementato una lista unidirezionale chiamata anche (lista semplicemente concatenata) avente il compito di mantenere le informazioni relative ai thread come: id, socket utilizzato e stato del thread.

3 TERMINAZIONE DEI THREAD

Nell'applicazione risedente sul server è presente un thread chiamato `terminatorThread` che ha il compito di effettuare le eventuali `join` sui thread.

Per evitare che il `terminatorThread` richiami la join su thread ancora in esecuzione, quindi portandolo in attesa della terminazione del thread, ho utilizzato una variabile booleana con il compito di informarlo sullo stato del thread in questione.

Il `terminatorThread` una volta aver eseguito la join su tutti i thread, portando di conseguenza la grandezza della lista uguale a zero, si mette in attesa di un segnale che lo informa sull'eventuale presenza di nuovi thread da *terminare*, l'invio di codesto segnale è onere del processo principale.

4 INFORMAZIONI GENERALI

Il codice del client ha la funzione di creare un numero definito di thread, con l'incarico di connettersi al server. Questo per poter testare la capacità del server di gestire più richieste contemporaneamente.