Tesina Sistemi operativi a.a. 2020/2021:

Studenti: Tamburrini Andrea n°matricola 0240885
Emanuele Valzano n°matricola 0268609

Manuale d'uso:

Per eseguire correttamente il software bisogna:

1) Compilare il file TCPServer.c ed eseguire su un terminale senza specifici parametri in input.

2) Compilare il file TCPClient.c ed eseguire su terminale inserendo in input l'indirizzo ip del server, ogni terminale che esegue questo software rappresenta un client.

Nella cartella e' già presente un MakeFile con le dovute regole di compilazione, pertanto per la corretta compilazione ed installazione basterà digitare su terminale:

>> make all

Specifiche progetto:

Chat multicanale

Realizzazione di un servizio "chat" via internet offerto tramite server che gestisce un insieme di processi client (residenti, in generale, su macchine diverse). Il server deve essere in grado di acquisire in input da ogni client una linea alla volta e inviarla a tutti gli altri client attualmente presenti nella chat.
Ogni client potrà decidere se creare un nuovo canale di conversazione, effettuare il join ad un canale già esistente, lasciare il canale, o chiudere il canale (solo nel caso che l'abbia istanziato).

Descrizione Progetto:
Il progetto e' stato sviluppato mediante l'utilizzo di system-call UNIX.

Il progetto si compone di due file principali : il TCPServer.c e il TCPClient.c.

Il TCPServer in particolare utilizza 2 header file denominati rispettivamente:
-chat_group.h
-reallocator.h

Istruzioni d'uso:
per interaggire con il server basta digitare i numeri che specificano le rispettive azioni.
Per abbandonare un gruppo bisogna digitare in input al client la stringa "abbandona gruppo".
Per utilizzare il comando per eliminare un gruppo bisogna digitare in input al client la stringa "elimina gruppo".

Descrizione main file TCPServer:

Questo file implementa le funzionalità tipiche di un server,ovvero ospitare in locale i gruppi di chat, permettere la ricezione e invio dei vari messaggi da/verso i client connessi.

Per evitare inconsistenza di dati sulle strutture dati dato il multi_threading è utilizzato prevalentemente il mutex lock.
Mediante le variabili num_client e num_group sono memorizzati rispettivamente il numero di client connessi e il numero di gruppi presenti in memoria.

Per tenere traccia dei client connessi il main-thread effettua una copia dell'id(canale) dei client all'interno dell'array di interi denominato client.

L'array di interi "group_client_presence" invece viene utilizzato per tenere traccia del gruppo di appartenenza dei singoli client.

L'assegnazione all'interno di questo array viene effettuata rispettando l'ordine di posizione nell'array "client" con corrispondenza uno a uno ossia, dato un id client presente nella posizione i-esima dell'array client,nella posizione i-esima dell'array group_client_presence viene registrato un intero corrispondente al gruppo di appartenenza di questo i-esimo client. Cosi con lo stesso indice(i) accedendo ai due array sono noti sia l'id(canale) che l'identificatore numerico del gruppo di appartenenza di uno specifico client.

Ogni volta che il main thread riceve la richiesta di collegamento da parte di un client e quest'ultima va a buon fine, esso provvede a lanciare un nuovo thread.

Questo thread va ad eseguire infatti la funzione "recive_mex".
Esso si fa carico del compito di attendere e successivamente scrivere nella pagina condivisa riservata per il gruppo di appartenenza, la sequenza di byte ricevuta dal client specifico.

Il main-thread ogni volta che arriva da parte di un client la richiesta di creazione di un nuovo gruppo, oltre a creare/configurare tutti gli elementi per l'implementazione del nuovo gruppo,lancia un nuovo thread il quale va ad eseguire la funzione "send_mex"

Il suo compito e' quello di inviare a tutti i client appartenenti al gruppo che questo thread gestisce,la sequenza di byte precedentemente riversata all'interno della pagina associata a questo gruppo.

Per evitare inconsistenza sui dati (byte letti),questo thread esegue una copia di questi ultimi in un proprio buffer locale e solo successivamente procede all'invio.

Il main thread una volta lanciati uno o entrambi i thread(nel caso di connessione e annessa creazione di un nuovo gruppo),aggiorna le strutture dati ossia i due array descritti in precedenza e si rimette in ascolto in attesa di un nuovo client.

I gruppi vengono allocati all'interno del server nel heap.
I riferimenti a questi gruppi vengono mantenuti mediante un array di pointer di tipo "gruppo" denominato buffer_group[MAX_GROUP].

Il tipo di dato "gruppo" viene descritto all'interno dell'header "chat_group".

Creazione nuovo gruppo:
Per ogni singolo gruppo all'atto della creazione viene mappata una pagina
in modalità MAP_SHARED rispettivamente di 4096 byte,il cui riferimento viene mantenuto all'interno dell'array di pointer di tipo char * denominato "shm_addr",il quale si occupa come si può intuire di mantenere il riferimento di tutti gli indirizzi di memoria delle varie pagine mappate per ogni singolo gruppo creato.

Questa pagina viene condivisa fra tutti i thread che eseguono la funzione "recive_max" (uno per ogni client appartente al gruppo) e dal singolo thread che esegue la funzione send_mex.
Questa pagina viene utilizzata per memorizzare la sequenza di byte ricevuta dal server da uno specifico client appartenete al gruppo.
L'accesso a questa pagina (una per ogni gruppo) e' gestita in modo sincrono mediante l'utilizzo di semafori.

Uno o più thread che eseguono la funzione recive_mex, i quali vogliono riversare i byte in queste pagine ,devono attendere che il thread che esegue la funzione send_mex termini le proprie operazioni di invio per poi rilasciare i token di accesso.

Descrizione file TCPClient.c:

Questo file implementa le basilari attività di un client per chat.

Nel momento in cui il client riesce a stabilire la connessione con il server mediante socket (modello TCP) esso lancia due thread dove un thread esegue la funzione "recive_mex" e il secondo la funzione "send_mex".

La funzione "recive_mex" si occupa semplicemente di ricevere e stampare su terminale la sequenza di byte ricevuta dal server, la quale corrisponde o a comunicazioni del server oppure corrisponde a messaggi inviati dai membri del gruppo di appartenenza.

La funzione “send_mex” invece e’ il duale della precedente funzione, ossia si occupa semplicemente di ricevere in input da terminale una stringa mediante la funzione fgets() e successivamente inviarla al server mediante la system-call send().

Nel momento in cui il client esprime mediante l’invio della stringa “abbandona gruppo” ,il server invierà ad esso la stringa “realloc” la quale mediante una condizione if permette di lanciare la funzione volonta().
Questa funzione(volonta()) altro non fa che prendere in input un intero da 1 a 3 e successivamente inviare questo intero al server.
Questo intero viene utilizzato in particolare dal thread che esegue la funzione recive mex associato a questo client, il quale all’arrivo in precedenza della stringa “abbandona gruppo” e’ passato ad eseguire la funzione “reallocated_client” descritta nel header “reallocator.h”.

Descrizione header file “chat_group.h”:

L’header-file “chat_group.h” viene utilizzato per la definizione di un nuovo tipo di dato che implementi il concetto di gruppo chat costituito da:
-variabile intero che mantiene memorizzato il canale del client che rappresenta l’amministratore del gruppo.
-variabile intero che memorizza il numero di client facenti parte del gruppo
-stringa che ospita il nome del gruppo
-array di interi che ospita la copia dei canali dei client connessi al gruppo

Al suo interno sono presenti anche metodi utilizzati principalmente dal main file TCPServer per la creazione, gestione ,eliminazione dei vari chat group.

Nel caso in cui l’amministratore dovesse abbandonare il gruppo ,viene designato come nuovo amministratore il primo client valido presenta nell’array che ospita i canali dei client connessi.

Descrizione header file “reallocator.h”:

Questo header viene anch’esso utilizzato esclusivamente dal main file TCPServer.
La funzione “reallocated_client” descritta al suo interno, viene utilizzata da uno dei thread del server che esegue la funzione recive_mex.
Dopo la prima assegnazione di un client ad un gruppo(creato da lui stesso/join ad un gruppo esistente) se quest’ultimo esprime esplicitamente, mediante l’invio al server della stringa “abbandona gruppo”, la volontà di abbandonare il gruppo di cui fa parte, il thread ad esso associato passa ad eseguire questa funzione.

La funzione offre la possibilità al client di :
1) creare un nuovo gruppo
2) joinare ad un gruppo esistente
3) stampare a schermo tutti i nomi dei gruppi esistenti ospitati nel server

Nel caso in cui il client dopo l'invio della stringa "abbandona gruppo" termina il collegamento,la funzione termina la sua attività e il Server registra il termine della comunicazione con il client.