

# **RELAZIONE SISTEMI OPERATIVI**

**2020/2021**

Docente: Prof. Francesco Quaglia



Macroarea Ingegneria

Anuar Elio Magliari, 0267412  
Michele Tosi, 0253591

Sommario

Introduzione ..... 3

Specifica ..... 3

Descrizione generale ..... 3

Breve descrizione sulla logica dei programmi ..... 4

    Server ..... 4

    Client ..... 4

Gestione segnali ..... 4

Manuale d’uso ..... 4

## Introduzione

Il modulo di Sistemi Operativi prevede lo svolgimento di un progetto da sviluppare utilizzando il linguaggio C in ambiente Unix. Questo documento descrive la struttura complessiva del progetto. Esso ha come scopo la realizzazione di un servizio “talk” via internet gestito tramite server.

## Specifica

Realizzazione di un servizio "talk" via internet gestito tramite server.

Il servizio tiene traccia di un elenco di client pronti ad iniziare una "conversazione".

I client (residenti, in generale, su macchine diverse), dopo essersi connessi al servizio acquisiscono l'elenco e lo mostrano all'utente il quale potrà collegarsi a un altro utente disponibile iniziando così la conversazione.

Due client collegati tra loro permetteranno ai relativi utenti di scambiarsi messaggi finché uno dei due non interrompe la conversazione, e saranno indisponibili ad altre conversazioni. Al termine di una conversazione i client ritorneranno disponibili fino a che non si scollegano dal server.

Si precisa che la specifica richiede la realizzazione del software sia per l'applicazione client che per l'applicazione server.

Per progetti misti Unix/Windows è a scelta quale delle due applicazioni sviluppare per uno dei due sistemi.

## Descrizione generale

Il servizio tiene traccia di un elenco di client pronti ad iniziare una "conversazione". I client (residenti, in generale, su macchine diverse), dopo essersi connessi al servizio acquisiscono l'elenco e lo mostrano all'utente il quale potrà collegarsi a un altro utente disponibile iniziando così la conversazione. Due client collegati tra loro permetteranno ai relativi utenti di scambiarsi messaggi finché uno dei due non interrompe la conversazione, e saranno non disponibili ad iniziare altre conversazioni. Al termine di una conversazione i client ritorneranno disponibili fino a che non si scollegano dal server.

La comunicazione tra client e server avviene tramite socket di tipo AF\_INET. Il server al suo interno implementa un sistema multi-thread in grado di gestire concorrentemente sia nuove connessioni da nuovi client che richieste su connessioni già stabilite.

Un client può collegarsi al server in ascolto identificandosi e rendendosi disponibile ad altri client.

Su richiesta può intraprendere una conversazione con i client disponibili nel display della tabella, se e solo se viene accettata da quest'ultimo. La comunicazione viene interrotta a seguito del messaggio

“quit” oppure con il segnale “SIGINT” rendendosi di nuovo disponibili. Possono scollegarsi dal server solo nel momento in cui non hanno nessuna comunicazione in corso.

Il server può gestire fino ad un massimo di connessioni concorrenti stabilito dalla “define” PENDING, se tale numero viene superato, il client rimarrà in attesa di un posto disponibile o altrimenti si scollegherà. Il server può essere terminato solo se viene chiuso il suo terminale oppure tramite un segnale di “SIGKILL”.

## Breve descrizione sulla logica dei programmi

### Server

- `client_online`: fornisce una tabella dei client disponibili ad intraprendere una conversazione, in caso di nessuna disponibilità il thread assegnato rimane in attesa
- `comunicazione`: permette una conversazione tra i client tramite socket fino a che uno dei due client coinvolti non digita la stringa “quit”
- `identificazione`: chiede al client il nome per poi inserirlo all’interno della tabella e imposta lo stato del client come disponibile
- `cambia_linea`: serve a mettere in relazione un client con quello che l’ha contattato
- `listen_client`: tiene traccia dei client che si disconnettono dal server
- `cerca_client`: controlla se il client con cui si vuole comunicare è connesso al server ed è disponibile per iniziare una nuova conversazione, in caso invia a quest’ultimo una richiesta per iniziare una conversazione
- `dealloca_memoria`: si occupa in caso di disconnessione di un client dal server di deallocare la memoria allocata per esso

### Client

- `gestione_comunicazione`: controlla se il canale di comunicazione con l’altro client è stato chiuso
- `gestione_uscita`: gestisce l’uscita dalla chat con conseguente chiusura della comunicazione

## Gestione segnali

Il segnale **SIGINT** viene gestito all’interno del client per uscire da una conversazione se questa è in corso mentre per terminare l’esecuzione dell’applicativo nel caso in cui questo sia in attesa di iniziare una nuova conversazione.

## Semafori

Nel server sono stati utilizzati semafori per evitare che i thread accedano in concorrenza ai dati condivisi che servono per istanziare la comunicazione tra client diversi evitando così un malfunzionamento del programma.

## Socket

Per la comunicazione sono stati utilizzati socket AF\_INET via IPV4 e TCP/IP tra il server e i client.

## Manuale d'uso

Manuale di compilazione:

È stato preparato un makefile per la compilazione, è possibile eseguire il comando:

- make per generare gli eseguibili del client e del server
- make Server per generare l'eseguibile del server
- make Client per generare l'eseguibile del client
- make clean per eliminare gli eseguibili generati

Manuale del server:

Una volta avviato il server questo va lasciato in attesa delle richieste dei vari client e funge come ponte tra essi

Manuale del client:

Una volta avviato il client inserire il nome per poter essere identificati dagli altri client connessi al server.

Una volta connessi si può digitare 'aggiorna' per aggiornare la lista delle persone connesse con cui poter iniziare una nuova conversazione oppure si può digitare il nome di un utente connesso al server per inviargli una richiesta che può essere accettata o meno da quest'ultimo.

Per uscire la conversazione si può digitare quit oppure si può utilizzare il segnale "SIGINT" facendo attenzione al fatto che se non si è in una conversazione questo chiuderà l'applicativo.