

Author: Michael Vasquez Otazu <u>michael.vasquezotazu@studenti.unicam.it</u>

Contenuti

1.	INTRODUZIONE	5
2.	FUNZIONAMENTO	
3.	INTERFACCIA	5
	IMPLEMENTAZIONE	
5.	METODI	Error! Bookmark not defined
6.	CONCLUSIONI	-

1. INTRODUZIONE

La presente relazione ha come scopo fornire una descrizione generale del progetto finale del corso di **Sistemi Operativi**, consistente in un programma **distribuito** basato sull'architettura **Client-Server** e la comunicazione di processi remoti tramite **socket**. Lo scopo di questo progetto è stato quello di permettere a due utenti collegati alla stessa rete di giocare a "**ping pong**" tramite la console dei loro sistemi.



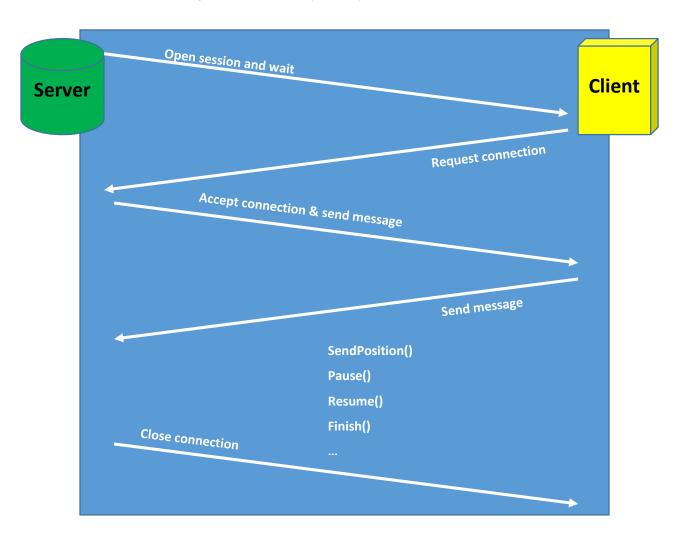
Il programma è stato sviluppato per essere **multipiattaforma** e quindi consentire la sua esecuzione su diversi sistemi operativi Unix (e.g. Debian) che Windows (W10). Tra le principale caratteristiche del programma si possono elencare:

- Selezione tra i protocolli TCP e UDP
- Personalizzazione della velocità della palla, dimensione delle racchette, ecc.
- Possibilità di fermare e riprendere il gioco
- Visualizzazione di informazione tramite una barra di stato
- Cattura di log contenenti le operazioni effettuate dal server e client
- Scambio di messaggi del tipo "json" tra client e server

2. FUNZIONAMENTO

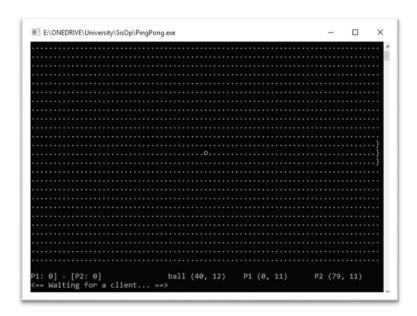
L'esecuzione sequenziale del programma è descritta a continuazione:

- 1. Il server apre una sessione (TCP o UDP) tramite la porta specificata dall'utente; rimanendo in attesa che un client si colleghi ad esso.
- 2. Il client si collega al server tramite l'IP e la porta di questo ultimo
- Il server invia un messaggio al client contenente la prima mossa
- Il client risponde al server tramite un altro messaggio contenente la sua mossa
- **5.** Questo scambio di messaggi si ripete, offrendo la possibilità di inviare, oltre le mosse, anche lo **stato attuale** del gioco e **commandi** specifici (pause, resume, finish, ecc.)



3. INTERFACCIA

La grafica con cui il programma è stato realizzato è basato sulla **console** del sistema operativo host, fornendo una interfaccia di gioco semplice ma amichevole arricchita con la visualizzazione di una barra di stato contenente informazione utile all'utente.



(execution on Windows)

Alla fine del gioco viene specificato lo **score** finale ed il giocatore **vincente**. Inoltre, i tasti supportati dal programma per lo svolgimento del gioco sono:

- Q: sposta in su la racchetta del player 1
- A: sposta in giù la racchetta del player 1
- Q: sposta in su la racchetta del player 2
- L: sposta in su la racchetta del player 2
- Backspace: ferma o riprendo il gioco
- Escape: finisce il gioco

4. IMPLEMENTAZIONE

L'applicazione è stata sviluppata completamente nel linguaggio C, utilizzando per la compilazione multipiattaforma i seguenti IDE:

Debian: Eclipse

Windows: VS2015

L'implementazione multipiattaforma ha richiesto la gestione diversificata di alcune librerie come quella dei socket (Winsock2.h per Windows, e Sys/Socket.h per Unix).

5. CODICE

Lo sviluppo del gioco ha richiesto l'implementazione delle seguenti classi:

- Main: Esecuzione principale del gioco
- Server: Gestione delle operazioni di socket del server
- Client: Gestione delle operazioni di socket del client
- Tools: Funzioni "utili" utilizzabili dalle suddette classi
- **PingPong**: Funzioni specifiche per lo svolgimento del gioco:

```
-Main
+-NewGame(create un nuovo gioco)
+-Start(inziare il gioco)
          ++-Clear(pullire lo schermo)
          ++-NewSet(aggiornare i parametri)
          ++-Turn(catturare le mosse ed aggiornare la posizione)
                       +++-GetMove(leggere le mosse)
                                 ++++-Server.SendMsg(leggere tasto ed inviare json)
                                 ++++-Client.GetMsg(ricevere e decodificare json)
                                 ++++-Client.SendMsg(leggere il tasto ed inviare json)
                                 ++++-Server.GetMsg(ricevere e decodificare json)
                       +++-Position (ricalcolare la posizione della palla e giocatori)
                                 ++++-Reverse(cambiare la direzione della palla)
                       +++-Display(scrivere a video tutti i dati)
                       +++-Idle(fermare e riprendere il gioco)
+-Finish(liberare la memoria e concludere il gioco)
```

6. CONCLUSIONI

Questo progetto è servito per mettere in pratica le conoscenze acquisite durante lo svolgimento del corso di **Sistemi Operativi**, sia nelle lezioni di teoria che quelle di laboratorio.

L'utilizzo dei **socket** ha permesso di entrare in contatto con uno scenario per lo scambio di messaggi tra due processi provenienti di computer diversi ma collegati alla stessa rete, venendo incontro a problematiche come il ritardo e la perdita di pacchetti, il tempo per gli intervalli d'invio, ecc.