Applicazione Client-Server mediante API dei Socket di Berkeley

LaTeX

Area di progetto

Lingua Italiana

Classe 3F

Anno 2015/2016

Indice

Elenco delle figure

1 Introduzione

Siamo dei ragazzi di terza dell'istituto tecnico informatico IISS GALILEO GALIEI. Abbiamo collaborato per realizzare un'area di progetto che ha lo scopo di creare un'applicazione Client-server del gioco del TRIS. Il Client comincia sempre la partita ed il Server risponde; la partita termina quando il Client o il Server vincono oppure quando il campo da gioco è completo.

2 Protocolli di rete [1]

I protocolli sono un'insieme di regole utilizzate dalle due macchine per scambiarsi informazioni e specificano cosa deve essere comunicato, in che modo e quando.
Se le due macchine sono remote, si parla di *protocollo di rete*. Una caratteristica comune dei protocolli di rete è il loro essere strutturati in **livelli sovrapposti**.
Il livello superiore esegue richieste al livello sottostante e i livelli uguali su macchine diverse conversano tramite lo stesso protocollo.
Uno dei vantaggi dei protocolli di rete è la progettazione di uno strato che deve esaminare solo alcuni aspetti del problema.
Ogni strato non dipende dall'implementazione degli altri strati.
Ogni strato fornisce servizi comuni a tutte le funzioni dello strato superiore.

2.1 Modello ISO/OSI [2]

Il modello ISO/OSI è uno standard per le reti di calcolatori che stabilisce per l'architettura logica di rete una struttura composta da una pila di protocolli di comunicazione di rete suddivisa in 7 livelli, i quali insieme eseguono le funzionalità di rete. I livelli del modello ISO/OSI sono i seguenti :

Livello	Vecchio modello	Nuovo modello
Livello 7	Strato applicazione	livello di Applicazione
Livello 6	Strato Presentazione	livello di Presentazione
Livello 5	Strato di Sessione	livello di Sessione
Livello 4	Strato di Trasporto	livello di Messaggio
Livello 3	Strato di Rete	livello di Pacchetto
Livello 2	Strato di Collegamento Dati	livello di Frame
Livello 1	Strato Fisico	livello Fisico

- Livello fisico: Definisce il modo in cui i dati sono fisicamente convertiti in segnali digitali sui media di comunicazione (impulsi elettrici, modulazioni della luce, ecc.)
- Livello collegamento dati: Definisce l'interfaccia con la scheda di rete e la condivisione del media di trasmissione.
- Livello di rete: Permette di gestire l'indirizzamento e il routing dei dati, cioè il loro invio tramite la rete.
- Livello di trasporto: É incaricato del trasporto dei dati, della loro divisione in pacchetti e della gestione degli eventuali errori di trasmissione.
- Livello di sessioni: Definisce l'apertura e la distruzione delle sessioni di comunicazione tra i terminali di rete.
- Il livello presentazione: Definisce il formato dei dati manipolato dal livello applicativo (loro rappresentazione, eventualmente loro compressione e loro codifica) indipendentemente dal sistema.
- Il livello applicazione: Assicura l'interfaccia con le applicazioni. Si tratta quindi del livello più vicino agli utenti, gestito direttamente da alcuni software.

2.2 TCP/IP [2]

I due protocolli TCP e IP furono concepiti da Bob Kahn e Vinton Cerf nel 1974, lo scopo era permettere l'interconnessione delle reti di calcolatori allora esistenti, come quella militare ARPANET, SATNET e altre network tecnologicamente diverse e indipendenti. Rappresenta in un certo modo l'insieme delle regole di comunicazione su internet e si basa sulla nozione d'indirizzamento IP, cioè il fatto di fornire un indirizzo IP ad ogni terminale di rete per poter inviare dei pacchetti di dati. Il modello TCP/IP, ispirato al modello OSI, riprende

l'approccio modulare, ma ne contiene solo quattro:

Modello TCP/IP	Modello OSI
Livello di Applicazione	livello di Applicazione
	Livello di Presentazione
	livello di Sessione
Livello di Trasporto(TCP)	Livello di Trasporto
Livello Internet(IP)	livello di Rete
Livello accesso rete	Livello di Collegamento Dati
	Livello Fisico

- **Livello di accesso di rete**: Descrive l' interfaccia fisica fra il dispositivo di trasmissione dei dati(calcolatore) ed il mezzo trasmissivo o la rete e definisce le caratteristiche del mezzo trasmissivo, natura dei segnali, tassi di trasmissione e lo schema di codifica dei dati.

  Il livello Accesso di rete è il primo livello della pila TCP/ IP,rappresenta i mezzi per realizzare una trasmissione di dati attraverso una rete. Il livello di accesso di rete contiene tutte le specifiche riguardo la trasmissione di dati su una rete fisica.

  Si incarica delle seguenti nozioni:

  1. Invio dei dati sul collegamento
  2. Coordinamento della trasmissione dei dati
  3. Formato dei dati
  4. Conversione dei segnali (analogico/digitale)
  5. Controllo degli errori all'arrivo

- **Livello di internetworking**: Descrive la trasmissione tra due nodi alla rete che sono la commutazione di pacchetto e la trasmissione non affidabile.

  Definisce il formato dei pacchetti, sistema di indirizzamento globale e il meccanismo di instradamento dei pacchetti. Il livello internet è il livello più importante dato che è quello che definisce i pacchetti di dati, e che gestisce le nozioni d'indirizzamento IP. Esso permette l'invio dei pacchetti di dati verso dei terminali remoti. Comprende 5 protocolli, i più importanti sono i protocolli IP, ARP e ICMP.

- **Livello di trasporto** Descrive la comunicazione fra due nodi della rete e garantisce il corretto ordinamento dei pacchetti e consegna affidabile dei dati. Il livello di trasporto fornisce un canale logico-affidabile di

comunicazione end-to-end per pacchetti. l livello trasporto permette a delle applicazioni che girano su terminali remoti di comunicare. Contiene due protocolli che permettono alle due applicazioni di scambiare dei dati indipendentemente dai livelli inferiori.

Questi protocolli sono:

- TCP: Esso fornisce un livello di trasporto affidabile e orientato alla connessione. Per affidabile si intende che prima di inviare i dati il server esegue l'handshaking, cioè chiede al client se è pronto a riceverli. Per orientato alla connessione s'intende la numerazione dei pacchetti in modo che il client li riceva in modo ordinato.Viene utilizzato per la sua sicurezza.
- UDP: Esso fornisce un servizio di trasporto datagram-oriented(non affidabile) e non orientato alla connessione.I pacchetti quindi vengono inviati senza chiedere al client se è pronto e inviandoli in modo disordinato. Viene utilizzato per la sua velocità.

Livello delle applicazioni Contiene la logica necessaria per supportare le varie applicazioni utente. Fornisce vari servizi di rete che sono il Login remoto, trasferimento file, posta elettronica e web. Un programma applicativo interagisce con uno dei protocolli di livello di trasporto per ricevere dati o inviarli passandoli nella forma richiesta.

Il livello applicazioni contiene le applicazioni di rete che permettono di comunicare grazie a livelli inferiori. I software di questo livello comunicano quindi grazie a TCP o UDP. Le applicazioni di questo livello sono di differenti tipi, ma la maggior parte sono dei servizi di rete, cioè delle applicazioni fornite all' utente per assicurare l' interfaccia con il sistema operativo.

2.3 Applicazioni di rete

Le applicazioni di rete sono composto da diversi elementi, in esecuzione su macchine differenti, che operano in modo indipendente e che possono scambiare informazioni. Le applicazioni sono processi comunicanti e distribuiti. La comunicazione avviene utilizzando i servizi offerti dal sottosistema di comunicazione. La cooperazione può essere implementata secondo vari modelli. Il modello più diffuso è quello client/server.

3 Socket [3]

3.1 Che cos è un socket?

Un socket, in informatica, è un'astrazione software, gestita dal sistema operativo, che rappresenta un canale di comunicazione di rete tra due processi. Per un programmatore, un socket è un particolare oggetto sul quale leggere e scrivere i dati da trasmettere o ricevere.

3.2 Funzionalità ed utilizzo dei socket

Il socket, nei sistemi operativi moderni, serve per poter utilizzare delle API standard, condivise attraverso la rete. Permette l'invio e la ricezione di dati, tra host remoti o tra processi locali. Socket locali e remoti in comunicazione, formano una coppia, composta da indirizzo e porta di client e server. Solitamente i sistemi operativi forniscono delle API per permettere alle applicazioni di controllare e utilizzare i socket di rete.
I protocolli utilizzabili per l'implementazione dei socket sono:

- protocollo TCP(Transfer Control Protocol)
- protocollo UDP (User Datagram Protocol)

Entrambi questi protocolli si appoggiano sul protocollo IP (Internet Protocol). I socket possono essere implementati con svariati linguaggi di programmazione, come C, C++ e java.

3.3 Nascita dei socket

L'origine dei socket risale al 1983, quando furono introdotti nel BSD (BSD- Berkeley Software Distribution) nell'università di Berkeley in California. La Advanced Research Project Agency finanziò l'università di Berkeley per implementare la suite TCP/IP nel sistema operativo Unix. I ricercatori di Berkeley svilupparono il set originario di funzioni che fu chiamato **"interfaccia socket"**.

3.4 Interfaccia Socket

L'interfaccia con le applicazioni non è solitamente definita all'interno dei protocolli di comunicazione, ma è invece parte dei sistemi operativi residenti che si limitano ad accogliere le linee generali dei protocolli. Così,le specifiche di un certo protocollo potrebbero suggerire che è necessaria un'operazione che consenta alle applicazioni di trasmettere dati, e l'API del sistema operativo stabilirà il nome della funzione relativa e il tipo dei suoi argomenti. Nonostante questa libertà di scelta, molti sistemi operativi, da Windows a varie versioni di UNIX, hanno scelto l'API detta socket (letteralmente, presa di corrente). Molti produttori di calcolatori hanno adottato il sistema di BSD come base per lo sviluppo di sistemi operativi commerciali, cosa che ha reso l'interfaccia socket lo standard di mercato.

3.5 API Socket

Si è detto che le applicazioni client server comunicano tramite i protocolli di trasporto. Tali applicazioni devono fornire al protocollo molte informazioni, come per esempio l'intenzione di fungere da server o da client, e altri dettagli relativi ai dati da trasmettere o ricevere. A questo fine, le applicazioni utilizzano la cosiddetta **Interfaccia con le Applicazioni** (API, Application Program Interface) che definisce le operazioni che queste possono eseguire. API determina quindi non solo gli strumenti disponibili per la comunicazione ma anche la difficoltà che si riscontra nello scrivere un programma che utilizzi tali strumenti.

4 Applicazione Client-Server [4]

4.1 Definizione

La struttura di un sistema informativo può essere il sistema client-server. Un sistema clientserver è un'architettura di rete formata da due tipi di moduli: il client e ilserver, che generalmente sono eseguiti su macchine diverse collegate in rete. Più semplicemente, i sistemi client-server sono un'evoluzione dei sistemi basati sulla condivisione semplice delle risorse. La presenza di un server permette ad un certo numero di client di condividerne le risorse, lasciando che sia il server a gestire gli accessi alle risorse per evitare conflitti di utilizzo tipici dei primi sistemi informatici. Un **server** è un componente informatico che fornisce servizi ad altri componenti, i

client,attraverso una rete. Un server svolge le operazioni necessarie per realizzare un servizio, ad esempio gestisce le banche dati, l'aggiornamento dei dati e la loro integrità. Con **Client** si indica una componente che accede ai servizi o alle risorse di un'altra componente, server, per effettuare alcune operazioni. Nel nostro caso, nel gioco del tris, il client richiede all'utente di inserire un simbolo (X) all'interno di un campo da gioco e il server legge quello che è stato scritto dal client e risponde con la stampa del campo più la sua mossa. Tipicamente il client gestisce la porzione di interfaccia utente dell'applicazione, verifica i dati inseriti e provvede ad inviare al server le richieste formulate dall'utente. Inoltre gestisce le risorse locali, come la tastiera, il monitor, la CPU, e le periferiche. In pratica il client è quella parte dell'applicazione che l'utente vede e con la quale interagisce.

4.2 Funzionamento

Un sistema client-server funziona secondo il seguente schema:

- Il client emette una richiesta con il suo indirizzo IP e il numero di porta verso il server ricevente che può così determinare a quale applicazione locale deve essere consegnato il messaggio, definendo così un servizio particolare del server.
- Il server riceve la richiesta e risponde attraverso l'indirizzo del terminale client e della sua porta.

4.2.1 Server Concorrenti e Iterativi

- concorrente, il server soddisfa più client "contemporaneamente" attraverso la gestione ricorsiva delle richieste.
- iterativo,il server accoglie e soddisfa una sola richiesta alla volta, attraverso la tipica procedura a coda di attesa.

4.2.2 Numeri di porta

- Le porte sono numeri (bit) utilizzati per identificare una particolare connessione tra quelle al momento attive su un dispositivo.
- In un determinato istante più processi possono usare i livelli di trasporto TCP o UDP;
- Sono necessari due livelli di indirizzamento:
  - Ogni macchina sulla rete deve avere un indirizzo che la individui univocamente;
  - Ogni applicazione su ogni macchina (multitasking) deve avere un porta che la individui univocamente;

4.2.3 Identificare una connessione

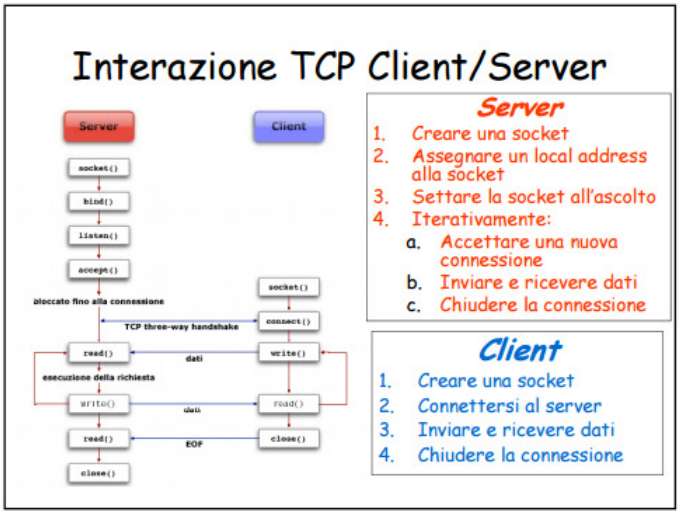
- Una porta (well known) viene assegnata ad un servizio, in caso di server ricorsivi o concorrenti ad ogni processo "figlio" viene assegnata una porta effimera con la quale comunicherà con il client.

- TCP e UDP usano 4 informazioni per identificare una connessione.
  - Indirizzo IP del server.
  - Numero di porta del servizio lato server.
  - Indirizzo IP del Client.
  - Numero di porta del servizio lato Client.

4.2.4 Endpoints e Server concorrenti

- L'endpoint è la coppia indirizzo IP e PORTA ed è nel protocollo IP.
- È considerato un server concorrente un server che genera un nuovo processo figlio per ogni richiesta Client.
- Un Nuovo Client effettua una richiesta allo stesso server.

5 Utilizzo client-server con Socket [5]



6 Makefile [6]

6.1 Nascita dei makefile

Come tutti i programmatori sanno, creare un programma eseguibile da più file sorgenti consiste nel compilare separatamente i file sorgenti uno ad uno e poi unirli nel file eseguibile finale. Quando i file sorgente sono molti, la compilazione potrebbe diventare noiosa e snervante. Per questo andiamo semplicemente a creare:

- un file contenente il compilatore da usare.
- il modo per usarlo.
- quali file compilare
- il file eseguibile (programma o libreria che esso sia) e altre cose sempre utili (come la directory dove si possono trovare le librerie necessarie ed i file di include). Tutto questo è il contenuto del file Makefile.

```
PROGS=error my_io my_signal str_srv_tris str_cli_tris tris_client tris_server
CC = gcc
CCN = gcc -std=c99

all:$(PROGS)

error:  error.c
        ${CC} -c error.c

my_io : my_io.c
        ${CC} -c my_io.c

my_signal: my_signal.c
        ${CC} -c my_signal.c

str_srv_tris: str_srv_tris.c
        ${CC} -c str_srv_tris.c

str_cli_tris: str_cli_tris.c
        ${CCN} -c str_cli_tris.c

tris_client:tris_client.o str_cli_tris.o my_io.o error.o
        ${CC} -o $@ $^

tris_server:tris_server.o str_srv_tris.o my_signal.o my_io.o error.o
        ${CC} -o $@ $^

clean:
        rm -f *.o
        rm -f *.*~
        rm -f *~
shish:
        rm -f *.o
        rm -f *.*~
        rm -f *~
        rm tris_client
        rm tris_server
```

Figura 1: Makefile

6.2 Il comando makefile

Il comando make permette di sviluppare programmi di notevoli dimensioni mantenendo traccia di quali porzioni dell'intero programma hanno subito modifiche. Solo tali parti però verranno compilate. In seguito viene proposto l'esempio del comando make utilizzato nell'esercizio per la compilazione delle varie parti del server.

6.3 Altri comandi

E' stato creato per cancellare tutti i file inutili.

Nel caso successivo, il comando "make shish", è stato creato per cancellare dei file binari. Con questo coman-

```
clean:
        rm -f *.o
        rm -f *.*~
        rm -f *~
```

Figura 2: comando per cancellare file inutili

do, andiamo a cancellare direttamente i file scritti (rm somma/client e rm somma/server) successivamente al comando "make clean", senza doverli selezionare uno ad uno.

```
shish:
        rm -f *.o
        rm -f *.*~
        rm -f *~
        rm tris_client
        rm tris_server
```

Figura 3: Comando per cancellare file binari

7 Segnali

Il server crea un processo figlio per gestire la connessione. Quando il processo figlio termina la connessione col client invia un segnale al processo padre e il processo figlio termina. Il processo figlio si trasforma in un processo ZOMBIE e sarà il processo padre a terminarlo con l'utilizzo della funzione wait() o waitpid(), altrimenti il processo ZOMBIE terminerà quando anche il processo padre verrà terminato(o ucciso).

8 Sviluppo Web

Per ogni progetto che si rispetti, c'è bisogno di un buon sito internet che lo descrive. Questo è il compito che i nostri sviluppatori Web hanno dovuto compiere. Per realizzare un buon sito web bisogna avere una buona fantasia e creatività e bisogna avere una buona manualità con il linguaggio web ce hanno utilizzato, ovvero HTML e CSS. Questo linguaggi fanno parte della famiglia dei linguaggi dichiarativi, più specificatamente si tratta di linguaggi a marcatori, poiché questi marcatori descrivono la struttura che si vuole ottenere come prodotto finale.

8.1 Esempio codice HTML

```
<html>
     <head>
                <title>Esempio HTML</title>
        </head>
<body>
        <font size="12px" color="black">
                   Questo è un esempio.
        </font>
</body>
</html>
```

8.2 Esempio Codice CSS

```
body{
        background-color: #fff;
        font-size: 12px;
        font-family: arial, sans-serif;
}
```

Come è visibile questi linguaggi sono ad alto livello e molto descrittivi, infatti ci lasciano immaginare quale sarà il risultato. Ma vediamo esattamente il procedimento che abbiamo svolto per la realizzazione di questa pagina web:

- inizialmente abbiamo realizzato con Photoshop il design del Layout della Pagina web.
- seguito abbiamo descritto il Layout utilizzato il linguaggio HTML per definire tutta la struttura, e il linguaggio CSS per definire ogni dettaglio della pagina (colori, grandezza del testo, ecc...) e per ordinare i vari Tag HTML.

9 LaTeX

9.1 Che cos'è Latex?

Il Latex è un linguaggio di programmazione sviluppato nel 1985 da Leslie Lamport. La cui ultima versione risale al 2011. Il Latex è un linguaggio a marcatori, in questo caso il marcatore più utilizzato è il (backslash). Il Latex è usato principalmente per produrre documenti in pdf. Questo linguaggio per produrre i file in pdf deve essere compilato.

9.2 Come abbiamo utilizzato il Latex

Per realizzare il nostro documento abbiamo utilizzato un template, ovvero un documento già creato da terze parti.Queste due stringhe servono per iniziare a programmare rispettivamente tutto il documento e per iniziare a programmare il vero e proprio documento pdf.Inoltre abbiamo aggiunto degli optional come l' indice e per distribuire bene il tutto abbiamo scelto di utilizzare dei capitoli e dei sottocapitoli. Per lo sviluppo ulteriore del codice abbiamo utilizzato alcuni pacchetti come quelli per i caratteri speciali, che comprende la possibilità di far riconoscere al sistema i caratteri speciali, come le lettere accentate.

9.3 Comandi Base

- \begin{document} = per iniziare il documento \end{document}
- \section{testo} = definisce un capitolo, aggiungendo sub davanti a section si crea un sottocapitolo.
- \tableofcontents= serve per creare un indice all' interno del documento
- \textbf= per grassetto
- \emph= corsivo
- \newpage=per passare alla pagina successiva e lasciare uno spazio
- \begin{itemize}= serve per creare un elenco puntato e si segna con \item segnare l'elenco

Figura 4: comandi Base LaTeX

10 Progetto

10.1 Istruzioni per la compilazione

Esegui il comando tris_server in background da terminale. Per compilare questo software é necessario possedere un Sistema Operativo Unix-like (es. Linux, BSD e Mac OS X). É inoltre necessario installare i seguenti programmi:

- make
- gcc
- tar
- xz

Percioó su distribuzioni debian/ubuntu-based, lanciare il seguente comando da superutente: apt-get install build-essential In distro basate su Red Hat e/o Fedora:

- yum: yum install make gcc gcc-c++ kernel-devel
- dnf: dnf install @development-tools

Su Arch Linux (e distro basate su quest'ultima): pacman -S base-devel

10.2 Compilazione

Lanciare da utente standard: tar -xvJf tris-1.0.tar.xz

cd tris

make

10.3 Installazione

Lanciare da superutente:

```
install -m755 tris_client -t /usr/local/bin/

install -m755 tris_server -t /usr/local/bin/
```

).

10.4 Utilizzo

Per utilizzare questa applicazione bisogna avviare tris_server e lasciare lavorare l'applicazione in background o in localhost (IP: 127.0.0.1) o su un altro PC lanciare

```
ifconfig $INTERFACE | grep inet | grep -v inet6 | awk '{ print $2 }'
```

oppure

```
ifconfig $INTERFACE | grep inet | grep -v inet6 | awk '{ print $1 }' | sed 's/inet://'
```

Ora, avviare l'applicazione client con

```
tris_client $IP
```

Riferimenti bibliografici

[1] Wikipidia.

[2] it.ccm.net/contents/42-tcp-ip .

[3] Wikipedia e www.dacrema.com/Informatica/Socket.html.

[4] Wikipedia,Appunti di Informatica: Architettura Client/Server e Pear to Pear- Indirizzi IP – Indirizzamento statico e dinamico ,“Dipartimento di Ingegneria dell’Informazione ” - Università degli Studi di Siena (Gianluca Daino),www.di.uniba.it.

[5] Materiale fornito dal Prof. Iaccarino.

[6] materiale didattico fornito dal Prof Iaccarino.