Prova di laboratorio di reti di calcolatori

20 Aprile 2012, Lab. Turing

Docente: R. De Prisco

Si progetti e si implementi un web server ed il corrispondente web browser che operano nel seguente modo:

- Il web browser apre una connessione TCP con il web server ed invia la stringa "CONNECT".

- Il web server controlla il numero di porta usato dal client e se è pari accetta la connessione inviando "ACCEPTED" se invece è dispari rifiuta la connessione inviando "DENIED" e chiudendo la connessione.

- Per le connessioni accettate, il web browser acquisisce da tastiera l'indirizzo della risorsa a cui si vuole accedere. Per semplicità assumiamo che le risorse (pagine web) siano identificate dal nome del file che contiene i dati da trasferire. Dopo l'acquisizione da tastiera il web browser invia "GET *nomefile*" al web server, dove *nomefile* è il nome del file che l'utente ha digitato sulla tastiera.

- Il web server riceve la richiesta, controlla che sia nella forma GET *nomefile* e controlla che il file esista. Se il file non esiste invia il messaggio di errore "404 FILE NOT FOUND". Se il file esiste invia il messaggio "200 OK". Poi invia il numero di righe del file e quindi il contenuto del file.

- Il web browser riceve il messaggio del web server e lo visualizza. Se il messaggio è "404 FILE NOT FOUND" permette all'utente di inserire una nuova richiesta. Se il messaggio è "200 OK" allora riceve il file (prima il numero di righe e poi il contenuto) visualizzando il file sul video. Al termine permette all'utente di richiedere una nuova risorsa.

- Per chiudere la connessione l'utente del web browser digita il comando "BYE". In tale situazione il web browser chiude la connessione con il server.

Gestire opportunamente le situazioni di errore. Far stampare su video da parte del server tutto ciò che succede (richiesta di connessione, messaggi ricevuti, messaggi inviati).

PS. Per semplicità si può assumere che sia l'utente stesso a inserire i comandi (CONNECT, GET, BYE) da tastiera

Prova di laboratorio di reti di calcolatori

16 Febbraio 2012, Lab. Turing

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione di una chat con più utenti. Ogni utente si registra al server con un nome e tutto ciò che un utente invia al server verrà inviato a tutti gli altri utenti connessi alla chat.

- Il client acquisisce da tastiera una stringa con cui l'utente specifica il proprio nome e lo invia al server.
- Per ogni client che si connette il server si aspetta di ricevere nella prima linea inviata il nome dell'utente.
- Dopo l'invio del nome il client spedisce al server tutto ciò che l'utente digita sulla tastiera e visualizza su video tutto ciò che riceve dal server.
- Per ogni riga R ricevuta da un client il cui nome è N il server spedisce una riga del tipo "[N] R" a tutti gli altri client connessi.

Gestire opportunamente le situazioni di errore.

Suggerimento1: per poter implementare questa applicazione è necessario usare la funzione "select".

Suggerimento2: usare come punto di partenza il codice dell'echo client-server utilizzato come esempio durante le lezioni.

Prova di laboratorio di reti di calcolatori

23 Gennaio 2012, Lab. Turing

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione di immagini relative alla situazione traffico delle autostrade. Si assuma che il server abbia già le immagini memorizzate.

- Il client acquisisce da tastiera una stringa con cui l'utente specifica il numero identificativo della telecamera oppure il comando "bye"

  125
  bye

- Il server riceve la richiesta e controlla se nel database c'è un'immagine relativa alla telecamera specificata. In tal caso il server spedisce il file che contiene l'immagine. Se non c'è l'immagine allora il server spedisce un codice di errore.

- Se il comando è "bye" il server chiude la connessione.

- Il client riceve la risposta e visualizza un messaggio in cui specifica o l'errore oppure il numero totale di byte ricevuti.

  Ho ricevuto un file di 8124 byte.

Gestire opportunamente le situazioni di errore.

Prova di laboratorio di reti di calcolatori

17 novembre 2011

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per il gioco della carta più grande. Il gioco prevede che almeno 3 client siano connessi al server.

- Il server funge solo da punto di raccolta delle puntate.

  Il server inizia il gioco inviando un messaggio ad ognuno dei 3 client con un numero casuale fra 1 e 10 (la carta).

- Ognuno dei client visualizza il numero ricevuto e chiede all'utente di scegliere se tenere o meno la carta ricevuta (Si/No oppure 0/1).

- Se la risposta da un client è No oppure 0, il server replica con una nuova carta. Se la risposta del client è stata Si oppure 1, il server non spedisce niente.

- Vince il client che ha la carta più grande. Il server invia il messaggio "Hai vinto" al client con la carta più grande ed il messaggio "Hai perso. Vince la carta x" agli altri client dove x è il numero della carta vincente.

- Se ci sono ancora almeno 3 client connessi il server inizia una nuova partita.

  Suggerimento: rand()%10 +1, genera un numero casuale fra 1 e 36

Prova di laboratorio di reti di calcolatori

5 settembre 2011

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per una roulette multi-utente online.

- Il server funge solo da punto di raccolta delle puntate.

  I client si connettono al server e possono o puntare o far girare la ruota. Un client può inviare due tipi di messaggio, una puntata (numero da 1 a 36, "pari" o "dispari") oppure il comando per far "girare" la ruota.

  Esempio:
  C->S **13**
  C->S **pari**
  C->S **dispari**
  C->S **gira**

- Per ogni puntata ricevuta da un client il server memorizza la puntata fatta (suggerimento: La puntata "dispari" può essere rappresentata dal 37 e la puntata "pari" dal 38)

- Il server stampa un opportuno messaggio sul video per ogni evento: connessione di un client, ricezione di una puntata, ricezione del comando "gira", invio dei messaggi.

- Quando un client invia il comando "gira", il server genera un numero casuale fra 1 e 36 e per ogni client che ha fatto una puntata (quindi escluso il client che ha inviato il comando gira) risponde con un opportuno messaggio del tipo:

  S->C **Esce il 25. Ha vinto!!!!**
  S->C **Esce il 25. Hai perso.**

  Suggerimenti:
  1. rand() % 36 +1, genera un numero casuale fra 1 e 36
  2. La puntata "dispari" può essere memorizzata come "37"
  3. La puntata "pari" può essere memorizzata come "38"

Prova di laboratorio di reti di calcolatori

13 Luglio 2011

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per una partita di "battaglia navale" online.

- Il server ha a disposizione (in un file) la mappa, di dimensioni 10x10, delle navi. Per semplicità assumiamo che ogni nave occupi una sola posizione.
  Il formato del file è libero, quindi si scelga la rappresentazione che si ritiene più opportuna. Una possibile rappresentazione è una semplice lista di coordinate che indica la posizione delle navi:

  A8
  B0
  E4
  E9
  F5

- Il client si connette al server.

- Il server invia una messaggio del tipo "Benvenuto! Devi colpire x navi", dove x è il numero di navi presenti nella mappa.

- Il client invia messaggi nel formato "xy" dove x è una lettera da A ad L e y un numero da 0 a 9. Tali messaggi indicano le coordinate della casella dove si intende piazzare la bomba.

- Il server riceve il messaggio e controlla il formato. Se il formato non è esatto risponde con un messaggio di errore. Altrimenti controlla sulla mappa per vedere se nella posizione specificata c'è una nave. Quindi risponde con "Colpo andato a vuoto" se non c'è nessuna nave oppure con "Nave colpita. Rimangono x navi." o con "Hai completato il gioco con x bombe!!!", dove x è il numero di "bombe" utilizzate dal momento della connessione.

  Esempio: Il client si connette al server con la funzione "connect"

  S->C Benvenuto! Devi colpire 3 navi
  C->S A7
  S->C Colpo andato a vuoto
  C->S H3
  S->C Nave affondata. Rimangono 2 navi.
  C->S H3
  S->C Colpo andato a vuoto
  C->S G2
  S->C Nave affondata. Rimane 1 nave.
  C->S F87
  S->C Messaggio non conforme alla specifica
  C->S F8
  S->C Hai completato il gioco con 4 bombe!!

Prova di laboratorio di reti di calcolatori

27 Giugno 2011

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione online di una biblioteca universitaria.

- Il client acquisisce da tastiera una stringa con cui l'utente specifica il codice numerico del libro che si vuole prendere (l'utente ha già consultato una lista con i titoli dei libri e da questa lista ha estratto il codice del libro) ed una lettera che specifica se il libro è un libro di testo di un corso ("T"), oppure no ("A")

  Esempio (client): `12345 T`
  (I due valori possono anche essere inviati su due linee separate)

- Il server riceve il codice e controlla se il libro è disponibile. Si assuma che le informazioni siano memorizzate in un file con il seguente formato:

  codice,0/1,giorni:
  `1234,0, 7`
  `2345,1,0`
  `3456,1,0`

  Spiegazione: Il libro 1234 non è disponibile (secondo campo 0), rientra fra 7 giorni. Il libro 2345 è disponibile (secondo campo 1), il terzo campo non ha significato in questo caso. Il libro 3456 è disponibile (secondo campo 0).

- Il server risponde con OK se il video è disponibile e garantisce la consultazione per 2 giorni se il libro è un libro di testo, per 7 giorni se è un altro tipo di libro. Inoltre il server aggiorna il database. Altrimenti risponde con "Libro non disponibile, rientra fra x giorni".

- Il client riceve tali informazioni e le visualizza all'utente.

Esempio:

C->S	`1234 T`
S->C	**Libro non disponibile, rientra fra 7 giorni**
C->S	`2345 A`
S->C	**ok, devi restituire il libro fra 7 giorni**
C->S	`2345`
S->C	**Libro non disponibile, rientra fra 7 giorni**
C->S	`3456 T`
S->C	**ok, devi restituire il libro fra 2 giorni**

Prova di laboratorio di reti di calcolatori

13 Giugno 2011

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione di una videoteca automatica. La videoteca ha varie postazioni utente ognuna delle quali funziona da client ed un server centrale che gestisce la disponibilità dei video. I client ed i server interagiscono nel seguente modo:

- Il client si connette al server.
- Il client acquisisce da tastiera una stringa con cui l'utente specifica il codice numerico del video che si vuole prendere (l'utente ha già consultato una lista con i titoli dei video e da questa lista ha estratto il codice del film) ed il numero di giorni per il quale si vuole fittare il video.

  Esempio (client): `2345  4`
  (I due valori possono anche essere inviati su due linee separate)
- Il server riceve il codice e controlla se ci sono video disponibili. Si assuma che le informazioni siano memorizzate in un file con il seguente formato:

  codice,0/1,giorni max,rientra fra:
  `1234,0, 5,3`
  `2345,1,6,0`
  `3456,0,3,2`

  Spiegazione: Il video 1234 non è disponibile (secondo campo 0), può essere fittato per al massimo 5 giorni, rientra fra 3 giorni. Il video 2345 è disponibile (secondo campo 1), può essere fittato per al massimo 10 giorni – il quarto campo non ha significato in questo caso. Il video 3456 non è disponibile (secondo campo 0), può essere fittato per al massimo 8 giorni, rientra fra 2 giorni.

- Il server risponde con OK se il video è disponibile per il numero di giorni richiesto ed aggiorna il database. Altrimenti risponde con "Video non disponibile, rientra fra x giorni", oppure con "Video disponibile solo per x giorni".

- Il client riceve tali informazioni e le visualizza all'utente.

Esempio:

C->S	`1234 3`
S->C	*Video non disponibile, rientra fra 3 giorni*
C->S	`2345 5`
S->C	*ok*
C->S	`2345`
S->C	*Video non disponibile, rientra fra 5 giorni*
C->S	`3456 5`
S->C	*Video disponibile solo per 3 giorni*

Prova di laboratorio di reti di calcolatori

19 Aprile 2011

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione di un magazzino merci.

- Il client si connette al server.

- Il client acquisisce da tastiera una stringa con cui l'utente specifica il codice numerico dell'articolo che interessa.

  Esempio (client): 40123

- Il server riceve il codice e legge da un file le informazioni relative all'articolo. Tali informazioni sono: la descrizione dell'articolo, il prezzo, ed il numero di articoli disponibili.
  Il server invia tali informazioni al client.
  Esempi 40123,Televisore LCD,120,5
  50124,Frigo classe AAA,480,7
  20688,articolo sconosciuto,0,0

- Il client riceve tali informazioni e le visualizza permettendo all'utente di aggiornare le quantità disponibili con messaggi del tipo "+x" o "-x".

- Il server riceve la richiesta e dopo aver aggiornato il file che contiene la disponibilità risponde con "ok" oppure con "errore".

  Esempio1: C->S 40123
  S->C 40123,Televisore LCD,120,5
  C->S -1
  S->C ok
  C->S 40123
  S->C 40123,Televisore LCD,120,4

  Esempio2: C->S 40123
  S->C 40123,Televisore LCD,120,5
  C->S -8
  S->C errore

  Esempio3: C->S 40123
  S->C 40123,Televisore LCD,120,5
  C->S 40123
  S->C errore

Prova di laboratorio di reti di calcolatori

17 Febbraio 2011, Lab. Turing, ore 15:00

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione di un cinema multisala.

- Il client si connette al server.

- Il client acquisisce da tastiera una stringa con cui l'utente specifica il nome del film a cui è interessato e lo invia al server.

  Esempio (client): **Terminator**

- Il server riceve il nome del film e legge da un file le informazioni relative allo spettacolo. Tali informazioni sono: le date in cui lo spettacolo è in programmazione e per ogni data i posti disponibili. Il server invia tali informazioni al client.
  Suggerimento: Si invii le informazioni relative ad ogni spettacolo su una linea; poiché il client non sa a priori quanti spettacoli sono disponibili si invii come segnale di fine lista una linea che contiene una stringa particolare (ad esempio "**fine**").

- Il client riceve tali informazioni e le visualizza permettendo all'utente di scgliere una data ed il numero di biglietti da acquistare. La scelta dell'utente viene inviata al server.

  Esempio (client):
  1. 18/02/2011 ore 20:00 sala 1 – 20 posti disponibili
  2. 18/02/2011 ore 21:00 sala 2 – 5 posti disponibili
  3. 19/02/2011 ore 20:00 sala 1 – 30 posti disponibili
  4. 20/02/2011 ore 20:00 sala 2 – 32 posti disponibili

  Si noti che il client non visualizza la stringa di fine lista.

- Il server riceve la richiesta e dopo aver aggiornato il file che contiene la disponibilità risponde con "x posti prenotati" oppure con "posti disponibili non sufficienti".

  Esempio1: 3 5
  5 posti prenotati
  Esempio2: 2 7
  posti disponibili non sufficienti

Nota: Per semplicità si utilizzi un file per ogni film. Si scelga un opportuno formato per tale file. Ad esempio, il file **Terminator.txt** potrebbe contenere 4 linee:

20,18/02/2011 ore 20:00 sala 1
5,18/02/2011 ore 21:00 sala 2
30,19/02/2011 ore 20:00 sala 1
32,20/02/2011 ore 20:00 sala 2

Prova di laboratorio di reti di calcolatori

25 Gennaio 2011, Lab. Turing, ore 14:00

Docente: R. De Prisco

Si progetti e si implementi un'applicazione client-server per la gestione di un negozio on-line che vende computer ed accessori.

- Il lato programma client acquisisce da tastiera una stringa con cui l'utente un numero identificativo per la transazione, l'oggetto richiesto, la quantità ed un numero di carta di credito sul quale addebitare il pagamento.
  Ad esempio, una possibile richiesta può essere

  12345 mouse 2 1234123412341234

- Il server riceve la richiesta dalla quale estrae i dati (numero transazione, oggetto, quantità e numero carta di credito); quindi controlla un file "magazzino.txt" che contiene le quantità residue per vedere se può soddisfare la richiesta. Se la richiesta può essere soddisfatta invia un messaggio di "ok" al client con il prezzo totale ed aggiorna il file magazzino.txt.
  Ad esempio, una possibile risposta può essere

  Ok 2 mouse 20

  Altrimenti invia un messaggio di rifiuto della richiesta di transazione.

- In ogni caso il server scrive la transazione in un file "log.txt" che conterrà tutte le transazioni effettuate

- Il client riceve la risposta e la visualizza e chiude la comunicazione.

Suggerimento. I file potrebbero avere la seguente struttura.

Magazzino.txt	log.txt
laptop 15 1000	12345 mouse 2 1234123412341234 Ok
mouse 30 10	33234 monitor 22 3333444455556666 rifiutata
monitor 20 15	…
…	