Sviluppo di un'applicazione Client-Server in Java

Camilletti Samuele
Boussoufa Yacine
Daniel Damiano Cerasuolo

13/01/2022

Indice

1 Cenni teorici

- **Socket**:
  Un socket può essere definito come il punto terminale di una comunicazione di rete. Per comunicare tramite rete, occorrono due socket che alternativamente si scambiano informazioni in input e/o output.

- **Server**:
  Nella sua definizione generale, è un’applicazione eseguita, la quale rimane in attesa di connessioni di client con cui comunica per fornire servizi.

- **Client**:
  Un client è un’applicazione realtime (ha un’esecuzione definita da un inizio e una fine), la quale si connette ad un server per usufruire dei servizi messi a disposizione da quest’ultimo.

- **TCP/IP**:
  E’ un acronimo che definisce i due protocolli necessari alla comunicazione ovvero TCP (Transfer Control Protocol) e IP (Internet Protocol).

- **Transfer Control Protocol**:
  E’ un protocollo di rete a livello di trasporto il quale si occupa di rendere affidabile la comunicazione tra mittente e destinatario. Il protocollo è orientato alla connessione (connection oriented) ovvero segue la procedura Setup-Comunicazione-Teardown.
  La fase di setup è utilizzata per instaurare in modo affidabile una connessione TCP ed è detta 3-way handshake, indicando la necessità di scambiare 3 pacchetti tra mittente ed host.
  Il client invia un segmento con flag SYN (utilizzato per la “sincronizzazione” o connessione) impostato ad 1 e un valore di Sequenza impostato su un numero casuale. Il server risponde con un segmento SYN/ACK con valore ACK Seqn del client + 1 e un valore di sequenza generato casualmente. Il client se riceve la risposta re-invia un ACK con i valori di sequenza ricevuti+1.
  La comunicazione come precedentemente spiegato è affidabile e utilizza dei pacchetti con un Sequence Number sequenziale i quali vengono “confermati” dal mittente con un pacchetto ACK con codice di sequenza aumentato di 1.
  Allo stesso modo ogni pacchetto inviato durante la fase di comunicazione è seguito da un ACK di conferma da parte del ricevente.
  Infine prevede una fase di Teardown (chiusura) affidabile basata su two-way handshake. Il protocollo TCP dispone inoltre del controllo di flusso e congestione. Il controllo di flusso è permesso dalla cosiddetta Sliding Window, ovvero il n. massimo di pacchetti inviabili senza ricevere un ACK. In caso di problemi di ricezione ripetuti la Sliding Window viene diminuita. Il controllo di congestione è permesso dal RTT (Round Trip Time) ovvero un contatore che misura il tempo necessario

per ricevere il pacchetto di conferma ricezione (ACK). Se questo numero dovesse superare una certa soglia viene diminuita la Sliding Window.

- **IP**:
  Il protocollo IP è un protocollo di rete che si occupa dell'instradamento ed indirizzamento dei pacchetti in rete.
  Il messaggio da trasmettere viene suddiviso in pacchetti i quali attraverso il protocollo IP arrivano all'indirizzo IP di destinazione attraverso i meccanismi di routing. Un pacchetto IP è composto da numerosi campi, quali l'identificativo del pacchetto, flag, Offset, il tempo di vita, Checksum, e infine gli Indirizzi IP:Porta (mittente/destinatario). I campi rilevanti ai fini del progetto sviluppato si limitano a quelli di Indirizzo IP mittente e destinatario e Porta. La porta indica il tipo servizio richiesto dal client e quello messo a disposizione dal server. La porta è un valore compreso tra 0 e 65535 di cui i valori da 0 a 1024 sono riservati ai servizi più diffusi quali FTP, HTTP etc. Il NAT (Network Address Translation) si occupa inoltre di creare una mappatura per la quale ad ogni servizio di un host in un'intranet corrisponde un IP esterno con la quale comunicare.

- **Thread**:
  Un thread (lightweight process) è una suddivisione di un processo in più istanze o sottoprocessi che vengono eseguiti "quasi in parallelo" sul processore. Possono essere anche visti come un singolo flusso sequenziale di controllo all'interno di un processo. Essi condividono uno spazio di memoria e indirizzi, sono più leggeri e facili da gestire, e in presenza di attività di I/O sono più performanti. Gli stati di un thread sono Created che si verifica immediatamente all'istruzione new, Runnable che indica che il thread è in esecuzione o in attesa, Not Runnable che indica invece in attesa di I/O o di un servizio e Dead ovvero quando il thread termina.

- **Multi-Threading in Java**:
  Il multi-threading, ovvero la tecnica che ci permette di avere più thread nello stesso processo, può essere implentata in Java attraverso due metodi: come sottoclasse della classe Thread o come classe che implementa l'interfaccia Runnable. Nel caso della classe Thread essa ha un metodo run() che deve essere ridefinito ed indica ciò che il thread esegue. L'oggetto deve essere allocato attraverso la parola chiave new, e deve essere avviato con il metodo start().

2 Analisi e Progettazione

E’ stato richiesto di progettare un software, ai fini dell’orientamento, in grado di mettere in competizione l’utente con la macchina attraverso un gioco interattivo. E’ stata quindi progettata una soluzione di gameplay efficace che potesse da un lato mostrare le caratteristiche di un’architettura client/server e dall’altro mettere in competizioni gli utenti del gioco. Per mettere a proprio agio l’utente è stata inoltre implementata un’interfaccia grafica user-friendly con la quale l’utente interagisce direttamente.

2.1 Modalità di gioco

L’utente dopo aver scelto il suo nickname in-game, viene posto di fronte a una serie di domande alla quale dovrà rispondere correttamente inserendo la risposta nell’apposito contenitore testuale. L’utente inizia la partita con un punteggio standard di 100 punti. Ogni domanda ha un suo valore in termini di punteggio ed un numero di tentativi massimi. Se l’utente indovinerà la domanda il punteggio salirà del valore di essa, se invece al termine dei tentativi non ci sarà riuscito il suo punteggio decrementerà del medesimo valore. Il termine della partita viene sancito nel momento in cui il giocatore esaurisce i suoi punti (0 punti) e il punteggio massimo raggiunto verrà registrato, insieme al suo nickname, in una classifica globale con tutti gli altri giocatori.

2.2 Architettura dell’applicazione

Al fine di realizzare un’applicazione che fosse indipendente dalla piattaforma su cui viene avviata e che potesse mostrare le caratteristiche dei socket, threading e oggetti è stata utilizzata un’architettura di comunicazione client/server.
Il server implementa la tecnica del multi-threading per consentire la connessione di più client connessi contemporaneamente. La connessione e il trasferimento dei dati per ogni thread è implementata attraverso l’IPC dei Socket i quali, in questo caso, lavorano sul protocollo TCP/IP.
Le informazioni che il server e i client si scambiano vengono racchiuse attraverso una classe condivisa Messaggio la quale contiene tutti i dati relativi al funzionamento del gioco (vedere progettazione ad oggetti).
Il server mette a disposizione questo servizio sulla porta 6076, la quale è stata definita in maniera standard per semplificare la connessione.
Di seguito il funzionamento dell’architettura sviluppata:


Server
Client
ServerSocket
In ascolto sulla porta
6706
Inserisce il Nickname
in Input
Connessione Socket
su IP:6706
Accetta la
connessione ed
alloca il Socket
Alloca un nuovo
Thread a cui passa
il Socket
In attesa del
Nickname
Invia il Nickname
Legge il nickname
In Attesa della
domanda
Estrae la domanda
casualmente dal file
e la invia
Legge la domanda
Loop fino
a che il
punteggio
non arriva
a 0
In Attesa della
risposta
Invia la risposta
In attesa del
responso
Legge, controlla la
risposta ed invia il
risultato
Legge il risultato
Chiude la
connessione ed
aggiorna il file della
leaderboard
Chiude la
connessione

2.3 Implementazioni di sicurezza lato server

per evitare un affollamento di troppe connessioni contemporanee è stato scelto preventivamente di limitare il numero di thread creabili a 10. E’ stata inoltre implementata una sleep di 5 secondi dopo l’allocazione di un thread per evitare flooding di richieste.

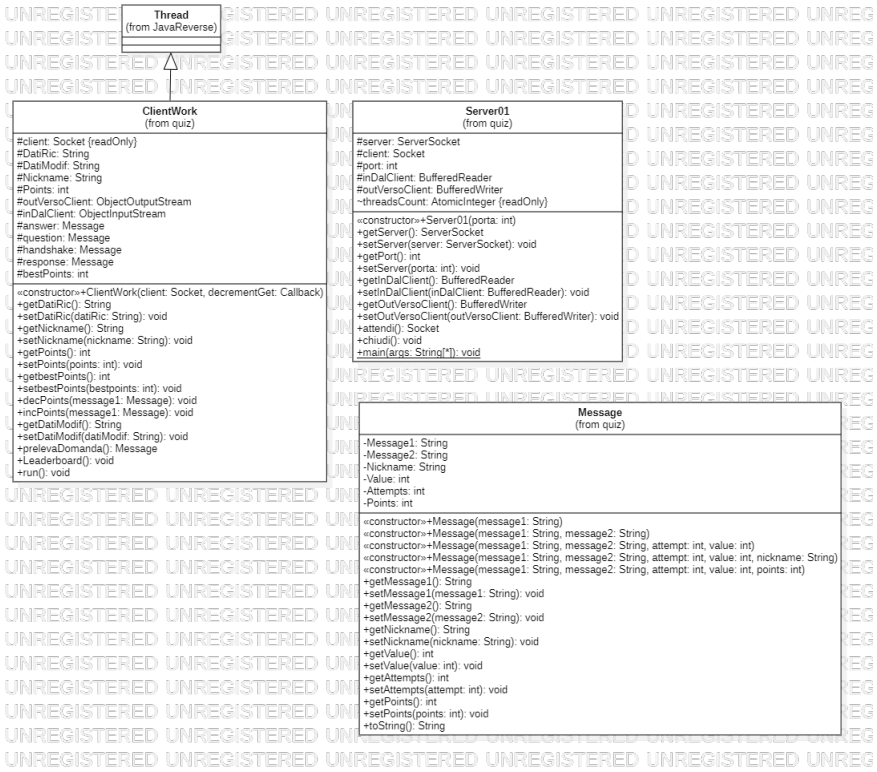
2.4 Interfacciamento con Web Server

Al termine della partita ogni thread accede in scrittura ad un file XML in cui scrive i migliori punteggi degli utenti. Tale file è letto dall’interprete php nel momento in cui viene richiesta la pagina web relativa al gioco, la quale è disponibile in rete attraverso un Web Server Apache in esecuzione sulla stessa macchina sulla porta 8291.

2.5 Requisiti non funzionali di prodotto

- RNF-001: Usabilità
  Il software (lato-client) presenta un’interfaccia grafica user-friendly realizzata tramite Swing in Java che illustra in maniera comprensibile il funzionamento del gioco.
- RNF-002: Portabilità
  Il software (lato-client) è stato provato in tutte le versioni di Windows superiori al 7. Il lato-server è invece multi-piattaforma e funziona in tutti i sistemi operativi Windows e Unix-like.
- RNF-003: Efficienza
  Il software (lato-client) richiede uno spazio minimo in memoria di massa di 20 MB e uno spazio minimo in memoria RAM di 200 MB.
- RNF-004: Affidabilità
  Il server sfrutta delle tecniche per prevenire attacchi DDoS descritte nella sezione 2.3.

3 Modello ad oggetti



4 Conclusioni

Il software dopo una lunga fase di test ha raggiunto una versione stabile completamente funzionante ed è scaricabile gratuitamente dal sito ufficiale o da GitHub. Il software è disponibile sotto la licensa Creative Commons.

5 Organigramma

***Yacine Boussoufa**: Modellazione e Analisi Client/Server, Implementazione lato client, Realizzazione Sito Web*
***Camilletti Samuele**: Modellazione e Analisi Client/Server, Implementazione lato server, Stesura documentazione, Implementazione interfacciamento Web.*
***Daniel Damiano Cerasuolo**: Modellazione e Analisi Client/Server, Stesura documentazione, Aggiunta delle domande*

6 Bibliografia

Sistemi e Reti Volume 2 (Luigi Lo Russo, Elena Bianchi).
Sistemi e Reti Volume 3 (Luigi Lo Russo, Elena Bianchi).
Moderni Sistemi Operativi (A. Tanembaum, H. Bos).

7 Sitografia

Thread in Java (Slide in classroom).
NetBeans official documentation.