

Documentazione software “Trivia Quiz”**Lista revisioni:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versione** | **Autore** | **Data** | **Commento** |
| **0.0** | Pierferdinando Valentini | 14 Gennaio 2022 | Versione iniziale |
| **0.1** | Pierferdinando Valentini | 21 Gennaio 2022 | Revisione dopo incontro del 21/01/2022 |

**INDICE**

[1. **Cenni teorici** 4](#_Toc94219166)

[**2. Analisi e progettazione** 6](#_Toc94219167)

[**2.1 Introduzione** 6](#_Toc94219168)

[**2.2 Architettura del software** 6](#_Toc94219169)

[**2.3 Modello logico** 7](#_Toc94219170)

[**2.4 Modello ad oggetti** 8](#_Toc94219171)

[**3. Testing del programma** 9](#_Toc94219172)

[**4. Conclusioni** 11](#_Toc94219173)

[**5. Bibliografia/Sitografia** 12](#_Toc94219174)

# 1. **Cenni teorici**

**- Client/Server**

Nelle reti Client-Server il computer che svolge la funzione del SERVER detiene in esclusiva parte del software come ad esempio il sistema operativo e/o gli applicativi.

I computer CLIENT sono elaboratori completi che hanno una propria CPU per le operazioni di elaborazione ma utilizzano il software messo a disposizione dal server.

Un server funge da punto di riferimento per gli altri calcolatori della rete, è un computer che mette a disposizione le proprie risorse (memoria, potenza di elaborazione, periferiche) agli altri PC della rete. I client sono computer dotati di memoria e capacità elaborativi locale che utilizzano le risorse che i server mettono a loro disposizione.

I vantaggi di questo tipo di modello consistono: nella scalabilità del sistema che può arrivare a collegare anche un numero elevato di computer, nella possibilità di gestire l’impostazione di sicurezza in maniera centralizzata, nella possibilità di ottimizzare l’utilizzo delle risorse.   
Lo svantaggio principale deriva dal fatto che l’implementazione e l’amministrazione del sistema richiedono maggiori competenze tecniche e personale specializzato.

**- TCP (Transfer Control Protocol)**

Il Transmission Control Protocol (TCP) è un protocollo di rete, che permette di gestire i dati provenienti (o a destinazione) dal livello inferiore del modello (cioè il protocollo IP). Una volta che i dati sono forniti al protocollo IP, quest'ultimo li incapsula in datagrammi IP, fissando il campo protocollo a 6. TCP è un protocollo orientato alla connessione, cioè permette a due terminali che comunicano di controllare lo stato della trasmissione.   
Definito nella RFC 793, su di esso si appoggia gran parte delle applicazioni della rete Internet.

**- Socket**

Un socket è un punto finale di un collegamento (“endpoint”) di comunicazione bidirezionale tra due programmi in esecuzione sulla rete. Un socket è associato a un numero di porta in modo che il livello TCP possa identificare l'applicazione a cui i dati sono destinati a essere inviati.

Un endpoint è rappresentato da una combinazione di un indirizzo IP e un numero di porta. Ogni connessione TCP può essere identificata in modo univoco dai suoi due endpoint. In questo modo puoi avere più connessioni tra il tuo host e il server. Il concetto di socket si basa sul modello Input/Output su file di Unix, quindi sulle operazioni di open, read, write e close; l'utilizzo, infatti, avviene secondo le stesse modalità, aggiungendo i parametri utili alla comunicazione, quali indirizzi, numeri di porta e protocolli.

**- IP**  
  
Un indirizzo IP (dall'inglese Internet Protocol address) è un numero che identifica univocamente un dispositivo detto host collegato a una rete informatica, formato da 4 numeri divisi da dei punti (Es. 192.168.1.1).

**- Porta**

le porte sono lo strumento utilizzato per permettere ad un calcolatore di effettuare più connessioni contemporaneamente verso altri calcolatori, facendo in modo che i dati contenuti nei pacchetti in arrivo vengano indirizzati al processo che li sta aspettando.

**- Java**

Java è un linguaggio di programmazione di alto livello e orientato agli oggetti. Esso fu progettato, nel 1995, specificatamente per essere il più possibile indipendente dalla piattaforma hardware di esecuzione (tramite compilazione in byte-code prima e interpretazione poi da parte di una Java Virtual Machine), permettendo la portabilità del software su diversi sistemi operativi.

**- DOS (Denial Of Service)**  
  
Il Denial of Service, nel campo della sicurezza informatica, indica un malfunzionamento dovuto ad un attacco informatico in cui si fanno esaurire deliberatamente le risorse di un sistema che fornisce un servizio ai client, ad esempio un sito web su un web server, fino a renderlo non più in grado di erogare il servizio ai client richiedenti. Il traffico in entrata inonda la vittima, rendendo impossibile da parte del server di erogare un dato servizio.

# **2. Analisi e progettazione**

## **2.1 Introduzione**

Lo scopo del software “Trivia Quiz” è quello di permettere all’utente di mettersi in competizione con il computer attraverso domande di carattere generale.  
Il sistema deve essere basato su un’architettura Client-Server dove il software Server rappresenta la macchina che somministrerà questionari ai giocatori, assegnando punti in caso di risposta affermativa altrimenti sottraendoli. Questo per poter mettere in competizione anche più giocatori presenti in rete o sulla stessa macchina.

Il software deve essere sviluppato con il linguaggio di programmazione Java.

## **2.2 Architettura del software**

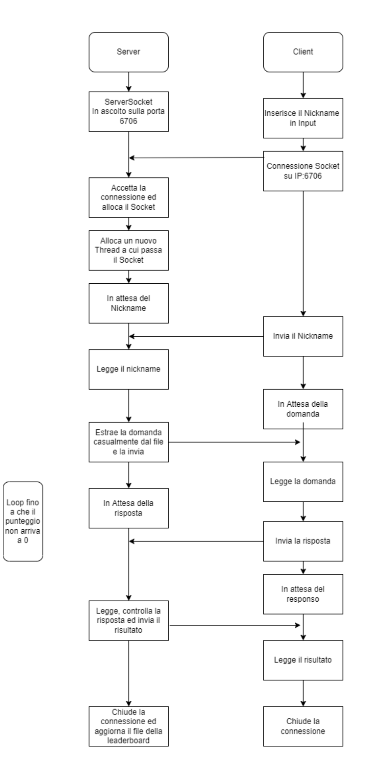
Un’applicazione client/server è un’applicazione distribuita divisa in due parti: un’applicazione server (back-end) che offre dei servizi e una applicazione client (front-end) che gestisce l’interfaccia con l’utente e permette di richiedere servizi al server.

Per realizzare un programma in grado di gestire una connessione client-server in java si utilizzano le classi disponibili nel package “java.net”.  
Per primo si va a definire la struttura dell’applicazione lato server: all’interno di tale software, viene definito un oggetto ServerSocket che si metterà in ascolto sulla porta inserita dall’utente, all’avvio del server, fino all’instaurazione di una connessione con un client.  
Il server deve poter accettare richieste da più client contemporaneamente per permettere l’attività ludica a diversi giocatori o sulla stessa macchina o connessi in rete. La multiplazione dei processi server è possibile estendo ServerThread con la classe Thread, realizzando un multithreading, in questo modo il server è in grado di accettare richieste da più client contemporaneamente.  
Una volta effettuata la connessione con il client, stampando i dati del client appena collegato, si fa partire l'esecuzione in multithreading chiamando il metodo run(). Nel metodo run vi è un ciclo infinito in cui il server è in ascolto sulla porta, ed ogni volta che riceve una richiesta crea i canali di comunicazione per poter comunicare con i client.  
Per ovviare la vulnerabilità da possibili attacchi DOS, il sistema server è stato dotato di un ciclo di controllo per il numero massimo di client che accedono contemporaneamente, in questo software non sarà accettato un numero di client maggiore di 10.   
Quando il server riceve una richiesta da un client crea una nuova istanza di un Socket per quel client, aprendo un canale di comunicazione con esso, effettuando il servizio per cui è predisposto, ovvero esporre questionari agli utenti.  
La connessione col client termina: in caso di errori, nel caso che il numero dei client collegati sia maggiore di 10, o quando viene letto un comando di terminazione.  
Una volta determinata la struttura del software lato server, si procede con la specificazione dell’applicazione lato client. Il client effettua una connessione con il server creando un nuovo oggetto Socket; tale oggetto avrà come parametri l’indirizzo del server a cui ci si andrà a connettere, e la porta che verrà inserita dal terminale al momento dell’esecuzione del programma client.

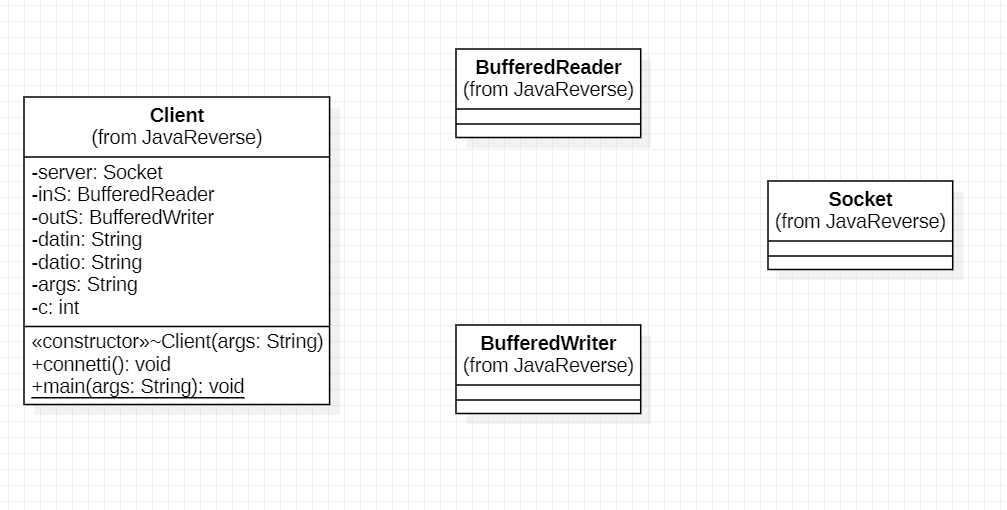
Quindi vengono creati i canali di comunicazione con il server ed a questo punto può incominciare la comunicazione. Una volta completata la comunicazione, il client disloca le risorse impiegate: chiude i canali di comunicazione e la connessione con il server, scrivendo sul file “leaderboard” il punteggio effettuato e il nickname del giocatore nel caso abbia digitato ‘si’ alla domanda per il salvataggio dei dati e in ogni caso, la connessione con il client viene terminata. Il server invece rimane in ascolto per eventuali altri client che si connetteranno al server.

# **2.3 Modello logico**

Il modello sottostante rappresenta il funzionamento del programma tramite una rappresentazione a diagramma a blocchi.



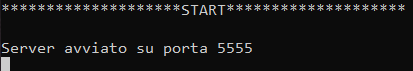
## **2.4 Modello ad oggetti**



# **3. Testing del programma**



Il server dedicato alla fornitura di quiz casuali ad un utente client, viene messo in ascolto sulla porta 5555, in attesa di una comunicazione con un client.



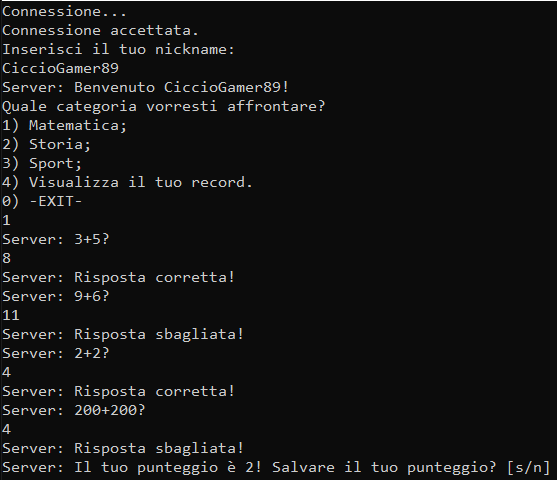
Il server è stato avviato e rimane in attesa di un giocatore.



Viene mandato in esecuzione il client che si andrà a collegare sulla porta aperta del server 5555.



La connessione è riuscita e il giocatore sta comunicando con il server. Così inizia l’attività ludica del quiz gestito dal server.



Una volta inserito il nickname, il server richiede all’utente su quale categoria verteranno le domande con cui vorrà cimentarsi. Gli verranno somministrate in modo casuale. Il client procede rispondendo alle domande e guadagnando punti ogni qual volta il cliente riuscirà ad indovinare la risposta corretta altrimenti gli verranno sottratti.

Una volta terminate le domande, all’utente verrà chiesto se salvare il punteggio appena effettuato, terminando la sessione oppure continuare con un’altra partita ricominciando da capo, dalla scelta della categoria di domande.

# **4. Conclusioni**

Il software dopo una lunga fase di test risulta funzionante e senza alcuna ambiguità a livello progettuale. L’applicazione è pronta per la sua commercializzazione e vendita al pubblica permettendo lo scaricamento dal sito di GitHub.

# **5. Bibliografia/Sitografia**

Hacker 7.0 (Stuart McClure, Joel Scamray, George Kurtz)

Dispensa “Architetture di rete” (Lugi Sacchi)

Oracle forum info