

06/05/2025

# RELAZIONE DI SISTEMI

**AUTORI: STUDENTI DELLA CLASSE – 4IB**

* Thomas Fanciullacci

RELAZIONE DI SISTEMI

## PAGINA RISERVATA AI DOCENTI

Relazione di Sistemi – Sistemi N.1 CLASSE 4IB

|  |
| --- |
| **Obiettivo dell’esperienza** |
| L’obiettivo principale di questo progetto è la realizzazione di un’applicazione software server, in grado di ricavare e elaborare informazioni anagrafiche delle scuole italiane. Le informazioni sono contenute dentro ad un file csv che verrà letto ed inserito nel server.  Il progetto è stato migliorato con l’ausilio di diversi tipi di Client per garantire la diversità di accesso al server e ai dati, rendendolo versatile. |

Assegnata il martedì 25 marzo 2025

Data di consegna: martedì 06 maggio 2025

Elenco studenti coinvolti nell’esperienza: Thomas Fanciullacci

Valutazione del docente firma del docente

Con riferimento alla griglia di valutazione:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **A** | **B** | **C** | **D** | **E** |
|  |  |  |  |  |

RELAZIONE DI SISTEMI

Attività svolta specificatamente da ciascuno studente: Thomas Fanciullacci – scrittura codice del server TCP e UDP, scrittura codice del client TCP, scrittura codice del client TCP con GUI, scrittura codice del client UDP e integrazione dell’uso del terminale sistemando delle problematiche.

|  |
| --- |
| **ELENCO STRUMENTI E TECNOLOGIE IMPIEGATE** |
| * **Visual Studio Code**: è stato l’editor del codice principale, usato per scrivere e modificare il codice del server e dei client. * **IntelliJ**: è stato l’editor che ho usato inizialmente, ma sostituito da Visual Studio Code * **GitHub**: è una piattaforma che viene usata da vari sviluppatori per salvare degli aggiornamenti del progetto, tenere traccia delle modifiche fatte e, in caso di errori, tornare indietro senza perdere il lavoro. * **Linguaggio di programmazione**: Java * **Sistema Operativo**: Windows |

RELAZIONE DI SISTEMI

## STRUTTURA DEL CODICE

*Il Server è stato diviso in più classi per rendere il codice più leggibile e facile da gestire. La classe principale si chiama “MainServer”, all’interno sono state definite le sottoclassi per gestire le connessioni al server.*

* ***Classe principale – MainServer****: all’avvio del server questa classe si occupa di dividere la connessione TCP con quella UDP avviando due Thread separati, in modo da permettere l’accesso ai dati in due modi diversi.*
* ***Sottoclasse - UDP\_Server****: implementa l’interfaccia “Runnable”, è possibile farla eseguire come un Thread separato. Si occupa di ascoltare le richieste di accesso ricevute dai Client che usano il protocollo UDP sulla porta 1771. Questa connessione è più veloce e meno affidabile e non causa il sovraccarico della rete, non si preoccupa dell’integrità dei pacchetti dati, né dell’avvenuta ricezione da parte del client. Per questo collegamento usiamo Datagram Socket perché supporta il dominio Internet dal protocollo UDP non istaurando una connessione diretta tra client e server, e non verificano l’arrivo del dato o il reinvio. Le classi che useremo per il progetto sono DatagramSocket e i dati sono impacchettati tramite la classe DatagramPacket.*
* ***Sottoclasse – TCP\_Server****: implementa l’interfaccia “Runnable”, è possibile farla eseguire come un Thread separato. Si occupa di ascoltare le richieste di accesso ricevute dai Client che usano il protocollo TCP sulla porta 1717. Questa connessione è più sicura e affidabile perché stabilisce un canale logico di comunicazione tra client e server finché rimane attiva la trasmissione o finché non viene chiuso il canale. La connessione garantisce la integrità dei dati, grazie a dei messaggi di “acknowledge. Il pacchetto dati viene diviso in frammenti per poi inviarli, questa tecnica sovraccarica la rete perché si inviano anche le informazioni di supporto. Il collegamento usiamo Stream Socket perché fornisce stream affidabili e ordinati e supportano TCP. Permettono di creare due canali separati tra client e server (uno per ciascuna direzione). Le classi che useremo per il progetto sono Socket (Client) e ServerSocket (Server).*
* ***Classe Server (logica principale)****: all’interno di questa classe sono presenti tutti i metodi necessari per effettuare la ricerca delle informazioni anagrafiche delle scuole. Sono presenti due costruttori, in base ai parametri inseriti la classe potrà definire una connessione TCP o UDP.*

*Le classi che gestiscono i Client sono state divise per emulare diverse situazioni:*

* ***Client****: gestisce i client TCP con l’output direttamente nel terminale della strumentazione.*
* ***ClientGUI****: gestisce i client TCP con l’aggiunta di una piccola GUI per l’invio dei comandi e la visualizzazione delle informazioni.*
* ***ClientUDP****: gestisce i client UDP con l’output direttamente nel terminale della strumentazione.*
* ***Terminale****: gestisce la comunicazione tramite protocollo TCP: usando il comando <telnet “indirizzo IP del server (127.0.0.1)” “porta del server (1717)”>. Quando viene eliminato un carattere, il terminale non lo elimina effettivamente, ma viene inviato il carattere “\b” che serve solamente per spostare il cursore del terminale. Il server gestisce questa problematica e ricostruisce il comando tramite il metodo* *changeCode(str): questo metodo crea una variabile temporanea che serve sa supporto per la ricostruzione del codice, con un ciclo continua ad inserire la lettera e se incontra il carattere “\b” cancella l’ultima lettera inserita.*

## INSTALLAZIONE

1. *Apri la bash.*
2. *Seleziona la cartella dove vuoi salvare il progetto*
3. *Clona questa repository:*

* *git clone <https://github.com/Fankyostro17/ProgettoScuoleItaliane.git>*

1. *Usa Visual Studio Codce o IntelliJ per aprire il progetto*
2. *Avvia il server*
3. *Avvia uno dei client (o più di uno)*

RELAZIONE DI SISTEMI

## DESCRIZIONE DEI COMANDI

|  |  |
| --- | --- |
| END | Chiude il client |
| STOP | Chiude il client e il server corrente |
| GET-COMUNE | Prende le informazioni di un rispettivo comune |
| GET-PROVINCIA | Prende le informazioni di una rispettiva provincia |
| GET-REGIONE | Prende le informazioni di una rispettiva regione |
| GET-CODICE | Prende le informazioni di un rispettivo codice |
| GET-ISTITUTO | Prende le informazioni di un rispettivo istituto |
| GET-TIPOLOGIA-ISTITUTO | Prende le informazioni di una rispettiva tipologia di istituti |
| GET-INDIRIZZO | Prende le informazioni di un rispettivo indirizzo |
| GET-CODICE-POSTALE | Prende le informazioni di un rispettivo codice postale |
| GET-TELEFONO | Prende le informazioni di un rispettivo numero di telefono |
| GET-FAX | Prende le informazioni di un rispettivo FAX |
| GET-EMAIL | Prende le informazioni di una rispettiva email |
| GET-EMAIL-PEC | Prende le informazioni di un rispettivo email PEC |
| GET-SITE | Prende le informazioni di un rispettivo sito web |
| GET-DIREZIONE | Prende le informazioni di una rispettiva direzione |
| GET-STATALE | Prende tutte le scuole statali |
| GET-PARITARIA | Prende tutte le scuole paritarie |

Scrittura del comando, esempio -> GET-COMUNE: VENEZIA (eccezioni -> END, STOP, GET-STATALE, GET-PARITARIA -> sono scritti da soli, esempio -> END).

## Osservazione e conclusione

Grazie a quello imparato in classe, non solo nella materia “Sistemi e Reti”, sono riuscito a realizzare questo progetto. L’obiettivo principale era creare un server in grado di fornire informazioni anagrafiche su diverse scuole italiane, e questo è stato raggiunto. Il funzionamento dei due protocolli di comunicazione, TCP e UDP, è diverso tra loro, e questo mi ha portato a trovare varie soluzioni per poter gestire entrambi: l’utilizzo di Thread per la gestione di più connessioni contemporaneamente, il sistema di ascolto delle richieste continuo nel server e anche la realizzazione dei vari Client. Grazie alle competenze acquisite nel corso di TPSIT sono riuscito a realizzare una piccola interfaccia grafica per il Client, anche se basilare. Inoltre, sono riuscito a far uso di Thread per dividere la parte dedicata all’ascolti e la parte di esecuzione del programma. Le nozioni apprese nel corso di Sistemi e Reti mi sono state utili per realizzare le connessioni TCP e UDP in modo efficace. Il progetto mi ha spinto a realizzare qualcosa molto vicino ad un software reale. Infine, grazie al corso di informatica ho avuto una base di programmazione solida. Anche se a scuola si studiano le materie separatamente, questo progetto ha riunito tutte queste conoscenze e facendomi comprendere l’importanza di ognuna di esse.