# Relazione di Progetto

**Titolo:** Applicazione Client-Server per la Consultazione delle Strutture Ricettive della Regione Piemonte  
**Autore:** Mattia Girometti

Questo documento illustra in modo approfondito l'analisi, la progettazione, l’implementazione e la verifica di un’applicazione client-server sviluppata in Java per la consultazione remota di strutture ricettive in Piemonte, con supporto ai protocolli TCP e UDP e gestione avanzata di dati CSV.

**1. Contesto e Obiettivi**

**1.1 Contesto del progetto**

L'applicazione nasce dall’esigenza di accedere facilmente e remotamente alle informazioni pubbliche sulle strutture ricettive della Regione Piemonte, contenute in un dataset ufficiale fornito in formato CSV.

**1.2 Obiettivi**

* Realizzare un sistema robusto e performante di interrogazione dati tramite rete.
* Implementare protocolli di comunicazione flessibili e affidabili (TCP e UDP).
* Garantire un’interfaccia intuitiva e documentata per utenti e sviluppatori.
* Fornire una soluzione scalabile e facilmente estendibile.

**2. Architettura del Sistema**

Il sistema si articola su due livelli principali:

* **Server**: Caricamento e gestione in memoria del dataset CSV, elaborazione dei comandi client e gestione della comunicazione di rete.
* **Client**: Interfaccia CLI Java con capacità multi-protocollo per interagire col server e consultare i dati richiesti.

**2.1 Diagramma Architetturale**

**2.2 Flusso funzionale**

1. Il server viene avviato, caricando e indicizzando il dataset CSV.
2. Il client avvia una sessione scegliendo il protocollo TCP o UDP.
3. Il client invia un comando (es. filtra comune: Torino).
4. Il server esegue il filtraggio e restituisce la risposta.
5. Il client riceve, ricompone (se UDP) e mostra il risultato.

**3. Specifiche Tecniche**

**3.1 Server**

* **ServerStrutture**: Entry point, carica il dataset, attiva socket TCP e UDP.
* **ThreadClientHandler**: Gestisce connessioni TCP con threading per supporto multi-utente.
* **GestoreCSV**: Parsing CSV con gestione avanzata delimitatori e virgolette.
* **Protocollo**: Gestione sintattica e semantica dei comandi inviati dal client.
* **StrutturaRicettiva**: Modello dati completo con tutti gli attributi della struttura.

**3.2 Client**

* **MainClientCLI**: CLI con prompt dinamico e comandi utente.
* **Connection (astratta)**: Definisce metodi comuni di comunicazione.
* **ClientConnection (TCP)**: Implementa connessione persistente tramite Java Socket.
* **UdpClientConnection (UDP)**: Implementa connessione stateless con frammentazione/rimontaggio pacchetti.

**4. Struttura del File CSV**

Il dataset utilizzato presenta i seguenti campi:

| **Campo** | **Descrizione** |
| --- | --- |
| ID | Identificativo unico |
| Denominazione | Nome della struttura |
| Tipologia | Categoria della struttura (Hotel, B&B, ecc.) |
| Stelle | Classificazione con stelle |
| Comune | Comune di ubicazione |
| Provincia | Provincia di riferimento |
| Indirizzo | Indirizzo completo |
| CAP | Codice postale |
| Telefono | Contatto telefonico |
| Email | Indirizzo email |
| SitoWeb | Sito web ufficiale |
| Marchi | Appartenenza a catene o marchi commerciali |

**Delimitatore:** ;, campi protetti da virgolette in presenza di caratteri speciali.

**5. Gestione del Dataset**

* **Parsing Avanzato**: Utilizzo di espressioni regolari per la tokenizzazione.
* **Normalizzazione**: Conversione campi vuoti in NON\_PRESENTE.
* **Indicizzazione**: Gestione interna zero-based, comandi utente con indici one-based.

**6. Comunicazione di Rete**

* **TCP**: Connessione orientata allo stream, gestione multithread per client simultanei.
* **UDP**: Connessione stateless con frammentazione in pacchetti da 1024 byte.
* **Logging dettagliato** per debugging e monitoraggio.

**7. Modalità di Utilizzo**

**Compilazione**

javac -d out src/server/\*.java src/client/\*.java

**Esecuzione**

java server.ServerStrutture

**Client TCP**

java client.MainClientCLI tcp localhost 1050

**Client UDP**

java client.MainClientCLI udp localhost 3030

**8. Conclusioni e Prospettive Future**

Il sistema realizzato soddisfa pienamente le specifiche iniziali con robustezza, scalabilità e facilità d’uso. Potenziali sviluppi futuri:

* Realizzazione di interfaccia grafica avanzata in JavaFX o web.
* Implementazione autenticazione utente e sicurezza tramite protocolli TLS.
* Supporto a formati dati aggiuntivi (JSON, XML).
* Integrazione API REST per maggior interoperabilità.