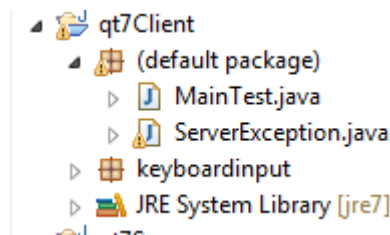


Client–Server (Socket)

Creare due progetti Eclipse distinti, [QTClient](#) e [QTServer](#). Distribuire le classi finora definite tra i due progetti.

CLIENT

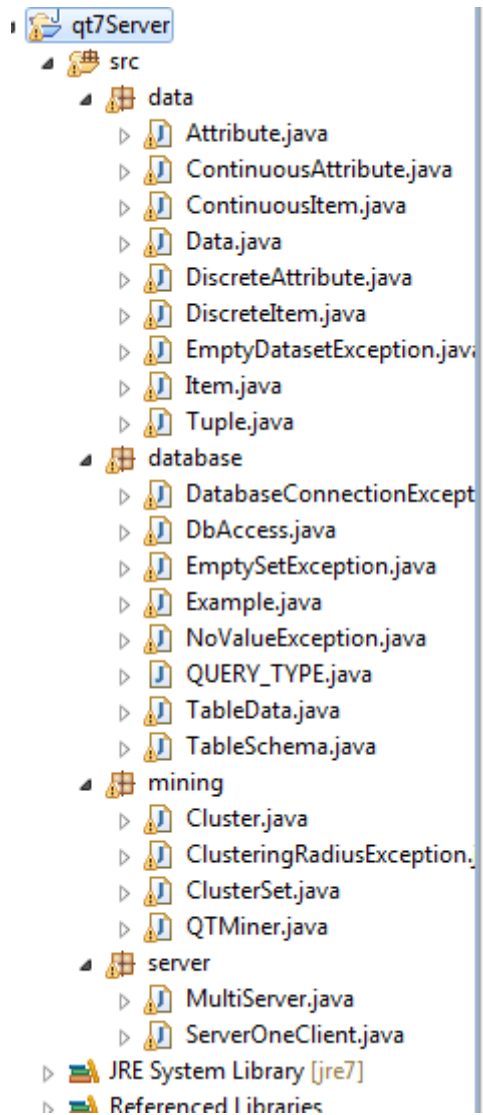


Il sistema client deve collegarsi al server tramite l'indirizzo e la porta su cui il server è in ascolto. Una volta instaurata la connessione l'utente può scegliere se avviare un nuovo processo di clustering o recuperare cluster precedentemente serializzati in un qualche file.

Includere la classe [MainTest](#) (**fornita dal docente**) che stabilisce la connessione al Server e, una volta avvenuta la connessione, invia e riceve messaggi, dipendentemente dalla scelta effettuata dall'utente. Attraverso un menu, l'utente del client seleziona la attività da svolgere, scoperta/lettura di cluster. Se la scelta è una attività di scoperta si invia al Server il [raggio dei cluster da scoprire, il nome della tabella di database, il nome del file in cui serializzare i cluster scoperti](#). Se la scelta è una attività di lettura si invia al Server [il nome del file in cui sono serializzati i cluster da recuperare](#). In entrambe le attività il cliente acquisisce il risultato trasmesso dal server o lo visualizza a video. Fare uso della classe Keyboard per l'input da tastiera

Definire la classe eccezione [ServerException](#) che è sollevata dal sistema server e trasmessa al client dallo stream di connessione. La eccezione è gestita dalla classe [MainTest](#)

SERVER



Il server colleziona le classi per l'esecuzione del QT (scoperta di cluster, (de)serializzazione).

- Definire la classe [MultiServer](#)

Attributi

`int PORT = 8080;`

Metodi

`public static void main(String[] args):` istanzia un oggetto di tipo [MultiServer](#).

`MultiServer(int port):` Costruttore di classe. Inizializza la porta ed invoca run()

void run() Istanza un oggetto istanza della classe ServerSocket che pone in attesa di crichiasta di connessioni da parte del client. Ad ogni nuova richiesta connessione si istanzia **ServerOneClient**.

- Definire la classe **ServerOneClient** estendendo la classe **Thread**.

Attributi

```
private Socket socket;  
private ObjectInputStream in;  
private ObjectOutputStream out;  
private QTMiner kmeans;
```

Metodi

ServeOneClient(Socket s) throws IOException: Costruttore di classe. Inizializza gli attributi socket, in e out. Avvia il thread.

void run() Riscrive il metodo run della superclasse Thread al fine di gestire le richieste del client.