**COMP90015: Distributed Systems Assignment 1**

**Multi-threaded Dictionary Server**

Xunkai HU

1145188

[xunkaih@student.unimelb.edu.au](mailto:xunkaih@student.unimelb.edu.au)

**Introduction**

This assignment aims to program a dictionary server that can be remotely accessed by multiple clients. The dictionary system follows a TCP protocol with a thread-per-connection architecture. A JSON file is used to store dictionary data.

**Structure**

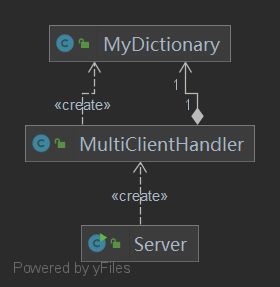
****

Figure 1, Server end

For the server end, the *Server* class keeps track of new clients connected to the server by running a loop. When a client sends a request to connect to an to the socket, the server socket accepts it and creates a new Thread for each client. The *MultiClientHandler* class handles the requests sent by the clients, such as searching the definition of a word in the dictionary, adding a new word with a definition to the dictionary, and removing or updating an exist word from the dictionary. Class *MyDictionary* manipulates data stored in the dictionary.json file via creating methods to convert JSONobject from / to HashMap.

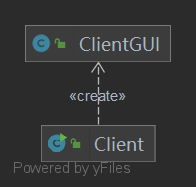


Figure 2, Client end.

As for the server end, class *Client* is the main class to send the request for the connection. An interactive GUI is implemented by the *ClientGUI* class. The class also handles messages to be sent and the responses received from the server, as long as the UI window is not closed.

**Protocols and Architecture**

The whole project is based on TCP protocol, the client connects to the server via sending a request. Once the connection is established, the server creates a thread for each client.

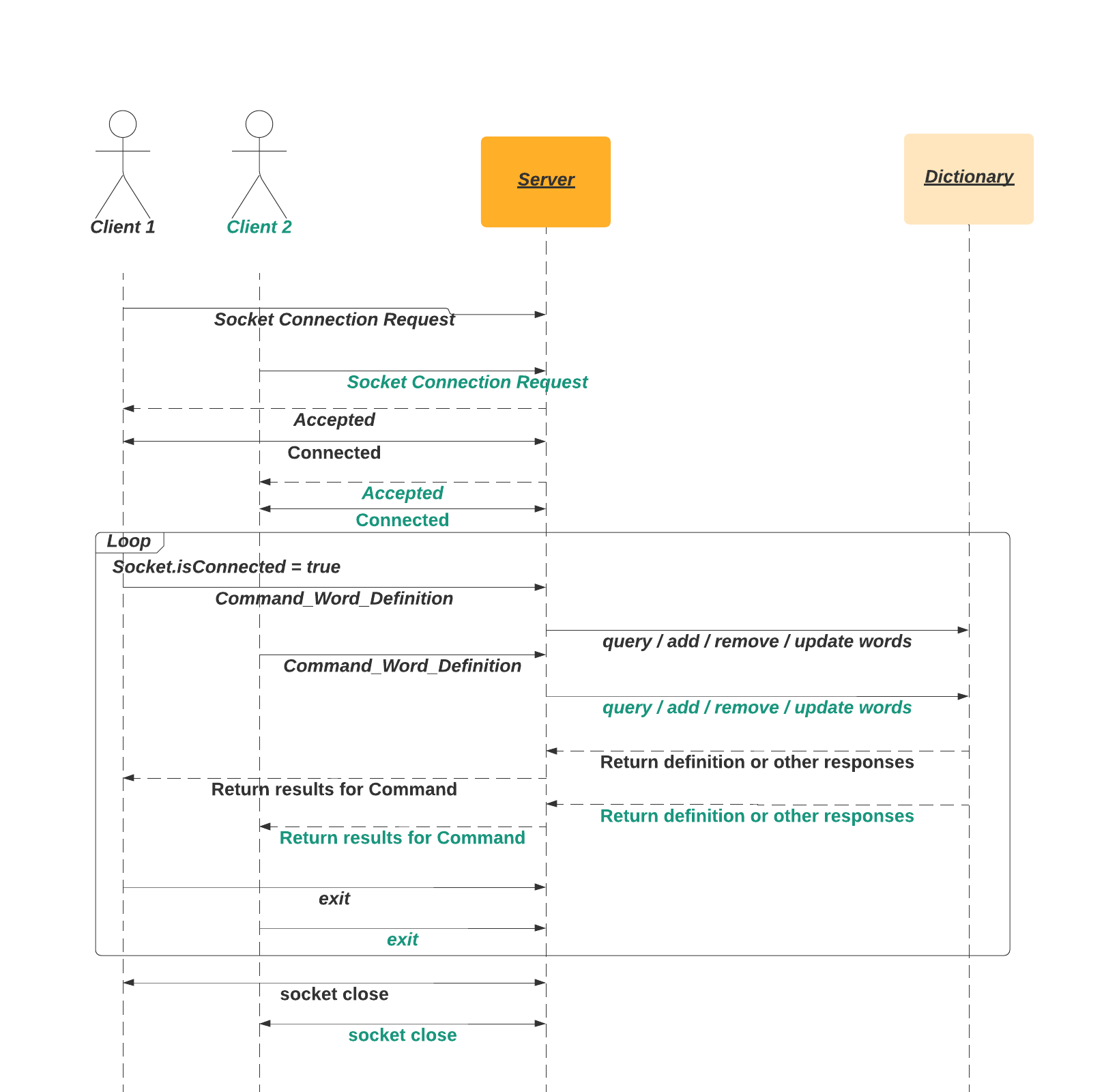


Figure 3, Interaction Diagram

As shown in fig. 3, in the functional loop, a String segment being sent to the server has a format of “Command\_Word\_Definition”. The segment then gets parsed at the server end, which invokes certain method from the *MyDictionary* class for that particular command.

**GUI**

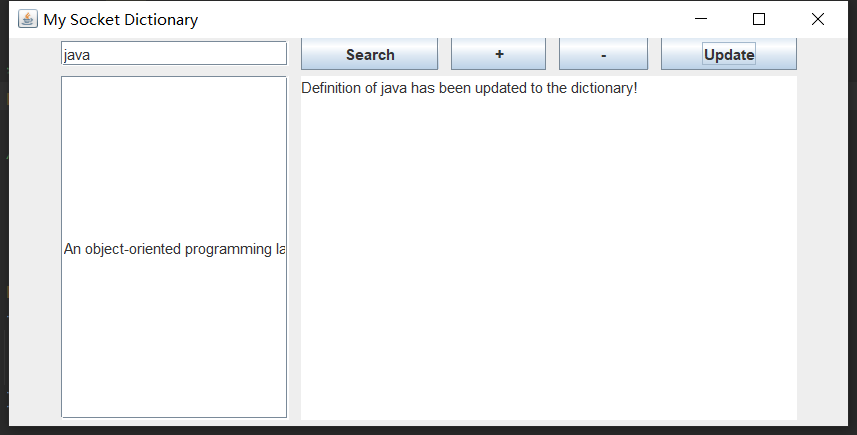


Figure 4, Dictionary Client

**Conclusions**

The given problem for designing a interactive socket-based multi-threaded dictionary has been accomplished, using TCP sockets and a thread-per-connection architecture. The program is capable to query the definition of words stored in the JSON file, and the dictionary can be customized by adding, removing or updating functions.