

COMP90015 Distributed Systems

A Distributed Scrabble Game

Student: Ruilin LIU (871076)

Student: Mingfeng LI (877347)

Student: Guanyu ZHU (884059)

Student: Junyu LONG (871689)

Table of Content

1. Introduction	4
2. Architecture	4
2.1 MVC Structure	4
2.2 Game Structure	4
2.3 System Structure	4
2.4 Design Diagram	5
3. Implementation Details	8
3.1 Network	8
3.2 Message Formats	11
3.3 Blocking Queue	11
3.4 Multithreading Design and Thread Pool	11
4. In-Scope Function Demo	11
4.1 Login	11
4.2 Invite	12
4.3 Start	13
4.4 Vote	15
4.5 Pass	15
4.6 Quit	16
4.7 Server Shutdown	16
5. Extra Functions	16
5.1 Leave Function	16
5.2 Drag Function	17
5.3 Status Constraint	17
5.4 Multiple Teams Allowed	17
5.5 Double-Checked Locking Singleton	17
5.6 Security	17
5.7 Quick Login Function	18
6. Conclusion and Future Work	18

7. Appendix.....	19
-------------------------	-----------

1. Introduction

This scrabble game is a multiple-players game that users can use different computers to play online game. There are several functions implemented in this game, such as invitation, vote and login. This report will discuss the architecture, implementation details and extra function about this game as well as show the process of operating the game. The important system architecture design is to separate services into different layers, and layers communicate via blocking queue. Therefore, each layer only processes their own logic.

2. Architecture

2.1 MVC Structure

To implement efficient code reuse and parallel development, the Model-view-controller model is used in the project. The used dynamic data structure is user list, managing the data, logic and rules of the users. And the GUIs are the view, representing the message to the users and server. Besides, both client and server controller are operated to accept input and convert it to commands for the model or view.

2.2 Game Structure

The game structure is mainly separated into two core parts, namely client and server part. The client part is responsible to presentation logic and some simple input logic, while the server part focuses on game logic operation and data management. For example, when a user is logging in, if the IP address or port is wrong, the client part will response this mistake to the user by itself. But, managing the game process is the duty of the server part. It needs to analyze and manage the data information and return some relevant message to the client part and let the client part present the update information about the board.

2.3 System Structure

The system structure can be divided into four section, namely client, model, protocol and server.

Client: The client section is provided to users, including game GUI, connection network and client control center. The control center uses connection network to transfer message between GUI and the server.

Model: The data structure used in this project is the list of users. So, in model, there are two parts, Users and Player. Meanwhile, Player list is encapsulated in Users list.

2.4 Design Diagram

Figure 2-1 The overview of UML

2.4.1 Server Control Center

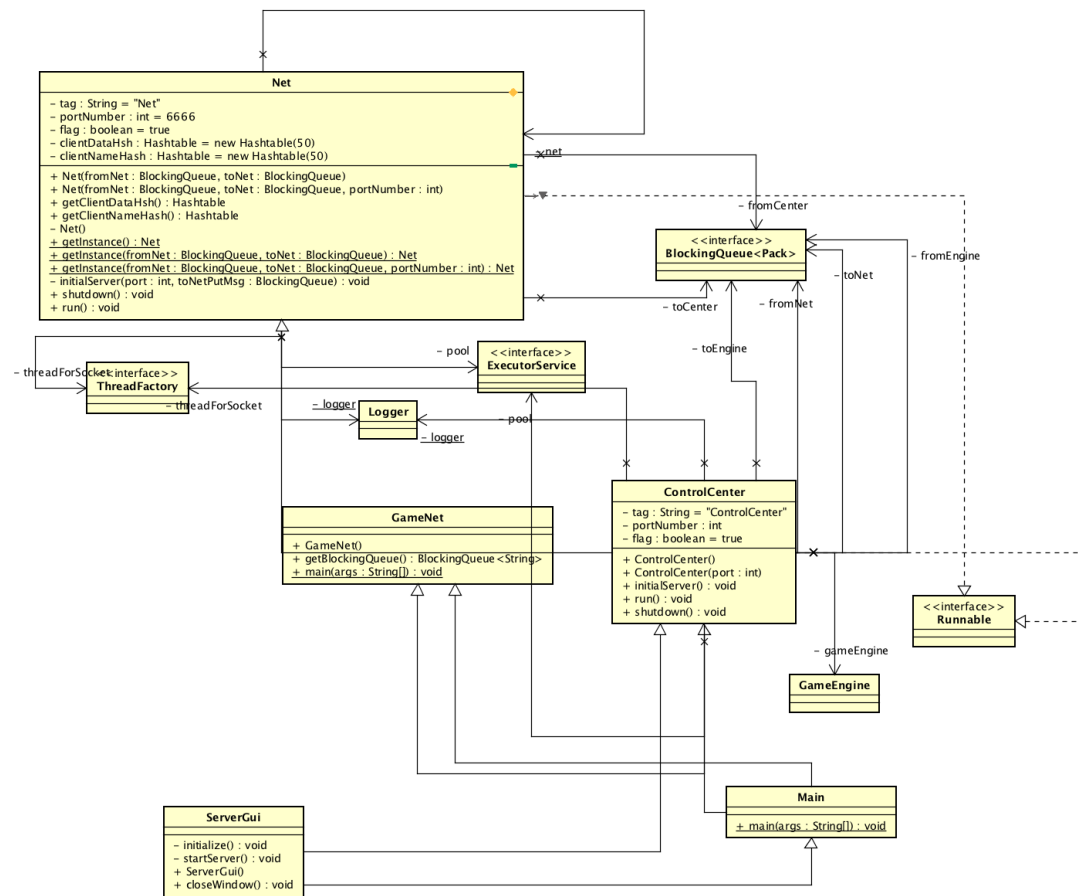


Figure 2-2 The server control center

2.4.2 Client Control Center

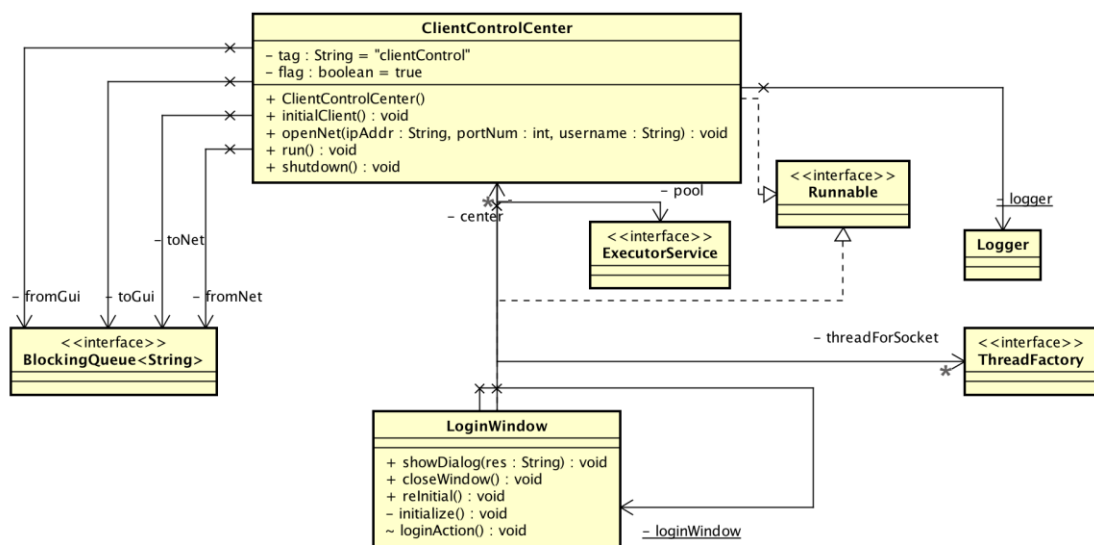


Figure 2-2 The client control center

2.4.3 Client GUI

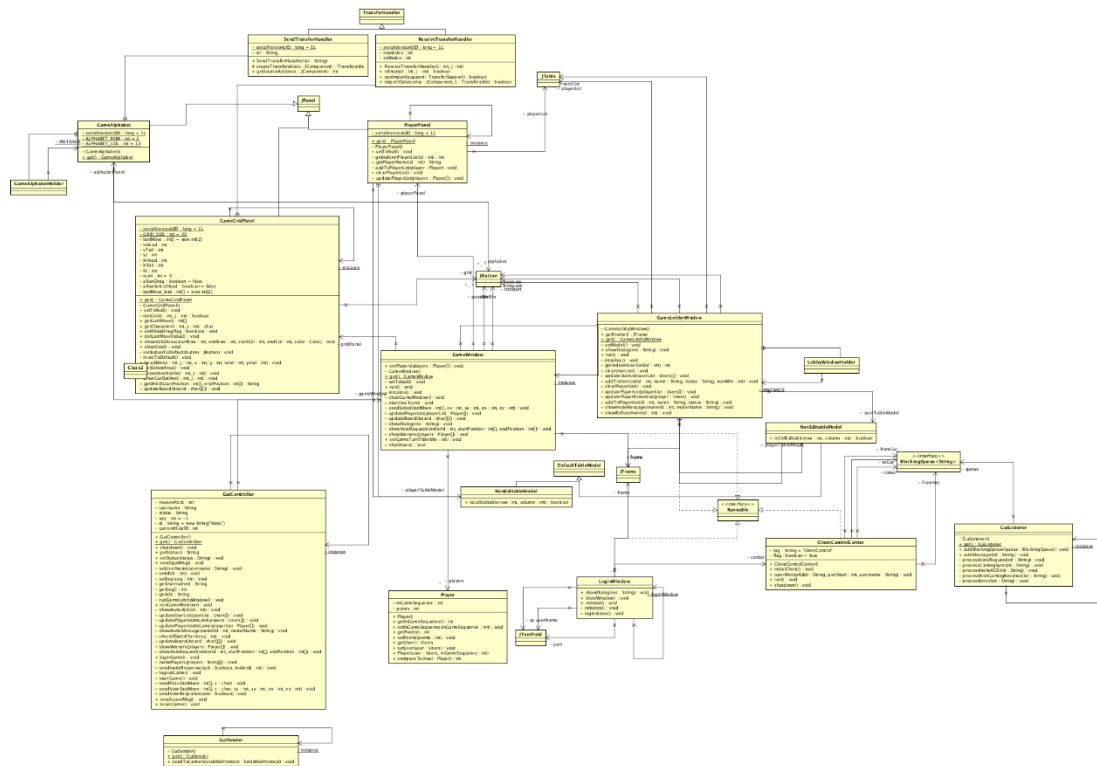


Figure 2-3 The client GUI

2.4.5 Sequence Diagram

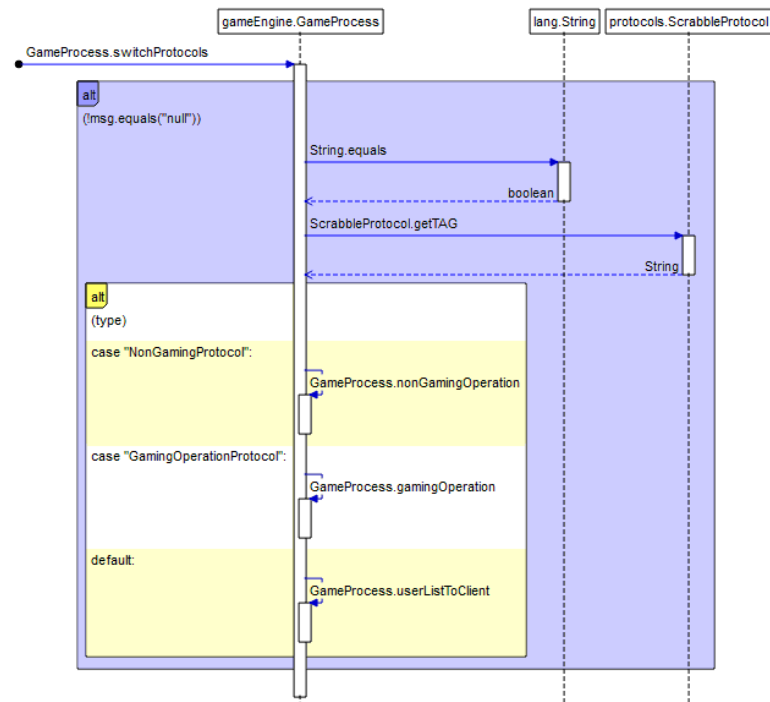


Figure 2-4 The Game Engine

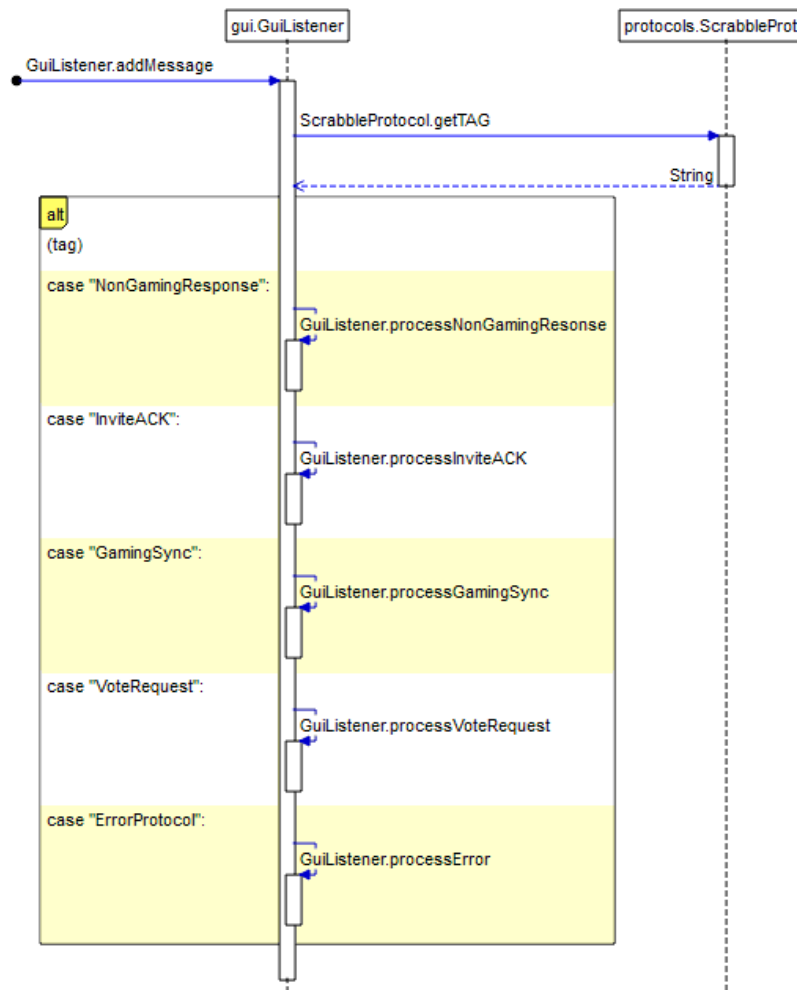


Figure 2-5 The Client GUI Listener

3. Implementation Details

3.1 Network

All messages of different protocols will be further packed by “Pack” class. It contains three fields, namely user ID, recipient and message.

1. User ID denotes whom a packet in transmitting is for or where this packet comes from.
2. The recipient filed records destination identity number. In addition, this variable is implemented in an Integer array. Firstly, “NetSendMsg” class checks whether the value of the variable is null. Then, if size of the array is larger or equal than one, the “sendToPeer” method will be called into a for loop for sending the message to each

aimed client. Furthermore, if the value of recipient variable is null, the “NetSendMsg” will check the value of user id.

3. According to the value of user id, if user id equals to zero, then the message will be sent to the all clients. Otherwise, the “NetSendMsg” class will called the “sendToPeer” to send message to the aimed destination.

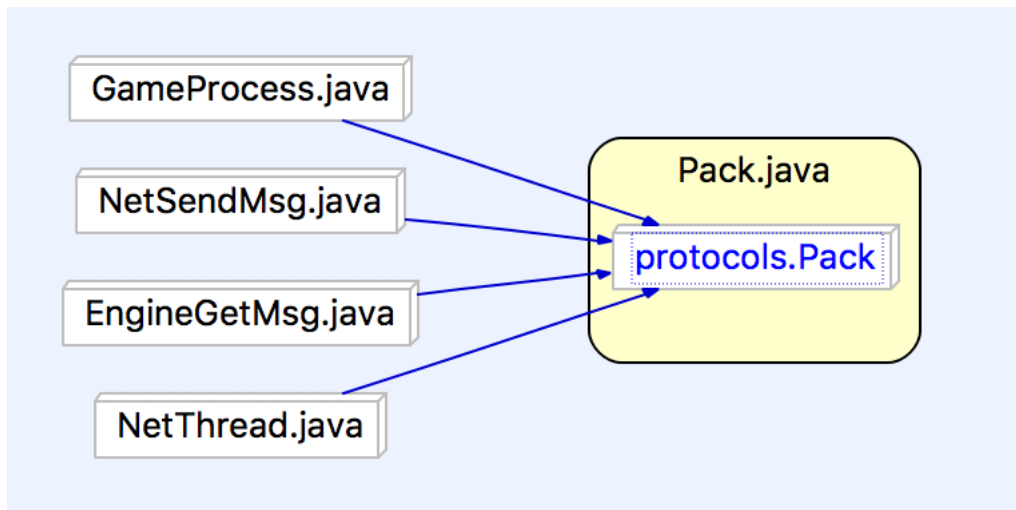


Figure 3-1 Protocol Structure

3.1.1 Communication Protocols

The message exchange protocols fall into three categories as a whole, and all of them extend the same super class called “ScrabbleProtocol”. The overview of protocol package structure is as the following figure shows:

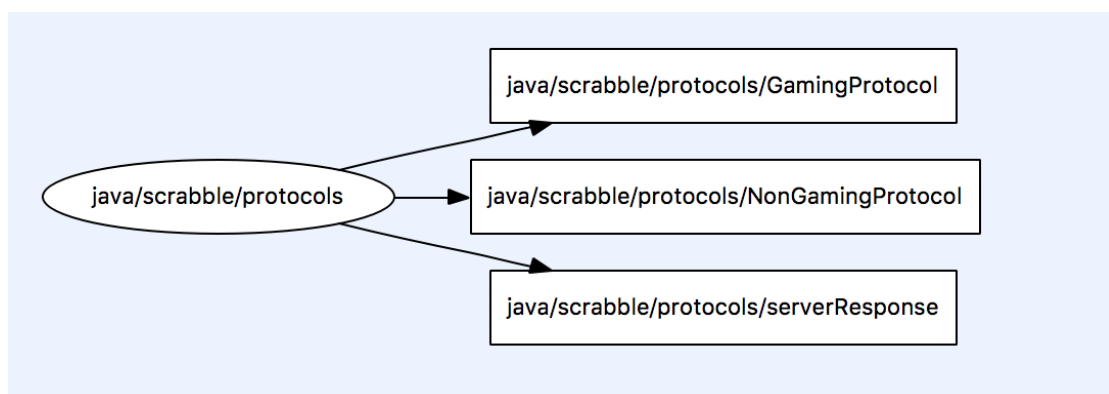


Figure 3-2 Protocol Structure

The “GamingProtocol” and “NonGamingProtocol” are exclusively designed for client-to-server message transmission process. And they are responsible for transmitting all game-related (i.e. vote, pass, etc.) and game-unrelated commands (i.e. login, logout,

invite, etc.) respectively. On the other hand, “serverResponse” refers to the transmission of responses from server. There are four protocols defined in this package as shown below:

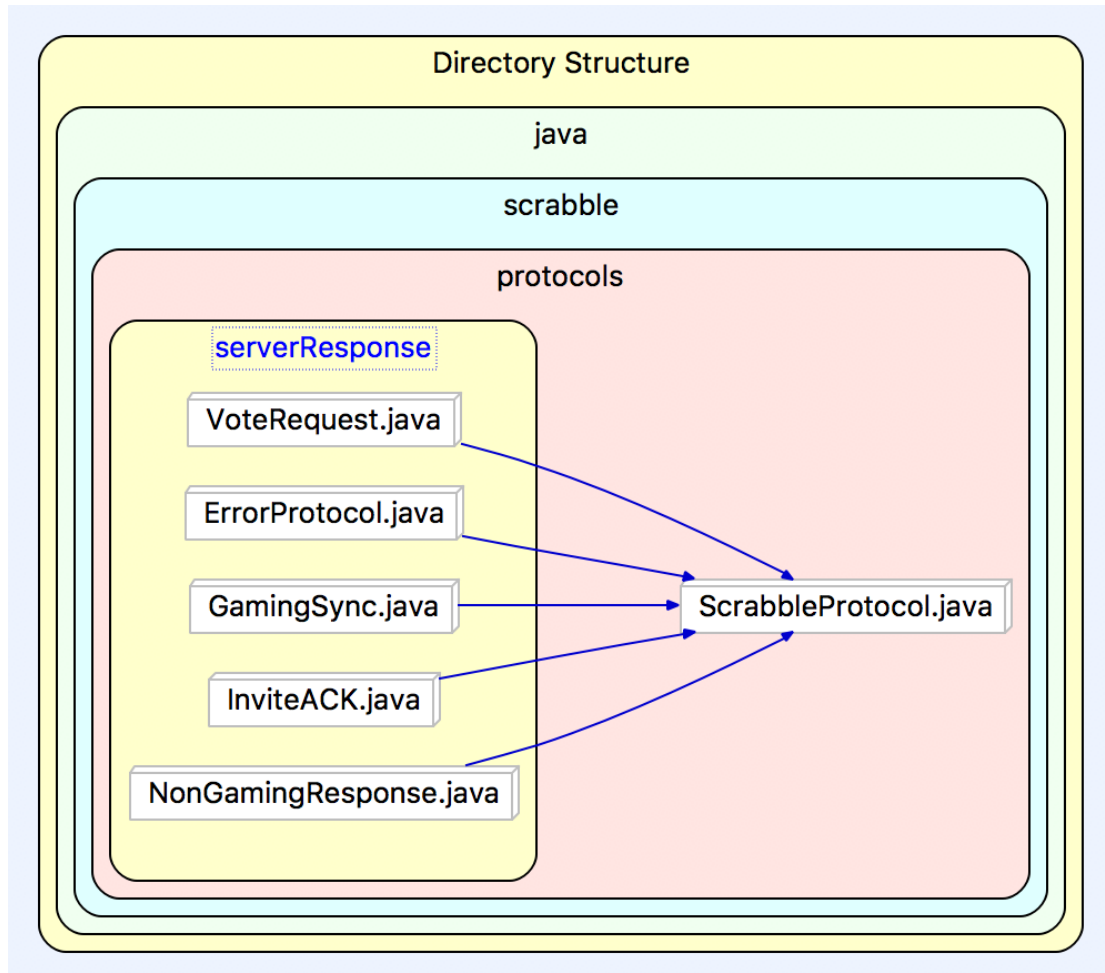


Figure 3-3 serverResponse Package Structure

To put it in detail,

- “VoteRequest”: It is used during an active game when a certain player intends to initiate a vote for their newly-made crossword puzzles. Once a message of this type is received at the client side, a window will pop up and wait for voting.
- “ErrorProtocol”: Error messages of different types will be encapsulated by this protocol and then send back to clients. For example, clients will receive login errors if they try to login with an existing username.

- “GamingSync”: During an active game, all game-related parameters and variables are processed and stored at the server side and then may partially packed with this protocol for gaming status syncing.
- “InviteACK”: It is used by the server to transfer users’ responses for teaming invitations from other users. Also, it is received by all members of a certain team if team status changes.
- “NonGamingResponse”: Server will respond to all game-unrelated requests from clients using this protocol, such as login, user update and so on.

3.2 Message Formats

The messages are parsed into JSON strings for transmission. The JSON jar file (fastjson-1.2.49.jar) used in this project is developed by Alibaba. It is a lightweight Java library used to efficiently convert JSON strings to Java objects and can also be used to convert a JSON string to an equivalent Java object.

3.3 Blocking Queue

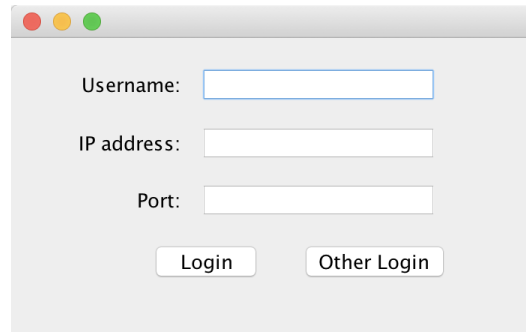
The blocking queue method is used to help transfer message. This method is thread safe when putting or taking instances. And when the producing thread put new objects into the queue, the queue has a limit in the quantity of objects. If the blocking queue reaches this limit, the producing thread is blocked until a consuming thread takes an object out of the queue.

3.4 Multithreading Design and Thread Pool

Both client and server are designed into multithreading structure. For example, in the server, each connection from client is allocated a single thread. Particularly, thread pool is to manage all the threads.

4. In-Scope Function Demo

4.1 Login

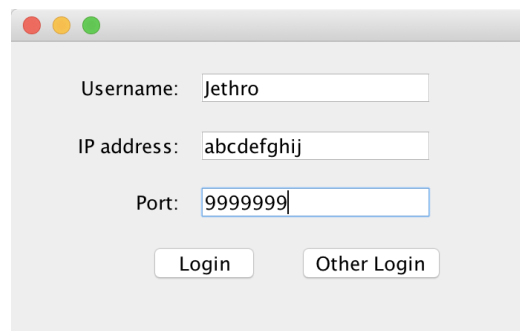


A login form window with a title bar containing three colored buttons (red, yellow, green). The form contains three input fields: "Username:" with a text box, "IP address:" with a text box, and "Port:" with a text box. Below the input fields are two buttons: "Login" and "Other Login".

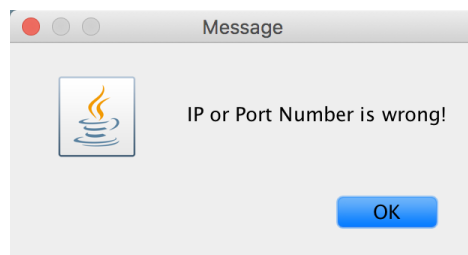
Users can login to the game lobby by entering valid username, correct IP address and port numbers. There is also a quick-login method to implement the same login function, which is discussed in Section 5.

For possible illegal inputs, the client GUI will respond as shown below:

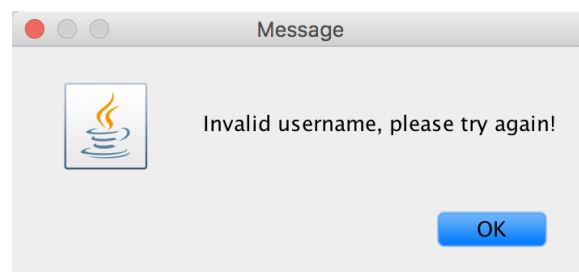
- Wrong IP address or Port number:



A login form window with a title bar containing three colored buttons (red, yellow, green). The form contains three input fields: "Username:" with the text "Jethro", "IP address:" with the text "abcdefghij", and "Port:" with the text "9999999". Below the input fields are two buttons: "Login" and "Other Login".

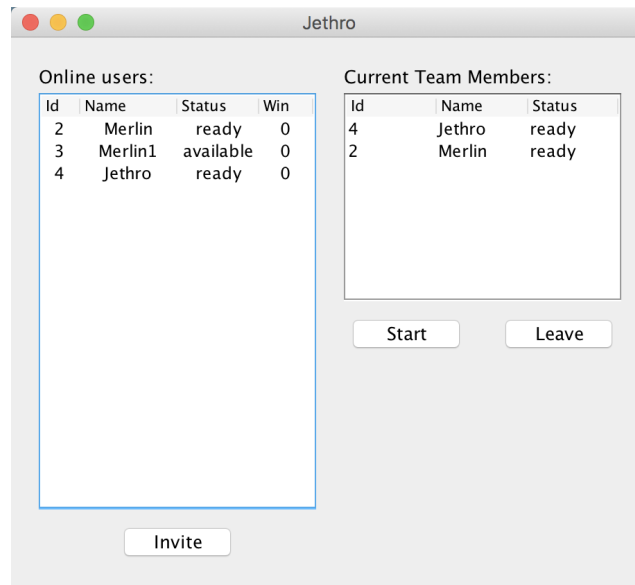


- Empty username field

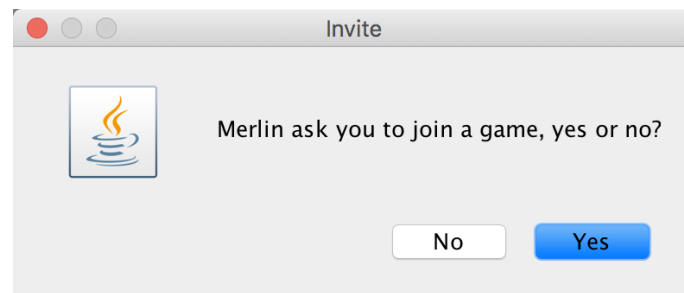


4.2 Invite

Once log in successfully, users have access to team with other online users as long as they are available or accept invitations from others.



- When received invitations from others



- When an invitation has been rejected

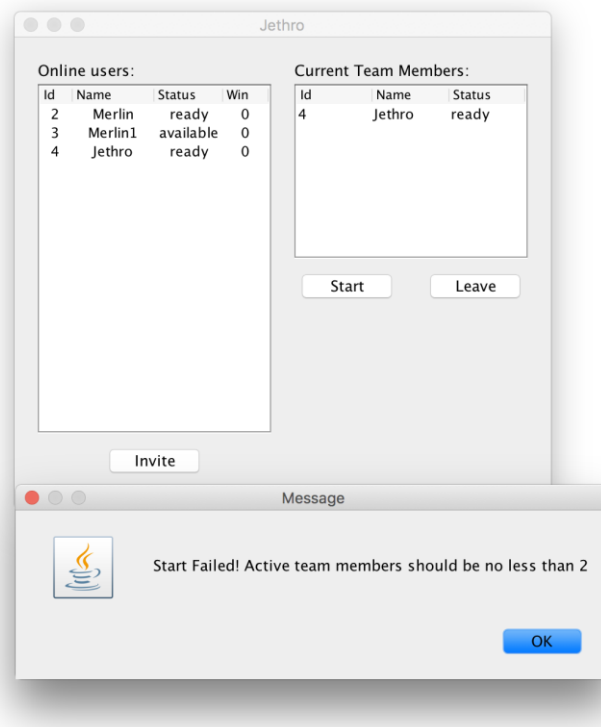


However, if accepted, the user will be added to the inviter's team and then get a whole list of current team members. So, does other members in the team.

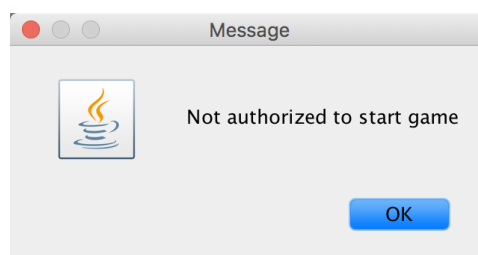
4.3 Start

The team host may want to start the game when there are enough members in the team.

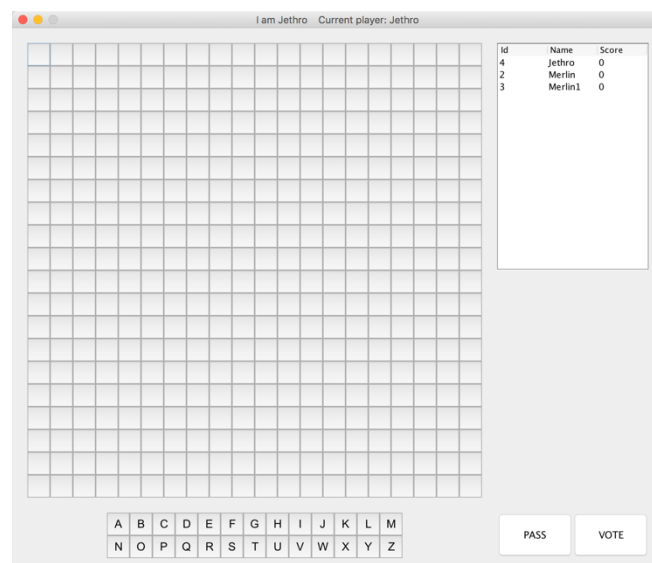
If the "start" button was clicked and the team size is less than 2, it will show:



The game engine allows only one game to be in progress and only the team leader can ask to start. Otherwise, there will be a pop-up like this:

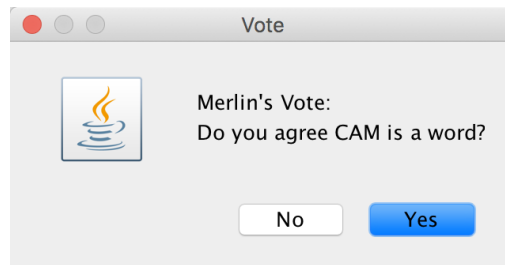


Once a "start" is successful, a game window will show up:

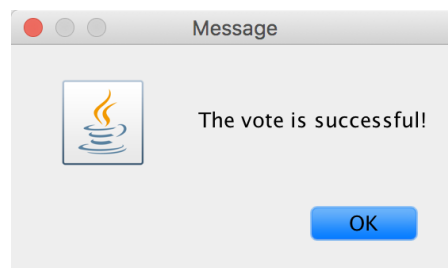


4.4 Vote

During a game, a player may feel like starting a vote for the crossword and click on the “vote” button after placing a new brick on the game board. All active players will then have access to choose “yes” or “no” for the given word on the popping-up:

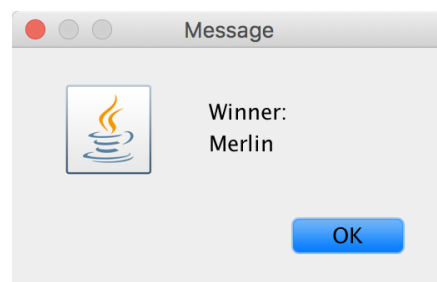


It is worth mentioning that a vote is successful only when the majority of players (>50%) agree on the word. Any successful vote will be notified by the following window:



4.5 Pass

A player can say pass after placing a brick or just skip his turn by clicking the "pass" button. If a player clicks "pass" after placing, it means he gives up the right to vote. The brick will remain on the game board and simply start the next turn. Moreover, when all players pass their turns, the game will end and show the winner in a dialog.

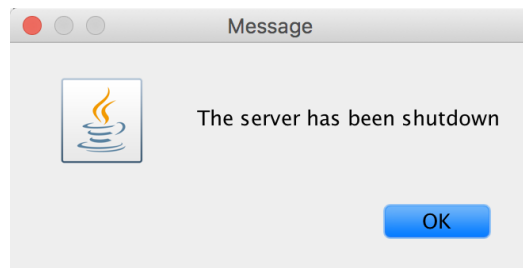


4.6 Quit

Players are able to quit the game at any time. It will lead to end the game, and the remaining players with the highest score will win the game. The result shows in a dialog as same as above.

4.7 Server Shutdown

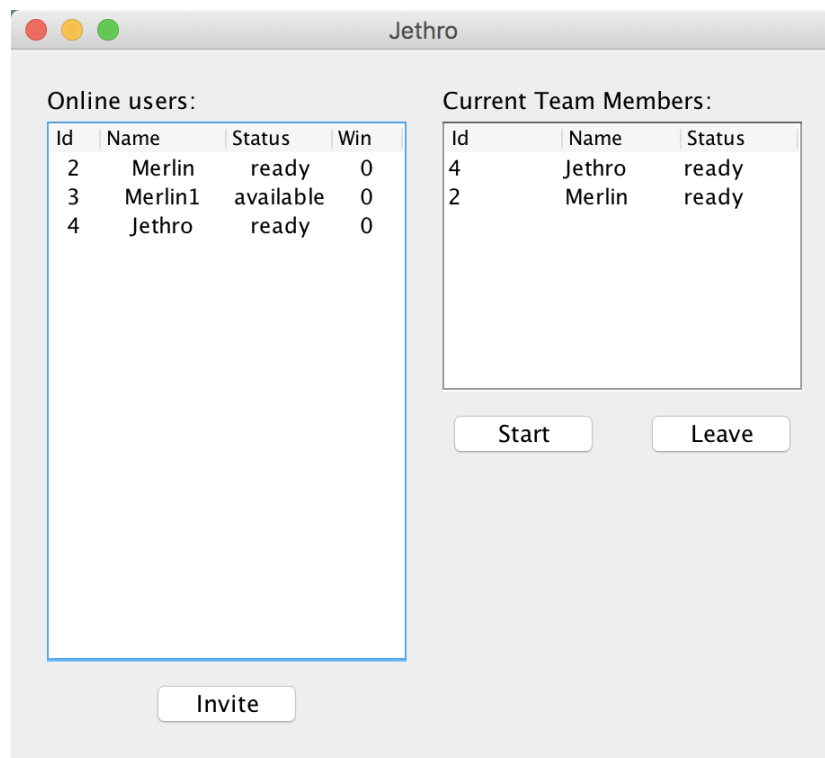
If the server is shutdown, at any time, all online users (including in-game users) will get a dialog window:



Then, the client process will be terminated gracefully after clicking on “OK”.

5. Extra Functions

5.1 Leave Function



Any user who is already in a team has access to leave for another team. There are two cases in “leave” function.

- Team members leave: the one who wants to leave will be removed from their current team member list and be reset to “available” status so that he or she is able to join another team.
- Team host leaves: the team will break up, all members will be removed from their own team member lists and their status will be reset to “available”.

5.2 Drag Function

This scrabble game implements the drag function, which means that the players can drag one letter each time to the board instead of inputting the letter through the keyboard.

5.3 Status Constraint

In this scrabble game, users have three status in the lobby, which are “available”, “ready” and “in-game”. And each status has different constraint. For instance, if some users start to play game, their status will be set as “in-game”, and when they end the game and return to the lobby, their status will be changed to “available”.

5.4 Multiple Teams Allowed

Although users only start a game at one time, they still be allowed to have multiple teams at the lobby. Besides, when the inviter has entered the team list, he/she can still invite the users whose status is “available” in the lobby.

5.5 Double-Checked Locking Singleton

```
synchronized (GameLobbyWindow.get()) {  
    gameWindow.updatePlayerList(playerList);  
}
```

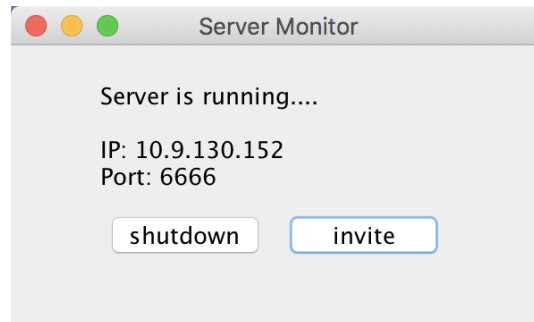
Through using the double-checked locking singleton, this game system double-checks whether the variable is initialized or inside the synchronized block. Therefore, it helps to prevent compilers from doing their own optimizations.

5.6 Security

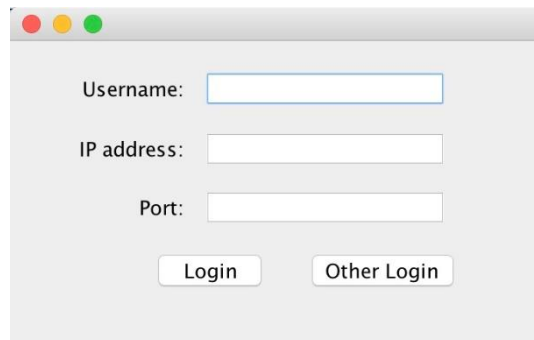
For security section, this game uses symmetrical encryption, namely Base 64, to encrypt the message transferred in the process of operation. Therefore, all message cannot be view directly and the game system can improve the relevant security to some extends.

5.7 Quick Login Function

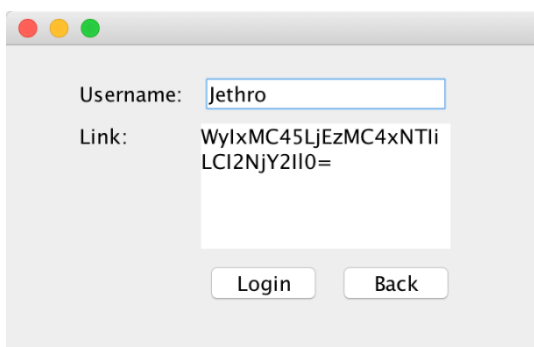
Instead of inputting IP address and port number, users can paste an invite link to login. By clicking the "invite" button on Server GUI, an invite link will be copied to clipboard.



After receiving the invite link, users may click "Other Login" to experience the Quick Login Function.



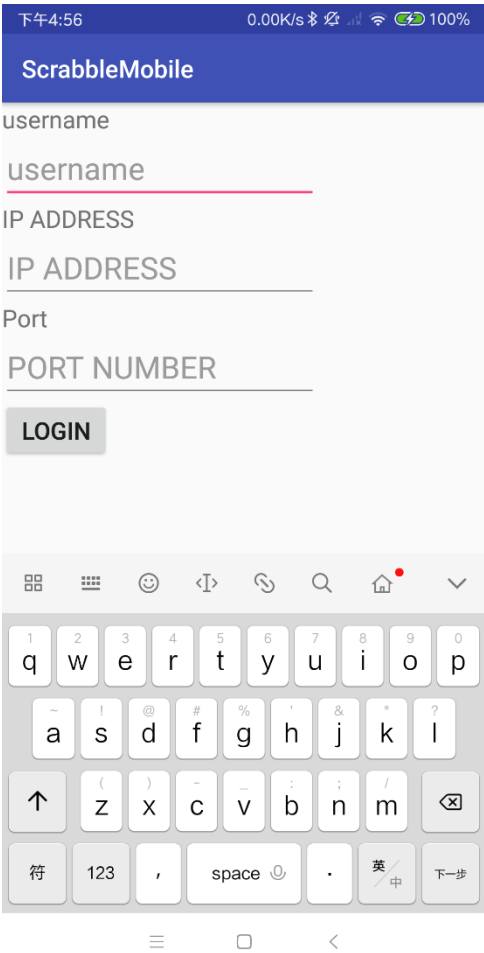
Now, users only need to provide a username. If users click the "Back" button, it will return to the normal login window.



6. Conclusion and Future Work

This scrabble game has implemented all relevant requirements and use several techniques, such as multithreading, thread pool and blocking queue to solve the blocking message error and keep the connection network stable. Meanwhile, this

scrabble game uses some security mechanisms to protect the message transferred in the process of game. However, there are several works can be done in the future, such as different teams can start different games simultaneously and the whole game system can be operated in the mobile phone.



7. Appendix

Mingfeng Li	Ruilin Liu	Junyu Long	Guanyu Zhu
Multithreading	Client GUI	Game Engine	Client Net
Server Net	Client Game	Server Relevant	Testing
MVC Design	Client Logic	Fixing Bugs	Fixing Bugs

Table of Individual Reflection