Wentworth Institute of Technology

2D Chess

Project Design Document

Team: Indecisive

Tiffany Phan and Yuliya Smilyanova

[phant9@wit.edu](mailto:phant9@wit.edu) and [smilyanovay@wit.edu](mailto:smilyanovay@wit.edu)

GitHub Repo: <https://github.com/smilyanovayatwit/2D_Chess>

**Table of Contents**

1. Introduction……………………………………………………………………………………………………………………………..3

2. Application Features…………………………………………………………………………………………………………………3

3. Application Architecture…………………………………………………………………………………………………………..3

4. System Design………………………………………………………………………………………………………………………….3

5. Plan………………………………………………………………………………………………………………………………………….6

6. References……………………………………………………………………………………………………………………………….6

**1. Introduction**

2D Chess is a console network application of the popular two player board game, chess, consisting of chess server and two clients. Developing this application allows us to practice and play around, while mastering the concepts of socket programming and multithreading. It is also a great way to understand how the game of chess is played. One of the unique features of our application is being able to choose different colors for the chess board, instead of the original black and white.

**2. Application Features**

* Allow client to join a game
* Allow client to choose a color (white or black) if they are the first one, otherwise be given the remaining color
* Allow client to make a move
* Allow client to leave the game
* Server broadcasts: client has joined, client’s color, which client has won, client has left (these are part of other features explained in system design)

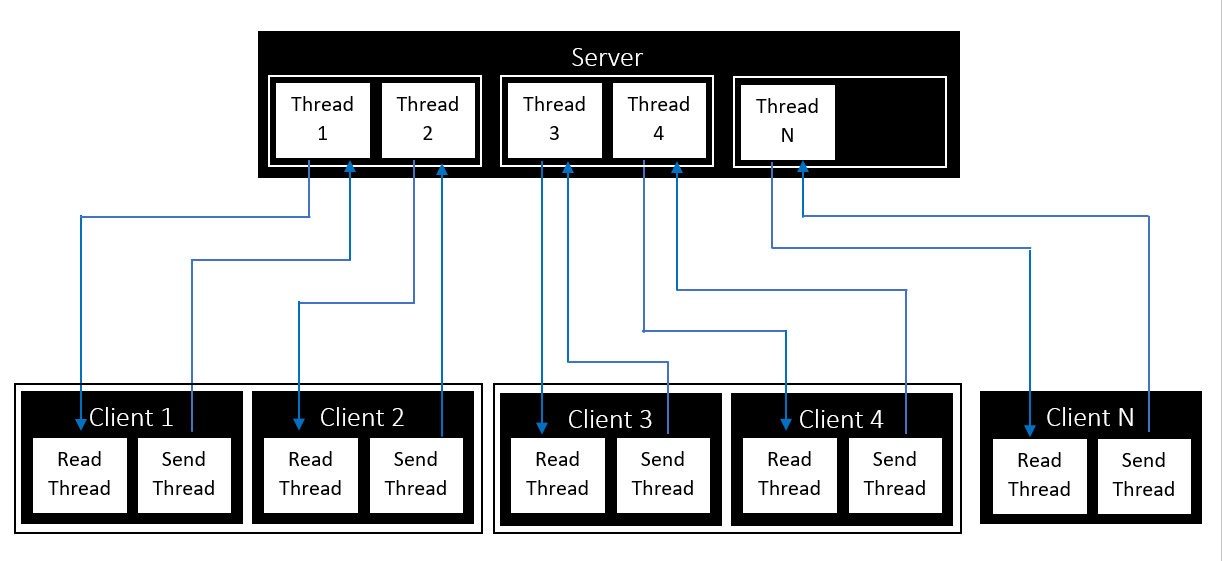
**3. Application Architecture**

This chess game application is server-client architecture applied to a regular chess game (multiplayer purposes).

Chess Game: Board, players, board color, all the pieces for the game

Server: Multithreaded, can handle multiple clients at the same time, there can only be 2 clients per game

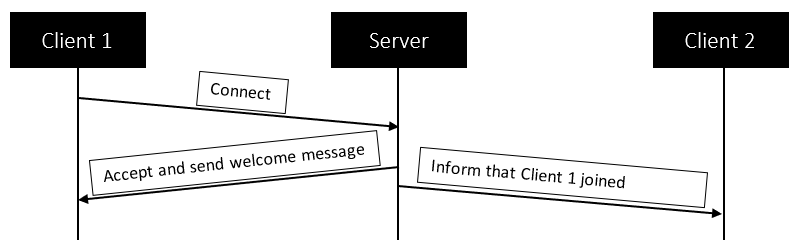
Client: Multithreaded, can receive messages from the server, can send messages to the server



**4. System Design**

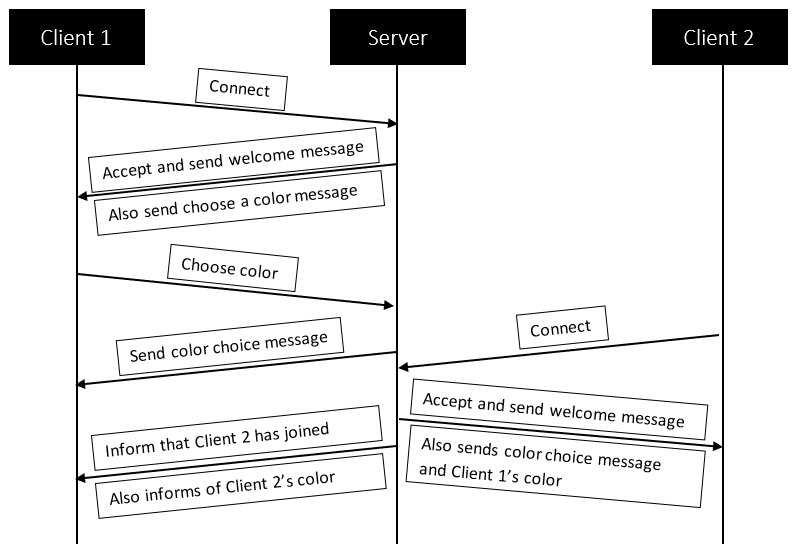
Allow client to join a game and choose a color/be given a color

* Client 1 connects to Server
* Server accepts connection and sends welcome message
* If there is a Client 2, Server notifies them of Client 1 joining the game



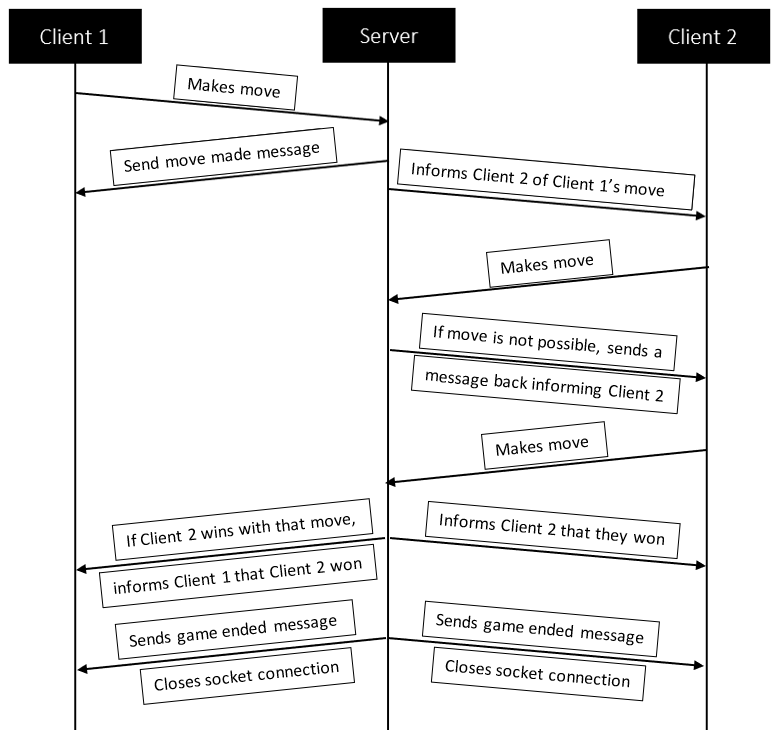
Allow client to choose a color (white or black) if they are the first one, otherwise be given the remaining color

* If Client 1 is the first player, Server asks them to choose a color (white or black) after the welcome message
* Client 1’s response is sent to the Server
* Server sends response message with the color they chose
* When Client 2 joins, Server sends a message after the welcome message with the color that is left, which is their color



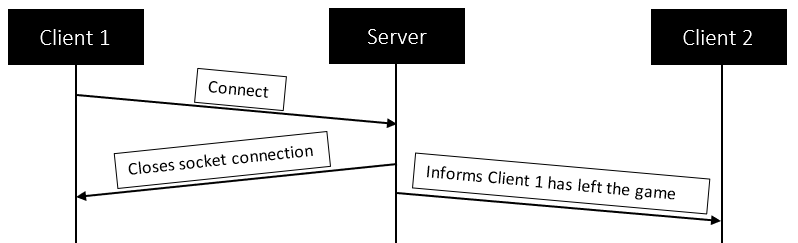
Allow client to make a move

* Client 1 makes a move
* Server repeats move message to Client 1
* Server notifies Client 2 of Client 1’s move
* Client 2 makes a move
* If the move is not possible, Server sends a message back to Client 2 saying that the move is not possible
* Client 2 makes another move
* If Client 2 wins, Server notifies everyone that Client 2 has won
* Server then sends a message saying game has ended, and closes socket connections for Clients 1 and 2



Allow client to leave the game

* Client 1 sends {quit} message
* Server closes the connection with Client 1
* Server notifies Client 2 that Client 1 left



**5. Plan**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Week | Feature | Task | Assign To | Done By |
| Week 9 | Chess Game | * Implement the chess game | Yuliya | Week 12 |
| Week 11 | Server-Client Interaction | * Implement the multithreaded server * Allow client to join a game * Allow client to leave the game | Tiffany | Week 13 |
| Week 12 | Server-Client Interaction | * Allow client to choose a color (white or black) if they are the first one, otherwise be given the remaining color * Allow client to make a move | Yuliya | Week 13 |
| Week 13 | Final Presentation | * Powerpoint Presentation | Team | Week 14 |
| Week 13 | Final Report | * Demo video | Tiffany | Week 14 |

**6. References**

* <https://web.cs.wpi.edu/~imgd4000/d07/projects/proj3/proj3.pdf>
* <https://www.youtube.com/watch?v=yyhNBV_5ayM>
* <https://www.youtube.com/watch?v=Bj6N0pEVC-I>
* <https://massivetechinterview.blogspot.com/2015/07/design-chess-game-using-oo-principles.html>
* <https://medium.com/analytics-vidhya/school-java-project-chess-1-85f97a2d1877>
* <http://www.java2s.com/example/java/2d-graphics/draw-a-chess-board.html>
* <https://www.openprocessing.org/sketch/309165/>
* [https://www.geeksforgeeks.org/design-a-chess-game/](https://nam04.safelinks.protection.outlook.com/?url=https%3A%2F%2Fwww.geeksforgeeks.org%2Fdesign-a-chess-game%2F&data=04%7C01%7Cphant9%40wit.edu%7C7876c08273e145d5fcfa08d88a5243b9%7C2af16cc576494528bc4d3d9b6f64c066%7C0%7C0%7C637411433235762309%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=syQlznFmJ3%2F6QyuWIPmpPHAlRyJsV1wIX%2F%2F832T6tDE%3D&reserved=0)
* [https://codemiles.com/finished-projects/java-chess-t618.html?mobile=on](https://nam04.safelinks.protection.outlook.com/?url=https%3A%2F%2Fcodemiles.com%2Ffinished-projects%2Fjava-chess-t618.html%3Fmobile%3Don&data=04%7C01%7Cphant9%40wit.edu%7C7876c08273e145d5fcfa08d88a5243b9%7C2af16cc576494528bc4d3d9b6f64c066%7C0%7C0%7C637411433235772264%7CUnknown%7CTWFpbGZsb3d8eyJWIjoiMC4wLjAwMDAiLCJQIjoiV2luMzIiLCJBTiI6Ik1haWwiLCJXVCI6Mn0%3D%7C1000&sdata=yx%2FVyQnVpsM7ZyCR6Z3quJ1TavGQFOMXIhqMaCk5sTc%3D&reserved=0)