PSG COLLEGE OF TECHNOLOGY

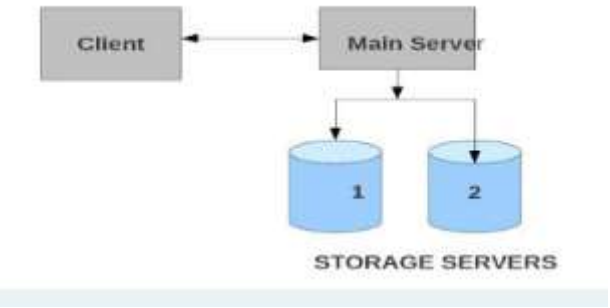
DEPARTMENT OF APPLIED MATHEMATICS AND COMPUTATIONAL SCIENCES

COMPUTER NETWORKS LAB
20XW68 DISTRIBUTED ENTERPRISE COMPUTING LAB

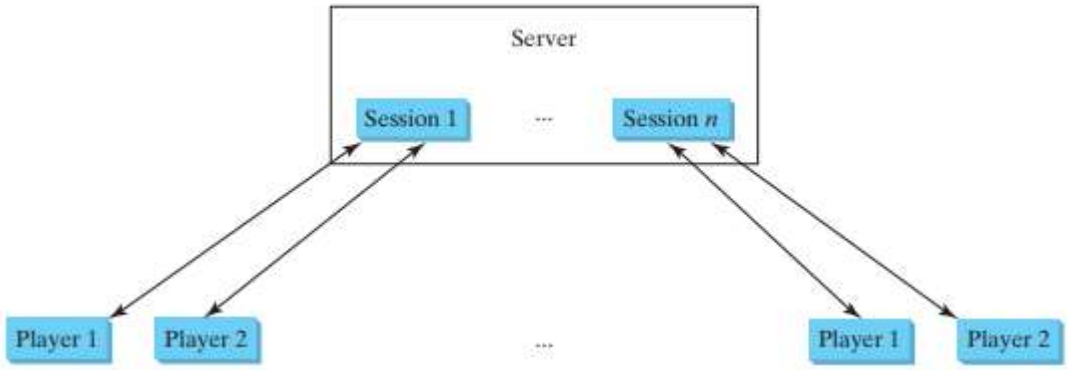
Problem Sheet 1

1) A chat room is an interface that allows two or more people to chat and send messages to everyone in the room. Set up a simple Chat Room server and allow multiple clients to connect to it using Socket Programming .The server and clients processes should run on different machines.

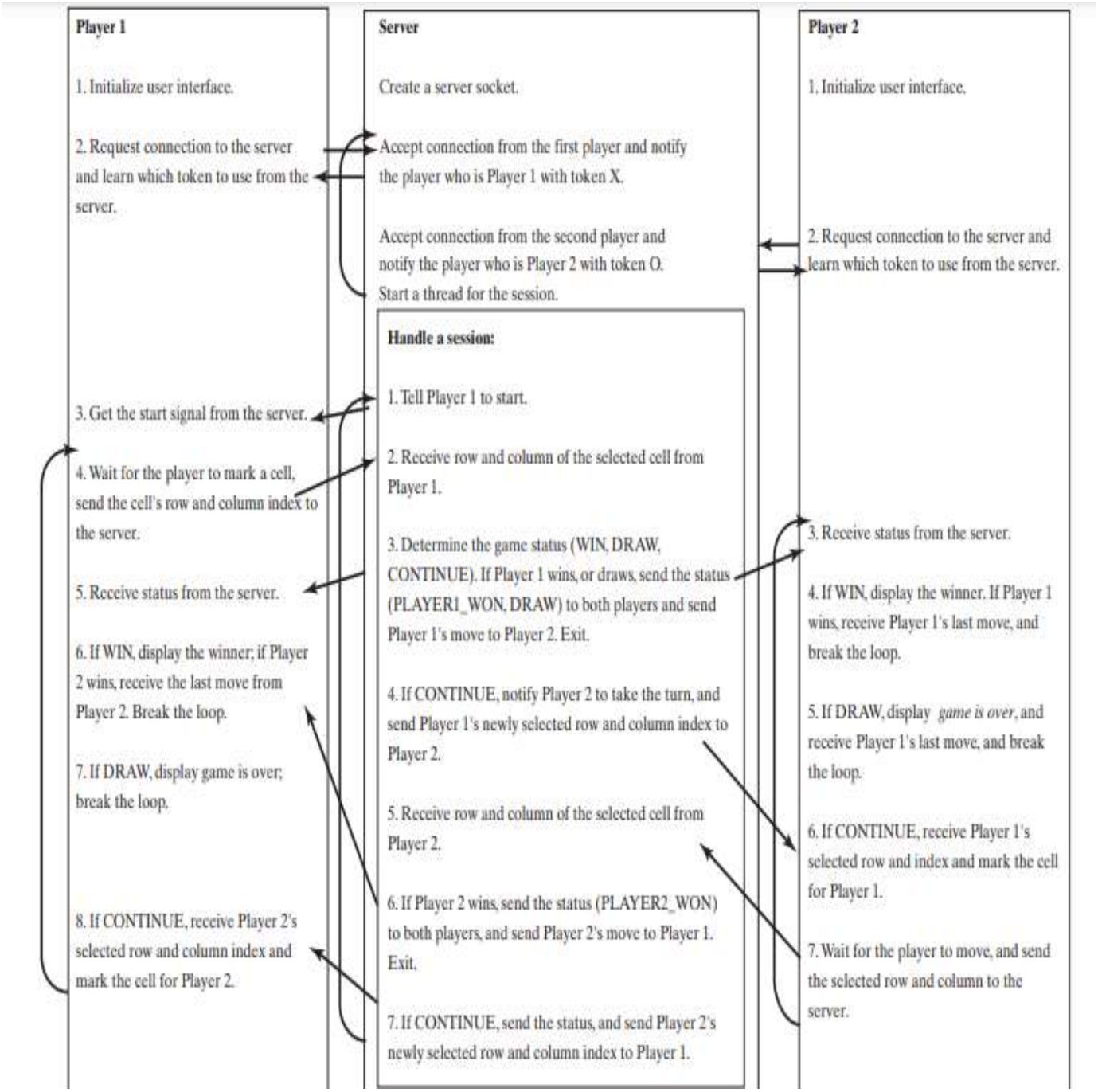
2)Imagine a Client-Server architecture (As shown in Figure), where user stores the file on a server. The main server splits that file into two or more fragments and store each fragment on separate storage server. When client retrieve the file from the main server, the main server again retrieves the file in fragments from storage servers and present it as a one file to user.



2) Develop a distributed tic-tac-toe game using multithreads and networking with socket streams. A distributed tic-tac-toe game enables users to play on different machines from anywhere .You need to develop a server for multiple clients. The server creates a server socket and accepts connections from every two players to form a session. Each session is a thread that communicates with the two players and determines the status of the game. The server can establish any number of sessions. For each session, the first client connecting to the server is identified as player 1 with token X, and the second client connecting is identified as player 2 with token O. The server notifies the players of their respective tokens. Once two clients are connected to it, the server starts a thread to facilitate the game between the two players by performing the steps repeatedly, as shown in Figure



The server starts a thread to facilitate communications between the two players is given in Figure.


Player 1
1. Initialize user interface.
2. Request connection to the server and learn which token to use from the server.
3. Get the start signal from the server.
4. Wait for the player to mark a cell, send the cell's row and column index to the server.
5. Receive status from the server.
6. If WIN, display the winner; if Player 2 wins, receive the last move from Player 2. Break the loop.
7. If DRAW, display game is over; break the loop.
8. If CONTINUE, receive Player 2's selected row and column index and mark the cell for Player 2.
Server
Create a server socket.
Accept connection from the first player and notify the player who is Player 1 with token X.
Accept connection from the second player and notify the player who is Player 2 with token O.
Start a thread for the session.
Handle a session:
1. Tell Player 1 to start.
2. Receive row and column of the selected cell from Player 1.
3. Determine the game status (WIN, DRAW, CONTINUE). If Player 1 wins, or draws, send the status (PLAYER1_WON, DRAW) to both players and send Player 1's move to Player 2. Exit.
4. If CONTINUE, notify Player 2 to take the turn, and send Player 1's newly selected row and column index to Player 2.
5. Receive row and column of the selected cell from Player 2.
6. If Player 2 wins, send the status (PLAYER2_WON) to both players, and send Player 2's move to Player 1. Exit.
7. If CONTINUE, send the status, and send Player 2's newly selected row and column index to Player 1.
Player 2
1. Initialize user interface.
2. Request connection to the server and learn which token to use from the server.
3. Receive status from the server.
4. If WIN, display the winner. If Player 1 wins, receive Player 1's last move, and break the loop.
5. If DRAW, display game is over, and receive Player 1's last move, and break the loop.
6. If CONTINUE, receive Player 1's selected row and index and mark the cell for Player 1.
7. Wait for the player to move, and send the selected row and column to the server.