CS F303: COMPUTER NETWORKS

GROUP NO: G25

PROJECT TOPIC: GAME DESIGN: - TAMBOLA

SUBHASH SAMOTA	2014A7PS060P
MIHIR SAXENA	2014A7PS098P
ABHISHEK TEOTIA	2014A7PS145P
SHIVESH GANJU	2014A7PS146P

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

25th April 2017

https://github.com/Shiveshganju/NetworkMiniProjectG25.git

1. Problem Statement

We have to conceptualize, specify, design and implement a multi-player, three-level, interactive network game involving whiteboard and live score display features. We propose to design a multiplayer game called ‘Tambola’. The game would be having three levels of difficulty. The stage 1 will be based on a simple number guessing game format in which the user can only win if all the numbers are crossed in the number matrix he has.

The stage 2 will be based on a similar format with the added features that the users will be given choices which they can implement to make the game more interesting.

The stage 3 will be based on the popular existing game known as Bingo. The user having the maximum number of points will be declared the winner. The game will be designed so as to handle unforeseen situations such as server and client crashes. There will be added feature in which the client can pause the game and can continue from the position in which he/she had left. Through this designing problem we will understand server client architecture and handling multiple clients.

2. Scope of work done

The given project covers the design and implementation of the ‘Tambola’ game. More focus has been given to the networking and synchronization aspect of the game along with taking care of the scoreboard, GUI and also ensuring the game is entertaining.

Many functionalities like pausing the game, reconnecting if disconnected etc. have been ensured in the game as well. All the four members have been involved in the design and testing of the game equally. Moreover all the members were responsible for putting the project into effect with the work distributed equally and each taking turns in module implementation.

3. Challenges and Limitations

- Server is the central point of failure. If the server fails, all the clients will get disconnected and the data will get lost. Backup for the data has not been created. The game has to be started afresh.
- Adding the pause functionality (the client can choose to pause the game and the game will start from that point itself) posed a challenge.
- Reconnecting the client to the server in case of a disruption in the network or in the case of the client getting disconnected was abstruse.
- Maintaining the matrix of the client before it gets disconnected as well as creating a backup for the scores with the server was hard.
- The system will work only for a single server.

- In case the user disconnects, the game waits for the user for a certain time limit after which the user won't be able to join the game and the game will continue from the same point.
- The scoreboard shows the score of all the possible players. It shows the scores as 0 for the clients who are not connected.

4. Design and Implementation details

4.1. An overview

It is an internet multiplayer game based on client-server architecture. The game has some added functionalities such as a whiteboard for every client on which all scores would be visible as well as a pause functionality and also a mechanism for handling server and client crashes.
The game proceeds in the following manner:

4.1.1. The server listens on a known socket and waits for clients to join to its network. When an incoming connection from a client arrives, the listening socket creates a new socket (the 'child' socket), and establishes the connection on the child socket.

4.1.2. After a given threshold time, once a connection gets established between the server and its clients, the game begins.

4.1.3. The server sends a list of rules of the game to each of its players.

4.1.4. This is a three level game and each level has a different gameplay and different design.

4.1.5. After the game finishes, the server announces the winner of the game and the whiteboard displays the cumulative score of all the players in the game.

4.2 Architectural design

I/O Multiplexing – I/O multiplexing has been used to improve the performance of the system. We call *select* or *poll* and *block* in one of these two system calls, instead of blocking in the actual I/O system call. The model can be explained in the figure given below.

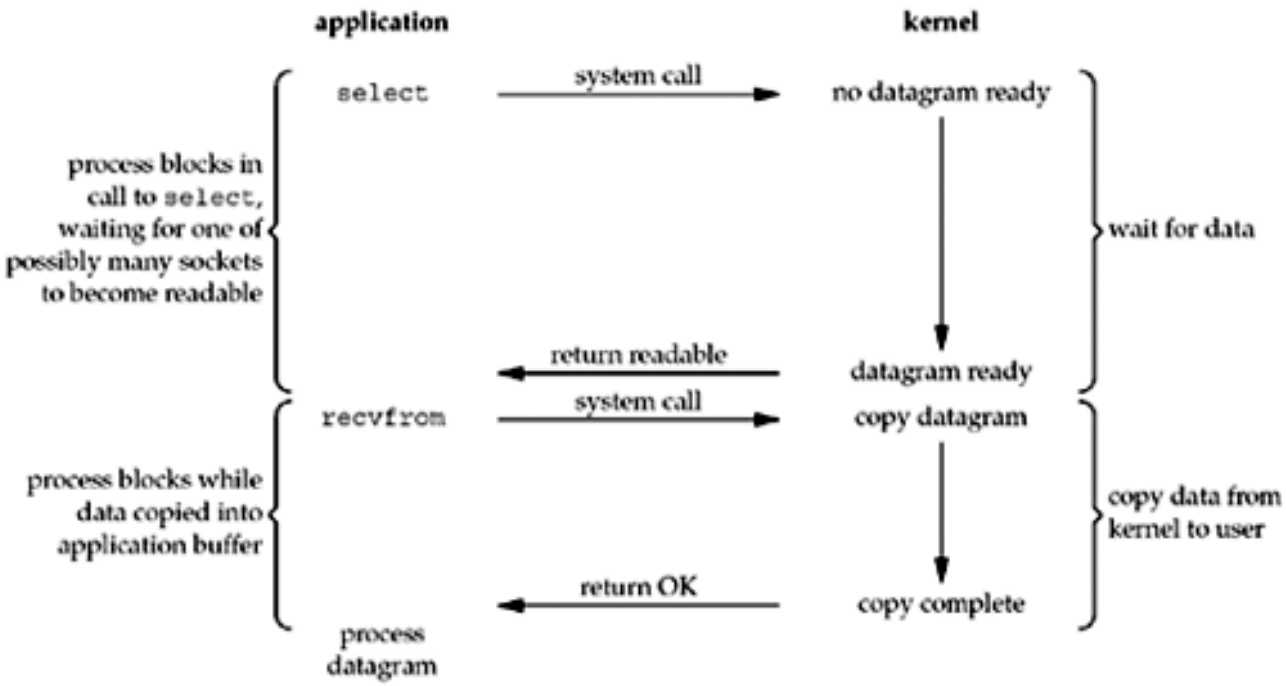

Fig 1: I/O Multiplexing model

Iterative solution was not used as it was slow and was having a lag time. The server is designed using I/O multiplexing model using *select* call. All the waiting timers for clients have been implemented properly. The Listening socket is in non-blocking mode and all the other sockets are in blocking mode.

4.3 Structural Design

4.3.1 Important Data Structures used

- **Three Dimensional matrix –** A three dimensional matrix has been used in the server program. The three dimensional matrix stores the player numbers along with their respective matrices. The first index is used to depict the player number and the other indices corresponding to the first index represent the matrix that has been given to the player.

- **Two dimensional matrix –** The client side maintains a two dimensional matrix which has been used in the game tambola and Bingo respectively. The server announces a number and the clients (players) check for the number in this matrix and cut that number if it is present.

4.3.2 Important Functions used

Server

- **encriptMatrixToMessage –** This function converts the 2 D matrix into a 1 D array by using row wise traversal.

- **generateRandomNumber –** This functions generates random numbers which get filled in random positions in the matrix which is sent to the client. Moreover this function is called when the server announces random numbers.

- **sendRules –** This function is used to send the rules of the game to the clients by transferring the message from the server socket to the client socket using TCP as the transport protocol.

- **check_winner –** This functions returns the winner by computing the maximum score in the scores array which it takes as the input parameter.

- **main –** In the main function the server generates matrices for the clients and stores it in a 3 dimensional matrix as a backup. During the game, random number gets generated which get sent to the client using the TCP socket. The winner is also calculated after every level in this function

Client

- **search –** The functions takes input as the random number generated by the server which s received by the client through the socket. It then searches for the number in the matrix and then replaces it with-1. This function gets implemented only in level 1 and level 2.
- **string_to_matrix** - This functions converts the string of numbers which is sent by the server to the client through the socket into a two dimensional matrix.

- **send_fnf** - The function returns 1 or 0 to the server if the number that has been announced by the server is found in the matrix or not respectively. The server then accordingly updates the score of the client.

- **bingo –** This function takes input an array of flags for rows, columns and diagonals along with a 2D matrix. This function then checks if the numbers present in a row or a column or a diagonal are -1. If they all are -1, it sets the flag of that row or column to 1 in the array and increments the bingo_count. This function returns the bingo_count of the player.

- **main –** The client connects to the server through a socket in the main function. After this the client proceeds to play the game according to the rules that were mentioned in the previous document. The game continues until a winner is declared.

4.4 Pause Functionality

The client has an option to use the pause functionality. The server will wait till the client sends a resume message, after which the game will resume from the point where the game was paused. The matrices of all the clients have been stored within the server in the 3 dimensional array and gets updated periodically. If the client gets disconnected after using the pause functionality, then the server waits for 5 seconds for the client to reconnect else the client will be removed from the game and the game will resume.

4.5. Whiteboard

The whiteboard is basically the command line window. The purpose of the whiteboard is to keep all the users updated about the scores of all users after the server announces a number. The whiteboard also shows the updated matrix of the user after the server announces a particular number. In round 2 it is also used to show the user the surprise card options that he/she has.

4.6. Reconnecting the disconnected clients

The server maintains an array in which the IP addresses of all the clients get stored in. In case one any of the client gets disconnected, the server listens for a time period of 5 seconds until the disconnected client reconnects using the *select* function. In case any other client beside the disconnected one attempts to reconnect, the server closes the socket.

5. Learnings and Conclusion

The project helped us gain a deep insight into socket programming. The project was implemented successfully across multiple terminals and thus fulfilled its requirement of functioning as a multiplayer game over a connected network.

We learnt the importance and functioning of sockets in real time applications as well as the working of I/O multiplexing. We also learnt the use of system calls along with network programming which required the knowledge of operating systems as well. We also learnt how I/O multiplexing is better than the iterative solution.

6. Future Extension

The game could be improved by creating a backup in case the server fails. The future implementation of the game can include creating a backup for the data. A better interface can be implemented which can make the GUI more attractive. The level 2 of the game can contain more jackpot options which can make the game more entertaining.

7. APPENDIX

7.1. Whiteboard: A whiteboard displays the current information about the progress of the game and is visible to all the players. As soon as a game round takes place, the necessary information gets updated by the server.

7.2. Tambola: Tambola is a game of probability in which players mark off numbers on cards as the numbers are drawn randomly by a caller, the winner being the first person to mark off all their numbers.

7.3. Bingo: Bingo is a modified and a difficult version of Tambola in which players aim to mark off numbers on cards in a particular sequence, say, horizontal, vertical or diagonal and as soon as a pattern is matched, one letter from 'BINGO' is marked off, the winner being the first person to mark off each of {B,I,N,G,O}.

7.4. Surprise card: It is a hidden option offered to players to ease/demote their progress in the game.

8. REFERENCES

8.1. "The socket programming and Software design for communication based on Client/server" by Ming Xue, Changchung Institute of Technology, Changjun Zhu HeBei University of Engineering.

8.2. "RFC 147 - Definition of a socket - IETF Tools."

8.3 "Computer Networking A top-down Approach" by James F. Kurose, Keith W. Ross.

8.4 "TCP/IP Socket I/O Multiplexing Using a Hybrid Polling Event System" by Jianli Sun.