Computer Science and Technology
Group 11
LIANG Zelong (1030300024)
ZHANG Yixin (1030300059)
FENG Xudong (1030300008)
Due Date: 2011-12-26

Project Report of Two Player Network Chinese Chess Gaming Program

Abstract: In project part II, we have designed a suit of two player Chinese chess program, includes a server, a player client, and other supportive class file. We have used GUI programming method, socket programming method and file i/o to implement the part. Now the program supports basic Chinese chess function, such as move/destroy chess and play back. The program also supports some additional functions, such as chat through two clients, show the move step in text and save the chat information. We have got experience of programming a GUI program, sending information through network and analysis real problems in object-oriented programming way.

Introduction:

In the second part of the final project, we are trying to develop a two-player Chinese chess program. This program need GUI design and network information exchange to implement its function. Thus, basically, we are solving three problems: how to organize the data, the field inside each side of chess, how to send the movement to the opponent without wrong meaning and how to get to information from the opponent correct.

The main difficulties of this part are: the program's stability and the game entertainment. There are decades of variable and function inside the program. Organizing the variable is not easy because one function modified it may cause a side effect on other variable or information. Most of the time, we need to write some functions, or even rewrite long lines of code for the same function. Another difficulties is how to make the game more interesting. It needs us to be strict of our chess rules, give human-friend solution to some basic game control like new game and play back. We have provided more functions, as showing the step and chat. More functions can make the program more interesting.

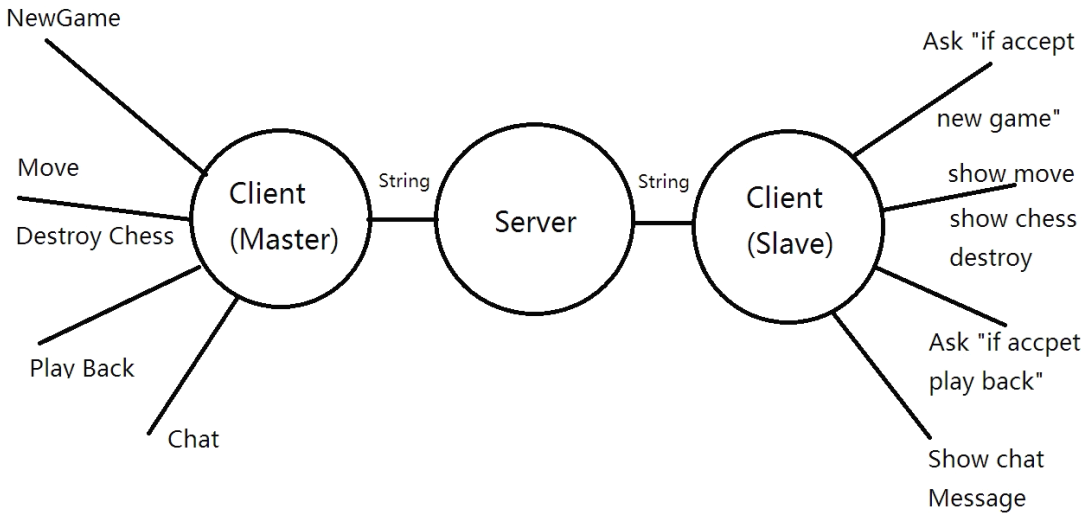
The main resource of knowledge is from the subject lectures, what's more, we have learned from some class demos: the network Guan-Poker program, the small-scale chat program and the single player Chinese Chess program. The basic chess rule and the data structure of the chess are inherited from the single player Chinese Chess program. The network structure is based on the the network Guan-Poker program and the small-scale chat program.

In this part, we have successfully built up the network Chinese chess program. By working on the project ,we have got on-hand experience of simple GUI programming, networking using, thread using. The knowledge we learned from this part is a clear model of transportation and controlling network information.

Design and Methodology

Architecture: The architecture of network of the program is "client-server-client".(show in the graph). There are two task for the server. It transmits the information from one client to another client, and makes sure no more than two client have connected to the server. The client receive input from graphic user interface, organize the information, and sent the string type information to the server. On

the other hand, receive string type information from the server, organize the information, and show them to the user in graphic way.



Graph-The Architecture of the information exchange.

Methodology:

Threads are used to complete different tasks at the same time. There are large and pivot threads: the thread to detect and analysis whether the chess is eaten or the game is over, the thread to receive information from the server and invoke local method to complete the move. There are also small-scale threads, such as a thread to make the chosen chess "blinking".

Socket are used to exchange information through network. The stream communication is used because it is a reliable date exchange method. Once the connection is set up, the information will be sent and receive in correct sequence. This correctness is very important for the game play: the user do not need to be afraid of losing control.

File output is used to save step and chat information. The chat information is already in string type. Thus, it is very easy to get the text from the chat board and save it. Text file is used because it is portable for most platforms. To export the information to a file, the File output method is used.

Formatted information is used to make sure the information can be read and transform to commands correctly. The formatted information concludes two part: the header and the information body. The header is a string of command name, such as "NewGame", "PlayBack". The information body contents basic date of the command, included the index of the moved chess, the destination coordinate of the chess, etc.

Implement and Experiments:

Unfortunately, we still can not save all the chess data on the game server. Thus, we need to synchronize two chess board and all chess on the boards. When one side move a chess or destroyed another chess, the movement will be sent to the other side, and make the same movement. If one side want to start a new game, or play back, he or she need to get permit from his or her opponent, the

message is sent between two client. If the opponent permits, both sides will do starting a new game or play back at the same time. The organization of the information exchange is as below. Before the data is sent, it will be added the header. On the other side client, a method split the information string into the header and the original date. The method calls the local functions according to the header and input the data as the parameter. This procedure can ensure the chess movement will be the same on both sides at the same time.

To make the gaming program more interesting, we add some methods to differentiate two clients. A chat board and a information input text field is add to the GUI of the program. If you like, the chess program can even be used as a simple chat program. When you input a short message and click the "Send" button, the information will be append to your chat board and be sent to the server. The sever will transmit it to the other client. On the other client, the information will be refined and put on the chat board. Both side of the player can save the chart record at anytime they like. The chart record is saved into a text file and it is named after the current system time.

What's more, we have redesigned the GUI so each player will see its own side "closer" to himself or herself. To implement this function, we assign "master" and "slave" to two client. The master client will draw the chess board and all the chess normally, the black side is on the lower side of the chess board(which is the same as the single player chess game). The slave client will draw the chess board and all the chess reversely, up side down. The red side is on the lower side of the chess board. All moving and drawing methods are modified for the GUI. But the actual ChessBoard Class and the ChessGroup Class are the same, so all the differences are for eyes only.

The experiment:

Several simple experiments are set up to demonstrate that the whole chess program suit can run normally. Beside, the chess program are robuster than this scale of test. Thus, it can be played without worrying whether it can function normally.

Start connection: two clients can successfully connected to the server, after this step, two clients can use the chat function. Click send, the information on the input text file will be showed on the other side.

New Game: when one side clicks "New Game" Button or the function on the menu bar, the other side will receive a message asking for starting a new game. When he or she clicks it, both side will pump out a panel saying "New Game Start", and the chess will be put into the chess board. For the black side, the chess on the lower side is black; for the red side, the chess on the lower side is red.

The information exchange and user interface demonstration are success.

Chess move: When the black side click his or her "cannon", and click an empty chess point on the chess board, the black "cannon" will move to the clicked point. On the other client, the black "cannon" moves to the same place, but on the reverse point on the chess board.

When it is the red side to move, and the black "cannon" is in the attacking range of the red "horse", click the red "horse" and the black "cannon", the black "cannon" is disappeared, and the chess point is taken by the red "horse". On the black side, it can be find that the black "cannon" is destroyed by the red "horse".

The moving and eating chess function is success.

Play Back: Now it is black side's turn. The black clicks the "Play Back" Button. A panel prints on the red side's interface and ask the red whether he or she would like to accept the play back movement. The red clicks the button "No". On the black side, a message box says the red has refused to playback, no change happened on both side. Now the black click the "play back" Button again. This time, the red

chooses "accept". Message are printed on both side "accept play back", and the red "horse" moves back to its original position, the black "cannon" appears again and moves back to its original position.

The Play Back function is success.

File save: click "save step" button, and go to the root direction of the chess game program. A file named by time and "save step" will be found. Open it by the Notepad program, information of each step taken is record in format "Chess group-Chess rank+original position+current position".

The File save function is success.

Conclusion:

In this project, we have get useful experience on GUI programming, network information exchanging and file input/output. What's more, we have learneds how to analysis a real problem or situation and find solution in object oriented programming way. Over one third of the project time was used on analysis on the reality and find the similarity in the program. The Chinese chess program can implement the function listed in the project proposal. Thus, we can demonstrate our system by our program.