Design of Networked Multiplayer Snake and Ladder Educational Game Based on Hash Map and Vector Data Structure

A'la Syauqi[1], Permata Rahmatul Hijjah[2]

[1,2] *Jurusan Teknik Informatika, Fakultas Sains dan Teknologi, Universitas Islam Negeri Maulana Malik Ibrahim Malang, Indonesia*

[1]syauqi@ti.uin-malang.ac.id
[2]tatha.permata@gmail.com

Abstract - Computer games have been used as educational media or popularly named as educational games. However, most educational computer games that have been created can be played by one player. This study aims to build a multiplayer ladder and snake educational game. The game is built on Java socket programming and local area network (LAN) as a data communication medium between players. Whereas to handle the username and socket address of the whole player, the hash map data structure is used. A vector data structure is also used to manage data package sending for each player group. The experiment shows that the system works properly where the computer server's performance is influenced by the specifications, especially the processor and random-access memory (RAM).

Keywords: Game, Multiplayer, Education, Network, Server

I. INTRODUCTION

Nowadays, computer games have been used in education, from elementary school to college. Several studies on the development of computer games for education, for instance, games for natural science courses [1], health education [2], mathematics education [3], [4], physics learning [5], learning English [6], teaching computer programming [7], [8], math logic [9], introduction of rare animals[10], educational game on waste handling[11], and five senses learning[12] has been conducted. It was concluded that the use of computer games in education showed positive results, namely increasing student interest in learning and making it easier for students to understand the learning material. However, the game from this study can be played by one player (singleplayer). The research showed that games played more than one player (multiplayer) were more challenging than singleplayer [13]. Besides, school-age children tend to like competition games and are played in groups [14].

Many multiplayer games have been conducted, battleships [15], for example. In this game, two players are connected in a peer to peer network with a cross cable, where one computer player as a server and the other as a client. However, the number of players is limited in this game, which is two players. Another multiplayer game is the snake and ladder game [16]. More than two players could play this multiplayer game; although, there is no player grouping feature (group); thus, the number of players in the game is potentially too much. The next multiplayer game is truf [17], which has a player group creation feature, although the group must wait until the fulfillment of four players. Another multiplayer game is gobak sodor [18], which is built based on flash, whereas there are still bugs, one of them when used to play more than two players, there will be a lag or stop. There are also spirit card multiplayer games [19], which can be played by six players; nevertheless, there is no group feature yet. Thus, the game system is limited to handle six players.

This paper aims to design and implement in build multiplayer snake and ladder educational games with arithmetic learning content. The game was built on Java Socket Programming and used a Local Area Network (LAN) to communicate between players. The game was designed for players to create groups or join existing groups, where each group consists of four players maximum.

II. METHOD

The game's main interface for the player is a snake and ladder graphics board composed of 100 small squares with ten vertical squares by ten horizontal squares. This board is equipped with images of snakes and ladders at particular square locations. Players use the board to play the game. On the right side of the board, a username is included as the players' identity in a group.

Underneath the player's identity is a dice visualization and a shuffle button to determine the number of steps the player has when taking a turn. For educational content, arithmetic questions are included. Questions would appear when a player stops at the snake's tail or ladder's base square. If the player succeeds in answering questions, the player does not go down to the square following the snake's body or go up to the square following the stairs, and vice versa. The main interface mockup of the game, as shown in Fig. 1.

Based on research about programmable network architecture has been conducted [20]; therefore, a distributed system is applied to the game with a client-server architecture model [21]. The server is a computer that serves the client, while the client is a player's computer that obtains service access to the server. In this system, the server works actively to process data packages from player groups or requests from clients. In general, the data package flow processes on the client-server consist of client-server connection, input data on the client, client-side data processing, server-side data processing, and broadcasting, as shown in Fig. 2. Each step will be discussed further in the next subsection.

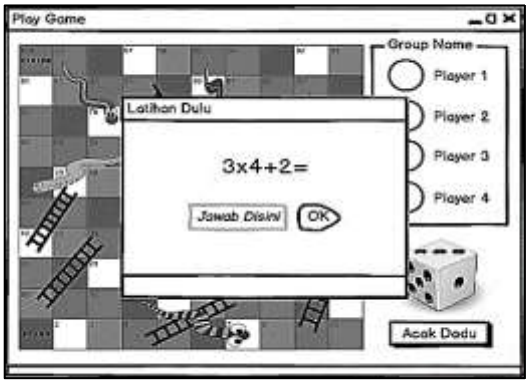


Fig. 1 Player game interface mock up

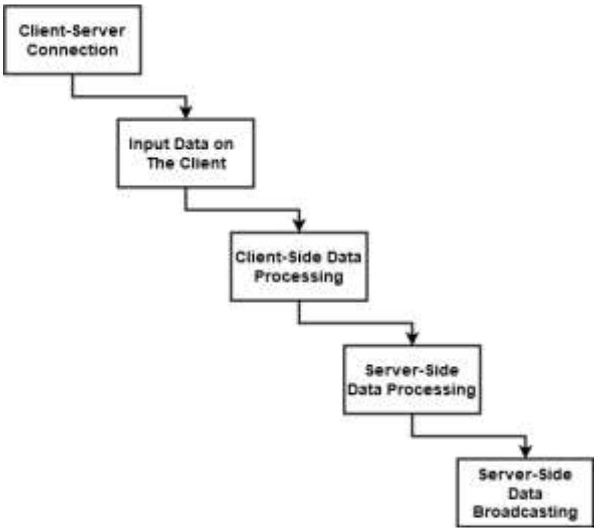

Client-Server Connection
Input Data on The Client
Client-Side Data Processing
Server-Side Data Processing
Server-Side Data Broadcasting

Fig. 2 Data package flow on the client-server

A. Client-server Connection

The game can be played by two to four players in each group as clients (Fig. 3). In Fig. 3 to simplify, two-player groups are used; however, the system could accept more than two groups. The first process in constructs a network is starting with the server opens a connection and waiting for the client to join. The server opens a specified port. A server socket is formed on the server application side and listen/waiting operations for connection requests from the client-side.

On the other hand, the client requests the IP and port that has been opened by the server. When the client connection request arrives at the server, the server will create a socket. This socket will be used to communicate with the socket on the client-side. Then the socket server does listen more to wait for connection requests from other clients. Games are designed to handle multiple players at the same time by implementing multithreading. Thus, the server can hold the connection process of several players at once.

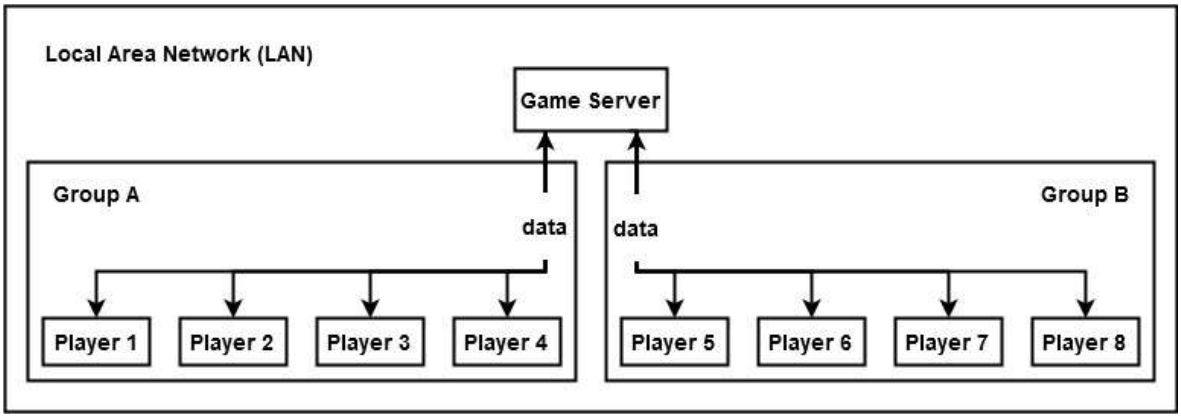


Fig. 3 Client-server game diagram

B. Input Data on The Client

A player that has been connected server, then enter a username that is used during play the game. Username is the identity for each player. After a username is entered, the player can create a group or join groups that have been created by other players. Then sends the data containing the username, socket address, and group name to the server. The server receives and stores it in the HashMap with the structure: HashMap(string, socket). The string is variable containing the username, and the socket is a variable for storing the player's socket address. Then the server saves the username and group name into the database. Username and group name is used to identify players who are members of the group in sending data to other players through the server. Table I shows the database design used in this game.

The database game consists of one table that contains three fields, namely username, group, and ket. Each field has the following function: username for storing the player username, group for the name of the group that the player follows, and ket for player or group information. The username is the primary key of the table to make it easier to find and manage the player's socket on the HashMap.

C. Client-side Data Processing

Before the game starts, the player can join an existing group or wait for other players to join the created group. After all players in a group are ready, the game can be started. While the game is in progress, each player sends a data package containing its position on the game board to the server to be broadcasted to other players in the same group. When the player gets turn to play, then shuffles the dice. The dice's number as a reference on how many squares must player walk through the game board. In this process then the player's computer (client) sends its position data to the server in a string data packet format as follow: <nama_grup>:<username>:<posisi_saat_ini>:<posisi_sebelumnya>

Where <nama_grup> is the name of the group the player is following, <username> is the name of the player who is playing, <posisi_saat_ini> is the last player's position on the square board, and the <posisi_sebelumnya> is the position before shuffling the dice. The ":" character is a delimiter used to split string to be processed separately. For example, GrupX:Budi:17:15, can be explained as follows: GrupX is the group's name, Budi is the username of the player, 17 is the last position of the Budi, and 15 is the previous position. Flowchart sending data from client to server is shown in Fig. 4.

Besides sending data to the server, the player also receives data from the server when other players in the same group get their turn to play. Data packages received by a player are split to get other player position data. Fig. 5 shows the player receiving data from other players through the server until the visualization of its position on the snake and ladder game board. Data packet splitting is done with a split function in Java based on delimiter characters (:). The data package is split into four parts: <nama_group>, <username>, <posisi_saat_ini>, <posisi_sebelumnya>. Then this data is stored in an array as follows: arraydata[]={[nama_grup], [username], [posisi_saat_ini], [posisi_sebelumnya]}. To get the player's position from the array can be accessed using the index, for example: userNm = arraydata[1], posSkrg = arraydata[2], and posSblm = arraydata[3]. After getting another player's position, the player game program visualizes it on the snake and ladder game board.

TABLE I
DATABASE DESIGN

Field Name	Type	Annotation
Username	varchar(10)	Primary Key
Grup	varchar(10)	
Ket	varchar(10)	

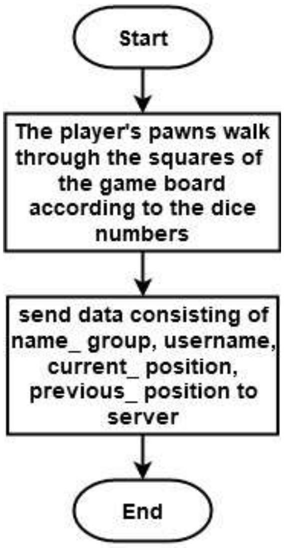

Start
The player's pawns walk through the squares of the game board according to the dice numbers
send data consisting of name_ group, username, current_ position, previous_ position to server
End

Fig. 4 Sending data from client to server

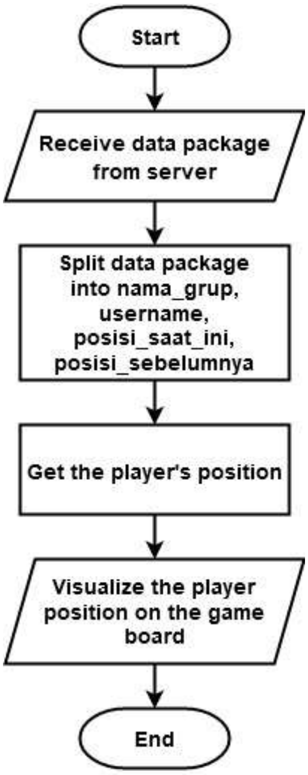

Start
Receive data package from server
Split data package into nama_grup, username, posisi_saat_ini, posisi_sebelumnya
Get the player's position
Visualize the player position on the game board
End

Fig. 5 Receiving data from server to client

D. Server-side Data Processing and Broadcasting

Besides, as a data broadcaster to the player, the server also processes data. When the server receives a data packet from the player, the server processes and splits it to find out the player's group name. The data processing by the server is shown in Fig. 6. Same as in the client, the server splits data package with split functions in Java with a delimiter (:). The format of a data package from the player is as follows: <nama_grup>:<username>:<posisi_saat_ini>:<posisi_sebelumnya> is split and saved to the array into: arraydata[]={[nama_grup], [username], [posisi_saat_ini], [posisi_sebelumnya]}.

Thus, the name of the player can be obtained from index 1 of the array data. Next, the server takes the name of the player group from the database. Then look for all player names with the same group name in the database. After the player's name is known, the server checks on HashMap whether the player with the name is valid. After that, based on the player's name, the player socket address is retrieved from HashMap and collected in a vector. Last, the data packet is sent to clients (players) through the socket address that has been stored in the vector. Thus, the server broadcasts the data packet to all members of the group.

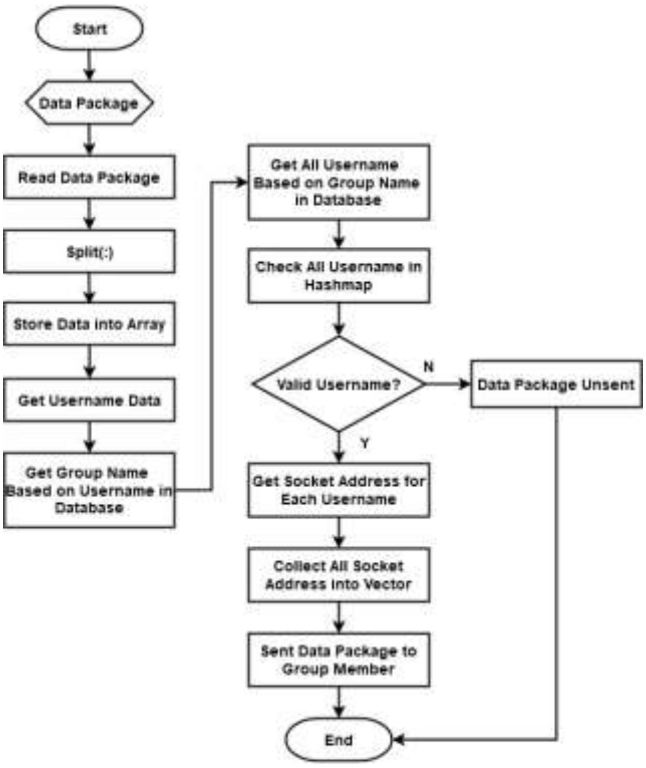

Start
Data Package
Read Data Package
Split(:)
Store Data into Array
Get Username Data
Get Group Name Based on Username in Database
Get All Username Based on Group Name in Database
Check All Username in Hashmap
Valid Username?
N
Data Package Unsent
Y
Get Socket Address for Each Username
Collect All Socket Address into Vector
Sent Data Package to Group Member
End

Fig. 6 Receiving data from server to client

III. RESULTS AND DISCUSSIONS

In this section, the results of research and examination of game software will be discussed, including experiment client software that displays an animated snake and ladder board with interactive graphics during the game and server software as moderator the player group during the game running.

A. Player's Game User Interface

This study's output is an educational ladder and snake game which is played together in a network, where players are collected in groups. The game interface for a player (client-side) is shown in Fig. 7. The game interface is an interactive graphical display that is used to play the game. It is a snake and ladder game's board consisting of the player's pawns, player's identity, live chat feature, and dice with the shuffle button. The arithmetic quiz questions would appear when the player stops at a square with a picture of a ladder base or snake tail.

Rules of the game as in a game of snakes and ladders in general, which players compete to reach the top square with number 100, starts from the bottom square with number 1 according to the appearance of dice numbers shuffled when one player gets a turn to play. The player shuffles the dice with the shuffle button with the text "turn."

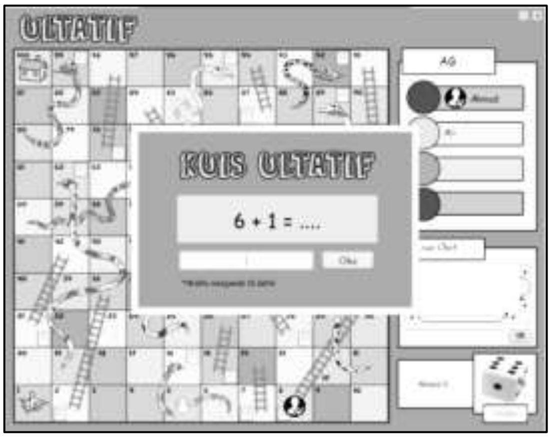


Fig. 7 Game user interface for player

When the player's pawn stops right at the square, which is the base of the ladder image, the player has to open the questions; hence interactive graphics of the quiz would appear. Players are allowed to answer questions for 10 seconds. If the answer is correct, the player's pawn will go up to the square where the ladder's end is. Otherwise, the player's pawn fails to go up. Likewise, if in 10 seconds the player does not answer, it is considered wrong, and the player's pawn fails to rise. Whereas when the player's pawn stops at the end of the snake's tail, the player has to open a quiz question yet with different rules. If the answer given by the player is correct, then the player's pawn does not go down to the square where the snake's head is. Otherwise, if the player's answer is wrong or the question is not answered within 10 seconds, the player's pawn will go down to the piece where the snake's head is.

The game also features a live chat feature for more excitement. Players who are members of the player group could see the text of conversations sent by other players. The player also could send a message by typing it in the text field then click the OK button; thus, the message appears in the text area and can be seen by all player group members.

B. Server User Interface

The server is a computer as a moderator to manage and control player activities. The appearance of the server program interface is, as shown in Fig. 8. There is an ON / OFF button in the server program interface to enable or disable the server, whereas the text area is used to monitor ongoing activities. Before the game can be played, this server must be active.

C. Login User Interface

The login interface is in the player (client-side) game program. Fig. 9 shows the login interface intended. In the login interface, three text fields must be filled by the player: host or server IP, the same port used by the server, and the username as the player's identity in the game group. Before playing, players must log in to connect to the server, and then they could choose or create a new player group.

D. Player Group List User Interface

The list of player groups that have been created is displayed in the player group list interface that appears after the player has logged in. The group list interface is shown in Fig. 10. If the player wants to create a new player group, it can be done by clicking the "Create Group" button.

There is a "refresh" button in the player group list interface to update the player group list created by other players. If the player wants to join an available group, it can be done by clicking on the group name that appears in the group list. When the player selects the intended group of players, a confirmation dialog will appear, as shown in Fig. 11. In this interface, there is also a "help" button to display playing instructions, an "about" button to display program information, and an "exit" button to stop and exit the program.

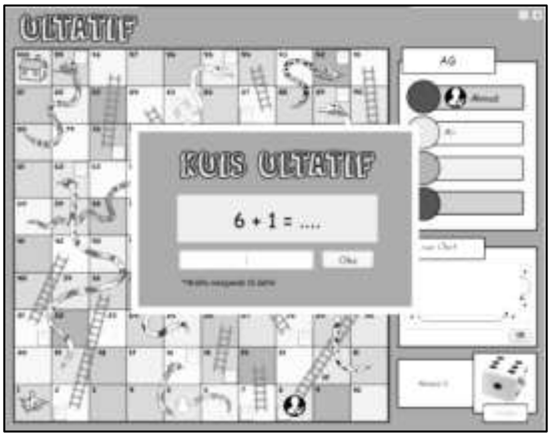
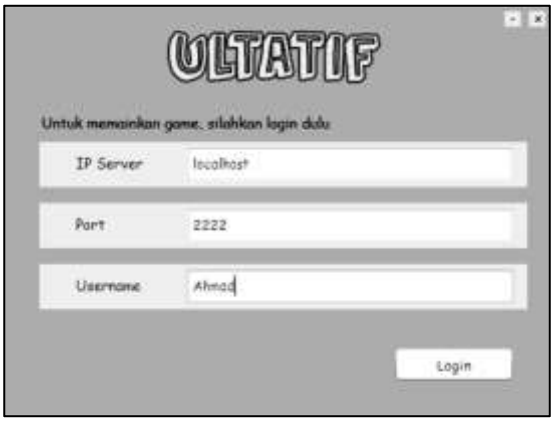


Fig. 8 Server user interface



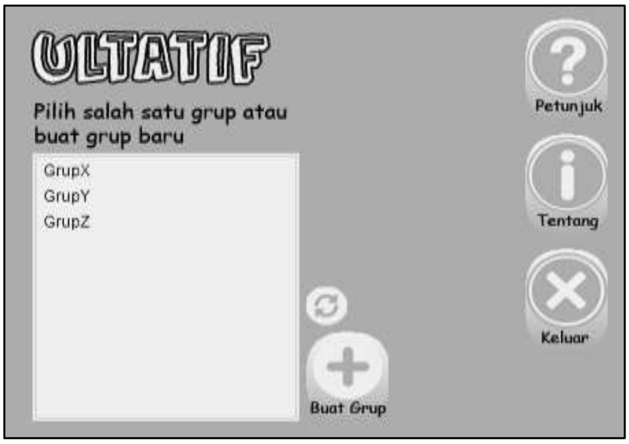
Untuk memainkan game, silahkan login dulu

IP Server: localhost

Port: 2222

Username: Ahmad

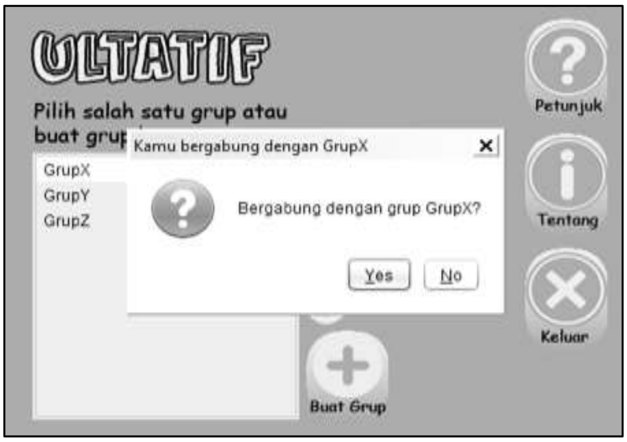
Fig. 9 Login user interface



Pilih salah satu grup atau buat grup baru

GrupX
GrupY
GrupZ

Fig. 10 Group list user interface



Pilih salah satu grup atau buat grup

GrupX
GrupY
GrupZ

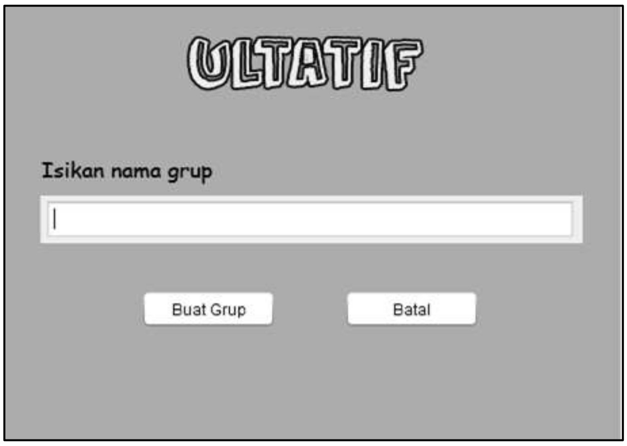
Kamu bergabung dengan GrupX

Bergabung dengan grup GrupX?

Fig. 11 Join group confirmation

E. Create a New Group User Interface

In the interface for creating a new group, players could create a new player group by entering the group name in the text field. The interface is shown in Fig. 12.



Isikan nama grup

Fig. 12 Create group user interface

F. The User Interface of The Group's Player List

The player list in a group is displayed in the group's player list interface, as shown in Fig. 13. The names of the players are displayed in text area. There is also a "play" button to start the game.

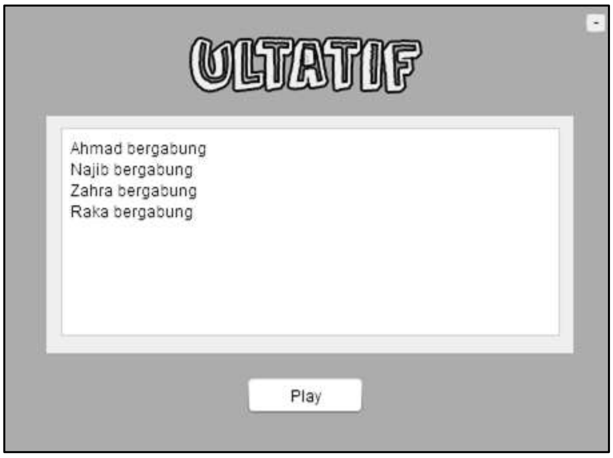
G. Player (Client-side) Program Examination

The program is examined by the black-box method to determine its execution and function in the client (player) and server program working as they should. The program was run to analyze the algorithms and functions. The examination is valid if the result is the same as expected. The results obtained are presented in Table II. These results indicate that both client and server programs run properly.

H. Server Performance Examination as a Player Group Moderator

Server performance is examined to determine the server to manage the player's data, players, and player groups successfully. The experiment results are used to analyze server capability to manage players in several groups and their data traffic without missing or unsent data.

Experiments are conducted by observing and analyzing client requests data from server and broadcast data trough server to several players in the right group. In one experiment session, one server is used with six group players, where the first and second groups consisted of two players, the third and fourth groups, consisted of three players, and the fifth and sixth groups consisted of four players.



Ahmad bergabung
Najib bergabung
Zahra bergabung
Raka bergabung

Fig. 13 Player list interface

TABLE II
BLACK-BOX EXAMINATION RESULT

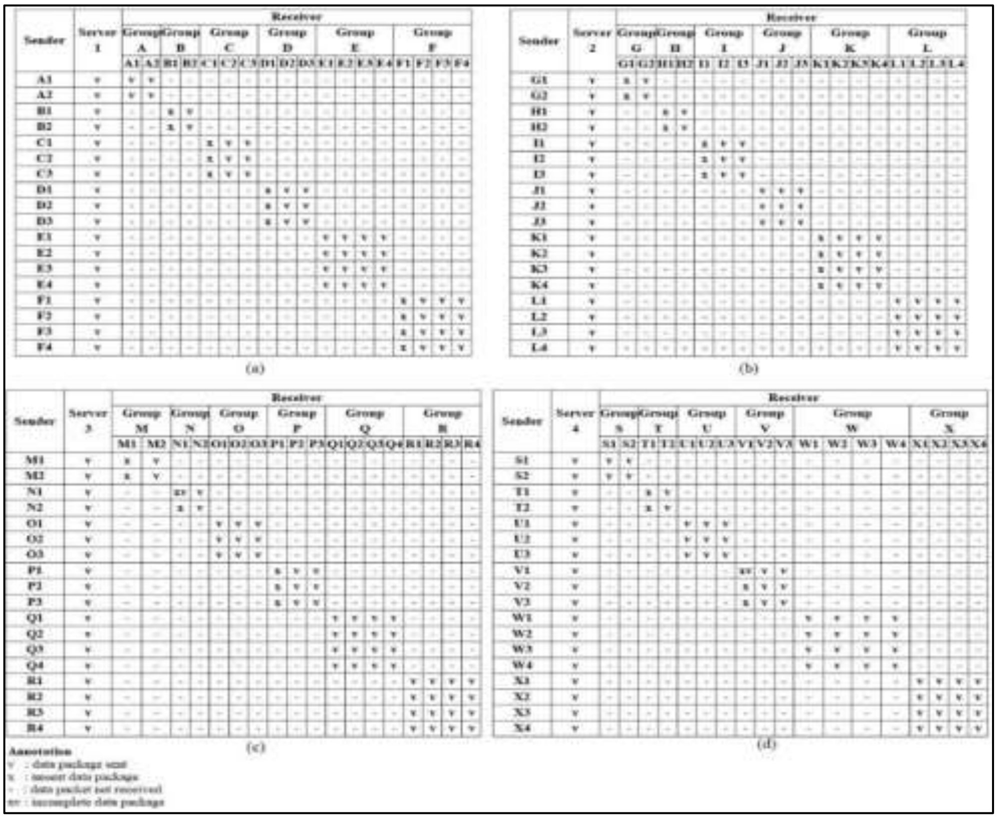
Experiment Scenario	Expected Result	Result
Server open connection (Player request connection)	The login data player appears on the server monitor	Valid
The server closed the connection	All saved data is deleted	Valid
Player login (Data entered is incomplete)	The system will not save the player data and the notification “Data not complete yet”	Valid
Player joins an available group (Player click on group list)	The option (Yes / No) appears “Do you want to join the group?”	Valid
Player creates new group (group name not filled)	Group not saved and notification appears (Group name must be filled)	Valid
Arithmetic quiz questions appear (Player inputting another character instead of numbers)	Textfield does not detect the input character.	Valid
Quiz questions appear (Player does not answer)	After 10 seconds the pop-up questions automatically close and the answer is wrong	Valid
Player sends a short message	All player computers in the same group display the short message	Valid
Player completes the game (wins)	Opponents get information that there are players who have won the game	Valid
Player exits the game	The option (Yes / No) appears “Do you want to exit the game”?	Valid

The server has different specifications, where the computer and its specifications are presented in Table III. The player (client-side) computer was not specified. However, a computer with 2 GB of RAM is used. From experiments could be obtained data presented in Fig. 14.

Based on the examination, varied results were obtained. When the first server was used in the experiment, there are many data packages not sent to the player (client), as shown in Figure 14(a). Similarly, with the second server, some data packages were not received by several clients, as shown in Figure 14(b). The first and second servers have the same processor specifications, although the second has the bigger RAM. However, it has not a significant impact on data package transmission.

TABLE III
COMPUTER EXAMINATION SPECIFICATIONS FOR SERVER

Computer	Processor	RAM	OS
Server 1	AMD E1-1200 APU with Radeon(tm) HD Graphics 1.40 GHz.	2,00 GB	32-bit *Operating System*
Server 2	AMD E-300 APU with Radeon(tm) HD Graphics 1.30 GHz.	4.00 GB	64-bit Operating System
Server 3	Intel(R) Core(TM) i3-4030U CPU 1.90 GHz.	4,00 GB	64-bit Operating System
Server 4	Intel(R) Core(TM) i7-7700HQ CPUCPU 2.81 GHz.	8,00 GB	64-bit Operating System



Sender	Server 1	Receiver																	
		Group A		Group B		Group C			Group D			Group E				Group F			
		A1	A2	B1	B2	C1	C2	C3	D1	D2	D3	E1	E2	E3	E4	F1	F2	F3	F4
A1	v	v	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
A2	v	v	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
B1	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
B2	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
C1	v	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-	-	-	-
C2	v	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-	-	-	-
C3	v	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-	-	-	-
D1	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
D2	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
D3	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
E1	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
E2	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
E3	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
E4	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
F1	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	x	v	v	v
F2	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	x	v	v	v
F3	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	x	v	v	v
F4	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	x	v	v	v

(a)

Sender	Server 2	Receiver																	
		Group G		Group H		Group I			Group J			Group K				Group L			
		G1	G2	H1	H2	I1	I2	I3	J1	J2	J3	K1	K2	K3	K4	L1	L2	L3	L4
G1	v	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
G2	v	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
H1	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
H2	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
I1	v	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-	-	-	-
I2	v	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-	-	-	-
I3	v	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-	-	-	-
J1	v	-	-	-	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-
J2	v	-	-	-	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-
J3	v	-	-	-	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-
K1	v	-	-	-	-	-	-	-	-	-	-	x	v	v	v	-	-	-	-
K2	v	-	-	-	-	-	-	-	-	-	-	x	v	v	v	-	-	-	-
K3	v	-	-	-	-	-	-	-	-	-	-	x	v	v	v	-	-	-	-
K4	v	-	-	-	-	-	-	-	-	-	-	x	v	v	v	-	-	-	-
L1	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
L2	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
L3	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
L4	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v

(b)

Sender	Server 3	Receiver																	
		Group M		Group N		Group O			Group P			Group Q				Group R			
		M1	M2	N1	N2	O1	O2	O3	P1	P2	P3	Q1	Q2	Q3	Q4	R1	R2	R3	R4
M1	v	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
M2	v	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
N1	v	-	-	xv	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
N2	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
O1	v	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-	-	-	-
O2	v	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-	-	-	-
O3	v	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-	-	-	-
P1	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
P2	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
P3	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
Q1	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
Q2	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
Q3	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
Q4	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
R1	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
R2	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
R3	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
R4	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v

(c)

Sender	Server 4	Receiver																	
		Group S		Group T		Group U			Group V			Group W				Group X			
		S1	S2	T1	T2	U1	U2	U3	V1	V2	V3	W1	W2	W3	W4	X1	X2	X3	X4
S1	v	v	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
S2	v	v	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
T1	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
T2	v	-	-	x	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-
U1	v	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-	-	-	-
U2	v	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-	-	-	-
U3	v	-	-	-	-	v	v	v	-	-	-	-	-	-	-	-	-	-	-
V1	v	-	-	-	-	-	-	-	xv	v	v	-	-	-	-	-	-	-	-
V2	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
V3	v	-	-	-	-	-	-	-	x	v	v	-	-	-	-	-	-	-	-
W1	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
W2	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
W3	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
W4	v	-	-	-	-	-	-	-	-	-	-	v	v	v	v	-	-	-	-
X1	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
X2	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
X3	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v
X4	v	-	-	-	-	-	-	-	-	-	-	-	-	-	-	v	v	v	v

(d)

Annotation
v : data package sent
x : unsent data packages
- : data packet not received
xv : incomplete data packages

Fig. 14 Server examination result

When the higher specification server is used, the results are better than the previous experiments that the unsent data package could be reduced, as shown in Figure 14(c). The last experiment using the highest server specification is intended to decrease the failure of sending a data package that has the best results, as shown in Figure 14(d). There are two of 18 players (clients) who were not sent the data package.

Overall experiments show that unsent data packages often occur when it received by the first player. From white-box observations of data entering and leaving the server, it indicates that this is due to the initial data package sent by the first player that could not be received and processed on the first computer's player.

Experiments showing there is also an incomplete packet data received by the client. This occurrence is due to the processing of data packages on the client or server, which corrupts the data; thus, the data package is incomplete. Data corruption could affect the next data packages send and receive. However, this rarely happens to the system.

Although there were several data package sending failures, the examination results generally showed the server's success as a moderator of the player group. From the experiment, it was shown that none of the data packages were sent to other groups. The data package was successfully sent to the players who are in a group of sending players. Based on the experiment data, the percentage of success in sending data can be measured with Eq. (1).

$$data\ sent\ successfully = \frac{\sum sent\ correctly}{\sum data\ sending} x100\% \tag{1}$$

From Eq. (1), the experiment results are the first, and the second server shows the success of sending data rate is 77.70%, whereas the third is obtained 83.30%, and the fourth with the result 88.80% as shown in the graph in Fig. 15.

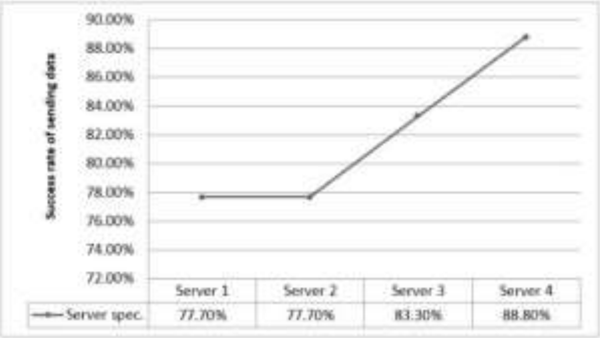


Fig. 15 The success rate of the server sending data

IV. CONCLUSION

Based on the discussion and examination results, it can be concluded that the moderator system in the educational multiplayer snake and ladder game works successfully. Experiment with a server with different specifications shows different results; the first and second servers achieved a 77.7% rate of success in sending data. The third showed 83.3%, and the fourth reached 88.8%. This server's performance is influenced by computer specifications, especially the combination of the processor and RAM, where the higher the computer specifications, the higher the success rate of sending game players data.

REFERENCES

[1] G.-J. Hwang, L.-H. Yang, and S.-Y. Wang, "A concept map-embedded educational computer game for improving students' learning performance in natural science courses," *Comput. Educ.*, vol. 69, pp. 121–130, Nov. 2013, doi: 10.1016/j.compedu.2013.07.008.

[2] Arbianingsih, Y. Rustina, T. Krianto, and D. Ayubi, "Developing a health education game for preschoolers: What should we consider?," *Enferm. Clínica*, vol. 28, pp. 1–4, Feb. 2018, doi: 10.1016/S1130-8621(18)30025-1.

[3] M. Hartono, M. A. Candramata, K. N. Adhyatmoko, and B. Yulianto, "Math Education Game for primary school," in *2016 International Conference on Information Management and Technology (ICIMTech)*, Bandung, Indonesia, Nov. 2016, pp. 93–96, doi: 10.1109/ICIMTech.2016.7930309.

[4] R. E. N. Arifah, S. Sukirman, and S. Sujalwo, "Pengembangan Game Edukasi Bilomatika untuk Meningkatkan Hasil Belajar Siswa pada Mata Pelajaran Matematika Kelas 1 SD," *J. Teknol. Inf. Dan Ilmu Komput.*, vol. 6, no. 6, p. 617, Dec. 2019, doi: 10.25126/jtiik.2019661310.

[5] G. Hookway, Q. Mehdi, T. Hartley, and N. Bassey, "Learning physics through computer games," in *Proceedings of CGAMES'2013 USA*, Louisville, KY, Jul. 2013, pp. 119–126, doi: 10.1109/CGames.2013.6632617.

[6] S. L. Rahayu and F. Fujiati, "Penerapan Game Design Document dalam Perancangan Game Edukasi yang Interaktif untuk Menarik Minat Siswa dalam Belajar Bahasa Inggris," *J. Teknol. Inf. Dan Ilmu Komput.*, vol. 5, no. 3, p. 341, Aug. 2018, doi: 10.25126/jtiik.201853694.

[7] J. M. Rodríguez Corral, A. Civit Balcells, A. Morgado Estévez, G. Jiménez Moreno, and M. J. Ferreiro Ramos, "A game-based approach to the teaching of object-oriented programming languages," *Comput. Educ.*, vol. 73, pp. 83–92, Apr. 2014, doi: 10.1016/j.compedu.2013.12.013.

[8] D. Topalli and N. E. Cagiltay, "Improving programming skills in engineering education through problem-based game projects with Scratch," *Comput. Educ.*, vol. 120, pp. 64–74, May 2018, doi: 10.1016/j.compedu.2018.01.011.

[9] S. A. Pramuditya, M. S. Noto, and H. Purwono, "Desain Game Edukasi Berbasis Android pada Materi Logika Matematika," *JNPM J. Nas. Pendidik. Mat.*, vol. 2, no. 2, p. 165, Oct. 2018, doi: 10.33603/jnpm.v2i2.919.

[10] D. Damayanti, M. F. Akbar, and H. Sulistiani, "Game Edukasi Pengenalan Hewan Langka Berbasis Android Menggunakan Construct 2," *J. Teknol. Inf. Dan Ilmu Komput.*, vol. 7, no. 2, p. 275, Feb. 2020, doi: 10.25126/jtiik.2020721671.

[11] R. M. P. Siregar and E. Sudarmilah, "Rancang Bangun Virtual Reality Educational Game Penanggulangan Sampah berbasis Android untuk Anak Usia Sekolah Dasar," *JUITA J. Inform.*, vol. 7, no. 1, p. 49, May 2019, doi: 10.30595/juita.v7i1.4315.

[12] A. Yusnita and T. Rija'i, "Implementasi Algoritma Shuffle Random pada Pembelajaran Panca Indra Berbasis Android," *JUITA J. Inform.*, vol. 7, no. 1, p. 19, May 2019, doi: 10.30595/juita.v7i1.3647.

[13] K. Siu, M. Guzdial, and M. O. Riedl, "Evaluating singleplayer and multiplayer in human computation games," in *Proceedings of the International Conference on the Foundations of Digital Games - FDG '17*, Hyannis, Massachusetts, 2017, pp. 1–10, doi: 10.1145/3102071.3102077.

[14] E. B. Hurlock and R. M. Sijabat, *Psikologi perkembangan: suatu pendekatan sepanjang rentang kehidupan*. Erlangga, Jakarta, 1990.

[15] M. Fikri, "Rancang Bangun Aplikasi Game Battleship Multi Player Pada Jaringan LAN," Thesis, Universitas Islam Negeri Sultan Syarief Kasim, Riau, 2013.

[16] M. Ika Wijayanti, "Perangkat Lunak Permainan Ular Tangga Multiplayer Berbasis Jaringan," Thesis, Universitas Islam Negeri Sunan Kalijaga, Yogyakarta, 2013.

[17] M. F. Syahputra, R. F. Rahmat, and I. I. Aksa, "Perancangan Permainan Multiplayer Truf Gembira Berbasis Android," Medan, Mar. 2013, vol. 1, p. 5, Accessed: Nov. 02, 2018. [Online]. Available: https://www.researchgate.net/publication/302896024_Perancangan_Permainan_Multiplayer_Truf_Gembira_Berbasis_Android.

[18] B. D. C. Putri, R. Kridalukmana, and E. D. Widianto, "Perancangan Aplikasi Permainan Multiplayer Gobak Sodor Berbasis Flash Di Lingkup Jaringan Lokal," *J. Teknol. Dan Sist. Komput.*, vol. 4, no. 2, p. 259, Apr. 2016, doi: 10.14710/jtsiskom.4.2.2016.259-265.

[19] A. J. Pratama, A. A. K. O. Sudana, and I. N. Piarsa, "Rancang Bangun Game Kartu Spirit Berbasis Android dengan Fitur Online Multiplayer," *Lontar Komput. J. Ilm. Teknol. Inf.*, pp. 49–60, 2015.

[20] A. Sivaraman, T. Mason, A. Panda, R. Netravali, and S. A. Kondaveeti, “Network architecture in the age of programmability,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 50, no. 1, pp. 38–44, Mar. 2020, doi: 10.1145/3390251.3390257.

[21] G. He and L. Li, “Design and Realize Communication System Based on LAN of C/S Architecture,” in *2009 WRI World Congress on Software Engineering*, Xiamen, China, 2009, pp. 78–81, doi: 10.1109/WCSE.2009.81.