RFC 42123 Nicolas MAGERE, Quentin LEFFRAY, Matthieu OSSATO-BOURGEON
Intended Status: Experimental Epitech
Expires: May 20, 2013 December 14, 2012

RTCP: A Communication Protocol For R-Type Project

Abstract

This memorandum describes RTCP, the communication protocol for a project from Epitech's school : R-Type.

This protocol is used for designing a "Shoot Them Up" game, based on server-client (in both ways) relation.

The server's aim is to interpret all incoming requests from clients to manage at both several games and all connections.

Client for its is to contact the server on the one hand to give his EP (End Point) to connect to it, and on the other hand to send regularly data to continue the game. The following document describes server-client interactions.

Table of Contents

1. Introduction

The RTCP (R-Type Communication Protocol) has been designed for creating a "Shoot Them Up" game, based on server-client relation. The game is specifically designed to host a party up to 4 players.

There is a server, which hosts all games created. It manages all game physics and rules.

From there, several clients connects to the server to play the game. They are used to receive data from server, and to interact with the player (input and graphics).

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Terms definition

- Endpoint : defines a service, designed by his IP-address.
- Fallback mode : defines the mode of the server that owns predefined ports, on which it attempts to connect until success or no more ports are available.

3. Concept of RTCP protocol

The RTCP protocol has been developed on systems using the TCP and UDP network protocol.
It is a binary one, and is based on client-server interactions.

The TCP network is used for all data which are not related with the in-game itself, but all the previous important information to create it. Like this, every important data, (such as map, textures, etc) will be transmitted without data loss.

The UDP protocol is used for all in-game transmitted data (like each client coordinates).
The UDP is used for performances : it is important the data transfers quickly, because there will be many packet transfers, and the loss is negligible.

3.1 Concept of client-server and server-client interactions

There is a list of commands for the client-server interaction :

Each client has to CONNECT to the server before trying any other command.
After that, he can CREATE_ROOM, JOIN_ROOM, or LEAVE_ROOM (see part 6.1 Room definition).

CREATE_ROOM tells the server the client wants to make a room to play a game.
JOIN_ROOM asks the server to enter in a room already created, to play a game with other clients.
LEAVE_ROOM tells the server the client quits a room and can no longer play with the other clients which still are in the room.

The client can also INVITE_PLAYER. He tells a second client to JOIN_ROOM the room where the first one is, by sending the INVITE_PLAYER command through the server (no client-client dialogue). After the second client has received the command INVITED by the server, he can JOIN_ROOM or just no respond.

SET_GAME_PARAM is used by the creator of the room, to set the rules of the game before the next play.

LAUNCH_GAME is used by the creator of the room to start the game.

PING is a command sent by the client to the server, to verify the connection between them is still valid.

ACK is the success code or the error code sent to the client.

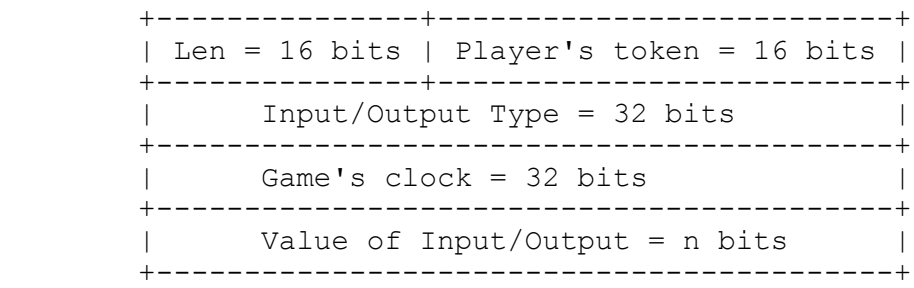
There is a list of commands for the server-client interaction :

The server responds to a client CREATE_ROOM's command by an ANSWER_CREATE_ROOM, to give him some information about the room he created.

The server MAY upload the game data to client through:
START_STREAM, which begins the transfer process with the client.
STREAM, which represents the data being upload to the client.
STOP_STREAM, which means the end of transfer processing.

4. Standard packet format for data transfer

4.1 Scheme of an UDP packet



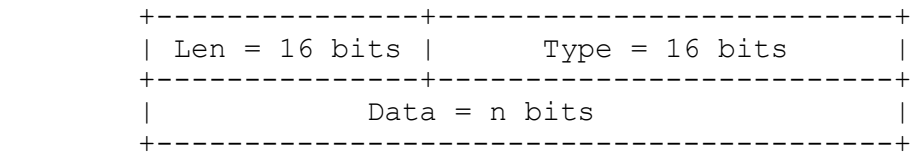
4.1.2 Structure of an UDP packet

An UDP packet is structured with 5 different fields :

- Len represents the total length of the packet.
- Player's token is an identifying number which differs for each client, in order to distinguish all of them.
- Input Type / Output Type represents the type of the packet, sent for INPUT (client to server transmission), or OUTPUT (server to client transmission).
- Game's clock contains the server game's clock, for the packet before being sent. This is used to synchronize players' clocks.
- Value of Input/Output represents the data itself in the case of data transfer.

4.2 TCP packet format

4.2.1 Scheme of a TCP packet



4.2.2 Structure of a TCP packet

A TCP packet is structured with 3 different field :

- Len represents the total length of the packet.
- Type contains the request or command type.
- Data represents the arguments of the command.

5. Errors identifiers

5.1 1xx : Success

- 101 : Success.

5.2 2xx : Client errors

- 201 : User refused.
- 202 : User did not respond.
- 203 : Client failed to connect.
- 204 : Client understands request but refused to process it.

5.3 3xx : Server errors

- 301 : Server failed to transfer request.
- 302 : Server failed to process request.
- 303 : Server refused this command.
- 304 : Server understands request but refused to process it.
- 305 : Insufficient privileges.
- 306 : Transferred user not found.

5.4 4xx : Global errors

- 401 : Invalid request.
- 402 : Incomplete request.
- 403 : Forbidden request.
- 404 : Request not found.

6. Commands

6.1 Room definition

A room is a virtual space where clients can be together in groups from 1 to 4 in order to start a game.

By default when clients connect to the server, they arrive in a special room called "Hall", a kind of waiting-area.
From there, each client can create a game, which means that they leave the "Hall" room to go to the room they created. In case of leaving a room, each client is redirected to the "Hall" room again.

6.2 Commands from a client to a server

6.2.1 CONNECT

Command :
CONNECT = 1201

Parameters : <username>:<password>

Description :
This command allow users to register on the server. It is used for the creation of the 'user account' on that game server.

The parameters are the following:
- Username : name of the user.
- Password : the password chosen by user.

Notice that this command MUST be implemented.

6.2.2 CREATE_ROOM

Command :
CREATE_ROOM = 1202

Parameters : Nothing

Description :
This command is sent by a user that wants to create a room on the server. Notify that the client JOIN_ROOM automatically after this command.

Notice that this command MUST be implemented.

6.2.3 LEAVE_ROOM

Command :
LEAVE_ROOM = 1203

Parameters : <RoomId>

Description:
This command is used by a user who wants to leave a room he has created or joined before.

The parameters are the following:
- RoomId : id of the room to leave.

Notice that this command MUST be implemented.

6.2.4 JOIN_ROOM

Command :
JOIN_ROOM = 1204

Parameters : <RoomId>

Description :
This command is used by a user who wants to join a created room.

The parameters are the following:
- RoomId : id of the room to join

Notice that this command MUST be implemented.

6.2.5 INVITE_PLAYER

Command :
INVITE_PLAYER = 1205

Parameters : <username>

Description:
This command is used when a user wants to invite another one in a room he has created.

The parameters are the following:
- Username : name of user to be invited.

Notice that this command SHOULD be implemented.

6.2.6 SET_GAME_PARAM

Command :
SET_GAME_PARAM = 1206

Parameters : <key>:<value>

Description: This command is used by a user to change in-game parameters (like players lives number for example)

The parameters are the following:
- Key : Parameter's id to be changed
- Value : New value of the parameter to change

Notice that this command SHOULD be implemented.

6.2.7 LAUNCH_GAME

Command :
LAUNCH_GAME = 1207

Parameters : nothing

Description:
This command is used to launch a game created. Note that a room has to be previously created (see CREATE_ROOM command) by a user before he launches the game.

Notice that this command MUST be implemented.

6.2.8 PING

Command :
PING = 1208

Parameters : nothing

Description :
This command is used every second by a client (a user) to inform the server that he is still connected. Without telling the server he is, the server will erase the client that seems to be disconnected after X seconds (like 10 for example) from his client's list.

Notice that this command MUST be implemented.

6.2.9 READY

Command :
READY = 1209

Parameters: <Endpoint>

Description:
This command is used by the client to notify the server (which has already sent him the command GAME_LAUNCHED) that he is ready for the game.

The parameters are:
- EndPoint : the Endpoint of the ready client.

Notice that this command MUST be implemented.

6.3 Commands from a server to a client

6.3.1 ANSWER_CREATE_ROOM

Command :
ANSWER_CREATE_ROOM = 1301

Parameters: <RoomId>

Description:
This command is used by the server to notify the client that his CREATE_ROOM's request has been accepted, and returns him the RoomId he needs.

Notice that this command MUST be implemented.

6.3.2 START_DATA_STREAM

Command :
START_DATA_STREAM = 1302

Parameters: <fileName>

Description:
This command is used to notify a client that the server is going to transfer him the file "fileName". It is useful for out-game work : to transfer properly and safely files to clients, like downloading textures.

The parameters are:
- FileName : name of the file to be transferred

Notice that this command MAY be implemented.
In case of this command is implemented, the following commands STREAM and STOP_DATA_STREAM MUST be implemented.

6.3.3 STREAM

Command :
STREAM = 1303

Parameters: <fileName>:<data>

Description:
This command is used to transfer data from a file designed by "fileName" to a client. This command have to be called after the command START_DATA_STREAM.

The parameters are:
- FileName : name of the file to be transferred
- Data : data which are sent. Notice that data chunks must be formatted to 1024 bytes. In case of the chunk contains less than 1024 bytes, you MUST use command STOP_DATA_STREAM instead of STREAM.

Notice that this command MAY be implemented. In case of this command is implemented, the following commands START_DATA_STREAM and STOP_STREAM MUST be implemented.

6.3.4 STOP_DATA_STREAM

Command :
STOP_DATA_STREAM = 1304

Parameters: <fileName>:<lastData>:<dataLen>:<hash>

Description:
This command is used to notify a client that the server stops transferring him the file "fileName".

The parameters are:
- FileName : name of the file on which the transfer is stopped.
- LastData : the last data available from the file to be stored.
- DataLen : Length in bytes of the "lastData".
- Hash : a MD5 hash that corresponds to "fileName".

Notice that this command MAY be implemented. In case of this command is implemented, the following commands START_DATA_STREAM and STREAM MUST be implemented.

6.3.5 CLIENT_INVITED

Command :

CLIENT_INVITED = 1305

Parameters: <usernameFrom>:<RoomId>

Description:
This command is used by the server to notify a client that he has been invited by another one, by a previous call to INVITE_PLAYER.

The parameters are:
- UserName : name of the user that invited the client.
- RoomId : the room's id from where the client which invited you id.

Notice that this command SHOULD be implemented.

6.3.6 ACK

Command :
ACK = 1306

Parameters: <error/successCode>

Description:
This command is used by the server to notify a client that a request has been treated with success or that an error occurred.

The parameters are:
- Error/Success Code : A code corresponding to a request sent by the client to the server.

Notice that this command MUST be implemented.

6.3.7 GAME_LAUNCHED

Command :
GAME_LAUNCHED = 1307

Parameters: <timestamp>:<gameclock>

Description:
This command is used by the server to notify a client that the game has been launched and it should switch to the UDP socket to get the in-game stream data.

The parameters are:
- Timestamp : Timestamp used by the server to make the the game's clock corresponds to a certain time (to avoid data loss or bad ordered packets).

- GameClock : Game's clock, MUST be initialized to 0 (the game is just beginning).

Notice that this command MUST be implemented.

7. References

7.1 Normative References

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC1776] Crocker, S., "The Address is the Message", RFC 1776, April 1 1995.

[TRUTHS] Callon, R., "The Twelve Networking Truths", RFC 1925, April 1 1996.

7.2 Informative References

[EVILBIT] Bellovin, S., "The Security Flag in the IPv4 Header", RFC 3514, April 1 2003.

[RFC5513] Farrel, A., "IANA Considerations for Three Letter Acronyms", RFC 5513, April 1 2009.

[RFC5514] Vyncke, E., "IPv6 over Social Networks", RFC 5514, April 1 2009.

Authors' Addresses

Nicolas MAGERE - magere_n
Quentin LEFFRAY - leffra_q
Matthieu OSSATO-BOURGEON - ossato_m

Epitech students

Email: magere_n@epitech.eu
leffra_q@epitech.eu
ossato_m@epitech.eu