R-Type Group Yermi Brie

Request for Comments: Frolian tire sur eux, Inc.

Category: Video Games November 2020

R-Type Protocol (RTP) Transport over UDP and TCP

Abstract

This document describes how the R-Type Protocol (RTP), works using and

UDP and TCP connection.

This protocol requires use of a R-Type server and at least one R-Type

client.

The RTP was developed during the R-Type student project.

Status of This Memo

This document is the official specification of the R-Type Protocol,

and defines an Experimental Protocol for the R-Type community.

Discussion and suggestions for improvement are requested.

Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Frolian tire sur eux Society (2020). All Rights Reserved.

Table of Contents

1. Introduction...........................................................

1.1 Conventions Used in This Document...............................

2. Session Management ....................................................

3. Message Exchange ......................................................

4. Network Protocol ......................................................

4.1 Types ..........................................................

4.2 Packet .........................................................

5. References ............................................................

4. Author's Address ......................................................

1. Introduction

The R-Type protocol has been developed on systems using the TCP

network protocol and UDP network protocol, although there is no

requirement that this remain the only sphere in which it operates.

IP is described in [RFC0791]. TCP is described in [RFC0793].

UDP is described in [RFC0768].

1.1. Conventions Used in This Document

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 [RFC2119].

The key word "oneof" indicates that one and only one of the following field can be filled.

The key word "repeated" indicates the variable is a vector of the type named right after.

2. Session Management

The RTP session management facilities onto the UDP and TCP service is

straightforward. For TCP, the RTP session first requires a creation of

a TCP connection between two peers, one that initiates the connection

request and one that responds to the connection request. The initiating

peer is called the "client", and the responding peer is called the

"server". An RTP server MUST listen for TCP connection requests

on a standard TCP port.

In TCP mode, the client MAY end the RTP session by closing the socket.

3. Message Exchange

In UDP, the user is connected in a party when the RTP server returns a RtypePack packet with a START\_GAME code to the client.

The user MUST be connected using the RtypePack packet with the code START\_GAME to take part in the parties.

After this, both client and server can initiate a RTP binary command.

In some commands, the server SHOULD NOT return a response for avoid

flooding the clients. For furthering details, go to the 4. Section.

The RTP packet’s command MUST be struct-packed with protobuf for better transfert.

4. Network Protocol

4.1 Types

RtypePack::PackType uint\_16

RESPONSE = 0

CREATE\_ROOM = 1

JOIN\_ROOM = 2

LEAVE\_ROOM = 3

START\_GAME = 4

ROOM\_CREATED = 5

ROOM\_JOINED = 6

ROOM\_PLAYER\_JOINED = 7

ROOM\_IS\_FULL = 8

ROOM\_LEFT = 9

ROOM\_PLAYER\_LEFT = 10

TRYING = 100

OK = 200

NOTFOUND = 404

RTypePack::ActionType uint16

OK = 0

BAD = 1

RGamePack::PackType uint\_16

PING = 0

ACTION = 1

COLLISION = 2

LOAD = 40

DESTROY = 41

POSITION = 42

EFFECT = 43

END\_GAME = 44

RGamePack::ActionType uint16

UP = 0

DOWN = 1

LEFT = 2

RIGHT = 3

SHOOT = 4

RGamePack::EntityType uint\_16

PLAYER = 0

ENEMY = 1

ALLY = 2

POWERUP = 3

BASIC\_SHOT = 4

ENEMY\_BULLET = 5

RGamePack::EffectType

SPEED = 0

ATTACKSPEED = 1

SHIELD = 2

LIFE = 3

4.2 Packet

RTypePack:

PackType code = 1

string clientip = 2

uint32 clientId = 3

oneof Packet:

ResponsePacket ResponseContent = 4

LobbyPacket LobbyContent = 5

This command can be invoked either by the client or the server.

This command is only used in TCP.

The “code” MUST be a PackType (described above)

The “clientip” MUST be a string containing the client’s IP address

The “ClientId” MUST be a number beetween 1 and 4 representing the client player ID

ResponsePacket:

ActionType Code = 1

string Content = 2

This command can only be invoked by server

the “Code” MUST be an ActionType (described above)

the “Content” CAN be set as a string containing an error message if the code is set to “BAD”

LobbyPacket:

uint32 PlayerID = 1

string RoomID = 2

uint32 MaxPlayer = 3

string udp\_ip = 4

uint32 port = 5

This command can only be invoked by server

The “PlayerID” MUST be the Id of the player the packet is intended to

The “RoomId” MUST be the Id of the room the packet is about.

The “MaxPlayer” MAY be the maximum number of player of the room the packet is about

The “udp\_ip” MUST be the IP address the client has to send / receive packet to / from with UDP.

The “port” MUST be the port of the “udp\_ip” the client has to send / receive on with UDP.

RGamePack:

PackType code = 1

uint32 playerId = 2

oneof uvar:

ActionPacket actionContent = 3

CollidePacket collidePacket = 4

LoadPacket loadContent = 5

DestroyPacket destroyContent = 6

PositionPacket positionContent = 7

EffectPacket effectContent = 8

This command can be invoked either by the client or the server.

The “Code” MUST be set to a RGamePack::PackType (described above)

The “PlayerId” MUST be set to an uint32 beetween 1 and 4 representing the client’s player ID

ActionPacket:

uint32 Id = 1

ActionType type = 2

oneof sLvl:

uint64 shootLevel = 3

This command can only be invoked by client

The “Id” MUST be set to an uint32 beetween 1 and 4 representing the client’s player ID

The “type” MUST be set to a ActionType (described above)

CollidePacket:

repeated uint64 Ids = 1

This command can only be invoked by client

The “Ids” MUST be set to a vector of uint64 containing IDs of game entities

Vector2:

double X = 1

double Y = 2

This command can be invoked either by the client or the server.

the “X” MUST be a double

the “Y” MUST be a double

LoadPacket:

uint32 Id = 1

EntityType type = 2

string fileName = 3

Vector2 position = 4

Vector2 scale = 5

Vector2 size = 6

Vector2 animFrameInfo = 7

This command can only be invoked by the server.

The “Id” MUST be the Id of the entity to be loaded by the client.

The “type” MUST be set to an EntityType described above and MUST be the type of the entity to load.

The “filename” MUST be the string describing the name of the file to be loaded by the client.

The “position” MUST be the position on which the entity will first be set.

The “scale” MUST be the scale on which the entity sprite will be scaled.

The “size” MUST be the size of the sprite in the texture asset.

The “animFrameInfo” “X” MUST be the number of frames and “Y” MUST be the frame rate of the sprite texture.

DestroyPacket:

repeated uint32 Ids = 1

This command can only be invoked by the server.

The “Ids” MUST be a vector of uint32 representing game entities ID to destroy.

PositionPacket

repeated uint32 Id = 1

repeated Vector2 position = 2

repeated int32 velocity = 3

bool has\_velocity = 4

This command can only be invoked by the server.

the “Id” MUST be the Id of the entity to be loaded by the client

the “velocity” CAN be a int32 representing the speed of an entity

the “position” MUST be a Vector 2 (see bellow) representing the position of the entity in game

the “has\_velocity” MUST be set to TRUE or FALSE depending of if the entity has speed

5. References

[RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791,

September 1981.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7,

RFC 793, September 1981

[RFC0768] Postel, J., "User Datagram Protocol", RFC 793,

28 August 1980

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate

Requirement Levels", BCP 14, RFC 2119, March 1997.