Design and Implementation of Streaming Media Processing Software based on RTMP

Xiaohua Lei, Xiuhua Jiang, Caihong Wang
School of Information Engineering
Communication University of China
Beijing, China

Abstract—**Along with the prosperity of streaming media business such as OTT (Over the Top) and Live Social Video Broadcasting Platform in recent years, the research on the streaming media processing technique has become a hot issue. This paper makes comprehensive specification of the RTMP (Real Time Message Protocol) in Adobe's Flash streaming media system, and introduces the implementation of software that can download RTMP-based streaming media. The software follows the RTMP specification, implements the connection with the server using Socket API in C language, processes multimedia data coming from server and saves it as a FLV (Flash Video) format file. The method mentioned in the structure of the software, can be used as the basis for the development of a more complete RTMP-based streaming media processing software.**

Keywords- RTMP; streaming media; processing

I. INTRODUCTION

In recent years, with the increasing of network bandwidth, and the development of multimedia compression technology, network multimedia technology is widely used in the Internet. Global market of network multimedia is at a high speed of development, and gradually replaced the traditional Internet-based text and pictures. According to the statistics of Cisco's Visual Networking Index , in 2005 network multi-media traffic accounted for only 5% of the total global Internet traffic, while in 2011 this proportion had increased to 40%, and this proportion is expected further enhance to 62% by 2015 [1]. At the same time, multimedia technology has broken through the limitations of the computer, into areas such as tablet PCs and smart phones. An era of "Video Everywhere" is on the way.

In such environment, streaming media technology is becoming more and more important on the Internet. Nowadays many streaming media systems are based on RTP / RTCP (Real-time Transport Protocol / Real-time Transport Control Protocol) [2]. But streaming media based on RTP / RTCP which usually use UDP (User Datagram Protocol) as its transport-level protocol often has problems such as packet loss when it transmitted on the Internet, thus lead to serious reduction of the QoS (Quality of Service) [3]. With the popularity of Flash technique on the Internet in recent years, RTMP-based (Real Time Message Protocol) streaming media system has been widely used. On the one hand, RTMP-based streaming media use TCP (Transfer Control Protocol) as its transport-level protocol, effectively prevent the occurrence of packet loss and other problems; on the other hand, due to using Flash as the client, which installed on 98% percent of the world's desktop computers [4], user don't need to install any other software or plug-in to play the streaming media.

Owning to these advantages, a lot of online multimedia providers have chosen RTMP-based streaming media system to build their business platform. Our investigation shows that the TV stations such as CNTV VOD (Video-On-Demand), China Education TV, Henan TV, Shenzhen TV, as well as network multimedia service providers such as 6.cn and Sina have chosen this system. The RTMP that used in this system therefore get a wide range of applications.

RTMP is the Adobe's network protocol used to transmit audio, video and data between its Flash platforms. As the basis of processing of RTMP, the paper will introduce two important structures in the RTMP: Message and Chunk; and will describe the Message flow during the process of playing RTMP-based streaming media. Following the specification mentioned above, we have implemented a software that can download the RTMP-based streaming media.

II. RTMP SPECIFICATION

RTMP belongs to the application-level protocol, and usually TCP is accompanied with it as transport-level protocol. The basic unit of the RTMP to transmit information is Message. During transmission, for consideration of multiplexing and packetizing multimedia streams, each Message will be split into some Chunks.

A. Message

Message is the basic unit of the RTMP. When the connection is established, Message is sent to communicate between the client and the server. There are about a dozen types of Message. Different Message Payloads play different roles in a RTMP connection, and are distinguished from Message Type field in Message Header.

Message Header format shown in the Figure 1 contains four parts: Message Type field indicates the type of the Message. Payload Length field indicates number of bytes of Message's payload. Timestamp field identifies the timestamp of the Message. Stream ID field identifies the Stream of Message [5].

Message Payload content is depending on the Message Type, such as video, audio, data and AMF (Action Message Format) Command.

978-1-4673-0964-6/12/$31.00 ©2012 IEEE

Figure 1. Message Format

The Messages shown in the Table I will be used in the process of playing a RTMP-based streaming media. The specific usage will be described in detail in Chapter IV.

In the process of playing a streaming media, the client can send Command Message such as "connect", "createStream", "play", "pause" to control the playback of streaming media [6]. These Command Message are a type of Message which carrying the AMF encoded data. AMF (Action Message Format) is a compact binary format that is used to transmit ActionScript built-in Object, such as Number, Boolean, String, Object, and Array, between two endpoints [7]. Just following the AMF protocol, the receiver can decode AMF encoded data into the appropriate Object. There are two versions of the AMF format: AMF0 and AMF3, while the coding efficiency of the latter is higher.

TABLE I. MESSAGE USED WHEN PLAYING A STREAMING MEDIA

Type	Message Name	Message function
1	Set Chunk Size	Notify the peer the chunk size to use.
3	Acknowledgement	Send it after receiving bytes equal to the window size
4	User Control Message	Notify the peer the user control events.
5	Window Acknowledgement Size	Inform the peer which window size to use
6	Set Peer Bandwidth	Update the output bandwidth of the peer.
8	Audio Data	Carry Audio Data
9	Video Data	Carry Video Data
17	Command Message	Carry the AMF encoded Command,AMF0 encoded
20	Command Message	ditto,AMF3 encoded

B. Chunk

Message need to be split into a number of Chunks when it transmits data in the network. Chunk provides multiplexing and packetizing services for a higher-level multimedia stream protocol. RTMP Chunk Stream Protocol prescribes that the Payload of each Message is divided into fixed-size Chunks (except the last one).

Chunk format is shown in the Figure 2. The Chunk Header can be divided into three parts: Chunk Basic Header, Chunk Message Header, and Extended Time Stamp. Chunk Basic Header contains the information of this Chunk, and is used to identify the Chunk Stream which Chunk is belong to; The content of Chunk Message Header is the same as the Message Header before the Message is split into Chunks, containing information of the Message which Chunk is belong to; Extended Time Stamp hardly used, and the field is appeared only when the Time Stamp is not enough [8].

Figure 2. Chunk Format

For the consideration of data compression, Chunk Header is variable-length. When the first Chunk Header's field and the second Chunk Header's field are the same, the second Chunk Header's field is often omitted. The length of Chunk Basic Header is 1-3 bytes, Chunk Message Header is 0, 3, 7, 11 bytes, and Extended Time Stamp is 0, 4 bytes.

C. Message to Chunks

The way of Message split into Chunks is shown in the Figure 3. The Payload of a Message is cut into the same size blocks of data (except the last one), and adds the Chunk Header in front of each data block, then constitutes the Chunk Stream. Streaming media which use the RTMP is transferred in form of Chunk Stream on the Internet.

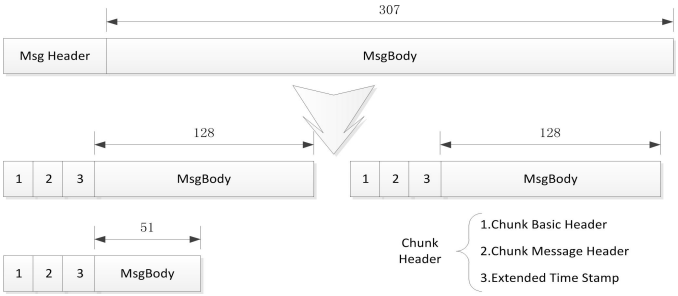

Figure 3. Message to Chunks (Chunk size is 128)

III. TECHNOLOGY OF PLAYING STREAMING MEDIA

A. Client

Playing a RTMP-based streaming media, under normal circumstances, need to use the Flash application as client. User can use ready-made Flash web player to play streaming media, for example JwPlayer or FlowPlayer, and can also use the ActionScript language's API (Application Programming Interface) to build a Flash application themselves as client receiving streaming media [9].

B. Server

Server is used to transmit multimedia data to the client in the form of RTMP. There are many kinds of server that support the RTMP, such as Flash Media Server and Red5. Streaming media server that supports the RTMP doesn't directly store streaming media content. There are some of Applications in the streaming media server, and streaming media content are stored in these Applications. When a client plays streaming media through the RTMP, it should connect to application deployed inside the server first. After the establishment of the connection with the application, the client can play the streaming media contained in the application.

C. Format of the streaming media URL

When playing a RTMP-based streaming media, the client needs to use the URL to locate the streaming media on the server. The format of RTMP-based Streaming media's playback URL is "rtmp://hostname:port/appName/playpath" [9]. The "rtmp" represents the type of protocol; the "hostname" represents the address of the server; "port" represents server's port that receive connection; the "appName" represents the name of the application on the server; the "playpath" represents the streaming media file path in the application.

D. The relationship between NetConnection and NetStream

RTMP regulate that playing a streaming media has two premise steps: the first step is to establish a NetConnection; the second step is to establish a NetStream [9]. The NetConnection represents the higher-level connection between server and client. The NetStream represents a channel of transmitting multimedia data. There is only one NetConnection between Server and Client, but the NetConnection can create a lot of NetStreams. Their relationship is shown in Figure 4.

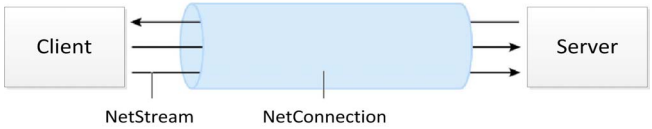

Figure 4. NetStream and NetConnection

IV. Process of Playing Streaming Media

Playing a RTMP-based streaming media needs to go through the following steps: Handshake, Create Connection, Create Stream, and Play. A RTMP connection begins with a Handshake; Create Connection stage is used to establish the NetConnection between the client and server; Create Stream stage is used to establish the NetStream between the client and server; Play stage is used to transmit video and audio data.

A. Handshake

1) The client sends C0, C1 block. Server receives the C0 or C1 and then sends S0 and S1.

2) When receiving all the S0 and S1, the client starts sending C2. When receiving all the C0 and C1, the server starts sending S2.

3) When the client received S2 and the server received C2, the Handshake is complete.

The process of Handshake is shown in the Figure 5 [8].

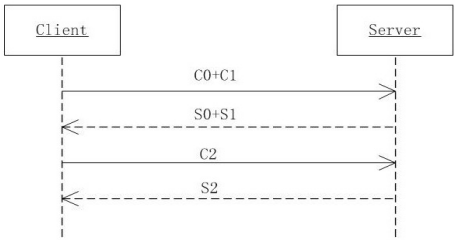

Figure 5. Handshake

B. Create Connection

1) The client sends a Command Message "connect" to the server to request to establish a NetConnection with a server application instance.

2) After receiving the "connect" Command Message, the server sends the Message "Window Acknowledgement Size" to the client, and connect to the application mentioned in the Command Message.

3) The server sends the Message "Set Peer Bandwidth" to the client to update the output bandwidth.

4) After dealing with the set bandwidth Message, the client sends the Message "Window Acknowledgement Size" to the server.

5) The server sends the User Control Message "Stream Begin" to the client.

6) The server sends Command Message "_results" to notify the client the result of the Command.

The Message flow of Create Connection is shown in the Figure 6 [6].

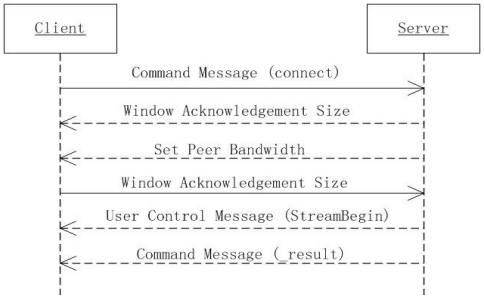

Figure 6. Create Connection

C. Create Stream

1) The client sends a Command Message "createStream" to the server to request to establish a NetStream with a server application instance.

2) The server sends Command Message "_results" to notify the client the result of the Command.

The Message flow of Create Stream is shown in the Figure 7 [6].

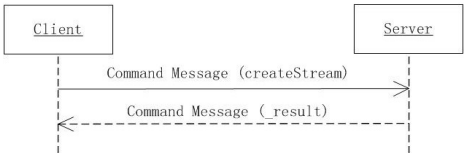

Figure 7. Create Stream

D. Play

1) The client sends the Command Message "play" to the server.

2) On receiving the "play" Command Message, the server sends "Set Chunk Size" Message to notify the client the chunk size used in the stream.

3) The server sends User control Message "StreamBegin" to inform the client that the stream has become functional.

4) The server sends Command Message "NetStream.Play. Start" and "NetStream.Play.reset" to notify the client the "play" Command is successful.

5) After this, the server sends audio and video data ,which the client plays.

The Message flow of Play is shown in the Figure 8 [6].

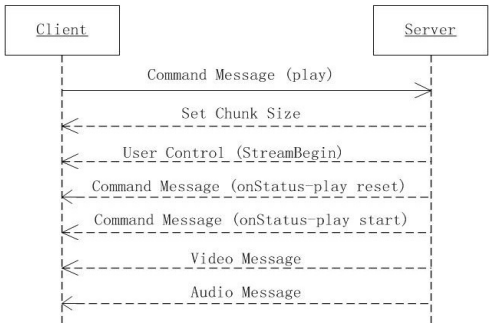

Figure 8. Play

V. IMPLEMENTATION OF THE SOFTWARE

A. The process of the software

RTMP support Flash applications as client to play the streaming media it transmits. However, the Flash application can only calls ActionScript API to display streaming media on the screen, hides the source code that it handles the RTMP. As a result, it is not convenient to study the features of the RTMP and to handle the data it transmits. Therefore, to study the detail of the RTMP, and save and analysis the data it transmits, it is necessary to implement the RTMP client by ourselves.

The RTMP client developed by ourselves follows the rules of the RTMP, uses Socket API in C language, implements the entire process of Message's sending/receiving of the client, and writes the received video and audio data to hard-disk, stores as a FLV format file. The development environment is Visual Studio 2010. The entire software is divided into different modules according to the different steps of playing the RTMP streaming media introduced in Chapter IV, and in accordance with the order of execution: Establishment of the Socket Connection, Handshake, Create NetConnection, Create NetStream, Play, Handling the multimedia data and Writes into FLV files. The process of the software is shown in Figure 9.

B. Module structure

Each module has to complete the process of Message's sending/receiving steps of the client introduced in Chapter IV. First, the module calls the function of Sending Message to encode data and send it to server; and then it calls the function of Receiving Message to receive and parse the data that server returns. Finally, the module judges whether itself is complete depending on the Message received from server. If the function of the module is completed, the next module would be called; otherwise the loop of Message's sending /receiving would be continued until the completion of this module. For example, the structure of Create NetConnection module is shown in the Figure 10.

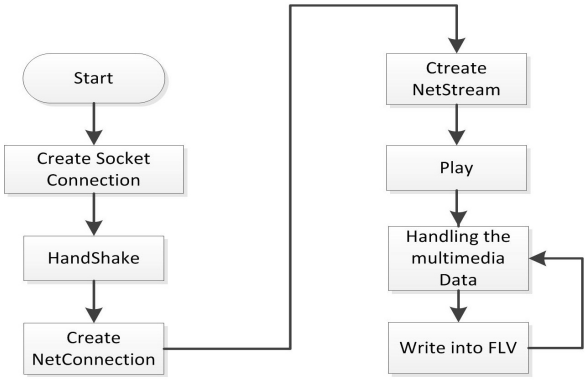

Figure 9. The process of the software

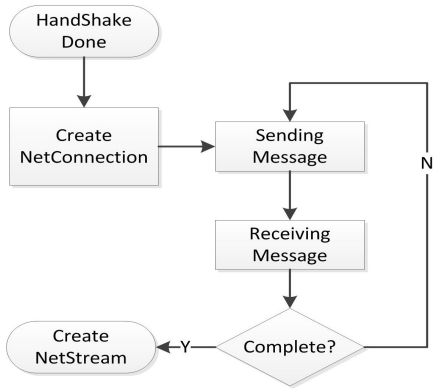

Figure 10. Module structure (Create NetConnection)

C. Message's sending/ receiving

Sending Message is a coding process. First, calling Message Encoding to encode the data that need to be send at the format of Message; Then, calling Chunk Encoding to encode Message to a form of Chunk stream; Next, putting Chunk stream into the Writing Cache; Finally, calling the send() function in the Socket API to send byte stream to the server. Message sending process is shown in Figure 11.

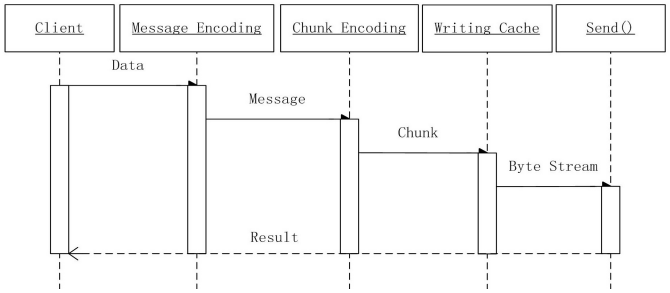

Figure 11. Sending Message

Receiving Message is a decoding process. First, calling the recv () function in Socket API to accepts a byte stream coming from the server, stored in the Reading Cache; Then, reading

byte by byte data in the cache, decoding byte stream at the format of Chunk; Next, decoding the received Chunk to restore the Message; At last, data can be obtained from the Message. Message receiving process is shown in Figure 12.

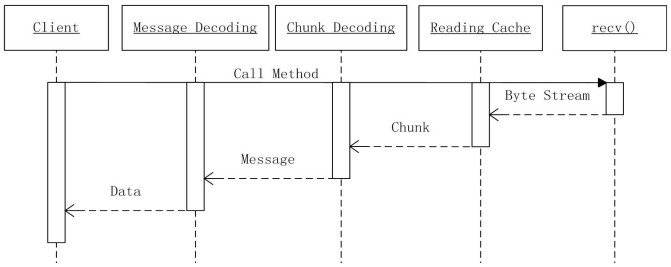

Figure 12. Receiving Message

D. *Experimental results*

Our software is based on Win32 Console Application. It accepts a RTMP URL as parameter to execute. When the software starts to run, it connects the input RTMP URL and prints out the progress of Message flow during the process of establishment of streams connection. At the same time, it saves the received media data as a FLV format file named "output.flv" in the local hard-disk.

To test our software, we used the software to connect the following RTMP-based Internet TV station shown in Table II. The result showed that the software successfully completes its function when dealing with all these RTMP URL. Figure 13 shows that the screenshot of software when it processing CETV-1 RTMP URL. As shown in the figure, it has prints out the progress of message flow, and successfully gets media data to write into FLV format file.

TABLE II. EXPERIMENTAL URL

Name	RTMP URL
CETV-1	rtmp://pub1.guoshi.com/live/newcetv1
CETV-3	rtmp://pub1.guoshi.com/live/newcetv3
CCTV-FINANCE	rtmp://live.cctvfinance.com/cctvcj/live1
Hongkong TV	rtmp://live.hkstv.hk.lxdns.com/live/hks
Beijing Mobile TV	rtmp://www.bj-mobiletv.com:8000/live/live1

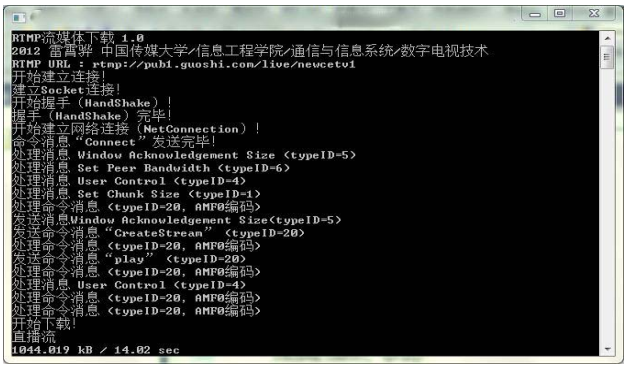

Figure 13. Software screenshot (CETV-1)

The FLV format file that we get from RTMP-based Internet TV station can play by ordinary Media Player. Figure 14 shows the screenshot of playback of the output FLV file that we get from CETV-1 RTMP URL.

Figure 14. Playback of Output FLV file (CETV-1)

VI. CONCLUSION

This paper introduces the basic structure of the RTMP: Message and Chunk. The whole process of playing a RTMP-based streaming media is described in detail. Moreover, the paper introduces the implement of software that can download RTMP-based streaming media. Method mentioned in the paper can be used as the basis for the development of more complete RTMP-based streaming media processing software.

ACKNOWLEDGMENT

This work was supported by the Research of Network Multimedia QoE Evaluation and Monitoring System Project in Communication University of China.

REFERENCES

[1] China Electronics Standardization Institute, CCBN2012 Proceedings. Beijing: China Standard Press, 2012, pp. 218–227.

[2] Wikipedia, Comparison of streaming media systems. http://en.wikipedia.org/wiki/Comparison_of_streaming_media_systems

[3] Dongcai Qiu, Delivery Measurement of Streaming Media Based on RTP. Chongqing: Chongqing University of Posts and Telecommunications, 2007.

[4] Ely Greenfield, "The internal mechanism of the Flash Player", Adobe Flash Platform Summit, 2010.

[5] Adobe Systems Inc., RTMP Message Formats. June 2009.

[6] Adobe Systems Inc., RTMP Commands Messages. June 2009.

[7] Adobe Systems Inc., AMF 3 Specification. 2006.

[8] Adobe Systems Inc., Real Time Messaging Chunk Stream Protocol. June 2009.

[9] Adobe Systems Inc., Adobe Flash Media Server 4.5 Developer's Guide. 2012, pp.130–131.