TELKOMNIKA, Vol.11, No.11, November 2013, pp. 6755~6761
e-ISSN: 2087-278X

Design of Network Middle-Ware Base on Flash Streaming Media

Jian Huang*, Xiaopei Liu, Hongmei Yan
Institute of Communication and Information Engineering, Xi'an University of Science and Technology, Xi'an 710054, China, 086-02985583167
*Corresponding author, e-mail: huangjian_work@163.com

Abstract

This paper analyzes the main information and streaming media service function of Real Time Messaging Protocol for Flash media remote play. It simplifies and achieves the Flash AV network live broadcast, network broadcasting and releasing function, which can support to operate in the cross-platform of Windows, Linux and embedded system with good property by the cross-platform testing. Meanwhile, it supports to connect with FMP seamlessly, being widely applied in the AV server of Windows, Linux platform or the network AV electronic products of embedded platform connecting with FMP seamlessly.

Keywords: *flash media player, realtime messaging protocol, embedded system, middleware*

1. Introduction

The present FMS (Flash Media Server) is applied widely, most of the network streaming media broadcast and live broadcast programs take FMP(Flash Media Player) as the client-side. FMP is a type of safe and reliable streaming media plug-in, supporting various of operating systems such as Linux, Windows and Mac, and these plug-ins [1] are being installed in over 1.3 billion PC machines globally. At present, the relative popular embedded electronic products hardly achieve the FMS function. Firstly, FMS seems too huge, with high demand for the software and hardware resource, the embedded system resource is hard to meet this requirement. Secondly, FMS should connect with FMP with lots of complicated control information.

To the embedded hardware platform with strict demand for the resource, for saving the hardware cost, most of embedded hardware platform procedure space and SRAM space are very small. According to the character of embedded system, this paper designs and installs the Live,VOD and releasing function of FMS to the middle-ware module to directly support the AV remote play of Adobe Flash Player.

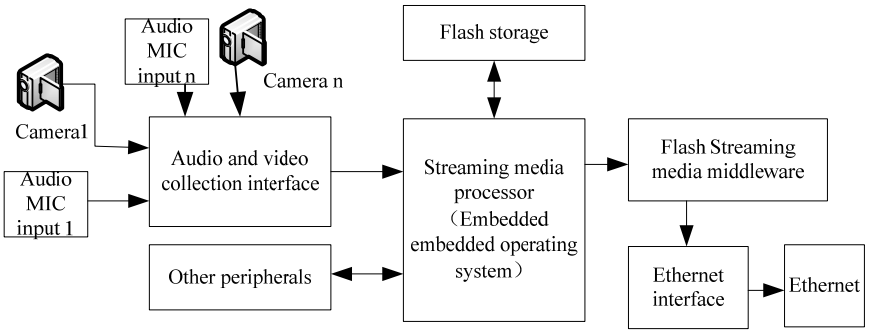

Figure 1. Working Principle Sketch Map for Flash Streaming Media Network Middle-ware

The working principle of this middle-ware is shown as Figure 1, the video from camera and audio from MIC are compressed into the digital media data flow through the hardware

Received April 19, 2013; Revised July 10, 2013; Accepted July 29, 2013

platform coding. The middle-ware block designed is adopted in this paper to do the Flash protocol packing. It is switched into the network streaming media format recognized by FMP, and delivered the packed digital media data to FMP by the network protocol interaction with FMP. FMP will decode and play the operation automatically.

2. Framework of Flash Network Middle-ware Software

The hardware coding platform shown in Figure 1, can be the S3C6410 processor of Samsung, the processor platform of TIDM8107 or MD385, also can be the platform of Intel X86 framework, as long as its coding compressing video format is H 264 or MPEG4, they can connect with FMP by this middle-ware. The developing platform of this paper is the PC machine base on X86 framework, and adopts the standard C language to develop. As it is shown in Figure 2, the whole function structure of the middle-ware includes seven blocks of media broadcast service, media live broadcast service, media releasing service, information core control module, FLV streaming format packing module, A/V buffer module and application programming interface etc.

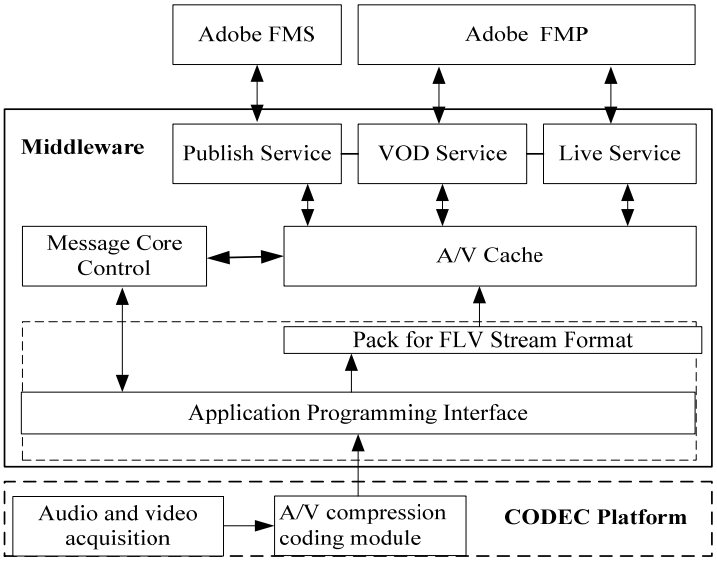

Figure 2. Function Structure Chart for Flash Network Middle-ware

FLV streaming format packing module mainly follows the RTMP protocol and AMF coding format. FMP and Middleware must carry out shaking hands before playing stream. The basic process can be seen from literature [1]. After the success of the shaking hands between the server and the client, the server will receive the connect message sent by the client, this message includes many parameters such as flashVer capabilities, swfUrl, tcUrl, pageUrl, audioCodecs, videoCodecs, objectEncoding and so on. For decoding h. 264 video, the flashVer, which is the current version number of the client, must be no less than 9.0.The objectEncoding indicates that coding format of the media stream data is AMF0 or AMF3. If the client connects successfully, the server must send NetConnection.connect.success message to response to the client. Otherwise, send NetConnection.connect.rejected message [2-3]. A/V buffer module mainly adopts the ring buffer technology, adopts the less RAM consumption, supports many circulating channel and provides various kinds of network streaming media service; in the reference [4], it describes the implement method of A/V buffer module in detail. In the following, this paper will analyze the protocol flow of media broadcast service, media live broadcast service and media releasing service three core modules in detail.

3. A/V Live Broadcast Service

A/V real time flow, requires the coding and decoding platform to collect camera and MIC (like PCM,16 data per sampling)according to the definite fps (such as PAL format, 25fps, with interval of 40 ms), then forms the A/V basic flow through video compressing (MPEG4 or H.264 coding) and audio compressing (AAC or MP3 coding), forms the A/V data of FLV packing[5] format through AMF coding, sends them into A/V delivery channel to wait the server for sending to FMP client-side. To the FLV packet of video data, it must be the first frame as the configuration frame, the second frame as the key frame, the later can be P or B frame. To the video of H.264 coding, the first frame can be the key frame, but it must be with AVC NALU information in the front, by carrying SPS(Sequence parameter set) and PPS(Picture parameter set) configuration data, after FMP is received, AVC NALU information can initialize the decoding machine correctly[6].

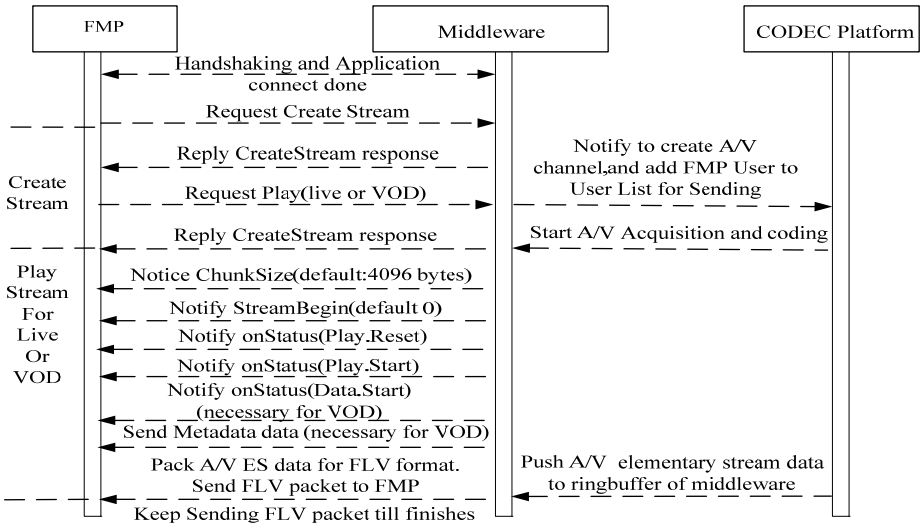

Figure 3. Service Flow for Middle-ware Broadcast and Live Broadcast

Its play flow is shown as Figure 3, FMP requests the coding platform for the live broadcast media data, the middle-ware will replace the coding platform to do the consultation of network play protocol with FMP, the middle-ware at this moment is equal to the function of FMS; after FMP connects with the middle-ware successfully and establishes the network application link successfully, FMP will request to establish the streaming media channel instantly, every media channel takes charge of the network sending task for one audio flow and one video flow. Every channel has its own user list. After FMP establishes the flow media channel with the middle-ware successfully, FMP will send the play control information to the middle-ware, and decide to allocate this user to the relative channel user list according to the inside StreamName parameter, if this channel is not existing, it will be back to the error information: NetStream.Play.StreamNotFound. If it is existing, it will be back to the information of NetStream.Play.Start and NetStream.Data.Star instantly, inform the client-side start to receive the data, and initiate the collecting and coding task of the relative A/V channel.

4. A/V VOD Broadcast Service

When the client-side needs to broadcast the streaming media files stored in Flv format remotely, sample the VOD application play mode. When FMP connects with the middle-ware successfully, after establishing the connection, first deliver the brand width checking information, judge how large of content can the brand width withstand, then inform the middle-ware how many seconds it has buffered(if 100 seconds, then according to the data of 40ms per frame,the middle-ware can fill the client-side buffer by sending 250 video continually), then send the play requirement, finally the middle-ware replies the media play start control information to FMP, and

send enough data to FMP for buffer play at one time, when FMP nearly finishes the play, it will restart to send. So that it can improve the network usage rate, and assure the client-side to play fluently.If for the mutation of the network quality, and that causes the deliver jam, then FMP finishes the buffer data without new data, it will re-send the notice to the middle-ware, and continue to request delivering the data. As Figure 4 shows, the broadcast control information is basically the same as the live broadcast control information flow, only when middle-ware informs FMP to send more NetStream.Data.Start and MetaData informs FMP to broadcast the file to load successfully, it will start to send the data; in MetaData, there are the basic parameters needed by the broadcast, such as video frame rate, height and width,coding format, audio sampling rate and bit wide etc.

5. A/V Real Time Publish Service

This middle-ware supports to release the real time streaming media to FMS server of Adobe at the real time. The FMS does the real time retransmit, shown as Figure 4, the coding platform starts to release one media flow to FMS by applying API connector of middle-ware, the middle-ware will do the information consultation with FMS automatically, such as setting brand width, streaming start location, establishing the streaming channel etc.. Once the releasing streaming channel is established successfully on FMS, the middle-ware will inform the coding platform to establish the real time A/V channel, and start the collecting and coding of the A/V, hereafter, as long as the coding platform delivers the A/V streaming data to the buffer area of middle-ware nonstop, the middle-ware will pack these streaming data into FLV steaming format automatically and send to FMS. Until the coding platform finishes releasing. To release the streaming service is equal to deliver the live broadcast service function to FMS, for FMS is the powerful streaming server, supporting large quantity of FMP client-end to visit simultaneously. This function is fairly proper for releasing the live broadcast news video or sharing the media demand.

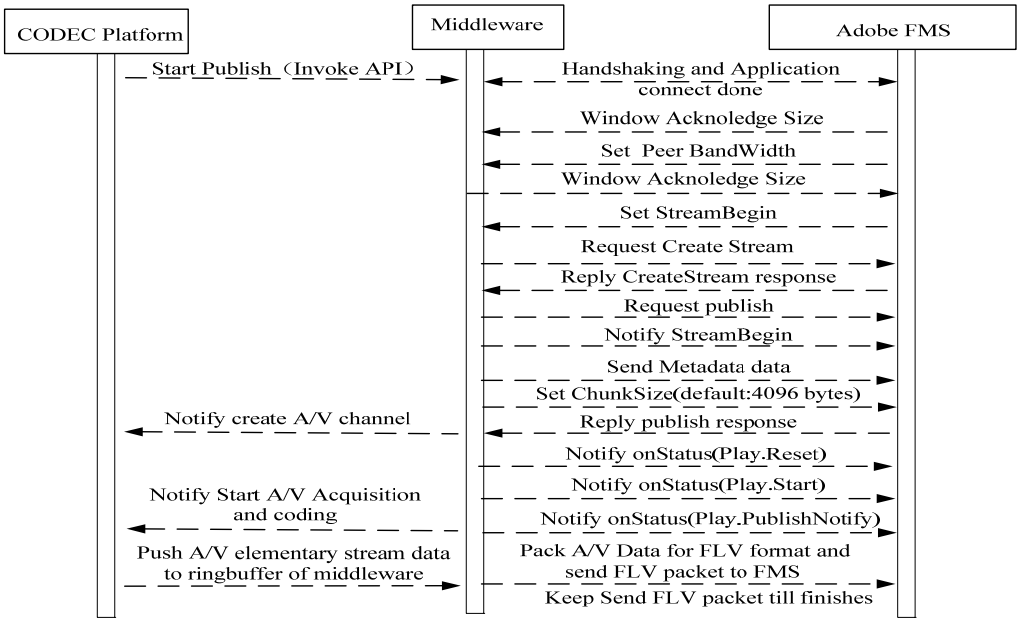

Figure 4. Sequence Diagram for Streaming Real Time Releasing Information

6. Flash Streaming Media Pack Format

The default package size of the network transmit is 128 bytes. After the client(Flash network media player) sends successfully connect message to FMS, FMS will decide the maximum chunk body size for the both sides and notify the client a chunksize message. When a package is greater than the chunksize, it must be split into several blocks whose body size is

less then the chunksize, the splitting process for a media Data whose length equal to 5000 bytes, shows in Figure 5:

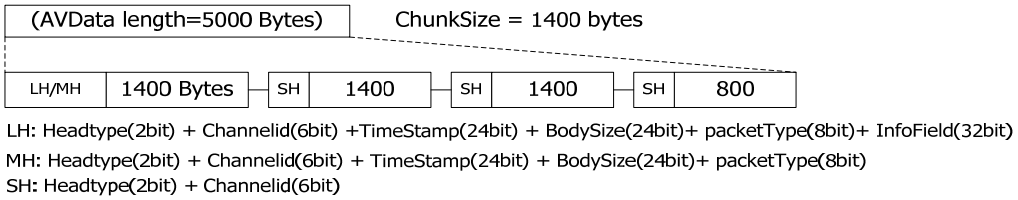

Figure 5. The Process of Splitting Audio or Video Data to Several Chunks

Assuming that the chunksize is 1400 bytes, the data can be split up into 4 blocks as shown in Figure 1. The first block carry complete information of the package head, such as package head size, package body size, timestamp, package type. For the first block of the first package, the LH head format who has 12 bytes for the size of the head ,will be adopted, and the stream identifier is stored in the InfoField parameter of the head, For the first block of follow-up packages, the MH (8 bytes) head format can be adopted, having not the InfoField parameter. The SH (1 bytes) format head is adopted by blocks ranging from the second block to the last block.

As it is shown in Figure 6, the network transmit of Flash streaming media can pack to transmit according to 4 types of format shown as the Figure 6, HeadSize character is with 2 bits, taking the value of 0,1,2,3 to represent the 4 types of packets for large, medium, small and mininum packet.

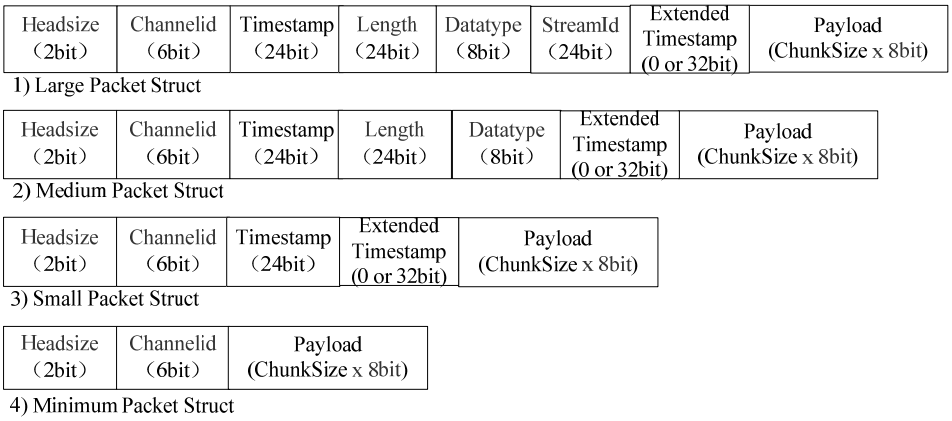

Figure 6. Network Data Format base on AMF Coding

ChannelId occupies 6 bits, being used to identify the delivered packet in this channel whether is order or media data. There are several typical values, such as 0x02 is usually used for ping order channel, being used to consult the network bandwidth and the window size. The value 0x03 is used for consulting the logical link and remote call order. The value 0x04 is used for consulting releasing streaming order(publish or invoke). Once the middle-ware the streaming media channel is established between the middle-ware and FMS or FMP and the middle-ware, its channel number usually starts from 8, does not exceed 64 channels at most. Once every logical channel is consulted to deliver the media data flow, it can support one audio flow and one video flow.

For the media data, the timestamp concerns the A/V synchronism,decoding and play. Usually when the value of the normal timestamp in the headers is less than 0xffffff, the

extensive timestamp occupies 0 byte (no use). When the normal timestamp in the headers is more than 0xffffff, the normal timestamp must be set to 0xffffff, and start the extensive timestamp.Its timestamp value is stored by using 32 bits of integer.

Length indicates the load length, this length is the full length of the whole A/V frame.

DataType indicates the packet type, for example, the typical value 0x08 represents this packet is audio frame, the value 0x09 represents the video frame, the 0x16 represent the compound frame of audio or video, the value 0x12 represents the metadata.

StreamID shows the number of transmitting the media streaming between FMP and middle-ware directly or the middle-ware and FMS. Support multi-streaming transmit. Every streaming supports one audio and one video streaming deliver.

There are 4 packet formats in Figure 6, all are related to the load data type, usually when delivering the order packet, we can adopt the packet of medium format. Once the streaming channel is established (for example StreamID is more than 0), after starting to deliver the media streaming, you can adopt the large packed or medium packet, for after you used the large packet for the first time, it will be considered to omit StreamID; the final load Payload field becomes long,considering the unstable network quality, the flash steaming media will consult ChunkSize between delivering between FMP and middle-ware or middle-ware and FMS, it is the load byte number that one packet carries most; if this value is 4096, then when one media data frame is more than 4096 bytes, it must be divided into many sub-packets. For example, one video frame is with 9000 bytes, we will use Large Packet format when sending for the first time, the packet header data is (0x08 0x000000 0x2328 0x09 0x01), the load part real size is 4096 bytes. We will use the Minimum Packet format when sending for the second time, the packet header data is (C8), the load part real size is 4096, we will also use Minimum Packet format when sending for the third time, the packet header data is (C8), the load part real size is 808 bytes. The streaming media load for three times sending does not exceed the size set by ChunkSize. When it does not reach ChunkSize, it will mark the unpack left data. The other person involved will compare and check the length data of the packet received at the first time with the data received in the former time automatically, if it is correct, do the next step processing, if it is incorrect, return back to the error instantly and break the streaming channel.

7. Conclusion

The middle-ware designed in this paper is tested by the cross-platform, and operates steadily in the platform of Windowx, linux_x86 and embedded linux, with good property. In the operation system (operating in the virtual machine of Vmware) of ubuntu 10.04 (linux x86) with 32bits, adopts gcc compiler, the compiling operation result is shown as Figure 7, the Linux virtual machine (Ip add. Is 192.168.1.196) operates the server testing procedure, and operates one simple FMP client-side in Windows operational system, the play address is rtmp://192.168.1.196/live/v1, that can achieve the real time network broadcast.

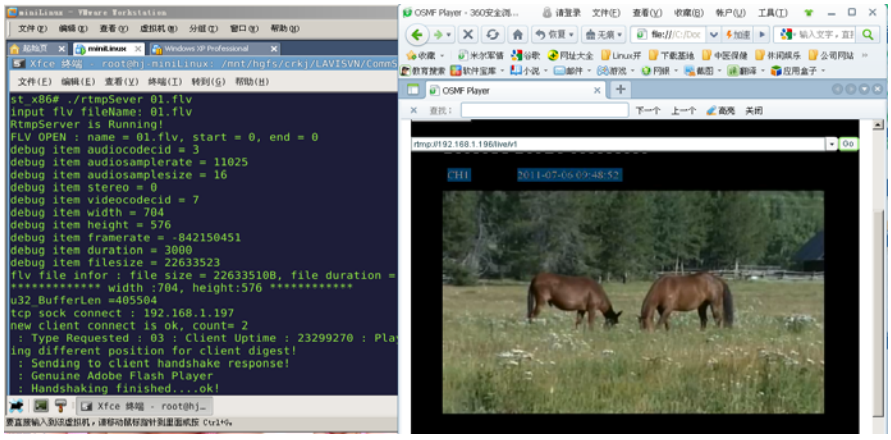

Figure 7. Operational Testing Result base on Linux x86 Operation System

On the operation system of Windows XP SP3, install FMP (version above 10), FMS version is 3.5, test the media releasing streaming of the middle-ware, send the video to FMS, when FMP visits the media stream retransmited by FMS, its operational result is completely the same as the test in the Linux system.

The testing result shows that this middle-ware can operate the coding by the cross-platform, with the definite generality and practical value, can be applied to the audio server in the platform of Windows,Linux or the network A/V electronic products in the embedded platform connecting with FMP seamlessly.

Acknowledgements

This work was supported inpart by the Science and Technology Plan of Shaanxi Province, China (Program No. 2011K09-46; 2012K06-49), the Science and Technology Plan Industrial Application Technology Research and Development Project of Xi'an (Program No.CXY1119; CX1258□; CX1258□), the Xi'an City Beilin District Application Technology Research and Development Project (Program No. GX1209), and the Scientific Research Program funded by the Shaanxi Provincial Education Department (Program No.12JK0508; 12JK0535).

References

[1] Adobe Systems Inc. RTMP Specification License. 2009.
[2] Adobe Systems Inc. Action Message Format -- AMF 0. 2006.
[3] Adobe Systems Inc. Action Message Format -- AMF 3. 2006.
[4] Huang Jian, Wu DongMei, Liu Xiaopei. *Implementation of the RTMP server Based on embedded SYSTEM.* American Journal of Engineering and Technology Research, Proceeding of the International Conference on Opto-Electonics Engineering and Information Science. 2011; 11(12): 1730-1733.
[5] Adobe Systems Inc.Adobe Flash Video File Format SpecificationVersion 10.1
[6] ISO/IEC 14496-10. Information technology-Coding of audio-visual objects-Part 10: Advanced video coding | ITU-T Rec. H.264, Advanced video coding for generic audiovisual services