Implementation of Mobile Streaming Media Player Based on BREW*

WANG Zhong-rong, LIU Zhao
School of Electronic Engineering, University of Electronic Science and Technology of China  Chengdu  610054  China

Abstract Nowadays mobile streaming service through cell phone is becoming the highlight of new value-added mobile services. Based on the present CDMA1x wireless data network and Binary Runtime Environment for Wireless (BREW) platform, adopting compression technologies of H.264 and QCP, a set of streaming media players are designed and implemented, and the principle, structure, key technologies and performance analysis of this system are introduced. This player works well in practice.

Key Words streaming media; Binary Runtime Environment for Wireless (BREW); Real-time Transport Protocol (RTP); Real-time Streaming Protocol (RTSP)

There are two major methods for audio/video files transfer through the network, downloading and streaming. The volume of storage and network bandwidth restricts transferring big audio/video files. Downloading needs minutes or hours while streaming only takes several seconds in starting and buffering. Streaming server will continuously send real-time packets to clients and the users don’t need to wait for the whole file to be downloaded. They can play and receive simultaneously without big cache. Streaming is certainly a better choice for the limited working space of cell phone and PDA, etc[1-2].

The expanded bandwidth for air interfaces has made a solid ground for streaming media application on wireless networks. With the advantages of wireless system in time and place, mobile streaming media service is very attractive. The China Unicom’s CDMA1x data network has an average steady rate of 70-80kbps and is capable of running streaming media business. This paper will introduce the implementation of a mobile streaming client player based on BREW platform.

1 Mobile Streaming Media System

Fig.1 shows the structure of the mobile streaming media system, including four basic parts: the server, the client, the transmission channel and the content source. The server adopts concurrent server of cluster; the client is a cell phone supporting BREW application; the transmission channel uses CDMA1x wireless network of China Unicom; and the content source are the streaming type documents or real-time audio/video data. In addition to providing the basic streaming service, the server deals with the additional function such as file management, access certification, and charging[3-4].

Fig.1  Mobile streaming media system structure diagram

2 Implementation of the Client

This client is based on the platform of Binary Runtime Environment for Wireless (BREW) which is a point-to-point solution for wireless data application released by QUALCOMM in 2001. It’s a basic platform for the development and operation of value-added services on CDMA wireless internet radio platform. BREW API provides a high-efficiency, low-cost, expansible Application-Executing Environment (AEE), customized to the development of hand-held device application. For developers, BREW AEE is an abstract layer above the operation system on embedded chips, with the characteristics of small volume, fast speed, extensiveness, stability and safety, and with abundant developing interfaces and strong support to multimedia applications such as audio,
BREW applications can be programmed in various computer languages, but the efficiency in C/C++ is of the highest. The BREW SDK is integrated in Visual C++ environment. After successful setup, there will appear a new menu item of BREW Application Wizard under New->Projects. It’s easy to establish a new project with this wizard. The unit of developing and deploying a BREW application is called Module, allowed with several Applets in one Module, but there is only one active Applet each time regardless of the number of Applets in a Module. BREW is a single-thread environment, and it is difficult to resolve the problems about synchrony and efficiency of multitask such as network telecommunication, data processing and audio/video displaying.

2.1 BREW Client Structure

Fig.2 illustrates the structure of the media player. It can be divided into four layers according to the function: conversation control layer (RTSP), data transmission layer (RTP), decoding layer and displaying layer.

The conversation control layer is responsible for sending/receiving and processing the RTSP control instructions, not transmitting data directly but by RTP/TCP; the transmission layer is responsible for the transmitting of audio/video data and other controlling information. The decoding layer decodes the compressed data on time according to the timestamp of RTP packet header, and then deposits the uncompressed data into waiting queue for displaying.

In this system, a software decoder decodes video data while audio data is decoded by hardware. The last layer is responsible for synchronous processing between audio/video displaying and sends the decompressed data into the corresponding displaying equipments.

2.2 Key Technologies

2.2.1 Performing Multitask in a Single-thread Environment

The environment of BREW is a single thread environment, multitask depends on the message loop mechanism and the asynchronous pattern realized by callback function. Divided into small segments, a big task would be executed with one segment each time. For example, decoding one frame image or receiving one packet, then generating an event for the following task and sending it into the system message queue, returning back the power of control to the system so that system can execute the next task. Thus it makes the system resource can be shared simultaneously by more than one task. In macro view, the system performs multitask at one time.

Message loop of BREW is driven by AEE_Dispatch: it checks the event queue of BREW, distributes the events to an upper function named APP_HandleEvent one by one, and also checks the queue of callback function and executes them one by one. After all events and callback function are processed well, a round of message loop is ended. Then the system will repeat the above task until the application is terminated.

Callback function is the key of asynchronous operations. When running a time-wasting operation, the system can’t return the result immediately (such as Socket I/O and playing audio). In order to avoid the system being waiting, asynchronous operation should be adopted. It means that the operation can return to execute the following instructions immediately under the condition of registering a notification function that would be called back by the system when the result is returned.

2.2.2 RTP Streaming Transmission

BREW provides a set of standard interfaces of SOCKET, supporting both UDP and TCP. Generally, it ought to choose UDP because of its high efficiency for the real-time demand of streaming data, but currently the error rate of wireless network is still so high, and
the memory applied by BREW applications is too small to realize resending of the lost packets\textsuperscript{[5-6]}. So this system selected the RTP/TCP protocol to guarantee the reliability of transmission.

There are some fields defined in the 12-byte-header of RTP for the assurance of continuum and synchronization of streaming data. The sequence number guarantees the RTP packets to arrive continuously and the timestamp helps to get synchronous audio/video stream. There is a great differentiation between an audio and a video packet: A video frame could be divided into several pieces and be encapsulated separately in a few RTP packets with continuous sequence numbers and the same timestamp; but each audio RTP packet includes sampling data of the same interval in time and continuous audio packets have different timestamps. After decoded, several packets would concatenate to become one frame that is called a play unit.

2.2.3 RTSP Session Control

RTSP provides to the media stream control function similar to Video Cassette Recorders (VCR), such as displaying, pause, fast-forward, etc. It means that the client can control the streaming server through remote network by RTSP session\textsuperscript{[7]}.

Firstly, the client sends to the server a DESCRIBE command that includes the URL address of a media document, the server will return the client the Session Describe Protocol (SDP) of the coding information of the requested media document. Secondly, the client initializes the media player according to the SDP, sending a SETUP command to the server to connect. At last, the server creates a new session for that client and begins a streaming service. In the course of the service, the client sends command like PLAY, PAUSE, TEARDOWN, etc. to carry on VCR control of the server.

2.2.4 Audio and Video Decoding

The selecting of audio and video coding methods in this system incarnates a compromise of various demands that include: compression and decompression velocity, variable code rate, robust under the dynamic transmission environment, signal to noise ratio, flexibility, etc.

Considering the network bandwidth and the terminal decoding capability, this system selects H.264, QCIF(176×144) image format as the video standard, and QCP as the audio standard. The compression performance of H.264 is as 200\% as that of widely used MPEG-4 SP, and its strong error correction is suitable in the wireless environment. But its algorithm complication is also increased greatly and requests the terminal having higher computing capability\textsuperscript{[8-11]}.

QCP is a kind of low-bit-rate audio coding released by Qualcomm, and it has been specially optimized to aim at the human voice. It has no background noise and still keeps very good quality under the condition of 8 kbps bit rate\textsuperscript{[2-3]}. What’s the most important is that CDMA mobile phone supports decoding by hardware, and provides interfaces to the BREW applications. Adopting QCP, the system consumes only a little time in dealing with audio decoding/playing and raises the whole performance of the system.

When video is decoded, the system takes out all the packets with the same timestamps from the queue to form a frame and sends it into the video decoder; When audio is decoded, the system synthesizes one frame each time according to the frame length, then calls relational functions of ISoundPlayer interface, and decompress and play the audio data by hardware directly.

After being decoded, the video frames will not be displayed right away, so it is necessary to design a circular queue as a buffer for displaying. Each unit will be assigned a fixed memory space in the queue; the decompressed frame will be inserted into the queue. After being displayed, the unit will be set free but does not need to release the memory so that the following frame can reused it, thus save much time for memory operation.

2.2.5 Synchronization between Audio and Video

To keep synchronization, there must be certain mechanism to guarantee that the audio/video frame has the same time to play. In this system, each audio frame is of equal interval, with each lasting 1000 ms. So we don’t have to set a special timer for it, but just play audio continuously according to its frame rate. It is necessary to introduce an important structure named AEESoundPlayerInfo.

```c
typedef struct
{
//set input type (SDT_BUFFER)
AEESoundPlayerInput eInput;
//set the buffer pointer or file name
```
void *pData;
    //set the length of the audio frame
    uint32 dwSize;
} AEESoundPlayerInfo;

BREW application processes events dispatched by low layer of the system in the function of APP_HanleEvent. There is an event EVT_NEXTSOUND defined to handle audio playing. The steps are as follows: first, set the variable of the structure mentioned above; then register a callback function (Player_SoundNotify) which will be inserted into the callback function queue and waits to be called in the next message loop after its processing is completed; finally, call the interface function ISOUNDPLAYER_Play to execute audio playing.

In this registered callback function, there is a definition of following processing: set audio/video synchronously at first, then send the EVT_NEXTSOUND event again to notify the system to play the next audio frame, which is so quick that human hardly feel the interrupt of sound.

```c
void AEE_HandleEvent(...) {
    ...
    case EVT_NEXTSOUND:
    // AEESoundPlayerInfo playinfo;
    ISOUNDPLAYER_SetInfo(m_pPlayer,&playinfo);
    ISOUNDPLAYER_RegisterNotify(m_pPlayer,
        Player_SoundNotify,...);
    // ISoundPlayer *m_pPlayer;
    ISOUNDPLAYER_Play(m_pPlayer);
    ...
    }
```

A timer is set for video displaying according to the video frame rate and the timestamp and timer determine the displaying time of a certain video frame. A CDMA mobile can obtain accurate system time from the base station, making the BREW application control accurately the displaying time of video frames. If the displaying time of current video frame is larger than the system time, it means that this frame should go on waiting; on the contrary, if the displaying time of current video frame is smaller than the system time, it means the current frame is already late, and the system must jump across all the late frames, and display synchronous frame directly.

The synchronous strategy of this system is to guarantee audio’s fluency first, and allow throwing some video frames delayed. So there will be phenomenon of skipping of video pictures sometimes.

2.2.6 Suspense and Recovery

There is an important difference of the running environment between BREW and Personal Computer (PC) ----a BREW application would be interrupted at any time by answering a call, or receiving a short message or even ringing the alarm clock. There needs a mechanism to deal with the suspending and recovery of the application. When any interrupt occurs, the application receives an event of EVT_APP_SUSPEND from low layer of the system and does some corresponding work, such as releasing the network connecting, revoking the callback function and canceling timing, etc. While recovering, the application receives an event of EVT_APP_RESUME that notifies it to resume again, and all resources released just now should be recovered immediately.

2.3 Performance Analysis

The coding scheme of the system is H.264 +QCP, requesting the client equipment equipped with ARM9 series chip. The video buffer should guarantee to preserve at least 10 frames and the audio buffer at least the data for 20 s. Tab.1 is about the testing result of Samsung W109 (MSM6300 chip), in Fig.3, there are two photos of BREW Player interface. In this system the audio is quite fluent and the video rate can reach 10-15 fps, enough to satisfy the demand of the screen with less violent motion[1,4,12].

<table>
<thead>
<tr>
<th>Video format</th>
<th>Frame rate/fps</th>
<th>Coding rate/kbps</th>
<th>Throw frames/fps</th>
</tr>
</thead>
<tbody>
<tr>
<td>H.264 QCIF</td>
<td>10</td>
<td>40.56</td>
<td>0.5</td>
</tr>
<tr>
<td>H.264 QCIF</td>
<td>15</td>
<td>62.44</td>
<td>0.9</td>
</tr>
<tr>
<td>MPEG-4 SP QCIF</td>
<td>10</td>
<td>78.28</td>
<td>1.2</td>
</tr>
</tbody>
</table>

At present this system has been put into practice
by certain mobile service provider and it is suitable for news programs in which human voice is of the main in audio and video is mainly about the head and shoulder of an announcer. QCP code is suitable for human’s voice and head-shoulder video helps to reduce video coding rate greatly, thus we can get a good performance in transmission and displaying. If we want realize VOD service for movies, higher quality audio coding scheme should be used, for example MP3 and AAC, and the video frame rate must be raised to 25 fps, which can’t be supported by current wireless network.

3 Conclusions

The media streaming system realized in this project is based on the current CDMA1x wireless data network; telecom operator can provide fluent streaming service for customers and need not to rebuild any equipment or lines. At the same time, the system also has the potential application in secure and defense industry. A set of live streaming media system can monitor the object very well by way of a mobile phone and current mobile network, no matter where the subscriber is.

References


Brief Introduction to Authors

WANG Zhong-rong (王忠荣) was born in 1975. He got his bachelor and master degrees from University of Electronic Science and Technology of China (UESTC), both in electronic engineering. He is now a lecturer with UESTC. His research interests lie in digital image/video transmitting and processing, pattern recognition.

LIU Zhao (刘钊) was born in 1943. He graduated from Chengdu Institute of Radio Engineering in 1966. He is currently a professor and doctoral tutor of UESTC and his interesting fields includes digital image/video transmitting and processing, pattern recognition.