The Architecture Design of Streaming Media Applications for Android OS

Xueliang Zhao
Research institute electronic science and technology
University of Electronic Science and Technology of China
Chengdu, Sichuan Province, P.R China
zhaoxueliang1986@163.com

Dan Tian
Research institute electronic science and technology
University of Electronic Science and Technology of China
Chengdu, Sichuan Province, P. R China
tiandan@188.com

Abstract—**In view of the imperfection of the Android operating system multimedia function and the complexity of streaming media system. At the same time, the development cycle is long, and efficiency is low. In order to solve those problems, the thesis designed the streaming engine layer between Linux kernel layer and application framework layer of Android platform, and constructed fast and convenient streaming media application development framework. First of all, Android operating system (OS) was analyzed, and then the media engine was added into the Android OS architecture. Based on the characteristic of the streaming media system, the steaming media engine is divided into five layers, including the user interface, data capture, and data output, codec, and network transmission layer. At last, the architecture was applied in the wisdom medical terminal project. Through an instance of video data transmission client, the feasibility of the architecture is confirmed.**

Keywords-Android; software architecture; streaming media engine; wisdom medical

I. INTRODUCTION

With the continuous development of semiconductor technology, the processing ability of mobile terminal continuously was enhanced, and mobile Internet bandwidth was expanded. Mobile terminal streaming media applications has become the main way of traffic consumption. Although many mobile terminal OSs, Android OS has been popular because of its open source. In the second half of 2011, its global share is top-ranked, and it has become one of the most popular mobile OSs [1-3].

Streaming media application for Android has become a hot spot, but the appearance of Android is late, streaming media program development is not mature. At the same time, due to the complexity of the streaming media system, so the multimedia part of Android is not easy to understand, and its expansion is very difficult, obstacling to the development of streaming media applications.

In view of the questions in Android multimedia architecture, the thesis designed a kind of steaming media engine which is located between the Linux kernel layer and application program layer of Android, the user-friendly APIs were provided for the development of streaming media applications. It was an open development platform designed for streaming media applications for Android. At last the wisdom medical project was used to verify the feasibility and show its usage.

978-1-4673-2008-5/12/$31.00 ©2012 IEEE

II. ARCHITECTURE OF ANDROID OS

Android OS is a kind of open source OS, which is launched by Google. In Android, the hierarchical structure is used just like other OSs; it is divided into five layers, including Linux kernel, hardware abstraction, libraries and Android runtime, application framework and applications. The system architecture is shown in Figure 1. Every layer description is as follows [4-5]:

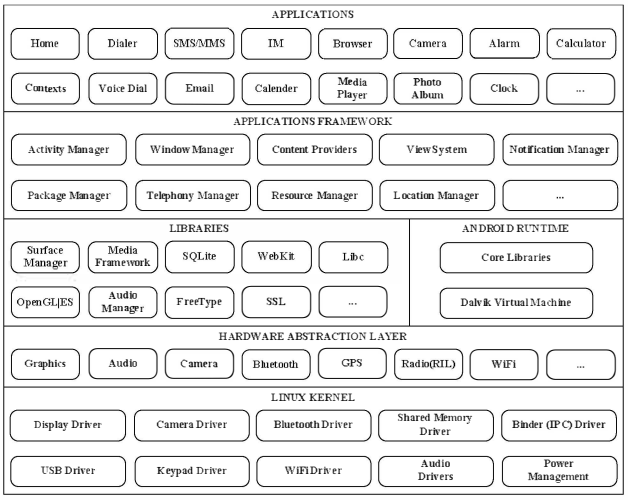

Figure 1. The software architecture of Android OS

A. Linux Kernel

Android OS bases on Linux 2.6 kernel, and Linux kernel as hardware abstraction of underlying layer provides the services of basic memory management, thread management, network protocol stack and device driver etc.

B. Hareware Abstraction

The hardware abstraction layer is used to abstract services provided by the underlying Linux kernel layer; it can shield the underlying implement details.

C. Libraries and Android Runtime

Android libraries provide many basic C/C++ libraries, which are called by applications. They are the ultimate

realization of the upper layer applicaitons. Android Runtime is a set of libraries which provides most of functions of Java programming language. Dalvik virtual machine is included. Every processing has own instance of Dalvik virtual machine. It is used to run the .dex excutive file.

D. Application Framework

The application framework layer provides a development platform for developers, which facilitates the reuse and replacement of components and the building of all kinds of applications.

E. Applications

Android application layer provides a range of packages, which contains most of Google's applications, for example, clock, calendar, etc. They are writen by Java.

III. THE DESIGN OF ARCHITECTURE

Generally, native code is faster than Java code [6-7]. So in view of of the efficiency requirements of streaming media applications, and the characteristics of Android stratification, the streaming media engine is located between Linux kernel layer and applications layer, and realized by C/C++ programming language. In the streaming media engine, the function of Linux kernel and libraries are called to realize streaming media codec and transmission functions. Android application layer could call the service provided by streaming media engine using JNI interface. Because the reliability of software is inversely proportional to the number of software stratification, so the whole structure is designed for four layers [8-9].The architecture of streaming media applications is shown in Figure 2.

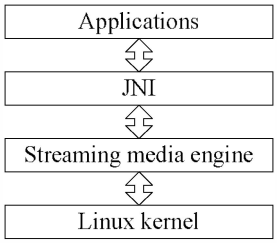

Figure 2. Android Streaming Media Application Development Framework

A. Overall design of streaming media engine

The characteristicsof Android platform is hierarchical design, while the modular design is typically used for streaming media system.Combining hierarchical and modular design, the streaming media engine is divided five layers, including the user interface, data capture, data output, codec, and network transport. The overall framework is shown in Figure 3. This of design strategy the can simplify media information proccessing. It is helpful for developers to develop and maintain streaming media application using the streaming media engine. It is also easy for sreaming media application to extend new codec or transport protocol.

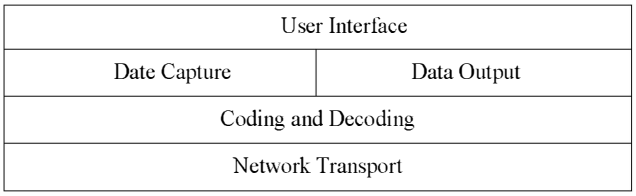

Figure 3. The Streaming Media Engine Framework

B. Data Capture

The main task of the data capture is to capture audio or video data. It is defined a derived class, which provides some interfaces including message input and output interface, device open and close interface. Developers only need to inherit the class, and then according to the data type which we want to capture, the functions of the class are realized. In fact, the device open function calls the appropriate driver interface to initialize and open an audio or video device. At the end of open () function, the thread is started to capture the audio or video data. At last, input function can be used to input data acquisition command, and then captured data by thread will be copied to message data area. When the output () function is called, we can get the capured messag included the audio or vedio data.

C. Data Output

The design of data output layer is similar to the design of the data capture layer. It also creates a derived class named by data output, and provides device open and close interface, data output interface. Because nothing can be got form the layer, there not is ouput () function in the layer. First of all, the developers inherit the class, and then realize it according to different output devices and business needs. When needs to be used, firstly it instantiated. Sencondly, the open () function which is realized by using device driver is called to open and initialize the device, and create the output thread for audio or vedio. In the thread, the input data is outputed the ouput device for playing audio or video. The ouput data is input by using input () function.

D. Codec

Codec layer mainly realizes the compression and decompression of streaming media data, it is defined a codec class, which has initialization, codec selecting, data input and output interface. After inheriting the class, developers could call the select () function to select different codec algorithm, and avcodecinit () to initialize the codec. The algorithm is realized in the input () function by different way. The output function is used to get data which has processed. The decoding processing is the contrary.

E. Network Transport

Network transport layer mainly to complete the transfer function in streaming media systems; it created a network transmission class, including data input and the session initialization interface. According to the requirement, developers inherit the class and initialize the transmition

session in init () function, then using input () function, developers can transmit the data to destination. The multi-thread mechanism can be used to achieve the transmission of data. The signal is used to communicate process with thread.

F. User Interface

The user interface layer is the interface of streaming media engine, and it is a JNI interface package of streaming media engine. Through JNI interface, Java applications can call the corresponding streaming engine functions. APIs implement the parameters control, parameter setting and query interface. The layer controls the flow of message from one layer to another. There is a mapping table between native functions and Android Java functions. It will be registered to Dalvik virtual machine when systemloadLibrary () is called. By this way, the time consumption will be reduced [10].

The user interface layer contains start and stop interface of streaming media engine. When the start function is called, the messages are sent from data capture via codec to network transport or from network transport via codec to data output.which one is selected can be configured by configuration file.

These classes in the streaming media engine each layer are compiled into dynamic libraries using Android NDK tools, which can be called by others. The purpose of this is easy to upgrade and reuse each layer. Only through changing the dynamic library, the developers can develop new application.

The relationship of those classes is shown in Figure 4.

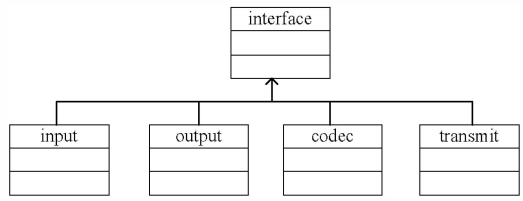

Figure 4. the relationship of the classes

IV. THE EXAMPLE OF ARCHITECTURE APPLICATION

The wisdom medical devices is a kind of terminal device which is used to help the doctor to save the patient in the ambulance before they arrive in hospital, it transmits the data from multi-channel physiological monitor as well as image or audio data to the control center for the preservation and remote assistance. Using the streaming media application development framework designed by this paper, we can quickly realize the video data capture and transmission function in the wisdom medical terminal device.

A. Development Methodology

In the design of wisdom medical terminal, we use H.263 compression algorithm for codec, and RTP protocol to transmit video streams. Development steps of the application are described as follows:

1) To inherit the class of every layer, use libav library to realize codec class, and use jrtplib library to realize RTP transmission protocol, use Android camera class to realize vedio acquisition, use Android AudioRecord class to realize audio acquisition. The user interface layer is realized by calling classes which are realized by other layers;

2) To compile these implemented codes into dynamic libraries by using Android NDK tools, generate streaming media engine library named libstreamingengine_jni.so;

3) In Android applications, the streaming engine class is created by using native function statement. It calls the interface of streaming media engine. To obtain data from the user configuration interface to configure streaming media engine, and then call the start () function to start the processing of aquisition and transmission. At this time, we have realized the video capture and transmission function.

B. Software Implementation

The software is divided into C/C++ layer and Java layer. In the C/C++ layer, it is the streaming media engine, which is deigned for reuse. In the Java layer, network transmission setting interface shown in Figure 5.

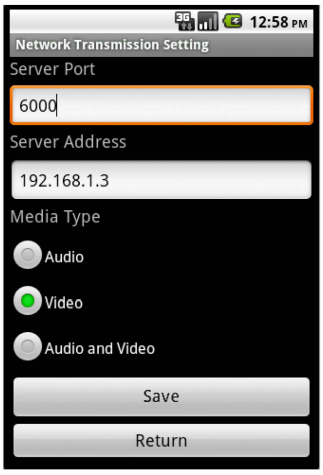

Figure 5. The network transmission seting interface

The remote control center server receives video display shown in Figure 6, 7. The Figure 5 is a static screen display; Figure 6 is a dynamic screen display.

Figure 6. The static screen display of server

Figure 7. The dynamic screen display of server

By experiments, the architecture designed by the thesis can quickly realize streaming media application development. It is very useful for Android developers.

V. CONCLUSION

The thesis proposed a kind of software architecture for streaming media application development in Android. The architecture bases on streaming media engine and provides developers with a convenient and efficient development framework in order to compensate for the deficiencies of the Android platform streaming media, and to reduce the streaming media application development cycle and complexity and improve software maintainability. Finally, the wisdom medical devices project has verified the feasibility of the architecture proposed in the thesis. In the future, the architecture will be used in more specific applications to enhance its stability.

ACKNOWLEDGMENT

We wish to thank Dean Zhong Tian of Reserch Institute Electronic Science and Technology of UESTC for his encouragement and suport on this thesis. Besides, we also need to appreciate teachers of Embeded System Lab, especially my tutor dan tian. At last, we need to appreciate seniors at laboratory for their care and modification for the thesis.

REFERENCES

[1] Li Zichen,"Streaming Media: Status, Prospect and Obsticle," information science, vol. 20, no. 9, pp.1005-1008, September 2002.

[2] Song Bi-lian, Wu Hua-ping, Chen Jia-xun and Sun Li, "Research on Media Stream and Comparison between Its System Design Schemes," Application Research of Computers, vol. 21, no. 1, pp. 204-207, 2004.

[3] http://baike.baidu.com/view/1241829.htm.

[4] "DevGuide|AndroidDevelopers," http://developer.android.com/guide/basics/what-is-android.html.

[5] Shanker, A. Lal, S, "Android porting concepts," 2011 3rd International Conference on Electronics Computer Technology, vol. 5, pp. 129-133, April 2011.

[6] Cheng-Min Lin, Jyh-Horng Lin, Chyi-Ren Dow, Chang-Ming Wen, "Benchmark Dalvik and Native Code for Android System," Innovations in Bio-inspired Computing and Applications, 2011 Second International Conference on, pp. 320-323, December 2011.

[7] Sangchul Lee, Jae Wook Jeon, "Evaluating Performance of Android Platform Using Native C for Embedded Systems," 2010 International Conference on Control, Automation and Systems, pp. 1160-1163, October 2010.

[8] Huang Jin-guo, Luo Zhen, "Research on the Architecture of Mobile Application Development," Computer engineering and science, vol. 32, no. 11, pp. 141-144, November 2010.

[9] Mo Dong-song, Peng Xiao-dong and Xia Ke-jian, "Three Architecture Patterns for Multi-tiers Object-Oriented Applications," Application Research of Computers, vol. 20, no. 2, pp. 34-38, 2003.

[10] He Dan-dan, Shi Zhan, "Design and Realization of VoIP Based on Android OS," Modern Electronics Technique, vol. 34, no. 6, pp. 28-31, March 2011.