

一种基于 Android 平台的移动终端多媒体播放器的扩展设计

李向军¹, 张小菲², 王书阵²

(1 西安文理学院 数学与计算机工程学院, 陕西 西安 710065; 2 西安电子科技大学 计算机学院, 陕西 西安 710071)

摘 要: 随着音视频技术研究的不断深入, 其应用也越来越广泛, 涉及到移动终端多媒体播放器的开发、视频会议、数字电视、视频点播软件等. 首先深入研究了音频系统和视频系统的体系结构, 并对它们的各个层次进行了深入的分析. 基于 Android 开发平台, 通过移植 FFmpeg 框架对音、视频系统进行了扩展, 使其能支持多种媒体格式的解析和解码. 最后, 给出了使用扩展后的音视频系统开发多媒体播放器的设计方案.

关键词: Android; Java 框架; 多媒体播放器; FFmpeg

中图分类号: TP391

文献标识码: A

文章编号: 1000-7180(2014)02-0118-04

An Extension Design of Multi-media Player System for Terminal Mobile Based on Android

LI Xiang-jun¹, ZHANG Xiao-fei², WANG Shu-zhen²

(1 School of Mathematics and Computer Engineering, Xi'an University of Arts and Science, Xi'an 710065, China;

2 School of Computer, Xidian University, Xi'an 710071, China)

Abstract: The wide application of audio-video technology is becoming more and more popular, involving the development of multi-media players for terminal mobile, video conference, digital television, video on demand software, etc. This paper first studies the structures of audio and video systems and analyzes each layers of them; then based on the developing platform Android, this paper expands the audio-video system by transplanting the FFmpeg framework, enabling it to support the parsing and decoding of a variety of media formats (audio and video). At last, this paper proposes the multi-media player development scheme by adopting the expanded audio-video system.

Key words: Android; Java Framework; Multi-media Player; FFmpeg

1 引言

随着 3G 时代的到来, 移动终端设备的高速智能化发展, 各类智能手机操作系统应运而生. 2007 年, 开放源代码的智能手机平台 Android 系统一经发布, 就受到了业界前所未有的热捧, 开发人员痴迷于 Android 技术的同时, 手机设备厂商也在不断推出 Android 系统的手机^[1], Android 技术被广泛应用于各种智能移动终端和平板电脑等上. 目前, 市场上主要流行以下几类手机操作系统: 谷歌的 Android、苹果的 IOS、Windows Mobile、Symbian 以及

BlackBerry^[2]. 其中, Android 采用了开源的 Linux 操作系统, 底层采用 C 语言实现, 应用层采用简单而强大的 Java 语言. 占有开放源代码优势的 Android 系统具有广阔的发展前景, 用户可以根据自己的需求将 Android 系统中的所有功能重新编写.

手机功能需求的不断增加促进了相关应用技术的发展, 其中音视频技术的应用最为广泛. 网络上流传的媒体格式日趋多样化, 要求媒体播放器能够支持这些媒体格式^[3]. 但是, Android 源码中音视频系统能够解码的媒体格式种类和效率越来越不能满足人们日益增长的需求, 对 Android 平台上音视频系

收稿日期: 2013-04-23; 修回日期: 2013-06-16

基金项目: 西安市科技计划创新基金项目“文理专项”(CX12189WL17)

统的研究和扩展有助于设计性能优良和多样化的媒体播放器。

2 音频系统

音频系统主要分为如下几个层次:给音频系统的应用层提供调用接口的应用程序框架层、给音频系统提供本地部分接口的 libmedia 本地库层、给音频系统提供服务的音频服务层、提供底层支持的硬件抽象层。

音频系统的各个层次接口主要提供了播放声音和录制声音两个方面的功能。如播放音乐时,音乐应用程序会在 Java 端发起调用,Java 框架中的 MediaPlayer、AudioManager、AudioSystem 等给应用程序的调用提供接口;经过音频文件的格式解析、音频流的解码,由 libmedia 库中的 AudioTrack 传递音频原始数据给音频服务 AudioFlinger;AudioFlinger 把所有的音频数据进行 Mixer,然后将音频数据通过硬件抽象层和驱动送入到音频硬件设备。音频系统结构图如图 1 所示。

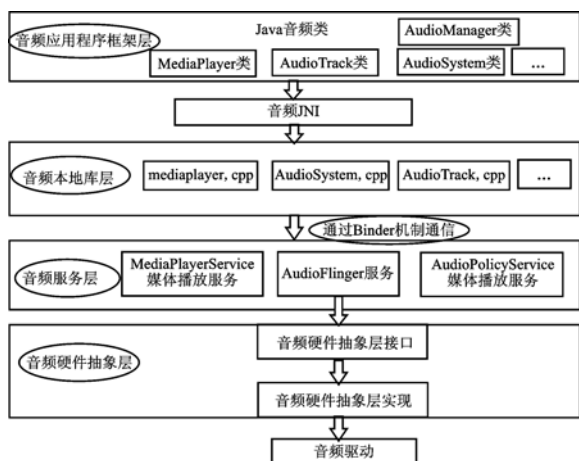


图 1 音频系统结构图

2.1 音频系统的应用程序框架层

音频系统的应用程序框架层是音频系统的第一层,提供了实现音频相关功能的 Java 类供应用程序开发者调用。音频应用程序框架层提供的关键类包括:音频播放类 MediaPlayer,音频录制类 MediaRecorder,音频输入类 AudioRecord,音频输出类 AudioTrack,音频策略类 AudioManager,音频服务类 AudioService,音频管理类 AudioSystem,音频文件的扫描类 MediaScanner,音频资源管理类 SoundPool,为铃声及 DTMF 音的播放提供支持的类 Ringtone、ToneGenerator 等^[4]。

2.2 音频系统的本地库层

音频系统本地库层是音频系统的第二层,通过

JNI 为音频系统应用程序框架提供接口。它是多媒体系统本地库 libmedia 中的一部分,libmedia 提供了音频、视频、照相机等多媒体部分的框架,主要实现了媒体播放器和媒体记录器的框架,向上通过 JNI 提供接口,libmedia 通过 AudioRecord 和 AudioTrack 提供音频的输入、输出功能,同时还提供媒体播放器、媒体元信息管理和媒体扫描等功能。

2.3 音频系统的服务层

音频系统的服务层是音频系统的第三层,通过 Binder 机制给本地库 libmedia 提供服务,包括 AudioFlinger、MediaPlayerService 及 AudioPolicyService 三种服务。AudioFlinger 主要负责管理音频数据处理及硬件抽象层相关的工作,向下访问硬件抽象层,实现音频数据的输入和输出,向上通过 AudioFlinger 提供音频服务。

AudioPolicyService 提供根据不同的音频策略设置不同的音频输入输出设备、设置和获取输入输出设备的连接状态、设置电话状态和响铃模式、设置和获取音量及音频参数等服务。AudioPolicyService 向外提供服务的很多方法都是通过音频策略管理类 AudioPolicyManagerBase 来实现的。音频策略管理类实现了 AudioPolicyService 中的大部分管理工作,包括音量管理、音频策略管理、输入输出设备管理等。媒体播放服务 MediaPlayerService 提供音频、视频、照相机等多媒体系统的服务,主要用于媒体播放的控制及根据音视频文件或 URL 传入的媒体源建立不同的播放器。

2.4 音频系统的硬件抽象层

音频系统硬件抽象层是音频系统的第四层,是音频服务 AudioFlinger 和音频硬件设备的接口。音频设备种类很多,Android 不可能为每种设备都提供支持,Android 提供了硬件抽象层来适配底层的设备。音频硬件抽象层可以有不同的实现方式,包括 AudioHardwareGeneric、AudioHardwareStub 及 AudioDumpInterface 等几个不同的音频硬件抽象层。AudioHardwareGeneric 是 Android 通用的,并且真正能够使用的音频硬件抽象层。AudioHardwareStub 用于模拟器,它不操作实际的硬件,不会有声音输出。AudioDumpInterface 是为了调试使用的类,只有音频输出功能,没有音频输入功能。也可以在开发时,自己实现音频硬件抽象层来发挥设备的性能优势,满足设备对更高质量音频的要求。

3 视频系统

视频系统应用程序框架层提供了 VideoView、

MediaPlayer、Surface 等 Java 类,供 Android 应用程序调用来实现视频的显示^[4]. MediaPlayer 通过视频系统本地框架层提供的 mediaplayer.cpp、MediaPlayerService 实现视频文件和视频流播放的控制,在媒体播放服务中建立 PVPlayer 播放器,经过 OpenCore 中视频文件的格式解析和视频流的解码,传递视频数据到 SurfaceFlinger. 视频系统本地框架层提供了 SurfaceFlinger 和 Overlay 两种视频数据输出的方式. 最后通过驱动层输出视频数据. 视频系统结构如图 2 所示.

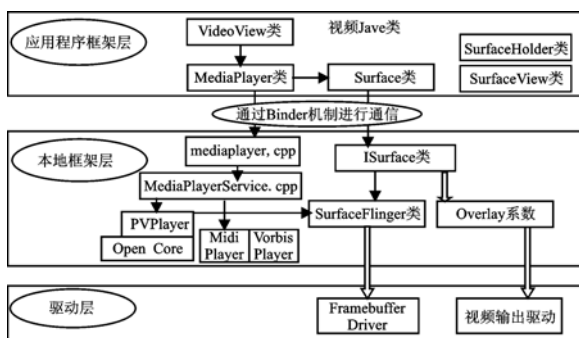


图 2 视频系统结构

SurfaceFlinger 是视频系统的一个重要部分,它能够将各种应用程序的 2D 及 3D 图像界面进行组合. 每个应用程序可能对应着一个或者多个图像界面 Surface,各个 Surface 之间可能有重叠, SurfaceFlinger 能够解决这种图形重合问题. 它会把各个 Surface 组合成一个主 Surface,最后将主 Surface 的内容发送给视频驱动,这样屏幕上就能看到想要的效果.

SurfaceFlinger 是视频系统的一个重要部分,它能够将各种应用程序的 2D 及 3D 图像界面进行组合. 每个应用程序可能对应着一个或者多个图像界面 Surface,各个 Surface 之间可能有重叠, SurfaceFlinger 能够解决这种图形重合问题. 它会把各个 Surface 组合成一个主 Surface,最后将主 Surface 的内容发送给视频驱动,这样屏幕上就能看到想要的效果.

Overlay 系统是视频输出的单独环节,可以利用硬件系统的支持实现. Overlay 系统的主要作用是:在主显示区域之外的显示区域使用叠加方式,放置于主显示区域之上. 主显示区域代表图像层,负责图像用户界面的输出. 经过 Overlay 系统叠加的显示区域代表视频层,负责视频播放输出. Overlay 系统包括本地框架层、服务层、硬件抽象层及驱动层,没有 JNI 和应用程序框架层. 应用层可以通过 ISur-

face 来获取 Overlay 系统的接口,从而通过 Overlay 实现视频数据的输出.

4 音视频系统的扩展设计

Android 音视频系统中的解码部分在 OpenCore 框架中实现,只支持 mp4 和 3gp 两种视频格式的解码,能够解码的媒体格式种类和效率越来越不能满足人们日益增长的需求. 在此通过移植 FFmpeg 框架中媒体文件的解析和音视频流的解码部分,以实现多种媒体格式的支持. FFmpeg 是一个开源跨平台的编解码框架,它提供了音视频录制、格式转换、编码及解码等功能^[5]. 在了解音视频系统结构的基础上,重新设计一套基于 FFmpeg 框架的音视频系统. 这样,Android 系统上就有了基于 FFmpeg 的音视频系统和 Android 音视频系统两套框架, Android 扩展后的音视频系统结构图如图 3 所示.

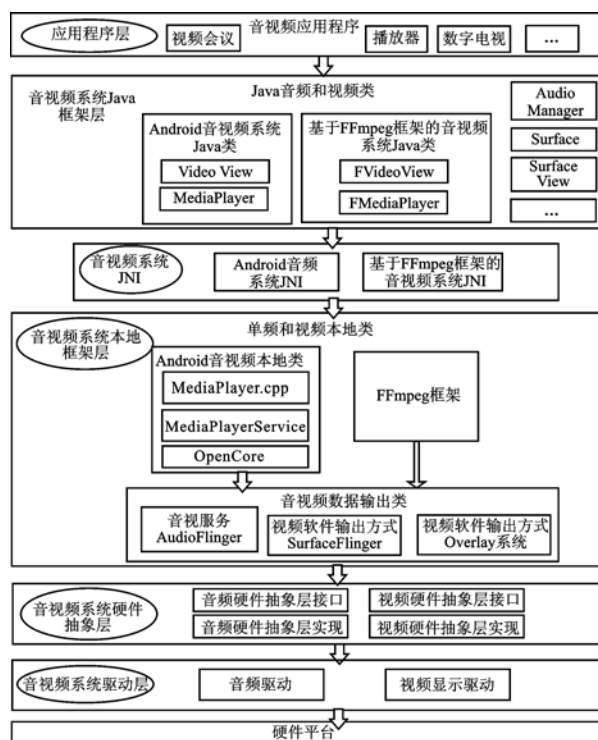


图 3 Android 扩展后的音视频系统结构图

5 音视频多媒体播放器的设计

为了支持多种媒体格式文件的播放,在此基于扩展后的音视频系统设计音视频播放器,主要应用 FFmpeg 框架中媒体文件的格式解析和解码库部分来完成媒体文件的拆分和解码功能^[6]. 音视频播放器的设计步骤如下:

Step1 剪裁和优化 FFmpeg 框架. FFmpeg 框架

提供了音视频录制、格式转换、编码及解码等多种功能.播放器实现时不需要这么多的功能,需要根据 Android 系统和 FFmpeg 框架的结构,将开源代码 FFmpeg 进行剪裁,消减 FFmpeg 中播放器不需要的功能.通过在配置命令中加入配置参数来实现 FFmpeg 的裁剪.同时对 FFmpeg 进行优化,使其适应移动终端设备的内存及处理器等限制条件,提高播放器的解码效率.

Step 2 移植 FFmpeg 框架到 Android 系统上.把 FFmpeg 框架放在 Android 的本地框架层,通过 JNI 和应用程序框架层中的类进行交互,这里需要定义一个包含 FFmpeg 框架接口的 JNI 来为应用程序框架层提供服务.用 C++ 语言编写一个封装 FFmpeg 提供的播放和播放控制等功能函数的源文件,作为 JNI 供 Java 上层调用.文件中定义 `zxfplayer_start()`、`zxfplayer_stop()`、`zxfplayer_pause()` 等播放控制方法.

Step3 将与媒体相关的结构体及功能在 Java 层进行封装,方便应用程序在 Java 层直接操作 FFmpeg API 来实现各种音视频文件的播放及播放控制功能.在音视频系统框架层,需要定义一个类似于 Android 音视频系统框架层中 `MediaPlayer` 的类 `FMediaPlayer`.在 `FMediaPlayer` 中添加 `start()`、`stop()`、`pause()`、`setDataSource()`、`seekTo()`、`getCurrentPosition()` 等播放控制方法,这些方法通过 JNI 调用 FFmpeg 框架提供的功能来实现.

Step4 把 FFmpeg 框架移植到 Android 系统上以后,可以使用 Android 音视频系统提供的服务来实现音视频数据的输出.通过音频服务 `AudioFlinger` 来实现音频数据的处理并传输音频数据到硬件设备.通过 `SurfaceFlinger` 或者硬件 `Overlay` 系统来实现视频数据的输出.

Step5 设计播放器界面和功能.调用 Android 应用程序框架层提供的各种服务设计播放器所需界

面和功能.播放器的播放和播放控制功能通过依次调用 `FMediaPlayer` 的方法来实现.

6 结束语

本文对 Android 源码中音视频系统的结构、Java 上层、本地库层、服务层及硬件抽象层等进行了分析;通过移植 FFmpeg 框架对音视频系统进行了扩展,使其能支持多种音视频格式的解析和解码;最后给出了使用扩展后的音视频系统开发移动终端多媒体播放器的方案,以供 Android 平台下音视频系统的研究者及播放器、视频会议等软件的设计者提供一个参考,并通过对音视频系统的扩展使其适用于日趋多样化的媒体文件格式,为基于 Android 的移动终端多媒体播放器设计者提供一种设计方案.

参考文献:

- [1] 郭宏志. Android 应用开发详解[M]. 北京:电子工业出版社,2010.
- [2] 朱婷婷,李惠. 基于 Android 的应用软件的综述[J]. 电脑与电信,2011(1):42-43.
- [3] Shawn Van Every. Pro android media:developing graphics, music, video,and rich media apps for smartphones and tablets[M]. New York:Apress,2009.
- [4] 胡泽,赵新梅. 流媒体技术与应用[M]. 北京:中国广播电视出版社,2006.
- [5] 韩超,梁泉. Android 系统级深入开发:移植与调试[M]. 北京:电子工业出版社,2011.
- [6] 韩超,梁泉. Android 系统原理及开发要点详解[M]. 北京:电子工业出版社,2010.

作者简介:

李向军 女,(1967—),硕士,教授.研究方向为计算机软件与理论、计算机体系结构、数据挖掘与知识发现等.

张小菲 女,(1987—),硕士.研究方向为计算机应用.

王书阵 男,(1978—),博士,副教授.研究方向为计算机软件与理论、计算机体系结构、物联网工程等.