资深 Android 系统工程师和软件开发工程师倾力打造 Android 专题书籍

知识框架完整、条理清晰、理论和实战相结合

深度解析 Android 手机定制开发中的重点和难点 Telephony 通信模块

杨青平◎著

Android Telephony

原理解析与开发指南

Android Telephony
Principle Analysis and Development Guide

中国工信出版集团

人民邮电出版社
POSTS & TELECOM PRESS

杨青平◎著

Android Telephony
原理解析与开发指南

Android Telephony
Principle Analysis and Development Guide

人 民 邮 电 出 版 社
北 京

图书在版编目（CIP）数据

Android Telephony原理解析与开发指南 / 杨青平著
. -- 北京 : 人民邮电出版社, 2018.9
ISBN 978-7-115-48915-9

Ⅰ. ①A… Ⅱ. ①杨… Ⅲ. ①移动终端－应用程序－
程序设计 Ⅳ. ①TN929.53

中国版本图书馆CIP数据核字(2018)第182123号

内 容 提 要

随着 Android 平台的应用越来越广泛，越来越多的人加入到 Android 系统的定制研发中来。Android 的基本通信功能是 Android 系统定制的核心模块，本书主要围绕 Android Telephony 关键业务流程展开，从接打电话、网络服务、数据上网三方面解析 Telephony。

全书共 10 章，主要内容包括初识 Android、搭建 Android 源代码编译调试环境、深入解析通话流程、详解 Telecom、详解 TeleService、Voice Call 语音通话模型、ServiceState 网络服务、Data Call 移动数据业务、SMS & MMS 业务、Radio Interface Layer。

本书适合计算机科学技术、信息技术、通信工程、软件工程等专业的研究生、本科学生、高职高专学生使用。

◆ 著　　杨青平
责任编辑　祝智敏
责任印制　马振武

◆ 人民邮电出版社出版发行　　北京市丰台区成寿寺路 11 号
邮编　100164　　电子邮件　315@ptpress.com.cn
网址　http://www.ptpress.com.cn
北京圣夫亚美印刷有限公司印刷

◆ 开本：787×1092　1/16
印张：19　　2018 年 9 月第 1 版
字数：519 千字　　2018 年 9 月北京第 1 次印刷

定价：59.80 元

读者服务热线：(010) 81055256　印装质量热线：(010) 81055316
反盗版热线：(010) 81055315
广告经营许可证：京东工商广登字 20170147 号

前　言

随着智能手机的日益普及，移动互联网正深刻改变着我们的日常生活，iOS 和 Android 两大移动操作系统功不可没。由于 Android 的开源，各大厂商纷纷通过定制 Android 系统降低研发成本和周期，从而快速推出智能产品。

Telephony 作为 Android 系统的核心业务，包括接打电话、手机上网、短信、彩信等应用场景，也是终端用户使用最多、最频繁的业务。因此，在定制 Android 系统的开发过程中，对 Telephony 业务的定制和修改也非常多，如优化通话界面、过滤骚扰短信和电话，分析短信内容和优化展示关键信息等，既能方便用户操作和使用，提升用户体验，也能增加广告收入。

Android Telephony 业务跨度大，涉及多个层之间的交互：应用层、系统框架层、HAL 硬件抽象层和 BP Modem。理解和掌握 Android Telephony 业务和技术，方能更好地完成 Android 系统定制、业务扩展和数据挖掘。

本书主要特点如下。

1．解析源代码并总结 Android Telephony 业务模型、设计原理和系统架构

Android 的源代码汇集了大量 Google 工程师的设计思想和理念。本书对 Telephony 源代码中的关键设计模式、设计原理和实现方式做了详细的分析和总结，可帮助读者拓展思路和加强训练，以提升软件设计水平和编码能力。

本书由浅入深地详细讲解了搭建 Android 编译环境、分析源代码并推导出 Telephony 业务模型，可帮助读者提升 Ubuntu 系统动手能力、Java 语言编程能力、Android 开发能力、UML 阅读能力，理解和掌握常用的设计模式，以及 Android 系统平台定制的核心工作。

2．合理、有效的组织

本书以 Telephony 关键业务流程为主线，分析和总结主要业务、核心实现和业务模型。首先分析和总结 Telephony 业务模型中 Voice Call 语音通话、ServiceState 网络服务、Data Call 移动数据和 SMS&MMS 的关键业务流程和核心实现机制，然后扩展 Telephony Phone、Voice Call、Data Call 等关键业务模型，同时输出时序图、类图、状态图等 UML 图和架构图，以帮助读者理解和掌握 Telephony 业务。

3．内容充实、实用并结合实践

本书紧紧围绕 Android Telephony 关键业务展开，主要包括 Voice Call 语音通话、ServiceState 网络服务、Data Call 移动数据和 SMS&MMS 中需要优化和定制的关键业务，以及 Radio Interface Layer（无线通信接口抽象层）的消息处理机制和系统架构。

本书总结的 Telephony 业务模型可在华为 Nexus 6P 手机上进行运行和验证，使读者对 Telephony 业务和实现有更加直观的认识和理解，并掌握调试 Android 代码的方法和技巧。

尽管作者在写作过程中力求准确、完善，但书中不妥或错误之处仍在所难免，殷切希望广大读者批评指正！恳请读者一旦发现错误，于百忙之中及时与作者联系，以便尽快更正，作者将不胜感激。E-mail：android_tele@163.com。

杨青平

2018 年 4 月

目　录

第 1 章

初识 Android

学习目标

- **学习智能手机基本硬件体系结构。**
- **掌握基于 Linux Kernel 的 Android 系统分层架构。**
- **掌握 Telephony 在 Android 系统中的结构。**

Android 中文意思为“机器人”，中文译名为“安卓”，是谷歌公司于 2007 年 11 月 5 日发布的基于 Linux 平台的开源手机操作系统，其由操作系统、中间件、用户界面和应用软件组成，号称首个为移动终端打造的真正开放和完整的移动软件。谷歌公司通过与电信运营商、手机设备制造商、芯片开发商及其他有关方面结成深层次的合作伙伴关系，希望借助建立标准化、开放式的移动电话软件平台，在移动产业内形成一个开放式的生态系统。

从 2007 年至今，经过长时间的考验，Android 已经成为全球最热门的手机操作系统之一。本章主要从智能手机的基本硬件结构、Android 手机操作系统整体架构和 Android 的 Telephony 模块的体系结构三个方面逐步认识 Android，特别将 Android 手机操作系统平台下的 Telephony 模块作为本章讲解的重点内容。

1.1 智能手机的系统结构

Android 手机的基本硬件结构符合智能手机的基本硬件结构，我们要学习 Android 移动开发，首先需要了解智能手机的硬件系统基本结构。

随着通信领域的快速发展，移动终端的发展和变化也非常巨大，已经由原来单一的通话功能、短信功能，向彩信、数据上网、图像处理、音乐和多媒体方向演变。到目前为止，市面上的移动手机基本上可以分成两大类：一类是功能手机（Feature Phone）；另一类是智能手机（Smart Phone）。

这两类手机如何区分呢？智能手机具有传统手机的基本功能，如打电话、发短信、照相等。智能手机的特点：具有开放的操作系统、硬件和软件的可扩充性和支持第三方的二次开发。相对于功能手机，智能手机就像计算机一样，可通过安装第三方软件来扩展其功能和应用，因此，智能手机

越来越受到人们的青睐，已成为手机终端市场的一种潮流。

那么先来看看智能手机较多采用的硬件基本结构，如图 1-1 所示。

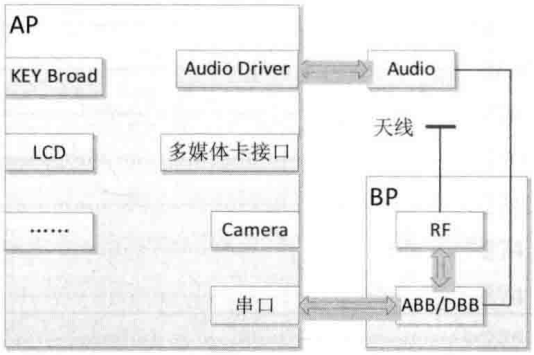

图 1-1　智能手机硬件结构图

智能手机的基本硬件结构大多采用双处理器架构：主处理器和从处理器。主处理器运行开放式操作系统以及操作系统之上的各种应用，负责整个系统的控制；从处理器负责无线通信基本能力，主要包括 DBB（Digital Baseband，数字基带）和 ABB（Analog Baseband，模拟基带），完成语音信号和数字信号调制解调、信道编码解码和无线 Modem 控制。

主处理器也叫 AP（Application Processor，应用处理器），从处理器也叫 BP（Baseband Processor，基带处理器），它们之间通过串口、总线或 USB 等方式进行通信。不同手机芯片生产厂家采用的集成方式都不一样，目前市面上仍以串口通信为主。

不难发现，在智能手机的基本硬件结构中，BP 部分只要再加一定的外围电路，如音频芯片、LCD 控制、摄像机控制器、扬声器、天线等，就是一个完整的普通手机的硬件结构。

注 意

现在我们能区分功能手机与智能手机吗？回顾手机终端的发展历程，不难发现这样一条规律：随着手机芯片处理能力的提升、上网能力的扩展和发展（蓝牙、WiFi、3G 网络），手机应用得到非常广泛的扩大和发展。在智能手机的硬件设计上，采用处理能力比较强大的处理器作为 AP，来支持开放手机操作系统及操作系统之上的扩展应用，由此可见智能手机发展的趋势和方向。

1.2 Android 系统架构

前面学习了智能手机的基本硬件结构，可通过功能手机与智能手机的特点和区别从本质上去认识它们。Android 作为一款运行在 AP 上的开源智能手机操作系统，其系统架构是什么样的呢？我们先来看看图 1-2。

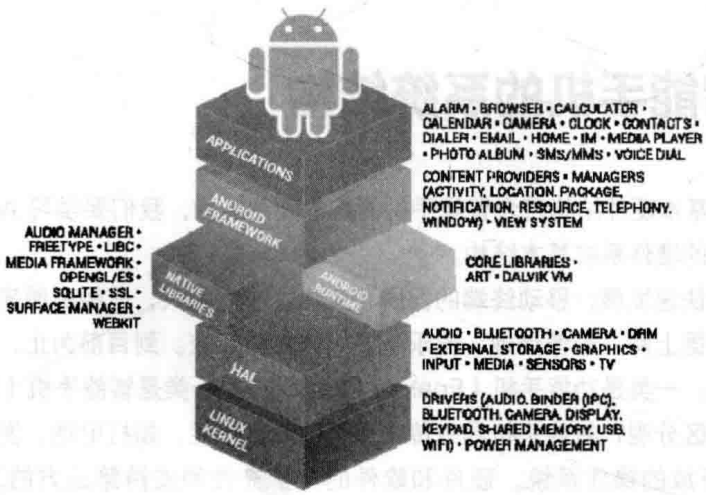

图 1-2　Android 系统架构

通过图 1-2 不难发现，Android 是一个分层的基于 Linux Kernel 的智能手机操作系统，共分为四层，从上到下依次是应用层（Applications）、应用框架层（Framework）、系统运行库层（Libraries）和核心层（Linux Kernel），下面将对这四层进行简要的分析和介绍。

1.2.1　应用层

Android 近几年的发展可谓是非常迅猛，有一个很重要的原因，那就是它的应用非常多。安卓市场已发布的软件个数和软件下载量目前仅次于苹果的应用商店，并且保持着快速增长态势。只有这些第三方开发的应用（如游戏、导航、播放器、桌面主题等）日益丰富，手机终端用户才能不断地发展和壮大，而这些应用均在应用层运行。

应用层包括了各种 Android 应用程序，这些应用程序是使用 Java 语言开发，并运行在 Dalvik 虚拟机上，处于 Android 系统架构中的第一层。在 Android 源码和 SDK 中，谷歌公司已经捆绑和发布了一些核心应用及源代码，如 Dialer、MMS、日历、谷歌地图、浏览器、联系人等。

1.2.2　应用框架层

如图 1-2 所示，Android 系统架构中的第二层是应用框架层，是用 Java 语言实现和开发的。有了应用框架层，开发者使用该层提供的 API 便可非常方便地完成访问设备硬件、获取位置信息、向状态栏添加通知消息、设置闹铃等操作，而不必关心具体的底层实现机制和硬件实现方式。这样，简化了 Android 应用开发者开发程序时的架构设计，从而能够快速开发新的应用程序。

应用框架层是谷歌公司发布核心应用时所使用的 API 框架，开发人员可以使用这些框架提供的 API 来快速开发自己的应用程序。下面是对 Android 中一些主要的组件的简要总结及相关说明。

- 视图（View）

在 Android SDK（Software Development Kit，软件开发工具包）中介绍了其丰富的视图的使用方法及相关属性，所有的 Android 应用程序都由这些视图构成，主要包括列表（List）、网格（Grid）、文本框（Text）、按钮（Buttons）等基础 Android 应用的界面控件。

- 资源管理器（Resource Manager）

提供非代码资源转换和访问，如本地字符串（xml 文件配置）、图片和布局文件（Layout File，使用 xml 文件配置）。

- 通知管理器（Notification Manager）

应用可以在状态栏中显示自定义的提示信息，如新短信通知、未接来电通知、手机信号量通知等。

- Activity 管理器（Activity Manager）

用来管理 Android 应用程序界面的生命周期（onCreate 创建、onResume 显示、onPause 暂停、onStop 停止等），一个手机屏幕界面可对应一个 Activity。

1.2.3　系统运行库层

如图 1-2 所示，Android 系统架构中的第三层为系统运行库层，这一层主要包含了手机操作系统平台必备的 C/C++核心库、Dalvik 虚拟机运行环境和 HAL 子层。我们跳过 HAL，先简单地介绍和分析 C/C++核心库和 Dalvik 虚拟机运行环境。

1. C/C++核心库

系统运行库层包含一个 C/C++库的集合，当使用 Android 应用框架的一些接口时，系统运行库层通过 C/C++核心库来支持对应的组件使用，使其能更好地为 Android 应用开发者服务。下面是一些主要的核心 C/C++库及其简要说明。

- libc（系统 C 库）

C 语言标准库，处于系统最底层的系统库，由 Linux 系统来调用。

- Media Framework（多媒体库）

Android 系统多媒体库，支持当前手机平台上主流的音频和视频格式播放和录制，以及静态图像。如 MPEG-4、MP3、AAC、JPG、PNG 等多媒体格式。

- SGL

2D 图形引擎库

- OpenGL

3D 效果的支持。

- SQLite

轻量级关系数据库引擎，可用来增、删、改、查通话记录、联系人等信息。

- WebKit

新式的 Web 浏览器引擎，支持当前非常流行的 HTML 5。

- SSL

基于 TCP/IP 网络协议，为数据安全通信提供支持。

2. Dalvik 虚拟机运行环境

系统运行库层包含了 Android Runtime，其核心为 Dalvik 虚拟机。每一个 Android 应用程序都运行在 Dalvik 虚拟机之上，且每一个应用程序都有自己独立运行的进程空间；Dalvik 虚拟机只执行 DEX 可执行文件。

DEX 格式是专为 Dalvik 设计的一种压缩格式，适合内存和处理器速度有限的系统。要生成 DEX 格式文件，首先通过 Java 程序编译生成 class 文件，然后通过 Android 提供的 dx 工具将 class 文件格式转换成 DEX 格式。

Dalvik 虚拟机的特性总结如下。

- 每一个 Android 应用运行在一个 Dalvik 虚拟机实例中，而每一个虚拟机实例都是一个独立的进程空间。
- 虚拟机的线程机制、内存分配和管理、Mutex（进程同步）等的实现都依赖底层 Linux 操作系统。
- 所有 Android 应用的线程都对应一个 Linux 线程，因而虚拟机可以更多地使用 Linux 操作系统的线程调度和管理机制。

注 意

因为 Android 的编程语言是 Java 的缘故，我们很容易将 Dalvik 虚拟机与 Java 虚拟机误认为是同一个东西。但 Dalvik 虚拟机并不是按照 Java 虚拟机的规范来实现的，两者并不兼容；它们之间最大的不同在于 Java 虚拟机运行的是 Java 字节码，而 Dalvik 虚拟机运行的是其专有的文件格式——DEX（Dalvik Executable）文件。

1.2.4 核心层

Android 4.0 基于 Linux Kernel 3.0.8 提供核心系统服务，如文件管理、内存管理、进程管理、网

络堆栈、驱动模型等操作系统的基本服务能力。Linux Kernel 同时也作为硬件和软件之间的抽象层，需要一些与移动设备相关的驱动程序来支持，主要的驱动如下。

- 显示驱动（Display Driver）

基于 Linux 的帧缓冲驱动。

- 键盘驱动（Keyboard Driver）

输入设备的键盘驱动，如 Home（待机）、Menu（菜单）、Return（返回）、Power（电源）等设备按键。

- 音频驱动（Audio Driver）

常用的基于 ALSA（Advanced Linux Sound Architecture）的高级 Linux 声音体系驱动。

- 电源管理（Power Management）

如电池电量、充电、屏幕开启关闭管理等。

- Binder IPC 驱动

Android 平台上一个特殊的驱动程序，具有单独设备访问节点，用来提供 IPC 进程间的通信功能。

- 蓝牙驱动（Bluetooth Driver）

基于 IEEE 802.15.1 标准的蓝牙无线传输技术。

- WiFi 驱动（WIFI Driver）

基于 IEEE 802.11 标准的 WiFi 驱动。

- 照相机驱动（Camera Driver）

常用的基于 Linux 的照相机驱动。

1.3 Android Telephony 框架结构

通过前面对 Android 手机操作系统整体框架结构及每一层的简单分析和说明，相信大家对 Android 智能手机操作系统已经有了一些基本的了解和认识。结合 Android 手机操作系统的整体框架，我们接着学习 Android Telephony 涉及的框架结构，首先看图 1-3。

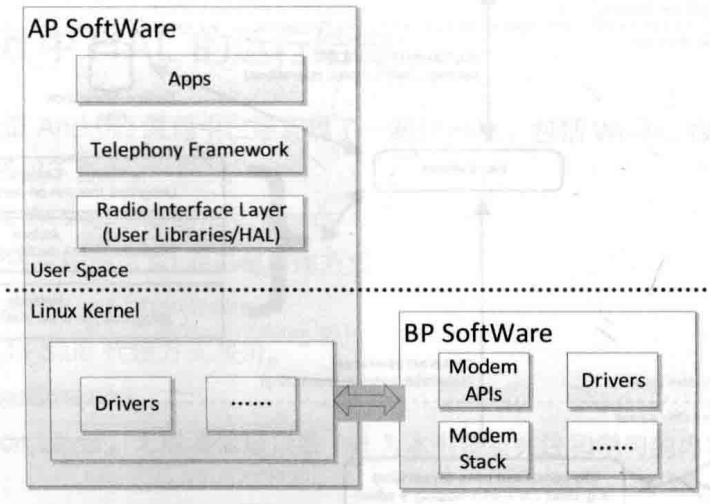

图 1-3　Android Telephony 框架结构

通过图 1-3 不难发现 Android Telephony 框架结构的一些规律，总结如下：

- Android Telephony 的业务应用跨越 AP 和 BP，AP 与 BP 相互通信，符合前面介绍的智能手机的硬件基本结构。
- Android 系统在 AP 上运行，而 Telephony 运行在 Linux Kernel 之上的用户空间。
- Android Telephony 也采用了分层结构的设计，共跨越了三层：应用层、应用框架层和系统运行库层，与 Android 操作系统整体分层结构保持一致；
- Android Telephony 从上到下共分三层：Telephony 应用、Telephony 框架、RIL（Radio Interface Layer，无线通信接口层，主要位于系统运行库层的 HAL 中，什么是 HAL，接下来会详细介绍）。
- BP SoftWare 在 BP 上运行，主要负责实际的无线通信能力处理，不在本书讨论的范围。

1.3.1 系统运行库层的 HAL

HAL（Hardware Abstraction Layer，硬件抽象层）在 Linux 和 Windows 操作系统平台下有不同的实现方式。

Windows 下的 HAL 位于操作系统的最底层，它直接操作物理硬件设备，用来隔离与不同硬件相关的信息，为上层的操作系统和设备驱动程序提供一个统一接口，起到对硬件的抽象作用。这样更换硬件后编写硬件的驱动时，只要实现符合 HAL 定义的标准接口即可，而上层应用并不会受到影响，也不必关心具体实现的是什么硬件。

Linux 下的 HAL 与 Windows 下的 HAL 不太一样，HAL 并不是位于操作系统的最底层，直接操作硬件；相反，它位于操作系统核心层和驱动程序之上，是一个运行在用户空间中的服务程序。

1.3.2 简析 HAL 结构

通过前面的学习，我们知道 Android 是基于 Linux Kernel 的开源智能手机操作系统，所以这里重点介绍基于 Linux 下的 HAL 结构，就不再单独介绍 Windows 下的 HAL 结构。

要想知道 HAL 结构，先看看来源于 HAL 0.4.0 Specification 的框图，如图 1-4 所示。

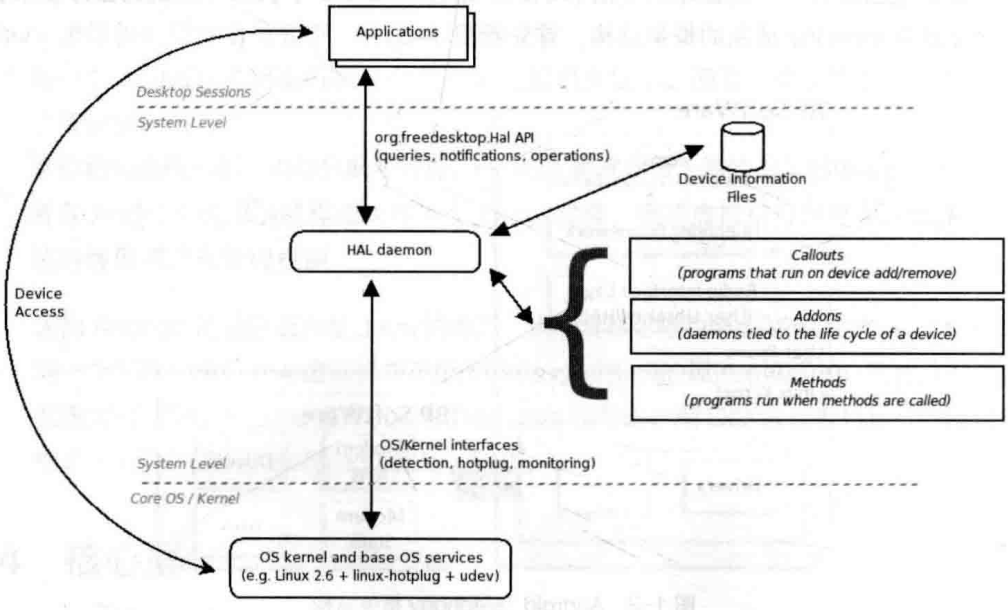

图 1-4 HAL 0.4.0 Specification