Volume 3, Issue 6, June 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Android Internals

Vaibhav Kumar Sarkania, Vinod Kumar Bhalla
Thapar university
Punjab, India

Abstract-Android is a linux based operating system which uses linux kernel. In this paper we will see how the boot up process of android is different from linux and how the different applications in an Android System communicate with each other. As an Android Application is made up of different activities so through this paper we will also come to know about how the activity changes their states and the whole lifecycle of an activity.

Keywords: - Android Boot up, Activity Lifecycle, Zygote, Dalvik VM.

I. INTRODUCTION

Android is a free and open source operating system. It was initially developed by Android inc. but later in 2005 it was purchased by Google. In 2007 a group of 78 companies formed a group called Open Handset Alliance (OHA) to develop and distribute Android [4]. It is an operating system for low powered devices, those runs on battery and is full of hardware. Android applications uses hardware features through abstraction and provide a defined environment to run applications. An Android application is written in java and run in a virtual machine which is called Dalvik virtual machine which executes its own byte codes [1].

II. ANDROID BOOTUP PROCESS

2.1. Power On

Master boot record (MBR) is a boot sector which contains partition table which has the information about how the device is partitioned in a structure. There is no MBR or partition when the device is started for the first time. When the phone is switched on, CPU will be in a no initialization state. Internal RAM is available and no internal clocks are set up. The device starts executing code located in the ROM and finds a specific block which has first Stage boot loader. The first boot loader points to a second stage boot loader, which is located in a known block. This "pointing" process is called raw partition table [6].

2.2. Boot loader

Boot loader is a code which is executed before android operating system runs. It loads kernel to the RAM and sets up the initial memories. Manufacturers use existing boot loaders or they create their own boot loaders.

- The First stage boot loader will find and setup the external RAM.
- Now Main boot loader is loaded and placed in external RAM as the RAM is available.
- The First important program is in the second boot loader stage which contains code for file systems, additional memory and network support etc.
- When the boot loader is done it goes to the linux kernel [6].

2.3. Linux kernel

A kernel acts as a bridge between hardware and software. It setups cache protected memory, scheduling and loads drivers. After initializing Memory management units and caches, virtual memory can be used and user space processes can be launched by the system. After finishing the setup Kernel looks for init process which can found under system/core/init and launch it [1].

2.4. Init process

This process is the root process. Every process will be launched from this process. Init process mounts directories like /sys, /dev, /proc. It will run init.rc script and system service processes. This script is located in system/core/rootdir in the Android open source project and describes system services, file system and other parameters [1].

2.5. Zygote

After starting various daemons like Android Debug Bridge (adbd), Radio Interface Layer Daemon (rild), etc, Init process initiates a process called Zygote. In java there is a separate instance of a Virtual Machine for each application. In android Dalvik, virtual machine is used as VM. So there is high consumption of memory and time

because of different instances of dalvik VM for every application. Now Zygotes comes into play. It enables shared code across Dalvik VM, lower memory footprint and minimal startup time. Zygote process starts at system boot up and it preloads and initializes core library classes. After initialization, zygote process waits for socket request coming from the runtime process. If any request comes then it forks starts processes with VM instances [6].

2.6. Runtime process

The next init initiates the Runtime process and this process starts the service manager. All the services should be registered with the service manager and it provides local lookup service and binds services given their name. The runtime requests zygote to start system server process. Zygote splits and starts up a new dalvik vm instance and starts the service. To control display device and audio output device the system server starts surface flinger and audio flinger. These services get registered with the service manager so that other applications can use display and audio. Now the system server will start all the core platform services and hardware services like activity manager, window manager and power manager etc. All of these services will get registered with the service manager [2].

I

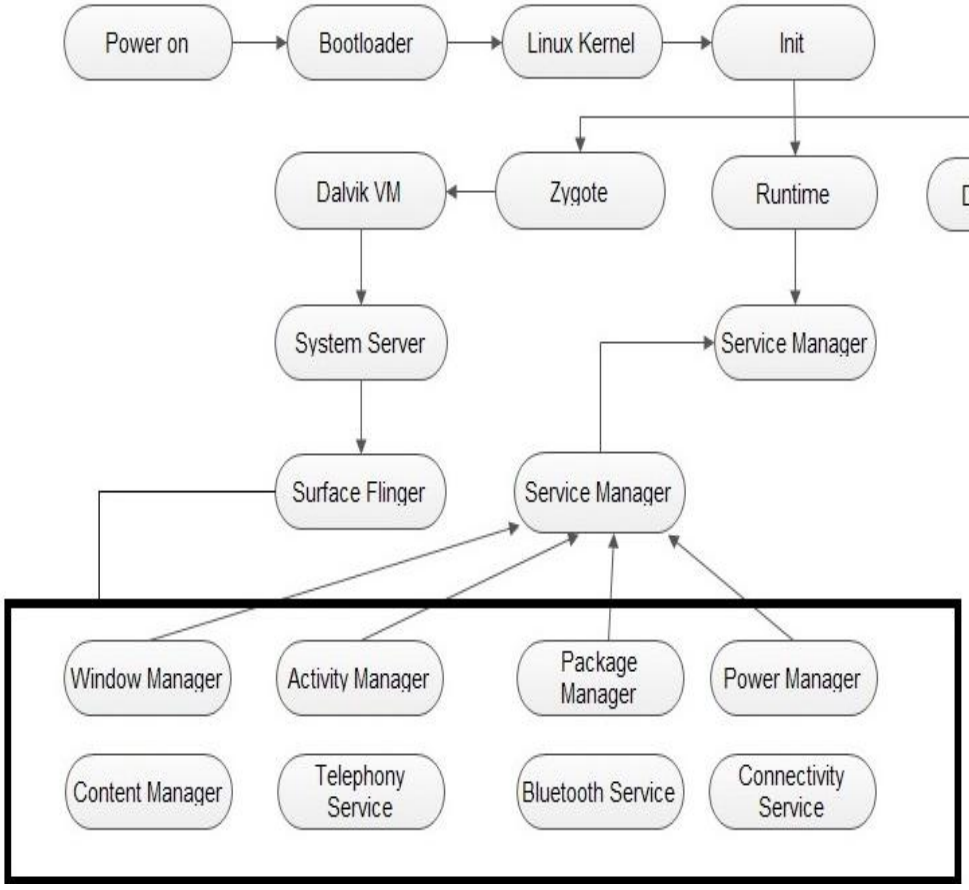

Fig 2.1 Boot up Process

III. INTERPROCESS COMMUNICATION (IPC)

At this point home screen or idle screen is launched. Activity manager will send a request to zygote to initiate the "home" activity and in return Zygote will fork a new process a dalvik VM and home activity. Now for each application launched by the user, Zygote will fork each time and create a new dalvik VM instance inside a new process. A unique user id is assigned to each application. An application has access to only those files which it needs and these permissions are set up by the system [3].

Applications run in separate processes, so to communicate with each other and with the system services an IPC (Inter Process Communication) is needed and this mechanism in android is known as Binder and is based on shared memory. On registration of each process with the service manager, it gets a reference called a context object. Let there be an application A and service B which is running in spate processes and wants to communicate with each other application. A passes the name of the service to context and requests for service. B in return context sends a reference to the service to A [3].

After getting the reference app, A calls a method which is intercepted by the Binder which arranges the object and passes the reference to Receiver. The object is serialized because a proxy object is passed and not the original objects.

There is a thread pool maintained by the binder at Receiver B side and one of the threads receives the incoming call, locates the actual object and makes the call. Return value is passed back to the application A [6].

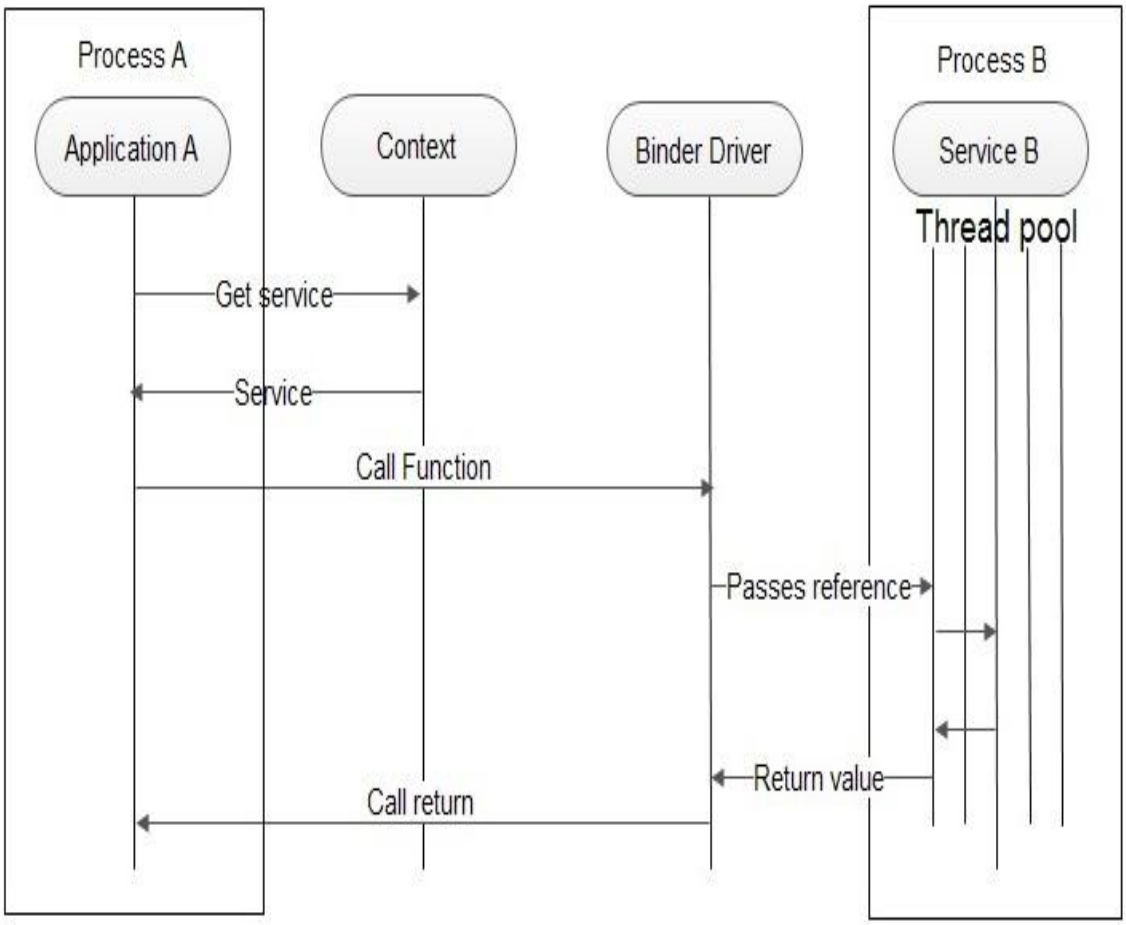

Fig 3.1 Interposes Communication

IV. APPLICATIONS

An android application consists of four components:

1. Activity: An activity is a single screen of an application with which user can interact like click a photo, dial a number etc. Intents are used for transition between different activities and each application can have multiple activities.
2. Service: A service is a component that does not provide any user interface but runs in the background to perform long-running tasks.
3. Content Provider: To exchange data between different applications, a component is used. It is called content provider which handles retrieval of data and stores data in database files or on a network.
4. Broadcast Receiver: The broadcast messages from other application or from system are called intents and the component which responds to these intents is called broadcast receiver [2].

4.1. Activity States

Activities can be seen as website. Just as a website contains multiple web pages, an android app contains multiple activities. One webpage can redirect to another page and so on. In an android app one activity can redirect to another and so on. All the handling activities is done by activity Manager. Its task is to create, destroy and manage activities.

4.1.1. Activity states

1. Active/Running: An activity is said to be in running state if it is completely visible and the user can interact with it. There can be only one running activity at a given time.
2. Paused State: An activity is said to be in paused state if it is not in focus but partially visible. For example while using an app on android if any notification or a dialog box appears then the activity of the app goes in paused state. While in paused state, an activity still maintains all states. It remains attached to the window manager and it can be killed by OS under low memory.
3. Stopped: An activity is said to be in 'stopped state' when it is not at all visible on screen but it is still alive and maintains all states. To fulfil the resource requirements of higher priority activities, it can be killed by OS.
4. Destroyed state: An activity is said to be destroyed when it is no longer in memory. OS destroys an activity after a 'paused' or 'stopped state' to free the resources [7].

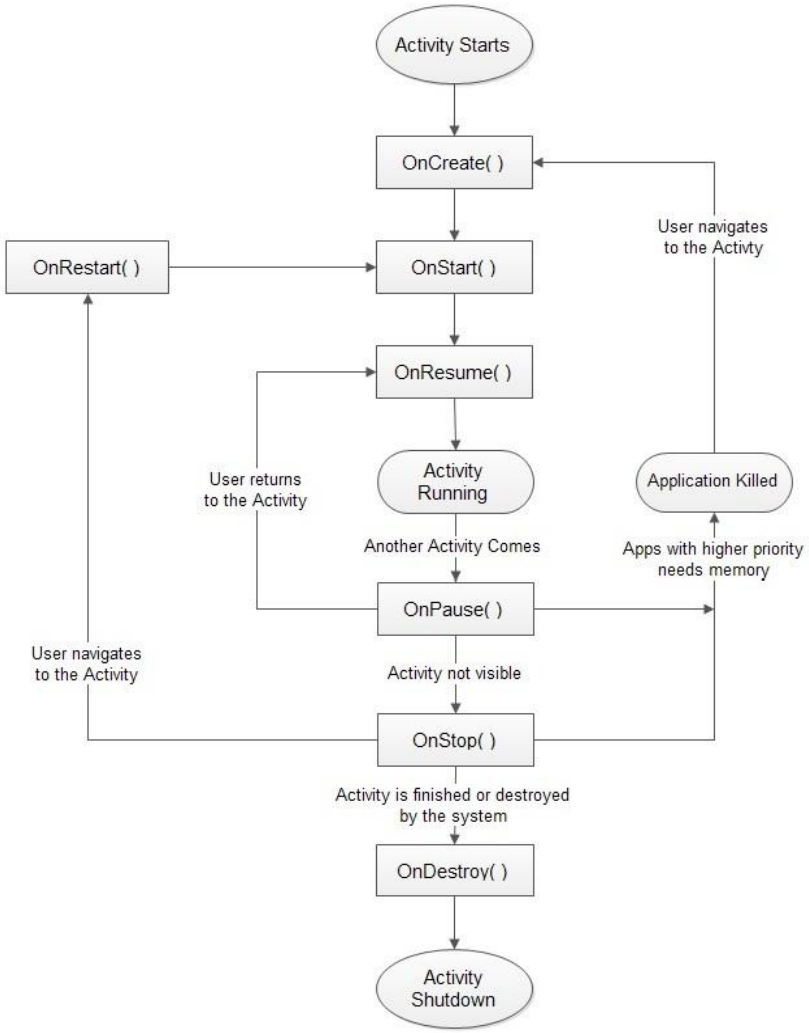

Fig 4.1 Activity Lifecycle

4.1.2. Activity Lifecycle Methods

1. Oncreate( ) : When an Activity is launched the first method called is the oncreate( ) method All the User Interface creation and initialization of data elements is done in this method. A bundle object as parameter is passed in this method to restore the UI state [7].
2. Onstart( ): Just before an Activity is being visible to the user, Onstart() method is called and if there is any task needed to be performed just before the Activity becomes visible to the user, it is defined in this method as ramping up frame rates [5].
3. OnResume( ): Now the activity is in 'active' or 'running state' and user can interact with the activity. This method can also be called when the activity is in 'paused state' to make it in 'running state' [8].
4. Onpause( ) : When an activity is in running state, a user might navigate to other parts of the system or the system is about to put an activity in background then Onpause( ) method is called. After this an activity can be killed by the system or Onresume( ) method can be called to resume an activity [5].
5. Onstop( ): When an Activity is not visible to the user, Onstop method is called and the application can be killed at anytime by the system in case of low memory. After this, an activity is either restarted or destroyed [8].
6. Onrestart( ): In 'stopped state' an activity can't be resumed but it can be restarted by calling Onrestart method and it is always followed by onstart method [5].
7. Ondestroy( ) : This method is called just before an activity is destroyed because the activity has finished or system kills it to save space. This is the last method called on an Activity [5].

V. CONCLUSION

Android has emerged as a strong competitor in mobile sector as it is supported by large companies especially by Google. Manufacturers can modify the system as per their needs due to Android's openness and extensibility. Today Android operating system is not only used in Mobiles and tablets, its implementation in electronic devices is increasing rapidly. Smart TV and Smart Camera are examples of new implementation and in future android will be in many household devices like washing machine, Oven and many more.

REFERENCES

[1] Benjamin Speckmann, "The Android mobile platform," MS.Thesis, Depart. comp. science, Eastern Michigan Univ., Michigan, US, 2008.
[2] Benny Skogberg "Android Application Development," MS.Thesis, Depart. Comp. Science, Malmo Univ., Sweden, 2010.
[3] Chien-Wei Chang, Chun-Yu Lin, Chung-Ta King,"Implementation of JVM Tool Interface on Dalvik Virtual Machine",paper appears in (VLSI-DAT), 2010 International Symposium on Digital Object Identifier 26-29 April 2010.
[4] Bimal Gadhavi & Khushbu Shah," Analysis of the Emerging Android Market ", Project Report Presented to SanJosé State University May 2010.
[5] Stefan Brahler," Analysis of android architecture", Department of computer science, Karlsruher institute of technologies, Germany, june 2010
[6] Marakana,"AndroidBootcamp TrainingCourse "http://marakana.com/training/android/android_bootcamp.html, 18 Oct, 2012.
[7] James Steele, The Android Developer's Cookbook, New York : Pearson, 2011.
[8] http://www.developer.android.com, 23 Oct, 2012.

