ISSN 2319 – 7757 EISSN 2319 – 7765

discovery Cover the uncovered

Indian Journal of Engineering

Dalvik - Virtual Machine

Ashish Yadav J[1], Abhishek Vats J[2], Aman Nagpal J[3], Avinash Yadav J[4]

1. Department of Computer Science, Dronacharya college of Engineering, Gurgaon, India, E-mail: ashish.y124@yahoo.com
2. Department of Computer Science, Dronacharya college of Engineering, Gurgaon, India, E-mail: abhishekvats_0909@yahoo.com
3. Department of Computer Science, Dronacharya college of Engineering, Gurgaon, India, E-mail: russel.aman@gmail.com
4. Department of Computer Science, Dronacharya college of Engineering, Gurgaon, India, E-mail: av9sh.hr@gmail.com

Received 26 September; accepted 19 October; published online 01 November; printed 16 November 2012

ABSTRACT

Android is a software stack for mobile devices that contains an operating system, middleware and key applications. Android is a software platform and operating system for mobile devices based on the Linux operating system and developed by Google and the Open Handset Alliance. It allows developers to Write handle code in a Java-like language that utilizes Google-developed Java libraries, but does not support programs developed in native code. The presentation of the Android platform on 5 November 2007 was announced with the founding of the Open Handset Alliance, a consortium of 34 hardware, software and telecom companies devoted to advancing open standards for mobile devices. When released in 2008, most of the Android platform will be made obtainable under the Apache free-software and open-source license. Open Android provide the permission to access core mobile device functionality through standard API calls. All applications are equivalent Android does not differentiate between the phone's basic and third-party applications even the dialer or home screen can be replaced. Breaking down boundaries combine information from the web with data on the phone such as contacts or geographic location to create new user experience. Fast and easy development The SDK contain what need to build and run Android applications, counting a true device emulator and advanced debugging tools.

Keywords:- Virtual machine, Java Virtual Machine, Register based VM, Dalvik Virtual Machine, .dex file format, Licence, Security, Architecture

1. INTRODUCTION

The Dalvik virtual machine is a register-based virtual machine, intended and written by Dan Bornstein with contributions from other Google engineers as part of the Android mobile phone platform. It is optimized for low memory needs, and is designed to allow multiple VM instances to run at once, relying on the fundamental operating system for process isolation, memory management and threading support. Dalvik is often referred to as a Java Virtual Machine, but this is not strictly exact, as the bytecode on which it operates is not Java bytecode. Instead, a tool named dx, included in the Android SDK, transform the Java Class files of Java classes compiled by a regular Java compiler into another class file format (the .dex format).

2. VIRTUAL MACHINE

A Virtual Machine (VM) is a software environment that can be an emulator, an operating system or a total hardware virtualization, that has an implementation of resources without the real hardware being present. As an emulator it permit applications and operating systems to run on hardware that has a different processor architecture than the present one, while as an operating system VM virtualizes the server on operating system, and in-case of hardware virtualization two or more different operating systems can run simultaneously on the same hardware. Thus the key idea of virtual machine is to provide an environment that can execute instructions other than those linked with the host environment regardless of the hardware and software.

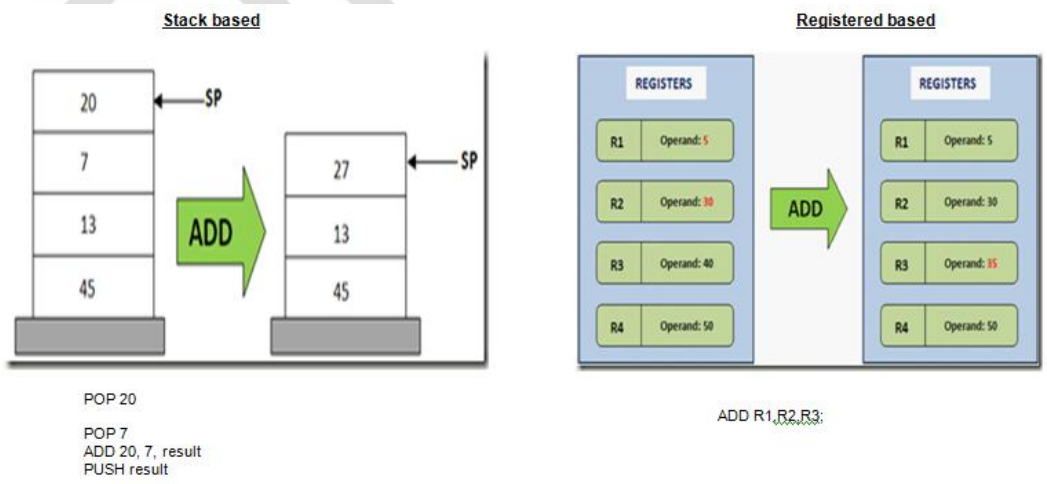

Figure 1

Satck Based V/s Registered Based

2.1. Types of Virtual Machines

VM's act as hardware for performing operations just as if genuine hardware is present. VM can be divided into two categories.

2.1.1. Based on its working and functionality

1. System Virtual Machine (supports execution of a complete operating system)
2. Process Virtual Machine (supports execution of a single process)

2.1.2. Based on its architecture

1. Stack based VM (uses instructions to load in a stack for execution)
2. Register based VM (uses instructions to be encoded in source and destination registers)

2.2. System VM vs Process VM

Virtual Machines mostly divide into two broad categories, i.e. System VM (also known as hardware virtual machine) and Process VM (also known as application virtual machine). The categorization is based on their usage and level of correspondence to the connected physical machine. The system VM simulates the complete system hardware stack and supports the execution of complete operating system. On the other hand, Process VM adds up layer over an operating system which is use to replicate the programming environment for the execution of individual process. The main advantages of system VM includes consolidation (it allows the multiple operating systems co- existence on single computer system with strong isolation to each other), application provisioning, maintenance, high availability & disaster recovery. Besides later advantages, regarding development aspects their advantages are that it allows sandboxing, faster reboot and better debugging access.

2.3. Stack Based VM vs Register Based VM

For many years, in virtual machine architecture, the stack based VMs was the architectural option due to simplicity of the VM implementation and the size of executables for stack architectures (Fig.1). Registered-based VM can be an attractive substitute to stack-based VM due to the reason that it allows a number of executed VM instructions to be substantially reduced. A study on analysis of different VM shows that register-based architecture require an average of 47% less executed VM instructions than the stack based. On the other hand the register code is 25% larger than the corresponding stack code but this increased cost of fetching more VM instructions due to larger code size involve only 1.07% extra real machine loads per VM instruction which is negligible. The overall performance of the register-based VM is that it takes, on average, a 32.3% less time to execute standard benchmark.

3. JAVA VIRTUAL MACHINE

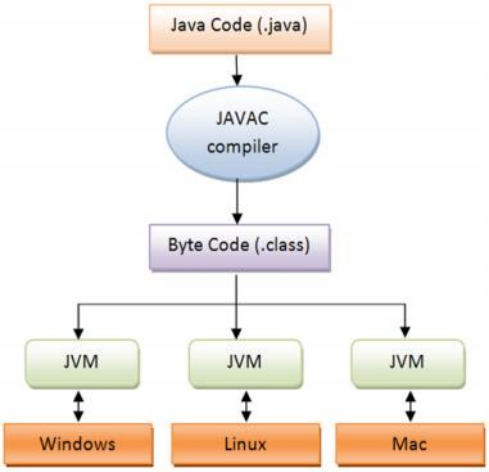

Figure 2
Working of Java Virtual Machine

A Java virtual machine (JVM), an implementation of the Java Virtual Machine Specification, interprets compiled Java binary code (called bytecode) for a computer's processor (or "hardware platform") so that it can execute a Java program's instructions. Java was planned to allow application programs to be built that could be run on any platform without having to be rewritten or recompiled by the programmer for each separate platform. A Java virtual machine makes this likely because it is aware of the specific instruction lengths and other particularities of the platform. The Java Virtual Machine Specification defines an abstract rather than a genuine machine or processor. The Specification specifies an instruction set, a set of registers, a stack, a "garbage heap," and a method area. Once a Java virtual machine has been implemented for a given platform, any Java program (which, after compilation, is called bytecode) can run on that platform. A Java virtual machine can also interpret the bytecode one instruction at a time (mapping it to a real processor instruction) or the bytecode can be compiled additional for the real processor using what is called a just-in-time compiler. Now we describe the working of the Java Virtual Machine. In this First the programmer write the program in java language and the save the program with (.java) file format. JVM consists of Just in Compiler which compiles the .class file into Bytecode (.class) file format. After that the Java Virtual Machine runs the class file format on different platforms (Fig.2).

4. DALVIK VIRTUAL MACHINE

Dalvik is a virtual machine that is planned specifically for the Android platform. Named after the fishing village of Dalvik in Iceland, it was initially written by Dan Bornstein. Unlike most of virtual machines that are stack based, Dalvik architecture is register based. It is optimized to use less space. The interpreter is simplifed for faster execution. It executes its own Dalvik byte code rather than Java byte code. Byte Codes are discussed (Fig.3).

4.1. Application Framework

Developers have complete access to the same framework APIs used by the core applications. The application architecture is designed to simplify the use again of components; any application can publish its capabilities and any other application may then make use of those capabilities (subject to security constraints enforced by the framework). This same mechanism allows components to be replaced by the user. Underlying all applications is a set of services and systems, including:

- A rich and extensible set of Views that can be used to construct an application, including lists, grids, text boxes, buttons, and even an embeddable web browser
- Content Providers that enable applications to admission data from other applications (such as Contacts), or to share their own data.
- A Resource Manager, providing access to non-code resources such as localized strings, graphics, and latfiles
- A Notification Manager that enables all applications to display custom alert in the status bar
- Activity Manager that manages the life cycle of applications and provide a common navigation back stack

4.2. Libraries

Android includes a set of C/C++ libraries used by different components of the Android system. These capabilities are exposed to developers through the Android application framework. Some of the core libraries are listed below System C library

- a BSD → derived implementation of the standard C system library (libc), tuned for fixed Linux-based devices
- Media Libraries → based on Packet Video's Open CORE; the libraries support playback and recording of lots of popular audio and video formats, as well as static image files, including MPEG4, H.264, MP3, AAC,AMR, JPG, and PNG
- Surface Manager → manages admission to the display subsystem and seamlessly composites 2D and 3Dgraphic layers from multiple applications·
- SGL → the underlying 2D graphics engine 3D libraries an implementation base on OpenGL ES 1.0 APIs; the libraries use either hardware 3Dacceleration (where available) or the included, highly optimized 3D software rasterizer
- Free Type → bitmap and vector font rendering
- SQLite → a powerful and lightweight relational database engine available to all applications.

4.3. Android Runtime

Android include a collection of core libraries that provides most of the functionality available in the core libraries of the Java programming language. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run multiple Vms efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for least memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been transformed into the .dex format by the included "dx" tool. The Dalvik VM depends on the Linuxkernel for underlying functionality such as threading and low-level memory management.

4.4. Linux Kernel

Ashish Yadav et al.
Dalvik - Virtual Machine,
Indian Journal of Engineering, 2012, 1(1), 100-104,
http://www.discovery.org.in/ije.htm

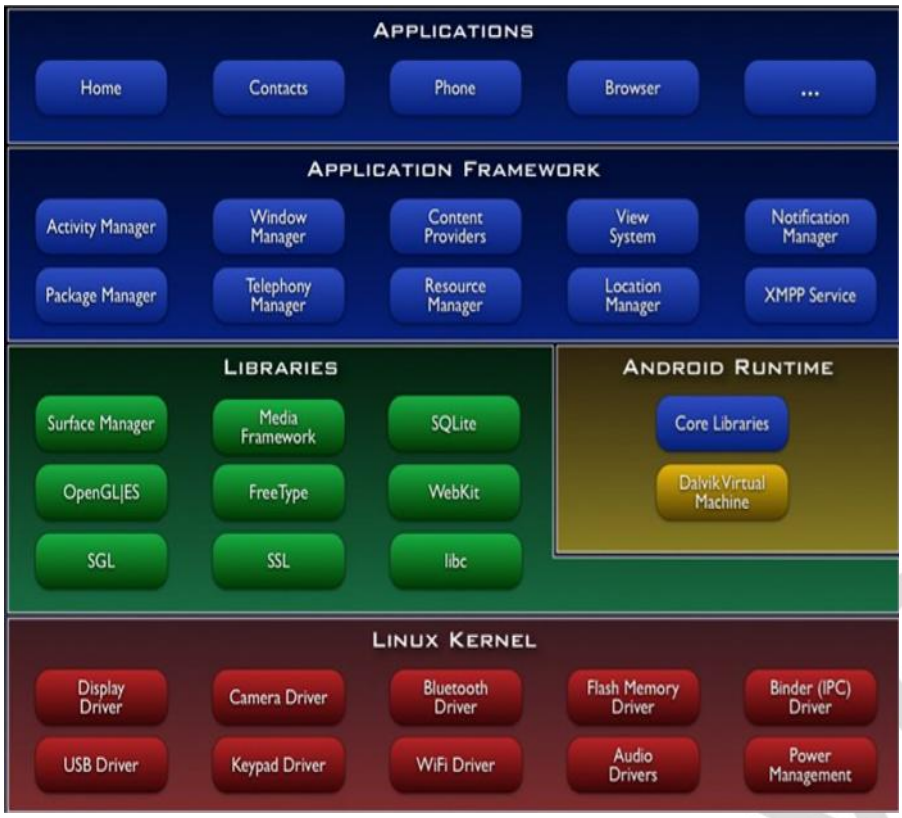

Figure 3
Description of each Layer of DVM

Android Architecture is based on Linux 2.6 kernel. It helps to handle security, memory management, process management, network stack and other important issues. Therefore, the user should bring Linux in his mobile device as the main operating system and install all the drivers required in order to run it. Android provides the support for the Qualcomm MSM7K chipset family. For instance, the current kernel tree supports Qualcomm MSM 7200A chipsets, but in the second half of 2008 we should see mobile devices with stable version Qualcomm MSM 7200, which includes main features:

1. WCDMA/HSUPA and EGPRS network support
2. Bluetooth 1.2 and Wi-Fi support
3. Digital audio support for mp3 and other formats
4. Support for Linux and other third-party operating systems
5. Java hardware acceleration and support for Java applications
6. Qcamera up to 6.0 mega pixels

All mobile systems character little RAM, low performance CPU, slow internal flash memory, and limited battery power. Therefore, a need was felt for a VM that could provide better performance with restricted resources. So came Dalvik, designed to run on Linux kernel, which provides process threading, pre-processing for faster application execution, User ID based security procedures and inter-process communication. Dalvik works on low resourced ARM devices (Advanced RISC Machines), is 32-bit processor architecture based on reduced instruction set computer developed by ARM limited). ARM processors are used because of their simple architecture making it appropriate for low power devices such as cell phones. Every Android application runs in its own process, with its own instance of the Dalvik virtual machine. Dalvik has been written so that a device can run several VMs efficiently. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The VM is register-based, and runs classes compiled by a Java language compiler that have been changed into the .dex format by the included "dx" tool. The Dalvik VM relies on the Linux kernel for original functionality such as threading and low-level memory management.

5. WORKING OF DVM

The working of Dalvik Virtual Machine is explained with the help of DVM diagram (Fig.4)

- Program is written in java language
- The program is compiled to Bytecode which create the .class file for that program
- “dx” tool converts the .class file into .dex file
- Dalvik VM interpret the .dex format file

6. LICENSE OF DALVIK

Dalvik has licensed its source code under the Apache license, making it attractive and economical for mobile phone carriers, since they can use and adjust it without paying licensing fees. Moreover, the Apache license lets them close any part of the code they want. Comparative to Java machines that are either closed source making them costly or open source.

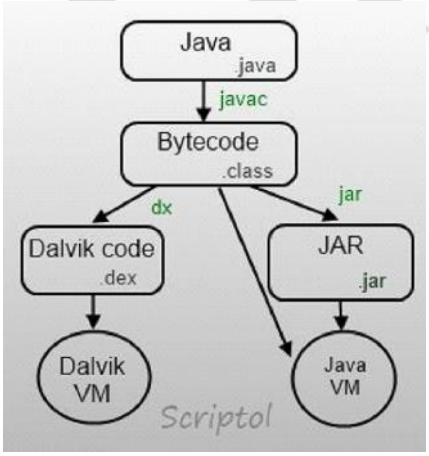

Figure 4
Working of Dalvik Virtual Machine

7. SECURITY ISSUES

Dalvik has an supplementary layer for security, including process separation and file permissions to the underlying Linux platform. Dalvik is comparatively secure because it provides sharing of code between processes without giving it permissions to edit the mutual code. Android’s security architecture ensures no application, by default, has permission to perform any operations that would adversely impact other applications, the operating system, or the user. This includes reading or writing the user's confidential data (such as contacts or e-mails), reading or writing another application's files, performing network access, keeping the device awake, etc These features are implement across the operating system (OS) level and framework level of Android and not within the Dalvik VM. As mentioned previously, each application is run within its own instance of the VM, which itself is running in its own OS process. Each process is assign its own user ID that can only access a limited set of features within the system and can only access its own data. Therefore, an application cannot interfere with other applications. If extra security permissions are required, the application can request them through its Manifest file. These permissions are verified at application installation time by the framework and are compulsory by the OS at runtime. To address memory constraints and allow for fast start-up times, Dalvik shares core, read-only libraries between VM processes. The sharing is done securely by giving the VM processes permission to read the code but not edit it.

8. ARCHITECTURE

Dalvik is a register base architecture making it faster and performance proficient for running application code. It has to operate on Dalvik byte code rather than Java byte code. Register based VM has potential advantages over the stack based VM as discussed. Now we discuss the presently supported functions.

Ashish Yadav et al.
Dalvik - Virtual Machine,
Indian Journal of Engineering, 2012, 1(1), 100-104,
http://www.discovery.org.in/ije.htm

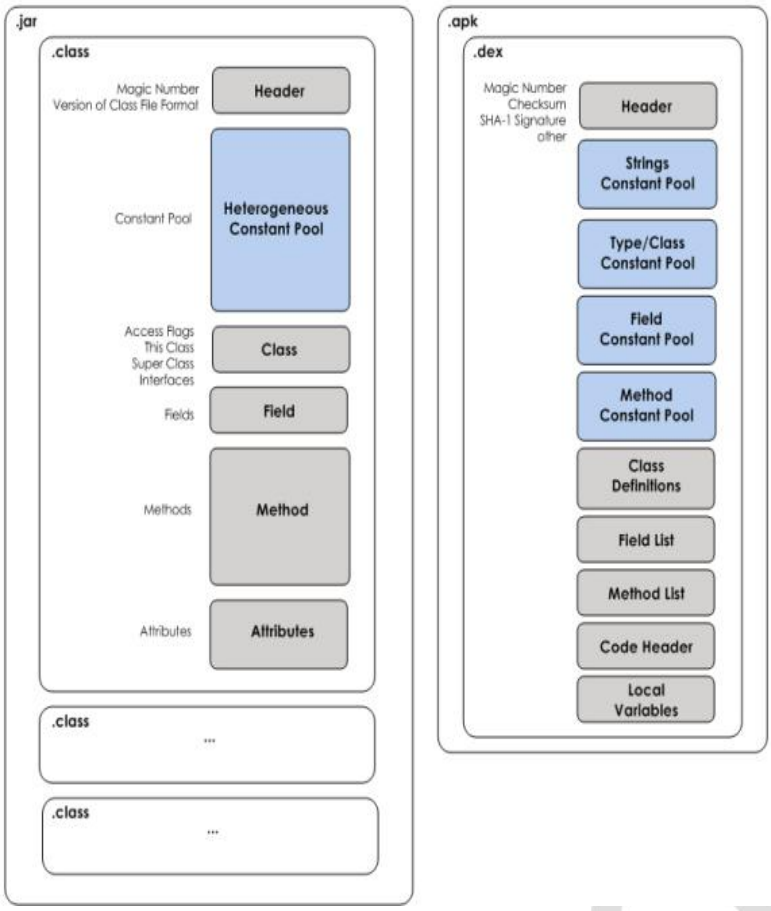

Figure 5

Comparison between .class file format and .dex file format

The supported functionalities are:

→ Dalvik execution file format.
→ Dalvik instruction set
→ J2ME CLDC API
→ Multi-threading

The supported platforms are:

→ Linux
→ BSD
→ Mac OSX
→ Unix

9. DEX. FILE FORMAT

Dalvik Executable is compiled by Dalvik VM, and are zip into a single .apk (Android Package) file on the device. The .dex files can be created automatically by translating compiled applications written in the Java programming language. The .dex file format uses mutual, type-specific constant pools as it's primary mechanism for conserving memory. A constant pool stores all literal constant values used within the class. This includes values such as string constants used in your code as well as field, variable, class, interface, and method names. Rather than store these values throughout the class, they are always referred to by their index in the constant pool. The constant pools in the .class and .dex files are highlighted in blue above. In the case of the .class file, each class has its own private, mixed constant pool. It is mixed in that all types of constants (field, variable, class, etc.) are mixed together. Contrast this to the .dex file that consists many classes, all of which share the same type-specific constants pools. Duplication of constants across .class files is eliminated in the .dex format (Fig.5).

- Java bytecode is stored within .class files
- The "dx" tool is used to convert the .class files into a .dex, or Dalvik Executable file.
- Whereas a .class file contains only one class, a .dex file contains multiple classes
- Then the .dex file is executed on the Dalvik VM
- The .dex file has been optimized for memory usage

10. SPECIFICATIONS

Feature	Minimum Requirement
Chipset	ARM-based
Memory	128 MB RAM; 256 MB Flash External
Storage	Mini or Micro SD
Primary Display	16-bit color or better
Camera	2MP CMOS
USB	Standard mini-B USB interface
Bluetooth	1.2 or 2.0

11. ADVANTAGES

- In DEX all the classes of the application are packed into one file.
- Different pools are created to store strings, constants, variables, field, methods
- It saves the memory via minimal repetitions, implicit labelling
- Verification of byte code is done
- An interpreter is simplified for faster execution
- Duplicate strings and other constants used in multiple class files are included only once in the .dex output to conserve space
- It works on low resourced ARM devices (Advanced RISC Machines)
- It provides process threading, pre-processing for faster application execution
- Able to work with limited resources

12. CONCLUSION

Android is a really open, free development platform based on Linux and open source. Handset maker scan use and modify the platform without paying a royalty A component-based architecture inspired by Internet mash-ups. Parts of one application can be used in another in ways not initially envisioned by the developer. can even replace built-in components with own improved versions. This will unleash a new round of creativity in the mobile space.

- Android is wide open to all: industry, developers and users
- Participating in many of the profitable open source projects
- Aims to be as simple to build for as the web.
- Google Android is stepping into the next level of Mobile Internet

REFERENCES

1. "Device Requirements" Android Source Code <git://android.git.kernel.org/platform/development/pdk/docs/guide/system_requirements.jd>

2. “Dalvik” Android Source Code <git://android.git.kernel.org/platform/development/pdk/docs/guide/dalvik.jd>
3. Lindholm, Tim and Yellin, Frank. The Java Virtual Machine Specification. Second Edition. Sun Microsystems, 1999:
4. Pavone, Michael. “Dex File Format” http://www.retrodev.com/android/dexformat.html
5. Bornstein, Dan. “Dalvik VM Internals” 2008 Google I/O Session Videos and Slides http://sites.google.com/site/io/dalvik-vm-internals
6. Jones, Derek. “Register vs. stack based VMs” http://shape-of-code.coding-guidelines.com/2009/09/register-vs-stackbased-vms/
7. “Dalvik Virtual Machine” < http://www.dalvikvm.com/>
8. “Working of Dalvik Virtual Machine” <http://www.scribd.com/doc/53133105/How-the-Dalvik-Virtual-Machine-Works>
9. “Usage of SecurityManger in Androids Permission-System.” Android Security Discussions <http://groups.google.com/group/android-security-discuss/browse_thread/thread/847e99bfb2276eff>
10. “Architecture” <http://en.wikipedia.org/wiki/Dalvik_(software)>
11. “Overview of DVM” <http://davidehringer.com/software/android/The_Dalvik_Virtual_Machine.pdf>
12. “Specification” <http://imsciences.edu.pk/serg/wp-content/uploads/2009/07/Analysis-of-Dalvik-VM.pdf>
13. “.dex file format” <http://imsciences.edu.pk/serg/wp-content/uploads/2009/07/Analysis-of-Dalvik-VM.pdf>

.