qwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmrtyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnmqwertyuiopasdfghjklzxcvbnm

|  |
| --- |
| The Android OS  Andrew Ribeiro  12/10/2009  CS 450: Assignment 2 |

Introduction & History

The Android operating system is a new operating systems that was developed by a company called Android Inc that was later bought out by google in 2005. Google and the open hanset alience developed the operating system after the accuisition. The main goals of the operating system is to provide a full featured open platform for mobile devices that was robust and reliable (Open Handset Alliance). The first version of the Android operating system was released on November 5, 2007. Initially, the Android operating system was implemented on one phone model by T-Mobile called the G-Phone. However after the success and promise of the G-Phone proved itself in the market, other industry players such as sprint and version started making phones with the Android OS. The popularity of the operating system is growing tremendously because of its open source nature, reliability, and robustness. Figure 1.0 shows how fast the Android OS popularity is growing. Currently, the Android OS is the one of the only open source operating system on the market carried by large industry partners—although Symbian OS is moving to open source (Symbian OS Open Source).

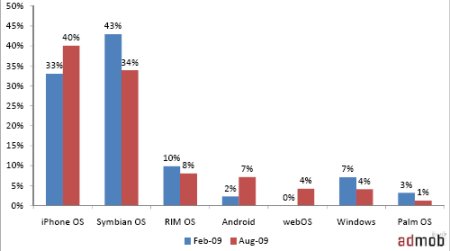


Figure 1.0

The Platform

The Android OS provides a powerful computing environment for mobile devices. It consists of four systems, the application framework, core libraries, the Android runtime, and the Linux Kernel. The application framework provides application access to features that make the application useful; for example, the application framework allows applications to notify the user of application events via the Notification Manager in the application framework. Another system that is tied directly to the application framework is the core libraries. The core libraries are native C++ code that provides all functionality to the application framework; an example of a component in the core libraries is the Media Libraries, which provide support for the playback of various media formats on the device. It is an important point that across several phones/devices with different architectures, the application framework will remain the same; however, the core libraries and kernel will change for every phone with a different architecture, because the core libraries and the kernel contain device specific drivers and features. The Android runtime is the heart of the OS; it creates a bridge between the applications and the kernel using the Dalvik Virtual Machine, which is essentially a java virtual machine. Each application runs in its own Linux process/thread with its own instance of the Dalvik virtual machine to ensure errors from one application do not propagate over into other applications. Finally, the Linux kernel provides all of the low level OS functionality, such as security, memory management, process management, network stack, and the driver model. Figure 1.1 shows these systems based on their reliance on other systems, i.e. the top system requires all of the systems beneath it. Due to the size restrictions of this paper, the linix kernal and a little bit of the android system runtime will be covered.



Figure 1.1 The Android OS (Android Basics)

The Linux Kernel & Android

The kernal of the Android operating system is a custom distrubution of Linux version 2.6 (Android Basics). The Linux kernel consists of various subsystems, including process management, system call interface, virtual file system, memory management, network stack, architecture dependent code, and device drivers. All of the subsystems of the Linux kernel ultimately provide all of the system functionality to the Dalvik virtual machine, which then passes that functionality onto the applications that are executed by the Dalvik virtual machine. Bellow you will find an overview of these Linux subsystems and other functionality of Linux such as security and protection. Figure 2.1 shows the various components of the Linux kernel.

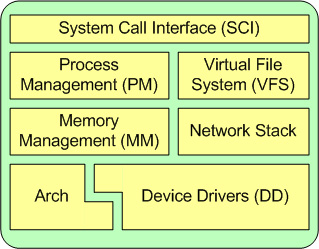


Figure 2.1 Linux Subsystems (Linux Kernel IBM analysis)

## System Call Interface (SCI)

The system call interface provides a way for applications in the user space to call kernel functionality; for example, if a user application needs to write a file to the disk, it must make a system call to do so, because the file management system falls within the domain of the kernel. In terms of the Android OS, a user application never directly makes a system call; the Dalvik virtual machine that is interpreting the application does instead, because Android applications are not natively run on the device.

## Process Management (PM)

Process management deals with the execution of threads/processes; these are the fundamental units of work in the Linux OS. In addition, it is important to point out that a thread and a process are considered the same thing in Linux. Each Linux thread consists of four parts: thread code, data, stack, and CPU registers. The thread code consists of the executable code contained in the process. The data consists of specific information that the thread requires during its execution. The stack contains variables and other constructs that are allocated during the runtime of the thread. The CPU registers refer to which registers in the CPU the thread requires access.

* Real-time task: OS task
* Usertask: user task
* Expired runque: time slice expired, priority and time slice recalculated when placed in the expired queue
* Switches with active queue if given priority is empty
* Active runque: gueue that is being processed.

The functionality of the process management system--creating threads, killing threads, etc—is exposed through the System Call Interface (SCI). In respect to Android, if an application wishes to create a new thread, the Dalvik virtual machine makes the system call to the SCI rather than the application itself. The process scheduling algorithm used by Linux is called the O (1) scheduler (Linux Process Scheduler). Each CPU in the system has two queues: a wait queue and a run queue. The wait queue is for processes that have elapsed their burst time. The run queue is the processes that need to be processed. Each of the queues are labeled 1-140, corresponding to the priority of the thread in that slot. If there is a thread with a higher priority than the rest, it will always be run before processing the lower level threads. If a slot in the run queue is empty and the corresponding spot in the wait queue is full, the pointers are swapped, so that the thread in the wait queue is moved to the run queue. The O (1) scheduler means that the scheduling of many threads is the same as scheduling one thread. The Linux kernel also supports Symmetric Multiprocessing (SMP), which allows the utilization of multiple processors.

## Virtual File System (VFS)

Linux provides a common interface for file I/O via a Virtual File System. The VFS has three components: the VFS API, device drivers, and physical devices. Linux exposes file system operations through the VFS API, which contains commands like close, read, and write. The device drivers are hardware dependant for the physical devices. The physical devices are the actual disks that store information.

## Memory Management (MM)

Linux uses a virtual memory scheme called slab allocation. Linux has a base buffer size for a page that is usually 4KB, “but provides abstractions over these 4KB buffers, such as the slab allocator. This memory management scheme uses 4KB buffers as its base, but then allocates structures from within, keeping track of which pages are full, partially used, and empty. This allows the scheme to dynamically grow and shrink based on the needs of the greater system.” (Linux Kernel IBM Analysis).

In the context of Android, the Linux memory management system is only responsible for the management of Dalvik virtual machines. Within the Android runtime, the Dalvik virtual machine takes care of memory management for each program, this includes the standard Java garbage collection cycle.

## Network Stack (NS)

The network stack consists of seven layers: application layer, system call interface, protocol agnostic interface, network protocols, device agnostic interface, device drivers, and physical device hardware. The application layer is in the user space and consists of the applications that require network functionality. The system call interface allows user space applications to make system calls to the networking platform. The protocol agnostic interface is considered the sockets level, which provides access to various communication channels regardless of protocol, meaning all sockets have a standard structure; for example, the sockets layer can establish sockets with “TCP and UDP protocols, but also IP, raw Ethernet, and other transport protocols, such as Stream Control Transmission Protocol (SCTP)” (Linux Network Stack). The network protocols layer establishes how the sockets are going to be used to exchange data over the network. The device agnostic interface “provides a common set of functions to be used by lower-level network device drivers to allow them to operate with the higher-level protocol stack” (Linux Network Stack). Device drivers contain specific code that enables the computer to establish I/O operations with the device. Finally, we have the physical device layer, which contains the actual networking hardware—i.e. a network interface card (NIC). Figure 2.2 shows the network stack.

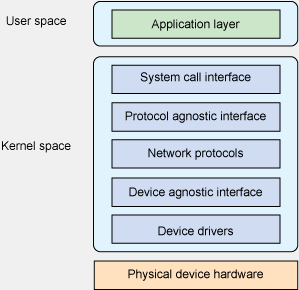


Figure 2.2 (Linux Network Stack)

## Arch

The Arch portion of the Linux kernel refers to code that links primary hardware like the processor to software. Arch differs from devices drivers, in that the device drivers can be added on to the system, but Arch code is platform specific.

## Device Drivers (DD)

Device drivers allow the Linux kernel to interact with external devices. In the context of the Android OS, a common device driver to have is a camera, so applications written that utilize a camera can run on phones that have camera hardware and the respective device drivers. External device functionality is accessed via the SCI.

## Protection and Security

The Linux kernel provides the basic standard OS protection rules such as thread safety and memory access boundaries. The Android runtime also adds a layer of protection to the system because it only allows manage code to be run on the system rather than native code.

Conclusion

The future for the Android operating system is bright. The Android operating system provides a robust, reliable, and open platform that rivals general computer operating systems in their functionality. The Android operating system is very new, but has grown at an extremely fast rate and has proven itself in the market. In addition, the Android operating system has been released even on some laptops due to the robustness of the operating system (Android Laptop).There are many upsides to the Android operating system; however there is one main problem, the Dalvik virtual machine garbage collection (Dalvik Problem). When there are many processes running at the same time, the garbage collection is sometimes inefficient, which leads to slow response time. In the end, the Android operating system is a great mobile operating system that rivals the top mobile operating systems in robustness and reliability in today’s market. Figure 3.0 shows the android logo.

**

Figure 3.0( Android Logo )

References

*Android Developer*

<http://developer.android.com/>

*Linux Kernel IBM Analysis*

<http://www.ibm.com/developerworks/linux/library/l-linux-kernel/>

*Open Handset Alliance*

<http://www.openhandsetalliance.com/press_110507.html>

*Symbian OS Open Source*

<http://www.symbian.org/symbian-feature-set/going-open-source>

*Android Basics*

<http://developer.android.com/guide/basics/what-is-android.html>

*Ad Mob*

<http://www.admob.com/>

*Linux Process Scheduler*

<http://www.ibm.com/developerworks/linux/library/l-scheduler/>

*Linux Network Stack*

<http://ldn.linuxfoundation.org/node/22518>

*Android Laptop*

<http://www.youtube.com/watch?v=SXq__YWVAew>

*Android Logo*

<http://www.topnews.in/arm-launches-solution-center-android-2237255>

*Dalvak Problem*

<http://android-developers.blogspot.com/2009/02/track-memory-allocations.html>