Presentation Writeup: Group Android
Ted Airey, Matt Fein, and Jason Georgis
3-13-18

Introduction and Main Goals - Matt and Ted

The Android OS has revolutionized how mobile operating systems work as well as providing necessary competition against the IOS operating system. In 2003 Andy Rubin began work on an open source OS originally intended to ease pain points with digital cameras. With the growth of the smartphone market, Blackberrys and Palm devices, the Android project pivoted towards mobile devices. Google acquired the platform in 2005. After Google's acquisition the platform decided to utilize the linux kernel as a basis for the platform. This decision forced the OS to be open source and be freely available to all manufacturers. Android's open source nature allowed manufacturers to hop on board immediately. In 2008 HTC released the first Android device with the G1. Since then Android has gone through 12 releases, all dessert themed: Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jellybean, Kitkat, Lollipop, Marshmallow, and finally Oreo have all improved on the Android OS and allowed developers and manufacturers more features to work with. Today there are over 2 billion active devices.

The three main goals of the Android OS as listed on the official website are as follows: "Enchant Me" - the OS intends to create a dynamic environment for the user, personalizing the device to suit the user's needs, and allowing the user to manipulate basic functions to make it more user-friendly. "Simplify My Life" - the OS aims to put only the most necessary items, apps, and notifications at the forefront of the user interface, and to use pictures instead of words to represent objects, allowing for a more user-friendly display. "Make me amazing" - the OS strives to use cutting edge techniques that allow the user to easily operate complex functions. Using constant feedback from the OS to help guide processes helps make the user feel confident and in control.

OS Structure - Jason

Once you actually look under the hood of the Android OS, there's quite a bit to unpack. The OS is comprised of 5 components that are organized in layers of decreasing abstraction. The first layer is known as the Application Framework which is what allows developers to create their own applications and is done on the code-level, utilizing very little hardware. Because the Android OS behaves like a Linux Environment, each application runs on its own Virtual Machine in order to keep different applications from infecting one another. One key factor of the Application Framework, is the Principle of Least Privilege on which it works. How this functions is if an application needs to run but the memory is currently full, the oldest application that is currently running in the

background that doesn't need to be, is quit. This principle is important for the OS security and keeps each application from accessing something it shouldn't.

The next layer in the OS, called the Binder Inter Process Communication (Binder IPC for short), is an interface that is responsible for communication between applications. When a user needs two applications, or more often, processes, to communicate they utilize the Binder IPC. Allowing inter process communication, helps the OS run multiple processes at once which is essential for concurrency.

Delving deeper into the Android OS structure, we arrive at the third layer, System Services. A service is an application component that performs long running operations in the background, and can be classified as either started or bound. A started service is one that is called by a different application component and can run indefinitely until stopped by the system. A bound service is one that can interface with the component that launched it (client); when a bound service is launched, the client binds itself to the service, allowing for the service to return data back to the client.

The fourth level of the OS structure is known as the Hardware Abstraction Layer or HAL. Interestingly, this layer isn't at all touched by the application developers, and is solely for the use of the device vendors, thus each device has a unique HAL. However, all HALs consist of two components, the module and device. The module is simply a shared library containing metadata while the device is the actual hardware making up the device. Overall, the purpose of the HAL is to allow more functionality in the OS without actually modifying the system.

The fifth and final layer of Android's OS is the Linux kernel on which the system is built. Like all operating systems, the kernel is the core of the system and has complete control over the entire OS. The Android Linux kernel is quite similar to the normal open-source Linux kernel, however it is slightly more advanced. Since the Android OS has to work on a myriad of devices, many of which are mobile, it needs to incorporate different features such as memory and battery management. These five, layered components of the Android OS provide the structure for the entire system.

User Interfaces - Matt

Core to the mobile experience is the UI, or user interface, elements that users interact with each time their device is used. As Android began to mature it found itself deeply lacking in its UX design compared to its IOS competitor. Android had prioritized other technical OS features over a smooth and pretty user experience. This all changed with the release of Lollipop and Google's introduction of Material Design guidelines. Material Design is now Google's accepted design format across all of its products. The release of Material Design gave developers access to animations and design elements, such as the side swipe nav bar. Access to these elements created a smoother, unified, and overall more enjoyable experience for the end user.

Boot Processes - Matt

When the user presses the power button boot ROM code executes from a hard coded location. This code loads the bootloader into RAM, the bootloader is not an Android program but a seperate application in itself. The Android kernel is then launched as a result of the executed bootloader code. The kernel sets the cache and schedules load drivers. The kernel then looks for init in the system files. The init process mounts the file directories /sys and /dev which are critical to running the OS. Through the init process, the first process begins. All other system processes fork off of this first process.

Thread Processes - Ted

When an application component is pushed to the processor with no other components in the app running, the android system starts a new linux process. All components within that same application run through the same “main” thread, so if a new component is started within that app, it runs through the same thread. You can manipulate it so that the components use different threads, but this is not the default method. The thread the application uses is almost always the same thread that interacts with components from the android UI toolkit.

Communication - Jason

Communication is crucial to every OS, since processes must communicate with one another, with the user, and with the system. Android utilizes the Interprocess Communications discussed earlier to accomplish this. The IPC functions by using remote procedure calls (RPCs), where a method is called by an activity or application component, and then executed remotely, with a return being given to the caller. The method being called and the data its passed are decomposed to a level the OS can understand and transmitted from the local process and address space to the remote process and new address space. Here the data is reassembled and the designated action is executed. Any return data is similarly broken down to basic level for the OS to transmit, and given back to the calling activity. In this manner, the processes communicate with one another, allowing complexities such as concurrency.

CPU Scheduling - Matt

Android is based off the Linux kernel it utilizes Linux’s scheduling processes. Processes are given an initial priority between 19 to -20, very low to very high. The priority ensures that higher priority processes will get more CPU time when needed. These levels are also dynamic, so a process can be moved up or down in priority level. Thread priorities are chosen inverse to the amount of work a thread is expected to do. The more work a thread does, the less favorable its thread priority should be. User interface threads are given default priority while Async threads (background server code) are given background priority. Android also utilizes the Linux cgroup (control group) process.

Through cgroup background processes are given background priority so that the user does not notice that high amount of processing power are being utilized. Android also ensures that threads belonging to applications not in the foreground are moved to the background level as well.

Memory Paging, Virtual Memory, and Memory Mapping - All

The android runtime or ART uses paging to manage memory. Any memory that an app uses or accesses remains in random access memory and cannot be paged out. The only way to release memory from an app is to dereference the objects held in memory, and that memory is then collected with the garbage collector. It does not need to be freed by the user; the garbage collector can detect when a piece of memory is no longer being used, and frees it back to the heap. Android's memory heap is generational, so when a piece of memory is used often enough, it can move up a generation so it has a less likely chance of being freed or overwritten.

Android OS uses virtual memory more efficiently than other OS's by making each application choose which data it would like saved to the storage rather than save all of its data. This frees up RAM space when an application isn't being used, but also saves vital data. It saves the data using the typical virtual memory mapping method. Every process started by an application is forked from an existing process called Zygote. Zygote starts when the system boots and loads common framework code and resources, and when users switch between apps Android keeps apps that are not visible in a least recently used cache. When a user leaves an app, the process does not quit, and when or if the user returns to the app, the system will reuse the process making the app switching process faster.

Storage and File System - Matt

Android utilizes the StorageManager interface to the systems storage service. The StorageManager handles storage related interfaces such as Opaque Binary Blobs (OBBs). OBBs contain a filesystem that can be encrypted and mounted on demand from an application. OBBs are not stored within the application itself, but rather stored in a shared pool accessible to all programs. Therefore using OBBs is not secure.

The Android file system is divided into two parts, internal and external. External is not as common as modern devices have moved away from memory card slots. Regardless of internal or external, the entire system has a single root known as /root. This is a feature of Linux. Android allows manufacturers to use different file systems as they choose. For example Samsung and Motorola utilize different systems. YAFFS (Yet another Flash File System) is commonly used as it is an open source file system designed to be fast and suitable for almost any device.

Particular Features - Jason

Much like other operating systems, Android rolls out a new and updated OS every few months with improvements and fixes to keep their technology and devices both functional and relevant. The current release, designated the Oreo, has a multitude of handy features that set it apart from other operating systems. The first feature we'll explore the ability to automate virtually any setting changes one does throughout the day. This app allows you to set your phone to disconnect from wifi when not at home, or adjust the screen brightness based on the time of day, and has a plethora of other uses, each unique to the user's needs. Another feature that's turning a lot of heads is the new multitasking abilities. This feature allows the user to run "picture in picture" displays where a video can be played in a smaller window while different apps are run such as surfing the web or texting. Although this feature was available on the previous release of the Android OS, it has yet to released by their main competitor, Apple, making it valuable asset. The Oreo release also has improved on both battery and memory conservation by limiting what apps can do while running in the background, as well as introduced new and improved security systems. Although Android has many more features, these are the ones that truly stand out and set the Android OS apart from its competition.

Sources:

1. https://www.educba.com/structure-of-an-android-operating-system/
2. https://developer.android.com/guide/components/processes-and-threads.html
3. https://developer.android.com/topic/performance/memory-overview.html
4. https://developer.android.com/training/data-storage/files.html
5. https://android.stackexchange.com/questions/62452/android-virtual-memory-support
6. http://www.telegraph.co.uk/technology/0/android-oreo-best-features-googles-new-operating-system-download/
7. https://www.cnet.com/pictures/android-oreo-best-new-features/16/
8. https://developer.android.com/reference/android/os/storage/StorageManager.html
9. https://developer.android.com/topic/performance/memory-overview.html
10. https://www.androiddesignpatterns.com/2014/01/thread-scheduling-in-android.html
11. https://hackernoon.com/the-android-boot-process-2ce4c498615b