CS 411 Operating Systems II Final Exam

Analysis of the Android x86 Build Tree

Due 07:30 Friday, 13 June 2014

Background

As all of you know, the Android mobile OS, developed by Google (and later, the Open Handset Alliance), has been one of the top smartphone OS solutions for a handful of years. One of the nice things about Android is that it is a mostly open mobile phone platform. One other project that has stemmed from Android proper is the Android x86 project. What this project aims to do is bring a robust version of Android to x86 systems. The Android x86 project is an Open Source project and can thus be pulled from their website to observe, analyze and modify.

The Android Software Stack

Contrary to what some believe, Android is not run purely on Java (the very notion is beyond silly). The creators of Android have described it as a software stack. There are three basic components that make Android what it is. On the lowest level, we have the Operating System (You should know what this layer does), the middleware level, which is where all of the necessary libraries live, along with a component called the Android runtime. Within the Android runtime, we have something called Dalvik. Dalvik is Android's custom Java Virtual machine. After that, we have the Application Framework Layer. This holds all of the frameworks and libraries that are directly available to Android App developers. And above that layer, of course, we have the Android Applications themselves.

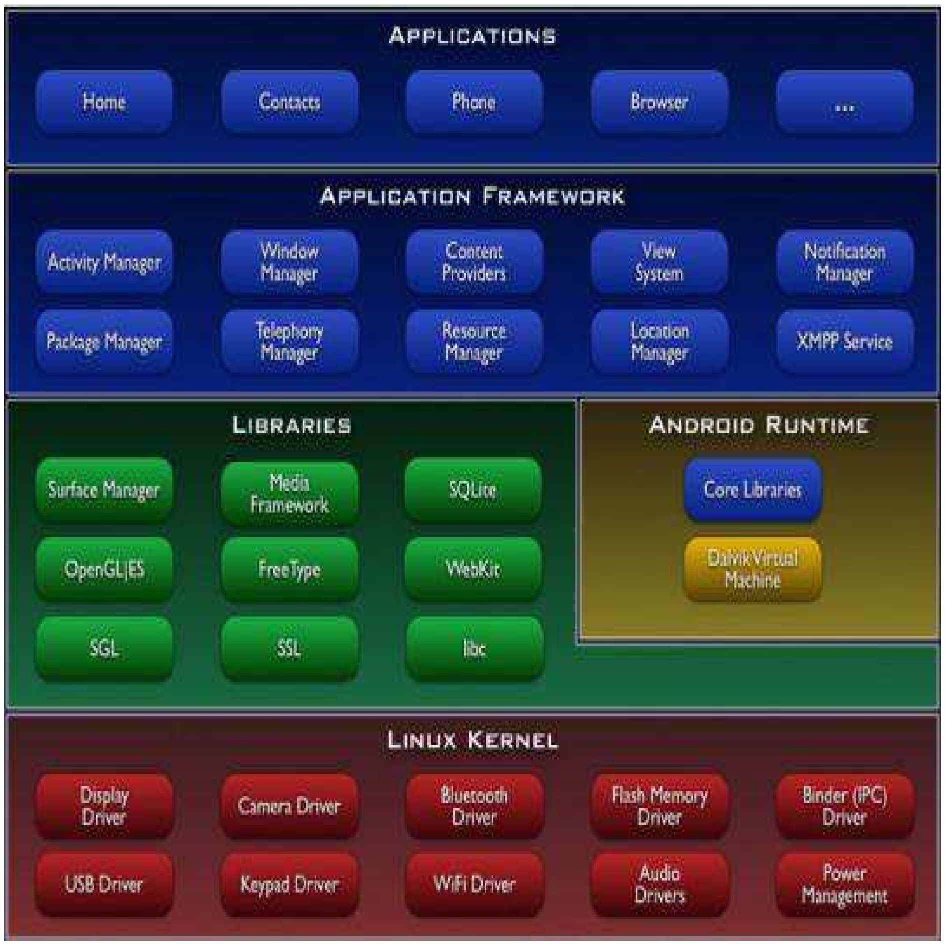

Figure 1: Android Software Stack

The Exam

If you look at the bottom level of the Software Stack diagram, you'll see that Android is built atop a Linux foundation. This makes Android a wonderfully practical example of the concepts you have been learning throughout this term. It is my personal belief that examining how the Linux kernel is implemented for Android is not only fascinating, but also practical, and it will help drive home the key concepts that you have been learning during your time in CS 411.

Here is what we want you to do: Look back at your previous assignments, choose 2 key concepts (plus another for extra credit) and briefly describe what your implementations for those to projects were. Then, dig into the android source (provided on `os-class` – `/scratch/android/` (this is just a directory, not a repo)) and find where that concept is implemented, read through the code to figure out how the Android developers implemented that portion of the Android-specific kernel, and compare and contrast implementation algorithms between the Android version and your implementation.

To be more clear, write a bit about how you implemented the 2 or 3 concepts of your choice, then write about how Android either does things differently, or how they are the same. Finally, write about why you think the Android Developers chose the implementation that they did – hardware capabilities, restrictions, etc. There should be, at the very least, one page of writing for each concept. You are to write your paper in LaTeX and submit it just as you would have submitted a write-up. I'm very excited to see what you guys find and read your thoughts on why such implementation decisions were made!

Why the heck am I doing this?

Again, it is so important for a student to see the concepts they learn applied in industry, and it just so happens that we have a very unique and popular source tree at our disposal for you to dig into. We want to help answer the "when am I ever going to use/need to know this?" question.

You may very well never write another process scheduler in your entire life, but knowing how it works, and having a solid understanding of what's going on under the hood will make you a much better Computer Scientist. Plus, the word "science" is in the name of your field of study. What kind of scientists would we be if we never got our hands dirty and observed and examined? Have fun with this exam; it's meant to prove to you that you know your stuff! You know real-world concepts implemented in real-world solutions. It really can't get much cooler than that (unless if you invent something; that's pretty awesome).

Addendum: Group evaluation

Please also include an evaluation of each of your partners, including contributions, ability to work well in a group, etc.

The more detail and support you can provide, the more weight will be given to your evalution.

Many thanks to Bryan Pawlowski for providing the concept and initial write-up.