The Operation and Application of Android System

I. INTRODUCTION

Android is a software stack for mobile devices which includes an operating system, middleware and key applications. Since its official public release, Android has captured the interest from companies, developers and the general audience. From that time up to now, this software platform has been constantly improved either in terms of features or supported hardware and, at the same time, extended to new types of devices different from the originally intended mobile ones. Google entered into the mobile market not as a handset manufacturer, but by launching mobile platform called as “Android” for mobile devices such as Smart phones, PDA and net books on 5th November 2007. Google has a vision that Android based cell phone will have all the functions available in the latest PC. In order to make this effort possible, Google launched the Open Handset Alliance. Google introduced Android as an OS which runs the powerful applications and gives the users a choice to select their applications and their carriers. The Android platform is made by keeping in mind various sets of users who can use the available capacity within Android at different levels. Android is gaining strength both in the mobile industry and in other industries with different hardware architectures.

The increasing interest from the industry arises from two core aspects: its open-source nature and its architectural model. Being an open-source project, Android allows us to fully analyze and understand it, which enables feature comprehension, bug fixing, further improvements regarding new functionalities and finally, porting to new hardware. On the other hand, its Linux kernel-based architecture model also adds the use of Linux to the mobile industry, allowing to take advantage of the knowledge and features offered by Linux. The Android platform consists of several layers which provide a complete software stack.

Android applications are Java-based and this factor entails the use of a virtual machine VM environment, with its advantages. Android uses its own VM called Dalvik, which interprets and executes portable Java-style byte code after transforming it, which is optimized to operate on the mobile platform. All of these aspects make Android an appealing target to be used in other type of environments.

The remainder of this paper is organized as follows: Section II briefly describes the Android’s background including architecture, features & programming framework. Section III presents detailed analysis of Android market including comparison with Symbian & Windows Mobile. Finally Section IV concludes this paper.

II. ANDROID BACKGROUND

A . Android Architecture

Android Architecture which consist of number of layers as Applications, Application framework, Libraries, Android runtime & Linux kernel [1]. Application layer is the uppermost layer which provides a set of core applications including an email, SMS program, calendar, maps, browser, contacts, and others. All applications are written using the Java programming language. It should be mentioned that applications can be run simultaneously; it is possible to hear music and read an email at the same time. The Application Framework is a software framework that is used to implement a standard structure of an application for a specific operating system. With the help of managers, content providers and other services programmers it can reassemble functions used by other existing applications.

Layer which is present below Application framework consist of two parts as Libraries which are all written in C/C++. They will be called through a Java interface. This includes the Surface Manager, 2D and 3D graphics, Media Codecs like MPEG-4 and MP3, the SQL database SQLite and the web browser engine WebKit. Second part is Android Runtime which includes a set 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. The Dalvik VM executes files in the Dalvik Executable (.dex) format which is optimized for minimal memory footprint. The lowest layer is Linux Kernel, Android basically relies on Linux version 2.6 for core system services such as security, memory management, process management, network stack, and driver model. The kernel also acts as an abstraction layer between the hardware and the rest of the software stack.

B. Features of Android

Google Android has many features which make it special, but one important feature is Dalvik virtual machine (DVM) [5]. Which is a major component of Android platform. It is optimized for low memory requirements and is designed to allow multiple VM instances to run at the same time. The DVM runs Java applications. However, it is different from standard Java virtual machine in some ways. First, most virtual machines use a stack-based architecture, but Dalvik is a register-based architecture. Second, Dalvik runs Java applications which have been transformed into the Dalvik Executable (.dex) format which is optimized for minimal memory footprint The Dalvik VM relies on the Linux kernel for underlying functionality such as threading and low-level memory management. Java virtual machine tool interface (JVM TI) is a native programming interface on Java virtual machine. The interface provides functionalities to inspect the state of a virtual machine, gather information during run time, and also control the execution of applications running on the Java virtual machine. Android has built in integrated browser based on the open source WebKit engine & built in powerful SQL database engine called SQLite, use for structured data storage. Android support for common audio, video, and still image formats such as AAC, MPEG4, H.264, MP3, AMR, & contains Rich development environment including a device emulator, tools for debugging, & a plug-in for the Eclipse.

C. Android Programming Framework

The environment requires to develop application for Android consists of the Android SDK, the Eclipse IDE and the Java Development Kit (JDK) which has to be preinstalled for the installation of both, Android SDK and Eclipse.

1) Android Software Development Kit: The Android SDK includes a comprehensive set of development tools. These include libraries, a handset emulator, documentation, sample code, tutorials & tools such as dx - Dalvik Cross-Assembler, aapt – Android Asset Packaging Tool & adb– Android Debug Bridge. Applications are written using the Java programming language and run on Dalvik, a custom virtual machine designed for embedded use which runs on top of a Linux kernel. The officially supported integrated development environment (IDE) is Eclipse (3.2 or later)

2) Android Emulator: The Android SDK includes a mobile device emulator -- a virtual mobile device that runs on your computer. The emulator lets you prototype, develop, and test Android applications without using a physical device. The Android emulator mimics all of the hardware and software features of a typical mobile device, except that it cannot receive or place actual phone calls. It provides a variety of navigation and control keys, which you can "press" using your mouse or keyboard to generate events for your application. It also provides a screen in which your application is displayed, together with any other Android applications running. To let you model and test your application more easily, the emulator supports Android Virtual Device (AVD) configurations. AVDs let you specify the Android platform that you want to run on the emulator, as well as the hardware options and emulator skin files that you want to use.

III. ANDROID MARKET ANALYSIS

A. Android Market

The Android Market, an online software store, is developed by Google for Android devices. It was made available to users on October 22, 2008. Most of the Android devices come with preinstalled “Market” application which allows users to browse, buy, download, and rate different available applications and other content for mobile phones equipped with the open-source operating system. Unlike with the iPhone App Store, there is no requirement that Android apps should be acquired from Android Market [2]. Android apps may be obtained from any source including a developer's own website. Also, Android developers can create their own application market. Google does not have a strict requirement for the application to show up on the Android Market compared to the process used by Apple. Lastly, the Android Market follows a 70/30 revenue-sharing model for applications developed by developers. The developers of priced applications receive 70% of the application price and remaining 30% distributes. As of May 04, 2010, Android apps hit around 49,000 applications which were around 12,500 in August 2009 and 20,000 in December 2009.

B. Android vs. Symbian vs. Windows Mobile

Portability: Portability is a very important assessment criterion. Symbian OS has many references in this area and having standardized architecture and the openness to software. But the fact that Symbian mostly runs on Nokia cell phones and that it is not Java based lets it fall behind Android. Unfortunately Windows Mobile also has several applications that are specific to certain hardware platforms and therefore are not portable. The Android Mobile platform is a Linux & Java based which allow us to use it on many different platforms unlike Symbian & Win Mobile. As a result Android gets one point, Symbian OS gets half a point and Windows Mobile zero points.

IV Creating Your First Android Project in Eclipse

To start your first Android project, open Eclipse. When you open Eclipse for the first time, it opens to an empty development environment, which is where you want to begin. Your first task is to set up and name the workspace for your application. Choose File | New | Android Project, which will launch the New Android Project wizard.

The New Android Project wizard creates two things for you:

A shell application that ties into the Android SDK, using the android.jar file, and ties the project into the Android Emulator. This allows you to code using all of the Android libraries and packages, and also lets you debug your applications in the proper environment.

Your first shell files for the new project. These shell files contain some of the vital application blocks upon which you will be building your programs. In much the same way as creating a Microsoft .NET application in Visual Studio generates some Windows-created program code in your files, using the Android Project wizard in Eclipse generates your initial program files and some Android-created code.

In addition, the New Android Project wizard contains a few options, shown next, that you must set to initiate your Android project.

For the Project Name field, for purposes of this example, use the title HelloWorldText. This name sufficiently distinguishes this Hello World! project from the others that you will be creating in this chapter.

In the Contents area, keep the default selections: the Create New Project in Workspace radio button should be selected and the Use Default Location check box should be checked. This will allow Eclipse to create your project in your default workspace directory. The advantage of keeping the default options is that your projects are kept in a central location, which makes ordering, managing, and finding these projects quite easy. For example, if you are working in a Unix-based environment, this path points to your $HOME directory.

If you are working in a Microsoft Windows environment, the workspace path will be C:/Users/<username>/workspace.

However, for any number of reasons, you may want to uncheck the Use Default Location check box and select a different location for your project. One reason you may want to specify a different location here is simply if you want to choose a location for this specific project that is separate from other Android projects. For example, you may want to keep the projects that you create in this book in a different location from projects that you create in the future on your own. If so, simply override the Location option to specify your own custom location directory for this project.

译文

安卓系统的操作与应用

1引言

安卓是一个包括操作系统和关键应用程序的移动软件堆栈设备。自谷歌官方公开发布该系统以来，安卓吸引了公司、开发人员和大众的兴趣。从那时到现在，这一软件平台不断的得到改善，支持更多的硬件，同时扩展到许多新类型的设备上。从此，谷歌进入了手机市场而不是作为一个手机制造商，不过谷歌于2007年11月5日推出自己的“Android”的智能手机、PDA等移动设备。谷歌的愿景是安装Android系统的手机将能够拥有与最新的电脑一样的所有功能。为了使这一努力变成可能，谷歌推出了开放手机联盟。谷歌推出安卓作为操作系统，以其强大的应用程序和运行能力，使用户可以选择他们的应用和手机设备。

主要是基于两个核心方面:它的开源性质和它的架构模型。安卓是一个开放的开源项目，Android完全允许我们分析和理解它，以使功能进一步增强，以及修复系统存在的故障，然后基于新功能进行改进后，移植到新的硬件中。另一方面，它的基于Linux内核的体系结构模型，允许利用Linux提供的知识和特性，因此，增加了Linux在手机行业中的使用。Android平台是一个几层完整的软件堆栈。Android应用程序是基于java代码，需要使用在虚拟机环境下来使用的，这是其优势。Android使用自己称为达尔维克的虚拟机和便携式java风格的字节代码来执行的，这是在移动平台上的一个优化操作。所有这些方面都使得Android成为一个有新引力的平台，可以在不同类型的环境下得到使用。

本文的其余章节主要是：第二部分简要描述了Android的后台、架构、功能和编程框架。第三节则介绍了Android的详细市场分析，包括与塞班和Windows Mobile的比较。最后，第四部分这篇文章进行了总结。

2 安卓的后台

A 安卓的架构

安卓系统架构，由应用程序、应用程序框架、库文件、Android运行环境和Linux内核组成。应用程序位于最上一层，提供了一组核心应用程序，包括电子邮件，短信程序，日历，地图，浏览器，联系人和其他。所有的应用程序都用Java语言编写。应该提及的是，应用程序可以同时运行。可以在同一时间听音乐和读电子邮件。应用程序框架是一个软件框架，用于在特定的操作系统中实现应用程序的标准结构。在管理者的帮助下内容提供者和其他服务程序可以重新组装其他应用程序现有的功能。在应用程序框架层之下的那一层由两部分完全由C/C++编写的库组成。它们通过Java接口被调用。这包括界面管理器，2D和3D图形，媒体编解码器如MPEG-4和MP3，SQL数据库SQLite和web浏览器引擎的WebKit。第二部分是Android Runtime，其中包括了一套可以提供大部分Java编程语言功能的核心库。每一个Android应用程序都在自己的进程中运行，有自己的Dalvik虚拟机实例。Dalvik虚拟机执行的Dalvik可执行文件（.dex）是被优化过的占用最少的内存空间。最底层为Linux内核，安卓的核心系统服务主要依赖于2.6版本的Linux内核，如安全、内存管理、进程管理、网络协议栈和驱动模型。内核也通常在硬件和软件堆栈之间作为一个抽象层。

B安卓的前景

谷歌的安卓有许多使它很特别的特点，其中最重要的一个特征是Dalvik虚拟机（DVM）。它是安卓平台的重要组成部分。它被优化到仅占有最少的内存，并被设计为允许在同一时间运行多个虚拟机实例。DVM运行Java程序。然而，它在某些方面和标准的Java虚拟机并不相同，首先，大多数虚拟机使用一个基于栈的架构，但Dalvik是一个基于寄存器的架构。第二，Dalvik的运行Java程序已被优化成最小的内存占用，Dalvik虚拟机优化依赖于Linux内核底层的功能，例如线程和低级别的内存管理。Java虚拟机工具接口（JVM TI）是Java虚拟机上的本地编程接口。该接口提供了在运行期间可以检查虚拟机的状态和收集信息的功能，并且可以控制正在虚拟机中运行的程序。Android已经内置了基于WebKit引擎的浏览器和被称为SQLite的强大的数据库，使用结构化的数据存储。安卓支持常见的音频，视频静态图像，格式包括：AAC，MPEG4，H.264，MP3，AMR以及含有丰富的开发环境包的设备模拟器，调试工具，和针对Eclipse的插件。

C安卓应用程序框架

开发安卓应用的环境包含安卓SDK，Eclipse IDE以及Java开发工具包(JDK)，并且需要预先安装安卓SDk和Eclipse。

1）Android软件开发工具包：Android SDK包含了一套完整的开发工具。这些包括库，手机模拟器，文档，示例代码，教程和工具，如DX - Dalvik交叉汇编教程及工具，AAPT–Android资源打包工具和adb–Android调试桥。应用程序是用java语言编写并运行在Dalvik虚拟机，一种专为嵌入式应用而设计的虚拟机。官方支持的集成开发环境（IDE）是Eclipse（3.2或以后）。

2）Android模拟器：Android SDK包含一个移动设备模拟器，一个在您电脑上运行的移动设备模拟器。模拟器可以让您的原型，开发和测试安卓程序不需要物理设备。安卓模拟器模仿一个典型的移动设备所有的硬件和软件功能，但它不能接收或拨打出实际的电话。它提供了多种导航和控制键，您可以使用您的鼠标或键盘来触发您应用程序的事件。它还提供了一个可以让您的应用程序显示的屏幕，并且可以与任何其他安卓应用程序同时运行。为了让您的应用程序模块化和测试更容易，模拟器支持Android虚拟设备（AVD）配置。模拟器可以让你指定任何你需要在模拟器上运行的Android平台，以及硬件的选择和模拟器皮肤文件。

3 安卓市场分析

A 安卓市场

安卓市场，是一个由谷歌为安卓设备开发的在线软件商店。它是在2008年10月22日开始面向用户的。大多数预装“应用市场”应用的安卓设备允许用户浏览、购买、下载和对不同的可用的应用程序进行排名以及其他为了在装有这个开源操作系统的手机上运行的内容。与苹果的应用程序商店不同的是，从安卓应用市场获取安卓应用程序没有任何要求。可以从任何来源获取安卓应用包括开发者自己的网站。此外，安卓开发者可以创建自己的应用市场。在应用在应用市场上架的方面，相比于苹果的审核，谷歌没有任何严格的要求。最后，安卓应用市场遵循一个7，3分成的模式给应用程序的开发者分成。开发者收取百分之70的应用程序的标价，剩余百分之30。截止2010年5月4日，安卓应用程序大约有49000个，而在2009年8月和2009年12月分别只有12500和20000个。

B. Android与Symbian和Windows Mobile的比较

可移植性：可移植性是一个非常重要的评估标准。Symbian操作系统在这方面有很多可以参考，具有标准的架构和开放的软件。但事实上，Symbian主要运行在诺基亚的手机，它落后于Android的原因是它不是基于Java语言的。不幸的是，Windows 手机同样也也有基于特定的硬件平台并且不可被移植的应用程序。Android是一个基于Linux和java的移动平台，它允许我们在许多不同的平台上使用，不像Symbian和Windows Mobile。所以结果是Android获取一分，Symbian OS得到了半分，和Windows Mobile的得分是零。

4 在Eclipse上创建你的第一个Android项目

要开始你的第一个Android项目，打开Eclipse。当您第一次打开Eclipse，它会打开一个空的开发环境，这就是开始的地方。你的首要任务是建立并为您的应用程序工作空间命名。选择File |new| Android Project，将会弹出安卓项目向导。

新的安卓项目向导为你做了两件事：一个使用android.jar文件关联到Android SDK的应用包，并且关联到安卓模拟器。这允许你的代码使用所有的安卓库和软件包，也可以让你在适当的环境中调试安卓程序。

你的第一个shell文件用于新项目。这些shell文件中包含一些重要可以构建程序的应用程序块。与创建Visual Studio Microsoft .NET应用程序时Windows自动在你的文件中创建某些的程序代码相同，使用Eclipse中的Andr​​oid项目向导生成您最初的程序文件和一些Android自动创建的代码。

此外，新的安卓项目向导包含了几个选项，显示下一步，您必须为了项目的启动而设置。

为了举这个例子，该项目的名称使用标题helloworldtext。这个名字足以区分这个项目和你将在本章创建的其他项目。

在内容区域，保留默认选项：Create New Project in Workspace的选项必须被选中，Use Default Location复选框必须被选中。这将允许在您的默认工作区目录创建您的项目。保持默认选项的优点是，您的项目被保存在一个中心位置，这使得您很容易分类、管理和查找这些项目。例如，如果你在一个基于Unix的环境下工作，这条路指向您的$HOME目录。如果你在windows环境中工作，工作空间路径为C:/Users/<username>/workspace。

然而，由于各种原因，您可能要取消选择Use Default Location的复选框，并且为您的项目重新选择一个位置。你想在这里指定一个不同位置的原因可能是您想为这个特殊的项目选择一个位置，以便于其他项目分隔。例如，你希望在本书中创建的项目与您在未来创建的属于自己的项目保持在不同的位置上，如果是的话，简单的重写位置选项为这个项目指定一个自定义的路径。