International Journal of Multidisciplinary Research and Modern Education (IJMRME)
Impact Factor: 7.315, ISSN (Online): 2454 - 6119
(www.rdmodernresearch.org) Volume 4, Issue 2, 2018

DEVELOPING MOBILE APPLICATION USING ANDROID FOR E-LEARNING CRAM RESOURCES

F. Angel Ingnishyaa*, T. Nagalakshmi & R. Padmavathi*****
Assistant Professor, Department of Computer Science and Engineering,
Dhanalakshmi Srinivasan Institute of Technology, Samayapuram,
Trichy, Tamilnadu

Cite This Article: F. Angel Ingnishyaa, T. Nagalakshmi & R. Padmavathi, "Developing Mobile Application Using Android for E-Learning Cram Resources", International Journal of Multidisciplinary Research and Modern Education, Volume 4, Issue 2, Page Number 108-115, 2018.

Abstract:

Mobile computing is human computer synergy by which a computer is expected to be transported during normal usage, which comprise confess transference of data, voice and video. Mobile computing involves mobile communication, mobile hardware, and mobile software. Communication contention encompasses ad-hoc networks and infrastructure networks as well as communication properties, protocols, data formats and substantial technologies. Hardware includes mobile devices or device components. Mobile software deals with the characteristics and stipulation of mobile applications. Android is a software assemblage for mobile devices that subsume an operating system, middleware and key applications. Android is a software platform and operating system for mobile devices situated on the Linux operating system and deviled by Google and the Open Handset Alliance. It confess developers to write persuaded code in a Java-like language that utilizes Google-deviled Java libraries, but does not sustentation programs deviled in hereditary code. The divulged of the Android platform of the Open Handset Alliance, a blunder bund of thirty four hardware, software and telecom companies adherent to advancing open standards for mobile devices.

Key Words: Middleware, Linux Operating System, Mobile Computing & Android

Overview of Android:

In July 2005, Google procure android Inc., a small startup company stationed in Palo Alto, CA. Android's co-founder who went to work to at Google include Andy Rubin (co-founder of danger), Rich miner (co-founder of wildfire communication, Inc), Nick sear (once VP T-Mobile), and chirs white (one of the first engineers at web TV). At the time, little was known about the functions of Android Inc. other than they made software for mobile phones. On 5 November 2007, the Open Handset Alliance, a syndicate of several companies which include Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, Sprint Nextel and NIVIDIA, was divulge with the goal to flourish open standards for mobile devices.

Linux Kernel:

Android Architecture is based on Linux 2.6 kernel. It helps to manipulate security, memory management, process management, network stack and other important contention. Therefore, the user should bring Linux in his mobile device as the main operating system and install all the drivers prescribe in order to run it. Android provides the substratum 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 encompass major features:
WCDMA/HSUPA and EGPRS network timber.

- Bluetooth 1.2 and Wi-Fi substratum.
- Digital audio rampart for mp3 and other formats
- Stiffener for Linux and other third-party operating systems
- Java hardware acceleration and shore for Java applications
- Camera up to 6.0 megapixels.

Existing System:

The Existing methods to Android are now the most used mobile operating system in the world. Android now has more users, more phones and more tablets worldwide than any other mobile operating system. The Google Play app store has been crescent at precipitate speed and with virtually as many apps as the Apple app store. This, for undertaker and defilers, is the chance of a lifetime to make even more money and grasp an even broader patron's base. This paper gives a replete wisdom of how to start alive on eclipse and develop an application and get it run on rival. Android is a Linux-based, open source mobile operating system by Open Handset Alliance led by Google to refined apps for Android devices. To start with we use a set of utensils that are subsumed in the Android SDK. Once we have downloaded and installed the SDK, we can approach these utensils right from our Eclipse IDE, concluded the ADT plug-in, or from the command line. refined with Eclipse is the preordered method because it can directly invoke the utensils that we need while refined applications..

The basic steps for refined applications are shown in Figure 1. The devilment steps encompass four development phases, which comprised:

- Setup: During this phase we install and set up our defilement environment. We also devise Android Virtual Devices (AVDs) and connect hardware devices, on which we can install our applications.
- Development: During this phase we set up and develop our Android project, which contains all of the source code and resource files for our application.
- Debugging and Testing: During this phase we build our project into a debug gable .apk package that we can install and run on the emulator.
- Publishing: During this phase we configure and build our application for release.

Eclipse is an integrated development environment (IDE). It contains a base workspace and an extensible plug-in system for customizing the environment. Written mostly in Java, Eclipse can be used to develop applications in Java. The initial codebase originated from IBM. The Eclipse software development kit (SDK), which includes the Java development tools, is meant for Java developers. Users can extend its abilities by installing plug-ins written for the Eclipse Platform, such as development toolkits for other programming languages, and can write and contribute their own plug-in modules. Released under the terms of the Eclipse Public License, Eclipse SDK is free and open source software. Our objective behind this paper presentation was to discuss all basic details to start android application and to overcome the technical jargons which come as a big constraint on the way of beginner programmer. Simplicity was the major factor in explaining all installation process of eclipse and a simple android application, which will give a bust to all aspiring android developers.

Drawbacks:

- E-Learning has not been developed as an specific application for study purpose.
- Mostly e-Learning concept is available in windows pc which is hard to carry to all the places.
- Eclipse is more complicated than android studio.
- Drag and drop action is not available in the eclipse.
- Eclipse requires higher amount of RAM space.

But nowadays everyone is using mobile phones and so many applications are available for studying purpose. No any applications are available for engineering students. We have developed an android application which can also be said as an e-Book for engineering students. The main concept of the project is to reduce the difficulty of students in need of their study materials. Students are facing much difficulty to refer the correct text book for studying. The materials are not available in all the books, they need to refer many books and even in their text books also every concept is not available. Students having arrears are also not able to get their proper study materials. Also it is very difficult for the students to search in internet because they need to search in many websites. All the materials are not available at one place itself. This project acts as the centralized concept for students by providing easy access. The project helps all the students to get their study materials at one place. Each year students can get all their study materials, question banks at one place even if the regulation changes. Once the admin of the application updates the materials inside the application, all the materials are available in the application. There is no need of an update in this application, and then once downloaded can be used. It is very easy to carry and helpful to study. It is not necessary for every students to download the application separately, once downloaded it can be shared.

System Architecture:

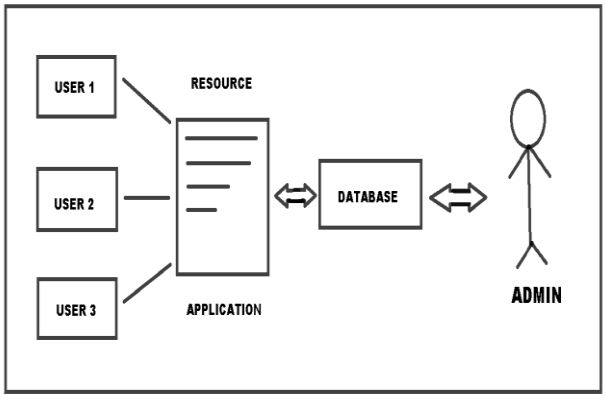


Figure: System Architecture

User:

The person having permission to access the application is the authorized user. The user can use the application and get benefit through the application. A system user is a person who interacts with a system, typically through an interface, to extract some functional benefit. User-centered design, often associated with human–computer interaction, considers a wide range of generic systems. System user also defines the behavior

of the system operations and how the audience (end-user) would interact with the system using pre-designed trigger such as buttons/mouse/keyboard. In order for the system to work on a larger scale using various databases system would have to create an interface that would be suitable for specific level of knowledge that the end user acquires.[1] Users authorized to change the way the system behaves are often called operators. Users who rely on a system, but do not operate it, are sometimes called the audience or end user.

Resource:

In computing, a system resource, or simply resource, is any physical or virtual component of limited availability within a computer system. Every device connected to a computer system is a resource. Every internal system component is a resource. Virtual system resources include files (concretely file handles), network connections (concretely network sockets), and memory areas. Managing resources is referred to as resource management, and includes both preventing resource leaks (releasing a resource when a process has finished using it) and dealing with resource contention (when multiple processes wish to access a limited resource).

Database:

A database server is a computer program that provides database services to other computer programs or computers, as defined by the client–server model. The term may also refer to a computer dedicated to running such a program. Database management systems frequently provide database server functionality, and some DBMSs (e.g., My SQL) rely exclusively on the client–server model for database access, Such a server is accessed either through a "front end" running on the user's computer which displays requested data or the "back end" which runs on the server and handles tasks such as data analysis and storage. In a master-slave model, database master servers are central and primary locations of data while database slave servers are synchronized backups of the master acting as proxies. Most of the Database servers work with the base of Query language. Each Database understands its query language and converts it to Server readable form and executes it to retrieve the results. Some examples of proprietary database servers are Oracle, DB2, Informix, and Microsoft SQL Server. Examples of GNU General Public License database servers are Ingres and My SQL. Every server uses its own query logic and structure. The SQL query language is more or less the same in all relational database servers. DB-Engines list over 200 DBMSs in its ranking.

Administrator:

Administrators use specialized software to store and organize data .The role may include capacity planning, installation, configuration, database design, migration, performance monitoring ,security, troubleshooting, as well as backup and data recovery. There are many certifications available for becoming a certified database administrator. Many of these certifications are offered by database vendors themselves. By passing a series of tests and sometimes other requirements, you can earn a database administrator .Schools offering Database Administration degrees can also be found.

In most cases, every Android application runs in its own Linux process. This process is created for the application when some of its code needs to be run, and will remaining until it is no longer needed and the system needs to reclaim its memory for use by other applications. An important and unusual feature of Android is that an application process's lifetime is not directly controlled by the application itself. Instead, it is determined by the system through a combination of the parts of the application that the system knows are running, how important these things are to the user, and how much overall memory is available in the system.

To determine which processes should be killed when low on memory, Android places them into an "importance hierarchy" based on the components running in them and the state of those components. These are, in order of importance:

A foreground process is one that is required for what the user is currently doing. Various application components can cause its containing process to be considered foreground in different ways. A process is considered to be in the foreground if any of the following conditions hold: It is running an Activity at the top of the screen that the user is interacting with (it's on Resume method has been called). It has a Broadcast Receiver that is currently running (it's Broadcast Receiver. on Receive method is executing). It has a Service that is currently executing code in one of its Callbacks (Service. On Create, Service. On Start, or Service. On Destroy).

There will only ever be a few such processes in the system, and these will only be killed as a last resort if memory is so low that not even these processes can continue to run. Generally, at this point, the device has reached a memory paging state, so this action is required in order to keep the user interface responsive.

- A visible process is one holding an Activity that is visible to the user on-screen but not in the foreground (it's on Pause method has been called). This may occur, for example, if the foreground Activity is displayed as a dialog that allows the previous Activity to be seen behind it. Such a process is considered extremely important and will not be killed unless doing so is required to keep all foreground processes running.
- A service process is one holding a Service that has been started with the start Service method. Though these processes are not directly visible to the user, they are generally doing things that the user cares

about (such as background mp3 playback or background network data upload or download), so the system will always keep such processes running unless there is not enough memory to retain all foreground and visible process.

- A background process is one holding an Activity that is not currently visible to the user (it's on Stop method has been called). These processes have no direct impact on the user experience. Provided they implement their Activity
- Life-cycle correctly (see Activity for more details), the system can kill such processes at any time to reclaim memory for one of the three previous processes types. Usually there are many of these processes running, so they are kept in an LRU list to ensure the process that was most recently seen by the user is the last to be killed when running low on memory.
- An empty process is one that doesn't hold any active application components. The only reason to keep such a process around is as a cache to improve startup time the next time a component of its application needs to run. As such, the system will often kill these processes in order to balance overall system resources between these empty cached processes and the underlying kernel caches.

Android Architecture:

Application Framework:

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse of components, any application

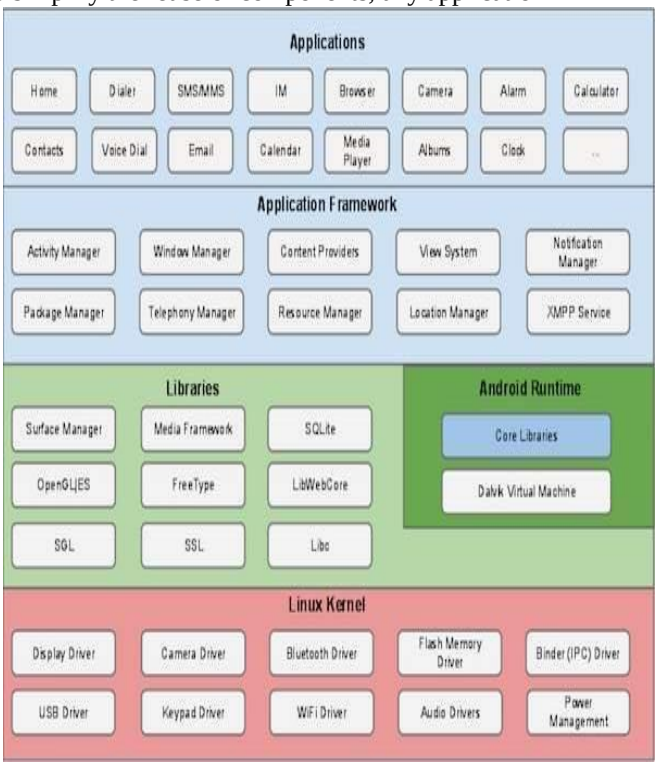


Figure: Architecture of Android OS

Developers have full access to the same framework APIs used by the core applications. The application architecture is designed to simplify the reuse 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. Content Providers that enable applications to access 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 lat files.
- A Notification Manager that enables all applications to display customer alerts in the status bar.
- An Activity Manager that manages the life cycle of applications and provides a common navigation back stack.

Libraries:

Android includes a set of C/C++ libraries used by various 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 (lib c), tuned for embedded Linux-based devices.

- Media Libraries – based on Packet Video's Open CORE; the libraries support playback and recording of many popular audio and video formats.
- Surface Manager – manages access to the display subsystem and seamlessly composite 2D and 3D graphic layers from multiple applications.
- Lib Web Core – a modern web browser engine which powers both the Android browser and an embeddable web view.
- SGL – the underlying 2D graphics engine.
- 3D libraries – an implementation based on OpenGL ES 1.0 APIs; the libraries use either hardware 3D acceleration (where available) or the included, highly optimized 3D software rasterizer.
- Free Type – bitmap and vector font rendering.
- SQ Lite – a powerful and lightweight relational database engine available to all applications.

Android Runtime:

Android 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. 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 minimal 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 relies on the Linux kernel for underlying functionality such as threading and low-level memory management.

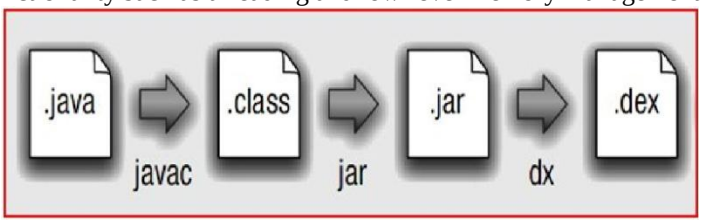


Figure: Conversion from .java to .dex file

As the result, it is possible to have multiple instances of Dalvik virtual machine running on the single device at the same time. The Core libraries are written in java language and contains of the collection classes, the utilities, IO and other tools.

Android Execution Environment:

We represent the regular Java and Android execution paths in the figures respectively. It is interesting to note here however is that the Android compilers do not operate on Java Language code. Instead, the Android translators work on the resulting Java byte code emitted from a traditional Java compiler.

As such, it is possible to reuse existing Java libraries, even if the original source code is not available. Such libraries must meet stringent requirements however, they need to:

- Adhere to the Java SE 5 dialect
- Not use any Java classes or packages found in Java SE 5 not found in the Android platform
- Not use any packages or classes specific to the Sun Microsystems platform
- Still behave in a predictable manner under the Apache Harmony Java environment.

Following these guidelines, it's possible to integrate existing Java source code, package and libraries piecemeal. Special care will be needed in the integration phase of such but the potential savings offered by such integration far outweighs the cost of rewriting well-coded, well-documented and well-tested libraries ready for use. Furthermore, it is expected that has Apache Harmony matures, more and more compatibility issues will be resolved further increasing the pool of available Java code that will be able to execute unmodified under the Android platform.

The Dalvik Virtual Machine:

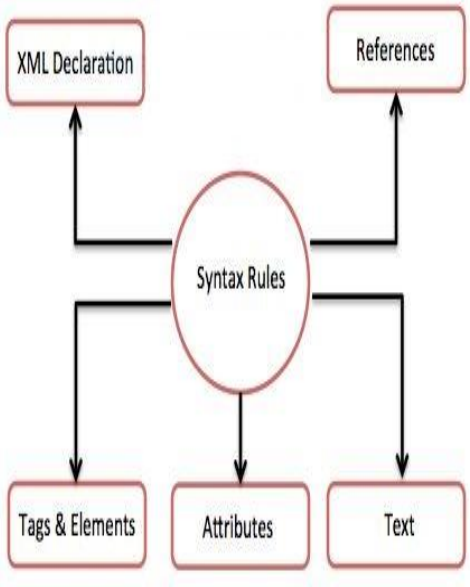
The Dalvik virtual machine is an interpreter only machine optimized for use on low powered, low memory devices like phones. Notably, Dalvik does not make use of just in time (JIT) Compilation to improve the performance of an application at runtime. Furthermore, Dalvik is not a Java virtual machine. This is because Dalvik is unable to read Java bytecode34; instead it uses its own byte code format called "dex". Google claims this format allows battery power to be better-conserved at all different stages of execution of an application. This means that standard Java SE applications and libraries cannot be used directly on the Android Dalvik virtual machine.

What is Markup?

XML is a markup language that defines set of rules for encoding documents in a format that is both human-readable and machine-readable. So what exactly is a markup language? Markup is information added to a document that enhances its meaning in certain ways, in that it identifies the parts and how they relate to each other. More specifically, a markup language is a set of symbols that can be placed in the text of a document to demarcate and label the parts of that document.

Is XML a Programming Language?

A programming language consists of grammar rules and its own vocabulary which is used to create computer programs. These programs instructs computer to perform specific tasks. XML does not qualify to be a programming language as it does not perform any computation or algorithms. It is usually stored in a simple text file and is processed special software that is capable of interpreting XML.



.

Figure: XML Declaration Syntax

Development Tools:

The Android SDK includes a variety of custom tools that help develop mobile application on the Android platform. The most important of these are the Android Studio and the Android Development Tools plugins for Android studio, but the SDK also includes a variety of other tools for debugging, packaging, and installing are applications on the emulator.

Android Emulator:

The Android SDK includes a mobile device emulator — a virtual mobile device that runs on your computer. The emulator lets you develop and test Android applications without using a physical device. This document is a reference to the available command line options and the keyboard mapping to device keys.

Android Development Tools Plugins for the Android Studio IDE:

The ADT plug-in adds powerful extensions to the Android Studio integrated environment; making creating and debugging are Android Applications easier and faster. If use Android Studio, the ADT plug-in gives an incredible boost in developing Android Applications:

- Flexible Gradle-based build system
- Build variants and multiple ark file generation
- Code templates to help you build common app features
- Rich layout editor with support for drag and drop theme editing
- Lint tools to catch performance, usability, version compatibility, and other problems
- ProGuard and app-signing capabilities
- Built-in support for Google Cloud Platform, making it easy to integrate Google Cloud Messaging and App Engine

Dalvik Debug Monitor Service (DDMS):

Android Studio includes a debugging tool called the Dalvik Debug Monitor Server (DDMS), which provides port-forwarding services, screen capture on the device, thread and heap information on the device, logcat, process, and radio state information, incoming call and SMS spoofing, location data spoofing, and more.

Android Asset Packaging Tool (AAPT):

This tool is part of the SDK (and build system) and allows you to view, create, and update Zip-compatible archives (zip, jar, apk). It can also compile resources into binary assets.

Android Interface Description Language:

AIDL is similar to other IDLs you might have worked with. It allows you to define the programming interface that both the client and service agree upon in order to communicate with each other using interprocess communication (IPC).

Trace View:

This tool produces graphical analysis views of trace log data that can generate from our Android application.

Mksd Card:

The mksdcard tool lets you quickly create a FAT32 disk image that you can load in the emulator, to simulate the presence of an SD card in the device. Because you can specify an SD card while creating an AVD in the AVD Manager, you usually use that feature to create an SD card. This tool creates an SD card that is not bundled with an AVD, so it is useful for situations where you need to share a virtual SD card between multiple emulators.

Activity Creator:

It is regarded as the initial stage involved in the creation of an android project. Shell scripts forms a part of activity creator, which is used for the creation of new file system structure that helps in writing codes in Android IDE.

Security and Permissions in Android:

Android is a multi-process system, where each application (and parts of the system) runs in its own process. Most security between applications and the system is enforced at the process level through standard Linux facilities, such as user and group IDs that are assigned to applications. Additional finer-grained security feature are provided through a "permission" mechanism that enforces restrictions on the specific operations that a particular process can per. Android mobile phone platform is going to be more secure than Apple's iPhone or any other device in the long run. There are several solutions nowadays to protect Google phone from various attacks. One of them is security vendor McAfee, a member of Linux Mobile (Li-Mo) Foundation.

Another solution for such attacks is S-Mobile System mobile package. Security Shield –an integrated application that includes anti-virus, anti-spam, firewall and other mobile protection is up and ready to run on the Android operating system. Currently, the main problem is availability for viruses to pose as an application and do things like dial phone numbers, send text messages or multimedia messages or make connections to the Internet during normal device use. It is possible for somebody to use the GPS feature to track a person's location without their knowledge. Android is a truly open, free development platform based on Linux and open source. Handset makers can be use and customize 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 originally 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 open to all; industry, developers and users. Participating in many of the successful open source projects. Aims to be easy to build for as the web. Google Android is stepping into the next level of Mobile Internet.

Conclusion:

In this paper Android is now stepping up in next level of mobile internet. There are chances of Android may become the widely used operating system in world. Our android application will meet all the expectations of engineering students. The accessibility is very easy in our application as it acts as the centralized concept for the students to get all study materials at one place. High cost is also not required to download the application. This application will be more helpful to the students and every student will love this application as this brings an easy access to their study purpose. This application may also be developed in windows and iOS. Not only the study materials also students profile, exam results from the first semester, fees details may also be developed in this application. It may also be uploaded in the play store and everyone will be able to download the application. This application can also be developed for all the course of study.

References:

1. Ma, Li, Lei Gu, and Jin Wang. "Research and Development of Mobile Application for android Platform." (2014).
2. Liu, Jianye, and Jiankun Yu. "Research on Development of android Applications." Fourth International conference on Intelligent Networks and Intelligent Systems. 2011.
3. Parada, Abilio G., and Lisane B. de Brisolara. "A model driven approach for android applications development." Computing System Engineering (SBESC), 2012 Brazilian Symposium on. IEEE, 2012.
4. Peng, Bin, Jinming Yue, and Chen Tianzhou. "The android Application Development College Challenge." High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems (HPCC-ICESS), 2012 IEEE 14th International Conference on. IEEE, 2012.
5. Grgurina, Robi, Goran Brestovac, and Tihana Galinac Grbac. "Development environment for android application development: An experience report." MIPRO, 2011 Proceedings of the 34th International Convention. IEEE, 2011.
6. Zhi-An, Yi, and Mu Chun-Miao. "The development and application of sensor based on android." Information Science and Digital Content Technology (ICIDT), 2012 8th International Conference on. Vol. 1. IEEE, 2012.

7. Yang, Zhilong, et al. "Research and Design of a Real-Time Interactive Application Development Model Based on the android Platform." Computational Intelligence and Design (ISCID), 2013 Sixth International Symposium on. Vol. 1. IEEE, 2013.
8. D. Gavalas and D. Economou, “Development Platforms for Mobile Applications: Status and Trends”, Software, (2011), pp. 77 – 86.
9. X. Zhao and D. Tian, “The Architecture Design of Streaming Media Applications for Android OS”, ICSESS, (2012), pp. 280 – 283.
10. M. Butler, “Android: Changing the Mobile Landscape”, Pervasive Computing, (2011), pp. 4-7