Chapter 1

Introduction to mobile Oses

1.1 Introduction of Mobile Oses

1.1.1 Mobile Phone

A mobile phone (also known as a cellular phone, cell phone, hand phone, or simply a phone) is a phone that can make and receive telephone calls over a radio link while moving around a wide geographic area. It does so by connecting to a cellular network provided by a mobile phone operator, allowing access to the public telephone network. By contrast, a cordless telephone is used only within the short range of a single, private base station.

It supports variety of services such as text messaging, MMS, email, Internet access, short-range wireless communications (infrared, Bluetooth), business applications, video games, and digital photography and many others.

The common components found on all phones are:

- A battery, providing the power source for the phone functions.
- An input mechanism to allow the user to interact with the phone. The most common input mechanism is a keypad, but touch screens are also found in most smartphones.
- A screen which echoes the user's typing, displays text messages, contacts and more.
- Basic mobile phone services to allow users to make calls and send text messages.

- All GSM phones use a SIM card to allow an account to be swapped among devices. Some CDMA devices also have a similar card called R-UIM.

1.1.2 History of Mobile Phone

This history focuses on communication devices which connect wirelessly to the public switched telephone network. The transmission of speech by radio has a long and varied history going back to Reginald Fessenden's invention and shore-to-ship demonstration of radio telephony. The first mobile telephones were barely portable compared to today's compact hand-held devices. Along with the process of developing more portable technology, drastic changes have taken place in the networking of wireless communication and the prevalence of its use. The world's first mobile phone call was made on April 3, 1973, when Martin Cooper, a senior engineer at Motorola, called a rival telecommunications company and informed them he was speaking via a mobile phone.

The major important dates in the history of mobile operating systems are:

- 1973—1993 Mobile phones use embedded systems to control operation.
- 1995 The first smartphone, the IBM Simon, has a touchscreen, email and PDA features.
- 1996 Palm Pilot 1000 personal digital assistant is introduced with the Palm OS mobile operating system.
- 1996 First Windows CE Handheld PC devices are introduced.
- 1999 Nokia S40 OS is officially introduced along with the Nokia 7110.
- 2000 Symbian becomes the first modern mobile OS on a smartphone with the launch of the Ericsson R380.
- 2001 The Kyocera 6035 is the first smartphone with Palm OS.
- 2002 Microsoft's first Windows CE (Pocket PC) smartphones are introduced.
- 2002 BlackBerry releases its first smartphone.
- 2005 Nokia introduces Maemo OS on the first internet tablet N770.

- 2007 Apple iPhone with iOS is introduced as an iPhone, "mobile phone" and 'internet communicator. "
- 2007 Open Handset Alliance (OHA) formed by Google, HTC, Sony, Dell, Intel, Motorola, Samsung, LG, etc.
- 2008 OHA releases Android (based on Linux Kernel) 1.0 with the HTC Dream (T-Mobile Gl) as the first Android phone.
- 2009 Palm introduces webOS with the Palm Pre. By 2012 webOS devices were no longer sold.
- 2009 Samsung announces the Bada OS with the introduction of the Samsung S8500.
- 2010 Windows Phone OS phones are released but are not compatible with the previous Windows Mobile OS. 2011 MeeGo the first mobile Linux, combining Maemo and Moblin, is introduced with the Nokia N9, a collaboration of Nokia, Intel and Linux Foundation
- 2011 Samsung, Intel and the Linux Foundation announced, in September 2011, that their efforts will shift from Bada, MeeGo to Tizen during 2011 and 2012.
- 2011 the Mer project was announced, in October 2011, centered around an ultra-portable Linux + HTML5/QML/JavaScript core for building products with, derived from the MeeGo codebase.
- 2012 Mozilla announced in July 2012 that the project previously known as "Boot to Gecko"(which was built on top of Android Linux kernel and using Android drivers, however it doesn't use any Java-like code of Android) was now Firefox OS and had several handset OEMs on board.
- 2013 Canonical announced Ubuntu Touch, a version of the Linux distribution expressly designed for smartphones. The OS is built on the Android Linux kernel, using Android drivers, but does not use any of the Java-like code of Android.
- 2013 BlackBerry releases their new operating system for smartphones, BlackBerry 10.
- 2013 Google releases Android KitKat 4.4.
- 2014 Microsoft releases Windows Phone 8.1 in February 2014.

- 2014 Apple releases iOS 8 in September 2014.
- 2014 BlackBerry release BlackBerry 10.3 with integration with the Amazon Appstore in September 2014.
- 2014 Google releases Android 5.0 "Lollipop" in November 2014.
- 2015 Apple releases iOS 9 in September 2015.

1.1.3 SIM (Subscriber Identification Module)

A subscriber identity module or subscriber identification module (SIM) is an integrated circuit that is intended to securely store the international mobile subscriber identity (IMSI) number and its related key, which are used to identify and authenticate subscribers on mobile telephony devices (such as mobile phones and computers).
GSM feature phones require a small microchip called a Subscriber Identity Module or SIM card, to function. The SIM card is approximately the size of a small postage stamp and is usually placed underneath the battery in the rear of the unit.
It is also possible to store contact information on many SIM cards. SIM cards are always used on GSM phones; for CDMA phones, they are only needed for newer LTE (Long Term Evolution)-capable handsets. SIM cards can also be used in satellite phones, computers, or cameras.

1.1.4 Operating System

An operating system (OS) is system software that manages computer hardware and software resources and provides common services for computer programs. The operating system is a component of the system software in a computer system. Application programs usually require an operating system to function.
For hardware functions such as input and output and memory allocation, the operating system acts as an intermediary between programs and the computer hardware, although the application code is usually executed directly by the hardware and frequently makes system calls to an OS function or be interrupted by it. Operating systems are found on many devices that contain a computer from cellular phones and video game consoles to web servers and supercomputers.

1.1.5 Mobile Operating System

Different types of Mobile Operating Systems are:

1. Android

   - Android is the most powerful and popular mobile platform in the world.
   - Based on the Linux Karnel.
   - Released by Google under an OpenSource license.
   - Runs on a range of devices including smart phones, tablets, smart watches, cars and even smart television interfaces.
   - Supports ARMv7 (Advance RISC Machine) ARMv8A, x86 and MIPS (Microprocessor without Interlocked Pipeline Storage) chips architecture.
   - Started in 2003 as Android Inc which was later acquired by Google Inc in 2005.

   Android is a mobile operating system (OS) currently developed by Google, based on the Linux kernel and designed primarily for touchscreen mobile devices such as smartphones and tablets. Android's user interface is based on direct manipulation, using touch gestures that loosely correspond to real-world actions, such as swiping, tapping and pinching, to manipulate on-screen objects, along with a virtual keyboard for text input. In addition to touchscreen devices, Google has further developed Android TV for televisions, Android Auto for cars and Android Wear for wrist watches, each with a specialized user interface. Variants of Android are also used on notebooks, game consoles, digital cameras, and other electronics. As of 2015, Android has the largest installed base of all operating systems.

   Initially developed by Android, Inc., which Google bought in 2005, Android was unveiled in 2007, along with the founding of the Open Handset Alliance a consortium of hardware, software, and telecommunication companies devoted to advancing open standards for mobile devices

2. Java OS: J2ME platform is a set of technologies, specifications and libraries developed for small devices like mobile phones, pagers, and personal organizers. Java ME was designed by Sun Microsystems. It is licensed under GNU General Public License. It defines a minimum

platform including the java language, virtual machine features and minimum class libraries for a grouping of devices. It supports higher-level services common to a more specific class of devices. A profile builds on a configuration but adds more specific APIs to make a complete environment for building applications.
It includes two kinds of platforms:
- High-end platform for high-end consumer devices. E.g. TV set-top boxes, Internet TVs, auto-mobile navigation systems
- Low-end platform for low-end consumer devices. E.g.cell phones, and pagers.

3. iOS: iOS (originally iPhone OS) is a mobile operating system created and developed by Apple Inc. and distributed exclusively for Apple hardware. It is the operating system that presently powers many of the company's mobile devices, including the iPhone, iPad, and iPod touch. It is based on Mach Kernel and Drawing core as Mac OS X.
The Mac OS X kernel includes the following component:
- Mach Kernel BSD
- I/O component
- File Systems
- Networking components
Mac OS X has a preemptive multitasking environment. Preempting is the act of taking the control of operating system from one task and giving it to another task. It supports real-time behavior. In Mac OS X, each application has access to its own 4 GB address space.

4. Ubuntu Touch: Ubuntu Touch (also known as Ubuntu Phone) is a mobile version of the Ubuntu operating system developed by Canonical UK Ltd and Ubuntu Community.lt is designed primarily for touch-screen mobile devices such as smartphones and tablet computers.

   Mark Shuttleworth announced on 31 October 2011 that by Ubuntu 14.04, Ubuntu will support smartphones, tablets, smart TVs and other smart screens (as car head units and smart watches). Ubuntu Touch was released to manufacturers on 16 September 2014.BqAquaris E4.5, the world's first Ubuntubased smartphone went on sale in Europe on 9 February 2015.

5. Tizen: Tizen is an operating system based on the Linux kernel and the Linux API. It targets a very wide range of devices including smartphones, tablets, in-vehicle infotainment (IVI) devices, smart TVs, PCs, smart cameras, wearable computing (such as smart watches), Blu-ray

players, printers and smart home appliances (such as refrigerators, lighting, washing machines, air conditioners, ovens/microwaves and a robotic vacuum cleaner). Its purpose is to offer a consistent user experience across devices. Tizen is a project within the Linux Foundation and is governed by a Technical Steering Group (TSG) composed of Samsung and Intel among others.

The Tizen Association was formed to guide the industry role of Tizen, including requirements gathering, identifying and facilitating service models, and overall industry marketing and education.

6. Windows Phone (WP): Windows Phone (WP) is a family of mobile operating systems developed by Microsoft for smartphones as the replacement successor to Windows Mobile and Zune. Windows Phone features a new user interface derived from Metro design language. Unlike Windows Mobile, it is primarily aimed at the consumer market rather than the enterprise market. It was first launched in October 2010 with Windows Phone 7.Windows Phone 8.1 is the latest public release of the operating system, released to manufacturing on April 14, 2014.

   Windows 10 Mobile will succeed Windows Phone 8.1 in 2015; it emphasizes a larger amount of integration and unification with its PC counterpart including a new, unified application ecosystem, along with an expansion of its scope to include small-screened tablets.

7. Firefox OS: Firefox OS is a Linux kernel-based open-source operating system for smartphones, tablet computers and smart TVs. It is being developed by Mozilla, the non-profit organization best known for the Firefox web browser.

   Firefox OS is designed to provide a complete, community-based alternative system for mobile devices, using open standards and approaches such as HTML5 applications, JavaScript, a robust privilege model, open web APIs to communicate directly with cellphone hardware, and application marketplace. As such, it competes with commercially developed operating systems such as Apple's iOS, Google's Android, Microsoft's Windows Phone, BlackBerry's BlackBerry 10 and Jolla's Sailfish OS.

   Firefox OS was publicly demonstrated in February 2012, on Android-

compatible smartphones. In January 2013, at CES 2013, ZTE confirmed they would be shipping a smartphone with Firefox OS, and on July 2, 2013, Telef6nica launched the first commercial Firefox OS based phone, ZTE Open, in Spain which was quickly followed by Geeks Phone's Peak+. As of December 16, 2014, Firefox OS phones are offered by 14 operators in 28 countries throughout the world. Mozilla has also partnered with T2Mobile to make a Firefox OS reference phone dubbed "Flame" which is designed for developers to contribute to Firefox OS and to test apps.

8. Symbian: Symbian is a closed-source mobile operating system (OS) and computing platform designed for smartphones.Symbian was originally developed by Symbian Ltd. The current form of Symbian is an open-source platform developed by Symbian Foundation in 2009, as the successor of the original Symbian OS. Symbian was used by many major mobile phone brands, like Samsung, Motorola, Sony Ericsson, and above all by Nokia. It was the most popular smartphone OS on a worldwide average until the end of 2010, when it was overtaken by Android. **Symbian OS was created with three systems design principles in mind:**

   (a) the integrity and security of user data is paramount

   (b) user time must not be wasted

   (c) all resources are scarce.

9. Blackberry: BlackBerry OS is a proprietary mobile operating system developed by BlackBerry Ltd for its BlackBerry line of smartphone handheld devices. The operating system provides multitasking and supports specialized input devices that have been adopted by BlackBerry Ltd. for use in its handhelds, particularly the trackwheel, trackball, and most recently, the trackpad and touchscreen.

   The BlackBerry platform is perhaps best known for its native support for corporate email, through MIDP 1.0 and, more recently, a subset of MIDP 2.0, which allows complete wireless activation and synchronization with Microsoft Exchange, Lotus Domino, or Novell GroupWise email, calendar, tasks, notes, and contacts, when used with BlackBerry Enterprise Server. The operating system also supports WAP I .2.

1.2 Build and Structures of Mobile OSes

A mobile OS is a software platform on top of which other programs called application program, can run on mobile devices such as PDA, cellular phones, Smartphone and etc.

1.2.1 Low Level Hardware

1. Mobile Processors: - Processors use RISC architecture. Mainly two types:
   a.ARM Processors (32-bit RISC processor with Low power consumption) b.MIPS - Microprocessor without Interlocked Pipeline Stages used in embedded systems

2. Mobile Memory:- The better the memory management offered by the OS, the wider the options available to applications developers. Mobile devices have two types of memories
   - ROM:- For operating system and preinstalled programs
   - RAM:- For user information
   Types of RAMs:-
   - DRAM (Dynamic RAM): cheapest, used in mobile devices
   - EDO (Enhanced Data Output): more expensive but offers a speed increase over DRAM
   - SDRAM (Synchronous Dynamic RAM): a further 50% speed (iPAQ)
   - DDR (Double Data Rate) SDRAM is twice as fast as SDRAM
   - OUM (Ovonics Unified Memory): experimental

3. Kernal: Responsible for services such as
   - Security
   - Memory Management
   - Process Management

It include the following components
- I/O Components
- File Systems
- Networking Components

content...

1.3 Introduction to development environment (Native v/s HTML5)

1.3.1 Native Apps

Any app installed directly into a device usually from its associated app store can be called a "native" app (although a native app can have HTML5 elements). Native apps are platform dependent i.e. an app made for iPhone can work only on an iPhone, not an Android device. **Native apps have distinct advantages and disadvantages, such as:**

Pros of Native App

- Better performance: A native app has complete access to a phone's hardware resources. The app also interacts directly with the phone without the mediation of a web browser. This lends it significantly better performance, particularly when rendering graphics and animation.

- Easier development: Developers can easily leverage platform SDKs to create native apps. This makes for easier development, provided you already have access to experienced engineering talent.

- Better distribution: Native apps can be distributed directly through relevant app stores. This can be an incredibly powerful distribution channel that not only makes app installation and updates easy, but can also help cut down on marketing costs through app stores' built-in discovery features.

- Better monetization: Since native apps are associated with app stores, they can take advantage of the store's built-in monetization features, such as one-click payments.

Cons of Native App

- Increased development time and costs: Since developers must build separate apps for each platform, total development time might be significantly higher than an equivalent HTML5 app. This also increases costs since developing for different platforms demands mastery over different languages and development environments. App developers themselves tend to be more expensive than their HTML5/JavaScript counterparts. Higher maintenance costs: From the above, it follows that maintenance costs for native apps are also higher. A business might require the services of separate iOS, Android, and Windows Phone developers to keep its apps up and running, eating into profits.

- App store content restriction: Every app distributed through the app store must adhere to strict content guidelines. For certain businesses (such as gambling), this may be a deal-breaker.

- App store fees: App store fees (up to 30 percent) can eat into your profit margins and make an already costly undertaking prohibitively expensive.

1.3.2 HTML5 Apps

A web app delivered through an HTML5 enabled browser may be called an "HTML5 app." This means instead of downloading your app from the app store, a user would open her/his browser and navigate to a URL to access your app. **HTML5 apps offer various advantages and disadvantages, such as:**
Procs of HTML5

- Platform independent: HTML5 apps are completely platform independent. It does not matter whether your users are on Android, iOS, or Windows Phone, they can all access your app as long as they have a web browser.

- Easier updates: Native apps update much like desktop applications — the user has to download each update individually. HTML5 apps, on the other hand, support centralised updates. Every time the user visits your app through her/his browser, she/he sees the latest version of your app with no additional download requirements.

- Faster development: HTML5 apps are written in HTML, CSS, JavaScript, and server-side languages such as ASP.NET. This cuts down on development time since there are already vast libraries for these languages (such as jQuery for JavaScript) and a huge body of trained developers.

- Cheaper: Since HTML5 apps are built on relatively simple web technologies such as HTML5 and JavaScript, you will find it easier to hire affordable HTML5 developers than similarly talented native app developers.
- No content restrictions: Since HTML5 apps need not seek approval from a closed app store, they can carry whatever content you want.
- No fees: HTML5 apps are delivered directly through the browser. This cuts out the app store and saves you on the 30 percent app store fees.
- Better data: Any customer data you collect through a native app is subject to the rules of the app store in question. There are no such restrictions with HTML5 apps; you can collect whatever customer data you want. In fact, the Apple App Store withholding subscriber data was one of the reasons why Financial Times switched to a web app.
- Distribution through app store: Until a couple of years ago, there was no easy method to distribute a HTML5 app through the app store. However, new frameworks such as Phonegap and Trigger.io have eased the distribution problem considerably bundling up HTML5 apps as native apps (which can then be distributed through the app store). Although performance remains a concern and Phonegap still doesn 't support every mobile OS fully, it's a capable alternative to building separate native apps for every platform.

Cons of HTML5

- Poorer performance: An HTML5 app has limited access to a phone's hardware. This leads to poorer performance, especially when dealing with heavy graphics, although new platforms such as Famo.us are trying to mitigate this problem.
- Device fragmentation: Device fragmentation within web browsers is a real thing. Different devices might render the same app differently. This means your developers will need extensive testing and fine-tuning to get the UI right, especially when working on more complex apps.
- Technical limitations: An HTML5 does not have complete access to a device's features and events. This poses a limit on what you can do with an HTML5 app. The UI options for HTML5 apps are also limited.
- Limited monetisation opportunities: A native app can make money through direct app sales and in-app purchases. These monetisation

options are not available for HTML5 apps. This can be particularly challenging for "freemium" apps.

1.4 Introduction to Android

It is a platform and an operating system for mobile devices based on the Linux operating system. It allows developers design applications in a java-like language using Google-developed java libraries.

1.4.1 API Levels and Versions of Android

The version history of the Android mobile operating system began with the release of the Android beta in November 2007. The first commercial version, Android 1.0, was released in September 2008. Android is under ongoing development by Google and the Open Handset Alliance (OHA), and has seen a number of updates to its base operating system since its initial release.

Name	Version	API Level
Alpha	1.0	1
Beta	1.1	2
Cupcake	1.5	3
Donut	1.6	4
Eclair	2.0, 2.0.1, 2.1	5, 6, 7
Foryo	2.2	8
Gingerbread	2.3, 2.3.3	9, 10
Honeycomb	3.0, 3.1, 3.2	11, 12, 13
IceCreamSandwich	4.0, 4.0.3	14, 15
JellyBean	4.1, 4.2, 4.3	16, 17, 18
KitKat	4.4, 4.4W	19, 20
Lolipop	5.0, 5.1	21, 22
Marshmallow 6.0	23	
Nuget	7.0, 7.1.1, 7.1.2	24, 25
Oreo	8.0, 8.1	26, 27
AndridP	9.0	28

1.4.2 Pros and Cons of Android

- Pros:
  - Android is open source Operating System and is customizable.
  - Android has large number of OEM's (Original Equipment Manufacturers)

  - It solves user's problems through Google support. - Android has large number of apps in its play store.

- Cons:
  - Not as secure as iOS and windowsOS.
  - Too many apps creating mess in the store, heavy OS in itself
  - Most of the apps are dolphin browsers, wallpapers, malwares (considerable part) or are not even downloaded once.
  - App compatibility issues for various versions of android. Not all phones run the same version at a given time.
  - Over time phones become slow in operation.
  - Quickly eat up the battery

1.4.3 Comparison of Android with other Mobile OSes

- The number of apps of Android in its store (Googleplay ,Mobango) is about 1.5 million, apple's store is about 1.4 million and windows has got about 0.4 million till date.
- Android was the first to support Wi-Fi Direct, multiple user accounts and screen mirroring support.
- The number of OEM for android OS is more than 100, for windows only about 25 while iOS has only one i.e. Apple Inc.
- Android is open source and customizable where iOS and windows are closed source. Introduction to android VM and Runtime (Dalvik and ART)

1.5 Introduction to Android VM and Runtime (Dalvik and ART)

1.5.1 Dalvik

Dalvik is a process virtual machine (VM) in Google's Android operating system that executes applications written for Android. This makes Dalvik an integral part of the Android software stack (in Android versions 4.4 "KitKat" and earlier) that is typically used on mobile devices such as mobile phones and tablet computers, as well as more recently on devices such as smart TVs

and wearables. Programs are commonly written in Java and compiled to bytecode for the Java virtual machine, which is then translated to Dalvik bytecode and stored in .dex(DalvikEXecutable) and .odex(Optimized Dalvik EXecutable) files; related terms odex and de-odexare associated with respective bytecode conversions. The compact Dalvik Executable format is designed for systems that are constrained in terms of memory and processor speed.

Unlike Java VMS, which are stack machines, the Dalvik VM uses a register-based architecture that requires fewer, typically more complex virtual machine instructions. Dalvik programs are written in Java using the Android application programming interface (API), compiled to Java bytecode, and converted to Dalvik instructions as necessary.

A tool called dx is used to convert Java .class files into the .dex format. Multiple classes are included in a single .dex file. Duplicate strings and other constants used in multiple class files are included only once in the .dex output to conserve space. Java bytecode is also converted into an alternative instruction set used by the Dalvik VM. An uncompressed .dex file is typically a few percent smaller in size than a compressed Java archive (JAR) derived from the same .class files.

The Dalvikexecutables may be modified again when installed onto a mobile device. In order to gain further optimizations, byte order may be swapped in certain data, simple data structures and function libraries may be linked inline, and empty class objects may be short-circuited, for example. Being optimized for low memory requirements, Dalvik has some specific characteristics that differentiate it from other standard VMs.

- The VM was slimmed down to use less space.. .. The constant pool has been modified to use only 32-bit indices to simplify the interpreter

- Standard Java bytecode executes eight-bit stack instructions. Local variables must be copied to or from the operand stack by separate instructions.

- Dalvik instead uses its own 16-bit instruction set that works directly on local variables. The local variable is commonly picked by a four-bit "virtual register" field. This lowers Dalvik's instruction count and raises its interpreter speed.

Android 2.2 brought trace-based just-in-time (JIT) compilation into Dalvik, optimizing the execution of applications by continually profiling applications each time they run and dynamically compiling frequently executed short segments of their bytecode into native machine code. While Dalvik interprets the rest of application's bytecode, native execution of those short bytecode segments, called "traces" provides significant performance improvements.

1.5.2 ART (Android Runtime)

Android Runtime (ART) is an application runtime environment used by the Android operating system. ART replaces Dalvik, which is the process virtual machine originally used by Android, and performs transformation of the application's bytecode into native instructions that are later executed by the device's runtime environment. ART and Dalvik are compatible runtimes running Dexbytecode, so apps developed for Dalvik should work when running with ART. However, some techniques that work on Dalvik do not work on ART.
Unlike Dalvik, ART introduces the use of ahead-of-time (AOT) compilation by compiling entire applications into native machine code upon their installation. By eliminating Dalvik's interpretation and trace-based JIT compilation, ART improves the overall execution efficiency and reduces power consumption, which results in improved battery autonomy on mobile devices. At the same time, ART brings faster execution of applications, improved memory allocation and garbage collection (GC) mechanisms, new applications debugging features, and more accurate high-level profiling of applications.
To maintain backward compatibility, ART uses the same input bytecode as Dalvik, supplied through standard .dex files as part of APK files, while the .odex files are replaced with Executable and Linkable Format (ELF) executables. Once an application is compiled by using ART's on-device dex20at utility, it is run solely from the compiled ELF executable; as a result, ART eliminates various application execution overheads associated with Dalvik's interpretation and trace-based JIT compilation. As a downside, ART requires additional time for the compilation when an application is installed, and applications take up slightly larger amounts of secondary storage (which is usually flash memory) to store the compiled code. Android 4.4 "KitKat" brought a technology preview of ART, including it as an alternative runtime environment and keeping Dalvik as the default virtual machine. In the next major Android release, Android 5.0 "Lollipop", Dalvik was entirely replaced by ART. **Some of the major features implemented by ART are:**

1. Ahead-of-time (AOT) compilation

2. Improved garbage collection

3. Development and debugging improvements like

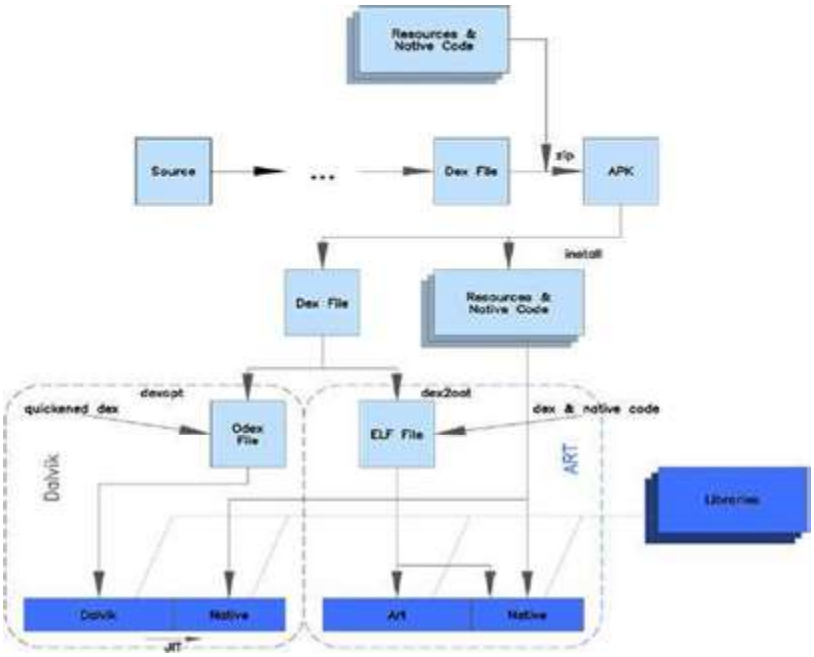


Figure : Architecture of Dalvik and ART

1.6 Installation and configuration of Android SDKs and Eclipse IDE

1. Eclipse:- Eclipse IDE is world's largest development IDE. From the moment the Android SDK was released, Eclipse has been the standard IDE for Android development and remains so to this day. From our perspective, these are the main reasons for this:

   - With the release of the Android SDK, Google immediately made available the extensive Android Development Tools (ADT) plugin for Eclipse.
   - ADT is used and maintained by the Google Android platform developers themselves.
   - Eclipse/ADT, like the Android SDK itself, is open source and available free of charge.

2. Eclipse Home and Download Area:-

- http://www.eclipse.org
- http://www.eclipse.org/downloads/

3. Android Development Tools Plugin for Eclipse ADT:-

   - http://developer.android.com/sdk/eclipse-adt.html

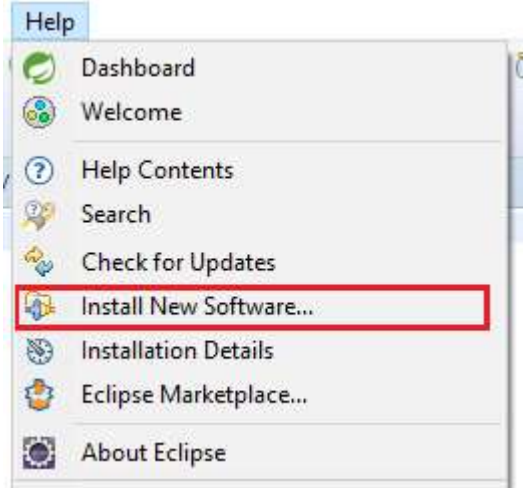
4. Official Google ADT Eclipse Update Site:-

   - https://dl-ssl.google.com/android/eclipse/
   - http ://dl-ssl.google.com/android/eclipse/

1.6.1 Their integration using ADT Plug-in

To Integrate ADT plug-in to Eclipse IDE the following Steps needs to perform:-

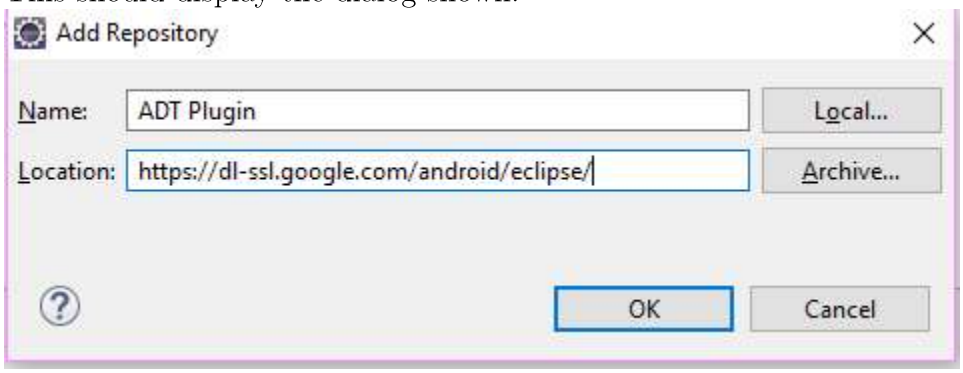
- To install the Eclipse ADT plugin, go to the Eclipse Help
- Click on Install New Software Menu as shown below:



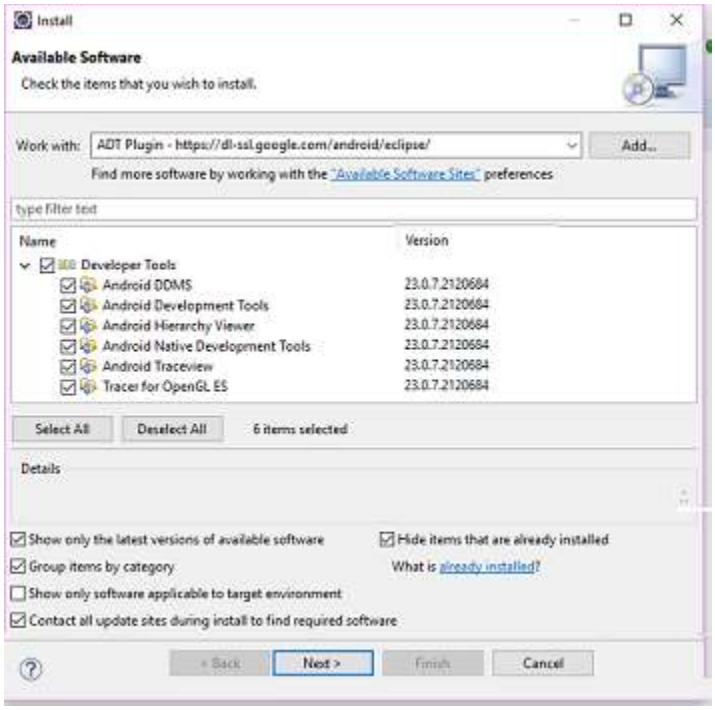
  and click the add (a New Software site) Button.

- Enter your own preferred Name and in the Location use either of the following resource locators. https://dl-ssl.google.com/android/eclipse/ OR http://dl-ssl.google.com/android/eclipse/

- This should display the dialog shown:



- The Eclipse Available Software (dialog shown) displays with the ADT listed. Select all the tools and click next or Finish as shown below:



- Note:- Continue with the setup work flow until the installation is complete.

1.6.2 Running an Emulator

With your project selected in the Package Explorer pane, click the green "play" button in the Eclipse toolbar to run your project. The first time you

do this, you will have to go through a few steps to set up a "run configurations", and so Eclipse knows what you want to do. First, in the "Run As" list, choose "Android Application": If you have more than one emulator AVD or device available, you will then get an option to choose which you wish to run the application on. Otherwise, if you do not have a device plugged in, the emulator will start up with the AVD you created earlier. Then, Eclipse will install the application on your device or emulator and start it up as shown in the figure below:-

1.7 Using ADB Command Line Interface

Android Debug Bridge (adb) is a versatile command-line tool that lets you communicate with a device. The adb command facilitates a variety of device actions, such as installing and debugging apps, and it provides access to a Unix shell that you can use to run a variety of commands on a device. It is a client-server program that includes three components:

- **A client**, which sends commands. The client runs on your development machine. You can invoke a client from a command-line terminal by issuing an adb command.
- **A daemon (adbd)**, which runs commands on a device. The daemon runs as a background process on each device.
- **A server**, which manages communication between the client and the daemon. The server runs as a background process on your development machine.

This makes at least two assumptions:

1. You have the Android SDK installed.
2. You have an Android emulator (or physical device) running.

At the adb shell prompt we can enter a variety of commands to interact with your Android emulator or device. **The brief list of frequently used commands is:**

adb devices	list the installed devices
adb pull ¡remote¿¡local¿	copy a file or directory to the emulator or device
adb install Foo.apk	install the APK file/app
adb install -r Foo.apk	update the already installed app
adb uninstall ¡pkg¿	uninstall the app given by pkg ls list files and directories
logcat	view the system's log buffers
reboot	reboot the device

This command lets us uninstall an Android application, where the argument given to the uninstall command is the root package name of the app:

$adb uninstall com.MyPackageName