Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

Mobile Operating Systems and Application Development Platforms: A Survey

Okediran O. O.
Department of Computer Science & Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria
Email: ookediran@lautech.edu.ng
Arulogun O. T.
Department of Computer Science & Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria
Email: otarulogun@lautech.edu.ng
Ganiyu R. A.
Department of Computer Science & Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria
Email: raganiyu@lautech.edu.ng
Oyeleye C. A.
Department of Computer Science & Engineering, Ladoke Akintola University of Technology, Ogbomoso, Nigeria
Email: caoyeleye@lautech.edu.ng

----------ABSTRACT----------

Earlier mobile communication technologies were dominated by vertically integrated service provision which are highly bounded mainly to voice and short message services that are organized in a monopolistic competition between few mobile virtual network operators, service providers and enhanced service providers. In the recent years, however, radical change driven by advancements in technology, witnessed the introduction and further development of smartphones where the user can get access to new applications and services by connecting to the device manufactures' application stores and the like. These smartphones have added many features of a full-fledged computer: high speed processors, large storage space, multitasking, high-resolution screens and cameras, multipurpose communication hardware, and so on. However, these devices market is dominated by a number of different technological platforms, including different operating systems (OS) and application development platforms, resulting in a variety of different competing solutions on the market driven by different actors. This paper detailed a review and comparative analysis of the features of these technological platforms.

Keywords - Mobile OS**,** Android, iOS, Windows Phone, Black-Berry OS, webOS and Symbian.

Date of Submission: 26 July 2014 Date of Acceptance: 17 August 2014

1. INTRODUCTION

Mobile communication devices have been the most adopted means of communication both in the developed and developing countries with its penetration more than all other electronic devices put together [16]. Every mobile communication device needs some type of mobile operating system to run its services: voice calls, short message service, camera functionality, and so on. The earlier mobile operating systems were fairly simple, since the capabilities of the phones they supported were limited. However, modern smartphones have added many of the features of a full-fledged computer which includes high speed central processing units (CPU) and graphics processing unit (GPU), large storage space, multitasking, high-resolution screens and cameras, multipurpose communication hardware and so on [15]. Modern mobile operating systems combine the features of a personal computer operating system with other features, including a touch screen, cellular, Bluetooth, Wi-Fi, global positioning system (GPS) mobile navigation, video camera, speech recognition, voice recorder, music player, near field communication and infrared blaster [7]. Mobile operating systems have had to grow in sophistication to support these features.

Furthermore, modern smartphones are designed to allow external developers to write software for these devices. With this feature, users can get access to new applications and services by connecting to the device manufactures' applications stores e.g. Apple's 'App Store', Google's 'Android Market', Blackberry's 'App World', Nokia's 'OVI Store', Palm's 'Palm App Catalog', Windows Mobile's 'Windows Market place' and so on (Figure 1). This has enabled these mobile devices to reap the advantages of the convergence process and brought advanced internet applications and services to these mobile devices.

However, the device market is dominated by a number of different technological platforms, including different operating systems and applications development platforms, resulting in a variety of different competing solutions on the market driven by different actors. This fragmentation of technological platforms and standards are seen as a barrier for development of contents and services, which locks the users to specific technologies or puts an immense load to the content and service provides to adopt their content /services to all these platforms.

Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

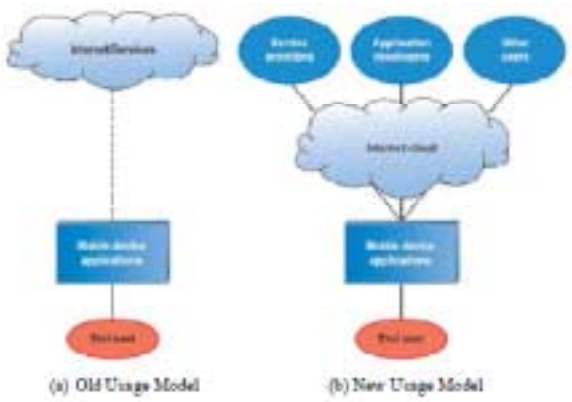

Figure1: High-level Usage Models of Mobile Devices [11]

The aim of this paper is to give a review and comparative analysis of the features of the six most popular mobile operating systems (Android OS, iOS, Windows Phone, Blackberry OS, webOS and Symbian OS) and user interface toolkits most frequently used to develop client applications (Qt, Java 2 Micro Edition, and Silverlight).

2. MOBILE OPERATING SYSTEMS

A mobile operating system (Mobile OS) is a software platform on top of which other programs called application programs, can run on mobile devices such as personal digital assistant (PDA) , tablets, cellular phones, smartphones and so on [3]. Over the years, Mobile OS design has experienced a three-phase evolution: from the PC-based operating system to an embedded operating system to the current smartphone-oriented operating system in the past decade. Throughout the process, Mobile OS architecture has gone from complex to simple to something in-between. The evolution process is naturally driven by the technology advancements in hardware, software, and the Internet [11]:

i. *Hardware:* The industry has been reducing the factor size of microprocessors and peripherals to design actual mobile devices. Before the form factor size was reduced enough, the mobile device could not achieve both small size and processing capability at the same time. We had either a PC-sized laptop computer or a much weaker personal data assistant (PDA) in phone size. Mobile operating systems for PDAs usually did not have full multitasking or 3D graphics support. Features like sensors, such as accelerometers, and capacitor-based touch screens were not available in the past mobile operating systems.

ii. *Software:* With a laptop computer, the software is mainly focused on the user's productivity, where support for keyboard and mouse that have precise inputs are essential. The software for a personal data assistant, as its name implies, helps the user to manage personal data such as contacts information, e-mail, and so on. The mobile operating systems were not designed for good responsiveness or smoothness with a rich user interface (UI) including both touch screen and other sensors.

ii. *Internet:* Along with Internet development, especially after Web 2.0, there is abundant information in the network waiting to be searched, organized, mined, and brought to users. People are increasingly living with the Internet instead of just browsing the Web. More and more people are involved in the development, including information contribution, application development, and social interactions. The mobile operating systems cannot be self-contained, but have to be open systems.

The aforementioned technological advancements have resulted in a variety of different competing mobile operating system solutions on the market driven by different actors. Some of these actors includes Google's Android, Apples' iOS, Nokia's Symbian, RIM's BlackBerry OS, Samsung's Bada, Microsoft's Windows Phone, Hewlett-Packard's webOS, and embedded Linux distributions such as Maemo and MeeGo to mention but a few. The following sub-sections review six of the most popular mobile operating systems.

2.1 Android OS

Android OS for mobile devices is developed by the Open Handset Alliance, which is led by Google. Google unveiled the Android distribution in November 2007. Most of the Android core is released under the open-source Apache License but a large amount of software on Android devices (such as such as Play Store, Google Search, Google Play Services, Google Music, and so on) are proprietary and licensed [15].

As of 2011, Android has the largest installed base of any mobile OS and as of 2013, its devices also sell more than Windows, iOS and Mac OS devices combined [13]. As of July 2013 the Google Play store has had over 1 million Android apps published, and over 50 billion apps downloaded (PHONEARENA, 2014). A developer survey conducted between April and May 2013 found that 71% of mobile developers develop for Android [4].

Android uses a Linux kernel with higher-level APIs written in C and applications are normally programmed in Java and run with the Dalvik virtual machine (DVM) using just-in-time compilation to translate Java byte code into Dalvik dex-code [3]. This combination brings up some secure features, like efficient shared memory management, preemptive multitasking, Unix user identifiers (UIDs) and file permissions with the type safe concept of Java. Every Android application runs in a separate process under a unique UID with distinct permissions, which means that applications can typically not read or write each other's data or code. The kernel sandboxes applications from each, so that resource and data must be share explicitly. To make a resource share between applications possible, the permissions which are required must be declare statically at the time the application is installed. The Android system prompts the user for consent at this time; a mechanism for granting permission dynamically at runtime is not possible and would lead to an increase of security transparency [20].

Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

Figure 2 below depicts android OS architecture.

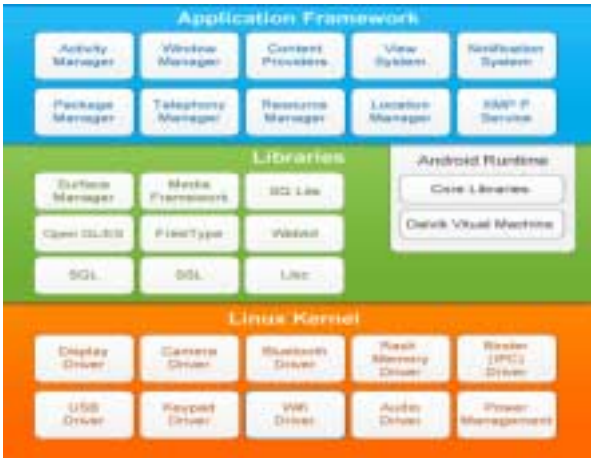

Figure 2: Google's Android OS Architecture [3]

The Android platform contains the following layers [3]:

i. *Linux Kernel*: Android relies on Linux for core system services such as security, memory management, process management and so on.
ii. *Android Runtime*: it provides a set of core libraries which supports most of the functionality in the core libraries of Java. The Android Virtual Machine known as Dalvik VM relies on the Linux kernel for some underlying functionality.
iii. *Libraries*: Android includes a set of C/C++ libraries. These libraries are exposed to developers through the Android application framework. They include media libraries, system C libraries, surface manager, 3D libraries, SQLite and so on.
iv. *Application Framework*: it provides an access layer to the framework APIs used by the core applications. It allows components to be used by the developers.

2.2 iOS

iOS (previously iPhone OS) is a mobile operating system developed by Apple Inc. and distributed exclusively for Apple hardware [6]. It is the operating system that powers iPhone, iPad, iPod Touch, and Apple TV. It is closed source and proprietary and built on open source Darwin core OS. iOS promoted a new style of user interaction for small screen, limited input devices, specifically, direct manipulation. Touch-based gestures like swipe, tap, tap and hold, and pinch are used to control on-screen interface elements, and to perform interface operations. Accelerometers support additional physical gestures like shaking and rotating the orientation of the device [15].

iOS is derived from Mac OS X, and shares its basic Darwin foundation, an open source POSIX-compliant UNIX OS. In this sense iOS can be considered a variant of UNIX. iOS is made up of four abstraction layers: Core OS, Core Services, Media, and Cocoa Touch6 [15, 20]:

i. *Core OS:* The kernel of the operating system, which includes basic low-level features: system support—threads, sockets, IO, DNS, math, memory—general security services—certificates, private/public keys, encryption—external hardware management, bluetooth, and sound and image processing.
ii. *Core Services:* Fundamental system-services, which are subdivided in different frameworks and based on C and Objective C. It includes basic application services, including accounts, contacts, networking, data management, location, calendar events, store purchasing, SQLite, and XML support.
iii. *Media Layer:* Considers the high-level frameworks, which are responsible for using graphic (support for 2d and 3d graphics), audio- and video technologies.
iv. *Cocoa Touch:* The UIKIT, which is an Objective- C based framework and provides a number of functionalities, which are necessary for the development of an iOS Application like the User Interface Management. It also includes APIs for building applications—multitasking, touch input, notifications, interface views, and access to device data. Figure 3 below depicts the ioS architecture.

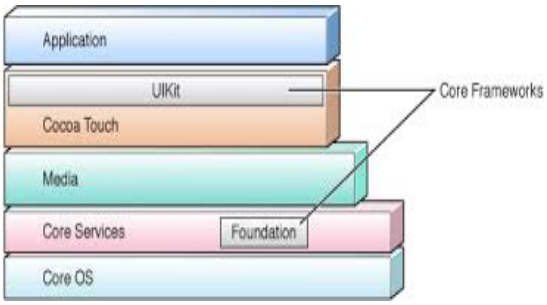

Figure 3: Apple's iOS Architecture [3]

Like in the sub-section 2.1 described above, iOS also uses a similar sandboxing model [1]. Furthermore applications must be signed with an issued certificate. This ensures that application have not been manipulated and ensures the runtime to check if an application has not become un-trusted since it was last used. Uneven Android applications, iOS applications can only be signed with an official certification [2].

2.3 Windows Phone

Windows Phone is a proprietary smartphone operating system developed by Microsoft [10]. It is the successor to Windows Mobile, although it is incompatible with the earlier platform [24]. It was launched in 2010 under the name Windows Phone 7. Various hardware manufacturers including HTC, Samsung, LG, and Nokia are developing Windows Phone devices. In February 2011 Nokia and Microsoft announced that Windows Phone 7 would be the primary OS for all future Nokia smartphones. Windows Phone 7 received a major upgrade (7.5 Mango) in

Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

February 2011, adding features that had been missing in the original release. The second generation Windows Phone 8 was released in October 2012 [15].

Windows Phone 7's architecture required a hardware layer that meets Microsoft's minimum system requirements: an ARM7 CPU, a DirectX 9-capable GPU, 256MB RAM and 8GB of flash memory, a 5-megapixel camera, a multi-touch capacitive display, an A-GPS, an accelerometer, a compass, proximity and light sensors, and six physical buttons: back, start, and search; camera, power/sleep, and volume [23]. The Windows Phone kernel handles low-level device driver access as well as basic security, networking, and storage. Three libraries: an App Model for application management, a UI model for user-interface management, and a Cloud Integration module for web search via Bing, location services, push notifications, and so on sit above the kernel [15]. Application-facing APIs include Sliverlight, XNA, HTML/JavaScript and the Common Language Runtime (CLR) that supports C# or VB .Net applications. The kernel itself is a proprietary Windows OS design for embedded devices that combines Windows Embedded CE 6.0 R3 and Windows Embedded Compact 77. Windows Phone 8 replaced the Windows CE kernel with one based on Windows NT. This is meant in part to mimic the Windows 8 desktop OS, allowing for easier porting of applications between the two operating systems. Figure 4 below depicts the windows phone architecture.

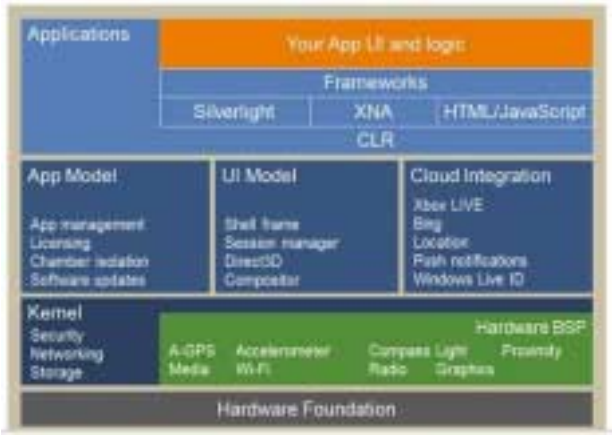

Figure 4: Microsoft's Windows Phone Architecture [23]

2.4 Blackberry OS

BlackBerry OS is developed by Research in Motion (RIM) for their BlackBerry smart phones and tablet devices [20]. BlackBerry OS 1.0 debuted in January 1999 as part of BlackBerry's pager/email devices. One of the main strengths of BlackBerry devices is their ability to handle corporate email. BlackBerry OS supports the Java Mobile Information Device Profile (MIDP) and the Wireless Application Profile (WAP). These protocols are used to synchronize through a BlackBerry Enterprise Server (BES) with push-based calendar, task, contact, email, and note exchange. BES provides the capacity, security, remote wipe, and other features that corporations require for mobile devices that access internal networks and/or corporate data. BlackBerry OS also provides the BlackBerry Internet Service (BIS), a client-specific method to allow Internet access for individual users [15].

This allows consumer customers to access personal email, browse the web, and so on. BlackBerry OS originally supported applications written in C++. One type of application is a Mobile Data Services (MDS) runtime, which is a container for processing and display data, usually pushed from a user's corporate system. Java ME was also supported, and was used to build applications with access to OS APIs that provide access to standard UI widgets and different OS services [15]. Both BlackBerry OS 6 and OS 7 were designed to encourage application development. Programming is now done in Java for phones, and in C++ or web-based languages for the PlayBook tablet. OS-supported APIs include browsing, mail, phone, PDA apps, LDAP, UI, http, math, cryptography, and so on. A C++ Native Development Kit was recently made available to support development on BlackBerry's PlayBook tablet OS. BlackBerry PlayBook OS 1.0, which is available only on the PlayBook, switched to be QNX-based. QNX is a UNIX microkernel that was originally developed in the 1980s and later re-purposed for embedded devices. RIM purchased QNX in April 2010, with plans to transition its upcoming smartphones to OS 10 and QNX. Blackberry 10, together with the Z10 and Q10 Blackberry smartphones, were released in January 2013 [15].

2.5 webOS

webOS is a proprietary mobile operating system running on the Linux kernel, initially developed by Palm, which launched with the Palm Pre [8]. The webOS interface revolves around "cards," individual applications that are presented one-at-a-time and can be scrolled through horizontally like a deck of cards to move applications from the foreground to the background. On startup, a launcher screen with a grid of icons is presented, together with a quick-launch bar holding commonly-used applications. The UI supports standard touch and gesture commands like tap, swipe, and pinch. webOS's Core OS is built on a Linux 2.6 kernel, with device drivers, an ext3 file system, network communication, and bluetooth. Above this sits the UI System Manager, which is responsible for window, UI, and application management. The Mojo JavaScript framework provides application-facing APIs, and the webOS Services Manager offers access to location, phone, camera, and so on. webOS applications are programmed in HTML, CSS, and JavaScript, and use Mojo and webOS services for UI and OS support [15].

Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

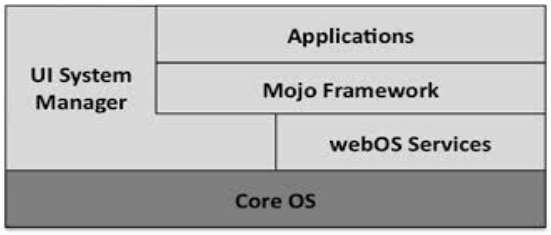

Figure 5: Simplified webOS Architecture [17]

2.6 Symbian OS

Symbian is a mobile device OS developed by Nokia. It was originally the EPOC graphical operating system for PSION portable devices [12]. In 1998 PSION, Nokia, Ericsson, and Motorola formed Symbian OS. Currently, however, the Symbian Foundation is run and maintained by Nokia alone, providing access to Symbian through standard licensing agreements.

The original Symbian OS was divided into two parts: a core OS that supported a Device Family Reference Design (DFRD) and a UI built on the DFRD. This allowed different UIs to be built for different types of devices, or for different manufacturers' handsets, but with a common OS core. Examples included the Pearl UI used by Nokia and the Quartz UI used by Ericsson. This strategy was later abandoned and different UIs were spun off to different companies.

The latest version of Symbian is Symbian OS 9.5, released in March 2007. Follow on versions include Symbianˆ1, Symbianˆ2, and Symbianˆ3, which was released in 2010. Symbianˆ3 includes modern mobile OS technologies like 2D and 3D graphics acceleration, touch-based interaction, and UI widgets. In May 2011, an update for Symbianˆ3, Symbian Anna, was officially announced, followed by Nokia Belle (previously Symbian Belle) in August 2011. Symbian OS follows a familiar architecture. It is built on a nanokernel/microkernel core with basic localization and screen drivers. Base services sit above the kernel, and include low-level libraries, media frameworks, XML, file system management, and hardware abstraction [15]. OS services provide communication, telephony, networking, multimedia, and graphics. These support an Application Services layer with application-facing APIs for development and an interface layer to manage the UI. A JVM (Java ME) is also included above the OS services layer. Nokia provides SDKs for Symbian development that supports a variety of languages, including C++ and Java.

3 Mobile Application Development

Mobile application development is the process by which application software is developed for low-power handheld devices, such as personal digital assistants, enterprise digital assistants or mobile phones. These applications can be pre-installed on phones during manufacturing, downloaded by customers from various mobile software distribution platforms, or delivered as web applications using server-side or client-side processing (e.g. JavaScript) to provide an "application-like" experience within a Web browser [21].

As part of the development process, mobile user interface (UI) design is also an essential in the creation of mobile apps. Mobile UI considers constraints & contexts, screen, input and mobility as outlines for design. The user is often the focus of interaction with their device, and the interface entails components of both hardware and software. User input allows for the users to manipulate a system, and device's output allows the system to indicate the effects of the users' manipulation. Mobile UI design constraints include limited attention and form factors, such as a mobile device's screen size for a user's hand(s). Mobile UI contexts signal cues from user activity, such as location and scheduling that can be shown from user interactions within a mobile application. Overall, mobile UI design's goal is primarily for an understandable, user-friendly interface. The UI of mobile apps should: consider users' limited attention, minimize keystrokes, and be task-oriented with a minimum set of functions.

This section, reviews briefly user interface toolkits most frequently used to develop client applications for the mobile platforms considered in section 2 above.

3.1 Qt

Qt is an open-source cross-platform user interface toolkit [19]. Qt is written in C++ and offers a powerful extension of C++ syntax in form of signals and slots (which are used for event generation and consumption by any class descending from Qt root class, QObject), and also meta-object model, which permits querying objects for the properties, signals and slots they support. Qt provides an intuitive C++ class library with a rich set of application building blocks for C++ development. Qt goes beyond C++ in the areas of inter-object communication and flexibility for advanced GUI development [19].

Qt is possible to use for Symbian development (using Eclipse-based Carbon IDE and Qt SDK for Symbian), Maemo/MeeGo development, along with desktop development for a wide range of OSs (Linux, FreeBSD, Windows, Mac OS X). Maemo/MeeGo and recent Symbian releases have Qt libraries pre-installed. For early Symbian versions a Qt installer is available, which ensures all the necessary libraries are set up. With careful development of user interface (using a versatile mechanism of Qt Layouts), it is possible to create different platform versions of the application by simple recompilation. Support of OpenGL ES 2.0 in the QOpenGL library allows for creation of compelling 3D interactive user experiences.

Qt Mobility library is also offered, which provides easy access to the mobility features, such as geolocation (GPS), accelerometer measurements, battery state and device system information [25]. Compared to Java, Qt has the advantage of being compiled directly to the OSs and hence it does not need any 'translating layer' like JVM. This solves the speed and complexities, which have been connected to JVM. Another important factor is the huge investment and focus from Nokia on Qt that is a heavy argument for seeing it to be a valid competitor with other software development tools.

Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

3.2 Java Platform Micro Edition

Java Platform, Micro Edition, or Java ME, is a Java platform designed for embedded systems (mobile devices are one kind of such systems). Target devices range from industrial controls to mobile phones (especially feature phones) and set-top boxes. Java ME was formerly known as Java 2 Platform, Micro Edition (J2ME). Java ME is the smallest addition to the Java family. The other members of the Java family are the Java Platform Standard Edition (Java SE) and the Java Platform Enterprise Edition (Java EE). The former is intended for conventional desktop applications development, while the latter one is specifically intended for building distributed applications with emphasis on the server side development and web applications. Java ME is intended to build applications running on mobiles and other embedded devices [5].

The Java ME architecture defines configurations, profiles and optional packages as elements for building complete Java runtime environments that meet the requirements for a broad range of devices and target markets. Each combination is optimized for the memory, processing power, and I/O capabilities of a related category of devices. The result is a common Java platform that fully leverages each type of device to deliver a rich user experience.

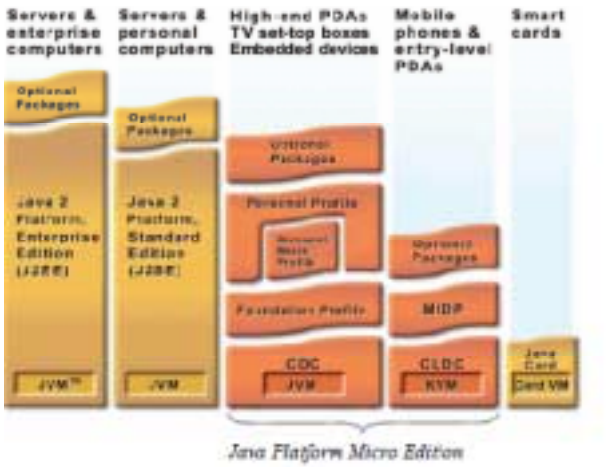

Figure 6: Java ME architecture [9]

In Java ME two different configurations exist [9]:

i. *Connected Limited Device Configuration (CLDC):* CLDC is the smaller of the two configurations, designed for devices with intermittent network connections, slow processors and limited memory – devices such as mobile phones, two way pagers and PDAs. These devices typically have either 16- or 32-bit CPUs and a minimum of 128 KB to 512 KB of memory available for the Java platform implementation and associated applications.

ii. *Connected Device Configuration (CDC):* CDC is designed for devices that have more memory, faster processors, and greater network bandwidth, such as TV set-top boxes, residential gateways, in-vehicle telematics systems, and high-end PDAs. CDC includes a full-featured Java virtual machine, and a much larger subset of the Java SE platform than CLDC. As a result, most CDC-targeted devices have 32- bit CPUs and a minimum of 2MB of memory available for the Java platform and associated applications.

3.3 Silverlight

Microsoft Silverlight is an application framework for writing and running Internet applications. The run-time environment for Silverlight is available as a plug-in for web browsers running under Microsoft Windows and Mac OS X. Silverlight is one of the two application development platforms for Windows Phone, but web pages which use Silverlight cannot run on the Windows Phone or Windows Mobile versions of Internet Explorer, as there is no Silverlight plug-in for Internet Explorer on those platforms [14]. It is also compatible with later versions of Internet Explorer, Mozilla Firefox, and Google Chrome web browsers on Microsoft Windows (except Windows RT) operating systems, with Firefox and Safari under Mac and OS X, and with mobile devices using the Windows Mobile and Symbian (Series 60 platforms) [22]. Silverlight provides a retained mode graphics system similar to Windows Presentation Foundation (WPF), and integrates multimedia, graphics, animations and interactivity into a single run-time environment. In Silverlight applications, user interfaces are declared in Extensible Application Markup Language (XAML) and programmed using a subset of the .NET Framework. XAML can be used for marking up the vector graphics and animations. Silverlight applications can be written in any .NET programming language. As such, any development tools which can be used with .NET languages can work with Silverlight, provided they can target the Silverlight CoreCLR for hosting the application, instead of the .NET Framework CLR.

4.0 Conclusion

Recent years have witnessed the emergence of a number of competing technological platforms for mobile communication devices where device manufacturers go beyond the terminal market and take share in the value creation at services and content level. The paper has been able review and compare the major mobile operating systems and user interface toolkits from technological and developers' standpoints. With respect to mobile operating systems, Symbian has for long time been the dominating technology, however, it seems that in the transition to the smartphones other operating systems like Android, iOS, Blackberry OS and Windows phone are in the lead presently. Specifically, the Google's Android initiative of developing an OS which can run on all mobile devices has many made the Android the most used and popular mobile operating the world over.

With respect to software platforms, Java ME has been the far dominating platform for mobile devices. However, it has been heavy and slow to work with. In the recent years Qt is getting more attention and focus as it is highly platform agnostic and does not have the extra layer of complexity.

Volume: 6 Issue: 1 Pages: 2195-2201 (2014) ISSN : 0975-0290

An analysis of the various mobile operating systems reviewed with their corresponding deployed software development platform is depicted in Table 1 below.

	Android OS	iOS	Windows Phone	BlackBerry OS	webOS	Symbian OS
Deployed Software Development	Java, C/C++, Python, Lua	Objective-C, C/C++	.NET Languages (C#, VB.NET, F#, etc.), VB, C++, JScript	Java	JavaScript	C, C++,Qt, Java ME, Python, Ruby, Flash Lite
Market Size	Very Large	Large	Medium	Small	Very Small	Very Small
SDK Platform	Windows XP, Vista and 7, Linux, Mac OS X	Mac OS X Snow Leopard 10.6.4	Windows Vista & 7	32-bit Windows XP, Vista and 7	OS X, Ubuntu, Windows	Windows XP Professional SP2, Vista & 7 for some SDKs
Openness to Application Developers	Very High	Very Low	Medium	Low	Very High	High
OS Family	Linux	Darwin	Windows CE 7 Windows NT 8	QNX	Linux	RTOS
Supported CPU Architecture	ARM, MIPS, x86	ARM	ARM	ARM	ARM	ARM
Application Store	Android Market	iPhone App Store	Windows Market Place	Blackberry App World	Palm App Catalog	Nokia Ovi Store
Future Prospect	Very High	High	Medium	Low	Low	Low

*RTOS: Real-time Operating System

*MIPS: Microprocessor without Interlocked Pipeline Stages

Table 1: Comparison of Major Mobile Operating Systems

References

[1]. Apple1, (2014): "The Application Runtime Environment." Available: http://developer.apple. com/ library /ios/ #documentation/iphone/conceptual/ iphoneosprogrammingguide/ Runtime Environment/ RuntimeEnvironment.html

[2]. Apple2, (2014): "iphone in Business Security Overview." Available: http:// images. apple. com / iphone/business/docs/iPhone Security.pdf

[3]. CMER, (2014): "Mobile Operating System" Centre for Mobile Education and Research

[4]. DEVECO, (2013) "Developer Economics Q3 2013 Analyst Report" available at http://www.visionmobile .com/DevEcon3Q13

[5]. Fitzek F. and Reichert F. (2007): "Mobile Phone Programming and its Application to Wireless Networking" available at http://www.springerlink. com/content/978-1-4020-5968-1

[6]. Gartner (2010): "Gartner Says Worldwide Mobile Phone Sales Grew 35 Percent in Third Quarter 2010; Smartphone Sales Increased 96 Percent" Gartner, Inc. available at http://www.gartner.com/it/ page.jsp?id= 1466313

[7]. Holwerda T., (2013): 'The Second Operating System Hiding in Every Mobile Phone" available at http://www.osnews.com /story/27416/ The_second_ operating_system_ hiding_ in_every_mobile_phone

[8]. HP, (2010): "HP Confirms Discussions with Autonomy Corporation plc Regarding Possible Business Combination; Makes Other Announcements." Available at http://www.hp.com/ hpinfo/newsroom/ press/2011/ 110818b.html?mtxs=rss-corp-news.

[9]. Java, (2014): "Java 2 Platform, Micro Edition" Java 2 Platform, Micro Edition Datasheet. Available at www.sun.com/software

[10]. Koh, D., (2010): "Q&A: Microsoft on Windows Phone 7" CNET Asia, CBS Interactive.

[11]. Li X., Wang Y., Wu J., Jiang K. and Liu B., (2012): "Mobile OS Architecture Trends" Intel Technology Journal, volume 16, issue 4, pp 178-198.

[12]. Lunden, I. (2011): "Symbian Now Officially No Longer Under The Wing of Nokia," Available at www.moconews.net.

[13]. Mahapatra, L. (2013): "Android Vs. iOS: What's the Most Popular Mobile Operating System in Your Country?" available at http://www.ibtimes.com/ android-vs-ios-whats-most-popular-mobile-operating-system-your-country-1464892.

[14]. Microsoft, (2014): "Silverlight 5 System Requirements - Compatible Operating Systems and Browsers". Available at http://www.microsoft.com/ getsilverlight /locale/en-us/html/ installation-win-SL5.html.

[15]. NCSU, (2014): "Mobile Operating System" available at http://www.csc.ncsu.edu/ faculty/ healey/csc563 /notes/ch-06.pdf

[16]. Okediran O. O., Omidiora E. O., Olabiyisi S. O. and Ganiyu R. A. (2013): " An M-voting System Framework for Electronic Voting", Proceedings of the Second International Conference on Engineering and Technology Research., 2:241-245.

[17]. Palm, (2014): "Overview of webOS - Palm webOS Architecture" available at https://developer.palm.com /content/resources/develop/overview_of_webos/overview_of_ webos_palm_webos_architecture.html

[18]. PHONEARENA, (2014): "Android's Google Play beats App Store with over 1 million apps, now officially largest" available at http://www. phonearena.com/news/Androids-Google-Play-beats-App-Store-with-over-1-million-apps-now-officially-largest_id45680

[19]. QT, (2009): "Qt 4.4 Whitepaper" available at http://qt.nokia.com/files/pdf/qt-4.4-whitepaper

[20]. Renner T. (2014): "Mobile OS - Features, Concepts and Challenges for Enterprise Environments" SNET Project Technische Universit¨at Berlin

[21]. SDDP, (2014): "An Overview of Mobile Development in the Context of Current Technology" Software Development Discussion Paper, available at http://www .bbconsult.co.uk/Mobile- Web-Software-Development.aspx

[22]. Silverlight, (2008): "FAQ: Silverlight for Mobile". Available at http://silverlight.net/learn/ mobile. Aspx

[23]. Windows, (2011): "Windows Phone 7 Platform Introduced to iPhone Application Developers", available at http://windowsphone.interoperability bridges.com/articles/chapter-1-windowsphone-7-platform-introduced-to-iphone-application-developers.

[24]. Ziegler, C., (2010): "Microsoft talks Windows Phone 7 Series development ahead of GDC: Silverlight, XNA, and no backward compatibility". Engadget, AOL.

[25]. Zubarev P. and Krinkin K. (2010): "QtMobility: Qt API Extensions for Mobile Development" 7th Conference of Open Innovations Framework Program FRUCT. Saint-Petersburg, Russia, available at http://osll.spb.ru/repositories/entry/osll/events/fruct7/ fruct7_qtmobility .odp?rev=68