Mobile application platform heterogeneity: Android vs Windows Phone vs iOS vs Firefox OS

Tor-Morten Grønli
Norwegian School of IT
0185 Oslo, Norway
tmg@nith.no

Jarle Hansen
Systek AS
Oslo, Norway.
jarle.hansen@systek.no

Gheorghita Ghinea
Brunel University
London, UK
george.ghinea@brunel.ac.uk

Muhammad Younas
Oxford Brooks University
Oxford, UK
m.younas@brookes.ac.uk

***Abstract*— Modern smartphones have a rich spectrum of increasingly sophisticated features, opening opportunities for software-led innovation. Of the large number of platforms to develop new software on, in this paper we look closely at three platforms identified as market leaders for the smartphone market by Gartner Group in 2013 and one platform, Firefox OS, representing a new paradigm for operating systems based on web technologies. We compare the platforms in several different categories, such as software architecture, application development, platform capabilities and constraints, and, finally, developer support. Using the implementation of a mobile version of the tic-tac-toe game on all the four platforms, we seek to investigate strengths, weaknesses and challenges of mobile application development on these platforms. Big differences are highlighted when inspecting community environments, hardware abilities and platform maturity. These inevitably impact upon developer choices when deciding on mobile platform development strategies.**

Keywords: Android, Windows Phone, iOS, Firefox OS, Mobile Development Environment, Mobile Computing, Cross Platform Development, Native Apps, Web Apps

I. Introduction

Mobile phone application development has taken a huge step from its first days of development on monochrome screens. Today, sophisticated features are available and there are a large number of platforms to develop new software on. In this paper, we look at three widely used development platforms for pervasive applications: 1) Android, a Linux based operating system from Google; 2) The Windows Phone operating system from Microsoft; 3) The iOS platform from Apple; and one platform representing a new generation: 4) The new web based Firefox OS from Mozilla. Three of these four platforms, 1,2 and 3, were in 2013 identified as market leaders for the smartphone market by Gartner Group [35]. Firefox OS, 4, were recognized as the chiefly representative for a new category of mobile operating systems. We will compare the platforms in several different categories through code examples, focusing on commonly used features such as persistent storage and opening a network connection. The different development tools used for these platforms are also introduced since they are a crucial part of modern software development.

The work presented in this paper follows the research performed by Huebscher et al [14], who back in 2006 remarked that an interesting future pursuit would be to look at Windows Mobile or Linux devices and compare the issues and limitations of the platforms. Accordingly, we focus on how the four different platforms can be used to create pervasive mobile device applications. Indeed, there are very few contributions in this area and what papers have been published target other languages. Although, the topic itself is too wide to be investigated in a single paper alone, in this paper we will start by investigating language central code implementations, software architecture, application development, and developer support – all of these have been identified as key issues in mobile application development [7].

The remainder of the paper is organized as follows. Section II reviews related work and identifies related challenges. In section III we will introduce the four different mobile development platforms. Section IV contains the comparison of the platforms, starting with architectural aspects, then moving onto implementation concerns, and ending with developer perspectives of the different platforms. Finally, conclusions are drawn in section V.

II. Related Work

Both hardware and software on mobile devices have improved considerably in recent years. It is illustrative to remark that when Sun conducted a customer survey in 2001 investigating the envisaged hardware profiles their software platform had to run on [24], they came up with a CPU speed of 50 to 200MHz and >600KB RAM as the typical figures for the next-generation mobile phones. When compared to the new smartphone devices released in 2013 we see an enormous increase in computing power [18]. Typical figures are now given by a CPU speed of 1.6 GHz Quad and 2GB RAM. Moreover, screen sizes have also increased significantly since touch screen devices have become mainstream. For instance, Nokia E61, released in 2006, had a 2.9-inch display, considerably less than that of the Samsung Galaxy S4, released in 2013, with a Full HD Super AMOLED 5.0-inch touch screen display.

Such new smartphones are essentially small and powerful computers and modern networking capabilities of the devices also mean that people stay connected to the network. When combined with the increase in screen sizes and networking capabilities, many new and interesting research topics surface, focusing on issues such as user interface design [25], context-awareness in mobile devices applications [6][17] and mobile service development [4][10][12][15][23].

In work closely related to ours, Harjula et al. [11] highlight technical aspects in mobile phone application programming. Aspects like limited bandwidth and small screens have until recently been major obstacles for building advanced and sophisticated mobile applications. However, with the advent of touch screen technology and unlimited-use subscriptions, such barriers belong increasingly to the past. Moreover, software development kits (SDKs) are becoming standard for all major mobile development platforms as well and all of these factors create opportunities for creating novel mobile and pervasive applications. In this paper, we will focus on how it is possible to build one such application – a tic-tac-toe game - and some of the main differences in the development environments of the various platforms

A. Platform heterogeneity

Java has a vision of supporting multiple platforms, "write once, run everywhere". To this end, in 2000 Sun Microsystems reduced and adapted the standard Java platform to fit on resource-constrained mobile devices [24]. The idea was that the same byte code created from the source code should be able to run on all sorts of mobile device models. Even though in principle this is a good idea, it has caused severe problems and limitations, as we have remarked in previous work [9].

However, according to Gartner [1] it was not Java ME phones but iPhone (Apple) and Android (Google) phones who were the winners in the smartphone market in 2010. In contrast to the Java ME model – which strives to be universal - both of these platforms have their own programming API (Application Programming Interface) and syntax. Although Android provides Java syntax, it uses its own Google libraries and creates byte code that will not run on the standard JVM (Java Virtual Machine). This means that consumers are carrying devices that support different programming languages and developers will usually need to create multiple clients in this heterogeneous environment. However, current research has usually focused on one or two specific platforms [1][6][16]. For instance, Koller et al. [16] look at two platforms, specifically Java ME and Adobe Flex. They undertake a comparison in the context of developing a new game for mobile phones and highlight important challenges such as portability when developing in native code, a factor which we have a closer look at in section 4.1. Indeed, as acknowledged by Heikkinen and Still [12] and Mukhtar et al. [18], developers must realize that the "write once, use everywhere" idea is not possible due to hardware and software platform heterogeneity

B. Mobile Application Development Challenges

There is relatively little published research on comparing mobile development environments. One of the earliest is that of Hall and Anderson [10], who compared the Android and iPhone operating systems, as well as briefly touching on the Symbian operating system, the erstwhile market leader in smartphones. Their comparison is based on core issues for software developers such as market base, ease of use, developer support/tools and technology. In their opinion the Android platform is the most exciting and best-positioned mobile operating system to enable developers to produce new applications.

One of the main inspirations for our work is the research done by Huebscher et al. [2]. They focused on issues surrounding ubiquitous computing development and, in particular, problems relating to Symbian C++ and Java ME. In so doing, they present an in-depth look at the possibilities and limitations of the Symbian Series 60 platform. The main advice they give is that developers should write their application code in Java ME due to the lack of portability in Symbian C++. However, they also point that in certain cases the Java ME API will not have the desired functionality. Moreover, when it comes to the emulator for Java ME they find it both lacking in support of testing and debugging. Nonetheless, tool support, including IDEs (Integrated Development Environments) and emulators, has improved constantly since 2006 when Huebscher et al. [14] published their work. We will, through the description in section IV, present a more up to date feature list and examine the state of the tools and development environments.

More recently, Jobe [15] compares experiences of developing the same mobile application/app (the app tracked the runs of semi-professional Kenyan runners) either in native mode or using a cross-platform development tool. His conclusion was that native apps were preferred if there was going to be interaction between the app and the hardware on the specific device (such as the GPS unit or the camera); otherwise, the app developed using the cross platform tool was just as good as the native one.

In summary, the debate currently in the research and the development community is whether to go down the native application route (i.e. write mobile applications for a specific operating system) or cross-platform (i.e. write applications that in theory work across multiple platforms). Whilst in the former approach the same app would have to be written once for each of the mobile platforms it would want to target, the latter approach usually entails applications running either on the mobile Web browser of the particular platform and mimicking the native app behavior, or using cross-platform mobile development tools, usually employing languages such as HTML5, Javascript and Cascading Style Sheets, to create the apps [2][13][19][20][23]. The last approach, whilst attractive in theory ('write once, run everywhere'), has in practice been beset by problems, made notorious by Facebook's dumping of HMT5 in favor of native apps [3].

Accordingly, in this paper, we focus on native apps and compare our experiences of creating a native tic-tac-toe app across four innovative mobile platforms.

III. Platform Presentation

Our research focuses on four of the main mobile platforms: Android, Windows Phone, iOS and Firefox OS. The next sections will give a short introduction to each of these.

A. *Android*

Google released Android in November 2007, under the framework of the Open Handset Alliance [21], with the goal of being an open source arena for software development on mobile platform. Android is an open source mobile operating system based on the Linux kernel and facilitates developers to write managed code in Java using Google developed Java libraries [8]. The Android platform does not only provide the mobile operating system itself including the development environment, but also provides a custom built virtual machine (Dalvik Virtual Machine) for the applications to run on as well as acting as the middleware between code and operating system [8]. For application development, Android facilitates the use of 2D and 3D graphic libraries, a customized, onboard SQL engine for persistent storage and advanced network capabilities such as 3G, 4G and WLAN (Fig. 1). The API is constantly evolving and the current release (4.4 KitKat) [26] is a huge increment compared with number of available features from release 1.0. Since Android is an open source mobile operating system, the community is welcomed to collaborate in the evolvement of the programming environment, the operating system and the API. Development tools for Android include the Eclipse and IntelliJ IDEA.

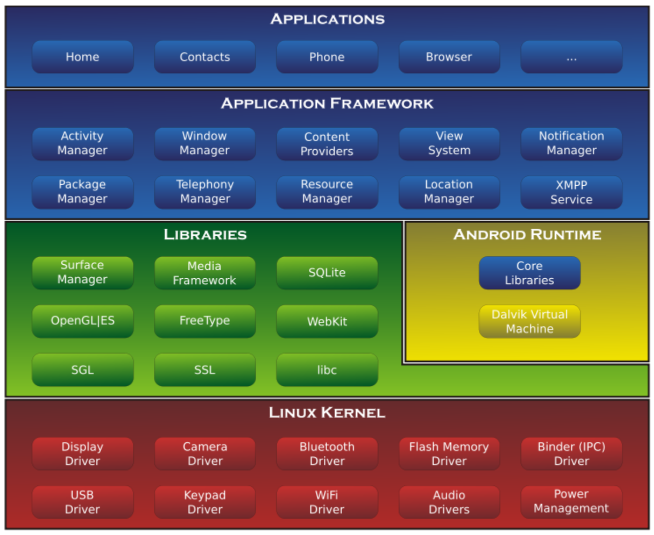

Fig. 1. Android Architecture [26]

B. *Windows Phone*

Previously, the mobile operating system created by Microsoft was called Windows Mobile. After the changes introduced by Apple (iOS) and Google (Android) in 2007, Microsoft decided to take a new direction and created Windows Phone. Similar to other alternatives, such as iOS and Android, Windows Phone is an operating system for smartphones. It is usually used on touch screen devices, and offers functionality such as networking, sensors and camera integration.

Programs for Windows Phone 7 are written in .NET managed code. Managed code is code written in languages that are available for use with the Microsoft .NET Framework, for example C#. One of the benefits is that many of the error-prone and often complex tasks, such as type safety checking, memory management and destruction of unneeded objects, are taken care of [32].

Windows Phone 7 supports two popular programming platforms, namely *Silverlight* and *XNA*. *Silverlight* is an evolution of the Windows Presentation Foundation (WPF). It provides developers with the ability to create sophisticated user interfaces. The second platform, *XNA*, is Microsoft's game platform. It supports both 2D and 3D graphics [22].

Development for Windows Phone is done in Visual Studio [37]. There is a range of various editions of Visual Studio, ranging from the free Visual Studio Express to the Ultimate edition. Although the Express edition if enough to get started, the limitations quickly get in the way of productivity. For example, no support for plugins is one of the main limitations. There are two languages that can be used to write programs for Windows Phone, 1) *Visual Basic .NET* and 2) *C#*. We will focus on the C# language in this paper. We chose to use this language because we were more familiar with it and also we found more resources, in books and on the Internet, for Windows Phone development with C#. Programs created for Windows Phone are packaged into XAP files, which is the Silverlight application package [31].

According to Gartner [35], Microsoft currently occupies the 3rd place in regards to market share (second quarter of 2013). For the first time Microsoft has a larger market share compared to Blackberry. Even with the recent increase in popularity, the Windows Phone platform is still a relatively small player with a 3.3% market share. The step up of the iOS (14.2%) and Android (79.0%) is considerable. However, it will be interesting to see how the acquiring of Nokia [36] will affect the further development of Windows Phone and the mobile devices.

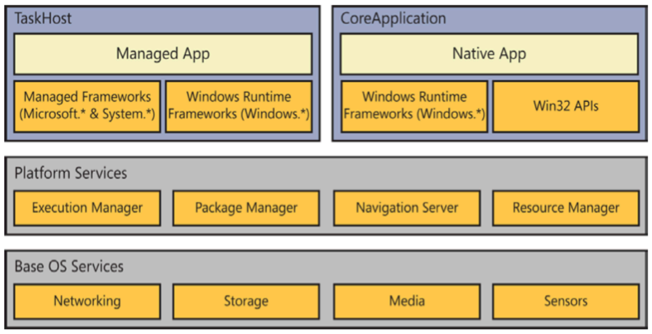

Fig. 2. Windows Phone Architecture [33]

C. iOS

iOS is the operating system for several Apple devices (Fig. 3), one of the most important of which is the iPhone. The iPhone was released in 2007 and changed the smartphone market [34]. It included a large touch screen and, at least for that time, impressive hardware specifications [27].

Applications for iOS are written in Objective-C using the Cocoa Touch library. Objective-C is an extension to the C language, while Cocoa Touch is a collection of classes [5]. While C# and Java (used for Android and Windows Phone development) are fairly similar in syntax, the Objective-C library provides a different alternative.

Objective-C, as the name implies, supports object-oriented programming. The language and platform has continuously improved over the years, and one especially noteworthy change came with the introduction of ARC (Automatic Reference Counting) [28]. This provided automatic memory management and meant that the amount of boilerplate code is reduced and in general memory leaks are less common.

Development for iOS requires a computer running Mac OS. The application usually used to write iOS applications in is Xcode [29]. It includes a powerful editor, as well as an analysis tool, iOS simulator and the SDK.

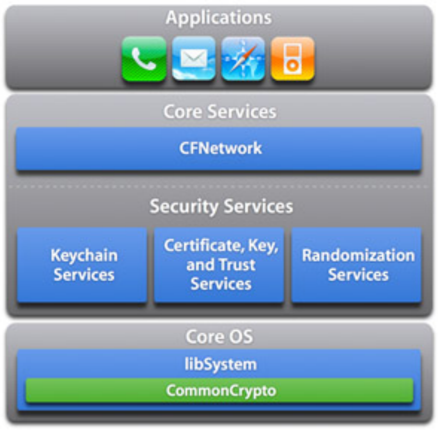

Fig. 3. iOS Architecture [27]

D. Firefox OS

Firefox OS (Fig. 4) represents a new generation of mobile operating systems, namely a web based OS [38]. It is designed based upon open standards and approaches from HTML5 applications, JavaScript and web APIs. This approach brings open web APIs communicating directly with cellphone hardware and it also features a direct link to the web-based application marketplace. Firefox OS was first demonstrated public in early 2012, running on an Android based phone. Later it has been demonstrated running on a Raspberry Pie, and Mozilla launched commercial phones together with ZTE in early 2013[39].

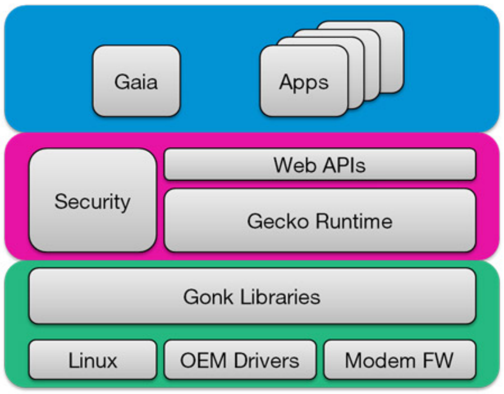

Fig. 4. Firefox Architecture [30]

IV. Platform Comparison

In this section we described the actual comparison of the four mobile application development platforms and show how they relate through a common example implemented across all environments. We start off by detailing the implemented scenario, before we present our findings and finally ending this section with a discussion.

A. Comparison Scenario

To investigate software architecture, application development, and developer support, all of which have been identified as key issues in mobile application development, we needed a common scenario implemented across all platforms. Following the results and practice by Gavalas and Economou [7], we chose the implementation of a game as scenario for comparison. The game chosen was tic-tac-toe, which is suitable because it covers all major aspects of application development. This implementation let us compare all four platforms in terms of technical functionality, APIs, development effort, development support and deployment to live devices. Figures 5 to 8 show the game running on the four platforms.

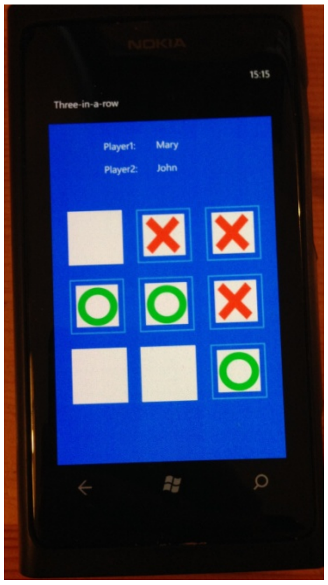

Fig.5. Windows Phone Client (Nokia Lumia 800)

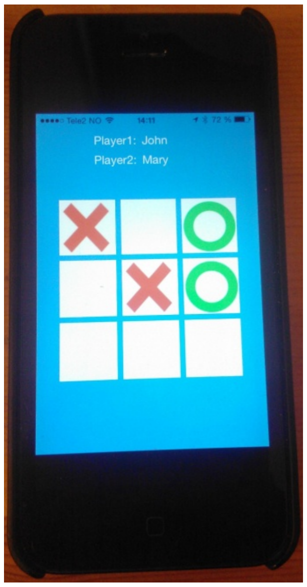

Fig. 6. iOS Client (iPhone 5)

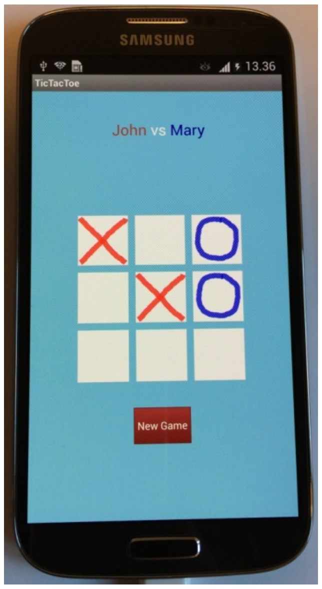

Fig. 7. Android Client (Samsung Galaxy S4)

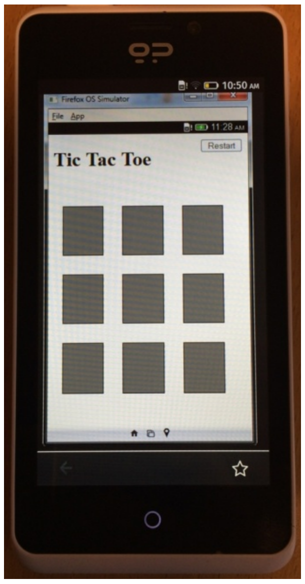

Fig. 8. Firefox Client (Firefox Geeksphone)

B. Discussion

Our findings (Table 1) suggest that there are two main platforms at the moment, with iOS and Android. Both have mature development environments and communities. Especially for the Android platform there is a large number of open source libraries and frameworks.

For the other two alternatives, namely Firefox OS and Windows Phone, there is more uncertainty about the future [33]. While Firefox OS is a new Operating System, Microsoft has a long history in the mobile space. Both Firefox OS and Windows Phone are small compared to the two main platforms, but they have potential. Firefox OS focuses on a different market, targeting more affordable devices. Windows phone, on the other hand, is more similar in that it focuses on the mid to high-end smartphones.

Windows Phone takes advantage of the excellent development support in Visual Studio. Combined with the C# language, together they provide a good alternative for developers. One of the main issues Microsoft has had with its focus on mobile devices is low market share. After the iPhone entered the market, they have conducted major changes. This has entailed efforts such as going from Windows Mobile to a completely redesigned OS with Windows Phone, or, indeed, acquiring handset manufacturers, as has happened with the recent acquisition of Nokia.

Android is clearly the most popular platform of the alternatives we have investigated. When developing for Android devices, the potential customer base is very large. However, the platform also has its challenges, mainly with fragmentation and the lack of updates. Fragmentation is simply the problem that there are so many different devices supporting Android, and it is difficult to create an App that works across all various sorts. The lack of updates is the case that certain devices, even quite new, will not receive updates of the OS. Additionally, there has previously been a problem that many users simply do not update their device. This means there are a considerable amount of devices with very old versions of Android, which needs to be supported. The development tools for Android have continuously improved. One area we feel that both Windows Phone and iOS provides a better experience is with the UI builders. While this is also improving, the Android platform has some major challenges due to fragmentation and backwards compatibility.

Apple, with the iOS platform, has had a very clear focus on high-end devices. The development language used is Objective-C, which can cause a challenge for users that are more familiar with Java/C# development. This can create a steeper learning curve compared to the other platforms.

In contrast to Android, the iOS platform does not suffer from the same issues with fragmentations. This is simply because the number of devices are limited, with Apple being the only manufacturer of device for iOS. This is very different from Android, where there are many different hardware manufacturers.

It is difficult to do an in-depth review of the newest platform, Firefox OS. It is still early days when it comes to the community and developer tools. This will probably improve as the platform becomes more mature. It is also particularly interesting to see the extensive use of HTML5 and Javascript. We feel that the development tools are currently not good enough compare the other platforms. This will improve with future releases. If the platform will handle the common challenges, such as fragmentation, remains to be seen. However, it is certainly an interesting platform that is worth a closer look

TABLE 1: Platform Comparison Matrix

Issue	**iOS**	**Android**	**Firefox OS**	**Windows Phone**
Software architecture				
Development language	*Objective-C*	*Java*	*Web (HTML5, CSS3, Javascript)*	*.Net C#*
Packaging	Apple application package (IPA) with a distribution provisioning file	Android package (APK) file	Packaged as a web application with an associated manifest file for properties	Windows Phone package (XAP) with manifest file for properties
Persistent storage and database support	Local SQL database support and local file access	Local SQL database support and local file access	Local IndexedDB database support and local key/value pair storage	Local SQL database support and local file access
Application development				
Debugger availability	Very Good	Excellent	Good	Excellent
Deployment speed (packaging, installing, testing)	Fast	Relatively fast	Fast	Relatively fast
Default deployment application size	Medium	Large	Very small	Large
Developer support				
Developer community and support	Very large	Very large	Very small	Average
Market penetration	Very large	Very large	Minimal	Limited
Integrated development environment (IDE) availability	Very good support through Apple Xcode and Jetbrains Appcode	Excellent – supported by all major IDE's	Very limited specialized tools, but regular web development tools can be applied	Very good, but limited to Microsoft Visual Studio
Development tools cost	Free for emulator Small fee for device and App Store	Free Small fee for Android Play	Free	Free (for emulator) Small fee for Marketplace

V. CONCLUSION

There is little doubt that mobile devices, and particularly smartphones and tablets, will be the device of choice for users in the very near future. In this context, mobile apps become an essential part of any software development project. It is therefore important to acknowledge and investigate the strength and weaknesses of the various devices and their associated mobile ecosystems – which we have explored in this paper.

Developer support has been greatly improved in the current development tools: performance is abstracted to new high-level formats, and access to performance-critical code is often wrapped through third party libraries. Indeed, language development itself also shows proofs of abstraction, such as in the Android platform, where a new programming dialect has been built on top of the Java language. Moreover, in the case of the iPhone, the higher level of abstraction is especially apparent in UI design and database integration. This is solved with specific tooling, making the abstraction level higher. Indeed, in these areas the iPhone/Xcode model stands out and provides the most efficient and best development environment in our opinion.

Windows Phone, Android, and iPhone have the benefit of being tightly integrated with the operating system on the mobile phone. This results in a good integration between the development environment and the actual devices. However, Firefox OS struggles with the different implementations, immature platform support, and a wide variety of mobile phones and browsers.

Our work opens up possibilities for interesting future research. Other areas of developer interest, such as multithreading could be included. Table 1, could be further expanded, to include code footprints and runtime issues, such as CPU load, battery usage and network performance, to name but a few. Apps generated by cross-platform development tools are also of potential interest. All are worthy future pursuits

REFERENCES

[1] A. Bottaro and A. Gérodolle, "Home SOA -: facing protocol heterogeneity in pervasive applications," *Proceedings of the 5th international conference on Pervasive services*, Sorrento, Italy: ACM, 2008, pp. 73-80.

[2] A. Charland and B. Leroux, "Mobile Application Development: Web vs. Native", *Communications of the ACM*, vol. 54, no. 5, pp. 49-53, 2011.

[3] J.J. Colao, "Facebook's HTML 5 Dilemma Explained", *Forbes,* http://www.forbes.com/sites/jjcolao/2012/09/19/facebooks-html5-dilemma-explained/ [Accessed: 13 November 2013]

[4] ContextPhone: A Prototyping Platform for Context-Aware Mobile Applications (2005)

[5] J. Conway and A. Hillegass, *iPhone Programming: The Big Nerd Ranch Guide (Big Nerd Ranch Guides)*, Addison-Wesley, 2010.

[6] W. Du and L. Wang, "Context-aware application programming for mobile devices," *Proceedings 2008 Canadian Conference on Computer Science & Software Engineering (C^3S^2E)*, Montreal, Quebec, Canada: ACM, 2008, pp. 215- 227.

[7] D. Gavalas and D. Economou, "Development Platforms for Mobile Applications: Status and Trends", *IEEE Software*, vol. 28, no.1., pp. 77-86, 2011.

[8] Google, "What is Android?" http://developer.android.com/guide/basics/what-is-android.html [Accessed 10 July 2010]

[9] T-M. Grønli, J. Hansen, and G. Ghinea. "Android vs Windows Mobile vs Java ME: a comparative study of mobile development environments." *Proceedings of the 3rd International Conference on PErvasive Technologies Related to Assistive Environments*. ACM, 2010.

[10] S.P. Hall and E. Anderson, "Operating systems for mobile computing," *Journal of Computing Sciences in Colleges.*, vol. 25, no.2, 2009, pp. 64-71.

[11] E. Harjula, M. Ylianttila, J. Ala-Kurikka, J. Riekki, and J. Sauvola, "Plug-and-play application platform: towards mobile peer-to-peer," *Proceedings 3rd international conference on Mobile and ubiquitous multimedia*, College Park, Maryland: ACM, 2004, pp. 63-69.

[12] M.T. Heikkinen and J. Still, "Benefits and challenges of new mobile service development in R&D network," *Personal Ubiquitous Computing*, vol. 12, 2008, pp. 85-94.

[13] A. Holzer and J. Ondrus, "Mobile application market: A developer's perpsective", *Telematics and Informatics*, vol. 28, pp. 22-31, 2011.

[14] M. Huebscher, N. Pryce, N. Dulay, and P. Thompson, "Issues in Developing Ubicomp Applications on Symbian Phones," *Proceedings of the international workshop on System Support for Future Mobile Computing Applications*, IEEE, 2006, pp. 51-56.

[15] W. Jobe, "Native Apps vs. Mobile Web Apps", *International Journal of Interactive Mobile Technologies (iJIM)*, vol. 7, no. 4, pp. 27-32, 2013.

[16] A. Koller, G. Foster, and M. Wright, "Java Micro Edition and Adobe Flash Lite for arcade-style mobile phone game development: a comparative study," *Proceedings SAICSIT 2008* Wilderness, South Africa: ACM, 2008, pp. 131-138.

[17] N. Milic-Frayling, M. Hicks, R. Jones, and J. Costello, "On the design and evaluation of web augmented mobile applications," *Proc. 9th Int. Conf. on Human computer interaction with mobile devices and services*, Singapore: ACM, 2007, pp. 226-233.

[18] H. Mukhtar, D. Belaïd, and G. Bernard, "A model for resource specification in mobile services," *Proceedings of the 3rd international workshop on Services integration in pervasive environments*, Sorrento, Italy: ACM, 2008, pp. 37-42.

[19] A. Nicolau, "Best Practices on the Move: Building Web Apps for Mobile Devices", *Communications of the ACM*, vol. 56 no. 8, pp. 45-51, 2013.

[20] J. Ohrt and V. Torau, "Cross-Platform Development Tools for Smartphone Applications", *IEEE Computer*, vol. 45, no.9., pp. 72-79, 2012.

[21] Open Handset Alliance, "Open Handset Alliance", http://www.openhandsetalliance.com/ [Accessed: 7 October 2012]

[22] C. Petzold, "*Programming Windows Phone 7*", http://www.charlespetzold.com/phone/ [Accessed: 13 November 2013]

[23] I. Singh and M. Palmieri, *"Comparison of Cross-Platform Mobile Development Tools"*, *Proceedings 16th International Conference on Intelligence in Next Generation Networks*, Berlin, 2012.

[24] Sun Microsystems (2005) "CLDC HotSpotTM Implementation Virtual Machine"

[25] T. Yamabe, K. Takahashi, and T. Nakajima, "Design issues and an empirical study in mobility oriented service development," *Proceedings 1st workshop on Mobile middleware: embracing the personal communication device*, Leuven, Belgium: ACM, 2008, pp. 1-6.

[26] http://developer.android.com/index.html [Accessed: 13 November 2013]

[27] https://developer.apple.com/ [Accessed: 13 November 2013]

[28] https://developer.apple.com/library/ios/releasenotes/ObjectiveC/RN-TransitioningToARC/Introduction/Introduction.html [Accessed: 13 November 2013]

[29] https://developer.apple.com/xcode/.[Accessed: 13 November 2013]

[30] https://marketplace.firefox.com/developers/ [Accessed: 13 November 2013]

[31] http://msdn.microsoft.com/en-us/library/cc838164(v=vs.95).aspx [Accessed: 13 November 2013]

[32] http://msdn.microsoft.com/en-us/library/windows/desktop/bb318664(v=vs.85).aspx) [Accessed: 13 November 2013]

[33] http://technet.microsoft.com/en-us/ [Accessed: 13 November 2013]

[34] http://www.apple.com/pr/library/2007/01/09Apple-Reinvents-the-Phone-with-iPhone.html [Accessed: 13 November 2013]

[35] http://www.gartner.com/newsroom/id/2573415 [Accessed: 13 November 2013]

[36] http://www.microsoft.com/en-us/news/press/2013/sep13/09-02announcementpr.aspx [Accessed: 13 November 2013]

[37] http://www.microsoft.com/visualstudio/eng/visual-studio-2013 [Accessed: 13 November 2013]

[38] http://www.mozilla.org/en-US/firefox/os/ [Accessed: 13 November 2013]

[39] http://www.ztedevices.com/product/smart_phone/index_1.html [Accessed: 13 November 2013]