International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 4 Issue 09, September-2015

Cross Platform in Mobile Applications

Miss. Priyanka Sinha
Faculty coordinator in Poornima University
Department of Computer Engineering
Poornima University
Jaipur, Rajasthan

Miss. Mamta Verma
B.Tech Student in Poornima University
Department of Computer Engineering
Poornima University
Jaipur, Rajasthan

***Abstract*--There are three types of mobile application development, which are native mobile application, cross platform mobile application and hybrid application. Many mobile devices that runs with different operating system and languages in current mobile market. Android and iOS is the most leading representative as the whole of current market. Developing application in different platforms has generated many problems; a lot of time, cost and man power is needed if there is no significant porting work for cross platform several mobile devices are propose For example- XMLVM, Phonegap, PhoneXml, DragonRad and RhoMobile, each with their own strength. Cross platform is necessary in current mobile market and for developing games in cross platform use XMLVM. With the rapidly evolving mobile game industry it has become increasingly lucrative to develop quickly across as many different platforms as possible. The solution for developing games in cross platform, based on a range of open source software packages to easily create cross platform games.web browsers are common platform for delivering cross platform mobile applications also create widgets for cross platform mobile applications they are easy to develop, easy to deploy, personalized, interactive, and less consumption of flow of mobile widget make it ideal for mobile internet. A graphical UI would enhance the development process in development process of creative games allowing for easy addition.**

Keywords*-- *Cross platform, mobile, framework, Android, ios, mobile widget, cross platform-MWPDL, games development, open source, bug patterns, bug detection, cross platform techniques.

INTRODUCTION

Technology innovation has leads peoples to be well dependence on mobile phones. Mobiles had become a necessity rather than need in our daily life. Smart phones today have large touch screens, nice user interfaces, and are highly optimized for browsing the web. There are many features in mobiles in addition to making calls such as Camera, Music, Social Networking and Global Precipitation System (GPS). However, there are numbers of smart phones with heterogeneous platform in current mobile market, which are android, iOS, Symbia,Window Phone, BlackBerry and others. Competition among the diverse operating system is the main factor that triggers the developers to add in new features to the operating system and feasibility to develop. The basic architecture of each OS is very different from each others, forcing re-development to be a must. However, it is difficult to develop mobile applications for a wide range of mobile phones without a significant porting effort since mobile runs in a diverse range of OS. A lot of time money, technologies and man power are needed to diverse application development. Statistics has shown that android and iOS has held their leading position in current mobile market as shown in Figure 1 below. Peoples tend to be prompted by the current trends and the total applications available in each platform. In order to avoid re-implementing the same application diversely, frameworks and tools such as XMLVM, Phonegap, PhoneXml, DragonRad and RhoMobile are needed. A study on cross platform for mobile device across iOS and Android is proposed for games environment as we found that game is the most suitable application for cross-compilation as they normally will be full screen view and use special purpose they also support almost alike of animation and graphics capabilities

Mobile Platforms	Q1/2012 Device Sales	Current Marketshare	App Downloads	Total Apps Available
Android/Google	**81,067,400**	**56%**	15,000,000,000	500,000
iOS/Apple	33,120,500	23%	**30,000,000,000**	**650,000**
Blackberry	9,939,300	7%	3,000,000,000	70,000
Windows Phone	2,712,500	2%	n/a	82,234
Other	n/a	12%	n/a	n/a

Source: Mobile Statistics (June 2012), Lovingly Organized by AlexanderBosika.com

Fig. 1 Mobile platforms Statistic

Since different mobile platforms (e.g., Android, iOS, Windows Phone, Blackberry OS, etc.) provide different programming languages and tools for the developers to code platform-specific mobile apps, the heterogeneity greatly increases the cost and latency of developing a single mobile service for all mobile platforms. To release a specific mobile app against multiple platforms, developers usually rewrite the app against each target platform. Such a manual process is labor-intensive, error-prone, and usually impracticable under a limited time or cost budget. As a category of mobile services, cross-platform mobile apps must intensively exploit device-native components (e.g. GPS, camera, accelerometer, etc.) to provide rich features

to their users. However, these features are usually implemented via respective sets of relevant APIs provided by the cross- platform tool in use. When developing cross-platform apps in Work light, developers can use the API"navigator.accelerometer.watchAcceleratio" to get the current acceleration of a mobile device at a regular interval, and then use the API "navigator.accelerometer.clearWatch" to stop watching the accelerometer sensor. Furthermore, each set of relevant APIs usually needs to be used under some specific temporal order (e.g., in order to avoid wasting energy, method *clearWatch* needs to be appropriately called in time to release the accelerometer sensor after the method *watch Acceleration* is called). Moreover, cross-platform tools usually provide many value-added capabilities to facilitate developing cross-platform apps (e.g., Work light provides encrypted offline cache, which makes it easy for the apps to manage offline security data in mobile systems for the users). In order to safely use these capabilities, the developers must also deal with the relevant sets of APIs under various temporal constraints (e.g., an encrypted offline cache in Work light application should be used after opened and be closed after used in time). Should these constraints be violated, temporal bugs are introduced. Bugs for the cross-platform mobile apps *at development time*. Compared with traditional mobile apps developed via static languages (e.g., Java, Objective-C), cross-platform mobile apps are mostly coded via JavaScript-like dynamic languages (which
support execution-time change of code modules and data structures). Such dynamic languages enable a more flexible means to operating the embedded components or resources in mobile systems. For example, as shown in Fig. 1, developers usually operate a native component via some global function invocations, and the positions the component handle may appear in these invocations are uncertain. Besides that, due to the event-driven programming nature of mobile apps, a set of relevant APIs may be separately invoked in different event handlers. Such dynamic characteristics pose additional challenges for the automatic detection of temporal bugs in cross-platform mobile apps at development time. In order to safely use these capabilities, the developers must also deal with the relevant sets of APIs under various temporal constraints (e.g. an encrypted offline cache in worklight application should be used after opened and be closed after used in time).Should these constraints be violated, temporal bugs are introduce.

I. RELATED WORK

This chapter summarizes the major literature findings relevant to the topic as cross-platform mobile for Android and iOS platform in games application development.

A. Games Development

Smart phone has been target by social network and Multi-player online games (MOGs) due the special capabilities it has, which are GPS and accelerometer. However, MOGs now days can only support single game platform, which is on PCs, game consoles, arcade game machines or mobile devices only . It could be troublesome if the players want to interact and play together using different platforms. Besides that, developing game into multi-platforms is required for "Serious Games" for education and training purpose instead of serve as an entertainment only together with problem solving. The 'Serious Game' also focuses on expose the public to the management and leadership challenges faced. However, game applications that could work on heterogeneous platforms are required. A porting work is needed to run application in different programming model of the smart phones. This will acquire high overhead and it will be a difficult and troublesome since mostly the developers more expert in developing application based on the particular platform. For example, Android application developers will more familiar on Java programming language [2], whereas for the iOS, Objective-C is used. There are several solutions proposed in order to solve the problem.

B. *Android*

Android is an open source and Android SDK is applicable in heterogeneous platforms with well documented API . Android is very different from the platform used in iPhone. Android is practically designed for a wide range of mobile systems and also the programming environment used, which is Java Programming language. Android application consists of a set of activities which included the user interaction that may have one or more input screens. For example, select a contact from the internal address book is an activity. User may flip through the contact list or may use a search box to find the specific contact number. Many actions are combined to become an activity. Activities have a well- defined life cycle and can be invoked from other activities even activities from other applications. Besides a variety of widgets, the declarative description of user interfaces is also allowed in Android. XML files describe the relative layout of a user interface which simplifies internationalization and also allows rendering the user interface on different screen resolutions.

C. *iOS*

iOS is the platform used in Apple product such as iPhone, iPad and iPod. The primary language offered by Apple for iOS development is Objective-C. Apple prohibits general background processes per license agreement of its SDK which implies that the life cycle of an application is much simpler compared to Android. We found that there are many possible or applied solutions for the cross-platform in iOS through our studies. One of the study we have made is about Robei. Robei is a cross platform FPGA design tool that purposes to simplify user interface design, transparent intellectual properties and reduce complexity. It is based on the GUI framework, QT that already ported to iOS and Android platform. Robei can be recompiled on many platforms without much modification. The code view tab allows engineers to add or modify algorithm code easily to realize certain behaviours. In Robei, there are only three basic elements to represent Verilog components in hardware design: module, port and wire. Module is used to

communicate with other chips. Port is to act as interface channel for module and model. Wire on the other hand is for signal transmission. Robei can generate code based on design diagram, integrating with algorithm code from engineer which reduces code input and avoids mistakes as shown in figure 2 below. Robei is the first FPGA design tool that runs on embedded platforms, which allowed the design runs anywhere on mobile phones or tablets [6].

D. *Framework*

There are various types of framework to cross compile a mobile system in Android and iOS. *XMLVM* [1] is introduced to be a solution for cross-compilation, which is a byte code level cross-compiler that based on the Java byte code instructions [1]. The developer only required to maintain one code-base only which is easier to parse compared to Java source code [1]. XMLVM tool chain [2] enables different mobile platforms to cross-compiling the application into other mobile devices without the need of the knowledge for the native language used in others. XMLVM not only can cross- compiles on a language level but also maps different API [1]. Generally, the source code will be compiled by regular Java compiler into Java byte code before input into the XMLVM.

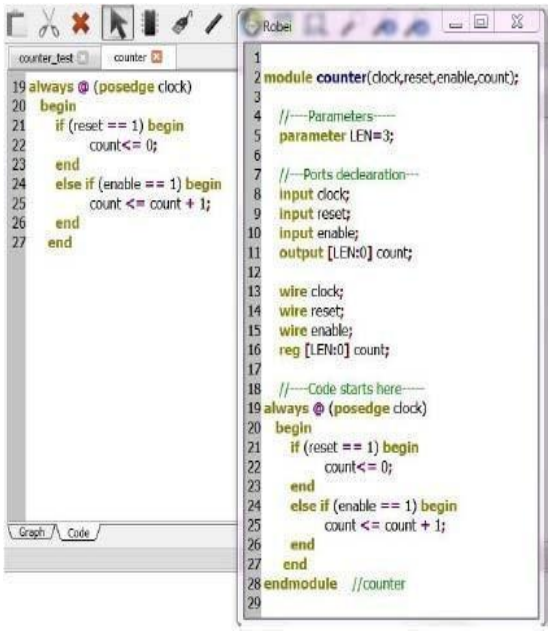

Fig. 3 Core algorithm that requires user to type in and generated code [6]

PhoneGap [3] is an open-source mobile development framework that supports most heterogeneous platforms including iOS and Android. PhoneGap is a useful solution using modern web technologies like HTML, CSS and JavaScript which make use of the functionality of SDKs. PhoneGap is the hybrid applications due to the layout rendering done through web-view but some of the functions are missing support of HTML. On the other hand, we have PhoneXML [3], a framework that could be parsed at every mobile platform to generate User Interface of phone application. It works in the form of an application that will render the PhoneXML returned by the application server. Same application could be used for different business logic and user interface provided by server. This XML could be static and dynamic at application runtime. DragonRad [3] supports iPhone, Android, BlackBerry and Windows mobile OS. It focuses on database driven mobile enterprise applications which provides drag and drop environment to save time. DragonRad Designer, DragonRad Host and DragonRad Client are the three major components for DragonRad architecture. RhoMobile [3] is another solution for the cross-complation, which is a Ruby-based mobile development aims in managing enterprise application and data that can be used across Linux, Mac and Windows OS. It provides a high level of productivity and portability of web programming. Rhodes uses Model View Controller (MVC) pattern to do applications written with HTML, CSS and JavaScript using Ruby programming language. App generator generates the business logic of views and controller with native applications. The applications can be compiled and to be executed natively on all mobile platforms.

E. Comparison

Smartphone platforms differ greatly in their native application development models. Android provides a mobile operating system can run on the Linux kernel while iOS is proprietary platform developed by Apple which is only available for Apple devices. Android employs Java whereas iOS uses Objective-C. Java feature strong typing and garbage collection while the version of Objective-C used on iOS supports dynamic typing without garbage collection. Android and iOS differ greatly in their APIs for user interface, application lifecycle and device management.

	Nexus S	iPhone 4
OS	Linux	iOS
CPU	Hummingbird S5PC110, 1 GHz	Apple A4, 1 GHz
RAM	512 MB	512 MB
Sensors	Accelerometer, GPS, proximity, ambient light, compass.	Accelerometer, GPS, proximity, ambient light, compass.
IDE	Eclipse	Xcode
Language	Java	Objective-C
GUI	Android	Cocoa Touch
VMs	Allowed	Not allowed
License	Open Source	Proprietary

Fig. 4 Smartphone comparison [2]

By comparing all the frameworks, we found that each framework has their own strength among others. Phonegap can support most of the different mobile platforms in the market such as iOS, BlackBerry, Android, Symbian, WebOS and Windows Mobile. DragonRad is the only framework that offers the possibility to produce both web applications as well as native applications. Rhodes is the only framework with MVC support which is able to write real business logic on local native application while XLMVM allowed API mapping to be done by knowing just one native language. Propose a pattern- based approach to detecting temporal bugs in cross-platform mobile apps at development time. The approach is app-

agnostic and scalable because the bug patterns are not coded in a programming language in an app-specific manner and they can be maintained and checked in parallel with no inter-pattern dependencies. A *Flexible Bug Pattern Specification Notation* (*FBPSN*) is created in support of this approach such that typical correct/buggy usage scenarios of the embedded components or services in cross-platform mobile apps can be easily specified as standalone bug patterns. A static defect detection approach is used to support systematic detection of temporal bugs per a set of bug patterns specified in FBPSN. We have realized and evaluated the proposed approach by developing the bug detection tool *PatBugs.*

II. METHODOLOGY

Smart phones are target by social network and MOGs. We realized that today multi-player online games can only support single game platform, which is on PCs, game consoles, arcade game machines or mobile devices only [4]. As a result, this will be troublesome for players who using different platforms to play the same game together as different platforms of users failed to interact in an MOG. There are generally two types of games: Interpreted Language Game, that used interpreted language that will be interpreted by interpreter program without translation, for example C#; and Compiled Language Game, that using compiled language that will be translate and generate machine code through compilers like C++ and C. As comparison, Compiled Language Game enables rich, high-performance games and is faster than Interpreted Language Game, and is more suitable to develop multi-platform mobile games [4]. Based on the idea of 'Serious Game', game designs, technologies, and development skills will be utilize in new series of policy education, exploration, management and so on [4]. So, it is important to have a full evaluation on the cross- platform mobile phone game development environments. There are basically three ways to implement MOGs on heterogeneous platform. One of the solutions are the 'world-wide first' multi-platform online game by implement it into the first-person shooting game [4]. The game service environment for the game is divided into six types of platforms based on devices capabilities either high-end or low-end machine. Its core components are 3D game engines and multi-platform game server [4]. Designing multi-platform online game required to take in account for the hardware performance of two classified types of platforms, network latency for wired and wireless networks and the quality and performance of the game in different platform [11]. Second is using mobile development environment which are Serve Studio and X-Forge. Swerve Studio is used to create wireless application with high quality interactive 3D graphics via generic toolset [4]. It also included Java and C/C++ SDKs so the scene can be fully control by the programmers.[4] Whereas, X-Forge is developed via C++ based multi-platform game engine and provides tools, documentation and examples for every phase of advanced mobile game development [11]. Both development environments has different target segments, Swerve Studio target on middle and low level mobile phone while X-forge target on advanced mobile phone such as smart phone [4].

However, we have suggested using the third solution which is using XMLVM framework to develop a cross platform for Android and iOS games environment. Android is very different from the platform used in iPhone but Android having a wider range of mobile systems and also the programming environment used, which is Java programming language [1][2]. Practically, Android is used as the canonical platform [1][2] in the XMLVM. XMLVM not only a byte code level cross- compile for mobile application but it also can map APIs from different platforms [1]. It is a great advantage since the platform used by each smart phone is greatly different so the developer can use XMLVM tool chain [2] to deal with cross- compiling the application into other smart phones without the need of the knowledge for the native language used in other smart phones. By using XMLVM framework, developer only required to well-known to Android system. API mapping allowed the Android applications to be translated and used in Cocoa Touch for iPhone 3GS. So, XMLVM has a higher potential to develop the games environment for heterogeneous platform

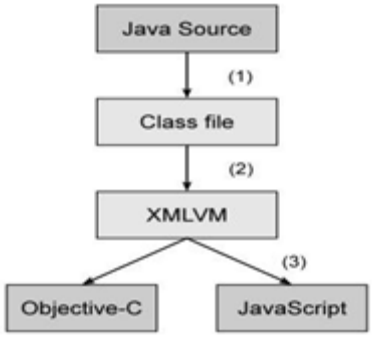

Fig.4. XMLVM framework Tool chain. (1) Java source code was compiled into class file (byte code) Via Java compiler.(2) The class file generated was input into the tool chain. The XML representation of the contents of the class file is generated. (3) It is then generate the code for the different language, such as Objective-C and JavaScript

API mapping works in two ways. First is using Cocoa Touch (iPhone) API that required deep analysis of Android application performance so to change and adapt to Cocoa Touch API which is efficient but difficult to run. Secondly, using compatible library like ACL and NAL, where a compatibility layer using Cocoa Touch API providing Android API without changing the original application. Xokoban, a puzzle game that rebuild from Sokoban is cross-compiled to test on the feasibility of the tool used in the XMLVM [2]. Xokoban is rebuilt for the Android platform and use some of the Android APIs and widgets, that are 2D animation, alert views, buttons, and checkboxes, accelerometer and swipe interface and saving or loading preferences [2].

Fig. 5 Xokoban in the Android (*Left*) iOS (*Right)*

III. CONCLUSION

This paper discussed about the cross-platform mobile game development focus on Android and iOS platform. Throughout the comparison and contrast, we have proposed XMLVM framework as the most suitable solution for cross-platform mobile game development. From this study, we have learnt that there are several frameworks that are feasible for mobile cross- platform. We found that it is crucial to compare and make use of the strength of each framework, so that a better, more compatible and more stable cross-platform can be developed in order to avoid re-implementation of the same applications all over again for each different platform separately. Studies also show that there are many potential existing works that are not being explored fully and will probably have a bright future if proper implementation is carried out. Therefore, our future work will still continue the research of studying the latest cross-platform framework. Apart from mobile game development, other mobile applications like the mobile learning system, IM system and phone AR also have the need of cross-platform technique to save time, money, manpower and effort. Hence, this study is important for a cross-platform mobile development. Temporal bug patterns in cross-platform mobile apps are usually much more complex than those of traditional object-oriented applications. Based on this observation, we proposed an effective pattern-based approach to detecting temporal bugs in cross-platform mobile apps at development time. In our approach, the temporal constraints among a set of relevant APIs can be easily specified as standalone bug patterns in an app-agnostic manner via our proposed flexible bug pattern specification notation (FBPSN). Per predefined bug patterns, our bug detection process can automatically detect related temporal bugs in cross-platform mobile apps. The experimental evaluation of Pat Bugs using real-world cross-platform mobile apps showed the proposed approach is effective and could be realized efficiently. In our approach, the bug-pattern specification is cleanly separated from the implementation of bug detection logic, which improves the scalability and the generality of our approach. Once a new bug pattern is defined with FBPSN, our approach can automatically own the power to detect related temporal bugs in a cross-platform.

IV. REFERENCES

[1] Google Android, http://www.android.com/, Accessed on Sep. 14, 2011.
[2] Screen Sizes and Densities, http://developer.android.com/resources/dashd/ screens.html, Accessed on Sep. 19, 2011.
[3] Android tops 81 percent of Smartphone market share in q3. http://www.engadget.com/2013/10/31/ strategy-analytics-q3-2013-phone-share/2013. Accessed: 10/31/2013.
[4] A. Marcus, A. Sergeyev, V. Rajlich, and J. I. Maletic, "An Informa- tion Retrieval Approach to Concept Location in Source Code," in *In Proceedings of the 11th Working Conference on Reverse Engineering (WCRE 2004)*. IEEE Computer Society, 2004, pp. 214–223.
[5] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, "Labeled LDA: A Supervised Topic Model for Credit Attribution in Multi- labeled Corpora," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*. Singapore: Association for Computational Linguistics, August 2009, pp. 248–256
[6] "Stanford Topic Modeling Toolbox," [Accessed 11-July-2012].[Online].Available: http://nlp.stanford.edu/software/tmt/tmt-0.4
[7] D. Han, C. Zhang, X. Fan, A. Hindle, K. Wong, and E. Stroulia, "Annotated Topic Data Used in this Study," http://softwareprocess.es/ static/Fragmentation.html, 2012.
[8] William Enck, Peter Gilbert, Byung-Gon Chun, Lan- don P. Cox, Jaeyeon Jung, Patrick McDaniel, and An- mol N. Sheth. Taintdroid: an information-flow tracking system for realtime privacy monitoring on smartphones. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, OSDI'10, pages 1–6, Berkeley, CA, USA, 2010. USENIX Association.