International Journal of Computer Networks and Communications Security

VOL. 3, NO. 3, MARCH 2015, 103–109
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

Towards a Multi-Platform Development Based on MDA Approach

CHARKAOUI Salma[1], ABDELBAKI Issam[2], BEN LAHMAR El habib[3] and MARZAK Abdelaziz[4]

[1, 2, 3, 4] Faculty of Sciences Ben M'SIK, Department of mathematics and informatics, Casablanca, Morocco

E-mail: [1]charkaoui.salma@gmail.com, [2]i.abdelbaki@gmail.com, [3]h.benlahmer@gmail.com, [4]marzak@hotmail.com

ABSTRACT

The diversity of platforms on the smart phones market, including the large number of operating systems that use different technologies, engenders a fragmentation, ie, the IOS environment / Objective-C for iPhone and iPad, Java SDK for Android, J2ME for Symbian, etc. This fragmentation, make the mobile applications development quite difficult and very expensive. Recognizing the importance of defragmentation and wanting to optimize the design process of mobile applications, cross-platform development could be a good solution to this problematic. Several approaches exist to address the cross-platform development, each responding to a variety of issues. The choice of a cross-platform development approach depends mainly on two things: to know what type of mobile application should be directed, and or targeted platforms. In addition to this, follow a models-based approach is one of our goals, this article describes the solution adopted for the realization of a multi-platform development framework while relying on models, and therefore go beyond current approaches.

Keywords: *Multiplatform Development approach, Mobile applications Type, Native, MDA, Hybrid.*

1 INTRODUCTION

Currently, in the world of new technologies of information and communication, smart phones and tablets become ubiquitous. Moreover, we can't talk without speaking about Smart phones mobile applications. Making a mobile application becomes a strategic issue for companies; as a result, the market for mobile applications has not finished growing.

For a moment the mobile market has been dominated by Apple but the replica of the open source community has been swift. Thus under the leadership of Google, the Android free operating system was introduced. According to current polls, Android is the first mobile operating system followed by Apple's iOS and others such as Symbian, RIM etc. Android has seen a huge variation in almost everything: screen size, resolution, processor speed, memory, functions and OS version.

The diversity that exists in the mobile area, especially the large number of operating systems that use different technologies, however, creates a "fragmentation: environment IOS/ Objective-C for the iPhone and iPad, Android SDK specific Java, J2ME for Symbian etc…".

Recognizing the importance of defragmentation and wanting to optimize the design process of mobile applications, the idea of developing a single application that works everywhere (or almost everywhere) became a goal that was much more difficult to achieve - but remains as attractive as ever.

In this research work we will start by presenting the cross-platform development, then move on to the different types of mobile applications, and then define and explain the different approaches used in the cross-platform development, in order to achieve a comparison between these approaches that allows us to identify the strong and weak points of each one. We will present our results in the form of a comparative table taking into consideration several criteria.

These benchmarks have shown that the choice of a cross-platform development approach depends mainly on two things: to know what type of mobile application should be directed, and or targeted

platforms. In the last section of this article we describe the solution adopted for the realization of multi-platform development framework while relying on models, since models driven software engineering is the future of applications.

2 MULTIPLATFORM DEVELOPMENT

Carry a mobile application is not an easy task, the current multiplicity of devices and ergonomic differences between them transform how design and develop applications. Each major smart phone platform or personal computer has its own programming language, its own set of APIs, its own development environment and its own app store.

This diversity has given rise to the cross-platform development that allows using the same code to deploy an application on multiple platforms (iOS, Android, BlackBerry, Windows Phone), and thus avoid having to develop the same application in multiple copies, each time in different languages.

Indeed, the ability to deploy software to multiple environments don't only save time (time to replicate the code for each target platform) but save also resources (cost of production). In addition, a significant gain comes from the notion of cross-platform development: The applications maintenance.

3 MOBILE APPLICATIONS TYPES

All studies converge: mobile devices, smart phones and tablets are becoming the main access points to the Internet. Offer a mobile applications becomes a strategic issue for companies, result, the market for mobile applications has not finished growing.

The current constraint faced by these companies in their entry into the mobile world is the choice of the strategy to cope with the variety of existing platforms before embarking on the creation of a mobile application, it's recommended to define the most appropriate type of solution according to the content type or the service to implement.

In the mobile world, applications can be broken down into three types: native, hybrid and web app.

3.1 Native or embedded applications

A native application is based on the targeted platform language which makes it strongly related to the programming language that supports the platform. Applications thus created are then downloaded from a platform dedicated system, usually Apple Store and Android Market, and App Store application types can be launched as separate software. These applications take advantage of the power and possibilities of the mobile phone. [1][10]

3.2 Web or connected applications (Web Apps)

These application types function as web sites. Indeed, you can access these applications via the web browser of the mobile device. They are however not available via apps blinds manufacturers. These applications are built entirely with web language and work with all mobile browsers.

The disadvantage of this approach is that compared to native apps, they cannot, however, take full advantage of the graphics capabilities of the devices, and they cannot even use all the functions of local terminals (calendar, etc directory.) And generally requires an internet connection. [1][10]

3.3 Hybrid or synchronized applications

The hybrid applications are born there a few years ago with frameworks like PhoneGap or Titanium. They are an ideal way to bring the advantages of web and native approaches (this is a mixture of native code and display HTML view / JavaScript). The principle is to provide a way to deliver applications that run locally with web technologies returned via the web rendering engine, and taking advantage of local hardware capabilities such as GPS, camera, etc… As web apps, they are immediately compatible on all platforms. There is no need to redevelop the application each time. As native apps, they can also be downloaded from the app stores and manufacturers are available on the device as a complete application. Ergonomics and graphic rendering of the application is also closer to the native apps (web application running inside a thin native package that provides a gateway to the operating system). [1]

The chart below gives you a quick overview of the pros and cons of choosing between Native, Hybrid or Web applications. In this table can be observed that Hybrid solution is the most appropriated for cross platform development, developers can achieve the best of both worlds, since Hybrid solutions offers a balance between the flexibility of web apps, and the functionality of native apps, without forgetting the ability to work across multiple device types and platforms, whilst also leveraging the capabilities of the mobile device hardware.

Table 1: Comparison between types of mobile app

	Native	WebApp	Hybrid
Device API			
App speed			
Stores/Market			
Cross Platform			
Instant Update			
Offline			
Dev cost			

4 CROSS-PLATFORM DEVELOPMENT APPROACHES

4.1 "Runtime" approach

"Runtime" or "scripted" approach allows taking advantage of cross-platform functionality through the use of a scripting language. There are many such tools on the market the main difference between them is the choice of the used language [2].

In this approach, the application developer writes the application using the selected scripting language [2], which can be JavaScript, Lua or Ruby. The multiplatform development tool takes the scripts as is, and copy the installation package without modification. At the same time, the tool adds also a copy of the script language interpreter in the same package, and puts everything in a package (the result is often called a "native" application). So to work on multiple platforms it must provide a specific version of the interpreter for each supported platform. [11]

4.2 "Source Code Translators" approach

This approach is also called "Byte code approach", it consists of the source code translation (Cross-compilation) to an intermediate byte code, native language (C + +, Objective-C, JavaScript ...) or directly to the assembler (machine code). It is always used with a performance element (Runtime).

This approach is so similar to the previous approach (the scripted approach), the only difference is the addition of the byte code compilation step. [2][11]

4.3 "Web-to-native wrapper" approach

This approach is also called "The embedded web browser approach", it represents a solution to create a native application with web technologies (HTML5, CSS and JavaScript). The web code is packaged with libraries linking it with the native functionality of the application all within an envelope of native application [3], which allows the application to use the platform capacity beyond those normally incurred by the browser (notifications, accelerometer, compass, geotagging ...). [2][9][11]

4.4 "App Factory" approach

It consists of visual design tools allowing users to develop their applications without code to quickly build simple mobile applications. [3]

This approach is dedicated to non-developers or people who want to develop simple applications without worrying about programming [3].

4.5 "JavaScript frameworks" approach

These are libraries (registers) for code designed to accelerate web development tasks such as managing complex tactile interactions, cross-browser user interfaces or management games sprites [1 creation]. The use of these frameworks allows a developer to do more with less code by chaining, and can accelerate the development process by using the code that already exists. [4][6][10]

Examples of JavaScript Frameworks: jQuery Mobile, SenchaTouch, Cocos2D, DHTMLX Touch, ZeptoJS, Impact.js, iUIet Wink.

5 ANALYSYS AND COMPARISON

This section will emphasize the differences between the approaches of cross platform development previously described. The factors considered to analyze and compare these approaches were: the targeted public, types of resulting app, programming language and the pros and cons of each approach.

Table 2 summarizes the result obtained after comparing the different approaches of cross platform mobile development.

As can be seen in this latter, Hybrid is the most appropriated solution for cross platform development. In table 2 can be observed that only two approaches allow us to create hybrid applications namely "App factories" and "Web-to-native wrapper". We can't choose "App factories" since it consist in visual tools that allow users to develop their app without code, in other words this approach is dedicated to non-developers or people who want to develop simple applications without worrying about programming. So the web to native wrapper approach is the most appropriate for this type of application, but it has many limitations and we do

not capitalize on the functional of the application regardless of the technical concerns to facilitate migration.

Another approach exists which is the MDA approach, whose the objective is to recover the functional application code to facilitate conversion to another platform. In other words, we must capitalize on the functional application regardless of technical concerns (business) in order to facilitate migration, the MDA approach had good return for application development business and bring a lot to the mobile applications.

Table 2: Comparison between approaches

Approaches	Examples	Targeted public	Programming language	Types of resulting App	Pros/Cons
Runtime	**Titanium (Appcelerator), AppMobi, Adobe Air/Flex, Corona, Rhomobile**	**Software Developers**	**Any scripting language (WYSIWYG and Lua …)**	**Native**	**Ease of obtaining a script interpreter.** **Big size of installer program** **Significant use of memory.** **The performance of execution is reduced**
Source Code Translators	**MoSync, Eqela, XMLVM**	**Software Developers**	**Any scripting language (WYSIWYG and Lua …)**	**Native**	**Big size of installer program** **Significant use of memory.** **The performance of execution is reduced**
Web-to-native wrapper	**PhoneGap, Sencha (touch v2)**	**Web Developers**	**HTML, Javascript, CSS**	**Hybrid**	**An easy way to convert existing websites into native application.** **Maximum utilization of web standard technologies** **Reduced development costs** **Deploying applications on the stores (AppStore, Android market …)** **More platforms supported.** **Poor performance** **Access to APIs of the platform remains limited**
App factories	**Games Salad, Wix Mobile, Spot Specific**	**Non developers**	**Visual Tools code-free**	**Native, Hybrid, Web**	**Ease of use without having to worry about programming**
Javascript frameworks	**jQuery Mobile, Sencha Touch, ImpactJS**	**Web Developers**	**Javascript**	**App web**	**Fast web browsing** **Lightweight frameworks.** **No access to native API platform**

After the procedural technology, object technology and component technology, the MDA (Model Driven Architecture) approach is a process-driven engineering models [5][8] (or MDE for Model Driven Engineering).

The MDA approach distinguishes two main aspects in the development process of an application; the business aspect is the functionality of the application and the technical aspect which represents the technology implementation of the application. Every aspect is expressed by a set of models that convey the information needed to generate the source code of the application. We move from a contemplative view models to a productive view.

The principle key of MDA is the use of models at different phases of the development cycle of an application. Three levels of models representing the levels of abstraction of the application, (CIM models requirement), the PIM (analysis and design) and PSM (code):

Computational Independent Model-CIM Model: This is the job or the application field model, it helps to represent what the system should do exactly. [5]

Platform Independent Model-PIM Model: A computer model that represents a partial view of a CIM. This is the analysis and design of the application model (represents the business logic of the system). [5]

Platform Specific Model-PSM Model: model that most closely matches the final application code. PSM is a model code that describes the implementation of an application on a particular platform; it is bound to an execution platform. [5]

Another model exists but is rarely used, it is the PM that describes the structure and technical functions related to an execution platform (file systems, memory, BDD ...) and explain how to use them. The PM is associated with the PIM for the PSM.

Model transformations recommended by MDA are essentially the CIM to PIM and PIM to PSM transformations. Code generation from the PSM is, in turn, not considered as a model transformation in itself.

EMODE [8] is an example of tools using the MDA approach.

6 MULTI-PLATFORM DEVELOPMENT BASED ON MDA APPROACH

In this section we describe the approach taken to make a cross-platform mobile development framework.

After evaluating different approaches to cross-platform mobile development it turned out that the choice of a development approach depends mainly on two things: to know what type of mobile application should be directed, and or targeted platforms.

Following a multi-platform development approach based on models is among our goals, since it is the future of applications. The MDA approach responds well to our need, its key principle consists of the use of models in the various phases of the application development cycle as explained in the previous section.

6.1 Application of MDA approach

MDA believes that the analysis and design models must be independent of any implementation platform, whether J2EE, .Net, PHP, etc.

In this research work we will first create a simple application for different platforms. We begin for example by such platforms, Android, Windows Phone 7 and Windows Phone 8. Once completed, we will take time to study the software architecture of each application separately, since each has its own characteristics.

Taking for example the configuration files generated by each platform, an Android application declares its parameters and its activities in the AndroisManifest.xml file while Windows Phone 7 does in WMAppManifest.xml.

The figure below shows the file structure of Android and Windows Phone 7 applications.

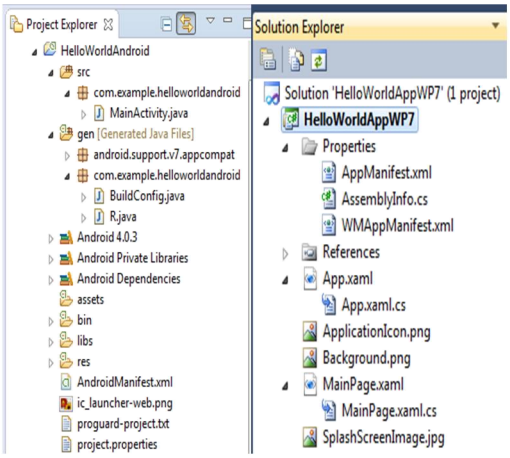

Fig. 1. Files Structure of the Android and WP applications

Develop a model following the study of the software architecture of each platform is the first step to do. From these, make a general meta-model.

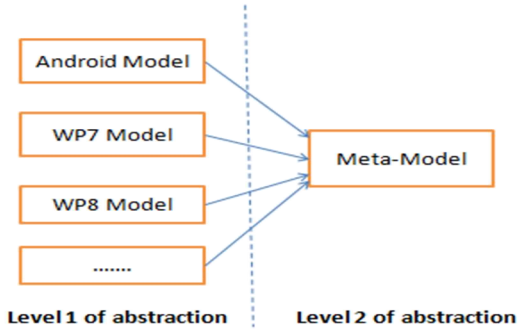

Fig. 2. Meta-Model representation

In our research work we will use UML (Unified Modeling Language), as it is recommended by the MDA approach as the language to use to carry out analysis and design models that are independent of implementation platforms.

This latter is considered stable and semantics is widely shared. In addition, the OMG only advocate the use of UML. It is thus not definitively connected with this language. If tomorrow a new language appears to replace UML, it will be always possible to transform UML models to this new language.

The MDA approach will allow us to automatically generating source code from a UML modeling and this through a models transformation, as shown in the following figure.

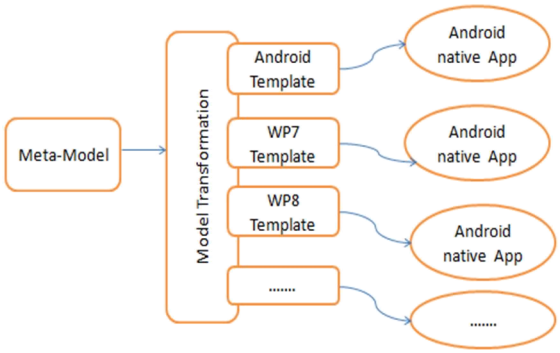

Fig. 3. Cross-platform approach

Based on the main types of models offered by the MDA approach, the main idea is to establish a model PIM (Platform Independent Model) according to business logic independently of the implementation technology, this latter will be transformed into PSM (Platform Specific Model) by model transformation and ultimately generate the executable code for the target platform.

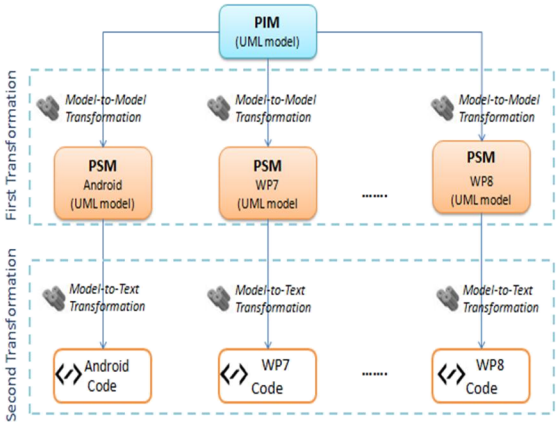

Fig. 4. Model transformation

Knowing that the models are an abstract entity which doesn't require computer representation to exist, MDA using the models for the purpose of productivity, it is however necessary they have concrete representations in order to be handled by computer.

Two different ways are defined by MDA to represent models by computer, in the form of textual documents or programming objects.

6.1.1 Representation in the form of programming objects

A representation in the form of programming objects is more suited to computer manipulation (transformation, execution, validation, etc.).

The JMI (Java Metadata Interface) standard and EMF (Eclipse Modeling Framework) Framework define how to represent models as Java objects.

6.1.2 Representation as textual documents

This representation is more suited to the storage of templates on hard disk or the exchange of models between applications.

The XMI (XML Metadata Interchange) standard defines how to represent the models in the form of XML document, and this is the standard that will be used in our research work. The principle of operation of this latter is to automatically generate the structure of representation formats of models from their meta-model.

6.2 XMI applied to UML

Applying XMI to UML enables the automatic generation of a DTD that allows represent UML models as XML documents. In other words, to represent the model in the form of XML document, XMI generates its DTD from the meta-model of the model. The XML document that defines the model is structured by the DTD generated.

The figure below illustrates the transformation of a model described in UML to an XML document from an XMI definition.

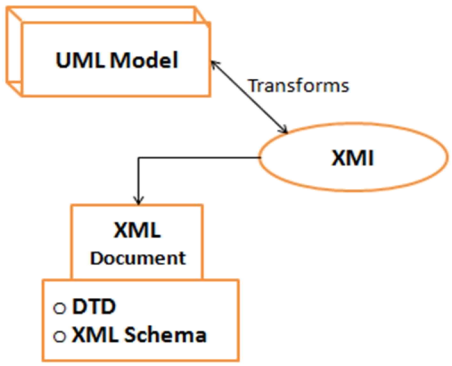

Fig. 5. Transformation of a business model

7 CONCLUSION

Throughout this work, we are interested in mobile platform development. This allowed us to approach the problem of platform fragmentation, which was the heart of the subject. Therefore, we discussed the different mobile applications types and the mobile cross-platform development approaches.

In the context of developing a mobile application for several platforms and / or different form factors, we should be carefully analyzing the needs and objectives of the application that we want to develop. We should know what mobile application type we target at, and which approach to use.

The study through a comparison of mobile cross-platform development approaches, made us able to understand each approach, and so we can say that the choice of which approach to use depends on three factors, our programming habits (JavaScript for PhoneGap Titanium and Ruby Rhodes), the importance of having an application that appears native and or OS that we touch (iOS, Android ...). Aside from these factors, choosing a model-based approach is another factor that comes to be added to previous.

The MDA approach responds well to these factors there, and this is the approach best suited for the realization of a cross-platform development framework.

Studying in detail the software architecture of different platforms to target, in order to emerge a meta-model from UML models realized for each platform, will be subject to a new research work.

8 REFERENCES

[1] N. Serrano, J. Hernantes and G. Gallardo, « Mobile Web Apps, » IEEE Software, vol. 30, no. 5, pp. 22-27, Sept.-Oct. 2013, doi:10.1109/MS.2013.111

[2] Henning Heitkötter, Sébastian Hanschke, Tim A. Majchrzak, « Web Information Systems and Technologies », 8th International Conference, WEBIST 2012, Porto, Portugal, April 18-21, 2012, Revised Selected Papers, Part II : « Evaluating Cross-Platform Development Approaches for MobileApplications », pp 120-138.

[3] Ribeiro, A.; da Silva, A.R., « Survey on Cross-Platforms and Languages for Mobile Apps, » Quality of Information and Communications Technology (QUATIC), 2012 Eighth International Conference on the, vol., no, pp.255, 260, 3-6 Sept. 2012

[4] Sivakumar, Department of Computer Science and Engineering, Indian Institute of Technology,Bombay, « JavaScript Frameworks That Can Make You a Better Web Developer »

[5] Andre, S. (2004). MDA (model driven architecture) principes et états de l'art, Technical report, CNAM, 05 Novembre 2004. 126 BIBLIOGRAPHIE.

[6] Dalmasso, I.; Datta, S.K.; Bonnet, C.; Nikaein, N., «Survey, comparison and evaluation of cross platform mobile application development tools», Wireless Communications and Mobile Computing Conference (IWCMC), 2013 9th International, vol., no, pp.323, 328, 1-5 July 2013

[7] Final report for the Department of the Navy (DoN) Grant # N62909-11-7026 issued by the Office of Naval Research Global (ONRG) and funded by the U.S. Army Materiel Command (USAMC), Telemedicine and Advanced Technology Research Center (TATRC), « Mobile App Infrastructure for Cross-Platform Deployment (N11-38) »

[8] Matthias Heinrich, Matthias Winkler, Hagen Steidelmuller, Manuel Zabelt, SAP AG, SAP Research, Dresden, Germany, « MDA Applied: A Task-Model Driven Tool Chain for Multimodal Applications », 6th International Workshop, TAMODIA 2007, pp 15-27, Toulouse, France, November 7-9, 2007

[9] Hasan, Yousuf; Zaidi, Mustafa; Haider, Najmi; Hasan, W. U.; Amin, I., « Smart Phones Application development using HTML5 and related technologies: A tradeoff between cost and quality», International Journal of Computer Science Issues (IJCSI). May2012, Vol. 9 Issue 3, p455-461. 7p. 5 Charts, 3 Graphs

[10] André Charland, « Mobile application development: web vs. Native », Communications of the ACM, Volume 54 Issue 5, May 2011, Pages 49-53

[11] Sommer, A., & Krusche, S. (2013, February). « Evaluation of cross-platform frameworks for mobile applications », In Proceedings of the 1st European Workshop on Mobile Engineering, pp365.