ISSN 2277-3061

Comparative Analysis of Cross-Platform MAD Frameworks

Kunal Verma[1], Digvijay Maniktala[1]

1-Bachelor of Technology, Bhagwan Parshuram Institute of Technology, Rohini, Delhi, India

ABSTRACT

Mobile Application Development is getting to be additionally difficult with differing stages and their product improvement packs. Lately, mobile computing has been having truly a revolution. Anyways one of the difficulties that has been conceived due to this revolution is technology and device fragmentation leaving developers stupefied. Platform developers, device manufacturers accompany such a variety of gimmicks and functionalities that it has been hard to give developers a less demanding method for creating applications and running the application on every cell phone with expense and time compelling measures. To lessen the expense of development and connectivity with maximum users across a plethora of platforms, developers are relocating to cross-platform application development tools. In this paper, we give a few choice criteria past the portability concerns toward picking suitable cross-platform solution for application development. Nonetheless, we discovered that cross-platform solutions might be suggested by and large, yet they are still constrained if high prerequisites apply with respect to execution, convenience or native user experience.

Keywords

Cross platform tools; PhoneGap; Titanium; Xamarin; Android; iOS; WP; MAD- Mobile application development

Academic Discipline And Sub-Disciplines

Computer Science and Engineering

SUBJECT CLASSIFICATION

Mobile Computing; Cross-Platform Architecture

TYPE (METHOD/APPROACH)

Experimental; Literary Analysis; Survey

ISSN 2277-3061

INTRODUCTION

The scene of mobile platforms has seen real development in recent past. While BBOS, Symbian, Bada withered and failed to gain any traction, iOS & Android have stunning market and mind share. Additionally Microsoft's Windows Phone may not have high piece of the pie however it still has solidified its third place in the race. In the time of cell phones and tablets, mobile applications are tremendously increasing the value of a few businesses including transport, travel, retail, ecommerce, etc.

The expanding interest for mobile applications in numerous zones presents designers and organizations with a few issues: implementing an application for multiple mobile platforms like Android and iOS, may require high exertion in terms of development time, assets, support and conceivably licenses, tools and deployment. One impacting element is the fragmentation of the market of mobile platforms, i.e. cell phones and tablets. Also the diversity of mobile platforms and the assortment of SDKs and different devices pose extraordinary difficulties. That incorporates decision of SDK, UI and UX of application, stability of framework, ease of upgrading, expense of development for multiple platforms and time to market an application.

Most of the developers might want to have their applications on all major platforms, with a consistent UX over all platforms. At the same time creating native experience for each platform obliges profound learning of every single platform and their SDKs. This methodology likewise raises the expense of development, time to market the application, effort in updating and maintenance of the application. This is where the cross-platform development solutions come into picture.

Table 1. Basic comparison between different approaches of application development

Decision criterion	Native approach	Mobile web approach	Cross platform approach
Quality of UX	Excellent	Very good	Not as good as native apps
Quality of apps	High	Medium	Medium to low
Potential users	Limited to a particular mobile platform	Maximum including smartphones, tablets and other feature phones	Large - as it reaches to users of different platforms
App development cost	High	Low	Medium to low
Security of app	Excellent	Depends onbrowser security	Not good
Supportability	Complex	Simple	Medium to complex
Ease of updating	Complex	Simple	Medium to complex
Time-to-market	High	Medium	Short
App extension	Yes	Yes	Yes

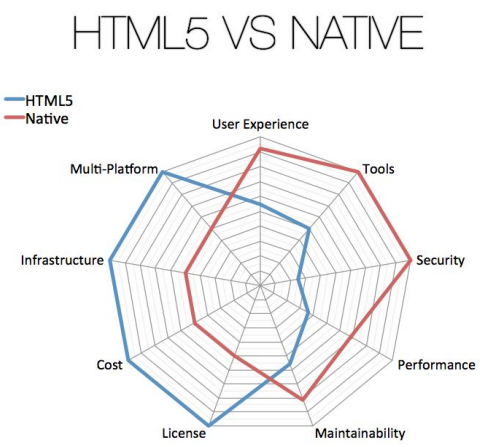

Fig 1: Comparison between HTML5 and Native development approach

Cross platform tools (e.g. Phonegap, Titanium, Xamarin) permit executing an application and its client interface (UI) utilizing web technologies like HTML, CSS, JS or managed code. At that point the application could be fabricated for a numerous mobile platforms (e.g. iOS, Android, Windows Phone, BlackBerry etc.). This methodology permits creating application for various mobile platforms at the same time. With this approach the expense of development and time to market the application is decreased drastically.

This paper introduces a few criteria beyond portability concerns to pick appropriate cross-platform tools for development. Numerous such tools and frameworks are available right now. We have put forward the prerequisites of cross-platform frameworks and the high level architecture of the same. In-depth analysis of most popular solutions enlightens about the API & documentation, deployment, development environment, advantages and shortcomings.

TYPES OF CROSS-PLATFORM FRAMEWORKS

Pure web application frameworks

It ought to be noted that few frameworks exist which basically depend on HTML for client interface designing and in this way could be joined together with pure web application tools. For instance, JQuery Mobile gives a structural architecture to web applications, including predefined user interface segments, page loading and transitions. Other frameworks even contain custom styles for mobile platforms, e.g. components that seem to be like iOS/WP native user interface components.

Partially based on web technology

Frameworks that primary utilize HTML and related languages to display the user interface, however also offer an API for calling device specific functionality, fit in with this classification. Existing systems utilize JavaScript to permit access to native functions, connecting JavaScript function calls and local APIs (e.g. functions of the vendor-provided SDK for the underlying platform). Then again, there are systems that help calling native functions but don't provide the means to execute native user interfaces. Case in point, PhoneGap offers numerous native functions however just utilizes the platform's web view component with a specific end goal to show client interfaces – no UI functionality is incorporated. Such frameworks might be joined together with UI-centred frameworks, for instance SenchaTouch etc.

Compiled or interpreted code

Unlike the previous category, frameworks of this sort don't chiefly utilize web technologies, yet typically fuse an alternate sort of client interface API and a solitary programming language which is then utilized over all supported platforms. For example, the Titanium Appcelerator framework packages applications with a standalone JavaScript interpreter. Web-related APIs like DOM (Document Object Model) manipulation are excluded and rather, Titanium offers its own API for producing a UI from code, and for other functions like network, device and file access. Since these frameworks often support native UI components, contrasts between the platforms' typical UI design must be considered. In this manner, distinctive framework architectures might be found. Titanium offers a abstract interface for common components like buttons, yet specific APIs for platform-specific ones (e.g. iOS toolbar, WP appbar etc.). Other frameworks just offer a programming language for platform-independent code, while requiring UI to be produced independently for every single platform.

Other types

Various existing cross-platform solutions use approaches that contrast from the above classifications. For example, the XMLVM venture has the objective of making an interpretation of programming languages into one another. One illustration use is the cross-compilation of an Android application to a native iPhone application by translating its source code and including a compatibility layer that impersonates the Android APIs on the iOS platform.

REQUIREMENTS OF A CROSS-PLATFORM FRAMEWORK

We have identified the desirable requirements of any cross- platform framework as stated below:

Multiple mobile platform support

The framework must support several mobile platforms. Support for Android and iOS are very essential since they have the largest share in the application markets. Windows Phone and Blackberry 10 are also major platforms in many parts of the world, supporting these would also be a major plus point for the framework. Most of the Cross-Platform frameworks does support all of the four OSes given above.

Rich user interface

As of now numerous cross platform tools provide sub-standard user interface (UI). Since the success of an application relies very much on the user experience of the interface, rich UI development ought to be incorporated. Support for modern graphics, animations, multimedia are vital.

Security

Applications developed by cross platform tools are not highly secure. Proper research needs to be carried out to secure the tools and applications. Many HTML based frameworks have security issues as they rely on the browser component to access the application data. These security loopholes can be fixed by updating the security paradigm of the browser itself.

Support for app-extensions

It is required to install app extensions on top of existing applications like in- app purchase/billing capability. The frameworks need to let developers add extra functionality such as advertising and billing for every platform.

Accessing built-in features

The tools must be able to access the built-in features of a smart device. Use of camera, sensors, geo-localization and more features helps to provide better user experience. Many apps require hardware access for some features to work, so the cross-platform framework should allow for apps to access the hardware.

GENERAL ARCHITECTURE OF CROSS PLATFORM APPLICATION DEVELOPMENT

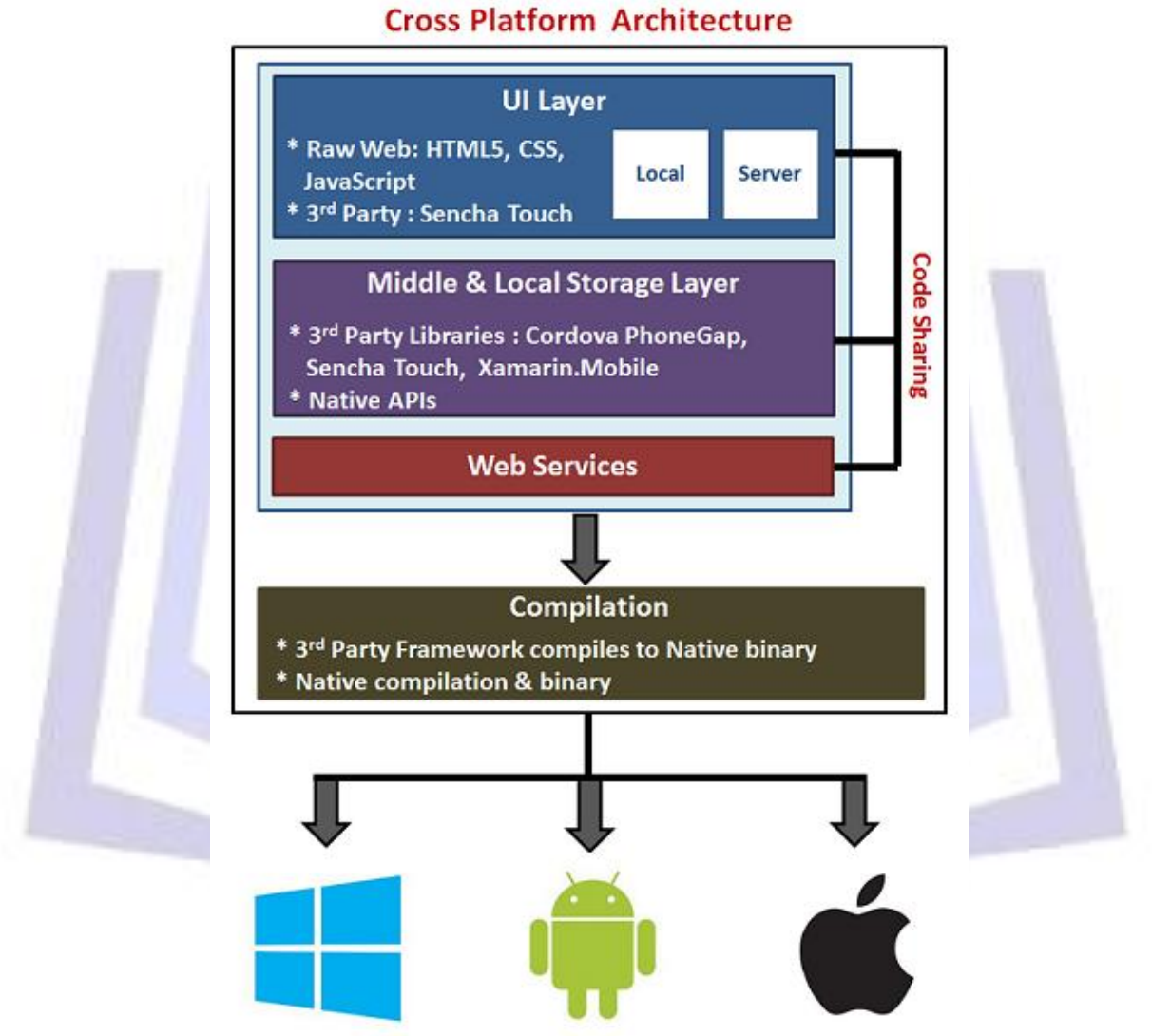

Fig 2: General architechture of cross-platform frameworks

The application developer implements the business rationale or the application functionalities utilizing web technologies or any another backed by the framework. The cross-platform framework permits implementing user interface and access filesystem and device features (sensors, camera, contacts etc.) which interfaces with the framework API (normally JS). The API will thusly interact with the native API of a mobile platform. The application is then compiled independently to produce the executables for mobile platforms. The APIs for the mobile platforms allow generating the respective application. Subsequently the created application might be deployed to corresponding mobile phone.

FRAMEWORK COMPARISON CRITERIA

Level of platform support

This will help a company identify a tool that will help it reach the maximum target audience through the number of platforms a tool supports. A company or a developer that wants to reach Android users should consider or rate a tool that has the best support for Android and similarly for others.

Level of device feature set support

Once a tool is selected or primed to be selected, the next evaluation will be the level of native feature the tool provides for the platform it supports and how good it supports them. Or considering some unique feature one tool provides others can't such as cloud data syncing, social network integration.

Programming language the framework uses

Learning a new programming language can make a developer less productive and as a result could potentially be a cost to a company sponsoring the developer or the development, plus learning a new language from the ground up might probably require a lot of effort and time. Developers must take into account the development language of the tools.

Framework performance in developing

Framework 1 is better than Framework 2 if it uses less code to build the same application. UI Rendering is also very important. We don't want an application to stuck in "Uncanny Valley". UI translation can lead to fatal compromises in UX. Most of the users prefer the UX of their native platform, so frameworks must provide tools to at least mimic the native UI. Also the perceived speed of the running application is very important. The build file size should also be less as compared to other frameworks.

Extensibility of the framework

Any framework, no matter how mature can still need extensions to support new features which are not usually provided. A framework needs to provide basic extensibility features for the developers like Maps, In-app billing, UI controls etc.

Licensing

No one would argue that getting anything free or cheap feels good. But paying a reasonable licence fee for a robust product should also not be a problem. But deploying exorbitant licensing fees would be a huge drawback of any framework.

Security structure

Nowadays most of the user's sensitive data is stored on their smartphones. Any lapse in the security of the application can cause data breach, possible cyber-attacks, theft of sensitive information etc. The security structure of any framework must be robust enough to prevent loss of information and security breach.

SAMPLE APPLICATION

In this simple experiment, the tools will be investigated to show how they implement the same functionality. Number of code lines, how a tool implements user interfaces, application size, compilation speed, how user controls such as buttons are constructed, the symmetry between simulation and actual device implementation of the applications, testing on simulation and running on actual devices should not cause misplacement of the user controls.

The sample application for this case study is a pizza cross platform mobile application. The applications provide facilities to users for selecting pizza type, selecting toppings type, filling delivery address and finally sending the order to a pizza shop.

The application will have four windows:

Pizza selection

This is where a user selects a pizza bread. The pizza bread choices are Hand Made, Natural, Pan Crust, Stuffed Crust, Thin and Crispy Crust and they will be arranged in a horizontally scrollable gallery.

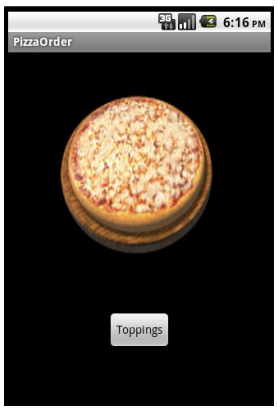

Fig 3: Pizza selection page

Toppings selection

After pizza selection, the application will navigate to this page so users can select the toppings of their preference. The available choices for topping are Bacon, Beef, Italian Sausage and Grilled Chicken.

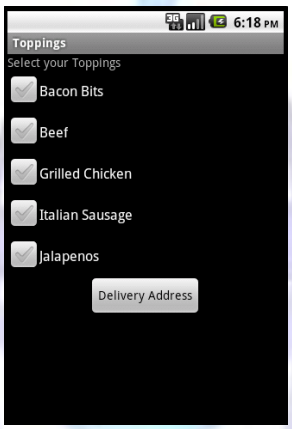

Fig 4: Toppings selection page

Address Filling

Having selected the pizza bread and the toppings, a user can proceed and fill the delivery address. The address is composed of name of ordering user, street, house no, zip code or any other preferred addressing system. The addressing window has four textbox bars.

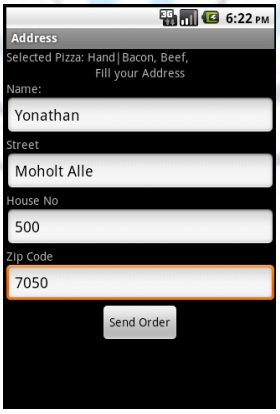

Fig 5: Address filling page

Order submission

This is the last window in the application where a user gets to send the order via SMS to a pizza shop using the address embedded in the application.

Fig 6: Order submission page

PERFORMANCE EVALUATION ON BASIS OF TEST APPLICATION

First we evaluate the frameworks used on the basis of the test application we created. We assess all the necessary criteria which can be used to compare all the frameworks and understand the performance of all the frameworks.

Table 2. Comparison on basis of test application

	XAMARIN	APPCELERATOR	PHONEGAP
Lines of Code	105	243	174
User Controls Implementation	XAML & C#	JS & Titanium Code	HTML & CSS
Perceived Speed	Fast	Slower	Fast
Code Type	C#	JS, Titanium Code	CSS, JS
Syntax Similarity	Java, C++	JavaScript	HTML, CSS, JS
SMS Support	Yes	Yes	Yes
Application Package Size	2.32 MB	20.71 MB	2.05 MB
UI Design	Most Native Like	Native Like	Web Controls Like
Ease of Use	Easy	Easy	Easy
Supported Platform	Major Platforms	Some Platforms	All Major Platforms

As it is clear that in our test of the three most popular frameworks Xamarin comes in at the top for this specific application. Our test consisted of major factors like Ease of use, Application size, UI of app, Speed of the application.

Here only Phonegap was not able to provide native like UI for application because of the web based structure. Only HTML & CSS are available in Phonegap for designing the UI which are not sufficient for an attractive design. Fortunately many add-ons and modules are available which can be added to the phonegap project to enhance the application or the UI (example – JQuery, Sencha Touch etc). We tested our PhoneGap project in conjunction with JQuery and Sencha Touch for CPU usage on android platform. The results are shown in the following table.

The values obtained from CPU 'snapshot' methodology are processed when the application is doing much calculation and is in this manner represented by very high values. Likewise it is to be noted that these values may differ a considerable measure from a millisecond to another since they are snapshot at CPU utilization for short measure of time.

The values acquired from 'top' result shows shifting CPU uses. The min value is constantly 0 as - once the application fetches the requested page, the application does not utilize any CPU processing. The average value is figured utilizing the aggregate elapsed time. It is clear that the first application uses less CPU yet the client experience is not exceptionally sophisticated. At the point when Sencha Touch is utilized alongside PhoneGap, the CPU utilization is all the more yet the client experience is altogether better.

Table 3. CPU usage of PhoneGap(+tools)

Framework & Add-Ons	CPU usage from snapshot approach	CPU usage from 'top' command approach
PhoneGap + HTML + CSS	78.9%	Max: 10% Min: 0% Average: 2%
PhoneGap + JQuery + HTML	79.2%	Max: 44% Min: 0% Average: 10%
PhoneGap + Sencha Touch 2.0	80.3%	Max: 31% Min: 0% Average: 8%

COMPARISON OF FRAMEWORKS ON UNIVERSAL SCALE

After evaluating for our test application the next step would be to assess the frameworks on a broader scope. Here the frameworks will be compared and weighed against numerous criteria which will affect all the developers. The requirements and weights are given below :

Platform Support

3 - Android & iOS

2 - Windows Phone

1 - Blackberry 10

0 - Symbian, Bada etc.(Not relevant now)

UI Rendering

3 - Purely Native

2 - Native Like But Slower

1 - Web Controls

API Support

3 - Very Important (SMS, Location, Network etc.)

2 - Important (Contacts, Notifications etc.)

1 - Nice to have (Compass, Storage etc.)

Technology Learning Curve

3 - Familiar/Popular Programming Languages

2 - Slight Learning Curve

1 - Completely New/Huge Learning Curve

Extensibility

3 - Extremely Extensible With Modules/Open Source

2 - Extensible, But Not That Freely

1 - Commercial Extensibility

0 - Non-Extensible

Licensing Cost

2 - Free of Cost

1 - Commercial Framework

Security :

3 - Secure As Native Application

1 - Security Depend On Other Factors (Browser etc.)

The following table weighs each framework according to each criteria. Max weight of each decision factor is given in the table.

Table 4. Evaluation and weightage of 3 popular cross-platform tools

	MAX WEIGHT	XAMARIN	APPCELERATOR	PHONEGAP
Platform Support	6	5	3	6
UI Rendering	3	3	2	1
API Support	6	6	6	6
Learning Curve	3	3	2	3
Extensibility	3	2	3	3
Licensing Cost	2	1	2	2
Security	3	3	3	1
Total	**26**	**23**	**21**	**22**

In this evaluation Xamarin comes at the top just marginally. Whereas Phonegap supports the most platforms in the market, it does not provide a native UX to the user. All the frameworks supports most of the advanced APIs that are available on the platforms, they can be deployed to. In terms of technological learning, Xamarin uses C# and Xaml, Appcelerator uses JS and PhoneGap uses HTML & CSS, all of which are proven and popular lanuguages. Every framework here is extensible in one way or another. The biggest drawback of Xamarin is the licensing cost. Xamarin is a commercial frameworks whereas Appcelerator & PhoneGap are free and open source for developers.

In our testing we found out that for small projects there is no clear winner between the 3 frameworks. But when the project become larger and more complex, then the distinctions between the frameworks become more pronounced. Because of the managed .net framework Xamarin would be most suited for such projects. But for small projects Phonegap might be preferred because of larger platform support and zero licence fee. Appcelerator can be used effectively for small to medium projects due to support for native UI rendering and JS code framework.

CONCLUSION

All in all the cross-platform development frameworks provides various advantages like code-reuse, fast deployment to all supported platforms, no need to learn a separate programming language for each platform and so on. Despite the fact that now and then the utilization of these stages have their own particular admonitions like non-native UX, slow performance of application, adjusting to another programming environment etc.

In this paper, we have focused on cross-platform development tools since they construct applications for several platforms and the development cost and time to market are diminished. In this current world everybody possesses smartphones and getting to those billions of users is the highest priority for the developers. What's more with the advent of such a great amount of platforms on smartphones it’s hard for developers to focus on each of them particularly. Because of this phenomenon the whole programming world is shifting it’s paradigm to a multi-platform approach. Here we have described the types of cross-platform frameworks, general requirements of cross platform framework and its general architecture.

HTML-based framework types have particular favourable circumstances. For example, Phonegap requires near no additional knowledge for an accomplished web designer: project setup is straightforward and applications could be created with web technologies for a myriad of devices. Anyhow the most major flaw in HTML based frameworks are non-native UI and security concerns. Native compiling based frameworks like Xamarin have added security features as well as native UI and feel. But here the programming is done in C# & XAML, both of which are powerful and popular languages but not as ubiquitous as HTML.

According to Vision Mobile Developer Survey Q3 2014, HTML5 is the most used language at 42% indicating a cross-platform approach. They also found out that 47% of iOS developers and 42% of Android developers are using something other than the native language on their platforms which can be easily a base for cross-platform application development in other languages.

Cross-platform development is a nascent technology, which is developing with a rapid pace. It could be a while before the dust settles and a framework turns out as a winner. Xamarin might be an ideal tool to start with native compiling framework, while with abstraction based cross platform development Appcelerator Titanium or Phonegap might be a good starting point.

REFERENCES

[1] PhoneGap http://phonegap.com/

[2] Appcelerator Inc. Appcelerator Titanium. https://www.appcelerator.com/

[3] Sencha http://www.sencha.com/

[4] JQuery Mobile http://jquerymobile.com/

[5] Xamarin Inc. http://xamarin.com/

[6] VisionMobile Q3 2014 http://www.visionmobile.com/product/developer-economics-q3-2014/

[7] XMLVM. Overview: Android to iPhone. http://www.xmlvm.org/android/

[8] Developer Economics http://www.developereconomics.com/look-4-mobile-ui-frameworks/

[9] Wikipedia http://en.wikipedia.org/wiki/Cross-platform/

[10] Yonathan Aklilu Redda, Cross platform Mobile Applications Development

[11] Survey, Comparison and Evaluation of Cross Platform Mobile Application Development Tools