International Journal of Technical Research and Applications e-ISSN: 2320-8163,
www.ijtra.com Volume 4, Issue 4 (July-Aug, 2016), PP. 13-18

STUDY ON XAMARIN CROSS-PLATFORM FRAMEWORK

Mukesh Prajapati[1], Dhananjay Phadake[2], Archit Poddar[3]
[1,2]Department of MCA, [3]B.Tech. - Information Technology
[1,2]University of Mumbai
[3]Manipal University Jaipur
Mumbai, India
[1]kunalprajapati365@gmail.com
[2]phadakedhananjay@gmail.com
[3]archit.poddar@gmail.com

Abstract— **Mobile Developers of this generation want their apps to be available on all the platforms. For example: - iOS, Android and Windows Phone etc. Since each platform works on a different environment and a completely different language it gets difficult for developers at times. Because of this we can see the beginning of many cross- platform technologies which can help to achieve different platforms. But in reality it is tougher to achieve all the platform-specific Native features and specially to get the performance similar to native applications. With cross-platform technology you can write the business logic once and shared among all the different platforms. Currently there are number of cross-platform frameworks which are available.**

In this paper we are focusing on Xamarin Framework. Xamarin is a cross-platform framework which promises many unbelievable features to build mobile applications. Xamarin claims whatever you used to do with Java, Objective-C or Swift, the same thing you can achieve with C#, a single language used by Xamarin. Accessing all the features of C#, and its .Net libraries you can build Android, iOS and Windows Native apps.

Keywords: **Xamarin, cross-platform, MvvmCross, Xamarin.Forms, Xamarin.iOS, Xamarin.Android.**

I. INTRODUCTION

Over the last couple of years, the usage of mobile technology has increased tremendously. We witnessed that the mobile devices are becoming very important in our lives. At the same time, we are also witnessing the creativity and the dynamics of mobile app development changing day by day with different types of applications. Building mobile applications for each and every platform is very costly and it takes a lot of time for developing them. However with cross-platform technology you can build applications with lesser time and money. Isn't it great! Building a cross-platform application sometimes gets too difficult because whether it can give you the native performance and features of every platform, API supports etc.

1] & 2] Xamarin Framework helps you to build a cross-platform mobile application by offering a single language using C#, class library, and runtime that works across all major mobile platforms of iOS, Android, and Windows Phone . It combines all the power of native platforms and adds a number of powerful features of its own. Using C# language, API and Data structures on an average you can share 70% of app code cross all major platforms and with Xamarin.Forms building user interfaces you can share approximately 100%.

II. EXISTING TECHNIQUES

Prior to Xamarin Framework there are different types of cross-platform tools for building mobile application. The most popular cross-platform technologies are:-

The First is Web-Based frameworks for example Phone Gap/Cordova which allows developers to build applications using HTML/CSS/JavaScript. 4] Among Web-Based Frameworks Phone gap is probably one of well-known cross-platform for mobile applications and pretty easy to develop application. Its owned by Adobe and is open source and is completely free to use which could be the reason of its popularity. With phone gap you can make hybrid applications using HTML, CSS and JavaScript and reuse existing web-development skills to develop apps quickly.

16] The advantage of using phone gap is easy to make application because any phone gap application is simply a collection of HTML pages which is rendered as a web view. Developers who like HTML/CSS/JavaScript find it easy to work on phone gap. 17] A single API can work across all the platforms and 100% code can be reuse. 16] The disadvantage can be many since everything is rendered as a web view, the performance of the app is really bad and it doesn't come close to native apps.

16] The Second is Titanium which is JavaScript based development platform. It uses JavaScript to write the code with native API and User Interface conventions for each platforms. It is a more complicated compare to other cross-platform technologies like Phone gap, Xamarin etc. It currently supports only iOS and Android. 6] Titanium is fully native it solves the problem of delivering great native apps. 16] Since it attempts to write applications with platform specific features using JavaScript the performance of the apps are great. The look and feel of Titanium apps are awesome as compared to other

technology since the User Interface is essentially build natively. 17] With Titanium if you not program for native User Interface 100% code reuse is possible. The performance of the Titanium apps are very good as compared to Phone gap because Compiled code is a combination of native and JavaScript.

7] The Disadvantage of using Titanium is there is no support for using third-Party Libraries mainly because of this many developers don't like it. Json Parsing sometimes takes forever for some reason. 16] It gets very difficult and complicated when developing larger and complex applications. The animations and DOM elements are sometimes laggy and less responsive mainly because it doesn't use HTML and CSS. 17] The Developers should know and Learn about Titanium API .

The above two cross-platform technologies have their own advantage and disadvantages. But Xamarin Framework overcomes most of the challenges faced by these two technologies.

Xamarin allows developers to use C# to write their codes which is later compiled for each supported platform. Xamarin application provides native performance and the look and feel is similar to native apps. 17]

8] Recently Xamarin has been taken over by Microsoft and it seems that Xamarin developers will have a great future. Microsoft integrated Xamarin in their own IDE, Visual Studio and made it open source. It supports MVC and MVVM patterns. Let us see what Xamarin really offers to all developers.

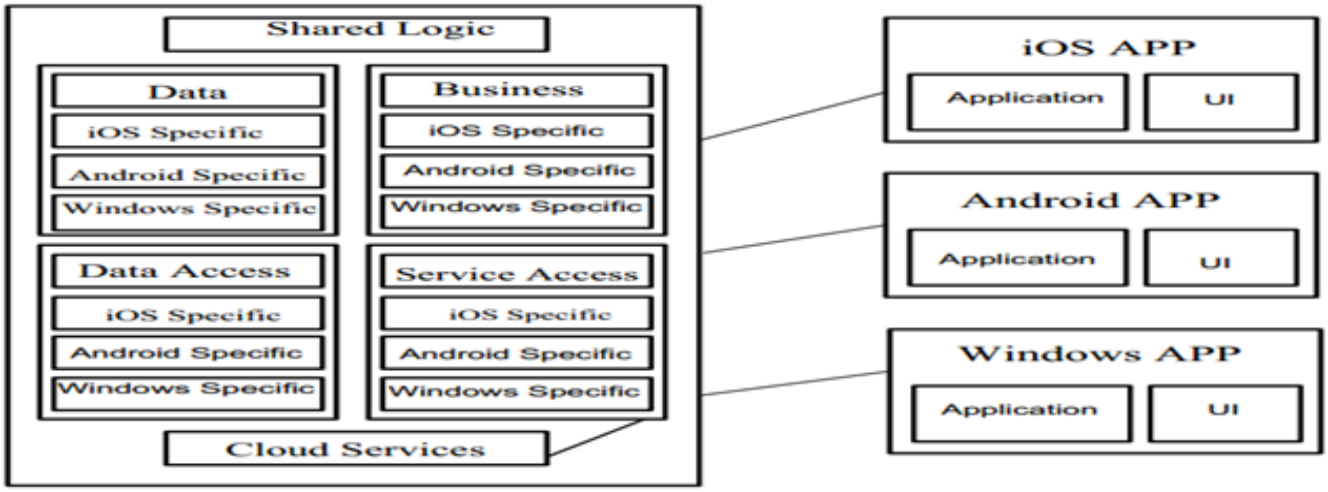

Figure 1 Xamarin Architecture

Understanding the Architecture:-

11] The Architecture of Xamarin project structure is shown in figure 1

User Interface:- It includes the screen, controls and the presentation code. The UI which you build looks, feels and performs much like native applications.

App Layer: - Platform specific features, Binding the business layer with the User Interface.

Business Layer: - It includes all the Business Logic and Business object classes.

Data Access Layer: - Abstraction layer between the business logic and data layer.

Service Access Layer: - Used to access services from complex web services like REST, JSON, and WCF. It encapsulates the networking behaviour and provides simple API to be consumed by the Application and User Interface Layer.

Data Layer: - It includes database binding like SQLite database.

III. XAMARIN FRAMEWORK

comes with lot of features such as:-

A. Xamarin.Android

9] Xamarin Android helps us to develop native android apps using the same Widgets or Controls which you normally use in Java, except with the flexibility and elegance of modern language C#. It gives C# developer leverage to build native android applications using .Net Base class library and two similar type of IDE's: One is Visual Studio and the other is Xamarin Studio. All those people who have already used C#

and Visual Studio will find it very easy to create mobile applications using Xamarin. Xamarin also provides Xamarin Studio, a similar type of IDE which you can use to create applications.

Xamarin. Android supports thousands of API which makes your life easy and gives you native performance. Xamarin also provides Xamarin android player as a emulator to test your applications which is lightning fast. 20] Xamarin. Android user interface can be created in XML files just like Android Studio or programmatically by writing code. Xamarin.Android Designer helps developers to create and modify layouts visually simply drag and drop on the layout. The designer gives a real-time feedback as well, which helps the developer to evaluate the UI before deploying it to the emulator.

Let us see how Xamarin. Android is able to pull off the magic

21] Xamarin. Android apps run within the Mono Execution environment. The Mono Runtime is written in C#. The ART Virtual Machine and the execution environment of Mono runs side by side. Both the execution environments runs on top of the Linux Kernel and exposes different types of APIs to the user code. For Example You could be using System.IO, System.Net, System and rest of .Net base class libraries to access the underlying facilities of Linux operating System. Basically Xamarin.Android apps gets compiled the same way as .Net applications the project first gets compiled into IL (intermediate language). The only difference lies in executing the IL code; It runs in parallel with Mono execution environment and Android runtime side by side.

The Architecture looks like this:

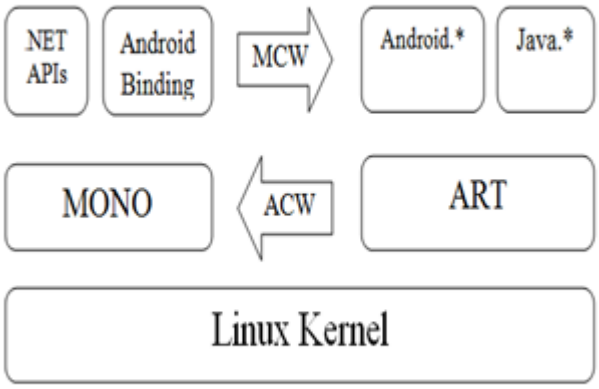

Figure 2 Xamarin. Android Architecture

Developers can access the different features either by calling the .Net APIs that they know very well or using the classes exposed in the Android Namespaces which provides a bridge to the Java APIs which is later exposed by ART (Android Run Time).

- *Mono Assemblies*: Xamarin. Android comes with several assemblies. Silver light is an extended subset of the .Net Assemblies on desktop, Similarly Xamarin. Android is also an extended subset of desktop .Net Assemblies and several Silver Light.
- *Application Packages*: Build System can generate two types of application packages(.apk files) which are:
- *Release*: These are builds, which don't require any additional packages in order to execute. These packages are provided to the App Store.
- *Debug:* It is only used build the project.

B. Xamarin.iOS

10] Xamarin.iOS helps us to build native iOS applications using the same UI controls which you use to build in X-Code and Objective-C. It provides you with the flexibility and elegance of a modern language C# and the power of .Net Base class Library. Now using C# you can make native mobile applications just like Objective-C or Swift.

Building iOS application now is very easy all you need is either Xamarin Studio or Visual Studio. Xamarin studio can be used to build iOS application on Mac. As a C# Developer we all love Visual Studio, you can build your iOS application on Visual Studio which is great, but you would still need a Mac to run the applications. Xamarin for windows helps us to build iOS applications to be written and tested within Visual Studio, with a networked Mac providing the build and deployment service. Developing iOS applications inside Visual Studio provides a number of benefits such as :-

- Creating cross-platform projects for Android, Windows and iOS applications.
- Using your favorite Visual Studio tools such as ReSharper and Team Foundation Server helps you build your application faster for all your cross-platform projects, which includes iOS source code as well which is awesome.
- Working with a familiar IDE, while taking many advantages of Xamarin. iOS bindings of all Apple's API

Requirements & Installation

Few things you need to consider when developing for iOS applications in Visual Studio. A Mac is required in order to compile the, and applications cannot be deployed to a machine without Apple's certificates and code-signing and iOS simulator is required which can only be used on Mac.

As per your requirement you can decide which configuration works perfect for your development needs. These are listed below:

- Use Mac as your development Machine and run a Windows Virtual Machine with Visual Studio installed.

- Use Mac just as a Build Host. In this scenario connect to the same network as your Windows machine with the necessary tools installed.

Now that we know how Xamarin.Android works let us see how Xamarin.iOS works:

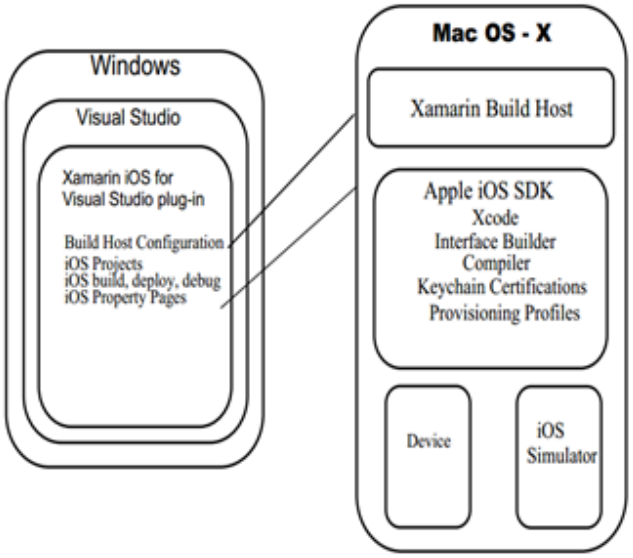

Figure 3 Xamarin.iOS Architecture

- *Xamarin. IOS with AOT (Ahead of Time Compiler)*: All Xamarin.iOS apps are compiled by Mono AOT(Ahead of Time Compiler) directly to native assembly code with apple's requirement, which makes your app native. 2]
- *Update Support*: Xamarin releases same-day support for any iOS update which gives advantage of all the latest iOS features to our developers. 2]
- *Xamarin iOS Designer*: A User Interface which gives a visual designer for the iOS Storyboard where you can simply drag and drop the controls, Interface Builder and Storyboard is fully integrated with Visual Studio and Xamarin Studio. The Designer of iOS is fully Compatible with .xib formats and storyboard so that the changes can be made in either Xamarin Studio or Visual Studio. Xamarin Designer also supports custom controls which renders at the design time in editor. 10]

Currently Xamarin designer is available above Xamarin Studio 5.0 and above OS X and version 3.0 and above Visual Studio.

In Xamarin.iOS you can create the controls programmatically also without using Storyboard. Creating controls programmatically helps when your Apps require everything to be in dynamic.

C. Component Store

23] Component store is where you can download all the libraries required for specific platforms. There are thousands of components available for user interface controls and user interface themes to code libraries. This helps developers to built applications with rich user experiences and integrates with Third Party services. Examples of Components are support library for Android such as Support v4 etc.

D. NuGet

24] C# developers already know what Nuget can do. Nuget is the Popular Package Manager which you can use in Xamarin Studio and Visual Studio for downloading the libraries for the project. In Visual Studio Simply Right-Click on the Project and Click on Manage Nuget packages and search for any library you want and install it. That's it!

E. Mvvm-Cross

MvvmCross was developed by Stuart Lodge and it is an open source plugin or a library. It is based on MVVM design pattern (Model - View - ViewModel) to enhance code reusability across all major platforms.

Working of MVVM pattern:

It separates the logic placed in a View object into two separate objects, one called View and the other is called ViewModel. The View provides the User Interface and the ViewModel are the core classes for providing all common logic for all the platforms.

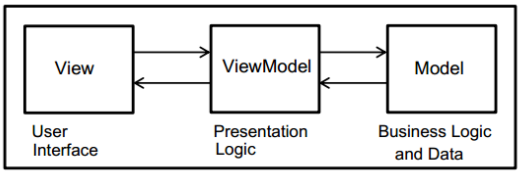

Figure 4 MvvmCross

13] With C# you can create Portable-Class-Library for cross-platform applications this way you can share most of your common code to all the major platforms and reference to the same project. MvvmCross Framework uses the PCL for their ViewModel. MvvmCross supports most of the platforms and provides different types support library's and plugin for each platform and even Third-Party Libraries.

14] It enables Data Binding between the Views and the ViewModel which is a powerful feature of MvvmCross. Data Binding makes your life so easy in each platform For example in Android If you have a ListView you normally create adapter to bind the ListView and the data but with MvvmCross you don't have to create adapter you can simply bind in xml file. Data binding is communication between the View and the ViewModel. In MvvmCross App navigating between different screens is an important capability this can be implemented in the ViewModel so that the navigation logic can be reuse fully. The ICommand property can us help to navigate between different ViewModels For Example ShowViewModel<HomeViewModel>();

F. Xamarin. Forms

22] Xamarin.Forms is perfect for cross-platform applications where you don't want much of native features for each and every platform. With Xamarin.Forms developers can build native UI layouts which can be shared among Android, iOS and Windows Phone and Windows Store Apps. Xamarin.Forms is based on MVVM pattern.

18] Xamarin promises up to 100% code sharing with Xamarin.Form. Xamarin.Forms can be use with full Advantage where code sharing is given more importance and require less Platform specific functionality. For User Interface design Xamarin. Forms Uses XAML. Once you design the layout Connect all the Controls with your shared backend code and get fully native Android, Ios and Windows Phone Apps. During runtime each controls are mapped to platform -specific native UI elements such as Textbox on Windows, UITextView for iOS and EditText for Android.

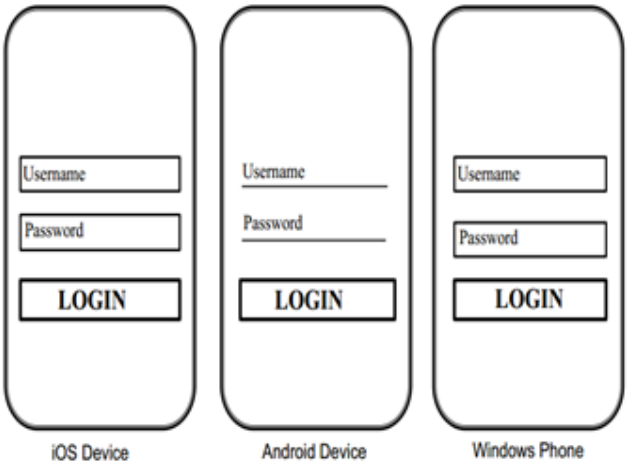

Figure 5 Xamarin.Forms look and feel on all platforms using one UI layouts

IV. Related Work

Building Cross-Platform Apps: - To the extent of our knowledge, our paper is the first one showing how to use Xamarin features to build cross-platform mobile applications. There are few papers in past which showed how to develop applications using cross-platform technologies. Our paper is different from them because we believe no one showed how Xamarin framework works or what its most important features are. Most useful feature about Xamarin is that it uses MvvmCross which makes life easy for developers. MvvmCross suits best for building cross-platform apps because of the features it provides. As we already discussed in section 1] there has been a significant rise in the cross-platform app development technologies and companies do prefer them for building applications because it takes less time and money to develop.

V. Conclusions

Developers want their apps to be available on all the major platforms, the emergence of cross-platform technology gives them advantage. However choosing the right technology is very important because some of the cross-platform technologies are too complex and difficult for developers. For e.g. Titanium is a great framework but very complex for building mobile applications. Similarly, few technologies doesn't give you native features and performance. Xamarin cross-platform provides us to develop native apps with all the features for specific platform and since it requires developers to know only one language i.e. C# that makes life easy for developers. Xamarin comes with lot of unbelievable features for the developers.

VI. Acknowledgements

I would like to thank all the people who helped us and encouraged us for choosing Xamarin as a topic for Research paper and I would also like to thank the reviewers for any comments.

References

[1] Abarbanel, H. D. I. & Brown, R. & Sidrovich, J. J. & Tsimring, L. S. (1993), "The analysis of observed chaotic data in physical systems" Rev. Mod. Phys.6s, 1331 -1392 .

[2] Alligood, K. T. & Sauer, T. D. & Yorke J. A. (1997), "Chaos: An Introduction to Dynamical Systems", Springer-Verlag ,New York, Inc.

[3] Al-Taie, Khalid (2011), "Iraq finalizes contracts to repair Mosul Dam", Mawtani.

[4] Aziz, M. M (1996) " Anew Analysis of Sunspot Time Series " Ph.D thesis, University of Mosul .

[5] Baek, E .G. and Brock, W. A. (1991),"Non Parametric Test For Independence of Multivariate Time series", SSRI.395.

[6] Brock, W.A. & Dechert, W. D. & Lebaron, B. & Scheinkman, J.A. (1996), "A test for Independence Based on the Correlation Dimension" ,SSRI,444 .

[7] Brock, W. A. & Pottert, S. M. (1993), "Non-Linear Time Series and Macroeconmtrics", SSRI.444 .

[8] Chatifield, C. (1975), "The Analysis of time series : Theory and Practice" Chapman and Hall , London.

[9] Falconer, I. & Gottwald, G. A. & Melbourne, I.(2007), "Application of the 0-1 Test for Chaos to Experimental Data", Xulvi – Brunet, SIAM J. 6, No. 2, pp. 395–402.

[10] Gottwald, G. A. & Melbourne, I. (2003), "A new Test for Chaos in Deterministic Systems" , Proc. R. Soc. Lond. A 2004 460, 603-611.

[11] Gottwald, G. A. & Melbourne, I. (2009) , " On the Implementation of the 0–1 Test for Chaos" , Xulvi – Brunet , SIAM J. Appl. Dyn. Syst.(8) 129–145.

[12] Grassberger, P. & Procaccia, I. (1983), "Characterization of Strange Attractor", physics Review letter So:340.

[13] Henk Broer & Floris Takens (2010), "Dynamical Systems and Chaos", Springer- Science+Business Media, Inc.

[14] Maria de Sousa Vieira,(1998)," Chaos and synchronized chaos in an earthquake model", arXiv:cond-mat/9811305v1.

[15] Paul, S. Addison (1997) ," Fractals and Chaos ", Napier University, Edinburgh.

[16] Takens, F. (1981), "Deterministic Chaos in Run of Series", Lecture Note in Mathematics, 898: 366-381 .