UDK 004.4'236

V.V. Gerasimov, S.S. Bilovol, K.V. Ivanova

COMPARATIVE ANALYSIS BETWEEN XAMARIN AND PHONEGAP FOR .NET

Annotation: Comparative analysis between technologies Xamarin and PhoneGap for .NET is discussed. The features of these technologies, their advantages and disadvantages, their sphere of application and prospects of their development are given.

Keywords: Xamarin, framework, .NET, C#, PCL, SAP, PhoneGap, architecture, virtual machine, platform, interface.

Foreword. With growing popularity of modern computer and mobile technologies business community pays more attention to different software products. The ability to be engaged in business everywhere can help to increase profit and minimize time wasting. But presence of huge variety of mobile OS complicates the application development process. This problem is caused by presence of completely different application development tools for appropriate platform. This problem can be solved by cross-platform development frameworks. In the beginning all of them were created for Java platform. But after release of Mono project, which helps to create application for non-Windows OS using C# and .NET Framework, Xamarin and PhoneGap for .NET frameworks appeared. These two frameworks are core of cross-platform development for .NET platform.

The aim is researching and comparing of cross-platform technologies Xamarin and PhoneGap for .NET.

The main part. Xamarin framework is an evolution of Mono project with all of its advantages and disadvantages. Comparing it to PhoneGap for .NET we can see that both of them has its own approach of cross-platform development despite the presence of common elements in both of the frameworks.

To understand all the difference between frameworks, we have to review every framework separately.

In the beginning PhoneGap had no development environment. It was just an extension of infrastructure for open source project called Apache Cordova. PhoneGap for .NET has no difference between PhoneGap, except the ability to work in Microsoft Visual Studio IDE.

ISSN 1562-9945

PhoneGap application is just a mobile website, handled in WebView-element. Despite this, PhoneGap application can use hardware features of phone such as camera, media, internal storage, accelerometer and G-Sensor. Developer has no access to native platform API directly. Nevertheless, developer can use PhoneGap plugins, which work with the native platform API of mobile operating system (Figure 1). If developer can’t find necessary plugin – he has to create libraries for all supported platforms, in order to create plugin.

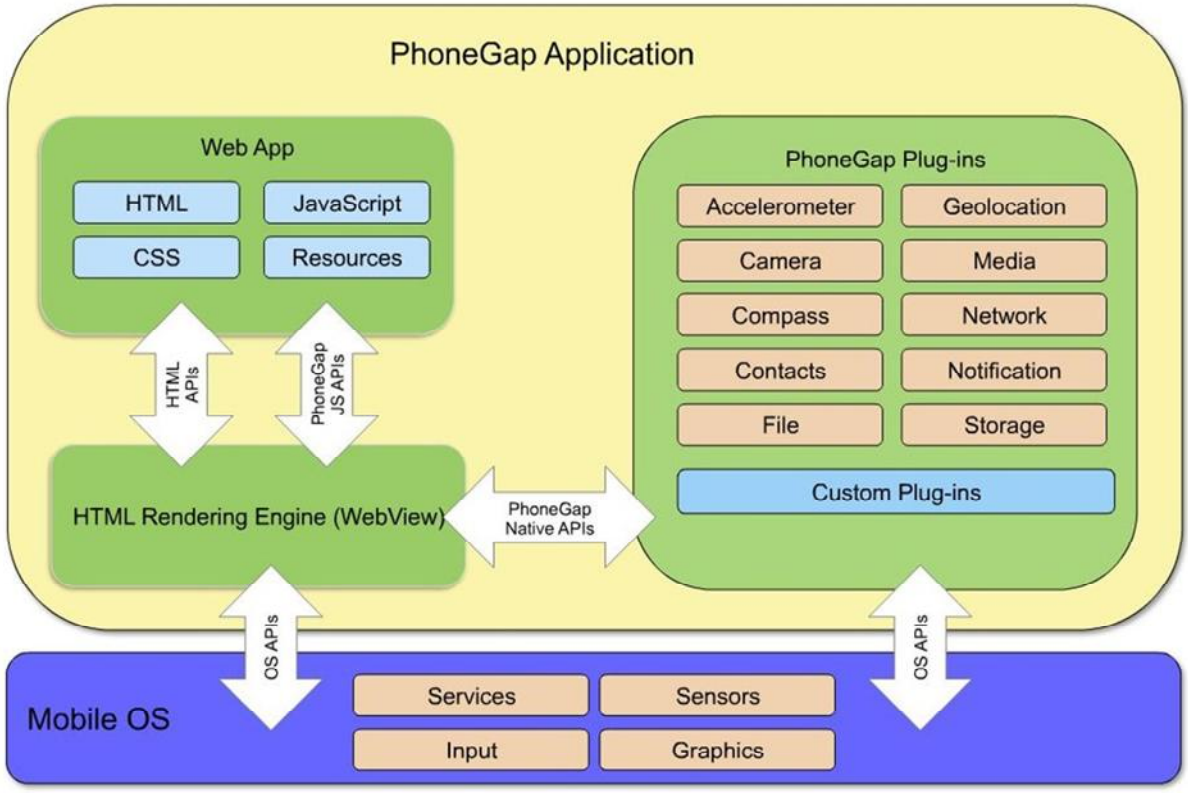

Figure 1 – PhoneGap application structure

User interface is created with HTML, CSS3 and JavaScript technologies. This interface is common for all mobile operating systems supported by application.

The biggest advantage of current approach is complete abstraction from mobile platform what can easily create application with wide list of supported mobile OS. PhoneGap supports Apple iOS, Android, Windows Phone, Blackberry, Symbian, Bada, webOS, Tizen and so on. However in real life big part of hardware functionality isn’t supported for some mobile platforms. Sometimes application can work a bit different on different devices.

Common user interface for different mobile operating systems simplifies and accelerates application development process. However the PhoneGap application

works much slower than natively written application. This is because of PhoneGap application structure.

Another cross-platform development approach is proposed by Xamarin. Unlike PhoneGap Xamarin has tinier connection with native API of OS. It helps to create application with high performance. Xamarin application performance is equal to the performance of application created with platform oriented tools.

Xamarin is divided into several parts: MonoAndroid, MonoTouch and Mono. Biggest difference between MonoAndroid, MonoTouch and Mono is that Mono is just a ported .NET framework for Unix oriented operating systems. Mono helps to create application for Mac and Ubuntu Linux using .NET Framework tools.

The structure of MonoAndroid and MonoTouch is much more complicated. They contain wrappers for native API and a compiler for concrete platform. Every platform has its own compiler, because iOS and Android OS are completely different.

Because of wrapper mechanism MonoAndroid and MonoTouch gives full functionality of Android OS API and iOS API. Close connection between Xamarin and API means close connection between Xamarin application and platform application lifecycle.

This relationship can be seen in the example of Android-app, created by the Xamarin, where together with the application running virtual machine Mono, which communicates with the virtual machine Dalvik (or ART). The presence of additional virtual machines significantly increases the size of the application output file, but does not affect the performance of the application. This is due to the perfect optimization of framework as well as the virtual machine Mono itself.

Due to the close relationship with the mobile operating system, cross-platform development achieved not only by a common writing language for application, but also by the structure of the application.

1. Platform specific code part
2. Shared code part

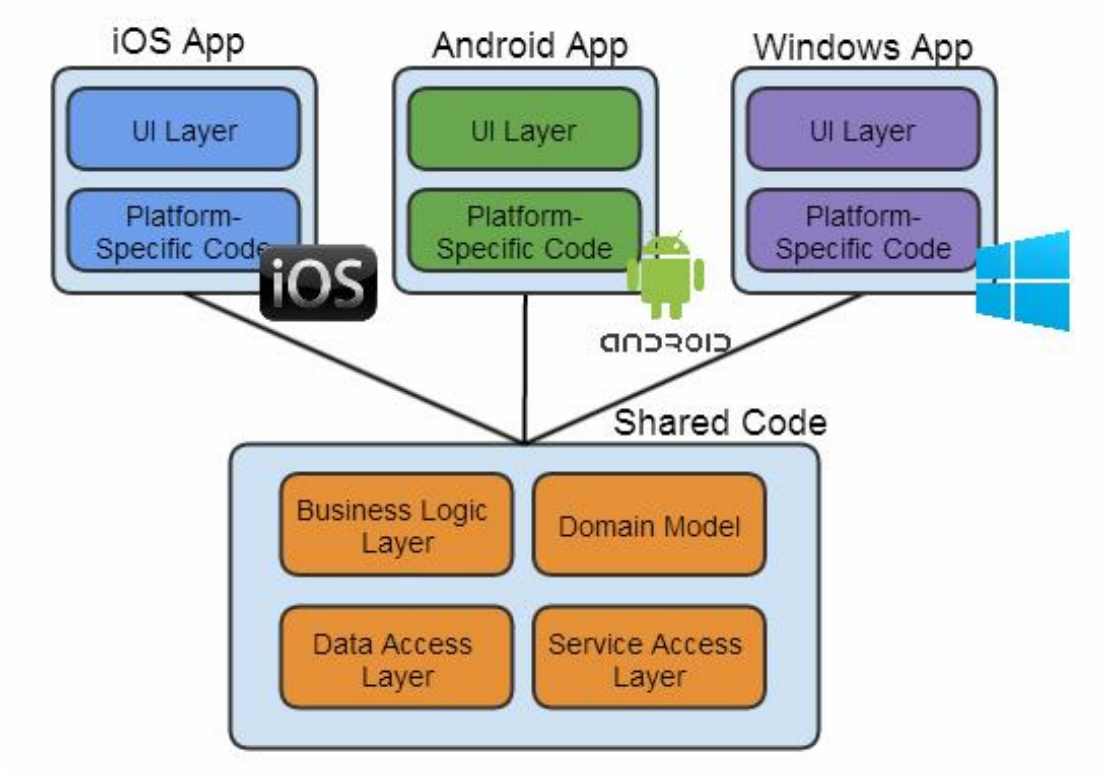

Figure 2 – Xamarin application structure

Platform specific code contains mechanisms specific only for current platform, as such as hardware features support. Platform specific code also contains user interface and its creation and working mechanisms. This problem caused by close enough connection between Xamarin and native API.

To increase code reuse coefficient developer has to divide application into following parts: business logic layer, service access layer, data access layer and so on.

As part of Xamarin framework there were created Xamarin Mobile and Xamarin Forms.

The biggest disadvantage of Xamarin like frameworks is a lack of universal user interface creation tools for all supported mobile platforms, without making changes in order to support target mobile operating system. In order to solve this problem Xamarin Forms was created. Xamarin Forms – is a layer is a tool which unifies a process of user interface creation. Xamarin developers says that Xamarin Forms helps to increase the quantity of shared code to 90% for all supported platforms.

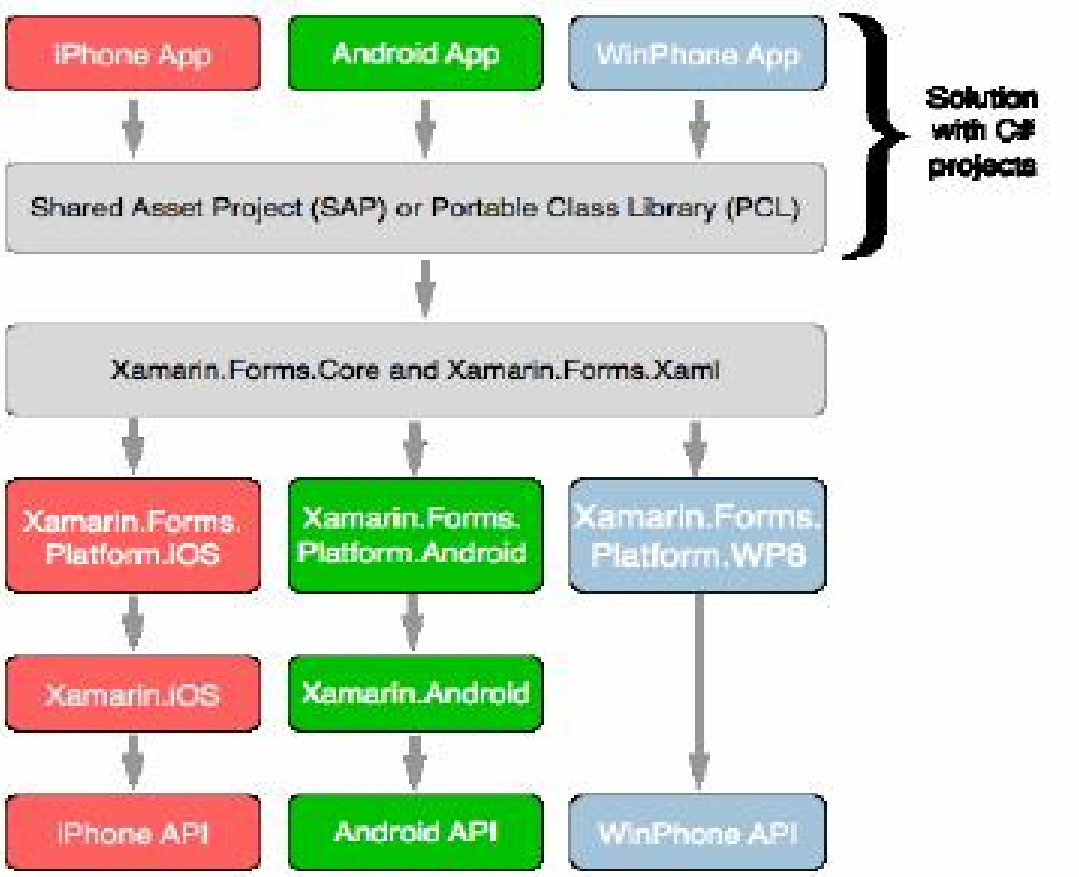

Figure 3 – Xamarin Forms working process

As we can see from Figure 3 – developer cannot reach full platform abstraction. Framework has to understand for what platform application was created. After that implementation is handled by PCL(Portable Common Library) or SAP(Shared Asset Project) which contains shared code for all supported mobile platforms. The main purpose of PCL and SAP libraries is to increase quantity of shared code, but they do it in a different way. In both cases portable library is a part of mobile application. In case of PCL approach all code will be collected into dynamic library and mobile application will use this library during the execution. In case of SAP approach – all code from portable library will be included into Xamarin application during the compiling process.

After that user interface will be analyzed by Xamarin Forms core and sent to API of target platform through adapters. Despite the big amount of iterations, Xamarin application performance is still as high as performance of application created with native tools.

Another functional feature of the framework is the existence of Xamarin Mobile. This library provides an abstraction from specific platform code.

Number of supported platforms by Xamarin compared with PhoneGap much smaller. Xamarin supports only three most popular of them, namely: Android, iOS and Windows Phone.

Despite the small number of supported mobile operating systems , Xamarin has several advantages:

1. Performance level as high as performance level of programs created by native development tools
2. Full support for functionality of native API.
3. Full support of all C# features, including LINQ, lambda expressions, async / await , etc.
4. Large coefficient of program code reuse

Conclusions. Currently market solutions for cross-platform development for the .NET Framework are not very well developed. There are only PhoneGap for .NET and Xamarin. Both technologies have their advantages and disadvantages, and choose a leader among them is practically impossible.

Cheap in developing user interface, a wide variety of available plug-ins with various functionality with a quite good performance make PhoneGap as a good solution to create small cross-platform mobile applications.

However, analyzing the possibilities of Xamarin framework and comparing it to the advantage of the PhoneGap for .NET, it can be concluded that the best choice for large cross-platform projects is just Xamarin. Despite the small number of supported mobile operating systems, Xamarin has a large set of powerful tools that provide both .NET Framework, and native libraries. In addition there is a large set of libraries that accelerate development and reduce the cost of the final product. The price for an opportunity to use the above described advantages is a large size of applications , but more than half of the cases the applications size does not have a critical value.

REFERENCES

1. Cross-Platform Mobile Development: PhoneGap vs Xamarin – Access mode: http://www.justinshield.com/2014/05/cross-platform-mobile-development-PhoneGap-vs-xamarin/
2. PhoneGap vs monotouch for data intensive app - Access mode: http://stackoverflow.com/questions/9191611/phone-gap-vs-monotouch-for-data-intensive-app
3. How to create HTML5 apps on Windows Phone with PhoneGap – Access mode: https://msdn.microsoft.com/en-us/hh771462.aspx
4. Разработка мобильных приложений на PhoneGap и jQuery Mobile – Access mode: http://habrahabr.ru/post/118059/
5. Xamarin.Forms recipes – Access mode: http://developer.xamarin.com/recipes/cross-platform/xamarin-forms/
6. PhoneGap Documentation – Access mode: http://docs.PhoneGap.com/en/edge/guide_overview_index.md.html