ТЕХНІЧНІ НАУКИ ТА ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ
TECHNICAL SCIENCES AND IT

UDC 004.4'22

COMPARATIVE ANALYSIS OF CROSS-PLATFORM FRAMEWORKS FOR MOBILE APPLICATIONS DEVELOPMENT

Yatsenko Roman
PhD student
Sumy State University
Obodiak Viktor
PhD in Technical sciences, associate professor of the Department of computer science
Sumy State University
Yatsenko Valerii
PhD in Technical sciences, associate professor of the Department of economic cybernetics
Sumy State University
Ukraine

Abstract. The article considers the comparison of existing cross-platform frameworks for mobile development. Analysis of their features and interaction with the mobile operating system. Identify key properties for maximum performance.
Keywords: *Cross-platform development, mobile development, programming, mobile applications, Android, iOS, PhoneGap, Xamarin, React Native, Flutter.*

FORMULATION OF THE PROBLEM.

Cross-platform development of mobile applications allows you to increase speed and minimize costs compared to native development. Today, there are several cross-platform development tools that have significant differences. When choosing a tool, you must be guided not only by its popularity or accessibility, but also by the level of complexity of the tasks that it allows to solve.

ANALYSIS OF RESEARCH AND PUBLICATIONS.

Today there are practically no scientific studies on cross-platform frameworks. In one study, Almér A. and Karlsson D. deal with the features of cross-platform mobile development [3].

Also Sventitskiy P. and Ivanova N. compare the differences between the tools for cross-platform development of mobile applications, their advantages and disadvantages [8].

In another study, Sekulovski E. examines various approaches to creating cross-platform applications in order to achieve minimal time costs [7].

A more detailed comparison of cross-platform tools for mobile development was carried out in the work of Ottka S. where not just frameworks but also mobile operating systems are compared [5].

In the existing articles the features of different frameworks are discussed in detail, but no attention is paid to the study of their architecture. Also, the articles do not cover the Flutter SDK, which was released at the end of 2018, and currently it is one of the leaders in cross-platform development.

THE PURPOSE OF THE ARTICLE.

Compare and identify key features of the most popular tools for cross-platform mobile application development. Identify ways to interact with cross-platform frameworks with the mobile operating system and determine the most productive of them.

THE MAIN MATERIAL.

Cross-platform solutions have long attracted those who want to quickly and inexpensively run their product simultaneously for several platforms. The reason is simple — a single code base. It is easier to maintain: artifacts are centralized, there is no duplication of logic and edits of the same bugs for each of the platforms. And fewer people are needed for its support and creation — there is no need to contain two native developers.

Existing frameworks are either complex or not very productive due to the nuances of technical implementation, or full of bugs. With their help, people tend to quickly get the minimum functionality, and eventually doom themselves to rewrite the project in the long term.

Although, compared to cross-platform development, the native provides better-quality applications at the output, there are many new cross-platform technologies that allow you to create great products.

The main advantages of native development are high application performance and access to operating system capabilities.

However, in their work, software engineers often encounter incompatibility of native tools among itself at almost all levels, including programming languages, architecture, working with libraries, etc. Therefore, in order to implement the same algorithms and custom business scenarios, programmers must create an application for several environments in different development languages.

Thus, to write a native application, a company must increase both working time and budget, and efforts to maintain the product.

Technologies of cross-platform mobile development arose to solve these problems. Despite some shortcomings, they greatly simplify the process of writing mobile applications, reducing the labor costs of the company and reducing customer costs.

Today, there are many good cross-platform technologies that allow you to create high-quality applications. In this article we will look at the most popular frameworks for cross-platform development.

Cross-platform development was born from the idea of being able to run a single application on both operating systems without having to rewrite all the code from scratch. According to statistics, by 2020 the market of cross-platform software should exceed $8.5 billion [11]. The introduction of cross-platform has grown with the advent of Facebook's React Native back in 2015.

Today there are many frameworks for cross-platform development, but the most popular and effective are PhoneGap, Xamarin, React Native and Flutter (fig. 1).

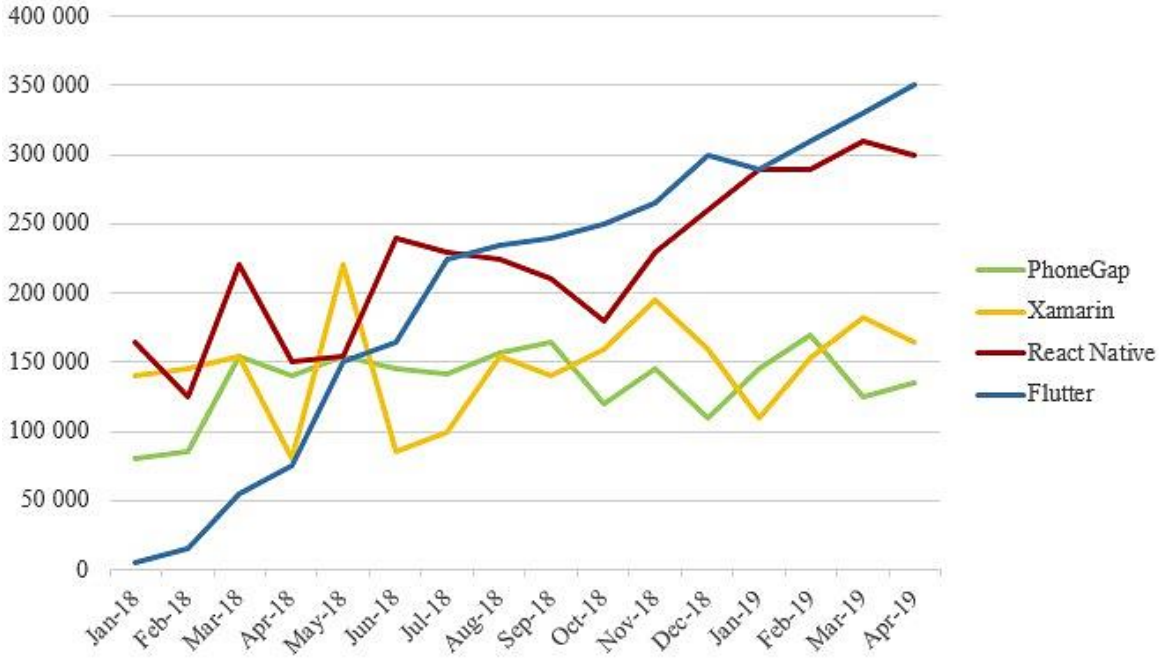

Fig. 1. **Popularity of frameworks for cross-platform development on Google search queries**

Adobe PhoneGap is an open source framework that you can use for free. Also, PhoneGap does not require hardware, SDKs and compilers to create mobile applications.

Today PhoneGap is one of the leading tools of cross-platform development, with which programmers create solutions for CSS3, HTML5 and JavaScript. Another plus of the framework is the set of provided plugins. Using this cross-platform framework, programmers develop applications that are embedded web browsers and single-page HTML pages [2].

Since all interface elements are stylized as native, there is no direct access to the API. To access the system's features, developers use plugins that add JS methods to the web browser and then associate them with a native implementation on each operating system (fig. 2).

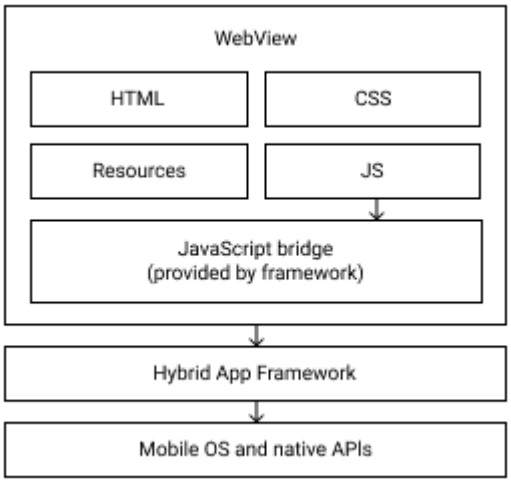

Fig. 2. **PhoneGap architecture**

Adobe PhoneGap is an ideal choice if you need to develop an application interface or a simple application with up to 10 screenshots and for a small audience, for example, some kind of corporate solution for the company's internal tasks. Working with the framework requires the team to experience writing one-page web applications in JavaScript, CSS and HTML.

Another great cross-platform solution that takes a leading position in the market and allows developers to create applications for various platforms is Xamarin. It includes a single common C# codebase and provides the ability to test applications on multiple devices using the Xamarin Cloud [9].

With the advent of Xamarin 2 in 2013, the framework has become one of the most popular cross-platform development tools. It is worth noting that the framework has a strong partner community, which includes such large corporations as Microsoft and IBM. Unlike PhoneGap, it requires a paid subscription, but you can start with a free trial.

Xamarin offers many useful features, such as the native Xamarin Studio IDE and Xamarin.Forms, allowing programmers to use almost 100% of the code written once for all platforms (fig. 3).

The framework also offers access to the native API and allows you to integrate backends such as Parse and Microsoft Azure.

Cross-platform development on Xamarin requires experience on iOS, Android, and C#. The advantage is that as a result, the application will be fully native (although written in C#) and the size of the code base will not exceed 40% [12].

The React Native framework comes from React.js created by Facebook to fix its chat. The interface builds from JavaScript and the adaptive design approach taken from the web turned out to be so successful that the company ported React to mobile platforms. This is how React Native came about.

At the moment, React Native is the most interesting and popular framework due to the fact that it takes the best from the web and React.js.

Thanks to the JS-engine provides high performance, comparable to the native one. Based on the idea of building the interface from blocks, React Native uses neither a browser nor a WebView — only JavaScript API (fig. 4).

So, programmers write code in JavaScript that works with native components of the platform, thereby transferring the benefits and usability of React.js from browser to mobile applications.

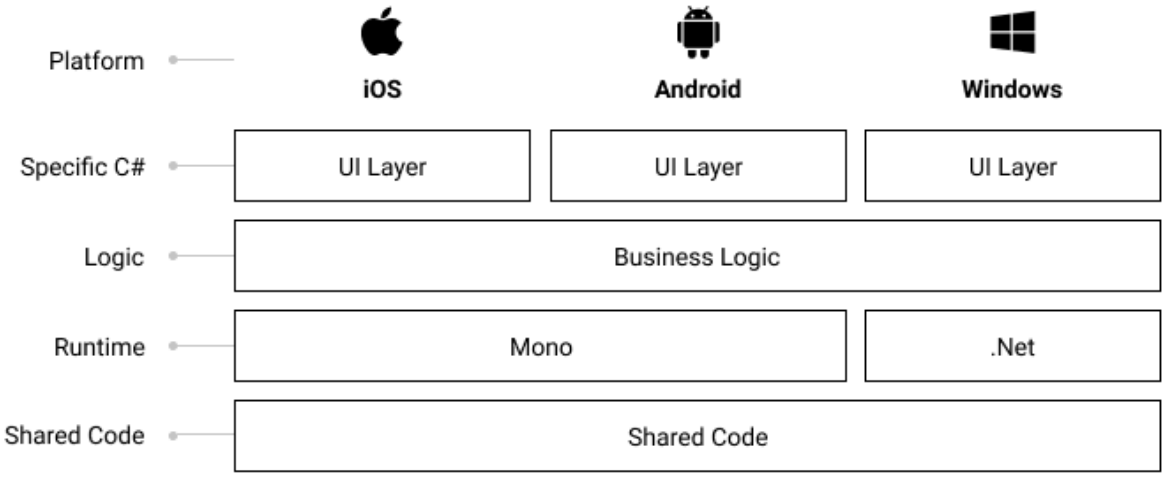

Fig. 3. **Xamarin architecture**

An important advantage of React Native is that the percentage of shared code here is up to 90%, which helps to write modern applications that look native. At the same time, the development is more simple and convenient [6].

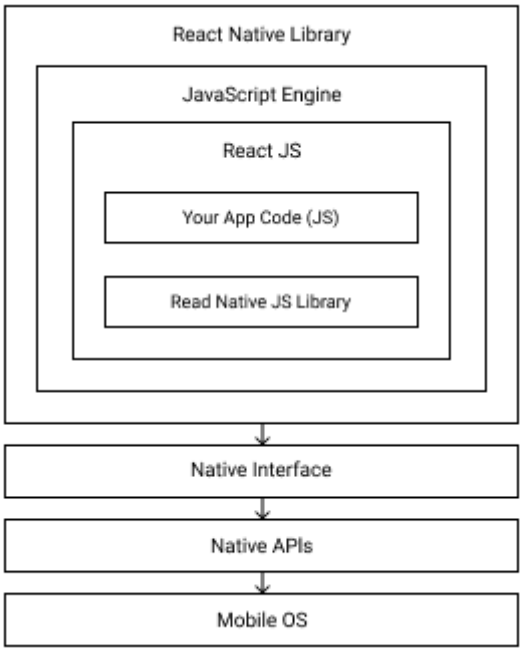

Fig. 4. **React Native architecture**

Flutter is a fairly new cross-platform open source solution released by Google for faster development of beautiful high-performance native apps for iOS and Android.

The framework provides a single code base and uses the Dart programming language, created by Google and having much in common with Java and JavaScript.

Like other popular cross-platform frameworks, Flutter offers many useful features. For example, hot reload simplifies UI development, functional implementation, and product testing.

Flutter includes third-party SDKs, APIs for 2D, animations, native Material Design widgets, and provides the ability to reuse existing Java, Kotlin, Swift, Objective-C code [4].

Despite the fact that this cross-platform solution is still quite young, it is possible to develop modern applications with high performance. Flutter uses its graphical elements to draw through the Skia graphics engine and compiles to native code (fig. 5).

oday, the main favorites of cross-platform mobile development are React Native and Flutter because they allow you to create complex and scalable applications without losing performance. For a better understanding of their differences, it is necessary to consider the features of their interaction with mobile operating systems.

In general, React Native is a JavaScript library, and Flutter is an SDK that works in a completely different programming language called Dart.

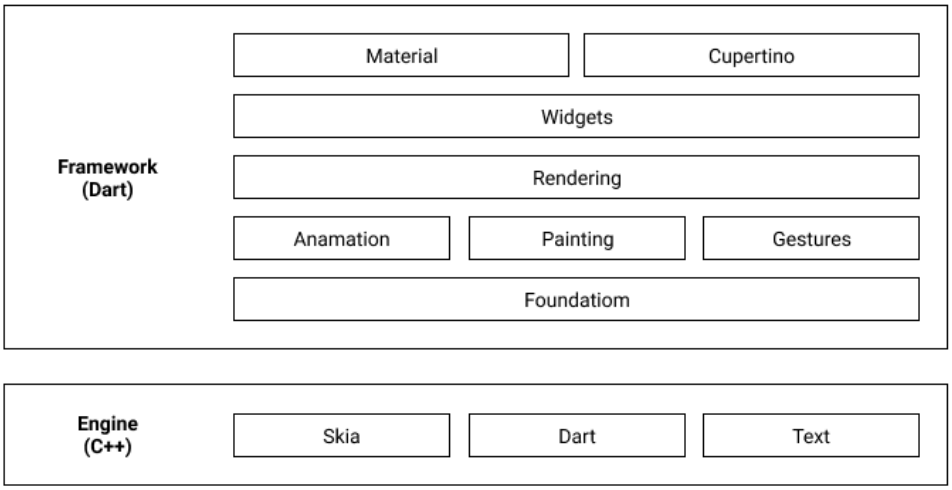

Fig. 5. **Flutter architecture**

React Native compiles its dynamic JavaScript code to native representation at run time. The rest of the code runs on an additional virtual machine that is packaged inside the application itself.

Dart is a General-purpose programming language developed by Google. It can be used to create web applications, servers and mobile applications, as well as for IoT devices. Dart is an object-oriented programming language that supports things like abstraction, encapsulation, inheritance, and polymorphism [10].

Because most of the logical code in React Native is written in JavaScript, it uses a bridge for JavaScript components to control the actual native components in Android and iOS (fig. 6). However, if the goal is to make the applications look the same, then a lot of conditional code will be required for individual platforms to achieve visual integrity [1].

Flutter does not use native components and is not a web view. Flutter draws its own user interface using hardware-accelerated graphics, which is implemented very quickly. Flutter also has material and Cupertino libraries which mimic the native interface language of Android and iOS (fig. 7). The fact that Flutter implements its own user interface greatly simplifies the creation of a more consistent interface on both platforms without any conditional coding.

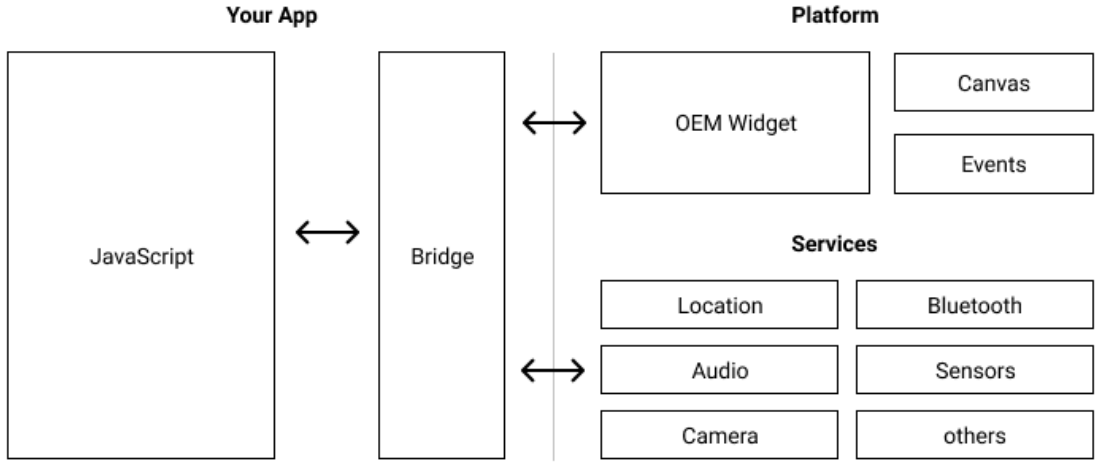

Fig. 6. **React Native interaction with mobile OS**

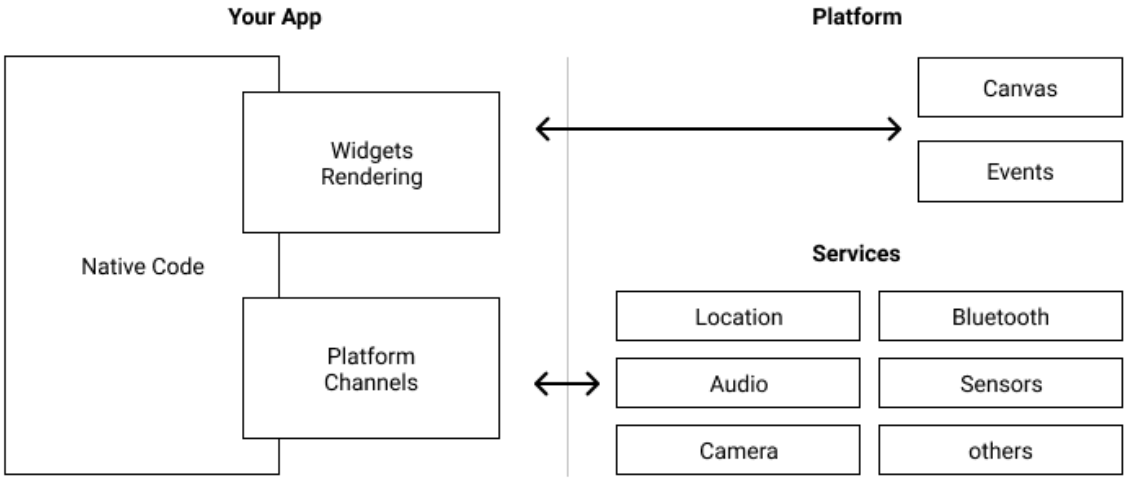

Fig. 7. **Flutter interaction with mobile OS**

Flutter, more precisely Dart, compiles to native code well in advance, which gives it a performance advantage over a cross-platform competitor that uses a JavaScript bridge to compile JS at run time.

Technically, you can achieve the same or similar result by choosing either of the two platforms. One of the key differences comes down to the experience of the development team. React Native is largely based on React JS, and there are already many React JS developers in the world.

Flutter is the best option. Free and visual documentation will help you get started quickly. Error messages are usually easy to understand and easy to identify the source of the problem.

CONCLUSIONS AND SUGGESTIONS.

A variety of cross-platform frameworks allows you to make the best choice for a particular project, given its complexity and knowledge of programming languages of the development team. At the same time, only two frameworks React Native and Flutter are becoming more and more popular, as they allow you to create mobile applications of any complexity without losing performance. Today, the trend of development of cross-platform frameworks for mobile application development shows that less popular frameworks such as PhoneGap and Xamarin will be less in demand, as they do not fully meet the requirements of modern mobile applications. At the same time, React Native and Flutter continue to improve and produce an increasing speed of execution, adapting to the increasing complexity of the tasks of mobile applications.

References:

1. A brief history of React Native. *Medium.* URL: Available: https://medium.com/react-native-development/a-brief-history-of-react-nativeaae11f4ca39 (date of reference: 30.05.2019).
2. Adobe PhoneGap. *PhoneGap faq.* URL: http://docs.phonegap.com/phonegap-build/faq/ (date of reference: 30.05.2019).
3. Almér A., Karlsson D. Cross-Platform Development for Mobile Applications. Lund University : Department of Computer Science, 2013.
4. Flutter. Flutter faq. URL: https://flutter.io/docs/resources/faq (date of reference: 30.05.2019).
5. Ottka S. Comparison of mobile application development tools for multi-platform industrial applications. School of Science : Aalto University, 2015.
6. React Native: What it is and how it works. *Medium*. URL: https://medium.com/we-talk-it/react-native-what-it-is-and-how-it-workse2182d008f5e (date of reference: 30.05.2019).
7. Sekulovski E. Cross-Platform Mobile Application Generation with a Model-Driven Approach Based on IFML and Cross-Compilation. School of Information Engineering. Milan: Como Campus, 2014.
8. Sventitsky P., Ivanova N.A. Instruments of cross-platform development of mobile applications. Innovations in science. Novosibirsk: SibAK, 2014.
9. Understanding the Xamarin Mobile Platform. *Microsoft Docs*. URL: https://docs.microsoft.com/en-us/xamarin/cross-platform/app-fundamentals/building-cross-platformapplications/understanding-the-xamarin-mobile-platform (date of reference: 30.05.2019).
10. Why Flutter Uses Dart. *Medium*. URL: https://hackernoon.com/why-flutter-uses-dart-dd635a054ebf (date of reference: 30.05.2019).
11. Worldwide mobile app revenues. *StatCounter.* URL: https://www.statista.com/statistics/ 269025/worldwide-mobile-app-revenue-forecast/ (date of reference: 30.05.2019).
12. Xamarin vs React Native. *Flush Arcad.* URL: https://www.youtube.com/watch?v=3-FmJk9bFCM (date of reference: 30.05.2019).

COMPARATIVE ANALYSIS OF CROSS-PLATFORM FRAMEWORKS FOR MOBILE APPLICATIONS DEVELOPMENT

Yatsenko R., Obodiak V., Yatsenko V.
Sumy State University
Ukraine

Abstract. The article considers the comparison of existing cross-platform frameworks for mobile development. Analysis of their features and interaction with the mobile operating system. Identify key properties for maximum performance.
Keywords: *Cross-platform development, mobile development, programming, mobile applications, Android, iOS, PhoneGap, Xamarin, React Native, Flutter.*