of statistical methods in the facility control system is proposed. A statistical method is used that implements the statistical hypotheses theory acceptance. As a hypotheses criterion for the adoption, a likelihood ratio criterion was used. It is shown that in the autonomous mode it is necessary to compensate for backlash and random disturbances. The functional diagram of the control system is developed. The architecture of the control program is developed. The cycle of obtaining the driving action and the control cycle of servos are divided. Reduced response time of the system to a change in the set action value. The receiving cycle the master action has a duration of 0.03 ms. The servo control cycle has a value from 8 ms.

Implementation of the developed servo pool management system has shown the effectiveness of the selected architecture for creating software and hardware for radio-controlled robotic systems. Solved tasks to reduce the transmission time of the master object exposure control system to minimize the influence of the time delay cycle master receiving the impact on the duration and servo control loop is provided a synchronous servo control pool.

control system, servo, backlash, object, statistical hypothesis

Одержано (Received) 25.11.2019 *Прорецензовано (Reviewed) 10.12.2019*
Прийнято до друку (Approved) 23.12.2019

УДК 004.4'23:004.057.5 DOI: https://doi.org/10.32515/2664-262X.2019.2(33).208-213

Nikita Shevtsiv, Graduate Student, **Dmitriy Shvets,** Seniour Lecturer, **Nadezhda Karabut,** Seniour Lecturer
Kryvyi Rih National University, Kryvyi Rih, Ukraine
e-mail: i.am.dmitriy.shvets@gmail.com

Prospects for Using React Native for Developing Cross-platform Mobile Applications

The article is devoted to the study of determining the best methods and tools for cross-platform development of mobile applications, allowing to accelerate the process of writing application program code. The analysis of the existing technologies used to write mobile applications is carried out, their advantages and disadvantages are considered. The nuances of using the React Native framework, the mechanism of its work, the advantages of working with it are described. The data on the interest of developers in the React Native framework, search trends, the number of downloads of its installation package have analyzed.

React Native, framework, Android, iOS, mobile application, development

Н. А. Шевцив, магистрант**, Д. В. Швец,** ст. препод., **Н. А. Карабут,** ст. препод.
Криворожский национальный университет, г. Кривой Рог, Украина

Перспективы применения React Native для разработки кроссплатформенных мобильных приложений

Статья посвящена исследованию вопроса определения оптимальных методов и инструментальных средств кроссплатформенной разработки мобильных приложений, позволяющих ускорить процесс написания программного кода приложения. Проведен анализ существующих на сегодняшний день технологий, используемых для написания мобильных приложений, рассмотрены их преимущества и недостатки. Описаны нюансы использования фреймворка React Native, механизм его работы, преимущества работы с ним. Проанализированы данные о заинтересованности разработчиков во фреймворке React Native, поисковых трендах, количестве скачиваний его установочного пакета.

React Native, фреймворк, Android, iOS, мобильное приложение, разработка

Statement of the problem. Traditionally, the process of developing mobile applications has been divided into two parts – creating an application for the Android operating system using the Java programming language, and developing a product for the iOS platform, for which the Objective C programming language was used.

However, the use of this approach significantly increases the time of writing code, accordingly, the cost of developing the application. In this regard, it seems appropriate to search for solutions that allow for cross-platform development of mobile applications, taking into account modern ergonomic requirements and, at the same time, providing the opportunity to significantly reduce the time spent on writing code.

Analysis of recent research and publications. One of the possible options for cross-platform development of mobile applications is the Apache Cordova framework [1]. This framework allows you to create applications for mobile devices using CSS3, HTML5 and JavaScript, instead of using specific API platforms such as Android, IOS or Windows Phone. This is achieved by converting from CSS, HTML and JavaScript to code that any platform perceives as a web element. This approach extends the capabilities of HTML and JavaScript to work with various devices [2]. The result of development is a hybrid application that has the features of both a web and a mobile application. However, the described technology has several disadvantages:

- Existing plugins for Apache Cordova quickly become obsolete and developers need to write independently;
- The application runs in the browser, which causes difficulties in obtaining feedback from the native application;
- Often there are problems with optimizing the application and its fast and stable operation.
- Ergonomics of applications developed using Apache Cordova, loses significantly compared to native applications.

Xamarin [3] is a framework for cross-platform development of mobile applications (iOS, Android, Windows Phone) using the C # language. This technology allows you to take advantage of the C # language in development, allowing you to get an application that practically does not differ in performance from native ones. The disadvantages of Xamarin are the larger application size, limited access to open-source libraries, limited ecosystems, and the complexity of developing applications with high-performance graphics.

PWA technology was developed by designer Frances Berriman and Google Chrome engineer Alex Russell back in 2015. It allows you to add sites the capabilities of applications for mobile devices (in particular, installed on smartphones, work offline, deliver push notifications). PWA applications are quite simple to develop, which leads to cheaper development compared to native applications. However, PWA does not allow access to most functions of the smartphone (calls, SMS, calendar, camera, sensors, etc.), and, in addition, it is not supported by all browsers.

Flutter [4] – Google's development in the form of an open source SDK for creating mobile applications. Flutter, unlike a number of analogues, does not use JavaScript. As a programming language for Flutter, Dart is chosen, which is compiled into binary code, thereby achieving a speed of operations comparable to Objective-C, Swift, Java, or Kotlin [5]. The disadvantages of development can be considered the instability of work, which is due to the relative youth of the project. This also explains the small number of existing libraries for Flutter and information on working with this technology.

Statement of the objective. The objective of the research is to determine the best methods and tools for cross-platform development of mobile applications to speed up the process of coding applications.

Statement of the main material. One of the most promising tools for solving tasks today is React Native [6-9]. This tool was created as a tool that allows you to write program code for a single application and receive a mobile application on various mobile platforms. The tool was designed to fix flaws that made it difficult to use Apache Cordova.

React Native is a Javascript framework based on the React library for developing cross-platform iOS and Android platform applications, released in 2015 by Facebook.

The main mechanism of React Native's operation is that each React Native application has two threads:

• Main thread - runs in absolutely every native application. It handles the display of user interface elements and user gestures.

• JavaScript stream - executes JavaScript code in a separate engine. JavaScript deals with the business logic of the application and defines its functionality.

The so-called bridge, the core of React Native, is responsible for the interaction between the two threads. It serves as an intermediary that directs requests and incoming data from two streams. This approach allows them to interact asynchronously, which ensures stable operation, protecting against mutual blocking of flows.

Other JavaScript solutions for mobile platforms simply wrap your JS code in a web view.

The final application running on the phone uses native code (Java for Android, Objective-C for iOS). The compilation results in the same application as written in the native language: controls, appearance and gestures work the same way as in the native application.

React Native provides high performance, especially when working with fast-changing data. Thanks to its component structure, React Native encourages you to write modular, reusable code. Greater speed is achieved through incremental assembly - only modified modules are reassembled, not the entire assembly.

All changes in the JS code are immediately visible in the simulator, which greatly speeds up development, while in Java, the developer each time has to start the application manually.

In order to develop applications on React Native, knowledge and experience in Java / Kotlin for Android or Objective-C / Swift for iOS are not required. But understanding HTML, CSS and the fundamental knowledge of React itself will be enough for the initial stages of work. This framework turns any web developer into a potential developer of mobile applications, significantly improving the process of creating them.

Used in React JavaScript XML (JSX) is an extension of JavaScript syntax. XML / HTML-like JSX extends ECMAScript so that XML / HTML-like text can coexist with JavaScript / React code. The syntax is intended for use by preprocessors (i.e., transpilers such as Babel) to convert HTML-like text found in JavaScript files to standard JS objects that the engine will analyze. It is also possible to use React Native without it, but the opposite is recommended, since JSX describes the appearance of the UI as clearly as possible, while including all the JavaScript features. Like HTML / XML, JSX tags have names, attributes, and children.

There are 5 ways to style the React Native UI components in 2019:

• Inline CSS
• CSS in JS
• Styled Components
• CSS Modules
• Stylable

The way of styling is the key to modularity, for the possibility of scaling the code base. These styling techniques help to avoid common scalability issues that slow down development speed.

The life cycle of all React Native components is completely different from iOS and Android. If you focus on the differences from native systems, then:

• UI elements themselves change state / appearance when changing input parameters;

• On Android, there is no need to manage onSaveInstantState. React Native does this on its own;

• on iOS, there are no methods that directly explicitly report the moment the application screens appear / hide.

However, React Native has its drawbacks.

Often a project stops building, for example, after:

• React Native kernel updates (including when updating the minor version);

• updates of npm modules;

• Xcode updates;

• CocoaPods updates (a fairly common problem);

• the need to write a large amount of native code for advanced projects.

To assess interest in technologies in the field of mobile application development, we will consider data on their use. The development effectiveness to a large extent depends on the interest of the developer in the technology that he uses to create the application. Figure 1. shows the data of the Medium.com resource [10] about frameworks, libraries and tools that:

- developers like

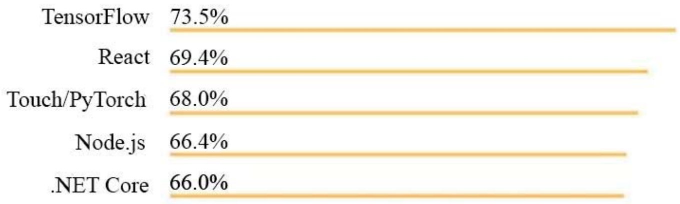

Figure 1 – Attitude of developers to technology

Source: developed by the authors

- wish to study the technology and work with it:

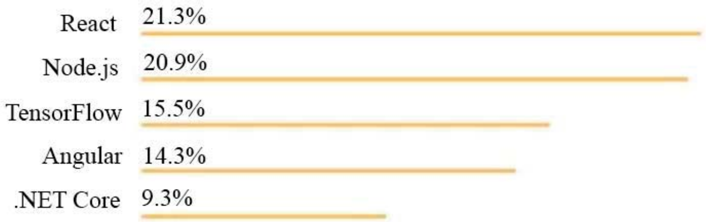

Figure 2 – Interest in the study of technology

Source: developed by the authors

To assess interest in technologies in this industry, we use the following metrics [10]:

- Google search trends [11].

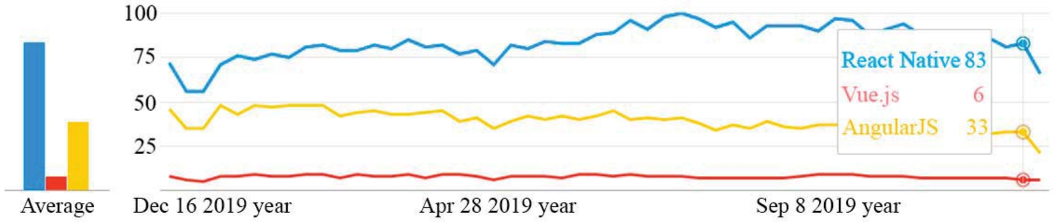

Figure 3 – Google search trends

Source: *[11]*

For search queries, React overtook Angular in January 2018 and maintained leadership throughout the year. Vue.js began to take a visible, but still relatively insignificant position.

• Number of downloads [10]. This parameter allows you to evaluate the actual use: installing the package, as a rule, means its necessity for the workflow.

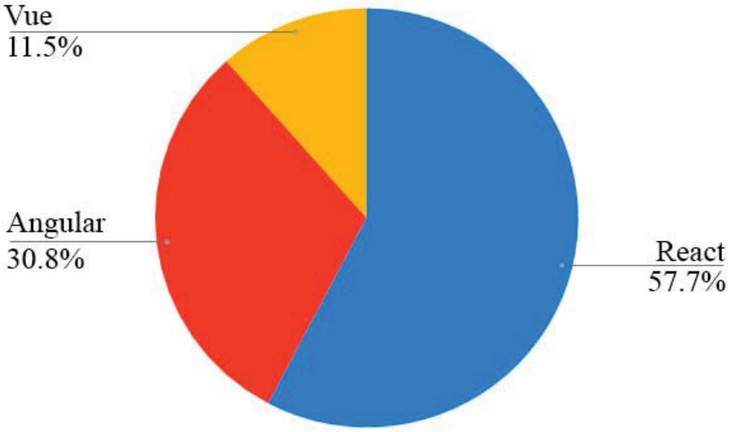

Figure 4 – The number of downloads of installation packages

Source: *[10]*

Conclusions. React Native has a serious development prospect given its ever-growing popularity among mobile application developers. React Native has been constantly reinforcing its position on the labor market and relevance since its development for three years. In addition, studying the React Native framework is useful due to the fact that while studying this technology, students simultaneously encounter a stack of web technologies and the intricacies of developing applications for the Android and iOS platforms. In general, React Native provides high-quality cross-platform mobile development and can significantly increase the speed of developing mobile applications.

References

1. Apache Cordova. URL: https://cordova.apache.org/ (Last accessed: 03.11.2019)
2. Мобильная среда разработки приложений Apache Cordova. URL: https://flexberry.github.io/ru/gbt_cordova.html (Last accessed: 07.11.2019)
3. Xamarin – an app platform for building Android and iOS apps with .NET and C#. URL https://dotnet.microsoft.com/apps/xamarin (Last accessed: 07.11.2019)
4. Flutter – Google's UI toolkit for building natively compiled applications for mobile, web, and desktop. URL: https://flutter.dev/ (Last accessed: 17.11.2019)
5. Flutter. URL: https://habr.com/ru/post/430918/ (Last accessed: 14.11.2019)

6. Темиргалиев Ж.Е. Особенности разработки геолокационных приложений для мобильных устройств с помощью фреймворка React Native. ИнтерКарто. ИнтерГИС: материалы Междунар. конференции 2019. Вып. 25(1). С.414–422 http://doi.org/10.35595/2414-9179-2019-1-25-414-422
7. Lazcano Calixto, Ricardo Neftali, Valencia González, Luis Ángel, Baena Díaz, Daniel Esteban y Venegas Guzmán, Ricardo. (2019). React Native: acortando las distancias entre desarrollo y diseño móvil multiplataforma / Revista Digital Universitaria (RDU). Vol.20 (5). DOI:http://doi.org/10.22201/codeic.16076079e.2019.v20n5.a5
8. BRITO, H. et al. Javascript in mobile applications: React native vs ionic vs nativescript vs native development [javascript em aplicações móveis: React native vs ionic vs nativescript vs desenvolvimento nativo] / IEEE Computer Society, 2018. v. 2018-June, p. 1– 6
9. H. Brito, Á. Santos, J. Bernardino, A. Gomes. Mobile development in Swift, Java and React Native: an experimental evaluation in audioguides / *2019 14th Iberian Conference on Information Systems and Technologies (CISTI),* Coimbra, Portugal, 2019, pp. 1-6. doi: 10.23919/CISTI.2019.8760864
10. JavaScript: топ тем и фреймворков для изучения в 2019 году. URL: https://medium.com/web-standards/top-js-2019-1116718b2b67 (Last accessed: 18.11.2019)
11. Google Trends. URL: https://trends.google.ru (Last accessed: 18.11.2019)

Н. А. Шевців, магістрант, **Д. В. Швець,** ст. викл., **Н. О. Карабут,** ст. викл.
Криворізький національний університет, м. Кривий Ріг, Україна

Перспективи застосування React Native для розробки кросплатформних мобільних додатків

Стаття присвячена дослідженню питання визначення оптимальних методів і інструментальних засобів кросплатформної розробки мобільних додатків, що дозволяють прискорити процес написання програмного коду додатків. Актуальність досліджень полягає в тому, що традиційно процес розробки мобільних додатків складається з двох частин – розробки програмного продукту для платформ Android та iOS на різних мовах програмування. Зазначений підхід значно збільшує затрати часу на розробку мобільних додатків. Поставлена мета полягає в аналізі існуючих на сьогоднішній день засобів розробки кросплатформних мобільних додатків для пришвидшення процесу створення мобільного програмного забезпечення.

В роботі проведено аналіз існуючих на сьогоднішній день технологій, які використовуються для створення мобільних додатків, розглянуті їх переваги та недоліки. Зазначено, використання яких мов програмування вимагають ті чи інші інструменти, розглянуті механізми їх роботи, описані можливості застосування та можливі обмеження щодо їх використання. В дослідженні описано нюанси використання фреймворка React Native як одного з найбільш вдалих рішень для кросплатформної розробки мобільних додатків, механізм його роботи, переваги використання. Проаналізовано дані про зацікавленість розробників у застосуванні фреймворку React Native для створення додатків, пошукові тренди стосовно даної технології, кількість завантажень інсталяційного пакету React Native.

Проведені дослідження показали, що використання фреймворку React Native для розробки кросплатформних мобільних додатків має значні переваги. Створюване за допомогою React Native програмне забезпечення при запуску на мобільному телефоні використовує нативний код та забезпечує високу продуктивність. Крім того, вивчення фреймворку React Native дозволяє одночасно опановувати тонкощі роботи з мобільними платформами та стек web-технологій.

React Native, фреймворк, Android, iOS, мобільний додаток, розробка

Одержано (Received) 25.11.2019 *Прорецензовано (Reviewed) 04.12.2019*
Прийнято до друку (Approved) 23.12.2019