**BELARUSIAN STATE UNIVERSITY OF INFORMATICS AND RADIOELECTRONICS**

**Cross-platform mobile applications development approaches**

Master student Kozyakov A. I.

Faculty of Engineering and Economics

Group 815042

Scientific adviser Klezovich O. V.

Minsk 2019



**АННОТАЦИЯ**

Данная статья рассматривает существующие кроссплатформенные подходы в разработке мобильных приложений и делает попытку получить глобальное представление: в ней подробно представлена категоризация кроссплатформенных подходов, определяются плюсы и минусы каждого подхода, разъясняются примеры решений для каждого подхода, сравнивается кроссплатформенные решения для мобильной разработки.

**ANNOTATION**

This article surveys the existing cross-platform mobile development approaches and attempts to provide a global view: it thoroughly introduces a categorization to the cross-platform approaches, defines the pros and cons of each approach, explains sample solutions per approach, and compares the cross-platform mobile development solutions.

**INTRODUCTION**

In the third quarter of 2014, the sales of smartphones to end users grew 20.3% to reach 301 million units. Besides, smart-phones accounted for 66% of the total mobile phone market. Gartner expects that by 2018, nine out of 10 phones will be smartphones. Smartphone users increase every year because of the variety of mobile applications (Apps) offered to the users in the App stores. With the huge number of smartphone users, there is a growing need to develop more Apps that serve their needs in different fields such as education, health, and tourism. There are many smartphone vendors in the market and each vendor uses a specific platform. These plat-forms include Android, Windows Phone, iOS, BlackBerry OS, Symbian, and others. The aim of any mobile development company is to target as many users as possible by providing the same App for different platforms.

Each platform vendor provides the developers with different Integrated Development Environments (IDEs), programming languages, APIs, and Apps distribution market (store). Therefore, the mobile development company has to choose one of two alternatives to develop the same App for different platforms. The first alternative is that the developers work together to produce one App for a specific platform at a time. This alternative will support the different platforms sequentially, but wastes a lot of development time in learning and getting familiar with the different platforms’ development environments. Another alternative is that the developers are divided into separate teams and each team works for a specific platform. This alternative will support the different platforms in parallel, but may require more developers than the first alternative thus more costly. There is always a need to deliver the App faster and more economically by saving the development time and efforts. This problem leads to the existence of the cross-platform mobile development solutions. The main concept of the cross-platform solutions is to develop the App once and run it anywhere. The cross-platform solutions extend the software development lifecycle by writing the App once and deploying it many times to support different. Many of the cross-platform mobile development solutions are still under research and development. Some solutions are available in commercial use but until now the final solution that solves the entire cross-platform mobile development problem does not exist.

1. Related work

There are many survey papers about the cross-platform mobile development approaches and solutions. The mobile devices have different screen sizes and different aspect ratios, so efforts are required to ensure that the interface is well scalable. The authors encourage the developers to clearly separate the UI definitions from the rest of the code base. The paper compares the advantages of the HTML5 web Apps versus the native Apps. To decide whether to use web or native Apps, the authors define a set of selection factors including the App’s intended features, the target audience, and the development team skills. The paper briefly describes three cross-platform frameworks: JQuery Mobile, PhoneGap, and Titanium Mobile. After using these frameworks to produce cross-platform Apps, it is both the responsibility of the designers and the developers to ensure that the Apps run properly on all the target platforms and devices because none of these frameworks can substitute thorough Apps testing which can be done on actual devices or on various configurations of the emulators provided by the platform IDEs.

Some developers classify the cross-platform mobile development approaches into four approaches: *Web Approach, Hybrid Approach, Interpreted Approach, and Cross-Compiled Approach*. These approaches were described along with the advantages and challenges of each approach. The authors recommend that the developers select the appropriate approach based on the App types. The App types are classified by the authors into server data-driven, sensor/IO based, standalone, and client–server Apps. This paper focuses only on a subset of the existing cross-platform mobile development approaches but does not explain how the existing solutions implement these approaches.

Other developers categorize the mobile web and native Apps into four different configurations: *Native Apps, Hybrid Apps, Dedicated web App* tailored to a specific platform and *Generic mobile app* which is a mobile website that runs on any platform. The mobile web development frameworks provide common characteristics to the delivered App including the following:

1. cross-platform as it supports different platforms;
2. lightweight by lowering the file size inside the framework due to current bandwidth limitations;
3. optimized for touch-screen devices;
4. uses HTML5 and CSS3 standards.

There is another version that proposes a new tool using the Component-Based and Cross-Compiler approaches. The authors classify the cross-platform solutions as follows: the Cross-Compilers based solutions, the solutions based on the Model-Driven Engineering, and finally the Interpreters of source code defined in two categories: the Virtual Machines (VMs) and Web-Based solutions. They use the following categorization methods:

1. *No Categorization*: these survey papers compare a set of cross-platform solutions to help the developer choose the solution that matches his/her needs to deliver the required Apps;
2. *App Categorization*: these survey papers categorize the cross-platform solutions based on the type of cross-platform App;
3. *Approach Categorization*: these survey papers categorize the cross-platform solutions based on their approaches.
4. Mobile application development

Mobile Apps development is a special case of the software development as the developers have to consider different aspects such as short development lifecycle, the mobile device capabilities, mobility, the mobile device specifications like screen sizes, the App user interface design and navigation, security and user privacy, and the Apps require marketing to gain popularity. Many Apps are available in the stores to help for a life based on mobile, or mLife. This includes mTourism, mEnvironment, mEducation, mGovernment, mHealth, mEntertainment, and such like. The mobile App development lifecycle consists of the following:

1. analysis of the App idea;
2. the user interface design;
3. the App development using the tools and programming languages of the target platform;
4. the App testing on different devices, and finally;
5. publishing the App on the target platform store.

New functions or updates to the App are released in successive versions of this App in the target platform store. To develop the mobile App for many platforms, the above development lifecycle is repeated for each target platform except the first step of analyzing the requirements of the App.

If the developer wants to develop the same App for all platforms, the following laboratory equipment are needed: one Personal Computer (PC) and one Mac (used for iOS Apps development), the development tools of each platform and the proper setup, at least one smartphone device and one tablet device for each platform and sometimes multiple devices for the same platform like Android because it is supported by many smartphones vendors. Then, it is required to create one developer account for each platform store to publish the developed Apps. The next subsections explain the following: the restrictions and challenges of the mobile Apps development, the types of mobile Apps, and the cross-platform mobile development.

* 1. Mobile application types

The types of the mobile Apps are web App, native App, and hybrid App.

*Web App*. The mobile web Apps are developed using the web technologies such as HTML, HTML5, JavaScript, and CSS. These Apps do not require to be installed from the store as they are accessed through a URL entered in the mobile web browsers. *The pros:*

1. Easy to learn and develop using the web technologies;
2. All the processing is done on the server and only the UI is sent to the mobile for rendering;
3. The maintenance of the App is simple because the updates of the App and the data are done on the server;
4. The same App is developed once and can run on different platforms using the mobile web browsers.

*The cons:*

1. The web Apps are not avail-able in the store;
2. Internet connection is needed to run the App;
3. The web Apps cannot access the mobile device software and hardware such as camera, and GPS sensors;
4. Less performance because the interpreter language of HTML and JavaScript is parsed and implemented through web browsers.

*Native App*. The mobile native Apps are developed using the tools and programming languages provided for a certain mobile platform. These Apps run only on mobiles with the target platform. The native App can be downloaded and installed from the store. *The props*: have full APIs to access all the mobile device features like camera, sensors, network access, GPS, file storage, database, SMS, and email; higher performance than the web Apps; native look and feel of the user interface. *The cons*: native Apps are more difficult to develop and require a high level of experience; they need to be developed separately for each platform, hence increasing the development time, cost, and efforts; restrictions and costs associated with developing and deploying to certain platforms (i.e. Apple developer license and Apple’s approval to distribute Apps to the iTunes Apps store).

*Hybrid App*. The mobile hybrid App combines the web App and the native App. It is developed using the web technologies like the web App but it is rendered inside the native App using a web view control. The device capabilities are exposed to the hybrid App through an abstraction layer (JavaScript APIs). This App can be downloaded and installed from the store.

*Cross-platform mobile applications development.* The cross-platform solutions help the developers to write the source code once and run the produced mobile application on different platforms. The interpreted App and generated App types are subtypes of native Apps. There are many cross-platform mobile development tools and the target users of these tools differ according to the goals of each tool, like some tools target students for learning mobile development, doctors to customize applications for patients, non-technical people for producing simple applications, and professional developers for producing advanced applications.

1. Cross-platform mobile development approaches

This paper proposes to categorize the cross-platform mobile development approaches into six main approaches: *Compilation, Component-Based, Interpretation, Modeling, Cloud-Based, and Merged*. Also, sub-categorization is proposed for the following approaches: *Compilation, Interpretation, and Modeling*. This categorization introduces an enhancement to the survey papers as follows:

1. Some approaches are not included in these survey papers such as Component-Based approach, Cloud-Based approach, and Merged approach;
2. Some sub-approaches are not included in these survey papers such as Trans-Compiler sub-approach and MD-UID sub-approach.

*Compilation approach*. Compilation is one of the cross-platform mobile development approaches. It consists of two sub-approaches: cross-compiler and trans-compiler. The compiler is a program that transforms the source code written in the source language (high-level programming language) into the target language (lower-level language like the assembly language or machine code). The common use of the compiler is to transform the source code written in high-level programming language to an executable program. The compiler is called cross-compiler when the compiler runs on a computer with OS different from the one on which the compiled program will run. It is called trans-compiler when it transforms one high-level programming language to another high-level programming language.

*Component-Based approach*. Component-Based is one of the cross-platform mobile development approaches. The software component is a package or a module that contains a set of related functions or data. The system functions can be divided into several separate components considering that the related functions are grouped together in the same component. Each component has an interface that specifies the services that can be used by other components. Components communicate with each other via interfaces. Therefore, there is no need to know the inner implementation of the component to use it. The Component-Based approach divides the system functions into a set of separate components with defined interfaces. Each component has the same interface for all platforms, but different inner implementations based on each supported platform. The next subsections explain in more details the following solutions: an unnamed solution that was mentioned in a paper with title ‘‘Component-Based Mobile Web Application of Cross-Platform” and another unnamed solution that was mentioned in a paper with title ‘‘Generic Framework for Mobile Application Development”.

*Interpretation approach*. Interpretation is one of the cross-platform mobile development approaches. It consists of three sub-approaches: Virtual Machine (VM), Web-Based, and Runtime Interpretation. The interpreter translates a source code to executable instructions in real time with a dedicated engine. The developers develop the mobile application once and the interpreter man-ages their execution on many platforms.

*Modeling Approach*. Modeling is one of the cross-platform mobile development approaches. It consists of two sub-approaches: Model-Based User Interface Development (MB-UID) and Model-Driven Development (MDD). The developers use abstract models to describe the functions and/or the user interface of the applications. Then these models are transformed to source code for different target platforms.

*Cloud-Based approach*. Cloud-Based is one of the cross-platform mobile development approaches. In this approach, the application processing is done in a cloud server instead of running the application locally. Therefore, it uses most of the cloud features, including flexibility, virtualization, security, and dynamic management. The client side can use the mobile thin device because only basic processing is required at the terminal. In the cloud environment, thin client devices are lightweight and potentially energy efficient.

*Merged approach*. The main idea of this approach is to merge multiple approaches together to benefit from the advantages of these approaches and minimize the drawbacks of each individual approach. The next subsections explain the following solutions: an unnamed solution that was mentioned in a paper with title ‘‘Component Based Framework to Create Mobile Cross-platform Applications” and ICPMD.

1. Open research areas

Although there are many cross-platform mobile development solutions, they have many limitations and they do not tackle or solve all the challenges of mobile development. Therefore, more research and development are required. This section identifies some of the most promising areas for cross-platform mobile development research, including the following:

*Different Mobile Platforms Support*: the main aim of any cross-platform mobile development solution is to support the different platforms and to simplify the supporting of a new platform. Most of the existing cross-platform solutions focus mainly on the common features of the different platforms and ignore the platform-specific difference.

*Native Programming Languages Support*: many cross-platform mobile development solutions are based on tailored DSLs or specific programming languages (for example, Titanium is based on JavaScript). Therefore, the developer has to learn a new language to use any of these cross-platform solutions.

*User Interface Support*: many of the existing cross-platform mobile development solutions focus to generate the source code of the App and do not support the UI generation. In addition, the cross-platform solutions that are based on the web technologies have a main limitation: the generated Apps lack the native look and feel.

*Source Code Reuse*: most of the existing cross-platform mobile development solutions do not support the code reuse of the existing legacy Apps. Therefore, the developer has to rewrite these Apps in order to upgrade them to a higher version of the same platform, or port them to other mobile platforms.

*Generating Full Mobile Applications*: there is a need to introduce solutions that consider generating full mobile applications instead of focusing on source code or user interface transformations only.

**CONCLUSION**

The cross-platform mobile development solutions extend the software development lifecycle by writing the mobile application once and run it on different platforms to save the time and efforts of the developers. Many of these cross-platform solutions are still under research and development.

These cross-platform solutions are based on different approaches. This paper surveys the cross-platform approaches, including Cross-Compilation approach, Virtual Machine approach, Model-Driven Development approach, and Web-Based approach. In addition, this paper proposes sub-categorization for several approaches and includes the most recent approaches: Component-Based approach, Cloud-Based approach, and Merged approach.

The mobile application types include web App, native App, and hybrid App. The native and hybrid Apps are more widely used than the web Apps because they can be downloaded from the App stores. Titanium and Xamarin are commercial solutions that are widely used to produce native Apps. These solutions are based on the Runtime Interpretation approach. PhoneGap is a solution that is widely used to produce hybrid Apps. This solution is based on the Web-Based approach. However, until now the final solution that solves the entire cross-platform mobile development problem does not exist.

The new trend that will probably play a major role in the future is the Merged approach, which merges multiple approaches together to benefit from the advantages of these approaches and minimize the drawbacks of each individual approach.

This article helps the researchers to know the existing cross-platform mobile development approaches and the open research areas in this field. This paper provides detailed descriptions for each approach, including the following:

1. describe the main idea of the approach and its sub-approaches;
2. define the pros and cons of the approach/sub-approach;
3. introduce a sample of the cross-platform mobile development solutions (per approach/sub-approach) and their limitations.

**BYBLIOGRAPHY**

[1] Taxonomy of Cross-Platform Mobile Applications Development Approaches [Electronic resource] – Ain Shams Engineering Journal, Volume 8, Issue 2, 2017. – Mode of access: [https://www.sciencedirect.com/science/article/pii/ S2090447915001276](https://www.sciencedirect.com/science/article/pii/%20S2090447915001276). – Date of access: 10.03.2019

[2] Model Driven Development Approaches for Mobile Applications: A Survey [Electronic resource]: Mobile Web and Intelligent Information Systems. – MobiWIS, 2017. – Mode of access: [https://www.sciencedirect.com/science /article/pii/S2090447915001276](https://www.sciencedirect.com/science%20/article/pii/S2090447915001276). – Date of access: 11.03.2019

**GLOSSARY**

|  |  |
| --- | --- |
| Abort | Сбрасывать |
| Accept | Принять |
| Accommodate | Вмещать |
| Adapt | Адаптировать |
| Adequate | Достаточный |
| Advanced | Продвинутый |
| Advantages | Преимущества |
| Agc (automatic gain control) | Автоматическая регулировка усиления |
| Algorithm | Алгоритм |
| Amount | Количество |
| Amplify | Усиливать |
| Analog | Аналоговый |
| Analog-to-digital converter | Аналого-цифровой преобразователь |
| Api | Интерфейс программного приложения |
| App platforms | Программная платформа |
| App stores | Магазин приложений |
| Approximate | Приблизительный |
| Apps | Приложения |
| Architecture | Архитектура |
| Aspect ratios | Соотношение сторон |
| Asynchronous transmission | Асинхронная передача |
| Attempt | Делать попытку |
| Attenuation | Затухание |
| Audibility | Слышимость |
| Audible | Слышимый |
| Background | Фон |
| Bandwidth | Пропускная способность |
| Battery drain | Потребление батареи |
| Benefit | Польза |
| Binary | Бинарный |
| Bit | Бит |
| Buffer | Буфер |
| Byte | Байт |
| Cancel | Отменить |
| Capability | Возможность |
| Categorization | Категоризация |
| Characteristic | Характеристика |
| Checksum | Контрольная сумма |
| Circuit | Цепь |
| Classification | Классификация |
| Cloud-based | Основанный на облачных технологиях |
| Commercial solutions | Коммерческие решения |
| Component-based | Основанный на компонентах |
| Compression | Сжатие |
| Compute | Вычислить |
| Concept | Концепция |
| Cons | Минусы |
| Consideration | Рассмотрение |
| Conversion | Преобразование |
| Cross-platform | Кросс-платформенный |
| Css | Каскадная таблица стилей |
| Date input | Ввод данных |
| Decoder | Декодер |
| Default | По умолчанию |
| Define | Определять |
| Definition | Определение |
| Delete | Удалять |
| Demand | Требовать |
| Develop | Разрабатывать |
| Development | Развитие |
| Device | Устройство |
| Digit | Цифра |
| Digital | Цифровой |
| Disable | Отключить |
| Discrete | Дискретный |
| Discussion | Обсуждение |
| Distort | Искажение |
| Domain | Сфера |
| Download | Загрузка |
| Drawback | Недостаток |
| Dynamic range | Динамический диапазон |
| Echo | Эхо |
| Emulators | Эмуляторы |
| Enhance | Усиливать |
| Erasing | Стирание |
| Examine | Проверять |
| Exposure | Воздействие |
| Extend | Расширять |
| Extension | Расширение |
| Feedback | Обратная связь |
| Fidelity | Достоверность |
| Fine-tuning | Тонкая настройка |
| Format | Формат |
| Formatting | Форматирование |
| Frame | Кадр |
| Frameworks | Бибилиотека компонетов |
| Frequency | Частота |
| Grid | Сетка |
| Hearing device | Слуховой аппарат |
| Hexadecimal | Шестнадцатеричный |
| High-level programming language | Язык высокого уровня программирования |
| Hybrid app | Гибридоное приложение |
| Impression | Впечатление |
| Inherent | Свойственный |
| Inner implementations | Внутренняя реализация |
| Install | Установить |
| Instruction | Инструкция |
| Integrated development environments (ides) | Интегрированная среда разработки |
| Intensity | Интенсивность |
| Interpreter | Интерпритатор |
| Interrupt | Прервать |
| Involve | Включать |
| Javascript | Язык программирования |
| Join | Соединение |
| Label | Метка |
| Letter | Буква |
| Level-shift | Уровень сдвига |
| Library | Библиотека |
| Life time | Срок службы |
| Lifecycle | Дизненный цикл |
| Limitation | Ограничение |
| Limitations | Ограничения |
| Loudness | Громкость |
| Magnitude | Величина |
| Measurement | Измерение |
| Memory | Память |
| Merged | Смешанный |
| Message | Сообщение |
| Miniature | Крошечный |
| Mismatch | Несоответствие |
| Mlife | Жизнь основанная на мобильном использовании |
| Modeling | Моделирование |
| Modem | Модем |
| Modify | Видоизменять |
| Mpo (maximum power output) | Максимальная выходная мощность |
| Multimedia | Мультимедиа |
| Native app | Нативное приложение |
| Negative effects | Негативные последствия |
| Network | Сеть |
| Node | Узел |
| Noninvasive | Бесконтактый |
| Oblige | Обязывать |
| Output | Выход |
| Overflow | Переполнение |
| Package | Набор инуструментов |
| Peak | Пик |
| Perception | Восприятие |
| Peripheral | Периферийный |
| Permission | Разрешение |
| Portable | Переносимый |
| Power supply | Источник питания |
| Precision | Точность |
| Probe-microphone | Акустический зонд |
| Processing | Обработка |
| Processing | Преобрабование |
| Pros | Плюсы |
| Protocol | Протокол |
| Pts (permanent threshold shift) | Постоянный сдвиг порога |
| Publishing | Публикация |
| Quality | Качество |
| Quantization | Квантование |
| Quarter | Четверть |
| Reading | Считывание |
| Record | Запись |
| Reduction | Сокращение |
| Repeatedly | Неоднократно |
| Reproduce | Воспроизводить |
| Requirements | Требования |
| Research | Исследование |
| Residual | Остаточный |
| Responsibility | Ответственность |
| Return | Возврат |
| Round | Округлять |
| Router | Маршрутизатор |
| Scheme | Схема |
| Sensorineural | Нейросенсорный |
| Signal | Сигнал |
| Simulate | Моделировать |
| Skin | Оболочка |
| Solution | Решение |
| Solutions | Решения |
| Source code | Исходный код |
| Source code reuse | Переиспользование кода |
| Spl (sound pressure level) | Уровень звукового давления |
| Standalone | Закрытое приложения |
| Storage | Хранилище |
| Sub-approaches | Подподход |
| Subsections | Подраздел |
| Support | Поддержка |
| System | Система |
| Technology | Технология |
| Temporary | Временный |
| Testing | Тестирование |
| Theme | Тема |
| Thoroughly | Подробно |
| Threshold | Порог |
| Tool | Инструмент |
| Tools | Инструменты |
| Touch-screen | Сенсорный экран |
| Trans-compiler | Транс-компилятор |
| Transducer | Преобразователь |
| Transforms | Преобразовывать |
| Transparency | Прозрачность |
| Treatment | Лечение |
| Tts (temporary threshold shift) | Временный сдвиг порога |
| Ui | Интерфейс пользователя |
| Update | Обновление |
| Url | Интернет адрес |
| Utilize | Использовать |
| Vendors | Поставщики |
| Version | Версия |
| Vm | Виртуальная машина |
| Web app | Веб-приложения |
| Web view | Браузерный вид |
| Web-based | Основанное на веб |