International Journal of Advanced Smart Convergence Vol.8 No.4 188-193 (2019)
http://dx.doi.org/10.7236/IJASC.2019.8.4.188

IJASC 19-4-23

Design and Implementation of a Low-Code/No-Code System

Chang Young Hyun

Professor, Dept. of Smart IT, Baewha Women's University, Korea
cyh@baewha.ac.kr

Abstract

This paper is about environment-based low-code and no-code execution platform and execution method that combines hybrid and native apps. In detail, this paper describes the Low-Code/No-Code execution structure that combines the advantages of hybrid and native apps. It supports the iPhone and Android phones simultaneously, supports various templates, and avoids developer-oriented development methods based on the production process of coding-free apps and the produced apps play the role of Java virtual machine (VM). The Low-Code /No-Code (LCNC) development platform is a visual integrated development environment that allows non-technical developers to drag and drop application components to develop mobile or web applications. It provides the functions to manage dependencies that are packaged into small modules such as widgets and dynamically loads when needed, to apply model-view-controller (MVC) pattern, and to handle document object model (DOM). In the Low-Code/No-Code system, the widget calls the AppOS API provided by the UCMS platform to deliver the necessary requests to AppOS. The AppOS API provides authentication/authorization, online to offline (O2O), commerce, messaging, social publishing, and vision. It includes providing the functionality of vision.

Keywords: *Low-Code, No-Code, MVC, DOM, OSMU, O2O, AppOS*

1. Introduction

The growth rate of smartphones is most evident in the mobile device market, and it is expected to continue expanding in the future. The rise in smartphone usage has led to an increase in the number of mobile-based applications[1]. Gartner analyzed that the demand for applications is growing at a rate more than five times the ability to develop apps. The application market is expected to grow to $ 6.3 trillion by 2023 while it is analyzed that the trend with low-code and no-code development tools that make it easy for non-experts to develop apps would be prevailed [2]. The coding of applications developed by non-profession user is a strategy for creating application programs as contents by completely removing technical elements while designing objects to be implemented just by means of a GUI tool in an execution engine environment where one app

Manuscript Received: November. 25, 2019 / Revised: December. 3, 2019 / Accepted: December. 8, 2019
Corresponding Author: cyh@baewha.ac.kr
Tel: +82-2-399-0822, Fax: +82-2-737-6726
Professor, Dept. of Smart IT, Baewha Women's University, Korea

operates the same as in all OS environments. The key technology in the aforementioned environment is implemented by developing and providing new architecture and technology for separating a program into „functions" and „logics" in application software to store, manage and execute them [3].

In the visual integrated development environment (IDE), users drag and drop application components to connect together and develop mobile or web applications. We can design and build powerful applications that efficiently scale the work of related departments without writing code [4].

This paper is about environment-based low-code and no-code execution platform and execution method that combines hybrid and native apps. In detail, this paper describes the Low-Code/No-Code execution structure that combines the advantages of hybrid and native apps. It supports the iPhone and Android phones simultaneously, supports various templates, and avoids developer-oriented development methods based on the production process of coding-free apps and the produced apps play the role of Java VM. The AppOS, which uses the cross-platform development technology, which is run-time, is shared by running on the runtime environment. As it is not related to the operation state, it is necessary to use the Low-Code/No-Code development technology to minimize development and management costs. It chose the cross-platform development environment that uses the same implementation language. Low-Code/No-Code is developed as a front-end technology of the web and can be run in a typical browser, which has the effect that a conventional browser can run a Low-Code/No-Code app in place of AppOS.[5-8].

2. Related studies

Hybrid applications are essential for constant content fluctuations, app updates, and when we want to introduce new technology standards such as HTML5. In native applications, we have native app developers with powerful app functionality in a single platform. must do it. Therefore, the necessity of Low-Code/No-Code system of cross-platform environments that can be developed even for cost-effectiveness and efficient development is also required.

The LCNC development platform is a visual integrated development environment that allows non-technical developers to drag and drop application components to develop mobile or web applications. Non-analysts, office managers, small business owners, and non-software developers can quickly build and test applications.

The LCNC platform does not know the development work required for app authors to build existing programming languages and configurable components of the platform, and all developers will have a user-friendly graphical user interface that can be tested by connecting components and third-party application interfaces. By rearranging the modules, we can test iteratively until the application works as expected.

In general, no-code platforms are specialized types of low-code cloud platforms where the specific visual components required support industry-specific functions, specific lines of business, or corporate branding of specific companies. Low code platforms, on the other hand, require developers to make minor changes to their backend code to make new apps developed compatible with other business software. There is a shortage of experienced software developers and the need for a Low-Code/No-Code system in a cross-platform environment due to the long duration of the development project. Platforms with little or no code continue to grow, and analysts at Forrester Research predict that the low-code market will grow to $ 15 billion by 2020 [3].

3. Low-Code/No-Code System Components

The common components for Low-Code/No-Code execution that combines the advantages of hybrid and native apps in the environment-based Low-Code/No-Code execution platform combining hybrid and native apps are shown in Figure 1.

It provides the functions to manage dependencies that are packaged into small modules such as widgets and dynamically loads when needed, to apply MVC pattern, and to handle DOM. As a common component for Low-Code/No-Code execution, various open-source libraries are used for Low-Code/No-Code app execution and can be replaced with other libraries that perform the same function.

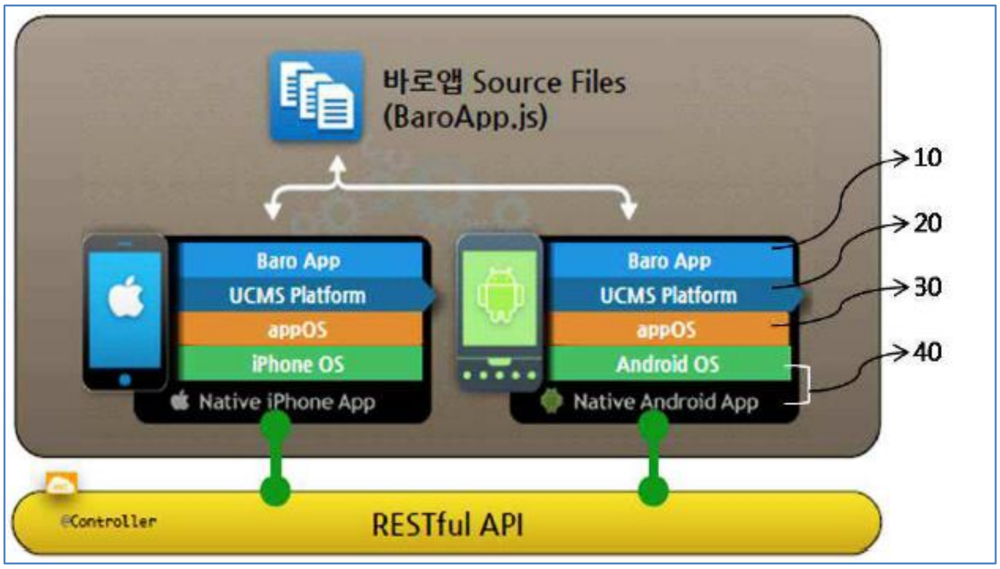

Figure 1. Low-Code/No-Code System Outline Diagram

The UCMS platform middleware located in the middle of the app and AppOS and the sub-components located at the bottom of the UCMS platform. It is developed in the OSMU (One Source Multi-Use) method because of the characteristics of the cross-platform. The AppOS code consists of code, and the component including AppOS can be configured differently according to the selected cross-platform development environment and has a structure that can execute several Low-Code/No-Code apps. One Low-Code/No-Code app runs inside AppHost and the other Low-Code/No-Code app runs under AppOS and AppOS, where Low-Code/No-Code apps run by AppRenderer implemented as a web rendering module. It is located in and includes the code of the lower stage that is configured depending on the characteristics of the selected cross-platform.

4. Low-Code/No-Code System Function

In the Low-Code/No-Code system, the widget calls the AppOS API provided by the UCMS platform to deliver the necessary requests to AppOS. The AppOS API provides authentication/authorization, online to offline (O2O), commerce, messaging, social publishing, and vision. It includes providing the functionality of Vision.

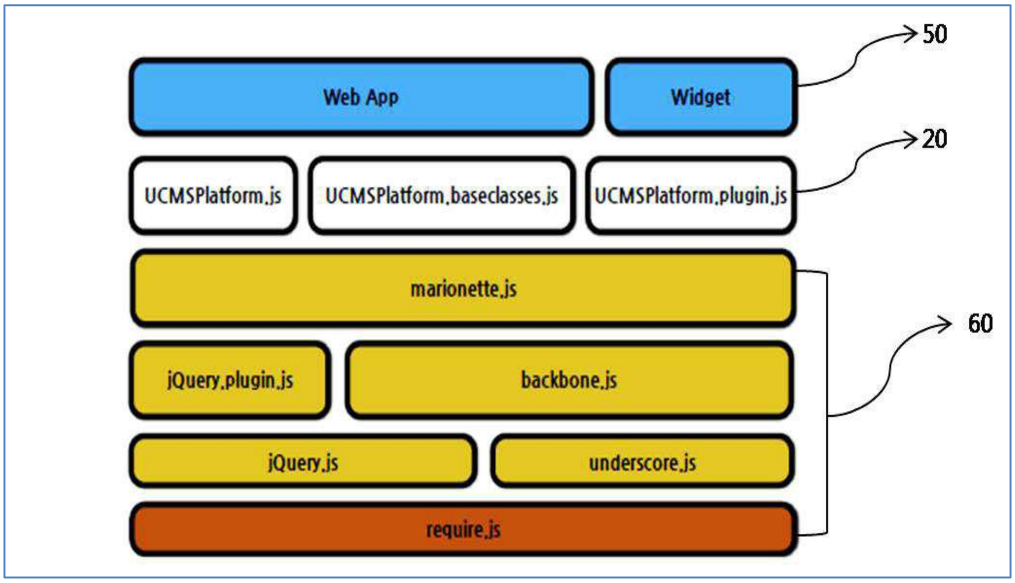

Figure 2. Low-Code/No-Code System Function Diagram

In Figure 2, the number 20 represents the UCMS platform, 50 for the widget, and 60 for an open source-based general-purpose library. The environment-based Low-Code/No-Code execution method that combines a hybrid app and native app consists of the initialization process, calling process, and phone process.

4.1 Initialization Process for AppOS

7 steps for the AppOS initialization process include the steps of starting the first AppOS initialization, executing the Low-Code/No-Code system delivered to the second scheme, and thirdly, the parameters delivered with the Low-Code/No-Code system execution request. Receiving, receiving a fourth option to initialize the back-end session for the back-end API call, performing the initialization for the fifth messaging API call, and performing the administrator initialization of the sixth Low-Code/No-Code system, The seventh step consists of implementing the Low-Code/No-Code system.

4.2 Call process for AppOS API

The AppOS API call process involves implementing the AppOS API call through the AppRenderer Proxy that corresponds to the client requesting the command. The second AppOS API call is converted into a request event that AppOS can receive and relayed to the AppOS. The relayed AppOS API call is interpreted by the AppOS stub code corresponding to the other commander receiving the command request and is processed by the event handler. Step 5, the processed result is returned to the widget through the Deferred object implemented in the asynchronous call pattern Promises.

When the requirement to transfer status from the AppOS side to the Low-Code/No-Code system occurs, a request event can be generated from AppOS to the Low-Code/No-Code system. Low-Code/No-Code system is executed based on the environment that combines a hybrid app and native app while processing requests.

4.3 Switching process for Low-Code/No-Code System

Low-Code/No-Code system switching process is a process after the request for the execution of the Low-

Code/No-Code system. It consists of firstly the event that the extinction event occurs continuously, the second event occurs when the newly requested Low-Code/No-Code system starts, and the third time the newly requested Low-Code/No-Code system is executed.

5. Conclusion

The environment-based Low-Code/No-Code execution platform we proposed that combines hybrid and native apps combine the advantages of hybrid and native apps to package them into small modules such as widgets. It can be replaced with other libraries that use the same functions, and are used by various open-source libraries to provide the ability to manage dynamically loading dependencies, to apply MVC patterns, and to provide and execute functions for DOM processing. -Code & No-Code The UCMS platform (middleware) located in the middle of the system and AppOS and the sub-components located under the UCMS platform are developed using the OSMU (One Source Multi-Use) method, which is a characteristic of the cross-platform. It contains an AppOS code that consists of the same code, regardless of which component contains AppOS. On it may be differently configured according. One Low-Code/No-Code app can be run within AppHost while the other Low-Code/No-Code app is executed by AppRenderer implemented as a web rendering module. It characterized in that it comprises a code of the lower stage is configured depending on the characteristics of the cross-platform.

Firstly, it as a development effect is an execution structure that combines the advantages of hybrid and native apps. simultaneously supporting iPhone and Android while it supports various templates, so there is no need for developers to produce coding-free apps (applications). Second, it runs on top of a runtime environment that acts as a Java VM, allowing us to share AppOS using cross-platform development technologies. Third, the runtime AppOS can induce a minimization of administrative costs due to the existence of various cross-platform development environments. Fourth, it is developed by the front-end technology of the web and can be run in a general browser so that the browser can run an app in place of AppOS.

Acknowledgement

This work was financially supported by Baewha Women’s University

References

[1] H. S. Kim, "A Study on Usability Improvement of Mobile Healthcare Services," *International Journal of Advanced Smart Convergence* Vol. 6 No. 2 pp. 72-73, 2017
DOI: https://doi.org/10.7236/IJASC.2017.6.2.72

[2] *http://blog.naver.com/PostView.nhn?blogId=simon9627&logNo=221168371306*

[3] Y. H. Chang, "A Study on the Public Data Activation Strategy based on App Developed by Non-Profession User," *International Journal of Advanced Smart Convergence*, Vol. 6 No. 1, pp. 33-34, 2017
.DOI: https://doi.org/10.7236/IJASC.2017.6.1.32

[4]*https://www.forbes.com/sites/johneverhard/2019/01/15/what-really-is-low-codeno-code-evelopment/#3b9c2ada2a8e*

[5] *https://searchsoftwarequality.techtarget.com/definition/low-code-no-code-development-platform*

[6] E. S. Choi, M. S. Kang, Y. G. Jung and J. K. Paik, "Implementation of IoT-based Automatic Inventory Management System," *International Journal of Advanced Culture Technology*, Vol. 5, No. 1, pp. 70-75, 2017.
DOI: https://doi.org /10.17703/IJACT.2017.5.1.70

[7] J. Y. Park, "A Study on Automatic Service Creation Method of Cloud-based Mobile Contents," *International Journal of Internet, Broadcasting and Communication*, Vol. 10, No. 4, pp.19-24, 2018.

DOI: http://dx.doi.org/10.7236/IJIBC.2018.10.4.19

[8] J. S. Lee, M. S. Pyo, S. C. Kwon and S. H. Lee, "A Study on SNS Applications in Broadcasting Based on Analysis on KBS'Reaches on Facebook," *International Journal of Internet Broadcasting and Communication*, Vol. 9, No. 3, pp. 27-34, 2017.
DOI: https://doi.org/10.7236/IJIBC.2017.9.3.27

[9] W. C. Jun, "A Study on Programming Ability Assessment Tool Development for the No-Programming Experienced," *International Journal of Internet, Broadcasting and Communication*, Vol. 9, No. 1, pp.56-63, 2017.
DOI: https://doi.org/10.7236/IJIBC.2017.9.1.56

[10] J. Lee and Y. H. Seo, "International Journal of Internet, Broadcasting and Communication," Vol. 10, No. 3, pp. 5-41, 2018.
DOI: http://dx.doi.org/10.7236/IJIBC.2018.10.3.35

[9] H. S. Lee and H. W. Lee, "Simulated Dynamic C&C Server Based Activated Evidence Aggregation of Evasive Server-Side Polymorphic Mobile Malware on Android," International Journal of Advanced Smart Convergence, Vol. 6, No. 1, pp.1-8, 2017.
DOI: https://doi.org/10.7236/IJASC.2017.6.1.1