Visión Electrónica Vol. 16 No. 1 (2022) • January – June • p.p. 126-134 • ISSN 1909-9746 • E-ISSN 2248-4728 • Bogotá (Colombia)

UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS

Visión Electrónica
*Más que un estado sólido*
https://doi.org/10.14483/issn.2248-4728

VISIÓN ELECTRÓNICA

A CASE-STUDY VISION

# Architecture of a 2D Game for Android and IOS Using Cocos2DX

*Arquitectura de un Juego en 2D para Android y IOS Usando Cocos2DX*

*Luis Felipe Wanumen-Silva* [1], *Gloria Andrea Cavanzo-Nisso* [2], *Miguel Ricardo Perez-Pereira* [3]

INFORMACIÓN DEL ARTÍCULO

**Historia del artículo:**
Enviado: 30/10/2021
Recibido: 09/11/2021
Aceptado: 13/03/2022

**Keywords:**
2D games
Cocos2DX
Mobile games

ABSTRACT

In this paper the architecture and tools used in the development a 2D game that works on both Android and IOS is presented. To do this, Cocos2DX is used; this tool allows to development 2D game with portability and power required to run on these two mobile environments. It is important to note that also shows how modeled and implemented in Cocos 2DX a game, 2D Labyrinth, serves to illustrate how Cocos2DX handles all processing OpenGL graphics and thus no problems when running submitted in various mobile devices. Finally, the game operation using the proposed architecture and tools is showed.

**Palabras clave:**
Juegos 2D
Cocos2DX
Juegos móviles

RESUMEN

El presente artículo muestra la arquitectura y las herramientas usadas en el desarrollo de un juego 2D que funciona tanto en Android y IOS. Para ello, se usa la herramienta Cocos2DX la cual permite el desarrollo de juegos 2D que tienen la portabilidad y potencia necesaria para ejecutarse en estos dos ambientes móviles. También se muestra cómo se modeló e implementó un juego, llamado Laberinto 2D, en Cocos 2DX. Esto ilustra cómo Cocos2DX se ocupa de todo el procesamiento de gráficos OpenGL y de esta forma no se presenten inconvenientes al ejecutarse en diversos dispositivos móviles. Finalmente, se muestra el funcionamiento del juego aplicando la arquitectura y las herramientas propuestas

1BSc. in Systems engineering, Universidad Distrital Francisco José de Caldas, Colombia, MSc. in Systems and Computing, Pontificia Universidad Javeriana, Colombia. Professor at Universidad Distrital Francisco José de Caldas, Colombia. E-mail lwanumen@udistrital.edu.co

2BSc. in Mathematics, Universidad Nacional de Colombia, Colombia, MSc. in Mathematical Sciences, Universidad Nacional de Colombia, Colombia. Professor at Universidad Distrital Francisco José de Caldas, Colombia. E-mail gacavanzon@udistrital.edu.co

3BSc. in Electronic Instrumentation and Control Engineering, Universidad Distrital Francisco José de Caldas, Colombia. MSc. in Educational Sciences, Universidad de San Buenaventura, Colombia. Professor at Universidad Distrital Francisco José De Caldas, Colombia. E-mail: mrperezp@udistrital.edu.co

Cite this article as: L. F. Wanumen-Silva, G. A. Cavanzo-Nisso, M. R. Perez-Pereira, "Architecture of a 2D Game for Android and IOS Using Cocos2DX", *Visión Electrónica*, vol. 16, no. 1, pp. 126-134, 2022. https://doi.org/10.14483/22484728.18956

## 1. Introduction

Today, mobile applications are becoming increasingly important [1], which has led to the development of many mobile applications dedicated to games [2]. This does not mean that any application for being mobile is entertaining [3], but there are applications that, because they are game type, should be entertaining. In the market there are a number of languages that come and go and that are analyzed about their future, past and present [4] [5] [6]. However, depending on the type of game, it is better to follow a specific type of architecture [7], that is, not all games that run on mobile devices have the same architecture [8]. It is necessary to evaluate multiple aspects before planning to select a tool [9] [10] [11]. The advantage of developing applications for mobile devices is that in the case of university students, it becomes a topic of great interest that covers various areas [9], not only the IT areas [12] and in the case of pedagogy, games are a useful pedagogical tool [13] [14] [15] in the case of 2D games on mobile devices, there are aspects so particular that they differentiate them from other games, which makes it a good idea to present an architecture that allows other people to know the essential elements to develop this type of software. The reason why showing an architecture is important is because it shows the important design decisions that are inherent in a system [16] [17], this makes it easier for society to advance in the development of these types of applications. A concrete example of this type of issue is the animation of script-based fragments, which are more appropriate for 2D games [18]. It is also important to note that designs can sometimes be improved based on experimentation with existing designs [3], for this reason, the article shows the architecture for this to be taken as a basis when developing new games and it is humanity that validates it.

On the other hand, the construction of 2D games requires the use of various tools to achieve a moderately acceptable game in terms of both performance and graphical interface [19], by way of illustration, the article presents the tools used in the construction of a labyrinth.

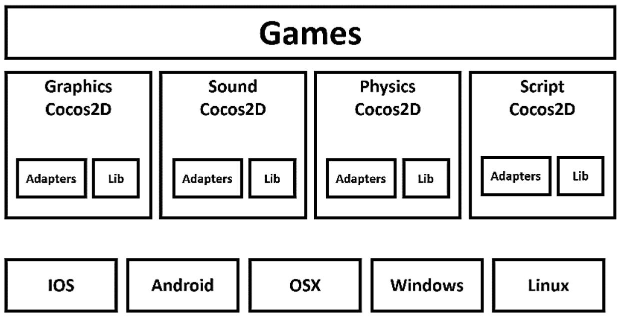
For the construction of the game presented in this article, the Cocos2DX tool was used, which is an environment widely used by 2D game developers. It should be noted that this is not the only existing environment for the development of 2D games, there are others such as Corona SDK [20], libGDX [21], Agk app game Kit, 2D Toolkit, Gideros Mobile, Shiva, Moai, BatteryTech, V-Play.

The paper begins a short explanation about the architecture implemented by the Cocos2DX APIs. Then an architecture is proposed that serves as the basis for the construction of 2D games on the Cocos2DX platform, This architecture was coded in the Objective-C language included in Cocos2DX, which is a language that has elements of C, C ++ and Objective-C. In addition, from Cocos2DX they were used as: Sound, Sprites [22], particles and fonts built with the sfxr tools for Windows, Texture Packer, Particle Designer and Glyph Designer, respectively. Then the execution of the application is shown where you can see that it works both on Windows, iPhone, and smartphone. Finally, some conclusions are presented on the work carried out.

## 2. The architecture of Cocos2DX

In the architectural figure of Cocos2DX (see figure 1), it can be seen that the architecture of the framework for the development of Cocos2DX games is quite robust in the sense that it allows the execution of games on the operating systems iOS, Android, OSX (from the company Apple Inc), Windows and Ubuntu.

**Figure 1.** Based on [23] Cocos2DX Architecture



Source: own.

To achieve the desired portability in the execution of games, a series of intermediate layers are implemented that are responsible for communicating between the game code and the device on which the software is running. It also has a series of interesting utilities such as Box2D, which is a 2D rigid body simulation library, written in portable C language. It also brings another simulation tool called Chipmunk in which it allows game objects to behave like real life objects, this means that it incorporates properties such as elasticity, rebound shock and other properties that allow real behavior.

In the programming part, Cocos2DX includes support for linking with programs written in C ++ language and this is achieved with Cocos2d-x lua [22]. The power of Cocos2DX even allows you to execute JS code and this is achieved because it includes in its architecture a scripting component that refers to the SpiderMonkey virtual machine [2], which is the Firefox virtual machine.

Incorporating and working with sounds can be done using additional tools like sfxr, however, regardless of whether these sounds are developed in external tools, some Cocos2DX own effects can be added to enrich the listening experience of users in games [17]. Among these APIs contained in cocos2DX, is the CocosDenshion API which is responsible for reproducing the sounds incorporated with other tools and has a series of optional effects that can be included or not depending on the needs of the programmer.

Finally, the Cocos2DX platform incorporates elements that allow its execution in various operating systems and for this, makes use of adapters and a library called "3rd libs", that communicates with the specific operating system depending on the portability needs.

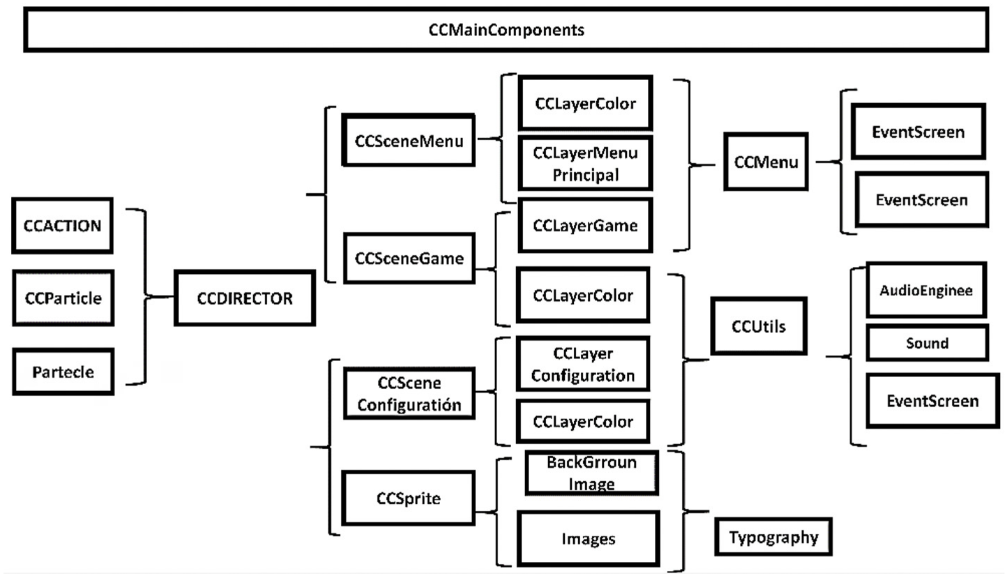
## 3. Methodology

Most 2D game engines are currently developing functionalities to develop 3D games [22] [24], And in the case of mobile gaming apps, interactivity is something to look for in any app for mobile devices [25], and obviously, games cannot escape this concept [26].

It is important to note that architectures and designs must be convergent and general to make these architectures work for various specific programming languages [27] [28]. Although Cocos2D provides a series of functionalities to model the 2D game, this article implements an architecture to other development environments such as Unity3D, Adobe Flash this is reflected in the prototype where these other two platforms are integrated, this to prove that the architecture was sufficiently applicable to these environments. This was done only as a test since the main objective is to continue with the development of 2D games in the Cocos2DX development environment.

The architecture of the game "Labyrinth" (see figure 2) is based on the concepts of "container", "node", "scene", "layer", "sprite" [16], "director", "particle" and " event. A more complex game might require the use of additional elements, however a traditional game such as a mobile labyrinth only required the concepts mentioned above. Figure 2 shows how most classes in Cocos2DX are based on the CCNode class, which is the way to designate almost any class in Cocos2D. Scenes in Cocos2DX are the most appropriate way to make functional divisions in a game-like application, because they allow separating the moments of action with respect to the screenshots shown on the various mobile devices. It is possible for the entire application to be rendered as a scene, but this

**Figure 2.** Architecture implemented in the labyrinth using Cocos2DX



Source: own.

is architecturally undesirable. The game "2D labyrinth", presents an initial menu for the user to start the game, a recommended practice [17], in this first scene the user is shown the name of the application. The second programmed scene is properly the game scene where the labyrinth is shown and finally the third scene is where the possibility of seeing the authors and making some configuration to the game is shown.

The architecture of the labyrinth (see figure 2) was built from the implementation of a methodological process of construction of software adapted to mobiles [19] that allowed us at first to make an analysis of the possible tools that can be used and analyzed the pros and cons of implementing them in the 2D labyrinth.

## 4. Results

It starts from the theory that it is necessary to delve into the tools that are going to be used to exploit them to the fullest and in this way make applications that take into account the real needs of the user [17], that have an impact at the level of usability on the user [5] and that allow to improve the way in which the first version of the prototype is developed from the experiences of the initial users of a system [14]. In this sense, a study was made of what tools could be used to improve the user experience in the development of 2D games and it was concluded that it was necessary to work with Cocos2DX, because it is a powerful tool with high performance for the development of this type of games. Some good tools were analyzed for this purpose like Unity3D [13], but in the end, it was concluded that tools natively based on Objective-C take better advantage of the characteristics of the processors provided in mobile devices [3].

The task of understanding the way of programming in Objective-C was undertaken, since it is a paradigm that, although it includes some aspects of the C language, has great differences that can confuse programmers of languages like C ++ [20]. There are countless types of applications that are based on this language and that have been widely accepted among people for their high performance [7]. In summary, it can be said that it is a language with a great future [10]. By its nature, the Objective-C language will be extremely popular in the future and based on this language, a variety of design tools will continue to be developed due to its high efficiency, especially in environments with few resources such as mobile devices [4].

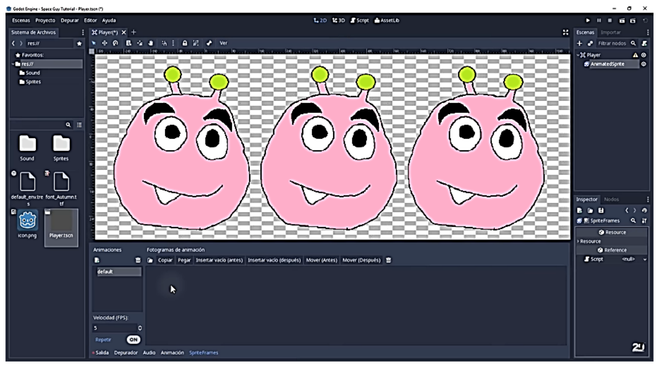
Because OpenGL is a widely used technology in the case of innovative mobile applications and with user interfaces it is surprising [10] [16] [17], It was being searched whether to use a tool that directly manipulates OpenGL language, and it was concluded that it does not matter if another tool generates it, the important thing is to have a tool that allows total interaction with this type of language [8],[18], because it is a language that continues to prevail due to its high performance [11], [12],[15]. This was another reason that strengthened the selection of the Cocos2DX platform.

If you want to get the most out of and improve the end-user experience of a game, you can take advantage of the advantage that Cocos2DX has of incorporating elements of the game made with other tools. This part of the paper shows the main environment used by Cocos2DX and the other tools that were used to add high-quality elements to the game that improve the user experience.

The cocos 2DX tool was used as the central axis of the development, where all the objects created with other tools are referenced. In this tool, all the classes of the labyrinth architecture were programmed with Objective-C (see figure 2).

The development environment allows the creation of canvases (see figure 3) which is a graphical distribution of the elements in the scenes where their positions are programmed. The development environment shown in figure 3 shows a screenshot of the development environment while programming the labyrinth.

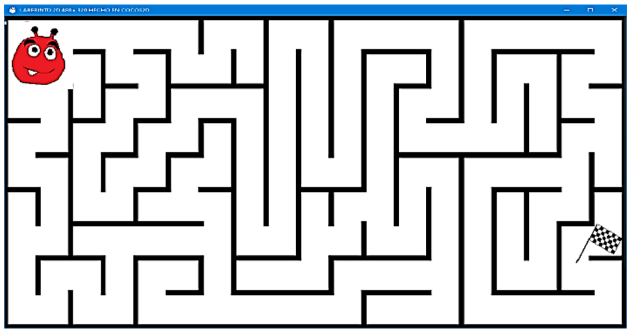
**Figure 3.** The Cocos2DX development environment



Source: own.

The game developed in Cocos2DX can be run even in non-mobile environments like Windows (see figure 4) thanks to all the interoperability support provided by the "3rd libs" libraries

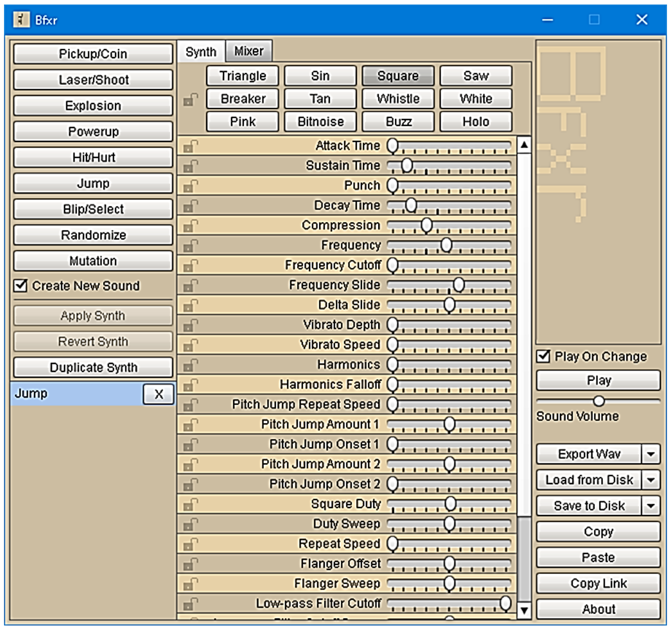
**Figure 4**. Running the game in a Windows environment



Source: own.

Sfxr is the program that was used to make sound effects to be incorporated into computer games, Figure 5. With this program you can do many things including: modify the waveform to apply effects to the sounds, harmonic and compression filters can be applied to remove glare from game sounds, it incorporates a powerful mixer and can be reversed to synthesizers.

**Figure 5**: The environment for creating Sfxr sounds

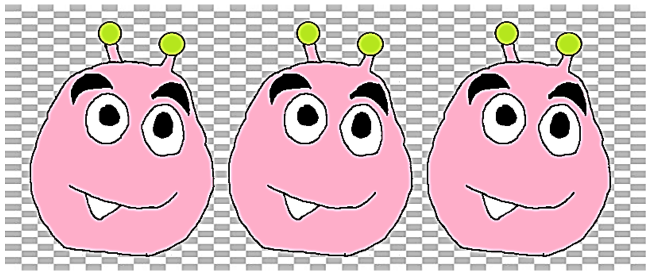


Source: own.

Sprites are important to improve the performance of a system [1]. A sprite sheet is an image file that contains multiple graphics in a mosaic-style grid. These graphics are loaded by applications such as cocos2DX and allow the game to dynamically display each of the images separately as an animation. The advantage of this type of technique refers to the memory loading of the images, because all the images are loaded at the beginning of the game and as required the images are loaded to give an animation appearance when it is being shown is a part of the image file.

The following graphic, Figure 6, shows a part of the sprite created for the development of the 2D game

**Figure 6**: Sprite created for the game



Source: own.

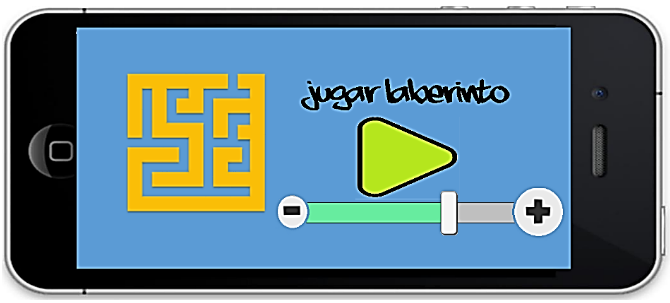
The tool used was "Texture Packer" and the fundamental reasons were: it reduces memory, not quality, when there are too many sprites you can handle this complexity with something called multipack, allows exporting to a large number of frameworks for performing games and automation tasks with command lines.

Another powerful reason for choosing the "Texture Packer" tool is that it is a framework that allows exporting to other frameworks such as: Corona SDK [1], libGDX [6], Agk app game Kit, 2D Toolkit, Gideros Mobile, Shiva, Moai, BatteryTech and V-Play. Particle system effects help a lot to give a pleasant experience to game users in general [10] and for this reason it was decided to incorporate these elements into the game. The tool used was "Particle Designer", and the fundamental reasons for its choice are given by the following functionalities it offers: creating particle effects from pre-set test effects and then incorporating them into the game, allows the generation of animation, alpha-channel, and even coded particle effects, all through a timeline. Loop-like effects can be created.

Fonts can be created by hand; it is possible that the time spent doing this type of thing has been seen and the need to use specialized software is seen. Some possibilities for the development of this type of typeface were being analyzed, among which were analyzed: FontForge, Robofont, FontLab, FontoGrapher and Glyphs. Initially, the game was developed using Glyphs, although, once the game was developed, it was possible to see that there are other possibilities that will most likely be considered for the development of game improvements. Among these future possibilities, FontForge will be considered, since it has an advantage over the rest of the possibilities analyzed and this basically lies in being a free tool of good quality and open source.

With the use of the previous tools, the application shown in figure 7 has been developed, where the interoperability of the applications with Cocos2DX is high, because the graphical interface does not change its appearance in any of the devices for which it is designed

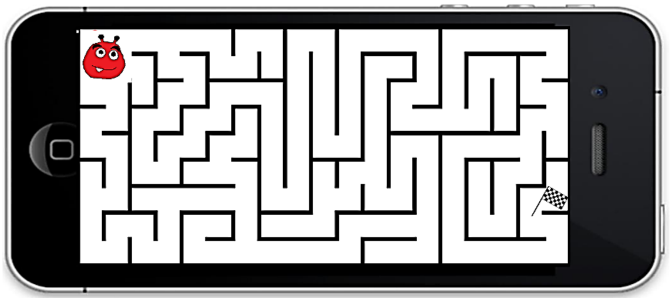
**Figure 7**. Initial screenshot of the 2D Labyrinth application on an iPhone



Source: own.

The game consists of showing a doll that the user can move either by using the finger or by moving the cell phone making internal use of the gyroscope provided by the mobile devices (see figure 8).

**Figure 8**. 2D Labyrinth game in action



Source: own.

## 5. Conclusions

It can be concluded that the decision on the use of additional tools to the central tool with which a 2D game is made is final for the success of a game project on mobile devices, as even though the various tool manufacturers auxiliaries promise full interoperability at the time of practice, they are not without drawbacks.

When the realization of a game is required for a specific technology such as Android, it would first be thought that the most recommended thing is to do this development in an environment specialized in Android, however this decision that would seem natural is not the best for several reasons, The first reason is that although it seems strange the development environments for the realization of games on Android are not adapted to the specific needs of 2D games and the development environments Unity3D, Cocos2DX, Marmalade or even Adoble Flash are environments that allow compiling for different platforms they turn out to be better choices since, although they present wear at the time of making the translation to the specific platform, It is also true that they have so well developed the specific functions of physics and 2D games that they generate a more efficient code than if these specific functionalities are included from scratch. The way in which the logic of a 2D game is implemented from scratch for a specific platform would have to be developed and studied quite a lot. On the other hand, the other reason why it is not a good choice to develop for a specific platform is that most companies that implement 2D game logic do so with multiple platforms in mind and this is how platforms for 2D games are based on a specific platform are almost nil and the trend is not aimed at this development. Finally, an important reason that supports the idea of developing for multiple platforms, is that the probability that a development is required to start running on different platforms increases in the sense that it is not clear at this time if a platform is going to be imposed. over the other and multiple current platforms are expected to continue to endure into the future.

Once the decision to develop on a multi-platform framework is made, the question that comes is: Was the best decision made when Cocos2D was chosen as the cross-platform framework for 2D game development? The conclusion is that the choice that was made to take Cocos2D as a multiplatform framework was good depending on the improvement needs that are required to be made on the developed labyrinth. For example, if the labyrinth game that was developed does not require major improvements, it can be assured that the decision to develop the game in Cocos2D was not necessarily the best option in the sense that the same game and with the same quality could most likely be achieved it has developed faster with Adobe Flash technology. However, if the needs of the game grow, it is possible that the performance of the maze will be seriously affected and in this sense the effort to learn Objectivo-C is rewarded with improvements in performance when the game takes on special complexity and becomes more computationally expensive. Another way of reasoning would show that the question asked may or may not be related to the need to do open source. If there is no need to run open source, the decision to choose Cocos2D, Flash, Unity3D or Marmalade, is indifferent in this sense, but if the decision is made thinking that it is a necessity to develop for open source, without a doubt, the best choice in this regard is Cocos2D in the sense that it is the most complete open-source platform at the moment and totally free. So much so that after developing the labyrinth it has been possible to see the possibility of making multiple improvements to the labyrinth that include improvements in the handling of physics based on free APIs that are found as one delves into the Cocos2D development environment. The possibilities for improving the labyrinth game are many and there is the possibility of continuing to improve it while maintaining the condition of development based on open source.

In the medium and long-term future, it cannot be concluded that Cocos2D continues to be the platform par excellence, in the sense that competitions such as Flash, Marmalade and Unity3D are expanding their functionalities and are still in the future battle.

However, of all these rivals, what can be concluded is that the most serious rival of all that Cocos2D has is Unity3D and in this sense it is not clear whether one or the other will prevail in the future. These two options are likely to remain. At this time, it can be concluded that the comparisons between Cocos2DX and Unity3D have not been objective, in the sense that almost always the comparisons found in the texts are comparisons of experts in one or another technology and it is not easy to find experts. in the two technologies that make more objective comparisons about the two technologies. It is concluded then that more experts in these two technologies are required in the world to make comparisons closer to reality on these 2D game development platforms.

## References

[1] J. C. Najar-Pacheco, J. A. Bohada-Jaime, W. Y. Rojas-Moreno, “Vulnerabilities in the internet of things”, Visión electrónica, vol. 13, no. 2, pp. 312–321, 2019. https://doi.org/10.14483/22484728.15163

[2] L. Iyuan, H. Wenfeng, “Development of Puzzle Game for IOS Platform Based on Unity3D”, in 3rd International Conference on Applied Computing and Information Technology/2nd International Conference on Computational Science and Intelligence (ACIT-CSI), 2015. https://doi.org/10.1109/ACIT-CSI.2015.89

[3] A. Lima and E. A. da Costa, “Experimental Approach of the Asymptotic Computational Complexity of Shaders for Mobile Devices with OpenGL ES”, in Brazilian Symposium on Computer Games and Digital Entertainment, 2014. https://doi.org/10.1109/SBGAMES.2014.9

[4] B. J. Cox, “The objective-C environment: past, present, and future”, 1987. https://doi.org/10.1109/CMPCON.1988.4852

[5] G. Bournoutian and A. Orailoglu, “On-device objective-C application optimization framework for high-performance mobile processors,” in Design, Automation & Test in Europe Conference & Exhibition (DATE), 2014. https://doi.org/10.7873/DATE2014.098

[6] R. Rawlings, “bjective-C: an object-oriented language for pragmatists” in Colloquium on Applications of Object-Oriented Programming, 1989.

[7] G. Song, S. Ren, D. Zhang, K. Liu, Y. Sun, X. A. Wang, “Research on War Strategy Games on Mobile Phone based on Cocos2d-JS”, in 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2015. https://doi.org/10.1109/3PGCIC.2015.128

[8] S. Guozhi, R. Shuxia, Z. Dakun, L. Kunliang, S. Yumeng, A. W. Xu, “Research on War Strategy Games on Mobile Phone”, 10th International Conference on P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), pp. 151-155, 2015.

[9] B. A. Brady, A. K. Jones, I. S. Kourtev, “Efficient CAD development for emerging technologies using Objective-C and Cocoa”, in International Conference on Electronics, Circuits and Systems, 2004, 2004.

[10] C. W. Cho, C. P. Hong, J. C. Piao, Y. K. Lim, S. D. Kim, “Performance optimization of 3D applications by OpenGL ES library hooking in mobile devices”,

in 13th International Conference Computer and Information Science (ICIS), 2014 IEEE/ACIS , 2014. https://doi.org/10.1109/ICIS.2014.6912179

[11] J. C. Piao, C. W. Cho, C. G. Kim, B. Burgstaller, S. D. Kim, “An Adaptive LOD Setting Methodology with OpenGL ES Library on Mobile Devices”, in International Conference on Convergence and Security (ICITCS), 2014. https://doi.org/10.1109/ICITCS.2014.7021727

[12] M. Ángel Barrera Pérez, N. Y. Serrato Losada, E. Rojas Sánchez, G. Mancilla Gaona, “State of the art in software defined networking (SDN)”, Visión electrónica, vol. 13, no. 1, pp. 178–194, 2019. https://doi.org/10.14483/22484728.14424

[13]R. Besas, R. O. Atienza, T. Tai, R. Cruz, “An implementation of a structured and highly engaging learning environment on educational games for elementary education”, in IT in Medicine and Education (ITME), 2011. https://doi.org/10.1109/ITiME.2011.6132115

[14] C. Carter, Q. Mehdi, T. Hartley, “Navigational techniques to improve usability and user experience in RPG games”, in 17th International Conference on Computer Games (CGAMES), 2012. https://doi.org/10.1109/CGames.2012.6314557

[15] C. Le Marc, J. P. Mathieu, M. Pallot, S. Richir, “Serious gaming: From learning experience towards User Experience”, in International Technology Management Conference (ICE), 2010. https://doi.org/10.1109/ICE.2010.7477028

[16] S. F. Hsiao, S. Y. Li, K. H. Tsao, “Low-power and high-performance design of OpenGL ES 2.0 graphics processing unit for mobile applications”, in International Conference on Digital Signal Processing (DSP) , 2015. https://doi.org/10.1109/ICDSP.2015.7251840

[17] S. F. Hsiao, P. H. Wu, C. S. Wen, L. Y. Chen, “Design of a programmable vertex processor in OpenGL ES 2.0 mobile graphics processing units”, in International Symposium on VLSI Design, Automation, and Test (VLSI-DAT), 2013.

[18] X. Zhao, X. Huang, “A general solution of script-based fragment animation”, in 6th IEEE International ConferenceSoftware Engineering and Service Science (ICSESS), 2015. https://doi.org/10.1109/ICSESS.2015.7339216

[19] L. Wang, “Design and Implementation of Four Arithmetic Operations Learning Games in Primary Mathematics Based on cocos2d-js”, 3rd International Conference on Mechanical, Control and Computer Engineering (ICMCCE), pp. 595-598, 2018. https://doi.org/10.1109/ICMCCE.2018.00130

[20]M. P. A. Balayan, V. V. B. Conoza, J. M. M. Tolentino, R. C. Solamo, R. P. Feria, “On evaluating skillville: An educational mobile game on visual perception skills. In Information, Intelligence, Systems and Applications”, in The 5th International Conference IISA 2014,, 2014. https://doi.org/10.1109/IISA.2014.6878828

[21]B. Cassidy, G. Stringer, M. H. Yap, “Mobile Framework for Cognitive Assessment: Trail Making Test and Reaction Time Test”, in Computer and Information Technology (CIT), 2014. https://doi.org/10.1109/CIT.2014.164

[22]Y. Lu, W. Gao, F. Wu, “Efficient background video coding with static sprite generation and arbitrary-shape spatial prediction techniques”, Transactions on Circuits and Systems for Video Technology, vol. 13, no. 5, pp. 394-405, 2013. https://doi.org/10.1109/TCSVT.2003.811607

[23]Cocos2D-x, “ARCHITECTURE OVERVIEW,” [Online]. Available: http://www.cocos2d-x.org/wiki/Engine_Architecture

[24]Y. Lu, Y. Liu, S. Dey, “loud mobile 3D display gaming user experience modeling and optimization by asymmetric graphics rendering”, IEEE Journal of Selected Topics in Signal Processing, vol. 9, no. 3, pp. 517-532, 2015. https://doi.org/10.1109/JSTSP.2015.2396475

[25]S. Arefin Riffat, F. Harun, T. Hassan, “An Interactive Tele-Medicine System via Android Application”, Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA), pp. 148-152, 2020. https://doi.org/10.1109/ACCTHPA49271.2020.9213200

[26] Y. Liu, H. Dar, R. Sharp, “Mobile Gamer Modelling and Game Performance Preference Measurement”,

IEEE Conference on Games (CoG), pp. 632-635, 2020. https://doi.org/10.1109/CoG47356.2020.9231860

[27] J. C. Piao, C. W. Cho, C. G. Kim, B. Burgstaller, S. D. Kim, “An adaptive LOD setting methodology with OpenGL ES library on mobile devices”, in IT Convergence and Security (ICITCS), 2014. https://doi.org/10.1109/ICITCS.2014.7021727

[28] E. C. Chan, “Appendix B: Introduction to Objective-C Programming in iPhone”, in Introduction to Wireless Localization: With iPhone SDK Examples, pp. 261-304.