Gamification Experience with Scratch in Teaching Programming in a Vocational Training Classroom

Antonio García-Gutierrez
Computer Science Department
Universidad Rey Juan Carlos
Móstoles, Madrid, España
a.garciagu.2018@alumnos.urjc.es

Raquel Hijón-Neira
Computer Science Department
Universidad Rey Juan Carlos
Móstoles, Madrid, España
https://orcid.org/0000-0003-3833-4228

***Abstract*— This article analyzes the impact that the use of an application with gamified environments can have on the learning of theoretical programming concepts and on the development of students' computational thinking. The application offers theoretical and practical learning environments based on the Scratch programming software. For this analysis, a pilot experience has been carried out in which the application is tested with students. The aim is to analyze the progress of both the student's learning of theoretical concepts and the development of computational thinking. For this, two tests have been carried out, analyzing the results, collecting data by performing a pre-test and post-test.**

Keywords*— *Programming, Computational Thinking, Scratch, Gamification

I. Introduction

Currently, video games are one of the most direct ways that children and young people have to interact with computers. Their ability to encourage participation can be of great help when it comes to involving the student and fostering a competition that encourages learning by awarding badges. By promoting a context of participation, the student breaks with the feeling of frustration that the appearance of obstacles in their training can cause and is useful to motivate students to carry out tasks that are complicated.

An experience is presented in the use of an application for the teaching of programming and computational thinking of gamification in professional training. A pilot experience was carried out that consisted of carrying out a pre-test of theoretical knowledge and computational thinking. Once the tests prior to the interventions had been carried out, theoretical concepts were explained from the Scratch-based application, making examples of the different concepts.

Once the interventions had been completed and all the concepts were explained, a post-test was carried out to evaluate the students' progress, comparing the results obtained both in the pre-test and in the post-test and determining if there had been significant progress.

II. State of the Art

A. Status of Gamification

Nowadays, society is faced with numerous technological and social changes. Technology changes the way society interacts and facilitates access to new knowledge. Gamification responds to this need for motivation, transporting game scenarios to formal educational contexts, in order to involve students and open the door to the acquisition of learning. To do this, it is necessary to surround students in playful environments, facing challenges and missions, increasing their commitment and increasing their participation.

It is reasonable to think that an activity that is surrounded by a similar technological environment and that uses the same components and dynamics as these games can motivate and change the student's perception of academic activities, resulting in more attractive and motivating performance. Three lines of work stand out that seek to gamify education..

The first seeks to use games in a controlled way, the teacher chooses the game and the moment, so that the student acquires the skills and abilities that are supposed to appear in them. The second way seeks to use the characteristic elements of the games (the levels, the points, the medals, the interface ...) to take advantage of the students' predisposition to play to increase the motivation for learning. The third way is to redesign a learning process as if it were a game. The teacher must design the subject in such a way that the student has to play and acquire knowledge, skills and competences [1].

It is necessary to define what conditions a process must meet to be considered a game or object to be gamified. The requirements or conditions are reduced to the fact that the activity carried out by the game or process can be learned by the user, that the feedback can be delivered in a pertinent way to the user and that the actions carried out by the player throughout the development of the game. process or activity can be measured [2].

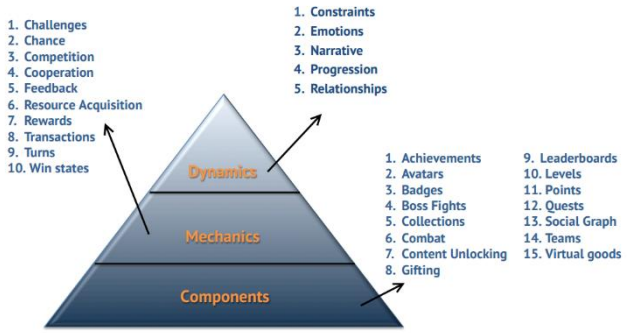

Figure 1. Hierarchy of gamification elements [3]

The components suppose the badges, points, rankings and other elements that manage to implement the mechanical and dynamic elements. The mechanical elements are made up of all those elements that make the game user act in the game. It can be the obtaining of a reward, a challenge or the same competition between users within the same class or center. Finally, the dynamic part is made up of the concept of the game.

To achieve the objective, the game must integrate all these elements, in addition to achieving the student's motivation, which may be intrinsic, when the student seeks to carry out activities for their own interest, or extrinsic, when the student performs the tasks for a reward you can get.

The teaching of programming is closely tied to computational thinking. Solving the problems using a series of instructions structured and sequenced in such a way that they allow to reach the resolution of the problem.

Therefore, it is necessary that, when learning programming, students structure the resolution of a problem analytically; analyzing the problem and establishing and ordering a series of steps to follow to solve the problem.

A simple tool when working on computational thinking is Scratch, since it allows the student to interact with a simple programming language, in which the programming language is sequenced by blocks, eliminating much of the complexity of the programming languages. programming and allowing focus on the logic and structure of problem solving, thus working computational thinking.

It is worth highlighting the role that the Dr. Scratch application can play in this part of learning. This application allows you to correct, analyze and catalog the degree of learning of a project by assigning a score distributed among the different computational concepts.

It is important to highlight the analysis of the Dr. Scratch application that was carried out in a program made up of 109 elementary and high school students in which it was intended to analyze the progress of computational thinking using the Scratch programming tool. To do this, some work sessions with Scratch were carried out, and the Dr. Scratch tool was also presented to the students so that students could practice, develop and correct their own codes through the tool.

To measure the progress of these students, a pre-test and post-test were carried out before and after practicing with the tool. The results were analyzed by means of a t-test, establishing a level of significance alpha = 0.05, returning a result of $p < 0.05$, so it could be stated that the learning was significant.

Another significant experience was the development of a study carried out in sixth grade students whose objective was to verify whether the Scratch programming tool allows the development of computational thinking. The methodology of work with these subjects consists of 4 phases.

A first phase in which a Scratch guide is developed as a learning resource. The guide consists of 5 units in which different activities are developed that increase the difficulty. The guide is aimed at subjects who do not have any knowledge of programming and use of Scratch.

A second phase where the guide is developed with the students and proposed exercises are worked on. This second phase occupies 14 sessions of 45 minutes where the subjects work with the teacher the concepts and activities proposed in the guide. In the third phase, the evaluation of computational thinking is carried out by means of a test.

In the last phase, the analysis of the data obtained is carried out. Due to the positive results obtained, it can be concluded that the Scratch programming software is a good tool to develop computational thinking at an early age [5].

Another interesting example for the introduction of theoretical concepts and computational thinking at an early age is RoDy. It consists of two parts: on the one hand, an application that allows teachers to generate their own activities according to the syllabus and, on the other hand, a robot that allows students to interact with the application. This robot is represented in the game by a virtual agent in the shape of a bear, which will interact with users through dialogues [7].

As a future project for the integration of gamification in the classroom, it is worth highlighting the proposal of the GameMo module for Moodle, which will allow integrating the most common elements of gamified courses within the Moodle platform, thus facilitating the teaching work for the implementation of gamified courses. Once an initial prototype has been developed, it will be tested in real environments by means of a quantitative usability assessment, leaving it open to future improvements after tests in real environments [6].

B. Hypothesis

Due to the need to study and analyze the impact that the application can achieve in the learning and development of the theoretical concepts of programming and computational thinking. Therefore, the hypotheses to be determined are the following:

- **H1:** With the methodology / intervention proposal, learning in programming concepts can be improved.
- **H2:** With the methodology / intervention proposal, one can improve the learning of Computational Thinking.

III. METHODOLOGY

The methodology to be followed in the interventions consists of developing the theoretical concepts outlined in the application guide. The development of the concepts consists of a first theoretical part in which the teacher explains and defines the concept that will be worked on in that topic, and later a practical part in which the student interacts with different programs developed in the software-based application Scratch programming.

For the development of the interventions a local web application is used that uses the Scratch block programming software.

At the beginning of the application there is a main page where the different topics to be developed by the students are shown. The application consists of two parts, presentations (Figure 2) and practical (Figure 3).

Figure 2. Screen of the interactive presentation of the theoretical concepts on Computational Thinking

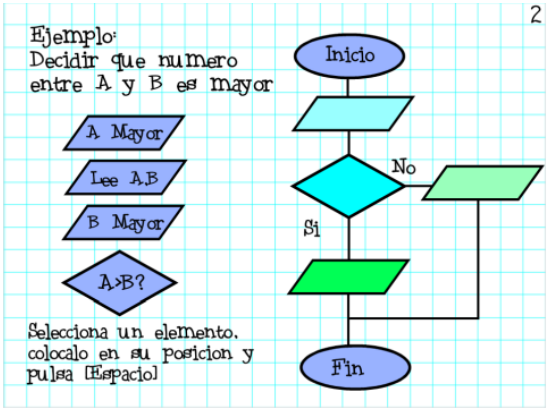

Figure 3 Scratch Practice Screen.

The presentation part is used to support the teacher in the presentation and development of theoretical questions. It offers an environment in the form of a blackboard in which theoretical questions are exposed, defining the concepts explained clearly and schematically and giving simple examples from day to day, thus facilitating the understanding of the theoretical concept.

The practical part offers an interactive environment in which the student can put into practice the concepts previously explained by the teacher. For this, the application has different exercises developed with the Scratch programming software.

The topics developed in the application are the following:

- Topic 1: Sequences.
- Topic 2: Variables and data.
- Topic 3: Operators.
- Topic 4: Conditionals.
- Topic 5: Loops.
- Topic 6: Events.
- Topic 7: Parallelism and synchronization.
- Topic 8: Computational thinking.

IV. Experimentation

A. *Participants and sequencing*

The participants are 1st grade administrative students, a group made up of 9 students aged between 16 and 17 years. The students belong to an institute in the province of Toledo. This group is chosen for accessibility and availability of the sample. Figure 4 shows the level of programming knowledge of the participating students. Most of them state that they do not have any prior knowledge about programming and, furthermore, they see theoretical concepts as abstract and difficult to understand.

In the first intervention, the activity was explained to the students. Before starting to work with the application, the theoretical concepts and computational thinking pre-tests were carried out. The session took place with the regular teacher, where the activity was explained to the students. The duration of the session was 50 minutes and the concepts sequences, variables, operators and conditionals were explained with examples in Scratch.

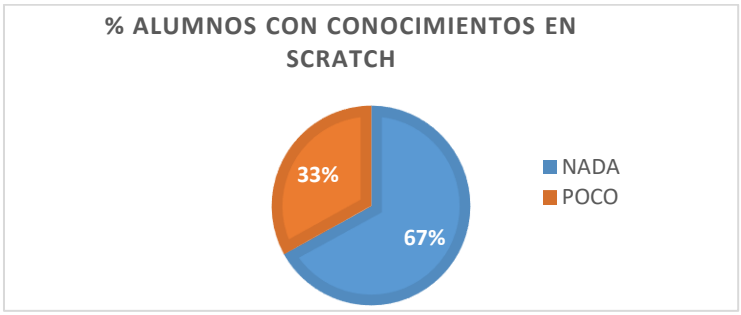

Figure 4 Percentage of students with prior computer knowledge

In the second intervention, the activity was resumed from the point where the previous session left off. The session lasted 50 minutes and the rest of the concepts (loops, events, parallelism and computational thinking) were explained. In addition, the two post-tests were carried out, thus ending the interventions.

B. *Data collection*

Two tests were used to collect samples. The first of them has been used to analyze the progress of knowledge of theoretical concepts, and the second has been used to measure progress in computational thinking experienced by the student thanks to the application. The research carried out is carried out within the framework of quantitative research, describing the observed reality and establishing a cause-effect relationship, explaining the results obtained. A quasi-experimental method will be followed, since due to the number of students it has not been possible to separate the sample into a test group and a control group. The two tests (pre-tests) will be applied before starting the sessions, and the two tests at the end (post-tests).

To carry out both questionnaires, the Google Docs forms tool has been used due to the ease of use and accessibility of the application.

The theoretical knowledge test consists of 16 questions, both multiple choice questions and open questions. The objective of the test is to evaluate how the application has influenced the learning of theoretical concepts. An example of the questions that make up the theoretical knowledge test is shown in Figure 5.

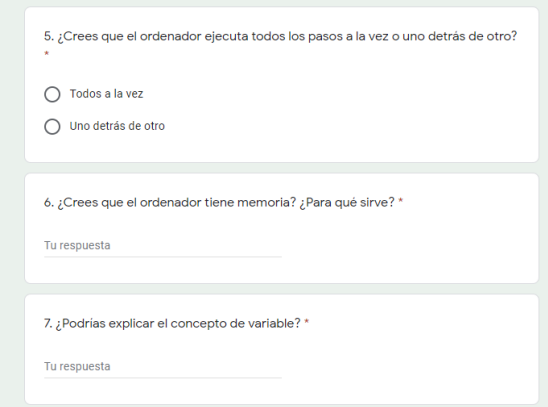

Figure 5 Theoretical concepts test

The computational thinking test consists of 28 questions, with closed questions in which the student must choose between 4 options, only one of them being the correct answer

(Figure 6). The computational thinking test is based on the work carried out by Román González, establishing the criteria for the elaboration of the test [4].

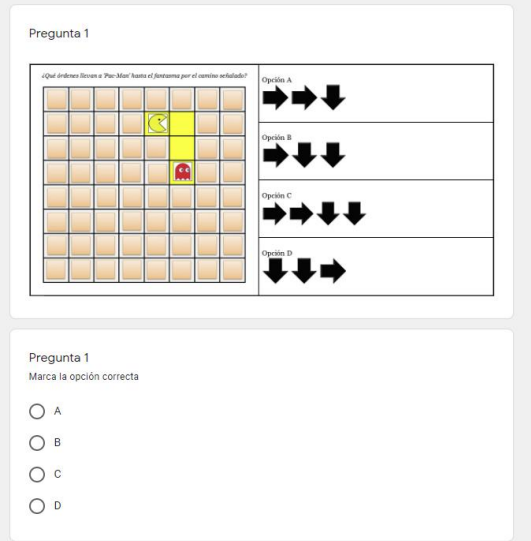

Figure 6 Computational thinking test

C. Analysis of the results

To evaluate the degree of significance of the learning of the theoretical concepts, the t_test for dependent samples has been performed, obtaining the following results (see Table 1 and Figure 7):

Table 1 Results T-test theoretical concepts

Var.	Obs.	Min.	Max.	Media	Typ. Devia.
PRE-TEST GRADE	9	1,538	4,038	2,393	0,764
POST-TEST GRADE	9	2,692	6,154	4,509	1,397

The results observed in Table 1 and in Figure 7, on which the t_test has been carried out, show a value p = 0.002, well below the alpha value = 0.05, so it can be stated that learning the concepts theoretical has been significant.

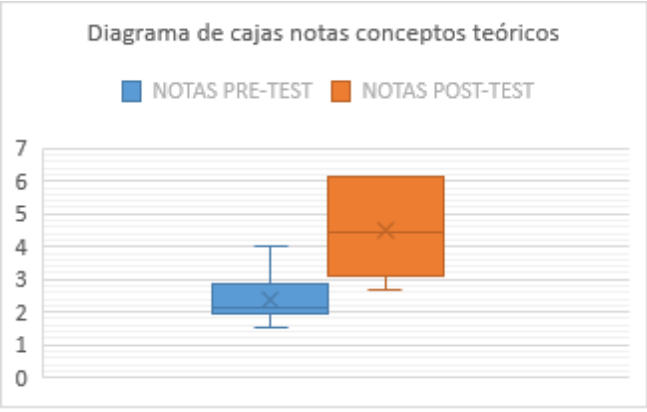

Figure 7 Boxplot of the t_test for samples of theoretical concepts

To verify that learning has been significant in the computational thinking test, the T_test is performed again, obtaining in this case a p-value = 0.316 that is greater than the alpha value = 0.05 (see Table 2 and Figure 8).

Table 2 Results T-test theoretical concepts

Var.	Obs.	Min.	Max.	Media	Typ. Devia.
PRE-TEST GRADE	9	1,071	7,143	5,198	1,899
POST-TEST GRADE	9	2,500	8,214	5,675	2,146

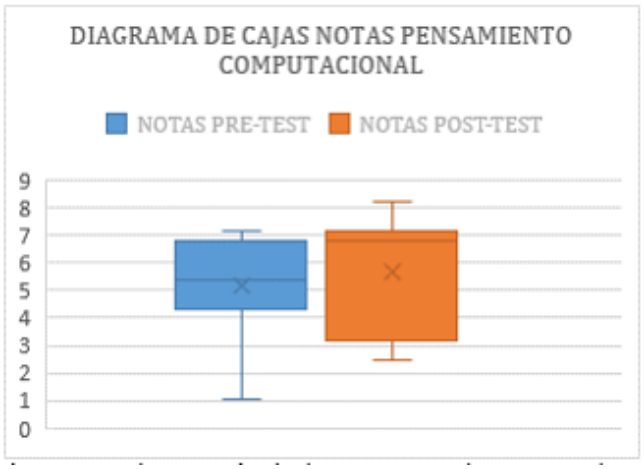

Figure 8 Boxplot of the t_test for the Computational Thinking samples.

Although in most cases a slight evolution is observed (see Figure 6), this is not significant as has been observed in the learning of theoretical concepts. In addition, we find the average obtained both in the pre-test and post-test (marked with quite a few crosses) quite close to each other. This lower evolution is attributed to the fact that computational thinking requires a systematic change in the way of thinking, analyzing and approaching problems much more profound than that obtained during the development of the sessions.

Analyzing separately the results obtained in the different theoretical concepts, it is observed that the concepts that have presented the most difficulty among the students have been the concepts of events, conditionals and variables. Instead, the concepts where the greatest progress can be seen have been the concepts of parallelism, events, and operators.

V. Conclusions

When comparing the results obtained with the results obtained in the experience described in section II. State of the art, in which the Dr. Scratch tool was used for its analysis, it is observed that the development of computational thinking is not significant, unlike the results analyzed with the use of the Dr. Scratch application that are described in that study.

Although in the present study, the learning of theoretical concepts has been significant, the application for the teaching of programming and computational thinking used has not had enough impact to achieve the development of computational

thinking with a sufficiently significant degree. The main difference with respect to the previous study is that the members of the study of this tool worked for weeks using the Scratch application, achieving a knowledge base that would help them progress more quickly when they started working with the Dr. Scratch.

At the beginning, the students had a certain feeling of rejection of the intervention approach, when they conceived the programming as something difficult and distant. Thanks to the intervention, most of the students abandoned this feeling that causes ignorance of concepts, they learned that programming is based on thoughts and logic present in everyday life. At the end of the sessions, although it is not directly related to the professional field of the higher grade, the activity was very well received by the students.

In view of the results, the duration of the interventions has been sufficient to achieve a significant learning of theoretical knowledge, but not to achieve enough impact to develop computational thinking, although it does improve it. How it improves future interventions, a higher number of sessions is proposed, or failing that, an increase in their duration that allows students, apart from the theoretical explanation and interaction with the application programs, to develop your own codes with teacher guidance.

The results obtained reflect a better understanding of the theoretical concepts after the sessions. In the case of the theoretical concepts test, once the sessions are over, an evolution is observed in all the students, being more noticeable in some students, and to a lesser extent in the rest of the students. The results of the t-test were positive, observing a significant degree of learning. The time of the interventions and the use of the application was correct to develop, in students who had no prior programming knowledge, the theoretical concepts that make up computational thinking.

It is concluded, therefore, that the use of the application has notably improved the disposition of students to learn and understand theoretical concepts that they conceived as difficult. After the sessions, the students have significantly improved their knowledge of these concepts, thus confirming the first hypothesis raised. This has not been the case in computational thinking, in which more sessions may be needed to provoke a more profound change in the methodology and approach that these students have to address problems (second hypothesis). The results reflected in the t-test indicate that, although there is an improvement and a development in the learning of computational thinking, it does not reach a significant degree, rejecting the second hypothesis raised.

One possible reason may be that the study subjects did not have previous programming knowledge, therefore, it is possible that the duration of the sessions and the application work, although they have had positive results, have been insufficient to achieve a significant development of the program. computational thinking. How aspects of improvement can be considered increasing the number and time of the sessions, being able to delve more deeply into the contents of the application.

The work with the students has been fun and fluid and, in general, they have perceived the sessions as something positive, a way of learning in a fun way, creating a relaxed atmosphere in the class dynamics in which the students were encouraged to participate. actively. The concepts that the students previously perceived as abstract and difficult to understand have been worked on in an easy way interacting with the different examples designed in Scratch, which allow the assimilation of theoretical concepts.

ACKNOWLEDGMENT

This work has been funded by the research projects iPROG: New generation of tools for learning Programming with emerging interactive technologies from MINECO (ref. TIN2015-66731-C2-1-R) and e-Madrid: Research and Development of Educational Technologies in the Community of Madrid (ref. P2018 / TCS-4307).

REFERENCES

[1] Fidalgo, A. (2014, 26 de marzo). Qué es gamificación educativa. Innovación Educativa. Blog de Ángel Fidalgo para reflexionar sobre innovación educativa. Recuperado de http://innovacioneducativa.wordpress.com/2014/03/26/que-es-gamificacion-educativa

[2] Cook, W. (2013). Training Today: 5 Gamification Pitfalls. Training Magazine. Recuperado de: http://www.trainingmag.com/content/training-today-5-gamification-pitfalls.

[3] Hunter, D., & Werbach, K. (2012). Gamificación. Revoluciona tu negocio con las técnicas de los juegos.

[4] Román-González, M., Pérez-González, J.C., Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. Computers in Human Behavior, 72, 678-691. doi: https://doi.org/10.1016/j.chb.2016.08.047.

[5] Álvarez Rodríguez, M. (2017). Desarrollo del pensamiento computacional en educación primaria: una experiencia educativa con Scratch.. Universitas Tarraconensis. Revista de Ciències de l'Educació, 1(2), 45-64. doi:https://doi.org/10.17345/ute.2017.2.1820.

[6] García-Iruela, M., & Hijón-Neira, R. (2018). Propuesta de interfaz de gestión de entornos gamificados en Moodle.

[7] Hidalgo Rueda, L., Pérez-Marín, D. "RoDy: peluche robótico para enseñar a compartir en Educación Infantil. 21st International Symposium on Computers in Education (SIIE 2019).pp:101-106.