

Using Coding Dojo with Mobile Game Development to Engage Students to Learn Programming

Angelo Magno de Jesus
Universidade Cruzeiro do Sul /
Instituto Federal de Minas Gerais
Rua Galvão Bueno, 868
São Paulo, SP
55-11-33853015
angelo.jesus@ifmg.edu.br

Gislaine Moura da Silva
Instituto Federal de Minas Gerais
Rua Afonso Sardinha, 90
Ouro Branco, MG
55-31-39381200
gislainemoura.s@gmail.com

Ismar Frango Silveira
Universidade Cruzeiro do Sul
Rua Galvão Bueno, 868
São Paulo, SP
55-11-33853015
ismar.silveira@cruzeirodosul.edu.br

ABSTRACT

This paper describes a teaching method adapted from Coding Dojo and based on mobile game development. This approach aims to motivate students to learn programming in a collaborative and playful way. The method was based on constructionism and Vygotsky's cognitive development theories. AppInventor, which was chosen as a implementing environment, allows students to develop mobile applications in a relatively quickly and easily way using a visual block programming environment. Experiments were conducted with Vocational High School students enrolled in an Information Technology course. The evaluation of the proposed method was performed by means of a survey, applying ethnography techniques. The results showed that students were able to collaborate and have fun under this approach.

Categories and Subject Descriptors

K.3.1 [Computers and Education]: Computer Uses in Education.

General Terms

Algorithms, Design, Experimentation, Theory.

Keywords

Coding Dojo, Game Development, Teaching Programming.

1. INTRODUCTION

"The kind of knowledge that children need most is that will help them to gain more knowledge" [1]. The constant and fast advances in technology and science challenge us to learn new things every day. Besides, schools generally focus on teaching content that is often meant to be learnt in a rote, non-meaningful way. It's necessary to think about new paradigms to educate according to the new learning realities inside classrooms. This educational challenge is also present on teaching computer

programming and algorithms. According to many authors, like [2], teaching programming is indeed a complex task. The lack of motivation has been appointed as one of the problems in learning and applying programming concepts. Also, the same authors claim that "several studies point to problems ranging from the difficulty of the students in understanding the programming concepts due the lack of motivation of them in performing the programming activity" [2].

The importance of learning programming goes beyond the technology-focused courses. This field of knowledge could stimulate the development of Computational Thinking skills in students. Computational Thinking, as defined by [3], is a basic skill for everyone, everywhere, which is characterized by promoting problem solving in different fields of knowledge, through mental tools that were derived from Computer Science. These mental tools include abstraction, data analysis, algorithmic thinking etc.

In this sense, the aim of this work is to describe a teaching method based on Coding Dojo and game development, applied to freshmen in different levels (regular and vocational high school and undergraduate students) in courses related to information technology. This approach was designed in order to teach and motivate students to learn programming in a playful way. This work is an extension of a previous work published in [4], when we described an overview of the method as a work in progress, without any theoretical relations, meaningful results or discussions. Now, we will present the entire method including its relations with the theoretical background, as well as complete results and discussion.

Coding Dojo could become a great contribution to the programming learning. This activity has been successful applied in many software companies [5, 6]. The basics of Coding Dojo activities involve collaboration in order to solve a problem, key concepts related to constructionism [1] and Vygotsky's learning theory [7].

Our method also uses game development to engage students to solve some proposed challenges. Researches have shown that game development could motivate students to increase their programming and algorithms-related skills [8, 9]. Game designing has some advantages in terms of regular software programming, due its high iterative aspect. It means that students could interact and have fun with the result of their work.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Conference '10, Month 1–2, 2010, City, State, Country.

Copyright 2010 ACM 1-58113-000-0/00/0010 ...\$15.00.

The proposed method also considers the software development for Android operating system running on mobile devices. It was done especially by the popularization (and access) of this kind of device among children and teenagers. In the end of the approach, students could install the app in their mobile devices and have fun with the game.

2. THEORETICAL BACKGROUND

This section will describe the theoretical basis of this research as we will explain how it is related to Coding Dojo and game development activities.

2.1. Codding Dojo

Coding Dojo was inspired in the Japanese martial arts' training places. According to [10], "the main principles of the Coding Dojo are to create a safe environment which is collaborative, inclusive, and non-competitive where people can be continuously learning". In agreement with [11] "a Coding Dojo is a meeting where a group of programmers come to work together in a programming challenge. They are there to have fun and, through a pragmatic methodology, to improve their programming techniques and group work skills". Activities are not focused only on creating a solution, but the discussions taken about different solutions are highly valuable. Meetings basically are carried in a room with at least one computer and one multimedia projector to show and test the algorithm designed to solve a proposed problem. A complete description about Coding Dojo can be found in [5].

Other Coding Dojo aspects that can bring contributions to the education field are listed below:

- A) Fault Tolerance: it is common to fail and learn from mistakes.
- B) Redundancy: think about different strategies to solve a single problem.
- C) Baby Steps: solving a problem by moving on through little changes. Every little change must be tested.
- D) Peer work: ideas and solutions are implemented by two people, who must help each other.

All these features make Coding Dojo an extremely valuable method for education. The collaborative aspect of this practice is very important because Vygotsky's theory claims that students learn with social interactions. To Vygotsky, in learning process, all the cognitive relations and functions appear twice, first at a social level and second at an individual level [7]. Also in Coding Dojo, students should understand or propose a solution discussing together and then reflect about the results of that solution achieved in performed tests.

Another aspect of Coding Dojo is about problem solving. According to [1], schools should allow children to play with problems. In this context, children could act as creators rather than just consumers of knowledge. This relation will be better described in the subsection 2.2.

2.2. Constructionism

The constructionism theory, proposed by Seymour Papert, is based on student-object interaction. This theory focuses on learning through the mental processes generated in the construction of something from the real world. [1] says that the

theory has "a connotation of sets of pieces for construction, starting with sets in the literal meaning, like Lego, and extending to the inclusion of programming languages, that could be considered as sets from which programs can be made, to kitchens, as 'sets' which are built not just pies, but recipes and math-in-use ways. One of my central mathematical principles is that the construction which happens 'on head' often happens on a specially pleasure way when it is supported by a more public 'world-wide' kind of construction - a sand castle or a pie, a lego house or a company, a computer program, a poem or a theory of the universe".

The acquisition and development of knowledge, according to constructionism, occurs in the constructions that exist in the world with a support for what happens in the mind. In this context, it is possible to affirm that the proposed method directly supports knowledge development, since students are challenged to build solutions to programming problems. Students should think about a solution, build it in real world (implementing it), test it and formulate new hypothesis about the results obtained in the tests. Also, we emphasize that the method focuses on the development of a complete game, which makes the construction experience even more concrete. That happens because a game is an artifact that has several building elements like graphics, animation, audio and interaction and the whole amalgama of such elements brings to students a sense of concreteness of building some 'product'.

2.3. Game Development as an Educational Strategy

According to [12], a game must always use the state-of-the-art in computational terms. The authors claim that these artifacts contain many different elements that include computer graphics modules, multimedia, artificial intelligence, network access, and so on. Thus, these elements need to function harmoniously in an efficient way to respond to the player in real time. So, the development of a game could be an opportunity to put into practice a wide range of concepts from many areas in an integrated way. The same authors also argue that, while most of the conventional softwares should meet a list of requirements, the main feature of a game is that it should be fun and enjoyable to use, as its main goal is to provide entertainment for its user. All these features "make game development a fascinating area full of challenges and 'magic'" [12].

In another hand, [9] reports that one of the reasons frequently mentioned by researchers in the use of such strategies is the attraction that games provide for the new generation students.

Some features that can make the game development a rewarding activity are listed below:

- Graphical Feedback: Students can see results through animations and colorful graphics instead of monochromatic text.
- Cultural aspects: games are part of everyday life of many people; nowadays one can play games on mobile phones, computers, and devices with internet access, like TV and so on.
- Fun: At any level of a game, it should be possible to play and have fun with it.
- Multiple Elements: The game creation involves elements that go beyond programming (and computer science fields). A

game requires creating elements such as drawings, animations and music. This field fusion can captivate creative people.

3. RELATED WORKS

In this section we'll describe related works that involve programming teaching through collaborative methods (such as Coding Dojo approaches) and/or through digital game development.

In a literature review, it is possible to conclude that many researches focus on teaching programming through approaches that involve the use of Scratch programming environment in order to build digital games or similar artifacts. Scratch uses the concept of visual block programming, as many other tools do, like AppInventor. Some results from the literature review are discussed as follows.

An approach to develop basics principles of Computational Thinking along the first year of high school students through game programming in Scratch environment is proposed by [13]. The method was divided into three stages: 1) exploration of Scratch features; 2) planning, implementing and evaluating an educational game project; 3) preparing the final report of activities and project evaluation to create a digital educational game. The same authors suggested that students should work on projects in groups in order to develop interpersonal skills and to experience a collaborative learning situation. They still claim that the experience allowed to stimulate on students logical reasoning and problem solving in a playful and dynamic way; developed basic programming and computational thinking skills; stimulated conflicts resolution and decision making to reach a common objective because of the collaborative nature of the projects.

On other hand, [14] proposed a Writers' Workshop to middle school children in order to connect educational practices between digital animation composition process and computer science. The researches used the Scratch programming environment, allowing students to design, review, and publish their own digital stories. The same authors argue that collaboration could be improved because students had the opportunity to share their digital stories with each other in the workshop and in the wider Scratch community. The results indicates that the biggest part of the students have enjoyed the workshop and agreed that they have learned more about programming.

An experience of applying a Scratch workshop in order to motivate students from the first semester of Information Systems course at the Federal University of Amazonas (UFAM), Brazil, was reported by [15]. The approach considered to propose challenges applied after teaching the main concepts. To evaluate this experience a survey was applied; results showed that Scratch can help Students to understand programming concepts and also claim that they were enthusiastic about using the environment.

Following the review, [16] propose the application of interactive activities using Scratch. The work was conducted as a teacher innovation project at the University of Santiago in Chile. Activities followed progressive levels of complexity. It was done to gradually improve students' Computational Thinking skills. A diagnostic test and a final test were applied to identify the level of Computational Thinking developed in them. According to the authors, these tests have got promising results.

A game building workshop was proposed by [9] in order to verify relations between Computational Thinking and Math knowledge.

The definition of this approach was inspired by some principles of constructionism and Problem-Based Learning (PBL). The data analysis methods used include ethnography, documental analysis and a quasi-experiment. Results showed the students' previous experience with games was a relevant motivational aspect to conclude the project.

A game development-based strategy for teaching software design patterns through challenge-based learning under a flipped classroom approach is presented by [17]. In this approach, a Game Design Document about a game proposal is presented to students in a gradual manner, during some weeks. Students are meant to understand the requirements of each week and develop a class diagram that gradually could model these requirements. A final pitch and evaluation is proposed in order to motivate students to discuss their results with colleagues and allow them to learn from others. To evaluate the method, [17] performed a survey in order to try to capture students' self-awareness of learning. The results showed advances on the software design patterns learning.

Finalizing the review, [18] propose adaptations to use Coding Dojo in Computer Science undergraduate courses. The authors applied a sort of class dynamics based on Coding Dojo on an advanced Data Structures course at the Fluminense Federal University (Brazil). The evaluation of this dynamics were made by surveys to be anonymously answered by students after the application of different models of Dojos. The results showed that shyness and demotivation are the major barriers faced by the new proposal since they reduced pilot exchanges. They also showed that laboratory classes with pre-built tests were the most accepted dynamic by students.

4. METHOD.

This section will describe the proposed teaching method as already described in [4]. As previously explained, the method is a Coding Dojo adaptation to be applied in a classroom environment in programming introductory classes. The main idea is to improve freshmen interest about the basic programming concepts.

The method defines some basic roles that are: technical leader, a teacher who will apply the activity; and the viceleader, a person who should have the technical knowledge about the technology used in the activity. The method also require some organizational structure: first, students must organize themselves in pairs and each pair must have only one computer available; second, in the front of the lab, a main computer should be previously prepared with a multimedia projector for everyone be able to see what have been done.

Due to time constraints, the artifact to be developed during the Dojo should be planned as a simple game. This will also facilitate the learning experience at the activities. It may be recommended to design a casual game, which involves a large heterogeneous audience and could be interesting for most students.

The implementation should regard a set of different logical challenges. These challenges could be solved in a progressive, but independent, sequence. An example should be: (1) make an object move on screen, (2) implement an user interaction with the object, (3) add points to player's score, etc. This sequence of challenges stimulates some Computational Thinking skills, like contemplating the division of a larger problem into less complex sub-problems that must be solved step by step through logical strategies.

As a programming tool, we recommend the MIT AppInventor [19] mobile application development environment. This environment allows the development of simple applications in an easy and fast way. Like Scratch, MIT AppInventor has the visual block programming model that abstracts complex concepts from programming languages and helps students to quickly learn how to create functional mobile applications including mobile games.

The proposed teaching method is divided into 3 main stages. The general guideline of the methodology is presented as follows:

1) Contextualization:

- A) The technical leader initiates the activity presenting the programming environment and the resources available;
- B) The leader must show and explain the game already implemented and working. This game should be the goal of the Dojo. He may ask some students to interact or play it, to stimulate the curiosity of the other students;
- C) The leader should begin the environment preparation by adding and explaining the objects that will be used in the development of the game. In this step, the scenario is built.

2) Action:

- A) Students will be asked about a feature to be implemented in the game. In this step, the answers will be analyzed and a viable proposal must be selected;
- B) Students should elect two students, preferably of different pairs, to be the pilot and the co-pilot of a new feature implementation;
- C) The pilot and co-pilot will try to implement a solution, with the help of other students, who can volunteer with the leader's help;
- D) Once the implementation is finished, an application test should be done, which the pilot and co-pilot should "play" the functionality that had been implemented;
- E) If the test shows that the programming was successful, one should return to step 2.A, otherwise, one should return to step 2.B to solve the detected error.

3. Finish: If the game finishes, or if the activity runs out of time, each student will be able to run his application and have fun with it, later suggesting improvements or new games for a possible new activity.

5. EXPERIMENTS AND RESULTS

5.1. Experiments

The experiments were carried out with a second year class of High School integrated with an Information Technology vocational course. About 60 students participated in the experiment, whose were divided into three groups for session. Each session took about 120 minutes. We propose the

development of the game called "Crazy Tomatoes." In this game, some tomatoes appear in random positions on the screen and the player aim is to smash them by touching the device screen.

Figure 1 shows the students performing the proposed activity. The Figure 2 shows a pilot and a copilot working on a solution. For the sake of protecting students' identification, their faces were substituted by a smiley.



Figure 1. Students performing the proposed Coding Dojo activity.

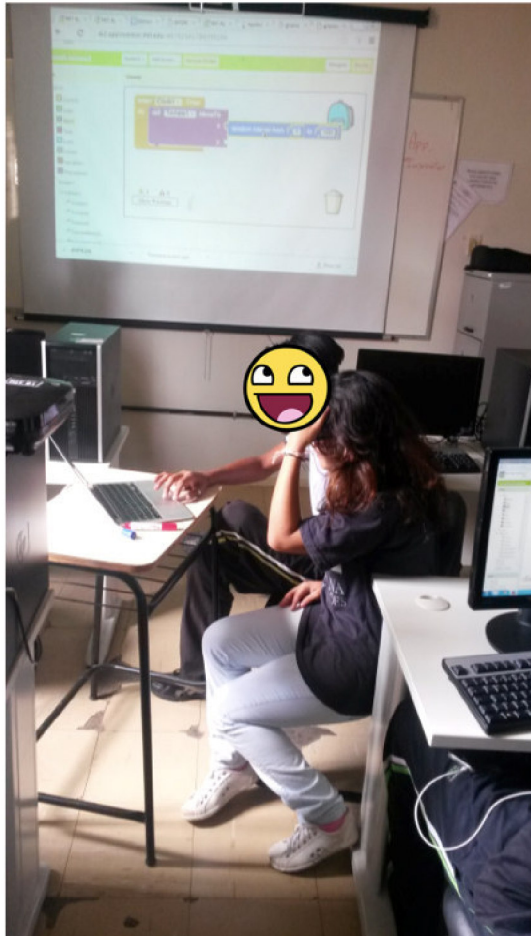


Figure 2. Pilot and co-pilot working in a game feature.

The data collection was done through two approaches: (1) Performing a survey with the students, in the end of the experiment; and (2) Performing an ethnographic observation. So, the data analysis consisted on both quantitative and qualitative approaches.

The survey contained 6 multiple choice questions and one discursive question. The form is listed below:

- 1) Did you enjoy participating the whole activity?
- 2) Did the activity help you better understand programming concepts?
- 3) Are you interested in participating in another Coding Dojo model activity?
- 4) Did the activity motivate you to develop some other application?
- 5) Did you like the activity's model (Pairs helping each other. Everybody replicating what was done by the pilot and co-pilot pair)?
- 6) In your opinion, was there commitment and collaboration between students to help themselves to solve the proposed challenge?

- 7) Make a general comment about the experience of have participated in a Programming Dojo.

Questions 1, 3 and 4 aimed to analyse if the method really could help to motivate students to learn programming concepts. Question 2 had an objective to verify if the approach assists in teaching programming basic concepts. Questions 5 and 6 had the purpose of verify if the students were interested in working collaboratively to solve problems.

The answers to questions 1-6 should be given in the Likert model. Therefore, students could check an item on a scale as follows: 1 - No; 2 - Little; 3 - Indifferent; 4 - Yes; And 5 - Yes, a lot.

Question 7 required a discursive answer. This was made in order to students give suggestions for improvements, describe issues that weren't foreseen in the previous survey's results or even make complaints and talk about some displeasure about the approach.

Ethnography was performed with the objective of adding qualitative data to the results. The approach consisted in observing students during the activities and also consider students' attitudes and decisions for a period of time after the activities were carried out. The ethnography was used in a perspective of understanding if and how the proposed method could increase students' motivation to learn programming as well as the collaborative learning.

5.2. Results

The students were able to develop the complete game in all sessions held. Figure 3 illustrates a given solution to make objects appear at random positions on the screen. Figure 4 shows the game developed by students as a result of the session.

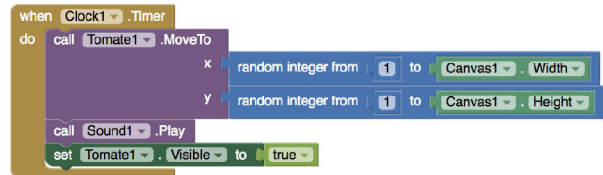


Figure 3. A solution proposed by students to randomly make tomatoes appear on the screen.



Figure 4. The mobile game developed by students

The survey response was optional and anonymous. So, 15 students answered the form. The answers of questions 1 to 6 are illustrated in Figures 5, 6, 7, 8, 9 and 10 respectively. Figure 11 shows the average obtained in each item of the Likert scale.

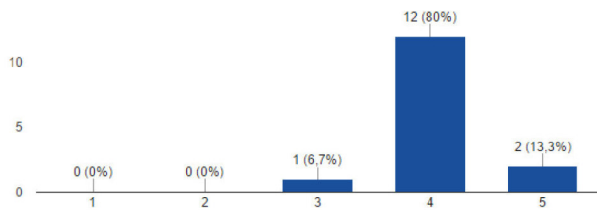


Figure 5. Answers of the question 1: Did you enjoy participating the whole activity?

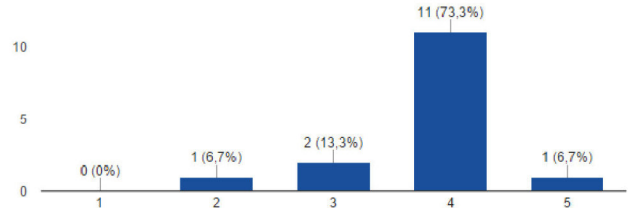


Figure 6. Answers of the question 2: Did the activity help you better understand programming concepts?

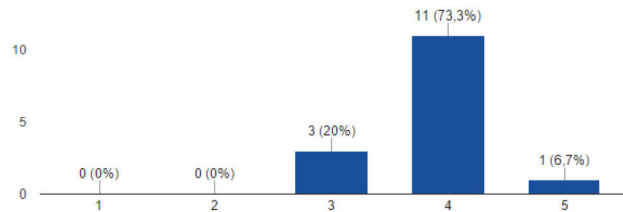


Figure 7. Answers of the question 3: Are you interested in participating in another Coding Dojo model activity?

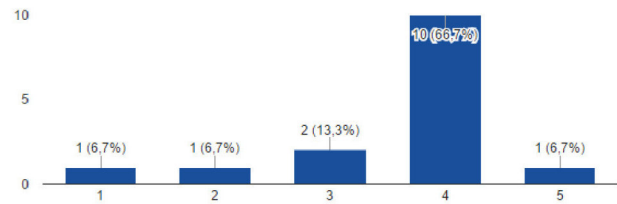


Figure 8. Answers of the question 4: Did the activity motivate you to develop some other application?

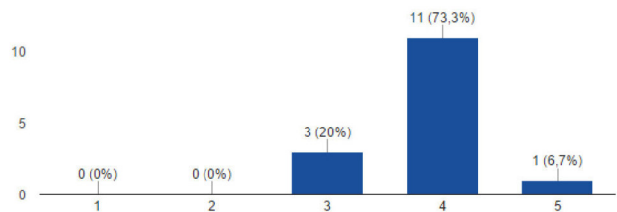


Figure 9. Answers of the question 5: Did you like the activity's model (Pairs helping each other. Everybody replicating what was done by the pilot and co-pilot pair)?

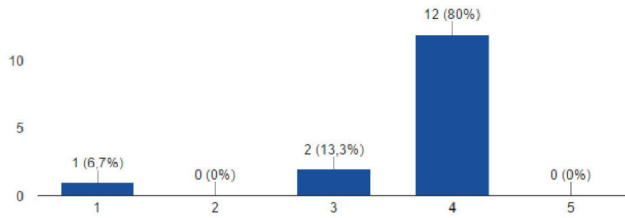


Figure 10. Answers of the question 6: In your opinion, was there commitment and collaboration between students to help themselves to solve the proposed challenge?

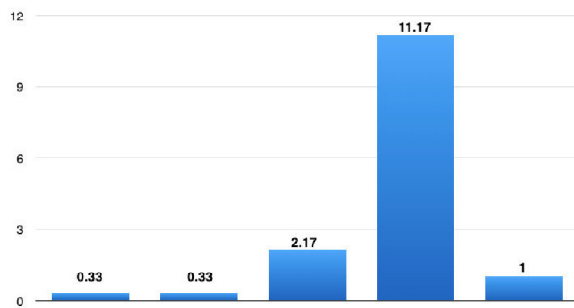


Figure 11. Mean of all the answers.

To perform the data analysis, scores were assigned to each question. This score fitted the following scale: No (-2 points); Little (-1 point); Indifferent (0 points); Yes (+1 point); and Yes, a lot (+2 points). The sum of the points gave us an overview of the results for such questions. A negative score may indicate a disgust about a certain aspect of the method, and a positive score a better contentment with a particular aspect of the approach.

As can be observed in Figures 5, 7 and 8, we consider that the method was a good way to motivate students in programming learning. These issues together reached a combined total of +38 points. According to the answer of question 2, the programming concepts learning can also be benefited through the proposed method. Figure 6 shows that most students understand the approach as a good way to learn these concepts. This question got a total of +12 points. The answers to questions 5 and 6, illustrated by Figures 9 and 10 respectively, indicate that the method has also proved to be a good way to improve collaborative work and consequently learning through social interactions among students. The two issues together reached 23 points. In general, the results showed that the method was considered by the students a good way to learn. Overall, Figure 11 illustrates that most responses were "Yes" and the method obtained a whole evaluation of +73 points.

The answers of the discursive question also showed that students was quite positive in learning thought the approach. Table 1 shows the answers given by the students in question 7. The names that appear in the table are fictitious in order to preserve the

students privacy and anonymity. The names were listed only for information organization purposes.

Table 1. List of responses to the discursive questions. Table with fictitious names to preserve student identity.

Answer ID	Student	Answer
1	Alves	"Interesting and interactive"
2	Bruno	"Interesting and interactive"
3	Cecilia	"It was good, but it should take more time... complicated, with lots of advanced languages."
4	Jorel	"Interesting and interactive"
5	Julia	"Interesting and interactive"
6	Fabiana	"It was a very good practice activity that helped a lot in understanding game development!"
7	Geraldo	"It was cool".
8	Henrique	"It was cool".
9	Iara	"It was pretty cool and it helped me to get more ideas for the same app."
10	Jaqueline	"Great"
11	Lúcio	"Good"
12	Maria	"Nice!"
13	Fernando	"It helped me to have a certain basis in programming. I think they should bring this kind of experience to elementary school students."
14	Otávio	"Cool and interesting"
15	Priscila	"It was a different dynamic, we had the opportunity to do a good practice activity"

As can be seen in Table 1, most students' answers show interest and a certain level of contentment with the experience. Some similar answers were given in sequence, such as answers 1 and 2. We believe that this happen because the same pair of students answered the proposed question together.

We analyzed the answers of the discursive question and raise up some hypotheses and conclusions. Many short answers with expressions of good connotation, such as "cool" and "nice" shows

that students enjoyed participating the activity. The interactivity of the method could be highlighted in answers 1, 2, 4, 5. These answers may have been given due the interaction promoted both among students and student-artifact, once they could discuss between each other and should interact with the game to perform the tests. Iara argues that "*it helped me to get more ideas for the same app*" (answer number 9). It may also show a positive feature of the interactivity of the method because new ideas may have arisen in the discussions between the students during the activity as well as in the interactions with the game. In this context, the answer 15 reports about a "*different dynamic*". It may indicate that these students typically had few interactive classes that follows an old classroom model. This answer reinforces the necessity of using new classroom teaching methodologies that focus on student collaboration and problem solving. A similar analysis can be made looking the answer 13. In this question, Fernando argues that "*they should bring this kind of experience to elementary school students*". This answer may indicate that in his personal experience (or observation of elementary education) this kind of interactive model of teaching normally isn't carried out.

Only one student (Cecilia) came up with a negative review, as can be observed in answer 3. It seems that for her the activity should have taken more time because the practice is complicated due advanced programming language. This criticism is understandable because students have never used environments like ApplInventor (with visual programming capabilities) as a development tool before. A possible solution to this situation would be to increase the period of contextualization which could include an application of a small workshop about the development tool.

As previously reported, the ethnography was applied during the activities and during a period of time after the approach were performed. The objective of applying this technique was to observe if students were interested and motivated about the subjects related to programming and game development after the application of the proposed method. The main results are listed below:

- A) Attention and focus: students were focused on solving problems. During the activity, we didn't notice students who stopped following the game development to access social networks, use cell phones or perform other activity that involve distraction.
- B) Engagement: Students demonstrated determination to solve problems in order to watch the game functionalities working. Students also sought to explore the capabilities of the ApplInventor tool in order to propose new solutions.
- C) Collaborative work: In general, students were willing to make suggestions about the implementation. No resistance was noticed from the students who were chosen to be a pilot and a co-pilot. Moreover, we have noticed a certain enthusiasm from some students. The students who assumed this leadership role also did not show resistance in implementing the solutions suggested by their colleagues. The work in pairs was also carried out cooperatively, which for us, there were no hard conflicts of ideas and disagreement between pilot and co-pilot.
- D) Fun: Students demonstrated amusement expressions as they tested and interacted with the game. This occurred even during initial functionality tests.

- E) Confidence: students felt confident when they realized they could develop a complete game to be played on their cell phones devices.

Some facts were also noticed after the method application. A student developed a game in the Star Wars movie theme based on Crazy Tomatoes (developed during our activities). This student presented the game to his friends as well as to some teachers who showed enthusiasm for this result. Another fact is about the motivation in game design. The Information Technology technical course's students are required to develop a final project in order to graduate on it, and these projects should involve a software design. In the class that students participated of the proposed method activities, 11 of them were interested in the game design as a conclusion project. This large number of interested students in games development was not observed in the two previous classes in their conclusion projects. It's not possible to know certainly that the method was a determining factor for the choice of these students, but hypothetically it may have influenced the choices since the students noticed that they are able to develop a complete game.

5.3. Discussion

In this section we present a results' triangulation of the performed research techniques. We also present some brief considerations about the method.

The triangulation has pointed to the main aspects of the results convergence:

- Interest: answers from question 3 of the survey, answers (1), (2), (4) and (5) of the discursive question; implicitly in other answers of the same question; and item (A) of the ethnography results.
- Collaboration: answers of questions 5 and 6 of the survey; item (C) of the ethnography results; and possible implicit in the answers (1), (2), (4), (5) and (14) of the discursive question.
- Fun: answers of question 1 of the survey; item (D) of the ethnography result; and implicit in answers 7, 8 10, 12 and 14 of the discursive question of the survey.

The problems reported by [18], which relate to student shyness to participate as a pilot and co-pilot of Dojo, were not found by any of the used research techniques. However, some limitations need to be reported. We don't measure how much students were actually learned programming concepts. The results are based on the students' own perceptions. A pre-test and a post-test could be applied to obtain this result. The short activity application period, around 120 minutes, also needs to be reevaluated. Although the time was enough to develop the complete game, it would be interesting to have some extra time for exploring and learning more about the development platform.

6. CONCLUSIONS

This paper presented an adapted Coding Dojo teaching method, designed to motivate students in programming learning through the development of games for mobile devices. Experiences were carried out with classes of about 60 students. To evaluate the method, techniques of ethnography and a survey application containing multiple choices and an open question were performed. The triangulation of results showed that the approach is capable of stimulating students' interest, collaboration and fun. As a

limitation, the research did not measure how much students have effectively learned programming concepts. As future work, we intend to apply a pre-test and a post-test to know the level of students' evolution in each programming concept. We also intend to apply the method in a longitudinal way, with more time available to teach the programming environment features before applying Coding Dojo. This will be done in order to find out if better results could be obtained.

6. References

- Papert, S. (1993). *The children's machine*. Technology Review-Manchester NH-, 96.
- Micael Souza, D., da Silva Batista, M. H., & Barbosa, E. F. (2016). Problemas e Dificuldades no Ensino e na Aprendizagem de Programação: Um Mapeamento Sistemático. *Revista Brasileira de Informática na Educação*, 24(1).
- Wing, J. M. (2006). "Computational thinking. *Commun*". *ACM*, 49(3): 33–35.
- Hidden Reference.
- Sato, D. T., Corbucci, H., & Bravo, M. V. (2008). Coding dojo: An environment for learning and sharing agile practices. In *Agile, 2008. AGILE'08. Conference* (pp. 459-464). IEEE.
- Bastos, H. (2014). Coding Dojo: Muito além do código. Retrieved August 23, 2017, from <https://www.youtube.com/watch?v=RaNcCOBb3RI&t=7s>
- Moreira, M. A. (2011). Aprendizagem Significativa: Um Conceito Subjacente in *Aprendizagem Significativa em Revista/Meaningful Learning Review – V1(3)*, pp. 25-46, 2011.
- Jesus, Â. M. de, Gonçalves, D. A. S., & Ferreira, L. A. C. (2014). Aplicação de Desenvolvimento de Jogos Digitais como um Meio de Motivação em Diferentes Níveis de Ensino de Computação. In *Proceedings of Workshop de Informática na Escola (Vol. 20, No. 1, p. 56)*.
- Barcelos, T. S. (2014). *Relações entre o pensamento computacional e a matemática em atividades didáticas de construção de jogos digitais*. 2014. PHD Thesis in Ensino de Ciências e Matemática, Universidade Cruzeiro do Sul. São Paulo, Brasil. 276 p.
- WEBGOAL. "Coding Dojo" Retrieved January 10, 2017, from <http://www.webgoal.com.br/coding-dojos>.
- CodingDojo.org. "Coding Dojo". Retrieved February 20, 2017, from <http://codingdojo.org/WhatIsCodingDojo/>.
- Feijó, B., da Silva, F. S. C., & Clua, E. (2009). *Introdução à Ciência da Computação com Jogos*. Elsevier.
- Rodriguez, C. L.; Zem-Lopes, A. M., Marques, L. e Isotani, S. (2015) "Pensamento Computacional: transformando ideias em jogos digitais usando o Scratch". In: *Proceedings of XXI Workshop de Informática na Escola (WIE 2015)*.
- Burke, Q., & Kafai, Y. B. (2012). The writers' workshop for youth programmers: digital storytelling with scratch in middle school classrooms. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 433-438). ACM.
- Belchior, J. H. F. et Al. (2016). Avaliando Aspectos Motivacionais do Uso da Ferramenta Scratch para Ensino de Programação: Um Relato de Experiência. In: *Nuevas Ideas en Informática Educativa TISE/Proceedings of XXI Congreso Internacional de Informática Educativa*, Volumen 12, p. 618 - 623. Santiago de Chile.
- Capot, R. B. & Espinoza R. M. Desarrollo del Pensamiento Computacional con Scratch. In: *Nuevas Ideas en Informática Educativa TISE/Proceedings of XX Congreso Internacional de Informática Educativa*, Volumen 11, p. 616 - 620. Santiago de Chile.
- Silveira, I. F. (2016). A Game Development-based strategy for Teaching Software Design Patterns through Challenge-Based Learning under a Flipped Classroom approach. In: *Proceedings of 24º Workshop sobre Educação em Computação WEI*. p. 1996-2005.
- Carmo, D. H., & Braganholo, V. (2012). Um estudo sobre o uso didático de dojos de programação. In: *Workshop de Educação em Computação (WEI)*. Sociedade Brasileira de Computação.
- MITAppInventor. AppInventor Retrieved August 10, 2017, from <http://appinventor.mit.edu/explore/>.