A Methodology to Conduct Computational Thinking Activities in Children's Educational Context

Flavio Nicastro, M. Cecília C. Baranauskas and Ricardo da S. Torres
Institute of Computing, University of Campinas, Campinas, Brazil

Keywords: Computational Thinking, Computational Thinking Methodology, Creative Learning, Educational Methods.

Abstract: Computational thinking (CT) development has been receiving considerable attention in academic discussions. In this work, we propose a methodology for conducting computational thinking activities. The results from a case study involving educators and children from 8 to 11 years old shows that they were pleased when engaging in the development of activities following a methodology that encourages the reflection about what they should do and about what results they are achieving. Some challenges were identified, as well as the need of addressing them to reach acceptance of this kind of methodology by both children and educators. Our next step will concern the evaluation of the use of this methodology, together with educators, in the planning of CT-related activities.

1 INTRODUCTION

Education has been undergoing great transformations in recent years. There is a growing number of papers that advocate changes in the traditional educational system. Some of those changes include: finishing the division by subject, finishing the current model of classrooms, or finishing the space-constrained activities that limit the creativity of students. In this new scenario, something that is in great evidence is computational thinking. According to Jeannette Wing (Wing, 2006), pages 33 and 35:

> *"Computational thinking is a fundamental skill for everyone, not just for computer scientists. To reading, writing, and arithmetic, we should add computational thinking to every child analytical ability. Computational thinking is reformulating a seemingly difficult problem into one we know how to solve, perhaps by reduction, embedding, transformation, or simulation. Computational thinking is a way humans solve problems; it is not trying to get humans to think like computers."*

From 2006 on, Wing published four more articles about the subject (Henderson et al., 2007; Wing, 2008; Wing, 2011; Wing and Stanzione, 2016), and by taking into account a literature review, we found several works about this research topic published in various selective conferences and journals (Bau et al., 2017; Yadav et al., 2017; Baranauskas and Carbajal, 2017; Grover and Pea, 2013; Denning, 2017; Araujo et al., 2016; Perković et al., 2010; Barr and Stephenson, 2011; Guzdial, 2008; Basawapatna et al., 2011; Repenning et al., 2010; Lu and Fletcher, 2009). In Figure 1, we can see the result of a Google Scholar search with the term "Computational Thinking", and we note the increasing number of publications related to this term in recent years, especially after 2006.

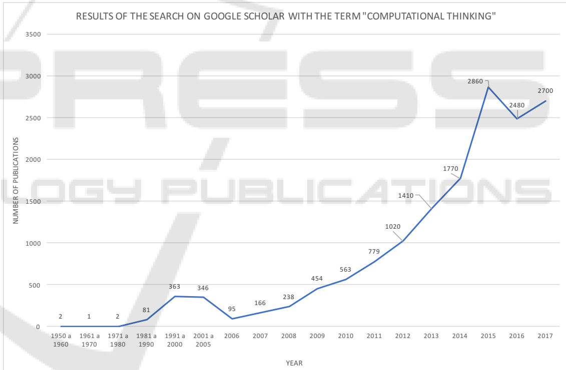


Figure 1: Results of the search on google scholar with the term "Computational Thinking", as of December 2017.

This figure shows that the subject of computational thinking has been gaining more and more space in academic discussions. However, there is still no consensus on its terms and concepts, its boundaries, and its scope (Baranauskas and Carbajal, 2017; Brennan and Resnick, 2012). Based on Wing's definitions

Nicastro, F., Baranauskas, M. and Torres, R.
A Methodology to Conduct Computational Thinking Activities in Children's Educational Context.
In *Proceedings of the 10th International Conference on Computer Supported Education (CSEDU 2018) - Volume 2*, pages 309-316
ISBN: 978-989-758-291-2

and on several other papers that aim to characterize the term, we understand computational thinking as the practice of the set of skills derived from Computer Science that make the person able to approach problems in situations of his daily activities.

In recent years, there has also been a considerable increase in the number of systems used to teach programming with lesson plans and guided activities, such as Scratch[1], Code.org[2], and AppInventor[3]. Systems like those are being used in several educational activities that aim at developing skills associated with computational thinking around the world. However, when conducting activities with such systems, educators do not follow or are not aware of well-established procedures to guide students in their problem-solving tasks. In the literature review, we have not found a methodology that was formally described with the objective of conducting activities that aim at the development of skills associated with computational thinking.

Given this scenario, this work proposes a methodology to help educators on conducting activities that aim at the development of skills associated with computational thinking. With the objective of analyzing the performance of children in activities related to the development of skills associated with computational thinking, using the proposed methodology, we carried out two workshops in August and September of 2017. These workshops are detailed in Section 4 and involved educators and children from 8 to 11 years. The results showed that both were pleased when engaging in activities following a methodology that encourages them to reflect on what they should do and on the results they are achieving. Some challenges were identified, and it is necessary to address them to foster the adoption of practices that aim at the development of skills associated with computational thinking.

The remaining of this text is organized as follows: Section 2 presents related work in the area of computational thinking; Section 3 introduces the proposed methodology, while Section 4 describes a case study concerning the use of the proposed methodology in activities conducted with children. Finally, Section 5 presents our conclusions and directions for future work.

[1]https://scratch.mit.edu (As of Dec. 2017).
[2]https://code.org (As of Dec. 2017).
[3]http://appinventor.mit.edu/explore/ (As of Dec. 2017).

2 BACKGROUND

Computational Thinking is not something new. In a search for publications with this term, we found some papers on mathematics teaching (Bowes, 1955; Arnold, 1962) published in the 1950s and 1960s. More specifically, related to the context of computer use in education, we found the article "Uses of Technology to Enhance Education" (Papert, 1973). In this paper, Seymour Papert describes the proposal sent to the National Science Foundation (in the United States) asking for support on research about children's thoughts and elementary education. In the 1970s, Seymour Papert pioneered the creation of the LOGO language and in the studies about the learning processes of children mediated by the use of programming languages and artifacts, such as a robot turtle that helped to understand the concepts of geometry through programming in the LOGO language (Papert, 1980).

In 2006, the publication of Jeannette Wing, "Computational Thinking" (Wing, 2006), had a significant impact in the computers & education community, bringing popularity and interest of academics to the subject. In this publication, Jeannette Wing presents computational thinking as a set of problem-solving mental processes derived from Computer Science applicable to any domain. It introduces the idea that computational thinking is a central practice for all sciences which is a fundamental and useful analytical thinking skill for all people. It is used to break a difficult problem into more familiar pieces that we can solve (problem decomposition), using a set of rules to find solutions (algorithms), and using abstractions to generalize these solutions to similar problems.

From this pioneering initiative on, several works have been published on this topic. The article, "Computational Thinking for Teacher Education" (Yadav et al., 2017), reviews various papers that have been written on this topic. The author draws attention to the growing enthusiasm for computer science education in many countries, such as Australia, USA, and England. He points out that in 2012 the Royal Society in England published that: "Every child should have the opportunity to learn concepts and principles of computing, including computer science and information technology, from the beginning of primary education". It also states that in 2016, the US College Board launched a new High School Computer Science curriculum called "Principles of Computer Science", with a focus on exposing students to computational thinking to help them understand how computing influences the world. The author stresses that within the computer science education community, computational thinking is a familiar term, but there is confusion among teachers and managers of what is really meant, mistakenly compared to the use of technology and computers. In addition to presenting the definition of computational thinking by Jeannette Wing in 2006, he suggests some ideas that would make it relevant to the key players involved in elementary education and teacher training programs. Complementing the definition of Wing, the authors state that, according to the Computer Science Teachers Association (CSTA) and the International Society for Technology in Education (ISTE), computational thinking relies on nine core concepts and capabilities: data collection, data analysis, data representation, problem decomposition, abstraction, algorithms and procedures, automation, parallelization, and simulation.

Although the concept is still formally undefined, there is some agreement on its meaning in the direction that computational thinking involves the mental processes that occur in solving problems by creating computer programs to solve them (Baranauskas and Carbajal, 2017).

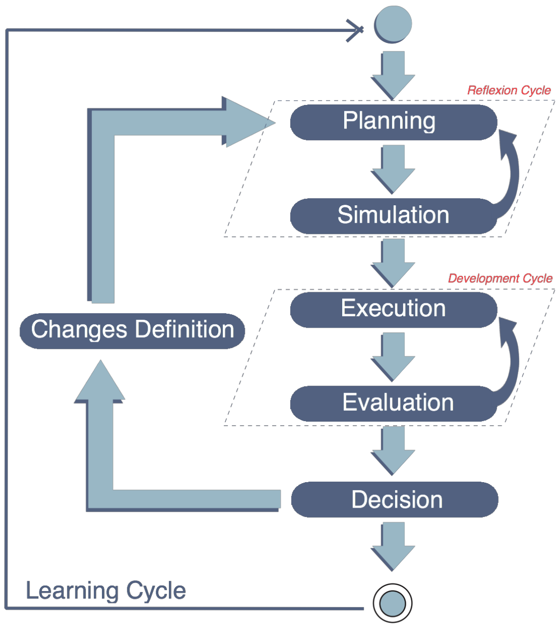


Figure 2: Proposed model for the process of conducting activities.

3 THE PROPOSED APPROACH

In this work, we propose a methodology to conduct activities that aim at the development of skills associated with computational thinking through a process model to be followed in the conduction of these activities. In Figure 2, the methodology steps, which will be detailed as follows, can be visualized. The symbol of a full circle is used to indicate the initial state of the process, while a small circle filled with an empty circle indicates the end of the process. The rectangles with the rounded corners represent the main steps. The first and second steps contemplate the reflection cycle and the third and fourth, the development cycle. The light blue full arrow represents a precedence relationship between steps. The full dark blue curve shows a step in a cycle that must occur while are not ready for the next one. The final arrow indicates that, at the end of the process, the work can be restarted by initiating a new learning cycle.

The model is comprised of the following steps:

1. Planning: This stage consists in defining the objectives of the activity and listing the steps that must be followed to achieve such objectives.
2. Simulation: This step contemplates passing the planning as if it were running, verifying if the defined steps lead to the objective of the activity. In case of inconsistency, the *planning* step should be performed again.
3. Execution: In this step, the plan will be put into practice. Considering, for example, an activity in which a computer program should be developed, one should code and test whether it is in accordance with the plan.
4. Evaluation: This step consists in evaluating the execution result. Should remain in the cycle Execution→Evaluation→Execution while failures in the execution are identified.
5. Decision: This step consists in verifying whether the objective has been reached satisfactorily. If so, the process should be terminated, and a new learning cycle could be started. If not, one should review the provided solution, define necessary changes and return to the *planning* step.

The proposed methodology has the purpose of assisting those interested in conducting an activity that aims to develop a skill related to computational thinking (CT).

In Section 4, we present a case study realized with a group of educators and children from 8 to 11 years old, in which we used this methodology to conduct some CT-related activities.

4 CASE STUDY

With the objective of analyzing the performance of children in activities related to the development of skills associated with computational thinking, using the proposed methodology, we carried out two workshops in August and September of 2017.

The workshops were conducted at PRODECAD[4] with the support of members of the InterHad group[5], as part of the project *"Oficinas de Comunicação Alternativa, Aumentativa e Criativa, com crianças, usando interfaces computacionais vestíveis e tangíveis"*[6], and it consisted in carrying out activities with educators and children using the proposed methodology.

4.1 Experimental Protocol for the Development of Activities

The activities consisted of creating a program on a tablet using ScratchJr[7] following the proposed methodology. The activities were carried out with six educators and 14 children from 8 to 11 years. At the beginning of each session, they were guided on the purpose of the task, and the methodology and its correspondent model was explained.

The solution development was expected to be performed in two parts: first, composed of the planning and simulation steps, the participants received a printed sheet with a scenario (Figures 3 and 6) and a paper with instructions, where they should describe what needed to be done to achieve the goal. After these steps, they received a tablet with ScratchJr and started the second part. In this part, they were expected to create a program in ScratchJr, test, verify the result, and make a decision (if they should redo/improve some aspect or if the result is satisfactory). It was requested that all the reasoning be registered and that if they redesigned something, previous solutions (planning) should not be discarded (erased from the sheet). This procedure aims to allow the analysis of the development of reasoning throughout the session.

The steps to follow were:

1. Plan what to do;
2. Simulate the activity with the help of the scenario on paper;
3. Develop the program in ScratchJr;
4. Test and Evaluate the Result;
5. Analyze and Decide. Was the goal achieved?

   No: What was wrong? Analyze what could have been done differently and return to Step 1.

   Yes: Are you satisfied with the result?

   - **No:** What can be done to improve it? Review and return to step 1.
   - **Yes:** Propose a new challenge and start a new activity.

At the end of each workshop all participants (children and teachers) filled a Self-Assessment Manikin (SAM) instrument (Bradley and Lang, 1994). SAM is a nonverbal instrument of self-assessment of emotions, specifically the level of pleasure, arousal, and dominance, associated with the affective reaction of a person to a stimulus, in this case, the activities with tablets. It consists of a pictographic representation in which each one could register their pleasure, arousal and dominance in relation to the system that was being used. For each of the three dimensions of SAM, the scale of responses ranges from one to nine, where one represents the lowest level (of pleasure, arousal or dominance) and nine represents the highest level. In addition, the children answered a short questionnaire about their perception of the workshop.

4.1.1 Workshop 1: Algorithmic Solution for a Structured Problem

In the first workshop, held on August,22 2017, the program to be developed consisted in indicating the "steps" of a character along a path until arriving at a specific place. More specifically, the program should contain the "steps" needed to get a rabbit to a house (to meet Alice) by traversing the predetermined squares on a board (Figure 3).

In this workshop, we held a preliminary session with the teachers, using the same steps that would be worked on with the children. The teachers have had previous contact with ScratchJr's programming-like environment through TaPrEC (Carbajal and

[4] Programa de Desenvolvimento e Integração da Criança e do Adolescente: offers complementary education for children between 6 and 17 years old, registered in the scholl Escola Sérgio Porto (http://www.dgrh.unicamp.br/dedic/prodecad – As of September 2017).

[5] http://interhad.nied.unicamp.br (As of September 2017).

[6] The project is duly registered in the ethics committee of Unicamp under the CAAE Number: 55678316.4.0000.5404)

[7] ScratchJr is an introductory programming language specifically for tablets, inspired by Scratch (scratch.mit.edu – As of Dec. 2017) and developed in collaboration between the DevTech Research Group of Tufts University, the Lifelong Kindergarten Group at MIT Media Lab and Playful Invention Company (https://scratchjr.org – As of Dec. 2017).

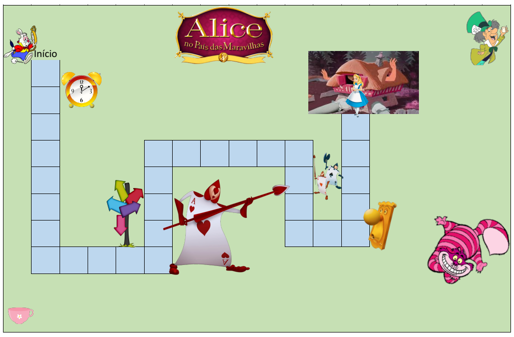


Figure 3: Scenario created for the activity on Workshop 1.

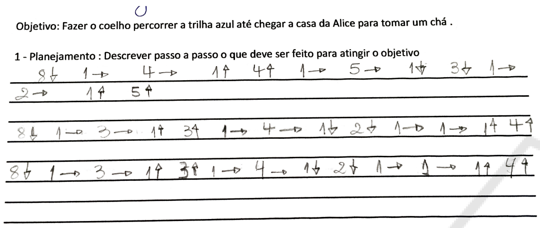


Figure 4: Procedure (in Portuguese) recorded by a group of teachers in Workshop 1.

Baranauskas, 2015). The TaPrEC allows programming based on the use of wooden blocks that are docked to define the movements of the character. The program developed in TaPrEC leads to an algorithm that is more symbolic than descriptive. With this, the two groups of teachers who participated in the workshop simply put a sequence with the number of steps to walk and the direction of the vertical movement (↓ or ↑) or horizontal (→ or ←) - for example, 8 →, 4↓, etc., as we can see in Figure 4.

After the session with the teachers, we performed similar procedures with the participation of the children. They were split into pairs and the forms were handed over to them. After the planning phase, the children received a tablet with ScratchJr, preconfigured with the board and programmed so that, when the rabbit meets Alice, a bell would be played and an invitation (to have a tea with Alice) would appear on the screen.

All pairs of children used literal descriptions of what should be done. Some children opted for more detailed descriptions such as *"He has to walk eight squares forward, turn right and ..."*. Other pairs were based on the drawings that were in the path: *"First go through the clock and then go through the plate and turn right ..."*. Examples of these solutions are shown in Figure 5.

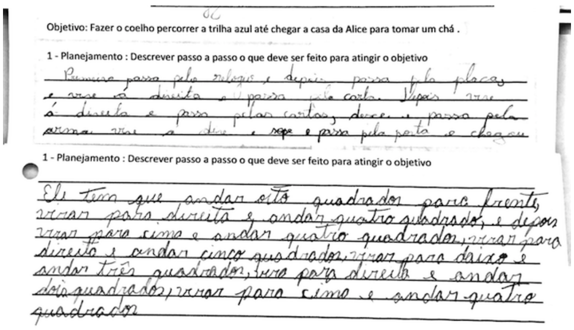
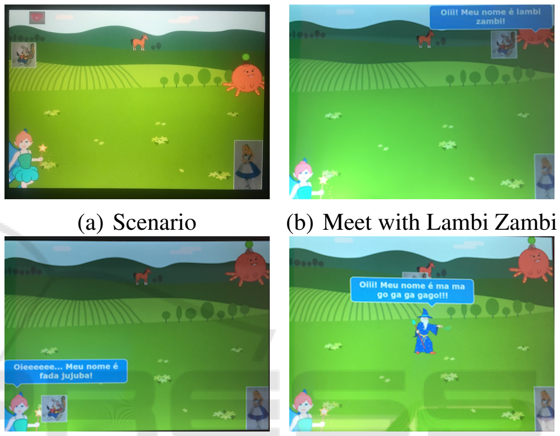


Figure 5: Procedures (in Portuguese) registered by two pairs of students in Workshop 1.



(a) Scenario (b) Meet with Lambi Zambi

(c) Meet with the fairy (d) Find the wizard

Figure 6: Example of screens related to the scenario and the meeting of characters in Workshop 2.

4.1.2 Workshop 2: Algorithmic Solution for an Unstructured Problem

The second workshop was held on September, 12 2017. In this workshop, a strategy was adopted in which the participants would have more freedom to create their solutions. They could determine their own path to be taken, without any paths predefined on the scenario. As in workshop 1, the goal was to take the rabbit character to meet Alice. We also included some other characters in the scenario, whose names should be discovered. They were also told about the existence of a hidden character and that they should find out who it was. The tablet was customized with the scenario as shown in Figure 6 (a). Also, when the rabbit meets another character in his way to Alice, a message appears on the screen with the character's name. When the horse in the scenario is met, a sound is emitted and a wizard (the hidden character) appears on the middle of the screen saying his name – as shown in Figure 6 (b), (c), and (d).

In this workshop, children were already aware of

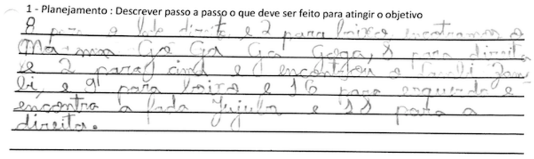


Figure 7: Visibly redone procedure (in Portuguese) by a pair of students in Workshop 2.

ScratchJr and how to develop their solutions on the tablet because all of them took part in workshop 1. This caused an anxiety in most of them in the planning phase, because they wanted the tablets to program as soon as possible. In fact, most of them finished the planning step very quickly, which led to the creation of very simplistic descriptions of the solutions. On the other hand, two pairs took much longer than the others at this stage, even comparing with mean time observed in the first workshop. These pairs actually engaged in the planning and simulation phase, including drawing paths in the scenario. When they started programming, they were much faster and made a much more succinct program.

Another point we noticed is that children resist the idea of not erasing what they planned when they redo it. Note, for example, in the procedure described by a pair in Figure 7, in which even the names of the characters are already present; something that could only be discovered after running the program on the tablet.

4.2 Results and Analysis

During the planning and simulation phase, all pairs were quite entertained and developed their own planning to solve the task. We note that everyone was engaged in the paper planning and simulation and after receiving the tablet, developed their programs in accordance with what was planned. But when the execution did not reach the goal, they did not go back and analyzed the planning to understand what was wrong, to devise and implement possible new solutions. In general, they opted for a trial-and-error-based solution strategy, instead of analyzing possible mistakes in a planned solution to redesign a novel solution for the problem. In the end, all the pairs succeeded in reaching the goal with regard to proposing a correct solution to the problem.

From the comments on the questionnaire (which contained questions about what one liked and disliked the most, and about suggestions from both the workshop and the system), we can observe that both educators and children generally liked the workshops and did not point any negative aspect.

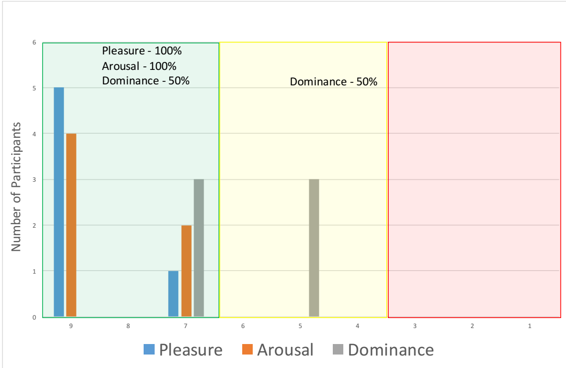


Figure 8: Results of SAM with teachers – 1st Workshop.

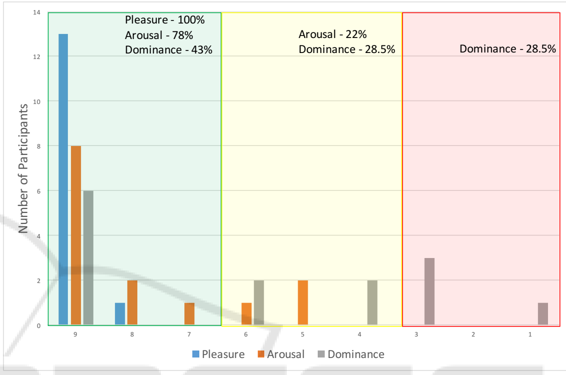


Figure 9: Results of SAM with children – 1st Workshop.

Among the teachers, one suggested the inclusion of commands to make the execution faster (e.g., commands to skip squares), another suggested the diversification of commands in the system, and two teachers complained about the size (possibly referring to the size of the scenario on the tablet screen).

Figures 8, 9, 10, 11, 12 show the results about the SAM and the perception of the workshops.

With regard to the teachers (Figure 8), in relation to the dimensions of Pleasure and Arousal, all of them experienced a positive affective response. In relation to the Dominance, 50% presented a positive affective response, and 50%, neutral.

With the children (Figures 9 and 10), we obtained 100% positive responses to the Pleasure dimension, both in the first and second workshops. In the dimension of Arousal, 78% positive affective response in the first one increasing to 93% in the second workshops. In the Dominance dimension, we noticed the negative affective response by 28.5% of the children in the first workshop, reducing to 14% in the second workshop.

Regarding how they evaluate the activities and how they evaluate the system (Figures 11 and 12), 100% evaluated as *Very Good* or *Good*.

Based on the conducted workshops, we came up with some guidelines that may improve the imple-

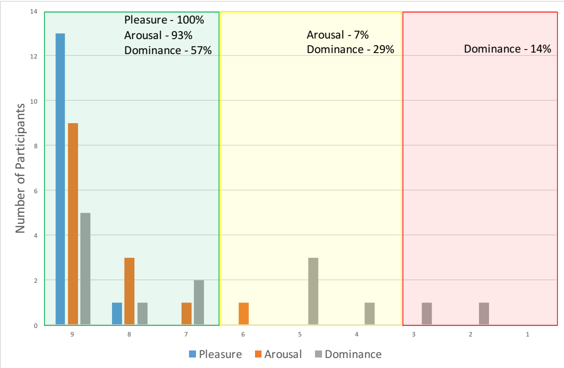


Figure 10: Results of SAM with children – 2nd Workshop.

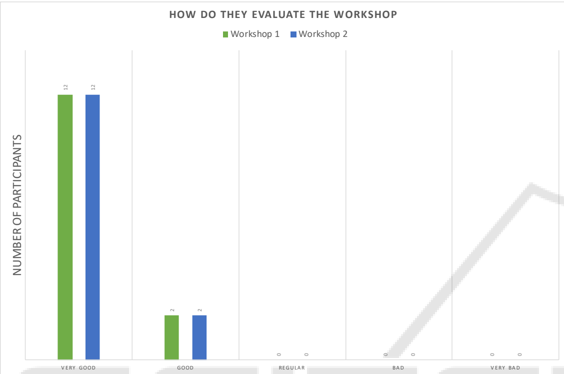


Figure 11: Workshop evaluation by children.

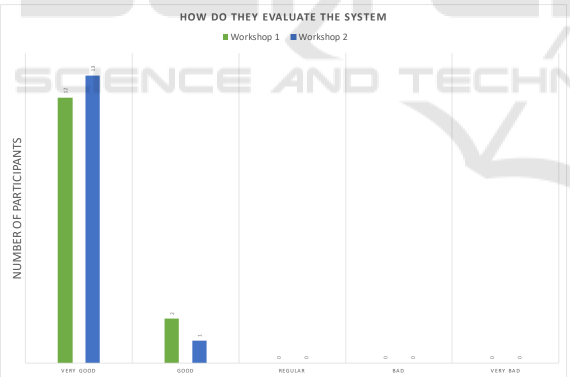


Figure 12: System evaluation by children.

mentation and evaluation of this kind of activities using the propose methodology:

- Record the time spent in each activity and correlate the observed times with the quality of the program developed.
- Observe the comments of children during their interaction, both in the planning and simulation phase, as well as in the programming and testing phase.
- Develop strategies to keep track of the development of the reasoning strategies of children.

5 CONCLUSIONS

Although the subject of computational thinking has attracted attention of researchers in the computers and education field, especially in the last years, literature still lacks a formal methodology to address the development of skills associated with computational thinking. In this paper, we analyzed existing literature on computational thinking and proposed a methodology to conduct activities that may foster its development in an educational context.

We experimented the proposed methodology with children from 8 to 11 years old and their teachers. The activities aimed at the development of computational thinking skills by means of solving proposed problems using ScratchJr in a tablet. The analysis of results from these workshop shows that both educators and children were pleased when engaging in the development of activities following a methodology that encourages them to reflect about what they should do to solve a problem and about the results achieved.

For future work, we intend to evaluate the use of this methodology in other educational scenarios, and conduct some workshops with educators using this methodology for planning activities. Additionally, we also plan to develop a platform for supporting the use of the methodology in the development of computational thinking, with the possibility of sharing activities, experiences, validations, and evaluations of the proposed methodology.

ACKNOWLEDGEMENTS

Authors are grateful to CAPES, CNPq (grants #308618/2014 − 9 and #307560/2016 − 3), and FAPESP (grant #2015/16528 − 0) funding agency for their support.

REFERENCES

Araujo, A. L., Andrade, W., and Guerrero, D. (2016). Um mapeamento sistemático sobre a avaliação do pensamento computacional no Brasil. In *Anais dos Workshops do Congresso Brasileiro de Informática na Educação*, volume 5, page 1147.

Arnold, H. N. (1962). A study of stern structural materials in arithmetic in the first and second grades. Master's thesis, The College at Brockport: State University of New York.

Baranauskas, M. C. C. and Carbajal, M. L. (2017). The social nature of programming: Children and fluency. In *International Conference on Human-Computer Interaction*, pages 291–308. Springer.

Barr, V. and Stephenson, C. (2011). Bringing computational thinking to k-12: what is involved and what is the role of the computer science education community? *ACM Inroads*, 2(1):48–54.

Basawapatna, A., Koh, K. H., Repenning, A., Webb, D. C., and Marshall, K. S. (2011). Recognizing computational thinking patterns. In *Proceedings of the 42nd ACM technical symposium on Computer science education*, pages 245–250. ACM.

Bau, D., Gray, J., Kelleher, C., Sheldon, J., and Turbak, F. (2017). Learnable programming: Blocks and beyond. *Communications of the ACM*, 60(6):72–80.

Bowes, M. L. (1955). *The relationship between the understanding of process meanings and the ability to solve verbal arithmetic problems at the sixth grade level.* PhD thesis, Boston University.

Bradley, M. M. and Lang, P. J. (1994). Measuring emotion: the self-assessment manikin and the semantic differential. *Journal of behavior therapy and experimental psychiatry*, 25(1):49–59.

Brennan, K. and Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada*, pages 1–25.

Carbajal, M. L. and Baranauskas, M. C. C. (2015). TaPrEC: desenvolvendo um ambiente de programação tangível de baixo custo para crianças. In *Anais XX Congreso Internacional de Informtica Educativa, Santiago, Chile*, pages 363–370.

Denning, P. J. (2017). Computational thinking in science. *American Scientist*, 105(1):13.

Grover, S. and Pea, R. (2013). Computational thinking in k–12: A review of the state of the field. *Educational Researcher*, 42(1):38–43.

Guzdial, M. (2008). Education paving the way for computational thinking. *Communications of the ACM*, 51(8):25–27.

Henderson, P. B., Cortina, T. J., and Wing, J. M. (2007). Computational thinking. In *ACM SIGCSE Bulletin*, volume 39, pages 195–196. ACM.

Lu, J. J. and Fletcher, G. H. (2009). Thinking about computational thinking. *ACM SIGCSE Bulletin*, 41(1):260–264.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. Basic Books, Inc.

Papert, S. A. (1973). Uses of technology to enhance education. *https://dspace.mit.edu/handle/1721.1/6213 As of September 2017*.

Perković, L., Settle, A., Hwang, S., and Jones, J. (2010). A framework for computational thinking across the curriculum. In *Proceedings of the fifteenth annual conference on Innovation and technology in computer science education*, pages 123–127. ACM.

Repenning, A., Webb, D., and Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM technical symposium on Computer science education*, pages 265–269. ACM.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3):33–35.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences*, 366(1881):3717–3725.

Wing, J. M. (2011). Computational thinking. In *Visual Languages and Human Centric Computing*, page 3.

Wing, J. M. and Stanzione, D. (2016). Progress in computational thinking, and expanding the hpc community. *Communications of the ACM*, 59(7):10–11.

Yadav, A., Stephenson, C., and Hong, H. (2017). Computational thinking for teacher education. *Communications of the ACM*, 60(4):55–62.