UMA ESTRATÉGIA DE APRENDIZAGEM COOPERATIVA PARA DESENVOLVIMENTO DO PENSAMENTO COMPUTACIONAL POR MEIO DE ATIVIDADES DE PRODUÇÃO DE JOGOS DIGITAIS

A COOPERATIVE LEARNING STRATEGY TO COMPUTATIONAL THINKING DEVELOPMENT

Ângelo Magno de Jesus

Universidade Cruzeiro do Sul/Instituto Federal de Minas Gerais, angelo.jesus@ifmg.edu.br

Ismar Frango Silveira

Universidade Cruzeiro do Sul/Universidade Presbiteriana Mackenzie, ismar.silveira@cruzeirodosul.edu.br

Resumo

A inclusão do Pensamento Computacional (PC) em sala de aula pode trazer grandes avanços para a educação. Através do Pensamento Computacional, os alunos podem exercitar o raciocínio lógico, resolver problemas complexos, lidar com a abstração entre outras habilidades. A colaboração também é um aspecto fundamental da aprendizagem. Interações sociais entre estudantes que podem advir de métodos cooperativos de aprendizagem podem contribuir para a construção de conhecimento de diferentes maneiras. Este artigo descreve uma estratégia de Aprendizagem Cooperativa para mobilizar habilidades do PC em estudantes. Características fundamentais do conceito de Aprendizagem Colaborativa e Cooperativa da literatura foram estudadas e projetadas para se encaixar na estratégia proposta. Além disso, o método proposto utiliza abordagens de Desenvolvimento de Jogos Digitais para envolver os alunos. Uma análise da estratégia foi feita por meio de atividades realizadas com grupos de alunos das séries finais do ensino fundamental. Foram investigados os artefatos produzidos e as interações entre estudantes. Os resultados mostraram que a estratégia foi capaz de mobilizar estratégias de resolução de problemas do PC e reflexões sobre as interações sociais no grupo.

Palavras-chave: Aprendizagem Cooperativa, Pensamento Computacional, Jogos.

Abstract

In Including Computational Thinking (CT) in the classroom can bring great advances to education. Through Computational Thinking, students can exercise logical reasoning, solve complex problems, deal with abstraction and other skills. Collaboration is also a key aspect of learning. Social interactions between students wreaked from cooperative learning methods could contribute to build knowledge in different ways. This article describes the development of a Cooperative Learning strategy to support and mobilize CT skills in students. Fundamental features of cooperative learning concepts from the literature have been studied and designed to fit the proposed learning strategy. Also, the proposed method uses Game Development approaches in order to engage learners. In addition to being present in students' daily lives, digital games enable direct interaction by giving feedbacks to student commands via animated graphics. A strategy analysis was performed through activities carried out with groups of students from the elementary schools' final series. We investigated the created artifacts and the interactions between students. The results showed that the approach was capable to mobilize CT problem-solving strategies and reflections about social interactions in the group.

Keywords: Cooperative Learning, Computational Thinking, Games.

Introduction

Papert and Solomon (1971) presented a series of activities that could be done by students at schools using computers in an innovative way. The cited authors already discussed how concepts of Computer Science affected the way of thinking in biology, psychology and even in the philosophy of mathematics. From this point on, the ideas of using computers and computing concepts in education have been developing. Papert (1993) discusses and presents some situations in which children could build knowledge by making artifacts - especially technological ones. Obviously, this idea (as Papert states) comes from Jean Piaget`s constructivism, but Papert gives special attention to building real-life artifacts, such as an automated little house built with Lego blocks. The potential of computers comes, from its huge possibility of creation. Computer programming is a skill that emerges from Computer Science. Later, Wing (2006)'s work received notoriety among researchers from the computing and education's community. Wing (2006), as well as Papert, proposes the computer science mindset as a essential skills to be learned by all, including to solve problems of various knowledge fields. However, the researcher emphasized in her discussion, tools and concepts of Computer Science that go beyond the use of computers. This approach has been called Computational Thinking and has been highlighted by several researchers that aimed to improve learning and teaching at different education levels. Computational Thinking involves problem-solving from different domains so that solutions can be represented on a computer in the shape of algorithms or data. However, teaching Computational Thinking should not be limited to transfer technical skills, but should also include the creation of an environment with motivations, feelings, and attitudes that require teamwork for a common purpose (BARR and STEPHENSON, 2011). This aspect leads us to another important concept of this study, the Cooperative Learning.

It's not a surprise that cooperation is a positive circumstance for learning. Since the beginning of their existence, human beings have been carrying out collective activities such as hunting and fishing. We are social beings. Vygotsky's sociocultural theory, according to Moreira (2011), already emphasized the importance of social interactions in the learning process. According to the sociocultural theory, learning occurs first in higher mental processes (thought, language and voluntary behavior) that have their origin in social processes. So cognitive development is the conversion of social relations into mental functions (MOREIRA, 2011). Obviously, Cooperative Learning involves using social interactions to learn and build knowledge.

In addition to Cooperative Learning, Digital Games Design and Development can be important allies in Computational Thinking. According to Barcelos (2014), strategies of digital games development are often used by researchers with the justification that games provide a great attraction for new generations' students. Some evidence of digital game development notoriety comes from the popularity of Scratch - a programming environment used to design animations and digital games. Scratch is used in more than 150 different countries and is available in more than 40 languages (MIT MEDIA LAB, 2019).

This article aims to describe a Cooperative Learning strategy for computational thinking development in digital games production activities. In the study Jesus and Silveira (2019), we present an overview of the strategy focused on the use of digital games. Based on this experience, this article describes an improved strategy focused on digital game development, with changes in method stages and using special materials. In order to evaluate how the proposed approach can mobilize learner's problem-solving strategies, the sessions were recorded. So that students' interactions and strategic decisions could be analyzed. The artifacts produced in the activities were also evaluated.

Computational Thinking

Education professionals have always attempted to encourage students to take a less passive attitude and to work actively in order to solve complex questions in different fields of knowledge. Computational Thinking can be a way to assist reaching this goal. Wing (2006) describes Computational Thinking as Computer Scientists think: it means more than being able to program a computer. Barr and Stephenson (2011) gives a definition for Computational Thinking in K-12 classrooms:

> CT is an approach to solving problems in a way that can be implemented with a computer. Students become not merely too users but tool builders. They use a set of concepts, such as abstraction, recursion, and iteration, to process and analyze data, and to create real and virtual artifacts. CT is a problem-solving methodology that can be automated and transferred and applied across subjects.
>
> (Barr and Stephenson, 2011, p.51)

As Brackmann (2017) describes, Computational Thinking is associated with four dimensions that can be seen as its main pillars. These dimensions are described below:

- Abstraction: Selecting relevant data to represent an idea(s).
- Decomposition: Breaking down problems into smaller, manageable parts in order to solve it.
- Pattern Recognition: Analyzing trends and sequences that repeat in data or situations.
- Algorithm Design: Designing an ordered series of instructions for solving problems or for doing a task.

The application of Computational Thinking in the classroom is directly related to Cooperative Learning. Barr and Stephenson (2011) cite values, feelings, attitudes of Computational Thinking that include: "*setting aside differences to work with others to achieve a common goal or solution*"; and "*knowing one's strengths and weaknesses when working with others*". Also, the authors cite that Computational Thinking should create a classroom culture that involves team work by students, with explicit use of decomposition, abstraction, negotiation and consensus building. Negotiation is related groups within the team working together to merge parts of the solution into the whole. Consensus building is associated with working to build group solidarity behind one idea or solution. Technologies, which can be used in Computational Thinking, can also be an important element. According to Webber and Vieira (2010), in collaborative scenarios, technologies take on roles in communication, mediation and motivation of the participants, contributing to the processes of interaction and learning.

Collaborative and Cooperative Learning

Although it seems to be a relatively simple concept, many people are mistaken about what Cooperative Learning really is. Johnson et al. (1984) state that cooperation is not having students sit side-by-side at the same table talking with each other and doing their individual assignments. As the authors explain, cooperation is much more than keeping students physically close, helping other students, or sharing materials with each other - although these are important acts of cooperation. The authors explained that positive interdependence, face-to-face interaction among students, individual accountability and appropriately use of interpersonal and small-group skills are basics elements of Cooperative Learning.

Dillenbourg (2007) states that there is a broad definition of collaborative learning: it is a situation in which two or more people learn or attempt to learn something together. The author argues that this definition is unsatisfactory because each element can be interpreted with different meanings. So, Dillenbourg (2007) defines collaborative learning as below:

> In summary, the words 'collaborative learning' describe a situation in which particular forms of interaction among people are expected to occur, which would trigger learning mechanisms, but there is no guarantee that the expected interactions will actually occur. Hence, a general concern is to develop ways to increase the probability that some types of interaction occur. (DILLENBOURG, 2007, p. 5)

The term Collaborative Learning has been used in different contexts with different characteristics. Many authors don't differentiate Collaborative and Cooperative Learning, assuming that they are the same concept. Besides that, some authors have made efforts to distinguish between the two concepts (PANITIZ, 1999).

According to Panitiz (1999) cooperation could be seem as an structure of interaction designed to facilitate the accomplishment of an specific end product or goal through people working together in groups. In a learning context, it`s a classroom technique. On the other hand, collaboration is a lifestyle and an interaction's philosophy between people. Individuals are responsible for their actions and should respect and highlight the contributions and skills of others. In this way, Collaborative Learning extrapolates the fact that it is only a teaching approach. A collaborative group can collaborate spontaneously in a variety of situations. People interact in order to reach a goal without having an obligation to do so.

According to Matthews et al. (1995), there are aspects in which collaborative and cooperative learning may diverge. As the authors claim, within the context of small-group learning, there is a wide range of views about: the issue of authority and power relationships between teacher and student; the style, function or degree of the teacher's involvement; the extent to which students need to be trained to work together in groups; how knowledge is assimilated or constructed; the purpose of groups to emphasize different outcomes such as the mastery of facts, the development of judgment, and/or the construction of knowledge; the importance of different aspects of personal, social, and/or cognitive growth among students; and many additional implementation concerns including, for example, group formation, task construction, and the degree of individual and/or group accountability necessary to ensure equitable distribution of work and accurate grading.

In this article, we consider the proposed approach as a Cooperative Learning strategy, however there is a possibility that students learn to take the collaboration philosophy out of the classroom, thus becoming a collaborative approach.

Cooperative Learning Features

Cooperative/Collaborative learning approaches involve a variety of features. Table 1 shows different cooperative and collaborative learning features grouped per line. It's important to observe that we clustered only the concepts that we considered explicit. This grouping is important to reduce the complexity of the cooperative strategy. It’s possible to verify that there are other implicit connections between concepts.

Table 1: Fontes de leitura dos alunos excluindo os livros didáticos.

ID	Collaborative/Cooperative Feature	Source
A	Positive interdependence; Members must have the feeling of belonging to the same team; Shared goal; Success or failure will be shared by all members.	Johnson et al. (1984), National Concil of Teachers of Math from Panitz (1999) and Dillenbourg (2007) on a Collaborative Situation.
B	Individual accountability; Individual work has a direct effect in the group`s success.	Johnson et al. (1984), National Concil of Teachers of Math from Panitz (1999).
C	Heterogeneous	Johnson et al. (1984).
D	Shared leadership; Sharing of authority; Consensus building.	Johnson et al. (1984), Panitiz (1999).
E	Shared responsibility for each other; Students teach each other by peer-to-peer exchange.	Johnson et al. (1984), Torres and Irala (2014).
F	Task and maintenance emphasized	Johnson et al. (1984).
G	Social skills directly taught; All students must talk with another to engage in discussion of all problems.	Johnson et al. (1984), National Concil of Teachers of Math from Panitz (1999).

H	Teacher observes and intervenes; Teacher as facilitator; Student centered; Loose - trusting students to do.	Johnson et al. (1984), Torres and Irala (2014) and Lee (199-?) apud Panitiz (1999).
I	Group processes their effectiveness.	Johnson et al. (1984)
J	Respects and highlights individual group members`abilities and contributions.	Panitiz (1999)
K	Metacognition's skills development; Knowledge construction.	Torres and Irala (2014) and Lee (199-?) apud Panitiz (1999).
L	Symmetry - peers with approximately the same level that can perform the same actions.	Dillenbourg (2007) on a Collaborative Situation.
M	Division of labour - work together.	Dillenbourg (2007) on a Collaborative Situation.
N	Intrinsic motivation.	Lee (199-?) apud Panitiz (1999).

Game Design and Development as a Learning Strategy

Lee et al. (2011) explain that Game Design and Development is one of the domains where Computational Thinking takes place. The authors state when learners are designing and developing games, they are exercising skills like: (1) Abstraction - games are abstracted into a set of scenes containing characters; (2) Automation - game responds to user actions; and (3) Analysis - students need to investigate if the elements incorporated make the game fun to play.

Developing games can be fun, Feijó, Silva and Clua (2010) argue that this is a magical, fascinating and challenging activity. The authors assume that digital games are softwares that have the main requirement to entertain its users. In addition, to produce a digital game it's necessary to put into practice many elements of Computer Science such as artificial intelligence, graphics modules, multimedia, etc.

Related Work

Interesting initiatives have been carried out with the purpose of developing Computational Thinking skills in a cooperative/collaborative way. It's a field that has a wide variety of open issues and procedures possibilities, so more researches are still necessary. In this section, we will present some studies related to the approach proposed in this article.

Fronza, Ioini and Corral (2017) proposed the Framework, based on Agile Software Engineering Methods, in order to teach Computational Thinking for middle schools. The framework use iterative processes and allows students to use brainstorming, mind maps and storyboards to develop games and animations. The authors applied the method using Scratch with two sixth-grade classes. Project analysis and artifact-based interviews were used to assess the approach. According to the researchers, the results showed that the framework is promising because students are encouraged to think about their processes and need to be able to define the concept and use it.

The Coding Pirates initiative was presented by Tabel et al. (2017). Coding Pirates is a nonprofit social organization dedicated to promoting computational competencies for children between 7 and 17 years old. Tabel et al. (2017) describe workshops promoting programming as a social activity. The authors applied Peer Programming approach with the Scratch environment. They state that the workshops achieved great success regarding to collaboration within teams and within the whole class. The authors also point out the importance of creating strategies to move beyond keyboard and screen, creating new classroom experiences that foster the engagement of computational thinking.

A teaching strategy as a support tool in the learning of Programming Logic and Basic Electronics is presented by Aquino Filho, Schimiguel and Amaral (2016). The approach adopted a blog, as a collaborative environment, and a learning object. The authors developed a learning object called Kweb. The approach was used with students of a technical course in Mechatronics and Industrial Automation. Surveys were applied to verify the student's level of knowledge and to analyze the learning object usability. The results showed that the strategy enabled students to learn in a more meaningful and motivating way.

Materials and Method

The proposed strategy was designed to fit all cooperative features from Table 1. Figure 1 shows the overview of the proposed cooperative learning strategy. As can be observed, the strategy is composed of 6 stages, each step attempts to explore different cooperative elements described above. It should be noted that Figure 1 was adapted from Jesus and Silveira (2019, in press) since the proposed procedure went through a refinement, changes were made in the order of the stages.

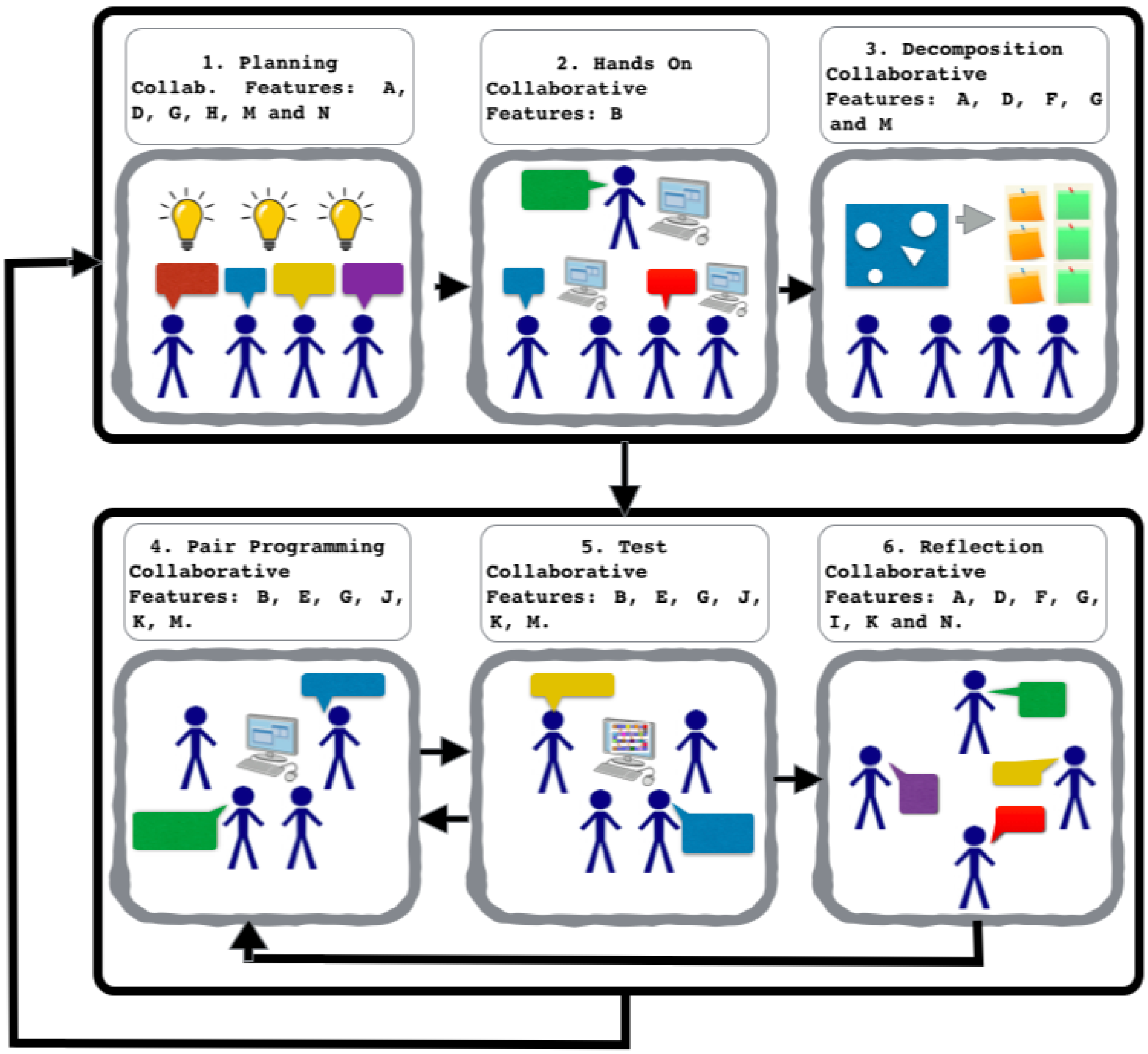

1. Planning
Collab. Features: A, D, G, H, M and N
2. Hands On
Collaborative Features: B
3. Decomposition
Collaborative Features: A, D, F, G and M
4. Pair Programming
Collaborative Features: B, E, G, J, K, M.
5. Test
Collaborative Features: B, E, G, J, K, M.
6. Reflection
Collaborative Features: A, D, F, G, I, K and N.

Figure 1: Stages of the Proposed Learning Strategy

To perform the activities, it is recommended to use the following materials: various post-its, colored pens, a stopwatch, a timer or an hourglass, a small blank or paperboard (preferably cardboard) at least A4 size (Figure 2). In addition, it's also necessary to use at least one computer per team. Before starting the activities, the learners' group can create a web page or a social network group so they can share their ideas with other groups. The stages and use of the materials will be described more detailed in the following paragraphs.

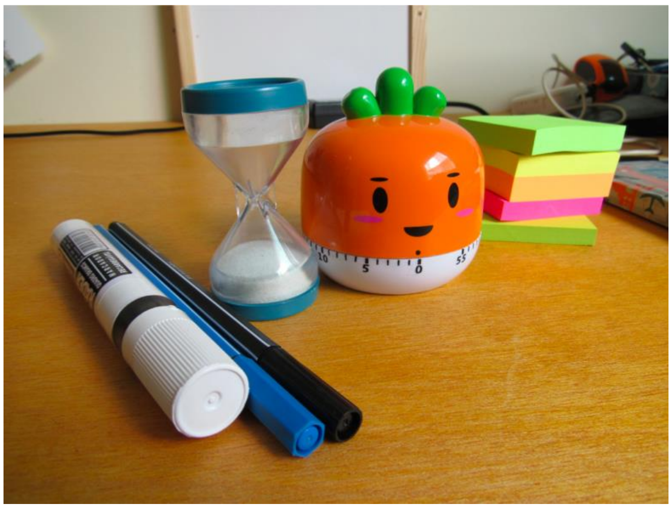


Figure 2: Materials to perform the Cooperative Learning Strategy

Planning - At this stage, the teacher and the students plan a feature or even an entire game to be developed. The game represents the problem/goal to be achieved by the group. Learners can discuss freely, but the teacher must intervene to keep the goal viable to be reached according to the students' current knowledge. The teacher should also be concerned in supporting the goal according to the Computational Thinks concepts which need to be developed. According to Jesus and Silveira (2019, in press) some questions need to be raised at this stage: "*What are the previous expectations? What previous knowledge do they have to solve the problems that may arise? What knowledge do they want to achieve? Which product/solution is expected?*".

Planning is related to the following cooperative features: "*A - Positive interdependence; D - Shared leadership, G - Social skills directly taught; H - Teacher observes and intervenes; M - Division of labour - work together; and N - Intrinsic motivation*" (JESUS and SILVEIRA, 2019, in press).

Hands On - At this stage, the teacher should apply a workshop addressing the concepts that should be used to implement the objective game (proposed in the previous step). Students should be mindful in order to contribute to the group later. This step is the most centralized in the teacher.

This step addresses the cooperative feature B-Individual Accountability as students must learn to help others.

Decomposition - At this stage, students should directly apply the Decomposition ability of Computational Thinking. Therefore, the group will propose the division of the game into several sub-tasks that can be implemented and combined in order to create the final product. Abstraction skills are also necessary because students have to describe imagined game objects and actions. The teacher should mediate this stage by encouraging students to interact and checking if the proposals are consistent. A Kanban board should be modeled by students using some of the materials presented in Figure 2. We suggest that Kanban should be composed of four sections: (1) Objective, (2) Tasks, (3) In Progress and (4) Completed. The first section of Kanban should contain the goal, defined in step 1, for the whole group to observe. When a student proposes a task, he/she should pick up a post-it and a colored pen, write the task and paste the post-it into the Kanban's tasks section. This artifact can help in the activity management by allowing the whole group to have a knowledge of what needs to be done. Figure 3 illustrates the Kanban template proposal.

Decomposition is related to the following cooperative features: "*A- Positive interdependence; D- Shared leadership; F- Task and maintenance emphasized; G- Social skills directly taught; and M- Division of labour - work together"* (JESUS and SILVEIRA, 2019, in press).

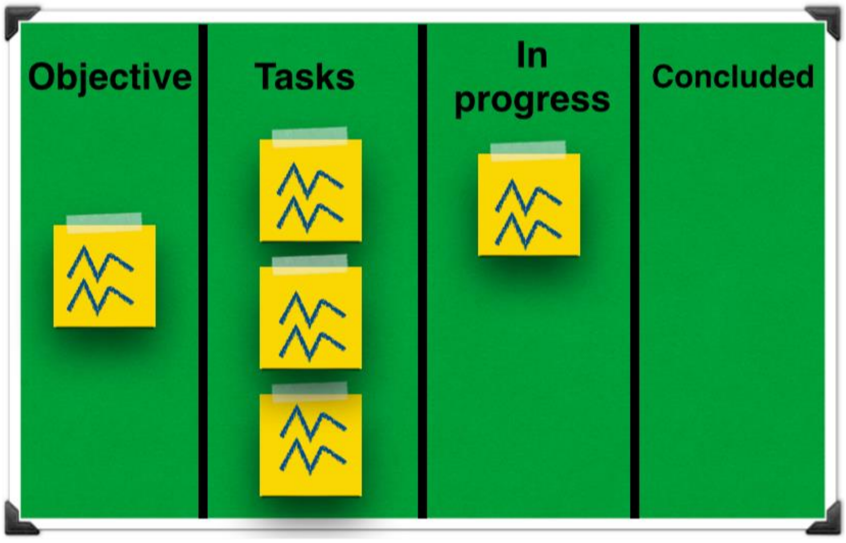


Figure 3: Kanban model for the Cooperative Learning Strategy

Pair Programming - In the Pair Programming step, students must implement one of the defined tasks. This programming model is inspired by Coding Dojo and Agile software development methodologies that consider the collaboration between developers (SOUZA; MARCZAK; PRIKLADNICKI, 2011, p.125). Only one computer per group is used, so students must take turns using it. At this time, it is necessary to use the timer or the hourglass. It's recommended to use a timer with a cartoon style design, as shown in Figure 2. This object can draw the attention of students to have a relation with the theme and culture of digital games incorporated in the proposed strategy. Each moment the time runs out, another learner turns over the pilot, so he/she takes over the keyboard and mouse in order to program the computer. The other students act as co-pilot assisting the pilot at all periods in the programming. The task being implemented should be submitted to the In Progress section of the Kanban board. The teacher's role is only to clarify questions when asked. The teacher can also take advantage of the moments of peer exchange to intervene.

According to Jesus and Silveira (2019, in press), this step addresses the following Cooperative Learning features: B - Individual accountability; E- Shared responsibility for each other; G- Social skills directly taught; J- Respects and highlights individual group members' abilities and contributions. K- Metacognition`s skills development; and M- Division of labor.

Tests - The tests take place in association with Peer Programming. At this time, students can have fun because they must interact with the artifact that they are developing. Students will also be able to record the solution's gameplay to share on the web page or social network group of the team. The cooperative features of this step are the same as in Par Programming.

Reflection - At the end of each session, a quick meeting is required to reflect on the progress of the team's activity and performance. The group must produce a kind of meeting minutes. Some questions that can be answered by the group are: (A) Was the goal achieved? If not, why? (B) What problems were faced? (C) What needs to be improved in the group interaction? (D) Is the game (becoming) fun? (E) What can be done in the next session?

In the next session, the teacher can review with the students what they discussed during the quick meeting so they can think about what actions could be taken during the activity. As described by Jesus and Silveira (2019, in press), this step includes the following cooperation features: A- Positive interdependence; D- Shared leadership; F- Task and maintenance emphasized; G- Social skills directly taught; I- Group processes their effectiveness; K- Metacognition`s skills development; and N- Intrinsic motivation.

Feature C involves the group being heterogeneous and feature L concerns the group being symmetrical. As these are points that depend on the team`s composition, these items must be observed in a pre-performing step where the students are forming the teams.

Data Collection and Analysis

Data gathering and analysis were done through the collection of artifacts produced by the learners and the audio-visual content of the implementation stages. These procedures are described below.

Produced Artifacts

The artifacts produced during the sessions were collected and analyzed. Among the artifacts, we highlight the documents produced in the meetings, the task board (Kanban) with the defined tasks to solve a problem and the proposed solutions (algorithms/source code). Each artifact will be analyzed according to the following procedures:

- Kanban: at each session, the resulting task board was photographed and analyzed in order to verify how students are using the Computational Thinking decomposition strategy.
- Meeting Minutes: The documents resulting from the reflection meetings will be collected. Thus, through these artifacts, we analyze and verify whether the group is managing itself in order to reach its goals and its relation to success/failure in the proposed activities.
- Proposed solutions: the algorithms produced by the students will be collected and analyzed. We first analyzed if the products developed by the teams correctly included key concepts of the Algorithms Design dimension of Computational Thinking delineated for the workshop. Generally, computer scientists measure the efficiency of an algorithm in terms of runtime or used space (memory). By means of Algorithm Analysis techniques, we can count the number of operations, considered relevant, performed by the algorithm and express it in number. In summary, we verify if the algorithm uses more steps than it would really be necessary to reach the problem's resolution or to satisfy the proposed strategy. It should be noted that we also observe if there are limitations of the algorithm when achieving the proposed task requirement.

So, we need to investigate "if the implementation can always run according to what was proposed or are there 'bugs'?" This procedure consists in understanding if the strategies used by the students are really working.

Recording Sessions

The implementation stages (Pair Programming and Tests) were recorded as audiovisual content with the purpose of studying, more closely, the interaction and adopted strategies for the solution proposed by the learners. To examine the recordings, we adopted the episode analysis approach described by Goos and Galbraith (1996) used by the authors to investigate metacognitive strategies in collaborative mathematical problem solving. According to these authors, this approach "aims to highlight major strategic decisions, suggest when they should have been made (if absent) and assess the quality of the decisions per se". The students' verbal interactions (verbal protocols) were parsed into macroscopic episodes which represent a session (periods of time) where learners are engaged in distinctive types of problem-solving behavior. Therefore, these episodes were classified in Reading, Analysis, Exploration, Planning, Implementation and Verification, according to Schoenfeld (1985) apud Goos e Galbraith (1996).

Each student's complete speech turn was named Move. To analyze the interpersonal strategies and contributions of learners working collaboratively, Goos and Galbraith (1996) propose a kind of classification:

1. New information points were subdivided into two types:
    - points where previously overlooked or unrecognized information came to light (abbreviated as NI)
    - points where the possibility of using a new procedure was mentioned (abbreviated as NP).

    The NI/NP's were classified further according to:

    - who initiated the NI/NP
    - how relevant the NI/NP was to the task
    - the nature of the response to the NI/NP (ignore, reject, accept)
    - how appropriate the response was in context.
2. Local Assessments (LA's) of a particular aspect of a solution were classified according to who made the assessment, and the function of the assessment:
    - knowledge (assessing what is known/not known)
    - task difficulty
    - procedure (checking accuracy of execution, assessing relevance or usefulness)
    - result (assessing accuracy or reasonableness).
3. Global Assessments (GA's) of the general state of the solution were also made.

(GOOS and GALBRAITH, 1996, p. 242)

Due to the nature of Computational Thinking activities having some peculiarities in relation to generic mathematical problem-solving activities, it was necessary to make some adaptations in the procedure. These adaptations were:

- The episodes classified as Reading and Analysis are practically non-existent since the students are guided by the goal and functionalities described in the

Kanban board. Therefore, it is not necessary that students constantly read and analyze the problem once the task has been defined.

- The procedures (strategies adopted by students) reported by Goos and Galbraith (1996), in general, are related to the proposal of mathematical equations, assignments of values to variables of these formulas and calculation. However, due to the essence of the algorithm design, learners' strategies are more related to the proposal of new instructions (blocks in visual programming), logical flow (new combination of blocks), the addition of objects, variables manipulation among others.
- Not only verbal protocols were considered. The nature of the game development and programming activity must be performed through technological artifacts, the interaction with such artifacts can also be seen as a means of communication. In particular, we use a development environment that works with visual programming - through blocks. In this way, gestures and students' actions in the programming environment were also considered. For example, when a student points to a visual block that hasn't yet been used in the algorithm, we consider that he/she is proposing a new procedure to add to the solution.

Activities' Context

The workshops were applied in two public schools and involved students from the final three years of elementary school. One of the schools was located at the countryside of a small town. The workshop started with 4 groups. Due to the availability of the students and other situations described in the next section, these groups changed until the end of the workshop. The students participated voluntarily, and their legal guardian should have signed an Informed Consent Form. A total of three sessions were held and they lasted for about one hour per week, for each team. The groups also went through a session in which they dealt with the early aspects of game design and development. Therefore, a total of 11 workshop hours were performed.

Because of student's educational level and the number of sessions performed with each group, it was expected that teams that reached the end of the workshop would be able to manipulate, in a partially accurate way, the basic concepts of algorithms involving sequential flow, user input events, conditional commands and repeating (simple loops). As a development environment, we chose Scratch (MIT MEDIA LAB, 2019) because of its popularity among children. Developing a game as complete, even simple product, may require years of experience and several hours of development. In this way, the objective was to obtain an interactive prototype and not a conclusive product. We, therefore, expect the partial fulfillment of the game's requirements. The distribution of the team members and the sessions followed the Table 2 scheme. It's important to observe that the names of the students are fictitious in order to preserve their privacy.

Table 2: Sessions and groups composition.

Session	Group	Group composition	Planning	Algorithm contents to be mobilized.
1	Group 1	Daiane, José Pedro and Hector.	Racing game.	Sequential Flow and user input events.

	Group 2	Maria and Kamila.	Racing game.	Sequential Flow and user input events.
	Group 3	Humberto and Carlos.	Racing game.	Sequential Flow and user input events.
	Group 4	Anália, Aline, Bruna and Yuri.	"Shooting" game (definition only).	Sequential Flow, user input events, conditional structure and repetition structure.
2	Group 2	Maria and Humberto.	Same as the previous session.	Same as the previous session.
	Group 4	Anália, Aline, Bruna and Yuri.	Same as the previous session.	Same as the previous session.
3	Group 2	Maria, Kamila and Humberto	"Shooting" game.	Same as the previous session.
	Group 4	Anália, Aline, Bruna and Yuri.	Same as the previous session.	Same as the previous session.

One of the authors acted as a teacher and also performed the data collection. The teacher pointed out that the "shooting game" (which proved to be a very popular choice between students) shouldn't make references to firearms. Teams that opted for this option should use "toy ball gun" or launch something like ninja stars. The teacher also explained his expectations to the students, as it is understood that developing games is a very complex task, the students were expected to produce incomplete artifacts with some "bugs".

Obstacles in the Strategy Implementation

It should be noted that we face many barriers to perform the workshop in schools. Here are some of these difficulties:

- Maintain symmetry between group's members - Symmetry means members having approximately the same level and can do the same actions (Dillenbourg, 2007, p.7). This feature (item L of Table 1) can't be observed in some moments. This happened due to a number of factors described below:
  - students who volunteered had very different backgrounds;
  - some students (four) declared that they did not have access to a computer at home - which makes the extra class practice harder;
  - students had limited scheduling;
  - some students missed the sessions forcing some members redistributions;
  - students wanted to do the workshop with others with whom they had a greater affinity.
- keep workshop sessions often - school routine at times requires students to travel to participate in sporting events. In addition, there was an occasion when the computer lab key wasn't found by school professionals. The lack of the sessions' frequency may harm students who already have difficulty assimilating the algorithms concepts.

- Maintaining Quality of Audio Recording - The laboratory provided by one of the schools was sometimes shared with other teachers and students who performed other activities. The other school's laboratory was next to kindergarten classes. The noise caused by such situations has hampered the quality of audio recording sometimes.

Results

This section presents the results achieved by the analysis of the collected data.

Decomposition Skills

Decomposition was a step that required more teacher mentoring than expected. At first, the students associated the game development only with the visual elements of it. They didn't observe the dynamics of the game involving, for example, responses to user actions. And experience has shown that there is the possibility of teacher solving problems through storytelling approaches or constructing narratives through visual schemas. Through this, the teacher can intervene by asking students to "take a narrative of the game to identify the dynamics of the game and turn them into tasks. These techniques will be incorporated into the strategy proposed in future experiments. The result of task decomposition is represented in the kanban produced. In general, the decomposition took place satisfactorily, carrying out the implementation tasks. There were few sessions where kanban no longer included a necessary sub-task.

Problem Solving Interactions

From the students' moves, according to Goos and Galbraith (1996) approach, a table was arranged highlighting proposals for new procedures and information (NI/NP's). In this way, the table was composed of 6 columns: (1) move identification - composed by the initial student 's name plus the sequential number; (2) NI/NP - if the Move is an NI/NP, mark a V if it is useful for solution and X otherwise; (3) Episode type - represents facets of problem solving; (4) context - description of the learner's act; (5) Response to NI/NP - description of the answers given by the other members in the case of an NI/NP; and (6) Adequate response - mark a V if the answer is suitable for solution and X otherwise. In order to visualize the interactions, we set up a graph from this table. The vertices represent the moves or NI/NPs. When a student proposes an NI/NP, an edge is directed from the vertice of its move to the NI/NP vertice. The NI/NP vertice is labeled with an X or V depending on its utility for the solution. Responses are represented by directed edges from the NI/NP for the responses moves. Edges were labeled according to students acts.

Graph analysis showed that when there is symmetry, everyone contributes in a more harmonious way to the group. When there isn't, the trend is that there are more contributions with new ideas and procedures, without any discussion around it, coming from the student who has greater ease. However, peer exchange allows even students, who presented difficulties to participate in the solution by implementing the suggested procedures. Although the implementation is often done passively, the student still directly participates in the solution. Figure 4 illustrates two parts of interaction graphs extracted from different sessions. In graph A there is the relationship of symmetry between Maria (M) and Kamila (K), graph (B) illustrates the activity sequence in another session but does not occur the symmetry relationship between Maria and Humberto (H). This situation may have occurred due to several factors, but it may be related to the fact that Humberto has access to computer at home while Maria and Kamila don't.

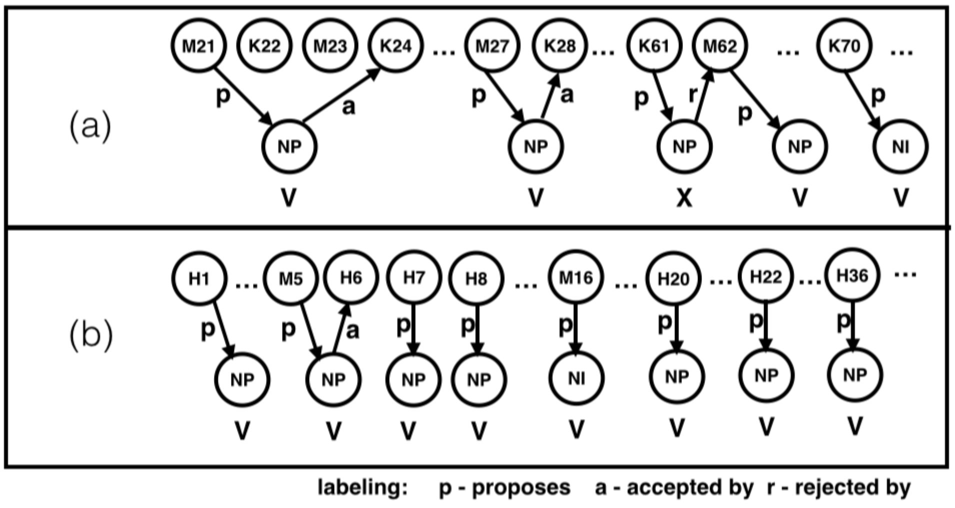


Figure 4: Part of the interaction graph developed by group 2, in the first session (a) and second session (b).

Student's Solving Strategy

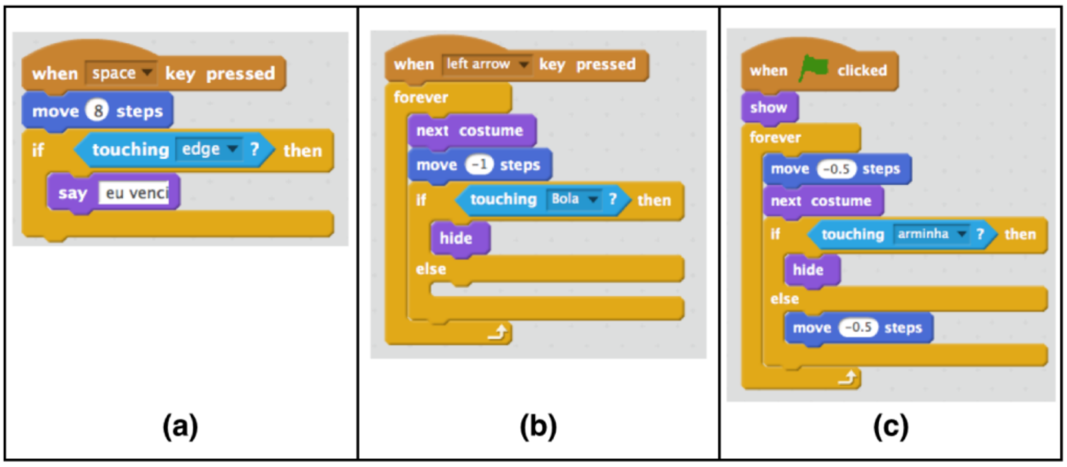
From the student moves analysis, we observed many occurrences of verification episodes that were presented as tests. This happened when new procedures were included or even to experience aesthetic elements of the game. Thus, there is evidence that students have demonstrated a problem-solving approach similar to the Test-Driven Development (TDD) technique. According to Teles (2004), it is a technique in which software developers conduct tests to obtain feedback quickly on what they are doing. Obviously, learners were expected to take tests in stage 5. However, the students took the tests "voluntarily" to examine each new procedure, without the need for teacher interventions. It's possible that students have adopted this method because of the main activity division into sub-tasks performed in step (2) and the use of game elements that are easily verifiable through animations and interactions.

Mobilized Algorithm Design Skills

Sequential Flow and User Event Concepts Mobilization: It is a concept that represents a starting point for the algorithm design. All groups were able to use these concepts correctly. However, students sometimes forgot the event responsible for starting the game.

Conditional Mobilization: Groups 1 and 3 were able to properly mobilize this concept during the first session involving programming to produce a racing game. Figure 5(a) illustrates a racing game algorithm. Group 2 was only able to mobilize this concept in the second session. This may have been due to the fact that the students in the first session did not have access to a computer at home. Group 4, during the first session defined that it would make a "shooting game" - the racing game could give an opportunity to use the concept in a simpler way. In Figure 5 (b), the students chose to use the "if-else" structure and the "else" was unnecessary. This team also used the "if" structure unnecessarily inside a loop, which, in addition to consuming processing time, a "bug" in the game happened. In addition, there was a situation where the team' students were unable to use such structure correctly. Group 2 also mobilized the "if" structure to develop the "shooting game". To make the player's character disappear when captured, an unnecessary "else" command was used, but the group also used the "if" structure correctly to make a character disappear when being hit - Figure 5(c).

Repetition Concept Mobilization: It was expected that the repetition command would be used at least in two situations. Both group 2 and 4 used the repetition correctly to make the rival character walk the stage. Group 2 missed this command in one of the citations. Group 4 used a second repeat structure partially correctly - in account of the misuse of a loop's "if" command (as mentioned earlier).



(a)
```
when space key pressed
move 8 steps
if touching edge ? then
    say eu venci
```

(b)
```
when left arrow key pressed
forever
    next costume
    move -1 steps
    if touching Bola ? then
        hide
    else
```

(c)
```
when clicked
show
forever
    move -0.5 steps
    next costume
    if touching arminha ? then
        hide
    else
        move -0.5 steps
```

Figure 5: Some algorithms designed by students during the workshop.

The following faults in the requirements, bugs and unnecessary steps in the algorithm were identified:

- Groups 1, 2 and 3 who proposed the simple racing game met all the requirements and were clear of failures.
- The shooting game proposed by group 2 failed to meet 1 requirement. There were 3 "bugs" in the game and 2 unnecessary steps.
- The shooting game proposed by group 3 left no longer fulfill 1 requirement. There were 3 "bugs" in the game and only 1 unnecessary step.

Reflection on Social Skills

The reflection meeting (step 6) demonstrated to be very useful in dealing with social aspects. Some examples recorded during the meetings are described subsequently. Group 4 during the second session reported that girls needed to stop laughing (making fun of each other) during the development period. This attitude could take the focus off programming from those who were attentive. In the following session, students found that "jokes" occurred less frequently. Another situation was reflected in group 1 meeting minutes. During the first session, there was a moment that the pilot was having problems, so one of the students interfered assuming the computer (for a short time) to perform a task. In the reflection meeting, they recognized that they needed to improve the coexistence in the group.

Playful Games

During the workshop application, we could observe that the students showed themselves interested. At many times during the tests, the learners demonstrated expressions that indicated that they were having fun (with laughs for example). In the reflection meeting's minutes, the students reported constantly that the games were getting fun.

Conclusion

This article presented a cooperative learning strategy with the aim of promoting Computational Thinking skills in students through digital games development. The results showed that the strategy is able to mobilize social skills and problem-solving strategies related to CT in a motivating way. The analysis of the interactions among the group members showed that, when there is symmetry between the participants, there is higher participation of the group members in the problem-solving process. On the other hand, we faced several situations where symmetry can't be maintained. Even in these cases, peer programming allowed learners who were contributing less to participate in the solution once they were being guided by another student. Souza, Marczak and Prikladnicki (2011) explain that in software development, pair programming usually consists of the most experienced programmer assisting the less experienced programmer who will be coding. The analysis of the episodes showed that the students were adopting, together, TDD techniques. Teles (2004) reports that when developers leave the tests to the end, the project ends up being harmed. In addition, the author reports that developers can learn more by this approach. Results also showed that students mobilized decomposition skills and algorithm design concepts. Although the artifacts developed presented some mistakes, each of the flow control structures proposed in the workshop was used at least once in a completely correct manner. The reflection meeting showed that students are able to think about their group interactions behavior.

Finally, the strategy was motivating to show evidence that the students were having fun and thinking that the games were getting funny. We believe this is a considerable aspect since it is important for the students to see meaning in what they are doing and learning.

References

AQUINO FILHO, G. F.; AMARAL, L. H.; SCHIMIGUEL, J. Ambientes colaborativos para ensino de eletrônica e lógica de programação. **Revista de Ensino de Ciências e Matemática (REnCiMa)** , v. 7, p. 31-39, 2016.

BARCELOS, T. S. **Relações entre o pensamento computacional e a matemática em atividades didáticas de construção de jogos digitais**. 2014. 276 p. Tese (Doutorado em Ensino de Ciências e Matemática) - Universidade Cruzeiro do Sul. São Paulo. 2014.

BARR, Valerie; STEPHENSON, Chris. Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. **Acm Inroads**, v. 2, n. 1, p. 48-54, 2011.

BRACKMANN, Christian Puhlmann. **Desenvolvimento do pensamento computacional através de atividades desplugadas na educação básica**. 2017. 226 p. Tese (Doutorado em Informática na Educação) - Universidade Federal do Rio Grande do Sul. Porto Alegre. 2017.

DILLENBOURG P. What do you Mean by Collaborative Learning? In: DILLENBOURG P. **Collaborative-learning**: Cognitive and Computational Approaches. Oxford: Elsevier, 1999. cap.1, p. 1-19.

FEIJÓ, B.; SILVA, F. S. C. da; CLUA, E. **Introdução à Ciência da Computação com Jogos**: aprendendo a programar com entretenimento. Rio de Janeiro: Elsevier, 2010. 263p.

FRONZA, I.; IOINI, N.; CORRAL, L. Teaching Computational Thinking Using Agile Software Engineering Methods: A Framework for Middle Schools. **ACM Transactions on Computing Education (TOCE)**, v. 17(4), p.1-28, 2017.

GOOS, M.; GALBRAITH, P. Do it this way! Metacognitive strategies in collaborative mathematical problem solving. **Educational Studies in Mathematics** v. 30, p. 229–260, 1996.

JESUS, Â. M.; SILVEIRA, I. F. A Collaborative Game-Based Learning Framework to Improve Computational Thinking Skills. **Transactions on Edutainment**. 2019. In press.

JOHNSON D. W.; JOHNSON, R. T.; HOLUBEE, E. J.; ROY P. Why Cooperative Learning is Important. In: JOHNSON et al. **Circles of Learning: Cooperation in the Classroom**. S. l.: Eric, 1984. cap.1 p. 7-17.

LEE, I. et al. Computational thinking for youth in practice. **ACM Inroads**, v. 2, n. 1, p. 32-37, fev. 2011.

MATTHEWS, R.S.; COOPER, J.L.; DAVIDSON, N.; HAWKES, P. Building Bridges between Cooperative and Collaborative Learning. **Change**, S. l., v. 27, p. 34-40, 1995.

MIT MEDIA LAB. **Scratch**: create, imagine, share. Available in <https://scratch.mit.edu/> Accessed on: 02 Jan. 2019.

MOREIRA, M. A. Aprendizagem Significativa: um conceito subjacente (Meaningful learning: an underlying concept). **Aprendizagem Significativa em Revista/Meaningful Learning Review**, Porto Alegre, v.1, n.3, p. 25-46, 2011.

PANITIZ, T. Collaborative Versus Cooperative Learning: a comparison of the two concepts which help us understand the underlying nature of interactive learning. 1999. 13p. **Educational Resources Information Center ERIC**. Available in <https://files.eric.ed.gov/fulltext/ED448443.pdf> Access on: April 2, 2018.

PAPERT, Seymour. **A Máquina das Crianças**: Repensando a Escola na Era da Informática. Porto Alegre: Artmed Editora, 1993. p. 220.

PAPERT, S.; SOLOMON, C. Twenty things to to with a Computer. Educational **Technology Magazine**, 1972. Available in: <http://www.stager.org/articles/twentythings.pdf>. Accessed on: 05 Jan. 2019.

RAU, W.; HEYL, B. S. Humanizing the College Classroom: Collaborative Learning and Social Organization Among Students. **Teaching Sociology**, S. l. v. 18, p. 141-155, 1990.

SOUZA, C. R. B. de; MARCZAK, S.; PRIKLADNICKI, R. Desenvolvimento colaborativo de software. In: PIMENTAL, M. and FUKS, H. (Orgs.). **Sistemas Colaborativos**. 1.ed. Rio de Janeiro: Elsevier, 2011. cap.8, p.122-134.

TABEL, O.; JENSEN, J.; DYBDAL, M.; BJØRN, P. Coding as a Social and Tangible Activity. **Interactions**, v.24(6), p.70-73, 2017.

TELES, V. M. **Extreme Programming**: Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade. São Paulo: Novatec Editora, 2004.

TORRES, P. L.; IRALA, Esrom Adriano Freitas. Aprendizagem Colaborativa: Teoria e Prática. In: TORRES, Patricia Lupion. (Org.). **Complexidade: Redes e Conexões na Produção do Conhecimento**. 1ªed. Curitiba: SENARPR, 2014, v. 1, p. 61-93.

WEBBER, C. G.; VIEIRA, M. B. Tecnologias digitais na educação: colaboração e criatividade em sala de aula. **Revista de Ensino de Ciências e Matemática (REnCiMa)**. v. 1, n. 2, p. 166-177, 2010.

WING, J. M. Computational thinking. **Communications of the ACM**, v. 49, n. 3, p. 33-35, mar. 2006.