*International Journal of Computer Science Education in Schools, May 2017, Vol. 1, No. 2*
*ISSN 2513-8359*

# Teaching Computing in a Multidisciplinary Way in Social Studies Classes in School – A Case Study

**Christiane Gresse von Wangenheim**[1]
**Nathalia Cruz Alves**[1]
**Pedro Eurico Rodrigues** [2]
**Jean Carlo Hauck** [1]
[1]Federal University of Santa Catarina
[2]University of São Paulo

DOI: 10.21585/ijcses.v1i2.9

**Abstract**

In order to be well-educated citizens in the 21st century, children need to learn computing in school. However, implementing computing education in schools faces several practical problems, such as a lack of computing teachers and time in an already overloaded curriculum. A solution may be a multidisciplinary approach, integrating computing education within other subjects in the curriculum. The present study proposes an instructional unit for computing education in social studies classes, with students learning basic computing concepts, including computational thinking, by programming history related games using Scratch. The instructional unit is developed following an instructional design approach and is applied and evaluated through a pilot case study in four classes (5th and 7th grade) with a total of 105 students at a school in (omitted for submission). Results provide a first indication that the instructional unit enables the learning of basic computing concepts (specifically programming) in an efficient, effective and entertaining way increasing also the interest and motivation of students to learn computing.

**Keywords:** computer science, social studies teaching, K-12, scratch, programming, computational thinking

## 1. Introduction

### *1.1 Why is computing important?*

Computing is becoming increasingly ubiquitous in our lives. Knowing fundamental concepts of computing, beyond the simple use of Information Technology (IT), enables people to be productive regardless of their professional area (Seehorn et al., 2011). Therefore, there is a growing consensus that it is important to provide opportunities for children to learn computing starting in elementary school (Naughton, 2012). And, it is no longer enough to only focus on teaching IT literacy, the capability to use today's technology (Lin, 2002). Students have to acquire IT fluency, which adds the capability to independently learn and use new technology as it evolves, including the active use of computing (Seehorn et al., 2011). Therefore, students need to learn computational thinking (Wing, 2006) an approach to problem solving in a way that can be implemented on a computer involving a set of concepts, such as abstraction, recursion, iteration, etc. as well as computing practice including programming and the use of software tools to solve problems.

### *1.2 Teaching computing in schools*

Currently, the introduction of computing education in schools is a global trend supported by several initiatives such as Code.org or Computing at School among others. These initiatives support computing education by providing age appropriate programming environments such as Scratch (MIT, 2016) or Snap! (2016)), curriculum guidelines, lesson plans and materials (Seehorn et al., 2011) as well as workshops, courses etc. (CodeClubBrasil, 2016). However, most of these initiatives focus on teaching computing as a stand-alone subject (Pazinato and Teixeira 2013) (Wilson and Moffat, 2010) (Aureliano and Tedesco, 2012). This approach may be problematic in practice, as it may be difficult to find time for teaching another subject in an already overloaded school curriculum (MEC, 1998) as to find well-trained teachers for computing education in schools (Google & Gallup, 2015).

### *1.3 Teaching computing in a multidisciplinary way*

A solution may be the integration of computing education in a multidisciplinary way within existing subjects in the curriculum (Qualls and Sherrell, 2010). There exist several proposals of teaching computing in related subjects such as physics, mathematics, etc. (Andrade, 2013) (Pinto, 2010). However, in order to obtain a broader acceptance and also to attract girls that may have a preference for different subjects, an alternative may be the integration in social studies classes, having students study a history topic by learning computing competencies at the same time (Code.org, 2013) (Ncwit, 1x). Yet, so far there exist only very few work focusing on integrating the teaching of computing in different knowledge areas. Table 1 presents an overview on related work as identified through a systematic literature review (Alves, 2016).

Table 1. Summary on related work

| Title/Reference | Objective | School year | Integration with | Programming language |
|---|---|---|---|---|
| Embedding Scratch in US History/Geography (Scratched; Randall, 2009) | Program an interactive animation on a theme related to social studies. | 5th | History and geografy | Scratch |
| Using App Inventor & History as a Gateway to Engage African American Students in Computer Science (Jimenez and Gardner-Mccune, 2015) | Use aspects of computational thinking aligned with historical thinking to introduce students to computer science within History classes. | Not informed | History | App Inventor |
| Animal tlatoque: attracting middle school students to computing through culturally-relevant themes (Franklin et al., 2011) | Use Scratch to engage students in creating animations about animals and Mayan culture, creating an interdisciplinary experience that combines programming, culture, biology, art, and storytelling. | 7th and higher | Biology, art and History | Scratch |

Thus, although, there exist already some work in this direction, the results are not readily applicable in our context in Brazilian schools, as the History theme addressed in the existing instructional units is culturally relevant only to where the studies were conducted, as well as the fact that the units are not available in Brazilian Portuguese. In this respect, this article presents an instructional unit (IU) for the development of a digital game dealing with a history topic as part of social studies classes.

## 2. Method

The objective of this research is the development, application and evaluation of an instructional unit for computing education in a multidisciplinary way in schools. To achieve this goal, an exploratory case study was conducted to understand the phenomena observed during the application of the IU in a particular context and identify directions for future work (Fig. 1).

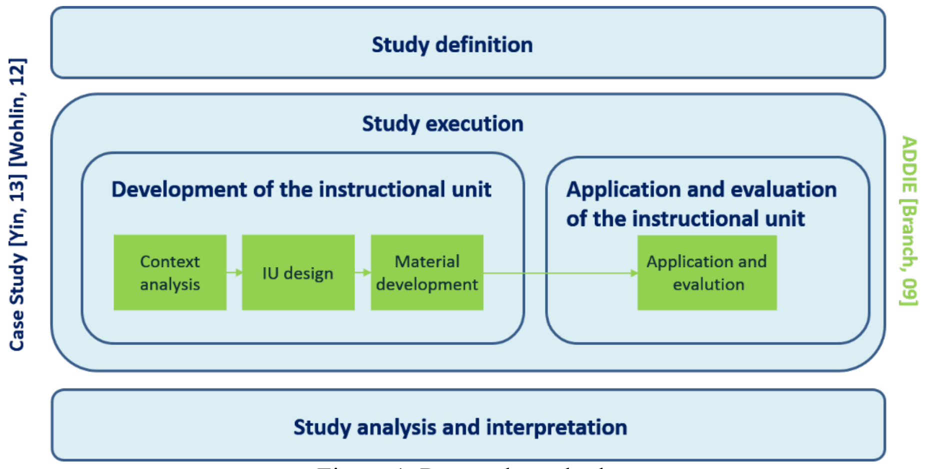
Case Study [Yin, 13] [Wohlin, 12]
Study definition
Study execution
Development of the instructional unit
Context analysis
IU design
Material development
Application and evaluation of the instructional unit
Application and evalution
ADDIE [Branch, 09]
Study analysis and interpretation

Figure 1. Research method.

The case study is performed according to the procedure proposed by Yin (2013) and Wohlin et. al (2012).

**Study definition.** The study is defined in terms of objective, research questions and research design. From the objective, analysis questions and measures are systematically derived using the Goal/Question/Metric (GQM) approach (Basili et al., 1994). Data collection instruments are defined with respect to the measures.

**Study execution.** The execution of the study is carried out adopting ADDIE (Branch, 2009) as instructional design approach. First, the instructional unit is developed. Therefore, the learners and the instructional environment are characterized. Learning needs are elicited and the learning objectives are defined. In accordance to the context, the instructional strategy is designed, defining its content, sequence and instructional methods to be adopted. Instructional material is developed in accordance to the instructional strategy. Then, the instructional unit is applied in the classroom and evaluated, collecting data as defined by the study definition.

**Analysis and interpretation.** The collected data is analyzed in relation to the research questions, using quantitative and qualitative methods. Then, the results are interpreted and discussed.

This research was approved by the Ethics Committee (omitted for submsission) (No. 1021541).

## 3. Multidisciplinary Instructional Unit UNIfICA

### *3.1 Definition of the IU*

In alignment with the ACM/CSTA K-12 curriculum guidelines (Seehorn et al., 2011), the purpose of the instructional unit UNIfICA[1] is to teach basic computing concepts by creating a game with Scratch, related to a History topic. The main focus is on teaching programming concepts (loops, conditionals, event handling, etc.), the application of the software engineering cycle and collaborative practices. Students should also understand what algorithms are and how algorithms and problem solving work. Regarding the Scratch environment, students should be able to describe what can be done with the environment as well as being able to use the environment to create, play and share a game. With respect to the History subject (MEC, 1998), the instructional unit should help to reinforce the understanding of the reality in multiple temporal dimensions focusing specifically on regional cultural issues in (omitted for submission) at 5th grade or global topics in ancient civilizations (Europe, Greece and Ancient Rome) at 7th grade. Students should also understand the differences between cultures and ways of living/thinking/doing.

The target audience is school students, aged 8 to 14 years, who already know how to use computers, but do not have computing competencies. The unit is expected to be taught by a History teacher (with a basic computing knowledge) together with a teacher providing IT support in schools.

### *3.2 Instructional strategy*

#### *3.2.1 Scratch*

The instructional unit uses Scratch (MIT, 2016), a free block-based visual programming language and online community developed at the MIT Media Lab. It is inspired by programming languages for young people like LOGO and Squeak Etoys (Resnick, 2007). Despite being based on languages aimed at children and young people, Scratch was designed to be different from other environments, to be simpler, easier to use and more intuitive (Guzdial, 2004). It allows to program interactive stories, games, and animations by simply using drag and drop blocks to perform different commands or actions (Fig. 2) (Malan and Leitner 2007) (Monroy-Hernández and Resnick, 2008). It is one of the most popular programming languages for young people with more than 16 million users worldwide. The primary goal of Scratch is to help children (ages 8 and up) to develop essential 21st century learning competencies (Rusk, Resnick and Maloney, 2006).

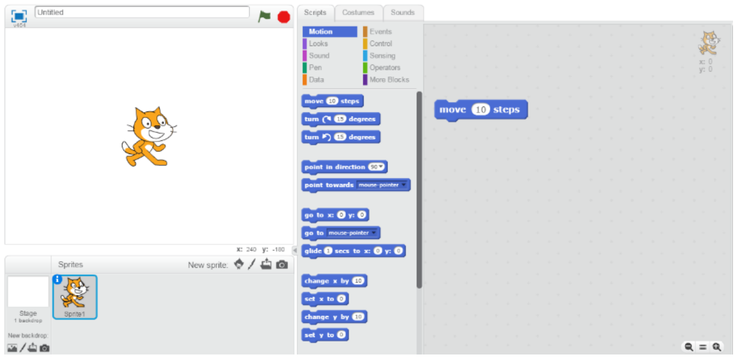

Figure 2. Scratch programming environment.

### *3.2.2 Design of the instructional unit*

During the instructional unit, first basic programming concepts and the Scratch environment are presented. The instructor teaches the development of an exemplar puzzle game in an active learning approach in which the concepts are presented step-by-step and the students immediately apply them by creating the game. After the initial familiarization, students begin creating their own game related to a History topic. These topics are assumed to have been studied beforehand in the social studies class through expositive lectures, reading and discussions. In order to give ideas, several examples of Scratch games related to History topics are demonstrated.

The students then develop an idea for a game related to the content of the class, choosing the kind of game and its mechanics. Then, they iteratively and incrementally design, program and test the game in pairs or small groups with up to three students. The created games are shared in a studio via the Scratch online community. The students are encouraged to play and comment on the games. In the end of the instructional unit, the experiences are discussed in class.

Table 2: Syllabus of the IU

| **Lesson (2 hours each)** | **Instructional method** | **Resources** | **Evaluation** |
|---|---|---|---|
| Measurement 1 | | - Student pre-unit questionnaire | |
| **1**. Introducing Scratch<br>- Access the Scratch environment<br>- Program an example game illustrating basic commands | - Practical activity following the step-by-step presentation of the instructor | - Instructor guide<br>- Scratch environment<br>- Computers<br>- Classroom projector | |
| **2**. Designing a game on a history topic<br>- Division of students in groups.<br>- Design of their game (selection of game type & mechanics and history topic). | - Presentation of example games by the instructor<br>- Practical activity in small groups/pairs. | - Example games<br>- Scratch environment<br>- Instructional material from history classes<br>- Computers<br>- Classroom projector | |
| **3**…**5**. Programming and testing the game | - Practical activity in small groups/pairs.<br>- Individual support provided by instructor(s) answering questions of the students. | - Example games<br>- Scratch environment<br>- Instructional material from history classes<br>- Computers | - Assessment rubric for evaluating the student's programs<br>- Dr. Scratch for analyzing complexity of the programs |
| **6**. Finalization of the projects<br>- Sharing and trying out the games developed by the class.<br>- Debriefing on the instructional unit. | | - Practical activity in small groups/pairs.<br>- Discussion | - Scratch environment<br>- Computers |
| Measureme[1]nt 2 | | | - Student post-unit questionnaire<br>- Teacher post-unit questionnaire<br>- Parent, teacher and student feedback collection |

In accordance to the instructional strategy, the instructional materials have been developed and are available online in Brazilian Portuguese under the Creative Commons license (omitted for submission).

[1] Instructional unit developed in Portuguese: "Unidade Instrucional Interdisciplinar de Computação e História".

### *3.2.3 Dr. Scratch*

To assess the complexity of the games developed by the students, the static code analysis tool Dr. Scratch (Moreno-León and Robles, 2015) is used. Dr. Scratch is a free open-source web tool to automatically analyze Scratch projects that assigns a score in terms of computational thinking and programming concepts used. The analyzed areas include: Logic, Parallelism, User Interaction, Data Representation, Flow Control, Synchronization, and Abstraction. Table 3 illustrates how the score is defined with respect to each of these areas.

Table 3: Areas analyzed by Dr.Scratch.

| **Area** | **Score definition** |
|---|---|
| Logic | Use of "if then else" commands. |
| Parallelism | Use of 2 or more commands that execute at the same time. |
| User Interaction | Use of commands that do some kind of interaction with the user, e.g.: mouse use, keystroke. |
| Data Representation | Use of commands that modify actor, variable, and lists properties. |
| Flow Control | Use of “for”, “while” commands. |
| Synchronization | Use of commands that wait or send messages to synchronize actions. |
| Abstraction | Use of commands to create functions and use of sprites in an advanced way. |

## 4. Application of the IU

The developed instructional unit was applied in two 5th grade classes (5M and 5V) and two 7th grade classes (7A and 7B) at the school (omitted for submission) during the first semester 2015 (Table 4). (Omitted for submission) is a private school that offers early childhood education, primary and secondary education.

Table 4: Overview on the application.

| **Grade** | **Class** | **Number of students** | **Average number of computing instructors present to support the application of the IU (in addition to the subject teacher)** |
|---|---|---|---|
| 5th | 5M | 24 | 4 |
| | 5V | 21 | 1 |
| 7th | 7A | 31 | 3 |
| | 7B | 29 | 1 |
| Total | | 105 | |

The classes were taught by the history teacher of the (omitted for submission). As the objective of the application was to pilot the developed instructional unit, the classes were also supported by computing instructors (professors and undergraduate students) from the (omitted for submission). The instructional unit was taught as part of the regular social studies classes. In total, six lessons were taught biweekly in accordance to the availability of each class/instructors/etc. The number of computing instructors (omitted for submission) varied among the classes depending on their availability (Tab. 4). The subject teacher and the computing instructors provided assistance during the development of the games by the students. Undergraduate students from (omitted for submission) supported the preparation of instructional materials and the management of backups of the students' work.

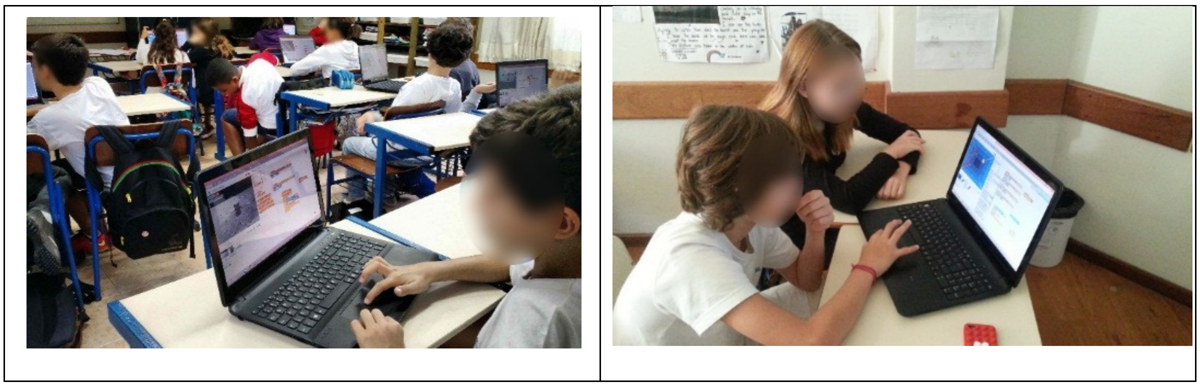
Figure 3. Students during the IU.

Since the school does not have a specific computer room, classes were conducted in the students' classrooms using notebooks. Due to some Internet connection problems, both Scratch versions (online and offline) have been used during the classes. Some examples of the results developed by the students are presented in Figure 4 and are available online[2].

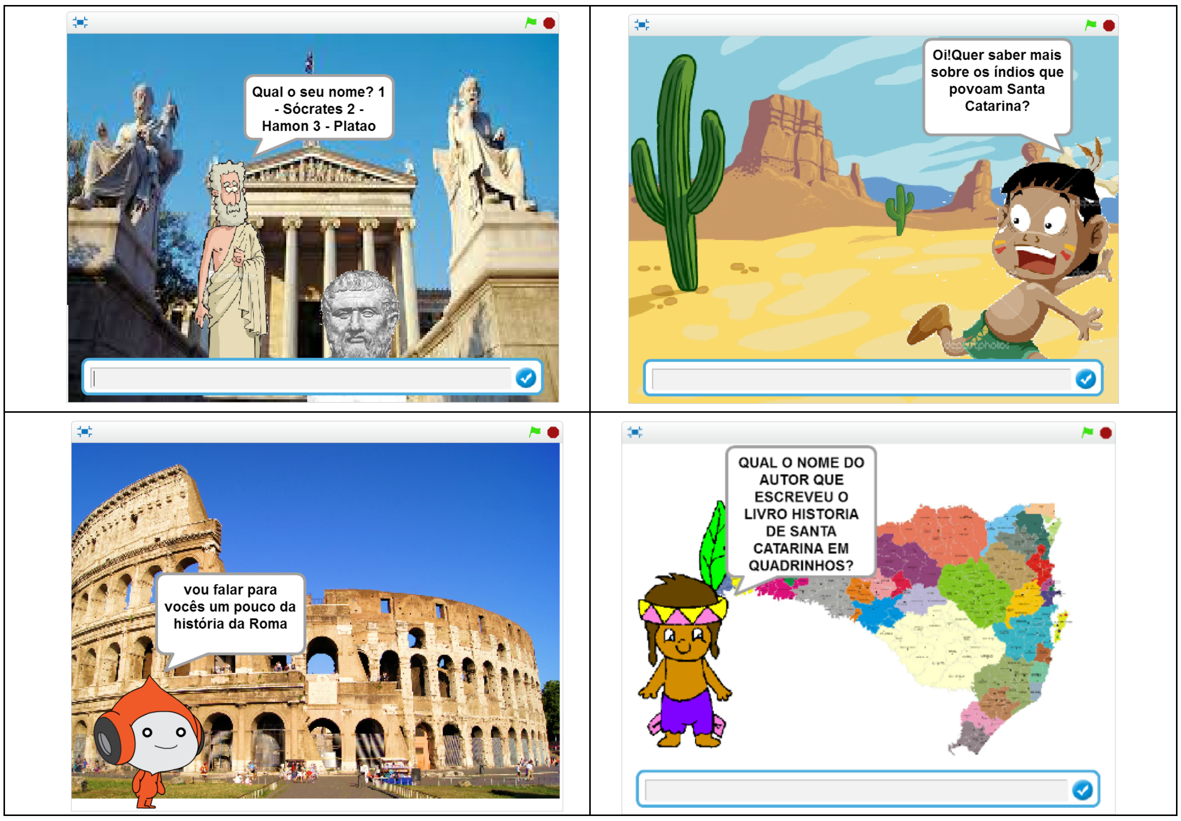
Qual o seu nome? 1 - Sócrates 2 - Hamon 3 - Platao
Oi!Quer saber mais sobre os índios que povoam Santa Catarina?
vou falar para vocês um pouco da história da Roma
QUAL O NOME DO AUTOR QUE ESCREVEU O LIVRO HISTORIA DE SANTA CATARINA EM QUADRINHOS?

Figure 4. Examples of games developed by the students.

## 5. Evaluation of the Instructional Unit

### *5.1 Definition of the evaluation*

The goal of this study is to explore and to understand aspects related to the instructional unit for teaching computing in schools, in a multidisciplinary way, in social studies classes. Based on this goal, the following analysis questions have been defined:

AQ1. Are the learning objectives (both in terms of computing and in terms of social studies) achieved using the instructional unit?

AQ2. Does the instructional unit facilitate learning?

AQ3. Does the instructional unit promote a pleasant and enjoyable learning experience?

AQ4. Does the instructional unit provide a positive perception of computing?

### *5.2 Data Collection*

Data has been collected from the students before and after the instructional unit via questionnaires and the teacher after the unit. Data on the complexity and commands used in the developed games has been analyzed using the tool Dr.Scratch (Section 3.2.3). In addition, observations from parents, students and teacher have been collected informally by the instructors.

[2] Class 5M: https://scratch.mit.edu/studios/1192816/
Class 5V: https://scratch.mit.edu/studios/1192820/
Class 7A: https://scratch.mit.edu/studios/1192828/
Class 7B: https://scratch.mit.edu/studios/1192830/

Table 5. Overview on the quantity of the data collected.

| **Grade** | **Student** | | **Teacher** | **Games** | **Observations from parents, students and teacher** |
|---|---|---|---|---|---|
| | **pre-unit questionnaire** | **post-unit questionnaire** | **post-unit questionnaire** | | |
| 5th | 45 | 45 | 1 | 18 | Yes |
| 7th | 60 | 54 | 1 | 23 | Yes |
| **Total** | **105** | **99** | **2** | **41** | **Yes** |

### *5.3 Data Analysis*

Data was analyzed in a qualitative and quantitative way using descriptive statistics with respect to the analysis questions. As no significant difference between the data collected in the different classes (neither related to grade nor history topic) was identified, it was analyzed by simple pooling without being weighted, creating one single sample.

In general, all students created a game, including various game genres as shown in Table 6. Most of the students actively participated throughout the computing lessons, demonstrating enthusiasm and willingness.

Table 6. Distribution of the games developed per genre.

| **Game genre** | **A game** | **Amount of games developed** |
|---|---|---|
| Action | that emphasize movements, usually based on reactions. | 13 |
| Quiz | where the player needs to answer questions to a particular knowledge area. | 12 |
| Adventure | where the player follows a story through texts/songs/images, e.g., puzzles. | 11 |
| Incomplete game or without genre | that does not have a clear genre or was not finished. | 5 |

#### *5.3.1 Are the learning objectives achieved using the instructional unit?*

All students were able to use the Scratch environment for programming a game with ease. The students used the initially presented commands and found the environment very intuitive to explore further commands, indicating that they understood and were able to apply programming concepts (Fig. 5).

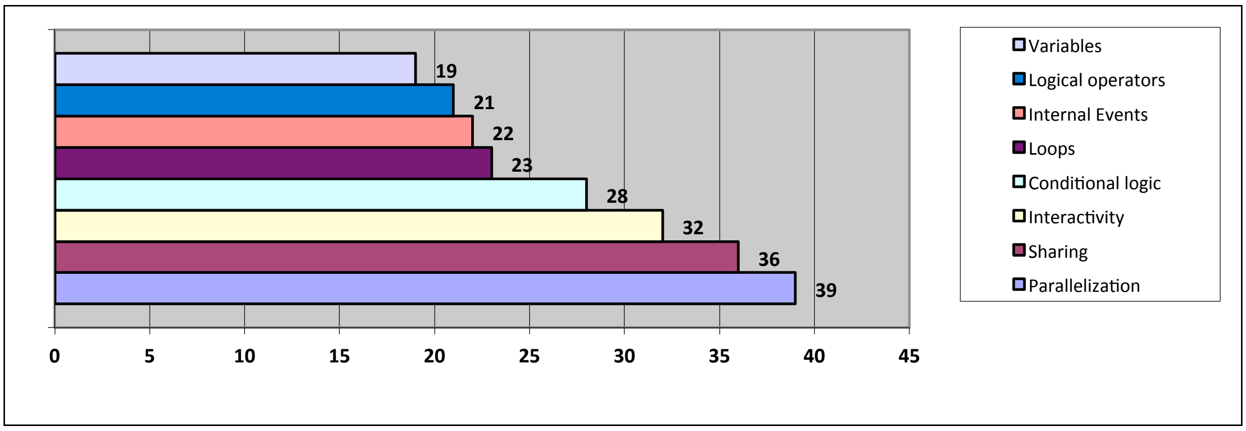

Figure 5. Frequency distribution of commands/resources used in the developed games

Many games (more than 68%) used basic commands such as conditional logic mostly to manipulate actor's appearances, backgrounds and game scores. Other commands widely used include loops, internal events and logical operators. Furthermore, a large number of games (more than 46%) also used variables mainly for controlling the game score. Among the most used ones also were interactivity commands in order to allow the players to interact with the game. In addition, a large number of games used commands for the parallel execution of scripts due to the games' mechanics and the intuitive way in which they are supported in the Scratch environment. This shows that students learned programming concepts varying from simple concepts such as logical operations to more advanced concepts such as synchronization.

A main strength of the Scratch environment is its flexibility, with the possibility to create any kind of game. It also provides diverse support for the game design. For example, to design characters or backgrounds, the students either used images from the Scratch image gallery or searched on the Internet (e.g. using Google image search). They also used the Scratch drawing tool to draw from scratch or to modify images. This allowed the students to use actors and scenarios in accordance to the respective History topic. The Scratch image gallery and drawing tool turned to be a great feature especially in moments without Internet connection, enabling the design of characters and backgrounds offline within the Scratch environment.

Indirectly by programming the games, students also learned how to use basic steps in algorithmic problem solving which contributed to the learning of computational thinking and to understand that software is a sequence of instructions being followed by a computer. Starting from an idea for their game they typically divided it into parts, and then designing, programming and testing each part immediately in the Scratch environment. Thus, implicitly following a cycle of problem statement, solution design, programming and testing, enabled them also to have an initial notion of a software engineering process for developing computer programs.

Working together on the game also helped the students to learn to develop software collaboratively. This has turned out to be a main strength of the IU as they helped each other by executing a kind of pair programming, with one student programming and the other accompanying, suggesting and reviewing the code being written. When facing a problem they were not able to resolve on their own, students asked for help of one of the instructors. It was clear that the possibility to freely choose both the game genre and the game design stimulated a discussion and contribution of ideas of almost all students within their groups. However, very few students were distracted by the possibility to access other websites and/or by problems with the notebook and/or instable Internet connection, not focusing on the game development. On the other hand, students were very eager to show the results they achieved to others, which motivated those to achieve similar results. We also observed that students willingly explained how they had done something implementing a kind of peer instruction.

The students themselves also perceived these learning effects. After the instructional unit, most students believed that they can make computer programs. However, only some of the students thought that they reached higher competence levels (e.g., being able to explain to a colleague how to make a program) (Fig. 6).

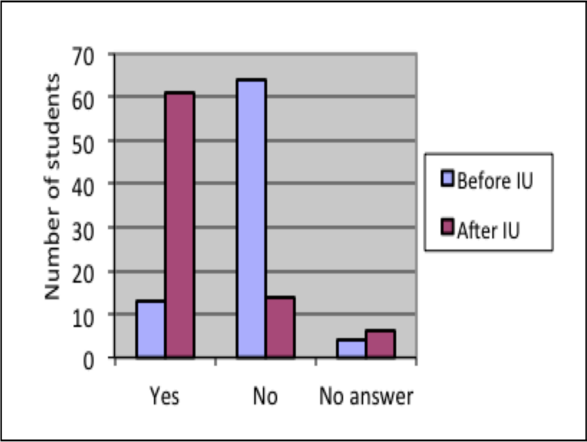

Figure 6a: “I can make computer programs" before and after the application of the IU.

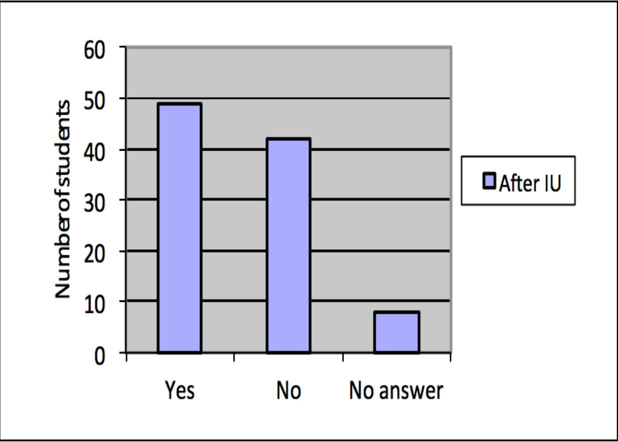

Figure 6b: "I can explain to a friend how to make a computer program" data after application of the IU.

Analysing the games with respect to their History content and based on feedback from the subject teacher after a debriefing session with the students at the end of the instructional unit, we can observe that the respective learning objectives have also been achieved by the majority of the students. By developing the games, the students demonstrated knowledge about the way of life of different groups, in different times and spaces, in their cultural and social manifestations, recognizing similarities and differences between them and their conflicts. This understanding can be observed not only through the choice of approapriate and differentiated actors and backgrounds in the games, but also through the selection of related actions and game flow (in the case of action games) and/or questions (in quiz games).

### *5.3.2 Does the instructional unit facilitate learning?*

Most students found the lessons very easy or easy, although, in general, students consider making computer programs rather difficult (Fig. 7).

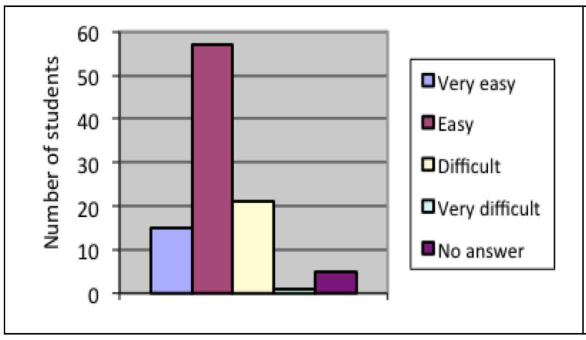

Figure 7a. "The lessons were …"

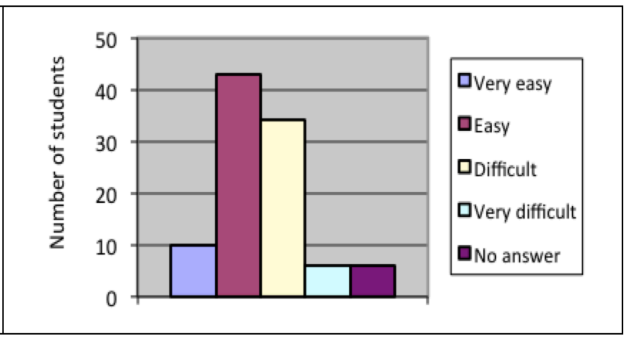

Figure 7b. "Making computer programs is …"

This perception of the students could also be confirmed through the feedback from the teacher (Tab. 7) and qualitative comments from the students (Tab. 8). In general, students easily learned how to use the Scratch environment. Many students learned how to use different commands on their own or asked their colleagues or the instructors for help. Especially the possibility to immediately execute and test programs was observed as essential to support the learning process, making it easy for students to find errors quickly and to correct them and, thus, learning by trial and error.

Table 7. Responses from the teacher's questionnaires.

| **Question** | **Answers** | | | |
|---|---|---|---|---|
| | **Very easy** | **Easy** | **Difficult** | **Very difficult** |
| I noticed that the students found the classes | | 2 answers | | |
| I noticed that for students to learn how to use Scratch was | | 2 answers | | |
| I noticed that for students programming is | | 2 answers | | |

Students considered the freedom to develop any kind of game in the context of the respective History topic very positive. Yet, on the other hand we observed that such a degree of flexibility also required a large amount of support by the instructors as students faced very different issues.

Table 8. Summary of qualitative feedback by the students.

| | |
|---|---|
| **General evaluation of the IU** | "Great, easy and fun."<br>"Cool, it was fun to create a game and play other games, although sometimes I had a hard time."<br>"I loved it! I had no idea how fun computing is and it is not necessarily that difficult."<br>"I found the lessons very interesting, even having some problems."<br>"Complicated.<br>"Very complicated with such a short time to develop the game." |
| **Strengths** | "That I was able to surpass the objectives."<br>"That I knew how to do things."<br>"The challenges."<br>"That I was able to program something that worked and was cool."<br>"The method of learning that was used." |
| **Weaknesses** | "When I did not know how to do things."<br>"When I was frustrated that my program did not work."<br>"The difficulty of the commands, I did not quite understand how they are used."<br>"Difficulties of the games."<br>"Sharing the game, I found this very complicated."<br>"Very complicated to share the game, it stopped working in the middle of sharing." |

In addition, problems with the technical infrastructure also increased the complexity of the lessons. For example, when being without Internet connection, students had to draw actors/scenarios from scratch (and/or by modifying some from the Scratch image gallery) rather than simply use images found on the Internet. Yet, it was impressive that even those problems did not hinder the majority of the students to develop their games, demonstrating a high degree of persistence. An additional problem has been the need of parents to confirm the creation of an account on the Scratch site before the account can be used.

*5.3.3 Does the instructional unit promote a pleasant and enjoyable learning experience?*

The majority of the students evaluated the lessons as excellent or good and considered them fun (Fig. 8).

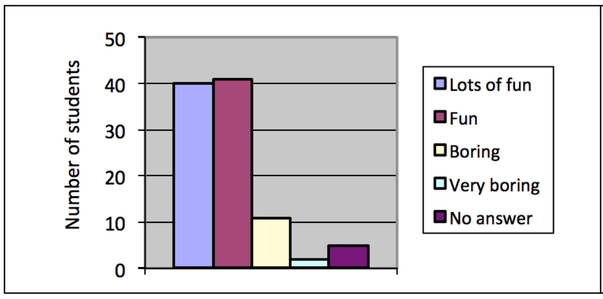

Figure 8a. The lessons were.

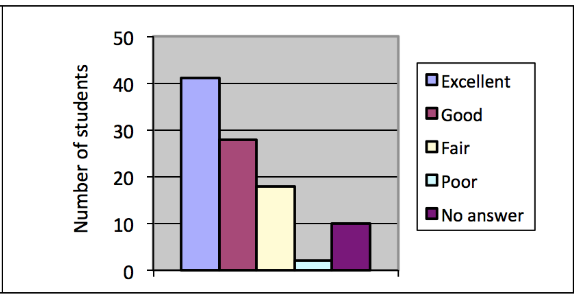

Figure 8b. The lessons were.

This positive assessment is also confirmed by the responses from the teacher (Tab. 9) and by the majority of students that indicates that the time in class passed quickly or very quickly (Fig. 9).

Table 9. Responses from the teacher's questionnaires.

| **Question** | **Answers** | | | |
|---|---|---|---|---|
| | **Lots of fun** | **Fun** | **Boring** | **Very boring** |
| I noticed that students found that programming is | | 2 answers | | |
| I noticed that students found that the classes were | | 2 answers | | |
| | **Strongly agree** | **Agree** | **Disagree** | **Strongly disagree** |
| I noticed that students like to come to computer classes | | 2 answers | | |
| | **Very quickly** | **Quickly** | **Slowly** | **Very slowly** |
| For me classes passed | | 2 answers | | |
| | **Excellent** | **Good** | **Regular** | **Poor** |
| The instructional strategy of the unit is | | 2 answers | | |
| The teaching material is | 2 answers | | | |
| | | | | |

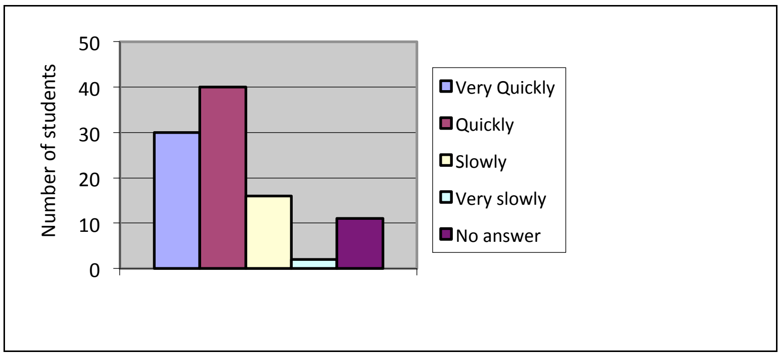

Figure 9. Students'responses on the time passing during the classes.

Still, several students thought that the time passed slowly/very slowly. One of the reasons may be due to the problems we faced with the IT infrastructure. Several times no Internet connection was available, which hindered the search for images or made it impossible to save projects on the online Scratch environment in order to continue working on them in the next class. Another problem observed was that notebooks started to run out of battery at the end of the school period and students had to switch notebooks in order to continue their work. A different problem was that in one class the lectures were taught by the subject teacher only without further assistance (see Tab. 4). In this situation, it took longer to answer the individual questions of the students, which delayed the continuation of their work.

However, in general, the students assessed the IU positively (Tab. 10). They enjoyed the possibility to create characters/scenarios and to freely design their games. A majority also indicated that they would like to have more such computing classes. The students generally welcomed the computing instructors enthusiastically commenting their delight in having a computing class, expressing also disappointment when the classes ended. Many students would have liked the classes to last longer or to occur more frequently. Only very few students did not like the classes.

Table 10. Discursive responses of the students.

| | |
|---|---|
| **What do you think of these classes?** | "I thought these classes were awesome!"<br>"I loved them! I had no idea that computing is fun and not necessarily that difficult. "<br>"I found the lessons great, fun, interesting and learning a lot "<br>"I think it was a unique and enjoyable experience."<br>"I found it interesting and cool, but it's was not enough time."<br>"I found it very cool to do this work with Scratch, as we learned a little bit how it is to work with computing, and I hope to have more classes like this."<br>"I found it very cool, fun and spectacular! I liked a lot to have learned this! Scratch is great."<br>"I thought it was cool, but I think that the instructors could have paid more attention."<br>"I was a bit bored with these classes and had a headache as well."<br>"The classes are boring, and I did not like them." |
| **What did you like most?** | "Being able to design the game freely."<br>"Everything, especially to create the characters."<br>"The freedom we had to create our own stories."<br>"That we learned how to make games and to work with computing in a little bit more elaborated way."<br>"Being able to learn how to work with Scratch, developing a great work."<br>"I liked to learn how to use Scratch, since I never had used it before."<br>"The outcome of the games."<br>"The last class when we played the games ourselves." |
| **What did you like less?** | "That the lessons passed very quickly and that we had only 1 lesson every 15 days."<br>"That the lessons were not given every week."<br>"That there were very few classes."<br>"The large number of classes until the final presentation."<br>"Instructors did not listen to me when I asked for help."<br>"Help was delayed in classes."<br>"To have to redo 8 times the game in Scratch."<br>"The notebook battery always ran out when it was my turn."<br>"Having to share one computer." |

### *5.3.4 Does the instructional unit provide a positive perception of computing?*

Students demonstrated willingness to learn computing and to continue programming with Scratch (Fig. 10).

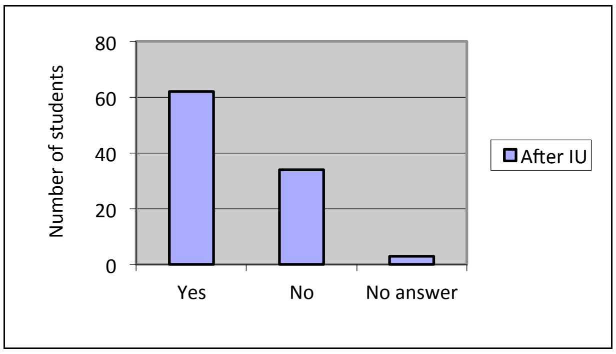

Figure 10. I want to learn more about how to make computer programs.

The IU also seems to help to transmit the perception that computing is fun (Fig. 11). This was also endorsed by the observations of the instructors and asnwers of the teachers, who noticed that students were motivated to learn more about computing. The teachers also confirmed several impacts by learning computing in a fun and motivating way (Tab. 11).

Table 11. Responses from the teacher's questionnaires.

| **Question** | **Answers** | | | |
|---|---|---|---|---|
| | **Strongly agree** | **Agree** | **Disagree** | **Strongly disagree** |
| I noticed that students want to learn more about computing | | 2 answers | | |
| I noticed that students liked programming | | 2 answers | | |
| I noticed that learning to program also teaches students abstraction and logic | | 2 answers | | |
| I noticed that learning computing encourages students to think creatively | | 2 answers | | |
| I noticed that learning computing encourages students to explore new things | 2 answers | | | |
| I noticed that learning computing teaches the student to deal with failures and successes | 2 answers | | | |
| I noticed that learning computing improves student concentration | | 2 answers | | |
| I noticed that computing classes stimulate the sharing of knowledge among students | | 2 answers | | |

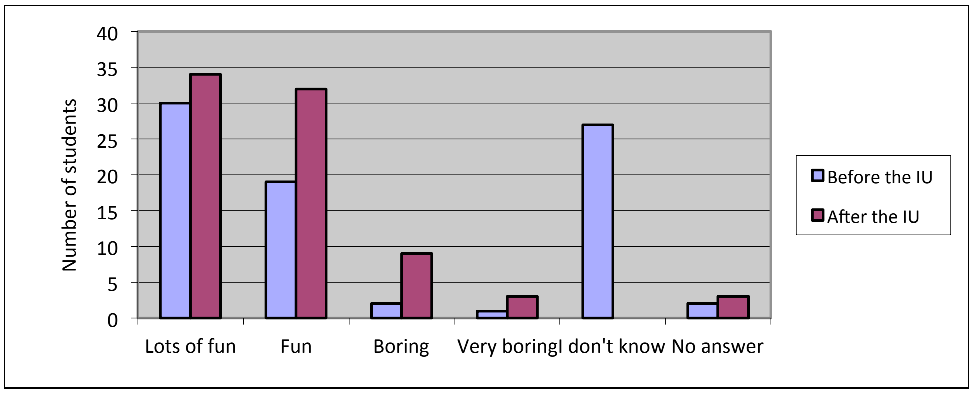

Figure 11. Making a computer programs is:

The ability to solve something that seemed impossible at first was reported by many students as one of the most motivating features of the IU. They also highlighted positively the collaboration with their colleagues, exchanging ideas as well as explaining programming commands to each other. Often students also showed each other what they achieved in their games, which kept many of the students motivated to also complete their game. Some parents also commented that they observed that the fact that their children learned how to make games instead of just simply playing games, kept them motivated. The fact that the game is made entirely by them, and that in the end they can play their own games, gave the children a feeling of empowerment.

### *5.4 Threats to Validity*

Several factors in the research design of our study may have influenced the validity of the results. One threat is related to the way of measuring the evaluation objectives. To reduce errors, we adopted a systematic measurement approach (Basili et al., 94) to systematically refine the evaluation objective in analysis questions and measures, operationalized by data collection instruments. In order to reduce threats due to misunderstandings, the questionnaires were carefully designed, reviewed and piloted using the target audience's language. Answers have also compared to the observations and informal comments collected during the IU.

Another treat may be the sample size. However, a sample size of 105 students can be considered acceptable, even, although, there have been small changes with respect to the participants during the study (e.g., children changing schools and/or not being present in all lessons). However, to allow generalizability of the results it will be necessary to repeat the study in other schools. Still, the results of this study are a first significant feedback on the application of the instructional unit in the context of an exploratory research.

## 6. Conclusion

Following the trend for computing education in schools, this article proposes a way on how to integrate computing education in a multidisciplinary manner in social studies classes in Brazil. An initial evaluation of this instructional unit with 4 classes (with a total of 105 students) provides a first indication that the unit can be effective to achieve learning outcomes with respect to computing practice & programming, computational thinking and collaboration besides reinforcing History topics. Participating students were able to understand and practice the design, programming and testing of games, thereby also learning concepts related to computational thinking. The students easily used the Scratch environment. The instructional unit was designed in a way it also stimulated their collaborative skills through pair programming and by sharing their knowledge and results with their colleagues. Moreover, although, students consider computing difficult, they enjoyed the classes having fun. In consequence, the classes resulted in an increased interest of the students in computing and programming expressing their eagerness for more computing classes in school.

## References

Alves, N. C. (2016). *Desenvolvimento de uma unidade instrucional interdisciplinar para ensinar computação no ensino fundamental*. Undergraduate thesis (Graduation in Computer Science). Brazil, Federal University of Santa Catarina.

Andrade, M., Silva, C., & Oliveira, T. (2013). Desenvolvendo games e aprendendo matemática utilizando o Scratch. *Simpósio Brasileiro de Jogos e Entretenimento Digital. São Paulo*, 260-263.

Aureliano, V. C. O., & Tedesco, P. C. A. R. (2012). Avaliando o uso do Scratch como abordagem alternativa para o processo de ensino-aprendizagem de programação. In *XX Workshop sobre Educação em Computação* (p. 10).

Basili, V. R., Caldeira, G., Rombach, H. D. (1994). *Goal Question Metric Paradigm*. In Encyclopedia of Software Engineering, John Wiley and Sons.

Branch, R. M. (2009). *Instructional Design: The ADDIE Approach*. Springer.

Code.org. (2016). *4 Ways to Recruit Girls to Try Computer Science*. [online] *Available at*: https://code.org/girls.

CodeClubBrasil. (2016). [website] Retrieved from http://codeclubbrasil.org.br/.

Google & Gallup. (2015). *Searching for Computer Science: Access and Barriers in U.S. K-12 Education*, [online] Retrieved from http://services.google.com/fh/files/misc/searching-for-computer-science_report.pdf.

Guzdial, M. (2004). *Programming environments for novices*. Computer Science Education Research. Lisse, The Netherlands: Taylor & Francis, 127-154.

Lin, H. S. (2002). *IT Fluency: What Is It, and Why Do We Need It?*. Technology Everywhere: A Campus Agenda for Educating and Managing Workers in the Digital Age. Jossey-Bass, San Francisco, 39-49.

Malan, D. J., & Leitner, H. H. (2007). Scratch for budding computer scientists. *ACM SIGCSE Bulletin*, *39*(1), 223-227.

MEC. (1998). *Parâmetros Curriculares Nacionais, Terceiro E Quarto Ciclos Do Ensino Fundamental*. Brazil.

MIT. (2016). *Scratch*. [online] Retrieved from http://scratch.mit.edu/.

Monroy-Hernández, A., & Resnick, M. (2008). FEATURE empowering kids to create and share programmable media. *interactions*, *15*(2), 50-53.

Moreno-León, J., & Robles, G. (2015, November). Dr. Scratch: A web tool to automatically evaluate Scratch projects. In *Proceedings of the Workshop in Primary and Secondary Computing Education* (pp. 132-133). ACM.

Naughton, J. (2012). *Why All Our Kids Should Be Taught How to Code*. The Guardian. Guardian News and Media.

Ncwit. (2013). *Top 10 Ways of Recruiting High School Women into Your Computing Classes*. [online] Retrieved from

https://www.ncwit.org/resources/top-10-ways-recruiting-high-school-women-your-computing-classes/top-10-ways-recruiting.

Pazinato, A. M., Teixeira, A. C. O. (2013). *Uso do Software SCRATCH no Desenvolvimento da Aprendizagem e na Interação Construtivista dos Alunos*. Proceedings of the 10th National Education Congress (EDUCERE), Curitiba, Brazil.

Pinto, A. S. (2010). *Scratch na aprendizagem da Matemática no 1º Ciclo do Ensino Básico: estudo de caso na resolução de problemas*. Master Thesis in Child Studies, 2010, University of Minho, Braga, Portugal.

Q Qualls, J. A., & Sherrell, L. B. (2010). Why computational thinking should be integrated into the curriculum. *Journal of Computing Sciences in Colleges*, *25*(5), 66-71.

Branch, R. M. (2009). *Instructional Design: The ADDIE Approach*. Springer.

Resnick, M. (2007). *Sowing the Seeds for a more creative society.* Learning and Leading with Technology. US & Canada: International Society for Technology in Education (ISTE), 18-22.

Rusk, N.; Resnick, M. and Maloney, J. (2006). *Scratch and 21st Century Skills*. MIT Media Lab. US: Lifelong Kindergarten Group.

Seehorn, D., Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., ... & Verno, A. (2011). CSTA K--12 Computer Science Standards: Revised 2011.

Snap! [online] Retrieved from https://snap.berkeley.edu/.

Wilson, A., Moffat, D. C. (2010). *Evaluating Scratch to introduce younger schoolchildren to programming*. Proceedings of the Psychology of Programming Interest Group Workshop, Madrid, Spain.

Wing, Jeanette M. (2006). *Computational thinking*. Communications of the ACM, Vol. 49, No. 3, 33-35.

Wohlin, C., Runeson, P., Höst, M., Ohlsson, M. C., Regnell, B., & Wesslén, A. (2012). *Experimentation in software engineering*. Springer Science & Business Media.

Yin, R. K. (2013). *Case study research: Design and methods*. Sage publications.