Towards an Open Educational Resource Sensitive to Student's Context to Support Introductory Programming Courses

Myke Morais de Oliveira*, Leo Natan Paschoal*, Patricia Mariotto Mozzaquatro Chicon†, Ellen Francine Barbosa*
*University of São Paulo (ICMC-USP), São Carlos, SP – Brazil
†University of Cruz Alta (NEAD - UNICRUZ), Cruz Alta, RS - Brazil
E-mail: mykeoliveira@usp.br, paschoalln@usp.br, pmozzaquatro@unicruz.edu.br, francine@icmc.usp.br

***Abstract*—Previous studies mention that students have a hard time learning introductory programming courses. Several studies have already been conducted to understand the problems concerning programming learning. Some investigations managed to classify the problems, they are: some students are unable to obtain a concrete understanding computer programming concepts; some students are not able to apply programming concepts in the construction of programs; some students have no motivation to learn the subject; some students cannot understand the programs already implemented, and some have difficulties in factoring and refactoring programs. To support the learning of these courses, researchers in the field have established teaching support mechanisms, which can be used by teachers when teaching content or by students to train and get prepare for assessments. In particular, some studies are proposing open educational resources. However, even if there is a strong interest in establishing these open resources, the existing educational resources do not consider the difficulties that the students may have in these courses (i.e., resources that increase students' motivation but do not support students who have difficulties learning programming concepts). Still, it is necessary to consider the previous knowledge that each student may have when starting a course. Some students may have some programming knowledge, while others may not. It is also necessary to consider that students have their preferences for learning materials when they are studying (i.e., some prefer video lessons, while others prefer slides). In this sense, this paper addresses the establishment and feasibility study of an open educational resource dedicated to teaching programming, which is specialized for students who have difficulties in learning and applying concepts, understanding programs, factoring and refactoring their programs and/or do not have the motivation to program. This open educational resource was planned considering some definitions related to the students' particularities (i.e., previous knowledge, preferences for teaching materials and programming difficulties), with the function of identifying them and adapting to these particularities. We describe how we have developed the educational resource, how we plan and conduct preliminary evaluations and, in the end, we raise directions for next steps.**

***Index Terms*—Computing education, Enginnering education, Cognitive styles.**

I. Introduction

Programming is one of the first subjects in which students will have contact when entering computing undergraduate programs. This subject is mentioned in some studies as being a course in which students have difficulty in the learning process. Therefore, several studies have already been conducted to understand the problems concerning programming learning. Some of them managed to classify the problems, they are: some students are unable to obtain a concrete understanding of computer programming concepts; some students are not able to apply programming concepts in the construction of programs; some students have no motivation to learn the subject; some students cannot understand the programs already implemented; some students have difficulties in factoring and refactoring programs [1]. Due to these problems, some students may fail or even drop out of the program. Even if they pass the exams, they may be in deficit.

There are many approaches and digital tools presented in the educational scenario that can facilitate and improve student's competence. Open Educational Resources (OER) are teaching and learning materials that are available for the public to be accessed, used, and adaptative with no restrictions [2]. However, just offering a great number of open materials is not enough [3]. As mentioned by Dichev and Dicheva [3], OER should support different teaching activities, and, among other characteristics, it should be adaptable.

Adaptation is one of the characteristics mentioned by Wiley [4] (revise) that can bring improvements to the interface and content of OER, which can benefit and motivate students' learning. The study by Triantafillou [5], for example, addresses the use of cognitive styles and learning styles to accommodate students' differences. Cognitive styles are understood by the habits that people have of solving problems, thinking, perceiving, and remembering [6]. Cognitive styles can also be used to represent user preferences in an educational system, for example. In the study by Geller [7], cognitive styles are used to characterize students' preferences for the formats of materials that will be made available.

Each student has a profile, constituted by their own goals, needs, expectations, experiences, and expertise level related to the subject they are going to learn. So it's important to make use of mechanisms to support teaching that consider information about the student at the moment the content is presented, in order to contemplate every particular preference of each student. With this perspective, this paper aims to present an open educational resource for teaching introductory programming concepts that adapts itself to the user profile, taking into account their cognitive style and expertise level. Cognitive styles are used in this study to represent students' preferences for the format of teaching materials that will be exposed in the OER [7]–[9]. While the student's expertise level is considered to identify their main difficulties related to introductory programming subjects [10], [11]. Based on these indicators, it was possible to implement the OER. We aim that the open educational resource can adapt to the student's profile and be composed of different didactic programming resources, such as tutorials, articles, videos, and textbooks so that the student can learn from teaching materials in formats of their choice.

In this paper we present the open educational resource to support the teaching of programming, the flexibility of the study, and the results obtained. A group of students who had already studied the subject volunteered to evaluate the educational resource. The aspects related to the presentation of the resource content and the extent to which open materials were able to motivate the student to study were evaluated.

II. Related Works

Several early studies describe the development of mechanisms to support teaching programming. As a result, many literature reviews and surveys have been conducted to synthesize studies already produced and help researchers to recognize subjects that have already received their proper attention. This type of research also allows researchers to find open research opportunities and gaps. Throughout this section, we will present a discussion of some educational resources that have been mapped by secondary studies in the area. It is worth mentioning that secondary studies helped us to recognize the state of the art on educational resources produced to support teaching programming.

The work of Mitamura et al. [12] presents a discussion on the development of serious games to support this content. This kind of resource can be quite effective in the learning process as it encourages the student to learn concepts and programming logic while having fun. Mitamura et al. [12] also cover HIT Typing, Serious Cube, a 3D action game, and a game defined by the authors. While Mitamura et al. [12] discuss some games for teaching programming that they defined, Silva et al. [13] carried out a systematic identification of these teaching support mechanisms. They identified 29 educational games, noting that there are games for different programming languages - namely: Java, C#, Visual Basic, Python, C, F#. Although the authors [13] promote discussions about the benefits that this type of educational resource can offer to students' learning, the analyzes did not raise a discussion associated with the possibility of making them available in an open way, as an open educational resource.

In Gonçalves et al. [14], during the description of a learning object to teaching repetition structures (i.e., one of the most important programming concepts), a survey of previous studies that address the establishment of learning objects to support teaching is done about the content. The study mentions the works of Sampaio and Souza [15], Cechinel et al. [16], Saraiva and Netto [17]. Each study addresses learning objects for the area, with different characteristics. Although the survey of these learning objects occurs, there is no mention of making them available as open content.

The systematic literature review planned and conducted in the study by Henrique and Rebouças [18] was intended to map learning objects dedicated to teaching the concepts of object-oriented programming that are considered difficult to understand. In this study, learning objects were found for software development environments (i.e., IDE), learning objects for teaching in the context of intelligent tutoring systems, learning objects defined based on animation software, and learning objects in the format of card games. Aspects of reusing these educational resources are not addressed.

A systematic literature review was also used by Marcolino and Barbosa [19], in a study whose intention was to identify the software systems and educational applications that support the teaching of programming. The authors identified more than 81 applications, which were classified according to their characteristics, in the following categories: visualization tools, automatic program evaluation environment, programming support system, microworlds. Still, the authors describe that they had difficulty in observing significant efforts to facilitate the reuse process. Therefore, it must be understood that educational resources are not characterized as an open educational resource.

Recently, Luxton-Reilly et al. [20] observed that secondary studies produced within the framework of mechanisms and approaches to support programming teaching were being focused on specific aspects of introduction to programming. In this sense, seeking to offer the community that has invested efforts in this theme, a systematic mapping was defined without focusing on a specific aspect. The authors identified 1666 studies and, during the analysis, grouped them according to their contributions. They identified programming environments, programming language extensions to facilitate learning, tools to support the practice of programming, tools to support problem-solving practices, tools that help students to collaborate, among others. The authors note some open educational resources in the study, paying special attention to the MOOC platform courses.

The work of Grandl et al. [21] fits into open educational resources. In the study, the authors address the MOOC definition to teach programming to children. Each learning content has been established as an open educational resource, with a Creative Commons license. The course covers program creation in a language similar to Scratch, repetition lessons, conditional structures, sensors, variables, and resources that enable the construction of a game. At the end of the course, participants are expected to learn to implement a game.

Based on the analysis of the works described, we were able to observe that many tools have already been established to support teaching programming, but the movement is still small when it comes to the production of open educational resources in this theme. Besides, the secondary studies analyzed did not identify support mechanisms adaptable to the student's profile. Whereas there is a growing movement in some areas for the production of educational environments sensitive to the student context, and the emergence of educational resources attached for some domains (e.g., a virtual world for teaching computer networks containing materials that take into account the student's expertise level and cognitive style; a conversational agent who solves the doubts of software engineering students according to preliminary knowledge), we believe that contributions that include open educational resources sensitive to students' context are welcome to the area.

III. Open Programming Resource

A. *Previous*

Intending to contribute to the teaching programming theme, we decided to define an open educational resource that considers student preferences. Therefore, we observed studies on the theme of computer education that has used concepts of sensitivity to the context of students to define educational environments and resources. In particular, those researched by Herpich et al. [8], Krassmann et al. [9] and Paschoal et al. [10] Each study will be described below.

In the study by Herpich et al. [8], the authors address the establishment of an immersive environment and a conversational agent, both dedicated to teaching computer networks. To promote an environment that addresses student characteristics, the authors considered the identification of cognitive styles and expertise levels. The students' cognitive styles represent how the student prefers to receive some type of information (e.g., some students prefer videos, others prefer to read texts, among others). The expertise levels, on the other hand, illustrate how much of the preliminary knowledge the student has when using the educational resource. This information and its treatment are important given that a class of students is composed of people with different backgrounds, and some of them already have some previous knowledge about the content addressed in the resource, while others do not have much knowledge about the subject. These definitions were incorporated into the environment and the agent, as follows: (i) the immersive environment that has teaching materials to support learning, presents itself to the student considering his cognitive style; (2) a conversational agent offers support answering students' questions, and indicating ways for learning considering the student's previous knowledge.

In the research by Krassmann et al. [9], an educational game called JASPION is presented, which is defined in an immersive environment with the training of computer network students. The educational game has a narrative that mixes the styles adventure, quest, and quiz, in a plot inspired by the television series "Kyojuu Tokusou Juspion", considering gameplay profiles. Each gameplay profile is defined based on the student's expertise level and cognitive style. This profile influences the quests and quizzes of the game since they would be presented based on the student's previous knowledge. At each stage of the game, the student will have at his disposal a set of teaching materials that consider his preferences for teaching material formats. These materials can be used by students to successfully overcome the challenges that are presented throughout the game.

Finally, the study by Paschoal et al. [10] addresses a conversational agent for software engineering courses registered on the Moodle. The conversational agent entitled Ubibot can help the student in exercises of software requirements and case tools, considering the expertise level of the students and it identifies their performance throughout the course. As Ubibot is integrated with Moodle, it tracks the grades that the student receives when performing activities of the course (i.e., the student's performance). Thereby, it will help the student by offering tips that are personalized to the knowledge in the period/moment when the student is in the course. When the student starts the course, his previous knowledge is identified, but if he already has activities and grades scored on the Moodle, the agent will adapt to the student's current performance to help him.

These previous studies helped us to define how we could include aspects of sensitivity to the student's context in our educational resource. In the next subsection, we present some characteristic definitions of the open educational resource for teaching programming, which considers student preferences.

B. *Features*

Considering that the problems associated with programming learning are mostly related to the understanding of programming concepts, whether they are related to theoretical issues (understanding a concept) and practices (using a concept), when planning the educational resource, we define that the educational resource would be designed considering the realization of practical activities for knowledge consolidation. Our intention is for the student to use the open educational resource to learn through activities. Thus, the educational resource presents activities in the form of challenges and the student has to solve them.

Context sensitivity is used in the educational resource to adapt the challenges that will be presented to students. In this sense, the educational resource considers the student's expertise level when presenting the activity, whether it is associated with programming concepts or the use of programming concepts. Also, to support the student, the educational resource offers support in each challenge, providing educational materials that can be useful for the student to solve the challenge. These materials are suited to students' cognitive styles.

Considering this planning, we model the educational resource to describe how it should work. When designing its operation, we recognize the need for functionality for user registration, so that the open educational resource has an understanding of who is the student who is using it. In addition, we have defined mechanisms for recognizing and treating information about the student (expertise level and cognitive style). At the same time, it was necessary to prepare the challenges for each expertise level supported by the educational resource and to retrieve teaching materials in different formats in order to contemplate the different cognitive styles.

C. Sensitivity to Cognitive Style

To address sensitivity to cognitive style, we used an instrument that has already undergone a validation process, which was defined by Geller [7] research. This instrument consists of 16 statements, which are related to six cognitive styles: holistic, serialistic, divergent, convergent, reflective and impulsive. A brief description of each style is described below based on discussions in the literature on each of the dimensions [22], [23].

- **Holistic:** holistic students are more prominent in the global context, since the beginning of an activity, they prefer to analyze a vast amount of information looking for patterns and relationships between them. They can solve complex problems with ease and are almost always good synthesizers.
- **Serialistic:** serialist students place more emphasis on separate topics and in logical sequences, looking for patterns and relationships in the process to confirm or not their assumptions. Their assumptions are simpler and have a logical-linear approach (from one assumption to another step by step). They are competent to solve problems and are often good analysts.
- **Convergent:** convergent students use both abstraction and common sense in the practical application of ideas and theories, always looking for the best solution to a practical problem and like to solve practical problems. They are better with technical tasks and problem-solving than with social and interpersonal events.
- **Divergent:** divergent students are more creative, with imaginative, original and clear responses. They have a preference for informal problems, which request the spread of several equally conceivable responses, in which they are highlighted in the quantity, diversity, and uniqueness of the responses. They are more willing to solve less structured problems. Socially, they are more impulsive and threatening.
- **Reflective:** reflective students think before doing anything, like a course, for example. They have the most organized, sequenced thoughts and they make a reflective assessment before responding to an answer.
- **Impulsive:** Impulsive students do not organize previous responses, they usually accept the first hypothesis and offer quick solutions to problems.

As the cognitive styles are related to the students' perceptions, there are recommendations on which formats of didactic materials can be more suitable for each dimension. According to Geller [7], students with holistic styles prefer books, while serial students prefer materials organized into topics. Students with convergent dimensions prefer didactic materials structured in article format, while students with divergent styles prefer tutorials. Students with reflective styles tend to prefer book chapters and students classified in the impulsive category like videos. This relationship between cognitive styles and teaching materials was adopted in previous studies (e.g., Herpich et al. [8] and Krassmann et al. [9]). In this sense, cognitive styles are identified and the resource modifies the information presented to the student based on what was detected.

To identify these cognitive styles, the student needs to respond to a set of statements (18 in total). Thus, after the student has signed up for the educational resource, he will need to complete an instrument containing the instructions (see Table III-C). The statements are related to cognitive styles, for example, statements 1, 3, and 7 correspond to the convergent style. Table III-C also shows the distribution of statements regarding the dimensions of cognitive styles (see the dimensions of cognitive styles on the left side in the vertical of Table III-C). We emphasize that the response options for each statement have weight (see column 3 of Table III-C).

D. Sensitivity to Expertise Level

To support the identification of the student's expertise level, we define an instrument capable of recognizing the expertise level. We were inspired by the work of Paschoal et al. [10], to define the content that would be included in the educational resource (more details will be presented in Section III-E), and we developed 12 exercises/questions about the content. We intended to use these questions to gain insights about student's previous knowledge. We emphasize that this strategy was originally described in Possobom et al. [11] work, in which a set of questions and rules are defined to classify students based on the number of correct answers and errors.

The questions that we developed to identify the student's expertise level are a mix of multiple-choice, multiple-select, shuffling (cloze), filling in fields with keywords, and true or false questions. Each question has a value of 100 points. In this sense, the maximum score that a student can achieve is 1200 points. Based on this value, we defined some partitions of the scores to establish the bounding factor (classifications based on rules defined in Possombom et al. [11]). This limiting factor is a range of values that represents the student's expertise level. Table III-D presents the classifications.

E. Definition of content, challenges and teaching materials

When defining the mechanism to identify the student's expertise level, we faced the need to determine which topics/content would be addressed by the educational resource. To designate the subjects, we take into account the syllabus of the introductory programming course of the university where we operate. In this sense, we stipulate the following subjects: forms of representation of algorithms, type of variables, writing, and reading of variables, simple conditional structure, composite conditional structure, repetition structures, one-dimensional array, and multidimensional array.

TABLE I
INSTRUMENT ASSERTIONS TO DETECT COGNITIVE STYLE

	Assertions	Response options	Weights
Convergent	1) I find it difficult to create something original.	O Strongly Agree	4.4211
		O Agree	3.3121
		O Disagree	2.2541
		O Strongly Disagree	0.51
Impulsive	2) In many situations, I am not an attentive person because I am hurried.	O Strongly Agree	4.421947
		O Agree	3.312937
		O Disagree	2.254927
		O Strongly Disagree	0.5917
Convergent	3) Usually, I follow the guidelines given without questioning.	O Strongly Agree	4.4212
		O Agree	3.3122
		O Disagree	2.2542
		O Strongly Disagree	0.52
Reflexive	4) I am a very attentive and organized person.	O Strongly Agree	4.421945
		O Agree	3.312935
		O Disagree	2.254925
		O Strongly Disagree	0.5915
Impulsive	5) In general, I don't usually think much to allocate my time.	O Strongly Agree	4.421948
		O Agree	3.312938
		O Disagree	2.254928
		O Strongly Disagree	0.598
Serialist	6) When performing a task, I prefer to use a step-by-step process, working with small amounts of data at a time.	O Strongly Agree	4.4219
		O Agree	3.3129
		O Disagree	2.2549
		O Strongly Disagree	0.59
Convergent	7) In general, I accept the rules established.	O Strongly Agree	4.4213
		O Agree	3.3123
		O Disagree	2.2543
		O Strongly Disagree	0.53
Serialist	8) I pay more attention to the small informational elements of a study or work material.	O Strongly Agree	4.42194
		O Agree	3.31293
		O Disagree	2.25492
		O Strongly Disagree	0.591
Holistic	9) I usually emphasize the global context and not the specific aspects of the tasks I perform.	O Strongly Agree	4.4216
		O Agree	3.3126
		O Disagree	2.2546
		O Strongly Disagree	0.56
Holistic	10) The overall context of a situation is the most relevant element for decision making.	O Strongly Agree	4.4217
		O Agree	3.3127
		O Disagree	2.2547
		O Strongly Disagree	0.57
Impulsive	11) In many situations, I give answers without giving much thought to them.	O Strongly Agree	4.421949
		O Agree	3.312939
		O Disagree	2.254929
		O Strongly Disagree	0.599
Divergent	12) I enjoy experiencing new situations.	O Strongly Agree	4.4214
		O Agree	3.3124
		O Disagree	2.2544
		O Strongly Disagree	0.54
Reflexive	13) I usually think a lot before making decisions.	O Strongly Agree	4.421947
		O Agree	3.312937
		O Disagree	2.254927
		O Strongly Disagree	0.5917
Holistic	14) When reading a text, I pay more attention to the general idea than to the informative details of it.	O Strongly Agree	4.4218
		O Agree	3.3128
		O Disagree	2.2548
		O Strongly Disagree	0.58
Divergent	15) I am able to formulate original and creative responses frequently.	O Strongly Agree	4.4215
		O Agree	3.3125
		O Disagree	2.2545
		O Strongly Disagree	0.55
Reflexive	16) I usually think well before giving an answer.	O Strongly Agree	4.421948
		O Agree	3.312938
		O Disagree	2.254928
		O Strongly Disagree	0.5918
Serialist	17) Faced with written material, I emphasize each topic separately and only then do I seek relationships between the parties.	O Strongly Agree	4.421946
		O Agree	3.312936
		O Disagree	2.254926
		O Strongly Disagree	0.5995
Divergent	18) I enjoy daring and trying to create something different.	O Strongly Agree	4.4215
		O Agree	3.3125
		O Disagree	2.2545
		O Strongly Disagree	0.55

Source: Based on Geller [7]

TABLE II
IDENTIFICATION OF THE STUDENT'S PRIOR KNOWLEDGE

Expertise level	Limiting factor
Basic expertise	0 <= scores <= 500
Intermediate expertise	500 <scores <= 800
Advanced expertise	800 <scores

Given the definition of the issues that would be addressed, we prepared challenges considering the content and expertise levels supported by the educational resource. We defined ten challenges, each with three different variations. Each variation has an increase in difficulty to contemplate the students' expertise levels. The challenges involve exercises in which the user must select a correct definition for some programming concept, relate programming concepts, inspect a flowchart, pseudocode or code, and indicate functionality or outputs, implement functionalities in pre-established algorithms.

When developing the challenges for each content and expertise level, we used our experiences when teaching the course of introduction to programming. One of the developers has been a professor of this course more than 15 times. Therefore, we believe that we have managed to produce challenges that can provoke students to carry them out. To exemplify the challenges, three exercises of the educational resource are described. Figure 1, for example, presents an exercise in which the student needs to indicate forms that can be used to represent an algorithm. In Figure 2, an exercise is presented in which the student needs to select the correct option that describes the definition of the command "printf" in C programming language. Finally, in the exercise shown in Figure 3, the student needs to inspect an algorithm and complement it, based on a description of the algorithm and some code snippets.

In addition to developing challenges for each content, we also defined didactic materials for student support for each one of the contents. We had to find teaching materials in different formats (webpage articles, tutorials, videos and books) and that were available in an open content format, to ensure that they would be able to reuse it and not injure our open educational resource characteristics.

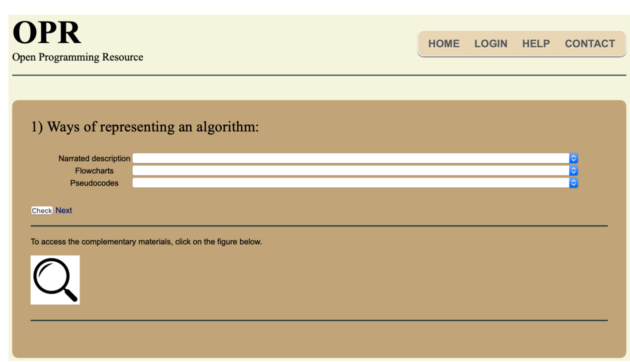

Fig. 1. Example of an exercise of the basic level of knowledge

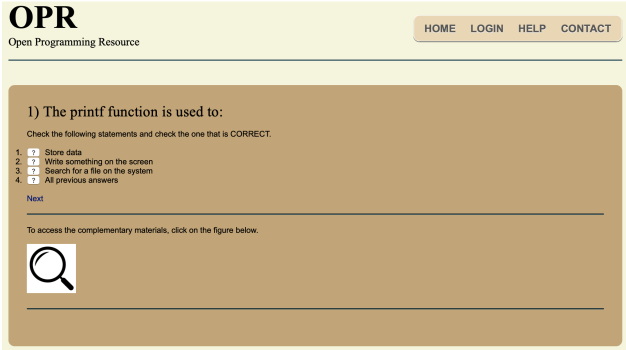

Fig. 2. Example of an exercise of the intermediate level of knowledge

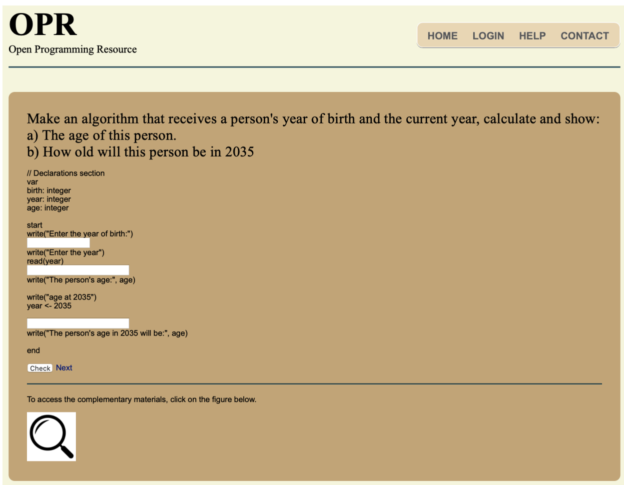

Fig. 3. Example of an exercise of the advanced level of knowledge

F. Educational resource implementation

The previous sections seek to guide decisions that were made in the establishment of the educational resource. This section, on the other hand, aims to clarify aspects of the development of the resource, in the sense of implementation (coding).

The open educational resource was designed as a software system intended for students who may be interested in developing or improving their programming skills. The resource has the four basic operations CRUD (Create, Read, Update and Delete) to handle user data processing. In this sense, the user will only be able to access the main functionalities (challenges and teaching materials) after signing up. Once the user is enrolled, he will be able to access the resource, which must be done through a login and password (both are defined by the user himself).

Once the student logs in to the educational resource, he will be faced with an interface containing the instrument that aims to identify his cognitive style (See Figure 4). Some instructions are provided indicating that the user is required to complete the instrument, as his responses will help the educational resource to better understand the student's profile, and this information will be used to improve the student's experience in using the resource. If the student does not complete it, the challenges will not be made available.

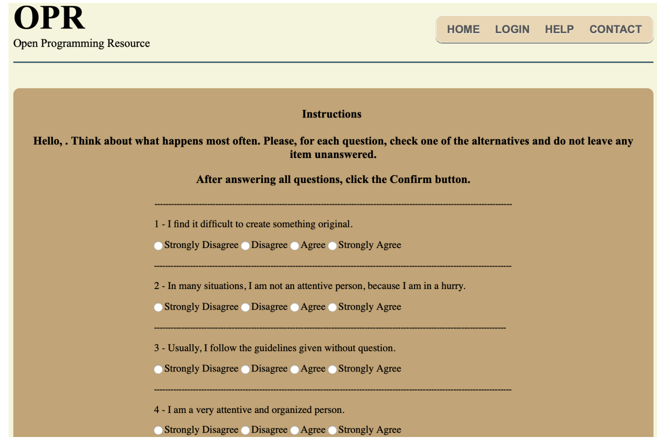

Fig. 4. Instrument to detect the student's cognitive style

After the user responds to the cognitive style instrument, he will come across a set of exercises (See an example in Figure 5). These exercises are used by the educational resource to gain an understanding of the experience that this user may have regarding the programming domain. There is a brief instruction on what the user will need to do. At that time, he will not have supporting teaching materials at his disposal and will be notified by the educational resource that he should avoid using any consultation with third-party materials because the educational resource intends to have a real understanding of the student's prior knowledge so that it manages to organize the challenges before presenting it to the student.

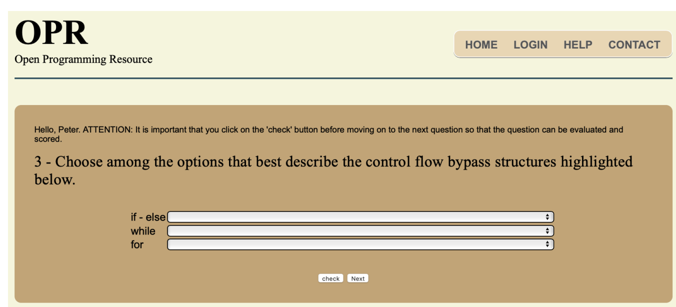

Fig. 5. Question to assess the expertise level

After the resource receives responses from users, it will have 'awareness' about students' contextual information. Based on this, he will present the challenges to the student. These challenges can be solved at any time the user wants. Some examples of exercises have been described in Section III–E. The teaching materials that we make available in the educational resource can also be accessed from the moment the student begins to perform the exercises.

To describe the flow of execution described throughout this section, we used a UML (Unified Modeling Language) model. Figure 6 graphically represents the behavior of the educational resource considering its functionalities.

To implement the features discussed throughout this section, we used the programming language PHP (Hypertext Preprocessor), JavaScript and HTML (Hypertext Markup Language). Besides, the data collected on user information (profile and data that characterize contextual information) are stored in a database. This database is managed, manipulated and organized by the MySQL database management system. Finally, it is worth noting that the challenges were implemented using the Hot Potatoes[1]. This software enabled the creation of interactive challenges in different formats, such as multiple-choice, short response, confusing sentences, filling in the blanks, among others. Because the software allows export in HTML and JavaScript format, we used the codes and incorporate them into the educational resource.

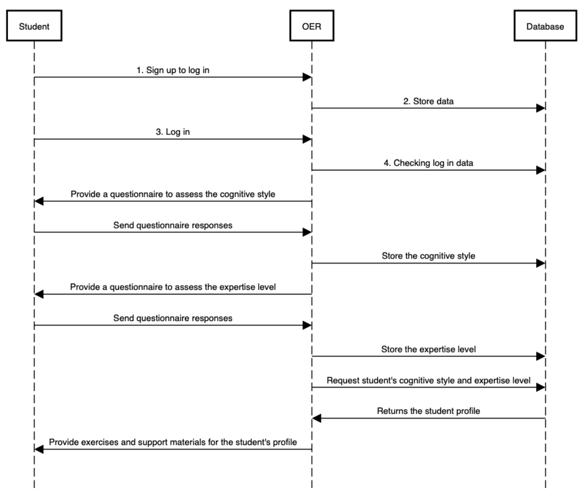

Fig. 6. Sequence diagram

IV. Feasibility Study

After the establishment of the open educational resource, we conducted a feasibility study [24]. This study was intended to determine whether the educational resource is feasible for use in the context of students who are beginning to learn to program. We carried out a plan to address this study, which had the voluntary participation of some students. Finally, we did some analysis to understand if the resource is viable for its purpose.

A. Planning

Considering the objective of the feasibility study, we planned an extension course with a 4-hour workload to be offered to students at our university, who were starting to take the programming course in the Computer Science bachelor's degree. The extension course was specific to students who were interested in learning to program and who had already started to study programming subjects.

We planned the course so that participants would use the open educational resource during a 4-hour session. We intended to present the resource and encourage students to use it. Thus, students would have to try the resource, perform the challenges and consult the teaching materials available in the educational resource. After interacting with the resource, we planned an evaluation, so that students made their perceptions of the educational resource explicit.

For the evaluation, we planned a form composed of six statements and closed response options. We use different response options, as shown in Table IV-A. We did this to draw the student's attention, aiming to make sure that he was careful when answering the form.

TABLE III
Educational resource evaluation questions

Assertions	**Response options**
1) When I first looked at the OER, I had the impression that it would be easy for me.	○ Not true ○ Slightly true ○ Moderately true ○ Mostly true ○ Very true
2) The learning process of OER commands is time-consuming.	○ Disagree ○ Undecided ○ Agree
3) It's easy to forget how to carry out the actions in this OER.	○ Disagree ○ Undecided ○ Agree
4) The presentation of the contents favored my learning.	○ Disagree ○ Undecided ○ Agree
5) It is clear to me how the content of the educational resource is related to things I already know.	○ Not true ○ Slightly true ○ Moderately true ○ Mostly true ○ Very true
6) I learned some things that were surprising or unexpected.	○ Not true ○ Slightly true ○ Moderately true ○ Mostly true ○ Very true

B. Execution

After carrying out the feasibility study planning, we invited undergraduate students to participate in our extension course. Despite widespread dissemination over a period of a few weeks, we found 16 students interested in voluntarily participating in our study. But even with a limited number of students, we carried out the feasibility study. We hosted the educational resource on a network server at our university and conducted the study in a computer lab. Thus, the students did not have to use their equipment.

As we had planned, we continued driving. During execution, we notified students about taking the time to use the educational resource to have fun, ask questions and learn. Above the objective of the feasibility study was our interest in making them learn through the educational resource.

After interacting with the educational resource, we asked students to be careful when answering the form that was intended to collect student feedback on some perceptions they had when using the resource. We warned students that they were supposed to select the response options that most closely related to their feelings. We warned that we would like sincere answers, because we intended to use the data to promote improvements in the resource.

[1]More information available in: <http://hotpot.uvic.ca/>.

C. Results

After performing the study, we tabulated the results and performed the analyzes. Table IV-C presents a summary with the percentages of responses from the participants.

TABLE IV
PERCENTAGES OF RESPONSES FROM THE PARTICIPANTS

⋆	Response option	Percentage (%)
1st	Not true	12.5
	Slightly true	12.5
	Moderately true	50
	Mostly true	6.4
	Very true	18.8
2nd	Disagree	56.3
	Undecided	15.5
	Agree	31.3
3rd	Disagree	62.5
	Undecided	18.8
	Agree	18.8
4th	Disagree	18.8
	Undecided	6.2
	Agree	75
5th	Not true	0
	Slightly true	18.8
	Moderately true	31.3
	Mostly true	31.3
	Very true	18.8
6th	Not true	18.8
	Slightly true	18.8
	Moderately true	43.8
	Mostly true	12.5
	Very true	6.3

Legend: ⋆ – Assertive

Regarding the first assertive, we intended to understand what was the initial perception that the OER provoked in the students. We observed that most students believed that it would be easy to use the educational resource. This result is important, as it provides the feeling of self-confidence, and it is the feeling that we would like the educational resource to prove to students.

In the second statement, the purpose was to observe whether the initial perception has changed. We hope to have raised a question associated with the difficulty of learning how to use the OER. Through the results, we found that most students had no difficulties in learning how to use the resource. A similar result was found in the third statement that questioned whether students forgot the actions that needed to be taken to handle the resources available in the OER.

As OER is sensitive to cognitive style and expertise level, statements 4 and 5 were related to contextual information. In the fourth statement, we try to understand if the students were satisfied with the format of the teaching materials (i.e., how the content was presented). In general, we observed that the majority of the students were quite satisfied with the formats of the supporting resources that are available in OER. The fifth statement, on the other hand, seeks to understand whether students were able to perceive that the challenges presented by the OER were in accordance with the knowledge they had when starting to use the resource. In our understanding, even if the answers to this question did not focus on one of the answer options, we believe that they managed to realize that there was a treatment of the information that was provided.

Finally, in the last statement, our goal was to find out if the students were able to learn by interacting with the OER. Through the results, we observed that the students pointed out different alternatives and the one that served a larger number of students was 'Moderately true'. This may indicate that the students' perception was not entirely true concerning the effective learning of the content through the OER. On the other hand, it also indicates that few students had a reverse feeling. We believe that we need more focused results to assume that the students can learn effectively through OER.

V. CONCLUSIONS

Previous studies have reported on the problems associated with learning to program [1], [25], [26]. In the work of Souza et al., the authors raise problems that are being discussed in the literature. Faced with these problems, the computer education community, in particular programming education, has established some mechanisms to support the teaching of the content covered in courses on the subject. Despite the innumerable initiatives promoted and the diverse educational resources developed, they have not been made available as an open educational resource. Also, there was no concern so far about establishing educational resources sensitive to the student's programming context. On the other hand, some discussions started to emerge in other contexts, such as teaching computer networks and teaching software engineering.

Given the problems encountered, we intended to present some actions that we carried out towards the establishment of an open context-sensitive educational resource. We take inspiration from previous work by computer colleagues to define what student information to use and how to handle that information in the open educational resource. After the establishment, we proposed and conducted a feasibility study to understand the students' perceptions of use. Although the results are not statistically relevant, because the sample was less than 30 participants, through the feedback expressed by the students, we were able to have a view on some issues.

The study can be considered an initial work because we do not yet have an empirical evaluation to show the community that we are in the right direction. In this sense, as future work, we hope to evolve our work, conduct an evaluation in a class of students and collect again the perceptions of these subjects. Although we do not have concrete results, we do provide the open educational resource on Heroku[2] as an app, which is hosted on GitHub[3], hoping that the community will contribute to its evolution.

ACKNOWLEDGEMENTS

We would like to thank the University of São Paulo, University of Cruz Alta, and the Brazilian funding agencies: the State of São Paulo Research Foundation - FAPESP (under processes no. 2018/26636-2), Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001, and CNPq.

[2] Available in: https://open-programming-resource.herokuapp.com/template.html
[3] Available in: https://github.com/mykeoliveira/OER.

References

[1] D. M. Souza, M. H. da Silva Batista, and E. F. Barbosa, “Problemas e dificuldades no ensino de programaçao: Um mapeamento sistemático (in portuguese),” *Revista Brasileira de Informática na Educação*, vol. 24, no. 1, pp. 39–52, 2016.

[2] S. United Nations Educational and C. Organization, “Forum on the impact of open courseware for higher education in developing countries. final report,” 2002.

[3] C. Dichev and D. Dicheva, “Open educational resources in computer science teaching,” in *ACM Technical Symposium on Computer Science Education*, p. 619–624, 2012.

[4] D. Wiley, “Defining the “open” in open content and open educational resources,” 2014.

[5] E. Triantafillou, A. Pomportsis, S. Demetriadis, and E. Georgiadou, “The value of adaptivity based on cognitive style: an empirical study,” *British Journal of Educational Technology*, vol. 35, no. 1, pp. 95–106, 2004.

[6] R. Riding and I. Cheema, “Cognitive styles—an overview and integration,” *Educational psychology*, vol. 11, no. 3-4, pp. 193–215, 1991.

[7] M. Geller, *Educação a distância e estilos cognitivos: construindo um novo olhar sobre os ambientes virtuais (in Portuguese)*. PhD thesis, Universidade Federal do Rio Grande do Sul, Porto Alegre, Brazil, 2004.

[8] F. Herpich, F. B. Nunes, G. B. Voss, and R. D. Medina, “Three-dimensional virtual environment and npc: A perspective about intelligent agents ubiquitous,” in *Handbook of Research on 3-D Virtual Environments and Hypermedia for Ubiquitous Learning* (F. M. M. Neto, R. C. de Souza, and A. S. Gomes, eds.), pp. 510–536, IGI Global, 2016.

[9] A. L. Krassmann, A. Falcade, G. Bernardi, R. D. Medina, *et al.*, “Exploring student’s motivational aspects by developing and applying a ubiquitous digital serious game approach,” *Creative Education*, vol. 8, no. 3, pp. 405–430, 2017.

[10] L. N. Paschoal, M. M. Oliveira, and P. M. M. Chicon, “A chatterbot sensitive to student’s context to help on software engineering education,” in *Latin American Computer Conference (CLEI)*, pp. 839–848, 2018.

[11] C. C. Possobom, A. R. K. Mühlbeier, F. B. Nunes, A. Carvalho, R. B. Gomes, and R. D. Medina, “Uma aplicação dinâmica para detectar o nível de conhecimento do aluno (in portuguese),” in *Ciclo de Palestras Novas Tecnologias na Educação*, pp. 01–10, 2014.

[12] T. Mitamura, Y. Suzuki, and T. Oohori, “Serious games for learning programming languages,” in *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1812–1817, 2012.

[13] T. R. Silva, T. J. Medeiros, and E. H. da Silva Aranha, “Jogos digitais para ensino e aprendizagem de programação: uma revisão sistemática da literatura (in portuguese),” in *Brazilian Symposium on Computers in Education*, pp. 692–701, 2014.

[14] B. Gonçalves, M. C. Moraes, B. Teixeira, and L. Lopes Leite, “Codex - learning object to support loop structure teaching,” in *Latin American Conference on Learning Technologies*, pp. 329–334, 2018.

[15] R. L. Sampaio and A. C. dos Santos Souza, “Uma experiência com objetos digitais de aprendizagem no ensino da lógica de programação (in portuguese),” in *Congresso Iberoamericano de Informática Educativa*, pp. 1–4, 2008.

[16] C. Cechinel, A. V. Silveira, R. S. Silveira, Érico de Morais Nunes, A. A. Moreira, G. B. Cogo, C. M. Betemps, and R. da Nóbrega Tavares, “Desenvolvimento de objetos de aprendizagem para o apoio à disciplina de algoritmos e programação (in portuguese),” in *Brazilian Symposium on Computers in Education*, pp. 1–10, 2008.

[17] I. B. Saraiva and C. M. Netto, “Monitor: um conjunto de objetos de aprendizagem para apoio ao ensino de programação de computadores (in portuguese),” in *Congresso da Sociedade Brasileira de Computação*, pp. 1–10, 2010.

[18] M. S. Henrique and A. D. D. S. Rebouças, “Learning objects to aid the teaching process of objetct-oriented programming (in portuguese),” *RENOTE–Revista Novas Tecnologias na Educação*, vol. 13, no. 2, pp. 1–10, 2015.

[19] A. Marcolino and E. F. Barbosa, “Softwares educacionais para o ensino de programação: Um mapeamento sistemático,” in *Brazilian Symposium on Computers in Education*, pp. 190–199, 2015.

[20] A. Luxton-Reilly, Simon, I. Albluwi, B. A. Becker, M. Giannakos, A. N. Kumar, L. Ott, J. Paterson, M. J. Scott, J. Sheard, and C. Szabo, “Introductory programming: A systematic literature review,” in *Annual ACM Conference on Innovation and Technology in Computer Science Education*, p. 55–106, 2018.

[21] M. Grandl, M. Ebner, W. Slany, and S. Janisch, “It’s in your pocket: a mooc about programming for kids and the role of oer in teaching and learning contexts,” in *Conference proceeding open educational global conference*, pp. 1–9, 2018.

[22] F. B. Nunes, F. Herpich, G. B. Voss, J. V. De Lima, and R. D. Medina, “An adaptive environment based on moodle with treating of quality of context,” *International Journal of Knowledge and Learning*, vol. 10, no. 2, pp. 198–221, 2015.

[23] F. B. Nunes, F. Herpich, G. B. Voss, and R. D. Medina, “A perspective about the application of quality of context in u-learning environments,” in *Handbook of Research on 3-D Virtual Environments and Hypermedia for Ubiquitous Learning* (F. M. M. Neto, R. C. de Souza, and A. S. Gomes, eds.), pp. 410–432, IGI Global, 2016.

[24] F. Shull, J. Carver, and G. H. Travassos, “An empirical methodology for introducing software processes,” *ACM SIGSOFT Software Engineering Notes*, vol. 26, no. 5, pp. 288–296, 2001.

[25] A. S. Marcolino and E. Barbosa, “A survey on problems related to the teaching of programming in brazilian educational institutions,” in *IEEE Frontiers in Education Conference*, pp. 1–9, 2017.

[26] D. M. Souza, K. R. Felizardo, and E. F. Barbosa, “A systematic literature review of assessment tools for programming assignments,” in *International Conference on Software Engineering Education and Training*, pp. 147–156, 2016.