This is the final peer-reviewed accepted manuscript of:

Sara Capecchi, Michael Lodi, Violetta Lonati, and Marco Sbaraglia. 2023. Castle and Stairs to Learn Iteration: Co-designing a UMC Learning Module with Teachers. In Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023). Association for Computing Machinery, New York, NY, USA, 222–228.

The final published version is available online at: https://doi.org/10.1145/3587102.3588793

Terms of use:

Some rights reserved. The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

This item was downloaded from IRIS Università di Bologna (https://cris.unibo.it/)

When citing, please refer to the published version.

Castle and Stairs to Learn Iteration: Co-designing a UMC Learning Module with Teachers

Sara Capecchi*
Dept. of Computer Science
University of Turin
Turin, Italy

Michael Lodi*
Dept. of Computer Science and Engineering
University of Bologna
Bologna, Italy

Violetta Lonati*
Dept. of Computer Science
University of Milan
Milan, Italy

Marco Sbaraglia*
Dept. of Computer Science and Engineering
University of Bologna
Bologna, Italy

ABSTRACT

This experience report presents a participatory process that involved primary school teachers and computer science education researchers. The objective of the process was to co-design a learning module to teach iteration to second graders using a visual programming environment and based on the Use-Modify-Create methodology. The co-designed learning module was piloted with three second-grade classes. We experienced that sharing and reconciling the different perspectives of researchers and teachers was doubly effective. On the one hand, it improved the quality of the resulting learning module; on the other hand, it constituted a very significant professional development opportunity for both teachers and researchers. We describe the co-designed learning module, discuss the most significant hinges in the process that led to such a product, and reflect on the lessons learned.

CCS CONCEPTS

• **Social and professional topics → K-12 education**.

KEYWORDS

participatory design, UMC, iteration, block-based programming

ACM Reference Format:
Sara Capecchi, Michael Lodi, Violetta Lonati, and Marco Sbaraglia. 2023. Castle and Stairs to Learn Iteration: Co-designing a UMC Learning Module with Teachers. In *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023), July 8–12, 2023, Turku, Finland*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3587102.3588793

1 INTRODUCTION

Despite the important role it has recently gained in public debate and policy, computer science (CS) education in primary school is still not compulsory in Italy, and primary school teachers' background does not usually include any specific education in CS, or CS teaching. However, several initiatives—often promoted by universities, associations, or teacher networks—have been launched that offer teaching materials, learning platforms, suggestions to schools and teachers on curricula, and professional development opportunities. Among those, two are particularly relevant to the experience we report in this paper: the Proposal for a national Informatics curriculum in the Italian school [11, 18] developed by CINI, a national consortium of academic CS organizations, and the Programma Il Futuro project [7], the Italian localization of Code.org [5], that provides a support website and training initiatives for teachers, with over 3 million students and 41,000 teachers involved since 2014[1].

Following up on these initiatives, the INFO-DIDA project [19] was recently funded to disseminate programming education in the early grades of primary school. The project aims to investigate the effectiveness and feasibility of different pedagogical approaches to teaching programming in the Italian school context, focusing in particular on iteration—a fundamental concept in introductory programming. As a preliminary step of the project, a learning module was developed based on the Use-Modify-Create (UMC) methodology [16] (see 2.2). The learning module (from now on referred to as the 'Module') is aimed at teaching iteration to second graders.

The topic, target learners, methodology, and programming environment for the Module were established in the context of the INFO-DIDA project [19]. In particular, the Code.org platform was chosen because it is already well known by Italian teachers through the popular Programma Il Futuro initiative.

The Module was developed by involving teachers with participatory design, an approach that actively involves stakeholders in the design process to ensure that its outcomes are usable and meet their needs [2]. This paper's objective is to describe the participatory process carried out with the teachers and to analyze how such a process positively influenced its outcome (i.e., the co-designed Module). We will highlight the positive effects of the participatory process (e.g., valuable training moments for teachers and researchers, resulting teaching materials that teachers were able—and willing—to use autonomously) and discuss what can be improved (e.g., more structuring and facilitation, better-defined roles in the process).

The report is organized as follows. Section 2 discusses the relevant literature on participatory design and UMC-related teaching methodologies. Section 3 describes the Module developed during

*Also Laboratorio CINI "Informatica e Scuola"

ITiCSE 2023, July 8–12, 2023, Turku, Finland
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM. This is the author's version of the work. It is posted here for your personal use. Not for redistribution. The definitive Version of Record was published in *Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023), July 8–12, 2023, Turku, Finland*, https://doi.org/10.1145/3587102.3588793.

[1] https://programmailfuturo.it/

the participatory process and the related material. Section 4 discusses the most significant hinges in the co-design process. Finally, Section 5 reflects on the lessons learned.

2 BACKGROUND AND RELATED WORKS

2.1 Participatory design with teachers

To develop the Module, we involved a group of teachers using a *participatory design* approach, which includes the perspective of users and stakeholders within the design process of the products they will use [1]. This approach was originally developed in the context of urban planning and applied in other fields such as software development and product design [2]. Participatory design utilizes methods that support mutual learning among participants to cooperatively design contextually relevant solutions. The process is usually supported by a *facilitator*, who works to make meetings and group interactions easier and more effective; facilitators are neutral in terms of content, their focus being the process itself.

Druin [9] analyzed the ways in which users can take part in design processes and defined a framework with a set of roles: users, testers, informants, and design partners. *Users* are the main audience for an existing technology/artifact; their use of the artifact can be investigated in order to improve it. *Testers* use an artifact that is not yet released for commercial use with the aim of developing it for a larger audience. *Informants* play an active part throughout the design process and provide input before, during, and after the product is developed. *Partners* are acknowledged as legitimate decision-makers and promoted to a role that equals that of the designers and researchers.

Participatory design methods such as *co-design* have been applied in educational contexts as well. In [21], Roschelle and Penuel analyze the key components of co-design, defined as "a highly-facilitated, team-based process in which teachers, researchers, and developers work together in defined roles to design an educational innovation, realize the design in one or more prototypes, and evaluate each prototype's significance for addressing a concrete educational need" [21, p. 606]. For a recent review of participatory design studies involving teachers, see [26].

In computing education, participatory methods are not frequently used. Notable recent exceptions are [6, 20], where co-design was used with teachers and students to integrate their perspective in their future educational material/approaches, with a particular focus on the cultural relevance of the curriculum. In particular, the work from Coenraad and colleagues [6] is relevant to our work, being about the design of a curriculum that uses the UMC methodology with Scratch.

2.2 UMC teaching methodologies

Lee and colleagues [16] proposed *Use-Modify-Create* (UMC) as a way to engage young learners in the development of Computational Thinking (CT). UMC is a three-stage progression in which students interact increasingly with computational artifacts.

In the first stage, students *Use* an artifact created by someone else (e.g., play a computer game or a simulation). In the second stage, students begin to *Modify* that artifact (e.g., starting from cosmetic changes and moving towards behavioral changes). Iteratively, they begin to *Create* their own (computational) artifacts. The three stages are not rigidly separated, with students going back and forth several times. Moving along the stages "requires increasing levels of abstract representation and understanding" and gives students an "increasing ownership of their learning" [16]. Lee and colleagues [15] propose different ways to introduce computational thinking in K-8: from puzzles to open sandbox; data generation, analysis, and exploration; UMC for computational science investigations. Lytle and colleagues [17] compared a middle school UMC course with a "create-first" course on science simulations with block programming. UMC group pupils' perceived the activities to be easier and felt greater ownership of the work. UMC group teachers' gained confidence while delivering the UMC sequence, requiring less and less support from researchers and even managing to add new tasks, while the other group teachers' felt the need for more scaffolding.

Various UMC specializations have been proposed. PRIMM [23] is a CS-specific model for teaching text-based programming in secondary school. Based on code-comprehension research, PRIMM expands the *Use* stage into more scaffolded phases: *Predict* (read the code and predict how (part of) the program will work before executing it), *Run*, and *Investigate* (analyse the program structure with program comprehension tasks at different levels of abstraction). The subsequent, more open stages are also scaffolded: *Modify* requests are narrow scope, and *Make* assignments have explicit and limited objectives. Although PRIMM is targeted at older students and text-based programming, it inspired us to use program comprehension tasks (e.g., *Predict* or *Investigate* tasks to help students understand the relationship between programming blocks and output) and narrow-scope *Modify* requests.

To shed light on student learning during the *Use* and *Modify* stages, Franklin and colleagues [12] conducted an extensive study on 536 students (age 9-14) on a Scratch-based UMC curriculum, *Scratch Encore*, which attempts to balance structure with flexibility and creativity. Students engaged in UMC activities following the TIPP&SEE approach, which provides more scaffolding to the UMC stages. TIPP&SEE's structure is explicitly based on the studies on reading comprehension in natural languages (especially on 'previewing' and 'text structure' constructs). Like PRIMM, it provides more scaffolding in the *Use* and *Modify* stages to benefit struggling students while creating an optimal experience (not too boring, not too overwhelming). Salac and colleagues [22] conducted another study that showed that students who followed TIPP&SEE performed significantly better on all intermediate and difficult assessment questions. Unlike PRIMM, TIPP&SEE is designed for Scratch, making it an implementation of UMC for young students with no programming experience. Both PRIMM and TIPP&SEE methodologies have been tested in different contexts [e.g., 13, 24], proving engaging, successful also for low-achieving students, and appreciated by teachers.

While some curricula based on TIPP&SEE [10, 27] are in line with our goals, we could not simply translate them into Italian for two main reasons. First, they are very "Scratch-centric" (whereas our constraints required us to use Code.org, see Section 3), and secondly, the storytelling needed adaptation to our cultural context.

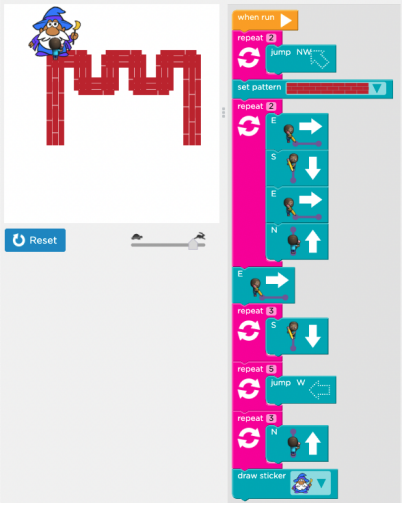

Figure 1: The program drawing the tower with the wizard.

3 THE LEARNING MODULE

In this section, we present the Module resulting from the co-design process. All material, anonymized and translated in English, is available at [14]. In Section 4 we will analyze how the process influenced this result.

The Module aims at introducing second graders to the programming iteration construct in a visual programming environment—namely in Code.org Studio—using the Use-Modify-Create methodology. The Module's learning goals are consistent with the Proposal for a national Informatics curriculum in the Italian school [11] suggesting that pupils should be able to "use loops to concisely express that a certain action has to be repeatedly executed a prefixed number of times".

As prior knowledge, we assume that pupils can build programs formed by a sequence of instructions, knowing that such instructions will be executed in the order they occur in the program. We also assume that they are familiar with Code.org Studio and the blocks of *Artist (Pre-Reader) lab* [4]. The prerequisites and learning outcomes for the Module, w.r.t. the CSTA K–12 CS Standards [8], are better articulated in [14].

3.1 Classroom activities

The Module is designed according to the UMC principles and structure and consists of two 2-hour lessons. Three purposely prepared Code.org programs [14] are available for pupils to run, look into, and modify. The first of such programs is shown in Figure 1. In each lesson, tasks are proposed, and questions are asked that guide the pupils to become familiar with the programs, understand their behavior, modify them, and finally learn how to use the iteration construct.

The first lesson uses the first program and focuses mainly on the *Use* and *Modify* stages. In the *Use* stage, pupils do not have access to the program's source code. They are asked to run the program, observe its behavior and the effect of its execution (e.g., "what did the artist draw?"), and reflect on what happened (e.g., "which part of the drawing was drawn first?"). They are invited to run it as often as they like, possibly changing the execution speed.

In the *Modify* stage, pupils see the source code; this stage aims at making them read and explore the blocks of the program. We propose three types of *Modify* tasks.

(i) *Syntactical change.* Under direct instruction, pupils are asked to make a specified change in the program (e.g., move or replace a block, modify the parameter in the repeat block), then run it, observe, and verbalize the effect of the change on the execution and/or resulting drawing.

(ii) *Prediction.* The teacher describes a specific program change (as in the previous task), and pupils are asked to predict how this change will affect the execution and/or resulting drawing before actually modifying the program.

(iii) *Intentional modification.* Pupils are given a simple, specific objective, such as modifying the program behavior or getting a drawing variation (e.g., "make the castle tower taller"), and are encouraged to take a trial-and-error approach. They can edit and run the program as often as they want; if necessary, they can discard their current program and start over from the original one.

All the modifications in the first lesson concern the simplest loops, that is, those defined as a `repeat` block with only one instruction. At the end of the first lesson, pupils are invited to modify the program, e.g., by adding characters or simple elements.

The second lesson uses two other programs, each producing a variant of the first drawing. For both programs, the *Use* stage is limited and aimed only at getting pupils familiar with the new program, while the *Modify* stage focuses on stair-drawing loops that contain two-instruction sequences. The second lesson then includes a substantial *Create* stage. Pupils are invited to make their own drawings by building a program using the `repeat` block. Some drawing ideas are available to possibly inspire pupils who do not know what to draw or are stuck with too complex and unfeasible drawings.

3.2 Developed materials

Programs. Three Code.org programs were developed purposely for the Module; see [14]. They produce drawings of a castle tower with additional characters and stairs; the drawings are similar but obtained differently (e.g., starting from a different corner of the castle). Figure 1 shows the program that is proposed to pupils first.

The programs are built using the *Artist (Pre-Reader) lab* [4]. Each contains multiple (five or six) `repeat` blocks; some loops include only one other block, while others are more complex. In particular, all programs include a `repeat` block containing a sequence of four movement blocks used to draw the tower battlements.

Support slides. We prepared a series of slides [14] to support teachers in conducting the lessons. The slides collect all the questions and tasks proposed to pupils and are to be displayed at the appropriate moment during the lessons. Slides mainly use visual language and include very little textual content; see, e.g., Figure 2. Pupils work in pairs following the indications given orally by the teacher, with the visual support of the slides.

Besides the slides with questions and instructions, there are also some "checkpoint slides" that serve to align the class at critical

moments (see the slide to the right in Figure 2). For instance, some checkpoint slides show the effect of some specific change in the program; they allow pupils to verify that they got the same effect or help them catch up if they had difficulties understanding or executing the questions/instructions. The checkpoint slides also allow the teacher to discuss or clarify crucial points, starting with pupils' answers and comments. The checkpoint slides are intended to allow pupils to proceed at their own pace, while maintaining an overall pace for the whole class. Hence, they can help the class and the teacher keep the focus on essential points, avoid dispersion and confusion, and still guarantee a certain level of autonomy for the pupils.

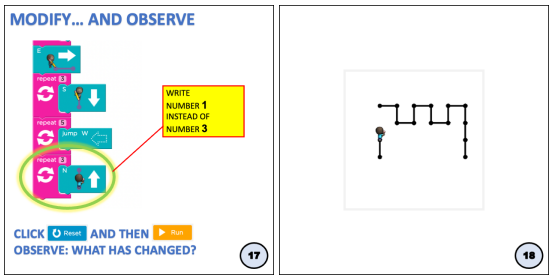

Figure 2: Examples of support slides. The slide on the right is used as checkpoint w.r.t. to the task posed in the slide on the left.

Teacher's guide. All classroom activities in the Module are presented and commented on in a 'Teacher's Guide' [14]. Although teachers could also use the guide as a reference during the lessons, its purpose is to support them in preparing for the lessons. The guide is threefold. First, it presents the pedagogical approach used in the Module, illustrating the three stages of the UMC methodology. Second, it provides a detailed description (consistent with the slides' content) of the series of tasks/questions to be proposed to the pupils in the two lessons. Last, it states the intended purpose of the proposed tasks/questions, anticipates possible reactions, comments, doubts, and mistakes that pupils might manifest, pinpoints the critical role of some specific tasks/questions in the learning process, and frames the tasks/questions in a general learning context.

4 CO-DESIGNING WITH TEACHERS

This section describes the participatory process we conducted, and how it affected its outcome, i.e., the Module described above.

4.1 Methods and participants

The participatory process involved four CS education researchers from three different universities (R1, R2, R3, R4—the authors of this paper), and four primary school teachers (T1, T2, T3, T4).

First, the group of researchers had a few meetings to draft some preliminary ideas of Code.org Studio programs and tasks for the Module, structured according to the UMC methodology. The programs were designed to be short enough for pupils to handle, rich enough to provide interesting and attractive behavior, and complex enough to challenge pupils' understanding and cognitive processes.

Rather than simply asking teachers for separate feedback on the preliminary material, e.g., through an online questionnaire, we promoted a participatory process and invited a diverse group of primary school teachers to act as *informants* (see Section 2.1). The design ideas drafted by the group of researchers would be submitted for open discussion. We opened up for a comprehensive review and revision of the material, where teachers would have the chance to interact with both the researchers and each other.

More precisely, we reached out to primary school teachers we had already worked with for other STEM projects. Four teachers from four different cities accepted our invitation to collaborate. They did not know each other; all had a strong motivation and long teaching experience; their background w.r.t. programming ranged from novice to expert, as well as their experience in teaching computer science. This collaboration lasted about two months and consisted of four online meetings (attended by both researchers and teachers), some intermediate meetings between researchers only, and continuous asynchronous work on the shared documents (by individual participants or small subgroups). One of the researchers acted as a facilitator of the process.

The co-design process ended with three of the four teachers piloting the Module in their classes with the related material. Since the teachers were overall very satisfied with the pilots, the Module was finalized by implementing just a few minor corrections.

Overall, about 60 pupils were involved in the pilots. We did not collect data on pupil performance or behavior, nor did the teachers use the learning experience to assess the pupils. The teachers were interviewed briefly after the completion of their pilots. All meetings with the teachers and the final interviews were conducted online and recorded. In addition, several documents were shared, collaboratively edited and commented on during the process. To write this report, we watched the recordings and transcribed the most relevant passages, then reviewed all the transcripts, together with the comments and history of the shared documents. All involved people were informed and agreed to be recorded for these purposes.

4.2 How the process affected the outcome

The programs and tasks that compose the Module presented in Section 3 are drastically different from those initially designed by the researchers and submitted to the teachers at the beginning of the participatory process. The comments and objections raised by the teachers led to fruitful discussions. In what follows, we report some concrete episodes that we deem both representative of the type of interactions that occurred and meaningful for their impact on the process outcome.

Overall, the discussion among researchers and teachers led to:

- confirmation and clarification of core ideas (the UMC methodology and the CS content of the Module);
- substantial change of some aspects of the Module content (storytelling and visual aspects);
- introduction of new ideas, especially concerning the presentation of the material and the teacher's role when conducting the activities (use of checkpoint slides, content and purpose of the teacher's guide);
- revision of linguistic aspects (wording of tasks and questions).

We organize the remaining part of the section along five main themes for the sake of readability, but it is worth mentioning that, as they are strongly interconnected, most of them were touched over in more than one meeting and occurred interleaved. Teachers and researchers are named respectively using the letters T and R.

Storytelling and visual aspects. The initial comments mostly focused on the setting for the programs to be proposed in the classroom, as the drafted programs produced simple drawings unrelated to each other. For example, T3 noted the importance of the storytelling and visual aspects, emphasizing that these also increase the effectiveness of the activities from a learning perspective; T2 pointed out the difficulty of devising programs with these visual and narrative enjoyable features and whose complexity is still manageable by young children. After several interactions, the initial programs were dropped. They were replaced by three new programs (those described in Section 3) with similar complexity that, however, refer to one single story with two characters (a wizard and a dragon) moving around a castle tower (see Figure 1).

CS content and relationship with other disciplines. When preparing the first draft of the learning material, the researchers mainly focused on its CS content. Since the first interactions with teachers, it was clear that this focus would likely conflict with other points of view that do not concern CS, but are very relevant, especially in early education.

For instance, when learning to write the parts that make up a letter, children are encouraged to move the pencil in a prescribed sequence and direction. However, in one of our initial programs, the artist draws a letter following a purposely scrambled sequence and directions so that the relation between the program blocks and their effect would be less predictable. Although this trick seemed justifiable—and even useful—from the CS point of view, it was challenged by T3 and finally deemed unacceptable by the teachers, who know that many second graders still have difficulty writing letters in the correct direction, and should not be presented with contradictory examples. Similar observations were made by T1 and T4 regarding difficulties with laterality, as some tasks relied on the ability to distinguish right from left or similar topological relationships that may still be under acquisition for some second graders. The revision mentioned above of the overall setting for the programs was also strongly affected by these discussions.

Presentation of questions and tasks: wording and format. A large part of the discussion concerned how to present tasks (including questions to be answered) so that pupils would easily understand them. The researchers had drafted a written document with preliminary notes describing the tasks to propose to pupils. Such a document was intended as a tool for discussion; thus, we deliberately omitted to work on refining its wording. Instead, we explicitly left this matter to be explored in the co-design process and asked the teachers to help us shape the format and linguistic aspects of the tasks and questions.

Indeed, all teachers found in the text several hurdles for pupils, such as difficult terms or long sentences. They also found imprecise or misleading wording and, while raising their doubts, offered the researchers the chance to clarify the intended purpose of the tasks from a cognitive point of view. This discussion turned out to be very formative for the teachers, who had the opportunity to understand better some aspects of CS involved in both the programs and the questions.

As for the format, the initial version relied very much on verbal communication, the main question from the researchers being whether the tasks should be presented to pupils in written form (e.g., with a worksheet, as in [22]) or explained orally. The teachers brought up a different approach strongly based on visual communication, as they observed that relying only on words is inappropriate for second graders. In particular, T3 suggested using colors and arrows to identify parts of the screen or portions of a program; T4 went further and prepared some slides to show how to present the tasks in a visual format. From then on, instead of acting only as an informant, T4 played a much more active role in preparing the slides and the teacher's guide, working closely with R1 and R2 as a *partner* in the process (see Section 2.1). Figure 2 shows an example of the resulting slides.

Class management and the teachers' role. The Module is equipped with clear guidance for the teachers about their role in managing the class during the activities. However, in the first phase, the researchers deliberately omitted to define this role precisely, beyond having the tasks structured and organized according to the UMC methodology. On the contrary, this issue was explicitly left to be explored in the co-design process, aiming at a balance that would let each pupil explore and reflect on the questions while avoiding dispersion and confusion.

The discussion was initiated by asking teachers how they would conduct the activities. T3 noted that the answers to that question did not depend only on the activities as such but on many other factors as well, including the characteristics of the pupils' group (e.g., listening skills, habit with open-ended problems), and the teachers' personal style in conducting any activity. T4 and T3 agreed that all pupils should be offered the opportunity to explore, thus suggesting that work in pairs or small groups should be encouraged. R2 and R4 were concerned that the teachers might have difficulties managing the class because of different group paces.

We then sketched out the following scheme of work. Pupils are set in pairs; the teacher explains the next task to the whole class by using the related support slide; pupils work, reflect, and discuss in pairs; after a while, the teacher asks the class what answers they found and helps them discuss, trying to include all groups. T1 and T2 raised some concerns: the discussions might take too much time and be annoying, and the resulting pace may not be respectful to the slower pupils. At this point in the co-design process, the idea of using checkpoint slides (see Section 3.2) as a tool to balance the pupils' freedom to follow their own pace while preserving the general pace of the class emerged.

The discussion also led to another significant decision: we agreed on the importance of providing a solid background to support teachers in conducting the Module. Therefore, we decided that the teacher's guide would contain not only a detailed description of the series of tasks and questions but also some crucial reflections on their purpose and meaning. At the same time, the guide would also recommend teachers consider possible adaptations of the Module (e.g., timing, additional details to provide when proposing tasks, use

of specific methods to conduct and promote discussion during the activities), taking into account the characteristics of their classes.

Foster reflection in pupils. The teachers learned about the UMC methodology from the researchers and appreciated it. R2 emphasized that the approach allows for more complex programs, balancing the greater difficulty with increased involvement of pupils; T3 noticed that the *Create* stage motivates learners and related the *Modify* stage to experiential learning. T4 remarked that the approach accommodates different levels of expertise, also involving pupils with prior extracurricular experience in coding. Several comments focused on the effectiveness of the UMC methodology in fostering reflection in pupils, whose trial-and-error style is often too impulsive and little principled. For instance, the tasks in the *Use* stage make pupils observe important details and reflect on what happened; the prediction questions in the *Modify* stage force them to pause to think and reason about the expected outcomes.

However, T4 noted that such "reflective" open-ended questions (e.g., "what do you notice?") are difficult for young children because they require them to give explanations, which is a hard task at their age. From a learning perspective, keeping things practical and concrete is fundamental. A question that is not "answerable" is bad both for the pupil—who does not know how to deal with it—and for the teacher—who cannot understand whether the pupil has grasped the point. This concern was addressed by rewording several questions, paying more attention to the cognitive progression of tasks, and adding checkpoint slides to help pupils check their reasoning without the need for verbose explanations.

4.3 Possible improvements

Even though external constraints or lack of resources often hinder this, the best practices in participatory processes suggest that the process should be planned thoroughly in advance, use structured participatory methods, and be facilitated by external experts not involved in the design itself [3].

In our case, we turned to co-design only after a preliminary design phase; hence we did not plan the participatory process in detail, and we conducted it in a short time, with close meetings. The process was facilitated informally, with one of the researchers acting as facilitator, based on her prior expertise in participatory design. Moreover, as we did not have the time to organize preliminary forms of infrastructure and "backstage work" to create rapport and trust between participants [3], we could only include in the process teachers with whom we were already in contact. This resulted in a high ratio between researchers and teachers, which however can and should be reduced, so that a larger number of teachers can enjoy the professional development opportunity, while still preserving the contribution from both kinds of expertise.

All the above aspects should be better addressed in future similar initiatives. Despite this, the participatory principles were attended to. In particular, we cared to create a context that truly allowed an open discussion and establish equitable power relations within the co-design group, as testified by the fact that T4 was unexpectedly promoted as a *partner* in the design process.

5 CONCLUSIONS

In this paper, we reported the process of co-designing a learning module aimed at teaching iteration to second graders with the UMC methodology in a block-based programming environment. Consistently with what was reported by other UMC research [e.g., 17], the process successfully produced teaching material that the teachers felt comfortable experimenting with in their classes without the researchers' support. After the pilot conducted by the teachers involved in the participatory process, the Module was used by 56 other teachers for the INFO-DIDA project [19]. Even though it is still ongoing, the preliminary results show that the teachers were able to use the Module's material confidently in their classes.

The heterogeneous co-design context (that included primary school teachers and CS researchers) helped to hold different educational needs together. The presence of diverse specific expertise guaranteed that fundamental aspects and objectives of such different expertise were correctly preserved. For example, this allowed us to find a satisfying mediation w.r.t. different—and sometimes contrasting—needs (e.g., balancing ease, economy, and length of the activity while keeping essential tasks to make students focus on specific CS aspects). This resulted in an educational product that maintains its CS scientific soundness while being usable by teachers and appropriate for second-grade students.

Crucial for the design's success were the ample space granted to the teachers' peer discussion, the researchers' openness to listen and—when needed—heavily revise the work (while always being firm on crucial CS concepts), and the teachers' generous participation in the process.

As illustrated in the analysis, the co-design process has been very formative for both the teachers and researchers. Teachers had a better understanding of critical cognitive aspects of teaching iteration, by diving into the material to understand it and looking for the best way to engage students. Researchers learned to better take into account pedagogical aspects not strictly related to teaching CS. Most importantly, while maintaining our constructivist orientation, we moved from a minimal-guidance proposal to a more scaffolded one, creating materials that can help teachers provide pupils with the optimal guidance they need [25].

Overall, we advocate for such participatory design methods as a helpful approach to both increase the quality of the resulting learning material and contribute to the professional development of the people involved. We hope this report will contribute to promoting wider adoption of such methods by the CS education community.

ACKNOWLEDGMENTS

Thanks to the teachers and the students participating in the co-design process and pilots. Work partially supported by the MUR grant PANN20_00690 to the CINI National Lab "Informatica e Scuola". M. Lodi's work has been supported by the Spoke 1 "FutureHPC & BigData" of the Italian Research Center on High Performance Computing, Big Data and Quantum Computing funded by MUR M4C2 Investimento 1.4: Potenziamento strutture di ricerca e creazione di campioni nazionali di R&S (M4C2-19)-Next Generation EU (NGEU).

REFERENCES

[1] Gro Bjerknes and Tone Bratteteig. 1995. User Participation and Democracy: A Discussion of Scandinavian Research on System Development. *Scandinavian Journal of Information Systems* 7, 1 (Jan. 1995). https://aisel.aisnet.org/sjis/vol7/iss1/1

[2] Susanne Bødker, Pelle Ehn, Dan Sjögren, and Yngve Sundblad. 2000. Co-operative Design—perspectives on 20 years with 'the Scandinavian IT Design Model'. In *proceedings of NordiCHI*, Vol. 2000. 22–24.

[3] Susanne Bødker, Christian Dindler, and Ole Sejer Iversen. 2017. Tying Knots: Participatory Infrastructuring at Work. *Computer Supported Cooperative Work (CSCW)* 26, 1-2 (April 2017), 245–273. https://doi.org/10.1007/s10606-017-9268-y

[4] Code.org. n.d. *Artist (pre-reader) lab.* https://studio.code.org/projects/artist_k1/new

[5] Code.org. n.d. *Code.org.* https://code.org

[6] Merijke Coenraad, Jen Palmer, Donna Eatinger, David Weintrop, and Diana Franklin. 2022. Using participatory design to integrate stakeholder voices in the creation of a culturally relevant computing curriculum. *International Journal of Child-Computer Interaction* 31 (2022), 100353. https://doi.org/10.1016/j.ijcci.2021.100353

[7] Isabella Corradini, Michael Lodi, and Enrico Nardelli. 2017. Computational Thinking in Italian Schools: Quantitative Data and Teachers' Sentiment Analysis after Two Years of "Programma Il Futuro". In *Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education* (Bologna, Italy) *(ITiCSE '17)*. Association for Computing Machinery, New York, NY, USA, 224–229. https://doi.org/10.1145/3059009.3059040

[8] CSTA. 2017. *CSTA K-12 Computer Science Standards, rev. 2017.* Technical Report. Computer Science Teachers Association. http://www.csteachers.org/standards

[9] Allison Druin. 2002. The role of children in the design of new technology. *Behaviour & Information Technology* 21, 1 (Jan. 2002), 1–25. https://doi.org/10.1080/01449290110108659

[10] ECforALL. n.d. *Act 1 Curriculum.* https://impactconectar.wixsite.com/website/act-1

[11] Luca Forlizzi, Michael Lodi, Violetta Lonati, Claudio Mirolo, Mattia Monga, Alberto Montresor, Anna Morpurgo, and Enrico Nardelli. 2018. A Core Informatics Curriculum for Italian Compulsory Education. In *Lecture Notes in Computer Science*. Springer International Publishing, 141–153. https://doi.org/10.1007/978-3-030-02750-6_11

[12] Diana Franklin, Merijke Coenraad, Jennifer Palmer, Donna Eatinger, Anna Zipp, Marco Anaya, Max White, Hoang Pham, Ozan Gökdemir, and David Weintrop. 2020. An Analysis of Use-Modify-Create Pedagogical Approach's Success in Balancing Structure and Student Agency. In *Proceedings of the 2020 ACM Conference on International Computing Education Research* (Virtual Event, New Zealand) *(ICER '20)*. Association for Computing Machinery, New York, NY, USA, 14–24. https://doi.org/10.1145/3372782.3406256

[13] Diana Franklin, Jean Salac, Zachary Crenshaw, Saranya Turimella, Zipporah Klain, Marco Anaya, and Cathy Thomas. 2020. Exploring Student Behavior Using the TIPP&SEE Learning Strategy. In *Proceedings of the 2020 ACM Conference on International Computing Education Research* (Virtual Event, New Zealand) *(ICER '20)*. Association for Computing Machinery, New York, NY, USA, 91–101. https://doi.org/10.1145/3372782.3406257

[14] INFO-DIDA Project. 2022. *UMC materials.* https://infodidaumc.web.app/

[15] Irene Lee, Fred Martin, and Katie Apone. 2014. Integrating Computational Thinking across the K–8 Curriculum. *ACM Inroads* 5, 4 (dec 2014), 64–71. https://doi.org/10.1145/2684721.2684736

[16] Irene Lee, Fred Martin, Jill Denner, Bob Coulter, Walter Allan, Jeri Erickson, Joyce Malyn-Smith, and Linda Werner. 2011. Computational Thinking for Youth in Practice. *ACM Inroads* 2, 1 (feb 2011), 32–37. https://doi.org/10.1145/1929887.1929902

[17] Nicholas Lytle, Veronica Cateté, Danielle Boulden, Yihuan Dong, Jennifer Houchins, Alexandra Milliken, Amy Isvik, Dolly Bounajim, Eric Wiebe, and Tiffany Barnes. 2019. Use, Modify, Create: Comparing Computational Thinking Lesson Progressions for STEM Classes. In *Proceedings of the 2019 ACM Conference on Innovation and Technology in Computer Science Education* (Aberdeen, Scotland Uk) *(ITiCSE '19)*. Association for Computing Machinery, New York, NY, USA, 395–401. https://doi.org/10.1145/3304221.3319786

[18] Enrico Nardelli, Luca Forlizzi, Michael Lodi, Violetta Lonati, Claudio Mirolo, Mattia Monga, Alberto Montresor, and Anna Morpurgo. 2017. *Proposal for a national Informatics curriculum in the Italian school.* Technical Report. CINI. ProposalforanationalInformaticscurriculumintheItalianschool

[19] Enrico Nardelli, Francesco Lacchia, Renzo Davoli, Michael Lodi, Marco Sbaraglia, Veronica Rossano, Enrichetta Gentile, Violetta Lonati, Mattia Monga, Anna Morpurgo, Luca Forlizzi, Giovanna Melideo, Sara Capecchi, Ilenia Fronza, and Tullio Vardanega. 2023. Learning Iteration for Grades 2-3: Puzzles vs. UMC in Code.Org. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V. 2* (Toronto ON, Canada) *(SIGCSE 2023)*. Association for Computing Machinery, New York, NY, USA, 1368. https://doi.org/10.1145/3545947.3576312

[20] Elena Novak and Javed Khan. 2022. A Research-Practice Partnership Approach for Co-Designing a Culturally Responsive Computer Science Curriculum for Upper Elementary Students. *TechTrends* (apr 2022), 1–12. https://doi.org/10.1007/s11528-022-00730-z

[21] Jeremy Roschelle and William R. Penuel. 2006. Co-Design of Innovations with Teachers: Definition and Dynamics. In *Proceedings of the 7th International Conference on Learning Sciences* (Bloomington, Indiana) *(ICLS '06)*. International Society of the Learning Sciences, 606–612.

[22] Jean Salac, Cathy Thomas, Chloe Butler, Ashley Sanchez, and Diana Franklin. 2020. TIPP&SEE: A Learning Strategy to Guide Students through Use - Modify Scratch Activities. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*. Association for Computing Machinery, New York, NY, USA, 79–85. https://doi.org/10.1145/3328778.3366821

[23] Sue Sentance and Jane Waite. 2017. PRIMM: Exploring Pedagogical Approaches for Teaching Text-Based Programming in School. In *Proceedings of the 12th Workshop on Primary and Secondary Computing Education* (Nijmegen, Netherlands) *(WiPSCE '17)*. Association for Computing Machinery, New York, NY, USA, 113–114. https://doi.org/10.1145/3137065.3137084

[24] Sue Sentance, Jane Waite, and Maria Kallia. 2019. Teachers' Experiences of Using PRIMM to Teach Programming in School. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (Minneapolis, MN, USA) *(SIGCSE '19)*. Association for Computing Machinery, New York, NY, USA, 476–482. https://doi.org/10.1145/3287324.3287477

[25] Keith S. Taber. 2012. Constructivism as educational theory: Contingency in learning, and optimally guided instruction. In *Educational theory*, Hassaskhah Jaleh (Ed.). Nova, New York, 39–61.

[26] Ari Tuhkala. 2021. A systematic literature review of participatory design studies involving teachers. *European Journal of Education* 56, 4 (Dec. 2021), 641–659. https://doi.org/10.1111/ejed.12471

[27] UChicago STEM Education. n.d. Scratch Encore. https://www.canonlab.org/scratch-encore