Scaffolding Creative Programming Projects

Alexander Repenning
School of Education
PH FHNW
Windisch Switzerland
alexander.repenning@fhnw.ch

Susan Grabowski
Center for Learning Science
EPFL
Lausanne, Switzerland
susanne.grabowski@epfl.ch

ABSTRACT

While most teachers welcome the idea of learning activities fostering creativity, it is not clear how to effectively scaffold creativity. Without suitable pedagogical approaches, it is difficult to provide appropriate levels of scaffolding. Over-scaffolding, on the one hand, while providing support appreciated especially by less experienced students, leaves little room for creative expression. Under-scaffolding, on the other hand, while fostering more authentic learning opportunities with a high potential for creativity, can lead to student frustration. The Process Artifact Creativity Landscape (PACL) is a framework that helps pre-service teachers scaffold creative projects. PACL consists of a two dimensional space providing four scaffolding approaches offering different tradeoffs between scaffolding and creativity. This paper introduces the PACL framework and outlines experiences with K–6 preservice teachers reasoning about scaffold creative programming projects.

CCS CONCEPTS

• Applied computing~Education~Interactive learning environments

KEYWORDS

Computer science education, creativity, scaffolding, computational thinking, preservice teacher education

ACM Reference format:
Alexander Repenning, and Susan Grabowski. 2024 Scaffolding Creative Programming Projects, In *Proceedings of the 19th WiPSCE Conference on Primary and Secondary Computing Education Research (WiPSCE '24), 6 pages*. https://doi.org/10.1145/3677619.3677634

1 Introduction

While creativity has long been identified as a key aspiration of a 21st-century workforce and pathway to computer science [15], it is somewhat unclear how creative projects can be systematically scaffolded [36] in school. The Computer Science Principles framework [7] makes creativity the first of the seven big ideas because creative projects are known to inspire students. It is clear that teachers play an important role in scaffolding creativity but there are different theories on how. Some, e.g., [32], suggest that teachers who are creative themselves teach creatively, resulting in high potential for creative expression by students. Others, e.g., [21] have found evidence that creative teaching practice can be developed through teacher professional development.

ACM ISBN 979-8-4007-1005-6/24/09. https://doi.org/10.1145/3677619.3677634

CS teachers can, and should, scaffold creative processes but need to be aware of the delicate nature of scaffolding [33]. Hammond defines scaffolding to have teachers provide essential but temporary support to assist learners with the development of new understandings [11]. The delicate nature is due to the fact that finding an effective sweet spot of scaffolding is quite difficult. Imagine a project where students would learn computational thinking (CT) [35] by creating a Frogger-like game. *Over-scaffolding*, such as step-by-step instructions or video tutorials, likely results in students creating identical artifacts with limited creative potential. These instructions often lack rationale, hindering students' grasp of implicit design spaces. *Under-scaffolding*, conversely, may lead to students struggling. For instance, merely suggesting to make a game allows high creative potential but may cause frustration and potentially causing many students to give up.

The contribution of this paper is to introduce a framework called the *Process Artifact Creativity Landscape* (PACL) aiding inexperienced computer science pre-service teachers in conceptualizing different kinds of scaffolds for creative programming projects. PACL aims to provide various pathways to gradually guide pre-service teachers away from instructionism, towards constructionism [23] to better scaffold creativity.

2 The Process Artifact Creativity Landscape

We developed the Process Artifact Creativity Landscape (PACL), as a two dimensional continuous space (Figure 1) outlining different scaffolding strategies useful in teacher education. In Switzerland, over 2000 preservice elementary school teachers were educated in CT through Scalable Game Design [19, 26] employing PACL in the context of two different courses. In the *CT Science course* PACL was employed to teach CT through game design. In the *CT Didactics* course teachers create their own learning designs [17].

PACL is about scaffolding creative projects. Like Bloom's Taxonomy [4], PACL provides teachers a common language [18] for learning designs. Four approaches combine *same* or *different* processes to create *same* or *different* artifacts.:

1. ***Executing*** (same process/same artifact). A user can follow explicit step-by-step instructions to build a specific artifact. The main goal of Executing is to successfully create an artifact but not necessarily in a creative sense. Frequently, instructions are minimal and not optimized for learning. For instance, the instructions generally do not include explanations on why a certain step was employed nor do they suggest alternative construction paths. General Example: IKEA instructions to assemble a bookshelf. CT Example: A tutorial to build a Frogger game [25].

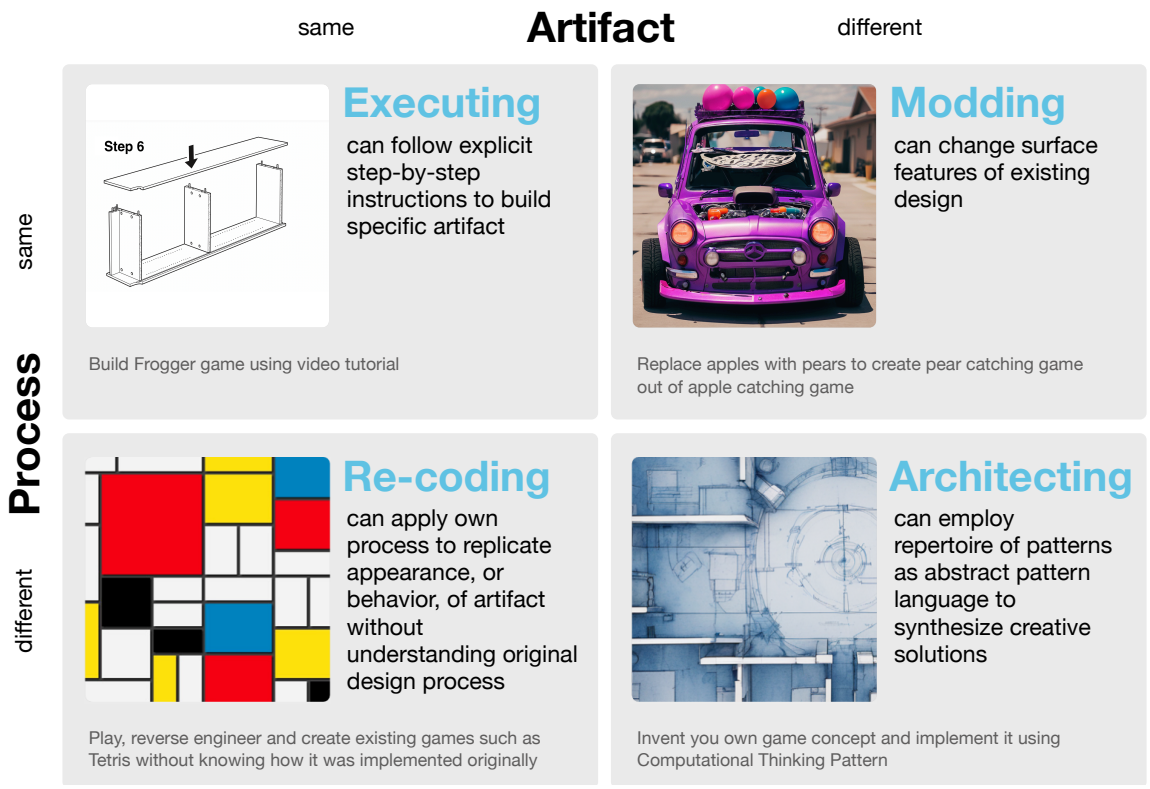

Figure 1: The Process Artifact Creativity Landscape (PACL) with examples from Computational Thinking.

2. ***Modding*** (same process/different artifact). The term modding is not defined very precisely but generally refers to the process of changing existing designs, in some typically superficial way, to adapt the design to a new purpose or need generally not anticipated by the original designer. A user can change surface features of an existing design without the need to understand or fundamentally change the underlying design or can also slightly modify existing code through direct editing or remixing [8]. Example: A well-known example of a game mod is the game "Counter-Strike" derived from the first-person shooter game "Half-Life."
3. ***Re-Coding*** (different process/same artifact). In Re-Coding [10] users try to recreate an existing artifact but, in contrast to Executing, are not provided instructions. Users may employ techniques such as reverse engineering [5]. In the context of game design users may play existing implementations of the game or watch videos describing game play. While artifacts produced in Re-Coding are not creative, processes still can be. Users may use radically different approaches (e.g., different programming paradigms, strategies, or algorithms) to achieve similar behaving games.
4. ***Architecting*** (different process/different artifact). A user can employ a rich repertoire of design patterns as design pattern language [1] to express innovative solutions. General Example: An architect creatively combines patterns using modules combining concrete, steel and wood to create a new kind of house design. CT Example: A user combines several Computational Thinking Patterns [16] to create a new kind of game. For instance, a user may combine object interaction patterns such as collision, generation, absorption, and pulling acquired through Executing by making Frogger-like and Snake-like games into a train simulation game.

Our position is that creative projects can and should be scaffolded in education using a repertoire of explicit scaffolding approaches which can be mixed and matched to fit specific learning situations. The goal is not to find a single one-size-fits-all approach but to define a continuous landscape suggesting concrete scaffolding approaches useful to everyday teaching practice. Some of these approaches, e.g., Executing may not be particularly creative, per se, but possibly helps to develop essential skills needed later for creative expression. Others, such as Architecting, may be quite ambitious and out of reach of inexperienced students. The contribution of PACL is to serve as a pedagogical framework. Where is my learning design in the landscape right now, and where could it go? The continuous nature of the landscape makes it possible to mix approaches and interpret them as new intermediate points. For example, a novice-aimed Hour of Code Frogger-like activity might inspire students to change the game's characters and narrative, gradually shifting from Executing to Modding in the PACL landscape.

PACL was first used to teach teachers Computational Thinking (CT) and then was employed by these teachers to design their own lesson plans to teach their students. In this paper we used a survey to explore which parts of the PACL landscape teachers would use and how they would combine them.

Finding the right approach, or combination of approaches, may depend on the individual users but also specific situations. To assess PACL we asked preservice elementary schools teachers which approaches they would use and how they would combine these approaches to design their own K-12 computer science education courses. The remaining sections of this paper explore related work and present the results of the teacher questionnaires.

3 Related Work

There is a vast body of literature exploring creativity including teaching guidelines [31] and instruments, such as the Torrance test, to measure it [2, 14]. Early on, creativity has been identified as a motivational pathway towards CS education [15]. The mix of CS and non-CS examples in Figure 1 is to suggest that the understanding of creativity in CS education can benefit from established creativity scaffolding strategies developed in other fields. One of the most influential conceptualizations of creativity is the 4P model developed by Rhodes [29]: Person, Process, Product and Press. PACL is the attempt of projecting the more theoretical 4D space, implied by the 4P model, onto a more concrete 2D landscape useful to learning design for teachers. Related work is organized according to the PACL.

3.1 Executing

Creative processes, such as the GenPlore model developed by Ward et al. [34], generally include generative and exploratory phases, creating, evaluating and selecting ideas. In contrast, processes merely following existing instructions to create predetermined artifacts without exploring alternatives are not considered creative. For instance, few would perceive following IKEA assembly instructions to build a bookcase, following LEGO car instructions to build the car depicted on the box, or following a cherry pie recipe to bake said pie to be particularly creative. In terms of Bloom's revised taxonomy, Executing is unlikely to reach the "create" level of the cognitive process dimension. That is, Executing is unlikely to produce new or original work. However, Executing when reaching up to the "apply" level [18], may serve as an important preparatory stepping stone resulting in foundational factual as well as procedural knowledge.

3.2 Modding

Successful modding requires the ability to draw simple connections among ideas approximately at the "analyze" level of the revised Bloom's taxonomy [3]. The process of game modding has been employed as engaging educational practice for some time [37]. In contrast to game design [12], game modding is typically focused on superficially changing existing games and not creating new games from scratch.

In computer science education some have observed complex interactions between modding and creativity. Franklin has explored modding-based scaffolding approaches providing students so-called themed starter projects [6]. She notices the challenge emerging from trying to find the appropriate level of scaffolding and reports negative impacts of Modding-based over-scaffolding. Lee outlines a scaffolding progression called Use-Modify-Create (UMC) in which the modding stage is first preceded by the use of existing projects and later followed by the create stage to successfully create simulations and games [20]. Expanding on UMC, Predict, Run, Investigate, Modify and Make (PRIMM) also connects to levels of abstraction and tracing and code comprehension research [30].

3.3 Re-Coding

Re-Coding [10] is difficult to characterize from a revised Bloom's taxonomy point of view. It transcends the "evaluate" level (second to top) but does not quite reach the "create" level (top) because, at least initially, it focuses on the replication of existing artifacts. The practice of re-coding goes back to the times where novice painters acquired painting skills by trying to imitate the works of the grand masters. Originally, imitation consisted of attempts to create exact copies of the works. But today re-coding tries also to reverse engineer [5] principles behind the artifacts with the goal to not only create imitations but, more importantly, to create original art based on the same principles. In the context of programming re-coding was first used to develop program code capable of re-creating works of early computer art and later as means to learn about programming as well as art [10]. Re-coding tries to reverse engineer [5] algorithmic thinking from artifacts. To reverse engineer a game students may play with existing implementations which they decompose into recognizable patterns such as Computational Thinking Pattern [16].

3.4 Architecting

Reaching the top of the revised Bloom's taxonomy ("create" new and original work), Architecting has huge potential to foster creativity but unless properly supported Architecting may be out of reach for inexperienced students. The challenge is to find intuitive constructs bridging expert-level design understanding with highly limited programming skills of novices [24]. These constructs need to be suited for educational purposes. Design patterns help to scaffold creative processes effectively for novices by embodying effective configurations of components that have been established by experts. Kim, et al., describe how they scaffold creative work by using storytelling patterns extracted from stories created by experts [13]. The notion of patterns has been established early in architecture as pattern language by the seminal book of Alexander [1] and later applied to software [9]. Computer science education specific patterns emerged over time including the Computational Thinking Patterns [16] embodying patterns found in common between game design and simulation building. Interdisciplinary patterns, such as the Computational Music Thinking Patterns [28], connect CS with music.

4 Implementation

66 subjects recruited from a total of 99 preservice elementary school K-6 teachers in four Computer Science courses taking place in Spring of 2021 at the PH FHNW School of Education were participating in a survey exploring teachers' viewpoints regarding the Process Artifact Creativity Landscape (PACL). This research was conducted according to the ethical standards of FHNW ensuring informed consent, participant privacy, and data integrity in our empirical studies with adult students. These preservice teachers are bachelor students with limited or no

experience in teaching CS. 76% of the teachers in this program were female. Teachers had previously participated in a mandatory CT Science and CT Didactics courses [26].

Teachers were exposed to the PACL framework in two different contexts. During the 14 week *CT Science* course PACL was used explicitly to classify educational activities such as following a tutorial to make a Frogger-like game (Executing) early in the course, and designing their own game (Architecting) as final projects of the course. Later, in the 14 week CT Didactics course, PACL was used again as a pedagogical framework to design three tutorials [26]. As a final project in the *CT Didactics* course teachers had to design their own lesson plan for elementary school K-12 students.

The survey took place before the final lesson plan design project with the idea to make teachers brainstorm about scaffolding creativity in their learning designs. The lesson plans created by the teachers were not part of the study. Lesson plans focused on teaching CT using game design [27] as an approach and leaving a lot of room for creativity. Creativity is a key idea of the 7 Big Ideas part of CS Principles framework [7] used to structure the CT Science course. The survey consisted of 5 open questions prompting an average of about 500 words to be written per subject. The first 4 questions asked if and how they would use each of the approaches (Executing, Modding, Re-Coding, and Architecting). The fifth question was the most important question for us as it asked teachers when, how and why they would transition from one scaffolding approach to another.

5 Use Cases Anticipated by Teachers

Anticipated use cases suggested by teachers indicate when to use each PACL approach and how to combine them into a strategy. Each section below reports one key observation and outlines the categories that emerged from coding teachers' answers. The categories are sorted from highest to lowest percentages. Quotes from teachers are included when appropriate. Sections 5.1 - 5.4 report use case percentages for the Executing, Modding, Re-Coding and Architecting PACL approaches. Only the three most frequent coded responses are presented. Section 5.5 reports strategies suggested to combine PACL approaches.

5.1 Executing

Key Observation: In spite of the perceived lack of potential to foster creativity, Executing is considered an important preparatory scaffolding approach for future creative activities.

Example**:** "In order to get to know the functions of [programming tool], I find executing useful. Because this is not about developing your own creativity, but getting to know the tool, which is later a "means to an end" for creativity and computational thinking."

#1 Introduction Activities (50.8%). The majority of teachers were thinking of Executing as a fitting approach to introduce new learning activities e.g., "Yes, I would use executing. I think it can be very helpful, especially at the beginning, because it provides children with precise instructions. In this way you get to know the principle and can collect many success stories, which will motivate you to continue working in the future."

#2 Teaching Basics (16.9%). Executing can teach basics to all students or serve as a just-in-time tutorial for those who missed classes. Example: "I would use Executing to show the pupils the very simple basics and to ensure that they can find their way around the programming interface independently. With a ZPF [Zones of Proximal Flow] video tutorial I show the pupils how to log in and how to clone my project."

#3 Support of less experienced students (12.3%). Executing can be targeted to support less experienced students, e.g., "I would use executing, because it helps students with learning challenges when they can get step-by-step instructions in order to come to a result at the end."

5.2 Modding

Key Observation: Modding is a way for teachers to provide a framework from which students can be creative.

#1 Extension of Teacher Project (42.2%). The modding of teacher-created projects is very popular. Example: "I would give [students] a pre-programmed game and show how they can change it (characters, colors, worlds and design)."

#2 Fostering Creativity (35.6%). A large number of use cases mention creativity as an important goal of the learning activity. In many cases creativity is assumed to be relevant for motivation, e.g., "I would also use modding in my lessons, as the pupils can develop their creativity, but are not left completely on their own. For example, I could create a maze as a template. The pupils should then program their own labyrinth according to their own ideas, drawing the walls and the agents themselves. They can also add different rules to complicate the maze (enemy, obstacle, etc.)."

#3 Follow Up to Execution (10%). A small number of use cases mention explicitly that they are considered to be follow ups to Execution activities. E.g., "I would also use modding in the classroom and probably do it after executing. The pupils can change the project, which they previously created."

5.3 Re-Coding

Key Observation: Re-Coding is for advanced students as they must make a specific artifact without low level direction.

#1 For Advanced Students (56.4%). Students should already have experience in programming and creating games to make this work. Example: "I think the Re-Coding is very good, because with this method the children have to think a lot for themselves. They know what they should get in the end and now they have to find a way to get there. I think this method is not suitable from the start because you need to be familiar with [the tool] first."

#2 Fostering Creativity (27.3%). In spite of also producing a predefined artifact, just like Executing, Re-Coding scored high in terms of creativity. Teachers perceive a high potential for creativity and engagement. Example: "In this way, the pupils can get creative and try out for themselves how to come up with a solution. Then they are probably also more proud of their product and thus more motivated for programming."

#3 Post Executing and Modding (7.3%). Relatively few mention Re-Coding as a natural progression following Executing and Modding. Example: "[I would] not use them until the students have already gained experience, e.g. with [tool], and have created a game based on the principles of Executing and Modding."

5.4 Architecting

Key Observation: Architecting has the highest potential to foster creativity, requires student experience and suits final projects.

#1 Fostering Creativity (32.5%). Creativity is mentioned most frequently in the Architecting use cases. The large potential for creativity is often expressed as implied contrast to other PACL approaches. Example: "Yes, this is how the pupils can express their creativity to the maximum. They could realize their own game idea - recreating games like Frogger or Pacman would of course not be allowed."

#2 For Advanced Students (29.3%). Many teachers perceive Architecting as an approach suited for advanced students. The skills required to engage in Architecting are identified in different ways. Some teachers identify age (e.g., "6th grade"). Others talk about levels of experience necessary to avoid frustration. Example: "I would use architecting for those pupils who already have their own ideas and have a certain degree of independence in programming. I think this is the basic requirement so that the pupils are not overwhelmed too quickly and they can move in the zone of proximal development."

#3 For Final Projects (20.3%). Some teachers believe that Architecting is suited best for final projects taking place after Executing, Modding, and Re-Coding scheduled typically at the end of a semester or school year. Example: "... here children could implement and realize their ideas. They would have learned a lot about their program or their game through executing and modding and could now create their own project from scratch with a lot of diligence and hard work (hard fun)." Hard fun a is a reference to the concept of difficult but highly engaging programming projects introduced by Papert [22].

5.5 Strategy

Key Observation: Most teachers outline a strategy using a sequence of Executing, Modding, Re-Coding and finally Architecting in order to cover a wide spectrum of scaffolding versus creativity tradeoffs.

Strategy is about when, and how, teachers would combine PACL approaches to deal with the scaffolding versus creativity balance. The top two suggested strategies were:

EMRA (57%). The vast majority of teachers outlined a **E**xecuting > **M**odding > **R**e-Coding > **A**rchitecting strategy. Example: "When I introduce a new topic, I would give the students a lot of guidance and then use the principle of scaffolding. This means that I first use the Executing, Modding and Recoding methods in my annual plan. I would actually always take these steps at the beginning of a new topic. If the students are already familiar with the newly learned topic, I would use the method of Architecting. This allows them to apply the knowledge they have already learned independently and also to deepen it."

Flexible (10.6%). Some teachers moved beyond the vision of a strategy consisting of fixed PACL approach sequences. They pointed out the need to be flexible by finding the right approach matching specific learning situations, e.g., "I would offer all [PACL approaches] in combination and thus offer pupils the opportunity to work individually at their own level and according to their personal interests but also skills."

6 Discussion

Only two subjects (3%) indicated negative dispositions towards Executing. Both subjects appear to be concerned not only about the lack of potential to foster creativity but also with the limited potential for learning e.g., "Because if you were to give them everything step by step, they don't really have to think for themselves what they are really doing here."

Modding was the least controversial approach. Modding was also the approach that most teachers noticed as an intuitive extension of other PACL approaches.

Teachers avoiding Re-Coding worry about students getting frustrated when there is no well-defined and explicit design process to create specific artifacts. How would teachers know that their students are ready to engage in a more open-ended process? Only two teachers mentioned the use of constructs such as Computational Thinking Patterns [16] which may be helpful to serve as decompositional abstractions. The tradeoffs between scaffolding and creativity appeared to be less clear compared to Architecting. Some teachers suggested that lack of instruction could results in students frustration. However, in the case of Architecting they see at least more options for students because they could build any artifact they want to. Some teachers suggested that this may offset frustration from lack of instruction.

While Architecting was perceived to have the highest potential for creativity, teachers were concerned with actually reaching this approach in their lesson plan. Some commented that students would need to be of a certain age, have a high level of programming experience, or already needed to be highly creative to be able to design their own game concept. In other words, it was clear that Architecting, while providing minimal scaffolding, leaves the most room for creativity.

While there was some evidence that teachers have compelling ideas, or even concrete teaching experience, on how to move horizontally in the PACL landscape (i.e., from "same artifact" to "different artifact") there was only little evidence that they have compelling strategies to move down vertically (i.e., from "same process" to "different process"). This may suggest the need to stress the notion of constructs such Computational Thinking Patterns as an essential part of scaffolding. In future versions of the CT Science, and even more so in the CT Didactics courses, one should contemplate the use of more explicit practices to provide guidelines for appropriate levels of competences as preconditions to transition to Re-Coding and Architecting approaches. Perhaps it would help to make the notion of pattern

competence, i.e., the understanding of Computational Thinking Patterns, more distinct. Research, for instance, could explore the role of gamification to reach concrete learning goals.

The EMRA (Executing > Modding > Re-Coding > Architecting) strategy was clearly preferred by a large margin (EMRA 56.1%, MRA 4.5%, EMR 3%, others 1.5%). Using that sequence in the CT Science course probably played an important role. Some answers hinted quite concretely in this direction, e.g., “Basically, my idea would be to structure the lessons [plan] as we have already experienced [in the CT Science course].” Additionally, the survey questions were also EMRA sequenced which could have further amplified this bias. However, the main point of this paper is not that teachers preferred a specific approach progression as method but, similar to the goal of Bloom’s taxonomy [18], that teachers had developed a common language to effectively reason about creativity versus scaffolding tradeoffs. Evidence of this reasoning was not only found in the questionnaires but also in the lesson plans which they had to produce as final projects for the course.

Conclusions

To foster creativity in K-12 computer science education teachers benefit from frameworks helping them to scaffold creative projects. The Process Artifact Creativity Landscape (PACL) provides a conceptual, two-dimensional space that can help teachers to classify pedagogical approaches providing different tradeoffs between scaffolding and creativity. PACL consists of four distinct approaches called: Executing, Modding, Re-Coding, and Architecting. A survey suggests that most teachers were able to use PACL to reason about individual activities and complete lesson plans. This reasoning allowed them to suggest appropriate scaffolding to support creative programming projects.

REFERENCES

[1] Alexander, C., The timeless way of building vol. 1: New york: Oxford university press, 1979.

[2] Amabile, T. M. and J. Pillemer, "Perspectives on the social psychology of creativity," The Journal of Creative Behavior, vol. 46, pp. 3-15, 2012.

[3] Anderson, L. W. and D. R. Krathwohl, A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives: Longman, 2001.

[4] Bloom, B. S., "Taxonomy of educational objectives. Vol. 1: Cognitive domain," New York: McKay, vol. 20, p. 1, 1956.

[5] Chikofsky, E. J. and J. H. Cross, "Reverse engineering and design recovery: A taxonomy," IEEE software, vol. 7, pp. 13-17, 1990.

[6] Coenraad, M., J. Palmer, D. Weintrop, D. Eatinger, Z. Crenshaw, H. Pham, and D. Franklin, "The Effects of Providing Starter Projects in Open-Ended Scratch Activities," in Proceedings of the 52nd ACM Technical Symposium on Computer Science Education, 2021, pp. 38-44.

[7] Cuny, J., "Transforming k-12 computing education: an update and a call to action," ACM Inroads, vol. 6, pp. 54-57, 2015.

[8] Dasgupta, S., W. Hale, Andr, #233, s. Monroy-Hern, #225, ndez, and B. M. Hill, "Remixing as a Pathway to Computational Thinking," presented at the Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing, San Francisco, California, USA, 2016, 1438-1449.

[9] Gamma, E., R. Helm, R. Johnson, and J. Vlissides, "Design patterns: Abstraction and reuse of object-oriented design," in European Conference on Object-Oriented Programming, 1993, pp. 406-431.

[10] Grabowski, S. and F. Nake, "Between the Trivial and the Impossible. ReCoding as Learning Strategy."

[11] Hammond, J. and P. Gibbons, "What is scaffolding," Teachers’ voices, vol. 8, pp. 8-16, 2005.

[12] Kafai, Y., "Playing and Making Games for Learning," Games and Culture, vol. 1, pp. 36-40, 2006.

[13] Kim, J., M. Dontcheva, W. Li, M. S. Bernstein, and D. Steinsapir, "Motif: Supporting novice creativity through expert patterns," in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, 2015, pp. 1211-1220.

[14] Kim, K. H., "Can we trust creativity tests? A review of the Torrance Tests of Creative Thinking (TTCT)," Creativity research journal, vol. 18, pp. 3-14, 2006.

[15] Knobelsdorf, M. and R. Romeike, "Creativity as a pathway to computer science," in Proceedings of the 13th annual conference on Innovation and technology in computer science education, 2008, pp. 286-290.

[16] Koh, K. H., H. Nickerson, A. Basawapatna, and A. Repenning, "Early validation of Computational Thinking Pattern Analysis," presented at the Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education (ITICSE), Uppsala, Sweden, 2014, 213-218.

[17] Koper, R., "Current research in learning design," Journal of Educational Technology & Society, vol. 9, pp. 13-22, 2006.

[18] Krathwohl, D. R., "A revision of Bloom's taxonomy: An overview," Theory into practice, vol. 41, pp. 212-218, 2002.

[19] Lamprou, A. and A. Repenning, "Teaching how to teach Computational Thinking," presented at the the 23rd Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2018), Larnaca, Cyprus, 2018.

[20] Lee, I., F. Martin, J. Denner, B. Coulter, W. Allan, J. Erickson, J. Malyn-Smith, and L. Werner, "Computational thinking for youth in practice," Acm Inroads, vol. 2, pp. 32-37, 2011.

[21] Nikolopoulou, K., "Creativity and ICT: Theoretical approaches and perspectives in school education," in Research on e-Learning and ICT in Education, ed: Springer, 2018, pp. 87-100.

[22] Papert, S. (1985). Hard Fun. Available: http://www.papert.org/articles/HardFun.html

[23] Papert, S. and I. Harel, Eds., Constructionism. Norwood, NJ: Ablex Publishing Corporation, 1993, 518 Pages

[24] Repenning, A. and A. Basawapatna, "Explicative programming," Communications of the ACM, vol. 64, pp. 30-33, 2021.

[25] Repenning, A., A. Basawapatna, D. Assaf, C. Maiello, and N. Escherle, "Retention of Flow: Evaluating a Computer Science Education Week Activity," presented at the Special Interest Group of Computer Science Education (SIGCSE 2016), Memphis, Tennessee, 2016.

[26] Repenning, A., A. Lamprou, and A. Basawapatna, "Computing Effect Sizes of a Science-first-then-didactics Computational Thinking Module for Preservice Elementary School Teachers," presented at the Special Interest Group on Computer Science Education Technical Symposium (SIGCSE TS 2021), Toronto, Canada, 2021, 7.

[27] Repenning, A., D. C. Webb, K. H. Koh, H. Nickerson, S. B. Miller, C. Brand, I. H. M. Horses, A. Basawapatna, F. Gluck, R. Grover, K. Gutierrez, and N. Repenning, "Scalable Game Design: A Strategy to Bring Systemic Computer Science Education to Schools through Game Design and Simulation Creation," Transactions on Computing Education (TOCE), vol. 15, pp. 1-31, 2015.

[28] Repenning, A., J. Zurmühle, A. Lamprou, and D. Hug, "Computational Music Thinking Patterns: Connecting Music Education with Computer Science Education through the Design of Interactive Notations," presented at the 12th International Conference on Computer Supported Education, Prag, 2020, 641-652.

[29] Rhodes, M., "An analysis of creativity," The Phi delta kappan, vol. 42, pp. 305-310, 1961.

[30] Sentance, S. and J. Waite, "PRIMM: Exploring pedagogical approaches for teaching text-based programming in school," in Proceedings of the 12th Workshop on Primary and Secondary Computing Education, 2017, pp. 113-114.

[31] Sternberg, R. J. and W. M. Williams, How to develop student creativity: ASCD, 1996.

[32] Trnova, E. and J. Trna, "Implementation of creativity in science teacher training," International Journal on New Trends in Education and Their Implications, vol. 5, pp. 54-63, 2014.

[33] Waite, J. and S. Grover, "Worked examples & other scaffolding strategies," Computer Science in K–12: An A to Z Handbook on Teaching Programming, pp. 240-249, 2020.

[34] Ward, T. B., S. M. Smith, and R. A. Finke, "Creative cognition," Handbook of creativity, vol. 189, p. 212, 1999.

[35] Wing, J. M., "Computational Thinking," Communications of the ACM, vol. 49, pp. 33-35, 2006.

[36] Wood, D., J. S. Bruner, and G. Ross, "The role of tutoring in problem solving," Journal of child psychology and psychiatry, vol. 17, pp. 89-100, 1976.

[37] Yucel, I., J. Zupko, and M. S. El-Nasr, "IT education, girls and game modding," Interactive Technology and Smart Education, 2006.