Jocius, R., O'Byrne, W. I., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2021). Infusing Computational Thinking into STEM Teaching: From Professional Development to Classroom Practice. *Educational Technology & Society*, *24* (4), 166–179.

Infusing Computational Thinking into STEM Teaching: From Professional Development to Classroom Practice

Robin Jocius[1*], W. Ian O'Byrne[2], Jennifer Albert[3], Deepti Joshi[3], Richard Robinson[3] and Ashley Andrews[3]

[1]University of Texas at Arlington, TX, United States // [2]College of Charleston, SC, United States // [3]The Citadel, SC, United States // robin.jocius@uta.edu // obyrnei@cofc.edu // jalbert@citadel.edu // djoshi@citadel.edu // rjmr@citadel.edu // aandrew1@citadel.edu

[*]Corresponding author

ABSTRACT: Despite increasing attention to the potential benefits of infusing computational thinking into content area classrooms, more research is needed to examine how teachers integrate disciplinary content and CT as part of their pedagogical practices. This study traces how middle and high school teachers (n = 24) drew on their existing knowledge and their experiences in a STEM professional development program to infuse CT into their teaching. Our work is grounded in theories of TPACK and TPACK-CT, which leverage teachers' knowledge of technology for computational thinking (CT), CT as a disciplinary pedagogical practice, and STEM content knowledge. Findings identify three key pedagogical supports that teachers utilized and transformed as they taught CT-infused lessons (articulating a key purpose for CT infusion, scaffolding, and collaborative contexts), as well as barriers that caused teachers to adapt or abandon their lessons. Implications include suggestions for future research on CT infusion into secondary classrooms, as well as broader recommendations to support teachers in applying STEM professional development content to classroom practice.

Keywords: Computational thinking, STEM, TPACK, TPACK-CT, Professional development, Teacher learning, Computer science education

1. Introduction

An emerging body of research (Kafai et al., 2020; Li et al., 2020) has outlined the potential promise--and potential pitfalls--of infusing computational thinking (CT) into disciplinary instruction. Since Jeannette Wing (2006) described computational thinking as a "fundamental skill for everyone, not just for computer scientists" (p. 33), researchers and practitioners alike have sought to identify models, supports, and common pedagogical practices for integrating CT into P-12 educational systems (Barr & Stevenson, 2011; Grover, 2017). Researchers (Yadav et al., 2016) argue that CT, which refers to a set of practices and habits of mind that students can learn with or without the introduction of technology (Papert, 1980; Voogt et al., 2015), can connect to and even enhance numerous disciplinary practices in content area classrooms.

CT provides a quintessential example of real-world problem solving that is integral to STEM education (Israel et al., 2015). Further, infusing CT into disciplinary classrooms can provide opportunities for students and teachers to solve problems while working collaboratively, embedding design methodologies, and using technology appropriately (Shute et al., 2017; Weintrop et al., 2016). However, teachers need explicit support in making these connections between CT and STEM content and understanding the potential benefits for their students (Li et al., 2020).

Although a growing body of literature has demonstrated the critical importance of professional development (PD) in shifting teacher beliefs and self-efficacy for integrated STEM teaching (Nadelson et al., 2013), a crucial and often under-investigated factor is how teachers apply and transform new learning to classroom practice (Goodnough et al., 2014). This is particularly relevant to integrated PD in STEM and CT, which is still in the nascent stages (Hestness et al., 2018). This study draws on theories of TPACK and TPACK-CT to examine data collected from a 3-year, NSF-funded research project to support middle and high school teachers as they infuse CT into their disciplinary teaching. During two week-long PD workshops held in Summer 2018 across two Southeastern states, 116 teachers designed CT-infused lessons using Snap!, a block-based programming tool similar to Scratch (Harvey & Mönig, 2010). In order to trace teachers' experiences with infusing CT, this paper examines data from a purposive sample of 24 teachers who implemented lessons in their classrooms during the 2018-2019 academic year. Data included elements of process and product related to teachers' CT-infused lesson implementation (pre- and post-lesson reflections, classroom video recordings, lesson plans, and interviews). We identify how teachers drew on their developing knowledge of TPACK-CT to transform three key pedagogical supports (articulating a key purpose for CT infusion, scaffolding, and collaborative contexts) for classroom contexts, as well as barriers that caused teachers to adapt or abandon their lessons. We conclude with suggestions

for future research on CT infusion, as well as broader implications to support teachers in applying STEM professional development content to classroom practice.

2. Conceptual framework

Technological Pedagogical Content Knowledge (TPACK; Mishra & Koehler, 2006) was used as both a guiding framework for the design of the Infusing Computing professional development and as an analytic lens for examining the supports and barriers that teachers encountered while implementing CT-infused lessons in their classrooms. TPACK focuses on the interactions among three types of knowledge: subject matter or content knowledge (CK), technology knowledge (TK), and pedagogical knowledge (PK) (Angeli et al., 2016a). In the differentiation across these three types of knowledge, the TPACK framework outlines how content is being taught, and pedagogy, or how the teachers instruct that content. This designation is important because the technology being implemented must communicate the content and support the pedagogical opportunities in order to enhance students' learning experiences. Recent critiques (Koh, 2019; Saubern et al., 2020) have called attention to the need to focus not on individual elements of TPACK, but rather to use it to understand and develop the knowledge teachers need to integrate technology into teaching.

While computing is not simply technology integration, Mouza et al. (2017) suggested that an expanded model of TPACK-CT can support pre-service teachers' understanding of CT integration into content learning. When learning about CT concepts, pre-service teachers develop specialized technological knowledge (i.e., TK-CT), which refers to the "computing tools, vocabulary, practices, and dispositions" that interact with CK and PK (p. 63). Rather than focusing on the individual elements and obscuring how teachers use technology to meet teaching and learning goals (Saubern et al., 2020), TPACK-CT is a holistic model that explains how teachers move from fragmented understandings to bring together CT concepts, tools, existing disciplinary practices, and pedagogies (Mouza et al., 2017). This conceptualization presents a shift in CT-infused instruction from a view of CT skills that are generalizable across the curriculum to a disciplinary perspective of practices specific to the specialized language, knowledge, and habits of thinking within particular subject areas.

While initially developed for use with pre-service teachers, we believe that TPACK-CT provides an important lens for supporting and studying in-service teachers' development. Unlike pre-service teachers, in-service teachers have specific student needs and teaching contexts to consider as they integrate CT into their classrooms. They also bring a wealth of expertise and pedagogical content knowledge developed over years of practice. Explicating how TPACK-CT functions in relation to in-service teacher PD offers a potentially valuable way to understand how experienced teachers transform content-driven, pedagogically sound, uses of technology to support students in understanding and using CT to enhance disciplinary learning.

3. Review of the literature

In the following sections, we describe prior research and perspectives emerging from various bodies of literature--CT in educational contexts (Grover & Pea, 2013; Wing, 2006), CT as disciplinary practice (Weintrop et al., 2016), and integrated PD models in STEM and CT (Nadelson et al., 2012; Ring et al., 2017). Given ongoing debates about what CT is and how to best teach it in disciplinary classrooms (Yadav et al., 2017), we draw from multiple perspectives targeting teacher professional learning STEM education in order to highlight the social and cognitive processes needed to embed CT into disciplinary teaching (Kafai et al., 2020; Li et al., 2020).

3.1. Computational thinking in educational contexts

Generally, CT is seen to encompass many of the critical problem-solving practices and concepts that draw on computer science (Wing, 2006), including "problem representation, prediction, and abstraction" (Israel et al., 2015, p. 264). Computer science (CS) in this lens is identified as the study of computers, hardware, software, and the algorithmic processes and applications that impact society (Lye & Koh, 2014). Researchers (Smith, 2016) have argued that students need a deep understanding of CT concepts and practices in order to navigate a workforce that is largely driven by networked communities, data analytics, and algorithmic processes. These intersections require individuals to think algorithmically to solve ill-formed problems with varying levels of ambiguity in facts and abstraction (Yadav et al., 2017). In educational contexts, CT also involves adapting problem-solving approaches from CS in order to enable deeper content area learning (Grover & Pea, 2013).

How computational thinking is defined—and distinguished from other terms such as programming, computing, and coding—has been the subject of much debate in the CS education literature. Yadav et al. (2014) found that classroom teachers initially view CT as the basic use of technology and computers in the classroom, while Cateté et al. (2018) suggest that teachers need clear definitions of CT elements to integrate CT into their teaching. For the purposes of this study and to make CT elements more adaptable to different disciplines, we refined the four elements of Google's (2018) CT definition as follows: (1) Pattern Recognition: observing and identifying patterns; (2) Abstraction: identifying ideas that are important by naming concepts and hiding details; (3) Decomposition: breaking down problems into meaningful smaller parts; and (4) Algorithms: providing instructions for solving a problem and similar problems (Dong et al., 2019). This conceptualization provided a model for teachers' development of specialized technological knowledge for CT instruction (TK-CT) and helped them draw on their pedagogical content knowledge (PCK) to use and adapt CT elements in discipline-specific and interdisciplinary ways.

3.2. Computational thinking as disciplinary practice

There is an urgent need to infuse "algorithmic problem-solving practices and applications of computing across disciplines" (Barr & Stephenson, 2011, p. 112). To address this need, CT can be taught outside of CS classrooms through integrated and interdisciplinary STEM lessons (Yadav et al., 2014). Weintrop et al. (2016) argue that disciplinary classrooms can provide "a meaningful context (and set of problems) within which computational thinking can be applied" (p. 128).

Despite a growing interest in how a reciprocal relationship between content learning and CT can enrich student learning (Hambrusch et al., 2009; Lin et al., 2009; Weintrop et al., 2016), practical models for infusing CT into disciplinary teaching have remained largely unexplored across both research and practice (Grover & Pea, 2013). As Lye and Koh (2014) argue, the teaching and learning of "computational thinking in naturalistic classroom settings are still not well-understood" (p. 57). Much of the existing research on teacher integration of CT has focused on shifts in teacher beliefs and self-efficacy (Chang & Peterson, 2018; Jaipal-Jamani & Angeli, 2017) and the development of teacher knowledge of CT (Angeli et al., 2016b; Yadav et al., 2017). However, while research has demonstrated that successful CT integration requires teachers to believe in its importance, belief alone is insufficient (Rich et al., 2020a).

Conceptualizing CT as a disciplinary practice that can support content area learning represents "a re-direction in CT education to explicitly and substantially link to STEM education, beyond just CS" (Li et al., 2020, p. 152). In an attempt to make explicit the connections between CT and STEM education, this study foregrounds the critical thinking elements of CT and the design thinking process, with a secondary focus on programming. This framing supports teachers in developing TPACK-CT, which connects CT concepts to classroom teaching practices. In response to the need to contribute to the "scarce" literature on teacher implementation of CT-infused disciplinary teaching (Yadav et al., 2020), this study explores how teachers transform professional learning for classroom practice and the barriers they face.

3.3. Integration of STEM and CT into professional development

As is the case with definitions of CT, there are numerous conceptualizations of integrated STEM education, including interdisciplinary approaches that connect two or more STEM disciplines (Sanders, 2009) and transdisciplinary approaches (Vasquez et al., 2013) focused on applications of STEM content to complex problem-solving (Brown, 2012; Rinke et al., 2016).

Previous research (Avery & Reeve, 2013; Nadelson et al., 2013; Norris & Soloway, 2016; Ring et al., 2017) suggests that effective STEM PD models support shifts in teacher self-efficacy about STEM teaching and offer explicit opportunities for teachers to practice integrating appropriate STEM content. As teachers face numerous pressures in terms of covering rigidly structured curricula (Avery & Reeve, 2013), a key element of STEM PD design involves explicitly linking innovative pedagogies to existing standards. Further, as teachers come to any professional learning experience with varied goals and learning trajectories, PD must be explicitly designed to personalize content and outcomes for classroom application (Ring et al., 2017). This is particularly relevant for PD that aims to develop teachers' CT knowledge and pedagogical practices, which are emerging areas for pre-service and in-service teacher education (Angeli & Giannakos, 2020). CT PD also represents unique challenges in that teachers must simultaneously develop CT knowledge and skills for themselves and for the students they teach (Mouza et al., 2017).

Despite inherent connections between CT and STEM, much of the research on STEM education does not consistently connect thinking and computational processes (Li et al., 2020). As Israel et al. (2015) argue, "More research is necessary in order to understand implementation, supports that teachers require, and the types of instructional strategies that could support diverse learners in engaging in computational thinking" (p. 277). There are a number of potential explanations for this gap in the knowledge base--CT infusion requires teachers to bring together complex, and often contradictory, disciplinary norms and practices (Israel et al., 2015); rethink pedagogical approaches (Guzdial, 2008); and reconsider their identities as experts (Israel et al., 2015).

While growing attention has focused on the needs of pre-service teachers (Chang & Peterson, 2018), researchers (Sands et al., 2018) have argued that in-service teachers need ongoing PD to correct misconceptions, learn how to engage their students with CT, and practice and refine new pedagogical practices. A key mechanism for this work is to provide in-service teachers with the training, tools, and resources to understand the benefits of CT infusion, all while they learn how to put concepts into practice. As described in the following sections, this study and the Infusing Computing PD draws on TPACK and TPACK-CT, as well as existing research on effective STEM PD, CT integration into disciplinary content, and CT as a disciplinary practice to investigate how teachers create and transform pedagogical supports for students and the barriers they face as they infuse CT into their pedagogical practice.

4. Methods

This study addresses the following research questions:

- How did pedagogical supports for CT-infused lessons function in classroom settings?
- What barriers did teachers face as they implemented CT-infused lessons into their disciplinary teaching?

Next, we describe the context for the study, the 3C professional development model, teacher perceptions of the PD experience, data sources, and analytic techniques.

4.1. Context for the study

This study emerges from a three-year research project, Infusing Computing, which aims to document how middle and high school teachers create and implement interdisciplinary, CT-infused lessons. The project incorporates multiple supports for teacher learning and lesson implementation: (1) week-long summer PD workshops designed for non-computing teachers; (2) academic year support; and (3) opportunities to return for additional summer PD sessions and/or serving as teacher-leaders.

This paper draws on quantitative and qualitative analyses of data from Year 1 of the project, which included two one-week summer PD sessions held in two Southern states in Summer 2018 and the 2018-2019 academic year implementation. The focus of this study is teachers' implementation of PD content, including their CT-infused lessons, into their classrooms. Of the 116 teachers who attended the Summer 2018 PD sessions, 24 teachers completed lesson implementation and shared the following data sources with the research team: pre-implementation reflections, post-implementation reflections, updated teacher-created lesson plans, end-of-year surveys, and classroom videos. 13 teachers also participated in semi-structured interviews. We chose to focus specifically on the sample of 24 teachers, rather than all teachers who implemented lessons but did not video-record or submit reflections, in order to triangulate findings across data sources.

Of the 24 teachers in this study, 37.5% identified as science teachers, 20.8% identified as math teachers, 16.7% identified as English teachers, 12.5% identified as social studies teachers, and 12.5% identified as other (i.e., business, Spanish, forensic science, instructional coach, and technology). Teachers had an average of 14.0 years of teaching experience; two teachers had fewer than three years and nine teachers had more than 20 years. Most teachers reported a relatively high level of comfort with using technology as a pedagogical tool; on a pre-PD survey item measuring teachers' self-efficacy in using technology, the mean score was 3.81 (n=24). However, only a small percentage of teachers (16.7%) had used programming tools prior to the PD.

4.2. The 3C model

The summer PD workshops were structured according to the 3C (Code, Connect, Create) model (Jocius et al., 2020). During the *Code* sessions, teachers developed their technological content knowledge (TCK) and

specialized technological knowledge (TK-CT) by learning how to code in Snap!, a block-based programming environment similar to Scratch that allows users to create animations, games, and other multimodal projects (Maloney et al., 2010). The Code sessions introduced concepts and operations in the Snap! programming environment (e.g., sprites, blocks), control structures (e.g., loops, conditionals, variables), and lists. Facilitators modeled pair programming, which positions one participant as the "driver" using the computer and the other as the "navigator" who provides verbal instructions (Hanks et al., 2011). Then, teachers worked in pairs to use, modify, and create (UMC) programs (Dong et al., 2019; Lee et al., 2011). As teachers used existing Snap! simulations and modified programs, they engaged with CT practices from the perspective of the learner.

Connect sessions were designed to support teachers in identifying authentic opportunities for bridging disciplinary practice and CT, which drew on their content knowledge (CK) and pedagogical knowledge (PK). During Connect sessions, teachers developed understanding of how CT-related concepts, computing tools, and practices (TK-CT) can be infused into disciplinary content (CK) and pedagogies (PCK) to create meaningful outcomes (TPACK-CT). As teachers learned about CT in different disciplinary contexts, they were also tasked with unpacking the practices involved with teaching within their disciplines. The goal was for them to see CT not as a technology-specific practice best left for CS classes, but as a learning process with the potential to create new opportunities for consuming and producing disciplinary content. During Connect sessions, teachers created curriculum maps to identify standards, activities, and supports for disciplinary CT infusion.

Finally, *Create* sessions supported teachers in creating CT-infused lessons, developing their pedagogical content knowledge (PCK) and knowledge of specific strategies to infuse CT (TPACK-CT). Lessons included (1) a Snap! prototype; (2) a detailed lesson plan; and (3) supplemental pedagogical materials. Each Create session included opportunities for reflection on new learning from Code and Connect sessions, structured goal-setting activities, and individualized support from members of the project team.

All elements of the 3C model were explicitly designed to scaffold teachers towards increasingly complex understandings of CT and to help them recognize integration opportunities in both discipline-specific and interdisciplinary ways. Figure 1 provides an overview of the 3C model, facilitator characteristics, and sample activities for each component.

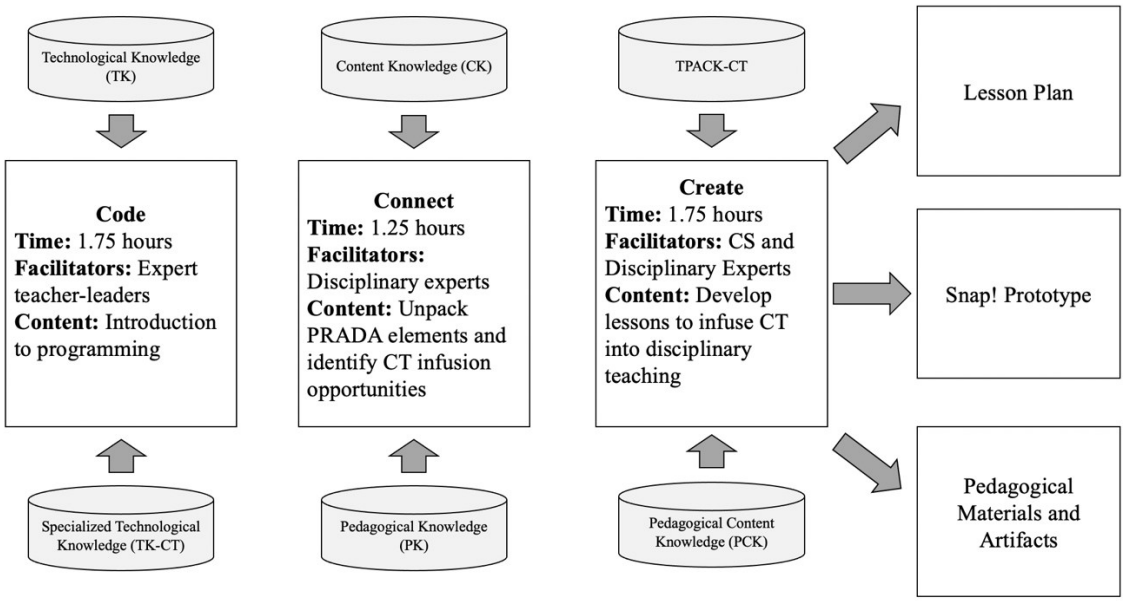

Figure 1. Code, Connect, Create (3C) professional development model

4.3. Teacher evaluation of the 3C model

A post-PD survey including 22 quantitative items and three open-ended items was administered to all participants at the end of the two PD sessions. Items from the validated Standards Assessment Inventory (Vaden-Kiernan et al., 2009) were adapted to understand teachers' perceptions of the PD in relation to three domains: content, context, and process. Initial analysis of quantitative survey results revealed significant shifts in teacher self-efficacy and beliefs regarding CT infusion into disciplinary teaching (Jocius et al., 2020). 74% of teachers

ranked the PD as excellent, with an average rating of 4.71 (5.0 scale). We also performed a paired samples t-test to measure teachers' perceptions of their knowledge and skills before and after the workshop, which demonstrated that there was a statistically significant increase in teachers' perceptions of their skills/knowledge ($p < .001$) (see Table 1). In quantitative and open-ended items, teachers reported overall satisfaction with the PD experience, particularly in relation to their self-efficacy in infusing CT, coding skill development, and experiences collaborating with colleagues.

Table 1. Teachers' perceptions of knowledge and skills before and after the workshop

Before/After the PD, I would rate my knowledge or skills as:							
	Mean	Paired samples *t*-test[1]	Poor (1)	Fair (2)	Good (3)	Very Good (4)	Excellent (5)
State 1 ($n = 55$)							
Before	1.51	$p < .001^{**}$	65%	20%	13%	2%	
After	3.61		7%	9%	49%	29%	5%
State 2 ($n = 66$)							
Before	1.69	$p < .001^{**}$	65%	11%	9%	9%	
After	3.31		4%	11%	31%	31%	9%

Note. [1]Paired samples *t*-tests assess significant changes in percent correct from pre to post; $^{*}p < .05$; $^{**}p < .01$.

4.4. Academic year support and teacher lesson implementation

Drawing on research on active collaboration for teacher learning and PD application (Darling-Hammond, 2005; Borko, 2004), we offered several academic-year supports, including monthly webinars, a Slack channel for online discussion, and technical assistance. Monthly webinar topics included: teacher reflections on lesson implementation, sample lessons, tips and tricks for infusing computing, Snap! functions, unplugged activities, and CT assessment.

4.5. Data analysis

Data analysis proceeded in two phases aligned with the research questions (see Table 2). In Phase 1, we examined teachers' post-lesson implementation reflections and open-ended end-of-year survey responses. Using inductive qualitative coding techniques informed by grounded theory (Charmaz, 2006), we analyzed post-lesson reflections ($n = 24$) using descriptive codes for common topics and in vivo codes to note participants' own words (Saldaña, 2015). During this coding cycle, the area of pedagogical support for CT-infused lesson implementation emerged as a salient theme. A second cycle utilized pattern coding techniques to develop thematic organization (Saldaña, 2015). All responses were double-coded, resulting in 89% inter-rater agreement.

Table 2. Research questions and data sources

Research question	Data sources	Analytic methods
How did pedagogical supports for CT-infused lessons function in classroom settings? (Phase 1)	-Pre-lesson reflection ($n = 24$) -Post-lesson reflections ($n = 24$) -End-of-year surveys ($n = 24$) -Teacher interviews ($n = 13$) -Video recordings of lessons ($n = 24$)	-Descriptive, in vivo, and pattern coding (Saldaña, 2015)
What barriers did teachers face as they implemented CT-infused lessons into their disciplinary teaching? (Phase 2)	-Post-lesson reflection ($n = 24$) -End-of-year surveys ($n = 24$) -Teacher interviews ($n = 13$)	-Idea units (Chafe, 1994) -Pattern coding (Saldaña, 2015)

In Phase 2, we identified two specific survey items: a post-lesson reflection question asking teachers to identify the least successful lesson elements and an end-of-year survey response that targeted teachers' perceived barriers to implementation. We began the analytic process by breaking the comments into idea units, which Chafe (1994) describes as pieces of discourse in which the speaker introduces a singular concept. We then reviewed idea units to eliminate redundancies across data sources, so that each idea unit represented a distinct teacher-identified barrier. Using pattern coding techniques, we engaged in multiple cycles of analysis to organize the barriers into categories and themes. All responses were double-coded, resulting in 83% inter-rater agreement.

Throughout both phases of data analysis, we utilized multiple strategies to ensure trustworthiness. We triangulated interpretations across data sources, including video recordings of lesson implementation and interview responses. We also utilized peer debriefing during weekly research team meetings to discuss interpretations and review emergent codes. Member checks were conducted with nine participants during follow-up interviews. Finally, we kept an audit trail to keep a record of changes throughout the analytic process.

5. Findings

Our analysis identified three primary pedagogical supports that teachers utilized as they infused CT into disciplinary teaching: articulating a key purpose for CT infusion, scaffolding, and student collaboration (see Figure 2). Of the 24 teachers, 79.2% (n = 19) identified scaffolding, 45.8% (n = 11) identified articulating a clear purpose for CT infusion, and 33.3% (n = 8) identified student collaboration as pedagogical supports. In the following sections, we draw on teachers' pre- and post-lesson reflections, revised lesson plans, interview responses, and video recordings of lessons to illustrate how these pedagogical supports for CT infusion functioned in classroom contexts.

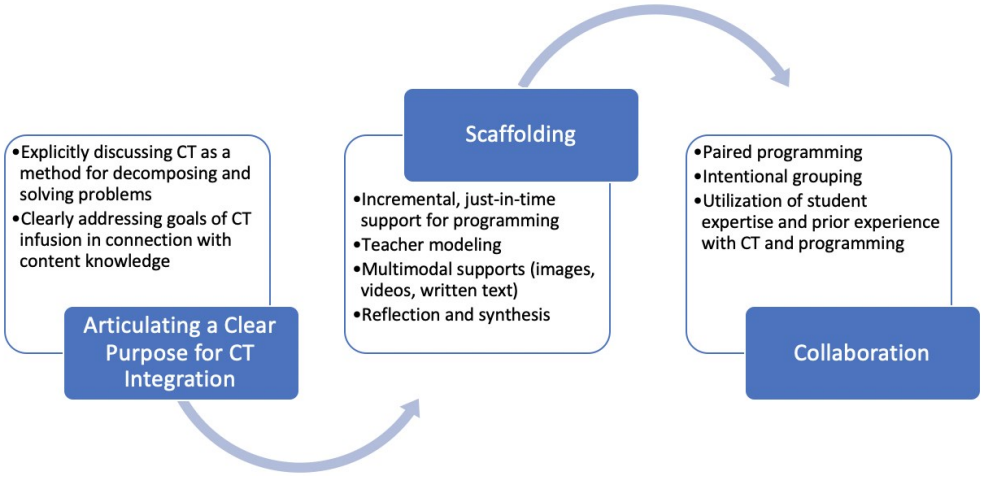

Figure 2. Overview of pedagogical supports for CT infusion

5.1. Articulating a clear purpose for CT infusion

45.8% (n = 11) of teachers described explicit discussion of the goals for CT infusion as a support for lesson implementation. For example, Jessica, a middle school English and speech teacher, drew on specialized technological knowledge (CT-TK) to introduce specific CT elements, such as decomposition, to students: "This is a skill that is essential to problem-solving. The students need to be able to recognize the problem, and then be able to break it down into smaller problems that can be solved." Jessica also said that including "points for reflection," which she adapted from the Connect sessions, supported students' creation of Snap! narratives that recreated Edgar Allen Poe stories.

Other teachers made clear connections between CT elements and disciplinary content to scaffold students' developing knowledge of CT. For example, Jane, a middle school forensics teacher, said: "We went into pattern recognition. Then, you got this big problem of, you know, criminal activity. They've got to learn how to break that down into little bite-sized pieces." Likewise, Alan, a high school math teacher, reported using concept mapping, similar to the work done during Connect sessions, to highlight connections between the CT elements of decomposition and algorithms and math. He recognized the importance of pedagogies utilizing holistic models of TPACK-CT, rather than focusing specifically on technology: "Technology for the sake of technology is not beneficial. I have to be able to use it to teach math. And students need to see that."

Phoebe, a 7th grade science teacher, reported that explicit discussion helped students apply CT concepts, including decomposition and abstraction, to problem-solving processes and disciplinary knowledge. Her lesson tasked students with creating a Snap! flowchart to solve a genetics problem using a Punnett Square. Students completed a storyboard illustrating the passing of genetic information through meiosis, DNA translation, simple dominant, and recessive trait behavior. Then, they used the storyboard to create a Snap! program (see Figure 3).

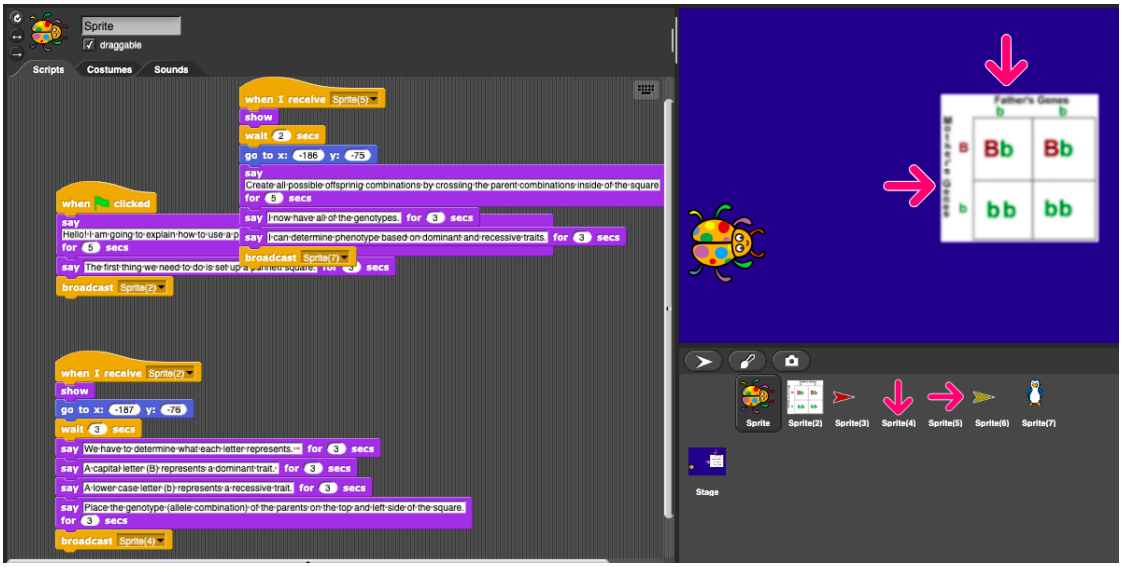

Figure 3. Teacher-created Punnett square Snap! storyboard

Phoebe's pre-lesson implementation reflection noted that she initially planned to share the Punnett square storyboard, introduce Snap! functions, and describe coding procedures. However, lessons she learned from the first day of instruction led her to include an explicit justification for CT infusion:

> I don't feel like I explained the purpose of it. It's really easy to just give a lesson and students just try to complete it to please the teacher but not understanding why this is helpful and how they can take it with them in other lessons. I did go back the 2nd day and stop the students and talk to them about what computational thinking was and how they can use it in other aspects.

Phoebe drew on her pedagogical content knowledge, as well as TK-CT, to utilize the CT definitions from the PD to scaffold both students' CT and content understandings. When asked to describe the most effective elements of the lesson, Phoebe identified students' end products "showing their understanding of the content" and "the 'ah-ha' moments when they really started understanding the coding."

5.2. Scaffolding

79.2% of teachers (n = 19) identified pre-lesson, during-lesson, and post-lesson scaffolds as a for CT-infused lesson implementation. Examples of *pre-lesson scaffolds* included: 10-15 minute Snap! programming lessons, storyboards, review of Snap! commands, and visuals to demonstrate CT concepts. Ellen, a middle school science teacher, repurposed PD Code session materials to introduce Snap! to her students: "We had that material available to us. So those step-by-step directions, that was very supportive because that's one less thing I have to create." Other teachers utilized activities demonstrated during the academic-year webinars, such as unplugged coding, to introduce algorithms and abstractions, using unplugged maze activities and warm-up discussions centered on CT concepts.

During-lesson scaffolds (e.g., graphic organizers, sample code, and guided reflection) offered just-in-time support for students to engage with the disciplinary content and CT concepts. Several teachers utilized TPACK-CT knowledge to scaffold lessons. Lauren, a high school science teacher, adapted the use-modify-create model from the Code sessions: "And you use that use, modify and then create your own--that's what I did. I used all of the things that they gave us. I modified the idea that I was going to do in my own head. And then I created something that also includes a 'complete your own code' and has the kids create their own costumes and sprites." Lauren mentioned that the code completion activity was similar to scientific lab work, indicating that she drew upon existing practices and PCK in transforming scaffolds for CT infusion.

Post-lesson scaffolds, which included presentations of programs, student-led discussions, and exit tickets, enabled student reflection on CT knowledge. Teachers noted that scaffolds that they had experienced as learners served as source material for their own adaptation of classroom supports. For example, Jessica shared that her students presented their coded Poe stories during a "demo fair" much like one held during the final day of the summer PD. Other teachers drew on existing pedagogical content knowledge and structures used frequently in

their classrooms, such as book talks and science experiment presentations, as part of their CT lesson implementation.

Teachers also referenced shifts in scaffolds during lesson implementation. Allie's high school English lesson required students to compose an interactive choose-your-own-adventure narrative using CT concepts (decomposition, conditionals, and loops). In her pre-lesson reflection, Allie noted that she planned to model a Goldilocks example but decided to draw on her pedagogical knowledge (PK) and content knowledge (CK), to transform her initial support to instead co-construct a model narrative with students. Allie noted that ultimately, most students were able to successfully reach the objectives related to both content and CT:

> The skill of decomposition was accomplished. They started at the top, recognizing that the story had to have a beginning, middle, and the end. Then, they started designing their game screens on paper, from the beginning story line, developing their story lines as they designed the body of the story, and then resolution/ending screens for each of their stories.

For future iterations of the lesson, Allie said that students will use specific textual evidence to craft their choose-your-own-adventure stories.

5.3. Student collaboration

33.3% of teachers (n = 8) identified student collaboration as a pedagogical support. Teachers noted several types of collaboration, including the intentional use of pair programming (as demonstrated during the PD), informal student collaborations, and leveraging student expertise with programming in other environments. Interestingly, as we detail in the next section on barriers, several teachers also reported that some forms of collaboration instead served to inhibit students from meeting lesson goals.

Teachers found that collaborative structures from the PD designed specifically to build teachers' TK-CT knowledge, such as pair programming, supported student learning during lesson implementation. Ellie, a middle school science teacher, found that using pair programming in her classroom engendered more collaborative interactions, particularly when students' designed algorithms: "The most successful part of the lesson was watching the students turn their written algorithms into a working application and then work to create a practice application. I am very pleased with their growth and engagement during the partner segments (pair programming)." Other teachers echoed the value of collaboration in leading to more complex content understandings of abstraction and algorithms: "Pairs had to rely on each other to 'present' the concepts and to provide correct answers to conversion problems."

Informal student collaborations, which related to teachers' existing pedagogical practices, also served as a support for lesson implementation. Ellie noted that pairs who had successfully translated their algorithms into applications shared tips and tricks with other students. Similarly, Demetria, a middle school Spanish teacher, said that while the Snap! Programming was challenging for her students, students worked together to share developing knowledge: "The other kids were really helpful and willing to show them what they had figured out as well."

Teachers also leveraged students' previous experiences with programming in other environments, such as Scratch, as a pedagogical resource. While Katie, an 8th grade science teacher, noted that all students were successful in programming a water molecule, students with greater expertise in programming utilized their pre-existing knowledge to construct more complex products. As Katie described, "The more complex programs were created by students with Scratch experience or students that received assistance from my student experts." These structures often drew on PD elements designed to support TPACK-CT; for example, Alan stated that student-led demonstrations of specific Snap! functions, like importing content and replicating code, which was similar to the way that teachers interacted during the PD Code sessions.

5.4. Barriers to implementation of CT-infused lessons

In connection with the second research question, a primary goal of this analysis was to explicitly analyze the barriers teachers faced. Across the analysis of post-lesson implementation reflections and end-of-year survey responses, teachers identified barriers (n = 44) in the following areas: time and pacing, scaffolding, infrastructure and technical issues, teacher collaboration, student collaboration, and teacher knowledge.

29.5% of teachers reported a lack of time to implement their CT-infused lessons. As Keith, the high school math teacher, said, “For me, time and flexibility in content was a barrier. It was difficult to free up any time in class to dedicate to Snap! instruction when I struggle as it is to get through the required course content.” Katie said that while teaching the lesson was “double the amount of time” she anticipated, “it was well worth it.”

15.9% of teachers (n = 7) identified technical issues, such lacking computer access and issues with district website filters, as barriers to lesson implementation. Ellie reported that “the school's email blocked Snap! and the school's server blocked their personal emails.” Other teachers said the lack of 1-1 computer ratios and difficulties with Snap! logins caused unanticipated problems during lesson implementation. Throughout implementation, members of the project team helped teachers to create Snap! accounts and to submit requests to district IT departments to allow Snap!, which helped to eliminate these barriers.

18.2% of teachers (n = 8) noted that anticipating student support needs was a barrier to infusing CT. Although several teachers drew on their existing pedagogical content knowledge in designing supports, some teachers expressed that their developing TPACK-CT knowledge limited the types and forms of support they provided. As Katie said, “When I implement Snap next year, I will develop lessons that sequentially build programming skills. For example, lessons will start by building simple skills and culminate by integrating all the skills.” She referenced drawing on her experiences with CT infusion, as well as Code session materials, to design future sequential lessons.

Likewise, while student collaboration was identified by several teachers as a pedagogical support, 13.6% (n = 6) and 11.4% (n = 5) of teachers specifically identified teacher and student collaboration (respectively) as barriers to lesson implementation. Teacher-identified collaboration barriers included: difficulties in navigating co-teaching schedules and structures, lack of alignment across course pacing guides, and failed attempts to collaborate with teachers who did not attend the PD. The student collaboration barriers were all specifically connected to issues arising from differences in students’ coding skill sets. While several teachers found value in utilizing pair programming (as described in the previous section), others found that it limited students’ engagement with the lessons. For instance, as Allie said, “We tried to pair students who struggled with students who were excelling. This ended up leading to less pair programming and more of the driver taking the lead.” This problem was echoed by four other teachers, indicating a need to critically examine the role of pair programming in either enabling or supporting CT-infused lesson implementation.

Finally, 11.4% (n = 5) of teachers reported a lack of teacher knowledge served as a barrier for infusing CT. Teachers specifically pointed to gaps in TK-CT in relation to the use of Snap! As a primary barrier. As Jessica said, “I was asked some questions that I didn't remember how to do, or never learned so I was honest with them and we worked together to find out how to do what was being asked.” In addition, teachers reported needing to reference Connect and Code session materials when both planning and implementing lessons.

6. Discussion and implications

Infusing computational thinking into disciplinary content is a complex and multi-faceted task that requires teachers to develop deep knowledge of CT concepts and principles (TK-CT) and capitalize on the inherent connections between their content and computational thinking (TPACK-CT). As researchers (Shute et al., 2017; Yadav et al., 2014) have argued, a key challenge for those interested in expanding access to CT across disciplinary boundaries is to design and study how teachers use and transform learning from professional development for classroom practice. Our analysis has identified three pedagogical supports--articulating a key purpose for CT infusion, scaffolding, and student collaboration--that teachers successfully used to integrate CT into their disciplinary teaching, as well as the barriers that they faced.

Israel and colleagues (2015) found that “deliberately embedding” CT into the disciplinary content was key for successful integration. However, our teachers found that their students needed the extra step of clearly articulating the connections between the CT and content and the goals for embedding the CT. This work requires teachers to utilize their content knowledge and pedagogical content knowledge to highlight connections between their disciplines and their developing understandings of TK-CT. In addition to drawing on experiences from Infusing Computing Connect sessions, teachers’ prior classroom experiences and knowledge of their students played a key role in how they unpacked the role of CT. For example, teachers reported needed to explicitly explain connections between CT and disciplinary content.

Next, while Mouza et al. (2017) classified TPACK-CT scaffolds used by pre-service teachers as either student-centered or teacher-centered, our study showed that it was easier for in-service teachers to think about scaffolds in relation to their positioning to the lesson components (pre-lesson, during-lesson, and post-lesson). These scaffolds replicate practices that draw on in-service teachers' existing pedagogical content knowledge and represent shifts in how teachers are using TPACK-CT knowledge as a way of making complex content more accessible to students. This suggests that explicitly drawing on teachers' existing pedagogical content knowledge when integrating new STEM pedagogies, particularly in regard to scaffolding classroom instruction for students, could help to address the gaps between teacher learning of new practices in PD settings and classroom implementation (DeJarnette, 2018).

Bower et al. (2017) also classified teaching strategies for CT infusion as student-centered and teacher-centered, finding that "collaboration" was frequently mentioned. Similarly, our teachers stressed that collaboration among students served as a support. Teachers drew on TPACK-CT knowledge to implement collaborative structures implemented during the PD, such as pair programming, as well as existing collaborative structures within their classrooms, to infuse CT into their disciplinary teaching.

Although many teachers were able to successfully implement their CT-infused lessons, our analysis also revealed that they faced a number of barriers, including a lack of time and technological resources, difficulty in anticipating student learning scaffolds, and concerns with both teacher and student collaboration. While some of these barriers, such as the lack of time, are well-aligned with the existing literature (Yadav et al., 2016), others (e.g., the need for specific scaffolds for infusion) suggest potential future directions for research and practice. For example, future research on the ways that teachers in different disciplines and teaching contexts adapt scaffolds (e.g., pair programming, coding mini-lessons, modeling, CT vocabulary support, and unplugged activities) based on their existing pedagogical content knowledge and new understandings of TPACK-CT would make a valuable contribution to the literature.

Finally, the fact student collaboration served as both a support and a barrier suggests that the forms and functions of collaboration play a key role in either supporting or restricting students' engagement and learning (Jocius, 2018; Rodriguez et al., 2017). Providing opportunities for teachers to engage in pedagogies of investigation and enactment (Grossman et al., 2009) to experience and enact different collaborative structures could allow teachers to develop specific instructional routines (TPACK-CT) for CT-infused instruction. Beginning this work in pre-service teacher education courses could also provide an important context for developing skills in infusing CT and designing supports for students with varied learning needs. However, further study is needed to unpack collaborative structures that can enable students of differing levels of expertise to engage fully in CT practices in disciplinary contexts.

7. Conclusion

Given existing gaps in the literature (Rich et al., 2020b), we believe that examining how teachers apply PD content and leverage technological, pedagogical, and content knowledge to transform classroom practice is a critical area for understanding the future of CT infusion. In this study, we identified three key pedagogical supports teachers used to implement CT-infused lessons (articulating a key purpose for CT infusion, scaffolding, and collaborative contexts), as well as barriers that caused them to adapt or abandon their lessons. It is our hope that the pedagogical supports identified in this study, as well as the illustrations of how teachers transformed scaffolds based on their existing pedagogical practices and TPACK-CT knowledge, will assist educators and researchers in developing and transforming supports for classroom use.

As suggested by Li and colleagues (2020), this study positioned CT as a transdisciplinary thinking practice that can support students' development of content knowledge as well as CT skills. We found that as teachers learned about and enacted scaffolds based on their developing knowledge of TPACK-CT in recursive cycles, they came to recognize transdisciplinary opportunities to bridge existing pedagogical practices with the new integrated STEM content. It is our hope that this research can contribute to the design of PD that guides teachers in transforming pedagogical supports based on their existing and developing knowledge, as well as helping them overcome barriers to CT infusion.

Acknowledgement

This material is based upon work supported by the National Science Foundation under grant numbers 1742351 and 1742332.

References

Angeli, C., & Giannakos, M. (2020). Computational thinking education: Issues and challenges. *Computers in Human Behavior, 105*, UNSP 106185. doi:10.1016/j.chb.2019.106185.

Angeli, C., Valanides, N., & Christodoulou, A. (2016a). Theoretical considerations of technological, pedagogical content knowledge. In M. Herring, M. Koehler & P. Mishra (Eds.), *Handbook of technological pedagogical knowledge for* educators (2nd ed.) (pp. 11–30). New York, NY: Routledge.

Angeli, C., Voogt, J., Fluck, A., Webb, M., Cox, M., Malyn-Smith, J., & Zagami, J. (2016b). A K-6 computational thinking curriculum framework: Implications for teacher knowledge. *Educational Technology & Society*, *19*(3), 47-57.

Avery, Z. K., & Reeve, E. M. (2013). Developing effective STEM professional development programs. *Journal of Technology Education*, *25*(1), 55-69.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads*, *2*(1), 48-54.

Borko, H. (2004). Professional development and teacher learning: Mapping the terrain. *Educational Researcher*, *33*(8), 3-15.

Bower, M., Wood, L. N., Lai, J. W., Howe, C., Lister, R., Mason, R., Highfield, K., & Veal, J. (2017). Improving the computational thinking pedagogical capabilities of school teachers. *Australian Journal of Teacher Education, 42*(3). doi:10.14221/ajte.2017v42n3.4

Cateté, V., Lytle, N., Dong, Y., Boulden, D., Akram, B., Houchins, J., Barnes, T., Wiebe, E., Lester, J., Mott, B., & Boyer, K. (2018). Infusing computational thinking into middle grade science classrooms: Lessons learned. In *Proceedings of the 13th Workshop on Primary and Secondary Computing Education* (WiPSCE '18). New York, NY: ACM. doi:10.1145/3265757.3265778

Chafe, W. (1994). *Discourse, consciousness, and time: The Flow and displacement of conscious experience in speaking and writing*. Chicago, IL: University of Chicago Press.

Chang, Y. H., & Peterson, L. (2018). Pre-service teachers' perceptions of computational thinking. *Journal of Technology and Teacher Education*, *26*(3), 353-374.

Charmaz, K. (2006). *Constructing grounded theory*. London, UK: Sage.

Darling-Hammond, L. (2005). Teaching as a profession: Lessons in teacher preparation and professional development. *Phi Delta Kappan*, *87*(3), 237-240.

Dejarnette, N. K. (2018). Implementing STEAM in the Early Childhood Classroom. *European Journal of STEM Education, 3*(3), 18.

Dong, Y., Catete, V., Jocius, R., Lytle, N., Barnes, T., Albert, J., Joshi, D., Robinson, R., & Andrews, A. (2019). PRADA: A Practical model for integrating computational thinking in K-12 education. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education* (pp. 906-912). New York, NY: ACM.

Goodnough, K., Pelech, S., & Stordy, M. (2014). Effective professional development in STEM education: The Perceptions of primary/elementary teachers. *Teacher Education and Practice*, *27*(2-3), 402-423.

Google, Inc. (2018). What is computational thinking? *Computational thinking for educators.* Retrieved from https://computationalthinkingcourse.withgoogle.com/unit?lesson=8%5C&unit=1

Grossman, P., Hammerness, K., & McDonald, M. (2009). Redefining teaching, re-imagining teacher education. *Teachers and Teaching: Theory and Practice, 15*(2), 273-289.

Grover, S. (2017). Assessing algorithmic and computational thinking in K-12: Lessons from a middle school classroom. In P. J. Rich & C. B. Hodges (Eds.), *Emerging Research, Practice, and Policy on Computational Thinking* (pp. 269-288). doi:10.1007/978-3-319-52691-1_17

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A Review of the state of the field. *Educational Researcher*, *42*(1), 38-43.

Guzdial, M. (2008). Education paving the way for computational thinking. *Communications of the ACM*, *51*(8), 25-27.

Hambrusch, S., Hoffmann, C., Korb, J. T., Haugan, M., & Hosking, A. L. (2009). A Multidisciplinary approach towards computational thinking for science majors. *ACM SIGCSE Bulletin*, *41*(1), 183-187.

Hanks, B., Fitzgerald, S., McCauley, R., Murphy, L., & Zander, C. (2011). Pair programming in education: A Literature review. *Computer Science Education*, *21*(2), 135-173.

Harvey, B., & Mönig, J. (2010). Bringing "no ceiling" to scratch: Can one language serve kids and computer scientists. *Proceedings Constructionism* (pp. 1-10). Retrieved from https://snap.berkeley.edu/old_site/BYOB.pdf

Hestness, E., Ketelhut, D. J., McGinnis, J. R., Plane, J., Razler, B., Mills, K., Cabrera, L., & Gonzalez, E. (2018). Computational thinking professional development for elementary science educators: Examining the design process. In *Proceedings of the SITE Conference* (pp. 1904-1912). Waynesville, NC: Association for the Advancement of Computing in Education (AACE).

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). Supporting all learners in school-wide computational thinking: A Cross-case qualitative analysis. *Computers & Education*, *82*, 263-279.

Jaipal-Jamani, K., & Angeli, C. (2017). Effect of robotics on elementary preservice teachers' self-efficacy, science learning, and computational thinking. *Journal of Science Education and Technology*, *26*(2), 175-192.

Jocius, R. (2018). Becoming entangled: An Analysis of 5th grade students collaborative multimodal composing practices. *Computers and Composition, 47*, 14-30.

Jocius, R., Joshi, D., Albert, J., Barnes, T., Robinson, R., Cateté, V., Dong, Y., Blanton, M., O'Byrne, I., & Andrews, A. (2021). The Virtual pivot: Transitioning computational thinking PD for middle and high school content area teachers. In *Proceedings of the 52nd ACM Technical Symposium on Computer Science Education* (pp. 1198-1204). New York, NY: ACM.

Jocius, R., Joshi, D., Dong, Y., Robinson, R., Cateté, V., Barnes, T., Albert, J., Andrews, A., & Lytle, N. (2020). Code, connect, create: The 3c professional development model to support computational thinking infusion. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (pp. 971-977). New York, NY: ACM.

Kafai, Y. B. (2016). From computational thinking to computational participation in K-12 education. *Communications of the ACM*, *59*(8), 26-27.

Kafai, Y.B., Proctor, C., & Lui, D. (2020). From theory bias to theory dialogue: embracing cognitive, situated, and critical framings of computational thinking in K-12 CS education. *ACM Inroads, 11*(1), 44-53.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads*, *2*(1), 32-37. doi:10.1145/1929887.1929902

Li, Y., Wang, K., Xiao, Y., & Friday, J. (2020). Research and trends in STEM education: A Systematic review of journal publications. *International Journal of STEM Education 7*, 11. doi:10.1186/s40594-020-00207-6

Lin, C. C., Zhang, M., Beck, B., & Olsen, G. (2009). Embedding computer science concepts in K-12 science curricula. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education* (pp. 539-543). New York, NY: ACM.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior*, *41*, 51-61.

Maloney, J., Resnick, M., Rusk, N., Silverman, B., & Eastmond, E. (2010). The Scratch programming language and environment. *ACM Transactions on Computing Education (TOCE), 10*(4), 1-15. doi:10.1145/1868358.1868363

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A Framework for teacher knowledge. *Teachers College Record, 108*(6), 1017- 1054.

Mouza, C., Yang, H., Pan, Y.-C., Yilmaz Ozden, S., & Pollock, L. (2017). Resetting educational technology coursework for pre-service teachers: A Computational thinking approach to the development of technological pedagogical content knowledge (TPACK). *Australasian Journal of Educational Technology, 33*(3), 61–76.

Nadelson, L. S., Callahan, J., Pyke, P., Hay, A., Dance, M., & Pfiester, J. (2013). Teacher STEM perception and preparation: Inquiry-based STEM professional development for elementary teachers. *The Journal of Educational Research*, *106*(2), 157-168.

National Research Council (2012). *A Framework for K-12 science education*. Washington, DC: National Academies Press.

Norris, C., & Soloway, E. (2016). Twelve factors leading to fundamental pedagogical change in a primary school: A Case-study. *Educational Technology*, 25-30.

Papert, S. (1980). *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books.

Ring, E. A., Dare, E. A., Crotty, E. A., & Roehrig, G. H. (2017). The Evolution of teacher conceptions of STEM education throughout an intensive professional development experience. *Journal of Science Teacher Education*, *28*(5), 444-467.

Rich, P. J., Larsen, R. A., & Mason, S. L. (2020a). Measuring teacher beliefs about coding and computational thinking. *Journal of Research on Technology in Education*, *53*(3), 1-21. doi:10.1080/15391523.2020.1771232

Rich, K. M., Yadav, A., & Larimore, R. A. (2020b). Teacher implementation profiles for integrating computational thinking into elementary mathematics and science instruction. *Education and Information Technologies, 25*, 3161-3188. doi:10.1007/s10639-020-10115-5

Rodríguez, F. J., Price, K. M., & Boyer, K. E. (2017). Exploring the pair programming process: Characteristics of effective collaboration. In *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education* (pp. 507-512). New York, NY: ACM.

Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K-12: In-service teacher perceptions of computational thinking. In *Computational thinking in the STEM disciplines* (pp. 151-164). doi:10.1007/978-3-319-93566-9_8

Saubern, R., Henderson, M., Heinrich, E., & Redmond, P. (2020). TPACK–time to reboot? *Australasian Journal of Educational Technology, 36*(3), 1-9.

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review*, *22*, 142-158.

Saldaña, J. (2015). *The Coding manual for qualitative researchers*. London, UK: Sage.

Sanders, M. (2009). STEM, STEM Education, STEMmania. *The Technology Teacher, 68*(4), 20-26.

Smith, M. (2016). *Computer science for all*. Retrieved from https://www.whitehouse.gov/blog/2016/01/30/computer-science-all

Weintrop, D., Beheshti, E., Horn, M., Orton, K., Jona, K., Trouille, L., & Wilensky, U. (2016). Defining computational thinking for mathematics and science classrooms. *Journal of Science Education and Technology*, *25*(1), 127-147.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, *49*(3), 33-35.

Vaden-Kiernan, M., Jones, D. H., & McCann, E. (2009). *Latest evidence on the National Staff Development Council's standards assessment inventory.* Research Brief. Oxford, OH: National Staff Development Council.

Voogt, J., Fisser, P., Good, J., Mishra, P., & Yadav, A. (2015). Computational thinking in compulsory education: Towards an agenda for research and practice. *Education and Information Technologies*, *20*(4), 715-728.

Yadav, A., Hong, H., & Stephenson, C. (2016). Computational thinking for all: Pedagogical approaches to embedding 21st century problem solving in K-12 classrooms. *TechTrends*, *60*(6), 565-568.

Yadav, A., Good, J., Voogt, J., & Fisser, P. (2017). Computational thinking as an emerging competence domain. In *Competence-based vocational and professional education* (pp. 1051-1067). doi:10.1007/978-3-319-41713-4_49

Yadav, A., Mayfield, C., Zhou, N., Hambrusch, S., & Korb, J. T. (2014). Computational thinking in elementary and secondary teacher education. *ACM Transactions on Computing Education (TOCE)*, *14*(1), 1-16.