Exploring Middle School Students' Reflections on the Infusion of CS into Science Classrooms

Mehmet Celepkolu
mckolu@ufl.edu[1]

David Austin Fussell
davidfussell2014@gmail.com[2]

Aisha Chung Galdo
aishagaldo@ufl.edu[3]

Kristy Elizabeth Boyer
keboyer@ufl.edu[1]

Eric N. Wiebe
wiebe@ncsu.edu[4]

Bradford W. Mott
bwmott@ncsu.edu[5]

James C. Lester
lester@ncsu.edu[5]

[1]Department of Computer & Information Science & Engineering, University of Florida, Gainesville, Florida
[2]Department of Chemical Engineering, University of Florida, Gainesville, Florida
[3]College of Education, University of Florida, Gainesville, Florida
[4]Department of STEM Education, North Carolina State University, Raleigh, North Carolina
[5]Department of Computer Science, North Carolina State University, Raleigh, North Carolina

ABSTRACT

In recent years, there has been a dramatic increase in teaching CS in the context of other disciplines such as science. However, learning CS in an interdisciplinary context may be particularly challenging for students. An important goal for CS education researchers is to develop a deep understanding of the student experience when integrating CS into science classrooms in K-12. This paper presents the results of a mixed-methods study in which 75 middle school students engaged in a series of computationally rich science activities by creating simulations and models in a block-based programming language. After two semesters, students reported their experiences on in-class computer science activities through reflection essays. The quantitative results show that both experienced and novice students increased their CS knowledge significantly after several weeks, and a majority of students (72%) had positive sentiment toward the integration of CS into their science class. Deeper qualitative analysis of students' reflections revealed positive themes centered around the visualization and gamification of science concepts, the hands-on nature of the coding activities, and showing science from a different angle. On the other hand, students expressed negative sentiments on weaknesses in the activity design, lack of CS/science background/interest, and failing to make connections between CS and science concepts. These findings inform efforts to infuse CS education into different disciplines and reveal patterns that may foster success of K-12 classroom implementations.

SIGCSE '20, March 11–14, 2020, Portland, OR, USA.

ACM ISBN 978-1-4503-6793-6/20/03…$15.00.
https://doi.org/10.1145/3328778.3366871

CCS CONCEPTS

• Social and professional topics – computing education

KEYWORDS

Middle School; CS+Science; Student Reflections

ACM Reference format:

Mehmet Celepkolu, David Austin Fussell, Aisha Chung Galdo, Kristy Elizabeth Boyer, Eric N. Wiebe, Bradford W. Mott and James C. Lester. 2020. Exploring Middle School Students' Reflections on the Infusion of CS into Science Classrooms. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education (SIGCSE '20), March 11-14, 2020, Portland, OR, USA*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3328778.3366871

1 Introduction

The CS education community has long been investigating best practices to prepare students for the essential skills needed in a computationally dependent world [24][14]. Recent developments in STEM and computer science education emphasize the importance of developing interdisciplinary work skills, in which students learn to meaningfully bridge concepts across different disciplines [12,20]. This approach requires students to learn CS in such a way that they can apply what they learn to different domains such as science [15,16]. This process requires going beyond simply teaching science to supporting students actively investigating a concept and creating solutions to address problems through authentic scientific inquiry [11].

In recent years, both science and CS education researchers have studied how CS and computational thinking can be integrated within the science classroom [26]. While some of these studies were implemented as out-of-class experiences or as stand-alone units [21,22] some specifically investigated the infusion of computational thinking directly into science or other STEM disciplinary courses [4,5]. Despite its effectiveness on the whole, transferring skills between different disciplines can be

challenging particularly for younger learners. An open challenge for CS education researchers is to develop a deep understanding of the student experience in integration of CS into science, because negative experiences during these activities can discourage students from developing a positive STEM identity [25,28] and pursuing STEM as a career [29].

To investigate these phenomena, we collected data throughout two semesters from 75 middle school students who first learned the fundamentals of programming such as variables, conditionals, loops, and object-oriented programming, and then created computationally rich science activities based on the science lesson topic (e.g., light waves, evolution) as part of classroom activities. The analysis in this study focuses on two research questions: *(1) What sentiments do students express toward coding activities during science classes? (2) What themes emerge in students' perception of the impact of CS activities on their understanding of science concepts?*

Analysis of student feedback suggests that the majority of students had positive sentiment toward integration of CS within their science class. Students also reflected on the ways in which computing can show the details of science processes; afford more active learning experiences; and allow them to see science from a different perspective. We also report on challenges expressed by students, which can guide us toward better supporting younger learners in interdisciplinary CS and Science activities.

2 Related Work

There have been important efforts with a recent widespread effort of CS4All [35] to make CS accessible to all students from early ages. Repenning et al. [21] suggested exposing all middle school students to CS by teaching computational thinking in programing environments in which students create games or science simulations. Their findings with more than 10,000 students demonstrate that middle schoolers are at a critical age for learning CS concepts, and with adequate support, they can benefit greatly from learning these concepts at a young age.

Weintrop et al. [30] argues that integrating CT into science can foster reciprocal learning between the two, while bringing science and real world professional practices together. As an example of how reciprocal learning can work, Sneider et al. [26] suggest that simulations in which students can change variables to explore *what-if* scenarios are extremely valuable. These simulation activities are more than simple animations; they are dynamic computer models allowing students to try different experiments, test different conditions and investigate different new outcomes. Simulations can help students for forming a better understanding of phenomena such as natural selection can be difficult to experience or observe directly.

Yadav et al. [34] created a computational thinking module as a core education course required for education majors. Even though only 30% of students initially indicated that there is a relationship between computational thinking and other fields, the number increased to 62% after training. Over 95% of the participants agreed that computational thinking can be integrated into other disciplines.

Similarly, Swanson et al. [27] analyzed reflections from 133 high school students who engaged in computational biology units with the NetLogo application. They found that these activities helped students develop competencies such as identifying the simplifications in the model and modifying models by changing parameters in the code.

Several studies have also suggested using block-based programming environments that allow students to create programs by eliminating issues caused by syntax and providing a simple graphical drag-and-drop interface [18,31]. However, despite the availability of various applications for CS+Science activities, integration of CS with science is still under-investigated [14]. In this study, we build on this related work by examining students' reflections on CS+Science activities developed in the *Snap!* block-based programming language.

Basu et al. [1] found that using a visual-programming system, CTSiM, in which middle school students can create models and simulations illustrating science, was effective for producing learning gains for science topics like ecology and kinematics. Sengupta et al. [23] suggested that through the use of the aforementioned CTSiM system, the development of a long-term curricular progression towards computational thinking is possible without introducing a programming course separately from the science curriculum. Our study seeks to explore this through examining the experiences of middle school students who were taught computer science in conjunction with their science curriculum.

3 Methods

3.1 Participants

The data was collected from 7th grade middle school students in a science course in a public school in the southeastern United States in the 2018-2019 school year. The class was taught by a science teacher and had a total enrollment of 97 students among 5 different class sections. Of these 97 enrolled students, 75 students' parents consented to have their child's data collected for research purposes. Out of 75 consenting students, there were 46 girls (61.3%), 28 boys (37.3%) and 1 unspecified (1.3%). Race/ethnicities were White (46%), Hispanic (19%), Asian (16%), Multiracial (14%), Black (4%), and Other (1%). Participants' mean age was 12.1 and 51% of students reported having had some prior coding experience at the beginning of the semester.

3.2 Procedure

Before the CS activities began (at the beginning of Fall 2018), students completed several surveys and tests. Our goal was to capture students' knowledge of and attitude toward CS before they were exposed to computationally rich science activities. To measure students' CS Technical knowledge, we administered a 17-item knowledge assessment [8], consisting of a combination of multiple-choice and short-answer items involving CS concepts and interpreting block-based code. Students also completed the CS Attitude Survey [33] prior to participating the classroom activities and at its conclusion. This validated survey has five

subscales of which we used three: confidence, motivation and usefulness. Finally, we collected students' demographics information as well as information about their background in CS such as their previous CS and coding experiences.

The research team spent 14 days (6 days in Fall 2018 and 8 days in Spring 2019) in the classroom facilitating the activities. During the first semester activities, students learned the *Snap!* programming language fundamental CS concepts such as loops, conditionals and variables. They created a Light Wave simulation model, which illustrated how the visibility and the color of light changes based on the wavelength value. Students were also asked to integrate nested conditionals into the model to gain further understanding of conditionals and variables.

Learning Gains. At the end of the first semester, students completed the CS Technical assessment and CS Attitude survey again. The pre- and post CS knowledge score and CS Attitude score comparisons showed that students performed significantly higher on the posttest (M=11.4, SD=4.03) compared to pretest (M=7.4, SD=3.22) and the result for both students with prior coding experience and students with no prior coding experience (t(66)=10.88, p=0.001). The results also showed that students with no prior coding experience (M=5.03, SD=3.1) increased their CS knowledge score significantly higher than students with prior coding experience (M=3.38, SD=2.6). This difference is significant (t(64)=2.24, p=0.03). The interventions (CS+Science activities) therefore were sufficient to increase students' CS knowledge significantly. However, despite the slight increase between pre (M=60.86, SD=10.6) and post (M=61.87, SD=11.3) CS Attitude scores, the difference was not significant (t(68)=1.08, p=0.28).

During the second semester activities, students learned more advanced CS concepts such as broadcasting and cloning (object-oriented programming). For example, during the *Evolution* science activities, students modelled evolution processes and showed how small changes (i.e., mutations) can yield significant changes across many generations (Figure 1). Students also made the model parameters more randomized so that each run of the model would generate different results. Although most activities required students to create their own code, we also gave them some simulations related to science (e.g., food web and water cycle) and allowed them test and explore different science topics.

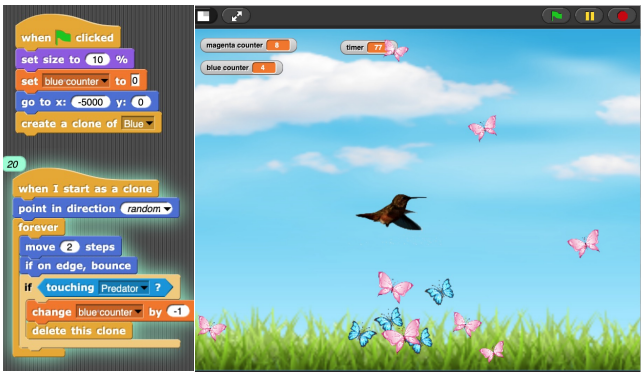

Figure 1: Sample Evolution activity created with *Snap!*

After students completed the CS and science activities at the end of second semester, they were asked to write their thoughts and feelings about CS integration with science classroom with the following high-level prompt: "*In what ways have coding activities helped you understand the science concepts from class?*" We required the responses to be at least 400 characters (about 4-5 sentences) to prevent very short uninformative responses such as "it was fine." As expected, students mentioned a wide variety of topics and reported different sentiments toward CS+Science activities. The average length of the reflections was 109.3 words (min=51 words; max=203 words). Some students were absent during pretest and posttest data collection days, but we included all the data available in quantitative analysis. Out of consenting 75 students, 65 were available during the essay data collection day and this study examines the responses from those students.

4 Sentiment Analysis of Student Reflections

To answer Research Question 1, we first extracted students' responses and applied a manual sentiment analysis method, which labels different emotional states through content analysis of written or spoken [2,13] and has previously been used for analyzing students' reflections in both in-class [6] and online settings [32]. We opted for manual analysis to improve reliability over still-noisy automated sentiment analysis toolkits. We first rated each statement for positive or negative sentiment based on a 5-point scale with 1 being the most negative and 5 being the most positive (Table 1).

Table 1: Sentiment Scores with Sample Excerpts

Sentiment Score	Sample Excerpts
5: Completely Positive	*Coding made science easier to understand because it put the sciencey terms into better context. For example, when we did the beak evolution. The coding process made it simpler and more clear to understand.*
4: Mostly Positive	*It opened up my brain more i guess to possibilities and showed some things that were harder to understand a lot easier. It didn't directly help a lot though and it didn't seem like it made a big difference. I guess it helped a little but not that much*
3: Neutral	*I have learned some things in this class about coding but some of it confused me and i am not sure if i understand coding that much more than i did when i first came here.*
2: Mostly Negative	*I did see some crossover, but it wasn't really teaching us about science. I honestly have never been interested in computer science or coding since I'm not really good with computers.*
1: Completely Negative	*The coding activities have not helped me understand the science concepts from class at all. I already understood the science concepts, so I did not need the coding activities to further demonstrate it.*

To avoid potential rater subjectivity bias, we followed the standard inter-rater reliability methodology: two researchers first independently rated the sentiment of each statement and then met to calculate the interrater agreement score. The ratings resulted in an interrater reliability kappa score of 0.75 and a weighted kappa of 0.9, indicating substantial agreement [17]. Kappa is a statistical measure that is more sophisticated than simple percent agreement calculation because it adjusts for the probably of chance agreement. Weighted kappa takes degree of disagreement into account; if the labels are more different across two raters, the penalty for disagreement is higher. Majority of students (72%) reported positive attitudes (rated as 4 or 5) toward CS integration with science; other students (28%) reported negative (scored as 1 and 2) or neutral (3) attitudes (Table 2).

Table 2: Student Sentiment Scores toward CS+Science Activities

Distribution of Student Sentiment Scores (N=65)				
Completely Negative	**Mostly Negative**	**Neutral**	**Mostly Positive**	**Completely Positive**
17% n=11	3% n=2	8% n=5	14% n=9	58% n=38

5 Thematic Analysis

Research Question 2 involves deep investigation of students' reflections on their perception of the CS+Science activities; thus, we employed the thematic analysis approach, which has been recommended as "a method for identifying, analyzing and reporting patterns (themes) within data" [3]. Using inductive content analysis [10], two researchers first independently open-coded[2] the raw reflection excerpts, and generated a total of 325 independent labels. For example, "coding is enjoyable," "coding enforced learning", and "coding and science seems unrelated" are some initial labels. Next, they collaboratively discussed and merged highly similar labels (e.g., educating and educational), and created a revised set of 182 similar labels. The researchers iteratively collapsed the labels into new higher-level labels, which led to 94 labels. In the last round, the researchers grouped thematically similar labels from this revised set, ultimately identifying six positive and four negative themes.

In this study, positive themes were defined as patterns and opinions that students express favorably toward CS+Science activities. The thematic analysis resulted in six positive themes, which are *enhanced learning*, *showing the processes of science*, *motivating*, *visual modelling of science concepts*, and *showing science content from a different perspective*. In contrast, negative themes reflect patterns and occurrences that students criticize or dislike about CS+Science activities. Four negative themes emerged from students' reflections: *uselessness*, *disassociation of CS and Science*, *weak CS/Science interest* and *activity design*. Table 3 shows the themes, sample labels and the distributions of number of labels in each theme.

[2] Open coding is a qualitative approach of creating tentative labels, or "codes", for small parts of data reflecting the main takeaway. We henceforth use the term "labels" to avoid confusion with the term "code" as in "source code."

Table 3. Themes from the Student Reflections

	THEMES	SAMPLE LABEL	COUNT
POSITIVE	Enhanced Learning	Creating lasting knowledge	**45**
	Showing the Process	Dissecting concepts into understandable parts	**36**
	Motivating	Helping with the science test	**31**
	Visual Modelling of Science	The concepts are not abstract anymore	**13**
	Active Learning	Better than reading the notes	**11**
	Different Perspective of Science Learning	Understanding from a different point of view	**7**
NEGATIVE	Disassociation of Science and CS	Coding and science are different things	**13**
	Utility	Already understood	**12**
	Activity Design	Simple projects	**8**
	Weak CS/Science Interest	Science is not my strong class	**6**

5.1 POSITIVE THEMES

Enhanced Learning: A large majority of the students indicated that using coding in the activities enhanced their learning. Students often mentioned that coding assisted with absorbing information and better grasping an idea, led to creating lasting knowledge, and helped the course content make more sense.

> *"I learned about how evolution works and it improved my test score a ton. Before this coding activity, I didn't really understand evolution but by the end, I could finally understand it!"* - Female student with no coding experience before the class.

Some students find the combination of CS and science interesting and fun, which make it easier to remember the information later.

> *"It opened up my brain more I guess to possibilities and showed some things that were harder to understand a lot easier."* -Female student with no coding experience before the class.

> *"...when we did the evolution code, it really helped me understand how evolution really works. Coding really has facilitated the way I understand science."* -Female student with coding experience before the class.

Showing the Process: Student reported that one of the biggest benefits of coding activities was allowing them to see the key concepts of the science process.

"One time we did an activity with waves. We had the ability to make the waves longer, faster, shorter, or slower. This helped me understand how sound waves and light waves work." -Female student with coding experience before the class.

"I also learned that adaptions take time to occur and when it does it helps animals live and be better in the wild so that they have better chances of survival." -Male student with no coding experience before the class.

Motivating: Students reported that coding activities helped them to develop higher interest and focus more on the activities.

"I think it is also appealing to the student because it is a game and is often entertaining and exciting. The concepts we learn in class such as wave lengths and evolution can be easily and briefly explained with models that are not only educational but are fun as well." -Female student with coding experience before the class.

Visual Modelling of Science: Students reported high appreciation for being able to visualize abstract science concepts.

"…when we did the coding activity for evolution it shows what evolution is without having to imagine it. Though I already knew how evolution worked, it was helpful to see with our eyes what's going on." -Female student with no coding experience before the class.

"Coding activities have helped me understand the science concepts from class better because we are able to see it visually." -Female student with coding experience before the class.

Active Learning: Many students emphasized the benefits of being more active and involved in the learning process compared to their regular class activities such as taking notes or reading.

"I like how we get to actually get to create many different activities where we get to see (for example evolution) in action and get to be more involved in the lesson then just sitting there and doing nothing." -Female student with coding experience before the class.

"Programming makes learning more fun so it is enjoyable and not boring like when you take notes." -Male student with coding experience before the class.

Different Perspective of Learning: Even though physical experimentation of lesson concepts is a common practice in middle school science classrooms, some students appreciated a different form of experimentation.

"For example, the Model of Evolution simulation activity helped me better grasp the notion of natural selection and survival of the fittest when I could see the cloning representing reproduction, the animals appearing and disappearing representing birth and death, and the counters showing me the population and the way it increased and decreased." -Female student with coding experience before the class.

5.2 NEGATIVE THEMES

Disassociation of CS and Science: Some students did not make a strong connection between CS and Science and perceived both fields fundamentally separate from each other.

"Coding doesn't relate to organisms ability to survive in the natural world. Coding seems like nothing more than sitting down while on the computer trying to make something. and life science is how real life organism work." -Female student with coding experience before the class.

"I see no way that coding can possibly help me understand meiosis or mitosis. Or how cells work, including the mitochondria and some other superfluous cells. The coding activities are creative but they do not help with my curricular activities in my class." -Female student with coding experience before the class.

Utility: Some students did not find the activities helpful for their grades or serving to their future goals.

"in a school sense, it isn't doing anything helpful. I see it being useful in a practical sense, but most of us won't have anything to do with coding." -Male student with coding experience before the class.

"It hasn't helped me at all and if anything made me stress more and hurt me because i could've been doing notes." -Female student with coding experience before the class.

Activity Design: One of main challenges during the classroom implementations was to create activities that are not too easy or not too difficult. To achieve this, the activities were presented in several parts (from easy to difficult) so that each student can complete at least some sections even if they had weak coding skills. However, there were still some students who found the activities either too easy or too challenging:

"I feel that what we did was a bit too simple and the activities should a bit harder so people have to actually think." -Male student with coding experience before the class.

"… but the more complicated ones confused me and i didn't know what to do. I feel like if we would have moved a long a little bit slower and took more time on specific concepts and topics." -Female student with coding experience before the class.

Weak CS/Science Interest: Students who had some pre-conceived attitudes toward technology did not enjoy the integration of CS and Science activities.

"Either way science is not my strong class, coding won't help me learn/understand any concepts*."* -Female student with some coding experience before the class.

"I honestly have never been interested in computer science or coding since I'm not really good with computers. I think that coding and the science we are learning now don't really have a relationship." -Female student with no coding experience before the class.

6 Discussion

In this mixed-methods study, we analyzed quantitative and qualitative data collected from 75 middle school students over two semesters. The initial quantitative analysis showed that students significantly improved their CS technical knowledge, and this improvement was even more in students with no prior coding experience. Next, we investigated the first research question and conducted sentiment analysis of students' reflections, which showed that the majority of students (72%) have a positive perception toward the CS+Science activities. In order to reach a better understanding of these results, we investigated the second research question by analyzing the positive and negative themes that emerge from students' reflections. Positive themes centered around enhanced learning and increased motivation for learning the science content through coding. More importantly, integrating CS into science provided students with a way to experiment with details of the science concepts and understand concepts that would otherwise be difficult. Another benefit noted by students is the visual modelling of science concepts and making them easier to comprehend. The age of 11-12 (middle school) is the intellectual evolution from adolescence to adulthood, and children start transitioning between concrete thinking to logical/abstract thinking [19]. In any given middle school classroom, there are likely to be students on both sides of this development milestone. These activities both helped students at the concrete thinking level by allowing them learn concepts such as evolution without having to imagine it, and also helped students at the logic/abstract thinking level by allowing them to experiment with different parts of the algorithm, resulting in different models. Similarly, aligned with previous literature [9], many students appreciated the opportunity for hands-on modelling and experimental learning.

The negative themes may be especially important for informing future efforts. Not all the students perceived CS+Science activities as helpful, and some students in particular did not see the potential for CS to be applied outside of physical science. However, developing activities in those contexts is a promising direction for illustrating the power of computing as a medium for investigating phenomena across many branches of science. Another theme was students' negative perception about the usefulness of these CS+Science activities, and we have the sense that these students' perception was related to the CS+Science activities not being "for a grade," meaning students' success on those activities did not directly impact their success in the deeper. Future, close integration of CS+Science should address this important issue.

Another negative theme focused on different students perceiving the activities as too easy or too difficult. Even though we attempted to create activities with several increasing difficulty levels to mitigate this issue, some students still found these activities inappropriate to their knowledge levels. Individual differentiation is a key issue in all classrooms, and may be especially crucial with CS learning activities where prior experience can vary so drastically among students.

Finally, students with lower self-reported CS or technology interest often expressed negative sentiment toward CS+Science activities. This theme emphasizes the crucial role of ensuring student readiness before implementing any computer related activities. Without fundamental computer knowledge or desire to learn new technologies, it may be difficult to help students understand the connection between CS and science activities.

Limitations and Threats to Validity. The activities reported in this study were implemented in actual middle school classrooms. Amid the richness coming from the classroom studies, there were complicating factors leading to limitations. Some students were absent during some of the important activities, which might have made an impact on their reflections. It is also important to note that all students were enrolled in classes with the same teacher, and classroom climate significantly impacts students' perceptions of work in that class.

7 Conclusion and Future Work

Our overarching goal for this study was to explore the success of CS+Science activities from middle school students' perspectives. The results from a mixed-methods study showed that the majority of the students felt positively toward CS+Science activities, due to benefits such as enhanced learning, visual modelling of science, and active learning; while some students questioned the usefulness of CS in their science classroom and found the activities to be inappropriate to their knowledge levels. The SIGCSE community has long studied best practices for interdisciplinary CS activities, and the outcomes from this study can help to increase enjoyment and improve learning outcomes in CS+Science activities.

For future work, it is important to analyze students' activities during the CS+Science interventions, which can provide even richer information about students' experiences. Also, analyzing the CS+Science artifacts and the steps students took while designing and developing solutions for CS+Science problems can be valuable for further understanding of students' expectations, goals, and concerns with CS+Science activities. It is hoped that this line of investigation can contribute to deep integration of CS into other K-12 disciplines, in order to bring rich CS learning opportunities to all students.

ACKNOWLEDGMENTS

This research was supported by the National Science Foundation through grant DRL-1640141. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the authors, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

REFERENCES

[1] S Basu, A Dickes, JS Kinnebrew, P Sengupta, and G Biswas. 2013. CTSiM: A Computational Thinking Environment for Learning Science through Simulation and Modeling. In *Proceedings of the 5th International Conference on Computer Supported Education*, 369–378.

[2] VK Bongirwar. 2015. A Survey on Sentence Level Sentiment Analysis. *International Journal of Computer Science Trends and Technology* 3(3).

[3] V Braun and V Clarke. 2006. Using thematic analysis in psychology. *Qualitative Research in Psychology* 3(2), 77–101.

[4] PS Buffum, MH Frankosky, KE Boyer, EN Wiebe, BW Mott, and JC Lester. 2016. Empowering All Students: Closing the CS Confidence Gap with an In-School Initiative for Middle School Students. In *Proceedings of the 47th ACM Technical Symposium on Computer Science Education - SIGCSE '16*, 382–387. https://doi.org/10.1145/2839509.2844595

[5] PS Buffum, KM Ying, X Zheng, KE Boyer, EN Wiebe, BW Mott, DC Blackburn, and JC Lester. 2018. Introducing the Computer Science Concept of Variables in Middle School Science Classrooms. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education - SIGCSE '18*, 906–911. https://doi.org/10.1145/3159450.3159545

[6] M Celepkolu and KE Boyer. 2018. Thematic Analysis of Students' Reflections on Pair Programming in CS1. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education - SIGCSE '18*, 771–776. https://doi.org/10.1145/3159450.3159516

[7] M Celepkolu and KE Boyer. 2018. Thematic Analysis of Students' Reflections on Pair Programming in CS1. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education - SIGCSE '18*, 771–776. https://doi.org/10.1145/3159450.3159516

[8] HE Chipman, FJ Rodríguez, and KE Boyer. 2019. "I Impressed Myself With How Confident I Felt": Reflections on a Computer Science Assessment for K-8 Teachers. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education - SIGCSE '19*, 1081–1087. https://doi.org/10.1145/3287324.3287478

[9] S Edwards. 2015. Active Learning in the Middle Grades. *Middle School Journal* 46(5), 26–32.

[10] J Fereday and E Muir-Cochrane. 2006. Demonstrating Rigor Using Thematic Analysis: A Hybrid Approach of Inductive and Deductive Coding and Theme Development. *International Journal of Qualitative Methods* 5(1), 80–92. https://doi.org/10.1177/160940690600500107

[11] EA Forman. 2018. The Practice Turn in Learning Theory and Science Education. In *Constructivist Education in an Age of Accountability*. 97–111. https://doi.org/10.1007/978-3-319-66050-9_5

[12] A Gendreau Chakarov, M Recker, J Jacobs, K Van Horne, and T Sumner. 2019. Designing a Middle School Science Curriculum that Integrates Computational Thinking and Sensor Technology. In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education - SIGCSE '19*, 818–824. https://doi.org/10.1145/3287324.3287476

[13] LA Gottschalk and GC Gleser. 1979. *The measurement of psychological states throught the control analysis of verbal behavior*. Univ of California Press.

[14] S Grover and R Pea. 2013. Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher* 42(1), 38–43. https://doi.org/10.3102/0013189X12463051

[15] M Guzdial. 1994. Software-Realized Scaffolding to Facilitate Programming for Science Learning. *Interactive Learning Environments* 4(1), 001–044. https://doi.org/10.1080/1049482940040101

[16] S Hambrusch, C Hoffmann, JT Korb, M Haugan, AL Hosking, S Hambrusch, C Hoffmann, JT Korb, M Haugan, and AL Hosking. 2009. A multidisciplinary approach towards computational thinking for science majors. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education - SIGCSE '09*, 183-187. https://doi.org/10.1145/1508865.1508931

[17] RJ Landis and GG Koch. 1977. The measurement of observer agreement for categorical data. *biometrics*, 159–174.

[18] SNH Mohamad, A Patel, R Latih, Q Qassim, Liu Na, and Y Tew. 2011. Block-based programming approach: challenges and benefits. In *Proceedings of the 2011 International Conference on Electrical Engineering and Informatics*, 1–5. https://doi.org/10.1109/ICEEI.2011.6021507

[19] J Piaget. 1972. Intellectual evolution from adolescence to adulthood. *Human Development* 15(1), 1–12. https://doi.org/10.1159/000271225

[20] B Priemer, K Eilerts, A Filler, N Pinkwart, B Rösken-Winter, R Tiemann, A Upmeier, and Z Belzen. 2019. A framework to foster problem-solving in STEM and computing education. *Research in Science & Technological Education*, 1–26. https://doi.org/10.1080/02635143.2019.1600490

[21] A Repenning, DC Webb, KH Koh, H Nickerson, SB Miller, C Brand, IHM Horses, A Basawapatna, F Gluck, R Grover, K Gutierrez, and N Repenning. 2015. Scalable Game Design: A Strategy to Bring Systemic Computer Science Education to Schools through Game Design and Simulation Creation. *ACM Transactions on Computing Education (TOCE)* 15(2), 1–31. https://doi.org/10.1145/2700517

[22] SH Rodger, J Hayes, G Lezin, H Qin, D Nelson, and R Tucker. 2009. Engaging middle school teachers and students with alice in a diverse set of subjects. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education - SIGCSE '09*, 271-275. https://doi.org/10.1145/1508865.1508967

[23] P Sengupta, JS Kinnebrew, S Basu, G Biswas, and D Clark. 2013. Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies* 18(2), 351–380. https://doi.org/10.1007/s10639-012-9240-x

[24] A Settle, B Franke, R Hansen, F Spaltro, C Jurisson, C Rennert-May, and B Wildeman. 2012. Infusing computational thinking into the middle- and high-school curriculum. In *Proceedings of the 17th ACM annual conference on Innovation and technology in computer science education - ITiCSE '12*, 22-27. https://doi.org/10.1145/2325296.2325306

[25] V Seyranian, A Madva, N Duong, N Abramzon, Y Tibbetts, and JM Harackiewicz. 2018. The longitudinal effects of STEM identity and gender on flourishing and achievement in college physics. *International Journal of STEM Education* 5(1), 40. https://doi.org/10.1186/s40594-018-0137-0

[26] C Sneider, C Stephenson, B Schafer, and L Flick. 2014. Computational Thinking in High School Science Classrooms. *The Science Teacher* 81(5), 53. https://doi.org/10.2505/4/tst14_081_05_53

[27] H Swanson, G Anton, C Bain, M Horn, and U Wilensky. 2019. Introducing and Assessing Computational Thinking in the Secondary Science Classroom. *Computational Thinking Education*, 99–117. https://doi.org/10.1007/978-981-13-6528-7_7

[28] P Vincent-Ruz and CD Schunn. 2018. The nature of science identity and its role as the driver of student choices. *International Journal of STEM Education* 5(1), 48. https://doi.org/10.1186/s40594-018-0140-5

[29] HC Webb. 2011. Injecting computational thinking into career explorations for middle school girls. In *2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, 237–238. https://doi.org/10.1109/VLHCC.2011.6070410

[30] D Weintrop, E Beheshti, M Horn, K Orton, K Jona, L Trouille, and U Wilensky. 2016. Defining Computational Thinking for Mathematics and Science Classrooms. *Journal of Science Education and Technology* 25(1), 127–147. https://doi.org/10.1007/s10956-015-9581-5

[31] D Weintrop and U Wilensky. 2017. Comparing Block-Based and Text-Based Programming in High School Computer Science Classrooms. *ACM Transactions on Computing Education* 18(1), 1–25. https://doi.org/10.1145/3089799

[32] M Wen, D Yang, and CP Rosé. 2014. Sentiment Analysis in MOOC Discussion Forums: What does it tell us? In *Proceedings of the 7th International Conference on Educational Data Mining*, 130–137.

[33] EN Wiebe, L Williams, K Yang, and C Miller. 2003. *Computer Science Attitude Survey*. North Carolina State University Technical Report TR-2003-1.

[34] A Yadav, N Zhou, C Mayfield, S Hambrusch, and JT Korb. 2011. Introducing computational thinking in education courses. In *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education - SIGCSE '11*, 465-470. https://doi.org/10.1145/1953163.1953297

[35] CSforALL. Retrieved from https://www.csforall.org/