"I Think We Should...": Analyzing Elementary Students' Collaborative Processes for Giving and Taking Suggestions

Jennifer Tsan
North Carolina State University
Raleigh, North Carolina
jtsan@ncsu.edu

Fernando J. Rodríguez
University of Florida
Gainesville, Florida
fjrodriguez@ufl.edu

Kristy Elizabeth Boyer
University of Florida
Gainesville, Florida
keboyer@ufl.edu

Collin Lynch
North Carolina State University
Raleigh, North Carolina
cflynch@ncsu.edu

ABSTRACT

Collaboration plays an essential role in computer science. While there is growing recognition that learners of all ages can benefit from collaborative learning, little is known about how elementary-age children engage in collaborative problem solving in computer science. This paper reports on the analysis of a dataset of elementary students collaborating on a programming project. We found that children tend to make several different types of suggestions. In turn, their partners address those suggestions in different ways such as by implementing them directly in code or by replying through dialogue. We observe that students regularly accept or reject suggestions without explanation or explicit acknowledgement and that it is often unclear whether they understand the substance of the suggestion. These behaviors may inhibit the development of a shared understanding between the partners and limit the value of the collaborative process. These results can inform instructional practice and the development of new adaptive tools that facilitate productive collaborative problem solving in computer science.

CCS CONCEPTS

• **Social and professional topics → Computing education**; **Computer science education**; **K-12 education**;

KEYWORDS

Elementary school, pair programming, collaboration, dialogue.

ACM Reference format:
Jennifer Tsan, Fernando J. Rodríguez, Kristy Elizabeth Boyer, and Collin Lynch. 2018. "I Think We Should...": Analyzing Elementary Students' Collaborative Processes for Giving and Taking Suggestions. In *Proceedings of SIGCSE '18: The 49th ACM Technical Symposium on Computing Science Education, Baltimore , MD, USA, February 21–24, 2018 (SIGCSE '18),* 6 pages.
https://doi.org/10.1145/3159450.3159507

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SIGCSE '18, February 21–24, 2018, Baltimore , MD, USA
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5103-4/18/02...$15.00
https://doi.org/10.1145/3159450.3159507

1 INTRODUCTION

Collaborative problem-solving skills play an important role in computer science education. A rich body of research has studied collaborative problem solving in CS at the undergraduate level [7, 15, 19], and we have seen the emergence of collaborative approaches such as peer instruction [4], process-oriented guided inquiry learning (POGIL) [8], and pair programming [18] that encourage students to work together on problem-solving tasks. In particular, pair programming has been shown to improve students' learning experiences [20], increase student retention in computer science courses [12], and promote good programming practices such as planning a solution before implementing it [14].

In contrast to the research on post-secondary CS learning, there has been comparably less work examining collaboration between elementary computer science students. Recent work has revealed that the equity of a relationship is dependent on how the individuals' personalities impact each other [11, 17], and in a collaborative programming context, students often discuss the problem-solving process and their achievements in that process [9].

In order to effectively support young learners in collaborative computer science learning, computer science education research needs to move toward a deeper understanding of children's collaborative practices. Our work focuses on this question in the context of elementary school pair programming. In a recent study, we found that an imbalance often exists between students not only in their prior experience, but in the extent to which they engage in talk and "driving" (controlling the keyboard and mouse). We have also discovered that suggestions to each other are the most common type of dialogue move our students made.

In this paper, we investigate the phenomenon of making and taking suggestions, addressing the following research question: *How do elementary students make and respond to suggestions during pair programming?* We examined transcripts of pair programming and the video and screen capture of collaborative work from which the transcripts originated. Our review of the dialogue contributions using a case study methodology reveals that students make suggestions through proposals, stronger commands, and suggestions about the next step to take. We also observe at least four types of responses to suggestions: implementing the suggestion (or not), combined with acknowledging the suggestion through dialogue (or not). We examine excerpts from students' interactions that illustrate these different ways of making and acting on suggestions. We hope this work can help educators promote successful collaborative dialogue among their students by identifying best practices, as well as inform instructional practice and the design and development of adaptive tools to support such collaboration.

2 RELATED WORK

This work is directly informed by a body of prior research on collaboration among young learners. This section first reviews related work on collaboration outside of the computer science domain, then focuses on the small but growing body of research on collaboration between young students learning computer science. Note that while the current paper focuses on fifth grade students (approximately ages 9-11), this discussion of related work includes young middle school students as well.

2.1 Children Collaboratively Problem-Solving in Non-CS Domains

Researchers in the learning sciences have actively studied collaborative problem solving in domains outside of computer science, chief among these mathematics. In a highly influential work, Barron studied successful and unsuccessful collaborative interactions among groups of high-achieving sixth-grade students in mathematics using both qualitative and quantitative methods [1]. One finding from that work is that groups which perform better collaboratively are more apt to accept one another's proposals and/or engage in discussion around them, while less successful groups are more prone to reject or simply ignore their partners' suggestions.

In another study of collaboration in math, Olsen and Finkelstein examined the conversations of fourth grade students collaborating on word problems within an intelligent tutoring system [13]. Students were randomly assigned to one of three conditions where they viewed problem-solving example videos: an individual student thinking aloud solving a math problem, a pair collaborating on a problem from an unrelated topic, and a pair collaborating on a math problem. Researchers annotated collaborative session segments based on the quality of the reasoning (talking about the problem) and rapport (synergy between the partners). There were significant positive correlations between the rapport and reasoning state and between the reasoning state and post test scores which indicate that pairs with better relationships may perform better when problem solving.

In a study of how roles naturally form in a collaborative group, researchers evaluated the discourse of a group of sixth grade students in a physics class [5]. Utterances were annotated as conceptual, procedural/negotiation, or social. From their analysis, the authors found that each student had a dominant discourse type, and students who contributed more conceptual utterances, or utterances about the concepts the students were learning, had the highest learning gains.

Downton studied the collaborative processes that unfold during group work on a fourth grade music production task [6], investigating how students reflected on their actions in their dialogue. The excerpt presented by the author illustrated how students reflected on aspects of the music production task and how these spoken reflections would be referenced and built upon by their partners, adding to the collective knowledge of the group.

2.2 Children Collaboratively Problem-Solving in CS Domains

Shah, Lewis, and Caires investigated the equity of elementary students pair programming in a summer CS course [11, 17]. The authors chose pairs of students where one student was in all of the pairs. For these pairs of students, the authors reviewed the distribution of talk, the content of the dialogue, and other key actions that impacted the equity of their relationships. The authors found that the student in common had inequitable relationships with two of his partners and equitable relationships with the other two. This shows that it is not necessarily an individual's characteristics that define the way an interaction plays out; instead, it is the way each individual's characteristics synergize with those of their partners.

In another study of collaborative discourse, researchers gave students the choice to work with others or work independently while solving programming problems and investigated what kinds of collaborative interactions emerged as students worked together in a computing environment. The students shifted between working independently and collaborating with a peer or teacher. The authors found that the majority of the students' interactions were about solving computational problems, off-topic discussions, or expressing excitement about their achievements during the problem solving process [9].

While research on how young students collaborate during computer science problem solving is growing, this is the first work specifically examining how elementary students give and take suggestions while pair programming. Of the papers we explored, Barron's [1] relates most closely to our current work, and we drew from its methodology. By looking into this part of the collaborative process, we can better understand how young learners solve problems together and how we may in turn support them most effectively.

3 DATA AND METHODS

3.1 Elementary CS Elective Course

Over the course of one year, we collected data from a computer science elective in an urban elementary school in the southeastern United States. The data was collected over four 30-day iterations of the class. The course did not exist prior to this research, and its curriculum was designed by the first author in collaboration with another PhD student and the elementary school teacher. In the class, the students learned about computational thinking, programming, computers in society, and other computer science topics. In the programming section, the students were taught basic concepts in Scratch including moving the visual Sprites, taking input, conditionals, and loops. In the middle of the course, students began working on their first programming project which occupied five to seven class periods. For the first programming project, the students were asked to create a program that demonstrated cause and effect using storytelling. The students were required to pair program, and the teacher assigned the students to their partners.

Pair ID	Student Pseudonyms	Gender Composition
1	Charlie and Quinn	Boy, Boy *
2	Ian and Aaron	Boy, Boy
3	Alonzo and Gigi	Boy, Girl *
4	John and Mia	Boy, Girl
5	Greg and Harry	Boy, Boy

Table 1: Identifiers, pseudonyms, and genders of the collaborating pairs. *These pairs had complete videos and screen recordings for two of their pair programming sessions, which we briefly discuss below.

3.2 Participants

Over the course of the year, 55 students signed up to take the class and 26 consented to data collection. We collected video and screen capture data from those students while they completed their projects in pairs. Before the start of the project, the teacher explained the concept of pair programming and the responsibilities of each role. Each pair programming session lasted 30-40 minutes. Due to technical problems, not every video-recorded session had screen recording data. In this work, we analyzed all of the videos and screen recordings we had of students working on their first project, a total of five pairs. Table 1 provides an overview of the five pairs on which this paper's analysis is based.

At the elementary school where we collected the data, the student body is roughly 53% African-American, 33% Caucasian, and 14% Hispanic, Latino, Native American, Asian, or mixed race. Approximately 47.4% of the students receive free or reduced cost lunch.

3.3 Methods

The present work utilizes qualitative analysis of transcribed dialogues through a case study methodology. In prior work, we transcribed the videos collected from the elective described in the previous subsection. We also previously tagged all *suggestion* dialogue moves and found that they comprised 77% of all student utterances.

We focus our case studies on the three minutes of student work that occur at the fifteen-minute mark (approximately halfway through collaboration in one class period) in each pair programming session. At this halfway mark, we are able to see students' collaboration in process because the initial setup is completed, and no pairs had finished their work.

4 CASE STUDIES

In order to answer our research questions, we examined case studies of the five pairs of students in the dataset. This led us to better understand the dynamics of the relationships and areas in which specific interactions could benefit from scaffolding.

4.1 Charlie and Quinn: "Now go to..."

We reviewed both available sessions of Charlie and Quinn pair programming. In the first session, Charlie appeared to be more open to collaborating and explained some of the code to Quinn. However, as time passed, Charlie also appeared to become frustrated with having to collaborate, and at times he did not verbally acknowledge Quinn's suggestions.

Excerpt 1 (Driving: Quinn, Navigating: Charlie)

	[Quinn typed "yes" in the "«answer> = [ ]>" block.]	
154	Quinn	Until equals yes, right?
155	Charlie	I guess so.
156	Charlie	Alright.
157	Charlie	Make everything over.
	[Quinn moved a sprite.]	
158	Charlie	We did it. We did it. It actually worked. No, no go to there. Go switch the backdrop first.
	[Quinn went to the Backdrop tab and changed the backdrop.]	
159	Charlie	Now go to them, and do the (()).
	[Quinn clicked on a sprite and clicked "show".]	

Regardless of which role he was in, Charlie spoke much more than Quinn, and gave many more suggestions than Quinn did. He also showed impatience multiple times when Quinn was driving, asking when it would be time to switch roles. We noticed that Charlie often gave specific suggestions in a series, and Quinn often followed those suggestions without verbally acknowledging them. Their behavior indicates that Charlie was the dominant member of the pair. Excerpt 1[1] further illustrates this point.

Excerpt 1 took place over approximately 30 seconds. In this time, Quinn made one specific suggestion at turn 154 while asking for confirmation from Charlie when he says, "...right?" When Charlie verbally accepted the suggestion during turn 155, Quinn continued working and implemented it. Once Quinn was done, Charlie then gave him a string of suggestions in turns 158 and 159, which Quinn followed without speaking. The suggestions that Charlie gave were very direct and confident, leaving no question of what he wanted Quinn to do. In fact, these suggestions felt more like commands.

At first blush, we might conclude that Charlie and Quinn worked relatively well together. They did not argue (a common problem for young collaborators), and they filled their respective roles roughly as expected. However, they were missing a crucial part of the collaboration process: discussion. The lack of discussion meant they simply followed one person's vision without establishing common ground [2] and may not have benefited as much from the collaboration as they could have. Throughout both sessions, Quinn implemented most solutions as a driver without questioning Charlie. He was also a quiet navigator, especially in the second session where he seemed more distracted and withdrawn. There was usually more discussion during the rare times Quinn disagreed with Charlie, but in most cases, Charlie stood firm and seemed to have the final say.

4.2 Ian and Aaron: "I know."

In this pair, Ian seemed to prefer working independently while Aaron appeared to enjoy collaborating. Aaron gave the majority of the suggestions, and Ian followed some and ignored others, often without verbally acknowledging them. Other times, when he acknowledged Aaron's suggestions, he said, "I know," before continuing to work. It is worth noting that in our previous analysis, we observed that there was one session where Aaron did not drive at

[1](()) is used in place of inaudible speech.

all. In this session, which occurred the day after, Aaron only drove for a few minutes before Ian became frustrated and took the mouse.

In Excerpt 2, Ian made some high-level suggestions, then solicited advice by saying he did not know how to implement those solutions. Aaron then gave him some low-level suggestions, which Ian did not verbally acknowledge, but implemented. When Ian implemented the suggestions, he did not seem to find what he needed, so he went to the "Scripts" category instead while stating it out loud. While Ian soliciting advice was a positive action to take, this was another instance of how a lack of discussion is a failure to establish common ground.

Excerpt 2 (Driving: Ian, Navigating: Aaron)

	[Ian opened their program, which he had closed when the pair went to talk to their teacher.]	
163	Ian	I think we should do this.
164	Ian	And then we have the house hide.
	[Ian moved an existing block slightly and gestured towards a sprite using his mouse.]	
165	Ian	I don't know how to.
	[Ian went to the Costumes tab.]	
166	Aaron	You gotta go to stage.
	[Ian went to the Scripts tab then went to Stage.]	
167	Aaron	Then you go to backdrops.
	[Ian went to the Backdrop tab.]	
168	Ian	(()).
169	Ian	Scripts.
	[Ian went back to the Scripts tab.]	

4.3 Alonzo and Gigi: "You're mean."

Alonzo and Gigi generally worked well together. In some ways, Gigi seemed more dominant than Alonzo, but Alonzo gave more suggestions than she did. Gigi often followed those suggestions without disagreeing with him. Alonzo's suggestions were often specific and she implemented them without problems. However, during the few times she disagreed with him, he stood his ground and they implemented his suggestions

In Excerpt 3, Alonzo gave Gigi suggestions and, similar to Charlie, Alonzo gave suggestions that were closer to commands. Then Gigi disagreed with one and said, "No," during turn 255. Alonzo responded by firmly stating that they were going to implement his idea in turn 256. Gigi protested again, but as she protested, she started to implement the change he suggested. In the end, despite the number of protests she made, they settled with his idea. This is an example of an interaction where a verbal acknowledgment and protest was not enough. The students would have benefited from self-explanation [3], where each student presents an argument for or against the idea that was presented.

Excerpt 4 illustrates Alonzo and Gigi deciding which sound to add to their program. Gigi made a suggestion in turn 440. In turn 442, Alonzo responded by telling her to wait and that he was thinking about what they needed. He did not implement her suggestion; instead, he continued searching and made his own suggestion, to which Gigi agreed. Instead of discussing Gigi's suggestion and why he did or did not agree with it, he decided not to implement it and came up with his own suggestion.

Excerpt 3 (Driving: Gigi, Navigating: Alonzo)

	[Gigi went to the Backdrops tab.]	
253	Alonzo	Now go to mine.
	[Gigi changed the backdrop and moved a sprite.]	
254	Alonzo	My drawn one.
	[Gigi moved a different sprite.]	
255	Gigi	No.
256	Alonzo	Yes bruh, we're using that one.
257	Gigi	The ugly one?
	[Gigi goes back to Stage.]	
258	Alonzo	Yes.
	[Gigi changes the backdrop.]	
259	Gigi	You're mean.

Excerpt 4 (Driving: Alonzo, Navigating: Gigi)

	[Alonzo is looking in the Sounds library.]	
439	Gigi	What- whatever, (()) just pick one.
	[Alonzo continues looking.]	
440	Gigi	Growl. Let's (()) growl.
441	Gigi	(())
442	Alonzo	Alright, wait. I'm tryin think of what we need bruh.
	[Alonzo goes to the animal category.]	
443	Alonzo	Alright, what do we need? An animal noise, right?
444	Gigi	Yeah.

This case study shows examples of interactions where a verbal acknowledgment and protest was not enough. In Excerpt 3, Gigi disagreed with Alonzo but conceded after he insisted multiple times. In Excerpt 4, Alonzo pushed off her suggestion to think and she did not mention it again. The students would have benefited from self-explanation, where each student presents arguments for or against the ideas that were introduced.

4.4 John and Mia: "What are you doing?"

In our previous study, we observed that John gave more suggestions than Mia did. Our current observations of their conversation and programming process revealed that Mia often did not react positively to John's suggestions or give reasons for disagreeing with him. John also did not give reasons for his suggestions.

Overall, the relationship of these two students seemed much more competitive than collaborative. Mia often attributed the problem to her partner rather than the task at hand [10]. They would sometimes even flaunt when one partner proved the other was incorrect. Additionally, neither student explained their thoughts and ideas and Mia was not open to her partner's suggestions. This undesirable behavior is illustrated in Excerpt 5.

John was making changes to the code when Mia noticed he was doing something she did not expect. Since he was working silently, she did not understand why he was making those changes and asked him what he was doing in turn 219. As John continued to work without answering her, she made a suggestion in turn 220. Once again, John did not answer and Mia repeated her question. Since John did not acknowledge her questions or suggestions and appeared not to have implemented her suggestion, Mia was obviously frustrated. It was not until the teacher asked John to explain what he was doing to Mia did John start talking to her again.

Excerpt 5 (Driving: John, Navigating: Mia)

	[John is making changes to the code.]	
219	Mia	What are you doing?
	[John continues to work without saying anything.]	
220	Mia	We have to do the same thing that we did for the first one!
	[John goes to the Sensing category.]	
221	Mia	What are you doin'?
	[John grabs an <Ask> block.]	

4.5 Harry and Greg: "No, 'cause..."

Harry and Greg were both very engaged and excited about their project. While they did not always agree with each other's ideas, they tended to follow similar trains of thought. This pair of students had one of the most successful collaborative interactions we observed from the participants. There were times when Harry had to make the same suggestion multiple times before Greg would implement it, but they had at least one situation where they gave a reason for disagreeing with each other. This situation is shown in Excerpt 6.

While Harry was working on a task in Excerpt 6, Greg gave a general suggestion in turn 255. Harry acknowledged the suggestion but disagreed and gave a reason for it in turn 256, which Greg accepted. Even though Greg had already accepted the explanation, Harry continued to explain his thoughts in turn 257, which Greg also agreed with. This interaction is encouraging, as previous work has found that students who approve suggestions or enter into discussion have more successful collaborations [1].

Excerpt 6 (Driving: Harry, Navigating: Greg)

255	Greg	How about when his shoes?
	[Harry modifies the x coordinate in a <Go to x:[ ] y: [ ]> block.]	
256	Harry	Touches black. No, cause then he'll be like that.
257	Greg	Yeah.
258	Harry	((He'll be like)) that looks better.
	[Harry modifies the y coordinate in a <Go to x:[ ] y: [ ]> block.]	
259	Greg	Okay.

5 DISCUSSION

Collaboration is often required in the classroom and workplace, especially in computer science. Our analysis revealed areas in which collaboration was successful and unsuccessful in an elementary computer science setting. A successful programming collaboration is one in which students are able to engage in meaningful discussion to create a program that integrates ideas from all students in a group. During the meaningful discussion, students may use various collaborative dialogue practices, including, but not limited to, self-explanation [3], question generation [16], attributing challenges to the task [10], and building on each others' ideas [2]. Students entering an elementary computer science classroom may not have had collaborative experiences or scaffolding in the past; therefore it may be beneficial to explicitly teach them collaboration and pair programming practices. Because the success of the students' collaboration process is likely to affect how much they learn, our results have implications regarding how students might better learn to collaborate and how we should design collaborative activities to foster good practices.

Type	**Example(s)**
Proposal	"So this should be no."; "How about let's test?"; "Let's try moving him up some."; "...I think we should take the ask block out."
Command	"When backdrop changes to wood, glide right here in front of her, right there. That's one thirty-four, negative one thirty-five."; "Get the apple out."; "Go to Jack."; "Now let's go to scripts."
Next Step	"We just n– need to make that turn into a horse."; "...she needs to lay down and ... she needs to wake up."; "... when backdrop changes to woods, they have to not be showing."; "And then we have the house hide."

Table 2: The list and examples of suggestion types.

A large number of the student utterances were suggestions, and previous work indicates that suggestions are vital to a collaborative interaction [6]. We identified three types of suggestions and responses from the excerpts discussed here (Table 2). A *command* is a direct instruction to the partner. A *proposal* is a suggestion that is detailed enough to inform the receiver of what exactly needs to be done next but less direct than a command. A *next step* suggestion is a more high-level suggestion, indicating a task that may need to be completed next or in the near future.

From reviewing the interactions in the excerpts we found four types of responses students had to their partners' suggestions. The response types were as follows: implement and verbally acknowledge; implement and do not verbally acknowledge; do not implement and verbally acknowledge; and do not implement and do not verbally acknowledge. Of these, perhaps the most productive response type is *implement and verbally acknowledge* because it is clear that the student understood the partner's suggestion and chose to follow it.

Implement and do not verbally acknowledge may still be a productive response because it at least implies that the partner heard and accepted the suggestion; however, without verbal acknowledgement it is unclear whether she actually heard her partner or if the action she tooks was her own idea. In addition, if she heard the suggestion and implemented it without any discussion, it is difficult to evaluate whether she understood the reason for the action.

Do not implement and acknowledge is another type of response we observed. Although choosing not to implement a suggestion is an implicit rejection of the idea, the verbal acknowledgment of the suggestion confirms that the partner actually heard it and that it may turn into a fruitful discussion about the suggestion's merit.

Do not implement and do not verbally acknowledge is when a partner does not acknowledge a suggestion, it is unclear whether she did not hear it, or the partner chose to ignore it. Such a response will likely deter the speaker from making another suggestion.

In an empirical study, Barron [1] found that students were more successful in their collaborative process if they accepted or discussed suggestions more than if they rejected or ignored them. Most commonly, the students in our dataset do not verbally acknowledge suggestions, whether they implemented them or not. Our observations and Barron's results support the notion that these actions prevent the students from having, through self-expression[3] and grounding [2], the meaningful discussion that is vital to the collaboration process. Therefore, it is important to encourage students to explain their process to their partners.

To improve students' collaborative interactions, we need to design activities that foster good collaboration practices. These activities may scaffold effective collaboration practices. While many activities require students to collaborate, they may not include the scaffolding necessary to mediate such interactions. Further research must be done in this area to find the best approach, but possible solutions could be as simple as having the students explain their thoughts to each other during specific points in the activities.

6 CONCLUSIONS AND FUTURE WORK

Collaboration and its role in learning has long been a topic of research. There have been countless studies about collaboration practices in computer science, such as pair programming, for undergraduate students. However, much is unknown about how elementary students collaborate while solving programming problems.

In our previous work, we collected a rich dataset from an elementary computer science elective and annotated students' collaborative dialogue for certain dialogue acts, including suggestions. Our current work extends this, taking a closer look at the context of the suggestions and the actions following those suggestions through a case study analysis. From this work, we identified at least three types of suggestions and four types of responses to those suggestions. In addition, we observed that more often than not, students in our dataset either accepted or rejected suggestions from their partners without verbally acknowledging them. Behaviors like this may hinder the collaboration process and the computer science education research community should continue investigating the complex phenomena associated with collaboration in elementary school. These investigations may include several open questions. For example, how soon do partners implement suggestions: do they implement them immediately after a suggestion or do they finish their current task first? How do partners solicit ideas from each other, and how often do they ask? Finally, how do students ask for help: do they ask for a high-level explanation or do they want the answer? By understanding the ways students collaborate and by scaffolding their interactions, we will help students have a better experience learning computer science and help them develop a skill they will use for life.

ACKNOWLEDGMENTS

This work is supported in part by the Wake County Public School System, Google through a CS Capacity Research Award, and the National Science Foundation through grants CNS-1453520 and DRL-1721160. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

[1] Brigid Barron. 2003. When smart groups fail. *The Journal of the Learning Sciences* 12, 3 (2003), 307–359.

[2] Courtney B Cazden and Sarah W Beck. 2003. Classroom discourse. *Handbook of Discourse Processes* (2003), 165–197.

[3] Michelene T.H. Chi, Nicholas Leeuw, Mei-Hung Chiu, and Christian LaVancher. 1994. Eliciting self-explanations improves understanding. *Cognitive Science* 18, 3 (1994), 439–477.

[4] Catherine H. Crouch and Eric Mazur. 2001. Peer Instruction: Ten Years of Experience and Results. *American Journal of Physics* 69, 9 (2001), 970–977.

[5] Catherine Dornfeld and Sadhana Puntambekar. 2015. Emergent Roles and Collaborative Discourse Over Time. In *Proceedings of the 10th International Conference on Computer Supported Collaborative Learning (CSCL)*, Vol. 1. 380–387.

[6] Michael P. Downton. 2015. "That's What Everyone Else Is Saying...": Collaborative Reflection-in-Action during Creative Activities. In *Proceedings of the 11th International Conference on Computer-Supported Collaborative Learning (CSCL)*. International Society of the Learning Sciences, 31–38.

[7] Katrina Falkner, Nickolas J.G. Falkner, and Rebecca Vivian. 2013. Collaborative Learning and Anxiety: A Phenomenographic Study of Collaborative Learning Activities. In *Proceedings of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. 227–232.

[8] John J. Farrell, Richard S. Moog, and James N. Spencer. 1999. A Guided Inquiry General Chemistry Course. *Journal of Chemical Education* 76, 4 (1999), 570–574.

[9] Maya Israel, Quentin M. Wherfel, Saadeddin Shehab, Oliver Melvin, and Todd Lash. 2017. Describing Elementary Students' Interactions in Puzzle-based Environments using the Collaborative Computing Observation Instrument (C-COI). In *Proceedings of the Thirteenth Annual International Conference on International Computing Education Research*. ACM, 110–117.

[10] Harold H Kelley. 1967. Attribution theory in social psychology. In *Nebraska Symposium on Motivation*. University of Nebraska Press, 192–238.

[11] Colleen M Lewis and Niral Shah. 2015. How Equity and Inequity Can Emerge in Pair Programming. In *Proceedings of the Eleventh annual International Conference on International Computing Education Research*. ACM, 41–50.

[12] Clem O'Donnell, Jim Buckley, Abdulhussain E. Mahdi, John Nelson, and Michael English. 2015. Evaluating Pair-Programming for Non-Computer Science Major Students. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education (SIGCSE '15)*. 569–574.

[13] Jennifer K. Olsen and Samantha Finkelstein. 2017. Through the (Thin-Slice) Looking Glass: An Initial Look at Rapport and Co-Construction Within Peer Collaboration. In *Proceedings of the 12th International Conference on Computer-Supported Collaborative Learning (CSCL)*. International Society of the Learning Sciences., 511–518.

[14] Leo Porter and Beth Simon. 2013. Retaining Nearly One-Third More Majors with a Trio of Instructional Best Practices in CS1. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education (SIGCSE '13)*. 165–170.

[15] Fernando J. Rodríguez, Kimberly Michelle Price, and Kristy Elizabeth Boyer. 2017. Exploring the Pair Programming Process: Characteristics of Effective Collaboration. In *Proceedings of the 48th ACM Technical Symposium on Computer Science Education (SIGCSE)*. 507–512.

[16] Carolyn P Rosé, Johanna D Moore, Kurt VanLehn, and David Allbritton. 2001. A comparative evaluation of socratic versus didactic tutoring. *Proceedings of Cognitive Sciences Society* (2001), 869–874.

[17] N Shah, C Lewis, and R Caires. 2014. Analyzing equity in collaborative learning situations: A comparative case study in elementary computer science. In *Proceedings for the 11th International Conferences of the Learning Sciences (ICLS)*. 495–502.

[18] Laurie Williams. 2001. Integrating Pair Programming into a Software Development Process. In *Proceedings of the 14th Conference on Software Engineering Education and Training (CSEE&T '01)*. 27–36.

[19] Laurie Williams. 2007. Lessons Learned from Seven Years of Pair Programming at North Carolina State University. *SIGCSE Bull.* 39, 4 (Dec. 2007), 79–83.

[20] Laurie Williams, Eric Wiebe, Kai Yang, Miriam Ferzli, and Carol Miller. 2002. In support of pair programming in the introductory computer science course. *Computer Science Education* 12, 3 (2002), 197–212.