June 2015

Problematizing Best Practices for Pairing in K-12 Student Design Teams

Gina Quan
University of Maryland, College Park

Ayush Gupta
University of Maryland at College Park

Andrew Elby
University of Maryland, College Park

Problematizing Best Practices for Pairing in K-12 Student Design Teams

Ms. Gina M Quan, University of Maryland, College Park

Gina Quan is a doctoral candidate in Physics Education Research at the University of Maryland, College Park. She graduated in 2012 with a B.A. in Physics from the University of California, Berkeley. Her research interests include understanding community and identity formation, unpacking students' relationships to design, and cultivating institutional change. Ms. Quan is also a founding member of the Access Network, a research-practice community dedicated to fostering supportive communities in undergraduate physics departments, and an elected member of the Physics Education Research Leadership and Organizing Council (PERLOC).

Dr. Ayush Gupta, University of Maryland, College Park

Ayush Gupta is Research Assistant Professor in Physics and Keystone Instructor in the A. J. Clark School of Engineering at the University of Maryland. Broadly speaking he is interested in modeling learning and reasoning processes. In particular, he is attracted to fine-grained analysis of video data both from a microgenetic learning analysis methodology (drawing on knowledge in pieces) as well as interaction analysis methodology. He has been working on how learners' emotions are coupled with their conceptual and epistemological reasoning. He is also interested in developing models of the dynamics of categorizations (ontological) underlying students' reasoning in physics. Lately, he has been interested in engineering design thinking, how engineering students come to understand and practice design.

Andrew Elby, University of Maryland, College Park

My work focuses on student and teacher epistemologies and how they couple to other cognitive machinery and help to drive behavior in learning environments. My academic training was in Physics and Philosophy before I turned to science (particularly physics) education research. More recently, I have started exploring engineering students' entangled identities and epistemologies.

Problematizing Best Practices for Pairing in K-12 Student Design Teams

Introduction

Research on group work in STEM education has documented that in some cases, students' relative expertise with respect to other group members can impact student participation in the discipline: expert-like students can help novice-like students gain conceptual understanding,[1] the success of pairings depends on the complexity of the task at hand,[2] and group dynamics and roles impact local equitable access to disciplinary knowledge and identities.[3] Within engineering education, Tonso highlights how roles and gender dynamics within design teams shape whether and how students' contributions to the design project are recognized.[4] Given this, group composition in design teams becomes an important instructional decision since access to disciplinary knowledge and identity can influence students' future trajectories. However, mechanisms by which roles affect broader relationships to design are underexplored. Our aim is to understand students' emergent roles in design teams, and how this may or may not interact with their complex relationships (epistemological and affective) to computer programming and design. We unpack how pairing students of different levels of expertise influences students' access, their sense of whether or not they can participate in a discipline. We suggest that to consider whether a student widened or narrowed their access to a discipline, their "expert" and "novice" roles in a team may not have enough predictive power. In this paper, we trace how access is also significantly impacted by students' identities, relationships between partners, and students' senses of knowing and learning within the discipline.

Best Practices for Roles

Group composition with respect to varying skill levels among students is an important instructional decision for team-based class projects since that, in combination with other factors, can influence group dynamics on a team. A student's positioning within a design team can be consequential to their participation in the team, their contributions being recognized, and their future participation in design.[4] Research on student pairings for teams often makes contradictory suggestions. For example, expert students often differentially gain more expertise than novices[2] and can even leave novice students behind.[3] Other work suggests that high-achieving students can benefit from being in groups of high-achieving students.[5] Work that builds off of Vygotsky suggests that working with expert students benefits novices, due to their being able to achieve more difficult tasks in their zone of proximal development.[1,2] Other work suggests that the presence of experts in collaborative groups can prevent equitable access to knowledge and positions.[3] While much of the work on roles based on skill-level distinctions studies students' in-the-moment equity, we choose to focus more deeply on how these roles in a design team impact a student's sense of long-term access to the discipline.

Positional Dynamics and Access

Within this paper, we discuss students' access to a discipline and access to practices within a discipline. By access, we mean the students' sense of whether they can participate in a discipline or practices within a discipline, either in the moment (for example, if an individual feels she could write a program as part of an immediate task) or at a future time (for example, if the individual feels she can "learn" to program or see pursuing a discipline in future as a viable option). In some sense, this broad notion of access overlaps with constructs such as self-efficacy

with particular tasks[6] and identity[7] within a discipline. We feel that the broadness of the construct, which invites us to examine interactions among self-efficacy, identity, and other contributors to access, can be useful in understanding students' sense of their future trajectories, especially as gleaned from interviews and classroom video. We are careful to tease apart access from students' competence and enjoyment, as a student's sense of alignment with normative disciplinary identities is not necessarily tied their enjoyment of, competence in, and epistemological beliefs about the discipline.[18]

The notion of access is also closely tied to recruitment and retention, in the sense that if a student feels a lack of access to a discipline, they are less likely to persist within or pursue that discipline.[8] Within STEM education research, there is consensus that students' early experiences affect whether they persist in STEM fields. For instance, within engineering education, Stevens et al. followed students' trajectories through their undergraduate years and found that students' early successes, even small ones, are consequential to their "becoming an engineer." [9] These experiences color their perceptions of engineering and their sense of self as engineering students (and future professional engineers). Early experiences also have a cascading influence via roles and opportunities available to students in the future.

While Stevens et al. attended to the macro scale dynamics of students trajectories, Tonso investigated the micro-dynamics of students' socially constructed roles within classroom engineering design activities.[4] The role of a student within a design team, her positioning with respect to other members of her team, and ultimately her experience of design were often strongly constrained by the broader school culture and the social roles of the student outside of class. This in turn influenced how centrally students participated in activities of design and were positioned as central contributors to the team's product. A similar theme comes from Boaler.[10,11] She found that students' decisions to study mathematics were strongly influenced by whether the classroom expectations of what it means to learn and practice mathematics were aligned (or not) with their identities as learners. Margolis and Fisher explored ways in which typical school culture, in particular "geek" practices produced by men, misaligned with the ways women expressed their enthusiasm for coding.[12]

Integrating the findings from these lines of research would suggest that the specific roles that students take up within early design experiences are coupled to their success and whether they and others see them as central players, which in turn could influence future access to engineering. But these links are still underexplored in research.

In this paper, we focus on students' sense of access to computer programming. We choose this focus because of the high-status nature of programming and persistent cultural beliefs of who can do it.[13] We analyze two pairs of students. In one pair, both students had a programming background (expert-expert). In the other, one student was more novice compared to her partner (expert-novice). We use "expert" to refer to having some experience coding, whereas "novice" refers to students who had no experience coding. In the expert-expert pair, students' complex identity negotiation, students' sense of what counts as programming, and the nature of design tasks led to neither student gaining (nor losing) access. Within the expert-novice pair, the more experienced programmer almost singlehandedly programmed yet the less experienced student reported gaining access to programming through learning from her partner.

These cases illustrate that group interaction patterns and access are not simply dependent on students' roles, knowledge, and personalities. It is also necessary to take into account other aspects of students' experiences, such as their relationships to their partners and how a sense of what it means to be a "programmer" develops in the learning environment.

Classroom Background

We designed and ran a project-based instructional module within Summer Girls, a two-week day-camp for high school students hosted by the University of Maryland. The module was piloted in Summer 2013, and small modifications were made and implemented in Summer 2014. As part of the camp, students worked on Arduino (microcontroller) controlled robot-tanks (henceforth, "Arduino-bot"). Roughly 1-2 hours per day were dedicated to Arduino-bot activities. The rest of the time was spent on modern physics lectures, lab tours, and demonstrations. The Arduino classroom was structured to promote student agency. Throughout the program, participants worked in groups of twos or threes through several open-ended Arduino design tasks before designing and completing a final project using Arduinos. The camp was co-taught by two instructors. Each day, 2-3 graduate student and undergraduate volunteers came to help students with their projects. Students were also given a reference library of sample code, and were strongly encouraged to use the internet.

We recruited all classroom participants to participate in the study. The day camp was free to participants and intended to serve students in the Washington, D.C. Metropolitan Area. Students completed a short application about their interest in physics and were selected based on interest in science and their likelihood to benefit. Students did not know ahead of time that they would be programming or doing Arduino, so they did not self-select for or against programming.

Methodology

Over two iterations of camp, we collected interviews, coursework, and classroom videotapes of focal groups. Due to limited resources, we filmed one pair in the pilot year. This pair of students was interviewed together during the second week of camp. In the second iteration, we filmed two pairs and two trios during design activities in the classroom. In the second year, interviews occurred at the beginning of the camp, beginning of the second week of camp, and after camp ended. Because interviews were voluntary, only two students in the second year completed all three interviews, but most completed one or two. In both years interviews focused on students' perceptions of programming, their experience working on design tasks, experiences working in their team, and their thoughts about future trajectories. Classroom videotapes were collected of all Arduino activities and some additional in-class activities. We also collected some written work, such as students' daily written feedback, from all consenting participants. For this paper, we selected two pairs with different levels of expertise in programming, to explore different ways that student roles impact access.

Interviews were watched and content logged to identify themes for analysis. After we generated preliminary descriptions of students' roles and access, we watched classroom video data to identify new patterns and to confirm and disconfirm our initial explanations. We content logged classroom video data to generate a chronological sequence of events in each group. Key events where group roles were particularly salient or seemed to shift were examined in more detail. We presented our analyses at multiple University of Maryland Engineering Education group

meetings, to ensure the consideration of multiple interpretations, and identify interpretations supported by the greatest fraction of data.

We draw on tools from interaction analysis[14] to look for evidence of role uptake through speech, gestures and actions. We do an in-depth analysis of both classroom and interview data, as both are needed to understand students' roles, how they made sense of these roles, and how these roles impacted access to various aspects of the design project. As in Stevens, et. al.,[9] we conducted interviews of the same student at multiple points during the camp, to draw out the most salient aspects of their experiences. As in Carlone,[15] our analysis necessitated unpacking how participants made sense of particular disciplines (e.g. What does it mean to do programming?) in order to understand whether or not they had access. Students' messy views of what "programming" is can influence whether or not they saw themselves as doing programming, and whether or not they had access to programming. For example, a student pseudocoding an algorithm on a design team might not recognize her contributions to developing a program. Our second case study illustrates the role that views of programming can have. We start, however, with a case study that illustrates other points.

Hazel and Olive

The example of Hazel and Olive, two students in the Summer Girls program of 2013, problematizes simplistic stories of the relation between role-taking and students' access with respect to programming. In their group, Hazel took the lead on the programming tasks while Olive focused on mechanical tasks. For example, Hazel programmed the Arduino-bot to self-navigate out of an enclosed area while Olive built a mechanical arm for the Arduino-bot to throw confetti. Hazel entered the camp as a relative expert; she had taken multiple programming classes and had entered computer programming (C++) competitions, whereas Olive had no coding experience.

Roles Impacting Participation for Olive

Olive attributes a shift in access to coding to her partnership with Hazel.

> Olive: It was really difficult [to learn coding] because I didn't actually have no idea what we were doing. And so I was glad to have Hazel as a partner because she helped me.
> Hazel: I hope I explained /Olive: You did, you did explain/[1] what I was doing...
> Olive: Yeah um, without her helping me, I actually had no idea what we were doing.

Hazel is positioned (by both Hazel and Olive) as the programming expert who almost singlehandedly executed the programming tasks, while Olive is positioned as the novice whose contributions were minimized.

On the one hand, we might think that Olive's membership on a cohesive team that did lots of programming would unproblematically help her gain access to programming. On the other hand, we might think that Olive's positioning as a complete novice, with Hazel as the expert who actually did all the programming, would deny Olive access to programming, taking away valuable opportunities for Olive to engage in programming and develop a sense that she could continue to program. But what we see is more nuanced and complex than either of these

[1] / indicates overlapping speech

"extreme" narratives. Within the earlier quotation we note the harmonious student-tutor positioning between Hazel and Olive that positively contributed to Olive's learning. It also contributed to her relationship to programming in the future:

> Olive: I had no idea what coding was… And something new that I didn't think I'd be open to, and now that I'm learning more about it, I want to learn more, like have more insight and know more about it cause it's something I obviously didn't know. And she was a good explainer...I think she was probably the best one to explain it to me.

Olive's description of being more open to programming, and not knowing what coding was before the camp experience, indicates a shift in access to the discipline. She ties her desire to want to learn more to Hazel being "a good explainer."

We can speculate several mechanisms by which the expert-novice pairing could have helped Olive gain access to coding. Olive could have developed a sense of vicarious self-efficacy, seeing Hazel as a capable peer and thereby being able to envision herself as a programmer.[6] The low-stakes classroom environment could have also provided a place for Olive to feel comfortable trying out (and being successful at) programming. Maintaining team harmony could have also discouraged Hazel and Olive from leaving one another behind. Another possibility is that Hazel scaffolded Olive's understanding of more complex coding ideas. Though we believe that many of these mechanisms (and more) likely interacted with Olive's sense of access, in this paper we focus on Hazel's scaffolding of content and processes, which enabled Olive to shift from more peripheral to more central participation in programming slowly but without losing a sense of agency. In the next sections, we present a broad description of how Olive's participation shifted as a result of Hazel's scaffolding. Then, we zoom in to look more closely at episodes at the beginning, middle, and end of their project to illustrate Olive's shifts in participation and the role that Hazel played in those shifts. Finally, we look more closely at their descriptions of their teamwork to highlight how compassion was critical to Olive's participation. We suggest that Olive's participation, scaffolded by Hazel, was an important part of her gaining access to coding.

Olive's Shift in Participation: Overview

Across five days of working together on their final project, we see Hazel guiding Olive's growing participation in programming. Though Olive never independently writes code, she gradually becomes more involved in code writing. Their group work also shifts from Hazel watching Olive closely and checking in with her, to Olive working more independently.

When they first start working together, they immediately begin by finishing a task in which the Arduino-bot was supposed to self-navigate a maze, using code that Hazel had been working on. This establishes an uneven footing from the start, because only Hazel is familiar with the algorithm, and it is unclear whether Olive had started working on this task. From the start of this clip, Olive asks questions while Hazel narrates what she is doing. Olive also asks Hazel to explain things when she is confused and Hazel responds with explanations.

In the same day, an instructor asks them to upload a program to another group's Arduino-bot. Olive attempts this task alone, unsuccessfully. Hazel then completes the task alone. After this, Hazel does more checking in with Olive, asking her if ideas make sense. Hazel's explanations to Olive are presented colloquially, reflecting Hazel's awareness of Olive's lack of experience. For

almost all of the coding in the first two days, Hazel types the code while Olive looks on attentively, sometimes with Hazel narrating her actions. Olive's contributions are mainly brainstorming ideas for the final project and helping to Google questions.

On the second day, they begin putting together the mechanical arm. Olive immediately takes the lead in constructing it, though Hazel watches attentively and makes comments. After they finish building the gearbox, they try to figure out how to hook up the arm to the breadboard using a complicated circuit. For the first two days, interactions with student helpers consist of Hazel talking to the helper while Olive watches attentively.

On the third day, Hazel continues to partially narrate her coding, and Olive starts to offer more suggestions. Olive still seems engaged and makes some comments about the project: "Isn't the transistor supposed to be connected?" "Yay! No error!" "I'm a little confused." Hazel then asks Olive to start adapting some code from someone's cell phone to the computer. Hazel keeps a close watch, giving direct feedback. Though Olive is mostly copying lines directly, this is the first time we see her enter code.

Olive then starts to ask helpers questions, rather than asking Hazel questions after the helpers have left. It is not clear whether she's gaining comfort asking questions or if she's generally more comfortable with those aspects of the project. At the end of the day, Olive remarks that tomorrow, they'll have to figure out how to attach the arm. That signaled a sense of co-leading their group project, indicating that Olive also feels a sense of agency in the project.

Olive embraces the role of arm-builder. She shares her ideas with Hazel, and Hazel takes them up, e.g., "That looks very promising." Though their work is siloed, they are engaged in each other's activities. Hazel still explains some code to Olive.

On the last day, as they wrap up their project, Olive had a more central role in the coding process. Olive unproblematically verbalizes code, and contributes debugging suggestions. At a few points, Hazel asks Olive to make changes ("Can you fix that delay?" "Let's see if it'll blink") and Olive makes those changes. These instances of Olive editing code feel drastically different from copying code on Tuesday, in that the task is not rote, and Hazel no longer watches Olive closely to make sure she is doing it correctly.

In brief, Olive showed evidence of being engaged in the project throughout the five days while progressing toward more central participation in coding. In the next section, we look more closely at episodes in the beginning, middle, and end of the project, to more clearly illustrate the different ways Olive participated.

Day 1: Early Expert/Novice Positioning

In the beginning of the project, Hazel sits in front of the computer and is the lead typist. She often explains bits of what she is doing to Olive, who asks questions while she's working. Their conversation is also marked with small celebrations "Yay, it's uploaded!" and "Done, yay!" In these instances, Hazel is driving the project, but Olive is still engaged.

After they spend some time working, an instructor comes over with another group's Arduino-bot, trying to see if a working program will run on it or if the Arduino-bot itself is broken. Hazel

is down on the floor with the Arduino-bot, so Olive starts to help the instructor. Olive finds that the computer is having trouble communicating with the Arduino-bot. Across the span of two minutes, Olive spends time searching for the 6th COM (computer) port, not knowing that the Arduino program would automatically assign a new port to the new Arduino-bot. She tries solutions that had been working before, trying to select other COM settings in the Arduino program and moving the USB cable to different ports. But these moves don't work. The instructor offers to let Olive go and work with Hazel on the ground. Shortly after, Hazel walks up and asks what is going on.

> Olive: I switched it to a different COM, the wire.
> Hazel: Oh don't do that. [Chuckles] Dooon't do that. Um, put it back on COM6
> Olive: Cause we couldn't find COM6
> Hazel: Yeaaah- [Clicking]
> Instructor: There's no COM6
> Hazel: Cause sometimes you have to undo the thing. Oh, let's put it on COM7 cause we didn't have that one before.
> Hazel: Yeah, oh. There you go.
> Instructor: Thank you, now I'll go see if I will not have to take it apart.
> Olive: So, what's the COM now?
> Hazel: It's still, I don't know. I want to try it with the spin kinda thing…

Hazel's ability to complete the task, especially following Olive's not being able to, reinforces Hazel's expert role. Olive had spent almost two minutes trying to figure out how to upload the program, and Hazel comes up and does it immediately. To clarify, Olive asks what the COM is now, and Hazel doesn't really answer the question and immediately moves onto the next task. She then rearranges items on the table, moving the mouse and keyboard in front of her.

Day 3: Growing Participation

In the third day, another student has downloaded a motor-running code onto her phone from the internet. Hazel brings the phone over so they can try the code to run the mechanical arm motor. Hazel asks Olive to type in the code as written on the screen. This is the first time Olive types up code in their partnership.

> Hazel: Just the stuff in comments.
> [Hazel pointing to screen]
> Hazel: You're not in void, you don't want it in comments
> [Hazel reaches over and hits backspace, Olive keeps typing]
> Hazel: You want it in, no, in here, in these brackets [pointing at screen]
> Hazel: Okay. Sorry.
> [Olive typing]
> Hazel: You don't want it in comments.
> Hazel: Olive, you do not want it in comments [Olive keeps typing]
> Hazel: There you go. Now you're good.
> Hazel: They have it in comments cause they don't need it. We're going to use it.

Though Olive is only copying lines from somewhere else, this participation is more central than just making comments and asking questions. Hazel keeps a close watch, giving direct feedback.

Later in the day, Hazel spends some time teaching Olive about comments, and gives advice on how to type more efficiently.

Hazel role as a teacher was reinforced in this episode. She had asked Olive to take on this task, which was simple enough for Olive to do with help. Hazel pays close attention to what Olive is doing and gives immediate feedback, including statements of affirmation when Olive is able to do something. We see Hazel as scaffolding Olive's participation in programming, by initially assigning her a coding task which is mostly complete, and helping her finish the rest. In interviews, Olive credits Hazel for helping her gain access to programming. Because Olive was participating in new ways, we see this instance as one in which Olive gained greater access to coding through Hazel's help.

Day 5: More Central Participation

On the last day of working together, Olive had a more central role in the coding process. Hazel is typing, but Olive starts to chime in with verbalized code.

> Hazel: Err, just go forward for 500, cause why not. First take uh--
> Olive: Else, Stop and then-
> Hazel: Yeah. I haven't actually tried stopping. Err, yeah I did, it didn't work out okay. So.
> Olive: Should we put the other variables from this one too?
> Hazel: This has no variables.
> Olive: Oh.
> Hazel: Oh we *do* need this statement, thank you.

In the second line, Olive is verbalizing parts of the algorithm while following along with Hazel. Olive seems to be one step ahead of Hazel, since Hazel says she had not done that part yet. This indicates that Olive is able to follow along with the code and figure out what needed to get done next. She also makes a suggestion to include other variables from a prior code, which Hazel initially dismisses, but then takes up, and acknowledges.

Olive later edits parts of the code as they debug the program. While debugging, they move back and forth between making coding changes on the table and testing the program with the Arduino-bot on the floor. Hazel asks, "Can you fix that to delay?" and Olive adds a delay to the code and uploads it, without needing to ask for help. Hazel's request "Can you fix that to delay" is much more ambiguous than the structured transcribing of code Olive did on Day 3. This signals Hazel's confidence that Olive will be able to complete the task. Olive also unproblematically implements the delay and uploads it, signifying her growing competence and participation in coding.

Maintaining Collaboration in the Team

As the above sections show, Hazel enacted her teacher role not just through enacting expertise but also through attentiveness toward Olive having a positive experience. In this section, we further support this point by looking at classroom and interview data in which Hazel expressed concern about Olive's participation. At the beginning of the third day, Hazel reflects on how she had done most of the work the day prior. Olive had started putting together the extra arm for the Arduino, using old cardboard and Kinex parts.

Hazel: I couldn't even do that, I'm so bad at cutting things sometimes. I'm sorry, I think I'm awful for saddling you with that stuff.
Olive: Yeah, Yesterday I was just like, do you feel like I'm holding you back?
Hazel: No, no, no, I feel terrible that you thought that at all. I'm just a bad partner. I was just very frustrated yesterday, with all the (Inaudible) at all and I'm very sorry if I came off like that.
Olive: It's okay, it's okay
Hazel: I just get frustrated when things don't work and they should work. I kinda get trapped in that and I kinda stop looking at how to actually make it work, and I get really dismissive of suggestions to make it work
(Olive laughs)
Hazel: Even though we need those because, if we're just stuck on it doesn't work, then it won't work and sometimes it takes me days to realize that. So thank you for sticking with me and I'm sorry I'm such a bad partner.
Olive: Oh no you're not a bad partner.

Up until this point, they hadn't (on video) explicitly addressed the fact that Hazel was doing most of the project. Hazel seemed surprised by Olive's suggestion that she was holding her back, and attempts to mollify Olive's self-consciousness by saying that she was just being a bad partner. Olive accepts Hazel's apology by saying "it's okay, it's okay," and disagrees with Hazel that she was being a bad partner. In this section, Hazel is taking responsibility for their group's inequity, blaming it on her frustration. She also welcomes Olive's contributions, positioning Olive as a valuable contributor. This same sentiment comes out in their interview, which happens later that day.

.

Hazel: I hope I explained what I was doing. /Olive: You did you did explain/ Cause I just went off on my own /Olive: Yeah she did, well/ a lot, I felt bad about that.
Olive: Well that was something too.
Hazel: I felt bad about that I was like oh no, I totally forgot she was there.

These discussions about their group work support the classroom observations that Hazel took the lead on the project, and Olive took a supporting role. From Hazel's embarrassment about leaving Olive behind, we see that she clearly cares about Olive's experience.

That Hazel cared about Olive's participation is critical toward Olive's growing access to coding. Olive was able to take on gradually more complicated tasks because Hazel was able to break them off into manageable chunks for her. Hazel also gave a considerable amount of feedback at the early stages of Olive's work, and frequently validated the importance of Olive's contributions. This contrasts with previously documented instances in which more expert-like students take on "explaining" roles in groups while neglecting the understanding of their peers.[3] Based on classroom data from days 1 and 2 and on some interview snippets, we might have characterized Hazel as playing this kind of dominant explaining role. However, over this longer timescale, we see Hazel as what Vygotsky refers to as a more competent peer, scaffolding Olive's learning, assessing what she is capable of in the moment, and giving feedback.[1]

In the Discussion below, we explore how the case of Hazel and Olive shows mechanisms by which a "novice" can gain access to disciplinary practices such as programming from an "expert," under the right conditions. But first, we turn to our second case study.

Bianca and Coral

Our next case study comes from the design team pair of Coral and Bianca who participated in the Summer Girls camp in 2014, one year after Hazel and Olive participated. Coral and Bianca both had some prior programming experience and displayed a good level of proficiency in completing the Arduino programming activities. However, unlike Olive, we did not see any shifts in their sense of access towards programming. In their pre- and post- interviews they maintained a relatively constant enthusiasm and interest in coding, and they did not show evidence of coming to think of themselves as more competent programmers. Both entered having had other robotics experiences and wanting to pursue STEM, and both left still wanting to pursue STEM. What was surprising, however, was that even though they both contributed significantly to the coding tasks over the two weeks of camp, each described the other as having done the bulk of the coding. We see this lack of self-recognition as potentially problematic for their being able to see themselves as capable programmers and gain increased access to programming. If students don't recognize their participation in and contribution towards disciplinary activities, that could potentially negatively impact their identification with and persistence within a discipline.[9]

Another important point illustrated by Coral and Bianca is that we as researchers cannot infer increased access to programming from even the most overwhelming evidence that (i) the group was well-functioning with respect to the project as a whole and with respect to programming in particular, and (ii) both participants were very competent at programming the Arduino. Other factors can "overcome" these positive aspects of their experience, as we discuss below.

Uptake of Roles in the Classroom: Overview

Bianca and Coral start working together from the beginning of camp. They only work with different students on the second day of Arduino when explicitly instructed to mix up their groups. From day 1, they work collaboratively. For example, when one is typing and clicking at the computer, the other watches, comments, and asks and answers questions. In the first week, they tend to alternate usage of the computer keyboard every 3-10 minutes evenly, with the exception of one one-hour chunk during which Bianca exclusively worked on typing and modifying the program. They also frequently both make suggestions while coding and ask clarification questions. At the end of the first week, they brainstorm ideas for their final project. They decide to build a dancing baby, with the body of the baby built out of cardboard and a separate Arduino-bot controlling the motion of each foot. They are both immediately enthusiastic about the dancing baby and seem enamored by the uniqueness of their idea; many other groups chose projects based off of earlier Arduino activities from the first week of camp.

They begin working on their final project in the beginning of the second week. Their interaction patterns on the first day of the final project resemble those in the first week—frequent questions and suggestions as well as frequent shifts in who was typing. Bianca takes the lead on one aspect of the project, having one Arduino-bot also play a nursery rhyme from a speaker, based on her music knowledge. Coral, who strongly identifies as a "builder," takes the lead on any mechanical tasks related to the Arduino-bot.

About halfway through the second day of the final project, they split off into two roles. Bianca makes the cardboard baby while Coral codes the feet motion. They stay in these roles throughout the rest of the project. This shift is initiated when Coral happens to be troubleshooting mechanical problems with one of the bots and asks Bianca if she can start on the baby. The shift seems to emerge pragmatically, as they realize they have multiple steps to complete and limited time. It also seems to happen by chance, in the sense that they had been working on mechanical tasks at that moment and Coral was the lead on mechanical tasks. This left Bianca with the baby construction. To divide the tasks, they physically separate, with Bianca's workspace on the floor and at another table. They stay separated for the rest of the project, with Bianca working on the cardboard baby, and Coral finishing up coding the Arduino-bots. In this stage, they check in periodically about each other's progress, and in some cases Bianca yells out suggestions when Coral gets stuck. Bianca's suggestions can be viewed as bids for her participation in that aspect of the project, and Coral takes up most of Bianca's suggestions.

Collaborativeness and fluid roles: Zooming in

To illustrate the collaborative nature of Coral and Bianca's interactions, we present two classroom data clips from the first week of the program, before Coral and Bianca had started working on their design project. In the clips, they seamlessly and frequently change control of the mouse and keyboard. They constantly bounce ideas off of each other, and are engaged in what the other is doing, regardless of who is at the keyboard.

When Bianca is at the computer, she tends to frequently check in with Coral about what she is typing. In this clip, they're trying to get an LED (light emitting diode) to turn on when the distance sensor indicates a distance less than 15 inches.

> Coral: You have to put another else-if statement
> Bianca: But else- but if none of these are true it says, else, turn off
> Coral: Oh yeah, I don't know why it's not working then
> Bianca: We could put another one
> Bianca [mumbling quietly]: let's put another one [Bianca types]
> Coral: No, but that's redundant
> Bianca: Well. Redundantly redundant? [Bianca smiles]
> Coral: Huh?
> Bianca: Nothing.
> [Coral Chuckles]
> Bianca: Else if the distance is greater than [Bianca types] uhh
> Coral: Digital write LED Low
> Bianca: Do we have to put a bracket?
> Coral: Yeah.

In this segment, Bianca checks in with Coral about the code while she is typing. Coral watches, engaged, and offers corrections as Bianca types. Coral also does some dictating of the code as Bianca is typing, indicating her attentiveness to what Bianca is doing on the computer.

While Coral is typing at the computer, she seems to invite fewer opportunities for Bianca to contribute. Nonetheless, Bianca watches attentively, and Coral takes up Bianca's contributions in conversation. Bianca also uses the mouse in this episode.

[Bianca looks at screen while Coral types]
Bianca: Setup?
Coral: Maybe?
[Bianca clicks compile]
Bianca: Expected, oh you have to do- initialize
Coral: Oh, initialize
Bianca: Before there, I think.
Coral: Hold on I'm gonna delete that, go up here, and try-
Coral: What do I have to initialize?
Bianca: Initialize... [clicks] motor, oh yeah I think you have to initialize these things cause those are the things we're controlling right? so.
Bianca [looks at Coral]: Or is that not how this works I have no idea.
Coral: Yeah it is.

Bianca's way of giving input positions Coral as having more expertise. Bianca asks a clarification question while Coral is typing: "Setup?" She also has more hedges, such as "I think" and "I have no idea." Based on Bianca's framing of her suggestions as questions, she could be positioning Coral as having more expertise; but Coral challenges that positioning in a few ways. Coral validates Bianca's expertise when Bianca expresses doubt ("I have no idea") by agreeing with Bianca's suggestion: "Yeah it is." Coral also gives consideration to all of Bianca's suggestions in this episode.

In summary, from the classroom data we have presented, one might expect these two students to have gained access to coding through participating in Summer Girls. Across the first week, they both contributed significantly to the programming tasks, both spending a lot of time at the keyboard. In the first week, they struggled through and completed tasks that required difficult[2] coding that they had not done before. One might expect their sense of accomplishment to lead to greater access through increased self-efficacy and positive emotions. In the second week, Coral ended up doing most of the programming, so we might expect her access to programming to shift in the second week as well. In the next section, we show that their access to programming did *not* seem to shift. We then identify plausible mechanisms for the lack of shift, by looking at their descriptions of coding, their identities in other settings, and the complex nature of design tasks.

Shifts (or not) in Access to Coding

Throughout camp, Coral and Bianca both stayed interested in STEM and considered engineering a possibility, but did not seem to grow in their sense of access to coding. Bianca entered the camp knowing she wanted to pursue a STEM field but was having trouble narrowing it down. She described having many family members in STEM jobs, such as medicine and engineering, and could see herself in those jobs. In the pre-interview she described having some prior experiences with programming.

Interviewer: Yeah so have you done programming or engineering stuff before?

[2] We are characterizing these tasks as difficult because at their grade level most students haven't yet handled programming tasks that require coordinating the program, an Arduino micro-controller, and sensors and actuators. As computing becomes more common-place in earlier grades in the future, this characterization is likely to change. But currently, these tasks are difficult even for some first-year undergraduate students as per our experience.

> Bianca: Um, I did GEMS last year, it's Gains in Engineering Mathematical Skills and um the topic of the program that I was in was robotics, so we did do a little bit … It was really cool. We did - it was different from the Arduino that I was doing here, it was a program where you like click and drag things and then it goes through and you can add it and do different things so it was like for the robot, so we made it like pick up things, move things, we put like a light sensor,
> Interviewer: So did you like programming?
> Bianca: Yeah I thought it was fun. I thought it was like a puzzle, you do it and you like see if it works and if it didn't work, you're like, 'dangit!' so you have to fix it.

Bianca here expresses that she has some access to programming; she has written code before and found it enjoyable. As noted above, we might expect her accomplishment with even more complex coding tasks in Summer Girls to make her feel more capable of programming. Instead, in the post-interview Bianca's description of programming emphasizes that she has room for improvement, and she does not yet classify herself as a capable programmer, despite doing much of the programming herself.

> Bianca: [Programming] is definitely something I can work on. Because I do think it's gonna come up again. I don't think I'm going to do it once here and never again. So I think it's important that I figure out how to do all that stuff and I like improve upon at least a little bit because if I had no idea how to do it, that's gonna be a problem.
> Interviewer: Sure. Do you have any ideas how you would do that?
> Bianca: I mean, I guess there's always like, I could always take a computer science class or something like that.

Bianca has not lost access; she still sees programming as a discipline she can learn. However, this sense of basic competence is accompanied by a sense of deficit in her programming skills as she emphasizes how much she has left to learn. Overall, this description of Arduino programming does not have markers of enjoyment, while in the pre-interview she described programming as "fun."

Bianca also positions herself as more novice than Coral. In the post interview, she describes her relationship with Coral as helping her program:

> Bianca: Yeah, Um, I think I did learn, I learned stuff about it. Um, I was asking questions trying to read the program after she made it and she would tell me why this works and this is how this has to happen… There was a lot of complicated stuff that went into that. But then she, like, taught it to me. \Interviewer: Cool\ So, I think I know how to do it. \Interviewer: That's great\ I could probably do it if I tried to do it on my own \Interviewer: Cool\ Yeah.

Bianca highlights Coral's contributions to programming, and credits Coral for helping her learn "stuff about" programming. Bianca's last statement, that she could probably do it on her own if she tried, reflects some sense of access to programming as a discipline. This sounds quite similar to what Olive said about Hazel. However, unlike Olive, Bianca often took the role of the lead programmer, and there weren't as many instances of Coral teaching. That Bianca doesn't recognize that she did a significant amount of programming seems problematic toward her

trajectory in the discipline, as she may continue to see herself as a novice, even if she continues to develop expertise.

We now turn to Coral and her lack of a shift toward greater access to programming. In Coral's pre interview, she already expressed a strong sense of what career she wants to pursue. She described wanting to be a mechanical engineer and having attended other camps related to engineering. Coral also identifies herself as a builder, and on her school's robotics team, she feels more competent building rather than programming. To her, a lack of background or coursework in programming prevents her from taking on a programming role, and she says she would be able to program after taking a formal class.

By the mid-camp interview, we don't see much of a shift in access. She describes the Arduino component as "brushing up on skills" and "adding to skills," suggesting a lack of increased access. She also mentions robotics engineering as a career path, which didn't come out in the pre-interview. It's not clear if this is a new interest or if she just didn't mention it before. She also expresses comfort with needing to be able to program in the future.

> Interviewer: So uh, in your life in the future, do you see programming as being a part of what you would be required to do?
> Coral: Definitely. Yeah I know I'm going to have to program, especially if I go the route of robotics engineering. But I- I don't know that I want to necessarily major in computer programming. But I know that it's a skill that I need to have. And it's going to help me a lot. Yeah.

In the post-interview, she describes how her past experience was important to her participation in Summer Girls, and says that if she had not done any Arduino previously, she would have had a more difficult time in camp.

In summary, neither Bianca nor Coral displays a shift toward being more of a programming person or shifts in self-efficacy toward or interest in programming. So, we see no evidence of increased access to programming. In the next section, we suggest that each participant, in identifying herself as the lesser programmer, cut off opportunities for her own growth in access. (The direction of causation runs the other way, too; we are suggesting a negative feedback loop.)

Identity Interacting with Access

We see Coral and Bianca's identities, or sets of identifications both self-described and constructed by others,[7,16] as being a critical piece of the story of access. Consistent across multiple interviews, Bianca and Coral describe how their partner was the primary coder. For example, in Bianca's mid-camp interview she discusses how Coral was the primary coder. At the time of this interview, they had spent an equal amount of time coding, so it is interesting that Bianca focuses on Coral's contributions to coding.

> Interviewer: So what is your group like for the Arduino project?
> Bianca: It's just me and Coral. So, it's only two people but I mean, it's fun, I think we work well together ... we kind of have like similar knowledge of everything so we're pretty equal. So we're not really that different so. She does a lot of coding cause she's done Arduino before. So she does a lot of that. I think I help with, like, figuring out like

> what we need to get it to do, and then I tell her this is what should happen, and she's the one who translates that into Arduino code.

Bianca describes her role in the group as coming up with ideas, whereas Coral does the majority of the "coding." Bianca's contrast of coding with idea generation suggests that she is conceptualizing "coding" narrowly as the act of producing written syntax but not inclusive of the ideas and algorithmic thinking preceding that act. In the next section, we explore how her notions of coding also interacted with access.

In Bianca's interviews, Coral's role as the coder is so salient that Bianca expects that division to continue through the final project.

> Interviewer: So, do you think that you guys will do, you guys will take over different roles in this [final] project?
> Bianca: … I don't think our group is gonna function differently cause Coral still knows more about coding than I do. And so I'll probably help saying like what we need to change and she'll take that and change it.
> Interviewer: What do you think would happen if you kinda switched places and you coded more?
> Bianca: We'd have to fix the code a lot more if I did it first. Because, I'm just not as familiar as she is at coding Arduinos and things like that. I could, I mean, it might take a lot more tries and like a lot more effort for me to do it but I think I could probably do it, probably.

Going into the second week, Bianca sees her novice programmer role in her partnership as remaining fixed. At the end, we see Bianca tentatively saying that she could also program in her group. This suggests that she perceives some access to programming, but her hesitation and hedging indicates less confidence in her ability to do it well.

In her post-interview, Bianca reiterates that Coral contributed most of the coding. That she described Coral as the primary coder in the post-interview is much less surprising, because Coral did most of the programming in the second week. Bianca justifies their roles.

> Bianca: I think we both knew what we were capable of doing and we used that and we fit it into our project and I mean, there were a bunch of different parts to our project, so, there was a way to utilize each of our talents to the best of our ability so we could get it done efficiently.

Unlike Hazel and Olive, we never see Bianca and Coral explicitly discuss roles with one another in the classroom or interview data. Bianca suggests that the split emerged out of utility. Given the chance nature of how the split emerged, it is possible that Bianca's description of Coral being more capable is a post-hoc justification of why they split. It also could be the case that their separate skills sets were considered in their implicit decision-making. Either way, their roles reinforced Bianca's sense of herself as a novice programmer. The data seem to suggest that Bianca's positioning of herself and Coral within the team, with Coral as the better programmer, influenced (and was influenced by) her sense of confidence in programming and, by extension, to her sense of access with respect to computing.

We now switch to how Coral perceived their relative positions on the team. At the end of camp, Coral suggests that Bianca may have taken up a bigger programming role. In the post-interview, Coral emphasizes the collaborative nature of their group and describes her role as the "building circuit side."

> Coral: Umm, I don't think there were roles that were set in stone where it was like you're going to do this and I'm going to do this… she might have done a little bit more of the, like, writing the code, but I think we came up with like the ideas and what type of statements to use together. \Interviewer: Ok\ and then I might have done like a little bit more building the circuit but we both kind of like figured out what needed to be connected where together.

Coral's description contradicts both the classroom video and Bianca's description. What is most salient in her description is that they had equal access to roles, and she suggests that if anything, Bianca may have been the primary programmer. Coral focuses on how collaborative they were, while not acknowledging the ways she contributed to coding.

Coral and Bianca's lack of identification of their programming accomplishments accompanied their identifying the programming accomplishments of the other. For each, not recognizing her own accomplishments was likely part of a lack of growth in access; identification of programming accomplishments is part of seeing oneself as a competent programmer. Bianca, for instance, easily identified Coral's coding skills and saw her as a stronger programmer. This is consistent with work by Fields and Enyedy which highlights the dependency of one's own identities on how one constructs the identities of one's teammates.[13] Similarly, Coral reflected most on Bianca's programming contributions, and thus did not describe being a good programmer herself.

Nature of Coding

In this section, we describe how Bianca's sense of coding as a discipline could have impacted her sense of access to that discipline. In her pre-interview, she referred to her participation in a different engineering camp, using Scratch, as "programming." She also described some of what programming meant to her in that context.

> Interviewer: So did you like programming?
> Bianca: Yeah I thought it was fun. I thought it was like a puzzle, you do it and you like see if it works and if it didn't work, you're like, 'dangit!' so you have to fix it.

In the pre-interview, programming seems to be like a fun puzzle to Bianca. After Summer Girls, her description of whether her Scratch experience was or was not programming seems more uncertain.

> Interviewer: Does this- um, and this is maybe one of your first experiences coding?
> Bianca: Yeah, it's- Yeah pretty much. I mean, I did one but it wasn't the same kind of coding, it wasn't words it was pictures and you like click and drag but it was, it was programming, still, kind of.

Bianca hedges "kind of" when tagging the GEMS camp as programming. Her experiences with Arduino likely impacted her views on what it means to do programming, and could be why she is more hesitant to call what she did before "programming." For example, Arduino is text-based, rather than image-based, and looks more like something a computer programmer would use. Bianca may be seeing the more complicated programming language as more authentic, and this could be why Bianca is reluctant to identify herself as a "programmer." By contrast, Coral sees Arduino coding as *less* authentic than what she had encountered in her robotics club; she refers to it as "pseudocode" and suggests that she will need to take a course in programming to take on a programming role in robotics.

In brief, both Bianca and Coral have conceptions of "real programming" that blocked greater access to programming in Summer Girls. Programming in the Arduino IDE (integrated development environment) stood out as being more authentic to Bianca than programming in the Scratch environment, making her reevaluate whether her enjoyment of and skill with Scratch in the context of the GEMS camp should count as evidence that she is a programmer. Coral considered Arduino work to be "pseudocode," which she (unlike professional programmers) did not consider part of real programming—and hence no amount of skill or enjoyment with Arduino could contribute to increasing her access to programming.

Complex Arduino Tasks

As our last point, we argue that the nature of the design task can also prevent students from gaining access to programming. As in typical design tasks, the Arduino environment required students to integrate multiple sub-tasks related to construction, programming, circuit-design and circuit-construction. The open-ended nature of the tasks also left many potential paths for them to approach these sub-problems. This was particularly evident in situations where something didn't work as intended and they had to figure out which aspects of the project needed troubleshooting.

Though we do not see either student gaining access in programming, we saw them shift along other dimensions. One aspect of the Arduino project most salient to Coral was that she did a lot of building for the group. Often she had to troubleshoot mechanical problems, and she was given many opportunities to be a builder. She was also recognized for being a building expert by instructors, who would hand tools directly to her. Coral's progression along a builder trajectory is important, even though it kept her from gaining programming access. Coral also gains a slight preference for studying robotics in college, indicating some access toward that discipline. Bianca saw herself as the person who designed the baby. The baby itself was a great point of pride for Bianca; they were recognized by classmates in classroom videos as having a creative, funny project. Because the baby emerges as a source of pride, and Bianca was recognized for her creativity by others, we argue that Bianca was able to grow in her access to crafts in design. But with time constraints, growth in access to one disciplinary practice can—and apparently did—come at the expense of growth in access to other disciplinary practices.

Discussion

Our main argument is that the mechanisms by which a student does or does not gain access to a discipline (or specific disciplinary practices) are more complicated than explanations predicated on initial expert-novice conditions. This work sheds light on how access might come about, and which factors besides expert-novice positioning might matter.

Of course, no researchers claim that expert-novice considerations are *all* that matters; and we are not claiming that those considerations play *no* role. Our argument is more nuanced: other factors may exert equal or greater influence than expert-novice factors do in determining to what extent a student gains access. For instance, some research on group work suggests that pairing novices with experts lets novices learn through doing more than they are capable of doing alone. In the case of Hazel and Olive, we see that Olive does indeed gain access through working with Hazel, but the mechanism is not simply Hazel's expertise enabling Olive to enter a sociocognitive zone of proximal development. Olive's participation was made possible by the fact that Hazel also took up the teacher role and cared deeply about Olive's growth, scaffolding Olive's growing access in ways that go beyond the purely sociocognitive. Hazel scaffolds Olive's learning through giving her manageable chunks of tasks *and conveying her confidence that Olive could complete them*, giving her constant feedback *with encouragement*, and assessing Olive's understanding. That Hazel valued Olive's learning also showed itself in her concerns about leaving Olive behind. Their compassionate relationship, by this argument, made it possible for their expert-novice pairing to lead to access for Olive.

Other research suggests that students in expert-expert pairings have higher achievement gains than students in expert-novice pairings, because students in expert-expert pairings give and received more high-quality help.[5] Though we do see that Coral and Bianca both made significant programming contributions and helped one another learn, we also need to consider how their experience impacted their sense of identity and epistemologies. Despite Coral's and Bianca's prior experiences coding, neither seemed to identify herself as a programmer (largely because of their narrow conceptions of what counts as "programming"), and this lack of identifying with programming prevented them from gaining additional access to programming (beyond the limited access they had previously gained). So, this case study illustrates the importance of students' recognizing their accomplishments as belonging to a discipline for their disciplinary identity and access to grow. Research in self-efficacy considers a students' consideration of previous performances, or *Mastery Experiences*, as the most significant predictor of positive self-efficacy.[17] In the case of Coral and Bianca, not seeing oneself as having done significant programming would mean fewer mastery experiences, and likely lower self-efficacy.

From an instructional perspective, this work shows the importance of considering not just the novice or expert status of potential pairs of students, but also their disciplinary identities, relationships to their partners, and sense of what counts as engaging in the targeted discipline or disciplinary practice (in this case, programming). Furthermore, this work illustrates the complexity of setting instructional goals related to increasing students' access to a discipline and of deciding whether those goals were met. If the goal of the Arduino projects was simply to increase students' access to programming, then the goal was not met for Bianca and Coral. But in much of engineering, the design process includes programming and other components. During Summer Girls, Coral's sense of herself as a builder grew, indicating that she gained access to that particular disciplinary practice and perhaps to engineering design more generally (though we lack sufficient data to know for sure). And even more broadly, Bianca's pride in her artwork may or may not be connected to her sense of engineering design or her view of "creativity." Our point here is that simply describing access to programming as something that was gained or lost obscures other ways that students may have gained access. We suggest that instructors and researchers of design environments recognize that isolating shifts in access to one aspect of

design can obscure other productive ways that students forge new trajectories in design. Conversely, probing for "access to engineering" writ large can obscure the nuanced ways in which students can gain access to some disciplinary practices (or subdisciplines) at the expense of others.

Acknowledgements
This material is based upon work supported by the National Science Foundation under Grant Numbers DRL-0733613. We thank student and instructors participants in the Summer Girls camp in 2013 and 2014 for cooperating with the data collection. We also thank members of the University of Maryland Engineering Education Group and Physics Education Group for their valuable feedback on this work.

References

1. Brown, A. L., & Campione, J. C. (1996). *Psychological theory and the design of innovative learning environments: On procedures, principles, and systems*. Lawrence Erlbaum Associates, Inc.
2. Cohen, E. G. (1994). Restructuring the classroom: Conditions for productive small groups. *Review of Educational Research*, *64*(1), 1-35.
3. Esmonde, I. (2009). Ideas and identities: Supporting equity in cooperative mathematics learning. *Review of Educational Research*, *79*(2), 1008-1043.
4. Tonso, K. L. (2006). Teams that work: Campus culture, engineer identity, and social interactions. *Journal of Engineering Education*, *95*(1), 25-37.
5. Hooper, S. (1992). Cooperative learning and computer-based instruction. Educational technology research and development, 40(3), 21-38.
6. Bandura, A. (1977). Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*, 84(2), 191.
7. Holland, D. C. (2001). *Identity and agency in cultural worlds.* Harvard University Press.
8. Seymour, E., & Hewitt, N. M. (1997). Talking about leaving: Why undergraduates leave the sciences (Vol. 12). Boulder, CO: Westview Press.
9. Stevens, R., O'Connor, K., Garrison, L., Jocuns, A., & Amos, D. M. (2008). Becoming an engineer: Toward a three dimensional view of engineering learning. *Journal of Engineering Education*, *97*(3), 355-368.
10. Boaler, J. (2000). Mathematics from another world: Traditional communities and the alienation of learners. *The Journal of Mathematical Behavior*, 18(4), 379-397.
11. Boaler, J., & Greeno, J. G. (2000). *Identity, agency, and knowing in mathematics worlds. Multiple perspectives on mathematics teaching and learning*, 171-200.
12. Margolis, J., & Fisher, A. (2003). Unlocking the clubhouse: Women in computing. MIT press.
13. Fields, D., & Enyedy, N. (2013). Picking up the mantle of "expert": Assigned roles, assertion of identity, and peer recognition within a programming class. *Mind, Culture, and Activity*, *20*(2), 113-131.
14. Jordan, B., & Henderson, A. (1995). Interaction analysis: Foundations and practice. *The Journal of the learning sciences*, *4*(1), 39-103.
15. Carlone, H. B. (2012). Methodological Considerations for Studying Identities in School Science. In Identity Construction and Science Education Research (pp. 9-25). SensePublishers.
16. Gee, J. P. (2000). Identity as an analytic lens for research in education. *Review of Research in Education*, 99–125.
17. Britner, S. L., & Pajares, F. (2006). Sources of science self- efficacy beliefs of middle school students. *Journal of Research in Science Teaching*, 43(5), 485-499.
18. H. B. Carlone, J. Haun-Frank, and A. Webb. (2011). Assessing Equity beyond knowledge- and skills-based outcomes: A comparative ethnography of two fourth-grade reform-based science classrooms, *Journal of Research in Science Teaching,* 48(5), 459-485.