Citation: Lui, D., Walker, J. T., Hanna, S., Kafai, Y. B., Fields, D., & Jayathirtha, G. (2020). Communicating computational concepts and practices within high school students' portfolios of making electronic textiles. *Interactive Learning Environments*, 28(3), 284-301.

Communicating Computational Concepts and Practices within High School Students' Portfolios of Making Electronic Textiles

Debora Lui, Justice T. Walker, Sheri Hanna*, Yasmin B. Kafai, Deborah Fields**, and Gayithri Jayathirtha

Graduate School of Education, University of Pennsylvania, Philadelphia, PA, U.S.A.

**String Theory Schools, School District of Philadelphia, Philadelphia, PA, U.S.A.*

***College of Education and Human Services, Utah State University, Logan, UT, U.S.A.*

3700 Walnut St. Philadelphia, PA 19104
deblui@upenn.edu; justicew@gse.upenn.edu; shanna@stringtheoryschools.org; kafai@upenn.edu; stareyes@gmail.com; gayithri@gse.upenn.edu

Debora Lui is a postdoctoral researcher at the Graduate School of Education in the University of Pennsylvania.

Justice T. Walker is a Graduate Student at the Graduate School of Education in the University of Pennsylvania.

Sheri Hanna is the STEM Coordinator and life sciences teacher at the String Theory School in Philadelphia, PA.

Yasmin B. Kafai is the Lori and Michael Milken President's Distinguished Professor of Education at the Graduate School of Education in the University of Pennsylvania.

Deborah A. Fields is an Associate Research Professor of Instructional Technology and Learning Sciences at Utah State University.

Gayithri Jayathirtha is a Graduate Student at the Graduate School of Education in the University of Pennsylvania.

For single agency grants: This work was supported by the National Science Foundation under Grant 1509245, 1512760, 1510725.

Communicating Computational Concepts and Practices within High School Students’ Portfolios of Making Electronic Textiles

Portfolios have recently gained traction within computer science education as a way to assess students’ computational thinking and practices. Whereas traditional assessments such as exams tend to capture learning within artificial settings at a single point in time, portfolios provide more authentic opportunities to document a trajectory of students’ learning and practices in everyday contexts. Furthermore, because communication itself has been defined as an important computational thinking practice, portfolios give students a place to practice this skill in the classroom. In this study, we report on the implementation of a digital portfolio with a class of 21 high school students used to capture the process of creating of an electronic textile mural project. While students’ understanding of computational concepts were only partially captured within the portfolios, their engagements with computational practices—such as debugging and iteration—were better highlighted. Much of this was due to the students’ existing communicative strategies themselves, both in terms of how precise they were in describing issues, as well as how they leveraged images and code to explain their process. Recommendations for designing more effective portfolio assessments are discussed, which include greater emphasis on creating shared classroom discourse, and leveraging students’ existing experiences with multimedia.

Keywords: computational thinking, portfolios, assessments, computational practices, communication

Introduction

Computational thinking (CT) has recently gained traction as an essential skill for students not only within computer science but also across the disciplines (Wing, 2006). Beyond the application of computational concepts, CT focuses on the particular perspectives and approaches to problems that can be derived from computational work, which can be productively applied to other fields (Grover & Pea, 2013; Grover, Cooper & Pea, 2014). While researchers and practitioners have pushed many efforts to implement different CT activities (see articles in this special issue), there have also been numerous efforts to develop tools and instruments to assess CT across a variety of platforms and activities. These include gaming (Koh, Basawapatna, Bennett, & Repenning, 2010; Werner, Denner, & Campe, 2014), 3D design (Repenning, Smith, Owen, & Repenning, 2012), modeling software (Basu, Kinnebrew, & Biswas, 2014), quizzes (Cooper, Perez & Rainey, 2010), and structured interviews (Brennan and Resnick, 2012). However, these efforts focus primarily on learning at the end of the process, rather than recording the experiential milestones achieved along the way. As computational instruction moves away from simply writing code toward activities that span across different academic disciplines, a more holistic assessment approach is needed—one that can capture students’ ongoing processes while engaging within these diverse contexts.

One promising solution for addressing this need is portfolio assessments, which have only recently received more attention within computer science education. Assessment portfolios can be characterized as artifacts that convey a student’s cumulative growth, activities and productions (Paulson, Paulson & Meyer, 1991). While popular in K-12 contexts, portfolios have primarily been used within arts and language education (Farr & Tone, 1994; Gitomer, Grosh, & Price, 1992; McKay, Keune, Peppler, Chang & Regalla, 2015), with sparing use within STEM fields. In computer science, portfolios have lately gained traction within the newly launched Advanced Placement Computer Science Principles (AP CSP) course (Arpaci-Dusseau et al., 2013; College Board, 2017), where they supplement the standard multiple-choice exam. This portfolio-driven approach aims to capture more robust insights into students' achievements, as well as situating assessment in more authentic, real-world contexts. Considering the well-documented issues of inequity within CS education, portfolios can additionally serve to support students who normally feel excluded from these spaces, allowing them to communicate and explore ideas in ways that might usually be suppressed within traditional computer science classrooms. These recent developments and potential benefits provide the impetus for examining how portfolios could be used to assess students' understanding of computation as well as capture the process through which students engage with this content. Additionally, use of portfolios as ongoing and formative assessments could provide new insights into the design of CT-infused STEM curriculum and activities moving into the future—ones that more appropriately address what students are actually learning and experiencing in these spaces.

In this paper, we report our initial efforts to analyze how students communicate computational concepts and practices through portfolios. As part of a separate study focused on how students collaborate when creating tangible computational projects (Lui, Litts, Widman, Walker, & Kafai, 2016; Litts, Lui, Widman, Walker & Kafai, 2017b; Litts, Widman, Lui, Walker & Kafai, in press), we conducted an electronic textiles workshop with 21 high school students to create an interactive, fabric-based school mural. Moving beyond our initial efforts, the class teacher (Author 3) implemented a digital portfolio assignment where she asked students to document their process for the purposes of classroom assessment. While students were given an outline of content to include in the portfolio, they were free to organize this information and use whatever supporting materials they wished. Working with the teacher, we then decided to analyze these portfolios, with a focus on what they could tell us about the students' computational thinking outcomes. We asked: (1) What evidence could we find of students' engagement with computational *concepts* and *practices* in these portfolios? (2) How did students communicate this information, in terms of language and media use? (3) What supports or structures of the portfolio yielded the most useful assessments? In our discussion, we develop a series of recommendations for other researchers and educators looking to use portfolios as a way of assessing computational thinking, and in shaping CT-activities and curriculum that best support students' actual experiences and processes.

Our emphasis on students' communicative strategies are key in developing these recommendations, considering that *communication of computational ideas and processes* is itself considered a computational thinking practice, alongside more typical activities such as writing code or debugging programs (College Board, 2017). While existing research speaks of portfolios' potential in capturing students' processes and thinking (e.g., Býrgýn & Adnan, 2007, Paulson et al., 1991), the actual success of these assessments is ultimately limited by how effectively students are able to share what they actually did. This is not just a matter of vocabulary (i.e., knowing the right words to describe specific concepts), but also how well students can articulate and accurately capture their process through text and other available media. Thus, our study not only looks at *what* students said, but also *how* they choose to communicate this information. Only by looking more closely at students' communication strategies will we be able to establish more effective guidelines for designing portfolio assessments that truly capture students' understanding of computational concepts and practices.

Background

While portfolios were initially derived from art and writing contexts (e.g., Farr & Tone, 1994; Gitomer et al., 1992), they have gained popularity within STEM fields—particularly Computer Science and Engineering—because of the potential benefits they offer over traditional test or task-based assessments (Býrgýn & Adnan, 2007). One argued advantage of portfolios is that they are better able to capture a more holistic view of student understanding and learning because they focus on *process* alongside product (Paulson et al., 1991). While traditional assessments tend to focus on single time points and are thus considered more artificial, portfolios are usually situated within everyday practice since students must record what they are already doing as part of their ongoing work (Býrgýn & Adnan, 2007). Portfolio documentation therefore replicates the already existing practice of keeping design notebooks in engineering education (Eris, 2006). Further, as outlined in numerous studies, the advent of new digital tools makes process-driven documentation easier for CS students capturing their code revisions along the way (e.g., Estell, 2001; Higgs & Sabin, 2005). Learning and assessment can therefore be more seamlessly integrated within classroom practice through portfolios (Gilman, Andrew & Raffert, 1995). For students, the creation of a portfolio can provide agency in shaping one's learning over time, whether through continuous self-feedback and monitoring of progress (Adams, 1998, De fina, 1992), or purposeful opportunities for goal setting (Owings & Follo, 1992). For teachers, portfolios can be leveraged as a type of formative assessment to help improve individual learning trajectories (Mullin, 1998), whether within a single activity or in improving this activity over future iterations.

Another important benefit of portfolios for CT is that they provide opportunities for students to practice their computational communication. While communication is considered essential within the humanities and social sciences, researchers and educators have also argued about its importance within STEM subjects. For the Advanced Placement Computer Science Principles (AP CSP) course, this need has been highlighted by placing communication—or students' capacity to describe and explain computational artifacts and related processes and behaviors—alongside other key computational practices such as abstraction and problem analysis (College Board, 2017, p. 9-10). Research has also demonstrated how student articulation of concepts can strengthen scientific understanding in and of itself through solidifying abstract ideas (Phelps, LaPorte and Mahood, 1997). For CT, this can even encompass the acquisition of a shared "vocabulary of computing" (Grover et al., 2014)—something that can both foster "deeper computational learning" and nurture students' abilities to think about "computational ideas more effectively" (p. 58). Because of this, there has been some effort to start teaching communication skills within CS courses—although at the university level rather than K-12 contexts (e.g., Falkner & Falkner, 2012; French, 2012). Portfolios, then, might help fill this gap, creating channels for novice students to practice and improve upon their technical communication skills within a more personalized context. Additionally, giving students opportunities to describe their personal process in their own words can provide deeper insights into their computational understanding and practices (Brennan and Resnick, 2012).

In looking at these claimed benefits of portfolio assessments, we therefore ask if these advantages still hold within high school contexts. Notably, most of the research in computer science and engineering portfolios has focused on higher education (e.g., Eris, 2006; Estell, 2001; Michael, 2000), where students are already being enculturated into the field. In dealing with high school students' computational experiences, it is essential to consider how well portfolios can capture both their knowledge and process, and to look at whether or not the communication strategies they choose (in terms of language and media forms used) can either help or hinder our ability to assess this knowledge.

Methods

Participants and Workshop

We conducted this study with 21 high school students (4 boys, 17 girls, 16-17 years old) at a charter school in a Northeastern city. Student racial demographics mirrored those of the school with 44% African American, 35% Caucasian, 13% Hispanic, 3% Asian, and 3% Multiracial students. Participants were members of a multi-year STEM elective class, which was taught by a teacher with a background in biology. While the course mostly focused on life science topics (e.g., human anatomy, ecology), the teacher occasionally engaged students within engineering and computation projects. Sixteen students had completed an introductory e-textiles project in the previous academic year (which was part of another study) (Litts, Kafai & Dieckmeyer, 2015), while five were new to the class and engaging with e-textiles for the first time. We started the workshop with 24 students working in 12 pairs, but one student transferred schools and left her partner working independently. Additionally, one pair ended up not submitting a final portfolio due to personal circumstances. As a result, we analyzed a total of 11 portfolios, produced by 21 students (10 pairs and 1 individual) for this study.

The workshop was jointly designed and led by the class teacher and our team of researchers. Over 15 90-minute class periods, student pairs created a collaborative interactive sign spelling out the school's name. Each pair was assigned a letter, which was previously designed by art students in the school and printed on canvas. Pairs were required to make each letter 'interactive' using e-textiles components (LilyPad Arduino microcontrollers, LEDs, sensors, switches) (Buechley, Eisenberg, Catchett, and Crockett, 2008), which were programmed such that different light patterns could be triggered by a sensor or switch (Figure 1). The workshop was originally designed to study students' collaborations and interactions when working on tangible computational projects (Litts et al., 2017b; Litts et al., in press; Lui et al., 2016). However, once the teacher decided to implement the portfolio as a way to evaluate students, we incorporated an analysis of these into our larger study.

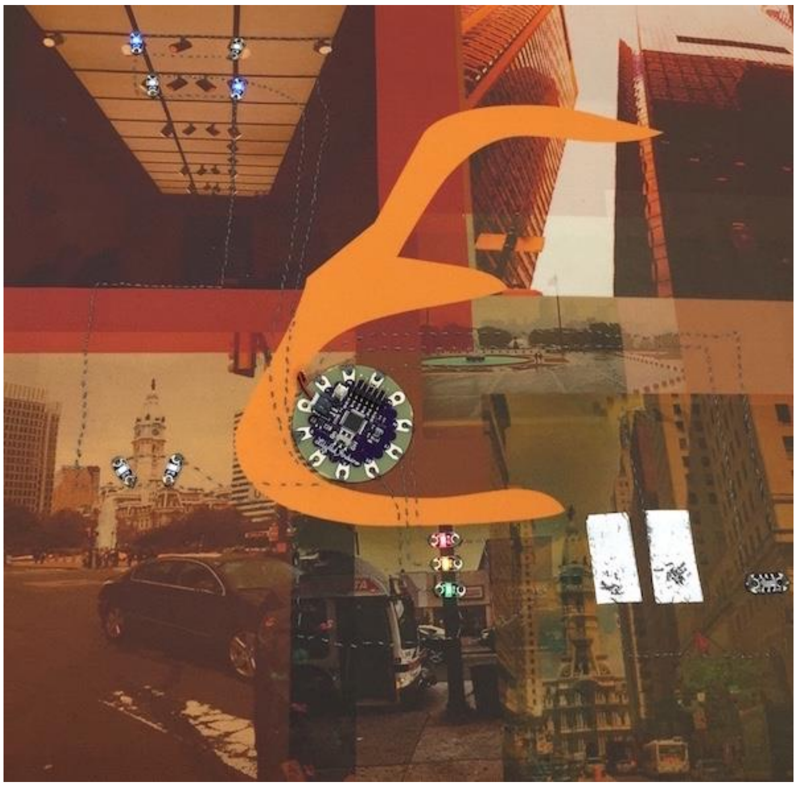
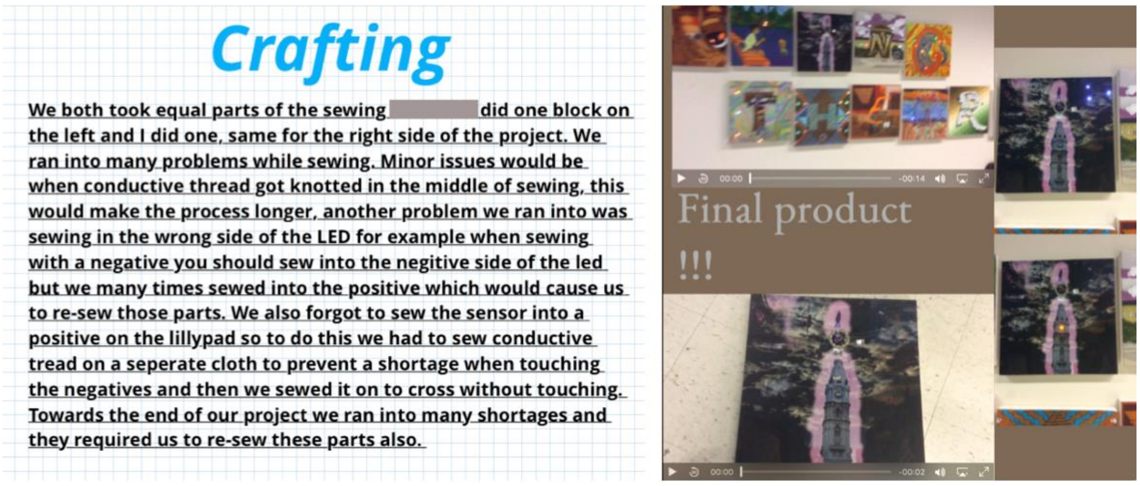


Figure 1. Example of a completed sign from the class

Portfolio Assignment and Data Collection

As students were in the midst of planning their projects (Day 3), the teacher introduced her portfolio assignment. Each pair was asked to document their process in an e-Book format, using Apple's iBooks authoring application. They were required to address all of the following topics: (1) uses of e-textiles in society (2) the overall class assignment, (3) the design, (4) crafting, (5) circuitry, and (6) coding of their project, (7) a video demonstration and explanation of the final product itself, and both a (8) pair and (9) individual reflection. The teacher also suggested including in-progress images, discussions of challenges faced, and 'tips' for others e-textiles makers. Students had the freedom to address and organize the required topics however they wished, whether together or in separate sections (Figure 2). The teacher used the portfolios as a summative assessment, along with evaluating their completed final projects. Following the end of the workshop, we collected all the available e-Book portfolio files (11) for further analysis.


Crafting

We both took equal parts of the sewing [redacted] did one block on the left and I did one, same for the right side of the project. We ran into many problems while sewing. Minor issues would be when conductive thread got knotted in the middle of sewing, this would make the process longer, another problem we ran into was sewing in the wrong side of the LED for example when sewing with a negative you should sew into the negitive side of the led but we many times sewed into the positive which would cause us to re-sew those parts. We also forgot to sew the sensor into a positive on the lillypad so to do this we had to sew conductive tread on a seperate cloth to prevent a shortage when touching the negatives and then we sewed it on to cross without touching. Towards the end of our project we ran into many shortages and they required us to re-sew these parts also.

Figure 2. Sample portfolio pages from two different portfolios illustrating the different compositions and combinations of text and media.

Data Analysis

After consultations with the classroom teacher and review of the existing literature on computational thinking and communication, our research team decided upon two rounds of portfolio analysis focusing content (*what* students wrote about) and communication (*how* they wrote or reported about these things).

Portfolio Content

While earlier efforts to define computational thinking tended to emphasize understanding of *concepts* within computer science (e.g., Wing, 2006), more recent research has focused on the importance of students' activities and *practices* in the field (e.g., Brennan & Resnick, 2012; Bienkowski, Snow, Rutstein, & Grover, 2015, Weintrop et al., 2016). For this analysis, we therefore looked at both computational concepts and practices.

Regarding students' understanding of *computational concepts,* we derived relevant categories from existing research on e-textiles learning, which highlights both *coding*—the programming of students' projects, and *circuitry*—the creation of electrical connections between components, as the two main areas of computation involved in e-textiles (Kafai, Fields, & Searle, 2012). Within these categories, we looked for evidence of student understanding in specific *underlying coding and circuitry concepts*, which respectively include: events, sequences, loops, conditionals, data, and operators (Brennan & Resnick, 2012), and polarity, connection types, and current flow (Litts, Kafai, Lui, Walker, & Widman, 2017). Further explanations of these concepts with sample quotes are included in the findings. For each portfolio, we marked whether evidence of student understanding of these concepts was present or not.

We also looked at evidence of students' engagement with *computational practices*, or the specific activities that learners engage with while constructing computational projects, thus "moving beyond *what* you are learning to *how* you are learning" (Brennan & Resnick, 2012, p. 6-7). Looking at existing literature (Brennan & Resnick, 2012; Fields, Lui & Kafai, 2017), we identified two major practices within the existing portfolios: *debugging and troubleshooting*, or "develop[ing] strategies for dealing with—and anticipating—problems" (Brennnan & Resnick, 2012, p. 7), and *iterating and revising*, or engaging in an incremental, continual "cycle of prototyping, testing, and revision" (Fields et al., 2017). For each portfolio, we marked whether this evidence was present or not. Again, further explanations with sample quotes are included in the findings.

Portfolio Communication

Our second round of coding focused on students' communicative methods. Following from the constructionist perspective of the portfolio as a "public entity" (Papert & Harel, 1991) that students create (alongside the physical project itself), we examined how students actually expressed their ideas for an audience. Here, we draw from existing research on communication within CS education that considers multiple levels of fluency. This ranges from initial facility with some "vocabulary of computing" (Grover et al., 2014) to the integrated use of these terms to explain, describe and clarify one's knowledge and designs (Falkner & Falkner, 2012). Additionally, we drew from research that highlights the centrality of using and creating representations (visual or otherwise) when becoming fluent in a science field (Hill & Sharma, 2015). This dual focus on text and media is further supported by the definition of communication in the AP CSP Guide that describes students' abilities to report on the outcomes and processes of creating computational artifacts using "accurate and precise language, notations, or visualizations" (College Board, 2017, p. 10).

For each portfolio where evidence of computational concepts or practices was present, we considered three factors. First, we categorized the different presentational *contexts* where evidence was located, for instance, whether their description of final project behaviors, the narrative of their experiences, or their "tips" for others. Second, we evaluated students' *language* in this evidence, specifically considering how precise or detailed students were. Third, we catalogued students' *media use* in relation to this evidence, looking not only at what images, video, or code was included, but also what presentational techniques students employed, whether image compilations, code excerpts, or color-coded annotations.

Across these three categories, we compiled trends regarding how students communicated their computational concepts or practices as a way of understanding the affordances of portfolios in capturing this information. For portfolios where evidence was *not* present, we considered factors that potentially limited what students shared or reported. In the discussion, we develop a series of recommendations based on these findings for designing future portfolio assignments to effectively assess computational concepts and practices.

Findings

Below, we report on trends of students' reporting of computational concepts and practices. First, we describe the structural differences between student portfolios across the class. Though students were given the same basic guidelines, pairs' portfolios greatly differed in terms of size and composition. Portfolios ranged from 11 to 21 pages (average: 16.1, median: 17, mode: 19), and each differed in combinations of text, images and video. While some pairs had numerous pages that only contained images or video, others had different combinations of text and image on every page, and still others had some pages with only text (Figure 2). Portfolios therefore ranged in number of words (range: 922 to 2256, average: 1360.1), images (range: 4 to 2, average: 12.7), and videos (range: 1 to 5, average: 2.4). For the most part, students included media on almost every page of their portfolios with an average ratio of images and videos to pages of 0.92 (range: 0.45 to 1.5). Trends relating to students' reports on computational concepts and practices are detailed below.

Computational Concepts

As expected, the design of the portfolio assignment significantly impacted what computational concepts students' portfolios evidenced. Pairs were explicitly asked to address the circuitry and coding of their projects. However, what they shared about these concepts and how they communicated these ideas differed greatly. While descriptions of their final projects tended to more precise (since they could rely on concrete details), discussions of students' process ranged from vague to specific—something that depended on both the scope and types of issues they encountered during production. Notably, students did not explicitly discuss underlying concepts of particular domains (e.g., sequences as a coding concept; polarity as a circuitry concept) *unless* they had explicit challenges relating to these areas. Additionally, students' use of media evidence—code excerpts, circuit diagrams—could be used as evidence of their understanding of numerous coding and circuitry concepts. However, this was heavily mediated by their legibility. Many pairs ended up including this media without much notation or explanation, thereby limiting their usefulness in assessing student understanding. However, some made efforts to either annotate these or create purposeful collections, thereby increasing their communicative power. The specifics of these conclusions are further described below.

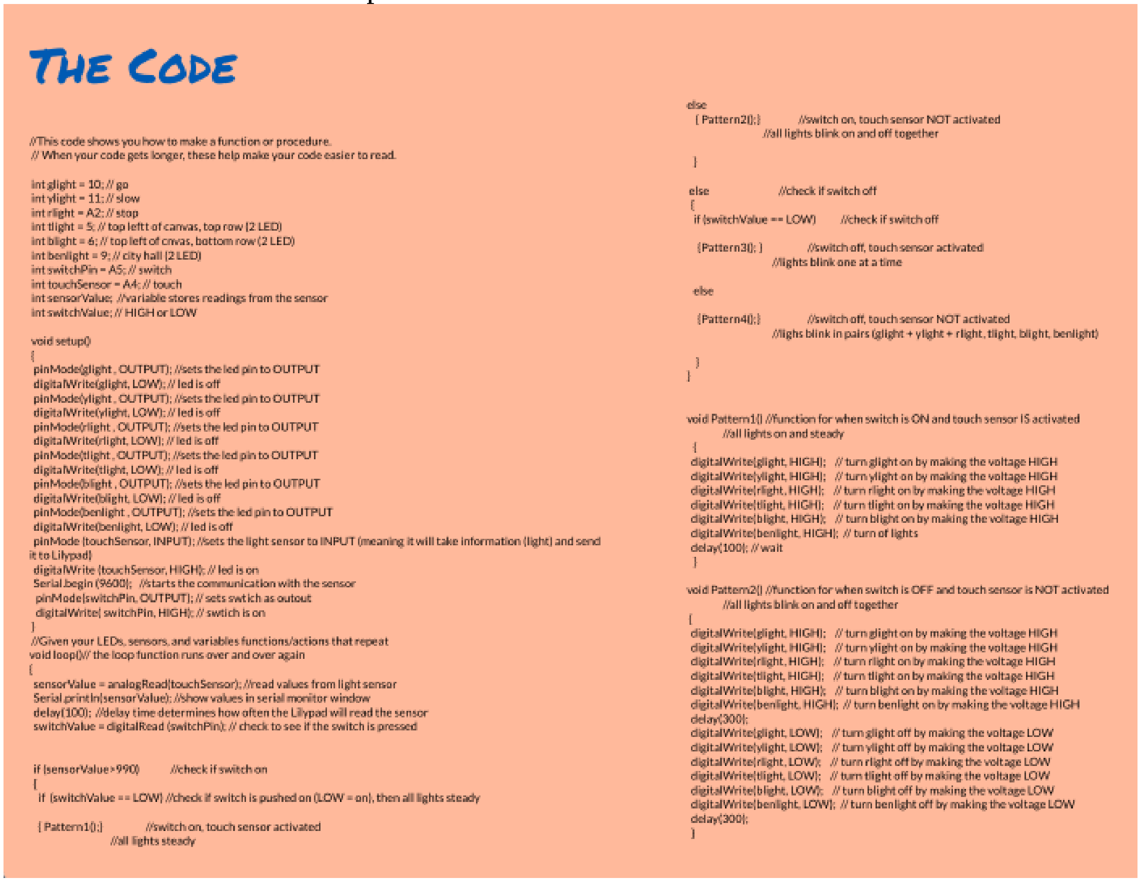
Coding

Evidence of students' understanding of code could be found in multiple contexts including: descriptions of their final project behaviors (8 of 11), their code excerpts (11 of 11), and descriptions of specific coding challenges (4 of 11). Project descriptions usually included a list of project behaviors (e.g., LEDs blinking), as well as their triggering actions (e.g., using a switch or sensor). Estelle and Adam, for instance, provided the following description of their programmed light patterns:

> So for our first pattern it was a cycle of all of the lights going clockwise. The second pattern was just [the LED on] city hall's clock lit up…The third pattern was city hall's clock light and the street lights [LEDs] going back and forth, at a slow paste [sic]. Lastly our fourth pattern consist [sic] of only the streetlights on. Pattern one and three were both with the switch on, and pattern 1 and 4 both were with the switch off. We used touch sensors to show the patterns. (p. 13)

Because this quote includes specific details and domain-specific language (e.g, "first pattern", "switch on"), this and other project descriptions generally demonstrated pairs' understanding of *events*—"one thing causing another thing to happen," *conditionals*—if/then branched logic "which supports the expression of multiple outcomes," (Brennan and Resnick, 2012).

Every portfolio (11) also included text or images of the code itself. Most pairs (8 of 11) included the entire program, including the 'starter' functions that everyone used to set up the sensor/switch behaviors (Figure 3). However, a few pairs (3 of 11) were more targeted, excerpting only the customized functions they wrote (e.g., the different light patterns). Beyond events and conditionals, student understanding of other coding concepts could be inferred by looking at this code, including: *sequencing*—the idea that an "activity or task is expressed as a series of individual steps or instructions," *operators*—"support for mathematical, logical, and string expressions," *data*—"storing, receiving, or updating values" and *loops*—the "mechanism for running the same sequence multiple times" (Brennan and Resnick, 2012). However, this understanding could really only be confirmed if students explicitly addressed these particular concepts through their prose, as seen within discussion of their experiences.


THE CODE

```
//This code shows you how to make a function or procedure.
// When your code gets longer, these help make your code easier to read.

int glight = 10; // go
int ylight = 11; // slow
int rlight = A2; // stop
int tlight = 5; // top leftt of canvas, top row (2 LED)
int blight = 6; // top left of cnvas, bottom row (2 LED)
int benlight = 9; // city hall (2 LED)
int switchPin = A5; // switch
int touchSensor = A4; // touch
int sensorValue;  //variable stores readings from the sensor
int switchValue; // HIGH or LOW

void setup()
{
 pinMode(glight , OUTPUT); //sets the led pin to OUTPUT
 digitalWrite(glight, LOW); // led is off
 pinMode(ylight , OUTPUT); //sets the led pin to OUTPUT
 digitalWrite(ylight, LOW); // led is off
 pinMode(rlight , OUTPUT); //sets the led pin to OUTPUT
 digitalWrite(rlight, LOW); // led is off
 pinMode(tlight , OUTPUT); //sets the led pin to OUTPUT
 digitalWrite(tlight, LOW); // led is off
 pinMode(blight , OUTPUT); //sets the led pin to OUTPUT
 digitalWrite(blight, LOW); // led is off
 pinMode(benlight , OUTPUT); //sets the led pin to OUTPUT
 digitalWrite(benlight, LOW); // led is off
 pinMode (touchSensor, INPUT); //sets the light sensor to INPUT (meaning it will take information (light) and send
it to Lilypad)
 digitalWrite (touchSensor, HIGH); // led is on
 Serial.begin (9600);   //starts the communication with the sensor
  pinMode(switchPin, OUTPUT); // sets swtich as outout
  digitalWrite( switchPin, HIGH); // swtich is on
 }
 //Given your LEDs, sensors, and variables functions/actions that repeat
void loop()// the loop function runs over and over again
{
 sensorValue = analogRead(touchSensor); //read values from light sensor
 Serial.println(sensorValue); //show values in serial monitor window
 delay(100);  //delay time determines how often the Lilypad will read the sensor
 switchValue = digitalRead (switchPin); // check to see if the switch is pressed


 if (sensorValue>990)         //check if switch on
 {
  if  (switchValue == LOW) //check if switch is pushed on (LOW = on), then all lights steady

  { Pattern1();}             //switch on, touch sensor activated
                   //all lights steady

else
  { Pattern2();}             //switch on, touch sensor NOT activated
                       //all lights blink on and off together

  }

 else                        //check if switch off
 {
  if (switchValue == LOW)        //check if switch off

   {Pattern3(); }              //switch off, touch sensor activated
                          //lights blink one at a time

  else

   {Pattern4();}               //switch off, touch sensor NOT activated
                          //lighs blink in pairs (glight + ylight + rlight, tlight, blight, benlight)

  }
}


void Pattern1() //function for when switch is ON and touch sensor IS activated
          //all lights on and steady
  {
 digitalWrite(glight, HIGH);    // turn glight on by making the voltage HIGH
 digitalWrite(ylight, HIGH);    // turn ylight on by making the voltage HIGH
 digitalWrite(rlight, HIGH);    // turn rlight on by making the voltage HIGH
 digitalWrite(tlight, HIGH);    // turn tlight on by making the voltage HIGH
 digitalWrite(blight, HIGH);    // turn blight on by making the voltage HIGH
 digitalWrite(benlight, HIGH);  // turn of lights
 delay(100); // wait
  }

void Pattern2() //function for when switch is OFF and touch sensor is NOT activated
          //all lights blink on and off together
 {
 digitalWrite(glight, HIGH);    // turn glight on by making the voltage HIGH
 digitalWrite(ylight, HIGH);    // turn ylight on by making the voltage HIGH
 digitalWrite(rlight, HIGH);    // turn rlight on by making the voltage HIGH
 digitalWrite(tlight, HIGH);    // turn tlight on by making the voltage HIGH
 digitalWrite(blight, HIGH);    // turn blight on by making the voltage HIGH
 digitalWrite(benlight, HIGH);  // turn benlight on by making the voltage HIGH
 delay(300);
 digitalWrite(glight, LOW);    // turn glight off by making the voltage LOW
 digitalWrite(ylight, LOW);    // turn ylight off by making the voltage LOW
 digitalWrite(rlight, LOW);    // turn rlight off by making the voltage LOW
 digitalWrite(tlight, LOW);    // turn tlight off by making the voltage LOW
 digitalWrite(blight, LOW);    // turn blight off by making the voltage LOW
 digitalWrite(benlight, LOW);  // turn benlight off by making the voltage LOW
 delay(300);
 }
```

Figure 3. A typical sample of how pairs presented their code, which includes text of their entire program.

Almost every pair (10 of 11) included general descriptions of their overall experience of programming. This was mostly described in vague terms, as illustrated by Kiara and Cassidy, who stated: "There was also a problem with the coding so we had to go back and read the code, and check what was wrong" (p. 5), and Jasmine and Melanie: "Programming was quite difficult because the led lights…were not responding to the functions we had in the coding…We double checked out programming and resolved the many coding conflicts" (p. 10). These ambiguous descriptions did not provide much evidence of students' understanding of underlying coding concepts. There were a few exceptions to this, seen when pairs (4 of 11) decided to focus on one specific issue as Roberto and Malik illustrated:

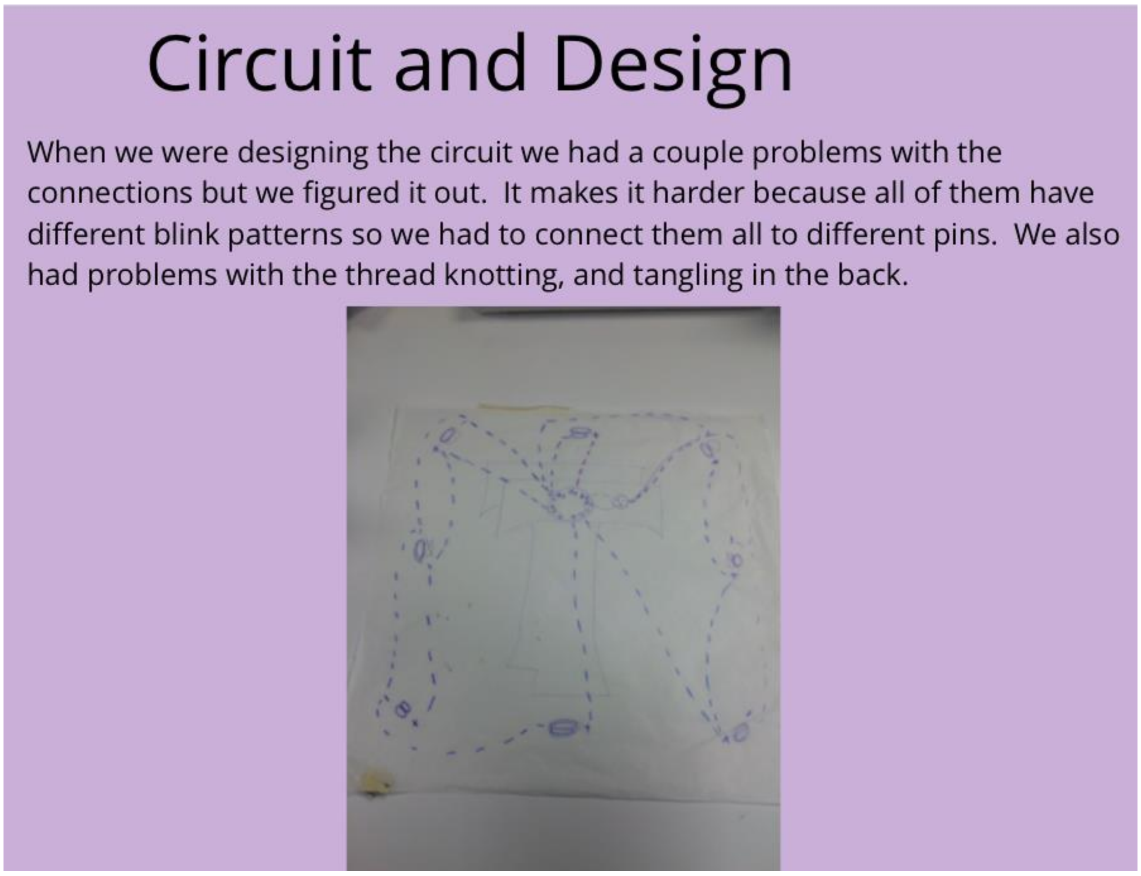
> The most difficult part of the coding was getting the light sensor to work. We at least looked at the code for 3 days straight and couldn't find out what was wrong. [An instructor] helped us try to fix it and noticed a simple mistake… If you look on the code you will see something names "sensorneg". We set the pin to be negative, but we never set the pin it that it was connected to an output. If we didn't make it an output then the sensor was only getting the possible electricity. (p. 10)

Here, Roberto and Malik demonstrate understanding of several coding concepts including both *sequencing* and *data* through this in-depth reporting of one issue. While students' attempts cover their overall coding experiences tended to produce more ambiguous descriptions, emphasis on discrete issues tended to yield more detailed and precise reports. These reports, in turn, helped confirm their understanding of particular concepts. This is not to say that the other pairs lacked understanding of these concepts. It only indicates they either never faced particular challenges in these areas, or did not explicitly name these within their portfolios, something further exacerbated by their lack of domain-specific terms to describe their coding errors. This need to further support student discussion of challenges—potentially through carefully designed scaffolds and use of shared language—is further addressed in our discussion.

Circuitry

Evidence of pairs' understanding of circuitry could be found in multiple contexts including: descriptions of their final project electrical connections (3 of 11), use of circuit diagrams (10 of 11), and descriptions of their process of creating their circuit diagrams (10 of 11). As with coding, students' descriptions of their final electrical connections included use of domain-specific language and precise details, as seen with Naomi and Yoana: "Each light [in the picture above] is attached to it's [sic] own pin and [can] blink… at its own time. This group of lights mimics a stop light" (p. 9). Here, we can see evidence of their understanding of circuitry *connections*—or the way that the components were connected to each other to allow for particular coded behaviors (Litts et al., 2017a). Notably, only three pairs included these descriptions into their portfolio—something that likely occurred since students thought this information was best conveyed through their circuit diagrams.

Almost every pair (10 of 11) also included circuitry diagrams within their portfolios, which visually demonstrated how the electrical components were connected together. Students' strategies for presenting these diagrams differed. Over half the pairs (6 of 11) just included unmarked photographs of the paper ones they had drawn in class, which were often difficult to see and decipher (Figure 4). However, four groups attempted to make these more legible either by creating new digital versions either with labels, close-ups, or strategic color-coding (Figure 5).



Circuit and Design

When we were designing the circuit we had a couple problems with the connections but we figured it out. It makes it harder because all of them have different blink patterns so we had to connect them all to different pins. We also had problems with the thread knotting, and tangling in the back.

Figure 4. A typical sample of how students presented their circuit diagram as a photograph of the paper drawings, which was often difficult to decipher.

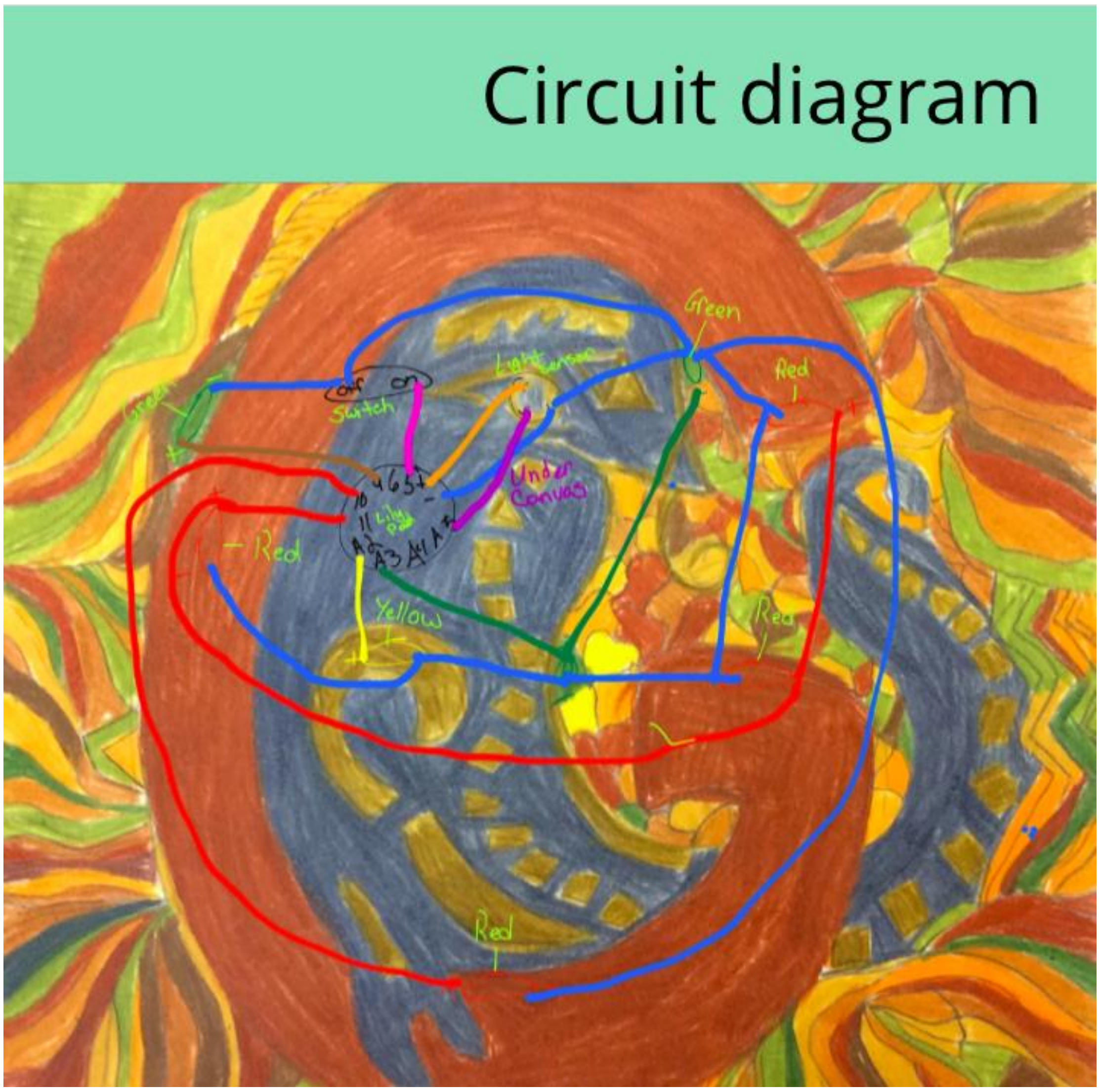


Figure 5. A rare example of a color-coded circuit diagram with labels, which makes it easier for readers to read.

For these few annotated circuit diagrams, it was possible to infer student understanding of circuitry concepts, not only connections, but also *current flow—or* the pathway of electrons through electrical connections and *polarity*—the existence of positive and negative poles of components that allow for current flow (Litts et al., 2017a). Inferring this knowledge was more difficult for the un-annotated diagrams, however, since they were difficult to interpret due to illegibility. Thus, use of diagrams to illustrate circuitry knowledge could only go so far without conscious efforts applied toward clarity and annotation, something that has further implications in the design of future portfolio assignments.

Compared with students' narratives of their *coding* experience, which was generally more vague, pairs' discussion of their *circuitry* experience (10 of 11) was usually more detailed and precise. For example, Mia and Matthew described their experience this way, along with two images of their diagram:

> The circuit diagram changed a little because we added two other lights on our canvas. We added a green light and a red. In the begin [sic] the circuit diagram

> changed a lot. As we were making it in the begin some sewing would cross so we would have to start all over again and we would have to rearrange everything so that there was no crossing. (p. 6)

Here, we can see how Mia and Matthew's discussion of avoiding "crossing" provides evidence of their understanding of short circuits and therefore *current flow* and *polarity*. Other reported circuitry issues included keeping track of positive and negative lines, and rearranging LED positions for more individually programmable pin connections, which illustrate understanding of *polarity* and *connection*s, respectively. Compare these examples, for instance, with the earlier described coding experience descriptions, where students only described having issues with code without more precisely describing why this was so. Thus, while student understanding of coding was harder to confirm through prose, here, circuitry experience discussions were more detailed. One potential reason for this might be the greater concreteness of e-textiles circuitry over coding; while circuitry has a tangible component made visible through physical sewn connections between components, coding is generally more abstract, since it is contained within functions on a screen. Ways of addressing this distinction through language use, as well as media use, are further considered in the discussion below.

Computational Practices

Unlike computational concepts, students' discussion of computational practices was not as strongly dictated by the given portfolio assignment. Evidence of students' computational practices—whether *debugging and troubleshooting*, or *iterating and revising*—was generally distributed under the teachers' suggested formats of 1) description of challenges encountered, 2) 'tips' for others, and 3) reports of pairs' design, circuitry, coding, or crafting experience. While pairs' discussion of challenges had the most potential to provide specific details about their engagement with computational practices, these sometimes yielded vague descriptions since they only chose to list their problems rather than describe their solutions. However, students' tips arguably provided greater insight into their engagement to computational practices since they were simultaneously general (i.e., applicable across different situations) and detailed (e.g., recommending specific actions). More specifics on students' reporting of their debugging/troubleshooting and revision/iteration practices are outlined below.

Debugging and Troubleshooting

As expected, students tended to discuss their debugging and troubleshooting while addressing the prompts to write about their project challenges (11 of 11), and their tips for other e-textile makers (6 of 11). Issues that students described primarily fell into two categories: *dealing with mistakes* (e.g., missing code, faulty sewing) or *being unfamiliar with particular tools and materials* (e.g. reading Arduino error messages, working with conductive thread). Students generally had different approaches toward reporting their mistakes. While sometimes students only outlined their problems, other times they detailed both their problems *and* solutions, which was more effective at illustrating their troubleshooting skills. Oftentimes these approaches were simultaneously present within the same portfolio, that is, students could be both vague *and* precise when describing the same experiences. This can be seen in Erin and Audrey's multiple descriptions of their coding challenges:

> There were difficulties when sewing because it was a lot of lights to put on. And the code was hard because it was a lot to write and it was confusing. (p. 6)

> A challenge that we encountered was with the touch sensors. When we were all done with the sewing and coding the touch sensor coding would work without having to touch the sensors. Turns out there was a problem with the sensor value. The serial monitor was reading at over a 1000 and the sensor value was reading at less than 100. So I just rewrote that part to read greater then 950 and the coding for the lights without touching the sensors worked regularly and the touch sensors would work when pressed. (p. 9)

As mentioned earlier, if students chose to describe their *entire* experience of coding or crafting, they tended to be more vague, describing only problems. This tendency could be overcome, however, when limiting their descriptions to just one challenge.

Students also described being unfamiliar with materials and tools in their portfolios. Though not typically considered part of debugging or troubleshooting, the process of becoming more comfortable or knowledgeable about domains did yield interesting insights about their computational practices. Mostly, this was revealed through their writing of 'tips' for future e-textiles creators (6 of 11). Pairs articulated tactics such as testing things out along the way (e.g., "Check to see if your code works [sic] after every line of code, so you don't have to go back and change the whole thing later") (Cassidy and Kiara, p. 6), or methods of avoiding issues in the future ("while sewing always check if the negative and positive are on the right sides") (Noel and Natasha, p. 12). Thus, tips were sometime even more useful than descriptions of challenges when inferring students' overall problem solving strategies and approaches precisely *because* they were general and applicable across a domain. Again, these discussions could have been improved, however, through use of more precise language. Often, assessment of student knowledge depended upon translating lay phrases into more domain-specific terms (e.g., above, use of the word "works" instead of "compiles," or "are on right sides" instead of "correctly aligned polarity").

Regarding students' use of media, it is striking that only two portfolios actually included any additional media (image, video, or code excerpts) to support their descriptions of debugging and troubleshooting, even though the teacher actively encouraged this. Even when describing some kind of physical mistake or coding error, students did not generally include relevant images such as a screenshot of a coding error, or a picture of an incorrectly sewn LED. This indicates the need to actually scaffold students during the process of creation, whether through regular intervals of taking photographs and screenshots, or working to develop the class' familiarity with domain-specific terms or language.

Iterating and Revising

While not required, most portfolios (8 of 11) addressed the practice of iterating and revising within their descriptions of their experiences. While reports of debugging and troubleshooting were spread across coding, circuitry, crafting and design, reports of revision and iteration were primarily contained within design and circuitry. These discussions primarily concerned students' decisions about where to place their lights, which was often based on both aesthetic preference and circuitry concerns. This can be seen in Joy and Caroline's description of this process:

> We wanted to position the LEDs in a way that would really bring out the letter from the background. Initially, we wanted to have 16 LEDs--one LED going on each bucket of the Ferris wheel--but in order to have the four light patterns each LED had to be sewed to separate analog pins and we only had 7 analog pins available on the lilypad. So we tried our best to scatter 7 LEDs around the Ferris wheel evenly. (p. 5)

Because these discussions involved concrete details, they tended to be both highly specific and precise. Sometimes these textual descriptions were accompanied by multiple versions of their circuit diagrams (4 of 11) (Figure 6). As mentioned earlier though, the effectiveness of these compilations was occasionally limited by the illegibility of the images themselves due to size or color.

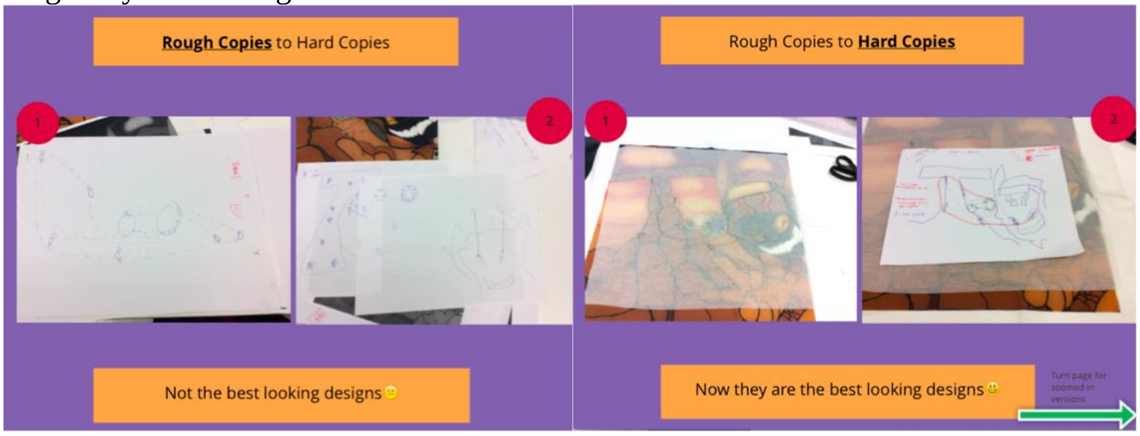

Rough Copies to Hard Copies
1
2
Not the best looking designs😊
Rough Copies to Hard Copies
1
2
Now they are the best looking designs😊
Turn page for zoomed in versions

Figure 6. Different versions of a circuit diagram presented within a single portfolio, illustrating the computational practice of iterating and revising.

Detailed descriptions of specific changes were not always required to prove students' engagement with revision and iteration. More general statements about a students' process could also shed light on students' overall strategies, as seen in Sara's portfolio:

> [I was]…trying to come up with a circuit design that would actually work, figuring out if either some positives and negatives would be [too] long/continuous when it came to [sewing with] conductive thread or would it be short, and [also] preventing positives and negatives from crossing or being too close to each other. (p. 12)

So even while Sara did not report on the specifics of her diagram (i.e., what was connected to what), this description still provided evidence about her general approach to circuit design, including what she tried to accomplish and what she tried to avoid. This included figuring out the most efficient sewing pathways (not making things too "long or short") and preventing short circuits (avoiding "crossing or being too close"). Thus, specific details that might be essential for assessing student understanding of computational *concepts* might not be as necessary when considering student engagement with computational *practices*, which focuses more on approaches and procedures.

Discussion

Our study examined the feasibility of implementing a portfolio to document students' processes of generating a computational artifact, and to assess their understanding of underlying computing ideas. While students' understanding of computational concepts could be loosely inferred through the portfolios (through students' code excerpts and circuit diagrams), this information was only confirmed in those portfolios that included explicit discussions of these elements. However, the portfolios were more successful in capturing student computational practices, something that likely occurred not only because students were required to keep track of their ongoing experiences, but also because students learned how to articulate and share these with an audience. In this way, the portfolios were successful at providing students opportunities to rehearse and potentially strengthen their skills of communication, itself a key element of computational thinking. Despite this, students' actual effectiveness expressing their ideas—especially for the purposes of evaluation and assessment—was variable. Below, we outline some reasons for these variances and discuss how one might address these in future research, both in terms of how portfolios can be used as assessments and for the purposes of learning and documentation.

Clarifying the Purpose Behind the Portfolio

One essential issue to consider when considering portfolio assessment is what actually drives students' descriptions of their process within these portfolios. While the portfolio assignment seemingly provided a solid structure for students to report on their computational projects, our findings illustrate how students' communication occasionally fell short of expectation since their language was often vague and lacking in relevant detail. As illustrated within existing research on portfolios in various disciplines, this often occurs when there is a lack of clarity from both instructors and students about the eventual purpose of the portfolio (Calfee & Perfumo, 1996), the appropriate materials that students should include to support this goal (Herman, Gearhart, & Aschbacher, 1996), as well as specific standards for evaluating this content (Owings & Follo, 1992).

From this perspective, one solution to overcome the vagueness of student descriptions would be to be explicit about the actual evaluative purpose of these computational portfolios, and to work collaboratively with students on creating shared or "public criteria" through which to judge their effectiveness (Gitomer et al., 1992; Farr & Tone, 1994). As described in the background, there are many possible student outcomes that can be evidenced through portfolios, whether as a showcase of one's best work or an active documentation of one's growth over time. By clarifying this purpose with students, they can not only have more agency in the process, but also work to develop their own sense of what counts as effective computational communication.

Within computational contexts, one method to help establish this shared criterion is to consider what kinds of language students are already using within their descriptions. While students tended to describe their experiences in vague terms, this is arguably less about lack of intention and more about the difficulty of describing certain experiences using collectively understood language. One strategy might therefore be to expose students to the "vocabulary of computing" as described in the background (Grover et al., 2014). While some forms of computational concepts or practices might be easier to write about because they are more concrete in nature (e.g., how things are electrically connected, how to avoid knotted thread), attending to the vocabulary of computing would make it easier to describe more abstract coding ideas (e.g., how sequences of functions lead to different behaviors, a knowledge of conditional logic).

Here, more thoughtful design direction that gets students to engage with domain-specific *language and vocabulary* could transform portfolios from an instrument that merely demonstrates what students know to a powerful platform upon which to reflect on their progress and strengthen their learning. For instance, introducing students to the difference between a "compile time" coding error (e.g., mistakes in the 'grammar' of the text) and a "runtime" error (e.g., problems with the underlying logic of the program) could not only have given pairs more precise vocabulary for describing their challenges, but also tools to help clarify, and perhaps more effectively tackle, these issues. By incorporating these active opportunities to practice communication and reflection and actively linking these to a shared goal, portfolio use in K-12 computational settings could therefore begin to reach the benefits long seen within other disciplines.

Supporting More Effective Use of Media

How students use media forms to communicate their ideas is another essential issue to consider when looking at portfolios. One advantage of digital portfolios is that they allow for the inclusion of multiple media forms that can perhaps convey more detail than text alone (McKay et al., 2015). Considering that research has illustrated how use of visual representations supports engagement with science and engineering fields ((Hill and Sharma, 2015), which is additionally supported by AP CSP standards (College Board, 2017), portfolios offer a way for students to practice use of these forms. While our findings highlight students' use of media on almost every page of the portfolios, it also illustrates the varied effectiveness of using these to communicate one's understanding or experience. While some pairs simply presented these with minimal annotation or guidance for the viewer, others used more intentional approaches such as creating picture collections, annotating code or images with arrows and text, and color-coding diagrams.

Rather than judging these strategies merely for their effectiveness however, our goal is to consider the myriad ways that students want to use media and support them in using these to their best advantage. In our case, the portfolio format was left open to students, but future research could investigate other arrangements. One such example could be a portfolio inspired by Do-It-Yourself (DIY) culture that could potentially create a new way for students share their ideas (e.g., see McKay et al., 2015). Here, students could lend their own situated expertise with social media toward the creation of the 'shared criteria' for evaluating portfolios mentioned above. Students might also start to compile successful examples and models of media use that can help guide their own portfolio development—something that has been successful within writing contexts as well (Paulson et al., 1991). For instance, this could include samples from existing social media sites that use known conventions such as collaborative hashtags or the creation of non-linear multimedia compilations. In this way, we not only can give students avenues to represent their ideas, but also validate their own background and expertise within this process.

Using Portfolios as a Formative Assessment

In this study, we ended up using digital portfolios as *summative* assessment of students' engagement with computational concepts and practices. This use was mediated by the existing conditions of our study, which are detailed in our methods. As mentioned in our review though, one major affordance of portfolios is their use as a *formative* assessment that can allow teachers to monitor and assist student along the way, and students to document and shape their own pathways of learning (e.g., Adams, 1998, De fina, 1992, Owings & Follo, 1992, Mullin, 1998).

One potential way of converting these portfolios into formative assessments is through the use of journaling, a technique which has been proven successful within writing courses (Mullin, 1998). Here, carefully designed prompts and feedback placed throughout the steps of production could support the recommendations from above—that is, helping develop a shared classroom culture of using domain-specific language and media annotations. One prompt, for instance, might ask students to document a runtime coding issue they have faced using both prose and screenshots of their code, and share these with classmates in order to build up a shared database of problems and issues. In this way, documentation and articulation of computational ideas and practices can become a part of the process of creating a computational artifact.

Another tactic for incorporating the portfolio as a formative assessment would be through creating more purposeful face-to-face interactions surrounding its creation. Within art studios, critique or feedback sessions have long been used as part of formal instruction, and have also successfully been used to support portfolio development (Gitomer et al., 1992). Here, we might consider how existing teacher consultations focused on developing and troubleshooting the computational product itself could also be used to focus on ongoing documentation and reporting. From a research perspective, this would not only highlight the kinds of problems student deal with throughout the process of creating a computational artifact, but also highlight their thinking about how they remember and record these moments while they are occurring. This, in turn, could further inform the design of future CT activities, not just with regard to their hands-on learning, but also to support their ability to continually reflect and learn through this process.

Conclusion

Our analysis of the affordances of portfolios and students' communication strategies helps lay groundwork for future use of portfolios as a form of computational thinking assessment. Based on these findings, our research team has already implemented a revised version of this portfolio assignment with students working on e-textiles as part of a yearlong introductory computer science curriculum (Lui, Jayathirtha, Fields, Shaw & Kafai, 2018). That version creates more defined structures for student reporting on their process, including limiting the number of challenges or revisions to present, as well as what types of media to include, in hopes of increasing students' tendency of using domain-specific details and visual annotation. It also implements a series of journaling prompts throughout the unit, as well as an engineering design notebook where students can keep track of their individual progress in creating their artifacts.

Following our discussion, future research might focus on portfolio-focused activities as a format to develop a shared vocabulary of computation, thereby providing opportunities for students to rehearse, develop and implement their CT-focused communication skills. Here, making a portfolio can become not just a way of reporting on students' computational experiences, but also as an important learning activity in-and-of itself. In this way, use of portfolios as a formative assessment can help to shape the development of future computational thinking activities such that they move beyond mere in-the-moment experiences and actually spur longer-term reflection upon, and subsequently deeper engagements with, computational thinking ideas and practices.

Acknowledgements: This work was supported by a grant # 1509245, 1512760, 1510725 from the National Science Foundation to Yasmin Kafai, Jane Margolis, and Joanna Goode. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation, the University of Pennsylvania, or Utah State University. Special thanks to Breanne Litts and Sari Widman for their work on the original study workshop, from which this paper was generated.

References:

Adams, T.L. (1998). Alternative Assessment in Elementary School Mathematics. Childhood Education, 74(4), 220-224.

Arpaci-Dusseau, A., Astrachan, O., Barnett, D., Bauer, M., Carrell, M., Dovi, R., ... Kick, R. (2013, March). Computer science principles: analysis of a proposed advanced placement course. In R. McCauley, T. Camp, P. Tymann, J.D. Dougherty, & K. Nagel (Eds.), *SIGCSE 13. Proceedings of the 44th ACM technical symposium on Computer Science Education* (pp. 251-256). New York, NY: ACM.

Basu, S., Kinnebrew, J. S., & Biswas, G. (2014, June). Assessing student performance in a computational-thinking based science learning environment. In S. Trausan-Matu, K. Boyer, M. Crosby, & K. Panourgia (Eds.), *IST 2014. Proceedings of the 12th International Conference on Intelligent Tutoring Systems* (pp. 476-481). Honolulu, HI, USA: Springer.

Bienkowski, M., Snow, E., Rutstein, D., & Grover, S. (2015). Assessment design patterns for computational thinking practices in secondary computer science: A first look. Menlo Park, CA, USA: *SRI International*.

Brennan, K., & Resnick, M. (2012, April). New frameworks for studying and assessing the development of computational thinking. In *AERA 2012. Proceedings of the 2012 annual meeting of the American Educational Research Association* (pp. 1-25). Vancouver, Canada.

Buechley, L., Eisenberg, M., Catchen, J., & Crockett, A. (2008, April). The LilyPad Arduino: using computational textiles to investigate engagement, aesthetics, and diversity in computer science education. In M. Burnett, M. Costabile, T. Catarci, B. de Ruyter, D. Tan, M. Czerwinski, & A. Lund (Eds.,), *CHI 2008. Proceedings of the 26th Annual SIGCHI conference on Human factors in computing systems* (pp. 423-432). Florence, Italy: ACM.

Býrgýn, O., & Adnan, B. A. K. Ý. (2007). The Use of Portfolio to Assess Students' Performance. *Journal of Turkish Science Education*, *4*(2), 75.

College Board (2017). Advanced Placement Computer Science Principles Course Guide. Retrieved from https://apcentral.collegeboard.org/pdf/ap-computer-science-principles-course-and-exam-description.pdf

Cooper, S., Pérez, L. C., & Rainey, D. 2010. K-12 computational learning. Communications of the ACM, 53(11), 27-29.

De Fina, A. (1992). Portfolio Assessment: Getting Started. New York, NY: Scholastic Professional Books.

Eris, O. (2007). Insisting on truth at the expense of conceptualization: can engineering portfolios help?. *International Journal of Engineering Education*, *22*(3), 551.

Estell, J. K. (2001). IPP: a web-based interactive programming portfolio. *ACM SIGCSE Bulletin*, *33*(1), 149-153.

Falkner, K., & Falkner, N. J. (2012, February). Integrating communication skills into the computer science curriculum. In L. S. King, D. R. Musicant, T. Camp, & P. Tymann (Eds.). *SIGCSE 2012. Proceedings of the 43rd ACM technical symposium on Computer Science Education* (pp. 379-384). Raleigh, NC, USA: ACM.

Farr, R., & Tone, B. (1994). *Portfolio and performance assessment: helping students evaluate their progress as readers and writers*. Orlando, FL, USA: Harcourt Brace and Company.

Fields, D. A., Lui, D. & Kafai, Y. B. (2017). Teaching computational thinking with electronic textiles: High school teachers' contextualizing strategies in *Exploring Computer Science*. In S. C. Kong, J. Sheldon, & R. K. Y. Li (Eds.), *CTE 2017. Conference Proceedings of International Conference on Computational Thinking Education 2017* (pp.67-72). Hong Kong, SAR, China: The Education University of Hong Kong.

French, J. H. (2012). Evaluating a communication-intensive core course in the CS curriculum. *Journal of Computing Sciences in Colleges*, *28*(2), 197-209.

Gilman, D. A., Andrew, R., & Rafferty, C. D. (1995). Making assessment a meaningful part of instruction. *NASSP bulletin*, *79*(573), 20-24.

Gitomer, D., Grosh, S., & Price, K. (1992). Portfolio culture in arts education. *Art Education*, *45*(1), 7-15.

Grover, S., Cooper, S., & Pea, R. (2014, June). Assessing computational learning in K-12. In A. Cajander, M. Daniels, T. Clear & A. Pears (Eds). *ITICSE 14. Proceedings of the 2014 conference on Innovation & technology in computer science education* (pp. 57-62). Uppsala, Sweden: ACM.

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

Hebert, E. A. (2001). *The Power of Portfolios: What Children Can Teach Us about Learning and Assessment. The Jossey-Bass Education Series*. San Francisco, CA: Jossey-Bass.

Herman, J. L., Gearhart, M., & Aschbacher, P. R. (1996). Portfolios for classroom assessment: Design and implementation issues. In R.C. Calafee & P. Perfumo (Eds.), *Writing portfolios in the classroom: Policy and practice, promise and peril* (pp. 27-59). New York, NY: Routledge.

Higgs, B., & Sabin, M. (2005, October). Towards using online portfolios in computing courses. In R. Friedman (Ed.), *SIGITE 2005. Proceedings of the 6th conference on Information technology education* (pp. 323-328). Newark, NJ, USA: ACM.

Hill, M., & Sharma, M. D. (2015). Students' representational fluency at university: A cross-sectional measure of how multiple representations are used by physics students using the representational fluency survey. *Eurasia Journal of Mathematics, Science and Technology Education*, *11*(6), 1633-1655.

Kafai, Y. B., Fields, D. A., & Searle, K.A. (2012). Making learning visible: Connecting crafts, circuitry & coding in e-textile designs. In van Aalst, J., Thompson, K., Jacobson, M.J., & Reimann, P. (Eds.), *ICLS 2012. Proceedings of the 10th International Conference of the Learning Sciences* (pp. 188-195). Sydney, NSW, Australia: International Society Of Learning Sciences.

Koh, K. H., Basawapatna, A., Bennett, V., & Repenning, A. (2010, September). Towards the automatic recognition of computational thinking for adaptive visual language learning. In C. Hundhausen, E. Pietriga, P. Díaz & M. Rosson (Eds.), *VL/HCC 2010. Proceedings of 2010 IEEE Symposium on Visual Languages and Human-Centric Computing* (pp. 59-66). Leganés-Madrid, Spain: IEEE.

Litts, B. K., Kafai, Y. B., & Dieckmeyer, E. (2015). Collaborative electronic textile designs by high school youth: Challenges and opportunities in connecting crafts, circuits, and code. In *FabLearn 2015. Proceedings of the 5th[t] Annual Conference on Creativity and Fabrication in Education*. Stanford, CA.

Litts, B. K., Kafai, Y. B., Lui, D. A., Walker, J. T., & Widman, S. A. (2017a). Stitching Codeable Circuits: High School Students' Learning About Circuitry and Coding with Electronic Textiles. *Journal of Science Education and Technology*, *26*(5), 494-507.

Litts, B.K., Lui, D., Widman, S. A., Walker, J.T., & Kafai, Y.B. (2017b). Reflections on Pair E-Crafting: High School Students' Approaches to Collaboration in Electronic Textiles Projects. In Smith, B. K., Borge, M., Mercier, E., and Lim, K. Y. (Eds.), CSCL 2017. *Proceedings of the 12th International conference on Computer Support for Collaborative Learning* (Volume 2). Philadelphia, PA, USA: International Society Of Learning Sciences.

Litts, B.K., Widman, S.A., Lui, D., Walker, J.T., & Kafai, Y.B. (in press). A Maker Studio Model for High School Classrooms: The Nature and Role of Critique in an Electronic Textiles Design Project. *Teachers College Record.*

Lui, D., Jayathirtha, G., Fields, D.A., Shaw, M., & Kafai, Y.B. (2018). Design Considerations for Capturing Computational Thinking Practices in High School Students' Electronic Textile Portfolios. In *ICLS 2018. Proceedings of the International Conference of Learning Sciences*. London, UK: International Society Of Learning Sciences.

Lui, D., Litts, B.K., Widman, S.A., Walker, J.T., & Kafai, Y.B. (2016). Collaborative Maker Activities in the Classroom: Case Studies of High School Student Pairs' Interactions and Perceptions in Designing Electronic Textiles. In P. Blikstein, M. Berland, & D. Fields (Eds.), *Proceedings of the 6th Annual Conference on Creativity and Fabrication in Education* (pp. 74-77). Stanford, CA: ACM.

McKay, C., Keune, A., Peppler, K. Chang, S. & Regalla, L. (2015). A networked vision for sharing and documenting - Open Portfolio Project. Maker Education Initiative. Retrieved from http://makered.org/wp-content/uploads/2016/01/MakerEdOPP_full-Research-Brief-Series_final.compressed.pdf

Mullin, J. A. (1998). Portfolios: Purposeful collections of student work. *New directions for teaching and learning*, *1998*(74), 79-87.

Owings, C. A., & Follo, E. (1992). Effects of portfolio assessment on students' attitudes and goal setting abilities in mathematics. Retrieved from https://files.eric.ed.gov/fulltext/ED352394.pdf

Papert, S., & Harel, I. (1991). Situating constructionism. *Constructionism*, *36*(2), 1-11.

Paulson, F. L., Paulson, P. R., & Meyer, C. A. (1991). What makes a portfolio a portfolio. *Educational leadership*, *48*(5), 60-63.

Phelps, A. J., LaPorte, M. M., & Mahood, A. (1997). Portfolio assessment in high school chemistry: One teacher's guidelines. *J. Chem. Educ*, *74*(5), 528.

Repenning, A., Smith, C., Owen, R., & Repenning, N. (2012, June). AgentCubes: Enabling 3D creativity by addressing cognitive and affective programming challenges. In T. Amiel, & B. Wilson (Eds.), EdMedia 2012. *Proceedings of World Conference on Educational Media and Technology* (pp. 2762-2771). Denver, CO: Association for the Advancement of Computing in Education (AACE).

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

Werner, L., Denner, J., & Campe, S. (2014, January). Using computer game programming to teach computational thinking skills. In K. Schrier (Ed.), *Learning, Education and Games* (pp. 37-53). Pittsburgh, PA, USA: ETC Press.

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