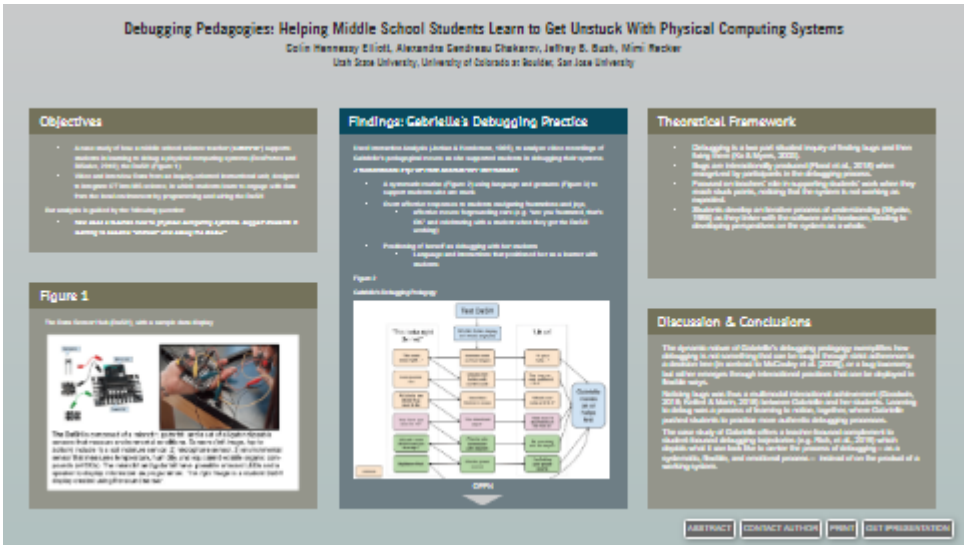
Debugging Pedagogies: Helping Middle School Students Learn to Get Unstuck With Physical Computing Systems



Colin Hennessy Elliott, Alexandra Gendreau Chakarov, Jeffrey B. Bush, Mimi Recker

Utah State University, University of Colorado at Boulder, San Jose State University

PRESENTED AT:

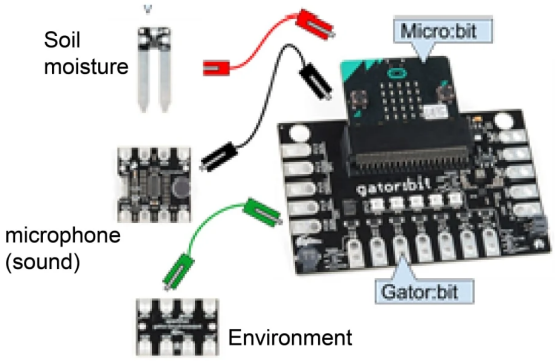
OBJECTIVES

- Case Study of a middle school science teacher's (**Gabrielle**) support of students learning to debug physical computing systems (DesPortes and DiSalvo, 2019), the DaSH (Figure 1).
- Video and interview data from an inquiry-oriented instructional unit, designed to integrate CT into MS science

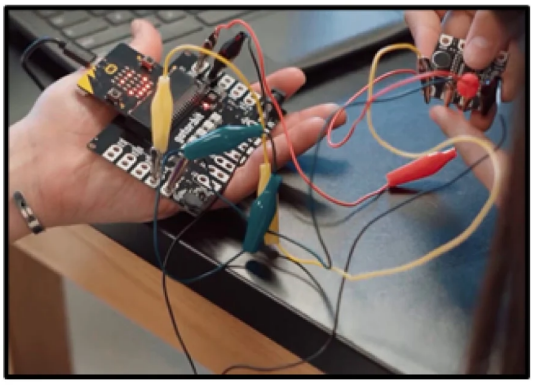
Our analysis is guided by the following question:

- *How does a teacher, new to physical computing systems, support students in learning to become "unstuck" and debug the DaSH?*

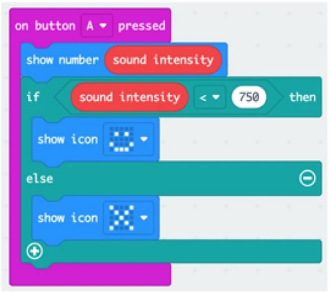
Figure 1: The Data Sensor Hub (DaSH), with a sample data display



A student's version using the microphone sound sensor



The Data Sensor Hub (DaSH) has clippable sensors that can measure environmental conditions like soil moisture, sound (microphone), Carbon Dioxide, Temperature, Air Pressure, and more. Programmed with MakeCode (block based language shown to the right)



FINDINGS: GABRIELLE'S DEBUGGING PRACTICE

Interaction Analysis (Jordan & Henderson, 1995), to analyze video of Gabrielle's as she supported students in debugging their systems.

3 foundational aspects that characterize interactions

- A systematic routine (Figure 2) using language and gestures (Figure 3) to support students who are stuck.
- Overt affective responses to students navigating frustrations and joys,
- Positioning of herself as debugging with her students
  - Language and interactions that positioned her as a learner with students

Figure 2: Gabrielle's Debugging Pedagogy

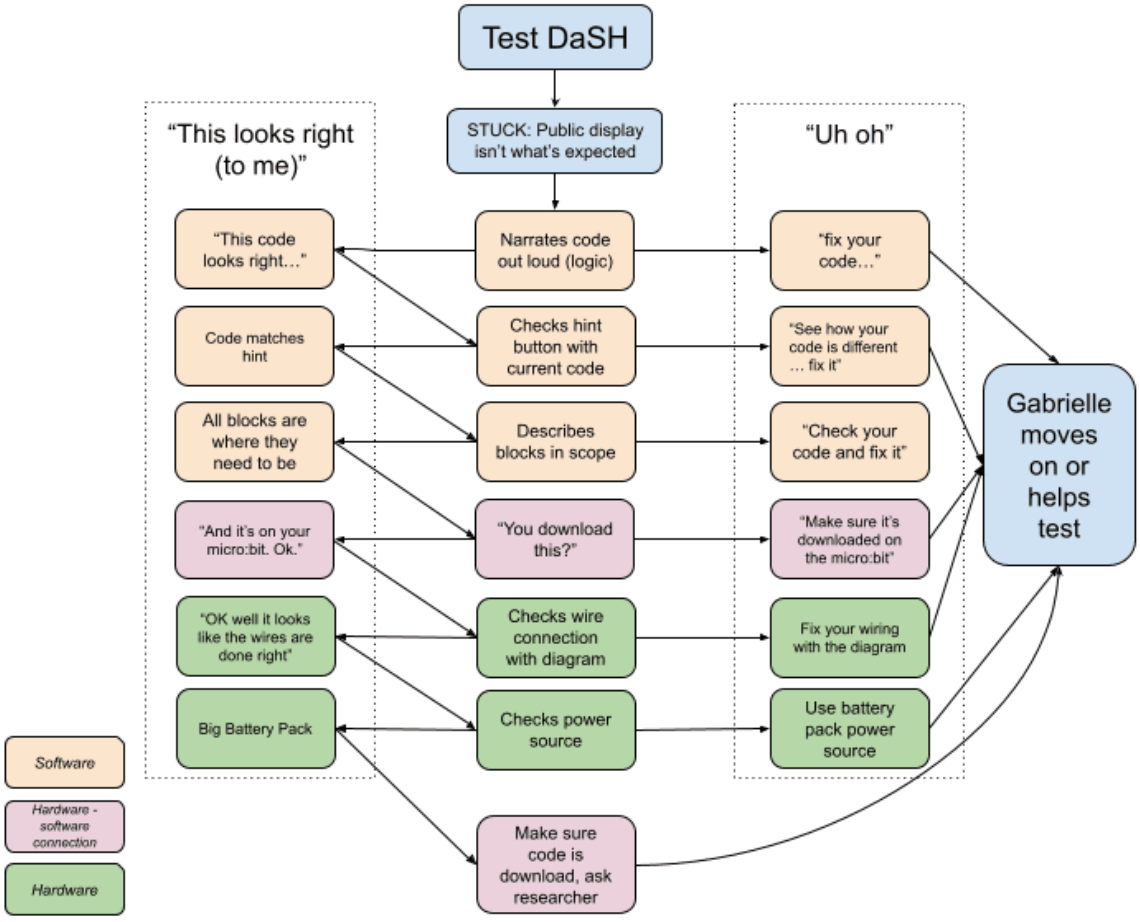
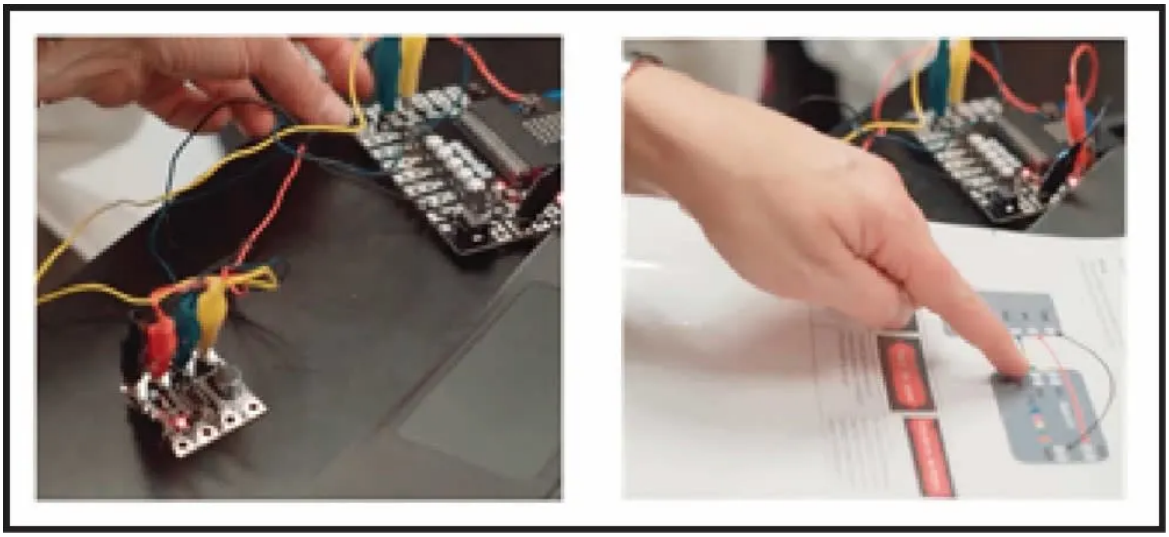


Figure 3: Gabrielle's gestures to the gator:bit connection and the corresponding wiring diagram



THEORETICAL FRAMEWORK: TEACHER'S ROLE IN DEBUGGING AS SITUATED INQUIRY

- Supporting students' work when they reach stuck points, noticing that the system is not working as expected.
- Debugging is a two part situated inquiry: finding bugs and fixing them (Ko & Myers, 2003).
- Bugs are interactionally produced (Flood et al., 2018).
- Students develop an iterative process of understanding (Miyake, 1986) as they tinker, leading to developing perspectives on whole system.

DISCUSSION & CONCLUSIONS

The dynamic nature of Gabrielle's debugging pedagogy exemplifies how debugging emerges through interactional practices deployed in flexible ways, not decision trees or beg taxonomies.

Noticing bugs was a multimodal interactional achievement (Goodwin, 2018; Keifert & Marin, 2018) between Gabrielle and her students. Learning to debug was a process of learning to notice.

The case study of Gabrielle offers a teacher-focused complement to student-focused debugging trajectories (e.g. Rich, et al., 2019) centering the process of debugging – as a systematic, flexible, and emotional process – instead of on the product of a working system.

Acknowledgements: We are greatful for our SchoolWide Labs collaborators Tammy Sumner, Jennifer Jacobs, Quentin Biddy, Srinjita Bhaduri, Michael Schneider, Jay Luther, and more. We are thankful for our partnering teachers, like Gabrielle, who have shown incredible humility and vulnerability letting us into their classrooms. And we are thankful for the students whose learning we are exaining. This research was funded by the National Science Foundation (Award No. 1742053, No. 2019805 and No. 1742046) and the James S. McDonnell Foundation. The opinions expressed herein are those of the authors and do not necessarily reflect those of the National Science Foundation.

ABSTRACT

This paper draws on data collected during an inquiry-oriented instructional approach in which students learn to program a sensor-based physical computing system to collect and display meaningful data from the world around them. As part of one instructional unit (Sensor Immersion Unit) students debug their system when it does not work as they expect it to. We present a case study of how one teacher (Gabrielle) acted as a caring collaborator with students as they addressed hardware and software problems. This included modeling and articulating a regular systematic approach to becoming “unstuck,” which we map in analysis. Gabrielle’s approach to supporting students, or her debugging pedagogy, positions debugging as core computing practice rather than as a means to overcome failure.

Debugging pedagogies: Helping middle school students learn to get unstuck with physical computing systems

Objectives

There is considerable demand to offer computing education to all students at all grade levels (Brennan & Resnick, 2012). While broad exposure helps, not all programs have equitable outcomes so the type of computing students in middle grades are exposed to needs research attention (Author & Colleagues, 2020). One promising approach, called physical computing, enables a range of students to pursue personally relevant design tasks (Anastopoulou et al., 2012) using a combination of hardware and software tools (e.g., Arduino, micro:bit). With the growth of this approach, there is a need to help teachers learn to support students, especially as they debug hardware and software issues that emerge across the system (DesPortes & DiSavlo, 2019).

This paper presents a case study of a middle school teacher, Gabrielle (all names are pseudonyms), focusing on instances where she supported students when they were "stuck" (Haduong & Brennan, 2019) and worked on debugging a programmable sensor system called the Data System Hub (DaSH). The case study primarily draws on analyses of videos of classroom practices to trace Gabrielle's *debugging pedagogy*. We focus on her practices in the classroom working with two small groups of students to highlight the pedagogical possibilities for: 1) systematic support of students during debugging, 2) affective support, and 3) positioning teachers as debugging *with* students and not *for* students.

Our analysis is guided by the following question: *how does a teacher, new to physical computing systems, support students in learning to become "unstuck" and debug the DaSH?*

Theoretical Framework

Physical computing systems offer an alternative set of introductory activities to engage students in computing and can broaden students' definition of computer science as a discipline (Kafai et al., 2014). Sensor-based physical computing can help make invisible phenomena visible for students and thus become an integral part of students' authentic engagement with increasingly technologically-driven scientific and data practices (Hardy et al., 2020). However, physical computing systems can be challenging for computing novices to work with because they involve both software and hardware. Bugs not only occur in software, but also hardware and hardware-software connections (DesPortes & DiSalvo, 2019). In the context of a classroom, we view bugs as interactionally produced (Flood et al., 2018) when discrepancies emerge between what a physical data display does and the expectations of the student, teacher or both.

There has been a wealth of prior research characterizing: 1) typical programming bugs (Ahmadzadeh et al., 2005), 2) novice misconceptions that lead to bugs (Grover & Pea, 2013; Pea, 1986), and 3) successful debugging approaches for learners (Ko & Myers, 2005) and teachers (Kim et al., 2018). Developing debugging skills is a core component of computer science and computational thinking (Grover & Pea, 2013) with several computational thinking frameworks specifically identifying debugging as a core practice (Blikstein & Mogham, 2019; Webb, 2010).

Studies focusing on how teachers learn to support students' debugging, including the affective dimensions, are more limited (Flood et al., 2018). Curricula that support students to learn through struggle have been shown to be effective (Kapur & Bielaczyc, 2012), and debugging tasks are often challenging. By supporting a greater focus on the process, debugging can be viewed as a core component in learning computer science, not just as a means to an end (Kafai et al., 2020).

A need remains to better understand how teachers who may lack content expertise draw upon their own pedagogical expertise to facilitate groups of students as they engage in debugging tasks and develop perseverance and problem solving skills while working to become unstuck (Haduong & Brennan, 2019). This paper contributes to such a general pedagogy of debugging support.

Methods

This study is part of a larger project between two universities and a large urban school district to develop instructional units that integrate computing in middle school science and STEM classes (Authors, 2021). These units use the DaSH (Figure 1), a physical computing system composed of a micro:bit, gator:bit, and a variety of alligator clippable sensors. The block-based programming interface (MakeCode) controls the DaSH, thus providing a simpler computational entry for students to prototype with.

Teachers and researchers co-designed the Sensor Immersion Unit (SIU), a one-week inquiry-oriented unit. The goal of the instructional unit is to introduce students to the affordances of the DaSH for supporting scientific investigations by collecting data and creating data displays (see Figure 1) (Authors, 2021). Classroom implementations of the SIU were video and audio recorded, focused on the teachers' actions.

Data Sources

Drawing from the tradition of Interaction Analysis (Jordan & Henderson, 1995), analysis began with a corpus review of classroom video data of all ten teachers who implemented the SIU during its first year. Gabrielle was selected as a focal teacher because her actions were captured clearly and her debugging process was systematic. We reviewed and logged recordings of Gabrielle's implementation, inductively coding her work with students. Once a working analysis was constructed, we triangulated with the secondary data sources to confirm our characterization of her practice (Table 1).

Below, we share a focal episode to depict analysis of her approaches to interactions with students. In the focal episode, Gabrielle performed supportive successive interactions for two students who did not know why their DaSH was not displaying as they expected. Following Interaction Analysis conventions, the six minutes of video were transcribed - paying attention to talk, gesture, and resources - and analyzed repeatedly, individually and as a group. We share a sample of the analysis below.

Findings

Gabrielle's interactions with students when they indicated they were stuck had several foundational aspects: a systematic checking routine, language and gestures that oriented students to possible issues, overt affective responses to students navigating frustrations and joys, and consistent positioning of herself as a learner with students such that she was debugging with them. The dynamic nature of Gabrielle's debugging pedagogy, therefore, helps depict how debugging emerges through interactional frameworks that must be deployed flexibly.

Systematically finding Bugs with students

Our analysis of Gabrielle's systematic approach to supporting students as they struggled with their DaSHs is summarized in Figure 3. It begins with "Test": when Gabrielle arrived at a student's work area they checked the output of the DaSH. This oriented a shared understanding of how the DaSH was functioning. Gabrielle would then initiate a series of checks (steps in the middle column of Figure 3). Doing so modeled a procedural approach to how to diagnose and address potential bugs. Gabrielle often narrated what she was looking at. Her next move depended on if she noticed an issue. If she verbalized that it looked correct, she moved to the next check (the arrow between the middle column and the left column, downward). If something seemed wrong, she indicated there was an issue with an utterance like "uh oh" (as in the focal episode) and made gestures and utterances to prime students to notice where the bug might be and plan to fix it.

If no bug was identified, Gabrielle offered a next step for students to further investigate (the bottom-most box in Figure 3): to download the program onto the micro:bit again. If this did not work, Gabrielle would sometimes reach out to the researcher if they were in the room. If the researcher was not in the room, she would say she planned on sharing the issue with them at another time.

Figure 3 represents the order that Gabrielle checked the DaSH components with students: software, then software-hardware connections (question to students), then hardware. This order appeared consistent across her recorded practice.

Noticing bugs to develop future debugging

Gabrielle's routine (Figure 3) did interactional work to get students to notice the bug and the process they could take to find bugs in the future. We focused on representing her language in Figure 3 to make sense of her moves to include students in the process and build their skills for debugging. This figure does not include the gestures and instructional resources she used in conjunction with talk.

In the following focal episode (see transcript in Figure 4), we share analysis to depict how she used interactional resources (including gestures) as she supported students. At the beginning of the episode, Gabrielle was working with Sarah (a student) whose micro:bit LED screen only showed numbers. Sarah expected the DaSH to show either a smiley face or an X depending on the measured level of sound. Gabrielle initially tested the DaSH, expecting it to show an X when she spoke loudly (repeated Hellos, line 1, Figure 4) into the sound sensor.

Leaning over the computer, Gabrielle read the program out loud, sharing the threshold number ("seven fifty then", line 1) that Sarah had input in the logic statement (as directed in the activity). She verbalized that it looked right (lines 1-2), then asked whether Sarah made the connection between the code and the DaSH ("you download this… and it's on the micro:bit," lines 2-4). She continued narrating her debugging process and used language that included Sarah ("we" and "our wires"; line 4) to signify doing it together. The order, tone, and content of these moves was consistent with other interactions with students (Figure 3).

Lines 4 through 8 depict how Gabrielle moved to looking at "the wires." After saying "uh oh" (line 6), she tapped the gator:bit connection labeled SDA (and connected with the green wire), and then exaggeratedly tapped the same spot on the wiring diagram (both depicted in Figure 5). These environmentally-coupled gestures (Goodwin, 2018) coordinated an exaggerated engagement with the sensor-Gator:bit connections (alligator clip wires) and the wiring diagram (line 8). Gabrielle continued to make bids to get Sarah to notice the possible issue, repeating "uh oh" twice (lines 9 and 10) and sharing that the bug is "a tricky one" (line 12) before pausing. Her embodied move, pointing from the wire to the wiring diagram and back shifted the source of recognizing the bug away from Gabrielle and towards the resource Sarah could use when she was not there: the wiring diagram.

Caring for frustrations and joys

Learning to debug is an emotional process that can be augmented with explicit focus on supporting youth in naming their emotional responses (Dahn & DeLiema, 2020). Gabrielle wove these affective register moves with the debugging focused interactional moves we articulate above. Caring moves were in-the-moment utterances and/or gestures that celebrated and supported students as they engaged in debugging practice, validating their frustrations when they struggled to get unstuck and encouraging excitement when the system worked.

Teacher as learning with the students

Gabrielle consistently positioned herself as a learner, along with her students, when facilitating debugging practices. She did this in multiple ways. One way was talking about one of the researchers who often visited the classroom (as in Figure 3, bottom-most box) as a person with a greater wealth of expertise about the DaSH. Gabrielle publicly recognized her own uncertainties and questions. She, therefore, modeled computing and debugging as an iterative and inquisitive process, not a finite achievement of perfection.

Significance

Our case study analysis revealed important characteristics of the teacher's pedagogical interactions with students during debugging: a systematic checking routine that she modeled out loud and highlighted resources, overt affective responses, and consistent positioning of herself as a learner debugging *with* not *for* her students. For Gabrielle, feeling less confident about the computer science content became a strength in helping students develop debugging skills. Her debugging pedagogy illustrates how debugging skills are neither specific content that can be distilled, nor practices that can be explicitly taught as a routine. The skills of debugging are core

parts of computer programming practice (DeLiema et al., 2019) and thus a major component in developing computational thinking. Gabrielle's case illuminates the need for future research to understand how to support teachers in developing facilitation skills as part of their own debugging pedagogy. It also underscores the importance of learning from teachers' practices in the classroom, particularly by engaging teachers of various skills and confidence in computer science instruction.

References

Authors & Colleagues. (2021)

Author & Colleagues. (2021)

Author & Colleagues. (2020)

Ahmadzadeh, M., Elliman, D., & Higgins, C. (2005). An Analysis of Patterns of Debugging Among Novice Computer Science Students. In *Proceedings of the 10th annual SIGCSE conference on Innovation and technology in computer science education* (pp. 84-88)

Anastopoulou, S., Sharples, M., Ainsworth, S., Crook, C., O'Malley, C., & Wright, M. (2012). Creating Personal Meaning through Technology-Supported Science Inquiry Learning across Formal and Informal Settings. *International Journal of Science Education*, *34*(2), 251–273.

Blikstein, P., & Moghadam, S. (2019). Computing education: Literature review and voices from the field. In S. Fincher & A. Robins (Eds.), The Cambridge Handbook of Computing Education Research (pp. 56–78). Cambridge: Cambridge University Press.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. 25.

Coburn, C. E., Penuel, W. R., & Geil, K. E. (2013). Research-Practice Partnerships: A Strategy for Leveraging Research for Educational Improvement in School Districts. William T Grant Foundation.

Dahn, M., & DeLiema, D. (2020). Dynamics of emotion, problem solving, and identity: Portraits of three girl coders. *Computer Science Education*, *30*(3), 362–389.

DeLiema, D., Dahn, M., Flood, V. J., Asuncion, A., Abrahamson, D., Enyedy, N., & Steen, F. (2019). Debugging as a Context for Fostering Reflection on Critical Thinking and Emotion. In E. Manalo (Ed.), *Deeper Learning, Dialogic Learning, and Critical Thinking* (1st ed., pp. 209–228). Routledge.

DesPortes, K., & DiSalvo, B. (2019). Trials and Tribulations of Novices Working with the Arduino. *Proceedings of the 2019 ACM Conference on International Computing Education Research*, 219–227.

Fields, D. A., Searle, K. A., & Kafai, Y. B. (2016). Deconstruction Kits for Learning: Students' Collaborative Debugging of Electronic Textile Designs. *Proceedings of the 6th Annual Conference on Creativity and Fabrication in Education*, 82–85.

Flood, V. J., DeLiema, D., Harrer, B. W., & Abrahamson, D. (2018). Enskilment in the Digital Age: The Interactional Work of Learning to Debug. In J. Kay & R. Luckin (Eds.), Rethinking Learning in the Digital Age: Making the Learning Sciences Count, 13th International

Conference of the Learning Sciences (ICLS) 2018, Volume 3 (pp. 1405-1406). London, UK: International Society of the Learning Sciences.

Goodwin, C. (2018). *Co-operative action*. Cambridge University Press.

Grover, S., & Pea, R. (2013). Computational Thinking in K–12: A Review of the State of the Field. *Educational Researcher*, *42*(1), 38–43.

Haduong, P., & Brennan, K. (2019). Helping K–12 Teachers Get Unstuck with Scratch: The Design of an Online Professional Learning Experience. *Professional Development*, 7.

Hall, R., & Stevens, R. (2016). Developing Approaches to Interaction Analysis of Knowledge in Use. In A. A. DiSessa, M. Levin, & N. J. S. Brown (Eds.), *Knowledge and interaction: A synthetic agenda for the learning sciences*. Routledge.

Hardy, L., Dixon, C., & Hsi, S. (2020). From Data Collectors to Data Producers: Shifting Students' Relationship to Data. *Journal of the Learning Sciences*, *29*(1), 104–126.

Jordan, B., & Henderson, A. (1995). Interaction Analysis: Foundations and Practice. *Journal of the Learning Sciences*, *4*(1), 39–103.

Kafai, Y. B., Lee, E., Searle, K., Fields, D., Kaplan, E., & Lui, D. (2014). A Crafts-Oriented Approach to Computing in High School: Introducing Computational Concepts, Practices, and Perspectives with Electronic Textiles. *ACM Transactions on Computing Education*, *14*(1), 1–20.

Kafai, Y., Hutchins, N., Snyder, C., Brennan, K., Haduong, P., DesPortes, K., DeLiema, D., Aalst, O. W., Flood, V., Fong, M., Fields, D., Gresalfi, M., Brady, C., Steinberg, S., Franklin, D., Eatinger, D., Coenraad, M., Palmer, J., Weintrop, D., … Bulalacao, N. (2020). *Turning Bugs into Learning Opportunities: Understanding Debugging Processes, Perspectives, and Pedagogies*. In Gresalfi, M. and Horn, I. S. (Eds.), The Interdisciplinarity of the Learning Sciences, 14th International Conference of the Learning Sciences (ICLS) 2020, Volume 1 (pp. 374-381). Nashville, Tennessee: International Society of the Learning Sciences.

Kapur, M., & Bielaczyc, K. (2012). Designing for Productive Failure. *Journal of the Learning Sciences*, *21*(1), 45–83.

Kim, C., Yuan, J., Vasconcelos, L., Shin, M., & Hill, R. B. (2018). Debugging during block-based programming. *Instructional Science*, *46*(5), 767–787.

Ko, A. J., & Myers, B. A. (2005). A framework and methodology for studying the causes of software errors in programming systems. *Journal of Visual Languages & Computing*, *16*(1–2), 41–84.

Lewis, C. M. (n.d.). The importance of students' attention to program state: A case study of debugging behavior. 8.

Pea, R. (1986). Language-Independent Conceptual "Bugs" in NoviceProgramming. *Journal of Educational Computing Research*, *2*(1), 25–36.

Penuel, W. R., Riedy, R., Barber, M. S., Peurach, D. J., LeBouef, W. A., & Clark, T. (2020). Principles of Collaborative Education Research With Stakeholders: Toward Requirements for a New Research and Development Infrastructure. *Review of Educational Research*, *90*(5), 627–674.

Peppler, K. (2013). STEAM-Powered Computing Education: Using E-Textiles to Integrate the Arts and STEM. *Computer*, *46*(9), 38–43.

Rich, K. M., Strickland, C., Binkowski, T. A., & Franklin, D. (2019). A K-8 Debugging Learning Trajectory Derived from Research Literature. *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 745–751.

Webb, D. C. (2010). Troubleshooting assessment: An authentic problem solving activity for it in education. *Procedia: Social and Behavioral Sciences*, *9*, 903–907.

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.

Tables

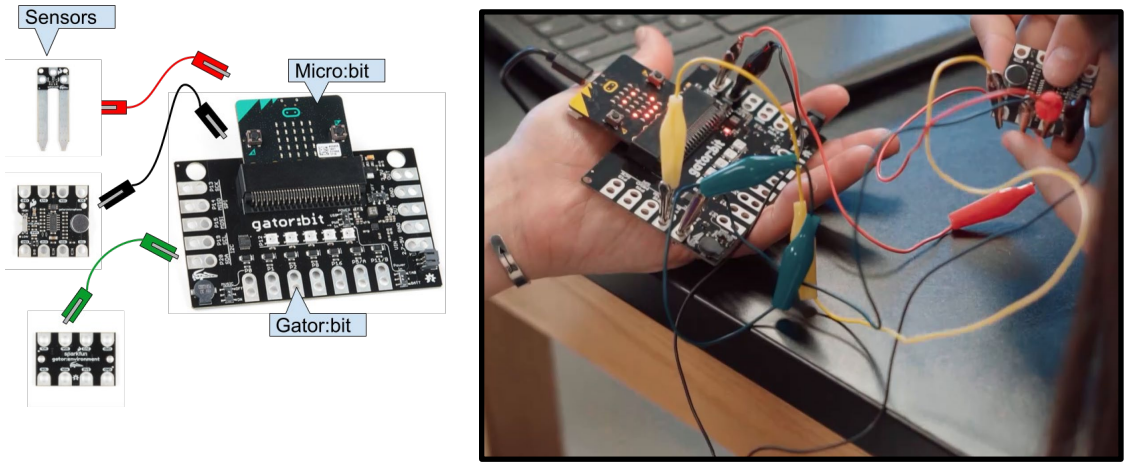
Table 1. Data Sources, methods, and triangulation for analysis of debugging pedagogy

Data sources	Analytic methods	Triangulation strategy
Classroom video recording of SIU implementation	Reviewed, content logged, and coded for debugging practice. Selected focal teacher and focal episode.	Focal episode analysis was then applied to other instances of debugging pedagogy for the focal teacher. Focal episode analysis was compared to reviews of other teachers' practice
Focal episode (6-minute video)	Transcribed for talk, gesture, and engagement with materials. Reviewed repeatedly by researchers together and individually, iterating the transcription and a developing analysis of Gabrielle's process	Focal teacher mentions 1) researcher and 2) professional development in the focal episode (and other moments of classroom practice), which prompted review of PD video data.
Professional Development Session video recording	Reviewed and content logged for places where the sensor immersion unit and debugging in particular were discussed.	Specific moments where focal teacher shares her experience or discusses the sensor immersion unit were logged and reviewed to draw connections to analysis of her practice

Figures

Figure 1

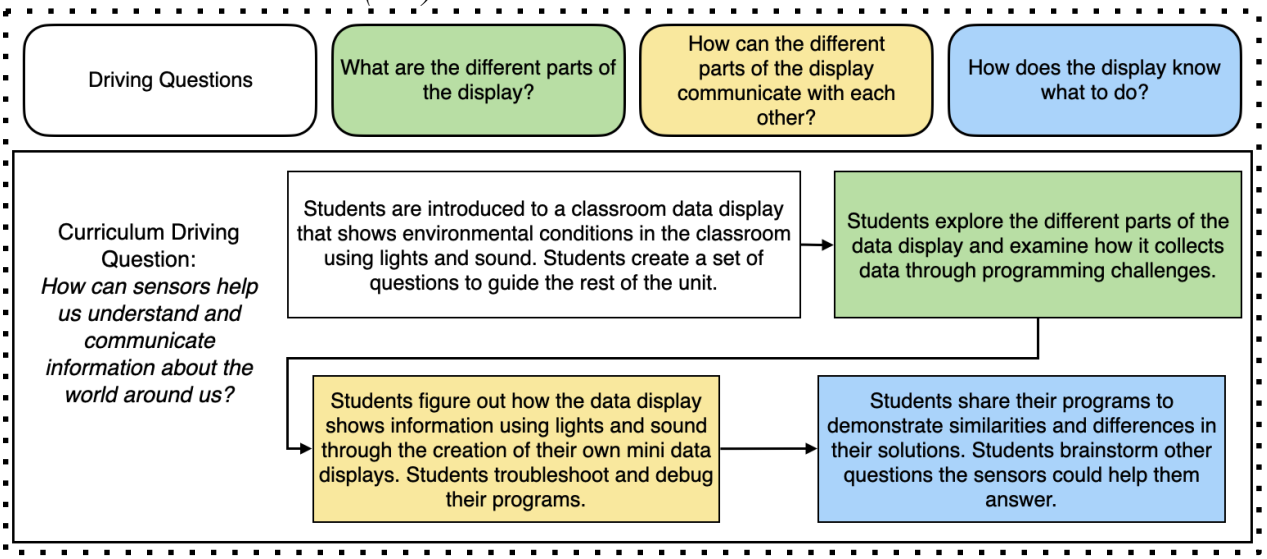
The DaSH system, including a sample data display



Note: The DaSH is composed of a micro:bit, gator:bit, and a set of alligator clippable sensors that can measure different environmental conditions such as temperature, noise level, or UV level. The micro:bit and gator:bit support the display of information through onboard LEDs and a speaker. The image on the right is one such display created by a student.

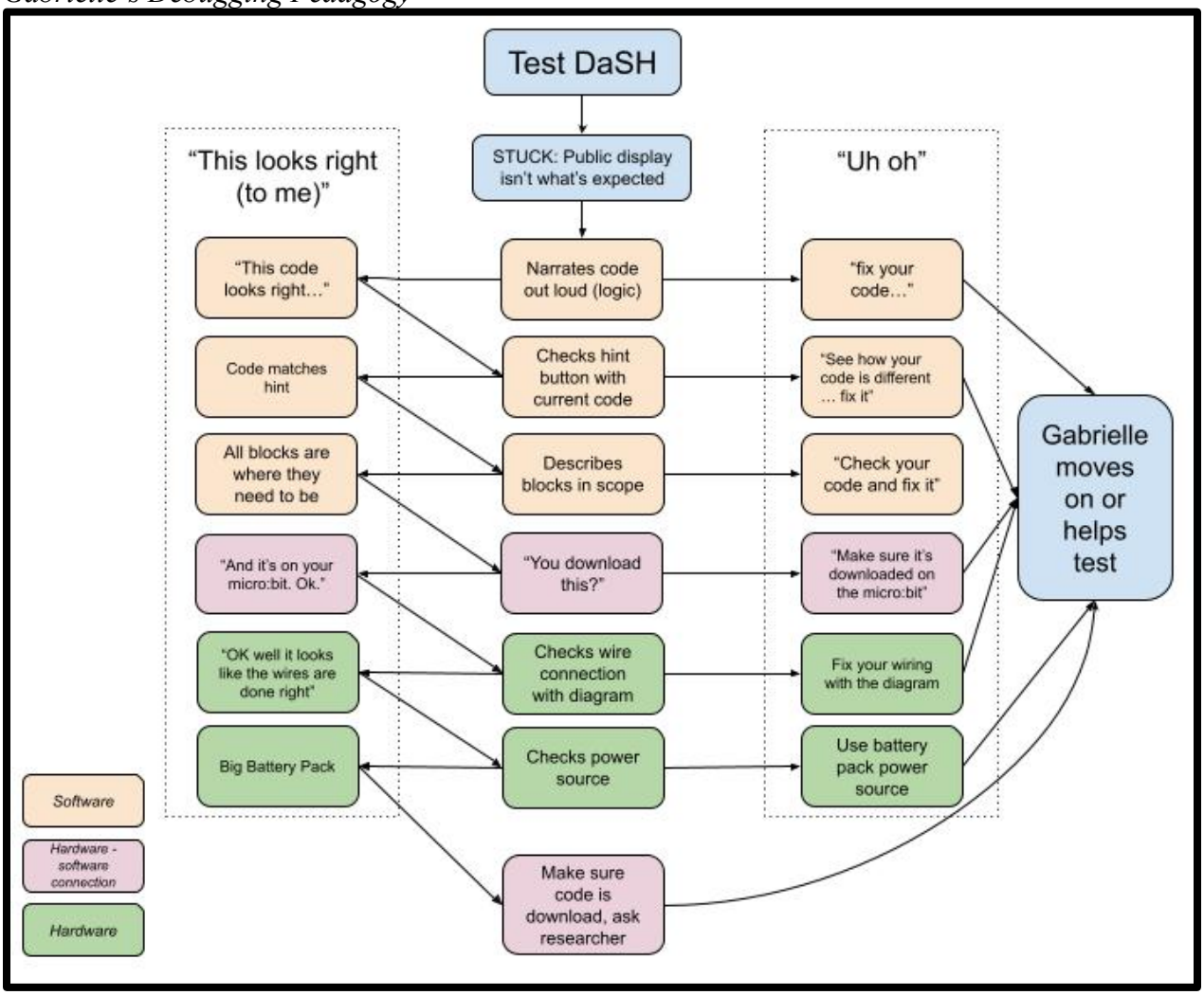
Figure 2

The Sensor Immersion Unit (SIU)



Note: The image outlines the flow for the sensor immersion unit and highlights the questions that students are trying to answer as they create their own data displays using the DaSH.

Figure 3
Gabrielle's Debugging Pedagogy



Note: Flowchart Representing Gabrielle's Debugging Pedagogical Practices

Figure 4

Transcript Excerpt

Gabrielle: Hello Hello Hello::: (1s) Hello Hello Hello::: Ok seven fifty then. Ok this code looks r::ight. (inaudible) Ok you downloaded this?

Sarah: Mmhm=

Gabrielle: =And it's on your micro:bit. Ok. So then we looked at our wires. Ok Three-V-Three. ((*picks up the sensor; another student says something*)) ((*Drops sensor and picks it back up*)) What. Ut oh. ((*puts sensor down and grabs the gator:bit connection; Taps the SDL connection with the green wire with left index finger*)). ((grabs the wiring diagram)) … Ok. Ok uh oh. Now that's a tricky one. Do you see this one ((*touches SDL gator:bit connection*)). Ut oh. Your green's s-- wait are you environmental. Or sound.

Sarah: Sound.

Gabrielle: Uh oh. ((points to the wiring diagram)) Look the green says SDA to SDA. ((Picks up the sensor)) Did you do that. Oh no there's a SDA with no green in it. ((sets down sensor)) Double check your wires again ok sweetheart. No big deal. It's not a big deal. Are you OK.

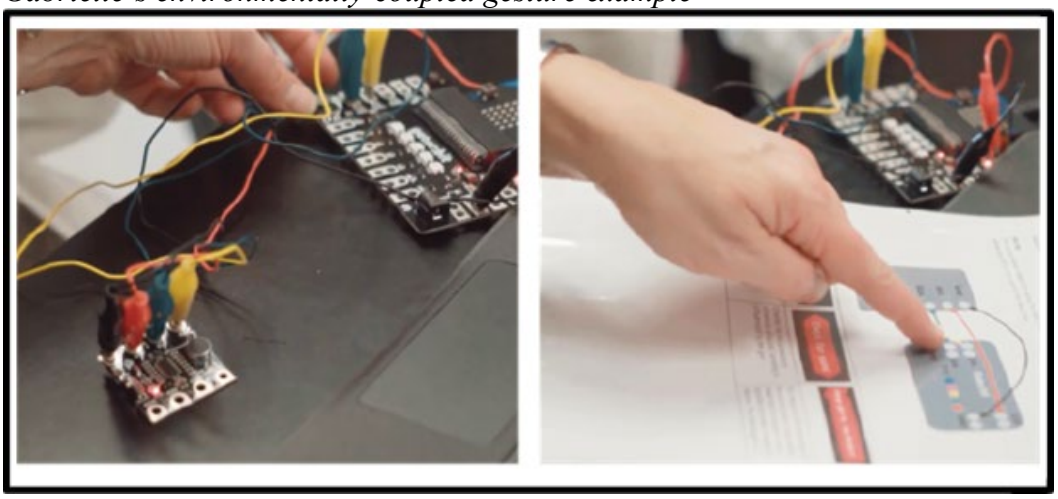
Sarah: ((picks up the sensor)) MMhmm.

Gabrielle: ((slight laugh)) are you frustrated.

Note: Transcribed with conventions adapted from Jefferson (2003)

Figure 5

Gabrielle's environmentally-coupled gesture example



Note: This shows Gabrielle's gestures with the DaSH and wiring diagram over 2 seconds.