

Probeware for the Modern Era: IoT Dataflow System Design for Secondary Classrooms

Leslie G. Bondaryk, Sherry Hsi, and Seth Van Doren

Abstract—Sensor systems have the potential to make abstract science phenomena concrete for K–12 students. Internet of Things (IoT) sensor systems provide a variety of benefits for modern classrooms, creating the opportunity for global data production, orienting learners to the opportunities and drawbacks of distributed sensor and control systems, and reducing classroom hardware burden by allowing many students to “listen” to the same data stream. To date, few robust IoT classroom systems have emerged, partially due to lack of appropriate curriculum and student-accessible interfaces, and partially due to lack of classroom-compliant server technology. We present an architecture and sensor kit system that addresses issues of sensor ubiquity, acquisition clarity, data transparency, reliability, and security. The system has a dataflow programming interface to support both science practices and computational data practices, exposing the movement of data through programs and data files. The IoT Dataflow System supports authentic uses of computational tools for data production through this distributed cloud-based system, overcoming a variety of implementation challenges specific to making programs run for arbitrary duration on a variety of sensors. In practice, this system provides a number of unique yet unexplored educational opportunities. Early results show promise for Dataflow as a valuable learning technology from research conducted in a high school classroom.

Index Terms—Data-driven inquiry learning, educational technology, Internet of Things, STEM education, sensor systems and applications.

I. PROBEWARE AND THE OPPORTUNITY IN K–12 EDUCATION

COMPUTER-BASED labs and probeware provide a uniquely beneficial component for science education in K–12 classrooms where they offer data-rich inquiry-driven learning opportunities in STEM subjects [1]–[6]. Classroom studies have shown that learners who interact with sensors and tools for physical measurements alongside well-designed curricula are better able to reason about their physical world and learn about complex phenomena [6]–[8]. Sensor-based laboratories closely coupled with physical model building and virtual simulations also help build students’ intuitive ideas and develop deeper engagement with phenomena [8], [9]. Additionally, sensor-based laboratories create opportunities for STEM students to create arguments with data, a critical science

practice and part of the Next Generation Science Standards being adopted across the U.S. [10]. Data argumentation skills and data production in the classroom are generative for learning: data activities in the classroom increase students’ confidence to ask questions, to reason with and interpret data, and to answer their own questions [11], [12]. A student’s ability to produce, reason about, and share data also supports computational thinking skills such as purposeful uses of computational tools for data acquisition, hypothesis generation, data interpretation and analyses, data organization, data visualization, and extraction of meaning [13]–[16].

Even with the many benefits of data-rich learning, many classes still have not incorporated probeware because of the significant cost, training, and technical burden. A single sensor creates a single measurement on a single computer. Many sensors are required for each classroom, and direct access to any physical environment in which data is to be collected. Classroom implementation requires teachers to be fluent with the use and maintenance of the devices and collection system [15]. One solution to source classroom data is to aggregate data across many students in a classroom from purpose-built or repurposed commercial devices [17]. Data files can also be shared more broadly with the community, as exemplified in citizen science sites such as iSense [18] and FieldScope [19]. Teachers who wish to incorporate data analysis without the burden of individual measurement may turn to the many open public scientific data clearinghouses from NOAA, NASA, and others. Reuse of preexisting data recorded by others, however, misses learning in the place and context where data is collected, aspects of the experimental design process, and hands-on learning [20]. The solution described here is an alternative that allows students to design and acquire datasets of their own.

Recent innovations in Internet of Things (IoT) hardware and Internet cloud services provide a promising data-driven inquiry alternative to the one student/one sensor class setup or to data aggregation services [21]. IoT sensors can instrument any object or experiment to communicate position, acceleration, orientation, local magnetic field, temperature, humidity, or frequency of use. Wired or Internet-enabled science classrooms can use IoT-enabled objects to collect any form of numerical, graphical, or photographic data in ways that are increasingly rich and useful for education [22]. The distinguishing

Manuscript received March 6, 2020; revised October 30, 2020 and January 18, 2021; accepted February 10, 2021. Date of publication TBD; date of current version February 10, 2021. This work was supported by the National Science Foundation under Grant DRL-1640054. (Corresponding author: Leslie Bondaryk.)

L. Bondaryk and S. Van Doren are with The Concord Consortium, Concord, MA 01742 USA (e-mail: lbondaryk@concord.org; svandoren@concord.org).

S. Hsi was with The Concord Consortium, Emeryville, CA 94608, USA. She is now with BSCS Science Learning, Colorado Springs, CO 80523, USA (e-mail: shsi@bscs.org).

Digital Object Identifier XXX

characteristic of an IoT educational sensor system is that all data serves all students in an on-demand fashion. A single IoT data stream from an instrumented object can be both shared and accessed by any student in the classroom, at home in a distance learning scenario [23], [24], or on the other side of the globe, if account permissions allow it [25]. Students can collect data in environments that would otherwise be inaccessible, such as on the International Space Station [26] or inside the human body [27]. Teachers and students no longer need to be limited by the particular hardware, resources, or setup of individual classrooms. Today, students at schools with fewer resources can and do share measurements with other classrooms or sensors provided by institutions, greatly reducing the cost per measurement and enhancing student experience [21]–[27].

The volume and extensive nature of rich IoT data, while potentially useful for education, presents a new interface design challenge. Students require systems that encourage them to think computationally about how to design programs and acquire data with appropriate characteristics to construct data stories [28]. In order to successfully run experiments, they must understand concepts like sampling rate, transformation, threshold, and latency, all of which are critical to their ability to correctly interpret IoT data. With IoT systems, it is now both possible and desirable to teach students how to reason about measurements and control their environment with them [17].

We describe an alternative to distributed data collection by using a novel Internet-of-Things (IoT) system called the IoT Dataflow System (“Dataflow”). In the classroom, students can use Dataflow to design their own science experiments, such as the control and monitoring of humidity in a terrarium, or investigation of primary science phenomena such as plant respiration. Dataflow is a solution to the challenges of setting up and maintaining a classroom lab full of traditional, non-networked devices [21], [22], [29] or reusing data from citizen or public data repositories. We aim to support more student autonomy and choice about which data to collect in which context. Simultaneously, we aimed to resolve some of the cost, maintenance, teacher expertise, and interface issues that exist with single-user collection scenarios. This approach has elements of both the constructivist probeware use [1], [30] and the more familiar microcomputer lab approach [2]–[5].

The Concord Consortium iteratively designed and tested the Dataflow system to assess its impact on student learning, interest, and participation [13], [31]. We present the design of our IoT system for use in secondary science classrooms that extends prior probeware systems [31] to enable student-driven laboratories with experimental data acquisition and control. We share the design rationale, implementation, and results from testing the system in a U.S. high school classroom.

II. INTRODUCTION TO THE IOT DATAFLOW SYSTEM

Dataflow consists of a modular hardware kit, a cloud connectivity interface, and a student-friendly dataflow programming interface (see Section IV). The kit includes programmable IoT relays for device control (actuators) and IoT-enabled sensors (CO₂, light, temperature, humidity, and oxygen) that together plug into a Wi-Fi enabled hub (Fig. 1). A single hub can connect up to six components using small cables and ports, making it easy to swap out relays or sensors. A

student’s hub device has a unique-assigned label to differentiate it from others in the classroom. Students access Dataflow in the classroom by logging into an online portal that directs them to

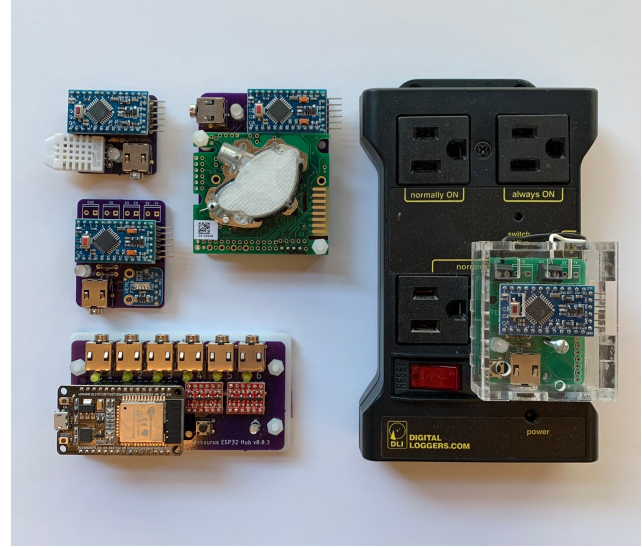


Fig. 1. Sensaurus ESP32Hub shown with a power relay and light, temperature/humidity, CO₂ gas sensors.

the Dataflow workspace where programs can be created. A program in Dataflow is made by dragging and connecting programming nodes into the workspace (Fig. 2). Within a given node, the student uses a pull-down menu to associate and connect a specific IoT hardware device to that node. In this way, students can construct programs that acquire, filter, and record data from sensors, or control relays available on the network.

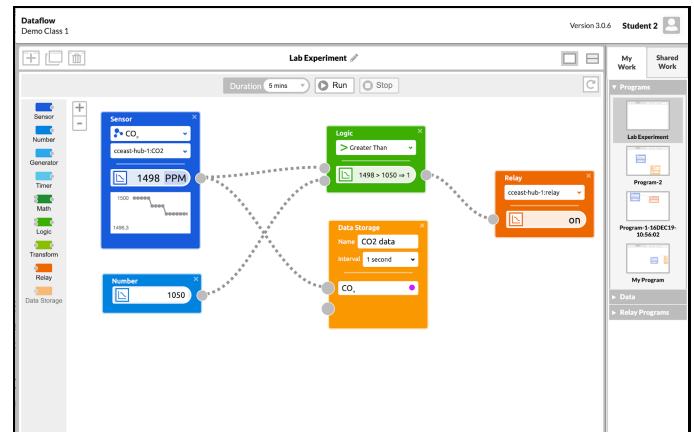


Fig. 2. The IoT Dataflow System with programming nodes for sensing, logic functions, data storage, and relay control.

Once a student pairs a digital sensor node with physical sensor hardware, the numerical values from that sensor immediately appear on the screen. A mini viewgraph of the values enables students to watch for evidence of real-time readings. Students create programs by connecting virtual wires to link nodes. They begin data collection by setting a duration and clicking the “Run” button. Experiments can be as short as one minute long or as long as a month (e.g., for observing plant growth in chambers). Students see all data collected in a graph

that updates in real time. Fig. 3 shows an example of a running three-sensor experiment generating data recordings.

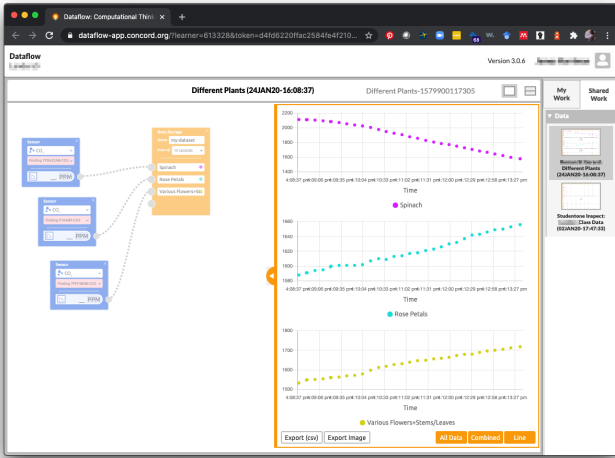


Fig. 3. Dataflow program during an experiment involving data recording from three sensors.

A. Dataflow in the Science Classroom

Dataflow allows students to engage in next generation science practices such as using sensors and data for model development and experimental design. Dataflow activities can incorporate mathematical and computational thinking, and creates opportunities for students to construct arguments from evidence [10]. Dataflow programming creates opportunities for engaging in realistic science, including both data measurement and logically controlled feedback in scientific lab investigations [16], [34].

Dataflow activities also provide teachers with opportunities to discuss calibration, accuracy, and sensitivities of the sensors, sources of error, and the science of sampling statistics. Teachers can further emphasize to students that data are not just theoretical or numerical values, but the product of physical sensors and computational systems that allow for the study of a scientific phenomenon. Classroom activities can incorporate practices of collecting, interpreting, transforming, organizing, and visualizing data. Students are introduced naturally to computing practices and computational performance issues such as latency and discontinuities from dropouts or mechanical failure (often caused by the students themselves).

The “flat data” lab is an example of a high school science curricular activity which integrated the use of Dataflow in a hands-on lab. Students are provided with the prompt, “Can we use computers to keep CO₂ levels in a healthy range?”. At the beginning of class periods, student groups decide what hardware they will need, the length of their experiments, the frequency they will record sensor readings, and whether they will manipulate conditions computationally or physically. Some students may choose to monitor CO₂ levels in a plant chamber while others may monitor CO₂ levels generated by students’ collective respiration in their classroom. Once students develop a tentative plan for their investigation and gather hardware, they log into Dataflow and begin building a program to record data to support their model.

When interacting with Dataflow for the first time, many student groups bring unrelated nodes into the workspace seemingly at random. They create and delete nodes until they gain a sense of Dataflow’s functionality and workflow. Eventually, they construct complete systems of connected sensors and can begin collecting and viewing data on the sensor blocks in recorded datasets. When students are viewing real-time data, they observe and interpret data trends, gaps in the data recording, or sudden peaks in the viewgraph. When returning to Dataflow in subsequent class periods, students prefer to modify or debug existing programs to fit their needs for that day’s investigations. As experiments evolve, students may have additional questions about previously produced datasets that encourage them to add number, logic, or additional sensor types to support sensemaking from the CO₂ data. The ease of modifying existing programs and tinkering with the hardware components gives students the ability to rapidly iterate on their strategies.

III. IOT CLASSROOM DESIGN CONSTRAINTS AND AFFORDANCES

A. Sensor Ubiquity: Distributed, Robust, Multi-user Sensors

Our goal for introducing IoT into classrooms was motivated by the promise of distributed, collaborative data sharing across individuals and classes, along with opportunities for computational thinking afforded by student design of experiments and experimental data. Because data from any given sensor can be broadcast and made public over the Internet, a given teacher with a single IoT sensor experiment in a resource-constrained classroom could share her data stream and virtually loan or borrow access to IoT hardware setups in other classrooms or institutions. Students also need to be able to construct an experiment using data from many sensors distributed across the classroom (e.g., monitor CO₂ levels over time near room exits and windows) and acquire data over meaningful time periods from minutes to days or weeks, depending on the application. The desire for spatially and temporally distributed data means that experiments should be unbounded by the number of sensors present in a given classroom, the time limit of a single class period, or the physical location of the sensor. Students needed straightforward ways to identify data sources from this larger pool, aggregate data from multiple sensors, and report about it.

B. Acquisition Clarity: Computational User Interface for Device Control

Dataflow enables students to manipulate data to control and analyze experiments, learning computational concepts (e.g., sequencing, stabilization, debugging, cycling, feedback) as part of the same interface they use for data acquisition. It is also possible to actuate a relay-controlled device, such as a lamp or fan, from within the same environment. Dataflow software user interface is designed as a browser-based visual dataflow programming language [35]. Research has shown dataflow languages to be a natural way to think about programming, and are productive in educational environments [36]–[39] and it is the format of language used in Amazon Web Services (AWS) IoT services programming; we have chosen a visual dataflow diagrammatic approach in an effort to invite and bootstrap

novice programmers into making Dataflow programs while reducing the need for mental translation of logic applied to sequences of values for students. The application also enables the user to record data, view and compare archived datasets as time-series graphs, and export datasets for further analyses.

C. Data Transparency: Teacher Monitoring and Student Sharing

Cloud storage, execution, and synchronization of sensor data and programs enables a variety of cross-user inspection scenarios for teachers and students. Real-time file and data synchronization with the server allow teachers to view student-written scripts that operate IoT devices while the students are working on them or asynchronously during classroom preparation. Teachers can monitor science experiments across all local and remote classrooms from an online dashboard. Traces of user interactions or data about learner usage can be captured for assessment purposes. Students are able to share their computational approaches and the resulting datasets with their instructor and their peers to create a community data lab in the classroom.

D. Classroom Reliability and Security

In addition to pedagogical affordances, IoT in the classroom needs to meet or exceed requirements for reliability, security, affordability, scalability, and performance for commercial IoT applications, given the relative lack of technical support or reliable Internet connectivity in the average school. The typical high school classroom presents a number of challenges for reliable implementation of an IoT setup. Wi-Fi firewalls often require special setup to enable access and can suffer inconsistency or slowness. Slow networks or unconditioned power can make sampling rates and sensor connections unreliable in reality or by perception. Delays in data recording or display, dropouts, failed sensor readings, or just incorrect programming logic (user error) may be perceived as malfunctions by students who are unfamiliar with the realities of hardware speed and connectivity. They need to be able to “open up the hood” in an accessible way that will keep them experimenting rather than giving up in frustration.

Feedback mechanisms in the form of live updating values and minigraphs have been built into the dataflow UI to help students orient to the physical sensors and debug their programs and sensors and to clarify the interpretation of data. Dataflow software uses a variety of automatic correction methods when connecting to devices to catch situations which may stall either the students or the teacher from proceeding in standard operation. We also designed administrative access so the teacher or the software team can terminate running programs. This facility also allowed software team members to evaluate live classroom scenarios and repair them remotely if needed.

Finally, any scenario which allows students to record measurements of their own design has the opportunity to reveal private information, however far removed a string of numbers might seem from the data producer. We designed Dataflow to have appropriate authentication not only for user access to recorded files and programs, but also with respect to access to the devices and IoT data streams. Only authorized users can access the data in its raw form or take readings from remote devices.

IV. DATAFLOW SYSTEM ARCHITECTURE AND IMPLEMENTATION

The Dataflow system’s physical hardware kit is a microprocessor-based hub and sensors, an AWS cloud-based system for collecting sensor values and running/storing student programs, and a browser-based flow programming environment and document management system. The components together allow students to sample streams of sensor data and write programs to manipulate and record data or use data to control actuating devices. We discuss the design of the system components along with justifications for design decisions that were focused on successful K–12 classroom implementations.

A. IoT Device Hub and Sensors

Sensors in the kit are plugged into a custom-built ESP32 chip-based hub, the central board which houses all the sensor connection points, power, and communication hardware/software. The sensors and hub were designed for minimal cost and ease of connection to Wi-Fi. They can be programmed with school authentication credentials so they can work on a secure school network. The microprocessor-based hub runs custom Python code called Sensaurus [40]. The onboard code is responsible for recognizing when individual sensors are plugged or unplugged, and communicating the device ID, type, signal and physical measurement units to the central AWS IoT Server. Hubs also listen for and respond to messages, which can control relays and adjust sampling rates, while the software execution of student-written programs is handled by the cloud server.

Communications with the hub use the Message Queue Telemetry Transport (MQTT) protocol. This is one of several standard IoT communication protocols, selected because of its ubiquity, and its ability to authenticate access to data, which is critical in school systems. Hubs are secured using cryptographic keys to authenticate and encrypt the communication with the MQTT broker.

Each hub has audio jacks for sensor and relay connections as well as power (Fig. 3). Audio jacks were chosen because of their lack of directionality and relative robustness. In class testing, they were found to be the most trouble-free in terms of getting sensor plugs seated correctly. They can (and did) withstand repeated plugging and unplugging of sensors as students learned their way around the hardware setup. The current suite of sensors can measure CO₂, O₂, temperature, humidity, lux, soil moisture, and particulate matter.

School affordability also drove successive iterations [30] of hardware kits to use less expensive microprocessor hardware. The ESP32 chips used cost only a few dollars each, and provide the necessary Wi-Fi and communications capability without the expense of a full computational chip and memory. We have instead incurred the expense of cloud computing and distributed device management, which is cheaper than individual computational edge systems with more expensive sensors. Similarly, tradeoffs were made to keep the cost of sensors low, especially the CO₂ sensor components, which had limited operating ranges (<10,000 ppm) to be affordable, yet still useful for school laboratory experiments.

B. Dataflow Visual Programming Interface

The IoT Dataflow System interface uses a connected-node diagramming convention along with synchronization features built into both local (in the browser) and cloud (on the server) execution to support and extend student understanding of data [35]. Users write programs using connected-node diagramming conventions with nodes and connectors similar to visual programming interfaces in a dataflow context that have been recommended broadly for professional use in this context [36], [37]. Data flows through nodes and wires from left to right in the user interface. As values progress through sensor, timer, math, logic, transform, generator, relay, and/or data storage nodes, the value of the data at that node is displayed in real time as soon as inputs to the node are present. Directed-node dataflow diagrams allow easy communication of ideas for novices [38] and have been successful in educational environments [38]. This metaphor was adopted for the system and is reflected in the naming of the software: IoT Dataflow System.

Students learn about computation and control systems as they create programs by live inspection of sensor data as it flows through the nodes. The moment a student selects a particular sensor or connects a downstream operation node they will see numerical results. We call this instantaneous execution “tinker mode” because it allows students to play and test their understanding in a constructive and low-stakes manner comparable to tinkering with physical systems. Each visible node includes a large updating numerical current value display, an equation display if a mathematical operation is performed, and a scrolling minigraph that shows the most recent 15 values in the sequence (Fig. 4). These values all update in real time as soon as inputs are available from a sensor or an upstream node. Tinker mode and the inherent model diagram nature of the dataflow interface allow students to reason about the stream of values that propagate across the network.

Tinker mode is enabled by browser-based evaluation that processes the same program as the server-side execution described below. Student programs execute on the client (student’s local computer) until the student chooses to run the program. Tinker mode’s live updating equations and minigraphs serve as valuable debugging tools for students, emphasizing sampling and the evolution of sensor and program results over time.

The student action of running programs is needed to record data or trigger relays in control scenarios. Once the Run button is pressed, the state of the program and the list of connected devices are frozen, and a new “data” file is created to store that program specification and the resulting recorded data values. The program instance is sent to execute on the server until it completes or is stopped manually. If a data storage node is used, the student sees a live updating graph, showing the requested data traces as they evolve, alongside the program used to produce them. The student can now close this file and begin a new file, return to their tinkering and run a different scenario, or close their session entirely; running programs will continue to operate in the background.

Previous versions of Dataflow executed programs on Raspberry Pi hubs [30], restricting the sensors on that hub to a single program and student at a time. This significantly reduced

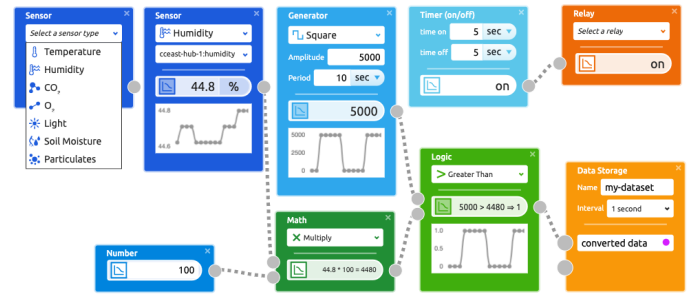


Fig. 4. A close-up view of Dataflow node functions including nodes for sensors, math operations, logic operations, simulated data generators, data transform, relays, and data storage.

classroom capacity and made it impossible for remote users to access ongoing experiments.

C. IoT Server/Hub/Program Communications

To enable the ubiquitous and multi-user IoT scenario desired for classrooms, the IoT Dataflow system consists of a set of sensors all continuously reporting values to an Amazon Web Services (AWS) IoT Core server [41]. AWS IoT Rules save the sensor values in AWS IoT Device Shadows, which are software replicas of the data stream from hardware. Shadows allow the current values of the sensors to be accessed without waiting for the next message from the sensor hub. This allows multiple programs to access a single sensor’s value without all individually communicating with the hub. It behaves like a data “party line,” where anyone with access may listen to the data stream coming from the shadow. Student-created Dataflow programs access sensor data streams through these shadow devices, so any number of programs can access the same device at the same time without colliding. This setup also reduces overall throughput for hubs, making it a scalable solution for large numbers of sensors all reporting around the world.

Cloud-based program execution is orchestrated through AWS Lambda functions communicating with the device shadows and stored in the DynamoDB database (Fig. 5). If the program has a data storage block, the data is also stored in DynamoDB. When applied to reactive systems such as sensor measurement and control, Dataflow programming has a built-in notion of data frequency, such that each measurement or calculation at each node is synchronized with all other program actions on each data point. We created a universal timer to synchronize the data readings, the program calculations at each node, and the data values displayed to the student. During every program cycle, Lambda functions read the sensor values from the Device Shadows, execute the program, and then send MQTT messages to trigger relays. In this way, data readings, recordings, calculations, and display move in lock step together.

D. File Management, Sharing, and Authorization

All software used in classrooms requires security to prevent outsiders from viewing student data, (e.g., images, geographic location, or other identifying data). The Dataflow user interface is accessed through Concord Consortium’s STEM Resource Finder, which is used to register users and provides the scope of allowable sensor and data viewing and sharing. Students and teachers can create and retrieve files associated with a user account and publish data and programs securely for sharing with others in their class. The authentication system allows us

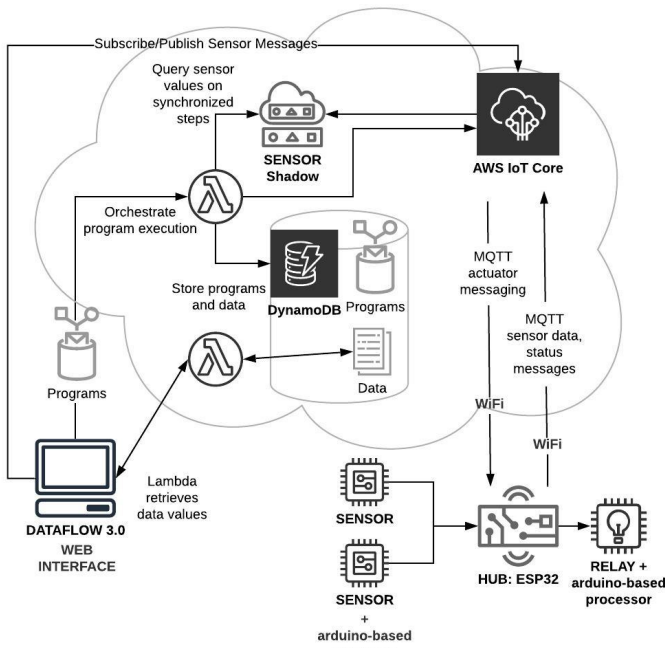


Fig. 5. Dataflow IoT network architecture and message passing.

to scope programs, data, and devices to the class, teacher, or institution, establishing circles of trust for the work in the system. Multiple students can perform experiments with the same sensor within their scope of authorization, and can access sensors in remote locations as long as they have permission to use those sensors.

The AWS IoT Server includes built-in authentication in their MQTT library, which requires passing a secret key for every device access. The software will automatically limit access to the institution associated with the instructor. It is possible for software administrators at the Concord Consortium to share devices more broadly so that citizen science-style experiments and analysis can take place.

The IoT Dataflow System browser application uses Google's Firebase database to provide real-time authenticated file management. Firebase connects to the IoT server's DynamoDB to queue programs to be run on the server, and to display recorded data back in the browser. The browser application also connects to the MQTT broker to monitor the real-time (shadowed) sensor values. This architecture makes it possible for multiple users to monitor the ongoing, updated state of any document. This WYSIWIS architecture allows a sidebar dashboard full of live updating views for the student so they can monitor multiple experiments at once. The document state saving, logging, and storage mechanisms also enable instructor and administrator monitoring, performance analysis, and diagnostics. An administrator uses a control panel dashboard to monitor hardware in use and live views of student work. This feature was especially important during the initial rollout of the system in the classroom to monitor and debug problems with data saving and restoration as students worked with sensor systems in unanticipated ways.

E. Hardware Dropout and Synchronization Error Handling

As described above, Dataflow programs in tinker mode display their values live so the student can see if they are

producing expected results, including display of sensor values so students can check both programs and hardware connections. In the course of normal operation, a sensor may, for reasons of interface or connectivity fail to report some values. If a few are missing, the overall data should still make sense. If a large series of values are missing the data become meaningless or at least misleading. The length of time we chose as a distinction between these cases was one minute. Once programs are debugged and the student formally runs them to record data, program blocks report when sensors have been offline for more than one minute, and performs this check on any running programs when a session is restored. These checks require orchestration between the running program, the IoT shadow, and the MQTT message stream from the devices. Messages include the status of each connector on a hub and the ID of devices plugged into each hub, along with a timestamp. This allows polling of devices associated with a program to tell whether data is being sent. In this way, overnight runs and other out-of-view programs can be highlighted when the student needs to adjust faulty hardware. Because the data is no longer being collected, these will not impact the data, but are instead a record of the sensor used, and a note that a new one must be chosen to run the program again.

In the more benign case of data dropouts under a minute, the server reports back simply that there is no value for that synchronized measurement. These are reflected as missing points, which are not connected to adjacent points in graphs. In a similar scenario, data storage and the start time on Dataflow graphs begin at the time the Run button is pushed. If for reasons of Wi-Fi or hardware delay, the first recorded point does not occur at the start of the program time, the actual time of collection on the graph is stored.

V. TECHNICAL CONSIDERATIONS FROM CLASSROOM TESTING

The progressive and iterative design of an IoT system for K–12 classrooms presented unique design challenges, which required not only meeting pedagogical requirements, but also engineering a system to meet the practical needs and constraints of K–12 classrooms. Here, we discuss these implementation challenges faced by educational IoT systems based on lessons learned during class testing.

A. Unexpected Tinkering

In addition to the known issues of classroom IT support, we encountered another source of system instability: the students themselves. Many students inadvertently or intentionally disconnected hardware as they learned about how the system worked. This created scenarios in which data was seemingly acquired but was not recorded. Server-side programs were designed to maintain records of the last known hub for any given sensor ID so that they could record over long periods when students were not logged in. We encountered situations in which students unplugged sensors that were in use and moved them to a different hub. We now manage this scenario so that programs that are recording data from these sensors can continue to do so, and will gracefully register the period during which the sensor was offline without failing outright.

We introduced numerous points in the Dataflow code to retry connections across the hardware messaging and server.

Dataflow can now reallocate sensor IDs to different hubs, refresh available hardware lists, and offer administrative access to end running programs. Checks run continuously to determine whether to handle the dropout silently or to warn the student depending on the duration of the dropouts.

As we analyzed the student interactions in the classroom, we elected to prevent students from beginning a run with failed sensors or relays, or when relays are already in use by another program. Once a program has started, we alert the student when a device fails, but do not end the recording until instructed, leaving that choice to the user. Because experiments can be running on the server while the student's browser session is closed, we also return alerts to the student in their list of ongoing data acquisition programs when they next open Dataflow.

B. Data Size, Sampling Rates, and Cost

One of the primary concerns in IoT technology is the cost to transmit and store the data. Data quickly become large at scale, and hence expensive. The class tests of this system allowed us to understand the realistic data usage in a several-week classroom setting. This encouraged us to create limits in the program settings relating the length of time a student records data to the minimum sampling rate. Programs that are run for 10 minutes or less can be sampled every second or two, while those that run for a month can be sampled no more frequently than once a minute. We are not typically concerned with under-sampling in these school scenarios because the biological phenomena in experiments changes slowly on the order of minutes or hours rather than in seconds. If different kinds of measurements were desired by an expanded curriculum, we would have to consider overrides for sampling restrictions.

C. Scalability

If any such IoT system is to become successful at scale, even in a single school district that serves multiple schools, it needs to accommodate large numbers of users. Many thousands of students could simultaneously read hundreds of thousands of data points from shadow sensors, address thousands of relays, and execute thousands of calculations in online programs. Our initial implementation of the Dataflow system was not capable of this kind of scale, but has since been built on top of the scalable and secure commercial AWS system and MQTT protocol that make data exchange and server redundancy possible.

Another issue of scalability has to do with the generally low-level client-side computing capability of school laptops. Chromebooks are ubiquitous in secondary schools, though older models often have difficulty with large client-side applications. The more computation we can move to the server and the less chatter we can communicate across the network, the better Dataflow will perform. We have already done some tuning in this respect by propagating only messages that affect the user's current view to the client, but more work could be done to reduce the client computing and rendering burden.

VI. DESIGN-BASED RESEARCH STUDY IN BIOLOGY

IoT Dataflow system was iteratively tested in design-based research studies in four cycles of testing between 2017 and 2020. In each iteration, the Dataflow user interface and

accompanying curricular activities were improved and revised. Here, we report on a study with one teacher from the final test and implementation of the IoT Dataflow system with high school students in an urban classroom in Northern California. This school served a larger percentage of non-White students (33% African American, 39% Latinx, 16% Asian American, 7% White, and other races) and a majority who qualified for free and reduced lunch (73%). Four ninth grade high school biology classes used the IoT Dataflow system in a sequence of lab activities taught by one early career teacher familiar with technology for classroom teaching. The broader project sought to answer questions about students' content learning of biology, interest in science, and computational data practices. This study aimed to understand if this version of Dataflow IoT system with a cloud-based implementation could support students in successfully carrying out the planned curricular activities. The results reported here focus on the following questions: 1) What kind of programs did students create and how many programs and datasets did students create? 2) How did the system perform with multiple simultaneous users of the IoT Dataflow system in the classroom? 3) How did students find the curricular activities with Dataflow? What aspects lead to positive or negative affectivity? 4) To what extent did students engage in computational data practices as they employed the IoT Dataflow system?

A. Curriculum and Lab Activities

The three-week curriculum unit focused on CO₂ in human and plant systems that integrated Dataflow in a high school biology class. Students used a storyline-based curriculum [32] about how and where CO₂ is generated, studied photosynthesis and plant respiration, and measured the changes in CO₂ in a closed chamber containing spinach leaves during light and dark conditions [31]. In each unit, students investigated a driving question such as "What does light have to do with plants and CO₂?" and "Why do I breathe out CO₂?" Any time people or animals are in a closed space, they produce carbon dioxide gas and over time, the levels can become unhealthy. Students figure out that they can use light to control the process of photosynthesis. Students were also given the challenge to stabilize CO₂ levels within a healthy range (400-1,000 ppm). The lab activities asked students in their groups of 2-3 to develop a data collection strategy that suited their experimental design preferences. Some students produced one long dataset where they changed conditions during the experiment, others produced smaller individual datasets, one for each trial condition. Students groups were provided with one set of hardware (CO₂ gas, light temperature, humidity sensors, cables, hub controller), as well as access to a range of lab materials to choose from which to stabilize CO₂ (food containers, plastic wrap, rubber bands, tape, spinach leaves, wax paper, lamps, etc.). During and after each lab, students shared their data with the class, and reflected on what worked and what did not in a whole class discussion and in writing.

B. Methods and Instruments

1) *Dataflow system performance*: System performance was studied via remote system monitoring and direct observations by researchers in the classroom. In each class period, twelve

hubs each with multiple sensors were used by 20 to 25 students to actuate physical devices using Dataflow programs.

2) *Usage data:* We gathered statistics around the number of programs per student and complexity of programs (number of program blocks) by collating the stored student program documents from the program's server storage. Because programs are stored as a series of blocks and metadata for playback, we were able to write a script to extract this structured information to count the number of blocks and connections.

3) *Student experiences and interests:* A post-experience survey was administered online via SurveyMonkey at the end of a three-week curricular unit. Eighty percent of the students (92 out of 115) students took the survey, which consisted of 36 questions covering topics such as student interests, attitudes, experience with the curriculum, and self-reported curriculum outcomes. Students were asked to rate statements such as "I liked using and learning about the electronics (sensors, hubs, relays)," "I liked making and running programs to collect data," and "My data reflected my own choices and decisions" on a five-point Likert scale. Students were also asked to report on their interests in particular activities: "In general in your life, how much do you enjoy or are interested in the following activities?" as well as open-ended questions such as "What parts over the last few weeks did you like the best? Least? Please explain."

4) *Screencast video sampling for computational data practices:* Screencasting software (Screencast-o-Matic) installed on student laptops captured interactions and video of students working in groups of 2-3. Data were recorded from a subset of consented students (12 focal groups) across four classes. Altogether 61 class sessions (63 hours) of screencast video was collected. Using a prior taxonomy for computational thinking in STEM [16], instances of computational data practices were tagged by the specific data practice afforded by Dataflow: collection, transformation, analysis/ interpretation, control, organization, and visualization. Each video was coded using a rubric to identify computational data practices (Table II). One third of the data, randomly selected from the video sessions, was coded independently by two researchers into six major categories and 17 subcategories of data practices. In these analyses, we focus specifically on whether students engaged in the expected computational data practices when using Dataflow in lab activities.

VII. RESULTS FROM CLASSROOM DATAFLOW TESTING

A. Results from Usage Data

Documents from the Dataflow program's server storage were tabulated (Table I). Forty four student groups successfully produced 129 editable programs and executed them a total of 646 times using Dataflow. These generated 359 stored program runs containing a relay node and 559 stored program runs that produced a dataset (a number of the total program runs produce a record of a run containing both a relay node and a program producing a dataset). Altogether, students created 775 documents (programs and record of program run). The ratio of programs to program runs (1:5) indicates programs were run multiple times while students iterated and tinkered. The repetitive refinement of computations is one of the established metrics to evaluate computational thinking [28].

TABLE I
OVERVIEW OF STUDENT USAGE OF DATAFLOW DURING THREE-WEEK CURRICULUM UNIT

	Documents	Editable Programs	Program Runs	Program Runs Using a Relay	Program Runs Producing Dataset
Total	775	129	646	359	559
Mean	17.61	2.93	14.68	8.16	12.70
SD	10.41	2.33	9.60	8.09	8.13

B. Results from Dataflow System Performance

In the first week of the described classroom tests of Dataflow, the newly implemented IoT server architecture revealed some performance and sensor identity issues. These issues included a lag in the system response, blank datasets, and unresponsive sensors. These kinds of issues are more critical in the classroom or with any group of novice users because they lack the skills or the class time to overcome imperfect infrastructure. Software team members were able to monitor and debug activity in classes remotely and in a large degree of detail. Student behavior revealed a new test case, which was critical for system design: students were unplugging and replugging the same sensor into different hubs, often randomly and while some running programs had already begun recording from the sensor as part of a previous hub. The interplay between this behavior and system checks for unresponsive hardware caused data loss. This led to an important refinement of warning/offline rules and handling of changed sensor addresses. We also optimized the number of messages from simultaneous running programs, which we chose to synchronize both on the student's display and on the server to avoid overflowing memory in both cases. In all of these scenarios, the ability of the software engineering team to remotely monitor what sensor and program values were produced was critical to correcting the issues in time to ensure successful class testing.

Once software and hardware issues were addressed, students were able to conduct experiments and encounter challenges that mirror authentic practices of scientists and engineers. Students generated many datasets requiring them to name their files for later retrieval, distinguish previously run program versions from the editable program, conduct multiple trials through tinkering with the hardware setups, and troubleshoot a computational system.

C. Results from Student Experiences and Interest Survey

When asked if they agree with the statement "I like using the online software, Dataflow" on a five-point likert scale, 43% of the students said they agreed or agreed strongly compared to 24% who disagreed or disagreed strongly (while 34% remained neutral). Similarly, students reported they liked making and running programs to collect data (44% agree; 23% disagree), as well as creating programs to turn on/off relays (45% agree; 23% disagree). Of most interest were student responses to learning about how data was created. This yielded the fewest disagreements (41% agree; 11% disagree). This is viewed as a favorable result by the teacher given that many of the students had difficulty engaging in school and in science class before

this activity was introduced. The teacher also commented that students who were typically disengaged in science class were participating more actively in the activities involving programming Dataflow and setting up experiments.

Students elaborated their feedback with open-ended written responses that included 68 positive comments and 67 negative comments. The following student quotes are representative of the set of positive comments:

- “I liked using the computers and electronics (CO₂ and oxygen thing) to see when the CO₂ levels became unhealthy.”
- “I like hooking the sensors and see how they collect data.”
- “Doing hands on experiments and working with my team.”
- “Doing the lab where we used sensors to see the CO₂ levels in the classroom or in spinach.”
- “I really enjoyed watching a program make data based on information the sensors recorded.”
- “My favorite part of last week’s activities was designing our own way to make the CO₂ levels stay stabilised.”

The negative comments fell into broad categories with some common trends. Thirty-one percent of the comments were related to technology; 15% complained about the amount of writing needed with the packet of worksheets; 9% described feelings of boredom; 6% were about not liking group work; and 39% were a wide range of general dislikes ranging from school to specific aspects of their labs. Some comments reflect some early issues with lag time and issues with relay devices. These are representative negative comments:

- “I disliked all the writing we had to do.”
- “The group work. I work better by myself.”
- “The experiments have no interest for me.”
- “I liked most of it but the thing that I liked least was probably when our computer program was laggy and we couldn’t do anything about it.”
- “I didn’t necessarily enjoy creating and running data sets especially with relays because they were more problematic and I didn’t feel I had enough class time to work through all the problems presented.”

While not representative, some students expressed a desire to exercise more creativity in their experiments such as ways to generate more CO₂ in the chambers by lighting a match to change conditions or wished for more time to explore things they noticed. For example, one student noted disappointment in “not [being] able to do something better like putting fire to see how much CO₂ are we going to get.” Students and teacher both valued having more time to tinker and use the tools than the normal classroom period would allow.

D. Results from Screencast Sampling

Screencast data indicated that Dataflow supported students in engaging in computational data practices. Interrater agreement reached 85.7% between two independent coders across major categories (collecting, transforming, interpreting, controlling, organizing, visualizing) and 72.2% in subcategories while observing a sample of focal student screencasts (Table II).

TABLE II
PROPORTION OF COMPUTATIONAL DATA PRACTICES ENGAGED BY FOCAL STUDENTS USING DATAFLOW DURING A LESSON WITHIN THE CURRICULAR UNIT SAMPLE

Major Categories	Subcategories of data practices	%
Collecting data	a. started a run b. made a data file	76.4%
Transforming data	a. applied a logic or math function b. changed shape of the data curve	71.4%
Interpreting data	a. discussed their data b. asked a question c. puzzled about measurement d. sought an answer	71.4%
Controlling data	a. used a relay node b. used program logic c. change conditions to control a relay	61.9%
Organizing data	a. named experiment runs b. created custom name when saving	57.4%
Visualizing data	a. viewed data graph on sensor node b. viewed data graph during data run c. retrieved data and inspected visually d. changed view type on a graph	71.4%

Consistent with the usage data, all student groups were successful in creating programs that collected data for their planned experiments at some point across the entire curricular unit, yet within the focal student sample data collection practices were taken up 76.6% of the time. Fewer student groups (71.4%) applied transformations (although this was not unexpected as students could have used external materials to manipulate data). There was a similar proportion of students who engaged in data interpretation with the data they produced; they discussed the data outputs, anomalies, and trends with their lab partners (71.4%). Using the features provided in the IoT Dataflow System, students also often visualized data during experiments (71.4%). Few students used the viewgraphs on the sensor nodes (28%), but instead chose to visualize data streams over a whole experiment. When students used a data storage node or started a new experiment, a majority provided custom names or names for experiments (57.4%) while the remainder of students simply used the automatic file naming feature. A majority of student groups used Dataflow’s features as designed to engage in computational data practices.

VIII. DISCUSSION AND IMPLICATIONS FOR FUTURE WORK

Our implementation of Dataflow balanced tradeoffs to develop a reliable, secure learning application for the high school classroom use with many pedagogical affordances for both science practices and computational data practices. The system exposes students to working with hardware and software control in a cost-effective manner as it allows as many students to acquire data streams from the same device. Students can perform open-ended experiments of an unrestricted and interleaved duration. Design iteration and student testing allowed us to refine a variety of sensor and system checks to

reliably record student data throughout multiple program runs, sessions, and time frames. Dataflow document and sensor authentication created a secure environment for sharing that can be scoped to the individual, group, or class, while still offering the future possibility of public sensors. The system protects student data privacy and allows users to make choices about how and with whom their data is shared. Student and teacher comments reflected positive reception of Dataflow to support lab activities. The data and programs produced reflected productive computational thinking.

Teachers and students who tested the IoT Dataflow System using a design curriculum showed that such a system resulted in successful use of computational tools in science class, even if students interest in engaging with data varied. Students could reliably engage in computational data practices with a Dataflow interface that enabled them to create programs to collect, transform, interpret, organize, and visualize data. While students' comments were mixed regarding their preferences for different aspects of the activities such as writing, working in groups, and the length of labs, the teacher and the students demonstrated they could use Dataflow's features to complete the planned activities.

Future research and implementation will continue to explore configurations to optimize program execution and data acquisition. We can explore running all programs in the cloud or with local hubs, and balancing program execution between these two modalities.

While the Dataflow application was designed for the secondary classroom, future applications can broaden learning scenarios to include out-of-school environments. Future classrooms that engage in community science and global lab science investigations may need to handle over 100,000 simultaneous users. Scalability of these kinds of citizen science activities would need reliable services with server redundancy. Students as community scientists can collectively monitor, view, and aggregate real-time data gathered from, for example, air quality, temperature and humidity across different environments and geographic locations [19]. These kinds of experiments emphasize both the benefits of working in distributed teams, and the importance of IoT technologies to enable them.

The IoT hardware developed in the Dataflow System to date is a research prototype that will be further developed into an open-source hardware product. To ensure the design and robustness of the hardware, questions exist regarding the lifetime and durability of IoT hardware with long-term use in schools, the replacement rate, and the cost over a five-year period. Unresolved issues and future designs could examine ways to integrate and support existing off-the-shelf home automation devices like relays and smart bulbs. Slowly changing scientific phenomena such as indoor farms, decomposition, and closed chamber biospheres require controlled environmental conditions which can be instrumented with Dataflow actuated devices. These scenarios would allow students to participate in long-term environmental monitoring to learn additional computational concepts (e.g., stabilization, cycling, data control-feedback). Multiple data sources over long timescales periodically sampled (particularly, over an Internet connection) may reveal new discoveries in conceptual understanding of the system and the data made possible by

timescales beyond a single class period afforded by IoT. Robust long-term operation will be key to these scenarios. To support such scenarios, expanding the use of IoT from a single classroom to multiple classrooms will need to further address and mitigate privacy concerns especially when actuated devices involve the use of cameras or other devices that control and cross social boundaries from private and public learning environments.

STEM classrooms will be expected to promote ways for students to learn with large and messy datasets to solve problems and generate insights about the world. Being immersed in the "data deluge" within science classrooms should become an integral part of science learning. Data and their analysis are already part of the Next Generation Science Standards (NGSS) currently being adopted by many states. We need to examine the interactions of students with Dataflow to better understand how students can meaningfully learn to extract and analyze data in classes with other domains (e.g. physics, chemistry, Earth science). Bypassing manual or single-source collection allows students to aggregate data from multiple sensors of the same type; the variation in sensor values even for sensors of the same type can become a data science teaching opportunity: a place to discuss calibration, accuracies and sensitivities of the sensors, sources of error, and the science of sampling statistics. Gracefully integrating these topics in science courses whose focus is not solely statistics will require careful orchestration.

Additional research in software interface and curriculum is needed. We expect to expand the data analysis, annotation, and export facilities of the Dataflow software in future versions as we learn more about how students need to manipulate data to support their own experimental designs.

ACKNOWLEDGMENTS

We acknowledge Peter Sand at Manylabs for Sensaurus hardware kits and team members at the Concord Consortium who have contributed to the project: Scott Cytacki, Colin Dixon, Lisa Hardy, Thomas Farmer, Sam Fentress, Sarah Haavind, Christine Hart, Evangeline Ireland, Matthew Lewandowski, Kirk Swenson, Michael Tirenin, and Charles Xie. We also thank the teachers and students who participated in the formative testing and studies. We thank Cynthia McIntyre and Lisa Buoncuore for their editorial assistance.

REFERENCES

- [1] R. F. Tinker and S. Papert, "Tools for science education," in *AETS Yearbook: Information Technology and Science Education*, J. D. Ellis, Ed., Columbus, OH, USA: ERIC Clearinghouse for Science, Mathematics, and Environmental Education, 1988, pp. 1–24. [Online]. Available: <https://files.eric.ed.gov/fulltext/ED307114.pdf>
- [2] C. Staudt, "Curriculum design principles for using probeware in a project-based learning setting: Learning science in context," in *Portable Technologies: Science Learning in Context*, R. F. Tinker and J. S. Krajcik, Eds., New York, NY, USA: Springer, 2001, ch. 4, pp. 87–102, doi: 10.1007/978-94-010-0638-5_5.
- [3] R. Tinker and J. Krajcik, Eds. *Portable Technologies: Science Learning in Context*, New York, NY: Springer, 2001, doi: 10.1007/978-94-010-0638-5.
- [4] D. Russell, K. Lucas, and C. McRobbie, "The role of the microcomputer-based laboratory display in supporting the construction of new understandings in kinematics," *Res. Sci. Educ.*, vol. 33, no. 2, pp. 217–243, Jun. 2003, doi: 10.1023/A:1025073410522.

- [5] S. J. Metcalf and R. F. Tinker, "Probeware and handhelds in elementary and middle school science," *J. Sci. Educ. & Technol.*, vol. 13, no. 1, pp. 43–49, Mar. 2004, doi: 10.1023/B:JOST.0000019637.22473.02.
- [6] A. A. Zucker, R. Tinker, C. Staudt, A. Mansfield, and S. Metcalf, "Learning science in grades 3–8 using probeware and computers: Findings from the TEEMSS II project," *J. Sci. Educ. & Technol.*, vol. 17, no. 1, pp. 42–48, Feb. 2008, doi: 10.1007/s10956-007-9086-y.
- [7] J. R. Mokros and R. F. Tinker, "The impact of microcomputer-based labs on children's ability to interpret graphs," *J. Res. Sci. Teaching*, vol. 24, no. 4, pp. 369–383, Apr. 1987, doi:10.1002/tea.3660240408.
- [8] J. L. Chiu, C. J. DeJaegher, and J. Chao, "The effects of augmented virtual science laboratories on middle school students' understanding of gas properties," *Comput. & Educ.*, vol. 85, pp. 59–73, Jul. 2015, doi: 10.1016/j.compedu.2015.02.007.
- [9] P. Blikstein, T. Fuhrmann, D. Greene, and S. Salehi, "Bifocal modeling: Mixing real and virtual labs for advanced science learning," in *Proc. 11th Int. Conf. Interaction Design and Children (IDC '12)*, Bremen, Germany, Jun. 2012, pp. 296–299, doi: 10.1145/2307096.2307150.
- [10] National Research Council, *Next Generation Science Standards: For States, By States*. Washington, DC, USA: The National Academies Press, 2013, doi: 10.17226/18290.
- [11] T. Erickson, M. Wilkerson, M., W. Finzer, and F. Reichsman, "Data moves," *Technol. Innov. Statist. Educ.*, vol. 12, no. 1, 2019. [Online]. Available: <https://escholarship.org/uc/item/0mg8m7g6>
- [12] R. Lehrer, M.-j. Kim, and L. Schauble, "Supporting the development of conceptions of statistics by engaging students in measuring and modeling variability," *Int. J. Comput. Math. Learn.*, vol. 12, no. 3, pp. 195–216, Dec. 2007, doi: 10.1007/s10758-007-9122-2.
- [13] L. Hardy, C. Dixon, and S. Hsi, "From data collectors to data producers: Shifting students' relationship to data," *J. Learn. Sci.*, vol. 29, no. 1, pp. 104–126, Nov. 2019, doi: 10.1080/10508406.2019.1678164.
- [14] V. R. Lee and M. H. Wilkerson, "Data use by middle and secondary students in the digital age: A status report and future prospects," commissioned paper for the National Academies of Sciences, Engineering, and Medicine, Board on Science Education, Committee on Science Investigations and Engineering Design for Grades 6–12, Washington, DC, USA, 2018.
- [15] A. G. Chakarov, Q. Biddy, J. Jacobs, M. Recker, and T. Sumner, "Opening the black box: Investigating student understanding of data displays using programmable sensor technology," in *Proc. ACM Conf. Int. Computing Education Research (ICER '20)*, Virtual Event, New Zealand, Aug. 2020, pp. 291–301, doi: 10.1145/3372782.3406268.
- [16] D. Weintrop *et al.*, "Defining computational thinking for mathematics and science classrooms," *J. Sci. Educ. & Technol.*, vol. 25, no. 1, pp. 127–147, Feb. 2016, doi: 10.1007/S10956-015-9581-5.
- [17] V. R. Lee and M. DuMont, "An exploration into how physical activity data-recording devices could be used in computer-supported data investigations," *Int. J. Comput. Math. Learn.*, vol. 15, no. 3, pp. 167–189, Oct. 2010, doi: 10.1007/s10758-010-9172-8.
- [18] F. Martin *et al.*, "ISENSE: A web environment and hardware platform for data sharing and citizen science," paper presented at the AAAI Spring Symp. Series, Palo Alto, CA, USA, Mar. 22–24, 2010. [Online]. Available: <http://www.aaai.org/ocs/index.php/SSS/SSS10/paper/view/1099>
- [19] E. Russell, D. Edelson, and A. Switzer, "National Geographic FieldScope: A collaborative geospatial platform for citizen science," in *Proc. 7th IEEE Int. Conf. e-Science Workshops*, Stockholm, Sweden, Dec. 2011, pp. 34–38, doi: 10.1109/eScienceW.2011.24.
- [20] B. Hug and K. L. McNeill, "Use of first-hand and second-hand data in science: Does data type influence classroom conversations?," *Int. J. Sci. Educ.*, vol. 30, no. 13, pp. 1725–1751, Oct. 2008, doi: 10.1080/09500690701506945.
- [21] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Future Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013, doi: 10.1016/j.future.2013.01.010.
- [22] S. Gul, M. Asif, S. Ahmad, M. Yasir, M. Majid, and M. S. A. Malik, "A survey on role of Internet of Things in education," *Int. J. Comput. Sci. & Netw. Secur.*, vol. 17, no. 5, pp. 159–165, May 2017. [Online]. Available: http://paper.ijcns.org/07_book/201705/20170520.pdf
- [23] J. Gómez, J. F. Huete, O. Hoyosa, L. Perez, and D. Grigori, "Interaction system based on Internet of Things as support for education," *Procedia Comput. Sci.*, vol. 21, pp. 132–139, 2013, doi: 10.1016/j.procs.2013.09.019.
- [24] P. Putjorn, P. Siriaraya, F. Deravi, and C. S. Ang, "Investigating the use of sensor-based IoT to facilitate learning for children in rural Thailand," *PLoS ONE*, vol. 13, no. 8, Aug. 2018, Art. no. e0201875, doi: 10.1371/journal.pone.0201875.
- [25] G. González, M. M. Organero, and C. D. Kloos, "Early infrastructure of an Internet of things in spaces for learning," in *Proc. 8th IEEE Int. Conf. Advanced Learning Technologies*, Santander, Spain, Jul. 2008, pp. 381–383, doi: 10.1109/icalt.2008.210.
- [26] NASA Earth Science & Remote Sensing Unit, "Earth science & remote sensing missions on ISS," https://eol.jsc.nasa.gov/ESRS/ISS_Remote_Sensing_Systems/ (accessed Mar. 1, 2020).
- [27] L. Norooz, M. L. Mauriello, A. Jorgensen, B. McNally, and J. E. Froehlich, "BodyVis: A new approach to body learning through wearable sensing and visualization," in *Proc. 33rd Annu. ACM Conf. Human Factors in Computer Systems (CHI '15)*, Seoul, South Korea, Apr. 2015, pp. 1025–1034, doi: 10.1145/2702123.2702299.
- [28] M. Wilkerson and V. Laina, "Middle school students' reasoning about data and context through storytelling with repurposed local data," *ZDM—Math. Educ.*, vol. 50, no. 3, pp. 1223–1235, Jul. 2018, doi: 10.1007/s11858-018-0974-9.
- [29] C. Xie, "National Science Foundation funds research and development of an IoT platform for smart schools," *Concord Consortium Blog*, May 2017. [Online]. Available: <http://molecularworkbench.blogspot.com/2017/05/national-science-foundation-funds.html>
- [30] M. Resnick, F. Martin, R. Sargent, and B. Silverman, "Programmable bricks: Toys to think with," *IBM Syst. J.*, vol. 35, no. 3–4, pp. 443–452, 1996, doi: 10.1147/sj.353.0443.
- [31] C. Dixon, L. Hardy, S. Hsi, and S. Van Doren, "Computational tinkering in science: Designing space for computational participation in high school biology," in *Proc. 14th Int. Conf. Learning Sciences (ICLS '20)*, vol. 1, M. Gresalfi and I. S. Horn, Eds. Nashville, TN, USA, Jun. 2020, pp. 154–161, doi: 10.22318/icls2020.154.
- [32] Next Generation Science Storylines. <http://www.nextgenstorylines.org> (accessed Oct. 12, 2020).
- [33] L. Hardy and M. Lewandowski, "Under the hood: Using Raspberry Pi and Wi-Fi to do more with data," *@Concord*, vol. 22, no. 2, p. 14, 2018. [Online]. Available: <https://concord.org/newsletter/2018-fall/using-raspberry-pi-and-wi-fi-to-do-more-with-data/>
- [34] Computer Science Teachers Association, "CSTA K–12 CS Standards," 2017. [Online]. Available: <http://www.csteachers.org/standards>.
- [35] W. M. Johnston, J. R. P. Hanna, and R. J. Millar, "Advances in dataflow programming languages," *ACM Comput. Surv.*, vol. 36, no. 1, pp. 1–34, Mar. 2004, doi: 10.1145/1013208.1013209.
- [36] K. N. Whitley, L. R. Novick, and D. Fisher, "Evidence in favor of visual representation for the dataflow paradigm: An experiment testing LabVIEW's comprehensibility," *Int. J. Human-Comput. Stud.*, vol. 64, no. 4, pp. 281–303, Apr. 2006, doi: 10.1016/j.ijhcs.2005.06.005.
- [37] E. Baroth and C. Hartsough, "Visual programming in the real world," in *Visual Object-Oriented Programming: Concepts and Environments*, M. Burnett, A. Goldberg, and T. Lewis, Eds. Shelter Island, NY, USA: Manning, 1995, ch. 2, pp. 21–42.
- [38] N. Halbwachs, P. Caspi, P. Raymond, and D. Pilaud, "The synchronous data flow programming language LUSTRE," in *Proc. IEEE*, vol. 79, no. 9, pp. 1305–1320, Sep. 1991, doi: 10.1109/5.97300.
- [39] S. Gauvin, M. Paquet, and V. Freiman, "Vizwik—visual data flow programming and its educational implications," in *Proc. 2015 World Conf. Educational Media and Technology (EdMedia '15)*, S. Carliner, C. Fulford, and N. Ostashewski, Eds. Montréal, Canada, Jun. 2015, pp. 1594–1600. [Online]. Available: <https://www.learntechlib.org/primary/p/151435/>
- [40] *Sensaurus*. (2020). Manylabs. Accessed: Feb. 16, 2020. [Online]. Available: <https://github.com/manylabs/sensaurus>
- [41] Amazon Web Services. "AWS IoT core documentation," https://docs.aws.amazon.com/iot/?id=docs_gateway (accessed Jan. 23, 2020).
- [42] *Dataflow*. (2020). The Concord Consortium. Accessed: Feb. 16, 2020. [Online]. Available: <https://github.com/concord-consortium/collaborative-learning/tree/dataflow-production>



Leslie G. Bondaryk received her B.S. degree in electrical engineering from the Massachusetts Institute of Technology in 1988 and M.S. degree in Electrical Engineering from the University of California, Santa Barbara in 1990.

From 1992 to 2020 she has introduced new technologies to educational research and publishing projects across computer science, mathematics, engineering and sciences, including the first Web Calculus text, *The Analytical Engine Online* (PWS Publishing, 1998), and *Schaum's Interactive Outline Series* (McGraw Hill, 1994-2000). She is the author of papers, articles and book chapters on technology adoption in traditional classrooms, citizen science, and more recently on collaborative technologies in STEM software. Her research interests include data visualization, collaborative learning technologies, and novel interfaces to communicate modeling concepts. She is currently the Director of Technology at the Concord Consortium, Concord, MA.



Sherry Hsi received her B.S., M.S., and Ph.D. degrees in engineering science, mechanical engineering and science education in 1986, 1988, and 1997 from the University of California Berkeley.

She has conducted design-based research, evaluation studies, and learning technology design for over 30 years working in science centers, out-of-school settings, and in middle and secondary school classrooms. She is co-author of *Computers, Teachers, Peers: Science Learning Partners*, and has served as an Associate Editor of the *Journal of the Learning Sciences*. She is currently a Principal Scientist at BSCS Science Learning, Colorado Springs, CO.

Dr. Hsi is the principal investigator of several National Science Foundation-supported research projects with an interest in supporting inquiry-based STEM education, tinkering, and agentive learning among youth.



Seth Van Doren received his B.A. degree in Chemistry from the University of California Berkeley in 2019.

He has worked as a research assistant at Lawrence Berkeley National Laboratory and the Berkeley Cal Teach program, primarily studying the effects of undergraduate research experiences on STEM identity development. From 2019 to 2020, he was a research associate at the Concord Consortium. He is currently a Research Assistant at BSCS Science Learning, Colorado Springs, CO. His research interests include authentic STEM investigations in K–12 settings, social learning, and increasing student epistemic agency.