Interactive Web Notebooks Using the Cloud to Enable CS in K-16+ Classrooms and PDs

Dr. Mike Borowczak, University of Wyoming

Dr. Mike Borowczak is currently a Professor of Professor in the Computer Science department at the University of Wyoming, where he leads the Cybersecurity Education and Research (CEDAR) center. He earned his Ph.D. in Computer Science and Engineering (2013) as well as his BS in Computer Engineering (2007) from the University of Cincinnati. His research focuses on detection and prevention of information leakage from hardware side channels. Mike's current research interests include developing homomorphic encryption, compression and parallelized algorithms for streaming and pseudo-streaming data sources while developing authentic cyber learning experiences for K-20 students. Mike also has over a decade of industry and research experience – mostly revolving around the semiconductor and bioinformatics industries – with specific experience at Texas Instruments, Intel, and Cincinnati Children's Hospital Medical Center. In addition to his industry experience, Mike spent two years, while completing his Ph.D., as a National Science Foundation GK-12 fellow – teaching and bringing real-world STEM applications in two urban high schools. Since then, he has worked with university faculty to promote and extend K20 STEM outreach in Ohio, Oregon, Texas, and Wyoming. He has authored peer-reviewed articles and papers, presented at national and international conferences, and taught undergraduate/graduate courses in Computer Security, Data Mining, VLSI and pedagogy in STEM. Mike is an executive committee member of the IEEE Computer Society's Technical Committee on VLSI, as well as an active member of the IEEE, ASEE, ASTE, among others.

Dr. Andrea Carneal Burrows, University of Wyoming

Andrea C. Burrows is an assistant professor in the Department of Secondary Education at the University of Wyoming, where she teaches courses in science methods and pedagogy. Dr. Burrows taught at Northern Kentucky University for five years. In 2010, she was hired as an external evaluator to conduct research on community/university partnership relations at the University of Cincinnati. She has received several awards including the: 1) Lillian C. Sherman Award for outstanding academic achievement (2011); 2) UW College of Education outstanding research award (2015); and 3) UW College of Education outstanding service award (2016). Her research interests include partnerships with in pre-service and in-service teachers in STEM Education with a focus on engineering education applications. An active member of AERA, ASEE, ASTE, NARST, and NSTA, Dr. Burrows has presented at over 50 conferences, published in ranked journals (e.g. Journal of Chemical Education), reviewed conference proposals (e.g ASEE, AERA), and co-edits the CITE-Science journal. Additionally, she taught high school and middle school science for twelve years in Florida and Virginia, and she was the learning resource specialist for the technology demonstration school in Florida.

Interactive Web Notebooks

Using the Cloud to Enable CS in K16+ Classrooms and PDs

Mike Borowczak[1] and Andrea Burrows[2]
[1]Department of Computer Science
[2]Department of Secondary Education
University of Wyoming
mike.borowczak@uwyo.edu , andrea.burrows@uwyo.edu

Cloud-based implementations of interactive code notebooks and multi-agent simulators enable K12 and collegiate level instructors to incorporate computer science (CS) into their existing science, technology, engineering and mathematics (STEM) courses and professional developments (PDs). Three different K16+ instructor-learner interactions are highlighted through case-studies. These case studies include: a university instructor's journey with professional development and undergraduate instruction, a high school instructor's self-guided learning, and finally an elementary instructor's transition from learner to instructor of CS content. In addition to the case studies, data from a CS PD is used to compare the impact and adoption rate of cloud based programming methods versus traditional approaches. Overall findings align with prior research, and show that teachers, after overcoming initial frustration and hesitation, are much more willing to implement freely available web-based technologies than other technology which requires additional levels of support and per-student expenditure.

Keywords: *Computer Science Education, Computer Science, STEM, K12 Teachers, Pre-Service Teacher Education, Engineering Education, Cloud Computing*

Introduction

One of the main challenges in incorporating computer science (CS) into existing K12 instruction surrounds the technical setup costs and infrastructure requirements. Instructors would like to easily incorporate computer science into their existing courses and professional development, but non-programming centric classrooms are rarely enabled to provide an appropriate learning environment. Many current web-based introductions to CS are limited to a finite set of activities (e.g. Hour of Code) and once the mini-course activity is completed, the learner is expected to transition their programming from a constrained programming environment to their local computer. While these activities provide learners the ability to experience and practice fundament computer science activities, the post-web activity transition generally stifles learners' continued use, learning, and involvement with the target programming language. Additionally, since K12 Science and Math teachers generally do not have the resources or support to establish the suite of tools needed to fully integrate computer science within their curriculum the CS exposure that they can provide to students is generally limited to several pre-packaged web-based activities. This work explores the use of web-based tools which have less emphasis on pre-packaged, constrained activities, and more emphasis on replicating fully functional programming environments. The two technologies investigated are 1) NetLogo Web and 2) Jupyter Notebooks running on a JupyterHub cloud instance.

Problems and Research Questions

Collegiate faculty in Science, Technology, Engineering and Mathematics (STEM) are being asked to implement more active engagement opportunities within their classrooms. In certain predominately theoretical courses, such as those found in Mathematics and Computer Science – active engagement may seem far unattainable – especially without practice. Simultaneously, with the current push for CS in K12 education, CS faculty are becoming more engaged in educational outreach opportunities that are developed and coordinated through educational researchers. The ability to teach novice K12 teachers CS concepts, using non-traditional pedagogies (including 'active learning' strategies such as inquiry, problem based learning, and so forth), can provide collegiate faculty insights into engaging their own students within the classroom.

One such mechanism for building sustained, active engagement in and out of the classroom involves the incorporation of manipulatives. In highly theoretical coursework, however, manipulatives need not be physical in nature but rather they can take the form of interactive web based models or code notebooks. As a preliminary study of how these cloud-based or web-based manipulatives impact learning, the researchers focuses on assessing the feasibility and impact on learning when incorporating cloud-based environments into a CS PD for K12 teachers. With this goal in mind, the following two questions formed the basis for the pilot study:

1. How do cloud-based programming environment enable K16 STEM instructors to incorporate computer science concepts into their instruction?
2. Do cloud-based programming environments increase learner content gains during and after initial instruction?

Theoretical Framework & Literature Review

The researchers selected a social constructivism theoretical framework, where interactions between people, in this case K-16 teachers, faculty, and students, allow for content understanding of the CS material presented (Vygotsky, 1978). There is a global push to increase K-12 student interest in Science, Technology, Engineering and Math (STEM) disciplines (Blackley & Howell, 2015). Unfortunately, relatively few K-12 students have the opportunity to engage in authentic, age and cost appropriate activities related to the computing spectrum in the classroom. In order to engage and develop student interest in computing professions, researchers propose the need for low-cost use of manipulatives focused on computing and construction, physics, and mechanics in STEM (Nadelson, Callahan, Pyke, Hay, & Schrader, 2010). To engage students in CS, educators must expose their students to computing applications that move beyond robotic (Margolis, Fisher, & Miller, 2000).

Using web-based programming environments so that instructors and students can explore and create using CS concepts, is an extension of prior research relating to four main areas including: 1) Teaching CS with Technology ; 2) K-12 students using and learning CS; 3) K-12 teachers

using and learning CS; and 4) Higher education student and faculty using and learning CS. Finally, previous research indicates that "involvement with modeling scientific phenomena and complex systems can play a powerful role in science learning" (Hashem & Mioduser, 2011, p. 151).

The area of interest in exploring the use of CS technology focuses on descriptions of NetLogo as a multi-agent programming language and modeling environment (Blikstein, Abrahamson, & Wilensky, 2005: Tisue & Wilensky, 1999; Tisue & Wilensky, 2004; Wilensky, 2001; Wilensky, 1999). NetLogo, a multi-agent simulator that leverages the popular Logo programming language was originally developed as a 'learning language.' Multi-agent simulators define the characteristics or behaviors of a singular agent (e.g. ant, worker ant, queen ant, bird, atom, photon, etc.). The simulator then implements many copies of pre-defined agents in a constrained world (e.g. ant colony, flock of birds, atoms, photons, etc.). Finally, the simulation engine controls the interactions of the agents within the world according to the predefined (programmed) rules (e.g. ants following pheromone trails, birds flocking, atoms binding, behavior of light, etc.). The next major theme in the literature explores technology use for K-12 students with CS interactions (Basu, Dickes, Kinnebrew, Sengupta, & Biswas, 2013; Berland & Reiser, 2011; Sengupta, Dickes, Farris, Karan, Martin, & Wright, 2015; Svihla & Linn, 2012; Vattam, Goel, Rugaber, Hmelo-Silver, Jordan, Gray, & Sinha, 2011). Ultimately, there is a plethora of CS projects for researchers to explore with K-12 teachers and students.

Third, there are extensive references to K-12 teachers encouraging use of CS in curriculum and instruction (Basu, Dickes, Kinnebrew, Sengupta, & Biswas, 2013; Blikstein, 2013; Borgman, Abelson, Dirks, Johnson, Koedinger, Linn, … & Smith, 2008; Chiu & Wu, 2009; Clark, Nelson, Sengupta, & D'Angelo, 2009; Dalke, Cassidy, Grobstein, & Blank, 2007; Donnelly, Linn, & Ludvigsen, 2014; Grover & Pea, 2013; Hashem & Mioduser, 2011; Jacobson & Wilensky, 2006; Klašnja-Milićević, Vesin, Ivanovic, & Budimac, 2011; Levy & Wilensky, 2009; Maroulis, Guimera, Petry, Stringer, Gomez, Amaral, & Wilensky, 2010; Pea & Collins, 2008; Pathak, Kim, Jacobson, & Zhang, 2011; Sengupta, Dickes, Farris, Karan, Martin, & Wright, 2015; Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013; Sengupta & Wilensky, 2009; Shen, Lei, Chang, & Namdar, 2014; Wilensky, Brady, & Horn, 2014; Wilensky & Papert, 2010). The Task Force on Cyberlearning identified "directions for leveraging networked computing and communications technology. It also calls for research to establish successful ways of using these technologies to enhance educational opportunities and strengthen proven methods of learning" (p. 7). Throughout all these research domains, researchers make arguments that "the cognitive and sociocultural factors related to learning complex systems knowledge are relevant and challenging areas for learning sciences research" (Jacobson & Wilensky, 2006, p. 11). Thus, teaching and allowing exploring of CS to K-12 teachers and/or students is complicated and should be systemic.

Lastly, CS and/or computational thinking is needed across many education disciplines within higher education (Blikstein, 2011; Blikstein, 2013; Blikstein & Wilensky, 2005; Blikstein & Wilensky, 2010; Chiu & Wu, 2009; Hashem & Mioduser, 2011; Klašnja-Milićević, Vesin, Ivanovic, & Budimac, 2011; Levy & Wilensky, 2009; Maroulis, Guimera, Petry, Stringer, Gomez, Amaral, & Wilensky, Pathak, Kim, Jacobson, & Zhang, 2011; Sengupta 2010; &

Wilensky, 2009; Shen, Lei, Chag, & Namdar, 2014; Wilensky & Papert, 2010). However, as Blikstein & Wilensky (2010) claim,

> *A common element in those [higher education] programs is to introduce courses in which students design products and solutions for real-world problems, engaging in actual engineering projects. These initiatives have met with some success and are proliferating into many engineering schools. Despite their success, they have not addressed one key issue in transforming engineering education: extending the pedagogical and motivational advantages of design-based courses to theory-based engineering courses, which constitute the majority of the coursework in a typical engineering degree, and in which traditional pedagogical approaches are still predominant" (p. 17).*

The culmination of current literature continues to drive a need to investigate higher education instructor pedagogy and tools as areas in need of additional study.

Methods

The three case studies in this research originated during a 16-day professional development funded through a math and science partnership (MSP) grant. During the professional development 20 K-12 instructors were exposed to the Python programming language (3 days) and NetLogo (3 days). All of the programming, notes, and lessons materials were hosted using interactive notebooks and open source repositories. The interactive notebooks used open-source Jupyter notebooks hosted using JupyterHub – a related open source project. Similarly, NetLogo activities and code were either hosted on NetLogo's own site, the modelling commons, or using GitHub, a freely available online code repository site.

The study focuses on the 32 hours of exposure to Python through use of Jupyter notebooks and 32 hours of exposure to NetLogo through use of NetLogo Web. Specifically, the researchers focused on the utilization of the cloud-based tools, formal and informal instructor/learner interviews, pre/post content knowledge scores, and finally a technology implementation survey to answer the two research questions surrounding the use of web-based programming environments to lower the barrier of entry for CS in existing STEM courses.

Qualitatively data (e.g. surveys and informal interview quotes) show initial hesitation in learning a new programing language, but instructors gain confidence and begin to use web-based tools to continue their learning

Context

The data for this study was collected during a two-week summer PD for K-12 STEM teachers that also consisted of 6 days of follow-up sessions. A total of thirty teachers participated in the yearlong PD. The PD, called RAMPED for *Robotics, Applied Mathematics, Physics, & Engineering Design*, focused on real-world applications of CS and the research team implemented a set of six, two-day sessions. The two-day sessions included: Naturally Inspired, Space, Robotics, Virtual Reality, Arduinos, and Raspberry Pi. Teachers chose four of the six total sessions to attend during the two summer weeks and then attended extension sessions on all six topics during the academic year. Due to logistics, each session was implemented twice in the summer, and limited to 10 teachers each time it was offered. Thus only 20 teachers participated in any given content session. Primary instruction for each session was provided by content experts and supplemented by pedagogical sessions led by education faculty.

The Participants

This study focused on 20 K-12 in-service teachers who participated in the NetLogo and Space sessions. The 20 in-service teachers were: 35% (7/20) high school, 63% science focused (13), had an average of 12.9 years teaching experience, with 125.5 students per year with about 12% of their students (16) on individualized instruction plans.

Qualitative: CS Activity Adoption

While the ultimate goal of this PD was for teachers to enhance their current instruction and content knowledge, the researchers were also interested in the relationship between programming environment locale (cloud or local) and classroom adoption rates. In order to answer the question, "Are CS skill and activity implementation rates and implementation type impacted by the use of cloud-based technology versus traditional technologies?" the research team collected teacher classroom implementation plans from a simple survey 3 months after the summer PD. The survey asked whether or not the lesson plan (created in the summer) had already been implemented, and asked for details of the lesson, student reactions, limitations, and barriers to implementation. In addition, each of the potential lessons was evaluated for cost, proposed teaching model, as well as the original session's type of technology uses. The synthesized aggregation of the implementation survey is located in Table 6.2.1. In addition, teachers were asked to create presentations for their peers relating to their implementation experiences. Two presentations, which were ultimately presented at a teachers' partnership conference, were selected as case-study profiles of K12 instructors use of cloud-based technologies for CS instruction.

Quantitative: CS Content Knowledge Gains

The objective of the RAMPED PD was to better teachers' instruction and content knowledge by introducing teachers to applications of CS. Thus, the formal assessment consisted of both application and fundamental CS theory questions. In order to answer the question, "dDo web-based programming environments increase learner content gains during and after initial instruction?" this study focused on a subset of the pre/post assessment questions related to the fundamental CS theory. Table 5.3.1 contains some of the questions from the actual assessment. It is important to note that question seven, regarding the illustration of sequential operation, only contained graphical illustrations while all the remaining questions were related to real code statements in one of three programming languages: C++, Python or Logo.

Table 5.3.1 Assessment question and corresponding computer science concept(s).

Q	Session (Lang)	Location	Assessment Question (Summary)
Q5	Robotics (C++)	Local	Which command queries a robot's joint state?
Q7*	Raspberry Pi (C++)	Local	Which of the following illustrates sequential operation?
Q14	Space (Python)	Cloud	Print out the numbers 1-10
Q17.b	NNI (Logo)	Cloud	Create a process to swap to two numbers

The participants of the PD answered these question (among others) prior to the start of the summer session (pre-pre), prior to the individual 2-day session (pre), immediately following the 2-day session (post), immediately following the summer session (post-post), prior to the follow-up session (pre-follow), immediately after the follow-up session (post-follow) and once again at the conclusion of 120 contact hours (final).

Limitations

The current participant pool is limited to only 20 participants, all of whom self-selected participation in the two week paid PD. The group of participants came largely from the same general population. The participant pool, while evenly split between elementary, middle and high school educators came mostly from the same school district. Additionally, the PD lasted two summer weeks with six follow-up sessions during the school year (120 hours), which while desirable for a PD, was short to teach novices a new technical content area. The STEM content focused almost exclusively on CS and further studies are required to generalize these results to other disciplinary PDs. The implementation survey occurred 3 months after the PD, however, in order to measure long-term adoption rates more data must be collected. Finally, 75% (6/8) of PD experts were accustomed to educational outreach, which may be atypical in other technically focused PDs.

Analysis & Findings

Qualitative - Two Case Studies (Through Teacher Created Presentations)

I. **A High School Teacher's Self-Guided Learning:** One of the K12 instructors that implemented CS in their existing curriculum was an early career physics teacher with less than 2 years of teaching experience. One of her post-implementation interview responses exemplifies an overarching theme amongst the teachers' willingness to self-learn the Python programming language through web-based tools such as Jupyter Notebooks, saying "the [Jupyter Notebook] tutorials really helped me understand Python code, and now I'd like to take a class in Python along the same lines." The teacher's presentation for her peers showed how she took the summer PD content, and applied it, not only in her own self-guided learning, but also in her instruction of her own physics class. Over 30% of her presentation was dedicated to the use of programming notebooks (Jupyter) in a cloud-based environment (JupyterHub) – see Figure 1. A follow-up interview determined that percentage aligned with the focus of her in-class implementation plan.

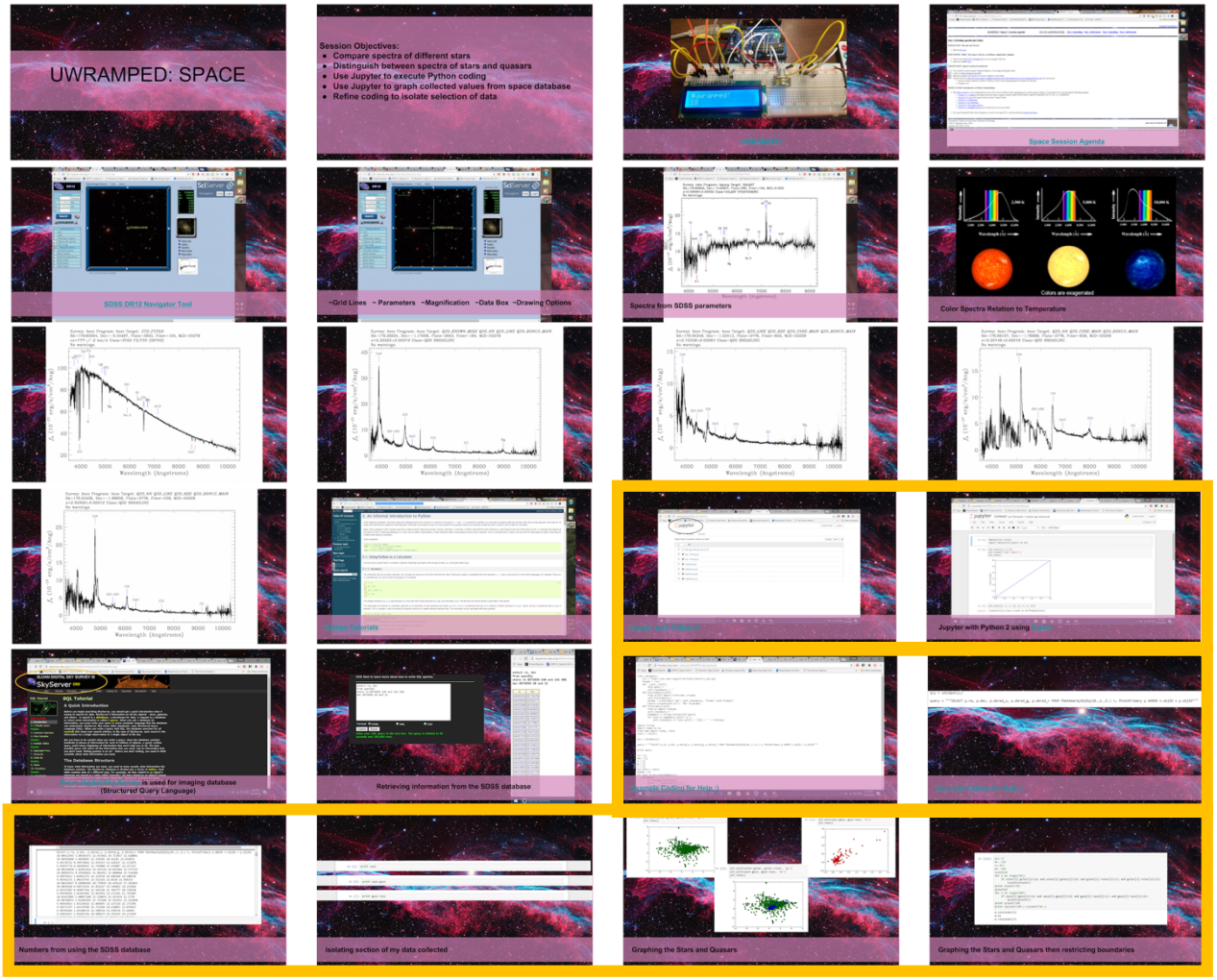

Figure 1: A teachers 24-slide presentation, with 8 slides dedicated to Jupyter/JupyterHub, highlighting her personal learning progression and her classroom implementation plan.

II. **An Elementary School Teacher's Transition from CS Learner to Instructor:** Another K-12 teacher, who implemented CS via cloud-based NetLogo Web, was a fourth grade teacher with about a decade of experience. During the professional development, this teacher specifically called out the ability to use freely available web-based technology as a benefit to her students, saying that "it would be great if I could have my students programming online, I know they would go home and continue working on it." During her presentation (shown below in Figure 2), she mentioned her fourth graders, who had never coded before, we're modifying NetLogo code within 15 minutes of their initial exploration of its available web-based interface!

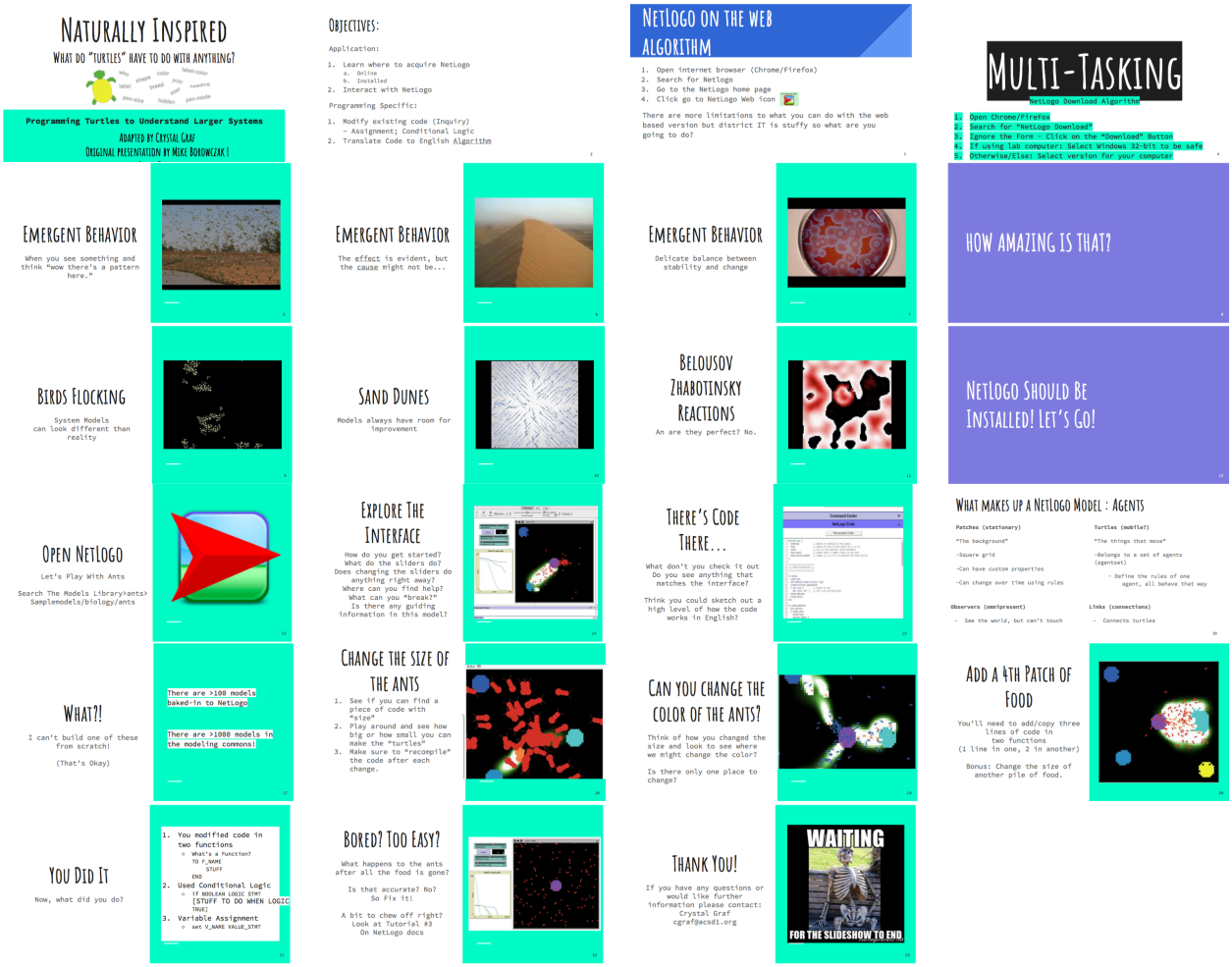

Figure 2: A teachers 23-slide presentation, with over half of the slides dedicated to NetLogo, highlighting her classroom implementation plan.

Quantitative Analysis & Findings

The 20 PD participants were asked four questions containing direct mappings to CS content knowledge on four separate occasions. Table 6.2.1 summarizes the percentage of correct responses among the 20 participants for each of the four instances: before the entire PD (pre-pre),

immediately prior to a specific session (pre), immediately following a specific session (post) and then finally at the end of the entire PD (post-post). Note that the general trend between the first three data points is strictly increasing and that the most fluctuation occurs between the post assessment and the final post-post assessment. This change is clearly visible in Figure 6.2.1, and summarized in Table 6.2.2.

Table 6.2.1 Percentage of PD participants (n=20) with correct answer for assessment questions in Table 5.3.1; Local Environments (5,7) and Cloud-Based Environments (14,17).

Question	Pre-Pre	Pre	Post	Post-Post
5	6%	18%	44%	26%
7	39%	45%	90%	82%
14	32%	39%	55%	86%
17b	29%	55%	59%	80%

Table 6.2.2 Deltas between subsequent assessments and average increases for questions over time. Local Environments (5,7) and Cloud-Based Environments (14,17).

Question	Pre-Pre to Pre Delta	Pre to Post Delta	Post to Post Post Delta	Average
5	+12%	+26%	-18%	*3.33%*
7	+6%	+45%	-8%	*14.33%*
14	+7%	+16%	+31%	*18%*
17b	+26%	+4%	+21%	*17%*

This evidence provides support of the effectiveness of using cloud-based as a technology to introduce novice teachers and ultimately, through classroom implementations, students to fundamental computer science concepts through inquiry based activities.

Findings indicate that instructors are much more likely to re-use and implement computer science in their existing courses when the challenges of doing so are not administrative or bound to computing resources. In other words, web-based technologies allow initial CS introductions without long-term commitment from district technology resources or facilitators. Additionally, the instructors in a CS based PD are more likely to become self-learners and implement CS concepts in their own classrooms when given access to a web-based programming interface.

In conclusion, exposure, continuous access, and use of live notebooks during the PD and in schools enables K12 instructor to more readily implement CS concepts within their own courses. The implication allows future PD and instructional facilitators a mechanism to get K20 instructors to utilize cloud-based coding environments as a means for incorporating CS into existing STEM courses. The researchers are currently investigating the impact of the PD on the collegiate STEM faculty who implemented the individual sessions.

Acknowledgments

This work was supported by 1) A federal grant – called RAMPED - under No Child Left Behind (NCLB) (P.L.107F110, Title II, Part B) administered by the Wyoming Department of Education (MSP Grant #1601506MSPA2); and 2) NSF Noyce – called SWARMS - (NSF Grant# 1339853).

References

Basu, S., Dickes, A., Kinnebrew, J. S., Sengupta, P., & Biswas, G. (2013). CTSiM: A Computational Thinking Environment for Learning Science through Simulation and Modeling. In *CSEDU* (pp. 369-378).

Berland, L. K., & Reiser, B. J. (2011). Classroom communities' adaptations of the practice of scientific argumentation. *Science Education*, *95*(2), 191-216.

Blikstein, P. (2011, February). Using learning analytics to assess students' behavior in open-ended programming tasks. In *Proceedings of the 1st international conference on learning analytics and knowledge* (pp. 110-116). ACM.

Blikstein, P. (2013). Digital fabrication and 'making' in education: The democratization of invention. *FabLabs: Of machines, makers and inventors*, 1-21.

Blikstein, P., Abrahamson, D., & Wilensky, U. (2005, June). Netlogo: Where we are, where we're going. In *Proceedings of the annual meeting of Interaction Design and Children, Press*.

Blikstein, P., & Wilensky, U. (2005). Less is more: Agent-based simulation as a powerful learning tool in materials science. In *IV International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2005), Utrecht, Holland*.

Blikstein, P., & Wilensky, U. (2010). MaterialSim: A constructionist agent-based modeling approach to engineering education. In *Designs for learning environments of the future* (pp. 17-60). Springer US.

Borgman, C. L., Abelson, H., Dirks, L., Johnson, R., Koedinger, K. R., Linn, M. C., ... & Smith, M. S. (2008). Fostering learning in the networked world: The cyberlearning opportunity and challenge. A 21st century agenda for the National Science Foundation. *Report of the NSF task force on cyberlearning*.

Chiu, M. H., & Wu, H. K. (2009). The roles of multimedia in the teaching and learning of the triplet relationship in chemistry. In *Multiple representations in chemical education* (pp. 251-283). Springer Netherlands.

Clark, D., Nelson, B., Sengupta, P., & D'Angelo, C. (2009, October). Rethinking science learning through digital games and simulations: Genres, examples, and evidence. In *Learning science: Computer games, simulations, and education workshop sponsored by the National Academy of Sciences, Washington, DC*.

Dalke, A. F., Cassidy, K., Grobstein, P., & Blank, D. (2007). Emergent pedagogy: learning to enjoy the uncontrollable—and make it productive.*Journal of Educational Change*, *8*(2), 111-130.

Donnelly, D. F., Linn, M. C., & Ludvigsen, S. (2014). Impacts and characteristics of computer-based science inquiry learning environments for precollege students. *Review of Educational Research*, *84*(4), 572-608.

Goel, A. K., Rugaber, S., & Vattam, S. (2009). Structure, behavior, and function of complex systems: The structure, behavior, and function modeling language. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, *23*(01), 23-35.

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

Hashem, K., & Mioduser, D. (2011). The Contribution of Learning by Modeling (LbM) toStudents' Understanding of Complexity Concepts.*International Journal of e-Education, e-Business, e-Management and e-Learning*, *1*(2), 151.

Jacobson, M. J., & Wilensky, U. (2006). Complex systems in education: Scientific and educational importance and implications for the learning sciences. *The Journal of the learning sciences*, *15*(1), 11-34.

Klašnja-Milićević, A., Vesin, B., Ivanović, M., & Budimac, Z. (2011). E-Learning personalization based on hybrid recommendation strategy and learning style identification. *Computers & Education*, *56*(3), 885-899.

Levy, S. T., & Wilensky, U. (2009). Crossing levels and representations: The Connected Chemistry (CC1) curriculum. *Journal of Science Education and Technology*, *18*(3), 224-242.

Maroulis, S., Guimera, R., Petry, H., Stringer, M. J., Gomez, L. M., Amaral, L. A. N., & Wilensky, U. (2010). Complex systems view of educational policy research. *Science*, *330*(6000), 38-39.

Pathak, S. A., Kim, B., Jacobson, M. J., & Zhang, B. (2011). Learning the physics of electricity: A qualitative analysis of collaborative processes involved in productive failure. *International Journal of Computer-Supported Collaborative Learning*, *6*(1), 57-73.

Pea, R. D., & Collins, A. (2008). Learning how to do science education: Four waves of reform. In Y. Kali, M. C. Linn, & J. E. Roseman (Eds.), *Designing coherent science education*. New York: Teachers College Press.

Sengupta, P., Dickes, A., Farris, A. V., Karan, A., Martin, D., & Wright, M. (2015). Programming in K-12 science classrooms. *Communications of the ACM*, *58*(11), 33-35.

Sengupta, P., Kinnebrew, J. S., Basu, S., Biswas, G., & Clark, D. (2013). Integrating computational thinking with K-12 science education using agent-based computation: A theoretical framework. *Education and Information Technologies*, *18*(2), 351-380.

Sengupta, P., & Wilensky, U. (2009). Learning electricity with NIELS: Thinking with electrons and thinking in levels. *International Journal of Computers for Mathematical Learning*, *14*(1), 21-50.

Shen, J., Lei, J., Chang, H. Y., & Namdar, B. (2014). Technology-enhanced, modeling-based instruction (TMBI) in science education. In *Handbook of research on educational communications and technology* (pp. 529-540). Springer New York.

Svihla, V., & Linn, M. C. (2012). A design-based approach to fostering understanding of global climate change. *International Journal of Science Education*, *34*(5), 651-676.

Tisue, S., & Wilensky, U. (2004, May). Netlogo: A simple environment for modeling complexity. In *International conference on complex systems* (Vol. 21, pp. 16-21).

Tisue, S., & Wilensky, U. (2004, October). NetLogo: Design and implementation of a multi-agent modeling environment. In *Proceedings of agent* (Vol. 2004, pp. 7-9).

Vattam, S., Goel, A. K., Rugaber, S., Hmelo-Silver, C. E., Jordan, R., Gray, S., & Sinha, S. (2011). Understanding Complex Natural Systems by Articulating Structure-Behavior-Function Models. *Educational Technology & Society*, *14*(1), 66-81.

Vygotsky, L. (1978). Interaction between learning and development. In Gauvain & Cole (Eds.), *Readings on the Development of Children* (pp. 34-40). New York: Scientific American Books.

Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

Wilensky, U. (2001). Modeling nature's emergent patterns with multi-agent languages. In *Proceedings of EuroLogo* (pp. 1-6).

Wilensky, U., Brady, C. E., & Horn, M. S. (2014). Fostering computational literacy in science classrooms. *Communications of the ACM*, *57*(8), 24-28.

Wilensky, U., & Papert, S. (2010). Restructurations: Reformulations of knowledge disciplines through new representational forms. *Constructionism*.