From Professional Development to the Classroom: Findings from CS K-12 Teachers

Lori Pollock, Chrystalla Mouza, Amanda Czik, Alexis Little, Debra Coffey, Joan Buttram
University of Delaware
Newark, DE 19716 USA
{pollock, cmouza, aczik, alittl, djcoffey, jbuttram}@udel.edu

ABSTRACT
The *CS for All* initiative places increased emphasis on the need to prepare K-12 teachers of computer science (CS). Professional development (PD) programs continue to be an essential mechanism for preparing in-service teachers who have little formal background in CS content, skills, and teaching pedagogy. While increased investment by federal agencies and the industry has raised the number of CS PD opportunities for K-12 teachers, there has been limited study of how teachers apply what they learn back in their classroom. This paper describes an in-depth qualitative study through interviews of 28 elementary, middle and high school teachers who participated in summer PD in preparation of teaching a full CS course or integrate CS modules into existing courses (e.g., science, engineering, business, technology, etc). The interview protocol focused on educators' involvement in the PD, specific skills and strategies they learned, whether and how they have been able to apply these new skills in the classroom, what facilitated or impeded this application, and how students have responded.

CCS Concepts
• **Social and professional topics – computing education**

Keywords
CS education; computational thinking; K12; CS principles, teacher professional development

1. INTRODUCTION
With the increasing focus on broadening participation in computing [3] through computer science (CS) in K-12, there has been a growing investment into creating professional development (PD) opportunities to prepare teachers to teach CS principles, either as the new CS AP course or in some other integrated way within their curriculum. Teachers play a key role in how curricula are implemented for students, and effective PD is key to helping teachers build capacity to use curricula, such as Computer Science Principles, effectively [1, 5]. Specifically, when teachers actively participate in approximately 49 hours of research-based PD spread over 6-12 months, they can increase student achievement by as much as 21% [12]. Particularly for CS, the Landscape study [2] found that 75% of the CS PD providers reported working with participants who were new to CS, while 30% reported that over half of their participants were new to CS.

While the wider availability of PD programs can increase the reach of these opportunities to more teachers, it is important to evaluate the design features of PD that appear to help teachers gain the CS content, skills and teaching pedagogy that ultimately make an impact on student learning and participation in the classroom. Some of these PD programs, which were started earlier and continue to be offered, have indeed been studied to learn how to improve the design of CS-focused PD [4,6-11]. Many of these studies, however, focus on surveys that help reach a large number of respondents. Albeit useful, they only allow the connection of a limited amount of data from each respondent.

To complement the larger survey studies, this paper describes an in-depth qualitative study through interviews of 28 elementary, middle and high school teachers who participated in summer PD in preparation to teach a full CS course or integrate CS modules into existing courses (e.g., science, engineering, business, technology, or other courses). Interviews are useful for gathering a wide range of qualitative data and insights and for gaining an understanding of the broad context and environment that the interviewees operate in. In addition, their interactive nature allows for collecting in-depth information about participants' thoughts and opinions. However, their high costs restrict the number that can be performed.

2. METHODS
2.1 Research Question
In this paper, we focus on the following research questions:

1. Which CS Principles did PD participants implement in their curriculum? How were they implemented and what factors contributed to implementing them?

2. What was the student response and engagement in different CS principles? What were the teaching successes and challenges in implementing CS Principles into the curriculum?

2.2 Context of Study
The study is situated in the Delaware Partner4CS summer teacher PD experience [12] with one instance each of three consecutive summers (2012-2015) with optional participation in teacher support during the academic year. The goal of the PD was to improve CS teaching by providing educators with content knowledge of CS and CS principles and helping them develop their pedagogical content knowledge related to computer science (PCK-CS). Each PD experience was 4 ½ days of face-to-face (F2F) instruction (9-4 each day).

SIGCSE '17, March 08-11, 2017, Seattle, WA, USA
© 2017 ACM. ISBN 978-1-4503-4698-6/17/03 $15.00
DOI: http://dx.doi.org/10.1145/3017680.3017739

Teachers can participate in either a track directed toward preparing them to integrate CS activities into existing STEM curricula at either the elementary, middle or high school level (Integrate Track), or a track focused on preparing high school teachers to implement a full CS course, particularly the new CSP AP course (Course Track). Both tracks include explicit attention to CS content, pedagogical strategies for teaching CS, and strategies for broadening participation in computing. In Year 3, teachers were also offered a hybrid PD option where the face-to-face meetings were supplemented by 5 weeks of online PD. This option was available only for Couse Track participants.

2.3 Data Collection

The PD organizers sent a request for interview to all 84 teachers who had participated in at least one week-long summer PD effort. Of the 84 requests, 28 teachers agreed to be interviewed. The interviews were conducted by professionals from an education research center, with no PD organizers present.

Appendix 1 shows the interview questions. The interview protocol included 14 questions that focused on educators' involvement in the PD, specific skills and strategies they learned, whether and how they have been able to apply these in the classroom, what facilitated or impeded this application, and how students have responded. All but one interview was conducted by telephone; the exception was conducted in person. Interview participants were promised confidentiality as part of their informed consent. The interviews typically lasted 30 minutes, though some interviews were shorter and some longer depending on each participant's responses to the 14 questions. All interviews were transcribed, and open-ended responses were analyzed to identify common and unique themes.

2.4 Participants

The 28 participants in this study varied in grade level taught, with an equal number teaching middle school as high school, and only 5 teaching at the elementary school level. About half the participants teach a CS course or advise a CS club. Most participants also teach mathematics, science, technology, engineering, or business courses. The interviewed educators participated in different years and tracks of the PD program, and differing numbers participated in additional activities related to the PD. Seven educators participated in the CS-course track, eight participated in the Integration track, seven took both tracks (in different years), and evaluators were not able to place six participants. A little more than a third of the interviewed educators participated in follow-up, academic-year teacher support, which began in Year 2. Only two participants indicated that they participated in the hybrid PD Course, which was offered in Year 3.

During the interviews, participants were asked to rate their levels of knowledge about seven CS principles: creativity, abstraction, data, algorithms, programming, Internet, impacts of CS. Participants reported that they are most knowledgeable about the Internet and the impact of computing. The principles that were rated the lowest, programming and abstraction, also had the highest standard deviations, which indicates that these were the principles about which participants chose ratings that differed most from other participants. When describing their ratings, teachers emphasized the importance of creativity, which they saw as sparking and reinforcing student interest and having the most potential to help students in areas outside of CS.

3. RESULTS

3.1 Implementation of CS Principles

Teachers were asked which, if any, of the seven CS principles they integrated into their curriculum, whether the principles were integrated into existing units or presented as discrete units or lessons, and what factors contributed to their implementation or lack thereof. All but six teachers (78.6%) indicated that, during the school day, they implemented at least some of the principles; the remaining six (21.4%) responded that they integrated all of the principles.

Table 1. Grade Levels and Subjects Taught by Participants

Grade Level (*n* = 26a)	**Number**	**Percent**
Elementary	5	19.2
Middle	12	46.2
High	12	46.2
Subject/Department (*n* = 26*)		
Computer Science (including 1 CS club and 1 Advanced Placement CS)	13	50.0
Mathematics	7	26.9
Science (including "science," chemistry, and physics)	5	19.2
Technology	4	15.4
Engineering	3	11.5
Business	3	11.5
Administrator	1	3.8
Librarian	1	3.8

Not all participants provided this information. The numbers do not total 28 because many participants who answered provided more than one response, e.g., they teach elementary school and middle school or mathematics and science.

The principles of Creativity (50.0%) and Programming (46.4%) were implemented by the largest numbers of teachers in their classrooms during the school day. The rates of implementation were lower for afterschool programs; slightly less than half (44.4%) indicated that they had implemented at least some of the principles. Creativity (27.8%) and Programming (33.3%) were again the most frequently addressed (along with Algorithms, at 27.8%).

In general, slightly more teachers presented the CS principles as part of integrated units (42.3%) rather than discrete units or lessons (30.8%). This seemed to be related in large part to the courses being taught and the school level: teachers who were teaching courses other than CS— and no middle school teachers taught a CS course—found integrating the CS principles into existing units more challenging than those who were teaching CS. In addition, most of the afterschool programs discussed were about CS or directly related to it (e.g., robotics club), and just over half (55.6%) of the participants indicated that they integrated the principles into those units.

Table 2 presents the data for the factors that the teachers reported as affecting their implementation of the CS principles into their curriculum. Very broadly speaking, factors related to student needs or interest were the most widely cited factors that contributed to implementation of CS principles. The one area in which there seems to be substantial difference between groups was curriculum fit. Slightly more than two-thirds (71.4%) of participants who attended only the Course track or both tracks (attended both Course and Integrate) identified curriculum fit as a factor that contributed to implementation, while fewer teachers who attended only the Integration track or whose track was unknown mentioned this as a factor, at 25.0 percent and 33.3 percent, respectively.

Table 2. Frequencies of Implementation Factors

Factor	Prog	Course	Integrate	Both	Unknown	All
Student interest or creativity; positive effects for students	In	4 (57%)	6 (75%)	4 (57%)	3 (50%)	17 (60%)
	After	3 (60%)	3 (75%)	5 (71%)	1 (50%)	12 (66%)
Developmental appropriateness/ Student readiness	In	5 (71%)	4 (50%)	2 (28%)	5 (83%)	16 (57%)
	After	3 (60%)	2 (50%)	3 (42%)	2 (100%)	10 (55%)
Curriculum/ Organizational Fit	In	5 (71%)	2 (25%)	5 (71%)	2 (33%)	14 (50%)
	After	2 (40%)	1 (25%)	3 (42%)	2 (100%)	8 (44%)
Time/Logistics	In	1 (14%)	3 (37%)	3 (42%)	2 (33%)	9 (32%)
	After	4 (80%)	0 (0%)	2 (28%)	2 (100%)	8 (44%)
Teacher comfort	In	1 (14%)	4 (50%)	2 (28%)	1 (16%)	8 (28%)
	After	0 (0%)	0 (0%)	2 (28%)	1 (50%)	3 (16%)

# of participants in each track: School-day: Course = 7; Integrate = 8; Both = 7; Unknown = 6.

Afterschool: Course = 5; Integrate = 4; Both = 7; Unknown = 2

Table 3 shows the participants' reported use of the various CS teaching strategies. When asked about the strategies that they used, which were emphasized in the PD, at least three-quarters of the participants indicated that they used collaborative problem-based learning, paired programming, kinesthetic activities, and demonstrations when teaching CS principles. Additionally, 60 percent or more of the teachers indicated that they used "Show and Follow," Process Oriented Guided Inquiry Learning (POGIL), and online activities.

Table 3. Participants' Reported Use of Teaching Strategies

Teaching Strategy	Number	Percent
Collaborative Problem-Based Learning	25	89.3
Paired Programming	24	85.7
Kinesthetic Activities	21	75.0
Demonstrations	21	75.0
Show and Follow	20	71.4
Process Oriented Guided Inquiry Learning (POGIL)	19	67.9
Online Activities	18	64.3
Other	7	20.0
Lecture	3	10.7

N = 28.

Teachers noted that their access to certain technological resources influenced what types of teaching strategies they employed. For example, one teacher did not use much paired programming in the classroom: "*We have a ton of technology, so they don't have to share, . . . it's not so much paired. They're doing their own thing and talking with one another.*"

3.2 Student-level Impact

Table 4 presents the data on teachers' perspectives of teaching successes and challenges. Every teacher was able to note various examples of student success while teaching the CS principles. Almost two-thirds (63.0%) of teachers noted that increased exposure and interest, especially for groups traditionally underrepresented in CS, were major successes in teaching CS principles.

Table 4. Teaching Successes and Challenges

Codes	Course		Integrate		Both		Unknown	
	n	%	n	%	n	%	n	%
What have been your greatest successes in teaching about the CS principles?								
Increased self-efficacy	4	57	1	12	6	85	3	50
Increased student exposure and interest	6	85	6	75	4	57	2	33
Getting underrepresented groups interested CS	1	14	1	12	1	14	0	0
Teaching life lessons	0	0	0	0	0	0	2	33
What have been your greatest challenges in teaching about the CS principles?								
Logistics and scheduling	3	42	2	25	2	28	2	33
Teacher content knowledge of CS	1	14	4	50	1	14	1	16
Accounting for different student levels and interests	2	28	1	12	0	0	3	50
Finding enough CS resources	1	14	1	12	2	28	2	33
Hardware issues	1	14	0	0	3	42	0	0
Incorporating CS into Non-CS curriculum	0	0	0	0	1	14	0	0

Note: CSP Track, *n* = 7; Module Track, *n* = 8; Multiple Tracks, *n* = 7; Unknown, *n* = 6

Many teachers also noted specifically that student interest in CS increased as they learned how to use CS principles to express their creativity. Many students enjoyed creating games, programming, and working with codes to create tangible actions, for example, when working with Lego Mindstorms robots. One teacher reported:

Last year coming in, I had five students who thought about using computer science as a major. At the ending of last year, I had a total of 27. As of right now, I already have a class of at least 38, and it's not just white males anymore.

A few teachers recalled students who decided to pursue a CS major in college because of their exposure in high school. In addition, more than half the teachers overall (57.1%) noted that increased student self-efficacy was also a major success (although only one teacher from the Integration track noted self-efficacy as a success). For example, students learned that their algorithms only worked as well as they wrote them, and that they had to go through their own work multiple times to solve problems. One participant explained: "*What is successful for me is when I see a kid do something themselves without me and that grin on their face when they've solved that problem themselves.*"

When asked to describe their students' experiences learning the CS principles, a little more than one-third (37.0%) of teachers noted that the students responded most positively to the CS principle of creativity.

Generally, I would say that creativity is what they most enjoy; once they know enough about computing and about programming, it gives them the latitude to explore their creativity a lot more.

Teachers frequently discussed the tools, strategies, and supports that worked best when answering the question about their students and the CS principles. Again, a little more than one-third (37.0%) of teachers highlighted the importance of technological and human supports, in helping them successfully teach CS principles:

Just getting them to utilize the Internet more and have them exposed to using the iPads and the laptops. The kids of this generation are very tech savvy, . . . but they also need another human right there. . . . They still want someone there to say that [they] did it right. A human is the biggest resource that they could have.

The principles of programming, abstraction, and algorithms and related concepts such as analytical thinking, planning, and precision were those most frequently mentioned by teachers as challenging for their students. The principle of data was identified as challenging by three participants, all of whom were middle school teachers. This was likely due to their students' preparation: "*I think part of what's challenging with data is developmental readiness. It's a challenge to find activities to work with data for fourth, fifth, and sixth graders*."

About one-quarter (25.9%) of teachers noted that sometimes it was their lack of content knowledge that posed a major challenge. They had to learn with the students, which many found rewarding, although several explained that it was difficult keeping up with students who learned quickly.

My biggest challenge [is] keeping up with the more advanced kids. They sort of take it and run with it, and then they get to a point where they don't know how to do something . . . and I don't know either.

4. DISCUSSION

Recommendations to other teachers. When given the opportunity to discuss their recommendations to other educators, participants reiterated the importance of early exposure to CS for students and how excited students are to learn CS in the classroom. Generally, there were no differences in recommendations by grade level taught—participants teaching all grade levels indicated it is important to start young. However, there were some differences in the types of recommendations depending on teachers' prior CS content knowledge. Several of the nine teachers who stated they had very little or no prior CS knowledge suggested that other teachers jump into CS and PD, saying "just do it" or "your kids will pick it up fast and you can learn together." Those teachers who indicated they had more extensive prior CS content knowledge had more logistical recommendations for organizing a class curriculum and getting school and district administrators on board. Examples of these themes include the following:

- Don't be afraid. There's nothing that you can do to a computer besides drop it that can break it. And if you don't know something help find out with the kids. So many times the kids and I sit together and problem solve. Sometimes they find an answer, sometimes I do.
- Come up with a plan and come with a packet of resources before you go ask for support [from your administration]. All the resources as far as [pervasiveness] of CS and technical fields was very persuasive to my administration, [as were] the statistics of how if you're not exposed at the K-12 level, how much less likely you are to study it in college.

Recommendations for PD organizers. Generally, teachers indicated that they need additional support and resources with PCK-CS and strategies specific to their roles in their schools. The CS principle they mentioned needing the most additional support with was programming. Teachers who came with more background in programming desired more advanced programming or learning different programming languages, while teachers without a foundation in CS found that they need to learn programming in order to keep up with their students.

Many participants, at all levels, discussed the desire to collaborate with other professionals. These teachers indicated that hearing from other teachers what has or has not worked or collaborating on a curriculum would help them fit the CS principles into their own curricula. The following recommendations were made by two participants who felt they still needed additional support incorporating CS into their schools.

I think more PD on basically integrating more into math or science curriculum, and . . . I would love conversations with other educators that are currently doing more than what I'm doing at my school. Just because when you have that collaboration time, you brainstorm about different ideas and I would be able to pull whatever information that would be help.

. . . All of my math teachers [and] all of my science teachers have iPads in the classroom. [I would love if] somebody could come in and help us with embedding strategies using those [technology] resources.

Finally, the interviews highlighted that participants overestimated the extent to which they would have the capacity, flexibility or administrative support within their schools to implement the full CS curriculum. As a result, teachers indicated that, at times, they were unable to implement the PD in the way it was intended and instead had to adapt the material. Participants also expressed a need for collaboration and communication amongst their peers. For this reason, creating a professional learning community through blogs, Google Drive, or other platforms should be considered.

5. ACKNOWLEDGMENTS

This work is supported by a grant from the National Science Foundation (Award # 1240905). All opinions are the authors and do not necessarily represent those of the funding agency.

6. REFERENCES

[1] Barab, S., & Luehmann, A. Building Sustainable Science Curriculum: Acknowledging and Accommodating Local Adaptation. Science Education, 87(4): 454-467, 2003.

[2] Century, J., Lach, M., King, H., Rand, S., Heppner, C., Franke, B., & Westrick, J. 2013. Building an Operating System for Computer Science. Chicago, IL: CEMSE,

University of Chicago with UEI, University of Chicago. Retrieved 2015, from http://outlier.uchicago.edu/computerscience/OS4CS/.

[3] Cuny, J. Transforming high school computing: A call to action. *ACM Inroads*, 3(2): 32-36, 2012

[4] Cuny, J., Baxter, D.D., Garcia, D., Hall, S., Gray, J., Morelli, R., and Baxter, D., CS Principles Professional Development: Only 9,500 to go! *In Proc. of the 45th Annual ACM SIGCSE Conf.*, pages 543-544, 2014.

[5] Desimone, L. Improving Impact Studies of Teachers' Professional Development: Toward Better Conceptualizations and Measures. Educational Researcher, 38(3): 181-199, 2009.

[6] Goode, J., Margolis, J., and Chapman, G. Curriculum is Not Enough: The Educational Theory and Research Foundation of the Exploring Computer Science Professional Development Model. *Proc. of the 45th Annual ACM SIGCSE Conf.*, pages 493{498, 2014.

[7] Gray, J., Haynie K., Packman S., Boehm M., Crawford C., Muralidhar, D., A Mid-Project Report on a Statewide Professional Development Model for CS Principles, *Proceedings of the 46th ACM Technical Symposium on Computer Science Education*, March 04-07, 2015, Kansas City, Missouri, USA.

[8] Guzdial, M., Ericson, B., Mcklin, T., Engelman, S., Georgia Computes! An Intervention in a US State, with Formal and Informal Education in a Policy Context. *ACM Transactions on Computing Education,* v14 n2 Article 13 Jun 2014.

[9] Morelli, R., Uche, C., Lake, P., & Baldwin, L. 2015. Analyzing Year One of a CS Principles PD Project. In *Proceedings of the 46th ACM Technical Symposium on*

[10] *Computer Science Education* (SIGCSE '15). ACM, New York, NY, USA, 368-373.

[11] Price, T., Cateté, V., Albert, J., Barnes, T., and Garcia, D., 2016. Lessons Learned from "BJC" CS Principles Professional Development. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (SIGCSE '16). ACM, New York, NY, USA, 467-472.

[12] Mouza, C., Pollock, L., Pusecker, K., Guidry, K., Yeh, C-Y., Atlas, J., Harvey, T., Implementation and Outcomes of a Three-Pronged Approach to Professional Development for CS Principles. In *Proceedings of the 47th ACM Technical Symposium on Computer Science Education* (SIGCSE '16). ACM, New York, NY, USA.

[13] Yoon, K. S., Duncan, T., Lee, S., Scarloss, B., and Shapley, K., Reviewing the Evidence on How Teacher Professional Development Affects Student Achievement (PDF). Issues & Answers Report, REL 2007–No. 033. Washington, DC: U.S. Department of Education, Institute of Education Sciences, National Center for Education Evaluation and Regional Assistance, Regional Educational Laboratory Southwest, 2007.

7. Appendix 1: Interview Questions

1. What was your original motivation for joining the Partner4CS project? Did the project meet those needs? Why or why not?
2. Rate your knowledge of each of the Computer Science Principles on a 5-point scale.
3a. Since your participation in the Partner4CS, please explain how you integrated the Computer Science Principles into your classroom curriculum. Which ones did you implement? For each, did you implement them as discrete additional unit? Did you incorporate them as part of other units?
3b. What factors contributed to your decision on how to include them in your curriculum? Why did you choose these principles to implement? Why did you not choose the others? Which activities that were modeled during the PD did you use? Did you adapt them or use them as is?
4a. Since your participation in the Partner4CS professional development, please explain how you integrated the Computer Science Principles into afterschool or other special programs? Which ones did you implement?
4b. What factors contributed to your decision on how to include them in your curriculum?
4c. Why did you choose these principles to implement? And why did you not choose the others? Which activities that were modeled during the PD did you use? Did you adapt them or use them as is?

5. Which of the following strategies did you use in teaching CS principles? How did you use them? Provide an example if possible. Were these new teaching strategies for you? Explain. Strategy list: Process-oriented guided-inquiry learning (POGIL); Collaborative problem- based learning; Paired programming; Kinesthetic activities such as CS unplugged; Demonstrations that you created, adapted, or used as is; Primarily lecture style; Show and follow; students perform online activities with an existing curriculum, other.
6. Did you participate in the Partner4CS Field Experience partnership where UD undergraduates or secondary education students co-plan and co-teach computing lessons with you? If yes, was this experience helpful in developing and teaching the CS principles? Why or why not? If so, how was it helpful? If no why did you not participate?
7. Did you complete the online course to scale the Mobile CS principles PD? If yes, was this experience helpful in developing and teaching the CS principles? Why or why not? If no, why did you not participate?
8. What, if any, technological resources did you use in teaching the CS principles?
9. What, if any, curriculum resources did you use in teaching the CS principles?
10. What have been your greatest successes in teaching about the CS principles? What have been your greatest challenges?
11. How did students respond to the CS principles? What areas did they find most engaging? What topics within CS principles did they find most difficult? What tools and resources worked best for the students?
12. Is your school community supportive of teaching the CS principles?
13. What would you recommend to others who want to integrate CS principles, what would be most useful for you?
14. If you could have more PD in CS principles, what would be most useful for you?