Field Experiences in Teaching Computer Science: Course Organization and Reflections

Lori Pollock, Chrystalla Mouza, James Atlas, Terry Harvey
University of Delaware
Newark, DE 19716
{pollock, cmouza, jatlas, tharvey}@udel.edu

ABSTRACT
A major challenge for broadening participation in computing within K-12 settings is the lack of trained teachers. While professional development programs provide opportunities for the development of knowledge, skills, and pedagogy in teaching computing, teachers need ongoing support throughout the academic year. In this paper, we describe a course-based model for partnering undergraduates with teachers and students in a field experience model. We describe the model focusing on learning objectives, curriculum, field component and partnership building. We subsequently report on the products that undergraduates were able to create with their partner teachers. Finally, we investigate the impact of the field experience model on undergraduates' content knowledge, pedagogical skills and career development.

Categories and Subject Descriptors
K.3.2 Computer and Information Science Education

General Terms
Design, Human Factors, Languages.

Keywords
CS education, computational thinking, K12, CS principles.

1. INTRODUCTION
Due to efforts from the National Science Foundation, Computer Science Teachers Association, and many other groups advocating for computer science (CS) education in K-12, more states and school districts are recognizing the need for integrating computational thinking into their curricula. To provide all students with the opportunity to develop the fundamental understanding of CS principles, advocates are working on addressing many challenges. One such challenge is the lack of qualified, valued teachers in CS [5].

In the long term, the demand for trained CS teachers could be met by creating teacher preparation programs coupled with certification, robust standards, and curricula [4, 8]. However, the current demand requires multi-faceted professional development helping teachers from across varied disciplines to gain the skills, knowledge, and confidence to become excellent CS teachers [8]. Unfortunately, research shows that one-shot workshops are not sufficient to adequately train and support teachers [12].

To support the efforts of existing teachers who are working towards integrating CS principles into their classes or establishing a full CS Principles AP course, we have developed a college course-based model that engages undergraduate CS, mathematics education and education majors in field experiences in teaching CS. This model exposes undergraduates to outreach and communication skill-building opportunities that involve their technical skills, while being a part of their curriculum.

In this paper, we first detail the course model in terms of objectives, curriculum, field component and partnership building. We subsequently report on the products that undergraduates are able to create with their partner teachers. Finally, we report on the impact of the field experience model on undergraduates' content knowledge, pedagogical skills and career development. In other related work, we report on the impact of the field experience model on middle school students learning of CS concepts, programming practices, and motivation towards computing [10].

2. RELATED WORK
The Field Experiences in Teaching Computer Science course builds upon earlier efforts to broaden participation in computing such as those of the STARS alliance project [11]. The STARS Alliance is a community of regional partnership among academic, K-12 schools, professional and community groups catalyzed by the STARS Leadership Corps (SLC), a leadership program that engages undergraduates in computing outreach, research, and service to broaden participation in computing [1].

The service learning component builds upon earlier NSF supported efforts to provide university CS students opportunities to develop technical and communication skills by sharing their computing expertise with middle school teachers in the context of game design [3,6]. It also leverages key findings related to the importance of field experience to help undergraduates connect classroom experiences with university coursework, reflect upon their work, and practice their lessons in an environment where the partner teacher is present to create a controlled environment [2]. Finally, the interdisciplinary nature of the model builds upon the efforts of the multi-tiered mentoring model for increasing minority and women participation in computing proposed by the University of Alabama [3,7].

Despite building upon earlier efforts, the Field Experiences model described in this work is unique in two ways: first, it is the first effort to combine and assess all three components identified in the literature in a single model. Second, it engages undergraduates in

SIGCSE '15, March 04-07 2015, Kansas City, MO, USA
Copyright 2015 ACM 978-1-4503-2966-8/15/03···$15.00
http://dx.doi.org/10.1145/2676723.2677286

a course experience that is central to their program and can be counted towards their CS concentration requirement.

3. FIELD EXPERIENCE COURSE MODEL

The field experience course is a 1-3 credits college course open to undergraduates in CS or other STEM (Science-Technology-Engineering-Mathematics) education fields. The course provides an introduction to CS teaching methodology as well as the opportunity to put that knowledge into practice in local schools. Specifically, participants become familiar with the state of CS education in terms of pedagogy, Advanced Placement (AP) exams, and K-12 requirements while also producing written lesson plans, reflective journals, and collaborative projects. The learning objectives are as follows.

In CS education:

- be aware of challenges/strategies for teaching CS
- be knowledgeable of models and approaches to integrating CS at national, state, district, and classroom scale
- be knowledgeable of the role and impact of assessment in education
- appreciate the history of CS in constructive learning approaches

In CS curriculum:

- be aware of existing curricula for teaching CS for K-12 including the AP course on Computer Science Principles (CSP) and the Exploring Computer Science (ESC) course
- be able to explain core ideas in each of the 7 CS principles areas
- have functional skills in using software tools to express/model K-12 CS problem solving skills

In teaching:

- be knowledgeable of education standards such as Common Core State Standards (CCSS) and how CS curriculum ties into these standards
- be confident observing and evaluating teaching of CS lessons
- be confident assisting a classroom teacher in the implementation of CS lesson plans
- be confident developing, planning, and assessing a CS lesson plan
- gain experience leading the implementation of CS lesson plans
- be able to collate a set of lesson plans to form a CS module

In communication:

- collaborate effectively with a team of education specialists
- be able to present the argument for CS education in K-12
- become more reflective on teaching, planning, and educational research
- gain experience in public speaking with multiple audiences

Table 1. Semester–long Activities in the Field-Experience Course

Week	Computational Activities	Pedagogical Activities	Field-Related Activities
1-3	Learn CS education trends and challenges; Explore online CS resources (e.g., CSTA, CS unplugged and ECS)	Initial self-assessment on confidence in teaching computing; Share experiences in teaching, learning CS and expectations; Discuss giving feedback.	Identify and teach a CS unplugged exercise to peers in preparation for teaching a similar activity to middle/high school students; Give and receive feedback.
4-5	Continue to explore and think about adapting CS resources (e.g., CSTA, CS unplugged and ECS) for middle school students	Discuss effective learning environments; Learn about each school's context and student population; Discuss middle school learning environments; Discuss ways to interact with teachers and students.	Assign field locations; Address transportation issues; Identify appropriate dress code and behaviors within school settings; Conduct a class observation to establish rapport with partner teacher and students.
6-7	Explore Scratch exercises and develop Scratch projects for showcasing to students.	Prepare, discuss, and analyze lesson plan proposals for after-school program.	Observations and co-planning sessions with partner teachers; Discuss field observation experiences, pedagogical strategies used by partner teachers, and anticipated challenges.
8-10	Present Scratch projects to peers; Give/receive feedback; Discuss potential Scratch teaching issues. Discuss technical problems emerging from field lesson plans and propose solutions.	Explore classroom management techniques; Identify a reward system; Discuss what skilled teachers think and do based on field observations; Present lesson plans for after-school program; Give/receive feedback on lesson plans.	Co-plan and co-teach with partner teachers; Weekly updates on participants' experiences in partner schools.
11-13	Create flyers for raising teachers' awareness of CS education to broaden students' and teachers' participation in computing.	Teamwork on generating ideas for the Partner4CS summer professional development workshop based on experience working with partner teachers.	Co-plan/co-teach with partner teachers; Discuss/share what works well and what needs improvement in field teaching; Discuss issues observed in the field (e.g., strategies for responding to students' questions, reinforcing understanding, etc.)
14	Final reflection on overall experiences in teaching and service as well as impacts on future learning and career development; Final self-assessment on confidence in teaching computing.		

3.1 Field Experience Course Activities

The course combines college classroom meetings with visits in middle or high school classrooms where undergraduate participants co-plan and co-teach computing lessons with a practicing teacher. The 75-minute on-campus class time is devoted to identifying and implementing CS teaching resources targeted toward CS principles, modeling classroom lessons with CS unplugged and other resources, discussing teaching pedagogy, preparing and analyzing lesson plans, and reflecting on the field experiences and teacher partnering experiences.

The field experiences take place in local middle and high schools where teachers are working on integrating CS principles into their courses and after school programs. Undergraduate participants in the course meet with teachers to discuss their role and plan lessons and activities, lead after school programs, and lead interactive classes and project work during the school's designated class times. Table 1 depicts the course schedule of activities by week, separated into computational, pedagogical, and classroom field-related activities.

3.2 Field Experience Logistics

Partnerships take time to establish. To get the course launched the first semester, we identified two partners, each from different local schools, and each identified through existing relationships with one of the authors who regularly goes into middle schools as part of her educational research projects. A pre-existing relationship helped us to work with teachers who trusted us to prepare the students for the classroom field experience and we knew were anxious to receive support knowing some logistics were not ironed out in the course yet. In following semesters, the partnering teachers had been attendees in our summer CS Principles Professional Development workshop. The week-long workshop experience with undergraduate teaching assistants demonstrates to the teachers the potential support that undergraduate CS and STEM education majors can provide.

After background checks are initiated, pairs or teams of undergraduates are matched with a local teacher for the whole semester based on availability, transportation capacity, and preference. Each participant is expected to go in the field for 1-2 hours per week. Specific schedules as well as CS content focus are identified through initial meetings with the partner teacher.

Throughout the semester, all undergraduate participants maintain a reflective journal on their experiences. Reflection is key to connecting their university experience with their experience in the field, identifying challenges experienced in the field, and thinking about their learning and future actions to improve their partnership connections. A typical reflective entry describes the field activities enacted by each participant during the week, identifies successes and challenges, and presents thoughts for future action (e.g., discuss aspects of the lesson that did not work well with the partner teacher, rethink components of the lesson, rethink approach of providing feedback to students, etc.).

3.3 Student Participation and Assessment

Students can register for 1-3 credits of this course, which determines the amount of field experiences they are required to complete each week. The prerequisites are minimal and include successful completion of at least one prior course in CS. This provides the knowledge needed to prepare lessons at the targeted level, while opening the opportunity to CS minors and other STEM majors that typically only take one course in CS. To encourage education majors to register along with computer science majors, the course is cross-listed as a CS and an Education course. During the three semesters in which the course was offered, however, the majority of the participants were undergraduates in CS. A smaller number of participants came from biomedical engineering and mathematics.

Course grades are based on the quality of a student's contribution in the following areas:

- 30% Journal of student reflection on their learning, readings, and field experiences
- 20% Portfolio related to field work
- 20% In-class participation
- 20% Mock teaching, presentations, and leading activities during the college classroom meetings
- 10% Final Reflective Essay

4. METHODOLOGY

4.1 Research Questions

In this paper, we focus on answering the following research questions with regard to the field experience course model:

1. What kinds of CS activities can undergraduates design and facilitate in their field placements?

2. What is the impact of the field experiences model on undergraduate student participants regarding their content knowledge, pedagogical skills and own career development?

Specifically for this work, we examined what worked well during the three semesters in which the field experience course was offered and what, if anything, should be changed to increase the impacts of the course on undergraduates and their targeted students.

4.2 Data Collection and Analysis

To answer our questions, we collected data using several sources. First, all undergraduate participants were observed during their visits to our partner schools. Each observation included information on the role of the partner teacher, the role of the undergraduate participant, and the lesson itself including introduction, materials, and CS concepts addressed. A short debriefing session with the participants accompanied each field observation to ensure that the observer understood the purpose of the activities. Further, all teaching materials created by undergraduates throughout the duration of the project were collected. These data helped us examine the CS activities designed and implemented by undergraduates in the field.

In addition to course observations, all reflective journals maintained by undergraduates were collected and examined. A total of 200 journal entries were collected throughout the three-semester period. Although the length of each reflective entry varies from week to week and from participant to participant, the average length of each entry was 400 words.

Additionally, at the end of each semester, all undergraduate participants responded to a written questionnaire, which asked them to consider the key benefits of the field experience course for their own learning, their ability to communicate computing concepts to students, and their future career goals.

To answer our two research questions we analyzed the data qualitatively. We first read all classroom observations and reflection entries and summarized the key CS activities designed

and implemented by undergraduate participants in their partner schools. We also read participants' reflective entries and identified excerpts in which they discussed the impact of the field experience course on their own learning of content and pedagogy. As analysis for each participant was completed, data were compared to those of other participants to identify similarities and differences as well as emerging themes [9]. Data from written questionnaires as well as course and field observations were used to triangulate findings and accept or dismiss emergent themes.

4.3 Participants

To date, we have offered the field experiences course three times, spring 2013, fall 2013, and spring 2014. The course is being offered each semester to continue the partnerships with the schools throughout their academic year. Table 2 shows participants over time.

Table 2. CS and K-12 Participants

Semester	CS participants	Partner Teachers	G6-12 students reached
Spring 2013	7	2	80
Fall 2013	9	3	30
Spring 2014	8	3	80

5. FINDINGS

5.1. CS Activities Designed and Implemented by Undergraduates

During the three semesters in which the course was offered, students helped plan and lead after school computing programs once a week for two different middle schools: a sub-urban charter school serving approximately 1,350 students in grades K-8 and a local independent school serving approximately 100 students with identified learning, attention, social/emotional, and/or behavioral issues that may impact school success. During those times, undergraduate participants engaged students in a series of programming exercises using Scratch – a programming environment that allows students to design and create interactive programs (or games) using graphical blocks. Table 3 shows the activities designed and implemented by CS undergraduates during the after school programs.

In addition to planning and leading after school programs, undergraduate participants worked collaboratively with classroom teachers to adapt lesson plans and activities from available resources (CSP, ESC, Code.org) and lead classroom sessions in high school engineering classes integrating CS principles into existing STEM curricula. In particular, during fall 2013 and spring 2014, undergraduates worked with a high school teacher to plan and enact three CS modules on (1) website design, (2) human-computer interaction and (3) programming.

Table 3. After-school Program Activities Designed and Implemented by CS Undergraduates in Local Schools

	After School Program Activities (Charter School)	After School Program Activities (Independent School)
Week 1	1. Introduction to Scratch. 2. Basic blocks: Motion, Pen Up/Pen Down (Pen) Repeat (Control), Sequence.	1. CS unplugged: Computers do what you tell them to do. Exercise on the importance of specifying *exactly* what computer must do. 2. Introduction to Scratch and basic blocks.
Week 2	1. CS Unplugged (Security and Message Broadcasting): Information hiding: Students calculate their average age without anyone having to reveal to anyone else what their age is. The facilitator announces the average age. 2. Introduction to Broadcasting. 3. Modeling Pair Programming.	1. Introduction to conditionals with daily life examples; "If-then, else" unplugged exercise. Students were asked to raise their hand if they played a sport. If more than half of the people in the room raised their hands, they were supposed to sing, "I'm a little teapot". 2. Intro to Variables (Operators, inputs) with an example of guessing game.
Week 3	1. Broadcasting: Communication with multiple sprites 2. Introduction to Hide/Show blocks 3. Mini Story Project: Creating a short story with 5 slides and 3 sprites that interact with one another	1. Introduction to Loop (forever, repeat…until) and activities. Unplugged exercise: Draw the letter X once on each line of a piece of paper until you run out of lines. 2. Designing a guessing game.
Week 4	1. Reviewing Scratch Concepts: move, turn, broadcast, show/hide, say, play. 2. Intro to Conditionals and Sensing with examples. 3. Work on Mini Story Project: Students worked on their products adding elements related to new CT constructs.	1. CS unplugged for Coding/Pseudo-coding: Peanut butter and Jelly (PB&J) exercise (pseudo-code): Students gave step-by-step instructions on making a PB&J sandwich. Subsequently, each student wrote down steps on papers, made one PB&J sandwich and tried to improve the steps.
Week 5	1. Students continued to work on their projects: Sharing successes and challenges.	1. Designing/programming point-and-click adventure game in a team: Brainstormed game design and created a storyboard.
Week 6	1. Intro to Variables (Operators, inputs) with examples. 2. Guessing game project or calculator project: Students chose one as their second project.	1. Continued to work on programming point-and-click adventure game in a team: Students improved their program flow, storyline and sprites; started making their program using conditionals, loops, algorithms, event handling, logical statements.
Week 7	1. Students continued to work on their second project: College students pseudo-coded the instructions to help them think through the project.	1. Continued to work on programming a point-and-click adventure game in a team: Reinforcing some concepts and introducing applicable blocks; testing and debugging.
Week 8	1. Complete game design & demo day.	1. Complete game design.

During the first unit (fall 2013) on website design, CS undergraduates helped high school students understand what makes a good website following a website evaluation rubric adapted from the ECS curriculum unit on web design. Subsequently, they helped them gain an understanding of the process of website design including storyboarding. In the third week into the unit, they introduced high school students to HTML, guided them through a tutorial and discussed how to create a basic website with HTML. They also discussed good practices in HTML coding. As a class, students worked on creating a website that incorporated data and research related to a science project they had just completed in class. Table 4 demonstrates the sequence of this module.

During the second module (spring 2014) on human-computer interaction, CS undergraduates guided high school students through a series of activities spanning a period of 5 weeks, focusing on understanding good interface design. To introduce the topic, CS undergraduates started with a CS unplugged activity (Chocolate factory activity) focusing on interface design practices away from the computer. Subsequently, they discussed forms of human input, such as touch-screens, buttons and the design of modern games including flappy birds, angry birds and fruit ninja. Subsequently, students explored advanced game design techniques by using the programming tool Scratch to create their own versions of flappy birds.

Table 4. Description of HTML Module

Week(s)	Activities
1	What makes a website good: rubric criteria for evaluating websites, examining sample websites
2	Storyboarding: What is it? How do we use it? What should a storyboard for a website contain?
3-4	Introduction to HTML: How to create content for the web, HTML tutorial, CSS tutorial, creating a basic website with HTML and CSS
5	HTML Lessons, good coding practices, Networking concepts, wrap up.

Finally, the third module (spring 2014) on programming was also designed and implemented over a 5-week period and engaged high school students in the design of their own version of flappy birds using Scratch. A unique feature of the game was the use of the webcam and the motion sensor in Scratch which enabled the program to sense motion and respond accordingly.

5.2. Impact of the Field Experiences Course on Undergraduate Participants

Findings from our work indicated that undergraduates became more confident in their CS skills, in their ability to communicate technical information to students and in their pedagogical skills (e.g., moving away from the role of dispenser of information).

Many of the participants, for instance, had not worked with Scratch before, so learning the program's interface was a new skill for them. Even those who had prior experience working with Scratch appreciated the opportunity to learn more. As one participant noted, *teaching the students how to code in Scratch helped me to solidify my knowledge of the basics of programming; so the students were not the only ones learning*.

Besides learning skills specific to Scratch, participants noted various instances in their reflective journals where they improved their content knowledge as a result of their interactions with classroom students. One participant, for example, noted that he was often caught off guard with students' programming solutions because they used approaches different than what he would have implemented. Similarly, another participant noted in one of her entries:

> *One of the students I helped was trying to get a ball to increase speed over time. I knew how to increment speed, but couldn't actually think how to do it based on the amount of time that had passed. Initially, the student had a loop that contained a block with a wait followed by an increment to a time counter. This caused the entire program to wait. I explained what was happening and the student removed the wait block. After that, I explained that the wait time was incremented every time he went through the loop, which was based on the code running rather than external time. I explained how usually, in programming, there was a way to assess how much time had passed while a program was running. That was when the student had the idea of creating a separate script, which started at the same time, and would wait, and then add to his variable. That was a solution that had not occurred to me.*

More importantly, however, participants acquired increased confidence in their content knowledge and often solidified their own understandings as a result of their interactions with teachers and students. One participant, for example, explained that as a result of helping his partner teacher teach the HTML module, he was more prepared for his Advanced Web Technologies course.

Additionally, findings from reflective journals provided ample evidence that close collaboration with experienced teachers created a reciprocal partnership conducive to learning – while undergraduates provided their content expertise, practicing teachers shared their pedagogical expertise. One participant, for example, wrote about feeling proud of offering her content expertise during her school visits, particularly at times when the teacher could not handle students' content questions independently and directed students to her. In their reflections, however, undergraduate participants noted a number of instances where they acquired important pedagogical skills from their partner teachers, particularly related to classroom management and pedagogical interactions with students. One course participant, for example, highlighted how he learned about appropriate times for providing one-on-one assistance versus whole class instruction. He wrote: *If a student has a problem that seems to be trending among the class, the teacher will pause the lesson to address the issue as a whole class. If it is more of an isolated problem, she provides individualized instruction.* Similarly, another undergraduate noticed the way in which his partner teacher provided feedback to her students by writing: *If they were wrong, she would not just tell them they were wrong; she would say something like, good try, but try thinking about it like this…really motivating them to try again and think harder about it.* Many students later adopted pedagogical strategies modeled by their partner teachers in their own efforts.

A key finding was related to participants' ability to explain concepts to students without directly giving out the answers. At the beginning of the semester, this presented a challenge as one participant reflected:

> *I realize how difficult it is to help students formulate their own ideas instead of just telling them what they need to do. I want to be able to help them effectively, but I want them to have the excitement and satisfaction that comes from*

figuring the problems out for themselves.

Similarly, other participants also pointed out the need to scaffold student learning without directly giving out answers.

Finally, many participants indicated that participation in the field experiences model directly improved their ability to communicate with diverse audiences, a skill they identified as critical for their own career development. One participant explained:

> *Often, I needed to answer questions on the spot, which included conceptual questions. This greatly increased my ability to communicate computing ideas. I am much more effective at sharing my projects as a result of the course.*

Similarly, another participant explained:

> *One of the benefits of the course was the interaction with classroom teachers. In my career, I need to be able to work with other people in the context of what they need, and the opportunity to talk to others was really helpful for my ability to speak to people.*

6. DISCUSSION AND CONCLUSION

In this work, we discussed the potential of a field experiences model to share undergraduate computing students' knowledge and skills with middle and high school teachers and students. Findings indicate that the experience has positive impacts on undergraduate participants in terms of technical skill confidence, communication skills, and knowledge of good CS teaching pedagogy, which may help with future mentoring opportunities. Further, the final products – lesson plans and activities - created by the undergraduate participants were all used in the field and documented with teacher feedback. The partner teachers continue to seek our support, another indication that the undergraduates are indeed able to create and enact these activities in the field.

In terms of future directions, findings from our work revealed three components that were key to the success of the field experience model but presented room for improvements.

- Trial lessons on teaching CS unplugged: Many students noted that the opportunity to rehearse the teaching of CS unplugged activities early in the semester was a valuable experience. It allowed them to witness firsthand the preparation and planning needed to lead a lesson, the teaching practices of their peers, and how to give and receive feedback in a safe environment. In the words of a student, these experiences helped him transform from a "capable speaker" to "confident classroom leader". Moving forward, it might be important to offer additional opportunities for teaching and modeling within the context of the college meetings.
- Learning theory: At the beginning of the course, we provide an overview of theory related to how people learn and the design of learning environments. Findings indicated that participants appreciated the opportunity to learn theoretical ideas that they could later apply to practice. Participants, however, indicated that they would like additional support in this area and suggested that the course include more opportunities to address issues related to how students learn.
- Lesson development: Participants found the experience of developing and implementing lessons valuable and rewarding. Nevertheless, some participants noted that they were not prepared for the amount of time it required for them to develop or adapt classroom materials for their field placement. As some participants noted, they thought the partner teacher would be taking a more active role in identifying the CS content school students needed. Partner teachers, however, often looked up to the undergraduates making those decisions. In future offerings of the course, we need to provide additional supports as undergraduates take on this leadership role.

In conclusion, our findings indicate that the field experience model holds promise for the successful infusion of CS content in middle and high school classrooms while simultaneously providing numerous benefits to CS undergraduates. Our next challenge is scaling up the model to involve more partners.

7. ACKNOWLEDGEMENTS

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.

8. REFERENCES

[1] Barnes, T. Dahlberg, T., Buch K., & Bean, K. 2009. The STARS Leadership Corps: An innovative computer science learning community. *Learning Communities Journal*, 1:5–18, 2009.

[2] Blue Ribbon Panel Report on Clinical Preparation and Partnerships for Improved Student Learning. 2010. *Transforming teacher education through clinical practice: A national strategy to prepare effective teachers*. NCATE.

[3] Burns, R., Harvey, T., & Pollock, L. 2012. An Experience Report on Cross-Semester Student Critique and Action in an Integrated Software Engineering, Service Learning Course. In *First International Workshop on Software Engineering Education based on Real-World Experiences (EduRex)*.

[4] 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. http://outlier.uchicago.edu/computerscience/OS4CS/.

[5] Goode, J., Chapman, G., and Margolis, J. 2012. Beyond Curriculum: The Exploring Computer Science Program. *ACM Inroads, 3*(2), 47-53.

[6] Gray, J., Shaundra, D., MichaelWyss, M., and Cotton, S. 2012. Building a K-12 Computing Pipeline in Alabama that Addresses Participation Diversity. NSF Showcase: SIGCSE.

[7] Johnson, D., MichaelWyss, J., Gray, J., Daily, S., Shih, A., & Abbott, G. 2012. Broadening Participation in Computing: The *Multi-tiered Approach. ACM Southeast Conference*.

[8] Lang, K., Galanos, R., Goode, J., Seehorn, D., Trees, F., Phillips, P., and Stephenson, C. 2013. *Bugs in the System: Computer Science Teacher Certification in the U.S.,* ACM CSTA.

[9] Miles, M. B. and Huberman, A. M. 1994. *Qualitative data analysis: An expanded sourcebook*. Sage.

[10] Mouza, C., Pan, Y., Pollock, L., Atlas, J., & Harvey, T. 2014. Partner4CS: Bringing Computational Thinking to Middle School through Game Design. *FabLearn*, Stanford University, CA. http://fablearn.stanford.edu/2014/wp-content/uploads/fl2014_submission_21.pdf

[11] STARS. STARS Alliance, http://www.starsalliance.org, 2011.

[12] Zepeda, S. 2012. *Professional Development: What Works*, 2nd Ed. Athens, GA: Eye on Education, Inc.