WHAT DO SECONDARY COMPUTER SCIENCE TEACHERS NEED?

EXAMINING CURRICULUM, PEDAGOGY, AND CONTEXTUAL SUPPORT

Olgun Sadik

Submitted to the faculty of the University Graduate School
in partial fulfillment of the requirements
for the degree
Doctor of Philosophy
in the School of Education,
Indiana University
July, 2017

Accepted by the Graduate Faculty, Indiana University, in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

Doctoral Committee

_____

Thomas Brush, Ph.D., Co-chair

_____

Anne T. Ottenbreit-Leftwich, Ph.D., Co-chair

_____

Barbara Dennis, Ph.D.

_____

Kyungbin Kwon, Ph.D.

Date of Dissertation Defense – December, 14, 2016

ACKNOWLEDGMENTS

I would like to dedicate my dissertation to my family in Bloomington and Turkey. I am grateful for their support and understanding along the way. To my parents, Mualla and Kemal, Sadık, I can't express enough my appreciation of the nights and days you spend taking care of and supporting me. Thanks to my brother, Onur Sait Sadık, his wife Aslı Sadık, and my sister, Aleyna Sadık, for their encouragement and love. I would like to thank my mother-in-law, Emine Ozcelik, for flying thousands of miles from Turkey to take care of us during my dissertation work.

I also would like to thank my country, the Republic of Turkey, and the Turkish people for supporting my doctoral studies with the Ministry of Education's scholarship for Master's and Doctoral studies.

Without the help of many colleagues at Indiana University this dissertation would not have been possible. My dissertation co-chairs, Dr. Thomas Brush and Dr. Anne Todd Ottenbreit-Leftwich, were generous with their time, advice, and understanding at every step, and committee members, Dr. Barbara Dennis and Dr. Kyungbin Kwon, gave me invaluable feedback throughout the process. My director, Dr. Cindy-Hmelo Silver, and my colleague, Janis Watson, in the Center for Research on Learning and Technology, provided me with help and support throughout the process, and my friends and colleagues, Bryan Hoey, Carly Patterson, Funda Ergulec, Matthew Callison, Meral Kucuk Yetgin, Soner Yetgin, Yasin Ramazan Basaran and Yahya Han Erbas in Bloomington supported me and shared my stress along the way. The executive board members of the Computer Science Teachers Association as well as its executive director, Dr. Mark Nelson, kindly supported my work by allowing me to conduct this research with their members.

And my sincere gratitude goes out to my study participants, who spent their time and shared their encouragement for completing the study.

And finally, special thanks and love to my wife Ozlem Sadık and my little daughters, Hazel and Gulce. Thank you for greeting me every evening with love, and sharing this beautiful life and success story with me. I love you so much.

Olgun Sadik

WHAT DO SECONDARY COMPUTER SCIENCE TEACHERS NEED?

EXAMINING CURRICULUM, PEDAGOGY, AND CONTEXTUAL SUPPORT

The primary purpose of this study was to identify secondary computer science (CS) teachers' needs, related to knowledge, skills, and school setting, to create more effective CS education in the United States. In addition, this study examined how these needs change based on the participants' years of teaching experience as well as their background in CS.

Participants were selected from secondary teachers who were teaching CS content in a school setting (public, private, or charter) or an after-school program during the time of data collection (between 2013 and 2016). This study followed a mixed-method research design using both qualitative and quantitative methods. The data were collected from email listserv discussions, questionnaire responses from 222 secondary CS teachers, and interviews with eight secondary CS teachers. Descriptive statistics, thematic analysis, and content analysis were used to analyze the data.

Findings report the most common needs identified by teachers with regard to effective CS teaching. Participants reported the need for updating their course(s)' curriculum resources constantly to keep up with changes in CS education practices and national standards. When CS was offered as an elective in their schools, teachers emphasized the need for increasing student enrollment. On the other hand, when CS was a required course in schools, teachers reported a high number of students who showed little interest in learning CS. Different pedagogical needs were discussed in different contexts. The most common pedagogical need expressed was learning student-centered strategies for teaching CS and guiding students' understanding with the use of scaffolding and team-management strategies. Administrators and parents were reported as

the main stakeholders in CS education practices, and teachers reported improving these stakeholders' understanding of CS as a need to increase collaboration toward improving CS education practices in their schools. Many secondary CS teachers also shared the need for a community of CS teachers in their district. Even though resources for computer labs (equipment, software, and network) did not appear as a common need, a group of teachers did list resources as a crucial need in their teaching context. Findings also revealed that secondary CS teachers have a variety of different backgrounds and years of teaching experience, and teachers reported different needs based on their experience.

_____

Thomas Brush, Ph.D., Co-chair

_____

Anne T. Ottenbreit-Leftwich, Ph.D., Co-chair

_____

Barbara Dennis, Ph.D.

_____

Kyungbin Kwon, Ph.D.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I: INTRODUCTION

In order to ensure the quantity and quality of people in computer science (CS), scholars have emphasized the value of including CS education at the K–12 level (Barr & Stephenson, 2011; Hug, Guenther, & Wenk, 2013). Starting CS education in K–12 could positively influence students' attitudes and encourage them to consider a career in CS (Downes & Looker, 2011; Hug et al., 2013). K–12 experience in CS could not only help develop future computers scientists, but also help students develop a variety of skills that may be useful in professional and everyday life.

Providing CS within a K–12 environment allows students to experience the challenges that professionals face as well as learn skills important for their career and life in general. For example, with CS knowledge in early stages of their education, students could use programming platforms to code their own games and become producers of technology (Overmars, 2004). Computer programming help students improve their critical thinking and problem-solving skills (Fessakis, Gouli, & Mavroudi, 2013). Furthermore, students transfer these skills to other work settings such as accounting and business (Kavanagh & Drennan, 2008), education (Bailin, 2002), nursing (Adams, 1999), and medical science (Tiwari, Lai, So, & Yuen, 2006).

The need for K–12 CS education has been emphasized more after the report titled *Running on Empty: The Failure to Teach K–12 Computer Science in the Digital Age* (Wilson, Sudol, Stephenson, & Stehlik, 2010). The report highlighted the issues associated with achieving this goal such as lack of available CS courses, inadequate teacher preparedness and development, and lack of standards and curriculum in K–12 CS education. Various public and private organizations have started to make efforts to solve these problems and have helped develop successful K–12 CS education in the US. For instance, the following organizations have developed well-known CS standards and frameworks for K–12:

- Association for Computing Machinery, Code.org, Computer Science Teachers Association, Cyber Innovation Center, and National Math and Science Initiatives' K-12 Computer Science Framework (K12CS, 2016),
- Computer Science Teacher Association's (CSTA) K–12 Computer Science Standards (CSTA, 2016),
- International Society for Technology in Education (ISTE)'s Standards for Computer Science Educators (ISTE, 2011).

Code.org (2013) released their campaign with Mark Zuckerberg and Bill Gates that promoted the need for K–12 CS education in the US education and professional worlds, and aimed to provide opportunity to learn CS for all the students. National Science Foundation (NSF) participated in these efforts by providing funding for research projects and started CS10K grant program for research institutions to train 10,000 teachers to teach CS courses for 10,000 schools (Kalil & Jahanian, 2013).

Even though these efforts have helped increase the number of CS classes in K–12, it is apparent that integrating CS in these schools is a systemic change (Barr & Stephenson, 2011). Systemic changes in education require well-prepared teachers and supportive environments (Reigeluth, 2011; Stephenson, Gal-Ezer, Haberman, & Verno, 2005). However, due to limited certification programs for teacher preparation in CS education (Robelen, 2013; Wilson et al., 2010), teachers with inadequate teaching and/or CS content knowledge/skills were assigned to teach CS courses in K–12 (Veletsianos, Beth, & Lin, 2016; Cimons, 2010; Ericson et al., 2008). Furthermore, some schools do not have the necessary resources or support for CS teachers to be successful in their teaching (Barr & Stephenson, 2011).

Problem Definition

There are two main pathways to prepare teachers: 1) teacher education (TE) and 2) professional development (PD). Teacher education refers to the regulations and practices that are organized to prepare pre-service teachers to effectively teach in their future classrooms. PD refers to the educational activities teachers participate in to improve their knowledge, skills, and attitudes in a certain area; PD is typically implemented with in-service teachers who are already in the classroom. However, there have been problems related to both CS teacher education and professional development (Barr et al., 2013; Ericson et al., 2008; Gal-Ezer & Stephenson, 2009; CSTA, 2013).

Problems in CS Teacher Education

Preparing CS teachers in teacher education is crucial to improve the quality and quantity of CS classes in K–12 schools (Ragonis, Hazzan, & Gal-Ezer, 2010). Teacher education programs are designed to provide preservice teachers theoretical and practical experiences (Darling-Hammond, 2010). However, in most states there have been very few well-defined CS education programs in teacher education institutions (Barr et al., 2013; Wilson et al., 2010).

According to a 2007 U.S. survey, only 53 % of states required "some level" of CS teaching endorsements from a higher education institution to teach CS (Khoury, 2007). According to the CSTA (2013) report, only 17 states required some kind of teaching licensure or endorsement for middle-school CS teachers. In the same report, even though more states (23) required CS certification programs for high-school CS teaching, the certification has been from other areas such as "Computer Science, Business, Mathematics, Technology Education, Fine and Practical Arts or Library Science and in various CTE areas" (Barr et al., 2013, p. 21). Therefore,

teachers who had certifications for these programs were able to teach CS courses but might not have the necessary preparation to teach CS.

Furthermore, in the states where certification or endorsement existed (23), there has been confusion regarding the scope of CS education (e.g., definition of CS and the courses offered) (Wilson et al., 2010). In some states, the teacher education programs provided to preservice CS teachers were not connected to the CS field (Barr et al., 2013; Khoury, 2007), but instead focused on the integration of technology into different subject areas or how to use computer applications (e.g., Microsoft Office, keyboarding). When asked to define computer education or computer science education in K–12, responses varied. For instance, school administrators and state officials used the terms *technology in education*, *technology literacy*, and *information technology*, but Barr and colleagues (2013) have argued that these were significantly different from *computer science*.

CS is an academic discipline that requires knowledge of how computers work. ACM's *Model Curriculum for K–12 Computer Science* defined computer science as "an academic discipline that encompasses the study of computers and algorithmic processes, including their principles, their hardware and software designs, their applications, and their impact on society" (Tucker et al., 2003, p. 6). According to *ISTE Standards for Computer Science Educators* (ISTE-CSE) (ISTE, 2011), CS pre-service teacher education should include content knowledge of data representation, abstraction, algorithms, programming, digital devices, systems, networks, and role of CS play in the modern world.

Problems in In-service CS Teacher Professional Development

CS education in K–12 has been changing rapidly due to the dynamic nature of CS content (Gal-Ezer & Stephenson, 2009; Ragonis et al., 2010; Stephenson et al., 2005). This requirement

for constant change was evident in various CS organizations, K–12 standards, curriculums, and resources in the last five years, including ISTE-CSE (ISTE, 2011), K–12 Computer Science Standards (CSTA, 2011, 2016), and CS Principles (College Board, 2016). These new standards led to a variety of challenges for in-service CS teachers in K–12 such as updating their pedagogical and content knowledge/skills, curriculums, and resources regularly (Ragonis et al., 2010).

For teachers to succeed, Avalos (2011) stressed that they must seek constant professional development opportunities to keep up with the ever-changing nature of their field. For CS teachers, PD programs should be able to improve teacher CS content and pedagogy knowledge/skills, and allow teachers to transfer the outcomes of the PD to students' learning (Fishman, Marx, Best, & Tal, 2003; Menekse, 2015). Furthermore, successful PD programs should be designed based on the participant teachers and their students' needs (Garet, Porter, Desimone, Birman, & Yoon, 2001). However, due to top-down approaches with content determined by academics and administrators, the real challenges and ongoing teacher needs are usually ignored in PD programs (Baird & Rowsey, 1989; Owston, Wideman, Murphy, & Lupshenyuk, 2008). Therefore, programs that were not aligned with teachers' challenges were limited in satisfying their needs and prevent the transference of new knowledge and skills to real classroom environments and, in turn, to students' learning (Pehmer, Gröschner, & Seidel, 2015; Schlager & Fusco, 2003).

Purpose

Based on problems coming from current CS teacher education programs, top-down in-service professional development strategies, and the constantly changing nature of CS education content, curriculum, and instructional methods, it is important to closely examine in-service

secondary CS teachers' needs for effective CS education in the US. Although previous research has recommended beginning CS education as early as kindergarten (Clements & Gullo, 1984; Fessakis et al., 2013), the present study focused only on secondary CS teachers, due to the limited number of CS teachers and courses at the K–5 level. In this study, secondary education refers to both middle- and high-school (grades 6–12). It is assumed that by identifying needs in this grade range, this study will:

1. address specific needs of secondary CS teachers,
2. inform administrators and scholars as they develop data-driven professional development programs and resources, and
3. inform teacher education programs about in-service secondary CS teachers' needs and assist them in preparing courses that address those needs.

Data related to the needs of secondary education CS teachers were captured through email communications in a listserv, surveying and interviewing teachers.

Furthermore, this study also examined how CS teachers' needs change as they gain experience teaching CS (e.g., beginner teachers versus experienced teachers) and according to the type of training (e.g., majored in CS, obtained CS teaching certification, no former CS training etc.) they received in CS. In other words, it is assumed that years of CS teaching experience and teachers' previous background in CS have influence on their needs (Melnick & Meister, 2008).

According to Organization for Economic Co-operation and Development (OECD) (2005), supporting beginning teachers should be a high priority in educational research. Beginning teachers have struggled with classroom management, long-term planning, accessing and organizing resources, differentiating for students with special needs, and accessing

professional development, as well as communicating with colleagues, administrators, and parents (Fantilli & McDougall, 2009). On the other hand, experienced teachers who have been teaching CS courses for many years have had to keep up with the ever-changing field of CS content knowledge. In terms of background in CS—depending on the state—the types of training and/or certification varied widely in in-service CS teachers … and some have had no previous CS training at all (CSTA, 2013). As previously stated, in most states there have been no standardized CS education programs in teacher education institutions (Barr et al., 2013), and teachers with little or no CS knowledge have been able to teach CS courses in K–12 (Ericson et al., 2008).

Research Questions

This study seeks to answer the following research questions:

1. What needs do U.S. secondary school computer science teachers share related to their knowledge, skills, and school settings?
2. How do teachers' needs vary based on years of CS teaching experience and background (e.g., education, training) in CS?

Overview of the Methods

This study followed a mixed-method research design using both qualitative and quantitative methods (Creswell & Clark, 2007). Using both the qualitative and quantitative methods offered a comprehensive understanding of research problems (Fraenkel & Wallen, 2010). This study contained both exploratory and explanatory design characteristics. In terms of exploratory purpose, initially qualitative email listserv data was used to develop a CS teachers' needs framework and to inform the development of a questionnaire based on secondary CS teachers' needs identified in their discussions. In terms of explanatory purpose, quantitative data (questionnaire), and qualitative data (open-ended items in the questionnaire and follow up

interviews) were used to gain additional information and amplify the results (Fraenkel & Wallen).

For the 1st phase of the study, ISTE-CSE (ISTE, 2011) standards and teaching research literature between 2005 and 2015 were used to create preliminary categories of general teacher needs. These categories were then used to analyze the data in the Computer Science Teachers Association (CSTA) email listserv between January 1, 2013 and January 1, 2014 (one year) using content analysis technique (Weber, 1990). In this first phase, some preliminary categories were removed, others were combined, and new categories emerged from the data. The results of the analysis informed the creation of the computer science teachers' needs (CSTN) framework for studying their needs. The term "CSTN framework" will be used throughout the rest of the study when referencing this framework, which included six categories 1) pedagogical needs, 2) curricular needs, 3) student related needs, 4) professional knowledge and skills needs, 5) resource needs, and 6) stakeholder-related needs.

In the 2nd phase, the researcher used the CSTN framework categories to analyze and classify the content of the email communications between January 2013 and October 2015 (~3 years). The data comprised 1,706 emails. Following the content analysis, data under each category were analyzed using thematic analysis technique (Braun & Clark, 2006) to identify CS teacher needs and to develop a questionnaire from their communications.

In the 3rd phase, conducted during the fall of 2015, the questionnaire was disseminated to 7,356 CS teacher members of CSTA through their email addresses. At the end of the questionnaire, teachers were asked to participate voluntarily in a follow-up, semi-structured interview (4th phase), in order to elaborate upon their needs with examples from their practices.

The interviews were conducted in the spring 2016 semester with eight CS teachers who were purposefully selected from the phase 3 participants based on their questionnaire responses. The qualitative interviews helped the researcher to provide more in-depth context and explanations of the data with thick descriptions and examples (Creswell & Clark, 2007).

Implications of the Study

The outcomes obtained from this study should initiate a discussion about the need for additional studies regarding strategies in determining CS teachers' continuous needs. The outcomes may help establish data-grounded methods for developing quality professional development for CS teachers in secondary school settings. Such programs would better influence teachers' attitudes, impact their future practices, classroom culture and student learning. The diversity of the population (years of experience, and certification/endorsements etc.) involved in this study may allow future professional development programs to identify important factors for different teachers and increase their programs' effectiveness and efficiency. Any possible resource needs identified in this research (e.g. need for materials in specific topics, communication channels with stakeholders) may guide administrators in the field to address those concerns more efficiently. It is believed that the outcomes of this research would inform teacher education institutions to train future CS teachers with the necessary knowledge and skills and prepare them to face the unique challenges they may experience in their classes.

Terminology Unique to This Dissertation

Secondary Teacher. In this study, a secondary teacher refers to teachers who are teaching students in grades 6–12 in a formal school (public, private, or charter) or an after-school program (e.g., science centers, summer camps, technology clubs).

Computer Science (CS) Education. CS education refers to teaching the principles and practices of computers in K–12 education. This includes teaching CS content that were identified in the email listserv from the teachers' communications and aligned with the current literature as CS. Even though there was a discussion in the literature about the title of the field as computing education versus computer-science education (Guzdial, 2015), the second was selected because of its broad acceptance in the education and academic society.

Knowledge and Skills. One of the purposes of this study is to understand secondary CS teachers' knowledge- and skill-related needs. This study approached them as two necessary competencies to teach CS in secondary schools. Knowledge applies to the conceptual knowledge of CS and CS education (content knowledge, curricular knowledge, knowledge about the school environment, and professional development knowledge), and skills refers to the application of that knowledge in teaching CS to secondary-level students (e.g. pedagogical knowledge, pedagogical-content knowledge).

School Setting. In this study, school setting refers to variables that may influence teaching CS in secondary-level teaching environments. Based on the review of the teaching literature, those variables include school demographics, school planning, resources, and stakeholders (colleagues, administrators, and parents).

Years of Experience. Even though the teachers in this study may have extensive teaching experience in a different field, in this study we focused on years of experience in teaching CS courses or content.

Background. In this study, background refers to any education or experience participants have completed within the field of CS. Because teacher preparation for teaching CS varies

among states, the teachers in this study came from various backgrounds (e.g., business, engineering, education) and experience (e.g., CS work experience, teaching other content).

CSTN Framework. This term was created for this study, and is defined for this purpose as the computer science teachers' needs framework. Its design was guided by ISTE-CSE standards and the teaching literature between 2005 and 2015 and developed from the data in the CSTA listserv email communications.

CHAPTER II: LITERATURE REVIEW

The first part of this chapter includes definitions and discussions about what is a need and why need identification is important. Furthermore, this chapter explains why perceptions through email communications, a questionnaire, and interviews served as data sources to understand CS teachers' needs. This research started with the assumption that in-service CS teachers may have needs that limit their teaching effectiveness due to school environment, lack of CS teacher preparation and changing nature of the CS field. In this study, teachers' needs are listed under two broad categories as teacher-related and school-related. Teacher-related needs include needs related to teachers' knowledge and skills. The ISTE-CSE (ISTE, 2011) standards were used as the preliminary framework for understanding those types of needs. School-related needs include factors that are part of the school such as resources, colleagues, administration etc. Review of peer-reviewed empirical research studies in K–12 teaching literature between 2005 and 2015 was used as a preliminary framework for understanding school-related needs. In this chapter, teacher-related and school-related teacher needs are also described in detail with key characteristics.

A Need

A need is the difference between a target state and an actual state in our personal and professional initiatives (Guba & Lincoln, 1982; Leagans, 1964). For a teacher, a need is a barrier that limits the quality of his/her teaching and students' learning in a classroom. However, satisfaction of a need should not be a luxury (Guba & Lincoln) for a teaching or learning context. For instance, computers with reliable Internet access would be considered a valid need for all classrooms today (Inan & Lowther, 2010). Moore (1977) defined a science teacher's need in the "Development and Validation of a Science Teacher Assessment Profile" study. Keeping these

definitions and conditions in mind for the purposes of the present study, Moore's definition was revised and used in this study by adding the word "computer" in brackets as follows:

> A [computer] science teacher need is defined as a conscious drive, interest, or desire on the part of the [computer] science teacher, which is necessary for the improvement of [computer] science teaching. The conscious drive, interest, or desire results, in part, from the [computer] science teacher's interaction with the students and is perceived by the [computer] science teacher as the assistance which is needed in order to do a better job of teaching [computer] science (p. 145).

Examining Teachers' Needs

Need identification and addressing needs are important parts of an organizational structure, and needs must always be considered for successful change in every organization (Leagans, 1964). Schools as educational organizations, recognizing and addressing teachers' needs offer various advantages. Teachers expect beneficial learning outcomes from professional development programs (Pehmer et al., 2015). Understanding teachers' needs and considering them in professional development planning can increase program effectiveness by allowing participants to make their own assessments and create their own learning goals (Lee, 2005). Moore and Hanley (1982) surveyed 600 elementary teachers to understand their perceived needs and developed a professional-development event based on those perceptions. The results suggested that professional development programs based on elementary teachers' needs were more effective in improving teaching and learning. Similar studies have been conducted with other teachers. In a Turkish study with 435 science teachers from 75 high schools, Ogan-Bekiroglu (2007) reported that professional development programs that are based on teachers' needs and convenience provided more successful outcomes.

Email Listservs as a Source of Data

An initial step of this study looked at CS teachers' communications in an email listserv to identify their needs. Online communities are powerful, and provide low risk/effort environments

for teachers' professional development. The term *online community* is defined as a group of people meeting with similar professional interests and needs through an electronic communication channel (A. Jones & Preece, 2006). While teachers ask and answer questions in these communities, they can also freely share their opinions and ideas about their experiences (Hur & Brush, 2009). Online communities are informal spaces where teachers do not have any time- and institutional constraints like they have in formal trainings. This allows teachers to express their immediate needs (W. M. Jones & Dexter, 2014). For instance, Hew and Hara (2007) conducted an analysis of 1,242 literacy teachers' communications in an email listserv. They identified seven factors that motivated the teachers to participate to online communities, including collectivism (sharing knowledge), reciprocity (helping others), personal gains, altruism (feeling empathy for other teachers' needs), easy access to the information online, a respectful environment, and interest in learning. In the same study, the researchers found that teachers share problems, needs, practical knowledge, and personal suggestions frequently in their communications.

Teachers' Perceptions

Following the email analysis, this study delved further into secondary CS teachers' perceptions using a questionnaire and interviews. Perceptions drive our motivations and decisions (Lanas & Kelchtermans, 2015). Addressing teachers' perceived needs can facilitate their classroom practices (Johnson, 2006) and provide suggestions for their professional improvement (Janke, Nitsche, & Dickhäuser, 2015; Vähäsantanen, 2015). However, teachers have argued that their opinions were not considered when a change was proposed in their schools (Vähäsantanen). Therefore, goals with no teacher input have negatively influenced teachers' instructional practices (Holzberger, Philipp, & Kunter, 2014). For instance, in a study with 236

Australian public school teachers, with less than three years' experience, lack of voice in school decisions was reported as a significant reason for teachers planning to leave the profession (Burke, Aubusson, Schuck, Buchanan, & Prescott, 2015).

CS Teachers' Needs

In a dissertation study examining teachers' needs for professional development, S. A. Reid (2007) categorized teachers needs as teacher-related factors and school-related factors. In this present study, the same broad categorization was used to identify CS teachers' needs. Reid identified teacher factors as teaching qualifications (ability, content knowledge, pedagogic knowledge, teaching credentials).

Teacher Related Needs

In order to identify *teacher-related needs* specifically for CS education, this research used ISTE-CSE (2011) as a preliminary framework. ISTE is well-known as a professional organization for its guidance in improving teaching and learning by promoting the use of technology within various subject areas in education. By including scholars, teachers, administrators, government officials, businesses, and private foundations, ISTE has been committed to developing frameworks and standards for professionals at various levels of education, including teachers, students, and administrators. The National Education Technology Standards (NETS) is one of ISTE's highly recognized efforts and represents their promise to advance education. As part of the NETS, ISTE (2011) developed the standards for computer science educators and announced them as knowledge and skills guidance for teachers in computer science. The following subheadings explain the ISTE-CSE standards in detail as the preliminary framework for teacher-related needs.

Knowledge of content. Shulman (1986) defined content knowledge as "the amount and organization of knowledge per se in the mind of the teacher" (p. 9). Shulman argued that knowing the rules, concepts, and facts of a subject matter is not enough for a teacher to claim content knowledge in any subject area. Content knowledge also involves being able to justify central concepts in a discipline, how to apply those concepts, and how they are related to each other and other concepts in different subject areas. Content knowledge is one of the fundamental requirements of teaching CS effectively. Students are self-learners from various resources, and teachers are expected to constantly update their own CS content knowledge in order to keep up with their students (Ragonis et al., 2010). Standards in CS education guide teachers to continue learning new content and meet students' learning needs. ISTE-CSE (ISTE, 2011) broadly defined knowledge of content as understanding, demonstrating, and modeling CS concepts, rules, and principles. Below is the list of the specific ISTE-CSE content standards used in this research. Later each standard is defined in detail with examples.

1. Demonstrate knowledge of and proficiency in data representation and abstraction.
2. Effectively design, develop, and test algorithms.
3. Demonstrate knowledge of digital devices, systems, and networks.
4. Demonstrate an understanding of the role computer science plays and its impact in the modern world.

Demonstrate knowledge of and proficiency in data representation and abstraction. Data representation includes knowledge of data types such as primitive types (bits, bytes, text, Boolean, etc) and static/dynamic data structures (collection of data items) in computer science (ISTE, 2011). With abstraction, a computer scientist simplifies the complexity and focuses on the central concepts of a computing problem and a solution (Grover & Pea, 2013). Barr and

Stephenson (2011, p. 117) clarified data representation and abstraction by providing examples of similar processes from different subject areas (see Table 2.1).

Effectively design, develop, and test algorithms. This standard involves designing, analyzing and testing an algorithm for "complexity, efficiency, aesthetics, and correctness" (ISTE, 2011, p. 1). In general terms, an algorithm could be defined as the steps to complete a task. Computer scientists are expected to create those instructions and then develop a product using a programming language.

Table 2.1

Bringing computational thinking to K-12: What is Involved and what is the role of the computer science education community? (Barr & Stephenson, 2011, p. 117)

Concept	CS	Math	Science	Social Studies	Language Arts
Data representation and analysis	Use data structures such as array, linked list, stack, graph, etc.	Use a histogram, pie chart, bar chart, etc. to represent data	Summarize data from an experiment	Summarize and represent the trends	Represent patterns of different sentence types
Abstraction	Use procedures to encapsulate a set of often-repeated commands that perform a function	Use variables in algebra; identifying essential facts in a word problem	Build a model of a physical entity	Summarize acts; deduce conclusions from facts	Use simile and metaphor

Demonstrate knowledge of digital devices, systems, and networks. Demonstrating knowledge of digital devices, systems, and networks involves understanding how computers work in the "machine level," "understanding of operating systems," how an operating system communicates with the computer hardware, and how computers and mobile devices communicate on digital networks (ISTE, 2011, p. 2).

Demonstrate an understanding of the role computer science plays and its impact in the modern world. CS has a special role in all disciplinary fields and helps answer important questions. Teachers are expected to connect and demonstrate CS central concepts with other fields (Ragonis et al., 2010). In addition, this standard involves ethical use of computer systems. CS teachers should possess the knowledge and responsibility to model ethical computer use in their teaching and remind students to apply these rules in their CS practices (Ragonis et al., 2010). The Association for Computing Machinery (ACM, 1992) defined the general guidelines for ethical use of computers for professionals in "ACM Code of Ethics and Professional Conduct" document, as follows:

1. Contribute to society and human well-being.
2. Avoid harm to others.
3. Be honest and trustworthy.
4. Be fair and take action not to discriminate.
5. Honor property rights including copyrights and patent.
6. Give proper credit for intellectual property.
7. Respect the privacy of others.
8. Honor confidentiality.

Effective teaching and learning strategies. ISTE (2011) defined effective teaching and learning strategies as demonstrating knowledge and application of CS concepts to help students understand and apply them easier. This standard included, as Shulman (1986) defined:

1. pedagogical content knowledge,
2. curricular knowledge, and
3. knowledge of learners.

Pedagogical content knowledge. Pedagogical content knowledge means effective presentation of content by using various instructional strategies and materials for learning purposes (Shulman, 1986). Previous research in engineering education suggested alternative ways to "chalk and talk pedagogy" (Mills & Treagust, 2003, p. 13). Chalk and talk was referred to lecture-based delivery of content in large classrooms. Even though it was tested and validated more in mathematics and science education, constructivism (Duffy & Jonassen, 1992) guided the development of new instructional strategies for CS education. Five of the successful strategies used in CS education, and discussed in the following sections, include problem-based learning (Kay et al., 2000; Yadav, Subedi, Lundeberg, & Bunting, 2011), project-based learning (Mills & Treagust, 2003), peer instruction (Porter, Lee, & Simon, 2013), pair programming (McDowell, Werner, Bullock, & Fernald, 2006), and media computation (Guzdial, 2003).

Problem-based learning (PBL). In problem-based learning (PBL), students work in groups to solve a complex problem using various types of scaffolds (Hmelo-Silver, 2004). Solving problems is not the only goal in problem-based learning. Hmelo-silver defined it as a self-directed learning practice in which students apply their former knowledge to solve a problem and increase their knowledge and understanding of a topic. She has argued that PBL experience creates an environment where students attain higher order problem solving skills and

improve their creative thinking while also gaining factual knowledge. However, classes that focus on PBL need specially designed curriculum, teacher development, and assessment (D. F. Wood, 2003). Wood summarized these important components:

1. Teachers should be trained to become facilitators in this process rather than providing direct instruction and delivery of the information.
2. Assessment should not be designed to evaluate students' factual knowledge gains. It should include formative and summative assessments that allow learners to reflect on their own learning experience.

Previous research in CS education provided successful PBL practices in higher education. In CS education PBL, students are given a complex CS problem in a computer lab environment with tutorials to facilitate their problem-solving process (Kay et al., 2000). In two different sessions of an introduction to programming course for higher education students in a CS department, the group trained with PBL provided higher learning outcomes compared to the direct-instruction session (Kay et al., 2000). In a different study with two classes of undergraduate electrical engineering students, one taught with the principles of PBL and another with traditional lecturing, learning acquisition was two times better in the PBL group, as shown in a post-test comparison (Yadav et al., 2011). Even though PBL has provided evidences of learning gains in higher education, there was limited evidence of the teaching practice in K–12 CS classes.

Project-based learning. The terms problem-based learning (which we shorten to PBL) and project-based learning (discussed next) are often confused, but there are distinct differences between the two. While PBL is problem-originated and was derived from the practices of medicine, project-based learning is project-originated and was derived from practices of industry

and engineering (Mills & Treagust, 2003). Tasks in project-based learning are designed similar to projects in real life and give learners opportunity to apply their knowledge in product design and development (Mills & Treagust). Even though there are similarities with PBL, project-based learning is product focused and requires students to be careful about resources and time, while PBL gives more flexibility in this process. Similar to PBL, project-based learning also requires carefully designed projects and assessments (Mills & Treagust).

Project-based learning has also been implemented in CS education. In an introductory programming course required two semesters for all the freshman in an engineering school, a traditional direct-instruction approach was applied in the first semester and then replaced with a project-based learning approach in the second. The students were given a semester-long programming project. They chose their own groups and decided on a project idea from a list of possible topics. At the end, the instructors and the students reflected positive perceptions about the change and achieved better learning outcomes in the second semester (Davenport, 2000). In a technical high school web-design and programming course using an online education system, students were tasked with a website design project using web-design and programming tools. At the end of the semester, the students reflected positive perceptions regarding the online education system and their own learning, which allowed them to work successfully in collaboration with others students while considering time limitations and finishing the project goal (Köse, 2010).

Peer instruction. Peer instruction was also reported as an effective instructional strategy in CS education. It was originally derived from physics education. Peer instruction in physics, students are given chance to express their difficulties in discussions (Crouch & Mazur, 2001). First, the students answer a question individually, then discuss the same question in groups, and finally vote for an agreement based on the group discussion (Crouch, Watkins, Fagen, & Mazur,

2007). In CS education, peer instruction was used to understand students' challenges and reduce fail or withdrawn-student rates (Chase & Okie, 2000). In an introductory Java programming course, Simon, Kohanfars, Lee, Tamayo, and Cutts (2010) asked challenging programming questions to students in two sessions of the class and tested how discussions in peer instruction influenced students' attitudes regarding the content and answers to the questions. The researchers found that the students' correct responses increased after collaborating with their peers and they reflected positive attitudes regarding the content and the activity.

Pair programming. Pair programming is a technique in which two programmers work on the same programming task design and development using one computer simultaneously. This technique was derived from CS industry practices and has been used as an instructional method in both K–12 and higher education. Previous research has provided evidence of its effectiveness for learning purposes. For example, in a research study with 600 students in an introductory programming course, the students were allowed to choose whether they would work independently or with a partner. Students who worked in pairs created better products and finished the course in better rates (McDowell, Werner, Bullock, & Fernald, 2002). The researchers emphasized the collaborative nature of the technique as an important component of the programming learning activity (McDowell et al., 2002). Williams and Kessler (2000) highlighted the following principles of pair programming that creates ownership of the work from both members of the pair:

1. Both programmers should equally participate and assigned to work on the same task with same resources and limitations.
2. The programmers should take turns and review each other's work when the other one is coding. They are both responsible for the mistakes and successes.

3. Both should actively participate and spend reviewing time for improvement of the work.
4. Both programmers need to accept that working with a teammate is difficult and should reflect a positive attitude toward mistakes or areas of improvements.

Media computation. Compared to aforementioned instructional methods, media computation was a recently developed instructional technique that emphasized learning computing concepts and skills in digital media design (Guzdial, 2003). Guzdial has argued that digital media (e.g. images, videos, audios) can be modified and redesigned using programming and this process can help students learn computation in a more meaningful way.

Other tools for CS teaching. In addition to the instructional methods discussed here, previous studies documented the success of using various other tools in CS education. For example, in a study with 88 high-school students learning computer memory concepts, Papastergiou (2009) explored learning effectiveness of a computer game and found it effective in supporting students' learning. Papastergiou highlighted problem- and discovery-based components of computer games for students' CS learning. Esteves, Fonseca, Morgado, and Martins (2011) used a virtual environment in a CS class at a university level and highlighted the pedagogical advantages in terms of allowing students to communicate and collaborate simultaneously in a virtual "community of practice" (p. 11). Robotics kits have been also popular in CS, and studies have emphasized a "learning by doing" approach when robotics are used in CS education. However, previous research has provided both positive and negative learning outcomes when robotics kits are used. Bers, Ponte, Juelich, Viera, and Schenker (2002) used Lego Mindstorm kits in a kindergarten classroom and tested outcomes of a robotics activity. The researchers found that the students were able to create and control computational objects while exploring and reflecting during the activity. On the other hand, an experimental study with 800

higher education students tested robotics versus non-robotics content in an introductory CS classroom and reached negative learning outcomes in the experimental robotics session (Fagin & Merkle, 2003).

Curricular knowledge. Curricular knowledge is also part of the *effective teaching and learning strategies* standard, and it means knowledge of a variety of curriculum materials available to teaching a subject (Shulman, 1986). Curriculum is the place where a teacher obtains the ideas, resources and tools to present, exemplify and evaluate the subject matter content. In terms of curricular knowledge, teachers are expected to know the available curriculum options and materials for their CS classrooms. Regarding curriculum, one major issue that current literature explained is the misconception about what CS education is. Barr and Stephenson (2011) have argued that curriculum about computer literacy is not CS education. They listed the following topics as computer science:

- programming,
- hardware design,
- network,
- graphics,
- databases and information retrieval,
- software design,
- programming languages,
- logic,
- programming paradigms,
- translation between levels of abstraction,
- artificial intelligence,

- the limits of computations (what computers cannot do), —applications in information technology and information systems, and —social issues (Internet security, privacy, intellectual property, etc.) (p. 113).

Various organizations provided different levels CS education curriculums about these topics. As detailed before, ISTE provided standards, computational thinking materials, multimedia resources, handouts, and booklets in their website (Sykora, 2014). Project Lead the Way (PLTW), a nonprofit organization, also provided curriculum and content for various courses for 9–12 grades CS education (PLTW, 2016). Computer Science Teachers Association (CSTA) is another active organization that provided standards and resources for K–12 CS teachers. CSTA standards started from kindergarten and provided supportive materials to align content with those standards (Carey et al., 2011). CSTA announced their CS education curriculum standards in 2011 and updated their standards with new content in 2016 with the active contribution of various stakeholders (CSTA, 2016). Another non-profit organization, Code.org, provided curriculum and resources for elementary, middle, and high school teachers through local workshops they conduct in most of the states (Code.org, 2013). College Board (2016), a non-profit organization, had over 30 advanced placement courses to prepare students and earn college credit or advance placement during high school years. After taking an advanced placement course, high school students could take an exam to transfer the course credit to college. College Board designed a new advanced placement (AP) CS course, focusing on computational thinking, called AP Computer Science Principles, applied in Fall 2016. They provided materials and guidance to help CS teachers prepare their students for college education in CS.

Assessment is another important part of curricular knowledge. In addition to providing

content, teachers need to identify their students' areas of improvements and make sure that their students are learning the content. CS education aims to provide students with various concepts and skills in computing, which needs higher order thinking assessment. According to Lewis and Smith (1993), higher order thinking involves using the previously learned knowledge to answer a different problem or achieve a different purpose.

Assigning grades should not be the goal of a CS teacher's assessment (Guzdial, 2015). Assessment in CS education should be formative. Formative assessment could provide teachers and students feedback to test their competency during the process of learning (Sadler, 1989). Even though not practical, Ben-Ari (2001) stressed teachers' observation and questioning of students' engagement with their learning experience as the best way for assessment. Furthermore, CS teachers could develop rubrics to provide students with an ongoing feedback source, and allow students to reflect on their own learning and see their weaknesses in content (Moskal, Miller, & King, 2002).

Knowledge of Learners. Effective teaching and learning strategies include knowledge of learners, such as students' 1) beliefs, 2) preconceptions (knowledge they brought from previous life experiences), and 3) special needs.

The first factor, student belief refers to a student's attitude and self-efficacy about a subject. A student's belief about a subject is an important indicator of achievement. In a comparative study in the United States and Japan, students' beliefs about mathematics were found to be related to mathematics achievement test scores (House, 2006). In the same study, the students who defined mathematics as boring and difficult received lower math test scores. In order to deal with these types of students, teachers' knowledge of learners and support was found to be important factors for changing student attitudes and self-efficacy in a subject area (Rice,

Barth, Guadagno, Smith, & McCallum, 2013).

The second factor, students' preconceptions, refers to their prior academic skills, which could influence students' learning of new concepts and their academic scores (Paulson & Marchant, 2009). For instance, previous reports and research have highlighted the importance of students' mathematics background in learning CS. An influential report created by an ACM curriculum committee described recommendations for designing undergraduate CS programs and highlighted mathematics as a requirement for learning CS (Austing, Barnes, Bonnette, Engel, & Stokes, 1979). In a study with 46 students aiming to teach CS concepts, Meerbaum-Salant, Armoni, and Ben-Ari (2013) used the Scratch programming platform. Scratch is a free programming environment where users can create interactive products (animations, games etc.) using block-based programming. The researchers have argued that lack of mathematics concepts influenced the students' understandings and expressions of CS concepts. In another study with public middle-school students, Groverm, Pea, and Cooper (2016) conducted a 7-weeks curriculum to teach algorithmic thinking and programming, each week focusing on a different concept in computing. At the end of the semester, the researchers found that academic preparation (based on state mathematics and English test scores) was an influential factor on students' CS learning outcomes.

The third factor, special needs, includes students who speak English as a Second Language (ESL) as well as those with learning and physical disabilities. ESL students have different learning-style preferences (J. M. Reid, 1987). Even though learning styles approach receive criticism from some researchers (Pashler, McDaniel, Rohrer, & Bjork, 2008), majority of the 1,388 ESL students in a comparative large-scale international study preferred different learning styles (J. M. Reid, 1987). Special needs also include hearing and visual impairments,

mental obstacles, speech and language impairments, emotional disorders, and specific learning disabilities (Hasselbring & Glaser, 2000). In a study with 600 K–12 teachers from various subject areas in Iowa, Minnesota, and Wisconsin, teachers mentioned difficulty of dealing with special needs students and aimed for programs that can prepare them to teach students with special needs (Snider & Roehl, 2007).

Effective learning environments. ISTE (2011) defined effective learning environment as creating and sustaining an accessible, inclusive and effective learning environment for all the students' learning. This refers to the effective use of computer systems and components, and offering equitable and accessible computer resources for all the students. This standard also emphasizes underrepresentation of gender and ethnic groups in CS classes. Margolis (2008), in the book *Stuck in the Shallow End: Education, Race, and Computing*, has emphasized how technology-rich schools show a lack of representation of girls in CS classes. This lack of representation has impacted the quality of the workforce by reducing diversity and the capacity to solve problems in science and technology (Abrams, 1989; Jackson, Starobin, & Laanan, 2013). This is a "social justice [issue]" in U.S. education and employment systems (Ong, Wright, Espinosa, & Orfield, 2011, p. 175). To reduce the lack of diversity problem, K–12 teachers could play an important role in encouraging underrepresented groups to take their places in CS courses (Ong et al., 2011). It is expected that teachers will recognize this as a problem and possess the knowledge and skills to become role models for underrepresented populations in their schools, providing these populations with equal access to the CS field. Teachers could be "institutional agents and gatekeepers" to encourage underrepresented groups and minorities to be successful in the school environment (Allen, 2015, p. 78).

Effective professional knowledge and skills. CS is a fast-evolving field that requires teachers to engage in ongoing professional development. (ISTE, 2011). According to ISTE-CSE standards, this involves knowledge of available organizations and groups for accessing resources and knowledge, following and applying new research to teaching practice, and keeping up with the content and professional development standards for secondary CS education. Table 2.2 demonstrates the overview of the teacher related needs preliminary framework.

Table 2.2

Overview of the Teacher Related Needs (ISTE, 2011, pp. 1-2)

Need categories				
Knowledge of content	Demonstrate knowledge of and proficiency in data representation and abstraction	Effectively design, develop, and test algorithms	Demonstrate knowledge of digital devices, systems, and networks	Demonstrate an understanding of the role computer science plays and its impact in the modern world
Effective teaching and learning strategies	Knowledge of instructional strategies and materials	Knowledge of learners	Curricular knowledge	Assessment knowledge
Effective learning environments	Effective use of computer systems and their peripherals	Equitable and accessible resources for all the students		
Effective professional knowledge and skills	Knowledge of organizations and resources for professional development	Knowledge and application of new research to teaching CS	Knowledge on secondary CS education standards	

School Setting-Related Needs

Schools have a responsibility to create conditions for effective teaching and student learning by providing resources and support for teachers (De Neve, Devos, & Tuytens, 2015). A satisfying teacher work environment can have a positive impact on teachers' instructional goals and students' learning (Janke et al., 2015). On the other hand, constraints in schools (e.g., equipment, time, support) can limit teachers' practices and students' learning (Webel & Platt, 2015). Some school-related constraints identified from the literature review were school demographics (Paulson & Marchant, 2009), resources (Hew & Brush, 2007), school planning (Beligiannis, Moschopoulos, Kaperonis, & Likothanassis, 2008), parents (Eamon, 2005), colleagues (Burke et al., 2015), and administration (Printy, 2010). CS education literature in K–12 is limited in this regard compared to other subject areas because of a lack of previous interest and knowledge in K–12 CS education research. Therefore, a preliminary framework of school-related teacher needs was created based on teaching literature between 2005 and 2015.

Table 2.3

Overview of School Related Needs

Categories	Themes	Citations
School Demographics	Large class size, issues related to diversity (Race, gender) and socio-economic status	Allen, 2015; Belfi, Gielen, De Fraine, Verschueren, & Meredith, 2015; Eamon, 2005; Melnick & Meister, 2008; S. A. Reid, 2007; Snider & Roehl, 2007
Resources	Lack of availability and access to resources (teaching materials, tools, funding etc.), lack of time/high workload, lack of PD	Betoret, 2009; Burke et al., 2015; Ertmer, Ottenbreit-Leftwich, Sadık, Sendurur, & Sendurur, 2012; Hew & Brush, 2007; Ogan-Bekiroglu, 2007; Snider & Roehl, 2007; Wachira & Keengwe, 2011

School Planning	School policy related problems, timetable related problems	Beligiannis et al., 2008; Hew & Brush, 2007; Zhang, Liu, M'Hallah, & Leung, 2010
Parents	Lack of cultural capital (parent knowledge and skills, education, access to technology and educational resources)	Allen, 2015; Eamon, 2005; Hughes & Kwok, 2007; Jovés, Siqués, & Esteban-Guitart, 2015; Paulson & Marchant, 2009; Shumow, Lyutykh, & Schmidt, 2011
Colleagues	Lack of support, collaboration and community	Burke et al., 2015; De Neve et al., 2015; Ertmer et al., 2012; Fox & Wilson, 2015; Salkovsky, Romi, & Lewis, 2015
Administration	Lack of support, negative attitude, and lack of knowledge about a field	Burke et al., 2015; Hew & Brush, 2007; Knoeppel & Rinehart, 2009; Printy, 2010; Salkovskyy et al., 2015

Table 2.3 presents an overview of possible constraints, problems, and limitations as identified by the literature review. As Guba and Lincoln (1982) discovered, while elimination of those problems can significantly benefit a teacher, having those problems constrains the teaching practice and leads to a need. Following is a short description of each category, identified and considered as a possible need.

School demographics. Demographic characteristics, size, diversity, and socio-economic status, of a school and a classroom are important variables in education. Demographics may influence teachers' expectations from the students in terms learning goals (Brault, Janosz, & Archambault, 2014). Teachers are expected to possess the correct attitude and skills to address issues related to students' gender, cultures, and languages (Barnes, McInerney, & Marsh, 2005; Blumenreich & Gupta, 2015).

Resources. Problems related to resources may include one or more of the following:

1. lack of availability/access to resources,

2. lack of time or high workload,
3. lack of professional development opportunities.

In regard to lack of resources and access, teachers mentioned instructional equipment and materials, and computer technologies as their needs in classroom (Hew & Brush, 2007; Ogan-Bekiroglu, 2007; Wachira & Keengwe, 2011). The second limitation, teacher lack of time and high workload has been reported as an important problem for decades. In recent studies, teachers complain in various studies about lack of time due to pressure of meeting standards and preparing students for high-stakes tests that prevented them from applying new content, instructional strategies, and tools to their classroom (Salkovsky et al., 2015; Wachira & Keengwe, 2011). Furthermore, high workload has been reported as an important reason for teacher burnout and early retirement (Košir, Tement, Licardo, & Habe, 2015; Salkovsky et al., 2015). The third limitation is lack of professional development (PD) opportunities in schools. In a study with elementary teachers, 40% reported that they were not sufficiently trained in teacher preparation programs and wished for more professional development in their schools (Snider & Roehl, 2007). In another study, physical education teachers reported a lack of PD activities as the most important constraint for teacher change (Bechtel & O'Sullivan, 2007).

School planning. In the present study, school planning refers to timetabling. Zhang et al. (2010) defines timetabling as the most efficient and effective sharing of school resources (spaces, times, equipment etc.) for the teachers and students' use for teaching and learning purposes. While unfair allocation can result in uncomfortable school environment for teachers, fair and effective allocation may reduce teachers' stress and increase their confidence to the school environment. (Salkovsky et al., 2015).

Parents. Parents' involvement in the students' learning process is critical for success at every level and subject in K–12 education. Parents' relationship with students in early grade levels is a precondition for the students' academic interest and achievement (Hughes & Kwok, 2007), and parents' positive relationship with teachers can encourage teachers to stimulate students' learning (Shumow et al., 2011). Parent involvement at higher levels can influence students' academic success and encourage them to develop positive attitude about a subject (Shumow et al., 2011). Parental support at home encourages communication with their children about school that may contribute to higher reading and math scores (Eamon, 2005). Benefits of parent involvement are clear from various points of views and lack of parent involvement may produce problems with student learning, especially in secondary education (Shumow et al., 2011).

Colleagues. Teachers' relationship with other teachers is a critical component of a successful school environment. Teachers expect to see role models and get support from other teachers. For instance, in K–12 technology integration studies, lack of colleague support has been reported as one of the most important barriers to using technology in education (Ertmer et al., 2012). In a study with 336 early-career teachers in Australia, over 70 % of the teachers reported that they had limited or no collaboration with their colleagues in their schools (Burke et al., 2015). Beginning teachers especially would also like to have environments in schools where they can discuss and learn from their colleagues (De Neve et al., 2015). Košir et al. (2015) reported that lack of collaboration creates job stress in the school environment. In Burke et al. study, lack of collaboration with other teachers and mentors were reported as one of the reasons for leaving the profession, and suggested that teachers need school environments that create spaces for teachers to share knowledge and resources.

Administrators. Administrators are the people who are responsible for creating a comfortable school environment for teachers and students. Their vision can provide teachers the space to apply and advance their practice (Printy, 2010). In a study with 349 elementary school teachers, principal support was found as an essential element of school context to build and support delivery of instruction (Knoeppel & Rinehart, 2009). Lack of administrative support could produce pressure on teachers and lead to teachers leaving the school or profession (Salkovsky et al., 2015).

Factors That Influence Teachers' Needs

Background in CS

In core subject areas, highly qualified teachers in the US are required to earn at least a bachelor's degree, a teaching license or certification, and have student teaching experience, before taking on the responsibilities of their own classroom (Department of Education, 2004). Studies in teacher education have suggested that quality of teacher preparation and licensing requirements are important indicators of students' success (Darling-Hammond, 2000).

Teachers' content and pedagogy knowledge are the first conditions for students' achievement in any subject area. However, CS teachers in the US come from various backgrounds and knowledge, and some without any teaching license and experience (Ericson et al., 2008; Guzdial & Fisher, 2014). Due to an increasing demand in CS education and a need for more CS teachers, higher education institutions have offered short-term professional fdevelopment programs for other content area teachers to teach CS (Menekse, 2015). However, these short-term programs have been limited in content preparation and may leave CS teachers with confusions or a lack of the necessary knowledge and skills to be able to design and teach their own CS courses (Franke et al., 2013, Liu, Hasson, Barnett, & Zhang, 2011).

Furthermore, some schools have allowed teachers without any teacher education to teach CS courses (Barr et al., 2013; desJardins, 2015). Even though CS content by itself has not been enough to teach CS, it has been the only criteria for hiring CS teachers in some schools, and they have often hired computer scientists, information technologists, and people with CS work experience (Barr et al., 2013). However, this content knowledge may come with limited or no knowledge about pedagogy. According to a CSTA (2015) high-school survey, about 35 % of 1,354 current secondary teachers in the US have not taken even one computer-science teaching methods class. Even though these content experts might have attended short-term PD programs, these programs might not prepare them effectively to become successful teachers to transfer that knowledge to their students (Menekse, 2015). Therefore, this study aims to look for differences in needs between people with different background in CS and CS education.

Years of Teaching Experience

Years of teaching experience in the classroom is another important factor that can make a difference in student achievement. Darling-Hammond (2000) has argued that teachers with less than four years' experience are usually less effective than teachers with more experience. This has been due to challenges that new teachers face in their early years of teaching, such as accessing curriculum resources, managing classrooms, planning and programming their teaching, and communicating with parents and administrators (Burke et al., 2015; Fantilli & McDougall, 2009). These new teachers have been trying to survive in the profession and learn how to teach in the early years of their experience (Huberman, 1989).

On the other hand, as they get more experience, teachers gain advantages such as access to collections of curriculum resources and knowledge of strategies for classroom management and instruction. For instance, Rockoff (2004) found a positive relationship between teachers'

years of experience and students' achievement in reading comprehension and vocabulary up to 10 years of teaching experience. After 10 years, this relationship changed to a negative correlation. On the other hand, experienced teachers may experience boredom after teaching the same content for 10 years (Dl Geroimimo, 1985). In a study with 1,430 in-service teachers testing their self-efficacy on teaching strategies, classroom management, and student engagement, the teachers' self-efficacy first increased after three years, continued up to midcareer, and then started to decline (Klassen & Chiu, 2010). In addition to beliefs and self-efficacy, if experienced teachers do not update their instructional strategies for today's classroom environments and their students' needs, their teaching effectiveness may suffer. Especially for CS, updating knowledge of content is very critical due to the ever-changing nature of the CS field (Gal-Ezer & Stephenson, 2009; Ragonis et al., 2010; Stephenson et al., 2005). Therefore, the present study also aims to look for differences in needs between CS teachers with different years of experience.

Summary of Literature Review

In this study, possible teacher knowledge- and skills-related areas of need were obtained from the ISTE-CSE standards. Such standards include knowledge of content, effective teaching and learning strategies, effective learning environments, and effective professional knowledge and skills. A review of the teaching literature between 2005 and 2015 was conducted to explore school-related needs. A teacher's needs in the school environment are infinite and impossible to frame entirely, and this literature review does not suggest covering all the needs a secondary CS teacher may have. However, school demographics, resources, school planning, parents, colleagues, and administration were identified for the preliminary framework.

This study looked at possible constraints, problems, and barriers to understanding secondary CS teachers' needs. Elimination of these problems is considered as a need (Guba & Lincoln, 1982) and offers benefits to pre-service and in-service CS teachers for teaching CS more effectively.

CHAPTER III: METHOD

This chapter covers the research design. The research design includes a description of why and how this dissertation utilized mixed-methods research (MMR) design. Following the description, an overview of the data collection steps used in this study is provided. There were four phases in this research. Each phase's data collection and analysis steps were explained in detail. A description of the participants was provided in each phase.

This research aims to answer the following research questions:

1. What needs do U.S. secondary school computer science teachers share related to their knowledge, skills, and school settings?
2. How do teachers' needs vary based on years of CS teaching experience and background (e.g., education, training) in CS?

Why Mixed-Methods Research

This mixed-methods study enabled the researcher to pursue both exploratory and explanatory purposes using qualitative and quantitative methods. At the time of this study, credentials and requirements of teaching CS varied and were lacking in some U.S. states. Therefore, in-service CS teacher population was not well defined (Barr et al., 2013) and it was not clear how to access a representative sample. Instead, the researcher used one-year email communications between teachers from the Computer Science Teachers Association (CSTA) email listserv as the first step to define the population. Furthermore, there was no previous framework developed to explore CS teachers' needs. Those same email communications were used again, to inform the development of a CS teachers' needs framework. Based on the guidance of this framework, about three-year email communications in the same listserv were analyzed and a questionnaire was developed to explore CS teachers' needs. Because of these

measures, this study is considered exploratory in nature by examining the representative sample, developing the framework, and identifying CS teachers' needs from the emails and the questionnaire. In terms of explanatory purposes, there were open-ended items in the questionnaire that asked CS teachers to provide examples and more details about their needs. In addition, follow-up interviews were conducted in the last phase to gain more descriptions and examples about the questionnaire findings to explain and enhance the study results (Fraenkel & Wallen, 2010).

Research Design

There are various forms of mixed-methods research designs in the literature. Greene, Caracelli, and Graham (1989, p. 259) provided five forms of mixed-methods research designs:

1. Triangulation [design] seeks convergence, corroboration, and correspondence of results from the different methods.
2. Complementarity [design] seeks elaboration, enhancement, illustration, and clarification of the results from one method with the results from the other method.
3. Development [seeks] to use the results from one method to help develop or inform the other method, where development is broadly construed to include sampling and implementation, as well as measurement decisions.
4. Initiation [design] seeks the discovery of paradox and contradiction, new perspectives of frameworks, the recasting of questions or results from one method with questions or results from the other method.
5. Expansion [design] seeks to extend the breadth and range of inquiry by using different methods for different inquiry components.

After reviewing existing models to find the most appropriate research design, a researcher may have to "generate [his/her] own" or "combine existing designs" (Teddlie & Tashakkori, 2009, p. 163). A similar approach has been successfully applied in Exter's (2011) dissertation study about software designers' experiences in education- and instructional technology-related fields. As Teddlie and Tashakkori further explain,

> In some cases, you may have to develop a new MM [mixed-method] design, using flexibility and creativity, because no one best design exists for your research project, either when it starts or as it evolves. Some MM studies change over the course of the research, resulting in designs with more strands than originally planned or with strands that change in relative importance. (p. 164)

Therefore, the current research study utilized the following the mixed-methods research design approaches described by Greene et al. (1989):

1. Phase 1: literature review and one-year email communications helped the researcher explore the teacher population and develop the computer science teachers' needs (CSTN) framework— INITIATION.
2. Phase 2: reviewing about three years' of email data helped to initiate a discussion regarding CS teachers' needs from their email communications—INITIATION.
3. Phase 2: the email conversations also helped the researcher develop a questionnaire—DEVELOPMENT.
4. Phase 3: conducting the questionnaire helped the researcher to compare, converge, corroborate, and expand the findings in the email communications from Phase 2—TRIANGULATION and EXPANSION.
5. Phase 4: qualitative interview data helped the researcher to explain, compare, and elaborate, as well as to enhance the data about, teachers' needs with examples and descriptions—TRIANGULATION and COMPLEMENTARITY.

This design is *interactive* and *sequential* (Creswell & Clark, 2007). The interaction occurred in the interval between the data-collection methods on designing the questionnaire instrument from the email analysis and developing the interview protocol from the questionnaire findings. Because each data-collection method emerged from the one before it, the design was perforce sequential. Data analysis was interactive in most cases as well. Initially, the email communications were analyzed individually, using both content and thematic analysis techniques (Creswell, 2003). However, questionnaire qualitative data and the phase 4 interview data were analyzed using a constant-comparison method (Creswell), and was then compared with the phase 2 data interactively. The quantitative data results were compared with the qualitative data results at the end of the study (Creswell & Clark, 2007). Figure 3.1 illustrates this process in the following page.

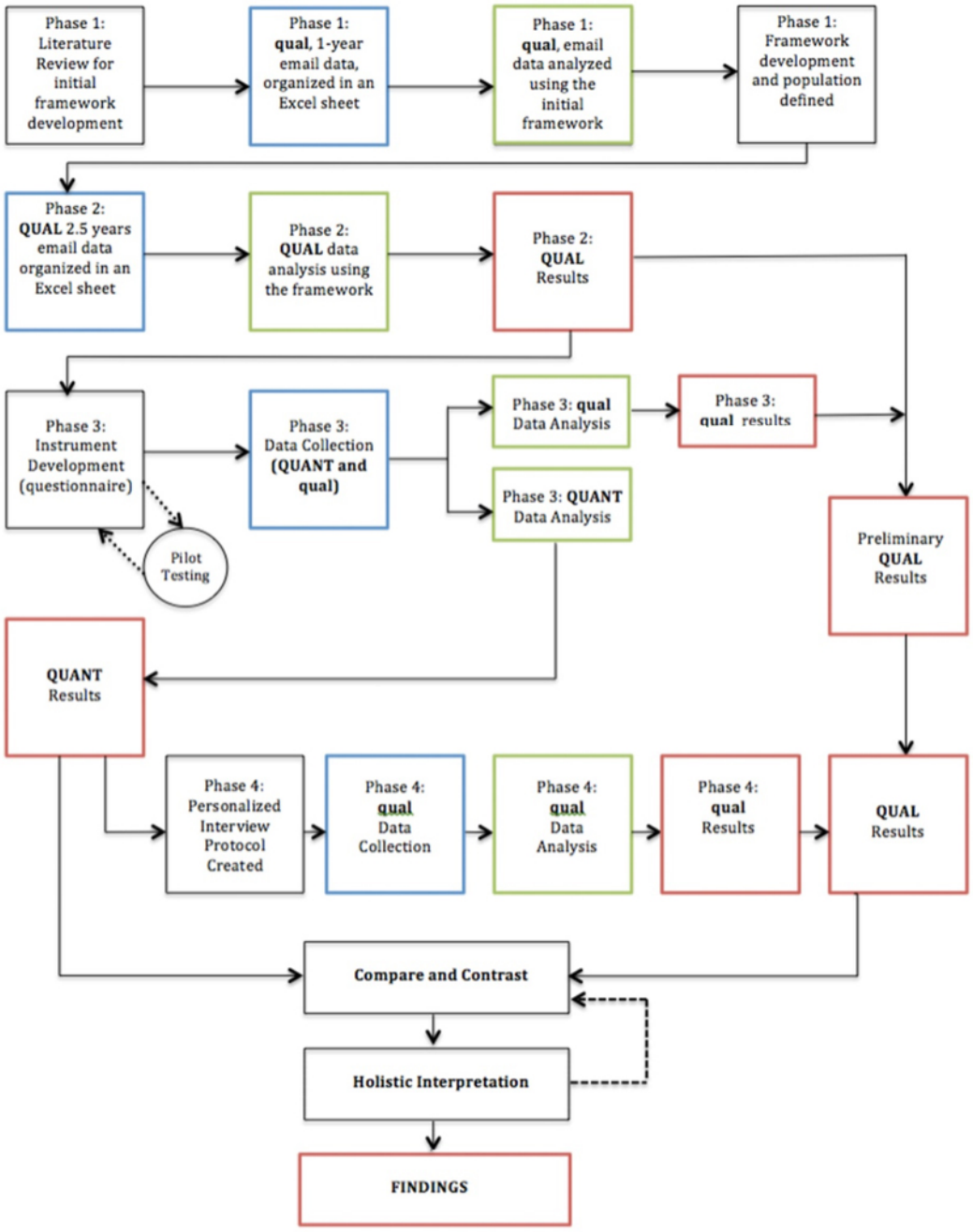

Figure 3.1. Overview of Research Procedures

Methods Overview

This study was divided into four phases:

1. Phase 1: The researcher developed the framework and defined the population using:
    a. the ISTE-CSE standards,
    b. previous teaching literature,
    c. over a year of the CSTA email listserv data.
2. Phase 2: The researcher analyzed secondary CS teachers' communications between January 1, 2013 and October 1, 2015 (about three years) in the CSTA email listserv and identified their needs using the framework categories. In this phase, the researcher also developed a questionnaire and pilot-tested it with four in-service teachers, using a think-aloud protocol (Cooke, 2010).
3. Phase 3: After reviewing and revising the questionnaire based on the think-aloud activity results, CSTA distributed the questionnaire on behalf of the researcher to secondary CS teachers through their teacher member database. The researcher conducted a descriptive analysis of the questionnaire's quantitative data and a constant-comparative analysis of the open-ended questionnaire responses with the phase 2 data analysis.
4. Phase 4: Based on their responses in the questionnaire, the researcher chose a purposive sample of teachers from the voluntary interview participants, conducted interviews, and asked participants to elaborate their needs with examples and evidences. The researcher conducted constant-comparative analysis of the interview data with the phase-2 and phase-3 analysis results.

In the remainder of this section, the data collection and analysis steps for each phase are explained in detail with participant descriptions.

Phase 1: Framework Development

The purpose of the first phase was to develop the CSTN framework and define the population. Teachers' needs were initially categorized into two groups: 1) teacher knowledge- and skills-related needs, and 2) school setting-related needs. Respectively, the ISTE-CSE and the teaching literature between 2005 and 2015 constituted the preliminary framework (see Appendix A) to guide the final framework development process grounded from the email data.

ISTE is a professional organization well known for its guidance in improving teaching and learning within various subject areas through technology. ISTE-CSE (ISTE, 2011) is one of the highly recognized efforts of the organization. ISTE-CSE was selected as part of this study's preliminary framework of teacher knowledge and skills because it provided comprehensive standards in terms of pedagogical, curricular, and professional knowledge and skills in teaching CS, and because it focused only on secondary level.

Categories related to school setting such as school demographics, resources, administrators, colleagues, students, parents, and school planning were generated from the teaching literature between 2005 and 2015. The criteria used for the literature review were that the articles had to be between 2005 and 2015, peer-reviewed, and empirical research papers. Keywords such as "teacher needs" and "teaching barriers" were used in Google Scholar and ERIC database searches. In addition, "Teaching and Teacher Education" journal was reviewed as a major literature source because of its focus on teaching and teachers.

Data Collection

In order to develop the CSTN framework, publicly available email communications (759 emails from 222 unique secondary CS teachers) in the CSTA email listserv between January 1, 2013 and March 1, 2014 were examined. The analysis for framework development ended at this stage due to saturation of data. The researcher decided that further examination did not contribute to the purpose of developing the framework and defining the population. The emails were collected from the publicly available CSTA email listserv (http://listserv.acm.org/SCRIPTS/WA-ACMLPX.CGI?A0=CSTA-MEMBERS). Members of the email listserv included teachers, academics, professionals, and students who were either working in or interested in K–12 CS education. The membership was free and anybody with a valid email address could become a member. The listserv was started on January 1, 2013 and was an active communication channel for members at the time of the present study's data collection. The discussions evolved around topics related to teaching CS in K–12, such as problems, needs, and sharing resources. Emails from all members were used in the data; however, the main focus of phase 1 data was communications among secondary CS teachers in the United States. For instance, when an academician was involved to the discussion and affected how the discussion evolved, their contributions became part of the data, and were used for the researcher to understand the context of the further discussion.

Data Analysis

Before starting the phase 1 analysis for framework development, an Excel spreadsheet was created and included the following column titles (see Figure 3.2):

1. the subject (email subject),
2. from (sender's name),

3. email (sender's email address),
4. date (the day email sent),
5. content (email content),
6. signature,
7. attachment.

Subject	From	Email	Date	Content
Advice...New Intro to Programming Course using C++	B.W.		3/25/14	Hello All, I posted a message a few weeks back (asking for suggestions on getting a brand new Intro to Programming course, using Java, off the ground) and got so many kind responses.... thanks for sharing your ideas. Unfortunately, my high school has informed me we won't be using Java as the programming language. So.... the same question but with a different language! Suggestions on getting a new course set up and other suggestions for an Introduction to Programming, featuring C++. High school level course. I will not be able to talk my administration into using a different language... they have attached themselves to C++.
Advice...New Intro to Programming Course using C++	M.M.		3/25/14	You might sucessfully argue Python. It is a great starter language, and is what MIT uses in their intro course...see their online course materials. They are also a nice resource.
Advice...New Intro to Programming Course using C++	J.M.		3/25/14	Stroustroup has written an introductory programming book based on C++. I also like Astrachan's Tapestry book, which you can get for free. JMM

Figure 3.2. Emails organized in an Excel sheet

The subjects, email content, and the dates were publicly accessible and copied from the listserv to the columns in the Excel spreadsheet. However, some members also provided their names, email addresses, school names, teaching titles, and additional contact information in the

signature area. That information was also included to the spreadsheet when available. Because members responded at different days and times to various topics, all the communications in the spreadsheet were grouped by subject and then sorted by date for each subject. Thus, the conversation starters and the context of each communication were seen clearly.

After organizing the data, the spreadsheet was imported to Nvivo qualitative data analysis software (version 10.2.0, MAC edition) and the Nvivo file was named "Phase 1 Email Analysis." Guided and coded by the preliminary framework variables and used as codes, the content analysis technique (Weber, 1990) was used to analyze the data. Fraenkel and Wallen (2010, p. 472) define content analysis as "the analysis of the usually written concepts of a communication." To do the phase 1 analysis, the researcher initially met one of the dissertation co-chairs to analyze the first 50 emails together in order to develop the procedure for data analysis. Following, the researcher explained the procedure to the methodology expert from the dissertation committee, reviewed the analysis procedure and received her feedback. Both faculty members' feedback and recommendations were applied to the content analysis procedure.

In the analysis, most preliminary framework codes were aligned with the data, but also new codes were emerged, some codes were combined, and some were deleted. For instance, even though "knowledge of content" is part of the ISTE-CSE standards and had a significant place in the preliminary framework, the data provided little to no evidence for in-service teachers' needs related to CS content knowledge. In addition, a new code was created after the analysis and labeled as "what is CS education". This code emerged from a curricular discussion about secondary-school-level CS curriculum. Furthermore, even though demographic information (e.g., the number of students in each class) was an important variable in the preliminary framework, it did not appear as a major issue for teaching CS. Only

underrepresentation of females and minorities was found as a concern in teachers' communications in terms of demographics.

This first phase served other purposes as well. After the analysis, the U.S. secondary teacher population was defined with more detail. Since the definition of a CS teacher in the US was not clear at the time of this study (Barr et al., 2013), phase 1 helped the researcher to define who a CS teacher was, where they taught, and what courses they taught in the US.

Participants

Even though it was also important to have CS experience in earlier grade levels (Kelleher, Pausch, & Kiesler, 2007), due to limited elementary CS teachers and teaching in the US, this research did not target K–5 teachers. The phase 1 data also supported this claim that there were not many elementary teachers participated to the email listserv discussions. The participants were teachers who had experience teaching CS content in a secondary level setting. They were teaching in a public, private, or charter school, or in an after-school program. However, most of them taught in formal school settings. In order to define those secondary CS teachers, their email communications were examined for evidences of a U.S. secondary-level CS teacher identity. The examples of these evidences included sharing a school name within the signature or providing evidence within the body of the email. For example, one email stated, "I need your help as I've never taught Computer Science at the high school level before." All the members who mentioned or provided evidence of teaching CS in secondary level were included in the analysis. In addition, when there was no opposing evidence found but the content was related to secondary CS education, those members' emails were also included in the data. Emails from academics, students, professionals, and from .edu domains were also read but only included in the analysis when an email was important to understand the context or content of a discussion.

Who Is a Secondary CS Teacher in This Study?

Based on the phase 1 analysis, members of the listserv who mentioned topics such as the following were considered to be CS teachers for the purposes of this study:

- teaching programming,
- AP computer science,
- computer hardware,
- robotics,
- game design,
- app development,
- algorithms,
- the impact of computers in the modern world.

Table 3.1 shows examples of how these teachers identified themselves in email signatures.

Table 3.1

Teachers' identification of their professional titles in email signatures

Business Education Instructor/Teacher	Computer Teacher	Math and Computer Science Teacher	Programming Instructor
Art, Design and Technology Teacher	Computer Science Teacher	Mentor in a CS club in a high school	Technology Integration Specialist
Computer Science and Math Teacher	Technology Department Chair	Math and Technology Teacher	Technology Educator
Instructional Technology Teacher	Computer Technology Teacher	Educational Consultant	Technology Teacher
Information Technology Teacher	Career Tech Teacher	Computer programming, game design and web design teacher	

Computer Science Teachers' Needs (CSTN) Framework

After completing the content analysis, the revised and newly created codes were synthesized to create broader categories (see Table 3.2), which constituted the CSTN framework. Those categories were used in phase 2 to categorize and identify CS teachers' needs in the teachers' communications.

Phase 2: Understanding Needs through Email Communications

Data Collection

In the second phase, the researcher extended the email data and used the CSTA email listserv communications between January 1, 2013 and October 1, 2015. This data was organized the same way as in phase 1 in a spreadsheet. The organized spreadsheet data was then imported to a new Nvivo project file and labelled "phase 2 email analysis."

Table 3.2

CSTN framework categories

Codes emerged from phase 1 analysis	New framework categories created
Curricular knowledge What is Cs education Assessment knowledge Curricular materials	Curricular Needs
Pedagogical content knowledge In-class collaboration	Pedagogical Needs
Student diversity Student beliefs and interest Student preconceptions Special needs	Student Related Needs
Knowledge of organizations for PD Updates on standards	Professional Knowledge and Skills Needs

Colleagues Community of teachers Parents Administration	Stakeholder Related Needs
Funding School planning Time and workload Equitable and accessible resources Tools and equipment PD resources	Resource Needs

Data Analysis

Using the categories in the CSTN framework (see Table 3.3), the email analysis started from the beginning from January 1, 2013.

Table 3.3

CSTN Framework

Framework Categories
Curricular Needs
Pedagogical Needs
Student Related Needs
Professional Knowledge and Skill Needs
Stakeholder Related Needs
Resource Needs

Content analysis technique (Weber, 1990) were used to organize the email communications under the need categories identified in the CSTN framework. After the content analysis, the researcher conducted thematic analysis of each category to identify specific CS

teacher needs with evidences. Thematic analysis "is a method for identifying, analyzing, and reporting patterns (themes) within data" (Braun & Clarke, 2006, p. 6). The thematic analysis followed the Braun and Clarke (2006) guidelines: (a) familiarization with the data, (b) initial coding, (c) searching for themes, (d) reviewing themes, (e) defining and naming themes, and (f) reporting. These themes are reported in the results section with evidences as initial findings of secondary CS teachers' needs. Furthermore, the themes were also used for developing the questionnaire for the phase 3 data collection.

Participants

The phase 2 data included 1,706 emails sent among all listserv members. 482 unique members participated in the discussions in the listserv. Based on the selection criteria explained in phase 1, there were 1,163 emails sent by 338 unique secondary CS teachers in phase 2. The participants were similar to the teachers identified in the first phase and mentioned teaching secondary level CS courses/content and/or identified themselves as a secondary CS teacher in the email signatures.

Questionnaire Development

The findings from phase 2 were used to develop the questionnaire (see Appendix C), based on the principles of survey design suggested by Groves et al. (2013). Creating the questionnaire from an examination of the rich communications between the teachers helped improve the validity of the instrument. Validity is an important concept in quantitative research to ensure "the appropriateness, meaningfulness, correctness, and usefulness" of a survey instrument (Fraenkel & Wallen, 2010, p. 148). One type of validity provided by this instrument is content validity. Content validity refers to the appropriateness of the items and comprehensiveness of the instrument (Fraenkel & Wallen, 2010). By using in-service CS teacher

communications from the email listserv, this research increased the content validity of the questionnaire.

Phase 3: Questionnaire

This phase includes revising the questionnaire, sending it out to participants in the CSTA CS teachers' email database, and analyzing the questionnaire data. In the first step, a think-aloud protocol was used to test the questionnaire with potential participants in order to increase instrument reliability and validity (Collins, 2003).

Data Collection

Although all efforts are made to ensure instrument validity, any survey measurement can be open to measurement error. A think-aloud protocol is a qualitative technique used in usability testing of a design and can be used for pretesting a questionnaire to reduce measurement error (Collins, 2003). Fraenkel and Wallen (2010, p. 150) emphasized this as a way to "obtain content related evidence of validity" (2010, p. 150). Patton (2015) defines the purpose of think aloud as to "elicit the inner thoughts or cognitive processes that illuminate what's going on in a person's head during the performance of a task" (p. 486). In this study, the researcher aimed to examine the participants' response to each questionnaire item to ensure that all participants shared the same understanding. The researcher followed Alwin (2010)'s guidelines for pilot testing the questionnaire to reduce possible measurement error and increase content validity in this study:

1. questions asked are appropriate and relevant, and all have answers;
2. questions are posed so that respondents or informants understand what information is requested;
3. respondents or informants have access to that information;
4. they can retrieve the information from memory;

5. they are motivated to report the information retrieved accurately; and
6. the response categories provided by the survey question allow them to communicate this information. (p. 406)

These steps allowed the researcher to increase the reliability and validity of the questionnaire design by checking the participants' comprehension, judgment, retrieval, and reporting of each item (Collins, 2003). By checking comprehension, the researcher noted the participants' interpretation of each question and compared it with the research purpose. By checking judgment, the researcher noted whether the participants delivered the response being requested. By checking retrieval, the researcher tested whether the participants were able to retrieve information from memory. Finally, by checking reporting, the researcher understood how a participant decided on a final response for each item.

Pilot Testing and Feedback Drawn

The first pilot participant was drawn from a convenient population of the researcher's former students, who had completed a CS education licensure program and was at that time working as a CS teacher in a U.S. public school. Following, five in-service secondary teachers were contacted using a snowball sampling method (Biernacki & Waldorf, 1981) and three more agreed to participate in the pilot testing.

The researcher conducted the pilot testing with the first two participants without making any revisions in the questions asked between the two sessions. Both participants reported the following similar issues in their comprehension and reporting level:

1. Difficulty understanding the question

Old: What CS courses do you teach?

New: What CS content do you teach currently?

2. Misunderstanding some items

Old: I need to learn how to better mentor students while coding (answering simultaneous questions while coding).

New: I need to learn how to answer my students' simultaneous questions while coding (e.g., finding errors in code, debugging, reading code).

3. Recommendation on separating a question and adding new questions

Old: I need to learn strategies to motivate girls and underrepresented populations (African-American, Hispanic) to enroll in my CS classes.

New 1: I need to learn strategies to motivate girls to enroll in my CS classes.

New 2: I need to learn strategies to motivate underrepresented populations (African-American, Hispanic) to enroll in my CS classes.

4. Difficulty selecting the response categories (see Figure 3.3)

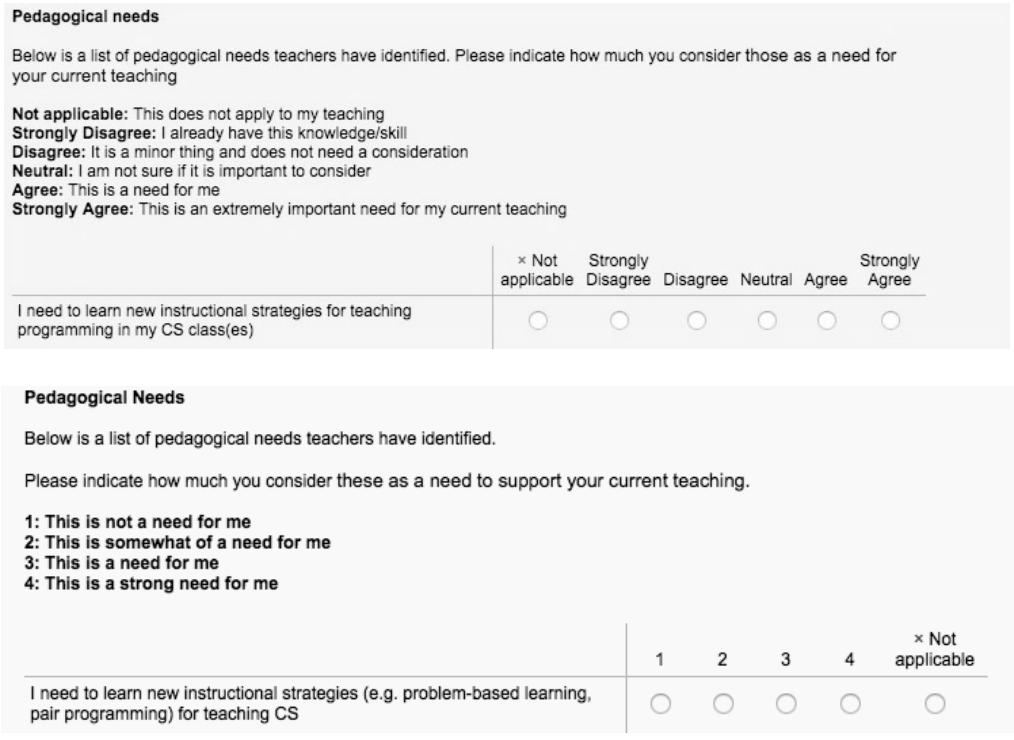

Figure 3.3. Difficulty selecting response categories, old and new versions

After these two think-aloud interviews, the researcher met with the dissertation co-chairs and addressed the necessary revisions and recommendations. The third think-aloud participant highlighted some grammatical issues. The researcher revised those questions and conducted the fourth testing with no major issues highlighted. The pilot testing with the qualitative think-aloud testing increased the content validity and reliability (Fraenkel & Wallen, 2010) of the questionnaire items based on the participants' feedback. Furthermore, all four pilot-test participants were asked their opinions about the comprehensiveness of the questionnaire to test face validity of the instrument (Groves et al., 2013). All respondents provided positive feedback. One teacher said, "I like it. It covers a lot of my needs: what I am teaching about the curriculum, to recruiting, to teaching, strategies for teaching, materials, computers, hardware and software." Therefore, the questionnaire established a successful harmony between the research purpose and the participants' reflections. At the end of the pilot testing process, the researcher received the dissertation co-chairs' confirmation of the instrument.

Before the questionnaire was made available online, the researcher sent the final questionnaire to five peers in his department to determine the time it takes to complete it. This testing confirmed that, on average the questionnaire took between five and eight minutes to complete. This information was also added to the description, and the questionnaire was then made available online using the Indiana University Qualtrics survey system.

Questionnaire

The questionnaire started with the study information sheet (See Appendix C), explaining the study purpose and procedure to participants, including Institutional Review Board (IRB) permission to conduct the research. The study information sheet also included confidentiality,

contact, and voluntary participation information. On the next page, participants were asked to sign a participation agreement (see Figure 3.4).

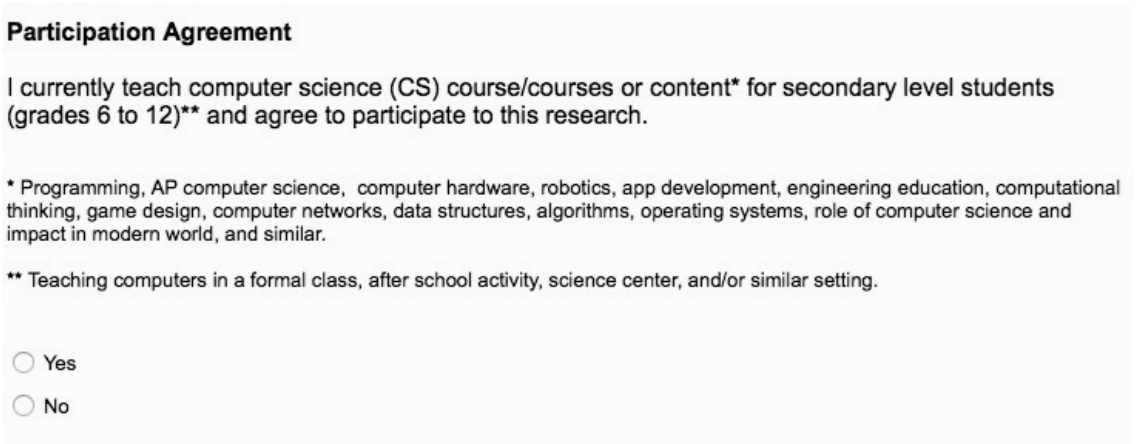

Figure 3.4. Screenshot of participation agreement, retrieved from the questionnaire

This participation agreement was based on the definition of a computer science teacher that emerged from the phase 1 data analysis. If a participant decided not to accept the agreement, the system ended the questionnaire with a thank you message. If a participant agreed to move forward, there followed a short description about answering the questions with current CS teaching in mind followed: "Think about your current situation in which you are teaching CS. What are the things you need to make yourself an even better CS educator? Keep this in mind as you respond to the following survey items." This explanation was included after conducting the think-aloud pilot testing with the first two participants. Providing this explanation, the researcher aimed to help participants to focus on their "current situation" when answering the questions at the time of the questionnaire data collection.

Based on the CSTN framework and research purpose, the final questionnaire included seven categories:

- pedagogical needs,
- curricular needs,

- student related needs,
- professional knowledge and skill needs,
- resource needs,
- stakeholder needs,
- demographic questions (e.g., years of experience teaching CS, background, gender, school type, school level).

The number of total closed-ended questions was 40. Each need category included an open-ended item for data complementarity (Daigneault & Jacob, 2014) by giving participants the opportunity to add other needs or information that were not listed in the instrument. There were some demographic questions asking the participants' gender, years of experience teaching CS, background in CS, school type, teaching level, and CS courses taught. In addition, at the end of the questionnaire, the researcher asked participants to voluntarily enter a raffle to win one of three $50 Amazon gift cards and/or participate in a follow-up interview.

Distributing the Questionnaire

The researcher consulted the CSTA executive board and asked them to share the questionnaire invitation in their email listserv. In fact, after reviewing the study purpose and questionnaire, the CSTA board decided to support the study and distributed it widely to all secondary CS teachers in their database. This created a much larger data pool, as the database included all secondary teaching members, while teachers signed up for the voluntary email listserv. The database included 7,356 secondary teachers (about 6,000 from US). The board sent the questionnaire invitation (see Appendix B) to the list twice. In the first attempt on January 19, 2016, 1,100 members opened the email invitation. In the second attempt, on January 26, 2016, the questionnaire was sent to the same population and 1,308 members opened the email

invitation. The researcher was unable to tell how many different members opened the email and accessed the questionnaire in the second attempt. After both attempts, 303 people partially or fully completed the questionnaire.

Data Analysis

Quantitative Data

Initial data organization and cleaning was conducted in both Excel and the Qualtrics system. Calculations were made using SPSS, a statistical analysis software package produced by SPSS Inc. Overall frequency scores were reported and interpreted to explore secondary CS teachers' needs. In order to interpret the secondary teachers' needs and how those changed based on their years of experience and background, the researcher reduced the response scale from four categories to two. The initial response categories included (1) This is not a need for me, (2) This is a slight need for me, (3) This is a need for me, and (4) This is a strong need for me. Due to the descriptive nature of the questionnaire items and response categories, the first two response categories were grouped together as "not a need" and the third and fourth response categories were grouped as "need." The third and fourth responses showcased that teachers indicated those as something they needed to work on. Although the second response indicated that it was a slight need, it is likely that this need is not as important to address as the other more critical needs (responses 3 and 4). Previous studies highlighted that respondents in some cultures do not like to use extreme options and suggested combining extreme responses with less extreme options (Harkness, Mohler, van de Vijver, 2002). Other studies have utilized Likert scale with similar wording and have grouped response scales. For instance, a national study on the quality of undergraduate education in Indiana., the researchers combined students' Strongly Agree/Agree and Strongly Disagree/Disagree responses to report their safety concerns (NSSE, 2016).

Furthermore, the quantitative data analysis did not include any inferential statistics. The primary purpose of the study was descriptive. A descriptive study was necessary because there were few studies available aiming to describe secondary CS teachers' needs with the recent national changes in CS education. Therefore, as a descriptive study (Creswell & Clark, 2003), the current work was designed to provide a basic overview of the current state of CS teachers needs, but the researcher did not aim to reach conclusions that apply to the general population. Using this data, however, the researcher does plan to follow-up on this in future studies.

Need comparisons across different years of experience (less than 4, 4–10, more than 10) and background in CS were reported using frequency for all the items under the seven categories of the CSTN framework categories. The years of experience group intervals were created based on Darling-Hammond (2000)'s study on teachers' practices. Darling-Hammond determined that the fourth year and tenth year are appropriate evolution points (4^{th}, and 10^{th} years) of teachers' experience and development in their job.

Based on the email analysis results, there were initially 15 different categories of background options used in the questionnaire (See Appendix D). However, based on previous research and national reports, the researcher only combined some initial categories and reported the comparisons across:

1. overall sample,
2. people with CS background (no education background),
3. people with no CS (had an education degree other than CS education),
4. people with CS education degree (by itself and with other categories).

For example, seven teachers reported "degree only in CS." Instead of comparing this group against others, the researcher combined this category with:

- degree in CS and attended professional development workshops,
- degree in CS and had work experience,
- degree in CS, attended professional development workshops and had work experience.

Compared to work experience and professional development experience, degree in CS was reported as more influential in teaching in previous research and national reports. Table 3.4 below shows the number of teachers in the comparison groups.

Table 3.4

Details of background options

Initial Background Category	Number of Teachers in the Initial Category	Initial Categories Combined Under
• Degree only in CS	7	People with CS background (no education background)
• Degree in CS and attended professional development workshops	26	
• Degree in CS and had work experience	11	
• Degree in CS, attended professional development workshops and had work experience	31	
• Only attended professional development workshops	47	People with no CS background (had an education degree other than CS education)
• Only had work experience	6	
• Attended professional development workshops and had CS work experience	11	
• Degree only in CS education	10	People with CS education degree (by itself and with other categories)
• Degree in CS education and attended professional development workshops	9	
• Degree in CS education and had work experience	2	
• Degree in CS education, attended professional development workshops and had work experience	12	

Cronbach Alpha reliability was used to test the internal consistency of the items in the instrument (Gliem & Gliem, 2003). The researcher calculated the Cronbach Alpha values for each framework category and overall questionnaire. George and Mallery (2002) suggested the following rule to test the values: "_ > .9 – Excellent, _ > .8 – Good, _ > .7 – Acceptable, _ > .6 – Questionable, _ > .5 – Poor, and _ < .5 – Unacceptable" (p. 231). All the tests performed provided acceptable to highly reliable values:

- pedagogical Needs (5 items; α=.76),
- curricular Needs: (5 items; α=.83),
- student Related Needs: (6 items; α=.84),
- professional Development Needs: (4 items; α=.81),
- stakeholder Needs: (5 items; α=.84),
- resource Needs: (5 items; α=.84),
- overall: (30 items; α=.92).

Qualitative Data

The phase 3 data analysis was conducted using constant comparative method of qualitative data analysis (Glaser, 1965). Glaser defines purpose of the constant comparative method as providing alternative to analysis, comparing themes and looking for agreements and conflicts ("negative cases or a consideration of alternative hypotheses"), and increasing credibility in study results.

Phase 2 analysis from the email data provided the researcher initial identification of the CS teachers needs using content and thematic analysis techniques. In phase-3, the researcher started used constant comparative method to find and compare themes within and across data collected. Initially, qualitative data from the questionnaire participants' responses were

downloaded from Qualtrics as MS Office document files. Following, a copy of the phase 2 Nvivo analysis was created and renamed as phase 3 analysis. The document files imported to phase 3 Nvivo project for constant-comparative analysis of the qualitative data.

In the constant-comparative method, data analysis included comparing new data with the previously collected and coded data in phase 2. When compatible evidence emerged in this phase, that piece was coded under the previously identified themes. For instance, in phase 3, a participant shared: "I need help teaching concrete thinkers to see things more abstractly." and this coded as "*p3 teaching computational thinking*" under the pedagogical needs category. P3 refers to phase 3 and the rest of the name was retrieved from the code developed in the phase 2 analysis. In the phase 3 analysis, new codes also emerged from the qualitative responses in the questionnaire. For instance, in phase 2, the researcher did not find students' special needs as a concern to address in CS classes. However, teachers in the questionnaire mentioned needs related to teaching students with disabilities. For instance, when asked to provide other needs in the pedagogy section of the questionnaire, one teacher mentioned: "How to better understand how dyslexia affects a student's coding and problem solving," and this coded as "p3 student disability." In terms of conflicting findings, even though previous data showed knowledge of content as not a need for CS teachers, the qualitative data in the questionnaire provided multiple evidences of this as a need: "I am trying to learn the languages I am teaching students—such as JavaScript, Python, HTML, CSS, JQuery, and Ruby—so I can be more comfortable with answering student questions, as well as becoming more proficient myself." In some instances, the same response was coded two times because the teacher was referring to two different important themes in the same response. For instance, the following response was coded both as a "curricular" need and the theme related to "background in teaching CS":

> I teach Python (Computer Programming 1) and Java (Computer Programming 2) languages, but I need more resources since I did not come from the IT field; I am a business teacher with programming (COBOL) experience from my bachelor degree back in the late 1980s. I need resources to help me teach current industry programming standards and implementation. Java is my bigger concern, but I can use assistance with Python as well.

While doing the analysis, the researcher also created memos for some of the codes in a word document to record his reflections about the important points in the data (Bazeley & Jackson, 2013).

Participants

Of the total 303 questionnaire participants, 14 participants opened the questionnaire but did not answer any questions, including the participation agreement. Another 11 participants (4 %) responded "no" to the participation agreement. This may be due to participants' lack of fit in the population. When these participants were removed from the sample, 278 participants fit the population and agreed to participate. In that 278, there were 18 participants located outside the US, from 10 different countries [Canada (4), China (1), Germany (2), India (3), Nigeria (3), Pakistan (1), Philippines (1), Saudi Arabia (1), Thailand (1), and the United Kingdom (1)]. When these participants were removed from the sample, 260 participants were left. Out of that 260, 222 participants fully completed the questionnaire and were included in the analysis.

The participants were from 43 states and the District of Columbia in the US (no respondents from Arkansas, Delaware, Idaho, Minnesota, Oregon, Rhode Island, and Alaska). Texas (18), California (16), New York (13) and Pennsylvania (13) were the states with the highest number of participants. Out of 222 participants, 121 (54.8 %) participants listed their gender as female, 98 (44.1 %) male, one selected "other," and one another chose "Decline to Answer." In terms of school type, 170 participants were teaching in public schools, 44 in private schools, 9 in charter schools, and 12 in technology centers, after-school programs, science

centers and similar. Some of the participants reported teaching in multiple school types (e.g., both public school and technology center). In the "other" section for this question, five participants provided examples of where they taught, including teaching in a "public STEM [Science, Technology, Engineering, Math] school," "career technical education," "virtual" and "private students."

In terms of grade level, 212 participants stated they were teaching in high school, and 26 in middle school. Some participants reported teaching in both middle and high school. In terms of years of experience, 77 teachers had less than 4 years (34.8 %), 58 teachers (26.2%) between 4–10 years, and 86 teachers (38.9%) more than 10 years' teaching experience in CS. In terms of background in both CS and CS education, the participants were able to choose multiple options from the following choices:

- degree in CS or a related field (e.g., minor, major, undergraduate, graduate),
- degree or license in CS education (e.g., undergraduate, graduate, licensure or endorsement),
- attended professional development workshops/courses (face-to-face or online).

In terms of background, 75 teachers reported background in CS (33.78 %), 33 teachers reported background in CS education (14.86%), and 64 teachers reported no background in CS or CS education (29.28%). When asked to provide a "list of any other education or experience," 80 teachers provided examples of a variety of backgrounds in the comments section. Appendix E provides selected examples (quotes from the participants' comments) for different backgrounds and experiences.

The qualitative data from 80 respondents was imported to Nvivo and analysis of their responses was also conducted using content analysis. The qualitative data showed that very few

teachers (8) reported having only a license or education toward teaching CS in K–12 as their main preparation for teaching CS. As one teacher reported, this is an issue for effective CS teaching: "I did not go to college for this area and have very little experience in this subject." However, more mentioned having degree in CS or a related field (15) in CS. Many people (44) from variety of fields without a degree in either CS or CS education reported teaching CS courses in the US. These people are self-learners with no CS background, people from other fields (e.g., business accounting, experimental physics), and teachers from other education areas who only attended professional development workshops.

Phase 4: Interviews

The final interviews aimed to expand and validate the findings from the previous phases with examples and descriptions, and then to triangulate the data with complementary qualitative evidences (Fraenkel & Wallen, 2010).

Data Collection

The interview questions were developed to gather more information on secondary teachers' needs. A semi-structured interview protocol (Fraenkel & Wallen, 2010) was used. Fourteen teachers who voluntarily agreed to participate in the follow-up interviews were purposefully selected and invited, based on their responses to the questionnaire, years of experience, and background.

The researcher developed the interview questions from the questionnaire items. Following was an example item in the instrument. In this item, respondents were asked to rank their needs, with "1" meaning no need, and "4" meaning a strong need:

	1	2	3	4
I need materials to assess my students' learning in my CS classes	O	O	O	O

When a teacher identified this as a need and selected 3 (This is a need for me) or 4 (This is a strong need for me) from the response categories, the following interview question was asked: "In the questionnaire, you reported that you need materials to assess your students' learning in your CS classes. Can you explain your need in more detail and provide me examples of the need in your CS teaching?" For selected participants, the same question format was followed and replaced with all the items that a participant reported as a need (3 or 4).

In terms of background, purposefully selected participants from different backgrounds were asked "How has your background in [CS], [CS education], [and] [professional development workshops] influenced the way you teach your CS classes?" Follow-up questions were asked to generate more details about the participants' responses. In terms of years of experience, the selected participants in different year ranges were asked: "How is your CS teaching changed in # years?" If it was a new teacher (less than 4 years), the participant was asked: "What kind of challenges/difficulties do you have as a # year CS teacher?"

The interview data collection ended when the researcher decided that the data was saturated and represented the population, as suggested by Lincoln and Guba (1985) ("(a) exhaustion of sources, (b) saturation of categories, (c) emergence of regularities, and (d) over-extension" (pp. 125-126).

Data Analysis

The interviews were fully transcribed. The teacher names and all the identifying information were replaced with pseudonyms. The phase 3 Nvivo analysis file, which included all the previous themes and analyses, were copied and renamed as phase 4 analysis. Following, all interview transcriptions were imported to the phase 4 analysis file in Nvivo. The new imported interview data was analyzed using the constant-comparative method in the same way as

conducted in phase 3 to triangulate the findings. The researcher looked for agreements and conflicts with the previous data. This phase also helped the researcher provide more descriptive information about CS teachers' needs with examples from the participants' explanations of their practices.

In phase 4 analysis, when a theme aligned with a previous theme in phase 2 or phase 3, the same node name was used in Nvivo, with the P4 prefix. For instance, time and high workload was consistently reported as an issue and this theme was coded as lack of time - high workload in all the phases (see Figure 3.5, highlighted in yellow).

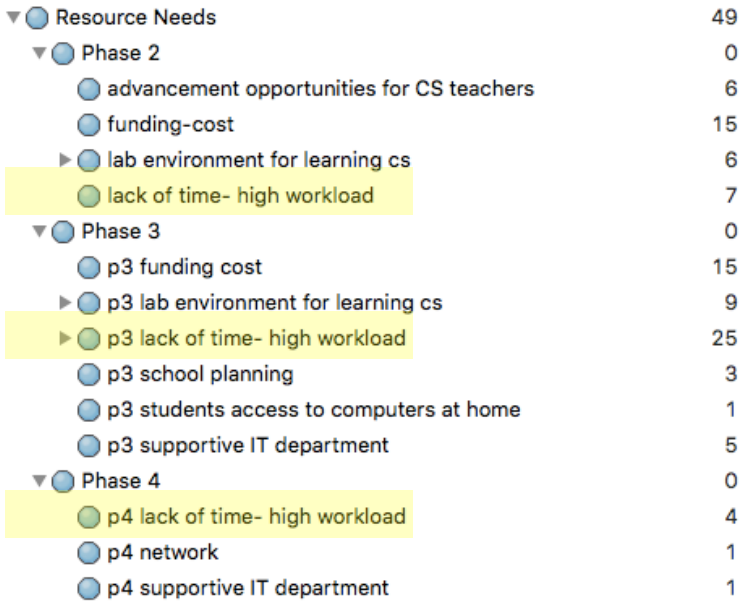

Figure 3.5. Lack of time - high workload coding

New themes also emerged during the phase 4 analysis. For instance, when the researcher asked participants about their professional development needs, one participant highlighted his need to participate in professional development events that are specifically designed by people from high schools: "If colleges could offer that, that's great, but the college model of education is dramatically different than the high school model. I would like to see there be more high-school-generated professional development than just college." This code provided another level of

description of the “need for continuous pd” and coded as “high school generated pd” under the professional knowledge and skills needs in phase-4.

Conflicting codes also emerged in the phase 4 analysis. Even though phase 2 and phase 3 analyses provided evidence for the importance of aligning curriculums with the CS education national and/or state standards, one of the participants highlighted that is not necessary for CS teaching:

> I think that the standards should support the teaching not lead the teaching. Teachers should be familiar with what the kids need to know. The teacher should be familiar with the standards and should incorporate those in a variety of ways into their projects, but I don't want to have projects designed just to meet the standards. (I-5)

Therefore, this was coded and named, as “p4 conflict cs standards should not lead teaching,” in which “conflict” refers to an evidence for ideas that conflict-with previous themes (see Figure 3.6, highlighted in blue).

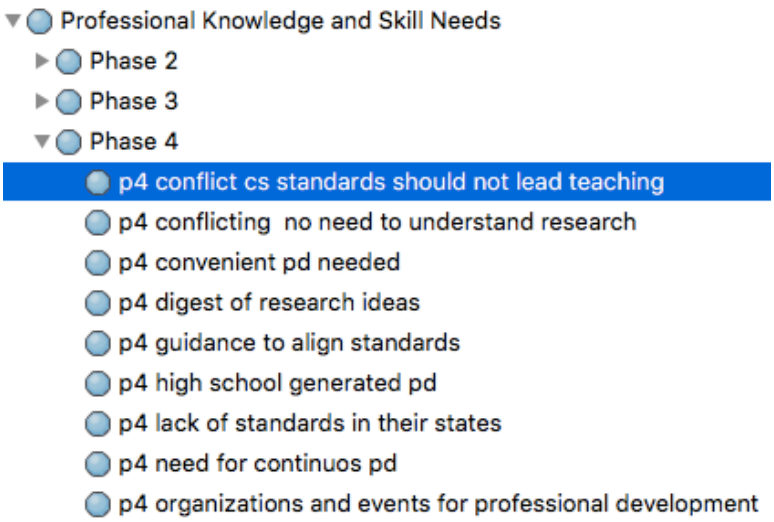

Figure 3.6. Phase 4 conflicting evidences example

Participants

Eight CS teachers from different states participated in the interviews. The eight interviews took 29, 30, 30, 43, 46, 54, 55, and 68 minutes, depending on the participants' responses to the questionnaire. Table 3.5 illustrates the participants' information about their level of teaching, years of experience, background, and school setting.

Table 3.5

Interview participants

Pseudonym	Level of teaching	Years of Experience	Background	School Setting
I-1	High school	13th year	BS in CS, background and minor in Physics, no teaching license	Public, high school, training other teachers online and face-to-face in CS.
I-2	High School	5th year	BS in CS, Industry career certification, no teaching background	Public school, 250 students, 30 % minority.
I-3	Middle and High School	3rd year	No CS, Industry work experience, alternate resident educator license (25 hours program)	Public school, 2000 students, Diverse school 15 % minority
I-4	High School	20th year	Doctorate in educational technology, Masters in CS.	Public school, IT Director of the district, low teaching load.
I-5	High school	15th year	Background in CS, no teaching license	Private school, extensive technology resources, very low minority
I-6	High school	3rd year	Degree in Business Education, attending PD in programming,	Public, 30% African-American and 5% Hispanic

I-7	Middle and High school	8th year	Degree in education, business teacher, math for business certification, attended professional development	CS are electives, Low minority in the school, 95 % white students, 300 students in the school.
I-8	Middle and High school	1st year	Degree in computer science, learned CS in high school from freshman year in a technology magnet school, teaching credentials in math and science, completed the PLTW workshop	Public independent charter school. School size is about 750 students. Scratch, App Inventor, Python

Member Checking

All the interview participants were contacted and three participants indicated their willingness to participate in member checking. These participants were sent a draft of the dissertation manuscript and were asked to review the findings and discussion chapters. Two provided feedback if the report represented their experience and possible needs other secondary CS teachers may have with regards to teaching CS. In online conversations, all the member checking participants shared their satisfaction with the study findings and provided minor recommendations. All the recommendations were incorporated into the final version.

CHAPTER IV: FINDINGS

Findings are divided into six main areas covering the following categories of needs that were identified using the CSTN framework:

1. pedagogical needs,
2. curricular needs,
3. student-related needs,
4. professional knowledge and skills needs,
5. resource needs,
6. stakeholder-related needs.

The researcher used multiple data sources to explore the needs across one computer science teacher organization:

1. teachers' communications from the email listserv,
2. quantitative data from the questionnaire (close-ended questions),
3. qualitative data from the comments in the questionnaire (open-ended questions),
4. qualitative data from the interviews,
5. quantitative and qualitative data across all study phases about how years of experience and background in CS (demographic information).

Quotations were derived from the teachers' email communications, comments in the questionnaire and the interviews. Within the findings section, email quotations are marked with "E," questionnaire open-ended quotes marked with "Q," and the follow up interviews marked with "I." Each quotation is also identified with a number indicating the unique participant.

Pedagogical Needs

Pedagogical needs can be defined as needs for teaching strategies that can improve students' learning. The pedagogical needs were identified from CS teachers' communications in both the listserv and questionnaire, and expanded on with the interview data. For the purposes of this study, teachers' perceived pedagogical needs included the following sub-themes:

1. new instructional strategies for teaching CS,
2. strategies for teaching computational thinking,
3. transferring skills between programming languages and platforms,
4. answering students' questions,
5. facilitating student interaction and collaboration.

New Instructional Strategies for Teaching CS

Secondary CS teachers stated that they need more instructional strategies to teach CS content and enhance student learning in their courses, which includes student centered learning strategies such as problem-based learning and pair programming.

Email listserv findings. Regardless of the focus of their courses (e.g., Systems Analysis, Linux, Programming), teachers in the email listserv expressed a need to learn new instructional strategies for teaching CS courses. Out of 35 teachers that shared pedagogical needs in the email listserv, 18 teachers asked for new instruction ideas to use in their classes. For example, one teacher wanted to know how to teach Linux with new instructional strategies: "I think it's really important for my students to learn Linux but I have no idea how to teach it. I learned by fumbling blindly in the dark freshman year of college" (E-2). Another teacher asked for advice on new instructional strategies that would help change her teacher-led classroom into a hands-on student-centered learning environment: "We are introducing Systems Analysis and Design

course for our seniors this semester. Any ideas/strategies on how to make this more hands-on and student-friendly as opposed to rote lecturing and presentations" (E-1).

In addition to searching for new ideas on the email listserv, 12 teachers asked for recommendations or feedback from other teachers about the instructional strategies that they had heard of or tried out. These teachers expressed a particular need to know more about student-centered ways of presenting CS content. For instance, problem-based learning was one of the student-centered strategies in one teacher was interested in:

> Rather than chalk and talk my way through a unit on Database (Information Systems). I thought problem-based learning may provide better results. I've included an outline of what I thought I might do, based on other stuff I've seen on the web. I'd appreciate anyone who has experience in problem-based learning offering some advice. (E-5)

Pair programming was another student-centered learning strategy that was discussed in the listserv. Pair programming uses a collaborative learning environment where two team students work together on the same programming task and design a solution (McDowell et al., 2002). One teacher shared her failure in using pair programming in a CS class and asked for advice from other CS teachers: "I have done pairing, but must not have done it correct, because it was not as productive as I'd liked. What ideas do you have about any of the above" (E-4)?

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement: I need to learn new instructional strategies (e.g., problem-based learning, pair programming) for teaching CS. Overall, 57% of the questionnaire participants (n=221) identified this as a need or strong need. In the open-ended questions, where teachers were asked to comment about pedagogical needs, 17 teachers asked for help on new instructional strategies for teaching CS, specifically related to problem-based learning and facilitating students' learning. One teacher expressed her need for information on problem-based learning for her Java programming class: "I am using hour of code to introduce programming. After that

my programming students are working on codes. However, it would be nice if we could teach Java in an interactive problem based method" (Q-1). While requesting help for better instructional strategies in teaching CS, one teacher also stressed the need for facilitating students' learning in student-centered learning: "I will appreciate further pedagogical help with teaching computer science and facilitating student knowledge, particularly helping students make better connections with the material, and more effectively debug without needing face time or one-on-one time from me" (Q-2). Another teacher emphasized his need for supporting student-centered learning via scaffolds in a computer-programming course: "I need better strategies for teaching computer programming to students who have never written computer programs before. What's best to teach first, second, etc. I want a scaffolded approach to teaching programming" (Q-3).

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related this pedagogical need. Seven out of eight final-phase interviewees expressed either a need or a strong need in the questionnaire for learning new instructional strategies. When asked to explain this need in more detail and provide examples in the interviews, they all described their current practices and explained why they want to learn new instructional strategies. For instance, I-2 described his current practice as "straight lecture" and explained it:

> I do a lot of straight lectures in my classroom, I do lot of type; I'll pull the projector, I'll put code on the projector and have kids follow through on their own computers, basically they copy down what I'm doing. (I-2)

The same teacher emphasized his need to learn new instructional strategies, problem-based and pair programming approaches for student-centered CS learning:

> I want to learn new instructional strategies that I never got, because I didn't go to a teaching school, I never went to school to learn to be a teacher. I feel that it's personal, that I needed to learn better how to do things like problem based learning in pair programming. (I-2)

Overall. The participants in all phases of the study shared their need for new instructional strategies, especially emphasizing the need for knowledge of student-centered ways of learning CS (e.g., problem based learning and pair programming). Furthermore, some of the participants already tried different instructional strategies and asked for feedback from other teachers to assess and improve their teaching.

Years of experience and background. Figure 4.1.1 shows the questionnaire findings for teachers with different years of experience and background.

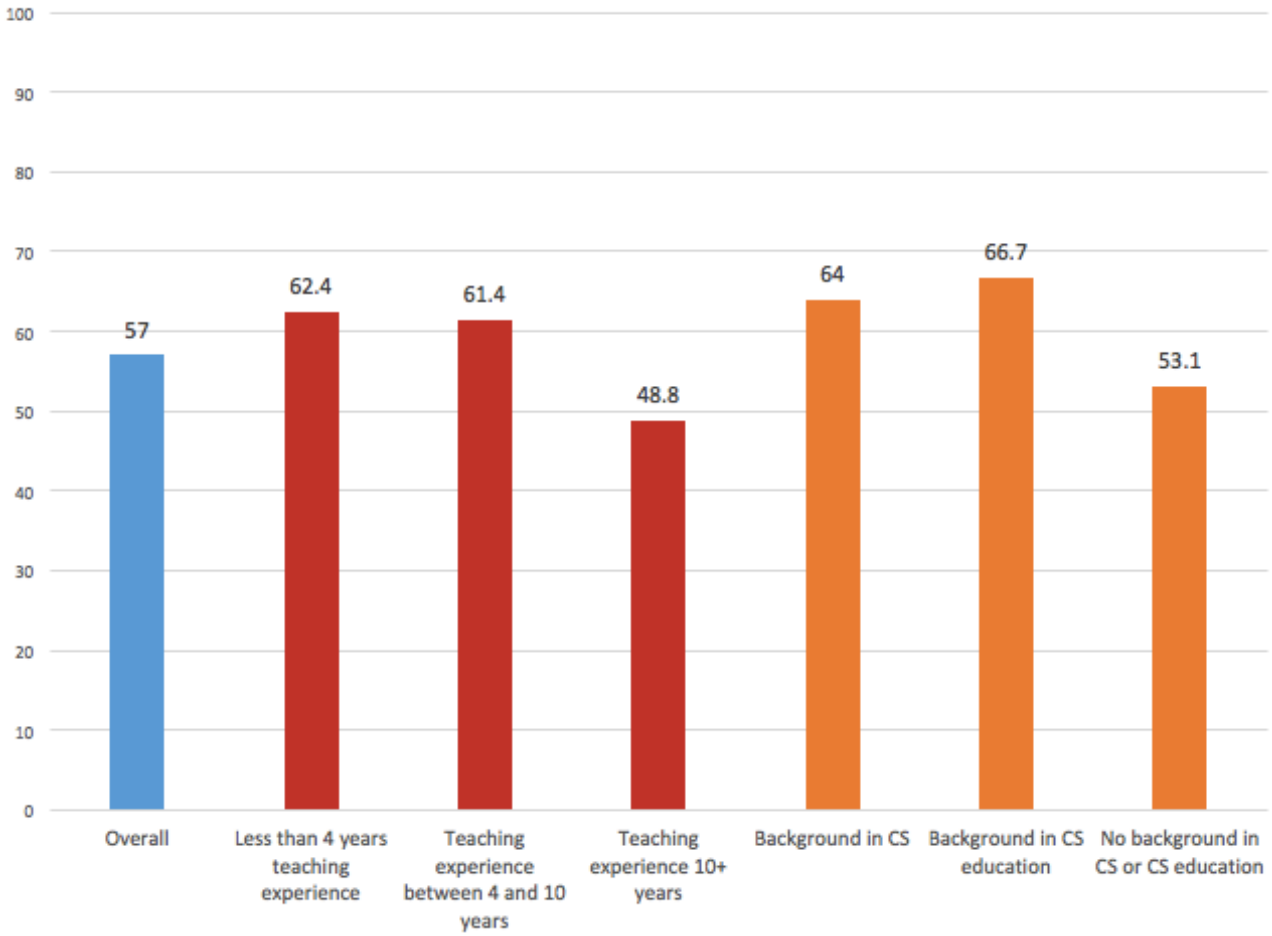

Figure 4.1.1. Frequency of teachers' perceived need for learning new instructional strategies.

Findings suggested that teachers with 10 years or more experience were less likely to report "learning new instructional strategies" as a need or strong need compared to less

experienced teachers. In other words, as teachers gained more teaching experience, it appeared that "learning new instructional strategies" became less of a need.

Teachers with no CS or CS education training were less likely to report "learning new instructional strategies" as a need or strong need. This could be due to the fact that most of these teachers with no CS or CS education background reported having a teaching license in a different area (e.g., math, science) and described that the pedagogical knowledge from their main teaching subject area helped them teach CS. In the phase 3 questionnaire comments, one teacher expressed his advantage as a math teacher using his knowledge of a variety of instructional tools: "And since I was a math teacher and had an awesome toolset for teaching…" (Q-7). In phase 4, one of the interviewees described that his early years were challenging as he looked for alternative ways to teach CS: "As a 1st, 2nd, 3rd year teacher, I was just figuring how, how to teach. I knew my content and I was looking for new innovative ways to make my content real and relevant to the students" (I-1). The same teacher continued to describe the influence of his background in CS, stating that although teachers from other areas had the tools and strategies for teaching, they may lack specific content knowledge in CS:

> I'm different from 90% of the other teachers because in Arkansas, most of the teachers that are starting to teach computer science already are teachers. They're teachers of other disciplines. They might be a business teacher. They taught Microsoft Word or keyboarding or something like that for a long time. They already know the technology. They know how to problem solve. They know how to plug it in and unplug it, but they don't know the computer science content. (I-1)

Another teacher emphasized his background in CS and his need to learn new instructional strategies to better teach it:

> Part of it's because of my background. My background is computer science, it's not education… I want to learn new instructional strategies that I never got, because I didn't go to a teaching school, I never went to school to learn to be a teacher. I feel that it's personal, that I needed to learn better how to do things like problem based learning and pair programming. (I-2)

Strategies for Teaching Computational Thinking

CS teachers stated that they need more strategies to help their students learn computational thinking, which primarily includes solving computational problems by purposefully designed solutions.

Email listserv findings. Out of 35 teachers that shared pedagogical needs in the email listserv, 13 teachers mentioned that they want to help their students attain computational thinking skills. Computational thinking is an approach to solving a computational problem by reformulating the problem and the solution (Wing, 2008). With computational thinking, students are expected to purposefully design solutions to the computational problems they encounter (Sengupta, Kinnebrew, Basu, Biswas, & Clark, 2013). Purposeful design in this study refers to developing a plan for the computational solution before transferring it to a computational product such as code. However, in the email listserv, teachers indicated that their students use a trial-and-error approach and lacked knowledge and skills for solving computational problems. One teacher described how trial-and-error approach was a limitation and highlighted using learning to program by designing solutions to computational problems:

> You are so right that a disadvantage is students seem to develop a trial-and-error approach to programming instead of learning to program by design. I find that a problem with what is popular in schools, robotics, where they code and then say, lets see if it works. And that carries over to math education. (E-7)

Another teacher shared the same concern for beginning-level students in a programming class, that students rely on trial-and-error instead of problem solving: "If beginning students have the choice between trial-and-error and thinking hard, they will often choose the former, even though it proves less efficient in the long run" (E-8).

This teacher described that trial-and-error did not help students develop problem-solving skills. Another teacher described the weakness of the trial-and-error approach and his inability to

get students to problem solving in their programming: “The students were happy with their creations, but overall, I considered such classes a failure because their independent programming and problem solving capabilities didn't grow the way I had hoped” (E-3). A common theme in these conversations was that the teachers in the listserv wanted their students to purposefully design a computational solution when programming, not just through trial and error: “I don't want my students to discover a solution - I want them to work purposefully toward solving it. Discovery and experimentation, to me, tastes too much like accidental” (E-9). Therefore, teachers highlighted their need for knowledge related to teaching computational thinking, as followed:

> It's mostly a change to a style of teaching where it's a reflective process between you and the students about where they are with their skills - and less about a progression of assignments and code samples. Keep the conversation going - what else can we do to bring computational thinking to students who have such big gaps? (E-10)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement: I need to learn how to teach computational thinking (e.g., problem solving, designing solutions) in my CS class(es). Overall, 61.6% of the questionnaire participants (n=219) identified this as a need or strong need. Eleven teachers shared comments about needing more knowledge of how to teach computational thinking, especially integrating problem solving: “The biggest need for me is helping kids with being able to read a problem and understand the parts, how they are related and how to make it work” (Q-4). Another comment stressed the importance of teaching computational thinking in CS classes by including problem solving: “Big need is how to teach problem solving skills. Maybe lists of problems that can help teach problem solving” (Q-5).

Interview findings. Before conducting the interviews in phase 4, the researcher considered the participants’ questionnaire responses to this need. Five out of eight final-phase

interviewees expressed either a need or a strong need for learning how to teach computational thinking. When these teachers were asked to explain this need in more detail and provide examples in the interviews, most teachers emphasized the problem-solving aspect of computational thinking in programming and CS, and the need to better help students solve computational problems. One teacher explicitly shared her need to teach students how to solve computational problems:

> I feel like I don't know well how to teach people how to think how to solve problems; they run into problems under code, and I'm having a hard time getting them to figure out, on their own, what the problem is. Some students can inherently do that, and that's great and fine; but those who can't, I'm having problems teaching them how to. (I-2)

In addition to their concerns related to teaching computational thinking, some of the interview participants argued that there is not a clear and consistent definition of computational thinking. One teacher summarized the need to better define and teach computational teaching:

> Computational thinking has been sort of nebulously defined for a while and it's kind of the way that we humans think about the problems that we then later solved with computers. I'm a fantastic programming teacher and I am not necessarily a fantastic strategies teacher or a fantastic problem solving approach teacher; I almost look at it like maybe I need some math pedagogy to see how math teachers teach students problem solving skills and adhere it to those strategies for math. I would like to learn how to teach these problem solving skills and decouple those from the actual problems or decouple those from the actual programming solutions. (I-1)

Another teacher shared her goal of embedding computational thinking in all the activities of her CS class and stressed the need to learn how to teach computational thinking better: "I think this comes back to the pedagogy question of I need to learn how to teach computational thinking" (I-8).

Overall. All the phases' participants stressed that their students were using trial-error approach and explained trial-error as an ineffective strategy in learning computational thinking. They emphasized problem-solving aspect of computational thinking and expressed the need to

better teach and help students solve computational problems by purposefully designing a solution. Inconsistent definition of computational thinking appeared to be one of the reasons for teachers' difficulty guiding their students to solve computational problems.

Years of Experience and Background in CS. Figure 4.1.2 shows the questionnaire findings for teachers with different years of experience and background.

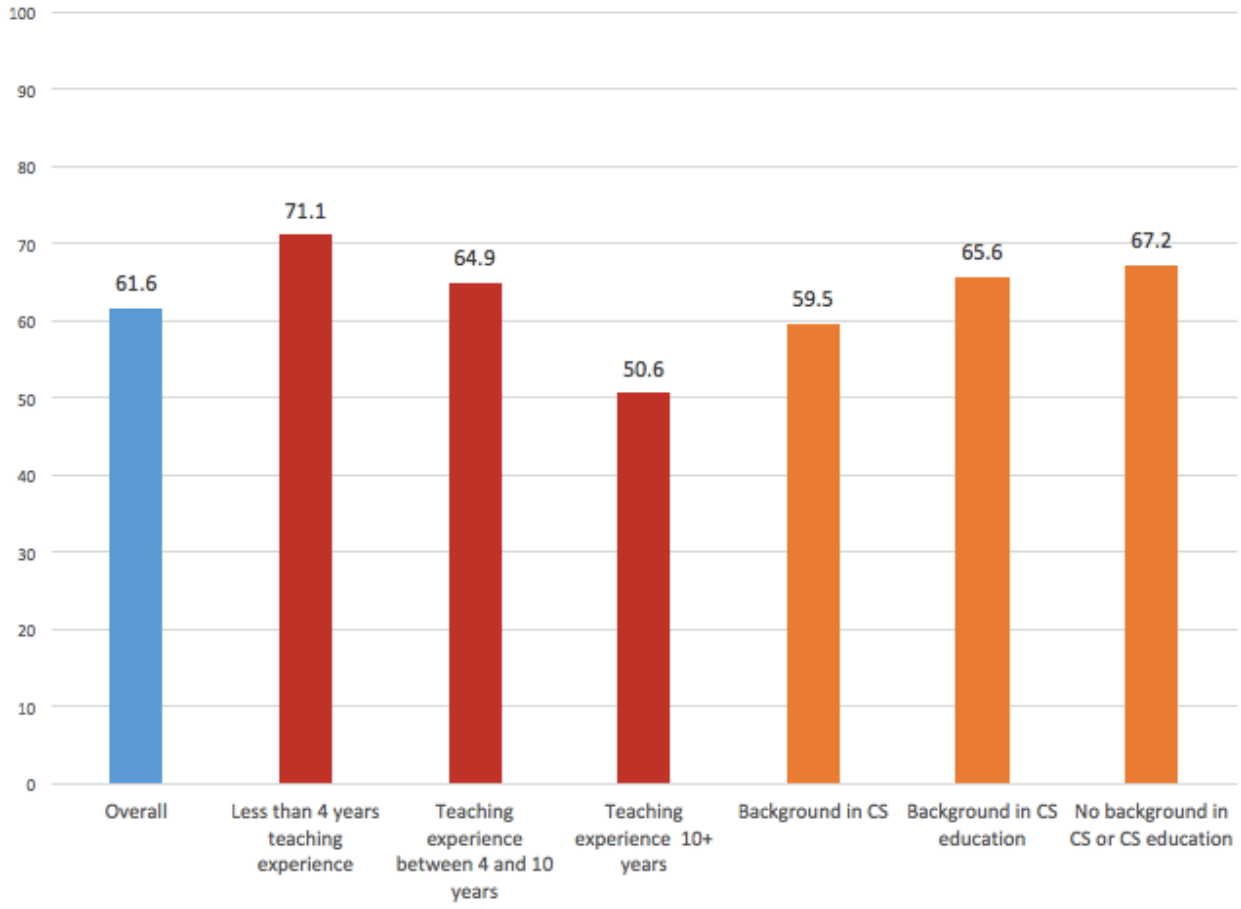

Figure 4.1.2. Frequency of teachers' perceived need to learn how to teach computational thinking.

Teachers with less than 4 years of experience more likely to report learning how to teach computational thinking as a need or strong need compared to more experienced teachers. The findings did not suggest a major difference between different backgrounds for this need.

Transferring Skills between Programming Languages and Platforms

Secondary CS teachers stated that they need more strategies to help students transfer their learning from one programming language to a different one and built on their previous CS content knowledge.

Email listserv findings. Out of 35 teachers that shared pedagogical needs in the email listserv, six teachers expressed the need for helping secondary students transfer their skills from visual programming platforms (e.g., block-coding) to text-based environments (e.g., Phyton), and between text-based environments (e.g., from Phyton to Java). For example, one teacher emphasized her students' difficulty transferring their CS learning from visual programming environment to text-based programming environments: "In my teaching, it seems that majority of students have difficulty migrating concepts they learn from visual environments into text based environments. Starting with Scratch/turtle I found that I essentially had to reteach concepts in Python/Java" (E-11). Another teacher shared the same concern between text-based programming languages:

> I have found the same thing with students going from python to Java or python to C. They will, in a more complex situation, revert to python to begin with, or verbalize a solution in Pyhton, then try it in the other language. I think this may be similar to what happens when a student is learning a new spoken language. Might there be educational or pedagogical parallels there that might work? (E-12)

Another teacher shared her goal to do better job helping students transfer their learning, as followed: "How do we structure assignments and assessments that help solidify learning and help them transfer that learning to new contexts? That transfer is what we're after, really. Transferring skills from one context to another" (E-13).

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need to learn strategies to help students transfer

their learning between programming languages and platforms. Overall, 46.6% of the questionnaire participants (n=217) identified this as a need or strong need. Three teachers shared comments about the need for transferring previous CS knowledge and skills to new environments in the questionnaire. For instance, a CS teacher started teaching an advanced placement CS (AP-CS) course for the first time and complained about her students' lack of transfer from her previous classes to the new AP CS class:

> I have found that the students have been too "spoon-fed" information all of their academic careers that they are not remembering material as readily or able to apply it after time has passed. For instance, we are working on Strings again and it seems they do not remember what we were supposed to learn previously. It makes me think there must be something else I can do to help the process. (Q-8)

Another CS teacher commented about lack of transfer from students' previous learning: "I do still have issues with students transitioning from one IDE or language to another and apparently forgetting everything they learned or not thinking it will be similar" (Q-9).

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to this pedagogical need. Seven out of eight final-phase interviewees expressed either a need or a strong need for learning strategies to help students transfer their learning. When asked to explain this need in more detail and provide examples in the interviews, most listed their students' lack of transfer between the CS courses when there were connections. For instance, one teacher shared that students do not see connections between Scratch and other programming languages when moved from one content to another:

> At the end of each semester I ask them what did they think of it, did they feel using Scratch was valuable? Were they able to transfer what they had learned from one language to another in them? The responses I get, for the most part, are that while they found it interesting, they didn't see the parallels. I know the parallel's there. They just don't make the connections. (I-4)

Another teacher provided examples about moving from Scratch to "high school coding" and HTML classes in her teaching environment, and expressed the need as helping students find the content and skill connections between programming languages:

> We start out with Scratch in here, and then we go in and we're moving ... Right now we're using Code High School. They're doing some stuff on Code High School. We're also doing a collaboration with some local businesses who want to help teach HTML, so they're doing HTML. It's just helping them learn how to say, "Okay, you learned how to do this in Scratch, now how do you apply that in an HTML environment?" Or, "How do you apply that with Java?" Helping them understand the thinking and the strategies are really universal across all these programming languages. (I-3)

Overall. The teachers in all the phases of this research underlined that transferring knowledge between courses and building CS knowledge on former knowledge are important for learning CS. Therefore, all the phases' participants shared the need for strategies that they can use in their classes to show connections between courses, especially in programming courses moving from visual platforms to text-based programming languages. This appeared as a critical component of pursuing CS learning in all the phases of education.

Years of experience and background in CS. Figure 4.1.3 shows the questionnaire findings for teachers with different years of experience and background. Even though "helping students transfer their learning between programming languages and platforms" did not appear to be a need for majority of the overall teacher population, the teachers with less than four years' experience more likely to report it as a need or strong need compared to more experienced teachers. As CS teachers had more experience, this became less of a need, especially for teachers with more than 10 years' experience.

Teachers with no CS or CS education background more likely to report this as a need or strong need compared to teachers with a background in CS or CS education. These *no CS or CS*

education background teachers reported having low CS content knowledge but had knowledge on instructional strategies and tools for teaching in another subject area.

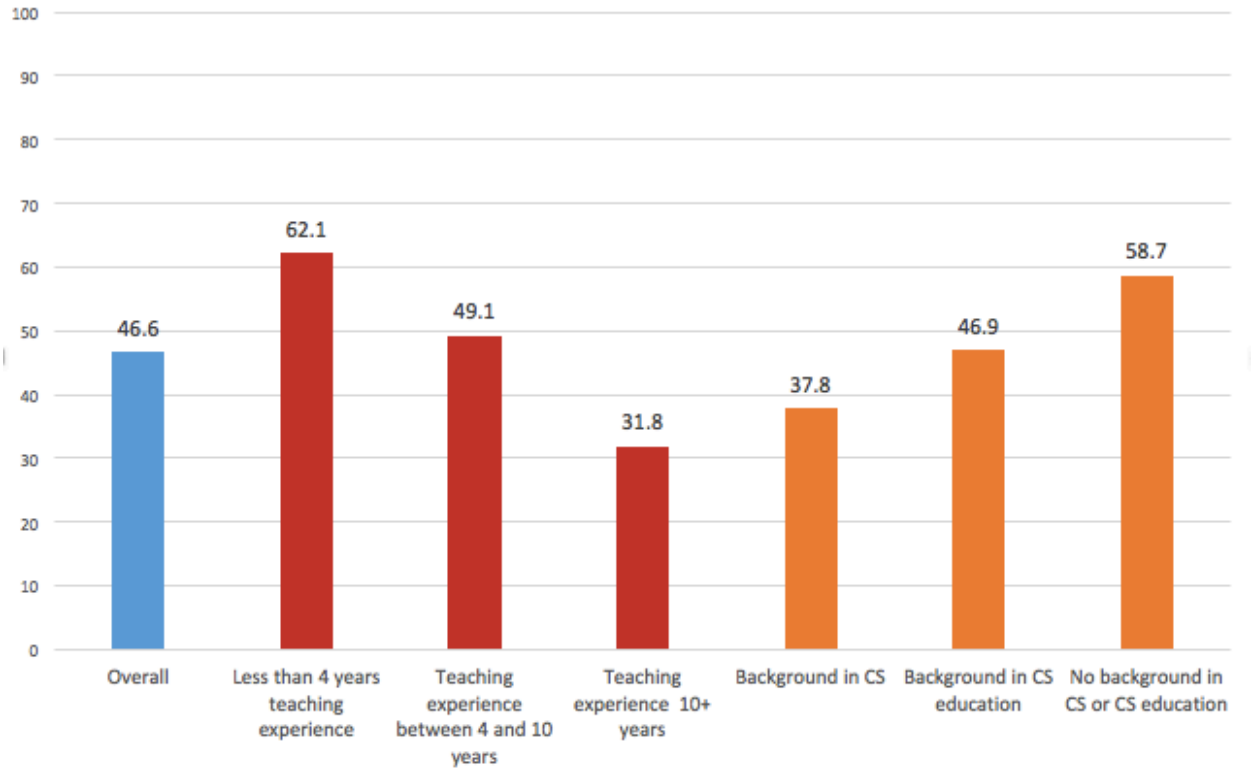

Figure 4.1.3. Frequency of teachers perceived need to learn strategies to help students transfer their learning between programming languages and platforms.

Answering Students' Questions

Secondary CS teachers stated that they need more strategies to help students to correct errors in coding projects and to guide them in correcting their own mistakes by providing appropriate resources.

Email listserv findings. Out of 35 teachers that shared pedagogical needs in the email listserv, five teachers stressed answering students' questions as a need in programming classes and explained it as analyzing code quickly and guiding the students' CS learning process while coding. One teacher stressed the importance of analyzing code quickly:

> When helping students with a project, the teacher needs to quickly analyze what is wrong with the frustrated students' program and then give some advice on how to fix it. The faster the teacher can do this, the more students that teacher can help during class. (E-14)

In a different discussion about developing students as programmers, another teacher mentioned the need to provide one-to-one guidance to students "We don't provide enough one-on-one mentoring to students while programming" (E-3).

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need to learn how to answer my students' simultaneous questions while coding. Overall, 41.3% of the questionnaire participants (n=218) identified this as a need or a strong need. Seven teachers shared comments related to this need. In these comments, the teachers explained this need as assessing code for correctness, giving students appropriate feedback, guiding students for solving code errors, and creating scaffolding that does not need face-to-face guidance. For instance, one teacher explicitly stated fixing students' code as a need for better CS instruction: "I need help with fixing students code, etc. -- sample debugging questions in C++, Java & C# -- and pseudocode" (Q11). Another teacher emphasized her need as strategies to assess students' code for correctness, structure, and efficiency: "I need to streamline the process of assessing my students' code for correctness, use of comments, ease of use, and grammar and spelling in output statements and comments" (Q10). In addition to providing face-to-face help, one teacher stressed the need for strategies that could help students find their own solutions without teacher guidance: "I will appreciate further pedagogical help with teaching computer science and facilitating student knowledge, particularly helping students make better connections with the material, and more effectively debug without needing face time or one-on-one time from me" (Q-2).

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to this pedagogical need. Four out of eight final-phase interviewees expressed either a need or a strong need for learning how to answer their students' simultaneous questions while coding. When asked to explain this need in more detail and provide examples in the interviews, all four teachers shared their own strategies and asked for more strategies that can lead their students to finding an error in their code. Similar to the questionnaire participants, all the interviewees asked for strategies that can lead students to find the errors themselves by exploration. One of these teachers used questioning as scaffolding and asked for more strategies that could guide his students:

> The easy answer that is the wrong one is to point the student to the bug and say, "Well, here's what you did wrong." The harder answer in mind, but the one that I prefer is to sort of ask the student, "What do you mean by this chunk of code" and have them explain to me what they are trying to say and then Socratic method or using questioning to get them to see what they have said, where what they have said doesn't add up with what they intended to say. I would like to learn more strategies other than just questioning and flat out giving them the answer to help them with that. (I-1)

Another teacher explained what she did and stressed that it is not possible to know everything in a programming language, and she expressed the need for strategies that lead students to correct resources or approaches to debug (correcting errors) their own code:

> For me, I've never been a professional programmer, so it doesn't just come automatically for me to look at someone's code and say, "Oh, here's what's wrong," or just learning some of the debugging strategies, or reading code… I'm not used to it enough. You learn how to try and figure out some strategies, like I've gotten to the point where we're like, with debugging, we just break down the program and we also a lot of times, I'll have the students huddle and I'll pull the other students in and I'll say, "Why do you guys think this is not working?" We do it collaboratively, to have them help look at problems and stuff. There's no way I can ... You just can't. There's not enough time to learn all of the languages, because to prepare the classes I have to teach (I-3)

Overall. The teachers in the email listserv solicited ideas for strategies to answer students' questions while coding. This need was explained as teachers analyzing code quickly

and guiding students to correct code. However, in the questionnaire and interview phases of this research, the participants expressed this need as creating environments and scaffolding resources where students can find their own solutions to the problems they experience in coding.

Years of experience and background. Figure 4.1.4 shows the questionnaire findings for teachers with different years of experience and background.

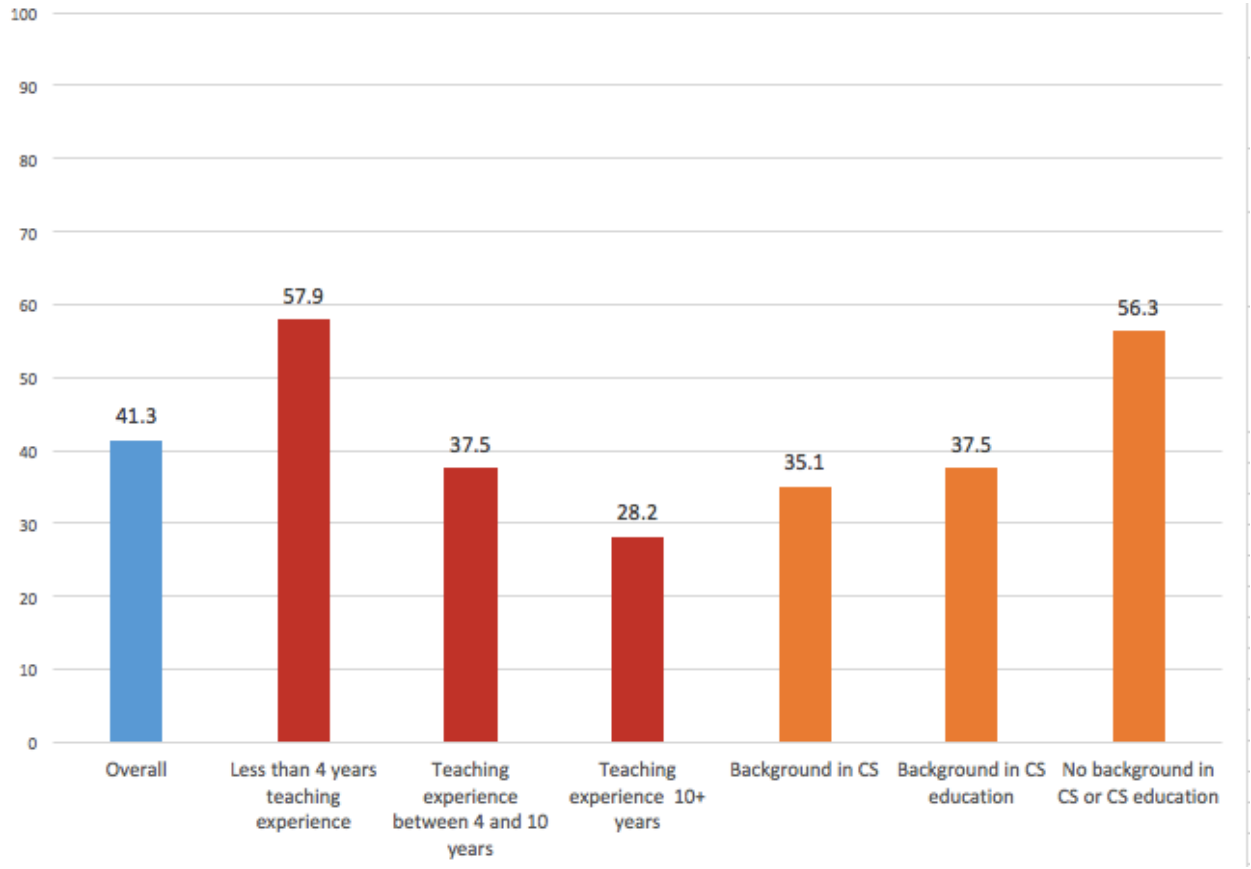

Figure 4.1.4. Frequency of teachers' perceived need to learn how to answer students' simultaneous questions while coding.

Even though "learning how to answer students' simultaneous questions while coding" did not appear to be a need for majority of the overall teacher population, the teachers with less than 4 years' experience more likely to report this as a need or strong need compared to more experienced teachers. As CS teachers had more experience, this became less of a need, especially

for teachers with more than 10 years-experience. In terms of background difference, teachers with no CS or CS education background more likely to report this as a need or strong need compared to teachers with CS or CS education background.

Facilitating Student Interaction and Collaboration

Secondary teachers stated that they need more strategies to facilitate their students' interaction and collaboration in CS classes, which includes creating an environment for students to collaborate and help each other's problems.

Email listserv findings. Facilitating students' interaction and collaboration was not a major and explicit discussion point in the email data. Out of 35 teachers that shared pedagogical needs in the email listserv, three teachers stressed the need for strategies creating a collaborative class environment and guiding discussion. For instance, when teachers discussed problems in their classroom, one teacher emphasized the importance of creating a class environment for learning: "How do kids learn and how do we create an environment to allow that learning to best take place" (E-13). In a different topic, one teacher asked for strategies to guide student discussion after watching a video about how computer changed the world: "I am planning to use segments of this video "The Machine that Changed the World Part" in my Intro to Coding class… Any comments on this series and anyone have some questions to guide conversations" (E-50)?

Questionnaire findings. In the questionnaire, teachers were asked to reflect on the following statement: I need to learn how to facilitate my students' interaction and collaboration between each other in my CS class(es). Overall, 46.1% of the questionnaire participants (n=221) identified this as a need or a strong need. Four teachers shared comments related to this need. In these comments, the teachers explained this need as creating a collaborative environment where

all the students help each other and attend learning activities. One teacher described this need as developing strategies to facilitate student interaction and help each other's problems: "How to facilitate student independence and helping each other and strategies for when they really are stuck and need individual help and I can't be there for everyone" (Q-12). Another teacher commented a collaborative environment as a need: "Creating collaborative environment and excitement around problems and solutions" (Q-13).

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to this pedagogical need. Four out of eight final-phase interviewees expressed either a need or a strong need for learning how to facilitate students' interaction and collaboration between each other in their CS class(es). When asked to explain this need in more detail and provide examples in the interviews, all four teachers described their own teaching practice, emphasizing the importance of student interaction and broaden the score of the need to ensuring all the students' active participation in collaborative work. One interviewee explained that collaboration is essential in a CS a class:

> [Programming projects] forces collaboration in the classroom between me and the students but especially between the students and the students. They quickly learn everybody's got questions and there's only one teacher. If you want your question answered, you've got to go to someone else in the class. The other people in the class don't have to be experts, they just have to be one step ahead of you, because they can help you with the thing you were trying to do. (I-5)

This collaboration requires a flexible environment where students help each other, become creative and learn from each other. One teacher explained the need for a less structured CS classroom for effective collaboration and need for ensuring student learning in-group work:

> It's really trying to map out your lesson and your unit plans to allow that time, balance that time for them to be able to do some of that collaboration together, or working together around problem solving, or evaluating each other, like a game or an animation or something that they created… All of this requires a little bit less structured environment than what the students are used to, so it's a fine line between opening it up, allowing them

> to be able to be creative, and have that time so they can just think and kick the tires, but also making sure that that time is effective, and they're learning. (I-3)

Another interviewee assigned students team coding projects and shared her need to help student collaboration and ensure all the students' learning in groups:

> Where it's one student who's very good at the computer, computational thinking and all this. Then the other student does not know anything. What I'm working with them now is to try and weigh out the groups better, so that everybody is really being able to participate and give, and not just take. I'm trying. Yeah. I need some ideas of how to do that. I mean I feel like maybe next year, because it will be my fourth teaching it. Each year I seem to come up with some new, different ideas. This year has probably been the most creative thus far, trying to do that. (I-6)

Overall. Even though this need did not appear as an important problem in the early phases of this research, the interview participants had chance to provide details about its importance. In all the phases, the participants defined this need as creating an environment in CS classes for students to discuss and help each other's problems. Furthermore, the teachers wanted to make sure all the students were collaborating, participating, and learning from group interaction. The participants emphasized that this interaction requires a flexible environment where teachers created less structure for teaching CS.

Years of experience and background. Figure 4.1.5 shows the questionnaire findings for teachers with different years of experience and background. "Learning how to facilitate students' interaction and collaboration between each other in their CS class(es)" did not appear to be different between teachers with different years of experience and background. The differences were small.

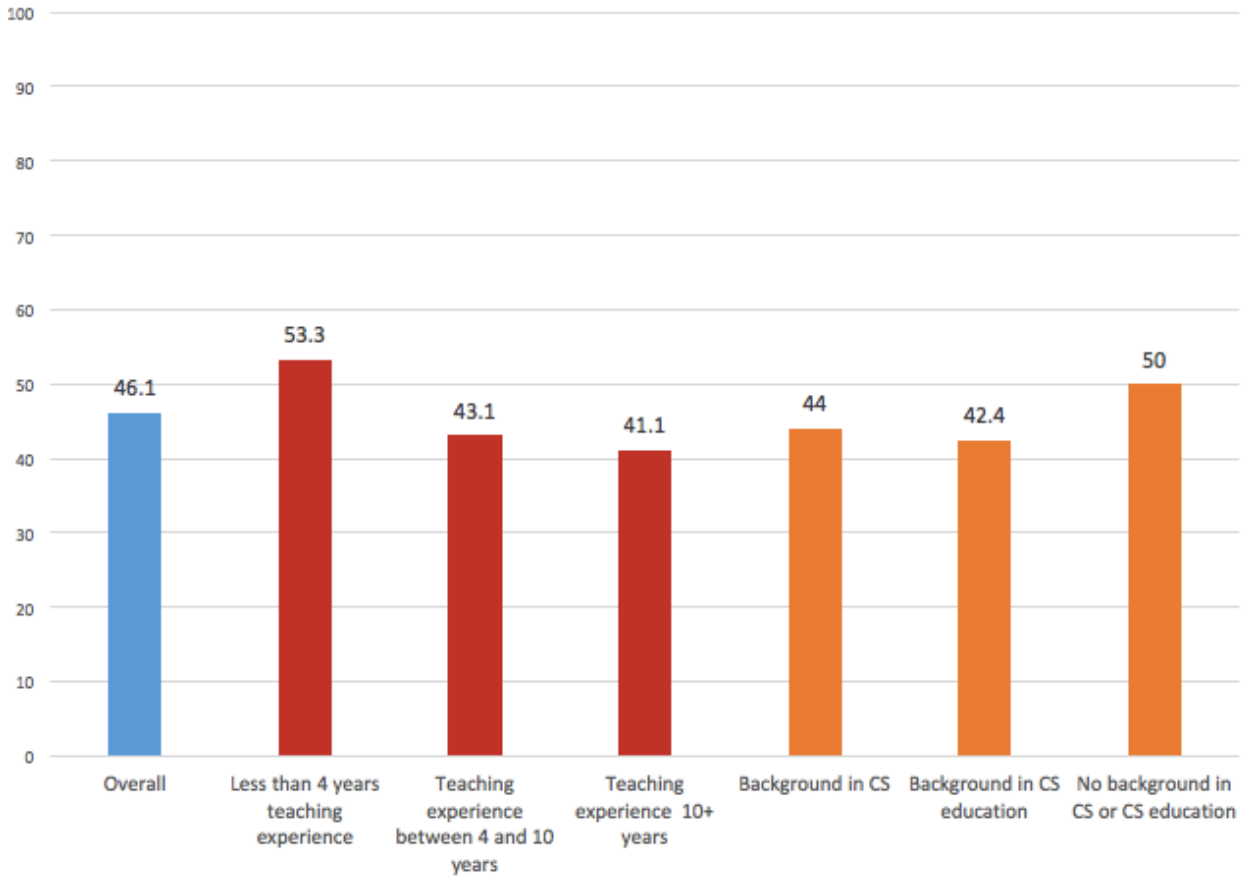

Figure 4.1.5. Frequency of teachers' perceived need to learn how to facilitate students' interaction and collaboration between each other in CS class(es)

Curricular Needs

Curricular needs were identified from CS teachers' communications in both the listserv and questionnaire, and have been enriched with interview data. For the purposes of this study, teachers' perceived curricular needs included the following sub-themes:

1. fully designed curriculum,
2. curriculum resources,
3. selecting a programming language or platform,
4. computational thinking in curriculum,
5. materials to assess student learning.

Fully Designed Curriculum

Secondary CS teachers stated that they need fully designed curriculum for various grade levels and topics in CS education.

Email listserv findings. Out of 132 teachers that shared curricular needs in the email listserv, 21 teachers mentioned the need for a fully designed curriculum for various grade levels and topics. For instance, one teacher in the listserv stressed the necessity of CS curriculum in all the grade levels: "You have a great summary of the problem of no curriculum. And having a K-20 curriculum is necessary" (E-15). Another teacher underlined this need by asking other listserv teachers the CS topics in different grade levels: "what topics to teach/learn at each grade level" (E-16)? Specifically, the listserv participants listed their fully designed curriculum needs as curriculum in programming and app development between grades 6–12. Some examples of these requests include:

- I was wondering if anyone would be willing to share their Java curriculum with me (E-17);
- I am looking for suggestions for Model Curriculum for a High School Intro to Programming Course (E-18);
- I am wondering where is the best place to start with app development for iPad with young children (E-51);
- I just started teaching Computer Science 7-8 and will need different [curriculum] to use next year with the 8th graders so I will look into this. (E-52)

In addition to the need for curriculum for specific grade levels and topics, the teachers in the listserv demanded for a unified curriculum that was derived from concurrent industry practices in the CS field. One teacher expressed the need for a curriculum that was aligned with the business practices in her region:

> We need an effort to compile the work that is going on with the curricula in Chicago, at the ACM, at Microsoft, at Oracle, at Intel, and at just about every other large CS related company and to make a unified curriculum that can be shared by everyone... (E-16)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need a fully designed and ready to implement curriculum for a specific grade level (or specific grade levels). Overall, 34.6% of the questionnaire participants (n=217) identified this as a need or a strong need. 21 teachers shared comments related to this need. In these comments, the teachers explained this need as accessing to curriculum for different grade levels, curriculum that can help students' needs after graduation, and curriculum for the new Advanced Placement CS Principles exam. One teacher briefly stressed the need for a fully designed curriculum: "A full curriculum would be a great resource" (Q-17). Another teacher followed a 10-years old book as her main curriculum, and described the need as an updated CS curriculum:

> My district does not see the need to provide current curriculum for CS. The books that I currently use are at least 10 years old. My instructions to the students are to take the book home and use it for reference. It takes time to plan curriculum and prepare assignments when you don't have curriculum. (Q-14)

Another teacher specified his challenge as accessing to curriculum for different grade levels and content:

> Curriculum is always a challenge. For example, I found a solid curriculum online for the high school Visual C# and XNA Game Studio Game Programming class that I was able to use for a small cost. But, when teaching Game Maker to 8th graders, I have had to learn the material and largely create the curriculum on my own. (Q-15)

The questionnaire participants mentioned the importance of preparing students for higher education and business life through K–12 education. These participants searched for curriculums that were designed by considering students' needs after graduation from K-12. When asked to comment about curriculum needs, one questionnaire participant highlighted her need as a curriculum that higher education or business practices were embedded:

> I need to know what I should be teaching the students to prepare them for post secondary school, Tech school or University. What are they expecting the students to know when

they arrive? Or what are businesses expecting of the students from high school. (Q-16)

Interview findings. Before conducting the interviews, the researcher considered participants' questionnaire responses related to this curricular need. Four out of eight final-phase interviewees expressed either a need or a strong need for a fully designed and ready to implement CS curriculum for a specific grade level (or for specific grade levels). When asked to explain this need in more detail and provide examples in the interviews, these four teachers shared that due to constantly changing nature of the CS content, they need curriculums that are updated continually.

The interview participants were mainly concerned that CS changes so quickly and their curriculums were outdated for the students' needs after school. For instance, one teacher emphasized her major need as providing curriculum that is updated and addressing students' needs: "I think the main thing is just like I said, it keeps changing so fast and trying to make sure the curriculum you're presenting is something that students will be able to use moving forward is always a major concern" (I-7). One interview participant mentioned using books as their main curriculum source and reported the content as outdated: "the book I use is currently outdated. It's from six or seven years ago… a lot of stuff has been changed" (I-8).

Overall. All the phases' participants accepted the importance of updating content and reported the need to a designed CS curriculum in different grade levels especially in programming and app development. Higher education institutions' standards and industry practices were the two main factors shaping the teachers' decisions and interest in new curriculums.

Years of experience and background. Figure 4.2.1 shows the questionnaire findings for teachers with different years of experience and background. "A fully designed and ready to

implement CS curriculum for a specific grade level (or for specific grade levels)" did not appear as a major need for overall, different years of experience and background teacher groups. However, teachers with less than four years' experience more likely reported this as a need or strong need compared to more experienced teachers. The differences were small between teachers with different backgrounds for this need.

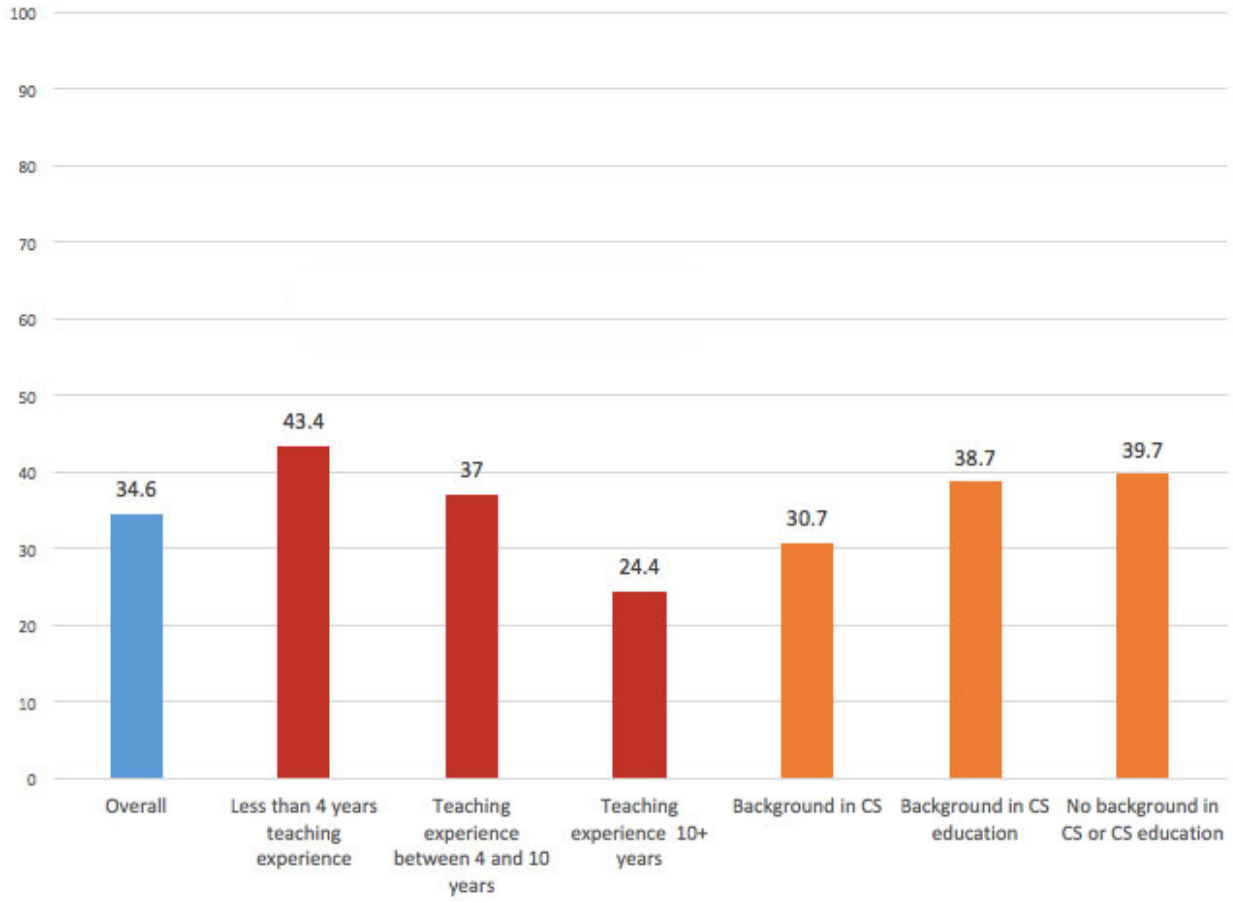

Figure 4.2.1. Frequency of teachers' perceived need for a fully designed and ready to implement CS curriculum for a specific grade level (or for specific grade levels).

Curriculum Resources

Secondary CS teachers stated that they need more curriculum resources for teaching CS, which includes project ideas, examples, tutorials for, programming, mobile app development, web design, AP classes and robotics classes.

Email listserv findings. Out of 132 teachers that shared curricular needs in the email listserv, 45 teachers asked supportive materials for their existing curriculums. These materials included sample projects, descriptive examples, tutorials, activities, and exercises in teaching CS on variety of topics.

With focus on programming and potential benefits in K–12 education, teachers were looking for curriculum materials that could support their teaching in programming in different languages. For instance, one teacher was asking materials for Phyton and Swift: "Would you be willing to share the material for what you have done with Python and Swift?" (E-19). A first year CS teacher shared the need for final projects in her Python and C++ classes for high school students:

> This is my first year teaching and I have HS juniors and seniors learning Python and C++. I am trying to come up with an end-of-year project that will allow the students to work together as a software development team. (E-23)

Another teacher was looking for project ideas in programming appropriate for her students' interest and level:

> I'm looking for ideas for projects for my Computer Programming class not too terribly advanced using Visual Basic/Visual Studio 2010. Our current curriculum and textbook are outdated (2002) which has made it challenging to balance the curriculum with students having no programming experience to students with programming experience. (E-57)

With widespread use of mobile applications in daily life, children's interest in learning app development increased, and the teachers in the listserv were asking for materials to teach mobile programming. One teachers were asking materials from the other teachers to develop applications for IOS environment: Would you kindly consider sharing your Xcode, Python and Swift material with me" (E-1). Another teacher was looking for materials in Android programming platform:

> 1) If anyone wants to post specific tutorials or activities that they do in AppInventor, I would love to look into this (E-20);
> 2) While we are on that subject, can I ask what resources, tutorials, etc. folks are using these days to teach HTML5 and CSS. (E-21)

Another teacher was looking for a tool that would allow the students to develop mobile applications both for IOS and Android:

> I need a multi-platform tool that allows software to be written for iOS and Android devices... ideally smart phones. Also on my wish list would be an emulator so that if a child does not/cannot get access to the intended platform, they can still participate. (E-41)

Even though it was not a major discussion point, materials in AP CS and AP CS Principles were another need for some listserv teachers. One teacher wanted to use their available funds to buy curriculum materials for his AP classes and was asking ideas from the other CS teachers: I was just informed that there are some funds available to order curriculum materials. Any suggestions for AP Computer Science and/or CS Principles" (E-18)?

Curriculum materials for teaching web design was identified as another important need in the listserv discussions. One teacher expressed her need for materials in HTML and CSS: "While we are on that subject, can I ask what resources, tutorials, etc. folks are using these days to teach HTML5 and CSS" (E-55)? Another teacher was asking for a final project idea in a web design class: "Is anyone willing to share a web design final project" (E-56)?

Even though they could access vast amount of online resources, the teachers in the listserv were looking for text books in different CS content as convenient sources in hand when they need to access to curriculum materials. One teacher needed a C# textbook for exercises for high school students and teacher resources: "I'm looking for a C# textbook that has fun exercises for high school students. I'm hoping to find one that has teacher resources (code samples, presentations, etc) with it" (E-22). Another teacher was looking for an online textbook in Java for a programming class: "For my Intro to Computer Science class, which is taught in Java, I

would like to use an online textbook. Does anyone have any suggestions" (E-24)? In addition to curricular needs in programming, one teacher was asking for a book about intellectual property and software rights, which is an important learning standard in teaching CS:

> I am searching for a good reference book that deals with issues related to Intellectual Property and Software Patents that could be made part of syllabus in CS. It would be good if the reference discusses case studies from different parts of the world." (E-26)

With encouraging findings about the benefits of using Robotics kits for teaching CS and other sciences concepts (Benitti, 2012), the teachers in the listserv were also looking for ideas and recommendations for robotics kits or teaching materials. One teacher was looking for an affordable robotics kit for his classroom: "I am creating a Robotics 2 class for next year using submersibles from Sea Perch and looking for affordable ($100.00 - $200.00 or less) quad-copter kits. Any suggestions" (E-53)? Another teacher shared the difficulty of finding K–12 level robotic kits for her classroom:

> I teach middle school robotics which is really quite baby stuff, mostly drag and drop, and it's hard to get the students are interested in principles of programming. Apart from the LEGO Mindstorms, it is hard to find appropriate robots." (E-54)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need curriculum resources (e.g., practice questions, assignments, activities, project samples) for my CS class(es). Overall, 71.2% of the questionnaire participants (n=219) identified this as a need or a strong need. 34 teachers shared comments related to this need. In these comments, the teachers explained this need as accessing curriculum materials (e.g., online materials) in programming and AP CS. One teacher provided an example of this need as accessing high school level curriculum resources in different programming languages: "I would like to have a textbook or set of resources that are applicable to the high school level for languages other than Java (the AP language). Like Python, C/C++"

(Q-18). Another teacher complained the similarity of the programming problems in text books and had difficulty finding appropriate level online programming activities for her students:

> I teach VB.Net, C++ and C#. The books all seem to have the same problems but in a different language. My students get board of doing the same problems. I do go online and find contest problems but sometimes they are harder for some of the students. (Q-19)

Another teacher defined his greatest need as lack of access to online resources: "My greatest need is for online resources: for demonstration/instruction, for creative exploration, for focused practice, and also for evaluation" (Q-22). In a Python course, one teacher mentioned her needs as accessing successfully applied curriculum materials in other's CS courses:

> My primary need is always PROVEN activities/projects/practice questions. I spend the vast majority of my time creating them from scratch, then modifying them each time I teach the course. I feel that my students get a much better experience every semester, but I would like to start them higher on that curve with proven resources. (Q-20)

Even though curriculum materials for AP CS did not seem as a major need in the email discussions, with plans about the new AP CS course and the exam in 2016, the questionnaire comments showed more evidence of AP CS materials as an important need. One teacher shared her plans for applying the new AP CS Principles course and shared her need for more resources: "I am instituting the AP CS P course next year and hope that College Board has more resources available" (Q-21). Another teacher shared lack of time as an issue for creating curriculum materials and looking for materials for the new AP CS Principles course:

> As the sole teacher of computer science in my school with multiple different course each semester I don't have time to create my own material for each class. I look for well-designed material I can modify as I incorporate into my classes… Looking at implementing AP courses in Computer Science Principles and Computer Science A within the next couple of years. (Q-23)

Interview findings. Before conducting the interviews, the researcher considered participants' questionnaire responses related to this curricular need. Five out of eight final phase interviewees expressed either a need or a strong need for curriculum resources (e.g., practice

questions, assignments, activities, project samples) for my CS class(es). When asked to explain this need in more detail and provide examples in the interviews, these five teachers shared that finding updated resources is a constant need and with new frameworks and standards, they expect to reach those resources in an organized way. One of the interview participants need to access a repository of curriculum resources that were organized with CS topics:

> If I'm trying to teach something that I'm really uncomfortable with, like maybe I'm trying to teach a standard on data representation or maybe I'm trying to teach a standard on iteration. Whatever it is, it would be nice if there was a repository where I could go find questions and find assignments and find activities and they all be correlated or filterable by a standard or a learning objective. (I-1)

Another teacher defined this need as a constant need with the changing content of CS regularly: "it's just a constant having to go online and look up solutions" (I-7).

Overall. Need for curriculum resources was one of the most frequent needs for teachers and was reported as a constant need in all the phases of this research. As teachers update their content with different standards and content, they sought materials from other teachers. Curriculum materials for programming classes, mobile app development, web design, AP classes, and robotics education were the most-requested content in the findings.

Years of experience and background. Figure 4.2.2 shows the frequency of teachers' responses to the need for curriculum materials (e.g., practice questions, activities, project

samples) for their CS class(es).

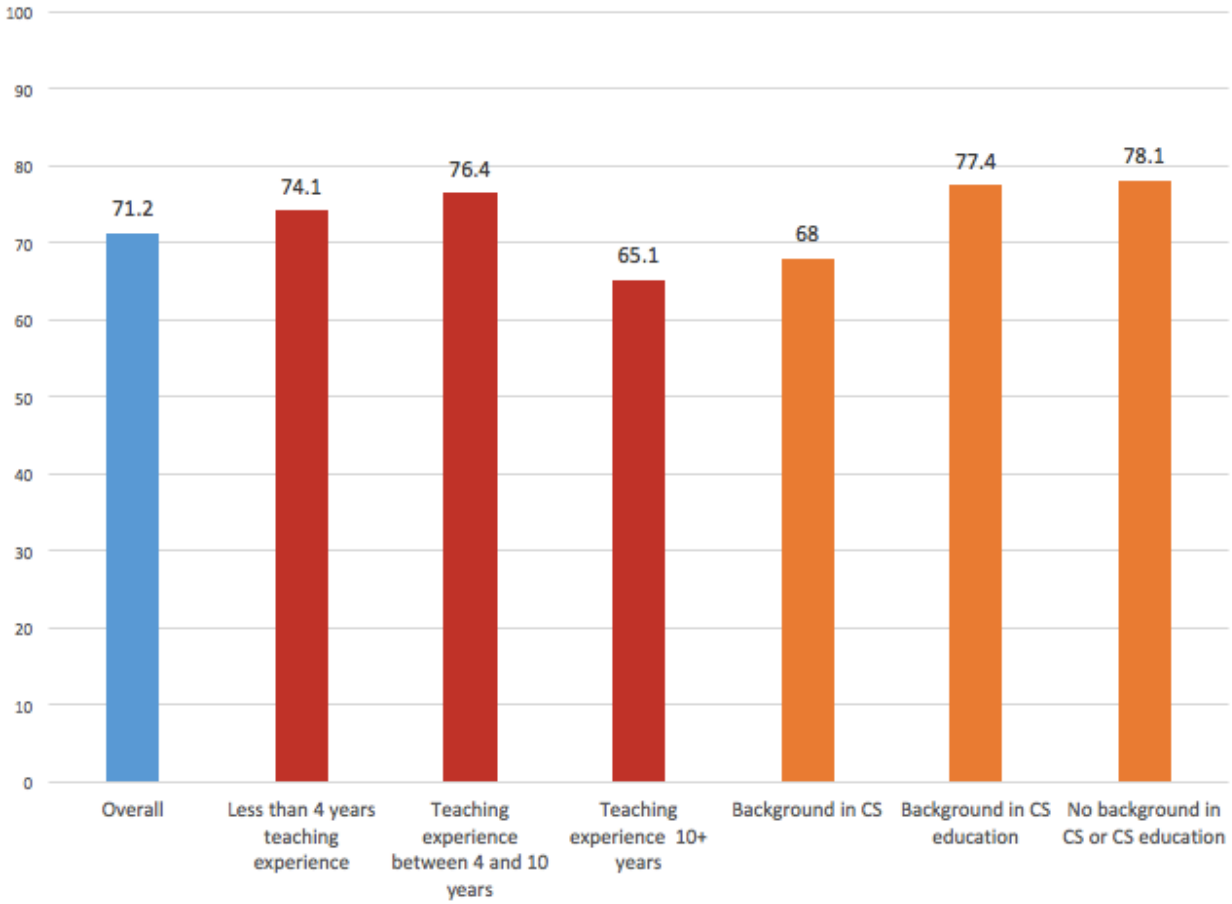

Figure 4.2.2. Frequency of teachers' perceived need for curriculum resources (e.g., practice questions, assignments, activities, project samples) for CS class(es).

Need for curriculum materials was one of highest needs for the population. Majority of the CS teachers in all the years of experiences and backgrounds reported this as a need or a strong need. The qualitative data support this finding with quotes from the email discussions and comments in the questionnaire. One first year teacher expressed her need in the email discussions for curriculum materials:

> This is my first year teaching and I have HS juniors and seniors learning Python and C++. I am trying to come up with an end-of-year project that will allow the students to work together as a software development team. (E-23)

Similar to years of experience, most of the teachers in different backgrounds reported curriculum resources as a need or strong need (see Figure 4.2.2). In the questionnaire comments, one teacher expressed lack of background in CS and shared the need for curriculum resources: "Not having come from a CS background, I find that the resource that is most lacking in my experience. Getting the time necessary to gain that experience is a real challenge" (Q-24).

Selecting a Programming Tool or Language

Secondary CS teachers stated that they need more ideas to select an appropriate programming tool or language for students with different levels and interests.

Email listserv findings. The discussion about which programming language or platform best fits for different levels of students was one of the important themes emerged from the email conversations. The teachers were reconsidering their long time practices and previous choices of programming languages after the development of new tools and approaches in programming. Out of 132 teachers that shared curricular needs in the email listserv, nineteen teachers were explicitly asking for recommendations and feedback in choosing the best programming language/tool option for different levels of students. However, many teachers involved into these discussions after these teachers shared the need. One teacher shared using Phyton and reconsidering Processing as a new language in introducing programming to her 10th grade students:

> I have been using Python (starting with RUR-PLE) in my intro class for quite a few years now to introduce programming to 10th graders. Although it's been reasonably successful, I'm thinking about switching to Processing. It strikes me as having more appeal to non-CS types, mostly because of the integrated 2D and 3D graphics, the natural framework for animation, and what seems to be a trouble-free cross-platform development environment. (E-27)

Multiple teachers asked ideas from the listserv participants about considering a new programming language (C++, Visual Basic etc.) in their classes with different level students:

- We are also starting C++ programming with our seniors. And I wonder if Visual C++ would be a better option (E-1);
- I wondered what programming languages people would recommend for the intro class as good preparation for the AP class (E-28);
- I have not seen VB mentioned in the string anywhere and just want to know your thoughts on preparing students for the programming world past high school. (E-29)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need help to make a decision for selecting an appropriate level of programming tool or language for my CS class(es). Only 27.1 % of the overall questionnaire participants (n=214) identified this as a need or a strong need. Nine teachers shared comments related to this need. Similar to listserv participants, the teacher comments explained their need as choosing a language appropriate for their students' level. One teacher expressed the need for choosing a mobile application programming tool appropriate for advanced students:

> There are online courses in developing android apps using Java, or developing IOS apps. I need to learn these other tools, which might appeal to more advanced/experienced students who find the block-based programming in App Inventor too cumbersome and inflexible. (Q-25)

Another teacher shared that he had conversations with other teachers about what content is appropriate for different level students and their reasoning level:

> I am frequently asked about what is good for Middle School and what is good for high school as well as what the future will be if more content is taught in the Middle Divisions. It would be helpful to have insight into what types of problems correlate to Middle School reasoning levels mathematically and how these skills can be developed as the children grow in the maturity through high school. (Q-26)

Interview findings. Before conducting the interviews, the researcher considered participants' questionnaire responses related to this curricular need. Two out of eight final-phase interviewees expressed either a need or a strong need to make a decision for selecting an appropriate level of programming tool or language for my CS class(es). When these teachers

were asked to explain this need in more detail and provide examples in the interviews, both shared that they wanted to learn what higher education and industry were expecting from the students in their district and state. One teacher conferred with a regional university about the best programming language to teach in high schools: “I actually have gone in and I have talked to some neighboring schools to see what they want. I know for [university name], they really wanted the students to have C++” (I-6). Another participant considered her own skills and industry practices as important factors for selecting a programming language:

> For me, it's like, "Okay, do I teach Scratch?," and it is really good to get them up and going, and understanding programming, but then should we move to HTML? Or, do you move to JavaScript? Or, do you move to C++? Or, do you move ... What language is realistic for an introductory course? I think it's just trying to understand what makes the most sense, and what's the skillset that you can teach them that would be recognized by industry? Does that make sense? (I-3)

Overall. Even though the questionnaire frequency was low, the participants in all the phases highlighted that they need help to make an appropriate language decision based on the students’ level and interest, and higher education and industry expectations. It was stressed important to meet these expectations and teachers know how to teach it.

Years of experience and background. Figure 4.2.3 below summarizes frequency of teachers’ responses to the need for making a decision for selecting an appropriate level of programming tool or language for their CS class(es). Overall, even though it did not seem a major need, CS teachers with *less than 4 years’ experience* more likely to report this as a need or strong need compared to more experienced teachers. Differences were not as big as the years of experience but seemed less of a need for teachers with CS background.

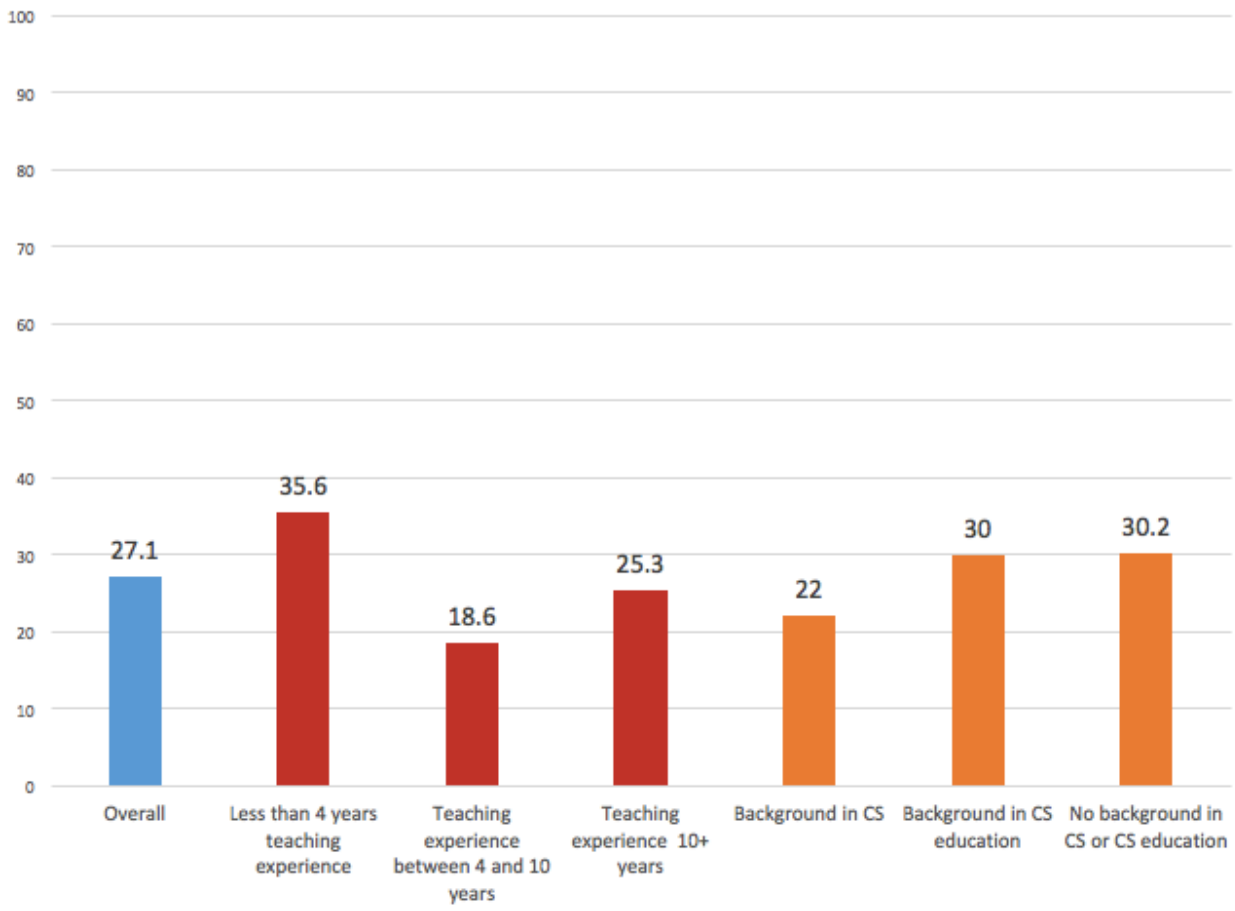

Figure 4.2.3. Frequency of teachers' perceived need to make a decision for selecting an appropriate level of programming tool or language for CS class(es).

Computational Thinking in Curriculum

Secondary CS teachers stated that they need more ideas and resources to embed principles of computational thinking into their curriculum, which involves learning the concept and principles of computational thinking and accessing curriculum resources that helps students design solutions to computational problems.

Email listserv findings. Out of 132 teachers that shared curricular needs in the email listserv, nine teachers highlighted the importance of embedding the principles of computational thinking into their CS curriculum. One teacher stated that the CS education efforts in early grade levels should focus on computational thinking but did not include it:

> I believe if we carefully align and include basic fundamental skills when we work with students in STEM and programming, we can reinforce these basic skills as well as providing students an opportunity to learn CS beginning at the elementary level. Students even at the 3rd and 4th grade level need computational skills and there is a weakness in this area in Math. (E-30)

The listserv participants stressed the importance of computational thinking in their curriculum regardless of the programming language choice. One teacher highlighted that programming language is just the tool but embedding solving problems should be the goal of CS education: "[Students] are not studying programming, they are using programming to solve a problem or accomplish something real… I would be interested in any successes and ideas that other teachers have" (E-31). Therefore, problem solving and designing solutions through computational thinking appeared as an important need in teachers' discussions, as followed:

- The question should not be what language, but what package includes a language that totally supports designing solutions (E- 7);
- I find it interesting that a lot of the discussions on this list revolve around preferences of language and platform instead of bigger questions around what we teach and why. The language/platform is one choice you make, but the bigger decisions are around what kind of thinking and problem-solving you're developing in the students. (E- 10)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need help embedding the principles of computational thinking into my CS curriculum. Overall, 51.4 % of the questionnaire participants (n=218) identified this as a need or a strong need. No teachers shared comments in the questionnaire about embedding computational thinking in curriculum.

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses to this curricular need. Four out of eight final-phase interviewees expressed either a need or a strong need for "embedding the principles of

computational thinking into [their] CS curriculum." When asked to explain this need in more detail and provide examples in the interviews, they identified understanding computational thinking and its components in curriculum as their need. One teacher expressed his lack of knowledge on the principles computational thinking: "When I read the sentence I don't immediately think of what the principles of computational thinking are. I can't list them off. I would like to at least have help learning with that" (I-1). Another teacher shared the same concern about computational thinking definition and components: "As you said, several different thought leaders have different definitions of what that is and what the key components are, so as that grows and changes, I need to be aware of it" (I-5).

Overall. Computational thinking seemed to be an important concept and skill that most teachers want to embed in their curriculum. However, based on the participants' comments during interviews, inconsistent definitions and unclear components of computational thinking were two main reasons for this need. Most teachers do not primarily focus on learning a programming a language, such as Python, Java, but emphasize computational thinking as the main purpose of learning in CS courses.

Years of experience and background. Figure 4.2.4 shows frequency of teachers' responses to the need for embedding the principles of computational thinking into their CS curriculum. Little over half of the population reported this as a need. However, CS teachers with *less than 4 years' experience* more likely to report this as a need or strong need compared to more experienced teachers. In the phase 4 interviews, a 3rd year teacher (I-6) shared her trials using cups to learn algorithms and asked for more ideas that she could embed into his curriculum.

We've done algorithms and they were fine. They fussed a bit with me but we did do algorithm sorts the other day. Which is real cute. They did them with cups and moved them around and that type of thing. They learned a lot about heap sort and shell sort and all of those. I sometimes feel like I just need one more or two more ideas. (I-6)

In terms of background, the differences were small.

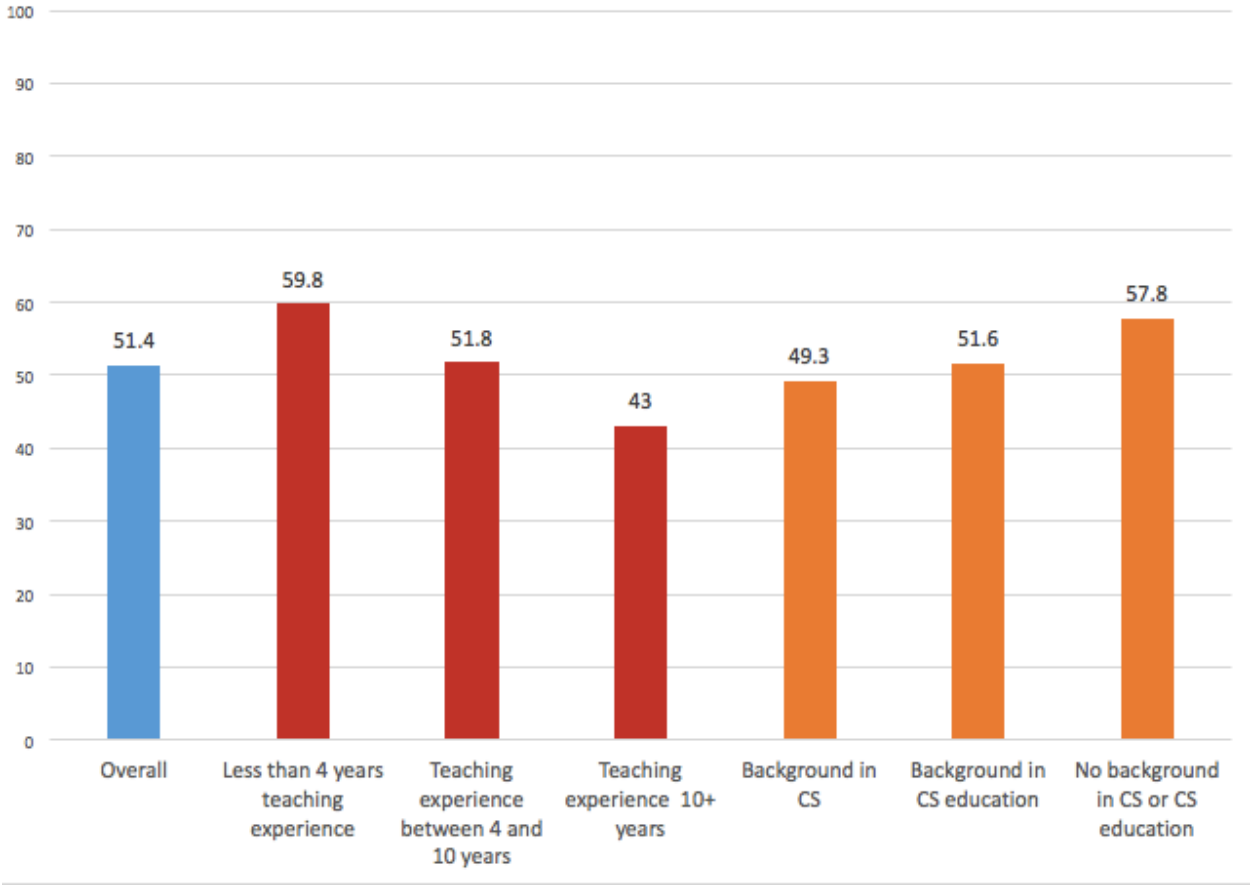

Figure 4.2.4. Frequency of teachers' perceived need for embedding the principles of computational thinking into their CS curriculum

Materials to Assess Student Learning

Secondary CS teachers stated that they need more materials to assess their students' learning in CS, which involves finding or developing student centered assessment materials and grading students work fairly and in a timely manner.

Email listserv findings. Out of 132 teachers that shared curricular needs in the email listserv 25 teachers asked other teachers' ideas and materials for assessing students' learning in different CS content and how to deal with plagiarism issue in programming classes. In a listserv discussion, one teacher asked ideas about how to assess his students' summaries after reading articles about CS: "I'm interested in having students read current computing news articles. While I have no shortage of articles for them to read, I'm looking for suggestions on how to assess that" (E-59). Another teacher asked how to create assessments that support students' learning in CS: "How do we structure assignments and assessments that help solidify learning and help them transfer that learning to new contexts?" (E-13).

Another discussion regarding assessment was about plagiarism in a programming class. Since programming involves team work and there is a possibility of finding previously written code online from other people and libraries, the listserv members asked other teachers for ideas on how they approach the issue of plagiarism in a programming class. One teacher shared his approach and discussed how CS plagiarism differs from other subject areas: "The problem is, the rules for copying code are not exactly the same as plagiarism with books/text" (E-60). The same teacher explicitly stressed the difficulty of assessing student learning in CS and suggested continuing this discussion with different topics: "It's probably worth outlining explicitly and having a discussion about this very topic, with specific examples of what constitutes infringement or not" (E-60). Based on this discussion, one teacher asked others about the amount of acceptable original work: "I wonder what everyone thinks about what the minimum original work percentage of a project should be" (E-58).

Questionnaire Findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement in the questionnaire: I need materials to assess my students' learning in

my CS classes. Overall, 64.2 % of the questionnaire participants (n=218) identified this as a need or a strong need. Ten teachers shared comments related to this need. In these comments, the teachers highlighted the need for materials for assessment and their concerns regarding assessing group work and plagiarism. One teacher shared a desire for ideas for assessment: "I especially need ideas for assessment." (Q-27) Another teacher shared the same need: "I depend too much on traditional assessment. I would like ideas for other ways to asses than regular projects and written tests" (Q-28). One teacher brought up the topic of assessment as a problem and her effort to get better in creating better assessments and projects in her CS teaching: "Assessment is a problem, though I am becoming better at creating tests and projects." (Q-29)

Assessing group participation and plagiarism in programming were two other concerns regarding assessment in CS. One teacher shared that programming involves team work and his difficulty grading the member's participation: "We do work in groups on several of the projects but you always have those students that let their partner do all the work. How do you grade this? I do give a peer evaluation which does help" (Q-19).

In regards to plagiarism, one teacher commented plagiarism as an issue in programming classes and highlighted that findings examples and solutions from other resources does not help students' learning CS: "Biggest concerns are rampant cheating/copying. Students not thinking and just doing rote copy of examples and not able to apply them" (Q-7).

Interview findings. Before conducting the interviews in phase 4, the researcher considered the participants' questionnaire responses to this need. Four out of eight final-phase interviewees expressed either a need or a strong need for materials to assess my students' learning in my CS classes. When these teachers were asked in interview to explain this need in more detail and provide examples, they provided detailed explanations and stressed the need for

open-ended project based assessment in CS classes. One of the interview participants highlighted assessment as his biggest need, stating that traditional ways of grading does not work in CS classes and solicited ideas for student-centered open-ended assessments:

> If there was one area where I need the most help with, that's it. Grading in a traditional sense, like you would grade a math class, doesn't make sense on a project based class. In open-ended projects, where every students project is unique and different than everyone else's. I create rubrics that I think are good, but I think they're probably not very good. If there was the one thing I could use more than anything else for our program it'd be a way to evaluate accurately open-ended projects that fit how we do class. Very collaborative, very out-of-your-seat, very student driven, but that leave room for open- ended, but also that let us evaluate parts of a project which are subjective by nature. (I-5)

The same teacher shared the need for materials to assess higher order thinking such as creativity:

> If I'm going to have creativity as one of the requirements of a project, how do I evaluate creativity? Are there strategies and people who have found ways to evaluate those things? If the kids are programming a game, is the game challenging and addicting and fun? Well, I don't know. So, how do you evaluate that? (I-5)

Another interview participant emphasized that her class is project-based and she struggles to assess students' work because it takes more time and effort than the traditional way of assessment:

> Okay, well the assessments can be formative and summative. I am more project based learning. I prefer to have things where they actually do it. I like hands on… When it comes to assessing their learning, I don't know. I feel like sometimes I struggle. That I'd rather not give them periodical assessments and just count the one big one. Sometimes, I tell you, it is tough for me to get everything graded. (I-6)

Overall. All the phases' participants emphasized assessment as a major need regarding findings materials and grading students work fairly and in a timely manner in project-based CS classes. Furthermore, clarification about plagiarism and the amount of acceptable work in a programming class was another need uncovered in CS teachers' discussions, comments, and statements. The interview participants solicited ideas for assessing higher-order thinking skills.

Years of experience and background. Figure 4.2.5 displays the frequency of teachers' responses to the need for materials to assess students' learning in CS classes. This is one of the highest frequency needs overall and this can be seen in different years of experience and background. However, teachers with less than 4 years' experience more likely to report this compared to more experienced teachers, especially teachers with more than 10 years' experience. The majority of all the background teachers reported "materials to assess students' learning" as a need or strong need.

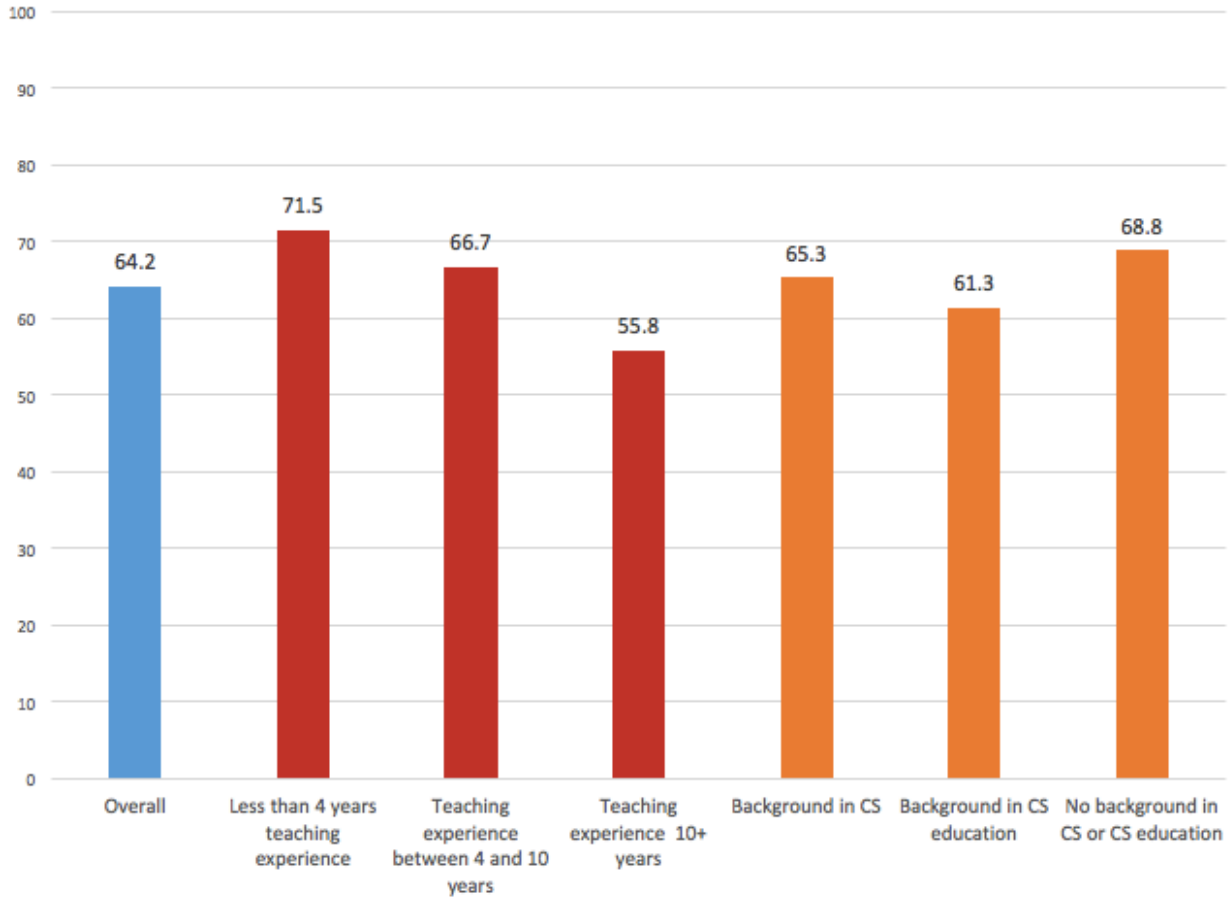

Figure 4.2.5. Frequency of teachers' perceived need for materials to assess students' learning in CS classes.

Student-Related Needs

Student-related needs were identified from CS teachers' communications in both the listserv and questionnaire, and have been enriched with interview data. For the purposes of this study, teachers' perceived student-related needs included the following sub-themes:

1. increasing student enrollment in CS classes,
2. motivating underrepresented populations (females and minorities) to enroll in CS classes,
3. teaching students with low interest in CS,
4. teaching students with limited fundamental skills (math and reading).

Increasing Student Enrollment in CS classes

Secondary CS teachers stated that they need more strategies to increase student enrollment (number of students) in CS classes, which includes informing students about what CS entails and strategies to develop positive student beliefs regarding CS.

Email listserv findings. Out of 67 teachers that shared student-related needs in the email listserv, nine teachers solicited strategies for increasing student enrollment in their classes. There were several factors influencing the low enrollment issue in CS classes in secondary schools. For instance, one teacher was concerned about the name of their CS course as an influential factor for students' low enrollment:

> I was wondering how successful recruiting has been for Computer Science Principles in schools. We converted our Visual Basic course to an Honors Computer Science Principles for next year. I am not sure if the description scared students away or they wanted to program but the numbers dropped substantially...almost 70% less. We also had 50% less for our gaming C# class. (E-62)

Other factors that may have influenced low enrollment were CS being an elective course and the way the course is perceived by students. Listserv teachers reported that CS classes are electives and it is difficult to increase enrollment for those classes with CS's reputation among

students as being difficult and boring. With these challenges, one teacher discussed increasing student enrollment as a need in his school for high school students:

> And at the high school level, at least right now, we are confronted with the issue that CS is an elective study for the majority of students. Trying to convince a 14 year old who is looking for a "fun" class to take something with the rigor of college level CS is extremely difficult. If it is not fun, entertaining, and slightly challenging we loose the students to courses like "Writing from the Cinematic Experience" or "History of Rock and Roll". Sure there are the rare few who enjoy the challenge of writing a program and tackling a problem and these are the rare who trickle through into the college programs, but for the vast majority computer science is looked at as "work" and not "fun." (E-33)

Another teacher stated that CS is an elective class in his high school. She was looking for strategies to increase student enrollment for elective programming classes:

> I'll be joining a high school this fall which has no Computer Science requirement in the upper grades. Consequently, not as many kids sign up for our programming offerings as the technology department would like. I've been given the happy task of designing several mini-courses with the aim of reaching out to kids and increasing (hopefully dramatically) the interest level in CS. (E-61)

In addition to the issue of providing less access to the benefits of CS, one teacher stressed another problem that the low student enrollment in CS classes may cause: the loss of teaching jobs: “Any thoughts would be very welcome, because we have also been told if we don't get more kids in our classes one of the teachers in my department will probably be cut within 2 years” (E-65).

Questionnaire Findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement: I need to learn strategies to increase student enrollment in my CS classes. Overall, 63.9% of the questionnaire participants (n=216) responded this as a need or strong need. Six teachers in the questionnaire shared comments about increasing student enrollment in CS classes. For those teachers, being an elective class, perceptions about programming and topics chosen for CS classes were challenges for increasing enrollment in CS

classes. For instance, one teacher was looking for strategies and shared student enrollment as a challenge in elective programming classes:

> I think any strategies are always welcome… It is a challenge in general to get kids interested in regular programming. It is hard anyway you slice it. And, with it being an elective, it requires even more of an effort to get kids into it. (Q-15)

Another teacher explicitly shared his efforts to increase student enrollment in CS classes: "I have been reading about the need to increase student participation in computer classes. I am looking for ways to increase interest" (Q-23). When asked to comment about student related needs, one teacher directly pointed out students' fear of failure in CS: "Address student's fear of failure" (Q-31).

Interview Findings. Before conducting the interviews, the researcher considered participants' questionnaire responses related to the need for increasing student enrollment. Six out of eight final-phase interviewees expressed either a need or a strong need for increasing student enrollment in the questionnaire. When these teachers were asked to explain this need in more detail and provide examples in the interviews, the comments underlined four reasons for low enrollment:

1. offering CS as an elective,
2. structuring the class name and content irrelative to students' interests,
3. students not seeing the advantages of CS for their education and future,
4. not knowing what CS is.

For instance, one of the interview participants highlighted that CS is necessary and all the students should take at least one CS class in their school curriculum. He solicited strategies for increasing students' CS enrollment in his school: "I believe that all students should take at least one computer science class to be capable and have mastery of their world living in the 21st

century. Strategies to increase enrollment in computer science is definitely needed" (I-1). One of the interview participants emphasized the name and content of the class as influencing factors, and he was able to increase students' interest in a game programming class. However, the teacher shared that he was unsuccessful in sustaining this interest when the class is beyond a students' ability:

> Well, yeah, this year, I think there's a large number of students. I have a large number of students in the game programming class. I think it's just because they heard the words "game programming" and once they got in some of them got a little overwhelmed. (I-7)

The same teacher mentioned that students in his school did not see CS as necessary for their education, making it harder for teachers to increase student enrollment:

> They just don't see computer science as a necessary competent. It's just been very hard to get kids interested in it. It's just a very hard class and kids are already taking a bunch of difficult electives for required class. They all want to do something easier. Not often, but many times want to do something a little easier if they can. (I-7)

Another teacher emphasized that not knowing what CS is made it difficult to convince students to enroll in CS classes: "The middle school students don't really know what computer science is to want to come to our school." (I-8)

Overall. Strategies to increase student enrollment in CS classes constituted an important need for the teachers in all phases of this research. Offering CS as an elective and not connecting CS classes' content to students' interest and goals were two barriers to increasing student enrollment in CS classes. CS has a reputation that it is difficult and students were not convinced about its benefits to their education and future career.

Years of experience and background. Figure 4.3.1 shows the questionnaire findings for teachers with different years of experience and background. There were small differences between teachers with different backgrounds and years of experience regarding the expressed need to increase student enrollment in CS classes.

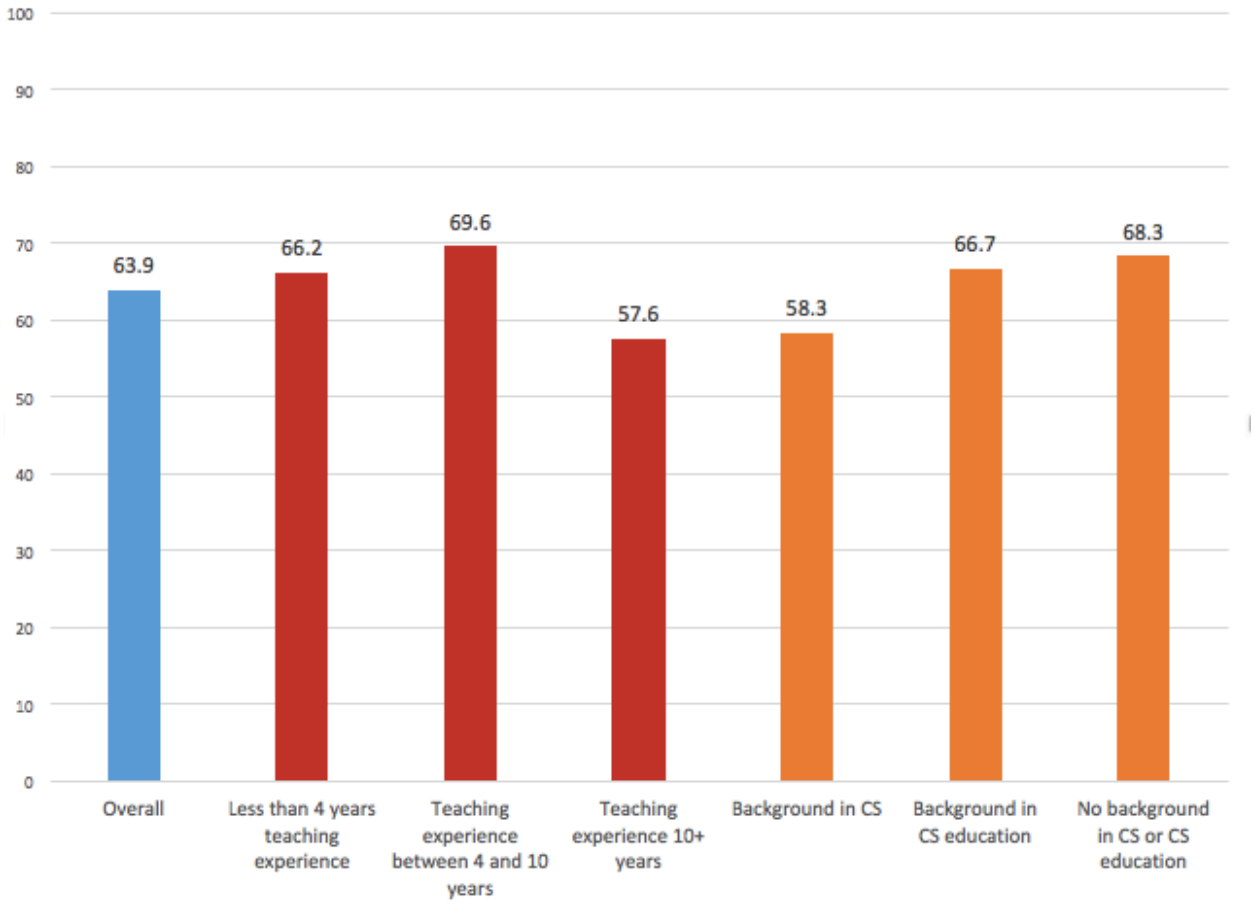

Figure 4.3.1. Frequency of teachers' perceived need to learn strategies to increase student enrollment in CS classes

Motivating Underrepresented Populations to Enroll in CS Classes

Secondary CS teachers stated that they need more strategies to motivate underrepresented populations (girls and minorities) to enroll in CS classes, which include changing underrepresented students' perceptions towards CS.

Email listserv findings. Out of 67 teachers that shared student-related needs in the email listserv, 35 teachers stressed the need to learn strategies to motivate underrepresented populations to enroll CS classes and bring CS education to all students, including females and minorities. For instance, one teacher stressed: "The problem with coding in after-school programs is the equity issue of bringing CS Education to all students" (E-7). Another teacher

shared the same opinion: “all kids, regardless of race, gender, level, etc. should get exposure to K–12 CS education” (E-36). One of the listserv participants emphasized the advantage of diversity in CS classes as leading to a diverse workforce:

> I would encourage you to keep your class (especially given its structure and your audience) very open, diverse, and inclusive. It's more challenging as a teacher, but if you make your class more exclusive, I agree with [name] - you'll shut doors for a possible future workplace that really desperately needs a diverse employee pool. (E-10)

Listserv participants demonstrated the underrepresentation of females and minorities by reporting the enrollment rates in their CS classes. For instance, a CS teacher stressed the lack of females: “only 3 girls out of 50 students enrolled.” (E-35). Another teacher also shared very low female representation in his courses: “I currently teach three full sections of the course [CS course], consisting of about 73 students (71 male)” (E-63). At the end of one discussion thread in the listserv, one teacher stated increasing underrepresented student population as his goal for CS classes: “Getting girls and other under-represented groups into CS is really what I'd like to achieve, so I'll definitely look into both” (E-64). Another teacher in the listserv shared the same goal and need to change the CS stereotype for higher enrollments:

> My goal is to encourage both genders, but especially girls, to see themselves as creators using computers not just end-users of applications. And also to try to change some of the stereotypes that CS and tech fields are for geeky boys who have no social lives. (E-65)

One teacher discussed some strategies to increase girls’ interest and realized that highlighting only the programming portion of CS might reduce female students’ interest to CS: “I might add that "coding for the sake of coding" is a turnoff for girls” (E- 36). Similar to this example, one teacher expressed that the topics chosen for CS classes should be related to different populations’ interests:

> I agree with you Barbara about the importance of carefully deciding on the topic. I am aiming to "hook in" kids from groups that are under-represented in CS, and I appreciate

> your suggestion about how to appeal to females. I'll definitely check out your link about Media Computation. (E-34)

Questionnaire Findings. The phase 3 questionnaire asked teachers to reflect on the following statements regarding the need for increasing underrepresented populations (girls and minorities) in CS classes:

1. I need to learn strategies to motivate girls to enroll in my CS classes;
2. I need to learn strategies to motivate underrepresented populations (e.g., African-American, Hispanic) to enroll in my CS classes.

Overall, 73.1% of the questionnaire participants (n=216) responded "learning strategies to motivate girls to enroll in CS classes" as a need or strong need. Six teachers' responses were not applicable to this question. Overall, 62.0% of the questionnaire participants (n=213) responded "learning strategies to motivate underrepresented populations (e.g., African-American, Hispanic) to enroll in my CS classes" as a need or strong need. Thirteen teachers in the questionnaire shared comments that stressed the lack of representation of girls and minorities in their classes, and argued that this is due to the topics offered that were not interesting for these populations.

The teachers shared their need for engaging girls and minorities in CS. For instance, one teacher commented: "How to engage girls in particular, and minorities." (Q-30) as her need. Another teacher mentioned low female interest in a game programming class and asked for strategies to increase the number of girls in CS classes:

> I think any strategies are always welcome. There have been girls in my classes before and they have done well. This year, we introduced a game programming class. It started with 16 boys and 1 girl… Overall, though, it is often a challenge to get girls as interested as boys in programming. (Q-15)

A female CS teacher shared her frustration with the low number of female students in her CS classes: "I have already seen a sway in fewer girls in my CS program, and as a female team lead, I find that very frustrating" (Q-2). On the other hand, one teacher shared her success story with using Scratch to increase underrepresented student enrollments in CS classes: "Enrollment of female and non-traditional students continues to increase in my Intro to Programming and other technology electives. Scratch has worked very well to engage and encourage students to try programming" (Q-32).

Interview Findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to the need for increasing underrepresented student enrollment. Six out of eight final-phase interviewees expressed either a need or a strong need in the questionnaire for motivating girls to enroll CS classes in the questionnaire. Five out of eight final-phase interviewees expressed either a need or a strong need in the questionnaire for motivating underrepresented populations (African-American, Hispanic) to enroll in CS classes. When asked to explain these needs in more detail and provide examples in the interviews, they all emphasized that CS is for all students, and underrepresented populations should be informed about and encouraged to take CS classes. The interviewees described their reasons for low underrepresented enrollments in CS classes. CS's reputation as difficult, perceptions of it as a male field and the topics chosen being perceived as related only to boys' interests (e.g., game programming) were the reasons reported that reduced girls' interest in CS. Furthermore, "not knowing what CS actually is" was reported as a factor that impacted the enrollment of minority populations.

When asked about these needs, one teacher described how he/she envisioned CS for all: "I would love to have a more diverse population taking computer science because it's not just for

the white kids. It's not just for the boys. It's for everyone" (I-1). When asked to explain this need, one teacher listed boys' interest in games and girls' perception of it as challenging as two reasons for low enrollment in his classes. He hoped to increase female student enrollment by rewriting his curriculum for his AP CS Principles course with less focus on coding:

> It's always a challenge… Historically it's all there seems to be far more boys who are interested in programming. I think especially with the gaming perception is the boys want to play games more than girls, I guess. Next year we're going to introduce an AP CS principles class. From what I understand about that, it's not as quite as overwhelming as jumping into a regular coding class. We're hoping to draw some more girls as well. (I-7)

Another teacher emphasized CS's reputation as a male-dominant and difficult field: "There still is that stigma. I've read many articles that the media really still makes it look like it's male dominated, and if you're into computer science, you're a nerd" (I-6).

In terms of increasing minority students in CS classes, one teacher shared that, she had difficulty explaining to low income Hispanic students and parents what CS is and getting students to enroll in CS classes:

> My school is about 99% Hispanic, 99% low income… Exposing them to computer science, this is a very new experience and in interacting with parents it's sometimes challenging to explain even what computer science is and all of the opportunities that come with it. (I-8)

Overall. The teachers in all phases emphasized the importance of increasing underrepresented populations in CS classes and providing access to CS education for all the students as their goal. In the questionnaire, a majority of the teachers reported increasing girls' and minority enrollments as one of their needs. Even though there have been efforts, the participants in this research observed their students' perspectives toward CS as a male-dominant and difficult field. Another reason identified was that the topics chosen were related mainly to boys' interests. Furthermore, "inadequate understanding of what CS is" was highlighted as a reason for underrepresentation of girls and minorities.

Years of Experience and Background in CS. Figure 4.3.2 and figure 4.3.3 show the questionnaire findings for teachers with different years of experience and background for the following items: 1) I need to learn strategies to motivate girls to enroll in my CS classes. 2) I need to learn strategies to motivate underrepresented populations (e.g., African-American, Hispanic) to enroll in my CS classes. The differences were small between teachers with different years of experience and background for these needs.

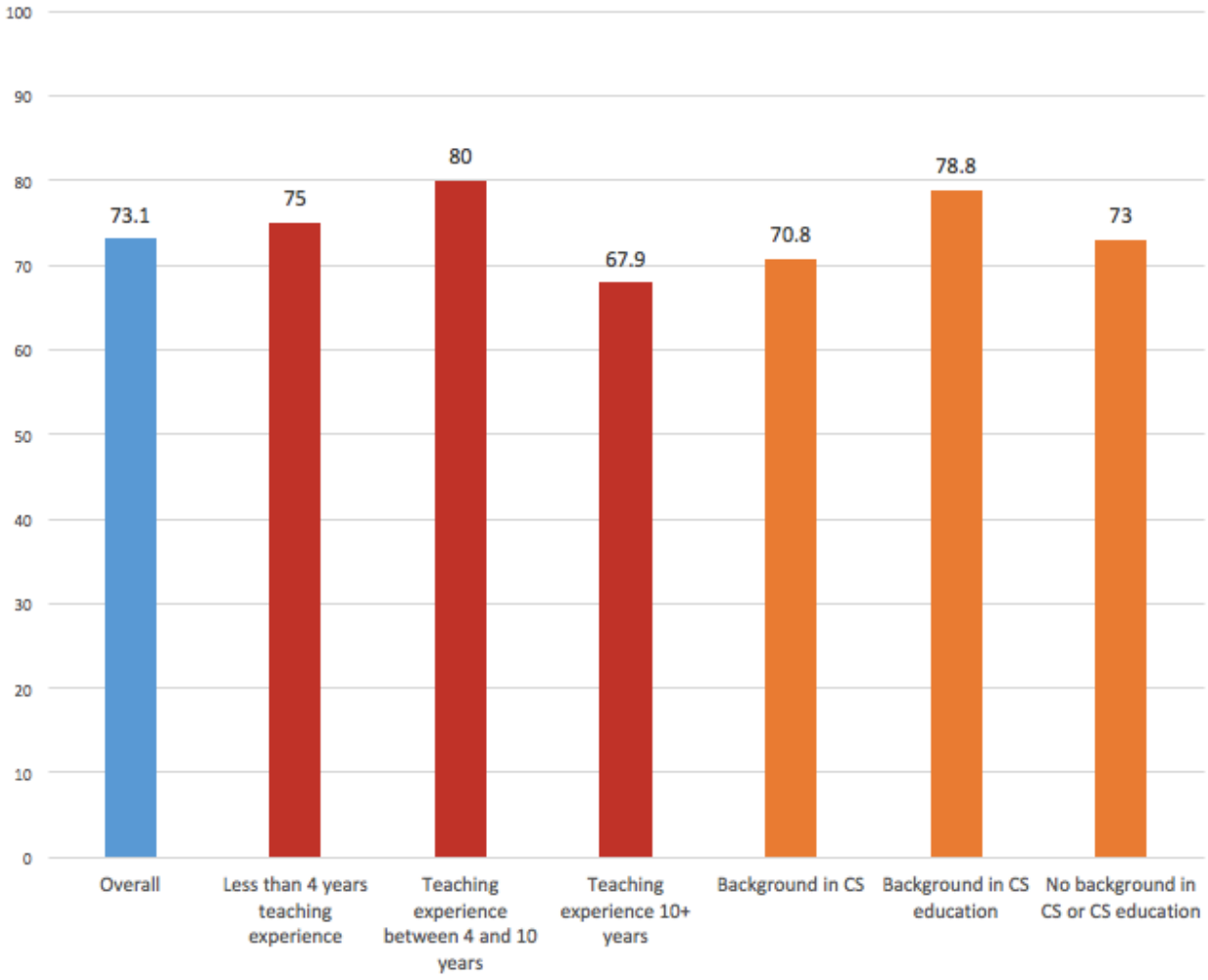

Figure 4.3.2. Frequency of teachers' perceived need to learn strategies to motivate girls to enroll in my CS classes.

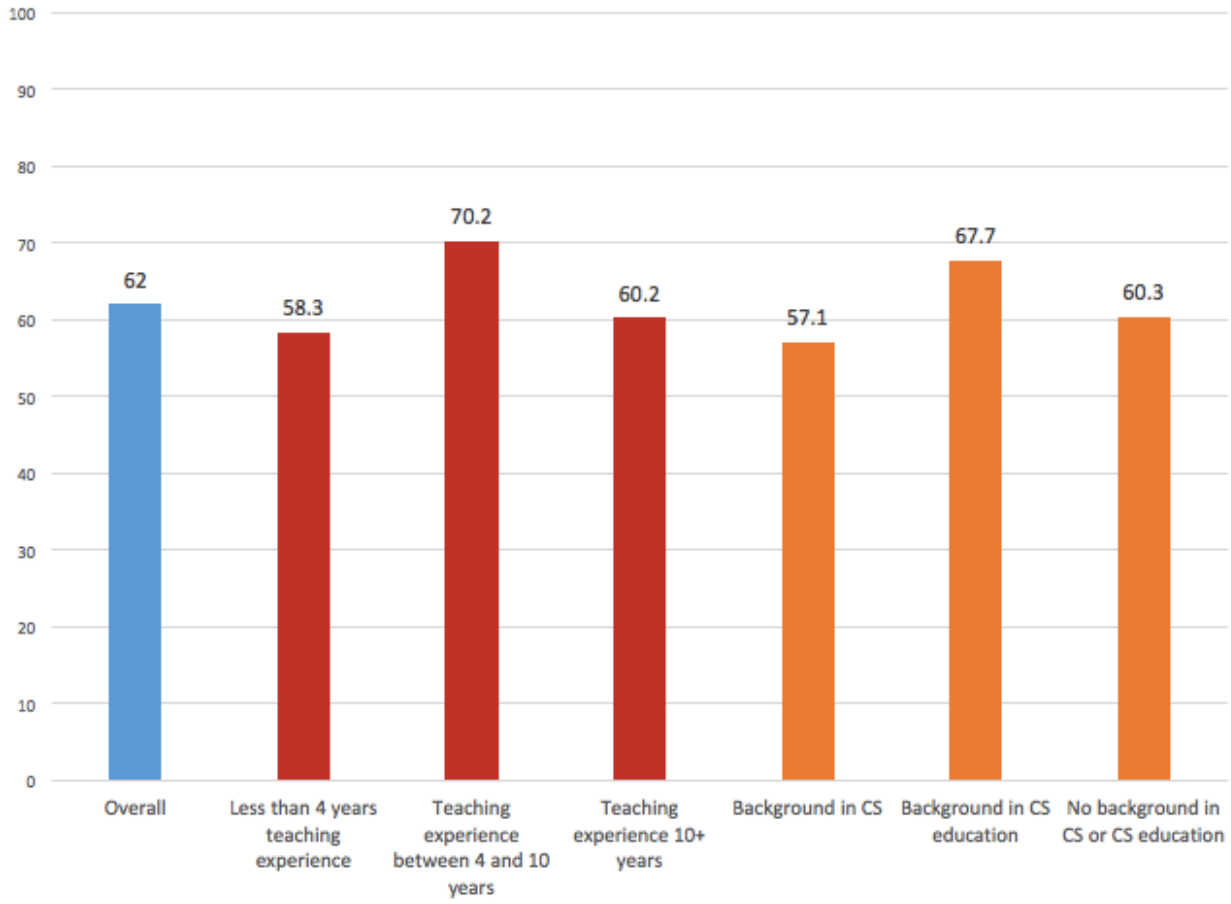

Figure 4.3.3. Frequency of teachers' perceived need learn strategies to motivate underrepresented populations (e.g., African-American, Hispanic) to enroll in my CS classes

Teaching Students with Low Interest in CS

Secondary CS teachers stated that they need more strategies to teach students with low interest in CS who are already in CS courses, which includes strategies to inform students about the challenges and benefits of CS and to provide students short term learning goals in CS classes.

Email listserv findings. Another issue discussed in the email listserv was teaching students with low interest in learning CS. Out of 67 teachers that shared student-related needs in the email listserv, 18 teachers stressed the need to increase students' interest and motivation in learning CS and defined those as preconditions, especially when learning programming. While some teachers discussed that being an elective class reduced the enrollment rates in CS classes,

some teachers in the listserv stressed this as a positive factor; thus, only motivated students and those who had interest attended their classes. Nevertheless, the need for increasing student interest and motivation in learning CS was emphasized as an important need for teachers. The following examples demonstrate how this was emphasized in the questionnaire:

- I have discovered as an in service teacher that there are a small number of factors that have a disproportionate impact on learning. They are centered on student motivation and interest. The primary question here is how do you get the student engaged and actively seeking knowledge (E-12);
- ALL kids need to be shown what code is and how it can be used, not just gamers. Interest relates to DRIVE. It is our jobs as CS educators to open students' minds and eyes to both interest and ability. (E-4)

In a discussion about increasing students' interest, teachers also shared the strategies they have tried to increase student motivation in learning CS. One teacher mentioned that she tried allowing students to create products they could sell:

> We have an after school program (started this school year) in which we are experimenting with ways to motivate our high school students to learn about creating, selling, and maintaining "software products" that sort of parallels some of what you are proposing. (E-16)

In the email discussions, some teachers supported making CS a required subject in their schools to increase enrollments in CS classes. On the other hand, some of them also suggested keeping CS as an elective to make sure only students with interest and motivation to learn take the classes. One of the teachers described this as: "all it really needs is a desire to learn the stuff, and the fact that it is an elective really helps on that front" (E-12).

Questionnaire Findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement: I need to learn strategies to teach students with little to no interest in CS. Overall, 59.8% of the questionnaire participants (n=219) identified this as a need or strong need. The qualitative data in the questionnaire were aligned with the email listserv findings. Fourteen

teachers in the questionnaire commented about student interest in CS classes. In these comments, the teachers stated that increasing interest in a need in the following conditions:

1. When the course is required, there are students in the class with no to little interest in learning CS;
2. Students come with low interest from various backgrounds;
3. It is hard to sustain student interest and motivation in programming classes.

For instance, teachers in the email listserv discussions mentioned CS being an elective as an issue for enrollment. However, when CS is required, some teachers commented that there were students with little to no interest in their classes. Therefore, those teachers solicited strategies for increasing students' interest in CS, as in the following example with underrepresented student populations:

> Students are now required to take a computer programming class before graduation - so I have many underrepresented populations and girls enrolled in my course. students who have little interest in the class. I would be greatly beneficial to share strategies on how to motivate students who have little interest in the class. (Q-33)

In a different example, one teacher also emphasized underrepresented students with low interest in learning CS and asked for strategies to increase interest:

> I most urgently feel the need to learn pedagogical techniques to support underrepresented students in my classes: students with less prior experience, or whose primary interests are outside of math and science, or who are of an underrepresented race, gender identity, class background, culture, etc. (Q-34)

Sustaining interest in CS classes was the main need mentioned in the comments. Most teachers in the questionnaire comments stressed that many students come in with interest in CS, but as the class moves forward to difficult concepts in coding, they lose that interest. They are looking for strategies to sustain interest throughout the course. For instance, one teacher approached this issue and asked for strategies to sustain interest:

> As a math teacher, I often encounter students not interested in a content. In that class, they have to accept it because it's a core requirement for graduation. In computer science, all students enter with an interest, but some lose that interest as the year progresses. How can I keep them motivated to finish the year even when interest wanes? I don't expect to keep everyone at 100% interest - computer science simply isn't interesting to the entire population. (Q-35)

Another teacher emphasized the need for generating students' interest in the content early in the class to be successful later in the coding class: "I need to learn to 'hook' students earlier so they do not simple fundamentals before getting into the finer points of coding" (Q-29). Increasing interest in AP classes was also a need for some teachers, as in the following example: "I am always interested in how to increase the interest in AP Computer Science courses" (Q-36).

Interview Findings. Before conducting the interviews, the researcher considered participants' questionnaire responses related to the need for learning strategies to teach students with little to no interest in CS. Seven out of eight final-phase interviewees expressed this either a need or a strong need in the questionnaire. When asked to explain this need in more detail and provide examples in the interviews, the interviewees discussed the problems when the class is offered as a required or elective course. They stated that there were students in their classes who lost interest in in the subject, and considered it both unrelated to their needs and too challenging. Therefore, most of the interviewees solicited strategies to motivate these students with low interest into learning CS.

For instance, one of the teachers stated that there were not many electives for students to choose from in his school, so students with low interest were forced to take his class. He expressed the need to be able to help those students care about and begin to learn CS concepts and skills:

> I do get a lot of kids in my classes because we're a small school, we don't have a ton of electives. Sometimes kids just get dropped into classes they don't necessarily want, but I still feel that there's value for those kids to understand. (I-2)

Another teacher shared a strategy that he used to increase enrollment: offering a game programming class, but he still ended up losing students' interest when more challenging tasks came up:

> Well, yeah, this year, I think there's a large number of students. I have a large number of students in the game programming class. I think it's just because they heard the words "game programming" and once they got in some of them got a little overwhelmed. (I-7)

One of the interviewees proposed the solution as having the option to explain the course content and its benefits to students before they register:

> Typically, the enrollment I get, I'll have 24 to 30 kids in the class, of which 3 of them, maybe 4 are interested in computer science and the rest of them have no idea why they're there. They have to take something and the guidance counselors put them in. I would like to be able to get people to understand what the course is going to entail and what they're going to get out of it before they end up coming into it. (I-4)

Another teacher shared the idea of explaining the skills required for completing and potential outcomes of a CS project for sustaining students' interests:

> I think the challenge for me is showing the students the potential end results of this is what we're going to be doing with the project, so you really do need to learn these. I try to do that on our last project, and at the end of the project I had a couple of, a handful of students come up to me and say, "You weren't kidding when you said we needed to have these skills." I was like, "No, I definitely was not. I was teaching you these skills because I knew that was exactly what you needed," sadly they did not come to that realization until it was too late to really master the skills they needed. They didn't see how those were going to be important. (I-8)

Overall. Increasing students' interest became an issue for the participants when a CS class is required or students enrolled in a class when they did not have another option. Most participants in all phases of this research reported similar student issues and asked for strategies to be able to increase those students' interest and motivation to learn CS in their classes. They believe when a student does know what CS is and understand both the challenges and benefits CS would provide, interest and motivation increase. Therefore, some teachers were willing to

select students before they were allowed to register. In addition, some of the teachers wanted to explain to students what the class entails before registration.

Years of Experience and Background in CS. Figure 4.3.4 shows the questionnaire findings for teachers with different years of experience and background. This need did not appear to be different between teachers with different years of experience. However, teachers with no CS or CS education background more likely to identify this as a need or a strong need to compared to teachers with a background in CS or CS education.

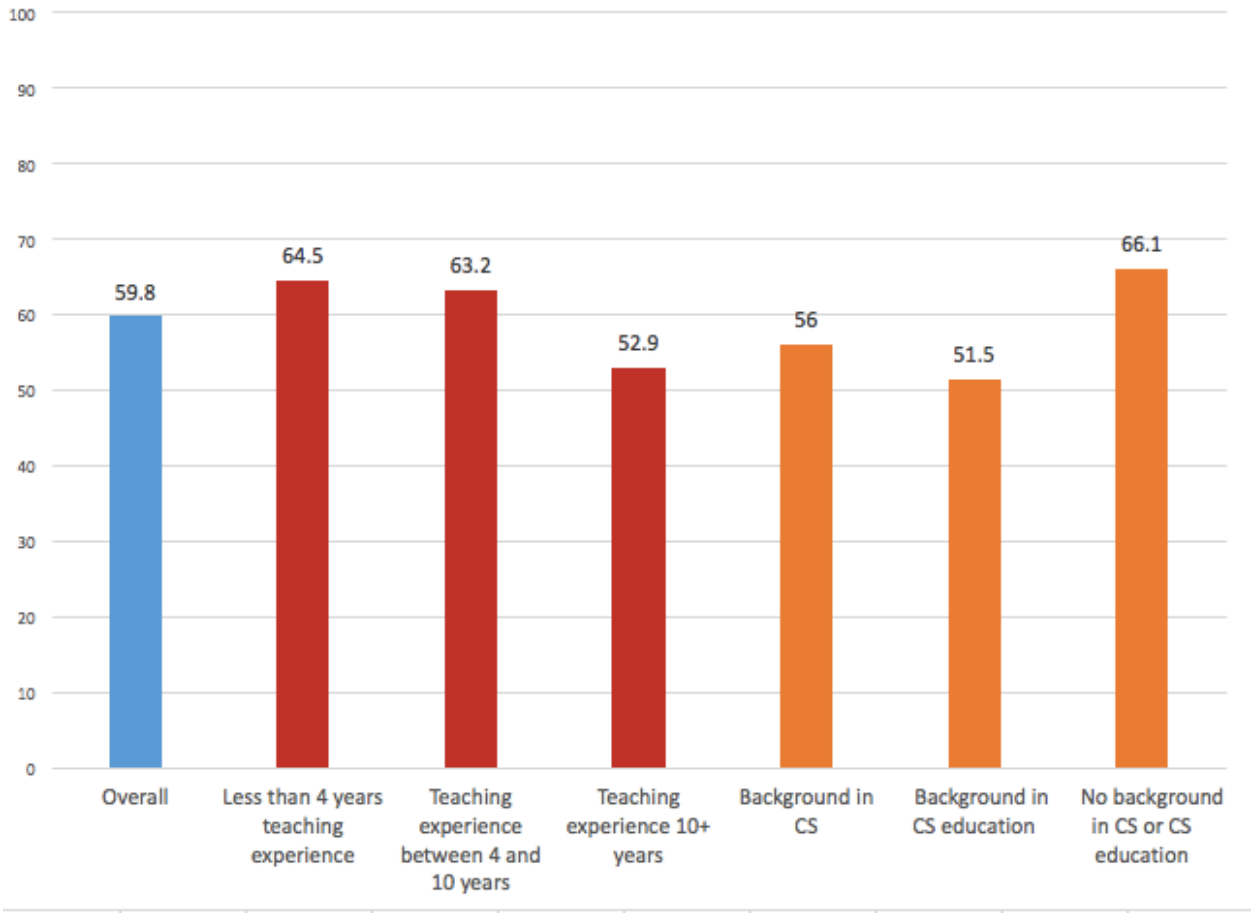

Figure 4.3.4. Frequency of teachers' perceived need to learn strategies to teach students with little to no interest in CS.

Teaching Students Who Lack Fundamental Skills

Secondary CS teachers stated that they need more strategies to teach students with low mathematics and reading comprehension skills, which includes teaching students with limited fundamental skills and developing strategies to teach students with special needs in CS classes.

Email listserv findings. Out of 67 teachers that shared student-related needs in the email listserv, 17 teachers stressed the need to learn strategies to teach students with limited math and reading comprehension. They stated that some of their students lacked these skills, which influenced their understanding and ability to apply the concepts and skills in CS classes. One teacher described the problem: "Somehow some of my students have core (base) knowledge missing or so confused that it makes it hard for them to progress" (E-37).

Math, especially algebra, emerged as the most important skill that students needed to be successful in CS classes. Some examples of this discussion were:

- I believe more has to be done giving students a strong background in mathematics and critical thinking skills (E-38);
- I have many instances in which I have to divert my curriculum to teach them Algebra concepts that should have known. (E-39)

Math skills were also reported as an important discussion topic for becoming successful in AP CS classes:

> I have one single hurdle for students. I have found that some students just aren't ready for APCS unless they have shown the aptitude in Algebra I. I require an 87 or higher in that class or permission of the instructor. The discipline and/or capability required is generally not there otherwise. (E-12)

Reading comprehension was another factor identified in the listserv, especially for understanding instructions in CS classes. One teacher highlighted her concern: "Reading comprehension is a struggle. Some of them can't follow specific instructions and don't

understand the importance of flawless execution, error free and clear thinking when writing programs" (E-39).

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements regarding the need for learning strategies to teach students with limited fundamental skills:

1. I need to learn strategies to teach students that have limited background in math (e.g. algebra);
2. I need to learn strategies to teach students with limited reading comprehension.

Overall, 59.9% of the questionnaire participants (n=217) identified "learning strategies to teach students that have limited background in math" as a need or strong need. In addition, 56.9% of the questionnaire participants (n=218) identified "learning strategies to teach students with limited reading comprehension." as a need or strong need.

Twenty-five teachers in the questionnaire shared opinions about fundamental skills they consider important for learning CS and their need for strategies in dealing with those students who lack them. These comments approached both math and reading comprehension as a set of skills. For instance, one teacher emphasized this as an evolving need with CS becoming more available in schools:

> Excellent question! As schools adopt CS for everyone, it will add the challenge of working with students with lower reading and math skills. This issue could be the game changer in CS ed since up until now the students in the CS program have typically been quite high academic achievers. (Q-26)

Another teacher underlined the same concern with the number of CS offerings increase: "CS for All is going to greatly increase the need for assistance with teaching students who have no interest, low math and/or reading ability" (Q-9).

In addition to math and reading comprehension, the comments also mentioned students with special needs as another consideration when they talked about students' fundamental skills. The teachers were asking for information about how special needs may affect students' ability to learn CS concepts and skills, and asked for ideas on how to deal with students with special needs. For instance, one teacher was asking how dyslexia might affect a student's learning in programming and computational thinking: "How to better understand how dyslexia affects a student's coding and problem solving" (Q-37). Another teacher asked for ideas about helping students with special needs: "I am always looking for new ideas especially with SE students" (Q-38).

Interview Findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to the need for teaching students with limited math and reading comprehension skills. Seven out of eight final-phase interviewees expressed either a need or a strong need in the questionnaire for learning strategies to teach students who lack fundamental skills. When asked to explain this need in more detail and provide examples in the interviews, the interviewees explained their challenges. The interviewees provided examples of scenarios where their students had difficulty on simple calculations and reading directions. Furthermore, some mentioned dealing with students with special needs and minority populations with low education background as problems.

One of the interviewees explained lack of math background as a problem with an example in his CS classes:

> Some students could think algebraically or abstractly without a transcript credit, but those students who have a limited background in math will struggle when solving complex problems. Even the very simple assignment like I'm going to prompt the user for the number of gallons of gasoline burned and the distance traveled and calculate the miles per gallon, I've seen students in 9th and 10th grade really struggle to figure out how to do that. (I-1)

When asked to define his need regarding students' prior experience, another teacher shared the same example and how he was surprised about his high school students' low math background when they need to apply it in a programming problem:

> I had a project where they had to figure out miles per gallon given how many miles they drove, how many gallons they bought. Given the cost per gallon I wanted them to figure out miles per gallon and cost per mile. I was so surprised. There's always a number of kids, 2, 3, 4 kids, who have no idea, and this is high school. No idea how to do that, how to do the math. I didn't want to make the course a math course. I wanted to build on what I thought they already knew how to do in math. It's simple algebra. Just apply it in the programming context. I was surprised at how many of them really were just lost, had no idea where to start. (I-4)

More examples were provided from other teachers about the need for dealing with students with low math skills. Reading comprehension was not explicitly mentioned in overall conversations; however, some teachers connected this issue to students' disadvantages and low content knowledge in general. For instance, one of the teachers mentioned students mainly from minority background and reported low reading and writing as a barrier to understanding content in CS:

> I do have a large number of students who are classified as English language learners. I would say the majority of my students, English is not their first language. The vast majority of their parents have very limited English and speak Spanish at home. I think in addition to the limited reading, we also run into that the students don't have very good writing skills. Trying to express themselves is sometimes difficult. (I-8)

One of the interviewees explained this further, and argued that learning CS requires a background in different content areas:

> I think it's a combination. I think the content knowledge is obviously extremely important because it is what's out there. I think it goes hand in hand with trying to figure out how to best present them to students. It involves all the related to other things they are learning about in school. It obviously ties in with a lot of math, science, English, a lot of other subjects want to make sure there's some development for students. (I-7)

Another teacher defined the reading issue as not liking to read and expecting all directions to come orally from the teacher, which limited students' individual learning:

> The deal with the reading comprehension is, there are a lot of boys, and they're not going to read. They just want me to say everything for them. They don't even like to turn the page in the book to it. I mean, how am I going to do that? (I-6)

Overall. For all the participants of this research, math background was reported as an issue for some of the students in CS classes. However, reading comprehension tended to be a problem in some contexts where there were economically disadvantaged and minority students with limited content knowledge in general. The findings suggest that learning CS requires a strong background in math as well as understanding in other content areas. Students with special needs like reading disabilities and English language learners were identified in the questionnaire comments and in one instance in the interviews. Learning strategies to deal with students with low math and/or reading comprehension skills was reported by slightly over half of the participants in the questionnaire.

Years of Experience and Background in CS. Figure 4.3.5 and figure 4.3.6 show the questionnaire findings for teachers with different years of experience and background for the following items: 1) I need to learn strategies to teach students that have limited background in math (e.g. algebra). 2) I need to learn strategies to teach students with limited reading comprehension. Teachers with less than four years of experience and 4-10 years' experience more likely to identify math skill as a need or a strong need compared to teachers with more than 10 years of experience. Teachers with "background in CS" and "no background in CS or CS education" more likely to share this as a need or strong need compared to teachers with CS education background.

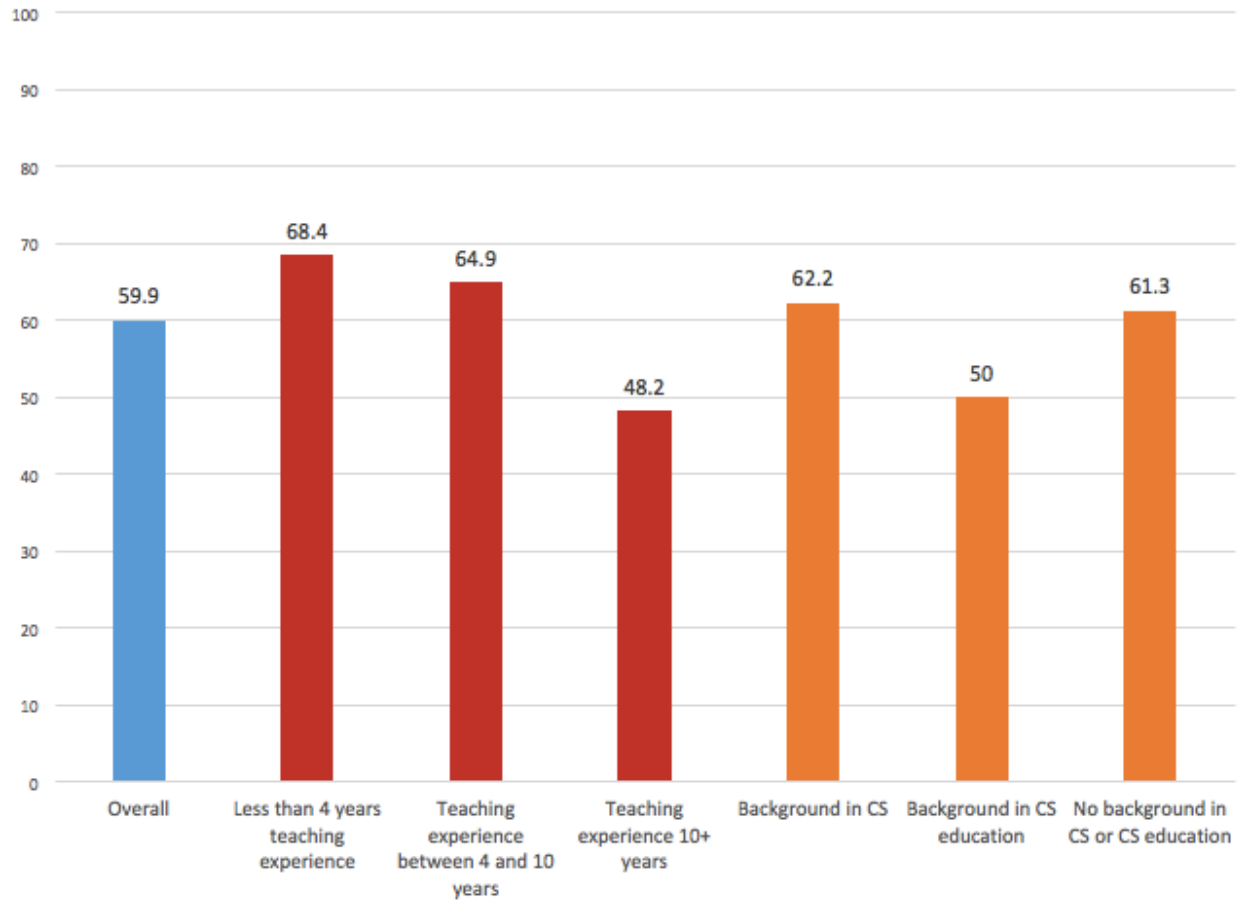

Figure 4.3.5. Frequency of teachers' perceived need to teach students that have limited math skills

Results were similar with regard to reading comprehension. Teachers with less than four years of experience and 4-10 years' experience more likely to identify reading comprehension skill as a need or a strong need compared to teachers with more than 10 years of experience. Teachers with "background in CS" and "no background in CS or CS education" more likely to share this as a need or strong need compared to teachers with CS education background.

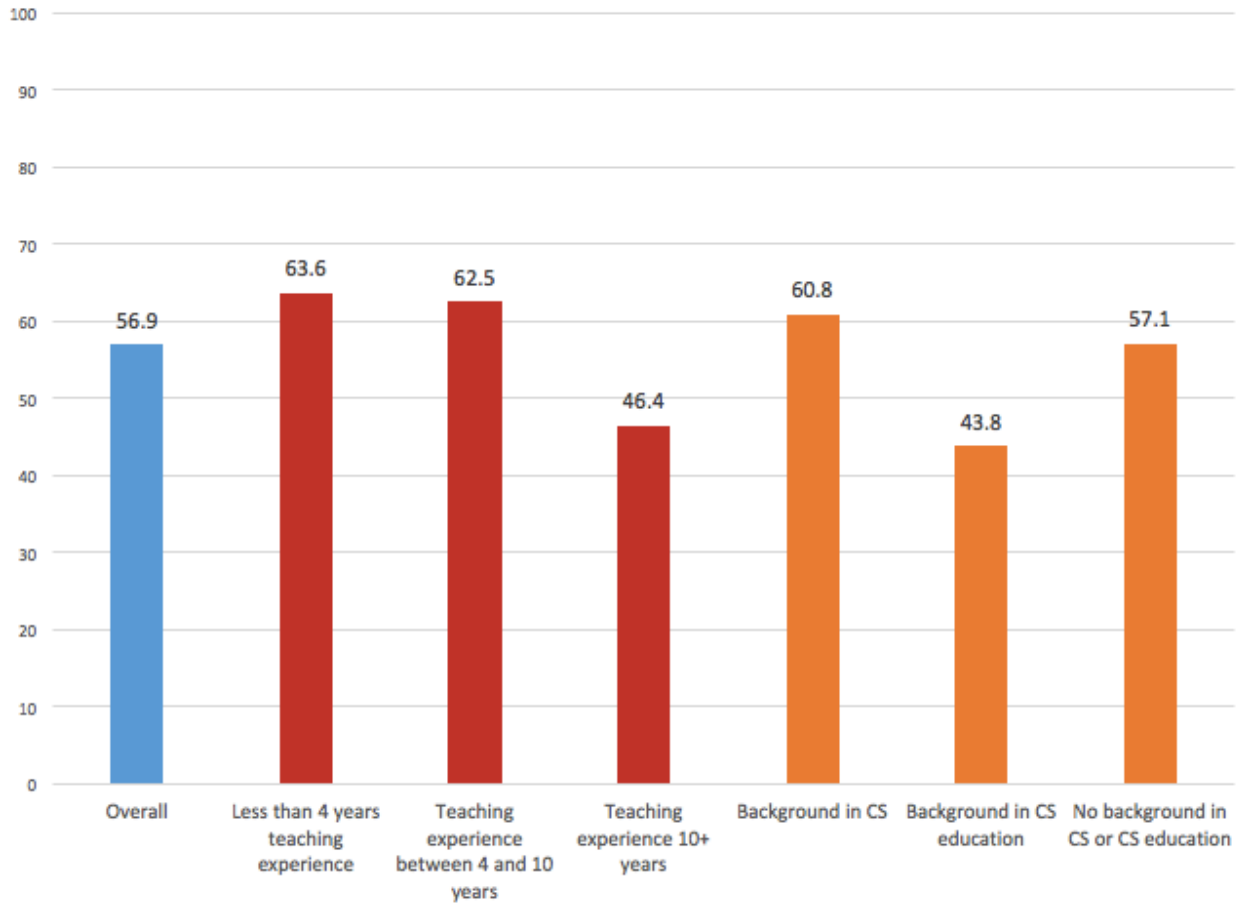

Figure 4.3.6. Frequency of teachers' perceived need to teach students with limited reading comprehension skill.

Professional Knowledge and Skills Needs

The next section discusses CS teachers' needs related to their professional development. For the purposes of this study, teachers' perceived professional knowledge and skills needs included the following sub-themes:

1. need for professional development (PD),
2. knowledge of current research and standards about CS education,

Need for Professional Development

Secondary CS teachers stated that they need more professional development (PD), which includes PD on teaching CS and knowledge of higher education institutions that offer professional development.

Email listserv findings. Out of 22 teachers that shared professional knowledge and skills needs in the email listserv, eleven teachers shared the need to continue attending PD programs and events. They emphasized developing their knowledge and skills in CS as a continuing need. For instance, one of the teachers in the listserv stressed that CS is an evolving field that requires a teacher to constantly update her knowledge and skills:

> And there's no shortage of new skills to learn. If there's one thing we know about CS, it's that the field is endlessly deep, and progressing faster than any one person can possibly master. If you think you know everything there is to know about computer science / programming / education, you're probably wrong. (E-3)

Another teacher shared the same concern and highlighted that CS is different than other sciences and requires being current all the time: "Teaching CS is about being current. This makes us different from other teaching areas. Learning new programs, teaching ideas and the latest trends in technology is more important at this point than what I learned in college" (E-40).

With the constantly changing nature of CS field, the teachers were looking for events and institutions where they can improve their CS education knowledge and skills. The teachers highlighted the need for finding institutions that focus especially on the pedagogical aspects of CS education in their communications. One teacher expressed it this way:

> I would like to know if anyone is aware of a post-secondary institution anywhere that currently offers undergraduate or graduate level study in the above subject area. Please note that I am *NOT* looking for programs in computer science. I am trying to find a post-secondary institution that instructs how to best educate students in the field of computer science. (E-41)

Another teacher shared the same need for finding institutions that offer PD for teaching K–12 CS curriculum: "Are any of you working with or know of contacts in the university sector related to training teachers to deliver the CS curricula? Would be most appreciative of introductions" (E-42)?

Questionnaire Findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements regarding the need for PD:

1. I need to continue attending professional development events to update my CS education knowledge and skills;
2. I need to learn the higher education institutions that offer professional development for teaching CS.

Overall, 81.5% of the questionnaire participants (n=222) responded "attending professional development events to update my CS education knowledge and skills" as a need or strong need. In addition, 62.6% of the questionnaire participants (n=222) responded "learning the higher education institutions that offer professional development for teaching CS as a need or strong need.

Seventeen teachers in the questionnaire shared opinions about the need for PD and institutions that offer PD. Similar to the discussions in the email listserv, the questionnaire participants emphasized the constantly changing nature of CS and the need for updating their CS education knowledge and skills. In addition to the needs in pedagogy and curriculum, the teachers in the questionnaire mentioned "knowledge of content" needs when they commented about their professional development. Furthermore, they discussed the importance of industry practices in planning the PD programs and highlighted the lack of PD events and institutions convenient to them.

The main emphasis in the questionnaire comments was a need for PD due to the constantly changing nature of the field. Most teachers commented about this need. For instance, one teacher identified updating his knowledge and skills as the biggest need:

> My intro course is a course taught across many schools. But my other courses are ones I've developed. I always want to tune and improve what I have - and try new things. So this is probably the area of biggest need. (Q-39)

Another teacher shared a similar need for PD due to the ever-changing nature of industry practices in the field:

> I constantly feel like I need to learn more and update my knowledge. The industry is constantly updating and the average teacher (like me) has a very hard time keeping up with that while also trying to teach students full-time. (Q-2)

Even though the email listserv discussions did not provide enough data about the need for "knowledge of CS content", the questionnaire comments were helpful in identifying and defining this need. These needs related to PD of knowledge of content were mostly about learning programming languages. One of the teachers mentioned her need to learn the languages she taught:

> I am trying to learn languages I am teaching students--such as JavaScript, Python, HTML, CSS, JQuery, and Ruby--so I can be more comfortable with answering student questions, as well as becoming more proficient myself. In addition, I want to better prepare myself to teach high-level languages such as C++ or Java in my high school Computer I class. (Q-41)

Other teachers shared similar statements related to learning programming languages:

- I need to understand how to program in Java better (Q-42);
- I need to learn the newest languages. I haven't had the time to teach myself Python or any App development languages (Q-43);
- I need to learn to tie things together like form to database to JavaScript. Integrating into php (Q-44);
- My biggest need, I think, is more CS content knowledge. I'm mostly self-taught. (Q-45)

Some teachers also expressed their dissatisfaction with the availability of PD events and institutions near them: "Professional Development events needs to be closer to home not in other states" (Q-46).

Interview Findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to these professional knowledge and skills needs. All the final-phase interviewees expressed the following needs as either a need or strong need in the questionnaire:

1. I need to continue attending professional development events to update my CS education knowledge and skills;
2. I need to learn the higher education institutions that offer professional development for teaching CS.

When asked to explain these needs in more detail and provide examples, the interviewees stressed the need for continuous PD due to the changing nature of the CS field. These teachers expressed a need for PD related to pedagogy and knowledge of CS content. In terms of the knowledge of content, learning programming was emphasized as a need in these PD events.

For instance, when asked to define the need for PD, teachers stressed the ever changing nature of CS. For example: "Just seems like as soon as you learn one thing, there's five new things out there. Just making sure staying on top of things is always important" (I-7). Furthermore, they explained the need as learning to become better teaching in pedagogy and content teaching CS in programming:

> You know, I think I need ongoing training for ... I think from a pedagogical standpoint, how to continue improving on running a project based, lab based, application based classroom, and then I would say actually learning how to program. Like, learning HTML, so I teach it better. Really learning how to do HTML so I teach it much better. (I-3)

The interviewees explicitly stated the need for knowing where to go for their PD and also discussed convenience as an important factor for attending such trainings. They wanted to have access to PD where they wouldn't need to spend too much time and money, and where their

specific needs can be met. For instance, the following teacher explained the need for finding institutions that offer PD:

> I am working to build the capacity of computer science teachers in my state… I don't yet know at my entire state level all of the resources available and so I need to learn the institutions that offer PD for computer science teaching. (I-1)

Another interviewee reported convenience of the PD events as an important factor for teachers' participation to these events. He also mentioned that he would prefer PD planned around high school practices and pedagogy:

> There are many languages, many tools I would love to be better at, but I don't necessarily have money or time in the summer to go take a workshop at a university in Chicago and do that. I don't know that an online course is also maybe the best choice. I don't know about that. Like I said before, if there was some regional professional development that would help teachers with time and money costs, I could go to a college a couple of hours away for a couple of days. That's much more reasonable than going to San Francisco for a week. If colleges could offer that, that's great, but the college model of education is dramatically different than the high school model. I would like to see there be more high school generated professional development that just college. (I-5)

Overall. In all the phases of this research, attending PD events was continuously mentioned as an important need for most teachers. The questionnaire findings reported this as a major need for most of the teachers. Due to the continually changing content of CS and CS education, participants asked for PD in both pedagogy and curriculum / content knowledge across different data sources in this study. However, even though knowledge of CS content did appear as a need in the email listserv discussions, those who participated in the questionnaire and interview emphasized a need for more knowledge of content, especially for learning programming languages in different comments and interviews. In order to attend PD events, the participants stressed the importance of finding institutions that are convenient in terms of time, place, and cost. Furthermore, participants wanted these events to be based on both CS education frameworks and industry practices.

Years of experience and background in CS. Figure 4.4.1 and figure 4.4.2 show the questionnaire findings for teachers with different years of experience and background for the following items: 1) I need to continue attending professional development events to update my CS education knowledge and skills. 2) I need to learn the higher education institutions that offer professional development for teaching CS.

Attending professional development events was a need or a strong need for the majority of teachers with different years of experience. Some examples of this need were stated in the questionnaire comments. For instance, a new CS teacher stated: "I am a new teacher of Computer Science. So I feel there is much I could learn" (Q-47). An experienced teacher with 24 years' experience stated that updating her knowledge in CS is a constant need: "Despite 24 years of teaching I find keeping up with changes a constant challenge. CSTA sponsored events even when on content I know thoroughly are always helpful, it is the teaching practice that I find always needs refreshing" (Q-26).

Regarding knowing where to find professional development training for teaching CS, teachers with less than four years of experience more likely to identify this as a need or a strong need compared to teachers with more experience. Teachers with "background in CS education" and "no background in CS or CS education" more likely to share this as a need or strong need compared to teachers with CS background.

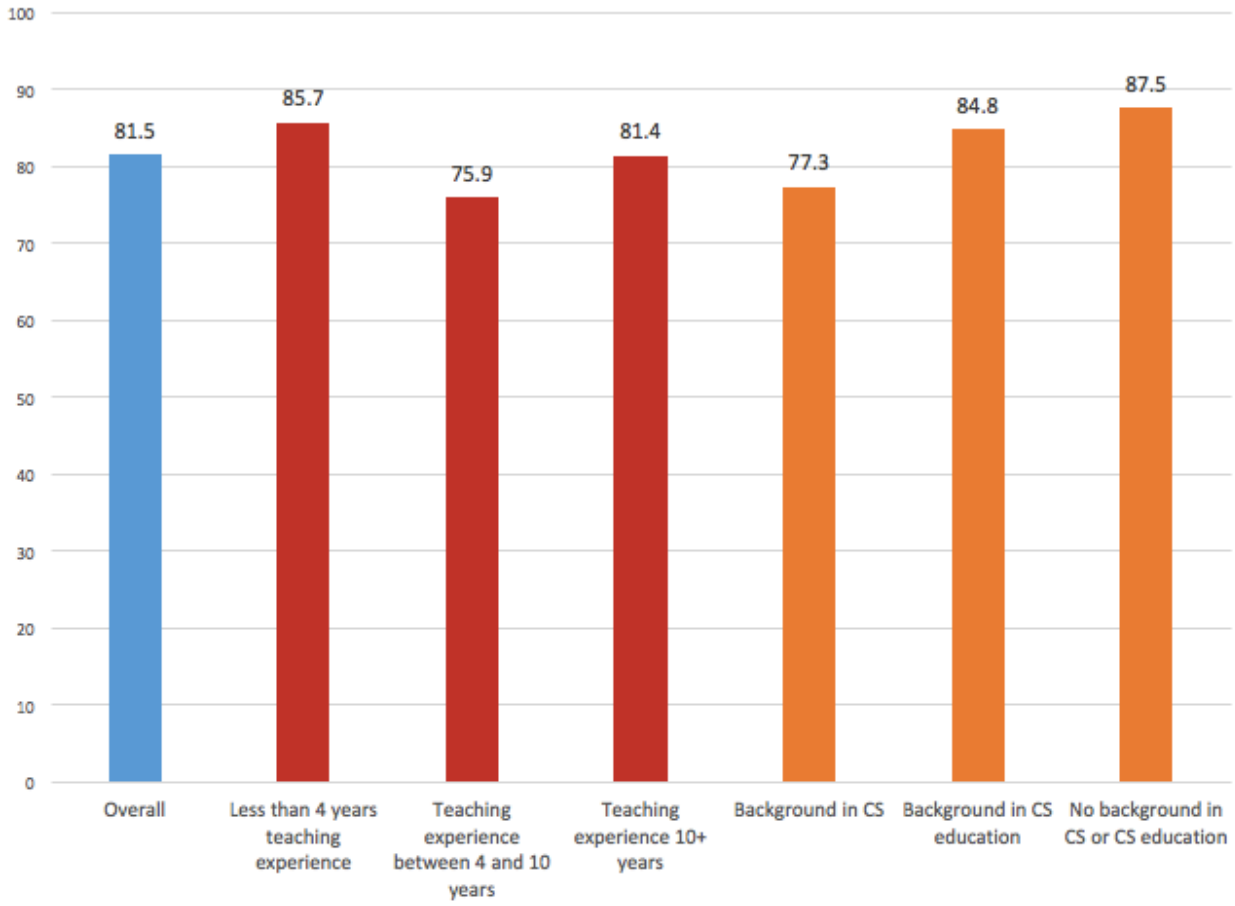

Figure 4.4.1. Frequency of teachers' perceived need to continue attending professional development events to update my CS education knowledge and skills.

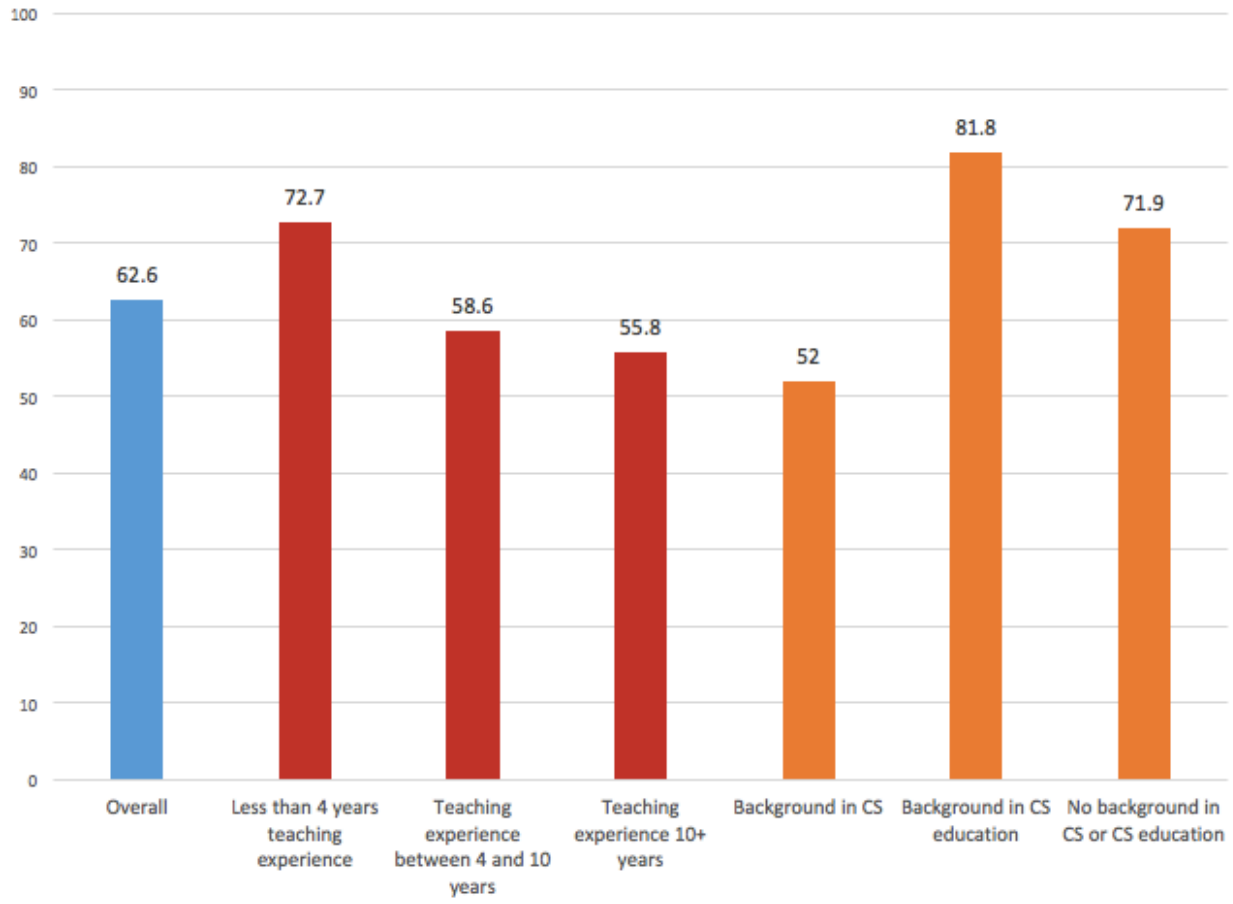

Figure 4.4.2. Frequency of teachers' perceived need to learn the higher education institutions that offer professional development for teaching CS.

Changes in CS Education Research and Standards

Secondary CS teachers stated that they need to access and use effective and practical ideas from current CS education research and need to apply state level and national CS education standards to their classroom.

Email listserv findings. Out of 22 teachers that shared professional knowledge and skills needs in the email listserv, eleven teachers mentioned transferring knowledge from research to practice as a need in their conversations. With new standards and evolving needs in CS education, teachers were interested in and needed to learn what scholars find effective in teaching CS. They highlighted the importance of making decisions based on research. For

instance, one teacher stated, “I will decide what I'm going to teach based on educational research” (E-13), and expressed the need to learn from educational research. Another teacher emphasized the need to synthesize the best practices in educational research: “I do think there is a growing need for research in general into cs ed - we have a lot of anecdotal “In my classroom…”, but it is time to actually codify some best practices” (E-43). At the time of the data collection from the email listserv, the new AP CS Principles course was not ready, but the teachers discussed the need for doing research about it. One teacher emphasized the importance of educational research to decide if the new course would be successful:

> I don't think you can necessarily point to the new AP CS Principles course or the A level in England and conclude that this is the way it should be. Hopefully in a few years we'll be able to look back at these experiments and judge whether this approach to CS for the masses is successful. (E-3)

Another factor that impacts teachers’ practices is any change in CS education standards. Organizations such as ISTE and CSTA promote CS education in K–12 and update their standards regularly. This created a need for teachers to follow and apply the standard changes in their CS teaching. The teachers in the email listserv expressed this as a need to learn about the changes in CS education national and state standards. One teacher expressed this need for keeping up with changing state level standards: “I don't know the perfect solution for our situation here in Texas, and since I'm fairly new to teaching CS, I haven't experienced all of the changes that have taken place over the duration of CS education in Texas” (E-44). Even though they were nationally announced and promoted in various locations, one teacher mentioned that she had difficulty finding the appropriate standards for a programming course, which implied the need for learning more about the standards: “When trying to set up a course (will probably end up being a math elective), I checked with our ICT experts and found that there are no standards that specifically address coding in any form” (E-45).

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements regarding the need for PD:

1. I need to learn current research that explores teachers' best practices teaching CS;
2. I need to learn about the changes in CS education national and state level standards.

Overall, 74.8% of the questionnaire participants (n=222) responded "learning current research that explores teachers' best practices teaching CS" as a need or strong need. In addition, 63.8% of the questionnaire participants (n=221) responded "learning about the changes in CS education national and state level standards" as a need or strong need.

Five teachers in the questionnaire shared opinions about the need for learning changes in CS education research and standards. The comments were limited and stressed the need for a repository of standards and research findings that teachers wanted to have access, as following: "Knowing where to find best practices and changes in standards would be very beneficial for me and my colleagues" (Q-33). Another teacher emphasized this need with a similar interest: "Scheme of studies for CS" (Q-49).

Interview findings. Before conducting the interviews in phase 4, the researcher considered the participants' questionnaire responses related to these professional knowledge and skills needs. All the final-phase interviewees expressed "learning current research that explores teachers' best practices teaching CS" as either a need or strong need in the questionnaire. Seven interviewees expressed "learning about the changes in CS education national and state level standards" as either a need or strong need. When asked to explain these needs in more detail and provide examples in the interviews, the interviewees mentioned knowing the available standards but stressed the need for finding curriculum materials that align all the standards. For instance,

one teacher mentioned the need for a resource that provides curriculum materials searchable by standards:

> If I'm trying to teach something that I'm really uncomfortable with, like maybe I'm trying to teach a standard on data representation or maybe I'm trying to teach a standard on iteration. Whatever it is, it would be nice if there was a repository where I could go find questions and find assignments and find activities and they all be correlated or filterable by a standard or a learning objective. (I-1)

On the other hand, one teacher stressed that standards should guide teaching but not lead all the decisions in a CS class:

> I say the model of incorporating standards in public education is a failed model. Where the standards lead the teaching and they put those as the number one. I think that the standards should support the teaching not lead the teaching. Teachers should be familiar with what the kids need to know. The teacher should be familiar with the standards and should incorporate those in a variety of ways into their projects, but I don't want to have projects designed just to meet the standards. Which I feel is what happens when they lead the way. (I-5)

When they were asked about the need for research in CS education, they mentioned needing a digest of research ideas from researchers that they could apply immediately to their classroom. One teacher stated this especially clearly:

> If it would be some way to place some of the best new research outcomes that are ... There's a lot of education research that's like, "It might work it might not, we don't have statistically significant outcomes," but when there is something where it's like, "This makes a big difference," it would be great to have someplace where as a teacher I could see what the latest new discoveries are. (I-8)

Another teacher expressed a desire for more practical ideas from research:

> I don't necessarily need to have a bunch of research which indicates that this is succeeding or this is not succeeding, I just want to know the ideas. I'll understand as a teacher what they're talking about and I can apply that immediately. (I-5)

Overall. Even though the need for learning changes in CS education standards and research reported with high frequency in the questionnaire, the qualitative findings were limited. The limited qualitative findings emphasized the need for a repository of curriculum materials and

ideas that match available CS education standards and frameworks, which emphasized the need for guidance in how to align standards with curriculum. In terms of the need for learning research, participants emphasized the need for accessing synthesis of educational research studies and learning practical ideas from them.

Years of experience and background in CS. Figure 4.4.3 and figure 4.4.4 show the questionnaire findings for teachers with different years of experience and background for the following items respectively:

1. I need to learn current research that explores teachers' best practices teaching CS;
2. I need to learn about the changes in CS education national and state level standards.

"Learning current research that explores teachers' best practices teaching CS" was a need or a strong need for the majority of the teachers in the questionnaire but this need did not appear to be different between teachers with different years of experience in teaching and background in CS. On the other hand, even though the difference was not a major one, teachers with less than four years' experience were more likely to report "learning about the changes in CS education national and state level standards" as a need or strong need compared to more experienced teachers.

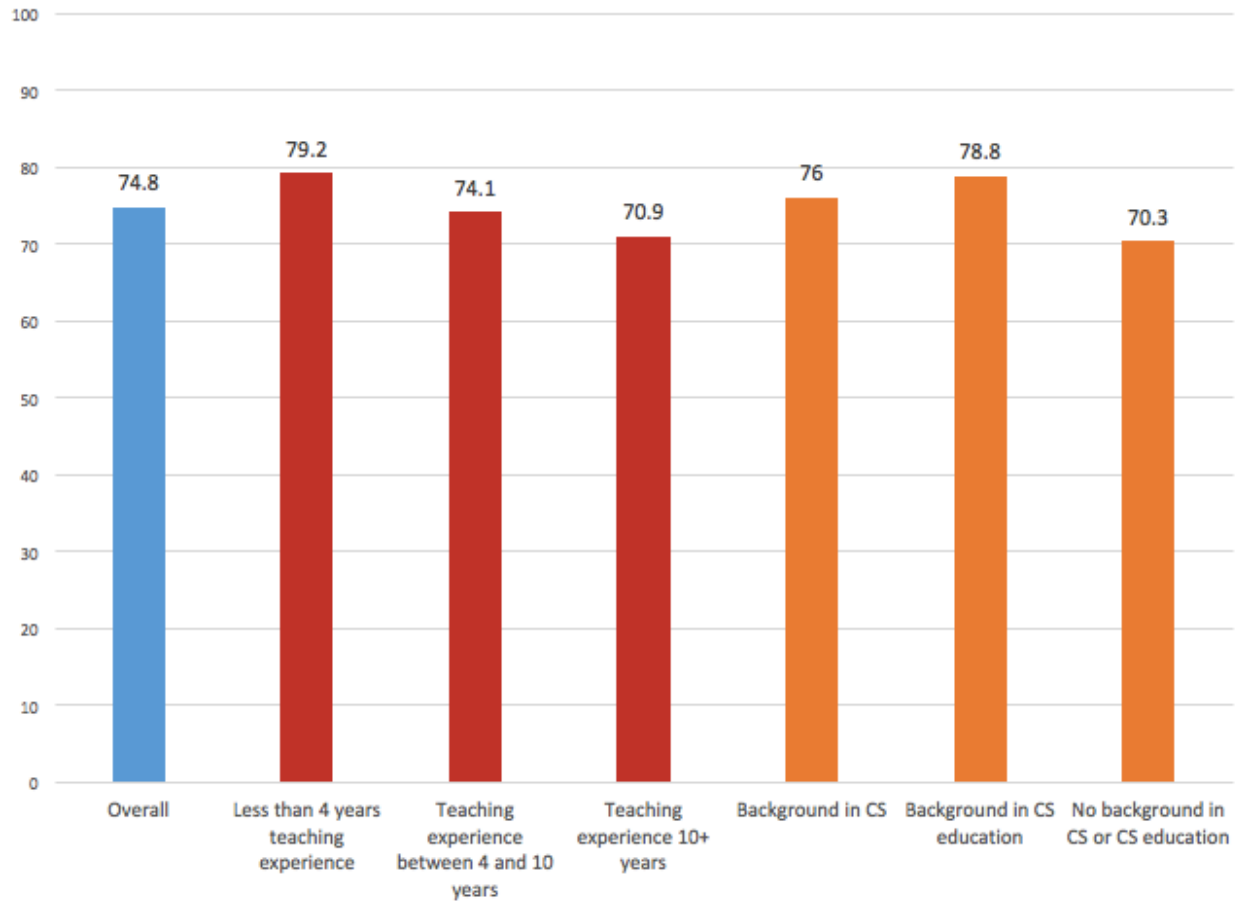

Figure 4.4.3. Frequency of teachers' perceived need to learn current research that explores teachers' best practices teaching CS.

More teachers with "no CS or CS education background" tended to report this as a need or strong need, compared to teachers with CS or CS education background. However, the difference was small.

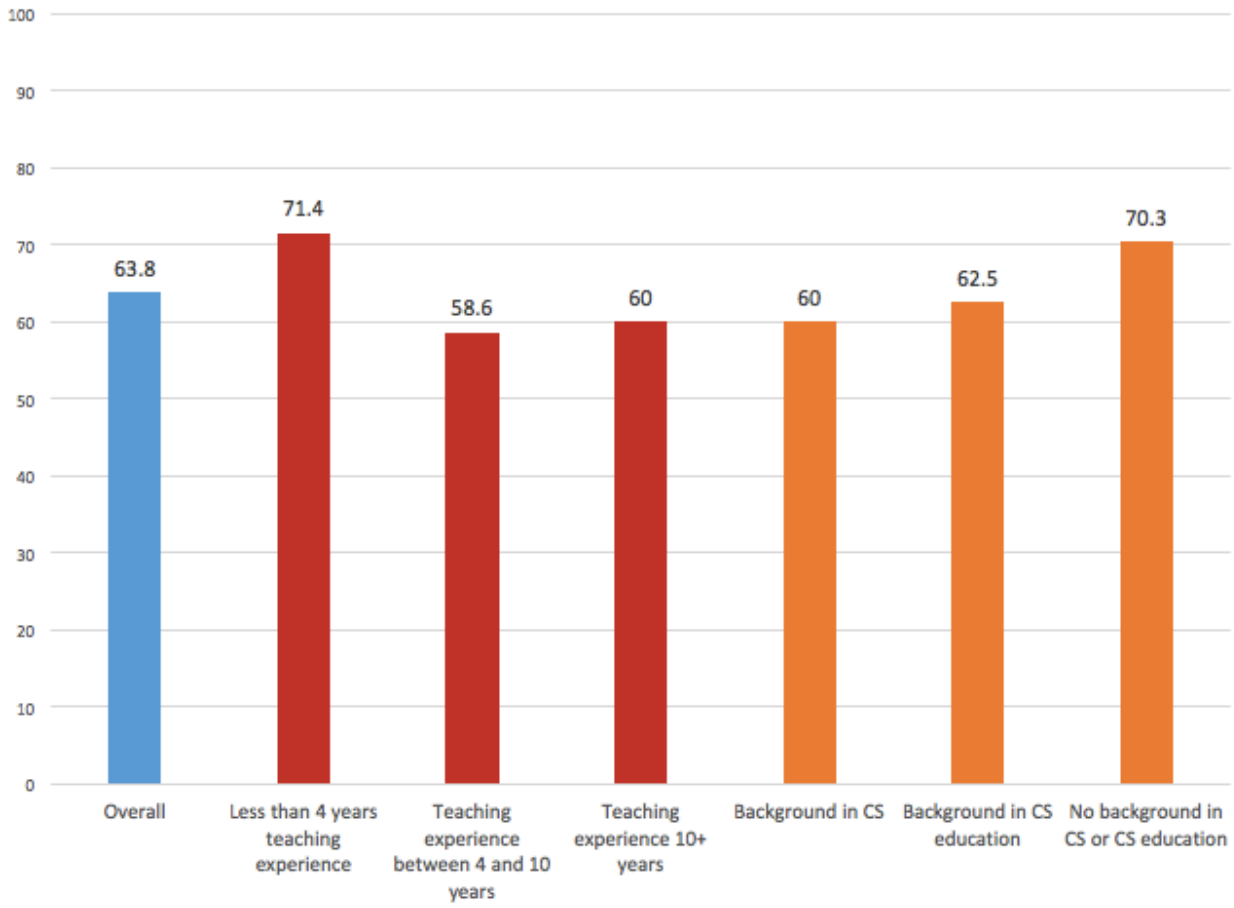

Figure 4.4.4. Frequency of teachers' perceived need to learn about the changes in CS education national and state level standards

Resource Needs

The next section discusses CS teachers' needs related to resources. For the purposes of this study, teachers' perceived resource needs included the following sub-themes:

1. computer lab environment,
2. funding and time.

Computer Lab Environment

Secondary CS teachers' stated that they need a computer laboratory environment designed with the necessary hardware, furniture, software, and network accessibility for teaching and learning CS content.

Email listserv findings. Out of 49 teachers that shared resource needs in the email listserv, 31 teachers mentioned problems related to environment, specifically poor functionality of technology equipment and furniture, software, and networks.

The teachers primarily emphasized their need for more functional and durable computer lab equipment (furniture, keyboard, mouse etc.). One teacher stressed: “I have the best-looking computer furniture suite with the worst functionality” (E-46). In addition, teachers complained about their equipment’s poor design (e.g., “I am looking for desks and chairs that can be used for at least 10 years” (E-18)), and mobility (e.g., “I'm not sure about specific tables and chairs, but make sure they have casters” (E-4)). Furthermore, they stressed the importance of using correct keyboard for health concerns, because their students might spend long hours typing code. One teacher emphasized the ergonomics of peripherals (e.g., keyboard, mouse, pointer) for CS practices: “different sorts of occupational overuse injury need different types of "ergonomic" keyboards and mice/pointer devices” (E-67).

Another discussion about environment was related to the need for lab room design for collaboration between students and communication with the teacher (e.g., “It’s good for students to collaborate and everyone can see the front board” (E-47)). Other teachers agreed that lab design is an important factor for better classroom management: “Do you want to be able to see the students' screens at all times?? That is an important consideration. When my students know that I can look up and see every screen, they stay on task” (E-66).

Software needs were another topic discussed intensively in the email listserv. Teachers especially complained about software limitations and asked for suggestions for better software such as integrated development environments (IDE). One teacher expressed: “Anyone using this

website text editor? It's an App for Google. It seems to want permission to everything" (E-48).

Other teachers shared a disappointment with software tools they have used:

- I am VERY disappointed in Google for not implementing a JVM for Chrome OS, especially since they are such heavy users and supporters of Java (E-14);
- It is nice to post product, but many tools, IDEs, and languages, require too much for the casual person browsing a CS page. Many allow the production OS a standalone EXEcutable, but that has many issues with downloading, virus concerns, etc. (E-49)

In order to deal with these issues and find more effective software tools, teachers complained that their technology coordinators' wouldn't give them administrative access to computers in their own labs for security reasons. Many could not gain permission to add even the most common and effective educational tools:

> I've asked for Storytelling Alice, App Inventor, and Python but was told no because it would allow students to access things the tech people don't want them to access. So I'm trying to run it as an after school club next year with the students bringing in their own devices. (E-65)

One teacher reported an issue with gaining access to a wireless network. Some of the tools require network access and without the technology coordinator's permission, teachers unable to use those tools. For example:

> I saw a demonstration of TouchDevelop at SIGCSE last month. I want to try this out with some of my students after the AP exam. The only problem is that I would need wireless access for my students. I have a wireless router that I have set up using the school's network. That is how I plan to give my students access. According to the school's tech, the students would have access to hack our network. (Q-14)

Questionnaire Findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements regarding the CS learning environment:

1. I need a computer laboratory designed for the purpose of teaching CS classes (reliable computers, furniture design);
2. I need software (e.g. IDE, design software) that allows me to conduct CS practices in my class;

3. I need a reliable computer network (wifi, wired) in the school.

Overall, 39.9% of the questionnaire participants (n=218) identified "a computer laboratory designed for the purpose of teaching CS classes" as a need or strong need. In addition, 37.6% of the questionnaire participants (n=218) identified "software that allow teachers to conduct CS practices in their class." as a need or strong need. Finally, 32.6% of the questionnaire participants (n=218) identified "a reliable computer network (wifi, wired) in the school" as a need or strong need.

Forty-one teachers in the questionnaire shared comments related to learning environment in the questionnaire. Even though the frequency scores were low compared to previously mentioned needs, the needs related to the learning environment were explicit. There were teachers who mentioned that they were satisfied with their computer lab environment and resources, but many others complained about inadequate computers with old hardware, and those teachers believe that reliable and updated computers are necessary for them to teach CS adequately. In addition, CS teachers described that their labs' room setup is ineffective for both learning and classroom management. They were looking for ideas from other teachers to make their labs more effective. Another issue was a lack of support from technology coordinators in both installing software and accessing the school's technology services (e.g. wireless network).

Beyond their own teaching needs, many teachers stated that improving the computer lab environment would also allow more interaction between students. For example: "The biggest issue is the placement of computers in computer labs. Would be nice to have work tables to promote getting students to interact without the interference of technology" (Q-50). Other teachers support this need:

> I hate how my room is laid out. It is difficult to have my students interact when they are stuck in fixed rows of computers. From the desk in the front all I can see it the backs of monitors. I cannot load software. (Q-40)

Other teachers talked more about the need for better computers in their labs than the room setup:

- I have an operational lab (most of the time) but it is slow and not ideal both technologically and furniture wise. Though my lab is considered a repair priority, it is not always an update/upgrade priority (Q-51);
- My lab is horrible...I constantly have to prepare workarounds to the system... (Q-52);
- The lab computers are over 10 years old. I have explained to the district administration that soon there will be no browser that can be used with Windows XP. They don't seem to understand how hard it is to conduct a class with unreliable computers and networks. (Q-14)

In terms of software, the teachers mentioned using free software and integrated development environments (IDE), and highlighted that those were meeting their needs in most instances in programming classes (e.g., "Sufficient IDEs are free. We use MPLAB (C, Assembly), Eclipse (C, JAVA), XCODE (Swift)" (Q-6)). However, some teachers reported lack of administrative access to change or add computer software and hardware as a major concern, as in the following example:

> Infrastructure is a big issue. I don't have administrator access to my machines, so if something goes wrong or I need to update a version, or any of the many things that can happen, I have to contact our IT department and wait several days. (Q-53)

In addition, the teachers commented about lack of access to their schools' computer network (wired or wireless). For instance, the following teacher was blocked from using some features of App Inventor: "Can't use the wireless feature of App Inventor due to school district policies. IT department doesn't understand my needs and hasn't been able to help me or do the software installations I need" (Q-25). Therefore, CS teachers discussed needing their technology coordinators' support in order to install software and access school technology services. The

following example illustrates this need: “A supportive IT department is key, up to date equipment and quality network/wifi infrastructure is essential” (Q-26).

Interview Findings. Before conducting the interviews, the researcher considered the participants’ questionnaire responses related to resource needs. Three final-phase interviewees expressed the following needs as either a need or strong need in the questionnaire: 1) I need a computer laboratory designed for the purpose of teaching CS classes (reliable computers, furniture design) 2) I need software (e.g. IDE, design software) that allow me to conduct CS practices in my class 3) I need a reliable computer network (wifi, wired) in the school. When asked to explain these needs in more detail and provide examples, the interviewees stressed needs similar to questionnaire participants such as infrastructure, technology coordinator support, and network.

In terms of infrastructure, two teachers mentioned the need for a better computer lab. One of the teachers specifically mentioned low-quality hardware:

> My students’ hard drives have screws fall out and then the hard drives will burn up and will no longer be usable. On the average day, I probably have send about 4 students to the IT person on campus because their computer is no longer functioning correctly or at all. (I-8)

Another teacher mentioned the need for new computers: “The computer lab, they have promised me that I would get better computers next year. I'm hoping so” (I-6).

In terms of technology coordinator support, the interviewees stressed the need for access to administrative permissions in their labs or a way to access/install the software and systems that they need in order to teach CS content. One of the teachers explained the challenge and the need with an example:

> The security around the school and their computers is obviously something that is a necessity, but it becomes extremely challenging. For example, we're trying to work with MIT App Inventor and it's the intention that when we get back from spring break, that

> will be the unit that we're working on. Our tech person still had not been able to come up with a way to get the software work through the security placed on their laptops that is impossible to install the necessary drivers. (I-8)

Two teachers in the interviews said that low Internet quality/speed impacts their CS teaching. For example: "I have it, it needs to be there but it's not been pretty good. I have too many problems with my network" (I-2).

Overall. Less than 40% of the teachers reported computer lab, software, and network as needs or strong needs. However, even though this seemed a lower percentage compared to previously reported needs, the teachers who commented about resource needs reported them as strong influential factors that prevented them from teaching CS successfully. A small group of teachers reported the need for reliable computers built to a CS teacher's needs (e.g., memory capacity). The room setup appeared to be an important need for CS classrooms to allow for cooperate student group work and better classroom management. The descriptions of software needs were mainly about having access to system administrative rights or arrangements that would allow CS teachers to install and use the software tools they need. This includes school's Wi-Fi network and/or servers. Technology coordinators at the schools where these teachers work seemed to be unsupportive due to security concerns. Finally, a small number of teachers reported low Internet quality as an issue and desired better Internet speed for their classrooms' online tools.

Years of Experience and Background in CS. Figures 4.5.1, 4.5.2, and 4.5.3 show the findings for teachers with different years of experience and background for the following questionnaire items: 1) I need a computer laboratory designed for the purpose of teaching CS classes (reliable computers, furniture design) 2) I need software (e.g. IDE, design software) that allows me to conduct CS practices in my class. 3) I need a reliable computer network (wifi,

wired) in the school. These needs did not appear to be different between teachers with different years of experience and background in CS.

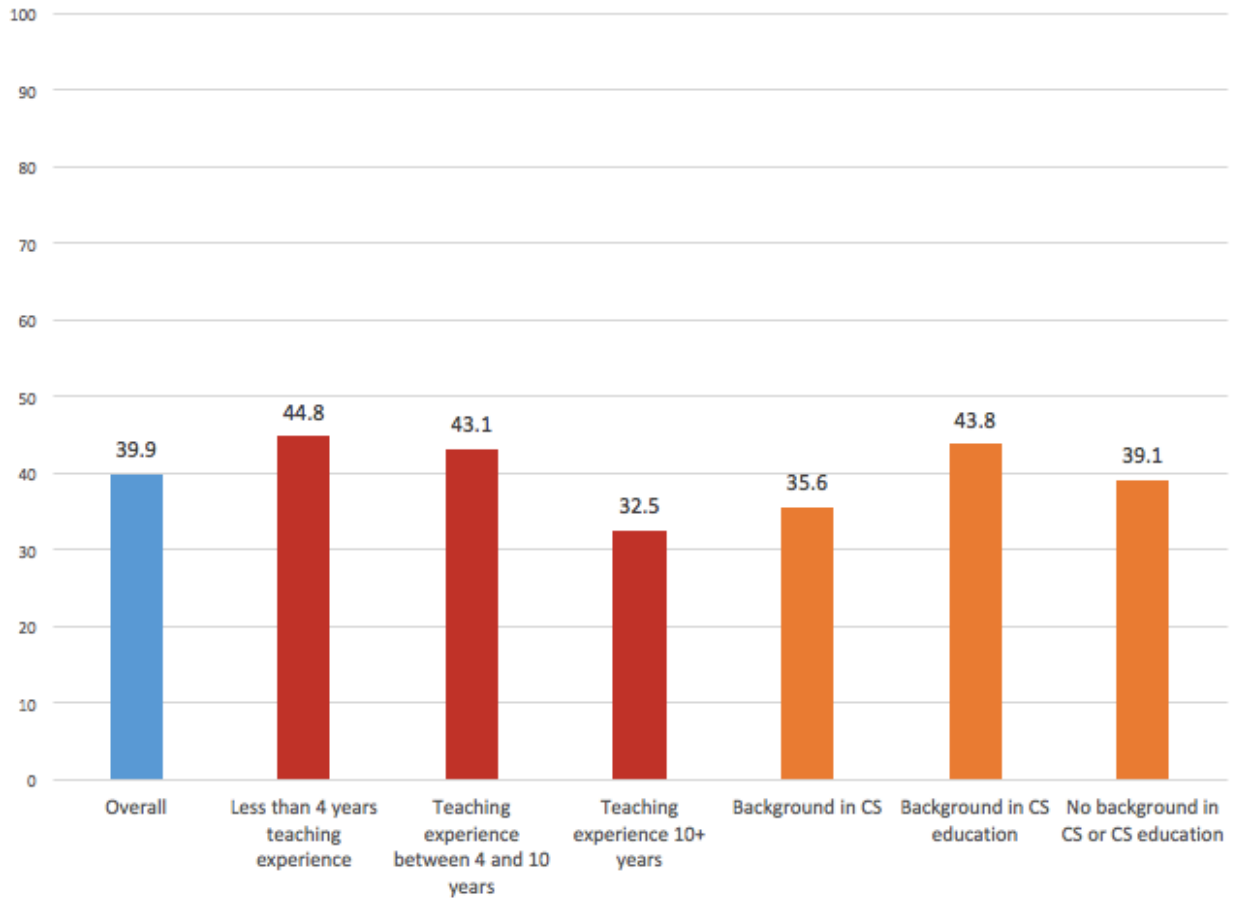

Figure 4.5.1. Frequency of teachers' perceived need for a computer laboratory designed for the purpose of teaching CS classes.

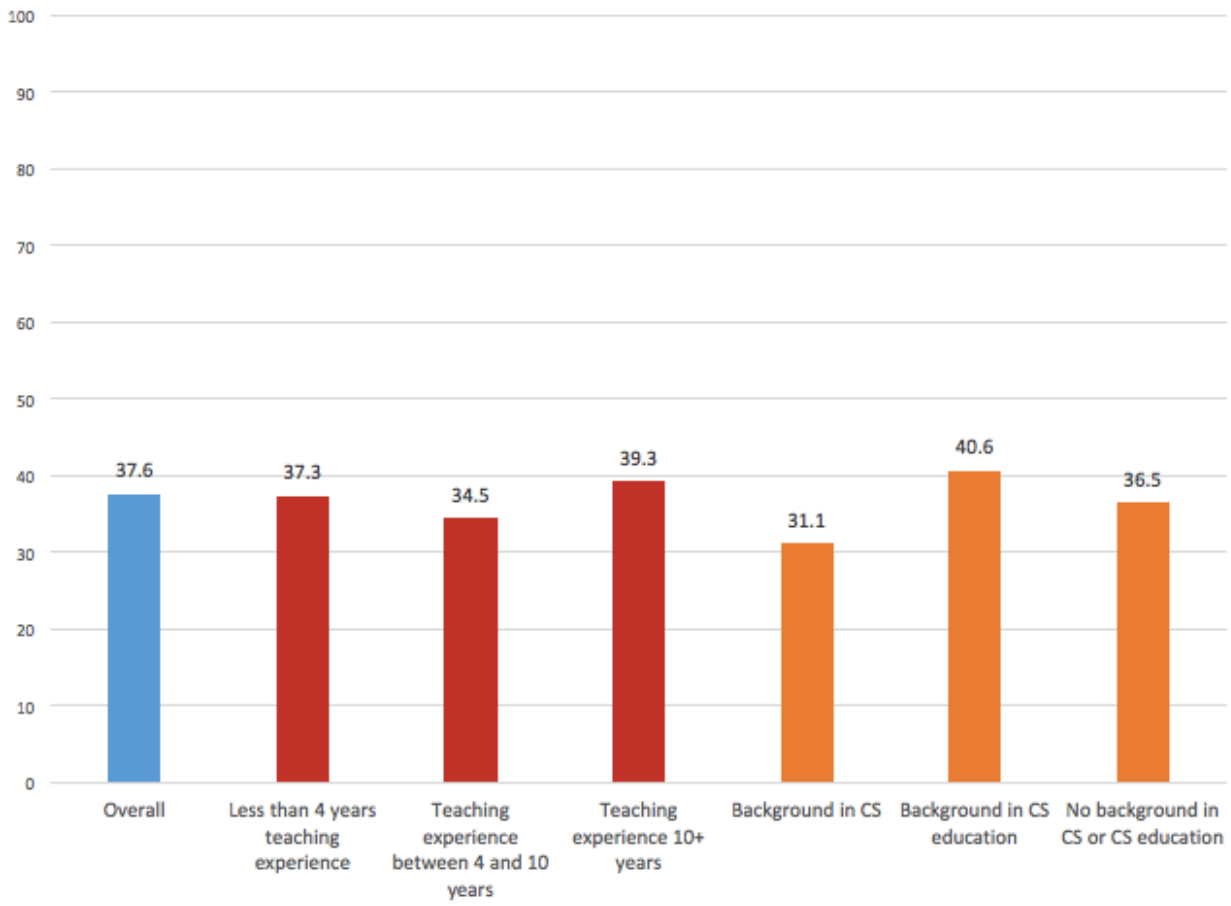

Figure 4.5.2. Frequency of teachers' perceived need for software (e.g. IDE, design software) that allows me to conduct CS practices in my class.

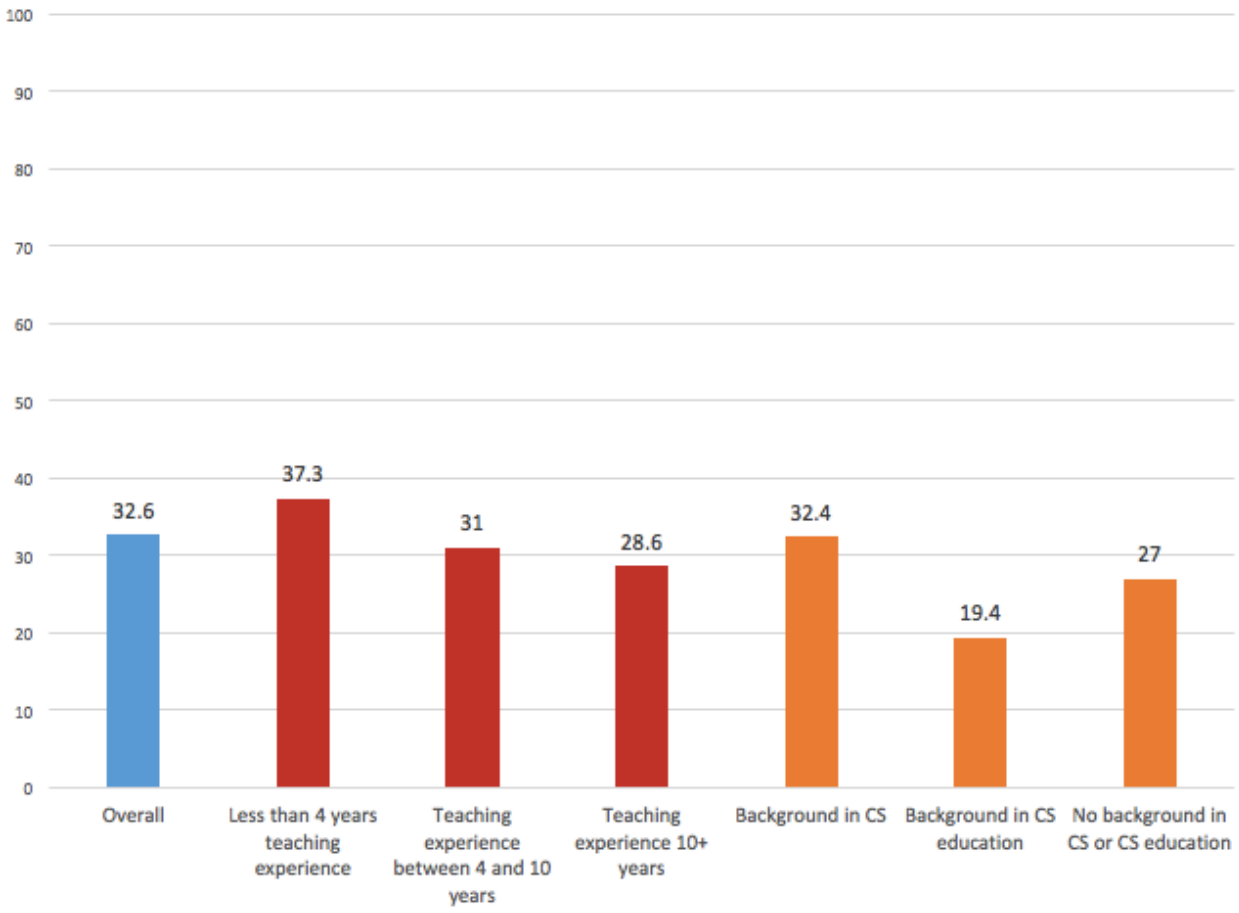

Figure 4.5.3. Frequency of teachers' perceived need for a reliable computer network (Wi-Fi, wired) in the school

Need for Funding and Time

Secondary CS teachers stated that they need adequate funding to buy technology equipment, attend PD events, and buy curriculum resources. Time becomes a need when teachers don't have the funding to meet these needs and try to improve their CS knowledge and skills and develop their own curriculum resources.

Email listserv findings. Out of 49 teachers that shared resource needs in the email listserv, 15 teachers mentioned funding and time as their needs. They stressed funding when discussing their need for professional development, buying equipment and software for computer labs, and offering CS learning opportunities for students. For instance, one teacher wanted ideas

for raising money for her CS club: “Looking for fundraising ideas for our new CS club. Something different, something other CS Clubs have had the best success with. Money means field trips and competitions” (E-68). Another teacher asked about lists of scholarships for students: “I’m looking for help locating a list of scholarships for students interested in technology especially women in technology. Can anyone help provide some links or resources” (E-70)? Others were interested in finding grant opportunities to buy computers and software. For example:

> I am at a rural private school. We need to update computer lab, but no budget. I would like to apply for grants that are available to private schools, but do not know where to begin. Any input would be helpful. We need to update all of the computers on campus. Currently running XP, with the changes in MS with no support, I need to do something. (E-69)

Some teachers were looking for funds to participate in programs that offer PD and curriculum. For instance, one teacher was looking for funding to attend the Project Lead the Way (PLTW) CS program: “Just one question. How on earth does a district find money for PLTW programs when everything else is getting cut to the bone?” (E-19). Other teachers also discussed their lack of funding to buy curriculum materials. For example:

> We still have Frontpage 2000 on our computers. The best thing I can do is teach a little HTML in it, teach them how to play with software, and then we use Photoshop and Premiere to do photo and video editing. Our school will NOT pay for new textbooks for this class so what can I get for free or what other ideas would you offer. (E-65)

There were some issues reported in the listserv that were attributed to the lack of funding to buy curriculum materials. One teacher reported that lack of time affected his ability to design targeted curriculums for his students:

> I have used Codecademy to start teaching Python to 7th and 8th grade students who have already done a robotics course using drag-and-drop GUI programming, and who are sent

back to my class for a second time. They start with Codecademy because I haven't had time to create my own Python tutorials. (E-71)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements:

1. I need funding to provide resources and tools for my CS class(es);
2. I need more time to prepare for my CS classes.

Overall, 47.5% of the questionnaire participants (n=219) identified funding as a need or strong need. Fifteen teachers in the questionnaire commented about funding. In addition, 68.5% of the questionnaire participants (n=222) identified time as a need or strong need. Seven teachers in the questionnaire commented about time.

Similar to the email listserv teachers, the comments highlighted the funding need for attending professional development programs and buying computer lab equipment/software. For example, one teacher shared that she was not able to attend PD events due to lack of funding: "Please also know that my school district will not fund PD travel over a couple hundred dollars per year. Attendance at conferences such as CSTA is not possible for me without funding" (Q-54). Another teacher looked for funding to attend PD: "I need a scholarship to help pursue more professional development in CS" (Q-56).

Funding needs for equipment and software included a variety of tools. Teachers were looking for funding to buy robotics equipment, software licenses, and new computers. For instance, one teacher was satisfied with the computers in her computer laboratory but needed funding to buy robotics equipment and defined this as her biggest need: "My greatest need is funding to stay current with technology. Funding for parts for robotics, software updates, sensors, for example Arduinos, raspberry PI the sensors and motors to do things" (Q-23).

Among all the requests for funding, the comments primarily emphasized the need for money to buy curriculum materials. This need appeared to correlate directly to a high workload for teachers, as they were then forced to create their own curriculum materials. For instance, one teacher reported that she teaches multiple subjects and doesn't have the time to create materials. The teacher asked for curriculum materials from other teachers:

> As the sole teacher of computer science in my school with multiple different course each semester I don't have time to create my own material for each class. I look for well-designed material I can modify as I incorporate into my classes. I teach the Exploring Computer Science class. Intro to computer animation (Blender), Java 1 & 2, HTML (Web Design), C++, Robotics. (Q-23)

Another teacher summarized the lack of funding and high workload problems:

> I make almost ALL of my material. There are some decent books available, but states are slow to adopt, which means no money to acquire them, which means I buy my own individual copy and build, build, build and hope it turns out well for my students. This is NOT the most sound model of instruction. (Q-55)

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to these two resource needs: 1) I need funding to provide resources and tools for my CS class(es). 2) I need more time to prepare for my CS classes. Three final-phase interviewees expressed the funding as either a need or strong need in the questionnaire. Six interviewees reported more time as either a need or strong need in the questionnaire. When asked to explain these needs in more detail and provide examples, the interviewees also mentioned the need for funding to purchase curriculum materials, the lack of which led to a higher workload for developing their curriculum materials. For instance, when asked about her need for more time, one teacher explained how she found resources online and created materials in her limited time outside of teaching:

> I mean I could do some of it, but a lot of it's you have to go out ... You have to take the time to go find ... CS Unplugged has a lot of good resources, but it's just finding the time

to go out there and read it, research it, understand it, and then turn it around and translate it for teaching. (I-3)

Overall. The participants in all the phases reported funding and time as a problem. Funding appeared to be an important need when teachers work in an environment with limited technology and CS curriculum resources. Overall, about 50% of the teachers perceived funding as a need for different reasons. These needs included money for equipment for computer labs, PD, funding to support students' scholarships and program participation, and curriculum development. When the need for funding was about curriculum, the teachers mentioned high workload and reported the need for more time.

Years of Experience and Background in CS. Figures 4.5.4 and 4.5.5 show the findings for teachers with different years of experience and background for the following questionnaire items:

1. I need funding to provide resources and tools for my CS class(es);
2. I need more time to prepare for my CS classes.

There was not a big difference between teachers with different years of experience and background in CS in reporting a need for funding. However, in terms of the need for "more time to prepare for my CS classes," teachers with less than four years of experience and 4-10 years' experience more likely to (see figure 4.5.5 below) identify time as a need or strong need. As teachers obtained more experience, they seemed to consider time less of a need. One of the teachers shared a relationship between experience and required classroom preparation time: "I find that it takes me hours and hours to prepare for class because of my lack of expertise and experience" (Q-42). In terms of background, even though the difference was small, fewer teachers with CS background tended to report time as a need or strong need.

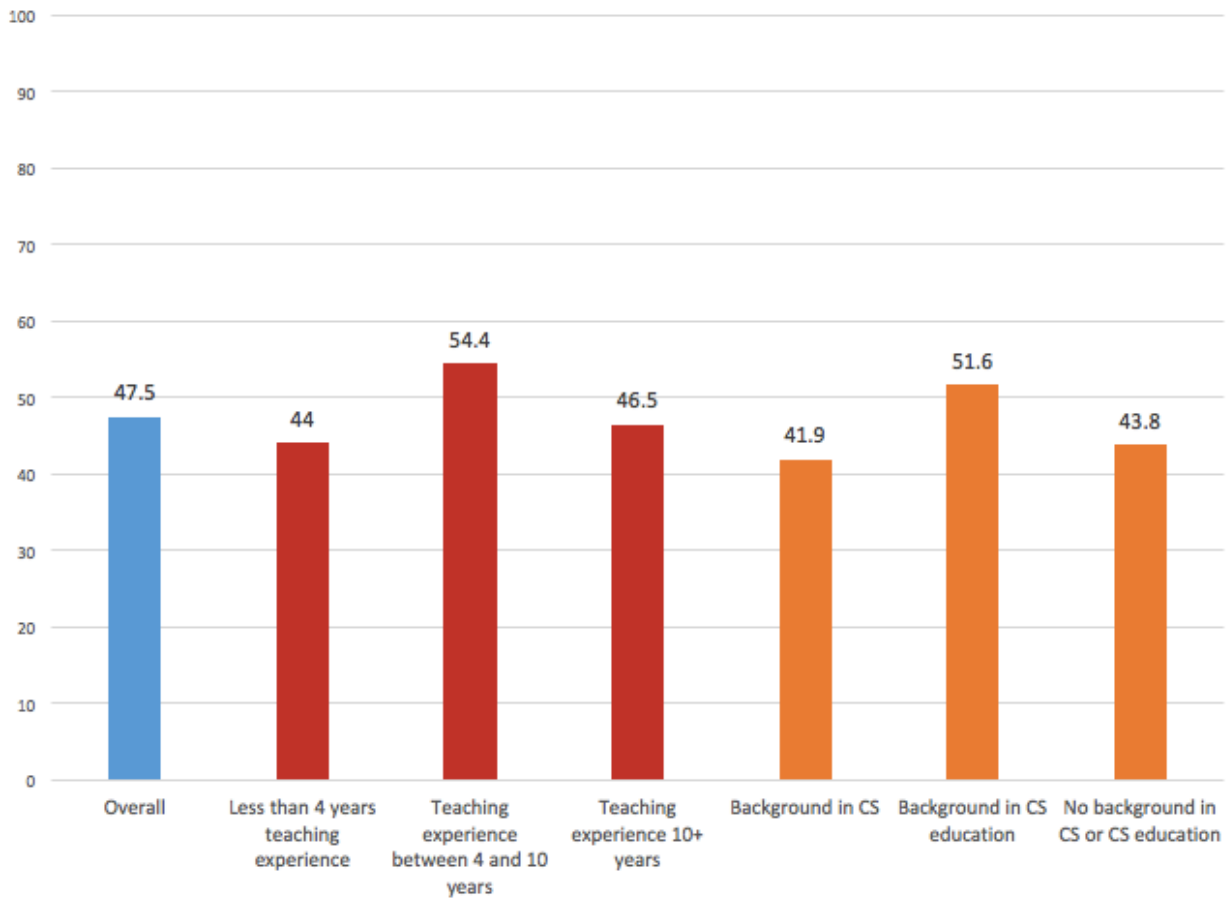

Figure 4.5.4. Frequency of teachers' perceived need for funding to provide resources and tools for my CS class(es).

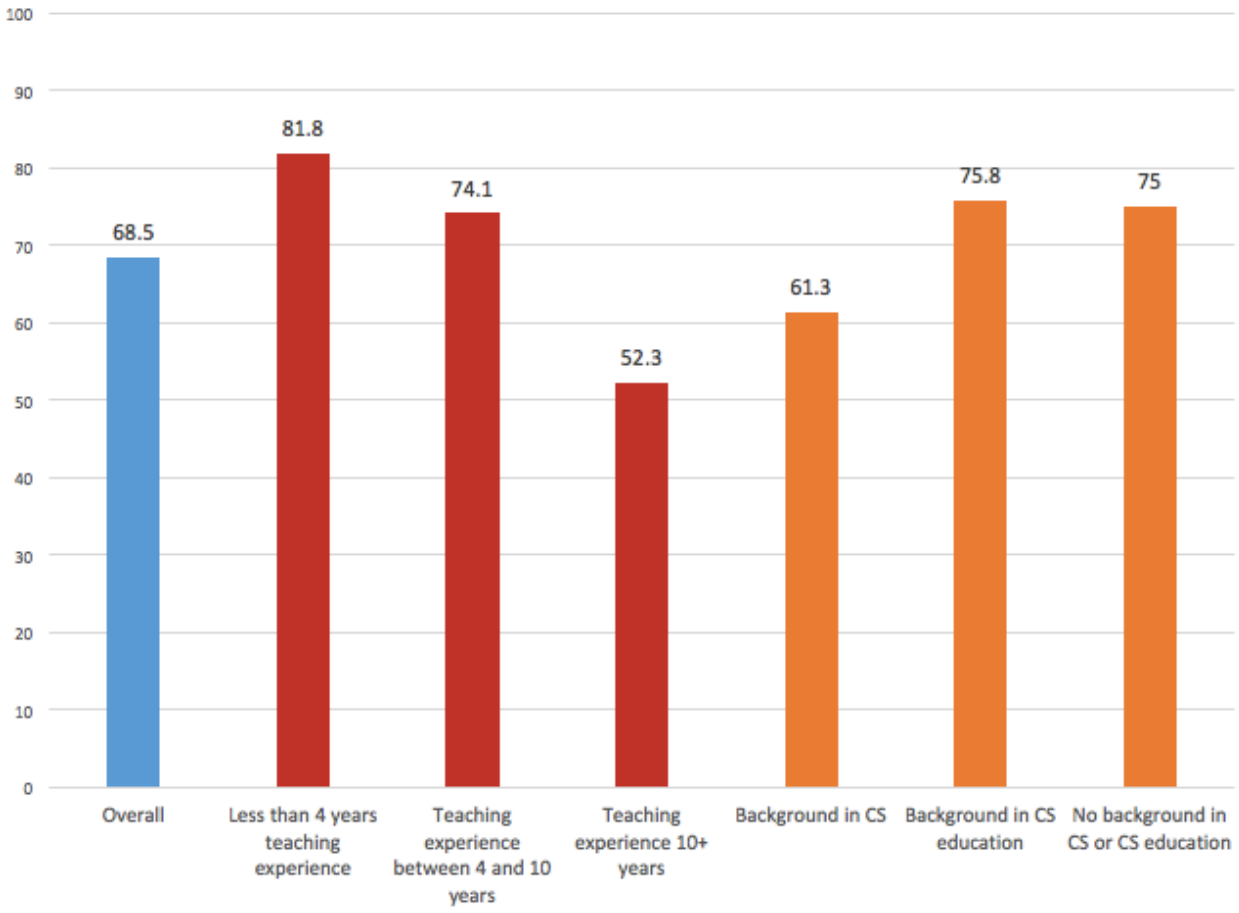

Figure 4.5.5. Frequency of teachers' perceived need for more time to prepare for my CS classes

Stakeholders-Related Needs

The next section discusses CS teachers' needs related to a support system of CS stakeholders. For the purposes of this study, teachers' perceived stakeholders related needs included the following sub-themes:

1. administrators,
2. colleagues,
3. parents.

Need for Administrators

Secondary CS teachers stated that they need administrators who understand what CS education is and support their teaching practices.

Email listserv findings. Out of 72 teachers that shared stakeholders-related needs in the email listserv, 17 teachers stressed that they need administrators who understand what CS is and support CS teachers' education practices. The misunderstanding of CS especially seemed to create an unfair distribution of resources, which in turn generated frustration within CS teachers. The emails revealed this frustration. For instance, one of the teachers in the listserv stated this lack of understanding quite simply: "Administrations just do not get it" (E-49). The same teacher emphasized that this lack of understanding caused her principal to give equal weight to classes on simple software skills as she does to advanced programming classes:

> CS is not a weighted subject. Students approach the principal about the recognized diploma which required a tech class and would in doing so lower their GPAs. Result - Principal declares both CS and the Microsoft Works classes to be honors. So same GPA for typing as for coding. (E-49)

Another teacher believed that CS education needs administrators with a background in science or engineering to be able to understand CS:

> Unfortunately, most of the administrators and paper pushers have little or no clue about computer science. They look at certification as a stamp of approval when many times it is bogus. Until the leaders of education (and the country) have some administrators who are scientists or engineers, this ignorance will continue. (E-37)

Teachers also reported frustration when their work was not acknowledged by administrators:

> I want real, current, up to date courses that help me with things like SCRUM and AGILE, Angular JS, JQuery, new IDEs .... This last Summer I learnt about Visual Studio 2015, so I could bring a fresh content to my programming class. No-one noticed except a few parents. I would like to get some acknowledgement for my efforts - it is hard work. (E-57).

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements:

1. I need my administrator to better understand what CS education is (e.g. the difference between information technology, computer science, technology integration);
2. I need my administrator to support my CS education efforts in the school/program.

Overall, 51.6% of the questionnaire participants (n=221) identified administrators' understanding as a need or strong need. Fifteen teachers in the questionnaire commented about administrator knowledge of CS. Overall, 44.5% of the questionnaire participants (n=220) identified administrator support as a need or strong need. Eight teachers in the questionnaire commented about administrator support as a need.

Similar to the email listserv teachers, the comments stressed administrators' limited knowledge of CS and emphasized the need for helping administrators to understand what it is. It was reported as crucial for administrators to know the fundamentals of CS in order to make appropriate decisions in their schools regarding school structure and plans about CS classes. When asked to comment about stakeholders, teachers stressed a limited knowledge of CS from a variety of administrative levels. For instance, one of the comments mentioned a principal approaching basic Microsoft Office software skills as CS: "Very definitely - I'm being asked to trash an AP course mobile CSP and teach 1/2 MS Word, Power-point... for half of the class instead. They still think this is CS" (Q-39). Another teacher highlighted this misunderstanding in higher levels:

> I need the County Education Office and State to become active stakeholders in CS Education. I doubt anyone in my county office knows much about either of the AP courses I am teaching (AP Computer Science, AP Computer Principles-Pilot course). (Q-6)

Some teachers believe that administrators' limited knowledge of CS leads to other issues in CS classes. For example, one teacher argued that the guidance department's lack of understanding actually influenced enrollment in her classes:

> I am very comfortable with all types of children and getting wider representation into my courses. However, there is little understanding of what I do from administration and guidance department. Guidance is key since they help enroll students in my courses. (Q-40)

Even though most of these examples referred to a lack of administrator knowledge about CS as an issue, some of the teachers were able to deal with this problem when they had a supportive administration. For instance, the following example showed the importance of a supportive relationship: "I have an amazingly supportive administration! They don't always understand, but they are willing to learn and support" (Q-51). In the following example, a teacher was able to reach her goal of offering two AP courses after talking to an administration with low CS knowledge but a supportive attitude:

> Having support by administration and parents is always a bonus. I spoke with my administration and it took me about 10 minutes to explain that AP Computer Science A and AP Computer Science Principles were two different courses. We are offering both next year. (Q-59)

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses to the following items that are related to the stakeholders need:

1. I need my administrator to better understand what CS education is (e.g. the difference between information technology, computer science, technology integration);
2. I need my administrator to support my CS education efforts in the school/program.

Six final-phase interviewees expressed administrator knowledge and support as either a need or strong need in the questionnaire. When asked to explain these needs in more detail and

provide examples, some of them mentioned administrator knowledge of CS as an important factor for allocating school resources and making plans about CS classes in schools. For example, one of the interviewees described administrators' knowledge of CS as an important need when allocating school resources and time among different subject areas:

> Let's say there's a high level of administrative meetings and there's the math chair and there's the English chair and there's not really a computer science chair. There's not someone at the meeting to represent computer science, but if your administrator knows what computer science is and can stand in when people start having turf wars or start talking about dividing up the resources or allocating time, if your administrator knows what it is that you're trying to achieve and how you do it, then they can better advocate for you in your absence. (I-1)

Another interviewee explained his perceptions about the importance of administrators' CS knowledge:

> I need my bosses and the administrators across the country to realize that it's not Microsoft Word and Microsoft Excel and Microsoft PowerPoint. It's not that. Those are different skills. I need them to understand that so that when decisions come up to find a place for Computer Science in the curriculum, that that place exists. As we expand that and as the demand becomes bigger, I want there to be administrators to see value in that and are willing to support that instead of not supporting it. (I-5)

The same teacher shared his opinions about his principal and the advantages of having a supportive administration. The interviewee discussed that as long as an administrator was supportive, she didn't need in-depth knowledge of CS:

> I think that I don't necessarily need my administrator to understand exactly what I'm doing or how I'm doing it. I need them to believe that what we're doing is right and that it's needed and it's valuable. They can trust my judgement on recommending what we do, I just need their support. If I go to them and say, "I need some money to do this for my students or I need some money for professional development or I need this software or this hardware." Or whatever the resource is or I want to start a new class. (I-5)

Overall. Understanding CS was reported as an important criterion for CS teachers' communications with their administrators. Nearly half of the participants in the questionnaire expressed a need for administrators with better CS knowledge. In all the phases of this research,

teachers reported that they expect their administrators to know the fundamentals of CS and differentiate it from other topics in technology. However, they require a supportive administration that considers their needs and provides a fair environment in the school.

Years of Experience and Background in CS. Figures 4.6.1 and 4.6.2 show the findings for teachers with different years of experience and background for the following questionnaire items:

1. I need my administrator to better understand what CS education is (e.g. the difference between information technology, computer science, technology integration);
2. I need my administrator to support my CS education efforts in the school/program.

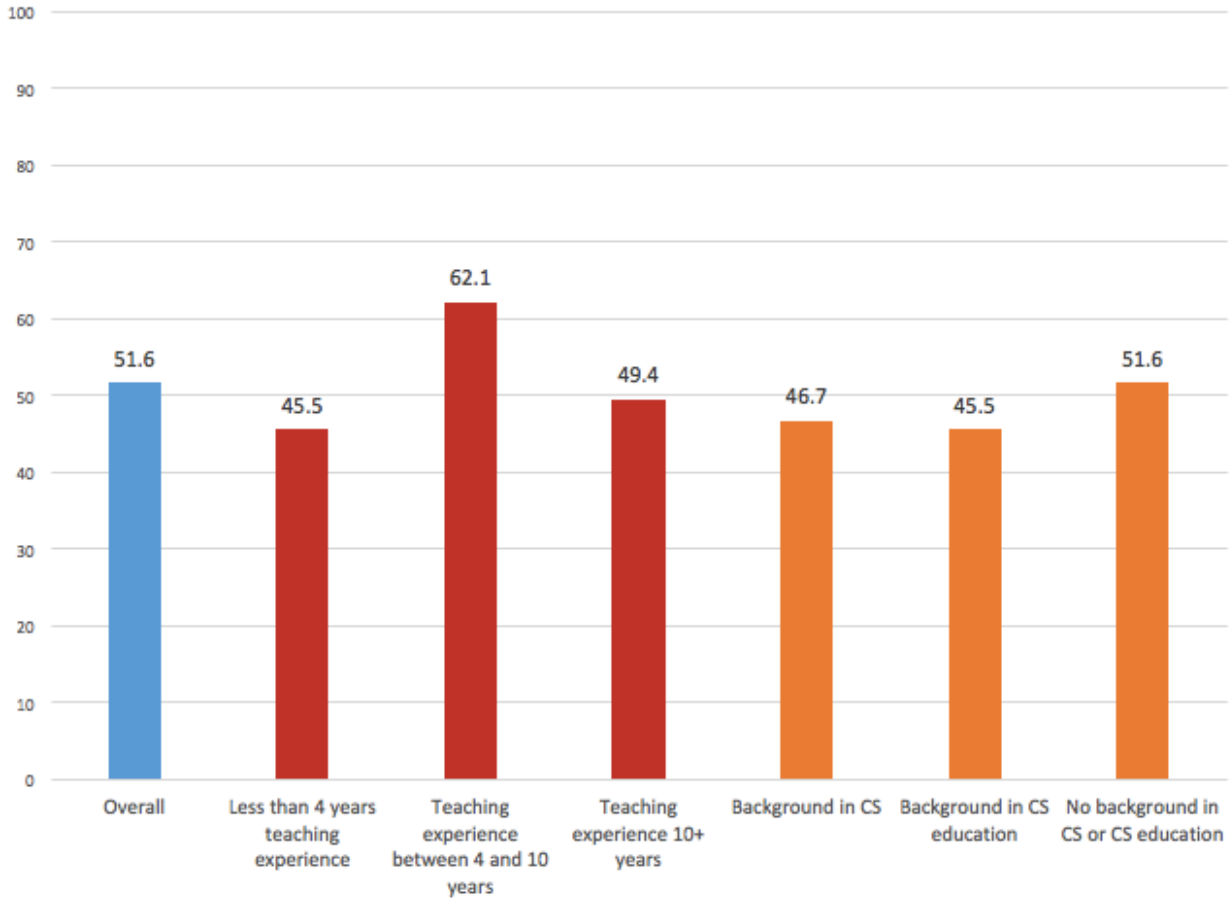

Figure 4.6.1. Frequency of teachers' perceived need for administrators to better understand what CS education is.

Teachers with experience between 4 and 10 years are more likely to identify administrator knowledge of CS as a need or strong need compared to teachers with less than 4 years and more than 10 years' experience. The questionnaire findings did not show a major difference between teachers with different background in CS. In terms of administrator support, the questionnaire findings did not show a major difference between teachers with different years of experience and background in CS.

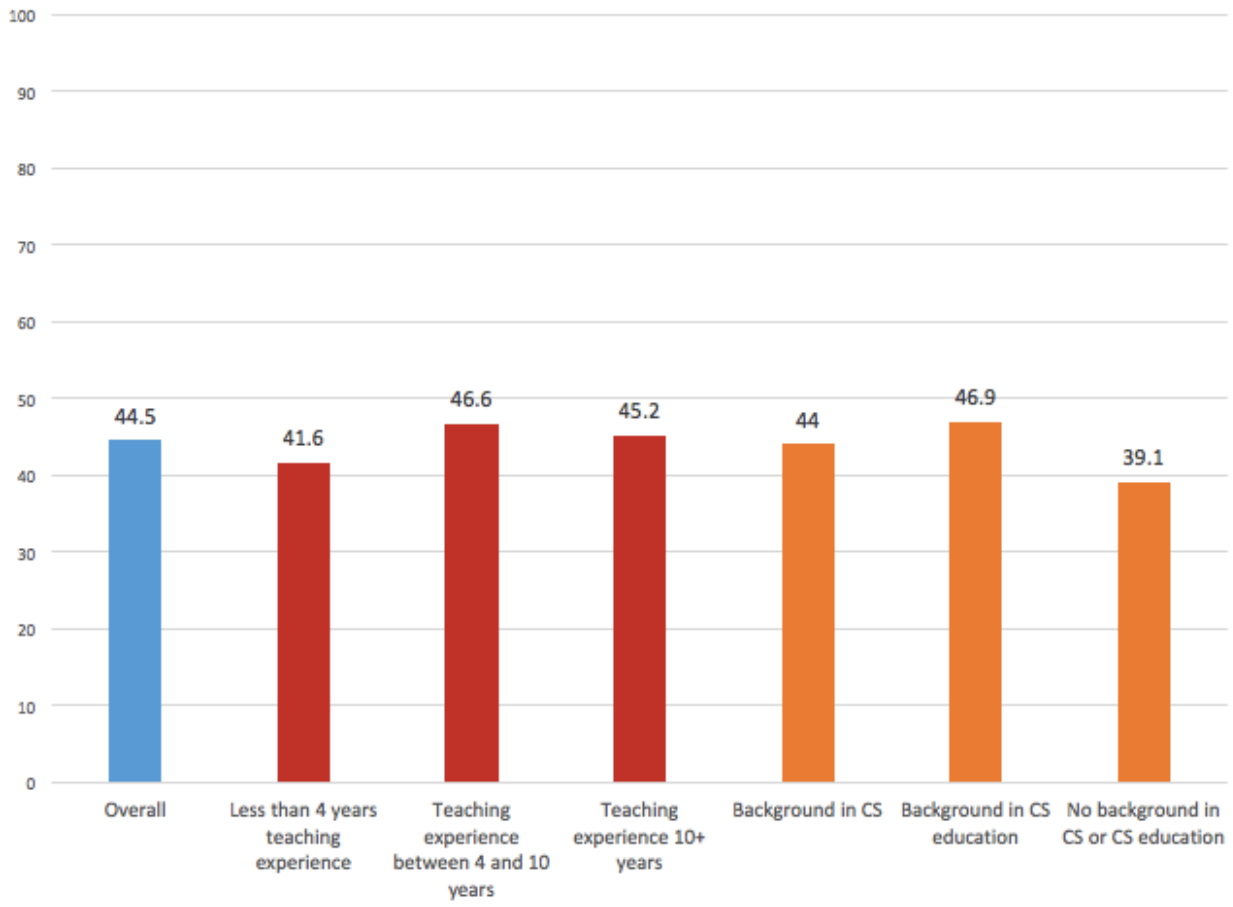

Figure 4.6.2. Frequency of teachers' perceived need for administrators to support their CS education efforts in the school/program

Need for Colleagues

Secondary CS teachers stated that they are the only teacher in most schools and need a community of CS teachers to collaborate, share ideas, and learn from each other.

Email listserv findings. Out of 72 teachers that shared stakeholders-related needs in the email listserv, 25 teachers stressed the need for collaboration with their CS teacher colleagues. Most of the listserv teachers mentioned being the only CS teacher in their school and expressed that they were looking for opportunities to collaborate and learn from each other: "Most of us are alone in our school, so do not have an opportunity for F2F collaboration" (E-7). Therefore, the CSTA email listserv served as an initial communication channel to further conversations with peers. Teachers expressed that they would like to be able to build lessons in small teams, get and give feedback among each other's work, and participate in online communities of CS teachers with related needs and contexts. Some teachers even expressed specific things they'd love to discuss with other CS teachers:

> I am working on the 3rd revision of a course (Introduction to Programming) that I've been teaching as a year-long game... I am seeking a small group of teachers who are interested in using this concept in their own classes, interested in providing regular feedback and possibly in collaborating on revising and expanding this game and course so it fully covers the level 2 CSTA standards. (E-72)

Many teachers had been trying to build opportunities to collaborate in their regions. One teacher shared an option in her region:

> If you are interested in meeting other Computer teachers in the X [region information hidden] area to collaborate, please respond to this post. Y [information hidden] College has offered a meeting space. (E-18)

However, due to distance and time limitations, meeting online in small groups was described as a convenient alternative. Many teachers discussed options:

- Great news, I also have a background in Mathematics, I look forward to an online discussion, in fact all the teachers in my department who are involved in writing CS content would love to be involved. We are back teaching as from the 6th January (E-73);
- We should create a sequence for k through 12 on CS. Let me compare yours with code academy and create a wiki for all of us can contribute. (E-32)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statement: I need access to an online community of CS teachers (e.g. collaboration, sharing ideas, receiving feedback). Overall, 52.0% of the questionnaire participants (n=221) identified an online community of CS teachers as a need or strong need. Twenty teachers in the questionnaire commented about colleague support and collaboration.

Almost all the teachers who commented about teacher collaboration mentioned being the only CS teacher in their schools and wanting collaboration with other teachers; however, most did not specify online spaces as an option. The following teacher defined the problem of having no one to discuss teaching practice with: "Typically each school has one computer science teacher, we used to meet 3 times year, then curriculum services stopped funding the release time. Long story shirt, I operate as a 'lone CS teacher" (Q-60). Another teacher shared the same concern: "I am the only CS teacher in my district. It is impossible for me to have informal and formal discussions with other teachers or administration to help develop my practice" (Q-40). Due to this isolation, some teachers were looking for face-to-face or online communication spaces. In the following example, one teacher was looking for face-to-face communication time with other teachers to share and discuss ideas:

> Because I am the only CS teacher at my high school I don't have anyone to share ideas or bounce ideas off of. I do use Edhesive's course material and that does include a forum for all teachers using the material. That I find to be helpful but it would be nice to have some face to face time. I'm currently trying to reach out to local colleges that have CS departments. (Q-61)

Other teachers were looking for online spaces. For example:

- An online community would certainly be helpful. There are some teachers through CHS who I communicate with for questions and to share ideas (Q-15);
- I am online with CSTA, but there must be more places I could look into. (Q-62)

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to the need for access to an online community of CS teachers. Four final-phase interviewees expressed access to a community of CS teachers as either a need or strong need in the questionnaire. When asked to explain these needs in more detail and provide examples, they specifically mentioned wanting to observe other teachers' successful practices and learn from them. They also stressed the importance of creating continuous and strong relationships with other CS teachers for successful collaboration.

In terms of learning from other teachers and observing their practices, one of the teachers proposed to share and watch each other's practices in a CS community:

> I think so, I'd love to see more about what, how other people are doing it, I'd love to shadow more other computer science teachers, or even have video of a classroom and say, "Hey" and just watch a video of someone teaching a unit and see how they're approaching it, what people are doing. (I-2)

Another teacher defined the need as learning from other teachers' successful practices, and then asked how this could be done easily:

> If there are teachers out there who have found really good ways of teaching computational thinking, that are working, that aren't public knowledge yet, I want to know that. If there was some way for a teacher to communicate, "Hey I've had tremendous success doing this style for this age group", I want to know that and maybe I can add it to my own collection of tricks that work. (I-1)

When the same teacher was asked what would be a place to communicate or collaborate, he suggested PD events and creating strong relationships with other CS teachers:

- Professional development events are a fantastic way to network and to meet and to stay in contact with other people who are doing the same kind of work that you are. For me, professional development is as valuable to building connections as it is for learning content (I-1);
- An online community also needs to be a little more personal so that you actually get to know people. (I-1)

Overall. Working as the only CS teacher in a school without support from colleagues was seen as a problem in all phases of this research. In this case, teachers were looking for face-to-face and online communication channels where they can learn from each other, observe successful practices, and share ideas and resources. Due to the convenience of online spaces, most of the collaboration suggestions involved online spaces such as social media groups and wiki spaces.

Years of experience and background in CS. Figure 4.6.3 shows the questionnaire findings for teachers with different years of experience and background. "Access to an online community of CS teachers" as an expressed need did not appear to be different between teachers with different years of experience. However, teachers with CS education background more likely to identify this as a need or a strong need compared to teachers with a background in CS and no background in CS or CS education.

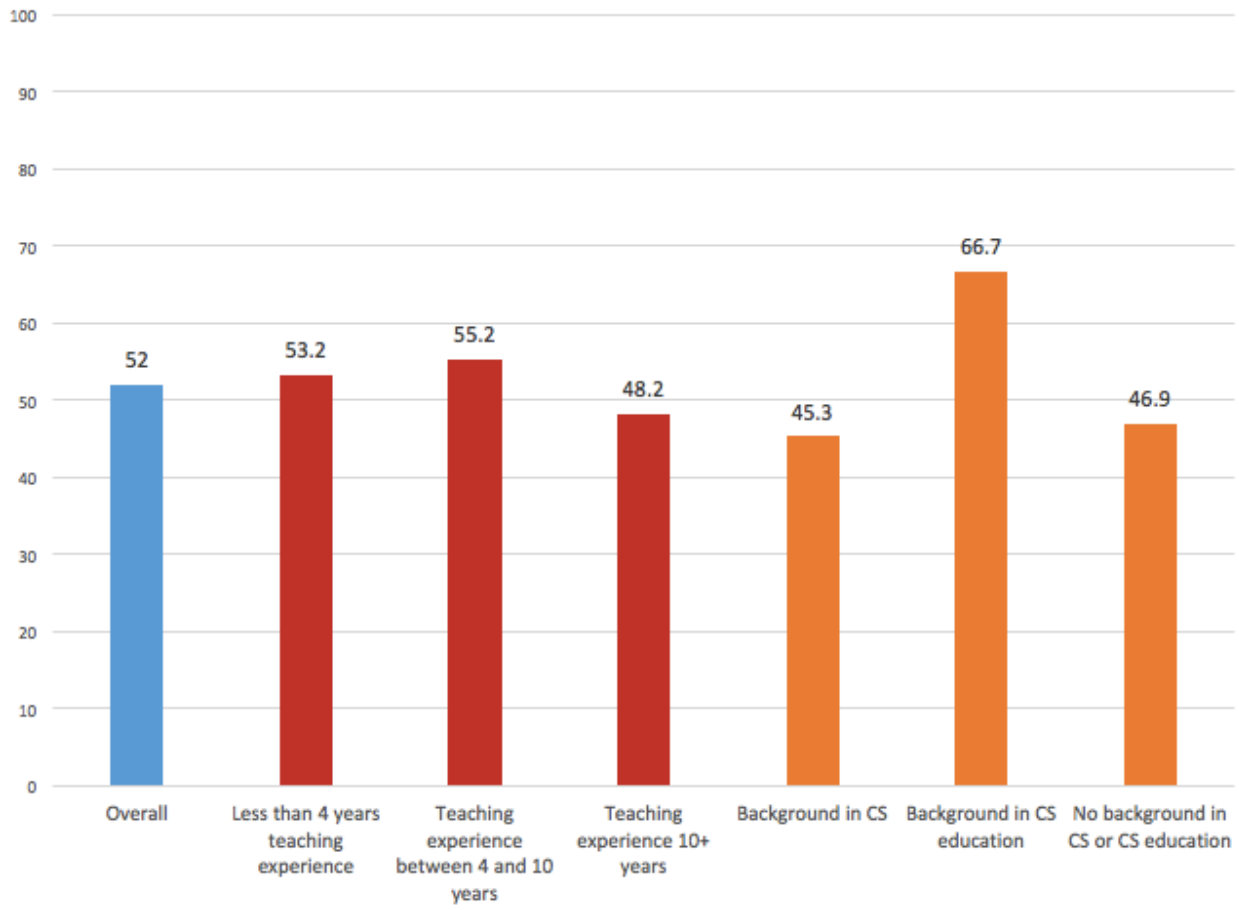

Figure 4.6.3. Frequency of teachers' perceived need to access to an online community of CS teachers.

Need for Parents

Secondary CS teachers stated that they need parents who understand what CS is and support CS learning activities in and out of schools.

Email listserv findings. Out of 72 teachers that shared stakeholders-related needs in the email listserv, seven teachers discussed parents' understanding of CS and involvement in CS activities in schools and after school programs. When they talked about important stakeholders in K–12 CS education, teachers listed parents along with administrators and students. In multiple emails, they shared their efforts to convince parents about the importance of CS. For example: "For someone who is also looking for ways to convince students, parents, and administrators that

computing is fun, creative, and collaborative and is where the jobs are, I am super excited about code.org's mission" (E-59).

Teachers also discussed that parents need to better understand CS. For instance, one teacher argued how introducing parents to CS as a core subject would help:

> It does not seem to me that we will be able to have CS as a credible core subject until we can present it to other folks just like we do math, science, reading, writing (i.e., when we can talk to a parent and say that their child will learn the concepts of X, Y, and Z with regards to CS in 3rd grade, 4th grade, 9th grade, etc.). The other core subjects can do this, which is probably why they are core subjects? (E-16)

When CS is offered only during out-of-school time (OST), teachers expressed the need for parents' involvement to a greater extent. One teacher felt OST CS clubs might influence parent's opinions and support: "I hope out of school CS experiences at all age levels result in demand from parents and community members for CS courses in schools" (E-74). Another teacher added their own ideas about this:

> You need parents who are dedicated to supporting such a program as they will be the ones transporting kids who stay after school (missing the bus). In a rural area where the school population is 67% minority and the majority of jobs are either farm related or oil field related, well - that kind of dedication is much harder to find. (E-49)

Questionnaire findings. In the phase 3 questionnaire, teachers were asked to reflect on the following statements:

1. I need my students' parents to better understand what CS education is;
2. I need my students' parents to get involved in CS education efforts in the school/program.

Overall, 52.7% of the questionnaire participants (n=220) identified parent understanding of CS as a need or strong need. Eleven teachers in the questionnaire commented about parents to better understand what CS education is. Overall, 43.8% of the questionnaire participants (n=219) identified parents' involvement in CS education efforts as a need or strong need. Five teachers in the questionnaire commented about parents' involvement.

The comments about parents primarily mentioned parents not understanding CS, and the benefits of CS as a subject. For instance, one teacher commented about parents' lack of knowledge or misinformation about future CS workforce needs: "I wish parents understood the statistics about future computer science needs in the workplace" (Q-63). Another teacher added: "My admins are supportive but too many parents don't understand or ask "What language are you teaching" (Q-64)? Therefore, some teachers commented about ways to help parents understand the field: "My administration and community of CS teachers is amazing! There is always room for improvement for educating parents and parent involvement (Q-33).

In the comments, teachers said that parents' limited knowledge and misinformation on CS negatively affected student decisions as well as teachers' practices. For instance, one teacher mentioned some students actually didn't want to take their course: "I am fortunate to have highest enrolment in cs in my region but most join due to parent's wish and with little understanding the scope of subject" (Q-58). Some teachers actually wanted less involvement from parents because of problems they caused in their classrooms:

> I down graded parent involvement only because I think they have too many opinions that come with too little knowledge and often ask for things like specific languages rather than trust the big picture knowledge of the faculty. Just as in any other department parents should value the decisions of the faculty, not push an agenda. I find too many parents misjudge student knowledge based on how much their children play with computers and what they do in one or two week summer classes. (Q-26)

Interview findings. Before conducting the interviews, the researcher considered the participants' questionnaire responses related to parents.

1. I need my students' parents to better understand what CS education is;
2. I need my students' parents to get involved in CS education efforts in the school/program.

Six final-phase interviewees expressed "*parents' understanding"* as either a need or strong need in the questionnaire, while five expressed "*parents' involvement"* in CS education

efforts as either a need or strong need. When asked to explain these needs in more detail and provide examples, the interviewees mentioned parents' limited CS knowledge. For instance, one teacher stated: "We have parents who still define Computer Science as keyboarding." (I-5). Another teacher shared that parents' limited CS knowledge influenced the decisions made by the districts' school board:

> I don't know that the parents in my community really understand what computer science is. I talked to a lot of people about like, "Oh, I teach computer science", and they think I'm teaching kids how to use Microsoft Word. Parents are ultimately who are responsible too, in the community they're the ones who are voting for the school board, school board provides people who are telling us what to do; because it's a smaller community we talk to parents quite a bit. (I-2)

Parents' limited CS knowledge seemed to affect their involvement in CS education activities in schools:

> Some parents have a sense of what is going on, but I think there's a lot of parents that weren't even around when [CS activities] were going through school. It's new unless they work in some type of technology field they might not even be familiar with it. (I-7)

Parent understanding and involvement seemed to be a bigger problem for schools in low-income and underrepresented populations, as described by this teacher:

> My school is about 99% Hispanic, 99% low income. Exposing them to computer science, this is a very new experience and in interacting with parents it's sometimes challenging to explain even what computer science is and all of the opportunities that come with it. We're trying our best to reach out to the community around [our students]. The middle school students don't really know what computer science is to want to come to our school. Or the parents don't really know what computer science is so they don't know how to interact with their students regarding the computer science class, even though they have a much better idea about each of the other classes. (I-8)

The same teacher mentioned parents' low academic background and limited access to the Internet as two reasons for these CS knowledge and involvement issues:

> A lot of the parents don't know how to navigate academics in general. Even though they probably have some experience with language, some experience with math, they have no frame of reference for computer science and talking about things like, when you don't have internet access at home, it would be really great if your student could have that, and

> they don't even really understand the significance of having internet access at home, not even for the computer science class but for the rest of their classes as well. (I-8)

Overall. Teachers viewed parents' knowledge about CS and involvement in CS activities as directly related to the success of CS education practices in schools. Some teachers said that parents' lack of understanding created environments where CS did not attain enough consideration. Furthermore, when a school has low-income and underrepresented populations, parents' education and involvement became a more serious problem. As a result, teachers in all phases of this research expressed a need for parent education in CS and the importance of CS education for their children's future.

Years of experience and background in CS. Figure 4.6.4 and figure 4.6.5 show the questionnaire findings for teachers with different years of experience and background for the following questionnaire items: 1) I need my students' parents to better understand what CS education is. 2) I need my students' parents to get involved in CS education efforts in the school/program.

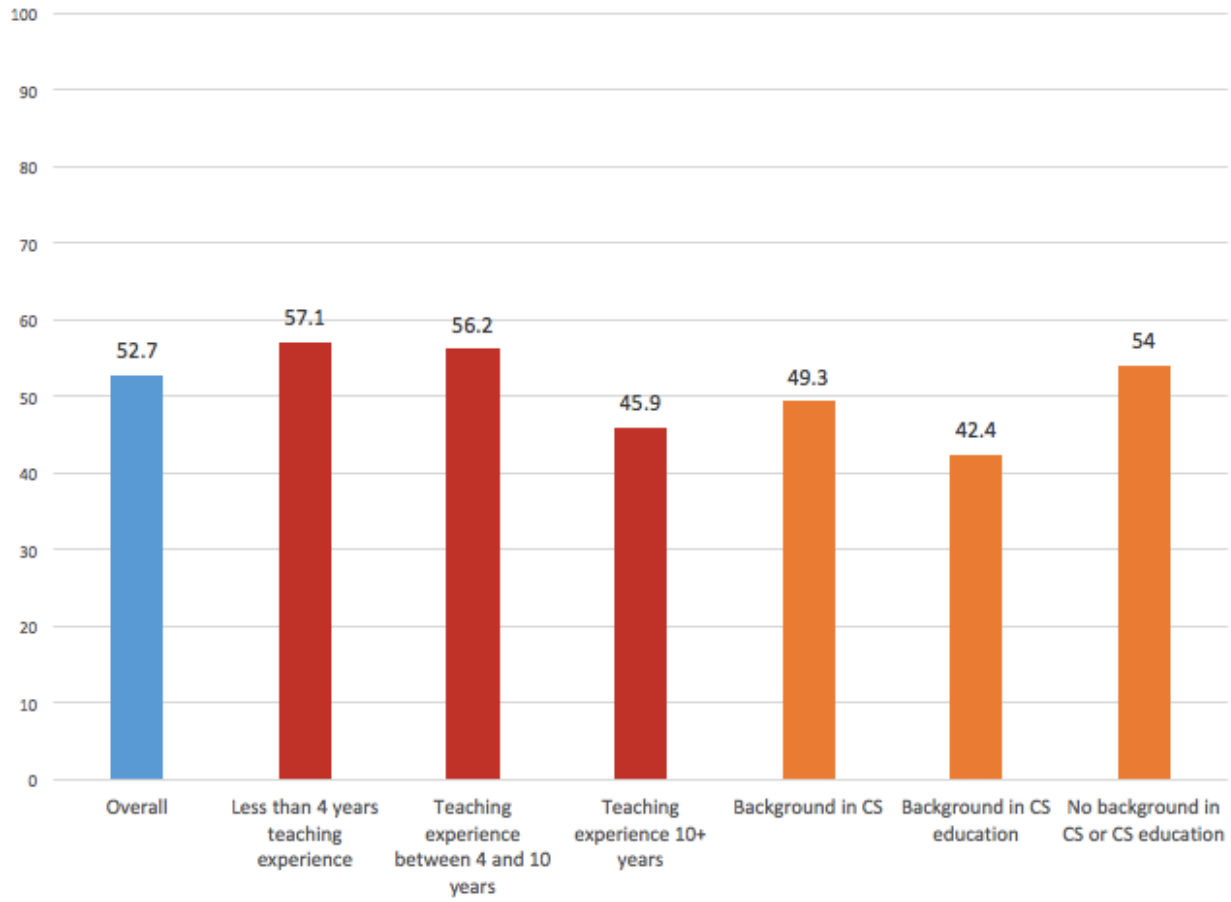

Figure 4.6.4. Frequency of teachers' perceived need for students' parents to better understand what CS education is

The needs "students' parents to better understand what CS education is" and "students' parents to get involved in CS education efforts in the school/program" did not appear to be different between teachers with background in CS. Only teachers with 4-10 years' experience more likely to (see figure 4.6.5 below) identify parent involvement as a need or strong need compared to teachers with less than 4 years' and more than 10 years' experience.

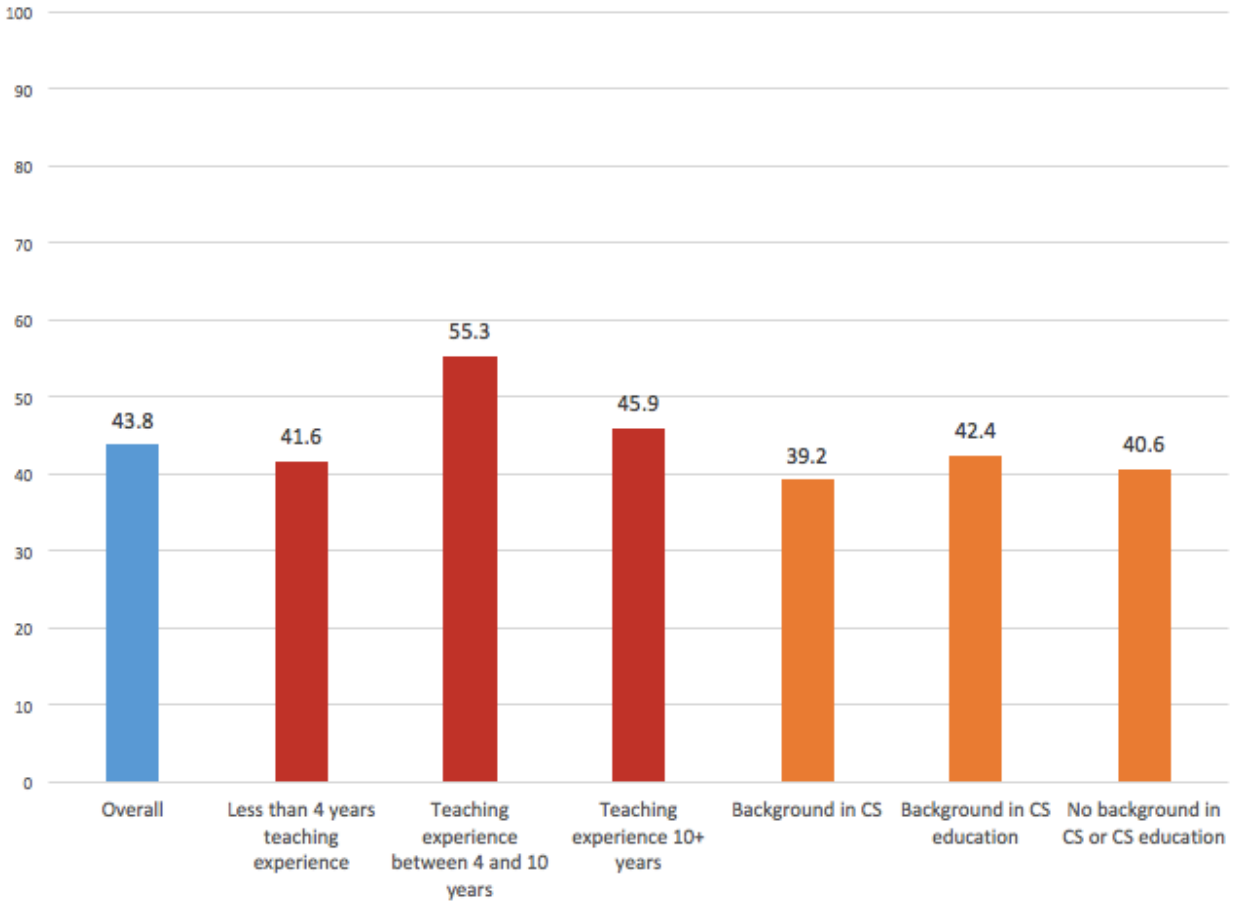

Figure 4.6.5. Frequency of teachers' perceived need for students' parents to get involved in CS education efforts in the school/program

CHAPTER V: DISCUSSION

The primary purpose of this study was to identify secondary CS teachers' needs, related to knowledge, skills, and school settings. Furthermore, the researcher examined how these needs change, based on the participants' years of teaching experience as well as their background in CS. Participants self-reported as secondary teachers teaching CS content in a school setting (public, private, or charter) or an after-school program at the time of the data collection. The data were collected from CSTA email listserv members' discussions (1,706 emails from 338 unique teachers), questionnaire responses from 222 secondary CS teachers, and interviews from eight secondary CS teachers. This chapter presents a discussion of the study results with implications for practice and research.

Research Question 1

What needs do U.S. secondary school computer science teachers share related to their knowledge, skills, and school settings?

The study found that the most common needs teachers perceived necessary for effective CS teaching were as follows:

1. Curricular needs. The teachers reported the need to update their course(s)' curriculum resources constantly to keep up with changes in CS education practices and national standards.
2. Recruiting more students to be interested in CS was a need for secondary CS teachers for both elective and required classes, in different ways. When CS was offered as an elective in their schools, teachers stressed the need for increasing student enrollment. On the other hand, when CS was a required course in schools, teachers reported a high number of students who showed little interest in learning CS. These teachers were looking for

strategies to increase students' interest and motivation. In addition, increasing female representation and minority populations in CS classes was also a need in both elective and required classes when these populations existed in the represented schools.

3. CS teachers expressed different pedagogical needs depending on their different contexts. The most common pedagogical need expressed was to learn student-centered strategies for teaching CS and how to guide students' understanding with scaffolding and team-management strategies.
4. CS teachers identified administrators and parents as the main stakeholders in CS education practices. Furthermore, teachers reported improving these stakeholders' understanding of CS to increase collaboration toward improving CS education practices in their schools. Many secondary CS teachers also shared the need for a community of CS teachers in their districts.
5. Even though resources for computer labs (equipment, software, and network) did not appear as a common need, a group of teachers did list resources as a crucial need for their situations.

Curricular Needs

Secondary CS teachers reported the need to constantly update their CS curriculum. The findings in this study suggested that curricular needs are one of secondary CS teachers' most common perceived needs; 132 teachers in the listserv identified constantly updating their curriculum as a need. Similar curricular needs were found in other fields in secondary education. In a study with 72 science and math teachers that surveyed their needs, curriculum was mentioned first, especially when teachers want to implement new instructional strategies in their classroom (Rogers et al., 2007). In the survey of the present study, secondary CS teachers

emphasized the need for curriculum and explained this need in more detail. One of their concerns associated with the curricular needs involved incorporating local stakeholders into their curricular decision-making and planning.

Local needs are important for curriculum design. Previous reports on CS education have highlighted national standards as the foundation for a comprehensive CS education curriculum for teachers (Seehorn, Stephenson, Pirmann, & Powers, 2013) to deliver the fundamental knowledge and skills of CS for all students at the K–12 level. However, some have argued that standards-based education may decontextualize curriculum and prohibit teachers from making curricular decisions because of its focus on predefined achievement goals and learning requirements (e.g., high-stakes testing) that are the same for all students (Vars, 2001; Wills & Sandholtz, 2009). Even though they did not emphasize it, CS education organizations changed from a focus on a standards-based approach to a place-based education approach in their recent initiatives. Place-based education means that the teachers can make curricular decisions based on their local community needs (G. A. Smith, 2002). This has been emphasized in educational policy studies (Guisbond & Neill, 2004). G. A. Smith (2002) stressed that place-based education targets the needs of a school's local environment, and teachers develop curriculums that allow students to identify and solve problems in their schools and communities. In meeting these curricular goals, students can have access to their community's resources, facilities, knowledge, skills, and experiences and meet needs within their own communities while at the same time improving their learning in different subject areas (Powers, 2004).

National CS education organizations similarly have argued the importance of considering local stakeholders' resources and needs when making curricular decisions in CS education. For instance, Google for Education representatives have advised CS teachers to collaborate with

local stakeholders when planning CS education programs and curriculums. College Board (2016) created a new AP CS Principles framework that allows CS teachers to select any programming language based on their context. Similar to College Board, a new K–12 CS education framework was developed by the Association for Computing Machinery, Code.org, the Computer Science Teachers Association, the Cyber Innovation Center, and the National Math and Science Initiative (K12CS, 2016). This framework aims to provide teachers with choices to make their own curricular decisions and stressed the need for districts and schools to develop their own standards and curriculums based on their specific local needs:

> The framework will identify key K–12 computer science concepts and practices we expect students exiting grades 2, 5, 8, and 12 to know. This effort will not develop educational standards. We expect that states and school districts will use the framework to create their own frameworks, guidance, and standards. (para. 3)

As emphasized in the new AP CS Principles and the new collaborative initiative, and based on the results from this study, former national standards for CS education did not fully represent and meet secondary CS teachers' curricular needs. The findings of this study suggest that CS teachers should consider local higher education institutions' expectations for prospective students, as well as business workforce needs, before designing CS curriculums. Targeting local community needs in curriculum development offers advantages for diverse populations (J. Wang, 2014). Initially, schools, teachers, and students may identify and define local needs and help address them in their communities (Mølstad, 2015), which may offer goals that students find meaningful and empowering in their own learning (Novak, 2002). Communicating with local higher education institutions may also facilitate students' transition to those institutions' programs. Previous CS education research studies suggested students' previous experiences as a crucial factor that influences students' success in introductory CS courses in higher education (Hagan & Markham, 2000). Furthermore, for students who do not go on to college, a CS

education curriculum targeting local business workforce needs may allow those students to find work in their own communities. Localization of CS education, thus, can help students go beyond learning CS content in a class toward success in their post-graduation endeavors.

Localization of curriculum has been emphasized in other education fields as well. Roberts and Suren (2010) stressed the importance of involving students in addressing local problems in their communities. The researchers conducted focus group discussions with 46 middle- and high-school students who participated a program that aims at addressing environmental problems in their community. The results of the study revealed that programs that include real-world, local problems enhanced secondary students' social and environmental awareness and improved their personal growth and leadership. In another study, Flowers and Chodkiewicz (2009) stressed the importance of creating relationships with local communities and schools to address issues of sustainability and climate change, which they argued created more authentic and transformative learning experiences for students in Australia. Dhorsan and Chachuaio (2008) conducted research with secondary students in a community school in Mozambique. Teachers in that study chose project topics (e.g., agriculture, fishing) that are relevant to students' daily lives. Dhorsan and Chachuaio argued that localization of curriculum provided students with the opportunity to serve their own communities and learn by doing in solving real-life problems.

Variety of curricular materials. The findings of this study also indicated that CS teachers had a constant need for varied curriculum materials in CS classes. This need may be due, in part, to the diversity of CS courses in K–12 education and changes in pedagogical approaches in teaching CS. For example, several teachers in the questionnaire had mentioned teaching programming, AP CS Principles, mobile application development, robotics, and web

design classes. Because the findings suggest that CS teachers typically teach more than one type of CS course, all of these courses may be taught by the same CS teachers. For example, in the most recent CSTA (2015) survey of 1,354 high school CS teachers, the participants reported teaching a variety of content in their introductory CS classes, including, but not limited to computer security, databases and information retrieval, ethical and social issues, game programming, graphics, hardware, networks, problem solving, programming, and web development. Since CS teachers typically have more than one CS course, each CS course reported requires the development of a different curriculum with different content and resources. For example, in the findings, teachers reported teaching programming classes with different languages, using different kits in robotics classes, and different programming platforms for mobile application development classes. In addition to these content and resource differences, CS teachers constantly need to update their curriculums based on their students' backgrounds, preconceptions, and learning needs (Goode & Margolis, 2011) and provide positive learning experiences for all students (Brown, 2003).

Other fields have indicated a need for additional curricular materials due to the constant evolution of pedagogy. For example, one of the more recent evolutions in pedagogy has focused on student-centered learning. In core subject areas (e.g., math, science), previous research reported the need for new curriculum materials for differentiated instruction that supported diverse students' needs in the classroom (e.g., Zakaria & Daud, 2009). Woodward and Brown (2006) conducted a study with 53 middle-school students with special needs. The researchers found that curriculum specifically designed for students with special needs led to better outcomes and attitudes in learning math. Furthermore, innovations in educational technology and integrating technology in the classroom has also led to the development of new curricular

materials for teachers in schools (Ertmer & Ottenbreit-Leftwich, 2010). Teachers use technology to support or enrich their existing curriculums, which require new curricular materials designed using new technology tools (Ertmer et al., 2012). Therefore, it would stand to reason that computer science pedagogy would also face similar pedagogical and technological evolution (as showcased in this study). These teachers described the importance of being able to implement student-centered pedagogical materials.

In addition to these material needs, one of the most common needs discussed in the email listserv was materials for assessment. One of the challenges teachers mentioned is finding valid and reliable assessment instruments to evaluate students' learning in different CS content (Moskal et al., 2006). However, due to limited funding in schools, the findings suggest that teachers are often forced to develop their own assessment instruments. Previous research studies have described CS teachers' challenges for designing assessment materials in their CS courses (e.g., Yadav et al., 2015). For example, Franklin et al. (2013) examined pair programming in a CS summer camp and reported the challenge of developing assessment, as they had difficulties gathering data from individual students in the pairs. Brennan and Resnick (2012) tested three different assessment approaches to evaluate the learning of students between the ages of 8 and 16 in programming using the Scratch platform, and then highlighted challenges for developing efficient and effective assessment strategies for CS classes. Those three approaches included: 1) project portfolio analysis, 2) artifact-based interviews, and 3) design scenarios. Project portfolio analysis focused on students' products in their Scratch accounts but was limited in terms of examining students' conceptual understanding in their learning processes. Artifact-based interviews examined students' learning processes and provided details about their conceptual understandings, but this method was found to be time-consuming 3) Design scenarios

assessments required students to explain what their completed Scratch projects did, identify any code errors (debugging), and then to improve the project with extra features. Even though this approach was also helpful in examining students' learning process through explanations, critiquing, extending, and debugging, it was also time-consuming.

In other subject areas, previous research reported the need for developing effective assessment practices, which should help teachers identify students' learning difficulties (Zakaria & Daud, 2009). However, English (2016) argued that due to teachers' lack of access and knowledge to create student-centered assessment materials, math teachers prefer standardized tests and have difficulty reliably measuring students' mathematical capabilities. English recommended using formative assessment techniques, which emphasize testing general knowledge as well as individual differences.

Computational thinking in curriculum. Embedding computational thinking in curriculum was one of the major teacher needs expressed in the study. This need emerged from secondary CS teachers' email listserv discussions about selecting a programming language for their CS courses. A group of teachers (n=9) in the email listserv explicitly shared concerns about selecting a programming language. Selecting a programming language has often been a discussion between CS teachers (Pears et al., 2007). Pears and his colleagues provided teachers with guidance on choosing the right programming language by considering curricular, pedagogical, language, and tools differences relevant to teaching introductory programming courses. In the present study, CS teachers reported that, before they can make a decision, they consider a language's ease of use, appropriateness for students' background, and usefulness after students graduate. In that regard, the email listserv conversations mirrored research literature about teachers' concerns in selecting programming languages. However, it was surprising to see

that only a small group of teachers in the later phases (questionnaire and interviews) expressed that choosing a programming tool or language was important to their teaching. This lack of concern for choosing a programming language may be due to current CS education and the AP CS Principles frameworks' lack of emphasis on any specific programming language. Current practices recommend using programming language as a tool but emphasize that computational thinking is the main goal of CS education (Lye & Koh, 2014). However, while embedding computational thinking in curriculum was mentioned often, the findings of this study suggest that secondary CS teachers have limited knowledge on the definition and principles of computational thinking. This lack of knowledge becomes a barrier for them to effectively embed computational thinking in their curriculums. This has been emphasized in the CS research literature. Barr and Stephenson (2011) stressed that embedding computational thinking in curriculum requires robust understanding of the concept and developing age appropriate practices for target student populations.

Using computational thinking in other subject areas has been discussed in the literature as well. Barr et al. (2011) emphasized computational thinking is a skill necessary for all students in the digital age and provided examples of how to use it in arts, social studies, and music classes. However, previous research reported that teachers in other subject area experienced difficulty understanding the concept and applying it in their teaching. In a study with 12 pre-service teachers developing lesson plans and curricular materials for elementary students, Sadık, Ottenbreit-Leftwich, and Nadiruzzaman (2017) identified pre-service teachers' misconceptions about computational thinking, which include incomplete concept definition and incorrect application of it to curricular activities. Kotsopoulos et al. (2017) researched this difficulty and developed a framework for embedding the principles and components of computational thinking

into a curriculum for student-centered learning. Kotsopoulos et al. suggested the use of non-computer activities that allow students to modify, construct, and remix different objects for a variety of learning goals. In both CS learning and other fields, successful implementation of computational thinking depends on clearly defining what computational thinking is and showing connections between its implementations in different fields.

Implications for research and practice regarding curricular needs. To assist with CS teachers' curricular needs, I suggest the following: (1) designing CS education curriculums that incorporates local community's needs and expectations, (2) developing a repository of curricular resources, (3) offering CS education PD programs to improve CS teachers computational thinking knowledge, and developing assessment materials for student-centered learning environments.

Designing CS education curriculums based on local organizations' needs may provide important benefits for improving K–12 CS education in different contexts. Curriculum decisions should be the result of communicating and collaborating with local stakeholders, including local businesses, higher education institutions, districts, schools, students, parents and the teachers themselves (K12CS, 2016). For instance, it may not make a big difference which language is offered in a school. However, when deciding which programming language to use in the curriculum, it may be helpful to communicate and consider local businesses' practices and needs in CS. Teacher education institutions and professional development (PD) programs need to build relationships with local stakeholders when they make decisions and train pre-service or in-service teachers. It seems especially crucial to include successful business practices in planning and delivering PD programs for secondary CS teachers.

Due to their extra preparation and teaching duties in multiple topics, CS teachers may not have time to develop all their own curriculum resources. Time was one of the most frequently expressed needs in this study as well as in previous national reports (CSTA, 2015). In order to meet the need for curricular materials and ideas, an online repository of curriculum resources aligned with national and state standards should be established. Although CS teachers reported finding some online resources, the main issue was the lack of quality lesson plans, resources, and ideas directly applicable to their schools and classroom environments. This may be why teachers in this study were looking for fully designed lessons recommended by other teachers who had implemented them. The need for obtaining quality and structured resources may have also been the reason why CS teachers also frequently requested textbook materials on the email listserv. If an online curriculum repository could be created, it may be helpful to add an evaluation system so that teachers can rate the resources. Thus, teachers could filter and access highly rated materials as well as upload their own created materials. Curriculum resources could be tagged and filtered by teacher users with course titles, content, description, or CS education standards. Additional filters might also increase the usefulness of such a repository, such as by instructional approach (e.g., problem-based learning, pair programming) and location (e.g., state, district). Finally, the repository may include an area for teachers' comments.

Online curriculum repositories have become a popular solution to teachers' problems associated with finding structured and quality curricular resources aligned with their contextual and instructional needs (Ackerman, 2015). The Open Educational Resource (OER) websites provide examples of online educational resources for teachers and students (Hylén, 2006), such as content, software tools, and pedagogical resources. American Modeling Teachers Association (AMTA) is another example of curricular repository as well as a collaboration space for teachers

to access exemplary curriculum resources and successful teachers' practices in STEM areas (American Modeling Teachers Association, n.d.). In AMTA, teachers can access full curriculum packages in different STEM subject areas that are aligned with the Next Generation Science Standards (2013). However, curriculums aligned with standards are limited to specific standards and do not meet broader teacher populations' needs with differing teaching content and student ability levels. Therefore, creating a repository of curricular resources with the aforementioned features may be a valuable asset to improve CS teachers' practices.

To return to the teachers' expressed need for assessment ideas, it is important to access or develop assessment materials to efficiently evaluate students' learning and guide students for their own learning. The suggested curriculum repository may be a good place for sharing and accessing assessment ideas in different CS topics. Learning CS involves attaining higher order thinking skills and requires continuous assessment of students' learning progress by using formative assessment techniques (Luxton-Reilly & Denny, 2010). The researcher suggests using formative assessment and rubrics in CS classes as a form of assessment guide in programming projects aligned with previous research (e.g., Ahoniemi & Karavirta, 2009; Basawapatna, Repenning, & Koh, 2015). Furthermore, instructional rubrics have been suggested as a method to facilitate teachers' grading and feedback processes, and assess students' higher-order learning outcomes (Becker, 2003). Instructional rubrics include criteria for successful projects with different levels of quality measures (H. G. Andrade, 2000). For instance, H. L. Andrade, Du, and X. Wang (2008) found that providing criteria and using rubrics for students' self-assessment led to more productive outcomes for students' projects. Even though previous research literature have asserted the advantages of using rubrics, CS teachers reported not using them frequently (Becker, 2003). This could be due to their limited time, or knowledge and experience building

rubrics for CS assessment tasks. PD programs may focus on preparing teachers for effective rubric development for evaluation and learning purposes in CS classes, especially in programming, and serve as collaboration spaces where they develop rubrics with their colleagues.

Regarding the need for embedding computational thinking in curriculum, recent studies emphasize the need for educating CS teachers on the concept and components of computational thinking (Hodhod, Khan, Kurt-Peker, & Ray, 2016). However, due to the difficulty of transferring research knowledge to practicing teachers (Kincheloe, 2012), this research suggests developing practical strategies and resources for teachers that define computational thinking and its components with real-life examples for teachers in different levels of schools (middle and high school) and student populations (based on students' preconceptions). Barr and Stephenson (2011) provided a definition of computational thinking with resources. In the Barr and Stephenson study, teachers, members from national CS education organizations, and experts from both academia and business came together to create a definition of computational thinking and developed examples and strategies to incorporate it in five subject areas. By the end of a two-day meeting, the committee had identified core computational thinking concepts and targeted student capabilities in CS, math, science, social studies, and language arts. The results of this meeting are important to provide teachers both the definition from different areas and clarify what is expected from the students in learning and applying computational thinking skills.

Both the quantitative and qualitative findings suggest attending PD events as the biggest need for secondary CS teachers in updating and improving their CS education practices. Thus, CS teachers may improve their CS content knowledge in computational thinking, find or design materials for assessment and computational thinking in collaboration with other teachers, and

embed new materials in their curriculums. However, PD events should be convenient in location, time, and funding, as well as directly applicable to teachers' needs.

Increasing Student Enrollment and Interest

The findings suggest that low student enrollment is a serious problem for the sustainability of K–12 CS education. CS is typically offered as an elective in most schools (if at all), and the CS teachers in this study described being apprehensive about losing their jobs due to low enrollment in their classes. CS teachers are the main advocates of CS education in their schools, and the researcher argues that it is critical for all CS teachers to learn and apply appropriate strategies to increase the number of students in their classes, and maintain CS classes in secondary schools and keep their jobs.

Students' knowledge and beliefs for increasing enrollment. CS teachers described that students' limited knowledge and negative beliefs about CS were the two primary reasons for low student enrollment. Teachers reported that students in K–12 do not perceive a connection to either their personal lives or professional futures; thus, they do not value learning CS. If we want to increase students' positive beliefs regarding CS, we need to find ways to make CS relevant to students. Perhaps if students have stronger conceptual understandings of what CS means, they may be more likely to see the relevance to their own lives and become more interested in learning CS.

Previous research emphasized that students do not understand the underlying elements of CS and how they apply to their everyday lives. Code.org's (2013) Hour of Code events have been successful national initiatives that provide K–12 students with simple but effective ways to explore computer science. Over the 3 years with code.org, 195 million students from 250,000 classes were involved in code.org project by the end of 2015. Through the code.org's learning

platform Code Studio, teachers are able to use predesigned lesson plans for teaching programming concepts to students. The lessons focus on fundamental concepts of programming such as sequencing, debugging, conditionals, nested loops, event-handlers, and functions, and students demonstrate their understanding and skills solving programming challenges. In their Hour of Code activities, code.org aims to break students' biases about CS as a difficult subject and requires advanced math, and show CS and programming is fun and anybody can do it. There have been increasing number of research studies conducted about code.org's initiatives and their influences. Umbleja (2016) stressed that code.org activities increase students' understanding and interest in CS if they are provided the concepts and skills in primary education or younger secondary level. Umbleja emphasized that, as they get older to teenager years, it may be difficult to change their biases about CS. In a study with 32 primary school students, Kalelioğlu (2015) highlighted that code.org activities provide basic concepts in programming and motivate students early in education to pursue their learning in the field.

In addition to providing them to early exposure to CS, it is also important to connect CS courses to students' interests and needs in schools. For instance, secondary CS teachers commented that even the name of their courses influenced students' perceptions and impacted the enrollment rates in their classes. The findings suggest that using technical terms (e.g., *object-oriented programming*) in a course name may have led to negative student perceptions and they may be hesitated to enroll to the CS course. On the other hand, CS teachers suggested that course names related to students' interests (e.g. game programming, media computation) may have helped increase the number of students in their CS classes.

Research literature contains extensive examples of student perception studies in K–12 math education. Meece, Wigfield, and Eccles (1990) found that students' perceptions about their

math capability, performance expectations, and value of math in their lives predicted their enrollment numbers in elective courses. Ramirez, Gunderson, Levine, and Beilock (2013) conducted a study with 154 first- and second-grade students about their math anxiety, arguing that early detection of students' anxiety problems about a subject area is crucial because it otherwise may cause a snowball effect and discourage students from enrolling in those courses or to pursue related careers in the future. Mallow et al. (2010) conducted a comparative study of American and Danish secondary students' science anxiety and concluded that students in both countries carry their science anxiety to higher education. This continued science aversion impacts the gender gap in STEM careers, which is discussed further in the following section.

Increasing underrepresented populations in CS classes. CS teachers identified that they needed strategies to motivate underrepresented populations (e.g., female and minority) to enroll in their CS classes (where these populations existed). This ongoing need can be seen in current statistics about underrepresentation of female and minority students in U.S. secondary CS classes (Exploring Computer Science, 2016).

Increasing female representation especially was a common need identified in the study. Teachers in this study reported that the main cause of female underrepresentation was topics chosen for CS classes in secondary schools. It is worth noting that several CS teachers shared success stories to increase female enrollment by using specific programming tools (e.g., Scratch) and approaches (e.g., creative expression), as well as relating the descriptions and focus of courses to align with female interests. Similar characteristics were employed in Guzdial (2013)'s media computation course to engage female students. After 10 years testing different curriculums and contextual variables in the media computation course, Guzdial used media, story-telling, and robotics content and resources to increase female students interest in learning

CS and retained average 40% of female students in their CS courses, where females succeeded better than male students. Guzdial embedded CS concepts in multimedia development and production activities, including graphics, video design, audio design, and production, which all require and emphasize creative expression. Hsu (2013) also emphasized female students' interest in creativity and embedding multimedia in programming similar to Guzdial. Offering alternative paths to CS education has potential to increase diversity in these learning initiatives (Rusk, Resnick, Berg, & Pezalla-Granlund, 2008).

Students' self-efficacy was reported as one of most important predictors of learning in CS classes (Bergin & Reilly, 2006; Ramalingam, LaBelle, & Wiedenbeck, 2004; B. C. Wilson & Shrock, 2001). Previous research studies reported societal influence on underrepresented students' self-efficacy about CS. A national Google and Gallup (2015) companys' report explained that one reason could be due to students' and parents' observations of TV and media that often present White or Asian men engaged in CS but few women and minority populations. Adults' negative stereotypes also shape female students' self-efficacy about CS. Umbleja (2016) reported female students' high self-efficacy and positive beliefs in learning CS in primary education but lower self-efficacy in secondary level. Umbleja explained this with teachers' stereotypes about gender (e.g., CS is for male students) and their negative influence on children. Even though African American and Hispanic students report similar and in some cases higher interest and motivation than White students in any school level to pursue a CS career (Lichtenberger & George-Jackson, 2012), they are not well represented in higher education CS programs (Exploring Computer Science, 2016). The findings of this study suggest families' economic status and access to technology resources (computers, Internet, income, time) at home as the barriers to minority students' (e.g., African American and Hispanic) access to CS

education in secondary level. Ma (2009) reported similar factors influential minority students' collage major choices.

This lack of female and minority representation in STEM areas has also been observed in the literature. National new and reports reported the underrepresentation of female students in STEM and explained how that impacts U.S. workforce's innovation capacity (e.g. Camera, 2015; NSF, 2015; Beede et al., 2011). Beede et al. (2011) stated that lack of role models, gender stereotyping, and negative influences from families were the reasons for female students' underrepresentation in science. In a study to understand college students' enrollment decisions and career choices, Le, Robbins, and Westrick (2014) found gender was an influential factor. In other words, if a student was female, they were less likely to enroll in STEM courses and/or select STEM careers. Le et al. stressed the importance of increasing female students' interest and participation to STEM areas early in their schools by seeing role models in school and family.

Making CS a required course to increase student enrollment. Secondary CS teachers in the study shared and discussed solutions to the low enrollment issue. Some of them suggested that CS courses be required for all K–12 students. Due to the expected need for increased number of employees in CS related fields (Lockard & Wolf, 2012), federal government made similar recommendations. The new "Computer Science for All" initiative from the President's office emphasized CS education for all students from kindergarten through high school in the U.S. (M. Smith, 2016). However, researchers working in CS education shared significant concerns. Guzdial (2014) stressed lack of qualified teachers to teach CS courses in schools and highlighted that only 1 in 10 schools has a CS teacher. Guzdial also shared concerns about teachers' limited content knowledge in CS and lack of a successful curriculum, which are also suggested as needs in this research.

The findings in this research reported schools that already required CS classes. However, the teachers in those schools reported that this left them with students who lacked interest in learning CS. Student interest and motivation are necessary conditions for learning any topic even with desired instructional conditions (Krapp, 1999, 2005). With the goal *CS for All* students (M. Smith, 2016), educators need to make more effort in meeting the requirements for this systemic change. The framework developed and the findings in this study may initiate a discussion regarding the need for systemic change and guide further research in this area.

Math and reading comprehension as learning barriers. Teachers in this study discussed another drawback of requiring CS courses: students' limited math and reading abilities. The findings suggest that when students have low abilities in Math and English, they struggle more with learning CS. For example, CS teachers reported that some students were unable to do simple calculations and read instructions to complete simple programming tasks. This finding aligns with previous research regarding the importance of math background in learning CS especially programming. In a study with 123 first-year introductory programming course at a higher education institution, Bergin and Reilly (2006) found that mathematics ability is one of the important predictors of students' success in programming classes. Grover, Pea, and Cooper (2016) reported correlation between middle school students' prior English and Math abilities for learning computer programming. The findings suggest that low abilities in Math and English created challenging teaching environments for CS teachers trying to manage and teach students with widely varying needs and learning goals.

Implications for research and practice regarding student enrollment and interest. To assist with CS teachers needs regarding increasing student enrollment and interest, the researcher suggests the following: (1) introducing CS to students early in their education through

short-term programs and local campaigns (2) developing curriculums that represent diverse student interests (3) defining short-term benefits and learning outcomes for increasing students' interest and learning in CS classes (4) developing strategies to teach students with different preconceptions.

To combat the problems with enrollment and diversity in CS education, teachers should be careful not to impose gender biases (e.g., CS is a male field) to their students. Furthermore, students should be introduced to CS earlier in their education, perhaps through short-term CS activities in primary and early secondary education levels (Umbleja, 2016). This will combat the problems by reducing assumptions in early ages about CS as a difficult and advanced math field, and gender biases about CS as a men field (C. Wilson, 2014). We need to continue exploring ways to introduce CS to larger populations with different social and economic resources. For example, some research studies have suggested the potential of local campaigns developed by public libraries, school districts, and higher education institutions to extend the national efforts to larger populations (Moorefield-Lang, 2014). Offering CS to all students require long-term planning, and programs in local institutions can quickly serve as convenient places to introduce students CS knowledge and skills. Students from various age groups can work in collaboration to develop CS projects and solve real-life problems (Moorefield-Lang, 2014). These local institutions can also provide access to their own resources for students who do not have access to technology at home.

It is also important to have a more diverse student population, which would lead to a more diverse workforce in CS. This idea was supported in previous research about meeting the U.S. population's needs by involving diverse workforce in the production of ideas and tools (Partovi, 2015; Prey & Weaver, 2013). In schools where CS classes are offered, teachers need to

offer curriculums and programs that are related to diverse students' interest (Papastergiou, 2008). For instance, CS teachers may rethink about defining their courses' titles and descriptions. A simple change like this may be an effective strategy to develop positive student impressions by connecting the course descriptions and content to their interests. Even though there are promising national efforts, the findings and federal reports suggest that increasing diversity in CS is a continuing need (NSF, 2015) and the researcher advocates exploring ways to transfer the strategies developed in national efforts to local districts. The National Center for Women and Information Technology's (NCWIT, 2016) Pacesetters program have been providing successful strategies to increase female representation in IT related fields. Another effective way to introduce female role models and career paths for prospective female students is the attendance of conferences related to women in IT (Alvarado & Judson, 2014), such as Anita Borg Institute's The Grace Hopper Celebration of Women in Computing Annual Meeting. Increasing the number of female and minority teachers in schools may be helpful to serve as role models and influence underrepresented students' perceptions about CS (Goode & Margolis, 2011). Teacher education programs may encourage female and minority pre-service teachers to teach CS and increase the number of these role models in secondary schools (Sadık, 2015).

Increasing students' interest in and motivation to learn CS is another important need expressed by secondary CS teachers in this study. Short-term programs and national promotions (e.g., code.org) have been reported as helpful in gaining interest in CS (C. Wilson, 2014); however, keeping that interest in the classroom may be challenging. The findings suggest that teachers need to define goals and benefits for students to retain their students' interests. Goode and Margolis (2011) highlighted the college-preparatory status of APCS courses as an influential factor increasing students' interest and motivation in learning CS in high school level. In short-

term, providing students with opportunities to develop products that they may use in their daily lives and even sell their own products in the market may be helpful strategies for sustaining interest. Since there are no testing requirements in CS education, a place-based education approach may be a successful approach to define goals for students to serve their local communities through service learning projects. Service learning projects in CS higher education provide motivational benefits for learning (Sandeson, 2003) while creating an engaging and motivating learning environment in K–12 (Billig, 2000); however, K–12 CS education literature appears to be limited in this regard. Service learning approached should be promoted and further research should explore effective strategies to employ service learning in K–12 CS education.

Regarding students' math and reading comprehension, secondary CS teachers were looking for strategies to exclude students with problems in those areas before they register for CS classes. However, this goes against the national goal known as "CS for All" (M. Smith, 2016), and would create bigger problems in the future. In fact, several teachers in the study mentioned that "CS for All" may create classes that are difficult to manage. Teachers need strategies to help them deal with the wide range in math and reading skills among their students. Further research may be helpful to develop student-centered practices for teaching students with different math and reading skills in secondary CS classes.

Pedagogical Needs

The findings suggest that a majority of secondary CS teachers in this research need to learn student-centered strategies for their classrooms. Similar results have been found in other studies about teachers' pedagogical needs. Zhang, Parker, Koehler, and Eberhardt (2015) conducted a study with 118 science teachers and emphasized the importance of understanding science teachers' needs in a constantly changing educational contexts and reforms (e.g.

pedagogical evolution, standards, curricular changes, students' needs). The researchers concluded that science teachers need improvement in multiple areas of pedagogical content knowledge, such as instructional strategies in student-centered teaching strategies.

In the present study, the need for learning student-centered strategies also emerged from the teachers' discussions and responses when they talked about their needs for PD. Specifically, secondary CS teachers stressed the need for two learning strategies in CS: problem-based learning (PBL) and pair programming. The sections below discuss these pedagogical needs in detail.

Learning and applying scaffolding strategies in CS classes. Instructional scaffolding involves dividing learning into chunks and providing temporary supports to assist students in accomplishing new tasks and concepts. As students learn, the supports are gradually removed. Learning and applying scaffolding strategies emerged as an important pedagogical need for secondary CS teachers to support students' learning in PBL-driven learning environments. PBL involves strong teacher involvement and guidance through scaffolding (Hmelo-Silver, 2004). Scaffolding helps students to complete complex learning tasks (Belland, 2014) that are difficult to achieve without assistance (Wood, Bruner, & Ross, 1976). Scaffolding is primarily important in supporting students to reflect on their own learning and develop higher order thinking skills (Azevedo, Cromley, Winters, Moos, & Greene, 2005; Davis & Miyake, 2004; Raes, Schellens, De Wever, & Vanderhoven, 2012; Saye & Brush, 2002).

Effective scaffolding requires strategies that most CS teachers shared as needs in this study. Brush and Saye (2000) conducted a case study in a social studies class with 21 students and one instructor, who described herself as a traditional teacher and mentioned that she had little experience with student-centered learning strategies. The researchers defined this teacher's

challenges as (1) understanding her role as a facilitator, (2) difficulty managing groups, and (3) difficulty with student accountability and feedback. In the present study, secondary CS teachers identified similar issues with their instruction when they teach computational thinking.

The teachers in this study stressed that when students are assigned a computational problem, they often use trial-and-error when they are not given adequate scaffolding. Trial-and-error is not an efficient problem-solving strategy (Sengupta et al., 2013)— it takes time and often yields no results. The findings suggest a more purposeful design of computational problem instruction, with teachers scaffolding embedded in the process. Purposeful design involves designing models to solve computational problems, which includes "modeling, abstraction, and automation" (Isbell et al., 2010, p. 201). Teachers are the main facilitators of this process and the teachers in this study shared a lack of understanding of the facilitator role. They need to learn scaffolding strategies that can guide students' problem-solving processes.

Teachers' scaffolding also plays a significant role in supporting students' coding practices. The findings indicate that secondary CS teachers want to learn more strategies to answer students' questions during coding practices, particularly when students debugging their own codes, finding and fixing errors, and increasing the efficiency of their codes. These are important components of computational thinking processes in programming activities (Grover & Pea, 2013) that students need most guidance. Students tend to expect teachers to point out their mistakes, but this is not an effective teaching method. Providing feedback in the form of guiding questions helps students to assess and reflect on their own learning (Nicol & Macfarlane-Dick, 2006).

Examples of scaffolding can be found in different subject areas. Anghileri (2006) stressed that math teachers are more effective when they use diverse teaching strategies and support

students' learning with scaffolding. Even though described as very important, Smit (2013) argued that scaffolding has been used in only very superficial ways in classrooms. Therefore, Smit recommended the concept of whole-class scaffolding for multilingual mathematics classrooms. Smit listed the characteristics of the whole-class scaffolding as diagnosis, responsiveness, and handover to independence. The way scaffolding is used has been changing with technological innovations and new approaches in instructional strategies. Sharma (2007) stressed that teachers need to develop and use scaffolds with the help of technology that allows them to access additional resources based on their learning and contextual needs.

Creating a collaborative teamwork environment. Teamwork is a crucial part of CS learning and benefits students' learning experience in sharing information and receiving feedback within a social community of peers (Sancho-Thomas, Fuentes-Fernández, & Fernández-Manjón, 2009). In PBL and pair programming, students work collaboratively in teams. PBL learning involves more teacher guidance and scaffolding, while pair programming involves strong pair collaboration within teams, with less teacher involvement (Nagappan et al., 2003). It is not surprising that teachers who requested feedback in the email listserv about pair programming were looking for team building and management skills to create successful collaborative environments (Cockburn & Williams, 2000). The findings indicate that CS teachers want to make sure that all their students actively participate to team work. Poor teams often involve one "expert" student taking all the responsibility, while other members become passive participants who may not benefit from the learning opportunities (Shimazoe & Aldrich, 2010). Teachers in the study also solicited strategies for creating a collaborative learning environment where students search for answers from their partners rather than the teacher. It is again important to note that CS teachers have limited time during a class period to answer all

questions, and team work in pair programming becomes an important opportunity to alleviate that problem (Sancho-Thomas et al., 2009). However, teachers need to be aware of the differences between team members in terms of knowledge/skills and personality (Ally, Darroch, & Toleman, 2005). Furthermore, CS teachers' need to carefully employ pair programming protocols and remind students to switch roles during pair programming activity (Williams, Wiebe, Yang, Ferzli, & Miller, 2002; Williams & Kessler, 2000).

Learning transfer between programming languages and platforms. Even though limited examples were found, the findings suggest that teachers need strategies to help students transfer learning and build on their previous knowledge in subsequent CS classes. Transfer of learning is important, not only to make connections between concepts and skills, but when developing new learning (Perkins & Salomon, 1992). Transfer of learning can make CS experiences more connected and meaningful. This need was emphasized in previous CS education studies related to programming. Franklin et al. (2016) conducted a study with high-school students' and explored transfer from a visual programming tool to a text-based programming language learning environment. Transfer of learning especially may be helpful in secondary schools. After learning the concepts and logic of programming (e.g., variables, loops) from visual programming tools (e.g., Scratch) in middle school, learning text-based languages (e.g., Python) in high school was found beneficial (Armoni, Meerbaum-Salant, & Ben-Ari, 2015).

Implications for research and practice regarding pedagogical needs. To assist with CS teachers' pedagogical needs, the researcher suggests the following: (1) conducting further research about scaffolding in CS education (2) research and teacher PD in team management (3)

Many studies in other fields recommend scaffolding as an effective instructional technique to support students learning (Ally et al., 2005; Azevedo, et al., 2005; Brush, & Saye, 2000). However, CS education research is limited in scaffolding. In one of those CS education related studies, Sengupta et al. (2013) developed a visual programming environment to support middle school students' science learning and computational thinking through scaffolds provided by the system. The goal of the system was providing timely feedback and support when necessary. The researchers argued that providing scaffolding is crucial in a learning environment that targets CT and science learning. In another study, teacher scaffolding support students learning, increased student engagement for those at risk for poor performance (Israel, Pearson, Tapia, Wherfel, & Reese, 2015). The teachers in this study feel confident about their own use and understanding of student-centered learning strategies and their ability to use differentiated instruction that is appropriate for students with different needs.

The researcher recommends continued data-driven research be conducted on scaffolding in K–12 CS classes. Researchers in CS education may want to address a possible list of questions about scaffolding, such as: (1) What is the definition of scaffolding in CS classes? (2) What kinds of scaffolding can be used in CS classes? (3) How can we support teachers in providing effective scaffolding for students in CS classes? The results of these studies may generate new questions for CS education researchers to address. A synthesis of future research studies' results may be embedded in PD program design and pre-service teacher education program development, and help teachers to facilitate students' self-reflected and active learning in CS classes.

In addition to scaffolding, teachers' team management and building skills are needed in CS classes. The results of the study suggest that teachers need tools and strategies to make sure

all students actively participate and equally contribute to team work in pair programming and PBL environments. The researcher recommends further research to identify successful team building and management strategies, especially in pair programming activities. CS higher education literature may provide frameworks and strategies for effective team building strategies (Sengupta et al., 2013), and become the basis for K–12 studies, guiding researchers to design studies specifically for secondary classrooms. Furthermore, when teachers are trained in pre-service teacher education or in-service PD programs to use student-centered learning strategies, team management strategies may become part of those programs' curriculum.

One idea to address pedagogical needs might be to create video repositories for teachers in different subject areas illustrating successful examples of teacher practices. Watching video recordings of effective teaching practices in classroom may help other CS teachers develop instructional strategies in scaffolding, team management, and learning transfer. For example, researchers at Rutgers University and the University of Wisconsin (Madison) developed an online video repository called Video Mosaic Collaborative (VMC) (Agnew, Mills, & Maher, 2010). VMC provides examples from teachers in both higher education and K–12 teaching different math concepts with a variety of curricular resources. In VMC, teachers can search videos by grade level, math tools, problems, and strands. In addition to the repository of classroom ideas, VMC also serves as a place for academics to conduct research. Similar video repositories could be developed of model CS teachers' classroom practices and collaborative spaces where teachers can learn from each other. Furthermore, a synthesis of research studies in CS education pedagogy in student-centered learning strategies, scaffolding, team management and transfer of learning may be made available to in-service CS teachers in online resources and PD events.

Stakeholders

Embedding CS education in national education goals is a systemic change and requires collaboration and contribution from all the stakeholders for success (Stephenson, 2014). Providing CS education for all students requires involvement and support from researchers, administrators, teachers, parents, policy-makers, students, and the media (Stephenson et al., 2005). Aligned with previous reports (Ericson et al., 2008), the findings suggest that teachers perceived the need for more collaboration especially from administrators, parents, and CS teachers.

Administrators' knowledge of CS. The teachers in this study reported that many of their administrators do not know what CS entails, and that some of those administrators incorrectly identify computer literacy courses (e.g., instruction in Office software) as computer science. Previous studies (e.g., Bell, 2014; Gal-Ezer & Stephenson, 2009) and national reports (e.g., Barr et al., 2013; Ericson et al., 2008) reported the same misconception. When administrators do not know what CS is, they become barriers for secondary CS teachers (Goode, 2008). According to a national survey of 9,805 principals and 2,307 school district superintendents (Google & Gallup, 2016) a majority of administrators in the US value CS and perceive it as a necessary subject. However, the findings suggest that administrators who have misconceptions about CS are not sure what to offer in CS courses. Furthermore, teachers in this study discussed some administrators who do not allocate the necessary school resources for CS courses and teachers, which includes providing funding for various needs (e.g., technology resources and PD programs), hiring CS teachers (Barr et al., 2013), and scheduling adequate rooms and times for CS classes. This imbalanced allocation of resources may be due to current high-stakes testing; they may wish to focus their resources on increasing students' scores on the core subjects on the

tests (Duke, 2004). In a meta-synthesis of 49 qualitative studies that discussed the influence of high-stakes testing requirements on public education, Au (2007) found that schools pay more attention to tested core subjects' curriculums in order to prepare students for high-stakes tests.

Due to limited access to school resources and lack of attention from administrators, CS teachers in this study reported frustration and discouragement. This is supported by research, as Ericson, Guzdial, and Biggers (2007) reported similar findings when looking at teachers' limited access to PD programs and limited CS course offerings in their schools. The findings suggest that it is important to educate school administrators on what CS is and how it differs from computer literacy (Google & Gallup, 2015). If administrators better understand the field, they may pay more attention to CS teachers' needs and understand them better (Israel et al., 2015). Furthermore, they may perceive CS's benefits for students' future and find a place for CS in their schools' curriculum. The Implications section related to stakeholders in this report provides additional details as well as suggestions for practice and research.

Parents' knowledge and involvement in CS. Teachers in this study indicated that parents are important in the promotion and sustainability of CS education classes in secondary schools. They perceived the need for parents' understanding of and involvement in their CS education programs. According to a national report by Google and Gallup (2016), districts and school administrators do not see a demand from parents to offer CS courses in their schools. However, the same report stressed that although most parents value learning CS in K–12 schools, only few approach administrators to indicate their support for CS education.

Previous research documented parents' concerns regarding computers and technology that may explain their limited support for CS education. Their main concern, reported in the research literature, is children's safety (R. Wang, Bianchi, Raley, 2005). Parents who do not feel

comfortable with computers are also concerned with not being able to protect their children's safety in online spaces (R. Wang et al., 2005). Furthermore, previous research reported parents' concerns about children spending too much time with computers and especially playing computer games (Jordan, Hersey, McDivitt, & Heitzler, 2006), which may negatively influence students' learning in core subject areas (Zhang, 2015). For example, the teachers in the present study mentioned that their students' parents' are concerned about children spending long periods of time playing computer games. In some of these instances, the children were actually learning coding practices using Scratch and Alice platforms, but parents interpreted the activities as playing games due to their limited knowledge of computers and CS. Another concern reported in previous research studies is CS stereotyping, where parents think that CS is mainly for male and white students. Therefore, some parents do not feel confident to support female and minority students' involvement in CS education (Clayton & Beekhuyzen, & Nielsen, 2012; Google & Gallup, 2015). For example, in a national report exploring the influential factors for women's career choices, parents' perceptions were found to be an important factor (Google & Gallup, 2015; Moorman & Johnson, 2003). All these concerns may be explained by parents' limited knowledge and low confidence in computers. Media and TV influence parents' perceptions and create those concerns about CS (Google & Gallup, 2016).

Contrary to these concerns, parents with a background in technology or who work in a technology-related field show more support for CS education (Hollingworth, Mansaray, Allen, & Rose, 2011). Parents with CS knowledge and confidence with computers are able to manage their children's computer usage and observe their practices. Those parents see the potential benefits of CS for their children's future (Hollingworth et al., 2011), which include higher salaries, job demand, job flexibility in a wide range of industries, and the ability to make

meaningful contributions for human good such as health research in diseases (Code.org, 2013). Parents with background in CS or a related field can serve as role models for their children (Google & Gallup, 2015) and influence their interest in learning CS.

Aligned with the previous research studies and national reports, the findings of this research suggest parents' knowledge in CS as an influential factor for availability and sustainability of CS education in secondary schools. It is important to educate parents about what CS involves and inform them about potential benefits for their children's future (Bell, 2014).

Community of CS teachers. Teachers in this study identified that they needed a community of CS teachers that they can collaborate to improve their teaching practices. This need emerged from secondary CS teachers' email listserv discussions (n=25) and questionnaire responses (n=20) about the need for colleague support and collaboration. Due to a low number of CS class offerings in schools, most of which are electives, most schools only have one CS teacher. In all the phases of this research, the participants expressed the challenges of being the only CS teacher in their schools, such as building lesson plans and curricular resources alone, lack of feedback on their practices and lack of model CS teachers. They share a feeling of isolation in teaching CS, and expressed a need for collaboration with other CS teachers.

Collaboration among teachers is an important component of effective teaching, which involves sharing resources (e.g., lesson plans, curricular activities), modeling effective teaching practices, and most importantly a place for ongoing teacher professional development (Darling-Hammond & Sykes, 1999). Furthermore, teachers can share their problems in community meetings and benefit both emotionally and intellectually from other teachers (Horn, 2010). Online teacher communities and their advantages are also examined in the previous literature. Hur and Brush (2009) found that online spaces offer freedom from time and distance limitations

and allow teachers to freely share their opinions, explore ideas, and find friends, when dealing with a sense of isolation in their schools. Teachers use social media channels to create a community mostly in their teaching subject areas (Carpenter & Krutka, 2015). However, email listservs are still popular community channels for teachers. The CSTA listserv used in this study serves as an active and successful communication channel for CS teachers.

Other examples of online CS teacher communities are available and active. College Board (n.d.) provides a community space for AP CS teachers where teachers can attend live meetings, comment on discussion threads, and share and access curriculum resources. Code.org (2013) also provides an online space where CS teachers from different states can communicate. Furthermore, they promote collaboration of teachers from their local communities. Even though more examples of these communities can be listed, and CS teachers find online communities beneficial (Tsiotakis, P., & Jimoyiannis, 2013), the findings suggested a community of CS teachers in their own districts is still an important need for face-to-face communication. In-person interactions allow teacher to receive feedback on their teaching practices and watch and model other successful teaching practices in CS, which is important for learning effective instructional strategies in teaching CS content.

Implications for research and practice regarding stakeholders. To assist with CS teachers' pedagogical needs, the researcher suggests the following: (1) increasing administrators' conceptual understanding of CS through professional development programs and meetings (2) increasing parents' conceptual understanding of CS and educating them about application and benefits of CS in their children's life (3) promoting district-wide CS teacher communities for face-to-face collaboration

None of the aforementioned stakeholders should be perceived as more important than any other. Administrators, parents, and CS teacher colleagues are all important roles and missing any of these contributions may negatively influence the ecology of CS teaching in a school (Clayton et al., 2012). While administrators make decisions about allocating school resources, parents need to support administrators in making those decisions and support their children to succeed, and teachers are the ones who live with the decisions and have a vested interest in the outcome.

Even though they do not need deep knowledge about the field to make their decisions, administrators need at least a conceptual understanding in CS, including understanding what CS courses aim to teach and what resources may be needed for a CS class (Israel et al., 2015) Administrators may be given the opportunity to learn more about CS education in K–12 through PD meetings where they can discuss its outcomes for schools and students. District leaders may encourage school administrators' participation in national campaigns (e.g., Hour of Code) in their schools and observe their students in practicing CS.

Parents' participation and support for CS courses appears to depend on their conceptual understanding of CS and its benefits for their children's future. Previous reports identified the influence that national campaigns had on parents' understanding and beliefs about CS education and the reported increase in interest (Google & Gallup, 2015). It is important to continue these promotion campaigns targeting parents. However, for parents living in economically disadvantaged conditions, schools should provide access to information and resources during out of school times. Observing their children developing CS projects (e.g., robotics teams) may increase parents' interest in and knowledge about what CS has to offer. These after-school programs are recommended especially for students who do not have access to computers and role models at home.

Secondary CS teachers do not have the opportunity to collaborate with other CS teachers in most schools. The findings indicate online spaces are convenient places to collaborate because they are free from distance and time limitations (Tsiotakis & Jimoyiannis, 2013). These communities may be built through social media sites. However, face-to-face communities in districts are smaller and may help to develop a better sense of community and close professional relationships (Etzioni, 1999). In addition to collaboration to meet curricular and resource needs, these communities may serve as a place where they can observe other teachers' successful practices. Administrators should encourage district-wide CS teacher meetings, as well as providing opportunities for teacher to watch and learn from colleagues in other schools. This may encourage CS teachers to collaborate and share curriculum resources, pedagogies, and ideas. In addition, national or local PD events may serve as places where teachers can build communities with similar interests and needs (Alvarado & Judson, 2014).

Resources

The findings suggest that computer labs particularly designed for the purpose of teaching CS content are an essential element of CS education in K–12 schools. Even though the majority of teachers in the study appeared to be satisfied with the resources in their lab environments, those who lacked necessary resources found it discouraging. Resources needs include lab environment with necessary equipment (technology and furniture), ideas for effective computer lab design, and administrative access to computer systems in their schools. Resources needs are discussed below in more detail.

Functionality, durability, and health concerns in computer labs. According to a report by the National Center for Education Statistics (Gray, Thomas, & Lewis, 2010), over 90 percent of all U.S. schools have computer labs and Internet access. However, some teachers in

this study reported a lack of access to necessary resources or outdated hardware and limited Internet access in their schools. There are some examples of other research studies reporting access to resources as barriers for teaching CS. In a Midwestern elementary school. Israel et al. (2015) found that teachers in the school perceive access to computer labs as their biggest need, due to an increasing need for using computers in different subject areas. Even though the number was small, lack of technology resources was a serious issue for some CS teachers in this study. Over 30 percent of the teachers who completed the questionnaire reported a lack of access to a satisfying computer lab. This dissatisfaction and need may be due to a lack of resources or a need for computer labs particularly designed for teaching CS classes, which includes not only specific hardware and software but also additional concerns related to students' health and the durability of the resources.

In addition to the functionality of the equipment, CS teachers reported the need for more durable equipment and furniture. Students spend extensive periods of time with computers during computer science classes. They do programming, design and develop products, and interact with their peers while working on computers. Long-time usage is related to health concerns including pain, vision, and muscle problems (Jones & Orr, 1998). There are extensive studies connecting college students' computer usage patterns and musculoskeletal discomfort. Noack-Cooper, Sommerich, and Mirka (2009) reported college students' discomfort due to poor posture habits. Therefore, it is important to provide students with computer peripherals and furniture that may lessen these health concerns. Furthermore, Israel et al., (2015) called attention to students with special needs in their study and stressed the need for specially designed equipment for students with disabilities. It is not surprising that durable and adaptable furniture and computer peripherals (keyboard, mouse) emerged as a need from the findings. Some

teachers in this study reported that the equipment and furniture in their computer labs are not adaptable for students needs and health concerns, and break easily and often.

Computer lab layout ideas. In addition to the aforementioned considerations, teachers in this study also solicited ideas for computer lab layout design ideas for students' interactions, collaboration, and classroom management. With the focus on pair programming and student-centered learning strategies, this is becoming more important for CS teachers. Higher education institutions and private companies offer different layouts for computer labs for business practices; however, there is limited research and consideration regarding computer lab layouts in CS education (Young & Huggard, 2003). It is important to emphasize that the layout of the lab can have a significant impact on a teachers' pedagogy and students' learning outcomes (Pretto, 2011).

In a study with IT teachers, Pretto (2011) reported that school computer labs do not allow for adapting workstation movement to the needs of team work. Because of this inflexible computer lab design, Pretto has argued that IT teachers tend to choose a more traditional teaching style and find it challenging to try student-centered learning strategies.

Administrative access to computer systems. The findings suggest that CS teachers lack administrative rights (e.g., for installing software) to the computers in their labs, which can become a barrier for effective teaching. Teaching CS content may require teachers to make instant changes on the computers during class time, such as installing plugins, accessing network settings, or installing new software. Some CS teachers reported that their technology coordinators block some of the essential features of integrated development environments (IDE) for the same security reasons, which sometimes limit students' problem solving and production

in their programming projects. IDEs are the software tools where students code and test their programming projects.

Lack of support from technology coordinators was reported as a barrier in technology integration studies in K–12 education (Ertmer et al., 2012; Gray, Thomas, & Lewis, 2010; Hew & Brush, 2007). Those studies refer to other subject-area teachers' issues with operating and maintaining technology resources. However, CS teachers in this study expressed that they have the knowledge and skills to deal with technology needs and problems without their technology coordinators' assistance. Teachers wanting a less structured and dynamic CS classroom environment with student-centered pedagogies are discouraged by a lack of administrative access to computer systems. Even teachers with rich computer facilities reported that their curriculums were limited by technology coordinators' security concerns.

Funding and time needs are connected. Constant changes in CS and CS education have a direct impact on funding (Barr et al., 2013; Ericcson et al., 2008). CS teachers expressed a desire for money to buy technology equipment for new CS class offerings (e.g., robotics kits) and to update their current computers' hardware and software. Because of changes in CS education content and pedagogies, they also want access to ongoing PD, which usually requires payment (Barr et al., 2013; Gal-Ezer & Stephenson, 2009). New standards suggest new curriculum resources and this would require funds be made available to CS teachers. However, teachers reported that they aren't getting the funding to meet these needs, and that they often use their non-work time to develop curriculums and improve their knowledge and skills. As mentioned previously, funding and time are seen as urgent issues in CS teaching, but they become even more problematic when teachers are responsible for multiple courses and are the only CS teacher in their schools.

Implications for research and practice regarding resources. To assist with CS teachers' resources needs, the researcher suggests the following: (1) providing access to teachers to computer labs designed with purpose of teaching CS content (2) creating collaboration between CS teachers and technology coordinators regarding CS teachers' needs and technology coordinators' safety concerns (3) providing funding and time for CS teacher PD.

The initial goal regarding the resources may be to equip classrooms with necessary technology resources. But whether this goal was met depends upon available funding. When there is funding, recommendation for CS labs would be considering health concerns and curricular, pedagogical, and contextual needs before selecting the technology resources. Most classroom activities involve computer use these days and more in CS classes. Teachers need to make sure that their students do not experience any health concerns (back and muscle pains) due to inappropriate computer usage (Oyewole, Haight, & Freivalds, 2010) Therefore, it is important to train students early in their education about healthy computer use and consider ergonomic standards for computer equipment selecton to reduce possible muscoloskeletal risks (Feathers, Rollings, & Hedge, 2013)

In addition to availability and access to resources considering the aforementioned factors, it is also important to carefully plan computer labs' layout for student-centered learning. Student-centered practices such as PBL and pair programming require less structured classroom environments so students can freely interact and collaborate (Ally et al., 2005; Nagappan et al., 2003; Williams & Kessler, 2000). Classroom management is another factor to consider when making decisions about room layout. Teachers need an environment that allows them to easily observe and assist students efficiently.

When technology resources are available and accessible by the classroom teacher, and the room layout is designed for collaboration and interaction, there are still other conditions to consider for an effective CS lab environment. CS teachers need access to administrative rights to the systems in their computer labs. Meeting this need requires cooperation between technology coordinators and CS teachers, and CS teachers' understanding about technology coordinators' security concerns. The relationship between CS teachers and technology coordinators may be promoted to develop a mutual understanding. CS teachers work alone in most U.S. schools, and technology coordinators may also become successful collaborators for them to improve their teaching.

The final piece of the resource need is funding. Funding is not only important for buying equipment but also for PD. This is emphasized in multiple parts of the study conversations, where teachers stressed the need for PD for increasing both content and pedagogical knowledge. They look for programs that are convenient in terms of cost, location, and time. Even though CS teachers focused their discussions on finding funding for PD, there are higher education institutions that offer free programs for CS teachers supported by national organizations (Menekse, 2015). These programs should be promoted more because the findings suggest that secondary teachers are unaware of the options available to them. These programs also offer curriculum materials and collaboration opportunities for curriculum development.

Research Question 2

How do teachers' needs vary based on years of teaching experience and background (e.g., education, training) in CS?

The findings suggest that CS teachers' needs vary according to their years of teaching experience and background in CS. The following section summarizes and discusses the findings, and explores possible recommendations for research and practice.

Years of Teaching Experience in CS

When the findings are examined in detail through the lens of years of teaching experience, novice teachers (i.e. those with less than 4 years of experience) were more likely to express needs, compared to more experienced teachers. This is not a surprising finding due to novice teachers' limited experience and the added responsibilities in instructional planning (Fantilli & McDougall, 2009). The findings suggest that novice teachers' needs are primarily pedagogical, curricular, students-related, stakeholders-related and PD. Each of these are examined below. Even though they were equally important for all teachers in the study, resource needs in general were no different for novice and experienced teachers.

Pedagogical needs. The findings suggest that novice CS teachers need help and guidance to be better prepared for teaching CS. In terms of pedagogy, novice teachers expressed the needs to learn new instructional strategies, teach computational thinking, help students transfer their learning between CS courses, and develop strategies for handling students' questions in the classroom. Relatively, the first three of these needs require PD. In fact, almost all the novice teachers in this study expressed the need for PD. However, the findings demonstrate that this group does not know about higher education institutions that offer PD programs. They may need guidance to find available programs appropriate for their needs. The fourth need, answering

students' questions in classroom, may be related to their concerns regarding classroom management. Previous studies indicate that novice teachers' have low self-efficacy (De Neve et al., 2015) about addressing management issues and students' concerns and problems in classroom (Fantilli & McDougall, 2009). Examining the data from this study, it is not surprising to find that as secondary CS teachers gain more teaching experience, they tend to perceive pedagogical needs less as a need.

Curricular needs. Novice teachers in this study perceived their needs as primarily curricular, especially embedding the principles of computational thinking into their curriculum. Even though the researcher does not have evidence specifically supporting this claim for novices, this may be due to these teachers' limited understanding of computational thinking and its components, as emphasized in the conversations of all the teachers.

Compared to experienced teachers, more novice teachers reported the need for materials to assess students' learning. This conflicts with some previous research. When Melnick and Meister (2008) compared beginning and experienced teacher concerns, they found that beginning teachers felt more prepared using different assessment techniques. The conflict in this research may be due to novice teachers' need to learn new instructional strategies for teaching CS. Using new instructional strategies involves developing new assessment techniques for them (Yadav et al., 2015). Therefore, it is not surprising to find the need for assessment materials among novice teachers who also want to use student-centered learning strategies in their classroom.

Student-related needs. The findings suggest that novice teachers were the ones most concerned with having students with limited math and reading comprehension skills in their CS classrooms. This concern appears to decline after 10 years' experience, but was a shared need for the majority of the subjects with fewer years' experience. Therefore, it is important to approach

this as a need for all teachers and develop strategies that can be applicable to students with different backgrounds. With the national goal of providing CS for All, meeting this need becomes a crucial concern (M. Smith, 2016).

Stakeholders-related needs. Unlike the previous findings, collaboration with stakeholders was expressed more among experienced teachers. Even though the findings do not suggest a major difference, teachers with 4-10 years of experience were more likely to report administrators' understanding of CS and parents' involvement in CS education activities as a need or strong need compared to both novice teachers and teachers with more than 10 years' experience. The researcher does not have supporting evidence to explain this finding; however, this may due to the confidence shared by this mid-range group of teachers, as well as their expressed goals for involving administrators and parents in CS education efforts. They may desire to build relationship with the stakeholders, and see them as critical in their teaching effectiveness.

Implications for research and practice for teachers with varying experience. To assist with CS teachers with different years of CS teaching experience, the researcher suggests the following: (1) offering PD programs designed by considering participants' years of CS teaching experience (2) building mentorship relationship with novice and experienced teachers (3) reducing novice teachers' workload and leaving them time for PD and CS teacher preparation.

Aligned with the findings in previous research studies, novice teachers are the critical group for PD (Menekse, 2015), as expressed in this study. As CS teachers receive more experience, they tend to perceive less of a need for PD in terms of pedagogy, curriculum, and classroom management. Because of this, it seems prudent to prepare prospective CS teachers for

these needs starting at the teacher education phase. However, there are only a small number of programs that offer CS education in the US, and most of those are poorly designed for training future CS teachers (Barr et a., 2013). Therefore, PD programs that aim to prepare in-service CS teachers should prioritize their goals considering teachers' years of experience. This research may offer guidance for PD planning.

The researcher also recommends that school districts to build mentoring relationships between novice and experienced teachers. Providing mentorship is essential for improving novice teachers' practices (Fantilli & McDougall, 2009). However, as the findings suggest, most CS teachers work alone in their schools; they need a community of CS teachers they can turn to for mentoring. In these cases, districts must be responsible for building mentoring relationships among teachers in different schools (Melnick & Meister, 2008).

Novice teachers have additional planning and PD tasks as they are getting prepared for teaching CS. Therefore, time becomes one of their biggest issues. Especially in a field constantly changing, it is important to reduce their teaching load and allocate mentorship times for novice teachers, both to collaborate with experienced teachers and to observe effective teaching practices (Darling-Hammond, 2005; Fantilli & McDougall, 2009). Allocating funding and time for novice teachers' PD may become a school's priority.

Background in CS

When teachers' expressed needs are examined through the lens of their background in CS, the findings suggest that needs vary for different groups. In general, even though important for all the participants, curricular needs, stakeholder-related needs, and resource needs did not vary among teachers with different background in CS. This section examines the variety of pedagogical, student-related, and PD needs that were expressed across these populations.

Pedagogical needs. Teachers with "CS" or "CS education" background reported the need for learning new instructional strategies teaching CS. Even though these teachers may have CS content knowledge, they may have less pedagogical knowledge. On the other hand, teachers with "no CS or CS education" background are less likely to report "learning new instructional strategies" as a need. The findings suggest that this is due to this group's teaching experience or education in a different subject area, such as math, business, or science education. It is important to note that teaching experience in a different subject area is useful in regards to instructional strategies in general. However, needs that involve pedagogical content knowledge (Barr et al., 2013) become a concern for this group. An example of pedagogical content knowledge need expressed by teachers in this study is "helping students transfer learning between programming languages." This need requires a teacher to understand both the CS content and pedagogy and be able to integrate them. Those teachers with "no CS or CS education" background may have the pedagogical knowledge but may need more CS content knowledge to effectively teach CS.

Student-related needs. Understanding a student groups' interests and limitations is an important condition for effective teaching. Teachers with "no CS or CS education" background and teachers with a "CS background" more likely to report "increasing students' interest" and "teaching students with limited math and reading comprehension skills" as needs, compared to teachers with a background in "CS education." This finding suggests the importance of the knowledge of learners (Shulman, 1986) and understanding how to approach students with different interests and preconceptions. CS education programs were designed with the purpose of preparing future CS teachers with pedagogical content knowledge (Melnick & Meister, 2008) and these programs might help the teachers with "CS education" background about student-related needs.

Professional development needs. All teachers reported the need for ongoing PD in the findings. However, more teachers with either a background in "CS education" or "no CS or CS education" reported this need; fewer teachers with "CS" background expressed a need for PD. This may be because these teachers have more CS content knowledge. The teachers in this group reported completing a B.S. or a graduate degree in a CS program, meaning they had taken a large number of CS classes.

Implications for research and practice for teachers with different backgrounds. To assist with CS teachers with different backgrounds in CS, the researcher suggests the following: (1) offering PD programs designed by considering participants' years of CS teaching experience (2) establishing an assessment system that ensures all the CS teachers have the required CS content and pedagogy knowledge.

For all teachers with different backgrounds, PD is an ongoing need (Darling-Hammond, 2005) and CS education institutions and organizations should be required to provide comprehensive PD for teachers to maintain and improve their CS content knowledge and skills as well as pedagogical knowledge and skills (Fantilli & McDougall, 2009). The findings suggest that teachers' background in CS makes a difference in teaching the field for secondary students. The researcher's initial recommendation is organizing PD programs that are designed based on teachers' different backgrounds in CS. The findings of this study may provide guidance for designing such PD programs. For instance, for teachers with CS background or work experience only, it is important to establish a PD system that provides pedagogical content knowledge for teaching CS, especially student-centered teaching philosophy and practices that includes instructional strategies and assessment. For the teachers with "no CS or CS education" background, PD programs may focus more on CS content and pedagogical content knowledge

needs. CS content knowledge was the primary need for this group and it may be helpful to allocate more preparation time in schools for teachers with low CS content knowledge to improve their CS knowledge and skills. For teachers with background in "CS" or "CS education," it may be helpful to educate them about instructional and assessment strategies for teaching CS.

With regards to certification and licensing requirements, this research suggest establishing an assessment system that ensures all the CS teachers have the required CS content and pedagogy knowledge as well as teaching experience before teaching the discipline (Barr et al., 2013). For future research, it may be helpful to define what makes a successful computer science teacher. Studies comparing student learning among teachers with different backgrounds may help to answer this question. The researcher suggests CS content knowledge as an important condition for teaching CS; however, defining the limits and goals of CS content knowledge for teaching CS in secondary schools is a necessary first step in making that determination.

Implications

The findings of this research have implications to planning PD programs for secondary CS teachers. PD programs designed with teachers' perceived needs in mind may have better outcomes for improving CS teaching. Lee (2005) developed a PD model that emphasized teachers as the decision makers in their own professional development. Lee tested the model with mathematics teachers and conducted surveys, interviews, concept maps (observing entry and exit knowledge and beliefs), participant assignments, classroom visits, and audio/video recordings. Lee concluded that evidence-based PD based on mathematics teachers' needs increased the chance of program success, which may have a direct influence on students' learning in classroom.

In the present study, developing student-centered learning strategies for CS classes and scaffolding strategies to support students' learning experiences were critical needs expressed by the majority of the secondary CS teachers in this study. Those designing PD programs should also carefully consider CS teachers' expressed needs for assessment materials in student-centered learning environments. PD programs may provide spaces where teachers develop new knowledge of instructional strategies, learn from each other, and develop curriculum resources by collaborating with other CS teachers.

The study findings have additional implications for the design of CS teacher education programs. National reports stressed that American CS education programs are poorly designed to prepare future CS teachers (Barr et al., 2013). These findings may be used to redesign teacher education programs based on the study teachers' perceived needs. These programs may include goals for teachers coming from different CS backgrounds. While prospective teachers with a strong background in CS may focus on pedagogical needs, prospective teachers pursuing a CS teaching license while coming from other subject areas may focus on developing their CS content knowledge.

Additional implications arose for district and school planning in CS education. Districts and school administrators play a critical role in meeting secondary CS teachers' needs. They may develop a systematic need assessment procedure to constantly evaluate in-service teachers' needs in their districts and schools. The quality of teachers' professional lives is related to their retention and recruitment (Cockburn, 2000), and retaining CS teachers in the profession is a major concern. Because their skills are attractive to private companies, CS teachers may leave the teaching profession if their personal and professional needs are not understood and met by administrators. Furthermore, administrator should take an active role to establish environments

where CS teachers can meet and collaborate with others. Most U.S. schools have only one CS teacher, so districts should plan and lead mentorship activities between experienced and novice teachers to collaborate between different schools.

Areas for Future Research

This study suggests that future studies be conducted to explore the following topics in CS classrooms:

1. Successful scaffolding strategies in secondary CS classes;
2. The design of effective curricular materials for computational thinking and assessment;
3. The development of strategies to increase student interest and motivation in learning CS;
4. Strategies for increasing student' enrollment in CS classes, including underrepresented populations;
5. The education of stakeholders about what CS, its learning goals, and outcomes for K–12 education;
6. Computer lab design for the purpose of teaching CS.

The recommendations made in this research are based on teachers' perceptions of their needs and may be limited. It may be helpful to develop research studies that observe teachers' practices in their classroom. Action research studies that allow teachers and researchers to work together (Barr et al., 2013) and aim to address secondary CS teachers' needs in practice could surface additional information. Developing PD programs based on the suggestions of this study and observing the outcomes in secondary classrooms may promote and develop data-driven strategies for teacher PD. With the goal of CS for All, this study may be replicated to explore the needs of elementary CS teachers, and then comparing the needs of elementary, middle, and high school teachers.

Limitations

Although it represents a large population of secondary CS teachers, the participants in this study are all members of one organization, and do not represent all CS teachers in the US. Therefore, the findings are not suggested to be generalized. The data represents teachers who are members of the CSTA, and were identified as secondary CS teachers in the email listserv based on available information. The CSTA member database was used to distribute the questionnaire based on teachers' self-reported information as a middle- or high-school teacher. There is a risk that some elementary teachers or higher education faculty were included in the study. Furthermore, there may be full time CS teachers and part-time CS teachers who were teaching others subjects (e.g., math, science) in the participants. The researcher did not collect any data about their CS teaching load and cannot report comparative findings in this respect.

The findings of this study are based on participants' own perceptions. The email listserv, questionnaire, and interviews provided opportunities for them to share their needs. The researcher did not observe the practices in their schools and classroom, and this limited the findings and the researcher's interpretations.

REFERENCES

Abrams, K. (1989). Gender discrimination and the transformation of workplace norms. *Vanderbilt Law Review, 42*, 1183-1248.

Ackerman, G. L. (2015). *Technology-rich teaching: Classrooms in the 21st century*. Lanham, MD: University Press of America.

ACM. (1992). Code of ethics and professional conduct. Retrieved from https://www.acm.org/about-acm/acm-code-of-ethics-and-professional-conduct

Adams, B. L. (1999). Nursing education for critical thinking: An integrative review. *Journal of Nursing Education, 38*(3), 111-119.

Agnew, G., Mills, C. M., & Maher, C. A. (2010). VMCAnalytic: Developing a collaborative video analysis tool for education faculty and practicing educators. In Proceedings of the *2010 43rd Hawaii International Conference on System Sciences*, Jan 5-8, 2010, Honolulu, HI, USA.

Ahoniemi, T., & Karavirta, V. (2009). Analyzing the use of a rubric-based grading tool. *SIGCSE Bulletin, 41*(3), 333-337.

Allen, Q. (2015). Race, culture and agency: Examining the ideologies and practices of U.S. teachers of Black male students. *Teaching and Teacher Education, 47*, 71-81.

Ally, M., Darroch, F., & Toleman, M. (2005). A Framework for understanding the factors influencing pair programming success. In H. Baumeister, M. Marchesi, & M. Holcombe (Eds.), *Extreme programming and agile processes in software engineering* (pp. 82-91).

Alvarado, C., & Judson, E. (2014). Using targeted conferences to recruit women into computer science. *Communications of the ACM, 57*(3), 70-77.

Alwin, D. F. (2010). How good is survey measurement? Assessing the reliability and validity of survey measures. In P. H. Rossi, J. D. Wright, & A. B. Anderson (Eds.), *Handbook of Survey Research* (pp. 405-436). Bingly, United Kingdom: Emerald Group Publishing.

American Modeling Teachers Association. (n.d.). *Curriculum repository*. Retrieved from http://modelinginstruction.org/teachers/resources/

Andrade, H. G. (2000). Using rubrics to promote thinking and learning. *Educational Leadership, 57*(5), 13-19.

Andrade, H. L., Du, Y., & Wang, X. (2008). Putting rubrics to the test: The effect of a model, criteria generation, and rubric-referenced self-assessment on elementary school students' writing. *Educational Measurement: Issues and Practice, 27*(2), 3-13.

Anghileri, J. (2006). Scaffolding practices that enhance mathematics learning. *Journal of Mathematics Teacher Education*, *9*(1), 33-52.

Armoni, M., Meerbaum-Salant, O., & Ben-Ari, M. (2015). From scratch to "real" programming. *ACM Transactions on Computing Education, 14*(4), 25.

Au, W. (2007). High-stakes testing and curricular control: A qualitative metasynthesis. *Educational Researcher, 36*(5), 258-267.

Austing, R. H., Barnes, B. H., Bonnette, D. T., Engel, G. L., & Stokes, G. (1979). Curriculum'78: Recommendations for the undergraduate program in computer science—A report of the ACM curriculum committee on computer science. *Communications of the ACM, 22*(3), 147-166.

Avalos, B. (2011). Teacher professional development in teaching and teacher education over ten years. *Teaching and Teacher Education, 27*(1), 10-20.

Azevedo, R., Cromley, J. G., Winters, F. I., Moos, D. C., & Greene, J. A. (2005). Adaptive human scaffolding facilitates adolescents' self-regulated learning with hypermedia. *Instructional Science, 33*(5-6), 381-412.

Bailin, S. (2002). Critical thinking and science education. *Science & Education, 11*(4), 361-375.

Baird, W. E., & Rowsey, R. E. (1989). A survey of secondary science teachers' needs. *School Science and Mathematics, 89*(4), 272-284.

Barnes, G., McInerney, D. M., & Marsh, H. W. (2005). Exploring sex differences in science enrolment intentions: An application of the general model of academic choice. *Australian Educational Researcher*, *32*(2), 1-24.

Barr, D., Harrison, J., & Conery, L. (2011). Computational thinking: A digital age skill for everyone. *Learning & Leading with Technology*, *38*(6), 20-23.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: What is involved and what is the role of the computer science education community? *ACM Inroads, 2*(1), 48-54.

Barr, V., Cronin, D., Finkel, S., Gal-Ezer, J., Morreale, P., Pirmann, T., . . . Yadav, A. (2013). *Bugs in the System: Computer Science Teacher Certification in the U.S.* Retrieved from http://csta.acm.org/ComputerScienceTeacherCertification/sub/CSTA_BugsInTheSystem.pdf

Basawapatna, A. R., Repenning, A., & Koh, K. H. (2015). Closing the cyberlearning loop: Enabling teachers to formatively assess student programming projects. In Proceedings of *the 46th ACM Technical Symposium on Computer Science Education*, March 4-7, 2015, Kansas City, MO.

Bazeley, P., & Jackson, K. (2013). *Qualitative data analysis with NVivo*. Thousand Oaks, CA: Sage Publications.

Bechtel, P. A., & O'Sullivan, M. (2007). Enhancers and inhibitors of teacher change among secondary physical educators. *Journal of Teaching in Physical Education, 26*(3), 221-235.

Becker, K. (2003). Grading programming assignments using rubrics. *ACM SIGCSE Bulletin. 35*(3), 253-253.

Beede, D., Julian, T., Langdon, D., McKittrick, G., Khan, B., & Doms, M. (2011). *Women in STEM: A gender gap to innovation*. Retrieved from U.S. Department of Commerce, Economics and Statistics Administration website: http://www.esa.doc.gov/

Belfi, B., Gielen, S., De Fraine, B., Verschueren, K., & Meredith, C. (2015). School-based social capital: The missing link between schools' socioeconomic composition and collective teacher efficacy. *Teaching and Teacher Education, 45*, 33-44.

Beligiannis, G. N., Moschopoulos, C. N., Kaperonis, G. P., & Likothanassis, S. D. (2008). Applying evolutionary computation to the school timetabling problem: The Greek case. *Computers & Operations Research, 35*(4), 1265-1280.

Bell, T. (2014). Establishing a nationwide CS curriculum in New Zealand high schools. *Communications of the ACM, 57*(2), 28-30.

Belland, B. R. (2014). Scaffolding: Definition, current debates, and future directions. In J. M. Spector, M. D. Merrill, J. Elen, & M. J. Bishop (Eds.), *Handbook of Research on Educational Communications and Technology* (pp. 505-518). New York, NY: Springer.

Ben-Ari, M. (2001). Constructivism in computer science education. *Journal of Computers in Mathematics and Science Teaching, 20*(1), 45-74.

Benitti, F. B. V. (2012). Exploring the educational potential of robotics in schools: A systematic review. *Computers & Education, 58*(3), 978-988.

Bergin, S., & Reilly, R. (2005). Programming: Factors that influence success. *ACM SIGCSE Bulletin, 37*(1), 411-415.

Bers, M., Ponte, I., Juelich, K., Viera, A., & Schenker, J. (2002). Teachers as designers: Integrating robotics in early childhood education. *Information Technology in Childhood Education, 2002*(1), 123-145.

Betoret, F. D. (2009). Self-efficacy, school resources, job stressors and burnout among Spanish primary and secondary school teachers: a structural equation approach. *Educational Psychology, 29*(1), 45-68.

Biernacki, P., & Waldorf, D. (1981). Snowball sampling: Problems and techniques of chain referral sampling. *Sociological Methods & Research, 10*(2), 141-163.

Billig, S. H. (2000). Research on K-12 school-based service learning: The evidence builds. Phi Delta Kappan, 81(9), 658.

Blumenreich, M., & Gupta, A. (2015). The globalization of teach for America: An analysis of the institutional discourses of teach for America and teach for India within local contexts. *Teaching and Teacher Education*, *48*, 87-96.

Brault, M.-C., Janosz, M., & Archambault, I. (2014). Effects of school composition and school climate on teacher expectations of students: A multilevel analysis. *Teaching and Teacher Education, 44*, 148-159.

Braun, V., & Clark, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3*(2), 77-101.

Brennan, K., & Resnick, M. (2012). Using artifact-based interviews to study the development of computational thinking in interactive media design. Paper presented at the annual American Educational Research Association meeting, Vancouver, BC, Canada.

Brown, K. L. (2003). From teacher-centered to learner-centered curriculum: Improving learning in diverse classrooms. *Education, 124*(1), 49-54.

Brush, T., & Saye, J. (2000). Implementation and evaluation of a student-centered learning unit: A case study. *Educational Technology Research and Development, 48*(3), 79-100.

Burke, P. F., Aubusson, P. J., Schuck, S. R., Buchanan, J. D., & Prescott, A. E. (2015). How do early career teachers value different types of support? A scale-adjusted latent class choice model. *Teaching and Teacher Education, 47*, 241-253.

Camera, L., (2015). Women still underrepresented in STEM fields. Retrieved from https://www.usnews.com/news/articles/2015/10/21/women-still-underrepresented-in-stem-fields

Carey, S., Fuschetto, B., Lee, I., Moix, D., O'Grady-Cunniff, D., Owens, B. B., . . . Verno, A. (2011). *K–12 computer science standards.* Retrieved from http://csta.acm.org/Curriculum/sub/K12Standards.html

Carpenter, J. P., & Krutka, D. G. (2015). Engagement through microblogging: educator professional development via Twitter. *Professional Development in Education, 41*(4), 707-728.

Chase, J. D., & Okie, E. G. (2000). Combining cooperative learning and peer instruction in introductory computer science. *SIGCSE Bulletin, 32*, 372-376.

Cimons, M. (2010). Operation reboot: IT professionals become computer science teachers. https://www.usnews.com/science/articles/2010/05/10/operation-reboot-it-professionals-become-computer-science-teachers.

Clayton, K., Beekhuyzen, J., & Nielsen, S. (2012). Now I know what ICT can do for me! *Information Systems Journal, 22*(5), 375-390.

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. *Journal of Educational Psychology, 76*(6), 1051-1058.

Cockburn, A. D. (2000). Elementary teachers' needs: Issues of retention and recruitment. *Teaching and Teacher Education*, *16*(2), 223-238.

Cockburn, A., & Williams, L. (2000). The costs and benefits of pair programming. In M. Marchesi & G. Succi (Eds.), *Extreme programming examined* (pp. 223-247). Boston, MA: Addison-Wesley Longman Publishing.

Code.org. (2013). *About us.* Retrieved from https://code.org/about

College Board. (2016). *AP Computer Science Principles.* Retrieved from https://advancesinap.collegeboard.org/stem/computer-science-principles

College Board. (n.d.). Welcome to the AP computer science: A teacher community. Retrieved from https://apcommunity.collegeboard.org/web/apcompsci/home

Collins, D. (2003). Pretesting survey instruments: An overview of cognitive methods. *Quality of Life Research, 12*(3), 229-238.

Cooke, L. (2010). Assessing concurrent think-aloud protocol as a usability test method: A technical communication approach. *IEEE Transactions on Professional Communication, 53*(3), 202-215.

Creswell, J. W. (2003). *Research design: Qualitative, quantitative, and mixed methods approaches*. Thousand Oaks, CA: Sage.

Creswell, J. W., & Clark, V. L. P. (2007). *Designing and conducting mixed methods research*. Thousand Oaks, CA: Sage.

Crouch, C. H., & Mazur, E. (2001). Peer instruction: Ten years of experience and results. *American Journal of Physics, 69*(9), 970-977.

Crouch, C. H., Watkins, J., Fagen, A. P., & Mazur, E. (2007). Peer instruction: Engaging students one-on-one, all at once. *Research-Based Reform of University Physics, 1*(1), 40-95.

CSTA. (2011). CSTA K—12 computer science standards. Retrieved from http://c.ymcdn.com/sites/www.csteachers.org/resource/resmgr/Docs/Standards/CSTA_K-12_CSS.pdf

CSTA. (2013). *National secondary school computer science survey: Comparison of results from 2005, 2007, 2009, 2011, and 2013 surveys.* Retrieved from http://www.csteachers.org/?page=Resources

CSTA. (2015). *CSTA national secondary school computer science survey 2015*. Retrieved from http://www.csteachers.org/?page=Resources

CSTA. (2016). *Draft 2016 CSTA K-12 CS standards: We need your feedback!* Retrieved from http://blog.csta.acm.org/2016/02/17/draft-2016-csta-k-12-cs-standards-we-need-your-feedback/

Daigneault, P.-M., & Jacob, S. (2014). Unexpected but most welcome mixed methods for the validation and revision of the participatory evaluation measurement instrument. *Journal of Mixed Methods Research, 8*(1), 6-24.

Darling-Hammond, L., & Sykes, G. (1999). *Teaching as the Learning Profession: Handbook of Policy and Practice*. San Francisco, CA: Jossey-Bass.

Darling-Hammond, L. (2000). Teacher quality and student achievement. *Education Policy Analysis Archives, 8*(1), 1-46.

Darling-Hammond, L. (2005). Teaching as a profession: Lessons in teacher preparation and professional development. *Phi Delta Kappan, 87*(3), 237-240.

Darling-Hammond, L. (2010). Teacher education and the American future. *Journal of Teacher Education, 61*(1-2), 35-47.

Davenport, D. (2000). Experience using a project-based approach in an introductory programming course. *IEEE Transactions on Education, 43*(4), 443-448.

Davis, E. A., & Miyake, N. (2004). Explorations of scaffolding in complex classroom systems. *The journal of the Learning Sciences, 13*(3), 265-272.

De Neve, D., Devos, G., & Tuytens, M. (2015). The importance of job resources and self-efficacy for beginning teachers' professional learning in differentiated instruction. *Teaching and Teacher Education, 47*, 30-41.

Department of Education. (2004). *New no child left behind flexibility: Highly qualified teachers.* Retrieved from http://www2.ed.gov/nclb/methods/teachers/hqtflexibility.html

desJardins, M. (2015). *The real reason U.S. students lag behind in computer science.* Retrieved from http://fortune.com/2015/10/22/u-s-students-computer-science/

Dhorsan, A., & Chachuaio, A. M. (2008). The local curriculum in Mozambique: The Santa Rita community school in Xinavane. *Prospects*, *38*(2), 199-213.

Dl Geroimimo, J. (1985). Boredom: The hidden factor affecting teacher exodus. *The Clearing House, 59*(4), 178-178.

Downes, T., & Looker, D. (2011). Factors that influence students' plans to take computing and information technology subjects in senior secondary school. *Computer Science Education, 21*(2), 175-199.

Duffy, T. M., & Jonassen, D. H. (1992). *Constructivism and the technology of instruction: A conversation*. Mahwah, NJ: Lawrance Erlbaum Associates, Inc.

Duke, D. L. (2004). The turnaround principal: High-stakes leadership. *Principal, 84*(1), 12-23.

Eamon, M. K. (2005). Social-demographic, school, neighborhood, and parenting influences on the academic achievement of latino young adolescents. *Journal of Youth and Adolescence, 34*(2), 163-174.

English, L. D. (2016). Revealing and capitalising on young children's mathematical potential. *ZDM*, *48*(7), 1079-1087.

Ericson, B., Guzdial, M., & Biggers, M. (2007). Improving secondary CS education: Progress and problems. *ACM SIGCSE Bulletin*, *39*(1), 298-301.

Ericson, B., Armoni, M., Gal-Ezer, J., Seehorn, D., Stephenson, C., & Trees, F. (2008). *Ensuring exemplary teaching in an essential discipline: Addressing the crisis in computer science teacher certification*. Retrieved from CSTA website: https://csta.acm.org/Communications/sub/DocsPresentationFiles/CertificationFinal.pdf

Ertmer, P. A., & Ottenbreit-Leftwich, A. T. (2010). Teacher technology change: How knowledge, confidence, beliefs, and culture intersect. *Journal of Research on Technology in Education*, *42*(3), 255-284.

Ertmer, P. A., Ottenbreit-Leftwich, A. T., Sadık, O., Sendurur, E., & Sendurur, P. (2012). Teacher beliefs and technology integration practices: A critical relationship. *Computers & Education, 59*(2), 423-435.

Esteves, M., Fonseca, B., Morgado, L., & Martins, P. (2011). Improving teaching and learning of computer programming through the use of the Second Life virtual world. *British Journal of Educational Technology, 42*(4), 624-637.

Etzioni, A. E. O. (1999). Face-to-face and computer-mediated communities, a comparative analysis. *The Information Society*, *15*(4), 241-248.

Exploring Computer Science. (2016). *Exploring computer science curriculum.* Retrieved from http://www.exploringcs.org

Exter, M. E. (2011). *The educational experiences of software designers working in education/instructional technology related fields* (Doctoral dissertation). Retrieved from ProQuest Dissertations & Theses Global. (UMI No: 3491471)

Fagin, B., & Merkle, L. (2003). Measuring the effectiveness of robots in teaching computer science. *ACM SIGCSE Bulletin, 35*(1), 307-311.

Fantilli, R. D., & McDougall, D. E. (2009). A study of novice teachers: Challenges and supports in the first years. *Teaching and Teacher Education, 25*(6), 814-825.

Feathers, D. J., Rollings, K., & Hedge, A. (2013). Alternative computer mouse designs: Performance, posture, and subjective evaluations for college students aged 18–25. *Work, 44*, 115-122.

Fessakis, G., Gouli, E., & Mavroudi, E. (2013). Problem solving by 5–6 years old kindergarten children in a computer programming environment: A case study. *Computers & Education, 63*, 87-97.

Fishman, B. J., Marx, R. W., Best, S., & Tal, R. T. (2003). Linking teacher and student learning to improve professional development in systemic reform. *Teaching and Teacher Education, 19*(6), 643-658.

Flowers, R., & Chodkiewicz, A. (2009). Local communities and schools tackling sustainability and climate change. *Australian Journal of Environmental Education*, *25*, 71-81.

Fraenkel, J., & Wallen, R. (2010). *How to design and evaluate research in education*. New York, NY: McGraw-Hill Education.

Franke, B., Century, J., Lach, M., Wilson, C., Guzdial, M., Chapman, G., & Astrachan, O. (2013). *Expanding access to K-12 computer science education: Research on the landscape of computer science professional development*. In Proceedings of the *44th ACM Technical Symposium on Computer Science Education*, *March, 6-9, 2013, Denver, Colorado, USA*.

Franklin, D., Conrad, P., Boe, B., Nilsen, K., Hill, C., Len, M., . . . Kiefer, B. (2013). Assessment of computer science learning in a scratch-based outreach program. In Proceedings of *the 44th ACM Technical Symposium on Computer Science Education,* March, 6-9, 2013, Denver, Colorado, USA

Franklin, D., Hill, C., Dwyer, H. A., Hansen, A. K., Iveland, A., & Harlow, D. B. (2016, March). *Initialization in Scratch: Seeking knowledge transfer*. In Proceedings of the *47th ACM Technical Symposium on Computing Science Education*, March 2-5, New York, NY.

Fox, A. R. C., & Wilson, E. G. (2015). Networking and the development of professionals: Beginning teachers building social capital. *Teaching and Teacher Education, 47*, 93-107.

Gal-Ezer, J., & Stephenson, C. (2009). The current state of computer science is U.S. high schools: A report from two national surveys. *Journal for Computing Teachers, 1*, 1-5.

Garet, M. S., Porter, A. C., Desimone, L., Birman, B. F., & Yoon, K. S. (2001). What makes professional development effective? Results from a national sample of teachers. *American Educational Research Journal, 38*(4), 915-945.

George, D., & Mallery, M. (2002). *Using SPSS for Windows step by step: A simple guide and reference*. Boston, MA: Allyn & Bacon.

Glaser, B. G. (1965). The constant comparative method of qualitative analysis. *Social Problems, 12*(4), 436-445.

Gliem, R. R., & Gliem, J. A. (2003, September). *Calculating, interpreting, and reporting cronbach's alpha reliability coefficient for likert-type scales*. Paper presented at the Midwest Research to Practice Conference in Adult, Continuing and Community Education, Columbus, OH.

Goode, J. (2008). Increasing diversity in K-12 computer science: Strategies from the field. *SIGCSE Bulletin, 40*(1), 362-366.

Goode, J., & Margolis, J. (2011). Exploring computer science: A case study of school reform. *ACM Transactions on Computing Education,11*(2), 12-28.

Google, & Gallup. (2015). Images of computer science: Perceptions among students, parents and educators in the U.S. Retrieved from https://services.google.com/fh/files/misc/images-of-computer-science-report.pdf

Google, & Gallup. (2016). *Trends in the state of computer science in U.S. K-12 schools*. Retrieved from http://services.google.com/fh/files/misc/trends-in-the-state-of-computer-science-report.pdf

Gray, L., Thomas, N., & Lewis, L. (2010). Teachers' use of educational technology in US public schools: 2009. *National Center for Education Statistics*. Retrieved from http://nces.ed.gov/pubs2010/2010040.pdf

Greene, J. C., Caracelli, V. J., & Graham, W. F. (1989). Toward a conceptual framework for mixed-method evaluation designs. *Educational Evaluation and Policy Analysis, 11*(3), 255-274.

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

Grover, S., Pea, R., & Cooper, S. (2016). Factors influencing computer science learning in middle school. In Proceedings of *the 47th ACM Technical Symposium on Computing Science Education*. March 02-05, 2016, Memphis, TN, USA

Groves, R. M., Fowler Jr, F. J., Couper, M. P., Lepkowski, J. M., Singer, E., & Tourangeau, R. (2013). *Survey methodology,* Hoboken, New Jersey: John Wiley & Sons.

Guba, E. G., & Lincoln, Y. S. (1982). The place of values in needs assessment. *Educational Evaluation and Policy Analysis*, *4*(3), 311-320.

Guisbond, L., & Neill, M. (2004). Failing our children: No Child Left Behind undermines quality and equity in education. *The Clearing House: A Journal of Educational Strategies, Issues and Ideas, 78*(1), 12-16.

Guzdial, M. (2003). A media computation course for non-majors. In Proceedings of *the 8th ACM Technical Symposium on Computing Science Education*. June 30-July 2, 2003, Thessaloniki, Greece.

Guzdial, M. (2013). Exploring hypotheses about media computation. In Proceedings of *the 9th Annual International ACM Conference on International Computing Education Research*. August 12-14, 2013, San Diego, California, USA.

Guzdial, M. (2014). The danger of requiring computer science in K-12 schools. Retrieved from http://cacm.acm.org/blogs/blog-cacm/173870-the-danger-of-requiring-computer-science-in-k-12-schools/fulltext

Guzdial, M., & Fisher, L. M. (2014). Teach the teachers, and contribute to humanity. *Communications of the ACM, 57*(11), 10-11.

Guzdial, M. (2015). Learner-centered design of computing education: Research on computing for everyone. *Synthesis Lectures on Human-Centered Informatics, 8*(6), 1-165.

Hagan, D., & Markham, S. (2000). Does it help to have some programming experience before beginning a computing degree program? *ACM SIGCSE Bulletin, 32*(3), 25-28.

Harkness, J. A, Mohler, P., & van de Vijver, F. J. R. (2002). *Cross-Cultural Survey Methods.* Hoboken, NJ: Wiley.

Hasselbring, T. S., & Glaser, C. H. W. (2000). Use of computer technology to help students with special needs. *The Future of Children*, *10*(2), 102-122.

Hew, K. F., & Brush, T. (2007). Integrating technology into K-12 teaching and learning: Current knowledge gaps and recommendations for future research. *Educational Technology Research and Development, 55*(3), 223-252.

Hew, K. F., & Hara, N. (2007). Empirical study of motivators and barriers of teacher online knowledge sharing. *Educational Technology Research and Development, 55*(6), 573-595.

Hmelo-Silver, C. E. (2004). Problem-based learning: What and how do students learn? *Educational Psychology Review, 16*(3), 235-266.

Hodhod, R., Khan, S., Kurt-Peker, Y., & Ray, L. (2016). *Training teachers to integrate computational thinking into K-12 teaching.* In Proceedings of *the 47th ACM Technical Symposium on Computing Science Education*. March 2-5, 2016, Memphis, Tenessee, USA.

Hollingworth, S., Mansaray, A., Allen, K., & Rose, A. (2011). Parents' perspectives on technology and children's learning in the home: Social class and the role of the habitus. *Journal of Computer Assisted Learning, 27*(4), 347-360.

Holzberger, D., Philipp, A., & Kunter, M. (2014). Predicting teachers' instructional behaviors: The interplay between self-efficacy and intrinsic needs. *Contemporary Educational Psychology, 39*(2), 100-111.

Horn, I. S. (2010). Teaching replays, teaching rehearsals, and re-visions of practice: Learning from colleagues in a mathematics teacher community. *Teachers College Record*, *112*(1), 225-259.

House, J. D. (2006). Mathematics beliefs and achievement of elementary school students in Japan and the United States: Results from the third international mathematics and science study. *The Journal of Genetic Psychology, 167*(1), 31-45.

Hsu, H.-M. J. (2013). Gender differences in elementary school students' game design preferences. *International Journal of Information and Education Technology, 3*(2), 172.

Huberman, M. (1989). The professional life cycle of teachers. *The Teachers College Record, 91*(1), 31-57.

Hug, S., Guenther, R., & Wenk, M. (2013). Cultivating a K12 computer science community: A case study. In Proceedings of *the 44th ACM Technical Symposium on Computer Science Education,* March, 6-9, 2013, Denver, Colorado, USA.

Hughes, J., & Kwok, O. (2007). Influence of student-teacher and parent-teacher relationships on lower achieving readers' engagement and achievement in the primary grades. *Journal of Educational Psychology, 99*(1), 39.

Hur, J. W., & Brush, T. A. (2009). Teacher participation in online communities: Why do teachers want to participate in self-generated online communities of K–12 teachers? *Journal of Research on Technology in Education, 41*(3), 279-303.

Hylén, J. (2006). Open educational resources: Opportunities and challenges. In Proceedings of Open Education, Septmber 27-29. 2006, Logan, Utah, USA.

Inan, F. A., & Lowther, D. L. (2010). Factors affecting technology integration in K-12 classrooms: A path model. *Educational Technology Research and Development, 58*(2), 137-154.

Isbell, C. L., Stein, L. A., Cutler, R., Forbes, J., Fraser, L., Impagliazzo, J., . . . Xu, Y. (2010). (Re) defining computing curricula by (re) defining computing. *ACM SIGCSE Bulletin, 41*(4), 195-207.

Israel, M., Pearson, J. N., Tapia, T., Wherfel, Q. M., & Reese, G. (2015). *Computers & Education, 82*, 263-279.

ISTE. (2011). *ISTE Standards Computer Science Educators*. Retrieved from http://www.iste.org/docs/pdfs/20-14_ISTE_Standards-CSE_PDF.pdf

Jackson, D. L., Starobin, S. S., & Laanan, F. S. (2013). The shared experiences: Facilitating successful transfer of women and underrepresented minorities in STEM fields. *New Directions for Higher Education, 2013*(162), 69-76.

Janke, S., Nitsche, S., & Dickhäuser, O. (2015). The role of perceived need satisfaction at work for teachers' work-related learning goal orientation. *Teaching and Teacher Education, 47*, 184-194.

Johnson, C. C. (2006). Effective professional development and change in practice: Barriers science teachers encounter and implications for reform. *School Science and Mathematics, 106*(3), 150-161.

Jones, C. S., & Orr, B. (1998). Computer-related musculoskeletal pain and discomfort among high school students. *American Journal of Health Studies, 14*(1), 26.

Jones, A., & Preece, J. (2006). Online communities for teachers and lifelong learners: A framework for comparing similarities and identifying differences in communities of practice and communities of interest. *International Journal of Learning Technology, 2*(2), 112-137.

Jones, W. M., & Dexter, S. (2014). How teachers learn: the roles of formal, informal, and independent learning. *Educational Technology Research and Development, 62*(3), 367-384.

Jordan, A. B., Hersey, J. C., McDivitt, J. A., & Heitzler, C. D. (2006). Reducing children's television-viewing time: A qualitative study of parents and their children. *Pediatrics, 118*(5), 1303-1310.

Jovés, P., Siqués, C., & Esteban-Guitart, M. (2015). The incorporation of funds of knowledge and funds of identity of students and their families into educational practice. A case study from Catalonia, Spain. *Teaching and Teacher Education, 49*, 68-77.

K12CS. (2016). *Announcing a new framework to define K–12 computer science education*. Retrieved from https://k12cs.org/wp-content/uploads/2016/09/K–12-Computer-Science-Framework.pdf

Kalelioğlu, F. (2015). A new way of teaching programming skills to K-12 students: Code. org. *Computers in Human Behavior*, *52*, 200-210.

Kalil, T., & Jahanian, F. (2013). *Computer science is for everyone!* Retrieved from https://www.whitehouse.gov/blog/2013/12/11/computer-science-everyone

Kavanagh, M. H., & Drennan, L. (2008). What skills and attributes does an accounting graduate need? Evidence from student perceptions and employer expectations. *Accounting & Finance, 48*(2), 279-300.

Kay, J., Barg, M., Fekete, A., Greening, T., Hollands, O., Kingston, J. H., & Crawford, K. (2000). Problem-based learning for foundation computer science courses. *Computer Science Education, 10*(2), 109-128.

Kelleher, C., Pausch, R., & Kiesler, S. (2007). *Storytelling alice motivates middle school girls to learn computer programming.* In Proceedings of *the SIGCHI Conference on Human factors in Computing Systems*. April 30-May 3, 2007, San Jose, California, USA.

Khoury, G. (2007). *Computer science state certification requirements: CSTA certification Committee report*. Retrieved from http://csta.acm.org/ComputerScienceTeacherCertification/sub/TeachCertRept07New.pdf

Kincheloe, J. L. (2012). *Teachers as researchers: Qualitative inquiry as a path to empowerment*. New York, NY, USA: Routledge.

Klassen, R. M., & Chiu, M. M. (2010). Effects on teachers' self-efficacy and job satisfaction: Teacher gender, years of experience, and job stress. *Journal of Educational Psychology, 102*(3), 741-756.

Knoeppel, R. C., & Rinehart, J. S. (2009). Student achievement and principal quality: Explaining the relationship. *Journal Of School Leadership, 18*(5), 501-527.

Köse, U. (2010). A web based system for project-based learning activities in "web design and programming" course. *Procedia-Social and Behavioral Sciences, 2*(2), 1174-1184.

Košir, K., Tement, S., Licardo, M., & Habe, K. (2015). Two sides of the same coin? The role of rumination and reflection in elementary school teachers' classroom stress and burnout. *Teaching and Teacher Education, 47*, 131-141.

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A Pedagogical framework for computational thinking. *Digital Experiences in Mathematics Education*, 1-18.

Krapp, A. (1999). Interest, motivation and learning: An educational-psychological perspective. *European Journal ofPpsychology of Education, 14*(1), 23-40.

Krapp, A. (2005). Basic needs and the development of interest and intrinsic motivational orientations. *Learning and Instruction, 15*(5), 381-395.

Lanas, M., & Kelchtermans, G. (2015). "This has more to do with who I am than with my skills" – Student teacher subjectification in Finnish teacher education. *Teaching and Teacher Education, 47*, 22-29.

Leagans, J. P. (1964). A concept of needs. *Journal of Cooperative Extension, 2*(2), 89-96.

Le, H., Robbins, S. B., & Westrick, P. (2014). Predicting student enrollment and persistence in college STEM fields using an expanded PE fit framework: A large-scale multilevel study. *Journal of Applied Psychology*, *99*(5), 915.

Lee, H. J. (2005). Developing a professional development program model based on teachers' needs. *Professional Educator, 27*, 39-49.

Lewis, A., & Smith, D. (1993). Defining higher order thinking. *Theory into Practice, 32*(3), 131-137.

Lichtenberger, E., & George-Jackson, C. (2012). Predicting high school students' interest in majoring in a STEM field: Insight into high school students' postsecondary plans. *Journal of Career and Technical Education, 28*(1), 19-38.

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic inquiry*. Thousand Oaks, CA: Sage Publications.

Lockard, C. B., & Wolf, M. (2012). Occupational employment projections to 2020. *Monthly Labor Review, 135*(1), 84–108.

Luxton-Reilly, A., & Denny, P. (2010). Constructive evaluation: A pedagogy of student-contributed assessment. *Computer Science Education, 20*(2), 145-167.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51-61.

Ma, Y. (2009). Family socioeconomic status, parental involvement, and college major choices—gender, race/ethnic, and nativity patterns. *Sociological Perspectives, 52(*2), 211-234.

Mallow, J., Kastrup, H., Bryant, F. B., Hislop, N., Shefner, R., & Udo, M. (2010). Science anxiety, science attitudes, and gender: Interviews from a binational study. *Journal of Science Education and Technology*, *19*(4), 356-369.

Margolis, J. (2008). *Stuck in the shallow end: Education, race, and computing.* Cambridge, MA: The MIT Press.

McDowell, C., Werner, L., Bullock, H., & Fernald, J. (2002). The effects of pair-programming on performance in an introductory programming course. *ACM SIGCSE Bulletin. 34(*1), 38-42.

McDowell, C., Werner, L., Bullock, H. E., & Fernald, J. (2006). Pair programming improves student retention, confidence, and program quality. *Communications of the ACM, 49*(8), 90-95.

Meece, J. L., Wigfield, A., & Eccles, J. S. (1990). Predictors of math anxiety and its influence on young adolescents' course enrollment intentions and performance in mathematics. *Journal of Educational Psychology, 82*(1), 60.

Meerbaum-Salant, O., Armoni, M., & Ben-Ari, M. (2013). Learning computer science concepts with Scratch. *Computer Science Education, 23*(3), 239-264.

Melnick, S. A., & Meister, D. G. (2008). A Comparison of beginning and experienced teachers' concerns. *Educational Research Quarterly, 31*(3), 39-56.

Menekse, M. (2015). Computer science teacher professional development in the United States: A review of studies published between 2004 and 2014. *Computer Science Education*, *25*(4), 325-350.

Mills, J. E., & Treagust, D. F. (2003). Engineering education—Is problem-based or project-based learning the answer. *Australasian Journal of Engineering Education, 3*(2), 2-16.

Mølstad, C. E. (2015). State-based curriculum-making: Approaches to local curriculum work in Norway and Finland. *Journal of Curriculum Studies, 47*(4), 441-461.

Moore, K. D. (1977). Development and validation of a science teacher needs-assessment profile. *Journal of Research in Science Teaching, 14*(2), 145-149.

Moore, K. D., & Hanley, P. E. (1982). An identification of elementary teacher needs. *American Educational Research Journal, 19*(1), 137-144.

Moorefield-Lang, M. H. (2014). Makers in the library: Case studies of 3D printers and maker spaces in library settings. *Library Hi Tech, 32*(4), 583-593.

Moorman, P., & Johnson, E. (2003). Still a stranger here: attitudes among secondary school students towards computer science. *SIGCSE Bull., 35*(3), 193-197.

Moskal, B., Miller, K., & King, L. A. (2002). Grading essays in computer ethics: Rubrics considered helpful. *ACM SIGCSE Bulletin. 34*(1), 101-105.

Nagappan, N., Williams, L., Ferzli, M., Wiebe, E., Yang, K., Miller, C., & Balik, S. (2003). Improving the CS1 experience with pair programming. *ACM SIGCSE Bulletin, 35*(1), 359-362.

NCWIT. (2016). Pacesetters program. Retrieved from https://www.ncwit.org/programs-campaigns/pacesetters

Next Generation Science Standards. (2013). *Get to know the standards*. Retrieved from http://www.nextgenscience.org/get-to-know

Nicol, D. J., & Macfarlane-Dick, D. (2006). Formative assessment and self-regulated learning: A model and seven principles of good feedback practice. *Studies in Higher Education, 31*(2), 199-218.

Noack-Cooper, K. L., Sommerich, C. M., & Mirka, G. A. (2009). College students and computers: assessment of usage patterns and musculoskeletal discomfort. *Work, 32*(3), 285-298.

Novak, J. D. (2002). Meaningful learning: The essential factor for conceptual change in limited or inappropriate propositional hierarchies leading to empowerment of learners. *Science Education, 86*(4), 548-571.

NSSE. (2016). *Engagement insights: Survey findings on the quality of undergraduate education.* Retrieved from http://nsse.indiana.edu/NSSE_2016_Results/pdf/NSSE_2016_Annual_Results.pdf

NSF. (2015). *Women, minorities, and persons with disabilities in science and engineering: 2015.* Retrieved from https://www.nsf.gov/statistics/2015/nsf15311/digest/nsf15311-digest.pdf

OECD. (2005). *Teachers matter: Attracting, develping, and retaining effective teachers.* Retrieved from http://www.oecd.org/education/school/34990905.pdf

Ogan-Bekiroglu, F. (2007). Bridging the gap: needs assessment of science teacher in-service education in Turkey and the effects of teacher and school demographics. *Journal of Education for Teaching, 33*(4), 441-456.

Ong, M., Wright, C., Espinosa, L. L., & Orfield, G. (2011). Inside the double bind: A synthesis of empirical research on undergraduate and graduate women of color in science, technology, engineering, and mathematics. *Harvard Educational Review, 81*(2), 172-209.

Overmars, M. (2004). Teaching computer science through game design. *Computer, 37*(4), 81-83.

Owston, R., Wideman, H., Murphy, J., & Lupshenyuk, D. (2008). Blended teacher professional development: A synthesis of three program evaluations. *The Internet and Higher Education*, *11*(3), 201-210.

Oyewole, S. A., Haight, J. M., & Freivalds, A. (2010). The ergonomic design of classroom furniture/computer work station for first graders in the elementary school. *International Journal of Industrial Ergonomics, 40*(4), 437-447.

Papastergiou, M. (2009). Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation. *Computers & Education, 52*(1), 1-12.

Partovi, H. (2015). A comprehensive effort to expand access and diversity in computer science. *ACM Inroads, 6*(3), 67-72.

Pashler, H., McDaniel, M., Rohrer, D., & Bjork, R. (2008). Learning styles concepts and evidence. *Psychological Science in the Public Interest, 9*(3), 105-119.

Patton, M. Q. (2015). *Qualitative research & evaluation methods: Integrating theory and practice.* Thousand Oaks, CA: Sage Publications.

Paulson, S. E., & Marchant, G. J. (2009). Background variables, levels of aggregation, and standardized test scores. *Education Policy Analysis Archives, 17*(22).

Pears, A., Seidman, S., Malmi, L., Mannila, L., Adams, E., Bennedsen, J., Paterson, J. (2007). A survey of literature on the teaching of introductory programming. *ACM SIGCSE Bulletin, 39*(4), 204-223

Pehmer, A.-K., Gröschner, A., & Seidel, T. (2015). How teacher professional development regarding classroom dialogue affects students' higher-order learning. *Teaching and Teacher Education, 47*, 108-119.

Perkins, D. N., & Salomon, G. (1992). Transfer of learning. *International Encyclopedia of Education, 2*, 6452-6457.

PLTW. (2016). *PLTW computer science - curriculum*. Retrieved from https://www.pltw.org/our-programs/pltw-computer-science/pltw-computer-science-curriculum

Porter, L., Lee, C. B., & Simon, B. (2013). *Halving fail rates using peer instruction: A study of four computer science courses.* In Proceedings of *the 44th ACM Technical Symposium on Computer Science Education.* March 6-9, 2013, Denver, CO, USA.

Powers, A. L. (2004). An evaluation of four place-based education programs. *The Journal of Environmental Education, 35*(4), 17-32.

Pretto, G. (2011). Pedagogy and learning spaces in IT. *Global Learn, 2011*(1), 702-711.

Prey, J. C., & Weaver, A. C. (2013). Fostering gender diversity in computing. *Computer, 46*(3), 22-23..

Printy, S. (2010). Principals' influence on instructional quality: Insights from US schools. *School Leadership and Management, 30*(2), 111-126.

Raes, A., Schellens, T., De Wever, B., & Vanderhoven, E. (2012). Scaffolding information problem solving in web-based collaborative inquiry learning. *Computers & Education, 59*(1), 82-94.

Ragonis, N., Hazzan, O., & Gal-Ezer, J. (2010). A survey of computer science teacher preparation programs in Israel tells us: computer science deserves a designated high school teacher preparation! In Proceedings of *the 41st ACM Technical Symposium on Computer Science Education*. March 10-13, 2010, Milwaukee, WI, USA.

Ramalingam, V., LaBelle, D., & Wiedenbeck, S. (2004). Self-efficacy and mental models in learning to program. *ACM SIGCSE Bulletin, 36*(3), 171-175.

Ramirez, G., Gunderson, E. A., Levine, S. C., & Beilock, S. L. (2013). Math anxiety, working memory, and math achievement in early elementary school. *Journal of Cognition and Development*, *14*(2), 187-202.

Reid, J. M. (1987). The learning style preferences of ESL students. *TESOL Quarterly, 21*(1), 87-111.

Reid, S. A. (2007). *An examination of the role of teacher perceptions of their professional development needs in the professional development process.* (Doctoral Dissertation). Retrieved from ProQuest Dissertations & Theses Global. (Order No. 304875494)

Reigeluth, C. M. (2011). An instructional theory for the post-industrial age. *Educational Technology*, *51*(5), 25.

Rice, L., Barth, J. M., Guadagno, R. E., Smith, G. P. A., & McCallum, D. M. (2013). The role of social support in students' perceived abilities and attitudes toward math and science. *Journal of Youth and Adolescence, 42*(7), 1028-1040.

Robelen, E. (2013). Computer science teacher certification 'deeply flawed,' report says. Retrieved from http://blogs.edweek.org/edweek/curriculum/2013/08/computer_science_teacher_certi.htm

Roberts, N. S., & Suren, A. T. (2010). Effects of an environmental education program on the environmental orientations of children from different gender, age, and ethnic groups. *Journal of Park and Recreation Administration*, *28*(4), 59-80.

Rockoff, J. E. (2004). The impact of individual teachers on student achievement: Evidence from panel data. *The American Economic Review, 94*(2), 247-252.

Rogers, M. P., Abell, S., Lannin, J., Wang, C. Y., Musikul, K., Barker, D., & Dingman, S. (2007). Effective professional development in science and mathematics education: Teachers' and facilitators' views. *International Journal of Science and Mathematics Education*, *5*(3), 507-532.

Rusk, N., Resnick, M., Berg, R., & Pezalla-Granlund, M. (2008). New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology, 17*(1), 59-69.

Sadık, O. (2015). Encouraging Women to Become CS Teachers. In Proceedings of *the Third Conference on GenderIT*, Philadelphia, PA, USA.

Sadık, O., Leftwich, A. O., & Nadiruzzaman, H. (2017). Computational thinking conceptions and misconceptions: Progression of preservice teacher thinking during computer science lesson planning. In P. T. Rich and C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 221-238). Cham, Switzerland: Springer.

Sadler, D. R. (1989). Formative assessment and the design of instructional systems. *Instructional Science, 18*(2), 119-144.

Salkovsky, M., Romi, S., & Lewis, R. (2015). Teachers' coping styles and factors inhibiting teachers' preferred classroom management practice. *Teaching and Teacher Education, 48*, 56-65.

Sancho-Thomas, P., Fuentes-Fernández, R., & Fernández-Manjón, B. (2009). Learning teamwork skills in university programming courses. *Computers & Education, 53*(2), 517-531.

Sanderson, P. (2003). Where's (the) computer science in service-learning?. *Journal of Computing Sciences in Colleges*, *19*(1), 83-89.

Saye, J. W., & Brush, T. (2002). Scaffolding critical reasoning about history and social issues in multimedia-supported learning environments. *Educational Technology Research and Development, 50*(3), 77-96.

Schlager, M. S., & Fusco, J. (2003). Teacher professional development, technology, and communities of practice: Are we putting the cart before the horse? *The Information Society, 19*(3), 203-220.

Seehorn, D. W., Stephenson, C., Pirmann, T. R., & Powers, K. (2013). *CSTA CS K-12 instructional standards and CS curriculum.* In Proceeding of the *44th ACM Technical Symposium on Computer Science Education*. March 6-9, 2013, Denver, CO, USA

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.

Sharma, P., & Hannafin, M. J. (2007). Scaffolding in technology-enhanced learning environments. *Interactive learning environments*, *15*(1), 27-46.

Shimazoe, J., & Aldrich, H. (2010). Group work can be gratifying: Understanding & overcoming resistance to cooperative learning. *College Teaching, 58*(2), 52-57.

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher*, *15*(2), 4-14.

Shumow, L., Lyutykh, E., & Schmidt, J. A. (2011). Predictors and outcomes of parental involvement with high school students in science. *School Community Journal, 21*(2), 81-98.

Simon, B., Kohanfars, M., Lee, J., Tamayo, K., & Cutts, Q. (2010). Experience report: Peer instruction in introductory computing. In Proceedings of *the 41st ACM Technical Symposium on Computer Science Education*. March 10-13, 2010, Milwaukee, Wisconsin, USA.

Smit, J., AA van Eerde, H., & Bakker, A. (2013). A conceptualisation of whole-class scaffolding. *British Educational Research Journal*, *39*(5), 817-834.

Smith, G. A. (2002). Place-based education. *Phi Delta Kappan, 83*(8), 584.

Smith, M. (2016). *Computer science for all.* Retrieved from https://www.whitehouse.gov/blog/2016/01/30/computer-science-all

Snider, V. E., & Roehl, R. (2007). Teachers' beliefs about pedagogy and related issues. *Psychology in the Schools, 44*(8), 873-886.

Stephenson, C., Gal-Ezer, J., Haberman, B., & Verno, A. (2005). *The new educational imperative: Improving high school computer science education*. Retrieved from https://csta.acm.org/Communications/sub/DocsPresentationFiles/White_Paper07_06.pdf

Stephenson, C. (2014). Working together to support computer science education. Retrieved from http://googleforeducation.blogspot.com/2014/09/working-together-to-support-computer.html

Sykora, C. (2014). *Computational thinking for all.* Retrieved from https://www.iste.org/explore/articledetail?articleid=152

Teddlie, C., & Tashakkori, A. (2009). *Foundations of mixed methods research: Integrating quantitative and qualitative approaches in the social and behavioral sciences*. In A. Tashakkori (Ed.). Los Angeles, California, USA: Sage Publications.

The American National Election Studies. (2012). People Don't Have a Say in What the Government Does 1952-2012. Retrieved from http://www.electionstudies.org/nesguide/toptable/tab5b_2.htm

Tiwari, A., Lai, P., So, M., & Yuen, K. (2006). A comparison of the effects of problem-based learning and lecturing on the development of students' critical thinking. *Medical Education, 40*(6), 547-554.

Tsiotakis, P., & Jimoyiannis, A. (2013). Developing a computer science teacher community in Greece: Design framework and implications from the pilot. In Proceedings of the EduLearn13. July 1-3, 2013, Barcelona, Spain.

Tucker, A., Deek, F., Jones, J., McCowan, D., Stephenson, C., & Verno, A. (2003). *A model curriculum for K–12 computer science.* Retrieved from https://csta.acm.org/Curriculum/sub/CurrFiles/K-12ModelCurr2ndEd.pdf

Umbleja, K. (2016). Can K-12 students learn how to program with just two hours? In L. Uden, D. Liberona, & B. Feldmann (Eds.), *Learning Technology for Education in Cloud – The Changing Face of Education* (pp. 250-264). Cham, Switzerland: Springer International Publishing.

Vähäsantanen, K. (2015). Professional agency in the stream of change: Understanding educational change and teachers' professional identities. *Teaching and Teacher Education, 47*, 1-12.

Vars, G. F. (2001). Can curriculum integration survive in an era of high-stakes testing? *Middle School Journal, 33*(2), 7-17.

Veletsianos, G., Beth, B., & Lin, C. (2016, February). CS teacher experiences with educational technology, problem-based learning, and a CS principles curriculum. In Proceedings of the 47th ACM Technical Symposium on Computing Science Education. March 2-5, 2016, New York, NY.

Wachira, P., & Keengwe, J. (2011). Technology integration barriers: Urban school mathematics teachers perspectives. *Journal of Science Education and Technology, 20*(1), 17-25.

Wang, J. (2014). *Increasing ACCESS to CS Education*. Retrieved from http://googleforeducation.blogspot.com/2014/12/increasing-access-to-cs-education.html

Wang, R., Bianchi, S. M., & Raley, S. B. (2005). Teenagers' Internet use and family rules: A research note. *Journal of Marriage and Family, 67*(5), 1249-1258.

Webel, C., & Platt, D. (2015). The role of professional obligations in working to change one's teaching practices. *Teaching and Teacher Education, 47*, 204-217.

Weber, R. P. (1990). *Basic content analysis*. Newbury Park, CA: Sage.

Williams, L. A., & Kessler, R. R. (2000). All I really need to know about pair programming I learned in kindergarten. *Communications of the ACM, 43*(5), 108-114.

Williams, L., Wiebe, E., Yang, K., Ferzli, M., & Miller, C. (2002). In support of pair programming in the introductory computer science course. *Computer Science Education, 12*(3), 197-212.

Wills, J. S., & Sandholtz, J. H. (2009). Constrained professionalism: Dilemmas of teaching in the face of test-based accountability. *Teachers College Record, 111*(4), 1065-1114.

Wilson, B. C., & Shrock, S. (2001). Contributing to success in an introductory computer science course: a study of twelve factors. *ACM SIGCSE Bulletin, 33*(1), 184-188.

Wilson, C., Sudol, L., Stephenson, C., & Stehlik, M. (2010). *Running on empty: Failure to teach K-12 computer science in the digital age*. Retrieved from Association for Computing Machinery website: http://runningonempty.acm.org/fullreport2.pdf

Wilson, C. (2014). Hour of code: We can solve the diversity problem in computer science. *ACM Inroads, 5*(4), 22-22.

Wing, J. M. (2008). Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 366*(1881), 3717-3725.

Wood, D., Bruner, J. S., & Ross, G. (1976). The role of tutoring in problem solving. *Journal of Child Psychology and Psychiatry, 17*(2), 89-100.

Wood, D. F. (2003). Problem based learning. *British Medical Journal, 326*(7384), 328.

Woodward, J., & Brown, C. (2006). Meeting the curricular needs of academically low-achieving students in middle grade mathematics. *The Journal of Special Education*, *40*(3), 151-159.

Yadav, A., Subedi, D., Lundeberg, M. A., & Bunting, C. F. (2011). Problem-based learning: Influence on students' learning in an electrical engineering course. *Journal of Engineering Education, 100*(2), 253-280.

Yadav, A., Burkhart, D., Moix, D., Snow, E., Bandaru, P., & Clayborn, L. (2015). Sowing the Seeds: A landscape study on assessment in secondary computer science education. *Computer Science Teachers Association*, New York, NY, Retrieved from https://csta.acm.org/Research/sub/Projects/ResearchFiles/AssessmentStudy2015.pdf

Young, A., & Huggard, G. (2003). Designing new computer laboratories: Fresh ideas and new layouts. In Proceedings of *the 16th Annual National Advisory Committee on Computing Qualifications Conference*. July 2003, New Zealand.

Zakaria, E., & Daud, M. Y. (2009). Assessing mathematics teachers' professional development needs. *European Journal of Social Sciences*, *8*(2), 225-231.

Zhang, D., Liu, Y., M'Hallah, R., & Leung, S. C. H. (2010). A simulated annealing with a new neighborhood structure based algorithm for high school timetabling problems. *European Journal of Operational Research, 203*(3), 550-558.

Zhang, M. (2015). Understanding the relationships between interest in online math games and academic performance. *Journal of Computer Assisted Learning, 31*(3), 254-267.

Zhang, M., Parker, J., Koehler, M. J., & Eberhardt, J. (2015). Understanding inservice science teachers' needs for professional development. *Journal of Science Teacher Education*, *26*(5), 471-496.

APPENDICES

APPENDIX A: Preliminary Framework Categories

Overview of the Teacher Related Needs (ISTE, 2011, pp. 1-2)

Need Categories	Themes			
Knowledge of content	Demonstrate knowledge of and proficiency in data representation and abstraction	Effectively design, develop, and test algorithms	Demonstrate knowledge of digital devices, systems, and networks	Demonstrate an understanding of the role computer science plays and its impact in the modern world
Effective teaching and learning strategies	Knowledge of instructional strategies and materials	Knowledge of learners	Curricular knowledge	Assessment knowledge
Effective learning environments	Effective use of computer systems and their peripherals	Equitable and accessible resources for all the students		
Effective professional knowledge and skills	Knowledge of organizations and resources for professional development	Knowledge and application of new research to teaching CS	Knowledge on secondary CS education standards	

Overview of School Related Needs

Categories	Themes	Citations
School Demographics	Large class size, issues related to diversity (Race, gender) and socio-economic status	Allen, 2015; Belfi, Gielen, De Fraine, Verschueren, & Meredith, 2015; Eamon, 2005; Melnick & Meister, 2008; S. A. Reid, 2007; Snider & Roehl, 2007
Resources	Lack of availability and access to resources (teaching materials, tools, funding etc.), lack of time/high workload, lack of PD	Betoret, 2009; Burke et al., 2015; Ertmer, Ottenbreit-Leftwich, Sadık, Sendurur, & Sendurur, 2012; Hew & Brush, 2007; Ogan-Bekiroglu, 2007; Snider & Roehl, 2007; Wachira & Keengwe, 2011
School Planning	School policy related problems, timetable related problems	Beligiannis et al., 2008; Hew & Brush, 2007; Zhang, Liu, M'Hallah, & Leung, 2010
Parents	Lack of cultural capital (parent knowledge and skills, education, access to technology and educational resources)	Allen, 2015; Eamon, 2005; Hughes & Kwok, 2007; Jovés, Siqués, & Esteban-Guitart, 2015; Paulson & Marchant, 2009; Shumow, Lyutykh, & Schmidt, 2011
Colleagues	Lack of support, collaboration and community	Burke et al., 2015; De Neve et al., 2015; Ertmer et al., 2012; Fox & Wilson, 2015; Salkovsky, Romi, & Lewis, 2015
Administration	Lack of support, negative attitude, and lack of knowledge about a field	Burke et al., 2015; Hew & Brush, 2007; Knoeppel & Rinehart, 2009; Printy, 2010; Salkovskyy et al., 2015

APPENDIX B: Questionnaire Invitation to the CSTA Members

Dear CS Teacher [insert their names],

As part of CSTA's commitment to encourage more researchers to investigate topics that are relevant to K-12 CS teachers, we agreed to provide occasional support for doctoral students at the dissertation phase of their programs. This message provides an invitation for you to participate in one of those studies. This study proposes some interesting questions related to the needs and experiences of computer science teachers. Results from the study will be shared with CSTA members and may help inform some of the new programs and benefits CSTA is creating. Participation is completely voluntary and your responses will be anonymous. We have not shared or disclosed any member's personal contact information with the researcher.

The study is being conducted by Olgun Sadık, a Ph.D. student at Indiana University and a former high school computer science (CS) teacher. For his dissertation research, he is examining secondary CS teachers' current needs and would like to invite teacher members of CSTA to participate a short survey as part of a larger scale study. The survey will take less than 10 minutes to complete and your responses will be anonymous. You may optionally provide your contact information if you would like to enter a drawing for one of the three $50 Amazon Gift certificates or to participate in a follow up interview. Take the survey or copy and paste the URL below into your Internet browser:

https://iu.co1.qualtrics.com/SE/?SID=SV_3kQtuWbfbS4nXCZ

APPENDIX C: Questionnaire

Default Question Block

Secondary Computer Science (CS) Teachers' Current Needs

You are invited to participate in a research study focusing on secondary computer science teachers' current needs for teaching CS courses in their schools or programs. You were selected as a possible participant because you may be teaching a CS course (or CS courses) for secondary level students (between grades 6 and 12) in a school or informal setting (e.g., after school program, science center, camp, etc.). We ask that you read this form and ask any questions you may have before agreeing to be in the study. The study is being conducted by Dr. Tom Brush, Professor of Instructional Systems Technology (IST) Department at Indiana University (IU) and Olgun Sadik, Ph.D. candidate in the IST Program at IU.

STUDY PURPOSE
The purpose of this study is to examine secondary level computer teachers' needs for effective computer science education in the U.S. and how these needs change based on years of experience and background in computer science.

PROCEDURES FOR THE STUDY:
If you agree to be in the study, you will complete a questionnaire to understand your needs. At the end of the questionnaire, we will also request to voluntarily schedule a convenient meeting time to interview you and investigate your needs with regards to teaching computer science courses. The questionnaire is expected to take 5-8 minutes. Voluntary interview will last around 30 minutes, and will be held by phone or online depending on your convenience. Interviews will be audio recorded and only the key researchers, Dr. Tom Brush and Olgun Sadik will have access to the data.

CONFIDENTIALITY
Efforts will be made to keep your personal information confidential. We cannot guarantee absolute confidentiality. Your personal information may be disclosed if required by law. Your identity will be held in confidence in reports in which the study may be published. The questionnaire results will be kept in a password protected online database. Interviews will be audio recorded and the audio recordings will be transcribed. The audio recordings will be deleted after the transcriptions. The transcriptions will be stored in password protected online database (box.iu.edu). Only key researchers will have access to the transcriptions. The research report will not include any identical information about you, and your name will be changed with pseudonyms in the research report. Organizations that may inspect and/or copy your research records for quality assurance and data analysis include groups such as the study investigator and his/her research associates, the Indiana University Institutional Review Board or its designees, and (as allowed by law) state or federal agencies, specifically the Office for Human Research Protections (OHRP) etc., who may need to access your research records.

PAYMENT
You will not receive payment for taking part in this study. However, all participants will choose to enter a raffle to win one of the three 50 dollars Amazon gift cards.

CONTACTS FOR QUESTIONS OR PROBLEMS
For questions about the study, contact the researcher Olgun Sadik from olsadik@indiana.edu.
For questions about your rights as a research participant or to discuss problems, complaints or concerns about a research study, or to obtain information, or offer input, contact the IU Human Subjects Office at (812) 856-4242 or (800) 696-2949.

VOLUNTARY NATURE OF STUDY
Taking part in this study is voluntary. You may choose not to take part or may leave the study at any time. Leaving the study will not result in any penalty or loss of benefits to which you are entitled. Your decision whether or not to participate in this study will not affect your current or future relations with any person involved to this research or with the School of Education and Indiana University.

Please click the forward arrow to start the survey

Page 1/10

Participation Agreement

I currently teach computer science (CS) course/courses or content* for secondary level students (grades 6 to 12)** and agree to participate to this research.

* Programming, AP computer science, computer hardware, robotics, app development, engineering education, computational thinking, game design, computer networks, data structures, algorithms, operating systems, role of computer science and impact in modern world, and similar.

** Teaching computers in a formal class, after school activity, science center, and/or similar setting.

- ○ Yes
- ○ No

Think about your current situation in which you are teaching CS. What are the things you need to make yourself an even better CS educator? Keep this in mind as you respond to the following survey items.

Page 2/10

Pedagogical Needs

Below is a list of pedagogical needs teachers have identified.

Please indicate how much you consider these as a need to support your current teaching.

1: This is not a need for me
2: This is somewhat of a need for me
3: This is a need for me
4: This is a strong need for me

	1	2	3	4	Not applicable

I need to learn new instructional strategies (e.g. problem-based learning, pair programming) for teaching CS	○	○	○	○	○
I need to learn how to teach computational thinking (e.g. problem solving, designing solutions) in my CS class(es)	○	○	○	○	○
I need to learn strategies to help students transfer their learning between programming languages and platforms	○	○	○	○	○
I need to learn how to answer my students' simultaneous questions while coding (e.g. finding errors in code, debugging, reading code)	○	○	○	○	○
I need to learn how to facilitate my students' interaction and collaboration between each other in my CS class(es)	○	○	○	○	○

Please share your comments or other pedagogical needs here

Page 3/10

Curricular Needs

Below is a list of curricular needs teachers have identified.

Please indicate how much you consider these as a need to support your current teaching.

1: This is not a need for me
2: This is somewhat of a need for me
3: This is a need for me
4: This is a strong need for me

	1	2	3	4	Not applicable
I need a fully designed and ready to implement CS curriculum for a specific grade level (or for specific grade levels)	○	○	○	○	○
I need curriculum resources (e.g. practice questions, assignments, activities, project samples) for my CS class(es)	○	○	○	○	○
I need help to make a decision for selecting an appropriate level of programming tool or language for my CS class(es)	○	○	○	○	○
I need help embedding the principles of computational thinking into my CS curriculum	○	○	○	○	○
I need materials to assess my students' learning in my CS classes	○	○	○	○	○

Please share your comments or other curricular needs here

Page 4/10

Student Related Needs

Below is a list of needs teachers have identified related to students.

Please indicate how much you consider these as a need to support your current teaching.

1: This is not a need for me
2: This is somewhat of a need for me
3: This is a need for me
4: This is a strong need for me

	1	2	3	4	Not applicable
I need to learn strategies to increase student enrollment in my CS classes	○	○	○	○	○
I need to learn strategies to motivate girls to enroll in my CS classes	○	○	○	○	○
I need to learn strategies to motivate underrepresented populations (African-American, Hispanic) to enroll in my CS classes	○	○	○	○	○
I need to learn strategies to teach students with little to no interest in CS	○	○	○	○	○
I need to learn strategies to teach students that have limited background in math (e.g. algebra)	○	○	○	○	○
I need to learn strategies to teach students with limited reading comprehension	○	○	○	○	○

Please share your comments or other student related needs here

Page 5/10

Professional Knowledge and Skill Needs

Below is a list of needs teachers have identified related to their knowledge of professional organizations and research in CS education.

Please indicate how much you consider these as a need to support your current teaching.

1: This is not a need for me
2: This is somewhat of a need for me
3: This is a need for me
4: This is a strong need for me

	1	2	3	4	Not applicable
I need to continue attending professional development events to update my CS education knowledge and skills	○	○	○	○	○
I need to learn the higher education institutions that offer professional development for teaching CS	○	○	○	○	○
I need to learn current research that explores teachers' best practices teaching CS	○	○	○	○	○
I need to learn about the changes in CS education national and state level standards	○	○	○	○	○

Please share your comments or other professional knowledge needs here

Page 6/10

Resource Needs

Below is a list of resources teachers identified as a need for teaching CS.

Please indicate how much you consider these as a need to support your current teaching.

1: This is not a need for me
2: This is somewhat of a need for me
3: This is a need for me
4: This is a strong need for me

	1	2	3	4	Not Applicable
I need a computer laboratory designed for the purpose of teaching CS classes (reliable computers, furniture design)	○	○	○	○	○
I need software (e.g. IDE, design software) that allow me to conduct CS practices in my class	○	○	○	○	○

I need a reliable computer network (wifi, wired) in the school	○	○	○	○	○
I need funding to provide resources and tools for my CS class(es)	○	○	○	○	○
I need more time to prepare for my CS classes	○	○	○	○	○

Please share your comments or other resource needs here

Page 7/10

Stakeholders Related Needs

Below is a list of stakeholders identified as important in successful CS education in K12 schools.

Please indicate how much you consider these as a need to support your current teaching.

1: This is not a need for me
2: This is somewhat of a need for me
3: This is a need for me
4: This is a strong need for me

	1	2	3	4	Not Applicable
I need my administrator to better understand what CS education is (e.g. the difference between information technology, computer science, technology integration)	○	○	○	○	○
I need my administrator to support my CS education efforts in the school/program	○	○	○	○	○
I need access to an online community of CS teachers (e.g. collaboration, sharing ideas, receiving feedback)	○	○	○	○	○
I need my students' parents to better understand what CS education is	○	○	○	○	○
I need my students' parents to get involved in CS education efforts in the school/program	○	○	○	○	○

Please share your comments or other stakeholders related needs here.

Demographic Questions

Page 8/10

What is your gender?

- ○ Female
- ○ Male
- ○ Other
- ○ Decline to answer

How many years have you been teaching CS content?

- ○ Less than one year
- ○ 1-3 years
- ○ 4-5 years
- ○ 6-10 years
- ○ More than 10 years

Please choose the education/experience you have gained that prepared you teach CS in your school/program (Please select all that apply)

- ☐ degree or licensure in CS education (undergraduate, graduate, licensure or endorsement)
- ☐ degree in CS or a related field (e.g. minor, major, undergraduate, graduate)
- ☐ Attended professional development workshops/courses (face-to-face and/or online)
- ☐ CS work experience in private sector

Please list any other education or experience you may want to add.

In what state do you currently teach CS?
If you are teaching outside the United States, please choose "I do not teach in the United States "

In which country do you teach CS?

Page 9/10

Where do you teach CS?
(Please select all the apply)

- [ ] Public school
- [ ] Private school
- [ ] Charter school
- [ ] Technology center, after school program, science center and similar
- [ ] Other

What level students do you teach CS?
(Please select all that apply)

- [ ] Elementary (K to 5th grade)
- [ ] Middle school (6th to 8th grade)
- [ ] High school (9th to 12th grade)

What CS content do you teach currently?
(Please select all that apply)

If not listed, please type your course name in the "Other" category.

- [ ] Computer Programming
- [ ] AP Computer Science
- [ ] Computer Hardware
- [ ] Robotics
- [ ] Mobile App Development
- [ ] Game Design/Development
- [ ] Computer Networks
- [ ] Data Structures

- [ ] Algorithms
- [ ] Operating Systems
- [ ] Role of CS and impact in modern world
- [ ] Computer Applications
- [ ] Other

Page 10/10

Would you like your name to be entered into the Amazon gift card raffle?

- ○ Yes
- ○ No

Would you like to participate to a voluntary follow-up phone or online interview to discuss your responses?

- ○ Yes
- ○ No

Please complete the details below if you agree to participate in the raffle or the interview.

You will not be contacted if you do not win the raffle or agree to participate in the interview.

Full Name	
Email Address	
Phone	

Please click the forward arrow to complete the survey and submit your responses.

APPENDIX D: The participants' CS backgrounds in frequencies

Background	Frequency (#)	Frequency (%)
degree ONLY in CS or a related field	7	3.2
degree ONLY in CS education or a related field	10	4.5
attended ONLY professional development	47	21.2
had ONLY CS work experience	6	2.7
background in cs and cs education	1	0.5
background in cs and professional development	26	11.7
background in cs and CS work experience	11	5.0
background in cs education and professional development	9	4.1
background in cs education and CS work experience	2	0.9
background in professional development and CS work experience	11	5.0
background in cs, cs education and professional development	18	8.1
background in cs, cs education and CS work experience	1	0.5
background in cs, professional development and CS work experience	31	14.0
background in cs education, professional development and CS work experience	12	5.4
background in all	25	11.3

APPENDIX E: Questionnaire participants' background in CS (quotes)

Degree in CS or a related field (e.g., minor, major, undergraduate, graduate)	• Masters in CS • BS in Engineering, 21 years computer programming & IT • 2 degrees in Electrical Engineering, 30 hours of graduate studies in CS • I had undergrad and grad degrees in engineering • Minor in Computer Science • I have a PhD in computer science • Graduated with BS in computer science and engineering • BS in Computer Science
Degree or licensure in CS education (e.g., undergraduate, graduate, licensure or endorsement)	• I do hold a teaching certificate in Computer Science • Technology Educator • Masters in Education, Masters in Instructional Technology • Math teaching license Computer / Technology Endorsement (Working on 2nd M Ed, Classroom Technology)
Attended professional development workshops/courses (face-to-faceand/or online)	• Took PLTW [Project Lead the Way]- year one training CSE Took PLTW- year two training CSE. Various conferences and PD through Indiana Dept. of Education over the years • PLTW training in ICS and CSE • Professional development specifically focused on how to teach CS principles • Professional development. CodeVA (Code Virginia) has been a great PD resource • Masters in Business Education. Self taught in Visual C#, the XNA Game Studio, and Game Maker through books, online curriculum, and research • BS in Business Education, MEd in Curriculum & Instruction, most CS is self-taught • I taught myself several languages and have taught programming of some sort for over 15 years. • I am primarily self-taught with a lot of help from youtube and books. • Certification in PA is under Business, there is no CS content on the Business Praxis exam or in the preparation courses

CURRICULUM VITAE

Olgun Sadık
olsadik@indiana.edu

Education

Indiana University, Bloomington 2017
Ph.D. in Instructional Systems Technology
Ph.D. Minor in Inquiry Methodology

Indiana University, Bloomington 2011
Master's in Instructional Systems Technology

Kocaeli University, Turkey 2005
B.S. in Computers and Electronics Education

Professional Experience

Teaching

Lecturer (School of Informatics and Computing- Indiana University)

- A290- Tools for Computing (PHP) (Spring 2017)
- A290- Tools for Computing (JavaScript) (Spring 2017)
- C295- Leadership and Learning (Spring 2017)
- B599- Teaching Computer Science (Spring 2017)

Associate Instructor (School of Education- Indiana University):

- W220- Technical Issues in Computer-Based Education (Spring 2014-Spring 2013)
- R341- Multimedia in Instructional Technology (Spring 2014, Fall 2013, Summer 2013, Spring 2013, Fall 2012)
- W200- Computers and Education (Spring 2011- Fall 2011)

Computer Teacher and Technology Coordinator

- Bozuyuk Technical High School, Turkey (2005-2006 and 2007-2008)
- Prof. A. Karahan Elementary School, Turkey (2006 - 2007)

Technology Coordinator and Support

Technology Coordinator
Center for Research on Learning and Technology, Indiana University, Bloomington:
Coordinates and supports technology resources for the center (September 2014- present)

Technology Consultant
Teacher Education Office, Indiana University, Bloomington: *Designed, developed, organized,* and *delivered* technology workshops for the School of Education students. (Spring 2012)

W200 Undergraduate Lab Assistants' Coordinator-
School of Education- Indiana University, Bloomington: *Coordinated* and *led* W200-Computers and Education Undergraduate Lab Assistants' (Fall 2011- Spring 2012)

Teaching Technology Lab Manager
School of Education- Indiana University, Bloomington: *Communicating and guiding* students and instructors to learn and use various instructional technologies (Fall 2010).

Journal Publication

Ertmer, P., Ottenbreit-Leftwich, A., **Sadık, O**., Sendurur, E., Sendurur, P. (2012). Teacher beliefs and technology integration practices: A critical relationship. *Computers and Education, 59*(2), 423-435.

Proceeding

Sadık, O. (2015, April). Encouraging Women to Become CS Teachers. In *Proceedings of the Third Conference on GenderIT* (pp. 57-61). ACM.

Book Chapters

Ertmer, P., Ottenbreit-Leftwich, A., **Sadık, O**., Sendurur, E., Sendurur, P. (2012). Teacher beliefs and technology integration practices: Examining the alignment between espoused and enacted beliefs. In J. König (Ed.), *Teachers' Pedagogical Beliefs* (pp. 149-169). Waxman: Munster Germany.

Sadık, O., Leftwich, A. O., & Nadiruzzaman, H. (2017). Computational thinking conceptions and misconceptions: Progression of preservice teacher thinking during computer science lesson planning. In P. T. Rich and C. B. Hodges (Eds.), *Emerging research, practice, and policy on computational thinking* (pp. 221-238). Cham, Switzerland: Springer International Publishing.

Conference Presentations

Sadık, O. (October, 2016). Secondary computer science teachers' knowledge and school setting related needs. Paper accepted at *Association for Educational Communications and Technology Conference,* Las Vegas, NV.

Ottenbreit-Leftwich, A., Liao, J., & **Sadık**, O. (June, 2016). Evolution of teacher technology knowledge, beliefs and practices: Preservice to inservice. Paper presented at *International Society for Technology Education Conference,* Denver, CO.

Sadık. O. (April, 2015). Encouraging women to become CS teachers. Paper presented at *Gender in IT Conference*, Pennsylvania, PA.

Ottenbreit-Leftwich, A., **Sadık, O**., & Liao, J. (June, 2014). Evolutions of teacher technology knowledge, beliefs and integrations: Beyond the college. Paper presented at *International Society for Technology Education Conference*, Atlanta, GA**.**

Sadık, O. (April, 2014). How to make a pre-service technology integration course better. Paper presented at *American Educational Research Association Conference*, Philadelphia, PA.

Sadık, O., Celik, S., & Ottenbreit-Leftwich, A. (2013, October). Preparing future teachers for the movement to increase opportunities and technology project goals. Paper presented at *Association for Educational Communications and Technology Conference, Anaheim, CA.*

Sadık, O., Ottenbreit-Leftwich, A., & Liao, Y. (2013, October). Investigating tech-savvy pre-service teachers' technology integration knowledge, beliefs and intentions. Paper presented at *Association for Educational Communications and Technology Conference, Anaheim, CA.*

Hoey, B., Aslan, S., Zachmeier, A., **Sadık, O**., Glazewski, K., Ottenbreit-Leftwich, A., & Brush, T. (2013, October). Technology integration concerns: Expanding the dialogue between pre-service teachers and exemplary technology-using in-service teachers. Paper presented at *Association for Educational Communications and Technology Conference, Anaheim, CA.*

Sadık, O. (2012, November). The Relationship Between Technology Integration Barriers and Pre-Service Teachers' Beliefs and Intentions. *Poster presented at Association for Educational Communications and Technology, Louisville, KY.*

Service

Journal Peer-Reviewer: Computers and Education	2013- Present
Social Media Officer: AECT Graduate Student Assembly	2014- 2015
Conference Proposal Reviewer: AERA (Chicago, IL)	2015
Conference Proposal Reviewer: AECT Conference (Indianapolis, IN)	2015
Conference Proposal Reviewer: AECT Conference (Jacksonville, FL)	2014
Conference Proposal Reviewer: ISTE Conference (Atlanta, GA)	2014
Concurrent Session Facilitator: AECT Conference (Anaheim, CA)	2013
Concurrent Session Facilitator: AECT Conference (Louisville, KY)	2012
Evening Reception Coordinator: IST Conference (Bloomington, IN)	2012
Volunteer: European Youth Olympic Games (Trabzon, Turkey)	2011

Turkish Student Association (TSA) President: Indiana University	2010- 2011
Director of Fundraising: IST Conference (Bloomington, IN)	2011
IST 2010 Conference Webmaster: (Bloomington, IN)	2010
Volunteer Technology Assistant: AECT 2009 Conference (Louisville, KY)	2009

Scholarships, Grants, Honors and Awards

Instructional Systems Technology Larson Award- $500	2015
Instructional Systems Technology Kemp Award- $500	2015
Indiana University Service Learning Graduate Fellowship Award- $500	2015
Center for Research and Technology Travel Grant- $500	2015
Instructional Systems Technology Larson Award- $300	2014
Indiana University- Scholarship of Teaching and Learning Grant- $5000	2013
Turkish Ministry of National Education Scholarship for Master's and Ph.D.	2008
Ministry of Education- Honor Certificate for Teaching	2007
Ministry of Education Honor Certificate for Teaching	2006

Computer Skills

Applications **:** Microsoft Office Suite, Adobe Dreamweaver, Adobe Flash, Adobe Photoshop, Adobe Captivate, Adobe Premier, Windows Movie Maker, Audacity, and many Web 2.0 tools

Research Software **:** SPSS, NVivo

Operating Systems **:** Ms-Dos, Windows, Mac OS, Linux

Programming Languages **:** HTML, CSS, PHP, C, Python, JavaScript

Database **:** MySQL

LMS **:** Oncourse, BlackBoard, Moodle, Canvas

Web Conferencing **:** Adobe Connect, Skype, Google Hangout

Technical **:** PC Hardware