Computer Science Principles: Analysis of a proposed Advanced Placement Course

Andrea Arpaci-Dusseau
University of Wisconsin, Madison
dusseau@cs.wisc.edu

Owen Astrachan
Duke University
ola@cs.duke.edu

Dwight Barnett
Virginia Tech
barnette@vt.edu

Matthew Bauer
Illinois Institute of Technology
bauerm@iit.edu

Marilyn Carrell
Springdale High School
mcarrell@sdale.org

Rebecca Dovi
Patrick Henry High School
rdovi@hcps.us

Baker Franke
Chicago Lab High School
bfranke@ucls.uchicago.edu

Christina Gardner
Georgia Tech
cmgardner@cc.gatech.edu

Jeff Gray
University of Alabama
gray@cs.ua.edu

Jean Griffin
University of Pennsylvania
griffin@seas.upenn.edu

Richard Kick
Newbury Park High School
kickrg@gmail.com

Andy Kuemmel
West High School
andykuemmel@yahoo.com

Ralph Morelli
Trinity College
ralph.morelli@trincoll.edu

Deepa Muralidhar
North Gwinnett High School
deepa.muralidhar@gmail.com

R Brook Osborne
Duke University
rbo@cs.duke.edu

Chinma Uche
GHAMAS
cuche@crec.org

ABSTRACT

In this paper we analyze the CS Principles project, a proposed Advanced Placement course, by focusing on the second pilot that took place in 2011-2012. In a previous publication the first pilot of the course was explained, but not in a context related to relevant educational research and philosophy. In this paper we analyze the content and the pedagogical approaches used in the second pilot of the project. We include information about the third pilot being conducted in 2012-2013 and the portfolio exam that is part of that pilot. Both the second and third pilots provide evidence that the CS Principles course is succeeding in changing how computer science is taught and to whom it is taught.

Categories and Subject Descriptors

K.3.2 [**Computers & Education**]: Computer & Information Science Education --- *Computer Science Education.*

Keywords

Advanced Placement, Portfolio Assessment, National Pilot.

1. Introduction

We report on and analyze the Computer Science Principles project, a project sponsored by both the National Science Foundation (NSF) and the College Board. CS Principles is intended and designed to be a rigorous, engaging, and broadly appealing Advanced Placement (AP) course taught in high schools for which students can earn placement and/or credit for a college course. In this paper we provide an historical context that helps explain the development of CS Principles in the AP program, we describe the process by which the CS Principles exam is being designed and planned, we analyze the second and third pilots of the course, and we describe the next phase of the project leading to a national, AP exam.

SIGCSE'13, March 6–9, 2013, Denver, Colorado, USA. Copyright © 2013 ACM 978-1-4503-1868-6/13/03...$15.00.

2. Historical Perspectives of APCS

The development of the current AP Computer Science exam provides both insight and foundation for the development of the CS Principles project. The College Board developed and delivered the first Advanced Placement Computer Science exam in the early 1980s --- the first exam was given to students in May of 1984 [1]. That exam covered what are conventionally called CS1 and CS2 using the language Pascal. Over thirty years that exam changed in many ways: first to report a CS1 (called the A exam) sub score, then to create two exams and courses: CS1 (APCS A) and CS2 (APCS AB), and then to discontinue the AB exam. The course and exam went through several changes including changing the language from Pascal to C++ and then to Java [2, 3]. Perhaps more important were changes made more for pedagogical reasons rather than to track the conventions used in college courses. The pedagogical changes included the adoption of case studies that were fundamental to both the curriculum and the exam [4]. These case studies permitted students to write small functions that fit

into a much larger code-base that had been (presumably) the object of study throughout the AP course leading up to the exam. The case studies engendered both robust and reliable questions on the AP exam, but they also allowed teachers to structure a course around a software artifact that was accompanied by a written design document. Very few colleges use a similar approach --- so case studies were used for purposes of pedagogy and assessment rather than to track the conventions used in most colleges. The College Board has plans to replace the case study by a set of labs that students would use during the year and that would be the basis of questions on the exam. The growth-rate in the number of students taking the AP Computer Science exam has lagged far behind the growth rate of the AP program in general, and the demographics [see Sections 2.1 and 3.2] of students taking the AP Computer Science exams have not included sufficient representation by women and under represented minorities (URMs). These characteristics of the student population were a principle impetus in the development of the CS Principles course and exam.

2.1 Underproduction, Underrepresentation

The issues of underproduction and underrepresentation are clear in the demographics of students currently taking AP CS as well as in historical trends of this population. In 2011, AP CS A accounted for 21,139 of the 3,365,617 total AP Exams taken by students in the US --- that is .62% of all AP Exams scored in the US. Of these 21,139 students, 19% were from underrepresented racial and ethnic groups. Women accounted for 19% of APCS test takers while women account for 54% of all AP test takers [5]. The data for 2012 have the same percentages for both URMs and women [6]. Between 2002 and 2012 the number of AP test takers across all exams increased by 57%. In the STEM fields of Statistics, Calculus, Physics, Biology, and Computer Science the corresponding increases were 68%, 45%, 52%, 49%, and 11% [7].

According to a longitudinal study of nearly 40,000 college and university students released by the College Board in 2011, 17.9% of students who took the AP CS A Exam went on to major in Computer and Information Sciences, compared to 2.3% of the total sample. The findings of this report are in line with prior research indicating students who take an AP Exam in a particular content area are more likely to major in a related discipline in college [8]. The low enrollment in AP CS means this positive relationship does not translate into a large cohort of CS majors. Under enrollment and underproduction of degrees in computer science is a particularly salient problem given the high demand for qualified employees in the field. For years there has been a gap between the number of available jobs requiring a background in computer science and the number of qualified job hunters --- in 2010 there were a total of 1.3 million job-openings (from both job creation and replacement) that fit into the Bureau of Labor Statistics "Computer Occupations" category, with expected job growth of 22% between 2012 and 2020 [9]. These data sit in stark contrast to the number of students majoring in Computer Science; according to the Taulbee survey from the same year, there were only 12,501 bachelor's degrees awarded in Computer Science. Of those degrees 13.8% were awarded to women, 3.4% to Blacks or African Americans, and 5.3% to resident Hispanics [10].

From these data two things are clear: the CS community must motivate and educate a substantially larger number of students to fulfill the demands of the market while reaching a broader, more diverse segment of the population. The Mattern et al. study demonstrates a strong relationship between AP CS A Exam participation and pursuing a major in a related field. The creation of CS Principles, an AP course that increases opportunities for engagement with CS and develops new ways to interest potential students that have not fit into prior paradigms of learning CS, has the potential to increase the number and diversity of students majoring in the field. By attracting more students into AP CS, we will see more students completing courses of study that will prepare them to enter the computer science workforce. Beyond addressing the issues of overall access to learning opportunities, both the content and structure of the CS Principles course and exam described below aim to attract students who do not traditionally enroll in CS courses.

2.2 Development of CS Principles

The approach recently adopted across the AP program for the revision or creation of an exam is based on evidence-centered design [11]. This design methodology has led to revisions in AP Calculus, Biology, Physics, and was used in drafting the curriculum framework for CS Principles. Detailed learning objectives (LOs) with evidence statements are used to determine whether the objectives have been met. These objectives are constructed by matching the so-called seven *Big Ideas* of CS Principles with six *Computational Thinking Practices*. Details can be found on the CS Principles website [12]. Here we use one example to illustrate how the 35 learning objectives were developed. One of the Big Ideas is Abstraction: *Abstraction reduces information and detail to facilitate focus on relevant concepts*. This Big Idea is divided into three key concepts, each of which is divided into three-to-four supporting concepts. Using the Big Idea and the Computational Thinking Practice of *Developing Computational Artifacts*, leads to the following learning objective: The student can develop an abstraction. The following points are the evidence statements used to determine if a student has met the learning objective described above.

- Creation of an abstraction for a hardware, software, or conceptual purpose
- Use of appropriate abstractions in the creation of the artifact
- Selection of appropriate algorithmic and information-management abstractions

More details on learning objectives and evidence statements can be found by examining all 35 learning objectives and their associated evidence statements.

The seven big ideas are:

1. Creativity: Computing is a creative activity
2. Abstraction: Abstraction reduces information and detail to focus on relevant concepts
3. Data: Data and information facilitate the creation of knowledge
4. Algorithms: Algorithms are used to develop and express solutions to computational problems
5. Programming: Programming enables problem solving, human expression, and creation of knowledge
6. Internet: The Internet pervades modern computing
7. Impact: Computing has global impacts.

The six Computational Thinking Practices are:

1. Connecting Computing
2. Developing Computational Artifacts
3. Abstracting
4. Analyzing problems and artifacts
5. Communicating
6. Collaborating

The entire curriculum framework was developed by a group of ten educators and professionals whose work was overseen by an advisory board of thirty drawn from colleges, secondary schools, and professional organizations.

2.3 Future of CS Principles

The CS Principles project is building toward a national AP exam by the 2016-2017 academic year. The next phase of the project is the expansion of the pilot program described in the next section and the development of portfolio questions and rubrics that will form the foundation of the CS Principles exam. This phase is underway in academic year 2012-2013 and is an integral part of the third pilot described in the next section. A new advisory board is overseeing the development of the course, the portfolio, and the professional development that the College Board supplies to all AP courses. This board is working to ensure that the approach taken appeals to a broad audience while maintaining the rigor that is an essential part of all AP courses.

3. Summary of Pilots

Three years of piloting the framework were planned in the first phase of the CS Principles project between 2010-2013. Phase II of the project begins in 2013 and includes a larger pilot program designed to lead to the launch of an AP course. In this section we describe the three pilots that are part of the first phase of CS Principles, these are Pilots I, II, and III.

3.1 Pilot I

The first phase, in 2010-2011, consisted of five institutions and courses with instructors drawn from the advisory board since they would be familiar with the framework and the rationale by which it was developed. These included: Beth Simon, UCSD; Dan Garcia, UC Berkeley; Jody Paul, Metropolitan University of Denver; Tiffany Barnes, UNC Charlotte; Larry Snyder, U. Washington. This first pilot was designed to determine how the curriculum framework could support many different approaches. For example, as reported in [13], the type of software and the programming languages used was different across the five pilots. The size of the classes and the pedagogical approaches used varied drastically. The number of contact hours per week with students varied, as did the kind of artifacts that students produced. As the first pilot took place, the CS Principles team coordinated a national survey of colleges and universities designed to assess the curriculum framework and to determine if it could lead to widespread adoption in terms of college placement and/or credit. This ultimately led to revisions to the framework, e.g., the Internet replaced a more generic and abstract Big Idea on networking and topics based on algorithmic and complexity concepts such as intractability, P, and NP, were replaced by topics that are less theory-centric.

3.2 Pilot II

A second pilot was conducted in 2011-2012 using the revised curriculum framework that resulted from the first pilot. This pilot consisted of ten high schools partnered with eight colleges.[1] The schools in this pilot were North Gwinnet HS/Georgia Institute of Technology; Northside College Prep with Chicago Lab School/Illinois Institute of Technology; Greater Hartford Academy of Math and Science/Trinity College; Booker T Washington Magnet High School/University of Alabama; Springdale High School/University of Arkansas, Little Rock; South Philadelphia HS/University of Pennsylvania; West High School/University of Wisconsin, Madison; Patrick Henry HS/Virginia Tech; Newbury Park HS (California). The student demographics from the second pilot are shown in Table 1, and provide evidence that CS Principles is addressing the issue of enrollment of underrepresented minorities in computer science (see Section 2.2).

Table 1. Aggregated Demographic Data for Pilot II

Race/ Ethnicity	American Indian	Asian/ Asian American	Black
	.23%	18.0%	23.4%
	Hispanic/ Latino	White	Other
	15.0%	41.3%	1.8%
Gender	Male	Female	
	60.5%	39.5%	

Instructors in the first pilot often differentiated their courses by emphasizing the programming languages used in their courses [13]. In the first pilot each of the five instructors used a language that was not used by another instructor though more than one of the five pilots used Snap! (formerly BYOB).[2] In contrast, more than half of the 18 schools in the second pilot used App Inventor [14]. The attraction of creating an app that can be used on a phone or tablet is a real draw for many students. However, it is not clear that using App Inventor will scale across hundreds of schools. Given the limitations of the simulator used in App Inventor (as opposed to deploying apps on a real device), testing and demonstrating code on an actual device is often necessary, and funds for cell phones, tablets, or other Android devices will almost certainly not be available for all schools. In the second pilot, two courses used the code-in-the-browser website [15]. This is a zero footprint, browser-only programming environment that has been met with reasonable success.[3] In the third pilot, instructors have indicated that they will make use of the new Khan Academy Computer Science program. This environment is based on work done by Resig and Bibeault [16] and provides a real-time, read-eval-print loop in the browser for experimenting and playing with code. The learning environment is unique in that each keystroke is immediately evaluated so that changes to a program are visualized as they are made rather than as part of an edit, compile, run, debug cycle. The quick adoption of this tool by the third pilot group indicates the potential of this approach,

[1] Two high schools in Chicago were partnered with the same college. One of the colleges originally in the pilot had to drop out before the pilot course was finished, but the partnership with the high school continued.

[2] In 2010-2011 Snap! was called BYOB (Build Your Own Blocks) and was used in two of the five pilots, though one of these two also used App Inventor.

[3] This environment was used in the Coursera offering of Compsci 101 (see coursera.org).

though its longevity and success in the context of CS Principles will arguably be a function of the degree of adoption by the much larger Phase II pilot program in 2013-2014.

3.3 Pilot III

A third pilot is being conducted in 2012-2013. The schools and teachers participating in this pilot were all drawn from the second pilot group. The third pilot group includes North Gwinnet HS, Greater Hartford Academy of Math and Science, Patrick Henry HS, Newbury Park HS, University of Alabama, and University of Wisconsin, Madison. This group of pilots will use an open-ended portfolio assessment during the course. The portfolio questions are available on the csprinciples.org website for use by the entire educational community.

Portfolio-based assessment often centers on a collection of student work that demonstrates achievement or improvement [17]. While the particular structure of a portfolio assessment can vary, two approaches are the most prevalent within the education research literature. The first approach treats the use of a portfolio as a capstone experience where the best student work is selected and placed into the portfolio, representing the student's current level of mastery of knowledge and skills in a domain. The second portfolio structure represents a student's learning process that demonstrates growth or relative change rather than the final level of achievement. Student work for this type of portfolio is selected along a continuum of student progress toward mastery throughout the duration of a course. Two national exams include a portfolio component: AP Studio Art [18] and the National Board Certification for teachers [19].

Portfolio assessments are ideal for capturing non-traditional, project, or performance-based student work. They have also been lauded for their ability to better capture the authentic application of a student's acquired knowledge [20, 21]. Portfolio assessments have been found to produce positive effects on students, teachers, and instruction [22]. Portfolios can also provide the opportunity for students to actively engage in and understand a rubric that can provide pre-determined criteria for success. Thus, portfolio-based assessments provide greater levels of student awareness of the process of gaining and applying knowledge and skill.

3.4 Future Pilots

Phase II of CS Principles includes three more years of pilots at roughly 40-50 high schools and colleges in each year. More schools than these 40-50 will certainly be involved in delivering a CS Principles course, but not as part of the formal research project carried out by The College Board. Thus, they will not be part of annual meetings, nor will these early adopters receive the same level of community support provided to the formal pilots in the CS Principles project. However, ongoing efforts supported by the NSF and industry to support CS Principles will extend beyond these 50 schools. Each of the Phase II pilot schools will use the portfolio assessment and participate in the research-based approach to grading the portfolios that is part of the ongoing development of the CS Principles project.

4. Prototype Assessment in Pilot II

As discussed in Section 3.2, seventeen high schools, colleges, and universities took part in Pilot II. While the first pilot was focused on determining if instructors would be able to create engaging and rigorous courses in diverse settings based on the curriculum framework, the second pilot was centered on investigating the feasibility of assessing student understanding of the Big Ideas and Computational Thinking Practices. To this end, each pilot course culminated in a prototype assessment with both objectively and subjectively scored questions. In this section we discuss and analyze the assessments used in Pilot II. In the next section we discuss and analyze the courses that were part of Pilot II.

Students in the Pilot II courses participated in an end-of-course exam intended to assess the feasibility of an online testing platform as well as both objectively scored questions (e.g., matching or multiple choice) and subjectively scored questions (e.g., open-ended, free-response questions (FRQs)). In a separate report, information about these results is available[4], we provide a quick summary of the results here as they affect the development of the portfolio used in current (Pilot III) and future pilots. Objectively scored questions will be a component of the planned AP exam for CS Principles. The chief lesson learned from the use of such questions in the online, computer-based exam that was part of Pilot II is that students will need practice with new kinds of questions --- those that go beyond simple multiple choice to include multiple-correct responses, drag-and-drop completion questions, and animated questions that require students to manipulate sliders to change values. The free-response questions illustrated what an analysis of the time-in-class on each big idea had already provided in a preliminary analysis of the second pilot courses: instructors did not have sufficient guidance in the depth-of-coverage anticipated in all areas of the curriculum framework. As an example, student performance on the question below was very poor:

> *The Internet Protocol IPv4 was in widespread use from 1980-2012. There is a more recent protocol named IPv6 now used more frequently than in the past. With IPv6 128 bits specify an IP address whereas 32 bits specify an address using IPv4. IPv6 also includes support for Internet security that is not present in IPv4. Describe an example for why the change in the number of bits per address is necessary and an example for why security is necessary in the new, more recent IPv6 protocol compared to the IPv4 protocol.*

The open-ended nature of the portfolio questions used in Pilot III is due in part to the desire to stay away from fact-based questions such as the one above since the release of such questions will likely have an immediate effect on what teachers discuss and cover in subsequent years.

5. Exploration and Analysis of Pilot II

This section will explore and analyze how courses in Pilot II were structured in terms of both content and pedagogy. This analysis is holistic rather than quantitative, aimed at providing an understanding of how courses were implemented and conducted.

5.1 The Curriculum Framework

The planning and development of the 18 courses in Pilot II differed in substantial ways from Pilot I. The instructors in Pilot II were not part of the development of the curriculum framework as were the instructors in Pilot I; Pilot II included high schools whereas Pilot I only included colleges; and high schools were

[4] This report will be published after all the FRQs have been completely graded.

partnered with universities. Perhaps as a result of these differences Pilot II instructors engaged directly with the curriculum framework while building and maintaining their CS Principles courses. For example, one instructor notes that when he was developing materials and activities during the year, he would turn to the CS Principles framework, find a learning objective that had not been covered thoroughly in the course, develop a sequence of activities leading to a project that both illuminated the objective and developed new skills as well. Many instructors report that they covered several Learning Objectives in each class project and revisited those LOs frequently in different contexts. By contextualizing topics across several domains students had more opportunities to experiment with those concepts. This is an important feature of Constructivism [24]; a learning theory adopted by many of the Pilot II instructors.

5.2 Constructivist Classroom

In his 1916 book, *Democracy and Education*, John Dewey says of education: "It is that reconstruction or reorganization of experience which adds to the meaning of experience, and which increases ability to direct the course of subsequent experience. The increment of meaning corresponds to the increased perception of the connections of the activities in which we are engaged." [23] This is the backbone of constructivist learning theory, or the high-level process in which the learner builds understanding of a concept through first-hand experience [24]. In his 2002 book *The Art of Changing the Brain*, James Zull highlights how the experiential learning model of David Kolb is tied to the biology of the brain. Kolb's model, which supports a constructivist approach, is broken down into four parts: (1) concrete experience with a topic, (2) reflection about the experience with the topic, (3) extending the topic to form generalizations and make connections to broader concepts, and (4) testing these concepts and generalizations in new situations [25]. Zull makes a case that this model is supported by neuroscience, that each of Kolb's parts is supported by the brain: (1) sensory cortex, (2) temporal integrative cortex, (3) frontal integrative cortex, (4) premotor and motor brain [26].

Many Pilot II instructors delivered courses built on the educational tenets of experiential learning and constructivism, a paradigm facilitated by the CS Principles framework. Perhaps the synergy engendered by including high school teachers in the second pilot facilitated this approach. Rather than conducting lecture-based courses, the overwhelming majority of Pilot II instructors produced curricula based on a constructivist theory.

5.2.1 Experiential Learning

Many of the instructors focused on student comprehension through the model of experiential learning. In one case, an instructor designed projects that were specifically open-ended and co-curricular. Students started by learning about and exploring tools and concepts, and then leveraged this new knowledge to complete projects that featured content from other courses. Another instructor structured projects similarly, giving students the option to explore topics and create artifacts that related to their personal areas of interest. This resulted in a variety of projects, from recreations of childhood games to tools for use in other courses. One of the instructors who used a drag-and-drop language early in the semester, but later switched to a text-based, procedural language, noted that students benefitted from working out logic in the visual programming language and using that as a blueprint for coding in the text-based language. In this case, students were abstracting a solution from an established context into a newer, less familiar situation. Another strategy employed was the creation of student-maintained concept dictionaries. In one case, the definitions were revisited daily and refined as the course progressed, providing students with a record of the growth of their understanding of the topics. In another case, students created and maintained a 'computing concept dictionary' where they translated their understanding of topics into jargon-free definitions.

One instructor integrated Service Learning into her course, with each student performing at least 3 hours of course-relevant community service for K-12 youth using drag-and-drop languages. Another pilot instructor noted the increase in student inspiration when topics were put into context and made relevant after completing a project. In another course, students would first examine magnified details of a topic and then explore the topic in the context of the "big picture" with the direction of exploration guided by student suggestions and preferences. In that same class, quizzes and tests were formatted in such a way that students were required to do much more than just answer questions; rather, they would speculate about possible solutions to problems, compare their speculations with sample solutions that were made available to them after their initial speculations, and then resolve any differences that were identified. Each of these examples highlight experiential learning and are rooted in the natural learning cycle, demonstrating instances of students extending the topic of instruction and making connections to broader concepts, and then testing these concepts in new situations.

5.2.2 Reflection

As a tenet of experiential learning, reflection gives students the opportunity to abstract their understanding of a topic, thereby creating versatile content knowledge that can be applied to new and different tasks. As such, reflection is a hugely important component of the teaching and learning process and was embraced by the Pilot II instructors. In many courses, the reflection process was formalized. Students would share their written reflections with other students or the instructor in a number of different ways --- from writing papers, to journaling in a shared Googledoc, to blogging. In several cases, students were encouraged to reflect on what they were learning about computing in the context of its impact on their lives and future careers. This exercise prompts the learner to consider the broader implications and applications of the content knowledge she is amassing; an important part of the learning cycle. As noted, students often reflected publically and online; these online spaces often went beyond repositories for self-reflection. Many students posted both their work products and their reflections in these spaces throughout the year. They would discuss their work with their peers and reflect on prior work after these discussions. This feedback and shared reflection process presents the learner with an opportunity to refine her understanding of the content, and in some cases, refine the actual work product.

Many instructors noted the need for providing guidance during the reflection process. When done correctly, reflection gives the teacher an opportunity to see the depth of a student's understanding; however many students have no experience with deep reflection. In one case where an instructor invested time in building an understanding of the mechanics of reflection, she saw the students' ability to write and reflect improve dramatically throughout the year, giving her a snapshot of how well they understood the material covered in class. All of these examples

point to the usefulness of reflection as a tool for gaining and demonstrating comprehension. Reflection is a cornerstone of the portfolio tasks being used in Pilot III, especially for those tasks that are collaborative.

6. Future Directions

The next steps in the Phase II CS Principles project are grounded in ensuring that the portfolio exam can be successfully administered on a much larger scale. This includes the possibility of a distributed grading of the portfolios --- a distinct contrast to existing grading of AP free response questions which requires travel and housing for more than a week to support the grading of each subjectively scored question in AP exams. As an example, nearly one thousand educators convene centrally to grade the AP English and Composition exam. Since the digital components of the CS Principles portfolio can be duplicated, a distributed grading is more than feasible. The development of such a process is an important component of the CS Principles project.

Professional development (PD) supported by the College Board is another important component. All AP courses have short, focused PD opportunities for educators, however, successful professional development requires an emphasis on pedagogical content knowledge, not simply on content [27]. As such, it is important that educators have access to PD that facilitates deep understanding and retention of pedagogical content knowledge.

7. ACKNOWLEDGMENTS

We thank the CS Principles commission, the advisory board and all the pilot instructors across all three pilots. We also thank the staff at College Board and our external evaluator.

8. REFERENCES

[1] D. Rine, J. R. J. W. Wadkins, and G. Steven, "Advanced Placement Program in Computer Science," *Proceedings of the SIGCSE Symposium,* vol. 15, p. 204, 1983.

[2] M. Stehlik, S. Rodger, K. Larson, A. Brady, and C. Nevison, "Current and Future Directions of the Advanced Placement Exam," in *Proceedings of the SIGCSE Symposium*, New Orleans, LA, 1999, p. 358.

[3] O. Astrachan, G. Chapman, S. Rodger, and M. Weiss, "The reasoning for the advanced placement C++ subset," *ACM SIGCSE Bulletin,* vol. 29, pp. 62-65, 1997.

[4] M. Linn and M. Clancy, "The case for case studies of programming problems," *Commun. ACM,* vol. 35, pp. 121-132, 1992.

[5] (2011, November 29, 2012). *AP Summary Reports: 2011*. Available http://bit.ly/apscores2011

[6] (2012, November 29, 2012). *AP Summary Reports: 2012*. Available http://bit.ly/apscores2012

[7] (2012, November 29, 2012). *AP Program Participation and Performance Data 2012*.

[8] K. Mattern, E. Shaw, and M. Ewing, "Advanced Placement Exam Participation: Is AP Exam Participation and Performance Related to Choice of College Major?," 2011.

[9] B.L. Statistics. (2010). *Employment by occupation* Available: http://www.bls.gov/emp/ep_table_102.htm

[10] C.R. Association, "Computing Research Association: Taulbee Survey Report 2009-2010," 2011.

[11] R. J. Mislevy and G. D. Haertel, "Implications of Evidence-Centered Design for Educational Testing," *Educational Measurement: Issues and Practice,* vol. 25, pp. 6-20, 2006.

[12] C. P. Commission. (2012, Computer Science Principles Big Ideas and Key Concepts Learning Objectives and Evidence Statements. Available:

[13] T. B. Larry Snyder, Dan Garcia , Jody Paul, Beth Simon. (2013, June, 2012) The First Five Computer Science Principles Project. *ACM Inroads*.

[14] D. Wolber, H. Abelson, E. Spertus, and L. Looney, *App Inventor*: O'Reilly, 2011.

[15] N. Parlante. (2012). *Code in the Browser*. Available: http://codeinthebrowser.org/

[16] J. Resig and B. Bibeault, *Secrets of the JavaScript Ninja*: Manning, 2012.

[17] G. Wiggins, "A true test: toward more authentic and equitable assessment," *Phi Delta Kappan,* vol. 70, pp. 703-713, 1989.

[18] N. Chudowsky and J. Pelligrino, "Large Scale Assessments that Support Learning: Will it Take?," *Theory into Practice,* vol. 42, pp. 75-83, 2003 2003.

[19] D. Goldhaber and E. Anthony, "Can Teacher Quality Be Effectively Assessed? National Board Certfication as a Signal of Effective Teaching," *The Review of Economics and Statistics,* vol. 89, pp. 134-150, 2007.

[20] D. P. Wolf, "Assessment as an episode of learning," in *Construction Versus Choice in Cognitive Measurement*, R.E. Bennett and W.C. Ward, Ed., Hilllsdale, NJ: Lawrenece Erlbaum Associates, 1993, pp. 213-240.

[21] F. L. Paulson, P.R. Paulson and C.A. Meyer, "What makes a portfolio a portfolio?," *Educational Leadership,* vol. 48, pp. 60-61, 1991.

[22] M. Gearhart and E. Osmundsun, "Assessment portfolios as opportunities for teacher learning.," *Educational Assessment,* vol. 14, pp. 1-24, 2009.

[23] J. Dewey, *Democracy and Education*. New York: MacMillan, 1916.

[24] C. T. Fosnot, Ed., *Constructivism: Theory, Perspectives, and Practice*. New York: Teachers College Press, 1996.

[25] D. Kolb, *Experiential Learning: Experience as the Source of Learning and Development*. New Jersey: Prentice Hall, 1984.

[26] J. Zull, *The art of changing the brain: Enriching the practice of teaching by exploring the biology of learning*. Virginia: Stylus, 2002.

[27] L. Darling-Hammond and N. Richardson, "Teacher Learning: What Matters," *Educational Leadership,* vol. 6, pp. 46-53, 2009.