Building Evidence and Building Practice in Computer Science Education
White Paper Submitted to CS Education Research Summit

Marie Bienkowski
Eric Snow
SRI International, Education Division

Need. The Computer Science education and practice community has made clear the need to increase the size and diversity of US-trained computer science professionals (e.g., see the December 2012 special issue of ACM Inroads magazine and the formation of code.org). Computer science is intricately involved in our politics, our social lives, and our science, and its failures and successes are equally denounced (*Disruptions: More Connected, Yet More Alone,* New York Times, September 1, 2013*)* and celebrated (*Stanford's Nobel chemistry prize winner honors computer science,* San Jose Mercury News, October 9, 2013). And the public is all too ready to treat software failures as indicators of larger issues at play (*Poll: Majority believe health-care Web site problems indicate broader issue with law.* Washington Post, October 21, 2013; *Obamacare's Black Box,* FoxNation website.)

Given events such as these, hosting a summit to prepare for the future of computer science education is timely. The call for white papers asks for research questions that, when answered, "will contribute evidence-based findings to the body of knowledge on teaching and learning of Computer Science within diverse student and teacher populations." We believe that is a goal that is **not ambitious enough**, and that the research program should contribute substantially to the *practice* of computer science education at the same time as the field builds *knowledge and findings* through research. Therefore, in addition to presenting our research topics, we propose that when applicable, they be studied using the latest design-based education research methodologies that serve the dual role of building theory and improving practice at scale (Penuel, Fishman, Cheng, & Sabelli, 2011).

We recommend research in areas involving the early portion of the pathway to computer science as a profession: K-12 education. We propose the following as critical research topics to advance the field of computer science education research and learning science theories of computational thinking and related practices. After describing these topics (in varying degrees of details due to space and time limitations), we talk about an overall approach to researching these questions based on the most current education research methodologies. We define *education practice* as encompassing a range of important activities in the K-12 education ecology, including classroom enactments, ways of teaching and learning, and administrative level policies and actions that shape institutional cultures.

Critical Research Areas for K-12 Computer Science Education

***Analyze and Operationalize the Computational Thinking Domain**.* The cognitive domain of computational thinking—however it is eventually defined—underlies computer science as a profession (NRC, 2010; NRC, 2011). We recommend research to articulate the specific knowledge, skills, and attributes that constitute computational thinking and computational thinking practices. This definitional work should be done at a grain size that allows the learning-sciences research and practitioner community to develop formative assessments for use by CS educators and that also allows the assessment research community to develop, test, and validate summative assessments. Within this research area, research questions such as the following are important to pursue:

1) How can computational thinking and related practices be carefully and precisely defined to support evidence-based assessment?
2) How can the field design assessment tasks that elicit evidence about students' computational thinking ability?

Additionally, computational thinking constructs and practices should be *operationalized* using techniques from learning analytics to support naturalistic assessment and adaptive learning in technology-enabled environments (including programming environments). In a recently funded NSF project (NSF SMA 1338487) under the program "Building Community and Capacity for Data-Intensive Research in SBE and EHR," SRI is bringing together an interdisciplinary community to build the capacity to model noncognitive factors (specifically, academic perseverance and learning strategies) using data collected by learning technologies and cyberinfrastructure. This work will be done using conventions from the fields of psychometrics and assessment design that support researchers in developing rigorous assessments of complex student behaviors extended with models built on educational data mined from digital environments (Mislevy, Behrens, DiCerbo, & Levy, 2012). Academic perseverance and learning strategies will be *operationalized* using design artifacts from evidence-centered design (ECD) to produce "worked examples" (Gee, 2010; West, Rutstein, Mislevy et al., 2009) that show the complex decisions that need to be made by researchers and software designers in order to develop measures of noncognitive factors. ECD systematically attends to models of student competency, models of the task requirements of an assessment, and models of evidence used to make inferences about assessment results throughout each stage of assessment development. The resulting models, embedded in worked examples, will demonstrate how noncognitive constructs can be assessed across a diverse array of contexts, learning technologies and cyberinfrastructure. We recommend a similar research approach for computational thinking.

Work in the intersection of ECD and educational data mining demonstrates the power of worked examples using ECD as a framework. A recent issue of the *Journal of Educational Data Mining* (Volume 4, No 1, October 2012) showed how ECD can support collaboration among distinct research communities and can support the work of assessing complex learning behaviors in a variety of digital environments. In that issue, authors illustrated their methodologies in context through a set of examples from each research group. These examples allowed readers to consider the impact of design decisions in the analyses conducted by each of the teams. The use of ECD by these groups provided a means to exhibit and compare how they operationalized noncognitive constructs (Nelson, Nugent, & Rupp, 2012) and we believe that a similar approach will bear fruit for computational thinking and its associated practices especially for projects that have started to investigate using student data from the cyberinfrastructure for research and teaching purposes (see Werner, McDowell & Denner, 2013). As Rupp et al. (2012) observe, "The power of ECD lies in unearthing the deep-structure communality of the challenges and associated solutions across projects despite the seemingly surface-structure idiosyncratic facets of each" (p. 6).

Results and Impact. Given the work that has already started in this area (NSF projects CNS 1240625, CNS 1132232) results in 5 years could include the creation of blueprints and templates for assessments in core computational thinking areas in middle and secondary schools (assessments in a specific secondary curriculum have already been developed). Work in this area will impact research to date and going forward by creating commonly accepted learning objectives and assessments that measure them, along with learning-analytics-based exemplars to measure CT in the cyberinfrastructure.

Define, Model and Validate Learning Progressions for K-12 Computer Science. As an extension to the domain analysis of computational thinking and related practices described above, define and validate learning progressions and connections among concepts. These learning progressions will define increasingly complex knowledge, skills, and attributes and will enable the researchers to specify a developmental model of knowledge across the grades, and practitioners to employ pedagogies relying on knowledge of learning progressions, including formative assessments. Learning progressions are based on a domain analysis and involve a pedagogy that emphasizes learning "big ideas" and practices over extended periods (Shea & Duncan, 2013). In science, for example, learning progressions have been defined for biodiversity, the carbon cycle, computer networking, earth system models, genetics, physical science (e.g., force and motion, mass and volume), scientific argumentation, and scientific modeling.

A starting point for research in this area would be the work that ACM/CSTA has done on defining standard learning objectives for grade bands and the domain analysis of computational thinking and related practices described above. Standards describe performance that is the endpoint of a progression, and students enter formal (and informal) learning environments with background knowledge and beliefs. Learning progressions describe intermediate steppingstones that connect the two. Intermediate steps may or may not be important standalone concepts in the domain, but they prepare students for achieving the final step along the progression. Learning progressions also specify connections among constructs that co-develop and may reinforce or support each other (e.g., genes and proteins). Learning progressions are empirically derived and/or validated through multiple iterations, and learning science methodologies (e.g., teaching experiments that collect data such as student work on open-ended assessments and assignments, cognitive think alouds, and classroom video) can be used to verify initial ideas about what students can be expected to know at each grade level. Research can be conducted simultaneously at a close range of grade levels to quickly advance the progressions work.

Empirically derived learning progressions have the benefit of supporting assessment and instruction. If they are defined at the right grain size, they can shape teacher's assessment and instruction practices (e.g., with weekly assessments to determine where students are along a progression) and serve as a complement to the "endpoint focused" nature of standards and learning objectives. Also, evidence-centered design accommodates the modeling of learning progressions with probabilistic techniques (e.g., Bayesian networks, see West, Rutstein, Mislevy et al. (2009)).

Learning progression work is difficult and time-consuming, so we would be guarded in predicting results in 5 years. Because of its potential to improve formative assessment, focusing on a difficult-to-learn and core area such as abstraction would be most impactful. Work in this area will impact learning sciences research to date (none of which has been done in the domain of computer science with a few exceptions, e.g., West, Rutstein, Mislevy, et al. 2009) by demonstrating the similarities and differences in CS vs. other domains.

Understand the Intellectual Demands of K-12 Computer Science Work. Help K-12 teachers understand ways to increase the intellectual challenge of the computer science assignments they give to students and the expectations they place on them. This will drive instruction toward a greater depth of knowledge and cognitive rigor. For example, science content knowledge and knowledge of science processes classify levels of depth of knowledge according to the extent to which students can be expected to (1) recall/identify/recognize/use/calculate (including applying simple procedures), (2) perform multi-step reasoning involving decisions about an approach to the problem, such as classify, organize, estimate, make observations, compare data, graph and the like, (3) explain and justify results of reasoning involving multiple possible answers, drawing conclusions from data, and explaining experiences in terms of concepts, and (4) relating ideas, generalizing, and creating new approaches to solve a problems and doing these over extended periods of time. In computer science, intellectual challenge of assignments may be reflected in asking students to evaluate tradeoffs in designing computational artifacts (rather than just employ programming elements), the degree to which students can create and explain extended programs, and the degree to which assignments are connected to those students will encounter in their lives.

K-12 educators are familiar with depth of knowledge as a rubric for measuring intellectual demand so creating these for computer science concepts can help adoption of curriculum modules structured in this way within 5 years. It will impact research going forward by setting standards or benchmarks for intellectual demands in educator-adapted materials or new approaches developed by the CS education research community.

Understand Implementation Factors in K-12 Computer Science. As CS curricula and programs begin to proliferate we need to systematically investigate implementation: policies and practices at districts and schools, teacher and student backgrounds, teacher practices and professional development, and, in the end, how these factors influence student learning as validly measured by assessments of computational thinking practices. SRI is proposing, to the NSF DRK-12 program solicitation (due December 2013), an implementation study of the Exploring Computer Science (ECS) curriculum across many districts in the US. Working with the ECS developers and the code.org scale-up project involving ECS, we expect, within 5 years, to have a program logic model and supporting evidence of the context factors that influence student learning as measured by our ECS assessments. Work in this area will influence policy, curriculum, and teacher professional development as these are the most likely factors that influence implementation.

New Pedagogies for Cross-Domain Constructs. Understand what pedagogical approaches at the K-12 level are appropriate and effective for cross-domain constructs like problem solving, design thinking, creativity, collaboration (with other designers and end users of designs), and communication. These constructs are a central part of many current conceptions of computational thinking practices and their relationship with core CS content and skills needs to be better understood. Educators will benefit from understanding how to detect and assess student competencies in these areas.

Teach Computer Science in the Context of Other K-12 Subjects. Develop tools that easily let students engage with science and other K-12 disciplinary content computationally so they can participate in authentic scientific experiences that involve computation, especially those involving data analytics. A good starting point is the work of the CT-STEM (ct-stem.northwestern.edu) group at Northwestern.

Promote Early Introductions to Computer Science for Families. For preschoolers and K-4, deliver programs that involve parents in understanding what computer science is about, what course of study it requires, and the economic benefits of pursuing careers in computer science. Good examples, especially for radically underserved youth, are programs for engineering in Detroit and Chicago funded by NSF's ITEST program (Detroit Area Pre-Engineering College Partnership NSF 929557; Chicago Science and Engineering Program, NSF 929544). In these programs, many stakeholders were deeply involved in developing the program, parents (not schools or teachers) apply to participate in a Saturday program at a college campus, and bridges from informal to formal are made through teachers' role in implementing the curriculum.

Identify the Knowledge and Skills of Self-Taught Programmers. Secondary and post-secondary students can teach themselves to program, but little is known about what they learn and precisely how this informal experience influences formal learning. The computer science education field should conduct studies with academically trained computer scientists (through secondary and post-secondary) and self-taught programmers (e.g., in other scientific disciplines and the open-source software community) to understand the differential knowledge, skills, and noncognitive factors each possesses. Conducting such studies can shed light on efforts to promote the participation of underrepresented communities in computer science and help understand how to teach computer science for use in other scientific disciplines.

Design-Based Implementation Research

An issue that runs through all of these research topics is how to produce a valid evidence base in computer science education research. K-12 education research has learned much about ways of doing this that move away from the realm of purely psychological laboratory studies or medical research trials (with its gold standard of randomized controlled experiments) to conceptualize how *design* and *research* can co-occur and inform one another (Barab & Squire, 2004). A methodological approach called design-based research has recently been extended into an approach that also encompasses implementation at scale (Penuel, Fishman, Cheng & Sabelli, 2011; Fishman, Penuel, Allen & Cheng, in press). Design-based implementation research (DBIR) permits the research team to understand what can be expected at any stage, involves systemic views of teaching and learning, and hold promise for sustainability. Build evidence for effectiveness at the same time as design occurs. DBIR involves multiple points of view on persistent problems of education practice, and engages multiple parties in collaborative, iterative design that builds capacity for sustaining change in participants' systems while building theory. In DBIR, plans for scaling and sustainability (in part through changes at institutional levels) are built in from the beginning. Phases of research design and development (e.g., co-design or early implementation) are accompanied by typical driving questions and sources of evidence (e.g., design rationales to explain co-design activities; studies of adaptation of innovations to local contexts for early implementations).

Benefits of Research in These Areas

Progress in the proposed research topics will benefit the US in several ways. First, more precise, concrete and widely accepted definitions of computer science and computational thinking in K-12 subjects along with validated learning progressions will ease the development of assessments and will enable educators and administrators to teach CT and CS in ways that fit the accountability context prevalent in K-12 systems. Trusted assessments of K-12 computer science education will be created that serve as boundary objects for multiple stakeholders (Nolen, Horn, Ward & Childers, 2011) leading to increased adoption and sustainability. Expanding assessment work into learning-analytics-enabled environments will boost the use of cyberlearning for computational thinking and computer science. Educators will benefit from the ability to better formatively assess the knowledge and skills of their students. Identifying the intellectual demands and depth of knowledge of computer science concepts will enable the comparison of rigor across subjects and will demonstrate the rigor of computer science education without the entanglements of programming. Understanding implementation factors in K-12 computer science will directly address the issues needed to scale successful methodologies to new contexts.

Taking mathematics as an example, domain modeling for assessment has led to nuanced expectations for performance which in turn has driven developments in pedagogy and automated learning environments such as intelligent tutoring systems. We can expect to see equally flourishing systems in computational thinking. Crossing domains to understand how 21st century skills such as problem solving and collaboration play out in computer science and how computer science is used in scientific and engineering domains will benefit our understanding of how to teach these CT skills in context. Finally, addressing workforce issues of early interest in the computer science and information technology workforce as well as the knowledge and skills of “self taught” programmers will increase the diversity and strength of our workforce.

As research approaches for computer science education, design-based research, design-based implementation research, evidence-centered design, and learning analytics hold promise for building evidence-based results that add to knowledge in the field of computer science education but also build the practice of computer science educators. The ultimate benefits will be a diverse population of students who are better prepared for post-secondary computer science pursuits, a more computationally literate society, and greater interest (and healthy skepticism) about the role of computer science in our lives.

References

Barab, S., & Squire, K. (2004). Design-based research: Putting a stake in the ground. *The Journal of the Learning Sciences*, *13*(1), 1-14.

Fishman, B.J., Penuel, W.R., Allen, A-R., Cheng, B.H. (Eds.) (in press). *Design-Based Implementation Research.* National Society for the Study of Education Yearbook.

Gee, J. P. (2010). *New digital media and learning as an emerging area and “worked examples” as one way forward*. Cambridge, MA: MIT Press.

Mislevy, R. J., Behrens, J. T., DiCerbo, K. E., & Levy, R. (2012). Design and discovery in educational assessment: Evidence-centered design, psychometrics, and educational data mining. *Journal of Educational Data Mining, 4*(1), 11–48.

National Research Council: Committee for the Workshops on Computational Thinking. (2010). *Report of a Workshop on The Scope and Nature of Computational Thinking*. National Academies Press.

National Research Council: Committee for the Workshops on Computational Thinking. (2011). *Report of a Workshop on Pedagogical Aspects of Computational Thinking*. National Academies Press.

Nelson, B., Nugent, R., & Rupp, A. A. (2012). On instructional utility, statistical methodology, and the added value of ECD: Lessons learned from the special issue. *Journal of Educational Data Mining, 4*(1), 224–230.

Nolen, S. B., Horn, I. S., Ward, C. J., & Childers, S. A. (2011). Novice Teacher Learning and Motivation Across Contexts: Assessment Tools as Boundary Objects. *Cognition and Instruction*, *29*(1), 88–122. doi:10.1080/07370008.2010.533221

Penuel, W. R., Fishman, B. J., Cheng, B. H., & Sabelli, N. (2011). Organizing Research and Development at the Intersection of Learning, Implementation, and Design. *Educational Researcher*, *40*(7), 331–337. doi:10.3102/0013189X11421826

Rupp, A. A., Levy, R., DiCerbo, K. E., Sweet, S. J., Crawford, A. V., et al. (2012). Putting ECD into practice: The interplay of theory and data in evidence models within a digital learning environment. *Journal of Educational Data Mining, 4*(1), 49–110.

Shea, N. A., & Duncan, R. G. (2013). From Theory to Data: The Process of Refining Learning Progressions. *Journal of the Learning Sciences*, *22*(1), 7–32.

Werner, L., McDowell, C., & Denner, J. (2013). *Middle School Students Using Alice: What Can We Learn from Logging Data?* Paper presented at SIGCSE'13. Denver, CO.

West, P., Rutstein, D.W., Mislevy, R.J., et al. (2009). *A Bayes Net Approach to Modeling Learning Progressions and Task Performances.* Paper presented at the Learning Progressions in Science (LeaPS) Conference. Iowa City, IA.