WHITEPAPER

BUILDING TEACHER CAPACITY IN K-12 COMPUTER SCIENCE BY
Promoting Formative Assessment Literacy

Satabdi Basu

Daisy W. Rutstein

Carol Tate

CCNETWORK National Center

InformCS Promoting Formative Assessment Literacy for K-12 Computer Science Education

Building Teacher Capacity in K–12 Computer Science by Promoting Formative Literacy

The National Comprehensive Center

The National Comprehensive Center (NCC) is one of 20 technical assistance centers supported under the U.S. Department of Education's Comprehensive Centers program from 2019 to 2024. The NCC focuses on helping the 19 Regional Comprehensive Centers and state, regional, and local education agencies throughout the country to meet the daunting challenge of improving student performance with equitable resources.

The contents of this publication were developed under a grant from the U.S. Department of Education. However, the contents do not necessarily represent the policy of the U.S. Department of Education, and you should not assume endorsement by the Federal government.

A copy of this publication can be downloaded from https://www.nationalcompcenter.org/.

1. Introduction

The rapid expansion of computer science (CS) instruction in primary and secondary education has highlighted the shortage of teachers qualified to teach the subject. A key strategy for building CS teaching capacity has been preparing teachers of other subjects (e.g., math, technology applications, business) to teach introductory CS through short-term professional development (PD) workshops, online training modules, micro-credentials, and teacher meetups or conferences (Christensen Institute, 2019). These professional learning experiences tend to be brief in duration and focused on short-term adoption of specific CS curricula, instructional activities, programming environments, or technology-enhanced tools at the expense of developing teachers' conceptual understanding of CS standards, learning progressions, and pedagogy for monitoring and supporting students' progress toward the standards. As states and school districts across the country ramp up efforts to increase high-quality CS education in K–12, it is critical to increase the number of qualified CS teachers and to provide existing teachers, many of whom did not originally train to be CS teachers, with the necessary tools, resources, and training to effectively teach CS. This white paper highlights some of the challenges faced by current CS teachers and presents a call to action for states and school districts to support CS teacher capacity building through standards-aligned, sustained, scalable, and reusable teacher PD that focuses on promoting teachers' CS formative assessment literacy as a way to improve teachers' ability to effectively teach CS.

Recently, national organizations and state agencies have introduced guidelines for CS instruction, including the K–12 CS Framework (K–12 Computer Science Framework, 2016), the Computer Science Teachers Association's (CTSA's) CS standards (Computer Science Teachers Association, 2017), and state-specific CS standards in 34 states as of 2019. While these documents map out the domain of K–12 CS, teachers still need additional support on several aspects of CS content knowledge and pedagogy to provide effective CS instruction. Though CS teaching currently differs greatly across and even within states, all teachers need support to understand (a) the fine-grained learning targets associated with various CS standards, (b) what makes CS concepts difficult or easy to learn in different programming environments, (c) common student challenges associated with various standards and environments, and (d) how to teach or assess a particular standard in different environments. Gaining an understanding of these issues provides teachers with the required content and pedagogical knowledge for effective CS instruction. The ability to measure and track students' progress toward target CS standards, and use the assessment information to inform future instruction, is known as *CS formative assessment literacy* and is also a critical aspect of effective CS instruction. It is a critical skill, albeit a challenging one, especially for teachers new to CS who are often self-taught (DeLuca & Klinger, 2010).

As various new CS curricula and activities using different computational representations or programming environments are rolled out each year, it can be difficult for teachers to make sense of how they map to CS standards or fit with existing activities. To benefit a wide range of teachers using a variety of CS curricula across different programming languages, CS teacher PD needs to focus on CS standards and CS content and pedagogical knowledge rather than merely the details of a particular curriculum. Curriculum-based PD was a good first start, but we must now focus as a community on building teacher capacity independent of the curriculum for a particular course. Because of the relative newness of CS in formal K–12 education, vendors have had more influence on the selection of curriculum materials and content of teacher PD than in other disciplines. As teachers are often tasked with teaching new CS curricula using different programming languages, a deeper grasp of the standards can aid teachers in understanding commonalities among curricula and reduce the burden of having to switch to teaching a new curriculum.

Curriculum-based PD was a good first start, but we must now focus as a community on building teacher capacity independent of the curriculum for a particular course.

The next few sections of this white paper detail the elements of effective CS education, known teacher challenges in CS education, and the current state of CS teacher PD. Sections 4 and 5 describe an approach to address some of these teacher challenges and help teachers understand CS standards and develop their CS formative assessment skills. We conclude with recommendations for CS teacher PD design and suggest actions for states and school districts to support CS teachers.

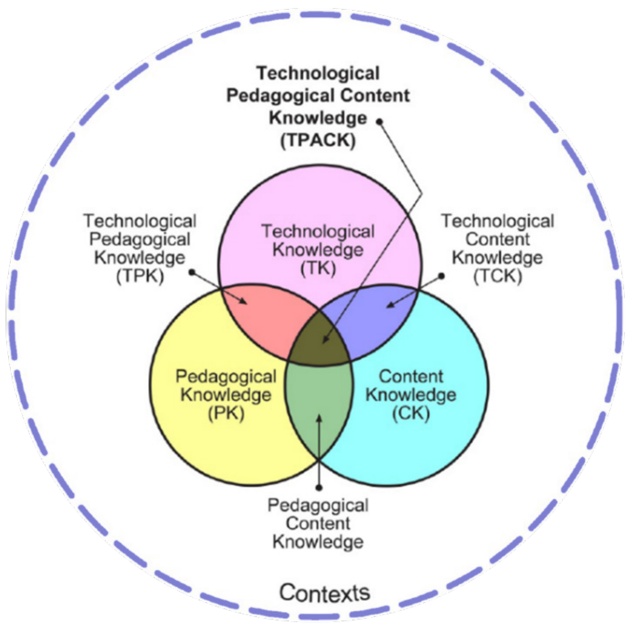
2. Effective Computer Science Instruction and Known Teacher Challenges

What Does Effective Computer Science Instruction Look Like?

Effective CS instruction involves applying both knowledge of the CS discipline and effective pedagogical strategies. It includes the ability to blend content and pedagogy into an understanding of how particular CS topics can be organized, represented, adapted to the diverse interests and abilities of learners, and presented for instruction in a comprehensible and accessible form. This ability to address issues related to content and pedagogy simultaneously and transform subject matter for teaching is the essence of *pedagogical content knowledge,* or PCK (Shulman, 1986). Each content area has unique PCK that may be particularly challenging for teachers who are developing content expertise and pedagogical skills in tandem, as may be the case for many newly minted CS teachers. In fact, CS PCK is not yet fully understood (Hubbard, 2018; Yadav & Berges, 2019) and is typically not emphasized in CS PD.

Additionally, CS instruction often involves technology in the form of programming environments, data visualization and analysis tools, hardware, software, and physical computing devices such as robots, sensors, and microprocessors. Teachers face multiple layers of challenges; they must learn how to use new technologies simultaneously with learning CS content and pedagogy. However, generic context-neutral approaches to learning about technology, which may be available to teachers, do not avail the full potential of technology for teaching specific subject matter. Technology use in the classroom is dependent on context such as subject matter, grade level, student background, and the kinds of computers and software programs available. *Technological PCK*, or TPACK (Mishra & Koehler, 2006), is the basis for effective CS instruction using technology and requires an understanding of (a) representation of CS concepts using relevant technologies; (b) pedagogical techniques that use technologies in constructive ways to teach CS content; (c) knowledge of what makes CS concepts difficult or easy to learn using certain technologies; and (d) how technology can help address some of the problems that students face. Figure 1 illustrates the interplay among technological, pedagogical, and content knowledge in specific contexts (Koehler & Mishra, 2009).

Figure 1. The Technological Pedagogical Content Knowledge framework and its components[1]



[1]Reproduced by permission of the publisher, © 2012 by TPACK.ORG.

Teacher Standards: What Computer Science Teachers Should Know and Be Able to Do

This interplay of technological, pedagogical, and content knowledge is described in the recently released CSTA standards for CS teachers (Computer Science Teachers Association, 2020) that outline what CS teachers should know and be able to do. The teacher standards span five distinct categories: (a) CS knowledge and skills, (b) Equity and inclusion, (c) Professional growth and identity, (d) Instructional design, and (e) Classroom practice. According to the standards, effective CS teachers must have thorough content knowledge and skills in CS, understand the expected student learning progression, and continuously refine their pedagogical content knowledge and skills to support all students in meeting age-appropriate learning outcomes. In particular, the following teacher standards emphasize the need for CS teachers to have a detailed understanding of the student-level CS standards and be able to adapt instruction to address student challenges.

Teacher Standard 4: Instructional Designs for CS:

- Standard 4a: Analyze CS curricula in terms of CS standards alignment
- Standard 4d: Develop strong student conceptual understanding and proactively address student misconceptions in CS
- Standard 4f: Inform instruction through assessment – Develop multiple forms of assessment and use resulting data for instructional decision-making and differentiation

Teacher Standard 5: CS Classroom Practice:

- Standard 5f: Use formative assessments to provide effective feedback to students and to adjust instruction in order to promote stronger achievement in CS

Teacher Challenges: What Computer Science Teachers Need

One way of helping teachers improve their teaching is to identify teacher challenges associated with specific standards and develop PD to provide support in addressing those challenges. In recent studies examining challenges encountered by current CS teachers, teachers reported their biggest issues as insufficient knowledge of CS subject matter, struggles with assessing student work, trouble with meeting individual student needs, inadequate planning time, feeling isolated as the only CS teacher, and IT challenges (Hebert & Worthy, 2001; Yadav et al., 2016). New CS teachers have described how they have taught themselves, their challenges related to lack of CS content knowledge and programming experience, and how these challenges limited their ability to explore CS concepts in depth. Teachers have also revealed their difficulties in meeting all student needs individually and evaluating student learning in classrooms. They have mentioned their challenges in creating assessment items, their lack of access to quality CS assessment tools, and their need for organizing online CS resources and assessments by topic and level. When teachers are self-taught and learn alongside their students, they may lack an understanding of how CS standards can be unpacked into fine-grained learning targets. This makes it difficult to navigate online CS resources and select activities and assessments appropriate for their classroom context and student needs.

Significant strides have been made in addressing some of these teacher-reported challenges around content, pedagogy, and assessments, but teachers' CS PCK and formative assessment literacy practices require more support. The CSTA Assessment Task Force's report (Yadav et al., 2015) highlighted the need for the design and implementation of PD opportunities to help develop and improve CS teachers' assessment literacy and use of classroom assessments.

3. Current State of Computer Science Teacher Professional Development

A review of available teacher PD for CS in K–12 indicates that most PD programs spanned a week or less, were focused on content rather than pedagogy, and were not conducted in collaboration with school or district leadership (Menekse, 2015). These limitations of current CS PD make learning to teach CS difficult for teachers. Further, retraining teachers originally certified to teach other subjects is a unique PD challenge because most teachers are encountering the CS content for the first time. As such, new CS teachers are simultaneously learning what to teach and how to teach it. They are often the sole CS teacher in their schools and thus lack a professional community (Menekse, 2015). For teachers to overcome these challenges and offer quality instruction in CS, they need more than a few days of training on CS content knowledge and use of specific technology-based tools. They need access to a professional community and ongoing contact with discipline-specific, classroom-relevant PD to keep their content and pedagogical knowledge current.

At present, PD for CS teachers is most often provided in a workshop or summer institute format, typically ranging from a few hours to a week. Given that CS is a new subject for many participants, this is not enough time, especially to discuss CS standards in depth or to focus on pedagogy and use of classroom-based assessments in CS instruction.

Research on effective professional learning for teachers has demonstrated that the duration of PD is of key importance. One-time, short-duration PD workshops without structured ongoing engagement have demonstrated limited success (Dickinson & Caswell, 2007; Yadav et al., 2016). Programs that engage teachers in a sustained fashion over a period of weeks, months, or even an academic year are the most successful (Darling-Hammond et al., 2009; Desimone, 2009). At present, PD for CS teachers is most often provided in a workshop or summer institute format, typically ranging from a few hours to a week. Given that CS is a new subject for many participants, this is not enough time to discuss CS standards in depth or to focus on pedagogy and use of classroom-based assessments in CS instruction. High-quality professional learning also includes built-in time for teachers to think about, try out, and receive feedback on new practices (Darling-Hammond et al., 2017). It is through this kind of feedback and reflection that expertise develops.

Also, new CS teachers tend to receive PD that is designed to support adoption of a particular curriculum or set of CS activities. As such, much of the focus is on the use of technology and curriculum, rather than understanding the CS standards, what effective CS pedagogy looks like, and how to support students' progress toward the CS standards. In a recent survey of a national sample of high school CS teachers, 70% of respondents reported that their PD emphasized "deepening their own CS knowledge, including programming," but fewer than 50% reported "learning about difficulties that students may have with particular CS ideas or practices" and "monitoring student understanding during CS instruction" as areas of emphasis (Banilower et al., 2018).

Additionally, there is limited collaboration between staff at institutions of higher education, who typically design and develop PD programs for CS educators, and local school organizations. This poses obstacles for sustaining PD and its long-term effects on teacher practices and student learning (Menekse, 2015). PD is more effective if there is a clear connection to the school context, and this is best accomplished in a community of practice so that teachers can provide mutual support, practice peer assessment, and reflect together on student work (Garet et al., 2001; Sentance & Humphreys, 2018). Because teacher PD in CS has primarily relied on outside funding from the National Science Foundation, nonprofits (most notably code.org and the College Board), and private companies (e.g., Google, Microsoft), this has not helped to nurture long-term, systemic teacher capacity building (Christensen Institute, 2019).

> In a recent survey of a national sample of high school CS teachers, 70% of respondents reported that their PD emphasized "deepening their own CS knowledge, including programming," but fewer than 50% reported "learning about difficulties that students may have with particular CS ideas or practices" and "monitoring student understanding during CS instruction" as areas of emphasis.

In summary, the current state of CS teacher PD poses several challenges to CS teacher capacity building. CS teacher training needs to be a sustained and systematized effort with a logical progression that provides cohesive training for teachers at different grade levels. It must emphasize both content and pedagogy to help teachers expand their understanding of CS concepts and pedagogical tools and assessments. This will allow teachers to gain skills and confidence in how to gather evidence of student learning and use it to differentiate and continuously improve their instruction.

4. Formative Assessment Literacy – An Underemphasized Aspect of Computer Science Teacher Professional Development

As previously described, the need to teach CS content and familiarize teachers with new curricula and technologies within a limited timeframe leaves little opportunity for developing teachers' abilities to assess student progress toward CS standards. When PD focuses on content knowledge, it supports an increase in teachers' knowledge of the domain, but that does not always translate into better student performance (Wiliam, 2018). Using formative assessment practices in a classroom supports student learning and engagement (Herman, 2013; Kingston & Nash, 2011; Popham, 2010), and formative assessments also enhance students' metacognition, goal setting, and sense of accomplishment. Since the ability to use formative assessments as part of instruction is such an essential part of the teacher PCK described in Section 2, and one that has received little attention in teacher PD, the remainder of this paper will focus on how to promote formative assessment literacy in CS. In the words of Dylan Wiliam (2018, p. 27), "While there are many possible ways in which we could seek to develop the practice of serving teachers, attention to minute-by-minute and day-to-day formative assessment is likely to have the biggest impact on student outcomes."

Formative assessment refers to "…the collaborative processes engaged in by educators and students for the purpose of understanding the students' learning and conceptual organization, identification of strengths, diagnosis of weaknesses, areas for improvement, and as a source of information that teachers can use in instructional planning and students can use in deepening their understandings and improving their achievement" (Cizek, 2010, pp. 6–7). Content knowledge is a necessary but not sufficient component of this skill. One of the key differences between formative and summative assessment lies in the purpose and use of the assessment. In a summative assessment, the goal is to reflect the current state of students' understanding and often to provide a score or grade for a student. However, the goals of formative assessment are to support student learning and inform instruction. Formative assessment is a process that involves the identification of important aspects of student learning, the creation (or selection) of assessment tasks to elicit information about student understanding of these aspects, and the use of the resulting information to inform instructional choices—for example, to revisit a certain topic or to address a specific misconception.

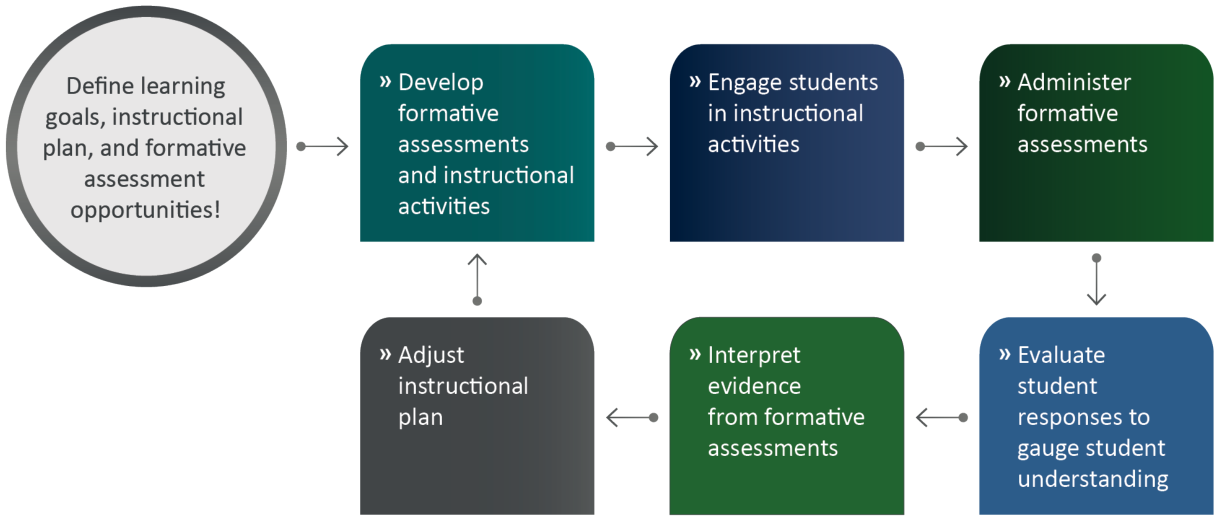
Formative assessment is a process that involves the identification of important aspects of student learning, the creation (or selection) of assessment tasks to elicit information about student understanding of these aspects, and the use of the resulting information to inform instructional choices.

Teachers struggle with multiple aspects of this process. A large survey of secondary teachers (Hoover & Abrams, 2013) concluded that much of the formative assessment data teachers gathered was analyzed at a superficial level and did not maximize the potential for informing instruction. Other studies have similarly shown that teachers struggle with using student assessment data to determine the next steps in student learning (Datnow & Hubbard, 2015; Herman, 2013). This indicates a need for teacher PD that promotes formative assessment literacy (Popham, 2009).

With CS, the first support that teachers often need is to develop an understanding of the CS standards, the relationships between different standards, and the potential learning difficulties for students (Saeli et al., 2011). Deep knowledge of the standards helps ensure that teachers are able to develop learning targets and identify evidence of student progress toward those targets. PD programs should also support teachers in using a wider range of formative assessment strategies beyond multiple-choice items and with interpreting and discussing the results of formative assessments with students (Gottheiner & Siegel, 2012). As teachers develop their CS formative assessment capabilities, they also expand their CS technological and pedagogical content knowledge. The result is a deeper understanding of the CS concepts and practices and how they relate to classroom activities.

Throughout the formative assessment process, teachers determine what it means for students to engage with the concepts and practices of interest and identify and interpret evidence related to student understanding. This requires more than giving students a quiz; it is a full process that can take many shapes and forms. Figure 2 illustrates this process or cycle (Herman, 2013).

Figure 2. Formative assessment cycle



The success CS teachers have in engaging with the following steps in this cycle reflects their CS formative assessment literacy (Cizek, 2010; Popham, 2009):

- ***Define learning goals, instructional plans, and formative assessment opportunities.*** This step entails identifying a set of learning goals and determining an instructional plan, including the identification of timepoints at which formative assessments can be used. Typically, this is after students receive initial instruction on a CS concept and there is the possibility to modify subsequent instruction. However, it could also be before students have received any instruction or during end-of-term review units.
- ***Develop formative assessments and instructional activities.*** Once teachers have identified the instructional plan, including when to administer formative assessments during their instruction, they need to create activities that follow that plan. For the assessments, this includes creating or identifying appropriate assessments and a plan for scoring student responses. The assessments should be aligned with the identified learning targets and provide information on student understanding and challenges with respect to CS content, practices, and the use of technological tools. While the scoring plan should be simple so that student responses can be scored quickly, it should also help teachers differentiate levels of student understanding. Further details on how to develop appropriate assessments are discussed in Section 5.
- ***Engage students in instructional activities.*** This step involves following the instructional plans and engaging in discourse, demonstration, discussion, and other pedagogical moves around unplugged and/or computer-based activities to help students progress toward the identified learning goals.

» ***Administer formative assessments.*** In this step, teachers must administer the formative assessment activity or activities that they have created or identified and collect information on the student responses. This can involve providing a set of questions to be answered online or using paper and pencil in class, or as homework, or during an interactive classroom discussion.

» ***Evaluate student responses to gauge student understanding.*** Once teachers gather student responses on the formative assessment tasks, they must evaluate the responses to understand students' progress toward the learning targets. This could be as simple as determining how many students responded correctly to a task or could involve applying rubrics or scoring guides to student artifacts.

» ***Interpret evidence from formative assessment.*** Once teachers grade student responses, they need to analyze and interpret students' performance to draw conclusions either at the class level or the student level, depending on the purpose of the formative assessment tasks. The interpretation could identify gaps in learning, or strengths and challenges of students, or help categorize students into distinct groups based on their progress toward the learning goals.

» ***Adjust instructional plan.*** Finally, teachers should decide how to adapt their instructional plan based on their interpretations of student performance on the formative assessment tasks. This is one of the most challenging steps for teachers in the formative assessment cycle. In some cases, teachers may try to address a common student misconception in a follow-up lesson. There may be other cases in which the instructional plan does not need to change (e.g., a teacher is checking for understanding and finds that the class as a whole has a good understanding of the concept). The cycle then starts over again with the teacher engaging in the next instructional activity.

The cyclical aspect indicates that formative assessment at one time point is not enough. Instead, teachers can use these practices at various time points in their instruction, including between lessons to help prepare for the next day's lesson, between units to determine if they need to revisit concepts, or during class to help determine the next instructional activity. Research indicates that formative assessment works best when both teachers and students participate. Teachers can use the results of the formative assessment to reflect on their teaching, while students can use the results to reflect on their learning (Herman, 2013; Pinger et al., 2018). This can shift a classroom climate such that both teachers and students take responsibility for learning, and both groups see the value in formal and informal assessments (Popham, 2010).

In the next section, we discuss, with examples, a five-step process for supporting teachers' development of CS formative assessment literacy. While this multi-step process may appear to create extra work, which can dissuade teachers from using it, we would like to point out that teachers are often not starting from scratch for each step. For example, teachers can draw on the work being done by researchers to unpack CS standards into fine-grained learning targets and can reuse or adapt existing formative assessment tasks for their instruction. For experienced teachers, they may be adapting assessment techniques they have used in other content areas. We detail the whole process here to help teachers be intentional in their selection of assessments and better able to use information from student work to inform instruction.

5. A Principled Approach to Develop and Use Computer Science Formative Assessments

When developing any assessment, formative or summative, using a *principled design process* helps focus developers on the decisions they need to make to ensure that the resulting assessment matches the purpose of the assessment (Mislevy, 2007). A principled design process refers to a process that provides guidelines (or principles) to walk the developer through the assessment design process. Engaging a teacher with this type of process supports the teacher's assessment literacy and improves their ability to design or use assessments in ways that support instruction.

The process we next describe is based on a well-known assessment design approach known as evidence-centered design (ECD) (Mislevy & Haertel, 2006; Mislevy & Riconscente, 2006). In this approach, a developer focuses on the evidence that is needed from students and the type of tasks that will produce the evidence. Tasks are specifically designed to ensure that the evidence teachers need is captured in student performance. The main steps of this approach are as follows:

» **Step 1.** Define and/or identify fine-grained learning targets to assess.

» **Step 2.** Determine evidence needed to measure progress towards learning targets.

» **Step 3.** Find or create tasks that elicit the desired evidence.

» **Step 4.** Determine how to evaluate and interpret the evidence provided by students.

» **Step 5.** Relate the interpretation and/or evaluation of the evidence to possible follow-up activities.

Figure 3 shows an example of the type of information expressed in each of the above steps and demonstrates how a single CS standard can be deconstructed into multiple learning goals. Each goal represents a curriculum objective that requires further refinement into fine-grained learning targets to determine what the expectations are for students. Guidelines for task design are specified along with the information on the type of evidence needed and how that evidence might be evaluated. Finally, there are some examples of actions that teachers could take based on the results of the assessments.

While the information in Figure 3 is not meant to be comprehensive, it provides examples of the types of information teachers can use to generate or select tasks. We elaborate more on Figure 3 in this section and have added a separate example in the Practice Guide in which we describe in detail how a standard can be deconstructed into fine-grained learning targets and how formative assessment tasks that align with these learning targets can be designed and used.

Figure 3. Example of information identified to help generate or select an assessment task

Example CS Standard

1B-AP-10
Create programs that include sequences, events, loops, and conditionals

Learning Goal 1
Loops

Learning Goal 2
Conditionals

Learning Goal 3
Events

Learning Target 1

Students are able to identify the results of a simple conditional statement (if/then or if/then/else).

Evidence

Selection of the results of a simple conditional statement from a set of option OR stating the results of a simple conditional statement.

Task Characteristics

Give students a conditional statement and ask what the result or output would be based on a specific scenario.

Evaluation

Determine if the student selected the correct option or stated the correct response.

Teacher Actions

Revisit the concept with students who face challenges. For other students, consider introducing more complex scenarios, e.g. multiple consecutive conditional statements.

Learning Target 2

Students are able to create a simple conditional statement (if/then or if/then/else) when given a narrative description.

Evidence

Creation or identification of a conditional statement (the condition, the consequent statement, and the alternate statement) given a scenario in a narrative non-programming language.

Task Characteristics

Give students a scenario and have them write out an appropriate conditional statement or select a conditional statement to represent the scenario.

Evaluation

Determine if the student's conditional statement aligns with the scenario. If not, determine whether the problem is with the condition or with the consequent and alternate statements.

Teacher Actions

Revisit the concept with students who face challenges. For other students, consider introducing more complex scenarios, e.g. multiple consecutive conditional statements.

Step 1: In the first step, teachers must define the fine-grained learning targets. In some cases, these might be provided to teachers as part of the CS curriculum (e.g., in the form of learning objectives). In other cases, the teacher may be starting with a CS standard or a learning goal that is very broad and encompasses multiple learning targets. A learning target is a statement that is specific and measurable, which means it must focus on one aspect of a domain and be something for which evidence of student performance can be defined. For example, while "Understand conditional statements" may be a learning goal, it is not a fine-grained learning target because it does not clearly define what it means for a student to "understand" a conditional statement. It could mean that students are able to identify a conditional statement, or it could mean that they can interpret a conditional statement, or it could mean that they can create a conditional statement. Breaking down the standards and broad learning goals into fine-grained learning targets helps to identify the critical aspects of student learning in the domain and clarifies what the expectations are for students. For example, the upper elementary CS standard shown in Figure 3, "*Create programs that include sequences, events, loops, and conditionals,*" needs to be broken down into separate learning goals around sequences, events, loops, and conditionals, as each of those aspects may be assessed separately and may be taught at different times. In addition, while the standard specifies that students should be able to create programs that include those concepts, a teacher may want to use learning targets around interpretation of the output of programs comprising these concepts or debugging programs comprising these concepts. This unpacking of the standard highlights aspects that should be covered in instruction and helps identify the different ways a concept might be covered in an assessment.

Two learning targets aligned to the learning goals involving conditionals are shown in Figure 3. Each of these learning targets demonstrates different aspects of how students might engage with conditional statements. While these learning targets are not the only ones that might be developed based on the given CS standard, we use them to highlight how multiple tasks and task types may be appropriate for the same CS standard (depending on which aspect of the standard is the focus of the assessment).

Step 2: Once CS teachers have identified a set of learning targets to assess, they must determine what type of evidence is needed to measure the targets. When determining evidence, it is important to consider what information should be collected. For example, is it sufficient to just look at the final product, or is it necessary to collect information about the process students went through to create the product?

In Figure 3, the evidence for each of the learning targets is stated as something that a student would do or produce. The focus is to identify how the students should engage with the learning targets and how a teacher could observe their efforts. For the learning target that requires students to identify the outcome of a given code segment, the evidence is that students select or state the output; for a learning target in which students create a code segment, writing code is evidence of engaging with the learning target.

Step 3: In this step, the specifications of the assessment task are developed. These can be related to task format, and/or the type of information that the task must provide to students. Different task formats can be chosen based on the degree of feedback desired. For example, if the learning target focuses on creating programs involving specific concepts (e.g., loop or conditional), the task can provide a scenario and a space for students to respond by constructing a program or making a multiple-choice selection. Once the task characteristics are specified, they can be used to develop new tasks or select assessment tasks that follow the specified characteristics.

In Figure 3, notice the alignment between the learning target, evidence, and task characteristic. For learning target 1, we want students to identify the results of a simple conditional statement. The evidence specifies what we mean by 'identify the results,' meaning students can select from a set of options or state the output. To ensure students can select a response, the task characteristics must provide a conditional statement and answer options. Aligning the information in each of these steps helps ensure that if a task meets the task characteristics, the results will provide teachers with information on their students' abilities related to the learning targets.

Step 4: Another critical aspect of formative assessment is the evaluation and interpretation of student responses and the follow-up instructional steps. The rubric or scoring guide should be developed alongside the formative assessment tasks to help ensure that the evidence provided by the assessment measures the desired learning target(s). The rubric may be as simple as checking "yes or no" if students have the correct or incorrect answer when teachers want to get a general sense of how the class is doing. Additional information can be gained from reviewing students' incorrect responses or assessing the answer option students chose, as it may reveal specific student challenges. When developing rubrics, teachers should check the alignment of the rubrics with the learning targets to ensure that the evidence being examined is related to a critical aspect of the learning targets. By developing and using rubrics, teachers can gain insights into how students might engage with the learning targets. This improves teachers' assessment literacy and also increases their understanding of the concepts they are assessing.

Step 5: Once the rubrics are developed, teachers must make decisions about what to do if they find most students struggling on a learning target (perhaps revisit the topic as a class and highlight the challenge that most students demonstrated), versus when only a few students respond incorrectly (perhaps check in individually with those students). It is important to think about how the information from the assessment will be interpreted and used. To support teachers and students, the assessment must provide evidence that it meets the intended purpose. Once the formative assessment tasks are developed and rubrics and follow-up actions are defined, the teacher can then administer the assessments.

Teachers will not always need to develop assessments from scratch. There are available resources—some are included with a curriculum and others may be found in online forums or from colleagues. Even when teachers use an existing assessment, it is still important to use the principled assessment development process described above to determine if the assessment fits. Teachers should define the learning targets and identify the type of evidence needed to measure them before selecting an assessment. Teachers should also examine the rubrics (if provided) to determine if they provide the necessary evidence. Finally, teachers should also consider and/or develop a set of rules for when and how they are going to use the results of the assessment in their classrooms. Available resources may need to be adapted for a particular classroom context or to ensure that the teacher is obtaining useful and actionable feedback about students. Ensuring coherence between the formative assessment tasks, the learning targets, and the evidence derived from the tasks will be invaluable for teachers' CS PCK.

Adaptations for Online Instruction

Designing and conducting formative assessments while teaching CS in an online environment presents special challenges, but the same general principles that apply for face-to-face instruction hold good. Whether students are participating in a synchronous online class meeting or completing assigned lesson activities asynchronously, it is still possible and important to gather feedback to inform instruction.

Similar to the process described in Sections 4 and 5, the first step in using formative assessments in online instruction is for teachers to precisely identify the learning targets about which they would like to gather evidence, and determine the timepoints in their teaching at which this feedback would be most helpful. In an asynchronous class, this may mean posing questions as checkpoints for students to pass through before they can proceed in the lesson. In a synchronous class, online platforms may offer the option for polling or other in-the-moment checks for understanding. These can be used before moving on in a lesson or as a bell ringer review or exit ticket.

Once teachers develop and administer the assessment(s), they need to consider how to react to the feedback provided by student responses. Real-time feedback on student work may be challenging in an online environment, limiting the types of suitable assessment formats. An important consideration for teachers planning to use formative assessment during online instruction is the timeframe before the next instructional move. For example, if students must answer a question before they can proceed in a lesson, teachers will want to decide if it should be machine scorable so that the delivery platform can automatically send the student the next activity or whether the teacher should have time to read student responses, score them manually, and respond to students individually. Similarly, if teachers are using the poll features of videoconference software or using student engagement platforms (e.g., Kahoot, Nearpod, Pear Deck) to gauge overall student understanding, they should consider how they will use results of such polls to inform their next instructional steps.

6. Recommendations for Teacher Professional Development in Computer Science

Quality teacher PD is crucial for building the content knowledge, pedagogical skills, and confidence that teachers need to teach K–12 CS effectively (Fischer et al., 2018; Kim & Kim, 2018; Yadav et al., 2016). In particular, CS teachers need support on how to use formative assessment to inform their teaching practices. An explicit focus on developing teacher PCK in general, and formative assessment literacy in particular, is essential to promote equitable CS classrooms. Issues of equity and inclusion are often addressed in PD through efforts to change teacher beliefs and foster positive student identities. These are certainly important, but access to high-quality instruction is equally important. Niral Shah and colleagues (2013) developed a framework for equitable CS instruction that includes four dimensions: rich course content, quality instruction, identities as computer scientists, and peer relationships. In this framework, quality instruction is characterized by tracking student progress, customizing teaching plans to individual students, and practicing formative assessment.

In general, PD for CS should follow all the known features of effective PD approaches (Darling-Hammond et al., 2017), including focusing on content, incorporating active and collaborative learning, and offering time to learn, practice, implement, and reflect. CS PD efforts focused on readying teachers to start teaching a particular CS curriculum need to be expanded to provide sustained support to CS teachers. The following are some recommendations for teacher PD in K–12 CS.

- **Include formative assessment literacy practices for CS** – Formative assessment literacy outcomes for teachers include: (1) deeper understanding of CS standards and how students develop the knowledge and skills described in the standards, (2) increased knowledge of how and when to assess the CS knowledge and skills described in these standards, and (3) use of assessment data to differentiate instruction and address specific student misconceptions and barriers to learning. CS PD programs should provide support for teachers to achieve all of these outcomes. Specifically, they should provide the following:
  - ***Opportunities for teachers to author assessment tasks*** – Introducing teachers to principled assessment development frameworks like ECD and providing opportunities to author assessment tasks aligned with specific learning targets will help teachers engage with formative assessment practices.
  - ***Opportunities for teachers to examine student work***. There are many different types of student artifacts created in CS curricula, and teachers need guidance on how to evaluate them. It is critical to provide teachers with examples of how students might engage with the CS concepts and practices at different levels of sophistication. Examining student work should enable teachers to reflect on students' proficiency on CS standards and identify the next steps to support students. As teachers develop or choose formative assessments, these types of activities can help them determine what types of assessment formats to select and how to group students based on their level of proficiency.

- **Focus on CS student standards and connections between standards and lesson activities** – Since many teachers are engaging with the CS standards for the first time, CS PD should familiarize teachers with the standards, what mastery of a standard looks like, and how lesson activities are aligned (or misaligned) to the standards. Teachers must understand which parts of each standard a curriculum covers and which parts are not covered. This allows teachers to see how lessons align with the broader CS domain and identify critical junctures during lessons to focus on formative assessment practices.
- **Align CS PD to CS teacher standards** – As described in Section 2, the Standards for Computer Science Teachers (Computer Science Teachers Association, 2020) describe what CS teachers should know and be able to do, and include the ability to use PCK to inform instruction through assessment. Hence, it would be beneficial to teachers and PD developers if all CS teacher PD was mapped to the teacher standards, including the need for formative assessment literacy.
- **Emphasize teachers' content knowledge, pedagogical practices, and use of technology** – In Section 2, we described the challenges CS teachers face with organizing and representing CS subject matter so that it is accessible for all students. CS teacher PD that focuses on teachers' ability to engage in ongoing formative assessment can support teachers' CS content knowledge, pedagogical practices, and use of technology.
- **Support both student engagement and learning outcomes** – Many introductory CS programs focus on sparking student interest, but turning that initial interest into sustained engagement and learning requires that teachers learn how to track and communicate student proficiency as it develops. Formative assessment practices allow both teachers and students to see, and be motivated by, growth in students' CS expertise.

While these recommendations can help improve the quality and effectiveness of CS teacher PD, it is also important to think about the supports given to CS teachers throughout the year. A PD program may provide different check-ins during the year or could help set up communities of practice for teachers to work together across schools and districts. Providing teachers opportunities to discuss their instruction and formative assessment practices in a supportive professional community can help address challenges related to feelings of isolation and help CS teachers develop their own learning and teaching practices.

7. How Can State and Local Education Agencies Take Action?

There are substantial differences in CS teaching from state to state, district to district, and even from school to school, which makes it difficult to generalize and build community around teacher PD, course content, and assessment. Because of the relative newness of CS as a school subject, especially in K–8, vendors have more influence on selection of curriculum materials and content of teacher PD in CS than in other disciplines. This has often led to a piecemeal implementation of CS education. As CS moves toward being a core subject, state and local education agencies (SEAs and LEAs) will have an important role in creating an environment in schools that supports cohesive CS programs aligned with CS standards and building the teaching capacity and quality to implement these programs. SEAs are well-positioned to aggregate resources and tools and disseminate guidance on best practices for choosing course materials and designing teacher PD. Next, we outline some recommended action plans for SEAs and LEAs to expand and sustain K–12 CS and build CS teaching capacity and quality.

Recommendations for SEAs

- **Create a statewide plan for K–12 CS if one does not already exist.** This should include state-specific CS standards, a formative assessment literacy development plan, and one or more dedicated CS content leads to steer statewide implementation of CS and communication with LEAs.
- **Aggregate resources, tools, curricula, assessments, and research on K–12 CS, highlighting promising resources and practices and issuing recommendations for CS implementation and for use of formative assessment practices.** Convene experts and veteran teachers to rigorously evaluate and recommend curriculum materials and assessments aligned to the state's CS standards and strategies.
- **Allocate funding for statewide rigorous CS teacher PD that includes formative assessment literacy, both for teachers new to CS and continuing CS teachers.**
- **Establish standards for CS teacher PD providers and evaluate and select PD providers using criteria similar to the PD recommendations articulated in Section 7.**
- **Seed efforts to develop an ongoing community of practice that will support teachers' CS formative assessment literacy.** For example, promote a pilot project building a network of teachers working on developing formative assessment practices.
- **Develop policy for CS teaching credentials and endorsements for both pre-service and in-service teachers.** Ensure that formative assessment literacy is included in certification of CS teachers.
- **Work with institutes of higher education (IHE) and industry experts to improve teacher preparation and support teachers.** Create programs at IHEs for certification or supplementary authorization to teach CS. Industry professionals can also provide expert help, and teachers will benefit tremendously from a platform that offers them on-demand access to experts (Christensen Institute, 2019).

Recommendations for LEAs

- **Establish a CS committee to track the quickly evolving landscape of CS in K–12.** Share findings with the local school boards and identify specific district needs.
- **Evaluate the needs of districts and schools in terms of support for teacher capacity to engage in formative assessment for CS across the grade bands.**
- **Establish a community of practice for CS teachers to combat isolation and receive ongoing support for professional growth.** If the CS faculty is small, develop regional partnerships to support a vibrant professional community.

» **Draw on the expertise of highly qualified CS teachers** to support more novice teachers through contributions to PD, facilitation of professional learning communities, and individual coaching.

» **Ensure funding allocation for sustained and systematic CS PD that includes a focus on teachers' use of formative assessment.**

8. Conclusion

With the rapidly evolving K–12 CS landscape, there is an acute need for trained CS teachers. A lack of qualified teachers is currently considered the biggest barrier to CS education in the United States (Patel, 2017). In the words of Code.org's Pat Yongpradit, "Laptops, robots, and 3D printers are important, but they don't make a CS class. A trained teacher makes a CS class" (Patel, 2017, p. 1). In this white paper, we have outlined the challenges related to content, pedagogy, and assessments that now confront teachers in CS classrooms and have highlighted gaps in current models of teacher PD.

Quality CS teaching requires a nuanced understanding of the complex relationships between technology, content, and pedagogy; it also requires the skill of using this understanding to develop appropriate, context-specific strategies and representations to help all students learn. Knowledge of the target CS standards and common student challenges helps teachers better understand and appreciate various curricular activities and progressions, and an enhanced understanding of formative assessment literacy practices empowers teachers to evaluate their own teaching and gauge their students' progress toward the standards.

Quality CS teaching requires a nuanced understanding of the complex relationships between technology, content, and pedagogy; it also requires the skill of using this understanding to develop appropriate, context-specific strategies and representations to help all students learn.

We recommend CS teacher PD that promotes formative assessment literacy by providing supports for teachers' exploration and understanding of the CS standards, demonstrating links between the standards and instructional activities, and providing ways to evaluate different levels of student understanding. Examining assessment tasks, rubrics, and student responses to such tasks can be helpful to new CS teachers learning what evidence of proficiency on a CS standard looks like in practice. These activities can help teachers identify the types of information they want to gather about their own students. Accordingly, this white paper includes some recommendations for teacher PD in K–12 CS and some recommendations for staff at state and local education agencies that will support teacher capacity building. These include allocating funding for rigorous and sustained teacher PD and incentivizing the use of standards-aligned formative assessments as part of K–12 CS instruction. As teachers become better prepared to teach CS in K–12, we hope to see effects on student engagement, retention, and learning in CS that will help students better navigate today's increasingly digital society.

Acknowledgments

We would like to acknowledge colleagues at SRI International and beyond who have provided constructive feedback and contributed in different ways to improving the white paper. In particular, we are grateful to Dr. Carol Fletcher (Director, Expanding Pathways in Computing, The University of Texas at Austin) and Mr. Sam Berg (Computer Science Coordinator, Oakland Unified School District) for providing their expert reviews on this white paper. We would also like to thank our SRI colleagues Andrea Beesley, Carrie Parker, Elizabeth McBride (now at AI4ALL), Katrina Laguarda, Laura Kassner, Lori Hodges, and Nonye Alozie for their contributions.

References

Banilower, E. R., Smith, P. S., Malzahn, K. A., Plumley, C. L., Gordon, E. M., & Hayes, M. L. (2018). *Report of the 2018 NSSME+*. Horizon Research, Inc. http://horizon-research.com/NSSME/wp-content/uploads/2020/04/Report_of_the_2018_NSSME.pdf

Christensen Institute. (2019). *The silver lining of computer science teacher shortages.* https://www.christenseninstitute.org/blog/the-silver-lining-of-computer-science-teacher-shortages/

Cizek, G. J. (2010). An introduction to formative assessment: History, characteristics and challenges. In H. L. Andrade & G. J. Cizek (Eds.), *Handbook of formative assessment* (pp 3–17). Taylor & Francis.

Computer Science Teachers Association. (2017). *CSTA K–12 computer science standards, revised 2017.* http://www.csteachers.org/standards

Computer Science Teachers Association. (2020). *Standards for computer science teachers.* https://csteachers.org/teacherstandards

Darling-Hammond, L., Hyler, M. E., Gardner, M. (2017). *Effective teacher professional development.* Learning Policy Institute.

Darling-Hammond, L., Wei, R. C., Andree, A., Richardson, N., & Orphanos, S. (2009). *Professional learning in the learning profession.* National Staff Development Council.

Datnow, A., & Hubbard, L. (2015). Teachers' use of assessment data to inform instruction: Lessons from the past and prospects for the future. *Teachers College Record*, *117*(4), 1–26.

DeLuca, C., & Klinger, D. A. (2010). Assessment literacy development: Identifying gaps in teacher candidates' learning. *Assessment in Education: Principles, Policy & Practice*, *17*(4), 419–438.

Desimone, L. M. (2009). Improving impact studies of teachers' professional development: Toward better conceptualizations and measures. *Educational Researcher, 38*(3), 181–199.

Dickinson, D., & Caswell, L. (2007). Building support for language and early literacy in preschool classrooms through in-service professional development: Effects of the Literacy Environment Enrichment Program (LEEP). *Early Childhood Research Quarterly*, 22, 243–260.

Fischer, C., Fishman, B., Dede, C., Eisenkraft, A., Frumin, K., Foster, B., Lawrenz, F., Levy, A. J., McCoy, A. (2018). Investigating relationships between school context, teacher professional development, teaching practices, and student achievement in response to a nationwide science reform. *Teaching and Teacher Education, 72,* 107–121. https://doi.org/10.1016/j.tate.2018.02.011

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,* 915–945.

Gottheiner, D. M., & Siegel, M. A. (2012). Experienced middle school science teachers' assessment literacy: Investigating knowledge of students' conceptions in genetics and ways to shape instruction. *Journal of Science Teacher Education, 23*, 531–557.

Hebert, E., & Worthy, T. (2001). Does the first year of teaching have to be a bad one? A case study of success. *Teaching and Teacher Education*, *17*(8), 897–911.

Herman, J. (2013). *Formative assessment for next generation science standards: A proposed model.* Invitational Research Symposium on Science Assessment. CRESST.

Hoover, N. R., & Abrams, L. M. (2013). Teachers' instructional use of summative student assessment data. *Applied Measurement in Education*, *26*(3), 219–231.

Hubbard, A. (2018). Pedagogical content knowledge in computing education: A review of the research literature. *Computer Science Education*, *28*(2), 117–135.

K–12 Computer Science Framework. (2016). http://www.k12cs.org

Kim, S., & Kim, H. Y. (2018). A computational thinking curriculum and teacher professional development in South Korea. In *Computational thinking in the STEM disciplines* (pp. 165–178). Springer.

Kingston, N. & Nash, B. (2011). Formative assessment: A meta-analysis and a call for research. *Educational Measurement: Issues and Practice*, *30,* 28–37.

Koehler, M., & Mishra, P. (2009). What is technological pedagogical content knowledge (TPACK)? *Contemporary Issues in Technology and Teacher Education, 9*(1), 60–70.

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.

Mishra, P., & Koehler, M. J. (2006). Technological pedagogical content knowledge: A framework for teacher knowledge. *Teachers College Record*, *108*(6), 1017–1054.

Mislevy, R. J. (2007). Validity by design. *Educational Researcher, 36*(8), 463–469.

Mislevy, R. J., & Haertel, G. (2006). *Implications of evidence-centered design for educational testing*. SRI International.

Mislevy, R. J., & Riconscente, M. M. (2006). Evidence-centered assessment design: Layers, concepts, and terminology. In S. Downing & T. Haladyna (Eds.), *Handbook of test development* (pp. 61-90). Erlbaum.

Patel, P. (2017). More teachers, fewer 3D printers: How to improve K–12 computer science education. *IEEE Spectrum* [Tech Talk blog].

Pinger, P., Rakoczy, K., Besser, M., & Klieme, E. (2018). Learning environments research. *Dordrecht, 21*(1), 61–79.

Popham, W. J. (2009). Assessment literacy for teachers: Faddish or fundamental? *Theory Into Practice*, *48*(1), 4–11.

Popham, W. J. (2010). *Everything school leaders need to know about assessment*. Sage.

Saeli, M., Perrenet, J., Jochems, W. M., & Zwaneveld, B. (2011). Teaching programming in secondary school: A pedagogical content knowledge perspective. *Informatics in Education*, *10*(1), 73–88.

Sentance, S., & Humphreys, S. (2018) Understanding professional learning for computing teachers from the perspective of situated learning. *Computer Science Education, 28*(4), 345–370.

Shah, N., Lewis, C. M., Caires, R., Khan, N., Qureshi, A., Ehsanipour, D., & Gupta, N. (2013). Building equitable computer science classrooms: elements of a teaching approach. In *Proceeding of the 44th ACM Technical Symposium on Computer Science Education* (pp. 263–268).

Shulman, L. S. (1986). Those who understand: Knowledge growth in teaching. *Educational Researcher*, *15*(2), 4–14.

Wiliam, D. (2018). *Embedded formative assessment*. Solution Tree Press.

Yadav, A., & Berges, M. (2019). Computer science pedagogical content knowledge: Characterizing teacher performance. *ACM Transactions on Computing Education (TOCE), 19*(3), 29.

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*. http://csta.acm.org/Research/sub/Projects/ResearchFiles/AssessmentStudy2015.pdf

Yadav, A., Gretter, S., Hambrusch, S., & Sands, P. (2016). Expanding computer science education in schools: Understanding teacher experiences and challenges. *Computer Science Education*, *26*(4), 235–254.