A Practical Guide to Developing and Validating Computer Science Knowledge Assessments with Application to Middle School

Philip Sheridan Buffum
Computer Science

Eleni V. Lobene
Computer Science

Megan Hardy Frankosky
Psychology

Kristy Elizabeth Boyer
Computer Science

Eric N. Wiebe
STEM Education

James C. Lester
Computer Science

North Carolina State University
Raleigh, NC, USA
{psbuffum, eleni.lobene, rmhardy, keboyer, wiebe, lester}@ncsu.edu

ABSTRACT

Knowledge assessment instruments, or tests, are commonly created by faculty in classroom settings to measure student knowledge and skill. Another crucial role for assessment instruments is in gauging student learning in response to a computer science education research project, or intervention. In an increasingly interdisciplinary landscape, it is crucial to validate knowledge assessment instruments, yet developing and validating these tests for computer science poses substantial challenges. This paper presents a seven-step approach to designing, iteratively refining, and validating knowledge assessment instruments designed not to assign grades but to measure the efficacy or promise of novel interventions. We also detail how this seven-step process is being instantiated within a three-year project to implement a game-based learning environment for middle school computer science. This paper serves as a practical guide for adapting widely accepted psychometric practices to the development and validation of computer science knowledge assessments to support research.

Categories and Subject Descriptors

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

General Terms

Human Factors.

Keywords

Computer science education, assessment, middle school.

1. INTRODUCTION

Assessing student knowledge is a crucial task across many levels of computer science education, as in any discipline. College and university faculty are most familiar with tests that they themselves author in order to measure their students' understanding and skill in course-related topics. These *assessment instruments* constitute an active area of investigation within the SIGCSE community [16, 17], and their role is often prominent in assigning student grades. Another crucial role for assessment instruments of computer science skills and knowledge is within research projects that aim to measure the promise or efficacy of interventions [7, 15]. While computer science educators and computer science education researchers have the expertise to craft assessment items for computing skills and concepts, these same educators and researchers often do not have formal training in establishing the *validity* (the extent to which a test measures what it is intended to measure) and *reliability* (the consistency of an instrument across administrations of that test) of the assessments they create. Only a small number of computer science education research projects have emphasized these steps [15, 17], and creating assessments is challenging for computer science due to the complexity and dynamic nature of the content [12]. Yet, in an increasingly interdisciplinary research landscape, it is becoming crucial to validate assessment instruments on which research findings rely [13, 14]. Moreover, developing assessments of student learning is an urgent area of need for the relatively young computer science education community as it advances toward the ranks of more mature disciplines such as physics that have established standardized assessments over time.

The wealth of literature on instrument validation within education, psychometrics, and other disciplines is vast. The approaches in much of this literature rely upon administering instruments repeatedly to the same students, or administering them to many hundreds or even thousands of students. These techniques are impractical for the vast majority of computer science education research projects. This paper presents a **practical approach to validating knowledge assessment instruments** that has been developed over several years of collaboration between computer scientists, computer science education researchers, educational psychologists, and an industrial/organizational psychologist. First, we present widely accepted practices for establishing validity and reliability of knowledge assessment instruments. Second, we describe the application of this approach to the ENGAGE project, a computer science education research project focused on developing a game-based learning environment for middle school computer science. This case study illustrates how, within the constraints of a typical research project, and with the availability of only a handful of classrooms in which to operate, widely accepted practices can be adapted to validate and iteratively refine computer science knowledge assessment instruments.

SIGCSE '15, March 04 – 07 2015, Kansas City, MO, USA
Copyright 2015 ACM 978-1-4503-2966-8/15/03 $15.00
http://dx.doi.org/10.1145/2676723.2677295

2. BACKGROUND

In psychometrics, to be effective a test must be both *valid* and *reliable* [2]. Validity refers to the extent to which a test, referred to as an instrument, measures what it was intended to measure. Validity is of crucial concern when constructing knowledge assessment instruments because authors of test items hold extensive expertise in the target domain, and the test items that these experts write intending to discriminate students' knowledge of a particular concept may, in practice, be testing a different skill. There are three primary types of validity: construct, criterion-related, and content [3]. Establishing *construct validity* is a process of testing and confirming inferences in measurement and prediction. This complex form of validation involves investigating the relationships between the theoretical variable and the measure for both the test and relevant outcomes. This type of validity can include a number of advanced statistical techniques to investigate each relationship. Alternatively, *criterion-related validity* is the extent to which the test scores correlate with other variables, or other related tests, as one would expect. Finally, *content validity* is how well the items, or questions, match the desired variable definition based on the opinion of subject matter experts.

Of the three types of validity, establishing content validity is an important step for assessments of educational interventions, and it is the most accessible type of validity study to perform in computer science education research. While there is no uniformly accepted process for establishing content validity, generally the steps include creating a panel of experts in the domain of interest and providing a structured framework for collecting and analyzing the panel's feedback on the items. Sections 3 and 4 will describe this process in more detail.

Reliability is the consistency of a test's measurement or the extent to which an instrument is free of error [2]. In other words, if an individual were given the same test multiple times, assuming there was no learning effect between administrations (and no testing effect from the instrument itself [11]), that individual would score nearly the same every time. However, it is often not practical to give the same test multiple times to the same individual; doing so requires securing an "empty treatment" control group that does not participate in the intervention. Additionally, for K-12 learners whose understanding of target computer science concepts could be affected by their learning in related subject areas such as mathematics, science, and technology, the "no learning effect" assumption is called into question.

There are a number of types of reliability estimates. For example, *inter-rater reliability* is used when two or more trained raters independently score a test [8]. This is an appropriate method for qualitative items, such as free responses, where scoring the responses may be subjective. Another kind of reliability estimate is *internal consistency*, which is based upon the extent to which items are correlated within a test itself [5]. If the test being developed is multiple-choice, typically a statistical measure called Cronbach's alpha is utilized to establish internal consistency. Cronbach's alpha can be computed using many off-the-shelf statistical packages (e.g., SPSS or SAS). In general, a score of 0.7 and above is desirable to indicate a "sound" test.

3. PRACTICAL GUIDELINES FOR TEST CONSTRUCTION

The process of constructing an assessment instrument, much like the process of building software artifacts, can proceed in many different ways depending upon the goals of the project and the expertise of the team members. This section presents a seven-phase process for developing an assessment instrument that can be used as both a pre- and post-test, addressing the open need in computer science education research for evaluating the strength of a given intervention. This process has been created and honed within a multi-year collaboration between the authors, a team including computer scientists and psychologists.

1. **Identify the purpose of the test.** It is important to articulate the purpose of the test, whether it is intended to assess conceptual knowledge, problem solving, other skills such as, for example, computational thinking or collaboration. This purpose guides the remaining phases of test development.

2. **Define the construct of interest.** Based upon the determined purpose of the test, defining the construct of interest typically involves a content analysis of the domain guided by literature and observations. For computer science projects that involve pre-defined curricula, guidance for the constructs of interest is often found within the curriculum and learning objectives.

3. **Prepare the test specifications.** There are numerous **test formats** ranging from multiple-choice to open-ended essay questions, with many in between. The choice of which type of test to use often comes with tradeoffs: open-ended questions can provide rich insight into student learning, but these items can be labor-intensive to grade and reliability of the grading must be established to ensure consistency and objectivity. On the other hand, multiple-choice questions are straightforward to grade but can be labor-intensive to construct and do not provide the same variety and richness of student responses as open-ended questions. Nonetheless, multiple-choice tests are scalable to many students, and many validation techniques including those detailed in this paper are aimed at multiple-choice tests.

   If a multiple-choice test format is selected, the literature provides guidance on the **number of responses per question**, which should never exceed five to nine for simple items [6]. For computer science knowledge assessments, four response options may be appropriate in order to minimize extraneous cognitive load, since the items are often not simple. Additionally, four choices strikes a balance with the likelihood of false positives from guessing, with an expected 25% rate of correct responses attributed to random chance alone.

   Another important specification is **test length**, which in classroom settings and many computer science education research projects is constrained by the available contact hours for administering the tests. For example, if a middle school class period is 45 minutes long then a test duration of 30 minutes may be an appropriate target to allow for classroom management activities that typically surround the beginning and ending of a class period. While there are no hard and fast rules for how many items should be included on an instrument [9], in our experience of developing learning assessments, adequate reliability is typically observed with tests that include at least 20 items. More items provides a more substantial basis for establishing validity, but "testing fatigue" can cause the quality of student responses to degrade as tests become lengthier, which is of particular concern for younger students.

4. **Generate the test items.** With the question format and desired length selected, the question writers are trained (if necessary), possible questions are generated, and the quality of the questions is checked. It is recommended that test developers write twice as many questions at this stage as are

needed on the final test, because many candidate test items will be eliminated during the next phases.

5. **Conduct a formal review for validity of the candidate test items.** While there are many forms of establishing validity, a highly recommended approach is to obtain the independent feedback of three to five subject matter experts. Depending upon the specific test purpose, these experts may all share the same general background (e.g., computer science faculty at postsecondary institutions) or may comprise a set of complementary expertise (e.g., one computer science faculty member, one middle school mathematics teacher, one middle school science teacher, and one educational psychology faculty member). Each expert will rate the question according to a pre-determined rubric that could be as simple as a binary "include/exclude" recommendation to, for example, a five-point Likert rating. Additionally, each expert may provide specific feedback for refining and improving test items. Following this process, the strongest set of refined items is retained, and the rest excluded. It is recommended that this set of items still be larger than the ultimately desired test length since some items are likely to be eliminated after piloting.

6. **Pilot the test items with a representative sample from the population of interest.** The closer that these students are to the target population being studied, the better. For K-12 research studies, for example, this means choosing students ideally in the same grade as the target population of study, and when possible, students who are at a similar point of maturity within the grade (for example, students who have just entered seventh grade are different from students who are in the final week of seventh grade). Additionally, a number of considerations typically contribute to determining the required number of respondents needed to adequately assess the quality of an instrument [9]. The type of analysis the test developer is planning to use, and the number of items on the test, typically drive recommendations ranging from 80-200 participants. In the experience of our interdisciplinary project team, adequate statistical power and reliability is achieved with a sample of approximately 100 students.

   Once pilot data are collected, a number of statistical analysis techniques can be used to investigate the quality of the items on a test and then refine accordingly. *Item analysis* describes a set of simple statistics, based on Classical Test Theory [2], which can help differentiate good and bad test questions and improve overall test quality. Usually item analysis includes procedures to determine "difficulty" and "discrimination" as well as looking at item-total correlations. *Difficulty* refers to the percent of students who correctly respond to an item. It is preferable to have a range of difficulty scores (excluding those that are excessively difficult or easy) on a test so that some items are harder than others, and thereby the test will have the ability to differentiate performance levels. *Discrimination* is the percentage of students who got the item correct in the group of highest performers, minus the percentage of students who got the item correct in the lowest group of performers. This index indicates how well an item distinguishes among high and low performers. Generally discrimination values of less than 0.2 are considered poor and suggest dropping an item. Values between 0.2 and 0.3 are marginal and may call for revision. Values between 0.3 and 0.4 are acceptable and values above 0.4 are considered good. Finally, item-total correlations (ITCs) are the correlation between the total test and each individual item. Typically, it is recommended that items with ITCs less than 0.3 be dropped because such a low score indicates a lack of consistency with the rest of the test.

7. **Iteratively refine and re-test.** Using the data collected when piloting, the final stage of test development involves refinement and re-testing. The reliability and item analyses, described previously, guide this process. Regardless of test format, most assessment instruments are iteratively refined from year to year or data collection to data collection, as test "development clearly involves a bit of art as well as a lot of science" [9].

4. DRAFTING AN ASSESMENT FOR MIDDLE SCHOOL COMPUTER SCIENCE

Informed by the psychometrics and psychology literatures described in Section 2 and following the test development processes described in Section 3, we are drafting an assessment for middle school computer science knowledge. This assessment is being created as part of the ENGAGE project, a three-year project to build and investigate the promise of an immersive game-based learning environment (Figure 1) that teaches a subset of the CS Principles course to middle school learners [4]. The focus is on grade seven but the learning environment is intended to be accessible to grades six through eight.

Because ENGAGE's three-year project timeline necessitates developing the intervention (the game-based learning environment and the accompanying lesson plans and in-class learning materials) concurrently with the assessment items, test questions were developed by a subset of the project team while the learning environment and curricular content were developed by another subset of the team. Designing such assessments for computer science knowledge is difficult. They are however crucial for evaluating the strength of interventions, such as ENGAGE's game-based learning environment. Even at the relatively young stage of middle school, students enter interventions with widely varying degrees of prior computer science knowledge. In order to understand how effective the game is at teaching students computer science, the students' *learning gains* need to be assessed, rather than merely their success in the game. This section describes the seven steps of the test development process as they were instantiated.

4.1 Identifying the Purpose of the Test

Step 1 of the test creation process is to identify the purpose of the test. For many computer science education research projects, the purpose of a test is to determine whether the novel intervention was effective in supporting learning for students. This is the case for this project, whose knowledge assessment has a purpose of, "Measure students' computer science learning from interacting with the ENGAGE game-based learning environment."

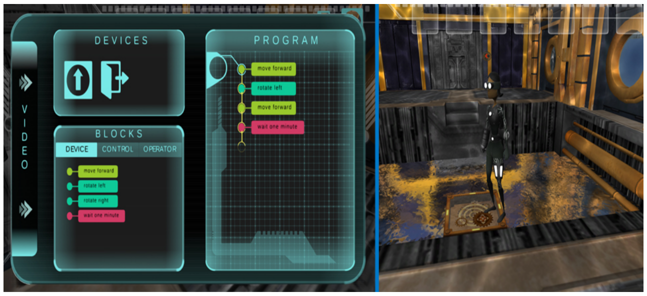


Figure 1. The ENGAGE game-based learning environment

4.2 Defining the Construct of Interest

Step 2, which is to define the construct of interest, was informed by the CS Principles curriculum [1] learning objectives and evidence statements, taken in concert with the US Common Core standards [10]. Crucially, expert input from middle school teachers within a summer workshop in 2013 guided the team in specifying the constructs of interest [4]. Through this process, some CS Principles evidence statements were designated as *overarching*, intended to permeate many if not all the learning challenges in the game. An example of one such overarching evidence statement is 5.1.3.B: *Collaboration facilitates multiple perspectives in developing ideas for solving problems by programming*. Other evidence statements are more *focused*, in that a specific portion of the game-based learning environment will address this concept. Determining which evidence statements to designate as focused, as well as the order they would appear, helped guide the design of ENGAGE's game-based learning environment and its accompanying assessment instruments.

With this correspondence between the intervention and the knowledge assessments, we have a clear understanding of where in the game a student should be expected to master the content needed to correctly answer a given assessment item. This crucially allows the team to use the knowledge assessment as a way to improve the game-based learning environment. For example, if students exhibit low learning gain on a particular assessment item, and the item itself is determined to be sound, this can point to an area of improvement for the intervention itself. An example of this is given in section 4.7.

4.3 Preparing the Test Specifications

In developing the knowledge assessment items, we decided to use a consistent multiple-choice format with four options per item. As described in the previous section, multiple-choice allows for objective, automatic grading, and reduces the subjectivity and manual labor that can be seen with open-ended questions. Moreover, students at the middle school level are known to sometimes contribute as little as possible to open-ended assessment items, and multiple-choice items mitigate this risk.

The target test length was selected as 20 questions, the minimum that has been observed in the team's research experience to provide sufficient reliability coefficients. The smallest plausible number of questions is desirable because the amount of classroom time available for assessments is limited. Only a small fraction of class days can be devoted to assessment, and moreover, in addition to the CS knowledge assessments, the research project requires administering numerous other instruments as well. Class time and testing fatigue are important considerations for the overall length of all instruments combined.

4.4 Generating the Test Items

With the goal test length of 20 questions, the research team set out to author 40 candidate items. Importantly, because the test items are tied closely to the learning objectives that were ultimately decided upon for inclusion in the game-based learning environment, the test was authored in several stages corresponding to the three "levels" of the game. In each phase a computer science education researcher generated the assessment items based upon the focused evidence statements [1] for each game level.

4.5 Conducting a Formal Review

The items underwent expert review in two formats. At first, the computer science education researcher who was new to writing test items held collaborative feedback sessions with an expert panel consisting of at least one other computer science education researcher, two computer science professors, an educational psychologist, and a human factors psychologist. During these face-to-face meetings recommendations for inclusion/exclusion of items were made verbally and discussed, and feedback for improving items was given. In later phases of test item authoring, the same experts provided feedback one at a time in series with refinements between. After several iterations of refinement based upon this expert panel content validation, it was determined that the assessment instrument was ready for piloting.

4.6 Piloting the Test Items

Two pilot studies were conducted in Spring 2014 at an urban middle school in the southeastern United States. Four teachers at this school agreed to have their classes participate in the studies.

For the first pilot, 103 sixth to eighth grade students played the entire first level of the game, which took approximately an hour for most students. ENGAGE emphasizes collaborative problem solving, so most students played the game in pairs on the same computer. In order to assess individual knowledge, however, each student completed assessments individually (taking them concurrently on separate computers). Of the 103 students, 43 were female and 60 were male. Race/ethnicity demographics were not collected from all students, but for those who responded, 32 identified as African-American, 4 Asian, 14 Hispanic, 31 White, and 6 Other. The version of the knowledge assessment in this first pilot consisted of six items taken both before and after interacting with the game-based learning environment. These six items represented the first set of knowledge assessment items (out of a total target of 20), covering the first of three levels of gameplay. Although more test items had been drafted, a small set was given in this round of piloting for two reasons: first, these six items corresponded to the only level of gameplay being piloted in this study, and second, other cognitive and affective survey instruments were being piloted in the same study, constraining the time available for piloting the knowledge assessment test items.

For the second pilot, 42 sixth and seventh graders returned to play a segment of the second level of the game, which took approximately half an hour for most students. The same protocol of paired gameplay and individual assessment was followed as in the first pilot. Of the 42 students, 17 were female and 25 were male. Of those who supplied race/ethnicity demographics, 10 identified as African-American, 3 Asian, 6 Hispanic, 13 White, and 3 Other. This pilot focused on ten questions: five were directly relevant to the gameplay segment from this pilot and were administered both pre and post. The other five items corresponded to future gameplay segments that had not yet been developed, and were only given pre. The reason for administering these items sooner than their corresponding gameplay segments were fully developed is because, as will be discussed in the next section, a ceiling effect (too many students answering items correctly even before being exposed to the learning content) was noted on several of the test items from the first pilot, and it was desirable to allow more time to revise and address any ceiling effects or other issues with the newest items.

4.7 Iteratively Refining and Re-Testing

Table 1 summarizes the fifteen items that were piloted within these first two studies, including the percent of students who answered each item correctly on the pre- and post-test. Based upon these statistics as well as finer grained response choice breakdowns, the items were refined.

Table 1. Evaluation of 15 knowledge assessment items piloted in Spring 2014

Item	Concept	Pre-test %	Post-test %	Notes
Pilot 1 (ENGAGE Tutorial Level)				
000	Sequencing	54	77	***Dropped***
001	Sequencing	56	61	One option too obviously wrong, so that option replaced with more plausible option
002	Iteration	37	38	Fine as written. During gameplay, a popular misconception was observed, so we replaced the least popular option with a distractor for this potential mistake
003	Domain Vocab: "run"	53	65	Ceiling effect suggests that some students enter with higher domain knowledge
004	Security	72	67	***Dropped***
005	Broadcast	35	49	Substantial revisions needed on the options (but not question itself)
Pilot 2 (ENGAGE Digital World Level)				
002	Iteration	36	---	Fine as written
100	Selection	16	---	One option too obviously wrong
101	Selection & Sequencing	63	---	One option too obviously wrong
102	Variables	74	---	Revision needed on the question itself (too easy)
103	Variables & Selection	37	---	One option too obviously wrong
104	Vocab: "binary"	42	27	***Dropped***
105	Vocab: "binary number"	26	32	Fine as written
106	Domain Vocab: "bit"	37	21	Fine as written. Results suggest revising game.
107	Interpret Binary: one bit "on"	10	47	Fine as written
108	Interpret Binary: multiple bits "on"	26	37	Fine as written

A particularly important consideration for determining the suitability of individual test items involves ceiling and floor effects. Ceiling effects occur when test items are "too easy," meaning that students tend to get the item correct more frequently than expected. Floor effects, in contrast, occur when test items are "too hard," and fewer students than expected answer the item correctly. On a pre-test, it is expected for most students to randomly guess the answers to questions since they have not encountered the learning material yet. With four response options in multiple-choice format, the random chance correctness rate would be 25%. As Table 1 shows, many of the test items display a substantially higher correctness rate on the pre-test. There are several possible reasons for this, one of which is that one of the response items was too clearly incorrect (it was not a viable distractor). This was found to be the case for some items. For others, the team hypothesizes that the instructions to each test item may have been too extensive, "teaching" prior to the pre-test. A floor effect is generally not problematic on a pre-test, but for items where a floor effect was observed on the post-test, reasons for this effect included that ENGAGE's game-based learning environment was not teaching that concept as well as intended, and in some cases as shown in Table 1 this feedback shaped the next steps of the development team to improve the game.

5. Example Test Items and Discussion

This section provides examples of three test items that were refined in different ways based upon pilot testing.

Example 1: Iteration. Item 002 was authored to assess student's understanding of iteration with an item in which they need to use a REPEAT block to navigate a moving platform to a specific location. Each of the four response options contains a sequence of program blocks in the same style as the ones used in the gameplay. The correct option is a sequence that starts with REPEAT 2, with a sequence of four various movement blocks nested underneath the REPEAT block. There was an acceptable distribution of responses on the pre-test for this item, with 37% answering correctly. However, during pilot gameplay observation the research team noted a common misconception: putting the REPEAT at the *end* of a sequence of blocks, rather than at the beginning. None of the original distractor options reflected this misconception, so accordingly the option that had received the lowest number of responses on the pre-test was replaced with this variation (Figure 2). In the second pilot, the new option received the largest number of incorrect responses (32%). This is a desirable outcome, because students will be exposed to the correct construct while interacting with ENGAGE, and this learning can now be captured at a fine-grained level from pre-test to post-test.

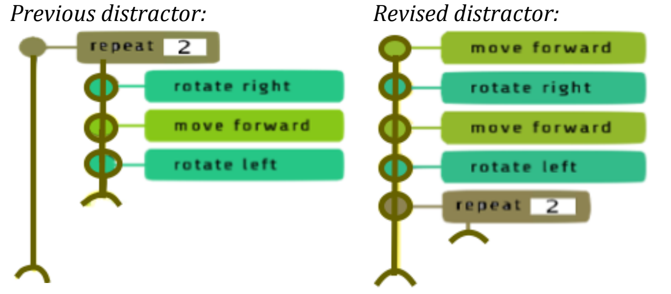


Figure 2. Original and revised distractor for *iteration* item

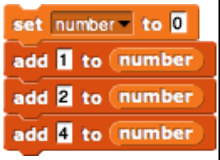
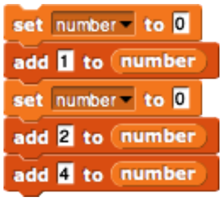
102.What will “number” equal at the end of this program?			
Version1:  set number to 0 add 1 to number add 2 to number add 4 to number	a) 0 b) 1 c) 4 d) 7	*Version 2:*  set number to 0 add 1 to number set number to 0 add 2 to number add 4 to number	a) 0 b) 4 c) 6 d) 7

Figure 3. Items on assessing understanding of *variables*

Example 2: Variables. Variables were not covered in the gameplay levels that students completed for these pilot studies, but the test items for variables were piloted within the pre-test. They were excluded from the post-test since there would be no expectation of students learning the concept during the gameplay. Figure 3 shows two version of Item 102. Version 1 appeared on the pre-test, and students overwhelmingly answered this item correctly by selecting option D, with 74%. All other students except one selected the distractor option C. The ceiling effect on pre-test indicated that it was too easy to get this question correct by chance, so the revised Version 2 requires thinking more deeply about the behavior of a variable.

Example 3: Domain Vocabulary. Item 106 reads, “How many bits does the binary number 100 have?” with four number choices (3, 4, 50, 100). This question had a slightly high but acceptable distribution of responses on the pre-test, with 37% of students selecting the correct option, 3. However, that percent dropped to 21% correct on the post-test, with more students selecting incorrect choices of 4 and 50. In this case, after careful consideration and observation of gameplay, the research team concluded that the game levels were not using this vocabulary word prominently enough, which led to revision of the game itself rather than the assessment item.

6. CONCLUSION AND FUTURE WORK

This paper has presented an approach to developing computer science knowledge assessment instruments, based upon literature from psychometrics and psychology. Creating these assessments for computer science education research projects is vital for two key reasons. Firstly, the ability to assess an intervention’s strength is an essential component to empirical research. Assessments are extensively validated in sister fields such as physics education; the time is ripe for emulating this practice in the comparatively young field of computer science education. Secondly, these assessments can provide invaluable insight into how to make targeted refinements to the given intervention. As shown here, a process of iterative refinement is essential to formulating effective test items, and this process can often take the same length of time (and run concurrently with) the research project that develops the intervention itself.

There are many promising avenues for future work, chief among these being the expanded development of validated knowledge assessment instruments at all levels of computer science. ENGAGE’s knowledge assessment, described in this paper, stands as an exemplar of an assessment for seventh grade students, and the process described in the paper provides guidance for how to design analogous knowledge assessments at other levels of K-12 education. The more of these instruments that are created, validated, and shared across research projects and interventions, the greater common foundation can be built for measuring and supporting student learning. These knowledge assessment instruments play a central role in the give-and-take of research for developing effective interventions that support rigorous computer science education.

7. ACKNOWLEDGEMENTS

This work is supported in part by the National Science Foundation through Grants CNS-113897 and CNS-1042468. Any opinions, findings, conclusions, or recommendations expressed in this report are those of the participants, and do not necessarily represent the official views, opinions, or policy of the National Science Foundation.

8. REFERENCES

[1] AP® Computer Science Principles Draft Curriculum Framework: 2014. *http://www.csprinciples.org/*. Accessed: 2014-09-05.

[2] Ayala, R.J. De 2008. *The Theory and Practice of Item Response Theory*.

[3] Binning, J.F. and Gerald V. Barrett 1989. Validity of Personal Decisions: A Conceptual Analysis of the Inferential and Evidential Bases. *Journal of Applied Psychology*. 74, 3 (1989), 478–494.

[4] Buffum, P.S. et al. 2014. CS Principles Goes to Middle School: Learning How to Teach “Big Data”. *SIGCSE ’14* (2014), 151–156.

[5] Cronbach, L.J. 1951. Coefficient Alpha and the Internal Structure of Tests. *Psychometrika*. 16, (1951), 297–334.

[6] Eli P. Cox III 1980. The Optimal Number of Response Alternatives for a Scale: A Review. *Journal of Marketing Research*. 17, (1980), 407–422.

[7] Franklin, D. et al. 2013. Assessment of Computer Science Learning in a Scratch-Based Outreach Program. *SIGCSE ’13* (2013), 371-376.

[8] Gwet, K.L. 2012. *Handbook of Inter-Rater Reliability*. Advanced Analytics, LLC.

[9] Hinkin, T.R. 1998. A Brief Tutorial on the Development of Measures for Use in Survey Questionnaires. *Organizational Research Methods*.

[10] National Governors Association Center for Best Practices 2010. *Common Core State Standards*.

[11] Roediger, H.L. and Karpicke, J.D. 2006. The Power of Testing Memory Basic Research and Implications for Educational Practice. *Perspectives on Psychological Science*. 1, (2006), 181–210.

[12] Shuhidan, S. et al. 2010. Instructor Perspectives of Multiple-Choice Questions in Summative Sssessment for Novice Programmers. *Computer Science Education*. 20, 3 (2010), 229–259.

[13] Sudol, L.A. and Studer, C. 2010. Analyzing Test Items: Using Item Response Theory to Validate Assessments. *SIGCSE ’10* (2010), 436–440.

[14] Tew, A.E. and Guzdial, M. 2010. Developing a Validated Assessment of Fundamental CS1 Concepts. *SIGCSE ’10* (2010), 97–101.

[15] Tew, A.E. and Guzdial, M. 2011. The FCS1: A Language Independent Assessment of CS1 Knowledge. *SIGCSE ’11 (2011)*, 111–116.

[16] Vasilevskaya, M. et al. 2014. An Assessment Model for Large Project Courses. *SIGCSE ’14 (2014)*, 253–258.

[17] Werner, L. et al. 2012. The Fairy Performance Assessment : Measuring Computational Thinking in Middle School. *SIGCSE ’12* (2012), 7–12.