

Interpreting the ABET Computer Science Criteria Using Competencies

Rajendra K. Raj
Rochester Institute of Technology
Rochester, New York
rkr@cs.rit.edu

Mihaela Sabin
University of New Hampshire
Manchester, NH
mihaela.sabin@unh.edu

Amruth N. Kumar
Ramapo College of NJ
Mahwah, NJ
amruth@ramapo.edu

John Impagliazzo
Hofstra University
Hempstead, NY
john.impagliazzo@hofstra.edu

ABSTRACT

Since the early 21st century, ABET's accreditation criteria have focused on learning outcomes (what students learn) rather than what professors teach. Such accreditation criteria bring to bear the need for programs to establish clear learning objectives and assessment processes that ensure that program graduates have the requisite technical and professional preparation. To this end, ABET defines *student outcomes* as "what students are expected to know and be able to do by the time of graduation," further noting that these outcomes "relate to the knowledge, skills, and behaviors that students acquire as they progress through the program." With the recent release of Computing Curricula 2020 (CC2020), the competencies of computing program graduates have received additional attention. CC2020 describes *competency* as "comprising knowledge, skills, and dispositions that are observable in accomplishing a task within a work context."

ABET's student outcomes thus largely correspond to the CC2020 competencies of program graduates. This paper is a first attempt to reconcile the two notions in the context of computer science. It presents the relevant background and discusses student competencies and their assessments that focus on competency-based learning in computer science. The contributions of this paper are (1) forging an improved shared understanding of computing competencies and (2) an interpretation of ABET's student outcomes to improve the competency, including dispositions, expectations of computer science graduates.

CCS CONCEPTS

• **Social and professional topics** → **Computing profession; Accreditation; Computer science education.**

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](https://permissions.acm.org).

SIGCSE 2022, March 3–5, 2022, Providence, RI, USA

© 2022 Association for Computing Machinery.

ACM ISBN 978-1-4503-9070-5/22/03...\$15.00

<https://doi.org/10.1145/3478431.3499293>

KEYWORDS

Computer science education, competency-based learning, computing competencies, ABET, accreditation criteria, student outcomes.

ACM Reference Format:

Rajendra K. Raj, Amruth N. Kumar, Mihaela Sabin, and John Impagliazzo. 2022. Interpreting the ABET Computer Science Criteria Using Competencies. In *Proceedings of the 53rd ACM Technical Symposium on Computer Science Education V. 1 (SIGCSE 2022)*, March 3–5, 2022, Providence, RI, USA. ACM, New York, NY, USA, 7 pages. <https://doi.org/10.1145/3478431.3499293>

1 INTRODUCTION

For around two decades, ABET's accreditation criteria used for computer science (CS) have attempted to focus on learning outcomes (what students learn) rather than what is taught [13, 29]. Thus, programs must establish clear objectives and assessment processes that ensure that their graduates have the requisite technical and professional preparation. ABET currently defines *student outcomes* as "what students are expected to know and be able to do by the time of graduation," which include "the knowledge, skills, and behaviors that students acquire as they progress through the program" [1].

The recent release of Computing Curricula 2020 (CC2020) has brought attention to the competencies of computing program graduates [10]. CC2020 describes *competency* as comprising *knowledge, skills, and dispositions that are observable in accomplishing a task within a work context*. With subtle differences in terminology and extent, ABET's student outcomes more or less occupy the same space as the CC2020 competencies of program graduates.

Competency is not a novel idea, and its roots span centuries and even millennia. From ancient times, building pyramids, palaces, and river dams required competent professionals to design and engineer such wonders [24]. They had to integrate knowledge and skills coupled with a sense of professional behavior. For decades now, schools of law, medicine, and education have been educating students using competency as part of their professional practice and licensing [20, 23]. However, competency always occurs in a context relevant to the discipline; being competent in law does not imply being competent in medicine. It is not an exaggeration to state that today, computing runs the world. Academics and practitioners thus must bring to bear the same professional practices to designing, implementing, and maintaining hardware and software today as they did to bridges and turbines in yesteryears. Achieving this goal

calls for actively promoting student competency by programs in CS and assured through its accreditation criteria.

This paper focuses on developing a shared understanding of CS competencies building on CC2020 and how ABET's student outcomes can define competency expectations of graduates of CS programs. The following section looks at the background in developing competency-based learning in computing. Section 3 focuses on the movement toward incorporating competencies in program curricula, especially *professional dispositions*, the component of competency that has remained elusive if not esoteric in computing education. Section 4 takes a fresh look at ABET Computing Accreditation Commission's (ABET/CAC's) student outcomes for CS to show that they are inherently supportive of competency-based learning, even though ABET has not fully used them as such. We conclude with some remarks about the need for ABET and its accredited programs to move more strongly toward a future that includes not just knowledge and skills, but also professional dispositions.

2 BACKGROUND IN CS COMPETENCIES

Most undergraduate CS students seek employment after graduation. They need to have the necessary real-world competencies to perform activities needed to succeed in professional careers. Using the CC2020 definition [10], a graduate needs to have the knowledge, skills, as well as appropriate professional dispositions. That is, undergraduate CS programs need to focus on all three aspects of competency:

- (1) developing knowledge,
- (2) enabling skills, and
- (3) inculcating the needed professional dispositions to work on goal-oriented tasks in professional settings.

Existing notions of competencies in computing generally stem from the concepts introduced in information technology curricular guidelines (IT2017) [25], subsequently refined in CC2020 [10], along with more recent curricular guidelines for other computing disciplines, such as information systems [27] and in part, computer engineering [16]. The data science document recently released by ACM and AIS explicitly focuses on competencies [11]. Frezza et al. [14] also took a broader approach to computing competencies.

There have also been efforts focused explicitly on understanding competencies in computing disciplines. For example, the CC2020 report included an attempt to develop a draft set of CS competencies [10, Appendix C.2.2]; however, this attempt was influenced by knowledge areas of CS2013 [2] rather than by a holistic look at appropriate competencies for CS. An ITiCSE 2020 working group explored how CS competencies can be generated using CC2020 approach [9]. Clear et al. [8] examined the transition between knowledge-based and competency-based computing education. Other practical competency and skills frameworks have also been developed, such as ISO 247773-2019 [17], the Institute of Coding's accreditation standard [6], and Skills Framework for the Information Age (SFIA) [26].

In 1996, the Computer Science Accreditation Commission (CSAC) published outcome-based accreditation criteria [13], and in 2000, ABET's "new" Engineering Criteria 2000 (EC2000) [29] helped to focus computing and engineering accreditation on student learning

outcomes. Both emphasized *what is learned* rather than *what is taught*. Besterfield-Sacre et al. [5] highlight that these outcomes should be "observable and measurable manifestations of applied knowledge". Among other things, EC2000 helped engineering programs innovate and develop flexible curricula that ensure that program graduates possess appropriate technical and professional skills expected by employers.

CSAC's and ABET's reformed accreditation criteria used for computing and engineering programs represent a significant change from the "old-school" approach (of ensuring whether the program curriculum covered a checklist of topics) into a more modern process (of checking whether students were learning their program's advertised student outcomes). As a result, computing and engineering educators have initiated reform actions making assumptions about the nature of the competency construct without really exploring its underlying meaning and implications for program evaluation. As discussed in the next section, it is time to rethink the past and consider competency-based learning as a new learning initiative.

3 TOWARD PROFESSIONAL DISPOSITIONS

This section discusses the gradual transition in the ACM computing curricula projects and related reports from knowledge-based to competency-based education. The *knowledge* aspect of competency has received emphasis since the beginning, with the *skills* aspect receiving some attention in recent years. However, the *professional dispositions* aspect has not received as much attention. Therefore, this section builds up the notion of professional dispositions in CS, as there is a compelling need to develop a shared understanding of this aspect of competency.

3.1 Knowledge-based Curricular Reports

Knowledge has been the primary focus of computing curricula from the start. Curricular guidelines in CS had their beginnings with ACM's Curricula'68 [3] and Curricula'78 [4]. The 1980s witnessed many debates on transforming curricula. Information systems specialists rejected earlier recommendations and developed their own guidelines and IEEE produced computer engineering guidelines as a separate entity. In 1991, ACM and IEEE jointly developed Computing Curricula'91 [28], which promoted knowledge areas and knowledge units for computing, accompanied by topics and outcomes.

CC'91 [28] did not achieve broad acceptance because existing programs desired recommendations at the course level, not topics organized in seemingly random order. The world witnessed ongoing expansion of information systems guidelines [12] in 1997. In 2001, ACM and IEEE-CS produced Computing Curricula 2001, abbreviated as CS2001 and CC2001 [19]. The document conceded that CS was a subset of all computing and left the door open for new curricular recommendations. Subsequently, ACM and IEEE-CS created computer engineering and software engineering guidelines in 2004. CC2005 Overview report published in 2005 used graphs and tables to illustrate that computing education had distinct curricula areas in computer engineering, computer science, information systems, information technology, and software engineering [18]. In 2008, ACM published IT2008, the first information technology curricular

guidelines [22]. An intermediate update of computer science was produced in 2008 [7], followed by a full update in 2013 [2].

Until recently, most of the computing curricular reports were primarily knowledge-based efforts. Competencies were addressed beginning with IT2017 guidelines [25] and the graduate MSIS2016 guidelines [27]. ACM and AIS published IS2020 for undergraduate information systems in 2020 [21] as a competency-based recommendation. ACM published a competency-based data science report [11] in 2021, recognizing that data science graduates had to demonstrate competencies to be successful in their careers. CS2013 report primarily focused on knowledge, but mentioned in passing that computer science graduates needed specific skills and professional dispositions [2, Chapter 3]. Finally, CC2020 report [10] attempted to give competency-based examples for each of the computing disciplines under its umbrella.

3.2 Skills Appropriate for CS

Some of the skills expected of CS graduates as listed in CS2013 [2] included:

- understanding the interplay between theory and practice.
- thinking at multiple levels of detail and abstraction.
- applying the knowledge gained to solve real problems.

The consideration of skills by computer science educators reflects workplace expectations: employment job postings almost always advertise for skills, not just knowledge, or subsume knowledge to skills. Once the industry generally eliminated probationary training periods for fresh graduates, the development of skills and skill sets became the responsibility of computing programs and educators.

Educators can address and have addressed the skills gap by many of the following methods.

- (1) Adding a practicum component to required and elective courses.
- (2) Making the practicum count toward the final course grade.
- (3) Moving introductory courses to closed-lab models and using apprenticeship-style learning in the courses.
- (4) Adding a capstone course in the major that expects students to demonstrate broad-based learning.
- (5) Incorporating team projects to show how the industry develops computing solutions in general, thereby playing up the importance of collaboration and teamwork.
- (6) Including a required internship experience.
- (7) Including a required or elective cooperative experience.

Scoring rubrics have been used to assess students' skill levels. Some programs with active industrial advisory boards have assisted in developing methods of skill assessment. However, there has not been much standardization or adoption of these methods for skills assessment across all CS programs.

3.3 Dispositions Appropriate for CS

Recent computing curricular reports have formally recognized the *professional dispositions* aspect of competency. CS2013 mentions professional dispositions in passing [2, chapter 3] without referring to them as professional dispositions.

Table 1: Professional Dispositions from CC2020 [10]

Disposition	Synonyms
<i>Adaptable</i>	agile, changeable, flexible, universal, versatile
<i>Collaborative</i>	collective, communal, concerted, cooperative, team-player
<i>Inventive</i>	clever, creative, exploratory, innovative, imaginative
<i>Meticulous</i>	accurate, attentive, careful, detailed, thorough
<i>Passionate</i>	commitment, compelling, fervent, intense, vehement
<i>Proactive</i>	farsseeing, forehanded, independent, provident, visionary
<i>Professional</i>	accomplished, adept, ethical, masterful, skillful
<i>Purpose-driven</i>	achiever, determined, goal-driven, persistent, tenacious
<i>Responsible</i>	accountable, amenable, dependable, reliable, trustworthy
<i>Responsive</i>	agile, prompt, reactive, receptive, respectful
<i>Self-directed</i>	ambitious, determined, distinctive, independent, unique

The CC2020 report [10] has highlighted eleven professional dispositions expected of all undergraduate computing students, including CS. Table 1 identifies these eleven elements and provides five synonyms to assist in determining the meaning of each disposition.

3.4 Competency as a 3-D Model

This subsection takes a more detailed look at the notion of competency to develop a shared understanding of competency in general and professional dispositions in particular. The nomenclature surrounding professional dispositions has varied by different phrases, such as human attributes, behaviors, attitudes, ability, habits of mind, and other human-related terms.

Publications have characterized the notion of competency in various ways. Both the IT2017 and the CC2020 reports have used:

$$\text{Competency} = \text{Knowledge} + \text{Skills} + \text{Dispositions}$$

for a task or context. The above characterization suggests that the components of competency are additive. The reports have also described competency as an intersection of the three components:

$$\text{Competency} = K \cap S \cap D$$

This is illustrated in Figure 1. These models suggest that there can be skills without knowledge and dispositions without skills or knowledge. This interpretation may be accurate for the whole individual; for example, a student may be persistent in life (a professional disposition from Table 1), but not necessarily in the context of computing endeavors. But, when applied to learning, we believe that the three co-exist, whether one chooses to consider all three dimensions or not.

We propose a three-dimensional representation with knowledge, skills, and dispositions as three axes and competency as a point in this 3-D space as depicted in Figure 2. The horizontal K-axis represents computing knowledge, the vertical S-axis represents

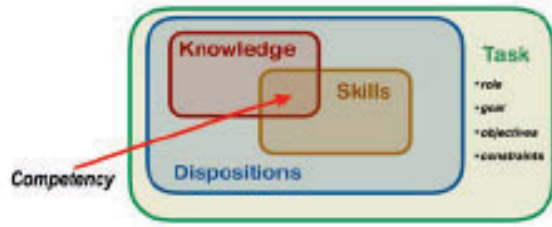


Figure 1: Knowledge, Skills, and Dispositions, as shown in CC2020 [10, figure 4.1]

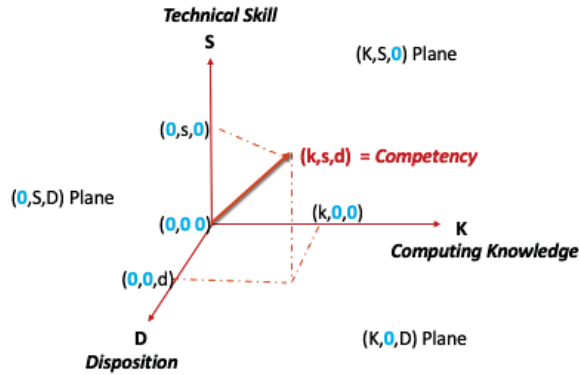


Figure 2: Knowledge, Skills, and Dispositions in 3-D

computing skills, and the D-axis (perpendicular to the page) represents dispositions. Any point on K-axis represents consideration of computing knowledge without regard to computing skills or dispositions. Thus, for example, the point $(K,0,0)$, representative of early curricular guidelines, focuses on knowledge areas to the exclusion of skills and dispositions [3].

Similarly, the three planes illustrate situations that ignore a single dimension. For example, the K-S plane (back wall) represents curricular guidelines focused on knowledge and skills while leaving out dispositions.

One should not view competency as additive ($K + S + D$) or conjunctive ($K \cap S \cap D$), but rather as “projective”: we do not build up to competency but rather peel off elements of competency to talk about individual components. That is, we project competency on either the Knowledge or Skill axis or on the Knowledge X Skill plane. We are not “adding” professional dispositions to prior discourse, but rather, prior discourse chose to ignore professional dispositions while discussing competencies.

3.5 Clarifying Professional Dispositions

The notion of competency has sometimes been viewed as controversial, as it is relatively new in CS. We attempt to identify and address the primary reasons for the confusion. IEEE-CS proposed the software engineering competency model [15], IT2017 defined the computing competency concept and its interrelated components [25], which was refined by CC2020 in its own definition [10]. These attempts not only emphasize knowledge and skills but they

also make professional dispositions explicit and salient in educational discussions to better prepare CS students.

Dispositions deal with behavioral traits, and computing educators typically do not have the qualifications to teach them. However, students can learn dispositions from observation and experience. So, when using a competency model for computing, it is not to enjoin an educator to be the “sage on the stage”—who knows how to teach professional dispositions, but rather to be the “guide on the side”—who encourages and helps students develop professional dispositions. We would argue that the latter is necessary to help students develop professional dispositions.

Can we inculcate dispositions in an academic setting, or do they need a professional setting to develop, such as an internship? While professional settings are the most authentic venues for learning professional dispositions, teachers can foster their development in academic settings via appropriate environments and activities.

CS educators cannot summatively evaluate all professional dispositions. However, they can formatively model dispositions. So, while computing educators may not be able to administer a test to “quantitatively measure” certain professional dispositions of students, they can “qualitatively promote” the development of professional dispositions through reflection activities that use guiding prompts and Likert-scale questionnaires. Competency models suggest that educators employ such low-stakes assessments in their courses to emphasize the importance of such dispositions for students’ careers, thereby helping students develop their professional dispositions.

Are professional dispositions inborn or nurtured through experiences and observations? We believe that they are a little of both, depending on the individual and their lived experiences. However, by making their importance explicit, educators can attempt to have each CS student address the need to develop professional dispositions in their life the best way they can.

Professional dispositions and character traits are sometimes interchangeably used. The distinction is that professional dispositions are observable, whereas character traits may not be. Observability is what makes it possible to foster professional dispositions and formatively assess them.

Professional dispositions are personal dispositions that are relevant in a professional setting. So, they are a subset of personal dispositions. Since the professional setting varies from one discipline to another, professional dispositions are by definition, discipline-specific. In other words, each domain (computer science, information technology) should identify a set of professional dispositions most appropriate for it.

Is the list of discipline-specific professional dispositions subjective or objective? We believe it is a list that should be arrived at by consensus by the professionals in the discipline. As such, it may be too much to hope for a canonical list of dispositions. It is also worth noting that some professional dispositions may have universal application, whereas others may only apply in specific contexts within the discipline. For example, self-directed (as in learning) is universally applicable to CS graduates. On the other hand, collaborativeness may not apply to consultants in a context where they work independently on specific components of a project.

Professional dispositions are learnable, although not always teachable. We can formatively evaluate them, but not always summatively. They must be observable and are discipline-specific. Not

Table 2: Required CS Student Outcomes [1]

Graduates of the program will have an ability to:	
<i>[Applicable to all computing programs, including CS programs]</i>	
1.	Analyze a complex computing problem and +apply principles of computing and other relevant disciplines to identify solutions.
2.	Design, implement, and evaluate a computing-based solution to meet a given set of computing requirements in the context of the program's discipline.
3.	Communicate effectively in a variety of professional contexts.
4.	Recognize professional responsibilities and make informed judgments in computing practice based on legal and ethical principles.
5.	Function effectively as a member or leader of a team engaged in activities appropriate to the program's discipline.
<i>[Applicable only to CS programs]</i>	
6.	Apply computer science theory and software development fundamentals to produce computing-based solutions.

all identified dispositions may apply to every task or context in a discipline. We may view a few dispositions as universal, often abstractly, whereas most may be context-dependent.

Helping CS students learn professional dispositions is essential; however, it should not be at the expense of either knowledge or skills but rather as a synergistic complement. Accounting for learning curve issues, we envisage that a course or curriculum will introduce professional dispositions gradually, increasing their importance as the course or curriculum progresses.

4 ABET CS PROGRAM CRITERIA

We now examine the role that the ABET program criteria for CS [1] might play in promoting competencies for CS graduates. We first look at student outcomes and the continuous improvement processes required by these accreditation criteria.

4.1 Student Outcomes

ABET's Student Outcomes "describe what students are expected to know and be able to do by the time of graduation. These relate to the knowledge, skills, and behaviors that students acquire as they progress through the program." [1]. Despite the explicit mention of both *skills* and *behaviors* (which are akin to dispositions), ABET visiting evaluation teams have placed primary emphasis on *knowledge*, with increasing consideration of skills more recently. However, as behaviors/dispositions are new to CS and all computing disciplines, ABET teams have generally not included them in their program evaluations. This practice is not surprising, given that curricular guidelines, especially CS2013 [2], emphasize knowledge topics with only a passing reference to skills and dispositions.

ABET requires each CS program to have documented and publicly stated student outcomes. These include five student outcomes from the general criteria applicable to all computing programs, along with a sixth outcome required only by CS program criteria, as shown in Table 2. Programs are free to define other student outcomes in addition to these six outcomes to characterize their uniqueness. All the outcomes, whether ABET-required or program-defined, must become part of the program's continuous improvement processes, as discussed in Section 4.2.

We examine the six student outcomes listed in Table 2 and discuss the traditional (current) perspective on each. We also propose a way forward to incorporate skills and dispositions in addition to knowledge to produce a competency-based perspective of computing accreditation without requiring changes to these criteria.

Outcome 1: From a knowledge perspective, students must learn basic computing principles related to problem-solving. The instructor will need to let students know which skills they should demonstrate in generating a solution and assess student skill understanding and level. To address dispositions, the instructor should try to have students work in pairs or small groups. From the eleven CC2020 disposition elements shown in section 3, the instructor could select all eleven or a subset of them. The instructor could develop appropriate assessment instruments for each student or group of students and score them using a simple rubric to assess skills and dispositions, thereby satisfying the ABET requirement.

Outcome 2: Students must design and implement a computer-based solution to a problem based on requirements. Once completed, students must evaluate the efficiency of the solution. Students must have substantial knowledge to undertake such steps, and they would have to demonstrate sufficient skills, as prescribed by their instructor, whether working individually or in small groups. This outcome also allows many opportunities for dispositional elements to manifest and be observed or reflected, such as (being) inventive and purpose-driven. The instructor can assess student knowledge, skills, and dispositions in various ways. For example, instructors can assess student knowledge by traditional methods; they can evaluate skills and dispositions by developing and applying appropriate rubrics to assess project artifacts and processes.

Outcome 3: This outcome focuses on skills and dispositions, where knowledge forms the basis of information exchange. Communication includes listening, reading, speaking, and writing. Related skills could include active listening, being friendly, having confidence, and encouraging interaction. Depending on the kind of communication, dispositional elements could include being passionate, responsible, and self-directed. Instructors should let students know the methods used for assessing student's knowledge, skills, and dispositions. Examples of assessments include oral presentations, project meeting notes, sprint reports, or portfolios of student work.

Outcome 4: Students must know some of the legal and ethical principles of computing. Depending on the assignment, skills and dispositions will vary. Instructors should inform students on how they will assess knowledge, skills, and dispositions. Students should follow their instructor's recommendations on the assignment, being aware that instructors will evaluate them beyond knowledge.

Outcome 5: This outcome is more a combination of skills and dispositions, with dispositions being the salient component. Knowledge plays a minimal role in this outcome. Prominent dispositions include adaptable, collaborative, passionate, proactive, professional, and self-directed. Instructors should inform students of their assessment methods and the performance expected of students.

Outcome 6: This student outcome applies only to CS students, where knowledge is prominent in theory and software development. The assessment of student knowledge is paramount, and instructors should inform students of the methods and scope of such evaluations. Computing skills are also significant, and students should know in advance the methods used by the instructor for assessing skills. Disposition elements could include inventive, meticulous, proactive, and self-directed. Again, instructors should inform students of the methods used for such assessment.

In Table 3, we make a first attempt to identify CC2020 dispositions most appropriate for each of the six ABET student outcomes. This exercise is mainly for illustration, as we expect these initial mappings to change as the CS community gains more experience with all three aspects of competencies. The value of creating such an initial mapping is to help programs and faculty become intentional about professional dispositions and design learning experiences that better prepare students for their future careers.

We also anticipate that programs may associate additional dispositions with each outcome to emphasize the unique aspects of their programs. For example, a program with a mandatory cooperative education component would see the utility of associating the *collaborative* disposition with outcomes 3 and 4.

4.2 Continuous Improvement

ABET [1] states: “The program must regularly use appropriate, documented processes for assessing and evaluating the extent to which the student outcomes are being attained. The results of these evaluations must be systematically utilized as input for the program’s continuous improvement actions.”

ABET defines *assessment* as one or more processes that identify, collect, and prepare data to evaluate the attainment of student outcomes. Effective assessment uses relevant direct, indirect, quantitative, and qualitative measures as appropriate for the measured outcome. It defines *evaluation* as one or more processes for interpreting the data and evidence accumulated through assessment processes. Evaluation determines the extent to which students attain

the outcomes. Evaluation typically leads to decisions and actions to improve programs.

For continuous improvement, each program needs to assess and evaluate its student outcomes periodically. Assessment should not focus on knowledge alone but also include skills and behaviors (dispositions). Programs specify some minimal threshold (e.g., 75%) for attainment of a student outcome. Unfortunately, ABET evaluation teams and most CS programs have focused on assessing knowledge, and occasionally skills, but have not emphasized dispositions.

We hope this work will lead ABET and its accredited programs to incorporate dispositions for continuous improvement and for meeting accreditation requirements. These changes will result in competent graduates prepared for successful CS careers.

5 CONCLUDING REMARKS

This work addressed some of the background surrounding computing education, knowledge- and competency-based learning, and ABET’s definition of student outcomes that includes skills and dispositions in addition to knowledge. The work also presented a three-dimensional model of competency and described how university instructors can implement all three competency components for ABET accreditation of CS programs.

The points related herein may be sufficient for ABET-accredited programs and ABET evaluation teams to consider CS programs from the standpoint of competency, i.e., the three-tuple (knowledge, skills, dispositions) rather than just from the perspective of knowledge. Although the paper focused on CS, most of the discussion is also applicable to other computing programs accredited by ABET.

In addition to Criterion 3, Student Outcomes, which focuses on knowledge, skills and behaviors (dispositions), the ABET accreditation criteria also include Criterion 5, Curriculum, which by definition deals only with knowledge topics. The question then arises whether it is appropriate to keep knowledge isolated from skills and dispositions, as the world of computing moves toward competency-based education. An answer will allow ABET to develop a true competency-based approach toward accreditation whereby its criteria and related documents support knowledge, skills, and dispositions for accreditation of undergraduate computing programs.

This paper represents a first step in convincing ABET-accredited CS programs, as well as ABET itself, to join the move toward a view of computing education that emphasizes competency—not just knowledge and skills, but also professional dispositions. As such, this move will get computing programs up to date with the larger societal movement towards competency-based education.

ACKNOWLEDGMENTS

This paper owes much to other members of the ITiCSE 2021 working group on competencies: David Bowers, Mats Daniels, Felienne Hermans, Natalie Kiesler, Bonnie MacKellar, Renée McCauley, Syed Waqar Nabi, and Michael Oudshoorn. We also thank Alison Clear, Tony Clear, Marisa Exter, Larry Jones, Allen Parrish, and Pearl Wang for many discussions on the need for and role of competencies. Raj and Sabin acknowledge support by the National Science Foundation under Awards 1922169 and 2110771, and Award 2110823 respectively.

Table 3: Mapping CC2020 Dispositions to Student Outcomes

Disposition	1	2	3	4	5	6
<i>Adaptable</i>	✓		✓		✓	✓
<i>Collaborative</i>	✓	✓			✓	✓
<i>Inventive</i>	✓	✓				✓
<i>Meticulous</i>		✓	✓		✓	✓
<i>Passionate</i>			✓		✓	
<i>Proactive</i>			✓		✓	✓
<i>Professional</i>	✓	✓	✓	✓	✓	
<i>Purpose-driven</i>	✓	✓			✓	✓
<i>Responsible</i>	✓	✓	✓	✓	✓	
<i>Responsive</i>			✓		✓	
<i>Self-directed</i>			✓		✓	✓

REFERENCES

- [1] ABET, Inc. 2021. Criteria for Accrediting Computing Programs. Effective for Review During the 2021-22 Accreditation Cycle. <https://www.abet.org/wp-content/uploads/2021/01/C001-21-22-CAC-Criteria.pdf>.
- [2] ACM/IEEE-CS Joint Task Force on Computing Curricula. 2013. *Computer Science Curricula 2013*. Technical Report. ACM Press and IEEE Computer Society Press. <https://doi.org/10.1145/2534860>
- [3] William F. Atchison, Samuel D. Conte, John W. Hamblen, Thomas E. Hull, Thomas A. Keenan, William B. Kehl, Edward J. McCluskey, Silvio O. Navarro, Werner C. Rheinboldt, Earl J. Schweppe, William Viavant, and David M. Young. 1968. Curriculum 68: Recommendations for Academic Programs in Computer Science: A Report of the ACM Curriculum Committee on Computer Science. *Commun. ACM* 11, 3 (March 1968), 151–197. <https://doi.org/10.1145/362929.362976>
- [4] Richard H. Austing, Bruce H. Barnes, Della T. Bonnette, Gerald L. Engel, and Gordon Stokes. 1979. Curriculum '78: Recommendations for the Undergraduate Program in Computer Science— a Report of the ACM Curriculum Committee on Computer Science. *Commun. ACM* 22, 3 (March 1979), 147–166. <https://doi.org/10.1145/359080.359083>
- [5] Mary Besterfield-Sacre, Larry J. Shuman, Harvey Wolfe, Cynthia J. Atman, Jack McGourty, Ronald L. Miller, Barbara M. Olds, and Gloria M. Rogers. 2000. Defining the outcomes: a framework for EC-2000. *IEEE Transactions on Education* 43, 2 (2000), 100–110. <https://doi.org/10.1109/13.848060>
- [6] David S. Bowers and Christopher Sharp. 2021. Institute of Coding: Accreditation Standard. <https://institute-of-coding.github.io/accreditation-standard/>.
- [7] Lillian Cassel, Alan Clements, Gordon Davies, Mark Guzdial, Renée McCauley, Andrew McGettrick, Bob Sloan, Larry Snyder, Paul Tymann, and Bruce W. Weide. 2008. *Computer Science Curriculum 2008: An Interim Revision of CS 2001*. Technical Report. ACM Press, New York, NY, USA.
- [8] Alison Clear, Tony Clear, John Impagliazzo, and Pearl Wang. 2020. From Knowledge-based to Competency-based Computing Education: Future Directions. In *2020 IEEE Frontiers in Education Conference (FIE)*. IEEE, New York, 1–7. <https://doi.org/10.1109/FIE44824.2020.9274288>
- [9] Alison Clear, Tony Clear, Abhijat Vichare, Thea Charles, Stephen Frezza, Mirela Gutica, Barry Lunt, Francesco Maiorana, Arnold Pears, Francois Pitt, Charles Riedesel, and Justyna Szykiewicz. 2020. Designing Computer Science Competency Statements: A Process and Curriculum Model for the 21st Century. In *Proceedings of the Working Group Reports on Innovation and Technology in Computer Science Education (Trondheim, Norway) (ITICSE-WGR '20)*. Association for Computing Machinery, New York, NY, USA, 211–246. <https://doi.org/10.1145/3437800.3439208>
- [10] Alison Clear, Allen Parrish, John Impagliazzo, Pearl Wang, Paolo Ciancarini, Ernesto Cuadros-Vargas, Stephen Frezza, Judith Gal-Ezer, Arnold Pears, Shingo Takada, Heikki Topi, Gerrit van der Veer, Abhijat Vichare, Les Waguespack, and Ming Zhang. 2020. *Computing Curricula 2020 (CC2020): Paradigms for Future Computing Curricula*. Technical Report. Association for Computing Machinery / IEEE Computer Society, New York, NY, USA. <http://www.cc2020.net/>.
- [11] Andrea Danyluk and Paul Leidig. 2021. *Computing Competencies for Undergraduate Data Science Curricula (DS2021)*. Technical Report. Association for Computing Machinery, New York, NY, USA. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/dstf_cdsc2021.pdf.
- [12] Gordon B. Davis, John T. Gorgone, Daniel Couger, David L. Feinstein, and Herbert E. Longenecker. 1996. *IS'97: model curriculum and guidelines for undergraduate degree programs in information systems*. Technical Report. ACM, AIS, AITP. <https://dl.acm.org/doi/10.1145/512249.512222>.
- [13] Gerald Engel, John Impagliazzo, and Patrick LaMalva. 2010. A Brief History of the Computing Sciences Accreditation Board (CSAB) Promoting Quality Education in the Computing Fields. *ACM Inroads* 1, 2 (jun 2010), 62–69. <https://doi.org/10.1145/1805724.1805740>
- [14] Stephen Frezza, Mats Daniels, Arnold Pears, Åsa Cajander, Viggo Kann, Amanpreet Kapoor, Roger McDermott, Anne-Kathrin Peters, Mihaela Sabin, and Charles Wallace. 2018. Modelling Competencies for Computing Education beyond 2020: A Research Based Approach to Defining Competencies in the Computing Disciplines. In *Proceedings Companion of the 23rd Annual ACM Conference on Innovation and Technology in Computer Science Education (Larnaca, Cyprus) (ITICSE 2018 Companion)*. Association for Computing Machinery, New York, NY, USA, 148–174. <https://doi.org/10.1145/3293881.3295782>
- [15] IEEE Computer Society. 2014. *Software Engineering Competency Model, Version 1.0*. Technical Report. IEEE, New York, NY, USA. <https://www.baltimoresun.com/opinion/op-ed/bs-ed-op-0328-minoritized-word-20190320-story.html>BN-13: 978-0-7695-5373-3.
- [16] John Impagliazzo, Susan Conry, Eric Durant, Joseph L.A. Hughes, Lorraine Herger, Lu Junlin, Herman Lam, Andrew McGettrick, Robert Reese, Liu Weidong, and Victor Nelson. 2016. *Computer Engineering Curricula 2016*. Technical Report. ACM Press and IEEE Computer Society Press. <https://www.acm.org/binaries/content/assets/education/ce2016-final-report.pdf>.
- [17] ISO. 2019. *Software and systems engineering — Certification of software and systems engineering professionals — Part 1: General requirements*. Standard. International Organization for Standardization, Geneva, Switzerland.
- [18] Joint Task Force for Computing Curricula. 2005. *Computing Curricula 2005: The Overview Report*. Technical Report. ACM Press and IEEE Computer Society Press. <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2005-march06final.pdf>.
- [19] Joint Task Force on Computing Curricula. 2001. *Computing Curricula 2001*. *J. Educ. Resour. Comput.* 1, 3es (Sept. 2001), 1–es. <https://doi.org/10.1145/384274.384275>
- [20] Eunjo Jung and Dent M. Rhodes. 2008. Revisiting Assessment in Teacher Education: Broadening the Focus. *Assessment and Evaluation in Higher Education* 33, 6 (December 2008), 647–660. <https://doi.org/10.1080/02602930701773059>
- [21] Paul Leidig and Hannu Salmela. 2021. *A Competency Model for Undergraduate Programs in Information Systems (IS2020)*. Technical Report. Association for Computing Machinery, New York, NY, USA. <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/is2020.pdf>.
- [22] Barry M. Lunt, Joseph J. Ekstrom, Sandra Gorka, Gregory Hislop, Reza Kamali, Eydie Lawson, Richard LeBlanc, Jacob Miller, and Han Reichgelt. 2008. *Curriculum Guidelines for Undergraduate Degree Programs in Information Technology (IT2008)*. Technical Report. Association for Computing Machinery, New York, NY, USA. <https://www.acm.org/binaries/content/assets/education/curricula-recommendations/it2008-curriculum.pdf>.
- [23] George E. Miller. 1990. The assessment of clinical skills/competence/performance. *Acad. Medicine* 69 (1990), S63–S67. Issue 9 Suppl.
- [24] Chris Procter and Mark Kozak-Holland. 2019. The Giza pyramid: learning from this megaproject. *Journal of Management History* 25, 3 (2019), 364–383. <https://doi.org/10.1108/JMH-11-2018-0061>
- [25] Mihaela Sabin, Hala Alrumaih, John Impagliazzo, Barry Lunt, Ming Zhang, Brenda Byers, William Newhouse, Bill Paterson, Svetlana Peltsverger, Cara Tang, Gerrit van der Veer, and Barbara Viola. 2017. *Information Technology Curricula 2017 (IT2017)*. Technical Report. Association for Computing Machinery / IEEE Computer Society, New York, NY, USA. <https://dl.acm.org/doi/pdf/10.1145/3173161>.
- [26] The SFIA Foundation. 2021. SFIA 8: All skills A–Z. <https://sfia-online.org/en/sfia-8/all-skills-a-z>.
- [27] Heikki Topi, Helena Karsten, Sue A. Brown, João Alvaro Carvalho, Brian Donnellan, Jun Shen, Bernard C. Y. Tan, and Mark F. Thoun. 2017. *MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems*. Technical Report. Association for Computing Machinery, New York, NY, USA.
- [28] Allen B. Tucker, Robert M. Aiken, Keith Barker, Kim B. Bruce, J. Thomas Cain, Susan E. Conry, Gerald L. Engel, Richard G. Epstein, Doris K. Lidtke, Michael C. Mulder, Jean B. Rogers, Eugene H. Spafford, A. Joe Turner, and Bruce H. Barnes. 1991. *Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force*. Technical Report. ACM Press and IEEE Computer Society Press, New York, NY, USA.
- [29] Jerry Yeargan, Dayne Aldridge, Ira Jacobson, and John White. 1995. ABET Engineering Criteria 2000. In *Proceedings of the Frontiers in Education 1995, 25th Annual Conference. Engineering Education for the 21st Century*, Vol. 2. IEEE, Atlanta, 1.