Toward Visualizing Computing Curricula: The Challenge of Competency

Leslie J. Waguespack
lwaguespack@bentley.edu
Computer Information Systems
Bentley University
Waitham, MA 02452

Jeffry S. Babb
jbabb@wtamu.edu
Computer Information and Decision Management
West Texas A&M University
Canyon, TX 79016

Abstract

Amidst academic societies and agencies that accredit computing education there is a growing enthusiasm to reexamine the efficacy of the traditional model of curricular description that focused on areas of knowledge. The knowledge model informed the architecture and design of programs of teaching and learning in post-secondary, degree-granting institutions. The incipient enthusiasm for change draws on a vocational heritage focusing on job performance together with outcomes-based and task-centered learning and assessment, competency. Competency's emergence is fueled by a waning confidence in the cost-effectiveness of college-based education, an industry perception of a persistent short-fall of technology-savvy hiring prospects, and the efforts of governments worldwide encouraging the alignment of public education with economic and workforce policy. The competency model represents the consequence of learning as a blend of knowledge, skills, and disposition – "*knowing* what," "*knowing* how," and "*knowing* why." The *knowing* is task-focused both as the learning *in doing* and the assessment as demonstration *in doing*. Coincidentally, ACM and IEEE have undertaken to reprise CC2005 with the goal of an online interactive curriculum modeling tool for comparing and exploring curricular guidelines and academic programs. We explore the competency-based curricular approach situated in a) the history of computing curricula standardization, b) its heritage in education originating with clinical and professional disciplines, and c) its implications on the CC2020 project's aspirations of designing a "tool" to facilitate current and future competency-based computing curricula development.

Keywords: Computing Competency, Curricular Guidelines, Knowledge (Areas, Units, Learning Outcomes), Computing workforce preparation

1. INTRODUCTION

This paper takes a brief look back at computing curricula and their constitution and a look forward in light of the growing and earnest interest in competency as a design medium of education. ACM and IEEE partnering with sibling computing associations are spearheading an ongoing inventory and forecast of computing curricula development, *Computing Curricula 2020*. (See www.cc2020.net.) The CC2020 effort is a response to an accelerating advance of technology and evolving perspectives on the valuation of educational outcomes of computing programs. Particular to that effort, this paper sketches a framework for curriculum description

that incorporates and normalizes the structure and intra-connectivity of computing theory and practice. There has never been a rubric for a top-down description of curriculum spanning computing education. The framework is a necessary foundation for a key CC2020 project goal, the design of an online visualization tool capable of both representing and comparing computing guidelines and programs to inform and advance computing education in the 2020's and beyond. As the authors are presently engaged in this ongoing design, our goal here is to report and explore the challenges intrinsic to this undertaking.

2. BRIEF COMPUTING CURRICULA RETROSPECTIVE

Since 1968, professional computing communities have invested in developing guidelines that chart a path for computing education in degree-granting institutions, and to some extent, the entire community of practicing professionals (Longenecker, Feinstein, Babb, 2013). By and large, the various curricular guidelines have focused on the delivery of subdiscipline-specific, fact-based information aligned with a scientific and technically-rational model of instruction. Sub-disciplines of computing have evolved generally independent of one another creating de facto silos of perspective on computing albeit sharing significant overlaps of theory, technology, methodology, and professional practice.

The traditional sub-disciplines of computing are codified in the collection of baccalaureate level, curriculum guidelines published under the sponsorship of ACM and IEEE with various partners over the past couple decades: Computer Engineering (2004, 2016), Computer Science (2001, 2008, 2013), Information Systems (1997, 2002, 2006, 2010), Information Technology (2008, 2017), Software Engineering (2004, 2014), and Cybersecurity (2017). (All guidelines are available at www.acm.org/education/curricula-recommendations.). As of this writing projects are underway for new and/or updated sub-discipline guidelines for data analytics, artificial intelligence, and information systems.

A baccalaureate degree in computing has been the traditional flagship of credentialed entry to the computing profession. In the last decade or so, instructional approaches focusing on a much narrower conception of professional preparation, "computing" bootcamps, have arisen as an alternative and to some extent, have become challengers to the traditional baccalaureate programs (Waguespack, Babb, Yates, 2018). Computing education is also widely accessible over the internet in both tuition and tuition-free models without the traditional "brick and mortar" institution (e.g. Coursera, 2018). The pace of computing technology's expanding influence on society and the growth of related scientific and technological information continues to accelerate; as does the growing need for workers in the computing domain. The time is nigh to take stock of the breadth and width of computing education both for orientation and navigation with a focus beyond 2020, hence the impetus for the CC2020 project.

3. CC2005 CURRICULUM VISUALIZATION

The Association for Computing Machinery, ACM, and the Institute for Electrical and Electronics Engineering, IEEE, have worked consistently to normalize the structure and evolution of computing education. The series of published curricular guidelines for particular computing disciplines, as well the mapping of the overall landscape as in Computing Curriculum 2005 (CC2005) are the current, standard references for computing education (Shackelford, McGettrick, Sloan, Topi, Davies, Kamali, Cross, Impagliazzo, LeBlanc, & Lunt, 2005).

In that CC2005 report (the most recent and comprehensive cross-discipline analysis), the task force created graphic characterizations of "what students in each of the disciplines typically do after graduation." Each discipline was portrayed on a field of credible, professional capability as a "footprint" of proficiency gained by completing the respective academic program. (See Figure 1.)

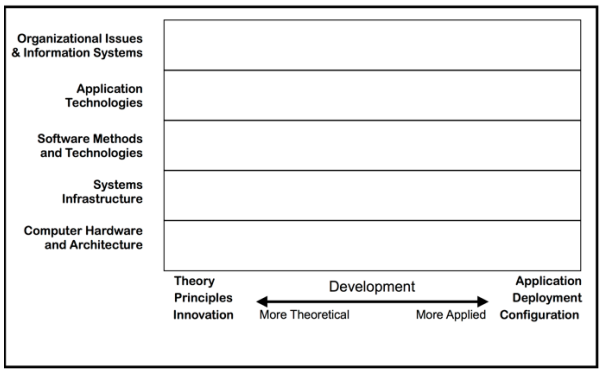

Figure 1 - CC2005 Field of Computing Competency

In that 2005 overview report, the five computing disciplines (CS, CE, IT, IS, and SE) were represented in individual "footprint" diagrams. Each diagram was a plausible depiction negotiated by a committee of curricular "experts." Figure 2 depicts IS education in 2005 as an example (Shackelford et al., 2005).

The "footprint" of baccalaureate proficiency lies on a field delineating computing activity ranging on the Y-axis from hardware issues on the bottom to organizational policy and information management at the top. The X-axis depicts the far right as purely *applied* involvement in computing activities while to the left is purely *theoretical* engagement with computing topics. In addition to the "footprint" representations, the CC2005 report also provided a tabular representation of the plausible relative emphasis of knowledge areas as documented among the respective curricular guidelines. (See Appendix A.)

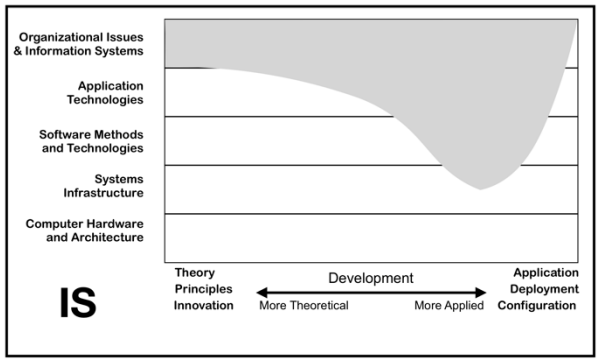

Figure 2 - Competency Target of IS (2005)

When CC2005 was published, the intention was to revisit the computing landscape every five or so years to both chart the "status-quo" and to offer a reflection on the emerging demands on computing education owing to advancing technology and society's expanding dependence on computing. However, that intention has waited nearly fifteen years for the mantel to be taken up again in CC2020.

Now underway, the CC2020 project aspires to reprise the effort to visualize the computing landscape. But in this attempt, rather than depictions of "expert" opinion as offered in CC2005, the goal is to derive the visualizations of the computing disciplines based upon actual descriptions in the respective published curriculum guidelines. Those CC2005 depictions, although speculative, have proven invaluable as approximate contrasts of scope and focus among the various curricula. But, the continuing addition of new disciplinary variations fuels the need to do more than speculate. "More" entails a theory-grounded articulation of the spectrum of computing education, an instrument to compare and contrast degree programs, guidelines, and disciplines; and an empirical, "algorithmically" rational visual representation of curricular specifications. While automating a visualization of the computing curricula as conceived in 2005 presents taxonomical and epistemological challenges in itself, the most recent curricular guidelines (MSIS2016 and IT2017) have undertaken a new descriptive strategy aligned more closely to training, certification, and job description that is more familiar to the domain of human resources. This strategy, *competency-based*, represents an educational goal of instilling students with a combination of knowledge, skills, and disposition amenable to validation using task-specific proficiency assessment.

In the spirit of the "footprint" diagrams of baccalaureate capabilities that grace the CC2005 report, CC2020 envisions an instrument able to:

a) visually represent curricular guidelines or programs for exhibition and analysis,
b) permit navigation and inspection of such guidelines or programs at varying levels of scale / detail,
c) detect commonalities and differences between or among curriculum descriptions and display same visually,
d) accommodate the drafting of curriculum guidelines (i.e. any proposed or published subdiscipline of computing) or any existing or proposed program of study in computing as an aid to prototyping and development,
e) support the manipulation and versioning of curriculum descriptions,
f) (implicit in a thru e) facilitate a repository for the cumulation and evolution of a taxonomy of computing to support cross comparison and future curricular development.

The CC2020 project team refers to this visualization capability as "the tool." The goal of this paper is to explore the challenge that CC2020 has accepted and conceptualize a framework for incorporating *competency* in the visualization effort.

4. KNOWLEDGE VS COMPETENCY FOCUS

All the published baccalaureate computing curricula in 2005 were conceptually grounded in a listing of knowledge areas, knowledge units, and learning outcomes (KA-KU-LO). (See

Appendix A, KA examples found in the CC2005 report.) Thus, the visual representations plausibly exhibit a degree of comparability across subdisciplines. The arrival of IT2017 (IT2017, 2017), heralds a shift in specification strategy, less conformant to the basis of KA-KU-LO – specifically due to its emphasis on *competency*. (See Appendix B comparing the published computing curricula that are the current focus of CC2020.)

The IT2017 project is the first of the ACM, IEEE baccalaureate curriculum projects to embrace competency as the primary characteristic of curriculum definition. MSIS2016 introduced competencies earlier but, also included prototypical course descriptions reminiscent of KA-KU-LO. In some ways the competency approach is more a facet of labeling than a dramatic shift in curricular description. Learning outcomes have been a prominent feature for some time (USDoE, 2018).

The learning outcome concept is key to the shift in education from a paradigm concerned with providing instruction to a paradigm of producing learning (Barr, 1995).

Table 1 – Six facets of learning transfer

Explain	Learners make connections, draw inferences, express them in their own words with support or justification, use apt analogies;
Interpret	Learners make sense of, provide a revealing historical or personal dimension to ideas, data, and events; interpretation is personal and accessible through images, anecdotes, analogies, and stories; turn data into information; provide a compelling and coherent
Apply	Learners use what they have learned in varied and unique situations; go beyond the context in which they learned to new units, courses, and situations, beyond the classroom.
Demonstrate Perspective	Learners see the big picture, are aware of, and consider various points of view; take a critical and disinterested stance; recognize and avoid bias in how positions are stated.
Show Empathy	Learners perceive sensitively; can "walk in another's shoes;" find potential value in what others might find odd, alien, or implausible.
Have Self-Knowledge	Learners show metacognitive awareness on motivation, confidence, responsibility, and integrity; reflect on the meaning of new learning and experiences; recognize the prejudices, projections, and habits of mind that both shape and impede their own understanding; are aware of what they do not understand in a specific context.

But, the prominence of competency further emphasizes that *fact knowledge* does not sum all the *knowing* sufficient to equip a practicing professional. Competency is a familiar term in the domains of education usually classified as *training* and *job* performance assessment. Competency is identified with job recruitment, placement, and performance assessment that underpins the core of its affiliations in human resources and workforce management in the commercial and governmental arenas (Bloom, Krathwohl, 1956; Dave, 1970; Harrow, 1972; Krathwohl, Bloom, Bertram, 1973; Wiggins, McTighe, Ebrary, 2005).

Generally, the term [competence] refers to the performance standards associated with a profession or membership to a licensing organization. Assessing some level of performance in the workplace is frequently used as a competence measure, which means measuring aspects of the job at which a person is competent (IT2017, p. 28).

The meaning of *competency* may vary widely among particular professions or registries; IT2017 adopts its own "working" definition: *Competency = Knowledge + Skills + Dispositions*. In IT2017's "working" definition, *knowledge* is understood as mastery and the transfer of content knowledge. *Skills* are understood as capabilities and strategies for higher-order thinking and interactions with others and the world around. *Dispositions* are understood as personal qualities (socio-emotional skills, behaviors, attitudes) associated with success in higher education and career (IT2017, p. 28).

Competency's epistemological roots are found in the formal training of established labor disciplines (e.g. nursing) where the procedures and behavior employed require consistent, predictable, and disciplined application or treatments (Heath, 1998; Johns, 1995). They also align with the rubrics of socially acceptable conduct that circumscribe a specific profession with consequent statutory implications (e.g. licensure and legal liability). IT2017's citation of "six facets of learning transfer" intimate a predilection for a

vocational trajectory of learning (IT2017, p. 28; Wiggins, 2005). (See Table 1.)

Curricular descriptions based upon *competency* aspire to "close the loop" on experiential learning, exploit *learning by doing*, and systematize performance-based assessment. At the same time, competencies of themselves are indifferent to specific pedagogy. In that sense, competency focuses on the accomplished learner's professional capacity rather than the agency of pedagogy.

5. MODELING CURRICULUM AS COMPETENCIES

In the simplest terms, curriculum developers adopt competencies to model the "product" of an educational process. Given a set of known facts defining a domain of interest, the function of competency-based education is envisioned as imbuing an understanding of practice in a subset of domain knowledge that suffices a desired range of practice; thus, supporting satisfactory performance in a particular profession, discipline, or situated task (Anderson, 2001). Curriculum is a model intended to define and instill competency.

In the following set theoretic representation, *Competency-Disposition-Knowledge-Skills-Task* (CDKST), we adopt three grounding propositions to conceptualize curriculum: 1) learning is acquiring knowledge elements arranged taxonomically that enable satisfactorily performing relevant tasks; 2) the concept of "skill" is a degree of mastery of a knowledge element modulated by disposition to achieve a valued outcome, and 3) disposition denotes the values and motivation that guide applying knowledge while designating the quality of knowing commensurate with a standard of professional performance.

Note: The original idea for this set theoretic model emerged in the collaborative work of the CC2020 "tool" Task Group of which the first author is a member. The collaborative process and contributions of CC2020 team as well as the "EDSIG Tool Auxiliary" are greatly appreciated.

CDKST Curriculum Framework

Competency-Disposition-Knowledge-Skills-Task

Knowledge elements, **K**, are factual concepts supported by science and/or professional practice that underpin a vocabulary of objects, behaviors, and relationships as the domain of interest in a discourse (be it curriculum, task, job, or profession). **S**, the skill attribute, denotes the *quality of knowing* (e.g. mastery, expertise, adeptness, or proficiency) that an accomplished learner must possess to satisfactorily apply a knowledge element in a circumstance of performance. In this sense it is the capacity to demonstrate a degree of *cognitive command* of that knowledge. In this conceptualization cognitive command is represented by Bloom's (revised) taxonomy of learning objectives: remember, understand, apply, analyze, evaluate, and create (See Appendix D, Anderson, 2001). Disposition, **D**, represents a commitment, motivation, toward an aspect of professional practice that reflects the attitude deemed critical to satisfaction in a professional circumstance or context. Task, **T**, is a situated instance of engaging knowledge with a degree of mastery. **C**, competency is a demonstrated sufficiency in a task with an appropriate disposition.

$$\mathbf{T} = \text{task}$$
$$\mathbf{T} \dashrightarrow \{(K_i,S_j)\} \text{ knowledge applied}$$

[A task is a set of one or more knowledge/skill pairs engaged in a purposeful act.]

Task, **T**, is *knowledge applied* in a "live" context to accomplish a designated purpose. **T** represents a *specification* of capability that curriculum is obligated to inculcate in the accomplished learner.

A task is the application of specific knowledge to a situation at hand. Note that tasks may be of varying complexity in terms of the range of knowledge elements engaged. Individual knowledge elements may participate in a variety of tasks. A task may be a collection of constituent tasks within which each knowledge element is applied with a distinct skill. As a collective, the task's satisfactory accomplishment exhibits a sufficiency of knowing and doing.

$$\mathbf{C} = \text{competency}$$
$$\mathbf{C} \dashrightarrow \{(\Sigma(K_i,S_j) \mid (K_i,S_j) \in T), D_k\}$$

[Competency is a task(s) satisfied demonstrating a disposition to professionalism.]

Competency, **C**, is the capacity to accomplish a task by applying knowledge and skills within a particular disposition. This is the goal sought by a competency-based perspective on curricular design. This forms a focus for assessment as each competency represents both a requirement and the instrument of certification to assure the learner's successful performance – success

denoted by the satisfactory outcome of applying the knowledge modulated by the disposition. It is reasonable to expect that a system of competency specifications would form a telescopic or hierarchical arrangement of modularized task complexity and thus, would lead to an incremental or progressive process of learning and experience accumulation that would subsequently justify advancement to more elaborate, intricate, or difficult tasks or higher degrees of professionalism.

E = education
E --> $\{C_i\}$
B = baccalaureate degree
$\mathbf{B}_e$ --> $\{\Sigma(C_i) \mid C_i \in E\}$

[A baccalaureate is the cumulation of competencies comprising a course of learning.]

E, is a composition of competencies relevant to (or defining) a professional or academic course of study, a curriculum. A baccalaureate degree, **B**, is granted by an authorized institution. In fact, the list of competencies may be the vary testimony to the focus of an intended career direction shaping an academic program's intension. This would be the construct for comparing educational programs, assessing guideline or accreditation compliance, or prototyping distinct perspectives on the larger domain of knowledge such as across subdomains of *computing*!

J = job description
J --> $\{C_i\}$
JP = job permit
$\mathbf{JP}_j$ --> $\{\Sigma(C_i) \mid C_i \in J\}$

[A job permit is the cumulation of competencies comprising a job description.]

In its own fashion, a particular job description is in effect a "mini-curriculum" as it prescribes performance requirements that usually distinguish the desired applicant or employee attributes. The particulars of the organization, the industry, or the marketplace would shape both the collection of knowledge elements, skills, and the disposition of their application, thus, aligning with a particular vocation.

P = profession
P --> $\{J_i\}$
L = professional license
$\mathbf{L}_p$ --> $\{\Sigma(J_i) \mid J_i \in P\}$

[A professional license is the cumulation of competencies denoting the profession as a set of jobs that distinguish it.]

In this last aggregation, professional societies and governmental agencies specify collections of competencies that qualify a legal standing as a licensed professional (e.g. professional engineer, medical doctor, physician's assistant, nurse, a member of the bar, barber, cosmetologist, etc.).

The CDKST model does not attempt to shape or bound the dimensions of pedagogy as that requires integration with the cultural context within which it must be applied. However, pedagogy must align with the designated disposition modulating the *professionalism* the student must demonstrate as competency in context.

6. VISUALIZING THE CDKST FRAMEWORK

Degree program, job description, and certification requirements, all represent conceptually identical competency-based structures. They would differ primarily in the "footprint" of **K** and **S** in the universe that delineates their individually relevant domain of discourse. Figure 3 depicts the various CDKST modeling elements comprising three instances of competency collection. We can readily assume that the availability of a directory of knowledge elements provides a ready resource for sharing relevant understanding across subdisciplines. Somewhat "bottom up" this might be the foundation for curriculum designers to frame a curriculum and then, flesh it out with selected pedagogy as an educational program. Similarly, somewhat "top down" employers and accrediting agencies might use a directory of competencies to assess the similarity or difference among job descriptions or professions in evaluating workforce policies or human resource arrangements.

It is likely that students will navigate their choice of programs by first examining their options "top down" and refining their choice of specific educational program at least in part "bottom up."

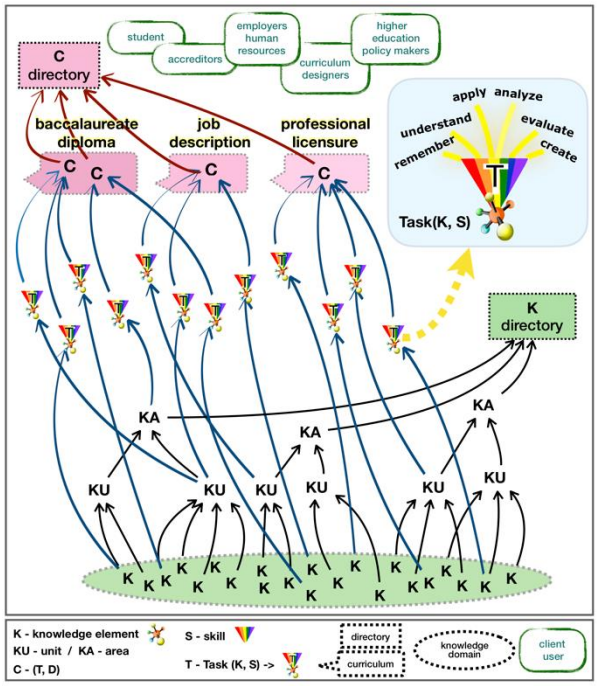

Figure 3 – CDKST Curriculum Framework

7. CHALLENGING CDKST DESIGN IMPLICATIONS

Indeed, by addressing some design implications in this section, we admit that our conceptualization is developmental rather than a complete design for the curriculum "tool." As the IT2017 Curriculum Guideline is the only published artifact currently available that is predominantly competency-focused, this section leans on that IT2017 document to discuss some challenging implications entailed by the set theoretic framework conceptualized above, CDKST.

Taxonomy – The first challenge for the CDKST framework is our first proposition that there must be a foundational taxonomy of knowledge elements of computing as in Figure 3 providing a basis to demonstrate any particular of competency.

If knowledge elements or competencies are to be comparable across curriculum descriptions, there must be a consistently shared curricular vocabulary. While both competency statements and knowledge elements are susceptible to revision in terminology and meaning, it seems clear that knowledge elements must be more stable, *less* likely to be deleted and replaced – more likely to be added to or fall into disuse. Competencies more so reflect a current policy or practitioner behavior often specific to an application context. These would seem far more dynamic than knowledge taxonomy. Epistemologically, competency is derived from knowledge, skills, and disposition rather than, knowledge, skills, and disposition derived from competency – these concepts are not commutatively derivative.

Paraphrasing the wisdom of Per Brinch Hansen – the [COBOL, Fortran, Algol, …] specification doesn't define the programming language, the compiler does (Hansen, 1973)! And so, competency does not inform the knowledge-skills-disposition but rather, knowledge-skills-disposition informs the competency.

Given infinite resources, the simplest plan for developing a *universal* taxonomy of computing knowledge might be to develop the taxonomy and then reconstruct all the existing curricular guidelines of computing. Even with infinite resources, the scope of such an effort is somewhat mind boggling. However, this taxonomy aspect of the conceptualization is basically unavoidable. The only apparent recourse is to protract the process of taxonomy development as an undertaking of "*continuous taxonomy improvement*."

The most likely "work around" for this task is extensive automated text analysis as a means of reducing the necessary manual effort by statistically detecting recurring language patterns to nominate prospective knowledge elements. The mining of text specific to *performance statements* in IT2017 and to KA-KU-LO statements in the existing guidelines along with course descriptions suggest the best opportunity for cataloging knowledge elements. (See Appendix A as an example of computing knowledge categorization and Appendix G that proposes performance statements applying IT skills.)

Correspondence of Knowledge Elements with Competency – Reviewing IT2017's current representation of the relationship between competency and performance statements, a rubric is needed to normalize the form and semantics for specifying competency that clearly expose the necessary constituent knowledge elements and dispositions.

IT2017 suggests action verbs to characterize task performance statements. (See Appendix C.) Most of its competencies are more specific in their reference to "outcomes" than to particular antecedent knowledge or skills that are necessary to satisfactory performance. Again, the most likely "work around" for this situation may be

automated text analysis to identify prospective correspondences, "outcome *relies upon* knowledge element(s) applied with this skill(s)," to reduce the manual workload of transliterating competencies. (See Appendix F.) At this juncture of our design analysis the cognitive process dimension verbs of Bloom's (revised) in Appendix D appear most appropriate for characterizing the degree of knowledge mastery for satisfactorily applying required knowledge – alternatives may arise!

Dimensional characteristics – Although knowledge elements may have designated titles, their applications in distinct task instances will reflect different ways of "understanding" the element. These differences are critical in specifying, evaluating and, particularly in the "tool," visualizing competency.

A benefit of adopting the revised Bloom's approach to cognitive processes to differentiate the skill required in applying knowledge is that the levels are intrinsically cumulative – that is the higher levels require skill support at all the lower levels (Anderson, 2001). This characteristic is analogous to magnitude enabling numerous visualization possibilities.

In Figure 2 the CC2005 "footprint" visualization ascribes the y-axis to the "continuum" of hardware through organizational policy. This particular taxonomical characterization is congruous in concept to the *semiotic ladder* (Stamper, 1991). With a formidable grounding in theory, the semiotic ladder likewise distributes and delineates the communicating media of meaning as agency extending from the *material* at the base through the *conceptual* at the top. (See Table 2.) The semiotic ladder characterizes a progression where steps offer homologous arrays of structural and behavioral metaphors with which to express successive degrees of abstraction from the material to the organizational (Stamper, 1973; Liu, 2000).

Table 2 - The Semiotic Framework

Semiotic Ladder	Semiotic Layer Description
Social World	Beliefs, expectations, functions, commitments, contracts, law, culture
Pragmatics	Intensions, communications, conversations, negotiations
Semantics	Meanings, propositions, validity, truth, signification, denotations
Syntactics	Formal structure, language, logic, data, records, deduction, software, files
Empirics	Pattern, variety, noise, entropy, channel capacity, redundancy, efficiency, codes
Physical	Signals, traces, physical distinctions, hardware, component density, speed, economics

The x-axis in Figure 2 traces the application of computing knowledge from the theoretical on the left to the applied on the right. In our conception of that x-axis we propose to use the *quality of knowing* the knowledge element to intimate the degree of conceptual insight, capacity for judgement, appropriate for applying that knowledge satisfactorily. (See Figure 4.) The x-axis positioning from left to right on a continuum would range from basic recollection and understanding over to the conceptual insight needed for in depth evaluation and creativity following Bloom's revised cognitive dimensions of learning in Appendix D (Anderson, 2001).

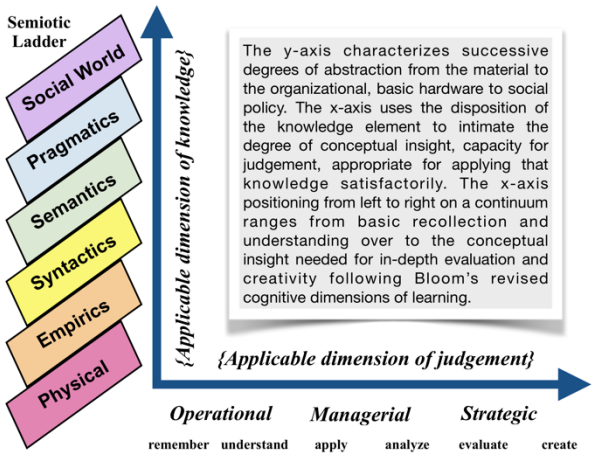

Figure 4 – Prospective CC2020 "Footprint"

Application Domain Competency – Naturally our attention is drawn primarily to knowledge elements in the domain of "computing." However, most computing curricula are embedded, intertwined, with the application domain to which they are applied. Curriculum in information systems is a particularly apt example embedded in commerce (Topi, 2017a). Also, virtually every aspect of computing relies to some degree on mathematics, verbal and written language, and organizational awareness – all of which are deserving of explicit education and thus, their own knowledge elements.

It may be possible to compartmentalize the knowledge of computing that is appropriate for information systems development in general. However, specific domain knowledge of an industry, regulatory, or cultural locale will probably require assessment of disposition adjustments to account for various degrees of indigenous sensitivity and relevance. This may lead to the merging or integrating curricular domains of computing with application specific competency. However, this is not a conceptual

hurdle since the CDKST model (although targeted in this discussion to computing) appears to be commensurately applicable in any knowledge domain.

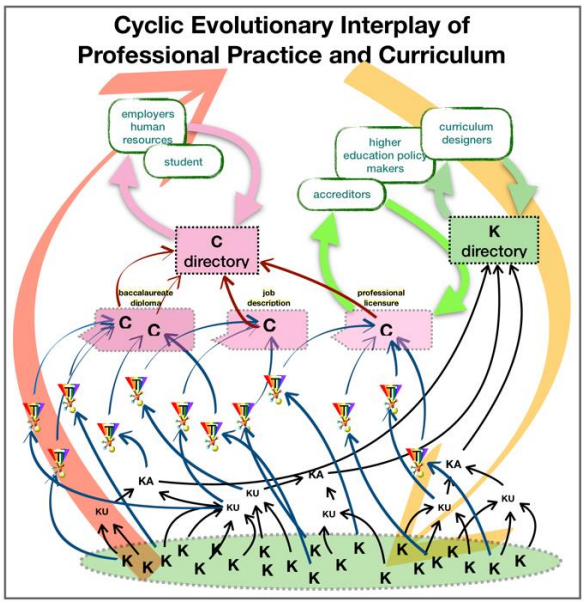

Figure 5 – Cyclic Evolution of Curricula

Reckoning with Competency Instability – Although the occurrence of computing curriculum retrospectives would seem to be infrequent (e.g. CC2005 – CC2020), the breadth and depth of computing's burgeoning influence on society far outpaces our current capacity for "self-reflection." To steward the investment in a competency-based taxonomy of computing knowledge proposed in this discourse, an ongoing curation program must be mounted not only to support the beneficial use of existing competency-based computing knowledge but also, the renewal and addition of new science, practice, and applications going forward. Although periodic publications chronicle the "contemporary state" of curricular evolution, that evolution is non-stop in breadth and depth. If the utility of the "tool" even weakly approximates the goals envisioned for examining, comparing, and developing curricular specifications, the "tool's" utility in academia may likely be eclipsed by industry interests in professional development, human resources, and by governmental policy makers involved in workforce analysis and development. In Figure 5, CDKST intimates four or more cycles of interactive co-evolution coupling computing practice with curriculum in response to the emerging science, technology, and applications of computing. While cycles provoked by science may span generations, technology applied often sparks rapid cycles best served by frequently reviewing and renewing various computing curricula.

8. Discussion

The CC2020 visualization tool must rely upon an interoperability and interrelationships encompassing competency, knowledge, skills, and disposition as spatially oriented concepts. In this discourse we presented a conceptualized design that exposes the epistemological and computational challenges that the "tool" must surmount to achieve the goals set out for it in CC2020.

The initiative to advance CC2005's conception of the visual representation of curriculum from the state of an interpretive sketch to a computationally accurate representation of curriculum specification is significant in multiple dimensions:

a) The goal of an empirically accurate representation of curricular specifications advances a normalized definition of computing and its components to correlate the perspectives of all the sub-disciplines of computing.
b) The ability to analyze and prototype curricular specifications through visual manipulation encourages a more active participation in curriculum development involving industry, government, and the public.
c) The "tool" offers the prospect of significantly increasing transparency of computing curricula to the benefit of students, academic programs, and employers. The "tool" can advertise the metaphorical "list of ingredients" in a more understandable and digestible form for curriculum consumers.
d) Technological, theoretical, and professional practice developments that naturally emerge in subdisciplines can be identified and explored earlier for their co-relevancy and implications across subdisciplines. This should facilitate opportunities for enhanced cooperation among interested parties and enable tangible economies of effort for researchers, funding agencies, and academic institutions.
e) An ongoing "continuous taxonomy improvement" process at the intersection of subdiscipline planning, development, and maintenance provides an opportunity for synchronizing and streamlining the collaborative efforts of curriculum developers, professional societies, and program assessment / accreditation agencies.

f) The prospect of compatible representations of job and curricular competencies can offer a great incentive for industry, government, and academia collaboration.
g) The potential exists for better understanding and exploiting various modes of learning and the relationships among computing education programs: primary, secondary, post-secondary, baccalaureate, graduate, certification, continuing professional development, etc. – perhaps, even a better understanding of the computing discipline as a whole.

We admit that the vision of a *universal* knowledge taxonomy of computing is unreasonable within the time horizon of the CC2020 project. However, a proof of concept and phased launch are doable. In the end, a crowd-sourced, "continuous taxonomy improvement" effort would be an invaluable legacy of the CC2020 project. We can hope that the nascent foundation provided by the "tool" will attract academics, professional societies, and workforce specialists who will contribute their own ideas and economic support as investment in the future of computing education for generations to come.

9. ACKNOWLEDGEMENTS

The ideas presented in this paper derive from a broad spectrum of experiences over the past forty or so years of the authors' computing careers: proposing, implementing, reviewing, and remodeling computing curricula. But, the impetus leading to the convergence of the thoughts presented herein is largely attributable to the collaborative energies and shared commitment of the CC2020 "tool" team. Although the authors accept full liability for the presentation herein, we wish to recognize and express our gratitude for the invaluable sharing of experience and imaginative ideas that emerged among the "tool" team members: Shingo Takada, Ernesto Cuadros-Vargas, Gerritt van der Veer, Steven Gordon, Linda Marshall, and Tania McVeety. We also recognize our EDSIG colleagues Musa Jafar and Jason Sharp who have contributed to the CC2020 "tool" project as members of the "EDSIG Tool Auxiliary." There is much work ahead of the "tool" team!

10. REFERENCES

Anderson, L. W., and D.R. Krathwohl (2001). *A taxonomy for learning, teaching and assessing: A revision of Bloom's Taxonomy of educational objectives*. Longman, New York, USA.

Barr, R.B. and Tagg, J. (1995). "From Teaching to Learning: A New Paradigm for Undergraduate Education." *Change*, 27(5), 12-25.

Bloom, B.S. and Krathwohl, D. R. (1956). *Taxonomy of Educational Objectives: The Classification of Educational Goals, by a committee of college and university examiners. Handbook I: Cognitive Domain*. Longmans, Green, New York, USA.

Coursera, (2018). https://about.coursera.org, (current July 4, 2018).

Dave, R.H. (1970). *Psychomotor levels in Developing and Writing Behavioral Objectives*, pp.20-21. R.J. Armstrong, ed. Tucson, Arizona: Educational Innovators Press.

Hansen, P.B. (1973). *Operating Systems* Principles, Prentice-Hall, Upper Saddle River, NJ, USA.

Harrow, A. (1972). *A Taxonomy of Psychomotor Domain: A Guide for Developing Behavioral Objectives*. David McKay Co., Inc., New York, USA.

Heath, H. (1998). "Reflection and patterns of knowing in nursing," *Journal of Advanced Nursing*, Vol. 27, pp. 1054-1059.

IT2017JointTaskGroup (2017). *Information Technology Curricula 2017*. ACM and IEEE-CS, https://www.acm.org/education/curricula-recommendations (current July 4, 2018).

Johns, C. (1995). "Framing learning through reflection within Carper's fundamental ways of knowing in nursing," *Journal of Advanced Nursing,* Vol. 22, pp. 226-234.

Krathwohl, D. R., Bloom, B. S., & Bertram, B. M. (1973). *Taxonomy of Educational Objectives, the Classification of Educational Goals. Handbook II: Affective Domain*. David McKay Co., Inc., New York, USA.

Liu, K. (2000). *Semiotics in Information Systems Engineering*, Cambridge University Press, Cambridge, UK.

Longenecker, H. E., Feinstein, D. L., & Babb, J. S. (2013). Is there a need for a Computer Information Systems model curriculum? In Proceedings of the Information Systems Educators Conference, ISSN: 2167, 1435-1447.

Shackelford, R., McGettrick, A., Sloan, R., Topi, H., Davies, G., Kamali, R., Cross, J., Impagliazzo, J., LeBlanc, R., & Lunt, B. (2005). *Computing Curricula 2005: The Overview Report*. ACM, AIS, and IEEE-CS. https://www.acm.org/education/curricula-recommendations (current July 4, 2018).

Stamper, R. K. (1973). *Information in Business and Administrative Systems*, John Wiley and Sons, New York, NY.

Stamper, R. K. (1991). "The semiotic framework for information systems research. Information systems research," *Contemporary approaches and emergent traditions*, 515-528.

Topi, H. (2017). "Role of Information Systems in the CC2020 Initiative". *ACM Inroads*, 8(4).

Topi, H, (2017). "Information Systems in CC2020: Comparing Key Structural Elements of Curriculum Recommendations in Computing". *2017 Proceedings*. 9. http://aisel.aisnet.org/siged2017/9

USDoE (2018). "Competency-Based Learning or Personalized Learning," U.S. Department of Education, https://www.ed.gov/oii-news/competency-based-learning-or-personalized-learning (current July 4, 2018)

Waguespack, L., Babb, J. S., Yates, D. (2018). "Triangulating Coding Bootcamps in IS Education: Bootleg Education or Disruptive Innovation?" *Information Systems Education Journal,* 16(6) pp 48-58. http://isedj.org/2018-16/ ISSN: 1545-679X.

Wiggins, G., McTighe, J., and Ebrary, I. (2005). *Understanding by design* (Expanded second edition). Alexandria, VA: Association for Supervision and Curriculum Development.

Wiggins, G., and McTighe, J. 2011. *The Understanding by Design Guide to Creating High-Quality Units*. Alexandria, VA: Association for Supervision and Curriculum Development.

Appendix A – Computing Topics Weighting in CC2005

Knowledge Area	CE		CS		IS		IT		SE	
	min	max	min	max	min	max	min	max	min	max
Programming Fundamentals	4	4	4	5	2	4	2	4	5	5
Integrative Programming	0	2	1	3	2	4	3	5	1	3
Algorithms and Complexity	2	4	4	5	1	2	1	2	3	4
Computer Architecture and Organization	5	5	2	4	1	2	1	2	2	4
Operating Systems Principles & Design	2	5	3	5	1	1	1	2	3	4
Operating Systems Configuration & Use	2	3	2	4	2	3	3	5	2	4
Net Centric Principles and Design	1	3	2	4	1	3	3	4	2	4
Net Centric Use and configuration	1	2	2	3	2	4	4	5	2	3
Platform technologies	0	1	0	2	1	3	2	4	0	3
Theory of Programming Languages	1	2	3	5	0	1	0	1	2	4
Human-Computer Interaction	2	5	2	4	2	5	4	5	3	5
Graphics and Visualization	1	3	1	5	1	1	0	1	1	3
Intelligent Systems (AI)	1	3	2	5	1	1	0	0	0	0
Information Management (DB) Theory	1	3	2	5	1	3	1	1	2	5
Information Management (DB) Practice	1	2	1	4	4	5	3	4	1	4
Scientific computing (Numerical mthds)	0	2	0	5	0	0	0	0	0	0
Legal / Professional / Ethics / Society	2	5	2	4	2	5	2	4	2	5
Information Systems Development	0	2	0	2	5	5	1	3	2	4
Analysis of Business Requirements	0	1	0	1	5	5	1	2	1	3
E-business	0	0	0	0	4	5	1	2	0	3
Analysis of Technical Requirements	2	5	2	4	2	4	3	5	3	5
Engineering Foundations for SW	1	2	1	2	1	1	0	0	2	5
Engineering Economics for SW	1	3	0	1	1	2	0	1	2	3
Software Modeling and Analysis	1	3	2	3	3	3	1	3	4	5
Software Design	2	4	3	5	1	3	1	2	5	5
Software Verification and Validation	1	3	1	2	1	2	1	2	4	5
Software Evolution (maintenance)	1	3	1	1	1	2	1	2	2	4
Software Process	1	1	1	2	1	2	1	1	2	5
Software Quality	1	2	1	2	1	2	1	2	2	4
Comp Systems Engineering	5	5	1	2	0	0	0	0	2	3
Digital logic	5	5	2	3	1	1	1	1	0	3
Embedded Systems	2	5	0	3	0	0	0	1	0	4
Distributed Systems	3	5	1	3	2	4	1	3	2	4
Security: issues and principles	2	3	1	4	2	3	1	3	1	3
Security: implementation and mgt	1	2	1	3	1	3	3	5	1	3
Systems administration	1	2	1	1	1	3	3	5	1	2
Management of Info Systems Org.	0	0	0	0	3	5	0	0	0	0
Systems integration	1	4	1	2	1	4	4	5	1	4
Digital media development	0	2	0	1	1	2	3	5	0	1
Technical support	0	1	0	1	1	3	5	5	0	1

The min value represents the minimum emphasis typically applied to the topic based upon the discipline-specific curriculum guidelines. The max value represents the greatest emphasis (Shackelford, 2005, p. 24).

Appendix B – Comparison of Computing Curricula

	CE2016	CS2013	IS2010	IT2017	MSIS2016	SE2014
Core structure	Knowledge area – Knowledge unit – Learning outcome	Knowledge area – Knowledge unit – Topic/ Learning outcome	Course – Topic/ Learning outcome	IT Domain – Competency	Competency area – Competency category – Competency	Knowledge area – Knowledge unit – Topic
Body of knowledge	Core structure	Core structure	Supporting material	No	No	Core structure
Detailed learning outcomes	Associated with Knowledge units	Associated with Knowledge units	Associated with Courses	"Performances" associated with subdomains are closely related	No	No
High-level graduate characteristics	Yes	Yes	Yes	Yes	Yes	Yes
Competency framework	No	No	No	Yes	Yes	No
Professional profiles	No	No	Yes	No	Yes	No
Note: The original idea for this table emerged from collaborative work at the August 2017 CC2020 Task Force meeting. The collaborative process and contributions of CC2020 are greatly appreciated.						

Adapted from (Topi, 2017b)

Appendix C – Bloom's (Revised) of Educational Objectives: Performance Verbs Proposed in IT2017

Explain	Interpret	Apply	Demonstrate Perspective	Show Empathy	Have Self-Knowledge
demonstrate derive describe how design exhibit express induce instruct justify model predict prove show how synthesize teach	create analogies critique document evaluate illustrate judge make sense of make meaning of provide metaphors read between the lines represent tell a story of translate	adapt build create debug decide design exhibit invent perform produce propose solve test use	analyze argue compare contrast criticize infer	assume role of be like be open to believe consider imagine relate role play	be aware of realize recognize reflect self-assess

IT2017, Table 6.4: Performance verbs to generate ideas for performance goals and professional practice [Wiggins, McTighe, 2011]

Appendix D – Bloom's (Revised) of Educational Objectives: Facets of Learning (Anderson, 2001)

Appendix E - IT2017 Domains of Competency

Essential IT Domains and Levels of Student Engagement	
ITE-CSP Cybersecurity Principles [6%] ITE-CSP-01 Perspectives and impact [L1] ITE-CSP-02 Policy goals and mechanisms [L1] ITE-CSP-03 Security services, mechanisms, and countermeasures [L2] ITE-CSP-04 Cyber-attacks and detection [L2] ITE-CSP-05 High assurance systems [L2] ITE-CSP-06 Vulnerabilities, threats, and risk [L2] ITE-CSP-07 Anonymity systems [L1] ITE-CSP-08 Usable security [L1] ITE-CSP-09 Cryptography overview [L1] ITE-CSP-10 Malware fundamentals [L1] ITE-CSP-11 Mitigation and recovery [L1] ITE-CSP-12 Personal information [L1] ITE-CSP-13 Operational issues [L2] ITE-CSP-14 Reporting requirements [L1]	**ITE-GPP Global Professional Practice** [3%] ITE-GPP-01 Perspectives and impact [L1] ITE-GPP-02 Professional issues and responsibilities [L1] ITE-GPP-03 IT governance and resource management [L1] ITE-GPP-04 Risk identification and evaluation [L1] ITE-GPP-05 Environmental issues [L1] ITE-GPP-06 Ethical, legal, and privacy issues [L1] ITE-GPP-07 Intellectual property [L1] ITE-GPP-08 Project management principles [L1] ITE-GPP-09 Communications [L1] ITE-GPP-10 Teamwork and conflict management [L1] ITE-GPP-11 Employability skills and careers in IT [L1] ITE-GPP-12 Information systems principles [L1]
ITE-IMA Information Management [6%] ITE-IMA-01 Perspectives and impact [L1] ITE-IMA-02 Data-information concepts [L2] ITE-IMA-03 Data modeling [L3] ITE-IMA-04 Database query languages [L3] ITE-IMA-05 Data organization architecture [L3] ITE-IMA-06 Special-purpose databases [L1] ITE-IMA-07 Managing the database environment [L2]	**ITE-IST Integrated Systems Technology** [3%] ITE-IST-01 Perspectives and impact [L1] ITE-IST-02 Data mapping and exchange [L2] ITE-IST-03 Intersystem communication protocols [L2] ITE-IST-04 Integrative programming [L2] ITE-IST-05 Scripting techniques [L2] ITE-IST-06 Defensible integration [L1]
ITE-NET Networking [5%] ITE-NET-01 Perspectives and impact [L1] ITE-NET-02 Foundations of networking [L1] ITE-NET-03 Physical layer [L2] ITE-NET-04 Networking and interconnectivity [L3] ITE-NET-05 Routing, switching, and internetworking [L2] ITE-NET-06 Application networking services [L2] ITE-NET-07 Network management [L3]	**ITE-PFT Platform Technologies** [1%] ITE-PFT-01 Perspectives and impact [L1] ITE-PFT-02 Operating systems [L3] ITE-PFT-03 Computing infrastructures [L1] ITE-PFT-04 Architecture and organization [L1] ITE-PFT-05 Application execution environment [L1]
ITE-SPA System Paradigms [6%] ITE-SPA-01 Perspectives and impact [L1] ITE-SPA-02 Requirements [L2] ITE-SPA-03 System architecture [L1] ITE-SPA-04 Acquisition and sourcing [L2] ITE-SPA-05 Testing and quality assurance [L2] ITE-SPA-06 Integration and deployment [L2] ITE-SPA-07 System governance [L2] ITE-SPA-08 Operational activities [L3] ITE-SPA-09 Operational domains[L3] ITE-SPA-10 Performance analysis [L1]	**ITE-SWF Software Fundamentals** [4%] ITE-SWF-01 Perspectives and impact [L1] ITE-SWF-02 Concepts and techniques [L2] ITE-SWF-03 Problem-solving strategies [L1] ITE-SWF-04 Program development [L3] ITE-SWF-05 Fundamental data structures [L2] ITE-SWF-06 Algorithm principles and development [L2] ITE-SWF-07 Modern app programming practices [L1]
ITE-UXD User Experience Design [3%] ITE-UXD-01 Perspectives and impact [L1] ITE-UXD-02 Human factors in design [L2] ITE-UXD-03 Effective interfaces [L2] ITE-UXD-04 Application domain aspects [L1] ITE-UXD-05 Affective user experiences [L1] ITE-UXD-06 Human-centered evaluation [L1] ITE-UXD-07 Assistive technologies and accessibility [L1] ITE-UXD-08 User advocacy [L1]	**ITE-WMS Web and Mobile Systems** [3%] ITE-WMS-01 Perspectives and impact [L1] ITE-WMS-02 Technologies [L2] ITE-WMS-03 Digital media [L2] ITE-WMS-04 Applications concepts [L2] ITE-WMS-05 Development Frameworks [L2] ITE-WMS-06 Vulnerabilities [L1] ITE-WMS-07 Social software [L1]

Essential IT Domains (IT2017, Table 6.2a, p. 50)

Appendix E - IT2017 Domains of Competency (continued)

Supplemental IT Domains and Levels of Student Engagement	
ITS-ANE Applied Networks [4%] ITS-ANE-01 Proprietary networks [L2] ITS-ANE-02 Network programming [L2] ITS-ANE-03 Routing protocols [L2] ITS-ANE-04 Mobile networks [L2] ITS-ANE-05 Wireless networks [L2] ITS-ANE-06 Storage area networks [L1] ITS-ANE-07 Applications for networks [L2]	**ITS-CCO Cloud Computing** [4%] ITS-CCO-01 Perspectives and impact [L1] ITS-CCO-02 Concepts and fundamentals [L2] ITS-CCO-03 Security and data considerations [L2] ITS-CCO-04 Using cloud computing applications [L2] ITS-CCO-05 Architecture [L2] ITS-CCO-06 Development in the cloud [L2] ITS-CCO-07 Cloud infrastructure and data [L2]
ITS-CEC Cybersecurity Emerging Challenges [4%] ITS-CEC-01 Case studies and lessons learned [L1] ITS-CEC-02 Network forensics [L2] ITS-CEC-03 Stored data forensics [L2] ITS-CEC-04 Mobile forensics [L1] ITS-CEC-05 Cloud security [L1] ITS-CEC-06 Security metrics [L1] ITS-CEC-07 Malware analysis [L1] ITS-CEC-08 Supply chain and software assurance [L1] ITS-CEC-09 Personnel and human security [L1] ITS-CEC-10 Social dimensions [L1] ITS-CEC-11 Security implementations [L1] ITS-CEC-12 Cyber-physical systems and the IoT [L1]	**ITS-DSA Data Scalability and Analytics** [4%] ITS-DSA-01 Perspectives and impact [L1] ITS-DSA-02 Large-scale data challenges [L2] ITS-DSA-03 Data management [L2] ITS-DSA-04 Methods, techniques, and tools [L2] ITS-DSA-05 Data governance [L2] ITS-DSA-06 Applications [L2]
ITS-IOT Internet of Things [4%] ITS-IOT-01 Perspectives and impact [L1] ITS-IOT-02 IoT architectures [L2] ITS-IOT-03 Sensor and actuator interfacing [L1] ITS-IOT-04 Data acquisition [L1] ITS-IOT-05 Wireless sensor networks [L2] ITS-IOT-06 Ad-hoc networks [L1] ITS-IOT-07 Automatic control [L2] ITS-IOT-08 Intelligent information processing [L2] ITS-IOT-09 IoT application and design [L2]	**ITS-MAP Mobile Applications** [3%] ITS-MAP-01 Perspectives and impact [L1] ITS-MAP-02 Architectures [L1] ITS-MAP-03 Multiplatform mobile application development [L2] ITS-MAP-04 Servers and notifications [L1] ITS-MAP-05 Performance issues [L1] ITS-MAP-06 Views and gestures [L1] ITS-MAP-07 Interface implementations [L2] ITS-MAP-08 Camera, state, and documents interaction [L1] ITS-MAP-09 2D graphic and animation [L1]
ITS-SDM Software Development and Management [2%] ITS-SDM-01 Process models and activities [L2] ITS-SDM-02 Platform-based development [L1] ITS-SDM-03 Tools and services [L2] ITS-SDM-04 Management [L2] ITS-SDM-05 Deployment, operations, maintenance [L2]	**ITS-SRE Social Responsibility** [2%] ITS-SRE-01 Social context of computing [L2] ITS-SRE-02 Goals, plans, tasks, deadlines, and risks [L2] ITS-SRE-03 Government role and regulations [L1] ITS-SRE-04 Global challenges and approaches [L1] ITS-SRE-05 Risk management [L1] ITS-SRE-06 Sustainable Computing [L1]
ITS-VSS Virtual Systems and Services [4%] ITS-VSS-01 Perspectives and impact [L1] ITS-VSS-02 Application of virtualization [L2] ITS-VSS-03 User platform virtualization [L1] ITS-VSS-04 Server virtualization [L1] ITS-VSS-05 Network virtualization [L2] ITS-VSS-06 Cluster design and administration [L2] ITS-VSS-07 Software cluster applications [L2] ITS-VSS-08 Storage [L1]	

Supplemental IT Domains (IT2017, Table 6.2b, p. 51)

IT Essential Mathematics and Levels of Student Engagement
ITM-DSC Discrete Structures ITM-DSC-01 Perspectives and impact [L1] ITM-DSC-02 Sets [L1] ITM-DSC-03 Functions and relations [L1] ITM-DSC-04 Proof techniques [L1] ITM-DSC-05 Logic [L1] ITM-DSC-06 Boolean algebra principles [L1] ITM-DSC-07 Minimization [L1] ITM-DSC-08 Graphs and trees [L2] ITM-DSC-09 Combinatorics [L1] ITM-DSC-10 Iteration and recursion [L1] ITM-DSC-11 Complexity Analysis [L1] ITM-DSC-12 Discrete information technology applications [L1]

Related IT Essential Mathematics (IT2017, Table 6.2c, p. 52)

Appendix F - IT2017 Essential IT Domain Cluster Competency Examples

ITE-IMA Domain : Information Management

Scope

1. Tools and techniques for efficient data modeling, collection, organization, retrieval, and management.
2. How to extract information from data to make data meaningful to the organization.
3. How to develop, deploy, manage and integrate data and information systems to support the organization.
4. Safety and security issues associated with data and information.
5. Tools and techniques for producing useful knowledge from information.

Competencies

A. Express how the growth of the internet and demands for information have changed data handling and transactional and analytical processing, and led to the creation of special purpose databases. (*Requirements*)
B. Design and implement a physical model based on appropriate organization rules for a given scenario including the impact of normalization and indexes. (*Requirements and development*)
C. Create working SQL statements for simple and intermediate queries to create and modify data and database objects to store, manipulate and analyze enterprise data. (*Testing and performance*)
D. Analyze ways data fragmentation, replication, and allocation affect database performance in an enterprise environment. (*Integration and evaluation*)
E. Perform major database administration tasks such as create and manage database users, roles and privileges, backup, and restore database objects to ensure organizational efficiency, continuity, and information security. (*Testing and performance*)

Subdomains

ITE-IMA-01	Perspectives and impact [L1]	ITE-IMA-05	Data organization architecture [L3]
ITE-IMA-02	Data-information concepts [L2]	ITE-IMA-06	Special-purpose databases [L1]
ITE-IMA-03	Data modeling [L3]	ITE-IMA-07	Managing the database environment [L2]
ITE-IMA-04	Database query languages [L3]		

ITE-IMA Domain: Information Management (IT2017, p. 56)

ITE-SWF Domain: Software Fundamentals

Scope

1. Skills and fundamental programming concepts, data structures, and algorithmic processes
2. Programming strategies and practices for efficient problem solving
3. Programming paradigms to solve a variety of programming problems

Competencies

A. Use multiple levels of abstraction and select appropriate data structures to create a new program that is socially relevant and requires teamwork. (*Program development*)
B. Evaluate how to write a program in terms of program style, intended behavior on specific inputs, correctness of program components, and descriptions of program functionality. (*App development practices*)
C. Develop algorithms to solve a computational problem and explain how programs implement algorithms in terms of instruction processing, program execution, and running processes. (*Algorithm development*)
D. Collaborate in the creation of an interesting and relevant app (mobile or web) based on user experience design, functionality, and security analysis and build the app's program using standard libraries, unit testing tools, and collaborative version control. (*App development practices*)

Subdomains

ITE-SWF-01	Perspectives and impact [L1]	ITE-SWF-05	Fundamental data structures [L2]
ITE-SWF-02	Concepts and techniques [L2]	ITE-SWF-06	Algorithm principles and development [L2]
ITE-SWF-03	Problem-solving strategies [L1]	ITE-SWF-07	Modern app programming practices [L1]
ITE-SWF-04	Program development [L3]		

ITE-SWF Domain: Software Fundamentals (IT2017, p. 58)

Appendix G - IT2017 IT Domain Performances

ITE-IMA Information Management (IT2017, p. 92)

ITE-IMA-01 Perspectives and impact
- a. Describe how data storage and retrieval has changed over time.
- b. Justify the advantages of a database approach compared to traditional file processing.
- c. Describe how the growth of the internet and demands for information for users outside the organization (customers and suppliers) impact data handling and processing.
- d. Tell a brief history of database models and their evolution.

ITE-IMA-02 Data-information concepts
- a. Describe the role of data, information, and databases in organizations.
- b. Compare and use key terms such as: information, data, database, database management system, metadata, and data mining.
- c. Illustrate data quality, accuracy, and timeliness, and explain how their absence will impact organizations.
- d. Describe mechanisms for data collection and their implications (automated data collection, input forms, sources).
- e. Describe basic issues of data retention, including the need for retention, physical storage, backup, and security.

ITE-IMA-03 Data modeling
- a. Design Entity Relationship diagrams based on appropriate organizational rules for a given scenario.
- b. Describe the relationship between a logical model and a physical model.
- c. Evaluate importance of database constraints.
- d. Design a physical model for the best performance including impact of normalization and indexes.
- e. Compare and contrast the differences and similarities between the relational and the dimensional data modeling (OLTP vs. OLAP).

ITE-IMA-04 Database query languages
- a. Create, modify, and query database objects using the Structured Query Language (SQL).
- b. Perform filtering and sorting data using various clauses including where, order by, between, like, group by, and having.
- c. Use joins to select data across multiple tables.
- d. Use embedded SQL queries.
- e. Perform calculations in a query using calculated fields and aggregate functions.
- f. Create updatable and non-updatable views.

ITE-IMA-05 Data organization architecture
- a. Demonstrate select, project, union, intersection, set difference, and natural join relational operations using simple example relations provided.
- b. Contrast and compare relational databases concepts and non-relational databases including object-oriented, XML, NewSQL and NoSQL databases.
- c. Express the relationship between functional dependencies and keys, and give examples.
- d. Evaluate data integrity and provide examples of entity and referential integrity.
- e. Analyze how data fragmentation, replication and allocation affect database performance.

ITE-IMA-06 Special-purpose databases
- a. Describe major concepts of object oriented, XML, NewSQL, and NoSQL databases.
- b. Demonstrate an understanding of online analytical processing and data warehouse systems.
- c. Describe methods of data mining and what insights may be gained by these methods.

ITE-IMA-07 Managing the database environment
- a. Contrast and compare data administration and database administration.
- b. Describe tasks commonly performed by database administrators.
- c. Create and manage database users, roles, and privileges.
- d. Consider the concept of database security and backup and recovery.
- e. Evaluate the importance of metadata in database environment.

ITE-SWF Software Fundamentals *(IT2017, p. 96)*

ITE-SWF-01 Perspectives and impact
- a. Reflect on how the creation of software has changed our lives.
- b. Synthesize how software has helped people, organizations, and society to solve problems.
- c. Describe several ways in which software has created new knowledge.

ITE-SWF-02 Concepts and techniques
- a. Compare multiple levels of abstraction to write programs (constants, expressions, statements, procedures, parameterization, and libraries).
- b. Select appropriate built-in data types and library data structures (abstract data types) to model, represent, and process program data.
- c. Use procedures and parameterization to reduce the complexity of writing and maintaining programs and to generalize solutions.
- d. Explain multiple levels of hardware architecture abstractions (processor, special purpose cards, memory organization, and storage) and software abstractions (source code, integrated components, running processes) involved in developing complex programs.
- e. Create new programs by modifying and combining existing programs.

Appendix G - IT2017 IT Domain Performances (continued)

ITE-SWF-03 Problem-solving strategies
- a. Explain abstractions used to represent digital data.
- b. Develop abstractions when writing a program or an IT artifact.
- c. Apply decomposition strategy to design a solution to a complex problem.
- d. Explain appropriateness of iterative and recursive problem solutions.
- e. Write programs that use iterative and recursive techniques to solve computational problems.

ITE-SWF-04 Program development
- a. Develop a correct program to solve problems by using an iterative process, documentation of program components, and consultation with program users.
- b. Use appropriate abstractions to facilitate writing programs: collections, procedures, application programming interfaces, and libraries.
- c. Evaluate how a program is written in terms of program style, intended behavior on specific inputs, correctness of program components, and descriptions of program functionality.
- d. Develop a program by using tools relevant to current industry practices: version control, project hosting, and deployment services.
- e. Demonstrate collaboration strategies that consider multiple perspectives, diverse talents, and sociocultural experiences.

ITE-SWF-05 Fundamental data structures
- a. Write programs that use data structures (built-in, library, and programmer-defined): strings, lists, and maps.
- b. Analyze the performance of different implementations of data structures.
- c. Decide on appropriate data structures for modeling a given problem.
- d. Explain appropriateness of selected data structures.

ITE-SWF-06 Algorithm principles and development
- a. Describe why and how algorithms solve computational problems.
- b. Create algorithms to solve a computational problem.
- c. Explain how programs implement algorithms in terms of instruction processing, program execution, and running processes.
- d. Apply appropriate mathematical concepts in programing: expressions, abstract data types, recurrence relations, and formal reasoning on algorithm's efficiency and correctness.
- e. Evaluate empirically the efficiency of an algorithm.

ITE-SWF-07 Modern app programming practices
- a. Create web and mobile apps with effective interfaces that respond to events generated by rich user interactions, sensors, and other capabilities of the computing device.
- b. Analyze usability, functionality, and suitability of an app program.
- c. Collaborate in the creation of interesting and relevant apps.
- d. Build and debug app programs using standard libraries, unit testing tools, and debuggers.
    1. Evaluate readability and clarity of app programs based on program style, documentation, pre- and post-conditions, and procedural abstractions.