Computing Curricula 2020

CC2020

Paradigms for Future Computing Curricula

encompassing undergraduate programs in

Computer Engineering
Computer Science
Cybersecurity
Information Systems
Information Technology
Software Engineering
with
Data Science
(Under Development)

A Computing Curricula Series Report

Association for Computing Machinery (ACM)
IEEE Computer Society (IEEE-CS)

Version 44 | 2020 November 15

ISBN: 000-0-0000-0000-0
DOI: 10.0000/0000000
Web link: https://dl.acm.org/citation.cfm?id=0000000

When available, you may order additional copies from:
ACM Order Department, P.O. Box 30777, New York, NY 10087-0777

Sponsoring Societies
This report was possible by financial support from:

Association for Computing Machinery (ACM)
IEEE Computer Society (IEEE-CS)
and
ACM China
ACM Europe
ACM India
Association for Information Systems (AIS)
Education SIG of the Association for Information Systems and Computing Academic Professionals (ED-SIG)
Italian Association of Computer Scientists (GRIN)
Informatics Europe
Special Interest Group on Computer Human Interaction of (SIGCHI)

The following organizations have officially endorsed this report:

ACM
ACM China
ACM SIGCSE China
Association for Information Systems (AIS)
Brazilian Computer Society
China Computer Federation (CCF)
Computing and Information Technology Research and Education New Zealand (CITRENZ)
IEEE Computer Society (IEEE-CS)
Information Processing Society of Japan (IPSJ)
Information Systems and Computing Academic Professionals - Education Special Interest Group (ISCAP/EDSIG)
Italian Association of Computer Scientists (GRIN)
International Federation for Information Processing (IFIP)
IT Professional New Zealand (ITPNZ)
Latin American Center on Computing (CLEI)
Spanish Scientific Association ADIE (Asociación para el Desarrollo de la Informática Educativa - Computers in Education Association)
Spanish Scientific Association SCIE (Sociedad Cientifica Informatica Española)
Spanish Scientific Society AENUI (Asociación de Enseñantes Universitarios de la Informática - Association of University Teachers of Informatics)

CC2020 Task Force

Co-Chairs

Alison Clear
Eastern Institute of Technology
New Zealand

Allen Parrish
University of Alabama
United States of America

Steering Committee

Paolo Ciancarini
University of Bologna
Italy

Ernesto Cuadros-Vargas
Latin American Center for Computing Studies (CLEI)
Peru

Stephen Frezza
Gannon University
United States of America

Judith Gal-Ezer
Open University
Israel

John Impagliazzo
Hofstra University
United States of America

Arnold Pears
KTH Royal Institute of Technology
Sweden

Shingo Takada
Keio University
Japan

Heikki Topi
Bentley University
United States of America

Gerrit van der Veer
Vrije Universiteit
Netherlands

Abhijat Vichare
ACM India
India

Les Waguespack
Bentley University
United States of America

Pearl Wang
George Mason University
United States of America

Ming Zhang
Peking University
China

Other Task Force Members

(Next Page)

CC2020 Task Force
(continued)

Hala Alrumaih
Imam Mohammad Ibn Saud Islamic Univ.
Saudi Arabia

Renata Araujo
Brazilian Computer Society
Brazil

Jeffrey Babb
West Texas A&M University
United States of America

Olga Bogoyavlenskaya
Petrozavodsk State University
Russia

Héctor Cancela
Universidad de la República
Uruguay

Juan Chen
National University of Defense Technology
China

Adrienne Decker
University of Buffalo
United States of America

Eric Durant
Milwaukee School of Engineering
United States of America

Marisa Exter
Purdue University
United States of America

Beatriz Florián Gaviria
Universidad del Valle
Colombia

Itana Maria Gimenes
Brazilian Computer Society
Brazil

Steven Gordon
Ohio State University
United States of America

Eiji Hayashiguchi
Waseda University, CTO VJP Co. Ltd.
Japan

Amey Karkare
India Institute of Technology – Kanpur
India

Richard Le Blanc
Seattle University
United States of America

Paul Leidig
Grand Valley State University
United States of America

David Lopez
Universitat Politècnica de Catalunya
Spain

Barry Lunt
Brigham Young University
United States of America

Linda Marshall
University of Pretoria
South Africa

Bruce McMillin
Missouri University of Science and Technology
United States of America

Tania McVeety
IBM
United States of America

Nancy Mead
Carnegie Mellon University
United States of America

Teresa Pereira
Instituto Politécnico de Viana Castelo
Portugal

Melinda Reno
Deloitte Consulting
United States of America

Ariel Sabiguero
Universidad de la República
Uruguay

Fermin Sanchez
Universitat Politècnica de Catalunya
Spain

Nello Scarabottolo
Università di Milano
Italy

Yann Secq
University of Lille
France

Simon
University of Newcastle
Australia

Ye Tian
ByteDance
China

Paul Tymann
Rochester Institute of Technology
United States of America

Barbara Viola
Viotech Solutions
United States of America

Xi Wu
Chengdu Univ. of Information Technology
China

Jane (Xiaochun) Yang
AchieveFun Info Technology
China

Stuart Zweben
Ohio State University
United States of America

Contents

Executive Summary

The field of computing has dramatically influenced science, engineering, business, education, philanthropy, and many other areas of human endeavor. In today's world, nearly everyone uses computers as part of everyday life. From smartphones and televisions to guidance systems, computing continues to be present in human life. This computing landscape offers students many challenging career opportunities. For those who are working in industry and government, computing is and will continue to be an essential component in shaping the future for humanity.

The computing disciplines need to attract quality students from a broad and diverse cross-section of the public and prepare them to be capable and responsible professionals. Over decades, professional and scientific computing societies have taken leading roles in providing support for higher education in various ways, particularly in the formulation of curricular guidelines. The landmark report Computing Curricula 2005 (CC2005), also known as the Overview Report, consolidated undergraduate computing curricula as they existed in 2005. It contrasted published computing curricular guidelines for computer engineering, computer science, information systems, information technology, and software engineering. It also illustrated the focus of these five curricula and provided tables to highlight the topic intensity and graduate profiles. CC2005 became a positive contribution to computing education.

Since 2005, much has changed in the computing field and, simultaneously, in the computing education world. The computer engineering reports progressed from CE2004 to CE2016; computer science reports from CS2001 to CS2008 to CS2013. Information systems progressed from IS2002 to IS2010 with a new report pending. The initial information technology report was in draft form in 2005, eventually to become IT2008, and then IT2017. The software engineering report SE2004 became SE2014. Additionally, the computing field saw an emergence of cybersecurity which led to the CSEC2017 report. A data science report is currently under development. It became apparent that a need existed to create a contemporary new report called Computing Curricula 2020, known also as CC2020.

For this purpose, a task force of fifty people from twenty countries, with a fifteen-member steering committee carrying the main operational responsibilities, has examined undergraduate curriculum guidelines and has referred to the computing professions and other supporting information as necessary. This report does not address graduate computing education or pre-baccalaureate education, although it occasionally mentions these areas.

This CC2020 report encompasses most of the themes contained in its predecessor. However, the changing dynamics of computing, computing education research, and changes in the workplace have resulted in many new "add-ons" and features that did not appear in the earlier report. Some of these additions include the following.

- Focusing on competency.
- Transitioning from knowledge-based learning to competency-based learning.
- Expanding curricular disciplines to include cybersecurity as well as data science.
- Expanding curricular and competency diagrams and visualizations.
- Establishing an interactive website that will bring CC2020 results to public use.
- Charting a framework for future computing curricular activities.

The CC2020 report covers undergraduate programs in computer engineering, computer science, cybersecurity, information systems, information technology, software engineering, and data science (under development.) It also provides a brief history of the evolution of previous curricular reports. Four important guidelines were followed.

1. The report must preserve and support the notion of computing in current and future decades world-wide.
2. The report must capture future trends and visions from industry, research, and "grass-roots" developments.
3. The report must be expansive and support existing, emerging, and future computing programs for its constituents.
4. The report must be flexible to achieve global enduring acceptance and be adaptable within multiple educational contexts.

The stakeholders or constituents of this report are prospective students and their parents, current students, industry and governmental officials, computing educators, and educational organizations and authorities. Although computing as a discipline has been around for more than eighty years, many population groups are still not clear about the subject area or what it means. The philosophy underpinning the CC2020 report is to treat computing as a meta-discipline – a collection of disciplines having a central focus of computing.

When compared with CC2005, the CC2020 report moves from knowledge-based learning to competency-based learning. Competency requires demonstration of human behavior with knowledge and skills. In general terms, one can think of competencies as the qualities an individual must possess to be effective in a job, role, function, task, or duty.

There is a general agreement in educational circles that career success requires three things:

- Knowledge – "know-what" – designates a proficiency in core concepts and content, and application of learning to new situations
- Skills – "know-how" – the ability to carry out tasks with determined results
- Dispositions – "know-why" – intellectual, social or moral tendencies.

Hence, any definition of competency must connect the three dimensions within a context or task represented as:

Competency = Knowledge + Skills + Dispositions

This report centers on competencies and develops a competency framework.

An important aspect of this report is that it addresses the need for visualization tools. For example, by utilizing the competency dimensions of knowledge, skills, and dispositions, it is possible to generate a visual diagram that shows the commonalities and the differences between two computing disciplines for prospective students. These competency-based visuals as well as other visuals provide a rich set of perspectives on computing disciplines.

The report also suggests directions for the future. It emphasizes the need for industry engagement to formulate workplace competencies and the need for professional advisory boards to become involved with the development of meaningful computing programs, for example, through internship programs.

It is not the intent of this report to completely solve the nomenclature problem surrounding the computing field. For example, “information technology” as used in this report refers to a subset of the computing field; some areas of the world use this term to represent the entire computing field. This "Tower of Babel" challenge may never achieve a solution. However, stakeholders must be aware of the nuances and differences in the meaning of different terminologies used in different parts of the world. Universal acceptance of global diversity and cultural sensitivity are essential in all fields, especially in the field of computing which is very diverse itself. Degree structures are different in different countries and sometimes even in the same country. Generally, there exist two, three, and four-year programs at the undergraduate level.

This CC2020 report does not provide specific curricula for each computing discipline. Instead, the report suggests and provides many opportunities. These include refreshing the paradigm of teaching and educating, moving from knowledge or outcomes to proficiencies, and engaging graduates to exploit the benefits of workplace competencies. These are described in the closing chapters of the report.

The report is the result of an unprecedented cooperative effort among several computing organizations spanning twenty countries. As one of the volumes of the “Computing Curricula Series,” it provides introspection and analysis of six computing disciplines based on current curricular guidelines that are the product of many years of experimentation and refinement by industry leaders, computing educators, and faculty colleagues in other disciplines. The academia-employer partnerships that will follow in the wake of this report will help build even stronger computing programs for undergraduates worldwide.

Finally, this report is more than an overview of curricular guidelines. The overriding goal is to provide a useful and dynamic pathway toward the 2030s. The report also provides a perspective on the major computing disciplines as they currently exist and how they might exist in the future. It will help guide students, industry, and academia in the preparation of capable computing graduates for the future. CC2020 will help shape the future of computing education.

— CC2020 Task Force

Chapter 1: Introducing CC2020

The Computing Curriculum 2020 (CC2020) project is an initiative launched jointly by several professional computing societies to summarize and synthesize the current state of curricular guidelines for academic programs that grant baccalaureate-level degrees in computing as well as propose a vision for future curricular guidelines. This project aims not only to reflect the state-of-the-art in computing education and practice, but also to provide insights into the future of the field of computing education for the 2020s and beyond. The participating societies engaged a task force of individuals representing organizations from academia, industry, and government. The principal organizations involved are the Association for Computing Machinery (ACM) and the IEEE Computer Society (IEEE-CS). Other organizations include the Association for Information Systems (AIS) and the Education Special Interest Group of Information Systems and Computing Academic Professionals (EDSIG/ISCAP), and the ACM Special Interest Group for Computer Human Interaction (SIGCHI). Project collaborators include: Information Processing Society of Japan (IPSJ), the Chinese Computing Federation (CCF), the Latin American Center on Computing (CLEI), ACM India, Informatics for All, and Informatics Europe. The results from this initiative provide a durable portfolio of resources useful to students, industry, government agencies, educational institutions, and the public on a global scale. This report is one key element of this portfolio.

1.1: CC2020 Expectations

The Computing Curricula 2005 Overview report CC2005 [Acm02] was an inaugural effort of several computing organizations to provide a perspective on several computing disciplines for which baccalaureate curricula existed. Much in the computing world has changed over fifteen years. Geography and varied conceptions of computing as disciplines, professions, and cultures have influenced the context of degree-granting computing programs. The CC2020 project considers regions of the world by involving organizational representatives from a variety of countries to be part of the project. While currently published curricular guidelines (i.e., computer engineering, computer science, cybersecurity, information systems, information technology, software engineering) and the currently emerging curricular models (i.e. data science) comprise CC2020's central domain of interest, the CC2020 deliverables are intended to inform the process of rethinking existing or shaping new computing degree programs and disciplines.

1.1.1: Project Purpose, Vision and Mission

The following statement reflects the purpose of this CC2020 project.

> *The purpose of the CC2020 project, as a modern extension of the CC2005 report, is to provide global guidance in an evolving computing environment as it affects baccalaureate degree programs in computing worldwide.*

The CC2020 project offers an up-to-date comparison and contrasting of the existing curricular guidelines in order to situate and contextualize them in the landscape of computing education globally. The report also provides a characterization of computing that facilitates designing and evaluating curricula and content, making a case for the development of interactive tools that may be employed by academia and industry to prototype models of proficiency development that are useful for exploring curricular opportunities. This report does not provide updates to the existing individual curricular guidelines. Updating these documents is a task to be carried out by the future respective curriculum guidelines committees.

The results of the CC2020 project should inform students and their guardians, industry, government agencies, and academia on the status and future of computing programs. This report should help current and prospective students and their parents or guardians make informed decisions in selecting and entering computer degree programs. It also assists industry and government to understand profiles and expectations of graduates of computing degree programs. Additionally, it helps computing programs to prepare students and resulting graduates, both academically and professionally, to meet the challenges of the next decade.

The CC2020 task force vision is that:

> *The CC2020 report shall become a sought-after and durable set of guidelines for use by (prospective) students, industry, governments and educational institutions worldwide to assist them to gain insight on the expectations of computing baccalaureate-degree graduates for the next decade.*

Likewise, knowledge alone is not sufficient for an individual to be productive in the changing world of computing. Graduates of computing programs will require technical skills and dispositions that are integrated with knowledge to achieve the professional expectations of a modern workplace. Therefore:

> *The mission of the CC2020 project and this report is to produce a globally accepted framework for specifying and comparing computing baccalaureate degree programs that meet the growing demands of a changing technological world and is useful for students, industry and academia.*

1.1.2: Project Strategies

The CC2020 task force has established a set of goals to achieve the project's vision and mission. These steps form a pathway toward completion of the CC2020 project:

1. Develop a project plan with achievable milestones to complete ongoing projects on time.
2. Develop a robust report that reflects the project's vision and mission.
3. Garner feedback from constituents for the development of this report.
4. Disseminate the CC2020 report worldwide.
5. Evaluate the efficacy of the CC2020 report.

Underlying these steps is the effort to extend the earlier overview report so it incorporates the developments of the past fifteen years as well as the advancements forecast in the next decade. Computing technologies have developed and continue to develop rapidly over time in ways that have had a profound effect on graduate expectations, curriculum design and learning.

The CC2020 report proposes a performance-centered framework expressing what graduates of baccalaureate computing programs should be able to learn and deliver with what they know. This report articulates computing competencies to enable faculty members to implement baccalaureate degree programs that focus on what students should be able to accomplish rather than what instructors should teach. The report draws on learning sciences and educational research and practices to advance the case for learning and curriculum development.

1.1.3: Project Diversity

The CC2020 project promotes sound principles regarding the ways in which computing permeates society on a global scale. Notwithstanding, it is not possible to cover all modes of thinking and all ways of learning. For example, a comprehensive analysis of experiential learning is beyond the scope of this report. Individual institutions and their faculties should use innovative strategies to engage students in the learning process.

There are many pedagogical challenges and opportunities involving the computing field. Although this report addresses the need for accessibility for all people, it does not discuss how this might be achieved. The task force believes such attention should take place at the institutional level as well as through ongoing research by scholars and practitioners.

One underlying theme of the report is the development of computing talent from all sectors and groups in our society. A lack of diversity limits creativity and productivity. It excludes many potentially qualified individuals and can be a significant concern for many employers. For example, the underrepresentation of women in computing in some countries has received much attention [Reg1]. This report recognizes the importance of diversity and recommends that academic computing departments promote best practices to attract and retain greater student diversity.

The CC2020 project has placed inclusion at the core of its activities from the very first step of forming its membership. The task force has a diverse composition by gender, type of work, affiliation, geography, and global professional presence. Some statistics for the task force are as follows.

Number of task force members: 50	Task force co-chairs: 1 woman; 1 man
Number of continents: 6	Steering committee: 5 women, 10 men
Number of countries: 20	Number of international professional societies: 11
Number of women: 21; men: 29	Number of industry-government members: 7
	Number of academic members: 43

The task force is aware that it cannot satisfy the desires of all people. It has made every effort to position this report within the broader computing landscape. As a global document, this report provides guidelines from diverse communities and is not prescriptive in its recommendations.

1.2: Project Stakeholders

The CC2020 project has identified five groups of stakeholders, the members of which may benefit from the competency-based approach to computing education that will be described in this report. The project stakeholders include: (a) prospective students and their parents or guardians, (b) current students, (c) industry, (d) educators, and (e) educational organizations and authorities. All stakeholders should benefit from the outcomes of this CC2020 project.

1.2.1: Prospective Students and their Guardians

When prospective students, supported by their parents or guardians, are considering studying computing at a university, they need to understand differences in computing programs when making their choice. This report could enable them to compare programs so that they can evaluate the extent to which a program aligns with their expectations of a job or a career path. Students may understand that they want to study computing, but very few will likely understand that there are many disciplines and the differences between them. A typical question posed by a prospective student might be:

> *I am considering a computing degree that fits my preferences. Among the candidate schools, there are several computing programs available. Are graduates of these programs expected to work primarily as individuals (e.g., doing coding) or also work with other people?*

1.2.2: Current Students

Current students enrolled at an institution of higher learning could face a choice of courses from their own institution or another institution. This stakeholder category could also apply to students in another discipline who are considering a hybrid curriculum that includes computing components such as a minor in a computing discipline. A typical question posed by a current student might be:

> *Which areas of study does the information systems curriculum of my university emphasize more (with more detailed coverage or longer duration) than the current information systems curriculum guidelines?*

1.2.3: Industry

Industry refers to entities that (1) are hiring graduates, (2) are collaborating with universities to choose or specialize a curriculum or need a special course, or (3) are collaborating in a curriculum by providing internships. For representatives of industry (such as hiring managers), the most important question relates to the preparation of the graduates of a program compared to the expectations of a specific employer. More importantly, industry employers and recruiters need to understand what incoming employees have learned. For example, employers who are looking for software developers would likely prefer to hire individuals who have engaged in deep studies of topics usually found in software engineering or computer science programs. On the other hand, if the employers are looking for individuals who have studied organizational issues in addition to a solid foundation in computing, then they would prefer graduates from an information systems curriculum. A relevant question that industry stakeholders may like answered could be:

> *Our industry requires our employees to have specific knowledge at relevant knowledge levels and several key dispositions. Are there outcomes of a course in curriculum XYZ that are appropriate for the continued professional education for our employees?*

1.2.4: Computing Educators and Curriculum Developers

Computing educators are faculty members or teachers of a computing academic unit within a school or university, and curriculum developers who are responsible for designing and implementing educational experiences related to a computing discipline. This includes academic administrators (for example, deans and department chairs.) These computing educators should understand how their current curriculum, or a prospective curriculum, fits within accepted curricular recommendations. It would be useful if they were able to compare their curriculum to professional curricula guidelines to help them understand what may be missing. They might ask a question as follows.

> *What knowledge areas are applicable for my course? Could I adopt an existing course from elsewhere to fill a gap or provide an alternative in my curriculum?*

1.2.5: Professional Associations, Educational Organizations and Authorities

Educational organizations or authorities are entities who have some authority over university education. Similar stakeholders might include professional organizations or societies, national or regional ministries of education that govern and finance universities, and national or international bodies that rate, assess, or accredit university education, or define qualifications for certification. The following shows a typical question that educational authorities may like answered.

> *Could we accept students from a specified curriculum X to complete curriculum Y?*

1.3: Project Background

Computing curriculum guidelines have been of interest to colleges and universities and their faculty members for more than six decades. The following is a brief summary of the project background.

1.3.1: Brief History

In the 1980s, the ACM and the Computer Society of the Institute for Electrical and Electronics Engineers (IEEE-CS) established a joint committee to update Curriculum'78. The committee's goal was to develop more modern computing curricula (CC) guidelines for baccalaureate, undergraduate degree programs in computing. The committee's effort created Computing Curricula 1991 [Tuc1], also called CC1991 or CC'91. This report, which many educators interpreted as computer science, received only moderate acceptance because by the early 1990s different computing disciplines were maturing (e.g., information systems) or emerging (e.g., computer engineering, information technology, software engineering). However, the efforts of the CC'91 committee resulted in a series of comprehensive reports that reflected not only maturing but also emerging computing disciplines. Many of these documents are available on the ACM website [Acm01]. Additionally, Europe has also formulated computing definitions through the European Higher Education Area (EHEA) [Eur2].

In the late 1990s, ACM and IEEE-CS cooperated to generate the CC2001 report [Acm01] that represented some major advances. This report called for the creation of an overview document; it also called for each of the major computing disciplines recognized at the time to develop its own curricular report. The major areas at the time included computer engineering, computer science, information systems, information technology, and software engineering, although information systems had published its own discipline reports already for two decades. CC2001 recognized the growing and dynamic nature of computing. The number of computing-related disciplines was increasing; hence, work in curricular development was to embrace new computing disciplines as they emerge. The tenets established within the CC2001 report eventually produced the broadly influential CC2005 overview report [Acm02] that was co-sponsored by the ACM, the Association for Information Systems (AIS), and IEEE-CS.

This CC2005 report has received worldwide acclaim by contrasting the differences and the commonalities of diverse computing discipline areas. It was an inaugural effort of several computing organizations to provide perspective on several computing disciplines for which baccalaureate curricula existed. It also described "how the respective computing undergraduate degree programs compare and complement each other" [Acm02 p1]. Chapter 3 will review CC2005 in greater depth.

Since the publication of CC2005, much has changed. Each of the curricula described in 2005 has been updated, in some cases multiple times. New areas of the computing field have gained prominence to warrant production of their own curriculum guidelines. The global and interdisciplinary nature of computing has become even more evident today [Sim1]. The 2005 document was by its own admission, "North-American-centric", and it mentioned the need for future such documents to be more international in scope. The CC2020 project fulfills that promise.

1.3.2: Project Organization and Structure

In 2015, the Association for Computing Machinery (ACM) began to explore avenues through which to update the overview report. In 2016, the ACM decided to proceed with CC2020. It established an exploratory committee to ascertain the need for a new report. Initially, ACM and IEEE-CS became the principal sponsors of the CC2020 project. Other professional organizations joined in the effort with additional sponsorship. These included the ACM China, the Association for Information Systems (AIS), the Education Special Interest Group of Information Systems and Computing Academic Professionals (EDSIG/ISCAP), and the Special Interest Group on Computer Human Interaction (SIGCHI). Project collaborators included ACM India, the Chinese Computing Federation (CCF), GRIN (Italian Association of Computer Scientists), Informatics for All (I4All), Informatics Europe, the Information Processing Society of Japan (IPSJ), and the Latin American Center for Computer Studies (CLEI).

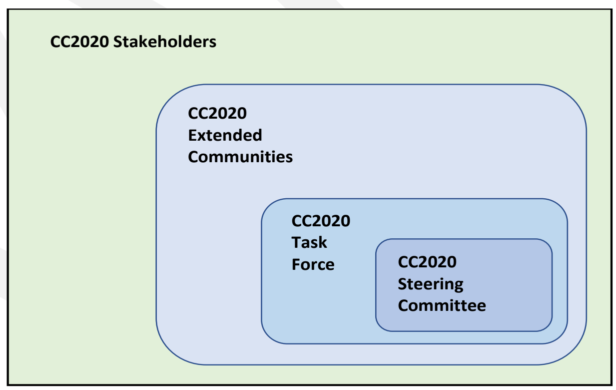


Figure 1.1. CC2020 project structure

The ACM and IEEE Computer Society initially appointed two respective CC2020 project co-chairs. In 2017, each co-chair then recruited representative members of the sponsoring organizations to serve on the CC2020 steering committee. The steering committee was expanded into a task force of fifty volunteers who joined the effort to work on the project and produce this report. Figure 1.1 illustrates the current structure of the CC2020 project. The responsibility of the steering committee has been to define the directions and content of this project, incorporating input from all members of the task force and the extended communities. The steering committee has been responsible for producing this written report, with a particularly important contribution by a small writing and editing team within the steering committee.

1.4: Overall Scope of Computing

There are many types of computing degree programs. Reliable information about the number of different kinds of computing degree programs is difficult to obtain. However, in the past fifteen years, the number and type of computing degree programs available to students has dramatically increased. This report suggests ways that future computing curricula guidelines might develop. To this end, the CC2020 report aims not only to describe the computing subdisciplines currently identified by curricular documents, but also to acknowledge that the boundaries of computing disciplines have expanded and will continue to expand greatly.

1.4.1: Current Discipline Structure

The baccalaureate disciplines for which computing curricula exist or are in the development process at the time of this writing are as follows.

- Computer engineering (CE)
- Computer science (CS)
- Cybersecurity (CSEC)
- Information systems (IS)
- Information technology (IT)
- Software engineering (SE)
- Data science (DS) [1]

Each of these disciplines has a recent volume (or will soon complete a volume) sponsored by ACM and IEEE-CS for undergraduate curriculum guidelines that one or more international professional and scientific societies have endorsed and published. These disciplines have affected a large majority of undergraduate students worldwide who are majoring in computing.

One would expect that groups from other disciplines in computing might undertake the effort to create and maintain international undergraduate curricular guidelines. In such cases, those guidelines will become part of future editions of this report.

1.4.2: Timeline of Curricular Guidelines

The foundation for this report is the set of curriculum standards that currently exist for undergraduate degree programs in major computing-related fields. The diagram in Figure 1.2 illustrates what has become the "computing curricula series," and the top-level overview block, CC2020, represents this report. For the six-existing discipline-specific curricula volumes, each one represents the best judgment of the volunteers representing relevant professional, scientific, and educational associations. Each report serves as a definition of what these degree programs should be and accomplish.

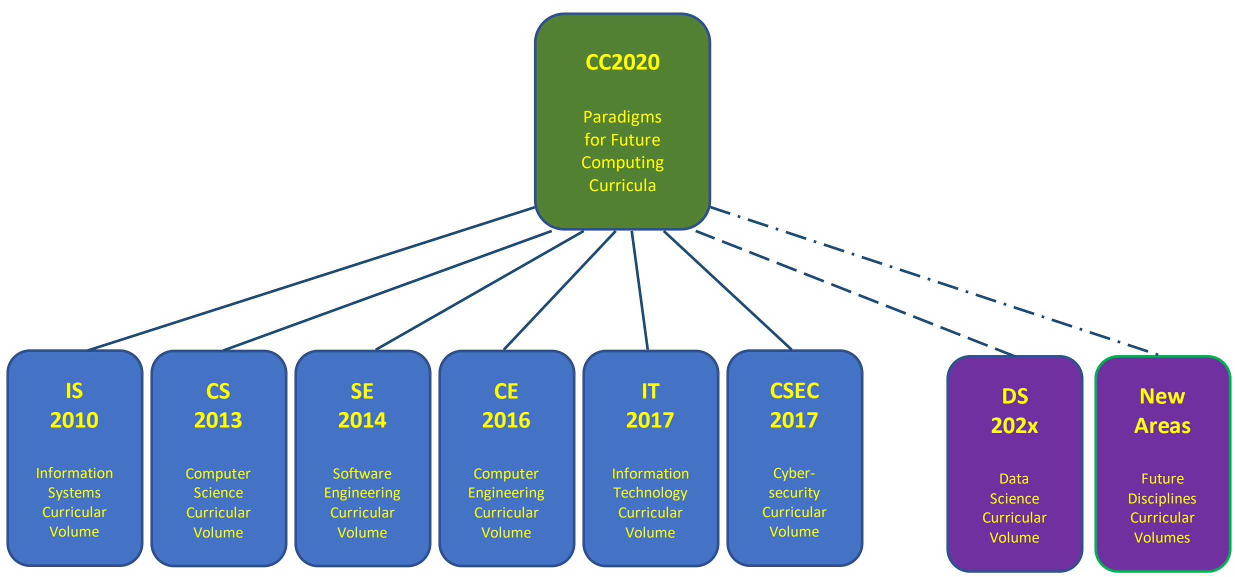


Figure 1.2 Structure of the Computing Curricula Series

[1] Under development with support from ACM.

This report encompasses recent and ongoing curricula efforts including the following:

- Curriculum Guidelines for Undergraduate Degree Programs in Information Systems 2010 (IS 2010)
- Computer Science Curricula 2013 (CS 2013)
- Software Engineering Curricula 2014 (SE 2014)
- Computer Engineering Curricula 2016 (CE 2016)
- Information Technology Curricula 2017 (IT 2017)
- Cybersecurity Curricula 2017 (CSEC 2017)
- Data Science Curricula (under development) 202x (DS202x)
- Other emerging disciplines

The data science curricular guideline report addressing the computing components useful for data engineering, big data, and data analytics is currently under development. Other recent publications that have had some influence in this area include the EDISON Data Science Framework [Edi1] and the document titled "Envisioning the Data Science Discipline: The Undergraduate Perspective: Interim Report" by the National Academies Press [Nas1].

Professional organizations should view the computing curricular reports mentioned above as suggested guidelines instead of strict prescriptions. Curriculum developers have had and still have the freedom to act independently for their constituencies. The anticipation is that undergraduate baccalaureate degree programs will greatly exceed the minimal recommendations suggested in these and subsequent curricular reports.

While some of the mentioned reports are undergoing revision at the time of this writing (e.g., IS2010), the task force has made no effort to update their contents as that endeavor is beyond the mission and authority of this project. Rather, the CC2020 task force has taken current curricula volumes, compared their contents, and synthesized what it believes to be essential descriptive and comparative information. The current curricular volumes contain much detailed information not included here. Readers who want detailed information about any of the disciplines covered in this report should consult the original sources found at the ACM website [Acm01].

In addition to using these reports, the task force has referred to the computing professions and other supporting information as necessary in preparing this report. The report has not focused on other types of undergraduate computing degree programs, on graduate education in computing, or on the identities of the computing research communities. Additionally, it has not included any information or comment about non-traditional computing education such as provided in conjunction with vendor- and government-specific certification programs and massive open online courses (MOOCs). Those areas continue to be deserving of evaluation, but such work is beyond the scope of this project.

1.5: Guiding Principles

The intent of the CC2020 report is to provide a conceptual framework within which to conceptualize computing education as well as the relationships between aptitudes, bodies of knowledge, professional profiles, educational contexts, and degree programs. The CC2020 Task Force acknowledges that the terminology in use has never had universal agreement, and it is subject to a great deal of variation. Terms such as "computer engineering," "computer science," and "information technology" mean radically different things to computing educators and practitioners in different parts of the world and indeed, within national localities. Section 6.1 discusses nomenclature and terminology in the computing field.

1.5.1: Four Principles

The CC2020 task group followed these four principles in developing this CC2020 report.

1. It must preserve and support the notion of computing, both now and in the future, throughout the world.

The term *computing* identifies the broad field involving computers and reflects the reality that this word has become globally ubiquitous. The task force is aware that some regions of the world use different terminologies such as "informatics" or "information and communication technology (ICT)" to have a similar meaning as the word *computing* to represent a field. This report assimilates these similarities and differences.

2. It must capture future trends and visions from industry, from research, and across the entire spectrum of society. This principle has many facets to it. The CC2020 project must remain responsive to general educational needs, changes in technology, as well as existing and emerging technologies. The intended audience for this report includes students, industry/employers, policymakers, professional societies, accreditation agencies, and academic institutions.

3. It must be expansive and support existing, emerging, and future computing programs for its constituents. It is important to contrast different types of computing programs. This principle is important for the CC2020 project and its constituents (e.g., students and prospective students) to aid in their awareness of how one computing discipline (e.g., computer engineering) differs from another computing discipline (e.g., information technology). This principle is also important for creators of educational programs who will use the pending resources to guide the development and enhancement of robust degree programs. The task force also seeks to establish a universal understanding of the terminology used to describe anticipated capabilities of computing graduates in different areas of specialization.

4. It must be flexible to achieve global enduring acceptance and be adaptable within multiple educational contexts. Authors in the computing education field must use language that is globally neutral (not specific to an educational system or context) to reflect documents such as the Bologna Declaration [Bol1]. To some degree, users of this report should be able to use different parts of this report to suit their own needs.

1.5.2: Constituents and Public Outreach

As a follow-up to this report, the CC2020 project website is under development and will include much more information than this report can convey. It is anticipated that this website will include an interactive toolbox of procedures to assist the constituents of this report and the public. A goal is to provide on-demand information for prospective students and for those who advise them to help them make well-informed choices. Another is to provide the means for comparing computing programs across national contexts as well as provide support for finding information for assessing and developing modern and future computing curricula. Figure 1.3 shows the landing page of the CC2020 website [Ccw1].

Figure 1.3. Landing page (partial) of the CC2020 website at https://www.cc2020.net/

Computing itself will continue to evolve. In addition, new computing-related fields are likely to emerge. As updates of existing discipline-specific reports develop and as additional reports for new fields of computing emerge, updated versions of this report will likely be produced. A visit to the ACM website [Acm00] or to the IEEE-CS website [Cos1] will allow users to access the most current version of this and other curricular reports.

1.6: CC2020 Report Structure

This CC2020 report addresses baccalaureate degree programs in computing. The main body of the report consists of seven chapters in addition to this Chapter 1 titled *Introducing CC2020*. Chapter 2, titled *Evolution of Computing Education*, addresses computing and its disciplines in general and the computing landscape. Chapter 3, titled *Knowledge-based Computing Education*, discusses computing curricula guidelines and the knowledge derived from CC2005. Chapter 4, titled *Competency-based Computing Education*, addresses market forces, the importance of skills and disposition, and the CC2020 definition of competency. Chapter 5, titled *Visualizations* describes representations of curricula for the project's stakeholders using modern visualizations. Chapter 6, titled *Global and Professional Considerations*, addresses worldwide nomenclature and degree structures as well as global examples of computing programs. Chapter 7, titled *Curricular Design – Challenges and Opportunities*, examines various aspects of computing education useful for a successful implementation, and Chapter 8, titled *Beyond CC2020*, concludes the report.

Appendix A shows an example of a poster that displayed the use of the CC2005 report for a broader audience. Appendix B presents several computing skill frameworks. Appendix C illustrates examples of competencies for various computing curricula. Appendix D explores the nature of competency-based computing curricula. Appendix E addresses the use of competencies for specification of degree programs in computing. Appendix F addresses a strategy for the development of a visualization repository. Appendix G contains a large set of visualization examples. Appendix H provides a glossary of terms as well as a crosscutting nomenclature as used in different parts of the world. Appendix I summarizes the Chinese "Blue Book" project surrounding agile competencies. Appendix J recognizes contributors to the CC2020 project as well as reviewer contributions.

The CC2020 task force is hopeful that this report will help students to decide on a computing path of study, industry representatives to understand better the profiles of graduating students, and educators to create effective curricula or help them improve the curricula they already have. This CC2020 report, with its recommendations and illustrations, should be a guiding light for computing education worldwide. Its intent is to help those who enable students to develop computing competencies so that the latter can achieve professional success in their future careers.

1.7: Digest of Chapter 1

This chapter reviews the vision, mission, purpose, and development of the CC2020 project. It describes the project strategies and project stakeholders and how they will benefit from this report. It also reviews the background associated with the CC2020 project and explains the guiding principles in detail. The chapter previews the structure of this report.

Chapter 2: Evolution of Computing Education

This chapter discusses some of the background related to the CC2020 project as well as the meaning and landscape of computing. It describes seven of the curricular reports either published or under development by ACM and IEEE-CS. It also addresses extensions of computing disciplines such as emerging curricula, Computing + X and X + Computing scenarios, as well as other curricular reports. The content of this chapter is primarily expressed from an academic perspective. Industry perspectives are covered starting in Chapter 4.

2.1: What is Computing?

In this report, the word *computing* refers to a goal-oriented activity requiring, benefiting from, or associated with the creation and use of computers. As originally expressed in CC2005 [Acm02], computing includes a variety of interpretations such as designing and constructing hardware and software systems for a wide range of purposes: processing, structuring, and managing various kinds of information; problem solving by finding solutions to problems or by proving a solution does not exist; making computer systems behave intelligently; creating and using communications and entertainment media; and finding and gathering information relevant to any particular purpose.

2.1.1: Early Meanings

Early on, computing had a somewhat singular meaning. In its short history, various shades of interpretation have evolved with varying specializations. For example, a person with a background in information systems will view computing somewhat differently from a computer engineer. The emergence of new information technology industries, the increased reliance on computation in all parts of society, and the shifts in the demand for computing throughout a worldwide economy reflect changes in the field and its broad applications [Nrc1]. Because society needs people to do computing well, it is important to understand that computing is not only a profession but also a collection of disciplines [Acm02].

Computing is not just a single area of study, but rather a family of study areas. During the 1990s, important changes in computing, communications technology, and their societal effects led to important changes in this family of disciplines. Those changes included:

- Computer engineering emerging from electrical engineering
- Computer science evolving into a more mature academic discipline
- Information systems expanding as computers became the foundation for organizational processes and work environments
- Information technology emerging as a new discipline that fosters building and maintenance of computing infrastructures
- Software engineering emerging as a discipline based on computer science and computer engineering.

After the 1990s, computing programs around the world saw a maturation. They continued to evolve, thereby creating a greater range of study opportunities for students and educational institutions [Acm02]. Additionally, there were many jobs available focused on software use rather than design and development that accelerated that maturation.

2.1.2: Recent Undertakings

Advances in worldwide curricula development have expanded the scope of the traditional computing disciplines: computer engineering, computer science, information systems, information technology, and software engineering. New curricular efforts have led to significant developments in cybersecurity, data science, and other emerging areas of study. While these efforts are generally acknowledged to be within the frontiers of computing education, what lies at the core of computing and how this core supports future expansions in computing education is less clear.

Section 2.3 of this report describes recent updates to curricular development in the traditional computing areas of study as described above. Additionally, it addresses a recent curricular report in cybersecurity published in 2017. It also previews an emerging ACM effort in the area of data science. The study area of artificial intelligence, an area of renewed interest, is not included in this report because an ACM/IEEE-CS sponsored curricular guideline does not currently exist.

In 2018, the National Academies of Sciences, Engineering and Medicine in the United States described the changing landscape of computing as follows [Nas2]:

> *Two areas have been central in the last decade: the continued and increased need for information security, and data as a resource and driver for decision making. The protection of digital information and data; the protection of software and hardware systems and networks from unauthorized access, change, and destruction; and the education of users to follow best security practices are crucial to every organization. We rely upon a connected, networked, and complex cyberspace with vulnerabilities that is almost continuously under attack. …*
>
> *During the last decade, computing has taken a new, more empirically driven path with the maturing of machine learning, the emergence of data science, and the "big data" revolution. Data science combines computing and statistical methods to identify trends in existing data and generate new knowledge, with significant applications throughout all sectors of the economy, including marketing, retail, finance, business, health care and medicine, agriculture, smart cities, and more. …*
>
> *Software tools and systems for animation, visualization, virtual reality, and conceptualization have emerged as a medium for the arts (digital media and multimedia practices) and are driving advances in the entertainment industry (computer-generated graphics in films and video games, and digital methods in music recording), as well as training and education using virtual environments.*
>
> *Computing has become more pervasive among a host of academic disciplines, beyond just the practical use of ubiquitous software tools. New algorithmic approaches and discoveries are helping to drive advances across a range of fields, leading to new collaborations and an increased demand for deeper knowledge of computing among academics and researchers, challenging conventional disciplinary boundaries.*

This National Academies report has the expectation of generating a profound influence on the development of data engineering and data science worldwide, as well as computer security.

2.2: Landscape of Computing Disciplines

This section of the report provides both historical and contemporary perspectives on the evolution of computing. The section places computing in context as viewed by professionals in computing.

2.2.1: Early Developments

At the earliest stages of the development of computing, education and training for computing jobs were closely associated with research and development of computing technologies as manifested by the manufacturers of the artifacts that industry produced. Relatively soon, however, universities started to offer courses associated with computing. By the end of 1950s, about 150 universities and colleges in the United States offered courses in computing in a broad range of topics ranging from "logical design of computers" through "programming of digital computers" as well as from "information storage and retrieval" to "business and industrial analysis" [Fei1; Ted1]. Fein also provides an insightful discussion that explores the concept of a "computer sciences" discipline and suggests that one such area of study is likely to emerge. Continuing:

> *Most aspects of computers, data processing and the related fields discussed in this study now meet (the specifications of a discipline articulated in the paper) or may be meeting them in the next ten years* [Fei1].

Fein also clearly defined computing as a field of study that consists of multiple disciplines, proposing five different departments: computer, operations research, information and communication, systems, and philosophy of organization. A modern interpretation would roughly correspond to current disciplines such as computer science/computer engineering, operations research/management science, information science, information systems, and computing ethics. It is interesting to see how the breadth of the field links computing as an academic discipline to the practical applications and contexts [Fei1].

In the 1960s, three major streams of academic computing program types emerged: computer science, computer engineering, and information systems. These three had clearly different perspectives: computer science was a highly theoretical study of "information structures and processes and how those structures and processes can be implemented on a digital computer" [Ted1 p45]; computer engineering was an offshoot of electrical engineering that focused on applying established engineering practices and processes to the design and construction of computing hardware; and (management) information systems focused on the practical use of computing in organizations (mostly businesses). Both computer science and information systems had ACM-sponsored curriculum recommendation projects, leading ultimately to *Curriculum 68* [Acm13] for computer science and IS curricula for graduate (1972) [Acm14] and undergraduate (1973) [Acm15] programs.

In 1989, a *Task Force on the Core of Computer Science* characterized the discipline of computing as a combination of three separate but tightly intertwined aspects: theory, abstraction (modeling), and design [Den1]. Those aspects relied on three different intellectual traditions (the task force called them paradigms): the mathematical (or analytical, theoretical, or formalist) tradition, the scientific (or empirical) tradition, and the engineering (or technological) tradition [Ted2, p153].

2.2.2: Contemporary Advances

In the 1970s, 1980s, and 1990s, relatively little changed structurally in computing education: computer engineering, computer science, and information systems all evolved but continued to have separate identities that made it relatively easy for prospective students to choose between different options. However, in the early 2000s, the landscape of computing education started to change significantly. Software engineering emerged as its own discipline with a curriculum recommendation after decades of organizational practice and research. Programs in information technology started to fill the need for graduates with an applied focus on developing and maintaining computing infrastructure and supporting users. At the same time, the five established computing disciplines (CE, CS, IS, IT, and SE) strengthened their collaboration which allowed computing to gain a stronger integrated identity. One of the achievements of CC2005 was the formation of an integrated computing discipline which was the result of the analysis, documentation, and clarification of the relationships between the five subdisciplines. The document illustrated the general characteristics of computing education with Figure 2.1 which summarizes the development of the field during the transformation that took place starting in the 1990s.

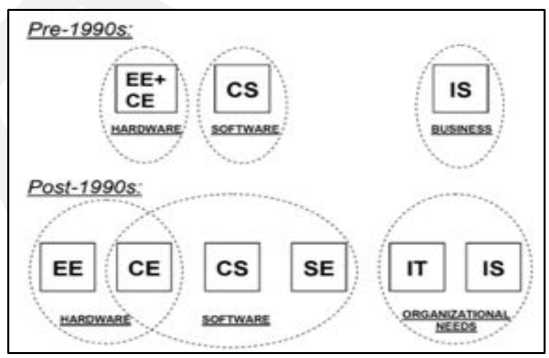


Figure 2.1 Computing disciplines compared, from CC2005

In the 2010s, two new areas emerged as new disciplines in the broader computing space: cybersecurity and data science. In 2017, a curriculum recommendation and accreditation criteria for cybersecurity emerged. Data science, however, often has different instantiations and possible directions depending on the disciplinary background of those engaging in a discussion [Cas1].

As Figure 2.1 suggests (based on academic curricular reports), hardware and software occur in different forms. Computing hardware is primarily the domain of computer engineering, often with close links to electrical engineering. The disciplines with the strongest focus on software development are computer science and software engineering; computer science is the foundational discipline with an emphasis on discovery related to programming, algorithms, and data structures, whereas software engineering addresses more applied concerns regarding the processes and actions needed for designing reliable, secure, and high-quality software systems**.** Information technology and information systems focus on organizational needs and uses for computing from infrastructural and information/organizational process perspectives, respectively.

2.3: Status of Computing Discipline Reports

This section briefly characterizes seven computing disciplines for which the ACM and IEEE-CS together with AIS have been developing as undergraduate curriculum recommendations over the past decade. The seven areas include computer engineering, computer science, cybersecurity, information systems, information technology, software engineering, and data science (in progress.) This section describes the disciplines with a focus on their educational programs.

2.3.1: Computer Engineering

Computer engineering (CE) brings together computing and electrical engineering in a way that embodies the science and technology of design, construction, implementation, and maintenance of software and hardware components of modern computing systems, computer-controlled equipment, and networks of intelligent devices. CE is the computing discipline that explicitly focuses on the development of hardware and software interface as a hardware embedded element of a computing system. The *Computer Engineering Curricula 2016 Report*, known also as CE2016, represents curriculum guidelines for undergraduate degree programs in computer engineering [Acm06]. The goals of the effort include incorporating past and future development needs, supporting professionals responsible for teaching a range of degree programs in computer engineering worldwide.

The capabilities of CE graduates integrate aptitudes of electrical engineering, software engineering, and computer science with a heavy emphasis on mathematics required as a foundation. CE2016 is very clear about the fact that graduates from CE programs should have the ability to design computers, design computer-based systems, and design networks with additional specifications that design needs to exceed simple configuration and assembly. CE is specifically an engineering discipline where graduates must have a breadth of knowledge in mathematics and engineering sciences with a preparation for professional practice or graduate work in engineering. Many countries provide CE graduates the opportunity to become licensed professional engineers according to local governmental rules.

The computer engineering discipline enables graduates to analyze and design circuits, manage the design of computer hardware components, and develop networking hardware solutions. For students interested in gaining experiences in integrating computing capabilities directly with computing hardware, computer engineering could be an appropriate degree program choice. Computer engineering also provides an excellent preparation for the design and development of modern technologies that closely integrate the physical world with the world of the artificial.

2.3.2: Computer Science

The *Computer Science Curricula 2013* project had two directives in developing its subsequent report, known as CS2013 [Acm04]. They included (1) a review of Computing Curriculum 2001 and CS2008, and (2) to seek input from diverse audiences to broaden participation in computer science (CS). CS2013 also had several high-level themes that provided overarching guidance for the development of its report. These include embracing an outward-looking view of the discipline, size management of the curriculum, providing actual exemplars to identify and describe existing successful courses and curricula, and being responsive to institutional needs, goals, and resource constraints.

Because of its theoretical foundations, computer science is often viewed as a fundamental discipline. It is, however, at times erroneously equated with all of computing. This misconception is understandable given that the theoretical roots of computer science have emerged separately from the engineering tradition of computing's earliest days [Ted1 sec3.2]. While the physical sciences are fundamental and offer theoretical basis to engineering fields, none subsumes the other and each has a well understood distinct identity. Similarly, this document and its predecessors have successfully established independent identities relative to computer science.

CS continues to have a more theoretical focus among the other computing disciplines, and its connection with abstract mathematics is still strong. A CS degree alone typically does not provide expertise regarding a specific context applicable to computing. Instead, CS programs emphasize abstract computational capabilities. CS2013 identifies abstraction, complexity, and evolutionary change as recurring themes in computer science, while sharing a common resource, security, and concurrency as general principles. These principles are closely linked to proficiency in programming and software development which are very important in most CS programs. CS2013 allocates about 40% of its core hours to algorithms and complexity, programming languages, software development fundamentals, and software engineering.

2.3.3: Cybersecurity

Cybersecurity is a highly interdisciplinary field of study. Specific degree programs are often associated conceptually and practically with one of the established disciplines in a way that has a significant effect on the fundamental identity of the program. The *Cybersecurity Curricula 2017* report [Acm08], known also as CSEC2017, became public in 2017. The report recommends security in eight areas to include data, software, component, connection, system, human, organizational, and societal. The CSEC2017 mission was to develop comprehensive and flexible curricular guidance in cybersecurity education that would support future program development and to produce a curricular volume that structures the cybersecurity discipline and provides guidance to institutions seeking to develop or modify a broad range of programs.

The report explicitly states that there is a broad spectrum of cybersecurity jobs from technical (such as cryptography and network defense) to managerial (such as policy and regulatory compliance) positions. At the same time, it also recognizes that every graduate of a cybersecurity program requires both technical skills and business acumen, essentially a managerial understanding of the organizational actions needed to ensure system-level security. A degree in cybersecurity prepares graduates for a broad range of application areas, including public policy, procurement, operations management, risk management, research, software development, IT security operations, and enterprise architecture.

The need for the specialized abilities that cybersecurity graduates have occurs almost daily. Continuous challenges of various types face organizations around the world who must secure data regarding their customers. Solutions that secure organizational data are multidimensional ranging from highly technical to organizational policies and societal legal and regulatory responses, creating a significant need for professionals with a broad range of specialized security expertise combined with the generic individual foundational abilities (such as problem solving, critical thinking, oral and written communication, teamwork, negotiation) that all computing professionals need.

Activities related to cybersecurity education have existed for some time. For example, in the United States, the National Security Agency Center of Academic Excellence program has been active for fifteen years [Nsa1], academic conferences associated with cybersecurity and education have been held for at least a decade, and accreditation bodies such as ABET [Abe1] have recently established cybersecurity accreditation criteria.

2.3.4: Information Systems

As the name suggests, the discipline of information systems (IS) focuses on information (i.e. data in a specific context) together with information capturing, storage, processing and analysis/interpretation in ways that supports decision making. The IS field also deals with building information processing into organizational procedures and systems that enable processes as permanent, ongoing capabilities. The discipline emphasizes the importance of building systems solutions, preferably so that they can be continuously improved. At the same time, IS recognizes that in terms of many of the technical computing knowledge areas and skills, it relies on knowledge developed by other computing disciplines.

The *Curriculum Guidelines for Undergraduate Degree Programs in Information Systems 2010* report is also known as IS2010 [Acm03]. The IS discipline is also preparing new curriculum guidelines to be completed in 2020 or 2021. The new IS report will emphasize that information systems as a discipline can make significant contributions to several domains, including business, and that its core areas of expertise are highly valuable or essential for the best practices within these domains. The IS discipline focuses on the ability of computing to enable transformative change within domains of human activity, sometimes called IS environments. That is, IS addresses the ongoing and innovative use of computing technologies to enable human activities to achieve their goals in ways that are better, faster, cheaper, less painful, cleaner, or more effective.

Degree programs in information systems always include coursework and other educational experiences in computing and information technology together with the coverage of an IS environment such as business. IS fosters foundational professional abilities that are important for all computing disciplines. Given the role of information systems as a bridge builder and integrator, communication and leadership skills have even more weight than in the context of the other computing disciplines. In the context of analytics, IS focuses on the integration of analytics into organizational systems.

2.3.5: Information Technology

Information technology emphasizes the central role of user needs. The *Information Technology Curricula 2017* report, known also as IT2017, is globally relevant and informed by educational research [Acm07]. Its task group sought to balance perspectives from educators, practitioners, and information technology (IT) professionals. The IT2017 report took a futuristic approach to curricular recommendation and proposed a learner-center framework for programs that prepare successful IT graduates for professional careers or further their academic study. It eliminated all notions of topics and learning outcomes, often represented by long lists of knowledge activities. Instead, the task group developed the use of competencies defined as a combination of knowledge, technical skills, and (human) dispositions. The IT task group followed pedagogical research and practice similar to what takes place in medical schools.

Degree programs in information technology started to appear in the 1990s. They were a precursor to the discipline that emerged in the 2000s through the development of the IT2008 curriculum recommendation and accreditation criteria. IT is a response to the need for professionals with the capability to develop, acquire, maintain and support the increasingly complex computing technology requirements of modern organizations. Information technology is "the study of systemic approaches to select, develop, apply, integrate, and administer secure computing technologies to enable users to accomplish their personal, organizational, and societal goals" [Acm07 p18]. For IT, the primary focus is on technology, closely aligned with user goals.

In the IT graduate profile specification, the focus is on analysis of problems and user needs, specification of computing requirements, and design of computing-based solutions. As general professional capabilities, communication, the ability to make ethically informed judgments, and the ability to function effectively as a team member augment this set. Of the currently identified computing disciplines, IT deals most directly with specific, concrete technology components in an organizational context.

2.3.6: Software Engineering

Software engineering is an engineering discipline that focuses on the development and use of rigorous methods for designing and constructing software artifacts that will reliably perform specified tasks. The term "software engineer" used to denote a profession is much more broadly employed than "software engineering" as an academic discipline or a degree program. There are many more individuals with a job title or professional identity of a "software engineer" than those who have graduated from software engineering programs. Adding to the confusion, software engineering or software development is often a part of computer engineering and computer science programs.

The purpose of the *Software Engineering 2014: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering* report, known also as SE2014, is to provide guidance to academic institutions and accreditation agencies about what should constitute undergraduate software engineering (SE) education [Acm05]. The SE2014 report identified a set of student outcomes describing the qualities of a SE graduate. These include professional knowledge, technical knowledge, teamwork, end-user awareness, design solutions in context, performance trade-offs, and continuing professional development. Similarly, the report presented a list of principles "that embraces both general computing principles as well as those that reflect the special nature of software engineering and that differentiate it from other computing disciplines."

Even though SE focuses on creating software-based solutions, it is much more than programming. SE emphasizes the use of appropriate software development practices and the integration of engineering rigor with the ability to apply advanced algorithms and data structures developed in computer science. Software engineering has strong focus on the design of reliable, trustworthy, secure, and usable software systems. The capabilities of trained software engineers often apply to large-scale systems with high reliability and security requirements such as complex manufacturing systems, industrial applications, business critical systems, medical devices, autonomous transportation systems, and military solutions.

2.3.7: Data Science (Under Development)

Data science (DS) is a new area of computing that is closely related to the fields of data analytics and data engineering. One definition of data science is "a set of fundamental principles that guide the extraction of knowledge from data … [and] involves principles, processes, and techniques for understanding phenomena via the (automated) analysis of data" [Pro1].

Several DS projects have emerged in recent years. These include the *EDISON Data Science Framework (2017)* project [Edi1], the *National Academies Report on Data Science for Undergraduates (2018)* [Nas1], the *Park City Report (2017)* [Par1], the *Business Higher Education Framework (BHEF) Data Science and Analytics (DSA) Competency Map (2016)* [Bhe1], and the *Business Analytics Curriculum for Undergraduate Majors (2015)* [Ban1]. ACM conducted initial DS workshops in 2015; a report described the discussions, reflected the diversity of opinions, and proposed a list of knowledge areas useful for the field [Cas1]. In August 2017, the ACM Education Council created a task force to articulate the role of computing in the DS field [Dat1]. The task force produced an initial draft report tentatively tagged as (DS202x) in February of 2019 [Dat2] followed by a second draft report in December of 2019 [Dat3].

The second draft describes a "competency framework" that addresses knowledge areas representing a body of material for data science degree programs that capture high-level competencies, skills, and dispositions. The knowledge areas include (a) computing fundamentals, (b) data acquirement and governance, (c) data management, storage, and retrieval, (d) data privacy, security, and integrity, (e) machine learning, (f) data mining, (g) big data, (h) analysis and presentation, and (i) professionalism. For a full curriculum, these areas need augmentation with courses covering calculus, discrete structures, probability theory, elementary statistics, advanced topics in statistics, and linear algebra.

2.4: Extensions of Computing Disciplines

Computing is much more than any of the individual disciplines alone. For a student of any one of these seven computing disciplines, it is useful to be aware of what the other disciplines offer, particularly in their areas of specific strength. All computing disciplines emphasize required professional knowhow of individual practitioners, including problem solving, critical thinking, communication, and teamwork. These professional capabilities bring computing disciplines closer together instead of separating them.

2.4.1: Computing Interrelationships

The discussion in Section 2.3 demonstrates that clear differences exist between the computing disciplines and that they all have distinguishing characteristics that are essential for their identity: CE is the only discipline that focuses on integration of hardware, software, and signal processing that are essential in areas such as cyber-physical systems, data communication, or medical imaging. CS has a strong and specific focus on developing strong conceptual foundations and computational capabilities. CSEC explores questions of safety, security, and continuity across the entire computing landscape. IS focuses on discovering and implementing positive organizational change using computing capabilities with a special emphasis on value generated by information. IT emphasizes building and maintaining organizational computing infrastructure capabilities and user support. SE addresses large-scale software development processes, particularly in safety and security critical areas. DS addresses large-scale data management, storage, and retrieval founded in mathematics and statistics.

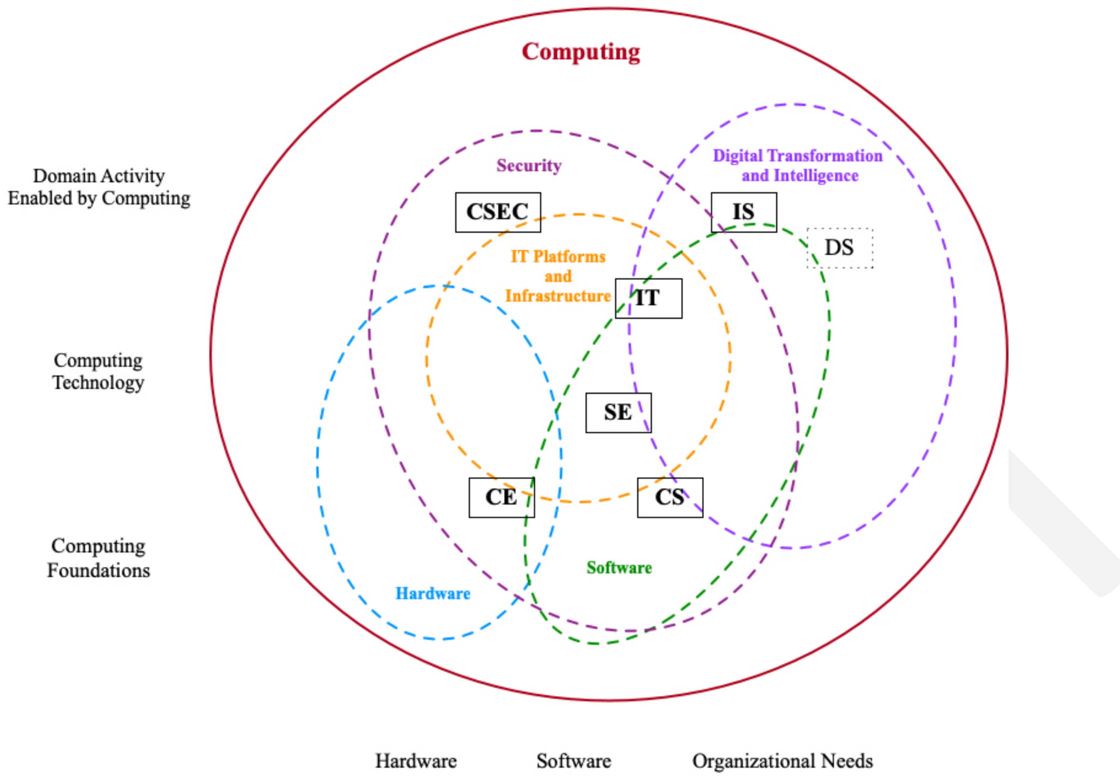


Figure 2.2. A contemporary view of the landscape of computing education
<u>Legend</u>: Curricular reports: CE=computer engineering; CS=computer science; CSEC=cybersecurity; IS=information systems; IT=information technology; SE=software engineering; DS=data science (under development)

Figure 2.2 illustrates three levels (foundations, technology, domain activity) of computing as related to hardware, software, and organizational needs. The internal regions are dotted because they are not absolute. Information technology platforms and infrastructure capture the integration of hardware and software into technology solutions that enable computing-based solutions having capabilities associated with data storage, processing, artificial intelligence, and visualization. Computer engineering, computer science, and software engineering provide the components required for these computing technology capabilities to exist. Information technology focuses on making and keeping them available for individual and organizational users. The area of digital intelligence and transformation covers the capture, management, and analysis of data enabling individuals, organizations, and societies to conduct their activities in a way that helps them achieve their goals better. The fields of information systems (and data science) enable digital intelligence and transformation. Security permeates the entire space of computing. These are the processes through which organizations change using computing capabilities.

2.4.1: Emerging Curricula

Computing curricula in different forms offer a rich variety of fields that continue to expand rapidly. Consequently, the number of educational fields that focus on the intersection of a specific scientific or business domain continues to grow. One of the more interesting but also the most complex of the emerging new computing-related disciplines is artificial and augmented intelligence (AI). The roots of AI go back to the 1950s, and these areas of computing have blossomed during the last ten years. AI and its allied field of robotics have become highly popular fields of study in computing. Although at the time of this writing no formal professionally endorsed AI curriculum exists, a curricular recommendation in these areas has the potential to emerge in the next few years.

Current curricular areas that have emerged in recent times include cloud computing, smart cities, sustainability, parallel computing, internet of things, and edge computing. Additionally, the predicted top-ten emerging computing trends are (a) deep learning (DL) and machine learning (ML), (b) digital currencies, (c) blockchain, (d) industrial IoT, (e) robotics, (f) assisted transportation, (g) assisted/augmented reality and virtual reality (AR/VR), (h) ethics, laws, and policies for privacy, security, and liability, (i) accelerators and 3D, and (j) cybersecurity and AI [Iee1]. All these areas have some coverage within existing curricula guidelines, and some areas (e.g., cybersecurity) even have its own formal guidelines. Other areas include 3D graphics and accelerators. One can only guess whether these top-ten trends will still be viable over the next dozen years. Section 7.6 surveys some of the current and emerging technology trends.

2.4.2: Computing + X

A growing interest in recent times is the development of computing programs (e.g., software engineering or information systems) that have an extension to a non-computing discipline (e.g., avionics or finance). This is referred to "Computing + X" where 'Computing' represents one of the computing disciplines and 'X' is a non-computing discipline. This mode of learning has the goal of integrating a non-computing area of study as an extension of a computing area. For example, if X is linguistics, then CE + X represents a linguistic extension of a computer engineering program. Such programs allow students to pursue their computing interests in other academic fields. It also allows computing students to pursue flexible programs of study that incorporate a strong grounding in a computing discipline with technical or professional exposure in other fields.

The relatively recent initiatives surrounding Computing + X are nothing new. For decades, computing programs had offered tracks, concentrations, or minors in a variety of subject areas to expand the knowledge base of students for computing programs. These programs continue today. However, the level of interest in the longtime practice has increased. So, the Computing + X phenomenon continues where X could be in areas such as astronomy, chemistry, economics, languages, linguistics, music, and other computing extensions. Computing + X allows students to discover transformational relationships between computing and non-computing fields. Degrees in this category often have the term 'informatics' included in them such as medical informatics, health informatics, legal informatics, bioinformatics, or chemical informatics. In many ways, information systems was the original "Computing + X" discipline, integrating computing primarily with business to transform the way businesses and other enterprises operate.

2.4.3: X + Computing

Computing is ubiquitous with application areas in almost every field imaginable. Therefore, the study of computing in other disciplines arises naturally. That is, computing becomes an extension to an established discipline of study. This representation is "X + Computing" where 'X' is the established non-computing domain usually in science, business, or humanities. For example, a program in computational biology would have its roots in some aspect of biology augmented by a study in computing related to it. Computational finance is another example where computing becomes an extension of finance.

As before, for decades, non-computing programs had offered tracks, concentrations, or minors in a variety of subject areas to expand the knowledge base of students from non-computing programs. The "X + Computing" practice continues today where 'X' could be principal interest areas such as accounting, biology, art, or other computing extensions. "X + Computing" allows non-computing students to discover transformational relationships between their principal area of study and computing. Hence, "X + Computing" is different from "Computing + X" because in the former, the base area is a non-computing discipline (e.g., chemistry) while in the latter, the base area is a computing discipline (e.g., computer engineering).

Whether it is X + Computing or Computing + X, both designations reflect the impact computing has had across a broad range of other, non-computing domains. Not surprisingly, in the German-speaking world, the term business informatics (Wirtschaftsinformatik) is used for degree programs that are globally aligned with those in information systems [Hel1]. In all extended degree programs, the fundamental question is the same: how do computing and computational thinking transform the way in which those working within non-computing domain X achieve their goals? In practice, this mode of thinking requires three types of abilities: domain, computing, and integrative related to the transformative opportunities offered by the computing field. Beyond graduation, graduates may benefit from interdisciplinary studies and lifelong learning.

There are many similar examples to represent other fields with computing. For example, high-performance computing (HPC) field is a representative field that is interdisciplinary. The students working for the HPC field not only need computing field knowledge, but also domain-specific knowledge as well as supercomputer system maintenance knowledge.

2.4.4: Other Tertiary Computing Models

In addition to the traditional three- or four-year baccalaureate program models, there are thousands of academic institutions around the world offering non-traditional educational programs in computing. Two-year colleges in the United States are one example. According to ACM's Educational Policy Committee's "Lighting the Path" report of 2018, 41% of all U.S. post-secondary students attend community colleges and the numbers are somewhat higher for most minority populations [Acm16]. Currently, ACM offers curriculum recommendations for two-year programs in information technology [Acm09] and computer science [Acm10]. The ACM CCECC's curricular guidance for two-year cybersecurity programs was recently published and endorsed by the ACM Education Board [Acm17].

Education in computing has been a shared interest of traditional universities and other academic institutions, other education providers, and employers. Even though specific research-based data are difficult to find, the number of people employed in computing-related jobs is much higher than those with an actual degree in computing [Nas2, p7]. A recent national academies report [Nas2, App C] refers to transfers from other fields of study and immigration as sources of employees to fill the gap in the context of computer science. The report does not, however, discuss the other broadly defined computing-related job roles, but anecdotally it is common knowledge that many individuals with an educational background in other fields (or those without a completed higher education degree) serve successfully in computing-related positions (e.g., IBM's "New Collar" jobs).

There are at least five commonly used pathways where a person with an academic background in a different field gains the proficiencies needed to perform well in a computing-related field. These are as follows [Dab1, Per3, Wag1].

- Self-study without any formal educational sources.
- Self-study using extensive computing education resources available online for free or at a very low cost, including providers such as various open universities, Udemy, EdX, Coursera, Khan Academy, and SkillShare.
- Coding bootcamps that are typically 8-12 weeks in length, intensive, and solely focused on providing their participants with software development and related skills that provide students with immediate employability.
- Specialized schools in coding, software engineering, or other related schools. Examples of this category are École 42 in Paris, 42 Silicon Valley, and the recently launched Hive Helsinki. Typical descriptions of these institutions are that they operate very closely with industry partners and declare themselves to be operating without teachers and without courses and classes.
- Computing diploma and masters level programs that offer conversion courses for existing graduates from non-computing disciplines.

There are many ways to transition into the computing field and to become successful in doing so. Cisco, CompTIA, and Microsoft offer certifications to achieve this goal [Cis1, Com3, Mic1]. Additionally, some universities offer tandem pathways to graduate school, and governments offer security certifications for professional enhancement.

2.4.5: Computing in Primary and Secondary Education

The computing education community around the world has done extensive work to improve the availability and quality of computing-related courses in primary and secondary education, with a specific focus on improving the diversity of students selecting computing as their career. Some examples of influential actors in this area include the following activities.

In the United States, *Computer Science Principles* is a year-long course offered in high schools that introduces students to the foundational concepts of computer science and challenges them to explore how computing and technology can impact the world. It is a rigorous, engaging, and approachable course that explores many of the foundational ideas of computing, so all students understand how these concepts are transforming the world in which we live. *Code.org* is an approved Advanced Placement (AP) CS Principles provider and is a not-for-profit organization founded by Hadi Partovi that focuses on "expanding access to computer science in schools and increasing participation by women and underrepresented minorities." Among other activities, code.org organizes an annual "Hour of Code" event with millions of participants annually, offers a library of computer science courses for primary and secondary education, and advocates computer science education with policy makers, primarily in the United States. Furthermore, *CSforAll* is a hub for the national "Computer Science for All" movement, which works to enable all students in grades kindergarten through twelfth year (K-12) to achieve computer science literacy as an integral part of their educational experience. It has currently 355 member organizations, including content providers, education associations, and both companies and non-profits as funders.

The *Computer Science Teachers Association* (CSTA) is a membership organization for primary and secondary education teachers in computer science with more than 25,000 members in 145 countries. CSTA's mission is to "empower, engage and advocate for K-12 CS teachers worldwide." ACM established it in 2004.

The importance of how teachers are being educated should not be overlooked. In Europe, the importance of a general education in computing (i.e. informatics) has been recognized. Digital literacy, computational thinking, and other informatics related competences are important for pre-university students, especially because it generates interest and understanding of what computing really is.

CSpathshala is an ACM India education initiative to bring a modern computing curriculum to Indian schools [Csp1]. Launched in 2016, CSpathshala has developed an unplugged curriculum to teach computational thinking (CT) without the use of computers along with sample teaching aids for the first eight years of school in India. More than 300,000 students largely from rural government schools in India are learning computational thinking using the CSpathshala curriculum. The draft "National Education Policy 2019," recently released by the Indian government, also recognizes CT as a fundamental skill and recommends teaching CT from age six using well-designed worksheets.

Similar initiatives include those in Finland, New Zealand, Sweden, the United Kingdom, and Europe overall [Cas2, Fra1, Ins1, Roy1]. Computing in the Middle East is described in section 6.3.7.

2.4.6: Computing Specializations

Many specializations exist in computing. One area that goes back to the 1940s is scientific computing—considered to consist of algorithms and the associated methods for computing discrete approximations used to solving problems involving continuous mathematics. Numerical methods and computational science are other names used for this area which deals with mathematical models to solve problems, methods for system optimization, and computing infrastructure in support of engineering and science problems.

Another such area is digital game design and development (DGDD). Another is media development. In the United States alone, more than five hundred DGDD programs currently exist [Are1], and many more exist worldwide. Curricular efforts in game and media programs are ongoing. The game and media industries develop specialized hardware and software that are now being utilized in higher education. The industry stands at about $43.4B in the United States alone [The1, Dea1] thereby making this emerging area a worldwide phenomenon.

In the future, the world should expect to see increasing specialization on the development of core computational capabilities within computer science and software engineering (software), and especially in computer engineering (integration of hardware and software). The number of types of computing degree programs should also increase dramatically that focus on the transformation of computing programs into specific domains of human activity (e.g., information systems and Computing + X) as well as a greater integration of computing in existing domains or other disciplines (e.g., X + Computing). The world should also witness more degree program types for specialized pervasive themes with a broad ranging effect across multiple domains (e.g., cybersecurity, data analytics, artificial intelligence), as well as continued contributions of degree programs that prepare professionals for roles focused on organizational computing infrastructure (e.g., information technology).

2.5: Digest of Chapter 2

This chapter examines the continuing evolution of computing education. In the context of undergraduate programs, computing can refer to a family of study areas corresponding to the discipline reports for computer engineering, computer science, cybersecurity, data science, information systems, information technology, and software engineering degrees that have been developed by the ACM and IEEE-CS with AIS in recent decades. On the other hand, the changing landscape of computing has now led to the recognition of the importance of information security and data as a resource for decision making. Within the computing education landscape, there exists a rich variety of fields that continue to broaden, including opportunities for Computing + X and X + Computing degrees, and tertiary models for computing programs. Around the world, computing education has also expanded into primary and secondary schools. At the same time, specialization areas such as scientific computing or digital game design have led to new degree programs, a trend that should continue.

Chapter 3: Knowledge-based Computing Education

The philosophical underpinning of the CC2020 project treats computing as a meta-discipline – a collection of disciplines having a central focus of computing. This chapter explains the concept of knowledge-based learning and how it has encompassed computing education over decades. It reviews the CC2005 report which is primarily a knowledge-based document. Additionally, it addresses how workplace and employment dynamics affect knowledge-based learning and related issues.

3.1: Knowledge-Based Learning

This section addresses some of the underpinnings of knowledge-based learning (KBL). It explores the definitions of learning and knowledge, the attributes of KBL, and the relationship between KBL and computing curricula.

3.1.1: Learning and Knowledge

Before discussing knowledge-based learning, it is useful to first understand the contextual meaning of learning and knowledge. The word *learning* refers to the endeavor of "knowledge or skill acquired by instruction or study" [Mer3] often in an environment conducive to the activity. In turn, the word *knowledge* refers to the "acquaintance with or understanding of a science, art, or technique" [Mer4].

There is an inextricable connection between the two words knowledge and learning. The former refers to content while the latter refers to activity. Thus, people acquire content and skills through the process of learning. Humans acquire (learn) content (knowledge) continuously, almost from the time of birth. For the purposes of this report, content acquisition refers to learning in formal settings or structures such as in classrooms or online environments.

Recently, the term *content knowledge* has come into use, which refers to the body of knowledge and information that teachers teach and that students should learn in a subject or content area. Content knowledge generally refers to the facts, concepts, theories, and principles taught and learned in specific academic courses [Edg1]. This form of knowledge occurs in core courses of study, curriculum, or learning standards.

3.1.2: Learning from Knowledge Contexts

In general, learning occurs by building on knowledge a person already has. That is, a person, namely a student, scaffolds new knowledge based on the student's existing knowledge. *Knowledge-based learning* (KBL) depicts this form of learning activity. More formally, "knowledge-based learning is learning that revolves around both the knowledge that the student already has, and the understanding that they are going to achieve by doing work" [Tes1].

Students, teachers, and the public have all experienced knowledge-based learning. Basic schooling allows advancement from one grade level to the next based on the verified knowledge acquired in one grade before advancement takes place. Often, verification takes place by evaluating students' knowledge content through tests, oral or written examinations, interviews, and other tools useful in assessing whether a student has achieved the expected knowledge base for a given level. At universities, course prerequisites attempt to ensure that a student has the necessary knowledge needed to advance to the next course level.

Knowledge-based learning has existed for millennia. Whether formally or informally, KBL has used approaches to elevate human knowledge on a global scale. Teachers deliver information to learners and then check their level of attainment. Reflective learners can assess themselves on the acquired new knowledge. Teachers direct learners on what they need to know and check whether they learned it. Using this approach and providing reliable comment, teachers can help students see where they have learned or where they have erred.

Knowledge-based learning enjoys many benefits. It builds on learners' existing knowledge; it helps learners see how they are progressing, and it helps them highlight gaps in their knowledge. With clear learning objectives, learners can see how their existing knowledge will help them to complete the task [Dso1]. Practicing knowledge-based techniques can identify where learners require more emphasis. By building on the knowledge a person already has, KBL lifts learners' confidence by showing them that they have the knowledge they need to finish a task [Icd1].

3.1.3: Knowledge and Computing Education

For computing and other disciplines, *knowledge* has always been the focus of the study area. Computing curricular reports often describe a discipline through knowledge areas (KAs), knowledge units (KUs), and learning outcomes (LOs). Sometimes, people refer to this structure as the "*knowledge area, knowledge unit, learning outcome*" (KA-KU-LO) model with lists of topics associated with each knowledge unit. These curricular reports generally do not provide guidance related to skills or guidance related to human behavior particularly reflected by performance in the workplace.

These documents reflected the KBL concept, viewed traditionally as a form of learning that involves knowledge students learned and already have, together with the understanding that they are going to achieve through work [Cla1]. That is, teachers transfer knowledge to students through experience, notes, textbooks or other means; having received the knowledge, students have an expectation of achievement because of it and work toward demonstrating that achievement. Almost all universities worldwide produce graduates through knowledge-based learning.

However, the traditional knowledge-based learning paradigm may be insufficient by itself to address all of the challenges in educating for the future. Technology now influences new ways of learning. Students use many non-traditional learning formats, thereby challenging traditional methods. Furthermore, universities produce computing graduates who may be intellectually smart, but have difficulties in workplace settings. Learning in computing education needs to incorporate knowledge along with other attributes.

3.2: Revisiting Computing Curricula 2005

The CC2005 report provided readers with an overview of five undergraduate computing degree programs that were available in the early 2000s. At that time, five computing curricula reports were in existence, which included computer engineering (CE2004), computer science (CS2001), information systems (IS2002), information technology (a work in progress that was later published as IT2008), and software engineering (SE2004). These computing fields were related but quite different from each other.

The CC2005 report explained the character of the various undergraduate degree programs in computing and assisted people to determine which programs best suited their goals and circumstances. Beneficiaries of the report included recruiters from industry and government, students and potential students, university faculty members and administrators who were developing plans and curricula for computing-related programs at their institutions. In addition, beneficiaries included those interested in accreditation of computing programs, and responsible parties in public education including boards of education, government officials, elected representatives, and others who seek to represent the public interest.

3.2.1: Intent of CC2005

Reliable information about the number of different types of computing degree programs was difficult to ascertain in the early 2000s. Hence, the focus on just five prominent computing programs (CE, CS, IS, IT, SE) satisfied the committee's criterion for proper inclusion to distinguish undergraduate curricular guidelines. These five computing areas represented the majority of all undergraduates specializing or majoring in computing. Notwithstanding, at that time, the committee expected additional computing disciplines to generate curricular expansion as extensions to the report. Candidate programs for future editions could include new fields that did not yet exist or established fields that did not have generally accepted curricular guidelines. In the end, each one of the five discipline-specific curricular volumes reflected in the CC2005 report represented the best judgment of the relevant professional, scientific, and educational associations and served as a definition of what these five computing degree programs should be.

The CC2005 committee made no effort to update the contents of existing curricular reports as that effort was beyond its mission and authority. Rather, the committee had reviewed the five curricular volumes, compared their contents to one another, and synthesized what they believed to be essential descriptive and comparative information. In addition to using the five curricular reports as the basis for the CC2005 report, the committee had referred to the computing professional organizations and other supporting information as necessary. The committee did not focus on other types of undergraduate computing degree programs (e.g., associate degree or similar programs), graduate education in computing, computing research communities, or nontraditional computing education such as vendor-specific certification programs. Additionally, the CC2005 committee realized that computing itself will continue to evolve and new computing-related fields would likely emerge.

3.2.2: Content of CC2005

The most significant part of the CC2005 report is the definitive articulation of the five computing disciplines. As mentioned earlier, these disciplines included computer engineering, computer science, information systems, information technology and software engineering. In addition to addressing the landscape of computing in the early 2000s, the report defined the meaning of "computing" and provided a brief history of the evolution of computing before, during, and after the 1990s as shown earlier in Figure 2.1 It then described (and defined) each of the five computing disciplines followed by graphical visuals for these five disciplines. Discussion of the graphical visuals occurs later in this chapter.

One of the useful aspects of CC2005 was the discussion on the expectations of graduates for the degree programs. The discussion revolved on two themes. One theme dealt with curricular summaries as a comparison of degree programs with an interpretation of the tabular representation and suggestions on its use. The other theme focused on expected degree outcomes with an expected comparison of degree graduates. Both these tabular representations are useful elements to contrast the outcomes of the five computing degrees.

The CC2005 report also acknowledged the rapid pace of change in academia and how computing might affect the offered degrees, specifically in the five focus degree areas of computer engineering, computer science, information systems, information technology, and software engineering. The pace of change particularly reflects the changes in the workplace where change is continuous. Because of this change, computing degree programs should be responsive to such variations.

Additionally, the CC2005 report addressed institutional considerations such as the evolution of degree programs and strategies to monitor course portfolios. It addressed diversity challenges, faculty development, adaptation, as well as organizational and curricular structures. Coupled with curricular response to market forces and academic integrity, the report discussed aspects of quality assurance and program accreditation as it exists in the United States and the United Kingdom. Regardless of the metrics used, all would agree that program quality should be paramount for all computing disciplines.

3.2.3: Comparison Tables

CC2005 provided a comparative view of the emphasis on computing topics among the five types of computing degree programs. A comparison table provided a summary of the topics studied at the undergraduate level in one or more of the computing disciplines, presented in its first column. The remaining columns showed the numerical values per topic for each of the five types of computing degree programs. These values range between 0 (lowest) and 5 (highest) and they represent the expected relative emphasis each type of computing degree program might place on each given topic.

In addition to this comparison table, the CC2005 report provided a table showing the relative performance capabilities of computing graduates by discipline [Acm02 p28 Tab3.3]. This table focused on outputs, summarizing the relative capability expectations of computing graduates. Table 3.1 shows an excerpt of that table.

Table 3.1. Computing Graduate Profiles (excerpt from [Acm02 p28 Tab3.3])

Area	Performance Capability	CE	CS	IS	IT	SE
Algorithms	Prove theoretical results	3	5	1	0	3
	Develop solutions to programming problems	3	5	1	1	3
	Develop proof-of-concept programs	3	5	3	1	3
	Determine if faster solutions possible	3	5	1	1	3
Application programs	Design a word processor program	3	4	1	0	4
	Use word processor features well	3	3	5	5	3
	Train and support word processor users	2	2	4	5	2
	Design a spreadsheet program (e.g., Excel)	3	4	1	0	4
	Use spreadsheet features well	2	2	5	5	3
	Train and support spreadsheet users	2	2	4	5	2
Computer programming	Do small-scale programming	5	5	3	3	5
	Do large-scale programming	3	4	2	2	5
	Do systems programming	4	4	1	1	4

3.2.4: Curricular Visuals

One highlight of the CC2005 report consists of the two-dimensional visual graphics that depicted the five computing disciplines. These graphics illustrated the commonalities and differences among computing disciplines. Their dimensions highlighted the relative degree to which a computing discipline focused on theory versus practice; it also highlighted the degree to which a computing discipline focused on hardware versus humans. They suggested how each discipline occupies the problem space framework of computing as shown in Figure 3.1. The focus is on what students in each of the disciplines typically do after graduation, not on all topics a student might study. Some individuals will have career roles that go beyond the scenarios described by these snapshots.

The horizontal range runs from theory, principles, and innovation on the left, to application, deployment, configuration on the right. Thus, someone who likes the idea of inventing new things or enjoys a university setting to develop new principles will want to work in a discipline that occupies the space to the left. Conversely, someone who wants to help people choose and use appropriate technology or who wants to integrate off-the-shelf products to solve organizational problems will want an area that occupies space to the right. Because there are many kinds of job tasks that fall between the extremes, one should not just look only at the far left and far right but consider possibilities between the extremes.

The vertical axis runs from computer hardware and architecture at the bottom to organizational issues and information systems at the top. As we go up this axis, the focus changes from wires, hardware, chips, and circuits at the bottom to people, information, and organizational issues at the top. Thus, someone who likes designing circuits or is curious about a computer's inner workings will care about the lower portion of the diagram; someone who wants to see how technology can work for people or who is curious about how technology affects organizations will care about the upper portions. We can consider the horizontal and vertical dimensions together. Someone who cares about making things work for people and is more interested in devices than organizations will be interested in the lower right, while someone who wants to develop new

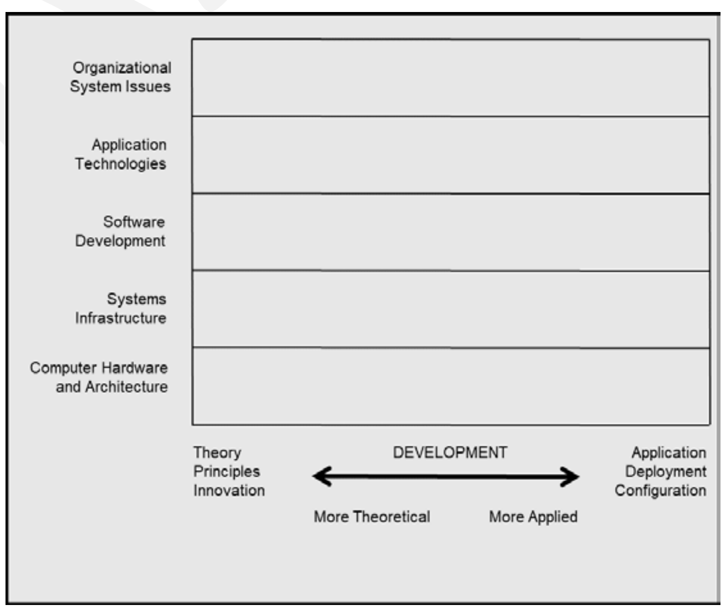


Figure 3.1. Problem Space Framework (CC2005)

theories about how information affects organizations will be interested in the upper-left area of the diagram.

Figure 3.2 provides curricular illustrations that sketches the conceptual territory occupied by each of the five computing disciplines. These are informal illustrations used to communicate the CC2005 task force's subjective interpretation of the various disciplines. They are not based on any precise quantitative foundation. Furthermore, they show only computing interests or themes. Computer engineering occupies a broad area across the bottom because computer engineering covers the range from theory and principles to the practical application of designing and implementing products using hardware and software. Computer science covers most of the vertical space between the extreme top and extreme bottom because computer science generally deals with theory and software development such as operating systems and web browsers. Information systems occupies the shaded area across most of the top-most level because it concerns the relationship between computing systems and the organizations they serve and often tailor application technologies to the needs of the enterprise. Information technology covers the shaded area along the top right edge because it focuses on the application, deployment, and configuration needs of organizations and people over a wide spectrum. Software engineering spans the entire horizontal dimension at the middle of the diagram because the subject covers a wide range of large-scale software applications with respect to the systematic development of software. The images from Figure 3.2 have received worldwide acclaim in computing educational circles. They have appeared in many contexts such as the poster shown in Appendix A.

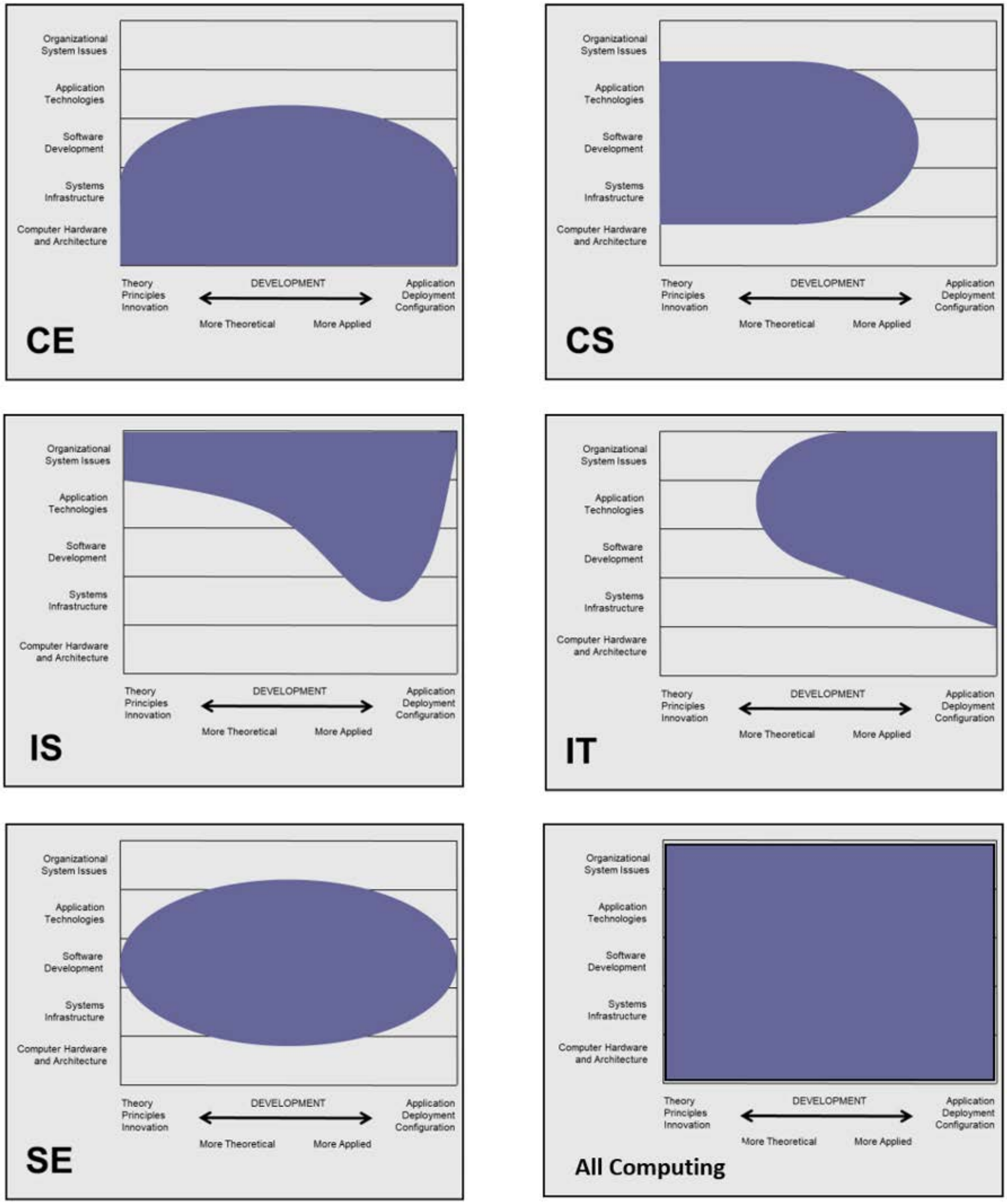


Figure 3.2 Visuals from CC2005

3.2.5: Global and Other Considerations

The CC2005 report and the associated five volumes of the Computing Curricula Series benefited to some degree from international input. Notwithstanding, the CC2005 task force was very conscious of the void of encompassing greater global contributions to its work. The task force recognized that future efforts must feature significantly expanded international participation. Some differences include the structure of the academic year, the emphasis given to the study of computing within a degree program, and quality control mechanisms such as different expectations and practices regarding accreditation. In addition, there were differences in approaches to defining the focus of degree programs and in terminology.

The CC2005 report addressed other issues useful for a global computing community. The CC2005 task force recognized that the computing field is evolving and as a result, it provided some suggestions for academia to keep in step with the pace of change. It also recognized that the pace of change that exists in the workplace whereby graduates of computing-degree programs should have the ability to fulfill their own career opportunities. In addition, the CC2005 report addressed institutional considerations and urged institutions to be mindful of the evolution of computing degree programs, to initiate strategic approaches to manage change, and to approach diversity through faculty adaptation and development as well as organizational and curricular structures.

The CC2005 report received universal acceptance as a document to differentiate computing degree programs at the time. Educators, students, and industry professionals are familiar with the illustrations shown in Figure 3.2 and the poster in Appendix A. Overall, the CC2005 project was a positive contribution to students, to industry, and to the computing academic communities.

3.3: Limitations of a Knowledge-Based View

CC2005 reflected a knowledge-based view of computing education. This view resulted in the ability to conceptualize specializations with respect to the types of knowledge that they contain, in models already shown in Figure 3.2. Such a view has been helpful in establishing the course structure of curricula within the various specializations, and it reflects the traditional model of education in that regard. In such a view, curricula reflect topics taught within a conglomeration of courses, but the skills learned within those courses depends heavily on the design of the individual course. Two wildly different curricula in terms of skills learned could both meet the same knowledge-based curricular requirements.

3.3.1: The Skills Gap

The variability stated above is a ubiquitous consequence of classical education in most disciplines. However, for jobs that require certain skills, it means that recent computing graduates may not have those skills even though they graduate from a curriculum that meets prescribed knowledge-based requirements. This has classically put the onus on industry to do training to meet the requirements of their workforce. However, with the fast-paced dynamics of change in the computing fields, industry is frequently no longer willing to train recent graduates of computing programs. Some industries and companies expect to have performance (and profit) almost immediately after hiring. People seeking computing careers will have a strong potential for success only if they possess relevant skills and appropriate temperament.

Currently there are still plenty of jobs in the computing industry, and this trend is expected to continue for the immediate future. For example, in the United States, a recent study by the Bureau of Labor Statistics (BLS) estimates that by 2024, computing employment in the United States will increase by 12% [Bls1], with information security leading by 36.5% [Bls2]. Employment growth for information security analysts projected for 2014-2024 is 18%. Other computing occupations have even larger projected growth: application software developers (19% across all industries, 31% within the computer industry), computer systems analysts (21% across all industries, 33% within the computer industry), and web developers (27% across all industries, 39% within the computer industry). Figure 3.3 presents these data.

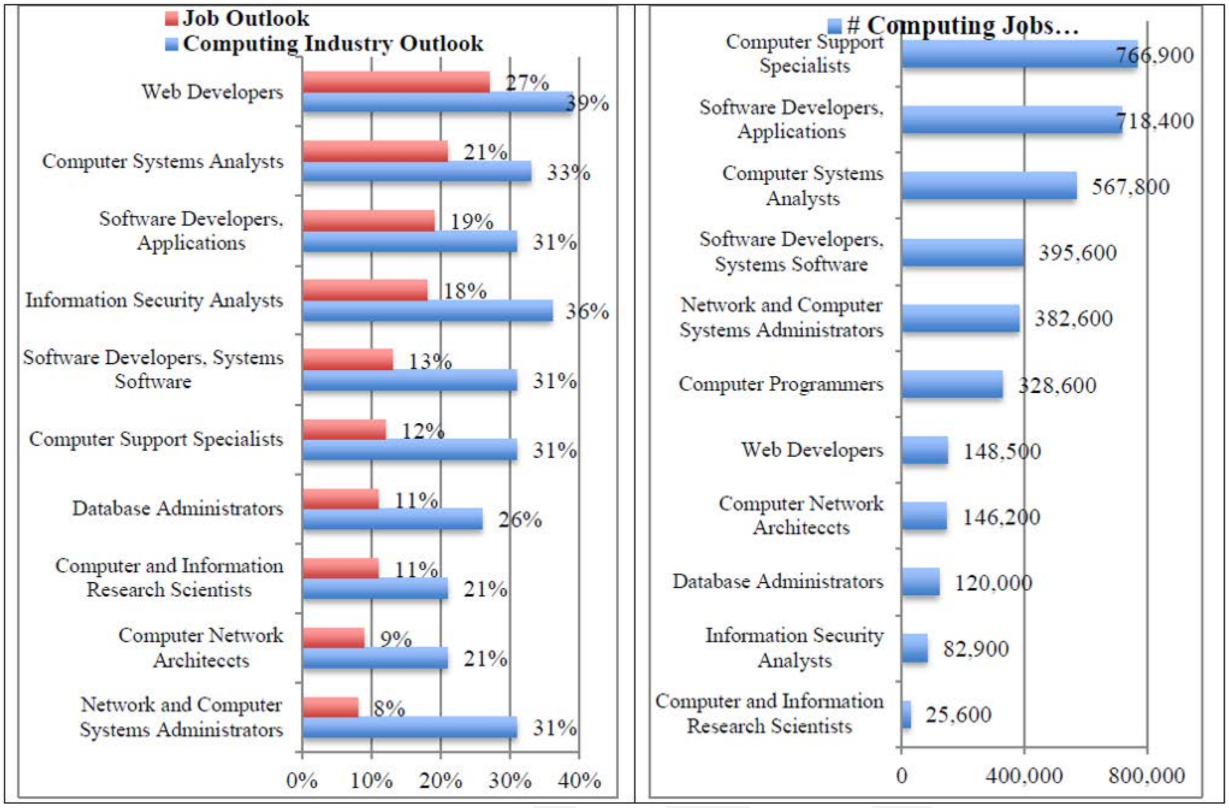


Figure 3.3. Left: Computing occupations projected growth 2014-2024 across all sectors (job outlook) and in the computing sector. Right: Computing jobs in 2014. (*Courtesy of Bureau of Labor Statistics*)

Thus, there are plenty of jobs available, but not every graduate has the skills and temperament to be successful. The gap between the skills of today's college graduates and the skills expected by employers is frequently known as a *skills gap*. The degree to which a skills gap exists in computing in different parts of the world would require analysis of labor and economic data that is beyond the scope of this report. Anecdotally, a recent survey by PSI Services found the following situation in the United States [Psi1].

- 80% of Americans (U.S.) agree there is a skills gap, and 35% say it affects them personally.
- 42.5% of recent graduates were underemployed as of March 2018, according to the Federal Reserve Bank of New York.
- $160 billion is the annual cost that researchers from the Centre for Economic Research calculated to be the total cost of the skills gap to US companies.
- 60% of U.S. employers have job openings that stay vacant for 12 weeks or longer. The average cost from job vacancies is at least $800,000 annually.
- 81% of employers indicated that prospective employees lack critical thinking and analytical reasoning skills. 75% think graduates lack adequate innovation and diversity skills.

Students who graduate from a university computing program might assume that the baccalaureate degree is a basic qualification to attain a position, and that those who have baccalaureate computing degrees will be easily employable in the computing field. The current high demand for computing professionals reinforces this idea. Yet the skills gap that exists for college graduates in general is arguably true for computing graduates in at least parts of the world today. And with the supply of computing graduates significantly lagging behind the demand for computing graduates, this skills gap simply exacerbates what is already a dire shortage.

Similar skill gap analyses can be performed in other parts of the world [Iee2]. Readers are encouraged to investigate their own situation.

3.3.2: Non-Degree Certifications

To address the skill challenge, industry market forces have non-degree certifications that have had some influence on academic institutions. Some of these certifications have even become part of some university computing curricula such as acquiring network certification in an academic networking class. Individuals who complete certification exams can use these credentials to supplement the value of their academic education to potential employers. Employees can use certifications to demonstrate their job readiness and pursuit of extra-curricular activities to demonstrate IT skills to potential employers. Table 3.3 lists some leading computing certifications for 2017 compiled by the CRN media outlet [Nov1]. The CC2020 public website will show a more inclusive collection of certifications on a global scale.

Table 3.3. Common Computing Certifications

Entry-level networking and security (CompTIA, Cisco)	Project management (Project Management Institute, Axelos)
Professional networking and routing and switching (Cisco, Citrix)	Security (ISC2)
	Security management (ISC2)
Virtualization and networking (Citrix VMWare)	Cloud computing (Amazon)
Windows servers and infrastructure (Microsoft)	Risk management (ISACA)
IT service management (Axelos)	IT auditing (ISACA)

3.3.3: Skills Frameworks

Non-degree certifications are an attempt to bridge the skills gap in computing. Industry has also utilized *skills frameworks*. Appendix B provides several summaries that address computing skills such as SFIA, e-CF, and iCD frameworks. Developed in consultation with professional societies, these skills frameworks are utilized as part of the hiring process to articulate expectations for specific types of jobs [Sfi1].

Industry can utilize skills frameworks to define its employment needs, and a combination of degrees and certifications as credentials to distinguish candidates for jobs. The question for baccalaureate computing education is: Is there a way to define curricular standards in a way that captures industry needs with greater fidelity? Generally speaking, industry and academia need to utilize a common language for defining outcomes and expectations. For academics to consider skills is a step in the right direction. A more complete answer is provided in Chapter 4.

3.4: Digest of Chapter 3

This chapter discusses the concepts associated with knowledge-based learning which has been the traditional focus of computing education. The CC2005 report was primarily an articulation of the five computing discipline reports that existed at that time. It provided a comparative view and visual representation of those disciplines and reflected a model of education that primarily aided in establishing the course structure of respective curricula. The skills that students learned within those courses then relied on the design of the individual courses. As noted in recent job reports, however, the traditional model of computing education can lead to computing graduates not having the skills and attributes needed to pursue computing careers.

Chapter 4: Competency-based Computing Education

While early work in defining model computing curricula was based on knowledge as presented in Chapter 3, more recent work has been transitioning toward a competency-based view of computing education. This trend has been an important pivot in defining curricular standards that codifies expectations that go beyond simply communicating knowledge. Within the broader context of industry, professions, and society, a curriculum description centered on competency focuses on an individual's capability to perform and to apply their computing education in a practical and professional service to society.

Adopting a coherent competency model to define computing curricula should promote and clearly describe the practical benefits of computing programs to its stakeholders: students, benefactors, faculty, administrators, employers, accreditors, lawmakers, and society. Describing computing competence in a practical context shifts the focus of curricula away from describing a body of knowledge in relation to a disciplinary area and channels it toward pragmatic student accomplishment and performance. Descriptions of what graduates can do in practical situations replace descriptions of content learning and memorization. Competency more effectively describes outcome expectations. It challenges educators to develop more proficient computing professionals, and it allows society to recognize the purpose and benefits of a computing education within a competency framework.

4.1: Competency and Competency-Based Learning

Competency is not a novel idea. The concept goes back centuries and millennia. The construction of the Giza Pyramids or the Roman Colosseum are examples of structures designed and engineered by competent professionals of the time. A general dictionary defines competency as "the quality or state of having sufficient knowledge, judgment, skill, or strength" [Mer1]. It is important to note that the use of this word always occurs in a context: being competent in law does not mean someone is competent in medicine.

4.1.1: Competency and its Meaning

The Harvard University Competency Dictionary [Har2] describes a useful overview of competency through the following definition and explanation.

> *Competencies, in the most general terms, are "things" that an individual must demonstrate to be effective in a job, role, function, task, or duty. These "things" include job-relevant behavior (what a person says or does that results in good or poor performance), motivation (how a person feels about a job, organization, or geographic location), and technical knowledge/skills (what a person knows/demonstrates regarding facts, technologies, a profession, procedures, a job, an organization, etc.). Competencies are identified through the study of jobs and roles.*

Thus, competency identifies closely with job-related behavior and performance. It is a person-centered concept that requires demonstration of human behavior together with technical skills and knowledge.

The CC2020 project has embraced competency as an underlying theme of its activities and as a principal component of this report. The task force believes that every career path in computing, whether industrial, academic, governmental, or any other career, is founded on competent performance. The project observes that knowledge is only one component of the idea of competency. While the working definition of computing competency may evolve, adopting the idea of competency as the foundational idea on which to base academic program design permits a stronger alignment between the product of an education and the needs of professional practice in the workplace. Thus, it is appropriate that competency should form the basis for expressing both the target of learning in computing education and the fitness to task in the workplace. This approach ensures that all graduates of computing programs have the preparation to be effective for specific career paths.

4.1.2: Previous Work on Computing Competency

In 2017, the Accreditation Committee of the European Quality Assurance Network for Informatics Education (EQANIE) published new program outcomes for accreditation of business informatics or information system or related programs in consultation with members and stakeholders. [Eqa1]. EQANIE describes program outcomes as "quality standards for knowledge, skills and competences that graduates of an accredited course should have achieved as the educational base for practicing their profession or for post-graduate studies." The European Commission's Digital Competence Framework 2.0 (DigComp 2.0) identified the key components of digital competence in five areas which can be summarized as 1) Information and data literacy, 2) Communication and collaboration, 3) Digital content creation, 4) Safety, and 5) Problem Solving [Eco1].

The IT2017 project was the first of the ACM/IEEE baccalaureate curriculum projects to embrace the concept of competency as the primary characteristic of curriculum definition. The arrival of the IT2017 report [Acm07] heralded a shift away from the *knowledge area, knowledge unit, learning outcome* mindset and redirected emphasis toward performance. The report stated that "competence refers to the performance standards associated with a profession or membership to a licensing organization" and that "assessing some level of performance is frequently used as a competence measure, which means measuring aspects of the job at which a person is competent." Independently of IT2017, the MSIS2016 report [Acm11] introduced competencies at the master's level, indicating that "competencies represent a dynamic combination of cognitive and meta-cognitive skills, demonstration of knowledge and understanding, interpersonal, intellectual and practical skills, and ethical values." The *Software Engineering Competency Model* [Iee3] defined competency as the "demonstrated ability to perform work activities at a stated competency level." These three publications suggest that competency is some combination of knowledge, technical skills, and human behavior within a computing context.

Information Technology

The information technology IT2017 report embraced competency-based learning, as opposed to the *knowledge area, knowledge unit, learning outcome* model, mostly because almost all graduates from information technology degree programs enter industry and the workplace. The report adopted the term competency as related to performance in the workplace, that is, what a graduate should bring to a job.

In education, success in career readiness requires that students in degree programs develop a range of qualities typically organized along three dimensions: knowledge, skills, and dispositions, so *competency* must connect these three elements or dimensions. The IT2017 report described this concept simply as:

COMPETENCY = KNOWLEDGE + SKILLS + DISPOSITIONS … *in Context*

The interrelated dimensions had the following meanings. *Knowledge* designates an awareness and understanding of core concepts and content. This dimension receives initial attention from teachers when they design their syllabi, from departments when they develop program curriculum, and from accreditation organizations when they articulate accreditation criteria. This is the "know-what" dimension. *Skills* refer to capabilities and strategies that develop over time through deliberate practice and through interactions with others. Skills also require engagement in higher-order cognitive activities such as programming. This is the "know-how" dimension. *Dispositions* encompass socio-emotional skills, behaviors, and attitudes that characterize the inclination to carry out tasks and the sensitivity to know when and how to engage in those tasks [Per1]. This "know-why" dimension is the most challenging for academics because some computing teachers may ignore disposition in educational settings.

There has been general agreement in education that success in career readiness requires that students in degree programs develop a range of qualities typically organized along these three dimensions. The IT2017 report also addressed approaches to learning. It rejected the content-driven mode of framing curricular guidelines using a disciplinary body of knowledge that can be subdivided into areas, units, and topics to track recent developments in the rapidly changing computing field. Instead, it proposed the use of the "Understanding by Design" approach to transform content-based curricular models into a competency-based curricular framework. Here, learning transfer is multi-faceted with the transfer blended with skills and dispositions. Dispositions relate to metacognitive awareness, for example, being responsible, adaptable, flexible, self-directed, and self-motivated, and having self-confidence, integrity, and self-control. They also include how to work with others to achieve common goal or solution.

Information Systems

Instead of specifying a body of knowledge or a set of courses as developed in the previous MSIS2006 report, the MSIS2016 curricular model identified a set of graduate competencies. Here, the term "competency" referred to graduate level ability to use knowledge, skills, and attitudes to perform specified tasks successfully. The report used a more formal definition for competency, as mentioned in the previous section, as follows:

> *Competencies represent a dynamic combination of cognitive and metacognitive skills, demonstration of knowledge and understanding, interpersonal, intellectual and practical skills, and ethical values [Loc1 p21].*

In this context, competency is an integrative concept that brings together graduate level knowledge, skills, and attitudes.

The report also specified four different levels of category attainment: awareness, novice, supporting (role), and independent (contributor). The awareness level implies that a graduate student knows that the competency category exists and is aware of the reasons it is important for the domain of practice. The novice level specifies that a graduate can effectively communicate regarding matters related to the competency, perform component activities under supervision, and develop on-the-job experience related to the competency. The supporting (role) level indicates that a graduate has achieved a level of knowledge and skill that allows him/her to collaborate effectively in a supporting role with colleagues who have achieved a higher level of the competency to produce the desired outcomes. Finally, the independent (contributor) level refers to a graduate who has achieved a level of knowledge and skills that allows the graduate to perform without continuous support/supervision, the tasks required to produce the desired outcomes. Higher levels of competencies do exist, at an expert level.

The MSIS2016 curricular model suggested that all programs should not expect to prepare students to attain competencies at the same level in all competency categories. Different professional profiles have different needs and the professional profiles that a program desires its graduates to achieve can vary. That is, programs should determine the level at which its graduates should attain each of the competency categories.

Software Engineering Competency Model

The software engineering competency model (SWECOM) [Iee3] described capabilities for software engineers who participate in the development of and modifications to software-intensive systems. The model specifies skill areas, skills within skill areas, and work activities for each skill. Activities occur at five levels of increasing proficiency.

The SWECOM suggests that competency is a combination of knowledge, skill, and ability. A competent person has the knowledge and ability to perform work activities (i.e., skills) at a given competency level. The competency model includes cognitive attributes, behavioral attitudes, and technical skills. Some cognitive skills include reasoning, analytics, problem-solving, and innovation skills. Behavioral attributes include aptitude, enthusiasm, trustworthiness, cultural sensitivity, as well as communication, teamwork, and leadership skills. The model also specifies lifecycle skill areas, cross-cutting skills (e.g., quality, safety, security), and related activities. It also defines competency levels to be that of a technician (able to follow instructions), an entry-level practitioner (can assist in performing activities with some supervision), a practitioner (able to perform activities with little or no supervision), a technical leader (capable to lead and direct participants), and a senior software engineer (capable to create new processes and modify existing processes).

The SWECOM is very similar to the IT2017 and the MSIS2016 philosophies of competency. Knowledge and technical (computing) skills are integrated with behavioral attributes that correspond to either disposition or ability. Competency is central to the model and provides a modern view to generate excellence in computing education.

4.1.3: Initial and Developing CC2020 Explorations of Competencies

As noted above, interest in CC2020 lies in both identifying the evolution toward competency-based model curricula that have taken place over the past several years and articulating a sound and clear approach to writing competencies that are useful for future curricular efforts. This chapter addresses the former. In pursuit of the latter, the CC2020 task force in 2017-2018 initially explored the creation of competency statements by organizing subgroups of experts from different computing disciplines. Their initial work produced competency statements compatible with the definition that *Competency = Knowledge + Skills + Dispositions*, in context. Appendix C presents a set of preliminary draft computing competencies for computer engineering, computer science, information systems, information technology, and software engineering competencies generated from this early effort and other work. These initial explorations from 2018 have inspired a more detailed expression of competency, presented in the next section.

4.2: A Competency Model

The CC2020 project has developed a definition of competency and a template for specifying the subject matter of baccalaureate computing education. This definition evolved from those developed and applied in the different educational frameworks reported in the IT2017 report, the MSIS2016 report, the SWECOM report, as well as the preliminary work conducted on developing competencies within CC2020 mentioned in Section 4.1.3

The CC2020 representation that this report develops supports a consistent, scalable model for writing curricular specifications and competencies. One could also use it for automated visualization and comparison of curricula. However, this CC2020 report provides only a framework for creating competencies. It does not create new competencies because they could vary greatly based on use, task, or context. That is, CC2020 provides readers with a competency framework and it lets each program unit or curricular group develop their own set of competencies for their purposes and interests.

4.2.1: The CC2020 Competency Model

CC2020 articulates a notion of competency as a practical educational goal [Wag5, Fre5, Tak1, Top5] that refines the Knowledge-Skill-Disposition (K-S-D) framework popularized in the IT2017 curriculum report. While the knowledge dimensions of computing have been extensively explored in the various computing curricula, what is meant by skill and disposition have had significantly less focus. Extending previous work, we specify competency as composed of K-S-D dimensions observed within the performance of a task, T.

COMPETENCY = [KNOWLEDGE + SKILLS + DISPOSITIONS] IN TASK

A competency specification enumerates knowledge, skills, and dispositions that are observable in the accomplishment of a task, a task that prescribes purpose within a work context [Wag5]. Figure 4.1 illustrates the conceptual structure of competency.

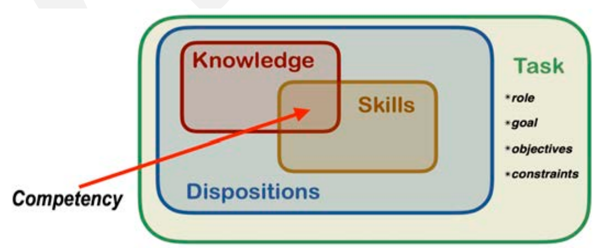


Figure 4.1. Conceptual Structure of the CC2020 Competency Model

4.2.2: Component Definitions

The four components (knowledge, skills, dispositions, and task) that structure the competency specification have the following meanings.

Knowledge

Knowledge is the "know-what" dimension of competency as a factual understanding. This dimension reflects the enumerated subject matter that teachers catalog as topics in their syllabi, departments distribute and balance among the courses they develop in an academic program, accreditation organizations stipulate in their accreditation criteria, and employers identify in job descriptions of their workers. An element of knowledge designates a core concept essential to a competency. Alone, however, a concept is static and inert; it must be acted upon with a degree of expertise to become a behavior.

Skills

Skills introduce the capability of applying knowledge to actively accomplish a task. Hence, a skill expresses an element of knowledge as acted upon with proficiency to define the "know-how" dimension of competency. Skills require time and practice to develop. Consequently, skill development often requires engagement in a progressive hierarchy of higher-order cognitive process. CC2020's definition of competency has adopted Bloom's levels of cognitive process [Acm22] to specify the degree of skill expected in successful task accomplishment.

One often assesses the skills dimension of competency indirectly through observation of the process or quality of work produced. The activation of "know-what" animated by "know-how" fuses the knowledge and skills dimensions. For that reason, the usefulness of any element of knowledge in a competency specification is only understandable when applied at a level of skillfulness; that is, specified or observed as a level of Bloom's cognitive process. Therefore, each element of knowledge and the requisite level of skill necessarily and naturally pair in the specification of a competency.

Dispositions

Dispositions frame the "know-why" dimension of competency and prescribe a temperament of quality of character in task performance. Dispositions moderate the behavior of applying "know-what" that becomes "know-how." How dispositions moderate knowledge and skill could be thought of as the extent that it accounts for the relation between the predictor and the criterion [Bar1] in that dispositions connect the 'better' or 'correct' application of knowledge and skill to the context where and why it is applied.

Dispositions are habitual inclinations that are socio-emotional tendencies, predilections, and attitudes (e.g., trustworthiness). Dispositions control whether and how an individual is inclined to use his/her skills. Dispositions can denote the values and motivation that guide applying knowledge while designating the quality of knowing indicative of a standard of professional performance.

Task

Task is the construct that frames the skilled application of knowledge and makes dispositions concrete. Task expressed as a colloquial prose statement provides the setting to manifest dispositions, where individuals moderate their choices, actions, and effort necessary to pursue and succeed in an efficient and effective manner. In this sense, task enfolds the purposeful context of competency, exposing the integral nature of knowledge, skills, and dispositions. To this end, a task definition stipulates pragmatic engagement that reflects professional practice relevant to the particular vision for the program graduates. For this reason, task descriptions provide an explicit context for the program to develop pedagogy that enables graduates to demonstrate competency as a computing professional.

4.2.3: Competency Statements

An effective specification of competency is a synthesis of 1) a colloquial, prose competency statement that sets out a task, and 2) the component structure of constituent K, S, and D elements necessary to succeed in that task. As a whole, a competency specification expresses a model of knowledge that is skillfully and professionally applied in some task execution.

The competency statement corresponding to a competency specification is a free-form colloquial expression that succinctly conveys the pertinent ability and goals attained through a course of study or the capabilities relevant to successfully performing a task in the workplace. The competency statement expresses the competency in terms that are familiar and comprehensible to a wide audience, typically using a vocabulary familiar to, and that resonates with, the stakeholder audience. The competency statement is then structurally augmented and amplified in the enumeration of knowledge, skills, and dispositions that complete the competency specification.

While the natural language of the competency statement favors a public audience, the competency component structure is more formal as it enumerates the components, e.g., knowledge elements demonstrated at a skill level and moderating dispositions determined necessary to demonstrate the competency in task. This structural enumeration of components is essential for automating comparative analyses and visualization of curricula. Having both the free-form of the competency statement alongside the more formal component-specific enumeration corroborates that the two perspectives align. Any divergence perceived between these perspectives would suggest the need for a closer reflection upon the correctness of one or both representations.

4.2.4: Component Elements

A competency is a collection of specific components of knowledge, skills, and dispositions. Tables 4.1, 4.2, 4.3, and 4.4 present suggested elements of these dimensions. The knowledge dimension of competency encompasses concepts that are technical (computing concepts), foundational and professional (indicative of a workplace), and domain specific (the task setting). Appendix D elaborates on these component tables in greater detail.

Table 4.1 illustrates thirty-four abbreviated knowledge areas partitioned into an ordered sequence of six categories. While the table is incomplete, it does provide an example high-level vocabulary for computing knowledge rooted in the collective wisdom of different computing communities. This summary of computing knowledge areas represents a well-understood and consistent vocabulary from which computing competency statements can evolve.

Table 4.1. Elements of Computing Knowledge

Users and Organizations	Systems Modeling	Systems Architecture and Infrastructure	Software Development	Software Fundamentals	Hardware
Social Issues and Professional Practice Security Policy and Management IS Management and Leadership Enterprise Architecture Project Management User Experience Design	Security Issues and Principles Systems Analysis & Design Requirements Analysis and Specifications Data and Information Management	Virtual Systems and Services Intelligent Systems (AI) Internet of Things Parallel and Distributed Computing Computer Networks Embedded Systems Integrated Systems Technology Platform Technologies Security Technology and Implementation	Software Quality, Verification and Validation Software Process Software Modeling and Analysis Software Design Platform-Based Development	Graphics and Visualization Operating Systems Data Structures, Algorithms and Complexity Programming Languages Programming Fundamentals Computing Systems Fundamentals	Architecture and Organization Digital Design Circuits and Electronics Signal Processing

The thirteen elements of foundational and professional knowledge listed in Table 4.2 represent a subset of the professional listings derived from the IT2017 report and subsequently from Appendix D. Computing professionals are commonly expected to demonstrate high levels of skill in applying this knowledge which deserves explicit attention in baccalaureate programs.

Domain knowledge represents elements of the context that situates the task. In the general, these elements may represent the disciplinary (e.g. business, medicine, manufacturing, etc.). In the more detailed they may be more specific (e.g. international currency exchange, radiographic imaging, automobile assembly, etc.). In any case, the scope and level of detail of domain knowledge will emerge from the intended use of the competency (i.e. Computing + X and X + Computing, See Section 2.4).

Table 4.2. Elements of Foundational and Professional Knowledge

Knowledge Elements	Meaning
Analytical and Critical Thinking	A mental process of simplifying complex information into basic parts and evaluating results to make proper decisions.
Collaboration and Teamwork	Apportion challenging tasks into simpler ones and then work together to complete them efficiently.
Ethical and Intercultural Perspectives	Ethical perspectives are the different viewpoints someone uses to view a problem in the context of individual human values.
Mathematics and Statistics	Use of numbers and theories abstractly especially in the collection and analysis of numerical data
Multi-Task Prioritization and Management	Processing several issues or tasks at once while arranging them according to importance to do specific one first.
Oral Communication and Presentation	Conveying a message orally using real-time presentations with visual aids related audience interests and goals.
Problem Solving and Trouble Shooting	A logical and orderly search for the source of a unit problem and making the unit operational again.
Project and Task Organization and Planning	A process to provide decisions about a project concerning organization and planning to achieve a successful result.
Quality Assurance / Control	Use of techniques, methods, and processes to identify and prevent defects according to defined quality standards.
Relationship Management	A strategy to maintain an ongoing level of engagement usually between a business and its customers or other businesses.
Research and Self-Starter/Learner	Someone who begins or undertakes work or a project without needing direction or encouragement to do so.
Time Management	An ability to use a person's time in an effective or productive manner to work efficiently.
Written Communication	Use of a written form of interaction between people and organizations that provides an effective way of messaging.

As CC2020 defines skill, the proficiently applying knowledge, Table 4.3 summarizes an ordered sequence of six cumulative levels of skill (cognitive skill) together with abbreviated definitions. These levels correlate with Bloom's taxonomy that permits the adoption of a commonly agreed vocabulary as described in the 2001 revisions to Bloom's taxonomy of educational objectives [And5]. The table lists the cognitive skills as verbs.

Table 4.3. Levels of Cognitive Skill Based on Bloom's Taxonomy

Remembering	Understanding	Applying	Analyzing	Evaluating	Creating
Exhibit memory of previously learned materials by recalling facts, terms, basic concepts, and answers	Demonstrate understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions.	Solve problems to new situations by applying acquired knowledge, facts, techniques, and rules in a different way	Examine and break information into parts by identifying motives or causes; make inferences and find evidence to support solutions.	Present and defend opinions be making judgments about information, validity of ideas, or quality of material.	Compile information together in a different way by combining elements in a new pattern or proposing alternative solutions.

Dispositions define the third dimension of competency. Table 4.4 displays eleven prospective dispositions derived from the literature. Disposition, as an intrinsic component of competency, represents the opportunity to express institutional and programmatic values expected in the workplace. Dispositional expectations enrich the description/assessment of competency and/or the related pedagogy. Ascribing a disposition to a competency indicates a clear commitment to self-reflection and examination that distinctly distinguishes a competency from a learning outcome.

Table 4.4. Prospective Elements of Dispositions

Element	Elaboration	Element	Elaboration
Proactive:	With initiative, self-starter, independent	Adaptable:	Flexible; agile, adjust in response to change
Self-directed:	Self-motivated, determination, independent	Collaborative:	Team player, willing to work with others
Passionate:	Conviction, strong commitment, compelling	Responsive:	Respectful; react quickly and positively
Purpose-driven:	Goal driven, achieve goals, business acumen	Meticulous:	Attentive to detail; thoroughness, accurate
Professional:	Professionalism, discretion, ethical, astute	Inventive:	Exploratory. Look beyond simple solutions
Responsible:	Use judgment, discretion, act appropriately		

Dispositions are an essential characteristic of a well-structured competency model, and they have an intricate involvement in statements related to workplace or academic activities. People inherently know and recognize these elements of human behavior. While it may be difficult to teach disposition, faculty members should instill these concepts in their students through assessment design, exercises, sustained practice, readings, case studies, and thei own example. The workplace and society assume that dispositions as in Table 4.4 are expected of every competent computing graduate.

4.2.5: Creating Competency Statements

The competency model adopted in this report suggests that statements surrounding competency include knowledge elements paired with skill level and with dispositions. The following examples demonstrate a way to do this. Each of the three example competencies that follow specifies a statement of the task to be undertaken and itemizes the components deemed pertinent to effectively and efficiently accomplishing that task.

Example A: From Computer Engineering

Competency Title: A	
Competency Statement Manage the design of a computer system for a manufacturer using appropriate tools, design digital circuits including the basic building blocks of Boolean algebra, computer numbering systems, data encoding, combinatorial and sequential elements.	
Knowledge Element [Table #]	**Skill Level** [Table 4.3]
Architecture and Organization [4.1]	Creating
Digital Design [4.1]	Creating
Circuits/Electronics [4.1]	Creating
Analytical and Critical Thinking [4.2]	Applying
Mathematics and Statistics [4.2]	Applying
Problem Solving and Trouble Shooting [4.2]	Applying
Research and Self-Starter/Learner [4.2]	Applying
Disposition(s) [Table 4.4]	
Self-directed Meticulous Inventive	

Example B: From Information Technology

Competency Title: B	
Competency Statement Analyze and compare several networking topologies in terms of robustness, expandability, and throughput used within a cloud enterprise.	
Knowledge Element [Table #]	**Skill Level** [Table 4.3]
Computer Networks [4.1]	Analyzing
Platform Technologies [4.1]	Analyzing
Analytical and Critical Thinking [4.2]	Applying
Mathematics and Statistics [4.2]	Applying
Quality Assurance [4.2]	Applying
Disposition(s) [Table 4.4]	
Self-directed Purpose=driven Responsible	

Example C: From Software Engineering

Competency Title: C	
Competency Statement Identify and document system requirements by applying a known requirements elicitation technique in work sessions with stakeholders, using facilitative skills, as a contributing member of a requirements team.	
Knowledge Element [Table #]	**Skill Level** [Table 4.3]
Requirements Analysis [4.1]	Evaluating
Oral Communication [4.2]	Applying
Written Communication [4.2]	Applying
Teamwork and Collaboration [4.2]	Applying
Disposition(s) [Table 4.4]	
Purpose=driven Responsible Collaborative	

4.3: From Competencies to Curricula

A coherent competency model permits the definition of a computing curricula (i.e., structured collections of learning experiences) in a manner that benefits its constituencies: students, benefactors, faculty, administrators, employers, accreditors, lawmakers, and society. It is of interest to examine how key stakeholders can identify and author competencies as well as develop curricula based on the outcome expectations associated with competencies. This section summarizes the more comprehensive discussion in Appendix E and reviews topics that are essential for enabling the practical definition and use of competencies.

4.3.1: Identifying and Authoring Competencies

Differing stakeholder groups may desire to identify and author collections of competencies. Computing educators at a university may wish to identify a collection of competencies to define the expected outcomes of the university's baccalaureate program(s) in computing. Computing educators who represent professional or academic societies might desire to specify competencies as a means towards establishing the outcome expectations of a national or global level model curriculum. Industry representatives might use a collection of competencies to specify their expectations for degree program graduates either for a specific job or for general use.

As described earlier, stakeholders can specify competencies using either narrative competency statements or a component specification that separately identifies the knowledge-skill pairs and dispositions. For most purposes, the process of identifying and authoring competencies involves elicitation of required narrative competency statements in collaboration with those who best know the expectations that program graduates will face both soon after graduation and throughout their careers. Educators writing competency statements for example, might collaborate with employers, students, or educational authorities and/or bodies.

The methods and techniques for discovering competencies for a program specification are quite similar to those of systems requirements elicitation, including interviews, surveys, evaluation of existing requirements, as well as an analysis of industry recommendations (e.g., SFIA, e-CF) and academic model curricula (e.g., ACM/AIS/IEEE-CS recommendations).

In order to define the highest level or most abstract competencies of a program, a course, or other curricular unit, it is necessary to articulate the knowledge, skill, and disposition components associated within a context. In a free-form competency statement, the focus is typically on the general outcome of the competency in the expected context. Expressing a competency in this manner, the knowledge, skill, and disposition components might not contain full or detailed exposure. Instead, users may need to infer the details from the free-form statement. Therefore, articulating the context of a competency is always crucial since it provides the motivation for the stakeholder, making it meaningful to learn and perform that competency.

The literature offers some insight for developing competency statements [Per5]. In this setting, writers of competency statements should:

1. Stipulate them as learner-oriented, essential competencies.
2. Specify them in "clear, specific, unadorned, and concise language," that are measurable.
3. Structure them as action oriented and begin with "the verb that most precisely describes the actual, preferred outcome behavior to be achieved."
4. Construct them to be consistent with "standards, practice, and real-world expectations for performance," thus reflecting what "the practitioner actually needs to be able to do."
5. Formulate them to contribute to a "cluster of abilities needed by the graduate to fulfill the expected overall performance outcomes."

Component specifications fully aligned with competency statements are essential for comparison and analysis purposes. In addition, the process of translating a free-form competency statement into a component specification may reveal non-desirable characteristics of the statement and can offer opportunities for significant improvement. The process of deriving component specifications from free-form statements is an iterative one and requires willingness and ability to interpret the statements in a way that allows identification of components inferred from the narrative statement.

In some cases, it could be useful to start from the competency components. Identifying the knowledge, skills, and dispositions components of a competency before constructing a competency statement could also be a good starting point. This is especially true in cases when the identity of a target competency is not fully clear and first requires calibration at a component level.

4.3.2: Competency Specifications and Curricular Specifications

Competencies alone do not address the question of how the educational experiences needed to enable students to acquire expected competencies by the time of graduation can be determined. Outcome expectations specified as competencies require transformation into a curriculum form consisting of educational activities that help to scaffold students' progression in various types of outcome areas.

Past experiences associated with processes that derived educational experiences from competencies can be helpful (e.g., the MSIS 2016 report [Acm11], the International Academy of Astronautics (IAA) Space Industry Systems Engineering competency model, and the business curricula for competency specifications [Chy5].) Guidance from these experiences includes the following:

- Determine the characteristics of learning experiences that constitute a curriculum based on outcome expectations specified with competencies.
- Indicate program competencies as a foundation for curricular specification benefits from existing competency models (e.g., those developed by industry, government, or professional societies).
- Develop educational experiences that require not only identification of competencies, but also specification of the expected attainment levels from novice to expert.
- Derive an initial set of learning outcomes associated with each competency; then, organize the learning outcomes into learning experiences. The sets of learning outcomes within each learning experience determine the topics in which students should engage and the pedagogical forms expected from the engagement.
- Assess continuously the extent to which the implemented learning experiences enable students to attain the expected competencies at the expected level.

The opportunity for students to develop skills and dispositions is a positive but potentially resource-intensive effect of specifying program outcomes with competencies. In many cases, such an approach requires a different set of pedagogical assumptions and methods compared to a mostly knowledge-based specification of assessable outcomes. In practice, competency-based outcome specification can lead to a broader set of types of learning experiences. These often include a much stronger focus on various forms of experiential learning, from interactive simulations, to intensive projects, to field experiences, and to internships and cooperative programs with industry. Domain-specific skills and dispositions require a learning environment that is different from a traditional classroom environment.

4.4: Digest of Chapter 4

This chapter deals with the nature of competency – a salient feature of the CC2020 project. It presents several competency statements to exemplify applications. Competency-based curricula are more expressive in their learning goals, and more easily translated to the language of graduate job descriptions and industry needs. Recognizing the knowledge-based approaches taken in many computing curricula to date, recent developments in computing curricula imply that the components of computing curricula should include not just knowledge and skills but also dispositions, skill levels, and typical (maybe "practical") tasks expected of graduates. The use of the competency model can also assist in potentially automating the comparative analyses and visualization of curricula programs in computing. For these reasons, the CC2020 task force recommends that future curricular reports adopt this competency-based approach to describe computing curricula.

Chapter 5: Visualizations

This chapter describes the analysis and visualization of curricula specified using different approaches. The goal is to handle the digital representation of these and be able to analyze courses, programs, and curricula that have been specified within paradigms described in Chapter 3 and Chapter 4. This chapter suggests illustrative examples and is not intended to be exhaustive.

This report is based on the curricula already established by ACM and other professional organizations. It does not introduce new curricula. Thus, these illustrative examples are just analyses and visualizations of existing curricula and are not introducing new curricular specifications. Readers should consult the original reports for curricular specifications.

5.1: On Visualization

The competency model introduced in the previous chapter specifically lends itself to visualization and analysis. Here we introduce a visualization toolset and present an approach to formally analyzing curricula that are defined using our competency model.

Data form the basis of analysis and visualization. A given specification (e.g., a competency statement consisting of knowledge elements paired with skills, and dispositions) forms a basic data set. A repository that stores knowledge and competency specifications is central to the data and its analysis. The elements of knowledge, skills, skill level, and dispositions appear in Tables 4.1, 4.2, 4.3 and 4.4.

5.1.1: Some Basic Functions

The basic functions of this set of tools include the following:

1. Content Management: User(s) may enter competency specification(s) in various formats.
2. Reporting/Presentation: User(s) may retrieve, display, format, and disseminate representations of competency specifications.
3. Analysis: User(s) may query repository content specifying any category attributes or specification content and represent the query results as listings, comparisons, or visualizations for the purpose of analyses.

The first function (content management) supports the collection and curation of glossaries of knowledge, skills, and dispositions along with synonyms and translations. Repository contents may be input manually or mechanically imported/exported using published formats and protocols (see Appendix F). This will accommodate not only competencies specific to formal curricular guideline development but also industry characterizations of professional and employment competency.

The second function (reporting/presentation) provides a facility for representing competency specifications formatted for copying into formal organizational or institutional documents such as academic programs, accreditation standards, or professional licensure reports.

The third function (analysis) concerns analyzing and visualizing knowledge or competency specifications. This may occur individually or collectively in comparison with each other. “Low-level” analysis involves individual specifications, and “high-level” analysis involves collections of specifications. This approach is useful for various users and stakeholders.

5.1.2: Analysis of Competencies

Competencies can be assembled into collections such as curricula, curriculum standards, specific job requirements, or requirements for categories of jobs. All such specifications are similar in nature and structure. Hence, it is possible to define the general notion of a "competency target" that reflects an entity defined by a collection of competencies.

Table 5.1 identifies four competency targets. The Singular-Aggregate dimension of the table reflects whether the target is for a single entity or for a category of entities. The Education-Workforce dimension reflects whether the target relates to an education product or to a workforce product.

Table 5.1. Competency Targets

	Education	**Workforce**
Singular	Programs	Jobs
Aggregate	Subdisciplines	Careers

Each of these four targets represent an important application of competencies. *Programs* are individual computing educational programs delivered by colleges and universities. *Subdiscipline* represents curricular standards for each computing subdiscipline developed by the professional societies such as future curricular reports for computer engineering or computer science. *Jobs* reflects a specific work opportunity where industry or government can specify the requirements in terms of a set of competencies. *Careers* is a category composed of similar jobs where industry or government can specify the requirements across a category in terms of a set of competencies.

Since a set of competencies can characterize a target, it is possible to view the structured K-S-D portion of each competency's specification as a point in 3-D space, and a set of competencies as a point cloud. This approach lends itself to a visualization of competencies that require further exploration as presented in this chapter and Appendix G.

While visualization of competencies in this model may provide insights, the idea of considering the proximity of targets is also a promising concept. Developing a specific distance metric between two targets is a potential area for future research and is based on the idea of obtaining an ordered value for this metric. Such a distance metric could enable pairs of targets ranked in terms of "closeness" similarity. For example, suppose students are searching for an educational program (i.e., *Programs*) to prepare themselves to be network administrators (i.e. *Careers*). If competency specifications exist for both the potential programs and the desired career, then the distance metric can rank programs in terms of how close they are to the desired career. The educational program that is the closest distance to the network administrator career target could be the optimal degree program for the students.

It is possible to use a distance metric to support comparisons among all four types of targets. The following scenarios provide opportunities for target proximity.

- For education providers, there is the opportunity to reduce the distance between the competencies associated with a program and the targeted jobs and careers by that program. The articulation of that distance could allow providers to make changes that close the gap and positively establish a reduction in that distance.
- For education providers, there is the opportunity to calibrate programs with national and international standards for curricula in various fields by evaluating the distance between the provider's program and the curricular standard for the subdiscipline.
- For pre-college students, there is the opportunity to select a program based on the program whose competencies are the shortest distance from the desired job or career.
- For hiring organizations, there is the opportunity to quantify the distance between the competencies required for a position, and the competencies exhibited by various candidates based on their completion of various educational programs and processes.
- For college graduates, there is the opportunity to search for jobs based on the distance between personally held competencies and the competencies required for targeted careers.

Based on pairs of targets, Table 5.2 conceptualizes several questions that could be addressable by this framework.

Table 5.2. Framework Questions

Target #1	Target #2	Practical Exemplar Question
Program	Career	How well does ABC University's information technology program prepare someone to be a network administrator?
Subdiscipline	Career	How well does a computer engineering degree from XYZ University prepare someone to be a chief information security officer?
Program	Job	How well does ABC University's computer science program prepare someone to be a senior programmer at ACME Corporation?
Program	Career	How well does ABC University's information systems program prepare a current business student to develop a career in programming?
Program	Program	What are the differences between ABC University's computer engineering program and XYZ University's software engineering program?
Program	Subdiscipline	How closely aligned is ABC University's current computer science program to the (hypothetical) competency-based curriculum espoused the pending CS202x report?

The target proximity approach attempts to unify the education and workforce sides of computing. The past approach was to define computing curriculum standards in terms of knowledge areas, knowledge units and learning outcomes. That approach complements academia and uses the classical role and scope of higher education as the curator of knowledge.

On the workforce side, there have been successful attempts to define job requirements in terms of competencies through the development of competency frameworks. Recall the examples mentioned earlier and the Skills Framework for the Information Age (SFIA), the European Competency Framework (e-CF), and the i Competency Dictionary (iCD).

As noted in Chapter 4, computing educators have been transitioning to competencies for several years for some of the recent model curricula. However, the process is a long way from utilizing a common language that transcends both education and workforce. The lack of a formal structure in many previous notions of competencies means that constituents have not had a way to quantify and analyze competencies in terms of the questions in Table 5.2. This CC2020 report advocates a transition over time to a common language that stakeholders can utilize across education and workforce constituencies to understand and minimize the gap between education outputs (graduates) and the inputs required for a successful contribution to a global workforce in computing.

5.2: Competency-based Visualization Examples

The questions or queries typically made by each of the stakeholder communities can suggest various visualizations of the query results. Users choose these representations to demonstrate the expressive potential of graphic communication. There is no intent to describe the computation required for the data to underpin the graphic. They are, however, all conceived as derived from the competency repository structure that implements the competency specification syntax. See Appendix G for use cases as well as many visualizations consistent with the CC2020 project.

The procedure used in the following discussion assumes that data is available for use and analysis. From this basis, it is possible to visualize competency for stakeholder use.

5.2.1: Student Use Case

A student is interested in entering undergraduate education in computing and wants to know what type of curriculum would best fit her interests. She might have some ideas about dispositions that are relevant in her future curriculum, and/or have a preliminary view on domains that would provide her with future job opportunities. She might start by checking promising dispositions (or, alternatively, she could start by choosing the knowledge categories and areas – we show only the first scenario, but the alternative would lead to the same results). She would see a list of dispositions (Figure 5.1(a)), from which she would choose, resulting in the interface showing the chosen dispositions as shown in Figure 5.1(b). Note that the dispositions are indicated by color, as there is no order dimension.

(a) Before choosing

	Dispositions
	Proactive
	Self-directed
	Passionate
	Purpose-driven
	Professional
	Responsible
	Adaptable
	Collaborative
	Responsive
	Meticulous

(b) After choosing

	Dispositions
✔	Proactive
	Self-directed
✔	Passionate
	Purpose-driven
	Professional
✔	Responsible
	Adaptable
	Collaborative
✔	Responsive
	Meticulous

Figure 5.1. Choosing dispositions by a prospective student

The student may also indicate which knowledge categories and knowledge areas seem interesting for her. Figures 5.2 and 5.3 show a possible process. She first chose three categories: *Users and Organizations*, *Systems Modeling*, and *Software Fundamentals*. In Figure 5.2, the ellipse of these three categories are highlighted with red borders. If needed, the student could indicate which individual knowledge areas are most relevant. Figure 5.3(a) shows the knowledge areas for each of the chosen three categories. The student chose the knowledge area *User Experience Design* for *Users and Organizations* category, and *Systems Analysis and Design* and *Requirements Analysis and Specification* for *Systems Modeling* category; the ellipse of the chosen knowledge areas are highlighted with red borders. The student did not want to make a detailed choice in the category of *Software Fundamentals*. The resulting choices are shown in Figure 5.3(b).

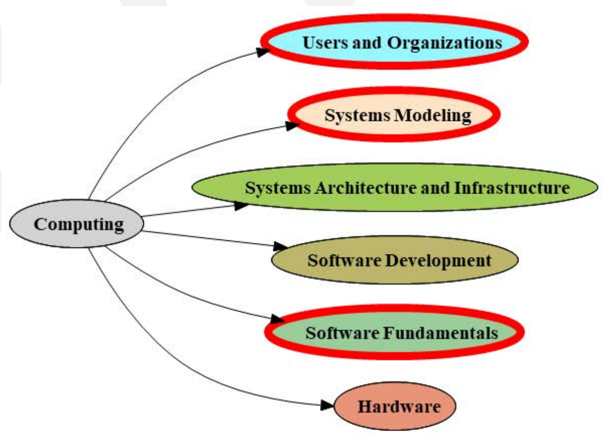


Figure 5.2. The student's choice of computing categories

(a) Choosing knowledge areas (b) Final result

Fig. 5.3 Detailed choice of knowledge areas

If the student is satisfied with this set of knowledge areas, she may confirm and ask for a global view of how the various curricula match her interests. Based on the student's choices, the system searches for curricula that fit this intended content. In Figure 5.4, the intended knowledge categories (which have been partly specified into knowledge areas) are mapped for each of the six curricular guidelines. The blue squares indicate the extent to which the knowledge area/category is relevant in the corresponding curriculum. The green square is the relative match of the student choices to that of the curriculum. The calculation of the size of the blue and green squares is not fixed yet, but for example, the green square could be based on the weights that are given in Table 5.3. Since the student is more interested in software modeling, based on the message given in Figure 5.4, the student decides to explore details regarding SE and her favored knowledge categories. By hovering over a square (Figure 5.5), the corresponding competencies are listed. Also displayed are the dispositions linked to the competencies along with the relative level computed from the student choices.

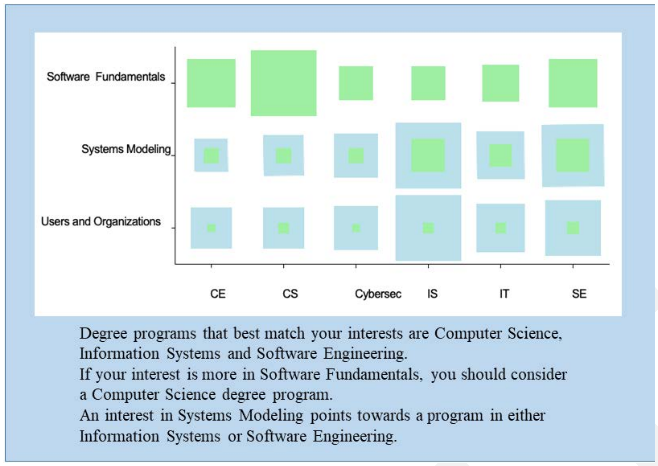


Figure 5.4. Mapping of chosen knowledge categories to the six curricular guidelines

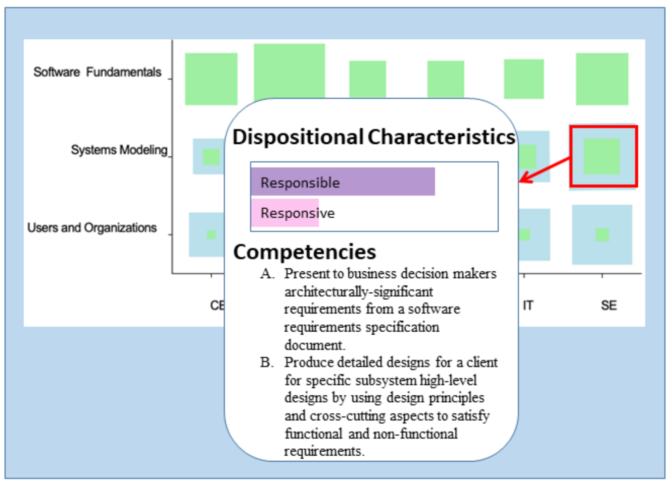


Figure 5.5. Disposition and competency details

5.2.2: Industry Use Case

A user from industry has developed a list of relevant knowledge areas for which relevant skills, knowledge levels, and/or dispositions are required for the company's computing employees. She wants to find out which curriculum might potentially provide professional education for the company's employees, in their context. Initially, CS and IT seem to be available and promising.

Similar to the process that the student took in Figures 5.2 and 5.3 in section 5.2.1, the industry user decides to choose *Hardware*, *Software Fundamentals*, and *Software Development* as categories that seem relevant, and removes the other three categories. She then checks the knowledge areas for each of the chosen categories and chooses the areas that she believes to be relevant for her, resulting in Figure 5.6.

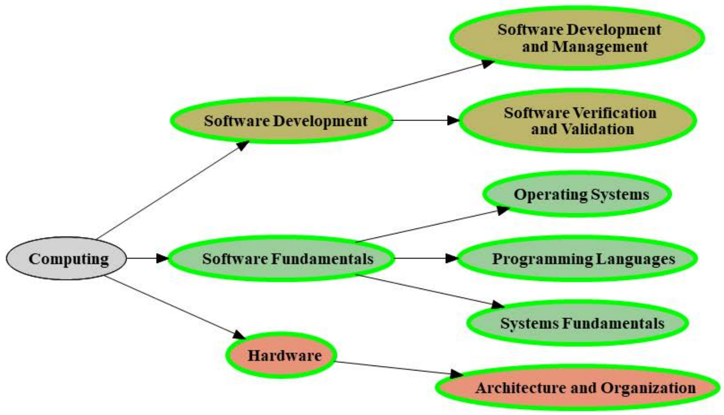


Figure 5.6 Result of knowledge areas selection

The user is now able to indicate for each of the selected knowledge areas to, either or both, indicate what skill level would be required, and what dispositions are relevant. Suppose that the user indicates that she is willing to provide specifications for the knowledge area *Software Fundamentals*. In Figure 5.7, the skill level is specified by using a slider, and the disposition is specified by choosing from a menu.

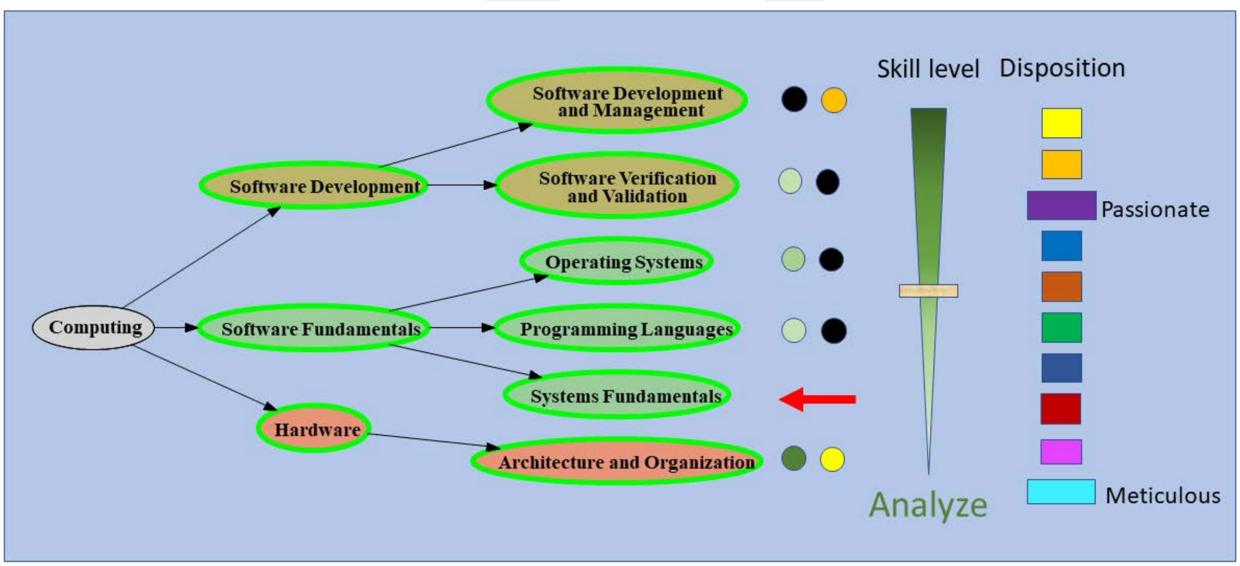


Figure 5.7. Detailing skill and disposition

When all relevant specifications for the selected knowledge areas have been provided, the system generates a radar chart comparing the knowledge level for selected curricula. The distance from the center indicates the skill level related to each knowledge category. Figure 5.8 compares the curriculum of CS and IT. The radar chart has been augmented with the specification from the user. In the example, it seems CS is the best match for the user's required knowledge levels. This is because there is a complete coverage of the user's specifications and the curriculum content; that is, the blue CS surface completely overlaps the user's green specification surface.

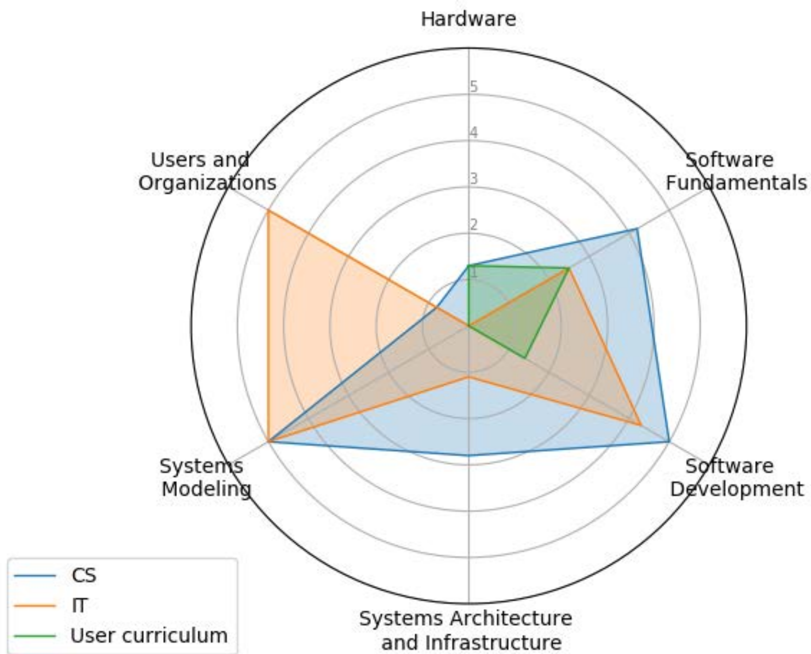


Figure 5.8. Comparison of CS and IT based on knowledge level

These two case examples have also been published in [Tak1].

5.3: Knowledge-based Visualization Examples

The procedure used in the following discussion assumes that data is available for use and analysis. From this basis, it is possible to visualize knowledge areas for stakeholder use.

5.3.1: Computing Educator

A computing educator has the question "How does my program fit with an international curricular guideline?" Figure 5.9 shows a comparison of an institution's evaluation of its program as a solid line to the evaluation based on the knowledge areas listed in Table 4.1 from Chapter 4. In this case, the evaluation denotes the weight of each knowledge element in the CS subdiscipline. The figure shows how this institution matches the guidelines and where the institution differs. For example, compared to the "standard" CS curriculum, this institution has a stronger emphasis on knowledge elements such as enterprise architecture, embedded systems, and on hardware related elements such as circuits and electronics.

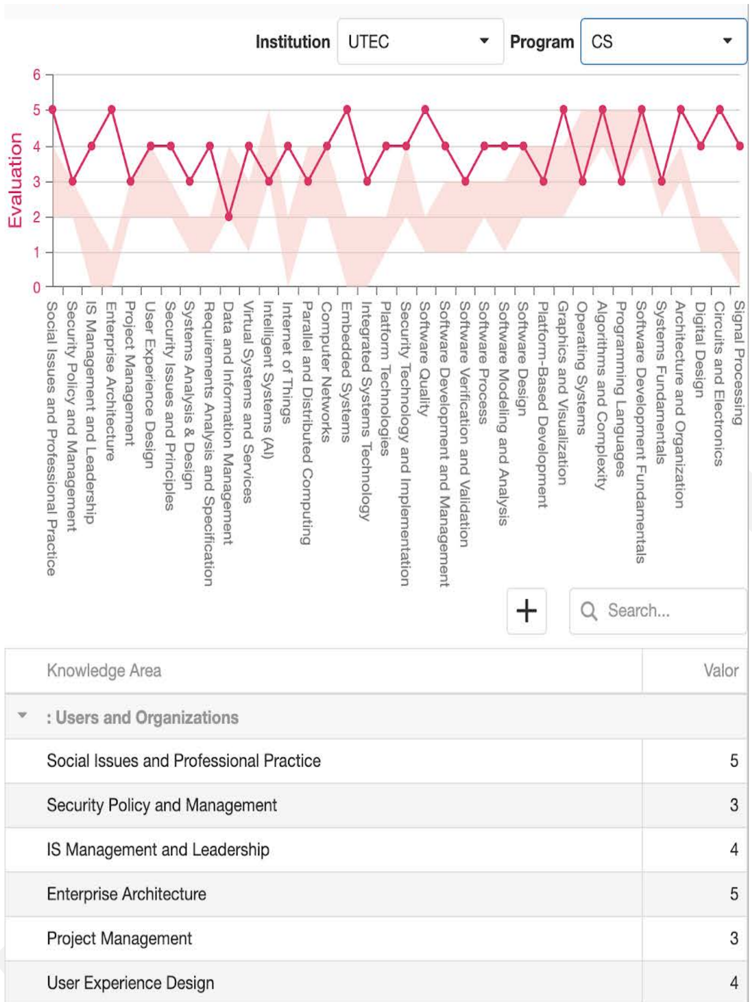


Figure 5.9. Comparison of an institution's evaluation against the knowledge table evaluation

5.3.2: Educational Authority

Educational authorities could also use Figure 5.9 to answer the question "Does this curriculum comply with the guidelines for curriculum X?" Figure 5.9 shows that none of the institution's evaluation falls below the minimum value of the "standard" CS curriculum. This outcome suggests that this institution's CS curriculum complies to the standard CS curriculum. Note that different stakeholders can use the same figure to address different questions.

5.3.3: Visualization of the Landscape of Computing Knowledge Table

In this section, Table 5.3 provides an overview representation of the landscape of computing knowledge. Structurally, it is similar to Table 3.1 in the CC2005 report. However, it was generated using a different process and its organization is different.

The values in Table 5.3 reflect the relative importance the integrated computing knowledge areas (represented by the rows) have for undergraduate degree programs in each of the included computing disciplines (represented by the columns). Each computing discipline specifies a minimum and maximum value suggesting an importance range within which most degree programs are likely to fall. This table was used as the foundation for the knowledge area set specified in Table 4.1 and for the visualization examples in this section.

Table 5.3 Landscape of Computing Knowledge

		CE		CS		CSEC		IS		IT		SE	
		Min	Max	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
1. Users and Organizations	1.1. Social Issues and Professional Practice	2	5	2	4	2	4	3	5	2	4	3	5
	1.2. Security Policy and Management	1	3	2	3	4	5	2	3	2	4	2	4
	1.3. IS Management and Leadership	0	2	0	2	1	2	4	5	1	2	1	2
	1.4. Enterprise Architecture	0	1	0	1	1	2	3	5	1	3	1	3
	1.5. Project Management	1	3	2	3	1	2	4	5	2	3	2	4
	1.6. User Experience Design	1	3	2	4	1	3	2	4	3	4	3	5
2. Systems Modeling	2.1. Security Issues and Principles	2	3	2	3	4	5	2	4	3	4	2	4
	2.2. Systems Analysis & Design	1	2	1	2	1	2	4	5	1	3	2	4
	2.3. Requirements Analysis and Specification	1	2	1	2	0	2	2	4	1	3	3	5
	2.4. Data and Information Management	1	2	2	4	2	3	3	5	2	3	2	4
3.Systems Architecture and Infrastructure	3.1. Virtual Systems and Services	1	3	1	3	1	2	1	2	3	4	1	3
	3.2. Intelligent Systems (AI)	1	3	3	5	1	2	1	2	1	2	0	1
	3.3. Internet of Things	2	4	0	2	1	3	1	3	2	4	1	3
	3.4. Parallel and Distributed Computing	2	4	2	4	1	2	1	3	1	3	2	3
	3.5. Computer Networks	2	4	2	4	2	4	1	3	3	4	2	2
	3.6. Embedded Systems	3	5	0	2	1	3	0	1	0	1	0	3
	3.7. Integrated Systems Technology	1	2	0	2	0	2	1	3	3	4	1	3
	3.8. Platform Technologies	0	1	1	2	1	2	1	3	2	4	0	2
	3.9. Security Technology and Implementation	2	3	2	4	4	5	1	3	2	4	2	4
4. Software Development	4.1. Software Quality, Verification and Validation	1	3	1	3	1	2	1	3	1	2	3	5
	4.2. Software Process	1	2	1	3	0	2	1	3	1	3	3	5
	4.3. Software Modeling and Analysis	1	3	1	3	1	2	2	4	1	3	4	5
	4.4. Software Design	2	4	2	4	1	3	1	3	1	2	4	5
	4.5. Platform-Based Development	0	2	2	4	0	1	1	3	2	4	1	3
5. Software Fundamentals	5.1. Graphics and Visualization	1	2	2	4	0	1	1	1	0	1	0	2
	5.2. Operating Systems	2	4	3	5	2	3	1	2	1	3	1	3
	5.3. Data Structures, Algorithms and Complexity	2	4	4	5	1	3	1	3	1	2	2	4
	5.4. Programming Languages	2	3	3	5	1	2	1	2	1	2	2	3
	5.5. Programming Fundamentals	2	4	4	5	2	3	1	3	2	4	3	5
	5.6. Computing Systems Fundamentals	2	3	2	3	1	2	2	3	1	3	2	3
6. Hardware	6.1. Architecture and Organization	4	5	3	4	1	3	1	2	1	2	1	3
	6.2. Digital Design	4	5	1	2	0	2	0	1	0	1	0	2
	6.3. Circuits and Electronics	4	5	1	2	0	1	0	1	1	2	0	1
	6.4. Signal Processing	3	4	0	1	0	2	0	1	0	1	0	1

The values in the table are based on expert opinions (instead of, for example, a representative survey of the disciplines.) They were derived using the following process:

1. The CC2020 steering committee analyzed knowledge areas included in the curriculum recommendation for six areas that included computer engineering, computer science, cybersecurity, information systems, information technology, and software engineering. In this process, 39 knowledge areas emerged that together cover the computing topics included in undergraduate degree programs in computing. Through follow-up analysis, the knowledge areas consolidated to the final 34.
2. Steering committee members were each asked to rate the importance of the knowledge areas for each of the six computing disciplines. The values in the table are the rounded arithmetic means of the responses.
3. A subgroup of the steering committee organized the knowledge areas based on the semiotic ladder [Liu1, Sta2] and labeled the six-layer categories as users and organizations, systems modeling, systems architecture

and infrastructure, software development, software fundamentals, and hardware. The organization of the knowledge areas in the final table followed this process.

Many types of visualizations can be made for the data found in Table 5.3. Figure 5.10 presents a radar chart that shows the maximum emphasis for the knowledge areas. Section 3 in Appendix G also gives a bar chart and line chart for the same data set, as well as a ribbon chart which could also be used for the same data set.

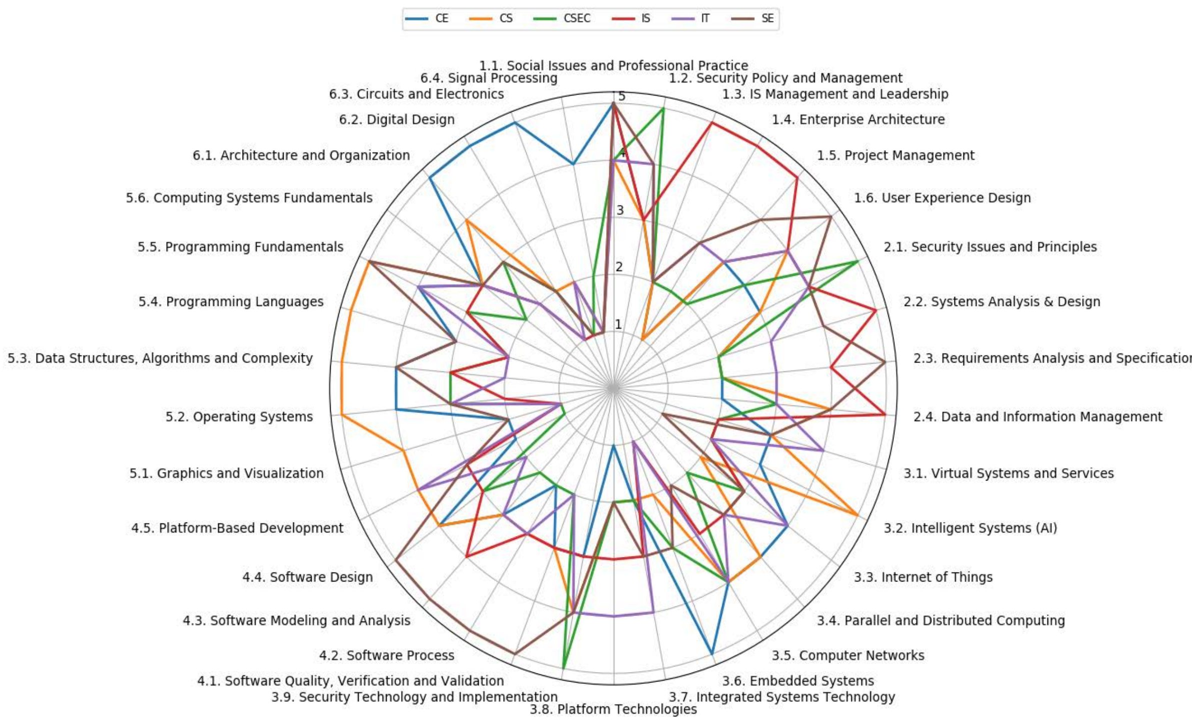


Figure 5.10. Radar Chart showing maximum emphasis of knowledge areas

Please note that the steering committee included these specific six computing disciplines in this integrative analysis because they were the ones for which a major computing society had approved an undergraduate curriculum recommendation. The seventh discipline, data science, to which this report refers did not have such a recommendation available at the time of this analysis. The same applies to any other new computing disciplines for which curriculum recommendations emerge. In fact, the process should repeat regularly after new versions of existing recommendations become available.

5.3.4: Other Visualizations

Other knowledge-based visualizations including the visualization of a whole curriculum appear in Appendix G.

5.4: Challenges Concerning Competency Visualization

The ability to visualize aspects of curricula and competencies present several challenges. These challenges can include the interpretation of nomenclature and vocabulary or the comparison of two entities.

5.4.1: Consistent Vocabulary

One issue that exists is the issue of standardized vocabulary for knowledge. For example, "refactoring" appears in both CS2013 and SE2014. In CS2013, the word refactoring is part of "software design" and "software evolution" knowledge units. In SE2014, refactoring is part of "software process" knowledge unit but is not in the "software design" knowledge unit. This suggests that refactoring may have a subtly different meaning between the two curricula, or that "software design" has a subtly different meaning.

In a perfect world, one would develop an ontology to which all computing curricular guidelines would adhere. This unfortunately may not be feasible in practice as there may be issues such as:

(1) Can one develop such an ontology?
(2) Would all computing sub-disciplines adhere to it?
(3) How does one handle emerging sub-disciplines?

These three issues are interesting topics that future work could consider.

5.4.2: Entity Comparison

There is also an issue of what it means to "compare" entities. A simple comparison would be to check for equality. But that may not always be appropriate. For example, as the skill level has an ordering aspect to it, if an element of knowledge K is required to be at the skill level of "understand", then any competency whose knowledge K is at a higher skill level, e.g., "apply", "analyze", "evaluate", "create", should satisfy or be sufficient for comparison purposes (for just that knowledge K).

Another issue is the comparison of composite competency specifications. Comparing two atomic competency specifications could be straightforward. We can just compare the respective knowledge-skill pairs and dispositions of the two atomic competency specifications. But there could be multiple ways to compare composite competency specifications. For example, one possibility is to drill down and compare the leaf competency specifications. Another possibility is to compare at the top-level competency specification.

5.4.3: Visualization types

Visualization can take many forms for even the same data set. For example, Section 3 of Appendix G shows several visualizations for the same data. Different people may prefer different types of graphs to understand a comparison. In fact, although color is an important part of visualizations, the choice of color may hinder understanding for someone with color vision deficiency. For example, some people will confuse red and green, even if most people consider these colors to be obviously different. Another important point is the amount of data presented in one visualization.

Thus, there are two challenges here. One is to have a feedback mechanism to understand which visualizations are better than others. Another challenge is to have a mechanism that will allow users to change the visualization which could mean "switching" between different types of visualizations, changing the use of colors, or filtering the amount of information shown. Both these elements would be part of future work.

5.5: Digest of Chapter 5

This chapter describes the analysis and visualization of curriculum information where the information takes the form of knowledge and competency specifications. The basic functions (i.e., content management, reporting/presentation and analysis) that the visualization tool set will have are first described. The target stakeholders generate the visualizations through typical questions they might ask. Visualizations help answer a question that a stakeholder may have and may answer different questions from different stakeholders. The chapter also addresses some challenges needed for a workable tool set that evolves and becomes continually useful. Appendix G contains many further visualization examples.

Chapter 6: Global and Professional Considerations

This chapter addresses some of the global issues surrounding computing education. Some of these issues include the lack of a common nomenclature among different countries and regions, the different forms of degree programs across the world, and the various dynamics that can influence the ability of universities to produce competent computing graduates.

Readers should note that it is not possible to include all aspects of a topic, an opinion, a country, a person, or a region. They should consider entries as samples or examples of different situations and explore further research in such areas.

6.1: Global Context and Computing Programs

There is a need to express curricular models and recommendations in a neutral manner that is acceptable and understandable by all people. Terminology and nomenclature are one aspect of this. Another is how any recommendation reflects the cultural context on a global scale.

6.1.1: Computing 'Tower of Babel'

The Tower of Babel is a well-known biblical account where populations that were building a tower to reach heaven had their singular speech confounded so that they could no longer understand each other. In some way, this seems to be the situation regarding computing today. It is neither intended nor possible for this report to determine a single terminology applicable to computing education worldwide. This report, initially written in English, may not be the *lingua franca* (common language; literally, French language) of some countries in the world. Although this report has been written using English terminology, it does not presume to tell speakers of other languages which terminology they should use in those languages.

For example, in most countries of Europe, a word that is associated with computing degrees translates directly into English as *informatics*. In 1957 Karl Steinbuck coined the German word *Informatik* [Ste1] and subsequently, other languages adopted the word. Examples are *l'informatique* in French, *informatica* in Italian and other languages, *informática* in Spanish, and *informatikës* in Albanian. In translating these words into English, they are sometimes translated as 'computing sciences' in addition to the possible US "computing + x" terminology. However, there is an increased tendency to use the word as 'computing' as an equivalence to 'informatics' lately. The informatics family of names developed independently of, and approximately at the same time as, computer science in the United States. The term 'computer sciences' first appeared in print in 1959, and it was another three years before the appearance of the first study program called 'computer science' [Ted1]. When not used as the name of a degree, universities sometimes use informatics as the overarching name for the academic discipline such as the School of Informatics at the University of Edinburgh or the Department of Informatics at Sussex University, as well as within countries.

In Latin America, a preference exists to include the word 'engineering' in degree titles. A working group organized to study the 'computing engineering' degrees in Latin America and elsewhere concluded that it would be unreasonable to expect everyone to use a common set of program names. Instead, it developed a common set of categories to describe the content of degrees. For example, applying those categories to the systems engineering degree in Uruguay, it found that the degree had good coverage of concepts from computer science and software engineering, lower coverage of concepts from information systems and information technology, and, despite its name, weak coverage of concepts from computer engineering [Ram1]. This example should suffice to make it clear that terminology is not uniform even when translated into English.

Australasia (which incorporates Australia and New Zealand) provides another example, this time with 'information technology' as its terminology. Australasia uses information technology to refer to the whole of the academic field of computing. For example, at the time of writing, the entry to the information technology office area at Monash University in Australia bears the following message.

> "Information Technology: algorithm, distributed systems, database, software engineering, network, information systems, computation, knowledge management, analysis, mobile computing, design, e-business, model, data mining, interface, business decision support"

A comparison of these terms with those found in a typical northern American information technology degree should make it clear that the same term means very different things in these two regions.

6.1.2: Computing Nomenclature

This report takes a proactive position in attempting to normalize the use of terminology in the computing field. The public cannot view computing as a valid profession if professionals within it cannot agree on the meaning of the words it uses. Established sciences and professions such as medicine have definitive meanings regarding the words they use. This simple understanding does not exist in computing. The computing field is relatively new in comparison, and many activities have emerged independently. This mixture of terminology will only bring more confusion to the field. Fortunately, this CC2020 project through Appendix H and other literature on global interpretations of computing education terminology [Sim1] have helped to dispel such misgivings.

This report necessarily uses specific terminology. The companion reports for identified subfields of computing necessarily use specific terminology. Unfortunately, there is no universal terminology in computing or in computing education, even within the English-speaking world. To some individuals, in a specific context, terms such as computer engineering, computer science, information systems, information technology, software engineering, and informatics have reasonably clear meanings. However, to other individuals in other contexts, they (particularly informatics) can have quite different meanings, and those different meanings have just as much legitimacy.

People need to be conscious of all this variation when writing about degrees, especially within a strictly local audience or when reading about degrees that are not from one's own region. For these reasons, the terminology used in this report is for convenience. It generally aligns with the terminology used in northern America. It is not a prescription on how people and universities around the world should name their degrees, majors, or individual courses of study.

To disrupt this misunderstanding, the CC2020 task force has suggested that the public use the word '*computing*' to describe the entire field. Such an adoption will take some time to become universal. However, using the word judiciously will eventually begin a convergence to a computing profession. For example, the word 'engineering' has relatively good universal understanding. Computing should have a similar understanding.

There are many other words that require clarification. Appendix H provides an equivalence table of terminology. The table should provide some guidance in trying to understand the meaning of computing words and how people use them in a global context.

6.1.3: Cultural Sensitivity and Diversity

One should understand computing degree programs within the global contexts. Thinking that "my program" is the only way a computing program should exist can be counterproductive, especially when trying to engage in cooperation and understanding with different people. It is important to be aware that cultural similarities and differences do exist between people and the computing programs they represent.

Universal acceptance of global diversity is essential in all fields of endeavor, particularly the computing field that is so diverse. Those who may not be sensitive to cultural diversity should explore ways to acquire more knowledge on the situation. Computing graduates will likely interact with professionals on a global scale, so developing a sensitivity to global customs and traits, communicating effectively with peers, listening carefully, and being sensitive to time zones and holidays go a long way in bridging cultural sensitivity gaps. Graduates of computing programs can benefit greatly by learning more about global customs and etiquette of the people with whom they will be working.

Computing should be accessible to all people, especially those with disabilities. Assistive technology centers with a focus on computing technology are becoming more common on a worldwide scale. The goal of such centers is to make every human being be capable of living a normal life. Sensitivity to people with special needs is important. Therefore, educators should ensure that curricula and educational systems allow full inclusion of people with disabilities. They should also teach students the necessary skills, so computer systems and applications enable full inclusion as well. There are policies on these issues in most countries.

6.2: Global Economics and Computing Education

The global digital economy continues to drive job creation and sustainment. The expansion of this digital economy has resulted in substantial demand for an increased volume within the labor pipeline. This situation has resulted in labor shortages, whereby the output of baccalaureate programs has been insufficient to meet the demands of the workforce. The education community has adapted to these forces by creating alternative pathways to education and training based on shorter duration programs. Community (two-year) college programs (programs generally unique to the United States) have seen substantial increases in student enrollments.

Furthermore, the idea of micro-credentials through short-term and online programs (or self-studies) are becoming increasingly popular, as are coding "bootcamps" and "academies" devoted to focused, short-term training. The effect of these market forces on the various computing disciplines are not fully clear. Generally, these shorter-term programs have areas of emphasis that are based on "just-in-time" market situations related to the various computing disciplines. However, no standard, expected outcome, or competency exists that has universal acceptance for these short-term programs.

6.2.1: Innovation Spaces

The digital revolution has provided humanity with a plethora of new technologies that have improved people's lives. Smart phones, medical imagery, aviation and aerospace, contemporary automobiles, communication infrastructure and tools, and complex video games are just a few applications touched by computing and digital technology. Computing affects all people in some way with new and emerging technologies that are still in their infancy.

The future promises even greater expectations on the ways computing innovations will affect people's lives and computing education. Computing education must be agile enough to address the rapid changes of the field. Acceptance of a status quo attitude would quickly make such programs either obsolete or ones whose graduates would lack the necessary skills and human temperament for gainful employment. Modern curricula must change to match any increase in technological innovation.

One activity that is emerging in universities universally is the creation of makerspace laboratories, especially for engineering and business environments. Makerspaces, such as those used in New Zealand [Min1], are part of a constructivist movement that allows students, especially first-year students, to have open accessibility to readily available materials that provides exposure to modern technology, availability to items for invention and innovation, and human inquiry. Makerspaces are now emerging in elementary schools and in high schools worldwide. They modify the emphasis from teaching to learning. Computing educators could take heed in this global movement and consider creating makerspace laboratories and making their use an initial and integral part of their computing programs. Figure 6.1 illustrates two examples of makerspace laboratories; Figure 6.1 (a) shows Africa's Maker Movement by Open Air [Ope1], Figure 6.1 (b) shows a makerspace lab at Lindenwood University [Lin1].

(a) Makerspace at OpenAir-Africa's Maker Movement

(b) Makerspace Lab at Lindenwood University

Figure 6.1. Examples of Makerspaces

6.2.2: Forces Shaping Academic Programs

This document describes seven basic categories of baccalaureate computing degree programs: computer engineering, computer science, cybersecurity, data science (under development), information systems, information technology, and software engineering. Few academic institutions offer more than three of these types of programs, although that situation might change over the next decade. As was the case when CC2005 was written, universities offering baccalaureate degree programs tend to be cautious and conservative. The complex nature of academic degree programs makes it difficult to implement significant changes quickly. The COVID-19 pandemic has further complicated the ability to promote change.

Baccalaureate programs at universities usually compete for prospective computing students, sometimes within the same institution. Such external and internal academic forces might affect the quality of computing programs because some programs could lower their academic standards to enroll more students. Some institutions of higher learning even create entities within the institution (e.g., schools of continuing education) that offer abridged courses like those offered in an academic program in the institution that may not apply toward an academic degree.

Depending on the goals and content of a computing program, prospective students should make judicious choices in selecting which program best serves their aspirations. Students who are weak in mathematics might not want to undertake a computer engineering degree or one in data analytics. Students considering computer engineering or data analytics curricula should be aware of the emphasis on mathematics that these courses of study normally require. Programming skills and computer language fluency seem to be the norm in computer science and software engineering programs. The set of competencies that become optimal to any career field could span a sample of many sub-disciplines of computing.

6.2.3: Degree Names and the Workplace

Degrees are not the only type of qualification that students can attain in computing. Around the world, colleges of various sorts offer certificates, diplomas, and advanced diplomas in different aspects of computing. For example, a micro-credential can certify competence in a specific skill through evidence created by practice. In some cases, these qualifications serve to qualify their holders for entry to a traditional university degree; in others, they suffice to give their graduates direct entry into professional employment. While these qualifications are typically vocational qualifications, it would be a mistake to think of them as inferior to university degrees [Tan1].

It should suffice to mention that 'a computing degree' is not a unique qualification for employment. Degrees and other qualifications in computing are astonishingly diverse in their duration, the extent of their focus on computing, and the scope of other material studied. Degrees are diverse in their terminology with the same name used for very different learning experiences; likewise, a variety of different names can correspond to very similar degrees.

The inconsistency of naming computing degrees has reached the level that in some job markets, the terms are more confusing than descriptive of any competence that graduates can offer in the workplace. In fact, many employers tend to ignore the value of a degree name. That is, although possession of a baccalaureate degree is important, the name of degree is of little consequence. Employers are more interested in a graduate's technical skill set and the human temperament that the graduate possesses. From the viewpoint of students and prospective students, it is best for them to enter a computing program they desire and have an ability to exceling rather than studying a computing program that sounds trendy.

6.2.4: Use of the Word "Engineer"

The word "engineer" has diverse meanings in different parts of the world. In some places, it has a prestigious meaning, reputationally equal to, for example, "doctor" or other important professional. In other places, it is a just a normal expression used in a degree title or a job position. In some instances, universities unnecessarily force the word into degree titles to suggest an element of prestige, often as a scheme to attract students into their programs. In these cases, the use of "informatics engineering" or "systems engineering" may not have the program quality commensurate with its actual meaning. Hence, a university that renames a typical computer science program as an informatics engineering program would naturally attract more students because of a more appealing or more "prestigious" name. In Latin America, for example, the reason that programs have a degree name of "systems engineering" is largely due to corporate positions promoted by industry.

The use of the term "civil" can be equally confusing. Specifically, in Chile, "civil engineering" contrasts with "military engineering." The word civil refers to people or engineering for the good of people. Hence a "computer civil engineer" is really a "computer engineer" rather than a civil or construction engineer who has a computing background. These types of traditions may cause confusion and unnecessary problems especially for international understanding and processes such as student exchange and accreditation.

6.2.5: Innovation in Computing

The computing field abounds with invention and innovation. Innovation means "the process of translating an idea or invention into a good or service that creates value or for which customers will pay" [Biz2]. Innovators often combine information, imagination, and initiative. Innovative ideas should satisfy a specific need, become a benefit for society, and be economically replicable.

In a computing context, innovation helps students and professionals to create inventive ways to solve computing problems. Often innovation is a continuous process. Such dynamic innovation occurs by many incremental advances in technology or processes such as the incremental improvement of hardware or software. When computing innovation is radical or revolutionary, its application may become a disruptive technology. Examples of recent disruptive innovations include blockchain technologies and the Internet of Things.

People often believe that risk-taking is synonymous with innovation. Those who create revolutionary technologies should be ready to undertake risks. Students in baccalaureate programs may find it difficult to become innovators during their studies, although there are counterexamples to this premise such as Bill Gates and Steve Jobs. Nevertheless, faculty members should encourage possible innovators in their programs and make allowances for students who show genuine promise toward innovative careers.

6.2.6: Entrepreneurship in Computing

Entrepreneurship is becoming an important area of study, including in the field of computing. Entrepreneurship is "The capacity and willingness to develop, organize and manage a business venture along with any of its risks in order to make a profit" [Biz1]. The key element for success is the entrepreneurial spirit driving toward innovation, risk-taking, and the drive to succeed in a rapidly changing global marketplace.

An infusion of entrepreneurial experiences into computing programs is very possible. Business schools usually teach such courses. In its simplest form, faculty members could advise students to take an entrepreneurial course as a substitute for an elective course. The same action is possible with technical electives; some, although not all, students are likely to benefit more from entrepreneurial experiences and an additional technical elective. A more aggressive approach is to construct a minor or cluster for computing students in harmony with business school offerings. For example, a student taking two entrepreneurial courses, two business courses (e.g., marketing and management), and a two-course major project computing experience could suffice in establishing a formal minor experience.

In today's world, business acumen may be as important as technical computing knowledge. Faculty advisors may have opportunities to encourage students who are inclined to be risk takers to take some combination of courses for an entrepreneurial educational environment. The experience would likely benefit them throughout their lives and offer positive contributions to society.

6.3: Worldwide Computing Degree Structures

Differences in the terminology and nomenclature in computing education cause confusion because of different degrees themselves in the nature. This section provides a few samples reflecting worldwide degree structures. Establishing a complete list of such structures is beyond the scope of this report.

6.3.1: Computing Education in Africa

Computing programs presented in Africa are mostly bachelor programs in science with specializations in computer science. Older institutions tend to present a wider variety of computing programs. In many cases these programs are within the same department. Very few universities have departments dedicated to computer science and information systems (also referred to as informatics). Computer science programs typically confer "Bachelor of Science" degrees where the curriculum includes mathematics, possibly statistics and a science. Information systems results in a "Bachelor of Commerce" degree, conferred where the curriculum includes studies in economic and management sciences. When the distinction between computer science and information systems is not clear, the program results as an information technology degree. These programs may include study from disciplines outside computing. Universities presenting engineering degrees may have a program in computer engineering. A focus in software engineering tends toward inclusion within a sciences program.

Disparate degree structures exist throughout Africa. There is, therefore, a movement to formulate a mapping between these structures. The main difference is the presentation of a bachelor's program. In some countries, a bachelor's degree takes four years; in other countries, a four-year degree program is seen as a professional program. All other degree programs are three years followed by one year of honors study.

6.3.2: Computing Education in Australasia

Degrees in Australasia have hybrid names that would correspond in some way to majors in Northern America and three-year programs in Europe. Degree names such as Bachelor of Computer Science, Bachelor of Information Science, and Bachelor of Software Engineering are common. Students choose a specific degree program before beginning university study, rather than choosing a major some way into a more generic degree. Some of the degrees are tightly focused, like those described for the United Kingdom, but there are also broader degrees where perhaps one-third of the courses studied are outside computing.

For example, consider the Bachelor of Information Technology degree at the University of Newcastle [New1]. Of the twenty-four courses that make up the degree, ten are core and must be taken by all students. Students typically use their major and/or their electives to supplement their computing studies with knowledge from non-computing areas to prepare them for their future computing work in those areas.

6.3.3: Computing Education in China

According to the China National Higher Education Catalog, there are six categories related to computing. They include the following: computer science and technology, software engineering, network engineering, information security, Internet of Things engineering and digital media technology. The curriculum usually consists of general education in the areas of politics, English, and liberal arts, foundation course for mathematics, physics and electronics engineering and computing curriculum. Compulsory courses and elective courses make up all the courses for a major. The Chinese education system includes junior colleges which offer two- and three-year degrees that have lower requirements than universities. These junior colleges offer vocational programs as well as programs that allow transfer to a university.

There is a parallel classification system issued by Ministry of Education of China from the perspective of graduate level education for university evaluation. There are three first-level disciplines for computing: computer science and technology, software engineering and cyberspace security. Recently, the ministry has approved artificial intelligence to be the fourth first-level discipline. Under computer science and technology, there are six second-level subcategories: information security, software engineering, computer software and theory, computer system structure, computer application technology and computer technology. There are some other inter-discipline second-level categories such as electronic and computer engineering, information systems management, information and communication engineering, health informatics, bioinformatics, and geographic information science.

Additionally, China and its education ministry have embraced competency as an important element in the development of computing and engineering programs. Appendix I summarizes its “Blue Book” project that addresses the need for competency in university environments, particularly as it applies to computing and engineering education programs.

6.3.4: Computing Education in Europe

Degrees in the United Kingdom and parts of Europe focus on a specialist area of study from the outset. Students do not begin a general degree and subsequently choose a specialization; they enroll from the outset in a specialist degree. For example, the three-year BSc Computer Science program at Exeter University [Exe1] includes required computing and mathematics courses, a choice of optional courses, and a major project accompanied by optional experiences.

The Bologna process ensures comparability in the standards and quality of higher-education qualifications. The process has 48 participating countries. The Bologna framework [Bol1] specifies three higher education qualification cycles: bachelor’s (three years), master’s (two years) and doctoral (three years). An important part of the European approach is the framework for qualifications of the European Higher Education Area. The so-called Dublin Descriptors provide “generic statements of typical expectations of achievements and abilities associated with qualifications that represent the end of each of a Bologna cycle” [Bol2 p65] in relation to five categories: knowledge and understanding; applying knowledge and understanding; making judgments; communication skills; and learning skills. These descriptors provide discipline-independent descriptions of what each of the degree cycles require.

“Informatics for All” is a new coalition involving ACM Europe, Informatics Europe, and the Council of European Professional Informatics Societies (CEPIS). Its purpose is to promote the advancement of informatics education within Europe, primarily at the level of primary and secondary high school education. Following a survey of the state of informatics education throughout Europe, “Informatics for All” developed a two-level strategy: (1) the view of informatics must be an important foundational discipline taught to all pupils, and (2) integrate informatics into the teaching of other disciplines in a manner leading to a deeper form of education in those other disciplines. These activities are gaining much support within Europe.

6.3.5: Computing Education in India

In India, the University Grants Commission (UGC) mainly regulates education in India [Ind1], and it defines the framework within which universities operate, including the names of degrees that they may award. The university system structured contains two levels: the university itself, and a set of colleges affiliated with it. The university determines the curriculum and assessment of most degree programs conducted at affiliated colleges with the colleges serving as the “delivery” mechanism. There are mainly two broad strategies followed in designing degree programs for bachelor level computing education in India. Four-year programs are the norm in computer engineering (CE) and information technology (IT) degree programs. The first year is devoted to physical sciences, mathematics and statistics for engineering. Most courses are common between CE and IT with business courses forming the main difference that distinguishes IT from CE. On the other hand, three-year programs are the norm in computer science and computer application programs, with some institutions offering an additional fourth year of study typically called an "honours" program. Most three-year computer science programs focus on applied aspects. Further, the master's in computer applications program is also typically a three-year program since the university envisions it as the first program for students who have a bachelor's degree from other streams. Therefore, although named as a "master's program." it often becomes a first-level degree in computing.

India operates 895 universities, 42,338 colleges, and 3,225 engineering institutes [Ind1, Ind3]. Table 6.1 illustrates enrollment figures for three categories of study.

Table 6.1. Enrollment data for university studies in India (2017)

	Male (x100,000)	Female (x100,000)	Total (x100,000)
UGC (All inclusive)	15.27	14.16	29.43
UGC (Non-Engineering CS)	Not Available	Not Available	9.68
AICTE	5.30	2.20	7.50

The National Assessment and Accreditation Council (NAAC) [Ind2] is an autonomous institution under the UGC that is responsible for quality assurance of higher education institutions in India. Additionally, the All India Council for Technical Education (AICTE) regulates technical streams like technology, engineering and pharmacy [Ind3]. The National Board of Accreditation (NBA), which is an autonomous institution with the AICTE, promotes international standards of technical education in India [Ind4].

6.3.6: Computing Education in Japan

In Japan, computing related bachelor's degree programs are of two types: those that focus on computing such as computer science, and those whose primary focus are in other fields. Most of the former type come under the broad umbrella of either a Bachelor of Engineering degree or Bachelor of Science degree. Some universities have more specific names, such as a Bachelor of Informatics or a Bachelor of Computer Engineering [Bac1]. For those of the second type, the degree name may be, for example, Bachelor of Business and Informatics, but the actual focus of the degree may be in fields such as business and design. In those cases, computing (informatics) would be a comparatively minor part of the degree. This situation also appears in a published survey [Kak1], where they found that nearly half of the students in a "computing" department are learning computing domains other than those defined in CC2005. Such students belong to interdisciplinary departments such as a department focusing on business with a computing component.

Even for the first type degree, there is a wide difference between universities on how students achieve a degree, especially at the beginning. At some universities, students start with a basic set of courses such as physics, chemistry, mathematics, and informatics in their first year of study. They then begin their actual major in their second year. At other universities, students will start their actual major in their first year. Computer programming, which is a basic requirement for any computing related degree, accentuates this difference; some universities have their first programming course in the student's first year while others have it in the second year.

6.3.7: Computing Education in the Middle East

The Middle East and North Africa (MENA) is a complex region consisting of twenty countries with a population of almost 600 million people. Most universities of the MENA countries follow the ACM/IEEE computing curricular guidelines. For example, since the mid-2000s, most countries have followed the Curriculum Guidelines for Undergraduate Degree Programs in Information Technology, known as IT2017 [Acm07] and formerly IT2008 [Lun1]. These reports recommend computing areas beyond programming and provide the potential to conduct projects, internships, and research together with an emphasis on components to enhance the practical experience of students. Such degree programs also foster adaptability to change in job market needs by providing in-depth knowledge through specific concentrations that are easily interchangeable. Hence, respected IT programs in the Middle East have enjoyed success with these principles, and they serve as models for IT programs in the region.

Computing in the MENA region is important for development and modern technology. The region needs computing specialists due to the penetration of computers in all aspects of life. In response to the reality of the projected need for national competence in the field of computing, countries are creating futuristic and specialized academic computing programs. For example, in Saudi Arabia, the futuristic approach has generated innovative programs in computer engineering and computer science with great demand and it has led to three additional programs in information systems, software engineering and information technology. Several universities in the country are following these innovations and the concept is spreading to other universities in the region. As another example, for more than twenty years, high schools in Israel have taught computer science just like physics, biology, and chemistry. In recent years, elementary and junior high schools have introduced computer science in an extended piloting stage. For non-computing teachers, the government has established a teachers' center, so no teacher feels isolated. The center serves as a vehicle where teachers can contact their colleagues, find materials, and receive invitations to attend workshops and conferences.

Many computing programs in the MENA region also seek accreditation from Western-type agencies such as ABET in addition to local or governmental accrediting agencies. Furthermore, many MENA countries have started efforts to make computing a core or compulsory course in the secondary schools. Such courses would cover fundamental topics in the computing field, an introduction to programming, as well as technologies and programing for smart devices. Some countries have begun to implement digital transformation plans to become an educational component for intermediate and elementary schools to produce a technologically advanced generation.

6.3.8: Computing Education in Latin America

In Latin America, a typical degree requires a duration from four to five years, with some extra content devoted to general subject matter such as literature, writing, mathematics, logic, and other related subjects. The subsequent years are more focused on computing topics. Additionally, degrees in Latin America have several hybrid names mostly oriented on teaching ways to use technology. Students in Latin America choose a degree path before beginning university studies. Most of the tightly focused degrees begin computing with the very first semester. The best example following international recommendations is Brazil where clear groups of well-defined programs are offered. Almost all computing programs orient themselves to guidelines presented in computing curricula from ACM and IEEE-CS. Furthermore, because of historical reasons, degrees in Chile distinguish between civil (people) engineers and military engineers. Similar situations occur in Peru, Colombia, Equator, and Venezuela. Decades ago, IBM had influenced the early computing programs with the degree designation for computing programs called "systems engineering" according to IBM's job position, a mistake that continues even today after several decades. Mexico has also worked to reduce the nomenclature of its programs to computer engineering, computer science, information systems, and software engineering, which people often call "informatica" in the region.

6.3.9: Computing Education in North America

Two-year degrees and four-year degrees encompass all possible degree structures in northern America. Most four-year degrees in northern America (Canada and the United States) have the designation of Bachelor of Science (BS) degree, Bachelor of Arts (BA) degree, Bachelor of Engineering (BE) degree, or other baccalaureate degree descriptors. Computing topics within a community college program should be equivalent to at least one full year of study in the four-year program of study. It also includes relevant mathematics and science as other important components of a computing program. Additional program requirements, often called general education requirements, depend on the characteristics and mission of the program and the institution. Almost half the undergraduates in the United States are enrolled in two-year colleges and more than half of all first-time college first-year students attend community and technical colleges. Students in two-year community college programs often earn associate degrees [Aac1]. Articulation agreements often exist between institutions of the two-year programs and the four-year programs to facilitate seamless transfer from two-year programs to four-year programs.

6.3.10: Computing Education in the United Kingdom

In the United Kingdom, computing is well embedded in primary and secondary school education, with mandatory standards set at country level. In England, the UK Government's Department for Education defines these. In Wales and Northern Ireland, the local governments set the standards, but they are broadly comparable to England. In Scotland, the government specifies the Curriculum for Excellence (CfE) covering primary and secondary education. Education Scotland defines the four-year CfE taken by all later year primary and early year secondary students. Computing and cognate disciplines are available in all UK universities, which are overwhelmingly public bodies. The independent Quality Assurance Agency (QAA) for higher education specifies post-school national benchmarks for computing curricula. For example, the computer science benchmark [6] explicitly follows the ACM/IEEE curricula. The British Computer Society (BCS) accredits UK computer science programs; its guidelines follow the QAA statement [Bri1]. See [Com1], [Com2], [Dep1], and [Edu2] for more comprehensive overviews of the national computing curricula.

6.4: Professionalism and Ethics

Professionalism and ethics should be a permanent element of any computing curriculum. The following discussion taken from the IT2017 report sheds some light on ways these elements can be part of a computing program of study.

6.4.1: Ethics in the Curriculum

The incorporation of professionalism and ethics must be a conscious and proactive effort in the context of every computing program because much of the material blends into the fabric of existing curricula. For example, the introductory courses in the major could include discussion and assignments on the impact of computing and the internet on society and the importance of professional practice. As students progress into their second-year courses, they could start to keep records of their work as a professional might in the form of requirements, design, test documents, and project documents such as charters and project reports.

Additional material such as computer history, digital libraries, techniques for tackling ill-defined problems, teamwork with individual accountability, real-life ethical issues, professional standards and guidelines, legal constraints and requirements, and the philosophical basis for ethical arguments may also appear either in a dedicated course or distributed throughout the curriculum. The distributed approach has the advantage of presenting this material in the context of a real application area. On the other hand, the distributed approach can be problematic in that faculty often minimize professionalism and ethics in the scramble to find adequate time for the technical material. Projects, however, may provide a natural outlet for much of this material particularly if faculty can recruit external clients needing non-critical systems. When they engage in service-learning projects in the community or work with external clients, students begin to see the necessity for ethical behavior in very different terms. As a result, those students learn much more about ways to meet the needs of a client's ill-defined problem. However, no matter how teachers integrate professional practice into the curriculum, it is critical that they reinforce this material with appropriate assessments.

For departments with adequate numbers of faculty members and resources, courses dedicated to teaching professional practice may be appropriate. For limited resources, this content should be covered in courses like professional practice, ethics, and computer law, as well as senior capstone and other appropriate courses. Additionally, more advanced courses on project management, financial management, quality, safety, and security may be part of the experience. These courses could come from disciplines outside of information technology and they would still have a profound effect on the professional development of students.

6.4.2: Professional and Ethical Work

Learning environments that support students in acquiring professional practice experiences include the following elements [Acm07]:

- Assessments.
- Appropriate inclusion of professional practice in traditional course assessments (assignments, projects, exams, presentations, reports, etc.)
- Sound measurements of student work to show student progress and improvement.
- Student involvement in the review and assessment process.
- Participation of professionals from industry, government, or other employers of IT graduates to assess student performance in internships, co-op programs, and on projects with outside clients.
- Standardized tests validated by professional societies.
- Post-graduation alumni surveys of alumni to see how well alumni thought their education prepared them for their careers.
- Program accreditation to demonstrate compliance with certain educational standards for professional practice.
- Course labs that meet employer needs to make sure students acquire professional experiences.

The assessment process should encourage students to employ good technical practice and high standards of integrity and ethics. The assessment process should hold students accountable on an individual basis even if they work collectively in a team. It should have a consistent set of measurements, so students become accustomed to using them and they learn how to associate them with their progress.

6.5: Digest of Chapter 6

This chapter encourages readers to be aware of the wealth of contexts in which computing education takes place around the world. It looks briefly at some of the terminology, establishing that the same word can be used to mean different things, and that different words can be used to mean the same thing. The report proposes that stakeholders accept the word ‘computing’ as the name of the overarching discipline. Furthermore, there is no such thing as a standard degree structure, and this chapter elaborates on many differences between countries and regions. The extent to which educational needs might be driven by economic needs and the universities’ responses to the latter is also discussed here. This chapter makes the case that ethics and professionalism must be explicitly incorporated into the computing curriculum.

Chapter 7: Curricular Design – Challenges and Opportunities

This chapter highlights some of the contemporary challenges for the development of modern computing programs. It also addresses ways in which industry and government can play a special role in generating modern programs through professional advisory boards, work-study programs, and internships. Academic institutions must also be proactive in supporting strong, contemporary computing programs for the benefit of its graduates.

7.1: Transforming to Competencies

The CC2020 project has provided an overview of the computing education landscape related to undergraduate, (baccalaureate) programs. This overview is global in scope. Furthermore, the task force has encompassed many perspectives with the goal of providing a modern update to its predecessor, CC2005. The CC2020 project has also presented possible frameworks for future curricular reports.

7.1.1: Distinguishing Competency from Knowledge

The central theme of this report is that competency should be the standard for describing computing curricula. Whether intentionally or not, the tradition of knowledge-focused descriptions of curricula have over-emphasized information for information's sake. Competency composes an expanded perspective on education that augments knowledge (knowing what) with its skilled application (knowing how) motivated by purpose (knowing why) to accomplish a task, an outcome of value. This expanded perspective elevates the aspect of student learning in education in order to align the graduate's capacity to act effectively, proficiently and ethically as a professional practitioner. Triangulating curriculum on competency (what, how, why) reorients education to enfold the effective and ethical use of knowledge not only for the student but also in service to the welfare of society as a whole.

Knowledge (as in the body of knowledge) is no less important in the success of education but it is the conscious treatment of that knowledge in performing professional activities that yield valuable benefits to all forms of communities that sets competency-based curriculum description apart. Skills define knowledge applied in relevant situations, environments, with a particular level of proficiency requisite to successful practice. Success is knowledge skilfully applied characterized by dispositions that instil the practitioner's actions with value. The intertwining dimensions of competency (knowledge, skills, and dispositions) offer a comprehensive vocabulary with which to describe a curriculum that enfolds the objectives of learning natural to each the teacher, the student, and the respective profession that the educational enterprise aspires to serve.

In this sense, the CC2020 project and this report encourage computing programs to establish a proper environment and to necessitate that future curricular reports incorporate competency as part of their structure and recommendations.

7.1.2: Curricular Dynamics

Computing curricula are always in a state of flux. The continually changing field of computing is dynamic with new ideas and inventions developing almost daily. Hence, computing curricula must be agile and be able to respond to change. Students and graduates of computing programs must be able to face change and become inventive in contributing to that change.

One way to address this challenge is to include experiences in innovation, entrepreneurship, and maker-space activities in computing programs. While foundation or core courses are important, what might be equally important is to have students experience new technologies, inventive creations, and even space to ponder with their own imagination. Engineering disciplines have been doing this for some time with their introduction to engineering exploration laboratories in the very first semester of study. Non-engineering computing programs are just experimenting with this proven idea. Computing programs should have a solid conceptual foundation and be responsive to meet the challenge of developing modern and futuristic student experiences if they expect the graduates of their programs to succeed in a quickly changing world of computing. Regular accreditation of programs including all practical subjects where practical skills are developed, is recommended.

7.1.3: Conveying Computing Competencies

The role of academics and the way they enable computing competencies is important to produce capable graduates of computing programs. As discussed in Chapter 4, computing competency is a triad of computing knowledge, skill, and disposition. There is no single method to enable competency which is a combination of these three triadic elements. The goal is to produce graduates of computing programs that are proficient at the time of graduation to enter the workplace, to attend graduate school, or to contribute constructively in some way to society. The following discussion provides some suggestions in transferring competency to students, building on the brief description in Section 4.3.2.

7.1.4: Knowledge Transfer

The transfer of knowledge is the cornerstone of academia and universities. Academicians have been transferring knowledge to students for centuries and millennia. They do this through personal knowledge, use of textbooks, personal notes, and other mechanisms for knowledge transfer. They administer tests, examinations or other assessment instruments to ascertain whether students have acquired the requisite knowledge. This is a classical mode for student learning.

In today's world, students can acquire knowledge on a subject via non-classical methods. One obvious method is using the internet to search for supportive or extra material on a topic such as video clips, wikis, experiences on professional development, MOOCs, and other supportive online materials available to the public. Encouraging students to explore such additional materials helps them develop lifelong learning skills when students must continue to learn long after they graduate from their computing programs.

There are many other contemporary ways of knowledge transfer than the traditional lecture mode. They could encourage students to learn in small groups (e.g., pair learning), construct learning groups (e.g., teams) of three or four students, and other learning strategies. Exploring with new methods of learning can augment the learning of knowledge and allow students to interact with each other of develop new skill sets as well as developing communication and teamwork skills by studying with others.

7.1.5: Skill Transfer

While academicians have a mastery of knowledge, they may not always understand the specific skills required. University professors often assume students will develop skills on their own and without direction.

All computing academic units and departments should specify a set of skills all students should expect to master by the time of graduation. Due to the uniqueness of individual computing programs, it is not possible to specify how their academic units would implement the development of such skills. Notwithstanding, computing academicians and academic units should provide instruction on ways students would develop these skills as an important element of computing competency.

7.1.6: Disposition Transfer

Educators often lack understanding of dispositions or ways to convey dispositions in computing programs. This is understandable because such understanding may not have been part of their own education. Chapter 4 identified eleven dispositions which include: Proactive, Self-directed, Passionate, Purpose-driven, Professional, Responsible, Adaptable, Collaborative, Responsive, Meticulous, and Inventive. The set of all dispositions may exceed these eleven.

Because of the uniqueness of individual computing programs, it may not be possible to specify how their academic units would develop these dispositions. Notwithstanding, computing academic units (e.g., departments) should specify the set of human behaviors (which is broader than the eleven dispositions stated) expected of their students by the time of graduation. Computing academicians and academic units could provide instruction on ways students acquire these traits as an important element of computing competency. They may do so by personal and peer examples, by viewing workplace attitudes, or by attending seminars on behavior as professionals.

Students could take courses offered in other academic units such as in social science and psychology that could be useful in developing dispositions. Although courses in these areas may cover conceptually (at a knowledge level) topics related to these dispositions, it does not mean that they help students develop them. Additionally, some of dispositions may not transfer well across contexts; a disposition demonstrated in one class may not transfer to the authentic context in another. Only through repeated practice across domains will students learn. Furthermore, people learn dispositions through modeling and enculturation. Institutions purposely need to build and develop these traits over time through "collaborative" or "responsive" activities, course experiences, internships, and other interactive experiences.

7.1.7: Need for Local Adaptation

The task of designing any curriculum is a difficult one, in part because so much depends on the characteristics of an individual institution and the interests and expertise of its faculty members. Even if every institution could agree on a common set of knowledge, skills, and dispositions for undergraduate education, many additional factors would influence curriculum design. These factors include the following.

- *Type of institution and the expectations for its degree programs:* Institutions vary in mission, structure, and scope of undergraduate degree requirements. A curriculum that works well at a small college in one country may be completely inappropriate for a research university elsewhere in the world.
- *Range of postgraduate options that students pursue:* An institution whose primary purpose is to prepare a skilled workforce for a profession is likely to have different curricular goals than one seeking to prepare students for research and graduate study. Each individual school must ensure that the curriculum it offers allows students the necessary preparation for their eventual academic and career paths including those outside their current interest.
- *Preparation and background of entering students:* Students at different institutions—and often within a single institution—vary substantially in their level of preparation. because of this, computing departments often need to tailor their introductory offerings so that they meet the needs of their students.
- *Faculty resources:* The number of faculty members supporting a computing program may vary from fewer than five to a hundred or more at a large research university. Program size heavily influences the flexibility and options available to a program. Independent of the program size, faculty members need to set priorities for ways in which they will use their limited resources.
- *Interests and expertise of the faculty:* Individual curricula often vary due to the specific interests and knowledge base of the department.

Creating a workable curriculum requires finding an appropriate balance among these factors, a balance which will require different choices at every institution. No single curriculum can work for everyone. Every college and university will need to consider the various models proposed in this document and design an implementation that meets the needs of their environment.

7.2: Industry Engagement for Workplace Competencies

An important way that industry can support the education process is to play a greater role in helping students. These professionals can offer support in several ways such as:

- Provide faculty members with the tools and insights to develop student competencies in the subjects they teach.
- Function in the role of mentors to students working on projects.
- Give special presentations to classes telling students and faculty about their firm, their work, and their development processes.

- Take part-time positions as part-time instructors to strengthen a university's course offerings by conveying material through a practical approach.
- Conduct site visits.
- Provide in-house training materials and/or classes to faculty and students in specialized research, process, or software tool areas.
- Explore industry-sponsored capstone experiences.
- Serve on industrial advisory boards, which service that allows them to provide valuable feedback to the department and institution about the strengths and weaknesses of the students. In each of these ways, enterprises in the private and public sectors can establish important lines of communication with the educational institutions that provide them with their future employees.

In addition to the various opportunities that take place on campus, industry and government also contribute to the development of strong professional practice by bringing students and faculty into environments outside of academia. For example, students and faculty may take field trips to local firms and begin to establish better relationships.

For faculty, their cooperation with industry and government can serve as a vehicle for developing student competencies in their courses. Such connections also provide opportunities for mutual benefit and they create a higher level of trust between the faculty member and the company. Because of these initiatives, employers, students, and faculty know more about each other and are more willing to promote the program.

Over a longer term, cooperative, practicum, and internship opportunities give students a better understanding of what life on the job will be like. In addition, students may become more interested in their studies and use that renewed interest to increase their market potential. Students may also form a bond with specific employers and be more likely to return to that firm after graduation.

7.2.1: Professional Advisory Boards

Based on the experience of the Task Force it is our recommendation that professional or industrial advisory boards are essential for the development of strong and meaningful computing programs. Professionals from industry and government are a great resource for insight on the needs of the workplace. These groups can become strong catalysts for bridging the computing program to needs of industry and government. They also establish personal connection between the computing program, its students, and the professional world.

Therefore, every computing program should have a professional advisory board. Ideally, advisory boards should meet once each semester, but an annual meeting is also sufficient. Its chairperson should not be a faculty member from the program. It is important to capture board activities by taking minutes of all meetings. Updates by email or other electronic media are also possible. A professional advisory board should also monitor the goals of the computing program to ensure they are in harmony and in balance with the mission of the institution and the requirements of the workplace.

7.2.2: Work-Study and Cooperative Programs

All computing programs should consider the possibilities of including a work-study or cooperative (co-op) program as part of their curricula. Typically, these programs allow students to enter industry or government before they graduate. The experience could be one or two semesters when the student is academically mature, usually during the third year of a four-year program. These programs often provide student credit and they also allow students to earn wages as they contribute to the company or government. Some universities make cooperative experiences a requirement and they tailor their sequence of courses accordingly.

Cooperative programs do have their challenges, despite their benefits. One challenge is that students will likely graduate beyond the normal period (e.g., four years). Those students who undertake a one-semester cooperative experience might lose two semesters of time if the program does not offer required courses every semester. Each computing program should evaluate whether a cooperative work-study program is suitable for its needs and for the benefit of its students.

7.2.3: Internship Programs

In contrast to cooperative programs that usually last an entire semester, programs should seriously consider internship programs as a required component of the computing curriculum. Internships are experiences that take place over a short period such as during a summer when regular classes are not in session. Internships could also be part-time experiences: in this case, students could join a company one day a week or two half-days a week during a semester. Internships are rather popular and many computing programs around the world require them as part of student learning. Students usually receive credit for an internship and in most cases, industry pays students for their services.

7.3: Institutional Resource Requirements

This report and the related curricular volumes provide significant resources for colleges and universities seeking to develop or improve their undergraduate programs. Implementing a curriculum successfully, however, requires each institution to consider broad strategic and tactical issues. This section enumerates some of these issues and illustrates ways to address these issues.

7.3.1: Attracting and Retaining Academic Educators

One of the most daunting challenges that computing departments face is the problem of attracting, and then retaining, qualified faculty members. In computing, there are sometimes more advertised positions than the number of highly qualified candidates. The shortage of faculty applicants, coupled with the fact that computing people command high salaries outside academia, make it difficult to attract and retain faculty members. Institutions should develop aggressive plans to both recruit and retain faculty members; incentives such as hiring packages and modified teaching responsibilities may prove advantageous for this endeavor. Additionally, active participation in professional organizations provides networking opportunities with leaders of peer programs, which, in turn, may result in greater visibility and access to potential faculty candidates. Other possible strategies include collaborative and/or interdisciplinary efforts with other programs and/or institutions.

While a computing program may draw on faculty from related disciplines, as a professional field there must be a core faculty with appropriate professional training and experiences. Additionally, faculty members must maintain currency with developments in the field. Institutions must make appropriate accommodations for the professional development of faculty, whether achieved through research, conference participation, sabbaticals (perhaps in industry), consulting, or other activities. Institutions must also recognize, respect, and reward teaching faculty members in the same way it does for research faculty members.

7.3.2: Need for Adequate Laboratory Resources

It is important for educational institutions to recognize that the financial resources required to support computing programs are significant. Software acquisition and maintenance can represent a substantial fraction of the overall cost of computing, particularly if one includes the development costs of courseware. Acquisition and maintenance of the hardware and instrumentation infrastructure required for experimentation and hands-on system development by students can be costly. Providing adequate support staff to maintain the laboratory facilities as well as technical assistants and tutoring support represent other expenses. To be successful, computing programs must receive adequate funding to support the laboratory needs of both faculty and students and to provide an atmosphere conducive to learning.

Because of rapid changes in technology, computer hardware generally becomes obsolete long before it ceases to function. The useful lifetime of computer systems, particularly those used to support advanced laboratories and state-of-the-art software tools, may be as little as two or three years. Planning and budgeting for regular updating and replacement of computer systems is essential. Computing curricula typically include many required laboratories. The laboratory component leads to an increased need for staff to assist in both the development of materials and the teaching of laboratory sections. This development will add to the academic support costs of high-quality computing programs. Close contacts with relevant industries can lead to the ready availability of interesting and up-to-date case study material; it also offers opportunities for students to engage in internships. Refreshing laboratory material on a regular basis serves to motivate and excite new students.

Finally, with the availability of up-to-date reference materials on the internet, institutions should provide access to such resources as the IEEE Xplore Digital Library, the ACM Digital Library, and the AIS e-library. Webinars, e-books, online tutorials, MOOCs, and other resources are all increasingly available and relevant; these are available through, for instance, the ACM Learning Center.

7.4: Program Quality Assurance and Accreditation

Academic accreditation is a process used to support continuous improvement of institutions and their degree programs. Accredited degree programs must meet certain external requirements to increase the level of confidence the public has in them.

7.4.1: Accreditation Overview

Accreditation can occur at different levels of an academic institution. In these cases, institution-wide accreditations certify that a university meets minimum standards for resources (e.g., laboratories or libraries) and operating procedures (e.g., admissions policies) required of any legitimate institution of higher learning. Similar guidelines may exist for an administrative unit within the institution (e.g., a business school) that encompasses degree programs in related fields. Accreditation for an academic unit that houses a group of programs models institutional accreditation but with greater specificity.

The most detailed form of accreditation concerns the evaluation of individual degree programs. This involves the participation of independent organizations or government agencies that establish quality standards and criteria for degree programs in a specific discipline. Program-specific accreditation involves an evaluation of specific degree programs and certifies that a degree program meets established criteria and has rigorous processes for ongoing improvement. Accreditation does not exist for every discipline, but it does exist for computing degree programs. The accreditation as well as the standard accreditation criteria normally includes such other aspects as student satisfaction, facilities to offer the program and quality assurance procedures.

In nations where accreditation can occur at different levels, an organization (e.g., a government-related entity) may accredit a university, but not its computing degree programs. For example, a university in the United States may have unaccredited degree programs even though the university itself has been accredited. A different organization would conduct accreditation of computing programs. The distinction to keep in mind is that the accreditation of a college or university *does not imply* that its computing degree programs meet the standards of quality established for the computing disciplines unless the computing programs have a program-specific computing accreditation.

7.4.2: Benefits of Program-Specific Accreditation

Discipline-specific or program-specific accreditation provides two important benefits for programs and for the institutions in which they reside. These include the following.

- It certifies that a degree program meets minimum quality standards established by independent professional or scientific societies or by government agencies. This helps an institution market its programs, and it gives the public and prospective students reason to be confident in a degree program's quality.
- The program receives an onsite consultation by a visiting team that provides expert opinions about a program's strengths and weaknesses and about its specific needs for improvement. This interaction helps an

institution have full understanding of how its programs are performing and what institutions must do to improve their quality.

Thus, accreditation provides the benefits of both a marketing aid for attracting students and an expert consultation focused on improving quality. Notwithstanding, some institutions may not need or desire the former benefit.

Some institutions commit to accreditation solely because the accreditation process helps them maintain and improve the quality of their programs which, in turn, further cements their reputation. In some nations, institutions have no choice because accreditation is a requirement for program existence. Discipline-specific accreditation processes determine whether a candidate degree program meets certain criteria. Not only does accreditation determine whether the program provides enough qualified teachers with acceptable workloads, it also determines how the program uses materials and assignments, how it evaluates assignments and examinations, and how it engages itself in continuous evaluation and improvement.

Professional bodies also use program accreditation to ensure that degree programs meet, at least in part, the requirements for membership in their profession. In some cases, graduation from an accredited degree program is a requirement for individuals before they can practice in a profession. This means that it is not sufficient for students who wish to practice a profession simply to earn a degree in the appropriate discipline; rather, they must have earned that degree from an accredited degree program. A given degree program does not choose whether its accreditation status has such professional elements; instead, the accreditation process determines what is customary for its discipline in its nation.

Perhaps the greatest misconception about accreditation is the belief that institutions pursue program accreditation only to obtain a credential for public image. Those unfamiliar with discipline-specific accreditation often do not understand the important role that the accreditation process plays in helping a program know what it must do to improve the quality of both its offerings and its graduates.

7.4.3: Quality Assurance

Program-specific accreditation is a means of demonstrating that a degree program meets an independent standard of quality, but the meaning of that standard varies. Its rigor is determined by the accrediting body's policies and practices and by any government regulations that might apply. In some cases, accreditation certifies that a degree program has met a minimum quality standard. In other cases, there exist both minimum standards and higher standards.

While discipline-specific accreditation addresses program quality, it is important not to reach unwarranted conclusions about the relationship between accreditation and quality. One must be familiar with both the discipline and the national context in order to reach appropriate conclusions. Lack of program accreditation does not mean a program is of low quality. Conversely, an accredited program does not mean a program is of high quality. All accredited programs must meet minimal requirements according to a given set of criteria; there is no ranking according to quality.

Notwithstanding, there are several aspects that reflect high quality. These include good teachers, a faculty workload that permits teachers to focus adequately on their classes and remain current in their field, as well as sufficient faculty support and infrastructure. Additionally, it is important to have evidence of rigorous procedures for monitoring and improving quality in an ongoing way.

For strong programs, integrating quality monitoring processes with initiatives for improving quality should form a continuous cycle. Activities include monitoring effort for effect, planning and implementing new improvement efforts, and evaluating the results; the cycle then repeats. Doing this properly is not difficult. However, it requires a measure of commitment, discipline, and information sharing to be successful.

7.4.4: Global Recognition

Many countries have embraced accreditation. Although the details vary, there is a common belief that a panel of experts who represent a profession evaluates a program's quality against established standards and criteria produces strong computing programs. The circumstances vary with respect to whether accreditation is mandatory, strongly encouraged, or completely voluntary. Some countries have rigorous program criteria and require that accreditation standards apply to every program offered at any college or university. In other countries, accreditation is voluntary.

The administration of the accreditation process also varies. In some countries (e.g., Australia, Canada, and the U.K.), professional societies conduct program accreditation for their respective fields. In other countries (e.g., the U.S.) a designated organization monitors and/or performs accreditation. In some countries (e.g., Estonia and the United Arab Emirates), a government agency conducts the accreditation process.

In some computing disciplines, accreditation agencies also cooperate across their national borders. Mutual recognition of evaluation and accreditation processes has encouraged a range of international agreements such as the Washington Accord for engineering programs, the Seoul Accord for computing programs, the Sydney Accord and the Dublin Accord for technology programs. Other accords include the European Federation of National Engineering Associations (FEANI) and the International Register of Professional Engineers (IRPE). Such agreements have a range of signatories, but they share a common goal to facilitate the movement of professionals across nations. That is, they recognize the substantial equivalence of programs accredited by these bodies. For example, the Australian Computer Society (ACS) accredits computing programs in Australia; ABET accredits computing programs in the United States and elsewhere. Graduates from ACS accredited programs and graduates from ABET accredited programs enjoy a mutual recognition for employment and other professional benefits.

Accreditation in the U.S. is voluntary in the sense that no law or regulation requires a degree program to acquire accreditation. As a practical matter, it is more voluntary in some computing disciplines than in others. In computer engineering, for example, a strong sense of a professional community exists, and state-regulated licensing of engineers can require applicants to hold an engineering degree from an ABET accredited program. In contrast, the computing community outside engineering is more of a loosely organized network of scientists and researchers than a tightly organized body of practicing professionals. Historically, there has been no compelling professional pressure for accreditation of non-engineering computing programs.

7.5: Degree Names and Job Titles

Differences in computing degree names and job titles can lead to confusion. The concept of licensure can be similarly affected.

7.5.1: Job Positions and Titles

In today's world, there is often confusion regarding what a person does and what a job position entails. For example, the phrase "software engineer" is a generic name used by many to identify someone who creates or develops software. That person may be a mathematician, a physician, a civil engineer, or even a practitioner without any specific university degree or title. In English, the normal phrase refers to the people that occupy a job position. In other languages such as in Spanish, it is common to address people based on the title they received when they finished their undergraduate studies. As an example, in English there may be an announcement that "Company X seeks to hire three-thousand software engineers" to fulfill a large government contract. In this case, the reference is for people prepared to create or develop software, regardless the degree title they obtained. In this case, it would be a mistake to assume that Company X is looking for people with a specific software engineering university degree title.

In the case of licensure, this is the case where a government agency allows an individual the authority to do something or to practice a profession. For example, for the privilege to drive an automobile, almost all developed countries require a driver's license. To practice medicine, dentistry, nursing, or law, a government agency requires a person to have a license to practice a profession. For professions such as medicine, this licensure occurs after a person completes formal studies and attains a degree such as a medical degree.

Some universities in different parts of the world issue a "licentiate" degree. It refers to a degree below a doctoral degree. Terms such as "licentia docendi" refer to permission or license to teach; the term "licentia ad practicandum" refers to someone having permission (license) for professional practice. The use of "license" can create much confusion in computing, and it is best to avoid the term except when issued for licensed practice such as a licensed professional engineer such as a licensed computer engineer or a licensed software engineer. Such professional licensure requires stringent legal regulations involving formal examinations, acquired university degrees, and years of professional practice.

7.6: Technology Trends for CC2020

This section addresses current and emerging technologies heavily dependent on computing and increasingly strongly integrated with a broad variety of types of human activity.

7.6.1: Current and Emerging Technologies

Current and emerging technologies have a potential of affecting society very significantly such as the way in which we communicate or interact with each other, conduct commercial activities, organize decision making, and the way people learn. Some well-established technologies exist as independent areas of study within computing with their own curriculum recommendations (e.g., cybersecurity, data science) while others are barely out of research laboratories (e.g., DNA computing).

The curricula for "cybersecurity" and "data science" are already specialized areas of study closely affiliated with computing that either already have their own curriculum reports (CSEC 2017) or have one in preparation [Dat2]. Both are essential areas of focus by these two areas.

The technologies addressed in this section reflect trends and reports by major global technology consultancies and the World Economic Forum (WEF). They include Accenture [Acc1], Deloitte [Del1], Gartner [Gar1], KPMG [Kpm1], and the WEF [Wef1].

7.6.2: Existing Computing Areas with No Endorsed Curriculum

The three areas of computing-driven systems and technology infrastructure discussed in this section are already in existence, but they have not yet reached the status of an academic discipline with a globally recognized curriculum. Curricula may exist within a region or country, but not endorsed by recognized institutions such as ACM or IEEE.

Internet of Things (or IoT) refers to a "system of interconnections between digital technologies and physical objects that enable such (traditionally mundane) objects to exhibit computing properties and interact with one another with or without human intervention" [Bai1]. IoT technologies give physical objects capabilities that allow them to measure and communicate their states to other similar objects and provide centralized data collection mechanisms to enable coordinated data-driven action. Some countries such as China have robust IoT degree programs at the undergraduate and graduate levels.

Cloud computing refers to the practice of offering computing capabilities (particularly data storage and processing power) on the internet or other shared networks as a service, typically charged based on usage and managed by the service provider. Cloud computing essentially implements the idea of providing computing power and storage capabilities (infrastructure as a service), infrastructure integrated with platform services (platform as a service), or applications (system as a service) as a commodity service.

Narrow artificial intelligence, also known as weak AI, supports specific tasks in a narrow, well-defined context. Weak AI already exists on a broad variety of systems to enable and support human decision making. General artificial intelligence (strong AI) and artificial super intelligence (forms of AI that mimic and generally exceed human capabilities) do not currently exist. However, the area is already under fierce debate from ethical and moral perspectives.

7.6.3: Emerging Computing Areas

Digital experience refers to the practice of providing various organizational stakeholders (such as customers) a personalized and consistent set of experiences across a range of different digital platforms from small form factor wearable devices to large workstations and across a variety of situations. The terms used to describe this set of technologies include digital experiences (as used by Accenture to refer to augmented reality with 5G), multiexperience (as used by Gartner to refer to multiple channels for interacting with the digital world), and digital experience (as defined by Deloitte as human experience platforms), such as customized, emotionally intelligent digital experiences based on individuals' behaviors, preferences, and emotions using an integrated array of AI capabilities. Other innovative computing areas include distributed ledger technology, artificial intelligence, extended reality, and quantum computing (DARQ) as Accenture's "key set of new tech" as well as digital reality (identified by Deloitte as one of its macro forces).

In addition, the area of *interactive technologies* is quickly moving from the traditional forms of point/click/swipe interfaces to those that most users will find more natural (such as speaking and gesturing, in the future potentially also thinking). Many of these technologies integrate with other capabilities that allow augmentation of the human experience with capabilities that naturally would not exist, often referred to with terms such as augmented reality, virtual reality, or mixed reality.

Ambient computing refers to contexts where the interaction experience between humans and technology has a tight integration with natural human experience that the technology as a separate entity becomes invisible. Various ways to describe this phenomenon include human augmentation (by Gartner) and digital reality as used by Deloitte to refer to augmented reality/virtual reality (AR/VR), mixed reality, voice interfaces, speech recognition, ambient computing, 360° video, and immersive technologies. Other technologies include ambient experience as described by Deloitte as input evolving from unnatural to natural (e.g., speaking, gesturing, and thinking) and the interactions between humans and technology moving from reactive (e.g., answering questions) to proactive (e.g., making unanticipated suggestions) as well as wearables, identified as major trends by KPMG and by WEF.

The area of *cognitive technologies* is a label frequently used to refer to a variety of artificial intelligence capabilities for addressing complex organizational and societal problems. For example, Deloitte specifies categories of these capabilities to include robotic process automation, textual and auditory natural language processing, machine learning, and computer vision. Other articulations of these categories include "AI and me" by Accenture and hyperautomation supported by AI and machine learning by Gartner. Other related technologies include cognitive technologies, consisting of machine learning, neural networks, robotic process automation, bots, natural language processing, neural nets, and the broader domain of AI by Deloitte, as well as Artificial Intelligence as part of DARQ technologies by Accenture.

Blockchain or Distributed Ledger refers to a set of technologies that allows set of actors to maintain a distributed record of transactions in a shared data storage environment in a way that is verifiable and permanent. Blockchain became first well-known as the technology underlying cryptocurrencies, but its potential areas of usage have expanded to financial services, management of contracts, health records, supply chain logistics, educational achievements, and many more. Sample reports refer to distributed ledger technologies with various names such as one of the DARQ technologies by Accenture, practical blockchain by Gartner, blockchain as a distributed ledger technology by Deloitte macro force, and distributed ledger by the WEF.

The area of *robotics* while it has been developed over the last few decades now brings together a broad range of areas of expertise to create non-human artifacts (both physical and intangible) to perform a variety of tasks in an increasing rich set of contexts. The best-known contexts for robotics are probably in manufacturing, but the advances have been very rapid recently in many other contexts, including warehouses, medical work, military operations, and even business processes. The sample reports refer to robotics with expressions such as broad expansion of context for the use of robotics by Accenture, autonomous things by Gartner, and symbiotic robots by Info Tech.

Quantum computing incorporates a broad range of activities across a broad range of academic disciplines and industry research laboratories towards a new type of computing model. The process harnesses quantum phenomena at the level of subatomic particles to solve complex problems at a scale that would not be possible with traditional computing models. Deloitte has selected quantum computers as one of its macro forces and defines its core contribution as its ability to solve certain highly complex problems that are too large and messy for current supercomputers in a broad range of areas from data to material sciences.

While *data privacy and digital ethics* are not technologies per say, it is important to note that each new generation of digital technologies, and all mentioned in this subsection, raise important questions about the relationship between humans and technology. Privacy is often the first context for these questions, but the range of questions is, in practice, much broader [Mar4]. Transparency and traceability by Gartner and data equity by Info Tech are two aspects in support of privacy and ethics.

7.7: Digest of Chapter 7

This chapter addresses some of the challenges faced by institutions in adapting their computing curricula to the current environment. It emphasizes the move from knowledge-based teaching to competency-based learning, accepting that this move will need to be managed differently in different educational contexts. It makes clear the need for universities to engage with industry in adapting their curricula and outlines a number of ways in which this engagement might take place. It explains what institutions must do to maintain currency in their programs and suggests some ways in which they might deal with the ever-changing needs and expectations of the people for whom the degree programs are designed. It concludes with a review of emerging future technologies that are likely to have a major impact on the future computing education.

Chapter 8: Beyond CC2020

CC2020 has surveyed the computing discipline and provided a structural view of computing that incorporates several sub-disciplines, including some that have recently emerged. This view of computing is based on ACM and IEEE-CS approved computing curricula that exist in 2020, which now includes cyber security and the data science curricula which is under development. However, the important contribution here is not a definition of the discipline as it currently stands, but the establishment of a foundation for curricular specification that is based on competencies. This competency-based view of computing intensifies prior work; a direction has been defined whereby competencies will be commonly used in the future. The pivot toward competencies in future computing curricular work is likely to be the most important contribution of this work. For this pivot to have impact, dissemination of these ideas is going to be critical. Aspects of this dissemination and pivot toward competencies are discussed below.

8.1: Public Engagement and the CC2020 Project

It is important that the efforts of the CC2020 project be available to the public worldwide. For the CC2020 project to be a success, it must engage the public. One means for accomplishing this goal is through an interactive website. Another is through a vigorous dissemination program sponsored by professional organizations, industry, and government.

8.1.1: CC2020 Project Website

The CC2020 project has established a preliminary website [Ccw1] where the public can obtain information regarding the project. Such information includes a current copy of this CC2020 report, information about the structure of the CC2020 project, samples of competencies and visualizations, and other accompanying material.

An important addition to the project website will be its ability for students, parents, industry and government professionals, and faculty members to have a dynamic interaction with the project website. This includes comparisons of programs in different computing disciplines, comparison of programs within the same computing discipline, contrasting competencies, and other interesting activities. This dynamic dimension of the project is and will be a work in progress beyond the publication of the CC2020 report.

8.1.2: Relating Curricula and Competencies

As noted in earlier chapters of this report, the notion of *competency* is the distinguishing feature of the CC2020 project, in contrast with the CC2005 project that focused on knowledge and knowledge-based learning. It is important for future curricular activities and development of curricular reports to consider embracing the use of competency in their work. The task force is aware that such future work will require greater effort in reaching a proper balance of dispositions, skills, and knowledge. However, since the majority (perhaps 99%) of graduates from computing programs will enter industry, government, or other workplace institutions, it is appropriate for all future curricula developers to embrace the competency-based approach.

8.1.3: Project Dissemination

The CC2020 project requires dissemination on a global scale. Such an undertaking requires the support of professional organizations and societies as well as educational institutions to underwrite this effort, which should be an ongoing undertaking several years after the publication of this CC2020 report.

The dissemination activity should spur new interest in competency-based learning and curricular structures. The activity should generate new research for grant opportunities in achieving graduates who are competent to enter industry as well as being prepared for graduate or post-baccalaureate education. The CC2020 project should become a catalyst for these endeavors.

8.2: Competency in Future Curricular Guidelines

This report highlighted competency as one of the salient features of the CC2020 project. It also examined various competency statements. Such statements may be useful in developing a uniform formalization of various disciplines. As presented, the competency-based approach makes it possible to compare computing disciplines and facilitate those comparisons. Recall that competency implies attaining a level of professional excellence and performance that goes beyond having only knowledge in a field. These extensions include technical skills and human attributes to function in the workplace at an acceptable level of performance. It is now important to extend the competency-based concept toward the development of future curricular guidelines within a common frame of reference.

8.2.1: Recent Curricular Development

The CC2020 task force believes that the use of competency in current and future computing curricular reports should be an important result of the CC2020 report. In today's world, graduates must be able to perform in the workplace with the technical skills and the human qualities in addition to knowledge.

The cybersecurity curricula project called CSEC2017 [Acm08], was published by ACM in December of 2017. The project used the traditional *knowledge area, knowledge unit, learning outcome* approach to develop its recommendations and its learning outcomes. Another effort parallel to the CC2020 project is the ACM data science curriculum project. The leaders of this effort have made a commitment to adopt competency as an ongoing theme because of this project. The curricular recommendations for information systems are currently undergoing a revision with a planned completion in 2020, IS2020 will be presented in the competency-based format.

8.2.2: Future Curricular Development

Given that most graduates of computing programs enter the workplace, it is very important that all computing programs prepare their graduates properly so they can perform as professionals and engage in productive careers. While the CC2020 project can only suggest its beliefs, its task force is confident that computing organizations and programs worldwide will heed the suggestions made in this report and transform their activities where competency becomes central to their future undertakings.

Naturally, other curricular such as software engineering, computer science, and computer engineering will continue to be revised and updated in the future. The task force is hopeful that all future curricular endeavors adopt the competency-based approach.

8.3: Competency Advocacy

The concept of using a common competency language to specify curricula, jobs, and careers provides an opportunity to bring all computing education stakeholders under a common umbrella. For this effort to be successful, all the stakeholders should reach consensus on the details of this language. For new curricular efforts that will emerge, authors should develop techniques that will support the deployment of a uniform competency-based approach. Industry, academia, and professional societies will need to develop these techniques together. The effort should incorporate a community of interest to oversee this development.

It will be necessary for the computing professional societies to take the leadership needed to develop model curricular standards using competencies. The computing professional societies should be part of any coalition of stakeholders, and the professional societies should mandate the use of competencies in developing future model curricula. Competencies should use a structure like the one advocated in this report, with both a prose statement and explicit knowledge, skills and dispositions components that can contribute analytically for visualization and comparison.

This project has planted the seeds for a high-quality public website that enables appropriate analysis of competency targets, and to provide career exploration and advice. This website should provide information about different types of computing careers as well as information on different types of degree programs that could prepare someone for a computing career. A variety of capabilities can be part of this site such as the ability to compare university programs in terms of their degree of similarity, and the degree of similarity between a program and a standard curriculum (such as the model curriculum for information systems), and the degree of similarity between an educational program and particular jobs and careers.

The efficacy of using competencies will occur as users gain experience with the approach. Once competency-based specifications of the various computing disciplines exist, then it would be possible to generate a comparative visual analysis among them – similar to the *ad hoc* visual representations of the CC2005 report, but with formal structural foundations.

8.4: Future Activities

The following list summarizes activities for curriculum-related developments that should take place over the next few years:

- Include the CC2020 report as an additional volume of the *Computing Curricula Series*.
- Establish new timetables for revision of each volume in the *Computing Curricula Series*.
- Strongly encourage a competency-based approach (rather than just knowledge-based learning) is part of all future computing curricular endeavors.
- Given the rapid pace of change in computing, consider more frequent revisions of computing curricular reports rather than the current approach, perhaps every six years instead of every ten to twelve years.
- Solicit improved support for more frequent updating of curricula.
- Continue processes for capturing feedback about each volume in the *Computing Curricula Series*.
- Establish new processes for ongoing evaluation of the adequacy of each curricular volume in the *Computing Curricula Series*.
- Improve current tools for visualizing computing programs.
- Develop new tools for visualizing computing programs.
- Become innovative in capturing new computing curricular areas to add to the *Computing Curricular Series*.

Consideration of the above projected activities should enhance computing education worldwide. The benefactors are the students who will enter those computing programs and the graduates of those programs who will find themselves as being competent professionals to enter the workplace or pursue further studies.

8.5: Digest of Chapter 8

Computing and computing education is more important now than ever before. This chapter emphasizes the need for the global dissemination of this CC2020 report with the support of professional organizations and educational institutions. The project further advocates that all current and future computing curricula adopt a competency approach to better prepare the computing professionals of the future. Lastly, future activities for curriculum-related developments that should take place over the next few years are presented.

References

R1: References for Report

[Aac1] American Association of Community Colleges (AACC). https://www.aacc.nche.edu/

[Abe1] ABET. Accreditation Board for Engineering and Technology. https://www.abet.org

[Acc1] Accenture: Technology Vision 2020: We, the Post-Digital People (https://www.accenture.com/us-en/insights/technology/technology-trends-2020)

[Acm00] ACM Website. https://www.acm.org/. Accessed 2019 May 6.

[Acm01] ACM Curricula Reports Website. https://www.acm.org/education/curricula-recommendations. Accessed 2019 March 11.

[Acm02] ACM (2005). *Computing Curricula 2005 The Overview Report.* ACM and IEEE Computer Society, ISBN: 1-59593-359-X.

[Acm03] ACM (2010) *IS 2010 Curriculum Guidelines for Undergraduate Degree Programs in Information Systems,* Association for Computing Machinery (ACM) and Association for Information Systems (AIS).

[Acm04] ACM (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science,* Association for Computing Machinery and IEEE Computer Society.

[Acm05] ACM (2014). *Software Engineering Curricula 2014 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering,* Association for Computing Machinery and IEEE Computer Society.

[Acm06] ACM (2016). Computer Engineering Curricula 2016 Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering, Association for Computing Machinery (ACM) and IEEE Computer Society, ISBN: 978-1-4503-4875-1, DOI: 10.1145/3025098, Web link: https://dx.doi.org/10.1145/3025098.

[Acm07] ACM (2017). *Information Technology Curricula 2017 Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology,* Association for Computing Machinery (ACM) IEEE Computer Society (IEEE-CS), ISBN: 978-1-4503-6416-4, DOI: 10.1145/3173161, Web link: https://dl.acm.org/citation.cfm?id=3173161.

[Acm08] ACM (2017). Cybersecurity Curricula 2017 Curriculum Guidelines for Post-Secondary Degrees in Cybersecurity. Association for Computing Machinery (ACM), IEEE Computer Society (IEEE-CS), Association for Information Systems Special Interest Group on Information Security and Privacy (AIS SIGSEC), International Federation for Information Processing Technical Committee on Information Security Education (IFIP WG 11.8), ISBN: 978-1-4503-5278-9, DOI: 10.1145/3184594, Web link: https://dl.acm.org/citation.cfm?id=3184594. Also at https://cybered.hosting.acm.org/wp/

[Acm09] ACM (2020) Information Technology Curricular Guidance for Transfer Programs. Available at http://ccecc.acm.org/files/publications/IT-Transfer2020.pdf .

[Acm10] ACM (2017). *Computer Science Transfer Curriculum 2017 Computer Science Curricular Guidance for Associate-Degree Transfer Programs with Infused Cybersecurity,* The Association for Computing Machinery (ACM) Committee for Computing Education in Community Colleges (CCECC), ISBN: 978-1-4503-5369-4, DOI: 10.1145/3108241, Web link: http://dx.doi.org/10.1145/3108241.

[Acm11] ACM (2017) Topi, H., Karsten, H., Brown, S. A., Carvalho, J. A., Donnellan, B., Shen, J., et al. (2017). MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems, 40(1). Retrieved from http://aisel.aisnet.org/cais/vol40/iss1/186

[Acm12] ACM (2001). *Computing Curricula 2001: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science,* Association for Computing Machinery and IEEE Computer Society. https://www.acm.org/binaries/content/assets/education/curricula-recommendations/cc2001.pdf. Accessed 2019 Jun 26.

[Acm13] ACM. 1968. Curriculum 68: Recommendations for academic programs in computer science: a report of the ACM curriculum committee on computer science. Communications of the ACM (CACM) Volume 11 Issue 3, March 1968, pp 151-197.

[Acm14] ACM. 1972. Ashenhurst, Robert L (Ed.). Curriculum recommendations for graduate professional programs in information systems. Communications of the ACM (CACM), Volume 15 Issue 5, May 1972, Pages 363-398.

[Acm15] ACM. 1973. Couger, J. (Ed.). Curriculum Recommendations for Undergraduate Programs in Information Systems, Communications of the ACM, Volume 16, Number 12, December 1973, pp. 727-749.

[Acm16] ACM Education Policy Committee (2018) Lighting the Path: From Community College to Computing Careers. Available at https://www.acm.org/binaries/content/assets/education/lighting-the-path-from-community-college-to-computing-careers.pdf

[Acm17] ACM (2020) Cybersecurity Curricular Guidance for Associate-Degree Programs. Available at http://ccecc.acm.org/files/publications/Cyber2yr2020.pdf .

[Als1] Alsop, Steve. 2005. Beyond Cartesian Dualism: Encountering Affect in the Teaching and Learning of Science. Vol. 29. Springer Science & Business Media, Berlin, Germany.

[And1] Anderson (2001). Bloom's Cognitive Process Dimension.

[And5] L. W. Anderson et al., "A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives, abridged edition," White Plains, NY Longman, 2001.

[Ann1] Annas, Julia, "Intelligent Virtue," Oxford Press, 2011.

[Atc1] Atchison, William F. et al., (1968) Curriculum 68: Recommendations for academic programs in computer science: a report of the ACM curriculum committee on computer science, *Communications of the ACM*: Volume 11 Issue 3, March 1968.

[Are1] Are-We-In-Your-State, https://www.areweinyourstate.org/

[Aus1] Austing, Richard H. et al. Curriculum '78: Recommendations for the undergraduate program in computer science— a report of the ACM curriculum committee on computer science, *Communications of the ACM*: Volume 22 Issue 3, March 1979.

[Bac1] Bachelor degrees in Japan (2017). http://www.niad.ac.jp/media/001/201901/no9_13_2017data_fuki_Bachelor_English.pdf. Accessed 2019 June 27.

[Bai1] Baiyere, A., Donnellan, B., Venkatesh, V., and Wyatt, J. (forthcoming). "Internet of Things (IoT) – A Research Agenda for Information Systems," Conditionally accepted for publication in *Communications of the AIS*. (Forthcoming)

[Bai2] D. Bairaktarova and A. Woodcock, “Engineering Student’s Ethical Awareness and Behavior: A New Motivational Model,” Sci. Eng. Ethics, vol. 23, no. 4, pp. 1129–1157, 2017.

[Ban1] Wilder, Coleen R. and Ozgur, Ceyhun O. (2015) *Business Analytics Curriculum for Undergraduate Majors*, published in INFORMS Transactions on Education 15(2):180-187. https://doi.org/10.1287/ited.2014.0134, Accessed 2010 Jan06.

[Bar1] Baron, R. M., & Kenny, D. A. (1986). The moderator-mediator variable distinction in social psychological research: Conceptual, strategic, and statistical considerations. Journal of Personality and Social Psychology, 51, 1173-1182. Retrieved from https://psych.wisc.edu/henriques/mediator.html (current September 12, 2019).

[Bau1] I. Baumgartner and V. Shankararaman, “Actively linking learning outcomes and competencies to course design and delivery: experiences from an undergraduate Information Systems program in Singapore,” in Proceedings of the 2013 IEEE Global Engineering Education Conference, Berlin, Germany, 2013, pp. 238–246.

[Bil1] S. Billett, “Realising the educational worth of integrating work experiences in higher education,” Stud. High. Educ., vol. 34, no. 7, pp. 827–843, Nov. 2009.

[Biz1] Business Dictionary. http://www.businessdictionary.com/definition/entrepreneurship.html

[Biz2] Business Dictionary. http://www.businessdictionary.com/definition/innovation.html

[Bhe1] BHEF (2016) *Data Science and Analytics (DSA) Competency Map, The Business Higher Education Framework (BHEF),* version 1.0, produced in November 2016. https://s3.goeshow.com/dream/DataSummit/Data%20Summit%202018/BHEF_2016_DSA_competency_map_1.pdf. Accessed 2020Jan06.

[Blo1] Bloom, B.S. (Ed.). Engelhart, M.D., Furst, E.J., Hill, W.H., Krathwohl, D.R. (1956). Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain. New York: David McKay Co Inc.

[Bls1] Bureau of Labor Statistics, U.S. Department of Labor, *Occupational Outlook Handbook,* 2016-17 Edition, Computer and Information Technology Occupations; https://www.bls.gov/ooh/computer-and-information-technology/home.htm. Accessed 2017 Dec 2.

[Bls2] Bureau of Labor Statistics, U.S. Department of Labor, *Occupational Outlook Handbook,* 2016-17 Edition, Computer and Information Technology Occupations, on the Internet at https://www.bls.gov/ooh/computer-and-information-technology/information-security-analysts.htm. Accessed 2017 Dec 2.

[Blu1] “Blue Book” (2019) *The Blue Book Working Group of the Forum of Chinese Twenty-Experts on Computing Education in China* Computer Education for Sustainable Competence, China Higher Education Publishing, January 2019 (in Chinese).

[Bol1] Bologna Declaration. http://www.ehea.info/cid100210/ministerial-conference-bologna-1999.html

[Bol2] Bologna Working Group on Qualifications Frameworks (2005). *A Framework for Qualifications of the European Higher Education Area*. The Danish Ministry of Science, Technology and Innovation.

[Bri1] British Computer Society. Guidelines on course accreditation Information for universities and colleges. BCS, January 2018. http://www.bcs.org/upload/pdf/2018-guidelines.pdf.

[Bur1] Burning Glass. (2017). *The Digital Edge: Middle-Skill Workers and Careers*. https://www.burning-glass.com/wp-content/uploads/Digital_Edge_report_2017_final.pdf. Accessed 14 May 2019.

[Cas1] Cassel, L., & Topi, H. (2016). *Strengthening Data Science Education Through Collaboration*. Retrieved from http://www.computingportal.org/sites/default/files/Data%20Science%20Education%20Workshop%20Report%201.0_0.pdf

[Cas2] Michael E. Caspersen, Judith Gal-Ezer, Andrew McGettrick, Enrico Nardelli.” Informatics as a fundamental discipline for the 21st century,” Communications of the ACM, April 2019, Vol. 62 No. 4, Page 58. https://cacm.acm.org/magazines/2019/4/235598-informatics-as-a-fundamental-discipline-for-the-21st-century/fulltext

[Ccw1] CC2020 Project Website. https://www.cc2020.net/. Accessed 2019 May 9

[Cha1] D. W. Chambers and J. D. Gerrow, “Manual for Developing and Formatting Competency Statements,” Journal of Dental Education, vol. 58, no. 5, pp. 361–66, 1994.

[Chy5] Chyung, S., Stepich, D. & Cox, D. (2006) “Building a Competency-Based Curriculum Architecture to Educate 21st-Century Business Practitioners” in The Journal of Education for Business 81(6):307-314.

[Cio] CIO Council. (2012). Clinger-Cohen core competencies & learning objectives. Washington, DC: CIO Council.

[Cis1] CISCO. Cisco Training and Certifications. https://www.cisco.com/c/en/us/training-events/training-certifications.html

[Cla1] Classroom, The. (2019) Knowledge Based Learning. https://www.theclassroom.com/knowledge-based-learning-5403738.html

[Cle1] Clear, T. (2017). Thinking Issues: Meeting employers’ expectations of devops roles: can dispositions be taught? Inroads, 8(2), 19–21. https://doi.org/10.1145/3078298

[Cle2] Clear, A., Clear, T., Impagliazzo J.. Wang, P. (2020). “From Knowledge-based to Competency-based Computing Education: Future Directions, submitted for Review.

[Col1] Collins Dictionary; http://www.collinsdictionary.com/dictionary/english/communication-skills. Accessed 2017 Dec 2.

[Com1] Computing at School, Computing in the national curriculum: A guide for primary teachers, 2013. https://www.computingatschool.org.uk/data/uploads/CASPrimaryComputing.pdf

[Com2] Computing at School, Computing in the national curriculum: A guide for secondary teachers, 2014. https://www.computingatschool.org.uk/data/uploads/cas_secondary.pdf

[Com3] CompTIA. https://www.comptia.org/

[Con1] Conceiving-Designing-Implementing-Operating (CDIO). Webpage. http://www.cdio.org/. Accessed 9 May 2019.

[Cos1] Computer Society of IEEE. http://www.computer.org/. Accessed 2019 May 6.

[Cou1] Couger, J. (Ed.). Curriculum recommendations for Undergraduate programs in information systems. Comm. ACM 16, 12 (Dec. 1973), 727-749.

[Cpt1] CompTIA, *Information Technology (IT) Industry & Association*; https://www.comptia.org/. Accessed 2017 Dec 2.

[Cpt2] CompTIA, *Building Digital Organizations*, February 2017.

[Csp1] CSpathshala. www.cspathshala.org/. Accessed 2019Aug17.

[Cua1] Cuadros-Vargas, E. (2018a). Escuela Profesional de Ciencia de la Computacion. https://education.spc.org.pe/Peru/CS-UTEC/Plan%202018/CS-UTEC-poster.pdf Accessed 9 May 2019.

[Cua2] Cuadros-Vargas, E. (2018b). 3.9 Compatibilidad de la carrera con relación a estandares internacionales. https://education.spc.org.pe/Peru/CS-UTEC/Plan%202018/3_9_Compatibilidad_carrera_.html Accessed 9 May 2019.

[Dab1] Dabbagh, N., Benson, A. D., Denham, A., Joseph, R., Al-Freih, M., Zgheib, G., ... & Guo, Z. (2016). Massive open online courses. In *Learning technologies and globalization* (pp. 9-13). Springer, Cham.

[Dat1] Data Science Task Force; http://dstf.acm.org/

[Dat2] Data Science Draft Report 1. (2019). http://www.cs.williams.edu/~andrea/DSReportInitialFull.pdf. Accessed 2019 May 6.

[Dat3] Data Science Draft Report 2. (2019) http://dstf.acm.org/DSReportInitialFull.pdf. Accessed 2020 Jan 6.

[Dat4] DataUSA. 2019. https://datausa.io/profile/cip/computer-and-information-sciences-and-support-services. Accessed 2019Aug17.

[Dav1] Dave, R.H. (1970). *Psychomotor levels in Developing and Writing Behavioral Objectives*, pp.20-21. R.J. Armstrong, ed. Tucson, Arizona: Educational Innovators Press.

[Dea1] Deadline.com, https://deadline.com/2018/07/film-industry-revenue-2017-ibisworld-report-gloomy-box-office-1202425692/

[Del1] Deloitte: Deloitte Insights: Tech Trends 2020 (https://www2.deloitte.com/us/en/insights/focus/tech-trends.html)

[Den1] Denning, P.J. et al., "Computing as a Discipline," Communications of the ACM, vol. 32, no. 1, pp. 9-23, (1989)

[Dep1] Department for Education. GCE AS and A level subject content for computer science. DFE-00359-2014. DFE, April 2014. https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/302105/A_level_computer_science_subject_content.pdf.

[Dic1] Dictionary.com; http://dictionary.reference.com/browse/soft+skills. Accessed 2017 Dec 2.

[Dic2] Dictionary.com; http://dictionary.reference.com/browse/teamwork. Accessed 2015 Dec 2.

[Dso1] Dsouza, John. https://www.linkedin.com/pulse/based-learning-11-kbl-knowledge-based-john-dsouza

[Edg1] Education Glossary, https://www.edglossary.org/content-knowledge/

[Eco1] European Commission. The Digital Competence Framework 2.0. https://ec.europa.eu/jrc/en/digcomp/digital-competence-framework

[Edi1] EDSF (2017). The EDISON Data Science Framework. Available at http://edison-project.eu/sites/edison-project.eu/files/attached_files/node-488/edison-general-introduction-edsf.pdf; accessed February 3, 2019.

[Edu1] Eduglopedia. https://eduglopedia.org

[Edu2] Education Scotland, Benchmark: Technologies, March 2017. https://education.gov.scot/improvement/documents/technologiesbenchmarkspdf.pdf

[Ent1] Enterprise Information Technology Body of Knowledge (EITBOK). http://eitbokwiki.org/Main_Page. Accessed 2019Aug17.

[Eqa1] European Quality Assurance Network for Informatics Education (EQANIE). https://eqanie.webs.upv.es/quality-label/the-euro-inf-framework-standards-and-criteria/

[Eur1] European e-Competence Framework (e-CF), 3.0. http://www.ecompetences.eu/

[Eur2] European Higher Education Area (EHEA). http://www.ehea.info/page-three-cycle-system. Accessed 2019 Aug 16.

[Eur3] European Qualifications Framework (EQF). European Commission. Learning Opportunities and Qualifications. https://ec.europa.eu/ploteus/en/content/descriptors-page/. Accessed 2019Aug17.

[Exe1] Exeter University. http://www.exeter.ac.uk/undergraduate/degrees/computerscience/comsci/#Programme-structure.

[Fei1] Fein, L. (1959). The Role of the University in Computers, Data Processing, and Related Fields. *Commun. ACM*, *2*(9), 7–14. http://doi.org/10.1145/368424.368427

[Fra1] Fraillon J., John Ainley, Schulz W., Friedman T., and Duckworth D. Preparing for Life in a Digital World. IEA International Computer and Information Literacy Study 2018 (ICILS 2018). Springer , Cham (2020). ISBN 978-3-030-38780-8. ISBN 978-3-030-38781-5 (eBook). https://doi.org/10.1007/978-3-030-38781-5. https://www.iea.nl/sites/default/files/2020-04/IEA%20International%20Computer%20and%20Information%20Literacy%20Study%202018%20International%20Report.pdf

[Fre1] Frezza, Stephen, Mats Daniels, Arnold Pears, Åsa Cajander, Viggo Kann, Amanpreet Kapoor, Roger McDermott, Anne-Kathrin Peters, Mihaela Sabin, and Charles Wallace. 2019. "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, 27. ACM.

[Fre2] Frezza, Stephen, Mats Daniels, and Aaron Wilkins, "Assessing Students' IT Professional Values in a Global Project Setting" In ACM Transactions on Computing Education, Special Issue on Global Software Engineering Education (19,2) 2019 pp. 9:1-9:34, http://doi.acm.org/10.1145/3231710.

[Fre5] S. Frezza et al., "Modelling competencies for computing education beyond 2020: a research based approach to defining competencies in the computing disciplines," pp. 148–174, 2018.

[Ful2] U. Fuller, A. Pears, J. Amillo, C. Avram, and L. Manilla, "A Computing Academic Perspective on the Bologna Process," ACM SIGCSE Bull., vol. 38, no. 4, pp. 115–131, Dec. 2006.

[Gar1] Gartner: Top 10 Strategic Technology Trends for 2020 (https://www.gartner.com/en/publications/top-tech-trends-2020)

[Gra1] Gray, J. M. (2015). Virtue Ethics: Examining Influences on the Ethical Commitment of Information System Workers in Trusted Positions. Dissertation: Nova Southeastern University. Nova Southeastern University. Retrieved from http://nsuworks.nova.edu/gscis_etd/364

[Gra2] G. Grant, 'On competence: a critical analysis of competence-based reforms in higher education.', 1979.

[Han1] H. Han, "Virtue Ethics, Positive Psychology, and a New Model of Science and Engineering Ethics Education," Sci. Eng. Ethics, vol. 21, no. 2, pp. 441–460, 2014.

[Har1] Harrow, A. (1972). *A Taxonomy of Psychomotor Domain: A Guide for Developing Behavioral Objectives*. David McKay Co., Inc., New York, USA.

[Har2] Harvard. https://apps2.campusservices.harvard.edu/cas/empldocs/cs/harvard_competency_dictionary_complete.pdf.

[Hel1] Helfert, M. (2008). Business informatics: An engineering perspective on information systems. *Journal of Information Technology Education: Research*, *7*, 223–245.

[Icd1] http://icdtranslation.com/skill-based-and-knowledge-based-online-training/

[Iee1] IEEE (2017) Los Alamitos, Calif., Dec. 14, 2017. https://www.prnewswire.com/news-releases/top-10-technology-trends-for-2018-ieee-computer-society-predicts-the-future-of-tech-300571274.html

[Iee2] The Informatics Europe and European Commission joint report on Industry-University cooperation. https://www.informatics-europe.org/news/544-bridging-the-digital-talent-gap-towards-successful-industry-university-partnerships.html, April 2020

[Iee3] IEEE Computer Society (2014) Software Engineering Competency Model: Version 1.0, SWECOM 2014.

[Imp1] Impagliazzo, John. (2018) The Role of Competency and the Future of Computer Education. *Communications of the China Computer Federation*; Volume 14, Number 9; September 2018. (in Chinese)

[Imp2] Impagliazzo, J., et al. (2018). Developing an Overview of Computing/Engineering Curricula via the CC2020 Project. In *Proc. of the IEEE EduNine Conference*. IEEE Education Society.

[Imp3] Impagliazzo J., Parrish, A., & Clear, A. (2018). Innovative Computing Curricula and the CC2020 Project. In *Proc. of the Frontiers in Education (FIE) Conference*.

[Ind1] India 1. www.ugc.ac.in

[Ind2] India 2. www.naac.gov.in

[Ind3] India 3. www.aicte-india.org

[Ins1] Institut de France, L'Académie des Sciences, “L'enseignement de l’informatique en France – Il est urgent de ne plus attendre,” Mai 2013, https://www.academie-sciences.fr/pdf/rapport/rads_0513.pdf

[Ind4] India 4. www.nbaind.org

[Inv1] Investopedia; http://www.investopedia.com/terms/s/soft-skills.asp. Accessed 2017 Dec 2.

[Ipa1] Information-technology Promotion Agency – Japan (IPA). IT Human Resources Development: i Competency Dictionary. https://www.ipa.go.jp/english/humandev/icd.html. Accessed 2019Aug18.

[Kak1] Kakeshita, Tetsuro. “National Survey of Japanese Universities on Computing Education: Analysis of Departments Majored in Computing Discipline”, Olympiads in Informatics, 2018, Vol. 12, 69–84, https://ioinformatics.org/journal/v12_2018_69_84.pdf (accessed June 29, 2019).

[Ken1] D. Kennedy, A. Hyland, and N. Ryan, “Learning outcomes and competences,” Introd. Bologna Object. Tools, pp. 2–3, 2009.

[Kpm1] KPMG: Technology Trends Index USA (http://technologytrendsindex.kpmg.com/)

[Kra1] 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.

[Kra2] M. Kramer, P. Hubwieser, and T. Brinda, “A Competency Structure Model of Object-Oriented Programming,” in 2016 International Conference on Learning and Teaching in Computing and Engineering (LaTICE), 2016, pp. 1–8.

[Kuh1] George D. Kuh. (2009). The national survey of student engagement: Conceptual and empirical foundations. New directions for institutional research 2009, 141 (2009), 5–20.

[Len1] Lenburg, C. B. (1999). The framework, concepts and methods of the competency outcomes and performance assessment (COPA) model. Online Journal of Issues in Nursing, vol. 4, no. 2, pp 1-12.

[Lin1] Lindenwood University Makerspace Lab, https://www.lindenwood.edu/about/news/details/converged-media-lab-makerspace-to-open-this-fall/. Accessed 2019Aug17.

[Liu11 Liu, K., Semiotics in Information Systems Engineering, Cambridge University Press, Cambridge, U.K, 2000.

[Loc1] Lockoff, J., Wegewiis, B., Durkin, K., Wagenaar, R., Gonzales, J., Isaacs, A. K., Donà dalle Rose, L.F., & Gobbi, M. (Eds.) (2010). A Tuning guide to formulating degree programme profiles: Including programme competences and programme learning outcomes. Bilbao, Spain: University of Deusto. Retrieved from http://core- project.eu/documents/Tuning%20G%20Formulating%20 Degree%20PR4.pdf

[Lun1] Lunt, B.M., Ekstrom, J.J., Gorka, S., Hislop, G., Kamali, R., Lawson, E., LeBlanc, R., Miller, J. and Reichgelt, H. 2008. Curriculum Guidelines for Undergraduate Degree Programs in Information Technology; https://dl.acm.org/citation.cfm?id=2593311. Accessed 2017 Dec 2

[Mar1] Marshall, Linda (2012) A comparison of the core aspects of the ACM/IEEE Computer Science Curriculum 2013 Strawman report with the specified core of CC2001 and CS2008 Review. Computer Science Education Research Conference. pp. 29–34. CSERC 2012, ACM.

[Mar2] Marshall, Linda (2014) A graph-based framework for comparing curricula. Ph.D. thesis, University of Pretoria, South Africa.

[Mar3] Marshall, Linda (2017) A Topic-Level Comparison of the ACM/IEEE CS Curriculum Volumes. In: Liebenberg J., Gruner S. (eds) ICT Education. SACLA 2017. Communications in Computer and Information Science, vol 730. pp. 309-324. Springer, Cham.

[Mar4] M. Lynne Markus and Heikki Topi. 2015. ”Big Data, Big Decisions for Science, Society, and Business: Report on a Research Agenda Setting Workshop.” Technical Report. National Science Foundation, USA.

[Mer1] Merriam-Webster. https://www.merriam-webster.com/dictionary/competency. Accessed 2019 June 3.

[Mer2] Merriam-Webster. Disposition. https://www.merriam-webster.com/dictionary/disposition/. Accessed 2019Aug30.

[Mer3] Merriam-Webster. Learning. https://www.merriam-webster.com/dictionary/learning

[Mer4] Merriam-Webster. Knowledge. https://www.merriam-webster.com/dictionary/knowledge?src=search-dict-box#synonyms

[Mic1] Microsoft Certifications. https://docs.microsoft.com/en-us/learn/certifications/

[Min1] Ministry of Education, New Zealand (2019) What are makerspaces? http://elearning.tki.org.nz/Teaching/Future-focused-learning/Makerspaces. Accessed 2019 Jul 3.

[Nas1] National Academies of Science (NAS) (2018). Envisioning the Data Science Discipline: The Undergraduate Perspective: Interim Report, ISBN 978-0-309-46502-1 | DOI 10.17226/24886.

[Nas2] National Academies of Sciences, Engineering, and Medicine (2018), National Academies of Sciences, Engineering, and Medicine, "Assessing and Responding to the Growth of Computer Science Undergraduate Enrollments," Committee on the Growth of Computer Science Undergraduate Enrollments, The National Academies of Sciences, Engineering and Medicine, Washington, DC, 2018. pp. 84-87

[New1] University of Newcastle, Information Technology. https://www.newcastle.edu.au/degrees/bachelor-of-information-technology.

[Nov1] Novinson, M. 2017. *Top 15 Moneymaking Certifications for 2017*. CRN, The Channel Company. http://www.crn.com/slide-shows/managed-services/300080027/top-15-moneymaking-certifications-for-2016.htm. Accessed 2017 Dec 2.

[Nrc1] National Research Council (2012) A Framework for K-12 Science Education: Practices, Crosscutting Concepts, and Core Ideas. The National Academies Press. 500 Fifth Street, NW, Washington, DC 20001. ISBN 13: 978-0-309-21742-2.

[Nsa1] National Security Agency. Centers for Academic Excellence. https://www.nsa.gov/resources/students-educators/centers-academic-excellence/

[Nwo1] Nwokeji, J. C., Stachel, R., & Holmes, T. (2019). Competencies Required for Developing Computer and Information Systems Curriculum. In Proceedings of the 49th Frontiers in Education Conference (FIE’19) (pp. 1–9). Cincinnati, OH: IEEE Computer Society.

[Ope1] OpenAir: African Innovative Research, http://www.openair.org.za/africas-maker-movement-an-overview-of-ongoing-research/. Accessed 2019Aug17.

[Ols1] Olson, M. 2016. “A Multilateral Approach to Bridging the Global Skills Gap.” *Cornell HR Review*, 5/8/15, p. 1.

[Par1] DeVeaux, R. et al. (2017) *Curricular Guidelines for Undergraduate Programs in Data Science*, Annual Review of Statistics and Its Application 4:15–30. https://www.amstat.org/asa/files/pdfs/EDU-DataScienceGuidelines.pdf. Accessed 2020Jan06.

[Per1] Perkins, David N., Eileen Jay, and Shari Tishman. 1993. “Beyond Abilities: A Dispositional Theory of Thinking.” Merrill-Palmer Quarterly 39 (1): 1–21.

[Per2] Perry, Mark J. 2018. Table of the Day: Bachelor’s degrees for the Class of 2016 by field and gender. Oh, and the overall 25.6% degree gap for men! http://www.aei.org/publication/table-of-the-day-bachelors-degrees-for-the-class-of-2016-by-field-and-gender-oh-and-the-overall-25-6-college-degree-gap-for-men/. Accessed 2019Aug17.

[Per3] Peris-Ortiz, M., Llera, J. J. A., & Rueda-Armengot, C. (2017). Entrepreneurship and Innovation in a Revolutionary Educational Model: École, 42. In Social Entrepreneurship in Non-Profit and Profit Sectors (pp. 85-97). Springer, Cham.

[Per4] Perkins, David N. and Shari Tishman. 2006. Learning that matters: Towards a dispositional perspective on education and its research needs. A report prepared for the Spencer Foundation (2006), 43.

[Per5] D. N. Perkins, E. Jay, and S. Tishman, “Beyond Abilities: A Dispositional Theory of Thinking,” Merrill. Palmer. Q., vol. 39, no. 1, pp. 1–21, 1993.

[Pik1] Pikkarainen, Eetu (2014). “Competence as a Key Concept of Educational Theory: A Semiotic Point of View.” Journal of Philosophy of Education 48 (4): 621–36.

[Pro1] Provost, Foster and Tom Fawcett (2013) Data Science for Business. O’Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

[Psi1] PSI (2018). 5 Stats About the Skills Gap That Demand Attention. https://blog.psionline.com/talent/5-stats-about-the-skills-gap-that-demands-attention, Accessed 27 May 2019.

[Rad1] A. Radermacher and G. Walia, “Gaps Between Industry Expectations and the Abilities of Graduates,” in Proceeding of the 44th ACM Technical Symposium on Computer Science Education, 2013, pp. 525–530.

[Ram1] Ramos, Teófilo J., Osvaldo M Micheloud, Richard Painter, and Moshe Kam (2013). IEEE Common Nomenclature for Computing Related Programs in Latin America. IEEE.

[Reg1] Reges, Stuart. 2018. Furor on Claim Women's Choices Create Gender Gap in Comp Sci. Inside Higher Ed. https://www.insidehighered.com/news/2018/06/25/lecturers-explanation-gender-gap-computer-science-it-reflect-womens-choices. Accessed 2019Aug17.

[Rhu1] Rhumbl. Webpage. https://rhumbl.com/. Accessed 9 May 2019.

[Roy1] The Royal Society, After the reboot:computing education in UK schools, November 2017, ISBN: 978-1-78252-297-3, https://royalsociety.org/~/media/policy/projects/computing-education/computing-education-report.pdf

[Sab1] Sabin, M., H. Alrumaih, and J. Impagliazzo. 2018. “A Competency-Based Approach toward Curricular Guidelines for Information Technology Education.” In 2018 IEEE Global Engineering Education Conference (EDUCON), 1214–21. Tenerife: IEEE.

[Sch1] Schussler, Deborah L. 2006. “Defining Dispositions: Wading Through Murky Waters.” The Teacher Educator 41 (4): 251–68.

[Sfi1] Skills Framework for Information Age (SFIA). https://sfia-online.org/en/assets/documents/pdfs-to-support-views/2019-08-software-engineering-competencies.pdf.

[Sii1] Siirtola, H., Räihä, K., & Surakka, V. (2013). Interactive Curriculum Visualization. In *2013 17th International Conference on Information Visualisation*, pp. 108-117.

[Sim1] Simon, Clear, A., Carter, J., Cross, G., Radenski, A., Tudor, L., & Tonisson, E. (2015). What's in a Name? International Interpretations of Computing Education Terminology. *ITiCSE2015* (pp. 173-186). Vilnius, Lithuania: ITICSE-WGR'15.

[Sim2] Simon, H. (1996), The Sciences of the Artificial, 3ed, M.I.T., Cambridge, MA, p. 119.

[Som1] Sommaruga, L., & Catenazzi, N. (2007). Curriculum visualization in 3D. In *Proc. of the 12th international conference on 3D web technology (Web3D '07)*. ACM. pp.177-180.

[Squ1] A. Squires and W. Larson, “Improving systems engineering curriculum using a competency-based assessment approach,” International Journal of Intelligent Defence Support Systems, vol. 2, no. 3, pp. 184, 2009.

[Sta1] Stamper, R. K. (1991) The semiotic framework for information systems research. Information systems research. *Contemporary approaches and emergent traditions*, pp. 515-528.

[Sta2] Stamper, R. K., Information in Business and Administrative Systems, John Wiley and Sons, New York, NY, 1973.

[Sta3] Stamper, R. K., Althous, K. and Backhouse, J., “MEASUR, Method for Eliciting, Analyzing, and Specifying User Requirements,” In Olle, T. W., Verrijn-Stuart, A. A. and Bhabuts, L., (eds.), Computerized Assistance During the Information Life Cycle, Elsevier, Amsterdam, The Netherlands, 1988.

[Ste1] Steinbuch, Karl (1957). Informatik: Automatische Informationsverarbeitung. Berlin: SEG-Nachrichten.

[Sto1] A. Stoof, R. L. Martens, and J. J. Van Merriënboer, “What is competence? A constructivist approach as a way out of confusion,” Onderwijs Res. Dagen ORD Leiden Neth., 2000.

[Tak1] S. Takada et al., “Toward Understanding Computing Curricula: Visualization Using Competency within the CC2020 Project,” Educ. Inf. Technol., 2020.

[Tan1] Tang, Cara, Elizabeth K Hawthorne, Cindy S Tucker, Ernesto Cuadros-Vargas, Diana Cukierman, Simon, Ming Zhang (2016). Global Perspectives on the Role of Two-Year/Technical/Junior Colleges in Computing Education. 21st ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE ’16), Arequipa, Peru, 204-205.

[Tec1] TechTarget. https://whatis.techtarget.com/definition/concept-map

[Ted1] Tedre, Matti (2015). *The Science of Computing: Shaping a Discipline*. Boca Raton, CRC Press / Taylor & Francis.

[Ted2] Tedre, M., & Sutinen, E. (2008). Three traditions of computing: what educators should know. *Computer Science Education*, *18*(3), 153–170.

[Tes1] https://www.tes.com/teaching-resource/kbl-knowledge-based-learning-11361440

[The1] https://www.theesa.com/wp-content/uploads/2019/05/2019-Essential-Facts-About-the-Computer-and-Video-Game-Industry.pdf

[Tre1] Trevelyan, James. 2014. The Making of an Expert Engineer. Taylor and Francis Group, Ltd.

[Top1] Topi, Heikki. 2017. “Information Systems in CC2020: Comparing Key Structural Elements of Curriculum Recommendations in Computing.” In 2017 Proceedings of SIGED: IAIM Conference. Association for Information Systems Electronic Library (AISeL). http://aisel.aisnet.org/siged2017/9.

[Top5] H. Topi, “Information Systems in CC2020: Comparing Key Structural Elements of Curriculum Recommendations in Computing,” in 2017 Proceedings of SIGED: IAIM Conference, 2017.

[Tuc1] Tucker, A.B. 1991. Computing curricula 1991. *Communications of the ACM*, 34(6), pp.68-84; http://dl.acm.org/citation.cfm?doid=103701.103710.

[Van1] M. Van der Klink and J. Boon, “The investigation of competencies within professional domains,” Hum. Resour. Dev. Int., vol. 5, no. 4, pp. 411–424, 2002.

[Van2] M. Van der Klink, J. Boon, and K. Schlusmans, “Competences and Vocational Higher Education: Now and in Future,”Eur. J. Vocat. Train., vol. 40, no. 1, pp. 67–82, 2007.

[Vor1] Voorhees, R. A., & Bedard-Voorhees, A. (2016). Principles for Competency-based education. In Instructional-Design Theories and Models, Volume IV (pp. 49-80). Routledge.

[Wag1] 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), 48.

[Wag5] L. Waguespack et al., “Adopting Competency Mindful of Professionalism in Baccalaureate Computing Curricula,” in EDSIGCON 2019, 2019.

[Web1] Weber, H. (2017). The New Virtues of Engineering and the Need for Change in the Engineering Curriculum. From https://www.researchgate.net/publication/325924314_The_New_Virtues_of_Engineering_and_the_Need_for_Change_in_the_Engineering_Curriculum

[Wef1] World Economic Forum (2020): Jobs of Tomorrow: Mapping Opportunity in the New Economy (https://www.weforum.org/reports/jobs-of-tomorrow-mapping-opportunity-in-the-new-economy)

[Wei1] F. E. Weinert, “Definition and selection of competencies: Concepts of Competence,” Organization for Economic Co-operation and Development, 1999.

[Wig1] Wiggins, G., McTighe, J., and Ebrary, I. (2005). *Understanding by design (Expanded second edition)*. Alexandria, VA: Association for Supervision and Curriculum Development.

[Wig2] Wiggins, G., & McTighe, J. (2011). *The Understanding by Design Guide to Creating High-Quality Units.* Alexandria, VA: Association for Supervision and Curriculum Development.

[Wil1] Willcox, K., & Huang, L. (2017). Mapping the CDIO Curriculum with Network Models. CDIO, In *13th International CDIO Conference*.

[Zuc1] Zucker, R. (2009). ViCurriAS: a curriculum visualization tool for faculty, advisors, and students. In *J. Comput. Sci. Coll.* 25(2);138-145.

R2: Extra References

[Acc1] Accenture: Technology Vision 2020: We, the Post-Digital People (https://www.accenture.com/us-en/insights/technology/technology-trends-2020)

[Acm20] ACM, & AIS. (2017c). *MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems*. https://www.acm.org/binaries/content/assets/education/msis2016.pdf. Accessed 14 May 2019.

[Acm21] ACM (2019). *Computing Competencies for Undergraduate Data Science Curricula* (Initial Draft). http://www.cs.williams.edu/~andrea/DSReportInitialFull.pdf. Accessed 9 May 2019.

[Acm22] ACM Education Policy Committee (2018). Lighting the Path from Community College to Computing Careers. Available at https://www.acm.org/binaries/content/assets/education/lighting-the-path-from-community-college-to-computing-careers.pdf.

[And1] Anderson, L. W., & Krathwohl, D.R. (2001). *A taxonomy for learning, teaching and assessing: A revision of Bloom's Taxonomy of educational objectives*. New York: Longman.

[Ash1] Ashenhurst, R. L. (1972). Curriculum Recommendations for Graduate Professional Programs in Information Systems. *Communications of the ACM*, *15*(5), 364–398.

[Atc2] Atchison, W. F. (1981). Computer Education, Past, Present, and Future. *ACM SIGCSE Bulletin*, *13*(4), 2–6. http://doi.org/10.1145/989306.989307

[Bar1] Barr, Valerie, and Chris Stephenson. 2011. “Bringing Computational Thinking to K-12: What Is Involved and What Is the Role of the Computer Science Education Community?” ACM Inroads 2 (1): 48.

[Big1] Biggs. Approaches to the enhancement of tertiary teaching. Higher Education Research and Development 8, 1 (1989), 7–25.

[Cas2] Cassel, Lillian N., Robert H. Sloan, Gordon Davies, Heikki Topi, and Andrew McGettrick. 2007. “The Computing Ontology Project: The Computing Education Application.” SIGCSE Bull. 39 (1): 519–20.

[Cou1] Council of Chief State School Officers. 2013. “Knowledge, Skills, and Dispositions: The Innovation Lab Network State Framework for College, Career, and Citizenship Readiness, and Implications for State Policy.” https://files.eric.ed.gov/fulltext/ED542708.pdf.

[Cle1] Alison Clear, Allen Parrish, Gerrit C. van der Veer, Ming Zhang. (2017) *CC2020: A Vision on Computing Curricula*. Panel, Proceedings of the 2017 ACM SIGCSE Technical Symposium, DOI: 10.1145/3017680.3017690

[Con] Conte, S. D., Hamblen, J. W., Kehl, W. B., Navarro, S. O., Rheinboldt, W. C., Young, D. M., Jr, & Atchinson, W. F. (1965). An Undergraduate Program in Computer Science—Preliminary Recommendations. *Commun. ACM*, *8*(9), 543–552. http://doi.org/10.1145/365559.366069

[Cog1] Couger, J. D. (1973). Curriculum recommendations for undergraduate programs in information systems. *Communications of the ACM*, *16*(12), 727–749. http://doi.org/10.1145/362552.362554

[Cou2] Couger, J. D., Davis, G. B., Dologite, D. G., Feinstein, D. L., Gorgone, J. T., Jenkins, A. M., et al. (1995). IS'95: Guideline for undergraduate IS curriculum. *Mis Quarterly*, 341–359.

[Dav1] Davis, G. B., Gorgone, J. T., Couger, J., Feinstein, D. L., & Longenecker, H. E., Jr. (1996). IS'97: model curriculum and guidelines for undergraduate degree programs in information systems. *ACM SIGMIS Database*, *28*(1), 101–194.

[Dav2] Davis, Martin (2011) *The Universal Computer: The Road from Leibniz to Turing*. New York: A K Peters/CRC Press, 2011, ISBN 9781466505209.

[Eds1] Edström, Kristina. 2017. “Exploring the Dual Nature of Engineering Education: Opportunities and Challenges in Integrating the Academic and Professional Aspects in the Curriculum.” Edited by Anette Kolmos. PhD in Technology and Learning, Stockholm, Sweden: KTH Royal Institute of Technology. http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-217315.

[Fre3] Frezza, Stephen T., Alison Clear, and Abhijat M. Vichare. 2018. “Voices on the Core of Computing.” In 48th Frontiers in Education Conference (FIE’18), 9. IEEE-CS.

[Ful1] Fuller, Ursula, and Bob Keim. 2007. “Should We Assess Our Students’ Attitudes?” In Proceedings of the Seventh Baltic Sea Conference on Computing Education Research, 187–90. Koli National Park, Finland: ACM.

[Fin1] Fincher, S., & Petre, M. (2014). Computer Science Education Research. CRC Press.

[For1] Forsythe, G. E. (1967). A University's Educational Program in Computer Science. *Commun. ACM*, *10*(1), 3–11. http://doi.org/10.1145/363018.363038

[Gor1] Gorgone, J., & Gray, P. (2000). MSIS 2000: model curriculum and guidelines for graduate degree programs in information systems. *Communications of the AIS*, *3*(1es), 1.

[Gor2] Gorgone, J. T., Davis, G. B., Valacich, J. S., Topi, H., Feinstein, D. L., & Longenecker, H. E. (2002). IS 2002 model curriculum and guideli1es for undergraduate degree programs in information systems. *Communications of the Association for Information Systems*, *11*(1).

[Gor3] Gorgone, J. T., Gray, P., Stohr, E. A., Valacich, J. S., & Wigand, R. T. (2006). MSIS 2006: Model Curriculum and Guidelines for Graduate Degree Programs in Information Systems. *Communications of the AIS*, *17*(17), 1–58.

[Gru1] Gruner, Stefan (2011) "Problems for a Philosophy of Software Engineering," *Minds & Machines*, p. 25, February 2011.

[Gib1] Gibbs, G. Teaching students to learn: A student-centered approach. Open University Press (1981).

[Gos1] Gosselin, David. 2017. “Competencies vs Learning Outcomes.” Strengthen Workforce Prep in Your Program. 2017. https://serc.carleton.edu/integrate/programs/workforceprep/competencies_and_LO.html.

[Got1] Gottipati, Swapna, and Venky Shankararaman. 1/2018. “Competency Analytics Tool: Analyzing Curriculum Using Course Competencies.” Education and Information Technologies 23 (1): 41–60.

[Hay1] Hayashiguchi, E., O. Endou, and J. Impagliazzo. 2018. “The ‘i Competency Dictionary’ Framework for IT Engineering Education.” In 2018 IEEE World Engineering Education Conference (EDUNINE), 1–6.

[Hub1] Hubwieser, Peter, and Sue Sentance. 2018. “Taxonomies and Competency Models.” In Computer Science Education: Perspectives on Teaching and Learning in School, edited by Sue Sentance, Erik Barendsen, and Carsten Schulte, 1st ed., 221–42. London: Bloomsbury Academic.

[Jaf1] Jafar, M., Waguespack, L., & Babb, J. (2017). A Visual Analytics Approach to Gain Insights into the Structure of Computing Curricula, In *Proc. of 2017 EDSIG Conference*.

[Liu1] Liu, K. (2000). *Semiotics in Information Systems Engineering.* Cambridge University Press: Cambridge (2000).

[Koe1] Koeppen, Karoline, Johannes Hartig, Eckhard Klieme, and Detlev Leutner. 2008. “Current Issues in Competence Modeling and Assessment.” Swiss Journal of Psychology: Official Publication of the Swiss Psychological Society Schweizerische Zeitschrift Fur Psychologie = Revue Suisse de Psychologie 216 (2): 61–73.

[Kem1] Kember, D., and Kwan, K.-P. Lecturers’ approaches to teaching and their relationship to conceptions of good teaching. Instructional Science 28 (2000), 469–490. 10.1023/A:1026569608656.

[Kro1] Kroeze, Jan H. 2010. “Ontology Goes Postmodern in ICT.” In Proceedings of the 2010 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists, 153–59. SAICSIT ’10. New York, NY, USA: ACM.

[Leb] LeBlanc, R. J., Sobel, A., Diaz-Herrera, J. L., Hilburn, T. B., others. (2006). *Software Engineering 2004: Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering*. IEEE Computer Society.

[Lon] Longenecker, H. E., Jr., Feinstein, D. L., Fournier, R. L., Doran, M. V., & Reaugh, W. R. (1991). IS’90: The DPMA Model Curriculum for Information Systems for 4 Year Undergraduates. *Park Ridge, Illinois: Data Processing Management Association*.

[Lun] Lunt, B. M., Ekstrom, J. J., Gorka, S., Hislop, G., Kamali, R., Lawson, E., et al. (2008). *Information Technology 2008: Curriculum Guidelines for Undergraduate Degree Programs in Information Technology*. New York, NY: ACM.

[Lan1] Lang, James D., Susan Cruse, Francis D. McVey, and John McMasters. 1999. “Industry Expectations of New Engineers: A Survey to Assist Curriculum Designers.” Journal of Engineering Education 88 (1): 43–51.

[Loz1] Lozano, J. Felix, Alejandra Boni, Jordi Peris, and Andrés Hueso. 2012. “Competencies in Higher Education: A Critical Analysis from the Capabilities Approach.” Journal of Philosophy of Education 46 (1): 132–47.

[Lyn1] Lynch, Daniel R., Jeffrey S. Russell, Jeffrey C. Evans, and Kevin G. Sutterer. 2009. “Beyond the Cognitive: The Affective Domain, Values, and the Achievement of the Vision.” Journal of Professional Issues in Engineering Education and Practice 135 (1): 47–56.

[Nun1] Nunamaker, J. F., Couger, J. D., & Davis, G. B. (1982). Information systems curriculum recommendations for the 80s: undergraduate and graduate programs. *Communications of the ACM*, *25*(11), 781–805. http://doi.org/10.1145/358690.358698

[Pas1] Passow, Honor J. 01/2012. “Which ABET Competencies Do Engineering Graduates Find Most Important in Their Work?” Journal of Engineering Education 101 (1): 95–118.

[Sab2] Sabin, Mihaela, Svetlana Peltsverger, Bill Paterson, Ming Zhang, and Hala Alrumaih. 2017. “IT2017 Report: Putting It to Work.” In Proceedings of the 18th Annual Conference on Information Technology Education, 95–96. SIGITE ’17. New York, NY, USA: ACM.

[Sch1] Schussler, Deborah L. 2006. “Defining Dispositions: Wading Through Murky Waters.” The Teacher Educator 41 (4): 251–68.

[Shi1] Shiveley, James, and Thomas Misco. 2010. “‘But How Do I Know About Their Attitudes and Beliefs?’: A Four-Step Process for Integrating and Assessing Dispositions in Teacher Education.” The Clearing House: A Journal of Educational Strategies, Issues and Ideas 83 (1): 9–14.

[Sic1] Sicilia, Miguel-Angel. 2010. “How Should Transversal Competence Be Introduced In Computing Education?” ACM SIGCSE Bulletin 41 (4): 95–98.

[Smi1] Smith, Brad and Harry Shum (2018) The Future Computed: Artificial Intelligence and its Role in Society. Microsoft Corporations. 1 Microsoft Way. Redmond, Washington 98052. ISBN 978-0-9997508-1-0

[Spe1] Spector, Alfred Z. (2017, August) Changing Nature of Computer Science and Its Impact on Undergraduate Education. [Online]. http://sites.nationalacademies.org/cs/groups/cstbsite/documents/webpage/cstb_173998.pdf

[Ted3] Tedre, M., Simon, & Malmi, L. (2018). Changing aims of computing education: a historical survey. *Computer Science Education*, *28*(2), 158–186.

[Tra1] The Royal Academy of Engineering, The Institution of Engineering and Technology, and The British Computer Society. 2009. “Engineering Values in IT.” 3 Carlton House Terrace, London: The Royal Academy of Engineering. https://www.raeng.org.uk/publications/reports/engineering-values-in-it.

[Top1] Topi, H., Valacich, J. S., Wright, R. T., Kaiser, K., Nunamaker, J. F., Sipior, J. C., & de Vreede, G. J. (2010). IS 2010: Curriculum Guidelines for Undergraduate Degree Programs in Information Systems. *Communications of the Association for Information Systems*, *26*(18). Retrieved from http://aisel.aisnet.org/cais/vol26/iss1/18/

[Top2] Topi, Heikki, Helena Karsten, Sue A. Brown, João Alvaro Carvalho, Brian Donnellan, Jun Shen, Bernard C. Y. Tan, and Mark F. Thouin. 2017. "MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems." The Joint ACM/AIS MSIS 2016 Task Force. *40*(1). https://doi.org/10.1145/3127597. Retrieved from http://aisel.aisnet.org/cais/vol40/iss1/18

[Tri1] Trigwell, K., Prosser, M., and Waterhouse, F. Relations between teachers' approaches to teaching and students' approaches to learning. Higher Education 37 (1999), 57–70.

[Tre1] Trevelyan, J. 2012. "Understandings of Value in Engineering Practice." In 2012 Frontiers in Education Conference Proceedings, 1–6. Seattle, WA, USA: IEEE.

[Ute1] UTEC-1. https://education.spc.org.pe/Peru/CS-UTEC/Plan%202018/CS-UTEC-poster.pdf

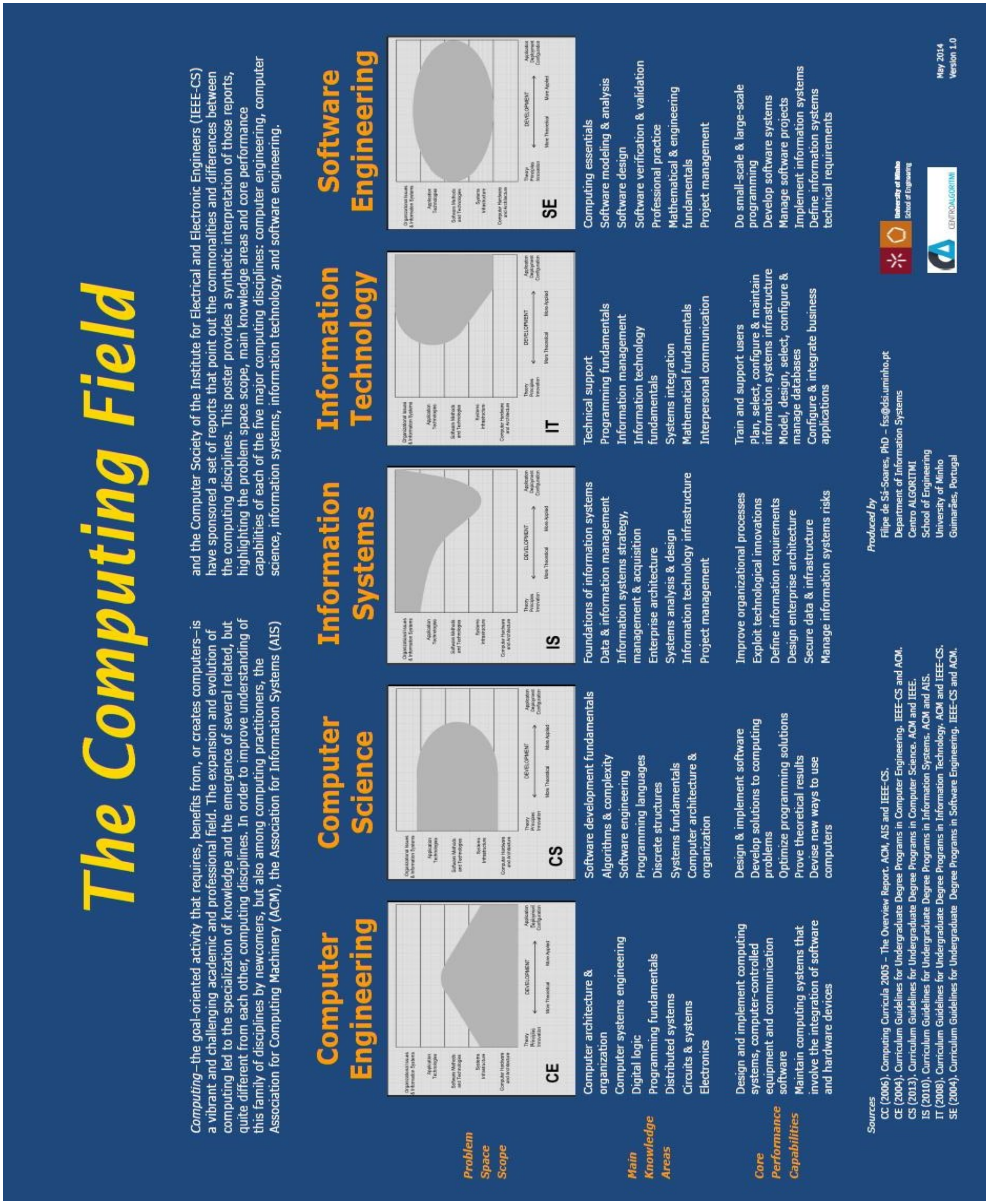
[Ute2] UTEC-2. https://education.spc.org.pe/Peru/CS-UTEC/Plan%202018/3_9_Compatibilidad_carrera_.html

[Win1] Winterton, Jonathan, Françoise Delamare- Le Deist, and Emma Stringfellow. 2006. Typology of Knowledge, Skills and Competences. Clarification of the Concept and Prototype: Cedefop References. CEDEFOP Reference Series. Luxembourg: Office for Official Publications of the European Communities.

[Usd1] 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. Accessed 14 May 2019.

[Wag2] Waguespack, L., Babb, J. (2019). Toward Visualizing Computing Curricula: The Challenge of Competency. In *Information Systems Education Journal*, 17(4) pp 51-69. http://isedj.org/2019-17/ ISSN: 1545-679X. (A preliminary version appears in *Proceedings of EDSIGCON 2018* <http://proc.iscap.info/2018/pdf/4633.pdf>)

Appendix A: Poster Explaining CC2005 Curricular Visuals



(*Courtesy of Filipe de Sá-Soares, PhD - University of Minho, Portugal*)

Appendix B: Computing Skills Frameworks

Different technology and informational skills are presented in a report by the IEEE Computer Society [Cos1]. This *Guide to the Enterprise Information Technology Body of Knowledge* (EITBOK) [Ent1] is a compendium of high-level knowledge areas typically required for the successful delivery of computing services vital to all enterprises. EITBOK defines the key knowledge areas for the IT (computing) profession, and it embodies concepts recognized as good practice in the IT domain and that are applicable to most IT efforts. The report emphasizes competence on a global scale. Frameworks enable the identification of skills and competencies required to perform duties and fulfill responsibilities in an enterprise IT workplace. Among the frameworks discussed are the Skills Framework for the Information Age (SFIA), the European Competency Framework (e-CF), and the i Competency Dictionary (iCD) of Japan. SFIA and e-CF had a major influence on the MSIS2016 report.

B.1: Skills Framework for the Information Age

The SFIA skills and competency framework was first published in 2000 and for the last 15 years has been truly global with use in over 180 countries. It is available in 10 languages: English, German, Spanish, Arabic, Japanese, Chinese, French, Polish, Italian and French Canadian. Originally developed for the UK and built on initiatives from the 1980s, SFIA was designed, from the very beginning, specifically to address the needs of industry and business and enable them to plan, acquire and develop the skills and competencies they need. The SFIA Framework is a global common language for skills and competencies and underpins the skills and competency management processes that organizations use to ensure they have the skills and competencies they need [Fig. B.1.].

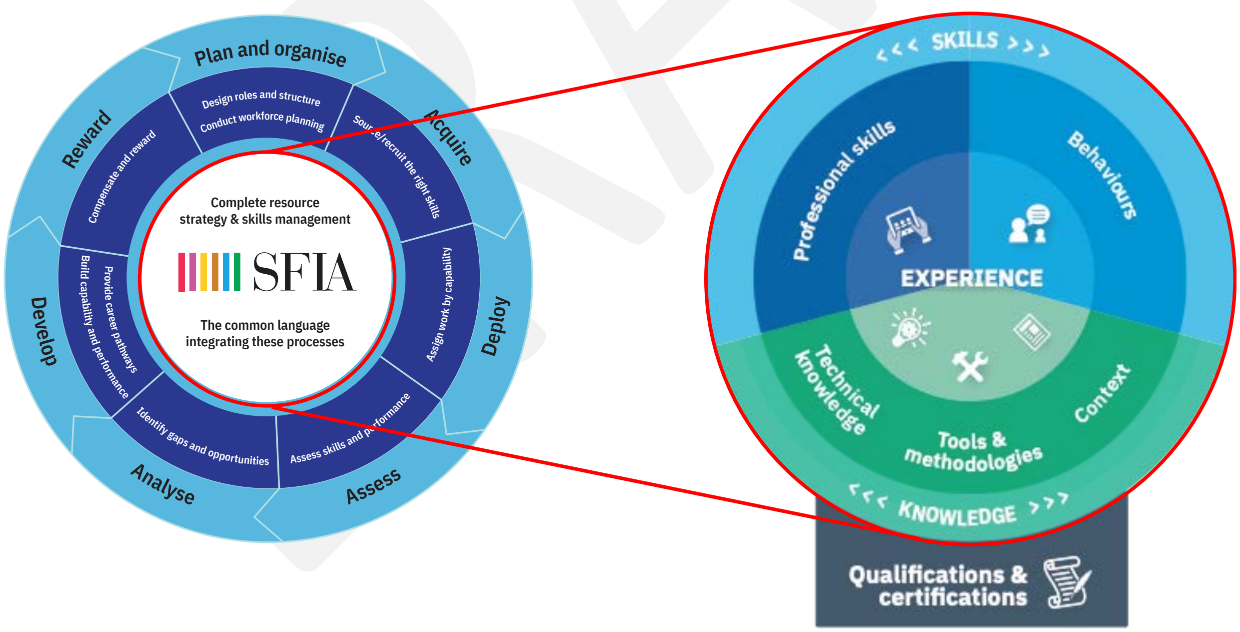


Figure B.1. The Context for SFIA – Supporting Skills and Competency Development in Industry

The not-for-profit SFIA Foundation was formed to maintain the SFIA Framework and support the global SFIA Ecosystem. There is a Governance Board which includes the British Computer Society and Institute of Engineering and Technology, an international SFIA Council which has industrial representation to lead the Foundation activities and an international Design Authority Board to oversee and approve updates to the SFIA Framework and its support assets. SFIA receives no central funding from government nor commercial stakeholder groups. It is funded through a license model which allows the majority of users (organizations and individuals) to use SFIA under a free-of-charge license. There is a modest annual license fee for large distributed organizations and for commercial exploitation. The SFIA Framework and all SFIA support assets are visible and available from the SFIA website at www.sfia-online.org

The SFIA Framework is updated using a well-established open consultation process involving volunteers throughout industry, throughout the world, for the benefit of the IT (computing) industry and IT professionals, for example, the IEEE Computer Society were significant contributors to the software engineering updates in 2018. A release is delivered every 3 years to reflect the changing skills and competency needs of industry through this open consultation process. SFIA 7, the most recent release was delivered in 2018. Consultation for SFIA 8 is in progress and release is scheduled for 2021Q3. Anyone, from any technical domain or country can contribute to the update of the SFIA Framework.

The SFIA skills and competency framework brings together a number of elements which industry needs including Professional Skills, Behaviors, Knowledge, Qualifications and Certifications with a focus on experience of performing a skill or competence as that is what industry values [Fig B.1.]. An individual has a skill at a particular level because they have demonstrated experience of performing that skill (skill, behavior and knowledge) at that level in a real-world situation.

The SFIA Framework has 7 Levels of Responsibility (as one dimension), characterized by 5 Generic Attributes (Autonomy, Influence, Complexity, Knowledge and Business Skills) each described at every SFIA Level (as a second dimension) [Fig.B.2.]. SFIA then identifies and defines 102 Professional Skills across the breadth of IT and supporting areas (as a third dimension) at each appropriate SFIA Level [Fig B.3.].

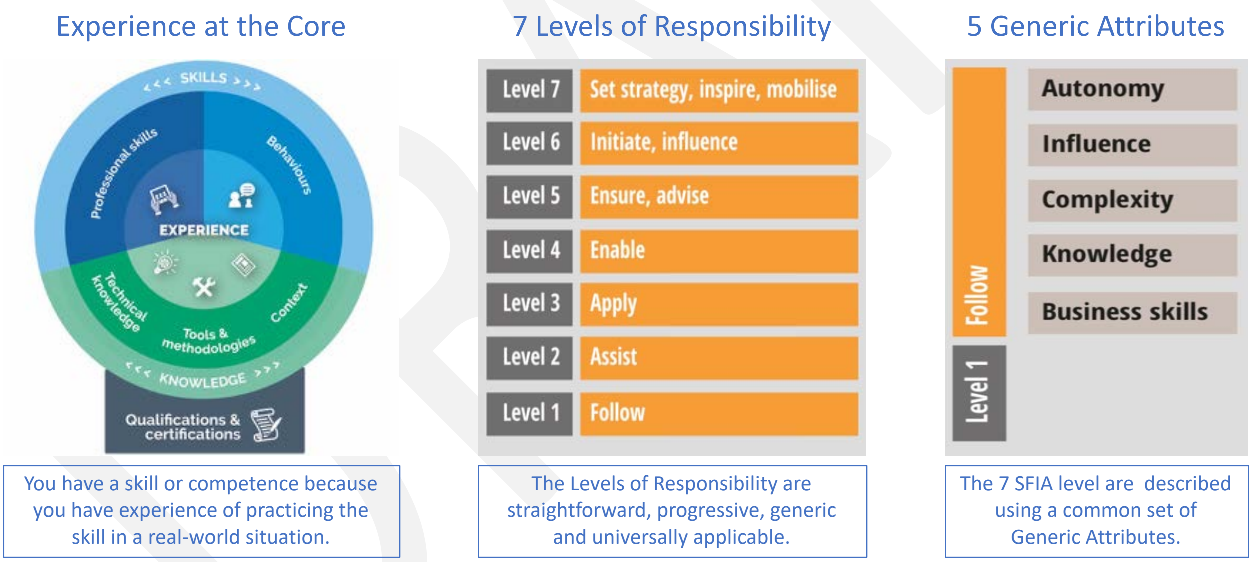


Figure B.2. The SFIA Context – Experience at the core

The SFIA Framework explicitly recognizes knowledge but does not define the specific knowledge required because it is highly context sensitive (technologies, domain, tools and methodologies, approaches and national requirements such as local legal requirements), knowledge also changes rapidly. SFIA therefore, in addressing knowledge, links to many bodies of knowledge that might be appropriate – depending on the context. Currently, links to 37 BoKs are provided on the SFIA website including the EITBOK, SWEBOK, SEBOK, NIST, CYBOK etc.)

The SFIA Framework recognizes the importance and place for qualifications and certifications but does not list specific requirements. This is because they are highly context sensitive and may only reflect knowledge recall (rather than verification of experience). While SFIA doesn't specify particular qualifications and certifications, many professional bodies use SFIA as the basis of their professional certification schemes usually at two levels; Technician, SFIA Level 3 and Chartered or Certified at SFIA Level 5, one professional body also uses SFIA Level 7 for their CIO certification.

The SFIA skills and competencies are commonly grouped by category and sub-category (Fig.B.3] – but these are purely for ease of navigation throughout the framework. A consistent structure and style is also adopted to further aid navigation: a skill, for instance, is described with a Skill Name, a Skill Code, a Skills Description (independent of the level), and then Skill Level Descriptors to describe that skill at each appropriate level.

The standard view of SFIA [Fig B.3.] is what would be widely recognized globally. The SFIA Foundation has also published alternative SFIA Views for particular contexts, these include: Digital Transformation View, Software Engineering View, DevOps View, Information/Cyber Security View, Big Data/Data Science View, other SFIA Views are in preparation. These SFIA Views are available from the SFIA website.

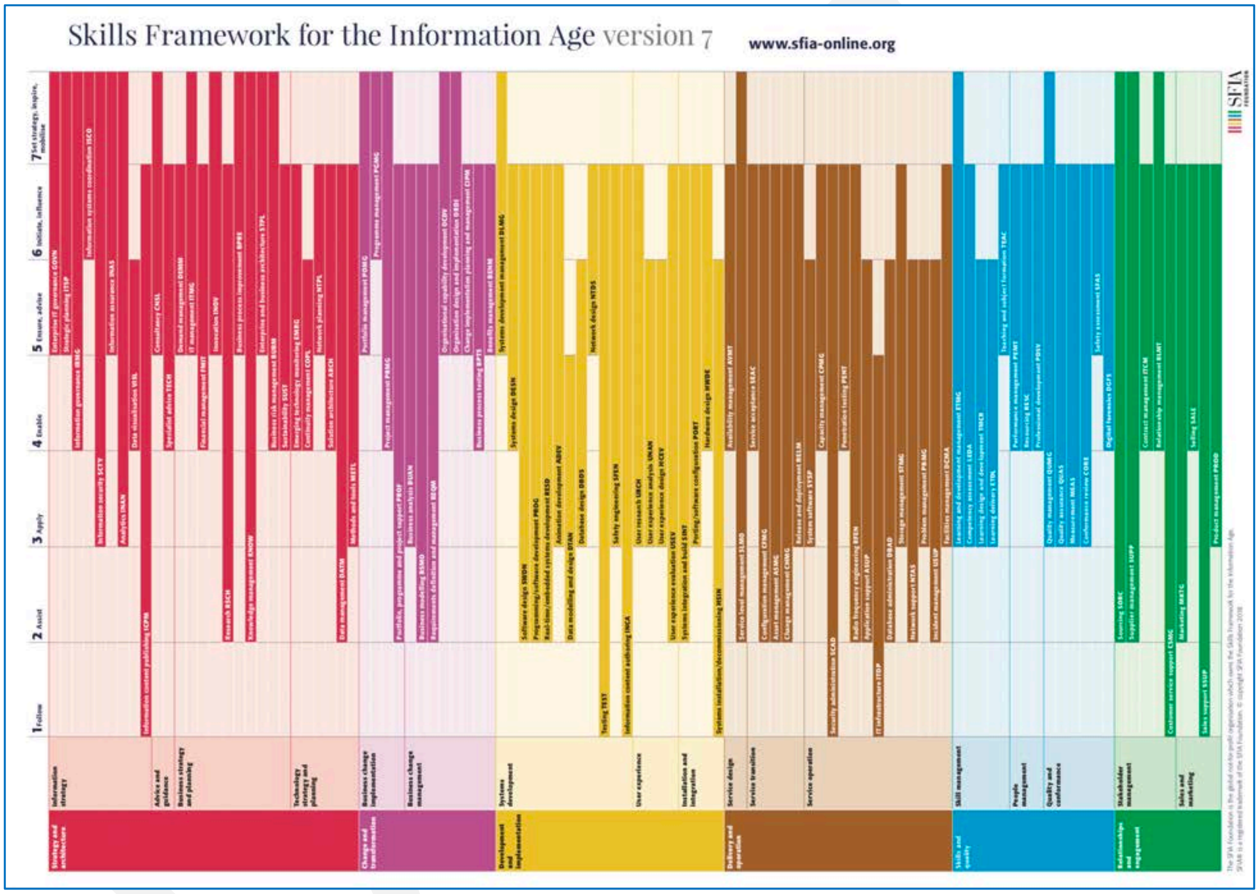


Figure B.3. The 102 SFIA Professional Skills – Skills and Competencies Performed by a Role or Individual

A key aspect of the SFIA Framework and the global SFIA Ecosystem is its openness – all SFIA assets are readily available from the SFIA website and this includes the core documentation set; the Complete SFIA Reference, the SFIA Summary Chart and the SFIA Framework in an Excel file (in the 10 languages). This enables organizations to build their own internal skills and competency support portals or upload SFIA into their corporate human resources or learning and development systems.

The SFIA Framework is easily extendable to cover other areas (outside of EIT/ICT) and many organizations do this internally or make suggestions during each SFIA refresh. SFIA's openness works both ways; just as the SFIA Foundation makes the SFIA Framework available it also welcomes ideas for enhancement and refresh and effort to contribute to content authoring, review and support.

B.2: Skills and the European Competency Framework

The European e-Competence Framework (e-CF) from the European Union provides a reference of 40 competencies required for performance in the ICT workplace, using a common language for competencies, knowledge, skills, and proficiency levels that can be understood across Europe. The use of the e-CF by companies and organizations throughout Europe supports the transparency, mobility, and efficiency of ICT-sector-related human resources planning and development.

As the first sector-specific implementation of the European Qualifications Framework (EQF), the e-CF can be used by ICT service, demand and supply organizations, and by managers and human resources departments. Additionally, they are useful for educational institutions and training bodies, including higher education, professional associations, trade unions, market analysts and policy makers, and other organizations and parties in public and private sectors. The structure of the framework is based on four dimensions shown in Figure B.2.

There are five e-CF proficiency levels, e-1 to e-5, which relate to EQF learning levels 3 to 8. Table B.2 shows a description of the EQF levels [Eur3].

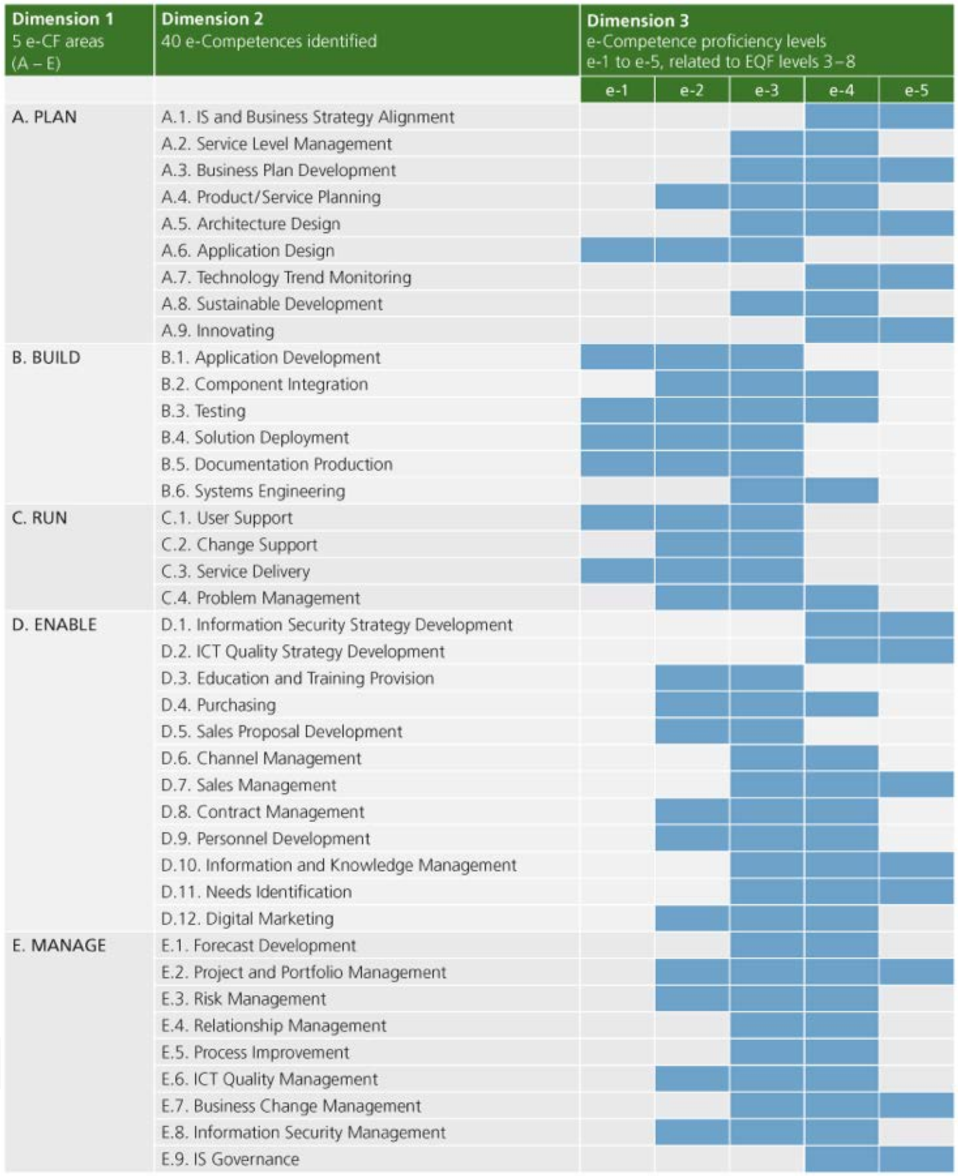
Dimension 1	Five e-Competence areas derived from the ICT business macro-processes PLAN – BUILD – RUN – ENABLE – MANAGE. The main aim of dimension 1 is to facilitate navigation through the framework.
Dimension 2	A set of reference e-Competences for each area, with a generic description for each competence. Forty competences identified in total provide the European generic reference definitions of the framework.
Dimension 3	Proficiency levels of each e-Competence provide European reference level specifications on e-Competence levels e-1 to e-5, which are related to EQF levels 3-8.
Dimension 4	Samples of knowledge and skills relate to e-Competences in dimension 2. They are provided to add value and context and are not intended to be exhaustive.

Figure B.2. Four dimensions of e-CF framework

Table B.2

e-Competence Level	EQF Level
5 (highest)	8
4	7
3	6
2	4 and 5
1	3

As in SFIA, not all skills are subject to all five levels. Figure B.3 shows the spread of competency levels for each skill.



Dimension 1 5 e-CF areas (A – E)	Dimension 2 40 e-Competences identified	Dimension 3 e-Competence proficiency levels e-1 to e-5, related to EQF levels 3–8				
		e-1	e-2	e-3	e-4	e-5
A. PLAN	A.1. IS and Business Strategy Alignment				■	■
	A.2. Service Level Management			■	■	
	A.3. Business Plan Development			■	■	■
	A.4. Product/Service Planning		■	■	■	
	A.5. Architecture Design			■	■	■
	A.6. Application Design	■	■	■		
	A.7. Technology Trend Monitoring				■	■
	A.8. Sustainable Development			■	■	
	A.9. Innovating				■	■
B. BUILD	B.1. Application Development	■	■	■		
	B.2. Component Integration		■	■	■	
	B.3. Testing	■	■	■	■	
	B.4. Solution Deployment	■	■	■		
	B.5. Documentation Production	■	■	■		
	B.6. Systems Engineering			■	■	
C. RUN	C.1. User Support	■	■	■		
	C.2. Change Support		■	■		
	C.3. Service Delivery	■	■	■		
	C.4. Problem Management		■	■	■	
D. ENABLE	D.1. Information Security Strategy Development				■	■
	D.2. ICT Quality Strategy Development				■	■
	D.3. Education and Training Provision		■	■		
	D.4. Purchasing		■	■	■	
	D.5. Sales Proposal Development		■	■		
	D.6. Channel Management			■	■	
	D.7. Sales Management			■	■	■
	D.8. Contract Management		■	■	■	
	D.9. Personnel Development		■	■	■	
	D.10. Information and Knowledge Management			■	■	■
	D.11. Needs Identification			■	■	■
	D.12. Digital Marketing		■	■	■	
E. MANAGE	E.1. Forecast Development			■	■	
	E.2. Project and Portfolio Management		■	■	■	■
	E.3. Risk Management		■	■	■	
	E.4. Relationship Management			■	■	
	E.5. Process Improvement			■	■	
	E.6. ICT Quality Management		■	■	■	
	E.7. Business Change Management			■	■	■
	E.8. Information Security Management		■	■	■	
	E.9. IS Governance				■	■

Figure B.3. The European Competency Framework Overview

B.3: Skills and the i Competency Dictionary

The i Competency Dictionary (iCD) was developed and is maintained by the Information Technology Promotion Agency (IPA) in Japan. It consists of a comprehensive Task Dictionary and a corresponding Skill Dictionary. The Task Dictionary contains all the tasks that EIT (Enterprise Information Technology) outsourcers or EIT departments are expected to accomplish, while the corresponding Skill Dictionary provides the skills required to perform those tasks.

The diagrams in Figure B.4 through Figure B.9 show how the task and skill dictionaries are structured to be used together. The skills needed to become competent at each task are enumerated in a Task vs. Skill table. The diagrams indicate the number of tasks and skills that are included in the full iCD. It is possible to obtain the complete iCD Task Dictionary (Layers 1-4) and Skill Dictionary (Layers 1-4) from the IPA website [Ipa1].

B.3.1: Task Dictionary

The Task Dictionary is intended to be used and applied by companies and organizations to determine tasks in line with their organizational strategies or organization plans. Tasks are used to define their organizational functions and the roles of personnel. The structure of the dictionary assumes a wide range of corporate activities, so that companies with any kind of business model can use and apply it. The Task Dictionary is comprised of four layers divided into three task layers plus the Task Evaluation Items layer, shown in Figure B.4.

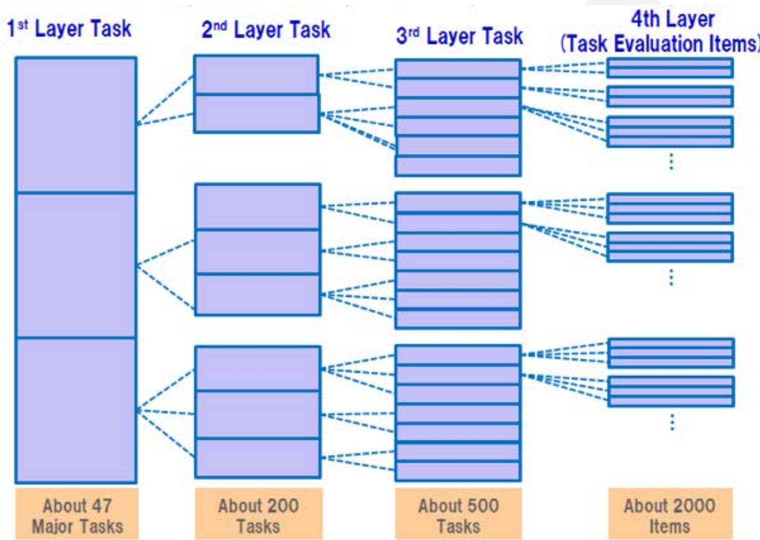


Figure B.4. The iCD Task Dictionary Structure

B.3.2: Task Dictionary Chart

The Task Dictionary Chart (Figure B.5) can be used to obtain a bird's-eye view of the entire Task Dictionary on the 1st Layer Task level. This chart presents a task structure composed of the organization lifecycle as vertical line (Strategy, Planning, Development, Utilization, Evaluation & Improvement) and tasks associated with entire lifecycle as horizontal line (Planning & Execution, Management & Control, Promotion & Support).

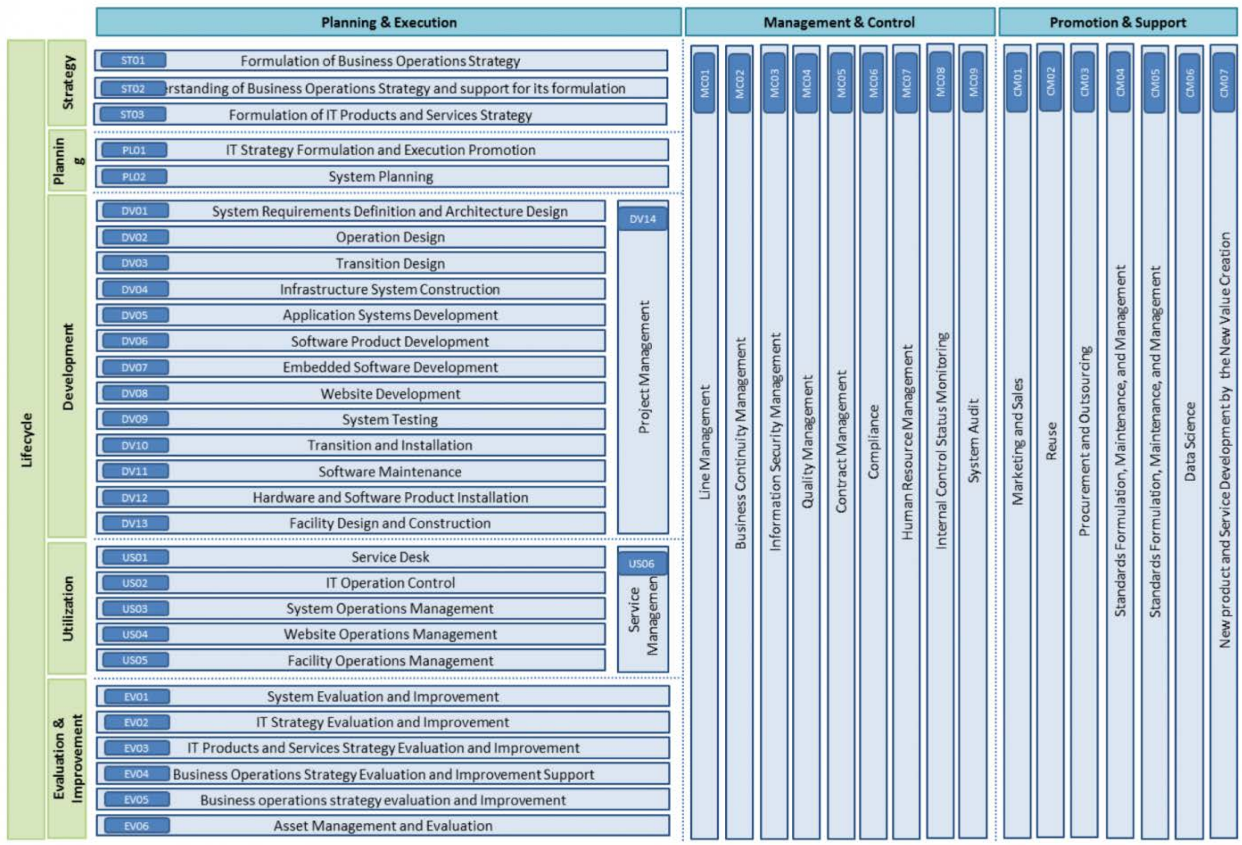


Figure B.5. The iCD Task Dictionary Chart

B.3.3: Examples of Task Evaluation Diagnostic Level and Criteria

Figure B.6 associates the task Diagnostic Level with Diagnostic Criteria. Diagnostic Criteria can be applied to task evaluation items or appropriate layer's tasks to evaluate one's task performance capability. The levels are from L0 to L4. This Diagnostic Criteria can be applied to individuals and the total task performance capability is obtained for each department by aggregating all department members' results.

Diagnostic Level	Diagnostic Criteria
L0	No knowledge or experience
L1	Has knowledge based on training
L2	Can carry out with support or has such experience
L3	Can carry out independently or has such experience
L4	Can instruct others or has such experience

Figure B.6. Examples of Task Evaluation Diagnostic Level and Criteria

B.3.4: Skill Dictionary

Skills are capabilities required to handle associated knowledge items to execute a task. The Skill Dictionary is comprised of four layers divided into three skill layers plus Associated Knowledge Items, shown in Figure B.7. The Skill Dictionary refers and sorts the items from the major Body of Knowledges/processes and skill standards in the world.

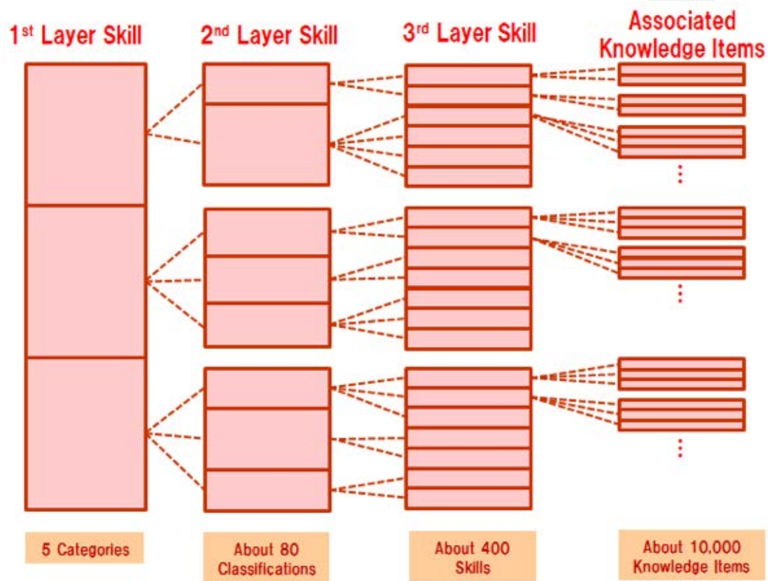


Figure B.7. The iCD Skill Dictionary Structure

B.3.5: Skill Dictionary Chart

The Skill Dictionary Chart (Figure B.8) can be used to obtain a bird's-eye view of the entire Skill Dictionary on the 1st and 2nd skill layers. The Skill Dictionary is divided into five categories based on the skill characteristics: methodology, technology, related knowledge, IT (human) skills, and specific skill (optional). This chart represents a skill structure on the perspectives of the IT orientation (Horizontal line: High-Low) and the application area (Vertical line: Wide-Narrow).

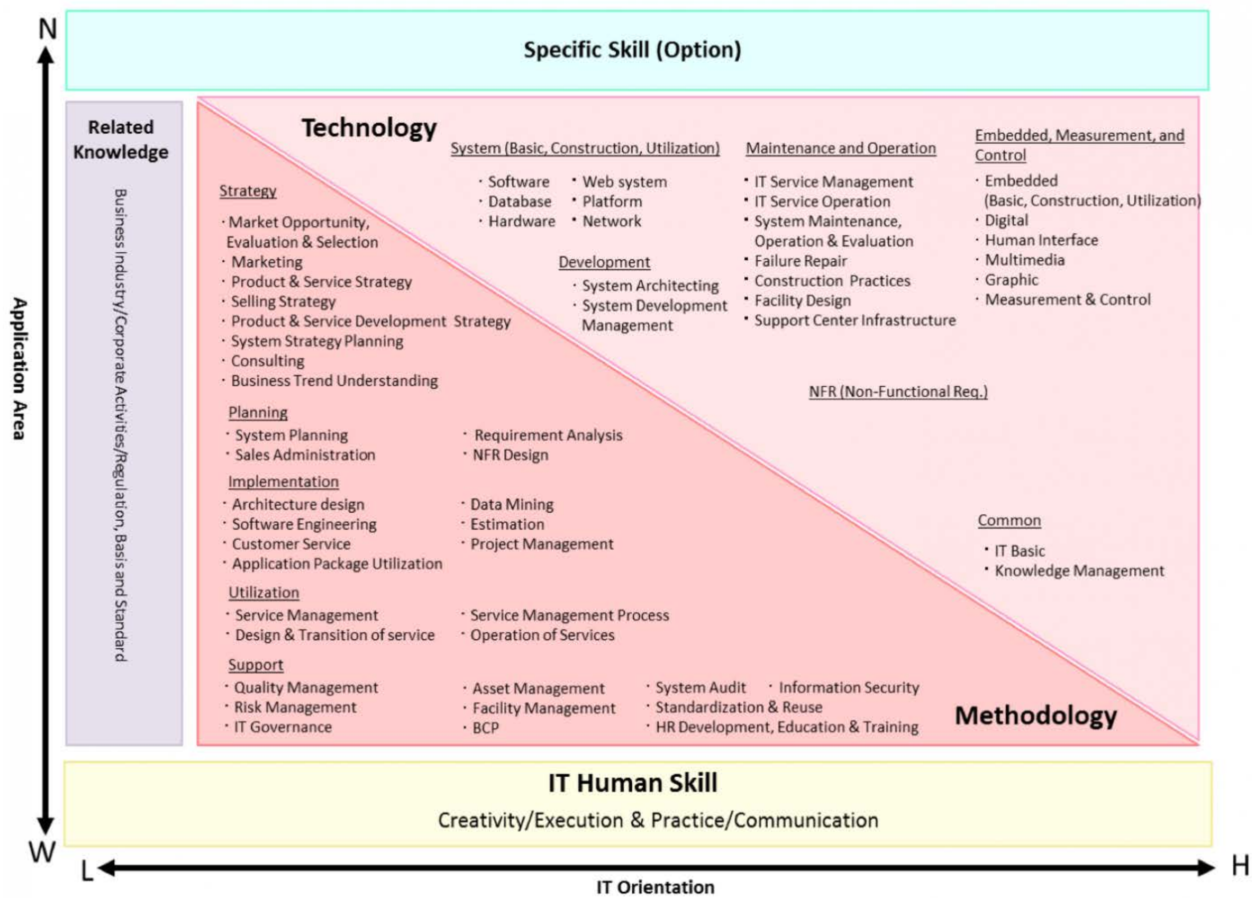


Figure B.8. The iCD Skill Dictionary Chart

B.3.6: Skill Proficiency Level

The chart in Figure B.9 measures the skill proficiency level using seven levels of skill proficiency criteria. Level 1 to 4 criteria differ according to contents of technology/methodology/related knowledge. Skill proficiency level 4 is the highest acquisition level for the skill of accomplishing tasks. Level 5 to 7 criteria are defined across the categories and are evaluated based on the degree of social contribution as a professional.

Level 7	Skills at the level of an industry leader who has influence on the market		
Level 6	Skills at the level of a recognized contributor to the industry		
Level 5	Skills at the level of a recognized contributor within affiliated associations and organizations		
Level 4	Level at which one is able to produce optimal solutions that take into account non-functional requirements, step outside of established tactics, and pass the advanced information technology examinations	Has mastered and can select the most suitable methods, and can freely apply the methods according to the situation	Is able to discuss what needs to be done with senior management within the industry or business they are involved in
Level 3	Is able to create functional requirements and to work independently under limited circumstances	Is able to apply the proper method according to the problem, and has utilized the methods on-site and drawn conclusions	Has proposed solutions to the IT-related problem points in the industry and businesses they are involved in
Level 2	Has implementation experience, and is able to use and apply the technology if instructions are available	Is able to perform analysis using the method, or is able to use the methodology under guidance	Understands the IT-related problem points in the industry and businesses they are involved in
Level 1	Has knowledge, and understands lectures and presentations of technical content	Understands lectures and presentations about the method, understands and can explain what it is, and understands textbooks about it	Understands and can explain what kind of industry and business they are involved in, and understands public information such as securities reports
Category	Technology	Methodology	Related Knowledge

Figure B.9. Skill Proficiency Level

B.4: Skills via Enterprise Information Technology

The emphasis on competence has become international as Enterprise IT (EIT) and ICT in general have become indispensable across the globe. EIT and ICT derive from a growing understanding of the need for a common language for competencies, knowledge, skills and proficiency levels that can be understood across national borders. A common framework enables the identification of skills and competencies that may be required to successfully perform duties and fulfill responsibilities in an EIT workplace. They provide a common basis for the selection and recruitment of EIT staff, as well as forming the basis for employment agreements, professional development plans, and performance evaluation for ICT professionals.

Many national and regional governments have come to require certification of EIT practitioners. Accordingly, they have had to develop their own definitions of ICT competencies. Given the increasingly international composition of the EIT workforce, the EITBOK has included information from three major frameworks that are emerging as inter-regional. In general, these frameworks work towards a common understanding of competence, defined by the e-CF, for example, as “demonstrated ability to apply knowledge, skills and attitudes to achieve observable results."

B.5: References for Appendix B

[Cos1] Computer Society of IEEE. http://www.computer.org/. Accessed 2019 May 6.
[Ent1] Enterprise Information Technology Body of Knowledge (EITBOK). http://eitbokwiki.org/Main_Page. Accessed 2019Aug17.
[SFIA1] SFIA Framework – SFIA Foundation: https://sfia-online.org/en . Accessed 2020
[Ipa1] Information-technology Promotion Agency – Japan (IPA). IT Human Resources Development: i Competency Dictionary. https://www.ipa.go.jp/english/humandev/icd.html. Accessed 2019Aug18.
[Eur3] European Qualifications Framework (EQF). European Commission. Learning Opportunities and Qualifications. https://ec.europa.eu/ploteus/en/content/descriptors-page/. Accessed 2019Aug17.

Appendix C: Preliminary Draft Competencies – Examples

At the onset of the CC2020 project, subgroups of task force experts explored the development of competency statements for different computing disciplines. To accomplish this task, these experts used an *implied process* to generate draft competencies. The IT2017 report already specified competencies by vacating the knowledge area/unit approach and by eliminating learning outcomes and topics. It published these competencies within sets of essential and supplemental domain clusters.

This appendix provides a first-pass approach to generate draft competencies for computer engineering, computer science, information systems, and software engineering. It also includes the published competencies from the IT2017 report as well as draft competencies for a master's program in information systems. These competencies, which task force subgroups created in 2017-2018, did not use the cluster model as described in Chapter 4. Instead, task force subgroups used a "common sense approach" as described in Section C.1 below.

C.1: Initial CC2020 Explorations of Competencies

For each of the established computing reports (IS2010, CS2013, SE2014, CE2016), the respective experts on the project teams had used knowledge-based strategies rather than competency approaches. The combined output resulted in thousands of learning outcomes derived from these published reports. Mastering all learning outcomes in a discipline is unattainable.

C.1.1: Drafting Competencies

By using a structured or algorithmic approach, some members of the CC2020 task force transformed essential learning outcomes from the four undergraduate curricular reports into competencies. This activity was not easy because of the novelty of the meaning of competency and because of the innovative use of competency in computing education. Notwithstanding, the CC2020 steering committee created a focus group and partitioned them into subgroups, each identified with one of the computing disciplines mentioned above.

In 2017-2018, each subgroup used the IT2017 canonical definition that

Competency = Knowledge + Skills + Dispositions in context.

Over six months, the subgroups prototyped competencies for their respective computing disciplines. The number of draft competency statements for each discipline varied; three-dozen was a target number. The IT2017 report already had 47 stated competencies so it was not part of these subgroup activities, although its published competencies are part of this appendix.

The subsections in section C.2 describe the results of the work from the competency subgroups for computer engineering, computer science, information systems, and software engineering, as well as those published for information technology. It also includes generated competencies from a master's in information systems (MSIS2016) report. Before summarizing the generated competencies, what follows describes the procedure or "algorithm" used to formulate draft competency statements.

C.1.2: Strategy for Generating Competencies

Since the CC2020 report focuses on undergraduate programs, it is best to use non-related curricular guidelines to serve as a model or example for this task. In this case, the MSIS2016 report was a good candidate. From it, the curricular area "Business Continuity and Information Assurance (BCIA)" can serve as an example. Page 16 of the MSIS2016 report has a stated area described as follows.

The Business Continuity and Information Assurance competency area mainly concerns the continuity, auditing, and assurance of information systems. It generally covers areas such as risk avoidance, security management, and quality auditing. The challenging issues related to business continuity and information assurance span from tactical and strategic to technical and operational levels. They often involve a range of processes from management, such as policy and standard-setting, to hands-on skills, such as system contingency and recovery planning.

From this description, a first attempt to generate BCIA draft competencies could be the following set.

A. Analyze policies and standards for business continuity and information assurance and present the findings to a group of peers.
B. Plan procedures, operations, and technologies for managing security and safety in a disaster recovery situation.
C. Monitor the protection and growth of hardware and software within an information system for a small company.

Note that for each competency, the action verbs (analyze, plan, and monitor) depict the skills needed. The knowledge needed is in the content of the activity. The notion of dispositions occurs in the context of the activities such as presenting to a group of peers, producing useful procedures, or monitoring activities in a small company.

Of course, many other possibilities exist, and competency sets are not absolute or unique. The set of draft competencies for BCIA must consider the context of development. Hence, two different graduate programs could easily have different sets of competencies.

Also note that in 2018, to generate competencies for BCIA or any computing area requires that the competencies satisfy the canonical definition of competency, which is Knowledge + Skills + Dispositions in context or task. *Knowledge* derives from the corpus of IS content. *Skills* are the activities taken with knowledge to create an accomplishment. *Dispositions* are the collective human attributes or characteristics expected of a professional who practices computing and information systems in the workplace.

A simple search on "dispositions" produces dozens of attributes, including the eleven dispositions stated in Table 4.3 of Chapter 4. The assumption made by the authors of the IT2017 report and the members of the four subgroups is that each person (graduate) possesses these innate characteristics, but with different emphases or proportions. For example, for the "punctuality" dispositional attribute, both Graduate X and Graduate Y possess this attribute. However, one Graduate X may be more inclined to be punctual while Graduate Y may be a better team player.

C.2: Draft Competencies by Discipline

This section describes the results of the work from the four competency subgroups for computer engineering, computer science, information systems, and software engineering completed in 2018 May. It also includes the area of information technology and a graduate program in information systems. It does not include cybersecurity because the subgroups began their work before that project concluded.

C.2.1: Computer Engineering Draft Competencies

The computer engineering material that follows contains two versions of the same CE competencies. The CE subgroup had several discussions on whether it should include the dimension of "disposition" as a self-contained statement or embed dispositions within each competency statement. The left column shows the former version with (human) disposition in Item B. The right column shows the latter version with embedded dispositions. The CE task force is neutral on which is the preferred representation.

Self-contained Disposition Version	*Embedded Disposition Version*
For each Knowledge Area: A. Communicate the essential elements of the history of computer engineering, including the development of tools, standards, and constraints to a technical audience. [*History & overview; relevant tools, standards, constraints*] B. Exercise all CE competencies in a contextually appropriate manner, demonstrating proper consideration of ethics, cultures, background, and human relationships. [*Dispositions - the human element*]	For each Knowledge Area: A. Communicate the essential elements of the history of computer engineering, including the development of tools, standards, and constraints to a technical audience. [*History & overview; relevant tools, standards, constraints*]
CE-CAE — Circuits and Electronics 1. Analyze and design circuits using electronic devices, and innovate in the context of new and existing systems using those components to create new functions on varying levels of complexity bearing in mind the tradeoffs involved. [*History & Overview; Tools & standards; electrical quantities, elements & circuits; electronic materials & devices; MOS transistors; data storage cells*]	**CE-CAE — Circuits and Electronics** 1. Analyze and design circuits for a local engineering company using electronic devices and innovate in the context of new and existing systems using those components to create new functions on varying levels of complexity bearing in mind the tradeoffs involved. [*History & Overview; Tools & standards; electrical quantities, elements & circuits; electronic materials & devices; MOS transistors; data storage cells*]
CE-CAL — Computing Algorithms 1. Design and/or implement classic and application-specific algorithms including parallel in multi-threading ones by relevant tools within engineering, marketing, commercial or legal constraints in the respectful and meaningful interaction with users and customers. [*Relevant tools; algorithms - common ones, analysis, strategies*] 2. Analyze correctness, efficiency, performance, and complexity of the algorithms using order of complex terms and present honestly and comprehensively the results of the analysis for either a professional or non-professional audience. [*Algorithmic complexity; scheduling algorithms; computability theory*]	**CE-CAL — Computing Algorithms** 1. Design and/or implement classic and application-specific algorithms including parallel in multi-threading ones by relevant tools within engineering, marketing, commercial or legal constraints in the respectful and meaningful interaction with users and customers. [*Relevant tools; algorithms - common ones, analysis, strategies*] 2. Analyze correctness, efficiency, performance, and complexity of the algorithms using order of complex terms and present honestly and comprehensively the results of the analysis for either a professional or non-professional audience. [*Algorithmic complexity; scheduling algorithms; computability theory*]
CE-CAO — Computer Architecture & Organization 1. Manage the design of computer hardware components and integrate such components to provide complete hardware systems that function reliably and efficiently demonstrating sensitivity for the context of the design envelope within which they were conceived. [*Measuring performance; Processor organization; Distributed systems architecture; Multi/Many-core architectures; Peripheral subsystems*] 2. Simulate and evaluate the performance of parallel and sequential hardware solutions and tradeoffs involved in designing complex hardware systems considering the	**CE-CAO — Computer Architecture & Organization** 1. Manage the design of computer hardware components for a multidisciplinary research project and integrate such components to provide complete hardware systems that function reliably and efficiently demonstrating sensitivity for the context of the design envelope within which they were conceived. [*Measuring performance; Processor organization; Distributed systems architecture; Multi/Many-core architectures; Peripheral subsystems*] 2. Present a report that discusses the simulation and evaluation of the performance of parallel and sequential hardware solutions and tradeoffs involved in designing

Self-contained Disposition Version	*Embedded Disposition Version*
design of memory and arithmetical units as well as characterizing system performance using appropriate metrics. [*Processor organization; Memory system organization & architecture; Computer arithmetic; Input/Output interfacing and communication*]	complex hardware systems considering the design of memory and arithmetical units as well as characterizing system performance using appropriate metrics. [*Processor organization; Memory system organization & architecture; Computer arithmetic; Input/Output interfacing and communication*]
CE-DIG — Digital Design 1. Using appropriate tools, design digital circuits including the basic building blocks of Boolean algebra, computer numbering systems, data encoding, combinatorial and sequential elements. [*Tools & standards; numbering systems & data encoding; Boolean algebra; digital logic, combinatorial & sequential*] 2. Design a control or datapath circuit using programmable logic and considering relevant system design constraints and testability concerns. [*Control & datapaths; programmable logic; system constraints; fault models & testing*]	**CE-DIG — Digital Design** 1. Manage the design of a computer system for a manufacturer using appropriate tools, design digital circuits including the basic building blocks of Boolean algebra, computer numbering systems, data encoding, combinatorial and sequential elements. [*Tools & standards; numbering systems & data encoding; Boolean algebra; digital logic, combinatorial & sequential*] 2. Design a control or datapath circuit for a small company using programmable logic and considering relevant system design constraints and testability concerns. [*Control & datapaths; programmable logic; system constraints; fault models & testing*]
CE-ESY — Embedded Systems 1. Design and/or implement basic and advanced I/O techniques, both synchronous and asynchronous and serial/parallel, including interrupts and time considerations. [*Parallel/ serial I/O; synchronous/asynchronous I/O; interrupts and timing*] 2. Design and implement an example of an embedded system in a non-electronic device, including sensor feedback, low-power, and mobility. [*Data acquisition & sensors; embedded systems characteristics; low-power operation*]	**CE-ESY — Embedded Systems** 1. Present to a group of peers the design and implementation of basic and advanced I/O techniques, both synchronous and asynchronous and serial/parallel, including interrupts and time considerations. [*Parallel/ serial I/O; synchronous/asynchronous I/O; interrupts and timing*] 2. Design and implement for a professional seminar an example of an embedded system in a non-electronic device, including sensor feedback, low-power, and mobility. [*Data acquisition & sensors; embedded systems characteristics; low-power operation*]
CE-NWK — Computer Networks 1. Develop, deploy, maintain, and evaluate the performance of wireless and wired networking solutions in the context of relevant standards and the needs of stakeholder groups and demonstrating awareness of the foundations and history of the area. [*History and overview; Relevant tools, standards*] 2. Relate general networking competence to integrated solutions in the Internet of Things considering security and privacy aspects and the impact of solutions on citizens and society. [*Network architecture; Local and wide-area networks; Network protocols; Network applications; Network management; Data communications; Performance evaluation; Wireless sensor networks*]	**CE-NWK — Computer Networks** 1. Develop, deploy, maintain and evaluate the performance of wireless and wired networking solutions for a manufacturer in the context of relevant standards and the needs of stakeholder groups and demonstrating awareness of the foundations and history of the area. [*History and overview; Relevant tools, standards*] 2. Relate general networking competence to integrated solutions in the Internet of Things considering security and privacy aspects and the impact of solutions on citizens and society. [*Network architecture; Local and wide-area networks; Network protocols; Network applications; Network management; Data communications; Performance evaluation; Wireless sensor networks*]
CE-PPP — Preparation for Professional Practice 1. Analyze the importance of communication skills in a team environment and within a computer engineering group setting, discuss and determine how these skills contribute to the optimization of organization goals. [*Communication and teamwork*] 2. Evaluate the philosophical and cultural attributes necessary for maintaining a global relationship in solving a computer engineering problem that involves a	**CE-PPP — Preparation for Professional Practice** 1. Analyze the importance of communication skills in a team environment and within a computer engineering group setting, discuss and determine how these skills contribute to the optimization of organization goals. [*Communication and teamwork*] 2. Evaluate the philosophical and cultural attributes necessary for maintaining a global relationship in solving a computer engineering problem that involves a

Self-contained Disposition Version	*Embedded Disposition Version*
system development in a political context. [*Philosophical, cultural and societal issues*] 3. Develop hardware policies that include professional, legal, and ethical considerations as they relate to a global engineering company. [*Professional, ethical, and legal issues*] 4. Evaluate contemporary issues facing a computer engineering project and develop an effective project plan using business acumen and cost/benefit analysis. [*Contemporary, business, and management issues*]	system development in a political context. [*Philosophical, cultural and societal issues*] 3. Develop hardware policies that include professional, legal, and ethical considerations as they relate to a global engineering company. [*Professional, ethical, and legal issues*] 4. Evaluate contemporary issues facing a computer engineering project and develop an effective project plan using business acumen and cost/benefit analysis. [*Contemporary, business, and management issues*]
CE-SEC — Information Security 1. Evaluate the current cybersecurity tools for their effectiveness in providing data security, side-channel attacks, and integrity while avoiding vulnerabilities, both technical and human-factor caused. [*Data security and integrity; Vulnerabilities; Network and web security; Side-channel attacks*] 2. Design a cybersecurity solution that provides resource protection, public, and private key cryptography, authentication, network, and web security, and trusted computing. [*Resource protection models; Secret and public-key cryptography; Message authentication codes; Authentication; Trusted computing*]	**CE-SEC — Information Security** 1. Write a report on the evaluation of the current cybersecurity tools for their effectiveness in providing data security, side-channel attacks, and integrity while avoiding vulnerabilities, both technical and human-factor caused. [*Data security and integrity; Vulnerabilities; Network and web security; Side-channel attacks*] 2. Design a cybersecurity solution for a network company that provides resource protection, public, and private key cryptography, authentication, network, and web security, and trusted computing. [*Resource protection models; Secret and public-key cryptography; Message authentication codes; Authentication; Trusted computing*]
CE-SGP — Signal Processing 1. Design signal processing systems applying knowledge of sampling and quantization to bridge the analog and digital domains. [*Transform analysis; frequency response; sampling & aliasing; spectra*] 2. Evaluate signal processing challenges (e.g., detection, denoising, interference removal) to support the selection and implementation of appropriate algorithmic solutions including non-recursive and recursive filters, time-frequency transformations, and window functions. [*Relevant tools, standards & constraints; convolution; window functions; multimedia processing; control systems*]	**CE-SGP — Signal Processing** 1. Design signal processing systems with an engineering team by applying knowledge of sampling and quantization to bridge the analog and digital domains. [*Transform analysis; frequency response; sampling & aliasing; spectra*] 2. Evaluate signal processing challenges (e.g., detection, denoising, interference removal) to support the selection and implementation of appropriate algorithmic solutions including non-recursive and recursive filters, time-frequency transformations, and window functions, and present the results to an electrical engineering team. [*Relevant tools, standards & constraints; convolution; window functions; multimedia processing; control systems*]
CE-SPE — Systems and Project Engineering 1. Manage a project that requires the analysis of a system (hardware and software), including system requirements, both technical (including functional and performance requirements) and in terms of suitability, usability and inclusiveness, taking a holistic perspective to craft specifications and evaluating reliability. [*Project management principles; User experience; Risk, dependability, safety & fault tolerance; Requirements analysis and elicitation; Hardware and software processes; System specifications; System architecture design and evaluation; Concurrent hardware and software design; System integration, testing, and validation; Maintainability, sustainability, manufacturability*]	**CE-SPE — Systems and Project Engineering** 1. Manage a project for an organization that requires the analysis of a system (hardware and software), including system requirements, both technical (including functional and performance requirements) and in terms of suitability, usability, and inclusiveness, taking a holistic perspective to craft specifications and evaluating reliability. [*Project management principles; User experience; Risk, dependability, safety & fault tolerance; Requirements analysis and elicitation; Hardware and software processes; System specifications; System architecture design and evaluation; Concurrent hardware and software design; System integration, testing, and validation; Maintainability, sustainability, manufacturability*]

Self-contained Disposition Version	*Embedded Disposition Version*
CE-SRM — Systems Resource Management 1. Analyze the role of single user, mobile, networked, client-server, distributed, and embedded operating systems, interrupts, and real-time support in managing system resources and interfacing between hardware and software elements considering economic, environmental, and legal limitations. [*History and overview of operating systems, Managing system resources, Operating systems for mobile devices, Support for concurrent processing*] 2. Design and implement an appropriate performance monitoring procedure for standard and virtual systems. [*Real-time operating system design, System performance evaluation; Support for virtualization*]	**CE-SRM — Systems Resource Management** 1. Analyze the role of single user, mobile, networked, client-server, distributed, and embedded operating systems, interrupts, and real-time support in managing system resources and interfacing between hardware and software elements considering economic, environmental, and legal limitations. [*History and overview of operating systems, Managing system resources, Operating systems for mobile devices, Support for concurrent processing*] 2. Preset to an organization the design and implementation of appropriate performance monitoring procedures for standard and virtual systems. [*Real-time operating system design, System performance evaluation; Support for virtualization*]
CE-SWD — Software Design 1. Evaluate and apply programming paradigms and languages to solve a wide variety of software design problems being mindful of trade-offs including maintainability, efficiency, and intellectual property constraints. [*Programming constructs & paradigms; problem-solving; history & overview; relevant tools, standards, constraints*] 2. Design software tests for evaluating a wide variety of performance criteria on subsystems (including usability, correctness, graceful failure, and efficiency) within the context of a complete hardware-software system. [*Software testing & quality*]	**CE-SWD — Software Design** 1. Write a report for a manufacturer regarding the evaluation and application of programming paradigms and languages to solve a wide variety of software design problems being mindful of trade-offs including maintainability, efficiency, and intellectual property constraints. [*Programming constructs & paradigms; problem-solving; history & overview; relevant tools, standards, constraints*] 2. Design software testing procedures for an engineering team that evaluates a wide variety of performance criteria on subsystems (including usability, correctness, graceful failure, and efficiency) within the context of a complete hardware-software system. [*Software testing & quality*]

Number of Draft Competencies = 24

Task Force Members on the CE Subgroup

Barry Lunt (Leader)
Olga Bogyavlenskaya
Eric Durant
John Impagliazzo
Arnold Neville Pears

C.2.2: Computer Science Draft Competencies

AL-Algorithms and Complexity

A. Present to a group of peers the data characteristics of conditions or assumptions that can lead to different behaviors of specific algorithms and from the analysis, illustrate empirical studies to validate hypotheses about runtime measures.
B. Illustrate informally the time and space complexity of algorithms and use big-O notation formally to show asymptotic upper bounds and expected case bounds on time and space complexity, respectively.
C. Use recurrence relations to determine the time complexity of recursively defined algorithms by solve elementary recurrence relations and present the results to a group of scholars.
D. Determine an appropriate algorithmic approach to an industry problem and use appropriate techniques (e.g., greedy approach, divide-and-conquer algorithm, recursive backtracking, dynamic programming, or heuristic approach) that considers the trade-offs between the brute force to solve a problem.
E. Implement basic numerical algorithm methods (e.g., search algorithms, common quadratic and O(N log N) sorting algorithms, fundamental graph algorithms, string-matching algorithm) to solve an industry problem and select the appreciate algorithm for a particular context.
F. Design a deterministic finite state machine for a local engineering firm that accepts a specified language and generates a regular expression to represent the language.

AR-Architecture and Organization

A. Use CAD tools for capture, synthesis, and simulation to evaluate simple building blocks
B. of a simple computer design for a local engineering company.
C. Evaluate the timing diagram behavior of a simple processor-implemented at the logic circuit level and develop a report expressing the findings.
D. Write a simple program at the assembly/machine level for string processing and manipulation and for converting numerical data into hexadecimal form.
E. Implement a fundamental high-level construct in both machine and assembly languages and present the results to a group of peers.
F. Calculate the average memory access time under a variety of cache and memory configurations and develop a short report of the findings.

CN-Computational Science

A. Create a simple, formal mathematical model of a real-world situation and use that model in a simulation for a local technology company.

DS-Discrete Structures

A. Present to a peer group some practical examples of an appropriate set, function, or relation model, and interpret the associated operations and terminology in context.
B. Use symbolic propositional and predicate logic to model a real-life industry application by applying formal methods (e.g., calculating the validity of formulae and computing normal forms to the symbolic logic).
C. Apply rules of inference to construct proofs and present results to a group of professionals, appropriate proofs, or logical reasoning to solve a strategic problem.
D. Map real-world applications to appropriate counting formalisms and apply basic counting theories (e.g., counting arguments, the pigeonhole principle, modular arithmetic as well as compute permutations and combinations of a set) to solve an industry problem.
E. Analyze an industry problem to determine underlying recurrence relations and present the solution to professionals by using a variety of basic recurrence relations.
F. Model a real-world problem using appropriate graphing strategies (e.g., trees, traversal methods for graphs and trees, spanning trees of a graph) and determine whether two graph approaches are isomorphic.
G. Calculate different probabilities of dependent or independent events and expectations of random variables to solve a problem and present to a group of peers the ways to compute the variance for a given probability distribution.

GV-Graphics and Visualization

A. Design and develop a user interface using a standard API and that incorporates visual and audio techniques used for a local organization

HCI-Human-Computer Interaction

A. Design an interactive application, applying a user-centered design cycle with related tools and techniques (modes, navigation, visual design), to optimize usability and user experience within a corporate environment.

B. Analyze and evaluate a user interface that considers the context of use, stakeholder needs, state-of-the-art response interaction times, design modalities taking into consideration universal access, inclusiveness, assistive technologies, and culture-sensitive design.
C. Design and develop an interactive application for a local charity, applying a user-centered design cycle with related vocabulary, tools, and techniques that optimize usability and user experience.
D. Create and conduct a simple usability test to analyze and evaluate a user interface that considers the context of use taking into consideration universal access and culturally sensitive design.
E. Create a simple application, together with help and documentation, that supports a graphical user interface for an enterprise and conduct a quantitative evaluation and report the results.

IAS-Information Assurance and Security

A. Write the correct input validation code for a cybersecurity company after classifying common input validation errors.
B. Demonstrate to a group of security professionals ways to prevent a race condition from occurring and ways to handle exceptions.

IM-Information Management

A. Contrast information with data and knowledge and describe to a group of professionals the advantages and disadvantages of centralized data control.
B. Demonstrate to a group of peers a declarative query language to elicit information from a database.
C. Contrast appropriate data models, including internal structures, for different types of data, and present an application to a group of professionals the use of modeling concepts and notation of the relational data model.

IS-Intelligent Systems

A. Determine the characteristics of a given problem that an intelligent system must solve and present the results to a project team.
B. Formulate an industry problem specified in a natural language (e.g., English) as a constraint satisfaction problem and implement it using an appropriate technique (e.g., chronological backtracking algorithm or stochastic local search).
C. Implement an appropriate uninformed or informed search algorithm for an industry problem by characterizing time and space complexities of informed algorithm or designing the necessary heuristic evaluation function for an uninformed search algorithm to guarantee an optimal solution, respectively.
D. Translate a natural language (e.g., English) sentence for a corporate query system into a predicate logic statement by converting a logic statement into clause form and applying resolution to a set of logic statements to answer a query.
E. Make a probabilistic inference in a real-world industry problem using Bayes' theorem to determine the probability of a hypothesis given evidence.

NC-Networking and Communication

A. Design and develop for a corporate customer a simple client-server socket-based application.
B. Design and implement a simple reliable protocol for an industry network by considering factors that affect the network's performance.
C. Contrast fixed and dynamic allocation techniques as well as current approaches to congestion and present the results to company executives.

OS-Operating Systems

A. Apply knowledge of computing theory and mathematics to solve problems and present comprehensively the results and methods of the solution for either a professional or non-professional audience.
B. Implement software solutions within system constraints of a target system considering its abilities and constraints, and document and explain the implementation to both technical and non-technical audiences
C. Predict the behavior of systems under random events using knowledge of probability and expectation and inform users of its potential behavior.
D. Assess the security of a system using the knowledge of confidentiality, availability, and integrity with an understanding of risks, threats, vulnerabilities, and attack vectors, and relate its societal and ethical impact to the system's constituents.

PBD-Platform-based Development

A. Design for a client a responsive web application utilizing a web framework and presentation technologies in support of a diverse online community.
B. Develop a mobile app for a company that is usable, efficient, and secure on more than one device.
C. Simulate for a company an industry platform.
D. Develop and implement programming tasks via platform-specific APIs and present the results to a group of peers.
E. Present the analysis of a mobile industrial system and illustrate correct security vulnerabilities.

PD-Parallel and Distributed Computing

A. Design a scalable parallel algorithm for a computer firm by applying task-based decomposition or data-parallel decomposition.
B. Write a program for a client that correctly terminates when all concurrent tasks terminate by considering actors and/or reactive processes, deadlocks, and properly synchronized queues.

C. Write a test program for a company that reveals a concurrent programming error (e.g., missing an update when two activities both try to increment a variable).
D. Present computational results of the work and span in a program by identifying independent tasks that may be parallelized and determining the critical path for a parallel execution diagram.
E. Implement a parallel divide-and-conquer (and/or graph algorithm) for a client by mapping and reducing operations for the real industry problem and empirically measure its performance relative to its sequential analog.

PL-Programming Languages

A. Present the design and implementation of a class considering object-oriented encapsulation mechanisms (e.g., class hierarchies, interfaces, and private members).
B. Produce a brief report on the implementation of a basic algorithm considering control flow in a program using dynamic dispatch that avoids assigning to a mutable state (or considering reference equality) for two different languages.
C. Present the implementation of a useful function that takes and returns other functions considering variables and lexical scope in a program as well as functional encapsulation mechanisms.
D. Use iterators and other operations on aggregates (including operations that take functions as arguments in two programming languages and present to a group of professionals ways of selecting the most natural idioms for each language.
E. Contrast and present to peers (1) the procedural/functional approach (defining a function for each operation with the function body providing a case for each data variant) and (2) the object-oriented approach (defining a class for each data variant with the class definition providing a method for each operation).
F. Write event handlers for a web developer for use in reactive systems such as GUIs.
G. Demonstrate program pieces (such as functions, classes, methods) that use generic or compound types, including for collections to write programs.
H. Write a program for a client to process a representation of code that illustrates the incorporation of an interpreter, an expression optimizer, and a documentation generator.
I. Use type-error messages, memory leaks, and dangling-pointer to debug a program for an engineering firm.

SDF-Software Development Fundamentals

A. Create an appropriate algorithm to illustrate iterative, recursive functions, as well as divide-and-conquer techniques and use a programming language to implement, test, and debug the algorithm for solving a simple industry problem.
B. Decompose a program for a client that identifies the data components and behaviors of multiple abstract data types and implementing a coherent abstract data type, with loose coupling between components and behaviors.
C. Design, implement, test, and debug an industry program that uses fundamental programming constructs including basic computation, simple and file I/O, standard conditional and iterative structures, the definition of functions, and parameter passing.
D. Present the costs and benefits of dynamic and static data structure implementations, choosing the appropriate data structure for modeling a given engineering problem.
E. Apply consistent documentation and program style standards for a software engineering company that contribute to the readability and maintainability of software, conducting a personal and small-team code review on program component using a provided checklist.
F. Demonstrate common coding errors, constructing and debugging programs using the standard libraries available with a chosen programming language.
G. Refactor an industry program by identifying opportunities to apply procedural abstraction.

SE-Software Engineering

A. Conduct a review of a set of software requirements for a local project, distinguishing between functional and non-functional requirements, and evaluate the extent to which the set exhibits the characteristics of good requirements.
B. Present to a client the design of a simple software system using a modeling notation (such as UML), including an explanation of how the design incorporated system design principles.

SF-Systems Fundamentals

A. Design a simple sequential problem and a parallel version of the same problem using fundamental building blocks of logic design and use appropriate tools to evaluate the design for a commercial organization and evaluate both problem versions.
B. Develop a program for a local organization that incorporated error detection and recovery that incorporates appropriate tools for program tracing and debugging.
C. Design a simple parallel program for a corporation that manages shared resources through synchronization primitives and use tools to evaluate program performance.
D. Design and conduct a performance-oriented, pattern recognition experiment incorporating state machine descriptors and simple schedule algorithms for exploiting redundant information and data correction that is usable for a local engineering company and use appropriate tools to measure program performance.
E. Calculate average memory access time and describe the tradeoffs in memory hierarchy performance in terms of capacity, miss/hit rate, and access time for a local engineering company.

F. Measure the performance of two application instances running on separate virtual machines at a local engineering company and determine the effect of performance isolation.

SP-Social Issues and Professional Practice

A. Perform a system analysis for a local organization and present the results to them in a non-technical way.
B. Integrate interdisciplinary knowledge to develop a program for a local organization.
C. Document industry trends, innovations, and new technologies and produce a report to influence a targeted workspace.
D. Present to a group of professionals an innovative computer system by using audience-specific language and examples to illustrate the group's needs.
E. Produce a document that is helpful to others that addresses the effect of societal change due to technology.
F. Adopt processes to track customer requests, needs, and satisfaction
G. Compare different error detection and correction methods for their data overhead, implementation complexity, and relative execution time for encoding, detecting, and correcting errors and ensure that any error does not affect humans adversely.

Number of Draft Competencies = 84

Task Force Members on the CS Subgroup

Bruce McMillin (Leader)
John Impagliazzo
Richard LeBlanc
Ariel Sabiguero Yawelak
Ming Zhang

C.2.3: Information Systems Draft Competencies

Identifying and designing opportunities for IT-enabled organizational improvement

1. Analyze the current fit between IT strategy and organizational strategy and take corrective action to align the two, when necessary.
2. Understand General Systems theory, including its key principles and applications

~~3.~~ Model organizational processes with at least one modern business process modeling language.

4. Extract information systems requirements from future state process models.
5. Building on the foundation of risk-based management theory, apply risk analysis to real organizations.
6. Determine information systems requirements based on demonstrated needs for organizational controls.
7. Identify process performance indicators and monitors, applying industry recommendations like ITIL
8. Understand emerging technologies to identify innovative business opportunities based on these technologies.
9. Develop business proposals based on the use of emerging technologies in an organization.
10. Apply entrepreneurial and creative thinking to transform organizations using emerging technologies
11. Analyze and document various business stakeholders' information requirements for a proposed system.
12. Apply modern industrial practices and techniques on system documentation and user interviewing (i.e. ITIL and PMBOK).
13. Apply foundational knowledge of human-computer interaction principles to systems and user interface design.
14. Apply knowledge of data visualization and representation for an application related to analytics and complex data representation.

Analyzing trade-offs

15. Identify and design the technology alternatives and manage risk across various options within an information systems project to select the most appropriate options based on the organizational needs and implement a solution that solves key business problems.
16. Justify an information systems project in terms of technical feasibility, business viability, and cost-effectiveness to demonstrate the project's feasibility.
17. Analyze and compare solution options according to a variety of criteria and policies to evaluate the different possible solutions according to how well they promote the organizational needs.
18. Create a budget for IT-based solutions and sourcing options to enable the organization to determine the financial impact of each option.
19. Analyze the cultural differences that affect a global business environment to show how cultural standards and expectations can have a positive impact on business success to support the process of selecting between options.

Designing and implementing information systems solutions

20. Design an enterprise architecture (EA) using formal approaches by identifying EA change needs and by addressing domain requirements and technology development.
21. Apply a systematic methodology for specifying system solution options based on the requirements for the information systems solution, considering in-house development, development from third-party providers, or purchased commercial-off-the-shelf (COTS) packages.
22. Design and implement a high-quality UX (user experience) for target users to enable effective support for the users' goals in their environment.
23. Design principles of information technology security and data infrastructure at the organizational level that enable them to plan, develop, and perform security tasks and apply them to organizational systems and databases.
24. Design and implement an IT application that satisfies user needs in the context of processes that integrate analysis, design, implementation, and operations.
25. Identify data and information management alternatives and suggest the most appropriate options based on the organizational information needs.
26. Design data and information models aligned with organizational processes and compatible with data and information security management criteria.
27. Select the suitable outsourcing contractors based on the external procurement selection criteria and manage people in development teams including selected contractors in multiple projects and complex situations.
28. Understand the processes, methods, techniques, and tools that organizations use to manage information systems projects.
29. Implement modern project management approaches to information systems project, demonstrating an understanding of complex team-based activities are an inherent part of the project management.

Managing ongoing information technology operations

30. Develop and implement plans of action for optimizing the use of enterprise technology resources.
31. Develop indicators to assess application performance and scalability.
32. Monitor application performance indicators and implement corrective actions.
33. Establish practices for optimized use of information systems and plan for a long term IS viability.

34. Monitor and control an IS to track performance and fit with organizational needs.
35. Implement corrective actions by modifying the system as necessary.
36. Negotiate and enforce contracts with providers of technology service to maintain the operational integrity of the technologies and services provided and be compliant with the roles and responsibilities of all parties involved.
37. Develop, implement, and monitor a security plan strategy based on a risk management model.
38. Implement corrective security actions as necessary.
39. Plan and implement procedures, operations, and technologies for managing security and safety ensuring business continuity and information assurance from a disaster recovery situation.

Leadership and collaboration

40. Manage interpersonal relationships in a cross-cultural, cross-functional team.
41. Provide a clearly articulated vision for the team so that it will be able to work towards a common goal.
42. Support each member of the team in their effort to achieve their best possible level of individual performance.
43. Specify sufficiently challenging goals for the team.
44. Create a work breakdown structure, a task dependency model, and a project schedule for a globally distributed project.
45. Ensure that the project has sufficient resources and manage those resources in a context-appropriate way.
46. Allocate project tasks to project resources in an equitable and achievable way.
47. Monitor the progress the project is making.
48. Respect different viewpoints between team members.
49. View differences between team members as richness and a resource.
50. Listen and consider carefully to the viewpoints of all team members.
51. Establish and support decision structures that ensure equal opportunity to participate by all team members.
52. Align the structure of an organization so that it supports the achievement of its goals.
53. Select the organizational form based on criteria known to be effective.
54. Execute the transformation of an organization's structure so that it does not unnecessarily disrupt its work.
55. Monitor the effectiveness of an organizational structure continuously.

Communication

56. Acquire facts and opinions regarding the domain of interest from various stakeholders in relevant organizational contexts using appropriate communication methods.
57. Extract information from digital archives using modern data retrieval tools.
58. Communicate effectively in writing in a broad range of organizational contexts.
59. Select the appropriate form of written communication for a specific organizational situation.
60. Use state of the art virtual collaboration tools (such as wikis, blogs, and shared collaboration spaces) effectively in a variety of organizational situations.
61. Communicate effectively orally with different audiences and using different channels in a variety of organizational situations.
62. Identify and articulate the key elements of a persuasive presentation to support a specific viewpoint.

Negotiation

63. Apply a detailed problem analysis to determine the interests of each party in the negotiation to provide a clear proposal of the funding, time, and staff required.
64. Articulate and justify service levels for an IT service in terms of metrics that guarantee a description of the service being provided, the reliability, the responsiveness, the procedure for reporting problems, monitoring and reporting service level, consequences for not meeting service obligations, and escape clauses or constraints.
65. Demonstrate the specification and measurements for each area in the level of service definitions to allow the quality of service to be benchmarked.
66. Identify and apply a more positive and confident approach to negotiating for each provider to support the quality enhancement of the project design as well as to ensure quality project preparation and implementation.
67. Classify the key decision points, identify who is involved in making those decisions, and understand the actions and information that will be required for such decisions to be made within an information systems team in the context of competing internal interests.

Analytical and critical thinking, including creativity and ethical analysis

68. Interpret and comply with legislative and regulatory requirements governing IT practices as well as industry standards for IT practices. Understand how culture and ethics shape compliance behavior.
69. Analyze privacy and integrity guide for all IT practices.
70. Identify complex situations and analyze the practices guide to ensure the ethical and legal corporate requirements are met.
71. Identify the value of the systems.
72. Identify the system's vulnerabilities.
73. Identify the occurrence of a threat that may exploit a system vulnerability aimed at compromising the systems.
74. Identify a complex problem in, but separate from, its environment.
75. Apply knowledge and understanding to solve the identified problem.

76. Apply creative problem solving to technology-related issues.
77. Select appropriate data collection methods and techniques for the investigation of domain activities.
78. Capture and structure data and information requirements using appropriate conceptual modeling techniques.
79. Reason effectively with a learned audience based on the results of quantitative analyses.
80. Apply adequate quantitative analysis techniques according to the data analysis goal.
81. 81 Develop innovative and creative models that rely on new uses of existing technology or new technologies themselves.
82. Develop a plan to exploit new and emerging methods and technologies for new purposes within an organization.
83. Devise new ways of structuring and performing domain activities at different levels (individual, team, process, and organization) while considering the enabling and enhancing effects of information technology applications.
84. Estimate the benefits of the new designs, assess the consequences of their implementation, and anticipate potential adverse consequences.

.

Mathematical foundations

85. Identify those domains of interest problems that can be addressed mathematically and find a mathematical formulation for those problems.
86. Use logical thought processes to divide a problem into smaller components and make inferences based on problem components.
87. Select and implement an effective mathematical strategy.
88. Communicate mathematical results effectively to a variety of stakeholders. (See, for example, Turner 2010 at https://research.acer.edu.au/cgi/viewcontent.cgi?article=1083&context=resdev)

Number of Draft Competencies = 88

Task Force Members on the Information Systems Subgroup

Eiji Hayashiguchi (Leader)
Hala Alrumaih
Teresa Pereira
Ariel Sabiguero
Heikki Topi
John Impagliazzo

C.2.4: Information Technology Competencies

ITE-CSP Cybersecurity Principles [6%]

A. Evaluate the purpose and function of cybersecurity technology identifying the tools and systems that reduce the risk of data breaches while enabling vital organization practices. (*Cybersecurity functions*)
B. Implement systems, apply tools, and use concepts to minimize the risk to an organization's cyberspace to address cybersecurity threats. (*Tools and threats*)
C. Use a risk management approach for responding to and recovering from a cyber-attack on a system that contains high-value information and assets such as an email system. (*Response and risks*)
D. Develop policies and procedures needed to respond and remediate a cyber-attack on a credit card system and describe a plan to restore functionality to the infrastructure. (*Policies and procedures*)

ITE-GPP Global Professional Practice [3%]

A. Analyze the importance of communication skills in a team environment and determine how these skills contribute to the optimization of organization goals. (*Communication and teamwork*)
B. Evaluate the specific skills necessary for maintaining continued employment in an IT career that involves system development in an environmental context. (*Employability*)
C. Develop IT policies within an organization that include privacy, legal, and ethical considerations as they relate to a corporate setting. (*Legal and ethical*)
D. Evaluate related issues facing an IT project and develop a project plan using a cost/benefit analysis including risk considerations in creating an effective project plan from its start to its completion. (*Project management*)

ITE-IMA Information Management [6%]

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*)

ITE-IST Integrated Systems Technology [3%]

A. Illustrate how to code and store characters, images, and other forms of data in computers and show why data conversion is often a necessity when merging disparate computing systems. (*Data mapping and exchange*)
B. Show how a commonly used intersystem communication protocol works, including its advantages and disadvantages. (*Intersystem communication protocols*)
C. Design, debug and test a script that includes selection, repetition, and parameter passing. (*Integrative programming and scripting*)
D. Illustrate the goals of secure coding, and show how to use these goals as guideposts in dealing with preventing buffer overflow, wrapper code, and securing method access. (*Defensible integration*)

ITE-NET Networking [5%]

A. Analyze and compare the characteristics of various communication protocols and how they support application requirements within a telecommunication system. (*Requirements and Technologies)*
B. Analyze and compare several networking topologies in terms of robustness, expandability, and throughput used within a cloud enterprise. (*Technologies*)
C. Describe different network standards, components, and requirements of network protocols within a distributed computing setting. (*Network protocol technologies*)
D. Produce managerial policies to address server breakdown issues within a banking system. (*Risk Management*)
E. Explain different main issues related to network management. (*Network Management*)

ITE-PFT Platform Technologies [1%]

A. Describe how the historical development of hardware and operating system computing platforms produced the computing systems we have today. (*Computing systems)*
B. Show how to choose among operating system options, and install at least an operating system on a computer device. (*Operating systems*)

C. Justify the need for power and heat budgets within an IT environment, and document the factors needed when considering power and heat in a computing system. (*Computing infrastructure*)
D. Produce a block diagram, including interconnections, of the main parts of a computer, and illustrate methods used on a computer for storing and retrieving data. (*Architecture and organization*)

ITE-SPA System Paradigms [6%]

A. Justify the way IT systems within an organization can represent stakeholders using different architectures and the ways these architectures relate to a system lifecycle. (*Requirements and development*)
B. Demonstrate a procurement process for software and hardware acquisition and explain the procedures one might use for testing the critical issues that could affect IT system performance. (*Testing and performance*)
C. Evaluate integration choices for middleware platforms and demonstrate how these choices affect testing and evaluation within the development of an IT system. (*Integration and evaluation*)
D. Use knowledge of information technology and sensitivity to the goals and constraints of the organization to develop and monitor effective and appropriate system administration policies within a government environment. (*System governance*)
E. Develop and implement procedures and employ technologies to achieve administrative policies within a corporate environment. (*Operational activities*)
F. Organize personnel and information technology resources into appropriate administrative domains in a technical center. (*Operational domains)*
G. Use appropriate and emerging technologies to improve the performance of systems and discover the cause of performance problems in a system. (*Performance analysis*)

ITE-SWF Software Fundamentals [4%]

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*)

ITE-UXD User Experience Design [3%]

A. Design an interactive application, applying a user-centered design cycle and related tools and techniques (e.g., prototyping), aiming at usability and relevant user experience within a corporate environment. (*Design tools and techniques*)
B. For a case of user-centered design, analyze and evaluate the context of use, stakeholder needs, state-of-the-art interaction opportunities, and envisioned solutions, considering user attitude and applying relevant tools and techniques (e.g., heuristic evaluation), aiming at universal access and inclusiveness, and showing a responsive design attitude, considering assistive technologies and culture-sensitive design. (*Stakeholder needs*)
C. For evaluation of user-centered design, articulate evaluation criteria and compliance to relevant standards (*Benchmarks and standards*)
D. In design and analysis, apply knowledge from related disciplines including human information processing, anthropology and ethnography, and ergonomics/human factors. (*Integrative design*)
E. Apply experience design for a service domain related to several disciplines, focusing on multiple stakeholders and collaborating in an interdisciplinary design team. (*Application design)*

ITE-WMS Web and Mobile Systems [3%]

A. Design a responsive web application utilizing a web framework and presentation technologies in support of a diverse online community. (*Web application development*)
B. Develop a mobile app that is usable, efficient, and secure on more than one device. (*Mobile app development*)
C. Analyze a web or mobile system and correct security vulnerabilities. (*Web and mobile security*)
D. Implement storage, transfer, and retrieval of digital media in a web application with appropriate file, database, or streaming formats. (*Digital media storage and transfer*)
E. Describe the major components of a web system and how they function together, including the webserver, database, analytics, and front end. (*Web system infrastructure*)

Number of Draft Competencies = 47

C.2.5: Software Engineering Draft Competencies

Software Requirements

1. Identify and document software requirements by applying a known requirements elicitation technique in work sessions with stakeholders, using facilitative skills, as a contributing member of a requirements team.
2. Analyze software requirements for consistency, completeness, and feasibility, and recommend improved requirements documentation, as a contributing member of a requirements team.
3. Specify software requirements using standard specification formats and languages that have been selected for the project, and be able to describe the requirements in an understandable way to non-experts such as end-users, other stakeholders, or administrative managers, as a contributing member of a requirements team.
4. Verify and validate the requirements using standard techniques, including inspection, modeling, prototyping, and test case development, as a contributing member of a requirements team.
5. Follow process and product management procedures that have been identified for the project, as a contributing member of the requirements engineering team.

Software Design

1. Present to business decision-makers architecturally significant requirements from a software requirements specification document.
2. Evaluate and compare tradeoffs from alternative design possibilities for satisfying functional and non-functional requirements and write a brief proposal summarizing key conclusions for a client.
3. Produce a high-level design of specific subsystems that is presentable to a non-computing audience by considering architectural and design patterns.
4. Produce detailed designs for a client for specific subsystem high-level designs by using design principles and cross-cutting aspects to satisfy functional and non-functional requirements.
5. Evaluate software testing consideration of quality attributes in the design of subsystems and modules for a developer/manufacturer.
6. Create software design documents that communicate effectively to software design clients such as analysts, implementers, test planners, or maintainers.

Software Construction

1. Design and implement an API using an object-oriented language and extended libraries, including parameterization and generics on a small project.
2. Evaluate a software system against modern software practices such as defensive programming, error and exception handling, accepted fault tolerances, in a runtime mode that considers state-based table-driven constructions on a large project, as a member of a project team.
3. Develop a distributed cloud-based system that incorporates grammar-based inputs and concurrency primitives for a medium-size project and then conduct a performance analysis to fine-tune the system, as a member of a project team.

Software Testing

1. Perform an integrative test and analysis of software components by using black-box and use case techniques in collaboration with the clients.
2. Conduct a regressive test of software components for a client that considers operational profiles and quality attributes specific to an application following empirical data and the intended usages.
3. Conduct a test utilizing appropriate testing tools focused on desirable quality attributes specified by the quality control team and the client.
4. Plan and conduct process to design test cases for an organization using both clear- and black-box techniques to measure quality metrics in terms of coverage and performance.

Software Sustainment

1. Describe the criteria for transition into a sustainment status and assist in identifying applicable systems and software operational standards.
2. Relate to the needs of operational support personnel for documentation and training, and help develop software transition documentation and operational support training materials.
3. Help in determining the impacts of software changes on the operational environment.
4. Describe the elements of software support activities, such as configuration management, operational software assurance, help desk activities, operational data analysis, and software retirement.
5. Perform software support activities; and interact effectively with other software support personnel.
6. Assist in implementing software maintenance processes and plans, and make changes to software to implement maintenance needs and requests.

Software Process and Life Cycle

1. Engage with a team to translate a software development process into individual areas of responsibility.
2. Commit to and perform tasks related to assigned or agreed-upon areas of responsibility. [MEE: Would it make sense to designate "on time, or "with a reasonable explanation for and plan for addressing delays"]
3. Propose and justify software lifecycle process improvements based on team capacity, project progress data, and quality analysis as part of a software development team's retrospective activities.

Software Systems Engineering

1. Provide a description of system engineering concepts and activities to identify problems or opportunities, explore alternatives, create models, and test them.
2. Develop the big picture of a system in its context and environment to simplify and improve system architectures for supporting system designers.
3. Develop interfaces, which interact with other subsystems. Use information hiding to isolate the contents and collaborations within subsystems, so that clients of the subsystem need not be aware of the internal design of subsystems.
4. Work effectively with engineers and developers from other disciplines to ensure effective interaction.

Software Quality

1. Distinguish quality attributes that are discernable at run-time (performance, security, availability, functionality, usability), from those not discernable at run-time (modifiability, portability, reusability, integrability, and testability) and those related to the intrinsic qualities of architecture and detailed design (conceptual integrity, correctness, and completeness). [Based on SWEBOK 2014]
2. Design, coordinate, and execute, within a project team, software quality assurance plans for small software subsystems and modules, considering how quality attributes are discernable. Correspondingly, measure, document, and communicate appropriately the results.
3. Perform peer code reviews for evaluating quality attributes that are not discernable at run-time.
4. Explain the statistical nature of quality evaluation when performed on software execution; develop, deploy, and implement approaches to collect statistical usage and testing outcome data; compute and analyze statistics on outcome data.
5. Interact with external entities including clients, users, and auditing agencies in conveying quality goals for processes and products.

Software Security

1. Apply the project's selected security lifecycle model (e.g. Microsoft SDL), as a contributing member of a project team.
2. Identify security requirements by applying the selected security requirements method, as a contributing member of a software project team.
3. Incorporate security requirements into architecture, high-level, and detailed design, as a contributing member of a software project team.
4. Develop software using secure coding standards.
5. Execute test cases that are specific to security.
6. Adhere to the project's software development process, as a contributing member of a software project team.
7. Develop software that supports the project's quality goals and adheres to quality requirements.

Software Safety

1. Describe the principal activities with the development of software systems, which involve safety concerns (activities related to requirements, design, construction, and quality);
2. Create and verify preliminary hazard lists; perform hazard and risk analyses, identify safety requirements;
3. Implement and verify design solutions, using safe design and coding practices, to assure that the hazards are mitigated and the safety requirements are met;
4. Be aware of the consequences of the development of unsafe software, that is, the negative effect on those who use or receive services from the software.

Software Configuration Management

[None]

Software Measurement

1. Develop and implement plans for the measurement of software processes and work products using appropriate methods, tools, and abilities.

Human-Computer Interaction

[None]

Project Management

1. Explain the principal elements of management for a small project team.

2. Assist in the managerial aspects of a small project team, including software estimation, project planning, tracking, staffing, resource allocation, and risk management.
3. Develop and implement plans for the measurement of software processes and work products using appropriate methods and tools.
4. Work effectively with other team members in project management activities.

Behavioral Attributes

1. Engage with team members to collaborate in solving a problem, effectively applying oral and/or written communication skills. Work done towards team effort is accomplished on time; it complies with the role played in the team: it uses established quality procedures; and it advances the team effort.
2. Assist in the analysis and presentation of a complex problem, taking into account the needs of stakeholders from diverse cultures, needs, and/or geographic locations. Help in developing a solution for the problem and presenting it to stakeholders, explaining the economic, social, and/or environmental impact of the proposed solution. Identify areas of uncertainty or ambiguity, and explain how these have been managed.
3. Analyze software employment contracts from various social and legal perspectives, ensuring that the final product conforms to professional and ethical expectations, and follows standard licensing practices.
4. Locate and make sense of learning resources, and use these to expand knowledge, skills, and dispositions. Reflect upon one's learning and how it provides a foundation for future growth.

Number of Draft Competencies = 56

Software Engineering Subgroup Members who are Task Force Members

Nancy Mead (Leader)
Hala Alrumaih
Marisa Exter
Rich LeBlanc
John Impagliazzo
Barbara Viola

Software Engineering Subgroup Members who are not Task Force Members (Contributors)

Kai H. Chang, Auburn University
Dick Fairley, Software and Systems Engineering Associates
Kevin Gary, Arizona State University
Thomas Hilburn, Embry-Riddle Aeronautical University
Gabriel Tamura, Universidad Icesi, Colombia
Chris Taylor, Milwaukee School of Engineering
Jim Vallino, Rochester Institute of Technology
Norha M. Villegas, Universidad Icesi, Colombia

C.2.6: Master's in Information Systems Draft Competencies

Business Continuity and Information Assurance [BCIA]

A. Analyze policies and standards for business continuity and information assurance and present the findings to a group of peers.
B. Plan procedures, operations, and technologies for managing security and safety in a disaster recovery situation.
C. Monitor the protection and growth of hardware and software within an information system for a small company.

Data, Information, and Content management [DATA]

D. Identify and report data and information management technology alternatives for a small organization and suggest to management the most appropriate options based on the organizational information needs.
E. Identify organizational policies and processes related to data and information management within a team environment and how to address information and content management solutions for policy infringement.

Enterprise Architecture [EARC]

F. Evaluate an enterprise architecture (EA) using formal approaches by identifying the EA change needs and by addressing domain requirements and technology development through a written report.
G. Describe to a group of managers an enterprise architecture (EA) highlighting software development and maintenance by gathering input from the enterprise to evaluate the level of maintenance involved.

Ethics, Impacts, and Sustainability [ETIS]

H. Apply sustainable system approaches by incorporating multiple IT practices for a corporate environment in a manner that ensures personnel privacy and integrity.
I. Develop a policy concerning contracts usable within an enterprise or government that ensures safety and health standards in compliance with regulatory statutes and requirements for mutual benefit irrespective of cultural and personal characteristics.

Innovation, Organizational Change, and Entrepreneurship [IOCE]

J. Report to the management of an organization's new IS methods and trends and suggest innovative activity models that rely on new uses of existing technologies.
K. Explain ways of exploiting emerging technologies at different levels (individual, team, process, and organization) and address the enabling or enhancing effects of information technology applications.
L. Report to peers the benefits of a new information system design and highlight the potential adverse consequences of the system.

IS Management and Operations [ISMO]

M. Identify the professional management skills needed to design and manage an effective IS organization that ensures operational efficiency n service delivery.
N. Analyze and report IS project management principles that support their use in the organization.
O. Evaluate the use of information systems and resources and present the finding to the management of an organization.

IS Strategy and Governance [ISSG]

P. Identify the effect of IS on industries, firms, and institutions and suggest to organizational managers plans for maximizing firm benefits associated with IS design, delivery, and use.
Q. Report to peers some oversight mechanisms by which an organization evaluates, directs, and monitors organizational IT by leveraging one or more governance frameworks and organizational decision-making practices.
R. Recommend to organizational managers some practices for minimizing environmental effects and suggest ways for long-term organizational viability.

IT Infrastructure [INFR]

S. Evaluate an integrated communication network for a medium-size organization that includes local-area and wide-area network technologies and specify requirements for a large-scale network expansion.
T. Analyze and provide a written report of an implementation architecture for organizational data processing system that uses both internal hardware resources.
U. Enhance the financial aspects of a contract that involves providers of several IT infrastructure services.

Systems Development and Deployment [SDAD]

V. Describe to an audience the requirements for an IT artifact that enhances the way existing domain activities are structured and performed.
W. Report on an IT artifact that meets specified requirements considering non-functional requirements and organizational constraints.
X. Deploy an IT application that satisfies user needs in the context of processes that integrate analysis, design, implementation, and operations.

Number of Draft Competencies = 24

C.3: References (Duplicated) for Appendix C

[Acm03] ACM (2010) *IS 2010 Curriculum Guidelines for Undergraduate Degree Programs in Information Systems,* Association for Computing Machinery (ACM) and Association for Information Systems (AIS).

[Acm04] ACM (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science,* Association for Computing Machinery and IEEE Computer Society.

[Acm05] ACM (2014). *Software Engineering Curricula 2014 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering,* Association for Computing Machinery, and IEEE Computer Society.

[Acm06] ACM (2016). Computer Engineering Curricula 2016 Curriculum Guidelines for Undergraduate Degree Programs in Computer Engineering, Association for Computing Machinery (ACM) and IEEE Computer Society, ISBN: 978-1-4503-4875-1, DOI: 10.1145/3025098, Weblink: https://dx.doi.org/10.1145/3025098.

[Acm07] ACM (2017). *Information Technology Curricula 2017 Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology,* Association for Computing Machinery (ACM) IEEE Computer Society (IEEE-CS), ISBN: 978-1-4503-6416-4, DOI: 10.1145/3173161, Weblink: https://dl.acm.org/citation.cfm?id=3173161.

[Acm11] ACM (2017) Topi, H., Karsten, H., Brown, S. A., Carvalho, J. A., Donnellan, B., Shen, J., et al. (2017). MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems, 40(1). Retrieved from http://aisel.aisnet.org/cais/vol40/iss1/186

Appendix D: Competency-Based Computing Curricula

Computing curricula are the educational matter that define the course of study in baccalaureate programs. CC2020 sees a strategic imperative to shape the future of computing education by reshaping the language used for defining curricular goals. Within the broader context of industry, professions, and society as a whole a curriculum description centered on competency focuses on the individual's capability to apply their computing education in the practical service to society.

A whole curriculum centered on competency informs pedagogy, academic and professional assessment, ethical conduct, relevance to industry, and society. Effective computing education must envelop the individual's *knowing what*, *knowing how*, and *knowing why* to engage their computing education. To better effect these ends, CC2020 proposes a philosophical shift in the format and emphasis of computing curricula through the adoption of a competency model for curriculum specification.

Adopting a coherent competency model to define computing curricula will more clearly promote describing the practical benefits of computing programs to students, benefactors, faculty, administrators, employers, accreditors, lawmakers, and society as a whole. Describing a computing competence in a practical context shifts the focus of curricula away from describing a body of knowledge in relation to a disciplinary area toward pragmatic student accomplishment. Descriptions of what graduates can do in practical situations replace descriptions of content and classroom time. Competency more effectively describes outcome expectations: challenge educators to develop more proficient computing professionals and lead society to recognize the purpose and benefits of computing education.

This appendix presents CC2020's definition of competency and a template for specifying the subject matter of baccalaureate computing education. The competency template is composed of a prose statement of *task* and a structured list of components: *knowledge*, *skills*, and *dispositions*. The model structure is elaborated by examining each of the components in relation to the others and as a whole in contrast to the time-honored definitions of the *knowledge area*, *knowledge unit*, *learning outcome* model. We will address how competency can be leveraged by different stakeholder groups and at different levels in undergraduate/first-cycle computing programs. The chapter concludes acknowledging the most recent efforts to incorporate competency in computing curriculum guidelines that informed and inspired CC2020's adoption of competency [1]–[4].

D.1: Competency in Computing Baccalaureate Education

The genesis of CC2020's commitment to a competency-based orientation for baccalaureate education is rooted in the specific subject matter of computing. However, computing knowledge alone has never been the limit of the preparation appropriate for computing graduates. It is only one part, a significant and crucial part, but not the whole of the competency relevant to educated, productive, computing professionals. Whether the terminus of formal education, the conduit to higher academic degrees, or as is the case most often, a gateway into the workforce of computing professionals, a baccalaureate education must address the wider world of competency that interconnects with the practice of computing in society. The fundamental tradition of published computing curriculum guidelines has focused almost exclusively on the subject matter of computing [2]. This is the case even though most if not all baccalaureate computing programs profess to develop practicing professionals who will apply their computing capabilities in a wide variety of workplaces [6]–[8]. To that end, the scope of competency encompassed by computing curricula cannot ignore capabilities that extend far beyond an emphasis on technical computing knowledge.

When leveraged like learning objectives, well-modeled competencies provide a richer language for expressing the goals of a learning experience. Competency modeling provides the ability to more clearly articulate the connection of learning experiences in language that is better connected to both the expectations of graduates and the broader goals of their education [1], [2]. Other advantages of competency modeling include that it connects knowledge and skills as they are expected to be observed in practical tasks. Another is the opportunity to enfold non-technical knowledge and/or skills as the goal or outcome of an educational experience, and that like learning outcomes, competencies should be observed at some level of skill preferably with a relevant performance of task.

This appendix presents a brief introduction to the theory and an outline of the structure of CC 2020's approach to competency modeling. We believe this can successfully serve as a foundation for adoption across the computing disciplines and their foundational educational enterprise. While CC2020's mission ends at conceptually and structurally defining competency in service to computing, it is our conclusion that this model of competency should underpin other areas of professional capability that are inexorably integral to educating computing professionals. These other aspects of competency play significant roles and are highly recommended for future curriculum designers' consideration. The following sections present the CC2020 in more detail competency model and illustrates its application using a high-level synthesis of the knowledge areas appropriate to a baccalaureate computing program.

D.2: The CC2020 Definition of Competency

CC2020's definition of competency has evolved from numerous models for competency developed and applied in different educational frameworks. A useful overview of competency occurs in the Harvard University Competency Dictionary [Har2] which offers the following explanation:

> "Competencies, in the most general terms, are "things" that an individual must demonstrate to be effective in a job, role, function, task, or duty. These "things" include job-relevant behavior (what a person says or does that results in good or poor performance), motivation (how a person feels about a job, organization, or geographic location), and technical knowledge/skills (what a person knows/demonstrates regarding facts, technologies, a profession, procedures, a job, an organization, etc.)."

CC2020 articulates a notion of competency as a practical means for expressing educational goals [1]–[4] that refines the Knowledge-Skill-Disposition (K-S-D) framework popularized in the IT 2017 Curriculum Report [5]. While the knowledge dimensions of computing have been extensively explored in the various computing curricula, what is meant by skill and disposition have had significantly less focus. Extending previous work, we specify competency as composed of K-S-D dimensions observed within the performance of a task, T.

$$\text{Competency} = [\text{Knowledge} + \text{Skills} + \text{Dispositions}] \text{ in Task}$$

A competency specification enumerates knowledge, skills, and dispositions that are observable in the accomplishment of a task, a task that prescribes purpose within a work context [1]. Figure D.1 provides a diagrammatic illustration of the conceptual structure of competency.

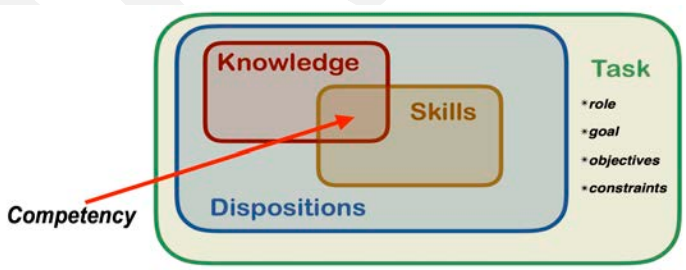


Figure D.1. Conceptual Structure of Competency

These four components that structure the competency specification have the following meanings.

Knowledge is the "know-what" dimension of competency that can be understood as factual. An element of knowledge designates a core concept essential to a competency. This dimension reflects the enumerated subject matter that teachers catalog as topics in their syllabi, departments distribute and balance among the courses they develop in an academic program, accreditation organizations stipulate in their accreditation criteria, and employers identify in job descriptions of their workers. Traditionally, curriculum guidelines for computing education have been predominated by the designation of knowledge elements composed of facts based upon scientific derivation or proof.

Skills refer to the capability and strategy for applying “know-what” to perform a task in context. Competency is realized when “know-what” knowledge is applied in action to accomplish a task, hence in application, skills express “know-how.” Skills are most often developed over time and with practice. Consequently, skill development often requires engagement in a progressive hierarchy of higher-order cognitive process. CC2020’s definition of competency has adopted Bloom’s levels of cognitive process [14] to specify the degree of skill expected in successful task accomplishment.

The skills dimension of competency is often assessed indirectly, through observation of the process or quality of work produced. The activation of “know-what” animated by “know-how” fuses the knowledge and skills dimensions. For that reason, the usefulness of any element of knowledge in a competency specification is only understandable when applied at a level of skillfulness, e.g., specified or observed as a level of Bloom’s cognitive process. Therefore, an element of knowledge and the level of skill with which it is applied are necessarily and naturally conjoined as paired in the specification of a competency. In this way the CC2020 competency model realizes a performance-based epistemology that animates an element of knowledge in achieving a task.

Dispositions frame the “know-why” dimension of competency and prescribe a requisite character or quality in task performance. Dispositions shape the discernment of skillful engagement of “know-what” and “know-how.” Specific to the task at hand, dispositions exert a moderating or controlling influence on a practitioner’s choices by proposing or projecting a desirable quality onto the outcome. How dispositions moderate knowledge and skill could be thought of as the “extent that it accounts for the relation between the predictor and the criterion” [21] in that dispositions connect the ‘better’ or ‘correct’ application of knowledge and skill to the context in which they are applied. For example, dispositions moderate a practitioner’s capabilities to discern a task as “*professionally accomplished”* rather than simply “*completed*.” In this sense, dispositions are able to reflect the professional values behind a competency.

Dispositions characterize socio-emotional tendencies, predilections and attitudes that characterize the inclination to carry out tasks and the sensitivity to know when and how to engage in those tasks [11]. Hence, dispositions denote the values and motivation that guide applying knowledge while designating the quality of knowing commensurate with a standard of professional performance. “Know-why” exhibits as enacted values and because of the difficulty of assessing values and intention, disposition is typically assessed indirectly, through the observance of patterns of behavior or reflective practice.

Task is the construct that frames the skilled application of knowledge and makes dispositions concrete. Task expressed as a colloquial prose statement provides the setting to manifest dispositions, where individuals moderate their choices, actions and effort necessary to pursue and succeed in an efficient and effective manner. In this sense, task enfolds the purposeful context of competency, exposing the integral nature of knowledge, skills, and dispositions. To this end, task definition stipulates pragmatic engagement that reflects professional practice relevant to the particular vision for the program graduates. In this sense, task descriptions provide an explicit context for the program to develop pedagogy and graduates to demonstrate competency as a computing professional.

D.3: The Anatomy of Competency Specification

An effective specification of competency is a synthesis of 1) a colloquial, prose competency statement that sets out a task and 2) the component structure of constituent K, S, and D elements necessary to succeed in that task. The graphical syntax of competency specification is illustrated in Figures D.2, D.3, D.4, and D.5.

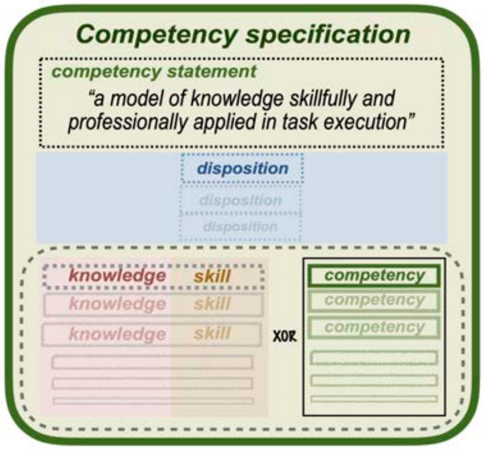


Figure D.2. Conceptual Structure of a Competency Specification

CC2020's definition of competency integrates knowledge, skills, dispositions with task establishes a framework designed to comprehensively describe criteria to support both understanding curricular subject matter in pedagogy and the requisites of task performance in the workplace from which the subject matter derives. A competency statement is a natural language expression of the competency that is more approachable and understandable to the general audience of curricula, while the more explicitly expressed component structure facilitates audit and analysis. Figure D.2 illustrates the relationship between a natural language (free-form) competency statement and the component structured representation of knowledge, skills, and dispositions.

In their most simple form, a singular (atomic) competency specification might address the goals for a solitary job function or curricular element constructed from a suitable competency statement and K, S, and D components [1] as per Figure D.3. That atomic competency might then be assimilated as a constituent of a more complex (composite) competency as per Figure D.4. Composite competency specifications unfold as tree structures with branch and leaf nodes. Figure D.5 illustrates this situation where a composite competency specification (C) may combine both atomic (A) and other composite (C) competencies. Competency specifications are normally considered in the aggregate as they are often used to formulate various configurations of educated ability: job descriptions, plans of study, academic degrees, training certifications, professional accreditations and licensure, and standards of performance evaluation. In this sense, more complex competencies are modeled as composites of other constituent, supporting competencies [2].

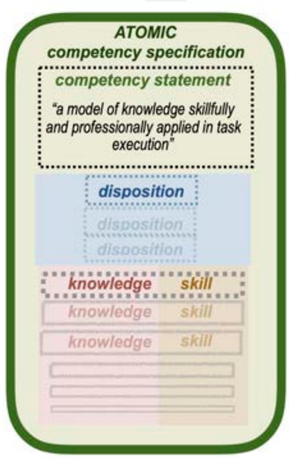


Figure D.3. Atomic Competency Specification: (A)

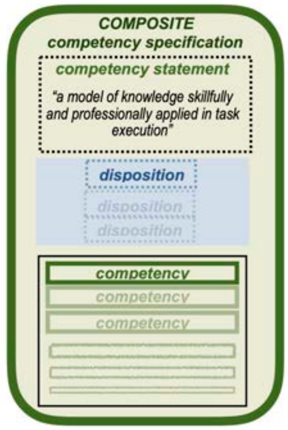


Figure D.4. Composite Competency Node: (C)

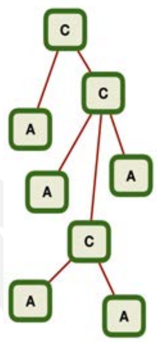


Figure D.5. Competency Tree of Atomic (A) and Composite (C) nodes

The sections that follow we survey the anatomy of a competency specification: the competency statement, the knowledge component (*knowing what*), the skills component (*knowing how*), and the dispositions component *(knowing why*).

D.3.1: The Competency Statement's Role in a Competency Specification

As a whole, a competency specification expresses a model of knowledge that is skillfully and professionally applied in some task execution. The competency statement of a competency specification is a colloquial expression that succinctly conveys the pertinent ability goals to be attained through a course of study or the capabilities relevant to successfully performing a task in the workplace. The competency statement's free-form represents the competency in terms that are familiar and comprehensible to a wide audience, typically using a vocabulary familiar to, and that resonates with, the audience. The competency statement is then structurally augmented and amplified in the enumeration of knowledge, skills, and dispositions that complete the competency specification.

While the natural language of the competency statement favors a public audience, the competency component structure is more formal as it enumerates the components, e.g., knowledge elements demonstrated at a skill level and moderating dispositions determined necessary to demonstrate the competency in task. This structural enumeration of components is essential for automating comparative analyses and visualization of curricula. Having both the free-form of the competency statement alongside the more formal component-specific enumeration corroborates that the two perspectives align. Any divergence perceived in these perspectives would suggest the need for a closer reflection upon the usefulness of one or both representations.

D.3.2: Knowledge, "Knowing What," as a Component of Competency

A single competency expresses knowledge skillfully and professionally applied in some task execution; its vocabulary will often draw from other implied or stated competencies, at some skill level, reflected through some task that contextualizes what is intended. The key aspect is expression: how to express the components in meaningful ways, what knowledge is applied, the expectation of how it is skillfully applied, and what dispositions should be demonstrated along with a successful task.

The richest and most expressive aspect is the knowledge that can and should be skillfully applied. The following subsections outline three perspectives on knowledge suitable for modeling professional competencies in computing. They each play a role in the expectations of computing graduates and practicing professionals. Computing graduates are normatively expected to skillfully apply computing disciplinary knowledge (relevant to their academic program), foundational knowledge consistent with baccalaureate education, and lastly professional knowledge relevant to how graduates operate as professionals. The identification of some knowledge areas as 'disciplinary', some as 'foundational' and others as 'professional' may be arbitrary but, in the end, what is needed is an understandable vocabulary useful to the audience for clear and consistent competency statements.

D.3.2.1: Computing Disciplinary Knowledge

The encyclopedia of computing knowledge that has accumulated with the efforts of the *knowledge area, knowledge unit, learning outcome* model over the last half century provides a rich foundation upon which to develop computing competency catalogs for the various subdisciplines of computing education. These de facto concepts of disciplinary competency are in common use but require formulation to facilitate interoperability and reusability among the stakeholders both academic and industrial. Employers frequently identify specific technologies or general knowledge areas (e.g. networking, cloud computing, systems analysis, and database). As a foundation across the Computing Curricula series, disciplinary knowledge is sometimes differently labeled or described among computing sub-disciplines (e.g. computer science, information systems). CC2020's efforts to promote competency as an overarching framework to describe computing's role in the classroom and the workplace presents an opportunity to normalize the vocabulary for describing computing competency. A normalized vocabulary based upon existing *knowledge area, knowledge unit, learning outcome* curriculum specifications can clarify the terms by which educators identify disciplinary knowledge and its skillful application.

Table D.1 presents a representative summary of computing knowledge areas, extracted from the computing disciplinary documents published since CC2005. While the table is incomplete, what it provides is a sample high-level vocabulary for computing knowledge rooted in the collective wisdom of the different computing communities.

Table D.1 Representative Summary of Computing Knowledge Areas

Catergorization	Computing Knowledge Area
1. Users and Organizations	**K**(**C**-1.1) Social Issues and Professional Practice
	K(**C**-1.2) Security Policy and Management
	K(**C**-1.3) IS Management and Leadership
	K(**C**-1.4) Enterprise Architecture
	K(**C**-1.5) Project Management
	K(**C**-1.6) User Experience Design
2. Systems Modeling	**K**(**C**-2.1) Security Issues and Principles
	K(**C**-2.2) Systems Analysis and Design
	K(**C**-2.3) Requirements Analysis and Specification
	K(**C**-2.4) Data and Information Management
3. Systems Architecture and Infrastructure	**K**(**C**-3.1) Virtual Systems and Services
	K(**C**-3.2) Intelligent Systems (AI)
	K(**C**-3.3) Internet of Things
	K(**C**-3.4) Parallel and Distributed Computing
	K(**C**-3.5) Computer Networks
	K(**C**-3.6) Embedded Systems
	K(**C**-3.7) Integrated Systems Technology
	K(**C**-3.8) Platform Technologies
	K(**C**-3.9) Security Technology and Implementation
4. Software Development	**K**(**C**-4.1) Software Quality, Verification and Validation
	K(**C**-4.2) Software Process
	K(**C**-4.3) Software Modeling and Analysis
	K(**C**-4.4) Software Design
	K(**C**-4.5) Platform-Based Development
5. Software Fundamentals	**K**(**C**-5.1) Graphics and Visualization
	K(**C**-5.2) Operating Systems
	K(**C**-5.3) Data Structures, Algorithms and Complexity
	K(**C**-5.4) Programming Languages
	K(**C**-5.5) Programming Fundamentals
	K(**C**-5.6) Computing Systems Fundamentals
6. Hardware	**K**(**C**-6.1) Architecture and Organization
	K(**C**-6.2) Digital Design
	K(**C**-6.3) Circuits and Electronics
	K(**C**-6.4) Signal Processing

This summary of computing knowledge areas represents a well understood and consistent vocabulary with which we will present computing competency statements and their composition at a very high level of abstraction as illustrations of plausible competency specifications. For reasons that will become clear later in the illustration of competency visualization we order the computing knowledge areas following the semiotic framework developed by Stamper et al. that explicates the expression and transmission of ideas, knowledge, and meaning through human communications [26,27,28]. This ordering offers an ordered arrangement of elements for locating in a cartesian space.

Table D.2 Semiotic Ladder

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

Although Table D.1 summarizes areas of computing knowledge gleaned and synthesized from the six established computing curricula (e.g., [5], [14], [15]), this vocabulary does not address many knowledge areas integral to computing practice. It lacks vocabulary for describing knowledge of a foundational and/or professional nature common to many (if not all) computing disciplines. Neither does Table D.1 address functional areas of business, science, or government where an understanding of the application context is crucial for effective computing. The following sections briefly illustrate these areas that should be given competency attention in baccalaureate education.

D.3.2.2: Professional and Foundational Knowledge

Computing disciplinary knowledge alone does not suffice to prepare graduates for successful occupations. While disciplinary knowledge distinguishes computing professionals among professionals, there are many knowledge areas other than computing that are foundational, that is they are normative in society and the workplace. Foundational knowledge deserves careful delineation in computing programs as it is integral to comprehending and succeeding in the full scope of challenges endemic to the computing practitioner.

There are abilities foundational to workplace conduct that are centered in the individual (e.g., basic academic literacy in: mathematics, physical sciences, language, social sciences, etc. Other typical foundational knowledge includes effective communication in written, spoken, and presentational mediums; self-management of time, decorum, protocols; and many others. Although in-depth study in any of these areas may be appropriate in particular programs, expectations for the application of foundational knowledge requires some stipulation in baccalaureate computing curricula.

Employers are seeking individuals who can apply their knowledge of computing technology in specific, commercial tasks and with a level of prudence evidencing a professional insight. It is well-reported that there is a burgeoning demand for technology-savvy job applicants as computing's role in commerce, government, and society continues to expand. Computing job advertisements are replete with openings for applicants possessing a variety of computing and/or foundational job know-how. However, it is the applicants' facility to effectively apply their computing knowledge to employers' needs that often predominates in assessment. Beyond specific mentions of applied 'professional' knowledge, this is also evidenced by the common requirements for years-of-experience as a proxy term for practical, demonstrated workplace acumen.

Industry managers often agree that professional, not just computing or foundational acumen is a primary criterion for hiring a computing graduate. For example, computing specialists teaming with other professionals from varied backgrounds lays at the heart of effective technical projects. The idea of teamwork is the "cooperative or coordinated effort on the part of a group of persons acting together as a team or in the interests of a common cause" [Dic2]. To be professional a practitioner must demonstrate an effective exchange of ideas through coherent and intelligible communication. Working in teams is often a normative part of a computing curriculum. Ideally, effective teamwork should encompass interdisciplinary opportunities where teams include computing expertise as well as proficiency gained from other areas of study. While not unique to computing professionals, development and mastery among certain of these abilities is essential to helping a beginner transition from beginner to professional.

Table D.3. Sample Professional and Foundational Knowledge Areas

K(P-1)	Oral Communication & Presentation
K(P-2)	Written Communication
K(P-3)	Problem Solving and Trouble-Shooting
K(P-4)	Project and Task Organization and Planning
K(P-5)	Collaboration and Teamwork
K(P-6)	Research and Self-Starter/Learner
K(P-7)	Multi-Task Prioritization and Management
K(P-8)	Relationship Management
K(P-9)	Analytical and Critical Thinking
K(P-10)	Time Management
K(P-11)	Quality Assurance / Control
K(P-12)	Mathematics and Statistics
K(P-13)	Ethical and Intercultural Perspectives

Where foundational and professional knowledge areas are relevant in competency description, a consistent vocabulary for foundational and professional areas will be essential. Table D.3 presents a sample vocabulary for foundational and professional knowledge. These are representative terms, not an exhaustive list. Similar to Table D.1, this vocabulary is drawn from IT2017 [16] and MSIS2016 [17], both internationally approved computing curricula. The table posits likely candidate domains of workplace acumen proven to be value-added to a computing graduate's portfolio.

D.3.2.3: Application Domain Knowledge

Professional practice in computing is manifested in an organizational or commercial context. Every computing artefact resides within some social context; that is, serving the intention of an individual, a community, or both. Knowledge of that social context informs the choices the computing practitioner is faced with to be interpreted as appropriate, beneficial, or not. To make appropriate choices, a computing professional must possess foundational, professional, and application context knowledge and integrate this with knowledge that is otherwise specifically computing. To benefit prospective students, employers, legislators, and the citizen electorate, computing curriculum guidelines should be as explicit as possible about the foundational, professional, and application domain experience promulgated programmatically.

Although computing programs variously focused exclusively on technology for software development (i.e. coding bootcamps and academies) have proliferated over the last decade [23], it should be normative for baccalaureate programs in computing to include requirements for application domain education and experience that informs the professional's potential in a field of practice. Cultural or societal contexts may also require prescribed education and experience: governmental, not-for-profit, non-profit, domestic, or international.

Application domains common to computing include business [4], medicine, engineering, transportation, entertainment, etc. There are many subdisciplines; some are Computing+X and others are X+ Computing where "X's" position indicates whether "X" is the primary disciplinary focus or it is computing's application domain. For example, the computing subdiscipline of information systems itself has numerous derivatives, X-IS programs, (e.g. accounting information systems, marketing-IS, finance-IS, medical-IS, …). Each of these X-IS programs is a discipline in its own right augmented with computing. Any delineated domain of application entails particulars of knowledge, skills, and perhaps, distinctive dispositions instrumental to making informed, astute choices that skillfully apply knowledge in artefact design and engagement.

Each of these example areas of computing knowledge deserve the careful categorization and formulation of practice and learning goals that can be served by authoring relevant competency specifications. In some cases, knowledge among these areas will be integrated in computing competency specifications specific to goals and objectives of a program or sub discipline. Others may be distributed among sibling disciplines in an academic setting or among particular professional development activities in the workplace. There is mutual benefit to both academia and industry when the articulation of the need and value of these knowledge areas associated with computing competency promote discussion and the possibility of inter- and intra- disciplinary normalization.

The synthesis of knowledge areas provided in Tables D.1 and D.3 is at a very high level of abstraction. These are provided here for illustration; in general, practical competency specifications need knowledge stipulated with greater detail.

D.3.3: Skills, "Knowing How," as Components of Competency

Competency descriptions focus on applying knowledge skillfully, *observable knowledge in action*. Writing and structuring competency statements is significantly simplified by recognizing computing skills as normatively cognitive in nature, rather than psychomotor. This simplification correlates with Bloom's theory of the Cognitive Domain in his taxonomy and permits the adoption of a commonly agreed upon vocabulary in the 2001 revisions to Bloom's taxonomy of educational objectives [13]. This reasoning results in *knowing how* expressed as a knowledge component paired with a skill level observable in practice. Table D.4 lists verbs that signify skill level.

Table D.4. Revised Bloom's Cognitive Skill list [13]

	B-I Remembering	B-II Understanding	B-III Applying	B-IV. Analyzing	B-V Evaluating	B-VI. Creating
Definitions	Exhibit memory of previously learned materials by recalling facts, terms, basic concepts, and answers.	Demonstrate understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions,	Solve problems to new situations by applying acquired knowledge, facts, techniques and rules in a different way.	Examine and break information into parts by identifying motives or causes. Make inferences and find evidence to support	Present and defend opinions by making judgments about information, validity of ideas, or quality of	Compile information together in a different way by combining elements in a new pattern or proposing alternative
Verbs	Choose, Define, Find, How, Label, List, Match, Name, Omit, Recall, Relate, Select, Show, Spell, Tell, What, When, Where, Which, Who, Why	Classify, Compare, Contrast, Demonstrate ,Expla in, Extend, Illustrate, Infer, Interpret, Outline, Relate, Rephrase, Show, Summarize, Translate	Apply, Build, Choose, Construct, Develop, Experiment, with, Identify, Interview, Make, use, of, Model, Organize, Plan, Select, Solve, Utilize	Analyze, Assume, Categorize, Classify, Compare, Conclusion, Contrast, Discover, Dissect, Distinguish, Divide, Examine, Function, Inference, Inspect, List, Motive, Relationships, Simplify, Survey, Take part in, Test for, Theme	Agree, Appraise , Assess, Award, Choose, Compare, Conclude, Criteria, Criticize, Decide, Deduct, Defend, Determine, Disprove, Estimate, Evaluate, Explain, Importance, Influence, Interpret, Judge, Justify, Mark, Measure, Opinion, Perceive, Prioritize, Prove, Rate, Recommend, Rule on, Select, Support, Value	Adapt, Build, Change, Choose, Combine, Compile, Compose, Construct, Create, Delete, Design, Develop, Discuss, Elaborate, Estimate, Formulate, Happen, Imagine, Improve, Invent, Make up, Maximize, Minimize, Modify, Original, Originate, Plan, Predict, Propose, Solution, Solve, Suppose, Test, Theory

This understanding of skills expressed as *observable knowledge in action* necessitates an expression of a context for observing the knowledge skillfully applied, portraying a purpose to be fulfilled. In operational terms K+S is normally observed in accomplishing a task, where task conveys a purpose that motivates applying the knowledge with a particular level of skill. The task serves as both the circumstance within which to observe K+S as operationalized by the actor, but also to animate the dispositions, D, that moderate the actor's performance to complete a task and and to what measure of success.

D.3.4: Dispositions, "Knowing Why," as a Component of Competency

The meta-language of competency, "knowing what," "knowing how," and "knowing why," crisscrosses domains of scientific fact, practiced behavior, and cultural norms. Scientific (technically-rational) fact and practiced behavior lend themselves to a categorical assessment: true or false, present or absent, consistent or inconsistent, it works or it doesn't. Dispositions enfold intellectual, social, and moral predilections or tendencies that influence behaviors that do not lend themselves as easily to a categorical assessment. These predilections reflect value judgements that are not amenable to scientific proof. Values may differ or be held differently among individuals or cultures. And, value judgements are also often mutable over time - affected by the experience of practice!

In the broader cultural domains, dispositions may assert positions regarding virtually any desirable quality that motivates human behavior (e.g. ethics, integrity, empathy, accountability, honesty, respectfulness). But in the end, the import of disposition is ultimately realized through individual persons applying their knowledge and skills, through their behavior – individuals leveraging their intellect through responsible decisions and actions [20]. In this applied context, dispositions incline enacted virtues that reflect the values expressed by the actor through their choices, decisions, and actions [24].

An important consideration in the specification of the disposition is the separation of the skilled application of professional or foundational knowledge (such as communication clarity, leadership, creative thinking, and time management, which include significant components from the "know-how" category) from dispositions ("know why"). For example, the development of the disposition categories provided in Table D.5, this was accomplished by analyzing research on job descriptions [18], [19] and other related sources [20] and then removing those statements which were identifiable as a K-S pair, or appear as a competency combining K-S, D, and other components. Hence something as complex as leadership is best modelled as a competency because it has implied K-S pairs and one or more dispositions. Other items may well be a collection of K-S pairs which then would be constituent parts for a competency.

Table D.5 offers a short list of prospective dispositions derived from the literature to round out the knowledge, skills, dispositions as components of competency. Disposition as an intrinsic component of competency represents the opportunity to clearly express institutional and programmatic values expected in a graduate's work. Dispositional expectations enrich the description/assessment of competency and/or the related pedagogy. Ascribing a disposition to a competency indicates a clear commitment to self-reflection and a sober examination of mission, goals, and objectives to reach the clarity that enables its effective integration in curriculum design, the agency of pedagogy, and the character of professionalism.

Disposition is an area that distinctly distinguishes a competency from a learning outcome and is an essential characteristic of a well-structured competency. As such it represents a significant extension in the expressiveness of learning goals and adds language common to professional expectations. However, when used in free form, such terms may easily become vague or difficult to interpret. This is where the specification of a competency, that is the combination of the free-form text with its constituent K+S+D in T framing becomes more valuable. The competency statement is prose that succinctly conveys the essential intention of curricular details, while the structured enumeration of the K-S pairs and D elements conveys intention in action.

Table D.5. Prospective Dispositions [18]-[20]

Disposition	Elaboration
D-1 Proactive	With Initiative / Self-Starter Shows independence. Ability to assess and start activities independently without needing to be told what to do. Willing to take the lead, not waiting for others to start activities or wait for instructions.
D-2 Self-Directed	Self-motivated / Self-Directed Demonstrates determination to sustain efforts to continue tasks. Direction from others is not required to continue a task toward its desired ends.
D-3 Passionate	With Passion / Conviction Strongly committed to and enthusiastic about the realization of the task or goal. Makes the compelling case for the success and benefits of task, project, team or means of achieving goals.
D-4 Purpose-Driven	Purposefully engaged / Purposefulness Goal-directed, intentionally acting and committed to achieve organizational and project goals. Reflects an attitude towards the organizational goals served by decisions, work or work products. e.g., Business acumen.
D-5 Professional	With Professionalism / Work ethic. Reflecting qualities connected with trained and skilled people: Acting honestly, with integrity, commitment, determination and dedication to what is required to achieve a task.
D-6 Responsible	With Judgement / Discretion / Responsible / Rectitude Reflect on conditions and concerns, then acting according to what is appropriate to the situation. Making responsible assessments and taking actions using professional knowledge, experience, understanding and common sense. E.g., Responsibility, Professional astuteness.
D-7 Adaptable	Adaptable / Flexible / Agile Ability or willingness to adjust approach in response to changing conditions or needs.
D-8 Collaborative	Collaborative / Team Player / Influencing Willingness to work with others; engaging appropriate involvement of other persons and organizations helpful to the task. Striving to be respectful and productive in achieving a common goal.
D-9 Responsive	Responsive / Respectful Reacting quickly and positively. Respecting the timing needs for communication and actions needed to achieve the goals of the work.
D-10 Meticulous	Attentive to Detail Achieves thoroughness and accuracy when accomplishing a task through concern for relevant details.
D-11 Inventive	Exploratory / Inventive Looking beyond simple solutions; Examining alternative ideas and solutions; seeks, produces and integrates appropriate alternative

What follows is an illustration of constructing well-structured competency statements and their specification. The purpose is not so much as to define a particular required or desired competency for all computing graduates, but rather to provide a point of discussion about the difficulties and value of such statements and the details contained in their modeled components. Relying on the vocabulary provided in Table D.1 (albeit at a high level of abstraction), and Tables D.3, D.4, and D.5, these examples provide a plausible illustration of how well-structured competency statements can be specified for a program, curriculum, or job description.

D.4: Structuring Competency Statements for Competency Specification

Competency statements have not been the most common means to express learning goals or outcomes. Properly formulated, competency statements should express clear, relevant, and actionable specifications. As such, they differ from learning outcomes in that they imply or imply all four (K, S, and D in T) components. In practice, useful, yet incomplete, competency statements may only imply some of these components, as their primary purpose is communication, not completeness. In the competency explorations carried out by the CC2020 Task Force (presented in Appendix C), the free-form competency statements collected rarely included all four components. Indeed, many expressions of computing competencies were incomplete, and were only explicit about some but not all components. The downside to incomplete competency statements is that they are less useful for assessment, comparison, or other forms of analysis. Hence, the pairing of the free-form statement with its elaborated specification of K, S, and D serves both purposes and in practice acts as means for assessing consistency. Well-structured statements should imply the structured components and in particular communicate a task context where the competency should be observable.

D.4.1: Developing Competency Statements and Specifications

In formulating a good competency statement, the author/designer of a useful statement is best counseled by contemplating the results of a task execution that describe desired actor behavior in clear, relevant, and actionable ways. The K, S, and D vocabularies in Tables D.1-D.5 provide a sample structure for developing and/or parsing competency statements. A particular competency statement can have a number of K and S pair components. Similarly, K-S pairs can be moderated by one or more D labels. This concept follows competency theory that provides for a hierarchical structure for modelling competencies [2]. In this formulation, a competency can be modeled individually as a K, S, and D in T but may also serve as a constituent to other competencies.

In practice, competency statements for curricular use should not be limited to structured language such as that provided in Tables D.1-D.5. Such a restriction limits the expression of the competencies in local contexts. Enumerating the meaning of a free-form statement with structured language by leveraging the K, S, and D in T structure has significant value. The additional work of making implied components more explicit makes the competency statement richer both in expression and meaning: more clear, relevant and actionable.

Modeling a competency statement using structured language improves communication and assessment. For communication, the process of structuring a competency statement into structured language documents the explicit meanings and helps to uncover the implicit meanings intended in the statement. For assessment, the different K+S and D components identified are often assessed in different ways. Identifying and classifying goal components promotes clarity in assessing individual competency components.

As Chapter 5 elaborates, analyzing a competency statement for its various K, S, and D in T modeled competency enables the comparison of competency statements [3]. Typically, unstructured competency statements taken from different computing curricula can be difficult to compare. However, if the constituent parts of the statements can employ a common structured vocabulary, competencies can be compared and modeled through visualization using automation.

In curricula the concept of disposition observable in a task presents the opportunity to enhance the comprehension of knowledge and skills as they related to a computing discipline or academic program. Competency statements offer an opportunity for students to realize more synthesis in their computing education. Applying relevant dispositions informs the students' educational experience by providing an approach explicit in purpose to the content they learn. Consequently, these stakeholders directly benefit from these qualities instilled in computing graduates.

D.4.2: Elaborating Competency Statements

To illustrate the process of elaborating the competency statement, the work is to enumerate the knowledge, skills, dispositions and task elements of the statement. Here we present first an example statement drawn from the System/Software Engineering domain, followed by another example from the Information Systems domain. These both are presented as 'atomic' examples as per Figure D.3. These two examples (Figures D.6-D.9) are then leveraged in an example of a compound competency presented in Figures D.10-D.12.

The goal of these examples is to illustrate how to unpack, in a structured form, phrase decompositions that represent the explicit and implicit K-S-D-T components of the three different free-form competency statements. These are mapped onto structure vocabulary and analyzed for completeness. This detailed mapping of a competency statement serves multiple purposes. To begin with, it very much helps one to understand the completeness of the statement, as well as the K-S pairs expressed or implied. The completeness of the statement suggests the nature of a contextually situated example that would have the opportunity of generating multiple and distinctive assessment opportunities. It also provides a connection to what is expected to be assessed, e.g., not just what the students did, but how they did it; the quality of both their work and the quality of how that work was accomplished.

The most important aspect of this exercise is the support for the actualization of this competency within this program. It provides a structured way of expressing what needs to be taught, a framework for determining how best to manage the learning activities, and clear discussion points for how best to assess different aspects of this competency within the program. For example, learning modules intended to support developing this competency could be inside of a single course, or across multiple courses. It could describe a key task within a requirements course, or a project-based course, or even in a learning exercise at an internship or other setting.

Identify and document system requirements by applying a known requirements elicitation technique in work sessions with stakeholders, using facilitative skills, as a contributing member of a requirements

Figure D.6. Sample Free-form Competency Statement for Systems Requirements

The natural-language text of Figure D.6 can be parsed into three constituent competency phrases for analysis. The list that follows suggests one examination of the explicit and implicit K-S pairs, as well as the implied context of the statement as a whole.

Leveraging the abstract vocabulary of Tables D.1, D.3 and D.4, this results in the following four sets of mappings:

(i) "Identify and document system requirements" (somewhat) explicitly expects students to be applying [S(B-III)] *Requirements Analysis and Specification* [K(C-2.3)] knowledge and understanding. It also implies students to demonstrate applying [S(B-III)] appropriate *Written Communication* [K(P-2)] knowledge and skills.

(ii) "applying a known requirements elicitation technique in work sessions with stakeholders" explicitly expects students to be applying [S(B-III)] *Requirements Analysis and Specification* [K(C-2.3)] knowledge and understanding and implies students to be applying [S(B-III)] *Systems Analysis and Design* [K(C-2.2)] knowledge and understanding.

(iii) "using facilitative skills, as a contributing member of a requirements team" explicitly expects students to be applying [S(B-III)] *Requirements Analysis and Specification* [K(C-2.3)] knowledge and understanding and to be applying [S(B-III)] *Collaboration and Teamwork* [K(P-5)] knowledge and skills.

Extending this to include the dispositional elements (e.g., Table D.5) implicated adds an additional mapping:

(iv) In context, this whole statement implies students to demonstrate capability of evaluating [S(B-V)] *Requirements Analysis and Specification* [K(C-2.3)] and Analyzing [S(B-IV)] *Collaboration and Teamwork* (P-5). These behaviors are expected to be moderated by students demonstrating that they are *Purposefully engaged* (D-4), with *Judgement* (D-6) and demonstrating that they are *Collaborative* (D-8).

Lastly, completeness warrants including the task specification that is stated or implied:

(v) The statement is explicit about having a particular (though unspecified) task (T) in which this work which has.

This example statement in Figure D.6 provides an example of a competency-based approach to describing a possible program or course-level goal or outcome. The statement appears complete, in that it reasonably captures all four K-S-D-T elements of a useful competency statement at a level of abstraction consistent with the vocabulary of interest. Note that with a more detailed vocabulary (not presented), each of the K elements could be expanded into other constituent competencies. Based on this level of analysis, the statement expands into an atomic competency as shown in Figure D.7.

Figure D.7. Example Systems Requirements Competency Specification

This competency statement focuses on a central aspect of systems analysis. If employed in a course or program, it sets up the opportunity for (and challenges) the educator teaching a systems and/or software requirements unit (or course) to set up a learning situation whereby not only will the students be challenged to engage in the context, but also that the instructor can observe student behavior for assessing to what extent the students demonstrate the K-S-D components. The relationships to program definition and assessment are explored in more detail in Chapter 5.

To illustrate this statement/analysis process for a different domain in computing, Figure D.8 presents a second example from the Information Systems (IS) domain related to the area of Enterprise Architecture.

<u>Analyze an enterprise architecture against an organizational business model</u>. <u>Consider several appropriate cloud service approaches</u>. <u>Substantiate the recommendation with cost-benefit details to present to management decision-makers</u>.

Figure D.8. Cloud Services in Enterprise Architecture

Leveraging the abstract vocabulary of Tables D.1, D.3 and D.4, this results in the following four sets of mappings:

(i) "Analyze an enterprise architecture against an organizational business model." It explicitly expects students to be analyzing [S(B-IV)] *Enterprise Architecture* [K(C-1.4)] leveraging that knowledge and understanding. This expectation also leverages understanding [S(B-II)] of *IS Management and Leadership* [K(C-1.3)].

(ii) "Consider several appropriate cloud service approaches" explicitly expects students to analyze [S(B-IV)] *Virtual Systems and Services* [K(C-3.1)] knowledge and understanding.

(iii) "Substantiate the recommendation with cost-benefit details to present to management decision-makers." explicitly expects students to be evaluating [S(B-V)] leveraging *IS Management and Leadership* [K(C-1.3)] knowledge and understanding, This work includes examining and breaking down the details, *i.e.* analyzing [S(B-IV)] *Research and self-starter/learner* [K(P-6)]. This information is then communicated by applying [S(B-III)] *Oral communication and presentation* [K(P-1)] knowledge and skills and applying [S(B-III)] *Written communication* [K(P-2)] knowledge and skills.

Extending this to include the dispositional elements (e.g., Table D.5) implicated adds an additional mapping:

(iv) As per items (i) and (iii), this statement implies students to demonstrate the capability of analyzing [S(B-IV)] *Enterprise Architecture* [K(C-1.4)] and evaluating [S(B-V)] leveraging *IS Management and Leadership* [K(C-1.3)] knowledge and understanding. In the learning context, these behaviors are expected to be moderated by students demonstrating that they are *Proactive* (D-1) in seeking out the information that is needed in a *Self-Directed* (D-2) manner. The purpose of the presentation suitable for management is that they demonstrate *Professional* (D-5) attitudes and behavior.

(v) Lastly, completeness warrants including the task specification (T) which was stated, but also left open to different settings in the application of enterprise architecture.

Similar to the previous example, this presents a reasonably complete statement, given the abstract vocabulary employed. Figure D.9 illustrates this statement, coupled with its mapping.

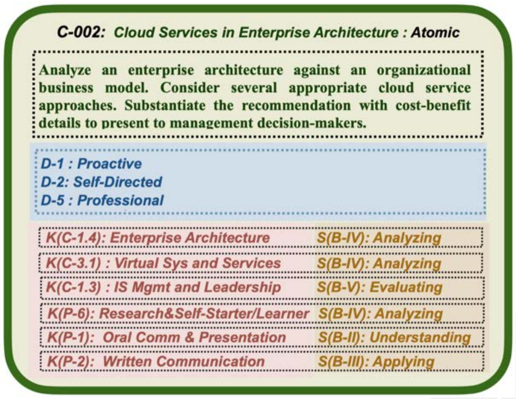


Figure D.9. Cloud Services in Enterprise Architecture

Figures D.6-D.9 present atomic competencies (e.g., those not dependent upon previously-stated/modeled competencies). Figure D.10 presents a statement that can be modeled as being dependent on these statements. This example relates to a competency related to the topic of cloud services and reflects the Information Systems domain.

> Propose an enterprise architecture based on the organizational business model and consistent with the mission and objectives of the organization. The architecture should propose appropriate leading edge technologies consistent with the organizational requirements.

Figure D.10. Cloud Services in Enterprise Architecture

Leveraging the abstract vocabulary of Tables D.1, D.3 and D.4, and the competencies of Figure D.10 results in the following four sets of mappings:

(i) "Propose an enterprise architecture based on the organizational business model" embraces the enterprise architecture competency displayed in Figure D.8 and modeled as C-002 in Figure D.9. The phrases "consistent with the mission and objectives of the organization" are normatively a part of effective business modeling, so are considered within this competency.

(ii) "Propose appropriate… technologies consistent with the organizational requirements" explicitly leverages the systems requirements competency displayed in Figure D.6 and modeled as C-001 in Figure D.7.

Extending this to include the dispositional elements (e.g., Table D.5) implicated adds an additional mapping:

(iv) In the learning context, these behaviors are expected to be moderated by students demonstrating that they are *Meticulous* (D-10) in seeking out the information that is needed in an *Inventive* (D-10) manner.

(v) Lastly, completeness warrants including the task specification (T) which was stated, but also left open to different settings in the application of enterprise architecture.

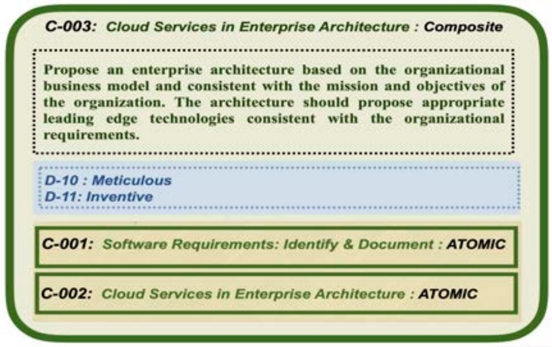


Figure D.11. Cloud Services in Enterprise Architecture

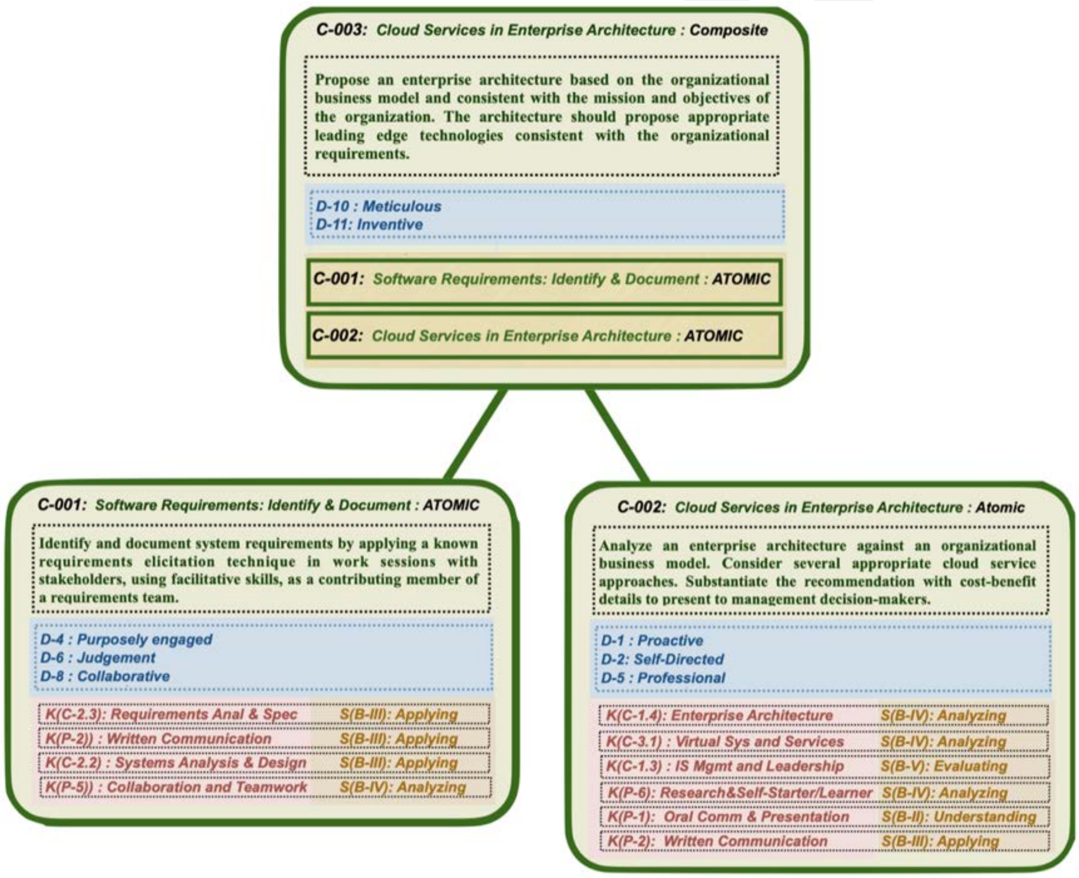


Figure D.12. Composite Competency Specification

D.5: Competency in Computing Education

Competencies as described here, are broad statements that better capture either aspirational or demonstrable goals of a degree, course, learning module, etc. [2] Competencies serve as outcome expectations at the level of a degree program (or other aggregate structure toward which students are working; for the sake of simplicity, we use the term program). Learning experiences are courses, modules, or other similar sets of learning activities that collectively constitute a program. Each learning experience leads to a set of learning outcomes, which collectively enable students to attain the required competencies. A curriculum specifies, at a minimum, the topics, pedagogical approaches, and learning outcomes for each of the learning experiences.

When described in other contexts, competency statements are often binary; either one has that competency or one does not. However, this is not the nature of competencies as described here or as are useful for computing education. Instead, like other types of learning outcomes, they are demonstrated at a particular skill level(s). The purpose of the K, S, and D in T formulation is to provide demonstrable goals that can be observed and in an educational context assessed for how well the student demonstrates the achievement of those goals. Beyond traditional learning outcomes, competency statements encompass the dispositional, 'enacted value' aspects of learning. Consequently, competency specifications at any level of education both inform pedagogy and situates assessment.

One of the most critical aspects of competency development is the use of consistent vocabulary. The K, S, and D in T framework is useful only to the extent that the terminology used for any of the components has a consistent meaning for its constituents. Historically, much of the terminology used for describing competencies is highly context-dependent or ambiguous [2,11]. Consequently, the need for both the authors and readers to have useful, juried vocabulary particularly for the terminology used to describe Knowledge, Skills, Dispositions is essential both to communicate and to comprehend the meaning(s) implied. For example, the vocabulary of Tables D.1 and D.3 are very abstract, whereas many of the knowledge area, knowledge unit hierarchies developed since CC2005 are relatively detailed. With more detailed vocabulary, more detailed competencies can be described.

This competency-based approach is providing a new mechanism for working with/describing curricula reflecting what graduates can do, vs. just what they know. One reason to make a transition from the *knowledge area, knowledge unit, learning outcome* model to competency-based learning is the skills gap that exists between the needs of industry and the capability of graduates from computing programs. In particular, the competency vocabulary leveraged in Tables D.3 and D.4 are all drawn from vocabulary used in computing job descriptions. This connection to the workplace, facilitated by a competency-based approach is important. For a typical university, an overwhelming majority of computing graduates enter the workplace directly. While universities are not training grounds for industry, often there is reported a disconnect between the products produced (computing graduates) by universities and the needs of industry [8,9]. Specifying program expectations as competencies will be more easily understandable by the employer partners of computing programs as well as graduates and other constituents. These themes (and others) are developed in more detail in Appendix E.

The CC2020 project has embraced competency as an underlying theme of its activities and as a principal component of this report. The task force believes that every career path in computing, whether industrial, academic, or government, or any other career demands an intentional level of performance in applicable competencies. It observes that knowledge is only one component of competency. Adopting competency as the foundational model on which to base academic program designs is a more effective bridge between the deliverables achievable by academia and their consumption by the society at large. Thus, it is logical that this report should foster competency-based learning instead of knowledge-based learning. When used intentionally, this approach ensures that graduates of computing programs have a better preparation to be effective in their career paths.

D.6: Competency in Future Curricular Guidelines

The CC2020 task force is committed to the use of competency in current and future computing curricular reports and recommends continued development of competency statements. The competency-based approach makes it possible to compare computing disciplines and facilitate detailed comparisons. Competency implies attaining a level of professional excellence and performance that goes beyond having only knowledge in a field. These extensions include technical and professional skills to function in the workplace at an acceptable level of performance. It is important to employ the competency-based approach in the development of future curricular guidelines within a common frame of reference. This is a major theme of Chapter 5, with the development of a means of collecting and comparing both the results of international curriculum guidelines, but also models that come not from just model curricula, but actual curricula around the world.

This report also assumes that specifying the knowledge, skills and dispositions desirable for computing graduates is beyond its scope. Specifying the detailed competencies is the responsibility of discipline-specific curricular reports and, more particularly, individual computing programs themselves.

Given that most graduates of computing programs enter the workplace, it is very important that all computing programs prepare their graduates properly so they can perform as professionals and engage in productive careers. While the CC2020 project can only suggest its beliefs, its task force is confident that computing organizations and programs worldwide will heed the suggestions made in this report and transform their activities where competency becomes central to their future undertakings. In today's world, graduates must be able to perform in the workplace with the technical and professional knowledge, skills and better opportunities to develop and understand the dispositions that help make their knowledge and skills effective in the workplace.

As the work on CC2020 has progressed, so, also, have other international curriculum development efforts, including IS2020. A cyber security project also constituted an ongoing curricular effort in parallel with the CC2020 project. ACM published this report, called CSEC2017 [16], in December of 2017. Some projects approaching initiation include information systems, with a report pending in 2021. Naturally, other curricular updates include software engineering, computer science, and computer engineering. The task force is hopeful that all future curricular endeavors adopt the competency-based approach.

The necessary inclusion of the task component in a competency suggests opportunities for workplace-bound learning experiences that engage authentic problems with industrial tools; that encourage employers' active involvement supporting professional development through internships, co-op programs, and expert mentorship. Dispositions materialized through task encourage promoting an appreciation for diverse teams, for collaborative norms in project-based activities, and a deliberate and critical reflective practice fostering effective decision-making and continuous learning.

The competency model for computing education presented herein frames the pattern and philosophy for future curriculum guidelines. While at the same time a careful consideration has been applied to the interoperability of competency-based curricular descriptions. The model facilitates the analysis of curricular specifications through comparison to identify the overlap or omissions that may exist between curricula. The potential to represent curricula, curriculum fragments, and job descriptions in competency form facilitates a wide variety of study and analysis. The mission of CC2020 addressed by the development of the competency model for computing education does not enfold the authoring of computing curricula that is a task that must be undertaken by curriculum guideline endeavors followed and integrated with the engagement of educational institutions in the shaping and evolution of their computing programs. And perhaps the greatest incentive supporting the adoption of competency-based curricular specification is the opportunity for a more efficient and effective partnership between academia and industry in addressing the shared goals of advancing the benefits of computing to society.

D.7: Summary

This appendix deals with the nature of competency – one of the salient features of the CC2020 project. It presented several competency statements to exemplify the application of the theory. Competency-based curricula are more expressive in their learning goals, and more easily translated to the language of graduate job descriptions and industry needs. Recognizing the knowledge-based approaches taken in many computing curricula to date, recent developments in computing curricula imply that the components of computing curricula should include not just knowledge and skills but also dispositions, skill levels and typical (maybe "practical") tasks expected of graduates. We recommend that future curricular reports adopt this competency approach to describing computing curricula and expand the theoretical foundation upon which curricula are designed.

D.8: References for Appendix D

[1] L. Waguespack et al., "Adopting Competency Mindful of Professionalism in Baccalaureate Computing Curricula," in EDSIGCON 2019, 2019.
[2] S. Frezza et al., "Modelling competencies for computing education beyond 2020: a research based approach to defining competencies in the computing disciplines," pp. 148–174, 2018.
[3] S. Takada et al., "Toward Understanding Computing Curricula: Visualization Using Competency within the CC2020 Project," Educ. Inf. Technol., 2020.
[4] H. Topi, "Information Systems in CC2020: Comparing Key Structural Elements of Curriculum Recommendations in Computing," in 2017 Proceedings of SIGED: IAIM Conference, 2017.
[5] Task Group on Information Technology Curricula, Information Technology Curricula 2017: Curriculum Guidelines for Baccalaureate Degree Programs in Information Technology. Association for Computing Machinery and IEEE Computer Society, 2017.
[6] D. Bairaktarova and A. Woodcock, "Engineering Student's Ethical Awareness and Behavior: A New Motivational Model," Sci. Eng. Ethics, vol. 23, no. 4, pp. 1129–1157, 2017.
[7] H. Han, "Virtue Ethics, Positive Psychology, and a New Model of Science and Engineering Ethics Education," Sci. Eng. Ethics, vol. 21, no. 2, pp. 441–460, 2014.
[8] A. Radermacher and G. Walia, "Gaps Between Industry Expectations and the Abilities of Graduates," in Proceeding of the 44th ACM Technical Symposium on Computer Science Education, 2013, pp. 525–530.
[9] S. Billett, "Realising the educational worth of integrating work experiences in higher education," Stud. High. Educ., vol. 34, no. 7, pp. 827–843, Nov. 2009.
[10] D. L. Schussler, "Defining Dispositions: Wading Through Murky Waters," Teach. Educ., vol. 41, no. 4, pp. 251–268, Mar. 2006.
[11] D. N. Perkins, E. Jay, and S. Tishman, "Beyond Abilities: A Dispositional Theory of Thinking," Merrill. Palmer. Q., vol. 39, no. 1, pp. 1–21, 1993.
[12] D. R. Lynch, J. S. Russell, J. C. Evans, and K. G. Sutterer, "Beyond the Cognitive: The Affective Domain, Values, and the Achievement of the Vision," J. Prof. Issues Eng. Educ. Pract., vol. 135, no. 1, pp. 47–56, Jan. 2009.
[13] L. W. Anderson et al., "A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives, abridged edition," White Plains, NY Longman, 2001.
[14] ACM/IEEE Joint Task Force on Computing Curricula, Software Engineering 2014 Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering. IEEE Computer Society and the Association for Computing Machinery, 2015.
[15] M. Kramer, P. Hubwieser, and T. Brinda, "A Competency Structure Model of Object-Oriented Programming," in 2016 International Conference on Learning and Teaching in Computing and Engineering (LaTICE), 2016, pp. 1–8.
[16] Task Group on Information Technology Curricula, Information Technology Curricula 2017, 2017th-10th- ed. ACM and IEEE, 2017.
[17] H. Karsten and S. A. Brown, MSIS 2016 Global Competency Model for Graduate Degree Programs in Information Systems. 2016.
[18] J. C. Nwokeji, R. Stachel, and T. Holmes, "Competencies Required for Developing Computer and Information Systems Curriculum," in Proceedings of the 49th Frontiers in Education Conference (FIE'19), 2019.
[19] T. Clear, "THINKING ISSUES: Meeting employers expectations of devops roles: can dispositions be taught?," Inroads, vol. 8, no. 2, pp. 19–21, 2017.
[20] J. M. Gray, "Virtue Ethics : Examining Influences on the Ethical Commitment of Information System Workers in Trusted Positions," Ph.D. Thesis, Nova Southeastern University, 2015.
[21] Baron, R. M., & Kenny, D. A. (1986). The moderator-mediator variable distinction in social psychological research: Conceptual, strategic, and statistical considerations. Journal of Personality and Social Psychology, 51, 1173-1182. Retrieved from https://psych.wisc.edu/henriques/mediator.html (current September 12, 2019).
[22] Simon, H. (1996), The Sciences of the Artificial, 3ed, M.I.T., Cambridge, MA, p. 119.
[23] 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.
[24] Annas, Julia, "Intelligent Virtue," Oxford Press, 2011.
[25a] Wiggins, G., McTighe, J., & Ebrary, I. (2005). *Understanding by design* (Expanded second edition). Alexandria, VA: Association for Supervision and Curriculum Development.
[25b] Wiggins, G., & McTighe, J. (2011). *The Understanding by Design Guide to Creating High-Quality Units*. Alexandria, VA: Association for Supervision and Curriculum Development.
[26] Liu, K., Semiotics in Information Systems Engineering, Cambridge University Press, Cambridge, U.K, 2000.
[27] Stamper, R. K., Information in Business and Administrative Systems, John Wiley and Sons, New York, NY, 1973.
[28] Stamper, R. K., Althous, K. and Backhouse, J., "MEASUR, Method for Eliciting, Analyzing, and Specifying User Requirements," In Olle, T. W., Verrijn-Stuart, A. A. and Bhabuts, L., (eds.), Computerized Assistance During the Information Life Cycle, Elsevier, Amsterdam, The Netherlands, 1988.

Appendix E: From Competencies to Curricula

Chapter 4 introduced and defined the competency concept and briefly discussed how competencies are related to graduate outcome expectations for degree programs. Underlying this model is an assumption that graduate competency specifications provide a foundation for designing learning experiences that are a foundation to the graduates' ability to execute relevant tasks as computing professionals. In this appendix, we will discuss various applications of the competency-based approach. We will outline the stakeholder groups who might benefit from the use of a competency-based model and a variety of ways in which these stakeholders, such as employers, students, and regulatory/accreditation bodies, in the educational ecosystem can benefit from the competency-based approach and utilize it effectively. In addition, we will describe characteristics of the processes with which competencies are identified and authored. Furthermore, we will discuss how a curriculum can be derived from a set of competencies.

The fundamental questions we address in this appendix are: a) What are the most appropriate processes and information sources for deriving competencies for a specific educational program in a specific context? (Section 5.2) and b) How can competencies be used to guide curriculum design and revision processes? (Section 5.3.)

E.1: Competency in Future Curricular Guidelines

In this section, we will define and discuss several key concepts essential for any practical use of the competency-based approach: stakeholders, competency targets, and the differences between traditional and a competency-based approach for specifying computing subdisciplines.

E.1.1: Stakeholders

The CC2020 project has identified five groups of stakeholders the members of which may benefit from the competency-based approach to specifying outcome expectations. These five are as follows:

1. Prospective students and their parents
2. Current students
3. Industry professionals
4. Educators
5. Educational organizations and authorities

Prospective students, supported by their **parents** or guardians, are considering studying computing at a university. They need to understand differences in computing programs when making their choices between universities and their programs of study. A prospective student and their parents might have a basic understanding that the student is interested in computing as a field of study, but it is likely that few prospective students comprehend the variety of computing subdisciplines or the differences between them. Members of this stakeholder group are going to be interested in comparing the various subdisciplines, as well as in understanding the relationship between the characteristics of specific programs and curriculum standards for different subdisciplines, as well as the relationship between the outcomes of a program and the expectations of one or more jobs, or a program or subdiscipline and a career.

Current students are students that are enrolled at an institution of higher education. They might consider a choice of courses from their own institute or another institute (in some cases another department when they intend to take a hybrid curriculum of multiple subdisciplines), or in another country. Alternatively, they may be interested in moving to another educational institution. These students would be particularly interested in comparing programs between different institutions.

Industry refers to organizations that (1) are hiring graduates, (2) are collaborating with universities to choose or specialize a curriculum or need a tailor-made course, or (3) are collaborating in a curriculum by providing internships.

Most importantly, industry professionals and recruiters need to understand what prospective incoming employees have learned, i.e., which competencies they have acquired during their studies. Computing professionals need various specific skills. For example, employers who are looking for software developers might be looking for individuals with strong competencies in software development, and thus they might be interested in software engineering graduates. On the other hand, if an employer wants individuals who have competencies in understanding and guiding the impact of technology on an organization in addition to a foundation in computing, then they might prefer graduates from an information systems curriculum. Thus, understanding how particular types of curricula would fit within their employer needs would help target which type of graduates they prefer in terms of curriculum studied.

Computing educators are faculty members and staff within a single school or university who are responsible for designing and implementing educational experiences, which may include a full curriculum leading to a degree or an individual course or module as part of one or more curricula. These people may be individual university faculty members or teams that design and teach courses, design educational resources (books, massive open online courses (MOOCs), websites, presentation slide decks), manage curricula as taught in their school, or assess student entry or exit levels. Computing educators need to understand how their current or prospective curriculum align with standard curriculum recommendations, as well as understand how well the competencies of graduates match the needs of the industry within their target market.

Educational authorities are organizations that have authority over university education such as (national) ministries of education that govern and finance universities and national or international (e.g., European) bodies that rate, assess, or accredit (university) education, or define qualifications or certificates. Educational authorities need to understand how well a specific program matches the curriculum standards for the field that it purports to teach. In many countries or broader regions, educational authorities are responsible for developing local curriculum standards for various subdisciplines, so they will have to apply the competency model along the lines of the process described later in this appendix.

E.1.2: Competency Targets

Here, we will first define the idea of a "competency target." A competency target reflects an entity that would be defined by providing a set of competencies. These competency targets could be in several categories—curricula, curricular standards, jobs and careers. Effectively, anything that can be specified with a set of competencies is a potential target.

Both pre-college and college students are driven to some degree by the choice of an eventual career. Our competency model can be applied not only to subdisciplines and to college programs, but also to careers and jobs. We note that a career reflects a broad category of specific jobs, just as a subdiscipline reflects a broad category of specific programs. Table E.1 clarifies these concepts.

Table E.1.

	Education	**Workforce**
Singular	(Degree) Program	Job
Aggregate	Subdiscipline	Career

Developing competency-based specifications for these targets can enable comparison between various targets. For example, a competency-based specification of a career can then form a baseline that can be used to compare against various subdisciplines. The relationship between a career and various subdisciplines can drive the choice of the subdiscipline that a student should specialize in. One could pragmatically decide on a subdiscipline based on the distance between a desired career and a subdiscipline, and a similar distance metric could provide guidance regarding the preparation a program provides for either a specific job or a career.

E.1.3: Outcome Expectations and Learning Specifications

In this section, we will briefly review the differences between the competency-based approach and the traditional approach to specifying degree programs. In computing, it has been a long-term tradition to articulate guidance and recommendations for educational programs in the form of curricula that specify the number of classroom hours that are dedicated to specific knowledge units. Knowledge units are typically further categorized into knowledge areas at a higher level of abstraction. In addition to the contact hours dedicated to it, a specification for a knowledge unit has traditionally included a list of topics and a set of learning outcomes. For most computing disciplines, curriculum recommendations do not define how the topics are structured into courses, although many of them include course exemplars to help programs determine how to organize the topic coverage into structured learning experiences (courses). The only exception to this practice in the ACM curricula are the IS curricula up to IS 2010, which have presented the curriculum recommendations as sets of courses.

In a traditional computing curriculum recommendation, a knowledge unit specification might consist of five to 10 topics and a similar number of learning outcomes. Therefore, the level of abstraction of learning outcomes is clearly quite low. For example, in CS 2013, the Information Management Concepts knowledge unit includes eight topics and 13 learning outcomes to be covered in a total of three core classroom hours (say, 12-15 total student work hours). Learning outcomes have to be narrowly specified to be achievable within the time available. Furthermore, in this approach, each learning outcome is associated with a specific knowledge unit. The structure does not include a way to specify higher level learning outcomes that would include components from multiple knowledge units (or knowledge areas).

Competencies as degree outcome specifications are quite different from knowledge unit -level learning outcomes: they reside at a significantly higher level of abstraction and are specified based on the performance requirements associated with an organizational task. As discussed in Chapter 4, competencies integrate multiple knowledge, skill, and disposition dimensions in a specific task context. Competency specifications are an excellent way of articulating the level at which the graduates of a degree program are expected to be able to perform at the time of graduation. As such, competencies alone do not, however, define a curriculum. In order to move from competencies (as outcomes) to a curriculum (a set of learning experiences), one has to determine a) the knowledge, skill, and disposition components of each competency, b) the learning sequencing requirements of these components, and c) a set of effective pedagogies that will allow the students to attain the required competencies. Chapter 4 has presented the conceptual groundwork for this process, and Section 5.3 discusses the process at a more detailed level.

As discussed earlier in Chapter 4, competencies have a hierarchical dependency structure. Competencies developed within various learning experiences will further be integrated with other competencies into larger-scale competencies as part of a longer-term integrative process.

E.2: Identifying and Authoring Competencies

In this section, we discuss the processes with which various stakeholder groups can develop competency statements for their purposes. These processes vary significantly depending on the stakeholder group, such as the following examples:

- A faculty team creating competency requirements as a foundation for a global curriculum recommendation
- A faculty team specifying competency requirements as a foundation for a university curriculum
- A government or industry group articulating competency requirements for professionals working within an industry within a country or a region
- An employer who is writing competency requirements for a specific job role
- An employer who is writing competency requirements for general undergraduate hires
- A prospective employee describing their own competencies for the purpose of presenting them to possible employers.

The process of formulating competencies had emerged in the last few decades as a method for describing educational outcomes, supporting dialogue about what is expected of education from a variety of stakeholder perspectives, and articulating the needs of employers for various job or career profiles. A common use of competencies in higher education policy debates has been as a means with which to facilitate an understanding of the value of education and educational experiences. This section discusses developing descriptions of competencies from the perspective of computing curricula and explores the interplay between the specification of competencies and ways of working with stakeholders in curriculum design and development.

Competency, as described in the previous chapter, is a means of capturing the desirable attributes of graduate performance in situations where one's profession and expectations for professional expertise form the context. From the computing education perspective, these are situations in which our graduates act in interaction with computing environments, systems, and processes.

Descriptions of competence, albeit sometimes often incomplete, are already in common use in the form of learning outcomes and graduate outcomes associated with courses and degrees worldwide. Learning outcomes are an excellent starting point in terms of deriving competencies from the academic stakeholder perspective. One shortcoming, however, is that such outcomes are often structured as ability to apply knowledge or skill to a problem or situation. Aspects of prudential judgement, for instance commitment to ethical standards, personal investment in the quality of the outcomes, and dedication to personal standards of quality, communication and collaborative behavior are often missing or poorly described.

To address this issue, we highlight several perspectives to the processes of deriving competencies in collaboration with stakeholder groups. All these approaches adopt a requirements analysis model familiar to computing practitioners and are guided by the definitions and competency component structures presented in Chapter 4.

E.2.1: Free-form Narratives vs. Semi-formal Specifications

Chapter 4 presents a semi-formal component structure for competencies in order to build a strong foundation for understanding the competency concept and for analyzing and comparing competency-based program specifications in a structured way. As already clear based on Chapter 4, a component-based structural specification is not, however, always the best way to specify curricula. For example, the curriculum guidance documents in computing that have followed the competency-based approach so far (IT 2017, MSIS 2016, and SWECOM) have all presented competencies without any formal specifications or limitations of the structure or vocabulary of the competency statements.

In the development of IT 2017 and MSIS 2016 industry and/or government documents (such as SFIA, e-CF 3.0 [15] and Clinger-Cohen [14]) were consulted as a source of guidance for the form of typical competency statements, in addition to academic research. Competency statements typically start with a command verb and are written to express an expectation set for an organizational role in a specific task context. The general expectation of a competency statement is that it should capture elements of knowledge, skills and dispositions, but when written in a typical free-form narrative, those three components cannot necessarily be identified directly as separate elements without further analytical work.

Experiences developing competencies as a path to curricula in other disciplines has concluded that it is important that the initial process of writing competency statements is not constrained by a fixed component structure or narrowly defined lists of options [9] [10]. When an employer articulates the competencies that their incoming employees are expected to have or gain rapidly after the beginning of employment, it is unrealistic to expect that they would be willing to constrain these statements to a limited vocabulary or a tightly specified grammar. In the same way, if the purpose of competency statements in a curriculum guidance document (either locally or globally) is to convey them to prospective students or prospective employers, limiting the vocabulary or forcing a highly constrained structure is unlikely to improve understanding. The competency statements need to be written in a way that they are understandable and meaningful for the key stakeholder groups that will be using them.

At the same time, as discussed in Chapter 4, there are good reasons to articulate a formal component structure for competencies and limit the set of possible elements for each component type: in practice, it is impossible to analyze, compare and visualize competencies effectively unless the narrative free-form is somehow converted into a semi-structured format. As we will discuss later, it is also possible that the quality of the free-form narratives can be improved by using the structural and vocabulary analysis as a way to improve their coverage of the competency structure established in Chapter 4.

E.2.2: Eliciting competencies

In many ways, specifying competency statements for a specific context is a requirements specification tasks. Instead of articulating requirements for the performance of a software application or a system, competency statements specify performance requirements for individual professionals in a context. Still, the process of eliciting and specifying competencies shares many characteristics with a requirements discovery and structuring processes.

First and foremost, specification of competencies is typically a collaborative process among multiple stakeholder groups. Competencies should be derived from interaction with stakeholder groups, such as employers, students, and regulatory/accreditation bodies, in collaboration with curriculum designers.

To derive expressions of competency a range of strategies can be employed, using approaches that could include all the typical tools of multi-method knowledge discovery processes, such as interviews, surveys, observation, face-to-face and online focus groups and other collaborative processes, etc. Regardless of the stakeholder type, the fundamental question driving the process typically is: what tasks should the graduates/future employees be able to perform in an authentic context at the time when they complete a particular program experience? The discovery process should lead to statements of professional expectations.

There is surprisingly little existing literature regarding the process of authoring competency statements. Chambers and colleagues describe a process for deriving competencies, as well as a constrained language with which to describe them [9] as does Lenburg [11]. Other good examples of how to work with competencies and curriculum design and implementation are presented in the work of Squires and Larson in Space Systems Engineering [12].

While it is impossible for us to provide specific guidance for each of the stakeholder groups and types of competencies that might emerge, it is, however, possible to provide general guidance for writing them. Lenburg [11] offers the following recommendations for writing competency statements:

1. They should be worded as learner-oriented, essential competencies
2. They should be worded in “clear, specific, unadorned, and concise language,” and they should be measurable.
3. They should be action oriented and begin with “the verb that most precisely describes the actual, preferred outcome behavior to be achieved.”
4. They should be consistent with “standards, practice, and real-world expectations for performance,” thus reflecting what “the practitioner actually needs to be able to do.”
5. They should contribute to the “cluster of abilities needed by the graduate to fulfill the expected overall performance outcomes.”

In a free-form competency statement, the focus is typically on the general outcome of the competency in the context; in this way of expressing a competency, the knowledge, skill, and disposition components might not be fully exposed; instead, they need to be inferred from the free-form statement in a way described in section 5.2. Articulating the context is, however, always crucial, since it provides the motivation for the student, making it meaningful to learn and perform that competency. It is, therefore, important to develop the competency's core content in tandem with practicing and developing relevant skills and demonstrating dispositions that positively influence learner agency with respect to sense of self, and responsibility for interactions with others.

E.2.3: Hierarchical Structure of Competencies

In order to define the highest level or most abstract competencies of a program, course, or curricular unit, it is necessary to articulate the knowledge, skill, and disposition components associated with an authentic context. Competency learning in curricula contexts can be represented as a progression, and this view also allows curriculum designers the opportunity to define lower order competencies from which a higher-level competency might be derived. A competency that does not depend on other preceding competencies provides curriculum designers with fundamental learning activities that are self-contained, in the sense that they involve fundamental knowledge, and skills associated with the specific dispositions needed to apply that knowledge and skill meaningfully in appropriate contexts.

In our approach, we assume for competency design purposes that a learner develops competencies in a progression, leveraging competencies she has already attained in the process of developing new ones. Hence competencies do not, in general, stand alone but coexist in a dependency framework, and each competency may be associated with a set of preceding competencies in addition to its knowledge, skill, and disposition learning components (see Figure 5.2). This leads to a directed acyclic graph of competencies, where each competency has a unique set of associated learning components. Note that the precedence graph is not necessarily a tree, and in most cases will not be, for two reasons: there may be no single culminating competency in the progression, and a single competency may be a component of multiple competencies later in the progression.

E.2.4: Deriving Semi-formal Specifications from Free-form Narratives

Section 5.2.1 discussed the differences between semi-formal competency specifications and free-form competency narratives. In this section, we describe the process of deriving the semi-formal specifications based on the free-form narratives developed, for example, in discussions with relevant external stakeholders. The purpose of this activity is to discover the underlying component structure of the free-form competency statements, the development of which we discussed above in 5.2.2. Through this process we not only gain a form of the competencies that can be used for analytics or visualization, but we will also gain a significantly more sophisticated understanding of the nuances of the competencies.

An example of such a process can be derived from the work of Squires and Larson [12]. They draw on earlier work in the Space Systems Engineering community to define a series of competencies in relation to practice in the profession. These competencies, as quoted in their paper, are free form and rather abstract in nature.

For example, “Manage systems engineering”, which implies a combination of the knowledge base for maintaining complex space engineering system solutions, together with skills in space engineering principles and procedures, performed in space engineering contexts where commitment to the quality and failsafe nature of the outcomes of the process were demonstrably of high value and always palpable in the context of the decision making of the individual in systems management and development.

These statements are then analyzed, decomposed and ultimately regrouped in the form we describe in Chapter 4, having used the original data as a means to derive competency statements that include elements of knowledge, skill combined with dispositions related to professionalism in an application context.

In terms of the “manage systems engineering” competency example given above, our recommended process would involve a subsequent decomposition and expansion step as the high-level competency was evolved into knowledge, skill and contextual dispositional components as is shown in the subsequent explanatory text in the same example. Once these high-level free form statements are transformed into the structure developed in Chapter 4, curriculum designers and instructional designers can transform them into statements describing learning activities and experiences where the relevant context can be created in order for the competency to be observed.

E.2.5: Authoring Free-form Narratives from Competency Components

It is also possible to move in the other direction and author free-form narratives by first focusing on the components. Identifying the knowledge, skills, and disposition components of a competency first may be a good starting point in cases when the full identity of the target competency is not clear and needs to be calibrated at the level of the components first. The danger of this approach is that the author may ignore the fact that the whole is typically much more than a pure aggregation of the components. Still, if the authors are able to specify the component structure, it certainly provides valuable guidance for formulating the competency narrative.

E.3: Using Competency Specifications as a Foundation for Curriculum Specifications

One of the essential questions for any academic administrator or faculty member who is developing a degree program or other collection of learning experiences based on a competency-based approach is as follows: How do we determine a set of educational experiences that, if not guarantee, at least significantly increase the probability of the proposed program's graduates being able to achieve the competency expectations that have been set for them. It is useless to specify competencies as program outcomes unless there is a meaningful mechanism for identifying and structuring a set of learning experiences that enable the students to achieve the specified competencies (Chyung et al [13], MSIS 2016). In other words, a set of competency statements have to be transformed into a curriculum form consisting of educational activities that help to scaffold the student's progression in various types of outcome areas.

In this section, we will discuss several examples of existing models that have been proposed for the purpose of converting a set of competency expectations into a curriculum. As you will see, they appear to share some basic characteristics.

Just as a reminder, we use the following terminology: *Competencies* serve as outcome expectations at the level of a degree program (or other aggregate structure toward which students are working; for the sake of simplicity, we use the term program). *Learning experiences* are courses, modules, or other similar *sets* of *learning activities* that collectively constitute a program. Each learning experience leads to a set of learning outcomes, which collectively need to enable the students to attain the required competencies. A *curriculum* specifies, at a minimum, the topics, pedagogical approaches, and learning outcomes for each of the learning experiences.

E.3.1: Existing Models

MSIS 2016 [18] presents a process for deriving a set of learning experiences (referenced as modules) based on a set of competency specifications (see Figure 5.1). This process assumes an underlying competency model similar to that of MSIS 2016, which includes 10 competency areas, 88 competency categories (each associated with one of the competency areas), and several detailed competencies within each category. Furthermore, it recognizes five attainment levels for each of the competency categories (awareness, novice, supporting, independent, and expert), specifying that educational programs are seldom sufficient to help anybody achieve the expert level.

The process of developing the learning experiences in the MSIS 2016 model starts with a program needs analysis (Step 1) combined with the determination of the job roles that the program plans to focus on (Step 2). Based on the outcomes of the first two steps, the program determines competency [category] attainment levels that it assumes its graduates to achieve (Step 3), including inclusion of potential brand-new competency statements. Next, the model suggests that the program should develop or confirm an initial architecture for the learning experiences (Step 4), followed by either verification learning experiences at the learning objective level (Step 5; program based on existing courses) or drafting a set of new learning experiences with learning objectives (Step 6; new programs). Following this, the model suggests that in Step 7, the results of Step 5/Step 6 are mapped with the competency attainment levels specified in Step 3. In Step 8, the differences between Step 3 and Step 7 are identified. Step 9 is for determining required modifications to the learning experiences and/or their learning outcomes to address the differences identified in Step 8. Finally, in Step 10, detailed learning experiences are designed, including the topics and pedagogies.

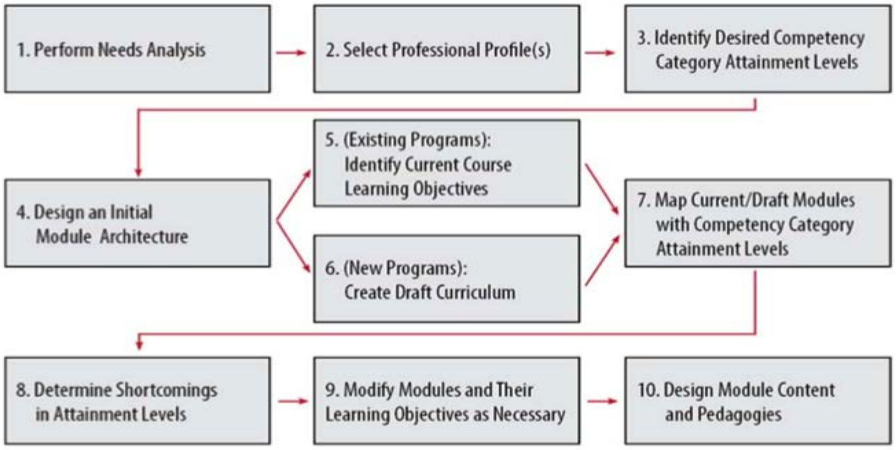


Figure E.1 Process for deriving learning experiences from competency specifications (adapted from MSIS 2016)

Squires (2009) summarizes another approach for developing a curriculum based on a set of competencies using the International Academy of Astronautics (IAA) Space Industry Systems Engineering competency model. This model itself consists of 10 competency areas and 37 capabilities each associated with one of the competency areas. Furthermore, the model specifies four proficiency levels: Participate (Know), Apply (Perform), Manage (Lead), and Guide (Strategize). Squires's process includes the following items:

1. Select the competency model to use
2. Validate the most important ('critical') competencies
3. Determine the current curriculum's ability to enable the students to attain the required proficiency levels
4. Determine the proficiency levels that the future curriculum needs to attain
5. Identify the gap areas between the current and the future curriculum
6. Assess and fix the gaps.

In this model, the sixth state is the part of the process that addresses the curriculum and specifies how the curriculum (both topics and pedagogy) have to be modified (or created in the case of a new program) so that the curriculum will enable students to attain the required proficiency levels.

Finally, Chyung et al [13] propose another six step process, which covers both authoring of the competencies and curriculum design based on them. In this process, the first three steps include— borrowing the authors' terminology— the use of three sources of data for determining the competencies (alumni & industry analyses, professional standards & curriculum benchmarking, and departmental goals and curriculum review). Thus, the fourth step, actual authoring of the competencies, will be based on two types of external resources (employer needs and national and global competency/curriculum guidance) and internal goals and review processes. Chyung et al [13] include the development of the learning experiences as the fifth step, during which "the key to this ongoing process is to carefully align the stated course objectives, the competencies that apply to that course, and the graded course assignments" (p. 311). In this process, it is important to ensure that course goals are aligned with "applicable competencies" and that the course includes learning processes that "both help students acquire those competencies and assess the extent to which they have been successful."

Summarizing key findings from these three models suggests the following:

- In all of the models, the characteristics of the learning experiences that constitute the curriculum are determined based on the outcome expectations specified with competencies.
- All of the models assume that the program competencies are identified (at least partially) based on existing competency models (developed by industry/government groups or by professional societies).
- Two of the three models recognize that identifying the outcome expectations as a set of competencies is not sufficient; in addition, in these models an expected attainment level needs to be specified for each competency.
- None of the models provide specific guidance for the process of deriving learning experiences from competencies. It appears that the models imply a need to first derive a set of learning outcomes associated

with each of the competencies. Then, the learning outcomes need to be organized into learning experiences. The sets of learning outcomes within each learning experience will determine the topics that the students need to be engaged with and the forms that this engagement takes (pedagogy).

- Each of the approaches offers at least hints regarding the need to continuously assess the extent to which the implemented learning experiences enable the students to attain the expected competencies at the expected level.

E.3.2: Building Curricular Guidelines by Based on Competency Specifications

In this section, we will discuss the special characteristics of the processes that entities providing educational recommendations to a large number of programs need to follow when developing those recommendations. As discussed earlier in this report, there are two existing recommendations (IT 2017 and MSIS 2016) that are based on a competency-based approach. These reports were developed separately, and they are not fully mutually consistent. The other curriculum recommendations (CE 2016, CS 2013, CSEC 2017, and SE 2014) are with one exception (IS 2010) based on the *knowledge area, knowledge unit, learning outcome* model.

The most important decisions that future curriculum recommendation authors need to make are as follows: a) will they follow some competency-based approach, focusing first on the outcome expectations specified with competencies and b) if the answer to the first question is affirmative, will they also include traditional *knowledge area, knowledge unit, learning outcome* material and/or course exemplars in addition to the competencies? Neither question is trivial, and particularly in the context of the second one, there are arguments supporting both approaches. On one hand, just focusing on competencies emphasizes the fact that providing globally applicable guidance regarding curriculum elements (learning experiences, knowledge area, knowledge unit, learning outcome elements) is very challenging because of the differences in local educational architectures and basic requirements. It is more effective just to specify the outcome expectations with competencies and provide strong guidance for transforming competencies into curriculum components. On the other hand, many schools and departments using professional society curriculum recommendations do not have the required resources to go through the time-consuming and resource intensive process of deriving learning experience specifications (courses) from the competencies. They expect learning experience (course) specifications. In addition, providing program-level recommendations in the form of outcome expectations will not remove the usefulness of the frequently maintained *knowledge area, knowledge unit, learning outcome* structures.

What are the resources that professional society recommendations should use to determine the recommended competencies? Obviously academic expertise will continue to be important, as has been the case throughout the history of curriculum recommendations. At the same time, engaging in a dialogue with relevant industry partners is even more important in a competency-based approach than it used to be. One of its main benefits is, after all, that it allows programs to align well with the employer expectations and make the alignment clearly visible through a formal mapping process.

Not coincidentally, there are several computing/IT competency models that have been developed in extensive, well-funded government/private sector processes, such as e-CF 3.0 [15] and SFIA (see Appendix B). Using them as a source of inspiration and improved understanding of industry requirements makes sense. Depending on the subdiscipline, there might be specialized industry or government efforts that provide additional focused guidance (such as, for example, the use of the Clinger-Cohen report in the context of MSIS 2016; CIO Council [14].

E.3.3: Building University-level Curricula Based on Competency Specifications

The examples specified in Section 5.3 have a relatively good direct fit at the university level. The key element of university level competency-based outcome expectation identification processes is that the process to identify the competencies includes input from key employer and/or alumni partners, professional/academic society guidance, and program's own identity and resource availability. Once the competencies have been identified, the authoring of the learning experiences requires the identification of learning experience outcomes based on the competencies, configuring the learning outcomes into groups that represent learning experiences, and then designing learning activities within each learning experience based on the learning outcomes. Obviously, this is seldom a process that starts from a clean slate—existing learning experiences form the foundation for the work.

E.3.4: Specifying Program Outcomes as Competencies from Pedagogical Requirements

One of the positive but potentially resource-intensive impacts of specifying program outcomes with competencies is that enabling students to develop skills and dispositions in many cases requires a different set of pedagogical assumptions and approaches compared to a mostly knowledge-based specification of outcomes (and their assessment). In practice, competency-based outcome specification will lead to a broader set of types of learning experiences, often including a much stronger focus on various forms of experiential learning from interactive simulations to intensive projects to field experiences to internships and co-ops. Particularly domain-specific skills and dispositions require a learning environment that is different from a traditional classroom environment.

E.4: Competencies and Stakeholder Value

Competencies, through their task context, are closer to the language with which employers describe their needs than the traditional *knowledge area, knowledge unit, learning outcome* model can achieve. Consequently, competency specifications communicate value for the (prospective) employer organization more directly and transparently than knowledge-based specifications do. Therefore, competencies help other stakeholders such as students, parents and the public sector understand what future careers degree programs are aligned with.

Competencies as a conceptual framework for valuing the outcomes of higher education can be traced back to the 1970's and legal, nursing and teacher vocational training programs in the U.S. These programs emphasized the acquisition of the behavior exhibited by outstanding professionals as a way to identify and develop desired skill sets through education and training [8]. The resulting approach focused on training by mimicking desirable behavior, and ultimately did not produce the intended competencies. Consequently, these experiments did not attract much of a following. Renewed interest from labor organizations and vocational education in the concept emerged throughout the late 1980's, but, it was only at the end of the 1990s that higher education began to renew its participation in the conversation. Klink, Boon, and Schlusmans [5, p. 2] highlight the following contributing factors: 1) a shift in the job market towards increased career and professional mobility; 2) the emergence of the "knowledge worker" and "knowledge economy" in which application of knowledge and skills and "the motivation to keep learning" are considered essential to personal and professional growth; 3) new trends in higher education in response to an increasingly dynamic and complex world that makes acquisition of technical knowledge insufficient; and 4) innovations in learning sciences and education, such as participatory learning, deep learning, and contextualization. As these trends became integrated into the mainstream of higher education, they also evoked a switch in the value proposition of education "from knowing to learning", and ultimately to ability to perform in a relevant and high value manner in professional contexts.

The link between competency and professionalism and high levels of professional performance in relevant domain areas has long been a part of promotion and salary processes in the public and private sector. Competencies emerge immediately in the discourse of job advertisements and employer vernacular. Van der Klink and Boon[4] make the case for the popularity of the concept due, ironically, to the lack of clarity over the term competency and maintain that the number of definitions "is probably incalculable" [5]. A literature study by Stoof, Martens, and Van Merrienboer [6] places the word in the "wicked words" category, meaning that its limits are hard to determine, which makes complete agreement on its meaning elusive. Despite its continuing fuzziness, the term promises to be useful in bridging the gap between educational outcomes and job requirements [7].

CC2020's definition of competency provides the opportunity for mutually consistent specifications of practitioner competency: relating attributes possessed by an applicant to those required by an employer. Of significant mutual benefit both to computing employers and to the academy would be the development of specification standardization between curricular competency and employer job descriptions. Modeled competencies offer the opportunity for academic computing to clearly describe their graduates' capabilities while at the same time help employers to more clearly communicate their functional job requirements. In such a circumstance the computing educators would have the opportunity to weigh their educational goal descriptions against industry needs. As a result, human resource activities in industry would find it easier to identify likely institutional sources of graduates with relevant competency profiles as prospective future employees.

Competency offers a contextualized model through which communication of practitioner capabilities of graduates can be realized. This in turn better serves the coordination and collaboration among institutions of computing education along with the human resource activities of industry. Furthermore, this model may better facilitate advising prospective students who wish to align their studies with clearly described employment opportunities. All the while such a collaboration can influence curricula in educational programs by providing a better understanding of job markets they may wish to serve. In any case, specific competency descriptors offer a facilitating bridge in the dialog between academia and industry locally, nationally, and internationally.

The explicit fusion of knowledge and skills adopted in CC2020 emphasizes the role of practice in the process of demonstrating "knowing" [17]. Enhancing the existing learning outcomes approach, which has been a prominent feature of curricular description, competency's fusion of knowledge and skills advocates for an explicit goal of crystalizing the dimensions of practical professional capability in curriculum description. The intrinsic role of task in both pedagogy and assessment provides a natural opportunity for an explicit articulation of the interdependence of curriculum and employability.

E.5: Assessing Competencies

For competency modelling to be useful for computing education, it is also essential that there be an effective coupling to assessment and curricula expectations. When examining issues related to competency modelling, this is, in fact, one of the issues that has been prevalent in the literature [1]. Outcomes assessment, both at the degree-program and course level are essential to the quality management of effective educational processes. In current practice, the target of most outcomes assessment in computing is predominantly declarative knowledge and skills related to computing (e.g. examine the definition of learning outcomes enshrined in the Bologna Process [1]). This is corroborated by Fuller, who observes that "Professional skills and attitudes form an increasingly large part of the requirements of computer science graduates. Students are assessed on their knowledge and cognitive skills but not on the attitudes that will lead them to practice in the workplace what they have been taught in the classroom"[3].

Hence the assessment of dispositions, and particularly student performance of the integration of disposition, skill and knowledge provides significantly more power to curricula designers working to improve the definition of outcomes at different levels of abstraction.

E.6: Summary

The main focus of this appendix has been on two key questions related to the use of competencies for specification of degree programs in computing: first, what processes should be used to produce competency statements (and how do we determine whether or not the produced statements are an appropriate reflection of the outcome expectations) and second, how do we create curricula (specifications of learning experiences) based on the competency specifications. Furthermore, the chapter also discussed the value of the competency-based approach to various stakeholder groups and pointed out the essential role competency specifications can serve in assessment. Overall, the goal of this appendix has been to provide applied guidance for the use of competencies in program and curriculum guidance development.

E.7: References for Appendix E

[1] I. Baumgartner and V. Shankararaman, 'Actively linking learning outcomes and competencies to course design and delivery: experiences from an undergraduate Information Systems program in Singapore', in *Proceedings of the 2013 IEEE Global*

Engineering Education Conference, Berlin, Germany, 2013, pp. 238–246.

[2] E. Pikkarainen, 'Competence as a Key Concept of Educational Theory: A Semiotic Point of View', *J. Philos. Educ.*, vol. 48, no. 4, pp. 621–636, Nov. 2014, doi: 10.1111/1467-9752.12080.

[3] U. Fuller, A. Pears, J. Amillo, C. Avram, and L. Manilla, 'A Computing Academic Perspective on the Bologna Process', *ACM SIGCSE Bull.*, vol. 38, no. 4, pp. 115–131, Dec. 2006.

[4] M. Van der Klink and J. Boon, 'The investigation of competencies within professional domains', *Hum. Resour. Dev. Int.*, vol. 5, no. 4, pp. 411–424, 2002.

[5] M. Van der Klink, J. Boon, and K. Schlusmans, 'Competences and Vocational Higher Education: Now and in Future.', *Eur. J. Vocat. Train.*, vol. 40, no. 1, pp. 67–82, 2007.

[6] A. Stoof, R. L. Martens, and J. J. Van Merriënboer, 'What is competence? A constructivist approach as a way out of confusion', *Onderwijs Res. Dagen ORD Leiden Neth.*, 2000.

[7] D. Kennedy, A. Hyland, and N. Ryan, 'Learning outcomes and competences', *Introd. Bologna Object. Tools*, pp. 2–3, 2009.

[8] G. Grant, 'On competence: a critical analysis of competence-based reforms in higher education.', 1979.

[9] D. W. Chambers and J. D. Gerrow, 'Manual for Developing and Formatting Competency Statements', *Journal of Dental Education*, vol. 58, no. 5, pp. 361–66, 1994.

[10] F. E. Weinert, 'Definition and selection of competencies: Concepts of Competence', *Organization for Economic Co-operation and Development*, 1999.

[11] Lenburg, C. B. (1999). The framework, concepts and methods of the competency outcomes and performance assessment (COPA) model. *Online Journal of Issues in Nursing*, *4*(2), 1-12.

[12] A. Squires and W. Larson, 'Improving systems engineering curriculum using a competency-based assessment approach', *International Journal of Intelligent Defence Support Systems*, vol. 2, no. 3, p. 184, 2009.

[13] Chyung, S. Y., Stepich, D., & Cox, D. (2006). Building a competency-based curriculum architecture to educate 21st-century business practitioners. *Journal of Education for Business*, *81*(6), 307-314.

[14] CIO Council. (2012). Clinger-Cohen core competencies & learning objectives. *Washington, DC: CIO Council*.

[15] European Qualifications Framework (EQF). European Commission. Learning Opportunities and Qualifications. https://ec.europa.eu/ploteus/en/content/descriptors-page/. Accessed 2019Aug17.

[16] Voorhees, R. A., & Bedard-Voorhees, A. (2016). Principles for Competency-based education. In *Instructional-Design Theories and Models, Volume IV* (pp. 49-80). Routledge.

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

[18] Topi, Heikki, Helena Karsten, Sue A. Brown, Joao Alvaro Carvalho, Brian Donnellan, Jun Shen, Bernard C. Y. Tan, and Mark F. Thouin (2017). "MSIS 2016: Global Competency Model for Graduate Degree Programs in Information Systems." The Joint ACM/AIS MSIS 2016 Task Force. *40*(1). https://doi.org/10.17705/1CAIS.04018. Retrieved from http://aisel.aisnet.org/cais/vol40/iss1/18.

Appendix F: Repository Development

An approach that can be taken is to create an experimental repository for the eventual structure of competency encoding. This section describes the development of an exploratory architecture for the digital repository using data drawn from published curriculum guidelines using screen scraping and vocabulary machine learning tools. The goal is to design a framework that can accommodate a three-dimensional concept of competency (knowledge, skill, and disposition) regardless of how one defines those three dimensions.

F.1: Repository Development

In the experimental repository, a select team used Eduglopedia [Edu1] as the source for knowledge area elements. This open and free global encyclopedia for higher education contains more than three thousand course descriptions and more than nine hundred program descriptions from approximately five hundred institutions. Furthermore, it uses Bloom's Cognitive Process Dimension [And1] as a placeholder for skill.

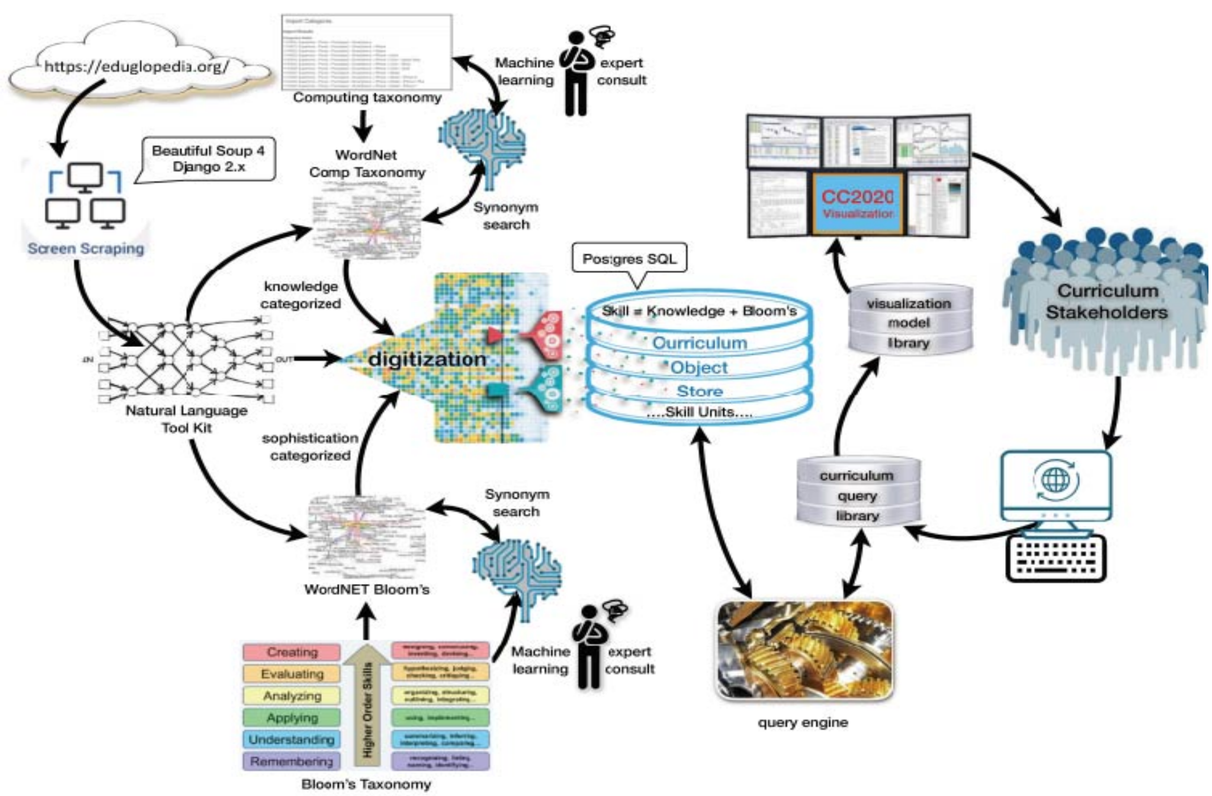


Figure F.1. Repository development process

The repository development uses Beautiful Soup (a Python package) to screen scrape the Eduglopedia that generates XML descriptions of various knowledge areas and relationships within curricula. It also uses tools such as synonym search and machine learning to generate a computing taxonomy and to identify specific verbs that applies to the different levels of Bloom's Cognitive Process Dimension. It also generates a database of skills and knowledge as a source for queries that in the end allows for visualization of a curriculum. Figure F.1 shows the result of the repository development process. The repository data collection and visualization support are done as follows:

1. Use Beautiful Soup to screen scrape Eduglopedia and obtain XML descriptions of various knowledge areas and relationships within curriculum.
2. Use tools such as synonym search and machine learning (as well as review by human experts) to:
   (a) generate a computing taxonomy, and
   (b) identify verb sets that apply to the specific cognitive process levels of Bloom's.
3. Generate/digitize a database of knowledge elements in semiotic order, and skills as applied knowledge in Bloom's cognitive process ordering.

The resulting repository, the curriculum object store, serves as the source for queries that interfaces with a library of representational models allowing users to select, visualize, and/or compare curricular specifications: curriculum guidelines, program catalogs, course descriptions, accreditation standards, and job advertisements.

F.2: References for Appendix F

[And1] Anderson (2001). Bloom's Cognitive Process Dimension.
[Edu1] Eduglopedia. https://eduglopedia.org

Appendix G: Additional Visualizations and Analyses

This appendix shows visualizations that were considered during the CC2020 project. Note that some of the terminology that appears in this appendix does not necessarily comply with the terminology that was given in Chapters 4 to 6.

G.1: Use Case-based Analysis

This section gives four example use cases. Note that the two use cases in G.1.1 and G.1.2 are taken from [Tak1].

G.1.1: Case 1: Question from Prospective Student

A student is interested in entering undergraduate education in computing and wants to know what type of curriculum would best fit her interests. She might have some ideas about dispositions that are relevant in her future curriculum, and/or have a preliminary view on domains that would provide her with future job opportunities. She might start by checking promising dispositions (or, alternatively, she could start by choosing the knowledge categories and areas – we show only the first scenario, but the alternative would lead to the same results). She would see a list of dispositions (Figure G.1(a)), from which she would choose, resulting in the interface showing the chosen dispositions as shown in Figure G.1(b). Note that the dispositions are indicated by color, as there is no order dimension.

	Dispositions
	Proactive
	Self-directed
	Passionate
	Purpose-driven
	Professional
	Responsible
	Adaptable
	Collaborative
	Responsive
	Meticulous

(a) Before choosing

	Dispositions
✔	Proactive
	Self-directed
✔	Passionate
	Purpose-driven
	Professional
✔	Responsible
	Adaptable
	Collaborative
✔	Responsive
	Meticulous

(b) After choosing

Figure G.1. Choosing dispositions by a prospective student

The student may also indicate which knowledge categories and knowledge areas seem interesting for her. Figures G.2 and G.3 show a possible process. She first chose three categories: Users and Organizations, Systems Modeling, and Software Fundamentals. In Figure G.2, the ellipse of these three categories are highlighted with red borders. If needed, the student could indicate which individual knowledge areas are most relevant. Figure G.3(a) shows the knowledge areas for each of the chosen three categories. The student chose the knowledge area User Experience Design for Users and Organizations category, and Systems Analysis and Design and Requirements Analysis and Specification for

Systems Modeling category; again, the ellipse of the chosen knowledge areas are highlighted with red borders. The student did not want to make a detailed choice in the category of Software Fundamentals. The resulting choices are shown in Figure G.3(b).

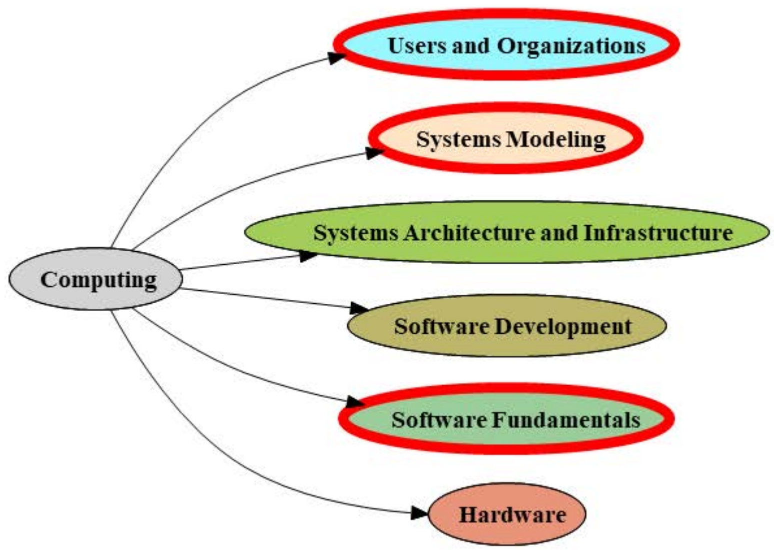
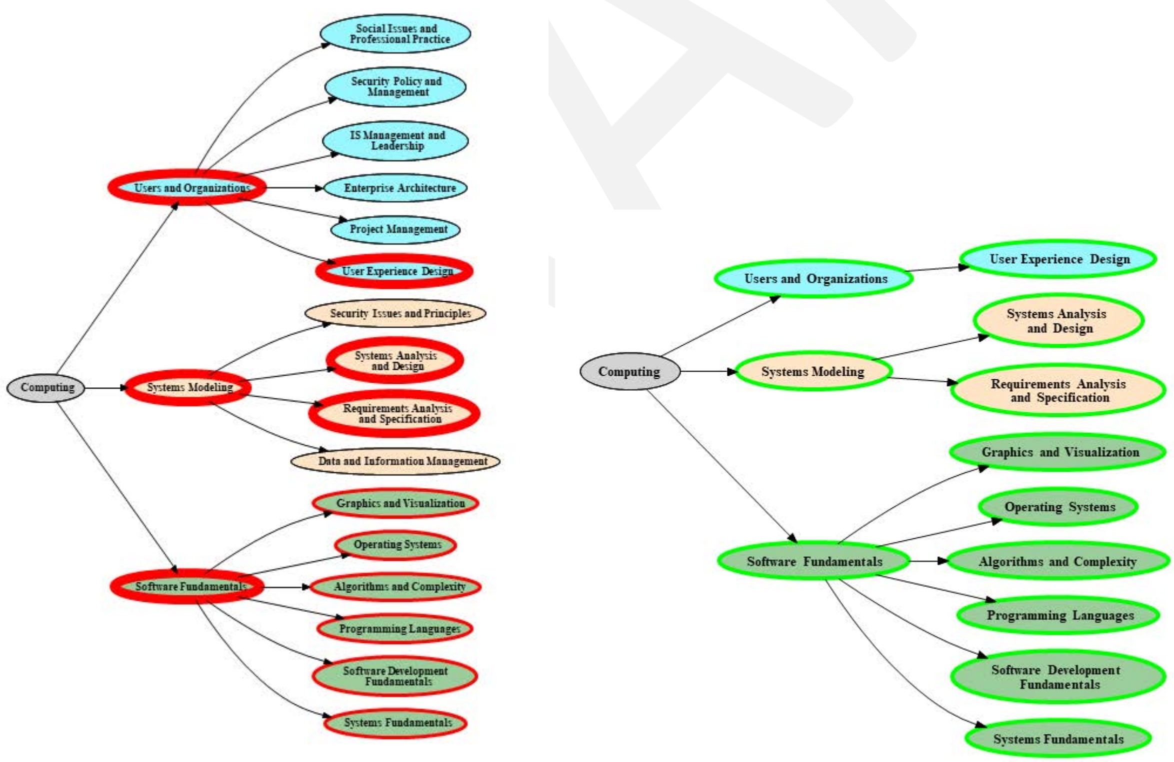


Figure G.2. The student's choice of computing categories



(a) Choosing knowledge areas (b) Final result

Figure G.3. Detailed choice of knowledge areas

If the student is satisfied with this set of knowledge areas, she may confirm and ask for a global view of how the various curricula match her interests. Based on the student's choices, the system searches for curricula that fit this intended content. In Figure G.4, the intended knowledge categories (which have been partly specified into knowledge areas) are mapped for each of the curricular guidelines. The blue squares indicate the extent to which the knowledge area/category is relevant in the corresponding curriculum. The green square is the relative match of the student choices to that of the curriculum. The calculation of the size of the blue and green squares is not fixed yet, but for example, the green square could be based on the weights that were given in the Knowledge table shown in Appendix D. Since the student is more interested in software modeling, based on the message given in Figure G.4, the student decides to explore details regarding SE and her favored knowledge categories. By hovering over a square (Figure G.5), the corresponding competencies are listed. Also displayed are the dispositions linked to the competencies along with the relative level computed from the student choices.

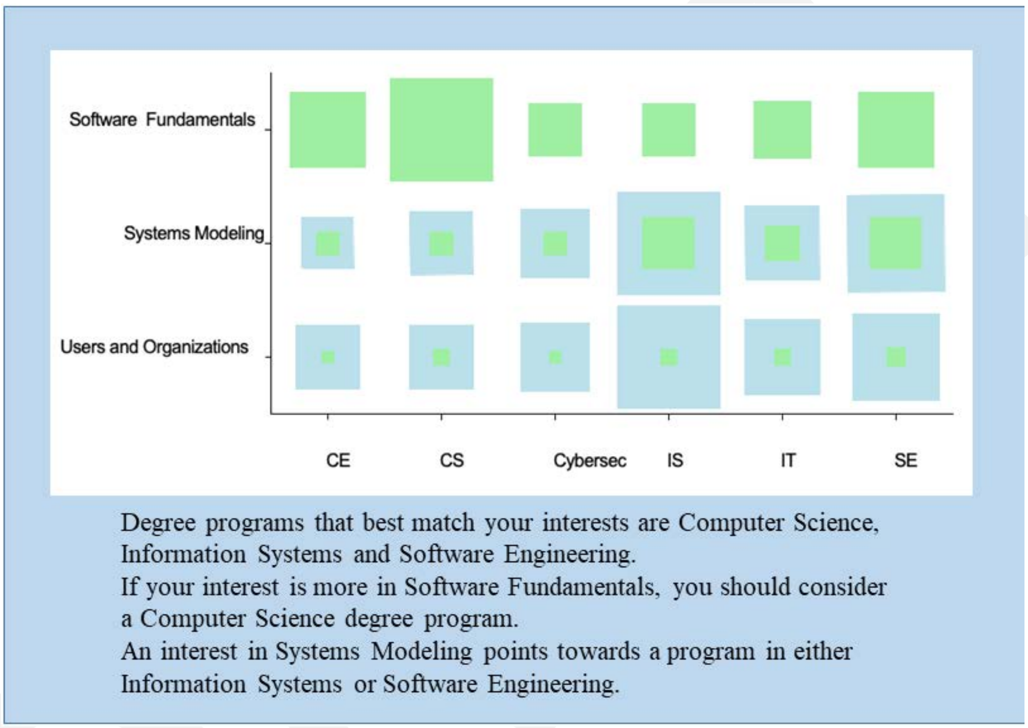


Figure G.4. Mapping of chosen knowledge categories to the curricular guidelines

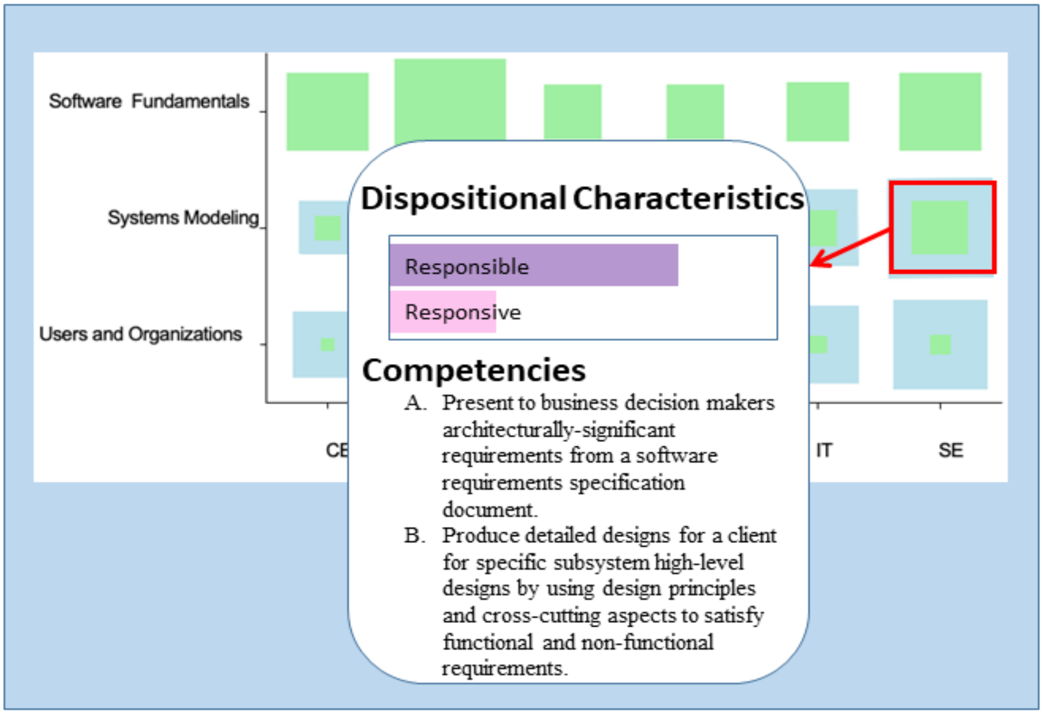


Figure G.5. Disposition and competency details

G.1.2: Case 2: Question from Industry

A user from industry has developed a list of relevant knowledge areas for which relevant skills, knowledge levels, and/or dispositions are required for the company's computing employees. She wants to find out which curriculum might potentially provide professional education for the company's employees, in their context. Initially, CS and IT seem to be available and promising.

Similar to the process that the student took in Figures G.2 and G.3 in Case 1, she decides to choose Hardware, Software Fundamentals, and Software Development as categories that seem relevant, and removes the other three categories. She then checks the knowledge areas for each of the chosen categories and chooses the areas that she believes to be relevant for her, resulting in Figure G.6.

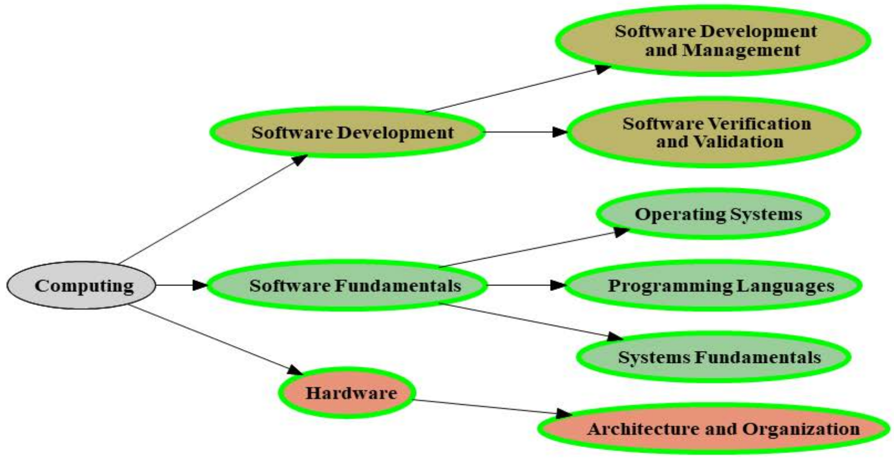


Figure G.6. Result of knowledge areas selection

The user is now able to indicate for each of the selected knowledge areas to, either or both, indicate what skill level would be required, and what dispositions are relevant. Suppose that the user indicates that she is willing to provide specifications for the knowledge area System Fundamentals. In Figure G.7, the skill level is specified by using a slider, and the disposition is specified by choosing from a menu.

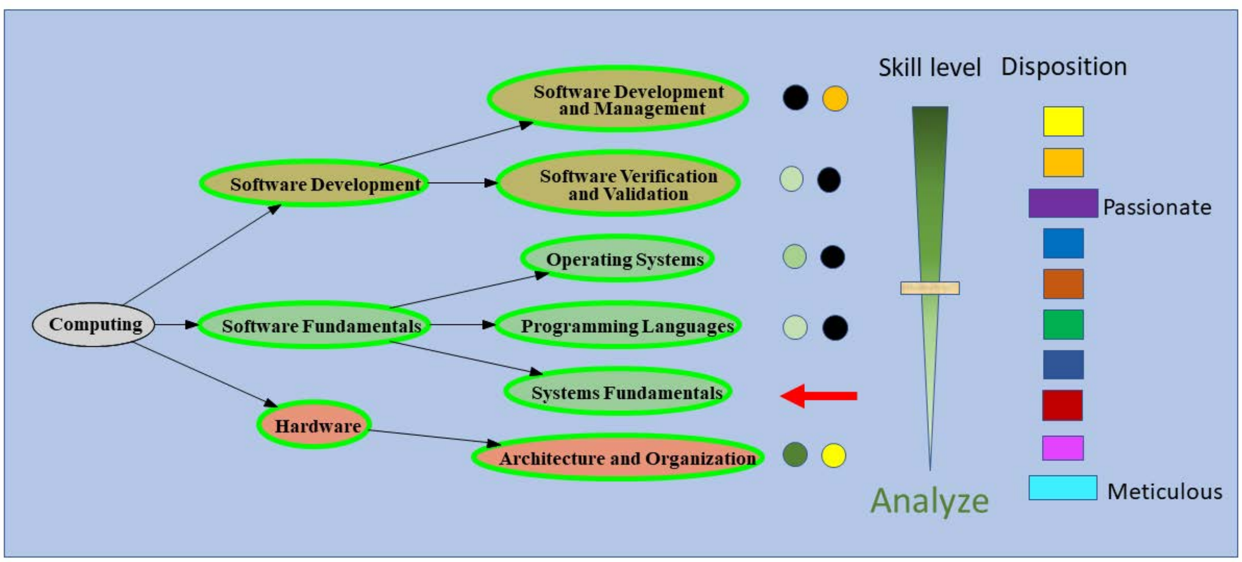


Figure G.7. Detailing skill and disposition

When all relevant specifications for the selected knowledge areas have been provided, the system generates a radar chart comparing the knowledge level for selected curricula. The distance from the center indicates the skill level related to each knowledge category. Figure G.8 compares CS and IT. The radar chart has been augmented with the specification from the user. In the example, it seems IT is the best match for the user’s required knowledge levels. This is because there is a complete coverage of the user's specifications and the curriculum content; that is, the blue CS surface completely overlaps the user's green specification surface.

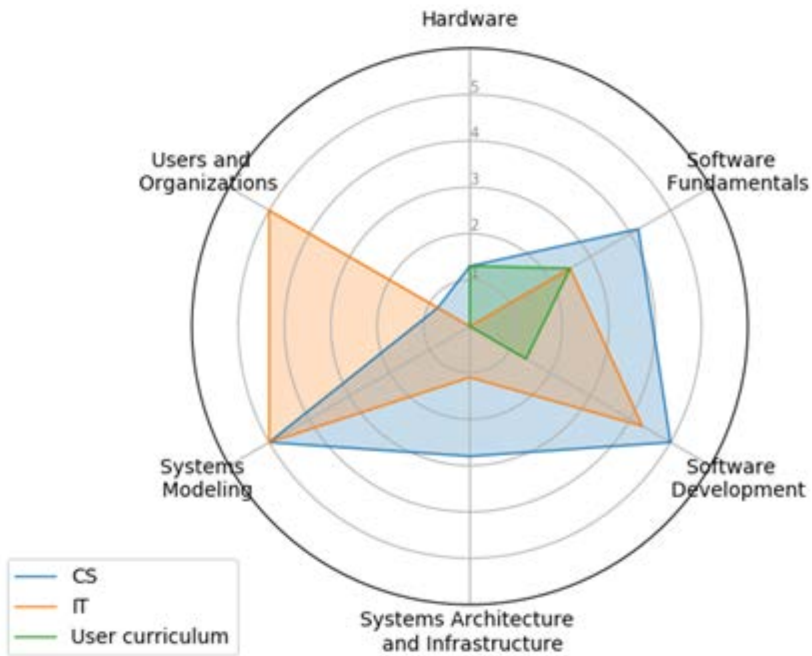


Figure G.8. Comparison of CS and IT based on knowledge level

G.1.3: Case 3: Question from Teacher

A teacher in a university faculty of Computing aims at developing a course for her domain "Human Factors in Computing". Instead of a course, it could also be a textbook, an interactive electronic learning environment, or a mixture of these. The content of this course will be considered an essential part in the Bachelor curriculum for the departments IT, SE, and CS. She decides to find out what would be relevant for each of these curricular guidelines, in order to compose a course that will be sufficient for all departments.

Similar to the process that the student took in Figures G.1 and G.2 in Case 1, she decides to choose Software Fundamentals, Software Development, and Users and Organization as categories that seem relevant, and removes the other three categories. She then checks the knowledge areas for each of the chosen categories and chooses the areas that she believes to be relevant for her, resulting in Figure G.9.

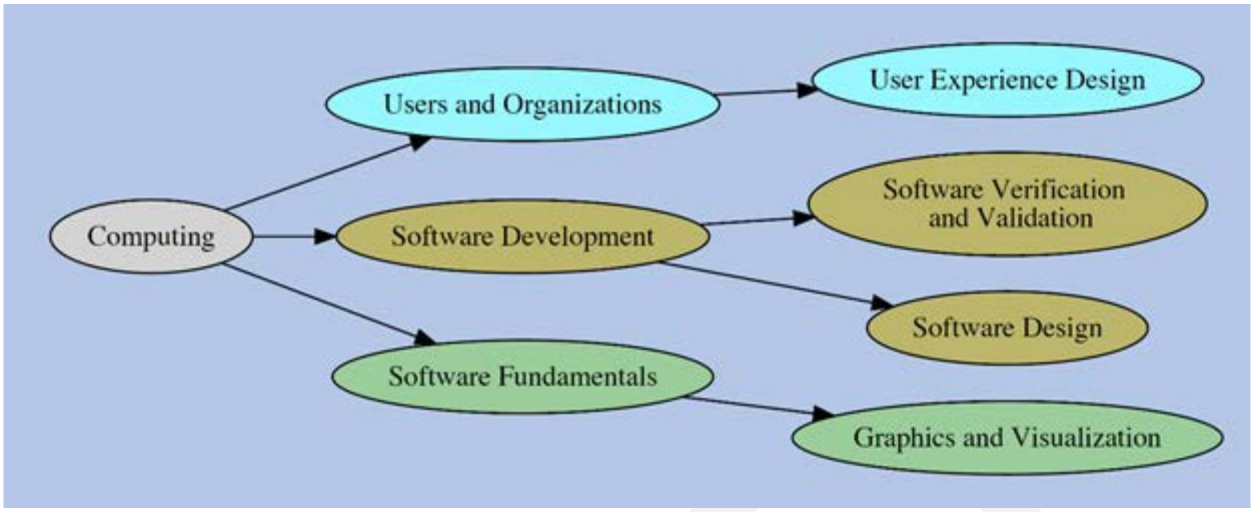


Figure G.9. Chosen knowledge areas for the new course design

For each of the chosen knowledge areas (User Experience Design; Software Verification and Validation; Software Design; and Graphics and Visualization), she will be able to find the relevant competency statements and dispositions from the chosen curriculum guidelines (IT, SE, and CS). Figure G.10 shows what she will get for the area User Experience Design after she chose the dispositions and competencies to keep for her course design. The process is the same for the other chosen knowledge areas.

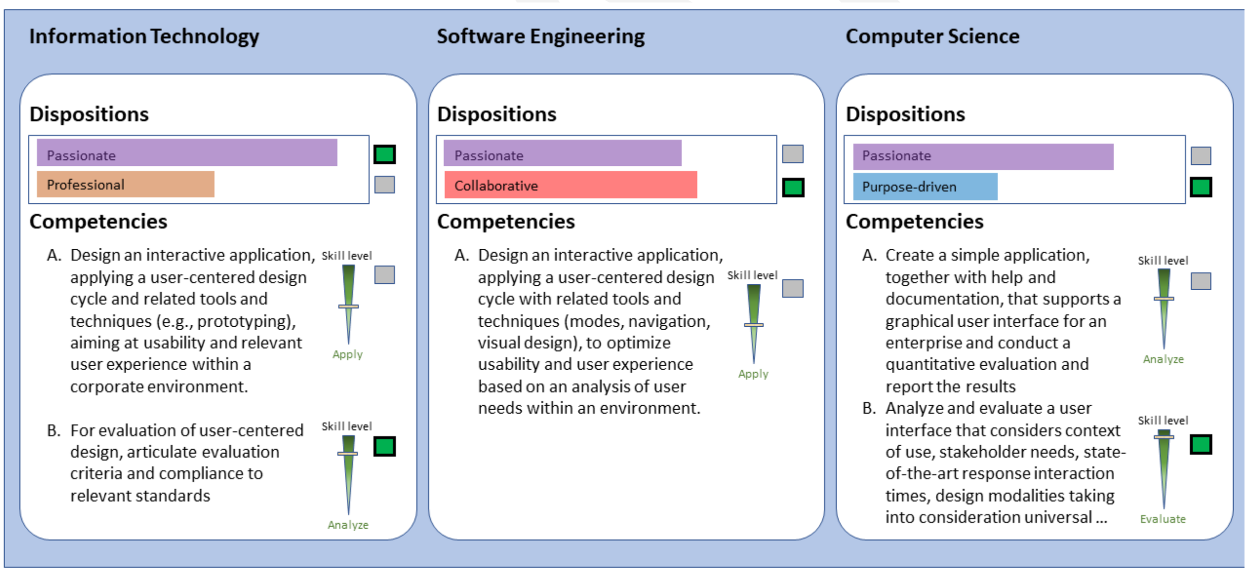


Figure G.10. Potentially relevant competencies with their skill level, and dispositions for the User Experience Design area.

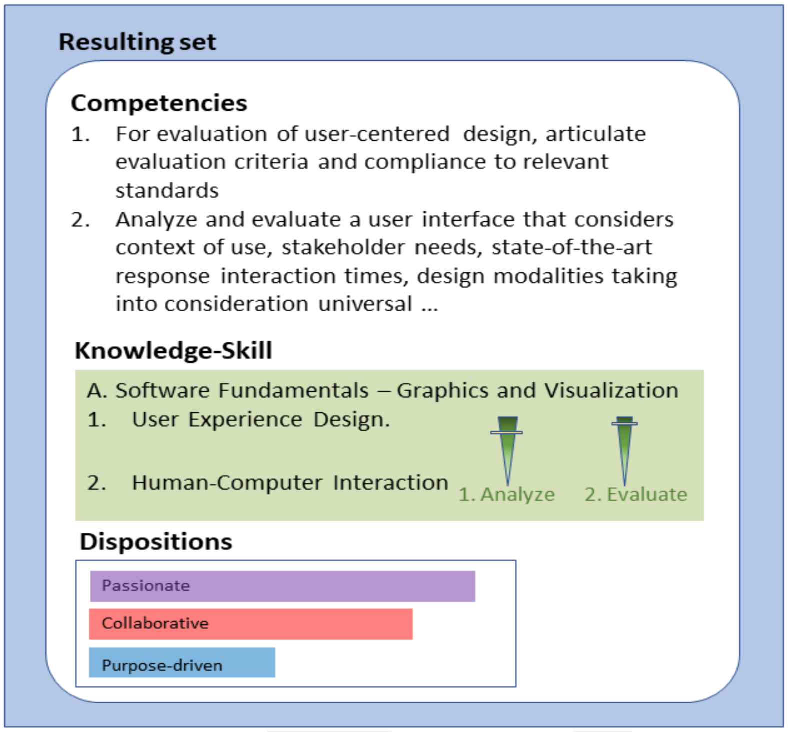


Figure G.11. Resulting chosen set of competencies and dispositions

Then the user may ask for an overview of the total set of chosen dispositions and knowledge with their skill levels (Figure G.11), and, if satisfied, consider this as the User Experience part for her course to serve students from the different departments.

The same procedure can help her to specify other parts of her course, in this example on Software Verification and validation, on Software Design, and on Graphics and Visualization. The user in this example might well consider finding adequate knowledge like Interactive Application Design at the Software Design knowledge area, and the disposition of Collaborative Attitude in the Software Verification and Validation knowledge area.

G.1.4: Case 4: Question from Educational Authority

An official examiner on behalf of the Government's Ministry of Education needs to assess or accredit a bachelor computing curriculum of one of the country's public universities. We presume she has listed the names of all universities in her country or region. She might start by selecting the name of the university to be assessed, and choosing the program of one of the departments, e.g., Software Engineering (SE) (Figure G.12).

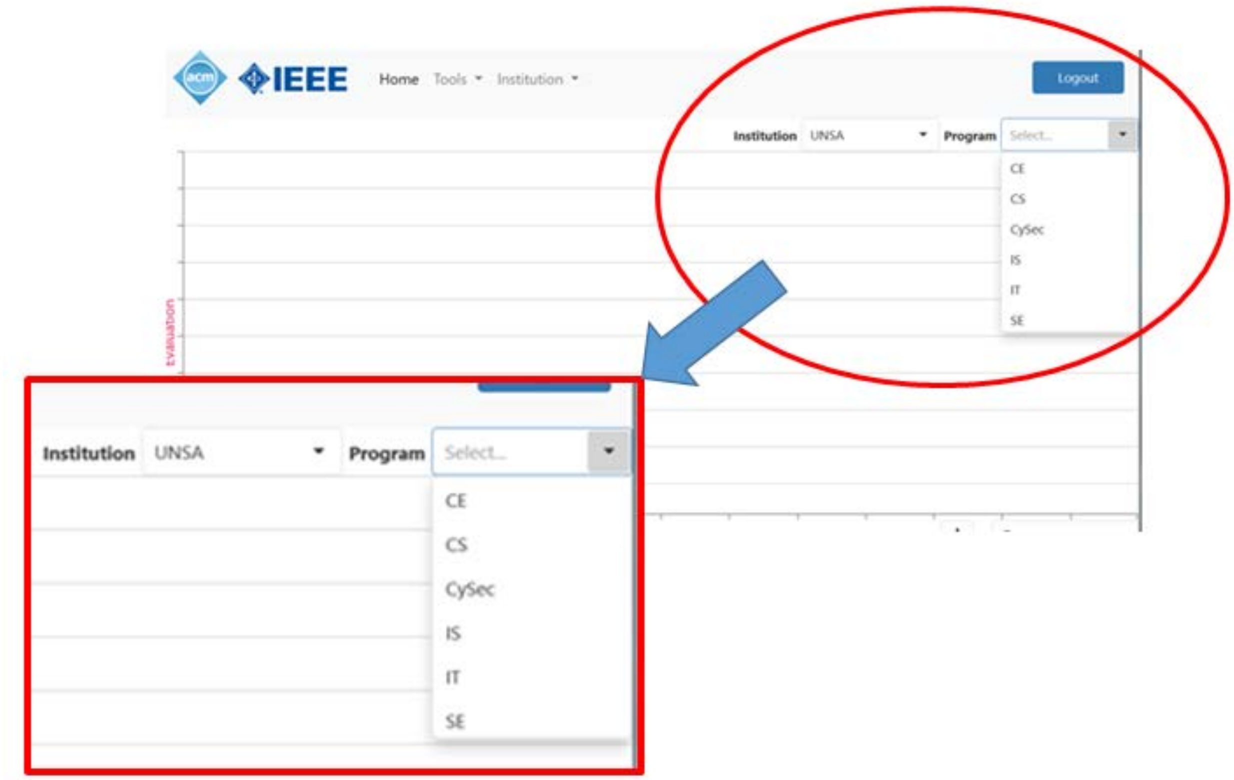


Figure G.12. Indicating the Institution to be assessed and choosing the curriculum

Figure G.13 shows the result: a graph that indicates the minimum and maximum weights of each knowledge element (given on the X axis) in the curriculum guidelines for SE. Below the graph, the six knowledge areas are displayed.

The user may expand each of these knowledge areas to access the individual knowledge elements. In Figure G.14, the user has expanded the Users and Organizations knowledge category, resulting in the knowledge elements, e.g., Social Issues and Professional Practice, Security Policy and Management, etc. On the right side, the user has started inserting the actual weight for each element as found in the BA curriculum description of the department assessed. For each weight inserted, the interactive visualization will update the graph to show how the faculty scores compared to the guidelines. So, for example, in Figure G.13, as the institution values have not yet been input yet, all of the knowledge elements have the Evaluation value 0 (in the y-axis). In Figure G.14, the graph has been updated to reflect the input value, e.g., the value for Social Issues and the Professional Practice is 6.

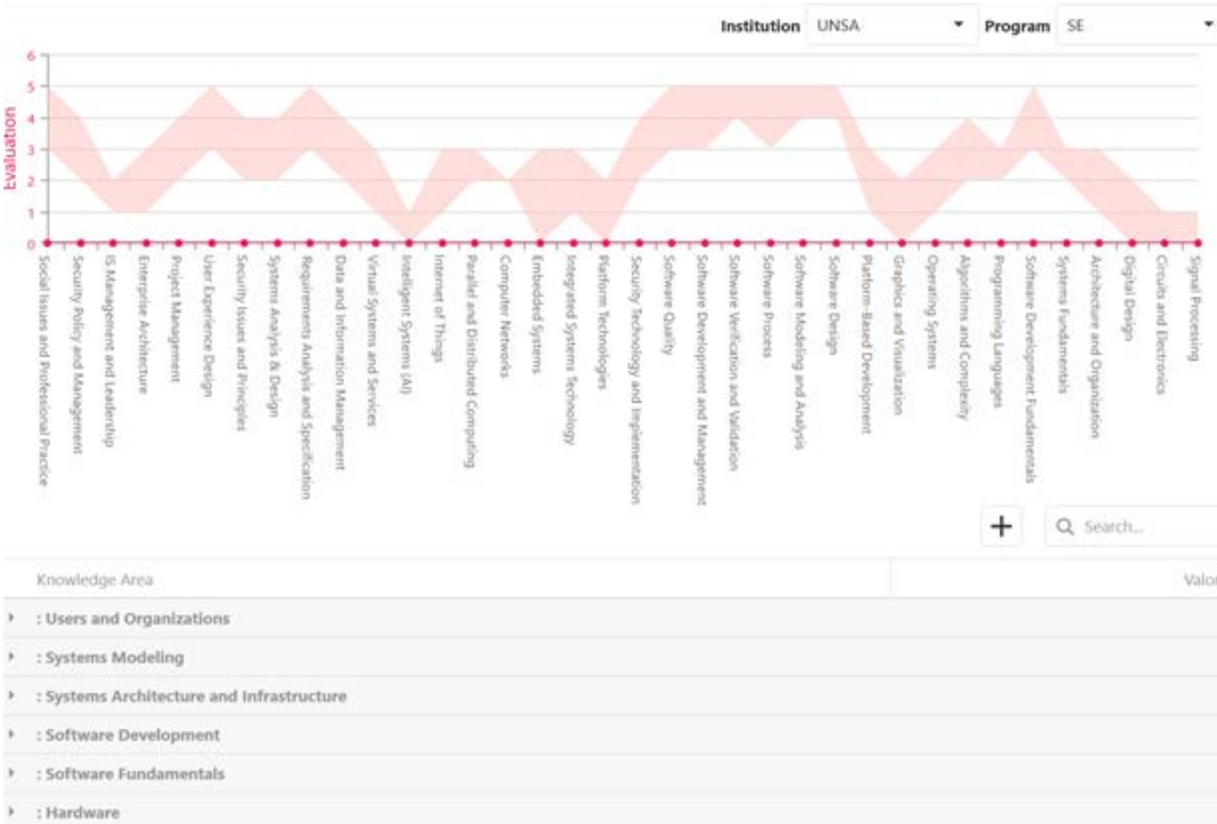


Figure G.13. Weight-range of knowledge areas in the curriculum guidelines for SE

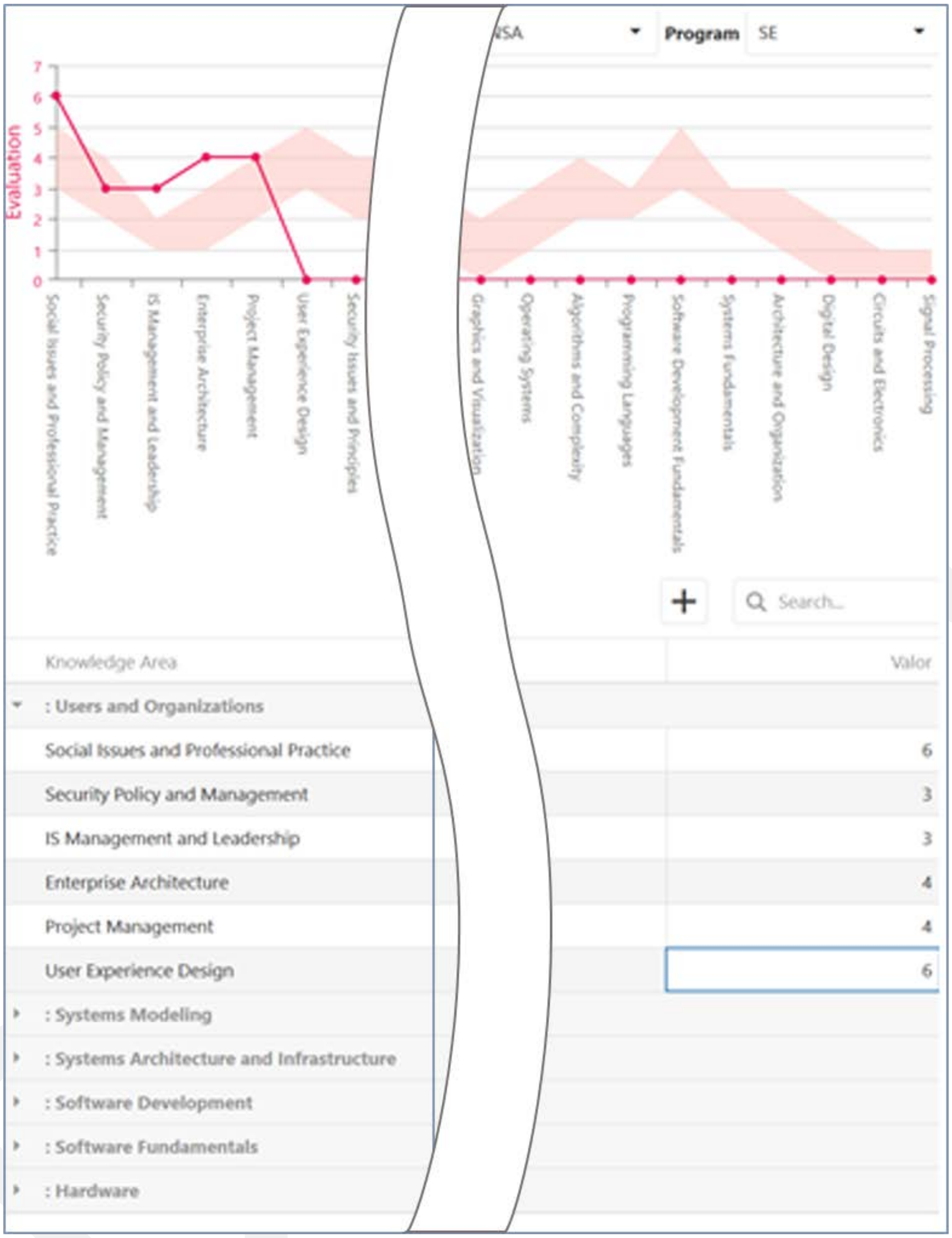


Figure G.14. Inserting weights found in the BA curriculum description of the department (note: the middle portion has been elided)

When the user has finished this input process for all knowledge domains, the resulting comparison looks like Figure G.15, showing that this faculty generally conforms to the guidelines, is relatively strong in the domain of Users and Organization, and relatively weak on Software Development.

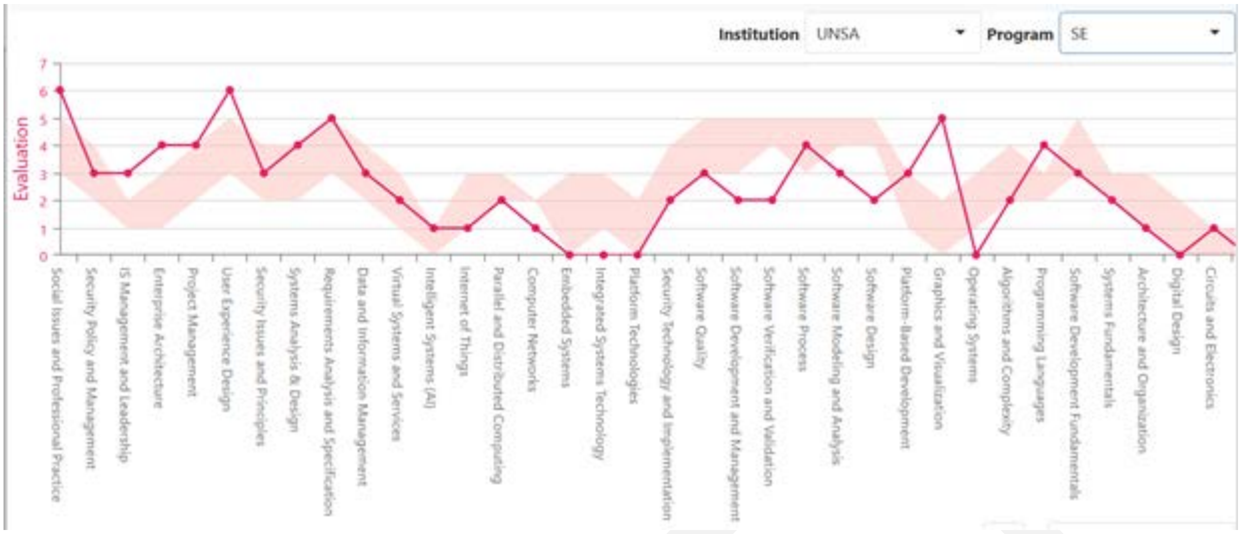


Figure G.15. Resulting state of the department's curriculum compared to the guidelines

G.2: Comparison of Competency Specifications

Figure G.16 shows two competency specifications in table format side by side. Ref# and Title denotes the reference number and title, respectively, of a competency specification. The other three columns show the competency statement, dispositions, and knowledge-skill pairs for the competency specification. The colors show changes and similarities between the two competency specifications. For example, Disposition D-2 and Knowledge K(X-3) and K(X-4) are colored pink as they are the same. However, the corresponding skills for K(X-3) and K(X-4) are different so they are colored orange.

Ref#	Title	Competency Statement	Dispositions	Knowledge-Skill Pairs	
CA1	*lorem ipsum*	*something*	D-1	K(X-1)	B-3
			D-2	K(X-2)	B-4
				K(X-3)	B-3
				K(X-4)	B-2

Ref#	Title	Competency Statement	Dispositions	Knowledge-Skill Pairs	
CA2	*lorem ipsum*	*something*	D-3	K(X-3)	B-4
			D-2	K(X-4)	B-3
				K(X-6)	B-2

Figure G.16. Side by side comparison of competency specification
(Note: Values are examples and not actual values.)

G.3: Various Visualizations of Knowledge

Figures G.17, G.18, and G.19 are all visualizations of the same data, specifically Table 5.3 in Chapter 5.

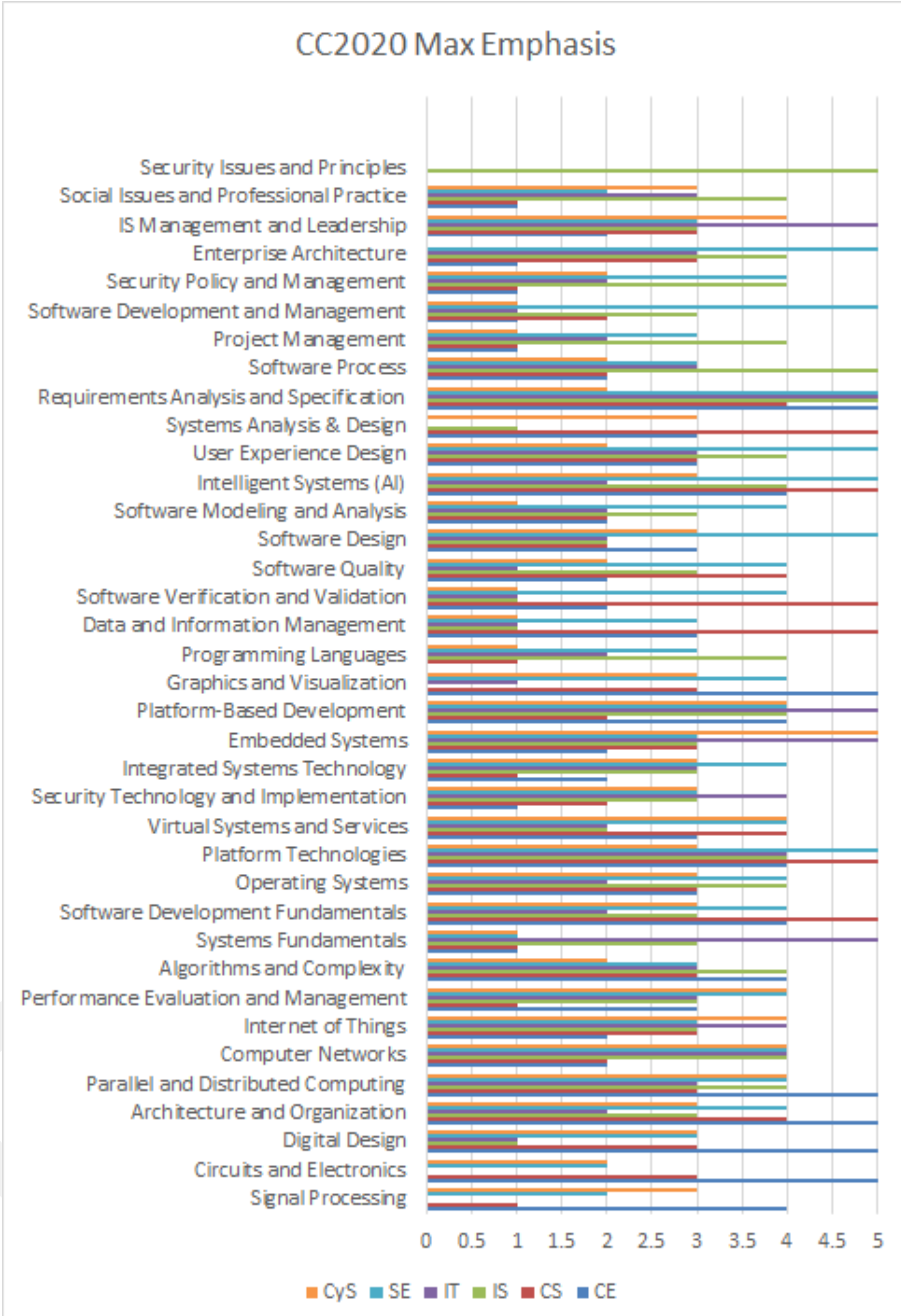


Figure G.17. Bar chart showing the maximum emphasis of knowledge areas

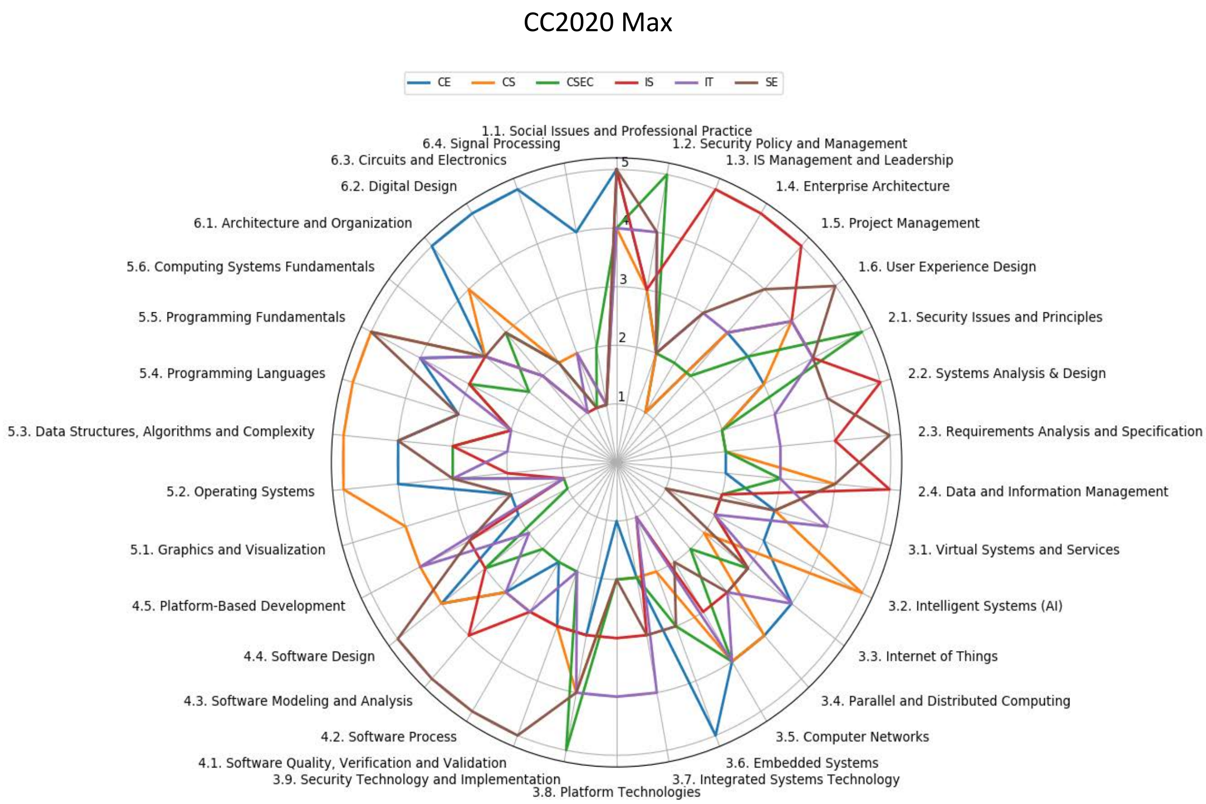


Figure G.18. Radar Chart showing maximum emphasis of knowledge areas

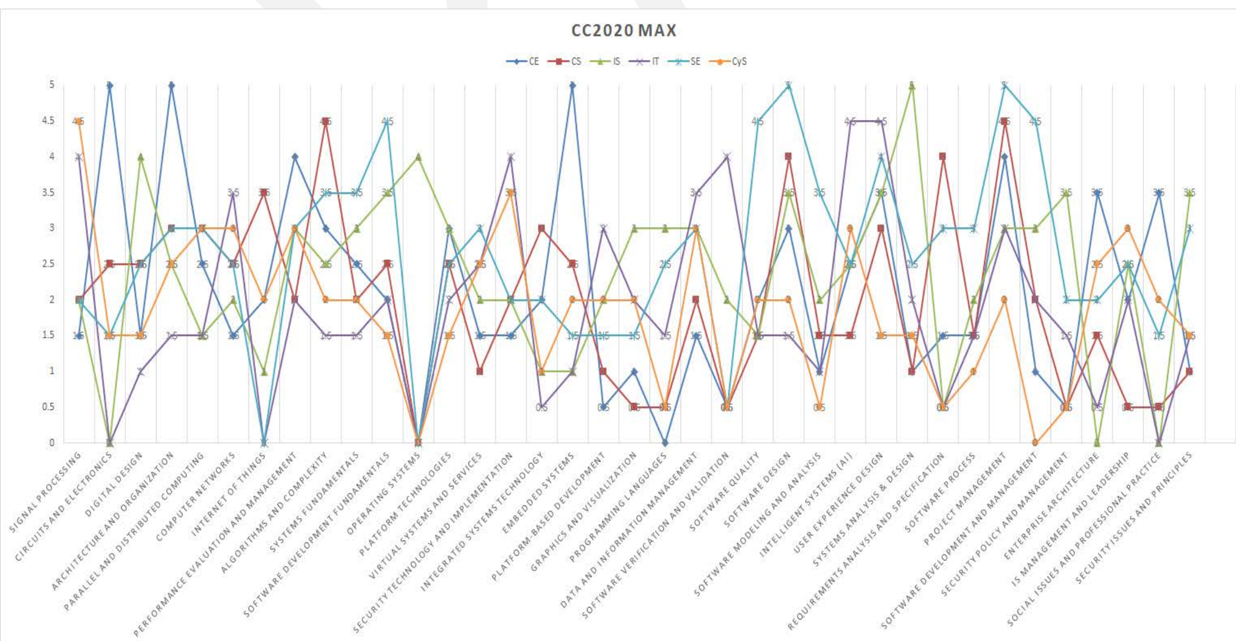


Figure G.19. Line Chart showing maximum emphasis of knowledge areas

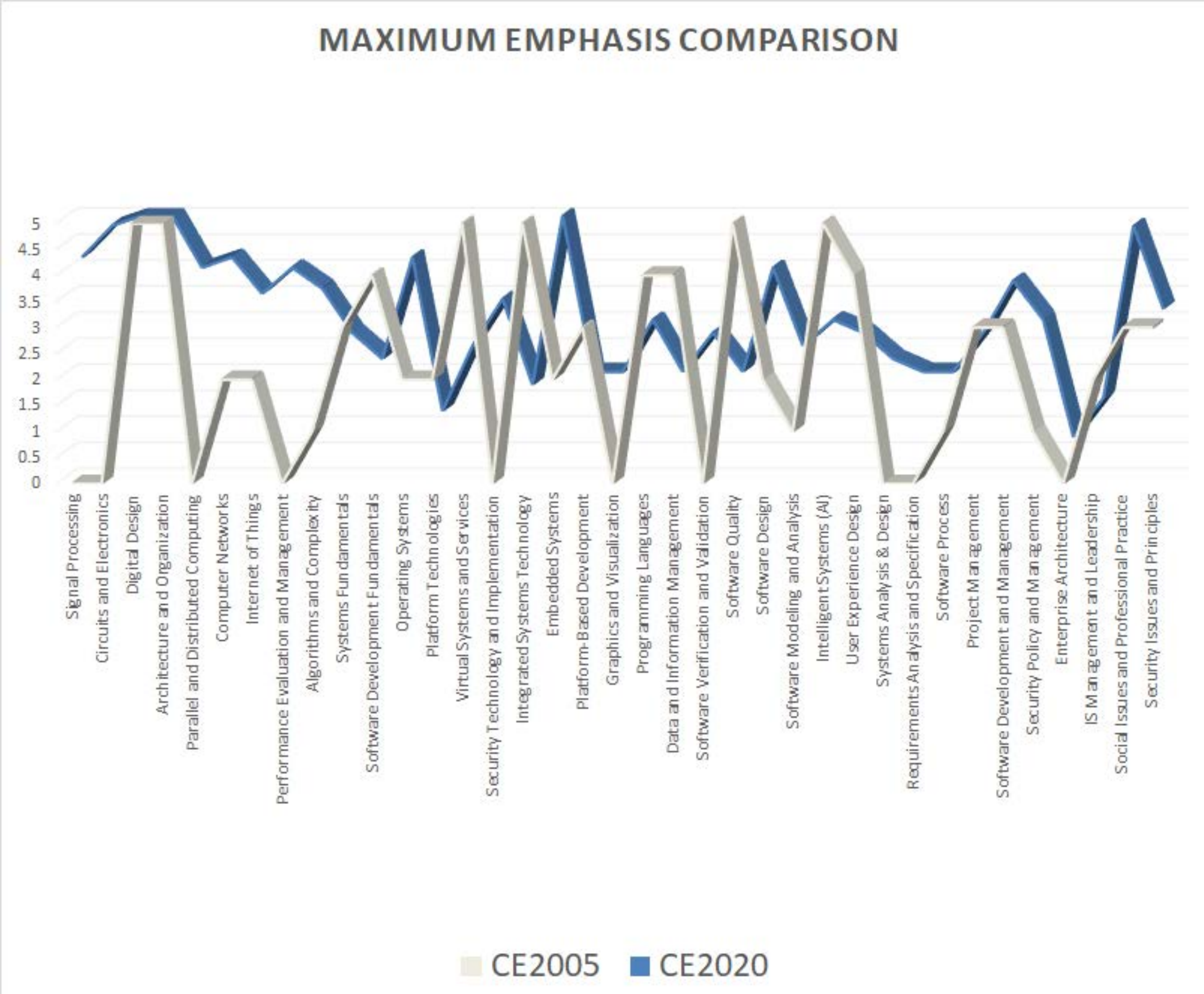


Figure G.20. Ribbon Chart comparing the maximum emphasis of knowledge areas between CE 2005 and CE 2020

Figure G.20 compares the CE knowledge area emphasis between the values that were given in CC2005 and CC2020. It shows that some of the knowledge areas, such as Signal Processing and Software Verification and Validation, had 0 emphasis indicating that they did not exist in CE2005.

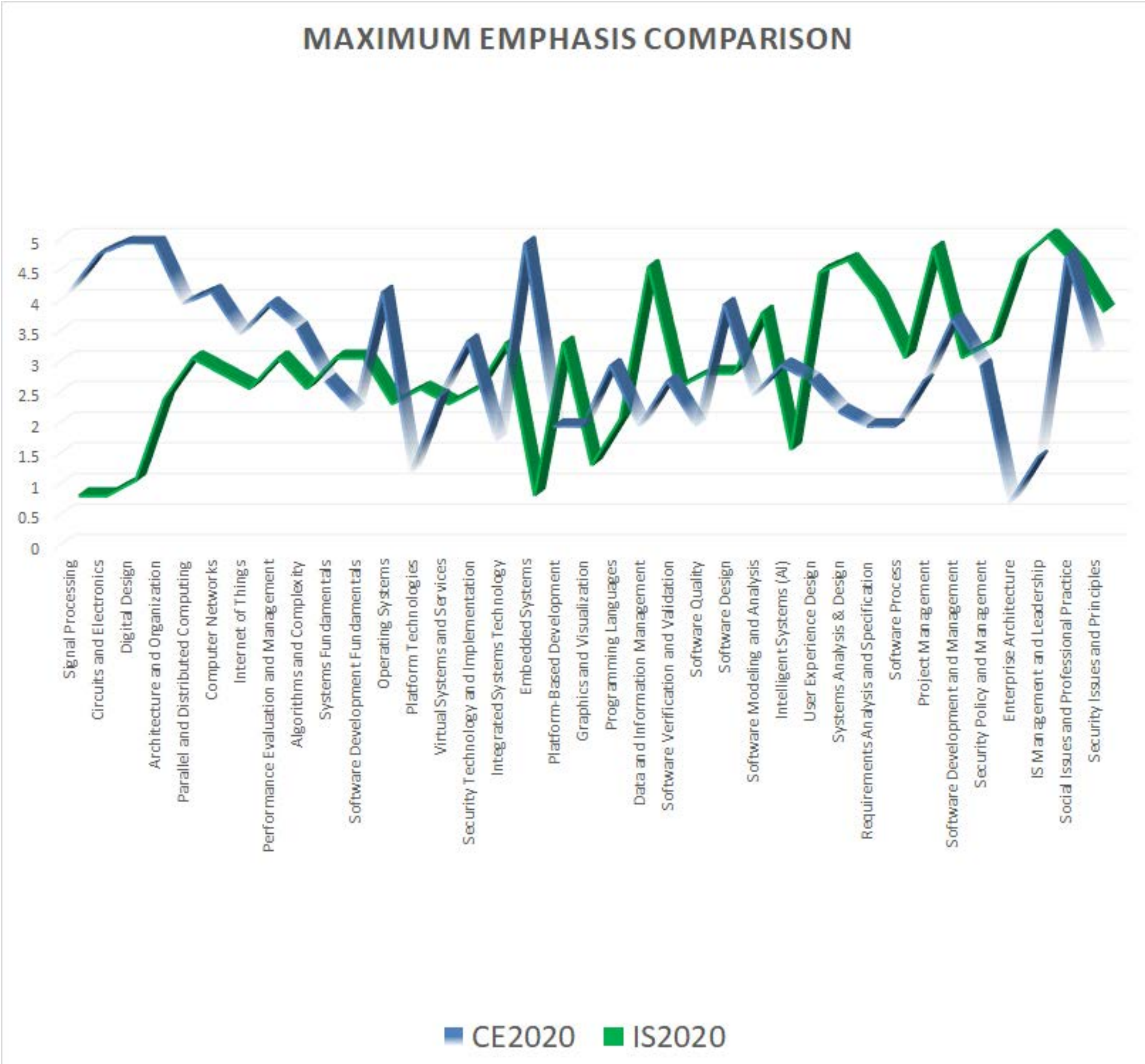


Figure G.21. Ribbon Chart comparing the maximum emphasis of knowledge areas between CE 2020 and IS 2020

Figure G.21 shows a comparison between the emphasis on knowledge areas for the curriculums CE and IS (in 2020). It shows a systematic difference at the left in the region of Hardware and Software Fundamentals where CE has a strong emphasis. At the other end of the graph, IS is emphasizing knowledge on Users and Organization more than CE.

G.4: Visualizing Full Curricula

The visualization in Figure G.22 centers around the CS node that links the knowledge areas (KAs), their core knowledge units (KUs) and their respective topics [Mar2]. KA nodes are near the center and colored gray. KU nodes are placed just outside of the KA nodes having labels starting with U, and topics nodes are placed in the outer part having labels starting with T.

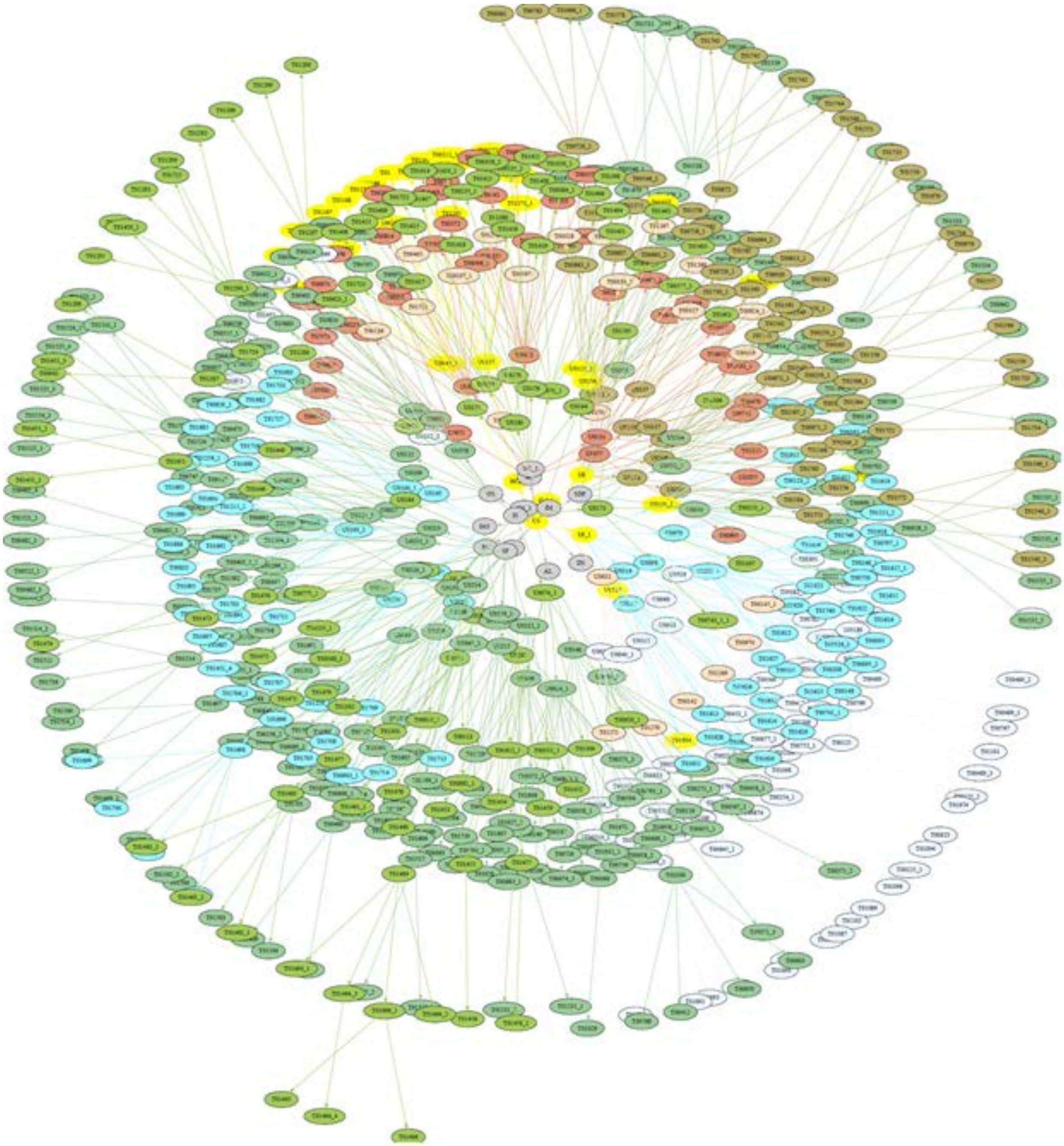


Figure G.22. Graph-based structure of the core components of CS2013.

To provide a detailed example, consider the highlighted parts in yellow. These yellow nodes represent the core aspects of the CS2013 curriculum specification that relates to User Experience Design (in CS2013 labelled "HCI").

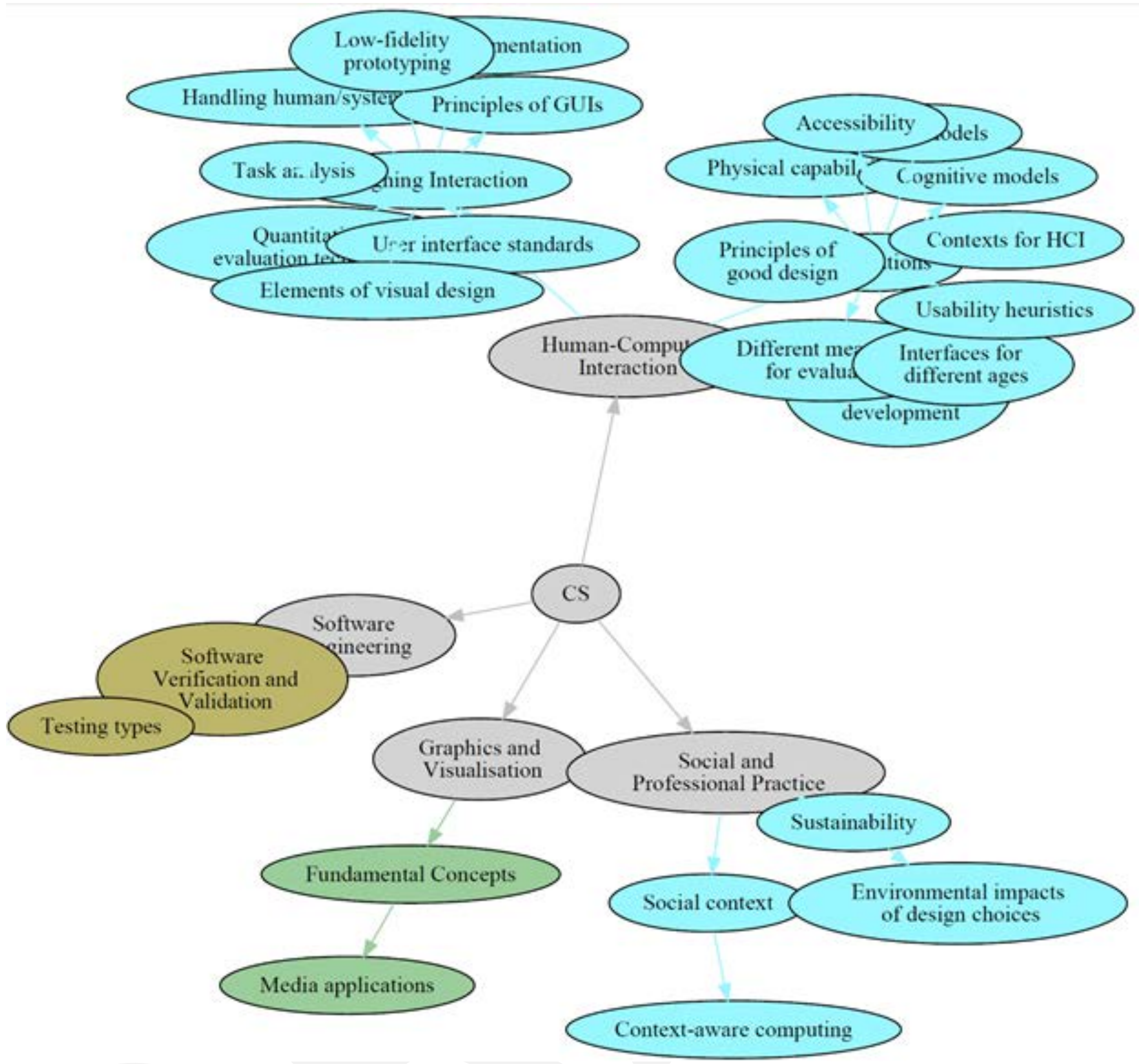


Figure G.23. Close-up of the HCI part of Figure G.22

Figure G.23 shows the actual labels of these nodes. Note that this includes not only HCI (Human-Computer Interaction), but non-HCI areas such as Software Engineering, Graphics and Visualization, and Social and Professional Practice. This is because the CS2013 curriculum mentions a link from these areas to HCI aspects. For example, the part concerned with Testing types which belongs to Software Engineering states "Testing types, including human computer interface, usability, reliability, security, conformance to specification (cross-reference IAS/Secure Software Engineering)" [p.182, CS2013]. As "human computer interface" is concerned with HCI, this part of Software Engineering is included in Figure G.23.

G.5: References for Appendix G

[Tak1] S. Takada et al., "Toward Understanding Computing Curricula: Visualization Using Competency within the CC2020 Project," Educ. Inf. Technol., 2020.

[Mar2] Marshall, Linda (2014) A graph-based framework for comparing curricula. Ph.D. thesis, University of Pretoria, South Africa.

Appendix H: Glossary and Nomenclature

This report has been written for the global community of educators, industry, students, and the general public. However, across the world in computing, different terms can be used to mean the same thing, and the same terms can have different meanings. While it would be ideal to have a consistent naming system globally, the task force recognizes that many terms are entrenched in a country's or region's culture. In an attempt to ensure transparency and readability, a list has been compiled for terms that could be confusing. A set of CC2020 definitions are summarized in this appendix and have been translated into the most common languages in the world. Hopefully, this list will enable the reader to understand the terminology used in computing in their own language around the world.

H.1: CC2020 Report Definitions

Table H.1 lists the working definitions that appear in this report, as compiled by the CC2020 task force.

Table H.1 Definitions for CC2020

Term	**CC2020 Definitions**
Accreditation	Official approval given by an organization stating that somebody or something has achieved a required standard
Algorithm	A set of rules to be followed in calculations or other problem-solving operations
AP	Advanced placement not used outside the USA
Baccalaureate	A bachelor's degree
Chair of Department	Head of Department or Chair of Department
Class	A group of students studying the same course or degree
College	Outside the USA it can be another name for a High School or an organizational unit in a university; in the USA it is a term for a university
Community College	Two-year school post high school primarily used in the USA, very rarely used elsewhere
Competency	Knowledge + Skills + Dispositions in context
Core course/curriculum	Compulsory courses towards a degree
Course	A component of a degree or in some countries a whole degree
Credit hours	The number of hours for each credit towards a degree
Credits	The points a student receives after passing the assessments towards the course or degree
Curriculum	All the different courses of study that are taught in a school or for a particular subject
Engineering	Concerned with the design, building, and use of something. It does not imply a title of engineer
Faculty	Teachers and researchers in a university
Freshman	Freshman a term for a first year degree student, generally common

Term	**CC2020 Definitions**
Graduate, Post-Graduate	Graduate – has completed a bachelor's degree; Post-Graduate – Masters and Doctoral degrees
Informatics	European term for computing or sometimes information systems or computer science
Information and Communication Technology (ICT)	A fairly common global term for the "computing" technology industry as a whole. Used in some places interchangeably with Information Technology
Information Technology (IT)	A branch of "Computing" with an approved curriculum. A fairly common global term for the "computing" technology industry as a whole. Used in many places interchangeably with Information and Communication Technology
Junior	USA term for a third-year student
K-12	Kindergarten to year 12 (rarely used outside the USA and Canada)
Lecturer	A rank of faculty or a teacher in a university
Middle School	Also known as intermediate school. Different meanings in different countries generally two or three-year schools at either 4 to 6, year 7 to 8 or 9, or 11 - 14.
Module	Either a course or a part of a course
Paper	Usually a product a student produces to pass a course or examination, or a published article
Professor or a visiting professor	A Visiting Professor or in some countries (USA) a part-time Professor
Program(me)	All the courses that make up a degree
Quarter	A quarter of an academic year
Semester	Half an academic year
Senior	USA term for a fourth-year student
Sophomore	USA term for a second-year student
Subject	Similar to a course but not always used at university level
Technology	The application of scientific knowledge for practical purposes, especially in industry
Trimester	One third of an academic year
Undergraduate	Studying towards a bachelor's degree

H.2: Definitions/Nomenclature on a Global Scale

Tables H.2 provides translations for the CC2020 working definitions into Arabic, Hindi, Japanese, Chinese and Russian. Similarly, Table H.3 translates the list of working definitions into French, Italian, German, Spanish (Latin American and European), and Portuguese.

Table H.2 Definition Equivalents for Arabic, Hindi, Japanese, Chinese, and Russian

Term	CC2020 Definitions	Arabic	India (Hindi)	Japanese	Chinese	Russian
Accreditation	Official approval given by an organization stating that somebody or something has achieved a required standard	اعتماد أكاديمي	मान्यता	認定	学科评估	Аккредитация
Algorithm	A set of rules to be followed in calculations or other problem-solving operations	خوارزمية	विधि	アルゴリズム	算法	Алгоритм
AP	Advanced placement not used outside the USA			アドバンスド・プレイスメント	大学先修课	Not used
Baccalaureate	A bachelor's degree	بكالوريوس	स्नातक	学士	学士	Бакалавриат
Chair of Department	Head of Department or Chair of Department	رئيس القسم	विभागाध्यक्ष	学部長 or 学科長	系主任	Заведующий кафедрой
Class	A group of students studying the same course or degree	صف دراسي	कक्षा	クラス	班级	Группа, поток (several groups at a big lecture)
College	Outside the USA it can be another name for a High School or an organizational unit in a university; in the USA it is a term for a university	كلية	महाविद्यालय	大学	学院	Not used in the sense
Community College	Two-year school post high school primarily used in the USA, very rarely used elsewhere	كلية المجتمع			大专	Not used
Competency	Knowledge + Skills + Dispositions in context	كفاءة	योग्यता	コンピテンシ	胜任力	Компетенции

Term	CC2020 Definitions	Arabic	India (Hindi)	Japanese	Chinese	Russian
Core course/curriculum	Compulsory courses towards a degree	مقررات أساسية / خطة دراسية	पाठ्यक्रम	必修コース/カリキュラム	核心课程/课程体系	Обязательные курсы
Course	A component of a degree or in some countries a whole degree	مقرر دراسي	विषय	コース	课程	Курс
Credit hours	The number of hours for each credit towards a degree	ساعات معتمدة		単位取得時間	学时	Часы
Credits	The points a student receives after passing the assessments towards the course or degree	نقاط مكتسبة		単位	学分	Кредиты
Curriculum	All the different courses of study that are taught in a school or for a particular subject	خطة دراسية	पाठ्यक्रम	カリキュラム	课程体系	Учебный план
Engineering	Concerned with the design, building, and use of something. It does not imply a title of engineer	هندسة	अभियांत्रिकी	エンジニアリング or 工学	工程	Разработка, проектирование
Faculty	Teachers and researchers in a university	عضو هيئة تدريس	संकाय	講師 or 教員 (or 学部)	学部	ППС (abbreviation) Профессорско-преподавательский состав
Freshman	Freshman a term for a first year degree student, generally common	طالب السنة الجامعية الأولى		1年生	一年级学生	Первокурсник
Graduate, Post-Graduate	Graduate – has completed a Bachelor's degree - Post Graduate – Masters and Doctoral degrees	خريج، طالب الدراسات العليا	स्नातक, स्नातकोत्तर	学部卒, 大学院卒	毕业、研究生院	Выпускник, Аспирант
Informatics	European term for computing or sometimes information systems or computer science	المعلوماتية	सूचना विज्ञान	情報学	情報学	Информатика
Information and Communication Technology (ICT)	A fairly common global term for the "computing" technology industry as a whole. Used in some places interchangeably with Information Technology	تقنية المعلومات والاتصالات	सूचना और संचार प्रौद्योगिकी	情報通信技術	信息和通信技术	Информационно-коммуникационные технологии ИКТ

Term	CC2020 Definitions	Arabic	India (Hindi)	Japanese	Chinese	Russian
Information Technology (IT)	A branch of "Computing" with an approved curriculum. A fairly common global term for the "computing" technology industry as a whole. Used in many places interchangeably with Information and Communication Technology	تقنية المعلومات	सूचना प्रौद्योगिकी	情報技術	信息技术	Инормационные технологии
Junior	USA term for a third year student	طالب السنة الجامعية الثالثة		3年生	三年级学生	No special term for this
K-12	Kindergarten to year 12 (rarely used outside the USA and Canada)	مراحل التعليم العام			K-12	Детский сад to year 7
Lecturer	A rank of faculty or a teacher in a university	محاضر	व्याख्याता	講師 or 教員	讲师	Лектор
Middle School	Also known as intermediate school. Different meanings in different countries generally two or three year schools at either 4 to 6, year 7 to 8 or 9, or 11 - 14.	المرحلة المتوسطة	माध्यमिक विद्यालय	中学校	中学	Средняя школа 7-15/16
Module	Either a course or a part of a course	وحدة			模块	Раздел, модуль
Paper	Usually a product a student produces to pass a course or examination, or a published article	ورقة أو مقالة علمية	परीक्षा	試験答案やレポートなどの成果物	论文	Работа, документ
Professor or a visiting professor	A Visiting Professor or in some countries (USA) a part-time Professor	أستاذ جامعي أو أستاذ زائر	प्राध्यापक	非常勤教授 or 非常勤講師	访问教授	No special term for this
Program(me)	All the courses that make up a degree	برنامج دراسي		プログラム	培养方案	Специальность
Quarter	A quarter of an academic year		तिमाही	学期（クオーター）	NA	Not used
Semester	Half an academic year	فصل دراسي	छमाही	学期（セメスター）	学期	Семестр
Senior	USA term for a fourth year student	طالب السنة الجامعية الرابعة		4年生	四年级学生	Старшекурсник
Sophomore	USA term for a second year student	طالب السنة الجامعية الثانية		2年生	二年级学生	No special term for this

Term	CC2020 Definitions	Arabic	India (Hindi)	Japanese	Chinese	Russian
Subject	Similar to a course but not always used at university level	موضوع	विषय	科目	科目	Предмет
Technology	The application of scientific knowledge for practical purposes, especially in industry	التقنية	प्रौद्योगिकी	テクノロジ	技术	Технология
Trimester	One third of an academic year					Not used
Undergraduate	Studying towards a bachelor's degree	طالب المرحلة الجامعية	पूर्वस्नातक	学部生	本科生	Студент

Table H.3 Definition Equivalents for French, Italian, German, Spanish (Latin America), Spanish (Europe), and Portuguese

Term	**CC2020 Definitions**	**French (Europe)**	**Italian**	**German**	**Spanish Latin America**	**Spanish Europe**	**Português**
Accreditation	Official approval given by an organization stating that somebody or something has achieved a required standard	Accréditation	Accreditamento	Akkreditierung	Acreditación	Acreditación	Credenciamento
Algorithm	A set of rules to be followed in calculations or other problem-solving operations	Un algorithme	algoritmo	Algorithmus	Algoritmo	Algoritmo	Algoritmo
AP	Advanced placement not used outside the USA			fortgeschrittene Platzierung	Not used	Not used	Validacao de creditos
Baccalaureate	A bachelor’s degree	Baccalauréat (to Enter university)	Laurea	Bachelor, Bachelorabschluss	Also “Estudios de grado”	Also “Estudios de grado”	Bacharelado
Chair of Department	Head of Department or Chair of Department	Doyen Directeur Responsable de département	Direttore del dipartimento	Abteilungsleiter (m)/ Abteilungsleiterin (f)	Jefe de departamento	Director de departamento	Chefe de Departamento
Class	A group of students studying the same course or degree	Une classe	Classe	Klasse	Clase	Clase	Turma

Term	CC2020 Definitions	French (Europe)	Italian	German	Spanish Latin America	Spanish Europe	Português
College	Outside the USA it can be another name for a High School or an organizational unit in a university; in the USA it is a term for a university	Université	Collegio	Hochschule (Polytechnic)	Universidad	Universidad	Faculdade ou Instituto de Tecnologia
Community College	Two-year school post high school primarily used in the USA, very rarely used elsewhere	Classe préparatoire (to enter prestigious schools) Brevet de Technicien du Supérieur Institut Universitaire de Technologie	Centro di formazione		Not used	Not used	Curso Técnico Profissionalizante
Competency	Knowledge + Skills + Dispositions in context	Compétences	Competenza	Kompetenz	Compentencia	Compentencia	Competência
Core course/ curriculum	Compulsory courses towards a degree	Cours du tronc commun	Corsi obbligatori	Kernpflichtfach (core course)/ Kerncurriculum (core curriculum)	Asignaturas básicas	Asignaturas obligatorias	Disciplinas obrigatórias
Course	A component of a degree or in some countries a whole degree	Un cours	Insegnamento	Studiengang, Lehrgang	Curso	Curso	Disciplina
Credit hours	The number of hours for each credit towards a degree	Le temps de présentiel	Ore corrispondenti ad un credito formativo	Semesterwochenstunden	Horas por crédito	Horas por crédito	Hora-aula
Credits	the points a student receives after passing the assessments towards the course or degree	Les crédits ou ECTS	Crediti	Leistungspunkte	Créditos	Créditos	Créditos

Term	CC2020 Definitions	French (Europe)	Italian	German	Spanish Latin America	Spanish Europe	Português
Curriculum	All the different courses of study that are taught in a school or for a particular subject	Contenu pédagogique Programme pédagogique	Curriculum	Studienplan/ Lehrplan	Plan de estudios	Plan de estudios	Currículo
Engineering	Concerned with the design, building, and use of something. It does not imply a title of engineer	Ingénierie	Ingegneria	Technik	Ingeniería	Ingeniería	Engenharia
Faculty	Teachers and researchers in a university	Faculté / Institut /École (institution) Enseignant-Chercheur (people)	Collegio dei professori	Kollegium, Lehrkörper	Profesor	Profesor	Corpo docente
Freshman	Freshman a term for a first year degree student, generally common	Un étudiant de première année	Matricola	Studienanfänger (m)/ Studienanfängerin (f)	Estudiante de primer semestre	Estudiante de primer curso	Calouro
Graduate, Post-Graduate	Graduate – has completed a Bachelor's degree - Post Graduate – Masters and Doctoral degrees	Licence (bac+3) Master (bac+5) Doctorat (bac+8)	laureato; laureato magistrale;	Studienabsolvent, Postgraduierter (m)/ Postgraduierte (f)	Graduado, Maestro, Doctor	Graduado, Post graduado	Graduado, Pos-graduado (Mestrado e Doutorado)
Informatics	European term for computing or sometimes information systems or computer science	Informatique (CS never Used in french)	Informatica	Informatik	Informática	Informática	Informática

Term	CC2020 Definitions	French (Europe)	Italian	German	Spanish Latin America	Spanish Europe	Português
Information and Communication Technology (ICT)	A fairly common global term for the "computing" technology industry as a whole. Used in some places interchangeably with Information Technology	Technologie de l'Information et de la Communication	Tecnologie dell'Informazione e della Comunicazione	Informations- und Kommunikationstechnologie	Tecnologías de la Información y la Comunicación	Tecnologías de la Información y la Comunicación	Tecnologia da Informação e Comunicação
Information Technology (IT)	A branch of "Computing" with an approved curriculum. A fairly common global term for the "computing" technology industry as a whole. Used in many places interchangeably with Information and Communication Technology	Technologie de l'Information	Tecnologie dell'Informazione	Informatik/ Informationstechnologie/ Informationstechnik	Tecnologías de la Información	Tecnologías de la Información	Tecnologia da Informação
Junior	USA term for a third year student	Un étudiant de troisième année		Student/ Studentin im 3. Studienjahr	Estudiante de tercer semestre	Estudiante de tercer curso	Veterano do terceiro
K-12	Kindergarten to year 12 (rarely used outside the USA and Canada)	Le primaire (3-10yo) Le secondaire au Collège (11-15yo)		vom Kindergarten bis zum Abitur	Educación preuniversitaria	Educación preuniversitaria	Educação Básica
Lecturer	A rank of faculty or a teacher in a university	Enseignant	Docente	Dozent (m) / Dozentin (f)	Profesor de tiempo completo	Profesor Titular	Professor

Term	CC2020 Definitions	French (Europe)	Italian	German	Spanish Latin America	Spanish Europe	Português
Middle School	Also known as intermediate school. Different meanings in different countries generally two or three year schools at either 4 to 6, year 7 to 8 or 9, or 11 - 14.	École maternelle (3-6) École élémentaire (6-10) Collège (11-15) Lycée (16-18)	Scuola media	Hauptschule (Year 5-9), Mittelschule (Year 5- 10), Gymnasium (year 5-12)	Educación básica	Educación primaria (4-12)	Ensino Fundamental
Module	Either a course or a part of a course	Un module / une unité d'enseignement	Modulo	Modul	Módulo	Módulo	Módulo
Paper	Usually a product a student produces to pass a course or examination, or a published article		Scritto: prodotto da uno studente per superare un esame	wissenschaftliche Ar beit	Prueba, examen	Examen, Trabajo, Artículo, Prueba	Artigo, if the last work of the degree is called Trabalho Final de Curso
Professor or a visiting professor	A Visiting Professor or in some countries (USA) a part-time Professor	Professeur invité	Professore	Professor, Gastprofessor (visiting professor)	Profesor visitante	Profesor visitante (visiting professor) Profesor asociado a tiempo parcial (part-time teacher)	Proferssor Titular ou Professor visitante
Program(me)	All the courses that make up a degree	Programme pédagogique	Manifesto degli studi	Studienplan/ Lehrplan (list of courses which make up degree), Studiengang (Study programme)	Plan de estudios, Programa Educativo	Plan de estudios	Currículo
Quarter	A quarter of an academic year	Un demi-semestre	Trimestre	ein Viertel eines akademischen Jahres	Not used	Not used	bimestre

Term	CC2020 Definitions	French (Europe)	Italian	German	Spanish Latin America	Spanish Europe	Português
Semester	Half an academic year	Un semestre	Semestre	Semester	Semestre	Semestre (often known as “Cuatrimestre” because classes last for 4 months + one of examinations)	Semestre
Senior	USA term for a fourth year student	Un étudiant de quatrième année		Student/ Studentin im 4. Studi enjahr	Estudiante de cuarto semestre	Estudiante de cuarto curso	Veterano do quarto (if last year Formando)
Sophomore	USA term for a second year student	Un étudiant de deuxième année		Student/ Studentin im 2. Studi enjahr	Estudiante de segundo semestre	Estudiante de segundo curso	Veterano do segundo ano
Subject	Similar to a course but not always used at university level	Un sujet	Materia	Fach	Asignatura	Asignatura	Matéria
Technology	The application of scientific knowledge for practical purposes, especially in industry	Une technologie	Tecnologia	Technologie	Tecnología	Tecnología	Tecnologia
Trimester	One third of an academic year	Un trimestre (3m) Un semestre (5m)	quadrimestre	Trimester	Trimestre	Trimestre	Trimestre
Undergraduate	Studying towards a bachelor’s degree	La Licence (L1-L3)	non laureato	grundständiges Studium	Estudios de grado	Estudios de grado	Graduação

Appendix I: Sustainable Computing and Engineering Competence in China

China and its education ministry have embraced competency as an important element in the development of computing and engineering programs. Over the past few years, publications emerged surrounding the importance of competency in computing and engineering education. The Forum of Chinese Twenty-Experts on Computing Education, in which more than twenty senior professors on computing have engaged, has recently published its "Blue Book" [Blu1] to address the need for competency in university environments, particularly as it applies to computing and engineering education programs. The China Computer Federation also emphasized computing education for competencies in its *2018 Future Computer Education Summit (FCES 2018)* publication [Imp1]. The first of these publications addresses the need for program agility in response to a rapidly changing technological world. The remainder of this section summarizes the "Blue Book" philosophy and ways China expects to adapt to technological change over the next dozen years. The second of these publications uses a modern approach that competency is a triad of knowledge, skill, and disposition as explained earlier in this report.

I.1: Adaptable and Sustainable Competencies

Over 2017-2019, a working group of the mentioned forum has written a "Blue Book" titled, "Computing Education and Sustainable Competence" [Blu1]. The emergence of this work in China has opened new ideas in the transformation of university computing and engineering education in China. The emerging fields of information technology (IT) and artificial intelligence (AI) have created novel opportunities for industry and academia. The internet has made possible new modern services and businesses coupled with innovative applications. The emerging AI industry has provided fertile ground for new industrial sectors such as smart enterprises and public services. The new industrial revolution (i.e., Industry 4.0) promises advances in networked intelligent manufacturing, service-oriented manufacturing, and robotics for industry and modern services.

Change on such a global scale brings new challenges for an information society. Societal changes present challenges for a digitally networked cognitive society, for sustainable development of society and the environment, and the transference of information and knowledge. People also change. Younger generations have new attitudes and demands for professional development that require multi-dimensional approaches to learning with sustainable competencies so they can adapt to an evolving future. That means education must also change—especially at the university level.

Educational challenges facing students include global competitiveness, massive open online courses (MOOCs) including the disruption caused by them, changes in university functions, and educational reform as needed for an information society. These changes place a strain on learning systems and society. To address these challenges, it is important to adjust to changes in society by developing sustainable competencies for higher education. *Sustainable competency* refers to the ability to (a) adapt to change and competitiveness of the future society, (b) be creative based on the missions and technology, and (c) perform and promote social and technical development [Blu1 p7]. Trans-boundaries and the rapid changes of new economies require that computing and engineering talents have stronger sustainable competencies for the future.

In recent years, the Chinese government and universities have explored and practiced new reformations of contemporary innovations in higher engineering education. The Chinese Ministry of Education (MOE) has been developing "New Engineering" educational initiatives to cultivate people for the future development of new technologies and new economies. The MOE is promoting innovative and entrepreneurial education as well as university-industry collaborative educational projects to nurture students with more innovative attitudes and a new consciousness for creativity and practical ability. Many Chinese universities have participated in such engineering education reform. This practice has led to an ongoing, competency-oriented movement in higher education.

An example of this movement is the "open loop university" at Stanford University. Under the new concept, students would enter the "open-loop" multiple times throughout their professional life. Called *Stanford 2025*, this movement presents a new concept of learning. Its ethos involves open education, purpose learning, and incremental development. Its characteristic of paced education promises to transform four-year-systems into lifetime multi-year systems over three phases that include calibrating, elevating, and activating. This axel flipped, self-fulfilling approach becomes a continuing spiral of Knowledge ==> Ability, followed by Ability ==> New Knowledge. Such a purposeful way of learning instills professional development as a driving force for learning.

I.2: Agile Education for Sustainable Competencies

Faced with multiple objectives and individualized demands for human development, it became important to create a sustainable competency-oriented agile form of education. *Agile education* is an approach that combines theory, knowledge, ability, and human quality into a comprehensive education system [Blu1 p25]. It is a method to stimulate student interest by enhancing their potential creative ability and their ability to adapt to change. Agile education realizes multiple iterative rounds of knowledge, learning, and promotional ability. It encourages the rapid learning of theory, techniques, practice, and efficient coordinated education by multi-university and multi-domain educational resources.

The concept of agile education derived its inspiration from industrial concepts, agile manufacturing, and agile software development from the 1990s. New product development and manufacturing models had to adapt to the rapid change of new technologies, new products, and new demands. For example, agile manufacturing became an advanced manufacturing technology with models appearing in the late 1990s. This strategy integrates agile virtual enterprise alliances, advanced flexible production technologies, and high-quality workers to promote industrial competence. As another example, agile software development emerged as a new customer demand-oriented software strategy that applies iterative construction models and cyclic progression methods to promote development efficiency.

Because of past successes, agile education is becoming a massive, customized mode of education that emphasizes the cultivation of accurate outcomes and content teaching. It intends to decompose dramatically training ability, to restructure content concurrently, and to rearrange content iteratively through advanced education networks and resources from multi-universities and multi-sectors. Agile education seeks to develop accurate coordination procedures for teaching and learning and to strengthen students' abilities through exploratory, active, and gradual learning procedures. Methods used include fostering university-industry cooperation for co-education, promoting teamwork and interaction of interdisciplinary teachers and students, and integrating education at the university with society.

Sustainable competency-oriented agile education involves instilling sustainable competencies in students over a four-year education experience. The learning perspective includes fundamental courses and general education followed by technical and core courses, followed by interdisciplinary elective courses, and culminating with individual development. A curriculum often defines this learning perspective. The practice perspective includes industry visits, yearly projects, course projects and scientific competence, professional training, industry internship, and graduate (capstone) projects. The combination of learning and practice develops sustainable competencies in students, which is at the heart of agile education. Figure I.1 illustrates this concept. The practice of agile education derives from practical experience of the recent reform and innovation actions of world-wide higher computing and engineering education. Examples of such practices include the open-loop university, interconnection networked learning, MOOC-based online and offline learning, flipped classroom, concurrent learning and doing, project-based learning, university and industry cooperation, and innovation and entrepreneurship education.

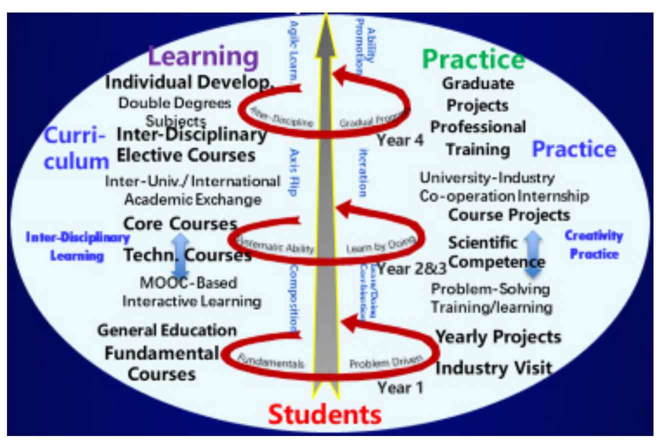


Figure I.1. Agile education and sustainable competencies
(*Courtesy of Prof. Xu Xiaofei*)

I.3: Factors Affecting Agile Computing and Engineering Education

In agile education, there is an emphasis on multiple cultivation objectives based on the diversity of students. Its purpose is to develop "education-on-demand" through a combination of a major-oriented program with individualized learning. Universities should provide massive, customized education systems through multiple cultivation objectives of their students. The teaching system would consist of a curriculum, teaching processes, teacher-student learning activities, resources, and quality evaluation methods. Universities would likely have to restructure their traditional education systems to establish multiple cultivation objectives, to construct a flexible composite curriculum, to develop iterative learning procedures, and to create a collaborative teaching support system. Considerations include multiple cultivation objectives, course and module classifications, systematic core courses, combining theory with practice, establishment of agile teaching and learning processes, and generating collaborative educational resources.

In agile education, universities would classify their curricula and courses into multi-clusters of modules according to the needs of individualized students or groups of specialties and directions to provide an environment of adaptive knowledge learning and ability training. Course modules include general education clusters and fundamentals clusters, specialty core course modules, interdisciplinary course modules, elective course modules, and experimental practice clusters. It might be necessary to make flexible compositions of the course modules, courses, or micro-courses to adapt the programs to student needs since they have more choices for their development. Because of emerging technologies and applications, it would be necessary to reconstruct the core courses in developing a systematic and flexible curriculum. The universities would have to redesign course content and teaching methods because of the interdisciplinary learning, the iterated cultivation and development of creativity, and the adaptability of the students.

Agile education is a multi-phase process of iterative learning. For computing programs, the *iterative learning* process begins with students entering a computing course to gain knowledge and to develop an ability for enhancement. The process includes:

- To know basic concepts of computers and their systems (year-1 courses and yearly projects)
- To understand and grasp systems, components, and techniques of computers (year-2 and year-3 courses as well as project-based learning)
- To design and develop systems and applications for computers (year-4 courses, projects, and internships)

Ultimately, students will gain further knowledge, technology, and a systematic understanding of computing through this three-round form of iterative learning and practice.

As already mentioned, agile education is a combination of theory and practice. The emphasis is to combine course-based teaching and project-based learning as well as to enhance the hands-on ability of students through wider and deeper practical activities. Figure I.2 captures that philosophy. In addition to knowledge transfer and skill training, it is important to create opportunities and environments for students to develop and mature. Creating such opportunities for students is particularly important. Therefore, the educational procedure for agile education consists of two significant points. First, the teaching and learning procedure should be very flexible and agile to allow students to arrange their learning plan and course selection based on their interests and characteristics. Second, teaching faculties should restructure the teaching and learning procedures as well as management rules to accommodate the needs of interdisciplinary learning, iterated student cultivation, and the individual development of students. Suggestions for doing this include a robust credit system, a flexible teaching plan, interdisciplinary studies, cumulative study and examinations, and credit for creative practice.

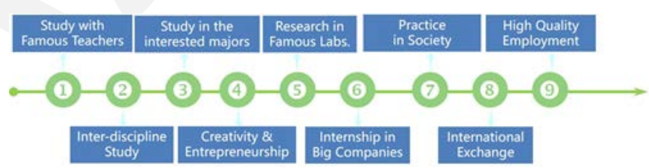


Figure I.2. Agile education: Combining theory with practice
(*Courtesy of Prof. Xu Xiaofei*)

Agile education requires a collaborative organization and resources. To implement agile education, it is necessary to coordinate the teaching and learning organization and resources, including multidisciplinary teams, educational facilities from multi-schools and multi-universities, as well as training resources from industry and society. Universities should establish virtual inter-school and inter-university collaborative teaching centers for agile education and purpose learning by combining teaching resources from multi-institutions. It is also important to consider MOOC resources through inter-university collaborative teaching and learning approaches.

I.4: Open Education Ecosystems for Agile Education

It is important to address the role of university management to implement agile education. Universities need to reform their management and support systems as well as their ecosystems for agile education. Focusing on individual students, small groups, and flexible learning are key elements for a successful transition. Universities should build an advanced agile education system, reform and restructure their management and support systems, and build an open education ecosystem for sustainable competencies. Suggestions for doing this include (a) setting up a flexible study-term and a full credit system for iterated learning and individualized cultivation, (b) establishing an undergraduate supervisor system and small group learning for individualized cultivation and development of students, (c) developing micro-courses and small course modules for flexible composition of learning contents, (d) developing ability-oriented courses and learning units as learning models, and (e) establishing a university-industry collaborative education system that includes internships, creative projects, and entrepreneurship.

It is also important to create support resources for agile education. This transformation to agile education requires abundant educational resources. These include advanced online and offline course resources, networked IT support platforms, laboratories for creative projects, entrepreneurship bases, student colleges, big data service platforms, and advanced infrastructure and facilities. In modern universities, they should build education support systems on IT-enabled network platforms to provide intelligent, coordinative, precise, and efficient services for agile education.

Quality assurance is an important part of agile education. Universities should create an ability-oriented agile education quality evaluation and assurance system to measure educational processes and results and to set up effective report mechanisms for improving teaching and learning quality. Analysis of teaching and learning status is useful for dynamic assessment of process evaluation, phase evaluation, and comprehensive evaluation for iterative learning and improvement. Quality metrics for agile education include key performance indicators (KPIs) for attainment (assessing development results of students' ability compared to education proposition and expectation), for process (assessing the quality of cultivation processes and key points), for cultivation bodies (assessing student quality and teacher quality, and for resource (assessing the investment for teaching and learning resources).

Agile education is conducive to open ecosystems for learning. Educational ecosystems encourage active promotion and constraint roles by engendering and developing an evolutionary education system. An open education ecosystem is a student-centric education system and environment that coordinates or integrates educational resources inside and outside a university. International resources are also possible through multi-channel collaborations for agile education and sustainable competencies development. A student-centric educational ecosystem can lead to interdisciplinary and comprehensive education, university-industry co-education, international joint education, creative and entrepreneurship education, and campus culture-based education. These in turn lead to agile education. Figure I.3 illustrates these findings.

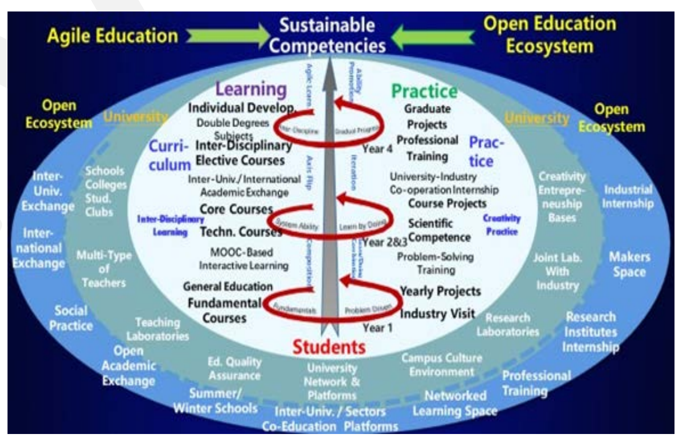


Figure I.3. Open Ecosystems for Agile Education
(*Courtesy of Prof. Xu Xiaofei*)

I.5: Service-oriented Computing Education

Service-oriented education is a natural outgrowth of agile education. To cultivate, ensure and enhance the sustainable competencies of students continuously in their entire professional life would become the important missions and educational service functions of universities in the future. Educational transformational trends suggest that:

Qualified Graduates ==> Student Lifetime Sustainable Competencies

Service-oriented education is a new form of student lifetime sustainable development education. It performs a multi-phased, interdisciplinary, ongoing, and adaptive education that provides continuous multi phases of agile education services for sustainable competencies during students' entire professional life. Fundamentally, service-oriented education leads to:

- Student lifetime centric sustainable development continuous education
- Individualized development purpose cultivation and learning
- Open trans-boundary and interdisciplinary co-education services
- Iterated multi-phased agile education and learning
- Professional online and offline education centers
- Smart education service networked platforms

Ultimately, the process leads to student lifetime sustainable competencies. Figure I.4 illustrates the promise of service-oriented education.

Figure I.4. Promise of service-oriented education
(*Courtesy of Prof. Xu Xiaofei*)

In conclusion, to cultivate innovative talent with sustainable competencies and adapt to the development of future emerging technologies and economies, it is important to reform and restructure current higher education systems, models, and ecosystems with new forms of engineering and education for sustainable competencies. As a new and advanced education form, agile education promises to improve higher computing and engineering education. These new forms of advanced education models and approaches (e.g., agile education, service-oriented education) will achieve realization with practice and exploration at universities throughout China and beyond.

I.6: References for Appendix I

[Blu1] "Blue Book" (2019) *The Blue Book Working Group of the Forum of Chinese Twenty-Experts on Computing Education in China* Computer Education for Sustainable Competence, China Higher Education Publishing, January 2019 (in Chinese).

[Imp1] Impagliazzo, John. (2018) The Role of Competency and the Future of Computer Education. *Communications of the China Computer Federation*; Volume 14, Number 9; September 2018. (in Chinese)

Appendix J: Contributors and Reviewers

The CC2020 Steering Committee and Task Force would like to acknowledge all contributors and reviewers who have provided valuable input, assistance, and feedback on this project.

First Name	Last Name	Affiliation	Country
At least 407	Anonymous	Various	Various
-	-	Jiangxi Normal University	China
Joerg	Abke	Technical University of Aschaffenburg	Germany
Shadi	Abou-Zahra	World Wide Web Consortium (W3C)	International
Ibrahim	Albluwi	Princeton University	United States
Hala	Alrumaih	Imam Mohammad Ibn Saud Islamic University	Saudi Arabia
Mohammed	Alshara	Information Technology Department, IMAM University - Riyadh	Saudi Arabia
Barbara	Anthony	Southwestern University	United States
Renata	Araujo	Brazilian Computer Society	Brazil
Jose Luis	ASencio Mera	Escuela Superior Politécnica del Litoral	Ecuador
Jeffrey	Babb	West Texas A&M University	United States of America
Olga	Bogoyavlenskaya	Petrozavodsk State University	Russia
Paolo	Bottoni	Univ. Roma “Sapienza”	Italy
Pierre	Bourque	École de technologie supérieure – Université du Québec	Canada
David	Bowers	The Open University, UK; The Institute of Coding, and The SFIA Council	UK
Premek	Brada	University of West Bohemia, Pilsen	Czech Republic
Kevin	Brunner	Graceland University	United States
Matthew	Burrows	-	UK
John	Calder	Manukau Institute of Technology	New Zealand
Héctor	Cancela	Universidad de la República	Uruguay
Alberto	Cannistraro	AICA member	Italy
Yongzhi	Cao	Peking University	China
F	Castro Lopes	Universidade Portucalense	Portugal
Kai H.	Chang	Auburn University	United States
Wenzhi	Chen	Zhejiang University	China
Juan	Chen	National University of Defense Technology	China
Juan	Chen	National University of Defense	China

		Technology	
Paolo	Ciancarini	University of Bologna	President of GRIN, the Italian association of informatics university professors)
Alison	Clear	Eastern Institute of Technology	New Zealand
Tony	Clear	Auckland University of Technology	New Zealand
Ernesto	Cuadros-Vargas	Latin American Center for Computing Studies (CLEI)	Peru
Alberto	Culatina	Università degli Studi di Pavia	Italia
Yafei	Dai	Peking University	China
Colin	de la Higuera	Univ. Nantes	Representing the Société informatique de France, the French association of Informatics university professors
Adrienne	Decker	University of Buffalo	United States of America
Jörg	Dese	Fernuniversität in Hagen	Representing Fakultätentag Informatik, German scientific society of university professors in informatics, and Gesellschaft für Informatik
Jörg	Desel	FernUniversität in Hagen	Germany
Juan Francisco	Díaz	Universidad del Valle	Colombia
Juan Manuel	Dodero	University of Cadiz	Spain
Tania Mara	Dors	Pontificia Universidade Catolica do Parana	Brazil
Dennis	Du	-	China
Eric	Durant	Milwaukee School of Engineering	United States of America
MJ	Escalona	University of Seville	Spain
Marisa	Exter	Purdue University	United States of America
Dick	Fairley	Software and Systems Engineering Associates	United States
Aaron	French	University of New Mexico	United States
Stephen	Frezza	Gannon University	United States of America
Shivanagowda	G M	Shri Dharmasthala Manjunatheshwara College of Engineering and Technology	India
Judith	Gal-Ezer	Open University	Israel
Kevin	Gary	Arizona State University	United States
Beatriz Florián	Gaviria	Universidad del Valle	Colombia
Markus	Geissler	Cosumnes River College	United States
Giorgio	Giacinto	University of Cagliari	Italy

Karina	Gibert	Intelligent Data Science and Artificial Intelligent Research Center and Barcelona School of Informatics (Universitat Politècnica de Catalunya) and Vicedean on Big Data, Data Science and Artificial Intelligence of the Illustrious Official Professional College of Informatics Engineering of Catalonia	Spain
Itana Maria	Gimenes	Brazilian Computer Society	Brazil
Steven	Gordon	Ohio State University	United States of America
Rohit	Goswami	University of Iceland	Iceland
Monique	Grandbastien	Univ. de Lorraine	France
Reyes	Grnagel Seguer	Universitat Jaume I	Spain
Mohammad Ali	Hammoudeh	-	-
Eiji	Hayashiguchi	Waseda University, CTO VJP Co. Ltd.	Japan
Thomas	Hilburn	Embry-Riddle Aeronautical University	United States
Javier	Hormigo	Universidad de Malaga	Spain
John	Impagliazzo	Hofstra University	United States of America
Nikolay	Kakanakov	Technical University of Sofia Plovdiv branch	Bulgaria
Katsuhiko	Kakehi	Waseda university	Japan
Amey	Karkare	India Institute of Technology – Kanpur	India
Uwe	Kastens	Univ. Paderborn	Germany
Aaron	Keen	Cal Poly	United States
Ajax	Kroos	UTEC	Peru
Johannes	Krugel	Technical University of Munich	Germany
Richard	Le Blanc	Seattle University	United States of America
Charles	Lee	-	China
Paul	Leidig	Grand Valley State University	United States of America
Wei	Li	School of Computer Science and Engineering, Xi'an University of Technology	Chinese
Wenlong	Liu	Vision Investment	China
David	Lopez	Universitat Politècnica de Catalunya	Spain
David	Lopez	Universitat Politècnica de Catalunya	Spain
Barry	Lunt	Brigham Young University	United States of America
Johannes	Magenheim	Univ. Paderborn	Germany
Linda	Marshall	University of Pretoria	South Africa
Massoud	Massoudi	Ph.D. Scholar at Delhi Technological University	Afghanistan

Bruce	McMillin	Missouri University of Science and Technology	United States of America
Tania	McVeety	IBM	United States of America
Nancy	Mead	Carnegie Mellon University	United States of America
Greg	Michaelson	School of Mathematical and Computer Sciences, Heriot Watt University	United Kingdom
Mattia	Monga	Università degli Studi di Milano, Departmente of Computer Science	Italy
Manuel	Mora	Autonomous University of Aguascalientes	Mexico
Lourdes	Moreno	Chair AIPO- Spanish Society for HCI	Spain
Enrico	Nardelli	Univ. Roma Tor Vergata	President of Informatics Europe, the European association of university departments and industrial research lab in Informatics
Vânia	Neris	Federal Universtity of São Carlos	Brazil
Salvatore	Orlando	Univ. Venezia	Italy
Allen	Parrish	University of Alabama	United States of America
Arnold	Pears	KTH Royal Institute of Technology	Sweden
Teresa	Pereira	Instituto Politécnico de Viana Castelo	Portugal
Domenick	Pinto	Sacred Heart University	United States
Melinda	Reno	Deloitte Consulting	United States of America
RITSI	Reunión de Estudiantes de Ingenierías Técnicas y Superiores de Informática	-	Spain
Ulises	Roman Concha	UNMSM	Peru
Ariel	Sabiguero	Universidad de la República	Uruguay
Fermin	Sanchez	Universitat Politècnica de Catalunya	Spain
Nello	Scarabottolo	Università di Milano	Italy
Yann	Secq	University of Lille	France
Ian	Seward	SFIA Foundation	International
Nicholas	Sheppard	Western Sydney University	Australia
Williamson	Silva	UNESPAR	Brazil
	Simon	University of Newcastle	Australia
YANCHUN	SUN	Peking University	China
Shingo	Takada	Keio University	Japan
Gabriel	Tamura	Universidad Icesi	Colombia
Chris	Taylor	Milwaukee School of Engineering	United States

JohnBarrie	Thompson	None - retired academic and UK National Teaching Fellow	United Kingdom
Ye	Tian	ByteDance	China
Steve	Tockey	Construx Software	USA
Heikki	Topi	Bentley University	United States of America
Paul	Tymann	Rochester Institute of Technology	United States of America
Jim	Vallino	Rochester Institute of Technology	United States
Gerrit	van der Veer	Vrije Universiteit	Netherlands
Eduardo	Vendrell Vidal	Univ. Valencia	Representing SCIE and CODDII, Spanish scientific societies of university professors in Informatics
Abhijat	Vichare	ACM India	India
Norha M.	Villegas	Universidad Icesi	Colombia
Barbara	Viola	Viotech Solutions	United States of America
Les	Waguespack	Bentley University	United States of America
Pearl	Wang	George Mason University	United States of America
Rupert	Ward	University of Huddersfield	United Kingdom
Ed	Weber	Millikin University	USA
Ning	Wu	-	China
Xi	Wu	Chengdu Univ. of Information Technology	China
Zichen	Xu	The Nanchang University	China
Xiaofei	Xu	Harbin Institute of Technology (HIT)	China
Jane (Xiaochun)	Yang	AchieveFun Info Technology	China
Ge	Yu	Northeastern University	China
Armita	Zarnegar	Swinburne University of Technology	Australia
Ming	Zhang	Peking University	China
Stuart	Zweben	Ohio State University	United States of America

Acknowledgements

The CC2020 project acknowledges organizations and people who have contributed to this important effort. The project acknowledges the Association for Computing Machinery (ACM), the IEEE Computer Society (IEEE-CS), the Association for Information Systems (AIS), the Education Special Interest Group of Information Systems and Computing Academic Professionals (EDSIG/ISCAP), and ACM SIGCHI. It also acknowledges project collaborators such as the Information Processing Society of Japan (IPSJ), the Chinese Computing Federation (CCF), the Latin American Center on Computing [EL Centro Latinoamericano de Estudios en Informática (CLEI)], ACM India, GRIN (Italian Association of Computer Scientists) ACM Europe, Informatics Europe, I4All, and the West Texas A&M University.

The CC2020 Project would like to thank the National Strategic Planning and Analysis Research Center (NSPARC) at Mississippi State University for its support of the Web development for this project, as well as its support of some of the publication costs associated with this project.

The CC2020 project also acknowledges the many people who participated in this project including the CC2020 task force and its steering committee as well as those listed in Appendix J who have contributed content to the project and volunteered their time to review the CC2020 report as it progressed.