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 data science. The study 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.*

It is expected that this National Academies report will generate a profound influence on the global development of data engineering and data science 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 strongly 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 the 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. Fein [Fei1] continues:

> *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.*

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.

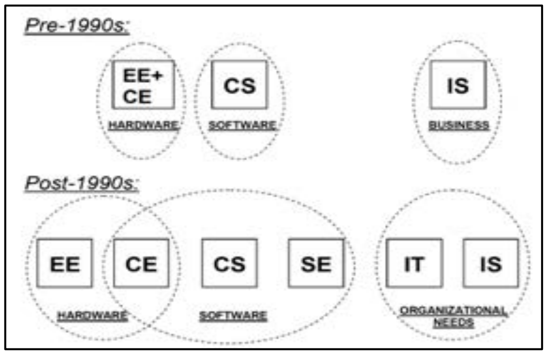

Figure 2.1 Computing disciplines compared, from CC2005

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.

[* Note: Cp.E. at CCNY includes a lot more software engineering & CS theory than is described below.]

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 noticeably 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 tightly 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) seeking 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 p3.2]. 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 Report 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 common resource, security, and concurrency as general principles. These principles are strongly linked to proficiency in programming and software development which are especially 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 (CSEC) 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 (e.g., cryptography, network defense) to managerial (e.g., 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 (IS2020) to be available in 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 (IT) 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 (SE) 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. The strong focus of software engineering is 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 current 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 two things—that clear differences exist between the computing disciplines and that they all have distinguishing characteristics that are essential for their individual identities. 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.

[* Note: Cp.E. at CCNY includes a lot more software & CS theory than is shown below.]

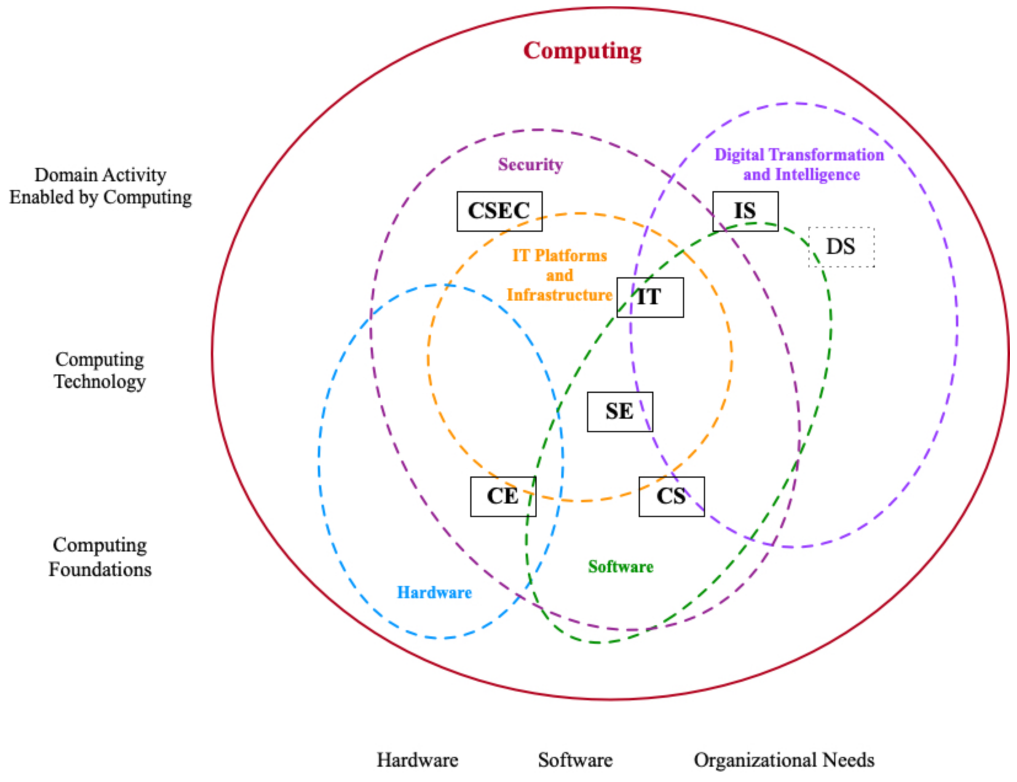

Figure 2.2. A contemporary view of the landscape of computing education
Legend: Curricular reports: CE=computer engineering; CS=computer science; CSEC=cybersecurity; IS=information systems; IT=information technology; SE=software engineering; DS=data science (under development).

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 better achieve their goals. 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 their 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 8.1.1 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 as "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, the computing area of 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. Archaeology uses many aspects of computing to understand where to find and how to study remains, presenting yet another example.

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).

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

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 of a simple computer design for a local engineering company.
B. Evaluate the timing diagram behavior of a simple processor-implemented at the logic circuit level and develop a report expressing the findings.
C. Write a simple program at the assembly/machine level for string processing and manipulation and for converting numerical data into hexadecimal form.
D. Implement a fundamental high-level construct in both machine and assembly languages and present the results to a group of peers.
E. 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 some 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 for 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 some 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. Build 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. 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.

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

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

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

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

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

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

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

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

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

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

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.
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).
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, considering 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 in 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

Table D.1 Representative Summary of Computing Knowledge Areas

Categorization	**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 an exceedingly 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 [Liu1,Sta2,Sta3]. See Table D.2. This ordering offers an ordered arrangement of elements for locating in a cartesian space.