Lecture 1 - What is Software Engineering?

SE 2217 - Principles of Software Engineering

Material from : Computing Curricula 2005 , SE 2004 and SE 2014 reports from http://www.acm.org/education/curricula-recommendations

“Our civilization is built on software”

Bjarne Stroustrup

Definitions of Software Engineering

- "The establishment and use of sound engineering principles (methods) in order to obtain economically software that is reliable and works on real machines" [Bauer 1972]

Definitions of Software Engineering

- "Software engineering is that form of engineering that applies the principles of computer science and mathematics to achieving cost-effective solutions to software problems." [CMU/SEI-90-TR-003]

Definitions of Software Engineering

- "The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software" [IEEE 1990]

Software Engineering

- In the workplace, the term software engineer is a job label
  - There is no standard definition for this term when used in a job description
  - It can be a title equivalent to computer programmer or a title for someone who manages a large, complex, and/or safety-critical software project

Software Engineering

- Software engineering had emerged as an area within computer science
- As computing is used to attack a wider range of complex problems, creating reliable software becomes more difficult

Software Engineering

- With large, complex programs, no one person can understand the entire program, and various parts of the program can interact in unpredictable ways
- Computing is also used in safety-critical tasks where a single bug can cause injury or death

Software Engineering

- Over time, it became clear that producing good software is very difficult, very expensive, and very necessary

Engineering

- Engineering is "the application of scientific principles to practical ends; as the design, construction of efficient and economical structures, equipment and systems

http://en.wikipedia.org/wiki/Enginering

Engineer

- An engineer is a professional practitioner of engineering, concerned with applying scientific knowledge, mathematics, and ingenuity to develop solutions for technical problems

http://en.wikipedia.org/wiki/Enginer

Engineer

- "An engineer can do for a dollar what any fool can do for two" (Arthur Mellen Wellington, an American civil engineer)

http://en.wikipedia.org/wiki/Arthur_Mellen_Wellington

Characteristics of Engineering

- Engineers proceed by making a series of decisions, carefully evaluating options, and choosing an approach at each decision point that is appropriate for the current task in the current context
  - Appropriateness can be judged by tradeoff analysis, which balances costs against benefits

Characteristics of Engineering

- Engineers measure things, and when appropriate, work quantitatively; they calibrate and validate their measurements; and they use approximations based on experience and empirical data

Characteristics of Engineering

- Engineers emphasize the use of a disciplined process when creating a design and can operate effectively as part of a team in doing so

Characteristics of Engineering

- Engineers can have multiple roles: research, development, design, production, testing, construction, operations, management, and others such as sales, consulting, and teaching

Characteristics of Engineering

- Engineers use tools to apply processes systematically
- Therefore, the choice and use of appropriate tools is key to engineering

Characteristics of Engineering

- Engineers, via their professional societies, advance by the development and validation of principles, standards, and best practices
- Engineers reuse designs and design artifacts

The Discipline of Software Engineering

The Discipline of Software Engineering

- Software products help us to be more efficient and productive
- They make us more effective problem solvers, and they provide us with an environment for work and play that is often safer, more flexible, and less confining

The Discipline of Software Engineering

- Despite these successes, there are serious problems in the cost, timeliness, and quality of many software products

Reasons for Problems in Software Engineering

- Software products are among the most complex of man-made systems, and software by its very nature has intrinsic, essential properties (e.g., complexity, invisibility, and changeability) that are not easily addressed [Brooks 95]

The Discipline of Software Engineering

- Programming techniques and processes that worked effectively for an individual or a small team to develop modest-sized programs do not scale-up well to the development of large, complex systems (i.e., systems with millions of lines of code, requiring years of work, by hundreds of software developers)

The Discipline of Software Engineering

- The pace of change in computer and software technology drives the demand for new and evolved software products
  - This situation has created customer expectations and competitive forces that strain our ability to produce quality of software within acceptable development schedules

The Discipline of Software Engineering

- The availability of qualified software engineers has not kept pace with the demand from industry, so that systems are designed and built by people with insufficient educational background or experience

History of Software Engineering

Software Engineering

- In the 1960s, a software product was usually created as a single, monolithic entity, executed on a computer with the support of a fairly basic operating system

Software Engineering

- Such a product had external operations that were mainly confined to basic file input/output

Software Engineering

- In contrast, a software system developed in today may well reuse major components of other systems, execute on multiple machines and platforms, and interact with other, runs on globally distributed systems

Software Engineering - History

- This led to the creation of software engineering, a term that emanated from a NATO sponsored conference held in Garmisch, Germany in 1968

Software Engineering - History

- In the U.S. it was not until the 1990s that one could reasonably expect to find software engineering as a significant component of computer science study at many institutions

Software Engineering - History

- While computer science (like other sciences) focuses on creating new knowledge, software engineering (like other engineering disciplines) focuses on rigorous methods for designing and building things that reliably do what they're supposed to do

Software Engineering - History

- Software engineering began to develop as a discipline unto itself
- Originally the term software engineering was introduced to reflect the application of traditional ideas from engineering to the problems of building software

Software Engineering - History

- In addition to its computer science foundations, software engineering also involves human processes that, by their nature, are harder to formalize than are the logical abstractions of computer science

Software Engineering - History

- Experience with software engineering courses within computer science curricula showed many that such courses can teach students about the field of software engineering but usually do not succeed at teaching them how to be software engineers

Software Engineering - History

- Many experts concluded that the latter goal requires a range of coursework and applied project experience that goes beyond what can be added to a computer science curriculum

Software Engineering - History

- Degree programs in software engineering emerged in the United Kingdom and Australia during the 1980s, but these programs were in the vanguard

Software Engineering - History

- In the United States, degree programs in software engineering, designed to provide a more thorough foundation than can be provided within computer science curricula, began to emerge during the 1990s

Computing Disciplines

Computing

- “We can define computing to mean any goal-oriented activity requiring, benefiting from, or creating computers”

Computing

- Computing includes designing and building hardware and software systems for a wide range of purposes;
  - Processing, structuring, and managing various kinds of information;
  - Doing scientific studies using computers;
  - Making computer systems behave intelligently;
  - Creating and using communications and entertainment media;
  - Finding and gathering information relevant to any particular purpose,
  - and so on…

Computing Disciplines

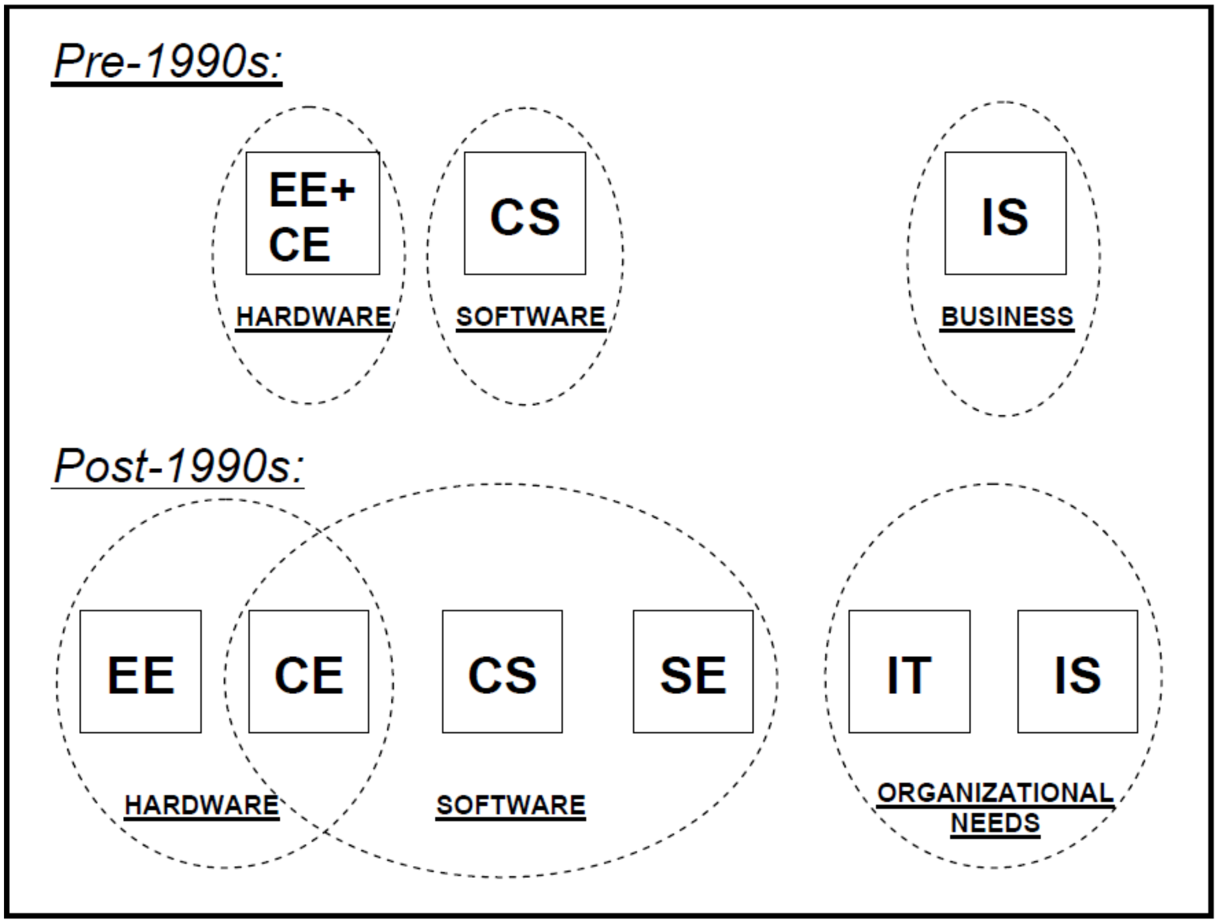


Figure 2.1. Harder Choices: How the Disciplines Might Appear to Prospective Students

Descriptions of the Major Computing Disciplines

- Computer Engineering
  - Concerned with the design and construction of computers and computer-based systems
  - It involves the study of hardware, software, communications, and the interaction among them

Descriptions of the Major Computing Disciplines

- Computer Engineering
  - Its curriculum focuses on the theories, principles, and practices of traditional electrical engineering and mathematics and applies them to the problems of designing computers and computer-based devices.

Descriptions of the Major Computing Disciplines

- Computer Science
  - Computer science spans a wide range, from its theoretical and algorithmic foundations to cutting-edge developments in robotics, computer vision, intelligent systems, bioinformatics, and other exciting areas

Descriptions of the Major Computing Disciplines

- Computer Science
  - We can think of the work of computer scientists as falling into three categories
    - They design and implement software
    - They devise new ways to use computers
    - They develop effective ways to solve computing problems

Descriptions of the Major Computing Disciplines

- Information Systems
  - Information systems specialists focus on integrating information technology solutions and business processes to meet the information needs of businesses and other enterprises, enabling them to achieve their objectives in an effective, efficient way

Descriptions of the Major Computing Disciplines

- Information Technology
  - Information technology is a label that has two meanings
    - In the broadest sense, the term information technology is often used to refer to all of computing
    - In academia, it refers to undergraduate degree programs that prepare students to meet the computer technology needs of business, government, healthcare, schools, and other kinds of organizations

Descriptions of the Major Computing Disciplines

- Software Engineering
  - Software engineering is the discipline of developing and maintaining software systems that behave reliably and efficiently, are affordable to develop and maintain, and satisfy all the requirements that customers have defined for them

Graphical Views of the Computing Disciplines

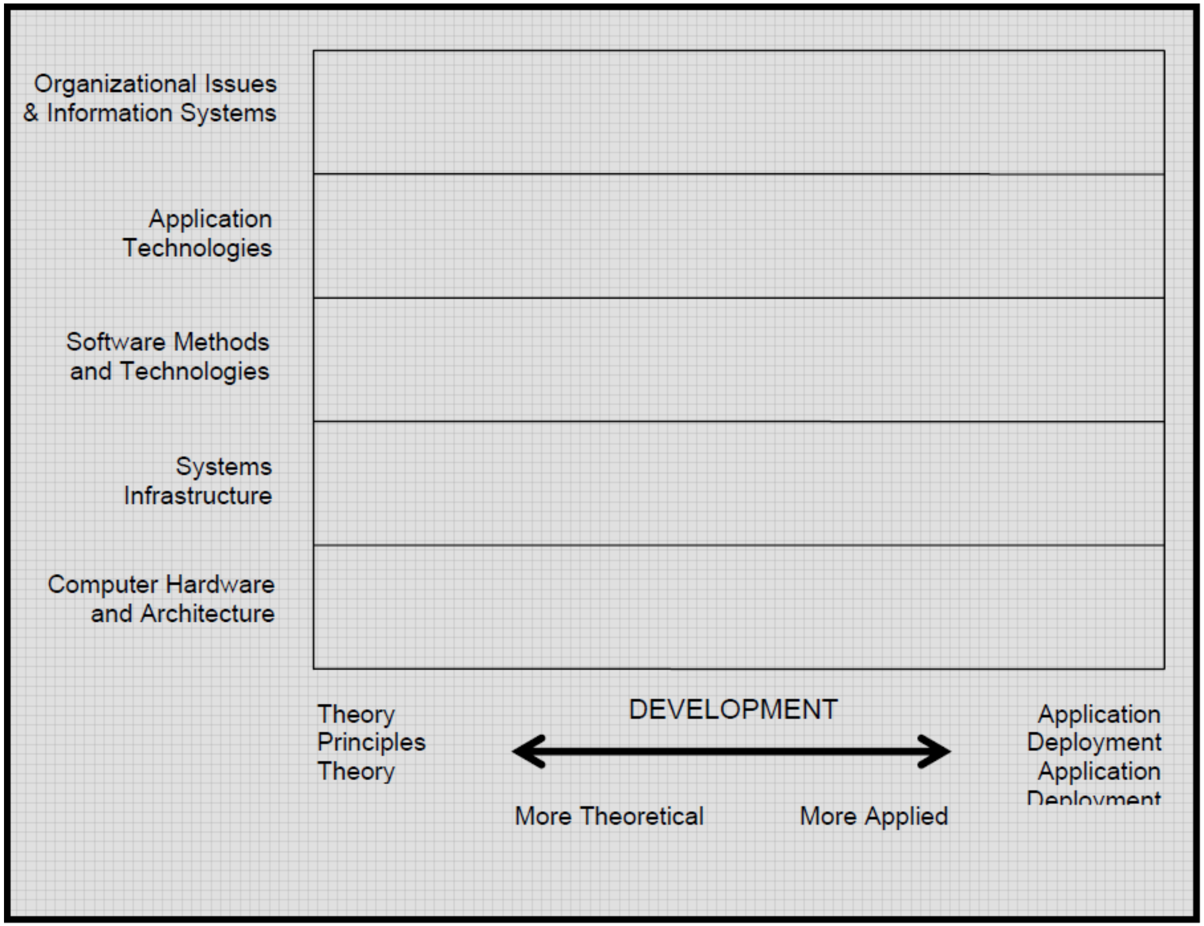
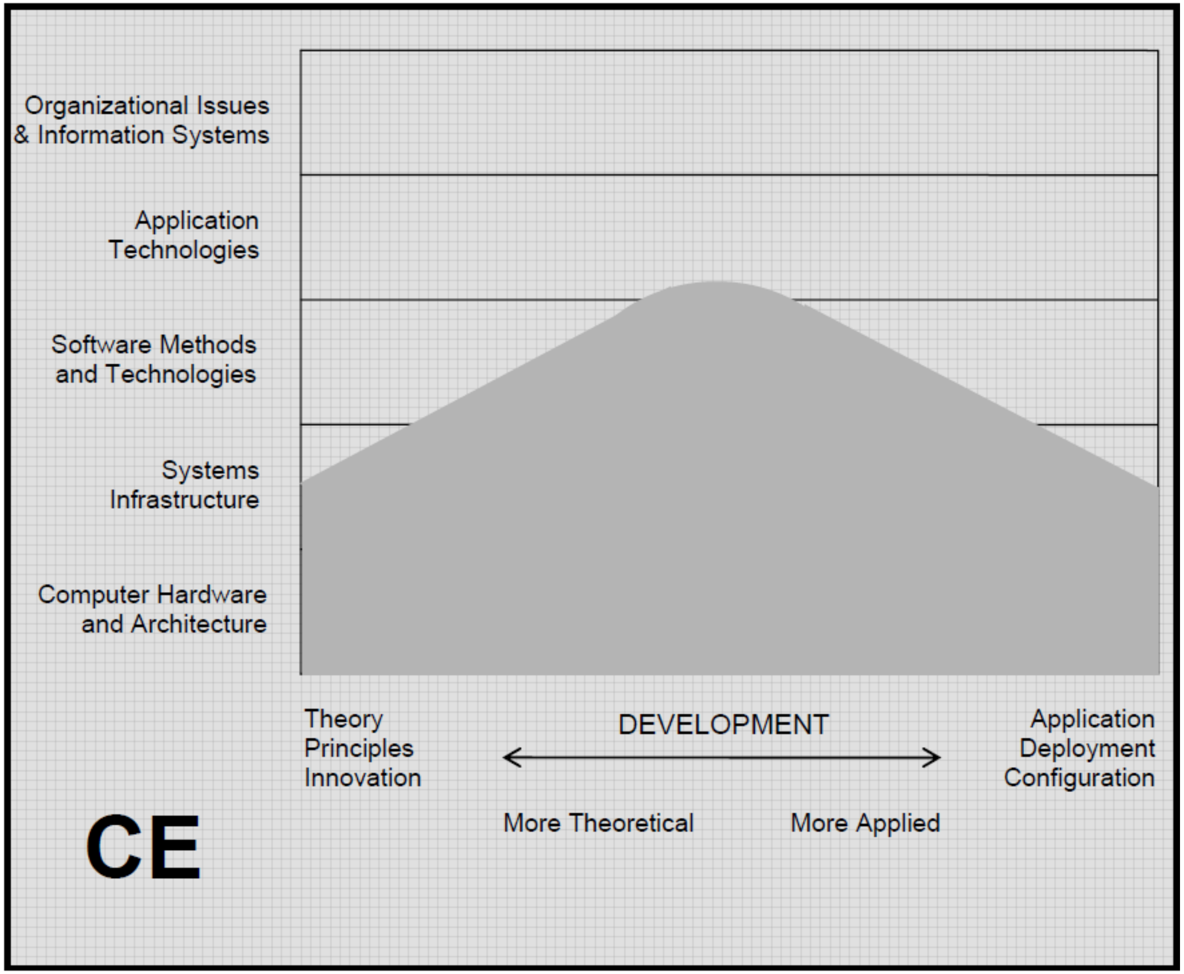
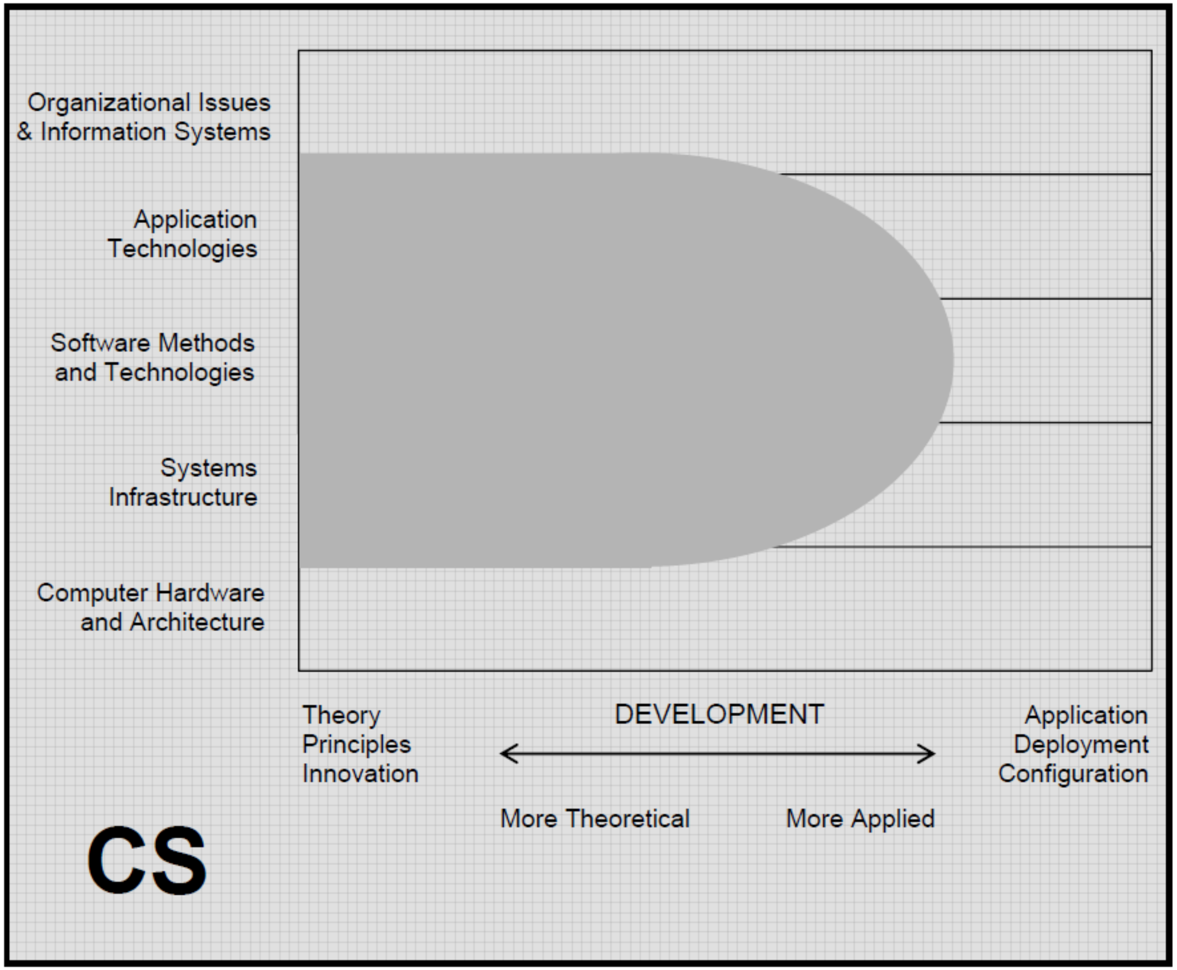


Figure 2.2. The Problem Space of Computing

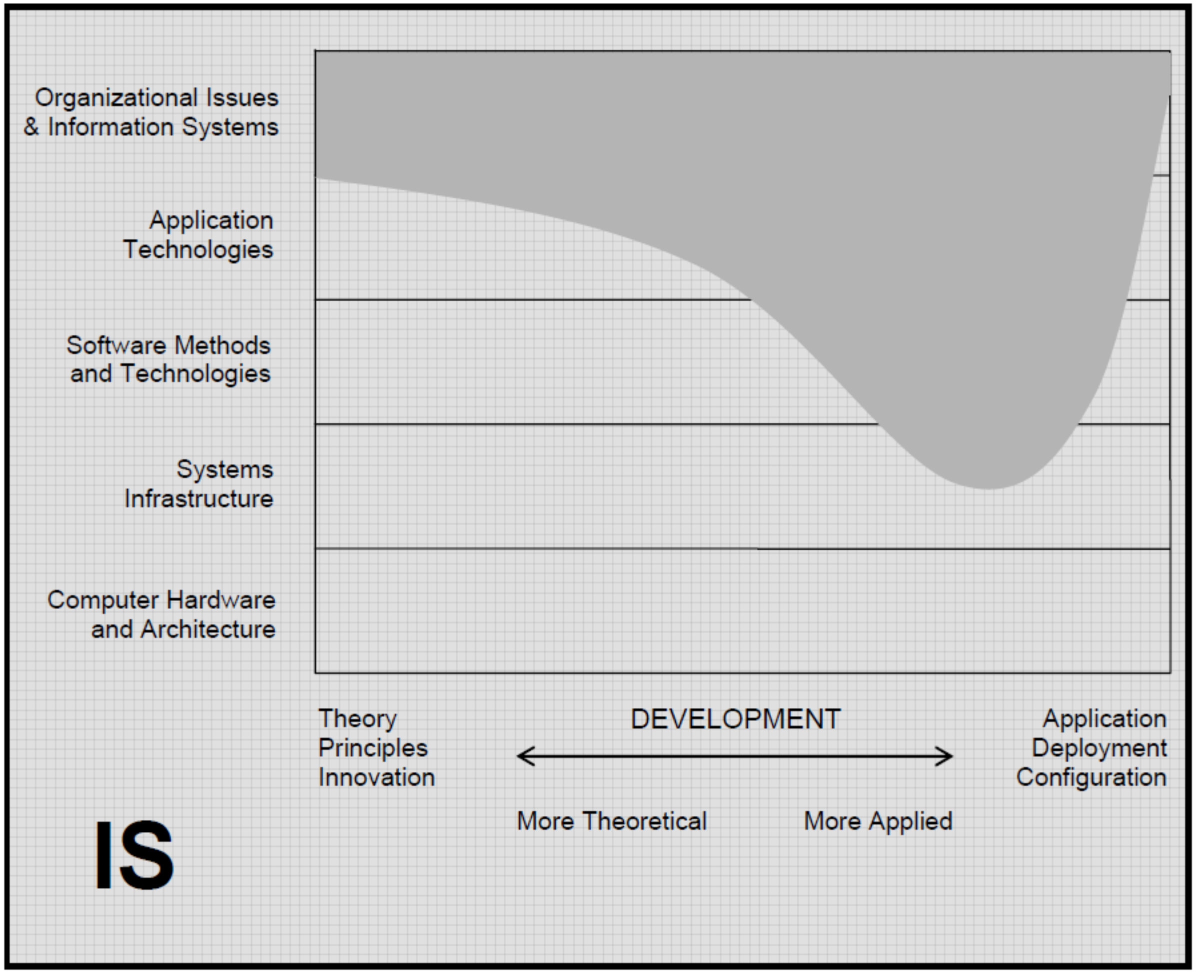
Computer Engineering



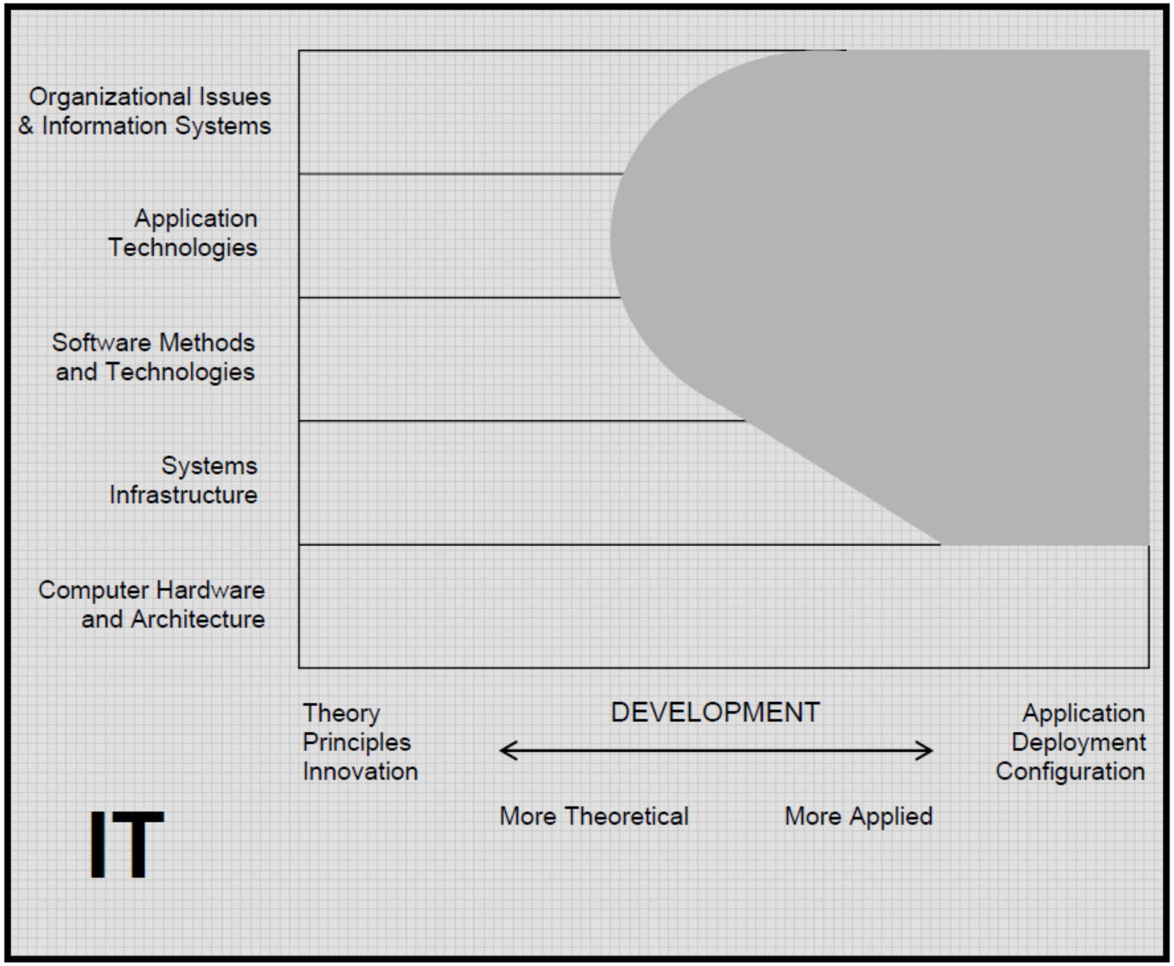
Computer Science



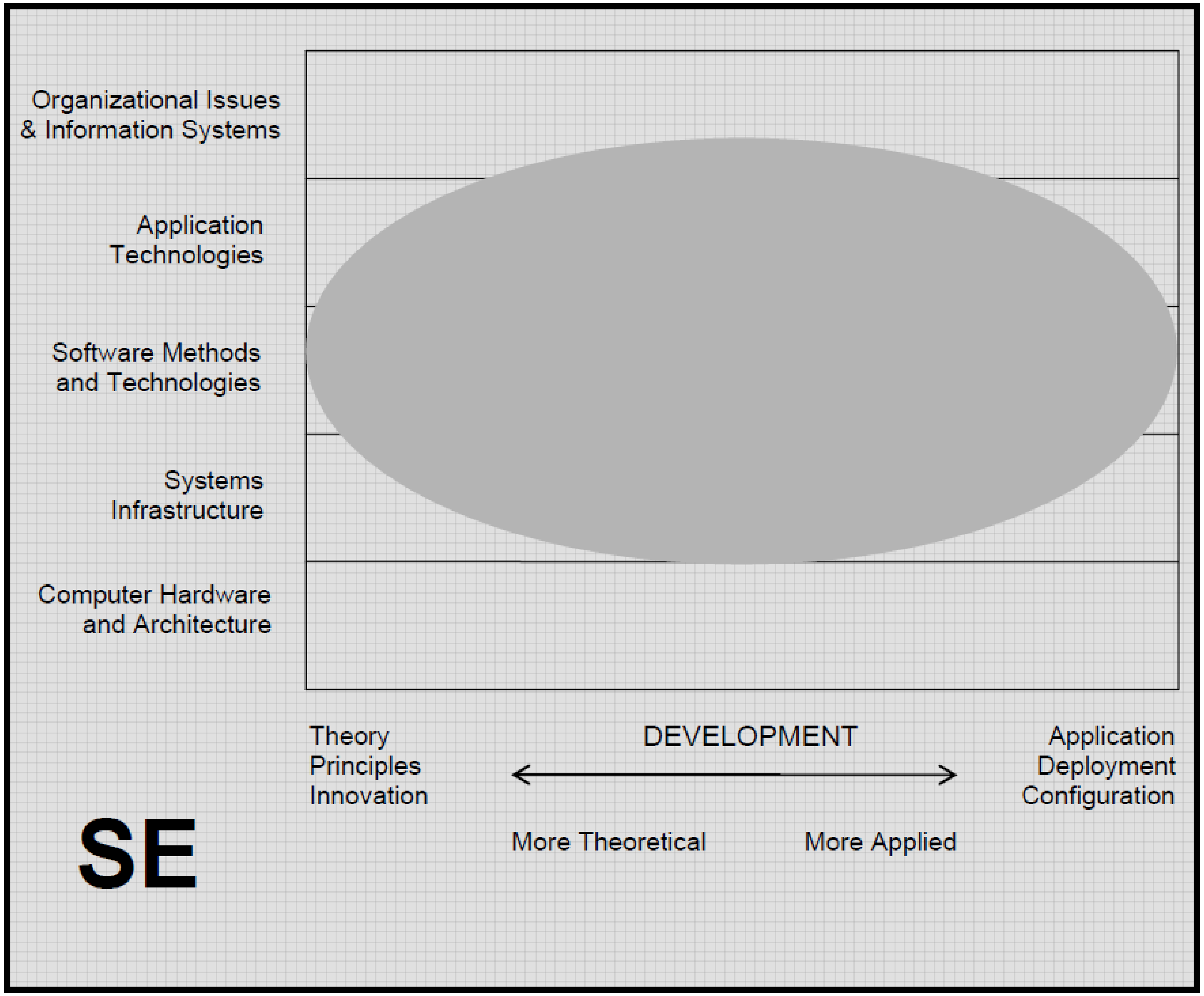
Information Systems



Information Technology



Software Engineering



Knowledge Areas for Computing Degree Programs

Comparative weight of computing topics across the five kinds of degree programs

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

Comparative weight of non-computing topics across the five kinds of degree programs

Knowledge Area	CE		CS		IS		IT		SE	
	min	max	min	max	min	max	min	max	min	max
Organizational Theory	0	0	0	0	1	4	1	2	0	0
Decision Theory	0	0	0	0	3	3	0	1	0	0
Organizational Behavior	0	0	0	0	3	5	1	2	0	0
Organizational Change Management	0	0	0	0	2	2	1	2	0	0
General Systems Theory	0	0	0	0	2	2	1	2	0	0
Risk Management (Project, safety risk)	2	4	1	1	2	3	1	4	2	4
Project Management	2	4	1	2	3	5	2	3	4	5
Business Models	0	0	0	0	4	5	0	0	0	0
Functional Business Areas	0	0	0	0	4	5	0	0	0	0
Evaluation of Business Performance	0	0	0	0	4	5	0	0	0	0
Circuits and Systems	5	5	0	2	0	0	0	1	0	0
Electronics	5	5	0	0	0	0	0	1	0	0
Digital Signal Processing	3	5	0	2	0	0	0	0	0	2
VLSI design	2	5	0	1	0	0	0	0	0	1
HW testing and fault tolerance	3	5	0	0	0	0	0	2	0	0
Mathematical foundations	4	5	4	5	2	4	2	4	3	5
Interpersonal communication	3	4	1	4	3	5	3	4	3	4

Comparing Expectations of Program Graduates

- **Computer engineers** should be able to design and implement systems that involve the integration of software and hardware devices

Comparing Expectations of Program Graduates

- **Computer scientists** should be prepared to work in a broad range of positions involving tasks from theoretical work to software development

Comparing Expectations of Program Graduates

- **Information systems specialists** should be able to analyze information requirements and business processes and be able specify and design systems that are aligned with organizational goals

Comparing Expectations of Program Graduates

- **Information technology professionals** should be able to work effectively at planning, implementation, configuration, and maintenance of an organization’s computing infrastructure

Comparing Expectations of Program Graduates

- **Software engineers** should be able to properly perform and manage activities at every stage of the life cycle of large-scale software systems

Graduates of an Undergraduate SE Program Must Be Able to

- Show mastery of the software engineering knowledge and skills, and professional issues necessary to begin practice as a software engineer
- Work as an individual and as part of a team to develop and deliver quality software artifacts

Graduates of an Undergraduate SE Program Must Be Able to

- Reconcile conflicting project objectives, finding acceptable compromises within limitations of cost, time, knowledge, existing systems, and organizations

Graduates of an Undergraduate SE Program Must Be Able to

- Design appropriate solutions in one or more application domains using software engineering approaches that integrate ethical, social, legal, and economic concerns

Graduates of an Undergraduate SE Program Must Be Able to

- Demonstrate an understanding of and apply current theories, models, and techniques that provide a basis for problem identification and analysis, software design, development, implementation, verification, and documentation

Graduates of an Undergraduate SE Program Must Be Able to

- Demonstrate an understanding and appreciation for the importance of negotiation, effective work habits, leadership, and good communication with stakeholders in a typical software development environment

Graduates of an Undergraduate SE Program Must Be Able to

- Learn new models, techniques, and technologies as they emerge and appreciate the necessity of such continuing professional development

SEEK Knowledge Areas and Knowledge Units (SE2004)

KA/KU	Title	hrs	KA/KU	Title	hrs
CMP	**Computing Essentials**	**172**	**VAV**	**Software V & V**	**42**
CMP.cf	Computer Science foundations	140	VAV.fnd	V&V terminology and foundations	5
CMP.ct	Construction technologies	20	VAV.rev	Reviews	6
CMP.tl	Construction tools	4	VAV.tst	Testing	21
CMP.fm	Formal construction methods	8	VAV.hct	Human computer UI testing and evaluation	6
			VAV.par	Problem analysis and reporting	4
FND	**Mathematical & Engineering Fundamentals**	**89**	**EVL**	**Software Evolution**	**10**
FND.mf	Mathematical foundations	56	EVO.pro	Evolution processes	6
FND.ef	Engineering foundations for software	23	EVO.ac	Evolution activities	4
FND.ec	Engineering economics for software	10			
PRF	**Professional Practice**	**35**	**PRO**	**Software Process**	**13**
PRF.psy	Group dynamics / psychology	5	PRO.con	Process concepts	3
PRF.com	Communications skills (specific to SE)	10	PRO.imp	Process implementation	10
PRF.pr	Professionalism	20			
MAA	**Software Modeling & Analysis**	**53**	**QUA**	**Software Quality**	**16**
MAA.md	Modeling foundations	19	QUA.cc	Software quality concepts and culture	2
MAA.tm	Types of models	12	QUA.std	Software quality standards	2
MAA.af	Analysis fundamentals	6	QUA.pro	Software quality processes	4
MAA.rfd	Requirements fundamentals	3	QUA.pca	Process assurance	4
MAA.er	Eliciting requirements	4	QUA.pda	Product assurance	4
MAA.rsd	Requirements specification & documentation	6			
MAA.rv	Requirements validation	3			
DES	**Software Design**	**45**	**MGT**	**Software Management**	**19**
DES.con	Design concepts	3	MGT.con	Management concepts	2
DES.str	Design strategies	6	MGT.pp	Project planning	6
DES.ar	Architectural design	9	MGT.per	Project personnel and organization	2
DES.hci	Human computer interface design	12	MGT.ctl	Project control	4
DES.dd	Detailed design	12	MGT.cm	Software configuration management	5
DES.ste	Design support tools and evaluation	3			

SEEK Knowledge Areas and Knowledge Units (SE2014)

KA/KU	Title	Hours		KA/KU	Title	Hours
CMP	**Computing essentials**	**152**		**DES**	**Software design**	**48**
CMP.cf	Computer science foundations	120		DES.con	Design concepts	3
CMP.ct	Construction technologies	20		DES.str	Design strategies	6
CMP.tl	Construction tools	12		DES.ar	Architectural design	12
				DES.hci	Human-computer interaction design	10
				DES.dd	Detailed design	14
				DES.ev	Design evaluation	3
FND	**Mathematical and engineering fundamentals**	**80**		**VAV**	**Software verification and validation**	**37**
FND.mf	Mathematical foundations	50		VAV.fnd	V&V terminology and foundations	5
FND.ef	Engineering foundations for software	22		VAV.rev	Reviews and static analysis	9
FND.ec	Engineering economics for software	8		VAV.tst	Testing	18
				VAV.par	Problem analysis and reporting	5
PRF	**Professional practice**	**29**		**PRO**	**Software process**	**33**
PRF.psy	Group dynamics and psychology	8		PRO.con	Process concepts	3
PRF.com	Communications skills (specific to SE)	15		PRO.imp	Process implementation	8
PRF.pr	Professionalism	6		PRO.pp	Project planning and tracking	8
				PRO.cm	Software configuration management	6
				PRO.evo	Evolution processes and activities	8
MAA	**Software modeling and analysis**	**28**		**QUA**	**Software quality**	**10**
MAA.md	Modeling foundations	8		QUA.cc	Software quality concepts and culture	2
MAA.tm	Types of models	12		QUA.pca	Process assurance	4
MAA.af	Analysis fundamentals	8		QUA.pda	Product assurance	4
REQ	**Requirements analysis and specification**	**30**		**SEC**	**Security**	**20**
REQ.rfd	Requirements fundamentals	6		SEC.sfd	Security fundamentals	4
REQ.er	Eliciting requirements	10		SEC.net	Computer and network security	8
REQ.rsd	Requirements specification and documentation	10		SEC.dev	Developing secure software	8
REQ.rv	Requirements validation	4				