Computer Science

Ethics, Professionals & More

何明昕 HE Mingxin, Max

Send your email to c.max@yeah.net with a subject like: *SE345-Andy: On What …*

Download from c.program@yeah.net /文件中心/网盘/SoftwareEngineering24s

Self-Evaluation Conformation

Send an email to the instructor (c.max) before June 22, in which list your course attendance (day by day) and all your submits for the SE course with basic email information, including contents, when, from email, to email … and give a self-evaluation mark in 100 for your participation performance on this course.

The Software Engineering Code of Ethics and Professional Practice

Software Engineering Code of Ethics and Professional Practice (Version 5.2) as recommended by the ACM/IEEE-CS Joint Task Force on Software Engineering Ethics and Professional Practices and jointly approved by the ACM and the IEEE-CS as the standard for teaching and practicing software engineering.

https://ethics.acm.org/code-of-ethics/software-engineering-code/

Software Engineering Code of Ethics and Professional Practice (Short Version)

PREAMBLE

The short version of the code summarizes aspirations at a high level of the abstraction; the clauses that are included in the full version give examples and details of how these aspirations change the way we act as software engineering professionals. Without the aspirations, the details can become legalistic and tedious; without the details, the aspirations can become high sounding but empty; together, the aspirations and the details form a cohesive code.

Software engineers shall commit themselves to making the analysis, specification, design, development, testing and maintenance of software a beneficial and respected profession. In accordance with their commitment to the health, safety and welfare of the public, software engineers shall adhere to the following Eight Principles:

1. PUBLIC – Software engineers shall act consistently with the public interest.

2. CLIENT AND EMPLOYER – Software engineers shall act in a manner that is in the best interests of their client and employer consistent with the public interest.

3. PRODUCT – Software engineers shall ensure that their products and related modifications meet the highest professional standards possible.

4. JUDGMENT – Software engineers shall maintain integrity and independence in their professional judgment.

5. MANAGEMENT – Software engineering managers and leaders shall subscribe to and promote an ethical approach to the management of software development and maintenance.

6. PROFESSION – Software engineers shall advance the integrity and reputation of the profession consistent with the public interest.

7. COLLEAGUES – Software engineers shall be fair to and supportive of their colleagues.

8. SELF – Software engineers shall participate in lifelong learning regarding the practice of their profession and shall promote an ethical approach to the practice of the profession.

A Computing Curricula Series Report
2020 December 31

Computing Curricula 2020

CC2020

Paradigms for Global Computing Education

encompassing undergraduate programs in

Computer Engineering

Computer Science

Cybersecurity

Information Systems

Information Technology

Software Engineering

with data science

Copyr

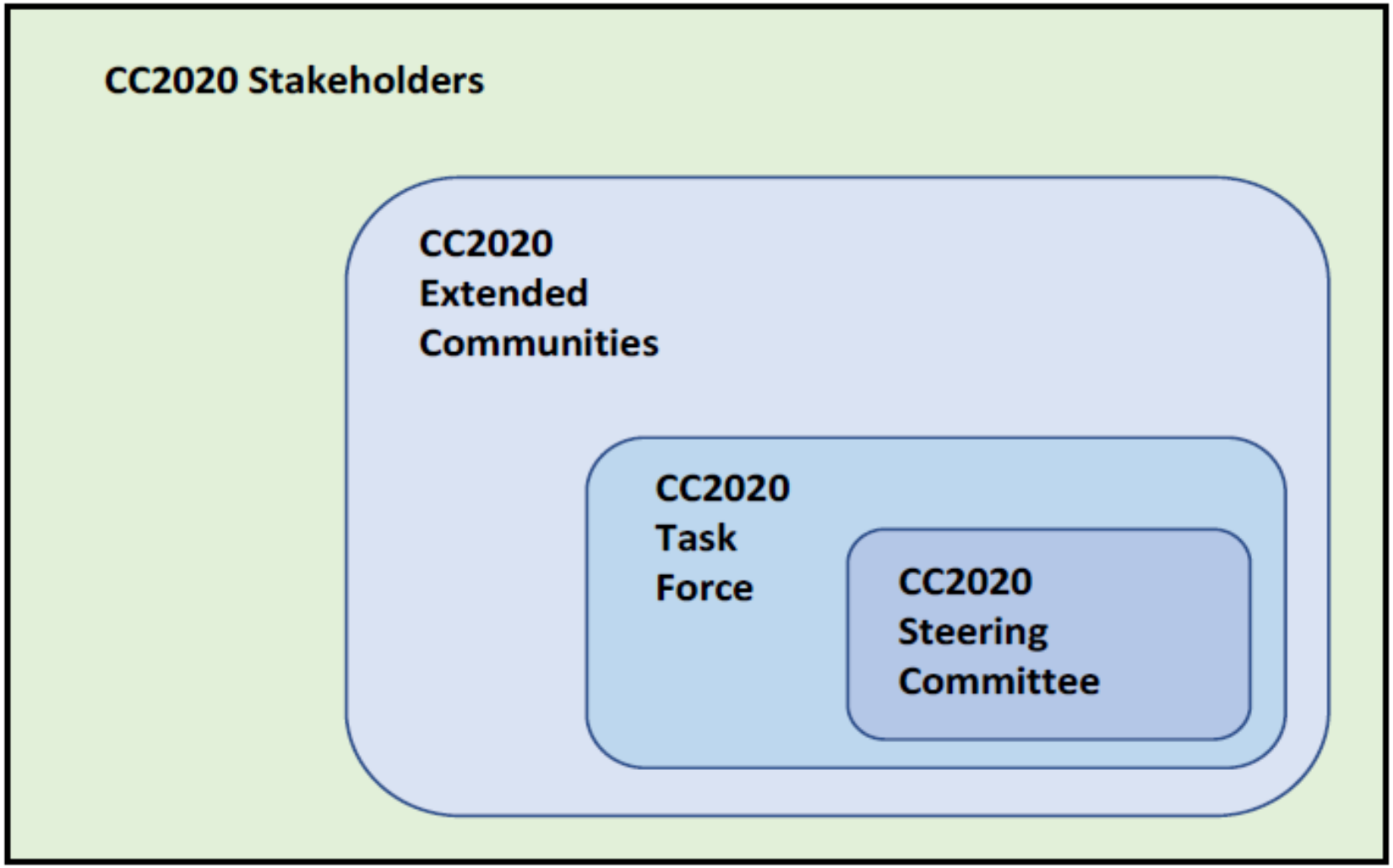


Figure 1.1. CC2020 project structure

Computing Curricula (ACM/IEEE)

Computing Curriculum Efforts

- CC2005 – Computing Curricula
- SE2014 – Software Engineering
- CS2013 – Computer Science
- CE2016 – Computer Engineering
- IT2017 – Information Technology
- CSEC2017 – Cybersecurity

https://www.computer.org/volunteering/boards-and-committees/professional-educational-activities/curricula

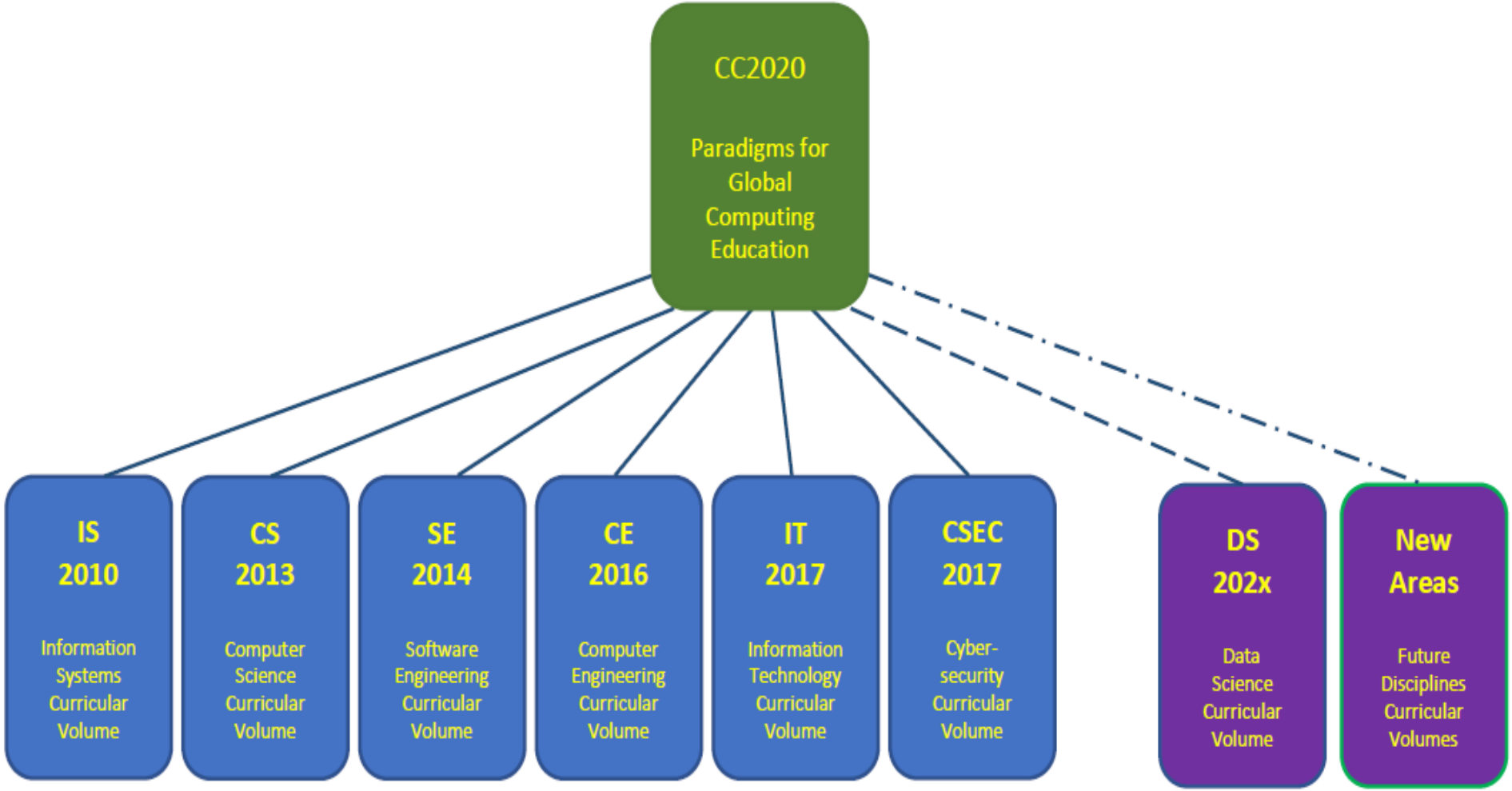


Figure 1.2 Structure of the Computing Curricula Series

Copyright © 2012 Pearson Education, Inc. Publishing as Pearson Addison-Wesley

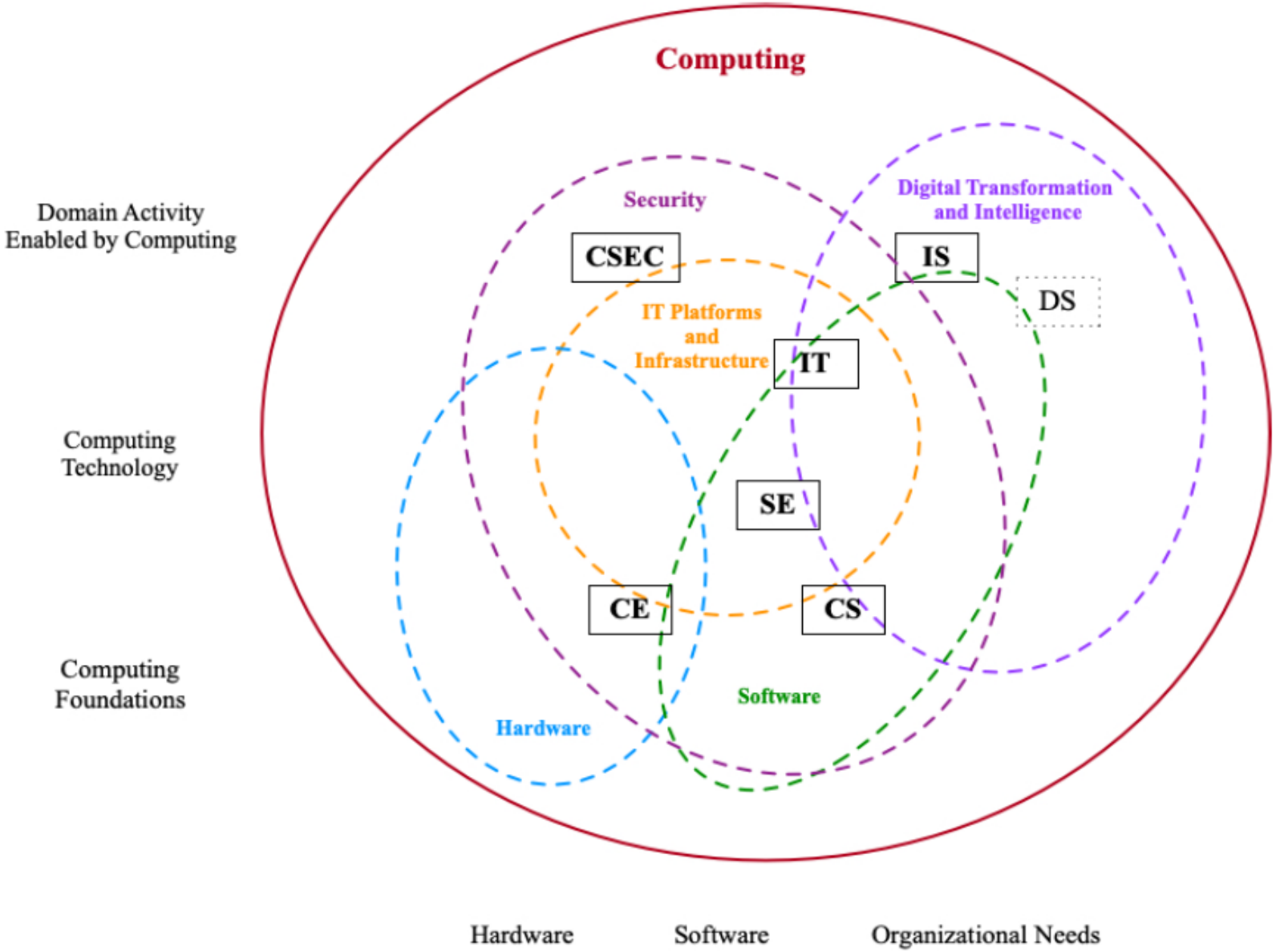


Figure 2.2. A contemporary view of the landscape of computing education

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).

From Knowledge-based Computing Education To Competency-based Computing Education

Competency = [Knowledge + Skills + Dispositions] *in Task*

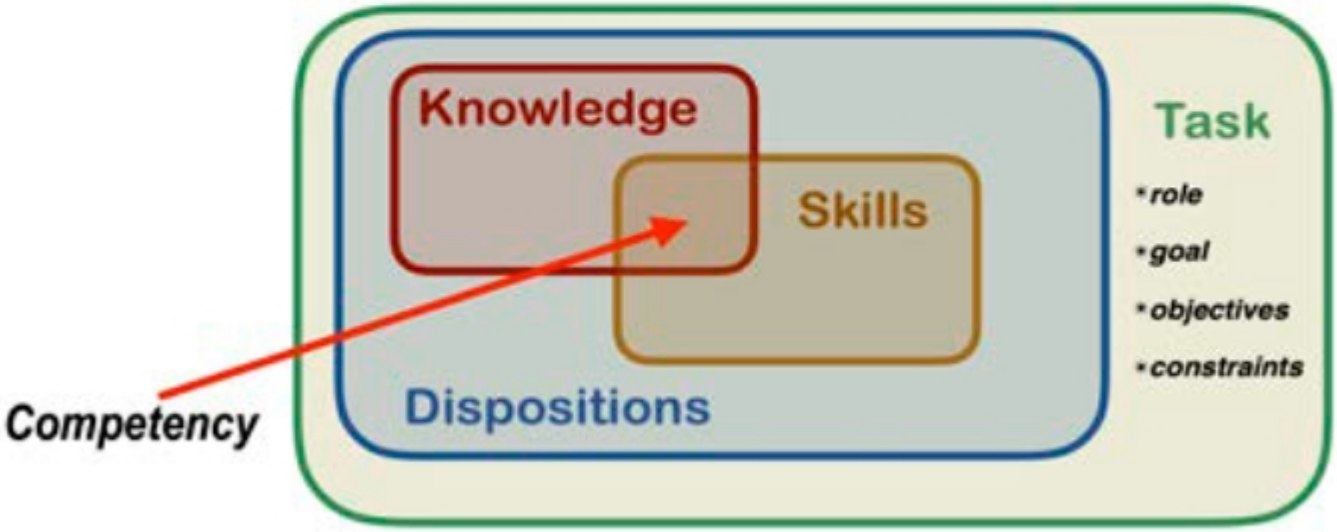


Figure 4.1. Conceptual Structure of the CC2020 Competency Model

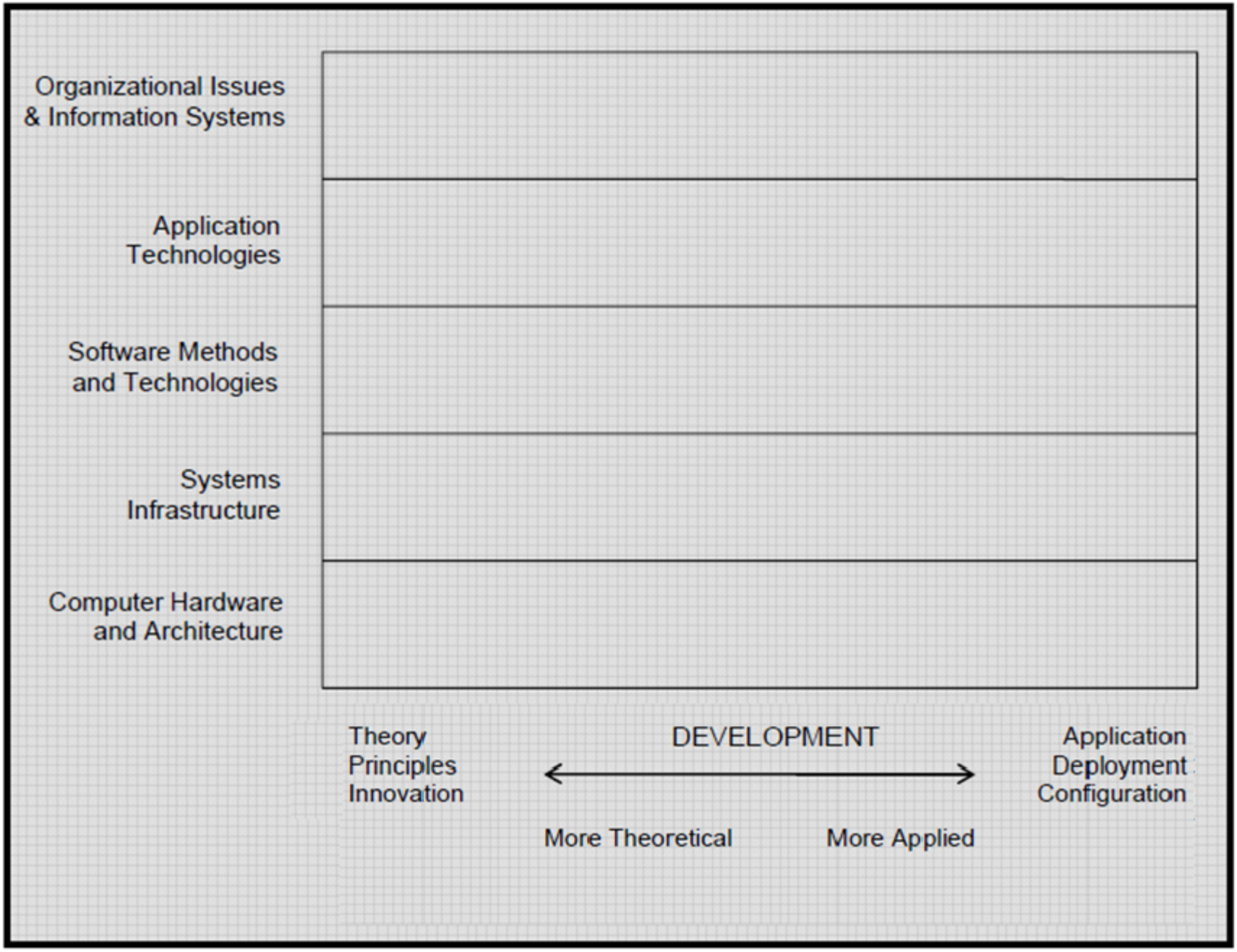


Figure 2.2. The Problem Space of Computing

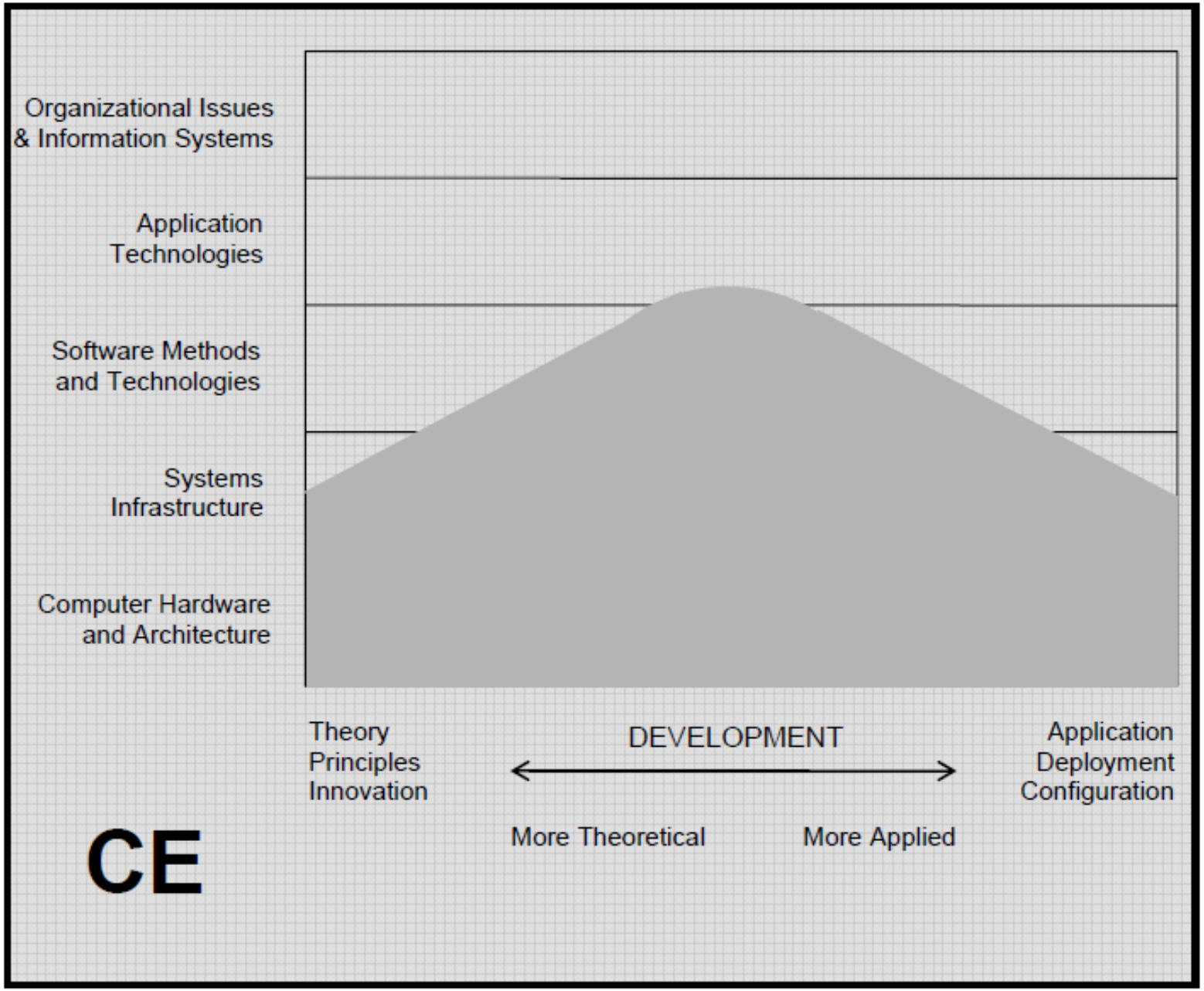


Figure 2.3. Computer Engineering

Computer Engineering

- *Computer engineering* is 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.

- Computer engineering students study the design of digital hardware systems, including *computers*, *communications systems*, and *devices that contain computers*. They also study *software development* with a focus on software used within and between digital devices.

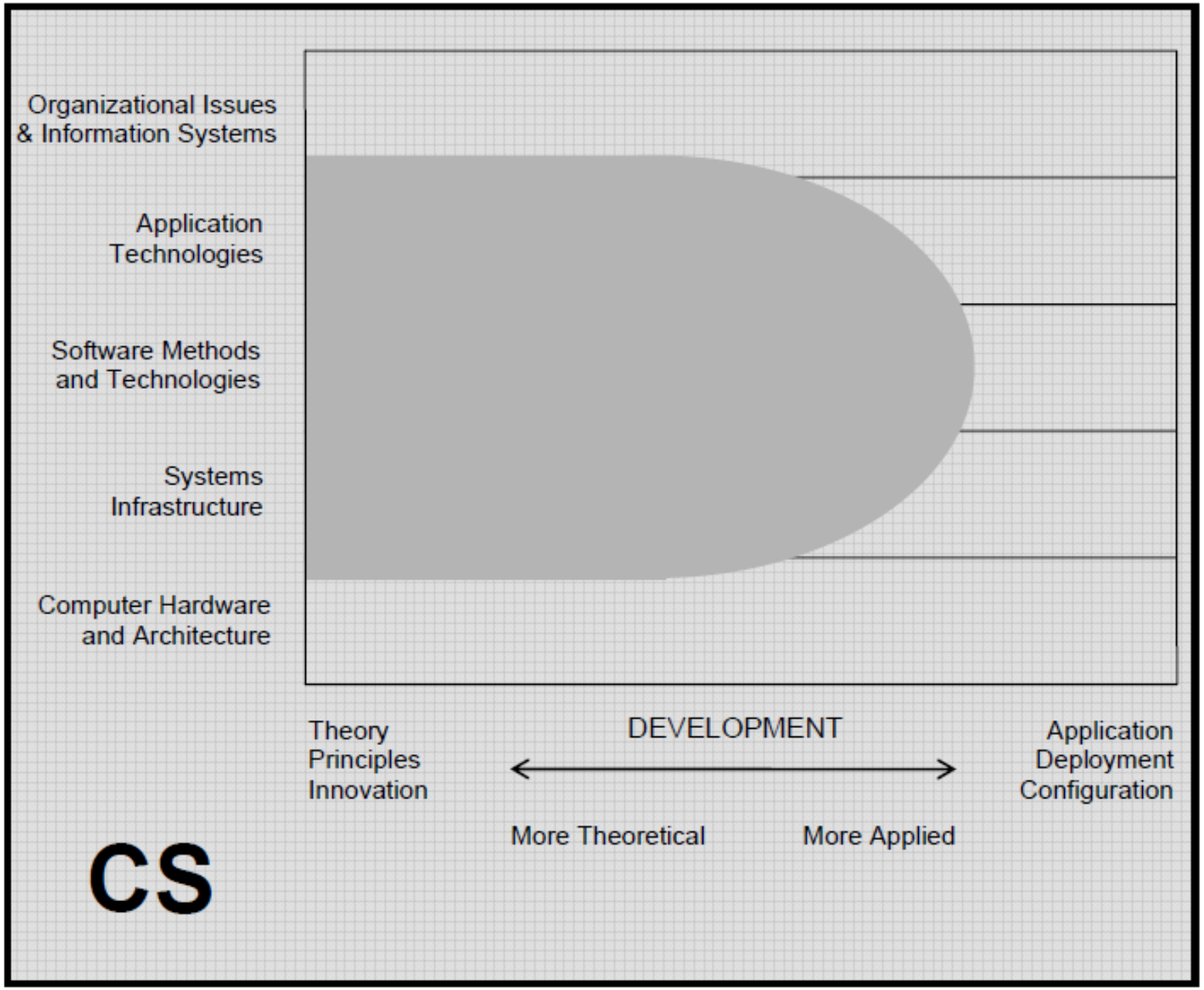


Figure 2.4. Computer Science

Computer Science

Computer science includes the work in three categories:

- *Design and implementation of software*. Computer scientists take on challenging programming jobs.
- *Devising new ways to use computers*, such as networking, database, human-computer-interface, World Wide Web, robotics, artificial intelligence, etc.
- *Developing effective ways to solve computing problems*. For example, computer scientists develop the best possible ways to store information in databases, send data over networks, and display complex images, and develop new approaches that provide better performance.

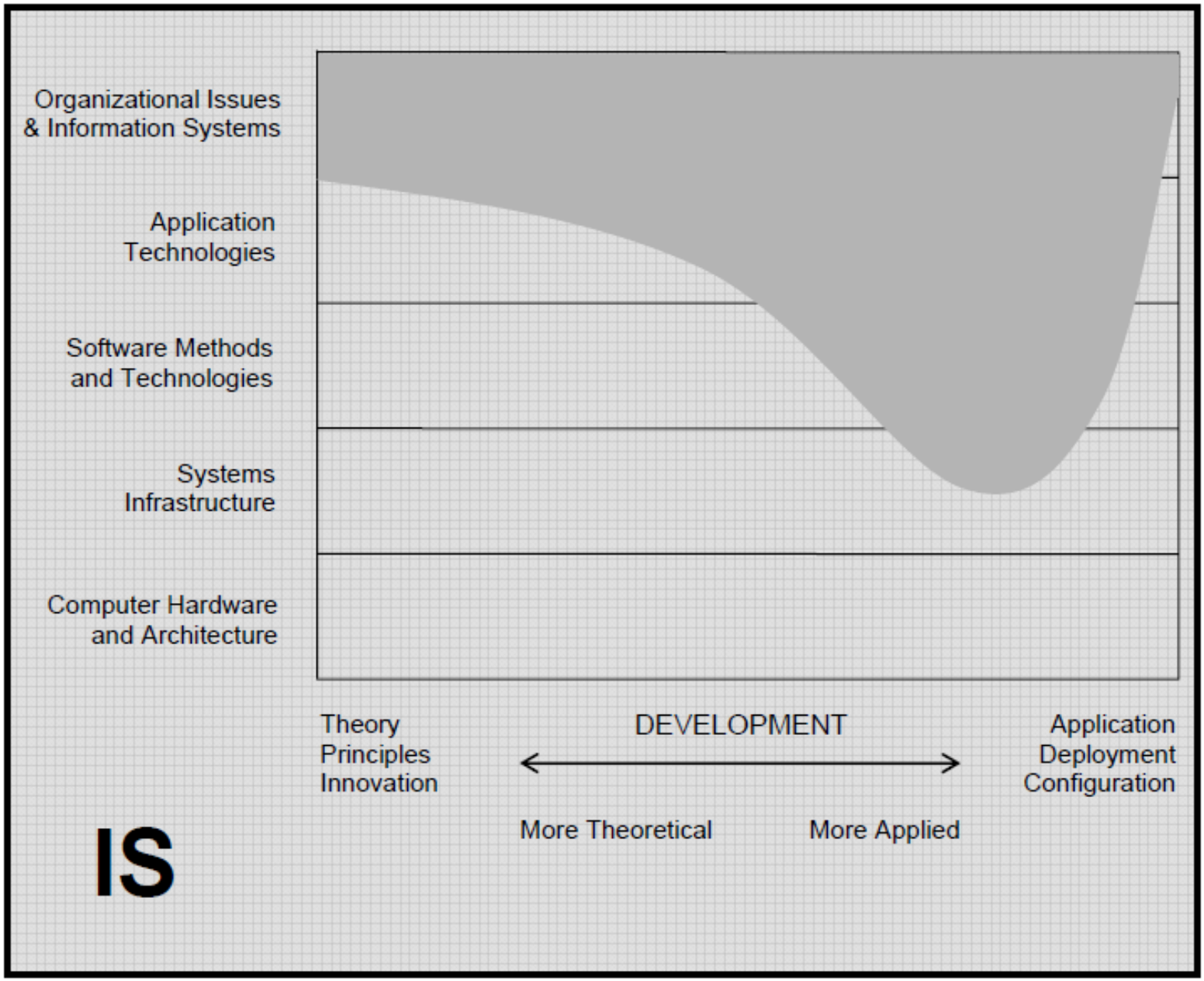


Figure 2.5. Information Systems

Information Systems (IS)

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.

Professionals in this discipline are primarily concerned with both technical and organizational factors.

The information systems specialist plays a key role in determining the requirements for an organization's information systems and is active in their specification, design, and implementation.

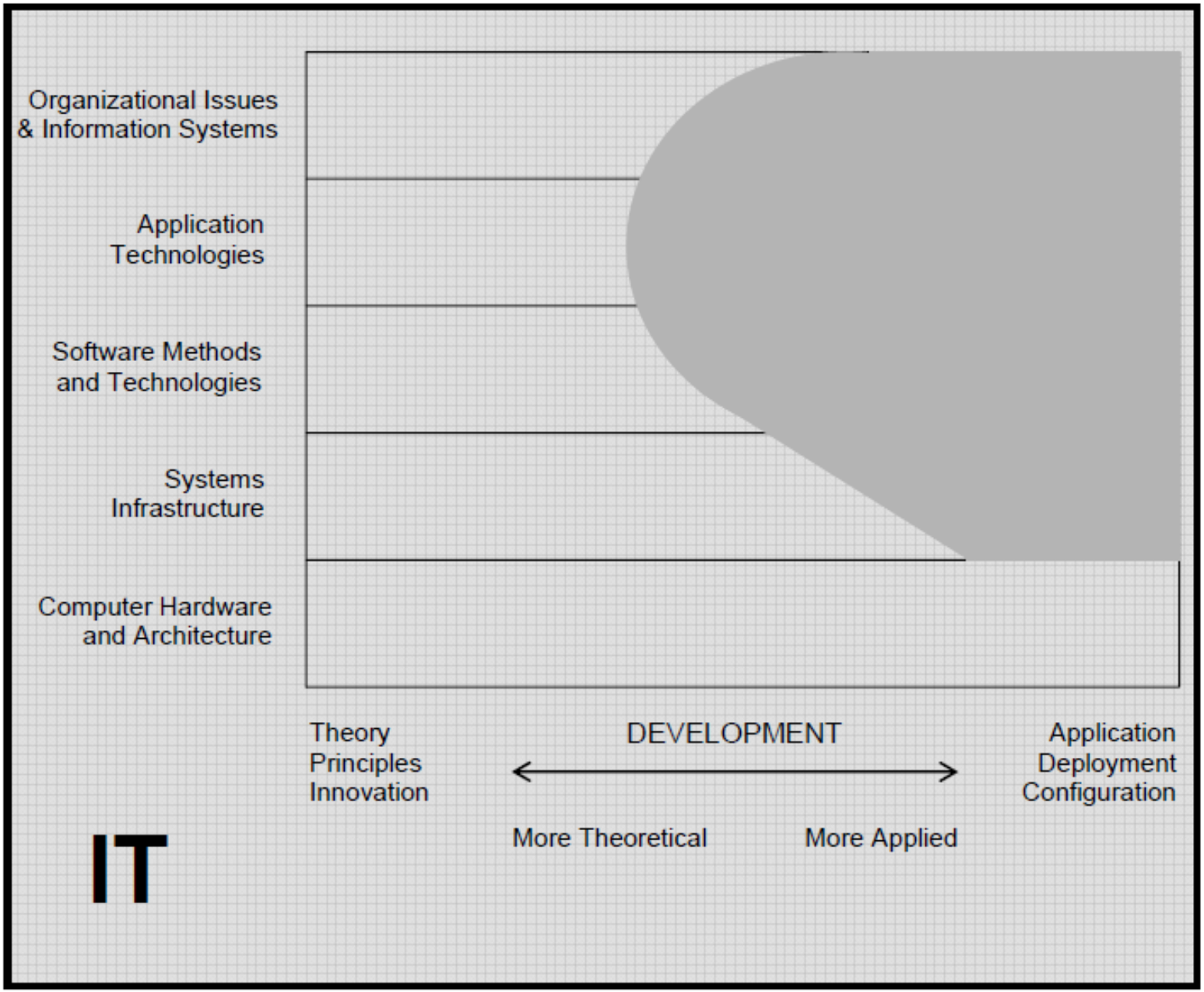


Figure 2.6. Information Technology

Information Technology (IT)

IT specialists take responsibility for

- selecting hardware and software products appropriate for an organization,
- integrating those products with organizational needs and infrastructure,
- and installing, customizing and maintaining those applications for the organization’s computer users.

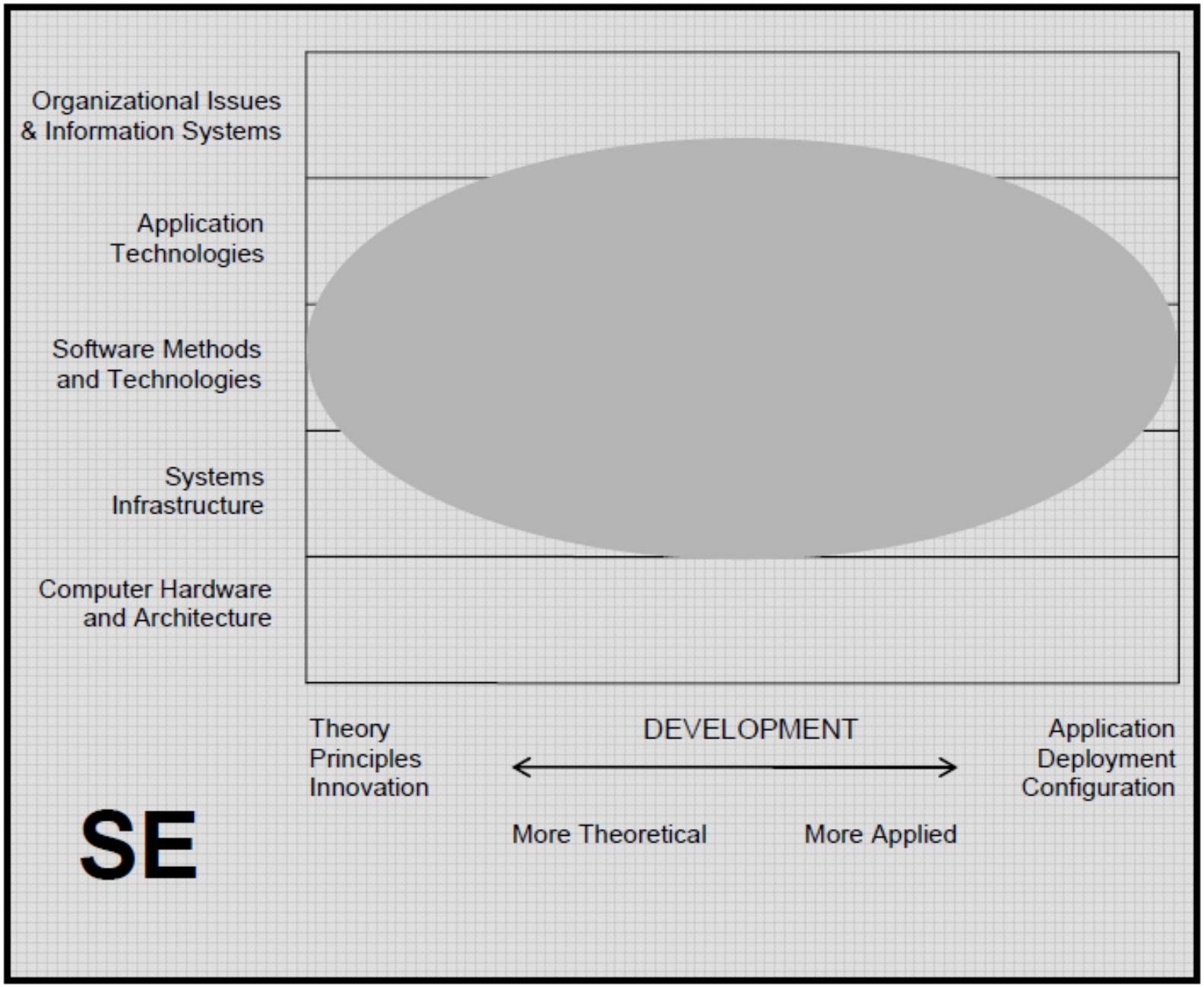


Figure 2.7. Software Engineering

Software Engineering

Software engineering is the discipline of developing and maintaining software systems that

- behave *reliably* and *efficiently*,
- are affordable to develop and maintain.

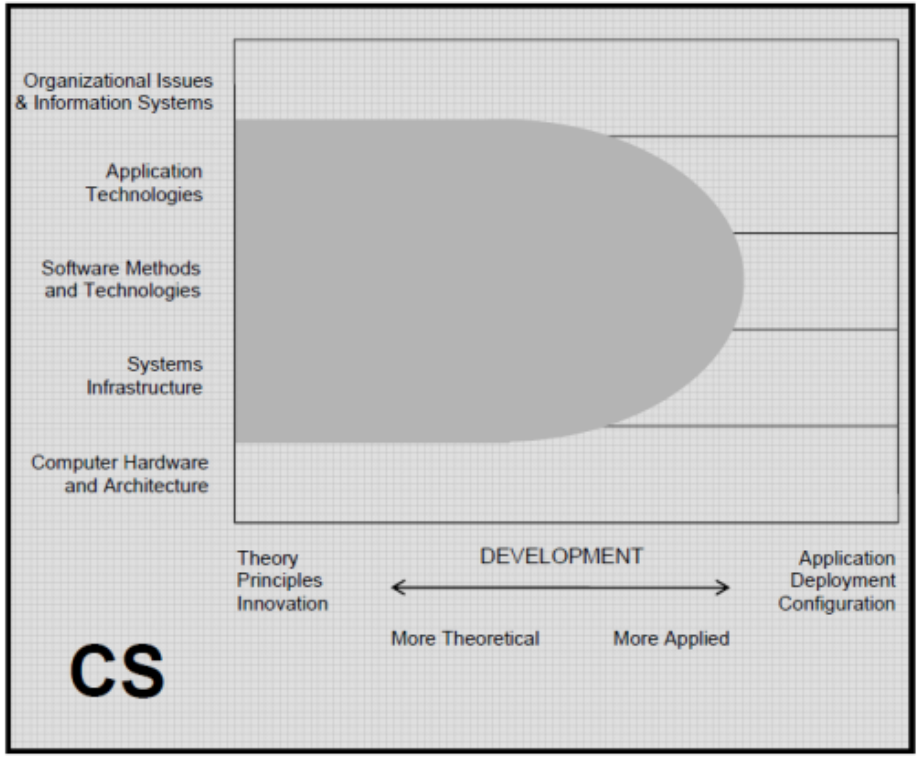


Figure 2.4. Computer Science

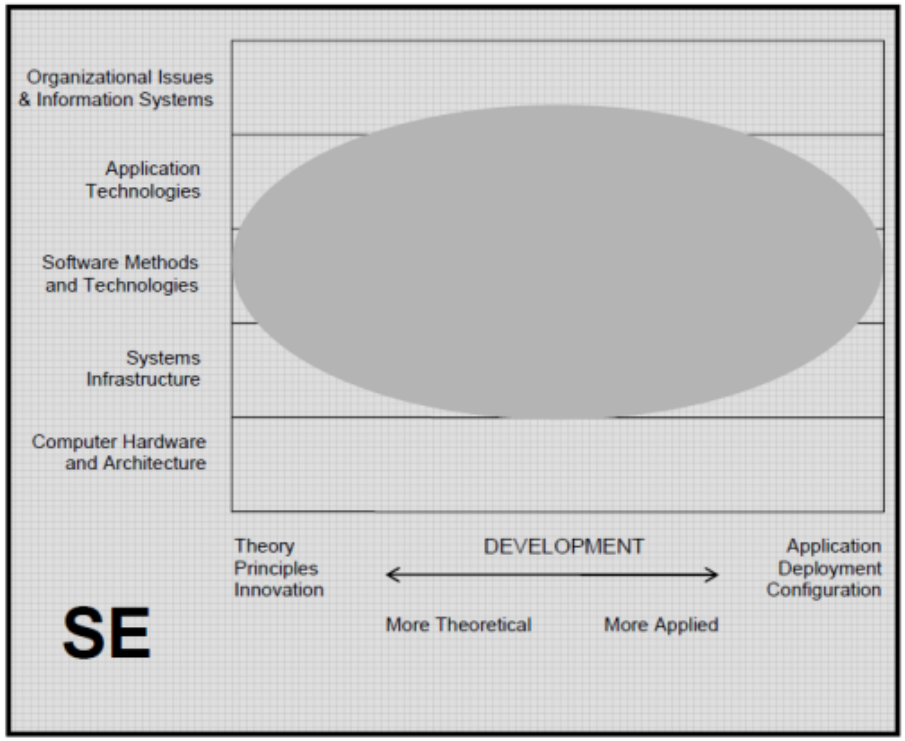


Figure 2.7. Software Engineering

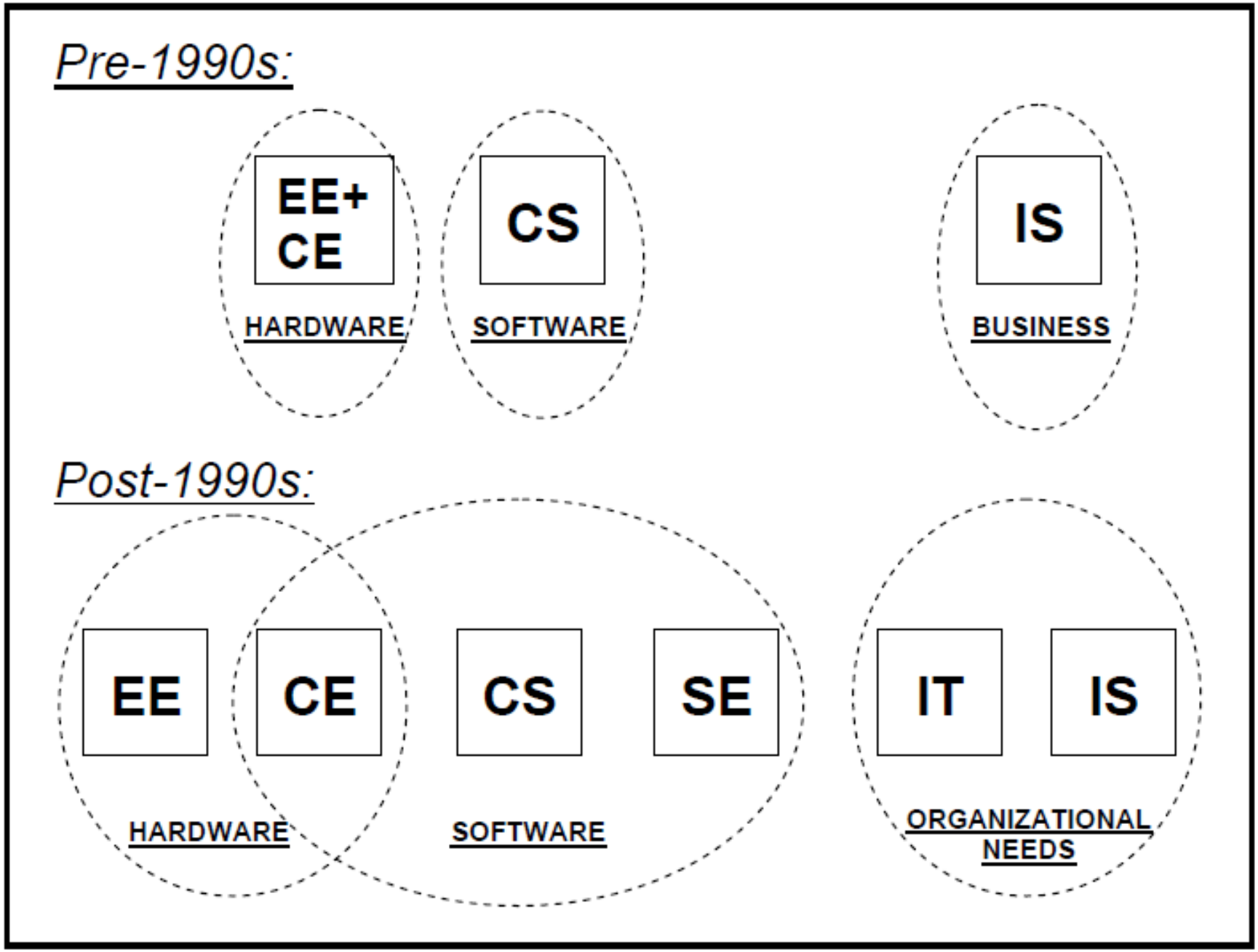


Figure 2.1. Harder Choices: How the Disciplines Might Appear to Prospective Students

Knowledge Areas

The CS2013 Body of Knowledge

- AL - Algorithms and Complexity
- AR - Architecture and Organization
- CN - Computational Science
- DS - Discrete Structures
- GV - Graphics and Visualization
- HCI - Human-Computer Interaction
- IAS - Information Assurance and Security
- IM - Information Management
- IS - Intelligent Systems
- NC - Networking and Communications
- OS - Operating Systems
- PBD - Platform-based Development
- PD - Parallel and Distributed Computing
- PL - Programming Languages
- SDF - Software Development Fundamentals
- SE - Software Engineering
- SF - Systems Fundamentals
- SP - Social Issues and Professional Practice

Core Hours in Knowledge Areas

Knowledge Area	CS2013 Tier1	CS2013 Tier2	CS2008 Core	CC2001 Core
AL-Algorithms and Complexity	19	9	31	31
AR-Architecture and Organization	0	16	36	36
CN-Computational Science	1	0	0	0
DS-Discrete Structures	37	4	43	43
GV-Graphics and Visualization	2	1	3	3
HCI-Human-Computer Interaction	4	4	8	8
IAS-Information Assurance and Security	3	6	--	--
IM-Information Management	1	9	11	10
IS-Intelligent Systems	0	10	10	10
NC-Networking and Communication	3	7	15	15
OS-Operating Systems	4	11	18	18

Core Hours in Knowledge Areas

Knowledge Area	CS2013 Tier1	CS2013 Tier2	CS2008 Core	CC2001 Core
PBD-Platform-based Development	0	0	--	--
PD-Parallel and Distributed Computing	5	10	--	--
PL-Programming Languages	8	20	21	21
SDF-Software Development Fundamentals	43	0	47	38
SE-Software Engineering	6	22	31	31
SF-Systems Fundamentals	18	9	--	--
SP-Social Issues and Professional Practice	11	5	16	16
Total Core Hours	**165**	**143**	**290**	**280**

All Tier1 + All Tier2 Total	**308**
All Tier1 + 90% of Tier2 Total	**293.7**
All Tier1 + 80% of Tier2 Total	**279.4**

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

Table D.1 Representative Summary of Computing Knowledge Areas

Categorization	Computing Knowledge Area
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

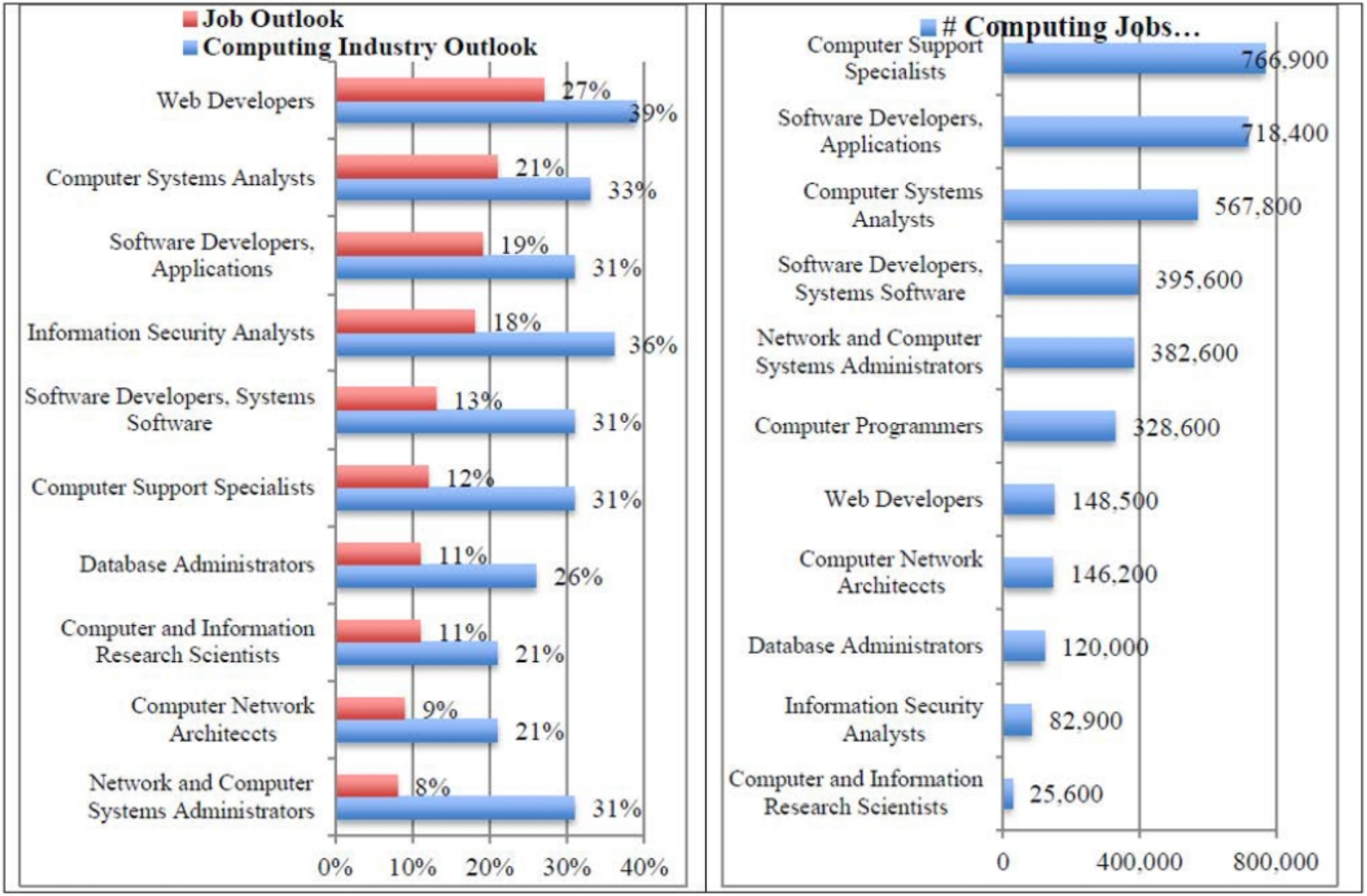


Figure 3.3. Left: Computing occupations projected growth 2014-2024 across all sectors (job outlook) and in the computing sector. Right: Computing jobs in 2014. (*Courtesy of Bureau of Labor Statistics*)

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

(a) Before choosing

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

(b) After choosing

Figure 5.1. Choosing dispositions by a prospective student

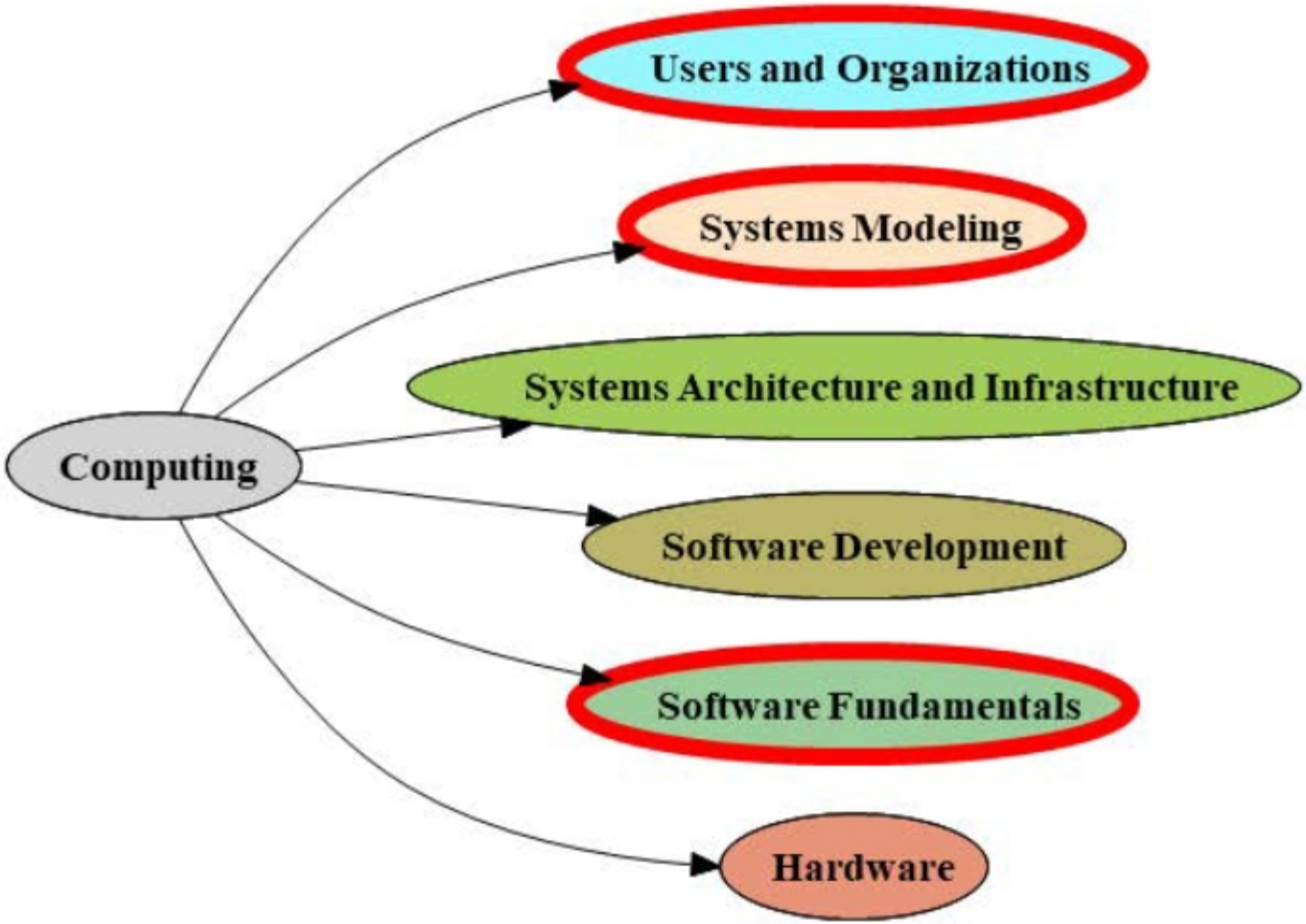


Figure 5.2. The student's choice of computing categories

(a) Choosing knowledge areas (b) Final result

Figure 5.3 Detailed choice of knowledge areas

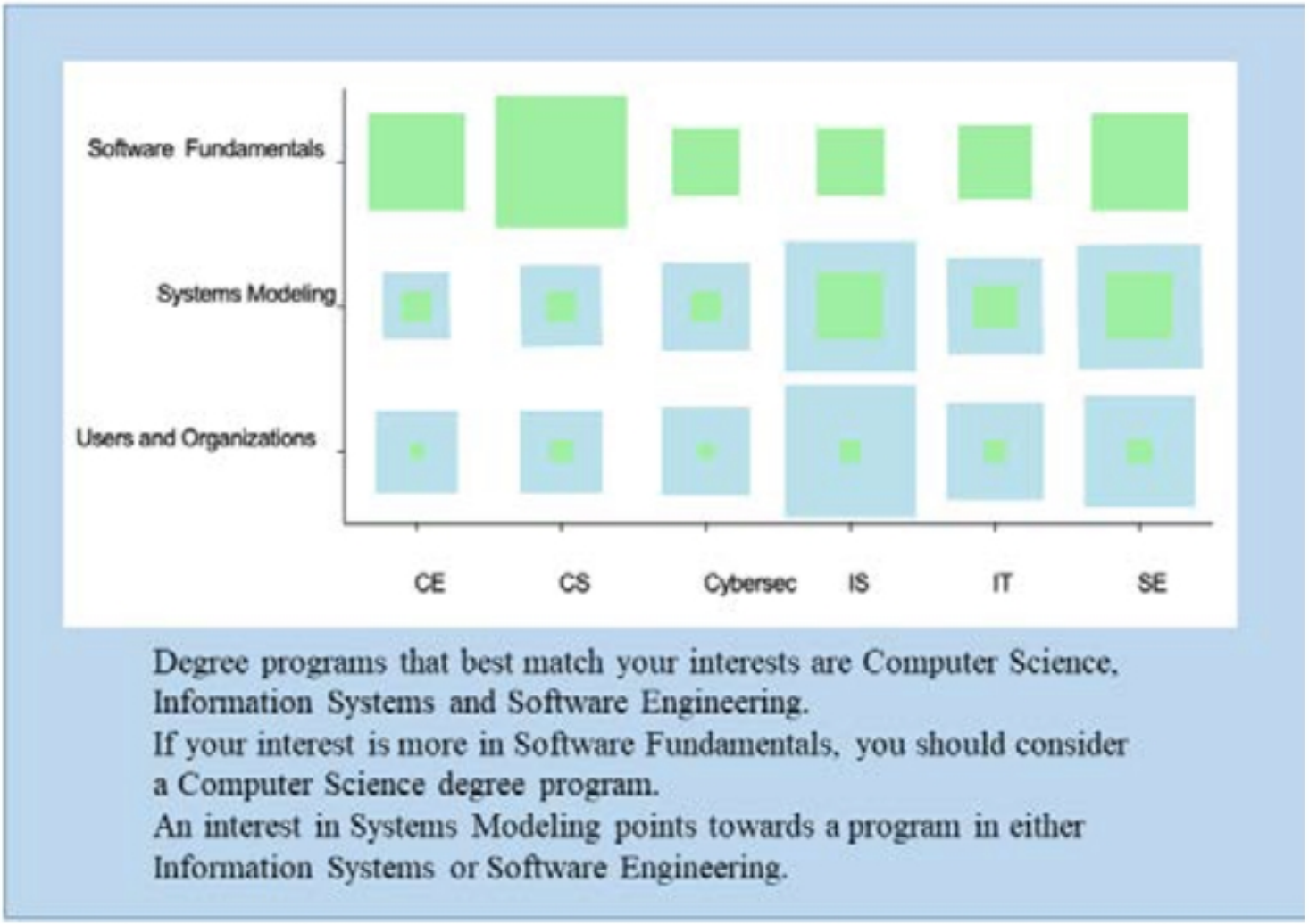


Figure 5.4. Mapping of chosen knowledge categories to the six curricular guidelines

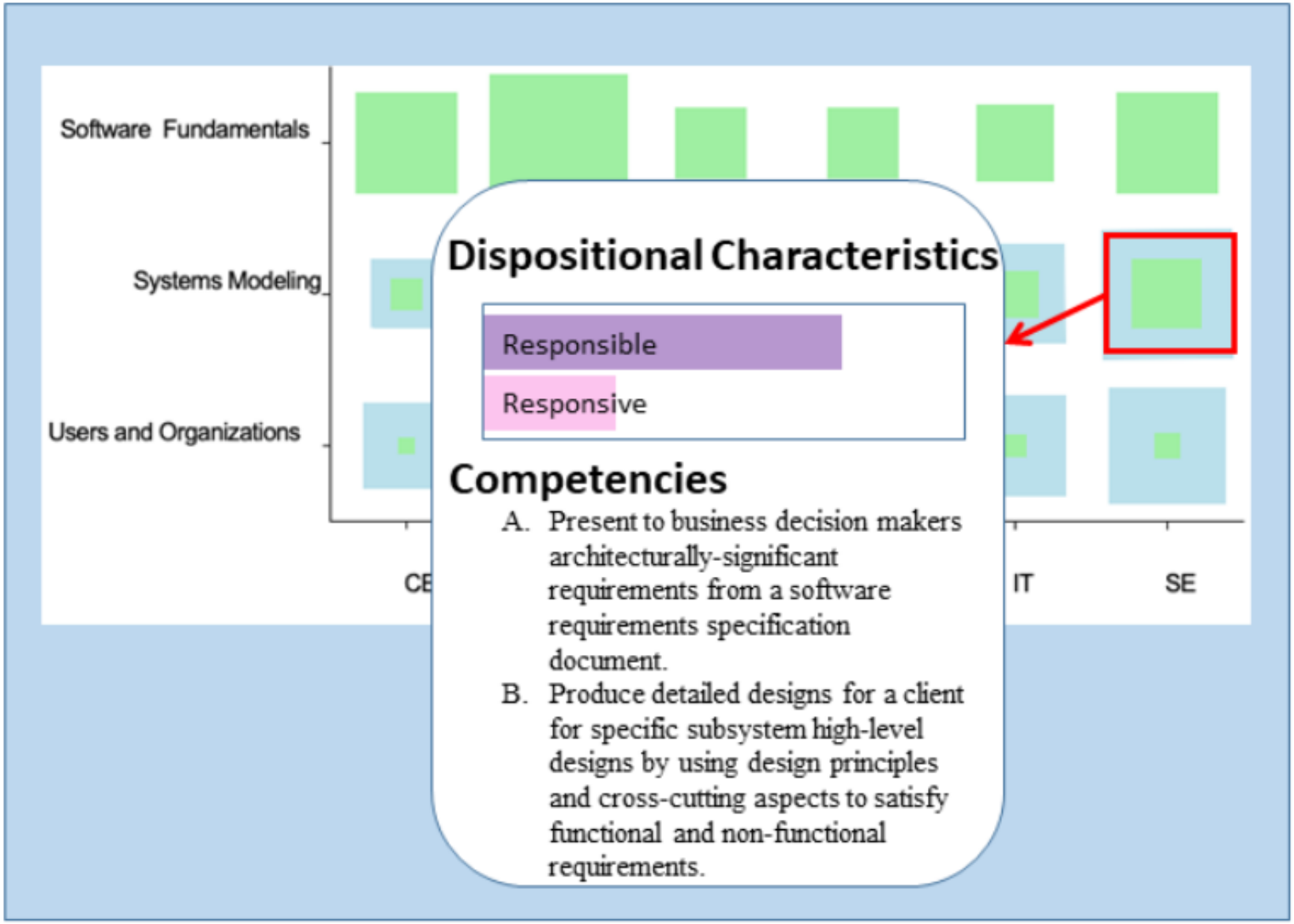


Figure 5.5. Disposition and competency details

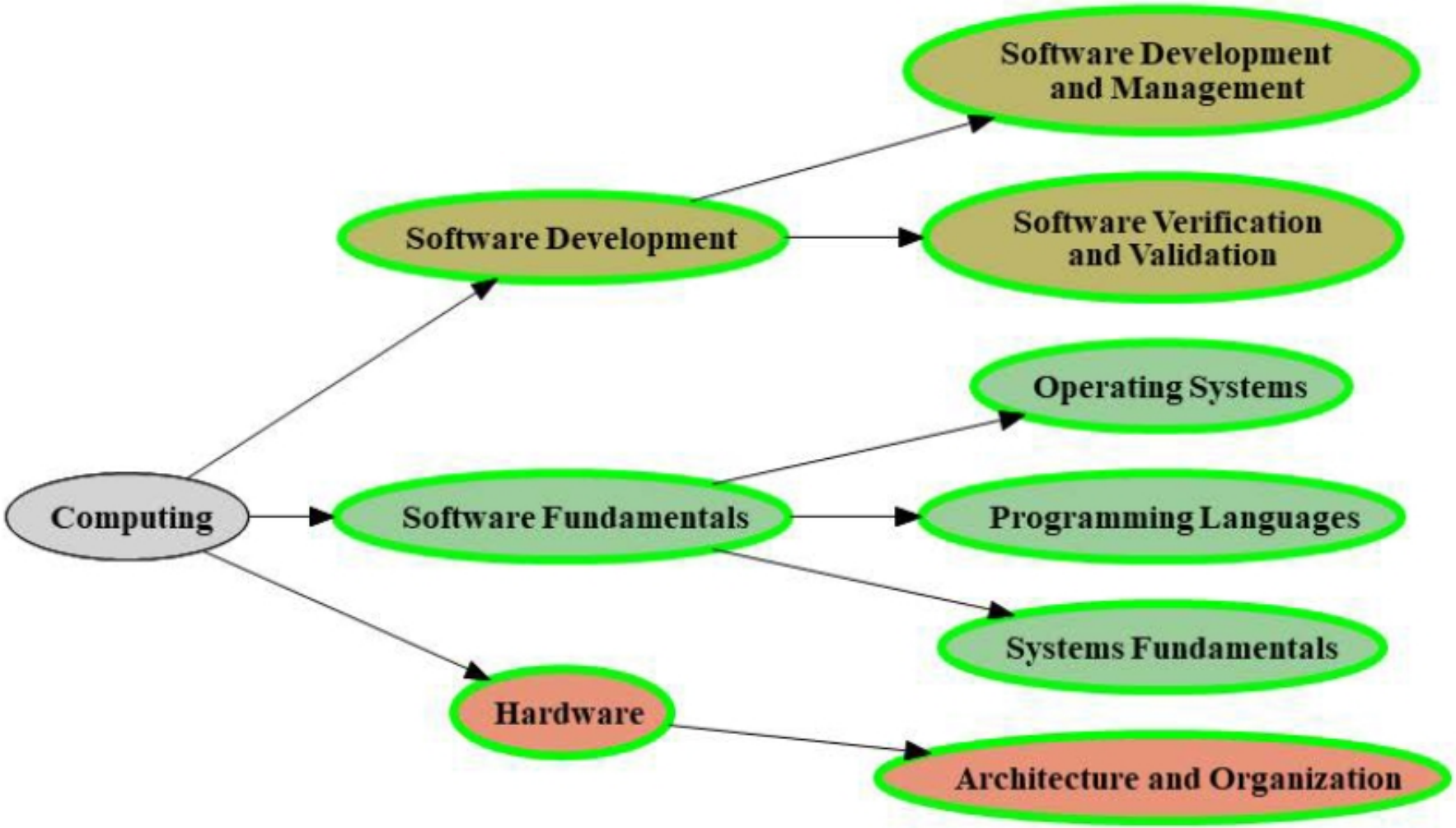


Figure 5.6 Result of knowledge areas selection

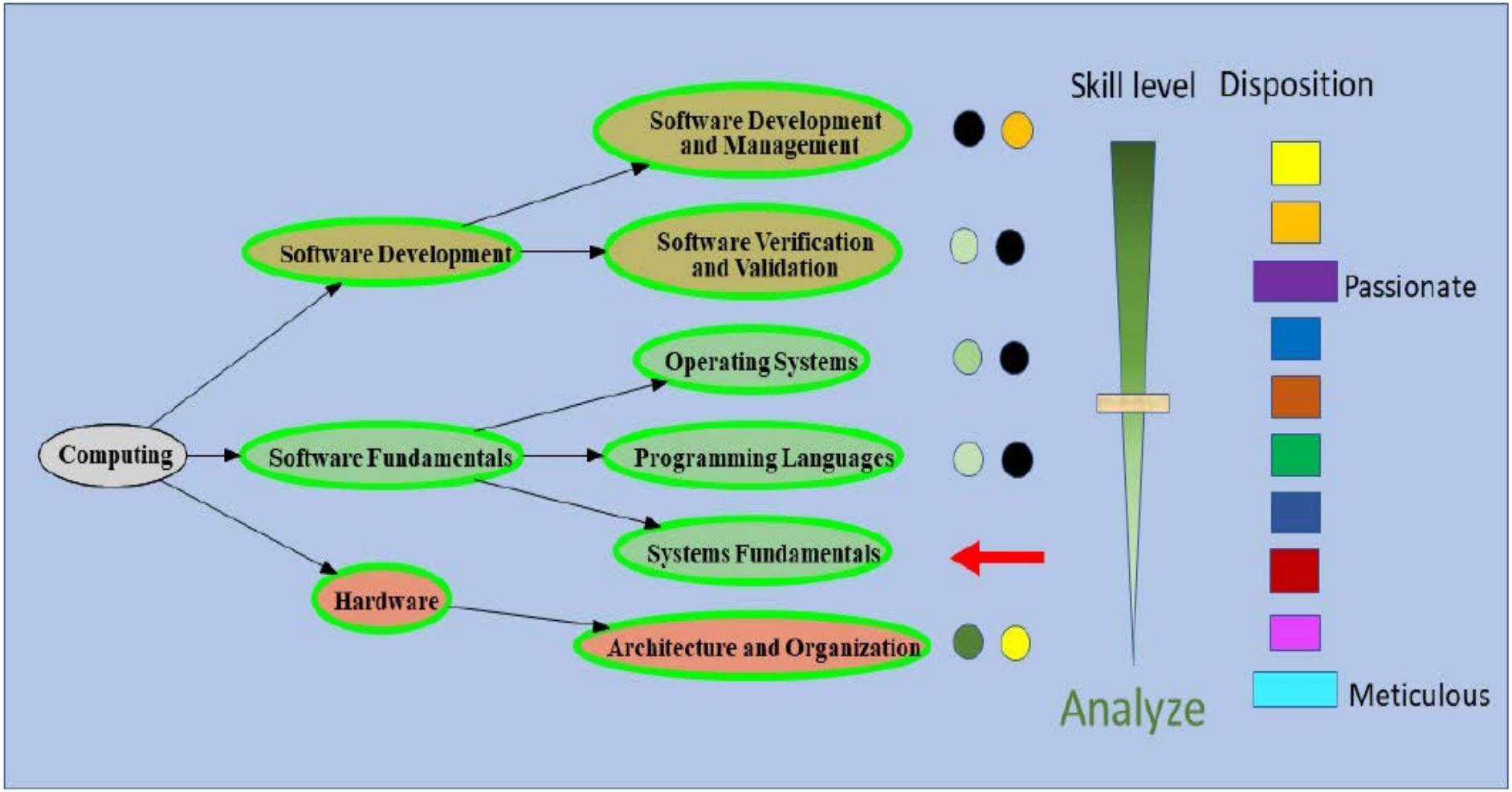


Figure 5.7. Detailing skill and disposition

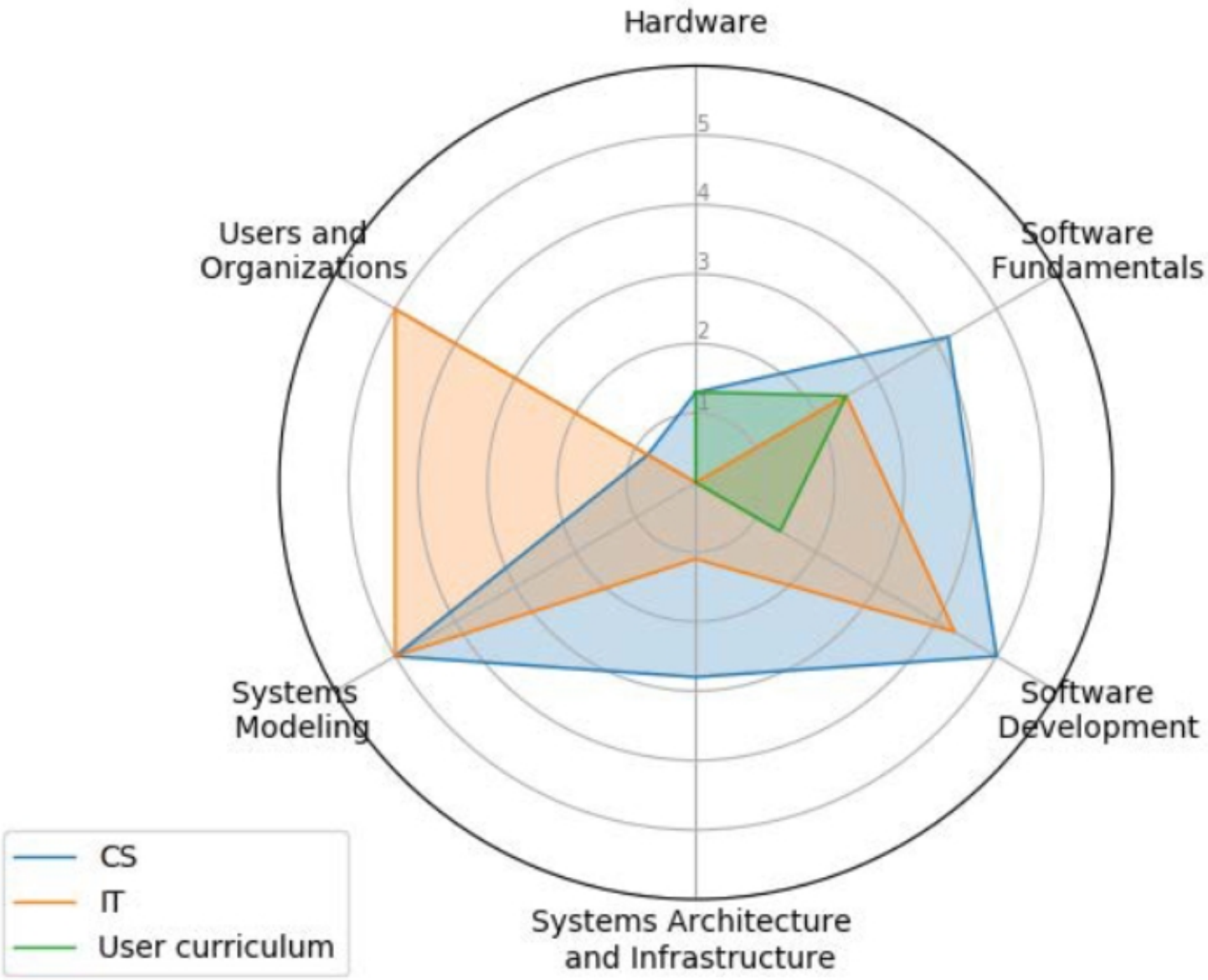


Figure 5.8. Comparison of CS and IT based on knowledge level

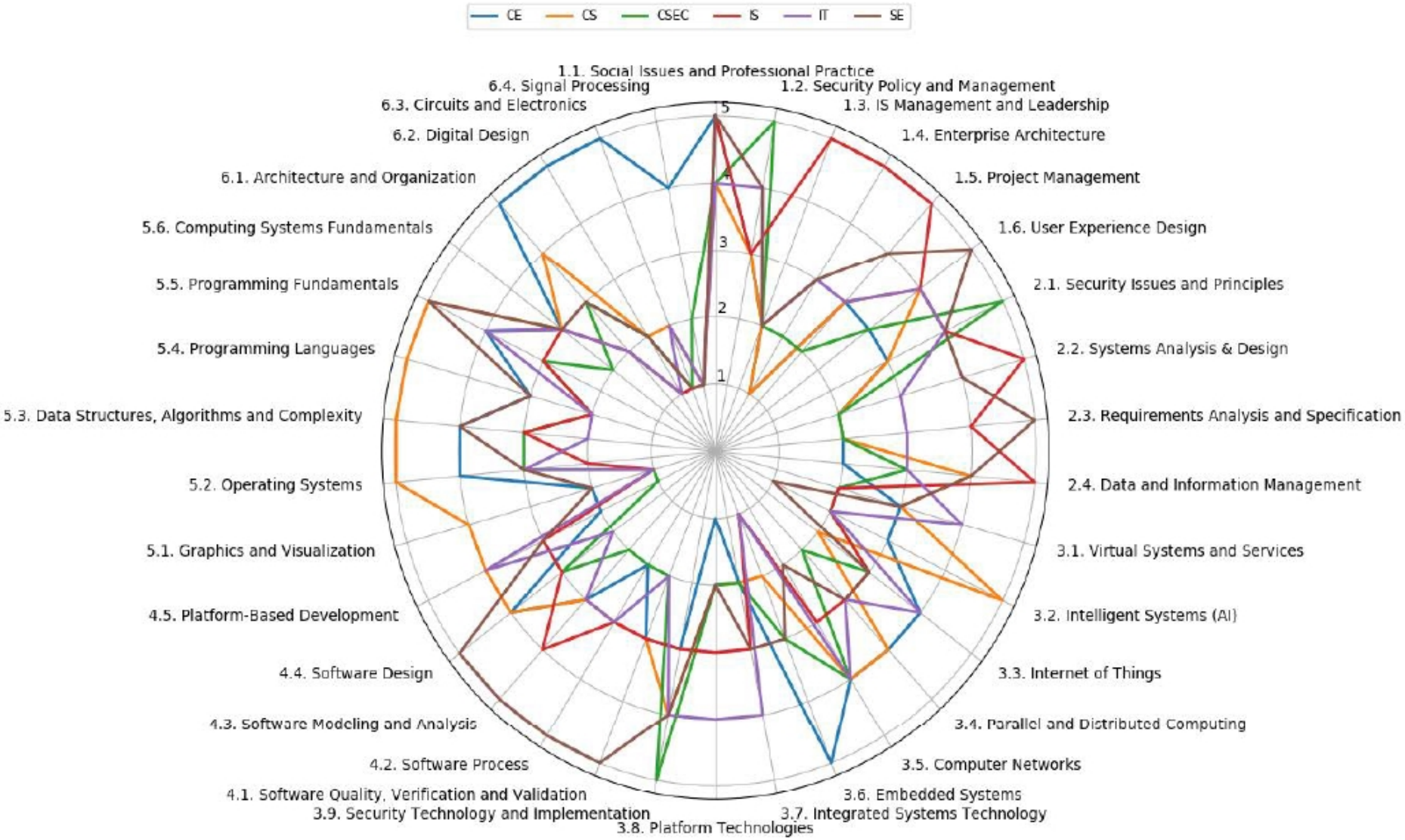


Figure 5.10. Radar Chart showing maximum emphasis of knowledge areas

Critical Thinking

Reasoning

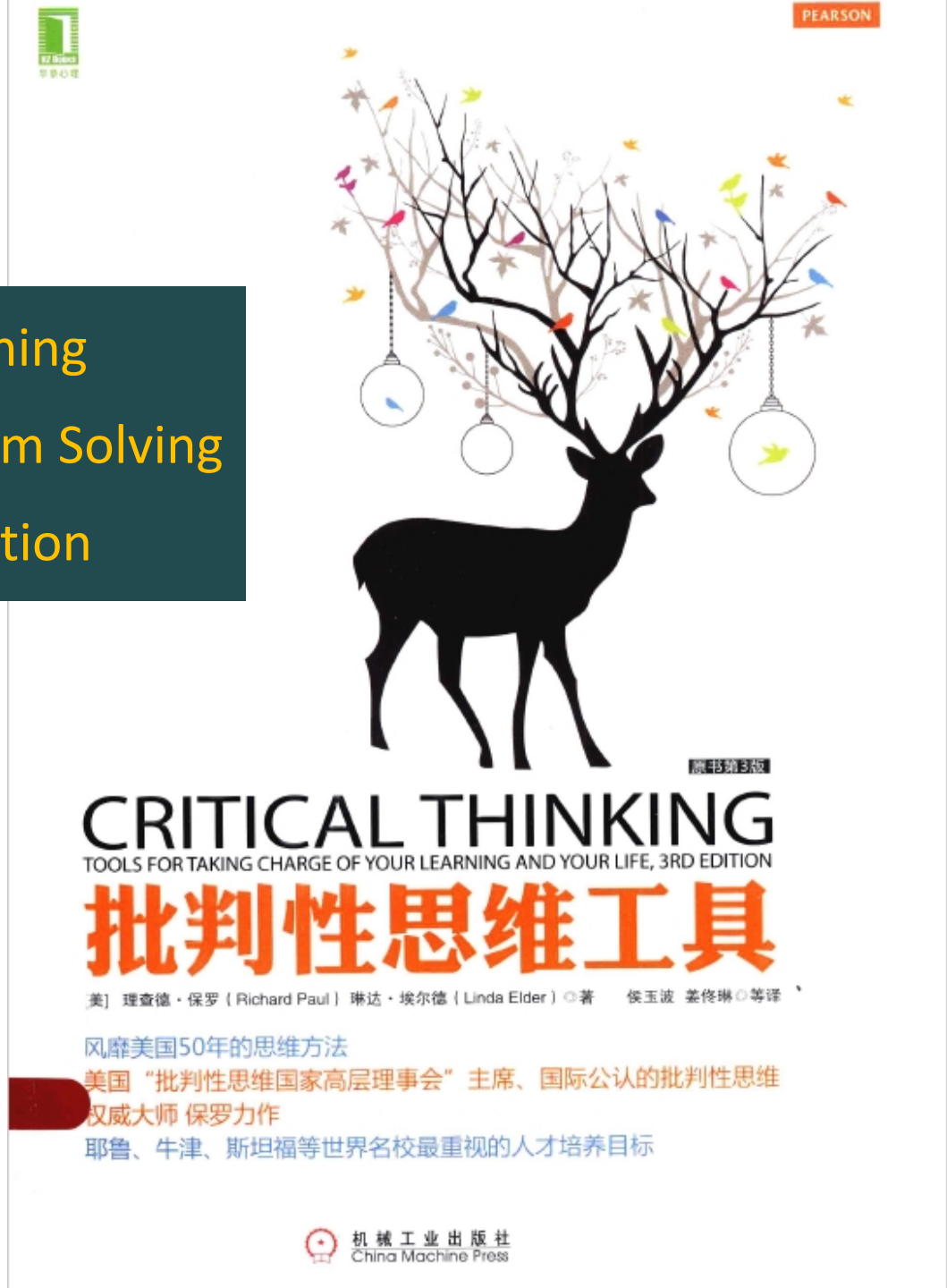
Problem Solving

Evaluation

- 关键性思考
  建设性思考

- Critical Observing:
  建设性观察

- 系统化认知问题 与
  构建解决方案的能力



```
x = x + y;
y = x - y;
x = x - y;
```

1.程序的作用是什么?

2.如何证明?

```
// x=A & y=B
x = x + y;
// x=A+B & y=B
y = x - y;
// x=A+B & y=A
x = x - y;
// x=B & y=A
```

前提条件

后置条件

3.程序真的完成了交换吗?

不一定!

x+y可能溢出。

比较前置条件与后置条件知:

程序交换了x与y的值!

```
x = x ^ y;
y = x ^ y;
x = x ^ y;
```

按位异或(for bit a, b, c)：

a ^ a = 0

a ^ 1 = ~a

a ^ 0 = a

a ^ b = b ^ a

a ^ b ^ c = a ^ (b ^ c)

```
// x=A & y=B
x = x ^ y;
// x=A^B & y=B
y = x ^ y;
// x=A^B & y=A
x = x ^ y;
// x=B & y=A
```

前提条件

后置条件

正确无误的交换！

一个简单的数学思考：

假设 1=0 或 0=1（只需要一个非真的前提），对任意的x和y，有：

1) $x = 0 \Leftarrow x*1 = x*0$

2) $x = y \Leftarrow (x-y)*1 = (x-y)*0$

什么是GROW模型？

GROW模型是约翰·惠特默在1992年提出的，现已成为企业教练领域使用最广泛的模型之一。基本的GROM模型来自于一个决策四阶段的模型英文缩写，即**Goal（目标）、Reality（现状）、Option（方案）、Will(意愿)这四个英文单词的首字母。**

3 Dimensions of Typical CS Courses

Most of Courses on CS may be organized by a combination of elements from the following 3 dimensions:

THEORY related: Concepts, Models, Maths, Algorithms, Principles, Mechanisms, Methods, … Need to understand Abstract Things

SYSTEM and TOOLS related: HW, Network, OS, PLs, IDEs, DBMS, Clients, Servers, Virtual Machines, Containers in Cloud, … Need to understand, setup and use them properly

DESIGN related: According to Requirements (Problems), need to use Theory and Tools to shape/create/implement/test Solutions

Put It All Together!

Thank You! ☺