Software Engineering
Course Introduction

Dr. 何明昕 HE Mingxin, MX, Max

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

Download from c.program@yeah.net

/文件中心/网盘/SoftwareEngineering2024

Textbook

http://eceweb1.rutgers.edu/~marsic/books/SE/

Software Engineering

textbook by Ivan Marsic

SHARE

& Software Engineering book [PDF document; size: 13.6 MBytes]

Last updated: September 10, 2012
Pages: 613
Table of Contents
Note: Problem solutions are included on the back of the book, starting from page 523.

@ Instructor materials

- Lecture slides

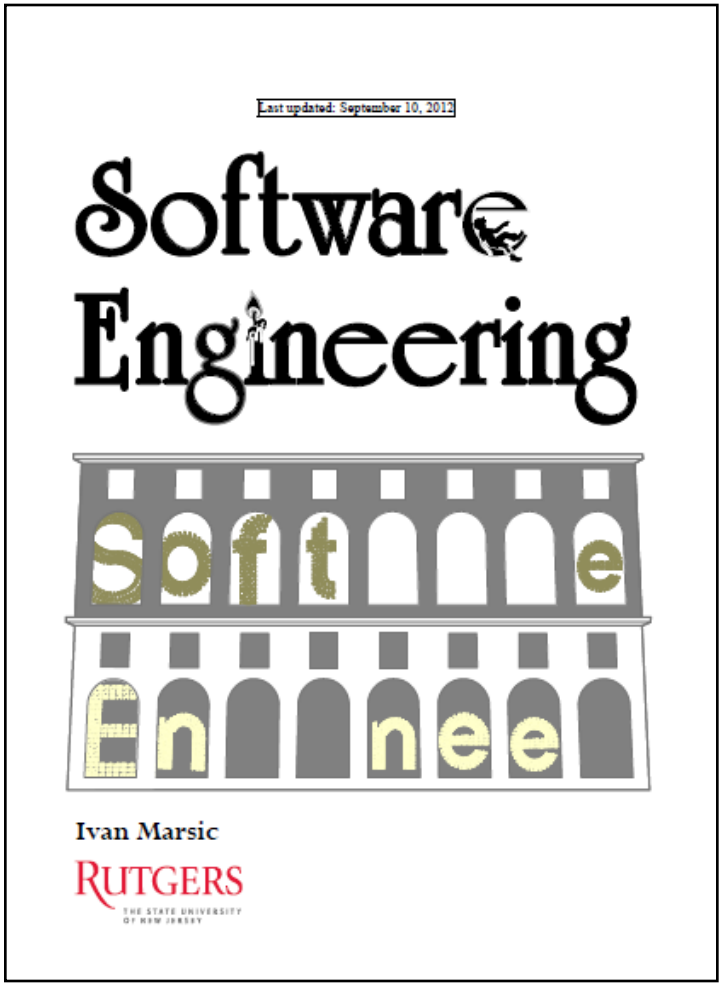
@ Team projects

" Related online resources

More books by Ivan Marsic
(Computer Networks, Wireless Networks, User Interfaces)

H Ivan Marsic's home page

This page last modified: Mon Jan 16 20:40:10 EST 2012 [Valid HTML 4.01]



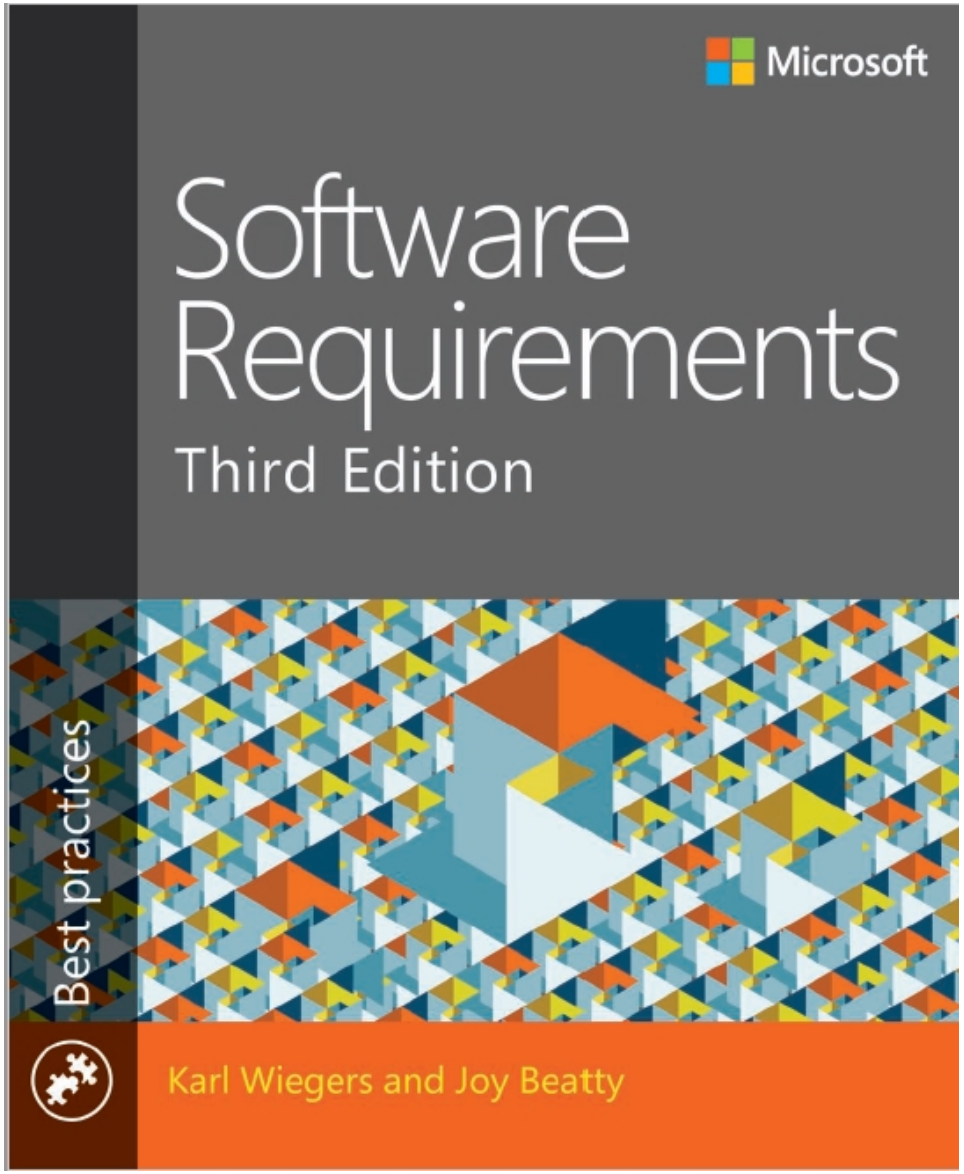
Reference Textbook

Karl Wiegers and Joy Beatty.

Software Requirements,

Third Edition

Microsoft, 2013



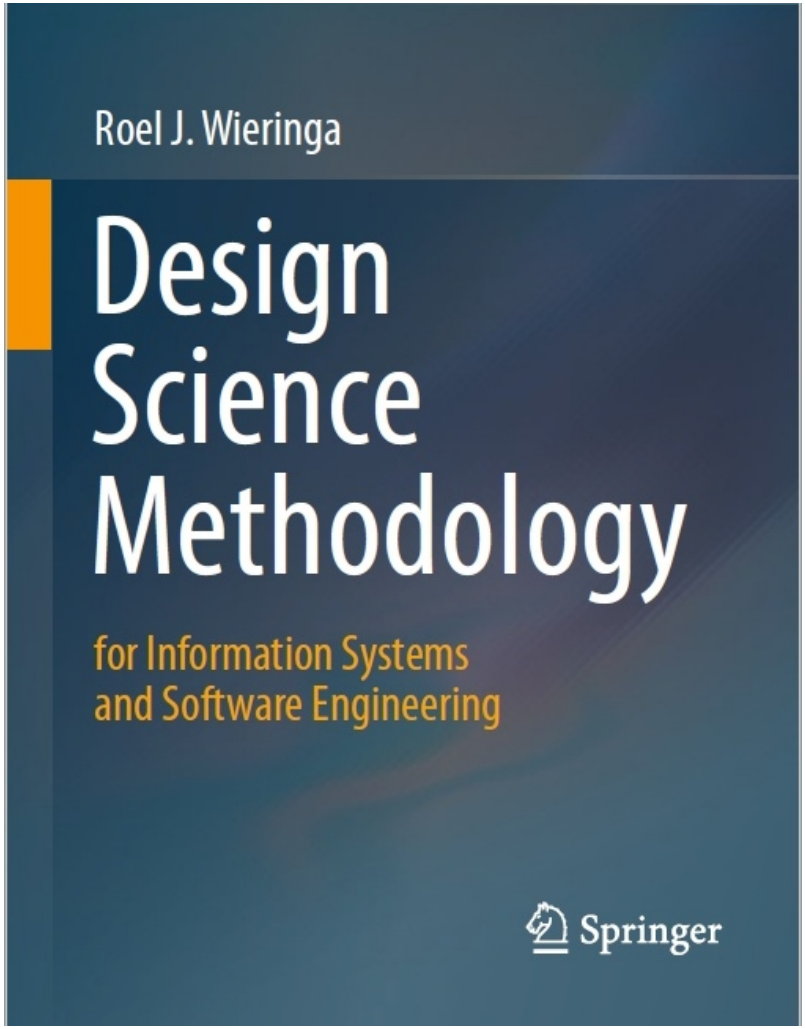
Reference Textbook

R.J. Wieringa:

Design Science Methodology

for Information Systems and Software Engineering.

Springer, 2014

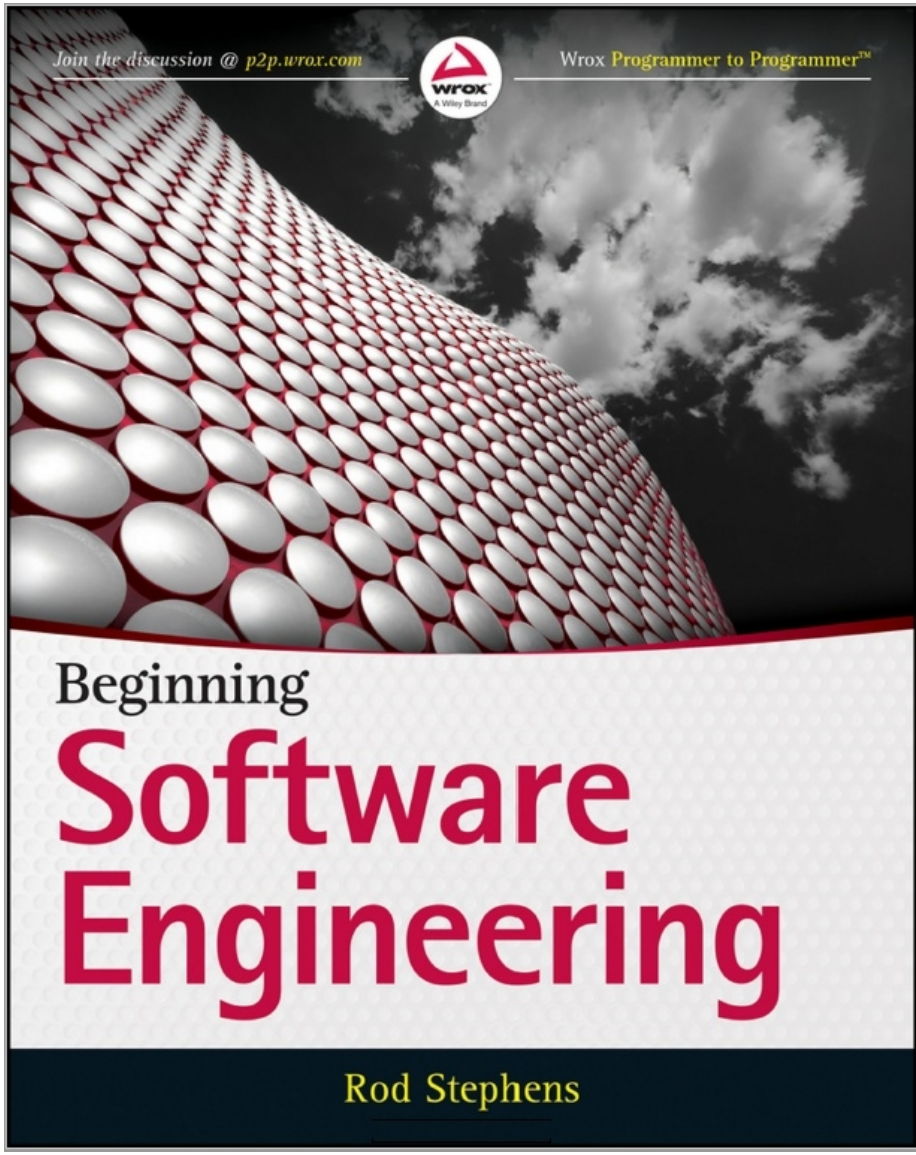


Reference Textbook

Rod Stephens.

Beginning Software Engineering,

John Wiley & Sons, 2015



Reference Textbook

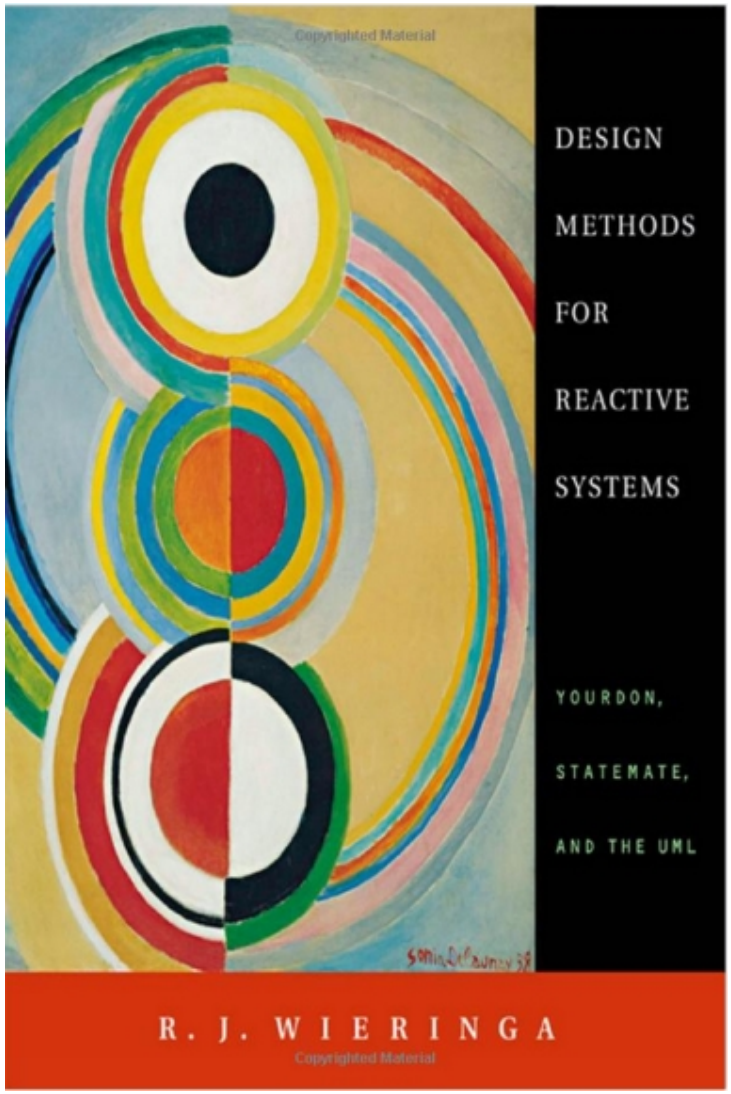
R.J. Wieringa:

Design Methods for Reactive Systems

Yourdon, Statemate, and the UML

Elsevier Science (USA), 2003

Morgan Kaufmann Publishers

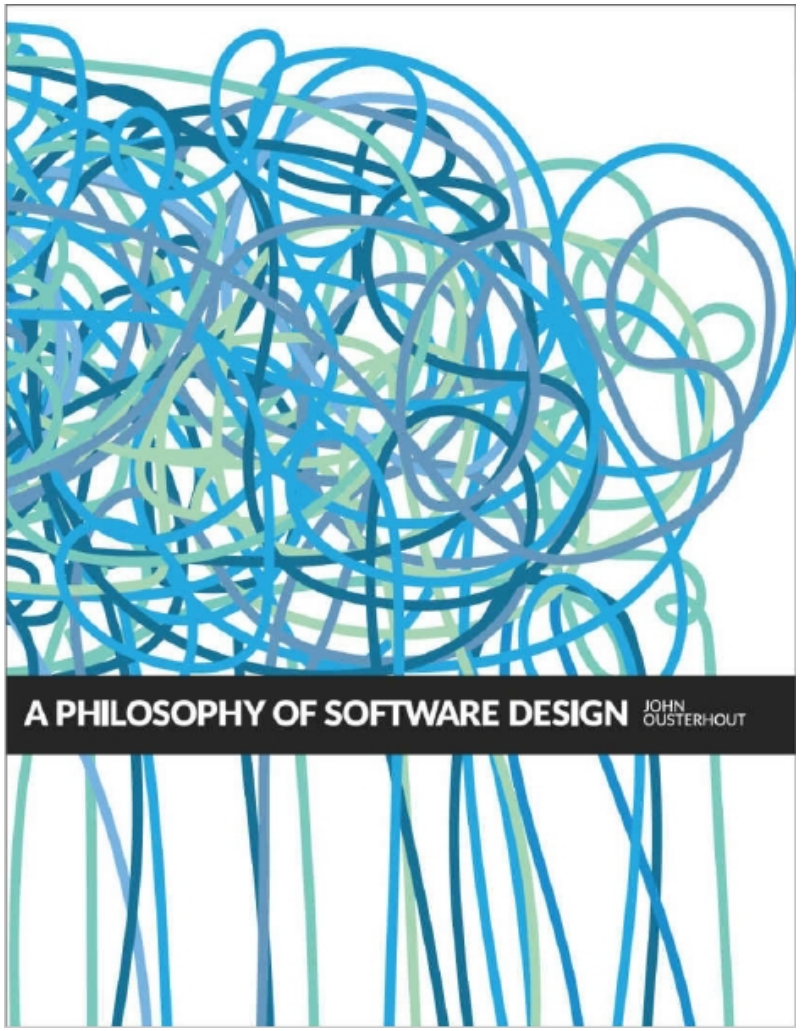


Reference Textbook

John Ousterhout.

A Philosophy of Software Design

Springer, 2014



Reference

Ivar Jacobson, Pan-Wei Ng, et la.

The Essence of Software Engineering: The SEMAT Kernel. Communications of the ACM, 2012.12

Course Description

- The key objective of this course is to learn modular design of software and documenting the design using symbolic representations, i.e., UML diagrams. The course will cover software life-cycle models and different phases of the software development process.
- Object-oriented techniques are key to the course. Since the ultimate result of software engineering is a working software package, the course will put a great emphasis on developing a demonstrable software package. However, this is not a programming course.

Course Description (cont.)

- The key characteristic is having three stages of project works: individual work (Programming), teams of two to three students work on developing complex software systems over a course of one semester.

Who is this course for?

- An introductory Foundations of Software Engineering and Practical Training on Software Engineering through Team Projects.
- Prepare to Design Solutions (Problem-Solving) for real world pragmatic problems
- Prerequisites: solid programming skills, algorithms and data structures
- Credit: 3 credits

Grading

- Exercise Homework and Class Participation: 15%
- Project: 25%

  Project reports (2 Projects, 4 version of reports): 25 + 10

- Exam: 60% (Closed-book, from projects, Lectures & Exercises)

Syllabus at a Glance (ch1~ch5 only, +...)

Topic 1 (3hrs): Software Lifecycle

Topic 2 (3hrs): Requirements Engineering and Use Cases

Topic 3 (2hrs): Software Architecture

Topic 4 (4 hrs): Object-Oriented Analysis

Topic 5 (4 hrs): Object-Oriented Design

Topic 6 (2 hrs): Test-Driven Implementation

Topic 7 (3 hrs): System Specification

Topic 8 (6 hrs): Problem Frames & Software Measurement

Topic 9 (6 hrs): Design Patterns

Contents for the first 4 weeks (negotiable)

教学周	学时安排	学习单元名称	主要教学内容
第一周	第 1-3 学时	0.Course Introduction 1.Intro.to Software Engineering(SE)	1. Course Syllabus with Teaching Methods 2. Software: Its Nature and Qualities 3. Complexity of Software and SE
第二周	第 4-6 学时	2.Basics of OO methods 3.Basics of Requirement Engineering	1. Design Science Methodology 2. Object Models 3. Intro. To Requirement Engineering
第三周	第 7-9 学时	4.Requirment Description	1. Concepts of Reactive Systems 2. Use case Models 3. Domain Models
第四周	第 10-12 学时	5.Implementation I	1. Sample Project Case Studies 2. Basic of OO design 3. Basics of Software Testing

Writing, writing, writing…

✓Good Engineers must be good writers.

✓This course is a **W (writing)** course. You will be writing **a lot** in this class, perhaps more than in any other college course (☺).

✓Homework and Project assignments will ask you to **write** and to **rewrite**. The project will involve even more **extensive** writing, subject to a **four-step revision cycle**: **draft requirements - specification with architecture - first rewrite - second rewrite**.

✓In addition to project-related **documents** you will be writing weekly **progress reports** and the **final project write-up**.

✓Your homework assignments and project documents will be graded not only on the subject matter, but also on correctness and writing (expression) qualities.

✓The exam will include **drawing UML models** and **writing essays**.

Writing, writing, writing…

3 key points:

- Understanding
- Expressing
- What Really Matter?

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

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

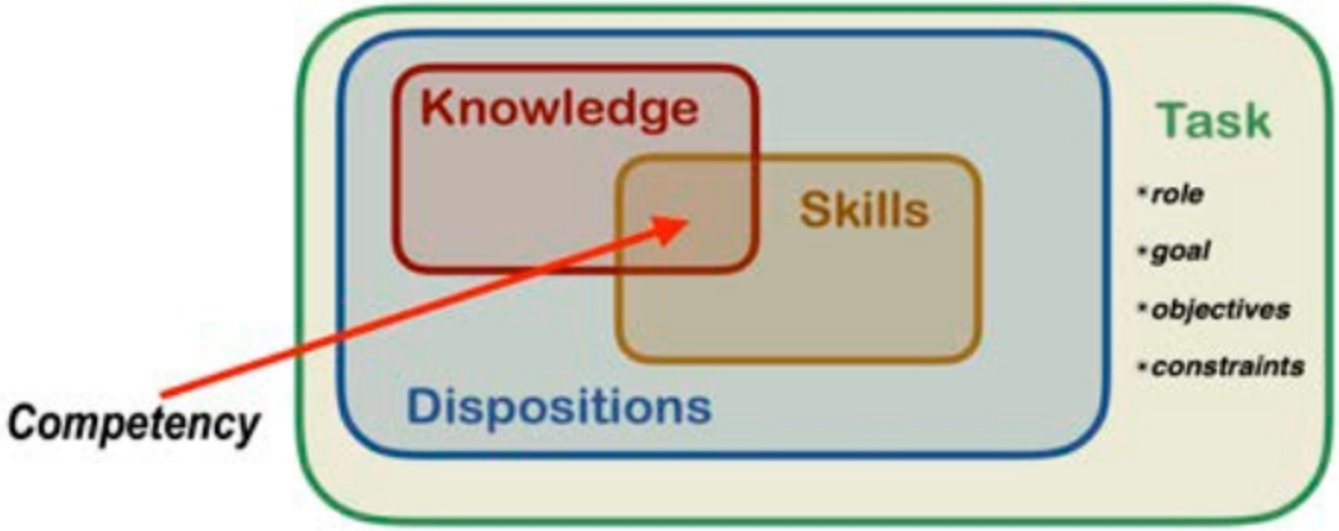


Figure 4.1. Conceptual Structure of the CC2020 Competency Model

	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

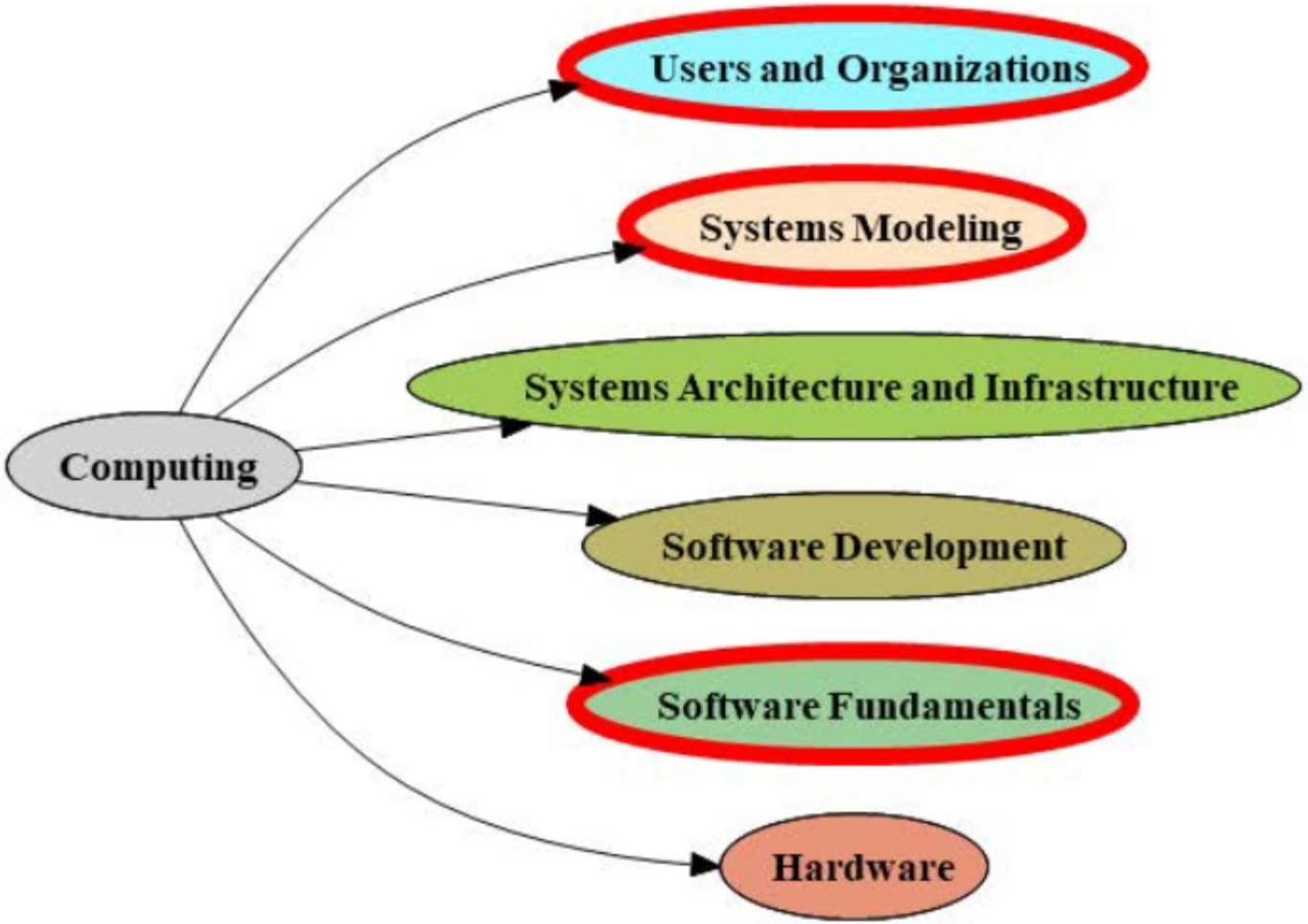


Figure 5.2. The student's choice of computing categories

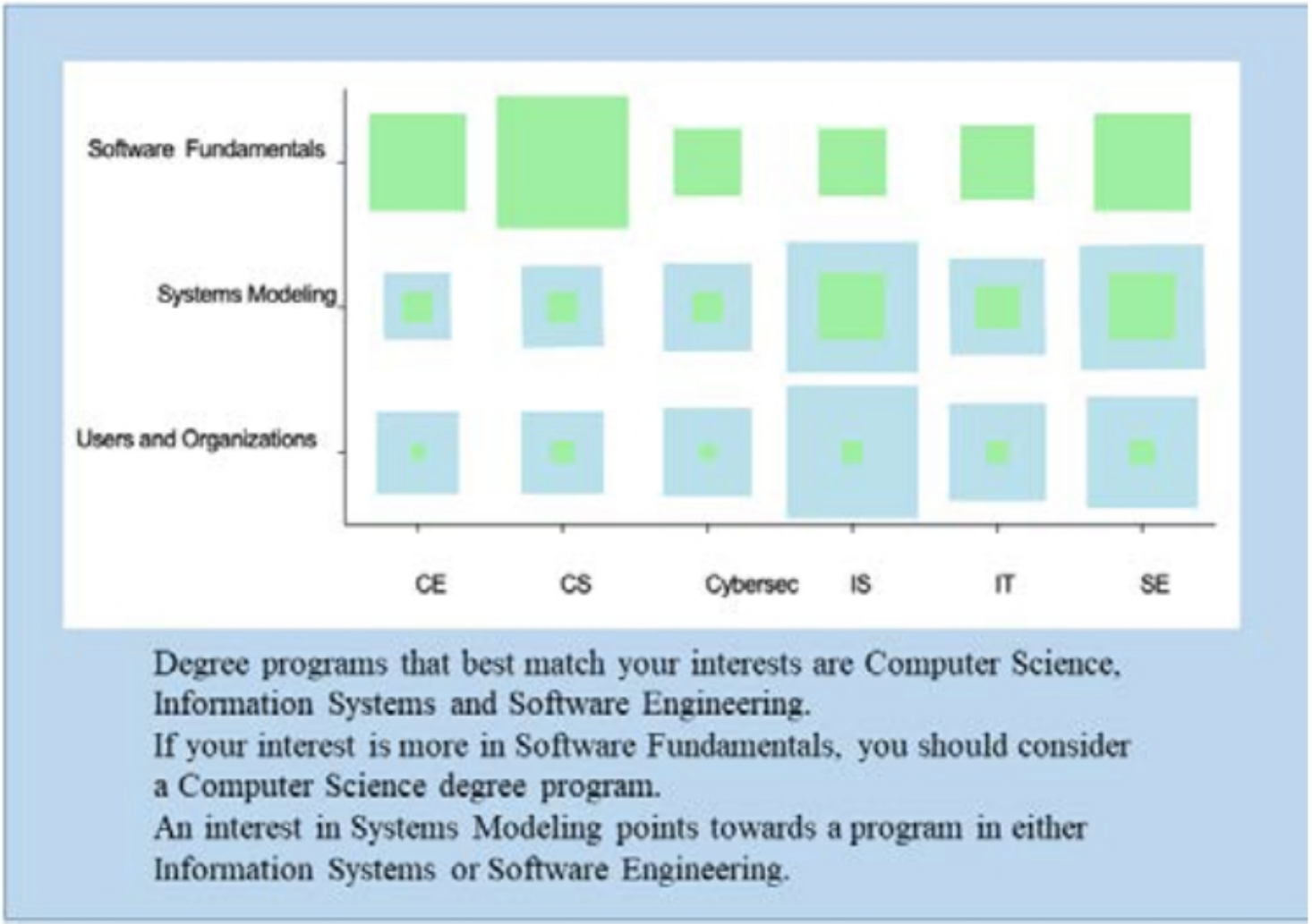


Figure 5.4. Mapping of chosen knowledge categories to the six curricular guidelines

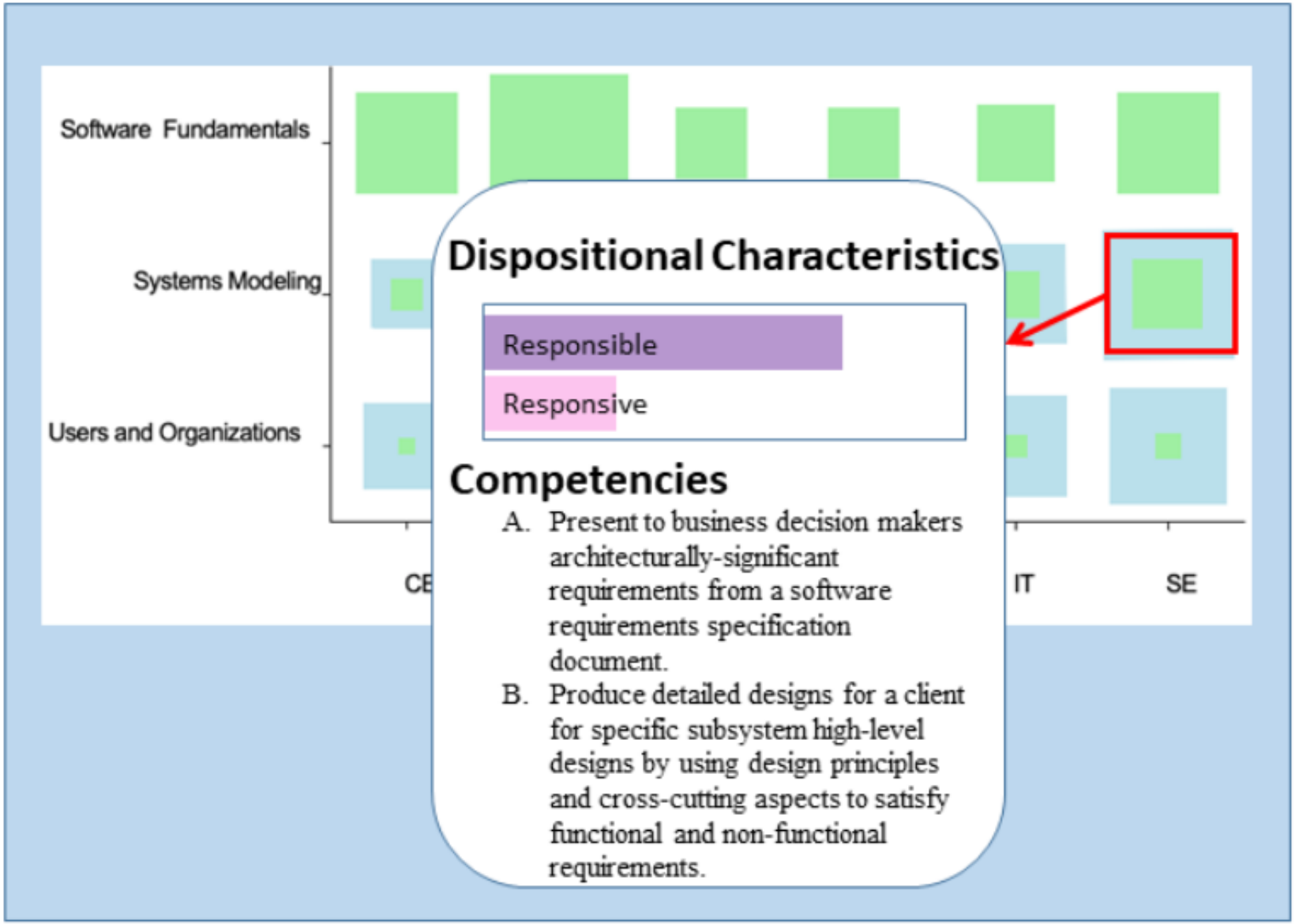


Figure 5.5. Disposition and competency details

Three Dimensions of most CST Courses

Most of Courses on CST is organized by a combination of elements from the following 3 dimensions:

THEORY related: Maths, Concepts, Models, 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, set and use them properly

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

Put them all together!!!

Guideline on Collaboration: What's OK and What's Not OK

Legend

OK	Permitted without restrictions or prior permission.
Ask	Permitted *only* with explicit instructor consent.
Forbidden	Forbidden under any circumstances.

Policies

Consult class textbook or assigned readings	OK
Search the internet on the topic of the assignment for basic definitions, terminology, etc.	OK
Look at/search for solutions to specific problems from an assignment	Forbidden
Look at/search for solutions to similar problems to those from an assignment	Forbidden
Discussions* with classmates	OK
Discussions with people outside of class	Forbidden
Explicit help from classmates in writing solutions	Forbidden
Explicit help from people outside class in writing solutions	Forbidden
Proofreading from classmates	Forbidden
Proofreading from outside class	Forbidden
Incorporate text from other sources	Forbidden
Incorporate figures/graphics from other sources	Forbidden
Reuse material from a previous or concurrent class that you took or are taking	Ask
Reuse material from a previous or concurrent class that somebody else took or is taking	Forbidden

*Discussions should be limited to small groups (2-3 people). If a solution or partial solution is reached during a discussion, each participant is individually responsible for writing down his/her solution based on their own understanding (so for example, transcribing a communally solved proof from some writing tools and then later using these notes to write your own solution is NOT okay). All collaborators on a given assignment should be noted at the beginning of the write-up.