

TIE-21100 Ohjelmistotuotannon menetelmät

TIE-21106 Software Engineering Methodology

Kari Systä
2014

The staff

- **Kari Systä**, lectures
- **Tero Ahtee**, practicalities and bureaucracy, overall coordination, weekly exercises, project
- **Marko Leppänen**, weekly exercises
- **Marie-Elise Kontro**, weekly exercises, project

Lecture 1 - introduction

Course

- Learning goals, requirements, assumed background of students
- Practical arrangements
- Material
- Position of the course in our curriculum

Software engineering

- What is Software engineering anyways?

Goals of this lecture

- This course
 - What to expect
 - Am I in right course
 - How to pass
- Software Engineering
 - What is Software Engineering

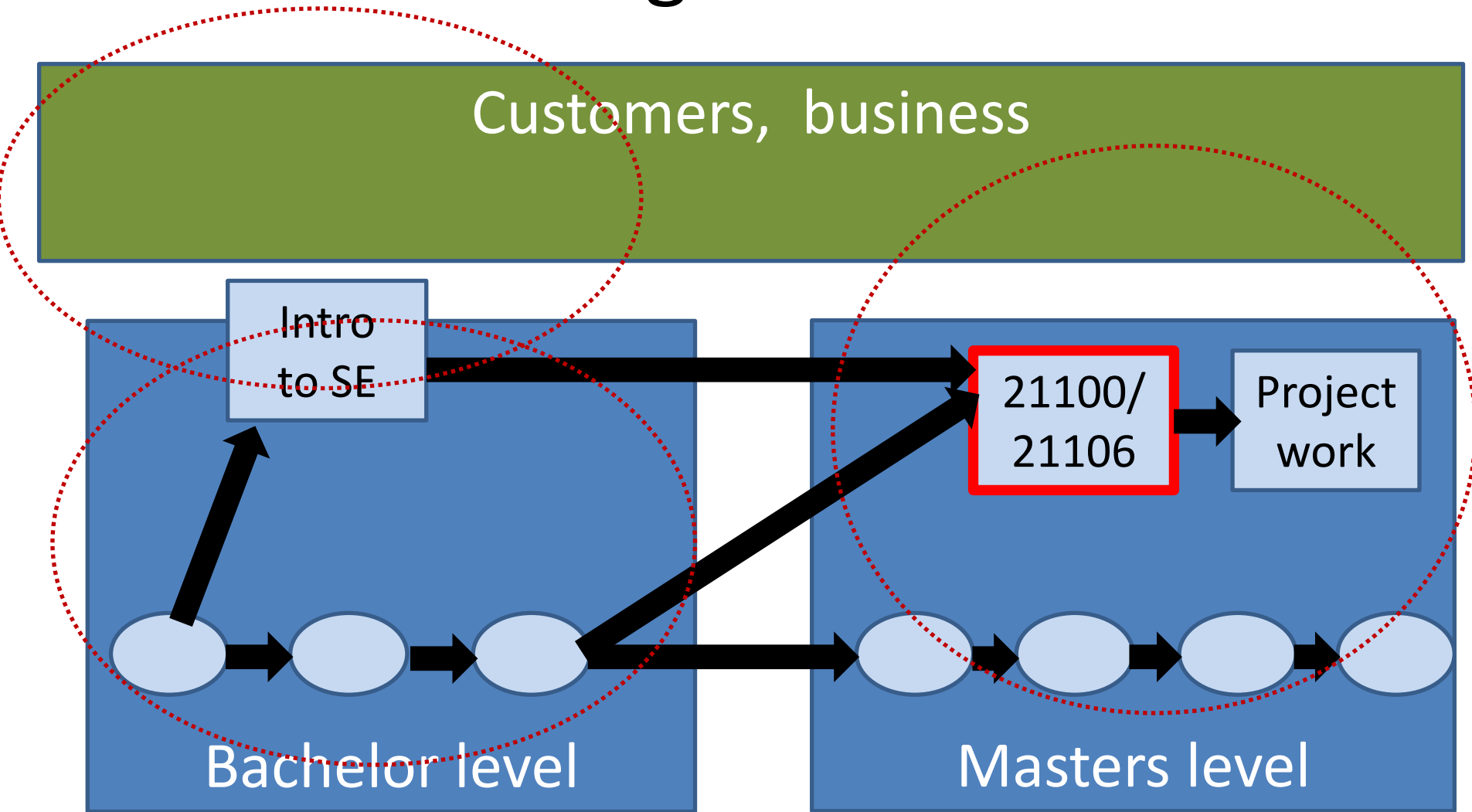
Learning goals

- ROCK/POP
 - Basic skills needed in software project work in different roles
- Content
 - Planning and running of software projects. Effort estimation and tracking techniques. Quality systems.
 - Software life-cycle models, their background, benefits and drawbacks.
 - Software quality, maintainability, usability, dependable systems.
 - Supporting activities: version- and configuration management, requirements management, product management, documentation

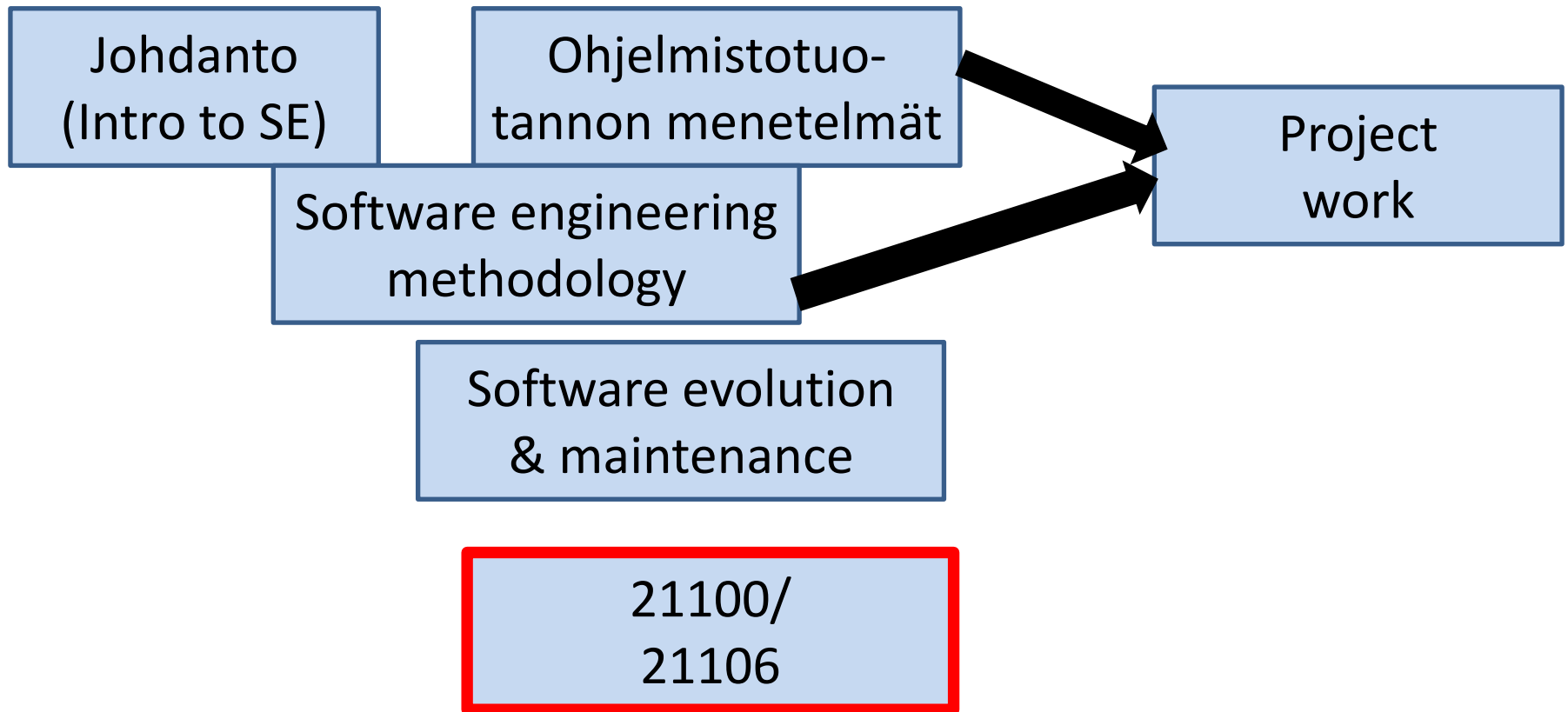
This is not a programming course

- But understanding of challenges in programming and basic programming skills are needed
 - Our project will make you to do simple programming
- Programming and SW design skills are appreciated, but we have different courses for that.

Positioning of this course



A bit of history of this course



Position in study modules

- Pervasive Computing
 - Especially for Software Engineering specialization (4th year)
- Ohjelmistotuotannon menetelmät/
Software Engineering
 - Compulsory (3rd year)
- Ohjelmistotekniikka/
Software Systems
 - Complementary course

A quick background survey

- Major of your bachelor degree?
 - Telecom
 - Digital and computer technology
 - Electrical engineering
 - Management, business or production technologies
 - Other
- OtuPK/JOTU fall 2013?

A quick background survey

- How many have programmed for salary?
- Who has been member of a programming team of more than 3 persons?
- Who has used version management tools?
 - SVN?
 - GIT?
 - Other?

Learning goals rephrased i.e. the relevance

- Organization of the SW development
 - Why important, why difficult?
 - Learn to plan and organize
- What (in addition to how)
 - Make SW that clients and users need and want?
- Quality
 - What it is and how to achieve?
- Business
 - How much SW development costs – and why

Buzz works

Agile, Scrum,
Requirements, Waterfall,
Quality Systems,
COCOMO, CMM, UML,
RUP, XP,

Practical arrangements

- Lectures
 - Mondays, TB109, 1415 – 1600
 - Thursdays, TB103, 1015 – 1200

Most lectures will be on Mondays, Thursdays are used when we have quest lectures
- Weekly exercises
 - Some practical hand's on
- Project
 - A simple SW project to be planned, organized, executed and reported

Weekly exercises

by Marie-Elise Kontro, Tero Ahtee and Marko Leppänen

- We start from 8 groups, but most probably the least popular will be discontinued
- Times
 - TUE 10:15 - 12:00 TC131 11
 - TUE 12:15 - 14:00 TC131 16
 - WED 10:15 - 12:00 TC163 08
 - WED 12:15 - 14:00 TC131 15
 - WED 14:15 - 16:00 TC128 11
 - **WED 16:15 - 18:00 TB207 05**
 - THU 12:15 - 14:00 TC163/TC131 20
 - THU 14:15 - 16:00 TC163 13
- Please sign-up if you haven't already!

First weekly exercise

- Read the article:
<http://www.cs.nott.ac.uk/~cah/G51ISS/Documents/NoSilverBullet.html> before
- Think about the following questions
- What is "silver bullet"?
- What makes SW development so difficult – according to article?
- What are the benefits of incremental SW development?

Project

A simple SW project is

- Planned
- Requirements are defined and prioritized
- Executed in 3 or 4 "sprints"

The work includes effort estimation, tracking and reporting

At the end we will have a seminar where

- Teams represent their project and end result
- Winner(s) is/are selected

The staff

- **Kari Systä**, lectures
- **Tero Ahtee**, practicalities and bureaucracy, overall coordination, weekly exercises, project
- **Marko Leppänen**, weekly exercises
- **Marie-Elise Kontro**, weekly exercises, project

Passing and grading

- Lectures
 - Not compulsory but will be useful for exam
- Exam
 - 24 points
- Weekly exercises
 - 6 points
- Project
 - 6 points

Initial content of lectures

- Introduction
- Life-cycle models, their background
- Project management, product management, project planning
- Scrum in details
- Kanban, Customer Development and DevOps details
- Requirement definition, requirement management, requirements prioritization
- Version management, configuration management, continuous integration
- Architecture issues, role of architect, architectural quality attributes, product families, (TIE-21300 will go deeper)
- Testing and quality assurance (TIE-21200 will go deeper)
- “Quality systems” and process improvement
- Embedded and real-time systems (other courses will go deeper)
- Safety-critical and dependable systems
- Effort estimation
- Software business, software start-ups
- Recap

Material

- Slides from lectures
- Haikala, Mikkonen: Ohjelmistotuotannon käytännöt
- Sommerville: Software engineering, Ninth edition
- Links will be collected to the web page.

Key drivers for redesign of this course

- Agile methods are dominant in the industry
 - But far too often not understood
- Compared to old times, majority of our students are employed by SME instead of big companies
 - Nokia
- One key reason for failed SW projects is poor customer – developer discussion
- Internet and Cloud influence everything
- Changes in other courses => this course needs to synchronize with those changes
- International version (TIE-21106) should not be a specific version
 - We give and demand the same to and from all

What is Software Engineering

A few views

A few definitions

- “ Software engineering may be defined as the systematic design and development of software products and the management of the software process”
 - Mills, H.D. , IBM Systems Journal
Vol19 , Issue: 4, 1980
- “Software Engineering is the study and application of engineering to the design, development, and maintenance of software.”
 - http://en.wikipedia.org/wiki/Software_engineering

Ohjelmistokriisi (Software Crisis)

- Term was invented at the first NATO Software Engineering Conference in 1968
- Symptoms (then and still)
 - Projects running over-budget.
 - Projects running over-time.
 - Software was very inefficient.
 - Software was of low quality.
 - Software often did not meet requirements.
 - Projects were unmanageable and code difficult to maintain.
 - Software was never delivered.

A quote

- “The major cause of the software crisis is that the machines have become several orders of magnitude more powerful! To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming has become an equally gigantic problem.”
-- E. Dijkstra, 1972 Turing Award Lecture

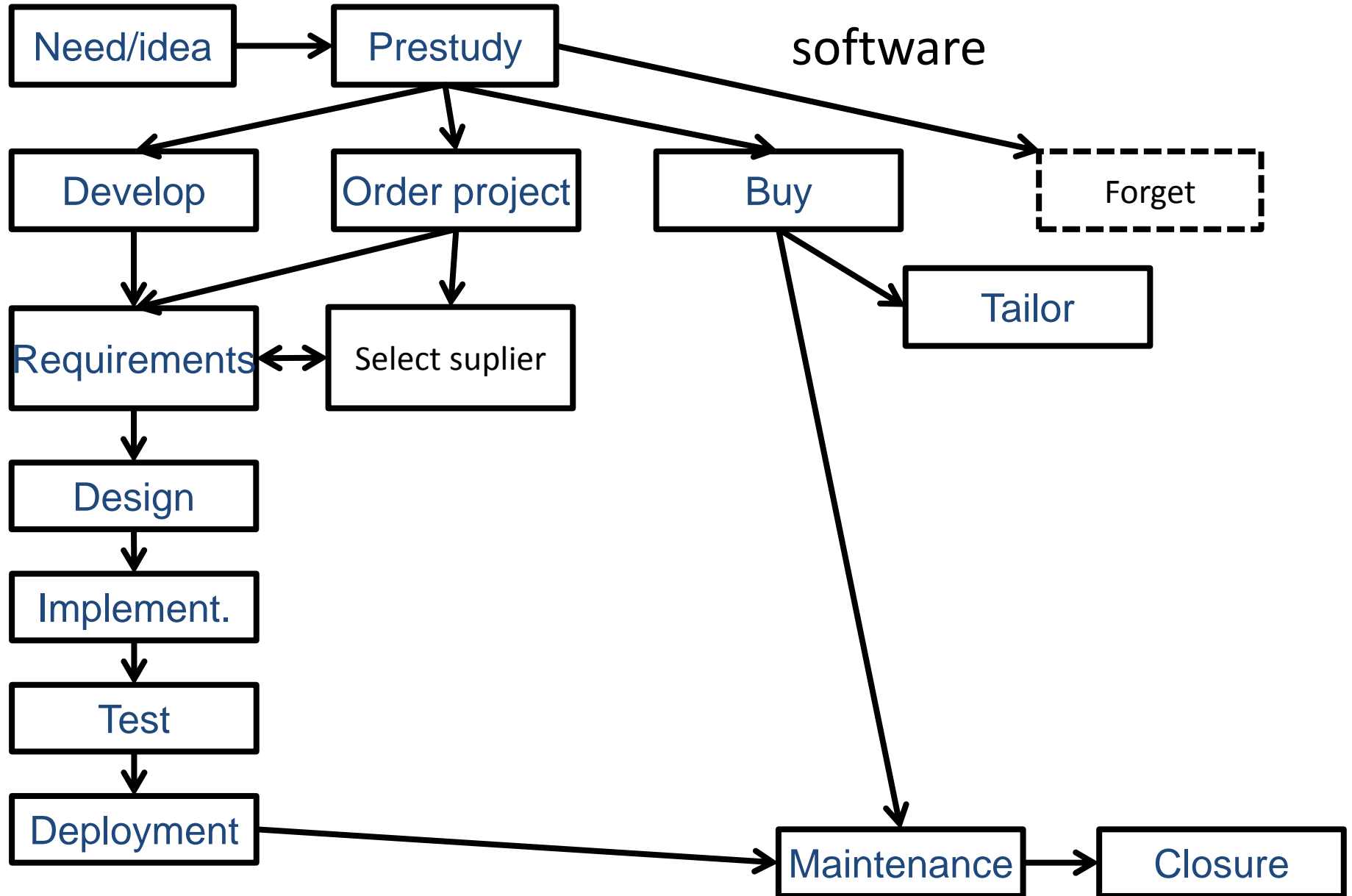
Three SW crisis

- 60-70's
 - Problem: Assembly programming
 - Solution: High-level programming languages (Fortran, C, Cobol)
- 80-90's
 - Problem: Development and maintenance of complex programs (millions of lines, many developer)
 - Solution: Libraries, object-oriented programming, architecture, testing, review practices
 - Solution: Good advanced design; extensive documentation
- 2000--
 - Problem: software does not meet real needs of the users
 - Solution: Agile methods

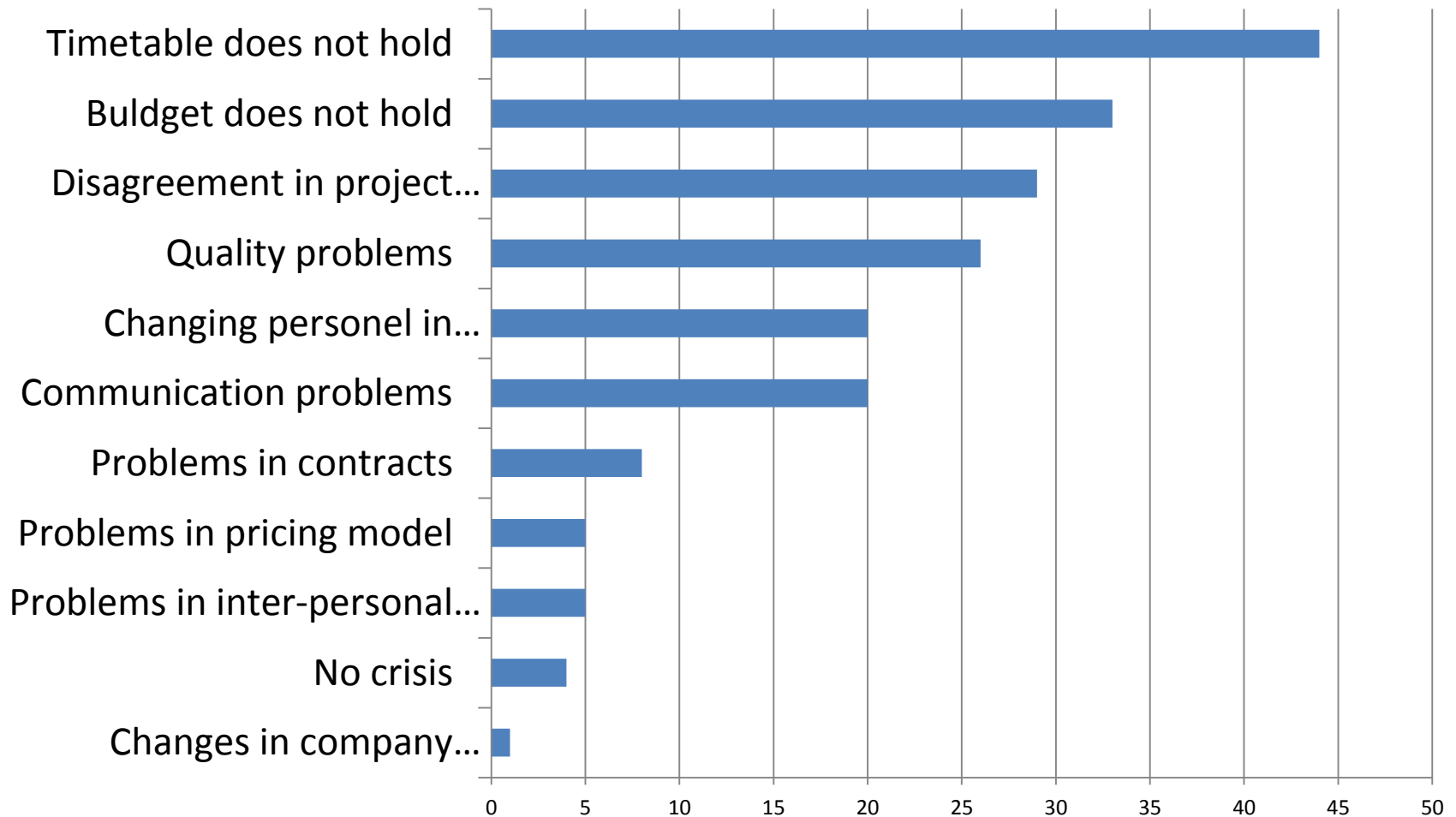
“Silver bullet”

- “There is no single development, in either technology or in management technique, that by itself promises even one order-of-magnitude improvement in productivity, in reliability, in simplicity.”
 - "No Silver Bullet — Essence and Accidents of Software Engineering" is a widely discussed paper on software engineering written by Fred Brooks in 1986.

From needs to
software



Problems – for customer point of view

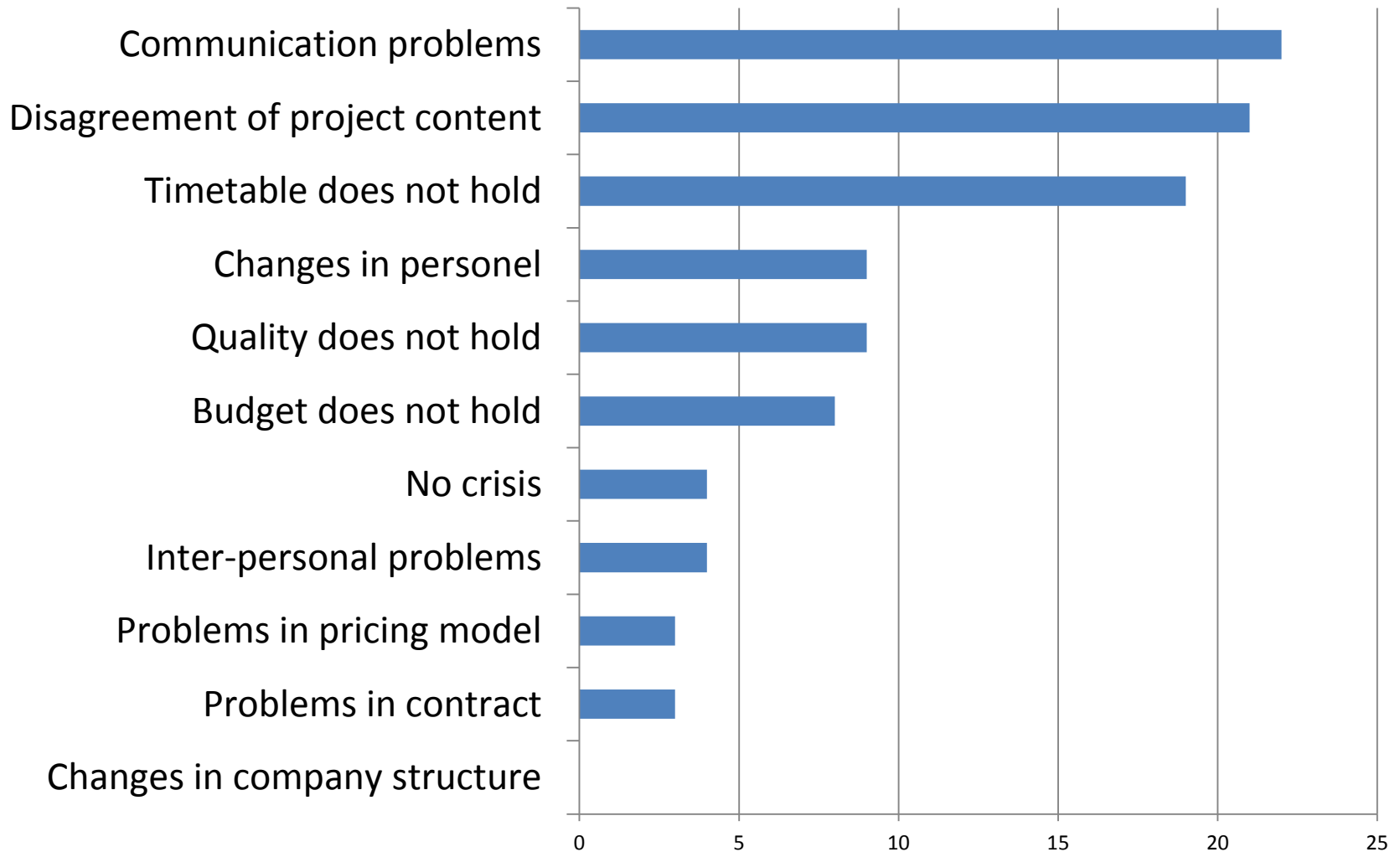


Lähde: tietotekniikan liiton, ohjelmistoyrittäjien ja Celkee OY:n tutkimus

2014-01-13

TIE-21100/21106

Problems – provider view



Lähde: tietotekniikan liiton, ohjelmistoyrittäjien ja Celkee OY:n tutkimus

Some definitions by Sommerville (p.6)

- **Good software** should deliver the required functionality and performance to the user and should be maintainable, dependable, and usable
- **Software engineering** is an engineering discipline that is concerned with all aspects of software production
- **Fundamentals of software engineering** are software specification, software development, software validation, and software evolution
- **Computer science** focuses on theory and fundamentals; **software engineering** is concerned with practicalities of developing and delivering useful software
- **System engineering** is concerned with all aspects of computer-based systems development ... software, hardware,
- Roughly 60% of the cost are development and 40% about validation. For custom software evolution costs often exceed development cost

Software Engineering Ethics

Software Engineering Ethics

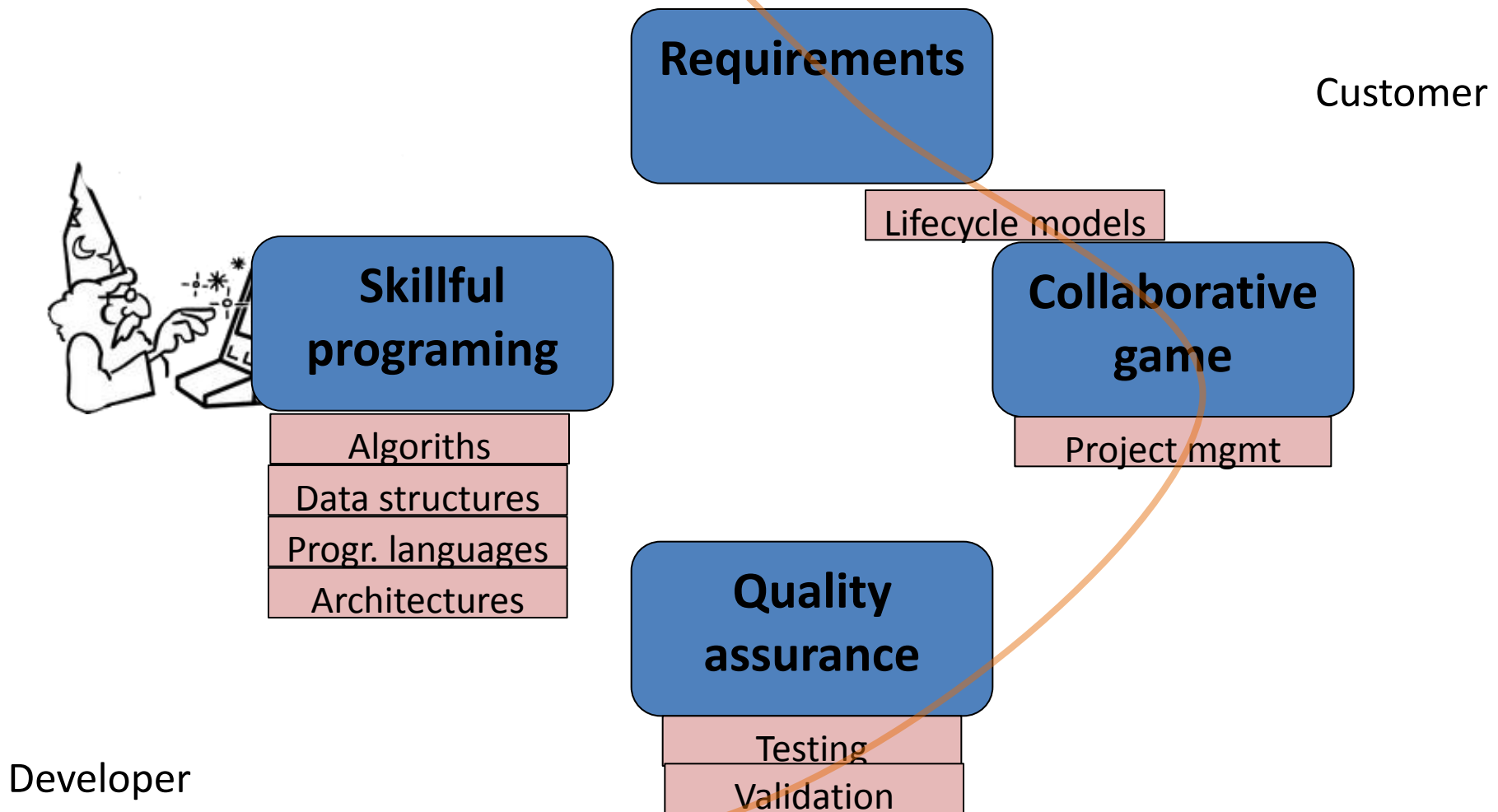
- Confidentiality
- Competence
- Intellectual Property Rights
- Computer Misuse

Software Engineering Code of Ethics and Professional Practice (Version 5.2) the ACM and the IEEE-CS as the standard for teaching and practicing software engineering.

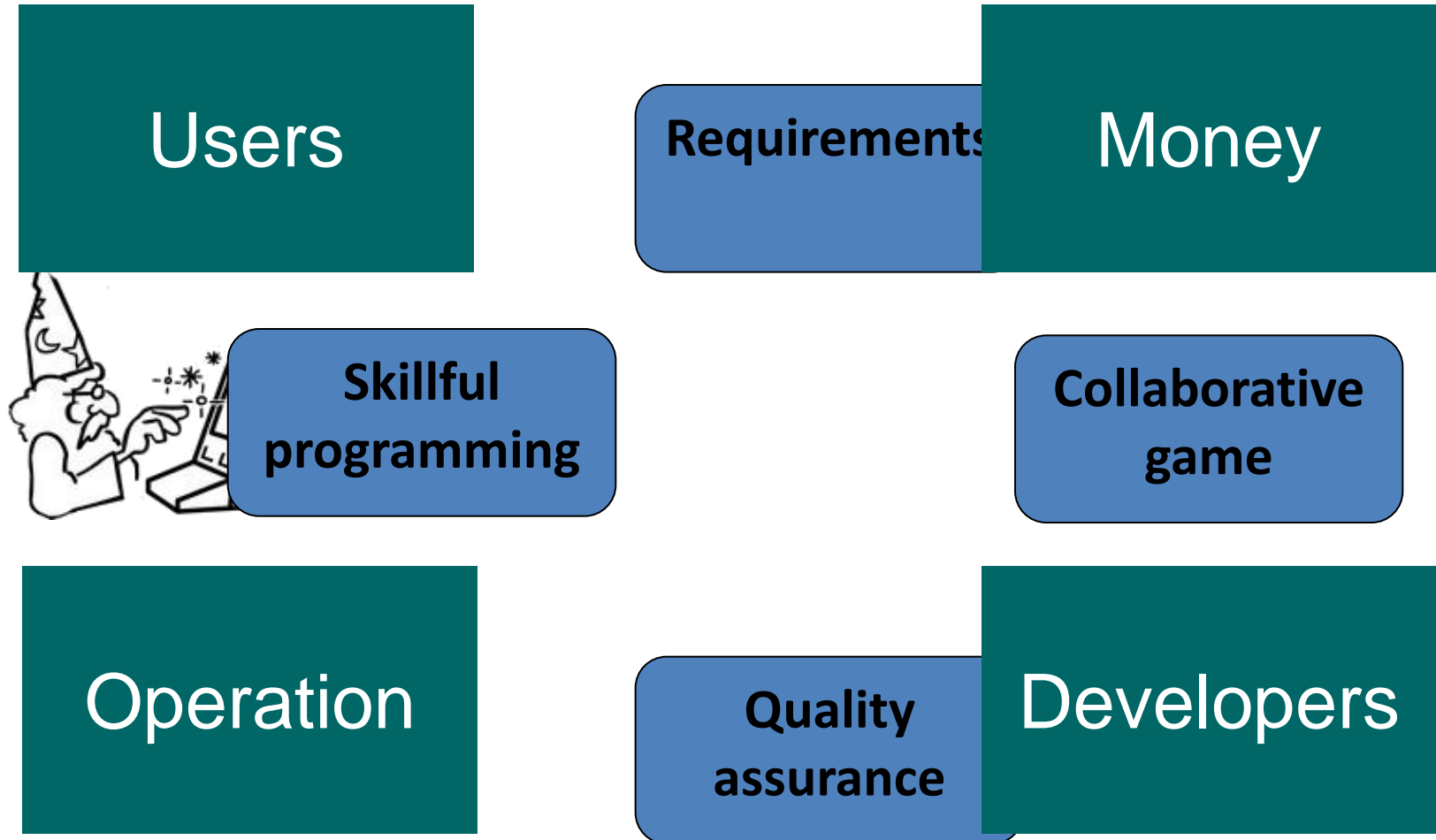
- See short and long version at
<http://www.acm.org/about/se-code>

- 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.

What is software engineering



Extended view



SWEBOK Guide V3.0 Topics

<http://www.computer.org/portal/web/swebok>

- **Chapter 1: Software Requirements**
- **Chapter 2: Software Design**
- **Chapter 3: Software Construction**
- **Chapter 4: Software Testing**
- **Chapter 5: Software Maintenance**
- **Chapter 6: Software Configuration Management**
- **Chapter 7: Software Engineering Management**
- **Chapter 8: Software Engineering Process**
- **Chapter 9: Software Engineering Models and Methods**
- **Chapter 10: Software Quality**
- **Chapter 11: Software Engineering Professional Practice**
- **Chapter 12: Software Engineering Economics**
- **Chapter 13: Computing Foundations**
- **Chapter 14: Mathematical Foundations**
- **Chapter 15: Engineering Foundations**

Software requirements

Key areas

- Fundamentals and definition
- **Requirements Process and management**
- Requirements Elicitation; sources and techniques
- Requirements Analysis
- Requirements Specification and documentation
- **Requirements Validation**
- **Practical Considerations, change tracing etc**
- Software Requirements Tools

Basics have been covered in introduction course, here we go a bit deeper and concentrate on bold topics

Sommerville Chapter 4, Haikala&Mikkone Chapter 3

Software design and construction

- Examples of topics
 - Languages
 - Concurrency
 - Data bases
 - Architectures
- We have plenty of other courses – not the focus in this course – we assume student to know quite a lot already
- Sommerville Chapters 6 and 7. Haikala & Mikkonen Chapter 14

Software Testing

Topics

- Software Testing Fundamentals
- Test Levels, targets and objectives
- Test Techniques
- Test-Related Measures
- Test Process
- Software Testing Tools

A separate course devoted, but in this course we discuss about process and connection to overall process.

Sommerville Chapter 8, Haikala&Mikkonen Chapter 16

Software Maintenance

Topics

- Software Maintenance Fundamentals
- Key Issues in Software Maintenance, technical, management, cost, measurement
- Maintenance Process and activities
- Techniques for Maintenance, Program, Comprehension, Reengineering, Reverse Engineering, Migration, Retirement
- Software Maintenance Tools

Since evolution course was discontinued, we will basic cover maintenance and evolution in this course.

Sommerville Chapter 9, Haikala&Mikkonen mentions in several places

Software Configuration Management

Topics

- Management of the SCM Process
- Software Configuration Identification
 - Identifying Items to Be Controlled
 - Software Library
- Software Configuration Control
 - Requesting, Evaluating, and Approving Software Changes
 - Implementing Software Changes
 - Deviations and Waivers
- Software Configuration Status Accounting
- Software Configuration Auditing
- Software Release Management and Delivery
 - Software Building
 - Software Release Management
- Software Configuration Management Tools

One lecture, Chapter 25 in Sommerville, Chapter 13 Haikala&Mikkonen

Software Engineering Management

Topics,

- Initiation and Scope Definition. Feasibility analysis
- Software Project Planning
- Software Project Enactment/implementation, monitoring
- Review and Evaluation
- Closure, Determining Closure, Activities
- Software Engineering Measurement
- Software Engineering Management Tools

In this course one lecture devoted, also discussed in other lectures. Sommerville Chapters (18,) 22-26.

Haikala&Mikkonen: Chapter 12

Software Engineering Process

Topics

- Software Process Definition – management and infrastructure
- Software Life Cycles , Categories, Models, Adaptation
- Software Process Assessment and Improvement
- Software Measurement
- Software Engineering Process Tools

Especially the life-cycle models are essential content for this course. Sommerville chapters 2, 3 (and 18), Haikala&Mikkonen Chapter 2.

Software Engineering Models and Methods

Topics

- Modeling
- Types of Models
- Analysis of Models
- Software Engineering Methods

Many notations have been discussed in other courses, but general concepts will come in this course, too.

Sommerville Chapter 5, Haikala&Mikkonen
Chapters 4-10 (with requirement view point)

Software Quality

Topics

- Software Quality Fundamentals, Culture, Ethics, value and cost
- Software Quality Management Processes
- Practical Considerations
- Software Quality Tools

One lecture at least – but also covered in other courses. Sommerville Chapter 24, Haikala&Mikkonen Chapter 11

Remaining topics

- **Software Engineering Professional Practice (ethics, legal, communication,...)**
 - Some elements will be included
- **Software Engineering Economics**
 - Some elements will be included
- **Computing Foundations,
Mathematical Foundations,
Engineering Foundations**
 - Not in this course – there are other courses

Goals of this lecture

- This course
 - What to expect
 - Am I in right course
 - How to pass
- Software Engineering
 - What is Software Engineering
 - Chapters 1 in both books

First weekly exercise

- Read the article:
<http://www.cs.nott.ac.uk/~cah/G51ISS/Documents/NoSilverBullet.html> before
- Think about the following questions
- What is "silver bullet"?
- What makes SW development so difficult – according to article?
- What are the benefits of incremental SW development?

Links to material

- *Software Engineering Body of Knowledge:*
<http://www.computer.org/portal/web/swebok>
- *Ethics* <http://www.acm.org/about/se-code>
- *Silver bullet (e.g.)*
<http://www.cs.nott.ac.uk/~cah/G51ISS/Documents/NoSilverBullet.html>
- *Chapter 1 in both books*