Software Engineering I: Software Technology

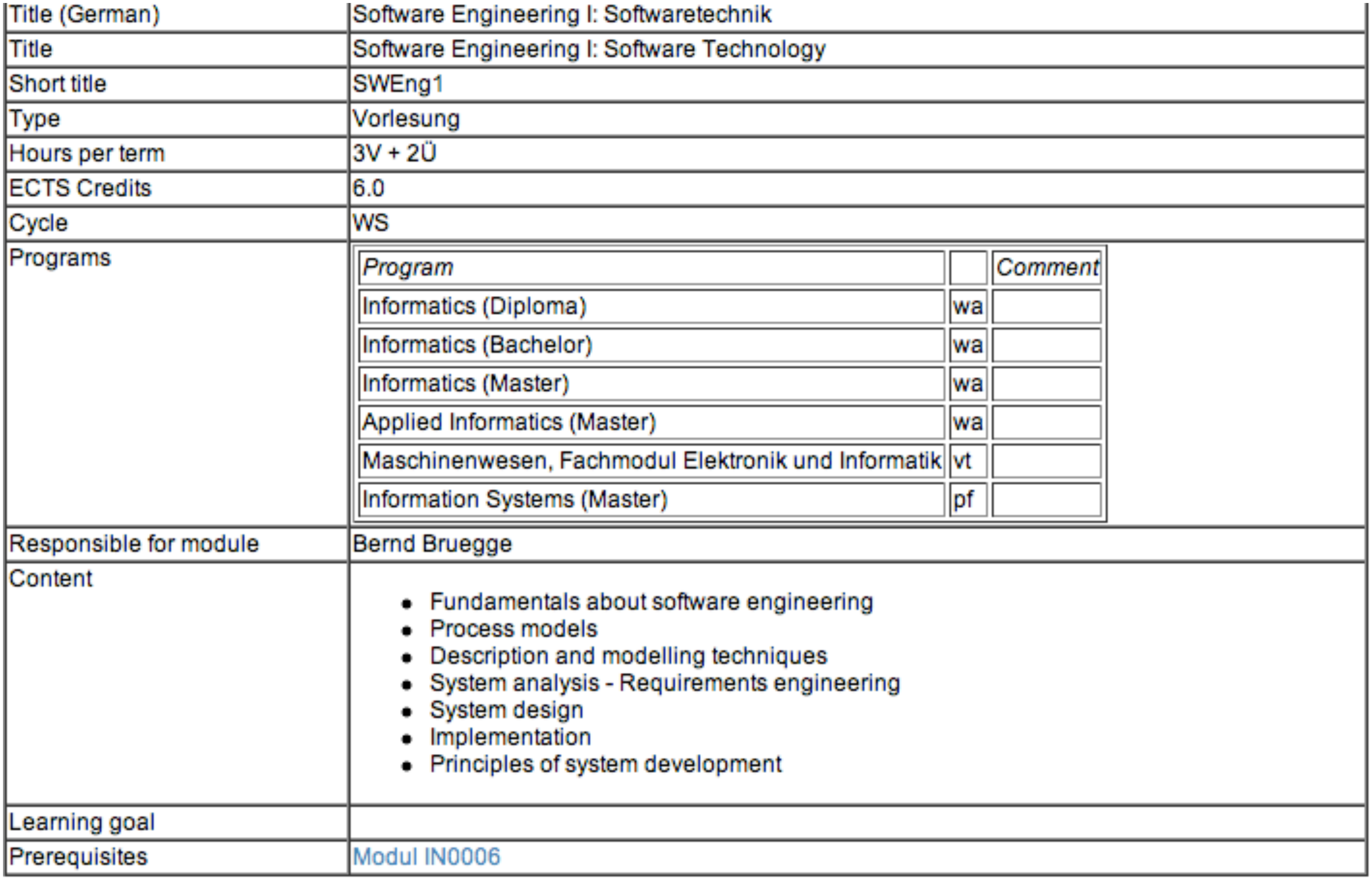
WS 2008/9

Introduction

Prof. Bernd Bruegge, Ph.D.
Applied Software Engineering
Technische Universitaet Muenchen

Module Description

- Software Engineering I: Software Technology
- http://www.in.tum.de/myintum /kurs_verwaltung/c.html?cid=881



<table>
<tr><td>Title (German)</td><td>Software Engineering I: Softwaretechnik</td></tr>
<tr><td>Title</td><td>Software Engineering I: Software Technology</td></tr>
<tr><td>Short title</td><td>SWEng1</td></tr>
<tr><td>Type</td><td>Vorlesung</td></tr>
<tr><td>Hours per term</td><td>3V + 2Ü</td></tr>
<tr><td>ECTS Credits</td><td>6.0</td></tr>
<tr><td>Cycle</td><td>WS</td></tr>
<tr><td>Programs</td><td>
<table>
<tr><td><i>Program</i></td><td></td><td><i>Comment</i></td></tr>
<tr><td>Informatics (Diploma)</td><td>wa</td><td></td></tr>
<tr><td>Informatics (Bachelor)</td><td>wa</td><td></td></tr>
<tr><td>Informatics (Master)</td><td>wa</td><td></td></tr>
<tr><td>Applied Informatics (Master)</td><td>wa</td><td></td></tr>
<tr><td>Maschinenwesen, Fachmodul Elektronik und Informatik</td><td>vt</td><td></td></tr>
<tr><td>Information Systems (Master)</td><td>pf</td><td></td></tr>
</table>
</td></tr>
<tr><td>Responsible for module</td><td>Bernd Bruegge</td></tr>
<tr><td>Content</td><td>
<ul>
<li>Fundamentals about software engineering</li>
<li>Process models</li>
<li>Description and modelling techniques</li>
<li>System analysis - Requirements engineering</li>
<li>System design</li>
<li>Implementation</li>
<li>Principles of system development</li>
</ul>
</td></tr>
<tr><td>Learning goal</td><td></td></tr>
<tr><td>Prerequisites</td><td>Modul IN0006</td></tr>
</table>

Intended audience

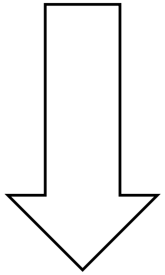
- Informatics (Master, Bachelor)
- Information Systems (Master)
- Applied Informatics (Master)
- Diploma Students (“alte Studienordnung”)
- Mechanical Engineering (Maschinenwesen): Fachmodul Elektronik und Informatik

Assumptions for Software Engineering I: IN 2126

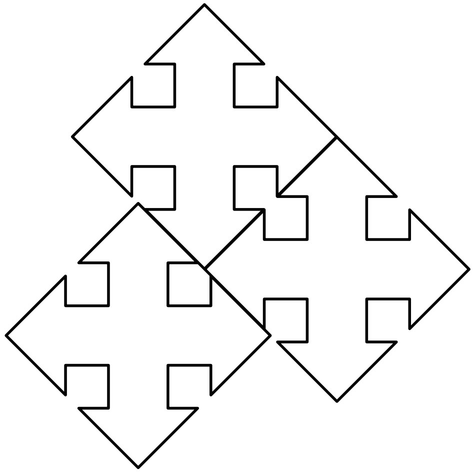
- Assumption:
  - You have taken Module 0006 EIST: Introduction to Software Engineering (Einführung in die Softwaretechnik)
  - You have already experience in at least one analysis and design technique
- Beneficial:
  - You have had practical experience with a large software system
  - You have already participated in a large software project
  - You have experienced major problems.

Objectives of the Lectures (IN 2126)

IN 0006



IN 2126



- Appreciate the Fundamentals of Software Engineering:
    - Methodologies
    - Process models
    - Description and modeling techniques
    - System analysis - Requirements engineering
    - System design
    - Implementation: Principles of system development

Times and Locations

- Main lecture: HS 2
  - Tuesdays 16:15 – 17:45
  - Fridays 9:15 - 10:00
- Exercises: Thursday 8:15-9:45
  - Exercise sessions start this Thursday , Oct 16
- You may register for the lectures and exercises:
  - Send e-mail with Subject: SE to florian.schneider@in.tum.de
    - Name, Matrikelnummer
- Preliminary Dates for Exams:
  - **Mid-term:** 18 Dec 2008, 8:00-10:00, HS2, Open Book
  - **Final exam:** 9 February 2009 , 8:00-10:00, Room to be announced, Closed Book

Grading Criteria

The final grade is the weighted average of the mid term (25%) and final exam grades (75%)

- To pass this course your final grade must be 4.0 or better
- Participation in the exercises is highly recommended, but not required (not an admission requirement for the final exam)
- Information about the exercises will be made available on the exercise portal (under construction)
  - https://teambruegge.informatik.tu-muenchen.de/groups/sews08/
- Hours per week: 3 hours (lecture) + 2 hour (exercises)
- ECTS Credits: 6.0.

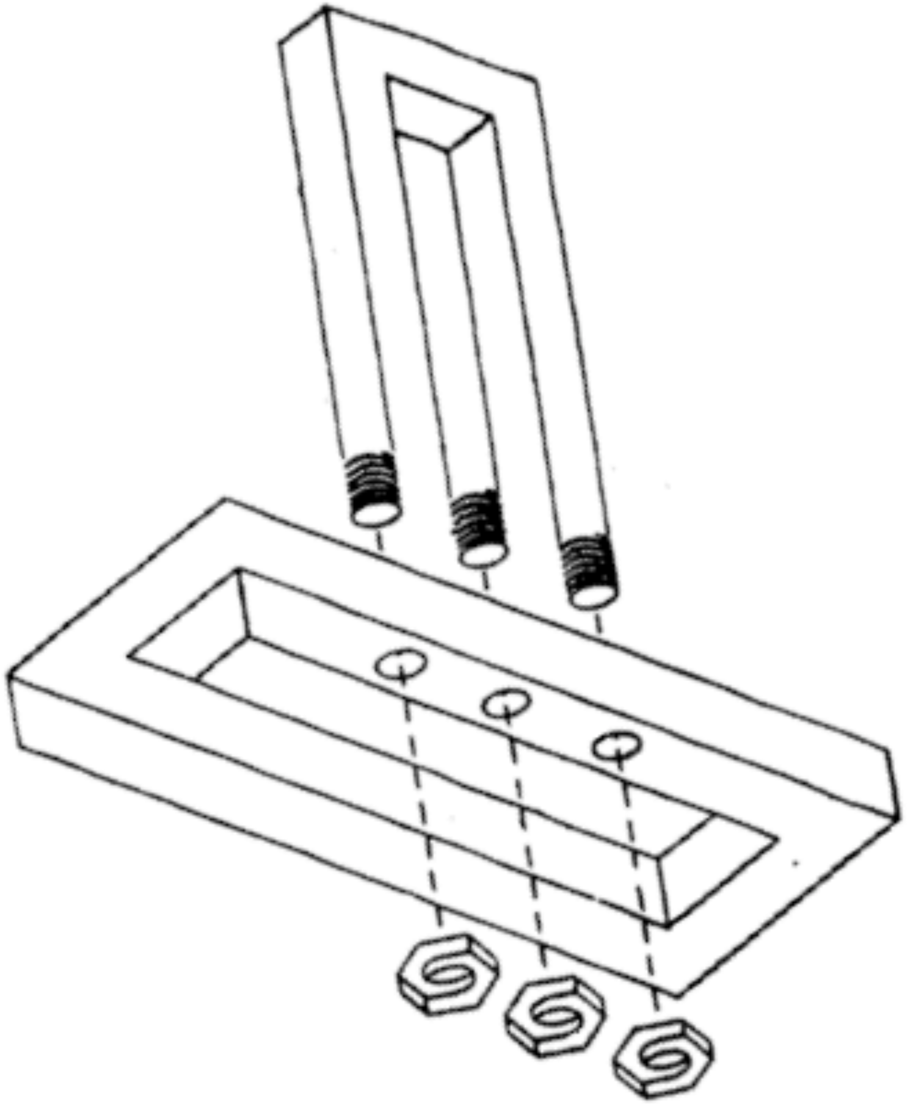
Focus: Technical Knowledge

- Understand different methodologies (“philosophies”) to model and develop software systems
- Be able to use different modeling notations
- Be able to use different modeling methods
- Understand various software lifecycle models (empirical control models, defined control models)
- Use different testing techniques (eg. horizontal testing, vertical testing)
- Understand Release & Configuration Management
- Knowledge Management (Rationale Management).

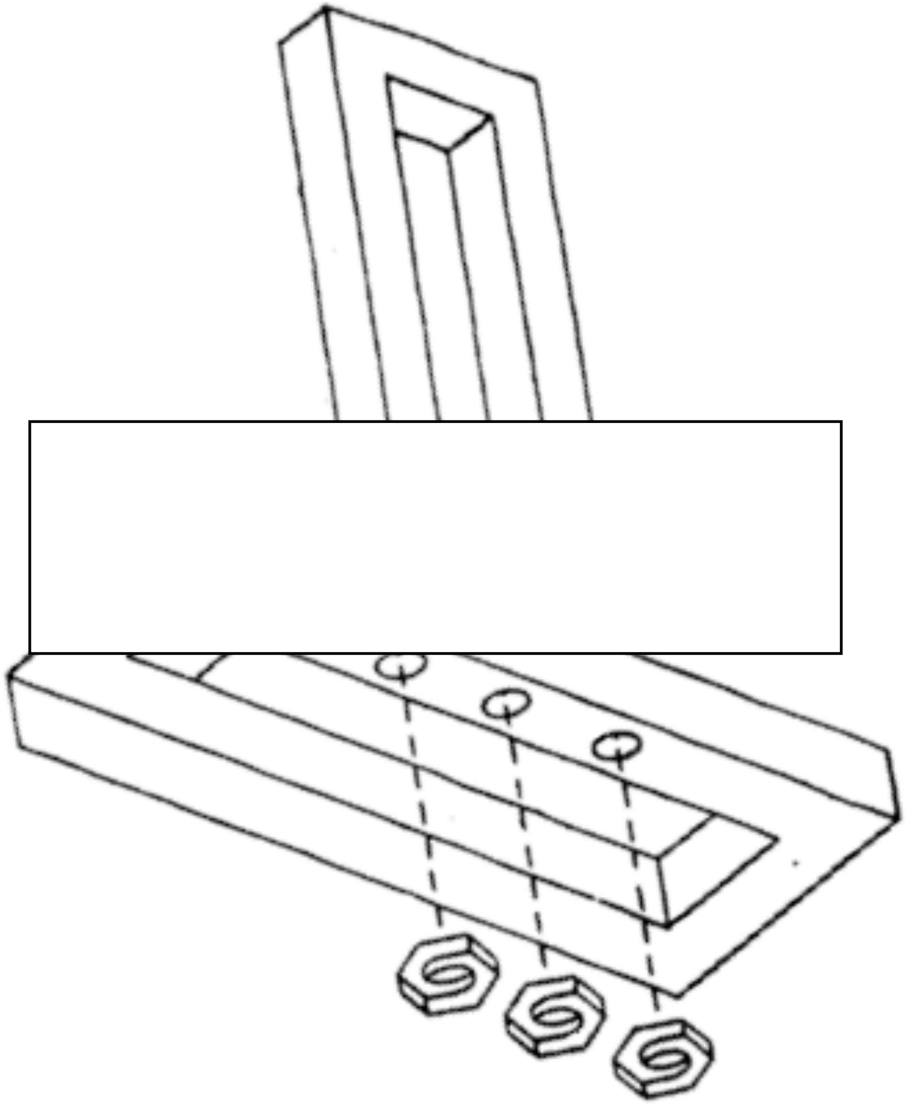
Outline of Today's Lecture

- ✓ Organizational issues
  - ✓ Scope of the lecture
  - ✓ Grading, Rooms, Times and Registration
- ➢ The development challenge
- Dealing with complexity and change
- Methodologies
- Organizational issues
  - Lecture schedule

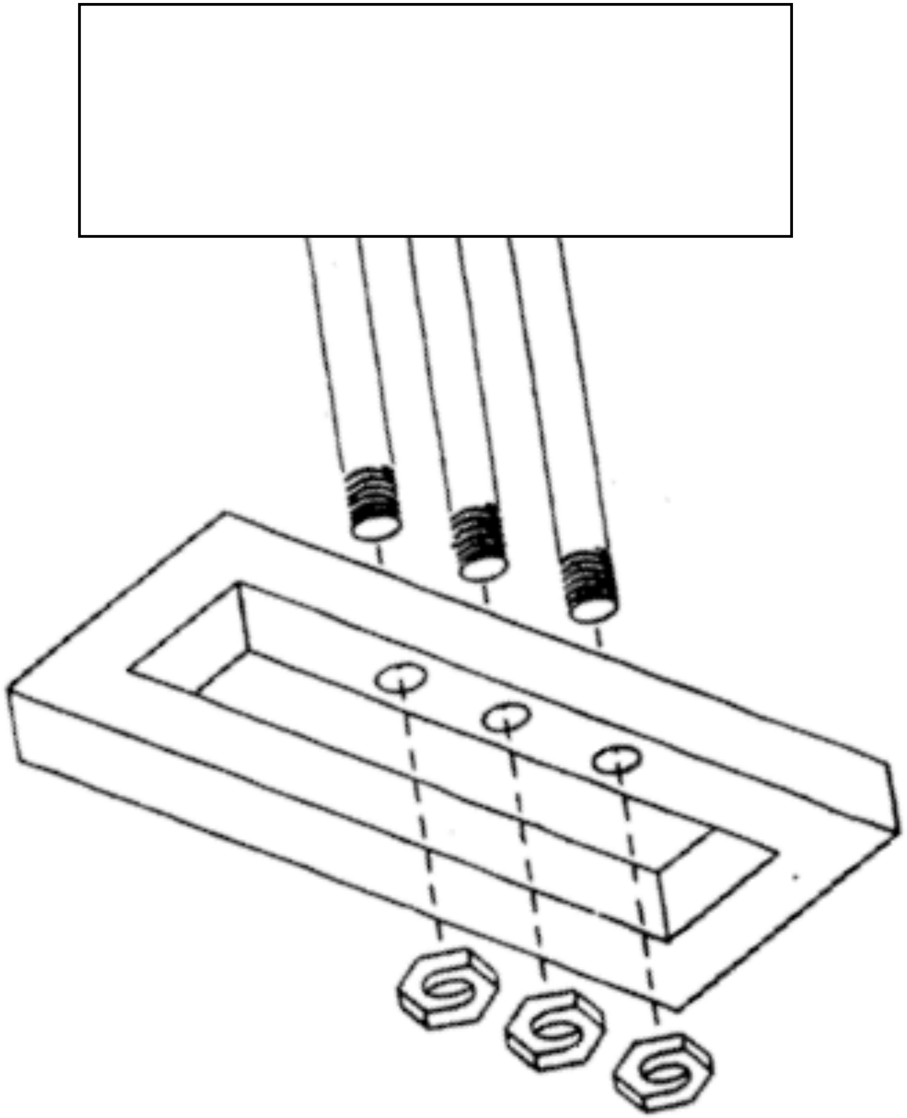
Can you develop this system?



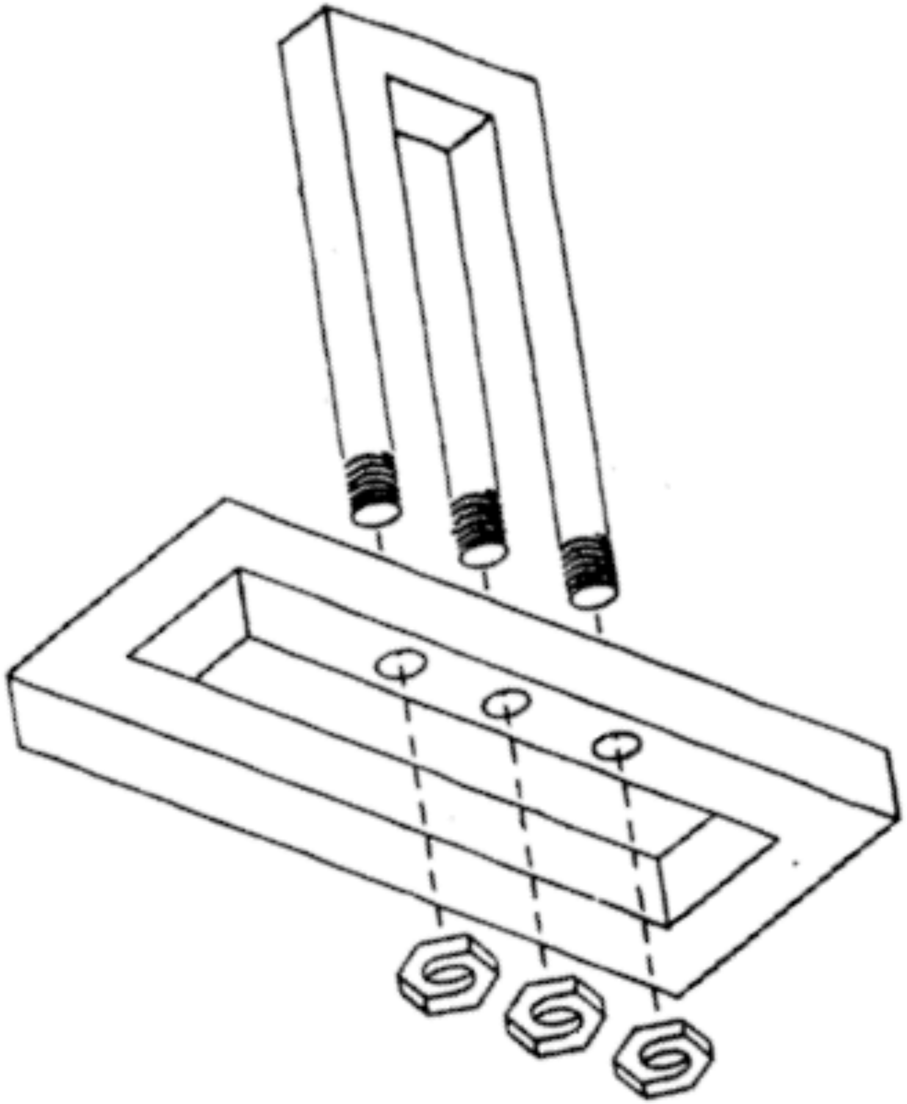
Can you develop this system?



Can you develop this system?

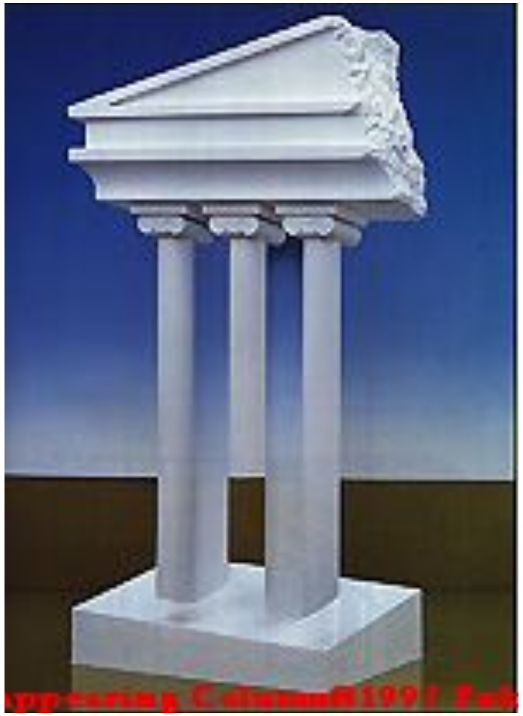


Can you develop this system?



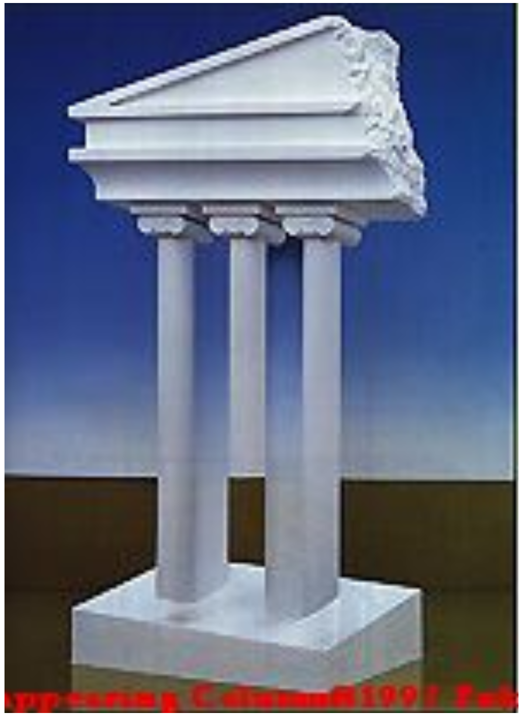
The impossible Fork

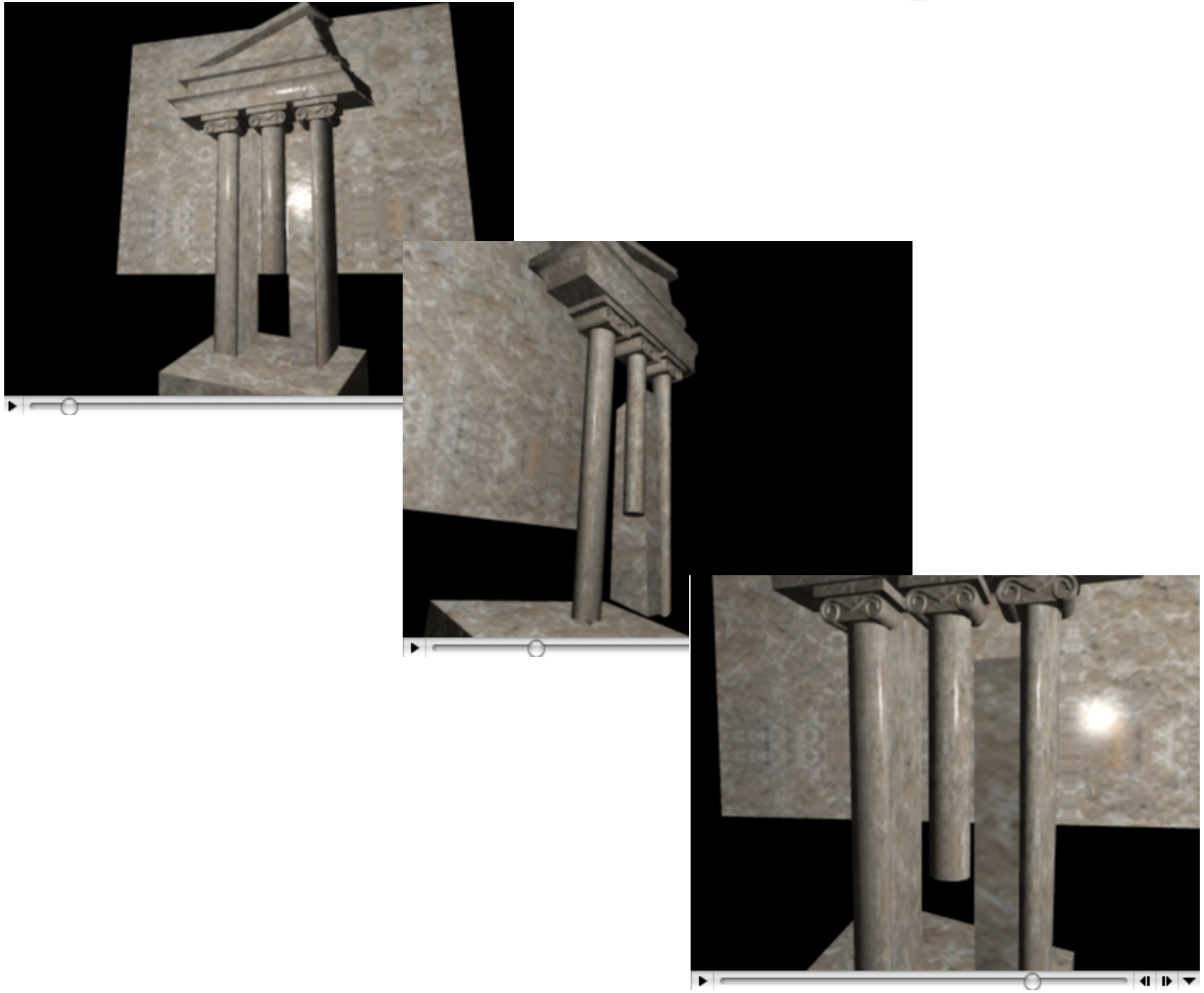
Physical Model of the impossible Fork (Shigeo Fukuda)



From: http://illusionworks.com/mod/movies/fukuda/DisappearingColumn.mov

Physical Model of the impossible Fork (Shigeo Fukuda)





From: http://illusionworks.com/mod/movies/fukuda/DisappearingColumn.mov

Why is software development difficult?

- The problem domain (also called application domain) is difficult
- The solution domain is difficult
- The development process is difficult to manage
- Software offers extreme flexibility
- Software is a discrete system
  - Continuous systems have no hidden surprises
  - Discrete systems can have hidden surprises! (Parnas)

David Lorge Parnas is an early pioneer in software engineering who developed the concepts of modularity and information hiding in systems which are the foundation of object oriented methodologies.

Software Engineering is more than writing Code

- Problem solving
  - Creating a solution
  - Engineering a system based on the solution
- Modeling
- Knowledge acquisition
- Rationale management

Techniques, Methodologies and Tools

- **Techniques:**
  - Formal procedures for producing results using some well-defined notation
- **Methodologies:**
  - Collection of techniques applied across software development and unified by a philosophical approach
- **Tools:**
  - Instruments or automated systems to accomplish a technique
  - CASE = Computer Aided Software Engineering

Computer Science vs. Engineering

- Computer Scientist
  - Assumes techniques and tools have to be developed.
  - Proves theorems about algorithms, designs languages, defines knowledge representation schemes
  - Has infinite time…
- Engineer
  - Develops a solution for a problem formulated by a client
  - Uses computers & languages, techniques and tools
- Software Engineer
  - Works in multiple application domains
  - Has only 3 months...
  - …while changes occurs in the problem formulation (requirements) and also in the available technology.

Software Engineering: A Working Definition

Software Engineering is a collection of techniques, methodologies and tools that help with the production of

A high quality software system developed with a given *budget* before a given *deadline* while *change* occurs

Challenge: Dealing with complexity and change

Software Engineering: A Problem Solving Activity

- **Analysis:**
  - Understand the nature of the problem and break the problem into pieces
- **Synthesis:**
  - Put the pieces together into a large structure

For problem solving we use techniques, methodologies and tools

Outline of Today's Lecture

- ✓ Organizational issues
  - ✓ Scope of the lecture
  - ✓ Grading, Rooms, Times and Registration
- ✓ The development challenge
- ➢ Organizational issues
  - Lecture schedule
- Dealing with complexity and change
- Methodologies

Course Schedule

Block 1 (Oct 14 – Oct 17) Introduction, Methodologies Overview (Ch 16)

Block 2 (Oct 21 – Oct 24) Notation (No lectures, only exercises)

UML Refresher (Ch.2, IN0006) => Tuesday (Oct 21), Friday (Oct 24)

Introduction to UML 2.0 => Thursday, Oct 23

Block 3 (Oct 28 – Nov 14) Requirements Elicitation, Analysis, Design

Scenario-based design, specification methods, OOSE, SA/SD, etc

Block 4 (Nov 18 – Nov 21) Build and Release Management

"Hello Dolly" instead of "Hello World", Continuous Integration, Maven

Block 5 (Nov 25 – Nov 28) Testing

Horizontal and Vertical Testing, Unit_Testing, Programming Contest

Block 6 (Dec 9– Dec 12) Software Lifecycle Models

Defined vs Empirical Process Control, linear, iterative, activities-vs entities

Block 7 (Dec 16– Jan 9) Detailed Design/Implementation Concepts

Inheritance, Reuse, Mapping models to code (Ch 9-10)

Block 8 (January and February): Methodologies in Detail

XP, Scrum, Rugby, etc, Invited lectures from industry

(Subject to Change).

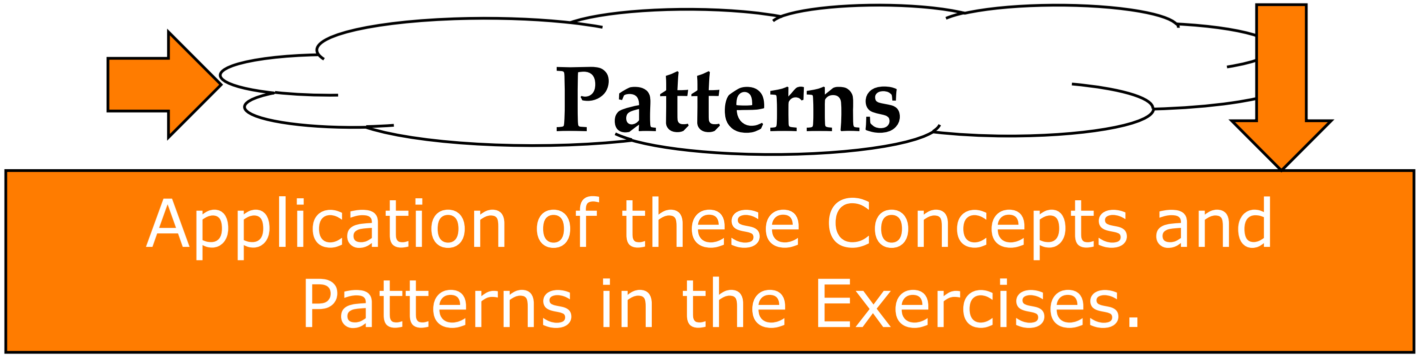
Course Outline

Dealing with Complexity

- Methodologies
  - Repeatable vs Empirical Control Models
- Notations (Structure, Dynamics, Functionality)
- Requirements Engineering, Analysis and Design
  - OOSE, SA/SD, scenario-based design, formal specifications
- Testing
  - Vertical and horizontal testing

Dealing with Change

- Knowledge Management
  - Rationale Management, Wikinomics
- Release Management
  - Big Bang vs Continuous Integration
- Software Life Cycle
  - Linear vs iterative models
  - Activity-vs Entity-based views



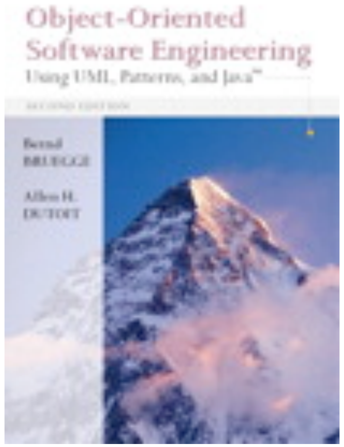
Patterns

Application of these Concepts and Patterns in the Exercises.

Exercises

- The exercises are organized and coached by Florian Schneider and Hans Breidler
  - Florian.schneider@in.tum.de
  - Hans.Breidler@in.tum.de
- One exercise session (90 min) per week: Thursday 8:15- 9:45 am
- Registration and attendance of the exercise sessions is highly recommended.

Reading



Bernd Bruegge, Allen H. Dutoit

Object-Oriented Software Engineering: Using UML, Patterns and Java, 2nd Edition

Publisher: Prentice Hall, Upper Saddle River, NJ, 2003;

ISBN: 0-13-047110-0

- German Version: "Objektorientierte Softwaretechnik mit UML, Entwurfsmustern und Java, Pearson Education, Oktober 2004.
- Spanish Version: Ingeneria De Software Orientado a Objetos
  - Prentice Hall, 1st edition, 2002
- Chinese Version also available

- Additional readings will be at the end of each lecture.

Questions?

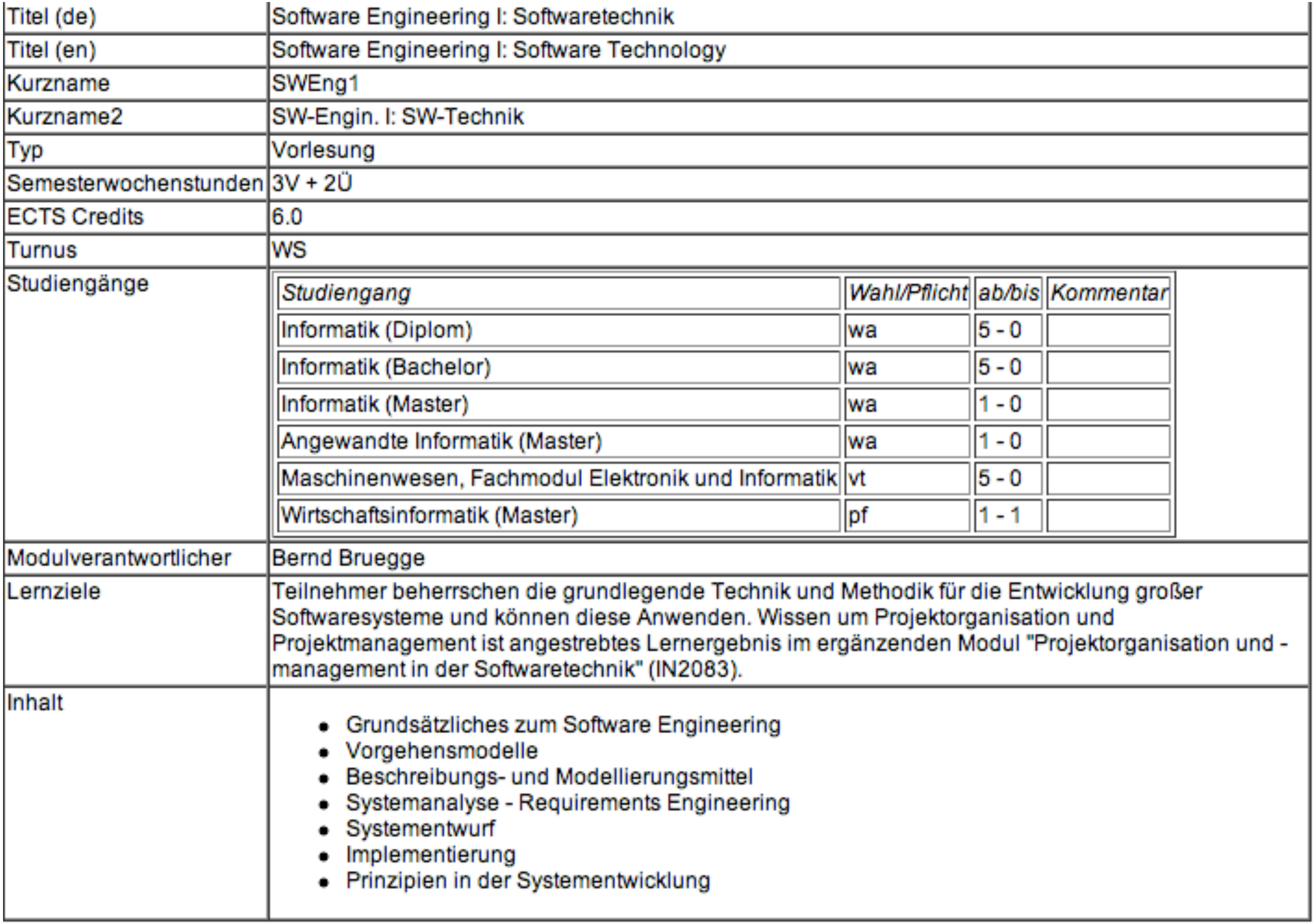
- Lecture Portal:
  - http://wwwbruegge.in.tum.de/twiki/bin/view/Lehrstuhl/TeachingWiSe2008
  - The lecture slides will be posted in PDF format after the lecture is given
- Exercise Portal:
  - Separate home page will be set up for the exercise materials

- What happens if I don't participate in the exercises?

More Questions?

©

Backup Slides



<table>
<tr><td>Titel (de)</td><td>Software Engineering I: Softwaretechnik</td></tr>
<tr><td>Titel (en)</td><td>Software Engineering I: Software Technology</td></tr>
<tr><td>Kurzname</td><td>SWEng1</td></tr>
<tr><td>Kurzname2</td><td>SW-Engin. I: SW-Technik</td></tr>
<tr><td>Typ</td><td>Vorlesung</td></tr>
<tr><td>Semesterwochenstunden</td><td>3V + 2Ü</td></tr>
<tr><td>ECTS Credits</td><td>6.0</td></tr>
<tr><td>Turnus</td><td>WS</td></tr>
<tr><td>Studiengänge</td><td>
<table>
<tr><th><i>Studiengang</i></th><th><i>Wahl/Pflicht</i></th><th><i>ab/bis</i></th><th><i>Kommentar</i></th></tr>
<tr><td>Informatik (Diplom)</td><td>wa</td><td>5 - 0</td><td></td></tr>
<tr><td>Informatik (Bachelor)</td><td>wa</td><td>5 - 0</td><td></td></tr>
<tr><td>Informatik (Master)</td><td>wa</td><td>1 - 0</td><td></td></tr>
<tr><td>Angewandte Informatik (Master)</td><td>wa</td><td>1 - 0</td><td></td></tr>
<tr><td>Maschinenwesen, Fachmodul Elektronik und Informatik</td><td>vt</td><td>5 - 0</td><td></td></tr>
<tr><td>Wirtschaftsinformatik (Master)</td><td>pf</td><td>1 - 1</td><td></td></tr>
</table>
</td></tr>
<tr><td>Modulverantwortlicher</td><td>Bernd Bruegge</td></tr>
<tr><td>Lernziele</td><td>Teilnehmer beherrschen die grundlegende Technik und Methodik für die Entwicklung großer Softwaresysteme und können diese Anwenden. Wissen um Projektorganisation und Projektmanagement ist angestrebtes Lernergebnis im ergänzenden Modul "Projektorganisation und -management in der Softwaretechnik" (IN2083).</td></tr>
<tr><td>Inhalt</td><td>
<ul>
<li>Grundsätzliches zum Software Engineering</li>
<li>Vorgehensmodelle</li>
<li>Beschreibungs- und Modellierungsmittel</li>
<li>Systemanalyse - Requirements Engineering</li>
<li>Systementwurf</li>
<li>Implementierung</li>
<li>Prinzipien in der Systementwicklung</li>
</ul>
</td></tr>
</table>