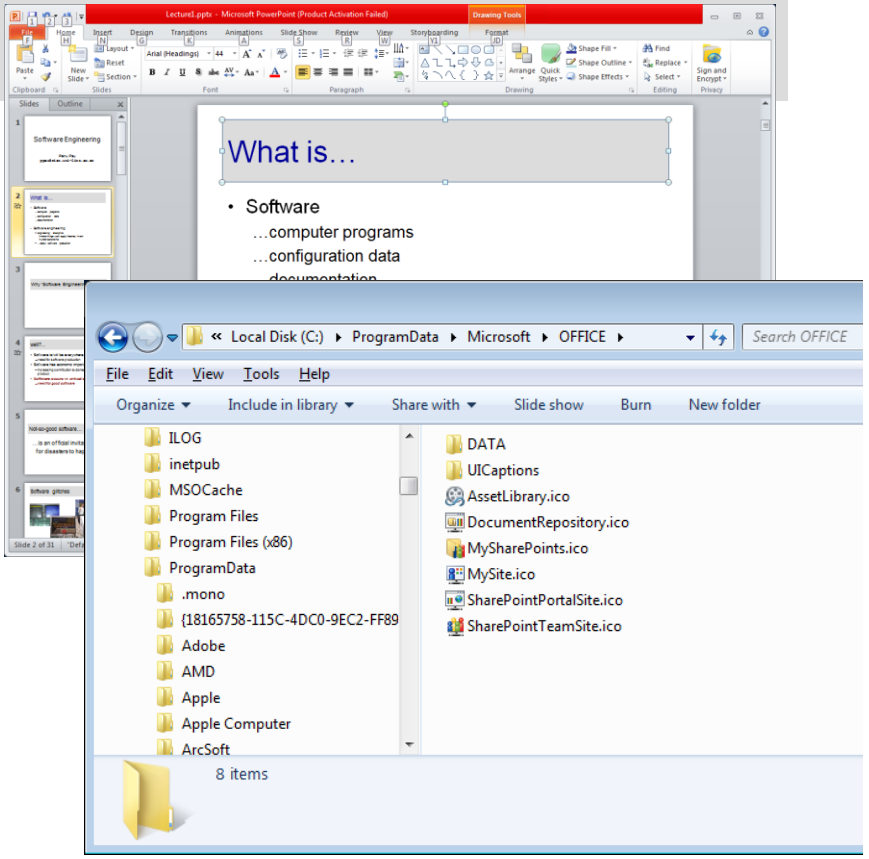
Software Engineering

Petru Pau
ppau@risc.uni-linz.ac.at

What is…

- Software
  - …computer programs
  - …configuration data / program data
  - …documentation



- Software engineering
  - engineering discipline
    - ➢ make things work:
      - apply theories, invent
    - ➢ *under constraints*
  - …about software production

Why “Software Engineering”?

well...

- Software is/will be everywhere (pervasive)
  →need for software production
- Software has economic importance
  – increasing contribution to domestic gross product
- **Software occurs in critical systems**
  →need for *good software*

Not-so-good software…

…is an official invitation for disasters to happen

Software glitches

Software glitches

- NASA Mars Polar Lander,
  - December 3, 1999
  - **destroyed**
  - its flight software falsely interpreted vibrations due to atmospheric turbulence
  - it believed that the vehicle had landed

  → shut off the engines 40 meters from the Martian surface ()

Software glitches

- radiological emergency in Panama (May 2001)
  - 28 patients overexposed (8 deaths)
  - radiotherapy equipment OK; cause in data entry;
  - for specific input data, radiation dose was incorrectly computed.

`http://www.fda.gov/cdrh/ocd/panamaradexp.html`

What does “good software” mean?

Software quality

- Measure:
  - bugs per thousand lines of code
- Problem:
  - can be estimated after delivery
- Industry average: 10

- However, **No Bugs ≠ High Quality**

What defines high-quality software?

- *…its attributes*:
  - Usability
  - Efficiency
  - Maintainability
  - Dependability

Back to Software Engineering…

Relations

- SE versus *Computer Science*:
  - Comp.Sci.: theories & methods
  - SE: practical methods for producing software

Relations

- SE versus *System Engineering*
  - System Engineering:
    - development & evolution of systems [where software may play a role]
    - includes hardware development, among others
      - specifying the system
      - defining architecture
      - integrating parts
  - SE – only a part of system engineering
  - System engineering – much older than SE

Relations

- SE versus *coding* (programming)
  - Coding: activity in software production
  - Coding: belongs to the problem, not to the solution

You should be aware…

- SE will help you step beyond restrictions imposed by code / programming language
- Thinking in advance always helps
- Delegation and teamwork are essential
- *Keep it simple* principle (it's not easy!):

> "…there are two ways of constructing a software design: One way is to make it so simple there are obviously no deficiencies and the other way is to make it so complicated that there are no obvious deficiencies."
>
> Tony Hoare, *Turing Award Lecture*, 1980

SE challenges

- heterogeneity
  - software systems are required to run on different types of machines, op sys, etc.
- delivery
  - shorten the delivery time without compromising quality
- trust
  - users must be confident that the software works without doing any damage

Some more definitions…

What is…

- Software [engineering] process:
  - all activities whose goal is the development & evolution of software

  - Specification
  - Development
  - Validation
  - Evolution

Documentation

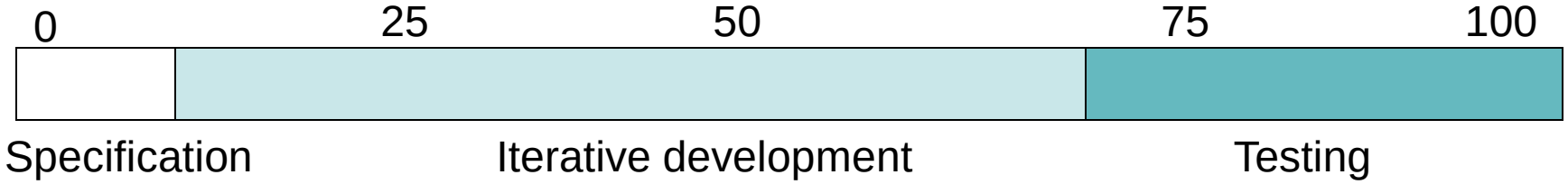
What is…

- Software process **model**
  - simplified description of a software process
  - paradigms:
    - *waterfall*:
      - activities are considered one at a time
    - *iterative development*:
      - interleaves specification, development, validation
    - *component-based SE*:
      - integration of pre-existing components

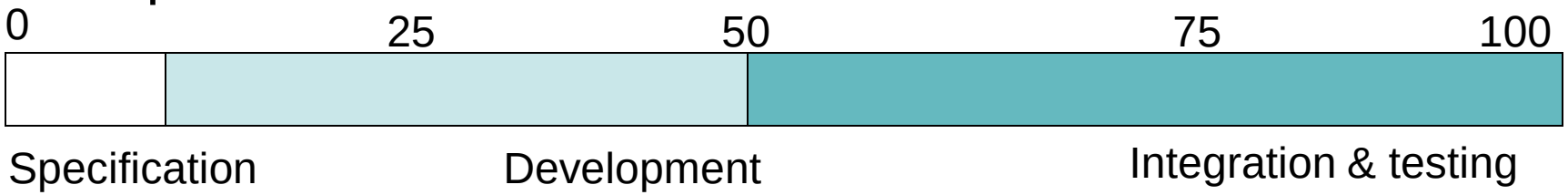
Costs

Waterfall

Iterative development



Component-based SE



Methods

- *SE Method*:
  - structured approach to software development
  - goal: enable, support, enforce production of high quality software
- Types:
  - function-oriented
  - object-oriented
  - ...

CASE

Computer Aided Software Engineering

- Tools that support al phases of a software process
  - Editors
  - Report generators
  - Code generators

You…

Your responsibilities…

…as (potential, future) software engineers:

- Confidentiality
- Competence
- Respect intellectual property rights
- Avoid computer misuse

Your work…

- proof of current knowledge
  - you get one question after each lecture, in the published slides
    → 12 questions
  - you must know the answer at the next lecture (you may be asked to prove this!)
  - you can fail at most 5 questions
  - 20% of the final grade
- either:
  - a project, or
  - a final theoretical exam
  - 80% of the final grade

Your work for a project…

- Prerequisites:
  1. a programming language (Java, C++, C#)
  2. experience in building user interfaces (UI)
  3. basics of algorithms and data structures
  4. e-mail addresses

  & access to computers with compilers for your preferred language

  & with packages for development of graphical UI

Your work for a project…

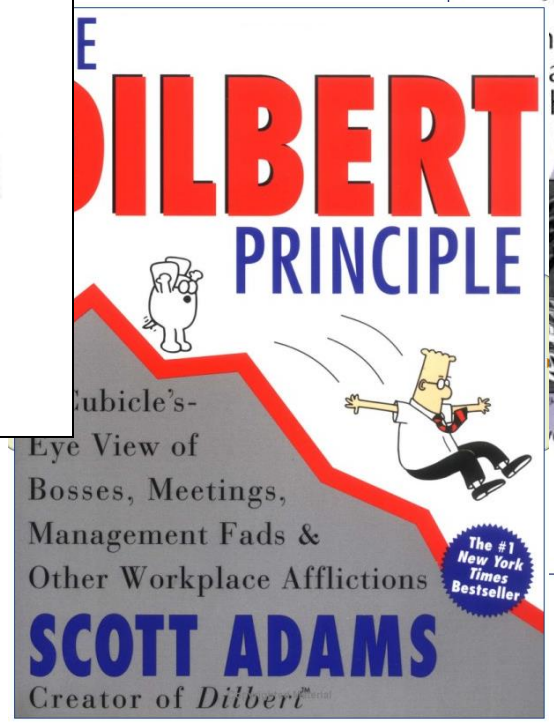
- proposed themes will be posted
- deliveries: 4 artefacts,
  - will be announced later, with their due dates

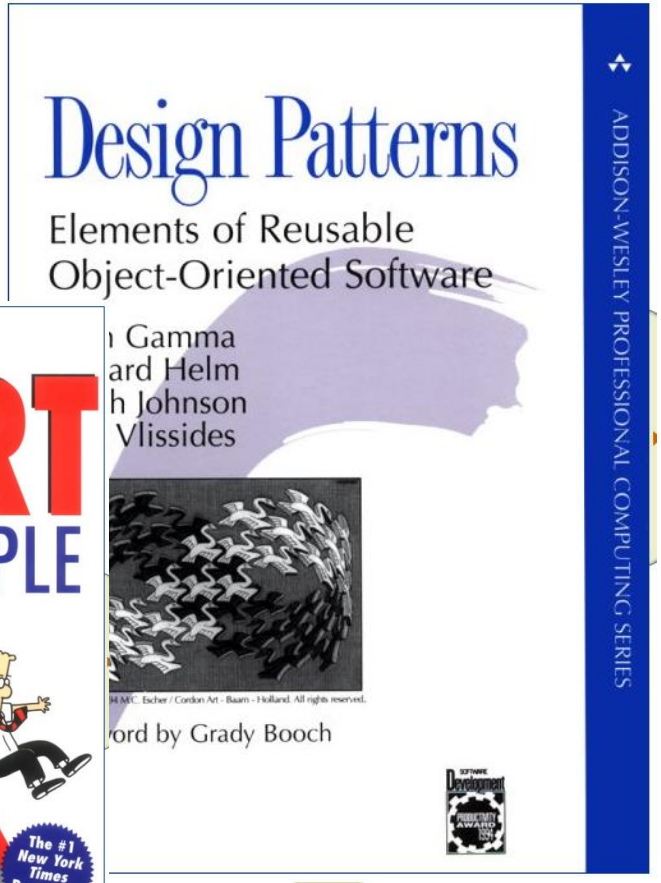
    → four grades
- required working time: ~2.5 hours/week

Development tools

- Microsoft Visual Studio 2010 or later, Express or Developer editions
- Eclipse - http://www.eclipse.org/downloads/
  - needs Java Runtime
    - see instruction in the download web page

Recommended texts





Homework

- Think of (at least three) other attributes that may define software quality.
  - You should be able to describe them