Motivation and Contents Overview

Software Engineering
Winter Semester 2011/2012

Department of Computer Science
cs.uni-salzburg.at

Dr. Stefan Resmerita

UNIVERSITÄT
SALZBURG

Course Contents

UNIVERSITÄT SALZBURG

Goals

- Learning about commonly used approaches to software development (in the small and in the large)
- Developing an understanding of what is good and what is bad software (-construction)
- Knowing and understanding related concepts and terms
- Developing a first understanding of the „Software development in the large“

UNIVERSITÄT SALZBURG

Software Engineering

- **Concepts and constructs for flexible software**
  - Programming language (OO)
  - UML representation
  - Frameworks and Design Patterns
  - Software parameterization (configuration files, resources, script languages)
  - Heuristics for adequate flexibility

Software Engineering

- **Concepts and constructs in Component-Based Design**
  - The Module concept
  - Overview of standards for components (WebServices, JavaBeans, OSGi)
  - Heuristics for adequate modularization (Balance between Coupling and Cohesion in a Discrete Event Simulation example)
- **Software architectures**
- **Automatic software generation**

UNIVERSITÄT SALZBURG

Software Technology: State of the Art and Challenges

Software Engineering
Winter Semester 2011/2012

Department of Computer Science
cs.uni-salzburg.at

Dr. Stefan Resmerita

UNIVERSITÄT SALZBURG

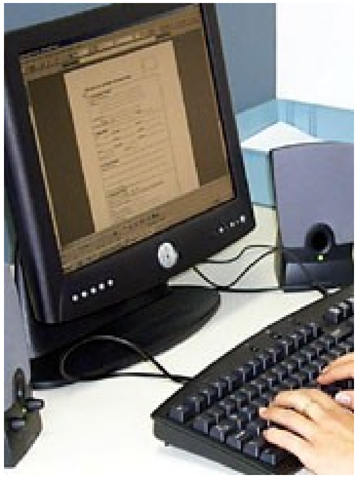
Context

- The phenomenon Software
- How can Software be engineered?

The Phenomenon Software

UNIVERSITÄT SALZBURG

The Computer as universal machine makes Software pervasive



Airplane/Rocket control

ca. 70 Processors
in a car

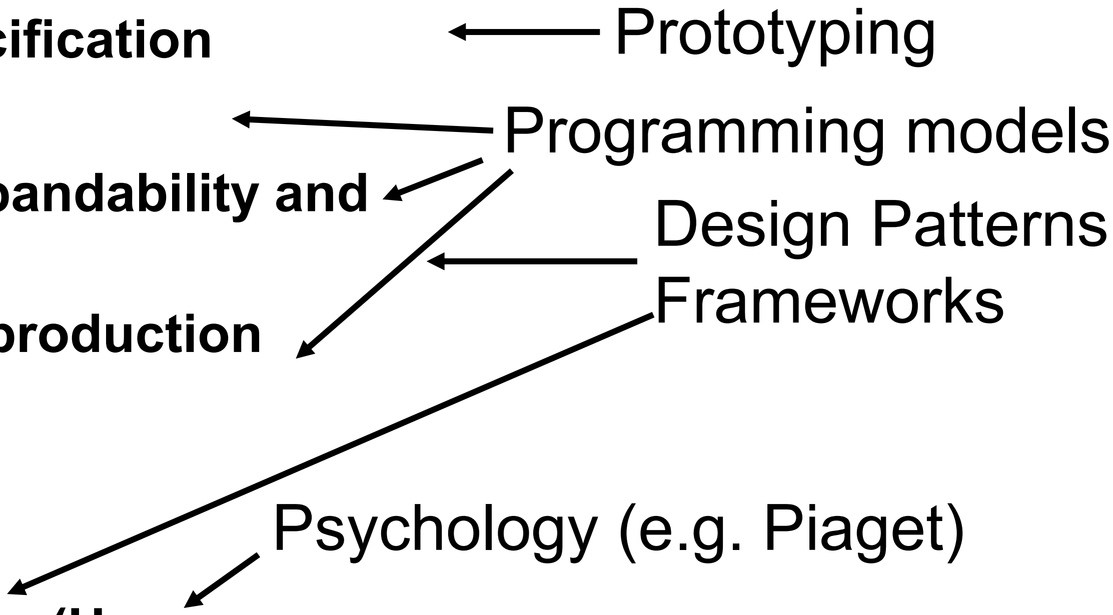
UNIVERSITÄT SALZBURG

What is so special about Software?

UNIVERSITÄT SALZBURG

The problems with software production is the complexity of the achieved product

- **Requirements specification**
- **Complexity control**
- **Re-use/Plug-in, expandability and changeability**
- **Automation in the production process**
- **Portability**
- Documentation
- **Product ergonomics (Human-Computer Interface)**
- Project organization and control
- Quality assurance and evaluation
- Cost estimation



Prototyping

Programming models

Design Patterns

Frameworks

Psychology (e.g. Piaget)

Quality problems

- Software bugs: deficiencies with drastic effects
  - Incorrect bank transactions
  - Y2K
  - Ariane
  - Mars adventures
    - PathFinder
    - Spirit

UNIVERSITÄT SALZBURG

Example: Ariane 5

- Construction:
  - 10 years & $7billion
- Maiden voyage: June 1996
- Payload: 4 scientific satellites

Example: Ariane 5

- Crashed at second 39 in flight
- Software bug: number overflow
  - Wrong sensor data
  - Wrong steering
  - Activate self-destruct
- Software component inherited from previous versions (Ariane 4)

Example: Ariane 5

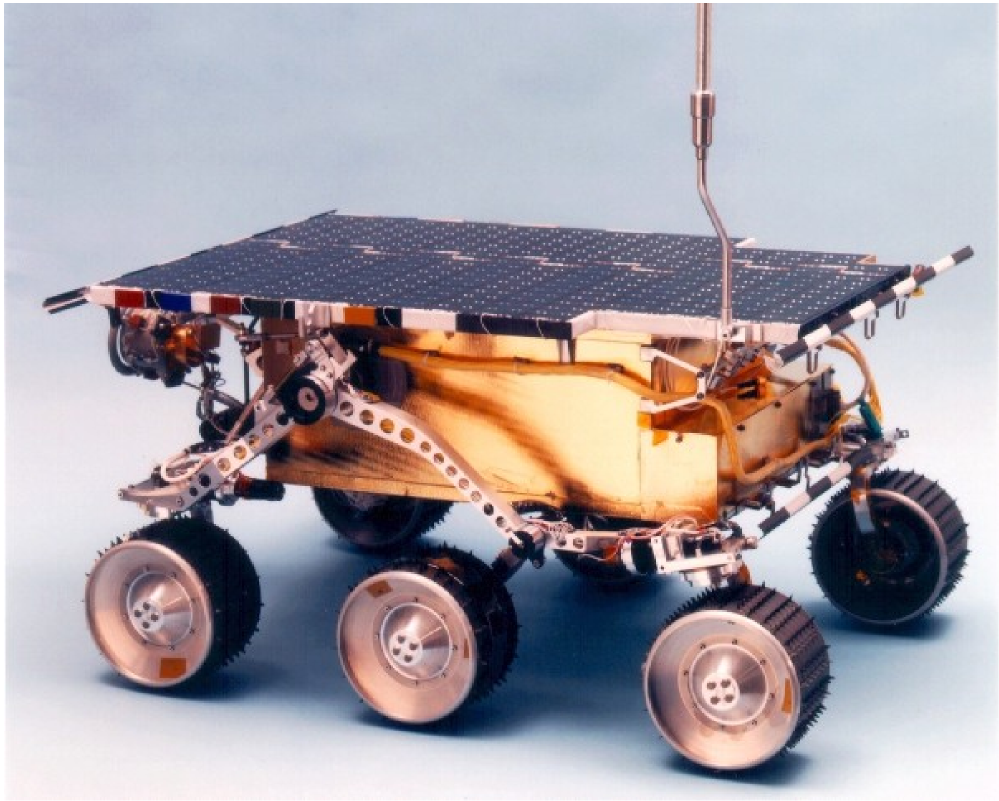
- Crashed at second 39 in flight
- Software bug: number overflow
  - Wrong sensor data
  - Wrong steering
  - Activate self-destruct
- Software component inherited from previous versions (Ariane 4)
- Inquiry board conclusion:

"The Board is in favour of the opposite view, that software should be assumed to be faulty until applying the currently accepted **best practice methods** can demonstrate that it is correct."

UNIVERSITÄT SALZBURG

Example: PathFinder Rover on Mars

- Landed on July 4, 1997
- Problem: frequent total system resets



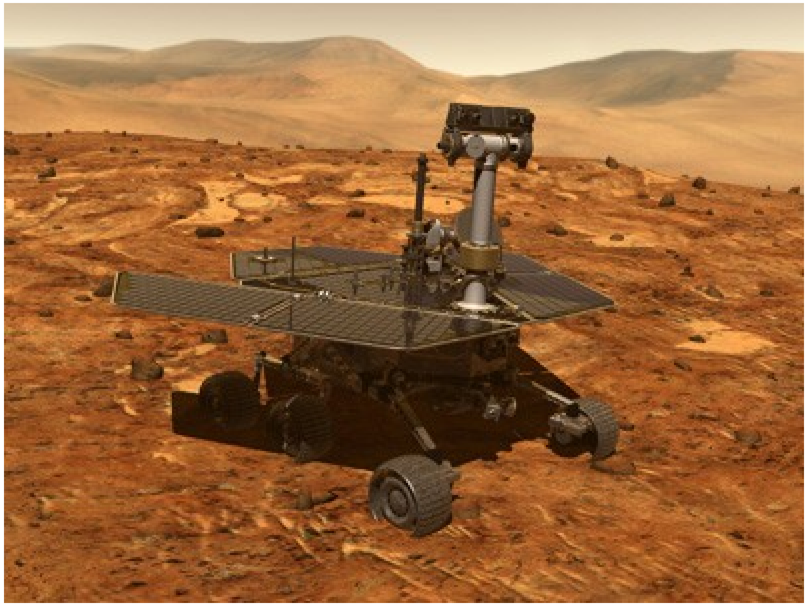
Example: PathFinder Rover on Mars

- Landed on July 4, 1997
- Problem: frequent total system resets
- Cause: data bus locked longer than expected
- Software tasks:
  - Bus management
  - Communication
  - Meteorological
- Solution:
  - Priority inversion

UNIVERSITÄT SALZBURG

Example: Spirit Rover on Mars

- Landed on January 4, 2004
- Problem: frequent total system resets



Example: Spirit Rover on Mars

- Landed on January 4, 2004
- Problem: frequent total system resets
- Cause: size of file system
  - DOS FS on flash
  - Mirrored in RAM
  - Sizeof(RAM) < sizeof (Flash)

Human interaction problems

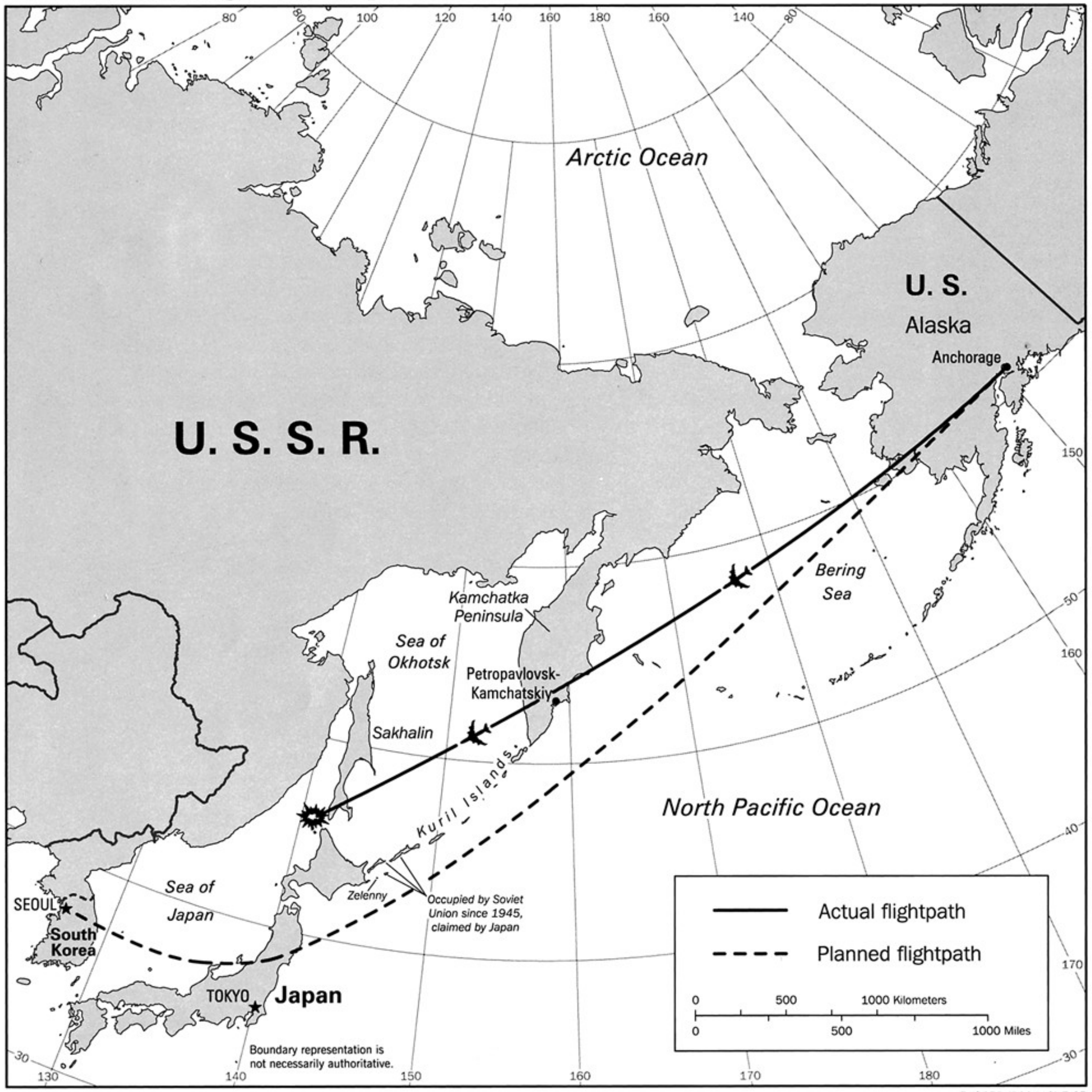
- Human-Computer Interaction

- Human-Machine Interaction
  - Interaction with automated systems
  - Example: Korean Air Lines Flight 007

- Computer pervasiveness makes the human interaction issue very important

KAL007 flight route

Korean Airlines Flight 007, 1 September 1983

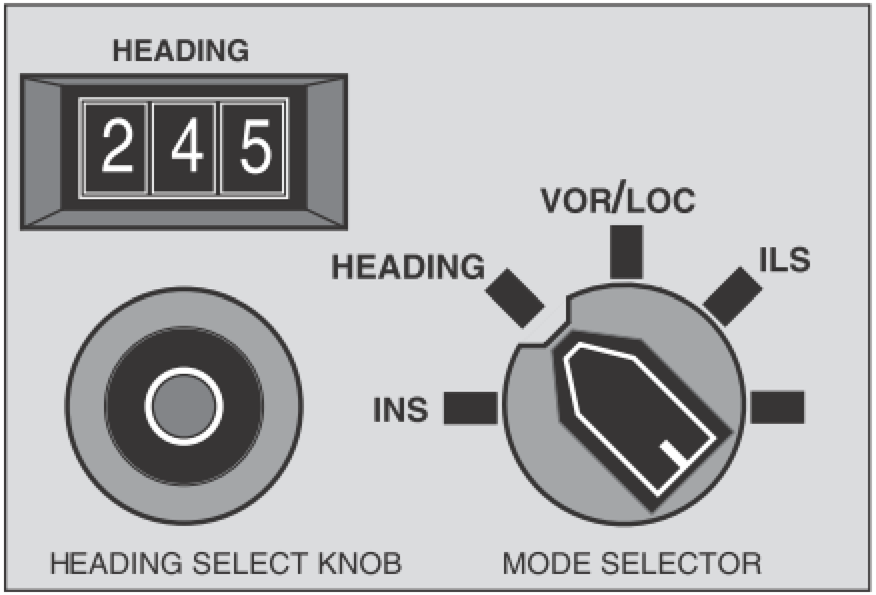


Boundary representation is not necessarily authoritative.

743555 (R02443) 7-97

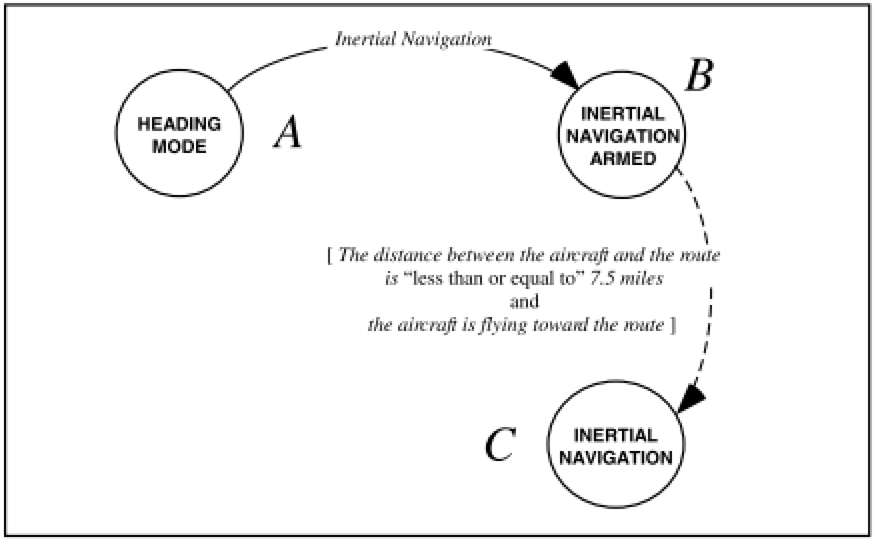
KAL007 Navigation Interface

- Navigation routine:
  - Start in Heading
  - Switch to INS



KAL007 navigation modes

- Operating modes:


Inertial Navigation
HEADING MODE
A
B
INERTIAL NAVIGATION ARMED
[The distance between the aircraft and the route is "less than or equal to" 7.5 miles and the aircraft is flying toward the route]
C
INERTIAL NAVIGATION

- Problem:
  - Transition from B to C not clear to the pilots!

Example: Specification problems

UNIVERSITÄT SALZBURG

An exact specification is often impracticable

given.: $n \geq 3$,

$$L: N_n \rightarrow N$$

find.: A Program P that computes

$$a: N_3 \xrightarrow{inj} N_n \text{ , such that}$$

$$\bigwedge_{1 \leq i \leq 3} \; \bigwedge_{\substack{j \in N_n \setminus \cup \{ a_k \} \\ 1 \leq k \leq j}} L(a_i) \geq L(a_J)$$

...while a verbal specification is often inaccurate

Given a list with at least three positive numbers

Find a program P that gives the indices of the three largest elements of the list.

Mastering Complexity

UNIVERSITÄT SALZBURG

In classical engineering disciplines

- Bad quality can hardly be hidden
  - Door cannot close well
  - Unnecessary artifacts
    - „Fifth wheel to the car“
- Resources are limited
  - Engineering approaches mean optimization under given basic conditions

Bad quality is not so visible in software

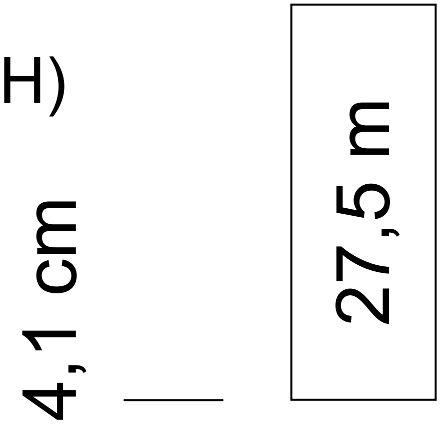
- Bad structuring
  - „Spaghetti“ program code:
    - Wheel change -> the motor works no more
  - Replicated program code

- Hardly re-usable code
  - The wheel is always re-invented

Engineering procedures do not seem to pay off

- Hardware resources evolve according to Moore‘s Law; thoughtless handling of this issue leads to:
  - Unnecessary complexity
  - No longer understandable artifacts

OberonOS (ETH ZH)
30.000 lines of
program code



4,1 cm

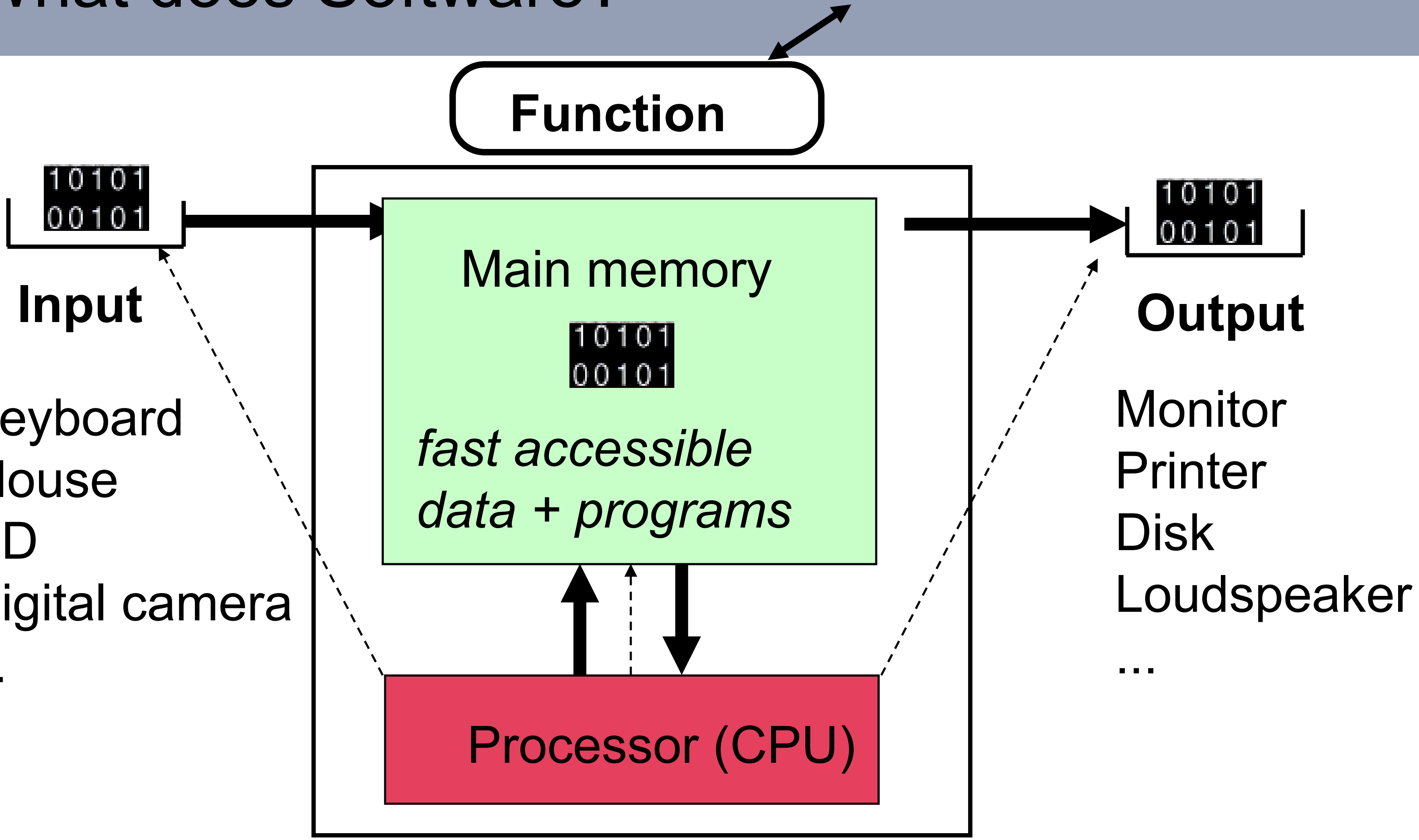
27,5 m

Windows XP (2002):
40.000.000 (!!) lines
of program code

How can Software be engineered?

UNIVERSITÄT SALZBURG

What does Software?



➡ Data

- - - ▸ Control signals

UNIVERSITÄT SALZBURG

Interaction with the environment

- Interactive systems: the computer is the leader of the interaction
  - Examples: Operating systems, Database systems
  - Main issues: Deadlock, Fairness

- Reactive systems: the environment is the leader of the interaction
  - Examples: Industrial process control, airplane control
  - Main issues: Safety, Timeliness

UNIVERSITÄT SALZBURG

Examples

- ABS in automotive
  - **Input:** Rotational speeds of the wheels and user braking
  - **Function:** Checking whether the speeds are zero when the user brakes
  - **Output:** Appropriate controlling of the braking force
- Bank transfers
  - **Input:** Transfer data (payee, payer, amount)
  - **Function:** Validation of the transaction
  - **Output:** New transaction lines in the accounts