

Dr. Michael Eichberg

Software Engineering

Department of Computer Science

Technische Universität Darmstadt

Software Engineering

What is Software Engineering?



TECHNISCHE
UNIVERSITÄT
DARMSTADT

What is Software?



TECHNISCHE
UNIVERSITÄT
DARMSTADT

⚡ ⚡ *Software - The programs and other operating information used by a computer.*

New Oxford American Dictionary; 2005

↳ ↳ [...] software is not just the programs but also all associated documentation and configuration data that is needed to make these programs operate correctly.

I. Sommerville

Software Engineering Eighth Edition; Pearson Education, 2007

↳ ↳ *The term software refers to a program and all of the associated information and materials needed to support its...*

installation,

operation,

repair and

enhancement.

W. S. Humphrey

The Software Engineering Process: Definition and Scope; ACM SIGSOFT Software Engineering Notes, Vol. 14, Issue 4, 1989

Software is more than just code.

- An executable program and its data
- Configuration files
- System documentation
(e.g. architectural and analysis model, a design document,...)
- User documentation
- A website
(To inform about issues, download updates,...)
- ...

Several types of software can be distinguished.

- **Generic products**
(in the past referred to as shrink-wrapped software)
e.g. Microsoft Word, Open Office, Acrobat, ...
to shrink-wrap =dt. einschweißen; in Schrumpffolie verpacken
- **Customized products**
(individual software, build-to-order software)
e.g. TUCaN (Campusnet), an Air Traffic Control System, ...

The borders are blurring (e.g. Enterprise Resource Planning (ERP) software is often customized to match the workflows in a particular company).

Properties of Software



TECHNISCHE
UNIVERSITÄT
DARMSTADT

Software has unique properties when compared to any hardware.

- **No “real” physical borders**

- ▶ **Software doesn't wear out / there are no spare-parts**

Nevertheless, Software has to be constantly updated to cope with changing environments; otherwise the software will become obsolete (software aging).

- ▶ **Software is hard to “measure”**

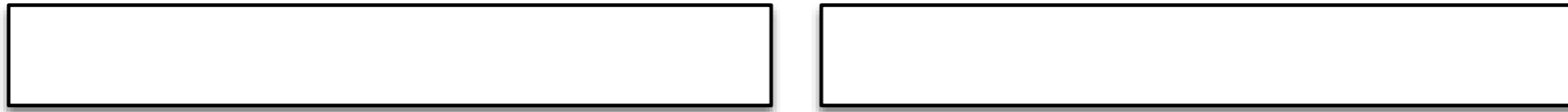
How to define the quality of software?

Are those things (e.g. the lines of code) that can be measured correlated to the quality? How can we measure progress?

“To Code is to Design”...



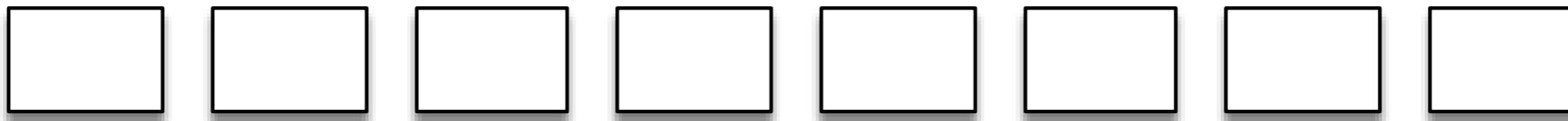
Application software lifetime



System software lifetime



Hardware lifetime



time

Balzert

*Lehrbuch der Softwaretechnik; Spektrum Akademischer Verlag,
1996*

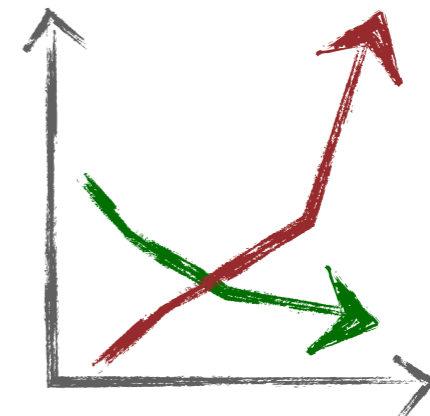
What is Software Engineering?



TECHNISCHE
UNIVERSITÄT
DARMSTADT

The term “Software Crisis” was coined in the 60’s and refers to multiple problems.

- The costs for hardware were falling, but the costs for software were rising significantly
- Software projects were not in-time, were not in-budget and contained too many errors
- Technological issues
 - Lack of suitable programming languages
 - Lack of methods
 - Lack of tool support
 - ...



The term “Software Engineering” was coined at the end of the sixties and is often attributed to F.L. Bauer.

What is Software Engineering? | 13



The NATO Software Engineering Conference
(Garmisch, Germany, 7-11 Oct 1968)

<http://homepages.cs.ncl.ac.uk/brian.randell/NATO/N1968/index.html>

(Software Engineering ~ dt. Softwaretechnik / Softwaretechnologie)

Software Engineering refers to the disciplined application of engineering, scientific, and mathematical principles and methods to the economical production of quality software.

[...] quality refers to the degree to which a product meets its users' needs.

W. S. Humphrey

The Software Engineering Process: Definition and Scope; ACM SIGSOFT Software Engineering Notes, Vol. 14, Issue 4, 1989

“Software Engineering”

(1) The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software.

(2) The study of approaches as in (1).

IEEE Standards Board

IEEE Standard Glossary of Software Engineering Terminology

Std. 610.12-1990, 1990

“Software Engineering”

Software engineering is a systematic and disciplined approach to developing software. It applies both computer science and engineering principles and practices to the creation, operation, and maintenance of software systems.

[Computer Science is concerned with the theories and methods that underlie computers and software systems, software engineering is concerned with the practical problems of producing software.]

University of Waterloo

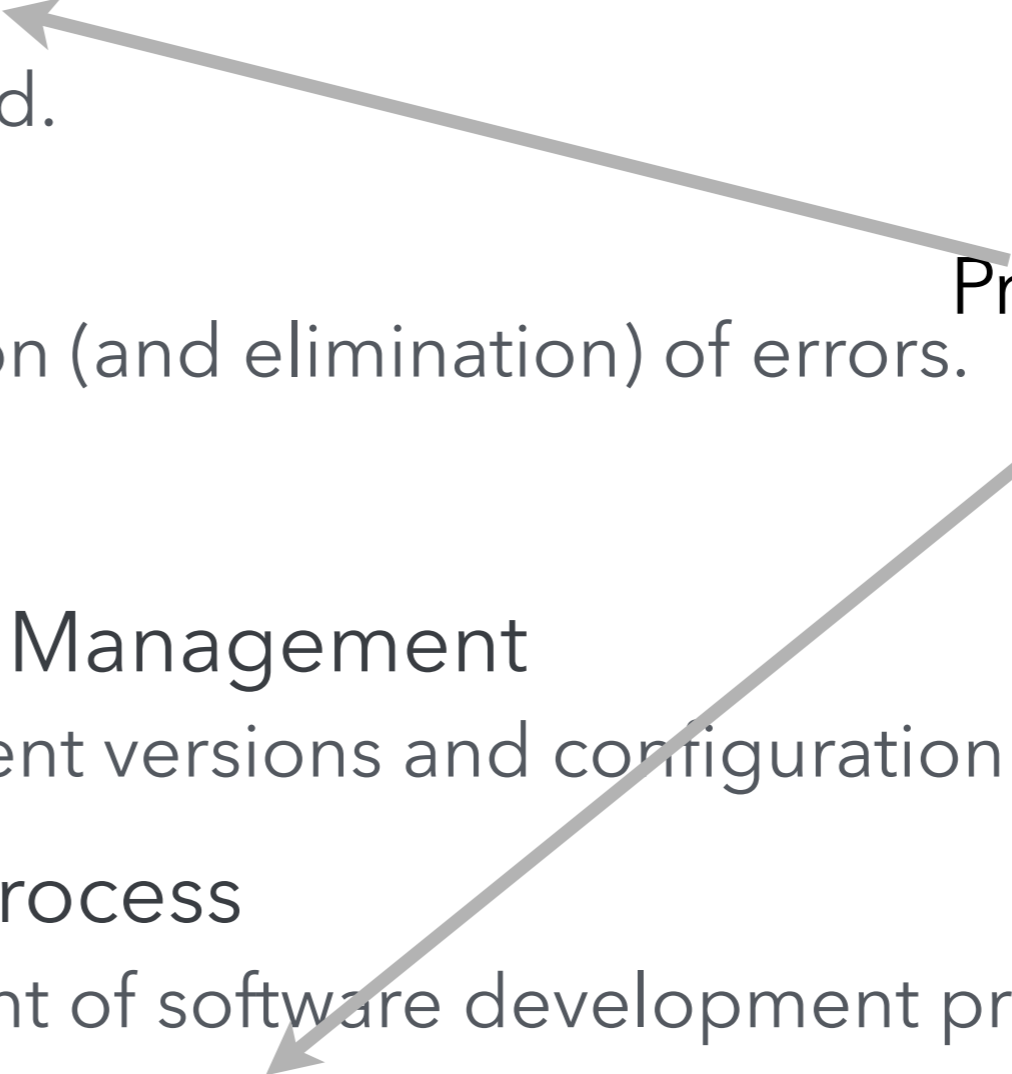
<http://www.softeng.uwaterloo.ca/> (since 2007)

↳ ↳ [...] Der Begriff *Software-Engineering* steht für die Auffassung, dass die Erstellung, Anpassung und Wartung von Programmsystemen kein "künstlerischer", sondern vorwiegend ein ingenieurmäßig verlaufender Prozess ist...

Software engineering encompasses several areas.

- **Software Requirements**
The requirements define what the systems is expected to do.
- **Software Design**
How the system is designed.
- **Software Testing**
The systematic identification (and elimination) of errors.
- **Software Maintenance**
- **Software Configuration Management**
The management of different versions and configuration of a software.
- **Software Engineering Process**
Definition and improvement of software development processes.
- **Software Engineering Tools and Methods**
- **Software Quality**

Software engineering encompasses several areas.

- Software Requirements
The requirements define what the systems is expected to do.
 - **Software Design**
How the system is designed.
 - Software Testing
The systematic identification (and elimination) of errors.
 - Software Maintenance
 - Software Configuration Management
The management of different versions and configuration of a software.
 - Software Engineering Process
Definition and improvement of software development processes.
 - **Software Engineering Tools and Methods**
 - Software Quality
- Primary focus of this lecture.
- 

Systems Engineering ↔ Software Engineering

- System related activities, such as defining the overall system objectives and requirements, allocating system functions between hardware and software, defining hardware / software interfaces, full system acceptance tests are essential, but they are part of systems engineering
- Software engineering is a part of systems engineering

We will not talk about systems engineering in this lecture.

What is the current state in Software Engineering?

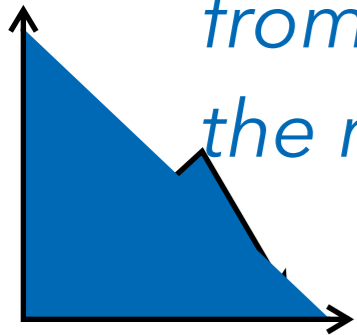


TECHNISCHE
UNIVERSITÄT
DARMSTADT

📌 📌 *This year's [2009] results show a marked decrease in project success rates, with 32% of all projects succeeding which are delivered on time, on budget, with required features and functions.*

44% were challenged which are late, over budget, and/or with less than the required features and functions and 24% failed which are cancelled prior to completion or delivered and never used.

These numbers represent a downtick in the success rates from the previous study, as well as a significant increase in the number of failures[...]



Standish Group, Boston, Massachusetts, April 23, 2009
CHAOS Summary 2009

Software projects fail due to several different reasons.

(A software project is considered to have failed as soon as the project is not on-time or is not in-budget).

What is Software Engineering? Is the Software Crisis still with us? | 23

- The requirements and system dependencies are not well-defined
- Changing the requirements during the development is much, much easier for software than for hardware;
(Software has to accommodate for hardware “issues”.)
- Lack of tools, methods, education, planning, ...

However, other complex and innovative hardware systems are also often behind schedule (e.g. the Airbus A380, the Boeing Dreamliner, the white iPhone).

Engineering Software is about getting the design right and less about building the 42nd A380.

👉 *In the just-released report, CHAOS Manifesto 2011, The Standish Group's shows a marked increase in project success rates from 2008 to 2010. These numbers represent an uptick in the success rates from the previous study, as well as a decrease in the number of failures. [...]*

This year's results represent the highest success rate in the history of the CHAOS Research.

[...] "We clearly are entering a new understanding of why projects succeed or fail." This understanding is spelled out in the CHAOS Manifesto research report.




Standish Group, Boston, Massachusetts, March 3, 2011
CHAOS Manifesto 2011

Fifteen Principles of Software Engineering



TECHNISCHE
UNIVERSITÄT
DARMSTADT

- 
1. *Make quality number 1*
 2. *High-quality software is possible*
 3. *Give products to customers early*
 4. *Determine the problem before writing the requirements*
(...before starting to code)
 5. *Evaluate design alternatives*
 6. *Use an appropriate process model*
 7. *Use different languages for different phases*
 8. ...



1994

Alan M. Davis

Fifteen Principles of Software Engineering;
IEEE Software 1994




7. ...
8. *Minimize intellectual distance*
9. *Put technology before tools*
(Before you use a tool, you should understand and be able to follow appropriate software technique.)
10. *Get it right before you make it faster*
11. *Inspect code*
(... Sometimes code inspections are claimed to be more effective than testing ...)
12. ...

The distance between the real-world problem and the computerized solution to the problem..

biopjew..

Alan M. Davis



Fifteen Principles of Software Engineering;
IEEE Software 1994

- 
11. ...
 12. *Good management is more important than good technology*
(... Management style must be adapted to the situation...)
 13. *People are the key to success*
 14. *Follow with care*
(Just because everybody is doing it, does not make it right for you...)
 15. *Take responsibility*

Alan M. Davis

Fifteen Principles of Software Engineering;
IEEE Software 1994

The goal of this lecture is to enable you to systematically carry out small(er) commercial or open-source projects.

-  Engineering software is hard; this lecture teaches you why and (to some extent) how to tackle common problems.
-  Software engineering is about designing software and not about building software.