

Software Engineering

Prof. Dr. Martin Deubler

Fakultät für Informatik

SE – SoSe 2020

Programmieren alleine reicht auf Dauer nicht ...



Motivation (1)

- Zentrale Zielsetzung
 - Aufzeigen des „Konstanten“
(gilt unabhängig von aktuellen Trends und „Hypes“)
 - Vermittlung von Wissen mit langer Haltbarkeit
- Verständnisaufbau für
 - Zentrale Probleme im Software Engineering
 - Grundlegende Konzepte
(Prinzipien, Methoden, Werkzeuge)
 - Systematische Vorgehensweise
(Prozessmodelle, Projekt- und Qualitätsmanagement)

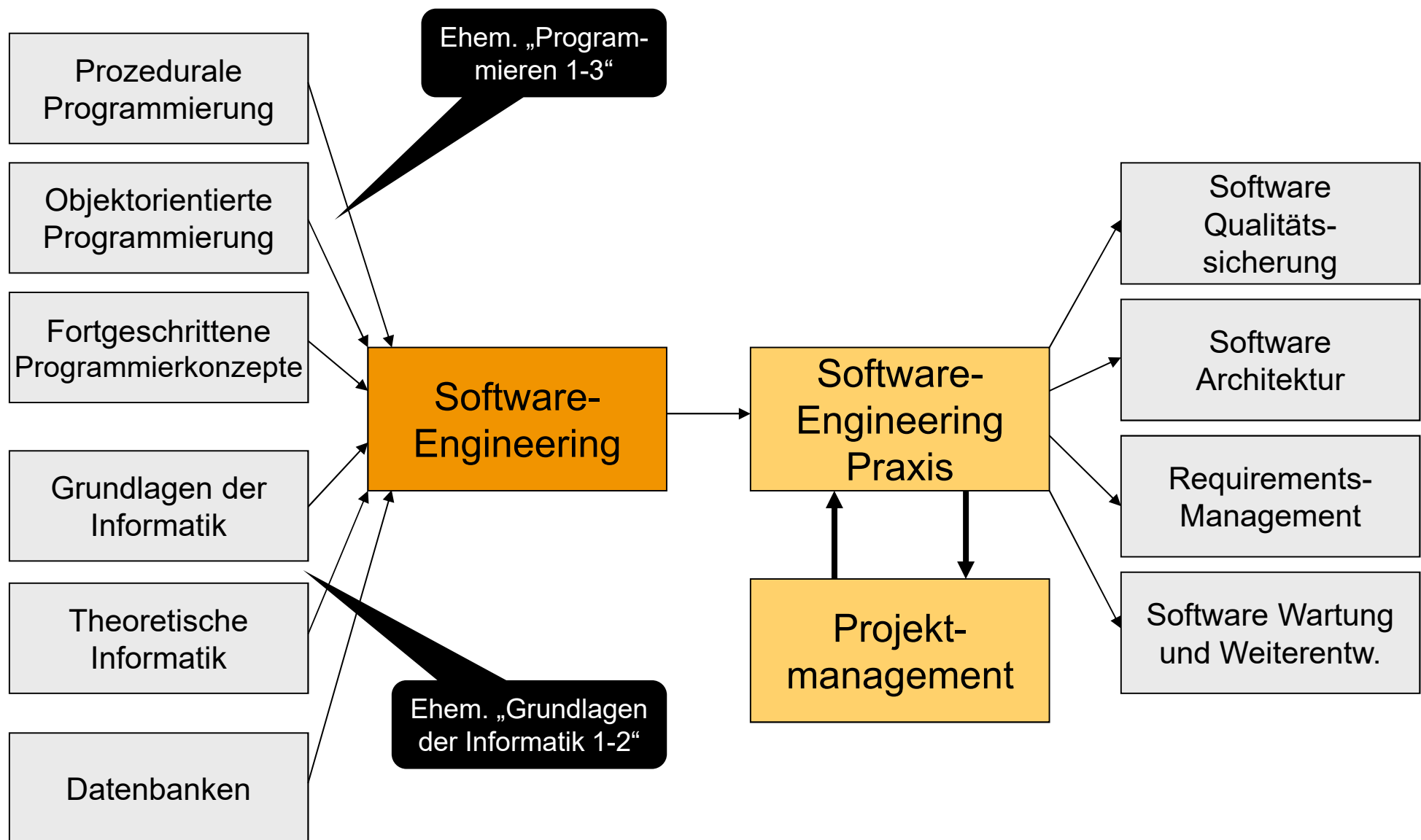
Motivation (2)

Ehem. „SE 1“

Ehem. „SE 2“

- Nach dem Besuch von SE und SEP sollten Sie ...
 - ... Software selbstständig und **systematisch im Team** erstellen können
 - von den **ersten Kundengesprächen**
 - bis zum **Übergang zur Wartung**
- ... Grundlagen erlangen, unabhängig von bestimmten Domänen (z.B. Technik oder Wirtschaft)

Einordnung der Vorlesung



Organisatorisches

- Vorlesungen
 - 2 Wochenstunden
 - Donnerstags, 1. Stunde (AZ 1.50)
- Übungen
 - 2 Wochenstunden
 - Aufteilung in **vier** Gruppen
 - Donnerstags, 2., 3., 4. oder 5. Stunde (B 0.07)
 - Anmeldung zu einer Übungsgruppe über Learning Campus

Bis auf weiteres, solange keine Präsenz-Veranstaltungen möglich sind:

Vorlesungen

- Donnerstags, 9:00 Uhr
- BBB-Konferenzraum über Learning Campus

Übungen

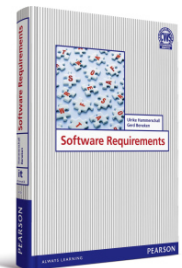
- Aufteilung in Übungs-Teams (max. 6 Personen)
- Team-Wahl über Learning Campus (nur ein Angebot → nicht obligatorisch)
- Selbstständige Bearbeitung der Aufgaben jeweils im Team
- Bitte nur die Studenten in Team organisieren, die auch tatsächlich die Übungen aktiv und gemeinsam bearbeiten möchten!

- Vorlesungs- und Übungsunterlagen
 - Download über Learning Campus
 - <https://learning-campus.th-rosenheim.de/course/view.php?id=1041>
- Hinweise zur Leistungserbringung
 - Klausur zu Semesterende
 - 90 Minuten
 - Keine Unterlagen!

Zeitraum für schriftliche Prüfungen:
Voraussichtlich in den beiden
letzten Wochen im September.

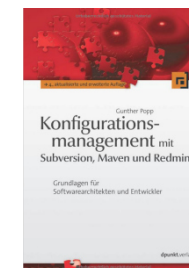
Literatur (1)

- Helmut Balzert: Lehrbuch der Softwaretechnik – Basiskonzepte und Requirements Engineering, Spektrum Verlag, 3. Auflage, 2009
- Ian Sommerville: Software Engineering, Pearson-Studium, 8. Auflage, 2007
- Ulrike Hammerschall, Gerd Beneken: Software Requirements, Pearson-Studium, 2013

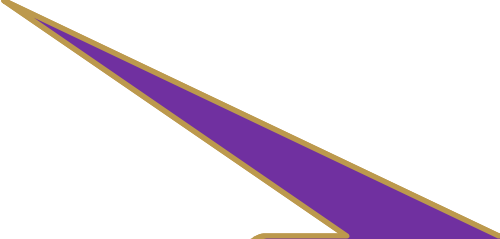


Literatur (2)

- Bernd Oestereich: Analyse und Design mit der UML 2.5 – Objektorientierte Softwareentwicklung, 10. Auflage, 2012
- Schatten, A., et al.: Best Practice Software Engineering, Spektrum Verlag, 2010
- Balzert, Helmut: Lehrbuch der Softwaretechnik – Softwaremanagement, 2. Auflage, 2008
- Popp, Gunther: Konfigurationsmanagement, dpunkt, 2013
- Preißel, René und Stachmann, Bjørn: Git, dpunkt, 2015



- 01 Einführung
- 02 Prozessmodelle
- 03 Konfigurationsmanagement
- 04 Requirements Engineering
- 05 Modellierung
- 06 Qualitätsmanagement



Kapitel 03 werden wir vorziehen.
→ Infrastruktur für Übungs-Teams

- Nach dieser Vorlesungseinheit ...
 - ... können Sie die wichtigsten Begriffe im Umfeld des Software Engineerings definieren
 - ... haben Sie ein Verständnis für wichtige softwarespezifische Aspekte

- Exemplarische Auswahl von Definitionen
 - Sammelbezeichnung für Programme, die für den Betrieb von Rechensystemen zur Verfügung stehen, einschließlich der zugehörigen Dokumentation (*Brockhaus*)
 - Menge von Programmen oder Daten zusammen mit begleitenden Dokumenten, die für ihre Anwendung notwendig oder hilfreich sind (*Hesse et al.*)
 - Computer programs, procedures, rules, and possibly associated documentation and data pertaining to the operation of a computer system (*IEEE Standard Glossary of SW Engineering Technology*)
- Software umfassenderer Begriff als **Programm** – zusätzlich **Dokumentation** und zugehörige **Daten**



- Wie unterscheiden sich die Begriffe?
 - Software
 - Software-System
 - Software-Produkt

Sichtweisen auf Software, Rollen

01 Einführung

● Software-System

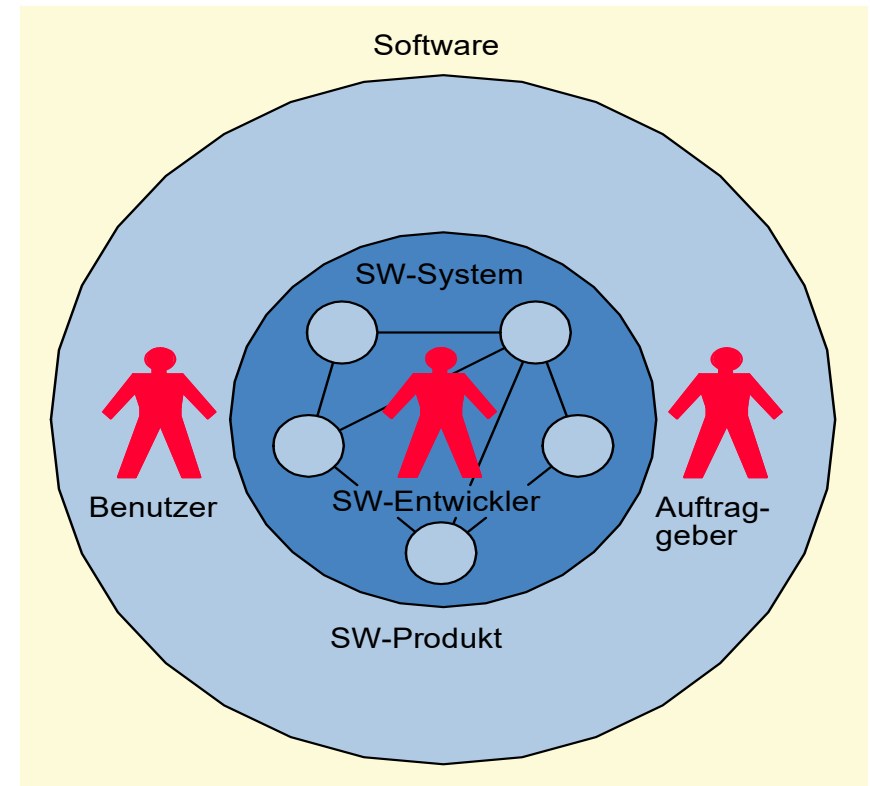
- Innensicht
- Besteht aus interagierenden Softwarekomponenten
- Aspekte der Realisierung im Vordergrund

● Software-Produkt

- Außensicht
- Sichtweise des Auftraggebers
- Aspekte der Nutzung im Vordergrund

● Software

- allgemeiner, neutraler sowie unbestimmter



Quelle: Balzert 2009

Wo kommt Software vor?

01 Einführung

● Beispiele aus dem alltäglichen Leben



Telekommunikation/
Mobile-Services



Auto / Telematik



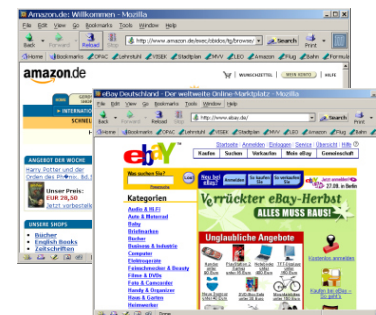
Entertainment



Gesundheit



Haushalt

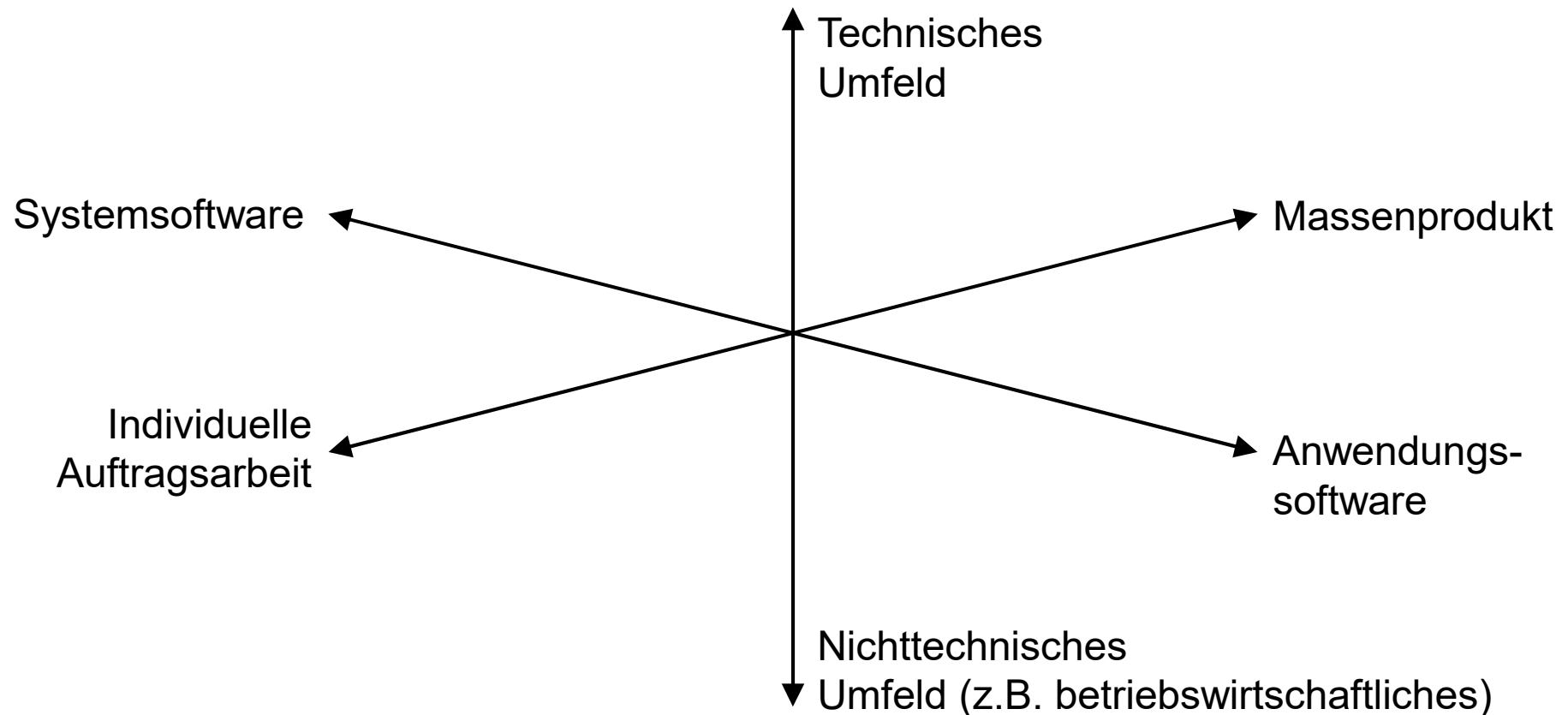


E-Commerce/
E-Government/

...
E-XYZ

Rapide steigender Software-Anteil (an der Wertschöpfung)

- Verschiedene Dimensionen der Klassifizierung möglich



- Weitere Kriterien möglich

- Softwareentwicklung

- Ausschließliche Entwicklung von Software

- Systementwicklung

- Entwicklung eines Systems, das aus Hardware- und Software-Komponenten besteht
- Zusätzliche Randbedingungen müssen berücksichtigt werden

- Entwicklung softwareintensiver Systeme

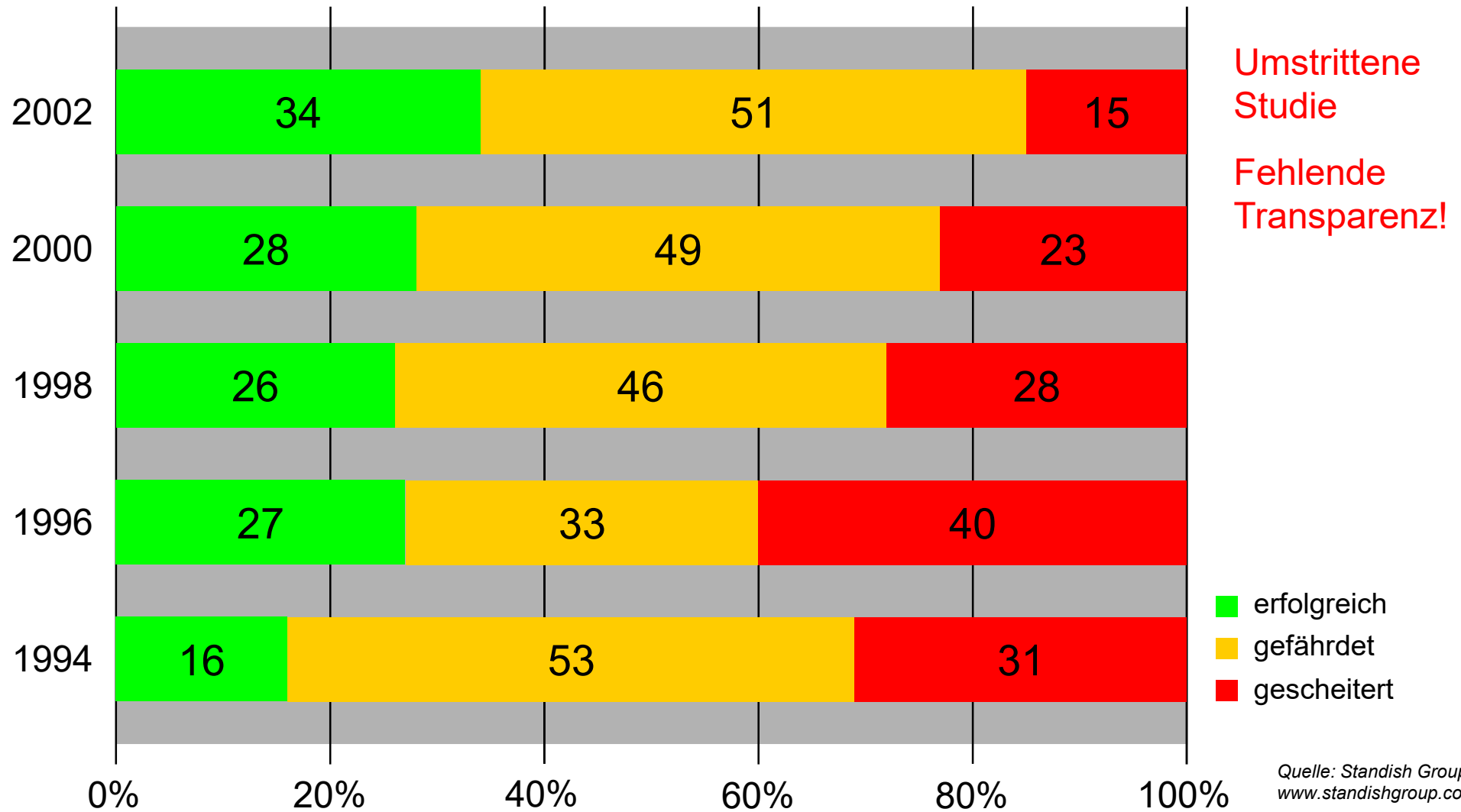
- Wesentliche Eigenschaften werden durch die in das System eingebettete Software realisiert
- Software ist ein wichtiger Systembestandteil



- Zunehmende Bedeutung
 - Wachsende Komplexität
 - Zunehmende Qualitätsanforderungen
 - Zunehmende „Altlasten“
 - Zunehmende „Außer-Haus-Entwicklung“
- Software Entwicklung ...
 - ist nicht einfach
 - wird im Grunde **nicht wirklich beherrscht**
 - offenbart viele ungelöste Herausforderungen

Erfolgsquoten von IT-Projekten

01 Einführung



Schwierigkeiten bei der Softwareentwicklung

01 Einführung

- Software ist immaterielles Produkt
- Software unterliegt keinem Verschleiß, altert aber trotzdem
- Software wird nicht durch physikalische Gesetze begrenzt
- Software ist i.A. leichter/schneller modifizierbar als ein technisches Produkt
- Keine Ersatzteile für Software
- Software ist schwer zu „vermessen“
- Kommunikationsprobleme
- Große Anzahl von Plattformen (Portabilität), Varianten
- Software in Unternehmen eng gekoppelt an Geschäftsprozesse
- Oft fehlen Standards, Techniken, Methoden, Werkzeuge
- ...



Was ist Software Engineering? (1)

01 Einführung

● Exemplarische Auswahl von Definitionen

- „Unter **Softwaretechnik** (engl. *Software Engineering*) versteht man allgemein die **(Ingenieur-) Wissenschaft**, die die kosteneffiziente Entwicklung von qualitativ hochwertiger Software behandelt“.

[Fachgruppe Softwaretechnik GI]

- 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 Std. 610.12 (1990)]



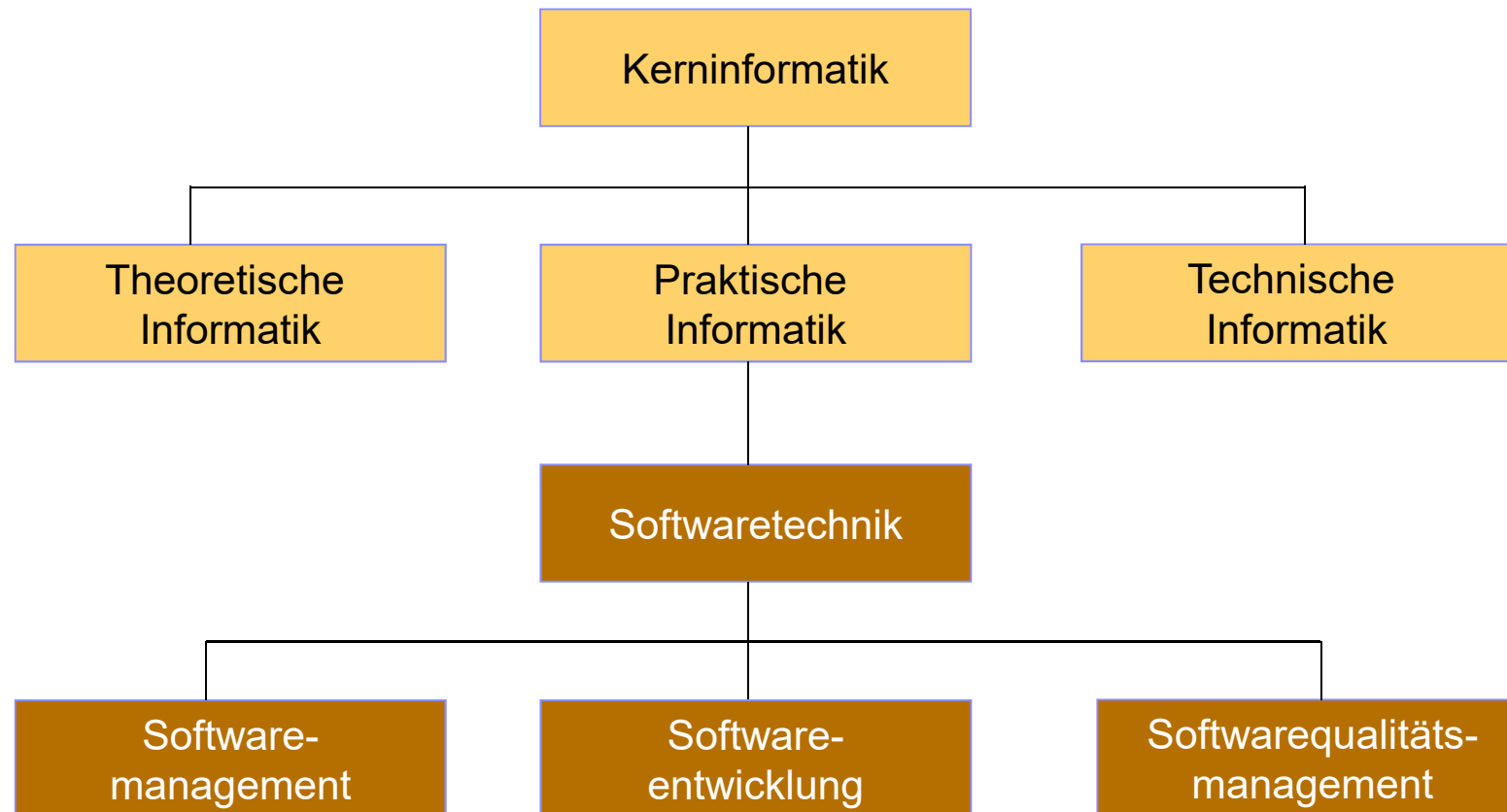
- Exemplarische Auswahl von Definitionen
 - „Softwaretechnik: **Zielorientierte** Bereitstellung und **systematische** Verwendung von Prinzipien, Methoden und Werkzeugen für die **arbeitsteilige, ingenieurmäßige** Entwicklung und Anwendung von **umfangreichen Softwaresystemen**. Zielorientiert bedeutet die Berücksichtigung z.B. von Kosten, Zeit, Qualität.“

[Balzert 2009]



Einordnung – Teildisziplinen

01 Einführung



Quelle: Balzert, 2009



- Softwareentwicklung läuft nicht von alleine ab
- Technischer Entwicklungsprozess muss
 - geplant,
 - organisiert,
 - geleitet und
 - kontrolliert werden
- Software-Projektmanagement stellt einen Teilbereich dar

- **Entwicklungsbegleitendes** Softwarequalitätsmanagement
 - muss für die Sicherstellung der **geforderten** Software**qualität** sorgen
 - Durchführung von **konstruktiven** und **analytischen** Maßnahmen
 - Früher „Softwarequalitätssicherung“ genannt



- Wichtige Charakteristika
 - Umfangreiche Software
 - Arbeitsteilige und ingenieurmäßige Entwicklung
 - Ziele des Software-Kunden sind zu erreichen
 - Ingenieur-Disziplin
 - Kostendenken
 - Qualitätsbewusstsein
 - Praktischer Erfolg - Nutzen
 - Einführung und Beachten von Normen
- Vergleichsweise „junge“ Disziplin → Begriffswelt noch nicht stabil
 - <https://gi.de/service/informatiklexikon/>
 - <https://www.computer.org/web/swebok>



Was sind Anforderungen an einen Softwareingenieur?

01 Einführung

- Definition: **Software Engineer, Softwareingenieur**
 - Good **programmer**, well-versed in data structures and algorithms, and fluent in one or more programming languages (1).
 - Must be familiar with several **design** approaches (2),
 - be able to **translate** vague requirements and desires into precise specifications (3), and
 - be able to **converse** with the user of a system in terms of application rather than 'computers' (4).

Quelle: Ghezzi, Jazayeri, Mandrioli: Fundamentals of Software Engineering, 2002





- Weitere unterstützende Rollen
 - Administrator des Konfigurationsmanagementsystems
 - Technischer Autor
 - Trainer ...