



Software Engineering

Teil 1: Einführung und Grundbegriffe des
Software Engineering

April 2018

Dr. Christian Bartelt



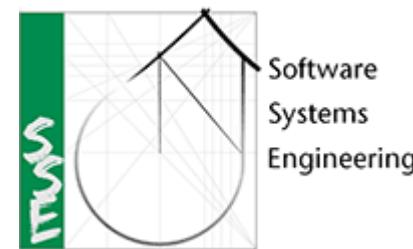
UNIVERSITATEA
BABEŞ-BOLYAI

Universität Mannheim
Institut für Enterprise Systems InES
Schloss
68131 Mannheim / Germany

UNIVERSITY OF
MANNHEIM

Vorlesungsunterlagen

- Die Vorlesungsunterlagen bauen auf der Softwaretechnik I-Vorlesung von Prof. Dr. Rausch an der TU Clausthal auf.



Inhalt

- **Bedeutung von Softwaresystemen**
- **State of the art im Software Engineering**
 - Software-Katastrophen
 - Erfolgsstatistik und –faktoren
- **Grundbegriffe des Software Engineering**
 - Software und Softwaresysteme
 - Projekt
 - Software Engineering

Bedeutung von Softwaresystemen

- Primärbranchen (DV-Dienstleister, Hersteller von Datenverarbeitungsgeräten und -einrichtungen)
 - Rund 10.550 Unternehmen
 - Ca. 300.000 Erwerbstätige
 - Überwiegend kleine Unternehmen mit 1-9 Mitarbeitern
- Sekundärbranchen (Maschinenbau, Elektrotechnik, Fahrzeugbau, Telekommunikation und Finanzdienstleistungen)
 - Rund 8.650 Unternehmen
 - 2,5 Millionen Erwerbstätige
 - Eher mittlere und größere Unternehmen
 - Heutige Produkte ohne Software oft undenkbar

... daß schon jetzt mehr als die Hälfte der Wertschöpfung von Siemens auf Software-Leistungen entfällt. Diese Entwicklung geht weiter ...

Heinrich von Pierer, Siemens AG

Bedeutung von Softwaresystemen

DAIMLERCHRYSLER

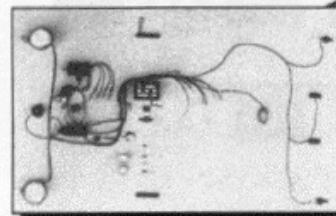
Research & Technology

Automobil im Wandel

Kabelbaum 1949 170V

Ca. 40 Kabel

Ca. 60 Kontaktierungen

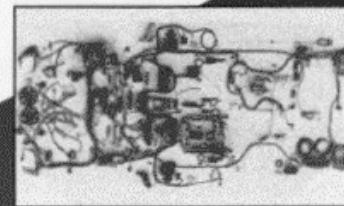
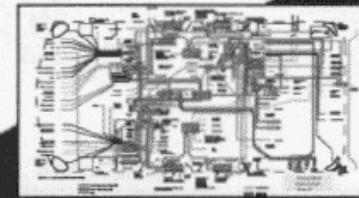


Kabelbaum 1999 S-Klasse

3 Bus-Systeme

ca. 60 ECU's

110 elektrische Motoren



Kabelbaum 1990 S-Klasse

Länge ca. 3 km

• Gewicht ca. 39 kg

Ca. 1900 Kabel

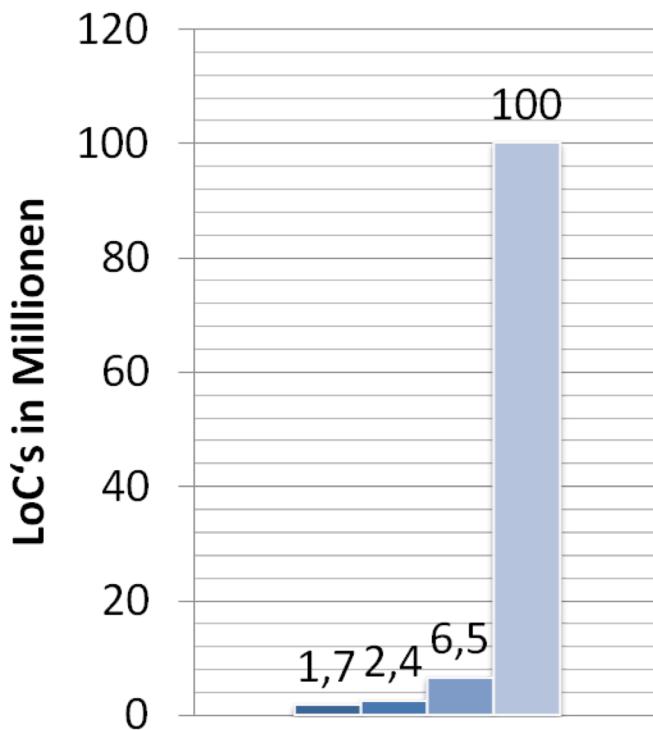
• Ca. 3800 Kontaktierungen

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Bedeutung von Softwaresystemen



- F-22 Raptor
- F-35 Joint Strike Fighter
- Boeing's 787 Dreamliner
- PKW Premiumklasse



F-22



787



PKW

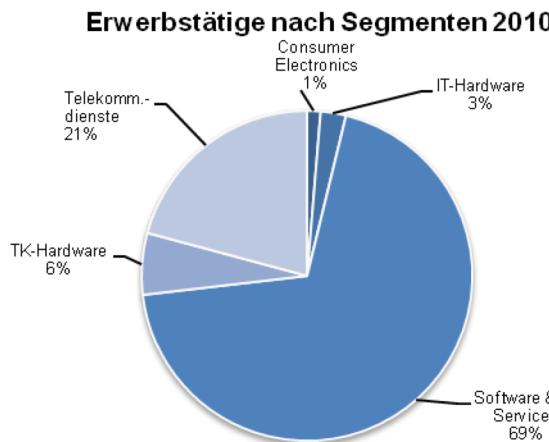
Beispiele: Verbreitung von Softwaresystemen

- Haushalts- und Konsumelektronik
 - Handys, DVD-Player, Digitalkameras → bestehen zu einem wesentlichen Teil aus Softwaresystemen
 - Aber auch Geräte wie Kaffee- und Waschmaschinen
- Automobilindustrie
 - Pro Fahrzeug bis zu 100 Mikrokontroller
 - Mehr als 50% der Pannen → Software beteiligt: Tendenz steigend
 - Prozesse und Produktion ohne Software nicht mehr möglich
- Informationssysteme
 - Finanzen, Medizin, etc. – zwischen 60% - 90% Durchdringung
 - 1 Geschäftsprozess integriert bis zu 15 Großanwendungen!



Jobs, Jobs, Jobs in der Informatik

- Ende 2012 gab es laut einer VDI-Studie über 25.000 offene Stellen, der Großteil davon in den boomenden Sektoren Software und IT-Services – Ein Anstieg um fast 50% innerhalb eines Jahres
- Gesamtbeschäftigung in ITK-Branche bleibt stabil
- Die VDI meldete zur CeBit2017 einen starken Mangel an IT-Fackräfte



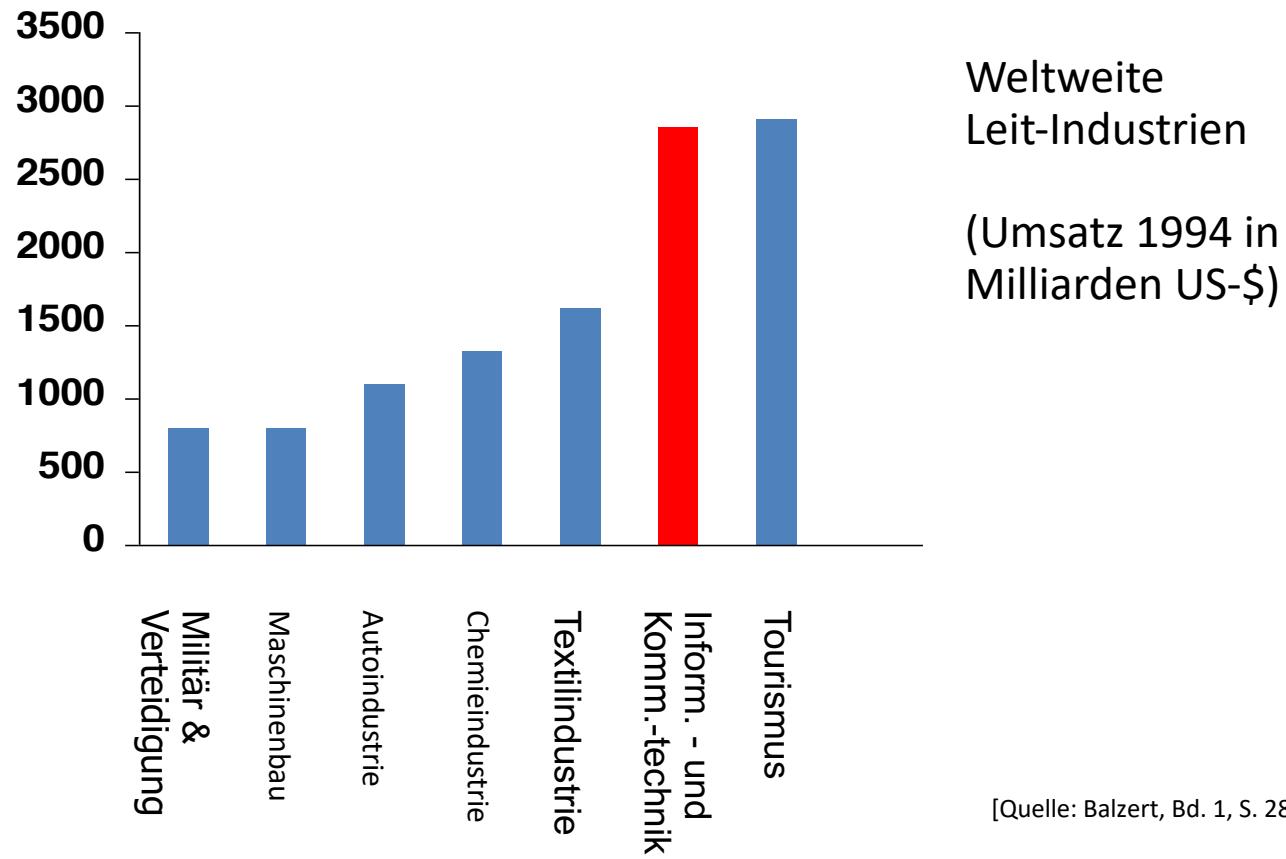
Erwerbstätige* in der ITK-Branche Deutschland (in Tsd.)						Wachstum			
	2007	2008	2009	2010	2011**	2008	2009	2010	2011
Summe ITK + CE	830,8	835,2	835,5	847,7	857,7	0,5%	0,0%	1,5%	1,2%
CE	12,1	12,1	10,9	11,1	10,7	0,1%	-10,3%	2,3%	-4,0%
Summe ITK	818,8	823,1	824,6	836,6	847,0	0,5%	0,2%	1,5%	1,2%
Informationstechnik	550,6	577,7	587,3	609,2	624,6	4,9%	1,7%	3,7%	2,5%
IT-Hardware***	26,3	26,5	23,8	21,0	19,5	0,6%	-10,1%	-11,8%	-7,1%
Software & IT-Services	524,3	551,3	563,5	588,2	605,1	5,2%	2,2%	4,4%	2,9%
Telekommunikation	268,2	245,4	237,3	227,4	222,4	-8,5%	-3,3%	-4,2%	-2,2%
TK-Hardware	63,5	57,2	53,6	51,1	50,0	-9,9%	-6,3%	-4,8%	-2,1%
Telekommunikationsdienste	204,6	188,1	183,7	176,4	172,4	-8,1%	-2,4%	-4,0%	-2,2%

* jeweils zum Jahresende, einschließlich Selbständige

** Prognose

*** Neue wirtschaftsfachliche Zuordnung eines größeren Betriebes im Jahr 2009. Werte für 2007 - 2008 wurden rückwirkend bereinigt.

Wirtschaftliche Bedeutung von Softwaresystemen



Inhalt

- **Bedeutung von Softwaresystemen**
- **State of the art im Software Engineering**
 - **Software-Katastrophen**
 - Erfolgsstatistik und – faktoren
- **Grundbegriffe des Software Engineering**
 - Software und Softwaresysteme
 - Projekt
 - Software Engineering
 - Vergleich mit anderen Ingenieursdisziplinen
 - Ethische und rechtliche Verantwortung

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Mariner 1



- 22. Juli 1962, Cape Canaveral / Florida
- Start der ersten amerikanischen Venussonde Mariner 1
- Trägerrakete Atlas-Agena B (NASA, 15. AAB-Start)



Explosion von Mariner 1

Ausschnitt aus dem FORTRAN-Programm

```
...  
IF (TVAL .LT. 0.2E-2) GOTO 40  
DO 40 M = 1, 3  
W0 = (M-1)*0.5  
X = H*1.74533E-2*W0  
DO 20 N0 = 1, 8  
EPS = 5.0*10.0**(N0-7)  
CALL BESJ(X, 0, B0, EPS, IER)  
IF (IER .EQ. 0) GOTO 10  
20 CONTINUE  
DO 5 K = 1, 3  
T(K) = W0  
Z = 1.0/(X**2)*B1**2+3.0977E-4*B0**2  
D(K) = 3.076E-2*2.0*(1.0/X*B0*B1+3.0977E-4*  
*(B0**2-X*B0*B1))/Z  
E(K) = H**2*93.2943*W0/SIN(W0)*Z  
H = D(K)-E(K)  
5 CONTINUE  
10 CONTINUE  
Y = H/W0-1  
40 CONTINUE ...
```

- Fehler: Komma gegen Punkt vertauscht in der Zeile
DO 5 K = 1. 3; korrekt wäre: DO 5 K = 1, 3
- Wirkung:
 - Wertzuweisung an eine nicht deklarierte Variable: DO5K = 1.3 (Kein Problem in FORTRAN)
 - Kein Durchlauf der (nicht vorhandenen) Schleife
- Folgen:
 - Abweichung der Trägerrakete von der vorgesehenen Flugbahn
 - Zerstörung der Rakete nach 290 Sekunden
 - Kosten: US\$ 18,5 Millionen
- Ursache: Programmiersprache FORTRAN
 - Blanks (Zwischenräume) in Namen und Zahlen erlaubt
 - Variablen-Deklarationen nicht notwendig
 - Strukturierte Schleifen (while ...) nicht möglich

Ariane 5

- **4. Juni 1996, Kourou / Frz. Guyana:**
Jungfernflug der neuen europäischen
Trägerrakete Ariane 5



```
...
declare
    vertical_veloc_sensor: float;
    horizontal_veloc_sensor: float;
    vertical_veloc_bias: integer;
    horizontal_veloc_bias: integer;
    ...
begin
    declare
        pragma suppress(numeric_error, horizontal_veloc_bias);
    begin
        sensor_get(vertical_veloc_sensor);
        sensor_get(horizontal_veloc_sensor);
        vertical_veloc_bias := integer(vertical_veloc_sensor);
        horizontal_veloc_bias := integer(horizontal_veloc_sensor);
        ...
    exception
        when numeric_error => calculate_vertical_veloc();
        when others => use_irs1();
    end;
end irs2;
```

Ariane 5

Ursache:

- 37 Sekunden nach Zünden der Rakete (30 Sekunden nach Liftoff) erreichte Ariane 5 in 3700 m Flughöhe eine Horizontal-Geschwindigkeit von 32768.0 (interne Einheiten). Die Umwandlung in eine ganze Zahl führte daher zu einem Überlauf, der jedoch nicht abgefangen wurde. Der Ersatzrechner hatte das gleiche Problem schon 72 msec vorher und schaltete sich sofort ab. Daraus resultierte, dass Diagnose-Daten zum Hauptrechner geschickt wurden, die dieser als Flugbahndaten interpretierte. Daraufhin wurden unsinnige Steuerbefehle an die seitlichen, schwenkbaren Feststoff-Triebwerke, später auch an das Haupttriebwerk gegeben. Die Rakete drohte auseinanderzubrechen und sprengte sich selbst.

Schaden:

- 250 Millionen DM Startkosten
- 850 Millionen DM Cluster-Satelliten
- 600 Millionen DM für nachfolgende Verbesserungen

Ariane 5

- Analyseproblem: Nicht erkannt, dass die korrekte Funktion des **wiederverwendeten** Moduls an Rahmenbedingungen geknüpft war, die für die Ariane 5 nicht galten (*Requirements Tracing*)
- Entwurfsproblem: Homogene Redundanz für Hardware **und** Software
 - Prinzip aus der Hardware-Sicherheitstechnik, das für Software nicht funktioniert
- Realisierungsproblem: Keine sinnvolle Propagation von Fehlverhaltenscodes, sondern Totalabschaltung
- Prüfung
 - Keine intensive, systematische Prüfung, da die Software bei der Ariane 4 problemlos funktioniert hatte (Betriebsbewährtheit)

=> Das ist kein **monokausales** Problem, ...
... und daher existiert keine **einfache** Lösung.

Software-Katastrophe: Patriot-Rakete (1)

Ein fataler Software-Fehler im Golfkrieg:

“During the Gulf war, a computer failure was responsible for the failure of a patriot missile to stop a scud missile that hit an American military barracks in Dharan ... 28 dead ...”



[Quelle: ACM SIGSOFT Software Engineering Notes, vol. 16, no. 3 (1991), S.19f]

Ursache:

- 💣 der Steuercomputer lief 4 Tage ununterbrochen (statt der vorgeschriebenen maximal 14 Stunden)
- 💣 dadurch lief das interne Timer-Register über 24 Bit hinaus und es entstanden Rundungsfehler bei der Bahnberechnung
- 💣 wäre das Timer-Intervall $1/8$ statt $1/10$ Sekunde gewesen hätte es keine Rundungsfehler gegeben
- 💣 das Intervall wurde entgegen der ursprünglichen Programmierung nachträglich von einem Manager auf $1/10$ Sek. geändert

Software-Katastrophe: Patriot-Rakete (2)

Schlussfolgerungen aus dem „Patriot-Missile“ -Beispiel:

- 👉 Software soweit wie möglich gegen Fehlbedienungen absichern (z.B. Warnungen nach 14 Stunden Laufzeit)
- 👉 Software soweit wie möglich gegen typische Programmierfehler absichern (Zählerüberläufe etc. durch geeignete Plausibilitätsprüfungen und „exception handling“ abfangen)
- 👉 Wichtige Entwurfsentscheidungen sind für spätere Wartung zu dokumentieren („1/8 Sek. Timer-Intervall wurde gewählt, weil...“)
- 👉 Für Software-Entwicklung feste Vorgehensweisen und Zuständigkeiten festlegen (um ad-hoc Änderungen durch unqualifizierte Personen zu verhindern)

[Quelle: Mark Minas, Vom Bild zum Programm, S.12f]

Software-Katastrophe: Kein Einzelfall (1)

- 1981: US Air Force Command & Control Software überschreitet Kostenvoranschlag fast um den Faktor 10: **3,2 Mio. US-\$**.
- 1987-1993: Integration der kalifornischen Systeme zur Führerschein- und KFZ-Registrierung abgebrochen:
44 Mio. US-\$.
- 1992: Integration des Reservierungssystems SABRE mit anderen Reservierungssystemen abgebrochen: **165 Mio. US-\$**.
- 1997: Entwicklung des Informationssystems SACSS für den Staat Kalifornien abgebrochen:
300 Mio. US-\$.
- 1994: Eröffnung des Denver International Airport um 16 Monate verzögert wegen Softwareproblemen im Gepäcktransport-System: **655 Mio. US-\$**.
- 2005: Das deutsche Maut Erfassungssystem "Toll Collect" konnte nur mit erheblicher Verzögerung (Vertragsabschluss: September '02, geplanter Starttermin: 31. August 2003), am 1. Januar 2005 in technisch reduzierter Form in Betrieb genommen werden: **~6,5 Mrd. €**.
- 2006: Airbus muss Auslieferung der verteilt produzierten A380-Maschinen um 1 Jahr verschieben: Die Standorte verwendeten keine einheitliche Designsoftware, wodurch Kabel nicht zusammenpassten: **~6 Mrd. €**

Software-Katastrophe: Kein Einzelfall (2)

- 1988: Ein Airbus schießt über die Landebahn hinaus, da sich bei Aquaplaning die Schubumkehr nicht einschalten ließ.
- 1999: Verlust der Sonde "Mars Climate Orbiter" wegen falscher Einheitenumrechnung.
- 1999: 20.500 3er BMWs müssen wegen eines Software-Bugs in der Airbag-Steuerung zurückgerufen werden. 50% aller Autopannen sind bereits auf Ausfälle der Bordelektronik zurückzuführen, Tendenz steigend.
- 2002: Aufgrund eines Softwareproblems konnten mit Postbank-EC-Karten bei allen anderen Geldinstituten außer der Postbank selbst mit beliebigen Pinodes Euro abgehoben werden, ohne dass das Sparkonto mit der abgehobenen Summe belastet wurde.
- 2004: Siemens S65 wird wegen Softwarefehlern, die Hörschäden verursachen können, aus dem Handel genommen.

Inhalt

- **Bedeutung von Softwaresystemen**
- **State of the art im Software Engineering**
 - Software-Katastrophen
 - **Erfolgsstatistik und –faktoren**
- **Grundbegriffe des Software Engineering**
 - Software und Softwaresysteme
 - Projekt
 - Software Engineering

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

IT-Katastrophen – nur Einzelfälle?

- CHAOS Report
 - Jährlicher Bericht seit 1994 über den Erfolg von IT-Projekten
 - Es wurden ca. 100.000 IT-Projekte in den USA untersucht
 - Herausgeber: Standish Group International, Inc.
- CHAOS Report ordnet IT-Projekte in drei Kategorien ein
 - **Successful:** Projekt wurde innerhalb der vorgegebenen Zeit und Budget abgeschlossen. Projektergebnis ist im Einsatz und erfüllt alle Anforderungen.
 - **Challenged:** Projekt ist abgeschlossen. Projektergebnis ist im Einsatz. Zeit, Budget oder Leistung sind aber nicht im vorgegebenen Umfang.
 - **Failed:** Das Projekt wurde vorzeitig abgebrochen oder das Projektergebnis wurde nie eingesetzt.

Erfolgsstatistik von IT-Projekten

	Succeeded	Failed	Challenged
1994	16%	31%	53%
1996	27%	40%	33%
1998	26%	28%	46%
2000	28%	23%	49%
2002	34%	15%	34%
2004	29%	18%	53%
2006	35%	19%	46%
2008	32%	24%	44%



The Modern Resolution (OnTime, OnBudget, with a satisfactory result) of all software projects from FY2011–2015 within the new CHAOS database. Please note that for the rest of this report CHAOS Resolution will refer to the Modern Resolution definition not the Traditional Resolution definition.

[Quelle: CHAOS Report, Standish Group International, Inc.]

Unsere IT-Landschaft bildlich dargestellt...

nicht akzeptabel !



Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Die 10 wichtigsten Erfolgsfaktoren

1.	User involvement
2.	Executive management support
3.	Clear business objectives
4.	Optimizing scope
5.	Agile process
6.	Project manager expertise
7.	Financial Management
8.	Skilled resources
9.	Formal methodology
10.	Standard tools and infrastructure

[Quelle: CHAOS Report, Standish Group International, Inc.]

Inhalt

- **Bedeutung von Softwaresystemen**
- **State of the art im Software Engineering**
 - Software-Katastrophen
 - Erfolgsstatistik und –faktoren
- **Grundbegriffe des Software Engineering**
 - **Software und Softwaresysteme**
 - Projekt
 - Software Engineering

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Was ist Software?

- Software (engl., eigtl. »weiche Ware«), Abk. SW, Sammelbezeichnung für Programme, die für den Betrieb von Rechensystemen zur Verfügung stehen, einschl. der zugehörigen Dokumentation (*Brockhaus Enzyklopädie*)
- Software: die zum Betrieb einer Datenverarbeitungsanlage erforderlichen nichtapparativen Funktionsbestandteile (*Fremdwörter-Duden*).

Was ist Software?

Definition: Software

Software ist eine Menge von Programmen und Daten zusammen mit begleitenden Dokumenten, die für die Anwendung notwendig oder hilfreich sind.

Beispiele

- Microsoft Office
- Linux
- Steuersoftware einer Waschmaschine

Bemerkung: Programme sind "weiche" Ware

- immateriell
- "vermeintlich" leicht änderbar
- kein Verschleiß, keine "klassische" Wartung

Was sind Softwareintensive Systeme?

Definition: Softwareintensives System

Ein softwareintensives System besteht aus einer Menge von Bausteine, die so zusammengestellt sind, dass sie gemeinsam den Zweck des Systems erfüllen. Bausteine, die vollständig oder zu wesentlichen Teilen aus Software bestehen, übernehmen dabei essenzielle Aufgaben zur Erfüllung des Systemzwecks. Der Softwareanteil des Systems besteht dabei aus einer Menge von Programmen und weiteren Prozeduren und Daten sowie zugehöriger Dokumentation.

Beispiele

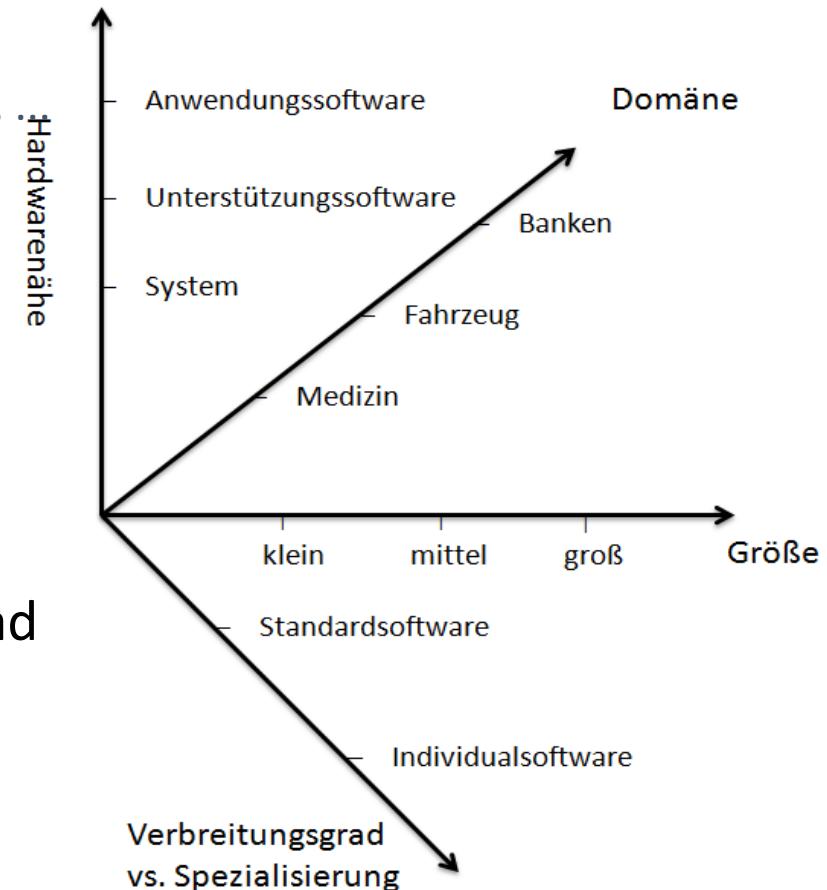
- Flugzeug
- Kontoauszugdrucker
- Versicherungs-verwaltungssystem

Gegenbeispiele

- Kugelschreiber

Mehrdimensionale Kategorisierung von softwareintensiven Systemen

- Nach Hardwarenähe
 - Anwendungssysteme: Microsoft Office, ...
 - Middleware: EJB-Server, DB2
 - Systemsoftware: Linux, Treiber, ...
- Nach Größe
 - Klein: Gerätetreiber mit 5.000 LOC
 - Mittel: Mobiltelefon mit 200.000 LOC
 - Groß: SAP R/3 mit 6 MIO LOC
- Verbreitung und Spezialisierungsgrad
 - Standardsoftware: Windows XP, SAP, ...
 - Individualsoftware: Banksystem, Eingebettete SW



Einfache Kategorisierung von softwareintensiven Systemen

- **Eingebettete Systeme:** beinhalten Software, die in physikalischen Gegenständen eingebettet ist.

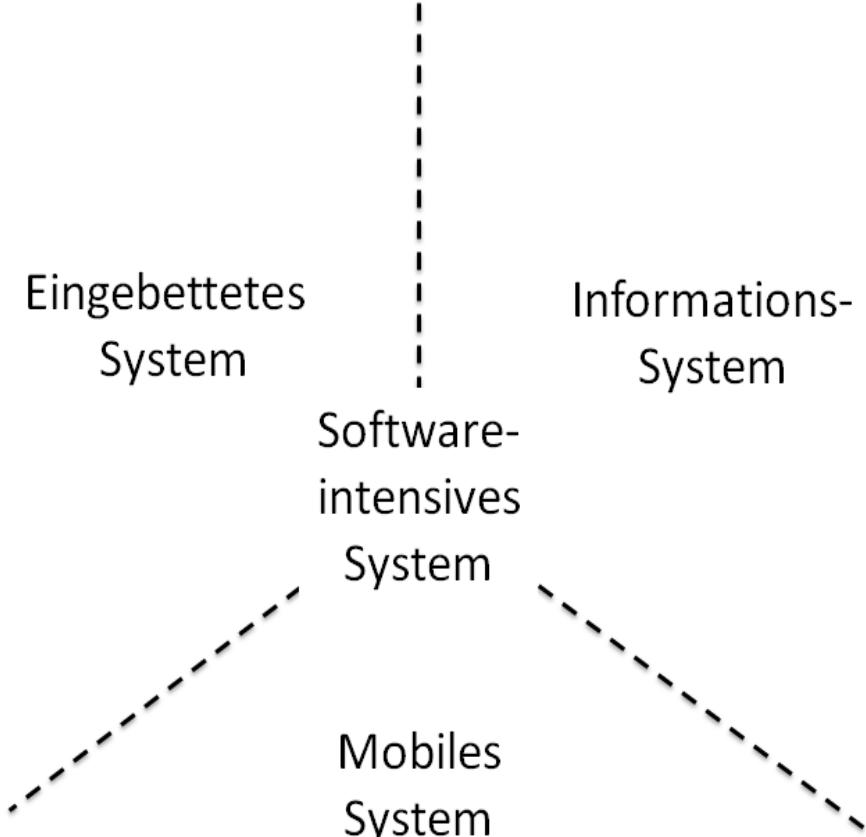
- Airbag-Steuerungen
 - Parkassistenten in Fahrzeugen

- **Informationssysteme:** die Verwaltung und Verarbeitung der Informationen steht im Vordergrund.

- SAP-Systeme

- **Mobile Systeme:** mobile, (semi)-autonome und personenspezifische Einheiten mit hohen Interaktionsanforderungen.

- Smartphones



Inhalt

- **Bedeutung von Softwaresystemen**
- **State of the art im Software Engineering**
 - Software-Katastrophen
 - Erfolgsstatistik und –faktoren
- **Grundbegriffe des Software Engineering**
 - Software und Softwaresysteme
 - **Projekt**
 - Software Engineering

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Was ist ein Projekt?

Definition: Projekt

Ein Projekt ist ein einmaliges Vorhaben mit einem gewissen Risiko. Ein vorgegebenes Ziel muss innerhalb einer vorgegebenen Zeit unter Einsatz von vorhandenen, meist beschränkten Mitteln erarbeitet werden.

Beispiele

- Umzug
- Mondlandung
- Entwicklung von Software-Systemen

Gegenbeispiele

- Betrieb eines Rechenzentrums (SCI)

Klassifizierungen von Projekten (1)

- Größe und Dauer
 - Klein: 1 PJ, 1-2 Bearbeiter, Entwicklung für Eigenbedarf
 - Mittel: 1-10 PJ, 3-10 Bearbeiter, Compiler, Steuerprogramme, Entwicklung für Kunden
 - Groß: 5-50 PJ, 10 bis 30 Bearbeiter, Datenbanken, Spiele, Individual-SW-Systeme
 - Riesig: 50-5.000 PJ, 20-1000 Bearbeiter, Gesamtlösungen für Unternehmen

Klassifizierungen von Projekten (2)

- Zielsetzung und Ergebniserwartung
 - Kurzfristige ökonomische Interessen
 - Strategisches Investitionsprojekt
 - Forschungsprojekt
- Anwendungsdomäne
 - Finanzwesen
 - Verwaltung
 - Militär
- Technologien
 - Programmiersprachen, Hardware, Systemsoftware

Inhalt

- **Bedeutung von Softwaresystemen**
 - **State of the art im Software Engineering**
 - Software-Katastrophen
 - Erfolgsstatistik und – faktoren
 - **Grundbegriffe des Software Engineering**
 - Software und Softwaresysteme
 - Projekt
- **Software Engineering**

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Was ist Ingenieurswesen?

Definition: Engineering

Engineering ist die Anwendung von naturwissenschaftlichen Erkenntnissen und praktischen Erfahrungen, um für die Menschheit sinnvolle Dinge zu entwickeln und bereit zu stellen.

Beispiele

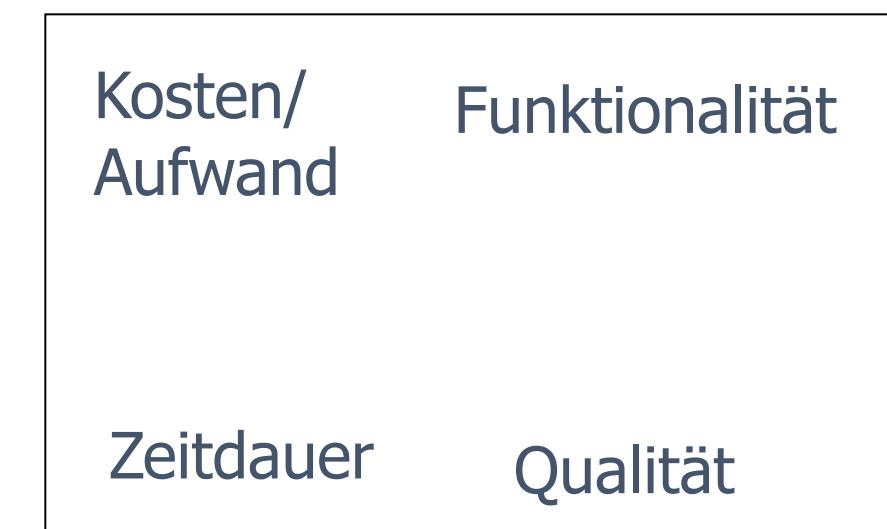
- Maschinenbau
- Bauingenieurwesen
- Architektur
- Software Engineering

Was will Software Engineering?

- Softwaresysteme sind ein zentrales Rückrat unserer Gesellschaft!
- Die Fähigkeit IT-Projekte durchzuführen muss verbessert werden!

→ Ziel: Verbesserung der Beherrschbarkeit des "magischen Viereck"

- Ansatz: Disziplin Software Engineering



Was ist Software Engineering?

Definition: Software Engineering

Software Engineering ist die zielorientierte Bereitstellung und Verwendung von **systematischen, ingenieurmäßigen und quantifizierbaren Vorgehensweisen für Entwicklung, Betrieb, Wartung und Stilllegung von Software-basierten Systemen.**

Zielorientiert bedeutet dabei die Berücksichtigung von Zeit, Kosten und Leistung.

Bedeutung von
Softwaresystemen

State of the art im Software
Engineering

Grundbegriffe des Software
Engineering

Was ist Software Engineering nicht?

- Programmierkurs, Programmier - Know - How
- AnwenderInnen-Kurs
- abstrakte Wissenschaft
- "A fool with a tool is still a fool"
- "Silver Bullet"