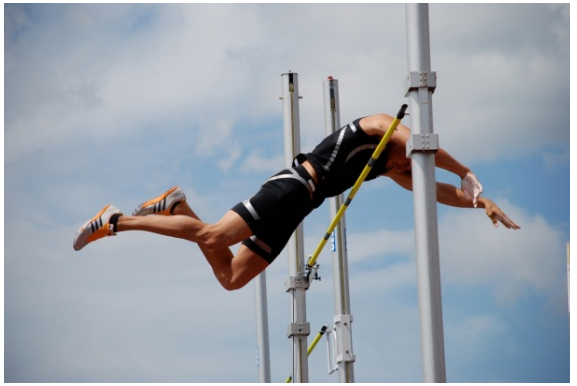
Software Engineering Basics

- Motivation
- Begriffe
- (Einige) Definitionen

HOCHSCHULE DER WIRTSCHAFT FÜR MANAGEMENT
UNIVERSITY OF APPLIED MANAGEMENT STUDIES

Ziele

- Sie kennen Herausforderungen bei der Entwicklung großer (und nützlicher) Systeme
- Sie kennen den Status der Software-Entwicklung heute sowie den Nutzen von Software-Engineering
- Sie kennen grundlegende Definitionen des Software-Engineering



Fragerunde

- Wer von Ihnen hat schon Software entwickelt?

Fragerunde

- Was war Ihr größtes Software-Projekt (das Sie beobachten konnten)?
- Wie groß war es?
- Was heißt, ein Software-Projekt ist groß?
- Wie misst man die Größe?
  - Anzahl Entwickler
  - Entwicklungsdauer
  - Lebensdauer
  - Lines of Code

Fragerunde

- Was sind typische Software-Projekt-Größen? (Anzahl Entwickler, Lines of Code)

Software	LoC	Entwickler
Linux Kernel	9.200.000	2399
Microsoft Office Mac OS X	30.000.000	70
Windows XP	40.000.000	?
SAP ERP	240.000.000	?
SAP Netweaver	42.000.000	1500
World of Warcraft	5.500.000	32

Diskussion

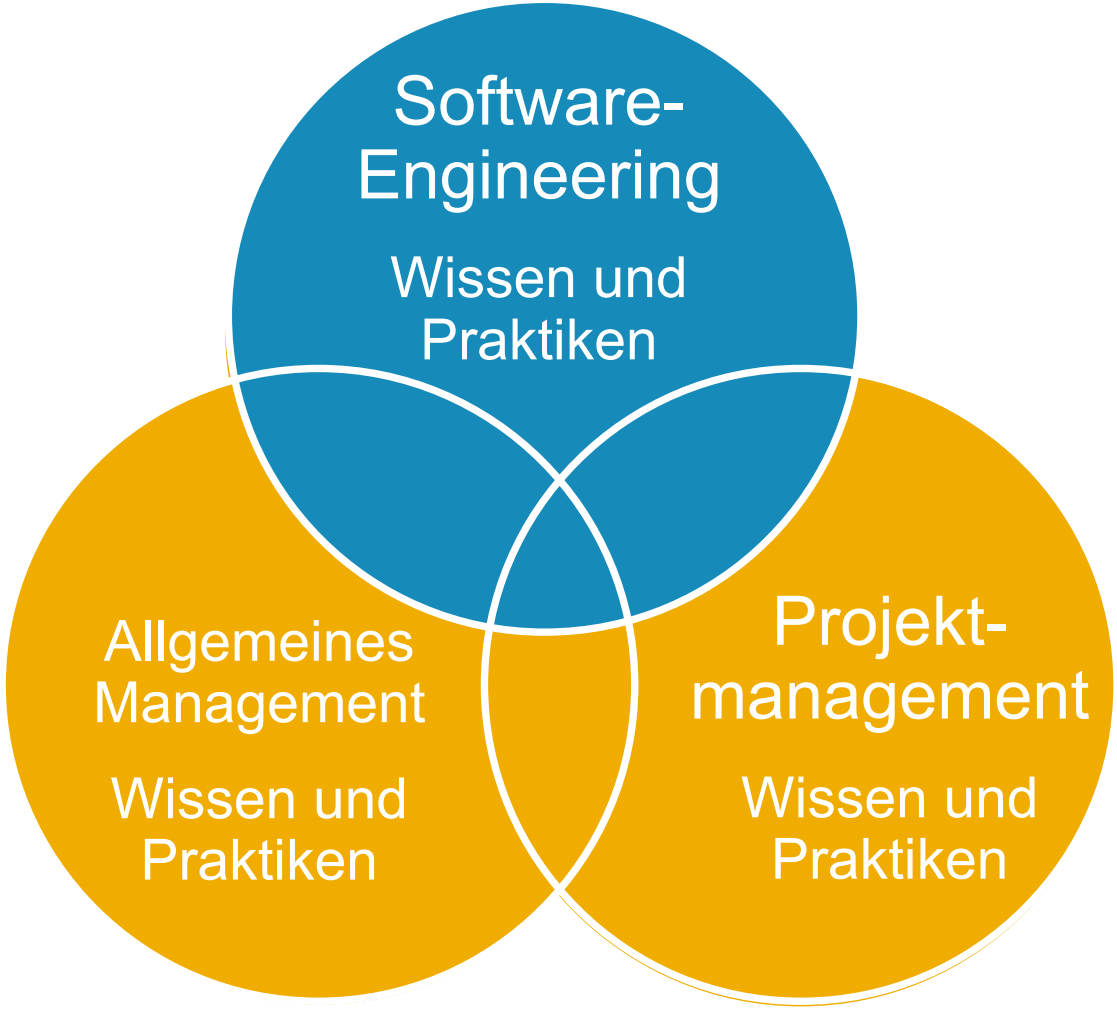
- Warum haben wir gerade über „Größe“ geredet – hat Größe Einfluss auf die Software-Entwicklung?

- Nennen Sie 5 Unterschiede zwischen der Entwicklung
  - alleine am PC ein kleines Programm zur Vereinsverwaltung
  - im Team eine neue Komponente für PowerPoint 2016

Fragerunde

- Was heißt, eine Software ist gut?
  * nützlich und nutzbar
  * zuverlässig
  * flexibel
  * kostengünstig
  * rechtzeitig verfügbar

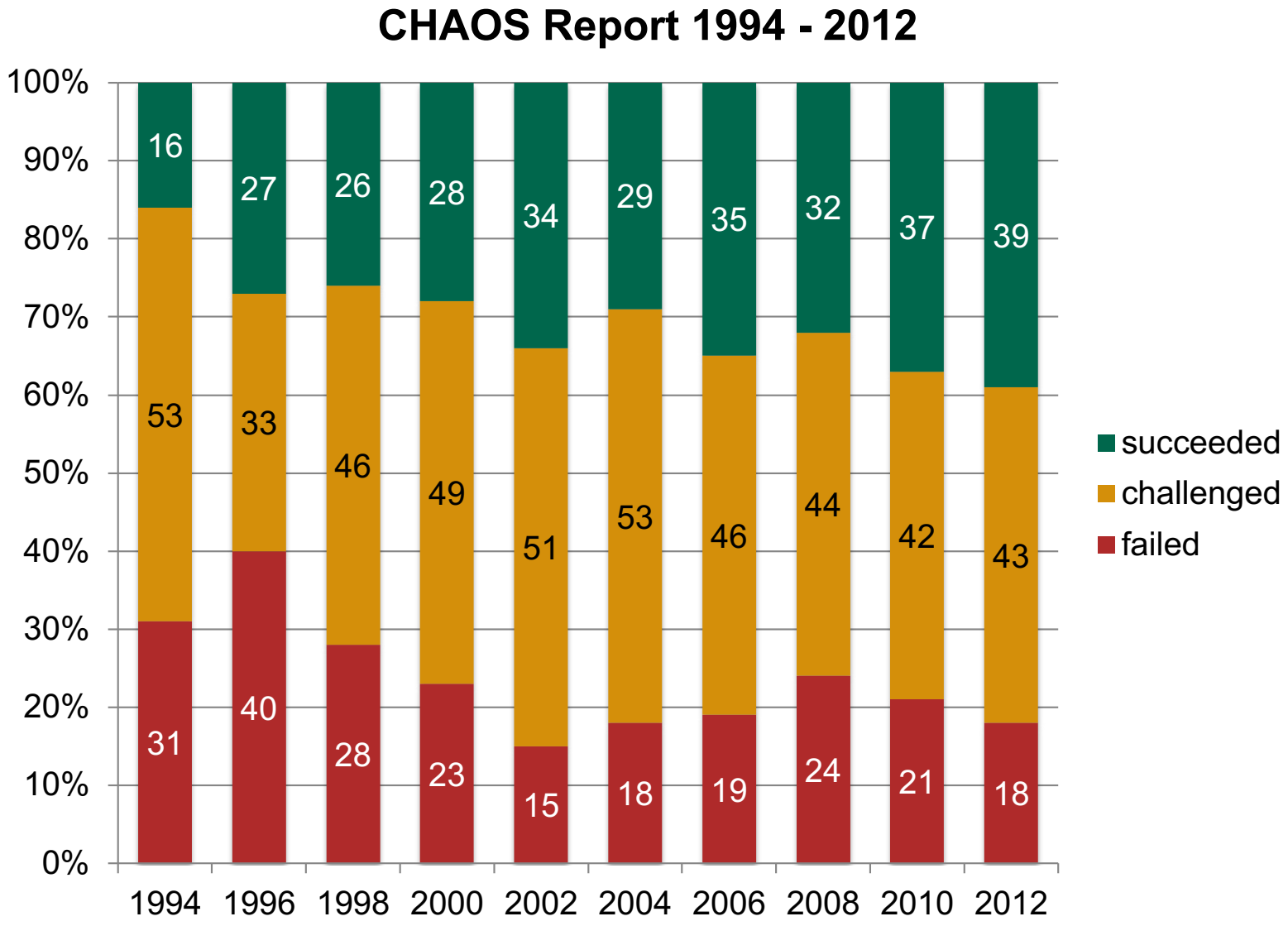
Software-Technik hilft mit unterschiedlichen Disziplinen, große gute Programme zu entwickeln



Brauchen wir Software Engineering? Ja… (1)

- CHAOS Report
  - Jährlicher Bericht seit 1994 über den Erfolg von IT-Projekten, herausgegeben von der Standish Group
  - Es wurden ca. 100.000 IT-Projekte (vornehmlich in den USA) untersucht
- Kategorisierung der Projekte
  - 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.

Brauchen wir Software Engineering? Ja… (2)



HOCHSCHULE DER WIRTSCHAFT FÜR MANAGEMENT
UNIVERSITY OF APPLIED MANAGEMENT STUDIES

A trend from previous reports that continued in the latest survey is how smaller projects have a much higher likelihood of success than larger ones, as shown in this table.

CHAOS RESOLUTION BY PROJECT SIZE

	SUCCESSFUL	CHALLENGED	FAILED
Grand	2%	7%	17%
Large	6%	17%	24%
Medium	9%	26%	31%
Moderate	21%	32%	17%
Small	62%	16%	11%
TOTAL	**100%**	**100%**	**100%**

The resolution of all software projects by size from FY2011–2015 within the new CHAOS database.

https://www.infoq.com/articles/standish-chaos-2015

With the take up of agile development methods over recent years it was possible to compare project outcomes between agile and traditional waterfall projects. Across all project sizes agile approaches resulted in more successful projects and less outright failures, as shown in this table

CHAOS RESOLUTION BY AGILE VERSUS WATERFALL

SIZE	METHOD	SUCCESSFUL	CHALLENGED	FAILED
All Size Projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Large Size Projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium Size Projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small Size Projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

The resolution of all software projects from FY2011–2015 within the new CHAOS database, segmented by the agile process and waterfall method. The total number of software projects is over 10,000

https://www.infoq.com/articles/standish-chaos-2015

Brauchen wir Software Engineering? Ja… (3)

- Haushalts- und Konsumelektronik
  - Einfachste Geräte wie Kaffeemaschinen, Waschmaschinen und Kühlschränke beinhalten Softwaresysteme
  - Moderne Geräte wie Handys, DVD-Player und Digitalkameras bestehen zum größten Teil aus Software
- Automobilindustrie
  - Betriebliche Abläufe, Verwaltung und Produktion wären ohne Softwaresysteme nicht mehr möglich
  - In einem Fahrzeug sind heute ca. 100 Mikrocontroller integriert
  - Mehr als die Hälfte aller Fahrzeugpannen lassen sich auf Softwareprobleme zurückführen

Brauchen wir Software Engineering? Ja… (4)

- Informationssysteme
  - Anwendungsbranchen: Finanzwesen, Gesundheitswesen, Verwaltung, ...
  - Informationssysteme haben inzwischen einen Durchdringungsgrad in der Geschäftsprozessunterstützung von 60% bis 90%
  - Die Abwicklung eines Geschäftsprozesses erfordert unter Umständen das Zusammenspiel von mehr als 15 (Groß-) Anwendungen

Most of all,
modern day systems are harder
because we built all the easy ones years ago.

Tom DeMarco

Wichtigste Erfolgsfaktoren

Ermittelt durch Befragung von Managern (CHAOS-Studie)

Erfolgreiche Projekte – Typ I

- Beteiligung der zukünftigen Nutzer
- Unterstützung durch das (obere) Management
- Klare Anforderungen
- Angemessene, ordnungsgemäße Planung
- Realistische Erwartungen

„Misserfolgsfaktoren“

Nachgebesserte/abgebrochene Projekte – Typ II/III

- *siehe oben – “anders” herum*
- Technologische Defizite
- Mangelnde Ressourcen

We know why projects fail,
we know how to prevent their failure
- so why do they still fail?

Martin Cobb, 1995

Herausforderungen des Software Engineering‘

1. Heterogenität
   - Verteiltheit, heterogene HW, heterogene OS, heterogene Programmiersprachen, alte vs. neue SW-Systeme
   - Kostengünstige Integration/Weiterentwicklung bei Sicherstellung der Geschäftsprozesse
2. Entwicklungsgeschwindigkeit
   - Hohe Innovationsgeschwindigkeit
   - Schnellere Software-Entwicklung bei gleichbleibender Qualität
3. Vertrauen
   - SW in allen Lebensbereichen (Web, eingebettet)
   - Nachweisbar sichere Systeme

Es gibt eine Grenze,
wie viel *ein* Mensch
zu einer gegebenen Zeit
verstehen kann.

Diskussion

- Okay. Wir wissen nun, dass es schwierig ist, Software für aktuelle relevante Anwendungen zu erstellen.
- Was ist die Lösung?

Arto Teräs
http://ajt.iki.fi/travel/debconf5/page2.html

F R A G E N

Software-Engineering: Historisches (1)

- 1940 – 1950
  Computer werden nur für die Ausführung einzelner Programme benutzt
  - Entwicklung des elektronischen Rechners
  - Keine Betriebssysteme → Programme werden von Lochkarten geladen
  - Typische Anwendung: technisch-wissenschaftliche Berechnungen
- 1950 – 1960
  Computer werden nur für die gleichzeitige Ausführung mehrerer Anwendungsprogramme benutzt
  - Single User Betriebssysteme
  - Höhere Programmiersprachen (u.a. Fortran, Cobol)

Software-Engineering: Historisches (2)

- 1960 – 1970
  Computer werden in großem Umfang in allen möglichen Bereichen eingesetzt
  - Leistungsfähigere und preiswertere Rechner
  - Multi User Betriebssysteme
  - Die Anwendungen werden zahlreicher, komplexer, kritischer
- 1968
  NATO-Konferenz in Garmisch-Partenkirchen
  - Qualitätsprobleme erkannt → Software-Krise
  - Lösung → Software-Engineering

Definition Software

Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

IEEE Standard 620.12 – 1990:
Glossary of Software Engineering Terminology

Definition Engineering

The application of a systematic, disciplined, quantifiable approach to structures, machines, products, systems, or processes.

IEEE Standard 620.12 – 1990:
Glossary of Software Engineering Terminology

Definition Software-Engineering (1)

Software engineering is the establishment and use of sound engineering principles in order to obtain economically software that is reliable and works efficiently on real machines

P. Naur and B. Randall (eds.),
Software Engineering: A Report on a Conference
Sponsored by the NATO Science Committee. NATO 1969

Diskussion

- „Engineering“, „Software“…

  Was bedeutet
  - systematic
  - disciplined
  - quantifiable approach

  in der Software-Entwicklung?

Überlegen Sie sich je ein Beispiel!

HOCHSCHULE DER WIRTSCHAFT FÜR MANAGEMENT
UNIVERSITY OF APPLIED MANAGEMENT STUDIES

Definition Software-Engineering (2)

Eine technische Disziplin, die sich mit allen Aspekten der Softwareherstellung befasst, von den frühen Phasen der Systemspezifikation bis hin zur Wartung des Systems, nachdem sein Betrieb aufgenommen wurde.

[Sommerville 2004]

Technische Disziplin

Suche nach Lösungen (auch, wenn keine theoretischen Grundlagen existieren) innerhalb organisatorischer und finanzieller Beschränkungen

Alle Aspekte der Softwareherstellung

Nicht nur technische Aspekte, sondern auch Projektverwaltung, Entwicklung neuer Theorien, Methoden, Werkzeuge etc.

Definition Software-Engineering (3)

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 Standard 620.12 – 1990:
Glossary of Software Engineering Terminology

F R A G E N