

Softwaretechnik

Einführung



Prof. Dr.-Ing. Norbert Siegmund
Software Systems



UNIVERSITÄT
LEIPZIG

Organisatorisches: Übersicht

- Durchführung:
 - Präsenzlehre: Prof. Dr.-Ing. Norbert Siegmund
 - Montags, 09:15 – 10:45 (HS1)
 - Dienstags, 09:15 – 10:45 (HS1)
 - Übung: M.Sc. Stefan Jahns; B. Sc. Annemarie Wittig
 - Mittwochs A, 11:15 – 12:45 (SG 1-27)
 - Mittwochs B, 13:15 – 14:45 (SG 1-27)
- Kursmaterial (gesamte Kommunikation über Moodle!):
 - Moodle: „Softwaretechnik WS24/26“ <https://moodle2.uni-leipzig.de/course/view.php?id=51684>
 - Folien, Podcasts, Diskussionen, Übungsaufgaben, Umfragen



Organisatorisches: Einordnung

- Einordnung:
 - Vorlesung: Theoretische Grundlagen, kleine Beispiele, Diskussionen, Live-Coding
 - Übung: Praktische Beispiele, Stoff orientiert sich an Erfahrungen und Problemen bei der Bearbeitung eines typischen, realen Softwareprojekts
 - **Keine** Pflichtabgaben in der Übung
- Stoff:
 - Ca. zwei Themen pro Woche bis Mitte Dezember
- Klausur:
 - 75 Minuten, 5 CLP

Vorlesung mit praktischer Relevanz!

Wissens- und Anwendungsvermittlung von Technologien, Prozessen und Konzepten, die direkt in der Praxis vorkommen

Industrievorlesungen zu ausgewählten Themen

Agile und Scrum in der Praxis



André Köhler
Professor für IT-Management
IU International Hochschule

Videovorlesung:
CI/CD + DevOps +
Automatisierung



Jonas Hecht
Solutions Architect
Codecentric



Danny Hucke
AI Lead & SW Engineer
IT Sonix

KI in der
Softwareentwicklung
IT SONIX

Podcast (in Moodle) mit Industrievertretern zu
Themen in der Vorlesung + Aufzeichnungen vergangener
Industrievorlesungen

Ablauf einer Vorlesung

- (Optional)Java Programmierung: Quizz / Programmiertätigkeit
- Vorlesung
- Pause (5min... strikt)
- Vorlesung

Interaktion:

- Unbedingt mitmachen! ☺
- Fragen, wenn was nicht klar ist, undeutlich geschrieben steht, zu leise gesprochen wird, etc.



Lernziele für Heute

Verständnis über das Themengebiet Software Engineering erlangen

Bewusstsein über die Wichtigkeit des Software Engineerings verschaffen

Java-Wissen auffrischen

Erster Ausblick über ChatGPT & Co. und deren Einfluss auf die Softwareentwicklung erhalten

Was ist Software Engineering?



Software Engineering Mythen und falsche Vorstellungen



Alleine
arbeiten



SW-Entwicklung
ist nur coden...

Richtig ist

Teamwork und Kommunikation sind
essentiell für den Erfolg

Verständnis der Bedürfnisse der
Nutzerinnen ist unumgänglich

Abstraktion und Code-Qualität
gehen über Schnelligkeit und Hacks

Anpassbarkeit, Wartbarkeit und
Modularität sind Schlüsselfaktoren
für Erfolg



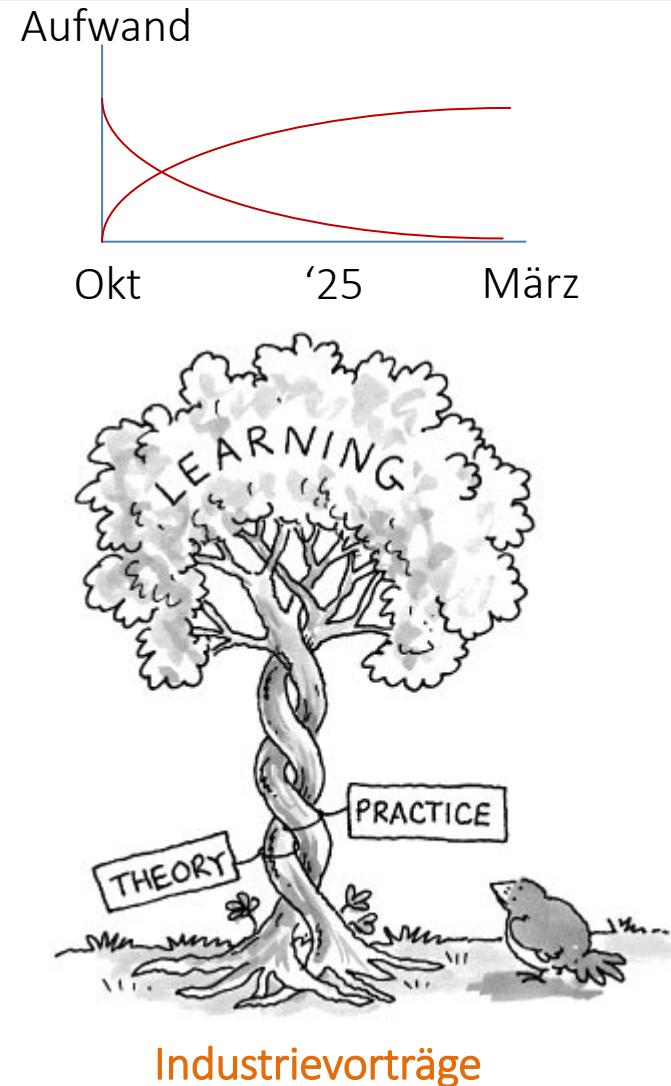
Softwaretechnikprojekt



Softwaretechnik + Praktikum

Softwaretechnik

- Grundlagen der Softwareentwicklung
- Phasen des Softwarelebenszyklus
- Softwareentwicklungsprozess
- Requirements Engineering
- Architektur und Design
- Softwareautomatisierung
- Softwarequalität
- Testing



- Durchführung eines Softwareprojekts
- Agile Softwareentwicklung im Team
- Continuous Integration&Delivery
- (Remote Softwareentwicklung)
- Teamrollen und Softskills
- Entwicklung in Sprints
- [Systementwurf (C4)]

Thema: Digitalisierung der Essensbestellung und -ausgabe für Sozial-Arbeiten-Wohnen Borna gGmbH

- Kontext:

Sozial-Arbeiten-Wohnen Borna gGmbH unterstützt Menschen mit Behinderungen und deren Angehörigen. Ihr Ziel ist es, Menschen mit Behinderungen eine möglichst selbstständige und inklusive Teilhabe am gesellschaftlichen Leben zu ermöglichen. Dabei werden drei Werkstätten für Menschen mit Behinderungen an verschiedenen Standorten betrieben.
- Ziel des Projekts:
 - Digitalisierung des Bestellprozesses des Essens; Kontrolle und vereinfachte Ausgabe des Essens und dadurch eine Verringerung des Verwaltungsaufwandes, sowie Fehler im Zuge dieses Prozesses
- Lernziele:
 - Softwareprojekt mit echten Kunden, echten Anforderungen, echten Menschen durchlaufen
 - Reales Team- und Projektmanagement mit Abgabefristen
 - Echte SoftwareexpertInnen an Ihrer Seite zur Beratung
 - In der Praxis eingesetzte Frameworks und Technologien angewandt einsetzen



Echte Industrieexpertise!

\SSIST
Converneo: D I G I T A L



IT SONIX

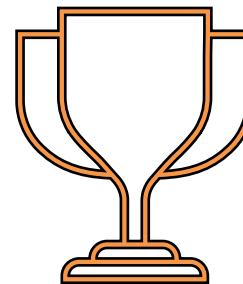
Die Teams werden 3-4x Konsultationen mit echten SW-EntwicklerInnen zum Projekt haben und so ein direktes Feedback aus Industriesicht erhalten.

Auszeichnung für die drei besten Teams



<https://www.optimax-energy.de/>

Erstmals eine Auszeichnung für die 3 besten (vom Kunden) ausgewählten Lösungen.
Jedes Teammitglied der Siegerteams erhält einen Gutschein in Höhe von 50€.



Lernziele: Technologisch

- Kennenlernen von agiler Softwareentwicklung (Arbeiten im Team, direkt mit Kunden, mittels kleinen Sprints)
- Verwendung von Standardwerkzeugen der Softwareentwicklung (Versionsverwaltung, Issue Tracking System, CI/CD Pipeline)
- Erlernen, wie man sich in neue Frameworks und Sprachen einarbeitet
- Client-Server Architektur implementieren
- Kommunikation zwischen Microservices und REST-API in kleinen Rahmen herstellen
- Frontend(Webseiten)-entwicklung mit Serverkommunikation

Organisatorisches

- Industrievorlesungen, Kundengespräche, Präsentationen, Tutorials:
 - Freitag: 13:15 – 14:45 HS 6
 - Tipp: Geeignet auch für Gruppentreffen davor oder danach
 - Diese Woche: Erstes Kundengespräch (unbedingt wahrnehmen, damit Sie wissen, worum es geht)
- Kurs: <https://moodle2.uni-leipzig.de/course/view.php?id=45790>

Warum ist Software Engineering ~~wichtig~~?

unabdingbar

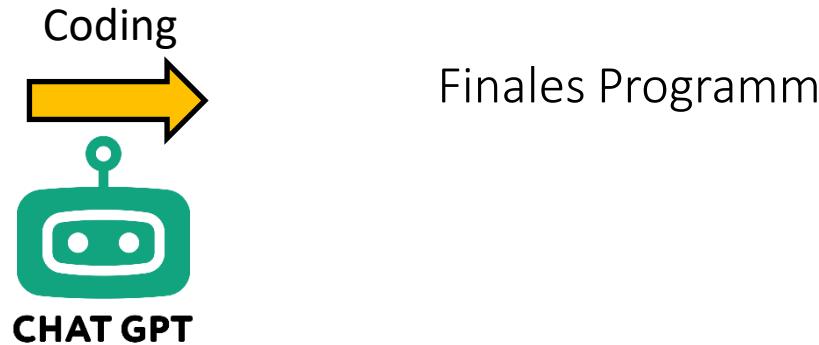
Software Engineering vs. Informatik

- Informatik beschäftigt sich mit der Theorie und den Grundlagen von Computersystemen
- Software Engineering beschäftigt sich mit praktischen Themen der Entwicklung und Auslieferung *guter* Software

Warum Software Engineering?

Naive Sicht:

Problemspezifikation



Aber:

- Woher kommt die **Spezifikation**?
- Wie korrespondiert die Spezifikation zu den **Nutzeranforderungen**?
- Wie entscheidet man, wie das Programm **strukturiert** wird?
- Wie weiß man, dass das Programm **tatsächlich den Spezifikationen entspricht**?
- Wie weiß man, dass das Programm immer **korrekt arbeitet**?
- Was macht man, wenn die Nutzeranforderungen **sich ändern**?
- Wie **teilt man Aufgaben auf**, falls man mehr als ein 1-Personen Team hat?

Warum scheitern Softwareprojekte?



Beispiele gescheiterte Projekte: Ariane 5

- Ariane 5 Flight 501: 4. Juni, 1996
- Selbstzerstörung nach 39 Sekunden
- ~ 500 million US-\$ Schaden
- Grund: Wiederverwendung von Referenzimplementierung der Ariane 4 ohne Anpassung auf neue Raketenspezifikation



Quelle(FU): <https://www.ima.umn.edu/~arnold/disasters/ariane.html>

Therac-25

- Medizinische Strahlentherapie
- Durch SW-Fehler zu hohe Strahlendosis, 3 Tote
- Grund: Ein einziger Entwickler schrieb Software und benutzte vorhandene Komponenten mit wenigen / keinen Tests

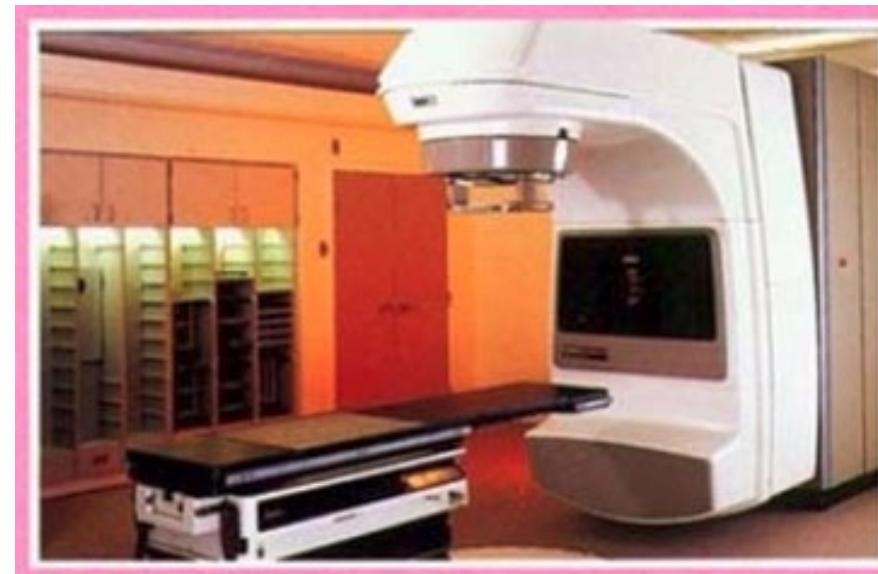


Image source (FU): <http://cr4.globalspec.com/blogentry/19025/Failure-of-the-Therac-25-Medical-Linear-Accelerator>

Apollo 11 PGNCS

- Während der Mondlandung:
 - Unerwartete “executive overflow” Alarms
 - 13% der Computerressourcen wurden durch einen Fehler im Radar verbraucht (5x Alarm + Neustart der Software)
 - Grund: Gleichzeitige Ermittlung von Daten durch das Landeradar und des Rendezvousradars



Image source (PD): https://en.wikipedia.org/wiki/Apollo_11#Lunar_descent

“Morris Worm”



- Erster “Computervirus” – per Zufall
- Entworfen als ein Forschungswerkzeug zum Zählen der Computer im Internet (1989)
- Grund: Durch Bug verbreitete er sich selber
 - 10% des Internetverkehrs
 - Ein PC konnte mehrmals infiziert werden
- 10k-10.000k\$ Schaden



Image source (CC): https://en.wikipedia.org/wiki/Morris_worm#/media/File:Morris_Worm.jpg

Mars Climate Orbiter

- Verloren 1999 beim Anflug auf den Mars
- ~ 330 Millionen-US-\$ Schaden
- Grund: Einheitenfehler im Navi (metrisch vs. US-Skala)

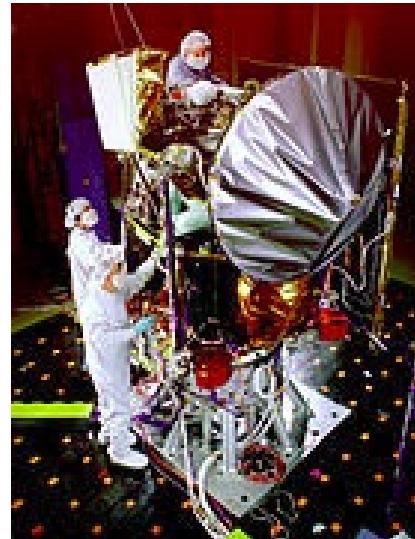


Image source (PD): https://en.wikipedia.org/wiki/Mars_Climate_Orbiter

Mars Polar Lander

- Verloren bei der Marslandung 1999
- ~ 330 Millionen-US-\$ Schaden
- Grund: Fehler in der Software interpretierte Vibrationen bei der Landung als tatsächliche Landung und schaltete Triebwerke zu früh ab



Image source (PD): https://en.wikipedia.org/wiki/Mars_Polar_Lander

Patriot Missile Disaster

- Fehler im Zielsuchsystem führte zum Verfehlen der abzufangenden Rakete und somit zum Tod von 28 Menschen während des Golfkriegs 1991
- Grund: Fehlerhafte Zeitberechnung durch ungenaue arithmetische Berechnungen (Kommastellen wurden nach 24Bit abgeschnitten, was zu Ungenauigkeiten führte)



Image source (CC): https://en.wikipedia.org/wiki/MIM-104_Patriot

Corrupted Blood Incident

- Unbeabsichtigte “Seuche” in WoW, 2005, 1 Woche
- Wurde später bei Epidemiologen verwendet, um die Ausbreitung von Seuchen zu studieren
- Grund: Keine Tests bzw. Konzept für lokal abgeschirmte Bereiche von Effekten



Image source (FU): https://en.wikipedia.org/wiki/File:WoW_Corrupted_Blood_Plague.jpg

Hamburg-Altona Railway Switch

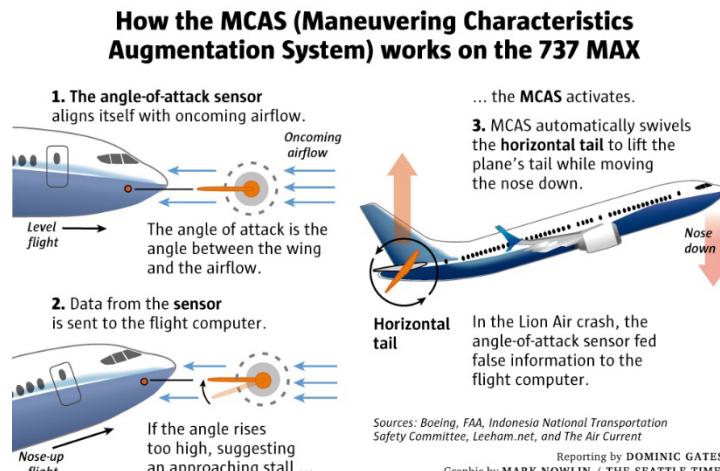
- Computer “upgrade” des Switch-Systems
- Ergebnis: Gesamter Bahnhof musste für 2 Tage schließen,
100 000s Passagiere betroffen
- Grund: Stackoverflow + Fehler im Code zum Behandeln des Overflows



Image source (CC): https://en.wikipedia.org/.../media/File:Bahnhof_Hamburg_Altona.JPG

Boing 737 Max

- <https://www.seattletimes.com/business/boeing-aerospace/failed-certification-faa-missed-safety-issues-in-the-737-max-system-implicated-in-the-lion-air-crash/>
- Softwaresystem MCAS (Maneuvering Characteristics Augmentation System) hatte Sicherheitsprobleme und war abhängig von einem einzelnen Sensor
- 346 Menschen starben; >32\$ Mrd. Aktienverlust, >8\$ Mrd. Kosten



Facebook Outage

POSTED ON OCTOBER 4, 2021 TO NETWORKING & TRAFFIC

Update about the October 4th outage



By Santosh Janardhan

To all the people and businesses around the world who depend on us, we are sorry for the inconvenience caused by today's outage across our platforms. We've been working as hard as we can to restore access, and our systems are now back up and running. The underlying cause of this outage also impacted many of the internal tools and systems we use in our day-to-day operations, complicating our attempts to quickly diagnose and resolve the problem.

Our engineering teams have learned that configuration changes on the backbone routers that coordinate network traffic between our data centers caused issues that interrupted this communication. This disruption to network traffic had a cascading effect on the way our data centers communicate, bringing our services to a halt.

- Facebooks Border Gateway Protocol Server erhielten ein Update mit einem Konfigurationsfehler, welcher dafür sorgte, dass alle Verbindungen von allen Datencenter von Facebook unterbrochen wurden.
- Dies führte zu weiteren Effekten, welche praktisch alle Facebook Dienste vom Rest des Internets abtrennten

So what happened to Facebook?

It turns out that BGP played a part in Facebook's issues but wasn't the root cause. In its [detailed explanation, released on Tuesday](#), the company says that a command issued as part of routine maintenance accidentally disconnected all of Facebook's data centers (oops!). When the company's DNS servers saw that the network backbone was no longer talking to the internet, they stopped sending out BGP advertisements because it was clear that something had gone wrong.

To the wider internet, this looked like Facebook telling everyone to take its servers off their maps. [Cloudflare's CTO reported](#) that the service saw a ton of BGP updates from Facebook (most of which were route withdrawals or erasing lines on the map leading to Facebook) right before it went dark. One of Fastly's tech leads tweeted that [Facebook stopped providing routes to Fastly](#) when it went offline, and [KrebsOnSecurity backed up the idea](#) that it was some update to Facebook's BGP that knocked out its services.

Crowdstrike Bug 2024

- Ein fehlerhaftes Update der *Sicherheitsfirma* Crowdstrike sorgte dafür, dass 8,5 Millionen Windowsrechner in einer Endlosschleife von BlueScreen und Reboot unbrauchbar waren
- Chaos in Reisebranche und Gesundheitssystem; 1,5 Mrd\$ Cyberversicherungsschaden; 5,4 Mrd\$ Verluste für Fortune-500-Unternehmen
- Update von Antivirus-Signatur wurde **ungetestet** auf der ganzen Welt ausgerollt



Warum scheitern Softwareprojekte?

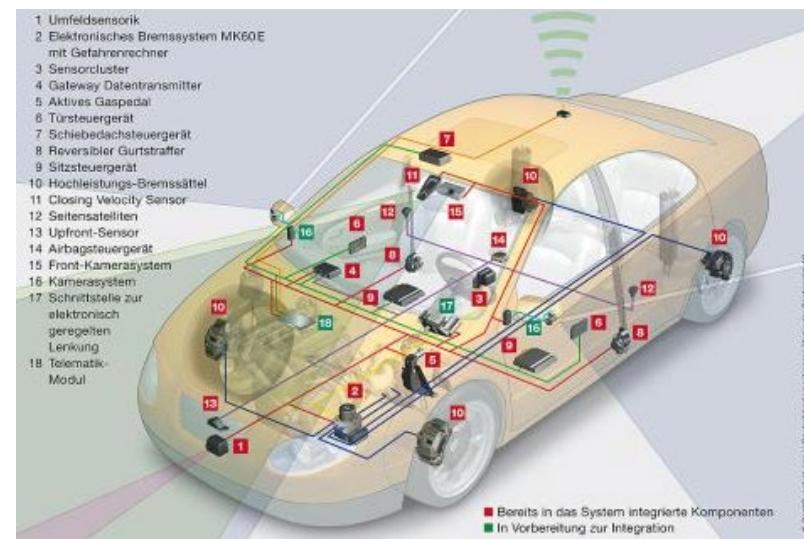
Budget zu klein oder aufgebraucht
Zeit aufgebraucht
Ineffizienz von Software
Schlechte Qualität von Software
Software erfüllt Anforderungen nicht
Projekte waren nicht mehr zu managen
Quelltext schwierig zu warten
Software wurde nie ausgeliefert



Warum scheitern SW-Projekte nicht?

Technologie unklar
Fehlende Programmierfähigkeiten
Kein Code produziert
10X ProgrammiererInnen fehlen
UML-Diagramm ist falsch

Lösung: Software Engineering



Begriff: Software Engineering

- Begriff wurde auf der NATO-Konferenz (1968) geprägt
- Idee: Softwareentwicklung in Anlehnung an Ingenieursdisziplin (Engineering)
- Im Folgenden: Definitionen
 - Geben einen Einblick über verschiedene Sichtweisen auf diese Disziplin

Definition: Software Engineering

„state of the art of developing quality software on time and within budget“

Aktuellster Stand der Technik entscheidet Community und erfordert Lebenslanges lernen
Ressourcenbeschränkung

„multi-person construction of multi-version software“ -- Parnas

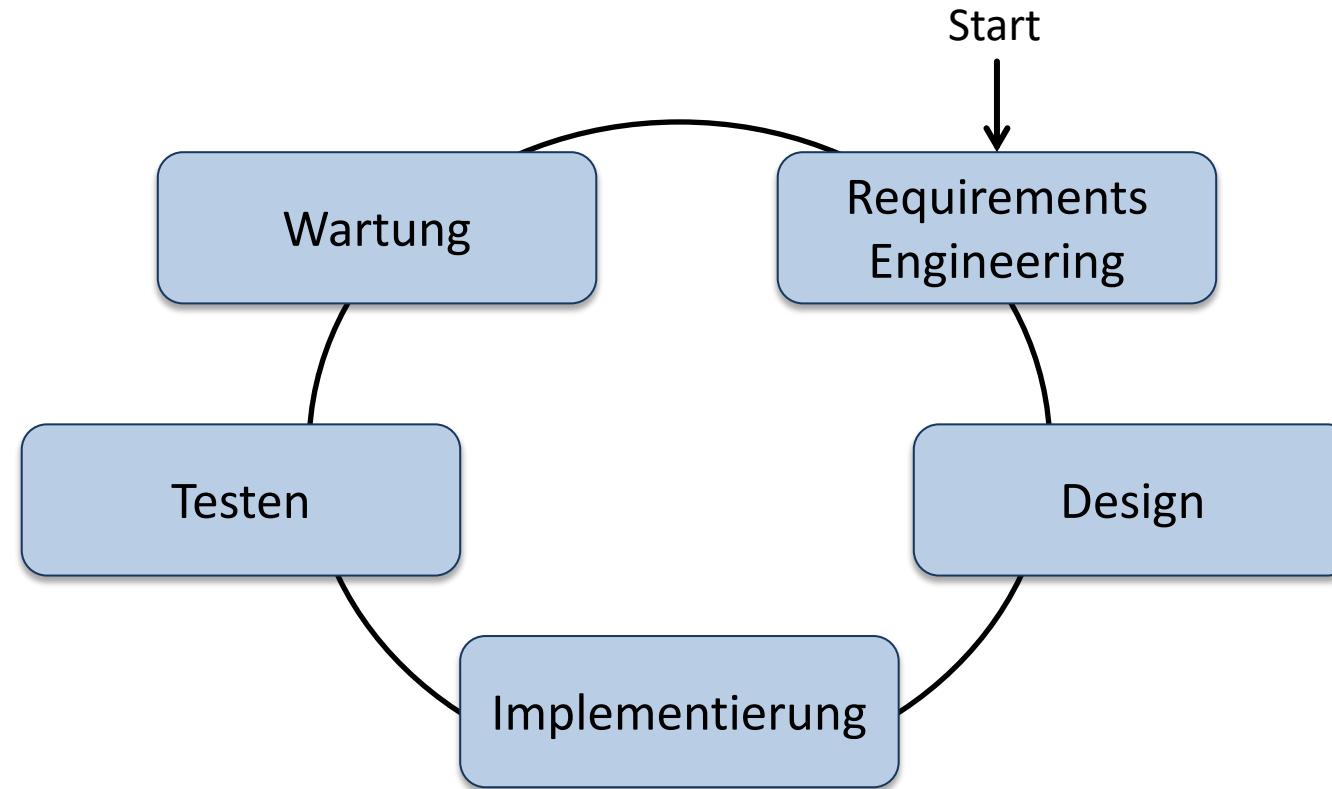
Teamwork

Erfolgreiche Software-Systeme müssen sich weiterentwickeln: Änderung ist der Normalfall, nicht die Ausnahme

„software engineering is an engineering discipline that is concerned with all aspects of software production“ -- Sommerville

Software ist ein Produkt, das für einen bestimmtem Kunden entwickelt wird
Nicht nur Programmierung, sondern alle Aspekte des Softwarelebenszyklus

Softwarelebenszyklus



Programmierung ist nur ein kleiner Teil von Software-Entwicklung

Large Language Models (LLMs): ChatGPT & Co.

Was sind Ihre Gedanken hierzu?

„Gute“ Software

- Maintainable (Wartbar)
- Dependable / Reliable (Verfügbar, Zuverlässig)
- Efficient (Effizient)
- Usable (Nutzbar, Anwendbar)

Die Grenzen von ChatGPT & Co.: Fundamentale Prinzipien und Konzepte der Softwareentwicklung

Prinzipien

Divide and conquer
Simplicity
Rigorousness

Konzepte

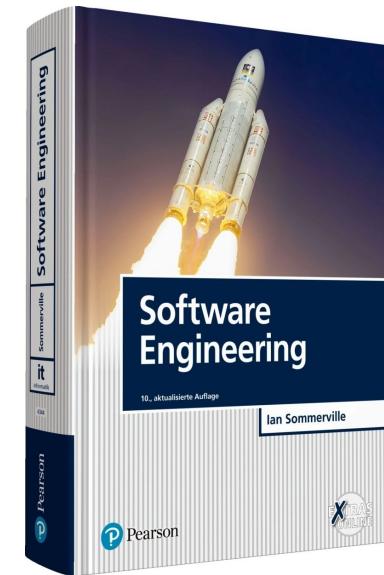
Modularität und Struktur
Abstraktion und Generalisierung
Design for change
Separation of concerns
Stepwise refinement
Information hiding

Zusammenfassung

- Grundlegende Vorstellung von Software Engineering haben
- Notwendigkeit des Software Engineerings erkennen
- Einordnung des Anteils der Programmierung an der Entwicklung von Software

Literatur

- *Software Engineering*. Ian Sommerville. Addison-Wesley Pub Co; ISBN: 020139815X, 10th edition, 2017
- *Software Engineering: A Practitioner's Approach*. Roger S. Pressman. McGraw Hill Text; ISBN: 0072496681; 5th edition, 2001
- *Using UML: Software Engineering with Objects and Components*. Perdita Stevens and Rob J. Pooley. Addison-Wesley Pub Co; ISBN: 0201648601; 1st edition, 1999
- *Designing Object-Oriented Software*. Rebecca Wirfs-Brock and Brian Wilkerson and Lauren Wiener. Prentice Hall PTR; ISBN: 0136298257; 1990
- Aber:
 - Kein einzelnes Buch für den gesamten Stoff der Vorlesung
 - Zu jedem Thema gibt es Empfehlungen
 - Vieles auch im Internet gut nachzulesen



Was Sie mitgenommen haben sollten

- Was ist Software Engineering?
- Nennen/Erklären Sie X mögliche Gründe für das Scheitern von Software-Projekten
- Was ist die Softwarekrise? Warum trat sie auf?
- Nennen/Erklären Sie die Prozesse im Software-Lebenszyklus.
- Was sind die Gemeinsamkeiten/Unterschiede von Software Engineering zur klassischen Ingenieursdiziplin?