INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

B.Sc. Wirtschaftsinformatik

FALLSTUDIE SOFTWARE ENGINEERING

Sommersemester 2024

DISCLAIMER

Dieses Skript/ dieser Foliensatz/ diese Präsentation darf ausschließlich im Rahmen der von Prof. Dr. Ulrich John durchgeführten Kurse verwendet werden. Eine anderweitige Verwendung oder Veröffentlichung – auch in Auszügen – ist ohne explizite Genehmigung durch Prof. John **nicht gestattet**.
Das vorliegende Lehrmaterial enthält auch Zitate.

WARMING UP

KENNENLERNEN, STAND, ZIELE, PLAN

WER ? *(WEITERES: WWW.IT-UNTERNEHMER.DE)*

Prof. Dr.-Ing. Dipl.-Inf. Ulrich John

Studium: **Informatik** und Mathematik (-1995; Humboldt Universität Berlin)
ABWL (TU Berlin)

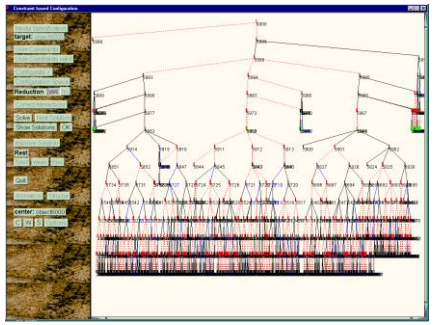
Promotion: Fachgebiete *Softwaretechnik* und *Künstliche Intelligenz*, TU Berlin (2002)

- *92 – 01:* GMD FIRST (German National Research Center for Information Technology/ Institut für Computer Architektur und Softwaretechnik -> FhG FIRST, jetzt FhG FOKUS)
  Bereich „Planungstechnik und Dekalarative Programmierung, *ab 5/95 wissenschaftlicher Mitarbeiter*
- *01 – 07:* DaimlerChrysler Forschung (*Wissensbasierte Entwicklung und Produktion/ Produktions- und Logistikoptimierung*)
  Lab „IT for Engineering and Processes“, *Senior Researcher & IT-Projektleiter*
- *seit 2007:* SIR Plan GmbH (-12/10) / SIR Dr. John UG
  Solutions for Information Management, Resource Optimization, Process Management & Interim Management
  Intelligente Digitalisierung
  Gründer und geschäftsführender Gesellschafter
- 09 – 12: HTW Berlin (Lehrbeauftragter für Informatik in Informatik-Masterstudiengängen)
- *SoSe 11:* Hochschule Lausitz (Lehrbeauftragter für Wirtschaftsinformatik)
- *10/13 – 9/21:* VICTORIA | Internationale Hochschule (HWTK)
  Professor für Wirtschaftsinformatik, Studiengangleiter „B.Sc. Informatik und Management“
- *seit 10/21:* *IU Internationale Hochschule*
  Professor für Informatik

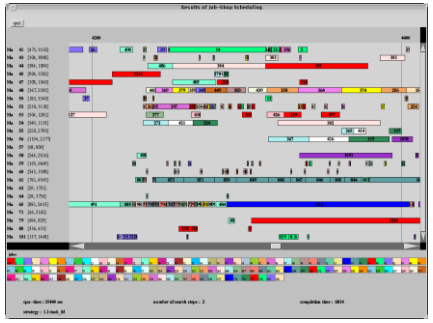
Kontakt: ulrich.john@iu.org

SOFTWAREENWTICKLUNG UND BERATUNG FÜR PROZESSMANAGEMENT UND PLANUNG

– AUSWAHL AUS EXPERTISE -

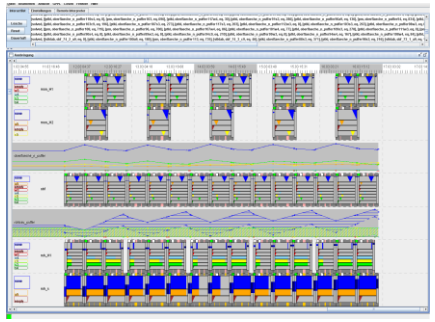


Anlagenprojektierung/
Produktkonfiguration

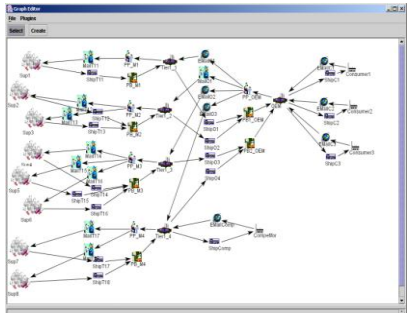


ERP/ Produktionsplanung

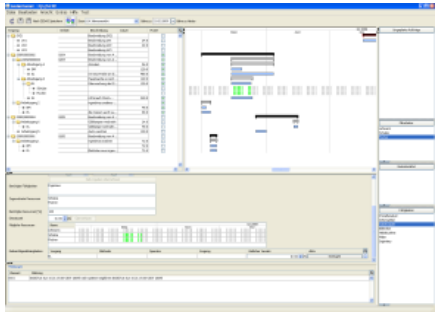
gewerke-orientierte Planung
(Standard der Daimler AG)



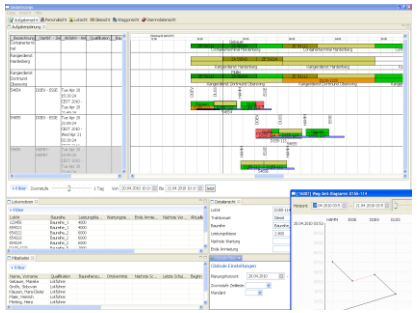
Multi-Purpose-Planning
for Automotive (AutoMPP)



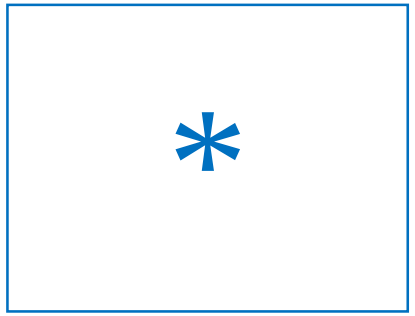
Simulation von Supply Chains,
kollaborative Planung in Netzen



Ressourcenmanagement
im MRO-Sektor



Multiressourcen- Planung &
Disposition im Cargo-Bereich



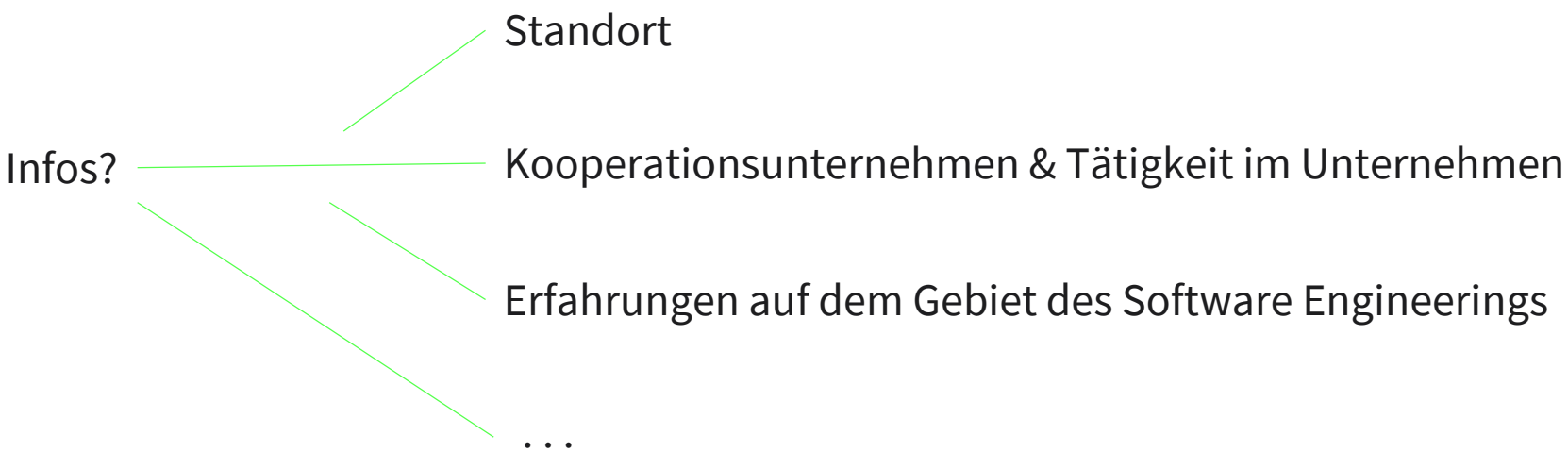
Planungsprozesse in
Aviation und Automotive

Automotive | Luftfahrt | Eisenbahn | Energie & Anlagenbau | Kommunikationslogistik

Logistik/ Supply Chain | Produktion | Management | Entwicklung | E-commerce

Marktforschung | Intelligente Digitalisierung

SIE SIND KURZ AN DER REIHE! WER SIND SIE?



AUSGANGSLAGE, ZIELE, WÜNSCHE UND ERWARTUNGEN, MÖGLICHKEITEN

Erwartungen, Ziele, Wünsche
(generell & fachlich)

?

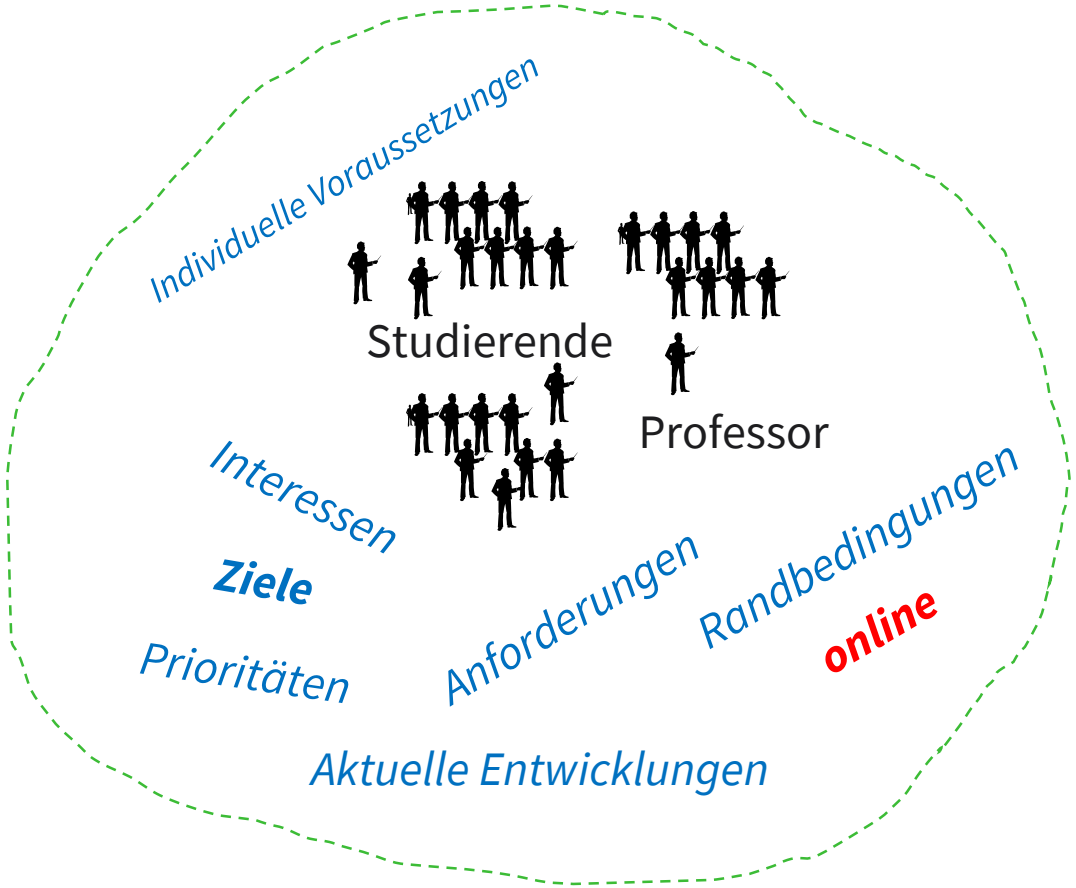
RAHMENVORGABEN

Präsenz: 50 UE

Zeitaufwand Studierende: 150 h

Prüfungsleistung: Fallstudie

UNSER KURS ALS SYSTEM – ZIELE DES KURSES



Wissen & Fähigkeiten (fachlich & persönlich), Motivation, Begeisterung

Kenntnisse & Fähigkeiten (Software Engineering)

Studierende **&** Professor:

ulrich.john@iu.org

MODULBESCHREIBUNG *(ZUR ORIENTIERUNG)*

02

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

KLASSISCHES SOFTWARE ENGINEERING

- KURZE WIEDERHOLUNG/ KURZER ÜBERBLICK -

LITERATUR

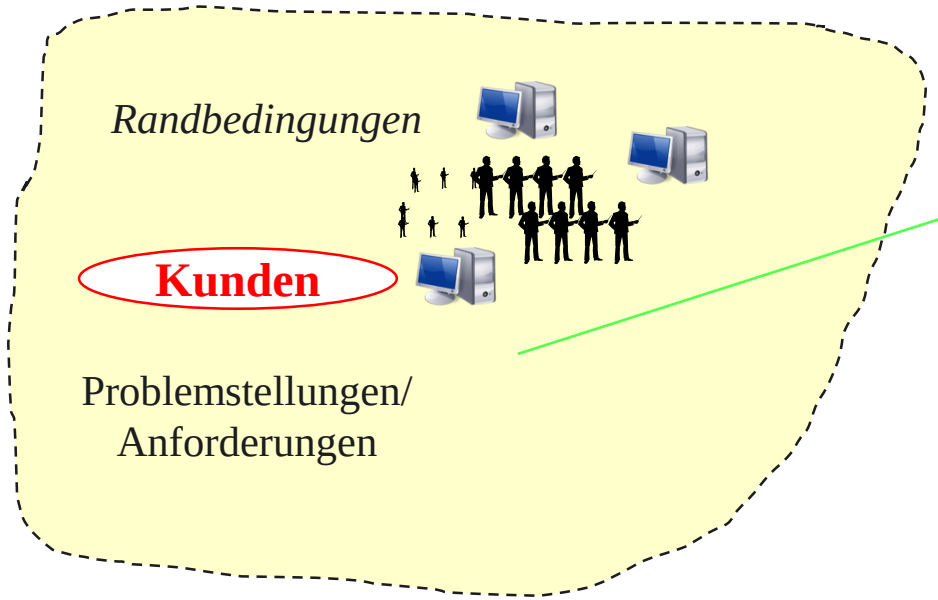
Anja Metzner: Software Engineering kompakt, Hanser, 2020. **[Met20]**

Ian Sommerville: Software Engineering, 10. akt. Auflage, Pearson, 2018. **[Som18]**

Ian Sommerville: Modernes Software-Engineering: Entwurf und Entwicklung von Softwareprodukten, Pearson Studium - IT, 2020 [**Som20**]

Gerd Beneken, Felix Hummel, Martin Kucich: Grundkurs agiles Software-Engineering, Springer Vieweg, 2022 **[BHK22]**

AKTUELLE PARADIGMEN DER SOFTWAREENTWICKLUNG – *WOFÜR?*



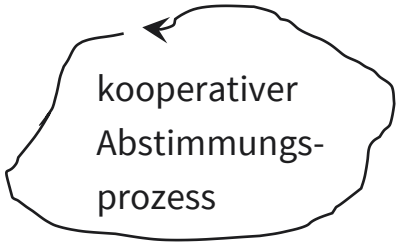
Software, die *Anforderungen erfüllt*, & *Randbedingungen* (z.B. Kostenlimit, Termine, Kosten/ Nutzen) *eingehalten*.

- Qualität,
- Effizienz,
- Benutzerfreundlichkeit,
- Wartbarkeit/ Adaptierbarkeit

Weitere Ziele: z.B. Folgeaufträge, Werbung etc.

DER SOFTWARE-LEBENSZYKLUS

1. **Planungsphase**
2. **Definitionsphase**
   - Anforderungen
   - Spezifikation



was geht?
was ist sinnvoll?

3. **Designphase** - *Modell, Entwurf, Systemdesign*
4. **Implementierungsphase** - SW-Entwicklung
5. **Abnahme- und Einführungsphase** - Testen (Verifikation, Validation, Rollout)
6. **Wartungsphase**

EINIGE GRÜNDE FÜR SOFTWARE ENGINEERING (SE)

- Softwareprojekte scheitern teilweise, liefern oft unzufriedenstellende Ergebnisse
- SE-Projekte laufen oft „suboptimal“ (teuer, zu lang)
- bereits kleine SE-Projekte sind oft schwer beherrschbar

SOFTWARE ENGINEERING (SE) - HISTORIE

- 1968: NATO-Tagung, die die *Softwarekrise* thematisiert

Antwort: **Software Engineering**

Standish Group CHAOS-Studie 2011 – 2015 (> 10.000 Softwareprojekte)

Agiles Modell	Wasserfall-Modell
9 % gescheitert	29 % gescheitert
39 % erfolgreich	11 % erfolgreich

Nach Meinung vieler Experten gilt Agiles Vorgehen derzeit als modernstes Vorgehensmodell.
Agilität kostet, kann aber den Grad des Erfolges und die Wahrscheinlichkeit,
dass die Anforderungen des Endanwenders erfüllt werden, deutlich erhöhen.

Merke: Weniger als 50% aller Softwareprojekte gelten als erfolgreich

Software Engineering ist erforderlich

SOFTWARE ENGINEERING (SE) – HISTORIE II

50er Jahre: Makroverarbeitung, Prozeduren niederer Programmiersprachen

60er Jahre: Entwicklung höherer Programmiersprachen

Compiler

Kosten von Software >> Kosten der Hardware (Softwarekrise)

- erste SE-Methoden kamen auf, z.B. Wasserfallmodell (70er Jahre)
- Algorithmen und Datenstrukturen wurden immer komplexer
- Standard = strukturierte prozedurale Programmierung

strukturierte SE-Methoden wurden benötigt, z.B. das V-Modell und das Spiralmodell (1986)

80er Jahre: OOP, Erfindung von grafischen Notationen, wie die der UML

heute zusätzlich: agile Methoden, wie SCRUM, Extreme Programming, V-Modell XT

Wissen aus zahlreichen SE-Projekten

Eigenschaften von Software

- Immatriell
- Software verschleißt nicht, aber Software-Aging

Schwierigkeiten bei der Software-Erstellung

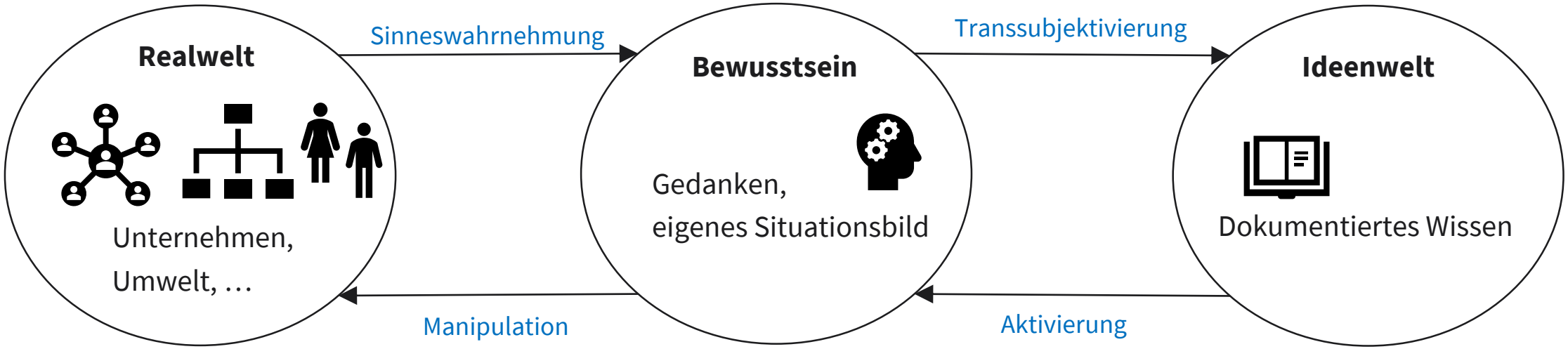
- Softwareentwicklung ist i.d.R. komplex

  → zahlreiche Schwierigkeiten
- SE-Spezialisten, Entwickler, Anwender (unterschiedliches Wissen, unklare Vorstellungen)
- *Akzeptanz- & Integrationsprobleme*
- *Konfiguration & Versionierung* schafft *zusätzliche Komplexität* der Software

Abhilfe: Standards, Methoden, Werkzeuge

Software-Entwicklung ist eine *„ingenieurmäßige“ Wissenschaft* mit *wohldefinierter Vorgehensweise*.

SIR Raimund Popper: **3 Welten**



Das Bewußtsein über den 3-Welten-Kreislauf von Popper erlaubt die Erstellung von besseren SE-Projekten.

BEDEUTUNG VON MODELLEN FÜR DAS SOFTWARE ENGINEERING

- Unterscheidung zwischen Realität und Modellen

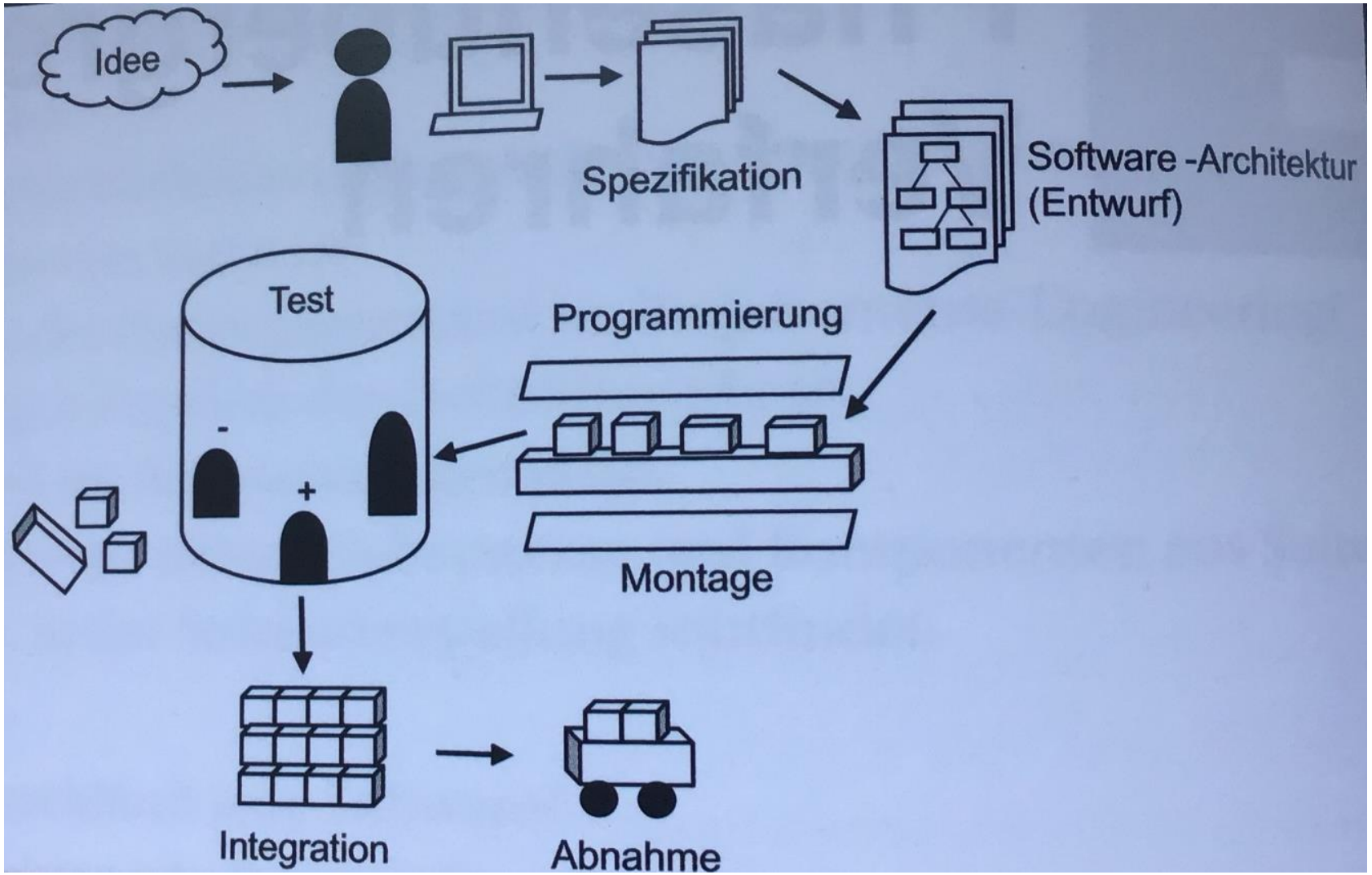
  Reales Objekt & Modell-Objekt

- Modelle sind für Verständnis notwendig

  Vereinfachung & / Abstraktion

- je ähnlicher ein Modell einem realen Objekt ist, desto besser ist das Modell
- Kosten/ Zeit – Nutzen – Verhältnis !

SOFTWARE-FACTORY



Software-Factory als Fließbandfabrik [Met20] aus (Wieken 1990, Software-Produktion)

VORGEHENSMODELLE DER SOFTWAREENTWICKLUNG

VORGEHENSMODELL

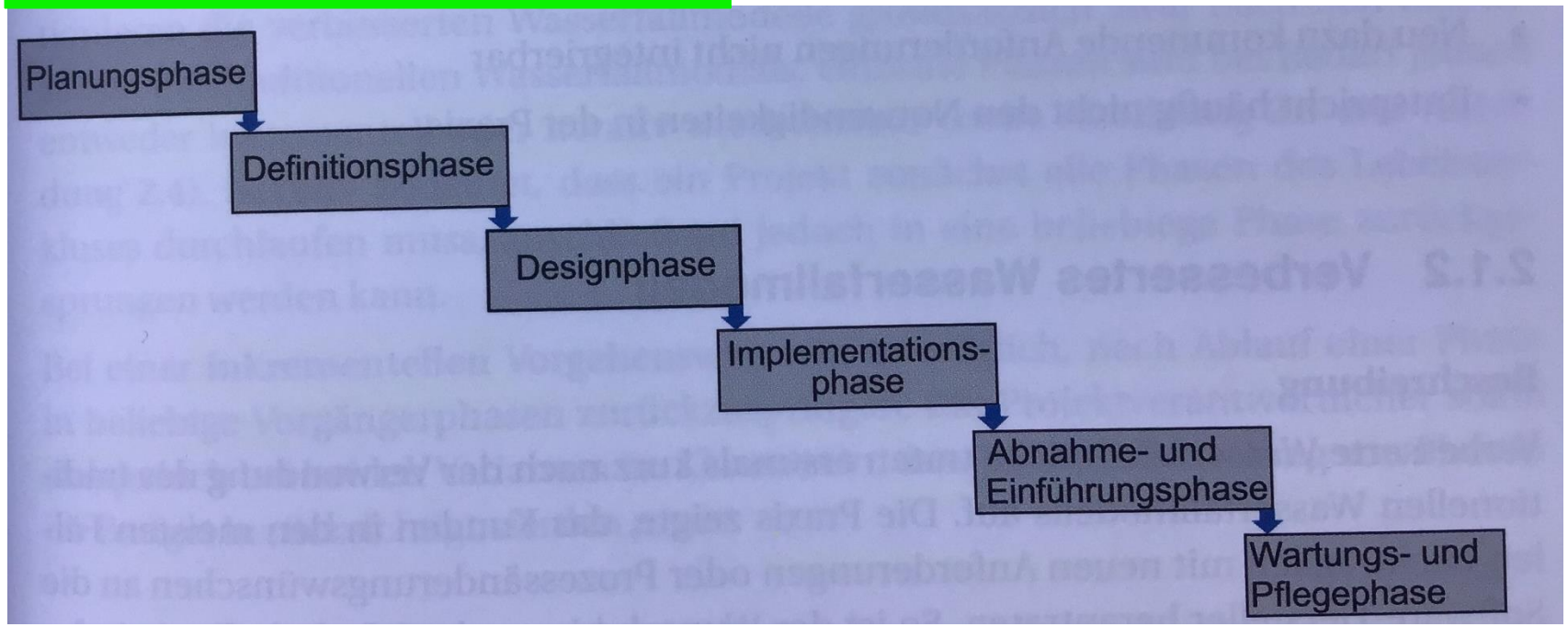
- abstrakte Darstellung für das Vorgehen während der Prozesse des SE
- stellt Lebenszyklen der Software dar
- legt Aktivitäten fest
- Reihenfolge der Aktivitäten
- Zusammenfassung von Aktivitäten zu Phasen
    - jede Phase
        - Phasenziele
        - Aktivitäten & Rollenzuordnungen
        - Dokumentation
        - Methoden, Konventionen/ Richtlinien, Werkzeuge & Sprache
        - vorgesehene/ erlaubte Phasenübergänge

VORTEILE VON VORGEHENSMODELLEN

- Leitfaden für die Systementwicklung
- gemeinsame, verbindliche Sicht der logischen und zeitlichen Struktur eines Projektes
- Anleitung für Dokumentation
- Verbesserte Planbarkeit
- Möglichkeit zur Zertifizierung
- höherer Grad der Personalunabhängigkeit
- frühzeitige Fehlererkennung durch festgeschriebene Testaktivitäten

Alle Vorgehensmodelle beinhalten immer alle Lebenszyklusphasen.

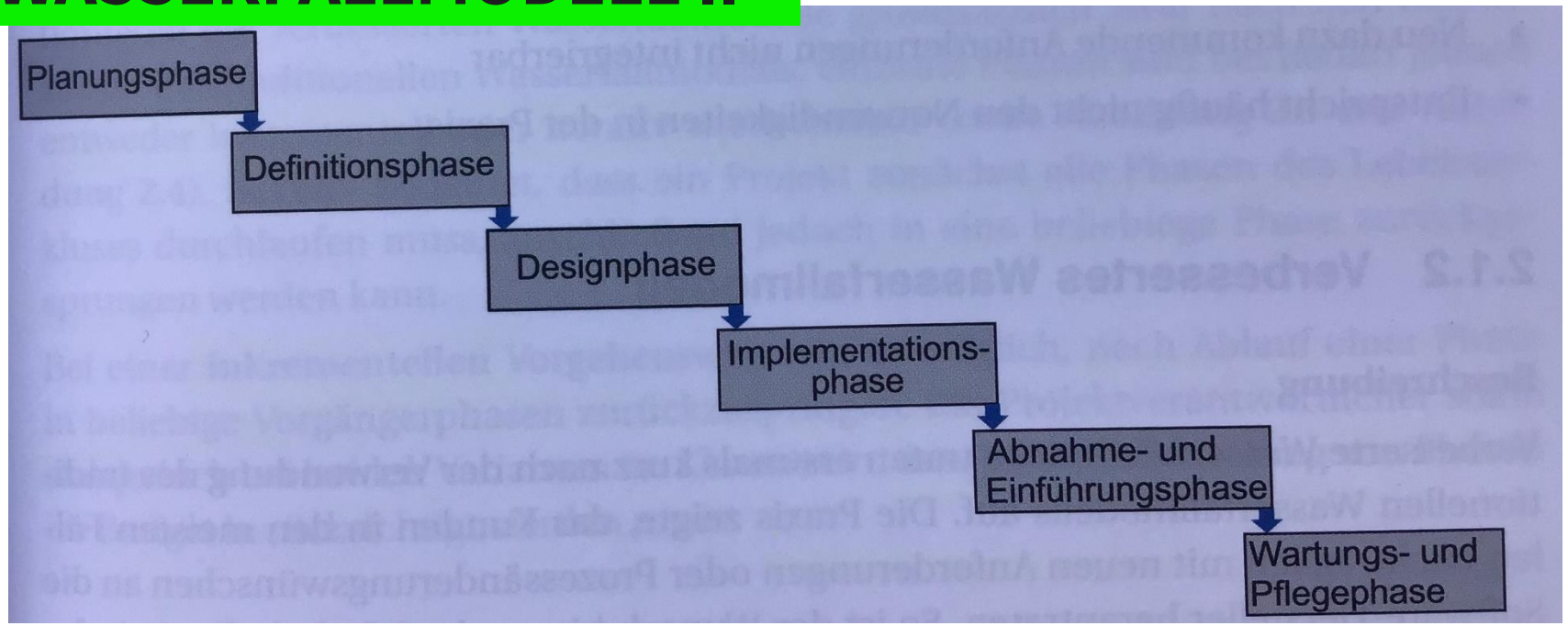
WASSERFALLMODELL



Wasserfallmodell nach Royce (1970) [Met20]

- älteste und bekannteste Vorgehensmodell
- Phasen werden strikt nacheinander abgearbeitet
- noch heute vielfältig im Einsatz
- in jeder Phase entstehen Dokumente, die begutachtet werden (z.B. durch Review)
- Projektteilnehmer haben in der Regel Rollen (PL, SE-Expertin, Entwickler)
- In der Praxis
    - eher bei kleinen Projekten mit wenigen P.-Teilnehmern
    - nur geeignet, wenn wenig Anforderungsänderungen zu erwarten sind
    - Bei klaren Projektgrenzen und fixen Budgets

WASSERFALLMODELL II

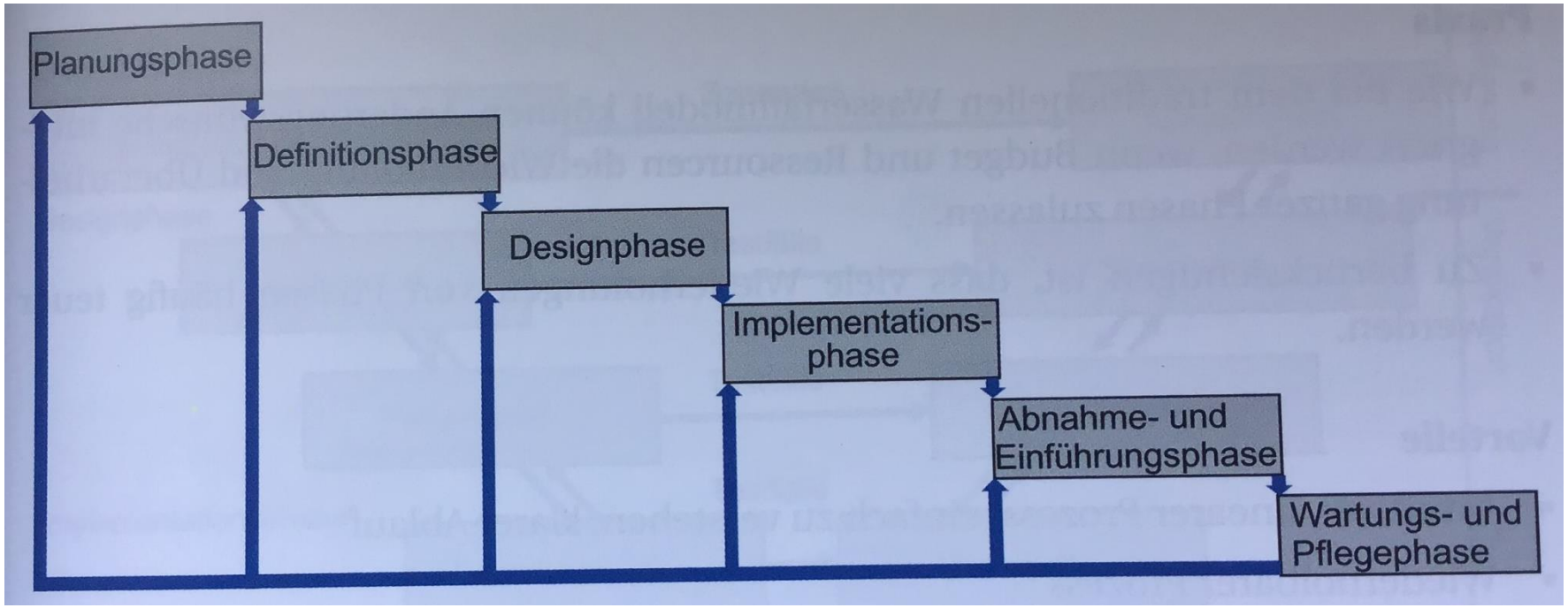


Wasserfallmodell nach Royce (1970) [Met20]

Vorteile:
- Intuitiver, einfacher, leicht zu verstehender Prozess, klarer Ablauf
- Nicht unterbrechbarer Prozess
- Top-down-Vorgang
- Gute Planbarkeit

Nachteile:
- starre Aufteilung in Phasen
- Keine Rückkopplungen zu früheren Phasen
- neue Anforderungen nicht integrierbar
- Entspricht häufig nicht den Notwendigkeiten der Praxis

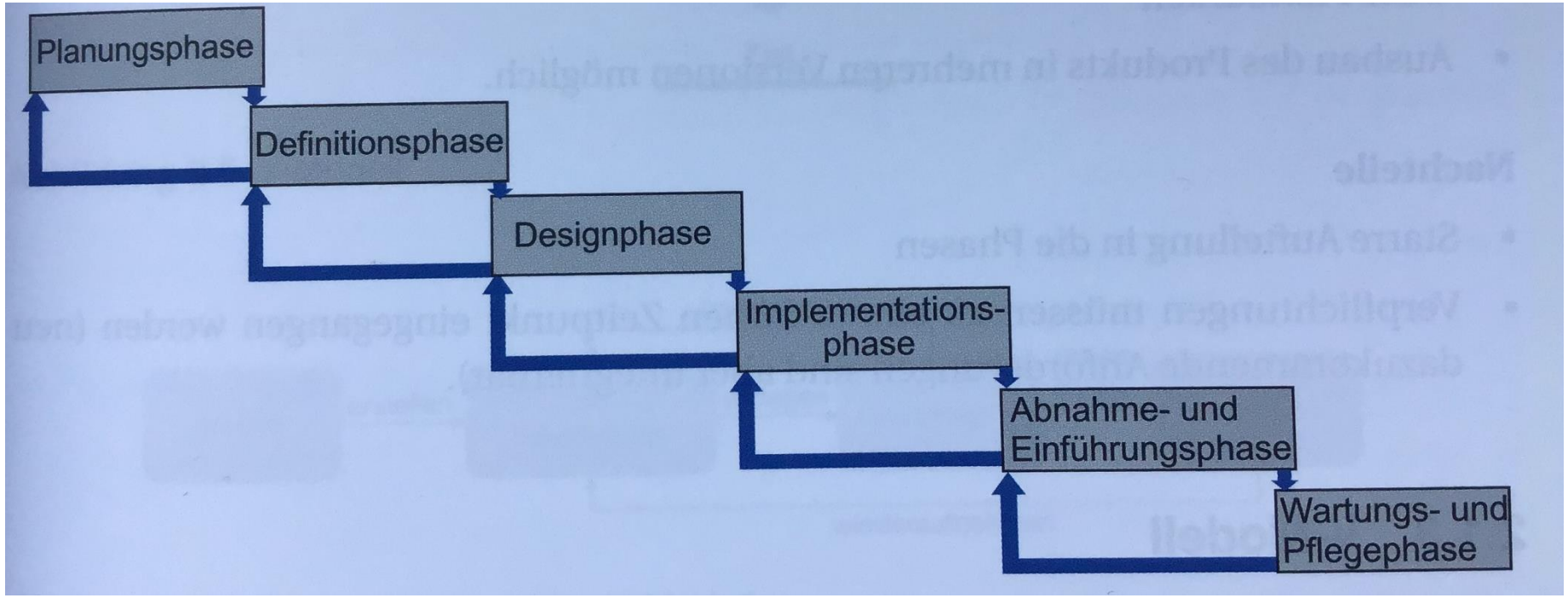
VERBESSERTES WASSERFALLMODELL



Verbessertes Wasserfallmodell – iterativ [Met20]

Iterativ – alle Phasen werden durchlaufen und dann kann zu einer beliebigen Phase des Projektes zurückgesprungen werden

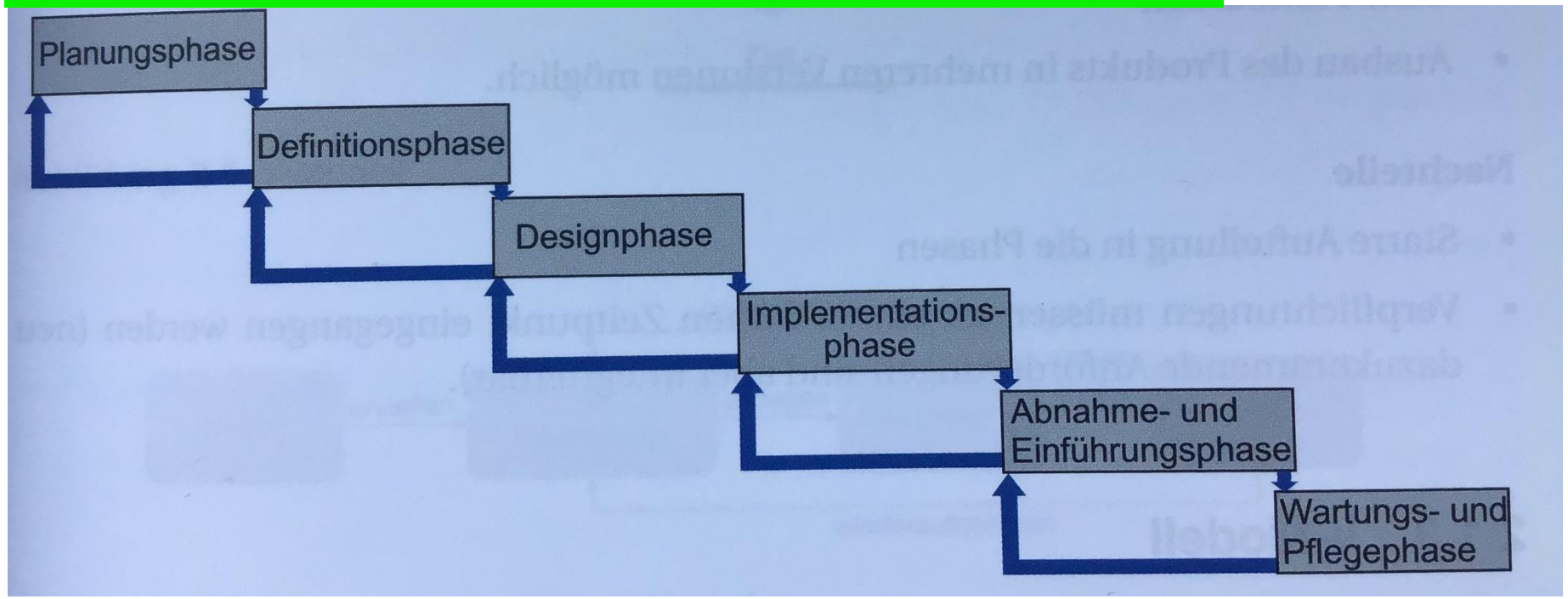
VERBESSERTES WASSERFALLMODELL II



Verbessertes Wasserfallmodell – inkrementell [Met20]

Inkrementell – nach Ablauf einer Phase kann zu einer beliebigen Vorgängerphase zurückgesprungen werden

VERBESSERTES WASSERFALLMODELL III



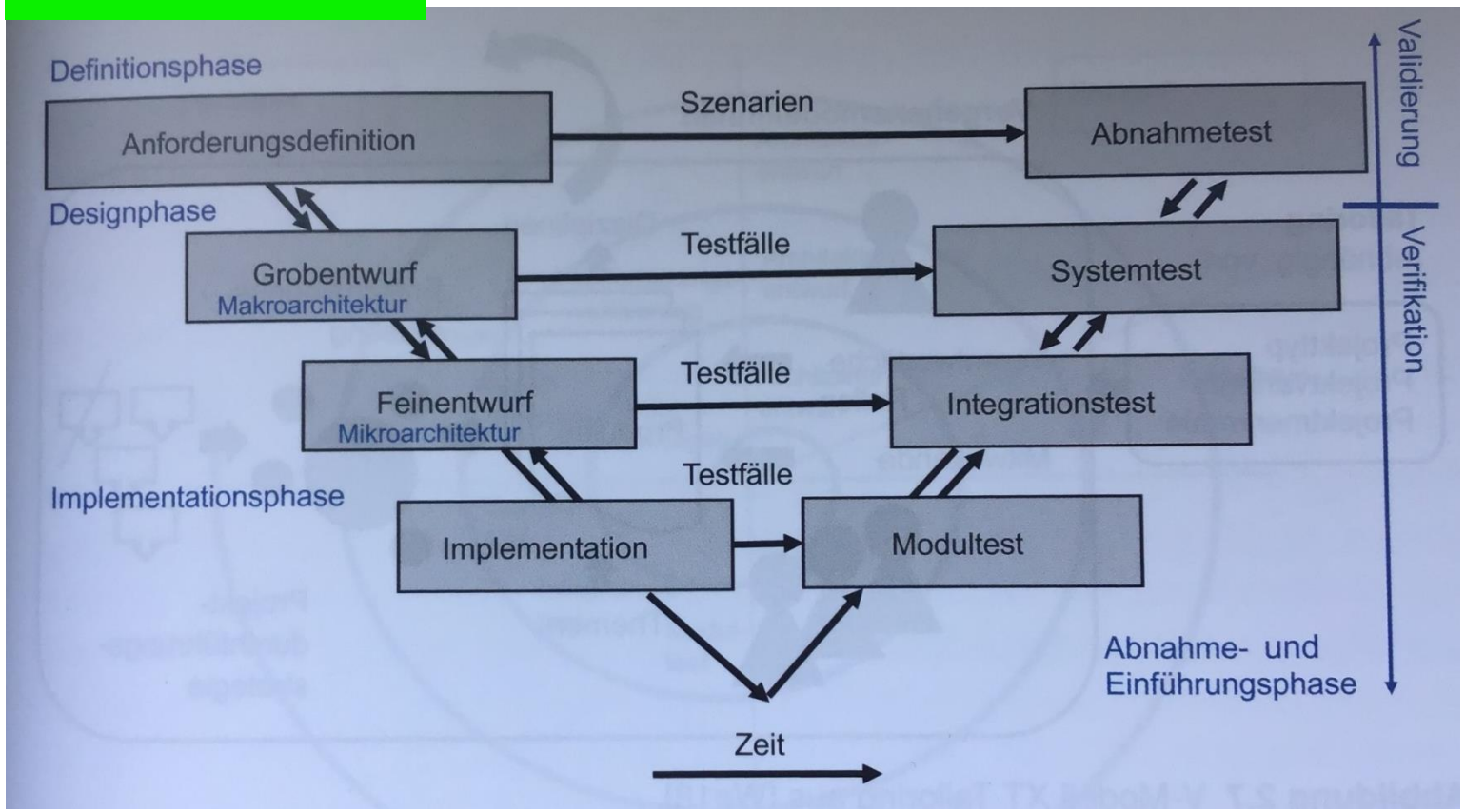
Verbessertes Wasserfallmodell – inkrementell [Met20]

Praxis:
- Änderungswünsche können integriert, wenn Budget vorhanden und Zeit ausreichend
- Wiederholungen von Phasen kostet!

Vorteile
- intuitiv, linearer Prozess
- Wiederholbarkeit
- Top-down- Vorgang
- gute Planbarkeit
- Produktentwicklung mit mehreren Versionen möglich

Nachteile:
- Starre Aufteilung in Phasen
- Frühzeitige Verpflichtungen

V-MODELL



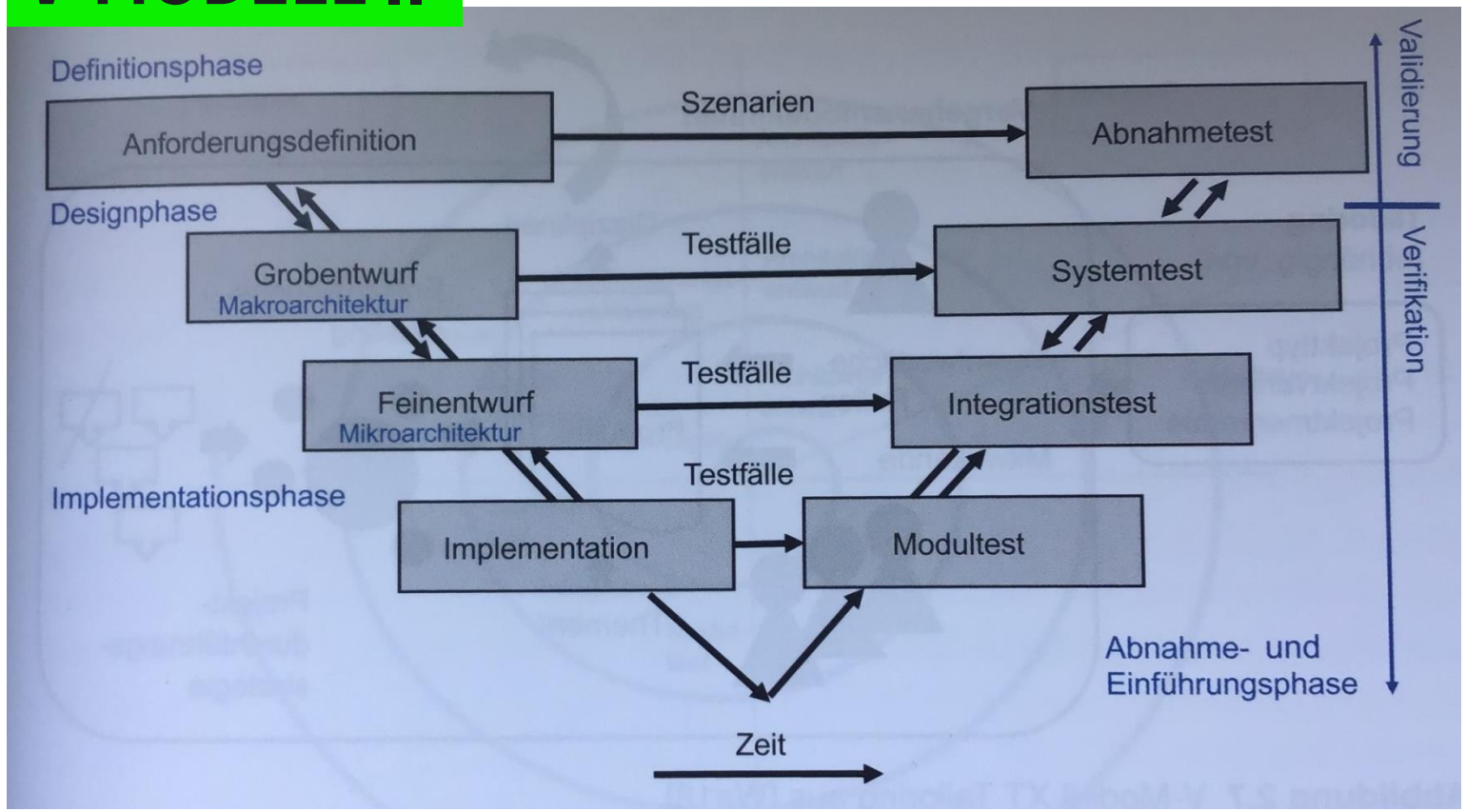
V- Modell [Met20]

- Häufig in administrativer Umgebung und in großen Unternehmen, heute eher: V-Modell 97 oder V-Modell XT, da diese agile Vorgehensweisen ermöglichen
- Dokumente werden im V-Modell ***Produkte*** genannt, jedes Produkt durchläuft definierte Zustände, als ***Aktivitäten*** werden Tätigkeiten bezeichnet, die Produkte ändern

Tailoring

- Merkmal der modernen Formen des V-Modells = Eigenschaft, dass das Vorgehensmodell anpassbar ist (auf Grundlage des Modularen Aufbaus von Produktmodell, Rollenmodell und Prozessabläufen

V-MODELL II



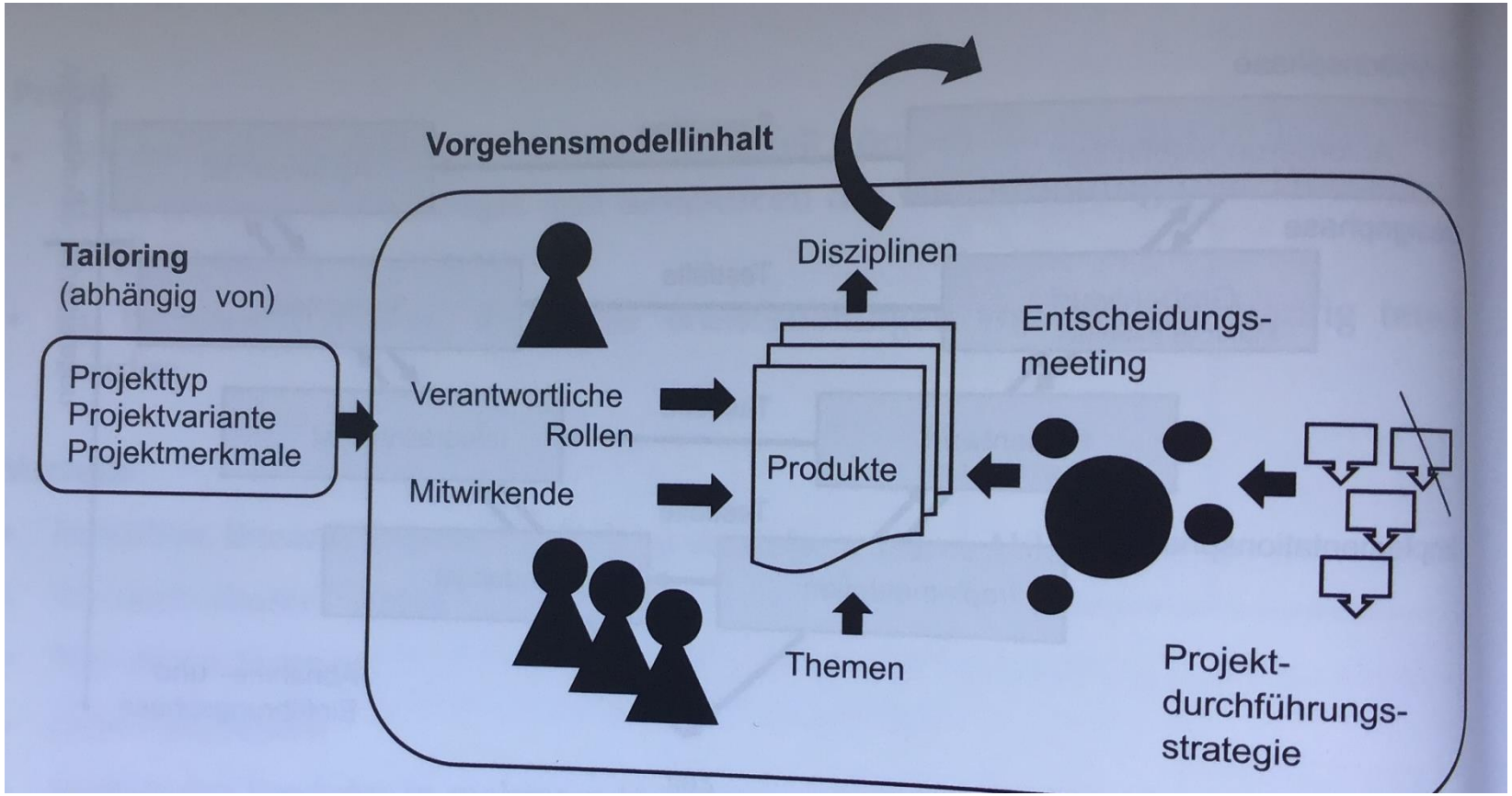
V- Modell [Met20]

Praxis:
- Häufig für Projekte in staatlichen Organisationen und Behörden
- gut für sicherheitsrelevante Projekte wg. Integrierter Testaktivitäten
- bausteinartiger Aufbau sehr praxisrelevant

Vorteile:
- geeignet für große Systeme und Komplexität, definierte Vorgaben, Ergebnismuster, Rollendefinitionen, qualitätsorientiertes Modell

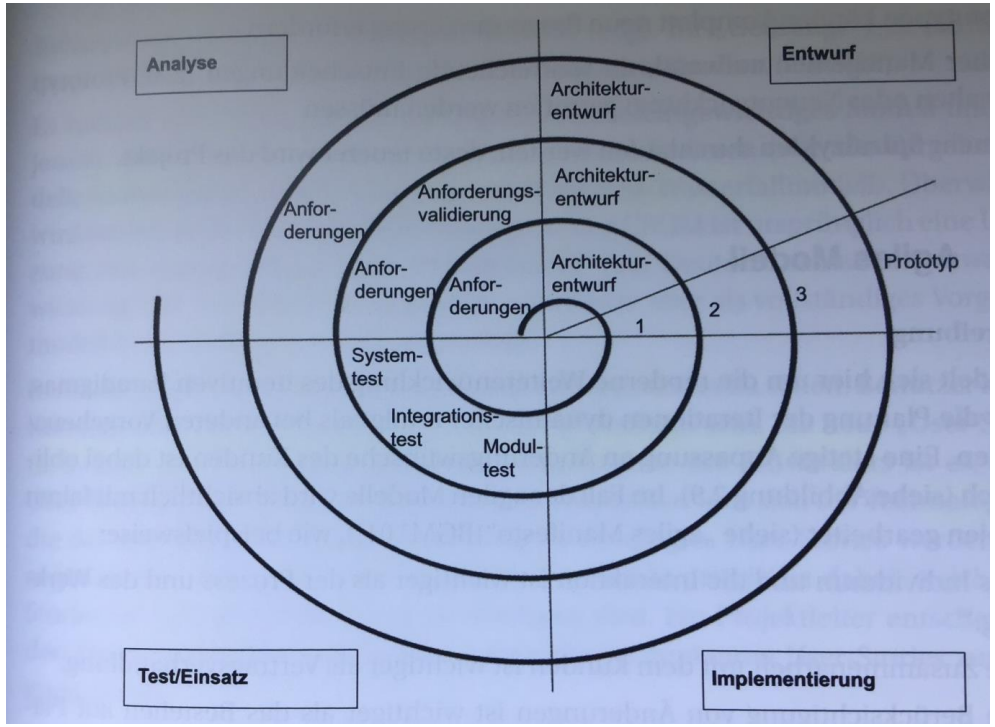
Nachteil:
- für kleine Projekte zu viel Overhead, Testaktivitäten finden recht spät statt, Phasenablauf zu strikt
- i.d.R. höhere Schulungsaufwand wg. Bausteinprinzip und Komplexität

V-MODELL XT



V- Modell XT [Met20], ursprünglich: Weit *(www.v-modell-xt.de)*

SPIRALMODELL



Spiralmodell [Met20], ursprünglich: Boehm (1988 A Spiral Model for Software Development and Enhancement)

Praxis:

- Gut geeignet für Projekte mit **Prototyping**
- viele Spiralzyklen -> teuer
- ungeeignet für große Projekte

Vorteile:

- Inkrementeller Ansatz, jeder Spiralzyklus führt zu Verbesserungen, Änderungen, Erweiterungen,
- einsatzfähige Produkte in kurzer Zeit, flexibles Modell

Nachteil:

- Phasen werden immer wieder erneut durchlaufen, komplett neue Programmierung kann erforderlich sein,
- hoher Managementaufwand, da weitreichende Entscheidungen getroffen werden müssen,
- bei vielen Spiralzyklen -> teuer

03

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

NÄCHSTE SCHRITTE

ORGA I

1. Gruppenfindung *(3 – 7 Studierende)*
2. eigener Themenvorschlag oder Themenwahl
3. Weitere Planung und (Selbst-)Organisation

04

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

THEMENVORSCHLÄGE

Generell dürfen die einzelnen Gruppen auch eigene „Zielsoftwaresysteme“ vorschlagen.

THEMENVORSCHLÄGE

1. Software für Terminvergabe und –verwaltung von (mehreren) Ämtern/ Behördenstellen.
2. Software für Lehrkräfte zur Verwaltung von Lehrveranstaltungen, Gruppen, Kursen usw..
3. Software für Wissensmanagement in einer Behörde bzw. einem Unternehmen.
4. Software für individuelles Wissensmanagement und Wissensmanagement von Gruppen.
5. „Nachbau“ von *WhatsApp* oder *Telegram*.
6. „Nachbau“ von *Amazon*.
7. „Nachbau“ von *Doctolib*.
8. Software für inhaltliche Verwaltung und Planung von Studiengängen (Modulkatallogen).
9. Nachbau von *Komoot*.
10. …