Prozesse

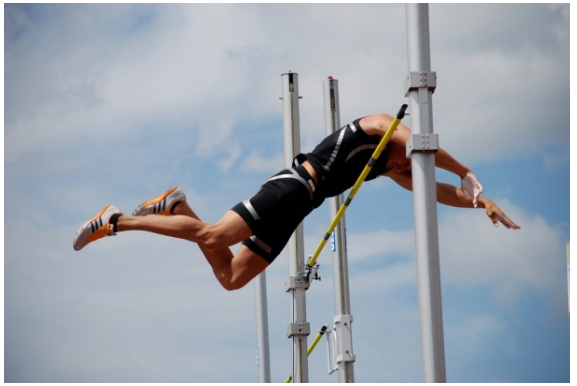
- Einführung
- Prozess-Paradigmen
  - Sequenziell
  - Iterativ
  - Notation
- Scrum
- Weitere Prozesse im Überblick

- Systemanalyse und –entwurf

DHBW Duale Hochschule Baden-Württemberg Mannheim

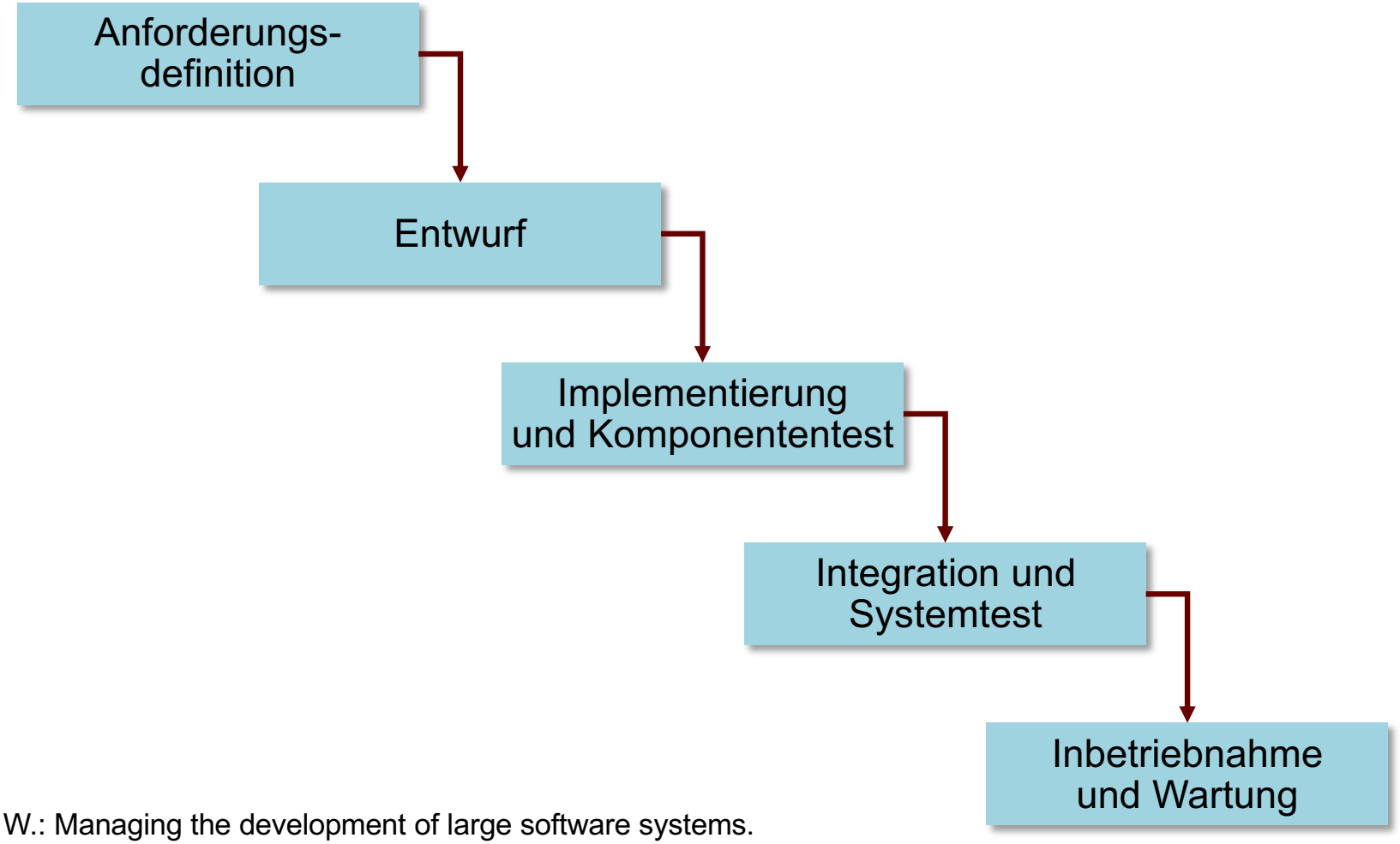
Ziele

- Sie kennen grundlegende Ansätze von Software-Entwicklungsprozessen
- Sie kennen das Basisvokabular für die Beschreibung von Prozessen



Wir nutzen diese Kenntnisse, um über den möglichen Input und den erwünschten Output von „(System-) Analyse und Entwurf“ zu sprechen

Ein einfacher Software-Prozess

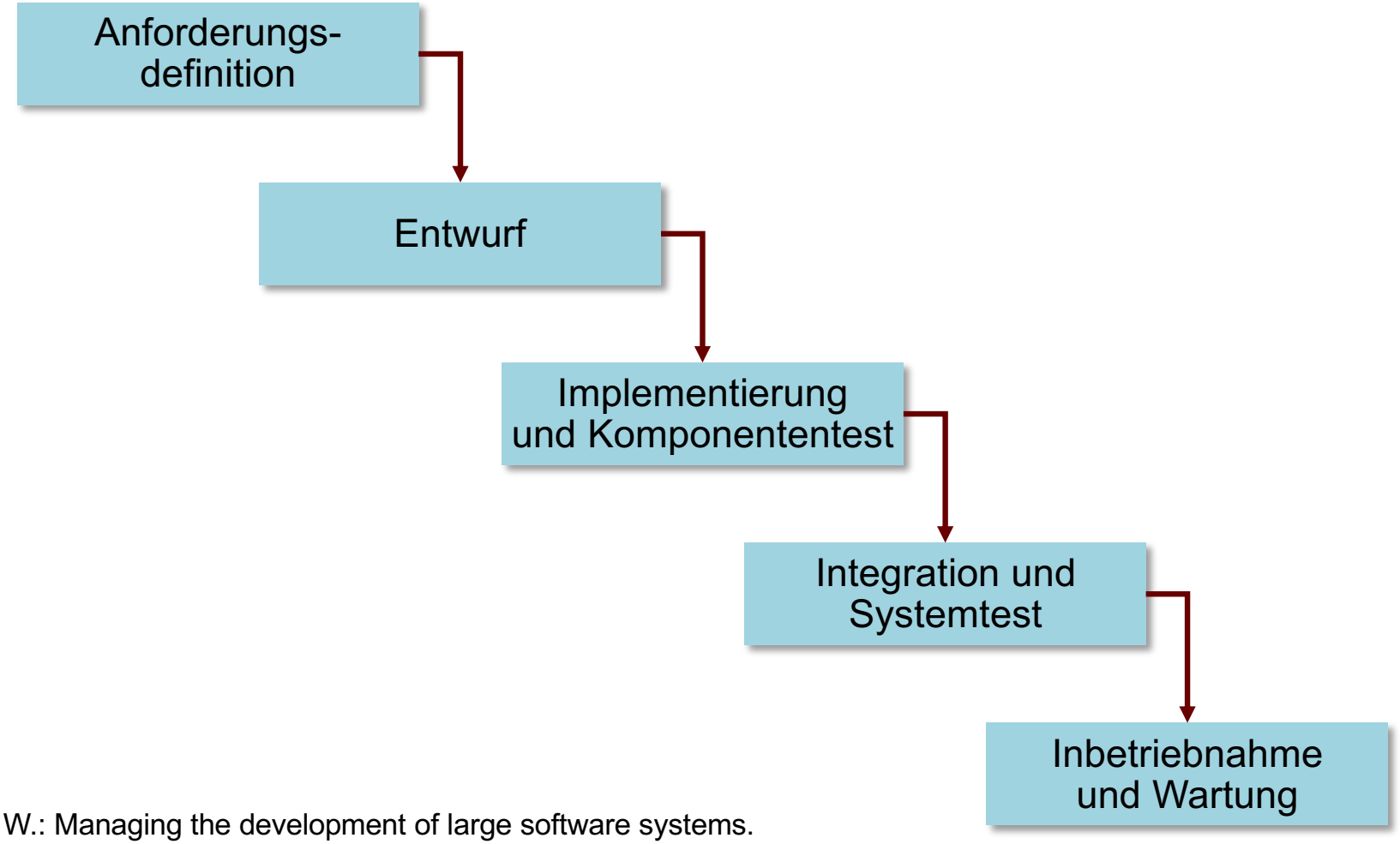


Royce, W. W.: Managing the development of large software systems.
IEEE Westcon, Los Angeles, 1970

Ein Software-Prozess definiert...

Wer?	*Verantwortung*, Durchführung
Was?	Aufgabe, *Ergebnis*
Warum?	*Begründung*, Ziel
Wann?	Zeitpunkt, Reihenfolge
Wie?	Art und Weise, Methoden
Womit?	Arbeitsmittel, Werkzeuge
Wonach?	Standard, Normen
Wofür?	Zielgruppe, Anwendungsdomäne

Ein einfacher Software-Prozess

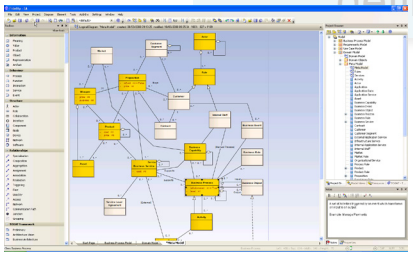


Royce, W. W.: Managing the development of large software systems.
IEEE Westcon, Los Angeles, 1970

Ein einfacher Software-Prozess

Software für Kontrollzentrum

Anforderungs-definition



Enterprise Architect

Architekt

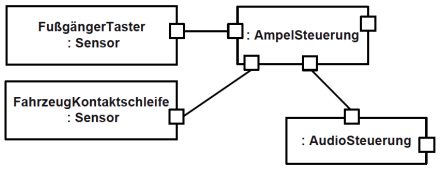
SW-Designer

Tester

Entwurf

ISO/IEC 12207

ECSS-E-40



OOA / OOD

Implementierung und Komponententest

Aerospace

Integration und Systemtest

Inbetriebnahme und Wartung

DHBW Duale Hochschule Baden-Württemberg Mannheim

Aufgaben eines jeden SE-Prozesses 1/2

- Anforderungsdefinition
  - Verstehen der Wünsche des Kunden
  - Definition der Funktionen, Randbedingungen (Kundensicht ←→ Entwicklersicht)

Specification

- Entwurf
  - Entwicklung der Architektur
  - Definition der Komponenten
- Implementierung und Komponententest
  - Entwicklung / Test der Komponenten

Design and Implementation

- Integration und Systemtest
  - Sicherstellen, dass die Software tut, was der Kunde will

Validation

Aufgaben eines jeden SE-Prozesses 2/2

- Inbetriebnahme und Wartung — Maintenance
  - Installation / Schulung beim Kunden
  - Wartung der Software
    - Weiterentwicklung
      - wegen neuen Kundenwünschen oder
      - geänderten Randbedingungen
    - Fehlerbehebung

Bei der SW-Entwicklung sind *immer alle* Aufgaben *erforderlich*
… unterschiedliche Prozesse unterscheiden sich lediglich in der Reihenfolge, in der diese Aufgaben erledigt werden (sollen)

Nutzen von Prozessmodellen 1/2

- Unterstützung bei der Kommunikation
  - „Software wird in Teams entwickelt“: Strukturierung des Ablaufs
  - Leichtere Einarbeitung von neuen Mitarbeitern
- Verteilung der Arbeit
  - Koordinierte Arbeit an großen Aufgaben
- Erleichterung / Kontrolle von „Standard“-Aufgaben
  - Standard-Vorgehen bei immer wiederkehrenden Aufgaben
  - Hilfestellung bei täglicher Arbeit
  - Verständnis der Arbeit(sergebnisse) von Anderen

DHBW Duale Hochschule Baden-Württemberg Mannheim

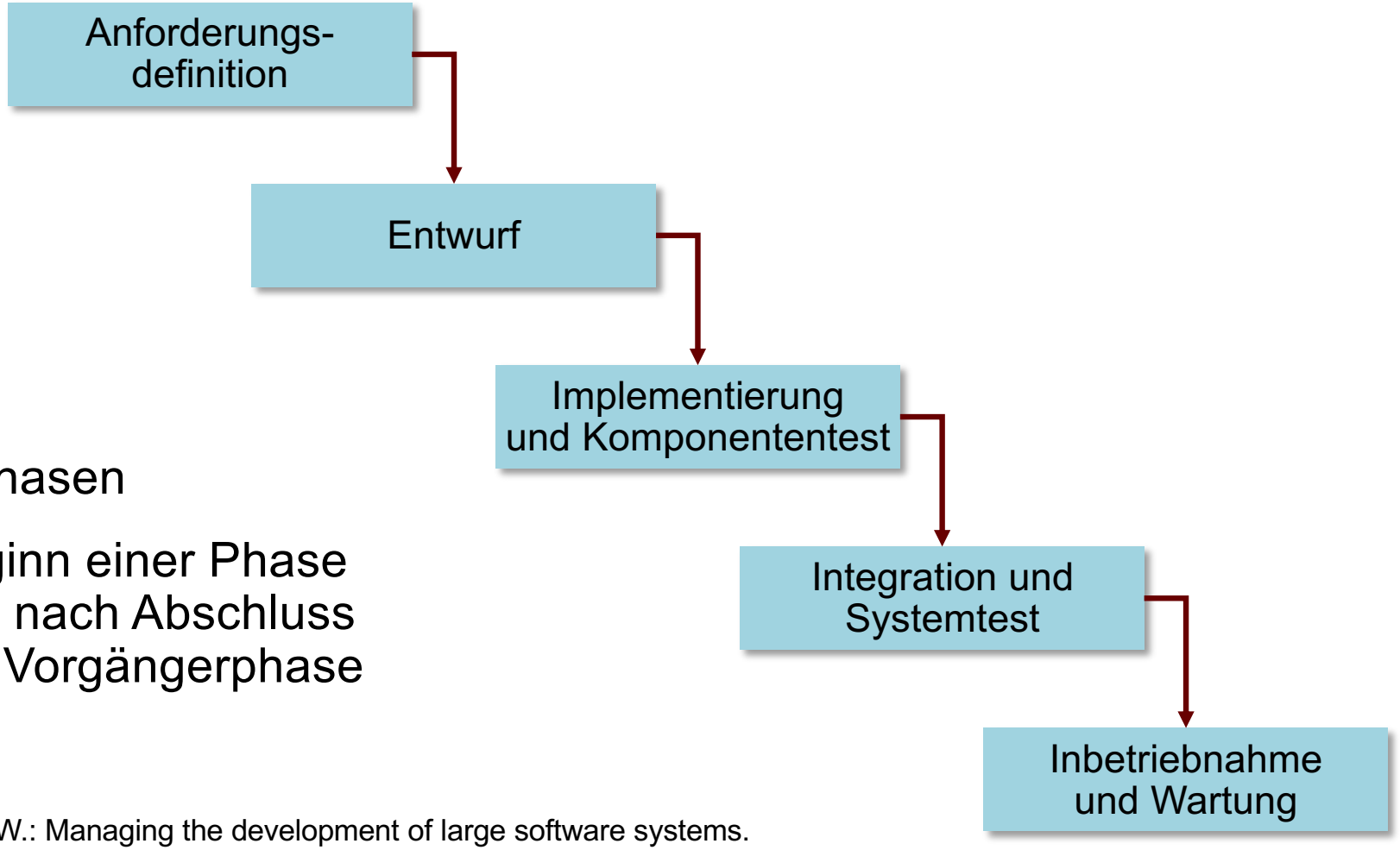
Nutzen von Prozessmodellen 2/2

- Erfahrungsaufbereitung
  - Lernen aus Erfahrung
- Projektkontrolle, -überwachung
  - Wissen über Aufgaben und Aufwand
    = Grundlage für Planung und Kontrolle
- Unterstützung durch Software-Entwicklungsumgebungen
  - „Software wird Tool-unterstützt erstellt“
  - Unterstützen von Standard-Aufgaben mit (Standard-)Tools

DHBW Duale Hochschule Baden-Württemberg Mannheim

Inhalt

Wasserfallmodell

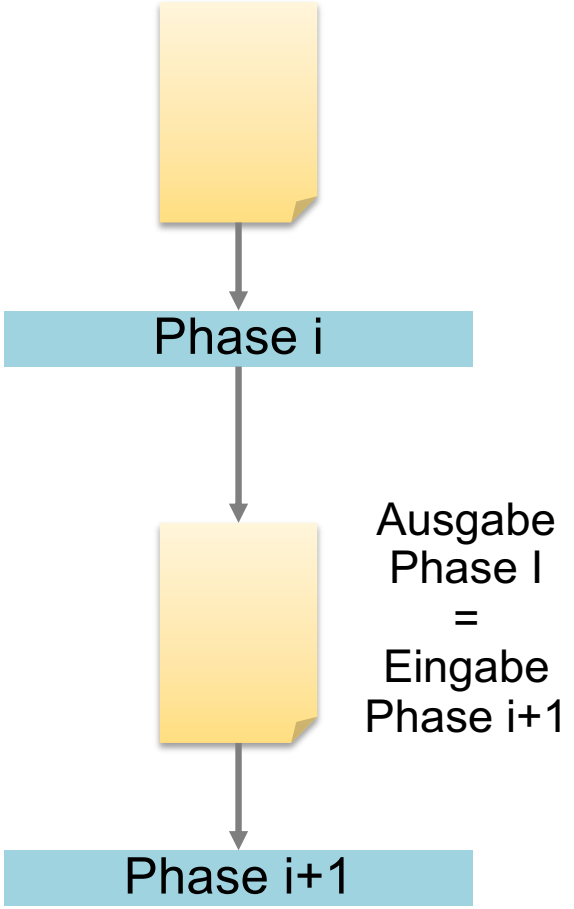


- 5 Phasen
- Beginn einer Phase erst nach Abschluss der Vorgängerphase

Royce, W. W.: Managing the development of large software systems. IEEE Westcon, Los Angeles, 1970

Dokumentation

- Eingaben
  - Dokumente, die in einer Phase verwendet werden
- Ausgaben
  - Dokumente, die in einer Phase entwickelt werden
- Beim Wasserfall
  - Eingabe in Phase 1: Anforderungsdefinition = Kundenwünsche
  - Ausgabe aus Phase i = Eingabe in Phase i+1
  - Ausgabe aus Phase 5: System plus ...



Ausgabe der Anforderungsdefinition

- Ergebnis
  - Was soll das zu entwickelnde System können?
    Wie soll es das durchführen?
- Zweck
  - Festlegen der Anforderungen
  - Gemeinsames Verständnis von Kunde (Auftraggeber) und Entwicklungsteam (Auftragnehmer)
- Beispiele
  - “Das System muss ein Produkt mit Bild, Details und Preis anzeigen (Angaben wie im bisherigen Papierkatalog)”.
    “Das System muss für einen angemeldeten Käufer einen Warenkorb führen”.
    “Das System muss 24/7 zur Verfügung stehen: Pro Woche sind max. 30 Minuten Ausfallzeit (für Wartung, Fehlerbehebung etc.) erlaubt.”

Ausgabe des Entwurfs

- Ergebnis
  - Wie ist das zu entwickelnde System strukturiert?
  - Welche Schnittstellen gibt es nach außen und innerhalb?
- Zweck
  - Planung, Basis für inkrementelle Entwicklung, Arbeitsaufteilung
  - Festlegen der Software-Architektur und -Struktur
  - Gemeinsames Verständnis des Programmaufbaus durch die (möglicherweise global verteilte) Entwicklungsmannschaft
- Beispiel

Ausgabe von Implementierung & Komponententest

- Ergebnis
  - Dokumentierter Code
  - Getestete, ausführbare Komponenten
- Zweck
  - Bereitstellen einer Menge von funktionsfähigen Komponenten
- Beispiele
  - Bibliotheken
    - a.dll, b.a, c.jar, d.war, e.ear
  - Skripte
    - A.sh, b.pl, c.py

DHBW Duale Hochschule Baden-Württemberg Mannheim

Ausgabe von Integration und Systemtest

- Ergebnis
  - Ausführbares System
  - Installations- und Benutzungsanleitung
- Zweck
  - Bereitstellen eines beim Kunden lauffähigen Systems
- Beispiel
  - Installierbare Anwendung in Form
    - eines Installationsprogrammes (a.exe)
    - eines Softwarepakets für einen Paketmanager (apt / apt-get, MacPorts, rpm)

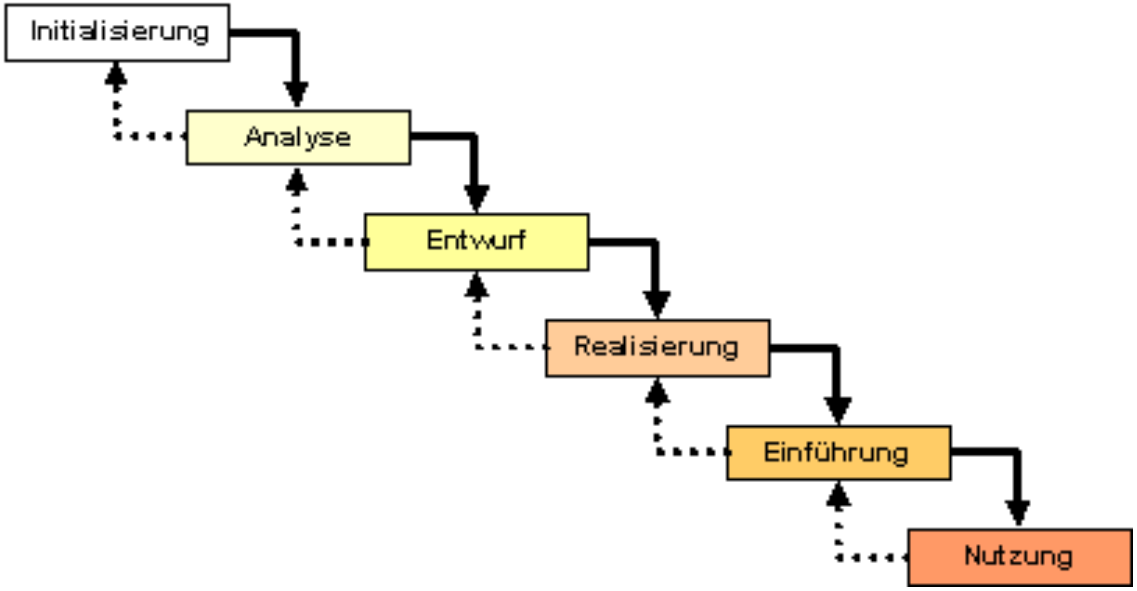
Ausgabe von Inbetriebnahme und Wartung

- Keine Ausgabe…
  - ... am Ende der Wartungsphase wird die Software ausgemustert

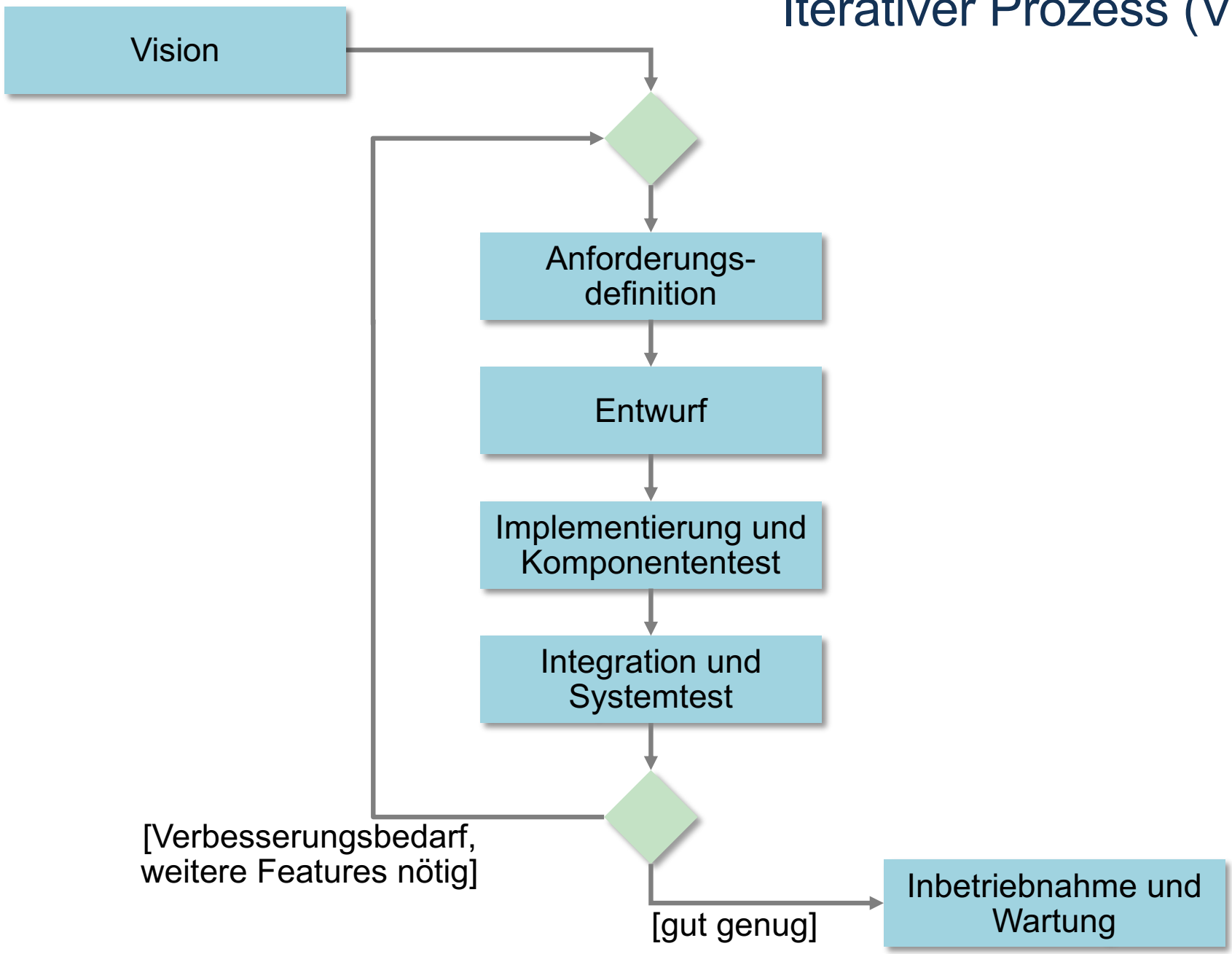
Anwendung des Wasserfallmodells

- bei gut verstandenen („klaren“) Anforderungen
- falls keine/wenig Änderungen der Anforderungen erwartet werden

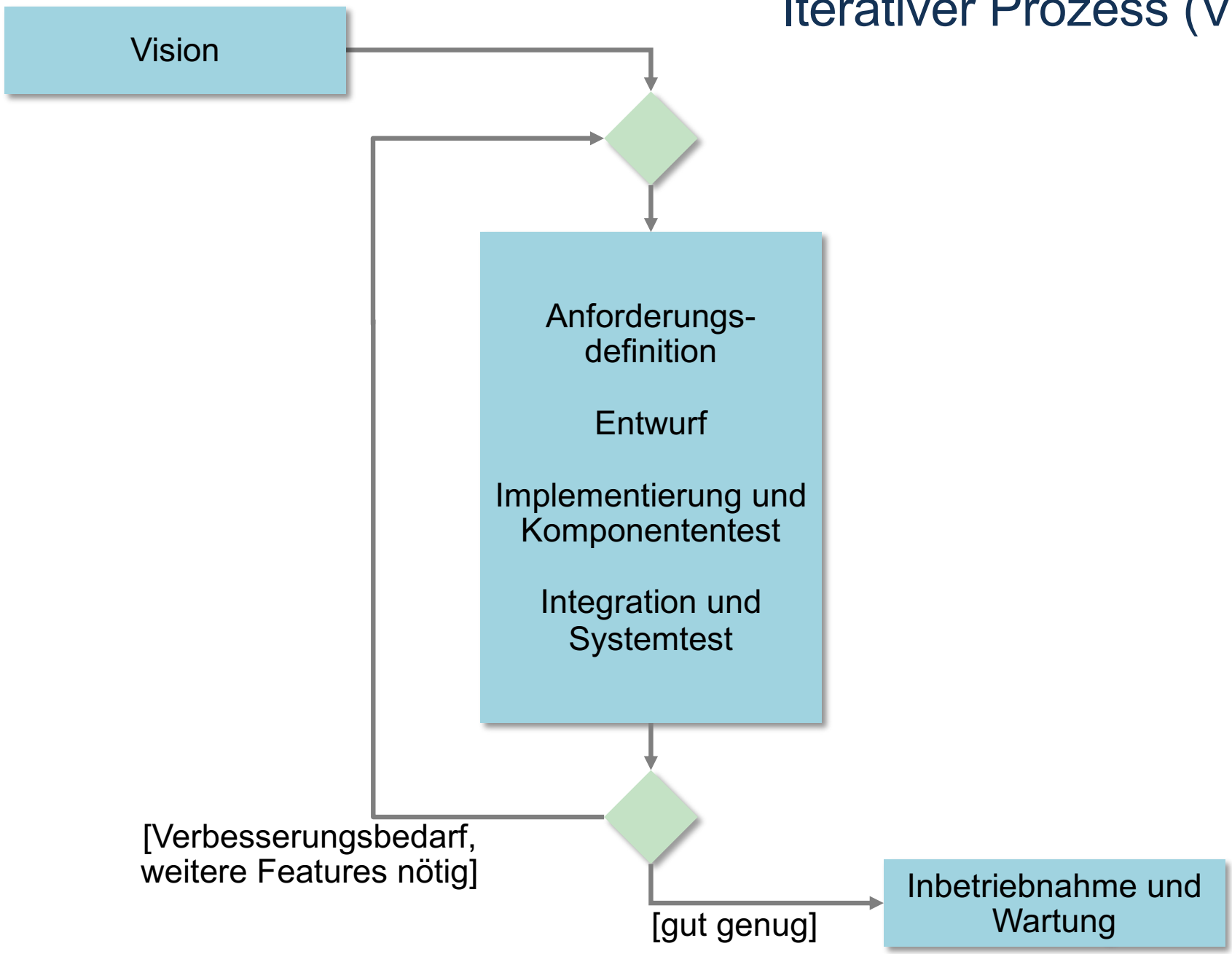
- Normalerweise in Varianten
  - z.B. „mit Rücksprung“



Iterativer Prozess (V1)



Iterativer Prozess (V2)

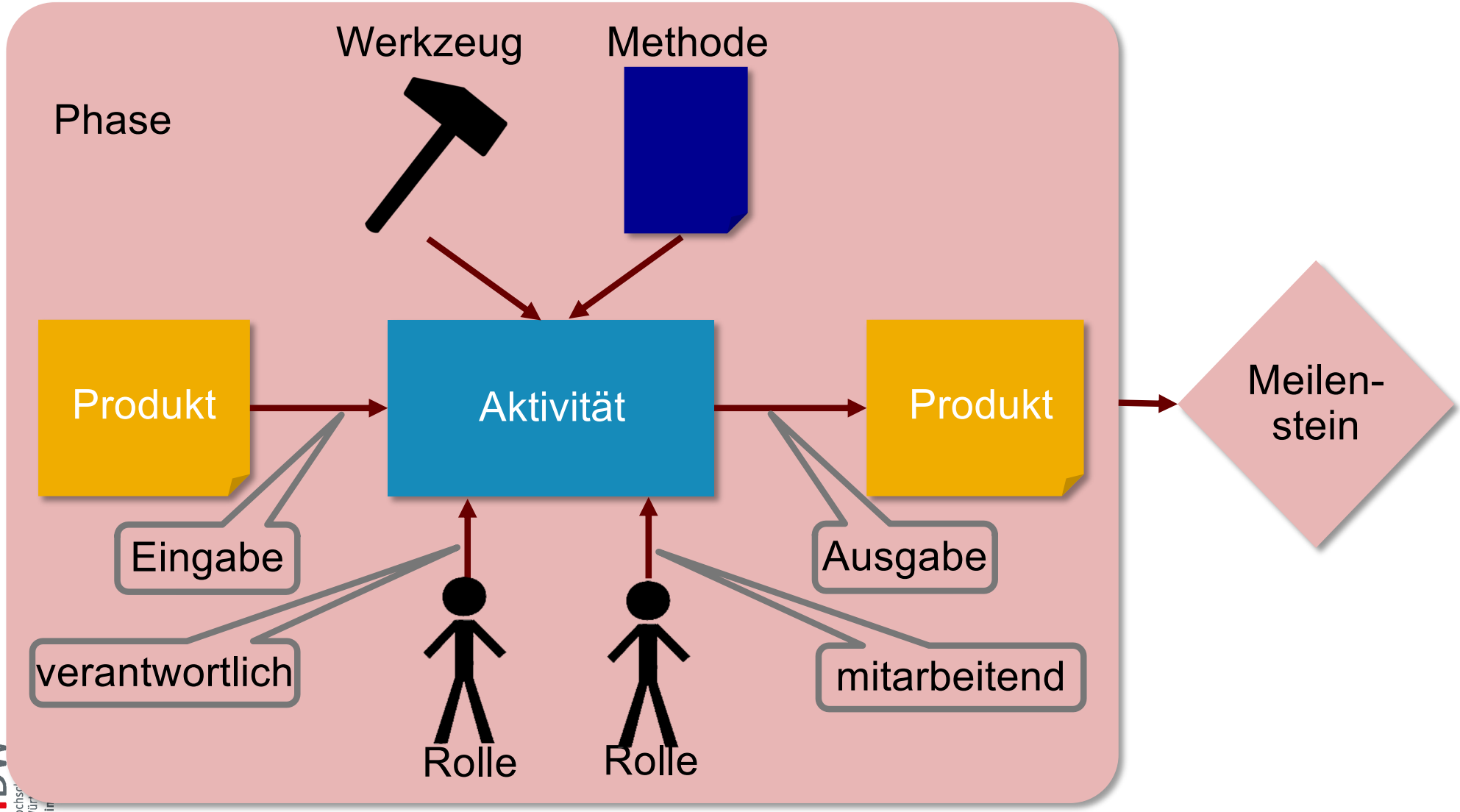


Anwendung iterativer Modelle

- Anwendung
  - Projekte mit offenen Fragen (Technologie, Domäne, …)
  - Projekte mit unklaren / sich ändernden Anforderungen
  - Komplexe (bzgl. Technik, Domäne, …) Projekte

...also bei den meisten Projekten möglich / sinnvoll

Elemente und Notation von Prozessen: Rollen, Werkzeuge, Produkte, Phasen, Meilenstein



Produkt

- Ein *Produkt (Artefakt)* ist (niedergelegte) Information, die erstellt, modifiziert oder verwendet wird

- Endprodukt
  - Wird an den Nutzer / Kunden geliefert
- Zwischenprodukt
  - Wird für andere Entwicklungsschritte verwendet

- Bsp.: Entwurfsspezifikation, Klassendiagramm, Code

DHBW Duale Hochschule Baden-Württemberg Mannheim

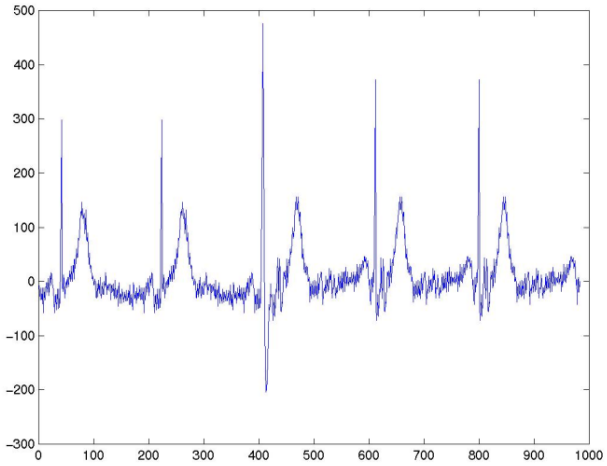
Rolle

- Eine *Rolle* fasst Fähigkeiten / Verantwortlichkeiten zusammen
- Zur Durchführung einer Aktivität sind i.d.R. bestimmte Fähigkeiten erforderlich, d.h. eine bestimmte Rolle kann diese Tätigkeiten durchführen
- Eine (jede) Person, die diese Fähigkeiten hat, kann eine Rolle wahrnehmen
- Eine Rolle kann von mehreren Personen ausgeübt werden
- Eine Person kann mehrere Rollen innehaben
- Bsp.: Entwickler, Tester, Konfigurator

Aktivitäten

- Eine *Aktivität* ist ein Arbeitsschritt zum Erreichen eines Zieles

- Typischerweise
  - werden Produkte verwendet
  - werden Produkte erzeugt / geändert
  - führen Rollen den Schritt aus
  - werden Methoden eingesetzt



- Hierarchische Struktur möglich
  - (große) Aktivität besteht aus (kleineren) Aktivitäten

- Bsp.: Entwerfen der Klassen, Reviewen der Klassen

Werkzeuge / Methoden / Phase / Meilenstein

- Werkzeuge
  - Bsp.: Eclipse, JUnit, Git
- Methoden
  - Bsp.:
    - Strukturiertes Interview (für Anforderungserhebung)
    - OOD (Objektorientiertes Design -> Entwurf)
    - Blackbox-Test (für Integration und Systemtest)
- Phase
  - Gruppierung von Aktivitäten
- Meilenstein
  - Überprüfbares (Zwischen-)Ergebnis
  - Bsp.: “Anforderungsdokument vom Kunden unterzeichnet”

Inhalt

1. Einführung
2. Prozessparadigmen
3. Scrum
4. Weitere Prozesse im Überblick

Scrum

Scrum und Rugby

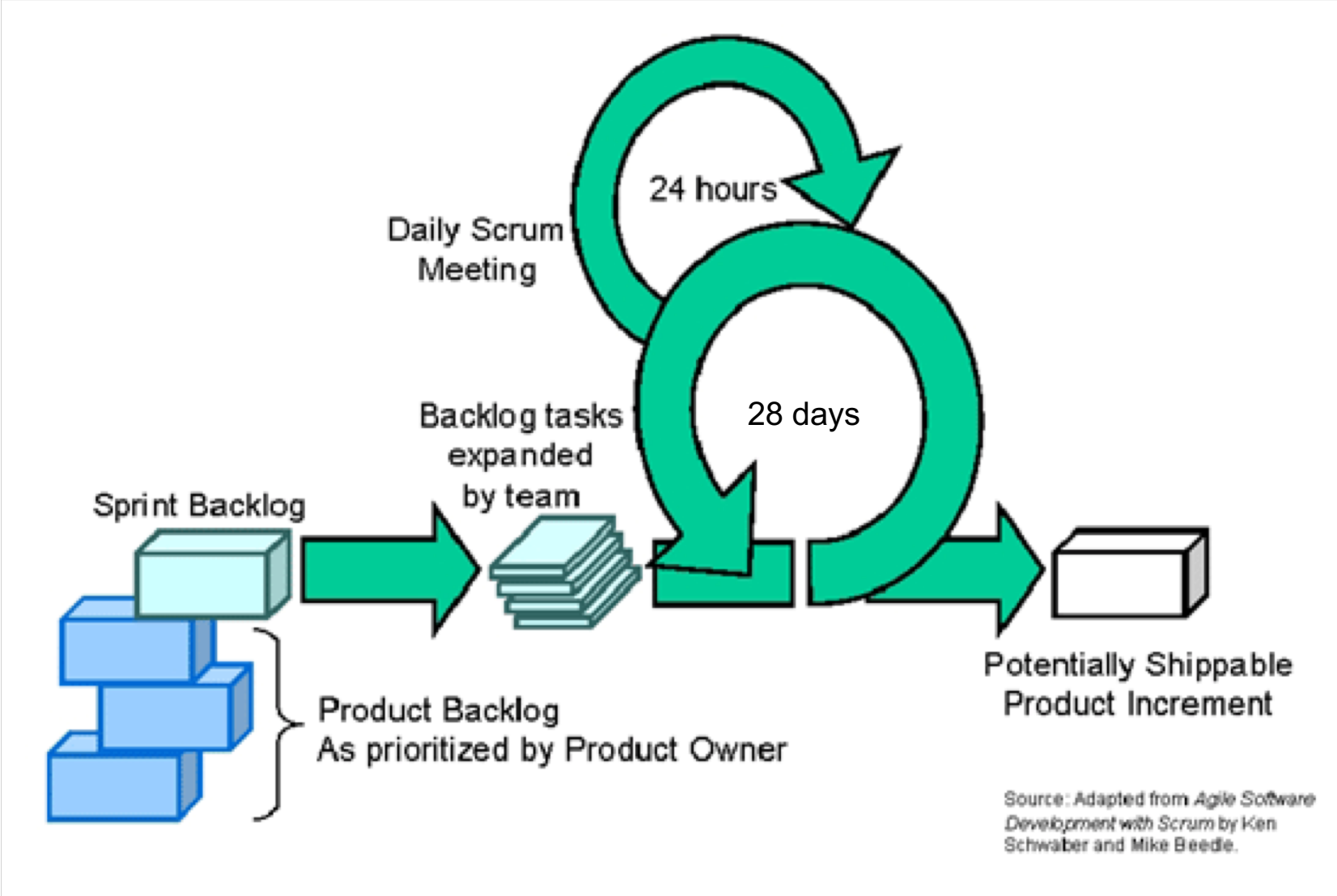
„Both are adaptive, quick, self-organizing, and have few rests“

Ken Schwaber

Annahme von Scrum

„You can‘t predict or definitely plan
what you will deliver,
when you will deliver it, and
what the quality and cost will be“

Scrum: Vorgehen



Rollen

- Product Owner
  - Priorisiert Anforderungen
  - Wählt Anforderungen aus dem Product Backlog in das Sprint Backlog
  - Gibt als einziger Ziele vor
- Scrum Master
  - Sorgt dafür, dass das Team ungestört arbeiten kann
  - Unterstützt den Scrum-Prozess (in Projekt und Organisation)
- Team
  - Setzt Anforderungen in Produkt um
  - Übernimmt alle anfallende Aufgaben
    - keine fachliche oder organisatorische Hierarchie
  - Teamgröße: 5 +/- 2

Kein Projektleiter --
das Team organisiert sich

Kernprodukte

- Product Backlog
  - Priorisierte Anforderungen (*User Stories, Epics*), die möglicherweise im Produkt realisiert werden können
- Sprint Goal
  - Wesentliches Ergebnis (aus Sicht des Product Owners) für den Sprint
- Sprint Backlog
  - Aufgaben (*Tasks*), die notwendig sind, um die für einen Sprint ausgewählten Anforderungen (User Stories) zu realisieren
  - Eine User Story sollte in max. 1 Woche umsetzbar sein

Das war alles?

- Scrum ist ein Managementprozess
  - …der gut zu Software, nicht aber zu Brücken passt
- Scrum (für die Softwareentwicklung) lässt sich gut verbinden mit
  - Kanban
    - „Visualisiere!“
    - “Begrenze den Durchfluss!“
      (wenige Arbeitspakete gleichzeitig bearbeiten)
  - XP (eXtreme Programming)
    - Verwenden von Praktiken wie
      - Pair Programming
      - Test Driven Development
      - Continuous Integration

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

CHAOS FACTORS OF SUCCESS

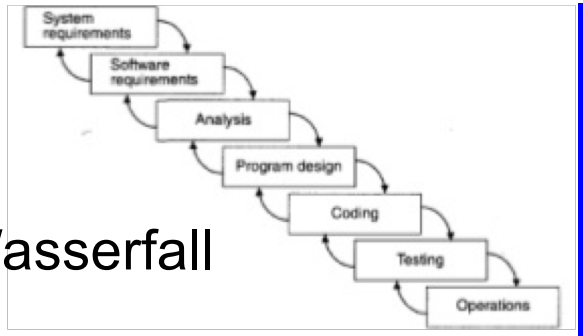
FACTORS OF SUCCESS	POINTS
Executive Sponsorship	15
Emotional Maturity	15
User Involvement	15
Optimization	15
Skilled Resources	10
Standard Architecture	8
Agile Process	7
Modest Execution	6
Project Management Expertise	5
Clear Business Objectives	4

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

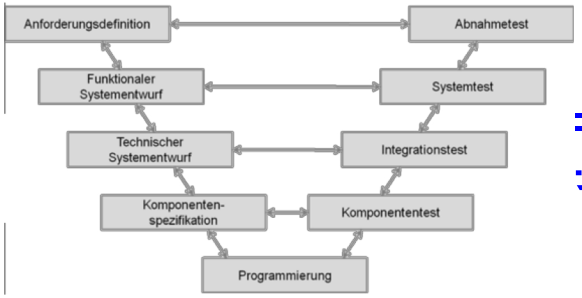
Inhalt

1. Einführung
2. Prozessparadigmen
3. Scrum
4. Weitere Prozesse im Überblick

Beispiele für SE-Prozesse



Wasserfall



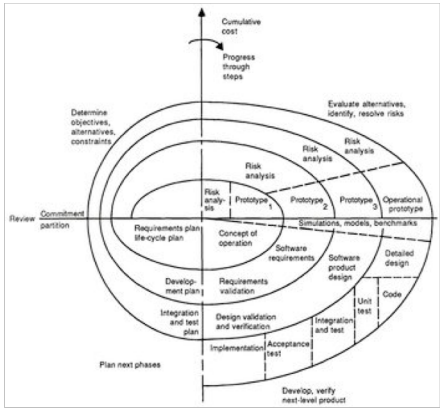
V-Modell

sequentiell

iterativ

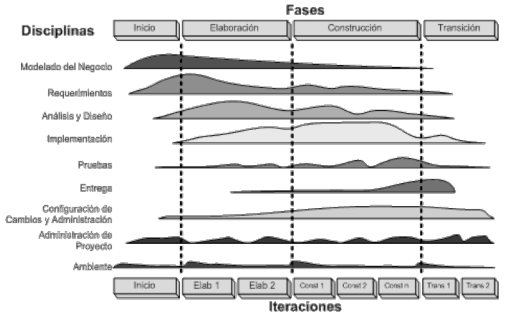
plangetrieben

agil

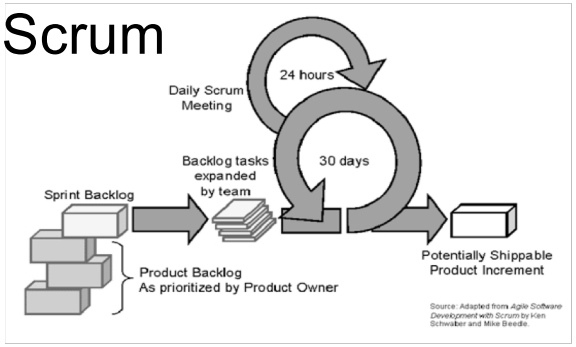


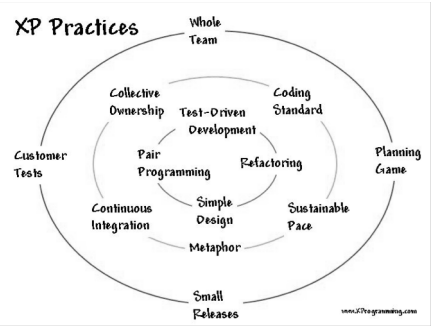
Spiralmodell

RUP
Unified Process



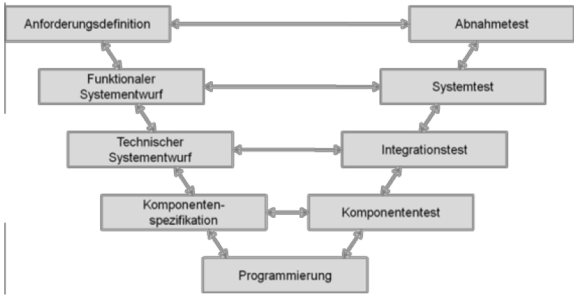
Scrum



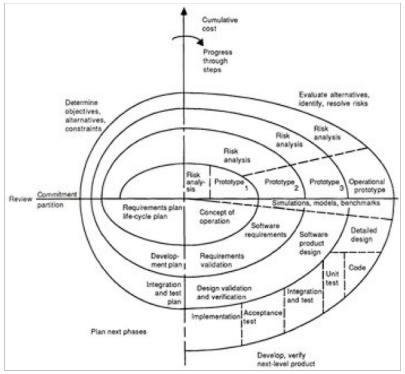


Extreme Programming

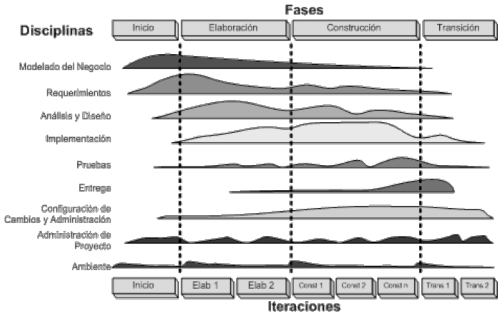
V-Modell / Spiralmodell



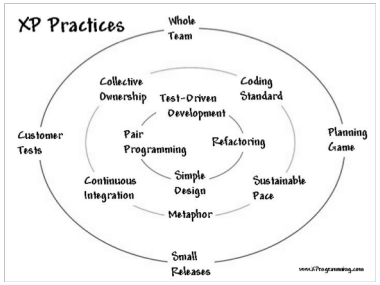
- V-Modell
  - Sequentiell (wie Wasserfall)
  - Wesentliche Idee
    - jeder Tätigkeit / Phase explizit eine Testtätigkeit / -phase zuordnen
  - Verbesserte / erweiterte Version: V-Modell XT
- Spiralmodell
  - Sequentiell oder iterativ
  - Wesentliche Idee
    - Vor jeder Phase wird eine Risikoabschätzung (rechter oberer Quadrant) gemacht und es werden mit Hilfe von Prototypen offene Punkte geklärt
    - Die “eigentliche” Projektarbeit findet im rechten unteren Quadranten statt. Die Tätigkeiten können dort sequentiell oder iterativ eingebettet sein.



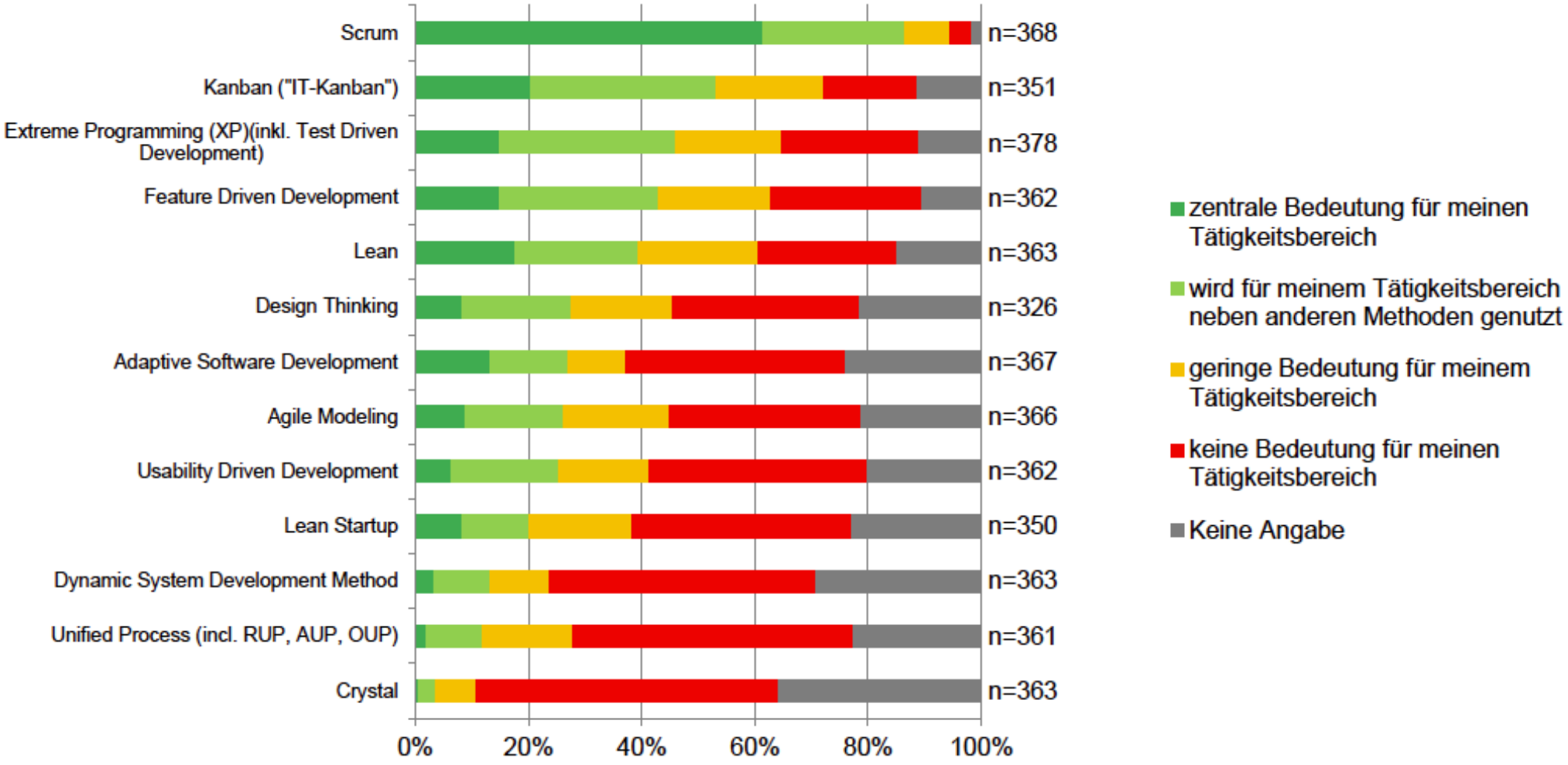
RUP / XP



- RUP – (Rational) Unified Process
  - Iterativ
  - Wesentliche Idee
    - Zu Beginn die Architektur festlegen
    - In jeder Iteration mit Anforderungsänderungen umgehen
    - Früh mit Testen beginnen
    - Grafische Modellierung der Software (UML)
    - Inkrementelle Erstellung der Dokumentation / des Produkts
- XP – eXtreme Programming
  - Iterativ, agil
  - Wesentliche Idee
    - Sammlung von *Praktiken* für erfolgreiche, agile Entwicklung
    - Bsp.: Pair Programming, TDD, Sustainable Pace



Studie an der HS Koblenz



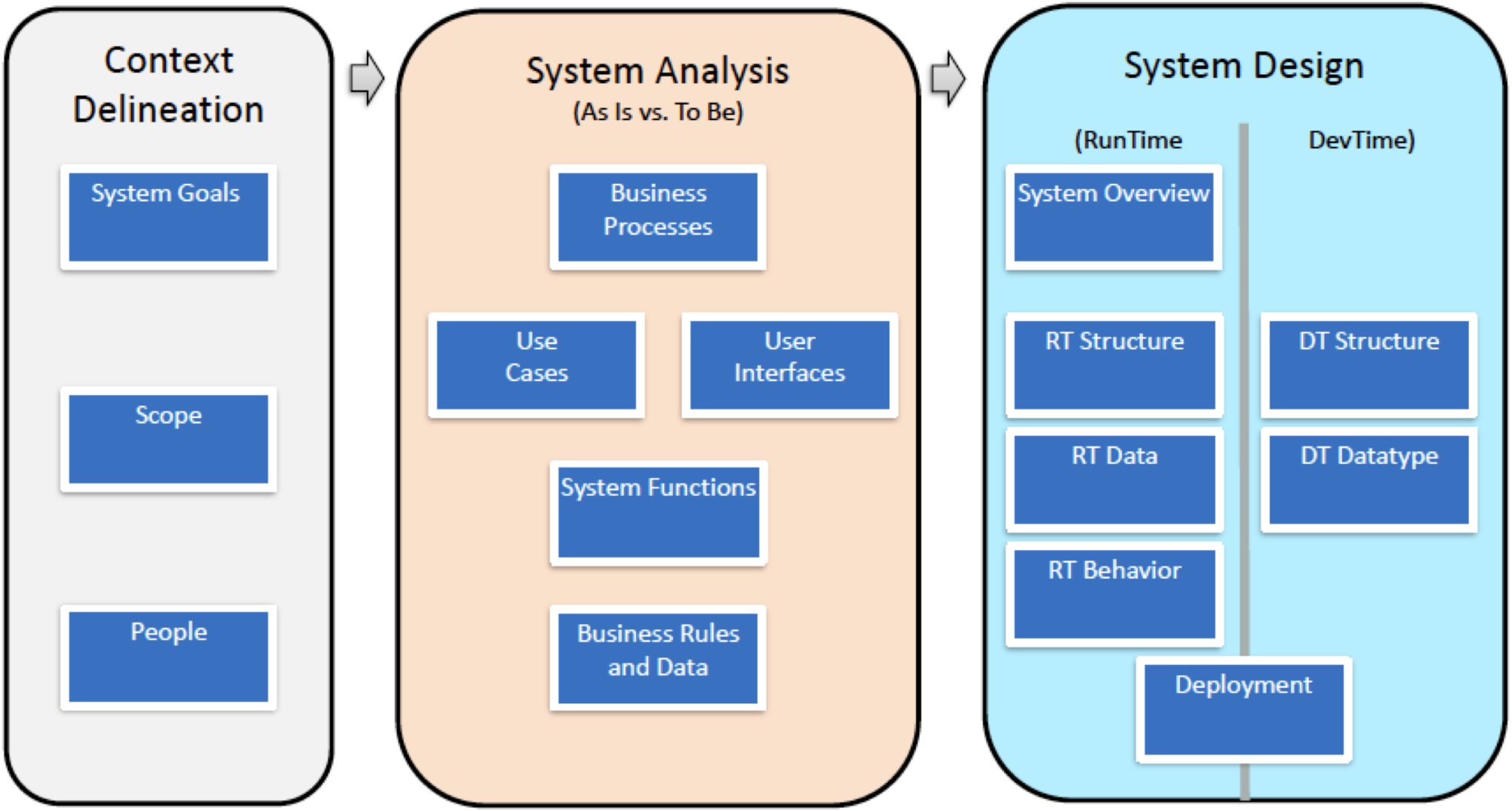
www.status-quo-agile.de | © BPM-Labor, Prof. Dr. A. Komus

Plangetrieben vs. Agil (sehr komprimiert)

	Plangetrieben	Agil
Planung	+ welche Funktion + welche Kosten + welcher Termin	+ welches Team + Sprintdauer + wie viele Sprints
Kontrolle	+ Überwachung Kosten, Termine	+ implementierte Funktionen abnehmen + neue Funktionen festlegen
Methoden	+ explizite Dokumentation	+ Kommunikation
Scope	+ „groß“	+ „klein“

F R A G E N

Überblick



Problemraum ⟵ ⟶ Lösungsraum

Abgrenzung des System-Kontexts

Zur Abgrenzung gehören:

- Beschreiben der Ziele für das System
  - Ziele des Systems / des Projekts
  - Gründe für die Investition / Erwartungen bzgl. ROI
  - Kosten des Projekts
- Abgrenzen des Projektumfangs
  - Alleinstellungsmerkmale (USP)
  - Wesentliche Features / Qualitätsmerkmale
  - Relevante Einschränkungen (gesetzlich, technisch, organisatorisch)
- Identifizieren der Stakeholder
  - Stakeholder, Betroffene (Abteilungen, Rollen, Personen etc.)

DHBW Duale Hochschule Baden-Württemberg Mannheim

Systemanalyse

Zwei mögliche Auffassungen des Begriffs:

Soll-Analyse

- Unter Systemanalyse wird meist der Prozess verstanden, bei dem ein *noch nicht existierendes* Hard-oder Software-Projekt [*System*] so beschrieben wird, wie es in der Zukunft aussehen soll.

Ist-Analyse

- Allerdings kann damit auch gemeint sein, dass ein *bestehendes System* so beschrieben wird, wie es in der Realität abgebildet ist.

[SOPHIST GROUP Chris Rupp. Systemanalyse kompakt. Spektrum Akademischer Verlag, Berlin Heidelberg, 2 edition, 2008]

Systementwurf

- Unter Systementwurf versteht man den Prozess, die Elemente eines Systems zu entwerfen / zu definieren, d.h. seine Architektur, Komponenten (Module), Schnittstellen dieser Komponenten und Datenflüsse durch das System.

- Durch ingenieurmäßiges Vorgehen soll ein kohärentes und gut funktionierendes System entworfen werden, das bestimmte Anforderungen einer Organisation / eines Unternehmens erfüllt.

[Techopedia - The IT Education Site;
https://www.techopedia.com/definition/29998/system-design]

F R A G E N