

Hochschule Karlsruhe  
Technik und Wirtschaft  
**UNIVERSITY OF APPLIED SCIENCES**

# IT-Projektmanagement

# Wer ist schon gern ein Dummkopf?

„Für Bent Flyvbjerg, Professor für Stadtplanung an der Universität Oxford, sind die Hiobsbotschaften von deutschen Großbaustellen kein Zufall:

## **"Die meisten Projektmanager sind Dummköpfe oder Lügner",**

sagt der Volkswirt. Um den Zuschlag zu bekommen, würden Baufirmen die Kosten ihres Vorschlags regelmäßig als möglichst niedrig darstellen, den wirtschaftlichen Nutzen als gewaltig und die Bauzeit als minimal. Und Politiker, gierig nach Prestigeprojekten und Fotogelegenheiten beim Spatenstich, würden ihnen nur zu gerne glauben.

Dazu kämen allzu menschliche Planungsfehler: Wer Monate seiner Arbeitszeit in die Planung eines Projekts steckt, will einfach ans Gelingen glauben - und blendet Problemchen [...] im Dienste der Vision einfach mal aus. "Hang zum Optimismus", nennt das der Psychologe und Wirtschaftsnobelpreisträger Daniel Kahneman.“

Quelle: Alexander Demling auf SpiegelOnline, 10.1.2013, <http://goo.gl/WJBn7>

# Gliederung IT-Projektmanagement



<b>IT-Projektmanagement</b>
1. Einführung
2. Der Projektbegriff
3. Klassische Vorgehensmodelle im Projektmanagement
4. Die Projektbeteiligten: Rollen und Funktionen
5. Lasten- und Pflichtenhefte
6. Vertiefung Netzplantechnik
7. Die Kapazitätsplanung
8. Risikomanagement
9. Vom Problem über die Idee zum Lösungsansatz: Design Thinking
<b>10. Agile Ansätze</b>
10.1. XP
10.2. Scrum
10.3. Kanban in der IT
10.4. Continuous Delivery
11. Projektcontrolling: Informationssystem und Berichtswesen im Projekt
12. Qualitätsmanagement im Projekt
13. Klausurvorbereitung

# Projekte

---

- Projekte begegnen uns tagtäglich
- Dynamik der Märkte führt zu Projektflut
- Komplexität der Projekte nimmt zu

# Bekannte Projekte

Arche Noah

Pyramiden

Brasília

EURO



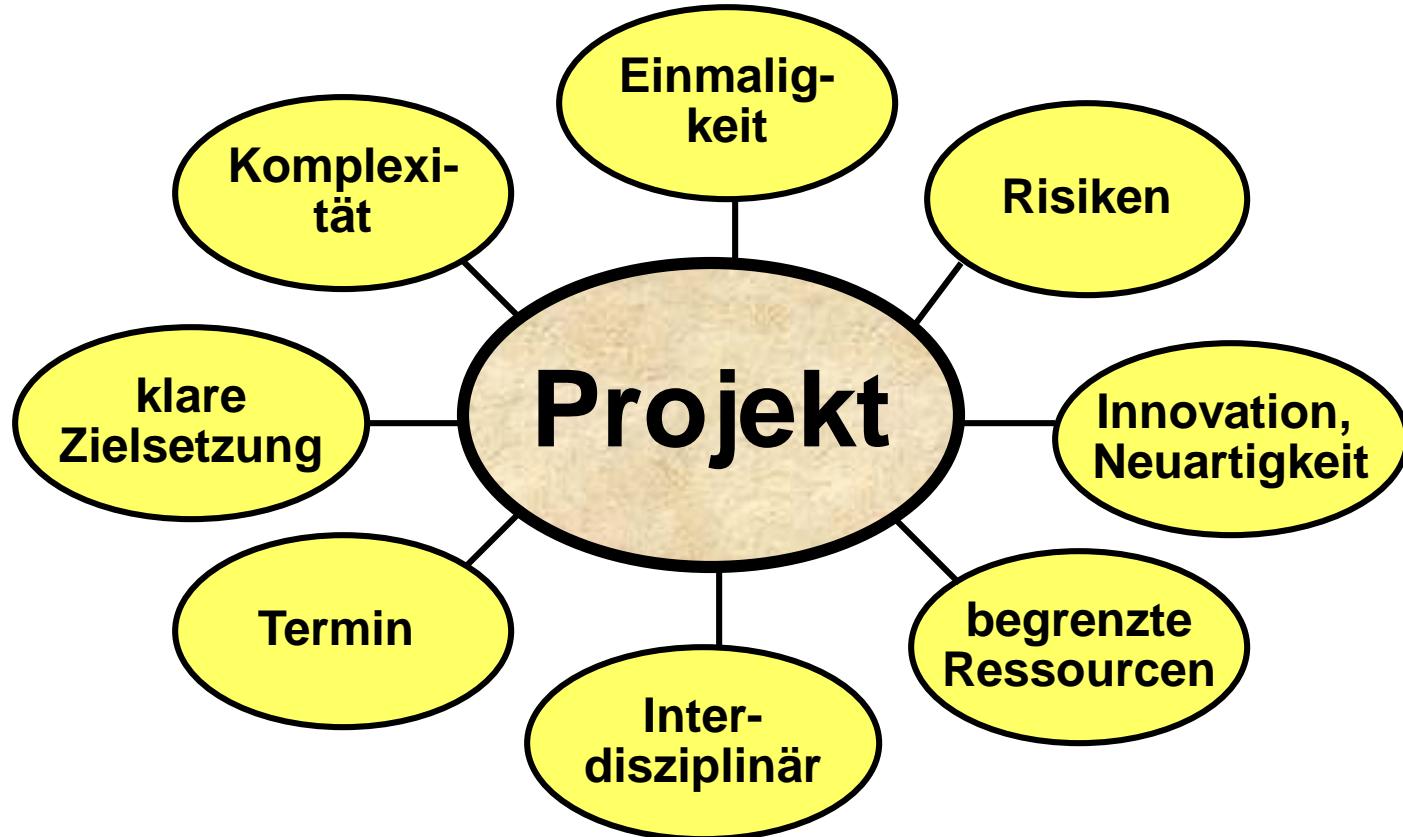
Stuttgart21

Flughafen BER

## Projekte in Unternehmen

- **Entwicklung eines neuen Produktes**
- **Entwicklung einer neuen Serviceleistung**
- **Neuorganisation der Firma / einer Abteilung**
- **Aufbau / Schließung einer Organisationseinheit**
- **Fusion mit einer anderen Firma**
- **Entwicklung / Einführung eines neuen Informationssystems**

# Projekte: Merkmale



DIN 69901

## Projekt

- Einmaligkeit
- Zielvorgabe
- Ressourcenbegrenzung
- Komplexität
- Interdisziplinarität

- Herkunft unklar. Vermutlich lat.: manum agere = an der Hand führen
- Führen
- Entscheiden
- Verantwortung übernehmen
- Aber auch:

---

## Wahrig Deutsches Wörterbuch

**Ma|nage|ment** <['mænidʒmənt] n. [15](#)> Gesamtheit der Führungskräfte eines Unternehmens; die Führungsmethode eines Unternehmens od. Betriebes [engl., „Leitung, Führung“]

## Projekt

- Einmaligkeit
- Komplexität
- klares Ziel
- begrenzte Ressourcen
- Risiko



## Management

- Führung
- Zielsetzung
- Organisation
- Planung
- Kontrolle & Steuerung

# Projektmanagement

# Projektmanagement II

---

- Lt. DIN 69901: Gesamtheit von Führungsaufgaben, Organisation, Techniken und Mitteln, für die Abwicklung eines Projektes.

# Was bedeutet das?

- **Führungsaufgaben**
  - Zieldefinition
  - Kontrolle und Steuerung
- **Führungsorganisation**
  - Projektorganisation
  - Projektabwicklung
- **Führungstechniken**
  - Motivation der Projektmitarbeiter
  - Besprechungs- und Präsentationstechniken
  - Entscheidungsfindungstechnik
- **Führungsmittel**
  - Projektplanungs- und Steuerungssysteme  
(Arbeitspakete, Kapazitäten, Termine/Meilensteine, Kosten)

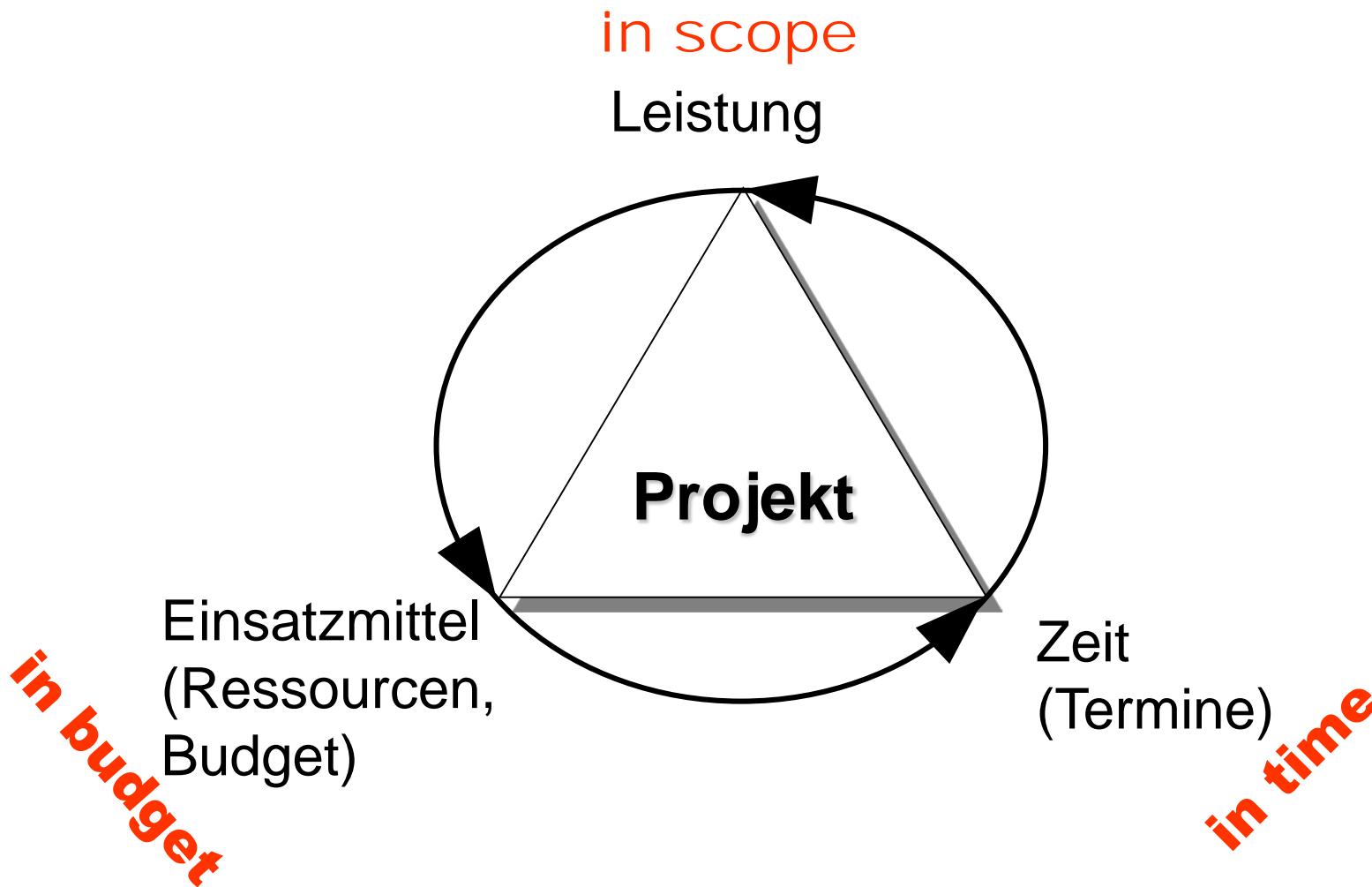
# Kurz gesagt:

---

## Projektmanagement ist:

- Die direkte, fachübergreifende Leitung/Steuerung aller Projektprozesse.
- Es gibt:
  - Fachliche Anforderungen
  - Wirtschaftliche Anforderungen
  - Soziale Anforderungenan das Projektmanagement.

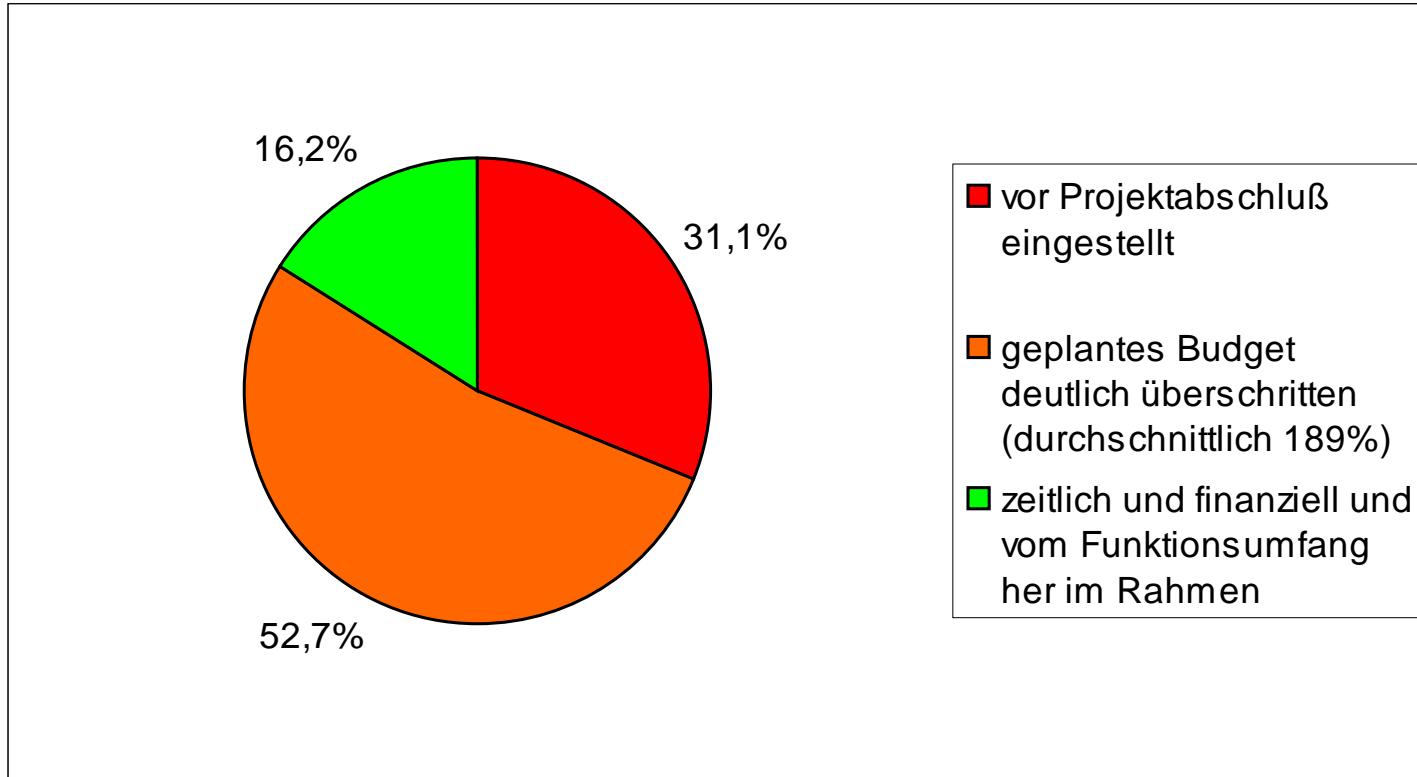
# Das magische Projektmanagement – Dreieck nach Burghardt



# Spezielle Problematik von Softwareprojekten

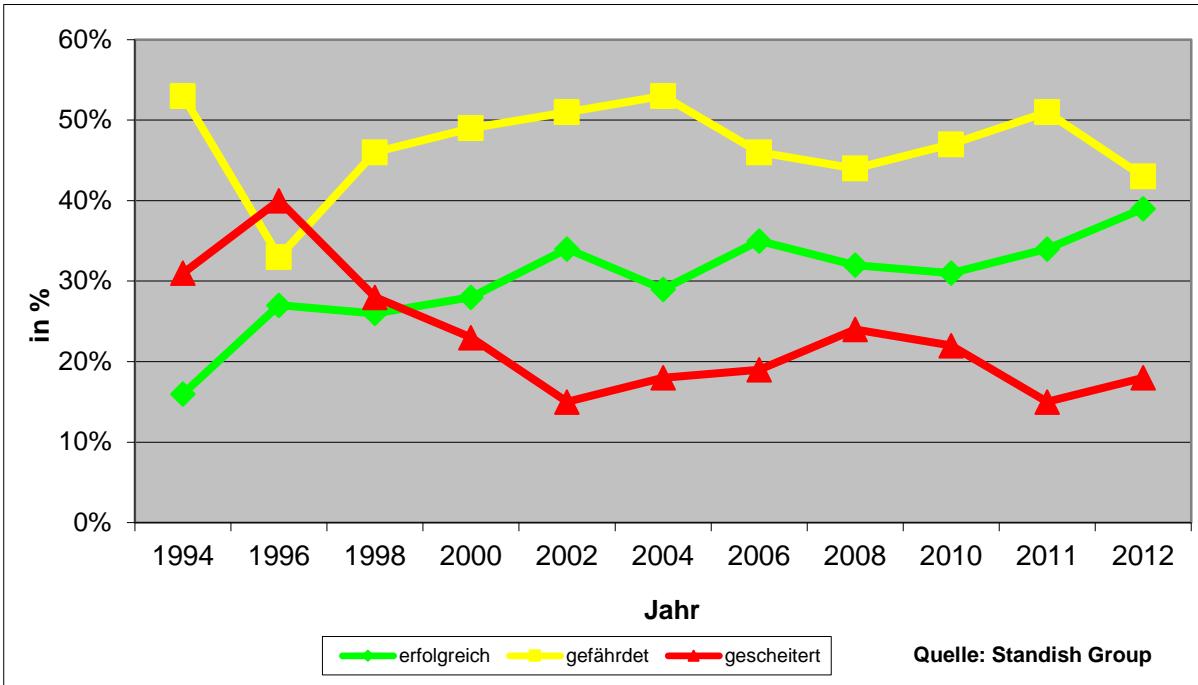
## Schlechtes Management

vgl. „Chaos-Report“ der Standish Group von 1995\*:



\*) Quelle: <http://www.scs.carleton.ca/~beau/PM/Standish-Report.html>

# Verbesserung?



Das Projekt ist:

- Erfolgreich:* rechtzeitig, Ressourcen eingehalten, voller Umfang
- Gefährdet:* lauffähig, aber mindestens einer der drei Parameter nicht eingehalten
- Gescheitert:* vor der Vollendung eingestellt

	1994	1996	1998	2000	2002	2004	2006	2008	2010	2011	2012
erfolgreich	16%	27%	26%	28%	34%	29%	35%	32%	31%	34%	39%
gefährdet	53%	33%	46%	49%	51%	53%	46%	44%	47%	51%	43%
gescheitert	31%	40%	28%	23%	15%	18%	19%	24%	22%	15%	18%

Quelle: Chaos-Report, Standish Group

# Chaos Report mit neuer Methode



		MODERN RESOLUTION FOR ALL PROJECTS				
		2011	2012	2013	2014	2015
SUCCESSFUL		29%	27%	31%	28%	29%
CHALLENGED		49%	56%	50%	55%	52%
FAILED		22%	17%	19%	17%	19%

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%

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%
<b>TOTAL</b>	<b>100%</b>	<b>100%</b>	<b>100%</b>

# Was ist ein Vorgehensmodell?

---

- Reihenfolge des Arbeitsablaufs (Entwicklungsstufen, Phasenkonzepte)
- Strukturierung des Software-Designs
- Definiert
  - \* Rahmenwerk und feste Meilensteine
  - \* Reihenfolge und Teilprodukte der Aktivitäten
  - \* Kriterien für die Abnahme der Produkte (Fertigstellungskriterien)
  - \* Verantwortlichkeiten und Kompetenzen
  - \* notwendige Mitarbeiterqualifikationen
  - \* anzuwendende Standards, Richtlinien und Werkzeuge



***Besseres Projektmanagement möglich!***

# Warum Vorgehensmodelle?

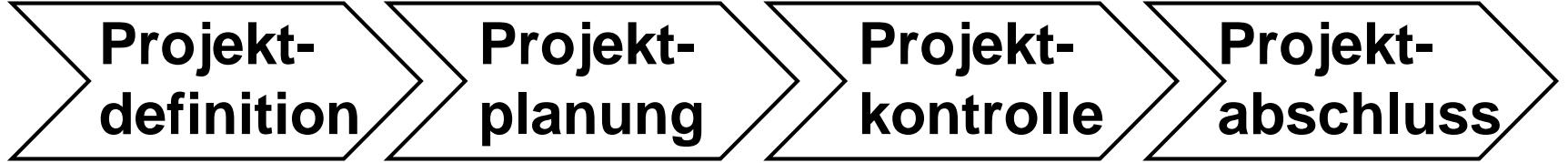
---

- Vorhersehbare, überschaubare, planbare und kontrollierbare Gestaltung der Software-Entwicklung
- Optimierung des Entwicklungsprozesses
- Zertifizierbarkeit des Entwicklungsprozesses
- Erhöhung der Prozessqualität



***Beherrschung des komplexen Prozesses  
der Software-Entwicklung***

# Traditionelle Projektphasen

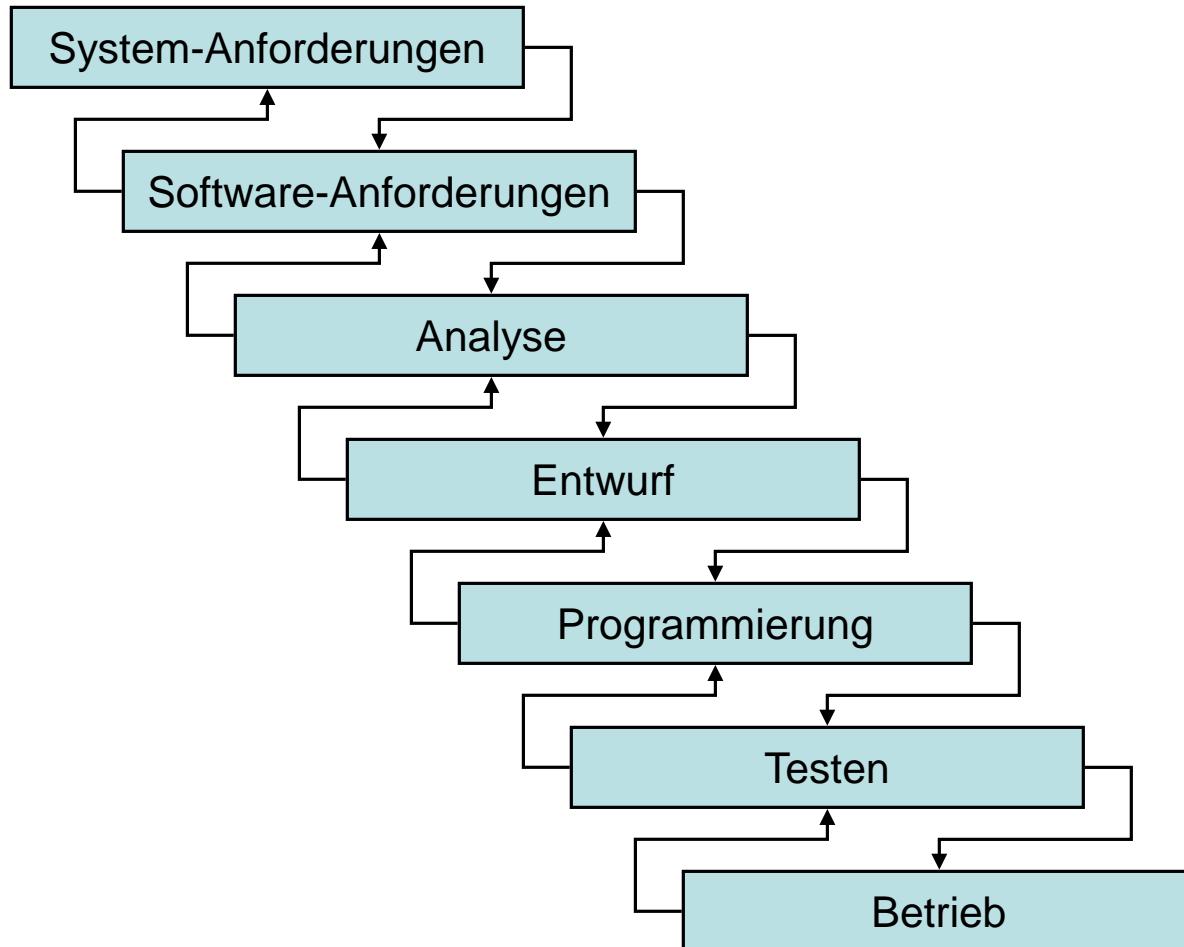


## ⌚ Klassische Modelle:

- Das Wasserfall-Modell
- Das V-Modell
- Das Prototypen-Modell
- Das Spiral-Modell

# Das Wasserfall-Modell

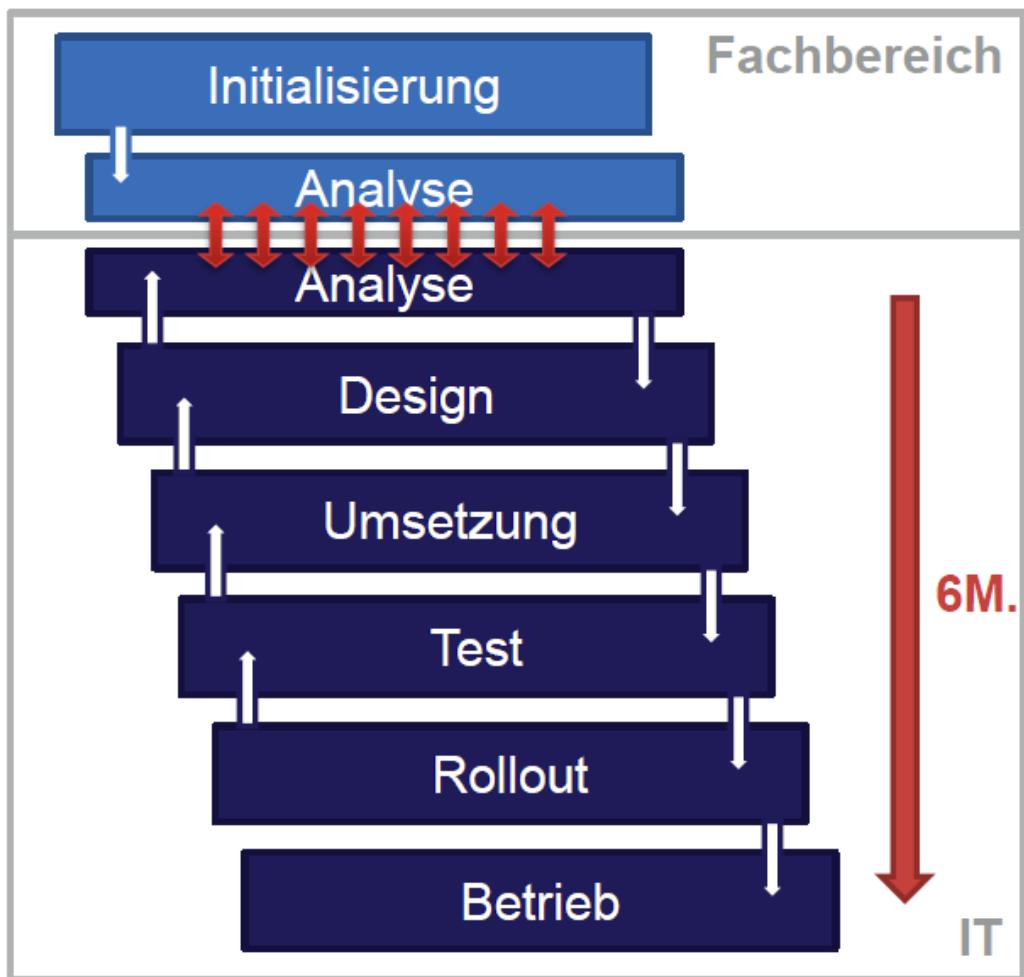
## Phasen:



## Ergebnisse:



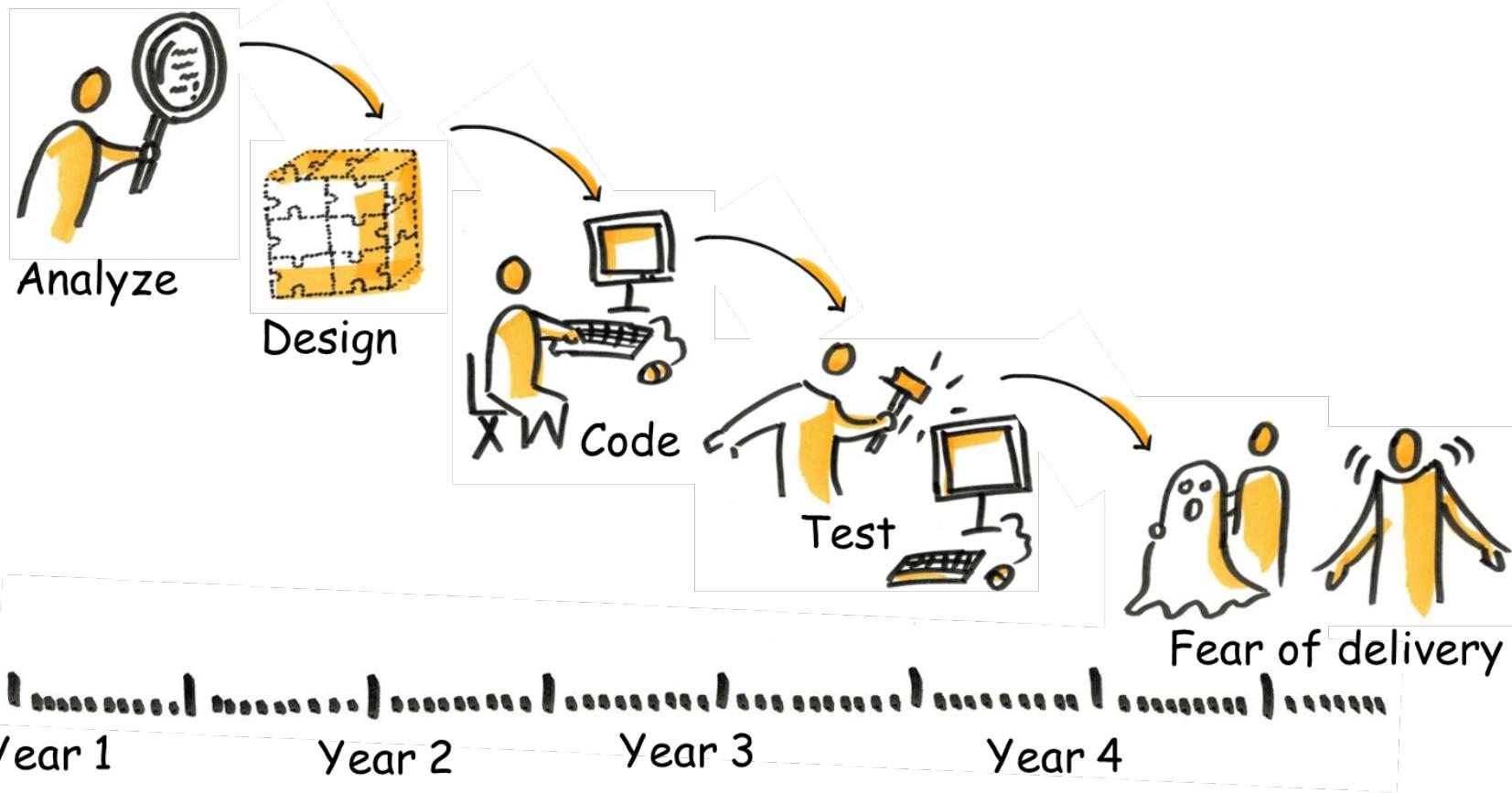
# Fehlende Kommunikation



- **Annahme:**
  - FB = Kunde
  - IT = Lieferant
- **IT bildet immer einen Bottleneck**
  - egal wie effektiv/effizient aufgestellt
- **Sichtbare Effekte**
  - endlose Projekte
  - Lange Wartezeiten auch für kl. Änderungen

# The “Waterfall” – a Bureaucratic Approach

Easy to plan and manage, but hardly ever works as planned



# Beschreibung: Wasserfall-Modell

---

- Durchführung jeder Aktivität in der richtigen Reihenfolge und in vollem Umfang
- Abgeschlossene Dokumentation am Ende jeder Aktivität
- Sequentieller Entwicklungsablauf
- Top-Down-Vorgehen
- Iterationen nur zwischen zwei aufeinanderfolgenden Stufen

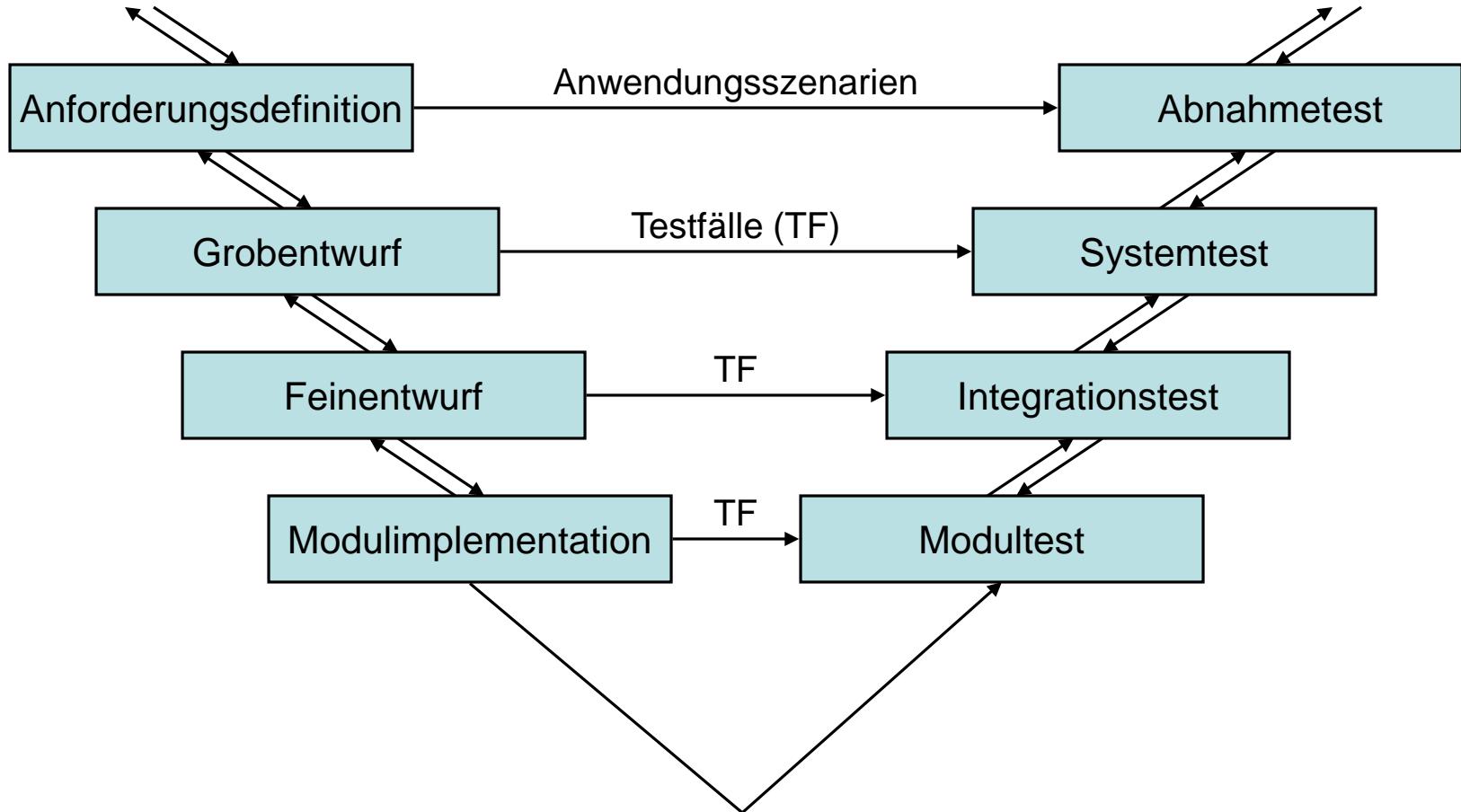
# Bewertung: Wasserfall-Modell

- + Extrem einfaches Modell
- + Geringer Management-Aufwand
- + Disziplinierter, kontrollierbarer und sichtbarer Prozessablauf
- Strenge Sequentialität oft nicht sinnvoll/machbar
- Möglichkeit von Feedback kaum gegeben
- Keine Möglichkeit für Risiko-Management
- Erkennen von Problemen erst am Ende
- Benutzerbeteiligung nur bei Anforderungen und im Betrieb
- Gefahr einer zu starken Gewichtung der Dokumentation



***Dennoch sehr beliebt bei Managern und noch weit verbreitet!***

# Das V-Modell



# Beschreibung: V-Modell



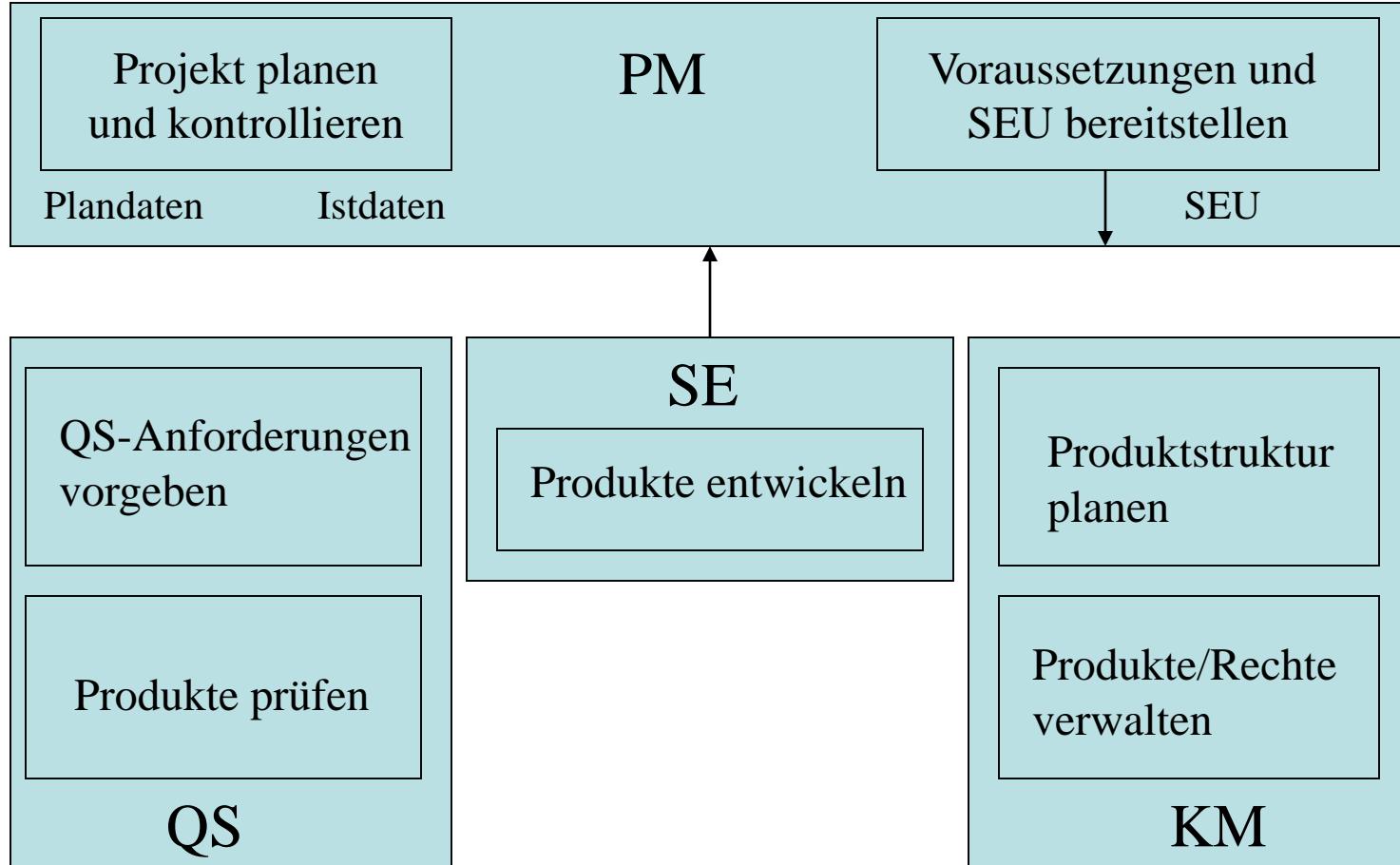
- Anspruch auf Allgemeingültigkeit
- Definition von 25 Rollen für Managementaufgaben
- Anpassung an konkrete Anforderungen
- Entwicklungsmodell für ein Gesamtsystem
- ISO-Standard: Militär- und Bundesbehörden
- Aktuelle Version: V-Modell XT 2.2 (2018; XT = extreme tailoring)
- Aufteilung in Sub-Modelle:
  - \* Systemerstellung
  - \* Qualitätssicherung
  - \* Konfigurationsmanagement
  - \* Projektmanagement

# Ziele des V-Modells

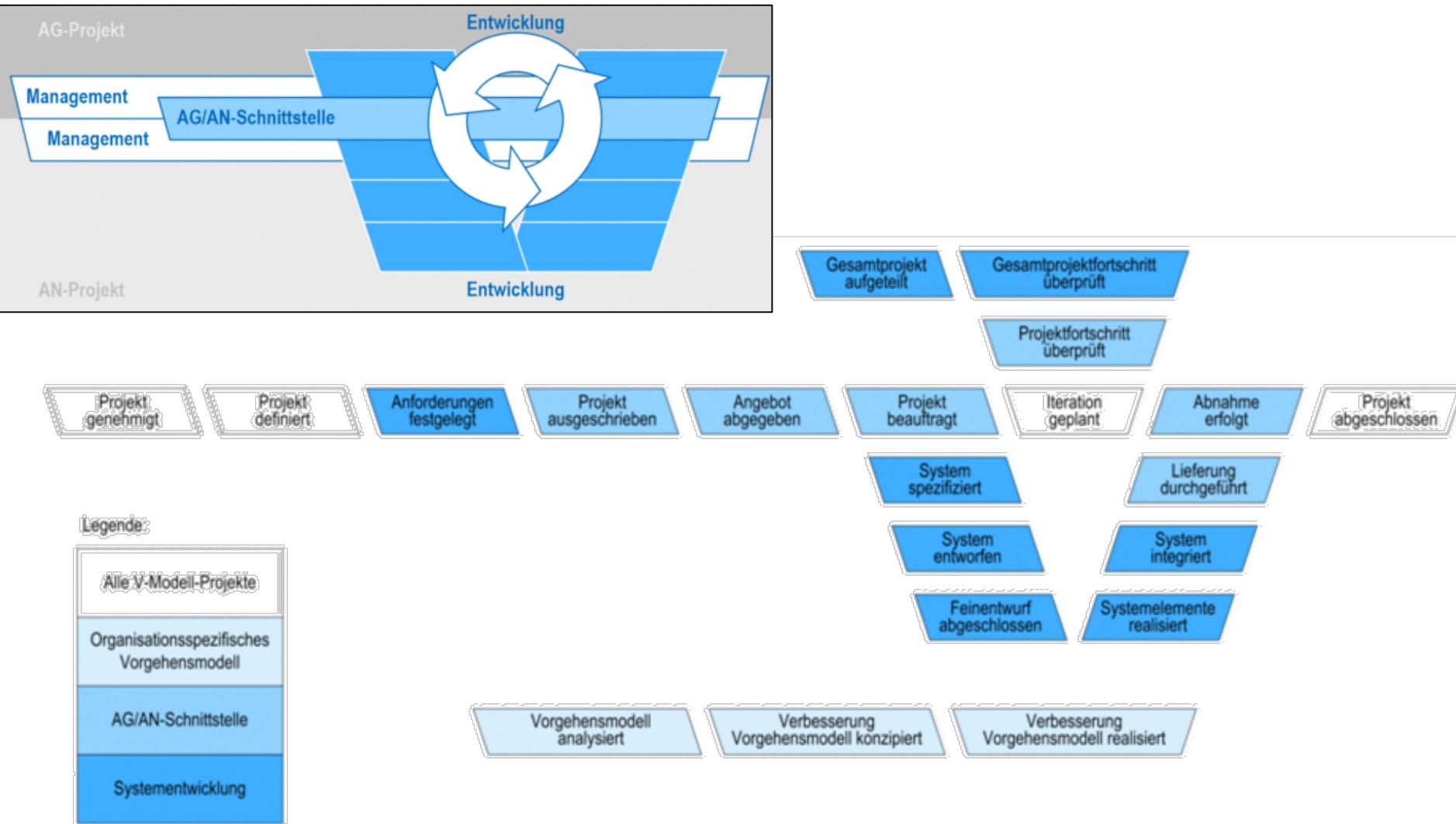
---

- Das V-Modell ist ein Leitfaden zum Planen und Durchführen von Projekten.
- Mit dem standardisierten Vorgehen werden folgende Ziele verfolgt:
  - Minimierung der Projektrisiken durch Verbesserung und Gewährleistung der Qualität
  - Eindämmung der Gesamtkosten über den gesamten Projekt- und Systemlebenszyklus
  - Verbesserung der Kommunikation zwischen allen Beteiligten

# Aufbau des V-Modells



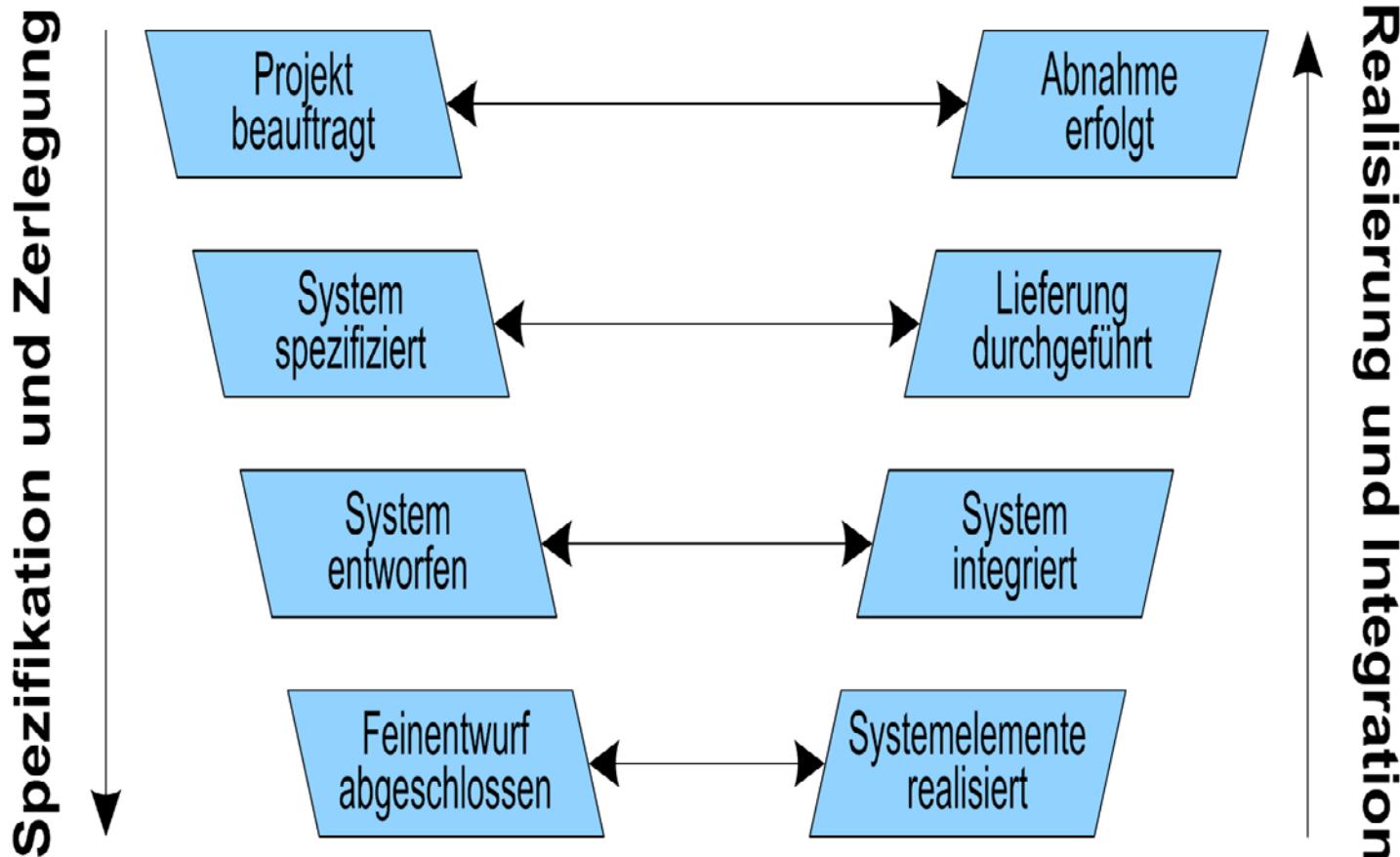
# Überblick: V-Modell XT



# Systemerstellung im Überblick



## Verifizierung und Validierung

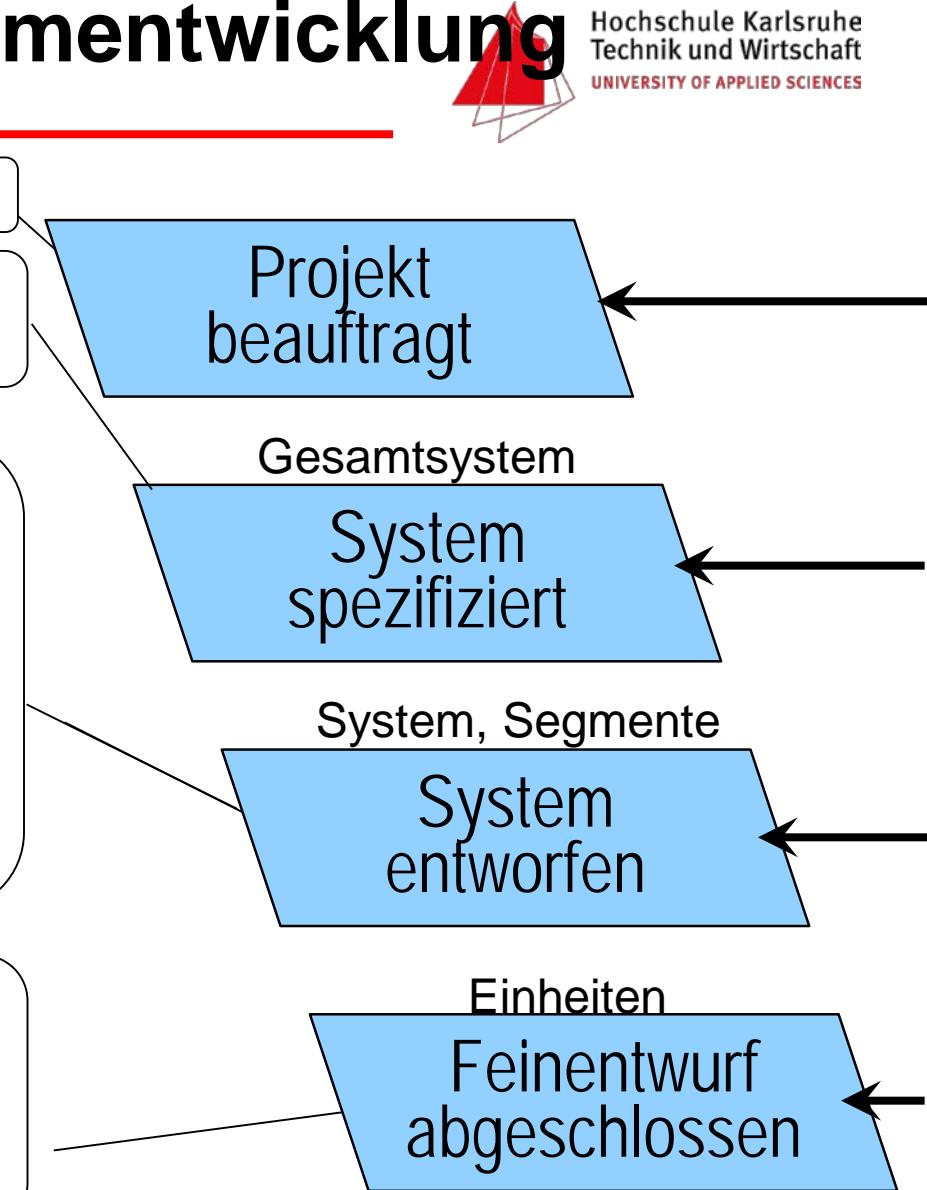


# Zerlegung in der Systementwicklung

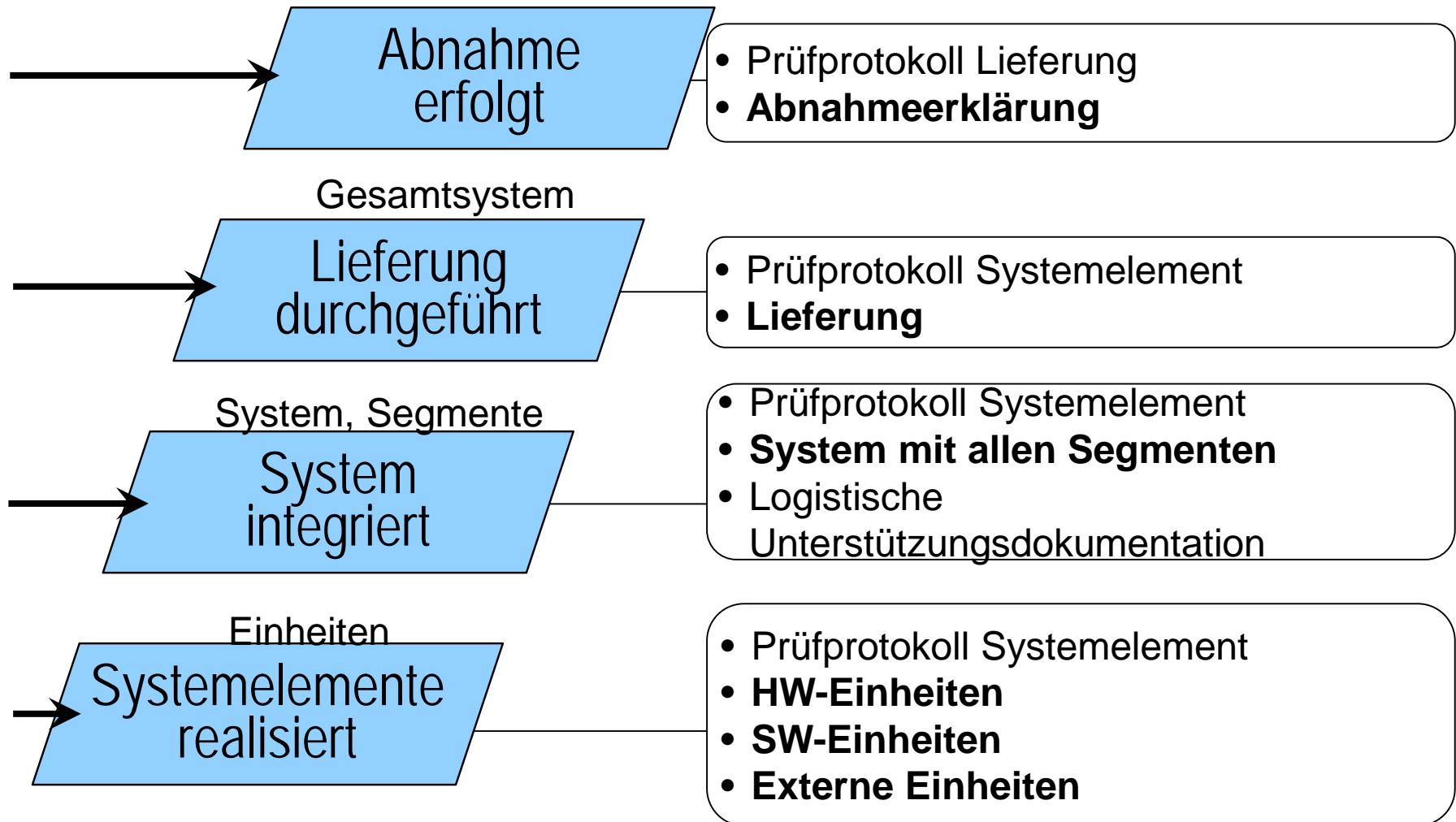
- Anforderungen (Lastenheft)
- Gesamtsystemspezifikation (Pflichtenheft)
- Gefährdungs- und Systemsicherheitsanalyse

- Systemarchitektur
- Unterstützungssystemarchitektur
- Systemspezifikation
- Spezifikation log. Unterstützung
- Prüfspezifikation Systemelement
- Implementierungs-, Integrations- und Prüfkonzept System/Unterstützungssystem
- Prüfspezifikation Systemelement

- HW-Architektur und SW-Architektur
- HW-Spezifikation und SW-Spezifikation
- Logistisches Unterstützungskonzept
- Externe-Einheit-Spezifikation



# Entscheidungspunkte - Produkte Systementwicklung



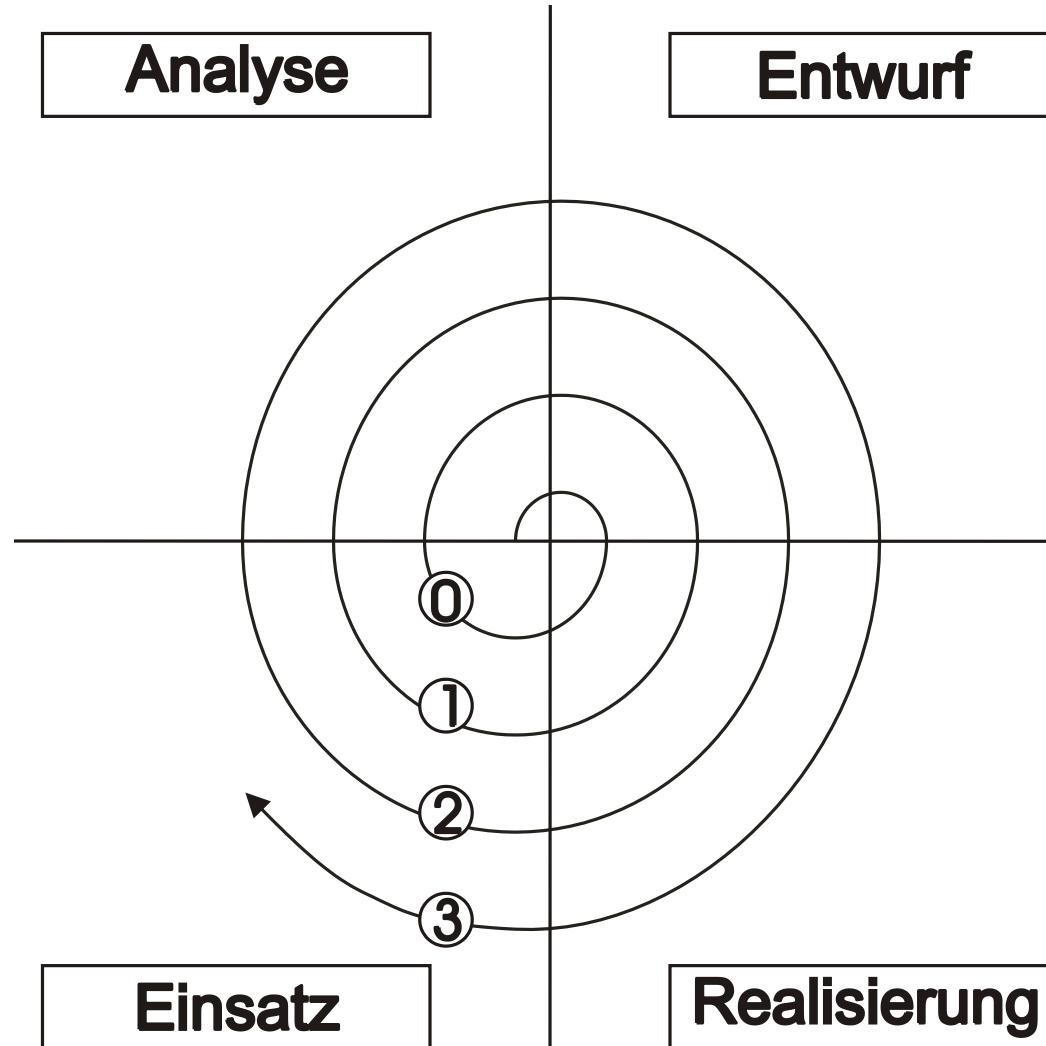
# Bewertung: V-Modell

- + Sehr detaillierte Darstellung
- + Anpassung an projektspezifische Anforderungen
- + Integration vieler Aspekte des Entwicklungsprozesses
- + Standardisierung der Abwicklung von Systemerstellungsprojekten
- Zu allgemein für kleine und mittlere Software-Modelle
- Späte Tests
- Sehr bürokratisch

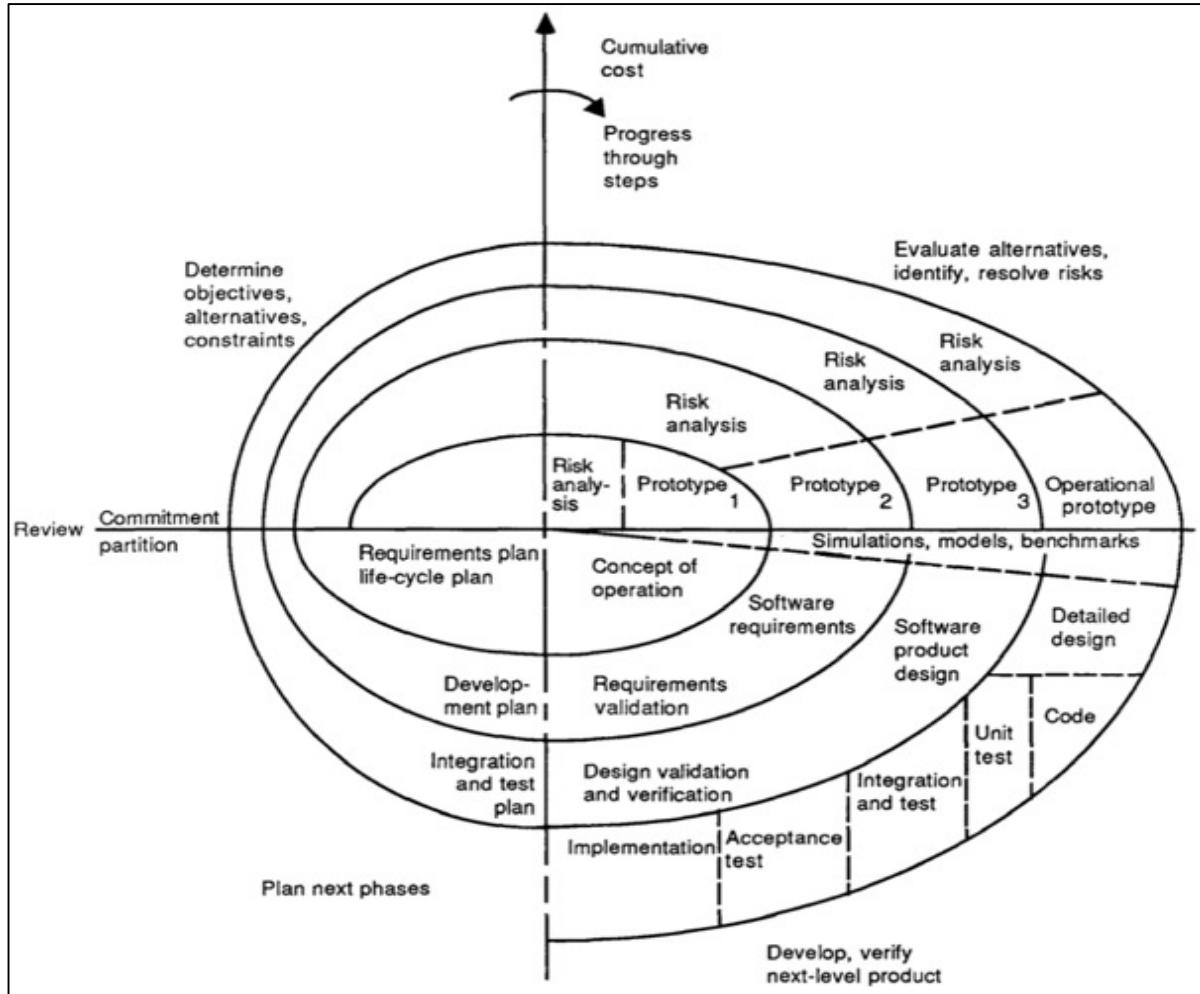


***Gut geeignet für große Projekte!***

# Das Spiral-Modell



# Das Spiralmodell



Quelle: Boehm (1988)

# Beschreibung: Spiral-Modell

---

- Meta-Modell
- Risikominimierung als oberstes Ziel
- Durchlaufen von vier zyklischen Schritten für jede Verfeinerungsebene und jedes Teilprodukt
- Ergebnisse des letzten Zyklus → Ziele des nächsten Zyklus
- Bei Bedarf separate Spiralzyklen für verschiedene Komponenten

# Bewertung: Spiral-Modell

- + Regelmäßige Risiko-überprüfung des Prozessablaufs
- + Keine Festlegung auf ein Prozessmodell
- + Frühzeitige Eliminierung von ungeeigneten Alternativen und Fehlern
- Hoher Managementaufwand (für kleine und mittlere Projekte weniger gut geeignet)



***Sehr flexibles Modell durch Betrachtung von Alternativen!***

## ⌚ Neue Modelle:

- ❑ Iterativ-Inkrementelle und agile Methoden

# Agile Methoden in der Softwareentwicklung im Überblick

- Lean development (LD; auch als LSD zu finden)
- Adaptive software development (ASD)
- Feature-driven development (FDD)
- eXtreme Programming (XP)
- Scrum
- Kanban

---

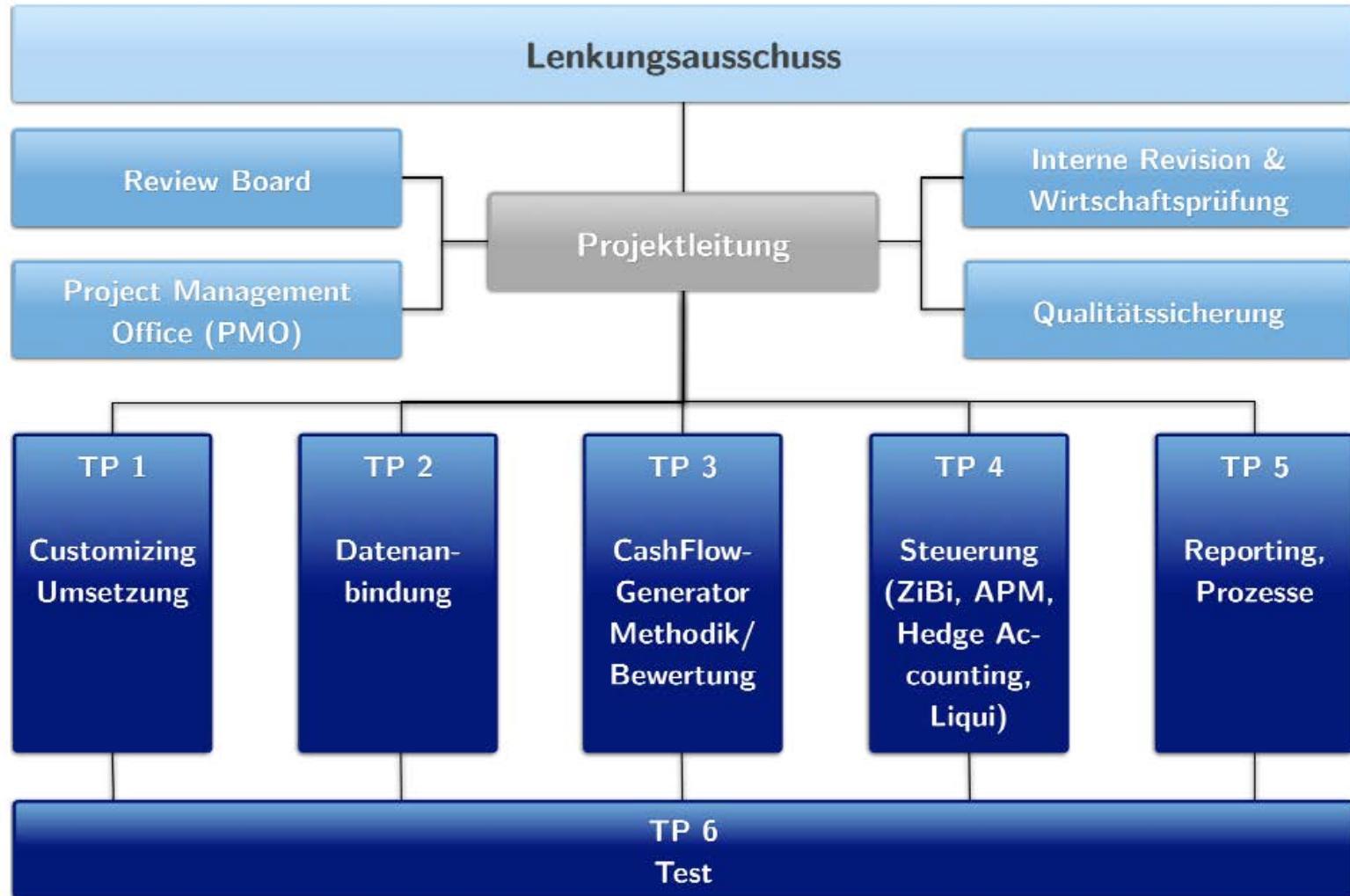
# Hierzu später mehr!

# Beteiligte in einem (klassischen?) Projekt

---

- Wer sind die Beteiligten in einem Projekt und wie sind sie vertreten?
  - Projektauftraggeber
  - Projektleiter
  - Teilprojektleiter
  - Lenkungsausschuss

# Beispiel für eine Projektorganisation



**Der Lenkungsausschuss ist das zentrale Steuerungsgremium in einem Projekt.**

## Aufgaben:

- Entscheidet über Projektaufträge/-erweiterungen
- Setzt Rahmenbedingungen für die Projektdurchführung
- Ernennt die Projektleitung
- Entscheidet über die Ressourcen- und Budgetverteilung
- Entscheidet über inhaltliche und budgettechnische Änderungen
- Setzt übergeordnete Unternehmensinteressen durch

## Mitglieder des Lenkungsausschusses:

- Es sollte sich um ein kleines und handlungsfähiges Gremium handeln.
- Vertreter des Auftraggebers (mit Entscheidungskompetenz)
- Vertreter des/der Auftragnehmer(s) (Umsetzungsverantwortliche)
- Projektleitung
- Beteiligte mit Sonderfunktionen (z.B. Mitarbeitervertreter, Datenschutzbeauftragte etc.)

# Die Projektleitung

## Projektgremium: Projektleitung

Aufgaben	Besetzung
<ul style="list-style-type: none"><li>• Gesamtprojektplanung und -steuerung</li><li>• Projekt-Controlling</li><li>• Projektberichtswesen</li><li>• Qualitätsmanagement und Integration</li><li>• Projektinitialisierung</li><li>• Steuerung und Koordination der fachlichen Projektteams</li><li>• Planung und Steuerung teilprojektübergreifender Aufgaben</li><li>• Durchführung und Erklärung der fachlichen Abnahmen</li><li>• Erarbeitung von Entscheidungsvorlagen für den Lenkungsausschuss</li></ul>	<ul style="list-style-type: none"><li>• Gesamtprojektleitung AN</li><li>• Fachliche Projektleitungen AG / AN</li><li>• Projektleiter AG</li><li>• Projektleiter Koordinierungsgruppe für Projekte bei AG</li><li>• Verantwortlicher AG für Qualitätsmanagement</li><li>• Schulungskoordinator</li></ul>

## Aufgabe:

„Hat die Abwicklung des ihm zugeordneten Auftrags in Zusammenarbeit mit dem Projektteam und den vom Projekt betroffenen Organisationseinheiten so zu leiten, dass die

- auftragskonforme,
- termin- und
- kostengerechte

Projektabwicklung gewährleistet ist.“

**Leichter gesagt, als getan....**

# Der Projektleiter - Funktionen

---

- Planen
  - des Gesamtprojekts
  - seiner Phasen
- Kontrolle des Projektverlaufs
- Koordination der am Projekt beteiligten Personen
- Führen, Anleiten und Motivieren der Teammitglieder
- Informieren der projektbeteiligten oder betroffenen Organisationseinheiten
- Aufbereitung von Entscheidungsvorlagen und Herbeiführen von Entscheidungen
- Vertretung des Projektes (und seiner Mitarbeiter) in übergeordneten Gremien
- Berichtswesen für AG & AN!

# Der Projektleiter – Weitere Aspekte

---

- Wird die Kontrollspanne zu groß (Grenze bei ca. 6-8 Mitarbeitern), sollten Teilprojektteams gegründet werden.
- Neues Gremium: Teilprojektleiterrunde
  - Abstimmung und Koordination der einzelnen Teilprojektaktivitäten
  - Informationsfluss nach oben ist zu gewährleisten

# Die fachlichen Projektteams (Teilprojekte)

## Projektgremium: Fachliche Projektteams

Aufgaben	Besetzung
<ul style="list-style-type: none"><li>• Fachliche Projektarbeit</li><li>• Ausführung von Aufträgen der Projektleitung</li><li>• Erklärung der Betriebsbereitschaft (=Bereitstellung zur Abnahme durch AG)</li><li>• Antragstellung von Change Requests an die Projektleitung</li></ul>	Teilprojektleiter AN Teilprojektleiter AG Teilprojektmitarbeiter AN Key-user AG

# Der Teilprojektleiter

- Übernimmt fachliche Verantwortung für einen Teilbereich
- Stimmt Aufwand und Leistungen seines Teams mit der Projektleitung ab
- Informationskanal der fachlichen, technischen oder auch zwischenmenschlichen Probleme im Teilprojektteam
- Eingeschränkte Entscheidungskompetenz
- Bei Großprojekten mit vielen Teilprojekten besteht die Kunst darin, die Fäden zusammen zu halten und im richtigen Moment zu verknüpfen ⇒ Aufgabe des PL, nicht des TPL

# Auftraggeber

---

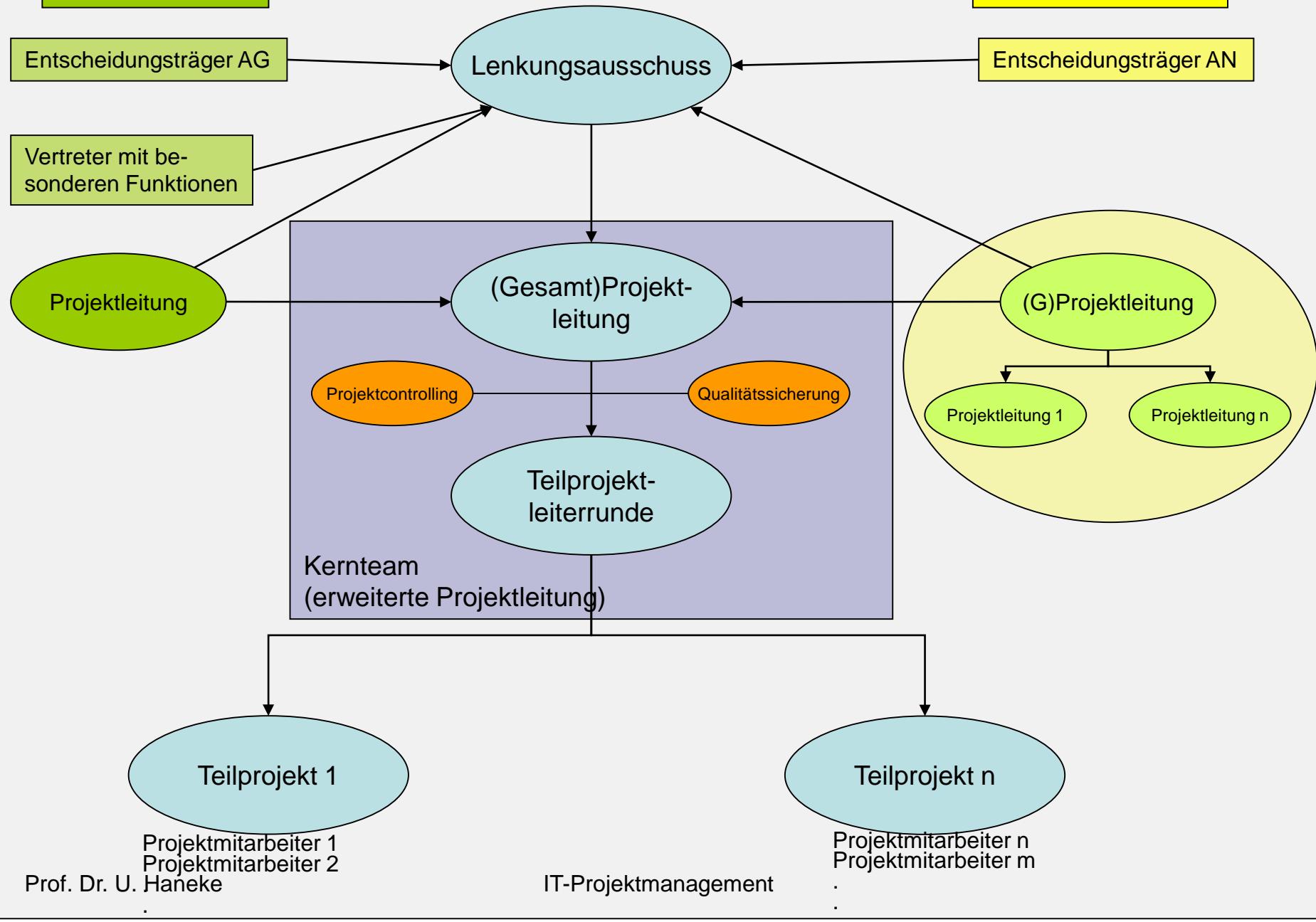
- Kann intern oder extern sein
- Seine Ziele müssen klar festgelegt sein
- Kein vernünftiges Projekt ohne ausreichende Ressourcen
- Lehnen sie ansonsten eine Projektleitung lieber ab (oder springen sie auch in halbleere Schwimmbecken???)!

## Auftraggeber

Entscheidungsträger AG

## Auftragnehmer

Entscheidungsträger AN



# Probleme aus der Praxis

---

- (Teil-)Projektleitung macht man doch nebenbei...
- Projektleitung zu stark in das Tagesgeschäft eingebunden
- Aus Wettbewerbsgründen zu wenig Ressourcen angesetzt („... sonst wäre das Projekt abgelehnt worden!“)
- Keine Konzepte für QM, Schulung und Dokumentation: Das holt jedes Projekt definitiv ein!
- Ziele nicht sauber definiert
- Projektteilnehmer AG im Vorfeld nicht ausreichend geschult (Skills Management; Personalentwicklung)

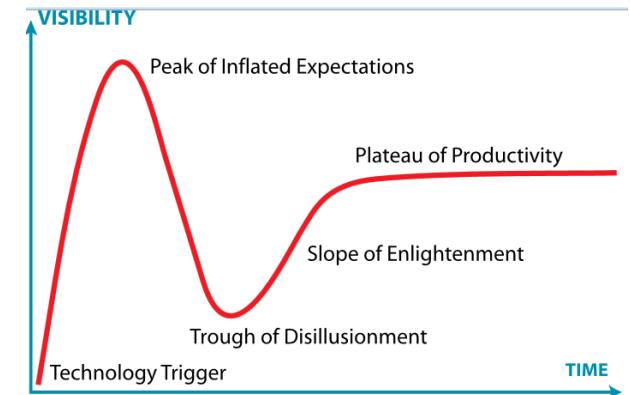
# Kurz zur Teambildung (I)

---

- Ein Team sollte aus nicht mehr als 7-8 Mitgliedern bestehen.
- Sind mehr als 8 Mitglieder beteiligt, lohnt es sich
  - das Team zu teilen oder
  - ein Kernteam zu bilden (kleine Führungsgruppe, die die übrigen Mitglieder bei Entscheidungen fallweise hinzu zieht)
- Sachkompetenz allein ist nicht ausreichend!
- Für spezielle Fragen können ad-hoc-Teams gebildet werden.
- Die richtige Mischung ist der Weg zum Erfolg: Zu einseitig besetzte Teams sind der Tod jeder neuen Idee!
- Teams ändern sich und müssen zum Teil auch geändert werden
  - Fachliche Kompetenz fehlt
  - Soziale Kompetenz fehlt
  - A kann nicht mit B: wer wird entfernt?

# Kurz zur Teambildung (II)

- Teamarbeit erfolgt in Phasen:
  - Euphorie des Projektstarts
  - Ernüchterung angesichts auftretender Probleme
  - Cliquenbildung, Frust und Konflikte: „Es klappt ja gar nichts!“
  - „Es geht ja doch!“ und „Das hätten wir uns vorher sagen müssen!“
  - 24/7: Der Produktivstart
  - Ruhephase: Nicht sofort weiter machen!
- Remember: Bei Großprojekten mit vielen Teilprojekten besteht die Kunst darin, die Fäden zusammen zu halten und im richtigen Moment zu verknüpfen. ⇒ Stressfaktor



# Von der Notwendigkeit der begleitenden Maßnahmen

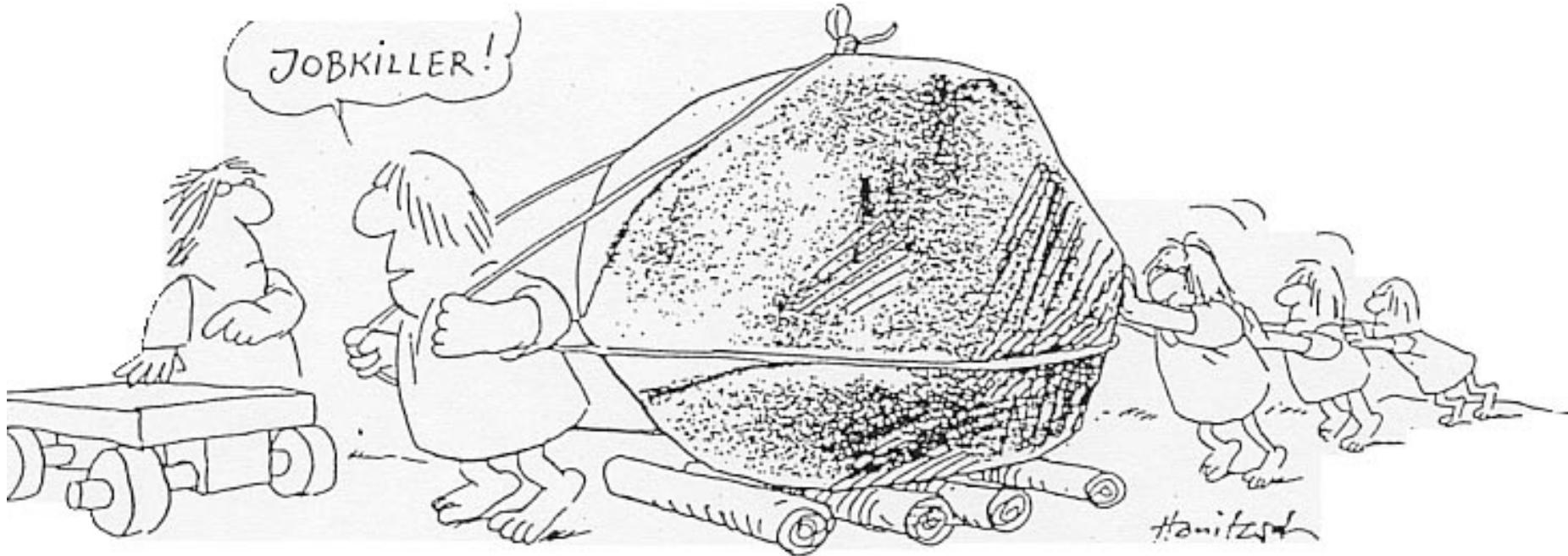
---

Häuptlinge haben ein Problem, wenn die Indianer keine Lust haben!

## Change Management:

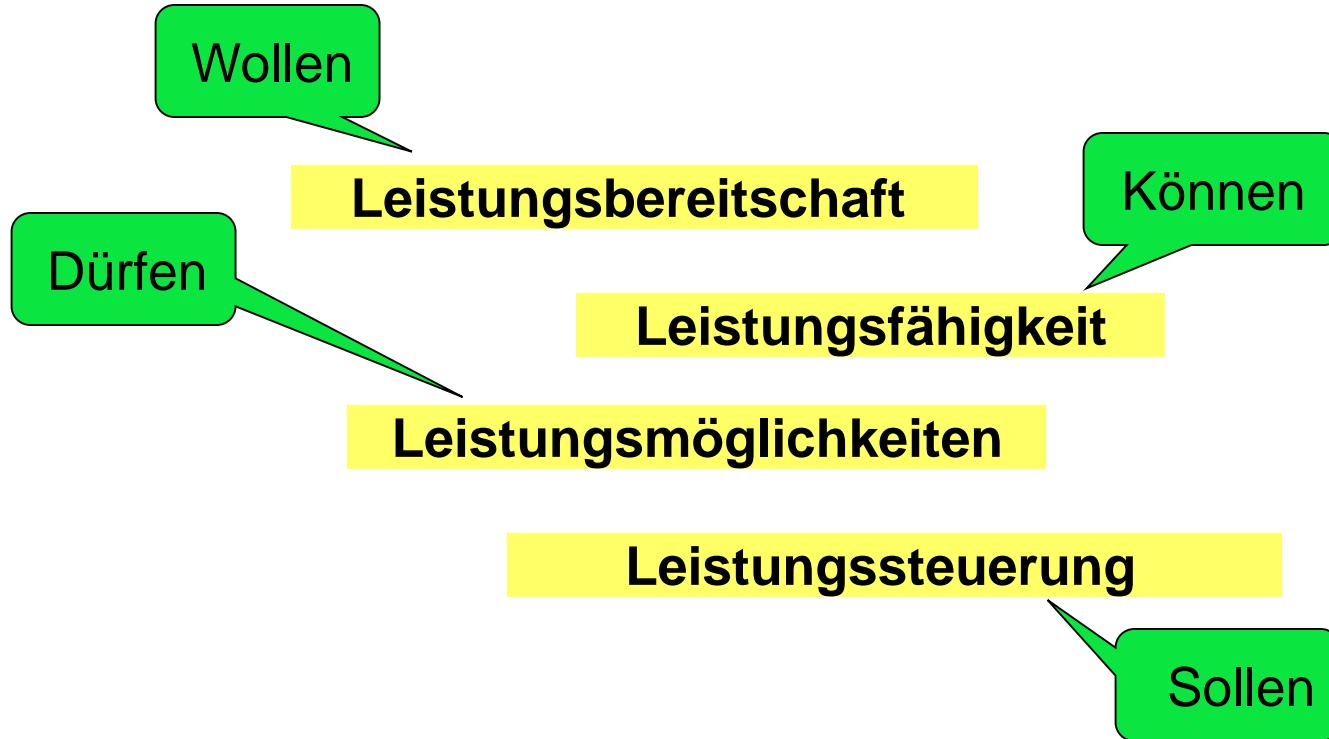
“... ist eine unterstützende Querschnittsaufgabe, die Veränderungen der Prozesse, Strukturen und Technologien unterstützt, damit der Erfolgsfaktor “Mitarbeiter” seine neuen Aufgaben annimmt und in der Lage ist, sie umzusetzen.”

# Change Management



Ist ihr Projekt über dieses Stadium schon hinweg?

# Die wichtigste Aufgabe: Menschen führen und motivieren!



# Lasten- und Pflichtenhefte I

## (Die inhaltliche Klärung des Projektes)



- Das Lastenheft: WAS und WOFÜR
  - Zusammenstellung aller Anforderungen des AG und des oder der Nutzer an das Projekt hinsichtlich
    - der Ziele
    - des Liefer- und Leistungsumfangs
    - der Randbedingungen
  - Beschreibt WAS zu erbringen ist und WOFÜR
  - Manchmal auch Wunschliste...
  - Wird vom AG (oder in dessen Auftrag!) erstellt

# Lastenheft: Millionenschäden durch handwerkliche Fehler

---

„Die Bundeswehr prüfte daraufhin, ob sie den Auftrag abbrechen konnte. Sie hielt eine Rückabwicklung dem Bericht zufolge aber für aussichtslos, **„weil die Anforderungen an die Software nur unzureichend beschrieben seien und Interpretationsspielräume bei der Umsetzung zuließen.“** Die Bundeswehr führte das Projekt daher weiter. Es ist nicht das erste Mal, dass der Rechnungshof dem deutschen Militär schlechtes Projektmanagement vorwirft.“ (2018)

# Lasten- und Pflichtenhefte II

## (Die inhaltliche Klärung des Projektes)

**Die Gliederung des Lastenhefts sollte folgende Punkte enthalten:**

- Die Spezifikation des zu erstellenden Produkts (die "Last")
- Anforderungen an das Produkt bei seiner späteren Verwendung (z.B. Webfähigkeit)
- Rahmenbedingungen für Produkt und Leistungserbringung (z.B. Normen, Richtlinien, Materialien usw.)
- Vertragliche Konditionen (z.B. Erbringen von Teilleistungen, Gewährleistungsanforderungen, Risikomanagement usw.)
- Anforderungen an den Auftragnehmer (z.B. Zertifizierungen)
- Anforderungen an das Projektmanagement des Auftragnehmers (z.B. Projektdokumentation, Controlling-Methoden)

**Beispiel**

# Lasten- und Pflichtenhefte III

## (Die inhaltliche Klärung des Projektes)

- Das Pflichtenheft: WIE und WOMIT
  - Weiterentwicklung des Lastenheftes
  - Beschreibt die Realisierung aller Anforderungen des Lastenheftes (Detaillierung des Lastenheftes)
  - Definiert WIE und WOMIT die Anforderungen realisiert werden
  - Oftmals verbindliche Vereinbarung zwischen Auftraggeber und Auftragnehmer (Vertragsgegenstand)
  - Manchmal werden Pflichtenhefte auch in der Form von Fachkonzepten erstellt
  - Werden oftmals nicht von denjenigen erstellt, die es auch umsetzen müssen

**Beispiel 1a**

**Beispiel 1b**

# Lasten- und Pflichtenhefte IV

## (Die inhaltliche Klärung des Projektes)



### Resümee:

Während das Lastenheft im Wesentlichen die Spezifikation des Produkts, die Rahmenbedingungen und evtl. noch den Produktstrukturplan enthält, beschreibt das Pflichtenheft in erster Linie, **wie** der Auftragnehmer die Leistung zu erbringen gedenkt.

# Pflichtenheft in der SW-Entwicklung I

---

- Pflichtenheft als organisatorische und technische Vorgabe zur Erstellung von Software
- Bei der Softwareentwicklung wird die Erstellung eines qualifizierten Pflichtenheftes oftmals mit geringer Priorität behandelt
- Ein Entwickler beginnt meist sehr früh mit der Produktentwicklung (schnelle Lösungen)
- Teufelskreis der Softwareentwicklung: Termine müssen angegeben werden, ohne dass der genaue Leistungsumfang definiert ist

Quelle: Hilgenberg, B., (2000): Das Pflichtenheft in der Softwareentwicklung. In: Projekt Magazin – Das interaktive Online-Magazin für Projektmanagement, [www.projektmagazin.de](http://www.projektmagazin.de), 03/2000.

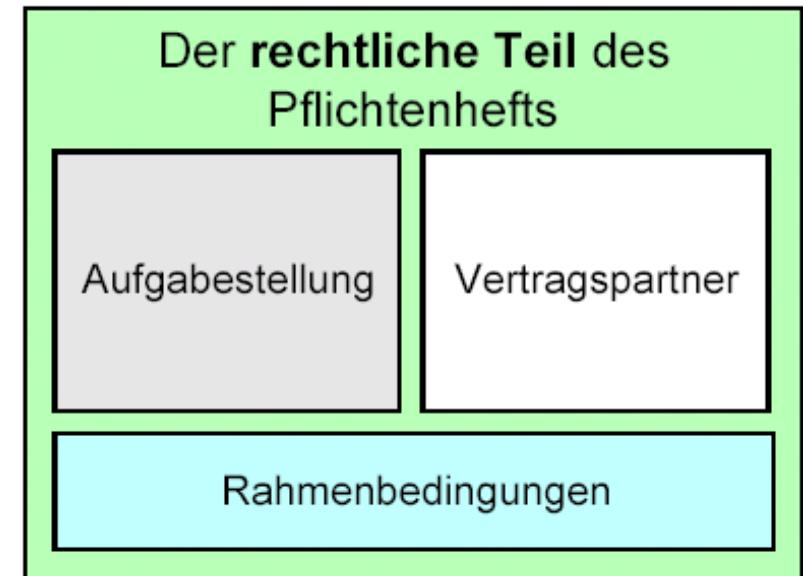
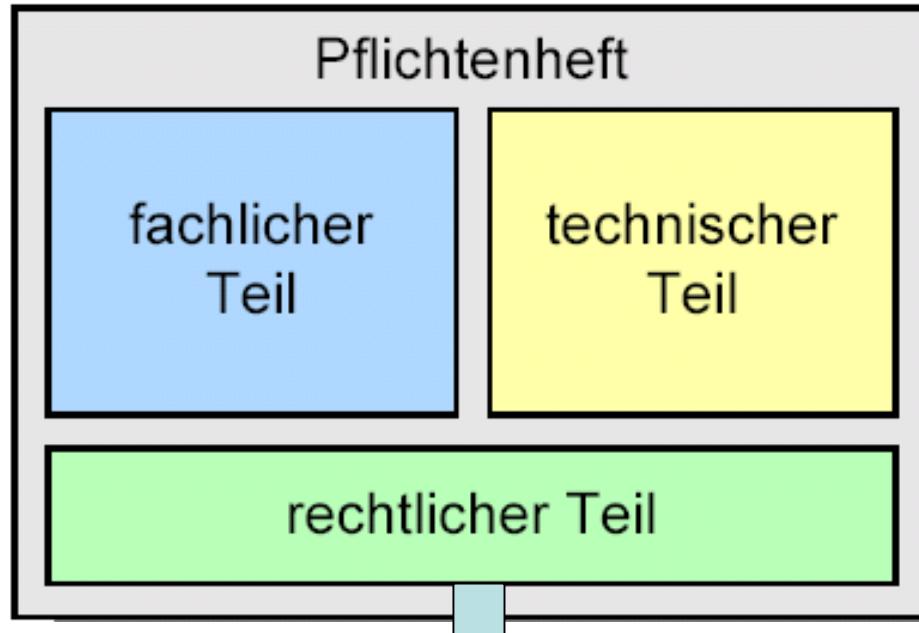
# Pflichtenheft in der SW-Entwicklung II



Der ideale erste Schritt: ein separater Auftrag für das Pflichtenheft.

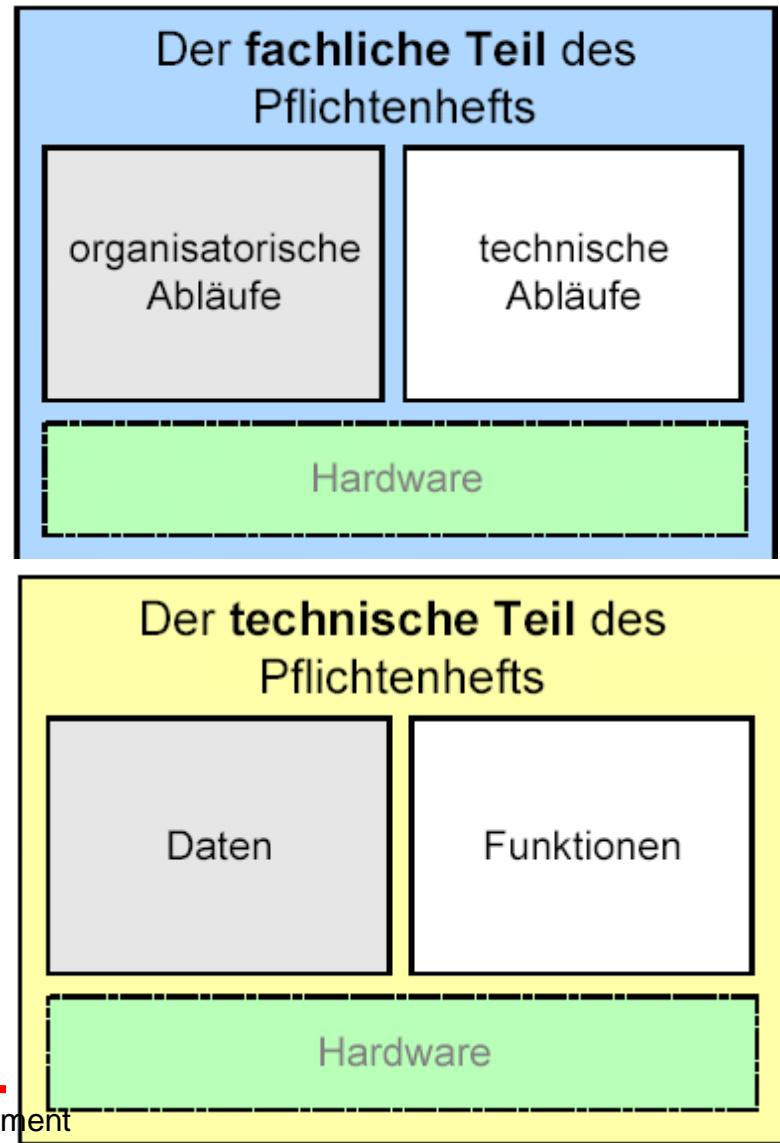
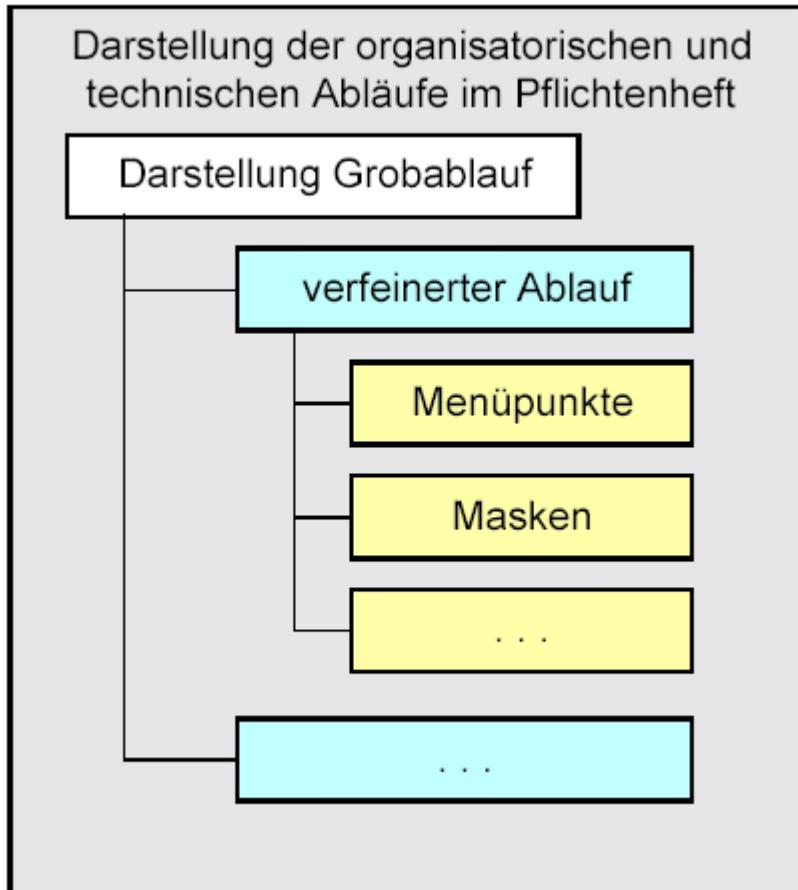
Quelle: Hilgenberg, B., (2000)

# Pflichtenheft in der SW-Entwicklung III



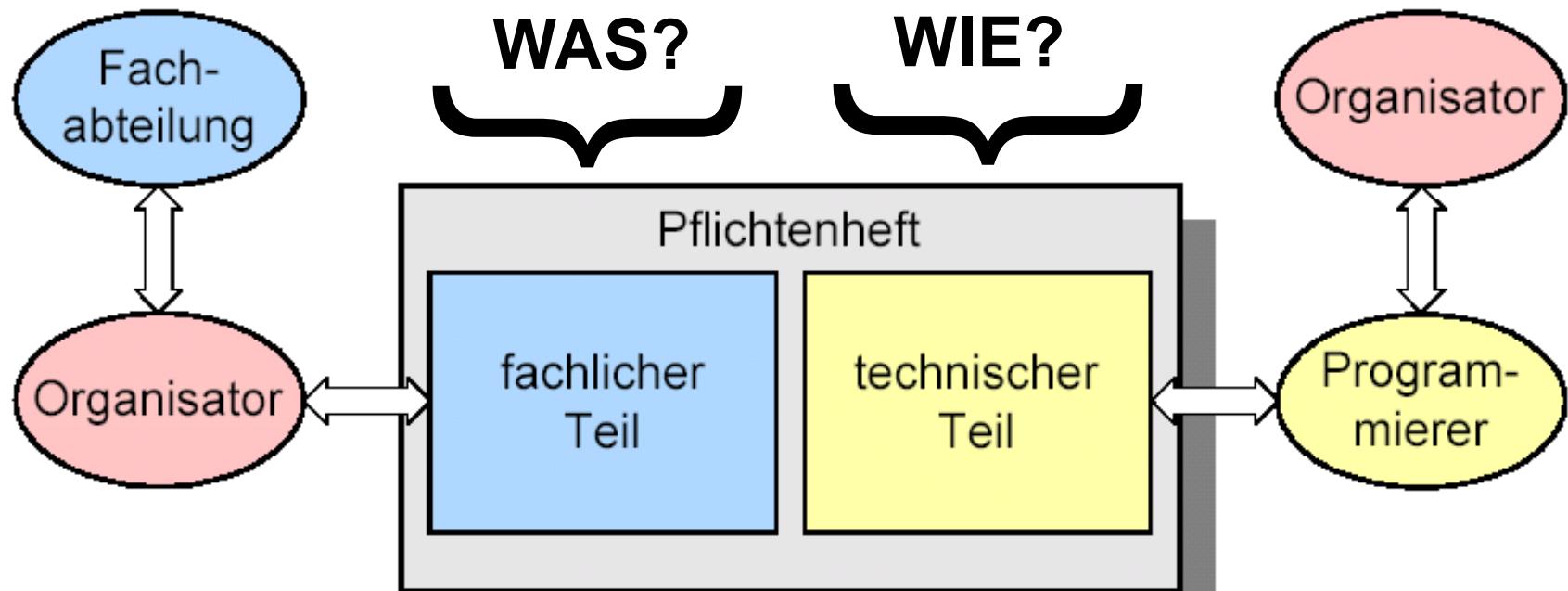
Quelle: Hilgenberg, B., (2000)

# Pflichtenheft in der SW-Entwicklung IV



Quelle: Hilgenberg, B., (2000)

# Pflichtenheft in der SW-Entwicklung V



Quelle: Hilgenberg, B., (2000)

# Planungsmethoden der Terminschätzung

---

- Netzplantechniken

➤ **Netzplan**

Der Netzplan ist die graphische oder tabellarische Darstellung von Abläufen und deren Abhängigkeiten.

➤ **Vorgang:**

Ein Vorgang ist ein Ablaufelement, das ein bestimmtes Geschehen beschreibt. Der Vorgang ist gekennzeichnet durch definierten Anfang und Ende sowie seine Dauer.

➤ **Vorgänger:**

Einem Vorgang unmittelbar vorgeordneter Vorgang.

➤ **Nachfolger:**

Einem Vorgang unmittelbar nachgeordneter Vorgang.

➤ **Ereignis:**

Ein Ereignis ist ein Ablaufelement, dass das Eintreten eines bestimmten Zustandes beschreibt. Ein Ereignis hat selbst keine Dauer.

- **Anordnungsbeziehung (AOB):**  
Eine Anordnungsbeziehung ist eine quantifizierbare Abhängigkeit zwischen Ereignissen oder Vorgängen.
- **Weg:**  
Durch einen oder mehrere Pfeile hergestellte Verbindung zweier Knoten.
- **Puffer:**  
Zeitspanne, um die die Lage eines Vorgangs verändert werden kann, ohne den Projektendzeitpunkt zu verändern.
- **Kritischer Weg:**  
Weg mit dem minimalen Puffer. Vorgangsverschiebungen führen auf dem Kritischen Weg stets zu einer Verschiebung des Projektendzeitpunkts.



- Ereignisknoten-Netzplan (EKN)
- Vorgangspfeil-Netzplan (VPN)
- Vorgangsknoten-Netzplan (VKN)

**Hinweis:** Wir beschäftigen uns hier nur mit Projekten unter **Sicherheit!** Ansonsten sind die Methoden der Entscheidungsnetzplantechnik einzusetzen.

# Ereignisknoten-Netzplan (EKN)

---

- Bekannteste Methode:  
Programme Evaluation and Review  
Technique (PERT)
- Ereignisorientierter Ablaufplan
- Kennzeichen:
  - Enthält nur Ereignisse (Knoten) und AOB (Pfeile),  
keine Vorgänge
  - Zeitlicher Abstand wird auf Pfeilen abgetragen ( $\neq$   
Dauer!)
  - Nur als Meilensteinpläne geeignet

# Vorgangspfeil-Netzplan (VPN)

---

- Bekannteste Methode:  
Critical Path Method (CPM)
- Vorgangsorientierter Ablaufplan
- Kennzeichen:
  - Pfeile stellen Vorgänge und AOB dar
  - Ereignisse werden als Knoten dargestellt
  - Dauer wird auf Pfeilen abgetragen
  - CPM ist v.a. in angelsächsischen stark verbreitet

# Vorgangsknoten-Netzplan (VKN)

---

- Bekannteste Methode:  
Metra-Potential-Method (MPM)
- Vorgangsorientierter Ablaufplan
- Kennzeichen:
  - Vorgänge werden als Knoten dargestellt
  - Ein Knoten kann auch ein Ereignis repräsentieren ( $\Rightarrow$  VKN kann auch als gemischtorientierter Ablaufplan aufgebaut werden!)
  - Pfeile stellen AOB dar
  - MPM wurde in Frankreich entwickelt und beim Reaktorbau eingesetzt

# Aufbau von Netzplänen

---

- Vom Groben zum Feinen
- Von der Gesamtaufgabe zu den Teilaufgaben zu den Arbeitspaketen oder
- Von den Phasen zu den Vorgängen zu den Terminen oder
- Vom PSP zum Ablaufplan zum Netzplan
- Das Arbeiten mit Teilnetzen

# Vorgangslisten

Vorg.-nummer	Vorgangsbeschreibung
1	Grobplanung
2	Feinplanung
3	Beschaffung
4	Fertigung Teil 1
5	Fertigung Teil 2
6	Montage (1+2)
7	Probetrieb
8	Abnahme

# Vorgangslisten

Vorg.-nummer	Vorgangsbeschreibung	Dauer (t)
1	Grobplanung	5
2	Feinplanung	10
3	Beschaffung	5
4	Fertigung Teil 1	10
5	Fertigung Teil 2	12
6	Montage (1+2)	5
7	Probetrieb	3
8	Abnahme	1

# Vorgangslisten

Vorg.-nummer	Vorgangsbeschreibung	Dauer (t)	Vorgänger AOB	Nachfolger AOB
1	Grobplanung	5	-	2
2	Feinplanung	10	1	3 NF-2
3	Beschaffung	5	2 NF-2	4; 5
4	Fertigung Teil 1	10	3	6
5	Fertigung Teil 2	12	3	6
6	Montage (1+2)	5	4 ; 5	7
7	Probetrieb	3	6	8
8	Abnahme	1	7	-

# Vorgangsliste = Liste der Arbeitspakte?

---

- Arbeitspakte:
  - Das unterste Element in einem PSP
  - Unterstruktur der Teilaufgaben
  - Haben eine klare Abgrenzung zu anderen Arbeitspaketen und können zur Abarbeitung an eine Org.-Einheit delegiert werden
- Vorgänge:
  - Arbeitspakte können zur Feinplanung in einzelne Vorgänge / Aktivitäten zerlegt werden
  - Sind Elemente der Detailnetz- und Balkenpläne, nicht aber des PSP

## Informationen im Knoten:

V.Nr.	V	D
<b>Vorgangsbezeichnung</b>		
FAZ	GP	FEZ
SAZ	FP	SEZ

# Nomenklatur II

---

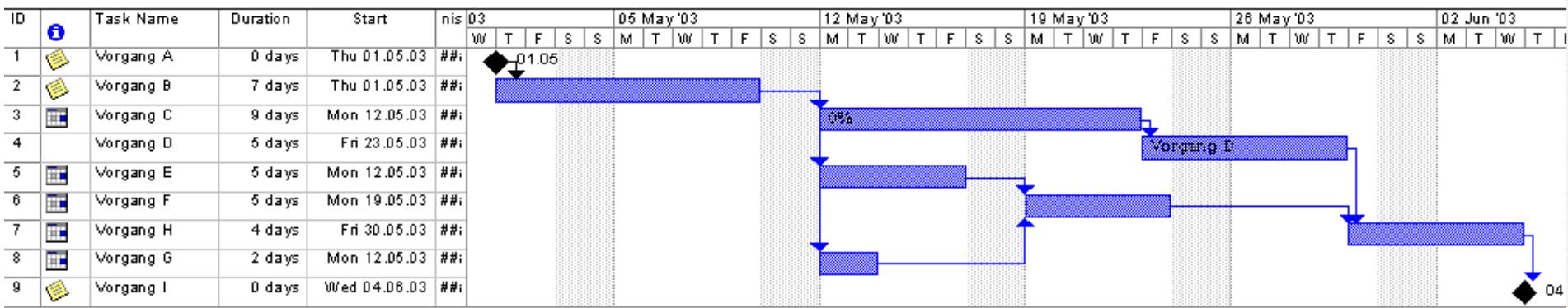
- V = Verantwortlicher (nicht zwingend Ausführender)
- D = Dauer des Vorgangs
- FAZ = frühester Anfangszeitpunkt
- FEZ = frühester Endzeitpunkt
- SAZ = spätester Anfangszeitpunkt
- SEZ = spätester Endzeitpunkt
- GP = Gesamtpuffer = Zeitrahmen, um den der Vorgang verschoben werden kann, ohne den Endtermin zu gefährden
- FP = freier Puffer = Zeitrahmen, um den der Vorgang verschoben werden kann, ohne den Nachfolger mit verschieben zu müssen (Autonomie des Vorgangs)

- Normalfolge (NF): Ende-Anfang-Beziehung
- Anfangsfolge (AF): Anfang-Anfang-Beziehung
- Endfolge (EF): Ende-Ende-Beziehung
- [Sprungfolge (SF): Anfang-Ende-Beziehung]

Die Terminberechnung erfolgt in 3 Schritten:

- Vorwärtsrechnung  
(Progressive Rechnung)
- Rückwärtsrechnung  
(Retrograde Rechnung)
- Berechnung der zeitlichen Spielräume  
(Gesamt- und Freie-) Puffer)

# Beispiel als Gantt-Diagramm



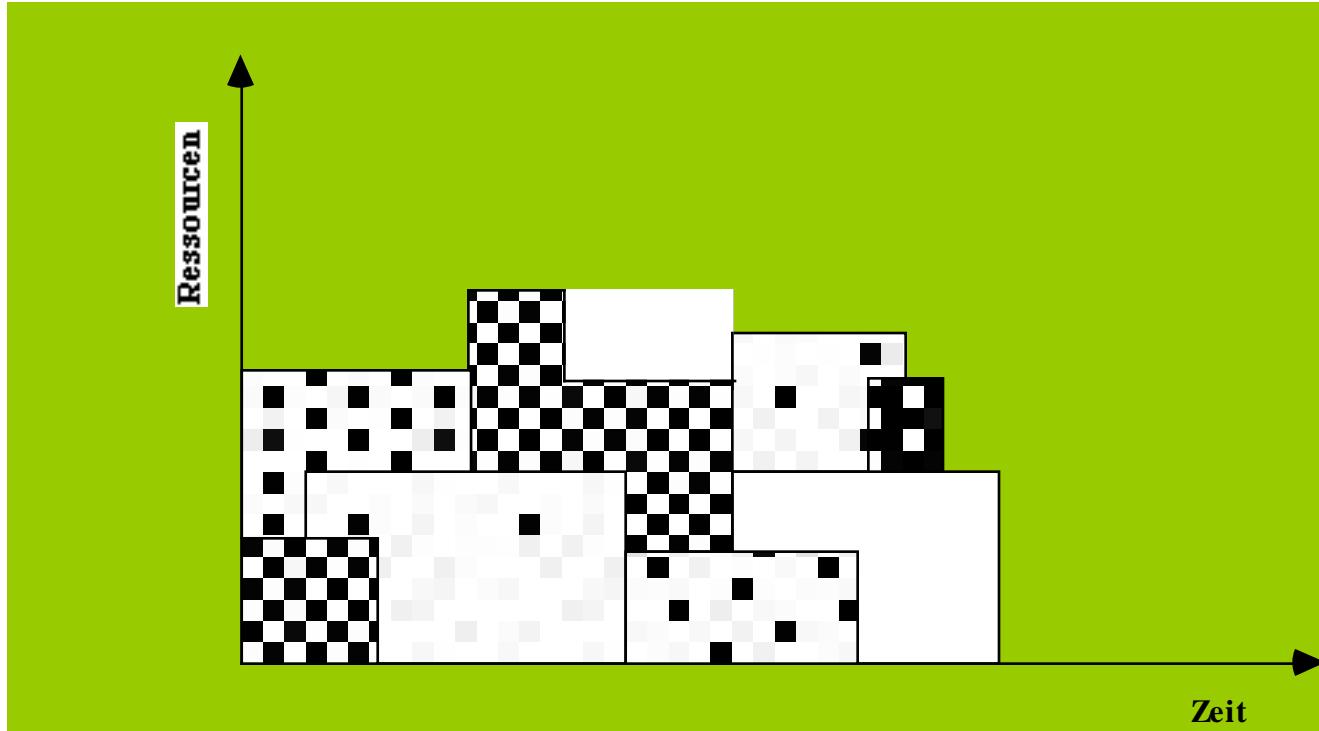
- MINZ: Minimaler Zeitabstand am Beispiel der NF
  - Positiv = minimale Wartezeit zwischen den Vorgängen (Rohbau muss x Wochen trocknen, bevor Innenausbau beginnen kann)
  - Negativ = maximale Vorziehzeit (Überlappung) für den Nachfolger (Produktion für Bauteile A kann maximal 1 Woche vor Verarbeitung beginnen (Lagerplatz))
- MAXZ: Maximaler Zeitabstand am Beispiel der NF
  - Positiv = maximale Wartezeit zwischen den Vorgängen darf nicht überschritten werden; ein Vorziehen ist jedoch möglich (Schulung vor Produktivsetzung)
  - Negativ = minimale Vorziehzeit (Überlappung) für den Nachfolger; ein Vorziehen ist jedoch möglich (Übergabe von Aufgaben an einen Nachfolger am Arbeitsplatz)

# Anfangs- und Abschlusstermine

- Bei vielen Projekten liegen Startbeginn und/oder Endtermin fest.
- Verwende T (=Termin) statt Z (Zeitpunkt)
- Vorsicht falls FET > SET
  - ⇒ Negative Gesamtpuffer
  - ⇒ Inkonsistenter Zeitplan
  - ⇒ Neuplanung erforderlich

V.Nr.	V	D
Vorgangsbezeichnung		
FAT	GP	FET
SAT	FP	SET

# Kapazitätsplanung



Ein Ablaufplan verliert ohne Einsatzmittel-Kapazitätsplanung an Aussagekraft!

## Beispiel:

Zwei Vorgänge werden für eine simultane Durchführung geplant, da dies technologisch möglich ist. Sie müssen infolge knapper Ressourcen (es gibt nur eine Person, die die Technologie beherrscht) jedoch nacheinander ausgeführt werden.

# Kapazitätsplanung

---

- Die Bearbeitung von Vorgängen beansprucht Ressourcen (Kapazitäten).
- Bei beschränkt zur Verfügung stehenden Kapazitäten muss die Ressourcenbeanspruchung geplant werden.
- Ziel ist eine möglichst hohe Auslastung der Kapazitäten.

## Definitionen nach DIN 69902:

**Einsatzmittel:** Personal und Sachmittel, die zur Durchführung von Vorgängen, Arbeitspaketen oder Projekten benötigt werden.

**Einsatzmittelart:** Gesamtheit von Einsatzmitteln, die nach bestimmten, allen gemeinsamen Merkmalen zusammengefasst sind (technisch, funktional, etc.).

## Definitionen nach DIN 69902:

**Einsatzmittelbedarf:** Menge von Einsatzmitteln einer bestimmten Einsatzmittelart, die zur Erzielung des Arbeitsergebnisses zu einem bestimmten Zeitpunkt oder innerhalb eines Zeitraumes erforderlich ist.

### **Einsatzmittelleistungsvermögen**

**(Kapazitätsbereitstellung):** Ist die Menge von Einheiten, die durch die Nutzung oder den Verbrauch eines Einsatzmittels in einer Zeiteinheit erzeugt werden kann.

# Kapazitätsplanung als Teil des Ganzen

---

Für die Gesamtprojektplanung bedeutet die Kapazitätsplanung eine Teillösung des Optimierungsproblems aus Einsatzmitteln, Zeit und Kosten.

Es wird ermittelt, wer die Arbeitspakete mit welchen Mitteln tatsächlich realisieren kann.

# Kapazitätsermittlung

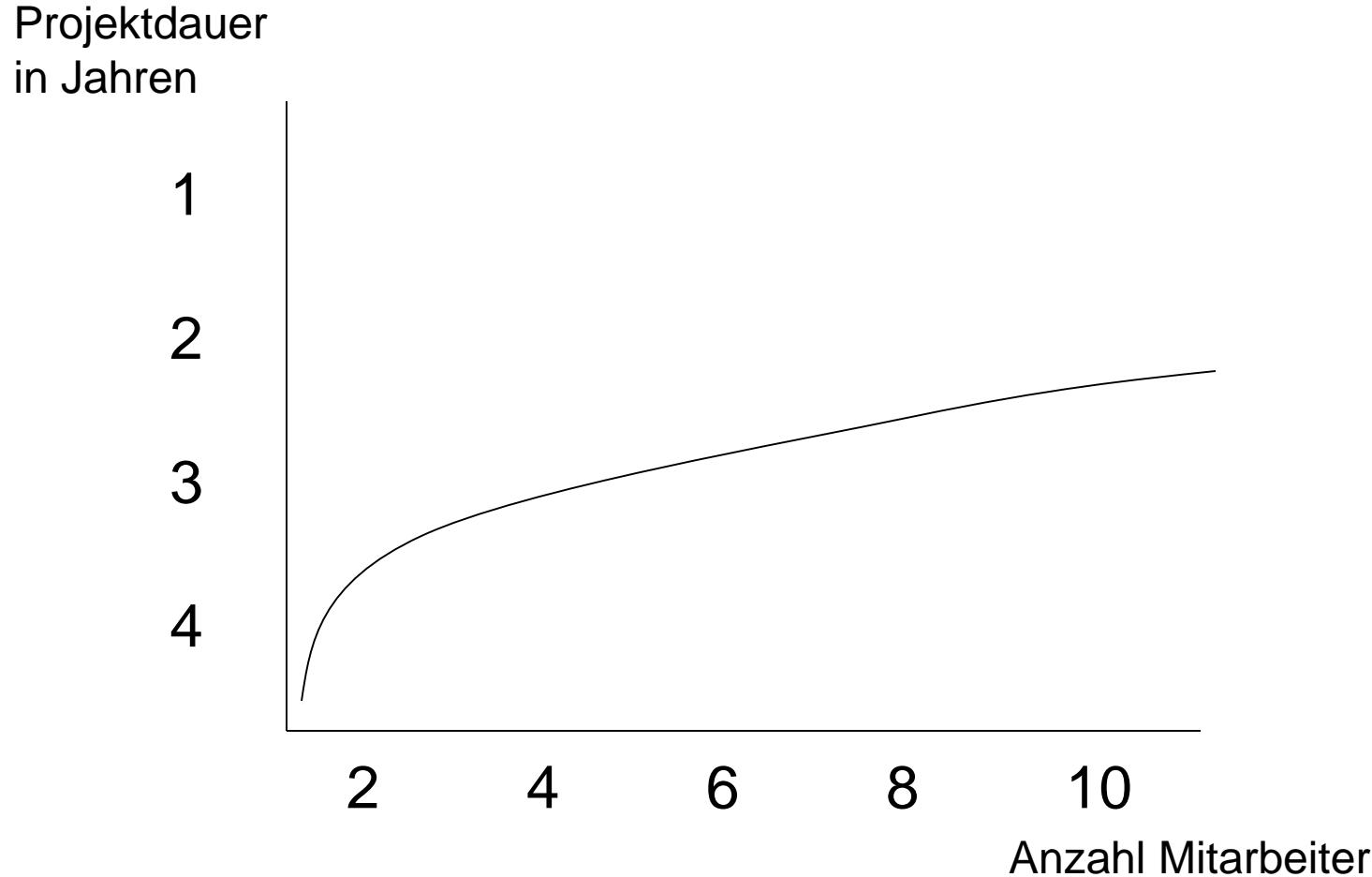
---

Wie sind Kapazitäten zu ermitteln?

Frage      nach der Arbeitsmenge,  
              nach der Anzahl der Einsatzmittel,  
              nach der Zeitdauer.

Arbeitsmenge = Anzahl Einsatzmittel \* Zeitdauer

# Mehr Personal bedeutet kein entsprechend kürzeres Projekt



# Abhängigkeiten

---

- Zwischen Einsatzmitteln kann eine feste Beziehung bestehen (z.B. Bagger – LKW).
- Es gibt einen deutlichen Zusammenhang zwischen Zeitplanung und Kapazitätsplanung. Die Kapazitätsermittlung ist aber i.d.R. von mehreren Nebenbedingungen für Einsatzmittel, von Zeit- und/oder Kostenbedingungen abhängig.

## Definitionen nach DIN 69902:

**Einsatzmittelbestand:** Anzahl der zu einem Zeitpunkt vorhandenen Einsatzmittel einer oder mehrerer Einsatzmittelarten.

**Einsatzmittelvorrat:** Teil des Einsatzmittelbestands, der für den zukünftigen Bedarf verfügbar gehalten wird.

# Einsatzmittelbewertung

---

Kosten = Zeitbedarf \* Einsatzmittelbedarf  
\* Stundensatz des Einsatzmittels

Auch bei scheinbar unabhängigen Arbeitspaketen können bei der Kapazitätsplanung wichtige Beziehungen bestehen.

→ Herstellen einer möglichst großen Übereinstimmung von Bedarf und Bestand an Einsatzmitteln unter Beachtung der Ziele und Randbedingungen (Abgleich der Kapazitäten nur bei zeitlich unkritischen Arbeitspaketen)

## Definitionen nach DIN 69902:

**Einsatzmittelauslastung:** Höhe der gesamten Inanspruchnahme des Einsatzmittelbestands in einem bestimmten Zeitraum.

**Bedarfsglättung:** Erzeugen eines möglichst gleichmäßigen Bedarfs einer Einsatzmittelart durch verschieben von Vorgängen innerhalb ihrer Pufferzeit.

# Einsatzmittelauslastung

---

- Kapazitätsganglinie eines Einsatzmittels bei frühestmöglichen Anfangszeiten für alle Vorgänge
- Kapazitätsganglinie eines Einsatzmittels bei spätest zulässigen Endzeiten für alle Vorgänge
- Durch das Verschieben von Vorgängen im Rahmen ihrer verfügbaren Pufferzeiten erfolgt die Glättung von Kapazitätsspitzen.

Man kann dazu Vorgänge

- **strecken**,
- **stauchen**,
- **unterbrechen**
- oder durch andere Einsatzmittel **ersetzen**. (Auswirkungen auf die übrigen Einsatzmittel beachten!)

## Möglichkeiten

- a) Kapazitätsplanung bei veränderbarer Projektdauer und festem Ablauf
- b) Kapazitätsplanung bei fester Projektdauer und veränderbarem Ablauf

Beide sollen immer das Ziel der Optimierung der Projektdauer haben!!

# Analyse der Kapazitätsauslastung

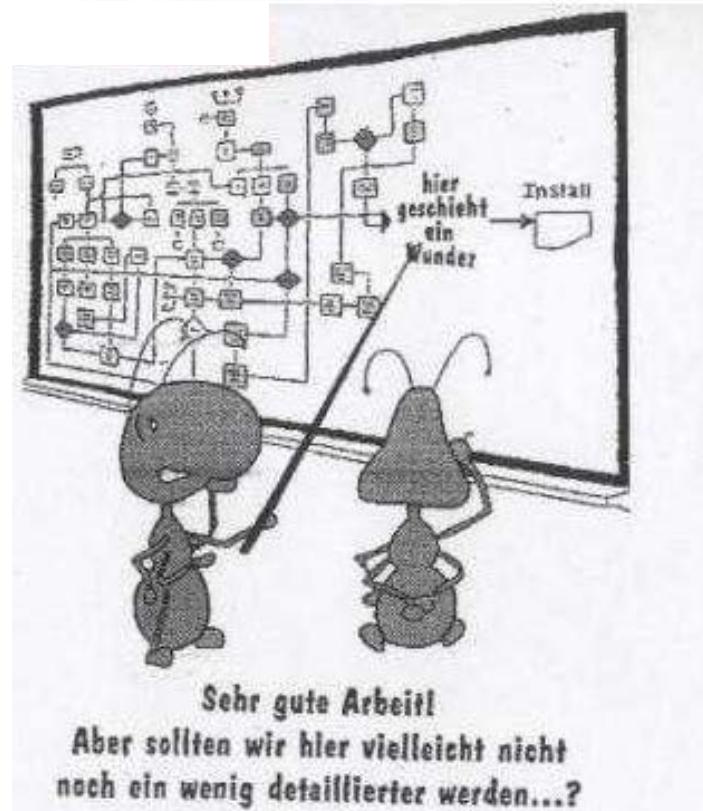
- Termin- und kapazitätstreue Auslastungsoptimierung
- Optimierung durch Untersuchung der frühesten und spätesten Ausführungstermine
- Kapazitätserweiterung
- Stückelung des Ressourceneinsatzes

# Projektplanung: Probleme in der Praxis

---

- Festpreis- versus Aufwandsprojekte
- Politisch motivierte Aufwandsschätzungen
- Interessenskonflikte: Projekt gewinnen oder ehrlich gegenüber dem Kunden sein?  
Oder geht beides?
- Wer plant auf welcher Ebene?
- Kunden- versus Anbieterressourcen

„Wenn ein Projekt kein Risiko birgt...  
lassen Sie die Finger davon!“



DeMarco/Lister (2003)

Ein Risiko ist ein potentielles, zukünftiges Problem, dessen Eintritt wichtige Projektziele oder Projektergebnisse gefährden kann.

Risiken bestehen bezüglich:

- der Realisierung des vereinbarten Leistungsumfangs
- der Qualität der Projektergebnisse
- der Kosteneinhaltung
- der Termineinhaltung

- Potentielle Risiken bleiben oftmals unberücksichtigt
- Aber: Murphy's Law gilt
- Und in Projekten immer in der schlimmsten Variante: das Unerwartete passiert genau dann, wenn man es am wenigsten gebrauchen kann.

# Risikomanagement:

## Definition

Unter Risikomanagement versteht man die Art und Weise, wie man beabsichtigt mit Risiken umzugehen. Dies beinhaltet

- die Identifikation und die Analyse von Risiken,
- die Beurteilung und Einschätzung von Risiken,
- die Entwicklung von Strategien im Umgang mit Risiken sowie
- die Kontrolle und Überwachung von Risiken

während der Projektumsetzung.

(Ruf/Fittkau (2007))

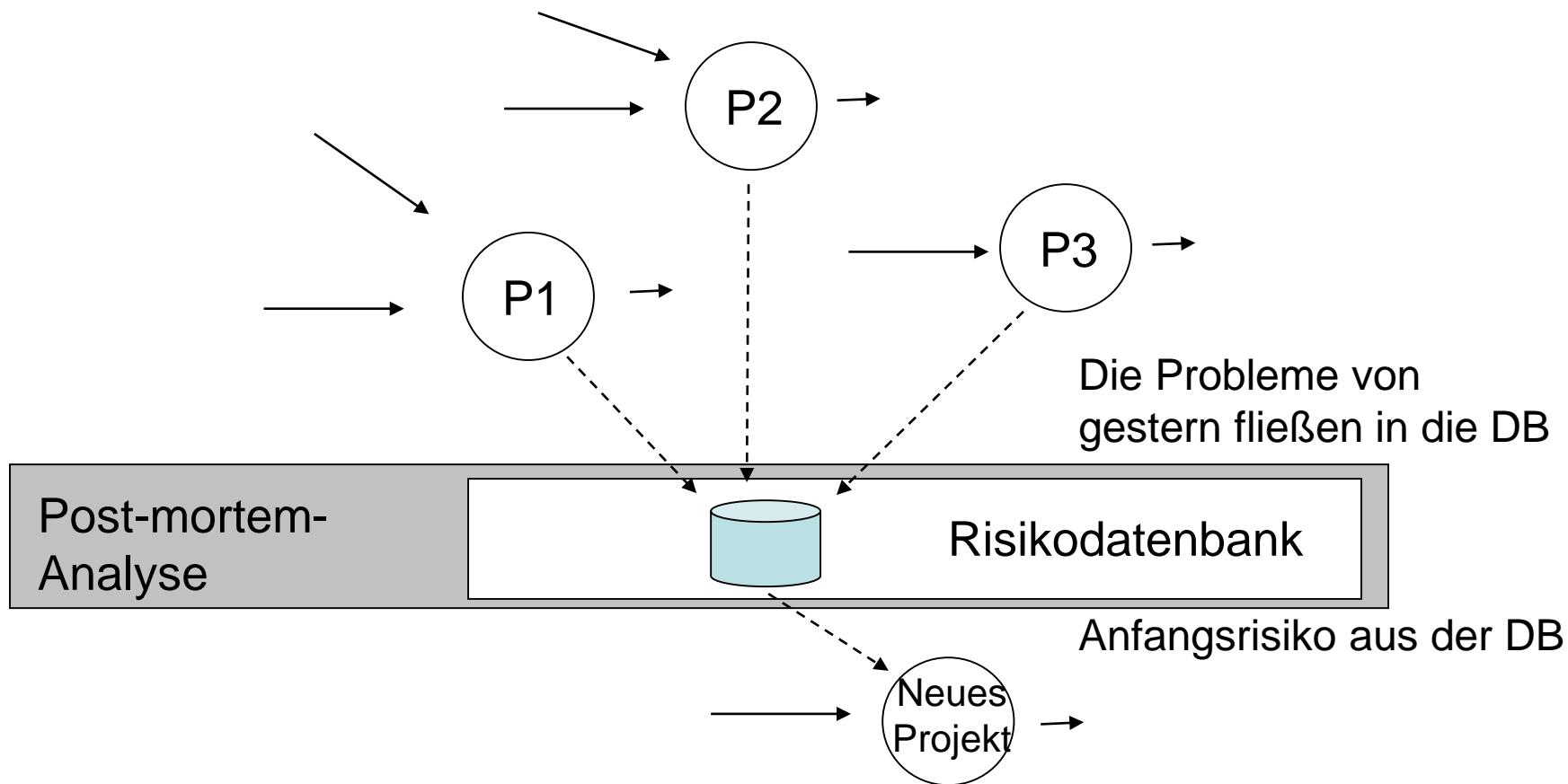
## Risikokultur

### Risikomanagementorganisation



# Risikoidentifikation

„Das Problem von gestern ist das Risiko von heute.“



## Technische Risiken

- Besitzen wir die notwendige Ausrüstung?
- Haben wir bereits Erfahrung mit der Entwicklungsumgebung?

## Betriebswirtschaftliche Risiken

- Ist die Bonität des Kunden in Ordnung?
- Gibt es genügend Puffer in der Kalkulation?

## Personelle Risiken

- Besitzen die Mitarbeiter die notwendige Qualifikation?
- Haben wir genügend Mitarbeiter zur Verfügung?

## Umwelt-Risiken

- Steht das Management hinter dem Vorhaben?
- Gibt es Einwände des Betriebsrates?

## Zulieferungs-Risiken

- Haben wir zuverlässige Lieferanten?
- Können wir kurzfristig auf andere Lieferanten ausweichen?

## Zeitrisiken

- Haben wir genügend Puffer eingeplant?
- Könnte es Einwirkungen geben, die wir nicht beeinflussen können (Streik, Wetter)?

## Mögliche Techniken

- Scoring Modelle
- Experten-Befragungen
- Portfolio-Analysen

# Risikobewertung: Scoring Modelle

---

## Scoring Modelle am Beispiel der Bewertung der Tragweite von Risiken:

**Tragweite:** Negative Auswirkungen eines Risikos auf die Projektziele im Falle des Risikoeintritts

- 1. Schritt:** Bestimme für jedes identifizierte Risiko die Auswirkungen auf die drei Ecken des PM-Dreiecks und belege jede Stufe mit einem Punktwert
- 2. Schritt:** Ermittle die aktuelle Gesamtbewertung des Risikos
- 3. Schritt:** Gewichte die Gesamtbewertung mit der Eintrittswahrscheinlichkeit des jeweiligen Risikos
- 4. Schritt:** Ermittle die Gesamtrisikobelastung des Projektes
- 5. Schritt:** Entscheide über notwendige Maßnahmen

# Risikobewertung: Scoring Modelle (Beispiel)

Auswirkung/ Projektziel	Sehr niedrig -1-	Niedrig -2-	Moderat -3-	Hoch -4-	Sehr hoch -5-
Liefer-/ Leistungs- umfang, Qualität	Kaum erkennbare Minderung	Nur geringfügige Minderungen	Wichtige Bereiche sind betroffen	Minderungen sind für den Kunden nicht akzeptabel	Projektergeb- nis ist unbrauchbar
Kosten	Unbedeutende Kostenstei- gerung	< 1% Kosten- steigerung	1-2% Kosten- steigerung	2-5% Kosten- steigerung	>5% Kosten- steigerung
Termine	Unbedeutende Terminverzö- gerung	< 2% Termin- verzögerung	2-5% Termin- verzögerung	5-10% Termin- verzögerung	>10% Termin- verzögerung

Bild 7: Berechnung der Tragweite

(aus: Harrant/Hemmrich (2004))

# Risikobewertung: Scoring Modelle (Beispiel)

Risiko: Die Radarantenne erreicht nicht die erforderliche Genauigkeit von +/- 3 Meter über die erforderliche Distanz von 10 km.

Auswirkung/ Projektziel	Sehr niedrig -1-	Niedrig -2-	Moderat -3-	Hoch -4-	Sehr hoch -5-
Liefer-/ Leistungs- umfang, Qualität	Kaum erkennbare Minderung	Nur geringfügige Minderungen	Wichtige Bereiche sind betroffen	Minderungen sind für den Kunden nicht akzeptabel	Projektergeb- nis ist unbrauchbar
Kosten	Unbedeutende Kostenstei- gerung	< 1% Kosten- steigerung	1-2% Kosten- steigerung	2-5% Kosten- steigerung	>5% Kosten- steigerung
Termine	Unbedeutende Terminverzö- gerung	< 2% Termin- verzögerung	2-5% Termin- verzögerung	5-10% Termin- verzögerung	>10% Termin- verzögerung

Die Gesamtbewertung der Auswirkungen des Risikos auf das Projekt beträgt demnach elf Punkte.

Bild 8: Beispiel zur Risikoeinstufung

(aus: Harrant/Hemmrich (2004))

# Risikobewertung: Scoring Modelle (Beispiel)

Eintrittswahrscheinlichkeit/Tragweite (Summe aus Kosten/Termin/Liefer-/Leistungsumfang)																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
100%	1	2	13	14	15	16	17	18	19	20	21	22	23	24	25	
90%	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	
80%	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
70%	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	
60%	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	
50%	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	
40%	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	
30%	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
20%	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	
10%	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	

Bild 9: Eintrittswahrscheinlichkeit/Tragweite Matrix

(10%-Eintrittswahrscheinlichkeit entsprechen je 1 Punkt)

(aus: Harrant/Hemmrich (2004))

# Risikobewertung: Scoring Modelle (Beispiel)

Eintrittswahrscheinlichkeit/Tragweite (Summe aus Kosten/Termin/Liefer-/Leistungsumfang) Die Eintrittswahrscheinlichkeit wurde mit 40% geschätzt.															
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
100%	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
90%	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
80%	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
70%	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
60%	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
50%	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
40%	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
30%	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
20%	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
10%	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Der „Risikowert“ des betrachteten Risikos beträgt 15 Punkte.

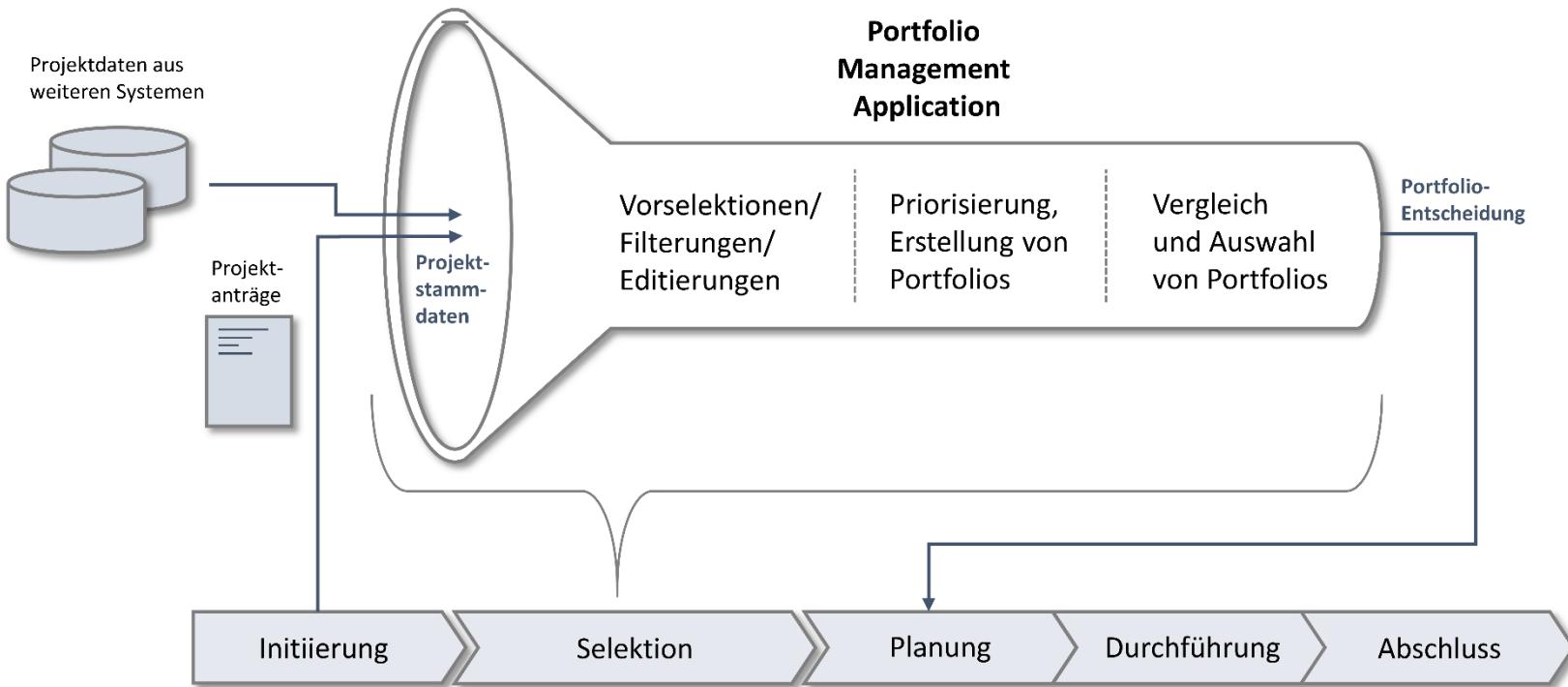
(aus: Harrant/Hemmrich (2004))

# Risikobewertung: Experten-Befragungen

---

- Delphi-Methode
- Ergebnisse:
  - Risikoliste
  - Eintrittswahrscheinlichkeiten

# Risikomanagement: Projektportfolio I



# Risikomanagement: Projektportfolio II

---

„Das Projektportfoliomangement ist ein Ansatz zum Erreichen von strategischen Zielen durch die Selektion, Priorisierung, Bewertung und das Management von Projekten, Programmen und damit zusammenhängenden Arbeiten auf Grundlage ihrer Ausrichtung und ihres Beitrages zu den Strategien und Zielen der Organisation.“ (PMI)

Ein Projekt kann anhand verschiedener Kriterien als richtige Wahl bezeichnet werden. Die wichtigsten Kriterien lauten:

- Strategische Bedeutung
- Wirtschaftlicher Nutzen
- Zwang
- Dringlichkeit
- Finanzielle und personelle Ressourcen
- Risiken

(Lomitz)

# Risikomanagement: Projektportfolio III

---

Die beiden Hauptprozesse des Projektportfoliomanagements (nach Kesten et al.):

- **Projektportfolioplanung:**

Umfasst im Wesentlichen die Auswahl der „richtigen“ Projekte und eine möglichst realistische Planung.

- **Projektportfoliosteuerung:**

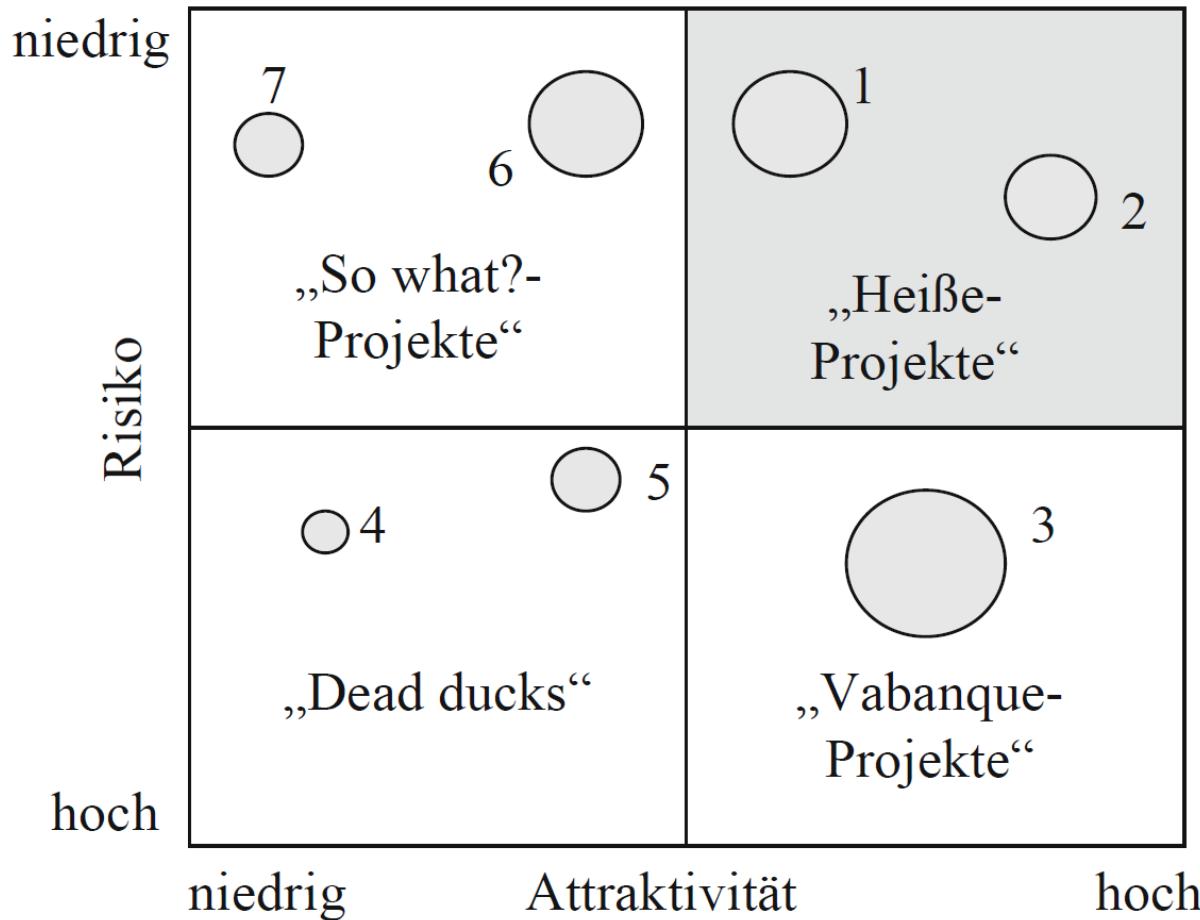
Das Ziel der Projektportfoliosteuerung ist die durchgehende Entwicklung des Projektportfolios, um mit der Unternehmenssituation gleich auf zu sein und bei Problemen rechtzeitig gegensteuern zu können. Dadurch lässt sich präventiv und vorausschauend handeln.

# Risikomanagement: Projektportfolio IV

---

- Portfoliotechnik zur Unterstützung der Auswahlentscheidung
- Vorschlag von Arthur D. Little: Achsen Attraktivität und Risiko
- Attraktivität: Beurteilung nach Umsatz- und Ertragspotential, Marktvolumen, -wachstum, Dauerhaftigkeit des Wettbewerbsvorteils
- Risiko: Man unterscheidet technische und wirtschaftliche Risiken

# Risikomanagement: Projektportfolio V



Quelle: Fiedler (2001)

# Risikomanagement: Projektportfolio VI

Portfolio Selection Application | Visualize and compare scenarios

© Campana & Schott 2017 - Portfolio Selection Application

Current Step: 4

Scenarios:

- Cost saving scenario
- Market expansion scenario

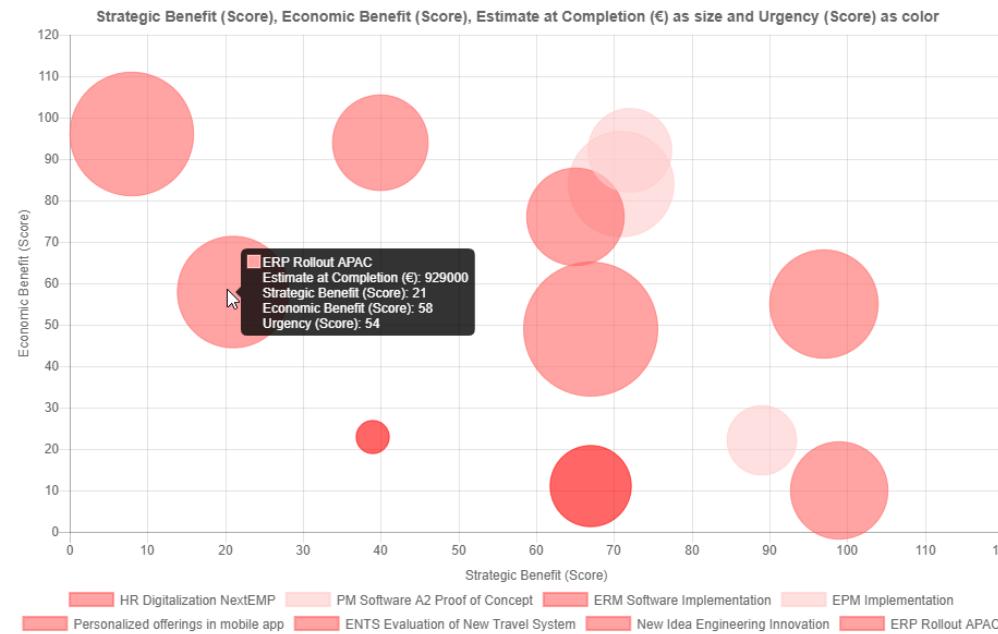
Total Resource Expenses:

Budget: € 9.976.100

Personal Resource Requirements:

	Jan 17	Feb 17	Mar 17	Apr 17	May 17	Jun 17	Jul 17	Aug 17	Sep 17
Software Engineers (FTE)	11	23 (-1)	23	31 (-1)	37 (-2)	37	42	42	42
Process Experts (FTE)	12	34	34	42 (-2)	43	43	47	47	47
UI Experts (FTE)	5	8	8	12	14	14	16 (-1)	16 (-1)	16
Testers (FTE)	4	15	15	17	18	18 (-3)	21 (-1)	21	21 (-1)

Strategic Benefit (Score), Economic Benefit (Score), Estimate at Completion (€) as size and Urgency (Score) as color



Estimated data points from the bubble chart:

Project	Strategic Benefit (Score)	Economic Benefit (Score)	Estimate at Completion (€)	Urgency (Score)
ERP Rollout APAC	20	65	929000	54
HR Digitalization NextEMP	10	110	120000	21
PM Software A2 Proof of Concept	40	25	100000	58
ERM Software Implementation	60	80	150000	58
EPM Implementation	70	100	100000	58
Personalized offerings in mobile app	75	70	80000	58
ENTS Evaluation of New Travel System	85	20	70000	58
New Idea Engineering Innovation	90	25	60000	58
ERP Rollout APAC	100	60	100000	58
Project Teams Integration	105	15	50000	58
Backup IT Vendor System Rollout	110	20	40000	58
PPM Implementation	115	10	30000	58
Acquisition Target Analysis	120	5	20000	58

# Risikomanagement: Maßnahmenplanung

---

- Suche nach gemeinsamen Ursachen für die identifizierten Projektrisiken (Mehrere Fliegen mit einer Klappe schlagen)
- Plane und initiere Risikomaßnahmen so früh wie möglich im Projekt. Rechtzeitige und schnell umsetzbare Maßnahmen sind oftmals effektiver und kostengünstiger als „perfekte“ Maßnahmen, die zu spät realisiert werden!
- Potentielle Risikomaßnahmen:
  - Angemessene und eindeutige Vertragsformulierungen
  - Gut funktionierendes Projektteam (der Mix ist wichtig!)
  - Strukturierter Projektprozess
  - Realistische Projektplanung
  - Klares Eskalationsmanagement
  - ...

# Risikomanagement:

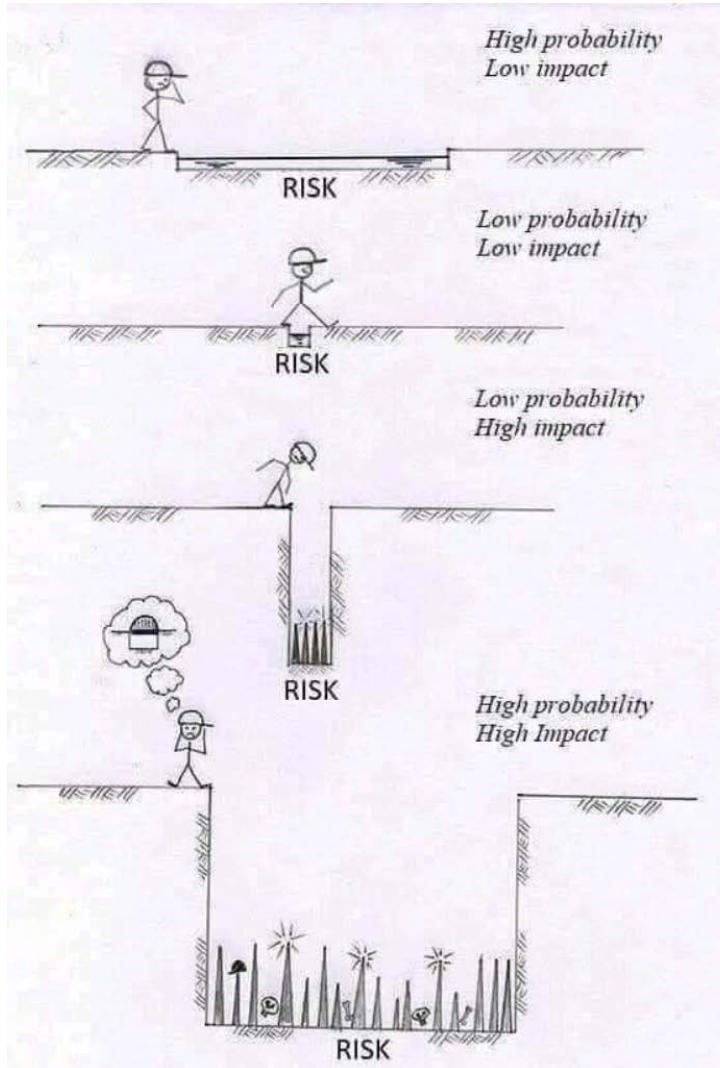
## Der Fertigstellungstermin

---

Sie können den zu Beginn eines Projektes genannten Termin (30.10.) definitiv nicht mehr halten. Jetzt (August) möchte ihr Kunde eine neue Terminabschätzung haben. Eine erneute Analyse führt zu folgenden Ergebnissen:

1. Es gibt keine Chance vor dem 1.1. fertig zu werden.
2. Das wahrscheinlichste Datum für die Lieferung ist Anfang April.
3. Wenn wir den 1. Mai nennen, werden wir es mit einer Wahrscheinlichkeit von etwas über 50% schaffen, den Termin zu halten.
4. Wenn wir einen Termin nennen müssen, der praktisch nicht platzen kann, so wäre dies Ende Dezember.

# Zum Zusammenhang von Impact, Eintrittswahrscheinlichkeit und Strategie



Wie gehe ich  
damit um?



# Risikomanagement: Risikostrategien

---

## 1. Risiken vermeiden:

- Die beste/billigste Strategie im Risikomanagement! (Kenne ich den ursprünglichen Auslöser des Risikos, kann ich ihn in der Projektplanung evtl. ausschalten.)
- Man spricht dann von *Risk-Mitigation-Plans*

## 2. Risiken transferieren:

- Auftraggeber (über Vertragsgestaltung)
- Unterauftragnehmer und Lieferanten (über *Back-to-Back*-Verträge)
- Versicherungen (z.B. Währungsrisiken)

## 3. Risiken vermindern:

- QM-Maßnahmen
- Bewährte Unterauftragnehmer einsetzen
- Erfolgreiche Standards verwenden

## 4. Risiken tragen:

- Aktive Akzeptanz: Es gibt einen *Contingency-Plan*
- Passive Akzeptanz: Reaktion erst bei Eintritt des Risikofalls (situativ/ad-hoc)
- Typische Maßnahmen: Zusätzliche Ressourcen, Projektverlängerung

# Risikomanagement: Die Realität

---

## Aus einer Umfrage:

In außer Kontrolle geratenen Projekten wurde in einer Untersuchung festgestellt, dass in

- 55% der Fälle gar kein Risikomanagement betrieben wurde
- 38% der Fälle nur halbherzig ein Risikomanagement vorhanden war und
- in 7% der Fälle wusste man nicht, ob ein Risikomanagement überhaupt gemacht wurde.

(Hindel et al. (2004))

- Risikomanagement ist ein kritischer Erfolgsfaktor
- Risikomanagement als „Lebensversicherung“ im Projekt

# Risikomanagement: Risikocontrolling

---

„Risikocontrolling ist ein kontinuierlicher Prozess zum Überwachen und Steuern der Projektrisiken. Er beginnt direkt nach der Identifizierung und Analyse der Projektrisiken und endet erst mit vollständiger Erledigung des Projekts.“

[...]

Risikocontrolling bedeutet:

- Regelmäßige Überwachung der identifizierten Projektrisiken
- Umsetzen der Risikomaßnahmen
- Überwachung der Wirksamkeit der Risikomaßnahmen
- Überwachen der Restrisiken
- Identifizieren und Analysieren neuer Risiken.“

(Harrant/Hemmrich (2004))

# Risikomanagement:

## Die 10 Gebote

---

1. Risiken gehören zu jedem Projekt und sind nicht zu vermeiden.
2. Die „Entdecker“ von Risiken sind nicht wie „Schuldige“ zu behandeln.
3. Alle Projektbeteiligten sind für das Risikomanagement verantwortlich.
4. Um drohende Risiken zu erkennen, ist offene und ehrliche Kommunikation im Projektteam unverzichtbar.
5. Risiken werden nicht nur vom Projektleiter bekämpft, sondern durch abgestimmtes und gemeinsames Handeln des Projektteams.
6. Risiken und diesbezügliche Abwehrmaßnahmen gehören zu den TOP-1-Themen jeder Projektbesprechung.
7. Beschlossene Abwehrmaßnahmen und Entscheidungen werden konsequent umgesetzt und auf Wirksamkeit hin überwacht.
8. Risikomindernde oder –vermeidende Maßnahmen werden möglichst auf der Arbeitsebene entschieden und nicht zum Management „delegiert“.
9. Sofortige, wirksame Maßnahmen sind perfekten Maßnahmen, die aber zu spät kommen, vorzuziehen.
10. Vorgehensweisen und Tools im Risikomanagement werden im Projektteam abgestimmt und einheitlich angewendet.

(Harrant/Hemmrich (2004))

# Risikomanagement: Konsequenzen

## Projektklassifizierung und –behandlung

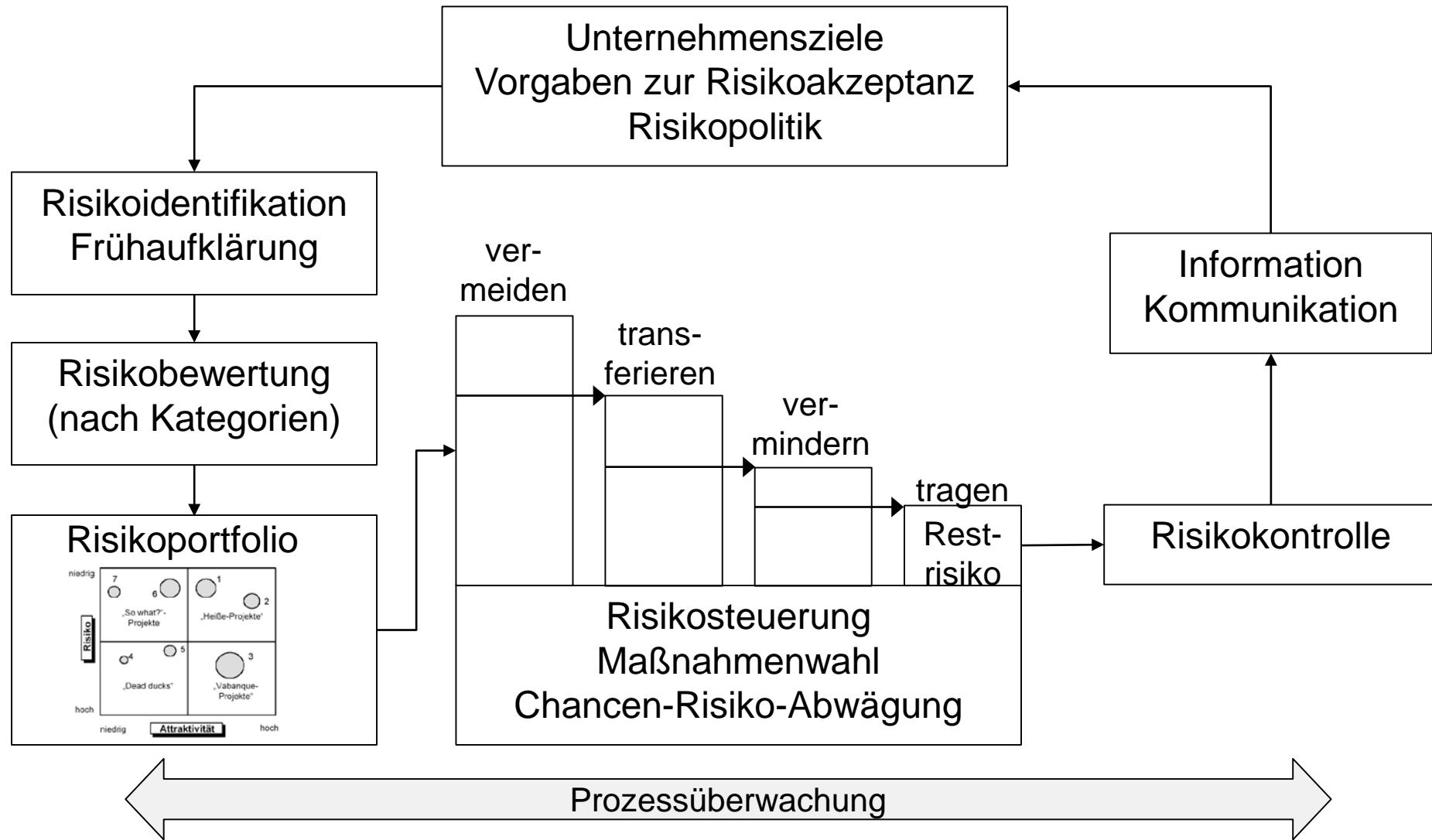
Hohes Risiko	Geringes Risiko
Aufwandsvertrag	Festpreisvertrag
Risikoaufschlag in der Kalkulation	Kein Risikoaufschlag in der Kalkulation
Wirtschaftlichkeitsanalyse	Keine Wirtschaftlichkeitsanalyse nötig

# Strategisches Projektcontrolling

---

- Besonders wichtig im Multiprojektumfeld
- Allokationsproblem der begrenzten Ressourcen
- Begrenzung des Projektrisikos
- Aufzeigen von Konsequenzen durch Planänderungen

# Risikomanagement-Kreislauf



# Risikomanagement: DeMarcos Tipps

---

- Definieren Sie, wer für das Risikomanagement zuständig ist!
- Alle Beteiligten müssen gefahrlos auf Risiken hinweisen dürfen!
- Es muss wahrscheinlich sein, dass Sie den Fertigstellungstermin um mindestens 20 Prozent unterschreiten können!
- Veröffentlichen Sie eine Liste der wichtigsten Risiken mit ihrer Wahrscheinlichkeit und den Konsequenzen für die Kosten und den Termin!

# Agile Vorgehensweisen

---

# Motivation: Probleme der Software-Entwicklung

---

- Kommunikationsprobleme
- Hohe Komplexität
- Inkonsistenzen
- Unzureichende Automatisierung
- Unflexible Architekturen

**Während einer Fahrt zum Mond würde vermutlich nicht darüber diskutiert, ob vielleicht der Mars das bessere Ziel wäre.**

- Grenzen der traditionellen Methoden im Bereich der Softwareentwicklung
- Vorgehensweise im agilen Projektmanagement
- Flexibilität nicht fürchten sondern nutzen!

# Grenzen der traditionellen Methoden im Bereich der Softwareentwicklung

- Die sehr hohe Innovationsgeschwindigkeit im IT-Bereich erzwingt entsprechend schnelle Produktzyklen.
- Kunden und Markt sind nicht mehr in der Lage, eigene Anforderungen zu definieren, sondern reagieren nur noch auf die Präsentation neuer technischer Möglichkeiten und Produkte.
- “The greater the uncertainty, the faster the pace of change, and the lower the probability that success will be found in structure.”
- SW-Entwicklungsprojekte und das „battlefield problem“:  
„Kein Plan überlebt die erste Feindberührung“  
(Feldmarschall Helmuth von Moltke)

# Agiles Manifest (2001)

## Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- **Individuals and interactions over processes and tools**
- **Working software over comprehensive documentation**
- **Customer collaboration over contract negotiation**
- **Responding to change over following a plan**

That is, while there is value in the items on the right, we value the items on the left more.

# Individuals and interactions over processes and tools

---

People are the most important ingredient of success. A good process will not save the project from failure if the team doesn't have strong players; but a bad process can make even the strongest of players ineffective. [...]

A strong player is not necessarily an ace programmer. A strong player may be an average programmer, but someone who works well with others. Working well with others, communicating and interacting, is more important than raw programming talent. A team of average programmers who communicate well are more likely to succeed than a group of superstars who fail to interact as a team.

The right tools can be very important to success. [...] However, tools can be overemphasized. An overabundance of big unwieldy tools is just as bad as a lack of tools.

# Working software over comprehensive documentation

---

Software without documentation is a disaster. Code is not the ideal medium for communicating the rationale and structure of a system. Rather, the team needs to produce human readable documents that describe the system, and the rationale for their design decisions. However, too much documentation is worse than too little. [...]

The code does not lie about what it does. It may be hard to extract rationale and intent from the code; but the code is the only unambiguous source of information.

# Customer collaboration over contract negotiation

---

Software cannot be ordered like a commodity. You cannot write a description of the software you want and then have someone develop it on a fixed schedule for a fixed price. [...]

It is tempting for the managers of a company to tell their development staff what their needs are, and then expect that staff to go away for a while and return with a system that satisfies their needs. But this mode of operation leads to poor quality and failure. Successful projects involve customer feedback on a regular and frequent basis. [...]

In most cases the terms it [the project] specifies become meaningless long before the project is complete. The best contracts are those that govern the way the development team and the customer will work together.

# Responding to change over following a plan

---

When we build plans, we need to make sure that our plans are flexible and ready to adapt to changes in the business and technology.

The course of a software project cannot be predicted far into the future. There are too many variables to account for. We simply aren't very good at estimating the cost of a large project. [...] It is difficult to write reliable requirements. Customers are likely to alter the requirements once they see the system start to function.

It is tempting for novice managers to create a nice PERT or GANTT chart of the whole project, and tape it to the wall. They may feel that this chart gives them control over the project. They can track the individual tasks and cross them off the chart as they are completed. They can compare the actual dates with the planned dates on the chart and react to any discrepancies.

But what really happens is that the structure of the chart degrades. As the team gains knowledge about the system, and as the customer gains knowledge about their needs, certain tasks on the chart will become unnecessary. Other tasks will be discovered and will need to be added. In short, the plan will undergo changes in *shape*, not just changes in dates.

# Die 12 agilen Prinzipien

1. Die höchste Priorität besteht darin, den Kunden durch frühe und zahlreiche Lieferungen hochwertiger Software zufriedenzustellen.
2. Veränderte Anforderungen sollten immer positiv aufgenommen werden, selbst wenn sie sich erst spät in der Entwicklung zeigen. Agile Verfahren nutzen Änderungen als Wettbewerbsvorteil für den Kunden.
3. Funktionierende Software muss regelmäßig geliefert werden, innerhalb weniger Wochen oder Monate, wobei der kürzeren Zeitspanne eindeutig der Vorzug zu geben ist.
4. Mitarbeiter aus den Fachbereichen und Entwickler arbeiten täglich gemeinsam im Projekt.
5. Bauen Sie Ihr Projekt um motivierte Mitarbeiter auf. Geben Sie diesen Mitarbeitern die Umgebung und die Unterstützung, die sie benötigen, und vertrauen Sie darauf, dass die Mitarbeiter ihre Arbeit gut machen.
6. Die effizienteste und effektivste Methode zur Informationsübermittlung für und innerhalb eines Entwicklungsteams besteht in der direkten Kommunikation.
7. Funktionierende Software ist der primäre Maßstab für den Fortschritt.
8. Agile Prozesse fördern eine kontinuierliche Entwicklung. Geldgeber, Entwickler und Benutzer sollten endlos ein beständiges Tempo beibehalten.
9. Ständige Aufmerksamkeit gegenüber technisch hervorragender Qualität sowie gutem Design verbessert die Agilität.
10. Einfachheit – die Kunst, die Menge der nicht geleisteten Arbeit zu maximieren – ist existentiell.
11. Die besten Architekturen, Anforderungen und Designs entwickeln sich in Teams, die sich selbst organisieren.
12. Die Teams müssen in regelmäßigen Abständen darüber beraten, wie sie noch effektiver vorgehen können, und ihr Verhalten entsprechend anpassen.

# Was heißt „agil“?

---

- Agil heißt **beweglich** und **flexibel** sein; Mitdenken statt „Dienst nach Vorschrift“ und Dogma.
- „Agilität“ bedeutet aber nicht das Zulassen von Anarchie und nicht einfach „Trial and Error“.
- Es gibt Vorgaben, diese lassen jedoch mehr Spielraum für Flexibilität.

# Agiles PM als Führungsmodell für Softwareprojekte

---

## Besondere Aspekte:

- Offenheit für Änderungen statt Festhalten an starren Vorgaben
- Ergebnisorientierung an Stelle von Prozess-/Vorgehensorientierung
- Kommunikation statt Dokumentation
- Vertrauen statt Kontrolle
- Untereinander „Best Practices“ austauschen und etablieren statt zentral Anforderungen festschreiben
- Kurze Iterationen statt langer Projektphasen
- Schätzungen der Wahrscheinlichkeit der Zielerreichung statt Restaufwände
- Reviews statt Zahlenkolonnen
- Schnelles Feedback statt endlose Abnahmetests
- Plankorrektur statt Ad-hoc Reaktionen

# Agile Methoden in der Softwareentwicklung im Überblick

- Lean development (LD; auch als LSD zu finden)
- Adaptive software development (ASD)
- Feature-driven development (FDD)
- eXtreme Programming (XP)
- Scrum
- Kanban

# Neue Ansätze im Innovations- und Lösungsmanagement

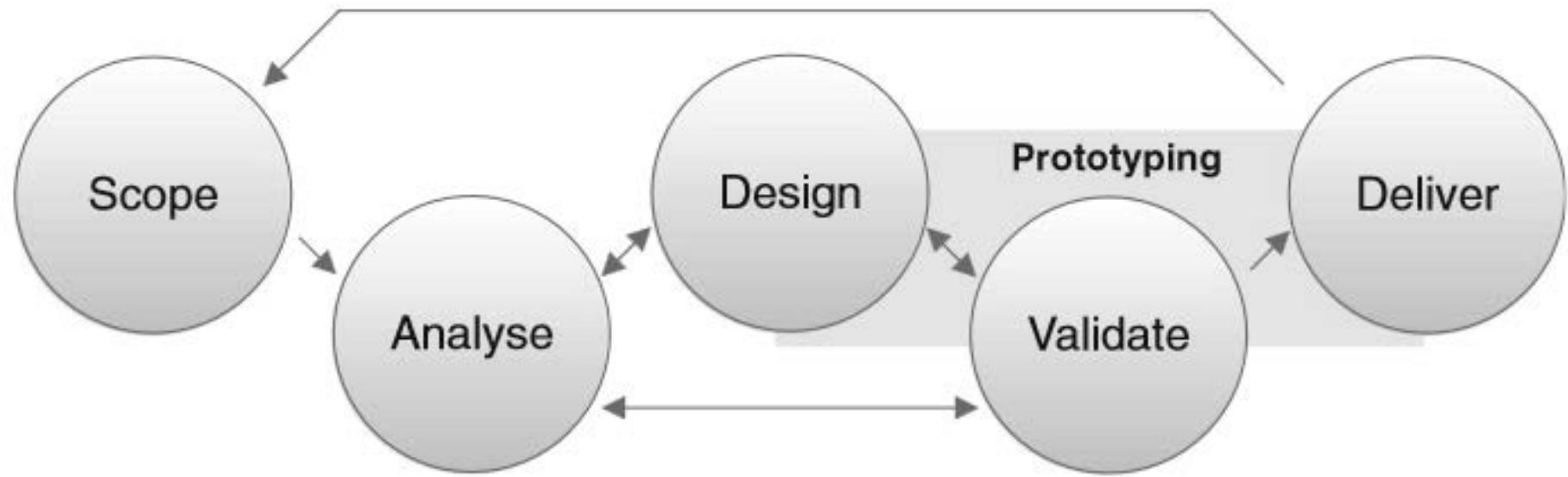
---

# Manchmal gibt es auch einfache Lösungen...

- Fahren Sie gern Cabrio?
- Probleme mit dem Wind?
- Lösung: Windschott
  - 1. Frage: Wann wurde das erste Cabrio gebaut?
  - 2. Frage: Wann wurde das Windschott entwickelt?



# User-centered design activities

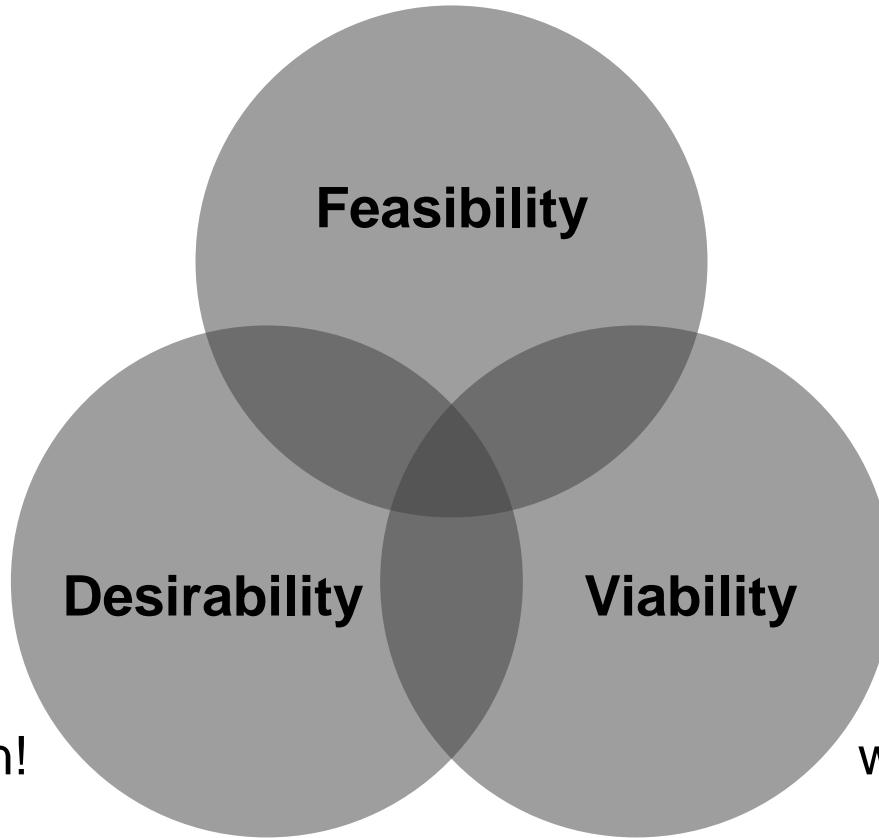


Quelle: Wallach und Scholz (2012)

“DESIGN THINKING means...  
...creating INNOVATION by  
combining diverse PEOPLE, creative  
SPACE and an iterative APPROACH.”

Quelle: SAP (2014)

Es muss technisch  
umsetzbar sein!

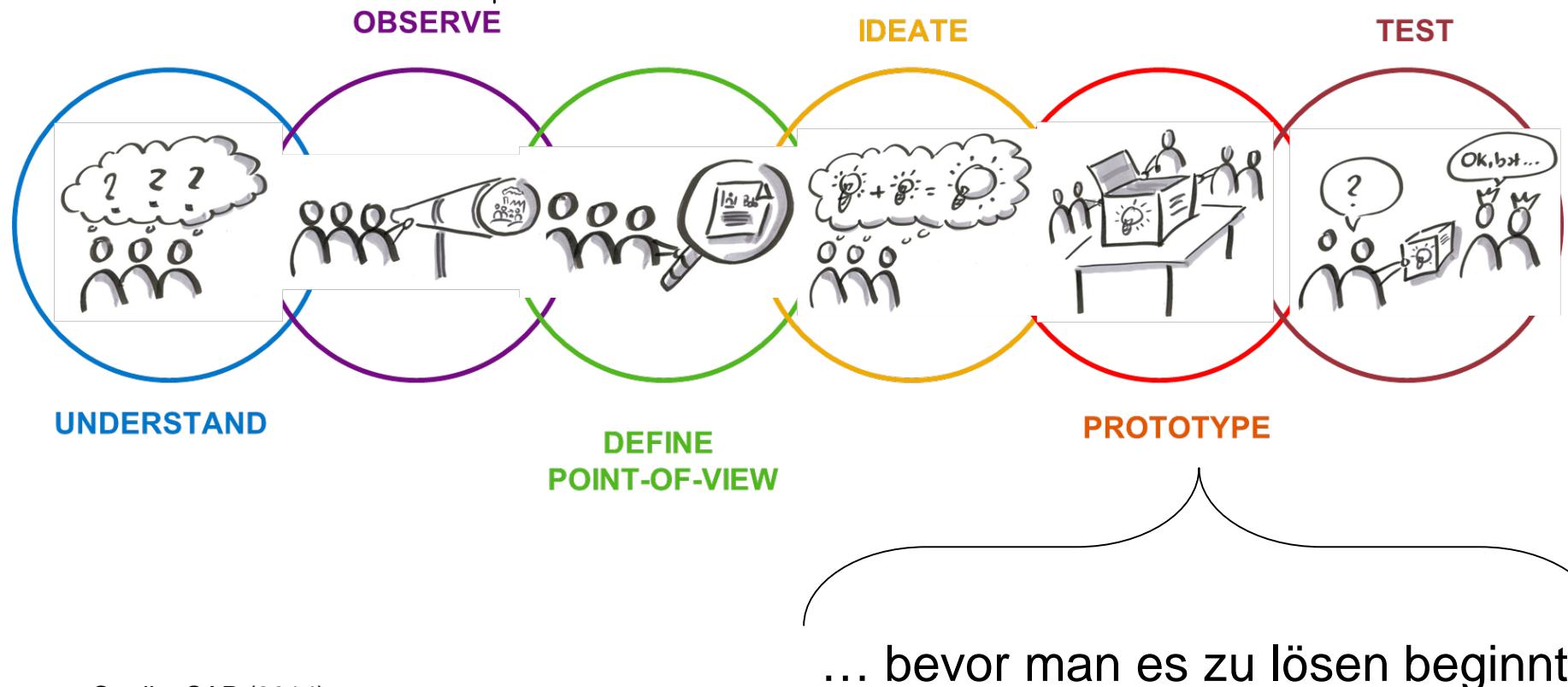


Es muss eine  
Nachfrage geben!

Es muss sich  
wirtschaftlich lohnen!

# Design Thinking Process

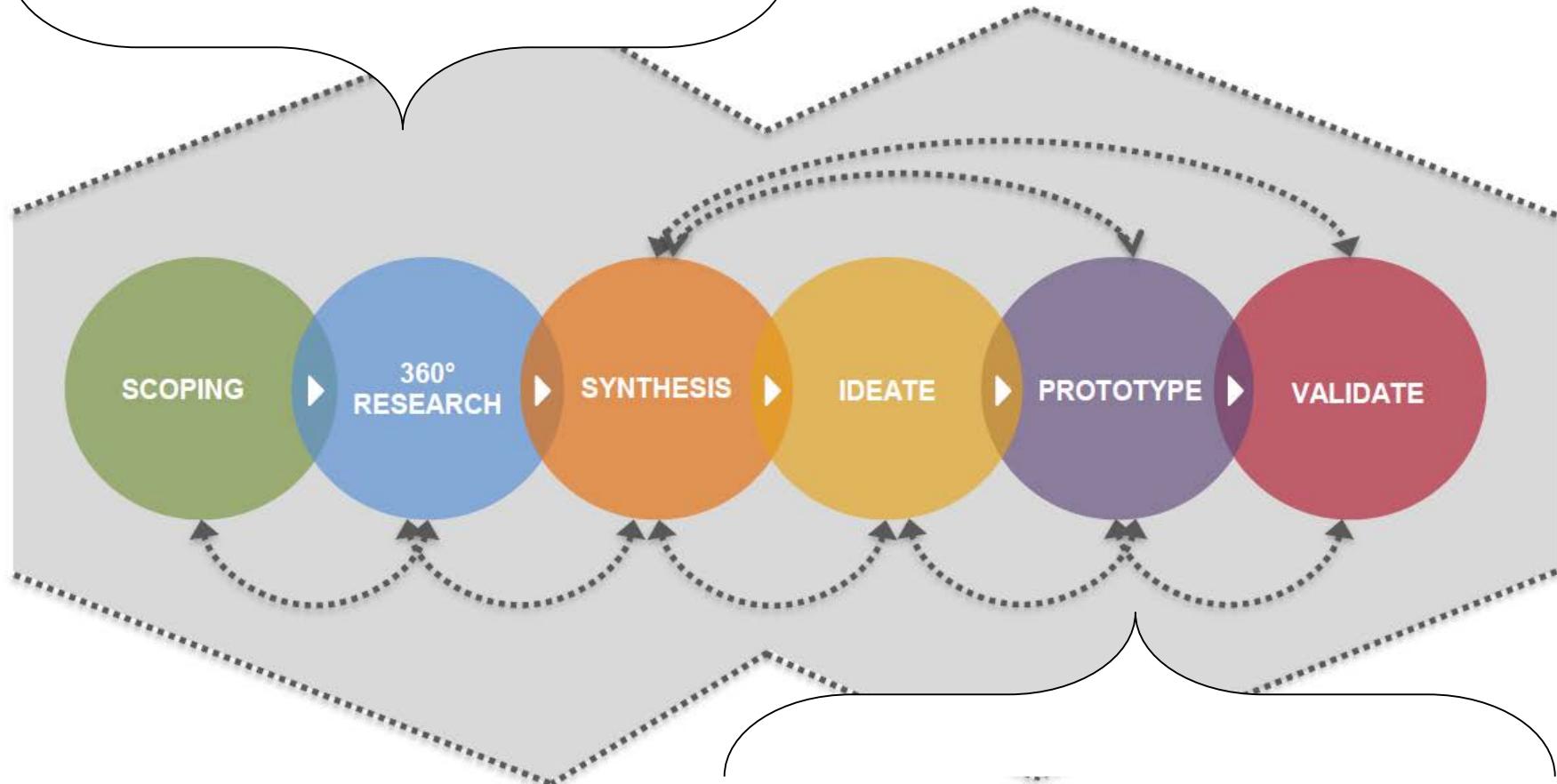
Das Problem verstehen...



Quelle: SAP (2014)

# Design Thinking Process – Die Phasen

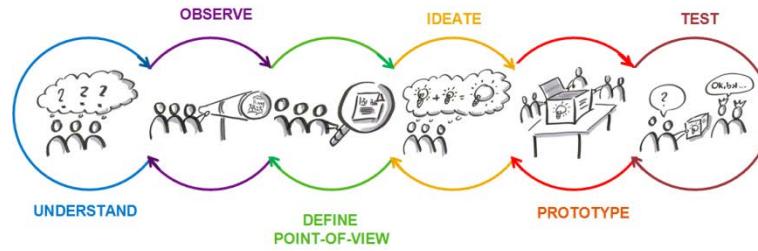
Das Problem verstehen...



Quelle: SAP (2014)

... bevor man es zu lösen beginnt!

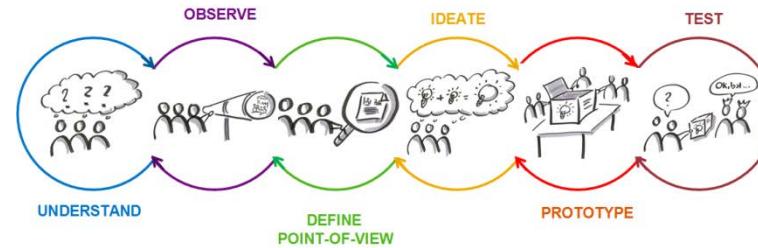
# Das Gesamtpaket für Design Thinking



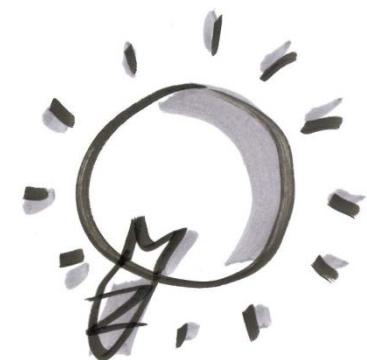
**People + Space + Approach → Innovation**



# Das Gesamtpaket für Design Thinking: People



**People** + Space + Approach → Innovation

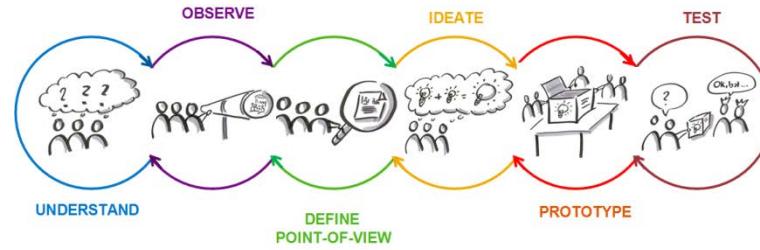
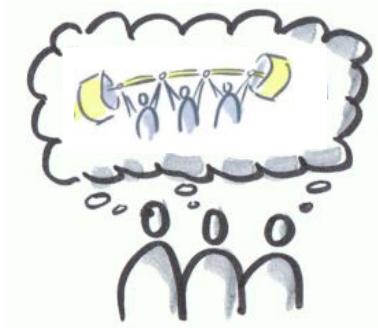


# Das Team ist wichtig

- Moderne SW-Entwicklung braucht Teams und keine „einsamen Entwicklerwölfe“...
- Interdisziplinäre Teams fördern die Kreativität
- In Team-Building investieren
- Team-Regeln definieren (und einhalten!)



# Das Gesamtpaket für Design Thinking: Space



# People + Space + Approach → Innovation



# Neue Arbeitsumgebung

- Flexible und angepasste Arbeitsumgebungen sollen Kreativität erhöhen
- Beispiel SAP App House Ireland



Gibt es auch  
näher!



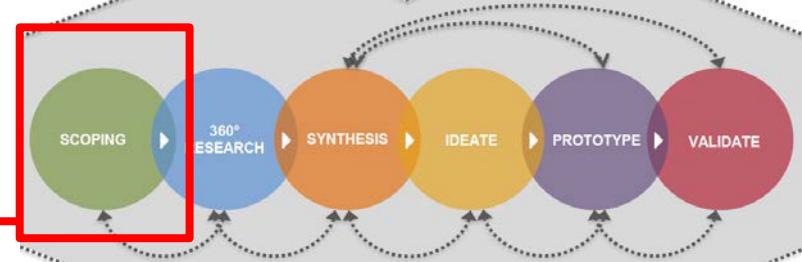
# Das Gesamtpaket für Design Thinking: Approach



People + Space + **Approach** → Innovation



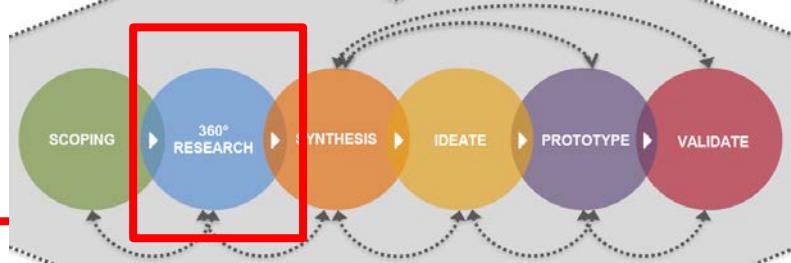
# Scoping



- Das Problem verstehen: Worum geht es?
- Den Fokus definieren und gegebenenfalls variieren/ändern/shiften
- Erste Ansätze entwickeln
- Research vorbereiten

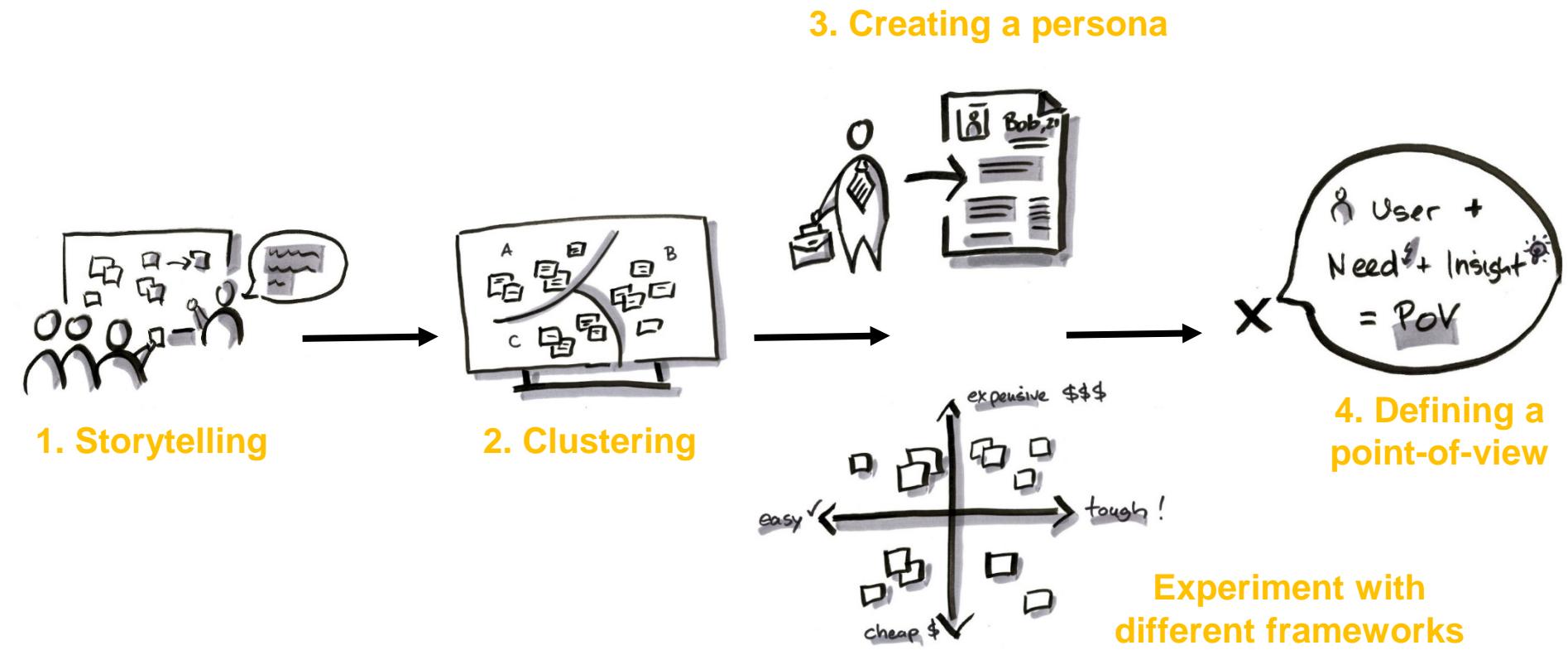
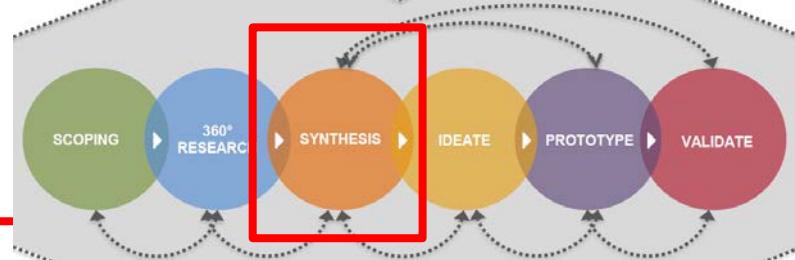
***“Building something nobody wants is the ultimate form of waste.” – Eric Ries***

# 360° Research



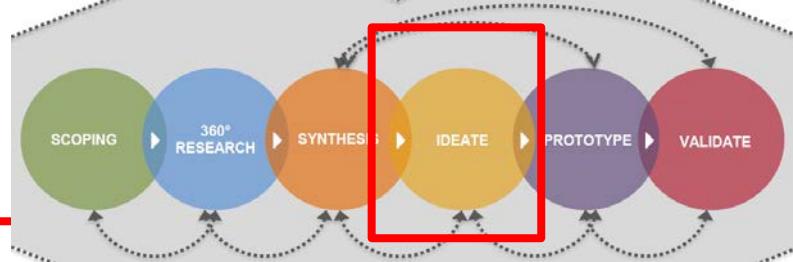
- Den Anwender/Kunden befragen: Direktes Feedback und Hinweise vom späteren Nutzer bekommen
- Empathie entwickeln!
- Fragen und Zuhören
- Wie vorgehen?
  - Feldforschung
  - Expertenbefragung
  - Konkurrenzanalyse
  - Ähnliche Situationen analysieren

# Synthesis: Die Kunst seine Einsichten zu strukturieren



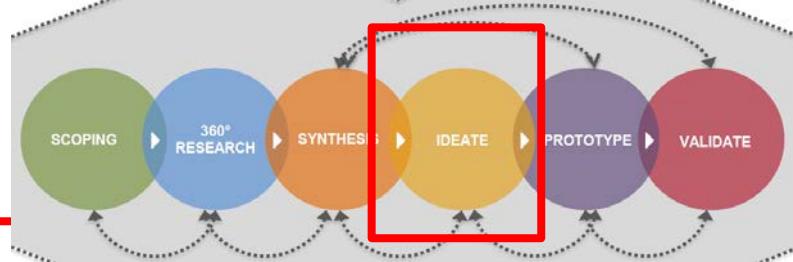
Quelle: SAP (2014)

# Ideate



- Der Übergang von der Problemsicht zur Lösungssicht
- Nachdem man das Problem (auch aus User-Sicht) verstanden hat, kann man Lösungsansätze (gemeinsam!) entwickeln
- Brainstorming: alt, aber noch immer nützlich
- Ideen clustern und priorisieren

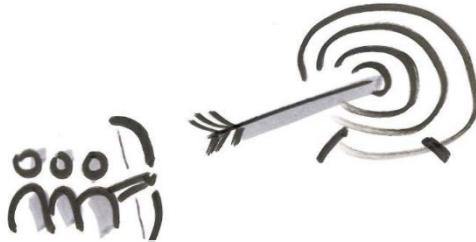
# Ideate: Lösungsansätze entwickeln



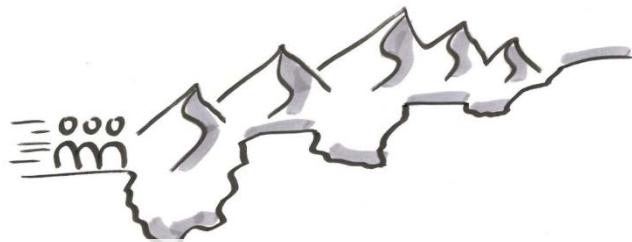
BUILD ON THE IDEAS OF OTHERS



DEFER JUDGEMENT



STAY FOCUSED ON TOPIC



FAIL EARLY AND OFTEN



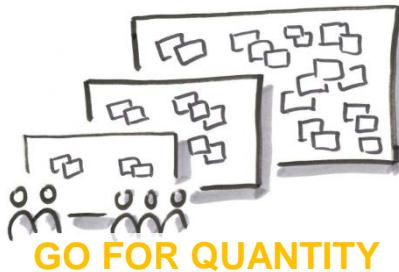
BE VISUAL



ONE CONVERSATION AT A TIME



THINK USER-CENTRIC

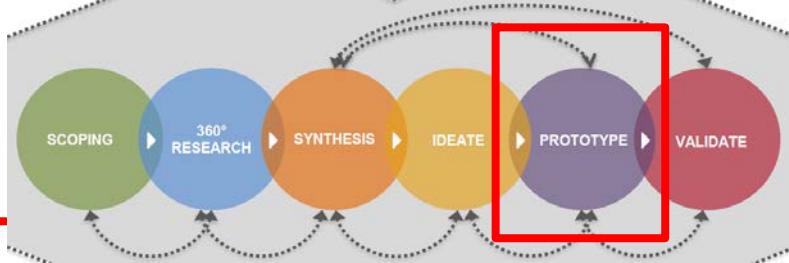


GO FOR QUANTITY



GO FOR WILD IDEAS

# Prototype

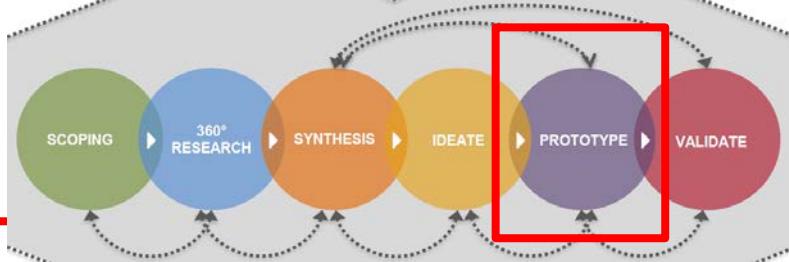


## Ein Prototyp:

- Ist ein erstes vorläufiges Modell
- Zeigt, wie sich die Lösung anfühlt, aussieht, funktioniert, macht sie greifbar
- Wird gebaut, um die Reaktion anderer auf den Lösungsansatz zu testen

**Das Scheitern eines Prototyps ist  
ein einfacher Weg Dinge zu  
verstehen und zu verbessern!**

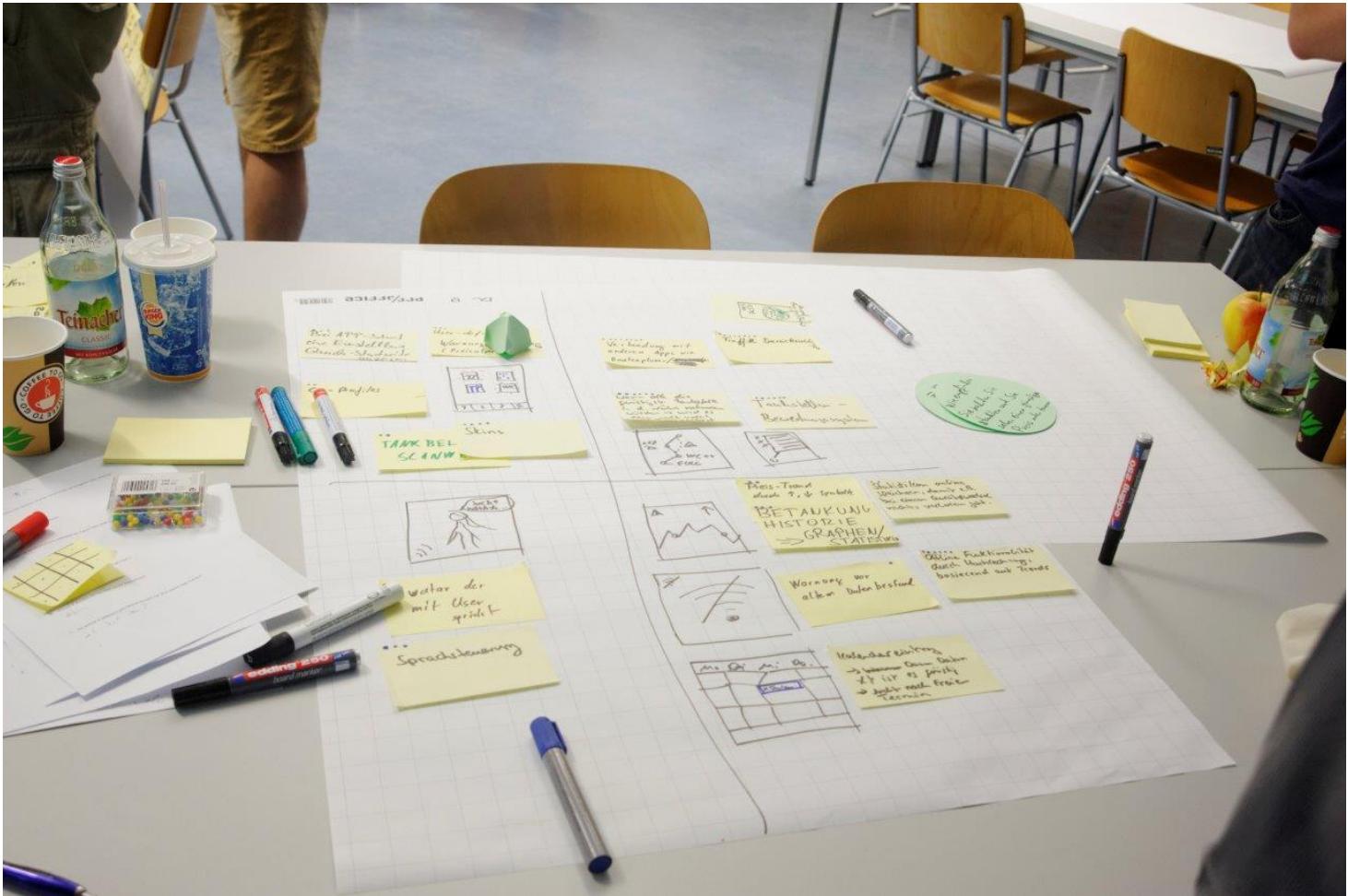
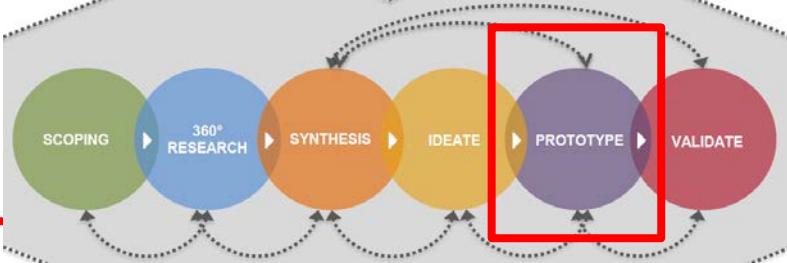
# Prototype



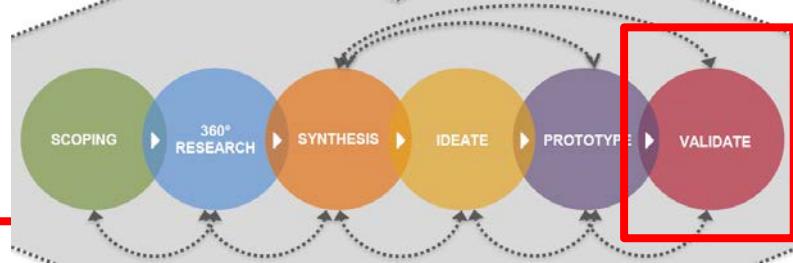
Kann sein:

- Ein physischer Prototyp
- Ein Paper-Prototyp (z.B. Mock-ups)
- Storyboards (manchmal auch als Rollenspiel vorgestellt)
- Ein technischer Prototyp (Software)
- ...

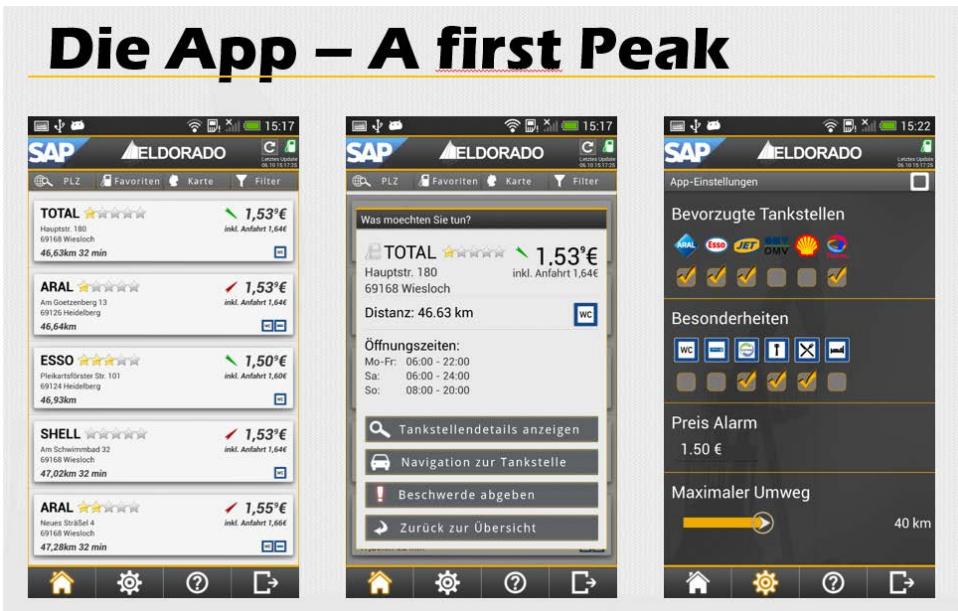
# Prototype



# Validate



## Die App – A first Peak



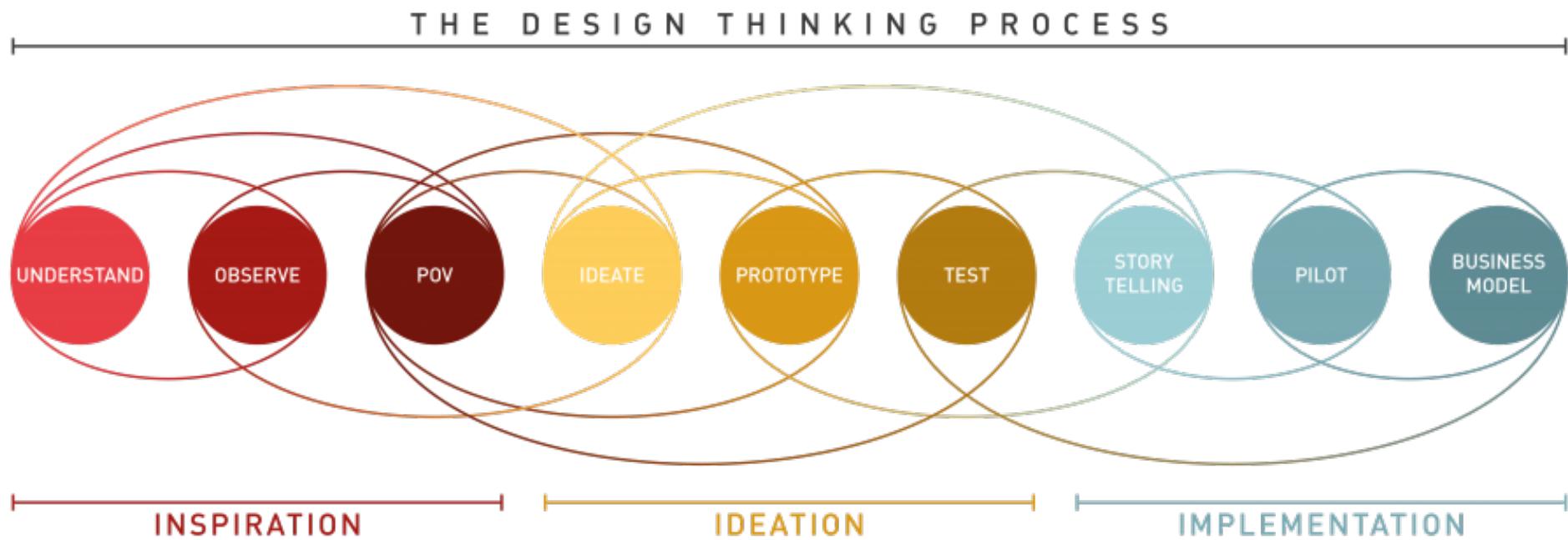
## Cached-Offline Results

- Permanente Datenverfügbarkeit
- Indikator bei veralteten Daten



- Keine Innovation ohne Implementierung!
- Design Thinking und Lean Development ergänzen sich wie Ying und Yang.
- Hier kommen die agilen Vorgehenskonzepte ins Spiel.

# Design Thinking Prozess II



## Beispiele

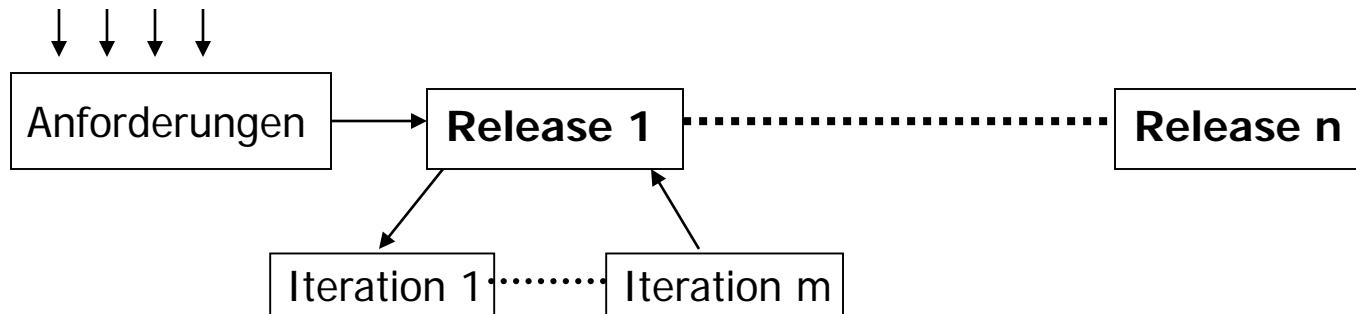
- eXtreme Programming
- Scrum
- Kanban

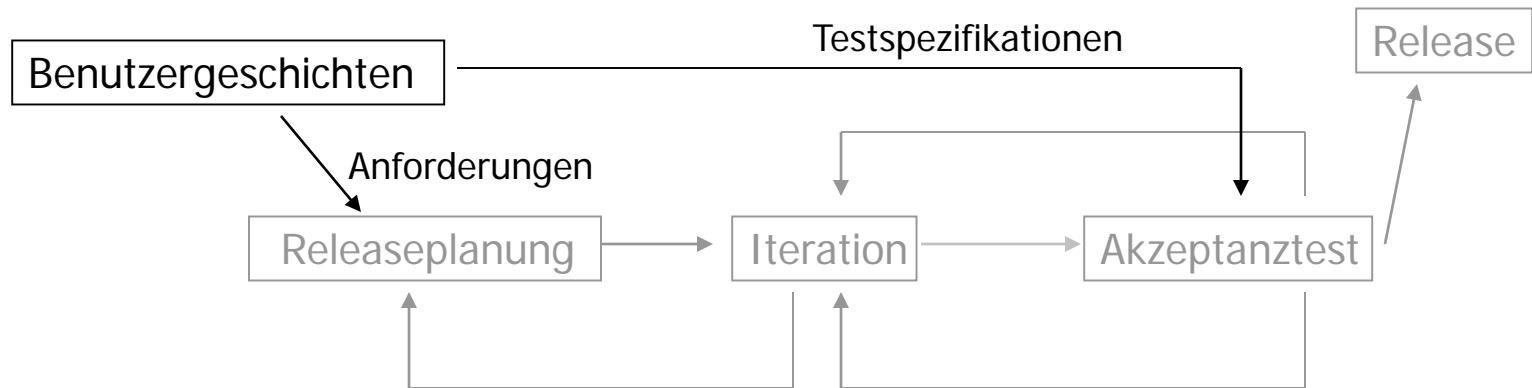
# eXtreme Programming (XP)

eXtreme Programming geht auf Entwicklungen von Kent Beck, Ward Cunningham und Ron Jeffries zurück

## Was kennzeichnet XP?

- Der Kunde ist ein Team-Mitglied!
- Kontinuierliche Reviews durch Programmieren in Paaren
- Kontinuierliches Testen nach jeder Änderung des Systems
- Kontinuierliches Design und Redesign („Refactoring“)
- Kontinuierliches Feedback durch die kurzen Release-Zyklen und ständiges Einbeziehen des Auftraggebers

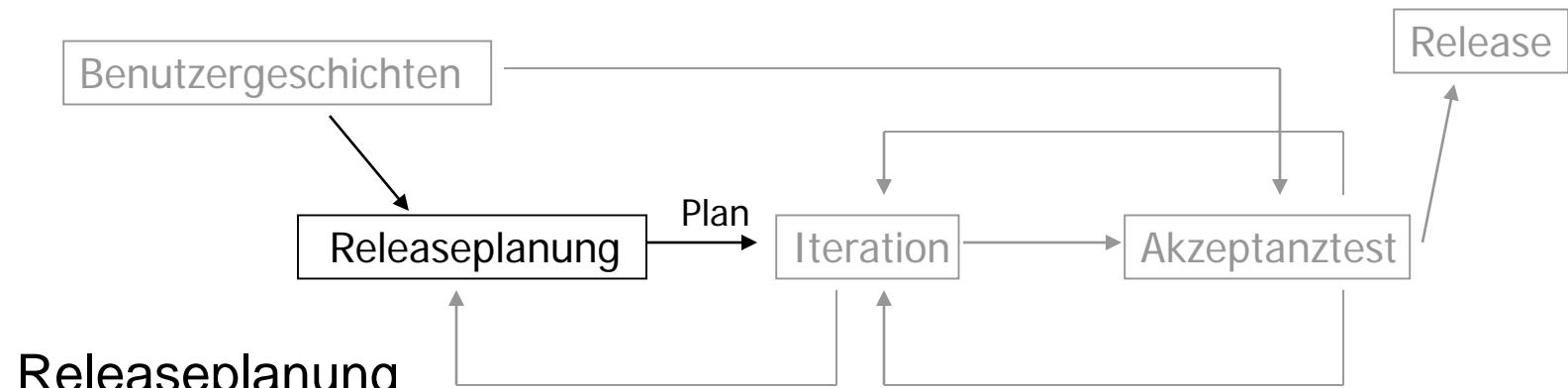




## Benutzergeschichten

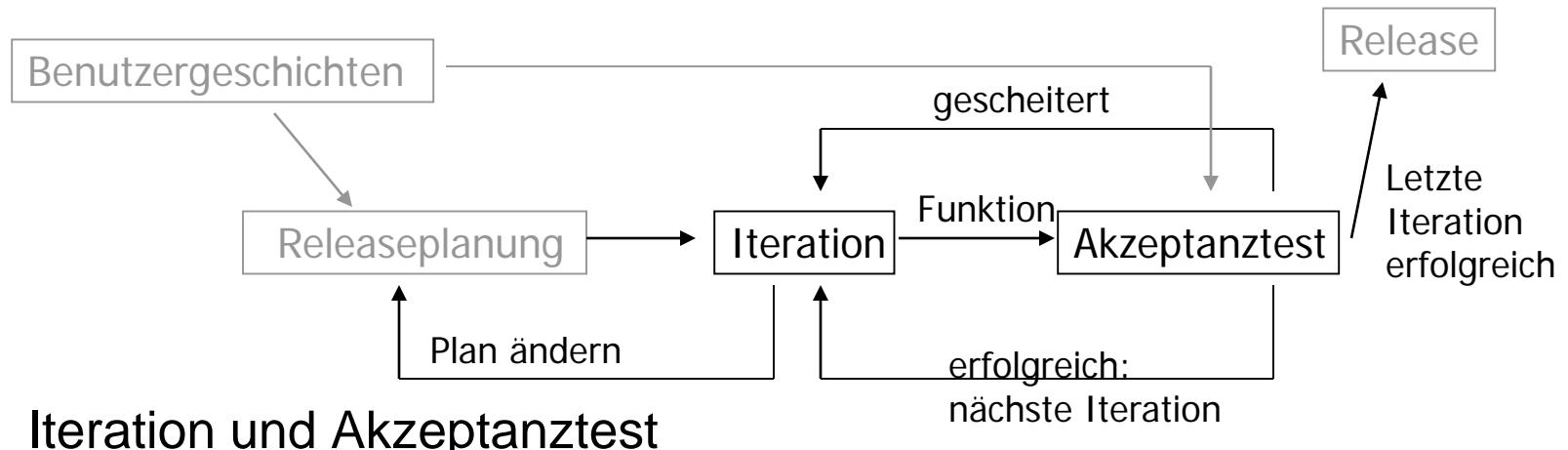
- Mit dem Kunden erfasste Anforderungen
- Jeweils eine Funktionalität
- Der dazugehörige Akzeptanztest wird sofort mitspezifiziert
- Aufwand für Programmierung und Test: ca. 2-3 Tage
- Aufgaben und deren Aufwand für eine Benutzergeschichte schätzen

# eXtreme Programming (II)



## Releaseplanung

- Planung eines Releasezyklus, Ergebnis ist lauffähige Software
- Zeitrahmen: 3-4 Monate
- Ressourcen: Programmierer
- Qualität: Durch Akzeptanztests definiert
- Zu klären: Funktionsumfang?
  - Der **Kunde wählt Benutzergeschichten** aus, die im Releasezyklus implementiert werden sollen
  - Die **Summe der Aufwände** der Benutzergeschichten darf nicht Entwicklungskapazität übersteigen!
  - Kunde wird also Benutzergeschichten mit dem für ihn **größten Nutzwert** wählen



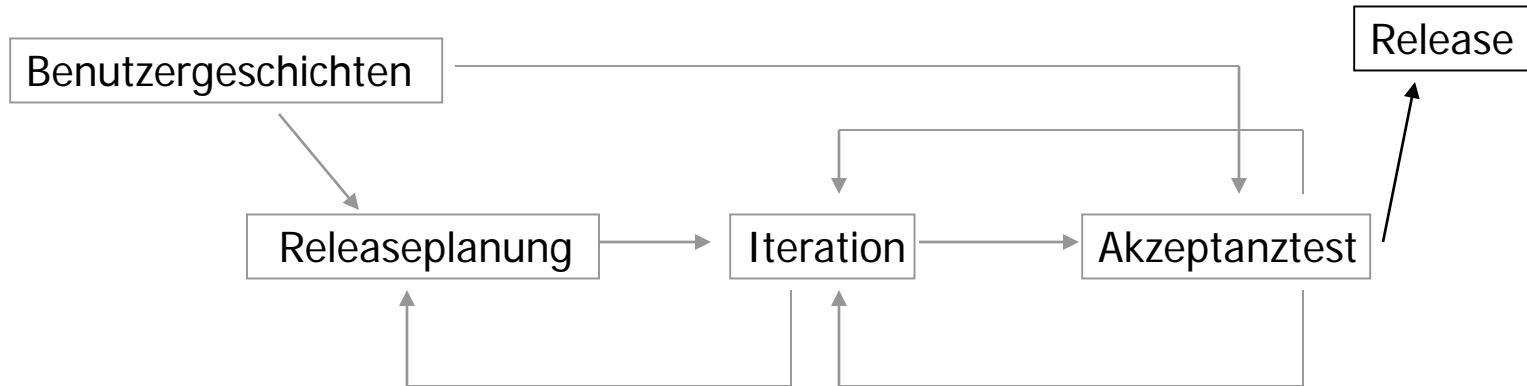
## Iteration und Akzeptanztest

- Jede Iteration **während** des Releases: 2-3 Wochen
- Genaue Planung der Benutzergeschichten und Aufgaben für die Iteration
- Tests nach jedem Einspielen einer neuen Funktionalität im System

## Programmieren in Paaren

- Für jede Aufgabe ein **Paar!** Aufwand 2 Stunden bis 2 Tage
- Zuerst Komponententest erstellen
- Danach Programmierung bis zum erfolgreichen Test
- Anschließend Test (Quellcode leserlich gestalten)

# eXtreme Programming (IV)



- Zusammenführung (Einspielung ins System, Akzeptanztest) an einer **zentralen Stelle**, damit immer nur eine Änderung eingespielt wird!
- Auch alle bereits erfolgreichen Akzeptanztests müssen **nach dem Einspielen einer neuen Funktion erneut** durchgeführt werden!

Ergebnis des Releasezyklus:

- Auftraggeber erhält ein **lauffähiges System**, mit einer garantierten (da wiederholt getesteten Funktionalität), die er selbst bestimmt hat.

Nebeneffekt:

- Sogar **jede Iteration ist lauffähig!** Auftraggeber kann dadurch Projektfortschritt verfolgen!

- Chrysler Comprehensive Compensation
- Neues System für Gehaltsabrechnung
- Der Gebrauch einer Metapher als Ausgangspunkt
  - *Payroll is an assembly line*
  - Gebrauch von Konzepten aus der Fertigung
    - Bänder (lines)
    - Stationen (stations)
    - Behälter (bins)
    - Werkstücke (parts)

# **Nahezu XP (I)**

## **(nach Alister Cockburn)**

---

**Zu Extreme Programming gehören (nach dem C3-Projekt) vier Dinge:**

- Man programmiert paarweise.
- Man liefert alle drei Wochen ein Inkrement.
- Man hat die ganze Zeit einen Anwender im Team.
- Man hat Regressionstests, die immer erfolgreich durchlaufen.

**Zur Belohnung für diese vier Dinge:**

- Muss man den Code nicht kommentieren.
- Muss man keine Anforderungs- oder Designdokumentation erstellen.

# **Nahezu XP (II)**

## **(nach Alister Cockburn)**

---

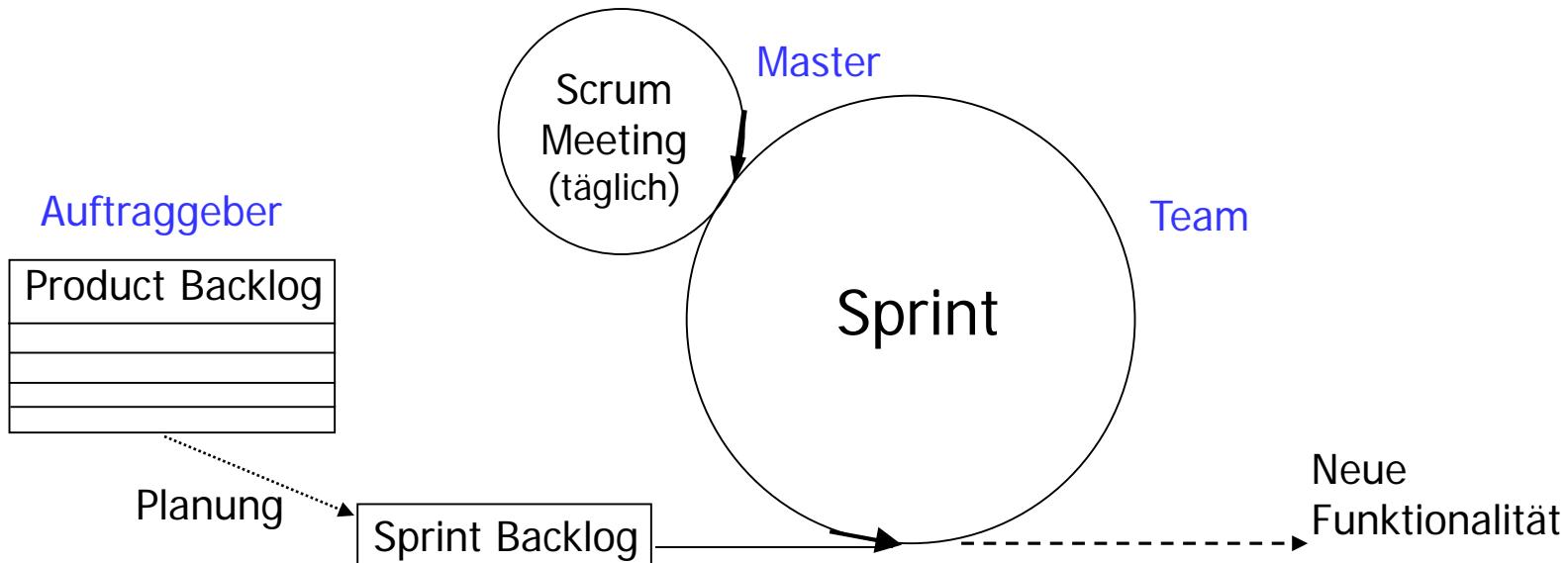
**In unserem Projekt sind wir ziemlich dicht dran:**

- Ok, einige unserer Leute sitzen im Erdgeschoss, ein paar im 5. Stock und ein paar arbeiten etwa 2 Stunden entfernt von hier; somit wird bei uns nicht wirklich paarweise programmiert,
- und genau genommen liefern wir unsere Inkremente alle 4-6 Monate aus,
- und bei uns ist auch kein Anwender in Sicht,
- und wir haben auch keine Unit-Tests.

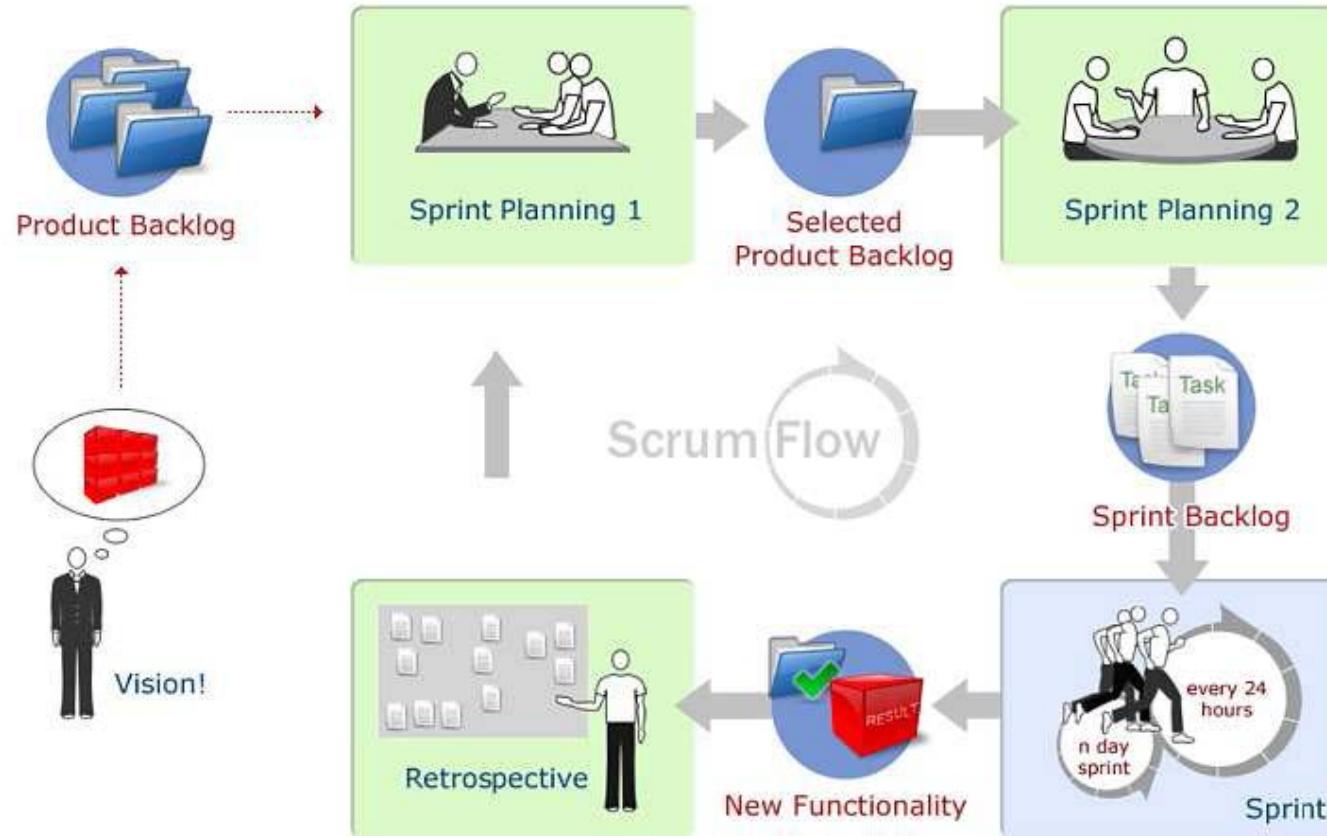
**Aber: Wir haben zumindest keine Designdokumentation und kommentieren unseren Code auch nicht. Man kann also durchaus davon sprechen, dass wir „nahezu XP“ machen.**

# Scrum: Struktur

- Entwickelt von Jeff Sutherland und Ken Schwaber
- (*living*) Backlog: Anforderungen an Produkt
- Team: ca. 7 Personen, selbstorganisierend, autonom!
- Master: Schnittstelle Team -> Management, hält äußere Einflüsse vom Team fern!
- Sprint: 30 Tage, Sprint Goal muss erreicht werden!
- Meeting: ca. 15 Minuten, über Projektstand informieren
- Planung: Immer nur ein Sprint

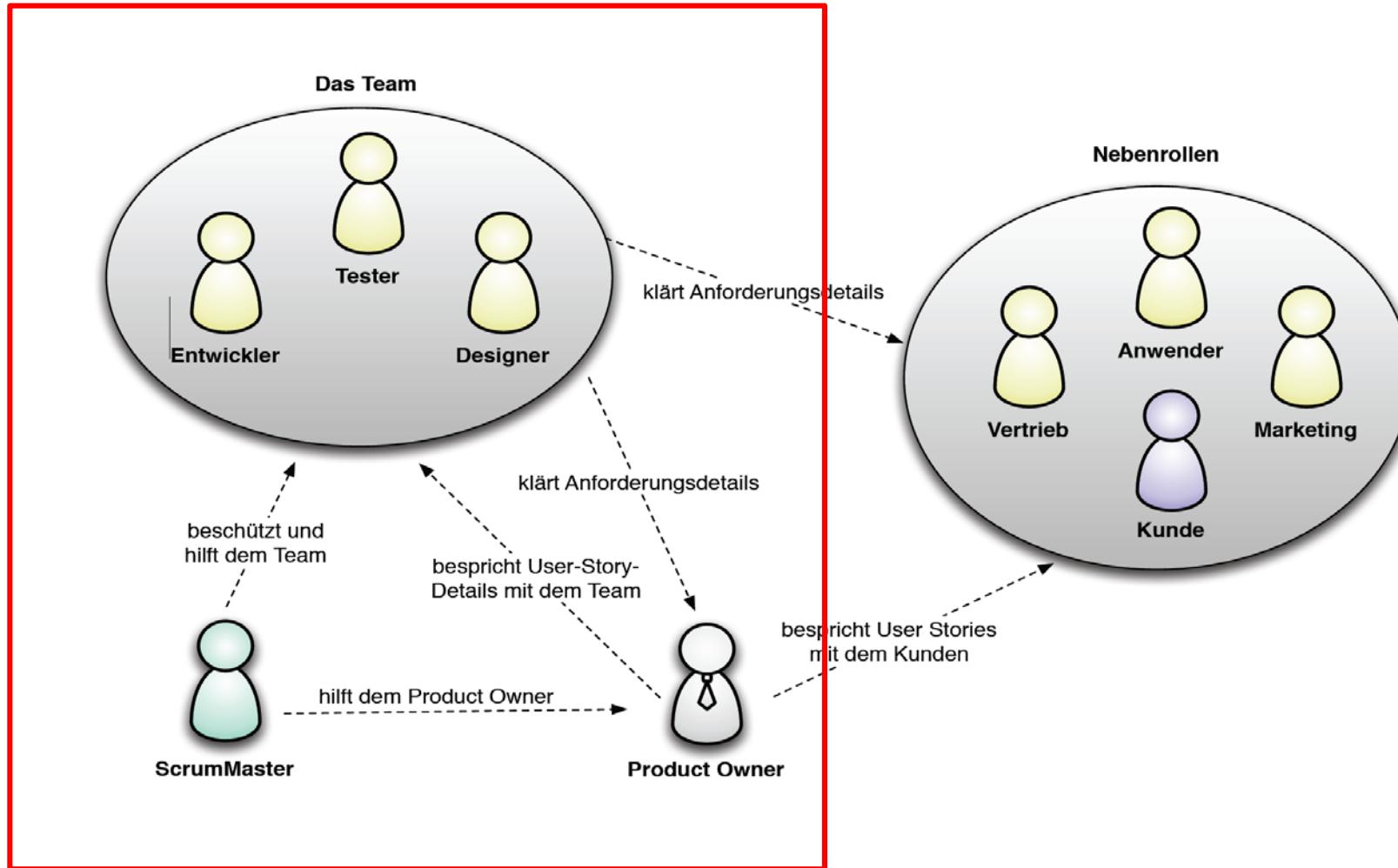


# Scrum: Der Prozess



Quelle: Gloger (2008)

# Scrum: Rollen und ihr Zusammenspiel



Quelle: Wirdemann (2017)

# Die Zusammensetzung des Teams

Da das Team für die Umsetzung der Anforderungen verantwortlich ist, sollte gewährleistet sein, dass sich alle Teammitglieder aktiv einbringen und Tasks übernehmen. Bei der Zusammenstellung der Teams sollte auf folgende Punkte geachtet werden:

**Multidisziplinarität:** Das Team ist im Idealfall multidisziplinär (i.e. cross-functional) zusammengesetzt. Es hat die Kompetenz, die anstehenden Aufgaben zu erledigen. Es besteht nur aus Experten für dieses Problem, aber alle im Team wollen die notwendigen Kenntnisse gegebenenfalls erwerben.

**Verantwortlichkeit:** Das Team ist verantwortlich für die Lieferung. Es führt alle notwendigen Schritte für die Erreichung des *Sprint Goals* durch. Es hat auch die Autorität, alle notwendigen Maßnahmen durchzuführen.

**Teamzusammenstellung:** Ein Team wird nicht nur aus Menschen aus unterschiedlichen Disziplinen gebildet, sondern auch aus Mitarbeiterinnen und Mitarbeitern mit verschiedenen Erfahrungen.

**Selbstorganisation:** Das Team ist für alle Aktionen innerhalb des Teams verantwortlich. Die Teammitglieder müssen ihre Aufgaben untereinander selbst verwalten und miteinander die Durchführung der Aufgaben absprechen. Selbstorganisation bedeutet nicht, jeder kann machen was er will, sondern jeder macht, was miteinander vereinbart wurde.

# Typische Phasen der Teambildung



## Phasen im Team-Building nach Bruce W. Tuckman:

- 1. Forming:** Teamgründung, Kennenlernen, Gruppengefühl bildet sich, erste Regeln werden festgelegt, Motivation ist hoch.
- 2. Storming:** Erste Konflikte tauchen auf, oftmals Gruppenbildung, Ziele und Prioritäten werden hinterfragt.
- 3. Norming:** Regel für Zusammenarbeit werden aufgestellt, engere Beziehungen zwischen den Teammitgliedern entwickeln sich („Wir-Gefühl“ nimmt zu).
- 4. Performing:** Team ist immer besser eingespielt (intuitive Zusammenarbeit), Effektivität steigt, jeder weiß, was der/die andere kann.

Quelle: Tuckman (1965)

# Die Aufgaben des Teams

Das Team verständigt sich im sogenannten *Sprint Planning* mit dem Product Owner auf das Ziel des nächsten Sprints (*Sprint Goal*). Hier werden aus dem *Product Backlog* diejenigen User Stories ausgewählt, die das Team im kommenden Sprint umsetzen will. Wie das Team die User Stories umsetzt und wer dabei welche Einzelaufgabe (*Task*) übernimmt, bleibt dem Team selbst überlassen. Bei der Umsetzung ist es den Teammitgliedern in Einzelfällen auch möglich mit den jeweiligen Fachbereichen / Fakultäten, aus denen die Anforderungen stammen, unklare Punkte zu besprechen.

Am Ende des Sprints präsentiert das Team seine Ergebnisse in Form der vollständig umgesetzten User Stories. Nur zum Teil abgeschlossene User Stories werden nicht vorgestellt.

Während des gesamten Projektes weist das Team auf bestehende Hindernisse (sogenannte *Impediments*) hin. Adressat ist hier vor allem der Scrum Coach, der diese *Impediments* auflösen oder beseitigen soll.

In der *Retrospective* analysiert das Team den Ablauf des letzten Sprints und diskutiert, was gut gelaufen ist, was eher schlecht war und wie man seine Performance im nächsten Sprint verbessern kann.

# Product Owner

---

Der Product Owner (kurz PO) ist letztlich verantwortlich für das Ergebnis. Er stellt die Schnittstelle zwischen dem „Kunden“ und dem Projektteam dar. Auf der einen Seite nimmt er die Anforderungen der Fachabteilungen auf und formuliert gemeinsam mit diesen User Stories, die eben diese Anforderungen abbilden. Jedes Scrum-Team hat genau einen Product Owner.

Der Product Owner ist die zentrale Schaltstelle, wenn es um die Anforderungen des zu entwickelnden Produkts geht. Er repräsentiert sämtliche Stakeholder, die in irgendeiner Form Interesse am Projekt und Einfluss auf die Anforderungen haben. Alle Anforderungen laufen beim Product Owner zusammen und werden von ihm im *Product Backlog* gesammelt und priorisiert. Der Product Owner muss die volle Autorität des Kunden besitzen. Er muss im *Sprint Planning Meeting* selbstständig und ohne Rückfrage entscheiden können, welche Story wichtig ist.

Der Product Owner schreibt die User Stories. Gemeinsam mit den Stakeholdern arbeitet er an den initialen und sich verändernden Anforderungen des Projekts und macht daraus geeignete Stories. Er ist der Eigner und Chef des Product Backlog.

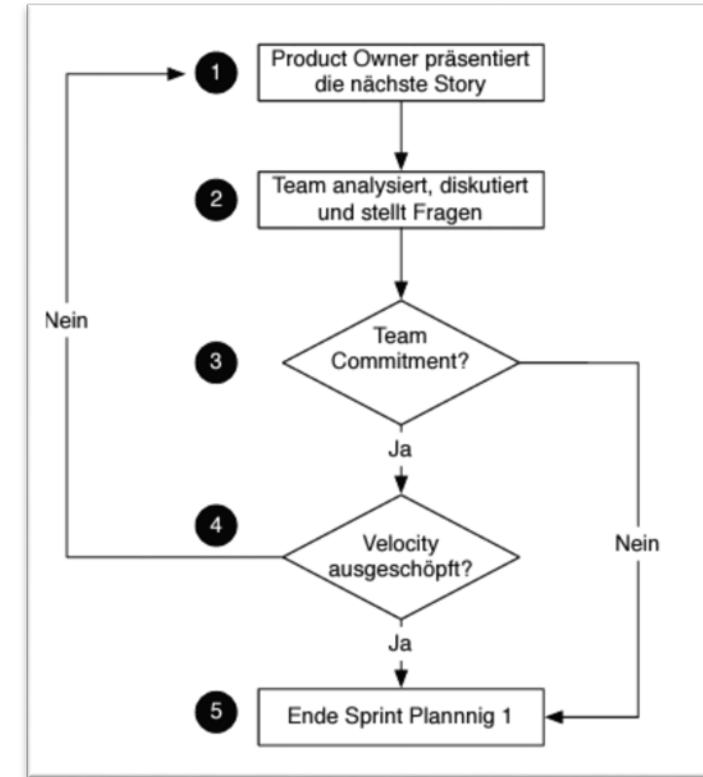
Alle dürfen das Backlog lesen, aber nur der Product Owner darf es verändern. Er tragt die User Stories ins Backlog ein und splittet sie in kleinere, konkrete Stories auf, wenn sie zu groß für die Umsetzung im Rahmen eines Sprint sind (sogenannte Epics). Der Product Owner ist für die Priorisierung des Backlog zuständig.

In Zusammenarbeit mit den Stakeholdern bewertet er die Stories hinsichtlich ihres Mehrwerts für die Organisation. Er sorgt dafür, dass immer die wichtigsten Stories als Nächstes umgesetzt werden.

- Der Scrum Master ist für den Erfolg des Vorgehensmodells verantwortlich. Er sorgt dafür, dass das Vorgehensmodell funktioniert und die zugrundeliegenden Regeln und Prinzipien eingehalten werden.
- Sorgt für optimale Arbeitsbedingungen für das Team (impediments auflösen!)
- Funktion eines Coaches (für Team und PO!)
- Ist stets für alle Projektbeteiligten ansprechbar

# Scrum: Meetings (I)

- Estimation Meetings (auch Pre-planning oder Release-planning Meeting):
  - Schätzung der Backlog-items
  - Überprüfung der Architektur
  - Dauer max. 90 Minuten
- Sprint Planning:
  - Sprint Planning 1:
    - Auswahl von Backlog-items
    - Definition des Sprint-goal
    - Dauer ca. 4 Stunden
  - Sprint Meeting 2:
    - Design: Wie soll Sprint-goal erreicht werden?
    - Task-list wird erstellt → Sprint Backlog
    - Dauer ca. 4 Stunden



Quelle: Gloer (2008)

- Daily Scrum:
  - Tägliche Abstimmung der Teammitglieder
  - Regeln: absolut pünktlicher Beginn und max. 15 Minuten Dauer
- Sprint Review:
  - „Deadline“
  - Nur „usable software“ darf präsentiert werden
  - Dauer max. 90 Minuten
- Sprint Retrospective:
  - Aus eigenen Erfahrungen lernt man am besten!
  - Scrum setzt auf kontinuierliches Verbessern!
  - Impediment Backlog

Quelle: Gloger (2008)

- **Product Backlog:**  
Liste von zu liefernden Funktionalitäten (Product Backlog Items), die vom Product Owner priorisiert werden müssen.
- **Selected Product Backlog:**  
In Sprint Planning Meeting 1 ausgewählte Backlog Items, die bis zum Ende des Sprint realisiert werden sollen.
- **Sprint Backlog:**  
Task-Liste aus dem Sprint Planning Meeting 2; wird täglich überarbeitet und aktualisiert.
- **Impediment Backlog:**  
Liste aller Blockaden, die einem Team aus dem Weg geräumt werden müssen, damit es produktiver werden kann; „Risikomanagement“

Quelle: Gloger (2008)

- **Defintion of Done**
  - Bekanntes Artefakt im Umfeld der agilen Methoden
  - Was muss erfüllt sein, damit einer User Story / ein Feature als erledigt/fertig/abgearbeitet eingestuft werden kann?
  - Z.B.: Coding fertig, Dokumentation liegt vor, Tests durchlaufen,....
- **Definition of Ready**
  - Neues Artefakt
  - Umstritten, da für einige nicht mit agilen Prinzipien vereinbar
  - Welche Anforderungen sind an eine User Story zu stellen, um sie in einen Sprint aufzunehmen?
  - Minimale Qualitätsanforderungen an die User Stories

# Definition of Done (Beispiele)

---

Für ein Software-Entwicklungsprojekt mit Änderungen am Code

- Akzeptanztests funktionieren
- Unit Tests sind erstellt und funktionieren
- Coding Guidelines und Standards sind eingehalten
- Code Review ist erfolgreich durchgeführt
- Code ist ins Repository eingecheckt
- Dokumentation ist angepasst

Bei der Erstellung von Dokumentation

- Die Dokumentation erfüllt formale und inhaltliche Vorgaben
- Die Dokumentation wurde peer reviewed & freigegeben
- Versionsnummer des Dokuments ist aktualisiert

Quelle: Heberle (2019)

„Eine User Story beschreibt eine Anforderung an ein Softwaresystem. Die Anforderung besitzt einen konkreten und sichtbaren Mehrwert für den Kunden.“

User Story:

„Als Anwender möchte ich mein Profil pflegen können“  
=> Neue Funktionalität! => Mehrwert beim Kunden.

Keine User Story:

„Die Anwendung soll in Java programmiert werden.“  
=> Anforderung schafft keinen Mehrwert im Sinne des Geschäftsmodells des Kunden.

Quelle: Wirdemann / Mainusch (2017)

## Grundmuster für das Formulieren einer User Story nach Cohn:

**WER?**

**WAS?**

**WARUM?**

Als <Benutzerrolle> will ich <das Ziel>, so dass <Grund für das Ziel>

Als **Studierender**

**will ich die  
Professoren  
bewerten,**

**so dass durch das Feedback  
die Lehre verbessert werden  
kann.**

Quelle: Wirdemann / Mainusch (2017)

# INVEST-Merkmale für User Stories nach Cohn

- **Independent (Unabhängig):**  
User Stories sollen unabhängig voneinander sein. Idealerweise sollten sie in beliebiger Reihenfolge umgesetzt werden können; Abhängigkeiten führen zu Risiken.
- **Negotiable (Verhandelbar):**  
Eine User Story sollte so formuliert sein, dass sie einen Gestaltungsspielraum enthält, über den in den Planungssitzungen verhandelt werden kann. Dies erhöht die Produktivität des Entwicklungsteams, da hier die Kreativität des Teams mit eingebracht werden kann.
- **Valuable (Nützlich):**  
Eine User Story sollte so formuliert sein, dass der konkrete Nutzen erkennbar ist.
- **Estimatable (Schätzbar):**  
Eine User Story muss verständlich formuliert sein, sodass sie vom Team geschätzt werden kann.
- **Small (Klein):**  
User Stories sollten in kleinen Paketen beschrieben werden. Komplexe Anforderungen sollten weiter unterteilt werden.
- **Testable (Testbar):**  
Alle User Stories sollten testbar sein. Dazu sollten Akzeptanzkriterien zu jeder Anforderung vorliegen

Quelle: Wirdemann / Mainusch (2017)

# Agiles Schätzen

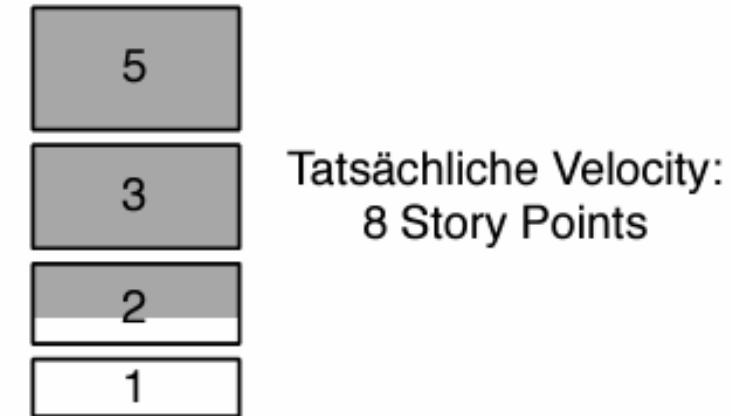
---

- Beim agilen Schätzen geht es im Gegensatz zu vielen anderen Schätzverfahren nicht um die Entwicklungsdauer, sondern ausschließlich um relative Größe.
- Idee: Auch mit relativ geringem Zeitaufwand lassen sich Schätzungen erstellen, die ähnlich gut sind, als hatte man deutlich mehr Zeit investiert.
- Letztendlich ist agiles Schätzen nichts anderes als das Ordnen von User Stories nach Größenklassen.
- Einheit der Schätzung: Story Points.
- Agiles Schätzen und Planen trennt die Themen Größe und Dauer voneinander: Größe wird geschätzt, und Dauer wird abgeleitet.
- Spielend schätzen: Planning Poker, Magic Estimation, T-Shirt Size

- Grundlage für die Sprint-Planung
- Übergang vom Schätzen der relativen Größe zur Dauer
- Wie viele Story Points kann das Team im kommenden Sprint abarbeiten?
- Velocity ist ein sich selbst adaptierender Wert, der von Sprint zu Sprint genauer wird.

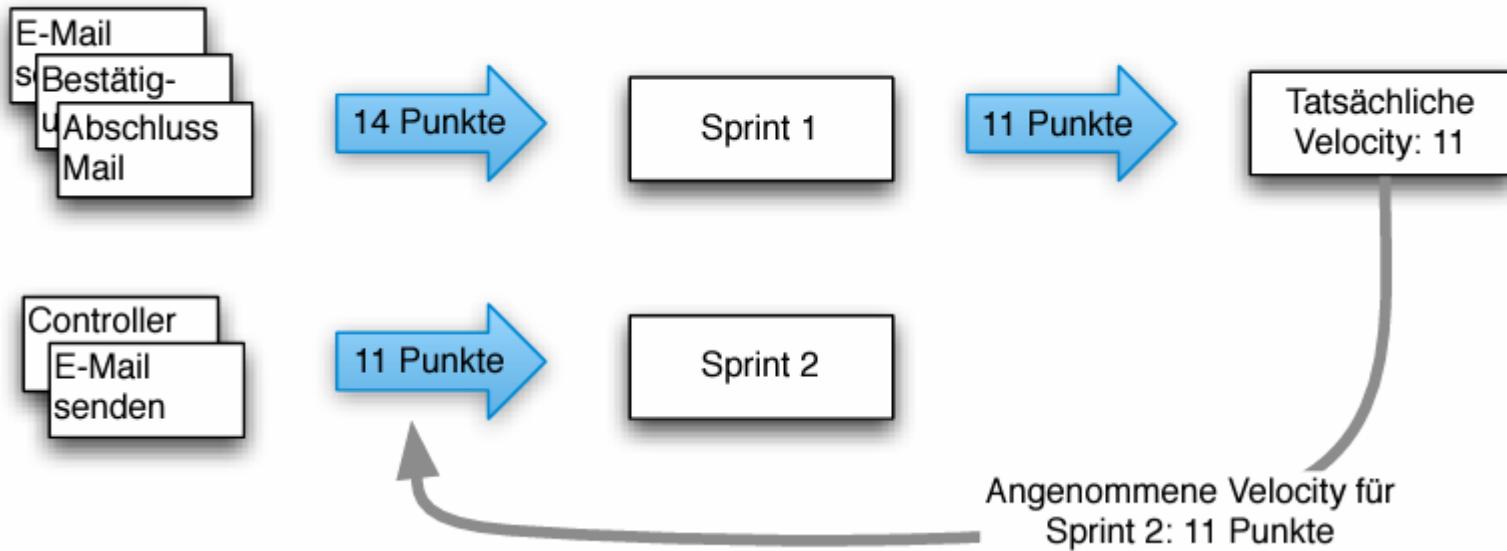
# Berechnung der tatsächlichen Velocity

Beginn des Sprints → Ende des Sprints



Quelle: Wirdemann / Mainusch (2017)

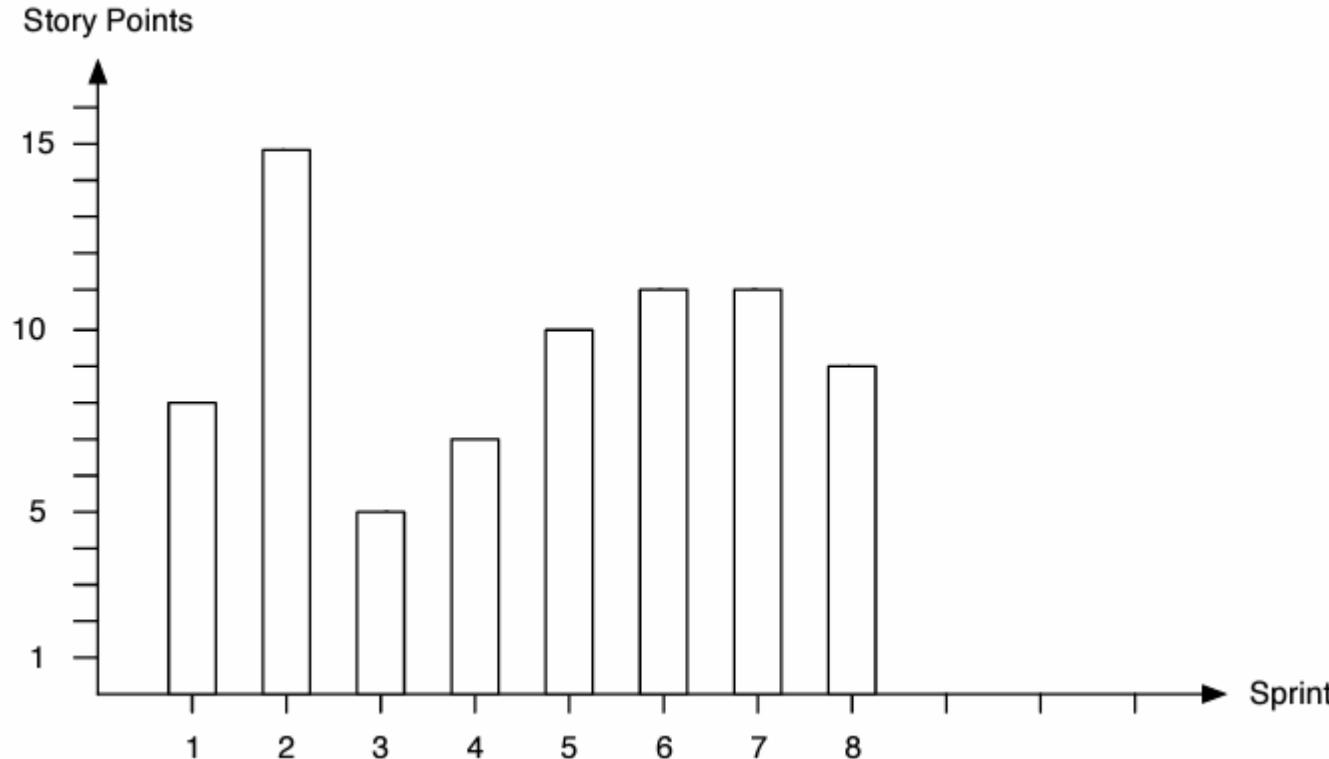
# Velocity-Lernkreislauf



Übernommene Velocity:  
Angenommene Velocity = Tatsächliche Velocity

Quelle: Wirdemann / Mainusch (2017)

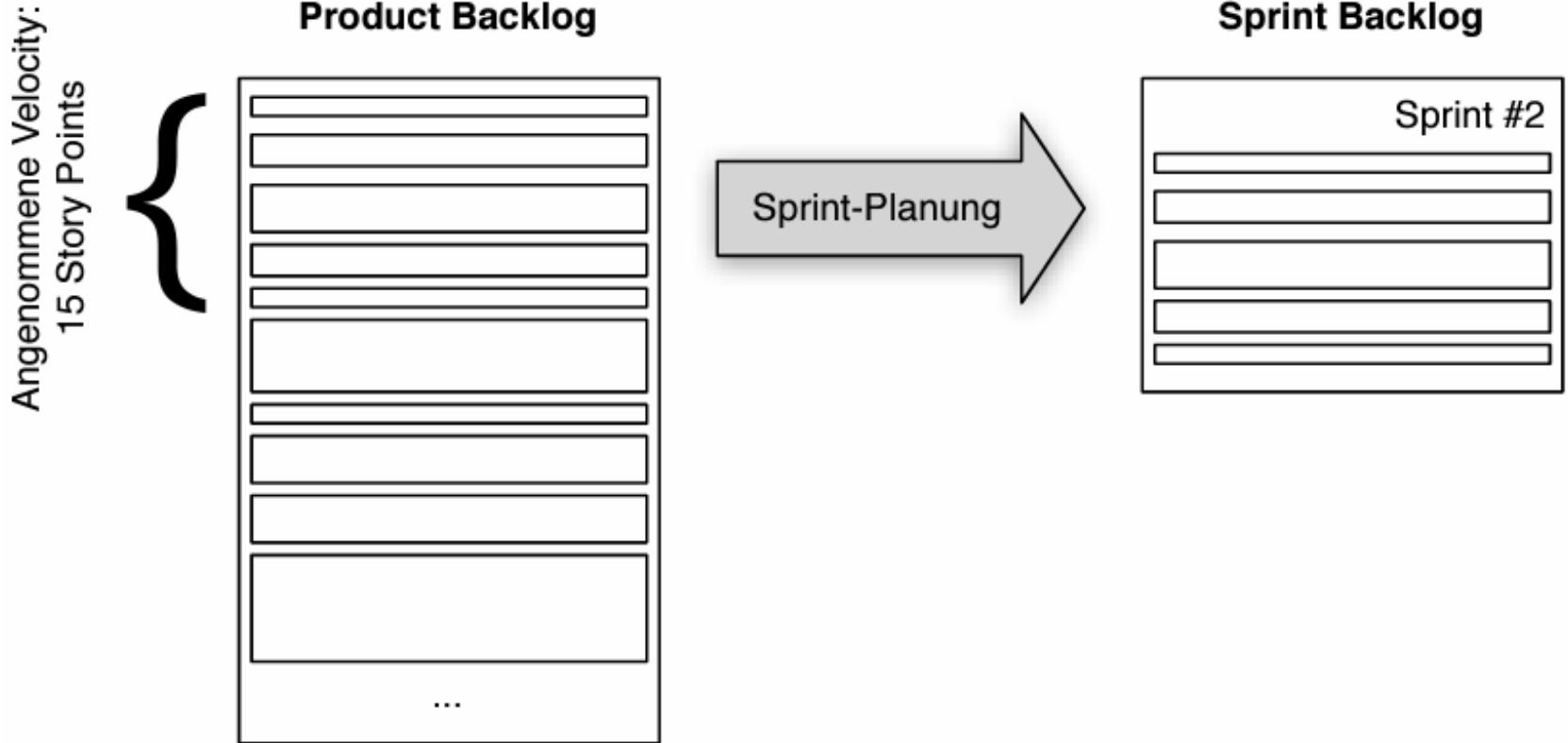
# Velocity-Chart: Tatsächliche Velocity pro Sprint als Planungsbasis



Mittlere Velocity:  
Berechnung des Velocity-Medians

Quelle: Wirdemann / Mainusch (2017)

# Velocity-basierte Sprint-Planung

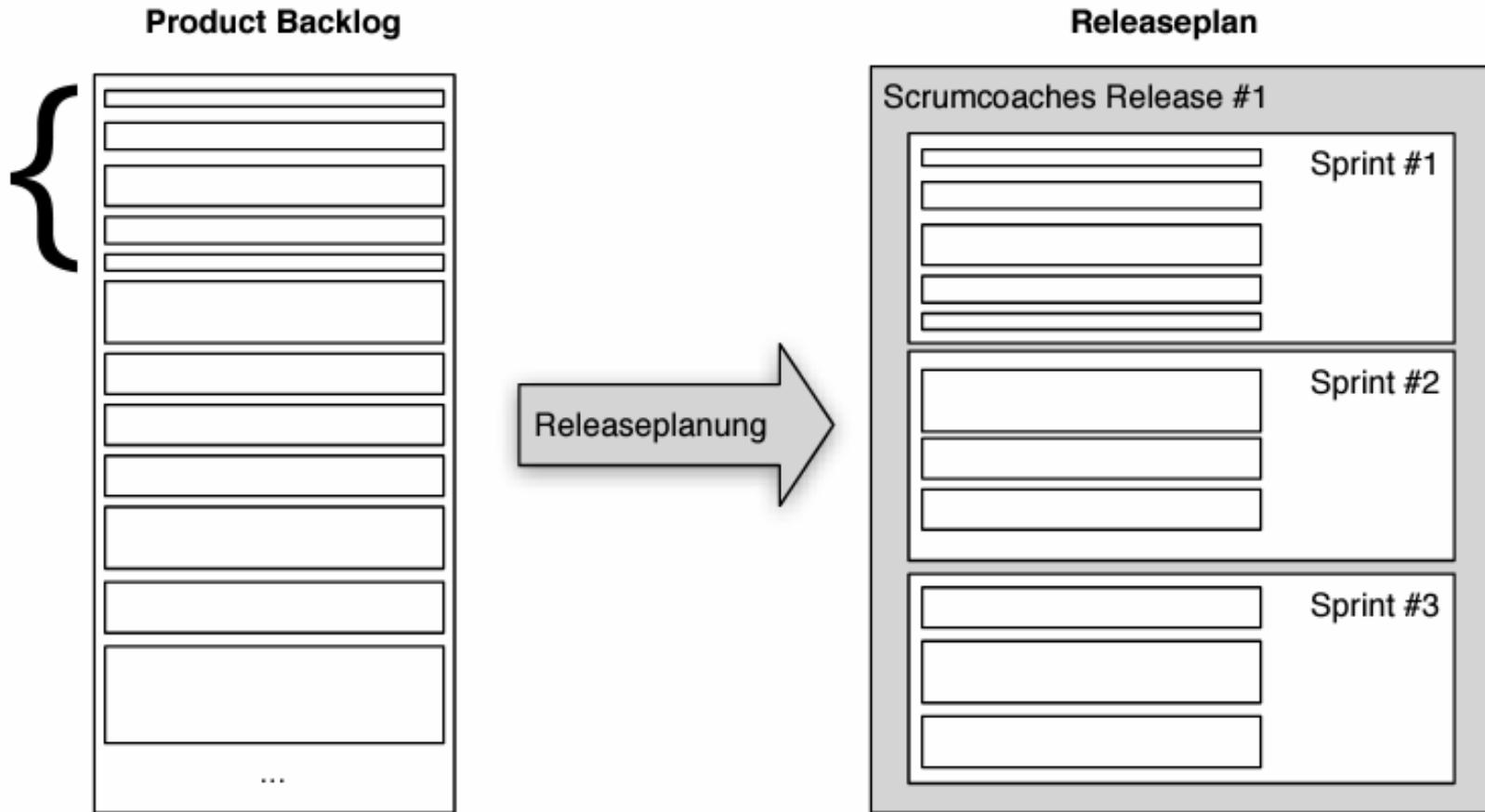


Quelle: Wirdemann / Mainusch (2017)

# Velocity-basierte Release-Planung



Angenommene Velocity:  
15 Story Points



Quelle: Wirdemann / Mainusch (2017)

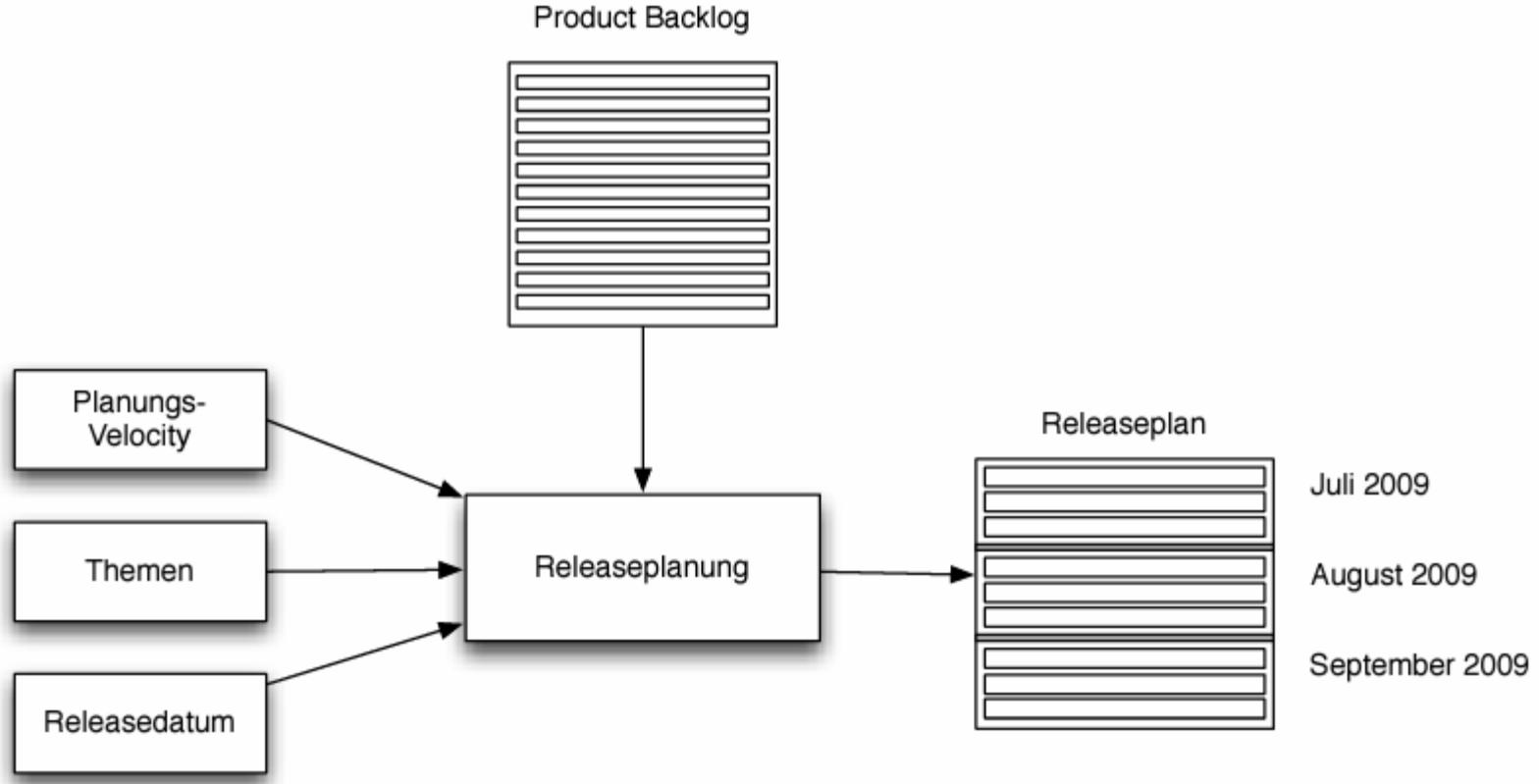
# Releaseplanung in Scrum (I)

---

- Der Releaseplan wird vom Product Owner erstellt und enthält ein oder mehrere Releases.
- Der Releaseplan ist ein öffentliches Dokument, das von anderen Personen als nur dem Scrum-Team eingesehen wird.
- Wann immer es geht, sollte ein Releaseplan frühestens nach drei Sprints erstellt werden. Die drei regulären Sprints werden dadurch zu Velocity-Test-Sprints erklärt.
- Ein Releaseplan erzeugt immer eine Erwartungshaltung beim Kunden, und egal, wie agil der Plan und der Kunde auch sind, es ist nie schön, die einmal erzeugten Erwartungen zu enttäuschen.

Quelle: Wirdemann / Mainusch (2017)

# Releaseplanung in Scrum (II)



Quelle: Wirdemann / Mainusch (2017)

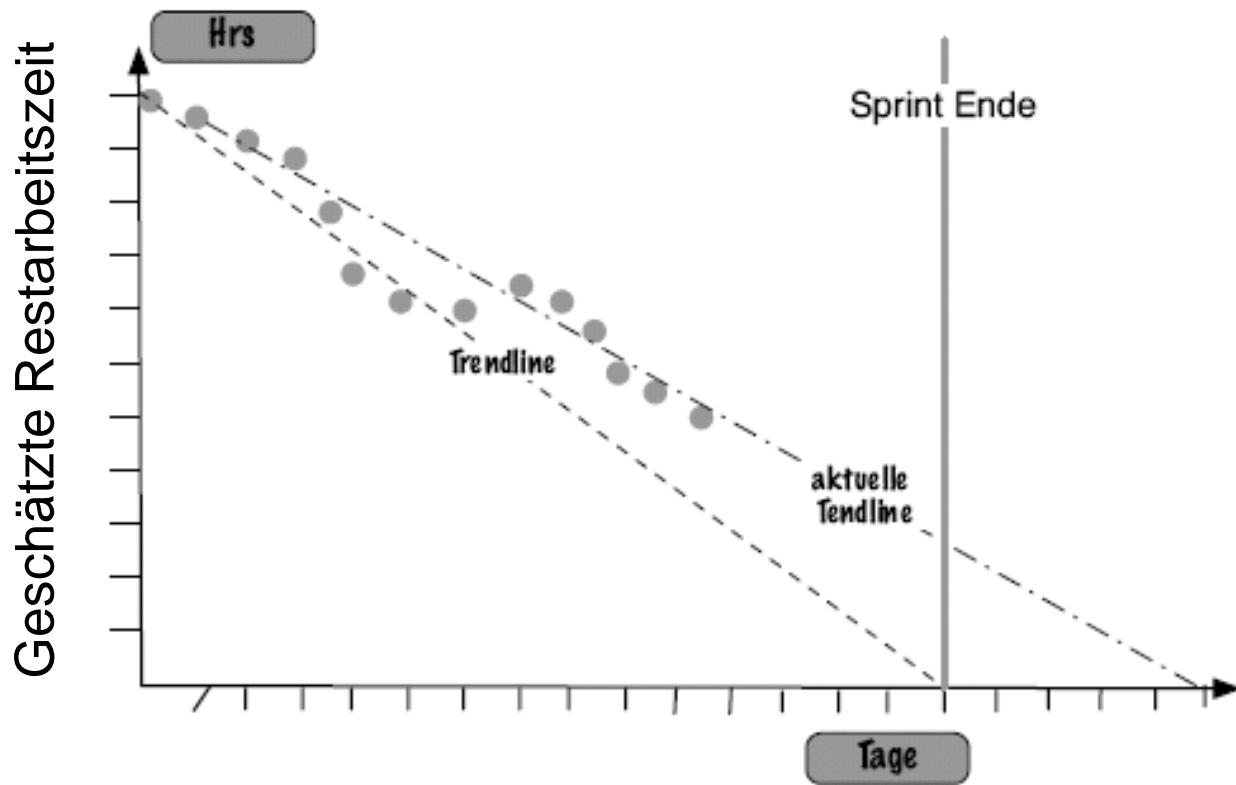
# Releaseplanung in Scrum (III)

---

- Es gibt „Cone of Uncertainty“: Der geschätzte Aufwand eines Projekts weicht um 60% bis 160% vom tatsächlichen Aufwand ab (Barry Boehm und Steve McConnell).
- Daraus lassen sich bei der mittleren Velocity verschiedene Szenarien für die Releaseentwicklung ermitteln.

# Scrum: Reporting (I)

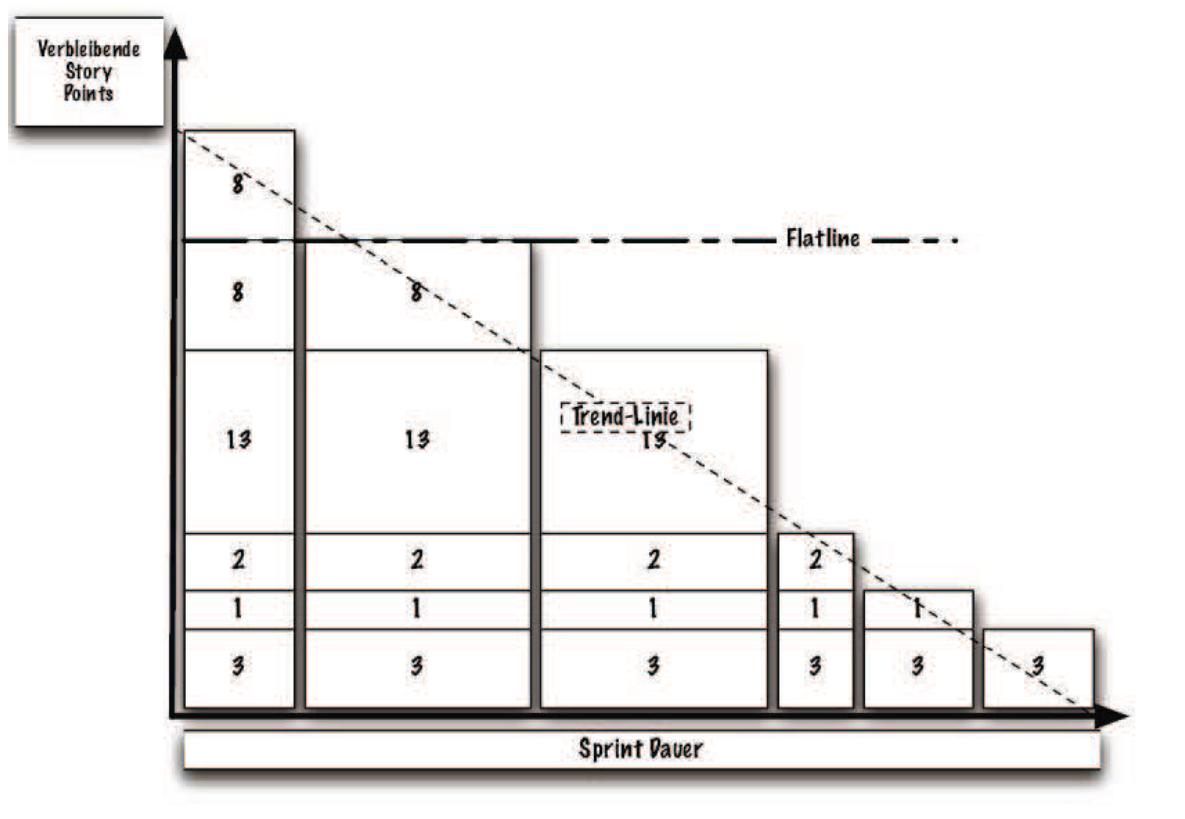
## Sprint Burndown Chart



Quelle: Gloger (2008)

# Scrum: Reporting (II)

## Sprint Product Burndown Chart

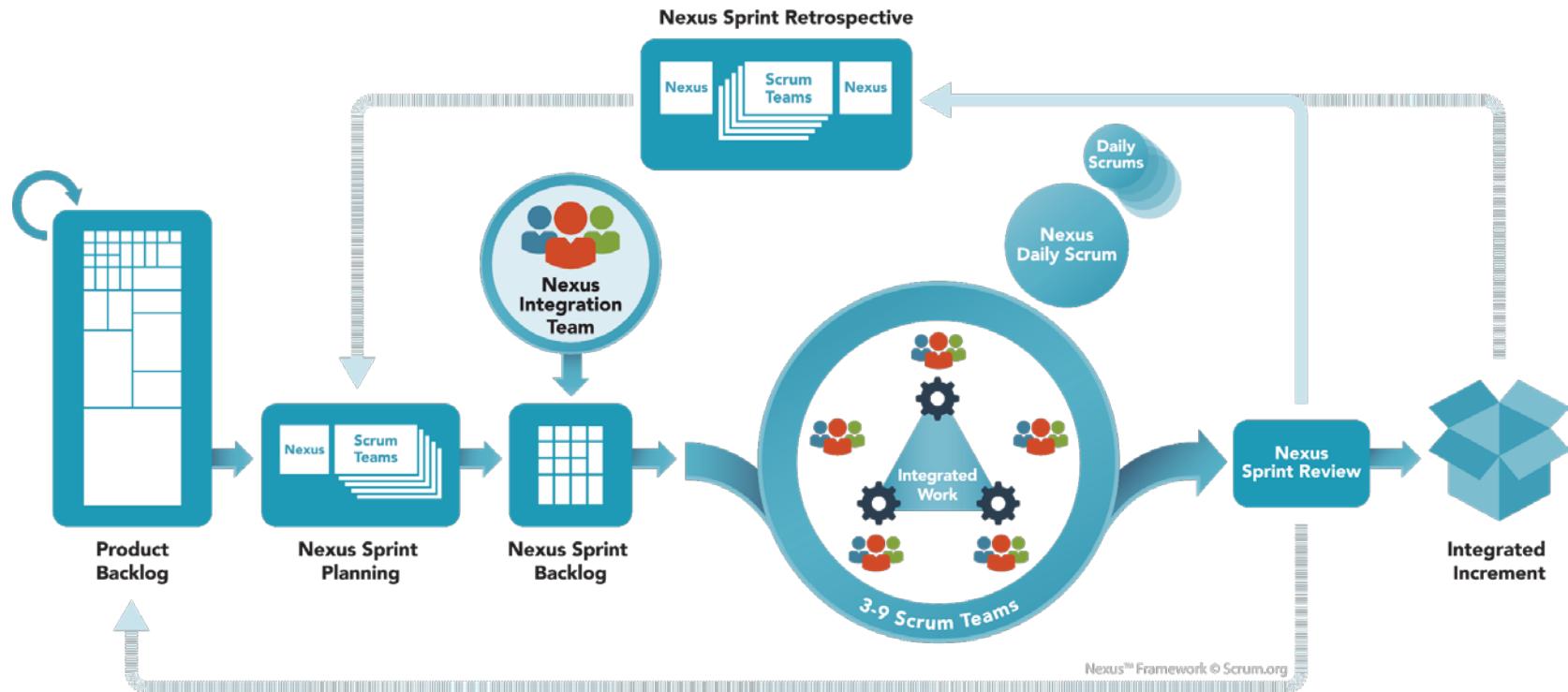


Quelle: Gloger (2008)

- Ist immer dann zu finden, wenn mehr als ein Team für das Erstellen eines Produkts/Systems/Programms benötigt wird (und dabei Scrum nutzt...).
- Höhere Kosten durch zunehmende Komplexität, nicht-lineare Effekte etc.
- Die Zusammenarbeit multipler Scrum-Teams muss koordiniert werden.
- Ansätze:
  - LeSS (Large-Scale-Scrum)
  - Nexus Scrum

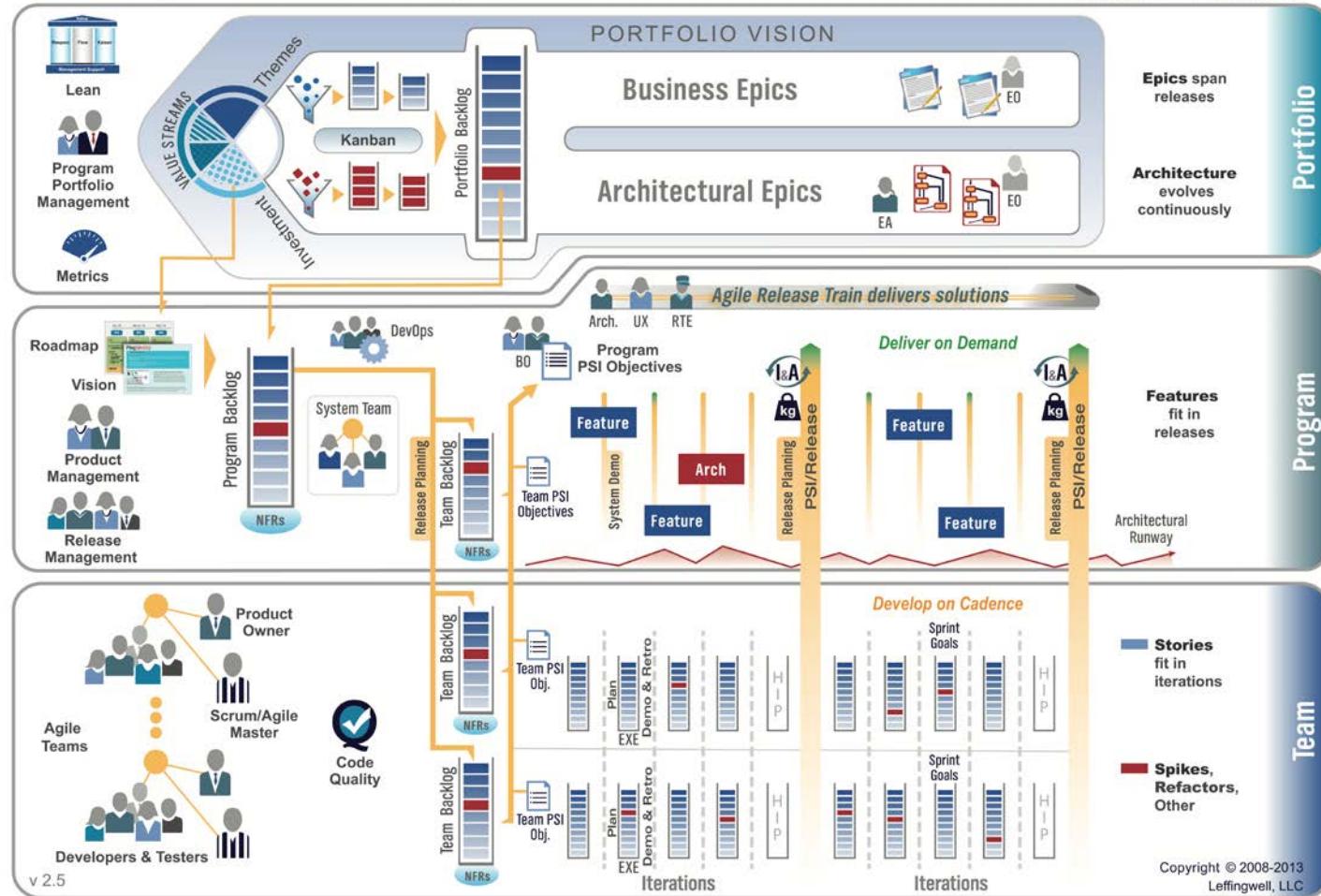
# Nexus Scrum

- „the Nexus is an exo-skeleton for 3-9 Scrum Teams working on a single Product Backlog to build an integrated increment that meets a goal.“ (Verheyen, 2015)



# Scaled Agile Framework

## Scaled Agile Framework® Big Picture



# Kanban in der SW-Entwicklung

---

- Woher kommt Kanban?
  - Man mische Methoden des Geschäftsprozessmanagements (Kaizen und Kanban) mit der *Theory-of-Constraints* von Goldratt aus dem Projektmanagement und Ansätzen aus dem Risikomanagement und wende dies bei der SW-Entwicklung an!
- Lean is beautiful: Aus der Produktion lernen
- Goldratt: „Der Engpass bestimmt den Durchsatz.“
- Die Grenzen/Nachteile von Scrum als Existenzberechtigung für Kanban?
  - Daher ist Kanban in der SW-Entwicklung auch ein sehr junger Ansatz! (ca. 2007)

# Kanban: Erste Begrifflichkeiten

---

- WIP:
  - Work in Progress
  - Begrenzung parallel laufender Arbeiten
- Cycle Time
  - Manchmal auch „lead time“
  - Zeitraum zwischen dem Beginn und dem Ende der Arbeit an einem Werkstück, gemessen über die gesamte Wertschöpfungskette hinweg
  - Entspricht dem Begriff der „Durchlaufzeit“ im Geschäftsprozessmanagement
- Average Completion Rate
  - „the average output of a production process per unit of time“
- Zusammenhang:  $CT = WIP/ACR$
- Little's Law
  - Kommt aus der Warteschlangentheorie
  - $L = \lambda W$  : eigentlich kein „Gesetz“, aber dennoch hilfreich

- Kanban ist ein Pull-System: Es wird immer „ein Stück Arbeit“ aus der gesamten Aufgabenmenge herausgenommen, bearbeitet und dann dem nachfolgenden Prozessschritt als fertig signalisiert.

- Kanban beginnt dort, wo sich ein System gerade befindet.
- Kanban respektiert die bestehende Ordnung.
- Kanban strebt inkrementelle, evolutionäre Veränderungen an.
- Kanban benötigt Leadership auf allen Ebenen in der Organisation.

- Kanban ist Veränderung im Ganzen.
- Kanban bedeutet ständige Reflexion mit dem Willen und der Bereitschaft zur Verbesserung.
- Kanban dreht sich um Menschen und nicht um Mechaniken.
- Kanban braucht ein Team.

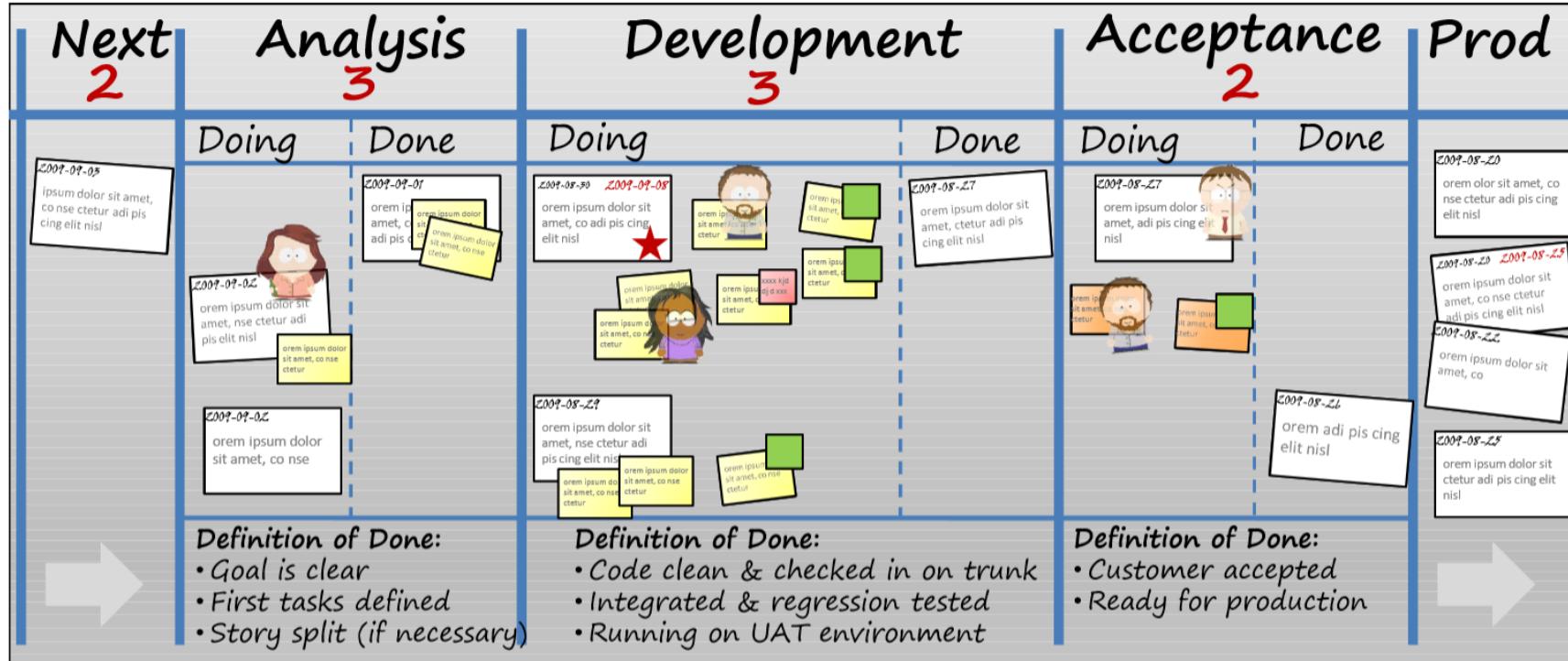
In Anlehnung an Leopold/Kaltenecker (2014)

# Die 6 Kernpraktiken einer Kanban Implementierung

---

- Mach Arbeit sichtbar.
- Limitiere den Work in Progress (WiP).
- Verwalte den Arbeitsfluss.
- Mach Prozessregeln explizit.
- Implementiere Feedback-Mechanismen.
- Führe gemeinschaftliche Verbesserungen durch.

# Kanban Board: Visualisierung für mehr Effizienz



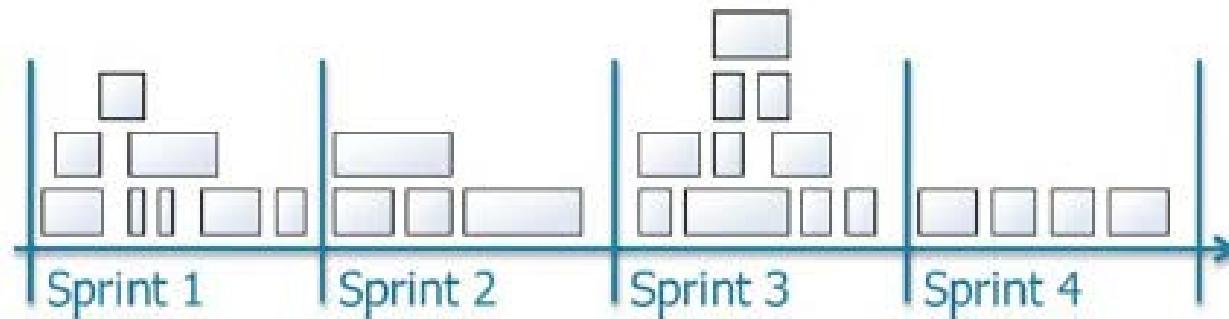
# Kanban vs. Scrum (I)

---

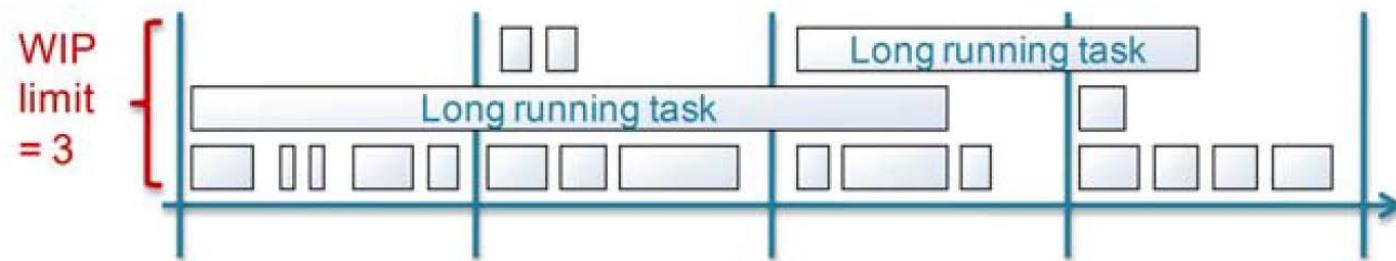
- Scrum and Kanban können beide als *process tools* gesehen werden.
- Beide sind *lean* und agil.
- Scrum schreibt Rollen und zeitlich begrenzte Iterationen (*timeboxed iterations*) vor.
- Kanban begrenzt WIP pro Workflow Stadium – Scrum begrenzt WIP pro Iteration.
- Bei Scrum müssen die ausgewählten *backlog items* in einen Sprint passen.
- Das Scrum Board wird nach jeder Iteration zurückgesetzt .
- Scrum schreibt interdisziplinäre Teams vor.
- Scrum schreibt Schätzungen der Items und der Geschwindigkeit vor.

# Kanban vs. Scrum (II)

Scrum backlog items müssen in einen Sprint passen

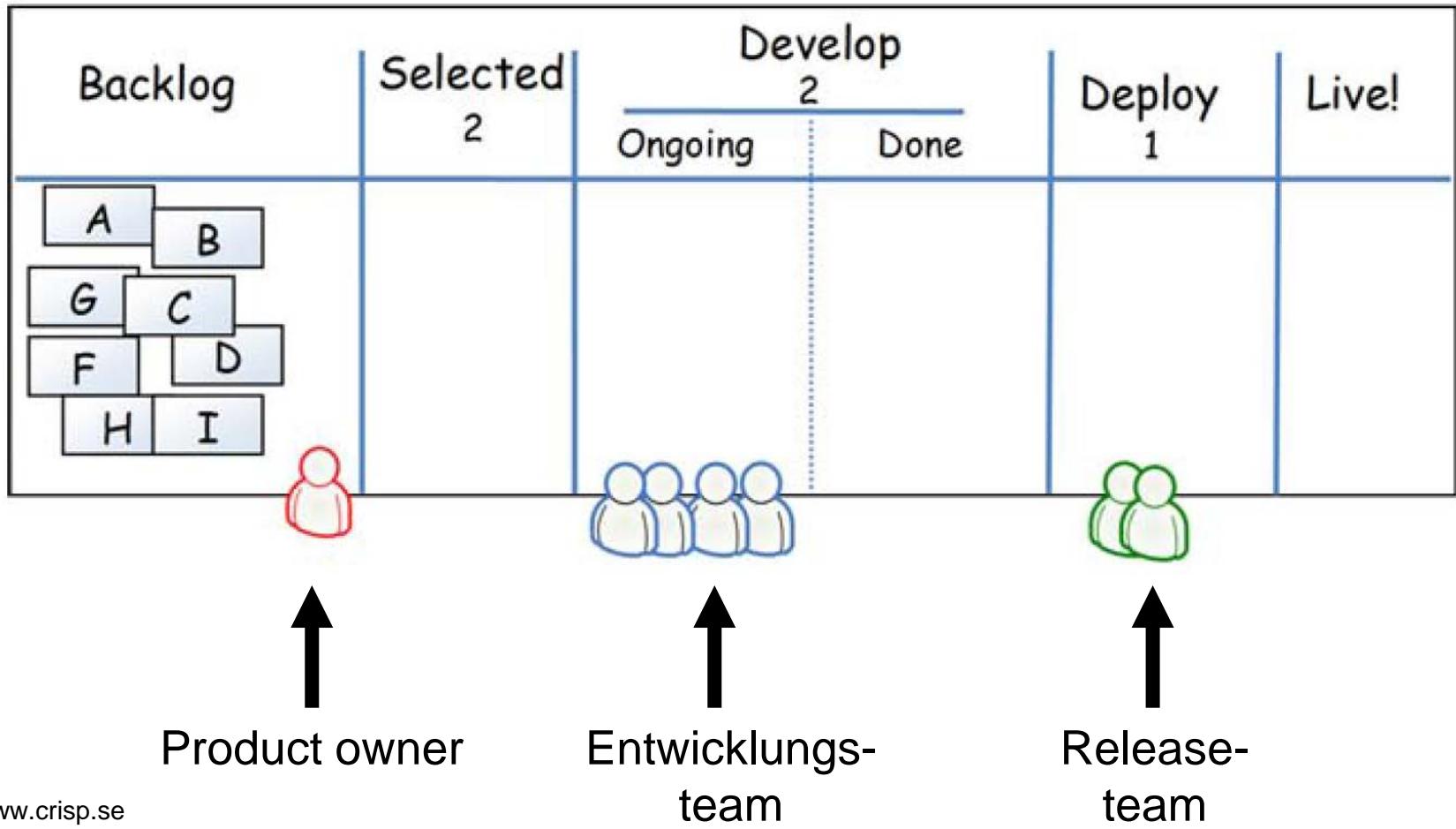


Kanban erlaubt kurze und lange Tasks

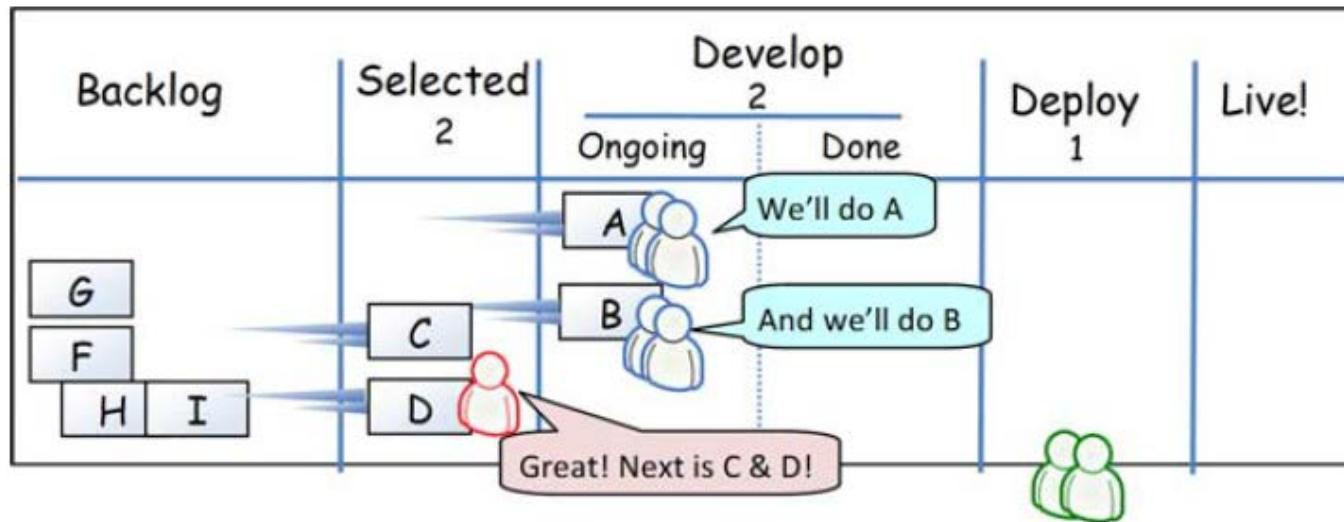
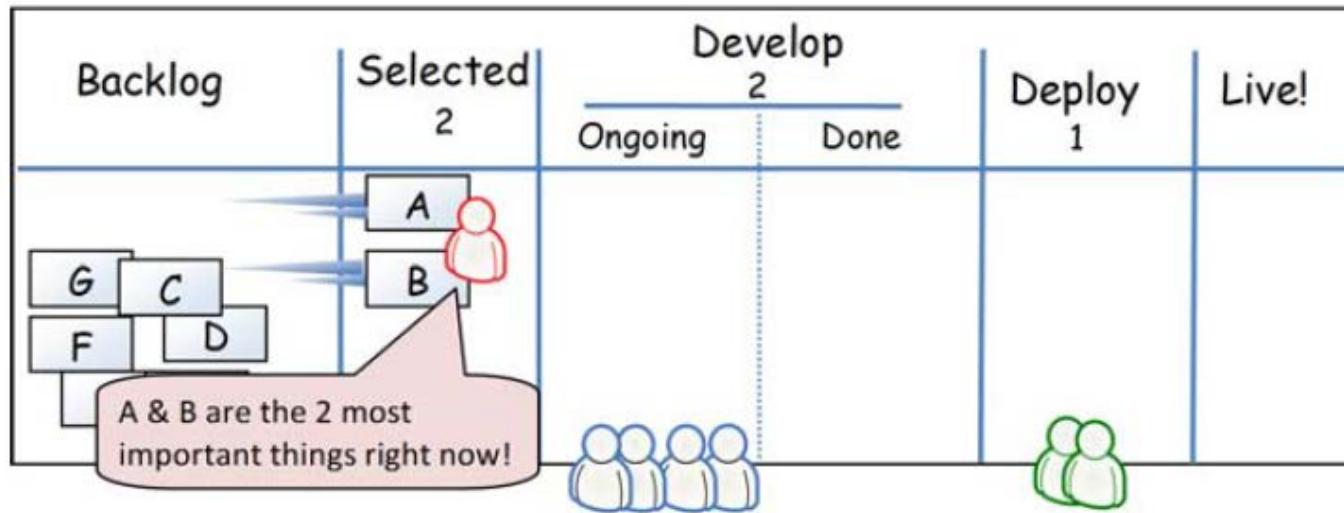


Quelle: [www.crisp.se](http://www.crisp.se)

# One day in Kanban-land (I)

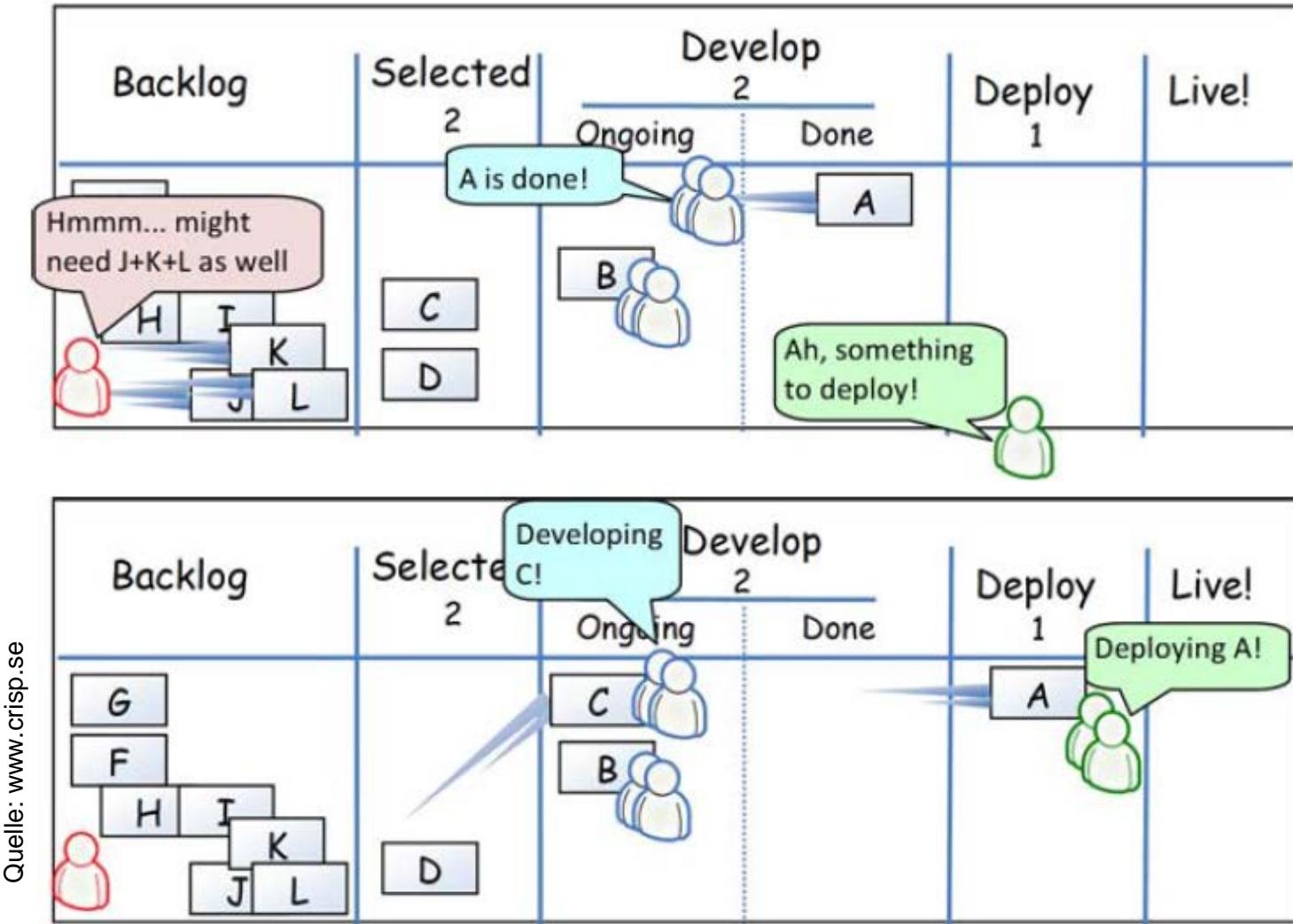


# One day in Kanban-land (II)

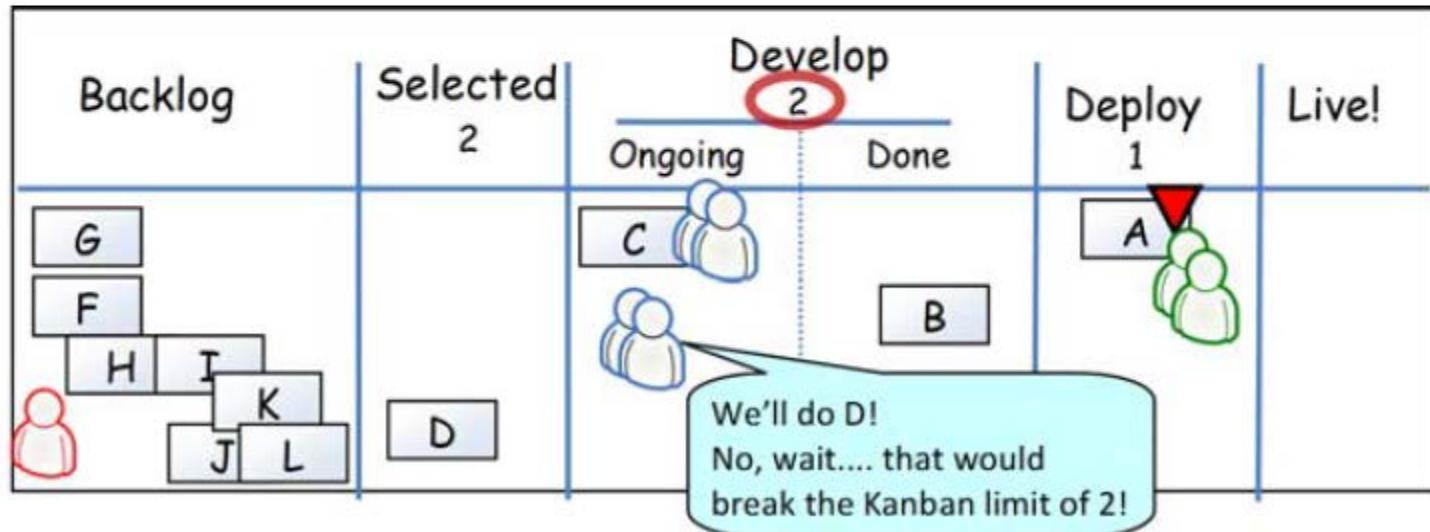
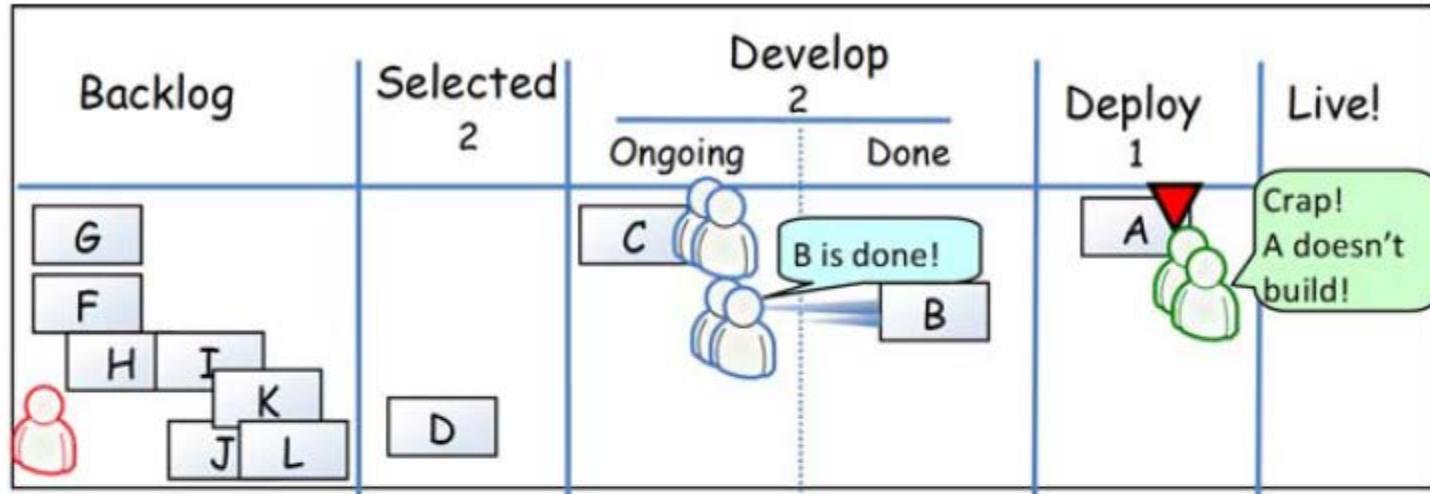


Quelle: www.crisp.se

# One day in Kanban-land (III)

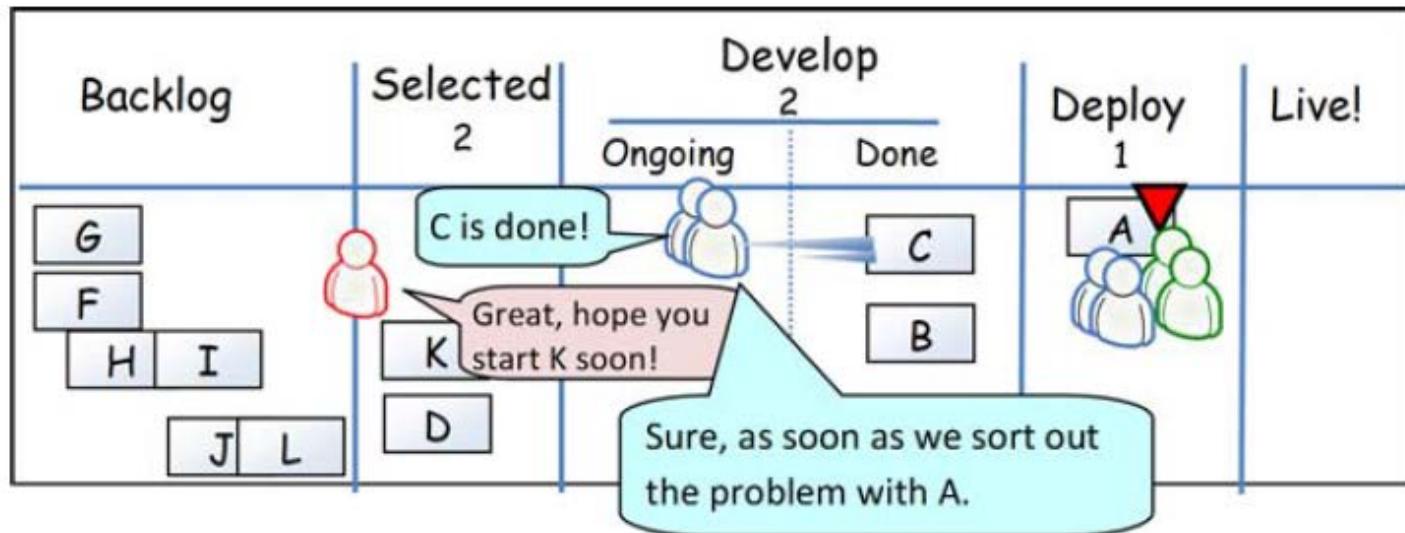
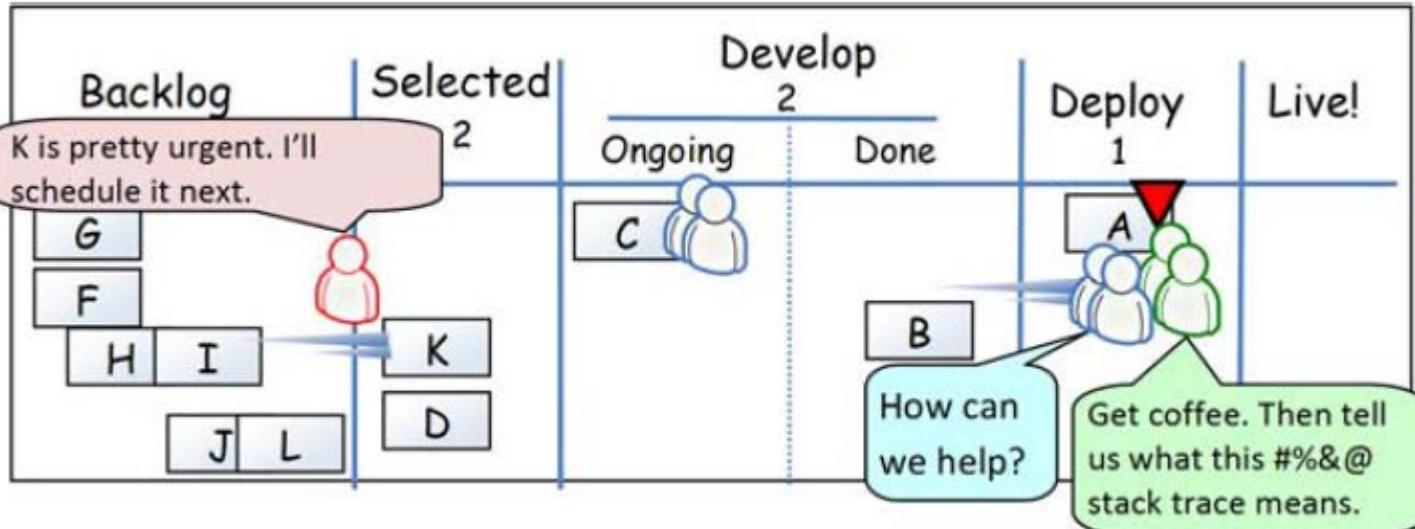


# One day in Kanban-land (IV)



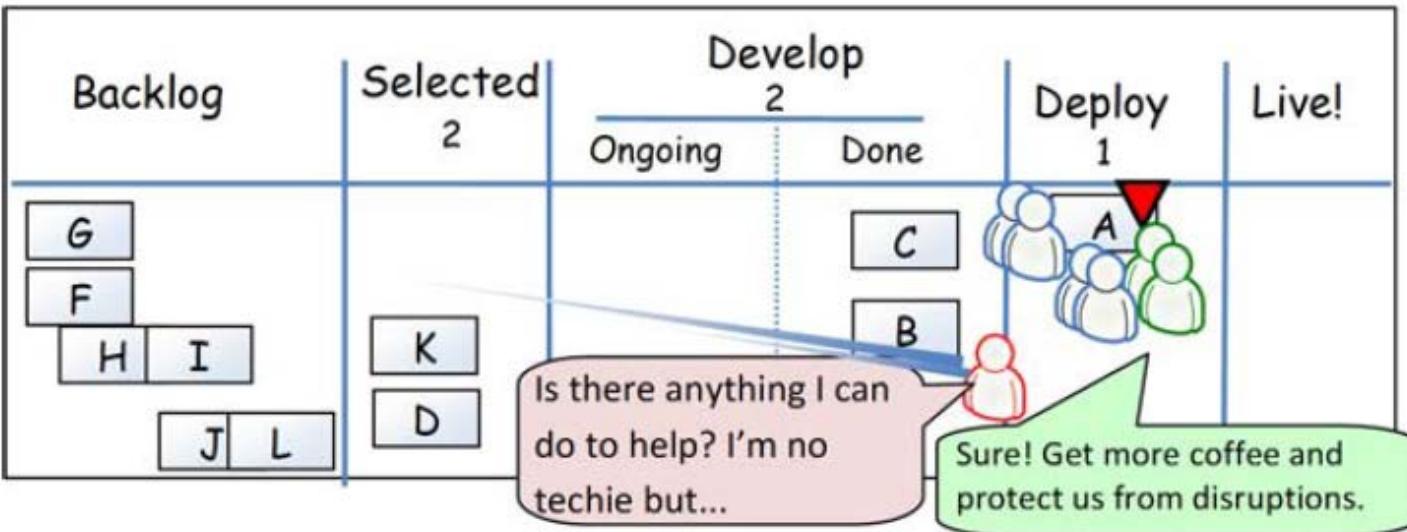
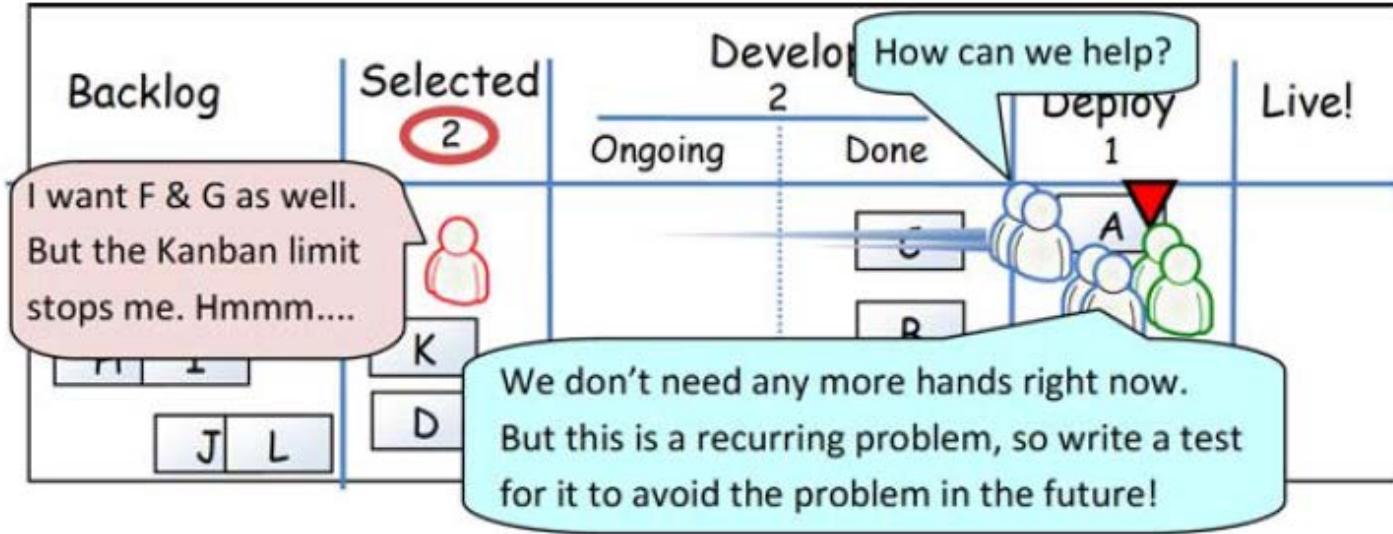
Quelle: [www.crisp.se](http://www.crisp.se)

# One day in Kanban-land (V)



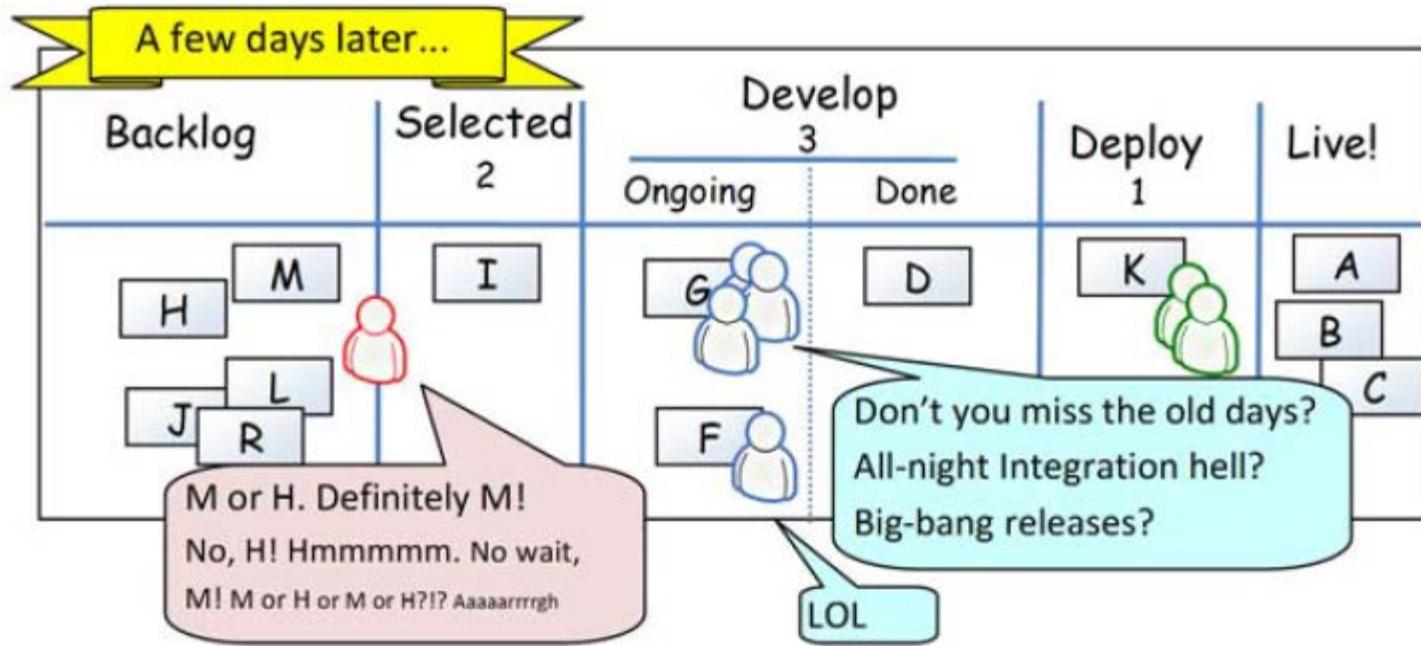
Quelle: www.crisp.se

# One day in Kanban-land (VI)



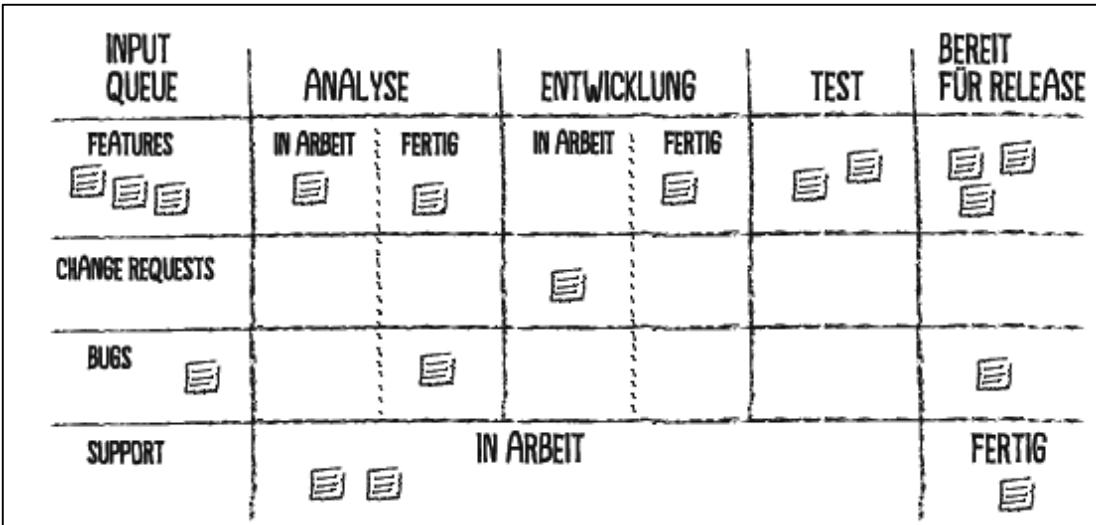
Quelle: www.crisp.se

# One day in Kanban-land (VII)



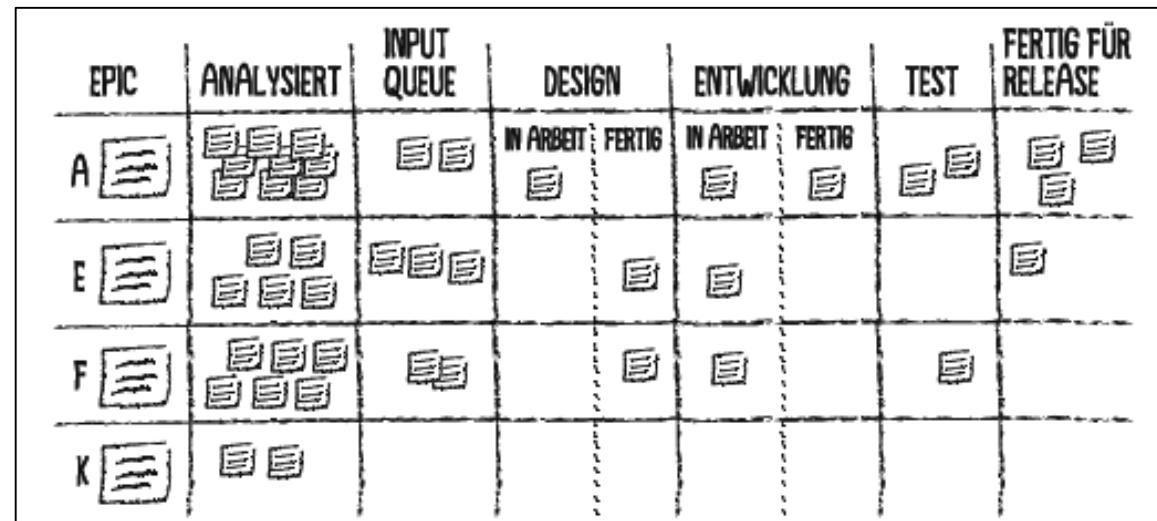
Quelle: [www.crisp.se](http://www.crisp.se)

# Varianten für das Kanban-Board



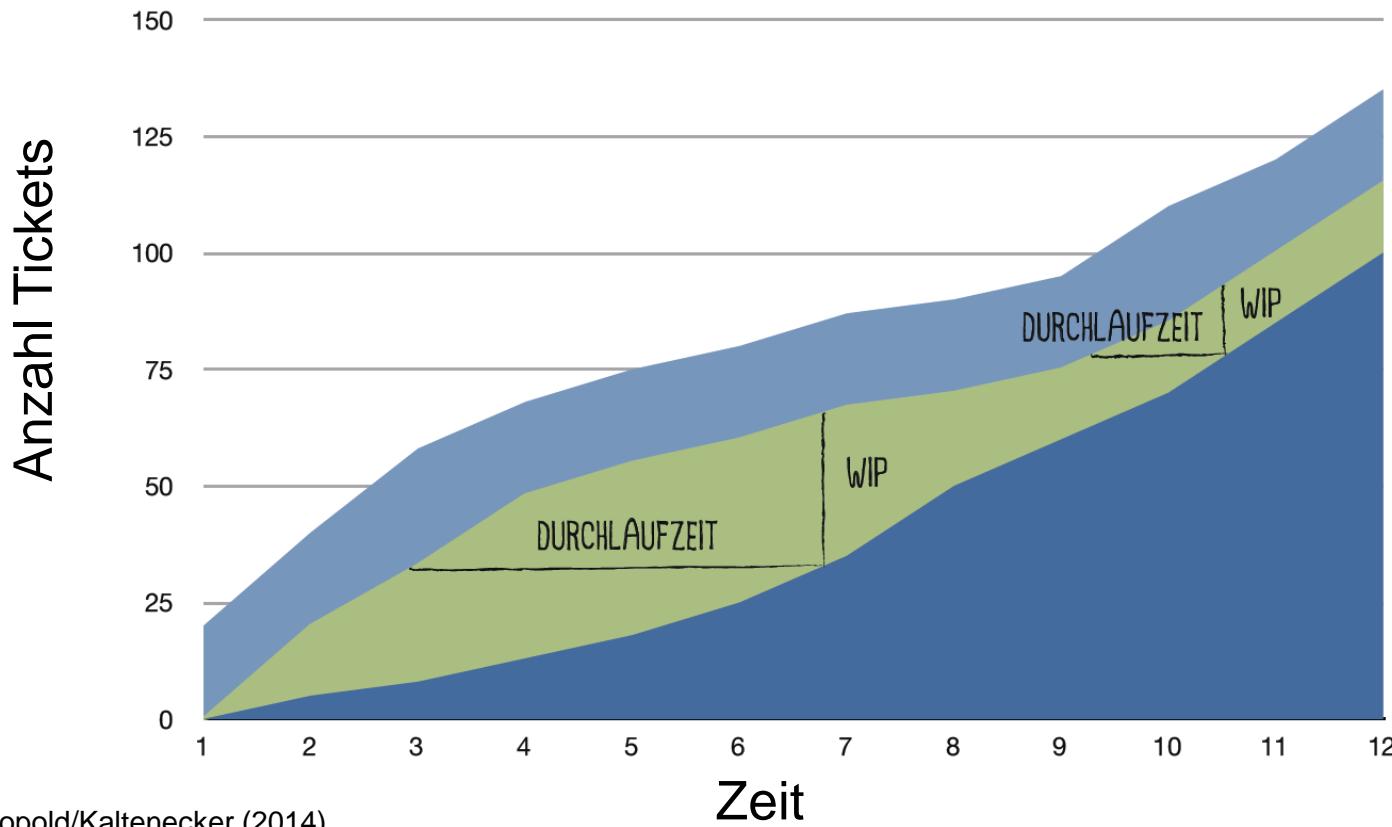
Kanban-Board mit Swim Lanes für die einzelnen Aufgabentypen

Zerteilung von Epics in kleinere User Stories



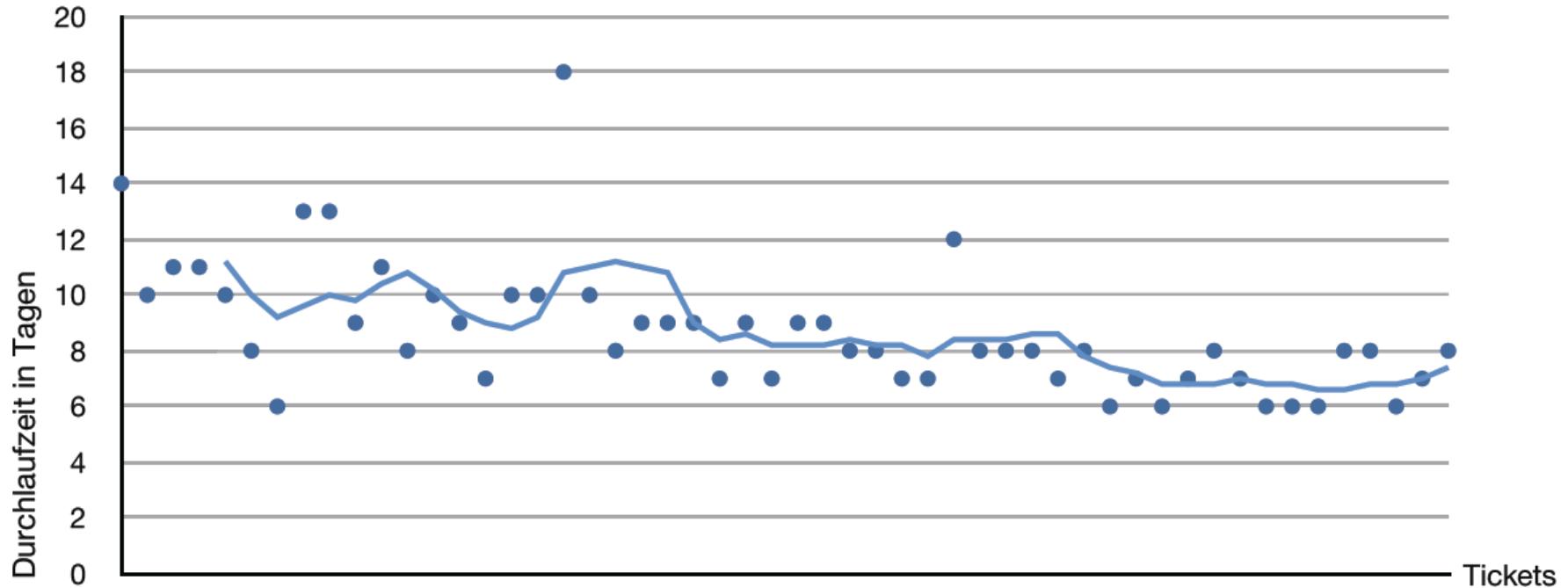
Quelle: Leopold/Kaltenecker (2014)

## Das Cumulative-Flow-Diagramm



Quelle: Leopold/Kaltenecker (2014)

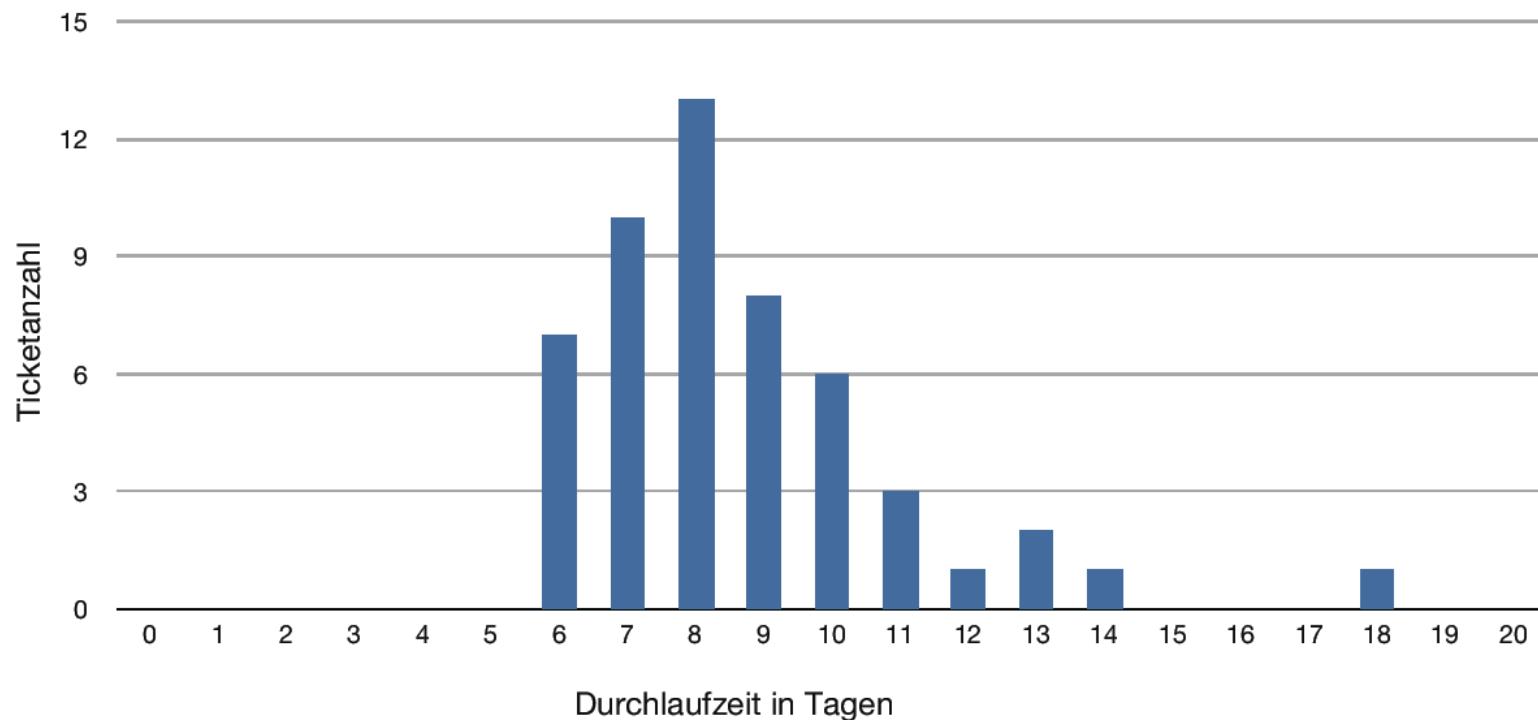
## Die Durchlaufzeit



Quelle: Leopold/Kaltenecker (2014)

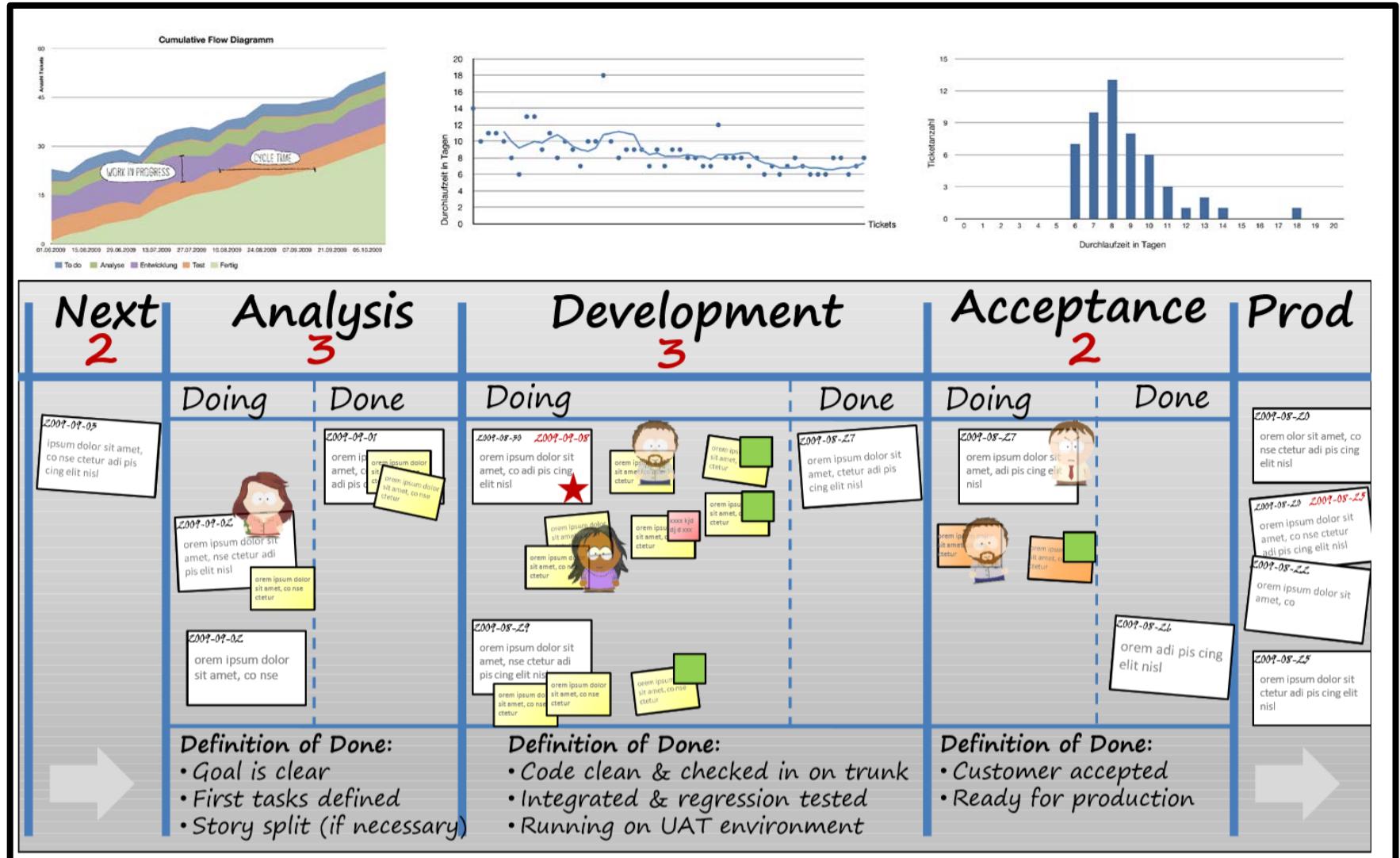
# Berichte in Kanban (III)

## Spektralanalyse der Durchlaufzeit

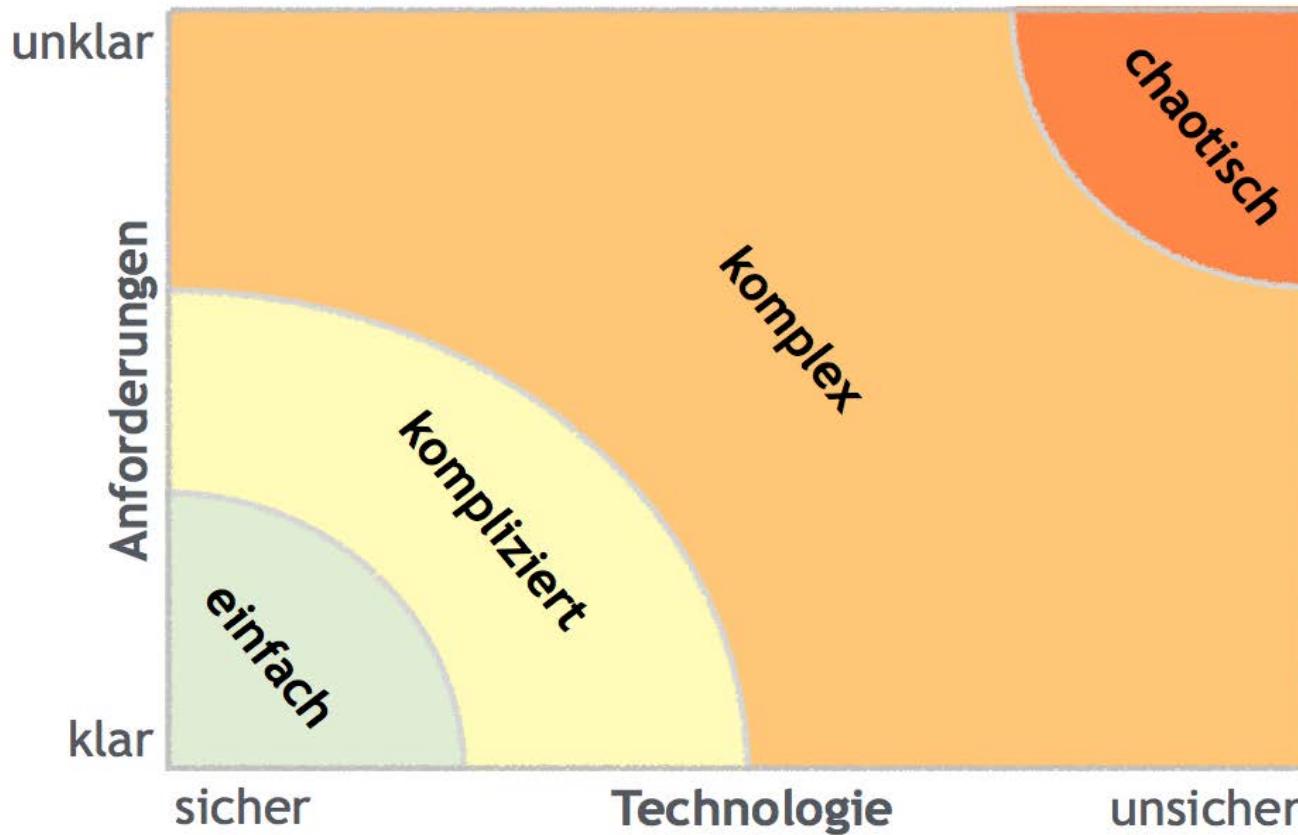


Quelle: Leopold/Kaltenecker (2014)

# Vollständiges Kanban-Board



# Agile Verträge: Einordnung



## Das Stacey-Landscape-Diagramm

Quelle: Pieper/Roock (2017)

# Agile Verträge: Basis des Vertrags

Drei gemeinsame agile Grundideen, die Auftraggeber und Auftragnehmer berücksichtigen müssen:

- *Unvollständige Funktionsbeschreibung*: Wir wissen zu Projektbeginn noch nicht vollständig, wie die Software aussehen muss, um den erhofften Nutzen zu bringen.
- *Gemeinsames Verständnis*: Wir erzeugen ein gemeinsames Verständnis über Gespräche und nicht über Dokumente.
- *Kooperative Problemlösung*: Wir reagieren auf Probleme mit größerer Nähe und nicht mit größerer Distanz.

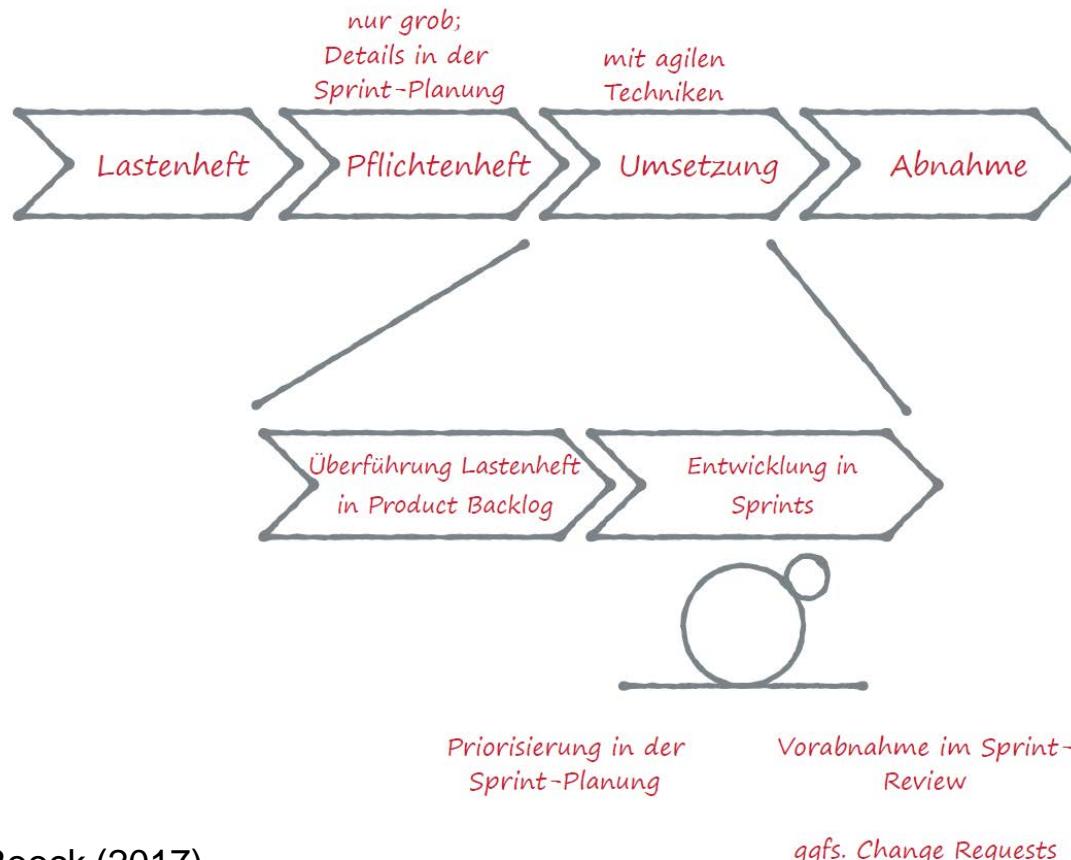
Quelle: Pieper/Roock (2017)

**Aber:** „Die meisten in der Praxis eingesetzten agilen Verträge sind leider Schönwetterverträge – sie gehen kaum auf die verwendeten Methoden ein und funktionieren nur, solange im Projekt alles wie geplant läuft und keine Konflikte auftreten. [...] Es ist jedoch die wichtigste Aufgabe eines Softwarevertrages, das Projekt auch dann auf der Schiene zu halten, wenn sich die Parteien in bestimmten Fragen uneins sind.“

(Brenner/Feiler (2018))

# Agile Verträge: Festpreis

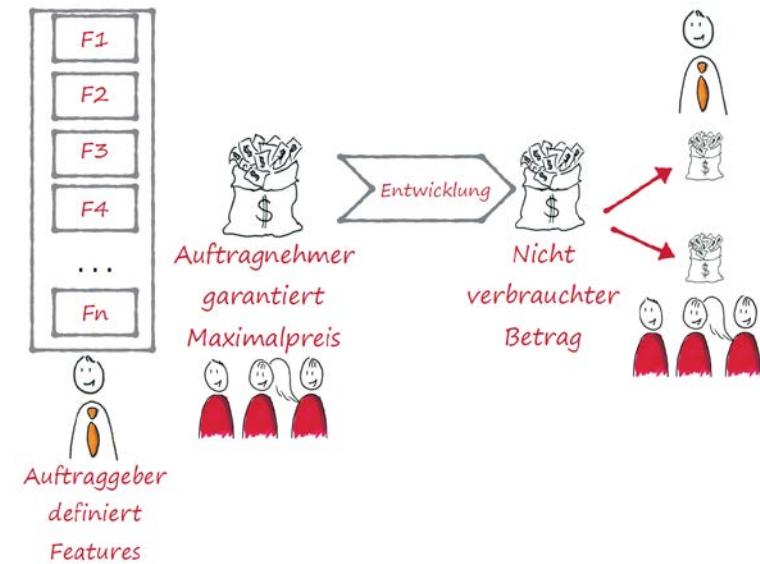
- Möglich, aber in der Regel nicht gut
- Änderungen bei den Anforderungen kaum durchführbar



Quelle: Pieper/Roock (2017)

# Agile Verträge: Garantierter Maximalpreis

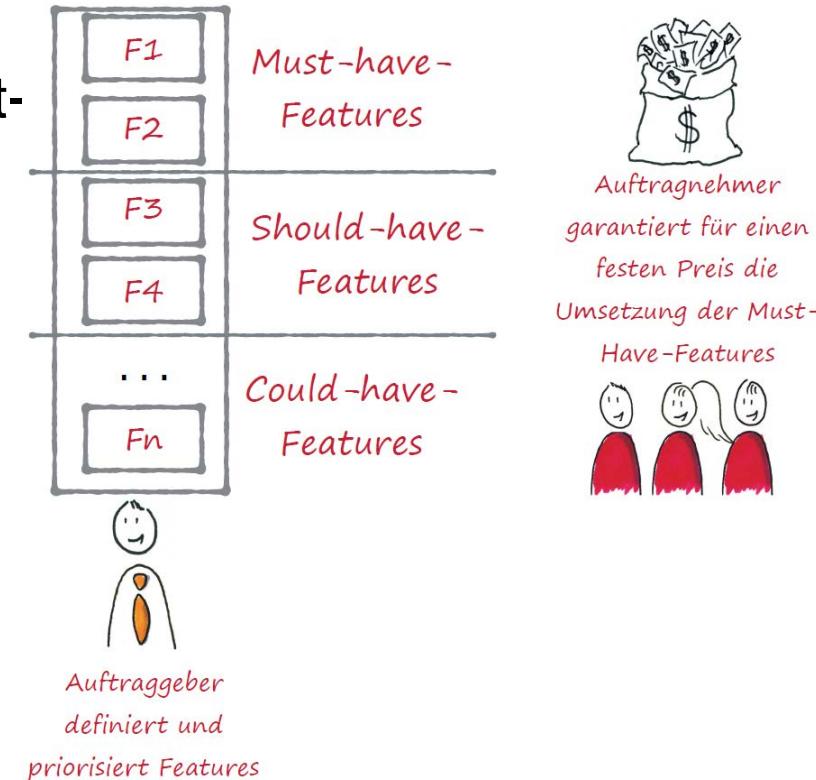
- Ursprung in der Baubranche
- Wird die Entwicklung teurer als der vereinbarte Maximalpreis, geht dies zulasten des Anbieters.
- Wird die Entwicklung hingegen günstiger als der Maximalpreis, wird der nicht verbrauchte Betrag nach einem vorher vereinbarten Schlüssel aufgeteilt.



Quelle: Pieper/Roock (2017)

# Agile Verträge: Garantierter Minimalumfang

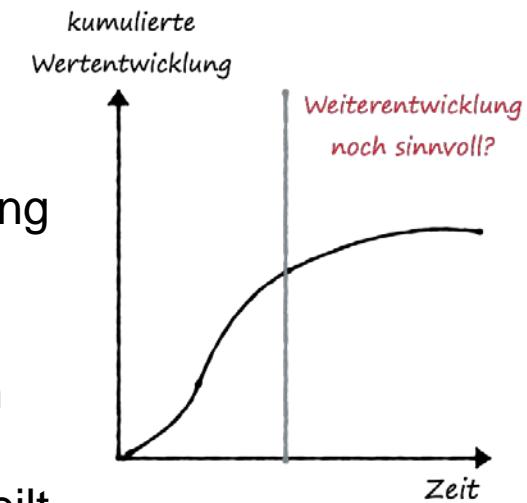
- Umgeht im Budgetierungsprozess Schwierigkeiten mit flexiblen Budgets
- Einteilung der Funktionalitäten in Must-have-, Should-have- und Could-have-Features
- Der Anbieter garantiert für einen bestimmten Preis einen Mindestumfang (die Must-have-Features).
- Ist mit dem Mindestumfang der Kostenrahmen nicht ausgeschöpft, werden so lange Should-have- und Could-have-Features entwickelt, wie es der Kostenrahmen zulässt.



Quelle: Pieper/Roock (2017)

# Agile Verträge: Weitere Möglichkeiten

- Agiler Fixpreis
  - Grobplanung in Epics
  - Ausgewählte Epics werden detailliert ausgearbeitet und geschätzt
  - Hochrechnung auf das Gesamtprojekt
  - Zunächst nur Umsetzung in einem Vorprojekt; erst später auf Gesamtprojekt anwendbar
- »Money for Nothing, Change for Free«
  - Ausgangspunkt ist Festpreis-Vertrag
  - Neue Features können nur in den Product-Backlog aufgenommen werden, wenn andere mit einem Umfang von ebenso vielen Story-Points entfernt werden („Change for free“).
  - Product-Owner kann Projekt jederzeit beenden, wenn die Wertschöpfung nicht mehr groß genug ist. Restbudget wird nach vereinbartem Schlüssel aufgeteilt („Money for nothing“).



Quelle: Pieper/Roock (2017)

# Continuous Delivery

---

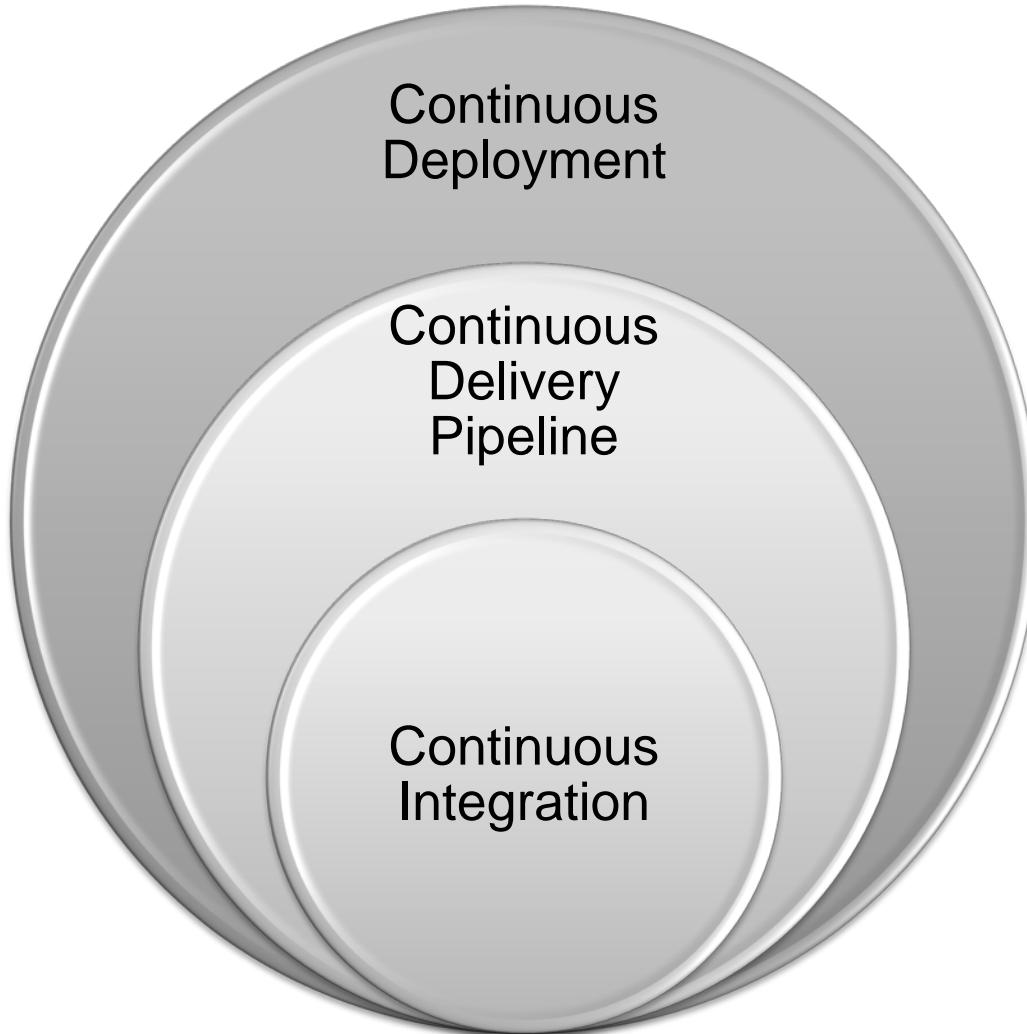
„Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.“

Aus: Principles behind the Agile Manifesto

„Continuous Delivery bedeutet, seine Software so zu entwickeln, dass sie sich jederzeit in einem Zustand befindet, in dem sie deployed werden kann.“

(Martin Fowler)

# Continuous Delivery (II)

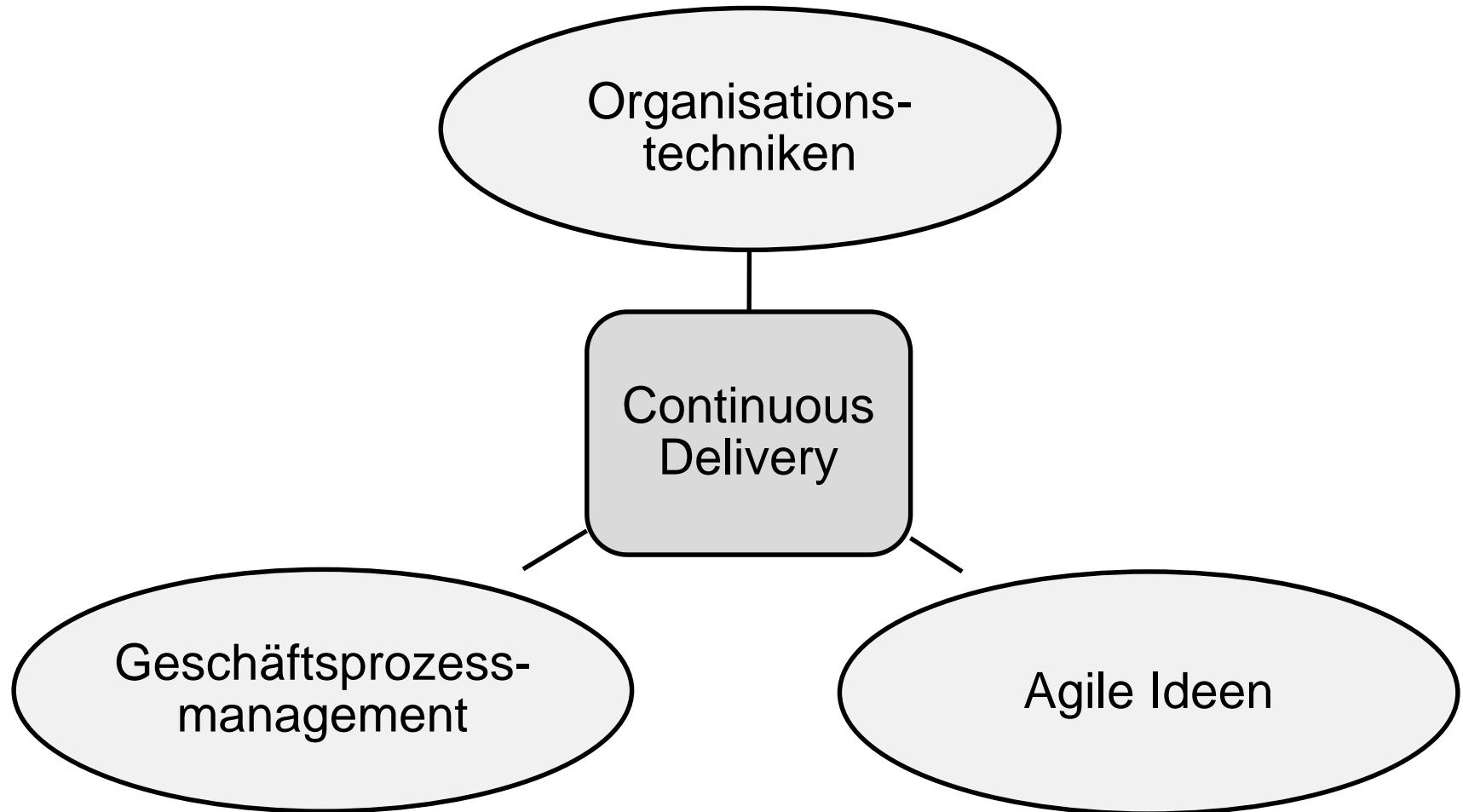


# Konzeptionelle Grundlagen

---

- Agilität
- Geschäftsprozessmanagement
- Lean Management/Production
- DevOps

# Konzeptionelle Grundlagen



# Continuous Integration

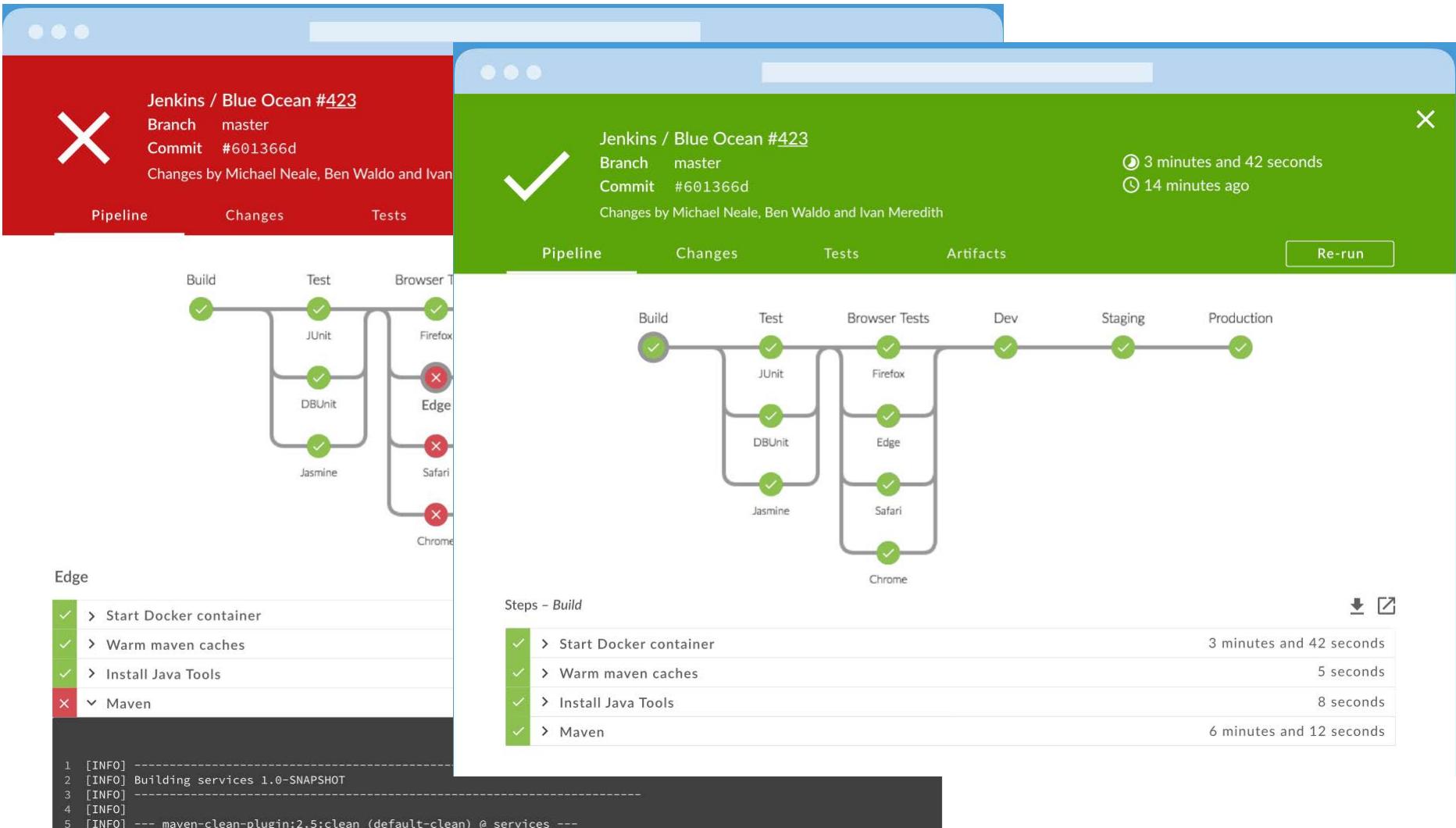
---

- Der erste Schritt
- Folge von Fowler und XP
- Zentraler Aspekte:
  - Version Control Systems
  - CI-Server (Jenkins, Hudson, Bamboo...)

# Continuous Delivery Pipeline



# Continuous Delivery Pipeline mit Jenkins



The screenshot shows two Jenkins Blue Ocean pipeline interfaces side-by-side.

**Left Pipeline:**

- Build Stage:** All steps (Start Docker container, Warm maven caches, Install Java Tools) are successful (green checkmarks).
- Test Stage:** JUnit, DBUnit, and Jasmine are successful (green checkmarks). Edge, Firefox, Safari, and Chrome are failing (red X's).
- Browser Tests Stage:** Firefox, Edge, and Safari are successful (green checkmarks). Chrome is failing (red X).

**Right Pipeline:**

- Build Stage:** All steps (Start Docker container, Warm maven caches, Install Java Tools) are successful (green checkmarks).
- Test Stage:** JUnit, DBUnit, and Jasmine are successful (green checkmarks).
- Browser Tests Stage:** Firefox, Edge, and Safari are successful (green checkmarks). Chrome is failing (red X).
- Dev Stage:** Successful (green checkmark).
- Staging Stage:** Successful (green checkmark).
- Production Stage:** Successful (green checkmark).

**Logs:**

```
1 [INFO] -----  
2 [INFO] Building services 1.0-SNAPSHOT  
3 [INFO] -----  
4 [INFO]  
5 [INFO] --- maven-clean-plugin:2.5:clean (default-clean) @ services ---
```

# Continuous Deployment

---

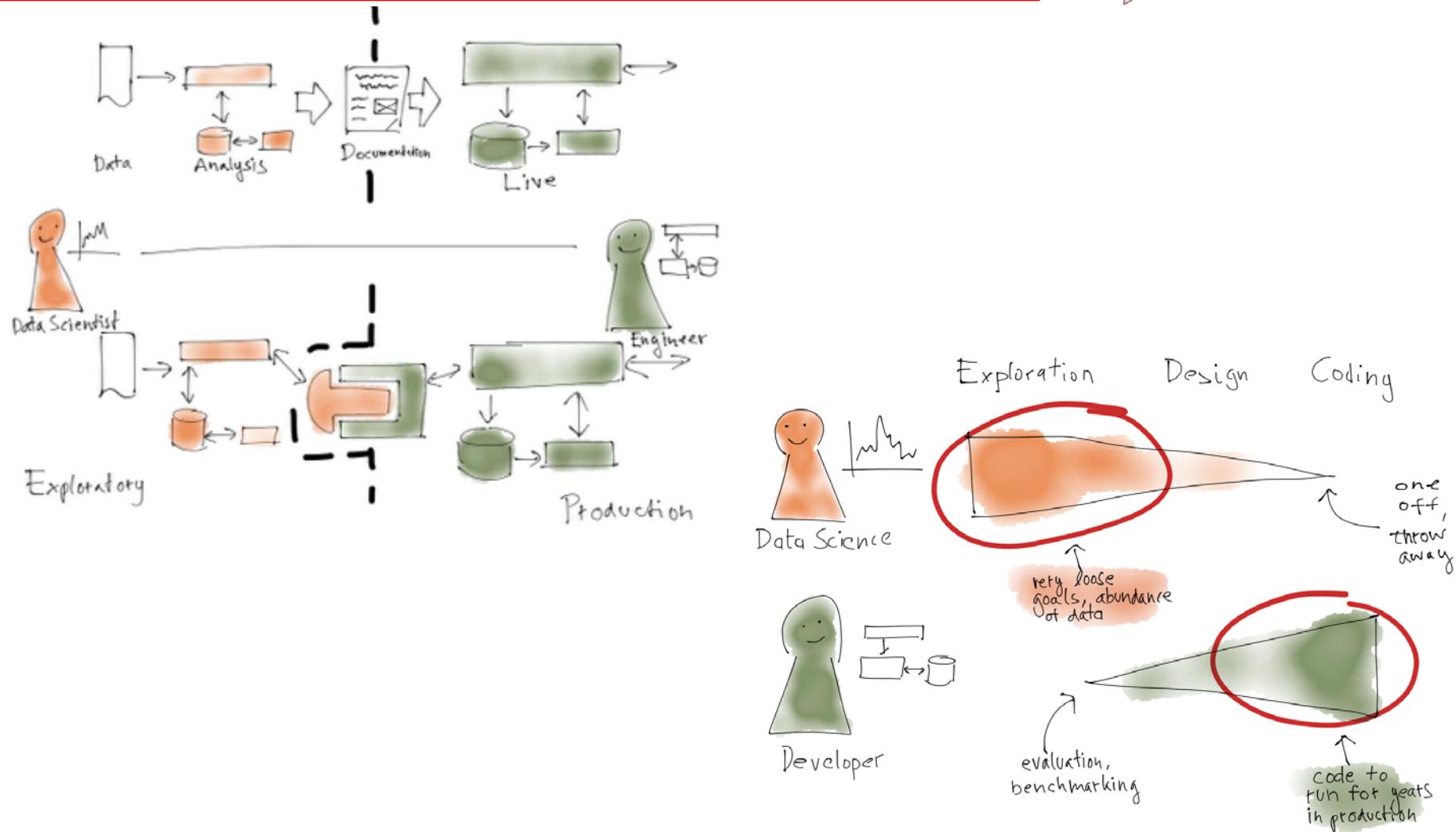
- Eigentlich nur noch ein letzter Schritt
- Bei *Continuous Delivery* ist die Software jederzeit „deployment ready“, aber die Übergabe an das Produktivsystem erfolgt manuell.
- Bei *Continuous Deployment* wird dieser letzte Schritt automatisiert.
- Folge: Zahlreiche Produktivsetzungen pro Tag.

- Kunstwort aus „Development“ und „Operations“
- Unterschiedliche Aufgaben und Zielsetzungen führen zu Ineffizienzen und Reibungsverlusten
- Lösung über engere Verzahnung der Abteilungen in gemischten Teams

**„Teams are composed of members with different functional backgrounds, thus fostering cross-functional communication and knowledge sharing.“**

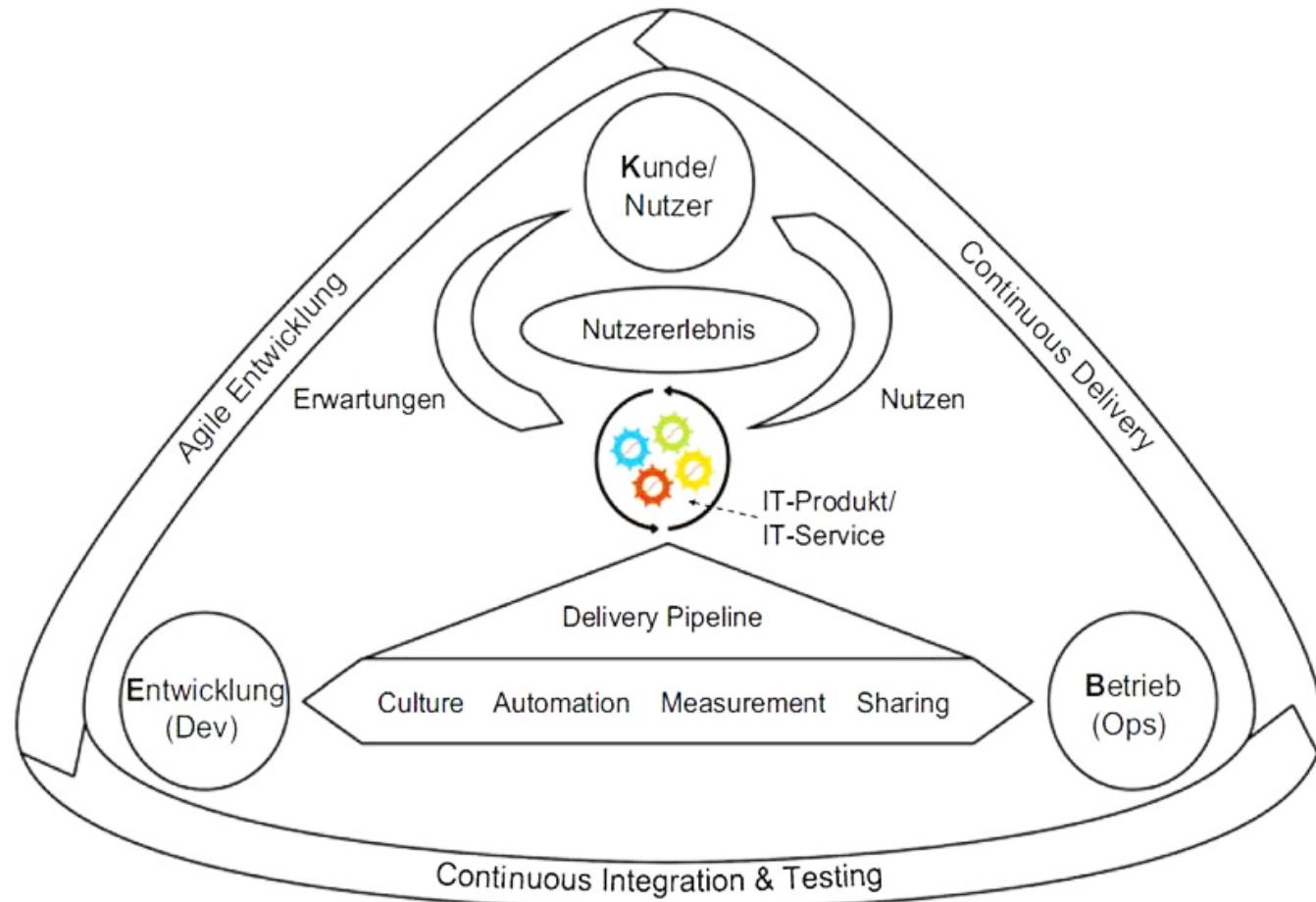
DeLone, Migliorati und Vaia (2018)

# Data Science mit DevOps bei Zalando



Quelle: Mikio Braun

# Das DevOps-Gesamtkonzept



Quelle: Alt et al. (2017)

„The key is having good people [...]. No process makes up for a lack of talent and skill.“

Jim Highsmith

**Aber er sagt auch:**

„Working code is the ultimate arbiter of real progress.“

**"Ihr würdet Euch wundern, wenn ihr wüsstet, mit wie viel Unverständ die Welt regiert wird."**

**Papst Julius II**

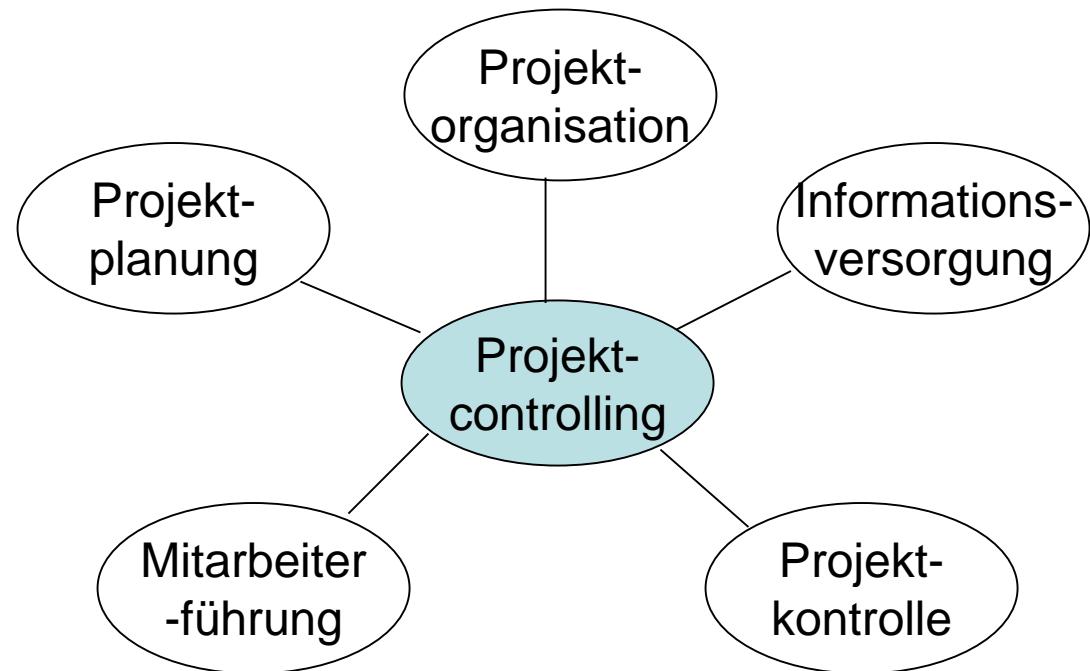
**Tun wir etwas dagegen!**

- ⇒ **Wer steuert muss informiert sein!**
- ⇒ **Information/Wissen ist in den Unternehmen ein wichtiger Wettbewerbsfaktor!**
- ⇒ **Die Dezentralisierung im Unternehmen (Projekte; Profit Center; Ausgliederung von Teilbereichen in selbstständige Unternehmen) führt zu einer steigenden Informationsnachfrage!**
- ⇒ **Im Unternehmen / Projekt werden Entscheidungen immer weniger auf der Basis von „Erfahrungen“ oder „aus dem Bauch heraus“ getroffen**
- ⇒ **Problem der Berichtsüberflutung in Unternehmen / Projekten**

# Aufgaben des Projektcontrolling

Projektcontrolling unterstützt die Ausgestaltung des Projektmanagements als Basis für die Projektarbeit:

- Planerstellung (Termine, Ressourcen, Kosten)
- Plankontrolle
- Projektorganisation
- Workflow im Projekt
- Berichtswesen
- Koordinationsaufgaben der laufenden Projektabwicklung



# Informationssystem & Berichtswesen im Projekt (1)

---

- Das Berichtswesen muss vor dem Projektstart definiert werden
- Am besten liegt bereits ein standardisiertes Berichtswesen vor (sonst entwickelt jedes Projekt sein eigenes Berichtswesen!)
- Das Projektberichtswesen muss kompatibel zum übrigen Informationssystem des Unternehmens sein

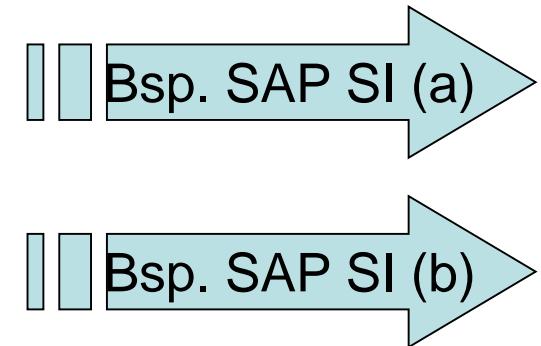
# Informationssystem & Berichtswesen im Projekt (2)

---

- Das Projektberichtswesen ist in der Regel empfängerorientiert aufgebaut. Empfänger können sein:
  - Auftraggeber
  - Lenkungsausschuss
  - Projektleiter
  - Teilprojektleiter
  - Mitarbeiter im Projektteam
  - Unterauftragnehmer (aber auch Generalunternehmer)
- Ein senderorientiertes Berichtswesen stellt den Ersteller der Berichte in den Mittelpunkt, z.B.:
  - Projektcontroller: Berichte über Termine und Kosten
  - Systemadministratoren: Berichte über Stand der DV-Systeme

# Begrifflichkeiten

- Berichtswesen (formal/ informell)
- Informationsfluss (formal/informell)
- Informationsversorgungssystem
  - Informationsbedarfsanalyse
  - Informationsbeschaffung
  - Informationsaufbereitung
  - Informationsübermittlung / -abgabe / -interpretation
  - Informationsspeicherung



# Aufgaben des Projektberichtswesens

- Entscheidungsrelevante Informationen rechtzeitig bereitstellen
- Einfachen Projektüberblick schnell geben
- Frühzeitig Risiken und klare Maßnahmen aufzeigen

⇒ **Berichte nicht mit Information/Daten überfrachten**  
⇒ „**Weniger an Information/Daten ist oft mehr an Wirkung.**“

# Anforderungen an den Informationsaustausch im Projekt

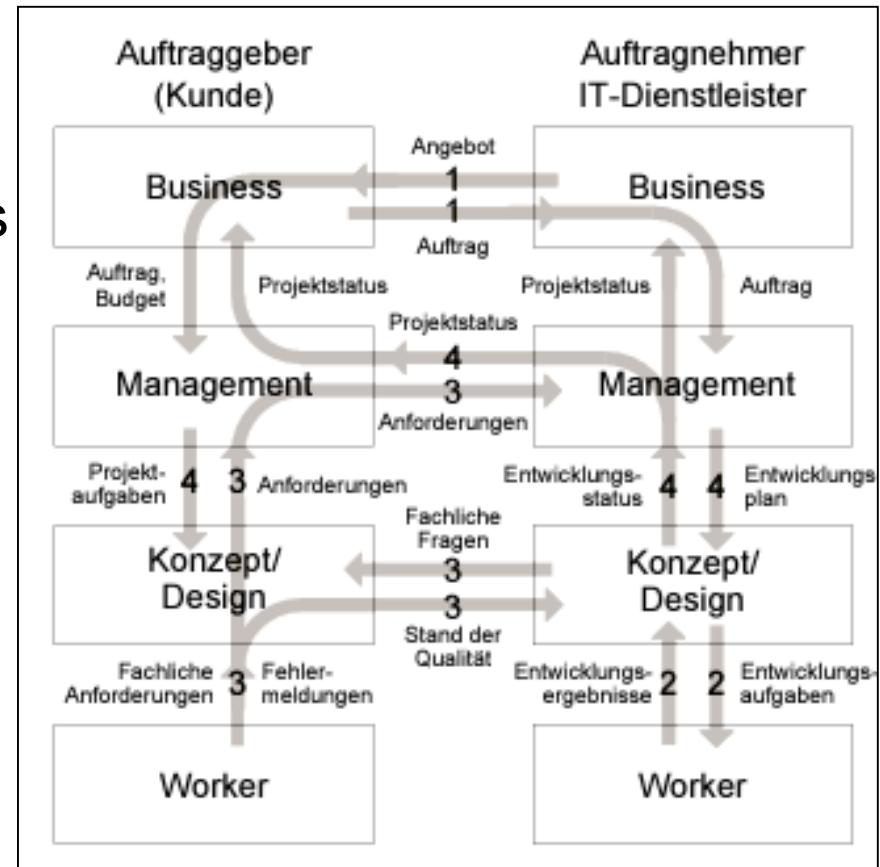
---

- **Höchstmaß an Aktualität:**  
*Motto „Schnelligkeit vor Genauigkeit“*
- **Faktenanalyse statt Schuldzuweisung**  
*Motto: „Kein Zuschieben des schwarzen Peters“*
- **Nicht nur Status dokumentieren, sondern geeignete Korrektur- und Steuerungsmaßnahmen aufzeigen**  
*Motto: „Blick nach vorn“*
- **Texte und Zahlen weitgehend visualisieren**  
*Motto: „Ein Bild sagt mehr als tausend Worte“*

- Ein systematisches Berichtswesen erleichtert die Projektdokumentation
- Eine allgemeine Ablage für Informationen unterstützt eine solche Systematik. Dies gilt u. a. für:
  - Standardberichte
  - Protokolle
  - Zwischenergebnisse
  - Testergebnisse und Abnahmen
- Softwareunterstützung durch geeignete Projektmanagement-Tools

# Informationsflüsse

1. Kaufmännischer Informationsfluss
2. Technischer Informationsfluss
3. Fachlicher Informationsfluss
4. Organisatorischer/administrativer Informationsfluss



Quelle: Tiemeyer (Hg.), Handbuch IT-Projektmanagement (2010)

# Vorteile systematischer Berichtssysteme

- Kontinuierliche Information aller Projektbeteiligten über Projektfortschritt wirkt motivierend
- Transparenz ermöglicht schnelles Erkennen und Lösen von Problemen
- Wissen und Erfahrungen der Beteiligten werden besser nutzbar gemacht (Warum sollen alle nacheinander in die Sackgasse laufen? Einer reicht doch!)
- Weiterbildung in fachfremden Gebieten findet unbewusst statt.

# Berichtsarten (Übersicht)

Zeitorientierte Berichte	Ergebnisorientierte Berichte
Situationsbericht	<i>Ad hoc</i> Bericht (Sofort-, Ausnahme-, Blitzbericht)
Statusbericht	Phasen-Abnahmebericht
Monats-/Quartalsbericht	Projektab schlussbericht
(Arbeitspaketbericht)	Abnahmeprotokoll

Der Situationsbericht soll:

- in **regelmäßigen Abständen**
- einen **bestimmten Adressatenkreis**
- in **knapper und übersichtlicher Form**
- über alle **relevanten Projekttereignisse**

informieren.

**Synonyme:** Projektstandsbericht

Der Statusbericht soll gegenüber dem Situationsbericht:

- ausführlicher sein
- zielgerichtet alle Projektparameter integrieren
- je nach Berichtsebene die notwendigen Informationen beinhalten.

**Synonyme:** Zwischenbericht, Fortschrittsbericht

## Merkmale:

- Lassen sich in der Regel nicht vorhersehen
- Sind häufig Folge einer Abweichung oder einer speziellen Anfrage
- Sind oftmals formlos und telegrammartig
- Verursachen enorm hohen Aufwand / sind sehr kostenintensiv
- Sollten möglichst vermieden bzw. auf ein Minimum beschränkt werden.

**Synonyme:** Sofort-, Ausnahme-, Blitzbericht

## Merkmale:

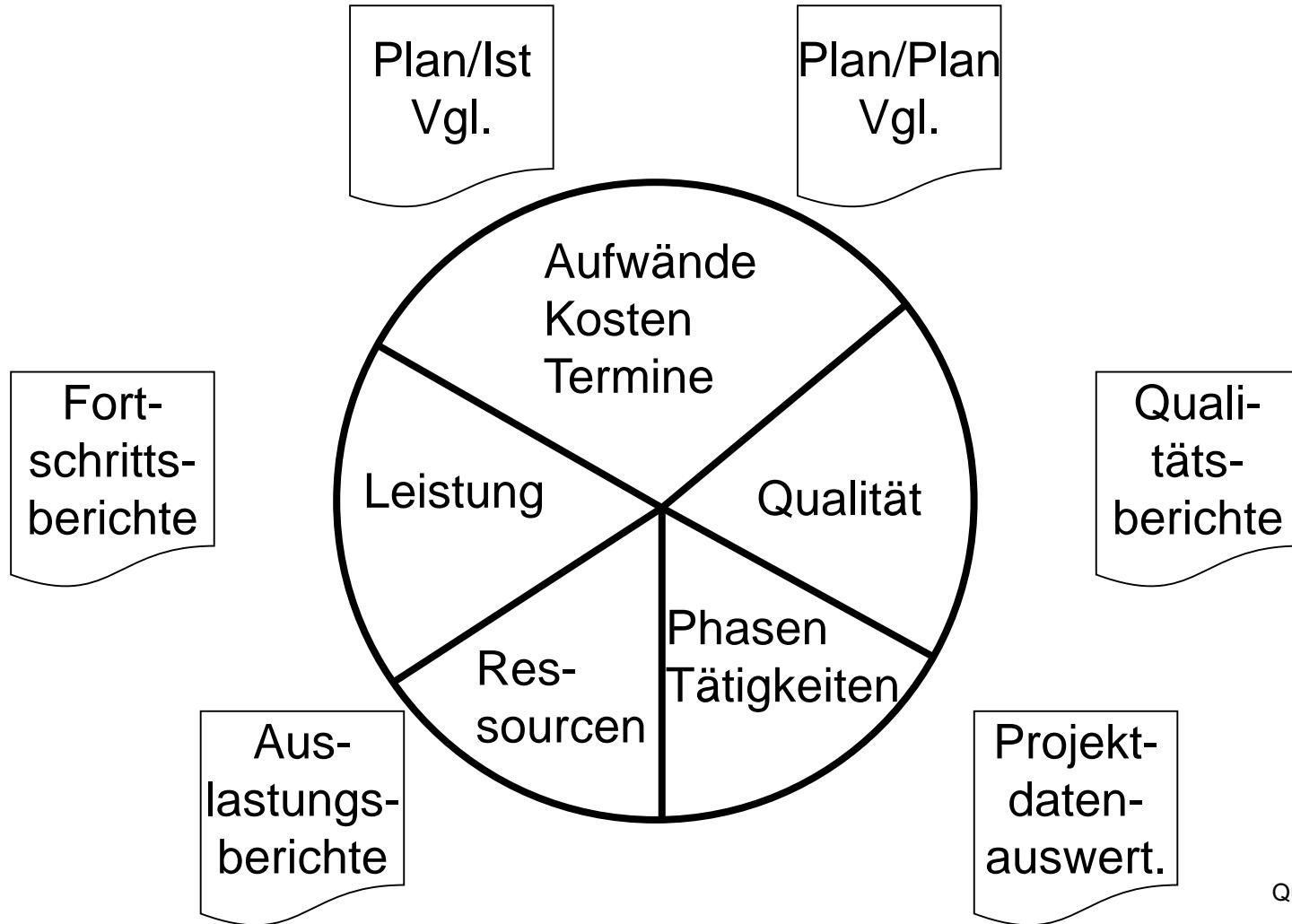
- Überprüfung der abgewickelten Phase (wurden die gesetzten Ziele vollständig erreicht?)
- Falls notwendig sind Abnahmetests durchzuführen
- Beinhaltet die Entscheidung über Fortsetzung des Projektes
- Bei externen Projekten fallen Phasen-Abnahmen und Zahlungszeitpunkte oftmals zusammen

## Der Projektabschlussbericht

- beschreibt das Ergebnis des Gesamtprojektes aus organisatorischer und fachlicher Sicht
- sichert die daraus gewonnenen Erkenntnisse für zukünftige Projekte
- stellt kein Äquivalent zur Projektdokumentation dar
- muss abgenommen werden

Der Projektabschlussbericht unterstützt das bewusste Analysieren von Stärken und Schwächen der Organisation mit dem langfristigen Ziel, die Leistungsfähigkeit des Unternehmens zu verbessern.

# Übersicht zu Projektberichten



Quelle: Burghardt (2002)

# Erkennen von Planabweichungen

---

- Ziel der Projektcontrolle ist das frühzeitige Erkennen von Planabweichungen
- Kontrolle von:
  - Terminen
  - Fertigstellungsgrad (Leistung)
  - Kosten
- Vorsicht vor dem 90%-Syndrom!

- Rückmelderoutinen sind im Projekt zu definieren
  - Wer meldet wem?
  - In welchem Zeitrythmus muss gemeldet werden?
  - Wie werden die gemeldeten Daten aufbereitet? (Templates?)
- Aktionen der Termineinhaltung
  - Einsatz von zusätzlichem Personal
  - Temporäres Erhöhen der Arbeitszeit
    - Überstunden
    - Urlaubssperre
  - Verbesserter Tool- und Methodeneinsatz
  - Optimierung der Arbeitsabläufe
  - Abstriche im Leistungsumfang

# Terminkontrolle: Plan/Ist Vergleiche

---

- Terminübersichten: Keine Terminkontrolle ohne klare Terminübersichten
- Inklusive Rückstandsübersichten bzw. Negativlisten
- Kennzahlen:
  - Plantreue
  - Termintreue

- Man unterscheidet die Plantreue bei Leistungsgrößen (Maximierung) und Lastgrößen (Minimierung)
- Beispiele für Leistungsgrößen:
  - Funktionsumfang
  - Verfügbarkeit
  - Qualität
- Beispiele für Lastgrößen:
  - Kosten
  - Termine

$$PT_{Leistung} = \frac{Y_{V' \text{ Ist}}}{Y_{Plan}} * 100$$

$$PT_{Last} = \left( 2 - \frac{Y_{V' \text{ Ist}}}{Y_{Plan}} \right) * 100$$

$PT_{Leistung}$  = Plantreue einer Leistungsgröße in %  
 $PT_{Last}$  = Plantreue einer Lastgröße in %  
 $Y_{V'Ist}$  = Voraussichtlicher Istwert  
 $Y_{Plan}$  = Planwert

# Termintreue

- Kontrollindex zum Beurteilen der Terminsituation eines Projektes eignet sich der arithmetische Durchschnittswert der terminlichen Plantreue-Quotienten aller Aufgabenkomplexe/Teilprojekte
- Dies wird als Termintreue des Gesamtprojektes bezeichnet

$$TT_{TP} = \frac{T_{Plan} - T_{\Delta}}{T_{Plan}} * 100$$

TT<sub>TP</sub> = Termintreue eines Teilprojektes in %  
T<sub>Plan</sub> = Geplante Dauer  
T<sub>Δ</sub> = Terminverzug

$$TT_{Ges} = \frac{\sum TT_{TP}}{n_{TP}} * 100$$

TT<sub>Ges</sub> = Termintreue des Gesamtprojektes in %  
n<sub>TP</sub> = Anzahl Teilprojekte  
T<sub>V'Ist</sub> = Voraussichtliche Dauer

$$T_{\Delta} = T_{V'Ist} - T_{Plan}$$

# Terminkontrolle: Plan/Plan Vergleiche

---

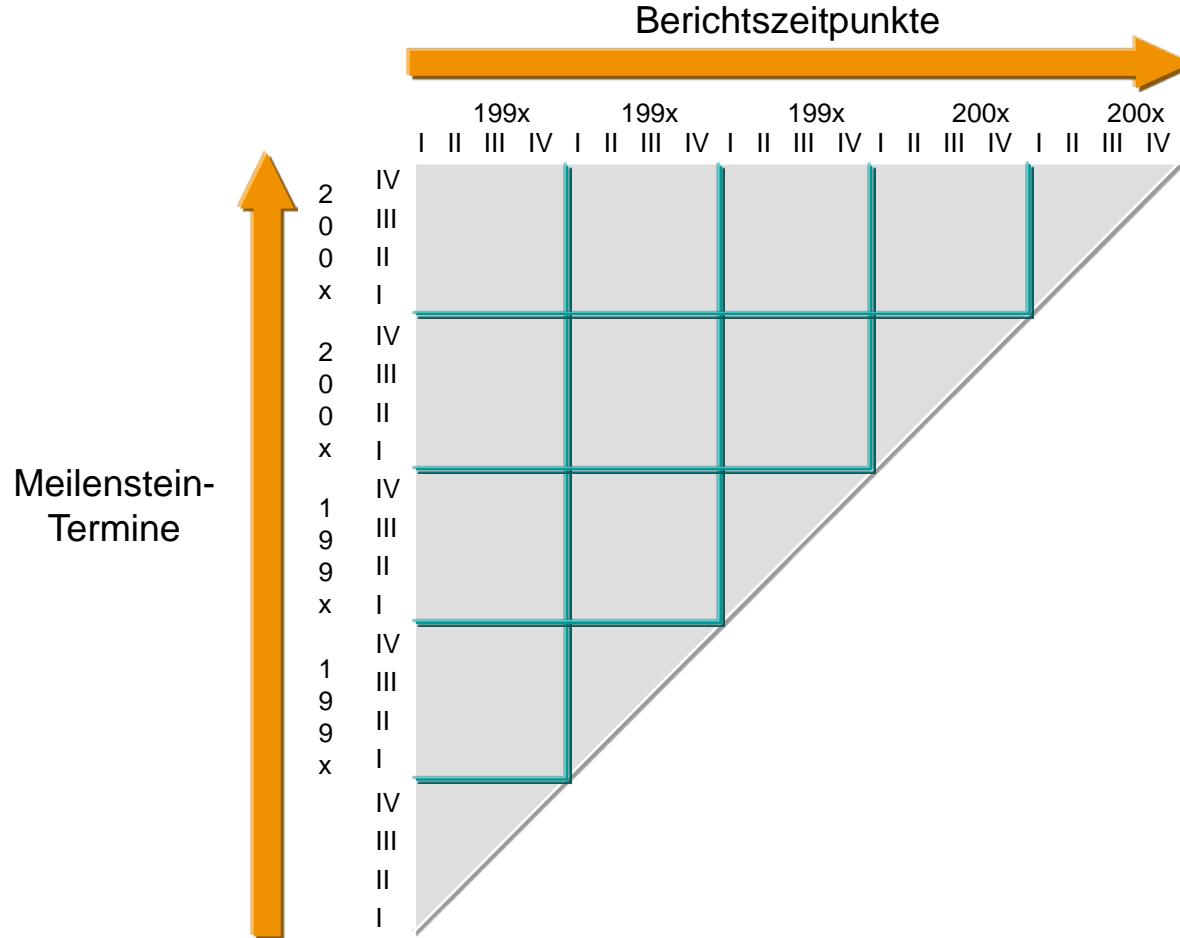
- Dynamisierung der Betrachtungsweise
- Trendanalysen
- Verfahren:
  - MTA

# Meilensteintrendanalyse (MTA)

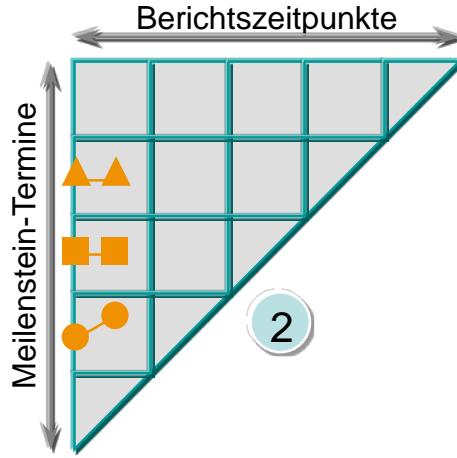
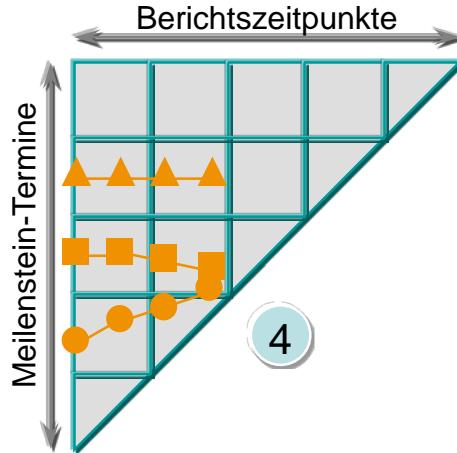
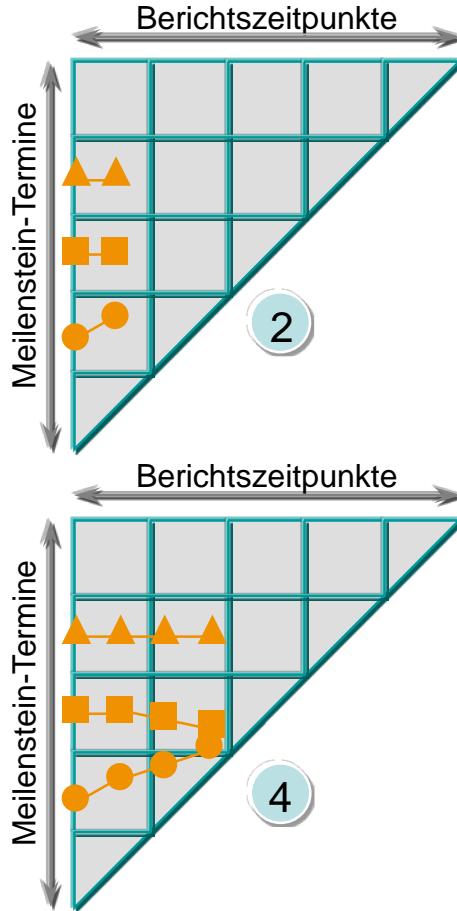
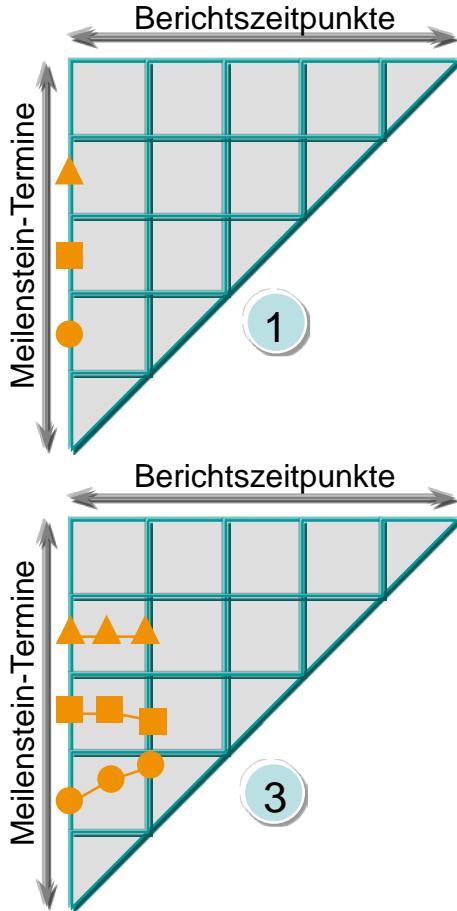
---

- Plan/Ist-Betrachtungen sind statisch
- Trendanalysen sind Plan/Plan-Vergleiche
- Die Planungstermine projektentscheidender Arbeitsvorgänge oder –ereignisse werden in einem Entwicklungsablauf gezeigt
- Meilensteine eignen sich hierfür sehr gut
- „Dreiecksraster“
- Unterschiedliche charakteristische Kurvenverläufe lassen sich beobachten:
  - Normaler Verlauf
  - Extrem ansteigender Verlauf
  - Trendwende-Verlauf
  - Divergierender Verlauf
  - Gleichmäßig fallender Verlauf
  - Zick-Zack-Verlauf

# MTA-Grafisch (I)



# MTA-Grafisch (II)



- 1 Ausgangssituation nach Planung
- 2 Erste Projektbesprechung mit Terminkontrolle nach einem Monat
- 3 Zweite Projektbesprechung mit Terminkontrolle nach zwei Monaten
- 4 Dritte Projektbesprechung mit Terminkontrolle nach drei Monaten

- Vorteile
  - Einfach
  - Schnell zu erstellen
  - Übersichtlich
  - Sehr gute Eignung als Kommunikationsmittel
  - Schärft das Terminbewusstsein
- Nachteile
  - Subjektive Schätzungen
  - Trendkurve allein reicht nicht
  - Kommentare sind erforderlich

# Trendanalysen für Aufwand/Kosten

---

- Neben den Terminen sind die Kosten unbedingt zu extrapolieren (PM-Dreieck!)
- Die aktuellen Ist-Kosten sollten periodisch ermittelt werden und in die Prognose einfließen
- Plan/Plan Vergleich lässt Erfolg der Steuerungsmaßnahmen schnell sichtbar werden

# Fortschrittsmessung I

---

$$FW_{Gesamt} = FGR_{IST}^{Gesamt} * Plankosten_{Gesamt}$$

$$FW_{Gesamt} = \sum Fertigstellungswerte$$

$FW$  = Fertigstellungswert

$FGR$  = Fertigstellungsgrad

# Fortschrittsmessung: Beispiel

---

Ein Projekt hat eine Gesamtdauer von 20 Monaten und die geplanten Kosten betragen 400.000 €. Nachdem 16 Monate der Projektlaufzeit verstrichen sind, schätzt der PL den Fertigstellungsgrad auf 90%. Bis zu diesem Zeitpunkt sind Ist-Kosten i.H.v. 350.000 € angefallen.

$$\begin{array}{ll} \text{Fertigstellungswert} & = 400.000 \text{€} \times 0,9 = 360.000 \text{ €} \\ \text{Ist-Kosten} & = 350.000 \text{ €} \end{array}$$

Schlussfolgerung PL: Projekt liegt im Kostenrahmen; keine Maßnahmen  
Situation nach 18 Monaten: PL schätzt den Fertigstellungsgrad weiterhin auf 90%. Bis zu diesem Zeitpunkt sind weiter Ist-Kosten i.H.v. 15.000 € /Monat angefallen.

$$\begin{array}{ll} \text{Fertigstellungswert} & = 400.000 \text{€} \times 0,9 = 360.000 \text{ €} \\ \text{Ist-Kosten} & = 350.000 \text{ €} + 2 \times 15.000 \text{ €} = 380.000 \text{ €} \end{array}$$

Schlussfolgerung PL: Projekt ist stark gefährdet; Maßnahmen notwendig!

---

## Messtechniken für den Fortschrittsgrad:

1. Statusschritte (z.B. Meilensteine)
2. 50-50-Verfahren (komplexere Aktivitäten oder Aktivitäten mit umfangreichen Vorarbeiten)
3. 0-100-Verfahren (Aktivitäten von kurzer Dauer wie z.B. Abnahmen)
4. Mengenproportionalität (Materiallieferungen)
5. Schätzung (sehr häufige Methode; nicht empfohlen, da subjektiv))
6. Zeitproportionalität (Projektleitung)

# Prognoseverfahren

- Lineare Hochrechnung
- Additive Hochrechnung
- „Erwartet = Plan“

$$EK = PK * \frac{IK}{FW}$$

$$EK = PK + (IK - FW)$$

$$EK = PK$$

$$EK = Kosten_{Erwartet}^{Gesamt} \quad (\text{zum Ende})$$

$$PK = Kosten_{Plan}^{Gesamt} \quad (\text{zum Ende})$$

$$IK = Kosten_{Ist}^{Gesamt} \quad (\text{zum Stichtag})$$

- Sind in manchen Projekten die halbe Dokumentation.
- Müssen nicht nur den Status des Projektes enthalten, sondern auch:
  - Offene-Punkte Liste / To-do-Liste (**Wer macht was bis wann?**)
  - Probleme und kritische Punkte kennzeichnen
- Müssen auch ABGENOMMEN werden!
- Stellen die Basis für wichtige Projektentscheidungen dar.

## Ziel: Stets aktueller Überblick

### Was wird berichtet?

- ◆ Termine
- ◆ Kosten
- ◆ Leistung, Qualität
- ◆ Kapazität

 Harte Daten

- ◆ Probleme
- ◆ Motivation
- ◆ Risikoerwartung
- ◆ Verhalten des Auftraggebers

 Weiche Daten

### Wie wird berichtet?

- ◆ MTA
- ◆ Projektberichterstattung
- ◆ Meilensteine, Arbeitspakete, QS-Berichte, KM- Control

 Monatsbericht  
 TOP Bericht

- ◆ Verbal
- ◆ Projektberichterstattung

# Merkmale eines guten Berichtswesens

- Standardisiert und schematisiert  
(Vorgaben und Vorlagen/Formulare)
- Eindeutig und übersichtlich  
(Visualisierungsmöglichkeiten nutzen)
- Zeithnah und aktuell
- Es wird verwendet, weil es nützlich ist,  
nicht weil man es verwenden muss!

- Qualitätsmanagement betrifft doch Prozesse, oder?
- Was sagt eine Zertifizierung aus?
- Führt ein Qualitätsmanagement zu einem Wettbewerbsvorteil?

## Galaxy Note7

### Samsung zertifizierte Pannenakkus selbst

Um die Sicherheit seiner Handyakkus zu prüfen, hat sich Samsung beim brandgefährlichen Galaxy Note7 auf ein eigenes Labor verlassen. Zukünftig sollte man die Qualitätssicherung verbessern, gelobt das Unternehmen.



Login | Registrierung

**SPIEGEL ONLINE POLITIK**

Politik | Wirtschaft | Panorama | Sport | Kultur | Netzwerk | Wissenschaft | Gesundheit | einestages | Karriere | Uni | Reise | Auto | Stil

Nachrichten > Politik > Ausland > Flugzeugunglücke > Airbus A400M: Militärmaschine stürzte wegen Software-Problemen ab

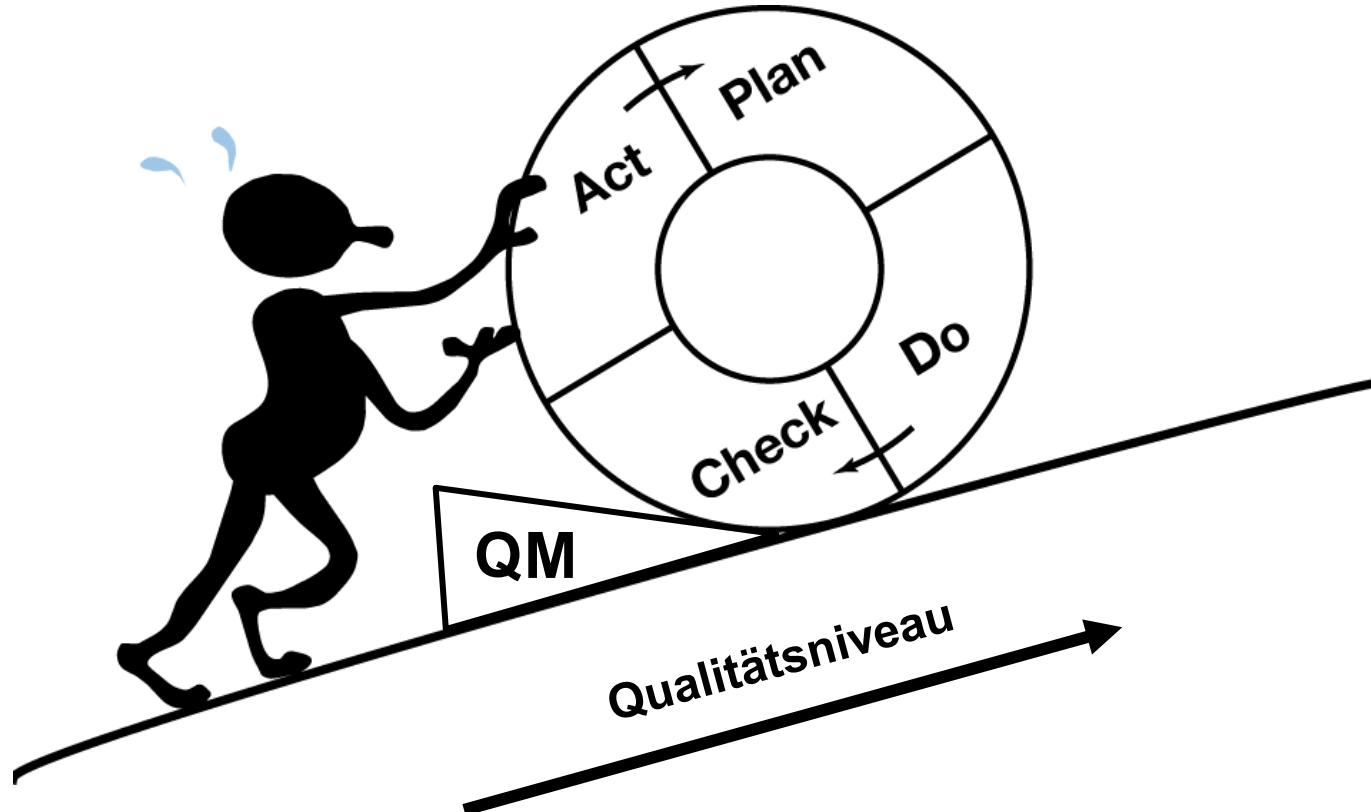
### Tödlicher Unfall des A400M: Airbus-Flieger stürzte wegen Software-Problemen ab

Von Gerald Traufetter und Matthias Gebauer



Abgestürzter Airbus A400M: Softwareprobleme führten zum Crash

# PDCA-Cycle



# Ansätze und neuere Modelle für das Qualitätsmanagement

---



- ISO 9000
- EFQM
- CMMI

- Normenreihe
- Wurde bisher bis ISO 9004 ausgebaut
- Grundsätze für Maßnahmen zum Qualitätsmanagement
- Es gibt verschiedene branchenspezifische Erweiterungen (z.B. die ISO/TS16949 für die Automobilindustrie)

ISO 9004  
Leitfaden zur Leistungsverbesserung

ISO 9001  
Grundlegende Anforderungen

ISO 9000  
Grundlagen und Begriffe

# ISO 9001:2008 und ISO 9004:2009

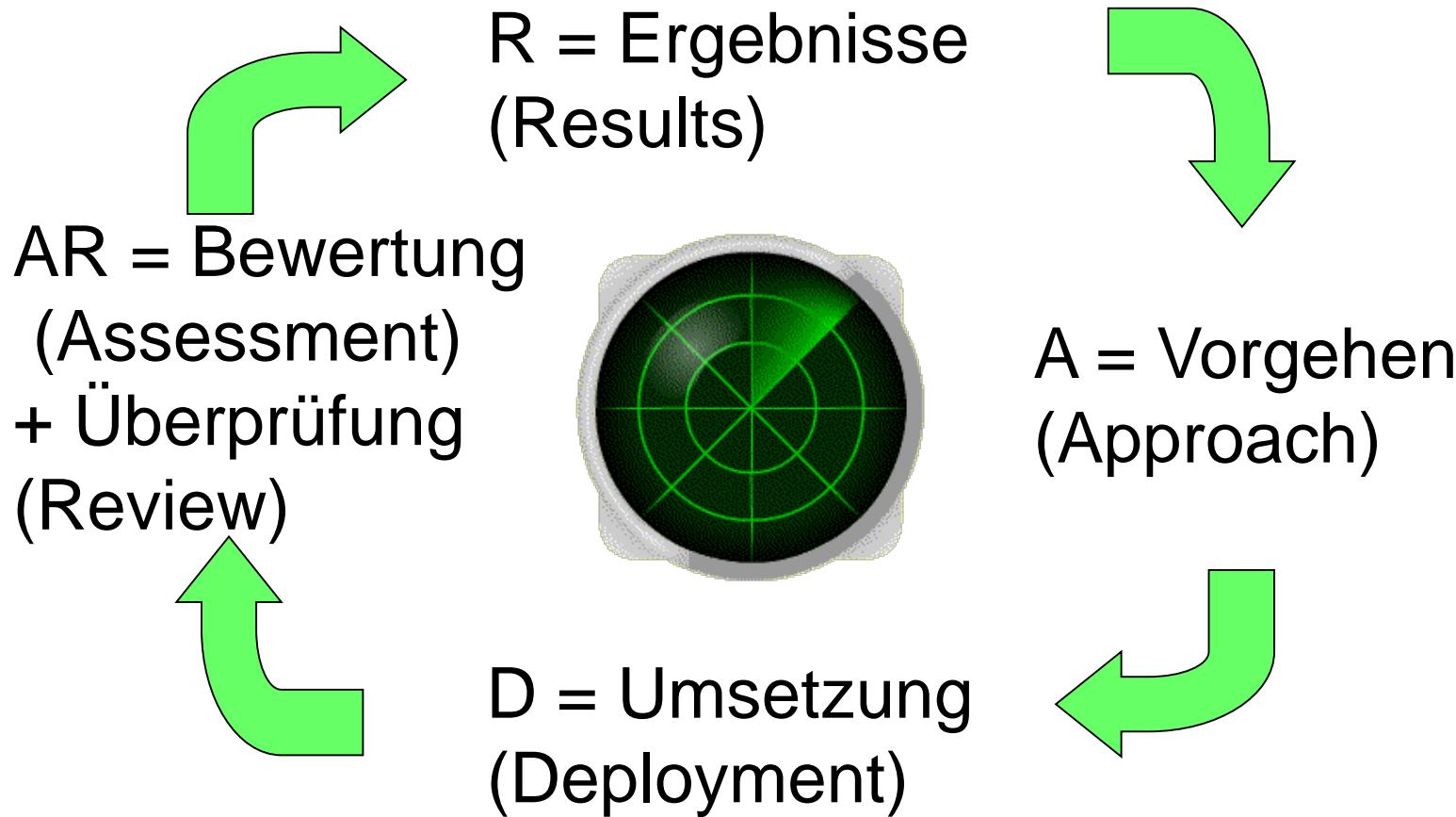


- ISO 9001:2008:
  - Mindeststandard („gut genug“-Modell)
  - Kann zertifiziert werden
  - Neue Version seit 2015 (ISO 9001:2015-09)
- ISO 9004:2009:
  - Ein „immer besser“-Modell
  - Wird nicht zertifiziert!
  - Interne Audits sind notwendig, um Fortschritt des Managementsystems zu validieren
- Der übliche Weg: Erst ISO 9001 Zertifizierung erlangen und sich dann auf die „Reise der Leistungsverbesserung“ begeben.
- Aber: Viele Wege führen nach Rom!

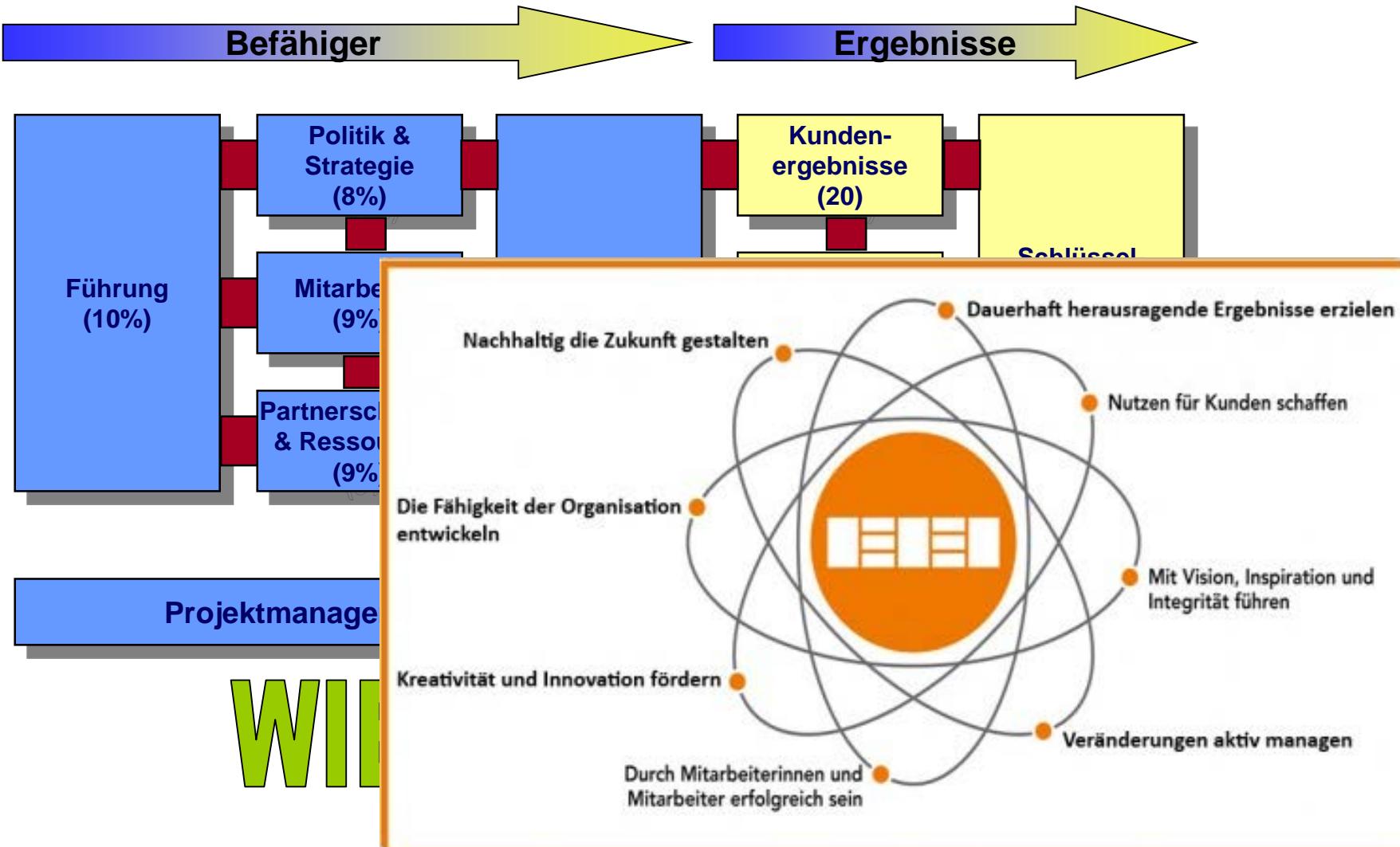
- Basiert auf dem Total Quality Management (TQM) Ansatz
- Ist als Hilfestellung bei der Umsetzung von TQM zu sehen
- Die European Foundation for Quality Management (EFQM) wurde 1988 gegründet

- Der *European Quality Award* ist eine Antwort auf die Initiativen der Konkurrenz aus Japan (Deming Preis; seit 1951) und den USA (Malcom Baldrige National Quality Award)
- 3-Säulen-Konzept:
  - Kunden
  - Mitarbeiter
  - Ergebnisse

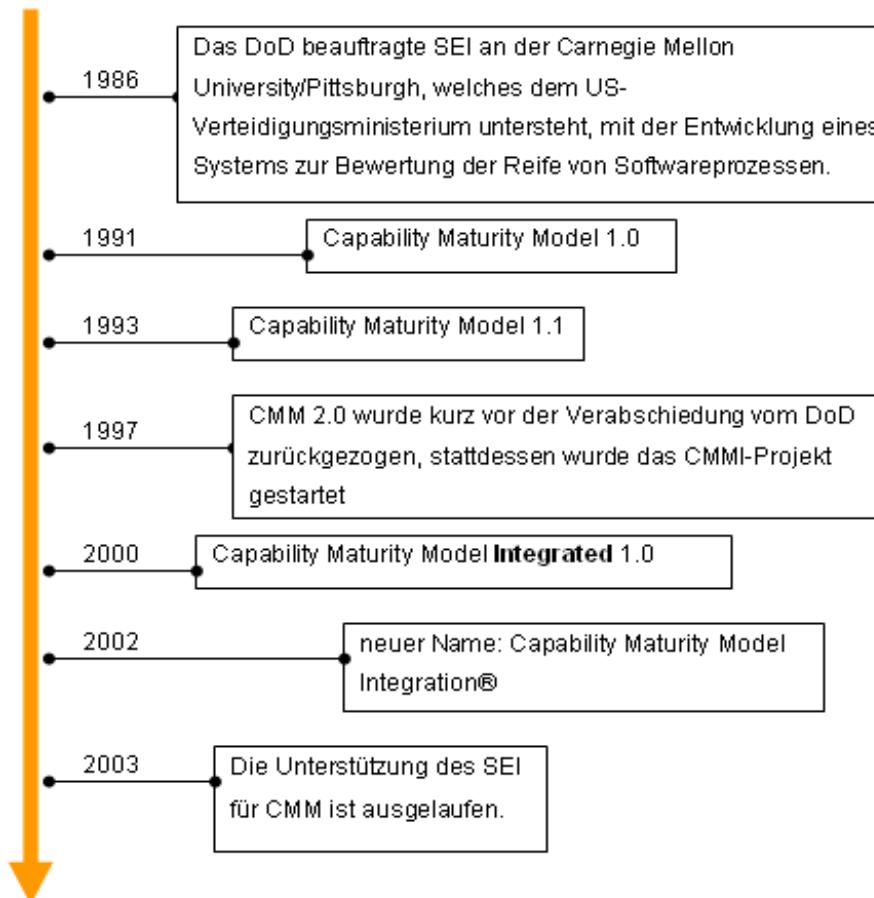
## Self-Assessment mit der RADAR-Methodik:



# EFQM-Modell 4 Project Excellence

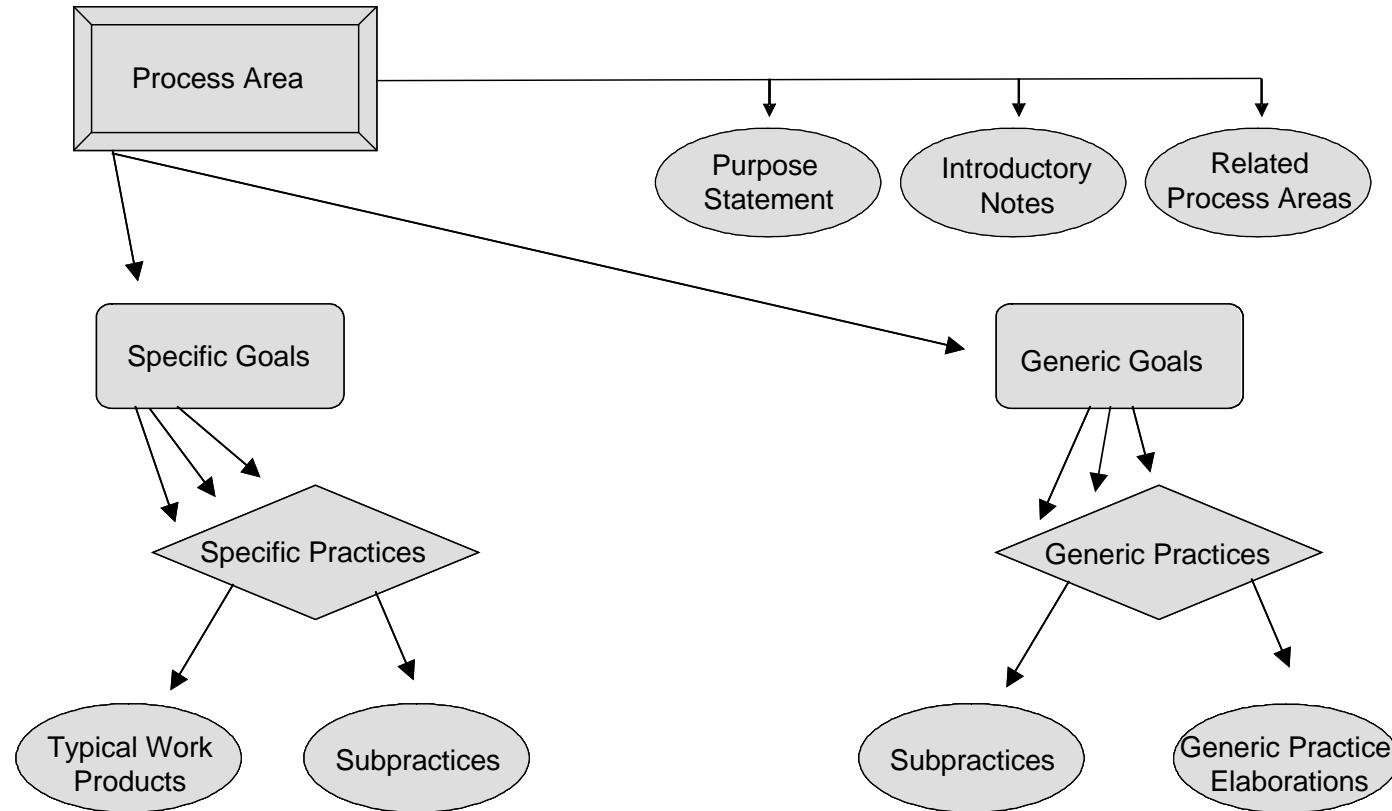


## • Capability Maturity Model Integration



Quelle: Breitenberger (Seminararbeit)

# Komponenten des CMMI-Modells



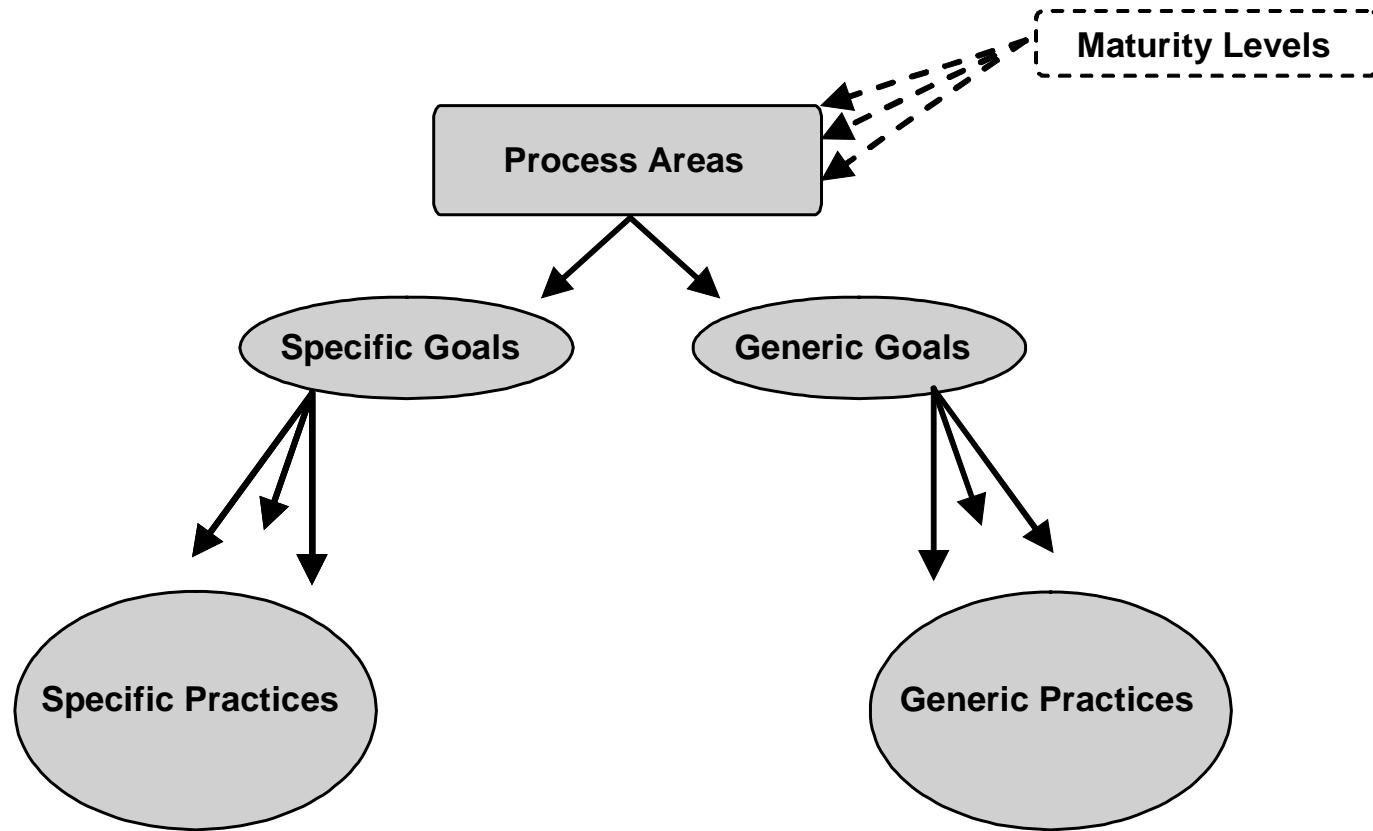
Prozess-Kategorie	Prozessgebiet	
Prozess-Management	Organisationsweite Prozessdefinition (Organizational Process Definition) Organisationsweiter Prozessfokus (Organizational Process Focus) Organisationsweites Training (Organizational Training) Performance organisationsweiter Prozesse (Organizational Process Performance) Organisationsweite Innovation und Verbreitung (Organizational Innovation and Deployment)	OPD OPF OT OPP OID
Projekt-Management	Projektplanung (Project Planning) Projektverfolgung und –steuerung (Project Monitoring and Control) Management von Lieferantenvereinbarungen (Supplier Agreement Management) Integriertes Projektmanagement (Integrated Project Management) Risikomanagement (Risk Management) Quantitatives Projektmanagement (Quantitative Project Management)	PP PMC SAM IPM RSKM QPM
Entwicklung	Anforderungsmanagement (Requirements Management) Anforderungsentwicklung (Requirements Development) Technische Umsetzung (Technical Solution) Produktintegration (Production Integration) Verifikation (Verification) Validation (Validation)	REQM RD TS PI VER VAL
Unterstützung	Konfigurationsmanagement (Configuration Management) Qualitätssicherung von Prozessen und Produkten (Process and Product Quality Assurance) Messung und Analyse (Measurement and Analysis) Entscheidungsanalyse und –findung (Decision Analysis and Resolution) Ursachenanalyse und Problemlösung (Causal Analysis and Resolution)	CM PPQA  MA DAR CAR

# CMMI Reifegrad-Niveaus (staged representation)

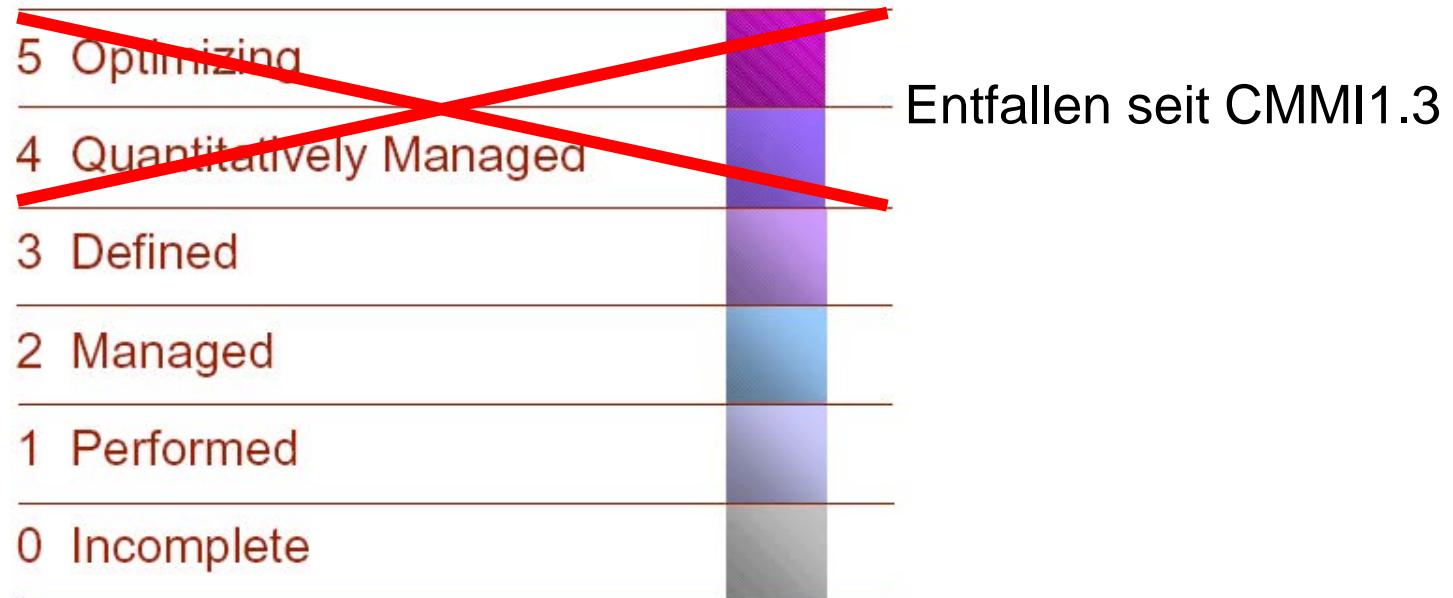


- Hohes Reifegrad-Niveau → hohe Produktivität und Qualität  
(Abhängigkeit vom Mitarbeiter sinkt)
- Niedriges Reifegrad-Niveau → hohes Projektrisiko  
(Abhängigkeit zwischen Erfolg und Mitarbeiter)

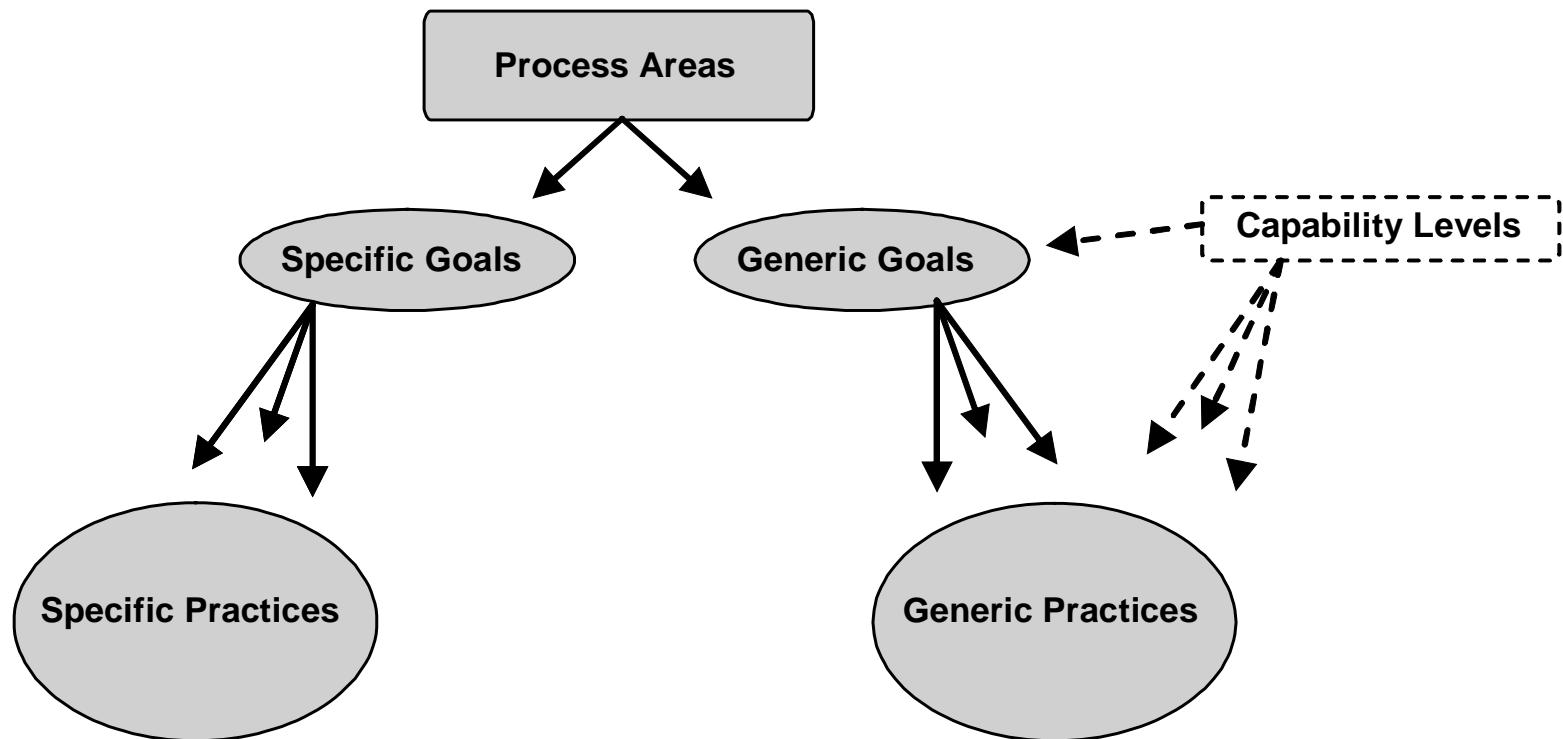
# Maturity Levels



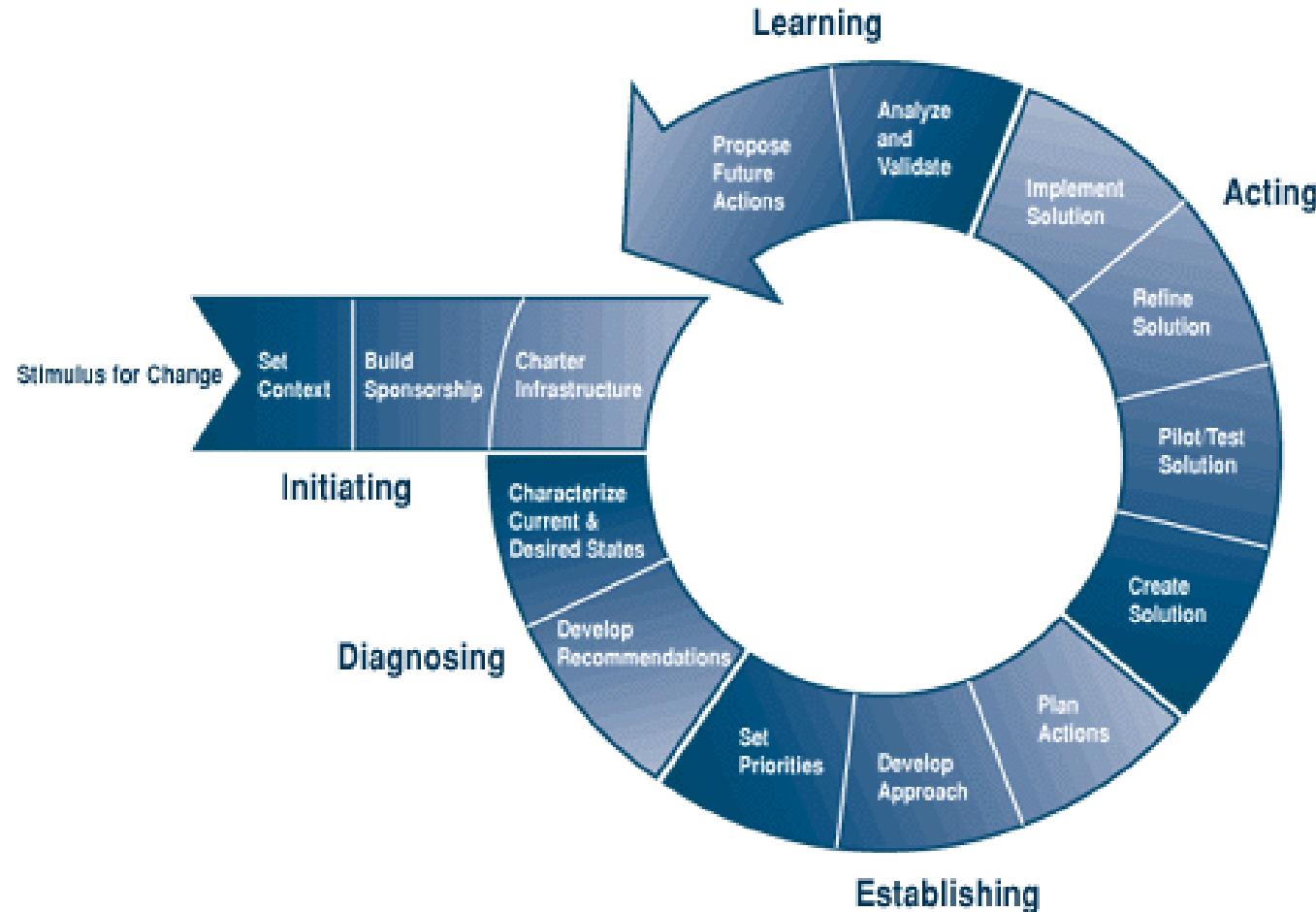
# CMMI Capability Levels (continuous representation)



# Capability Levels



# Das IDEAL-Modell des SEI



# Die Begutachtungen (Appraisals)

---

- Dienen der Ermittlung des Reifegrad-Niveaus, auf dem sich das jeweilige Unternehmen befindet.
- SCAMPI: Standard CMMI Appraisal Method for Process Improvement
- Eine Selbstbewertung (self-assessment) ist möglich!
- Es gibt 3 Klassen von Assessments:
  - Class A (SCAMPI)
  - Class B (“abgespeckte” Variante)
  - Class C (für häufige und schnelle Überprüfungen)

# Voraussetzungen für erfolgreiche Projektumsetzung



<b>Vision Ziel</b>	+	<b>Qualifikation</b>	+	<b>Belohnung</b>	+	<b>Ressourcen</b>	+	<b>Aktionsplan</b>	=	<b>erfolgr. Projekt- umsetzung</b>
<del>Vision Ziel</del>	+	<b>Qualifikation</b>	+	<b>Belohnung</b>	+	<b>Ressourcen</b>	+	<b>Aktionsplan</b>	=	<b>Verwirrung</b>
<b>Vision Ziel</b>	+	<del>Qualifikation</del>	+	<b>Belohnung</b>	+	<b>Ressourcen</b>	+	<b>Aktionsplan</b>	=	<b>Unsicherheit Angst</b>
<b>Vision Ziel</b>	+	<b>Qualifikation</b>	+	<del>Belohnung</del>	+	<b>Ressourcen</b>	+	<b>Aktionsplan</b>	=	<b>langsafter, zäher Wandel</b>
<b>Vision Ziel</b>	+	<b>Qualifikation</b>	+	<b>Belohnung</b>	+	<del>Ressourcen</del>	+	<b>Aktionsplan</b>	=	<b>Frustration</b>
<b>Vision Ziel</b>	+	<b>Qualifikation</b>	+	<b>Belohnung</b>	+	<b>Ressourcen</b>	+	<del>Aktionsplan</del>	=	<b>Fehlstart</b>