Vorgehensmodelle / Projektmethoden

Lufthansa-Reservierung in den 60-er Jahren

Schwierigkeiten bei der Softwareentwicklung

Warum wurde die Softwareentwicklung immer schwieriger?

- Die ständig steigende **Leistung der Hardware** ermöglichte ständig neue Anforderungen an Softwareprogramme

  → Softwaresysteme wurden immer komplexer

- Konzepte, wie große Entwicklungsaufgaben auf **viele Leute** aufgeteilt werden können, waren zunächst nicht vorhanden

  → Vorgehensmodelle mussten entwickelt und verbreitet werden

- Wegen **Mangel an qualifiziertem Personal** explodierten die Kosten für große Softwaresysteme

  → Es mussten Verfahren erfunden werden, wie Termine, Qualität und Kosten unter Kontrolle gehalten werden

Erfahrungen mit Softwareprojekten (ca. 1995)

Im Chaos-Report wurde 1995 von der Standish Group eine Untersuchung von 8.000 IT-Projekten veröffentlicht:

- ~ **31%** aller IT-Projekte wurden **abgebrochen**
- ~ **53%** aller IT-Projekte **dauerten doppelt so lange** und **lieferten nur die Hälfte** der geforderten Funktionalität (= Kostenfaktor 4)
- nur **~ 16%** aller IT-Projekte waren termingerecht, im Budget und liefern annähernd das Gewünschte (sogenannte **„erfolgreiche“ Projekte**)

Was ist ein Vorgehensmodell?

Ein Vorgehensmodell ist eine (mehr oder weniger) **genaue Anleitung**, in welchen **Schritten und durch welche Tätigkeiten** das Projektziel erreicht werden kann.

Alternative Bezeichnung: **Projektmethode**

Ein Vorgehensmodell liefert typischerweise **Festlegungen** für:

a) Projektphasen mit Meilensteinen
b) Rollen und Verantwortlichkeiten
c) Aufgaben / Aktivitäten
d) Arbeitsergebnisse
e) Einheitliche Begriffe
f) QS-Maßnahmen
g) Evtl. Methoden, Techniken, Werkzeuge, Richtlinien / Standards

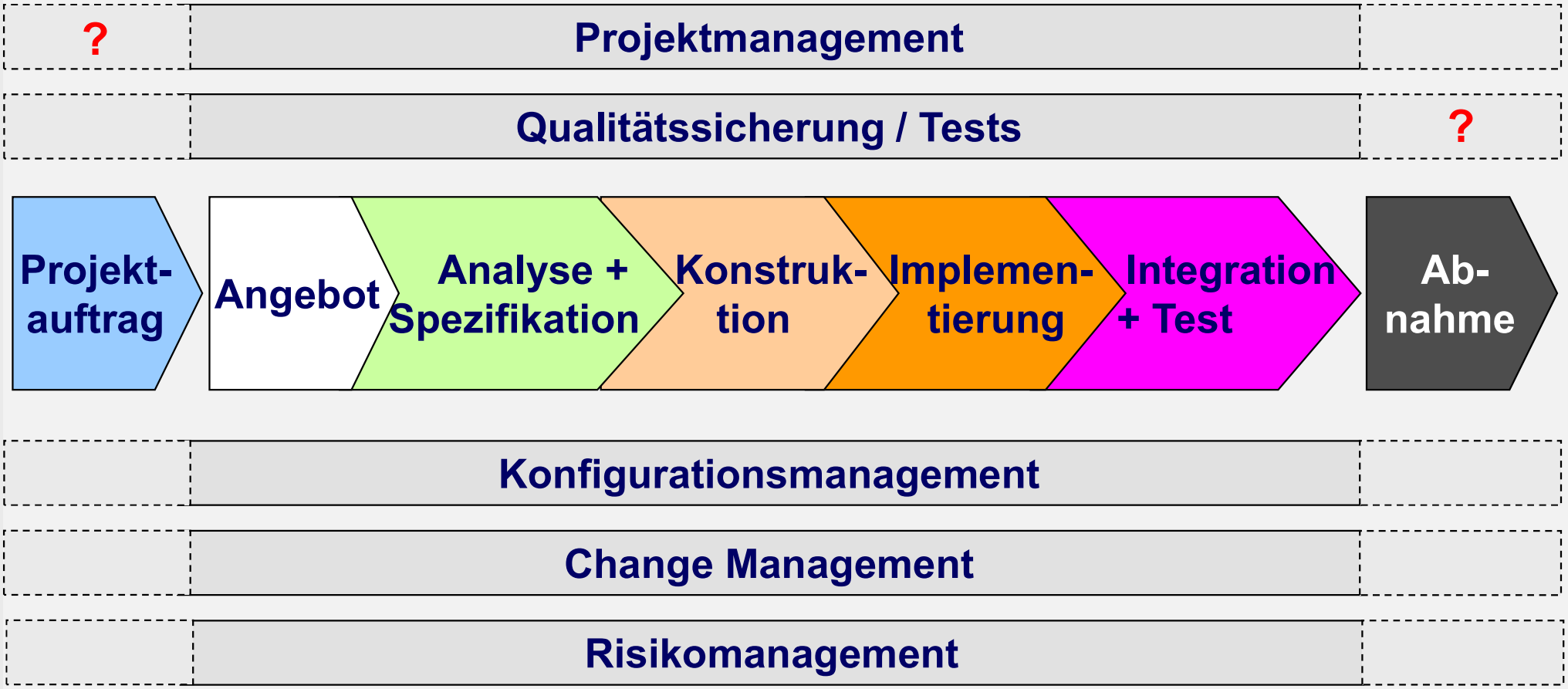
Nutzen eines Vorgehensmodells

- Erhöhung der **Übersichtlichkeit** der Projektdurchführung
- Steigerung der **Beherrschbarkeit**
- Steigerung der **Planbarkeit**
- **Kontrollierte** und (weitgehend) **einheitliche Durchführung** des Projekts
- Verbesserte **Kommunikation** im Projekt
- **Senkung von Aufwänden**
- Frühzeitige **Erkennung von Fehlern**
- Verbesserte **Dokumentation** im Projekt
- Erzielung einer höheren **Qualität** von Projektergebnissen
- **Minimierung** von **Projektrisiken**
- Möglichkeit, **Erfahrungen** zum Vorgehen zu sammeln und **zu verbessern**

→ **Insgesamt höhere Wahrscheinlichkeit, dass das Projekt innerhalb festgelegter Qualität, verfügbarem Budget und zum Termin fertig wird**

Allgemeine Ansätze von Vorgehensmodellen

- Aufteilung in **Phasen** (oft auch detaillierte Beschreibung der Phasen)
- Anleitungen für die **Querschnittsthemen** (PM, QS, KM, CM, RM, …)



Vorgehensmodelle – Beispiele (1)

I. Herkömmliche Vorgehensmodelle

- Wasserfallmodell
- Rational Unified Process
- V-Modell XT

II. Agile Vorgehensmodelle

- Scrum
- Crystal
- Extreme Programming
- Microsoft Solutions Framework

Vorgehensmodelle – Beispiele (2)

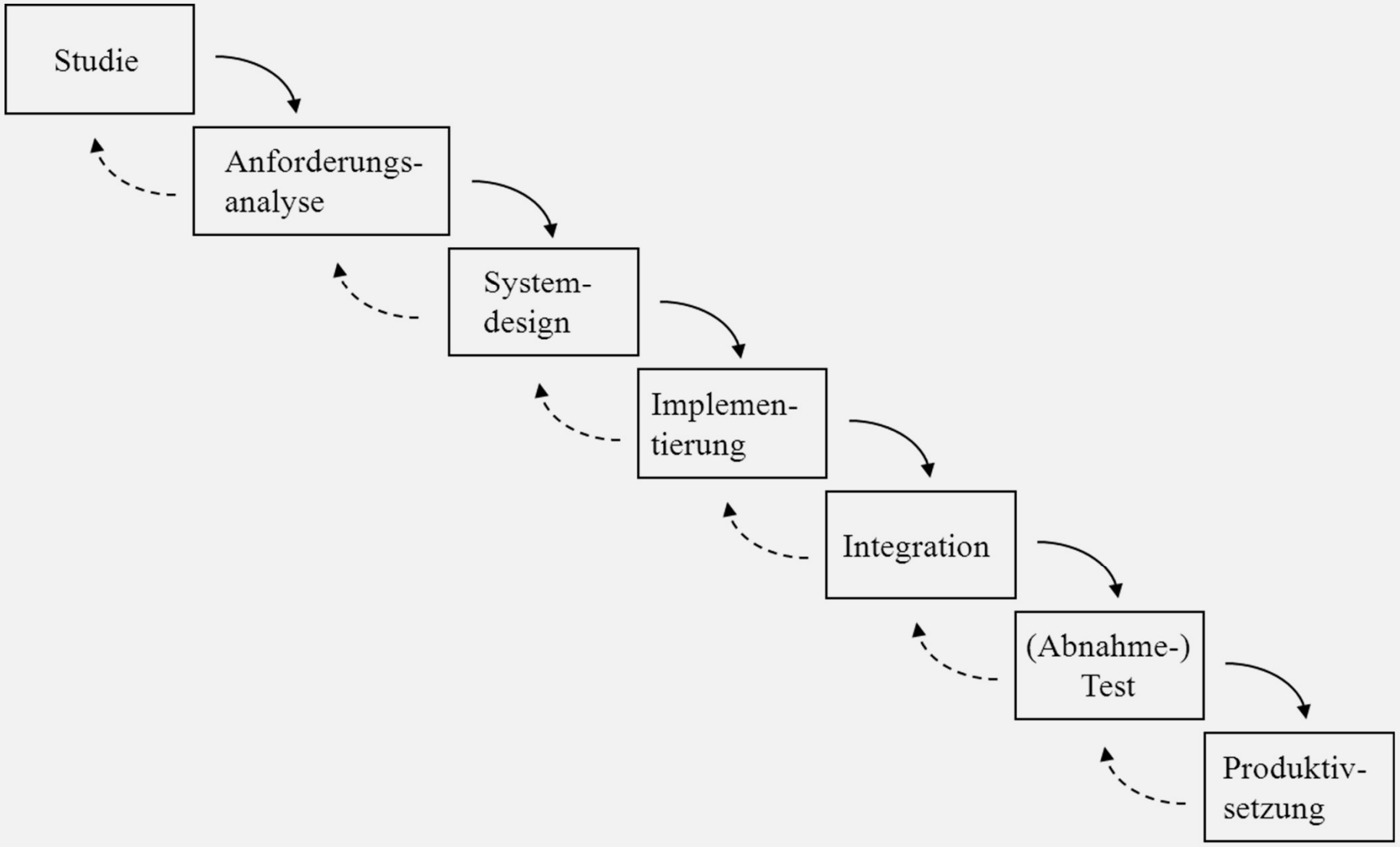
Unternehmensspezifische Prozesse

- ITPM (BMW)
- Aladin (HVB Information Services)
- SE Book + … Books (T-Systems)
- BUP (Bayerische Landesbank)
- SEP (Audi / VW)
- …

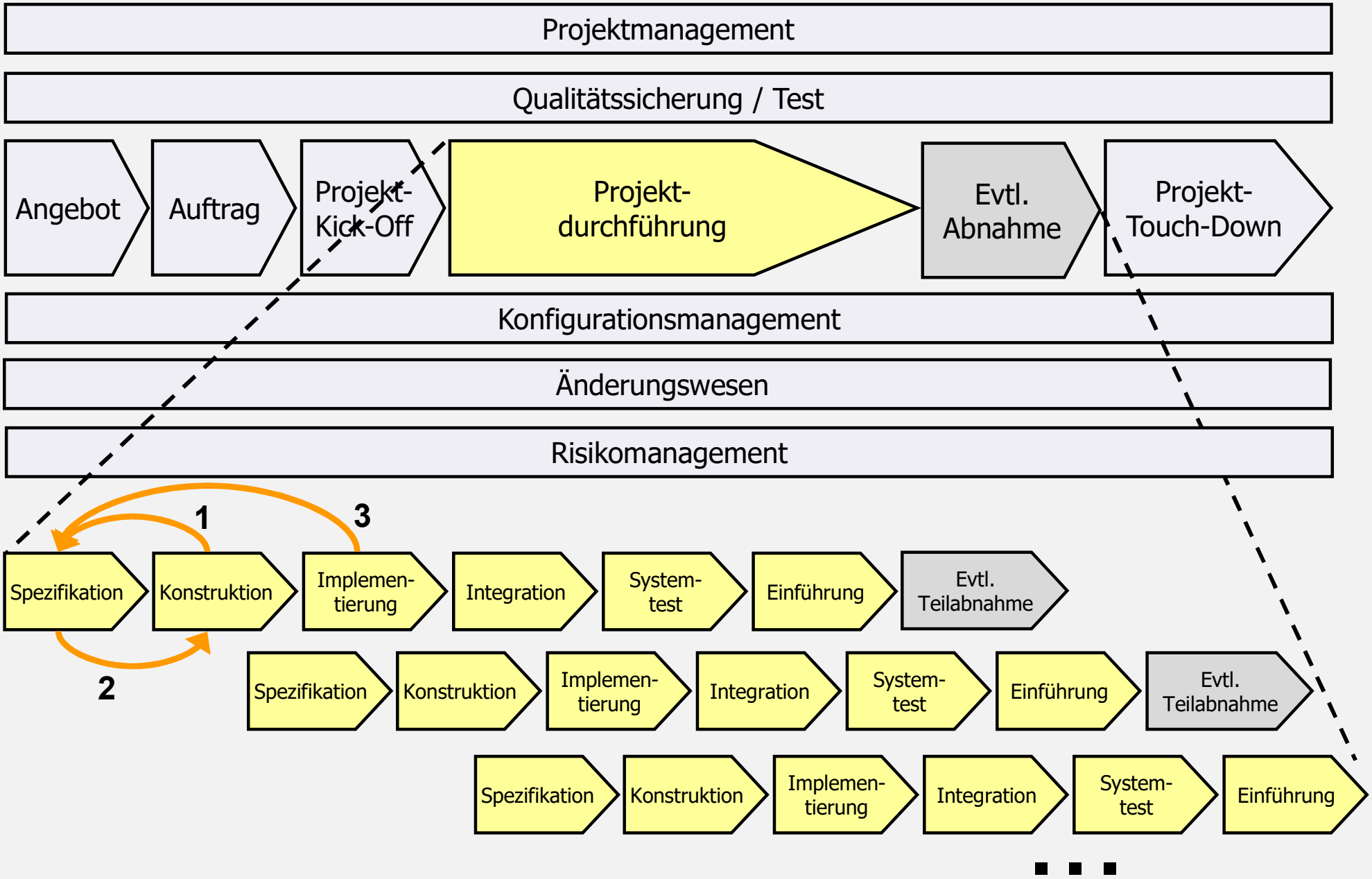
Qualitätsmerkmale von Vorgehensmodellen

- **Vollständigkeit** im Hinblick auf die abzudeckenden Phasen
- Einheitliche und verständliche **Begriffswelt**
- Erfolgreiche **Erprobung** in realen IT-Projekten
- Änderbarkeit und **Erweiterbarkeit**
- **Anpassbarkeit** an verschiedene Projekttypen und Organisationen
- **Skalierbarkeit** hinsichtlich unterschiedlicher Projektgrößen
- Berücksichtigung neuester **Standards**, Vorschriften und Normen
- **Werkzeugunterstützung**
- **Kompatibilität** zu einem organisationsspezifischen **Verbesserungsprozess** für das Vorgehensmodell (CMMI, SPICE, ...)

Wasserfall


Studie
Anforderungs-
analyse
System-
design
Implemen-
tierung
Integration
(Abnahme-)
Test
Produktiv-
setzung

Gestufter Wasserfall mit Iterationen



V-Modell ® XT

Im Internet zu finden unter www.vmodellxt.de (BIT)

- **Nachfolgemodell** zum bekannten **V-Modell ´97**
- Überarbeitet durch die TU München, TU Kaiserslautern, EADS, IABG und Siemens AG
- Für **öffentliche Auftraggeber** empfohlen, aber nicht verpflichtend!

Das V-Modell ® XT enthält:

- Beschreibungen für alle **Projektergebnisse** mit allen **Abhängigkeiten** untereinander
- **Vorgehensweisen** für alle Ergebnisse in allen Projektabschnitten, auch detaillierte Beschreibung von **Aktivitäten**
- **Verantwortlichkeiten** / **Rollen** aller Beteiligten

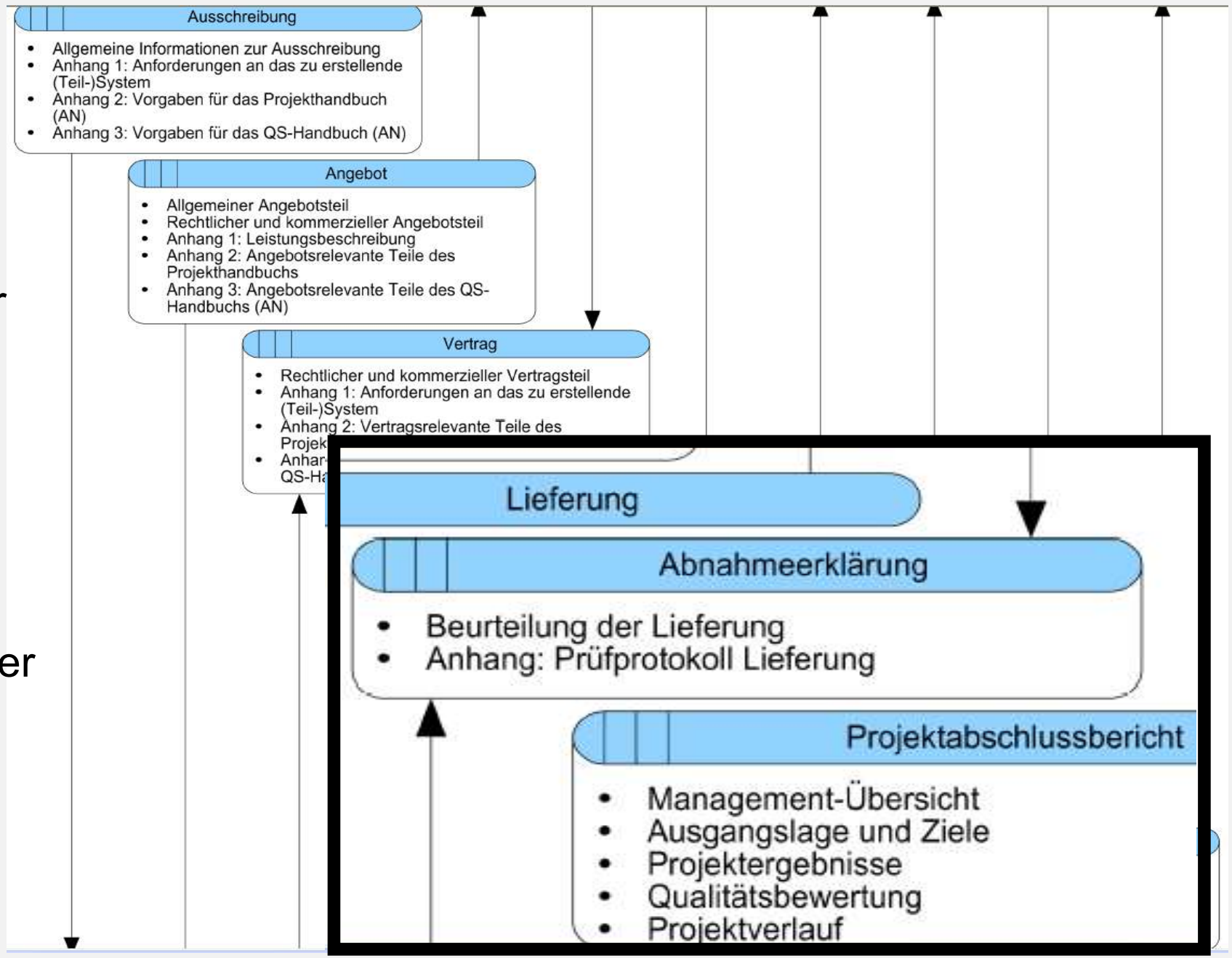
Kernpunkte der V-Modell ® XT Philosophie

- **Projektergebnisse** sind der Dreh- und Angelpunkt des Modells (hier „Produkte“ genannt)
- **Projektdurchführungsstrategien** und Entscheidungspunkte geben die Reihenfolge der Produktfertigstellung und somit die grundlegende Struktur des Projektverlaufs vor
- Die detaillierte **Projektplanung und -steuerung** wird auf der Basis der Bearbeitung und Fertigstellung von Produkten durchgeführt.
- Für jedes Produkt ist eindeutig eine **Rolle** verantwortlich und im Projekt dann eine der Rolle zugeordnete Person
- Die **Produktqualität ist überprüfbar** durch definierte Anforderungen an das Produkt und explizite Beschreibungen der Abhängigkeiten zu anderen Produkten

Schnittstelle Auftraggeber / Auftragnehmer

Auftraggeber

Auftragnehmer



Auswirkungen auf IT-Verträge

- Das V-Modell ® XT hat sich als Bestandteil der EVB-IT-Vertragsmuster für **alle IT-Projekte der öffentlichen Hand** stark durchgesetzt.
- Da ca. 50% des gesamten IT-Projektvolumens in Deutschland von der öffentlichen Hand vergeben wird, hat sich das Modell (bzw. Abwandlungen davon) **auch in der privaten Wirtschaft** etabliert.
- Das **konkrete Vorgehen** im Projekt sollte in jedem Fall durch Verfeinerung / **Tailoring** des V-Modells ® XT genau definiert werden. Tools helfen dabei.
- **IT-Vertrag** und **konkretisiertes Vorgehensmodell** sowie die geplante Art des **Projektmanagements**, **Qualitätsmanagements** und **Änderungsmanagements** sollten **eng verzahnt** werden.

Manifest für agile Softwareentwicklung

1	Individuals and interactions	over	processes and tools
2	Working software	over	comprehensive documentation
3	Customer collaboration	over	contract negotiation
4	Responding to change	over	following a plan

Prinzipien der agilen Softwareentwicklung

1. **Die Zufriedenheit des Kunden** wird durch frühe und kontinuierliche Auslieferung von wertvoller Software erreicht.
2. Agile Prozesse nutzen **Veränderungen** (selbst spät in der Entwicklung) zum Wettbewerbsvorteil des Kunden.
3. Lieferung von **funktionierender Software in regelmäßigen, bevorzugt kurzen Zeitspannen;** der **Projektfortschritt** wird hauptsächlich an der **Menge an lauffähiger Software gemessen**
4. Die Effektivität wird durch **periodische Reviews und Verbesserungen** gesteigert.
5. Die **tägliche Zusammenarbeit** von Fachexperten und Entwicklern ist zwingend notwendig.
6. Motivierten Individuen ist für die Erfüllung ihrer Aufgaben ein **produktives Umfeld und eine hervorragende Unterstützung** zu gewähren.
7. Informationsübertragung nach Möglichkeit **im Gespräch von Angesicht zu Angesicht.**
8. Einhalten eines **gleichmäßigen Arbeitstempos** des Auftraggebers, Entwicklern und Benutzern für eine nachhaltige Entwicklung
9. Ständiges Augenmerk auf **technische Exzellenz und gutes Design – Einfachheit ist essenziell.**
10. **Selbstorganisation der Teams** bei Planung und Umsetzung
11. **Selbstreflexion der Teams** über das eigene Verhalten **zur Anpassung** im Hinblick auf Effizienzsteigerungen

XP (Extreme Programming)

Wesentliche Merkmale (Teil 1):

1) Die Funktionalität des Systems wird in Users Stories zusammengefasst (GUI, Funktionalitäten, Testszenarien)

2) Softwarequalität, Qualitätssicherung
   - Jeweils zwei Entwickler programmieren gemeinsam („programming in pairs“)
   - Gemeinsamer Besitz von Code („collective code ownership“)
   - Ständige Refaktorisierung („continuous refactoring“)
   - Schnelle Code-Reviews („rapid code reviews“)

3) Vor der Entwicklung werden (automatisierbare) Tests erstellt

XP (Extreme Programming)

Wesentliche Merkmale (Teil 2):

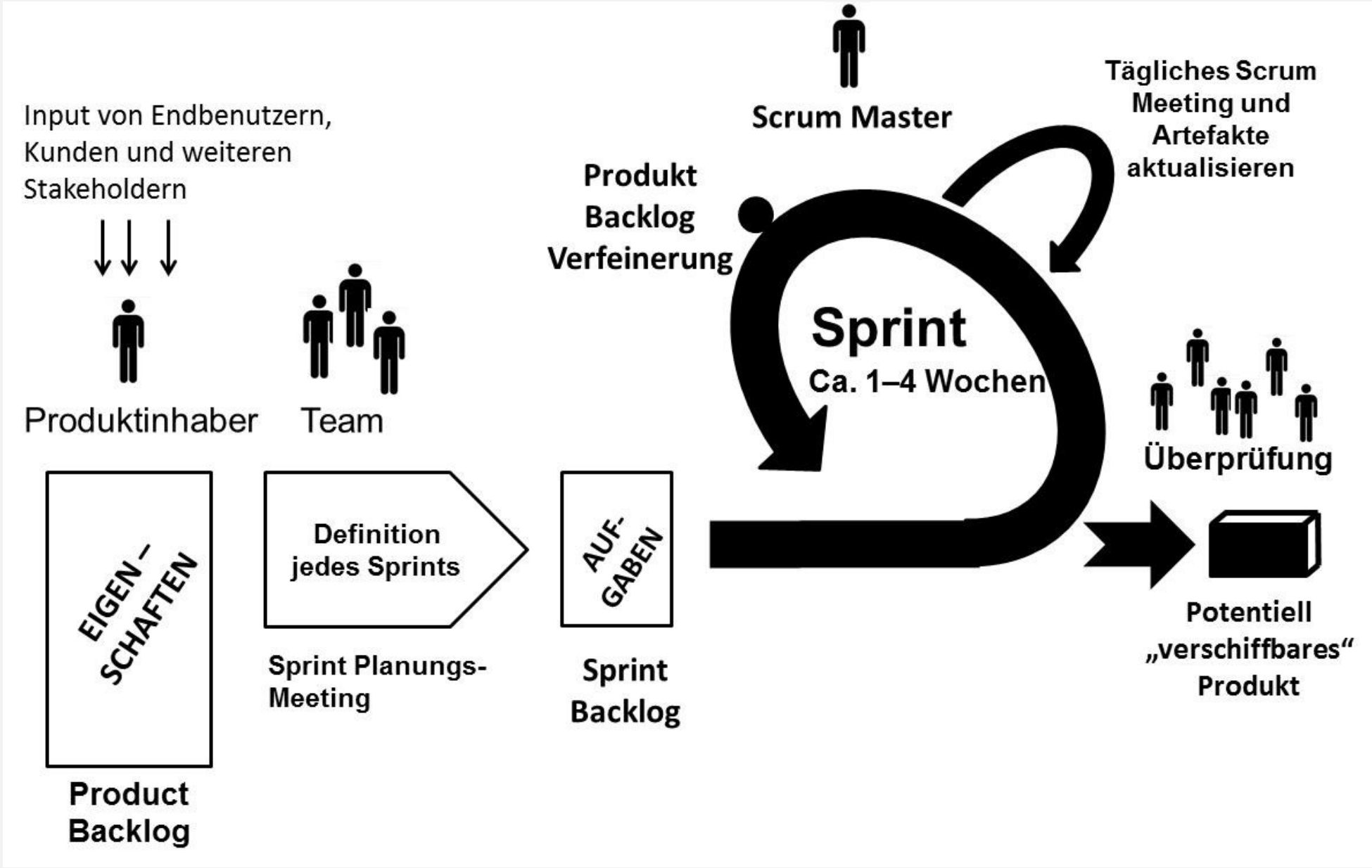
4) Auf unnötige Features wird verzichtet
   (YAGNI - you aren‘t gonna need it)

5) Kunde ist bei der gesamten Entwicklung dabei
   („on-site customer“)

6) Extrem kurze Zyklen für Anforderungsanalyse, Design, Implementierung und Test.
   Das Ergebnis pro Zyklus ist immer ein lauffähiges Programm
   („small releases“)

7) Insgesamt entsteht keine oder nur sehr wenig Dokumentation

Scrum



Quelle: http://javamaster.files.wordpress.com/2009/07/scrum1.png?w=460&h=292, Letzter Aufruf: 15.11.2011

Betrachtung aus rechtlicher Sicht

Fragen:

a. Wie müsste ein IT-Projektvertrag ausgestaltet sein, damit sich zum Beispiel ein **Werkvertrag** ergibt?

b. Gehen die „**Errungenschaften**“ klassischer Projekte beim Übergang zu agilen Methoden verloren?

c. Wie sind **Vergütungsfragen** zu regeln, was ist zum Beispiel mit dem „klassischen“ Festpreis?

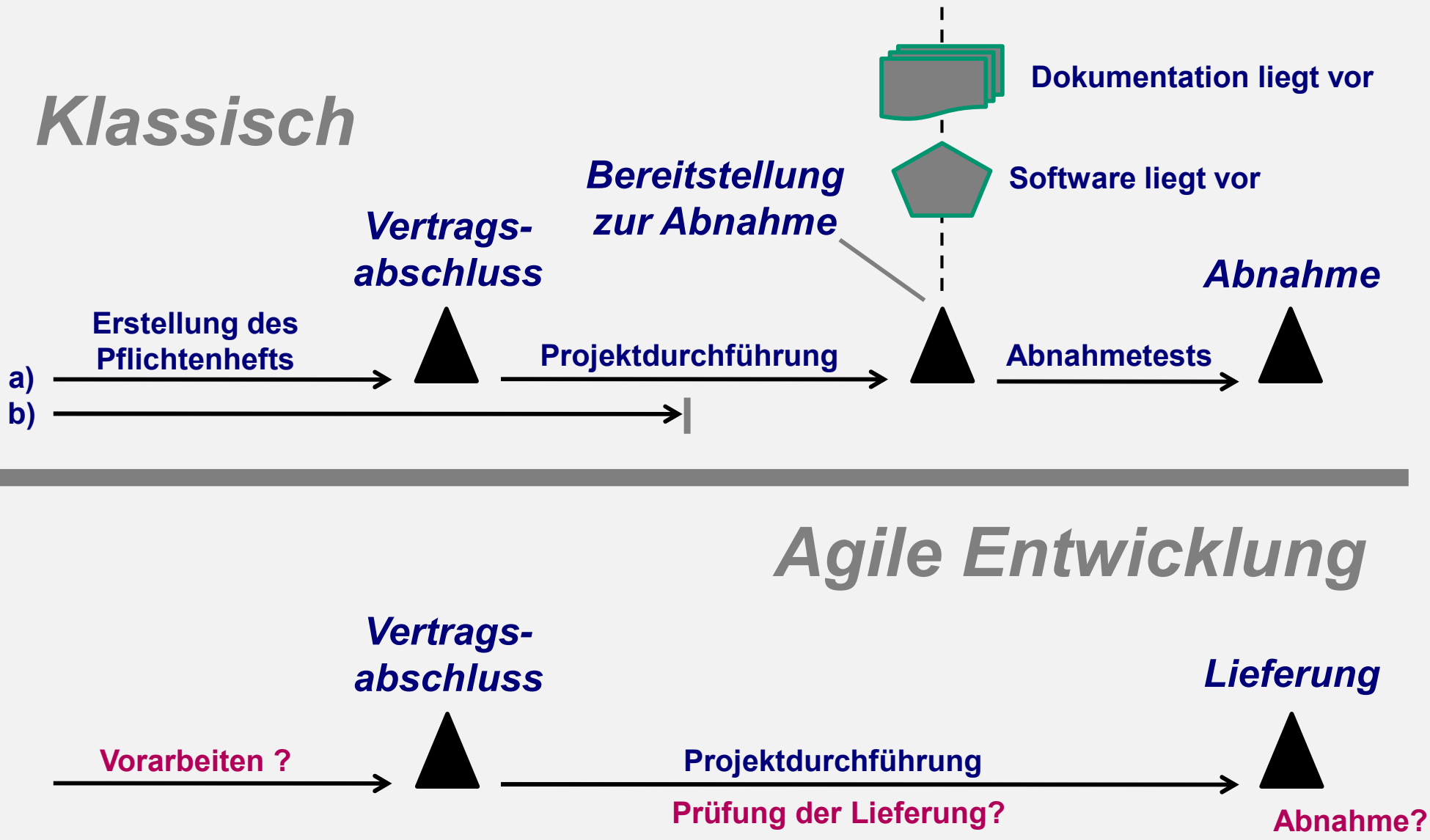
Thesen (1)

- Auch agile Projekte sind **steuerbar**
- Agile Projekte werden nur bei **kleinen Projektteams** funktionieren
- Ein agiles Projekt braucht **erfahrene, besonders teamfähige Mitarbeiter**
- Je agiler das Projekt, desto mehr **Verantwortung** „wandert“ zum Auftraggeber
- Auch agile Projekte brauchen ein **Change Request Verfahren**
- Agile Projekte brauchen eine hohe **Transparenzbereitschaft**
- Auch agile Projekte können eine sehr **hohe Qualität** erreichen

Thesen (2)

- Der **Projektverlauf** ist häufig spärlich bis gar nicht dokumentiert
  → schwierige Beweislage, wenn Streit aufkommt
- Die „klassische“ **Dokumentation** wird in den meisten Fällen kein Projektergebnis sein
  → Der Kunde wird die benötigte Dokumentation selbst erstellen müssen
- Das **Pflichtenheft / die Leistungsbeschreibung** existiert häufig nur in rudimentärer Form
  → Kann überhaupt noch eine Abnahme der Leistungen erfolgen?
- Falls eine Abnahme überhaupt vorgesehen ist, wird sehr stark der „**mittlere Ausführungsstandard**“ heranzuziehen sein
  → großes **Risiko** sowohl für Auftraggeber als auch für den Auftragnehmer
  → Weitere **juristische Implikationen** zunächst unklar

Möglicher Projektverlauf bei agilem Vorgehen



Klassische Entwicklungsprojekte (1)

- Vertragstyp: Praktisch immer Werkvertrag
- Zu erstellendes Werk im Pflichtenheft / Feinkonzept beschrieben
- Klare Trennung der Verantwortlichkeiten:
  - Auftragnehmer verantwortlich für den Erfolg
  - Auftraggeber wirkt „nur“ mit
- Vergütung nur nach Erklärung der Abnahme durch Auftraggeber
- Abnahme nur, wenn Werk im Wesentlichen mangelfrei
- Nach Abnahme: 24 Monate Gewährleistung, d.h. kostenlose Beseitigung von Mängeln
- Oft weniger klar: Rechtseinräumung an Arbeitsergebnissen

Klassische Entwicklungsprojekte (2)

- Dynamik im Projekt muss rechtlich formalistisch bewältigt werden, damit das Projekt nicht rechtlich „entgleist“
- Abweichung von ursprünglicher Leistungsbeschreibung (Pflichtenheft, Feinkonzept):

  Change Request-Verfahren

  - „Antrag“, Anpassung Vergütung, Zeitrahmen
  - Schriftliche Vertragsänderung
  - Sonst: Geschuldetes Werk nicht mehr identifizierbar

- Trennung der Verantwortung muss strikt aufrechterhalten werden
  - AG darf keine aktive Rolle übernehmen, weder bei der Projektleitung noch bei der Ausführung
  - Sonst: Werkvertrag kann sich in einen Dienstvertrag wandeln!

Agile Entwicklungsprojekte (1)

- Vertragstyp: Dienstvertrag oder ArGe (BGB-Gesellschaft) ?
- Arbeitsergebnis wird im Projektfortschritt gemeinschaftlich definiert
- Keine klare Trennung der Verantwortlichkeiten:
  - Auftragnehmer nicht alleine verantwortlich für den Erfolg
  - Auftraggeber übernimmt aktive Rolle, wirkt nicht lediglich mit
- Vergütung des Auftragnehmers idR nach Aufwand
- Grundsätzliche keine Abnahme, allenfalls iterative „Freigaben“
  - Rechtliche Bedeutung „Freigabe“ nicht definiert
- Keine Gewährleistung nach Gesetz
- Rechte an Arbeitsergebnissen: Miturheberschaft ?

Agile Entwicklungsprojekte (2)

- Unklare Zuordnung zu Vertragstyp macht Projekt rechtlich unkalkulierbar
- Risiko für Auftraggeber:
  - Zahlung nach Aufwand
  - Arbeitsergebnisse nicht brauchbar
  - Keine Gewährleistung
  - Bei Miturheberschaft beider:
    Darf SW nicht alleine ändern / weiterentwickeln
- Risiko für Auftragnehmer:
  - Miturheberschaft des Auftraggebers – darf SW nicht alleine ändern, weiterentwickeln, vertreiben

Zusammenfassung zu den Vorgehensmodellen

1. Vorgehensmodelle helfen, ein Projekt **strukturiert** durchzuführen, **Risiken zu senken** und die **Ergebnisqualität anzuheben**.

2. In aller Regel ist die Anpassung eines Vorgehensmodells auf die **juristischen Rahmenbedingungen** erforderlich. **Umgekehrt** bestimmt auch ein konkret ausgewähltes Vorgehensmodell einen Teil der juristischen Rahmenbedingungen.

3. Agile Projektmethoden sind stark auf dem Vormarsch, erfordern aber **einige Ergänzungen**, um mit den klassischen Vertragstypen (z.B. Werkvertrag) kompatibel sein zu können.

4. Die **Auswahl einer passenden Projektmethode** erfordert eine gründliche Analyse aller Projektumstände zu einem möglichst frühen Zeitpunkt.