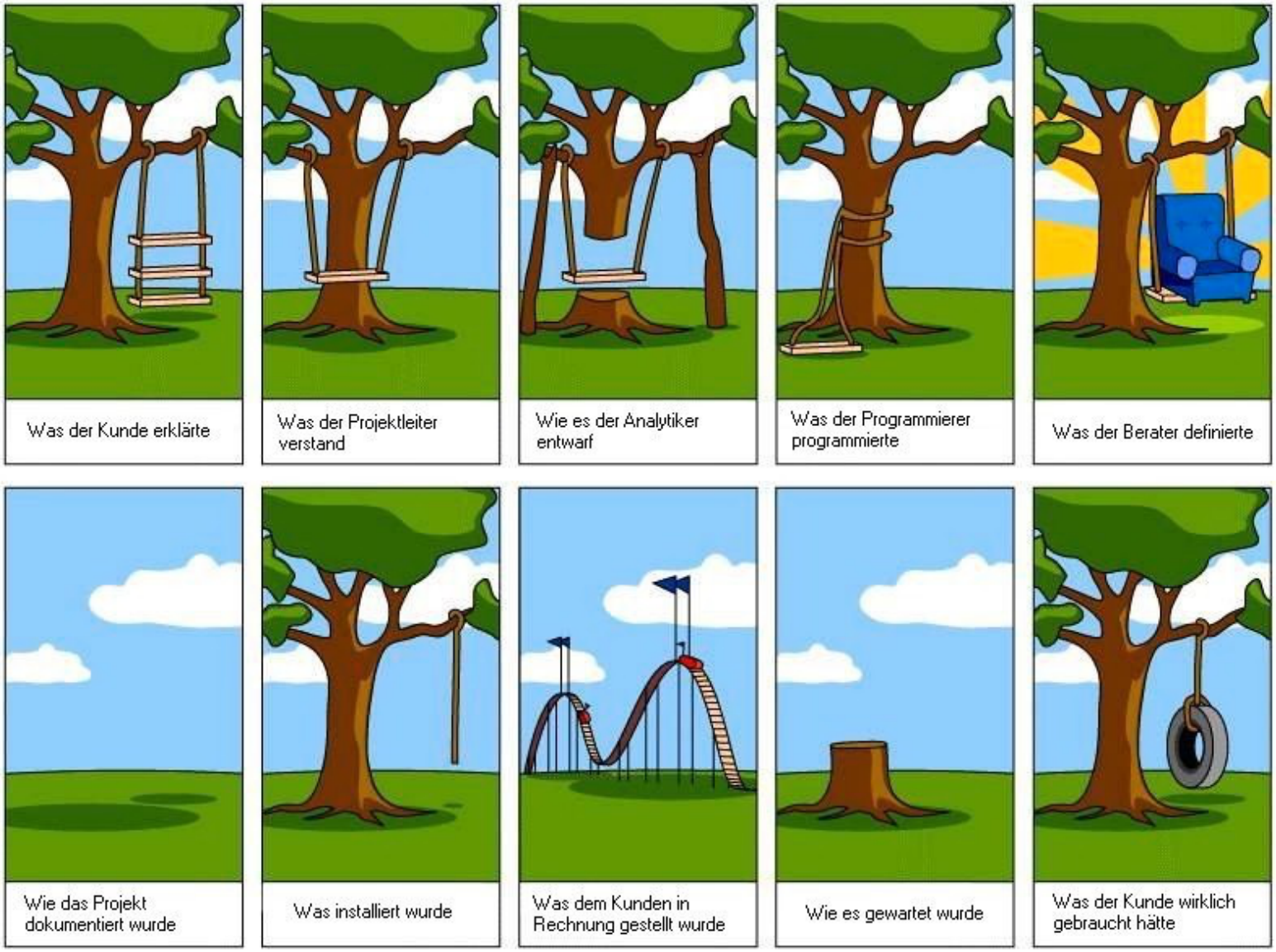
Softwaretechnik 1(A)

LE 02: Klassische Vorgehensmodelle

Fachhochschule
Dortmund
University of Applied Sciences and Arts

Vorgehensmodelle - Motivation



[Autor unbekannt]

Vorgehensmodelle - Ziele

- 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**

Vorgehensmodelle - Definition

- Ein Vorgehensmodell ist die Beschreibung des Arbeitsablaufs (Entwicklungsstufen, Phasenkonzepte)

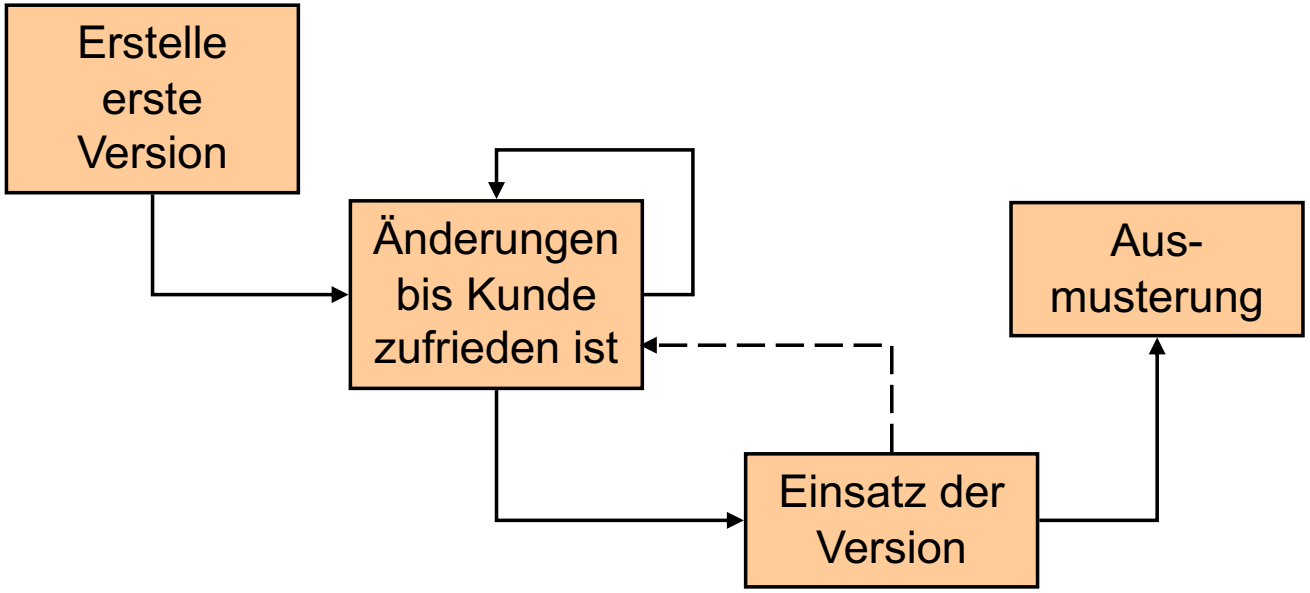
- es definiert
  - Rahmenwerk und feste Meilensteine
  - Reihenfolge und Teilprodukte der Aktivitäten
  - Kriterien für die Abnahme der Produkte (Fertigstellungskriterien)
  - Verantwortlichkeiten und Kompetenzen
  - notwendige Mitarbeiterqualifikationen
  - anzuwendende Standards, Richtlinien und Werkzeuge

⇒ **Besseres Projektmanagement möglich!**

Typische Vertreter klassischer Vorgehensmodelle

- (Code & Fix ⇔ Hacking)
- Wasserfall-Modell
- V-Modell/ V-Modell XT
- Prototypen-Modell
- iteratives Modell
- inkrementelles Modell
- Spiral-Modell
- Rational Unified Process (RUP)

Code & Fix - Modell



Entwicklung

Verbesserung

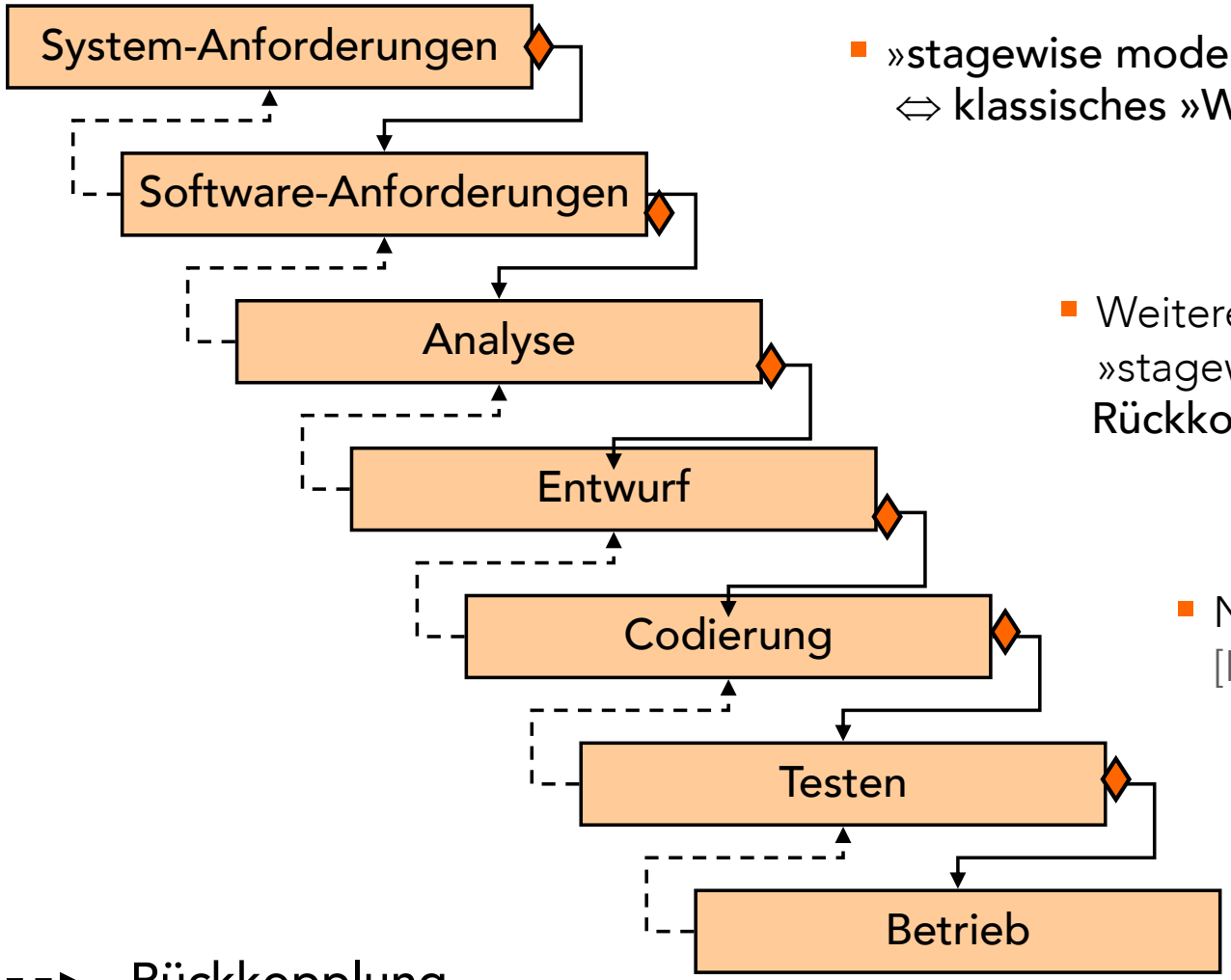
Code & Fix - Bewertung

+ geringer (kein) Managementaufwand

- schlechte Zuverlässigkeit, Wartbarkeit und Übersichtlichkeit des Codes
- starke individuelle Abhängigkeit vom Programmierer
- Funktionsumfang wird zwischen Entwickler und Anwender ausgehandelt
- keine Entwicklung von Dokumentation und Testfällen

⇒ **nur für kleine Projekte geeignet!**

Wasserfall-Modell



- »stagewise model« [Benington 1956]
  ⇔ klassisches »Wasserfall«-Modell
- Weiterentwicklung des »stagewise models« um **Rückkopplung** [Royce 1970]
- Name: **»Wasserfall«-Modell** [Boehm 1981]

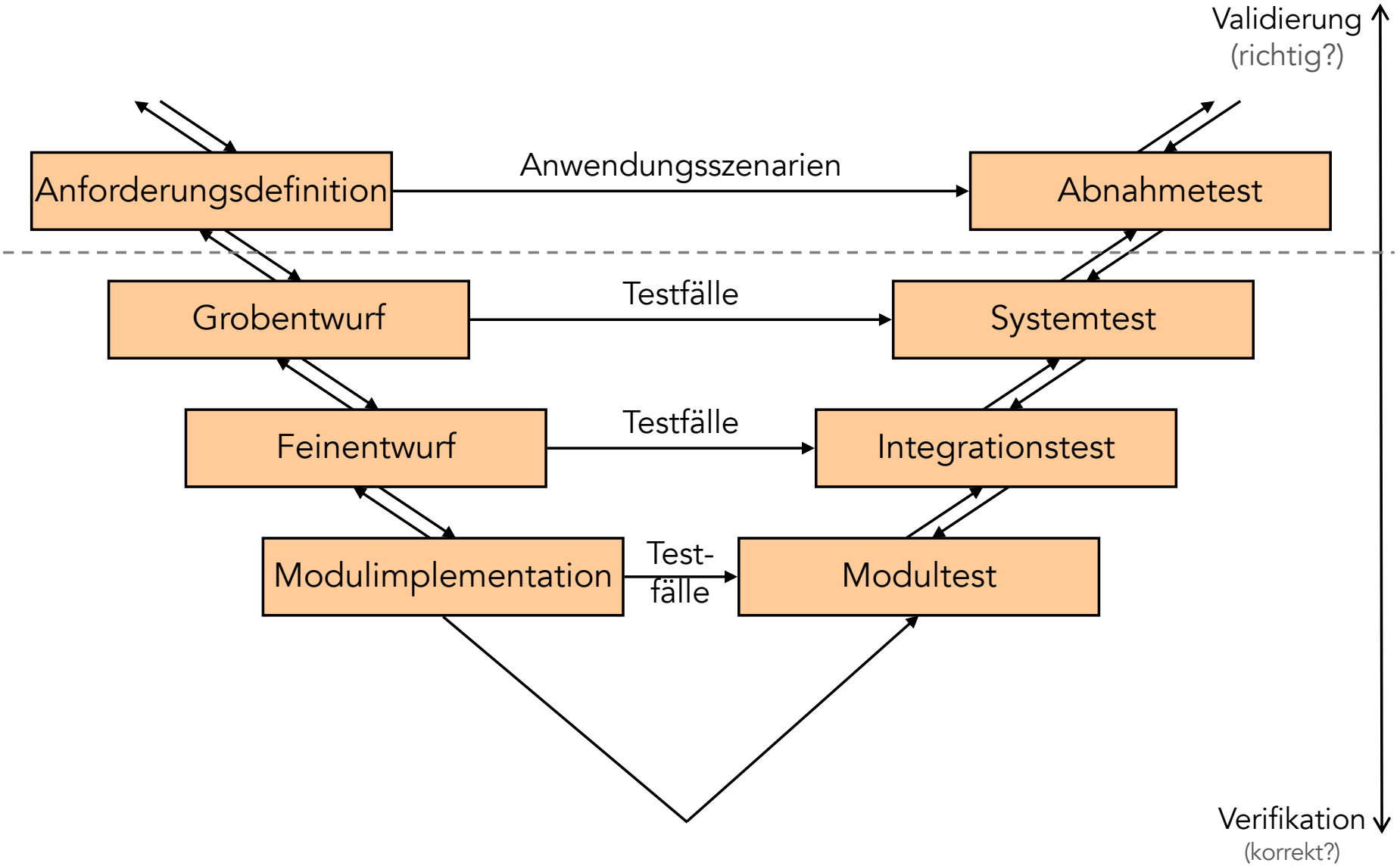
Wasserfall - Bewertung

+ extrem einfaches Modell
+ geringer Management-Aufwand
+ disziplinierter, kontrollierbarer und sichtbarer Prozessablauf

– strenge Sequentialität oft nicht sinnvoll bzw. 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

⇒ **war sehr beliebt und weit verbreitet**

V- Modell


Validierung
(richtig?)
Anforderungsdefinition
Anwendungsszenarien
Abnahmetest
Grobentwurf
Testfälle
Systemtest
Feinentwurf
Testfälle
Integrationstest
Modulimplementation
Test-
fälle
Modultest
Verifikation
(korrekt?)

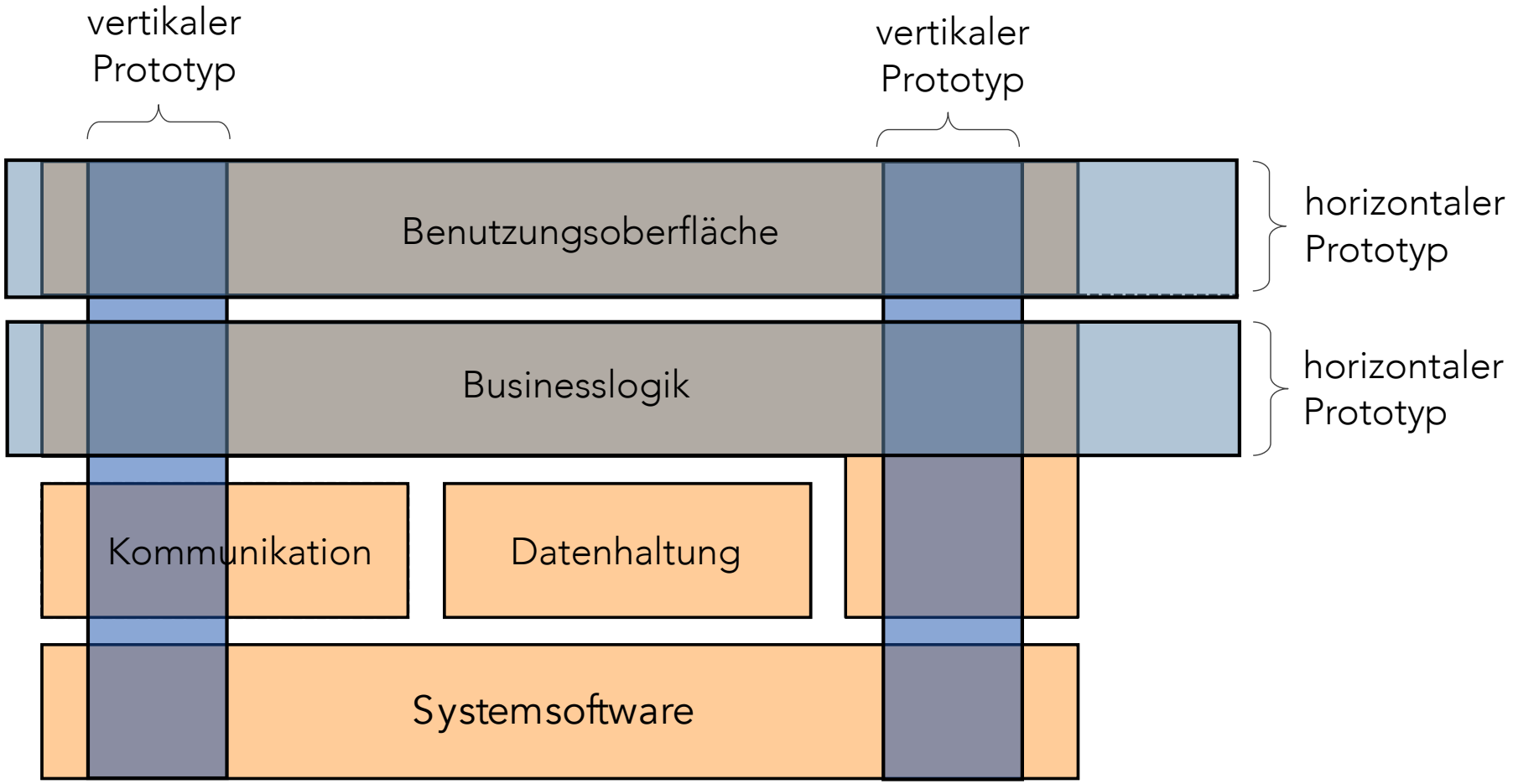
V-Modell - Bewertung

+ zerlegt Vorgehen in einen konstruktiven und prüfenden Anteil
+ unterscheidet zwischen Verifikation und Validation
+ Korrektheit und Richtigkeit haben hohen Stellenwert
+ anschauliche Darstellung durch V-Zuordnung
+ Betonung der Relevanz von Tests

– geht von strikter modularer Zerlegung aus
– Validierung erst am Ende möglich

⇒ **leicht zu erklären und zu erlernen** mit starkem Fokus auf **Qualität** (im Sinne von Fehlerfreiheit und Richtigkeit)

Prototypen - Modell



Varianten des Prototypen – Modells

- Demonstrationsprototyp
  - dient zur **Auftragsakquisition**
  - wird i.d.R. schnell aufgebaut und später »weggeworfen«.
- Prototyp i.e.S.
  - wird parallel zur Modellierung des Anwendungsbereichs erstellt zur **Analyse des Anwendungsbereichs**
  - soll **Aspekte** z.B. der Benutzungsschnittstelle oder Teile der Funktionalität **veranschaulichen**
- Labormuster
  - soll konstruktionsbezogene **Fragen und Alternativen beantworten**
  - sollte technisch mit dem späteren Produkt vergleichbar sein
- Pilotsystem
  - ist **Kern eines Produkts**
  - Unterscheidung zwischen dem Prototyp und dem Produkt verschwindet
  - Pilotsystem ist für die Benutzung in der Einsatzumgebung entworfen und nicht nur unter Laborbedingungen

Prototypen – Modell - Bewertung

+ sinnvolle Integration in andere Prozess-Modelle möglich
+ hilft Alternativen zu Evaluieren
+ schafft mehr Klarheit bei und eine starke Rückkopplung mit dem Endbenutzer und dem Auftraggeber
+ Labormuster fördern die Kreativität für Lösungsalternativen

– hoher Entwicklungsaufwand durch zusätzliche Herstellung von Prototypen
– Gefahr der Umwandlung eines „Wegwerf-Prototyps“ zu einem Teil des Endprodukts aus Termingründen
– Prototypen ersetzen fehlende Dokumentation
– Beschränkungen und Grenzen von Prototypen sind oft unbekannt.

⇒ **reduziert Entwicklungsrisiko**

Iteratives Modell/ Evolutionäres Modell

- **Vermeidung** der Implementierung eines Produkts in **voller Detaillierung**
- Kernanforderungen des Auftraggebers führen direkt zur „**Nullversion**“ (wird ausgeliefert)
- nur dieser Produktkern wird entworfen und implementiert
- der Auftraggeber sammelt Erfahrungen
  - **neue Anforderungen ⇒ neue Version**
- das Software-Produkt wird **allmählich und stufenweise entwickelt**
- durch den Auftraggeber gesteuert
- **Pflegeaktivitäten** werden ebenfalls als Erstellung einer **neuen Version** betrachtet

Iteratives Modell/ Evolutionäres Modell - Bewertung

+ Integration der Erfahrungen der Anwender in die Entwicklung
  → schrittweise Verfeinerung
+ korrigierbare Entwicklungsrichtung
+ überschaubare Projektgröße
+ keine Ausrichtung auf einen einzigen Endtermin

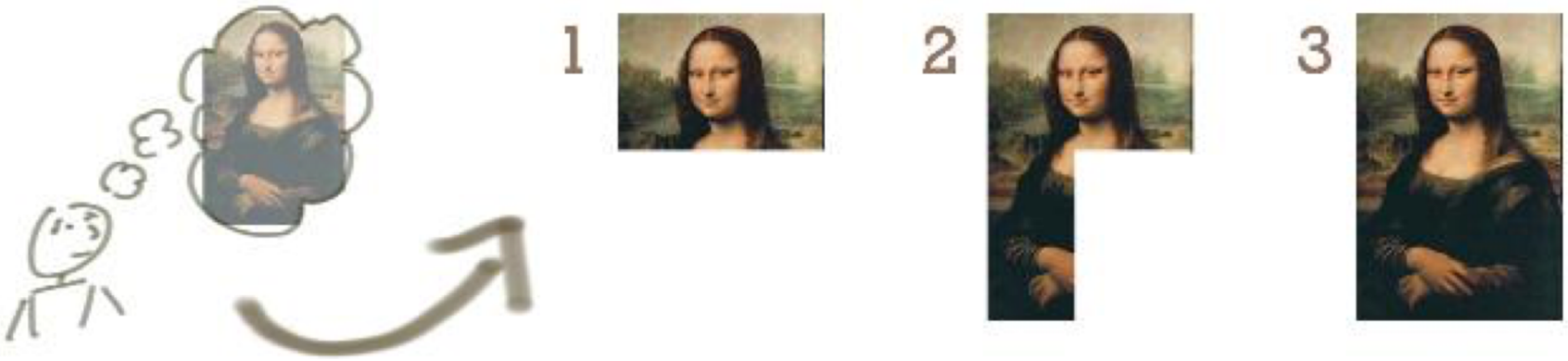
- eventuell mangelnde Flexibilität der Nullversion zur Anpassung an unvorhersehbare Evolutionspfade
- eventuell komplette Änderung der Architektur in späteren Versionen

⇒ **Gut geeignet, wenn der Auftraggeber die Anforderungen noch nicht vollständig überblickt!**

»I can't tell you what I want, but I'll know it when I see it«

Inkrementelles Modell

- vermeidet die Nachteile des evolutionären Modells
- **Anforderungen** an das zu entwickelnde Produkt werden **möglichst vollständig** erfasst und modelliert
- nur ein **Teil der Anforderungen** wird entworfen und implementiert
- anschließend wird die nächste Ausbaustufe realisiert
  unter Berücksichtigung der Erfahrungen des Auftraggebers mit der laufenden Version



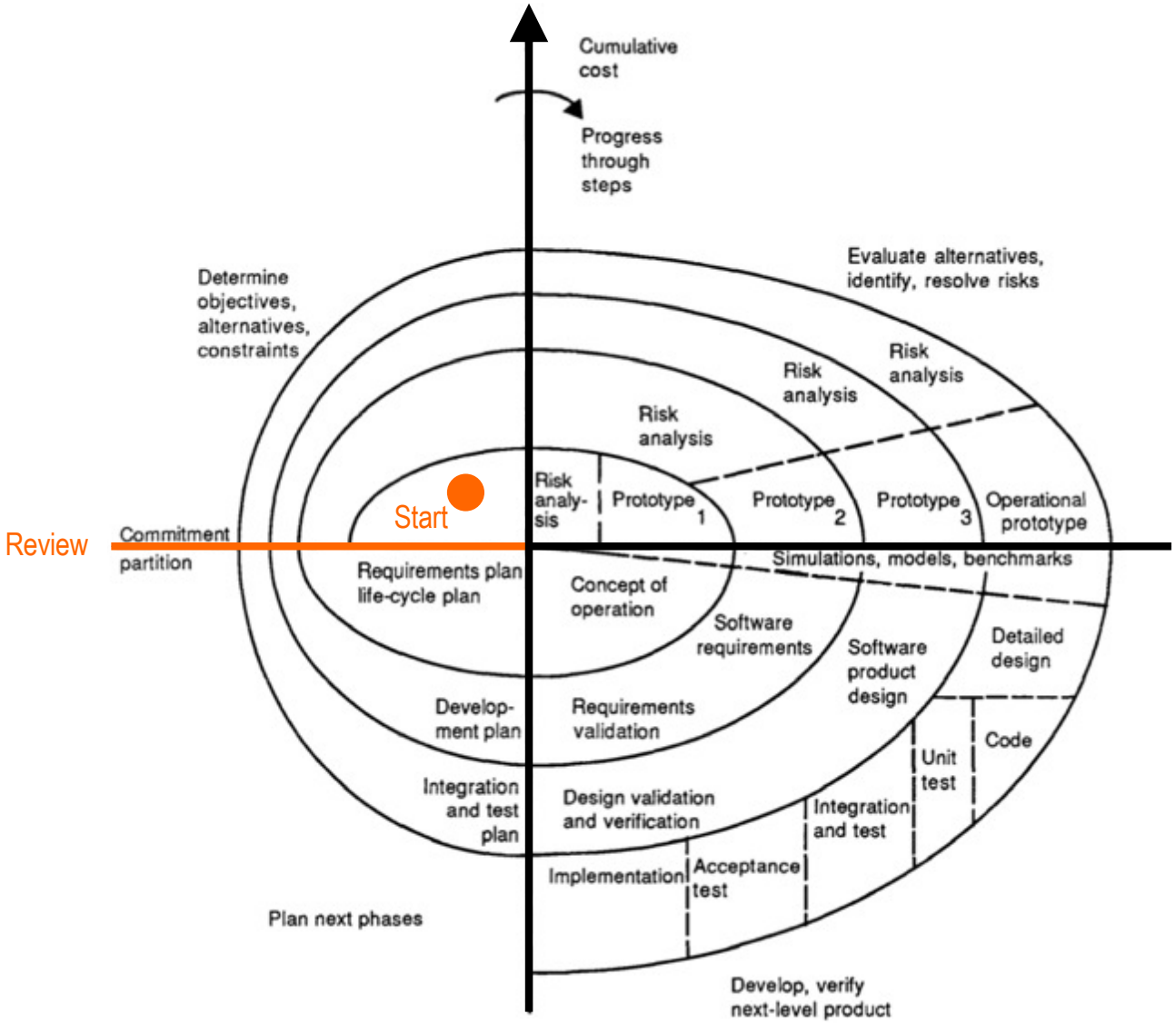
Inkrementelles Modell - Bewertung

+ grobes Konzept über alle Module vorhanden
+ inkrementelle Erweiterungen passend zum bisherigen System
+ nutzbare Module für den Auftraggeber in kurzen Abständen
+ Integration der Erfahrungen der Anwender in die Entwicklung
+ Priorisierung der Module “on the fly“

- Vollständige Spezifikation nicht immer möglich

⇒ **Gut geeignet, wenn der Auftraggeber schon mit Teillösungen arbeiten kann und Module priorisiert sind.**

Spiralmodell



Spiralmodell

- **Metamodell**
- **Risikogetriebenes** Modell
  - Richtlinien
    - Beginne im Kleinen
    - Halte die Spirale so eng wie möglich
    - Erreiche so die Entwicklungsziele mit minimalen Kosten
  - Für jede Aktivität und jeden Ressourcenverbrauch wird gefragt
    - »Wieviel ist genug?«
      dadurch wird ein »Overengineering« vermieden.
- jede Spirale stellt einen **iterativen Zyklus** durch dieselben Schritte dar
- Ergebnisse des letzten Zyklus ⇒ Ziele des nächsten Zyklus
- bei Bedarf separate Spiralzyklen für **verschiedene Komponenten**
- **keine Trennung in Entwicklung und Wartung**

Spiralmodell - Bewertung

+ **risikobetontes iteratives** Vorgehen
+ Vorgehensmodell ist **flexibel**, d.h. es wird **nicht** für die gesamte Entwicklung festgelegt
+ Integration **anderer Prozess-Modelle** als Spezialfälle möglich

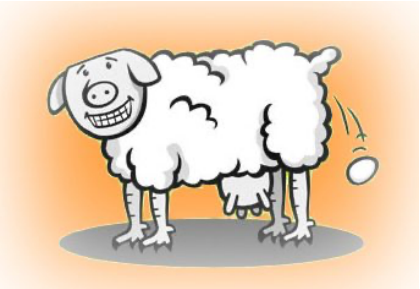
- hoher **Managementaufwand**
- **Wissen** über das Identifizieren und Managen von **Risiken** noch nicht weit genug verbreitet

⇒ **Flexibles, risikobetontes Metamodell.**

V-Modell XT

Vom V-Modell zum V-Modell XT

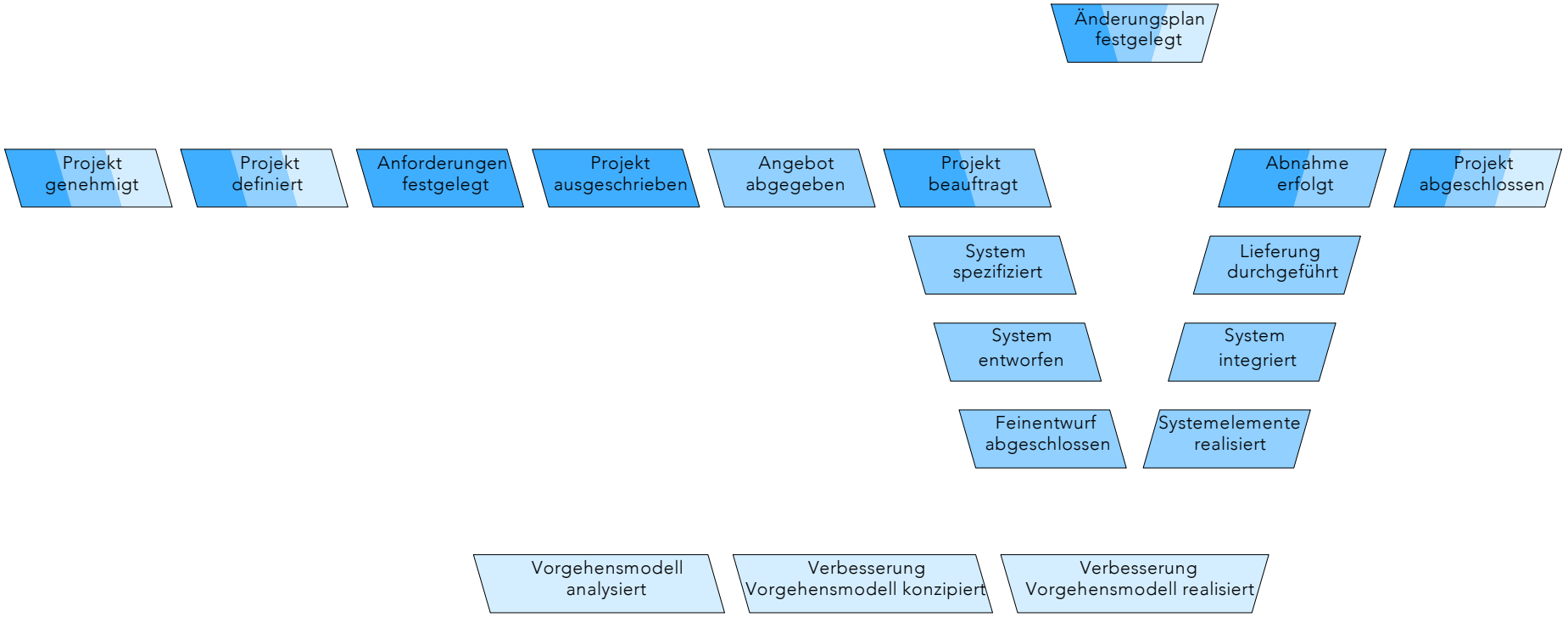
- V-Modell entstand 1992
- wurde 1997 zuletzt zum VM-97 aktualisiert und ist seitdem verbindliche Richtschnur für **IT-Projekte der Bundesverwaltung**
- Weiterentwicklung zum **V-Modell XT** kostete **4 Mio. €**, die je zur Hälfte vom Bund und von den beteiligten Firmen bezahlt wurde
  - Veröffentlichung Februar **2005**
  - 700 Seiten stark
  - **Open-Source-Werkzeuge** (VM Editor, VM-Projektassistent, ...)
  - trennt zwischen **Auftraggeber- und Auftragnehmersicht**, weil häufig viele Unternehmen in größeres Projekt involviert sind -> Ziel Reibungsverluste reduzieren
  - iteratives und inkrementelles Vorgehen
  - **Baukastenprinzip** → XT steht für „eXtreme Tailoring“ Anpassung von Set vorgefertigter Vorgehensbausteine



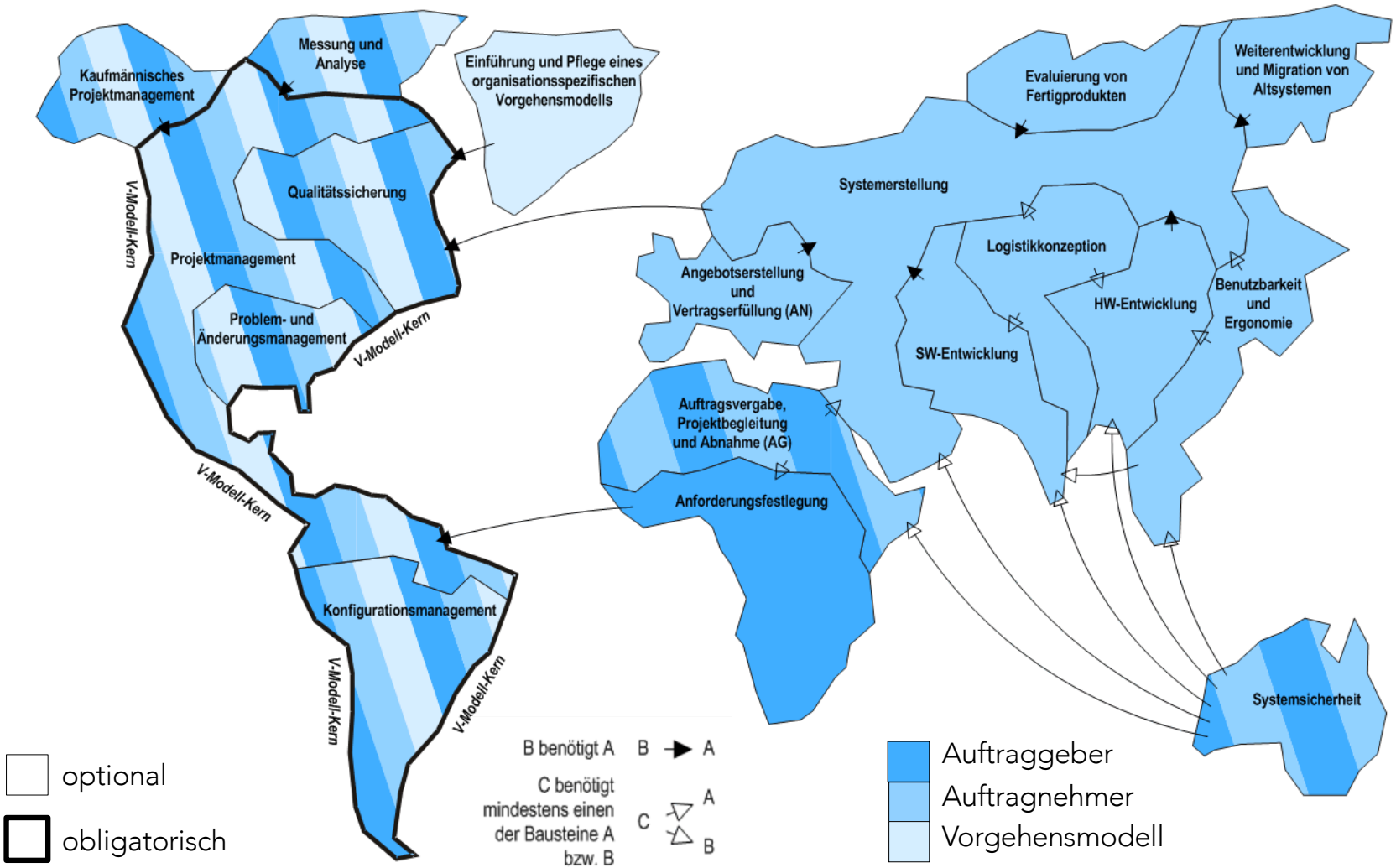
Exkurs: Zuordnung von Entscheidungspunkten zu Projekttypen

Entscheidungspunkte für die Projekttypen:

- Systementwicklungsprojekt eines **Auftraggebers**
- Systementwicklungsprojekt eines **Auftragnehmers**
- Einführung und Pflege eines **organisationsspezifischen Vorgehensmodells**

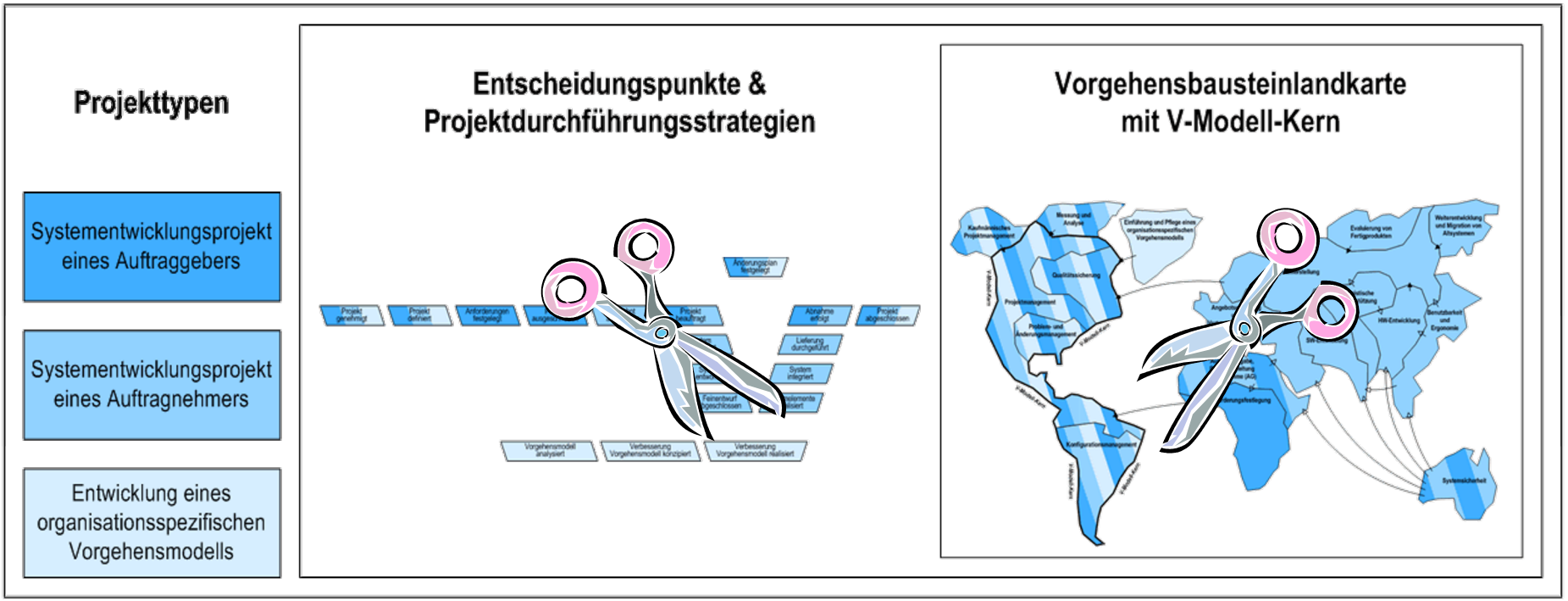


Exkurs: Vorgehensbausteine



Exkurs: V-Modell XT - Tailoring

- Auswahl des **Projekttyp**
- Auswahl der anzuwendenden **Vorgehensbausteine** (Produkte, Aktivitäten, Rollen)
- Auswahl der **Projektdurchführungsstrategien** mit ihren dazugehörigen **Entscheidungspunkten**



V-Modell XT- Bewertung

+ anpassbares Modell bzw. Template → Baukasten
+ Integration vieler Aspekte des Entwicklungsprozesses
+ projektbegleitende Tätigkeiten (Kern) erhalten gleichen Stellenwert wie eigentliche Entwicklung
+ das Modell ist öffentlich zugänglich und kann ohne Lizenzkosten benutzt werden
+ Standardisierung der Abwicklung von Systemerstellungsprojekten

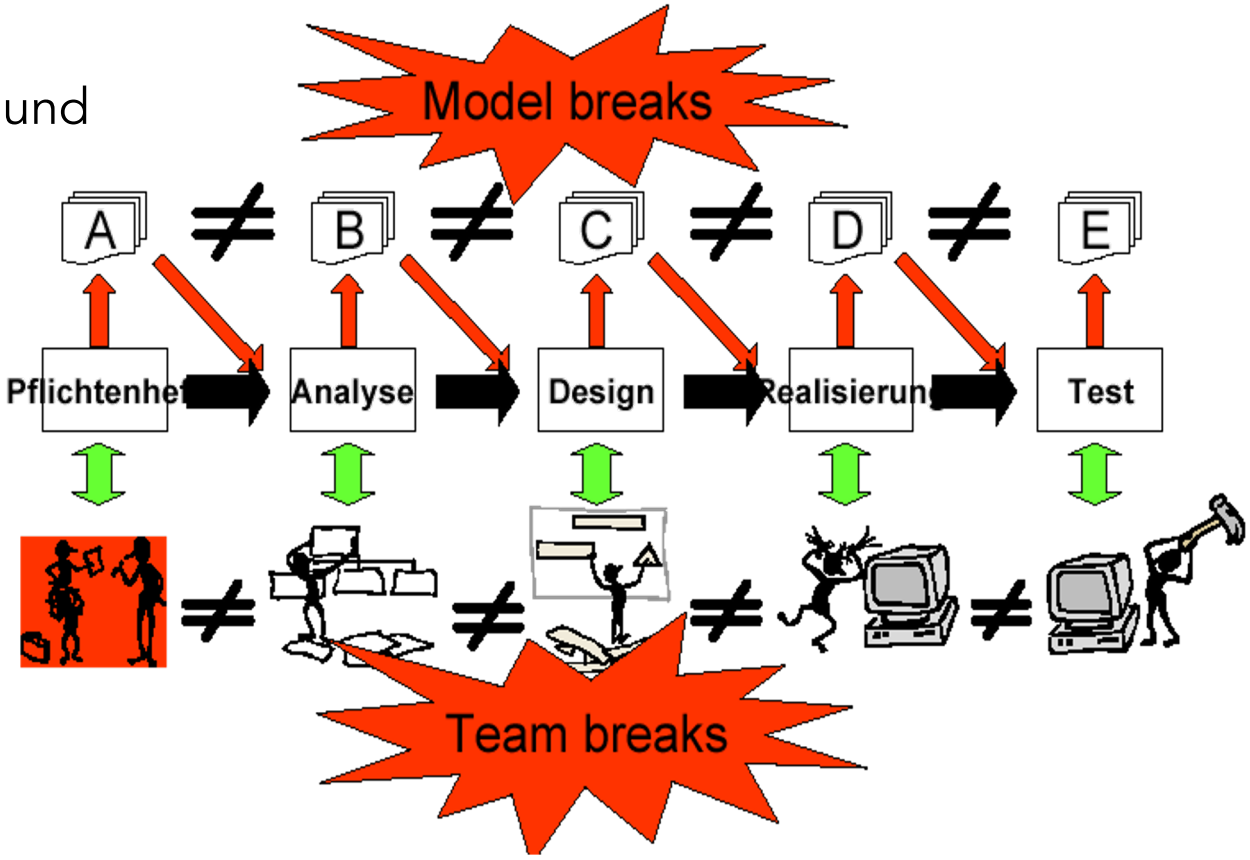
– sehr komplexes Modell
– großes und bürokratisches Modell
→keine Handhabung ohne Unterstützung durch Tools

⇒ **nur für große Projekte** (in formalem Umfeld) **geeignet**

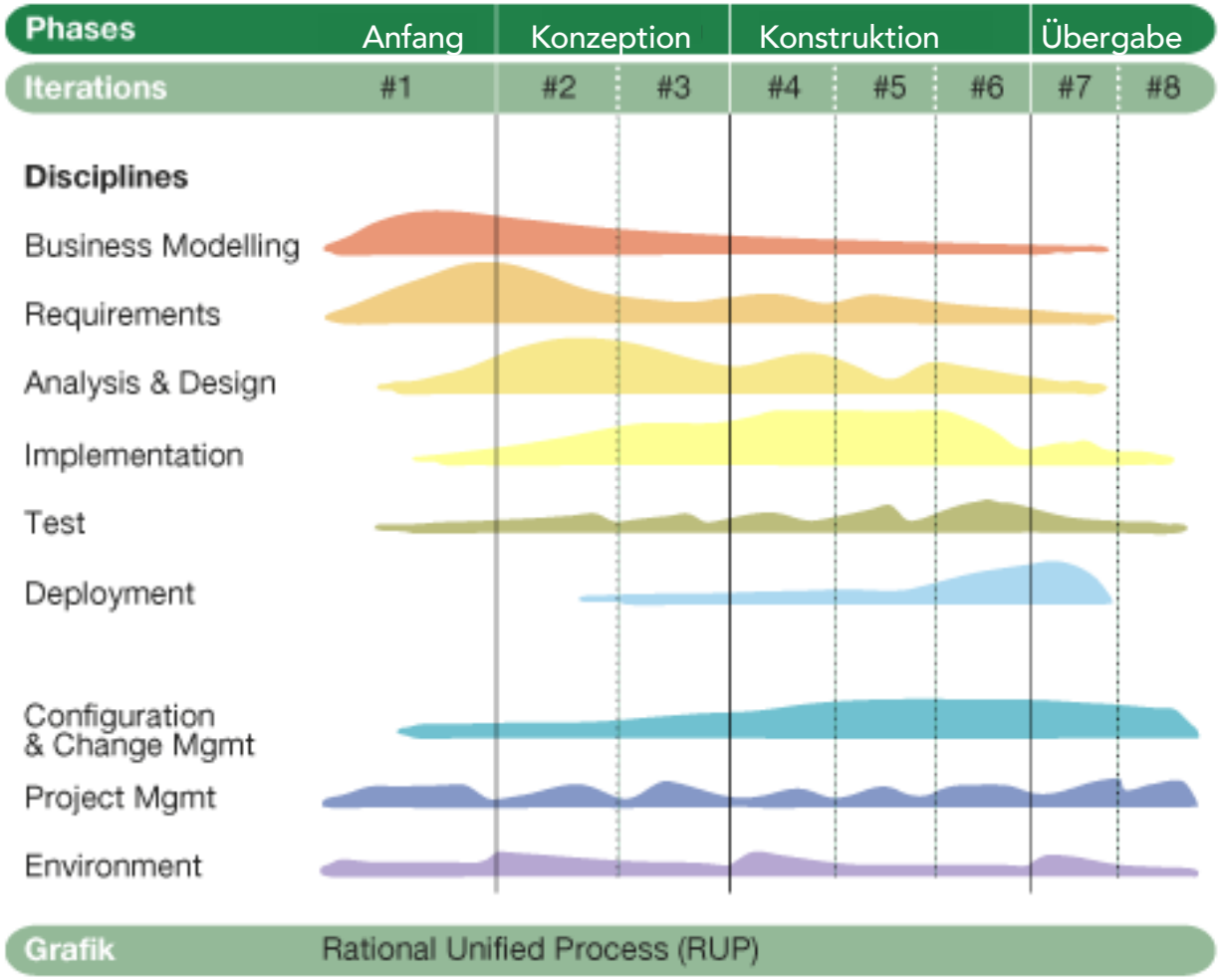
Rational Unified Process (RUP)

- adaptierbares Metamodell der Firma Rational, mit folgenden wesentlichen Merkmalen
  - architekturzentriert,
  - Use-Case-gesteuert und
  - iterativ-inkrementell
- Ziel
  - Modellbrüche
  - Teambrüche

  vermeiden



Rational Unified Process (RUP)



- in jeder Phase und Iteration werden die aufgeführten Diziplinen - wie ein eigenes kleines Projekt - komplett durchlaufen
- sie werden zu unterschiedlichen Zeiten und mit verschiedenen Geschwindigkeiten vorangebracht

Rational Unified Process (RUP) - Bewertung

+ **iterative-inkrementelles** Vorgehen ⇒ Flexibilität
+ **nutzt Standards (UML)**
+ **Werkzeugunterstützung** vorhanden
+ **umfangreiche Dokumentation** (Papier und Web)

- proprietärer Prozess mit **Lizenzgebühren**!
- detailliertes Modell ⇒ **aufwendige Anpassung**
- ist eine **Produkt** und wird kontinuierlich weiterentwickelt
  ⇒ kontinuierliches Updaten
- Einhaltung von Arbeitsabläufen führt **nicht implizit zu brauchbaren Arbeitsergebnissen**
- basiert auf **OO-Konzept**

⇒ Verbindet Vorteile phasenorientierter Entwicklung mit iterativer Entwicklung, ist aber aufwendig ⇒ eher für große Projekt

Übersicht der Ausrichtungen

Wasserfall-Modell	minimales Management
V-Modell	maximale Qualität
Prototypen-Modell	minimale Unsicherheit/Unklarheit
Iteratives Modell **Evolutionäres Modell**	minimales Kernsystem
Inkrementelles Modell	minimaler Code
Spiralmodell	Risikominimierung und Ressourcenschonung
V-Modell XT	Standardisierung und Baukastenkonzept
RUP	Verknüpfung von Phasenorientierung und Iteration

Fragen