

# Software Management

## 4. Organisation: Prozess-Modelle

Prof. Dr. K.-P. Fähnrich

09.06.2009

# Übersicht der Vorlesung

1. Grundlagen
2. Planung
3. Organisation: Gestaltung
4. Organisation: Prozess-Modelle
5. Personal
6. Leitung
7. Innovationsmanagement
8. Kontrolle: Metriken, Konfigurations- und Änderungsmanagement
9. CASE
10. Wiederverwendung
11. Sanierung

- 1. Einführung**
- 2. Das Wasserfall-Modell**
- 3. Das V-Modell**
- 4. V-Modell XT**
- 5. Das Prototypen-Modell**
- 6. Das evolutionäre/inkrementelle Modell**
- 7. Das objektorientierte Modell**
- 8. Das nebenläufige Modell**
- 9. Das Spiralmodell**

Begleitliteratur: Helmut Balzert, Lehrbuch der Software-Technik  
Quelle der Grafiken und Tabellen: Helmut Balzert, Lehrbuch der Software-Technik,  
wenn nicht anders angegeben

# Prozessmodell

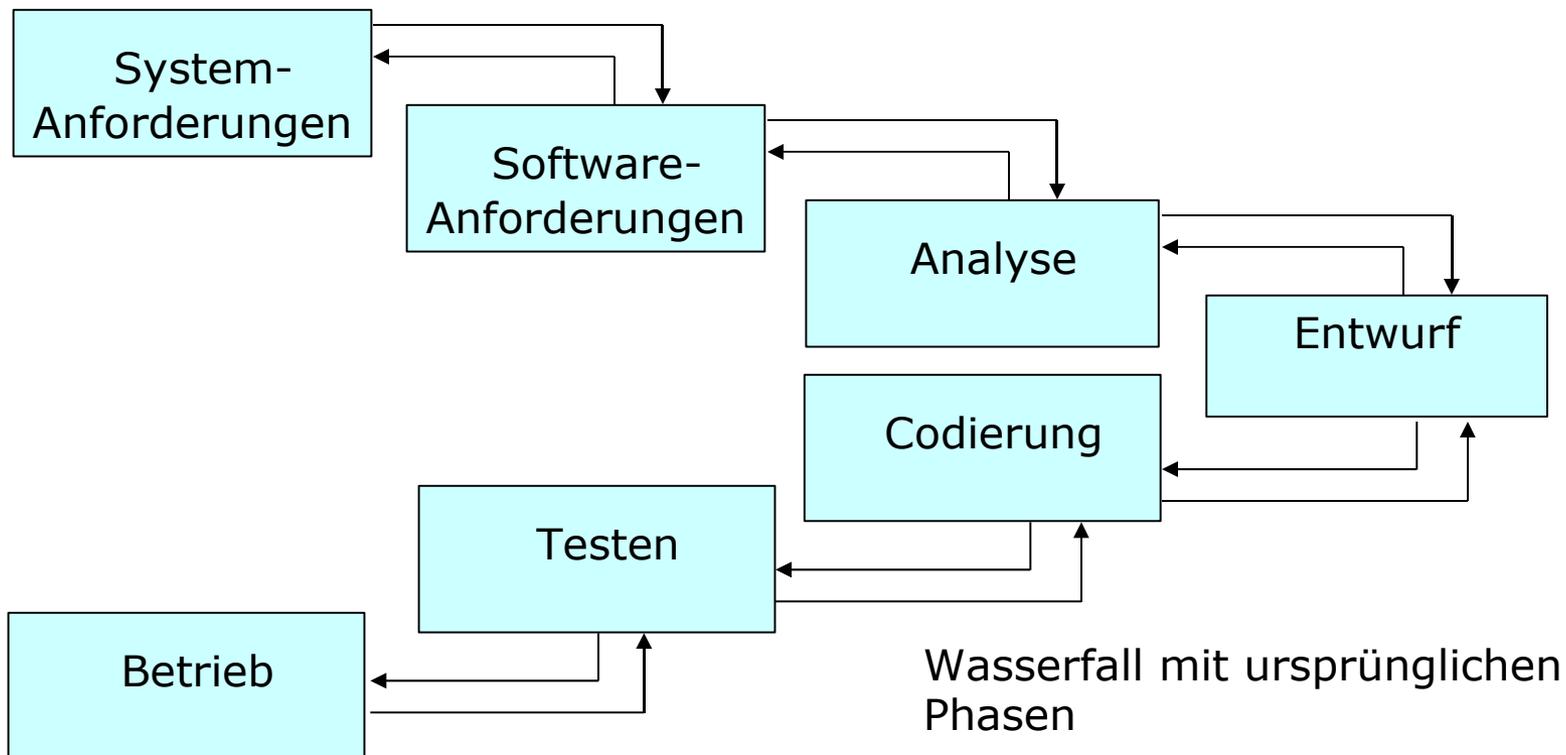
- **Prozessmodell** legt den Organisationsrahmen fest, in dem eine Software-Erstellung erfolgt.
- Grundmodell der Anfangstagen der Software-Technik: **code & fix** [Boehm 88].

## Prozess-Modell legt fest ...

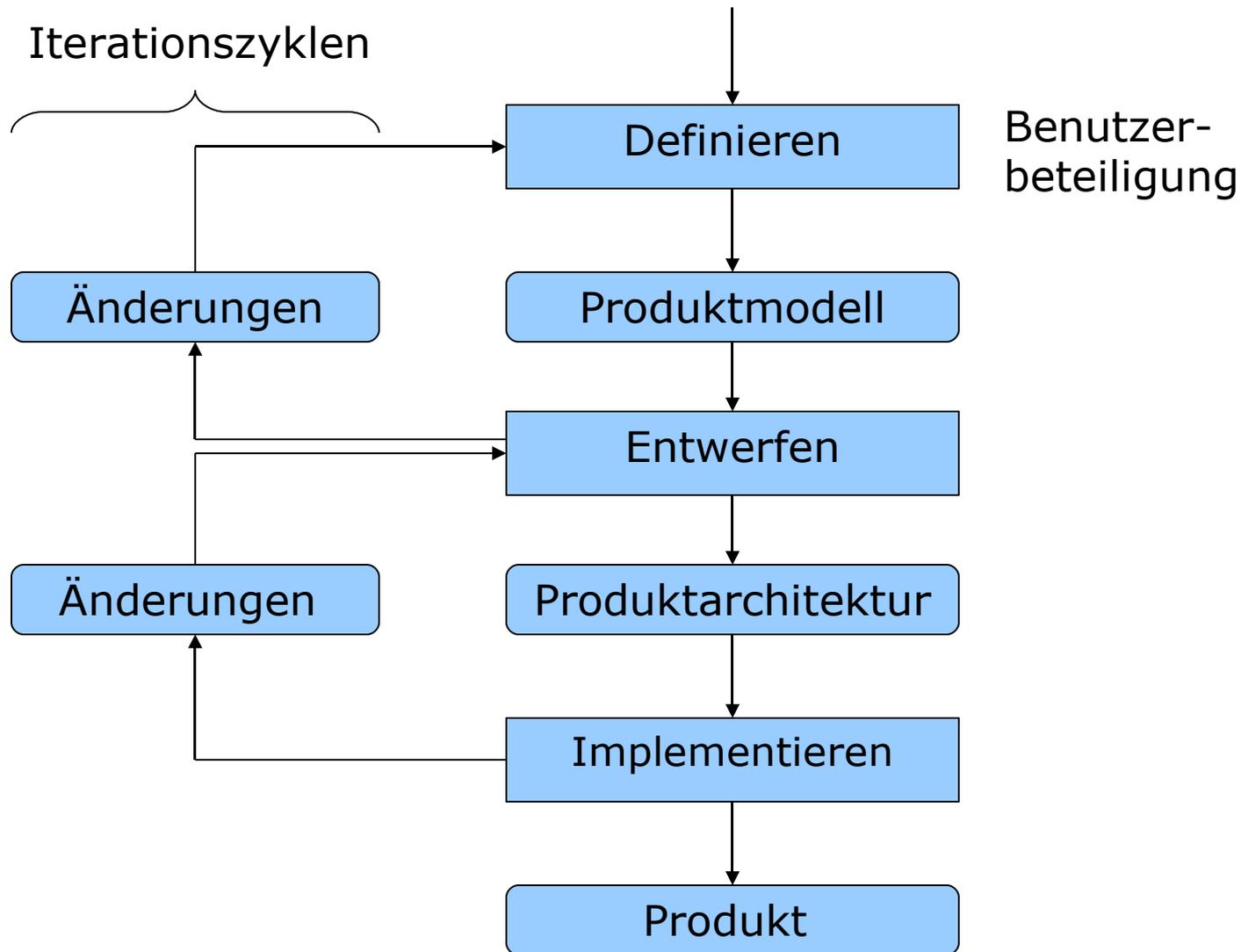
- Reihenfolge des Arbeitsablaufs.
- Jeweils durchzuführende Aktivitäten.
- Definition der Teilprodukte einschließlich Layout und Inhalt.
- Fertigstellungskriterien.
- Notwendige Mitarbeiterqualifikation.
- Verantwortlichkeiten und Kompetenzen.
- Anzuwendende Standards, Richtlinien, Methoden und Werkzeuge.

# Das Wasserfall-Modell

- **Wasserfallmodell:** Weiterentwicklung des „stagesweise models“ [Benington 56].
- Wasserfallmodell von [Royce 70] erweitert das „stagesweise model“ um Rückkopplungsschleifen zwischen den Stufen.



# Wasserfall-Modell – normierte Darstellung



# Wasserfall-Modell - Zusammenfassung

## Merkmale

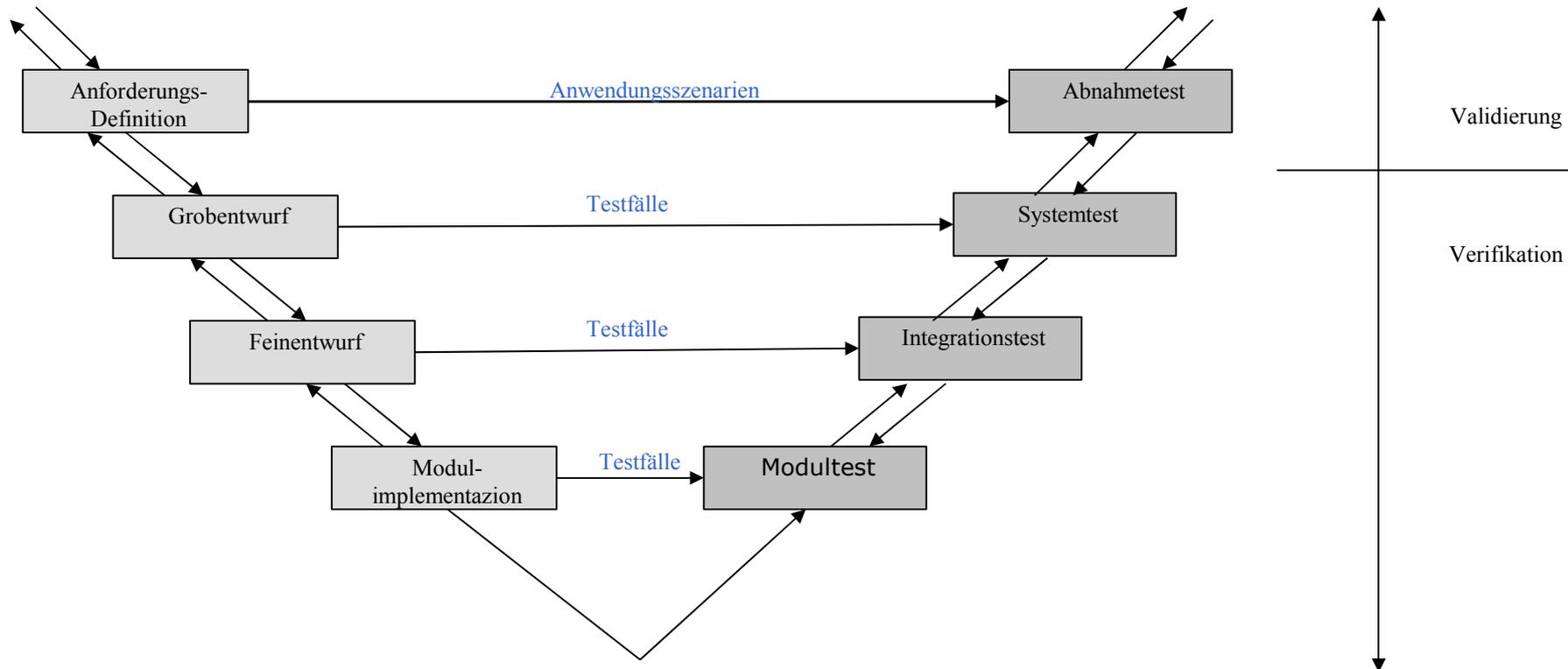
- Jede Aktivität ist in der richtigen Reihenfolge und in der vollen Breite vollständig durchzuführen.
- Am Ende jeder Aktivität steht ein fertiggestelltes Dokument.
- Der Entwicklungsablauf ist sequentiell.
- Es orientiert sich am top-down Vorgehen.
- Es ist einfach, verständlich und benötigt nur wenig Managementaufwand.
- Eine Benutzerbeteiligung ist in der Definitionsphase vorgesehen.

- Vollständige Durchführung aller Entwicklungsschritte nicht immer sinnvoll.
- Sequentielle Durchführung aller Entwicklungsschritte nicht immer sinnvoll.
- Gefahr einer falschen Prioritätsverteilung (Dokumente-System).
- Keine Berücksichtigung der Risikofaktoren.

## Nachteile

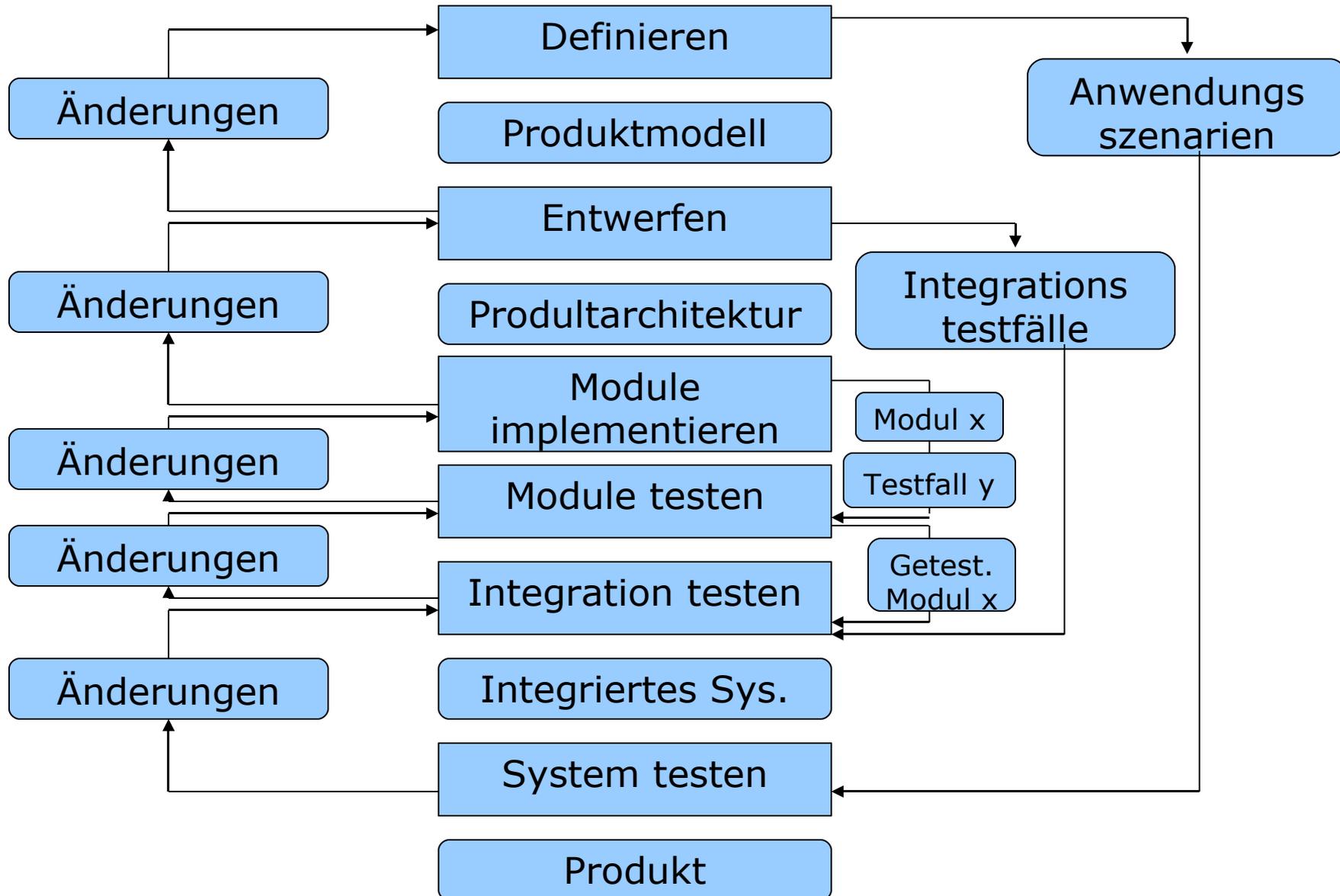
# V-Modell nach Boehm

- Erweiterung des Wasserfall-Modells durch Integration einer expliziten Qualitätssicherung
- Verifikation und Validation der Teilprodukte sind Bestandteile des V-Modells
  - **Verifikation** - Überprüfung der Übereinstimmung zwischen einem Software-Produkt und seiner Spezifikation - „Wird ein korrektes Produkt entwickelt?“
  - **Validation** - Eignung bzw. der Wert eines Produktes bezogen auf seinen Einsatzzweck - „Wird das richtige Produkt entwickelt?“



Quelle: Balzert, H.; Lehrbuch der Software-Technik

## V-Modell – normierte Darstellung

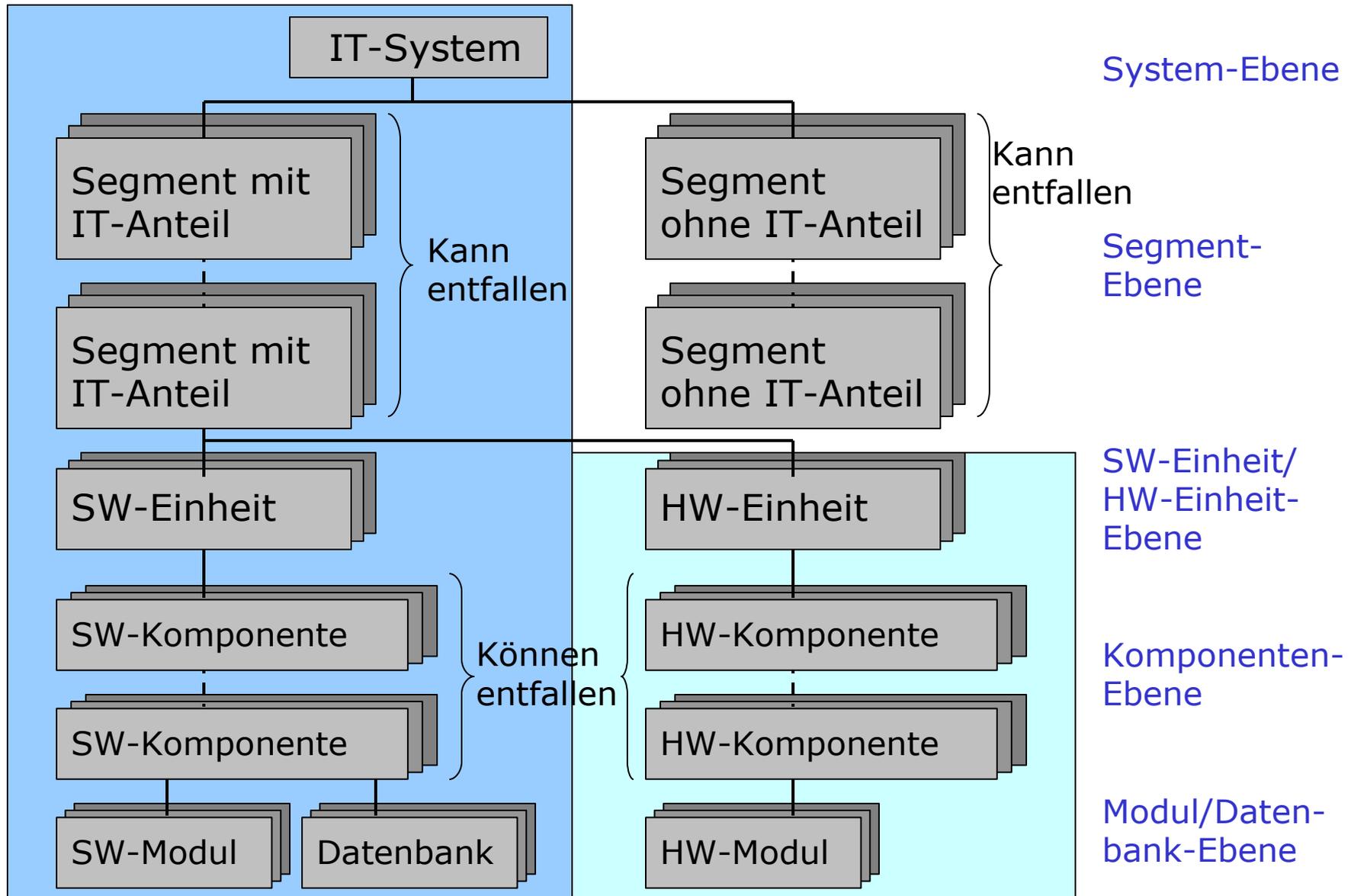


## V-Modell - Submodelle

- Legt die Aktivitäten und Produkte des Entwicklungs- und Pflegeprozesses fest.
- Produktzustände und logische Abhängigkeiten zwischen Aktivitäten und Produkten werden dargestellt.
- Vier Submodelle:
  - Systemerstellung (SE),
  - Qualitätssicherung (QS),
  - Konfigurationsmanagement (KM),
  - Projektmanagement (PM).

→ Erzeugnisstruktur des V-Modells

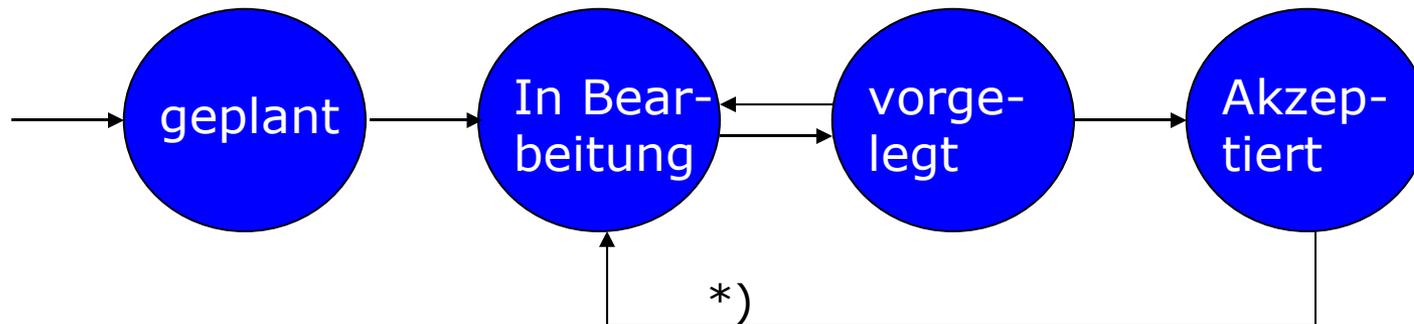
## V-Modell – IT System



## V-Modell – Aktivitäten und Produkte

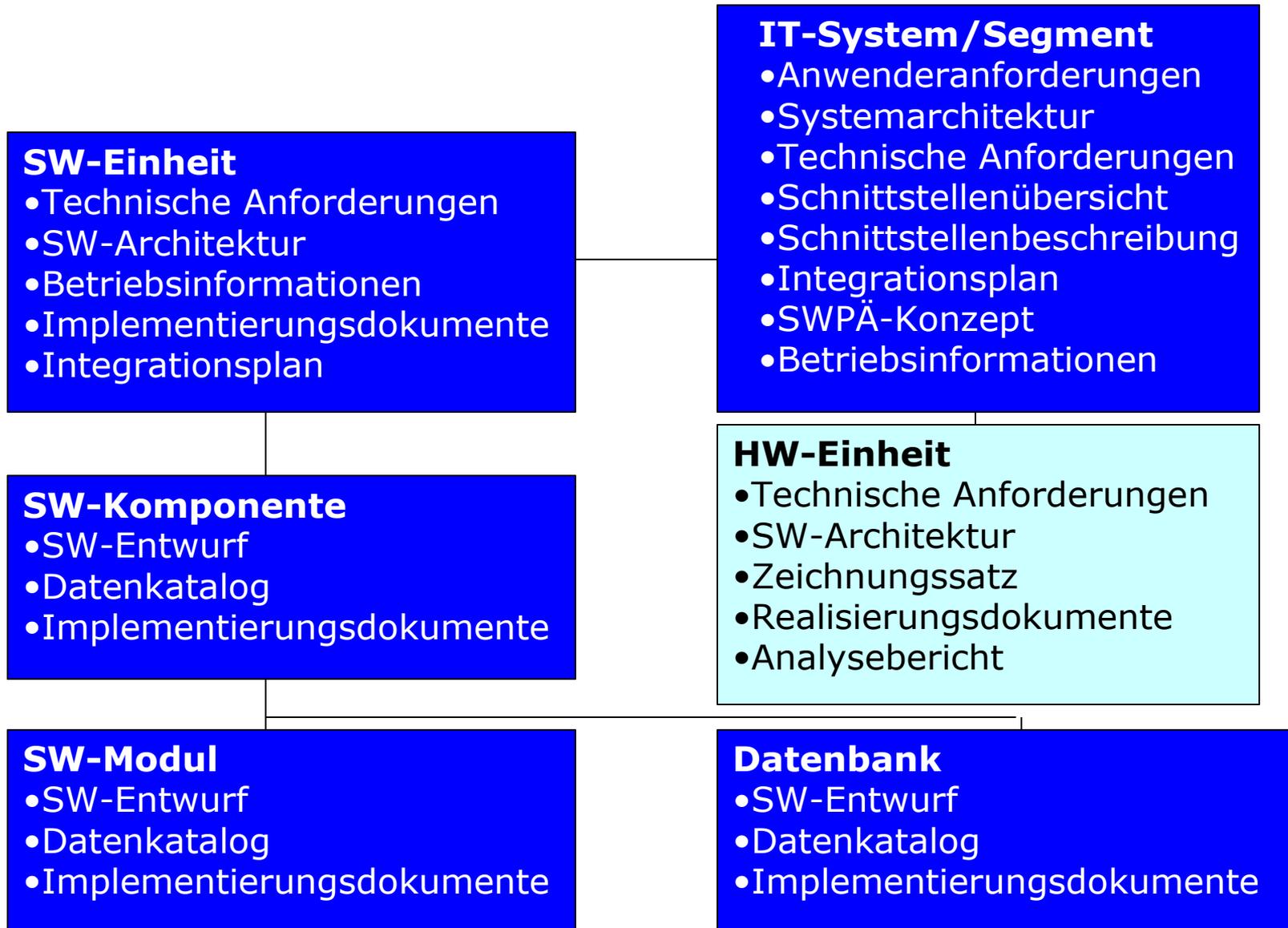
- Grundelemente des V-Modells sind Aktivitäten und Produkte.
- **Aktivität:** Tätigkeit, die bezogen auf ihr Ergebnis und ihre Durchführung genau beschrieben werden kann.
- **Produkt:** Ergebnis bzw. Bearbeitungsgegenstand einer Aktivität.
- Ziel einer Aktivität:
  - Erstellung eines Produkts,
  - Änderung des Zustands eines Produkts,
  - Änderung des Inhalts eines Produkts.

## Zulässige Zustandsübergänge von Produkten

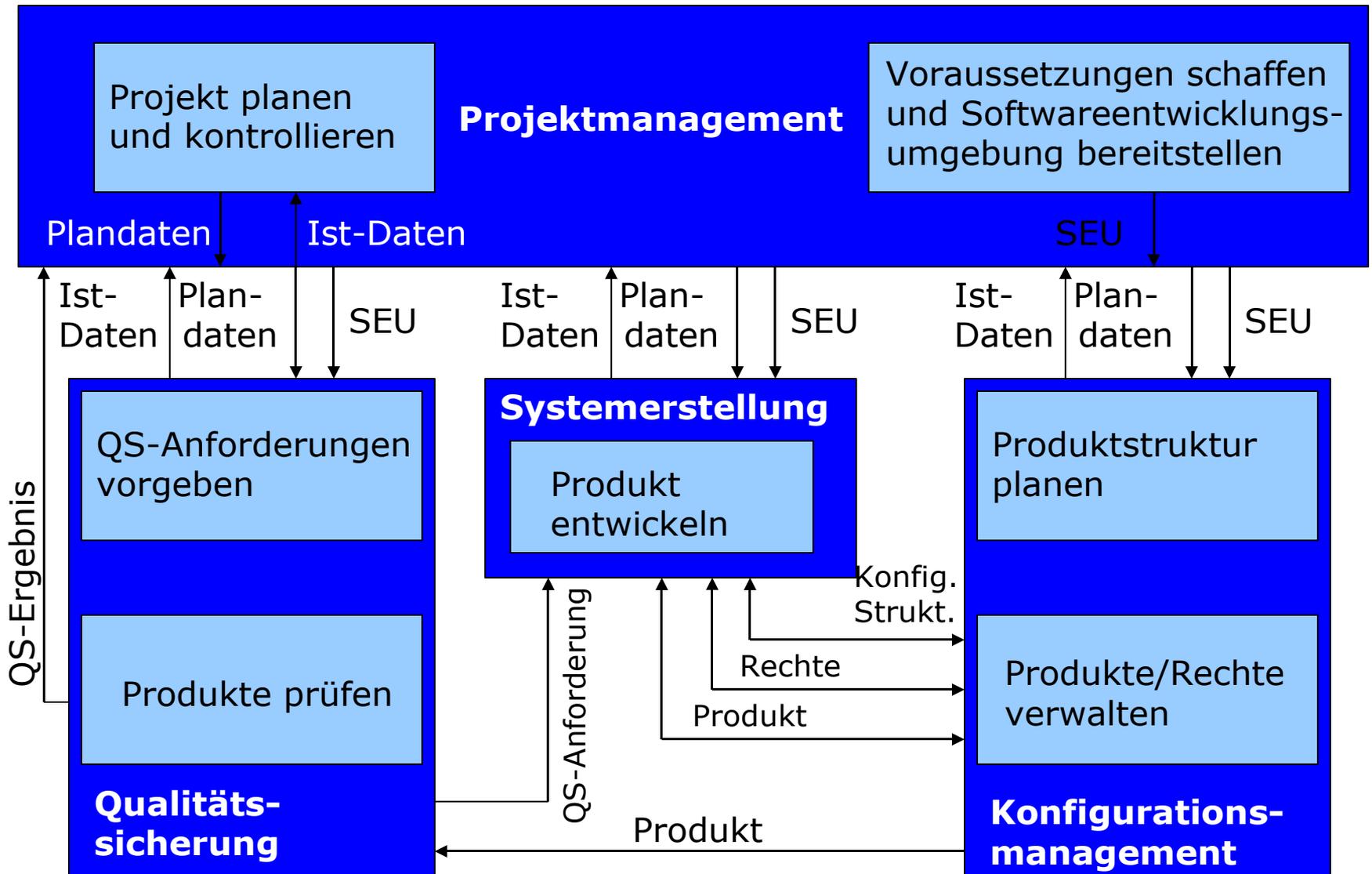


\*) Wird durch das Konfigurationsmanagement verursacht und führt zu einer neuen Produktversion

# V-Modell – Produkte des Submodells: Systemerstellung



# V-Modell – Zusammenwirken der 4 Submodelle



## Rollen im V-Modell

- Beschreiben die notwendigen Erfahrungen, Kenntnisse und Fähigkeiten, um Aktivitäten durchzuführen.
- In jedem Submodell gibt es:
  - einen (Projekt)Manager,
  - einen Verantwortlichen (Projektleiter),
  - einen oder mehrere Durchführende (Projektadministrator, ..., KM-Administrator).

**V-Modell – Rollen**

	<b>Manager</b>	<b>Verantwortliche</b>	<b>Durchführende</b>
<b>PM</b>	Projektmanager	Projektleiter Rechtsverantwortlicher Controller Projektleiter	Projektadministrator
<b>SE</b>	Projektmanager IT-Beauftragter Anwender	Projektleiter	Systemanalytiker Systemdesigner SW-Entwickler HW-Entwickler Technischer Büro SEU-Betreuer Datenadministrator IT-Sicherheitsbeauftragter Datenschutzbeauftragter Systembetreuer
<b>QS</b>	Q-Manager	QS-Verantwortlicher	Prüfer
<b>KM</b>	KM-Manager	KM-Verantwortlicher	KM-Administrator

## Rollen im V-Modell

## Organisationsanalyse

- Folgende Aspekte sind zu berücksichtigen:
  - Organisationsanalyse vorbereiten,
  - Geschäftsprozesse aufnehmen,
  - Geschäftsprozesse analysieren,
  - Analyseergebnisse auswerten.
- Ergebnisse dienen als
  - Ausgangspunkt für die Definition von Anforderungen
  - Basis für eine nachfolgende Geschäftsprozessmodellierung.

## Tailoring

- Tailoring = Maßschneidern
- Zwei Stufen:
  - Ausschreibungsrelevante Tailoring
  - Technisches Tailoring
- Ziel:
  - Vermeiden einer übermäßigen „Papierflut“ sowie sinnloser Dokumentation,
  - Vermeiden vom Fehlen wichtigen Dokumente.

# V-Modell - Zusammenfassung

## Bewertung

- Integrierte, detaillierte Darstellung von Systemerstellung, QS, KM und PM.
- Generisches Vorgehensmodell mit definierten Möglichkeiten der Anpassung.
- Standardisierte Abwicklung von Systemerstellungsprojekten.
- Gut geeignet für große Projekte, insbesondere für eingebettete Systeme.

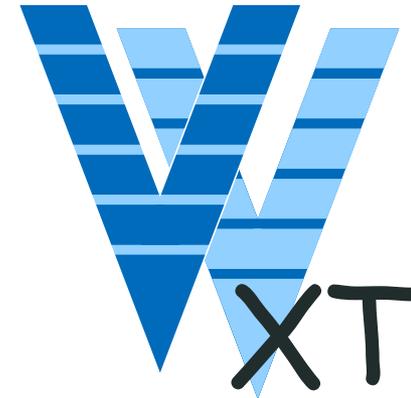
## Vorteile

- Die für eingebettete Systeme sinnvolle Vorgehenskonzepte unkritisch übertragbar
- Unnötige Produktvielfalt und Software-Bürokratie (kleine und mittlere Software-Entwicklungen)
- Nicht handhabbar ohne geeignete CASE-Unterstützung.
- Unrealistische Rollendefinition (außer für Großprojekte).
- Es fehlen in der Dokumentation vollständige Beispiele.
- Gefahr, dass bestimmte Software-Methoden festgeschrieben werden

## Nachteile

# V-Modell XT - Überblick

- **V-Modell XT - das Vorgehensmodell für IT-Projekte**
- **Nachfolger des V-Modells 92 und 97**
- **Standard in der Bundesverwaltung**
- **Für Auftraggeber, Auftragnehmer und Projektorganisationen nutzbar**
- **Ca. 700 Seiten Umfang, liegt als HTML und PDF vor**

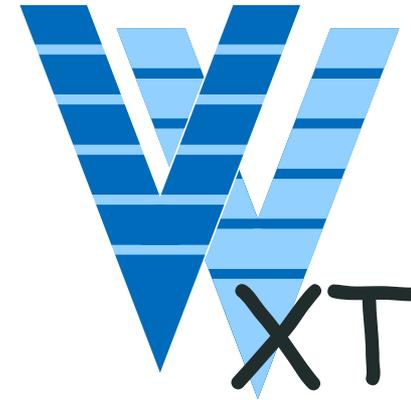


© Bundesrepublik Deutschland, 2004,  
Alle Rechte vorbehalten

Quelle: Thomas Klingenberg, microTOOL GmbH

## V-Modell XT - Historie

- Das V-Modell XT - ein modulares Vorgehensmodell
- XT= „eXtreme Tailoring“
- WEIT-Projekt 2002-2004
  - KBSt/IT-AmtBw
  - TU München / TU Kaiserslautern und Partner
- Kosten ca. 4 Mio. €, ca. 30 beteiligte Personen
- Nach WEIT geht's immer WEITER...
- Version 1.2 seit dem 1. Februar 2006

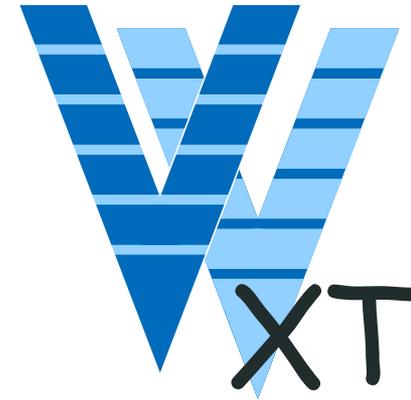


© Bundesrepublik Deutschland, 2004,  
Alle Rechte vorbehalten

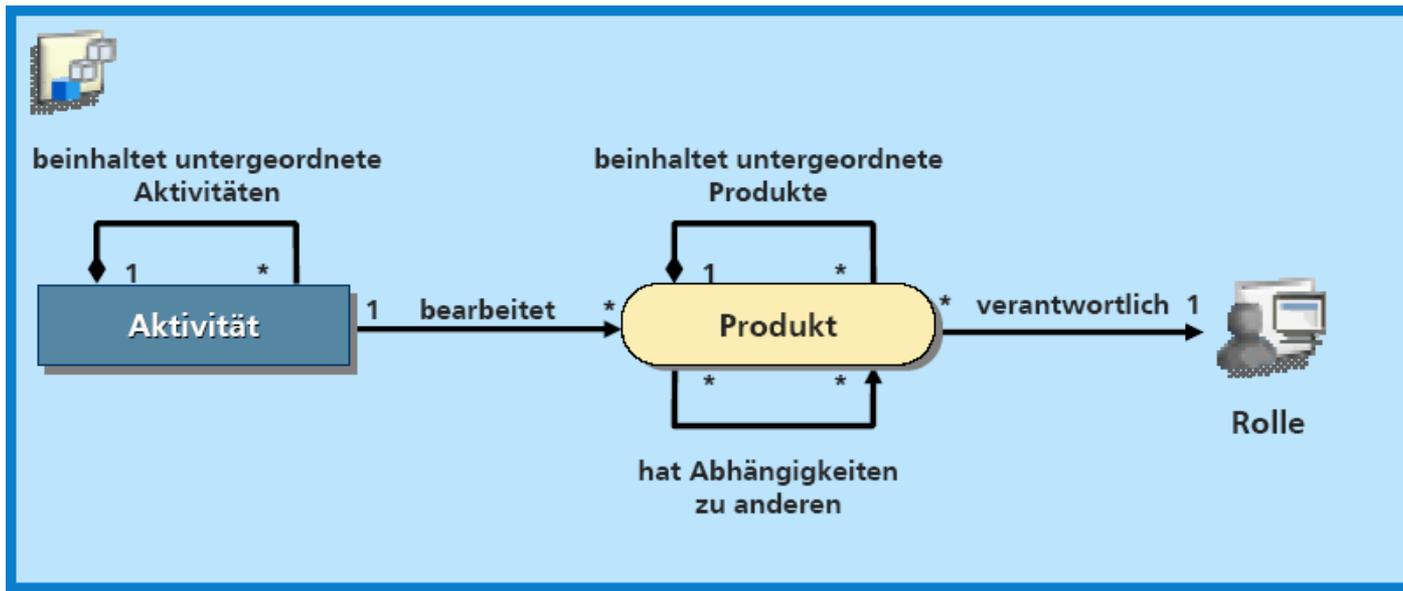
Quelle: Thomas Klingenberg, microTOOL GmbH

# V-Modell XT - Grundlagen

- **Vorgehensbausteine**
  - Kapseln **Aktivitäten**, **Produkte** und **Rollen**
  - sind die modularen Einheiten für das Tailoring
  - können von anderen Vorgehensbausteinen abhängen

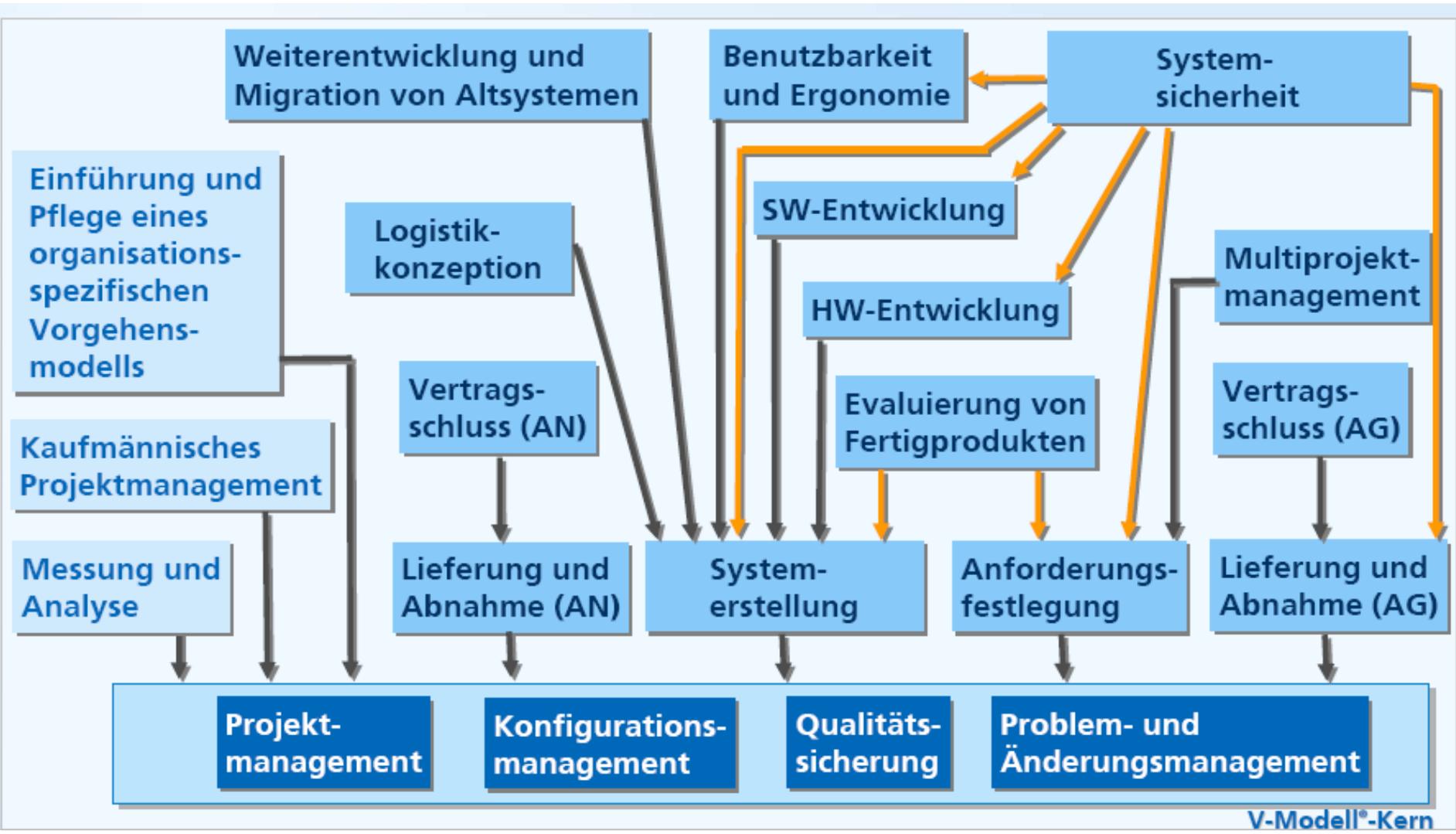


© Bundesrepublik Deutschland, 2004,  
Alle Rechte vorbehalten



Quelle: Thomas Klingenberg, microTOOL GmbH

# V-Modell XT - Vorgehensbausteinkarte



Quelle: Thomas Klingenberg, microTOOL GmbH

# Was ist neu am V-Modell XT 1.2 ?

- **Multiprojektmanagement für Auftraggeber**
- **Veränderte Projektdurchführungsstrategien:  
Iterationen und Parallelitäten in der Planung**
- **Neuer Projekttyp Systementwicklung AG/AN**

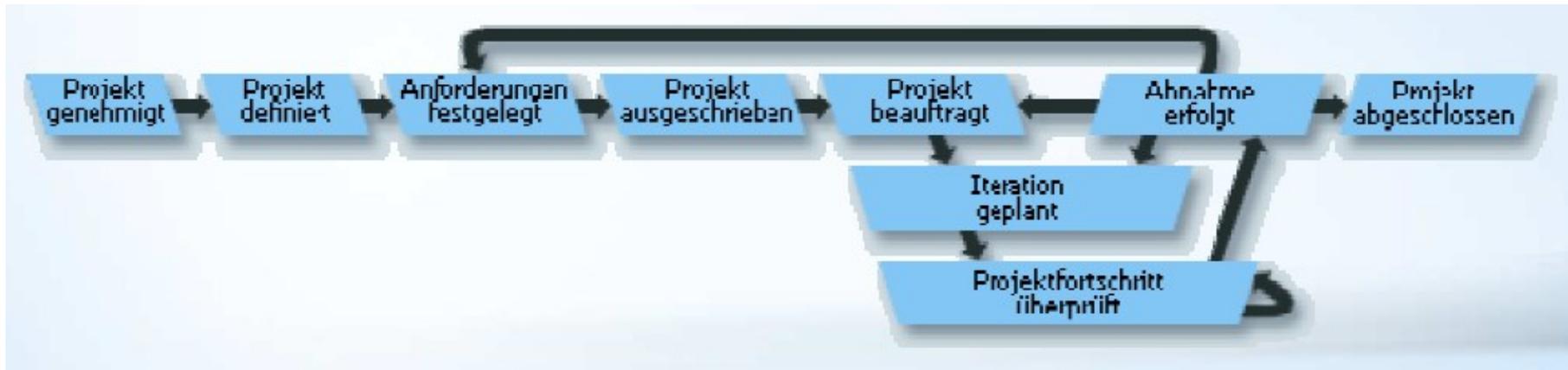


© Bundesrepublik Deutschland, 2004,  
Alle Rechte vorbehalten

Quelle: Thomas Klingenberg, microTOOL GmbH

# Multiprojektmanagement für Auftraggeber

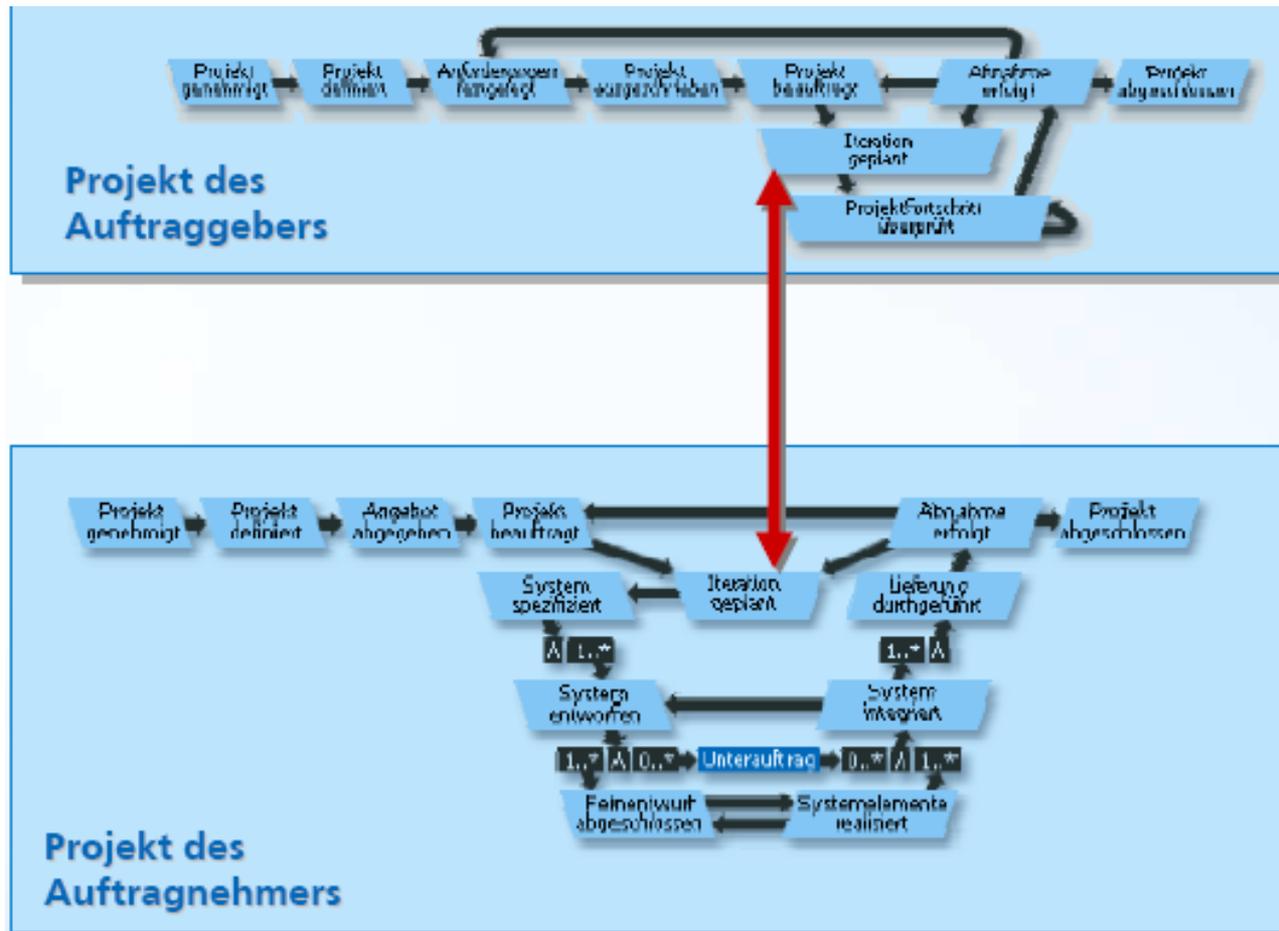
- Vergabe und Durchführung eines Systementwicklungsprojektes (AG)



Quelle: Thomas Klingenberg, microTOOL GmbH

# Multiprojektmanagement für Auftraggeber

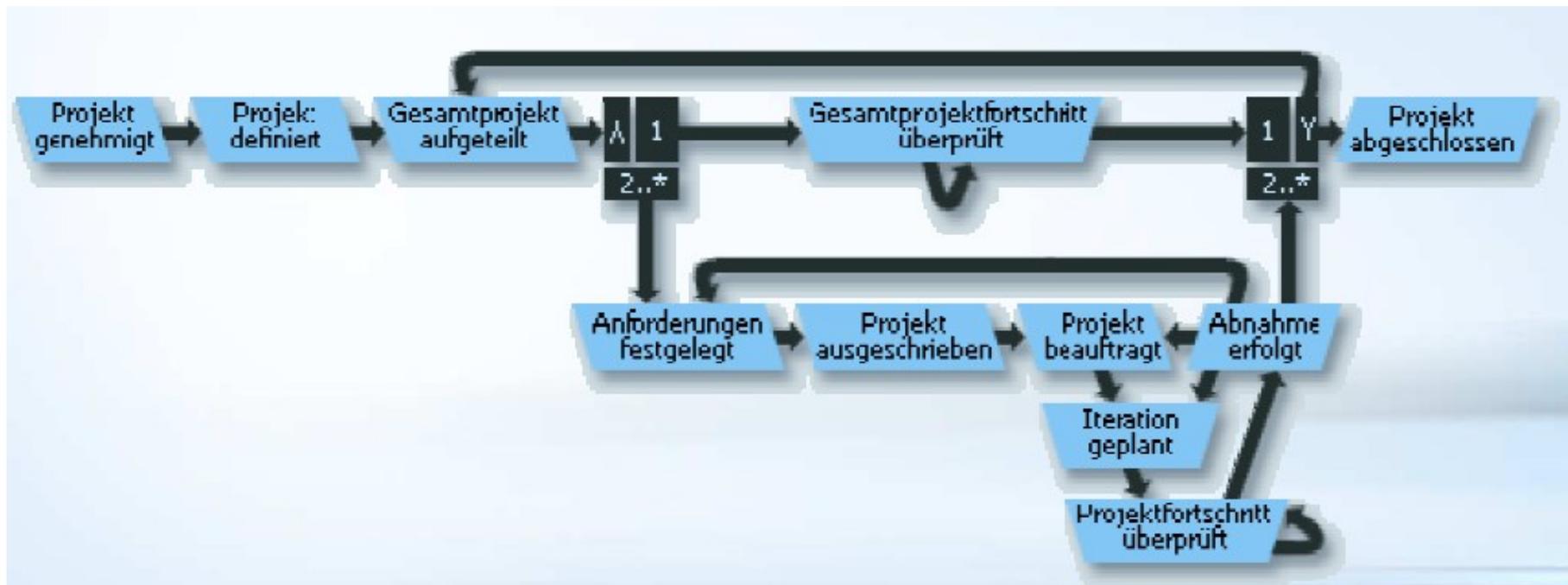
- Gemeinsames Planen der Iterationen



Quelle: Thomas Klingenberg, microTOOL GmbH

# Multiprojektmanagement für Auftraggeber

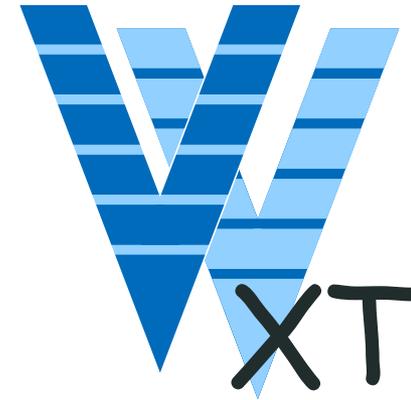
- Vergabe und Durchführung mehrerer Systementwicklungsprojektes (AG)



Quelle: Thomas Klingenberg, microTOOL GmbH

## Was ist neu am V-Modell XT 1.2 ?

- **Multiprojektmanagement für Auftraggeber**
- **Veränderte Projektdurchführungsstrategien:  
Iterationen und Parallelitäten in der Planung**
- **Neuer Projekttyp Systementwicklung AG/AN**

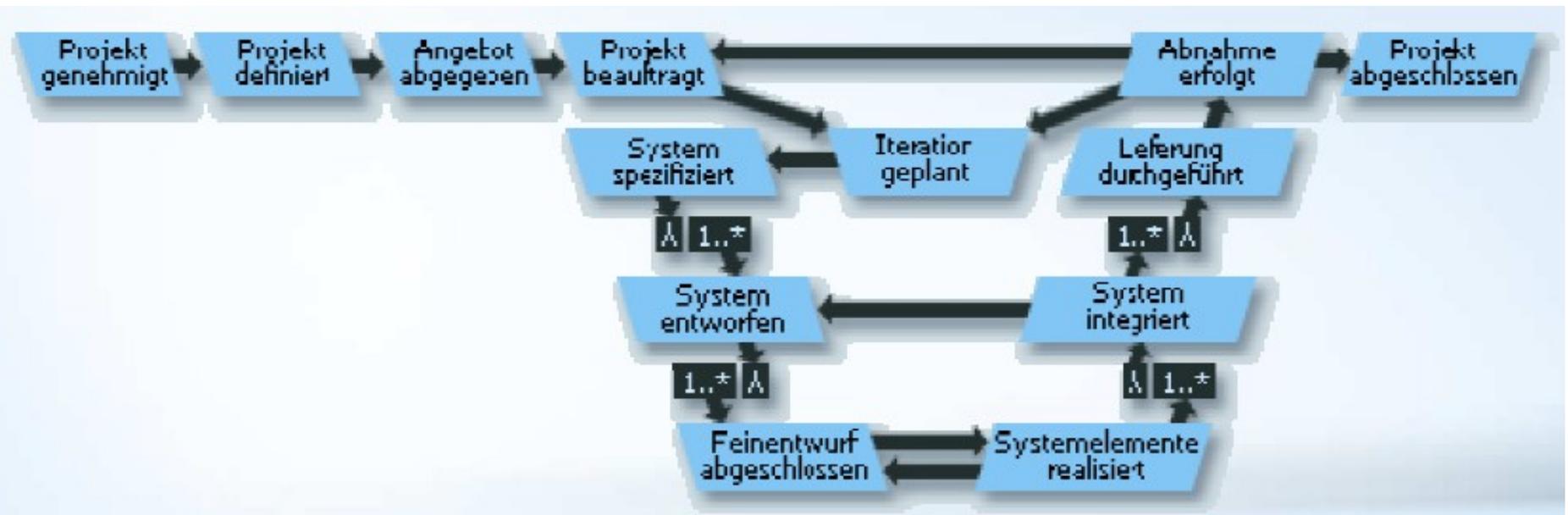


© Bundesrepublik Deutschland, 2004,  
Alle Rechte vorbehalten

Quelle: Thomas Klingenberg, microTOOL GmbH

# Projektdurchführungsstrategien

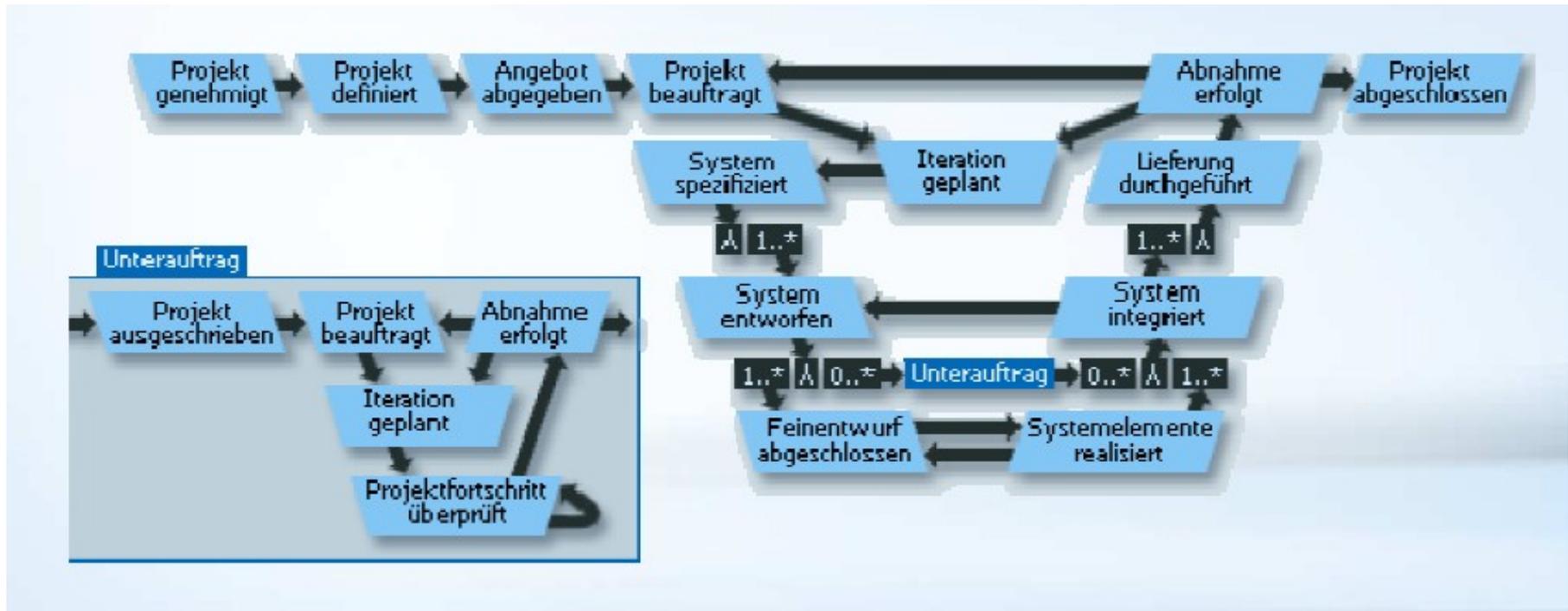
- Inkrementelle Systementwicklung (AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

# Projektdurchführungsstrategien

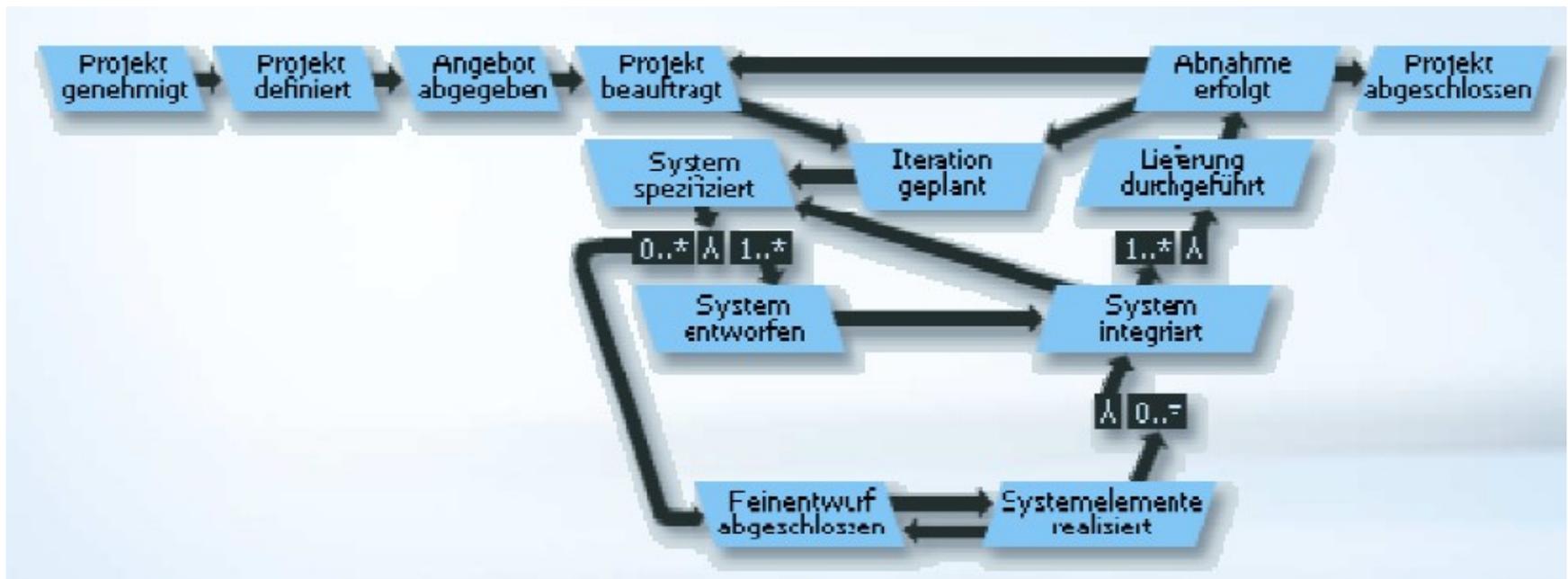
- Inkrementelle Systementwicklung (AN) mit Unteraufträgen



Quelle: Thomas Klingenberg, microTOOL GmbH

# Projektdurchführungsstrategien

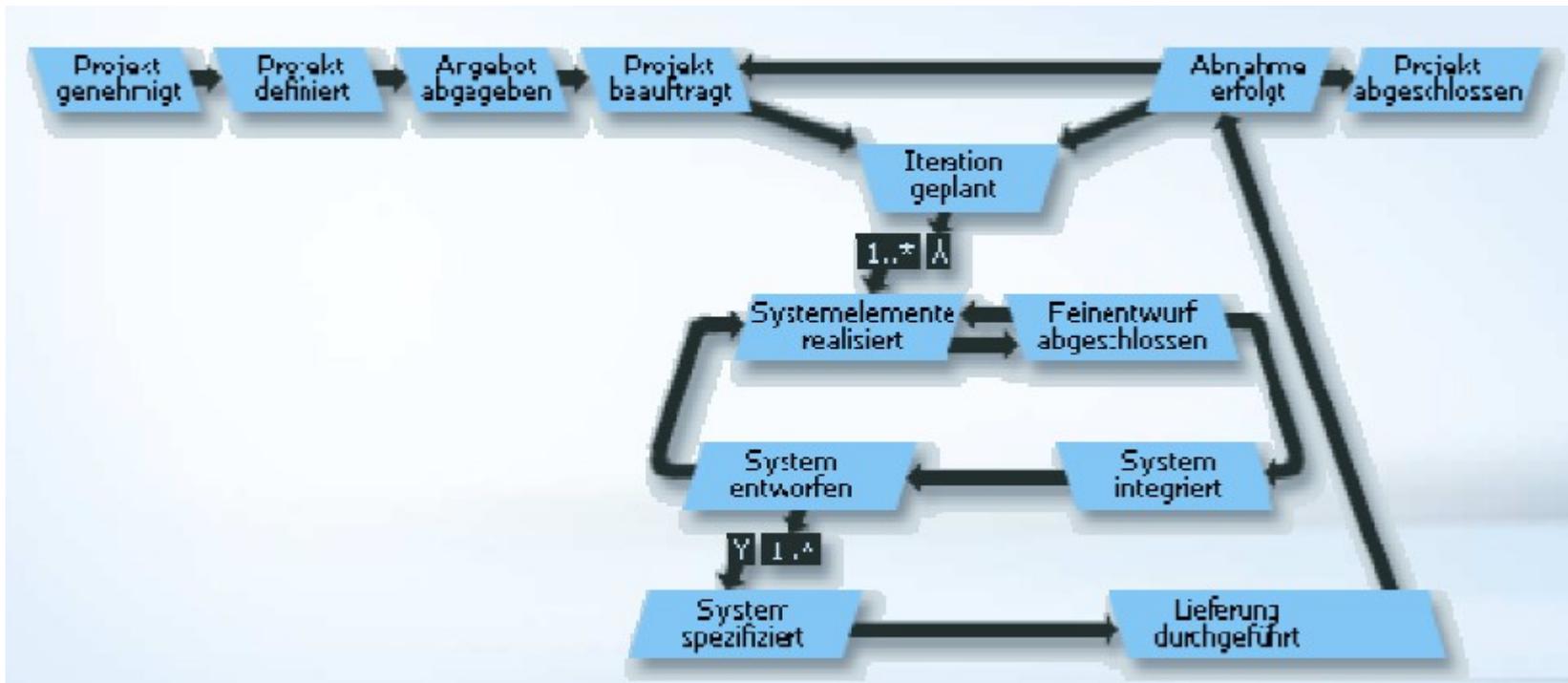
- Komponentenbasierte Systementwicklung (AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

# Projektdurchführungsstrategien

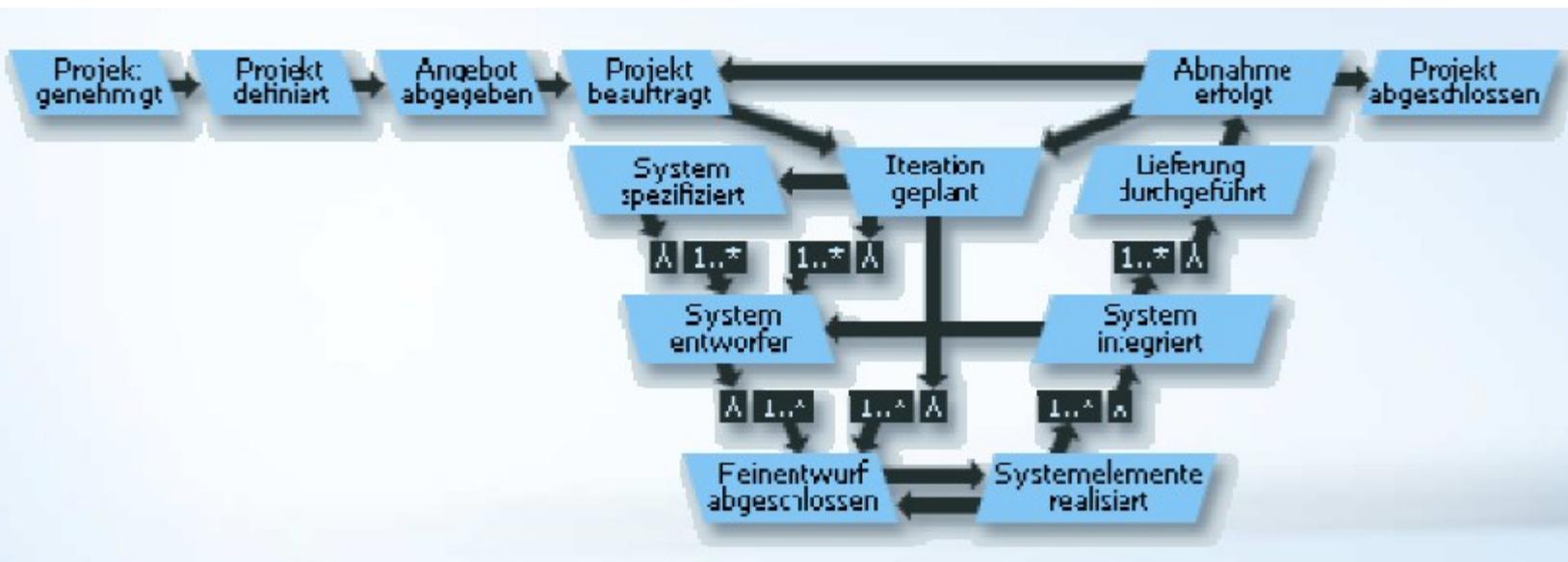
- Agile Systementwicklung (AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

## 4. Projektdurchführungsstrategien

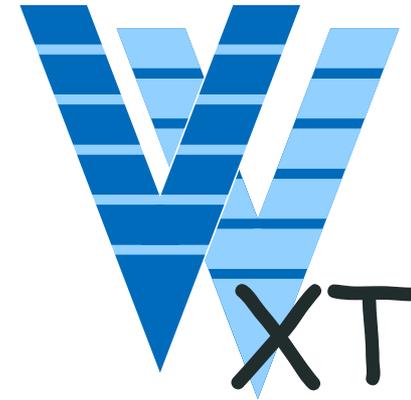
- Wartung und Pflege von Systemen (AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

## Was ist neu am V-Modell XT 1.2 ?

- **Multiprojektmanagement für Auftraggeber**
- **Veränderte Projektdurchführungsstrategien:  
Iterationen und Parallelitäten in der Planung**
- **Neuer Projekttyp Systementwicklung AG/AN**

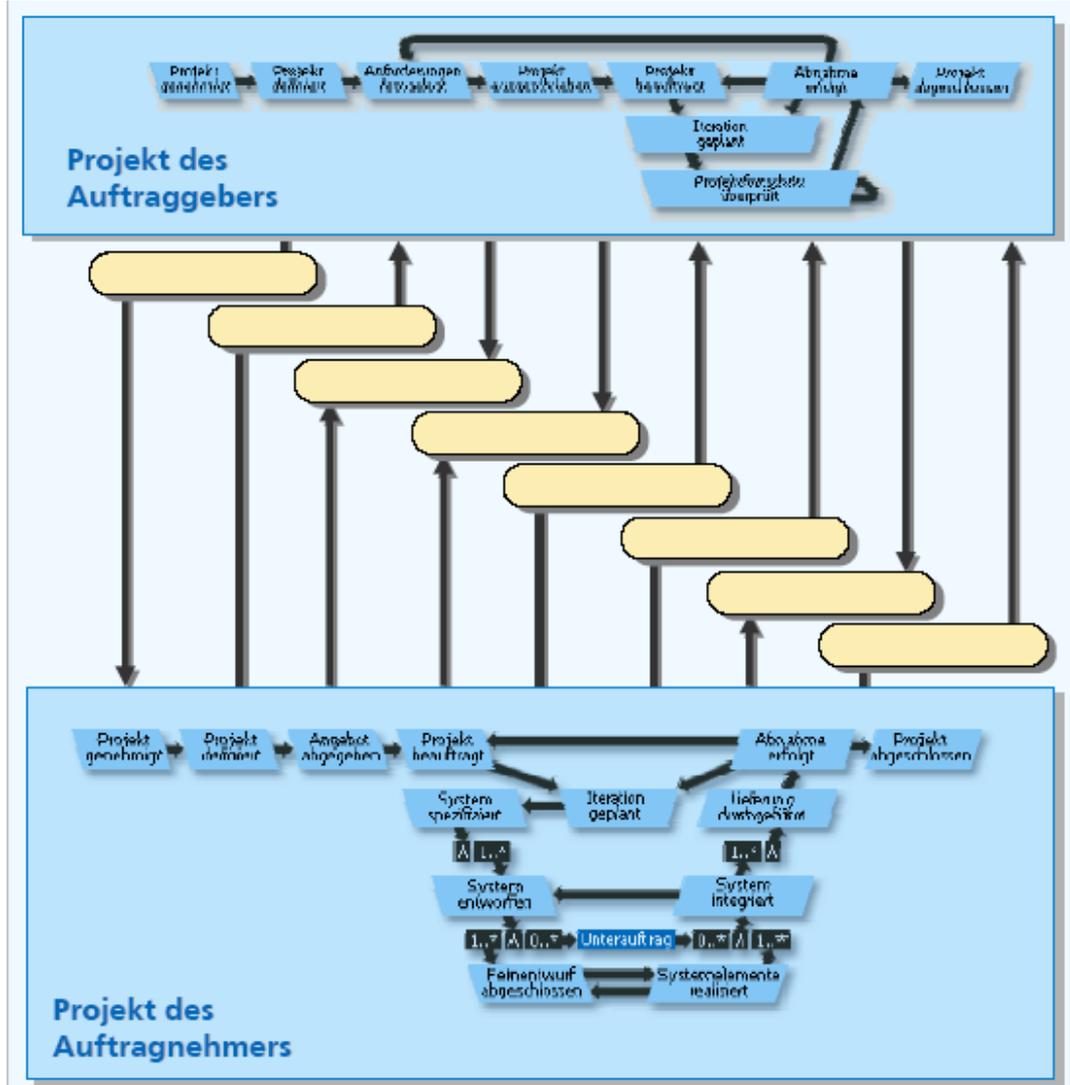


© Bundesrepublik Deutschland, 2004,  
Alle Rechte vorbehalten

Quelle: Thomas Klingenberg, microTOOL GmbH

# Systementwicklung AG/AN

- Die Trennung in Auftraggeber und Auftragnehmer passt nicht immer!



Quelle: Thomas Klingenberg, microTOOL GmbH

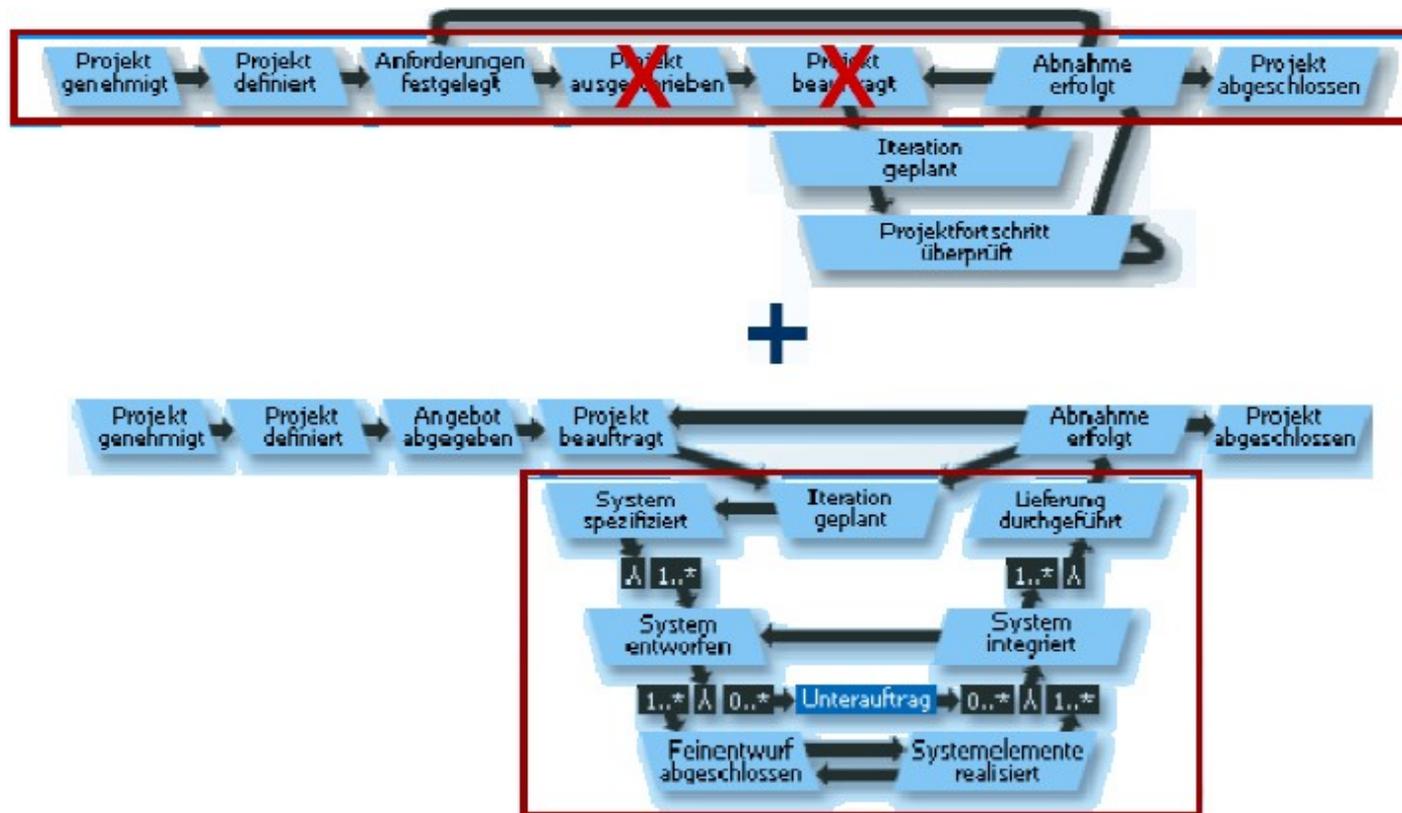
# Systementwicklung AG/AN

- Neuer Projekttyp **Systementwicklung AG/AN**
- Für Projekte
  - innerhalb einer Organisation oder
  - in zwei eng kooperierenden Organisationen
  - wo es nur **1 Projektleiter** gibt.
- Trotzdem erfolgt die Trennung in die fachliche fachliche und technische Sicht technische Sicht, daher „AG/AN“

Quelle: Thomas Klingenberg, microTOOL GmbH

## 4. Systementwicklung AG/AN

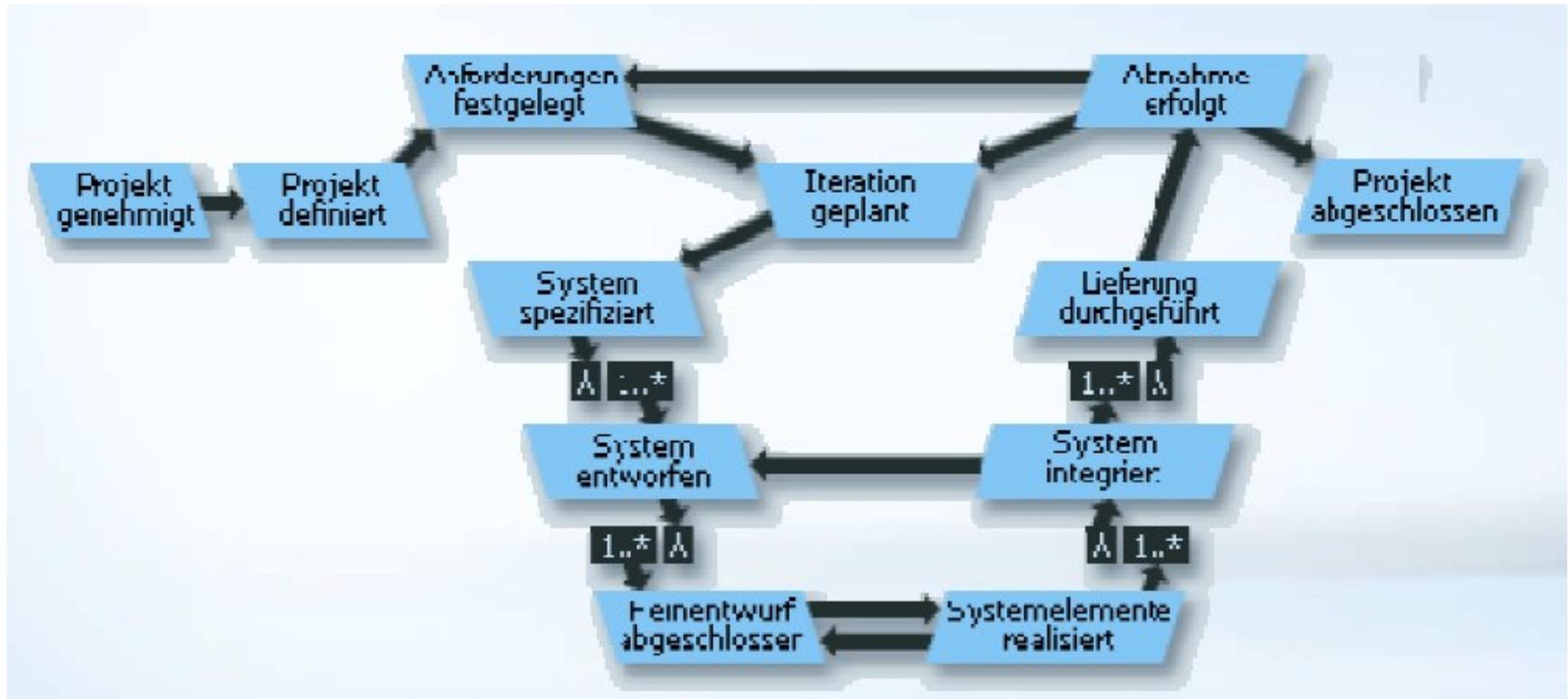
- Die Projektdurchführungsstrategien des Projekttyps „Systementwicklung (AG/AN)“ als Kombinationslösung



Quelle: Thomas Klingenberg, microTOOL GmbH

# Systementwicklung AG/AN

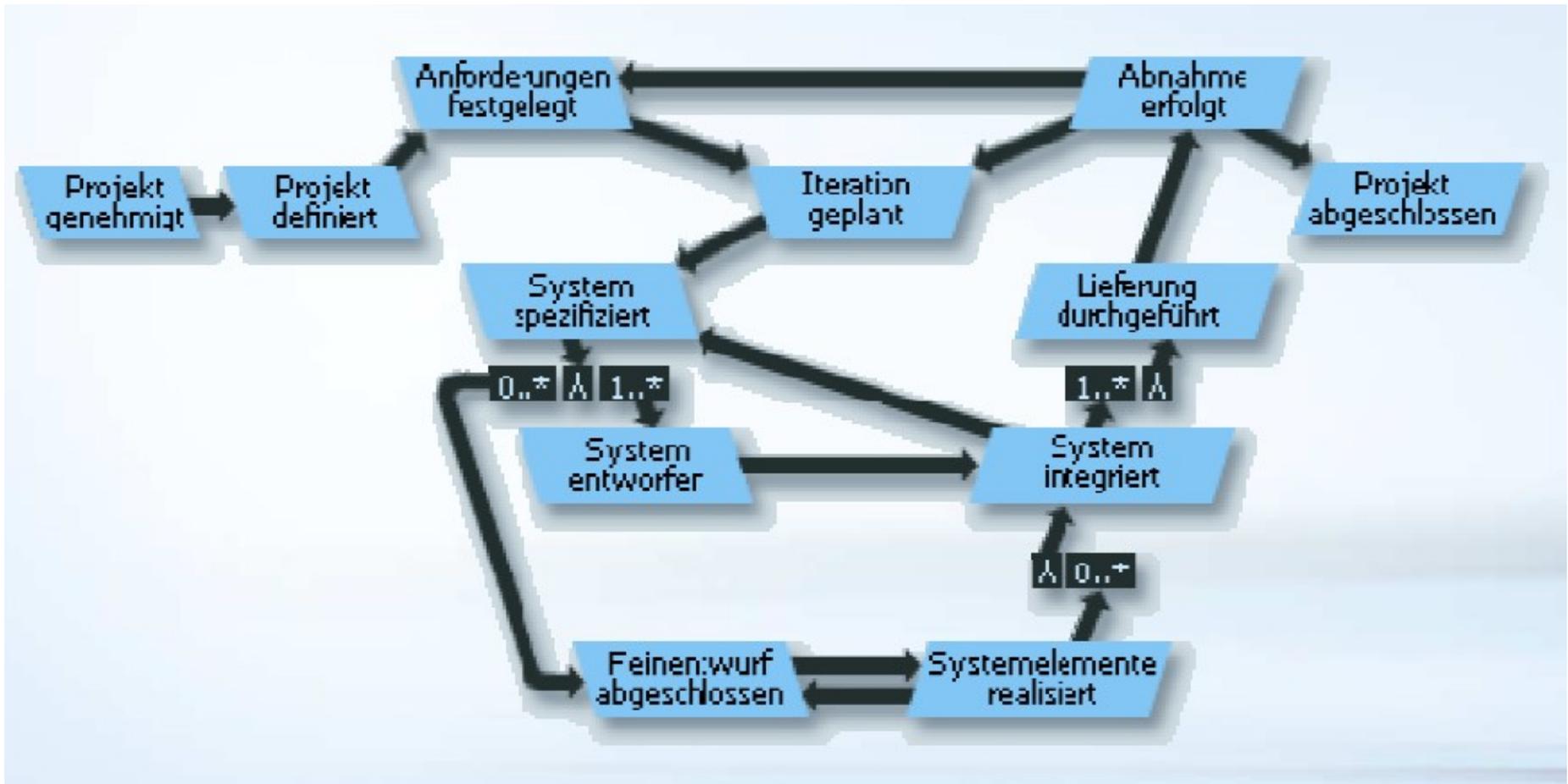
- Inkrementelle Systementwicklung (AG/AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

## Systementwicklung AG/AN

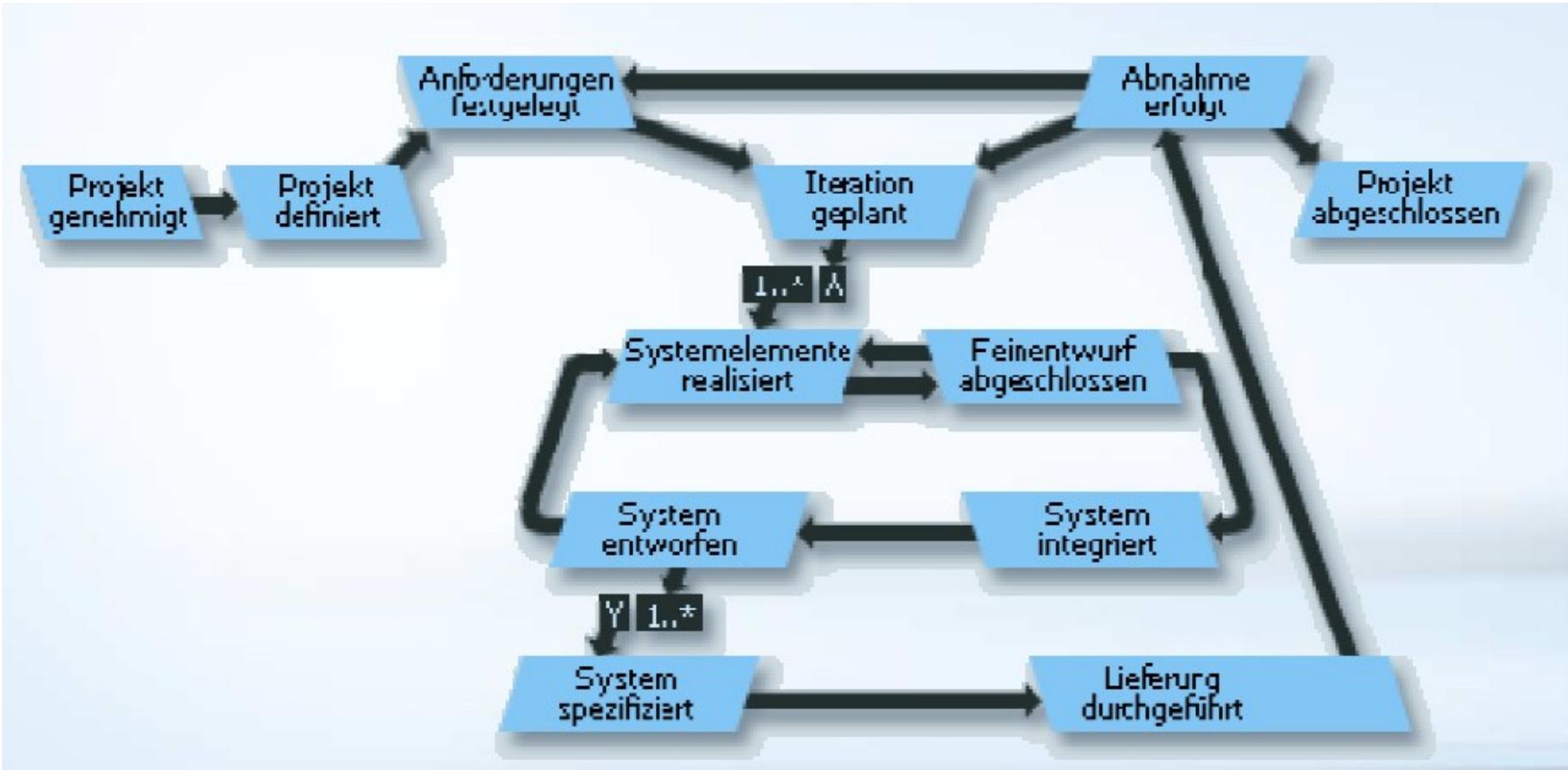
- Komponentenbasierte Systementwicklung (AG/AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

# Systementwicklung AG/AN

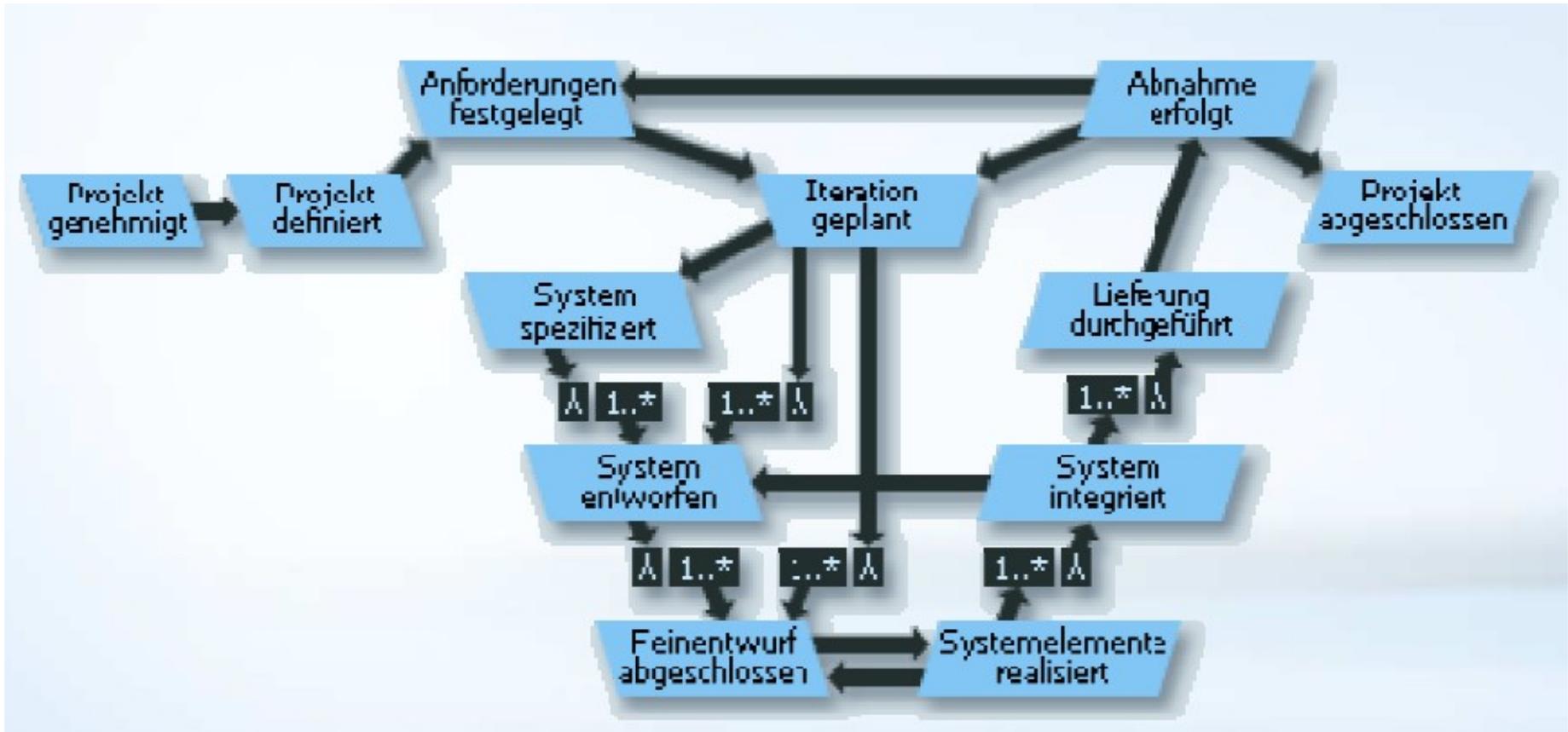
- Agile Systementwicklung (AG/AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

# Systementwicklung AG/AN

- Wartung und Pflege von Systemen (AG/AN)



Quelle: Thomas Klingenberg, microTOOL GmbH

# Das Prototypen-Modell

- Unterschiede zwischen einem Software-Prototypen und einem Prototypen in anderen Ingenieursdisziplinen. Ein Software-Prototyp:
  - ist nicht das „erste Muster“ einer großen Serie von Produkten (da Vervielfältigung kein Ingenieursproblem).
  - zeigt ausgewählte Eigenschaften des Zielproduktes im praktischen Einsatz.
- Ähnlichkeiten in der Anwendung der Prototypen:
  - Sie werden verwendet, um relevante Anforderungen oder Entwicklungsprobleme zu klären.
  - Sie dienen als Diskussionsbasis und helfen bei Entscheidungen.
  - Sie werden für experimentelle Zwecke verwendet und um praktische Erfahrungen zu sammeln.
- **Prototypen-Modell** unterstützt die Erstellung ablauffähiger Modelle des zukünftigen Produkts.

# Das Prototypen-Modell

Vier Arten von Prototypen:

- Demonstrationsprototyp,
- Prototyp im engeren Sinne (für Analyse im Anwendungsbereich),
- Labormuster,
- Pilotsystem.

Klassifikation aus der Sicht des fertigen Software-Produkts:

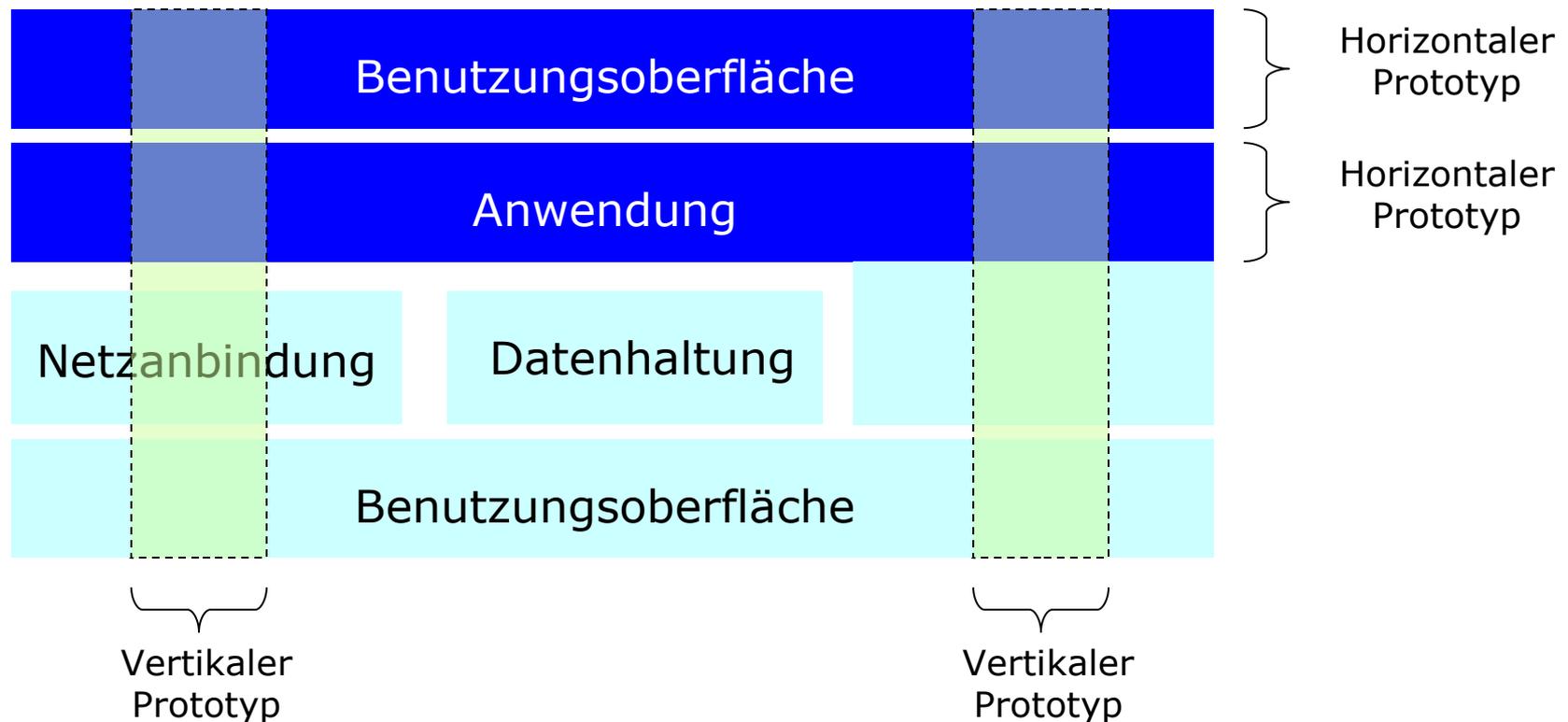
- Horizontaler Prototyp,
- Vertikaler Prototyp.

Beziehungen zwischen Prototypen und Software-Systemen:

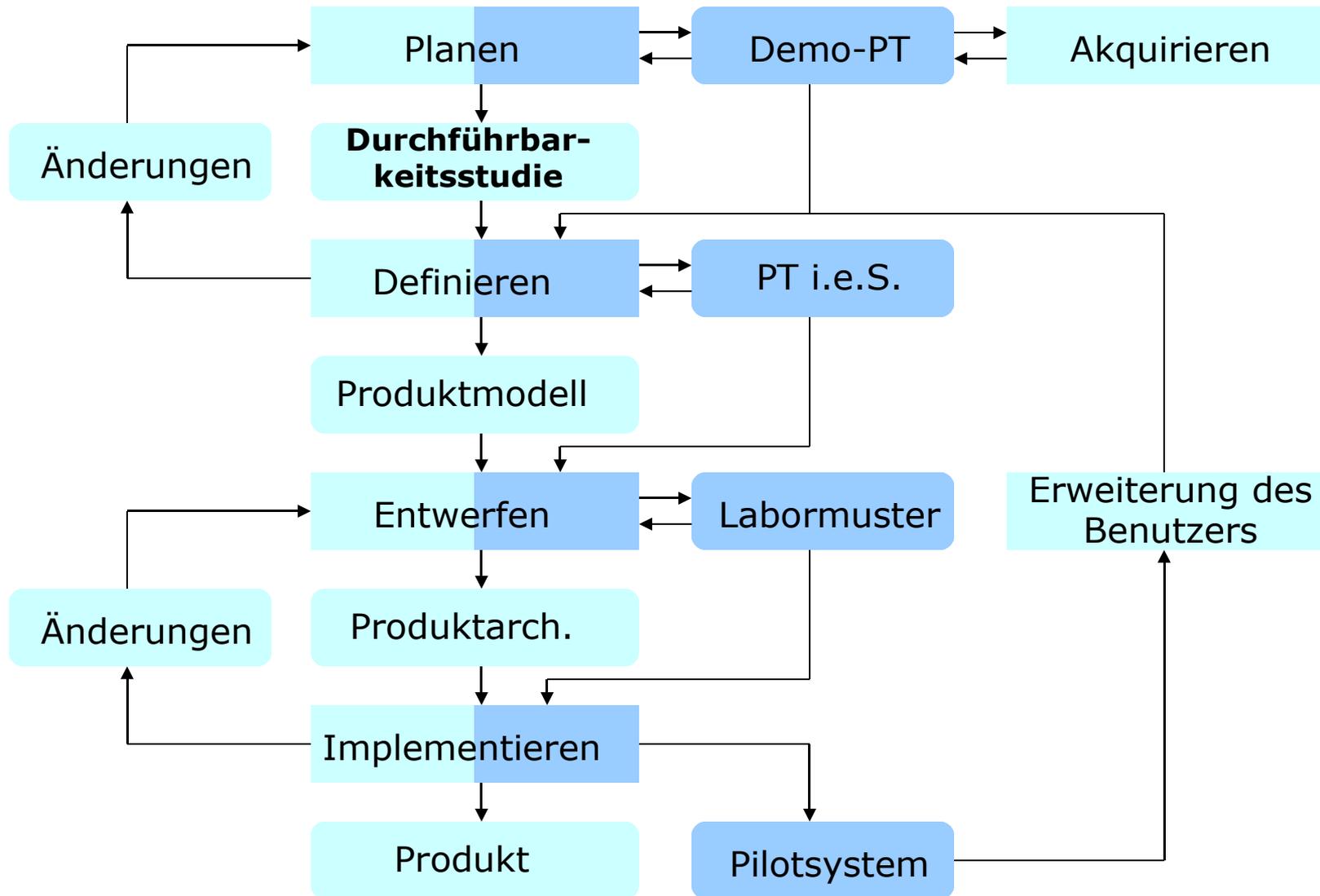
- Prototypen dienen zur Klärung von Problemen.
- Ein Prototyp ist Teil der Produktdefinition.
- Prototypen werden inkrementell weiterentwickelt, um ein marktfähiges Produkt zu erhalten.

## 5. Das Prototypen-Modell

### Horizontaler und vertikaler Prototyp



## Das Prototypen-Model



# Prototypen-Modell - Zusammenfassung

## Bewertung

### Vorteile

- Reduzierung des Entwicklungsrisikos.
- Kann sinnvoll in andere Prozessmodelle integriert werden.
- Prototypen können durch Werkzeuge erstellt werden.
- Verbessert die Planung der Software-Entwicklung.
- Labormuster fordern die Kreativität für Lösungsalternativen.
- Starke Rückkopplung mit dem Endbenutzer und dem Auftraggeber.

- Höherer Entwicklungsaufwand.
- Gefahr der Integration eines Wegwerf-Prototyps im Produkt.
- Verträge für die Software-Erstellung berücksichtigen noch nicht das Prototypen-Modell.
- Prototypen werden oft als Ersatz für die fehlende Dokumentation gesehen.
- Die Beschränkungen und Grenzen von Prototypen sind oft nicht bekannt.

### Nachteile

## Das evolutionäre Modell

**Ausgangspunkt:** Kernanforderungen des Auftraggebers

### Charakteristika

- Stufenweise Entwicklung des Produkts.
- Pflegeaktivitäten gelten als Erstellung einer neuen Version.
- Gut geeignet, wenn der Auftraggeber seine Anforderungen noch nicht vollständig überblickt.
- Code-getriebene Entwicklung.



# Evolutionäre Modell - Zusammenfassung

## Bewertung

### Vorteile

- Auftraggeber erhält in kurzen Zeitabständen einsatzfähige Produkte.
- Kann mit dem Prototypen-Modell kombiniert werden.
- Auswirkungen des Produkteinsatzes gut in die folgende Version einzubringen.
- Richtung der Entwicklung korrigierbar durch schrittweise Entwicklung.
- Entwicklung nicht nur auf ein einzigen Endabgabetermin ausgerichtet.

- Gefahr, dass im nachfolgenden Modell die gesamte Architektur überarbeitet werden muss.
- Gefahr, dass die Nullversion nicht flexibel genug ist, um sich an geplante Evolutionspfade anzupassen.

### Nachteile

## Das inkrementelle Modell

- Nachteile des evolutionären Modells werden bei dem **inkrementellen Modell** vermieden.
- Alle Anforderungen an das zu modellierende Produkt werden möglichst vollständig erfasst und modelliert.
- Es wird aber nur ein Teil der Anforderungen entworfen und implementiert (analog zum evolutionären Modell).
- Wegen dem vollständigen Analysemodell ist sichergestellt, dass die Erweiterungen zu dem bisherigen System passen.



## 7. Das objektorientierte Modell

### Wiederverwendung

- Wesentlicher Vorteil einer OO-Entwicklung: Wiederverwendung
- Wiederverwendungsebenen:
  - Ebene der OOA-Modelle,
  - Ebene der technischen Entwürfe,
  - Ebene der implementierten Klassen und Klassenbibliotheken
- Wiederverwendungsgebiete:
  - Anwendungs- bzw. branchenspezifische Klassen und Subsysteme,
  - Klassen, die die Anbindung einer Anwendung an die Umgebung ermöglichen,
  - Klassen, die die Anbindung an die Systemsoftware ermöglichen
- Herkunft der Klassen:
  - frühere Entwürfe,
  - auf dem Markt eingekaufte Klassenbibliotheken

## 7. Das objektorientierte Modell

### Wiederverwendung - Charakteristika

Mögliche Zeitpunkte des Einsatzes von wiederverwendbaren Klassen:

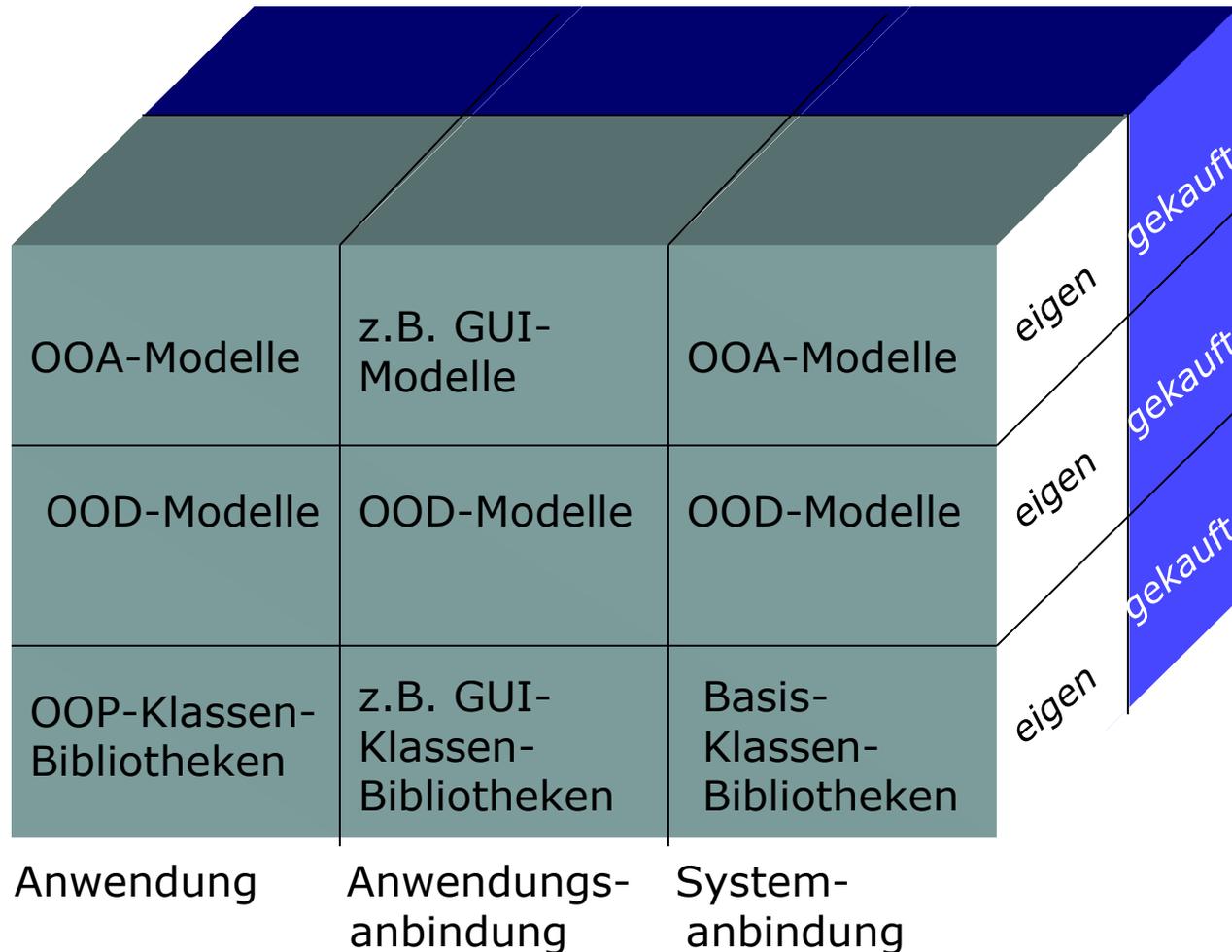
- während der laufenden Entwicklung,
  - am Ende der laufenden Entwicklung,
  - nach einer nachträglichen Analyse abgeschlossener Entwicklungen
- Starker *bottom-up*-Aspekt aufgrund der Wiederverwendbarkeit

#### Charakteristika

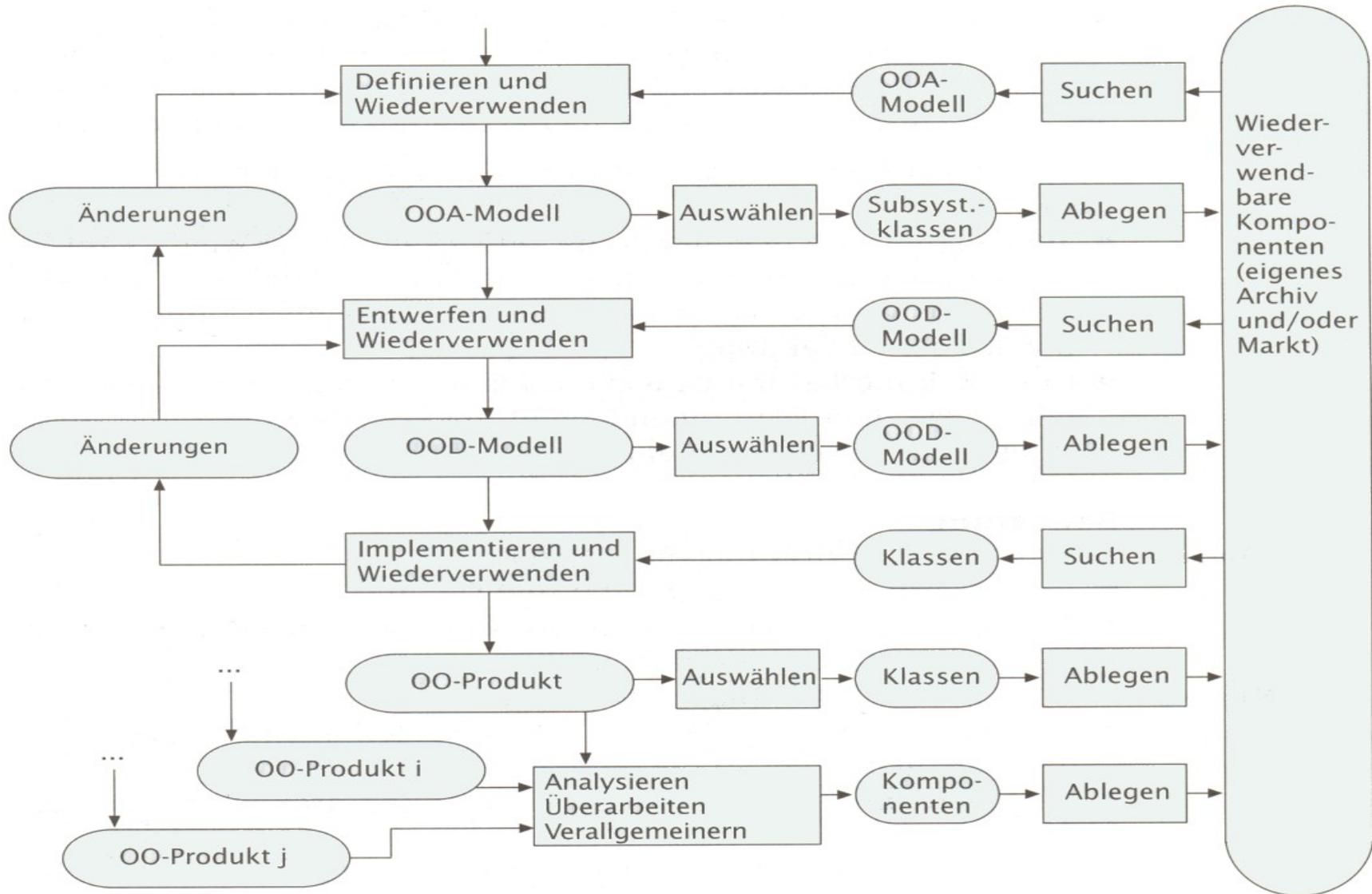
- Tendenz zur Entwicklung in voller Breite.
- Fokus auf Wiederverwendung.
- Gut kombinierbar mit dem evolutionären, dem inkrementellen und dem Prototypen-Modell.

## 7. Das objektorientierte Modell

### Quader der Wiederverwendbarkeits-Klassifikation



## 7. Das objektorientierte Modell



## 7. Das evolutionäre/inkrementelle Modell

### Bewertung

#### Vorteile

- Verbesserung der Produktivität und Qualität;
- Konzentration auf die eigenen Stärken, Rest zukaufen.

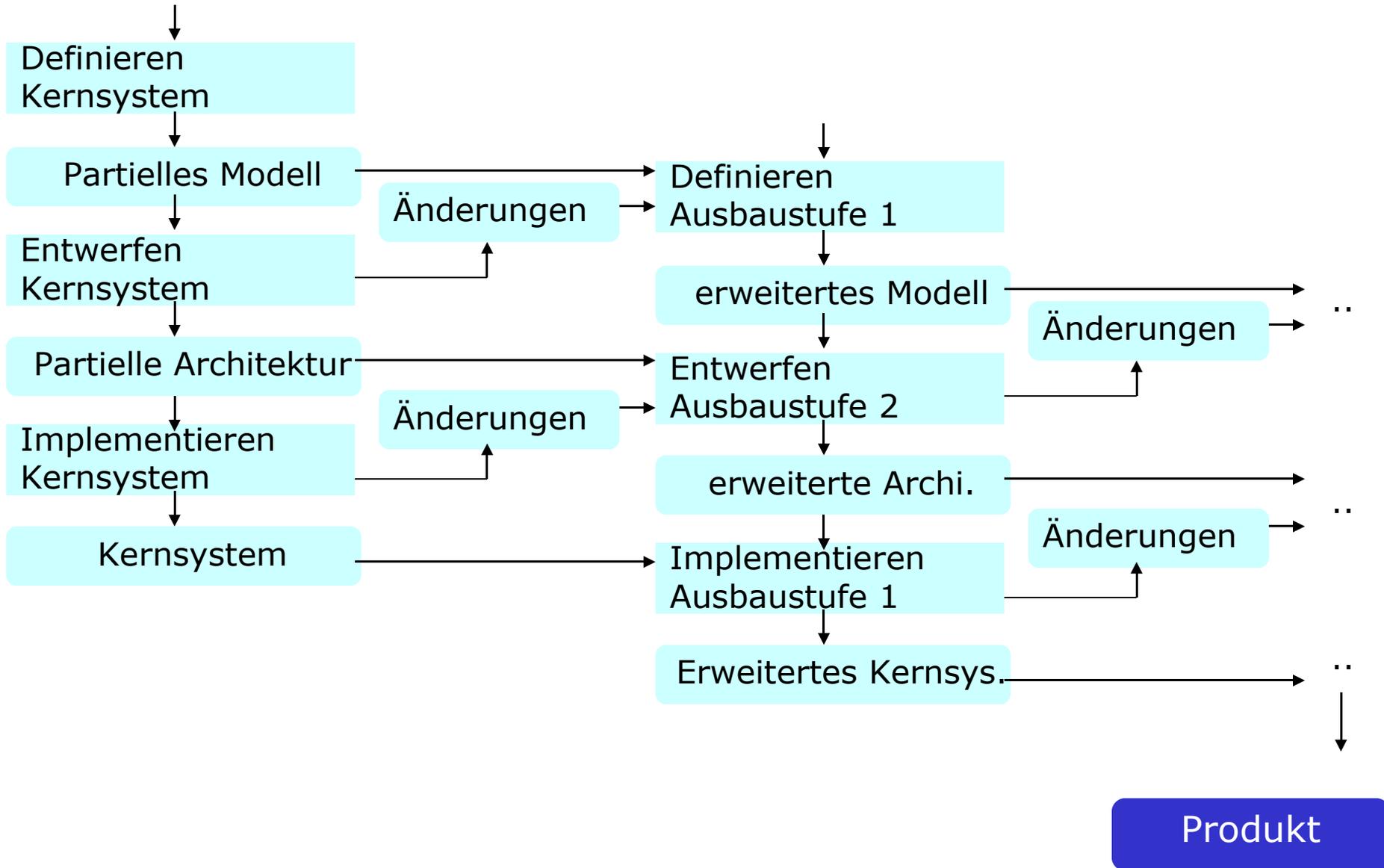
- Null voll nutzbar, wenn OO-Methoden eingesetzt werden.
- Geeignete Infrastruktur (Widerverwendungsarchiv) muss vorhanden sein.
- Firmenkultur der Wiederverwendung muss aufgebaut sein.
- Technische Probleme müssen überwunden werden.

#### Nachteile

## 8. Das nebenläufige Modell

- Das **nebenläufige Prozess-Modell** stellt einen Ansatz dar, um eine termingerechte Fertigstellung zu erreichen [Spectrum 91].
- Stammt aus der Fertigungsindustrie.
- Alle betroffenen Entwicklungsabteilungen sind in einem Team vereint.
- Es soll ein möglichst hoher Parallelisierungsgrad erreicht werden.
- Risiken der Parallelisierung:
  - Verwendung unreifer Entwürfe führt zu Verschrottung der Werkzeuge.
  - Erfordert ein ständiges Abwägen zwischen finanziellen Risiken und vorzeitiger Parallelisierung von Arbeitsfolgen.

# 8. Das nebenläufige Modell



## 8. Das nebenläufige Modell

### Charakteristika

- Versuch einer Parallelisierung des prinzipiell sequentiellen Entwicklungsprozess.
- Förderung des auf Problemlösung gerichteten Zusammenarbeiten der betreffenden Personengruppen.
- Reduzierung von Zeitverzögerungen durch:
  - Teilweise Parallelisierung vorwiegend sequentiell organisierter Arbeiten.
  - Minimierung des Ausprobierens und Begrenzung des Improvisierens.
  - Reduktion der Wartezeiten.
- Frühzeitiges Zusammenbringen aller Erfahrungen der betroffenen Personengruppen.
- Auslieferung des vollständigen Produkts ist das Ziel.

## 8. Das nebenläufige Modell

### Bewertung

#### Vorteile

- Frühes Erkennen und Eliminieren von Problemen durch Beteiligung aller betroffenen Personengruppen.
- Optimale Zeitausnutzung.

- Es ist fraglich, ob es wegen der speziellen Charakteristika der Software möglich ist, das Ziel „right the first time“ zu erreichen.
- Risiko, dass die grundlegenden und kritischen Entscheidungen zu spät getroffen werden und dadurch Iterationen nötig werden.
- Hoher Planungs- und Personalaufwand.

#### Nachteile

## 8. Das nebenläufige Modell

### Bewertung

#### Vorteile

- Überprüfung in periodischen Intervallen und u. U. erneute Festlegung des Prozessablaufs in Abhängigkeit von den Risiken.
- Prozess-Modell nicht für die gesamte Entwicklung festgelegt.
- Integration anderer Prozess-Modelle als Spezialfälle.
- Fehler und ungeeignete Alternativen werden frühzeitig eliminiert.
- Flexibles Modell.
- Leichtere Neudefinition der Entwicklungsziele wenn nötig.
- Unterstützung der Wiederverwendung von Software.

- Hoher Managementaufwand.
- Ungeeignet für kleine und mittlere Projekte.
- Identifizieren und Managen von Risiken schwierig.

#### Nachteile

## 9. Das Spiral-Modell

- **Spiralmodell** von [Boehm 86,88] ist Metamodell.
- Für jedes Produkt und jede Verfeinerungsebene sind 4 Schritte zu durchlaufen.

Schritt 1

- Identifikation des Teilprodukts.
- Alternative Möglichkeiten, um das Teilprodukt zu realisieren.
- Randbedingungen bei den verschiedenen Alternativen.

- Evaluierung der Alternativen unter Berücksichtigung der Ziele und Randbedingungen.
- Entwicklung einer kosteneffektiven Strategie im Falle von Risiken.

Schritt 2

Schritt 3

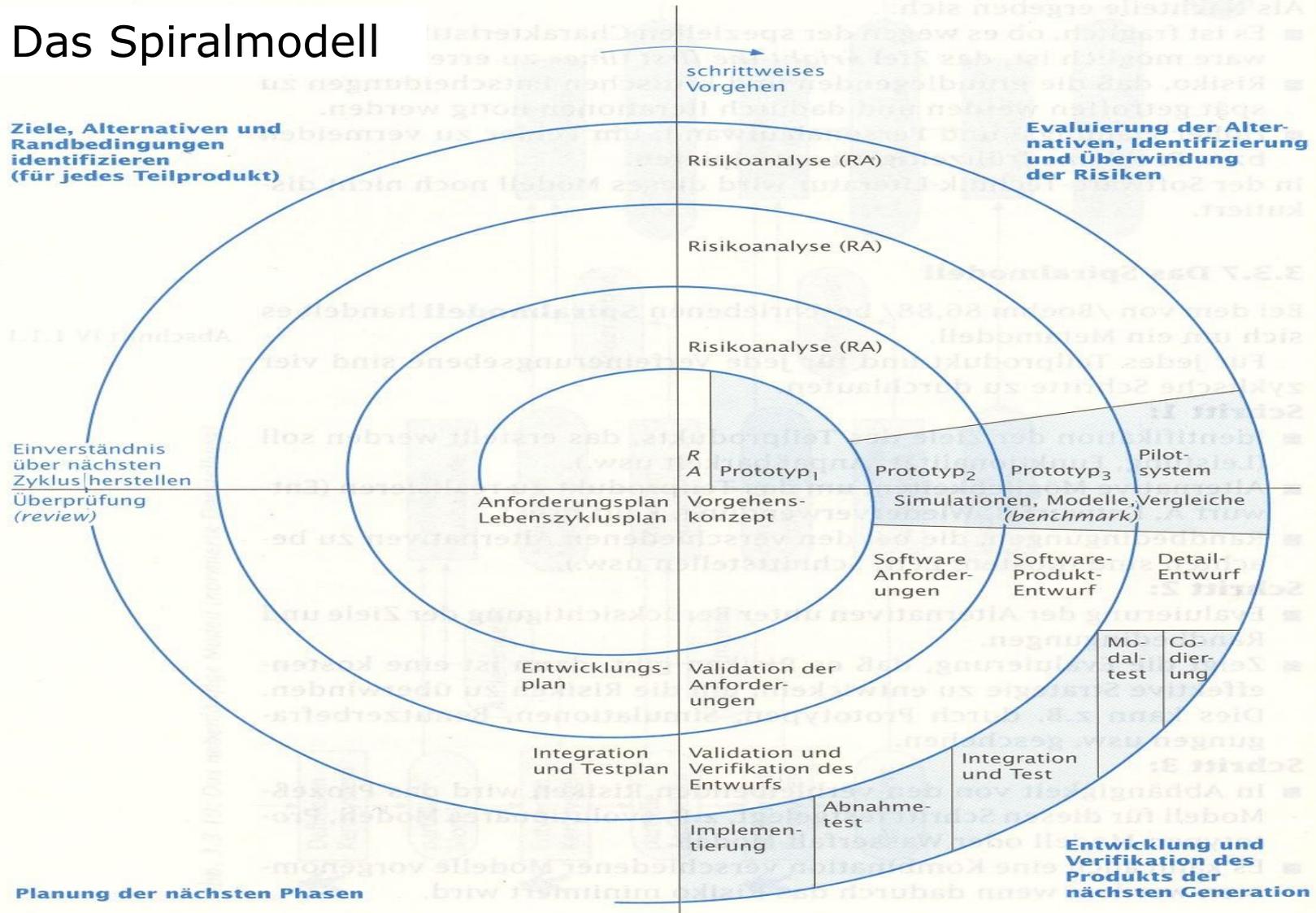
- Festlegung des Prozessmodells für diesen Schritt.
- Die Kombination verschiedener Modelle kann vorgenommen werden.

- Planung des nächsten Zyklus.
- Überprüfung (*re-view*) der Schritte 1 bis 3.
- Einverständnis über den nächsten Zyklus herstellen.

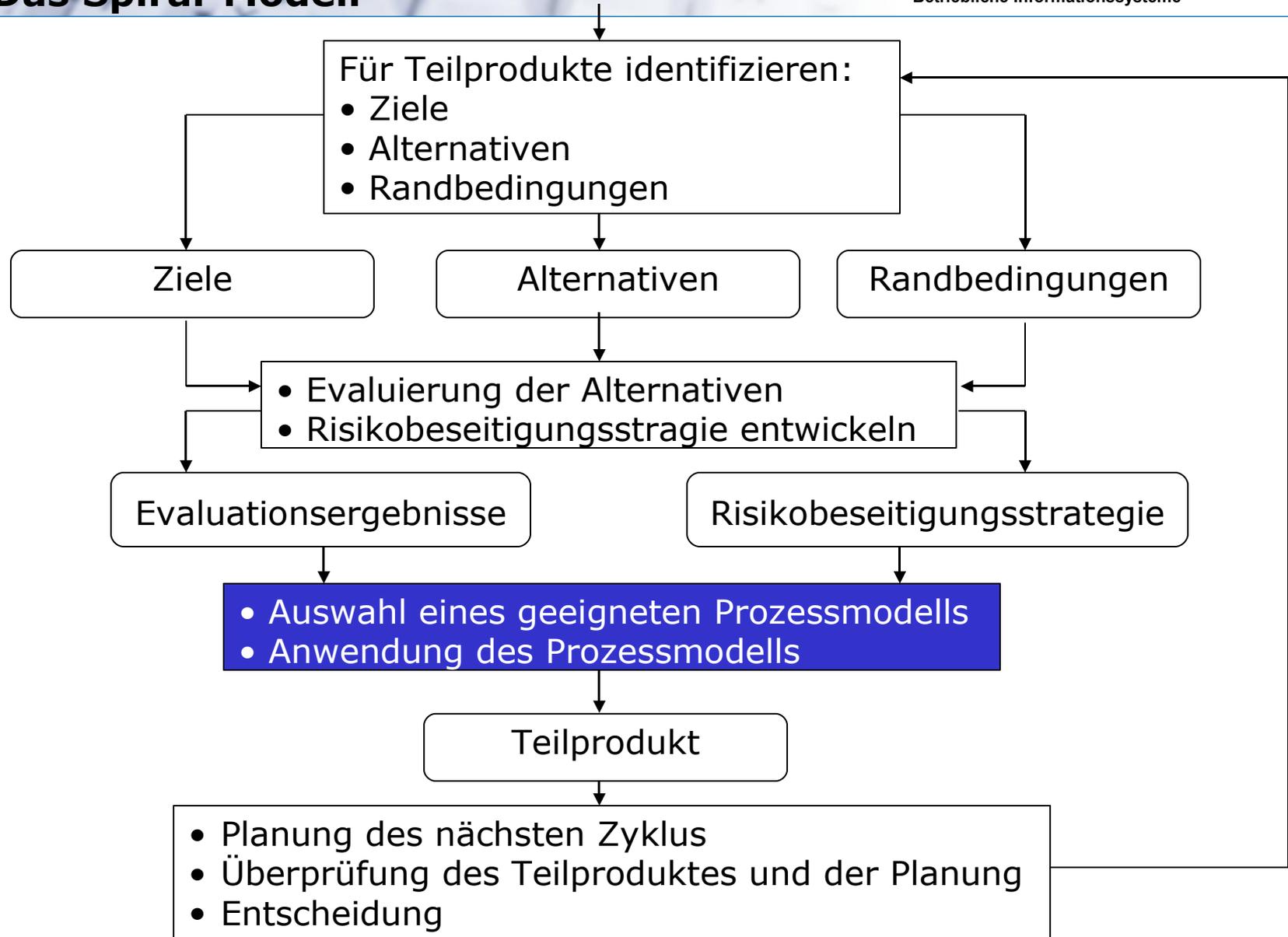
Schritt 4

## 9. Das Spiralmodell

### Das Spiralmodell



## 9. Das Spiral-Modell



## 9. Das Spiral-Modell

### Charakteristika

- Risikogetriebenes Modell, bei dem oberstes Ziel die Minimierung des Risikos ist.
- Jede Spirale stellt einen iterativen Zyklus durch dieselben Schritte dar.
- Die Ziele für jeden Zyklus werden aus den Ergebnisse des letzten Zyklus abgeleitet.
- Separate Spiralzyklen können für verschiedene Software-Komponenten durchgeführt werden.
- Keine Trennung in Entwicklung und Wartung.
- Ziel: Beginne im kleinen, halte die Spirale eng und erreiche die Entwicklungsziele mit minimalen Kosten.
- Bei der Zielbestimmung werden auch Qualitätsziele aufgeführt.
- Für jede Aktivität und jeden Ressourcenverbrauch wird gefragt „wieviel ist genug?“.

# Zusammenfassung

Prozessmodell	Primäres Ziel	Antreibendes Moment	Benutzerbeteiligung	Charakteristika
Wasserfall-Modell	min. Managementaufwand	Dokumente	Gering	Sequentiell, volle Breite
V-Modell	max. Qualität	Dokumente	Gering	Sequentiell, volle Breite, Validation, Verifikation
Prototypen-Modell	Risikominimierung	Code	Hoch	Nur Teilsysteme
Evolutionäres Modell	min. Entwicklungszeit	Code	Mittel	Sofort: nur Kernsystem
Inkrementelles Modell	min. Entwicklungszeit und Risiko	Code	Mittel	Volle Definition dann zunächst nur Kernsystem

# Zusammenfassung

Prozessmodell	Primäres Ziel	Antreibendes Moment	Benutzerbeteiligung	Charakteristika
OO-Modell	Zeit und Kostenminimierung	Wiederverwendbare Komponenten	?	Volle Breite in Abh. Von WV-Komponenten
Nebenläufiges Modell	min. Entwicklungszeit		Hoch	Volle Breite, nebenläufig
Spiralmodell	Risikominimierung		Mittel	Entscheidung pro Zyklus über weiteres Vorgehen

- [Benington 56]  
Benington H.D., Production of large computer programs
- [Boehm 81]  
Boehm B.W., Software Engineering Economics, Prentice Hall
- [Boehm 84]  
Boehm B.W., Verifying and validating Software Requirements and Design Specification, IEEE Software
- [Boehm 86]  
Boehm B.W., A Spiral Model of Software Development and Enhancement, ACM SIGSOFT
- [Boehm 88]  
Boehm B.W., A Spiral Model of Software Development and Enhancement, IEEE
- [Royce 70]  
Royce W.W., Managing the development of large software systems, IEEE
- [Spectrum 91]  
Special Report: Concurrent Engineering, IEEE Spectrum