

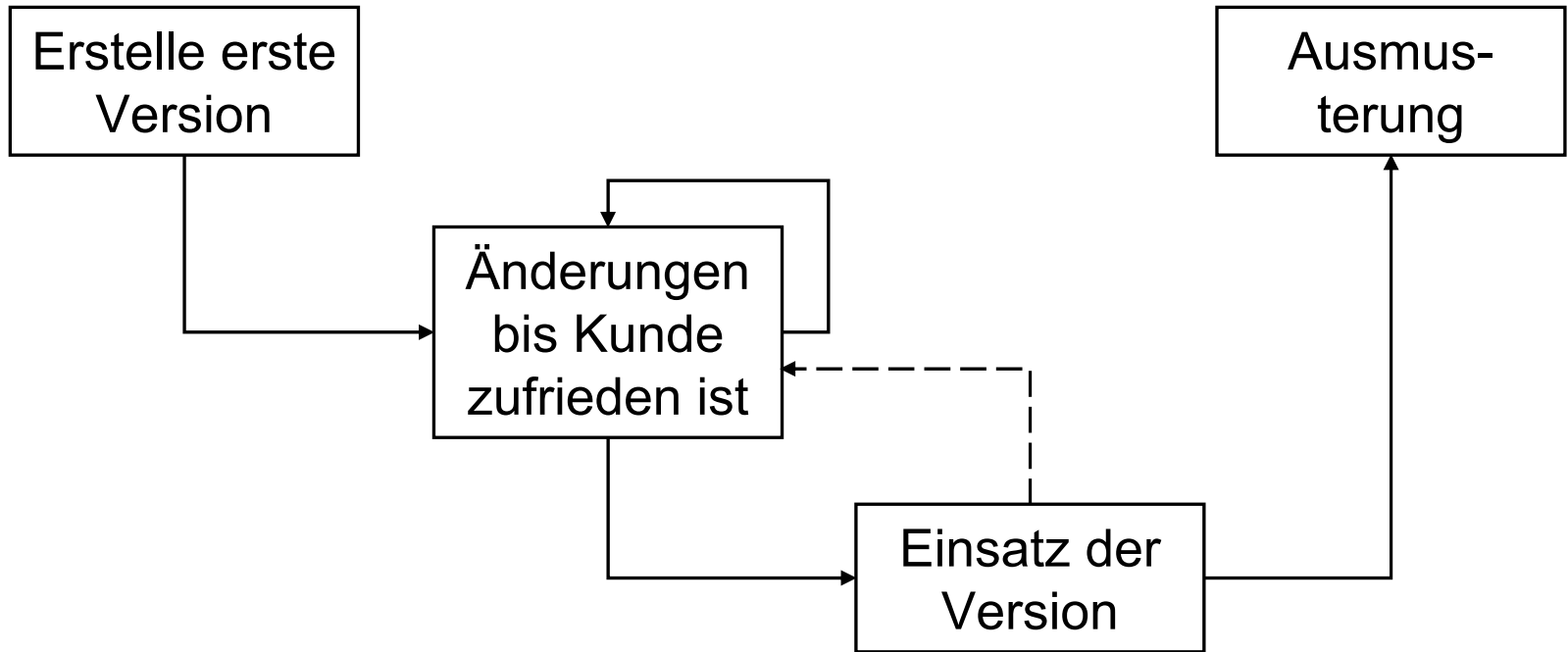
State of the Art

Seminar „Prozesse und Profile“

Gerhard Fuchs

SS 2002

Damals?!? Code & Fix

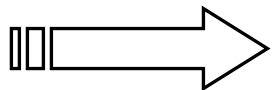


—> Entwicklung

- - -> Verbesserung

Bewertung: Code & Fix

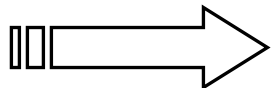
- + Geringer (kein) Managementaufwand
- Schlechtere Zuverlässigkeit, Wartbarkeit und Übersichtlichkeit des Codes
- Starke individuelle Abhängigkeit vom Programmierer
- Differenzen über Funktionsumfang zwischen Entwickler und Anwender
- Keine Entwicklung von Dokumentation und Testfällen



Nur für kleinere Projekte geeignet !

Warum Prozess-Modelle?

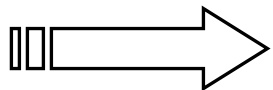
- ❑ Vorhersehbare, überschaubare, planbare und kontrollierbare Gestaltung der Software-Entwicklung
- ❑ Optimierung des Entwicklungsprozesses
- ❑ Zertifizierbarkeit des Entwicklungsprozesses
- ❑ Erhöhung der Prozessqualität



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

Was ist ein Prozess-Modell?

- ❑ 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

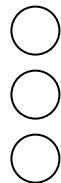


Besseres Projektmanagement möglich!

Welche Prozess-Modelle gibt es?

Klassische Modelle:

- Das Wasserfall-Modell
- Das V-Modell
- Das Prototypen-Modell
- Das evolutionäre Modell
- Das inkrementelle Modell
- Das nebenläufige Modell
- Das Spiral-Modell

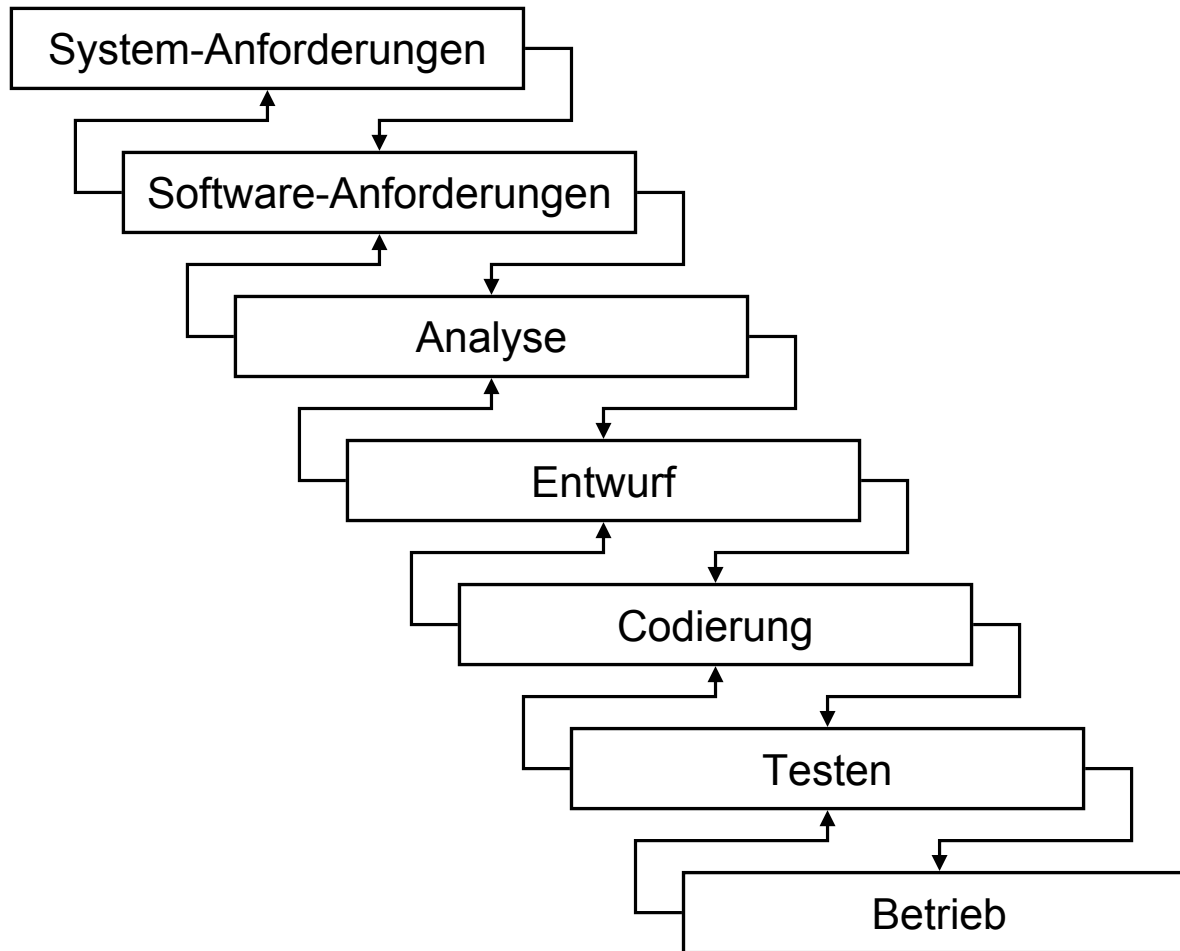


Neue Modelle:

- The Rational Unified Process (RUP)
- The Rapid Object-Oriented Process for Embedded Systems (ROPES)
- Extreme Programming (XP)



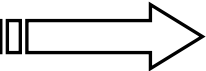
Das Wasserfall-Modell



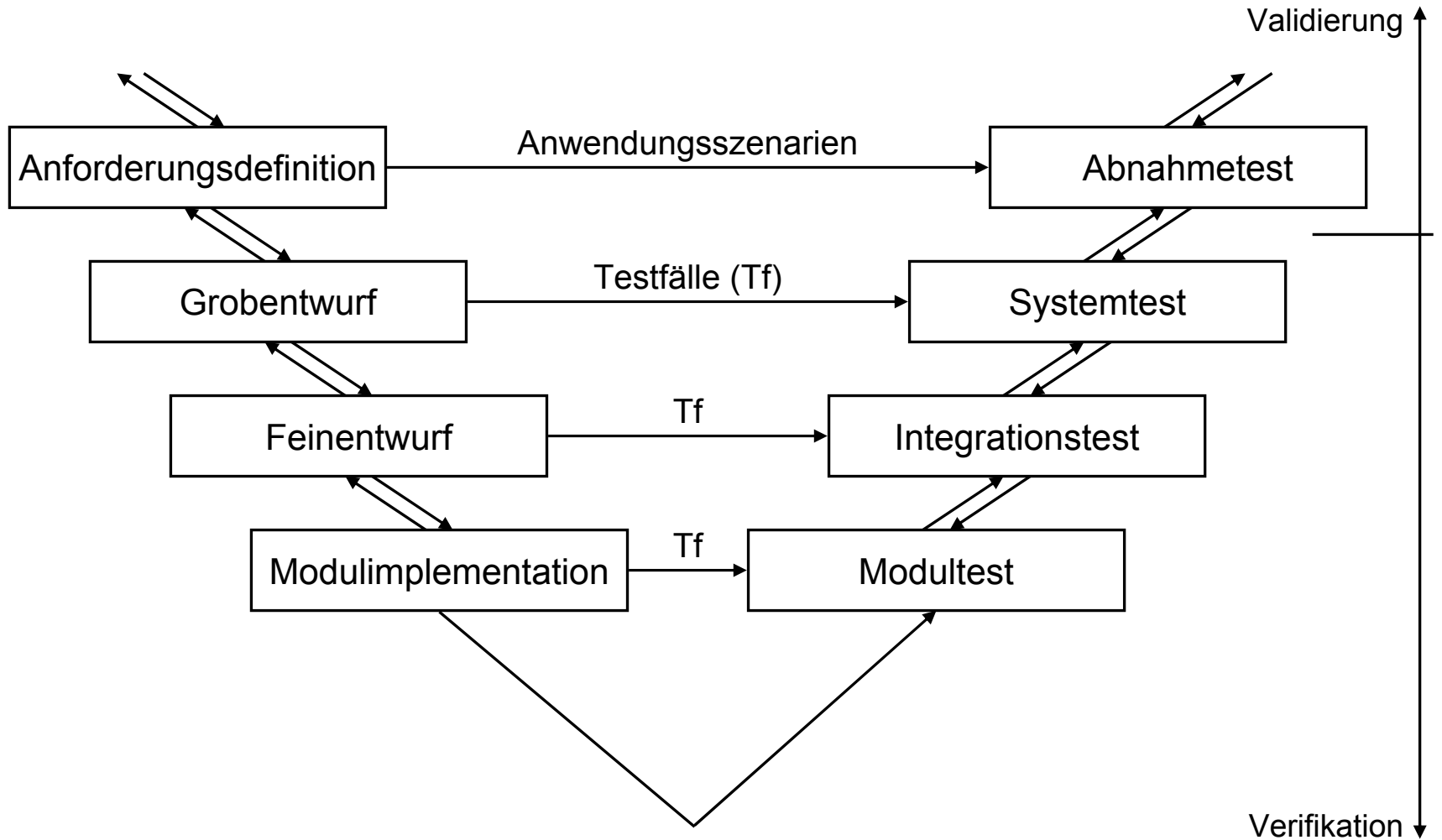
Beschreibung: Wasserfall-Modell

- ❑ 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

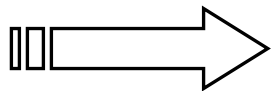


Beschreibung: V-Modell

- ❑ Anspruch auf Allgemeingültigkeit
- ❑ Definition von 25 Rollen für Managementaufgaben
- ❑ Anpassung an konkrete Anforderungen
- ❑ Entwicklungsmodell für ein Gesamtsystem
- ❑ ISO-Standard: Militär- und Bundesbehörden
- ❑ Aufteilung in Sub-Modelle:
 - * Systemerstellung
 - * Qualitätssicherung
 - * Konfigurationsmanagement
 - * Projektmanagement

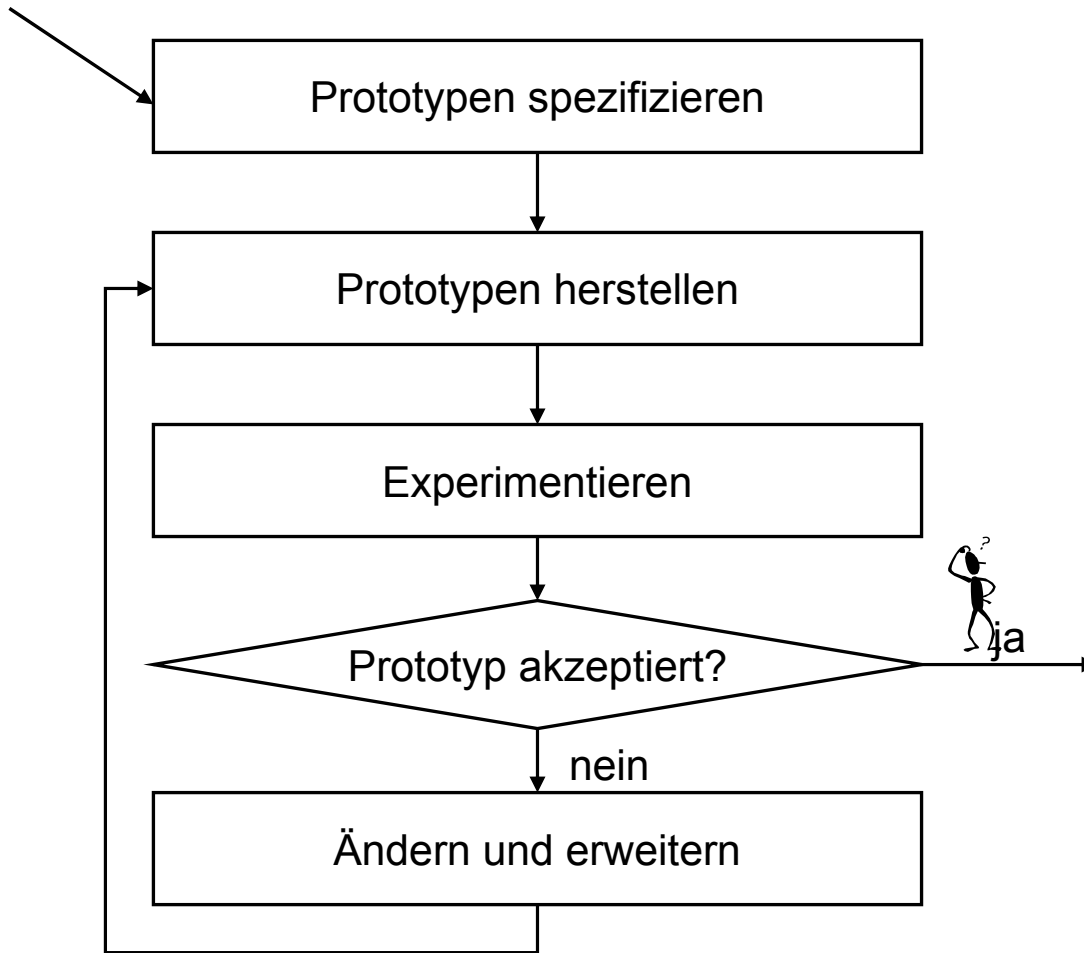
Bewertung: V-Modell

- + Sehr detaillierte Darstellung
- + Anpassung an projektspezifische Anforderungen
- + Integration vieler Aspekte des Entwicklungsprozesses
- + Standardisierung der Abwicklung von Systemerstellungprojekten
- Zu allgemein für kleine und mittlere Software-Modelle
- Sehr bürokratisch
- Keine Handhabung ohne Unterstützung durch CASE-Tools



***Gut geeignet für große Projekte
insb. für eingebettete Systeme!***

Das Prototypen-Modell



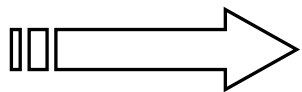
Beschreibung: Prototypen-Modell

- ❑ Ziel: Lösung der folgenden Probleme
 - * Schwierigkeiten bei der vollständigen Definition von Anforderungen
 - * Auswahl alternativer Lösungsmöglichkeiten
 - * Einbeziehen von Anwendern in die Entwicklung
 - * Sicherstellung der Realisierbarkeit
 - * Frühzeitiges Marketing

- ❑ Ansatz: Frühzeitige Erstellung von lauffähigen Prototypen für
 - * Tests und Klärung von Problemen
 - * Reine Produktdefinition, dann Neuentwicklung
 - * Frühe Produktversion mit inkrementeller Weiterentwicklung

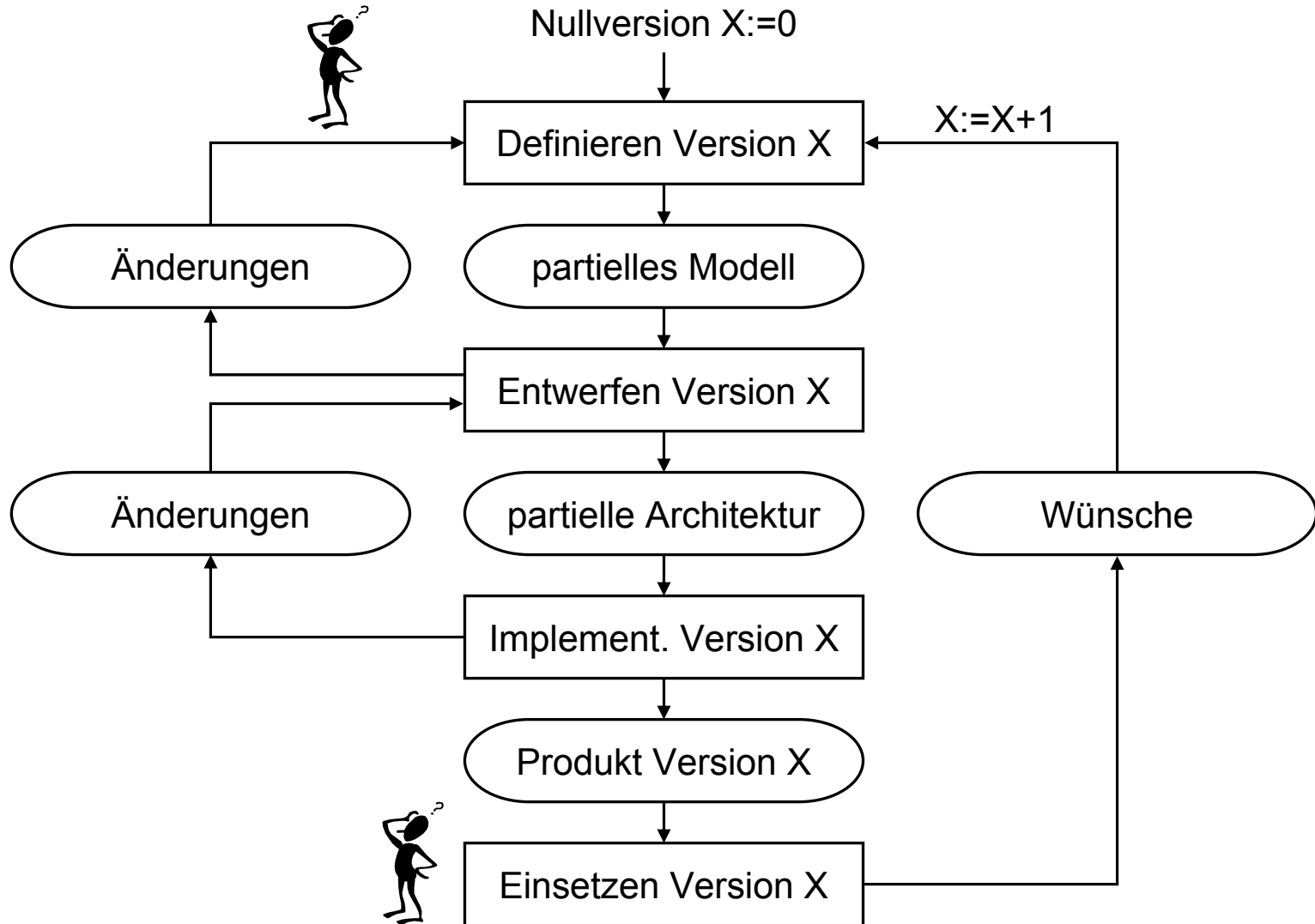
Bewertung: Prototypen-Modell

- + Sinnvolle Integration in andere Prozess-Modelle möglich
- + Schaffung einer starken Rückkopplung zwischen Endbenutzern und Herstellern
- + Schnelle Erstellung von Prototypen durch geeignete Werkzeuge
- Hoher Entwicklungsaufwand durch zusätzliche Herstellung von Prototypen
- Gefahr der Umwandlung eines „Wegwerf-Prototyps“ zu einem Teil des Endprodukts aus Termingründen
- Prototypen ersetzen fehlende Dokumentation



Reduzierung des Entwicklungsrisikos!

Das evolutionäre Modell

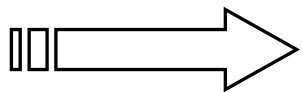


Beschreibung: evolutionäres Modell

- ❑ Vermeidung der Implementierung eines Produkts in voller Breite
- ❑ Kernanforderungen des Auftraggebers führen direkt zur „Nullversion“ (wird ausgeliefert)
- ❑ Neue Anforderungen → neue Version
- ❑ Stufenweise Entwicklung durch Modell, Steuerung durch die Erfahrung der Benutzer
- ❑ Integration von Pflegeaktivitäten in den Prozess
- ❑ Konzentration auf lauffähige Teilprodukte

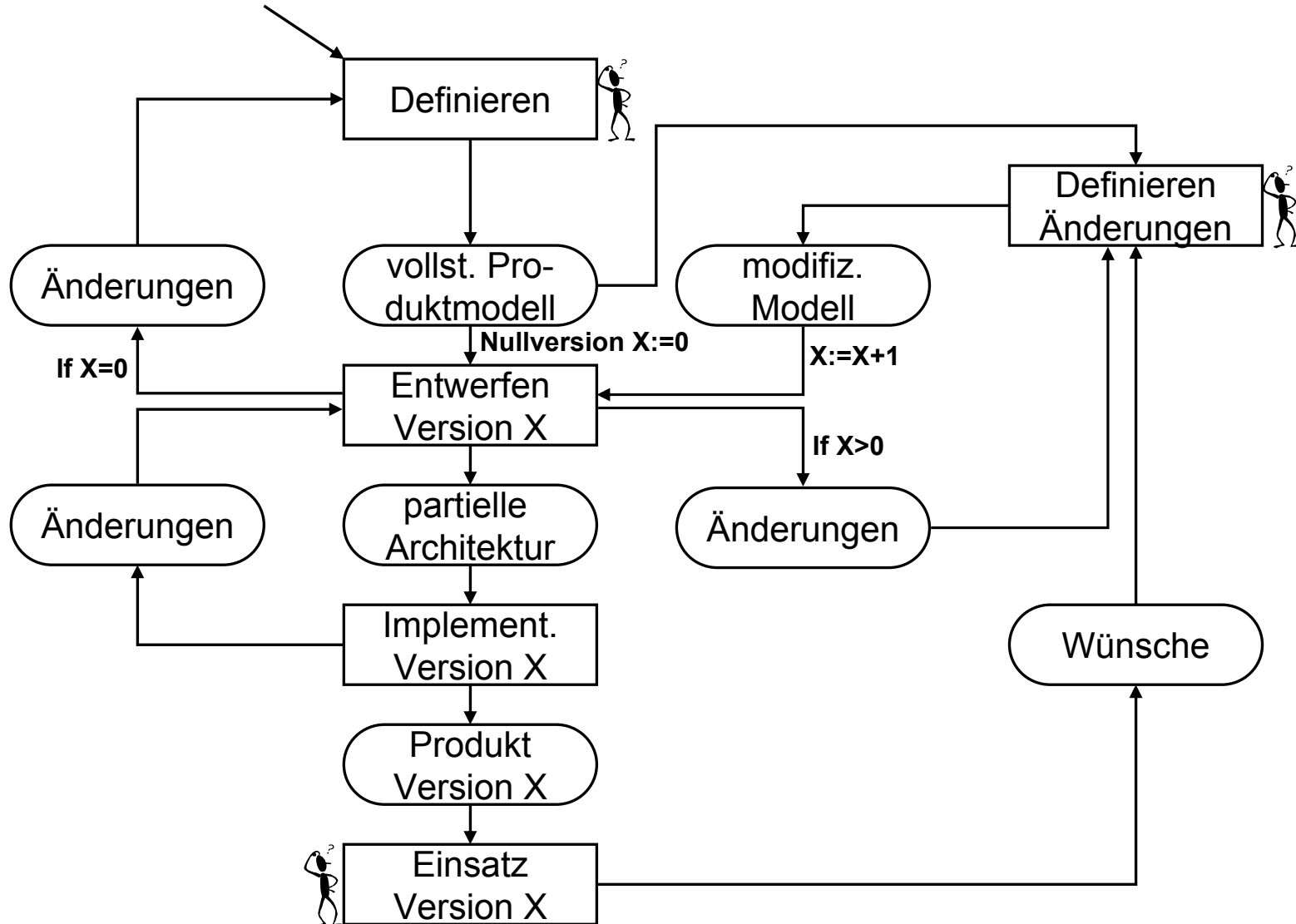
Bewertung: evolutionäres Modell

- + Einsatzfähige Produkte für den Auftraggeber in kurzen Abständen
- + Integration der Erfahrungen der Anwender in die Entwicklung
- + Überschaubare Projektgröße
- + Korrigierbare Entwicklungsrichtung
- + Keine Ausrichtung auf einen einzigen Endtermin
- Eventuell komplette Änderung der Architektur in späteren Versionen
- Eventuell mangelnde Flexibilität der Nullversion zur Anpassung an unvorhersehbare Evolutionspfade



Gut geeignet, wenn der Auftraggeber seine Anforderungen noch nicht vollständig überblickt!

Das inkrementelle Modell

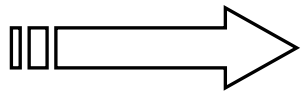


Beschreibung: inkrementelles Modell

- ❑ Möglichst vollständige Erfassung und Modellierung der Anforderungen an das zu entwickelnde Produkt
- ❑ Einsatzfähiges Modell für den Auftraggeber schon nach kurzer Zeit
- ❑ Realisierung der nächsten Ausbaustufe unter Berücksichtigung der Erfahrungen des Auftraggebers mit der laufenden Version

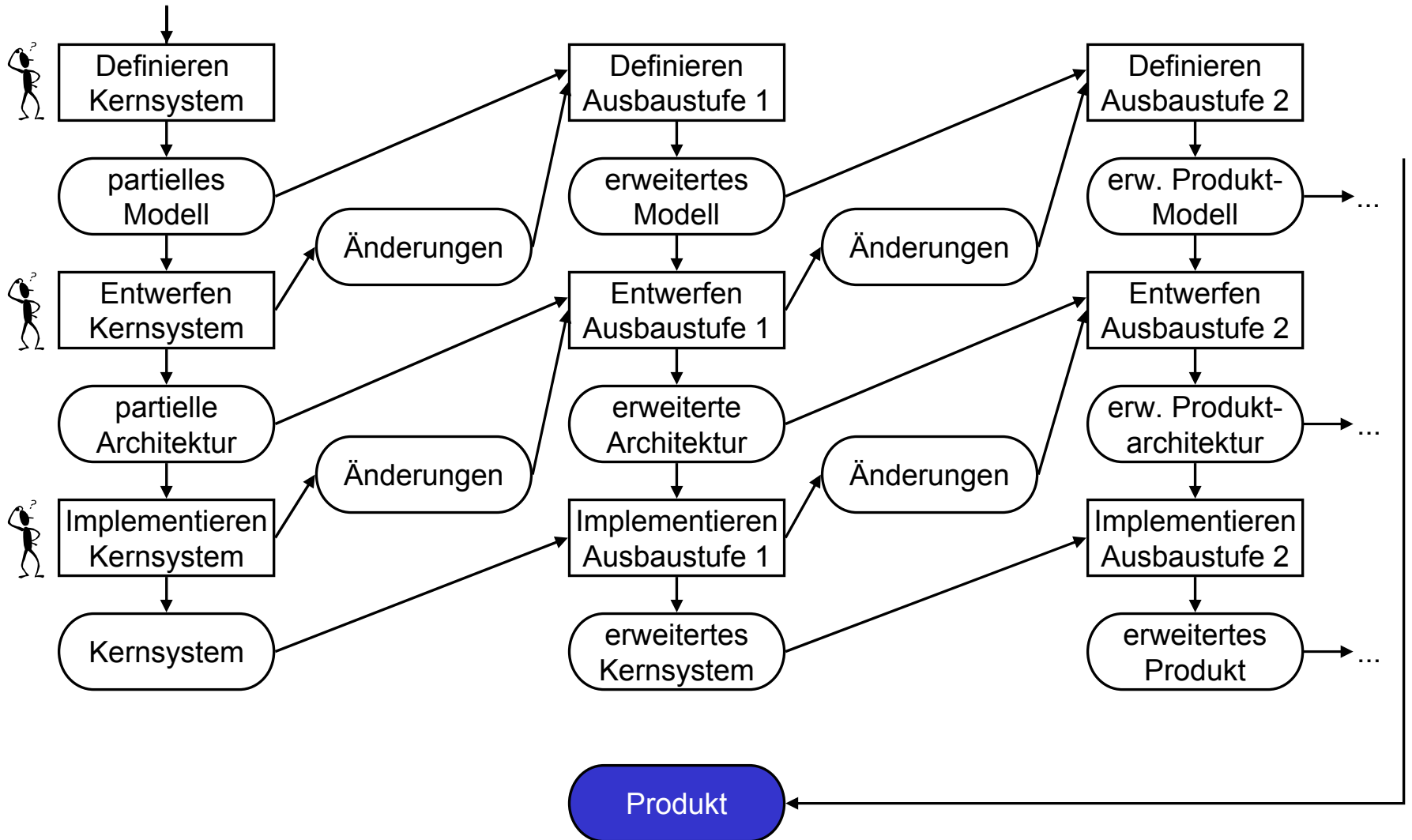
Bewertung: inkrementelles Modell

- + Inkrementelle Erweiterungen passend zum bisherigen System
- + Entwicklung im Umfeld von Rational und passt hervorragend zu OO-Techniken
- Vollständige Spezifikation nicht immer möglich
- Starke Verknüpfung mit objektorientiertem Paradigma (Übertragung auf andere Paradigmen schwierig)



Berücksichtigt Nachteile des evolutionären Modells!

Das nebenläufige Modell

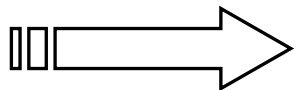


Beschreibung: nebenläufiges Modell

- ❑ Parallelisierung von sequentiell organisierten Vorgängen
- ❑ Minimierung des Improvisierens und „trial and errors“
- ❑ Förderung der Zusammenarbeit der einzelnen Personengruppen
- ❑ Reduzierung von Wartezeiten und Zeitverzögerungen

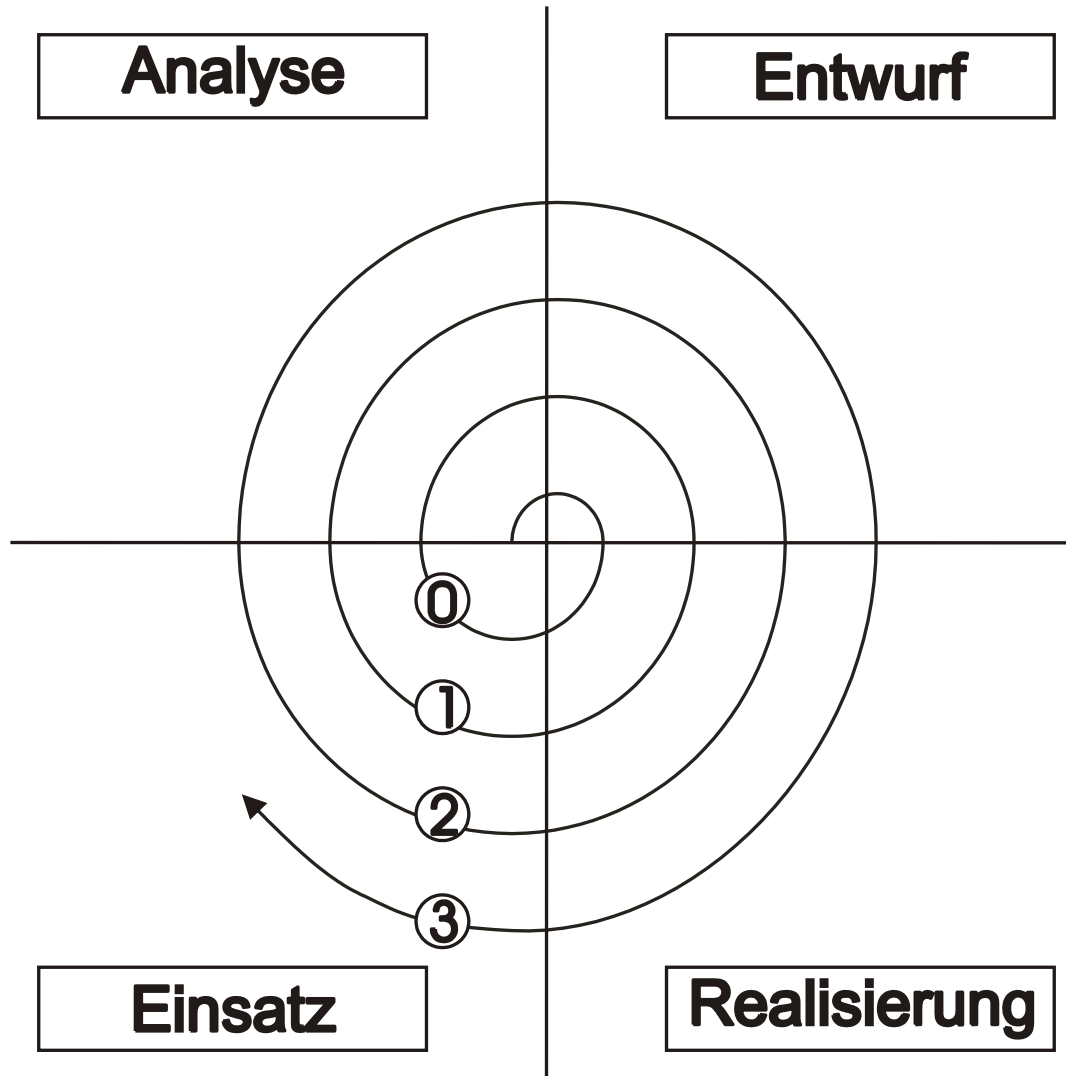
Bewertung: nebenläufiges Modell

- + Optimale Zeitausnutzung
- + Frühes Erkennen und Vermeiden von Problemen durch Beteiligung aller betroffenen Personengruppen
- Ehrgeiziges Ziel: „Right the first time“
- Risiko, grundlegende Entscheidungen zu spät zu treffen
- Hoher Personal- und Planungsaufwand



Ziel: Auslieferung des vollständigen Produktes!

Das Spiral-Modell

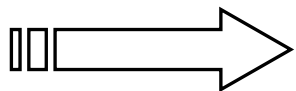


Beschreibung: Spiral-Modell

- ❑ Meta-Modell
- ❑ Risikominimierung als oberstes Ziel
- ❑ Keine Trennung von Entwicklung und Wartung
- ❑ 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 Prozess-
ablaufs
- + Keine Festlegung auf ein
Prozessmodell
- + Frühzeitige Eliminierung von
ungeeigneten Alternativen
und Fehlern
- Hoher Managementaufwand
(für kleine und mittlere Pro-
jekte weniger gut geeignet)



Sehr flexibles Modell durch Betrachtung von Alternativen!

Zusammenfassung:

Prozess-Modell	Primäres Ziel	Antreibendes Moment	Benutzerbeteiligung	Charakteristika
Wasserfall-Modell	minimaler Managementaufwand	Dokumente	gering	sequentiell, volle Breite
V-Modell	maximale Qualität (<i>safe-to-market</i>)	Dokumente	gering	sequentiell, volle Breite, Validation, Verifikation
Prototypen-Modell	Risikominimierung	Code	hoch	nur Teilsysteme
Evolutionäres Modell	minimale Entwicklungszeit (<i>fast-to-market</i>)	Code	mittel	nur Kernsystem
Inkrementelles Modell	minimale Entwicklungszeit Risikomimimierung	Code	mittel	volle Definition, dann zunächst nur Kernsystem
Nebenläufiges Modell	minimale Entwicklungszeit	Zeit	hoch	volle Breite, nebenläufig
Spiral-Modell	Risikominimierung	Risiko	mittel	Entscheidung pro Zyklus über weiteres Vorgehen

Einsatz der Prozessmodelle:

- ❑ Wasserfallmodell: bei Managern sehr beliebt
- ❑ V-Modell : offizieller Standard sowohl im öffentlichen als auch im militärischen Bereich (Erfolg → Banken, Versicherungen, Automobilindustrie übernehmen dieses Modell)
- ❑ nebenläufiges Modell: bisher wenig Erfahrung
- ❑ Spiralmodell: Problem, dass Wissen über das Managen und Identifizieren von Risiken noch nicht weit verbreitet ist
- ❑ RUP ist im Kommen
- ❑ Viele Entwickler sympathisieren mit XP (schlank, entwicklungsorientiert)
- ❑ Es existieren Versuche, XP und RUP zusammenzuführen

Fazit:

- ❑ Prozesse helfen bei der Software-Entwicklung
- ❑ Prozessmodelle können gemischt werden
- ❑ Jeweilige Situation und Problemstellung erfordert angepasstes Prozessmodell

- ❑ Aber:
 - * Prozessmodelle sind kein Dogma!
 - * Diskrepanz zwischen Theorie und Praxis
 - * Keine Lösung des Problem Unformales → Formales
 - * Keine Beseitigung der „essentiellen Komplexität“

- ❑ Probleme:
 - * Mangelnde Akzeptanz wechselnder Methoden
 - * Mehrarbeit, wenn schlechte oder keine Werkzeuge

Literatur / Links:

- ❑ Balzert, H.; Lehrbuch der Software-Technik, Software-Entwicklung; Spektrum, Akad. Verl.; Heidelberg [u.a.]; 1996

- ❑ http://www.ipt.ch/deutsch/projekte_ref/aktuell.html
- ❑ http://www.ifi.unizh.ch/groups/req/ftp/se_l/kapitel_3.pdf
- ❑ <http://www.andrena.de/Links/Paper3.pdf>
- ❑ <http://graphics.cs.uni-sb.de/Courses/ss01/sw/sw/ProzessModelle.pdf>
- ❑ <http://www.dfki.uni-kl.de/~klink/Postscript/seminar.ps.gz>
- ❑ <http://www.cosy.sbg.ac.at/~aichhorn/prprakt/node6.html>
- ❑ <http://vislab.informatik.fh-kl.de/lectures/pm/prozesse.pdf>
- ❑ <http://www8.informatik.uni-erlangen.de/IMMD8/Lectures/WEB/vorlesung/sources/14112001/Vorlesung4.pdf>
- ❑ <http://www.uni-koblenz.de/~ist/lehre/WS0102/ST4IM/gliederung.html>
- ❑ http://www-is.informatik.uni-oldenburg.de/~sauer/lehre/swp2_01/vl_2.pdf
- ❑ http://www.n-consulting.ch/n-c%20vorlesungen/Prozess_Modelle.pdf
- ❑ <http://www.informatik.uni-freiburg.de/~amarkert/seminar/Vorgehensmodelle.htm>
- ❑ <http://www.cs.colorado.edu/users/kena/classes/5828/s01/guestlectures/SLC-guest.pdf>
- ❑ http://lglwww.epfl.ch/teaching/software_project/documentation/introduction/softwareengineeringwhywhat.pdf