Einführung in die Softwaretechnik
9. Softwareprozesse

Klaus Ostermann

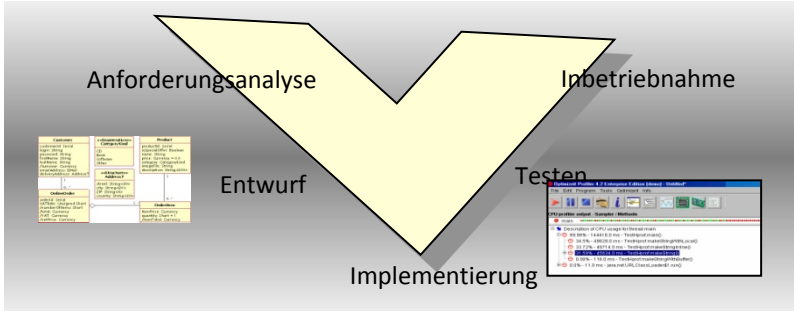
(Mit Folien von Christian Kästner, Gabriele Taentzer und Wolfgang Hesse)

Agenda

- Wie kommt man vom Kundenwunsch zur fertigen Software?
- Wie strukturiert man ein Softwareprojekt?
- Welche Arbeiten müssen koordiniert werden?
- Wann sollte getestet werden?
- Welche Vorgehensmodelle gibt es für die Softwareentwicklung?

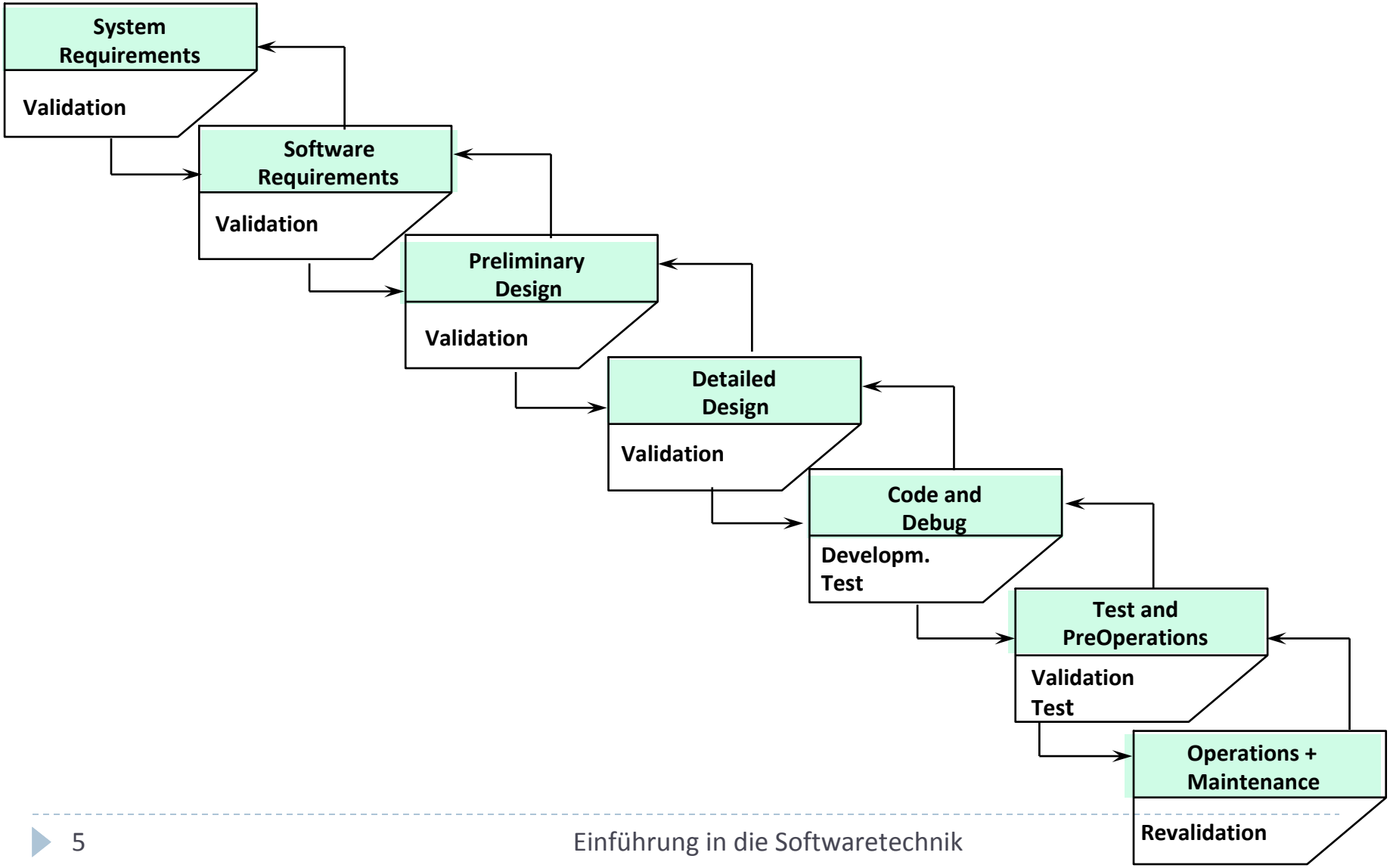
Softwareentwicklungsprozesse

- Wasserfallmodell
- V-Modell
- Spiralmodell
- Inkrementelle Entwicklung
- Unified Process
- Agile Softwareentwicklung
  - Extreme Programming



Sequenzielle Modelle

Wasserfallmodell von B. BOEHM

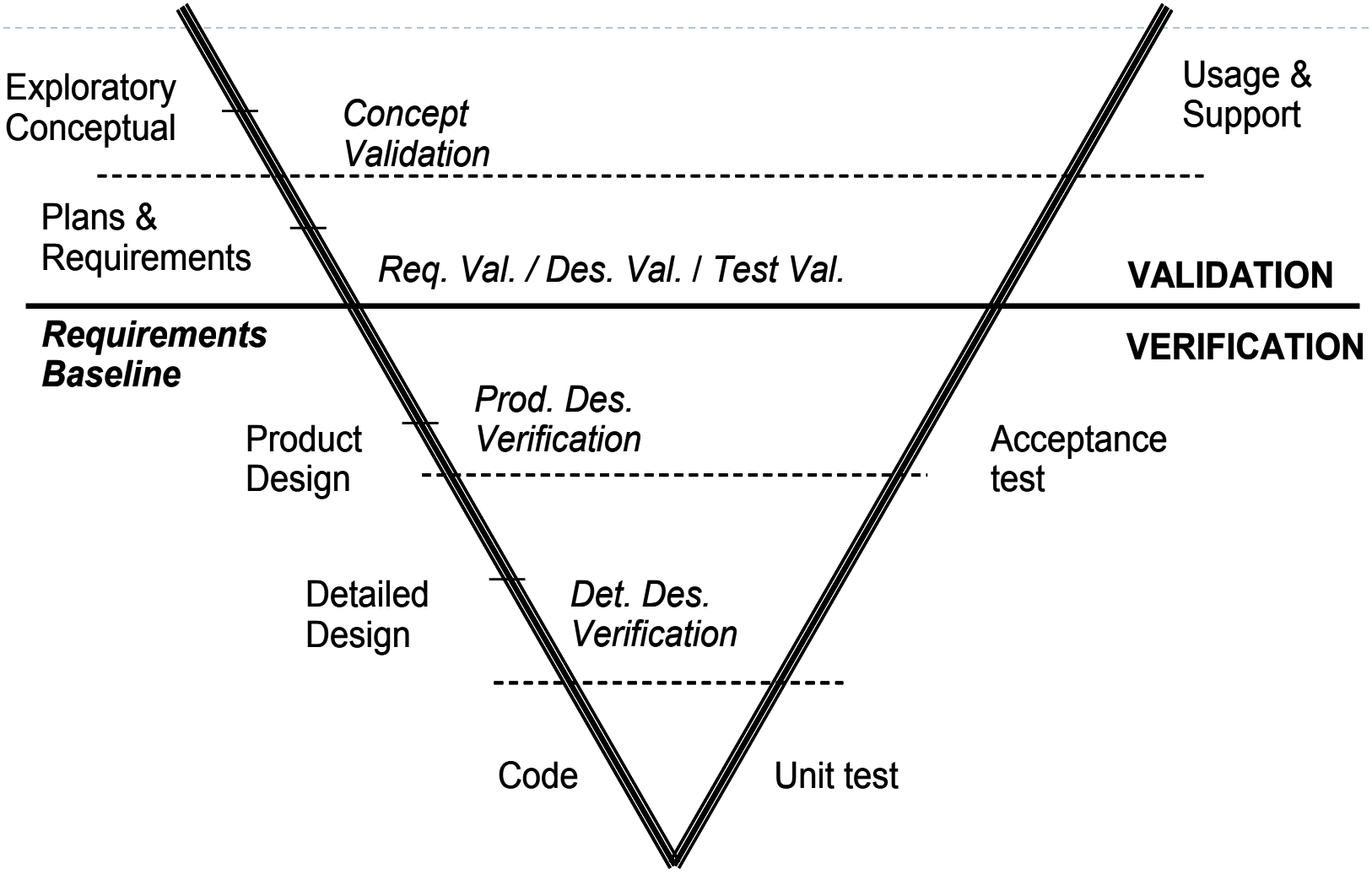


Diskussion Wasserfallmodell

- Dokumentation nach jeder Phase verfügbar
- Klare Trennung der Phasen und Verantwortlichkeiten
- Analog zu Ingenieursprojekten (Brückenbau etc)

- Starres Vorgehen
- Reaktionen auf geänderte Anforderungen schwierig
- Anforderungen, Design, etc früh fixiert, Änderungen nicht vorgesehen
- Späte Qualitätsprüfung

V-Modell


Exploratory
Conceptual
Concept
Validation
Usage &
Support
Plans &
Requirements
Req. Val. / Des. Val. / Test Val.
VALIDATION
Requirements
Baseline
VERIFICATION
Prod. Des.
Verification
Product
Design
Acceptance
test
Detailed
Design
Det. Des.
Verification
Code
Unit test

Iterative Modelle

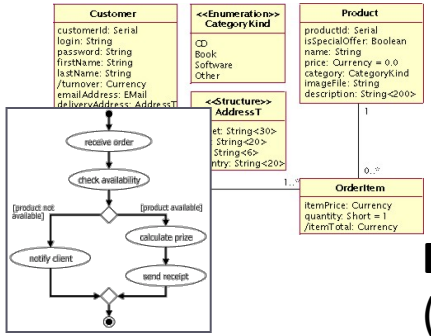
Iterative Softwareentwicklung

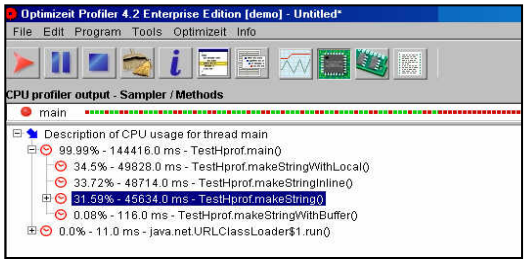
Vorbereitung

Nachbereitung

Analyse
(Anforderungsmodell)

Test
(Evaluierung)





Design
(Lösungsmodell)

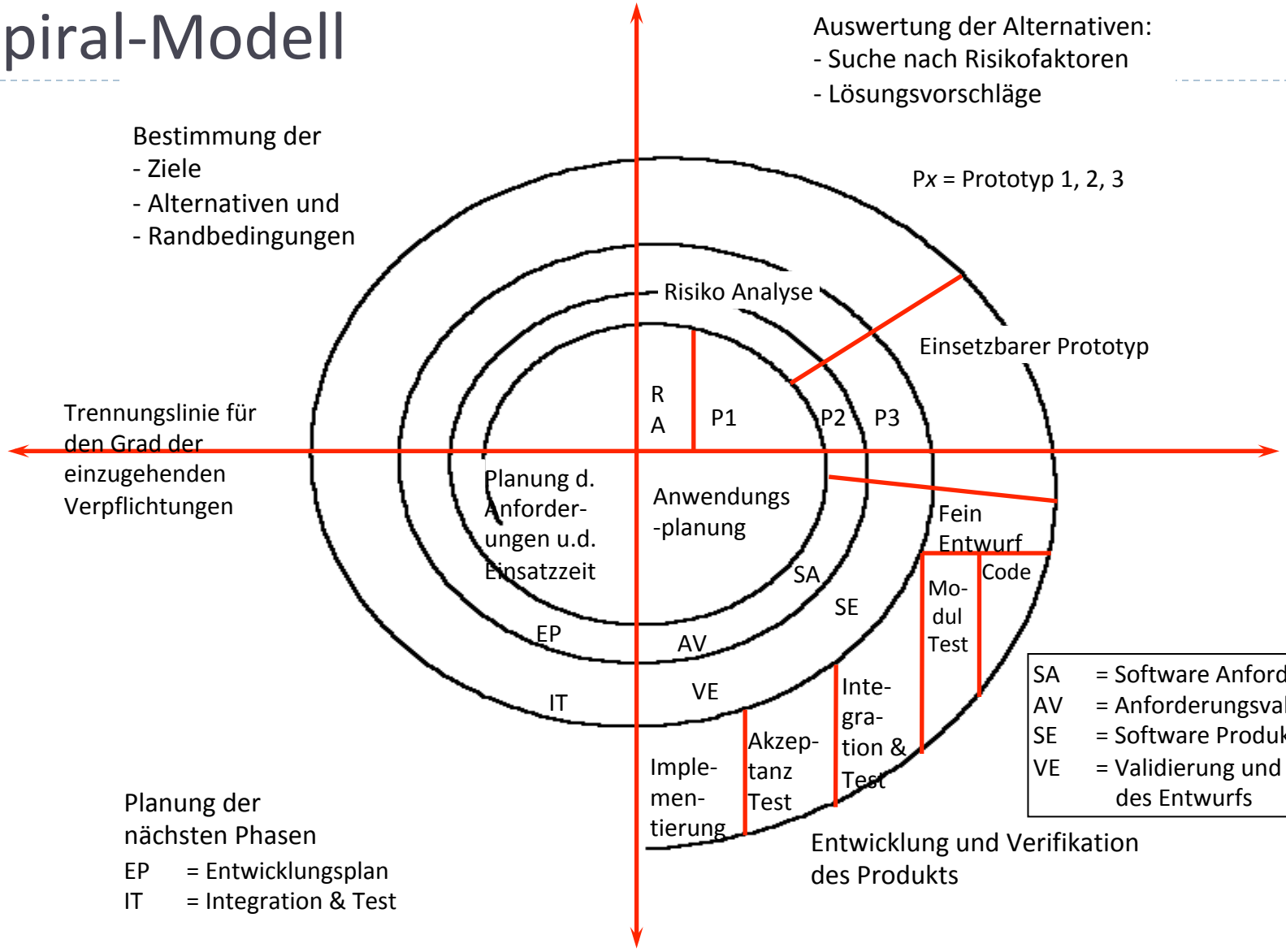
Implementierung
(Lösung)

Prototypen

- Möglichst früh eine erste Version erstellen
- Insbesondere bei Unklarheiten bei Kundenwünschen
- Selbst wenn Kernfunktionalität fehlt
- Kunde kann Fortschritt erkennen und (Teil-)Lösung bewerten

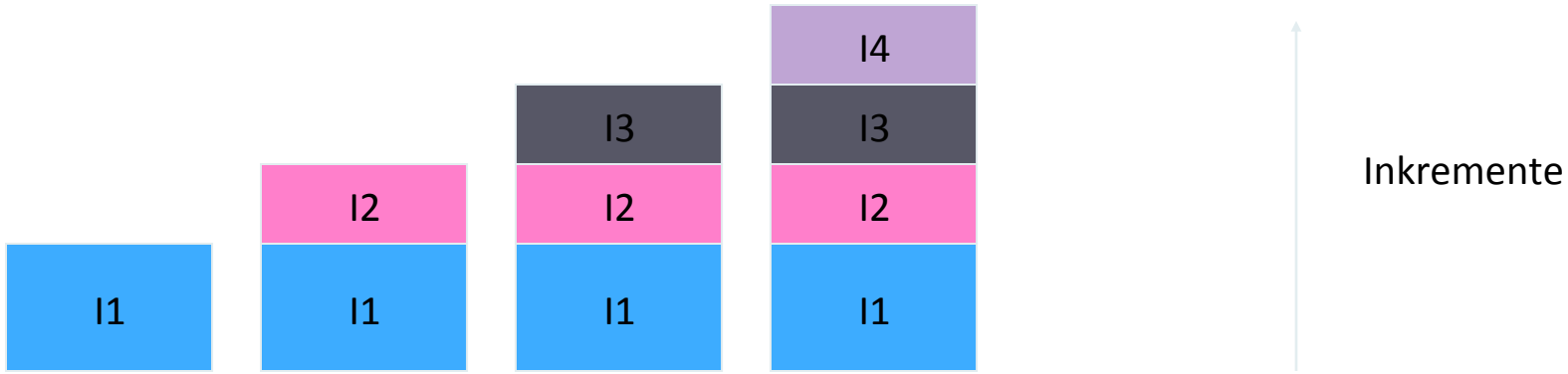
- Wegwerf-Prototypen vs. Inkrementelle Entwicklung

Spiral-Modell

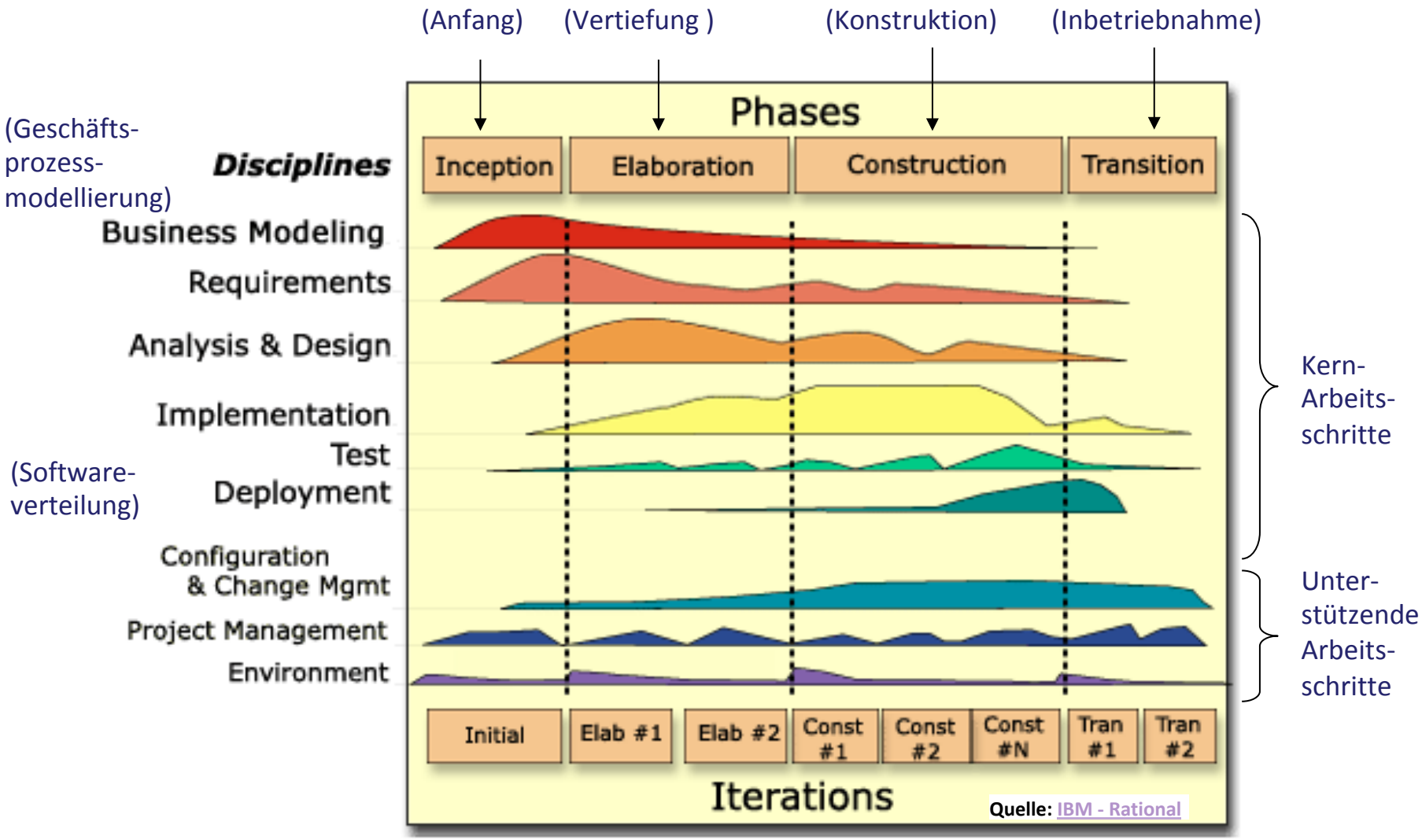


Inkrementelle Systementwicklung (Stepwise Refinement)

- Inkrement: Software-Baustein(e), die zu einem existierenden System oder Subsystem hinzugefügt werden, um dessen Funktionalität oder Leistung zu vergrößern oder zu verändern
- Inkrement kann Subsystem entsprechen
- Inkrementelle Systementwicklung: beginn mit Kernsystem, schrittweise Erweiterung



Unified Process



Prozessphasen

- Anfang (inception): grundlegender Umfang des Projekts ist bekannt, endet mit der Zusage des Auftraggebenden
- Vertiefung (elaboration): Endet mit
  - Grundlegender Systemarchitektur
  - Konstruktionsplan
  - Identifizierten Risiken
- Konstruktion (construction): iterativ, endet mit Beta-Release
- Inbetriebnahme (transition): Einführung des Systems beim Anwender

Unified Process

- Jede Iteration führt zu einem ausführbaren Gesamtsystem. Neue Anforderungen können nach einer Iterationsstufe noch berücksichtigt werden.
- Die Phasen sind zeitlich geordnet. Die Iterationen beschreiben verschiedene Abschnitte innerhalb einer Phase.
- Die Disziplinen beschreiben die Aktivitäten im Prozess.
- Der Prozess ist anwendungsfallgetrieben. Wichtige Anwendungsfälle werden zuerst behandelt.
- Parallel zur UML von Ivar Jacobson, Grady Booch und James Rumbaugh entwickelt
- Rational UP als Implementierung (aktuell: Ver. 9)

Agile Softwareentwicklung

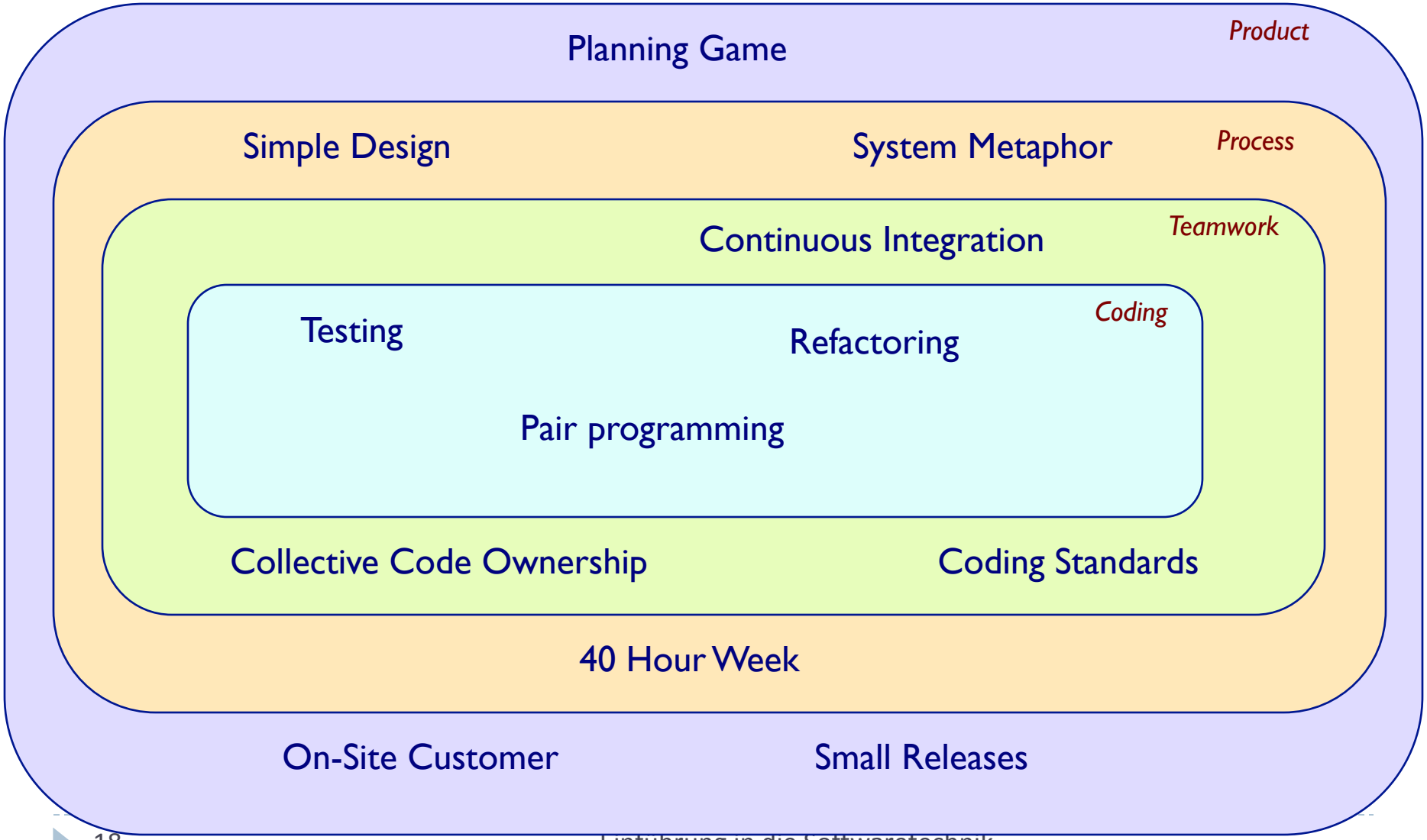
Agile Softwareentwicklung

- … ist ein relativ neuer Ansatz
- im Spannungsfeld zwischen Qualität, Kosten und Zeit
- in ungenauen Kundenwünschen und instabilen Anforderungen,
- in langen Entwicklungszeiten und überzogenen Terminen,
- in unzureichender Qualität.

Manifest der agilen Softwareentwicklung:

- Menschen und Kooperation vor Werkzeugen und (automatisierten) Prozessen,
- funktionsfähige Software vor umfassender Dokumentation
- Zusammenarbeit mit Kunden vor bürokratischen Vertragsverhandlungen
- dynamische Reaktion auf Veränderungen vor statischer Planeinhaltung

Extreme Programming - Overview



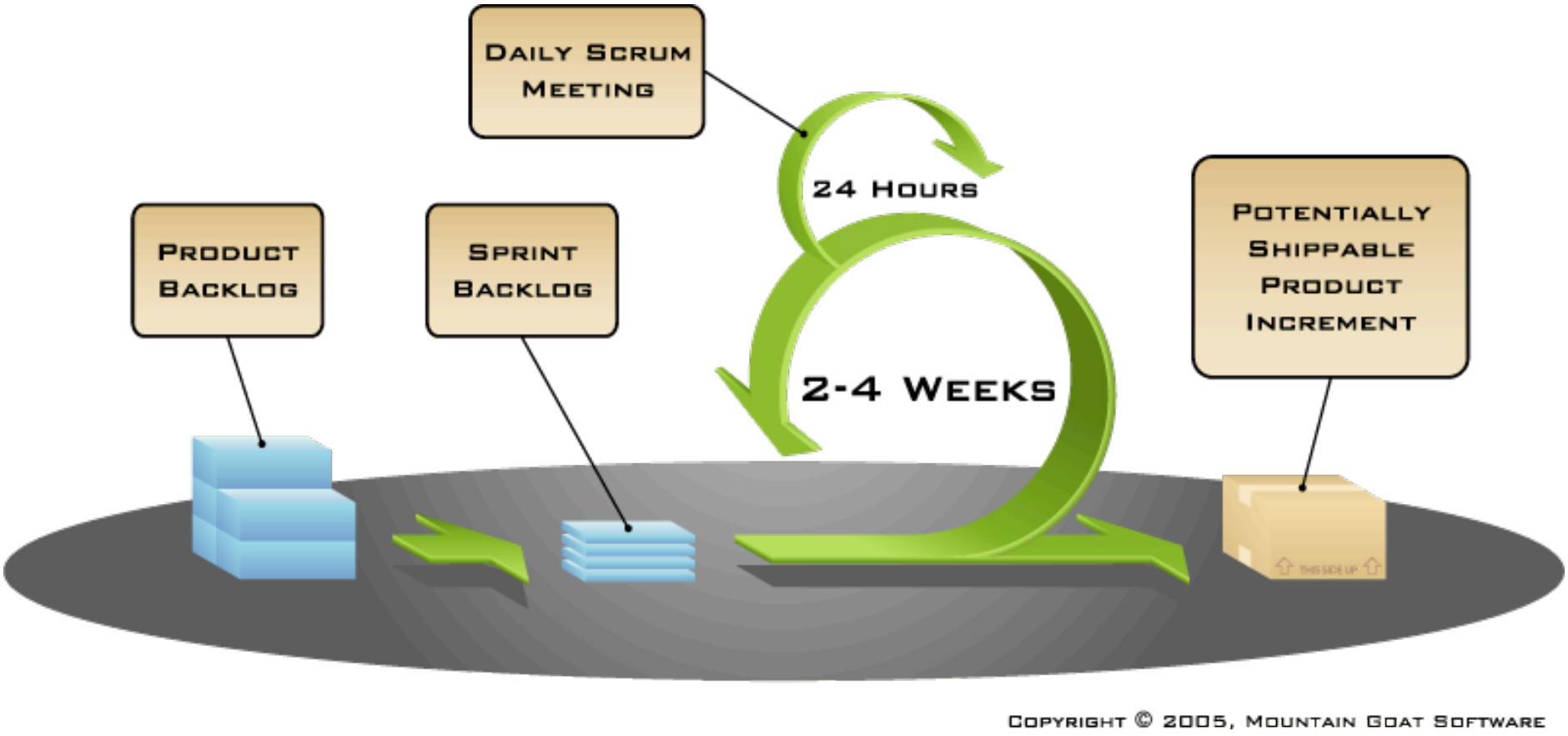
Extreme Programming

"Do the simplest thing that could possibly work"

- iterative Entwicklung (getrieben durch neue oder geänderte Features)
- Wenig Analyse- und Entwurfstätigkeiten, nur rudimentäre Spezifikationen, selbst-dokumentierender Code ("simple design")
- Frühes Programmieren, prototypisches Umsetzen einzelner "stories"
- Testfälle stehen am Anfang und ersetzen Spezifikation ("test first")
- Ständige Kommunikation der Entwickler mit Management und Benutzern, kurze Rückkopplungsschleifen, schnelle Rückmeldungen
- Schrittweise Änderungen, schrittweise angepasste Tests ("refactoring"), fortlaufende Integration ("continuous integration")
- Fahrer-/Beifahrer-Prinzip beim Programmieren ("Pair programming"); schnelle Code Reviews
- Gemeinsame Standards aller Entwickler, gemeinsames Eigentum am Code ("collective code ownership")

Einige dieser Grundsätze sind problematisch und umstritten!

SCRUM


Daily Scrum Meeting
24 Hours
Product Backlog
Sprint Backlog
2-4 Weeks
Potentially Shippable Product Increment
Copyright © 2005, Mountain Goat Software

Zusammenfassung

- kontinuierliche Entwicklung des Softwareentwicklungs-modells vom Wasserfallmodell zu iterativer Entwicklung
- Trend zu
  - ...kürzeren Entwicklungszyklen
  - ...weniger Bürokratie
  - ...systematischerer Qualitätssicherung (Testen)

- Weiterführende Literatur:
  - *eXtreme Programming Explained: Embrace Change*. Kent Beck. Addison-Wesley Pub Co; ISBN: 0201616416; 1st edition (October 5, 1999)