Teil III der Vorlesung Objektorientierte Analyse (OOA) 30) Überblick über die OOA

Prof. Dr. Uwe Aßmann
Institut für Software- und Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
TU Dresden
Version 13-1.0, 03.06.13

Obligatorische Literatur

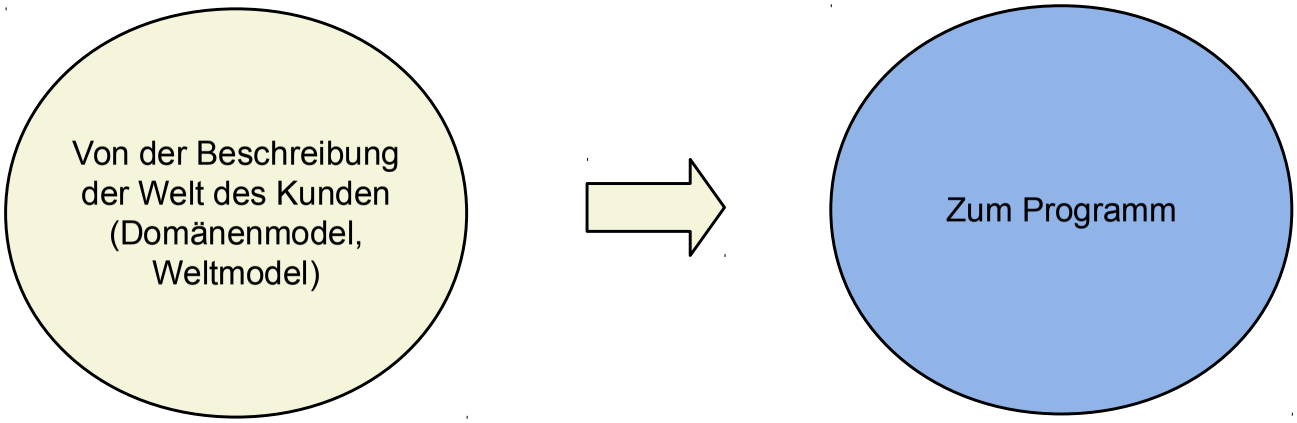
- Zuser, Kap. 7-9
- Störrle, Kap. 5

- Manfred Broy und Andreas Rausch. Das neue V-Modell® XT. Ein anpassbares Modell für Software und System Engineering. Informatik-Spektrum. Springer Berlin / Heidelberg. Volume 28, Number 3 / June, 2005, Pages 220-229 http://www.springerlink.com/content/l173638386334305/

30.1 Überblick über die Objektorientierte Analyse

Die zentralen Frage der Softwaretechnologie

Wie kommen wir vom Problem des Kunden zum Programm (oder Produkt)?



Die zentralen Fragen des objektorientierten Ansatzes

Wie kommen wir vom Problem des Kunden zum Programm (oder Produkt)?

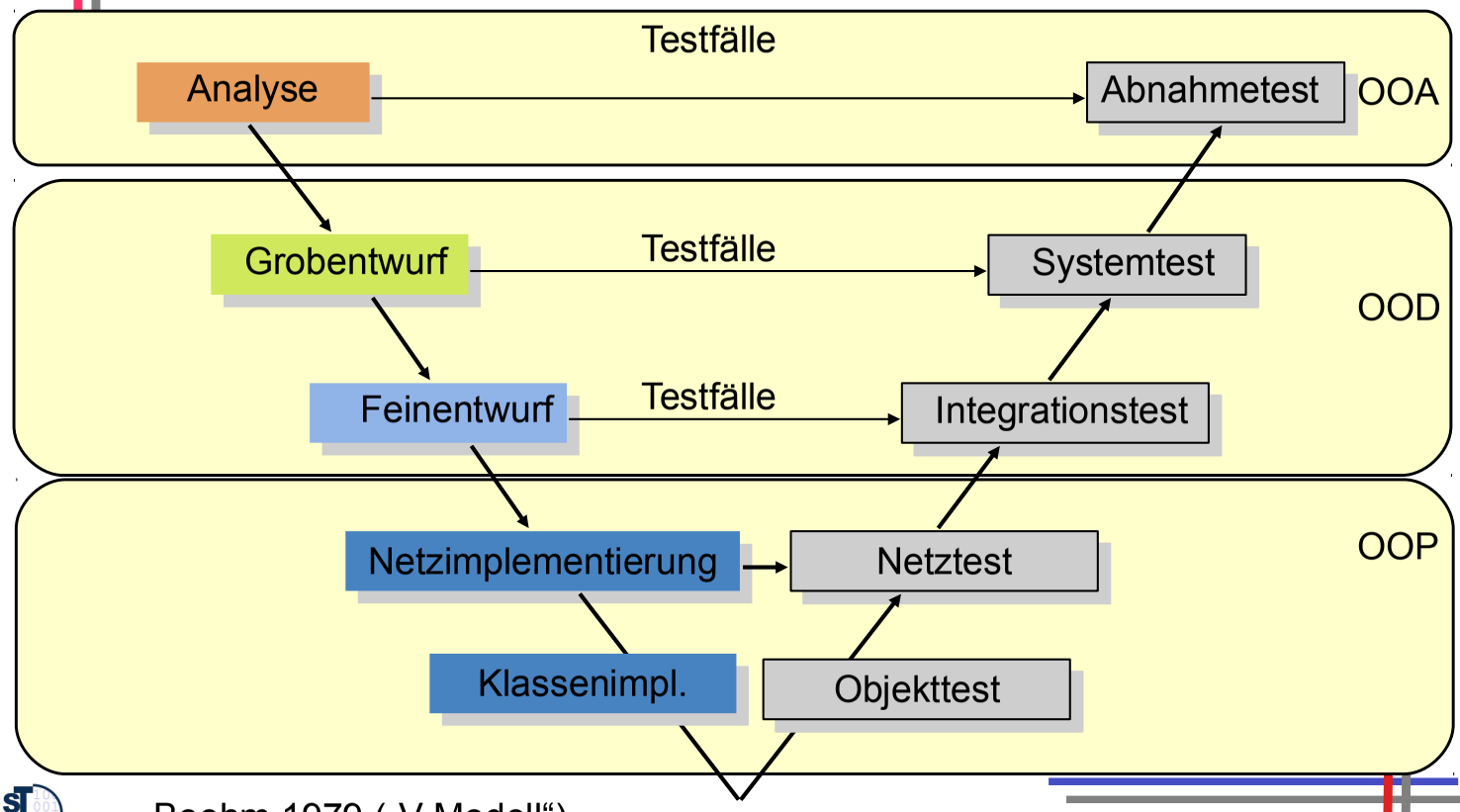
Von der Beschreibung **der Objekte** der Welt des Kunden (objektorientiertes Domänenmodel)

Domänenmodell-Anreicherung
Domänenobjekt-Anreicherung

Zum objektorientierten Programm, das die Objekte der Welt des Kunden **um Programminformation anreichert**

Anreicherung/Verfettung: Anreicherung durch technische Programminformation
„object fattening“: Anreicherung von Objekten des Domänenmodells

Softwareentwicklung im V-Modell

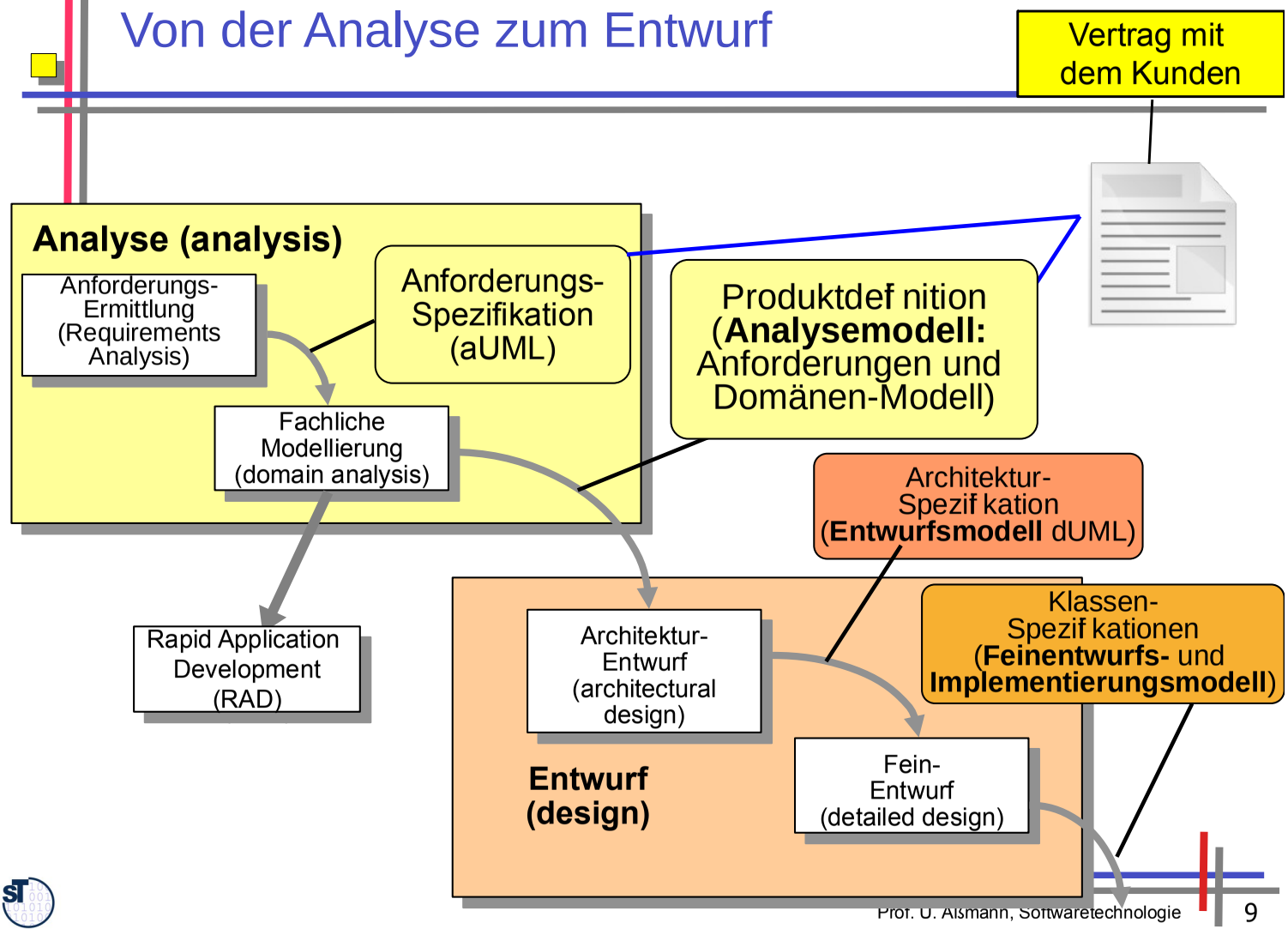


Boehm 1979 („V-Modell“)

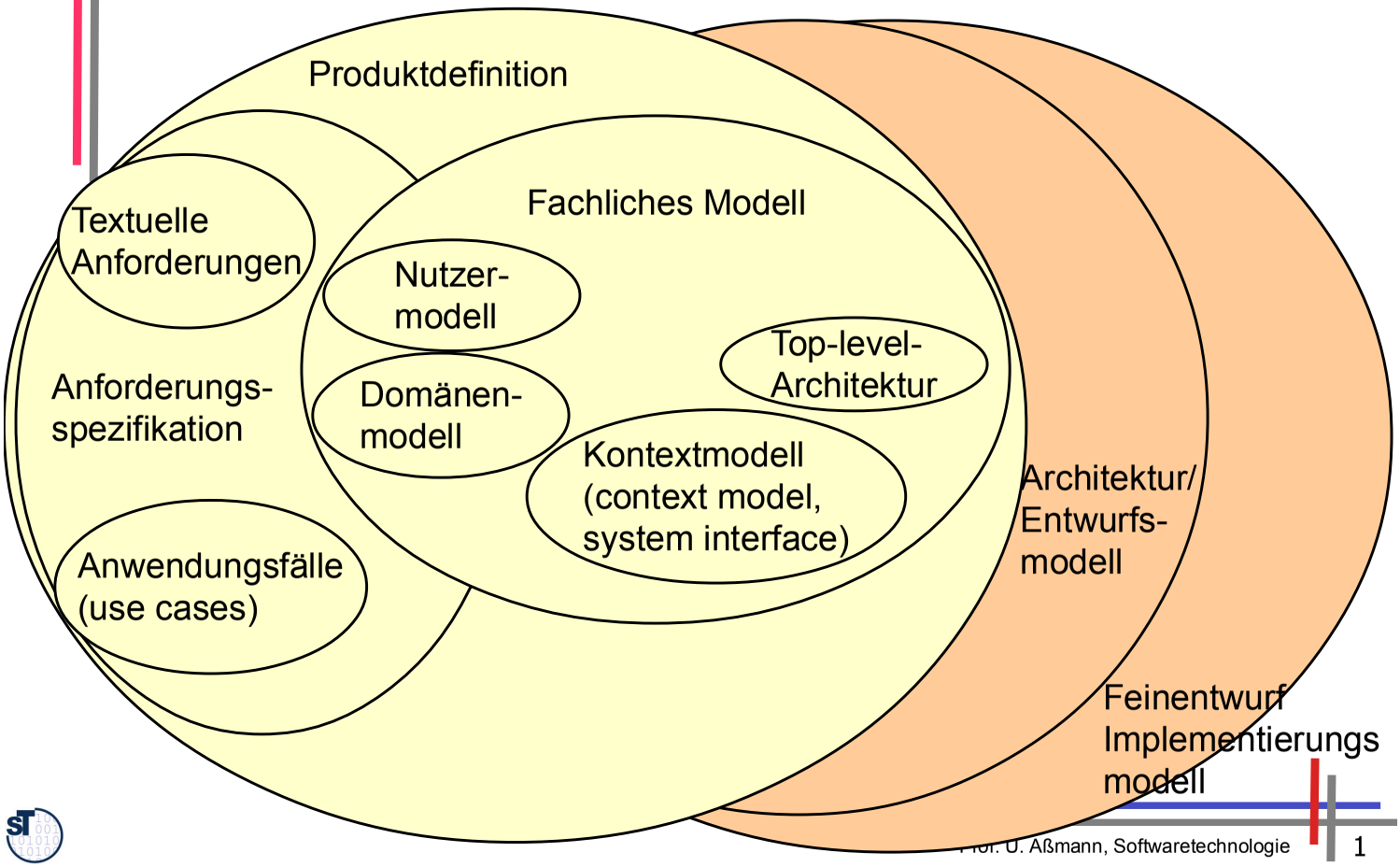
Objektorientierte Analyse und Entwurf

- Die Arbeitsphase **Analyse** ermittelt, was der Benutzer vom System benötigt, das **Analysemodell** in aUML
  - *Fachliche Analyse:* Begriffe und Objekte der Anwendungsdomäne (Domänenmodell, fachliches Modell)
  - *Anforderungsanalyse:* Formulierung der Anforderungen
  - *Systemanalyse:* Schnittstellen des Systems (Kontextmodell mit Funktionen, Daten, Klassen, Code)
- Die Arbeitsphase **Entwurf** ermittelt, was der Entwickler zusätzlich zu den Ergebnissen der Analyse ins System aufnehmen muss, was aber der Benutzer nicht sehen muss: das **Entwurfsmodell** in dUML
  - Der **Grobentwurf** entwickelt die Architektur ("Programmieren im Großen") im **Architekturmodell**
  - Der **Feinentwurf** entwickelt die Struktur von Klassen oder Subsystemen (**Feinentwurfsmodell**)
- Die Arbeitsphase **Implementierung** vervollständigt und konkretisiert den Entwurf
  - zum **Implementierungsmodell** (partielle Implementierung in jUML)
  - dann durch Ergänzung zum abläuffähigen (Java-)Programm
  - Klärt die Plattformabhängigkeiten des Programms

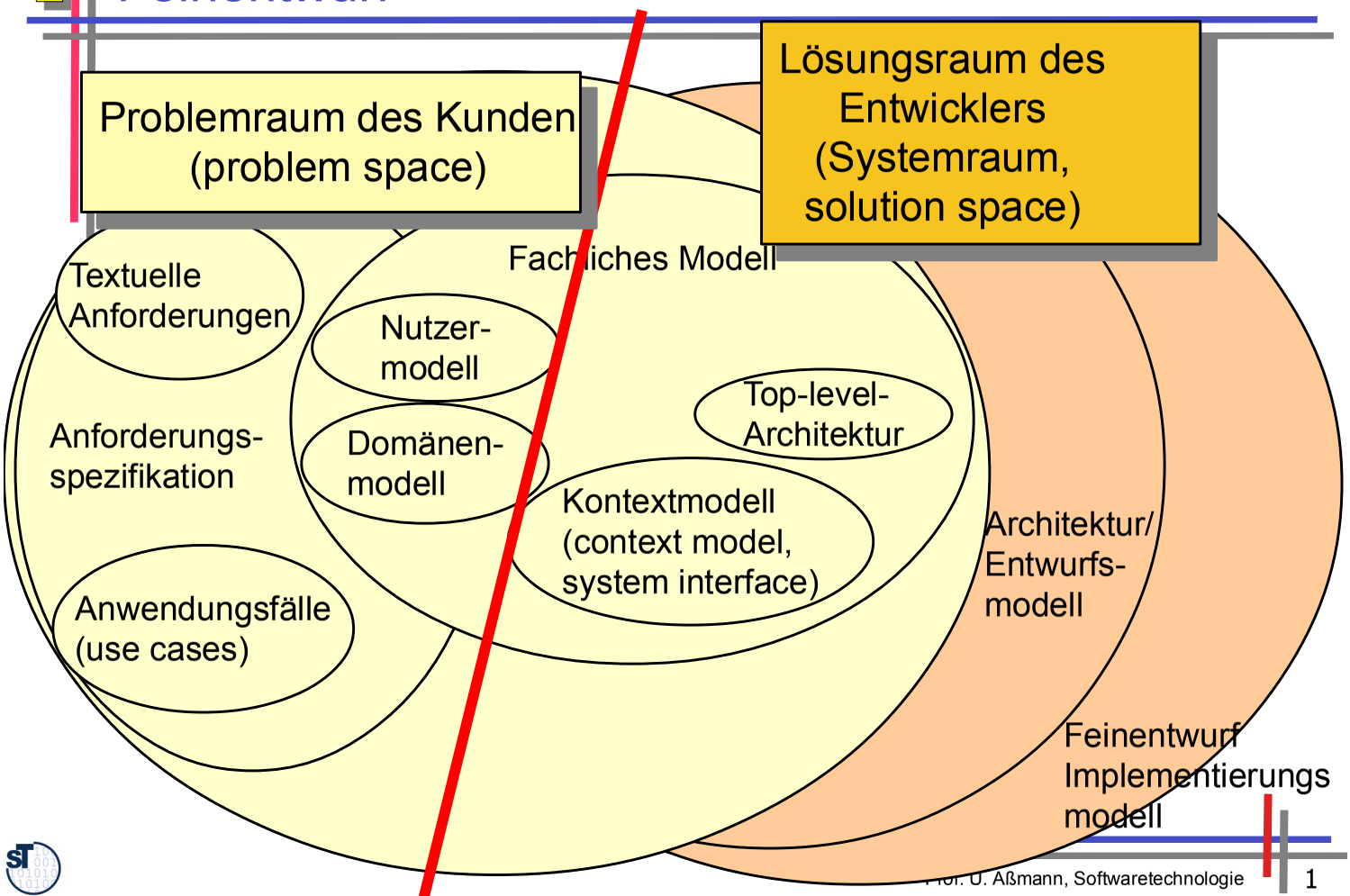
Von der Analyse zum Entwurf



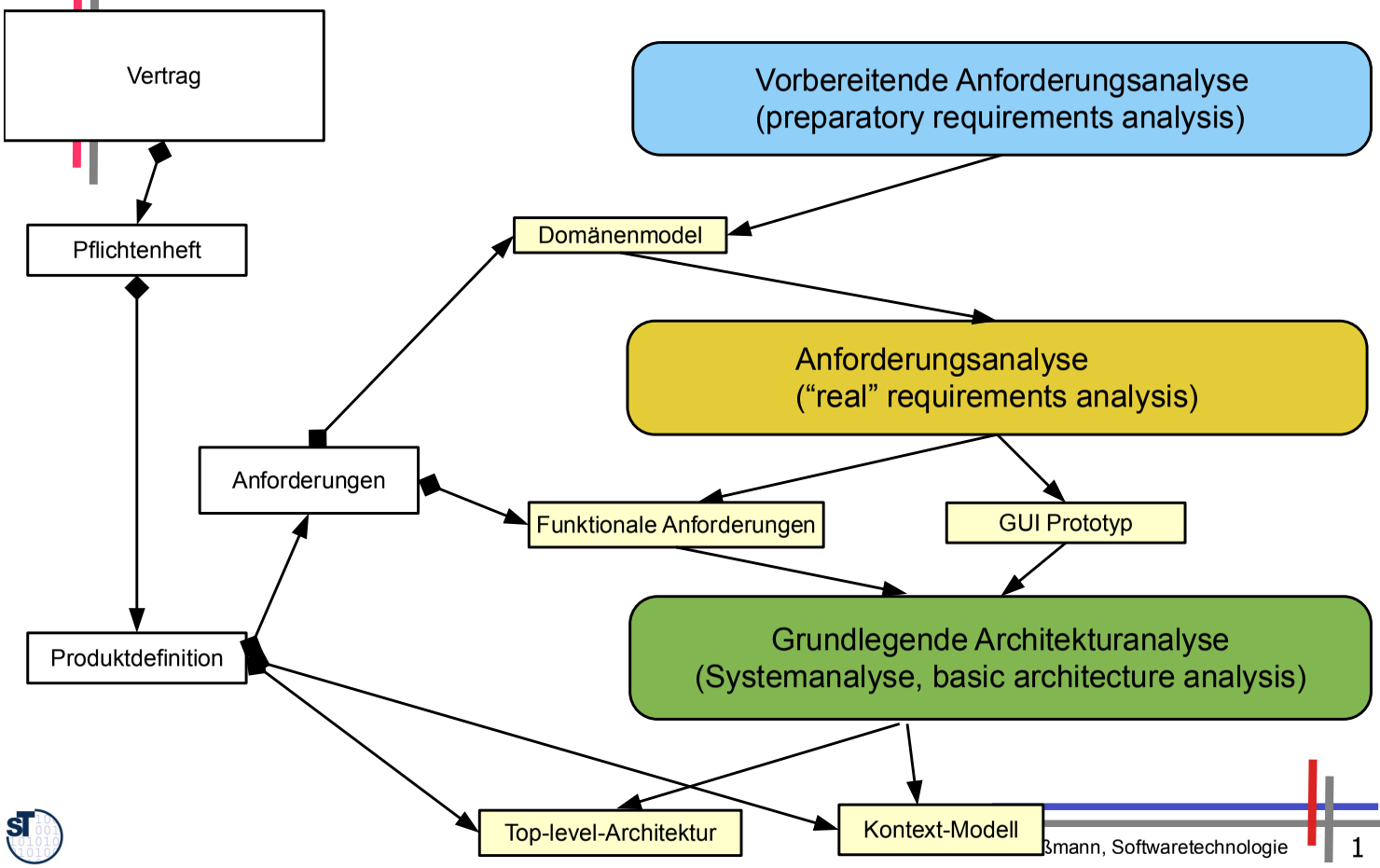
Artefakte im Prozess von den Anforderungen zum Feinentwurf



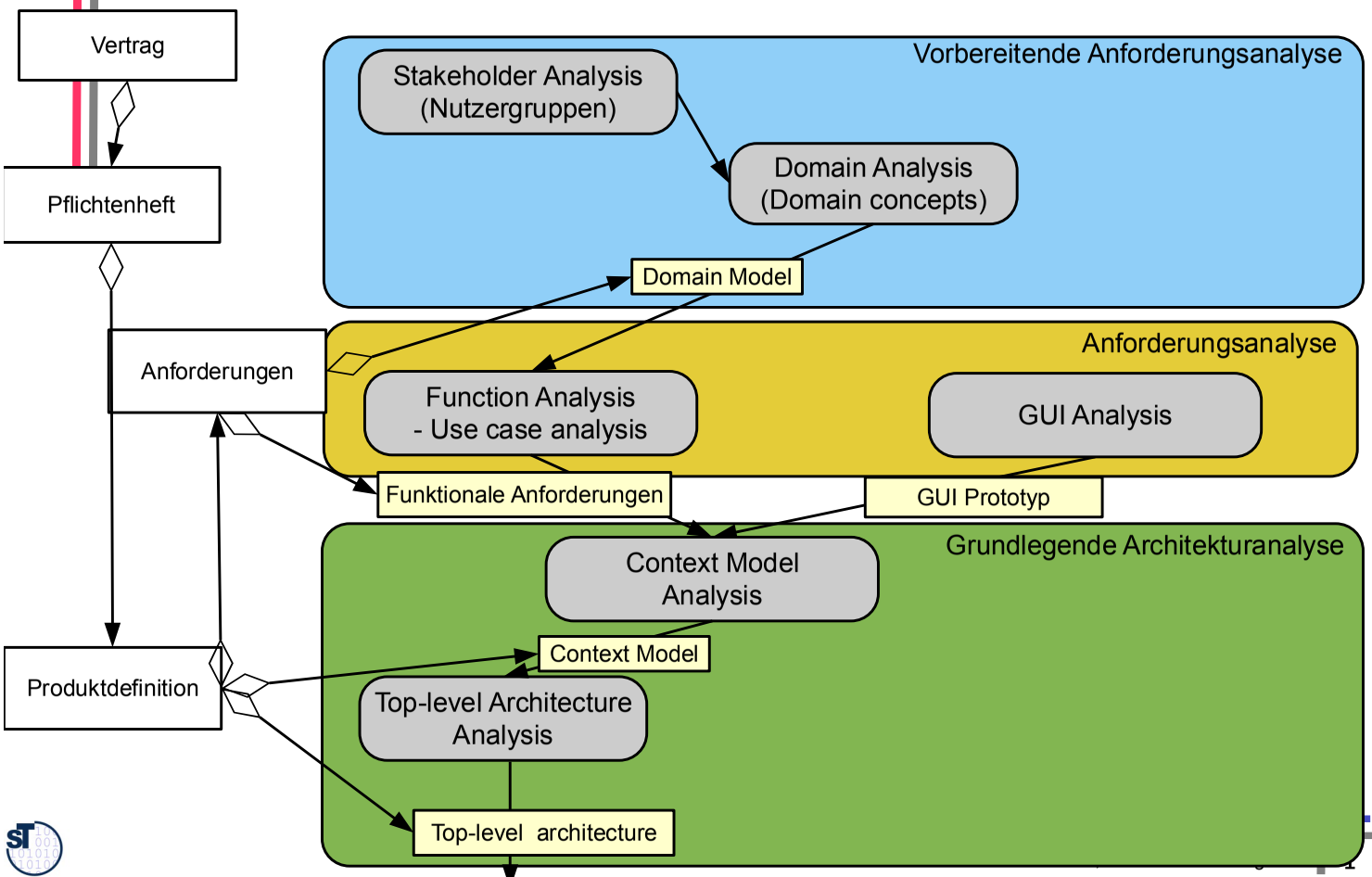
Artefakte im Prozess von den Anforderungen zum Feinentwurf



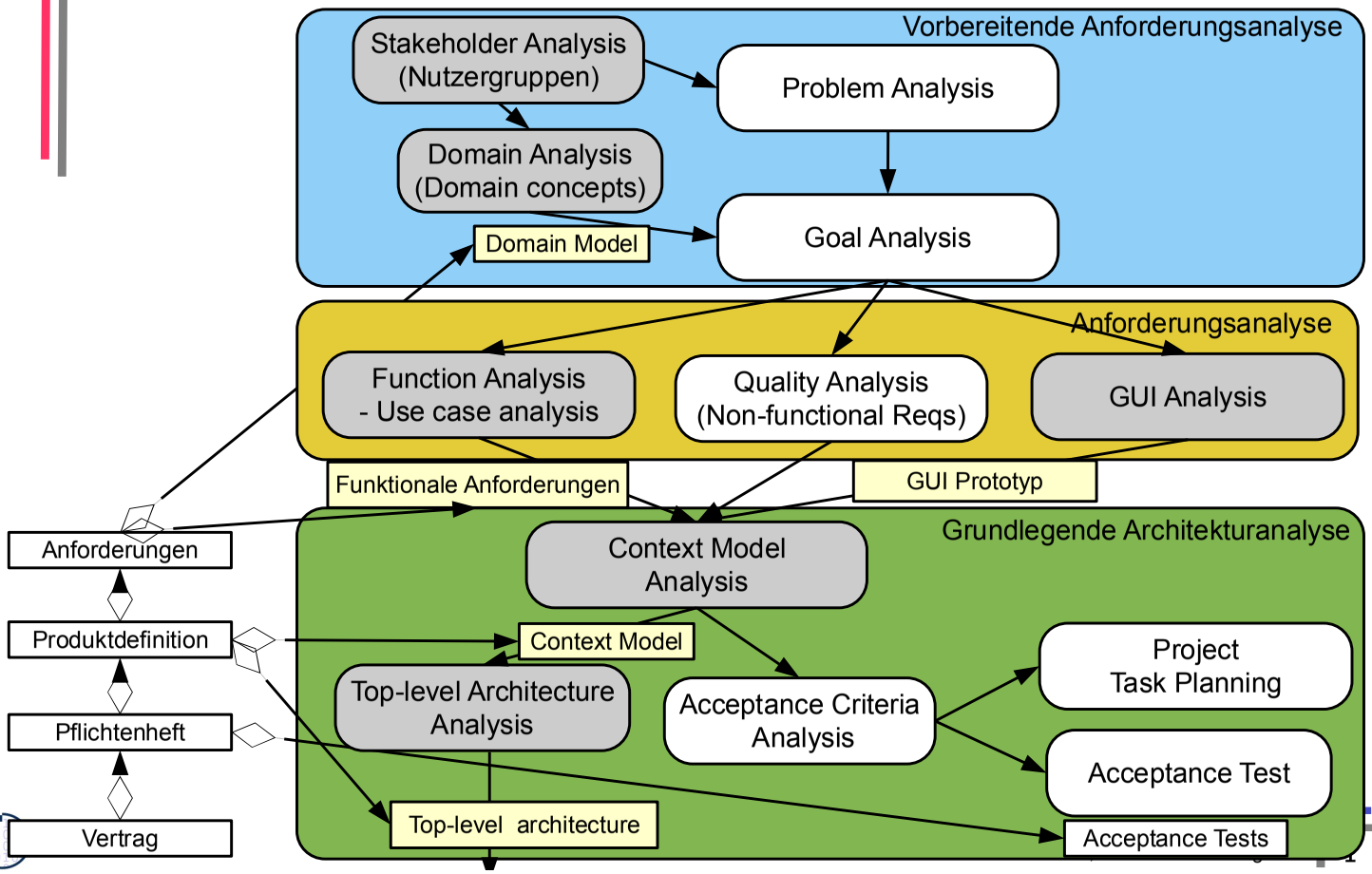
Drei-Schritt der Analyse (Anforderungen und fachliches Modell)



Drei-Schritt der Analyse (Anforderungen und fachliches Modell)



Voller Ablauf der Analyse (s. SWT-2)



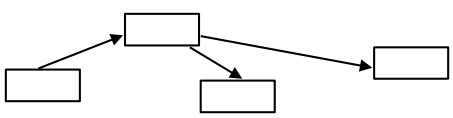
Objektorientierte Analyse (OOA)

- Grundidee: Modellierung der fachlichen Aufgabe (Produktdefinition mit fachlichem Modell und Anforderungsspezifikation) durch ***kooperierende Objekte*** in einem *statischen* und *dynamischen* Modell (*Struktur-* und *Verhaltensmodell*)

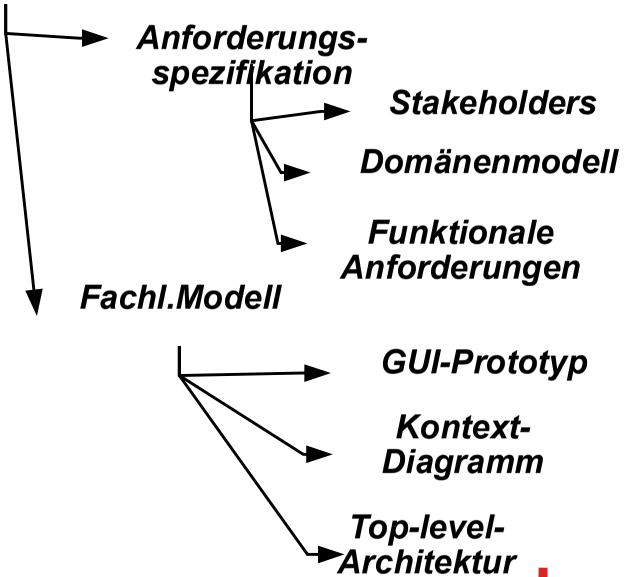
Produktdefinition (OOA Modell)

Statisches Modell (Struktur)	Dynamisches Modell (Verhalten)
Klassen: Eigenschaften und Aufgaben von Objekten	*Zustände* und *Verhalten* von Objekten
Beziehungen zwischen Klassen (Netz) *Konnektoren*	*Scheiben/Schnitte* durch das Verhalten mehrerer Objekte (*Szenarien*)

beschreibt

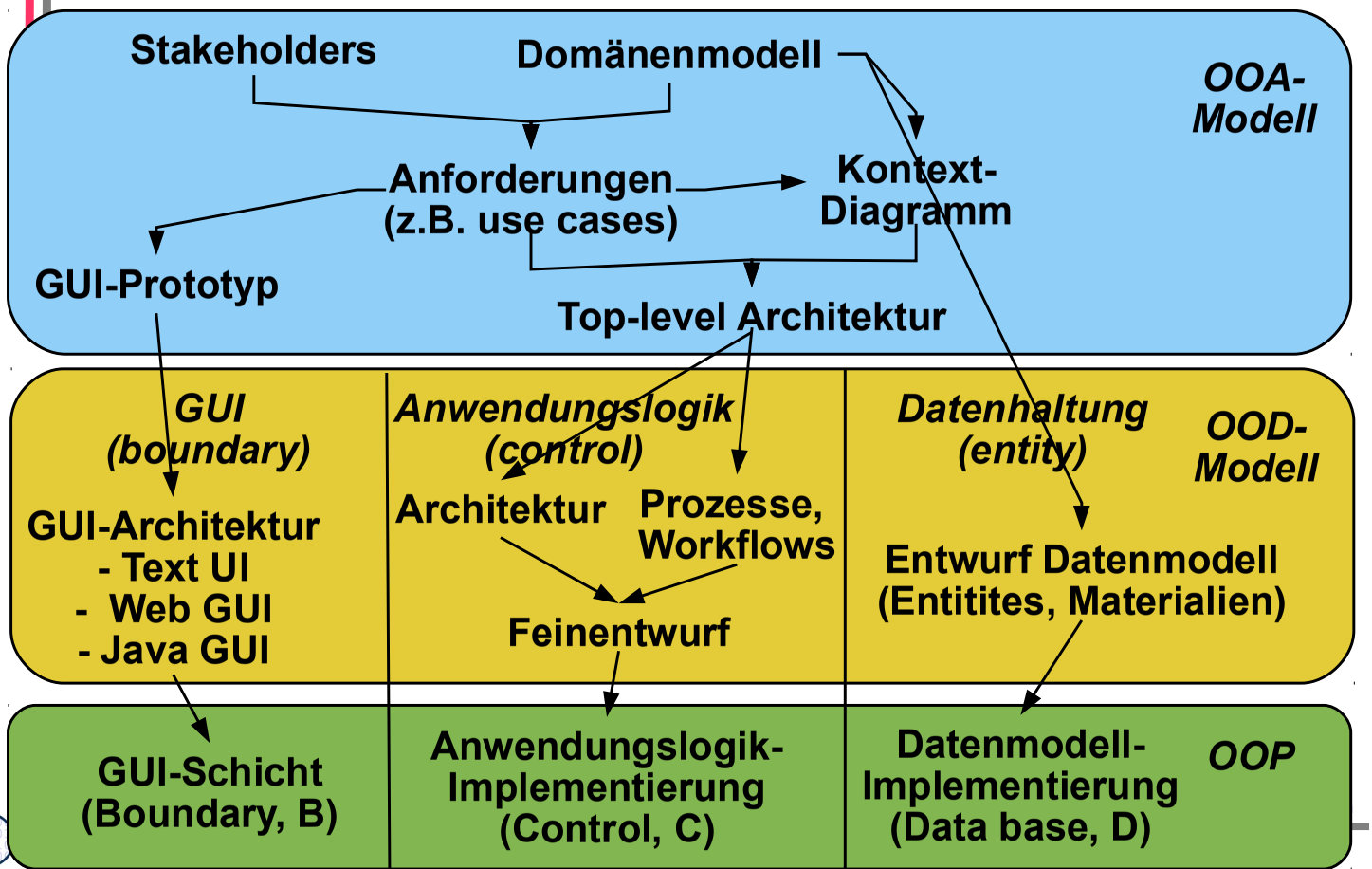


System als "Gesellschaft" kooperierender Objekte



	Statisch	Dynamisch	Mittel der UML
Punktweise Modellierung	Klassen- bzw. Objektmodellierung Schichten und Architektur		Objekt-zentriert Metamodell-getrieben, Canvas-getrieben
		Lebenszyklen	Aktionsdiagramme
Scheiben- (Schnitt-) Modellierung (slice modeling)	Relationale Modellierung (Netzmodellierung), Konnektoren		Relationen-zentriert Assoziationen Kollaborationen (Teams)
		Scheiben-Modellierung (Querschneidende Modellierung)	Szenarien-getrieben Interaktionsdiagramme

Von Analysemodellen zu BCD-Entwurfsmodellen (3-Schichtenarchitektur)



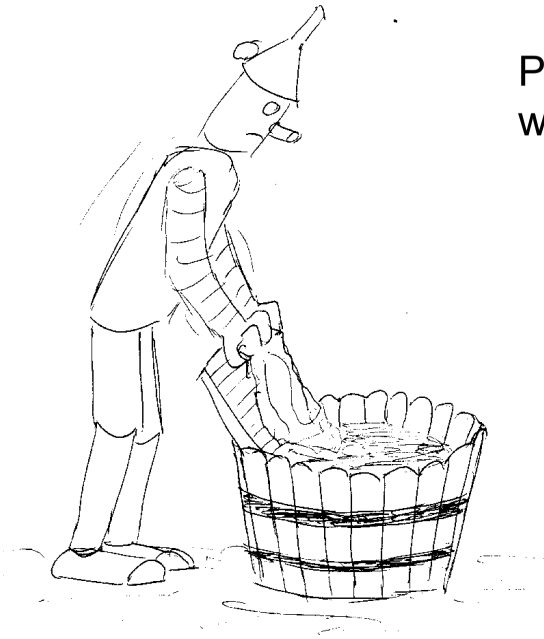
Zum Praktikum

- mehr als 95% aller Themen im Praktikum sind BCE-Architekturen
  - Oft mit Web-GUI
  - Unterscheidung zwischen GUI, Anwendungslogik und Datenhaltung ist essentiell
- Man sollte verstehen, dass aus dem Domänenmodell
  - die Datenhaltung folgt
  - die Typen der Daten folgen, die im Kontextmodell und in der Top-Level-Architektur fliessen
  - der GUI-Prototyp stark bestimmt wird (Kommunikation mit dem Benutzer)
- Die Entwickler oft nur für B, C, oder D zuständig sind

Überblick Teil III: Objektorientierte Analyse (OOA)

1. Überblick Objektorientierte Analyse
   1. (schon gehabt:) Strukturelle Modellierung mit CRC-Karten
2. Strukturelle metamodellgetriebene Modellierung mit UML für das Domänenmodell
   1. Strukturelle metamodellgetriebene Modellierung
   2. Modellierung von komplexen Objekten
      1. Modellierung von Hierarchien
      2. (Modellierung von komplexen Objekten und ihren Unterobjekten)
      3. Modellierung von Komponenten (Groß-Objekte)
   3. Strukturelle Modellierung für Kontextmodell und Top-Level-Architektur
3. Analyse von funktionalen Anforderungen
   1. Funktionale Verfeinerung: Dynamische Modellierung und Szenarienanalyse mit Aktionsdiagrammen
   2. Funktionale querschneidende Verfeinerung: Szenarienanalyse mit Anwendungsfällen, Kollaborationen und Interaktionsdiagrammen
   3. (Funktionale querschneidende Verfeinerung für komplexe Objekte)
4. Beispiel Fallstudie EU-Rent

Modeling the Real Need of the Customer



Prototype of a
washing machine

Find the right abstractions!

The Basic Laws of Misunderstanding in Analysis [K. Lorenz]

Spoken is not heard

Heard is not listened

Listened is not understood

Understood is not accepted

Accepted is not done

Appendix

- Einige Folien sind eine überarbeitete Version aus der Vorlesung Softwaretechnologie von © Prof. H. Hussmann, 2002, used by permission.

Inhalte eines Vertrags mit einem Kunden

- Pflichtenheft
  - Produktdefinition
    - Anforderungsspezifikation (das WAS)
      - Nutzermodell (stakeholders)
      - Domänenmodell
      - Funktionale Anforderungen
      - In SWT 2: Problemmodell, Zielmodell, Nicht-funktionale Anforderungen
    - Fachliches Modell (der Teil vom WIE, den der Kunde wissen muss)
      - Kontextmodell
      - GUI-Prototyp
      - Top-level-Architektur
  - Akzeptanztestfälle:
    - Messbare Akzeptanzkriterien, die bei der Abnahme vom Kunden abgehakt werden können. Ohne bestandenen Akzeptanztest keine Bezahlung!
- Preisliche Regelung
- *Achtung: In der Literatur wird der Begriff "Analysemodell" sowohl für die Produktdefinition als auch nur für das fachliche Modell verwendet!*

Inhalte der Anforderungspezifikation (WAS?)

- *Nutzermodell (stakeholder model):* Liste oder UML-Klassendiagramm aller am System Interessierten
  - In SWT-2 verfeinern wir das, in dem wir über die Ziele der stakeholder nachdenken
  - Enthält die Benutzer des Systems (die *Aktoren*)
- *Domänenmodell (domain model):*
  - Termini, Struktur und Grundkonzepte des Aufgabengebiets
  - Schaffung einheitlicher Terminologie für die Anforderungsspezifikation
  - Aus der Sicht des Kunden
  - Zusammenhang mit Anforderungsspezifikation sichern
  - Implementierungsaspekte ausklammern: Annahme *perfekter Technologie*
- *Problemmodell, Zielmodell* (s. SWT-2)

Inhalte der Anforderungspezifikation

- *Funktionale Anforderungen: Funktionale Essenz* des Systems. Was muss das System können?
  - Nicht das Wie, sondern nur das Was
  - möglichst quantitativ (z.B. Tabellenform)
  - eindeutig identifizierbar (Nummern)
  - Notation: Anwendungsfalldiagramme (Nutzfalldiagramme), Funktionsbäume oder textuell. Manchmal auch mathematisch
- *Nicht-funktionale Anforderungen (Qualitätsanforderungen)* (s. SWT-2)
  - Effizienzanforderungen
    - Resourcenausnutzung: Antwortzeit, Speicherbedarf, Last, Durchsatz, Energieverbrauch
  - Sicherheitskriterien
    - Zuverlässigkeit, Einbruchssicherheit, Privatsspährenschutz
  - HW/SW-Plattform
  - Entwicklungs- und Produkt-Standards

Inhalte des Fachlichen Modells (Fachkonzept) (Das WIE, das der Kunde wissen muss)

- *Kontextmodell*: äussere Schnittstellen des Systems
  - Ein- und Ausgabekanäle, Masken, Abfragen
  - Daten, die ein und aus fliessen, im Domänenmodell typisiert
- *GUI Prototyp:* Prototypische Masken, Formulare, Bildschirme, die den GUI ausmachen: Wie sieht das Programm aus?
- *Top-level Architektur (Initiale Architektur, Facharchitektur):* Bestimmt die Hauptkomponenten des Systems und ihre Interaktionen, ohne auf Details einzugehen
  - Verfeinert das Kontextmodell um eine Stufe, d.h. die top-level Architektur
  - Stellt das dar, was der Kunde von der Systemarchitektur wissen muss