

Teil III der Vorlesung

Objektorientierte Analyse (OOA)

30) Überblick über die OOA

1

Prof. Dr. Uwe Aßmann
Institut für Software- und Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
TU Dresden
Version 15-1.1, 13.06.15



- ▶ 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/>



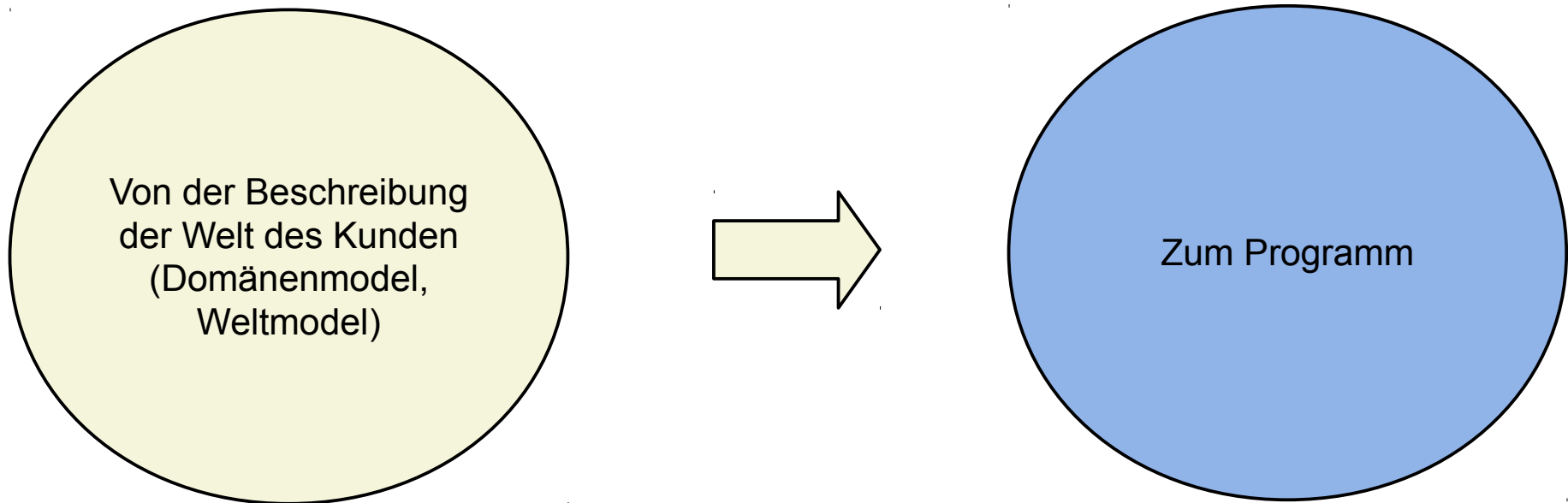
30.1 Überblick über die Objektorientierte Analyse

3

Die zentrale Frage der Softwaretechnologie

4

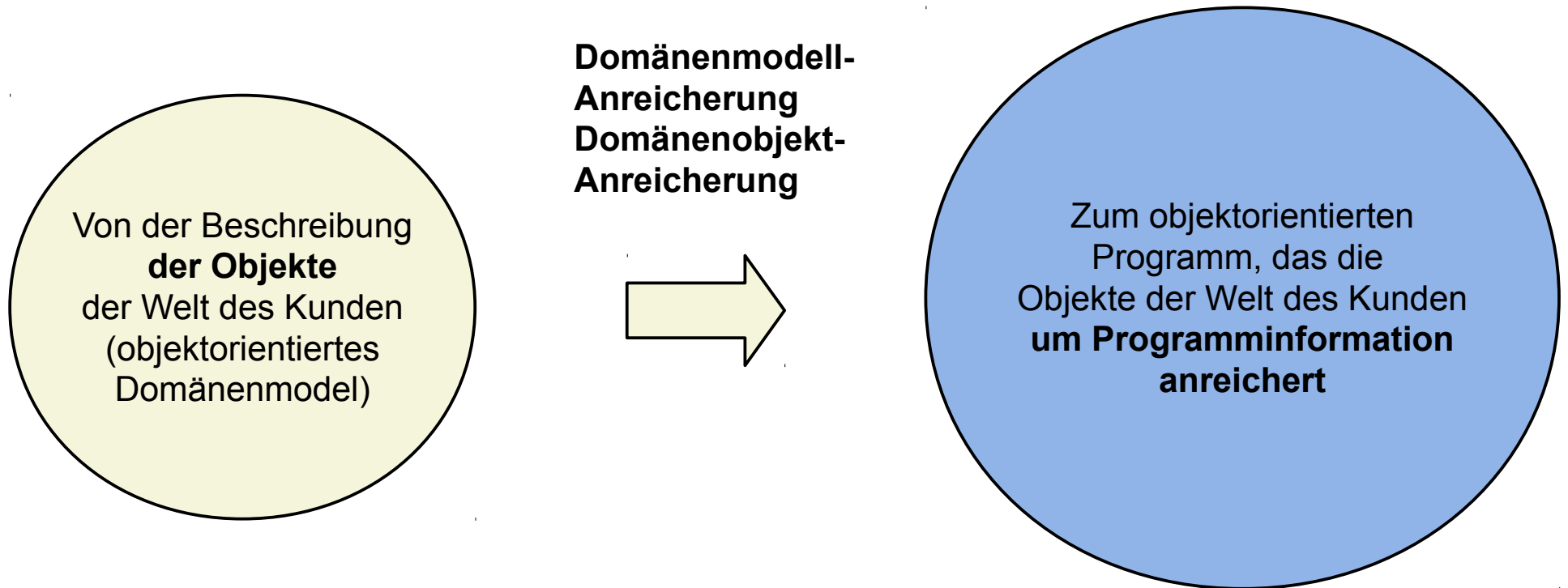
Wie kommen wir vom Problem des Kunden zum Programm (oder Produkt)?



Die zentralen Fragen des objektorientierten Ansatzes

5

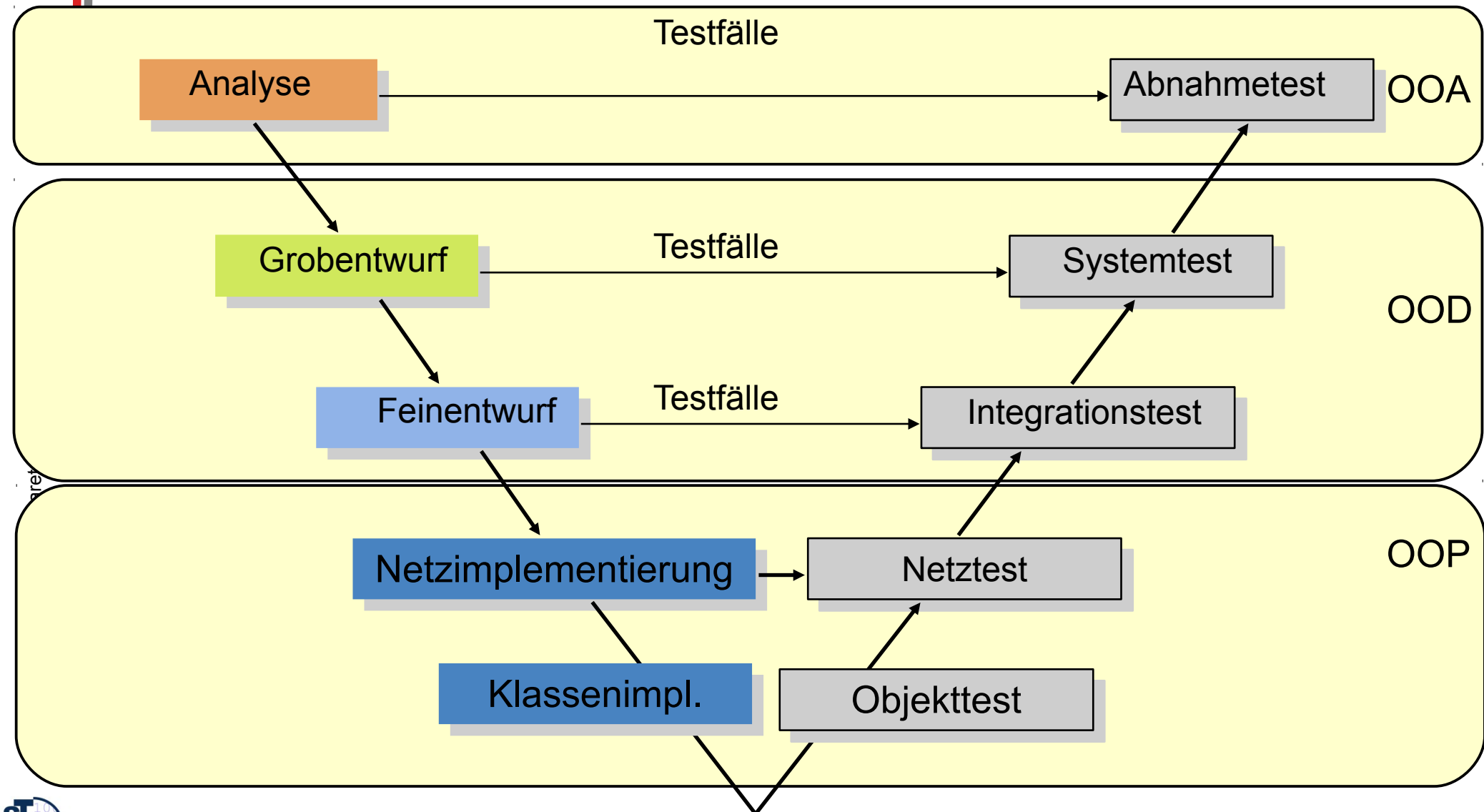
Wie kommen wir vom Problem des Kunden zum Programm (oder Produkt)?



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

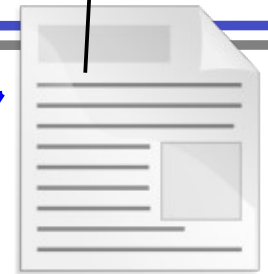
Softwareentwicklung im V-Modell

6



Von der Analyse zum Entwurf

Vertrag mit dem Kunden



Analyse (analysis)

Anforderungs-Ermittlung
(Requirements Analysis)

Anforderungs-Spezifikation
(aUML)

Fachliche Modellierung
(domain analysis)

Produktdefinition
**(Analysemodell:
Anforderungen und
Domänen-Modell)**

Architektur-Spezifikation
(Entwurfsmodell dUML)

Rapid Application
Development
(RAD)

Architektur-Entwurf
(architectural design)

**Entwurf
(design)**

Klassen-Spezifikationen
**(Feinentwurfs- und
Implementierungsmodell)**

Fein-Entwurf
(detailed design)

Implementierung

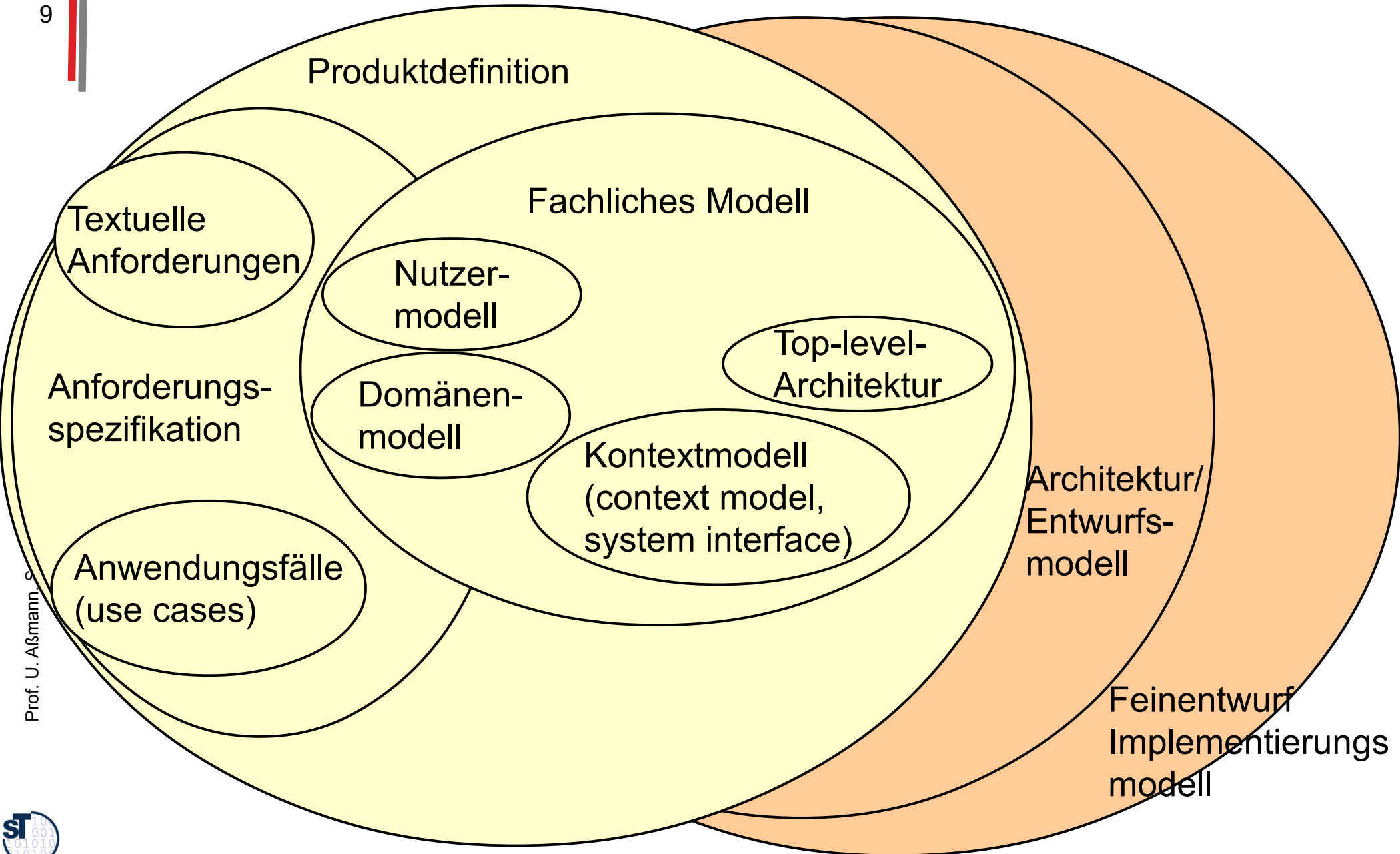
Modellierung vs. Realisierung

8

- ▶ **Zuerst** das Problem verstehen (Analyse),
dann Architektur festlegen (Entwurf)
dann in Implementierungsmodell überführen (Realisierung, Feinentwurf)
und zuletzt alle Details festlegen (Implementierung)

Artefakte im Prozess von den Anforderungen zum Feinentwurf

9



Artefakte im Prozess von den Anforderungen zum Feinentwurf

10

Problemraum des Kunden
(problem space)

Lösungsraum des
Entwicklers
(Systemraum,
solution space)

Textuelle
Anforderungen

Anforderungs-
spezifikation

Anwendungsfälle
(use cases)

Nutzer-
modell

Domänen-
modell

Fachliches Modell

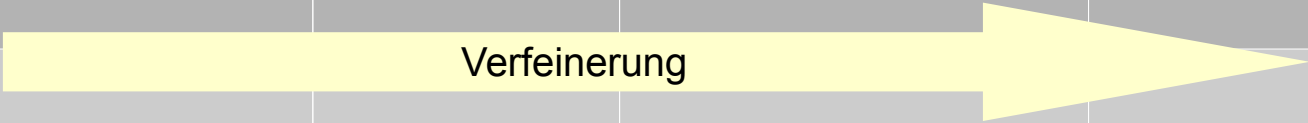
Top-level-
Architektur

Kontextmodell
(context model,
system interface)

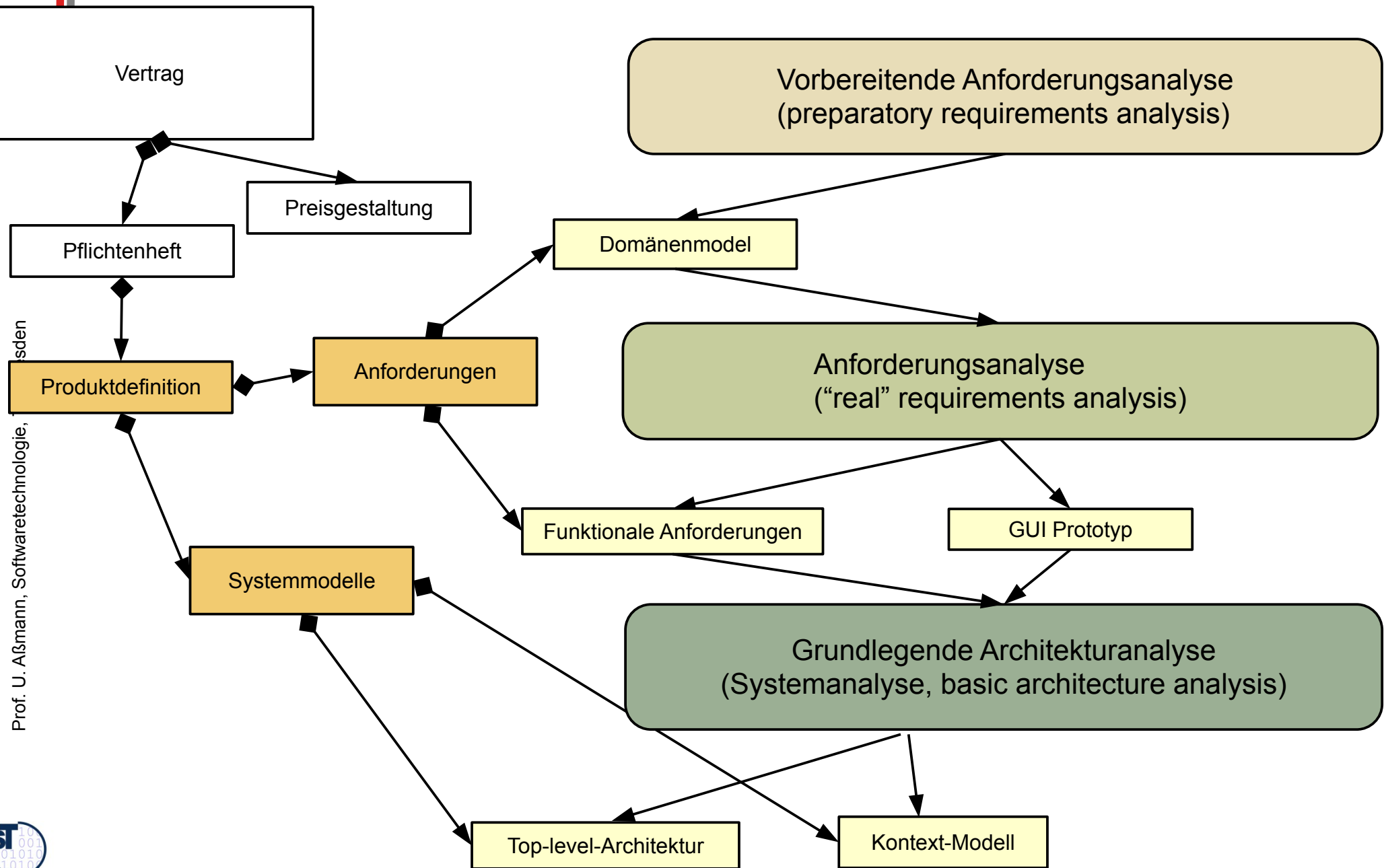
Architektur/
Entwurfs-
modell

Feinentwurf
Implementierungs
modell

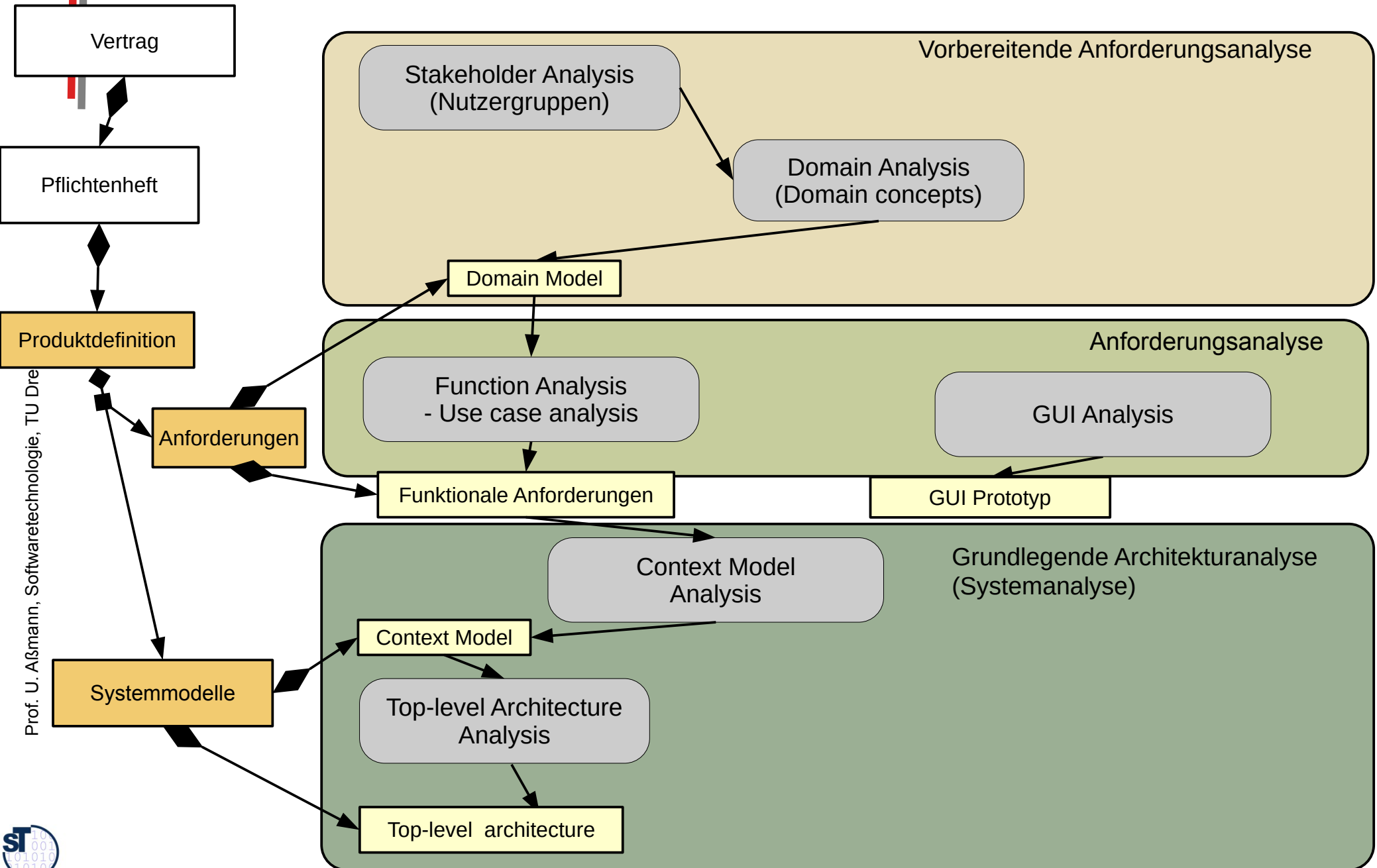
Q5: Schritte der Modellierung in Bezug auf Schichten des Systems

	Schichten	Analyse	Entwurf	Feinentwurf	Implementierung
Benutzungs-schnittstelle (Boundary)	GUI				
	Controller				
Anwendungs-logik (Control)	Kontextmodell	Aufstellung aus Domänenmodell; Erarbeitung System-schnittstellen	stabil	Umsetzen auf jUML	stabil
	Top-Level-Architektur	Verfeinerung des Kontextmodells	stabil	Umsetzen auf jUML	stabil
	Architektur		Ausarbeitung Architektur (PSM)	Einziehen von Plattformabhängigkeiten (PSM); Umsetzen auf jUML; Sequentialisierung	Details ausfüllen, Methoden ausprogrammieren
	Tools		Ausarbeitung Tools	Einziehen von Plattformabhängigkeiten (PSM); Umsetzen auf jUML; Sequentialisierung	Details ausfüllen, Methoden ausprogrammieren
Datenhaltung (Database)	Material	Aufstellung aus Domänenmodell		Einziehen von Plattformabhängigkeiten (PSM); Umsetzen auf jUML;	Details ausfüllen, Methoden ausprogrammieren

Drei-Schritt der Analyse (Anforderungen und fachliches Modell)



Drei-Schritt der Analyse (Anforderungen und fachliches Modell)



Statische und dynamische Aspekte der Modellierung

1.1

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

Zum Praktikum


15

- ▶ mehr als 95% aller Themen im Praktikum sind BCD-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)

16

- 
1. Überblick Objektorientierte Analyse
 1. (schon gehabt:) Strukturelle Modellierung mit CRC-Karten
 2. Strukturelle metamodellgetriebene Modellierung mit UML
 - Strukturelle metamodellgetriebene Modellierung für das Domänenmodell
 1. Modellierung von komplexen Objekten
 1. Modellierung von Hierarchien
 2. (Modellierung von komplexen Objekten und ihren Unterobjekten)
 3. Modellierung von Komponenten (Groß-Objekte)
 2. 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

Attention:

Model the Real Need of the Customer

17



Prototype of a
washing machine

Find the right abstractions!

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

18

Spoken is not heard
Heard is not listened
Listened is not understood
Understood is not accepted
Accepted is not done

Appendix

19

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



30.A.1 Dokumente der Anforderungsanalyse

20

Inhalte eines Vertrags mit einem Kunden

21

- ▶ 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 Anforderungsspezifikation (WAS?)

22

- ▶ *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)*

- ▶ *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
 - Ressourcenausnutzung: 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)

24

- ▶ *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

Voller Ablauf der Analyse (s. SWT-2)

