TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Fakultät Informatik

Professur Softwaretechnologie

OOSE_05

MODELLIERUNG MIT DER UML

Dr.-Ing. Birgit Demuth

Sommersemester 2017

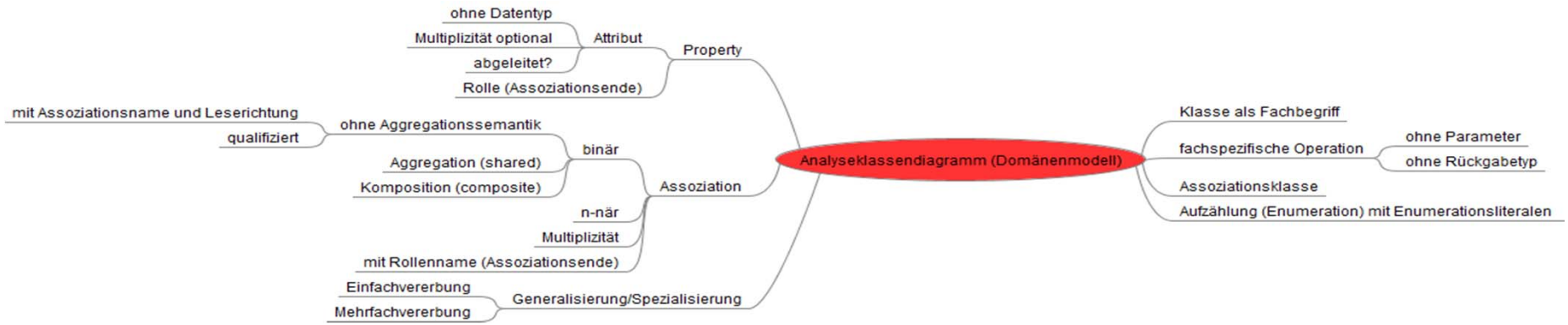
TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Modellierung mit der UML

OOA: KLASSEN- UND OBJEKTDIAGRAMME

Welche Modellelemente enthält ein UML-Analyseklassendiagramm (Domänenmodell)? [1][2]



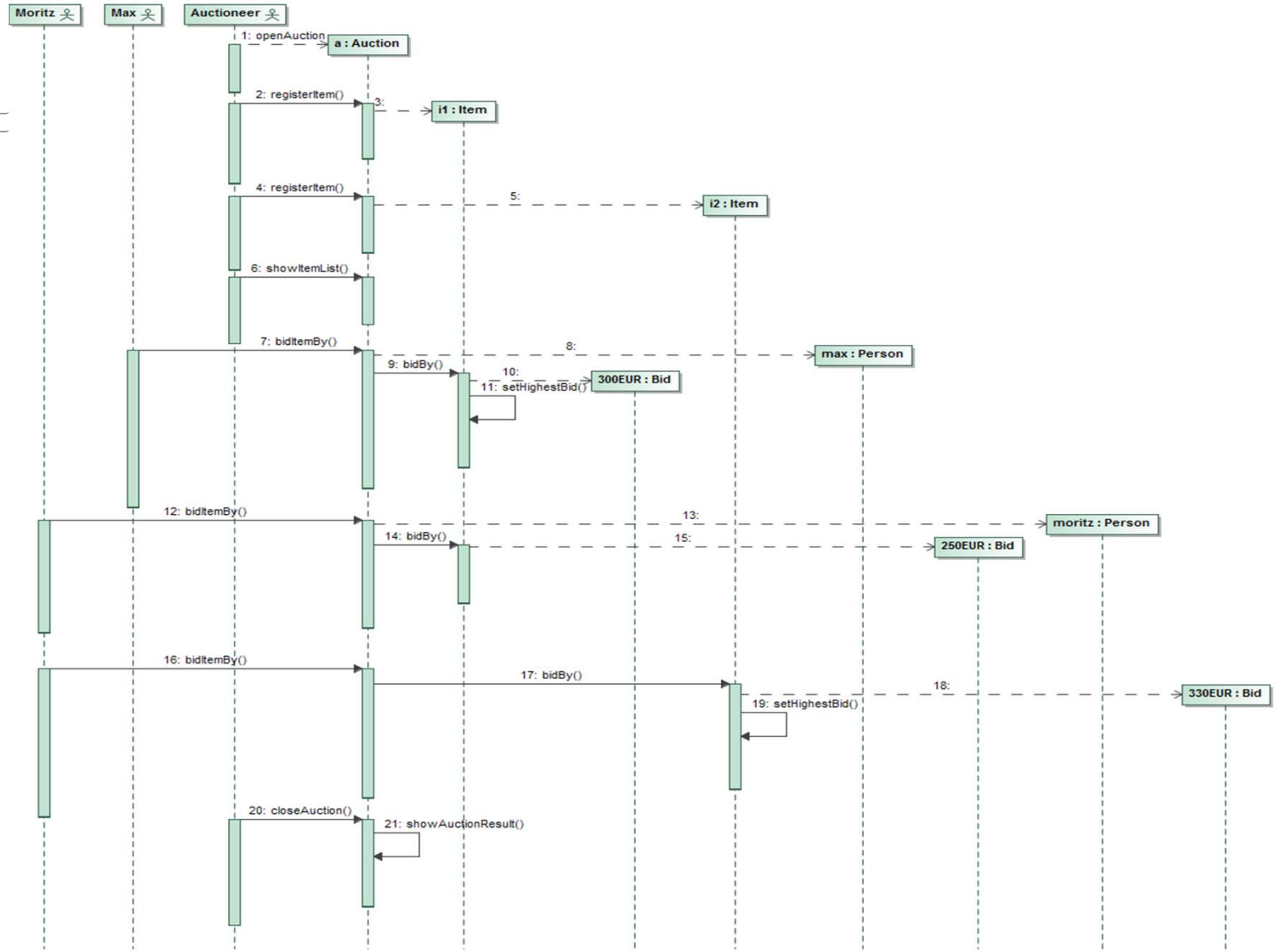
Wie gehen wir vor, ein Analyseklassendiagramm zu erstellen? (Variante: Verhaltensgetriebene Modellierung)

Schritte (Wiederholung, CRC-Kartenmethode, vgl. OOSE_02):

1. Textanalyse → Klassen(kandidaten) → „leere“ **CRC-Karten**
2. Analyse von Szenarien (Rollenspiel)
   - Schrittweises Ausfüllen der CRC-Karten
   - Protokollierung jedes Szenariums in einem UML-Sequenzdiagrammen
3. Abbildung der CRC-Karten in einem ersten UML-Analyseklassendiagramm
4. Anreicherung des UML-Analyseklassendiagramms

TECHNISCHE UNIVERSITÄT DRESDEN

Auktion Rollenspiel in einem UML-Sequenz-diagramm



Auktionen: CRC-Karten (zur Erinnerung)

Auction

Super Classes:

Sub Classes:

Description: repräsentiert eine Auktion

Attributes:

Name	Description
allItems	Liste von Posten
bidders	Liste alle Bieter einer Auktion

Responsibilities:

Name	Collaborator
openAuction	
registerItem	Item
bidItemBy	Item, Person
closeAuction	
showAuctionResult	

Item

Super Classes:

Sub Classes:

Description: Posten zur Versteigerung

Attributes:

Name	Description
number	identifiziert den Posten
description	Beschreibung des Postens
minPrice	Einstandspreis des Postens
allBids	Liste der abgegebenen Gebote für den Posten
highestBid	bislang höchstes Gebot für den Posten

Responsibilities:

Name	Collaborator
bidBy	Bid
setHighestBid	

Bid

Super Classes:

Sub Classes:

Description: ein Gebot für einen Posten

Attributes:

Name	Description
price	gebotener Preis
bidder	zugehöriger Bieter (Person)

Responsibilities:

Name	Collaborator
createBid	

Person

Super Classes:

Sub Classes:

Description: Person, die ein Gebot abgibt

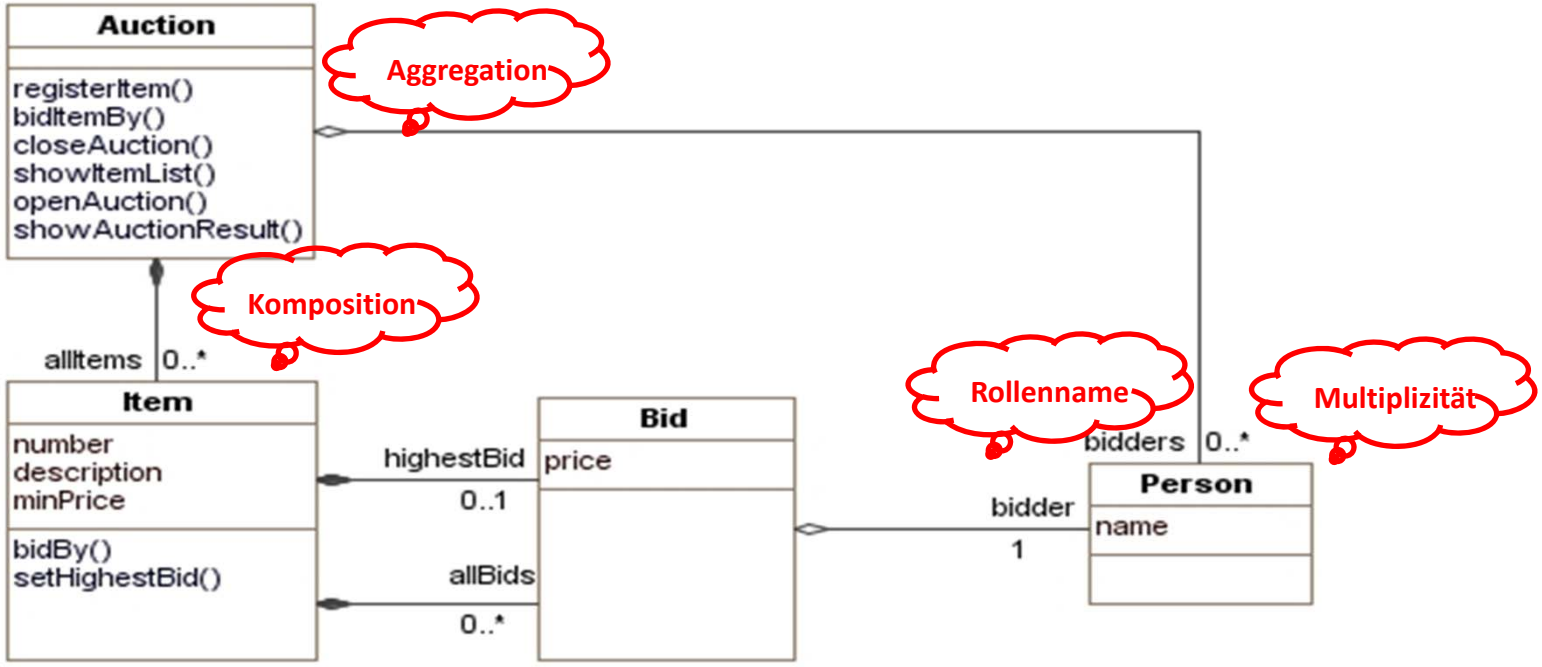
Attributes:

Name	Description
name	

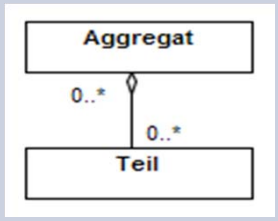
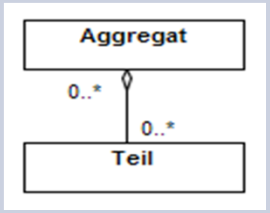
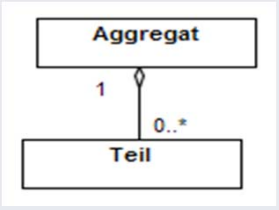
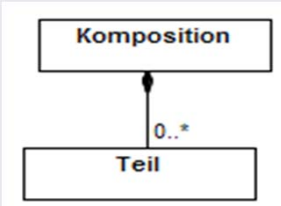
Responsibilities:

Name	Collaborator
createPerson	

Analyseklassendiagramm



Teil-Ganzes-Beziehungen in UML

Eigenschaften	Unabhängiges Teil (independent part)	Abhängiges Teil (dependent part)
Geteiltes Teil (shared part)	Aggregation 	Aggregation 
Nicht geteiltes Teil (exclusively-owns-part)	Aggregation 	Komposition 

Wie gehen wir vor, ein Analyseklassendiagramm zu erstellen? (Variante: Strukturgetriebene Modellierung)

Textanalyse und schrittweise Anreicherung des Analyseklassendiagramms

Schritte:

1. Klassen identifizieren
2. Attribute identifizieren
3. Operationen identifizieren
4. Enumerationen identifizieren
5. Klassenbeziehungen identifizieren
   - Vererbungsbeziehungen (Generalisierung/Spezialisierung)
   - Assoziation/Aggregation/Komposition/Assoziationsklassen/n-äre Beziehungen
6. Verfeinerte Beschreibung der Klassenbeziehungen

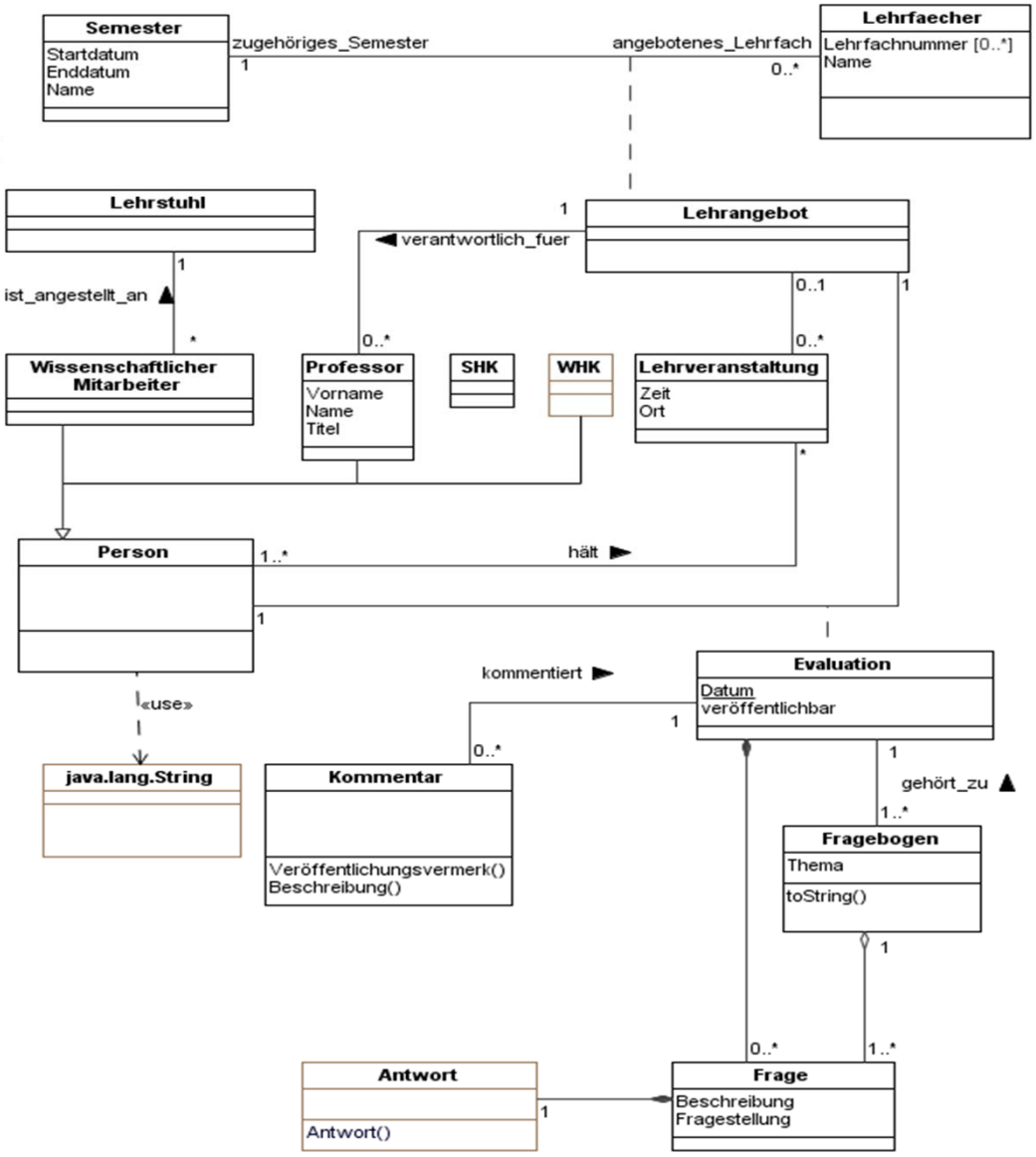
Review eines Analyseklassendiagramms
Fallbeispiel Campus Management System
(Klausur WS 2013/14)

Analyseklassendiagramm für den Teil **Lehrevaluation** gegeben

Gesucht

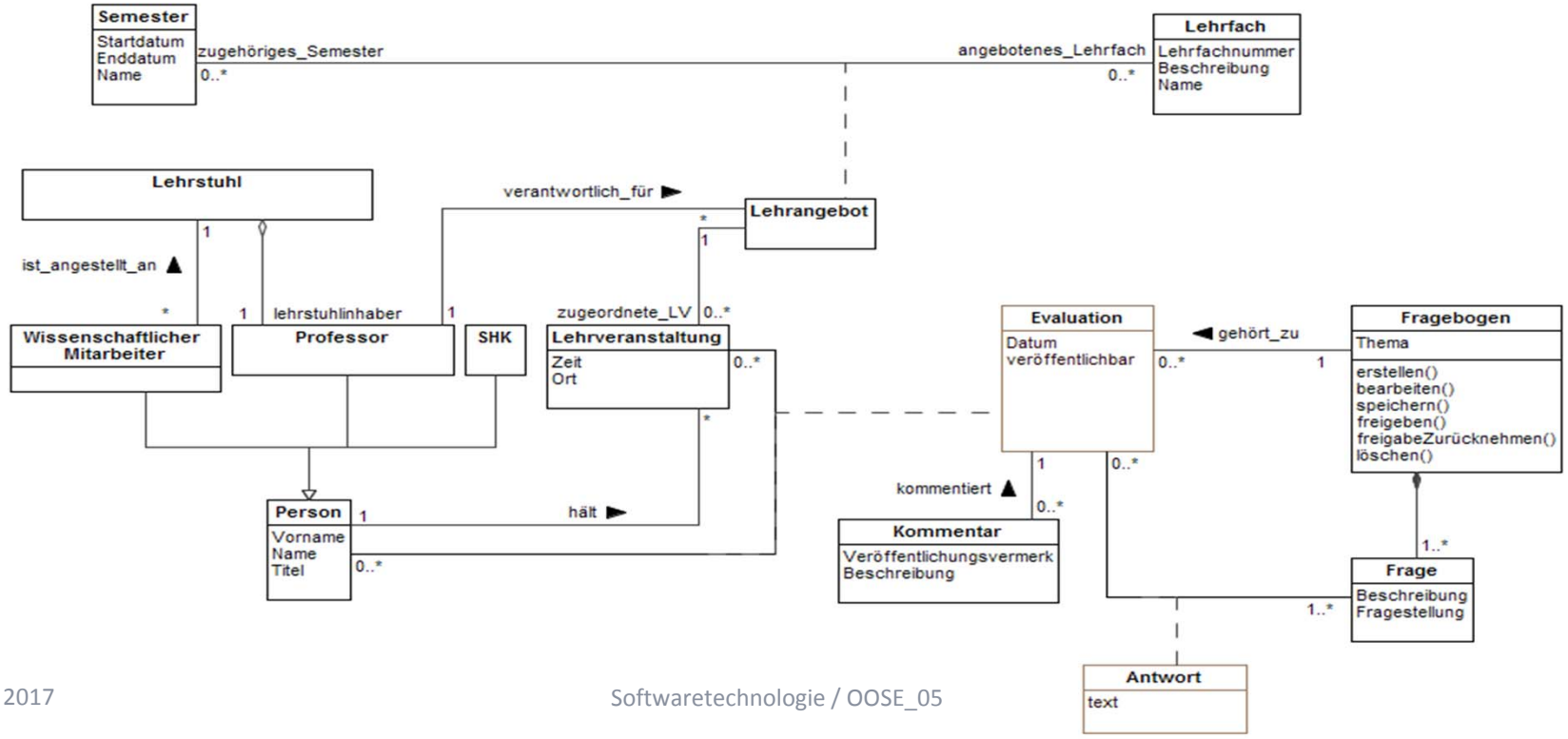
→Modellierungsfehler (F)
→Verletzung von Modellierungskonventionen in der Analyse (K)
→Vorschlag für eine Verbesserung (V)
Vorgehen: Lesen der textuellen Spezifikation und mit dem Modell vergleichen!

Analyse-klassen-diagramm mit Fehlern

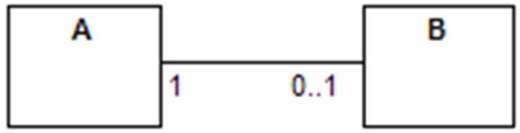


Nr	Änderung betrifft …	(optionaler) Kommentar	Art
1	Lehrfaecher	Singular verwenden	K
2	Lehrfachnummer[0..*]	Eindeutig bedeutet [1]	F
3	Lehrfaecher	Attribut Beschreibung fehlt	F
4	Assoziationsklasse Lehrangebot	Multiplizität Semester falsch	F
5	Leserichtung „verantwortlich_fuer“	umdrehen	F
6	Multiplizitäten Professor - Lehrangebot	umdrehen	F
7 8	Assoziation Lehrangebot-Lehrveranstaltung - Multiplizität von Lehrangebot - Nähere Beschreibung der Assoziation	 - 1 statt 0..1 - Z.B. mit Rollennamen	 F K
9	Assoziation „hält“ – Multiplizität von Person	1 statt 1..*	F
10	SHK: Vererbung fehlt	Zu Person	F
11	WHK	Nicht erklärt und damit falsch	F
12	Assoziation Lehrstuhl - Professor	fehlt	F
13	Professor: Attribute	Gehören zu Person	F
14 14a	Assoziationsklasse Evaluation: Multiplizitäten und Beteiligte Klassen: Lehrangebot und Person	0..*/0..* statt 1/1 Lehrveranstaltung statt Lehrangebot	F+F F
15	Datum der Evaluation	Statische Eigenschaft gehört nicht in die OOA	K
16	toString()	Gehört nicht in die OOA	K
17	Evaluation – Fragebogen: Multiplizitäten	0..* und 1 statt 1 und 1..*	F+F
18	Antwort	Attribut Text fehlt	F
19	Antwort()	Konstruktor zu technisch, gehört nicht in die OOA	K
20	Kommentar - Methoden	Sind Attribute	F+F
21	Evaluation – Frage - Antwort	Antwort ist Assoziationsklasse	V
22	java.lang.String()	gehört nicht in das Analyseklassendiagramm	K

Fallbeispiel Campus Management System
(refaktorisiertes Analyseklassendiagramm)

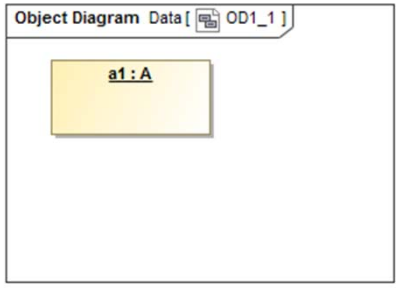


Objektdiagramme: Beachte die Konsistenz zwischen Klassen- und Objektdiagrammen!

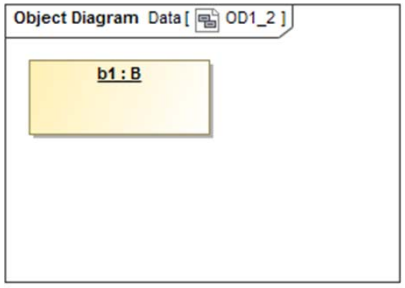


Was sind valide Objektdiagramme?

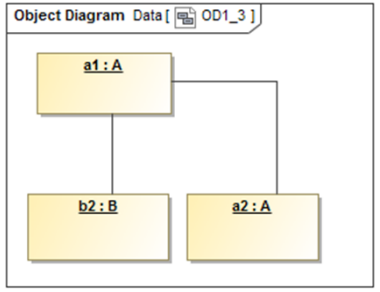
(OD_1)



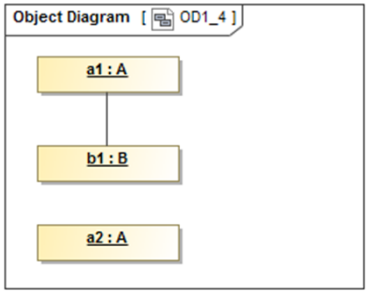
(OD_2)



(OD_3)



(OD_4)



TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Modellierung mit der UML

OOA: ZUSTANDSDIAGRAMME

Klassifikation von UML-Zustandsmodellen

nach Abstraktionsebene

- Analyseebene → ***Analysezustandsmodell (ASM)***
- Entwurfsebene → ***Entwurfszustandsmodell (DSM)***

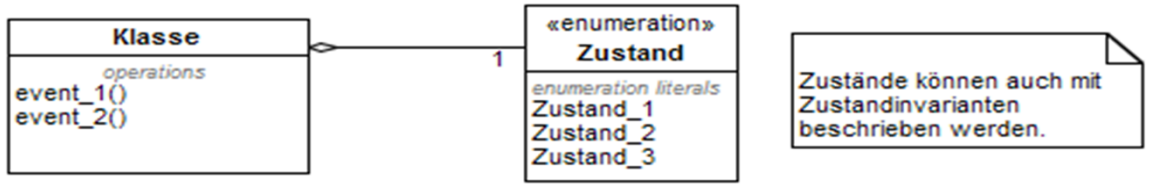
danach, worauf sich das Zustandsmodell bezieht

- Objekt → ***Objektlebenszyklus (OLC)***
- Anwendungsfall → ***Anwendungsfalllebenszyklus (ULC)***
- Beschreibung eines technischen Gerätes/Systemes
  → *(technische)* ***Steuerungsmachine (CSM)***

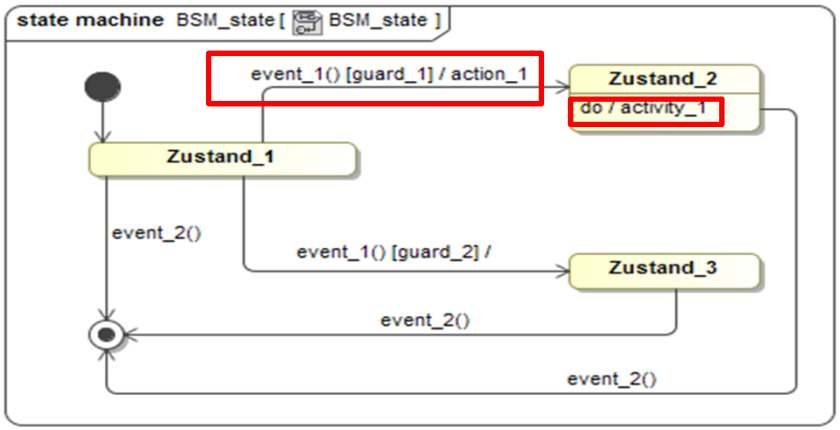
entsprechend der UML2.0-Spezifikation

- Spezifikation des Verhaltens
  → ***Verhaltens(zustands)maschine (BSM)***
  (Behavioral state machine)
- Spezifikation der zulässigen Reihenfolge von Operationen
  → ***Protokollmaschine (PSM) (Protocol State Machine)***

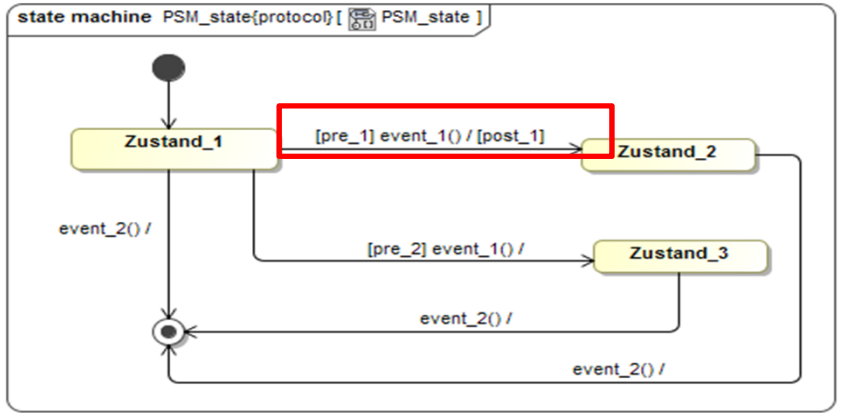
UML 2: Verhaltenszustandsmaschine (BSM) versus Protokollmaschine (PSM) [OLC]



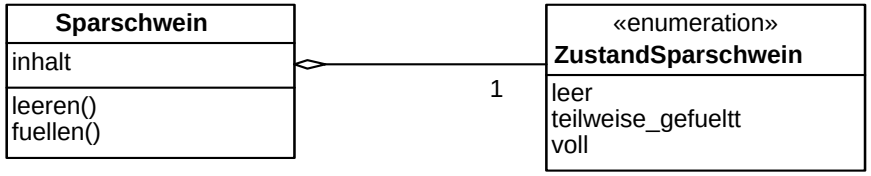
BSM

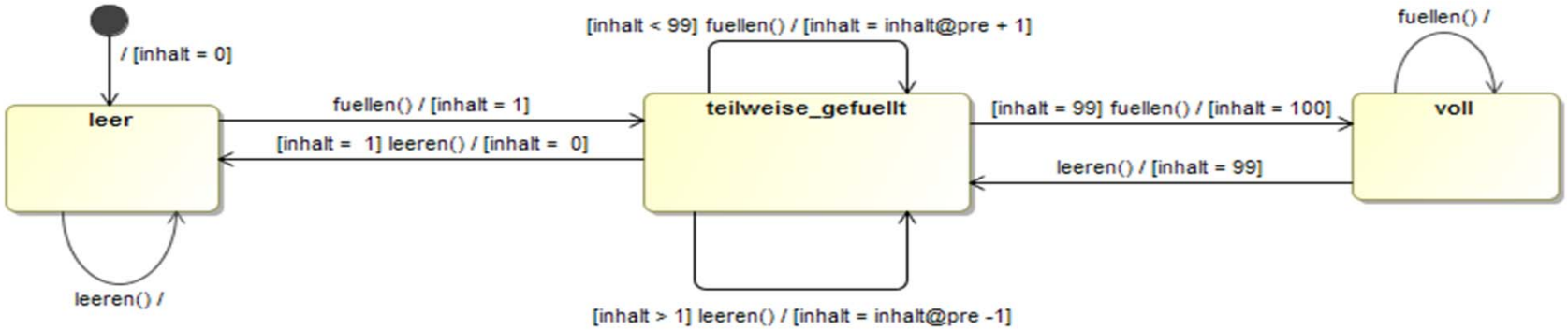


PSM



Objektlebenszyklus (PSM) [OLC]
Lebenszyklus eines Sparschweins (Klausur SS 2013)





Habe ich mein Zustandsmodell [OLC] richtig konstruiert? (2)

(1)Habe ich das Zustandsdiagramm für einen **nicht-trivialen Lebenszyklus** erstellt?

(2)Welche **Zustände** enthält das Diagramm?

(3)Existieren **Endzustände**?

(4)Welche **Ereignisse** sind zu modellieren?

(5)Welche **Operationen** besitzt das Objekt?

(6)Sind Operationen als **Aktivitäten** oder **Aktionen** zu modellieren?

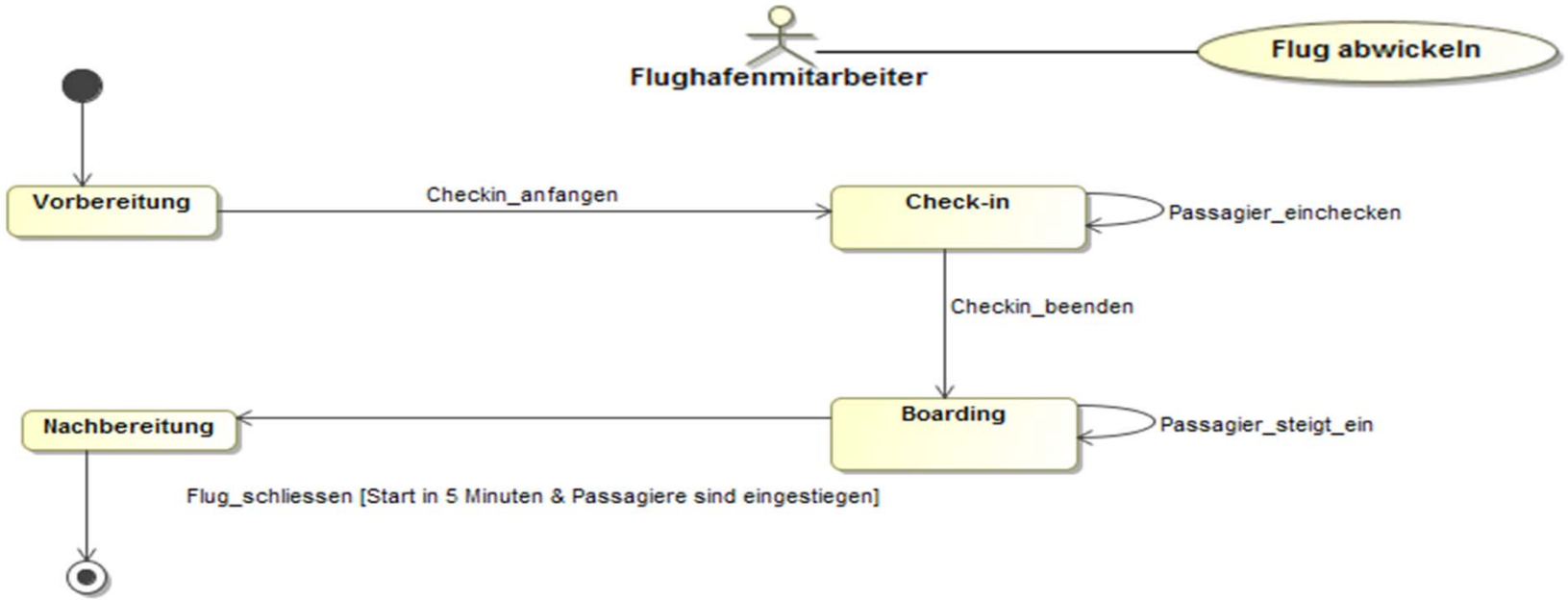
(7)Hängen Zustandsübergänge von Bedingungen ab?

(8)Habe ich geeignete **Zustandsnamen** gewählt?

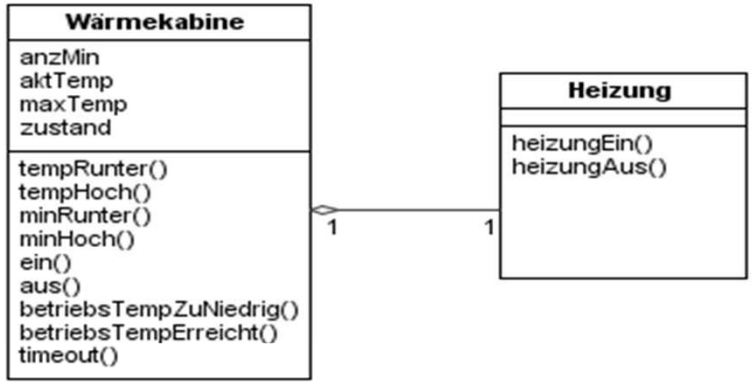
(9)Ist das Zustandsmodell **konsistent** mit dem zugehörigen Klassendiagramm?

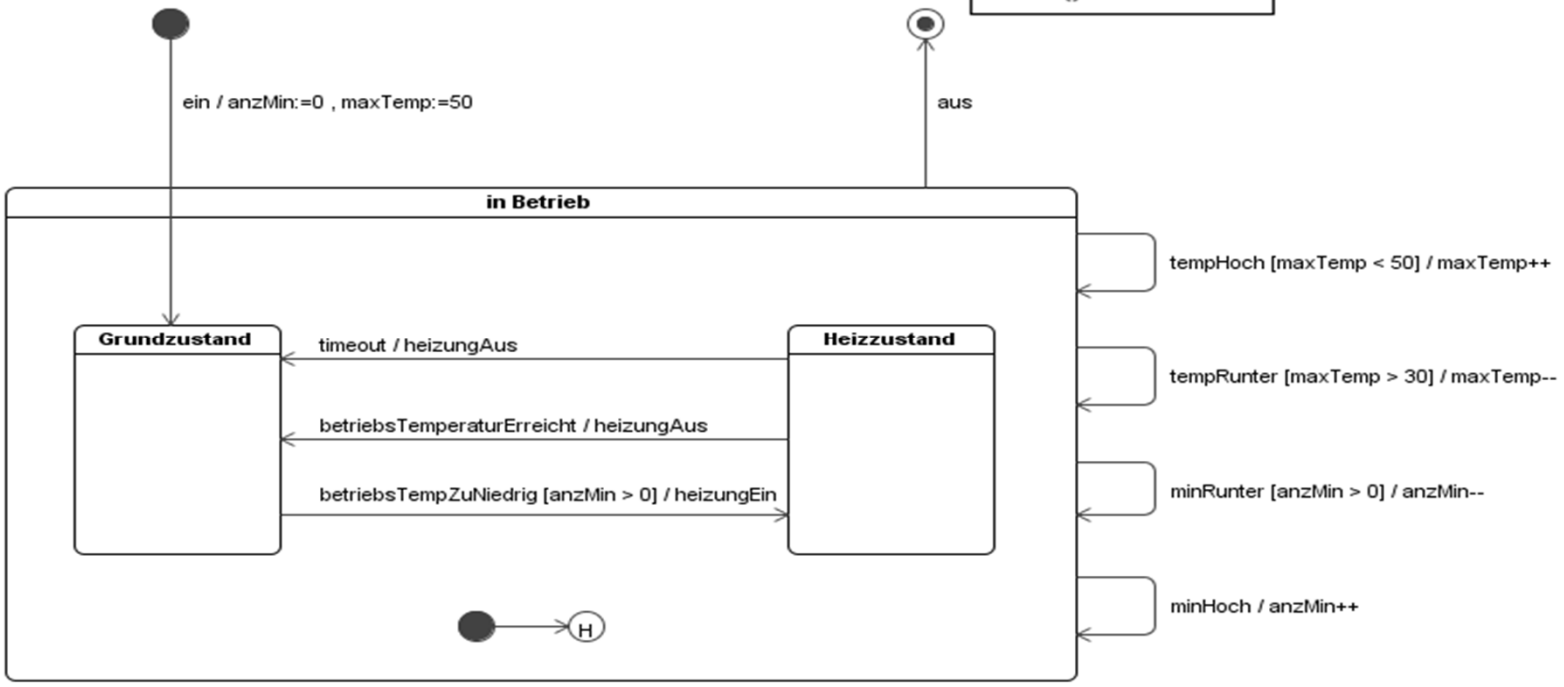
(10) Sind alle **Zustandsübergänge** korrekt eingetragen?

Anwendungsfalllebenszyklus (BSM) [ULC]
Flug abwickeln (1)



Steuerungsmaschine (BSM) Infrarot-Wärmekabine (CSM) (Klausur WS 2006/07)





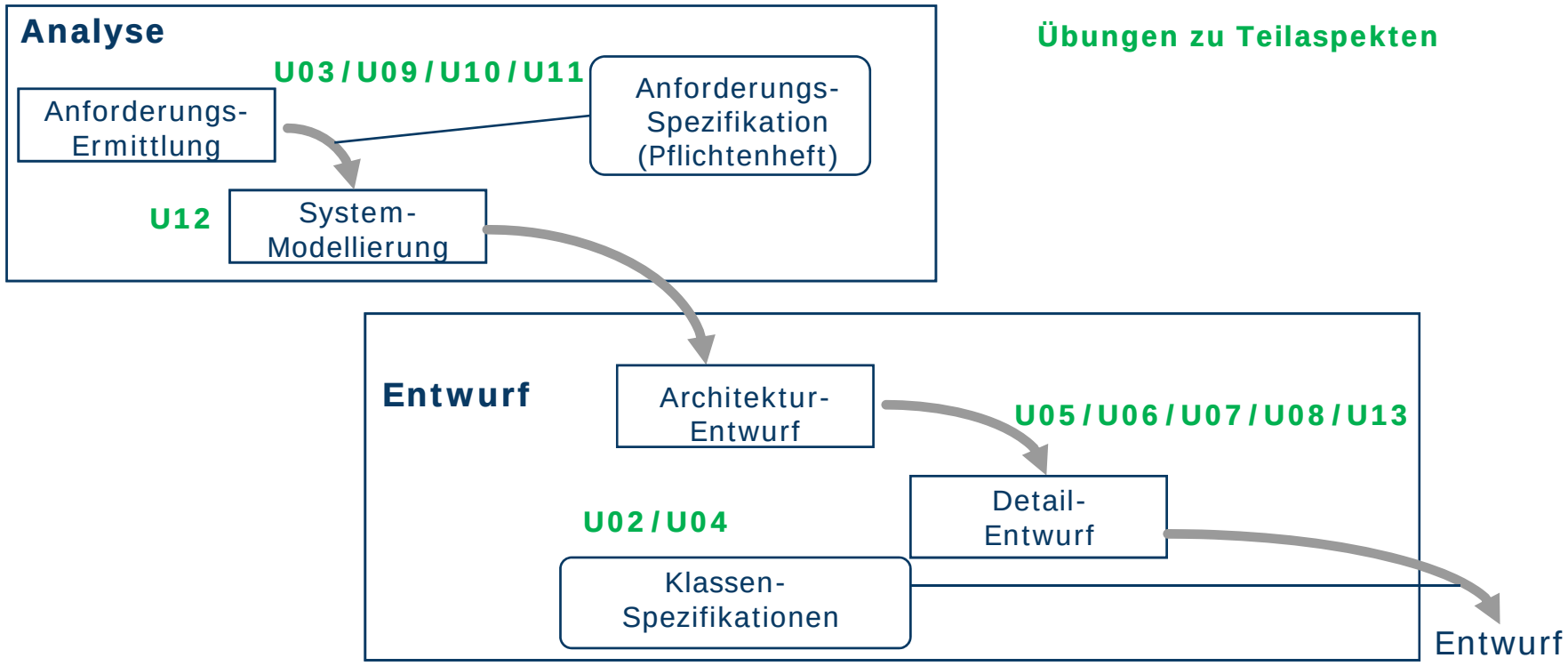
TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Modellierung mit der UML

VON DER ANALYSE (OOA) ZUM ENTWURF (OOD)

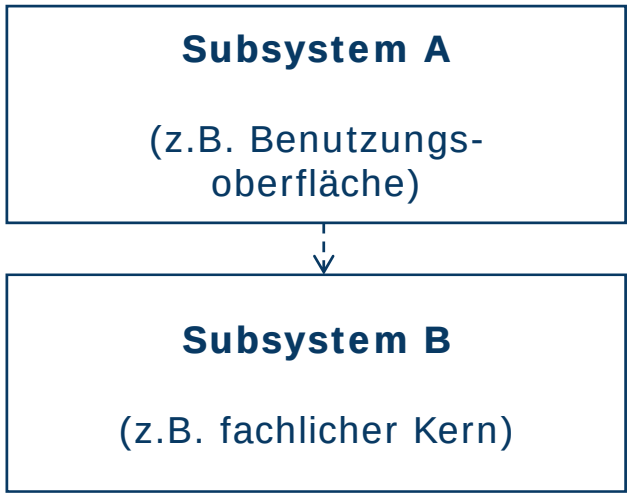
Von der Analyse zum Entwurf



Einige Kriterien für guten Entwurf

- Korrektheit
  - Erfüllung der Anforderungen
  - Wiedergabe aller Funktionen des Systemmodells
  - Sicherstellung der nichtfunktionalen Anforderungen
- Verständlichkeit
  - Gute Dokumentation
- Anpassbarkeit
- Hohe Kohäsion innerhalb der Komponenten
- Schwache Kopplung der Komponenten
- Wiederverwendung
- Stabilität und Zuverlässigkeit
- Angemessene Ressourcenverwendung

Hohe Kohäsion + Schwache Kopplung



Subsystem B darf keine Information und Funktionalität enthalten, die zum Zuständigkeitsbereich von A gehört und umgekehrt.

Es muss möglich sein, Subsystem A weitgehend auszutauschen oder zu verändern, ohne Subsystem B zu verändern.

Die meisten Änderungen von Subsystem B sollten nur relativ einfache Änderungen in Subsystem A nach sich ziehen.

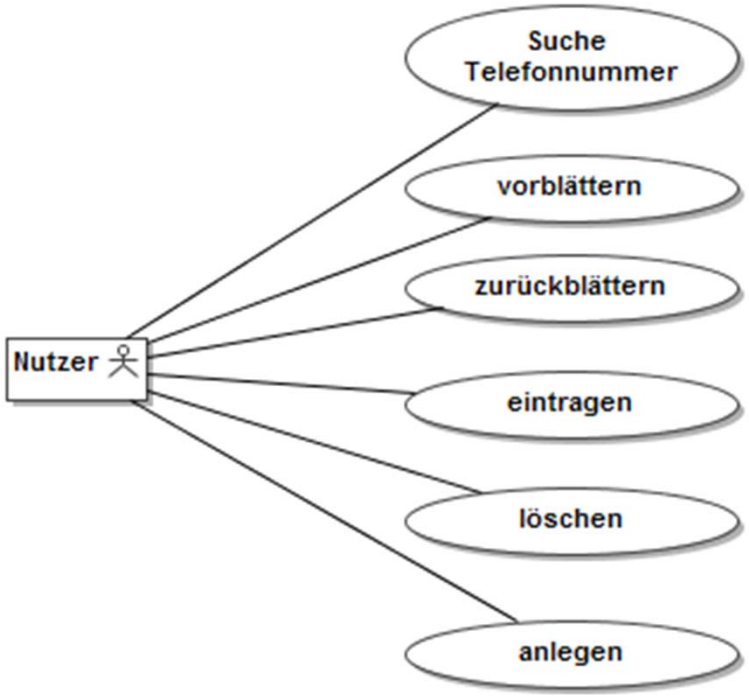
Beispiele zur konkreten technischen Realisierung siehe MVC-Architektur und Entwurfsmuster

Schnittstellen und abstrakte Klassen in Java

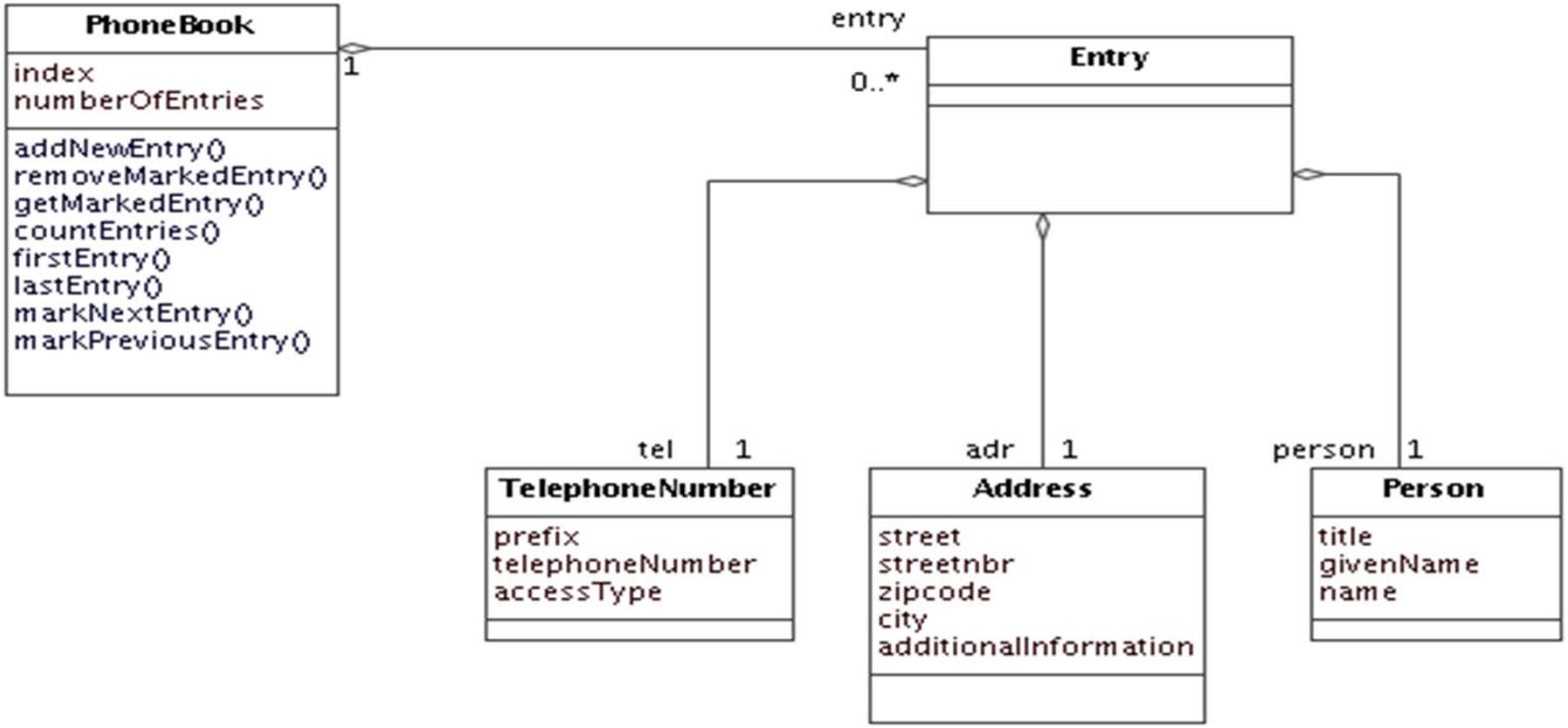
Abstrakte Klasse	Schnittstelle (Interface)
Attribute, Konstanten, Operationen	Operationen und ggfs. Konstanten
Kann Default-Verhalten festlegen	Kann KEIN Default-verhalten festlegen
Default-Verhalten kann in Unterklassen überschrieben werden	Überschreiben von Methoden ist nicht möglich
Unterklasse kann nur von einer Klasse erben	Eine Klasse kann mehrere Schnittstellen implementieren
Verwendung für Implementierungsvererbung	Verwendung für Spezifikationsvererbung

Beispiel Telefonbuch

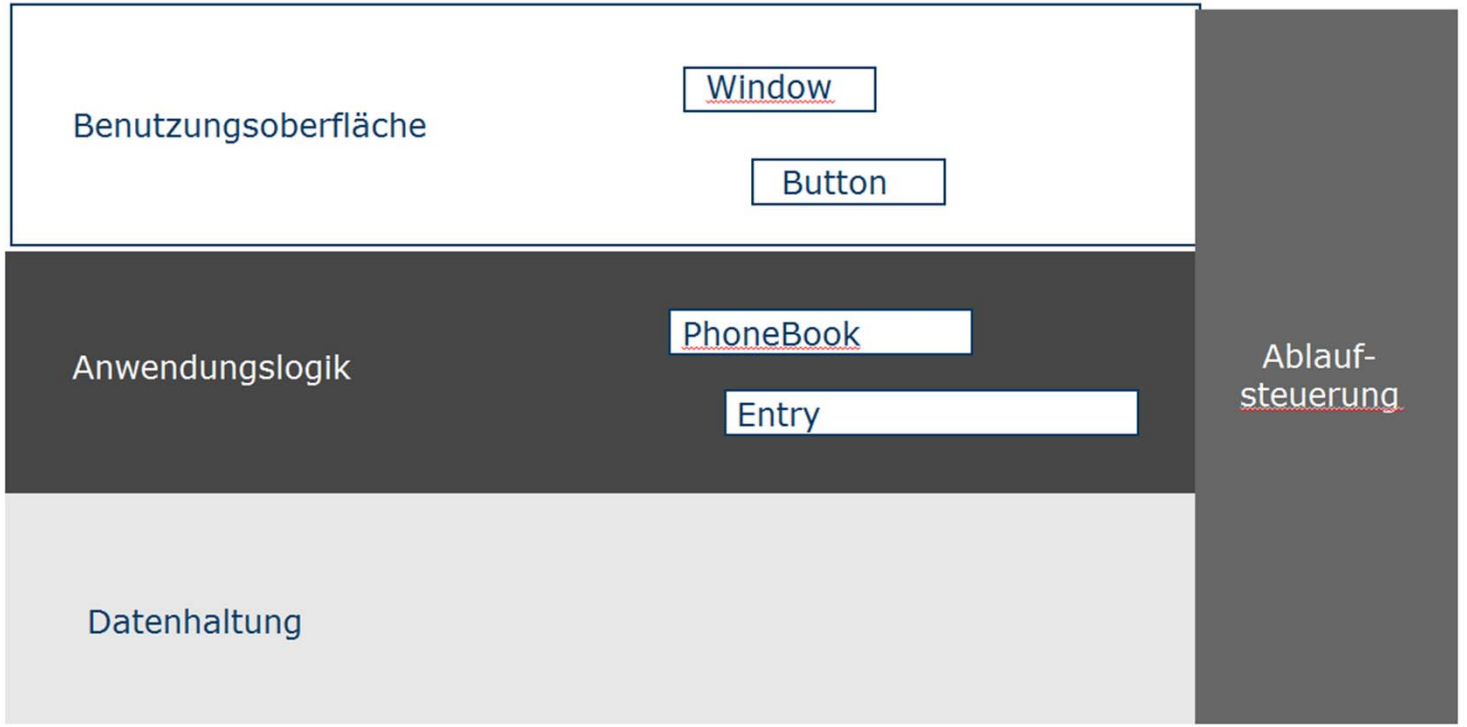
Use Case Diagramm Telefonbuch



Analysemodell des Telefonbuches

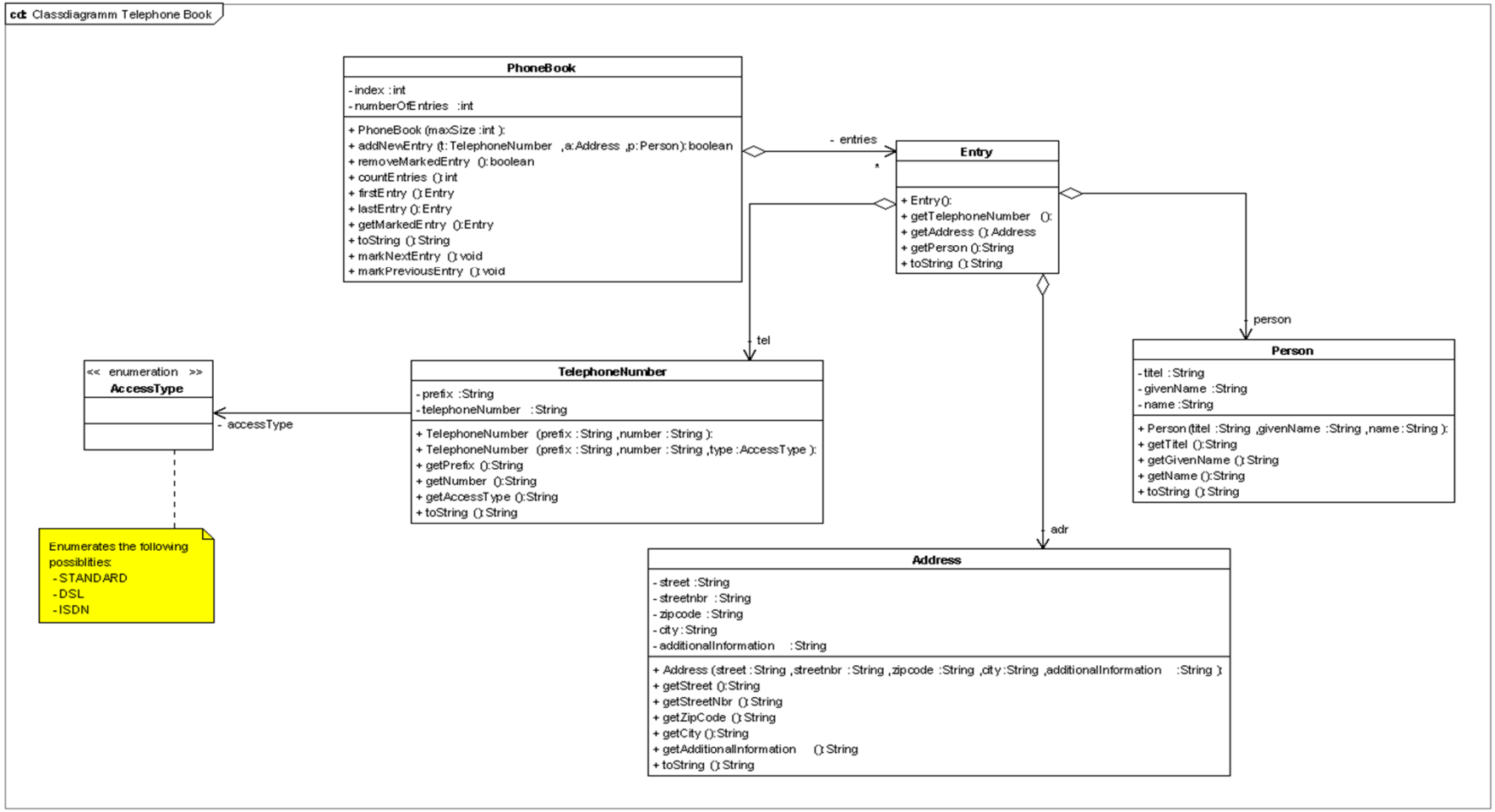


Top Level Architektur Telefonbuch (Blockdiagramm)



Top Level Architektur Telefonbuch (UML)

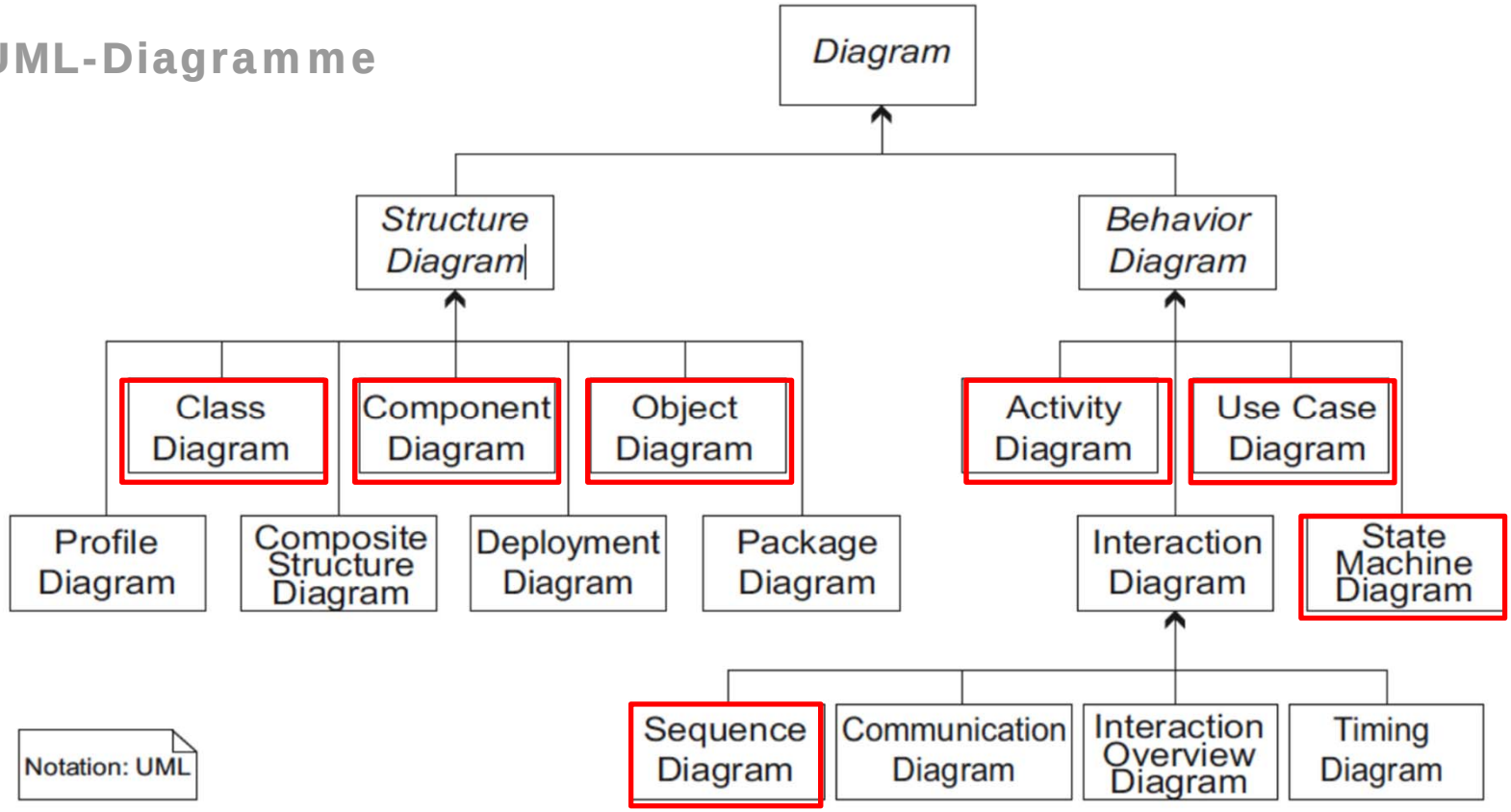
Entwurfsmodell des Telefonbuches



Artefakte in der LV Softwaretechnologie

ZUSAMMENFASSUNG

UML-Diagramme



Anwendung von Artefakten in den Phasen des SE-Lebenszyklus (ST-Übungen)	Analyse (OOA)/ aUML	Design (OOD)/ dUML	Implementierung + Test (OOP)/ jUML[1]
Anwendungsfalldiagramm (UML)	x[2]	-[3]	
Klassendiagramm (UML)	Domänenmodell/ Analyseklassen-diagramm	Entwurfsklassendiagramm mit Kollaborationen für Entwurfsmuster	
Objektdiagramm (UML)	x	x	
Zustandsdiagramm (UML) /Protokollmaschine (PSM) + Verhaltenszustandsmaschine (BSM)[4]	OLC, ULC, CSM	OLC, CSM	
Aktivitätsdiagramm (UML)	Typischerweise für Workflows/ Geschäftsabläufe	-	
Sequenzdiagramm (UML)	Typischerweise zur Modellierung von Szenarien für einen Anwendungsfall	Typischerweise zur Modellierung komplexer Methoden	
Komponentendiagramm (UML)	Kontextdiagramm Top-Level-Architektur	Verfeinerte, komponentenbasierte Softwarearchitekturen	
Blockdiagramm		-	
CRC-Karten	x (verhaltensgetriebene Analyse und Entwurf) -		
Testfalltabelle	Akzeptanztest	Unit-Test	
Java + Standardbibliothek	-	-	x
junit (3.8.1)	-	-	x

Fußnoten zur Tabelle

(1) In den Übungsaufgaben unterscheiden wir aus Komplexitätsgründen nicht zwischen dUML und jUML. Wir entwerfen oft gleich in jUML. D.h. wir gehen davon aus, dass unser Entwurf für Java-Programme gemacht ist: z.B. Angabe von Java-Datentypen, nur Einfachvererbung, Arbeit mit Interfaces usw.

(2) Ein einzelner Anwendungsfall sollte durch eine Tabelle oder weitere UML-Diagramme (meistens Aktivitätsdiagramm oder Sequenzdiagramm) beschrieben/verfeinert werden.

(3) „-„ bedeutet: wird nicht angewendet

(4) Objektlebenszyklus (OLC), Anwendungsfalllebenszyklus (ULC), Steuerungsmaschine (CSM)

Beachte …

- dass die einzelnen Artefakte verschiedene Sichten auf ein- und dasselbe Softwaresystem darstellen!
- Deshalb müssen die Zusammenhänge zwischen den Artefakten verstanden und auf deren Konsistenz geachtet werden!
- Der Übergang von aUML zu (dUML und) jUML bedeutet im Allg. Verfeinerung oder anders gesagt den Übergang vom Problemraum zum Lösungsraum.

Coding Rules

Berücksichtige Codierungsregeln in Projekten!

- Im Softwarepraktikum werden vorgegebene Coding Rules automatisiert geprüft.
- Diese Regeln wurden aus einer Vielzahl von vorgeschlagenen Regeln ausgewählt und liegen als Dokument auf dem Web (`Coding Rules ST.pdf`).

Literatur

(1) Robert C. Martin: *Clean Code: A Handbook of Agile Software Craftsmanship.* Prentice Hall, 2008

(2) Google Java Style Guide: https://google.github.io/styleguide/javaguide.html

Literatur zur Modellierung

(1) Birgit Demuth (Hrsg.): Softwaretechnologie für Einsteiger. Pearson Studium, 2. geänderte Auflage, 2014

(2) Heide Balzert: UML kompakt mit Checklisten. Spektrum Akademischer Verlag, 2001

(3) http://zugang.sophist.de/webUMLGlossar.nsf/

Klausuren und zugeordnete Dokumente liegen auf

ftp://ftp.ifsr.de/klausuren/SWT/

Ende