






www.uni-frankfurt.de

Modul: Programmierung B-PRG Grundlagen der Programmierung 1

V18 UML – 2: Diverse Diagramme

Prof. Dr. Detlef Krömker
Professur für Graphische Datenverarbeitung
Institut für Informatik
Fachbereich Informatik und Mathematik (12)

Unsere heutigen Lernziele

Weitere Elemente des Systementwurfs kennenlernen

UML in Grundzügen begreifen:

- Objektdiagramm
- Weitere Strukturdiagramme: Paketdiagramm; Abhängigkeiten
- Zustandsdiagramm
- Sequenzdiagramm

und hierzu die grafische Notationen kennenlernen.



2 Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten) Prof. Dr. Detlef Krömker




Übersicht

- **OO - Analyse**
- **OO - Design**
 - Prinzipielles Vorgehen
 - Entwicklung eines Klassenmodells
 - Klassendiagramm
 - Nachträge zum Klassendiagramm
 - Objektdiagramm
 - Weitere Strukturdiagramme: Paketdiagramm; Abhängigkeiten
 - Zustandsdiagramm
 - Sequenzdiagramm

3
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömker

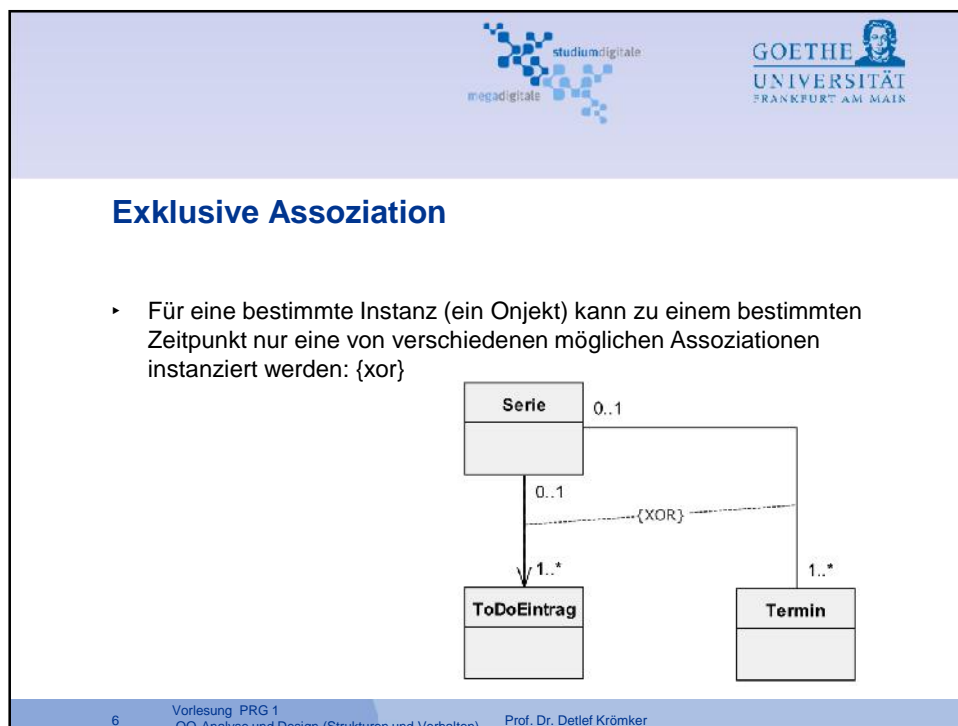
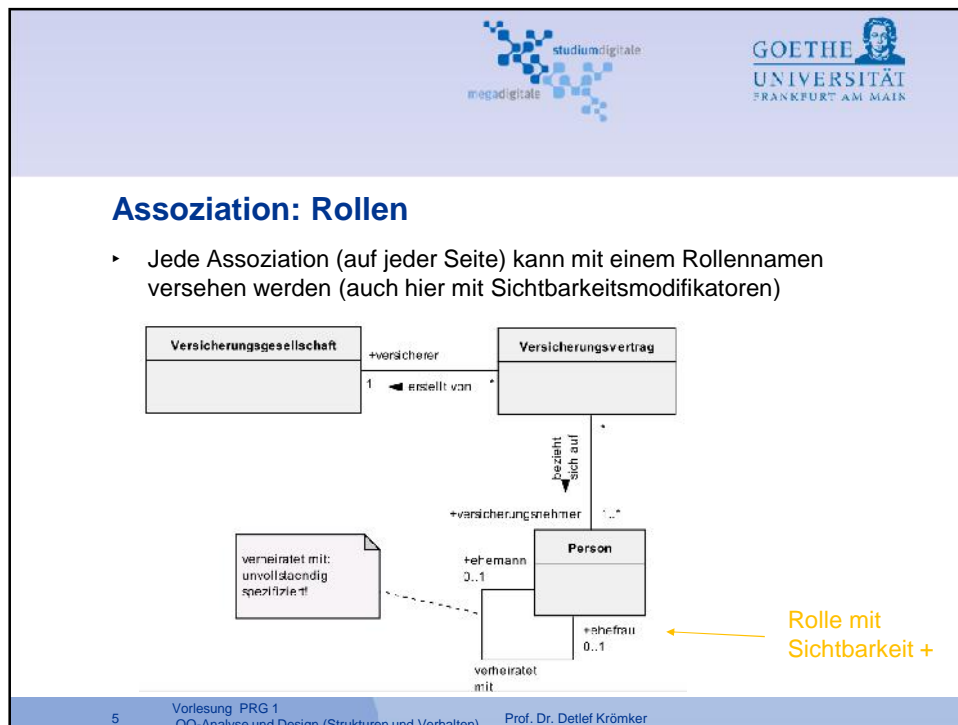



Nachtrag zum Klassendiagramm

Assoziation: Multiplizität

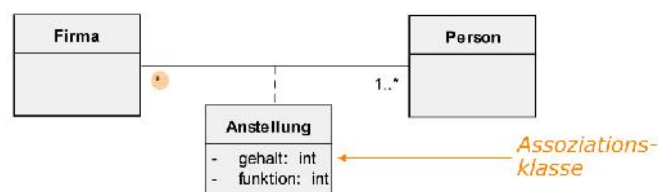
- **Bereich: "min .. max"**
- **Beliebige Anzahl: "*" (= 0..*)**
- **Aufzählung** möglicher Kardinalitäten (durch Kommas getrennt)
- **Defaultwert: 1** (wird meist nicht angegeben)
- **Beispiele**
 - genau 1: 1 (oder nichts)
 - ≥ 0 : * oder 0..*
 - 0 oder 1: 0..1 oder 0,1
 - fixe Anzahl (z.B. 3): 3
 - Bereich (z.B. ≥ 3): 3..*
 - Bereich (z.B. 3 - 6): 3..6
 - Aufzählung: 3,6,7,8,9 oder 3, 6..9

4
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömker



Assoziationsklasse (1/3)

- Eine Assoziation kann Attribute enthalten
- Bei m:n-Assoziationen mit Attributen notwendig:

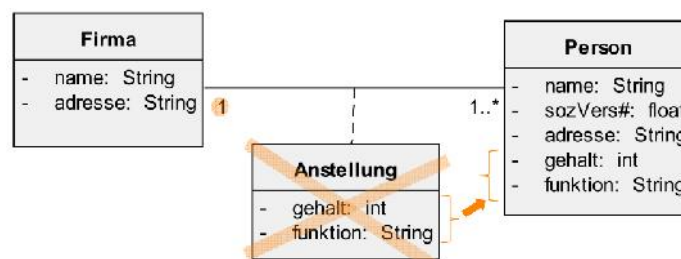


7

Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten) Prof. Dr. Detlef Krömer

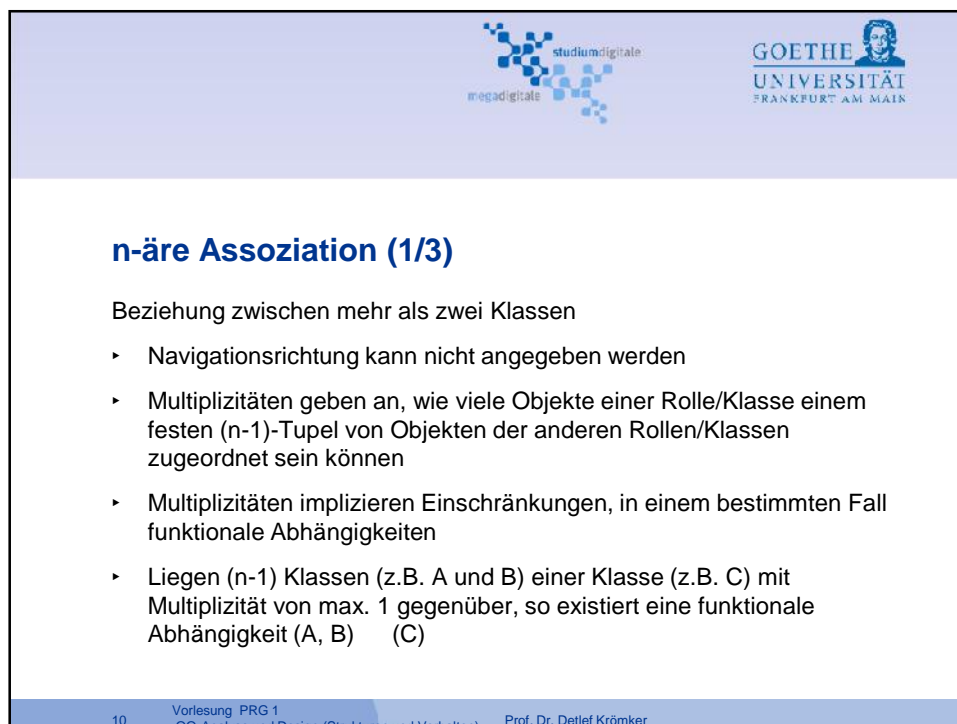
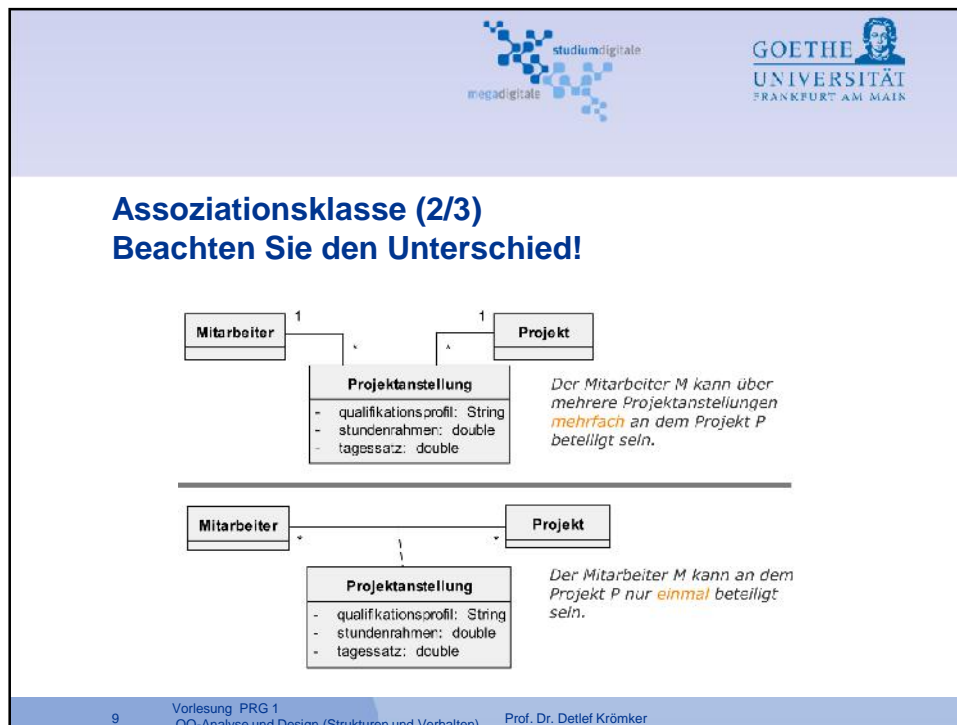
Assoziationsklasse (2/3)

- Bei 1:1 und 1:n-Assoziationen ggf. sinnvoll aus Flexibilitätsgründen (Änderung der Multiplizität möglich).



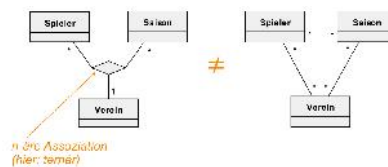
8

Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten) Prof. Dr. Detlef Krömer



n-äre Assoziation Beispiel

- (Spieler, Saison) (Verein)
Ein Spieler spielt in einer Saison bei genau einem Verein
- (Saison, Verein) × (Spieler)
In einer Saison spielen bei einem Verein mehrere Spieler
- (Verein, Spieler) × (Saison)
Ein Spieler spielt in einem Verein in mehreren Saisonen



11

Vorlesung PRG 1

OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

Anmerkungen zur Aggregation

Aggregation ist eine spezielle Form der Assoziation mit folgenden Eigenschaften:

- Transitivität:
C ist Teil von B u. B ist Teil von A → C ist Teil von A
Bsp.: Kühlung = Teil von Motor & Motor = Teil von Auto
Kühlung ist (indirekter) Teil von Auto
- Anti-Symmetrie:
B ist Teil von A → A ist nicht Teil von B
Bsp.: Motor ist Teil von Auto, Auto ist nicht Teil von Motor

UML unterscheidet zwei Arten von Aggregationen:

- Schwache Aggregation (shared aggregation)
- Starke Aggregation – Komposition (composite aggregation)

12

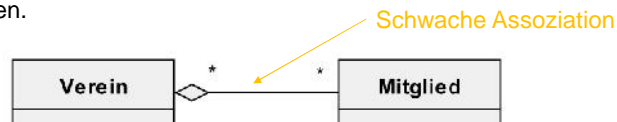
Vorlesung PRG 1

OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

Schwache Aggregation

- Schwache Zugehörigkeit der Teile, d.h. Teile sind unabhängig von ihrem Ganzen.
- Die Multiplizität des aggregierenden Endes der Beziehung (Raute) kann > 1 sein.
- Es gilt nur eingeschränkte Propagierungssemantik.
- Die zusammengesetzten Objekte bilden einen gerichteten, azyklischen Graphen.



13

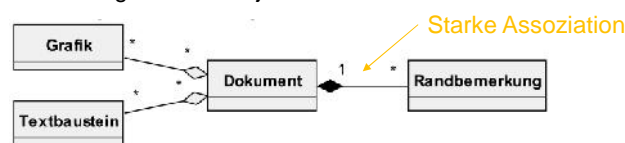
Vorlesung PRG 1

OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

Starke Aggregation (= Komposition) (1/2)

- Ein bestimmter Teil darf zu einem bestimmten Zeitpunkt in maximal einem zusammengesetzten Objekt enthalten sein
- Die Multiplizität des aggregierenden Endes der Beziehung kann (maximal) 1 sein
- Abhängigkeit der Teile vom zusammengesetzten Objekt Propagierungssemantik
- Die zusammengesetzten Objekte bilden einen Baum



14

Vorlesung PRG 1

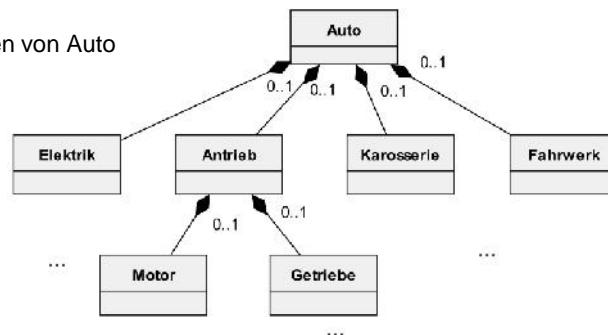
OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

Starke Komposition (2/2)

Mittels starker Aggregation kann eine Hierarchie von „Teil-von“-Beziehungen dargestellt werden (Transitivität!)

Beispiel: Baugruppen von Auto



15

Vorlesung PRG 1

OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

a. Starke Aggregation vs. b. Assoziation - Faustregeln

- ▶ **Einbettung**
 - a. Die Teile sind i.A. physisch im Kompositum enthalten
 - b. Über Assoziation verbundene Objekte werden über Referenzen realisiert
- ▶ **Sichtbarkeit**
 - a. Ein Teil ist nur für das Kompositum sichtbar
 - b. Das über eine Assoziation verbundene Objekt ist i.A. öffentlich sichtbar
- ▶ **Lebensdauer**
 - a. Das Kompositum erzeugt und löscht seine Teile
 - b. Keine Existenzabhängigkeit zwischen assoziierten Objekten
- ▶ **Kopien**
 - a. Kompositum und Teile werden kopiert
 - b. Nur die Referenzen auf assoziierte Objekte werden kopiert

16

Vorlesung PRG 1

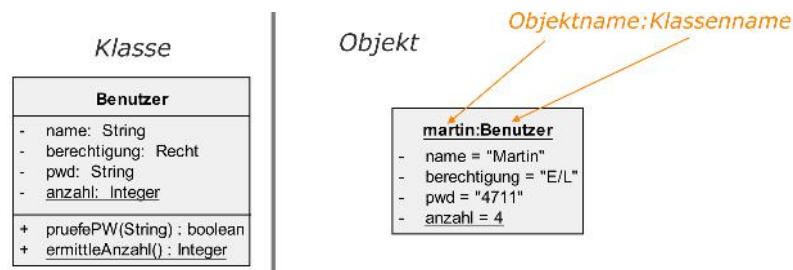
OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

Objektdiagramm

- Beschreibt den strukturellen Aspekt eines Systems auf **Instanzebene** in Form von Objekten und Links
- Momentaufnahme (snapshot) des Systems – konkretes Szenario
- Ausprägung zu einem Klassendiagramm
- Eigentlich eine »Instanzspezifikation«
- Prinzipiell kann jede Diagrammart auf Instanzebene modelliert werden

Beispiel:



Objektdiagramm: Basiskonzepte

- Instanz einer Klasse: Objekt
- Instanz einer Assoziation: Link
- Instanz eines Datentyps: Wert

Einheitliche Notationskonventionen

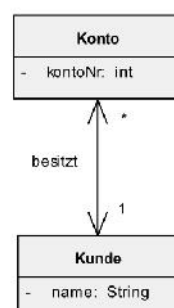
- Gleiches Notationselement wie auf Klassenebene benutzen
- Unterstreichen (bei Links optional)

Objektdiagramm muss nicht vollständig sein

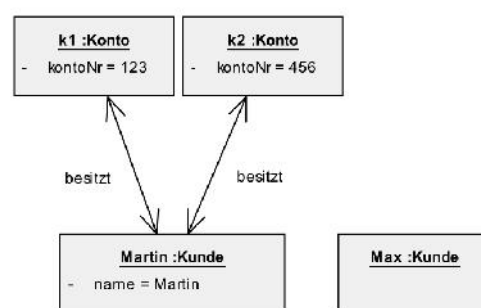
- z.B. können Werte benötigter Attribute fehlen, aber auch Instanzspezifikation abstrakter Klassen modelliert werden

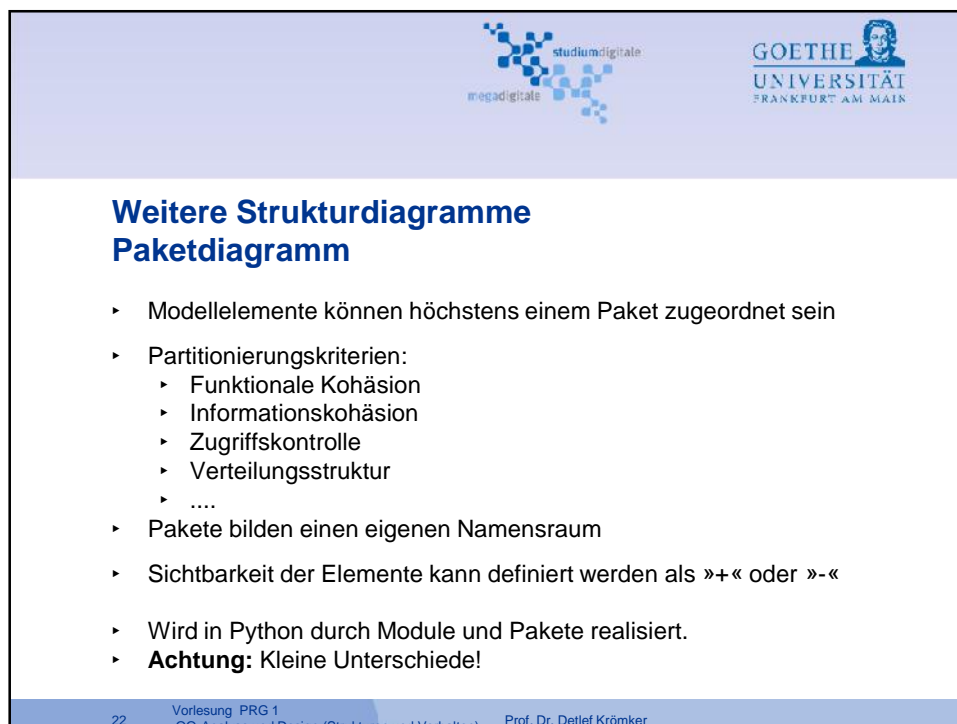
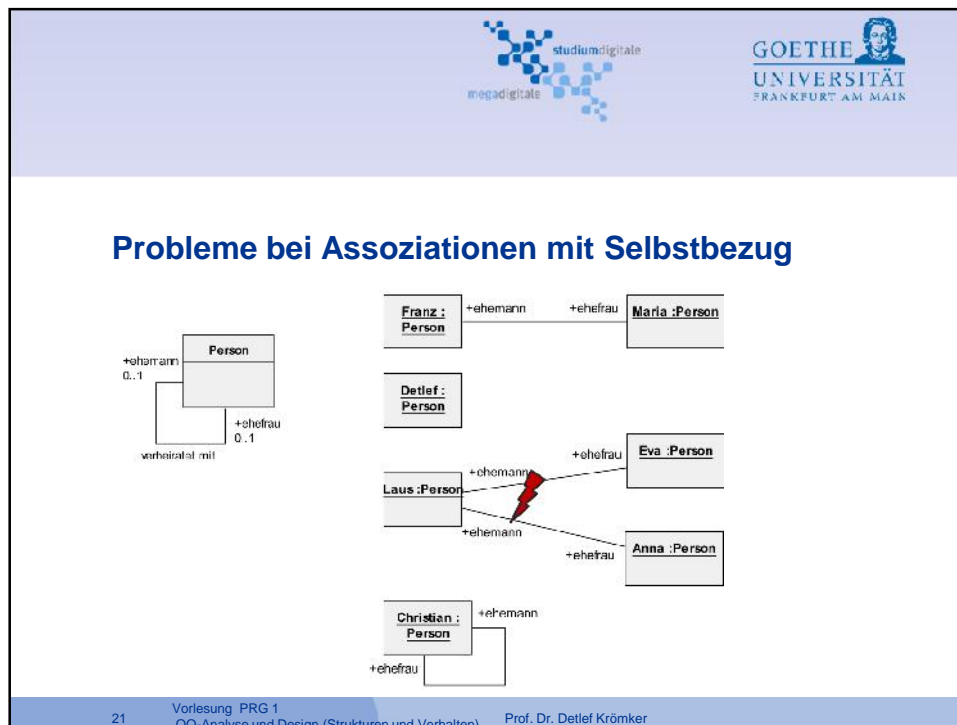
Beispiel

Klassendiagramm



Objektdiagramm

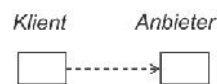




Abhängigkeit

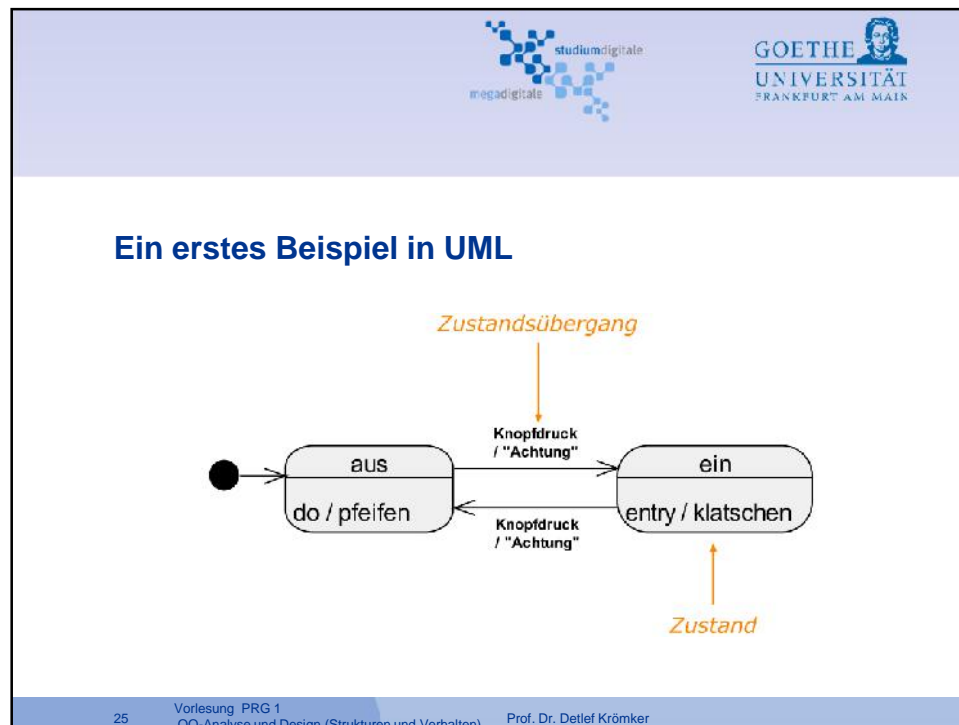
Eine Abhängigkeit stellt eine allgemeine Kopplung zweier Modellelemente (Klassen, Interfaces, Pakete usw.) dar


Änderungen in einem Element (Anbieter) können Änderungen im abhängigen Element (Klient) nach sich ziehen



Übersicht

- **OO - Analyse**
- **OO - Design**
 - Prinzipielles Vorgehen
 - Entwicklung eines Klassenmodells
 - Klassendiagramm
 - Nachträge zum Klassendiagramm
 - Objektdiagramm
 - Weitere Strukturiagramme: Paketdiagramm; Abhängigkeiten
 - **Zustandsdiagramm**
 - Sequenzdiagramm






Begriffsbestimmungen



Ein Zustandsdiagramm (State Machine Diagram) beschreibt die möglichen Folgen von Zuständen eines Modell-Elements, i.A. eines Objekts einer bestimmten Klasse

- Während seines Lebenslaufs (Erzeugung bis Destruktion)
- Während der Ausführung einer Operation oder Interaktion

Modelliert werden:


- Die Zustände, in denen sich die Objekte einer Klasse befinden können
- Die möglichen Zustandsübergänge (Transitionen) von einem Zustand zum anderen.
- Die Ereignisse, die Transitionen auslösen.
- Aktivitäten, die in Zuständen bzw. im Zuge von Transitionen ausgeführt werden.

27
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer

Beispiel: Digitaluhr

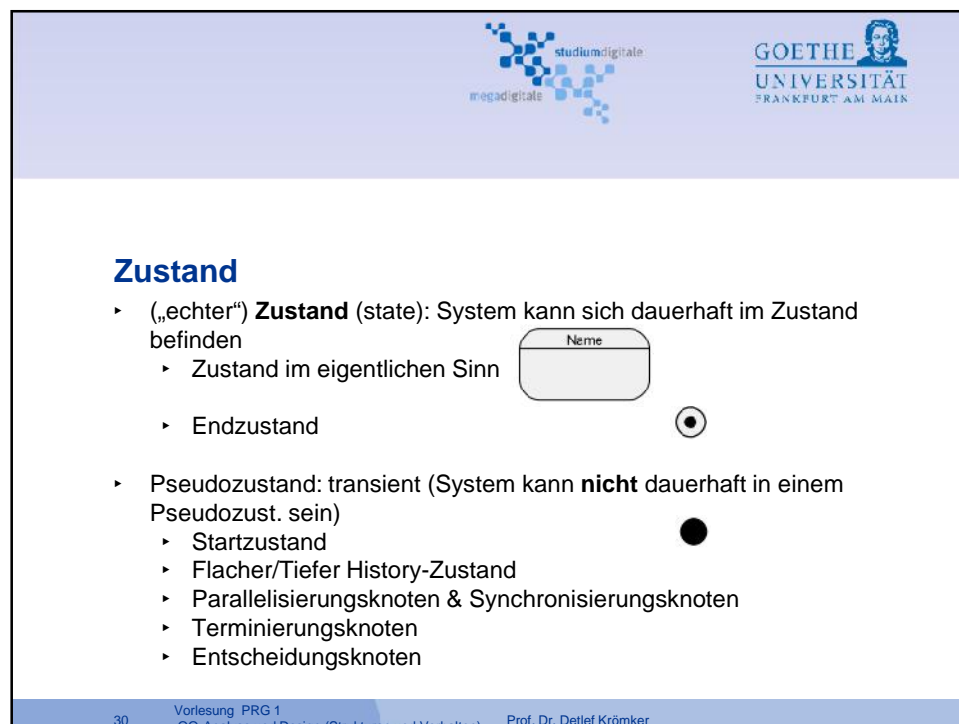
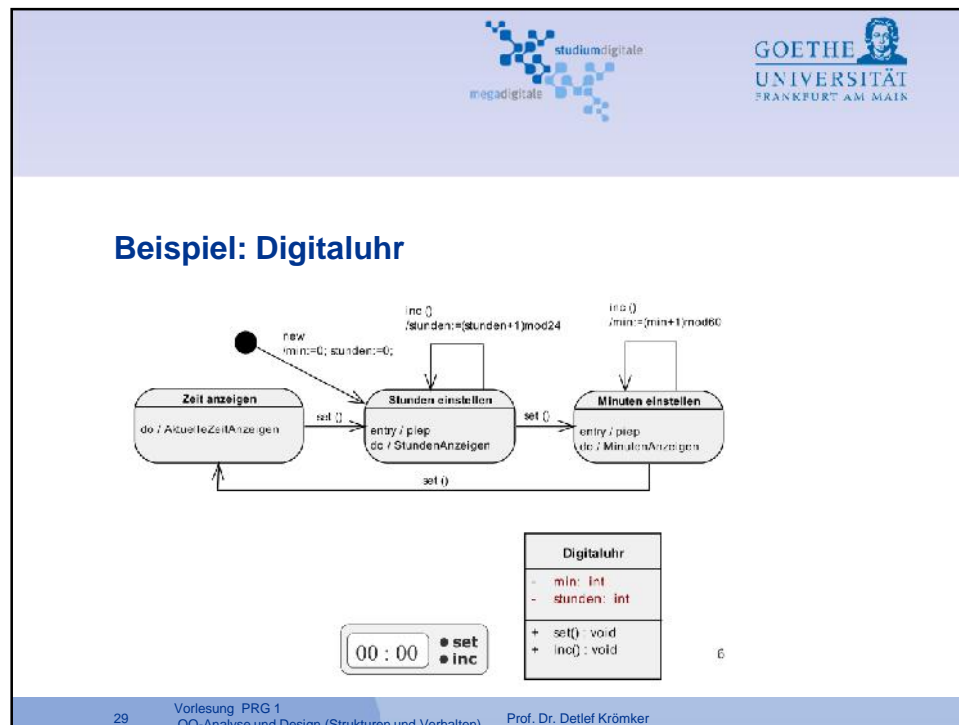
- Modelliert werden sollen die Zustände, die eine Digitaluhr beim Stellen der Uhr einnehmen kann.
- Die Uhr kann 3 Zustände einnehmen:
 - Zeit anzeigen
 - Stunden einstellen
 - Minuten einstellen



Digitaluhr	
-	min: int
-	stunden: int
+	set(): void
+	inc(): void

- Durch die Betätigung des Einstellungsknopfes "set" wird von "Zeit anzeigen" in "Stunden einstellen" gewechselt, von "Stunden einstellen" in "Minuten einstellen" und schließlich von "Minuten einstellen" wieder in "Zeit anzeigen".
- Wird in "Stunden einstellen" gewechselt, piepst die Uhr und zeigt die Stunden an. Durch die Betätigung des inc-Buttons, wird die Stundenanzahl um 1 erhöht.
- "Minuten einstellen" funktioniert analog.
- Am Anfang befindet sich die Uhr im Zustand "Stunden einstellen".

28
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer






Aktivitäten innerhalb eines Zustands

entry / aktivität
Wird beim Eingang in den Zustand ausgeführt

exit / aktivität
Wird beim Verlassen des Zustands ausgeführt

do / aktivität
Wird ausgeführt, Parameter sind erlaubt

event / aktivität
Aktivität behandelt Ereignis innerhalb des Zustands.
Wird ausgeführt, wenn sich das System in dem Zustand befindet und das Ereignis eintritt.

Zustand Z

entry / Aktivitaet(...)

do / Aktivitaet(...)

event / Aktivitaet(...)



exit / Aktivitaet(...)

Stunden einstellen

entry / beep

do / display hours

31
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer

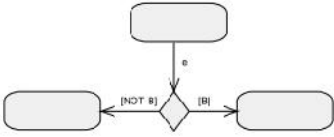
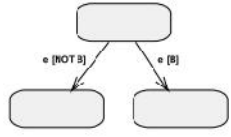



Zustandsübergang (1/2)

Ein Zustandsübergang (Transition) erfolgt, wenn

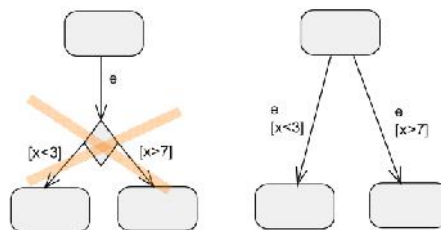
- **das Ereignis eintritt:** Eine eventuell noch andauernde Aktivität im Vorzustand wird unterbrochen!
- und**
- **die Bedingung (guard) erfüllt ist:** Bei Nicht-Erfüllung geht das nicht »konsumierte« Ereignis verloren => wenn die Bedingung erst zu einem späteren Zeitpunkt erfüllt wird, kann die Transition ohne neuerliches Ereignis nicht durchgeführt werden

Durch entsprechende Bedingungen können Entscheidungsbäume modelliert werden.

32
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer

Zustandsübergang (2/2)

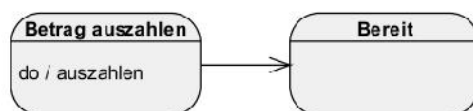


33

Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten) Prof. Dr. Detlef Krömer

Default Werte für Zustandsübergänge

- ▶ Fehlendes Ereignis entspricht dem Ereignis »Aktivität ist abgeschlossen«
- ▶ Fehlende Bedingung entspricht der Bedingung [true]
- ▶ Beispiel: Bankautomat



34

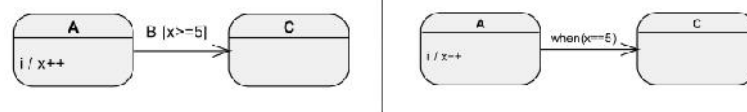
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten) Prof. Dr. Detlef Krömer



Zustandsübergang: Ereignistypen (1/2)

- CallEvent: Empfang einer Nachricht (Operationsaufruf)
 - Bsp.: stornieren(), kollidiertMit(Termin)
- SignalEvent: Empfang eines Signals
 - Bsp.: right-mouse-button-down, ok-Taste-gedrückt
- ChangeEvent: Eine Bedingung wird wahr
 - Bsp.: when($x < y$), when($a = 1$), when(terminBestaetigt)
- TimeEvent: Zeitablauf oder Zeitpunkt
 - Bsp.: after(5 sec.), when(date=31.01.2008)




Unterschied ChangeEvent und Bedingung

- ChangeEvent:
 - Bedingung wird permanent geprüft.
 - wenn Bedingung wahr ist, kann zugehöriger Zustandsübergang ausgelöst werden (falls nicht durch zugehörige Überwachungsbedingung blockiert).
- Bedingung:
 - wird nur geprüft, wenn zugeordnetes Ereignis eintritt.
 - kann selbst keinen Zustandsübergang auslösen.





Start- und Endzustand, Terminierungsknoten


- ▶ **Startzustand** 
 - ▶ "Beginn" des Zustandsdiagramms
 - ▶ keine eingehenden Transitionen
 - ▶ genau eine ausgehende Transition
 - ▶ wird sofort ausgelöst, wenn sich das System im Startzustand befindet
 - ▶ keine Bedingungen und Ereignisse (Ausnahme: Ereignis zur Erzeugung des betrachteten Objekts)
 - ▶ Angabe von Aktivitäten ist erlaubt
- ▶ **Endzustand** 
 - ▶ keine ausgehenden Transitionen
 - ▶ kein Pseudozustand!
- ▶ **Terminierungsknoten** 
 - ▶ Objekt, dessen Verhalten modelliert wird, hört auf zu existieren

37
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer





Zustandsübergang: Innere Transitionen

- ▶ Werden wie »äußere« Transitionen von Ereignissen ausgelöst, verlassen aber den aktuellen Zustand nicht.
- ▶ Äquivalent zu Selbsttransition, sofern keine entry / exit-Aktivitäten vorhanden.



- ▶ Gleiche Aktivitäten können in den Zustand hineingezogen werden:

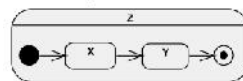


38
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer

Komplexe Zustände

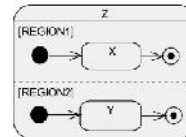
= Zustände, die aus mehreren Subzuständen zusammengesetzt sind

- geschachteltes Zustandsdiagramm
- Die Subzustände sind disjunkt, d.h. genau ein Subzustand ist aktiv, wenn der komplexe Zustand aktiv ist





Zu einem Zeitpunkt kann nur X ODER Y aktiv sein!


- Teilung des Superzustandes in mehrere Regionen:
→ die Subzustände sind nebenläufig, gleichzeitig aktiv
Z = „orthogonaler Zustand“




Zu einem Zeitpunkt sind X UND Y aktiv!

Historischer Zustand

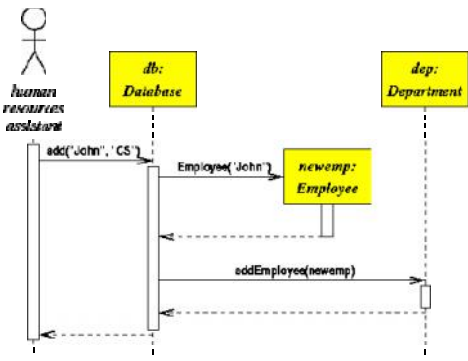
- Historische Zustände können sich jenen internen Zustand in einem komplexen Zustand merken, von dem die letzte Transition (vor einer Unterbrechung) ausgegangen ist.
- Zu einem späteren Zeitpunkt kann zu diesem Zustand über Transitionen aus übergeordneten Zuständen zurückgekehrt werden
 - alle Entry-Aktivitäten werden wiederum ausgeführt
- **Flacher History-Zustand** merkt sich eine Ebene 
- Über einen **tiefen History-Zustand** »H*« werden alle Zustände über die gesamte Schachtelungstiefe hinweg gesichert. 






Sequenz-Diagramme


New Employee:



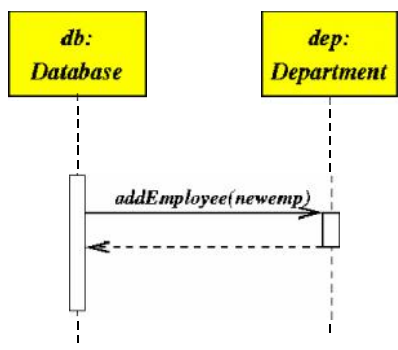
- Sequenz-Diagramme zeigen den **Kontrollfluss** für ausgewählte Szenarien.
- Die Szenarien können unter anderem von den Use-Cases abgeleitet werden.
- Sie demonstrieren wie Akteure und Klassen miteinander in einer sequentiellen Form operieren.

41
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer





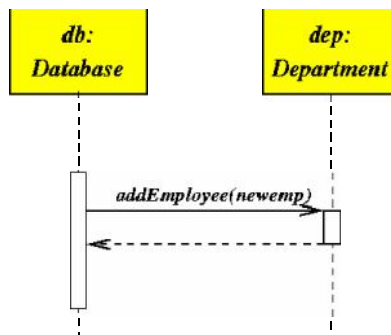
Methodenaufrufe in einem Sequenz-Diagramm



- Die Zeitachse verläuft **von oben nach unten**.
- Jedes an einem Szenario beteiligte Objekt wird durch ein Rechteck dargestellt, das die **Klassenbezeichnung** und optional einen Variablennamen enthält.
- Die Zeiträume, zu denen ein Objekt nicht aktiv ist, werden mit einer gestrichelten Linie dargestellt.

42
Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)
Prof. Dr. Detlef Krömer

Methodenaufrufe in einem Sequenz-Diagramm



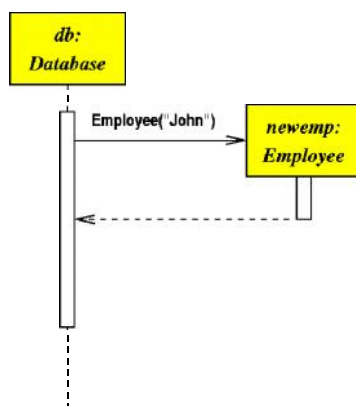
- Ein Objekt wird dann durch einen Methodenaufruf aktiv. Der Zeitraum, zu dem sich eine Methode auf dem Stack befindet, wird durch langgezogenes Rechteck dargestellt.
- Der Methodenaufruf selbst wird durch einen Pfeil dargestellt, der mit dem Aufruf selbst beschriftet wird.
- Die Rückkehr kann entweder weggelassen werden oder sollte durch eine gestrichelte Linie markiert werden.

43

Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

Konstruktoren in einem Sequenz-Diagramm



- Objekte, die erst im Laufe des Szenarios durch einen Konstruktor erzeugt werden, werden rechts neben dem Pfeil plaziert.
- Ganz oben stehen nur die Objekte, die zu Beginn des Szenarios bereits existieren.
- Da ein neu erzeugtes Objekt sofort aktiv ist, gibt es keine gestrichelte Linie zwischen dem Objekt und der ersten durch ein weißes Rechteck dargestellten Aktivitätsphase.

44

Vorlesung PRG 1
OO-Analyse und Design (Strukturen und Verhalten)

Prof. Dr. Detlef Krömer

UML: Es gibt noch Aktivitätsdiagramme

- Fokus des Aktivitätsdiagramms: prozedurale Verarbeitungsaspekte
Spezifikation von Kontroll- und/oder Datenfluss zwischen
Arbeitsschritten (Aktionen) zur Realisierung einer Aktivität
- **Aktivitätsdiagramm in UML2:**
 - ablauforientierte Sprachkonzepte
 - basierend u.a. auf Petri-Netzen
- Sprachkonzepte und Notationsvarianten decken ein breites
Anwendungsgebiet ab.
 - Modellierung objektorientierter und nichtobjektorientierter Systeme
wird gleichermaßen unterstützt
 - Neben vorgeschlagener grafischer Notation sind auch beliebige
andere Notationen (z.B. Pseudocode) erlaubt.

Ausblick

- *Versprochen: Ich update noch die Quizzes. Sorry.*
- *Die Hausübungsaufgaben für EPR 06 und PRG
werden bis kommenden Mittwoch nachgeliefert.*
- *Wir sehen uns wieder am **Freitag, 12. Januar 2018.***



... Frohe Festtage und einen guten Rutsch

und, herzlichen Dank für Ihre Aufmerksamkeit!