

Einführung in die Informatik 2

– Objektorientierung –

Sven Kosub

AG Algorithmik/Theorie komplexer Systeme
Universität Konstanz

E 202 | Sven.Kosub@uni-konstanz.de | Sprechstunde: nach Vereinbarung

Sommersemester 2010

1 Analyse

- Untersuchung der Realität auf Objekte und Objektbeziehungen
- Erstellung eines objektorientierten Modells der Realität
- Erstellung eines Nutzungsprofils: Welche Funktionalität wird wem zur Verfügung gestellt? Was passiert mit Objekten und warum?

2 Entwurf

- Übertragung des objektorientierten Modells auf Software-Architektur
- Einbeziehung programmiertechnischer Notwendigkeiten

3 Implementierung

- Konkretisierung der Software-Architektur in Programm
- Repräsentierung von Objektzuständen in Datenstrukturen
- Realisierung und Ausprogrammierung der Objektfunktionalitäten als Algorithmen (Methoden)

Objekt:

- gedankliche oder reale Einheit in Umwelt oder Software
- beschrieben durch **Zustand** und **Funktionalität**

Name
Zustand
Funktionalität

Fernsehapparate:

TV2000
EinAus z; Seriennummer s; Kanal k; Lautstärke l;
ein(); aus(); wähle_Kanal(); wähle_Lautstärke(); get_Seriennummer();

- Fernsehapparat „TV2000“
- Zustand setzt sich zusammen aus Werten von Zustandsvariablen
- Funktionalität setzt sich zusammen aus Methoden

(Objekt-)Klasse

- Zusammenfassung von Objekten mit gleichen Zustandsvariablen und Methoden
- Objekte einer Klasse durch Werte der Zustandsvariablen beschrieben
- modelliert einen bestimmten **Typ** von Objekten

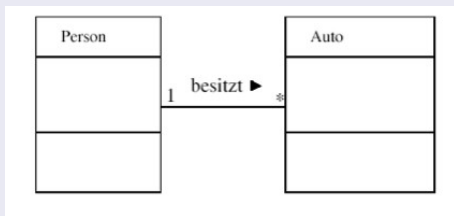
Fernsehapparate (Forts.):

- Klasse: Fernsehapparate vom Typ „TV2000“
- Objekt: Fernsehapparat bestimmtes Gerät mit konkreter, eindeutiger Seriennummer

Beachte: graphische Repräsentierung von Objekten beschreibt Klassen

Objektbeziehungen

- Objektbeziehungen modellieren Interaktion von Objekten
 - objektorientierte Software-Systeme = Objekte + Objektbeziehungen
 - **graphische Repräsentierung:** Linien mit Beschriftung (Annotation)
- Objekte der Klasse „Person“ und Objekte der Klasse „Auto“ stehen in gerichteter Beziehung „besitzt“



- Person darf beliebig viele Autos besitzen (Beschriftung „*“)
- Auto darf nur von einer Person besessen werden (Beschriftung „1“)

Klassifikation von Objektbeziehungen:

- **Nachrichtenaustausch**
(*message passing*)
- **Methodenaufruf**
(*method call, client-server*)
- **Einschluss**
(*containment, „has-a“*)
- **Vererbung**
(*inheritance, subtype, „is-a“*)

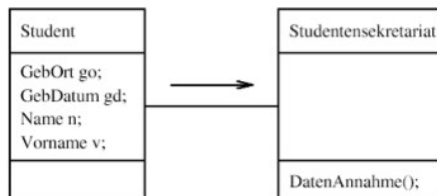
verhaltensbezogene
Beziehungen

strukturelle Beziehungen

Objektbeziehungen: Nachrichtenaustausch

Austausch von Nachrichten zwischen Objekten (*message passing*):

- Methode des Absenders schickt Nachricht an Empfänger
- Methode des Empfängers nimmt Nachricht an und verarbeitet sie
- **graphische Repräsentierung:** Pfeil zwischen Objekten



Objekt der Klasse „Student“ schickt die Personaldaten einem Objekt der Klasse „Studentensekretariat“

Beachte: Nachrichtenaustausch erfolgt **asynchron**

- Absenden und Empfangen geschieht unabhängig voneinander
- Absender kann nach Absenden sofort weiter arbeiten

Objektbeziehungen: Nachrichtenaustausch

Interaktionsdiagramme:

- Modellierung komplexer Informationsflüsse
- Informationsfluss zwischen zwei Objekten ist Folge ausgetauschter Nachrichten
- Nachrichten werden durchnummeriert in zeitlicher Abfolge

„Anmeldung mit Mahnung“



Grundschemata von Client-Server-Beziehungen:

- ein Objekt (Kunde, Auftraggeber, Client) fordert Dienstleistung an
- anderes Objekt (Dienstleister, Lieferant, Server) erbringt Dienstleistung

Methodenaufruf als Dienstleistung:

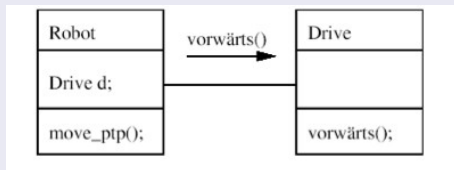
- Client ruft Methode des Servers (mit geeigneten Parametern) auf
- Server führt Methode aus und liefert Ergebnis zurück
- **graphische Repräsentierung:** Pfeil mit geschlossener Spitze

Beachte: Methodenaufruf erfolgt **synchron**

- Client und Server im Gleichklang bei Methodenaufruf
- Client wartet, bis Server Aufruf akzeptiert und Ergebnis liefert

Objektbeziehungen: Methodenaufruf

- „Roboter“ gibt „Antrieb“ einen Auftrag durch Methodenaufruf „vorwärts()“



- Methodenaufruf in Java:**

Client-Objekt ruft auf einem Server-Objekt `d` der Klasse `Drive` eine Methode auf

...

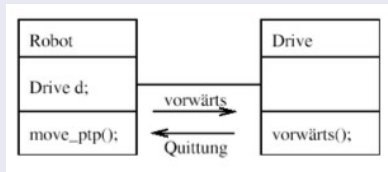
```
d.vorwärts();
```

...

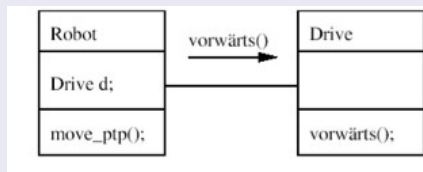
Objektbeziehungen: Methodenaufruf

Methodenaufruf als spezielle Interaktion zwischen Objekten

„Roboter“ interagiert mit einem seiner „Antriebe“



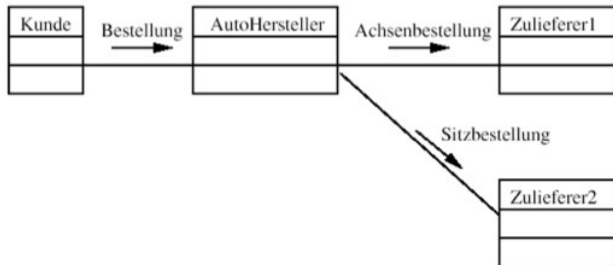
Modellierung als Nachrichtenaustausch
(asynchrone Kommunikation)



Modellierung als Methodenaufruf
(synchrone Kommunikation)

Objektbeziehungen: Methodenaufruf

Objekt kann sowohl Client als auch Server sein

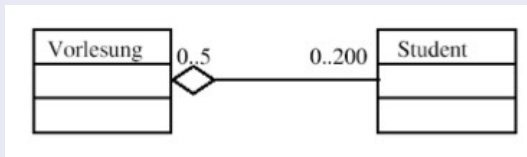


Objektbeziehungen: Einschluss

Einschlussbeziehung zwischen Objektklassen:

- Objekte der einen Klasse schließen Objekte der anderen Klasse ein
- „hat“-Beziehung (*has-a*), da Objekt mehrere Teilobjekte besitzt
- modelliert Aggregation von Teilobjekten in Objekt
- **graphische Repräsentierung:** Raute am umfassenden Ende der Linie

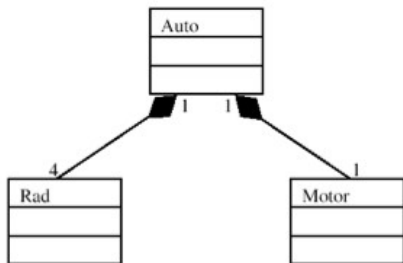
- Vorlesung „hat“ 0,1, ..., 200 Studenten
- Studenten nehmen an 0, ..., 5 Vorlesungen teil



Objektbeziehungen: Einschluss

Komposition als spezielle Einschluss-Beziehung:

- jedes Teilobjekt ist Attribut des umfassenden Objektes und Teil seines Zustandes
- **graphische Repräsentierung:** schwarz gefüllte Raute



- ein Auto „hat“ 4 Räder

- **Komposition in Java:**

```
class Auto {  
    Rad vr, vl, hr, hl;  
    Motor m;  
    ...  
    m.ein();  
    ...  
}
```

Objektbeziehungen: Vererbung

Subtyp-Beziehung zwischen zwei Klassen A und B

- A ist **Subtyp** von B , falls A alle Eigenschaften von B besitzt, d.h.

$$\text{Variablen}_B \subseteq \text{Variablen}_A \quad \text{und} \quad \text{Methoden}_B \subseteq \text{Methoden}_A$$

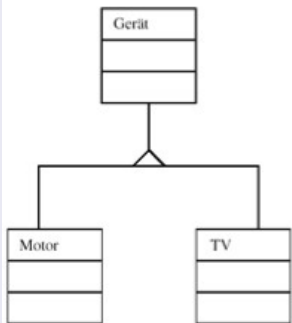
Vererbungsprinzip:

- Subtyp erbt Eigenschaften des Supertyps
- Subtyp durch Hinzunahme von Eigenschaften spezieller als Supertyp
- Supertyp durch Weglassen von Eigenschaften allgemeiner als Subtyp

Vorteil:

- Vermeidung von Replikation gleicher Variablen und Methoden in ähnlichen Typen durch Zusammenfassung zu gemeinsamem Supertyp

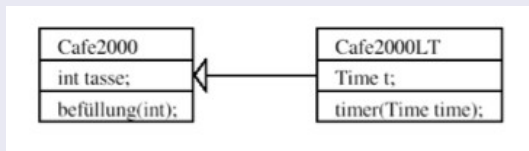
- **graphische Repräsentierung:** Pfeil mit breiter hohler Spitze



- Zustandsvariable „Seriennummer“ kommt in „Motor“ und „TV“ vor
- wird nur im gemeinsamen Supertyp „Gerät“ geführt und von dort vererbt

Objektbeziehungen: Vererbung

- vollautomatische Kaffeemaschine vom Typ „Cafe2000“ hat eine Funktion zur Befüllung von Kaffee und Wasser bis zu 12 Tassen
- Luxusmodell vom Typ „Cafe2000LT“ hat zusätzlich noch eine Timer-Funktion zur Eingabe der gewünschten Startzeit



- **Vererbung in Java:**

```
class Cafe2000LT extends Cafe2000 {
    Time t;
    void timer(Time time);
    ...
}
```

• objektorientierte Analyse

- umgangssprachliche Beschreibung auf Objekte, ihre Attribute, Funktionalität und Beziehungen hin untersuchen
- Substantive geben Hinweise auf Objekte und Zustände,
- Verben geben Hinweise auf Funktionalität,
- Aussagen wie „hat ein“ oder „ist ein“ deuten auf Objektbeziehungen

• objektorientierter Entwurf

- bilde Strukturen auf jeweilige Programmiersprache ab
- ergänze um softwaretechnische Notwendigkeiten und Hilfskonstrukte (z.B. höhere Datenstrukturen)

• objektorientierte Implementierung

demnächst!