Folien zum Lehrmodul

Objektorientierte Modellierung

Lernziele:

- UML-Notationen für Analyseklassendiagramme kennen und benutzen können (Übungen!)
- Unterschiede zwischen und Gemeinsamkeiten von ER-Modellen und Klassendiagrammen kennen
- zusätzliche Konzepte gegenüber der ER-Modellierung verstanden haben (Aggregation, Composition, Operationen, Pakete)

Inhaltsverzeichnis

1 Übersicht und Einordnung

Ziel: Grundlagen der objektorientierten Analyse (OOA), insb. (Analyse-) Klassendiagramme

Notation: UML *Unified Modeling Language* (OMG-Standard)

- vereinheitlicht mehrere frühere OOA-Methoden

Ursprünge der OOA-Methoden:

- Datenmodellierung mit ER-Diagrammen
- Datenkapselung, Klassenhierarchien von den oo-Programmiersprachen

OOA-Klassendiagramm $\approx$ erweitertes ER-Diagr. (andere Syntax)

- Daten-Modellierungsregeln i.w. wie bei ER-Diagrammen
- Umsetzung in Tabellen wie bei ER-Diagrammen
- Erweiterung: Operationen $\rightarrow$ Klassendiagramme (zzgl. Datenlexikon und weiterer Dokumente) sind Daten- und Funktionsmodelle

für die OOA relevante Diagrammtypen der UML:

- **Klassendiagramm** (*class diagram*)
- **Anwendungsfalldiagramm** (*use case diagram*), auch **Geschäftsprozeßdiagramm** genannt.
- Verhaltensdiagramme:
  - **Zustandsübergangsdiagramm** (*statechart diagram*)
  - **Aktivitätsdiagramm** (*activity diagram*)
  - **Interaktionsdiagramme** (*interaction diagrams*):
    - **Sequenzdiagramm** (*sequence diagram*)
    - **Kooperationsdiagramm** (*collaboration diagram*)

verschärfte Frage der Konsistenz der Dokumente + deren Semantik

UML ist ohne relativ leistungsfähige Werkzeuge kaum praktikabel; Spezifikation [UML99] schlägt an vielen Stellen Fähigkeiten von Werkzeugen / Editoren vor

2 Objekte, Objekttypen und Klassen

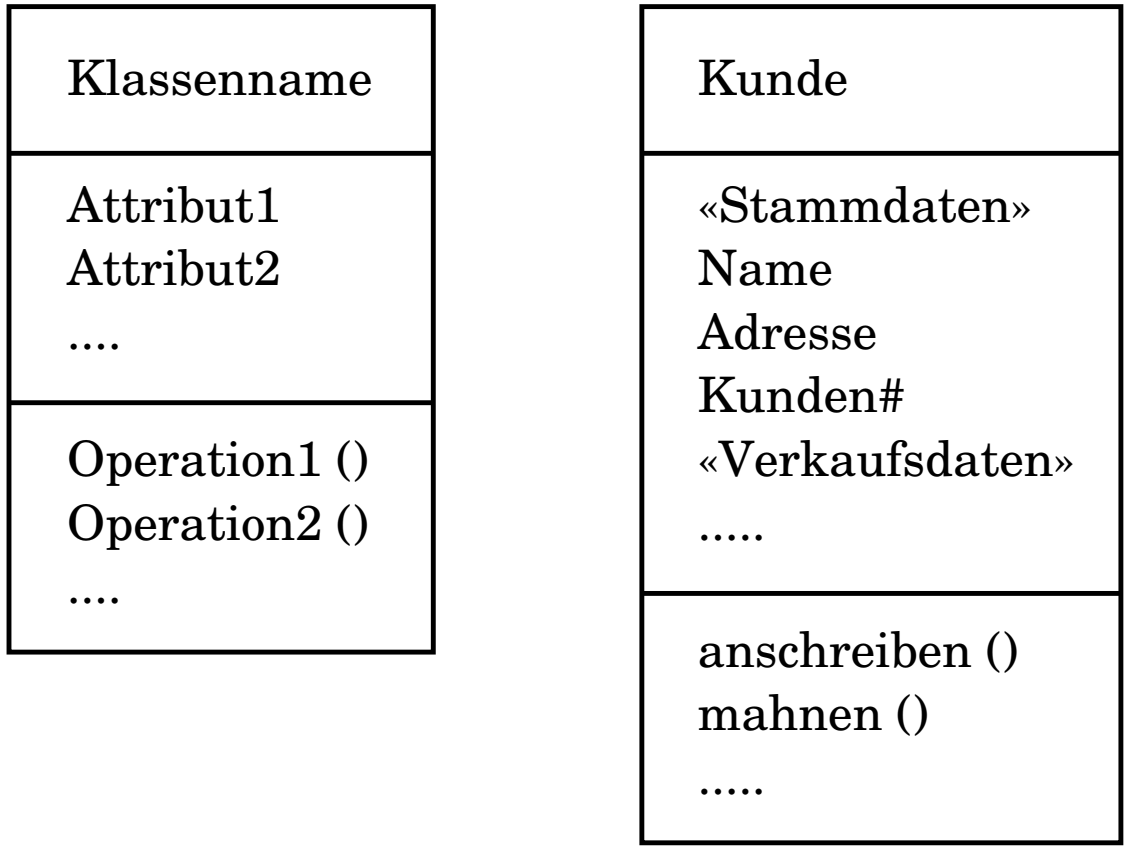
interessierende Entitäten der realen Welt - Personen, Gegenstände, Ereignisse usw. - werden durch **Objekte** modelliert

Klasse

- Objekttyp bzw. die Menge der seiner Instanzen werden als Klasse (keine Unterscheidung in der Analyse)
- Spezifiziert durch
  - ihren Namen
  - eine Liste von Attributen
  - eine Liste von Operationen

Klassenname: Substantiv im Singular, groß geschrieben, eindeutig innerhalb eines Pakets

graphische Darstellung einer Klasse:

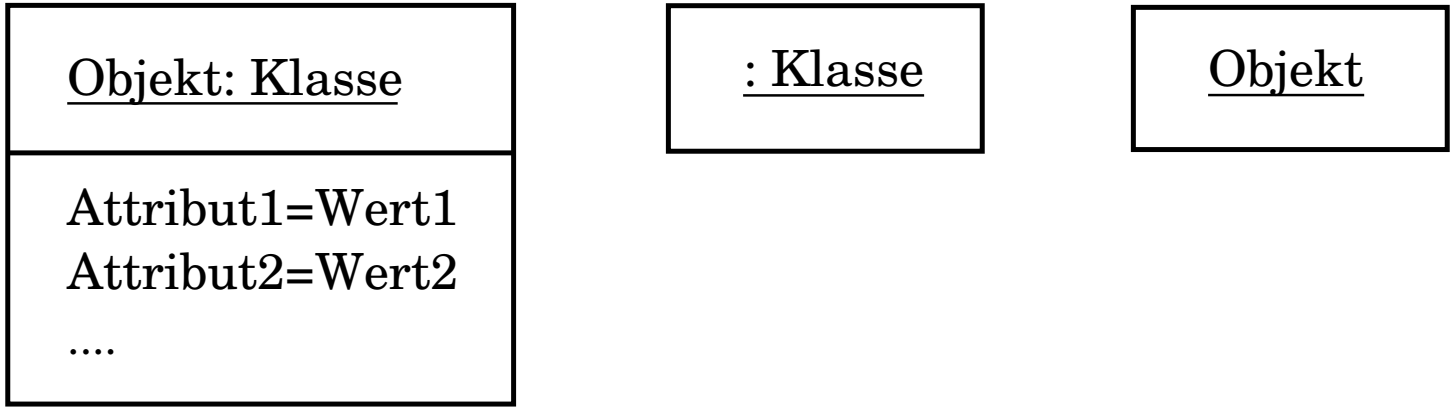


Stereotyp: zwischengeschobene Überschrift in der Liste der Attribute oder Operationen; in französische Anführungszeichen

oft nur teilweise Darstellung von Klassen

2.1 Einzelobjekte

werden in bestimmten Diagrammtypen dargestellt



- der Name des Objekts (kein Name: anonymes Objekt, als Beispiel aufzufassen), Namen müssen verschieden sein
- der Name der Klasse bzw. des Typs des Objekts
  im Kopf in der Notation `Objektname: Klassenname` , unterstrichen
- eine Liste von Attributwerten (keine Liste der Operationen)

3 Attribute

Angaben analog zu ER-Modellen:

- Name (Substantiv im Singular)
- Typ
- Initialwert.

Datenkapselung: Attribut X steht als Ersatz für (nicht in der Operationsliste aufgeführte) Operationen

- `setzeX`
- `liesX`

Attributtypen:

- komplexe Attributtypen zulässig
- elementare Klasse (*support class*)

Abgeleitete Attribute: nicht verboten

ein abgeleitetes Attribut `X` kann nicht direkt gesetzt werden (keine Operation `setzeX`)

Notation: `/` vor dem Attributnamen

Beispiel:

...

```
BLZ
/Bankname
```

...

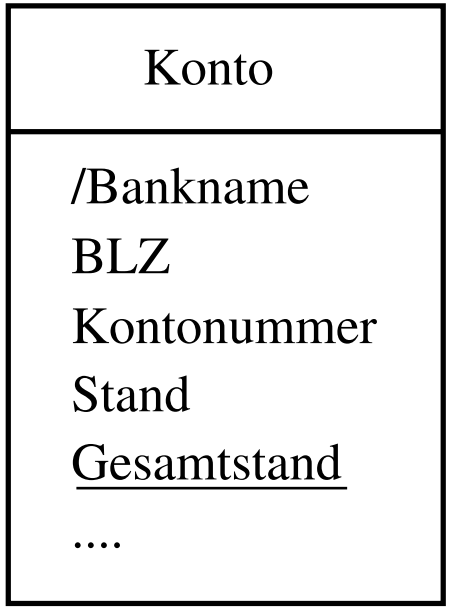
Klassenattribute:
Beispiele:

- Durchschnitt aller Kontostände
- Minimal- oder Maximalwerte usw.

Klassenattribute

- sind abgeleitet aus *allen* Objekten einer Klasse
- in der OOA der Klasse zugeordnet

Darstellung: unterstrichen



Objektidentität:

Grundannahme aus objektorientierten Sprachen:

→ jedes Objekt hat eigene Identität, unabhängig von seinen Attributwerten

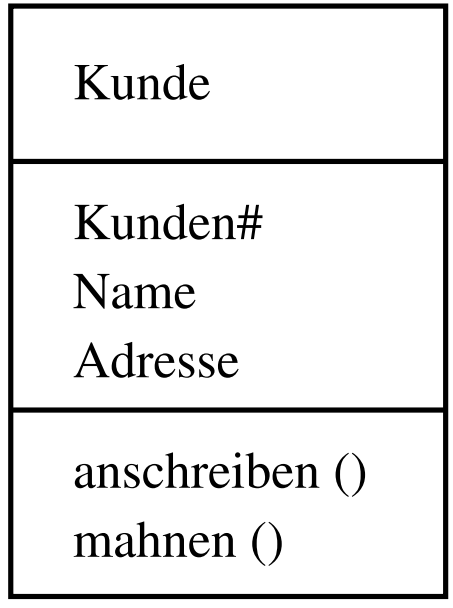
→ Attribute eines Objekttyps brauchen nicht unbedingt einen Identifikationsschlüssel zu enthalten

oo-DBMS: Objektidentität mit Hilfe von Surrogaten realisiert

[Objekttypen ohne Identifikationsschlüssel sind trotzdem i.d.R. Modellierungsfehler]

4 Operationen

Operationen modellieren “Dienste” von Objekten dieses Typs
OMG-Slang: Methode = Implementierung (also Rumpf) einer Operation (“A method is the implementation of an operation.”)



Darstellung:

- Name mit *leerer Parameterliste*
- *keine Angaben zur Sichtbarkeit*

Klassenoperationen: analog zu Klassenattributen, unterstrichen

Normale Operationen: haben implizit das Objekt als Parameter;
Klassenoperationen / Konstruktoren: haben implizit die Objektmenge als Parameter

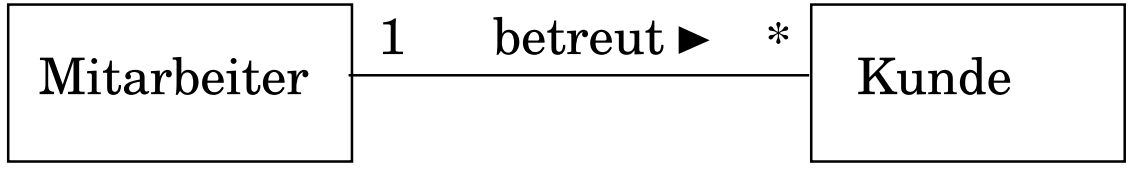
5 Beziehungen

... heißen **Assoziationen**

analog zur ER-Modellierung keine Unterscheidung zwischen Beziehungsmenge bzw. Typ von Beziehungen.

graphische Notation:

- 2-stellige Beziehungstypen: *Verbindungslinie* zwischen den involvierten Klassen
- n-stellige Beziehungstypen: *Raute* mit Verbindungslinien



Rollennamen: nur bei Bedarf angeben, am Ende der Verbindungslinie

Kardinalitäten (*multiplicity*):

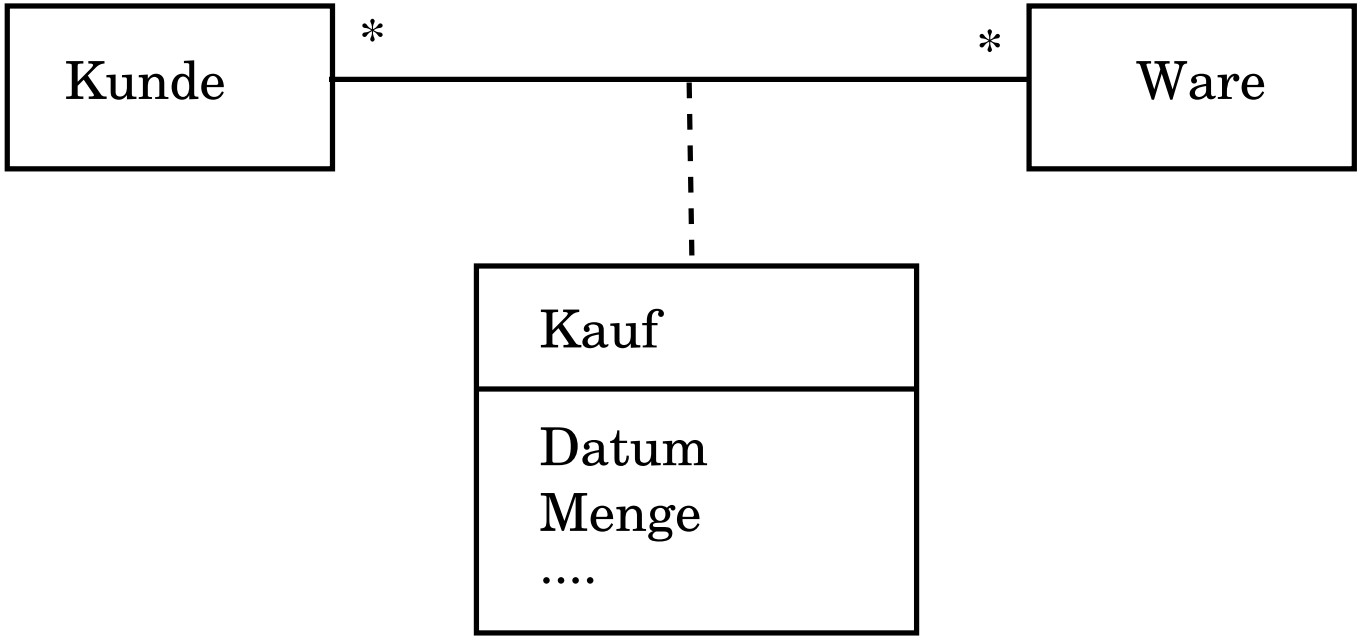
vielfältigere Ausdrucksformen als bei ER-Modellen:

1	genau 1
3	genau 3
0..1	0 oder 1
2..*	2 oder mehr
*	beliebig viele, incl. 0
.. , ..	kommagetrennte Liste mehrerer Angaben

Attributierte Beziehungen:

→ **assoziative Klasse** (analog zu schwachem Entitätstyp)

graphische Darstellung: Symbol für die assoziative Klasse wird mit einer gestrichelten Linie mit der Assoziationslinie verbunden



Aggregationen: modellieren Teil-von-Beziehungen

nur bei 2-stelligen Beziehungstypen erlaubt

graphische Darstellung: Verbindungslinie mit einer Raute auf der Seite der Klasse, die das Ganze darstellt

gemeinsame Komponente (*shared aggregation*): Teil-Objekt kann in mehreren Ganzes-Objekten enthalten sein

Beispiel: 1 Glossar in mehreren Büchern

Komposition (*composite aggregation*): Komponente kann hier nur exklusiv Teil *eines* anderen Objekts sein

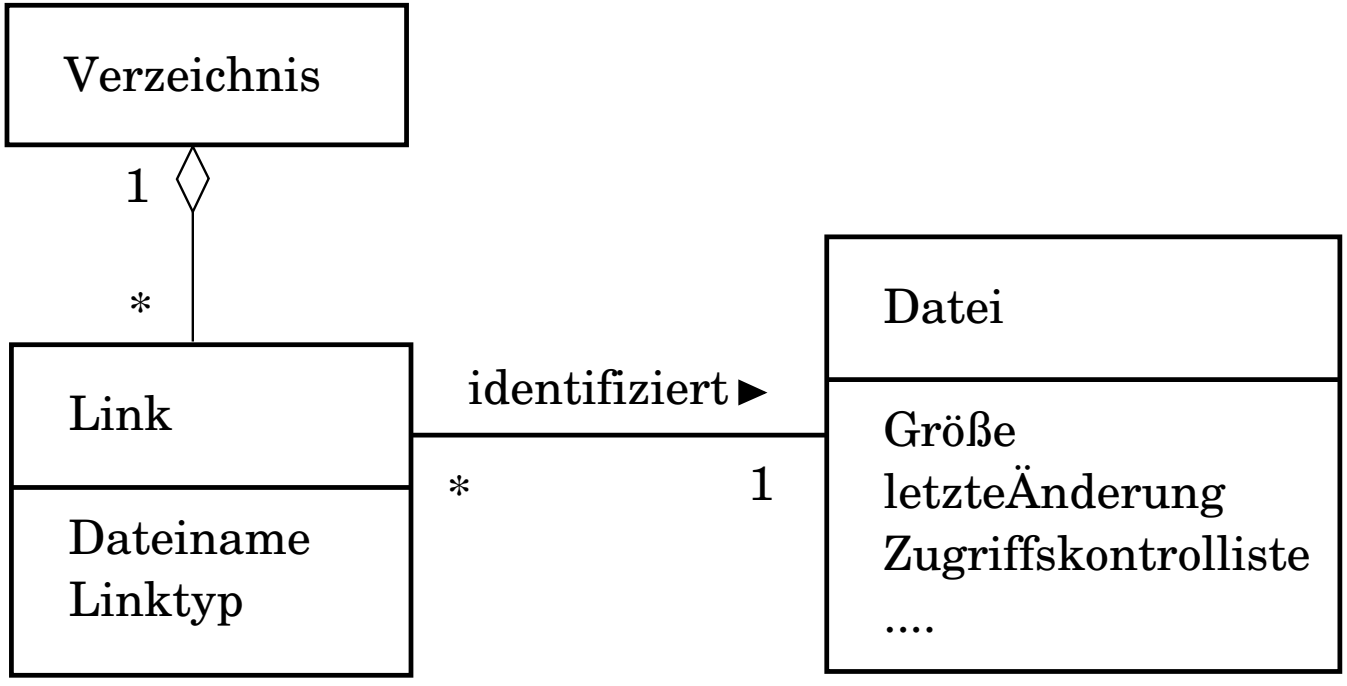
Beispiel: Wagen - Motor

Ganzes-Objekt ist für Erzeugung / Löschung seiner Komponenten verantwortlich

graphische Darstellung:

- Komposition: schwarz gefüllte Raute
- Aggregation: nichtgefüllte Raute

Beispiel: UNIX-Dateisystem



6 Typhierarchien

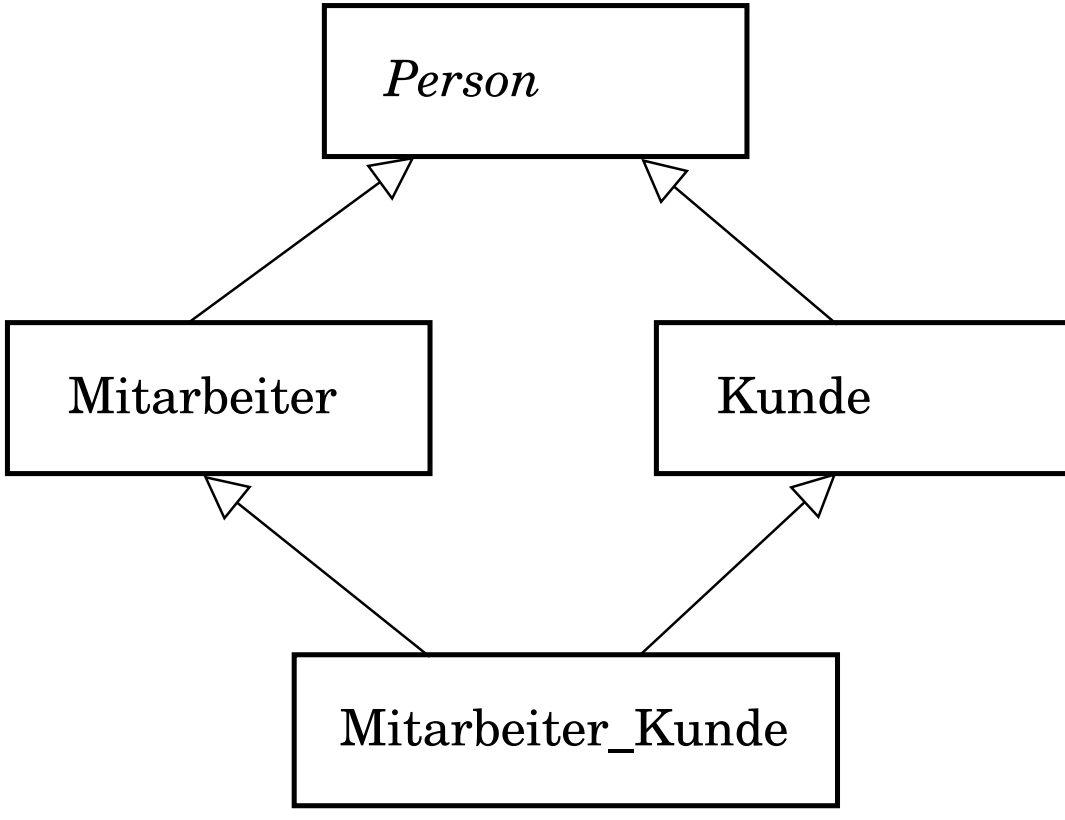
grundlegende Konzepte wie bei ER-Modellierung

graphische Darstellung: Pfeil von Subtyp zum Supertyp

Mehrfaches Erben ist zulässig

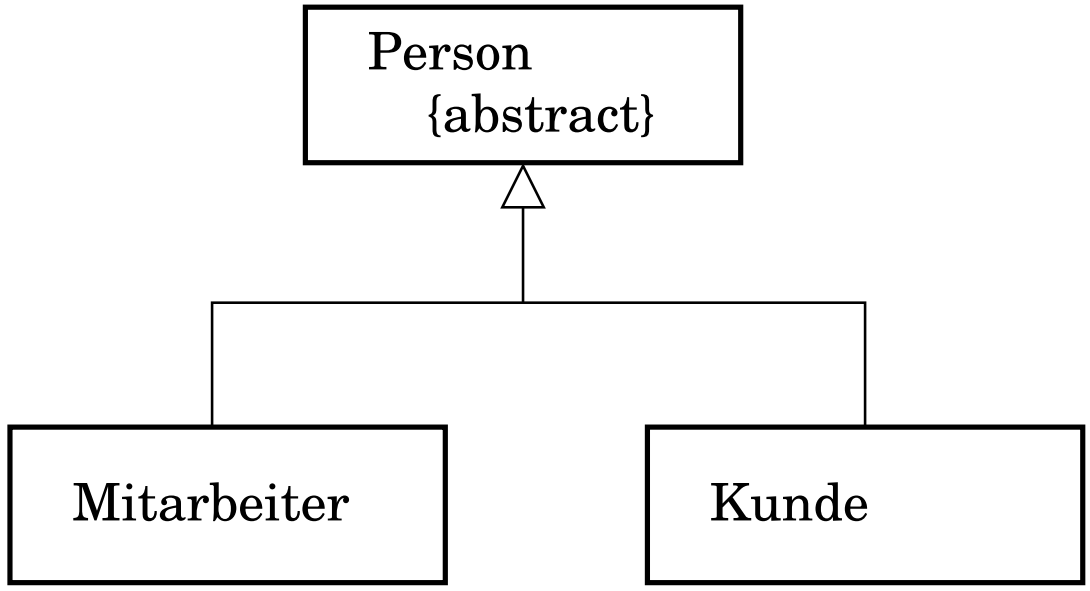
Eine Subklasse erbt von ihren Superklassen

- die Attribute
- die Operationen
- die Assoziationen (incl. Aggregationen und Kompositionen)



abstrakte Klassen: haben keine Instanzen
(entstehen bei Generalisierungen)

Darstellung:
- Name kursiv geschrieben oder
- Merkmalsangabe `abstract`



Merkmale (*properties*): bei diversen Modellelementen möglich, in geschweiften Klammern

7 Pakete

dienen zur Strukturierung größerer OOA-Modelle

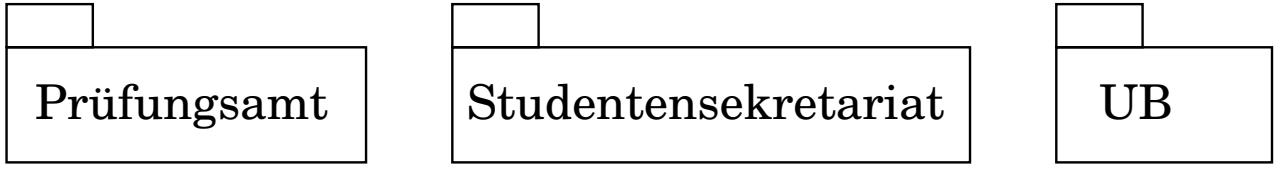
Inhalt: Klassen und/oder Pakete

jedes Paket bildet einen Namensraum für die in ihm enthaltenen Modellelemente.

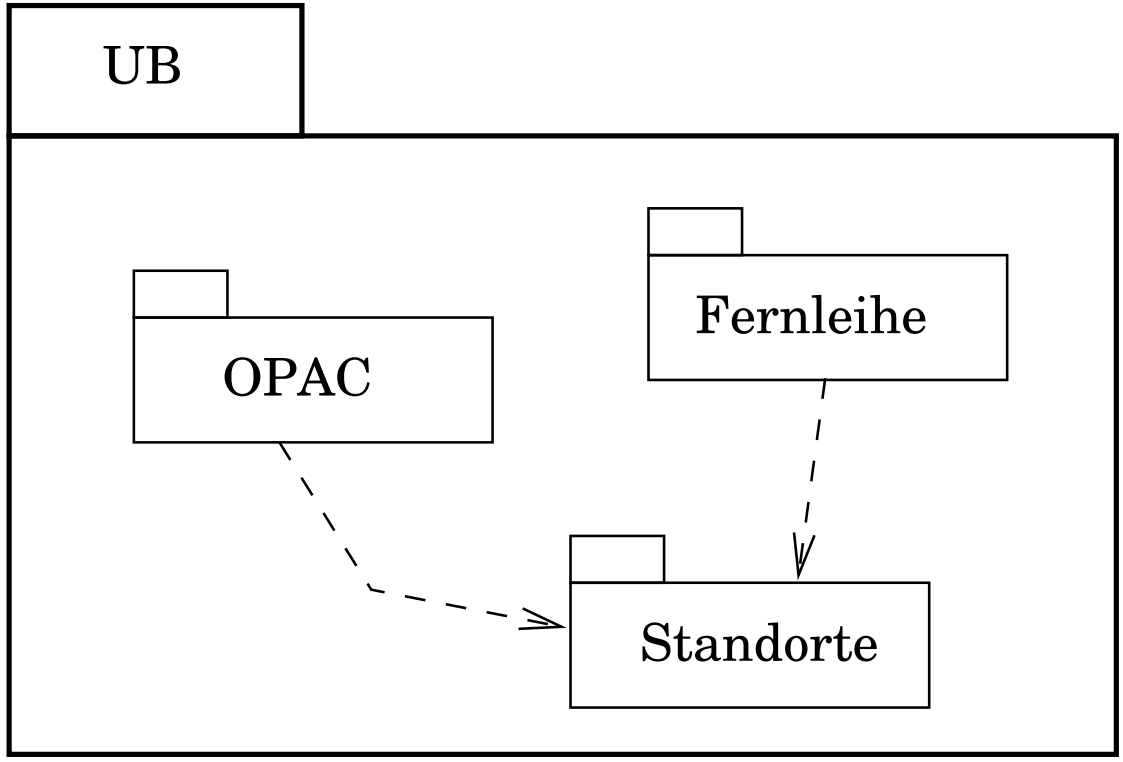
Paketdiagramm: enthält nur noch Pakete

graphische Darstellungen von Paketen:

1. kompakt: Rechteck, innen der Name des Pakets



2. detaillierter: Rechteck mit einem kleinen Reiter links oben, Name des Pakets im Reiter
   im Rechteck: Inhalt (Zeichenfläche)



“benutzt”-Beziehung (*import or access relationship*) zwischen Paketen: als gestrichelter Pfeil notiert

Referenzieren von Elementen in Paketen:

`paketname::modellelement`

z.B. `UB::Standorte::Öffnungszeiten`

Subsystem: wie Paket, aber zusätzlich:

- Schnittstelle nach außen, definiert selbständig eine Semantik
- kann instantiierbar sein

Modell: Paket, das Analysemodell oder Entwurfsmodell repräsentiert; Wurzel der Pakethierarchie

Stereotyp: «`systemModel`»