

Martin Glinz

Informatik für Ökonomen II: Modellierung von Informatiksystemen

Herbstsemester 2008

4. Modellierung von Daten



Universität Zürich
Institut für Informatik



4.1 Grundlagen und Motivation

Wie kommt Sonja Müller in den Computer



und wie kommen die Fotobücher, die sie im Web bestellt hat, zu ihr?

Welche Daten muss der Buchversand über Sonja Müller und ihre Bestellungen kennen und speichern?



Grundlagen und Motivation – 2

- Geschäfts- und Informationsobjekte müssen als Daten in Informatiksystemen repräsentiert werden, damit diese ihre Aufgaben erfüllen können
- Solche Daten sind in der Regel
 - langlebig
 - stabil
 - wertvoll

Was ist zu tun?

- **Analyse** der Geschäfts- und Informationsobjekte eines Problembereichs
 - **Welchen Ausschnitt** aus dem Problembereich muss ein geplantes System **kennen** (damit es seine Aufgaben erfüllen kann)?
 - **Wie** soll dieser Ausschnitt **auf Daten** des Systems **abgebildet** werden?
 - **Wo** liegen die **Grenzen** (welche Daten gehören zum System und welche nicht)?

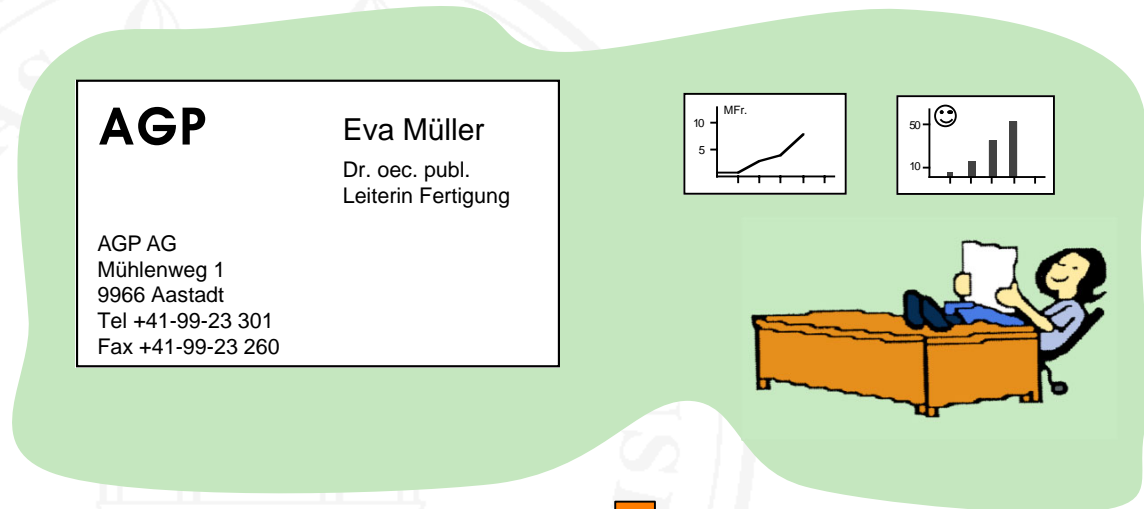
Mögliche Modellierungstechniken

- Klassische Technik für die Modellierung von Daten: **Entity-Relationship-Modell** oder Gegenstands-Beziehungs-Modell (Chen 1976)
 - Vor allem als Basis für den Entwurf von Datenbanken
 - In den Vorlesungen von Prof. Dittrich behandelt
- Direkte Modellierung von Objekten: **Klassen- und Objektmodelle**
 - Bezieht auch Operationen auf den Daten sowie Verhalten ein
 - Universeller Ansatz
 - Basis der Modellierungssprache UML

Entity-Relationship-Modelle

Das Prinzip

Die interessierenden
Elemente eines
Problembereichs ...

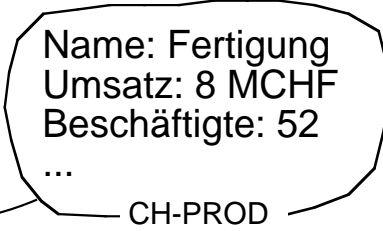
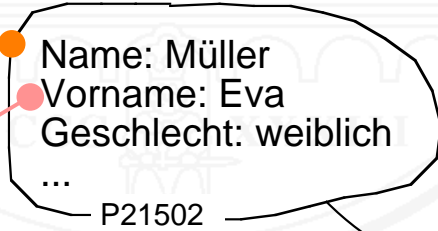


... werden abgebildet auf

○ **Gegenstände**

○ **Beziehungen**

○ **Attribute**



leitet

Die „ganze Realität“ modellieren?

- Meist interessieren nicht die konkreten Gegenstände und Beziehungen,
- ... sondern die zu Grunde liegenden **Konzepte**
- ⇒ **Klassifizieren** der Daten nach **gemeinsamen Merkmalsarten**,
- ⇒ **Abstrahieren** von den individuellen **Merkmalsausprägungen**

Eva Müller, Hans Peter Maier, Anne Weber, ...	⇒	Mitarbeiter(in) mit Attributen Name, Vorname, Geschlecht, ...
Fertigung, Einkauf, Finanzen, ...	⇒	Abteilung mit Attributen Name, Umsatz, Beschäftigte, ...
Eva Müller leitet Fertigung Hans Peter Maier leitet Finanzen Anne Weber leitet Einkauf, ...	⇒	Mitarbeiter(in) leitet Abteilung
...		

Bedeutung der Modellierungsabsicht (Pragmatik)

- Die Klassifizierung hängt vom **Umfeld** und der **Modellierungsabsicht (Pragmatik)** ab

Beispiele:

- Eva Müller wird klassifiziert als
 - **Mitarbeiterin** in einem Personalinformationssystem
 - **Patientin** in einem Spitalinformationssystem
- In einem Reisebüro werden Reisen gebucht. Die Buchung kann modelliert werden als
 - **Beziehungstyp** zwischen Reise und Kunde
 - als **Gegenstandstyp** mit Beziehungen zu Reise und Kunde

4.2 Klassen- und Objektmodelle: Grundlagen

Idee:

- Beschreibung eines Systems durch eine Menge von **Objekten**
- Jedes Objekt beschreibt einen **in sich geschlossenen Teil**
 - der **Daten**
 - der **Funktionalität**
 - des **Verhaltens**eines Systems
- Zusammenfassung **gleichartiger Objekte** zu **Klassen**
- Weiterentwicklung des Konzepts der **Entity-Relationshipmodelle**
 - Viele Ähnlichkeiten, v.a. in der Modellbildung
 - **Aber:** Entity-Relationshipmodelle erfassen **nur** den **Datenaspekt**

Beispiel: Ein Fahrausweis in einem öV-System



Operationen
Ausstellen
Validieren
Kontrollieren

Daten
Art: Einzelfahrt
Anzahl Zonen: 3
Validiert: ja
Gültigkeitsdauer: 2
Validierungsdatum: 03-05-27
Validierungszeit: 16:23
Validierungszone: 4

~~Fahrzeug Nr.
Linie
Entwerfer Nr.
Farbband wechseln~~

Verhalten
Der *ausgestellte* Fahrausweis muss vor Fahrtantritt *validiert* werden.
Die Fahrausweise können jederzeit während der Fahrt *kontrolliert* werden

Objekte und Klassen

Objekt (object) – Ein **individuell erkennbares**, von anderen Objekten **eindeutig unterscheidbares Element** der „**Realität**“, das heißt des betrachteten Problem- oder Lösungsbereichs.

Beispiel: Der **konkrete Einzelfahrschein** für drei Zonen, validiert am 27.5.03 um 16:23 Uhr,...

Klasse (class) – 1. Eine eindeutig benannte Einheit, welche eine **Menge gleichartiger Objekte** beschreibt. 2. Ein **Typ**, welcher den **Aufbau**, die Bearbeitungsmöglichkeiten und das mögliche **Verhalten** von **Objekten dieses Typs** beschreibt.

Beispiel: Die Klasse **Fahrausweis** mit den Attributen Art, Anzahl Zonen, Validierungsdatum, Validierungszeit,...

Zwei Sichten: Extension und Intension

- **Extensionale** Sicht: Eine Klasse ist eine **Menge** gleichartiger Objekte
- **Intensionale** Sicht: Eine Klasse ist ein **Typ**. Sie beschreibt, wie die Objekte der Klasse aufgebaut sind und wie diese Objekte bearbeitet werden können
- Beide Sichten werden häufig **vermischt**
- **Kompromissicht:**
Eine Klasse **repräsentiert** eine Menge von Objekten. Sie **beschreibt** den **Aufbau**, die **Bearbeitungsmöglichkeiten** und das mögliche **Verhalten** ihrer **Objekte**.

Eine Vielfalt von Modellen und Modellvarianten

- 1990 – 1998: **Vielfalt** von Modellen und Notationen, die in die Kategorie „Klassen- und Objektmodell“ fallen
- Zum Beispiel: Booch (1994), Coad und Yourdon (1991a, b), Jacobson et al. (1992), Rumbaugh et al. (1991), Wirfs-Brock et al. (1990)
- Seit ca. 1998 **dominiert** ein Industriestandard:
Die **Unified Modeling Language UML**
[Rumbaugh, Jacobson und Booch (1999), Oestereich (2004),
OMG (2007)]
- In dieser Vorlesung:
 - Allgemeingültige, grundsätzliche **Konzepte**; kompatibel mit UML
 - **Notation**
 - für Klassenmodelle: UML
 - für Klassendefinitionen: angelehnt an UML

4.3 Klassenmodelle

Klassenmodell (class model) – eine Menge zusammengehöriger Klassendefinitionen

- Eine **Klassendefinition** definiert
 - **Attribute** der Objekte der Klasse (lokale Merkmale)
 - **Beziehungen** der Objekte der Klasse zu Objekten anderer Klassen oder der Klasse zu anderen Klassen
 - **Operationen**, die auf Objekten der Klasse oder auf der Klasse selbst möglich sind
- Zusätzlich kann jeder Klasse ein **Verhaltensmodell** zugeordnet sein

Beziehungsarten

- **Assoziation:** Die Objekte einer Klasse sind Merkmale von Objekten einer anderen Klasse. Gilt in der Regel auch umgekehrt, d.h. Assoziationen sind meistens bidirektional
- **Benutzung:** Die Objekte einer Klasse benutzen Attribute oder Operationen einer anderen Klasse zur Bereitstellung ihrer eigenen Attribute und Operationen
- **Vererbung:** Eine Klasse ist Unterklasse einer anderen Klasse. Sie «erbt» alle Attribute und Operationen dieser Klasse, d.h. alle Objekte der Klasse verfügen über diese, ohne dass sie in der Klasse lokal definiert worden wären
- Zu den Assoziationen werden **Kardinalitäten** modelliert: wieviele Objekte der assoziierten Klasse müssen mindestens / dürfen höchstens mit einem Objekt der eigenen Klasse assoziiert sein.

Beispiel einer Klassendefinition

Eine Klasse aus dem Klassenmodell eines (vereinfachten) Personalinformationssystems:

KLASSE Mitarbeiter im Monatslohn
UNTERKLASSE von Mitarbeiter
ATTRIBUTE (Name, Kardinalität, Wertebereich)
Leistungslohnanteil [1,1]:CHF
Überzeitsaldo[1,1]:Stunden
Ferienguthaben [1,1]:Tage
...

BEZIEHUNGEN (Name, Kardinalität, mit Klasse)
eingestuft in [1,1]:Lohnklasse

Beispiel einer Klassendefinition – 2

OPERATIONEN

Lohn zahlen

Voraussetzung: Mitarbeiter ist aktiv

Ergebniszusicherung: Zahlungsauftrag zugunsten des Mitarbeiters ist erteilt mit Grundlohn aus Lohnklasse und Leistungslohnanteil

BENUTZT (Klasse.Operation)

Zahlungsauftrag.erteilen

Hinweis: Da Mitarbeiter im Monatslohn eine Unterklasse von Mitarbeiter ist, werden alle Attribute (zum Beispiel Name und Vorname), Beziehungen und Operationen der Klasse Mitarbeiter automatisch übernommen, ohne dass sie nochmals definiert werden müssten.

Aufgabe 4.1

In einem Produktionsplanungssystem sind Lagerartikel charakterisiert durch: die Artikelnummer, den Namen, die vorrätige Anzahl, die insgesamt reservierte Anzahl und den Preis. Ferner gibt es Funktionen, welche die aktuell verfügbare Anzahl liefern bzw. eine bestimmte Menge für einen Produktionsschritt reservieren.

Stellen Sie diese Information in Form einer Klassendefinition dar.

Klassendiagramme

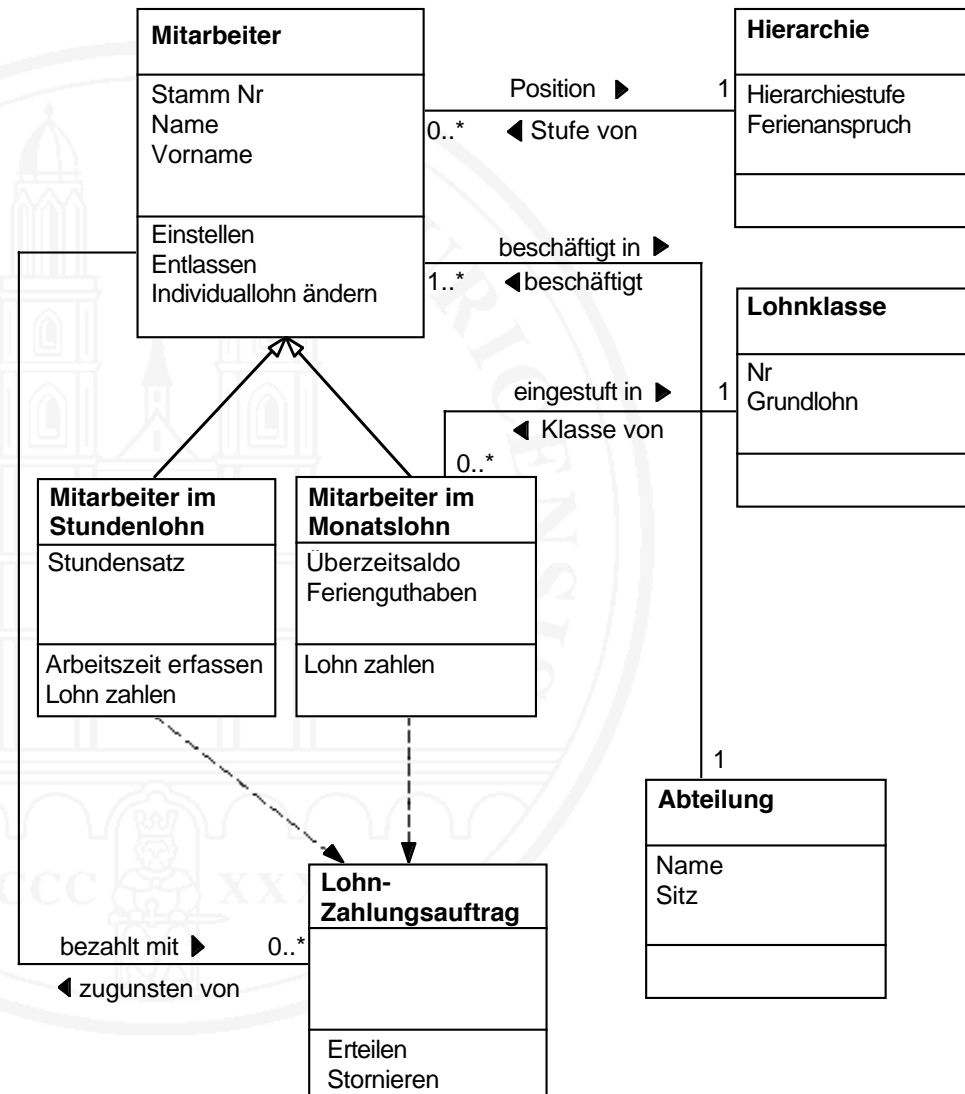
- **Klassendiagramme** beschreiben die Klassen und ihre Zusammenhänge (analog zu Entity-Relationship-Diagrammen).
- Es gibt eine Vielzahl von verschiedenen Notationen; wesentliche Gemeinsamkeit:
 - **Rechtecke** für Klassen
 - **Linien** für Assoziationen
- Heute **dominiert UML** (Unified Modeling Language) als **Notation**

Klassendiagramme in UML

- **Rechtecke** für Klassen
- Unterteilung der Klassensymbole durch zwei **Linien**: Oben der Klassenname, in der Mitte die wichtigsten Attribute, unten die wichtigsten Operationen
- **Linien** für Assoziationen
- **Gestrichelte Pfeile** für Benutzung
- **Linien mit leeren Pfeilspitzen** für Vererbung
- **Pfeilspitzen** geben die Beziehungsrichtung an
- **Kardinalitäten**: Die erste Zahl (notiert bei der Klasse, auf welche die Pfeilspitze zeigt) ist die Mindestzahl, die zweite Zahl die Höchstzahl der verknüpften Objekte. Ist nur eine Zahl angegeben, so sind Mindest- und Höchstzahl gleich.

Beispiel eines Klassendiagramms

Klassendiagramm eines Personalinformationssystems in UML-Notation (stark vereinfacht)



Aufgabe 4.2

Die folgenden Klassen aus dem Modell eines Produktionsplanungssystem seien gegeben: Lagerartikel (vgl. Aufgabe 4.1), Artikel, Sonderartikel, Lager, Lieferant. Lagerartikel und Sonderartikel sind Unterklassen von Artikel.

Zu diesen Klassen sind folgende Attribute bekannt:

- Artikel: Artikelnummer, Name
- Lager: Name, Ort
- Lieferant: Nummer, Name

Ferner seien folgende Operationen und Beziehungen bekannt:

- Lagerartikel sind in genau einem Lager eingelagert
- Zu jedem Lagerartikel gibt es mindestens einen Lieferanten
- Für Sonderartikel gibt es eine Beschaffungsoperation.

Modellieren Sie diese Informationen als Klassendiagramm in UML-Notation.

Verhaltensbeschreibung

- Verhalten kann in den heutigen Klassenmodellen **nicht integriert** beschrieben werden
 - Aktueller Forschungsgegenstand, u.a. in Zürich
- Stattdessen:
 - **Lokale** Verhaltensmodellierung durch **separates Verhaltensmodell** für jede Klasse
 - Mittel: In der Regel **Zustandsmodelle** (in dieser Vorlesung nicht behandelt)
- **Globales** Verhalten (über Klassen hinweg)
 - In vielen Modellen (einschl. UML) **nicht modelliert**
 - **Möglich** über ein **neben dem Klassenmodell** stehendes globales Verhaltensmodell

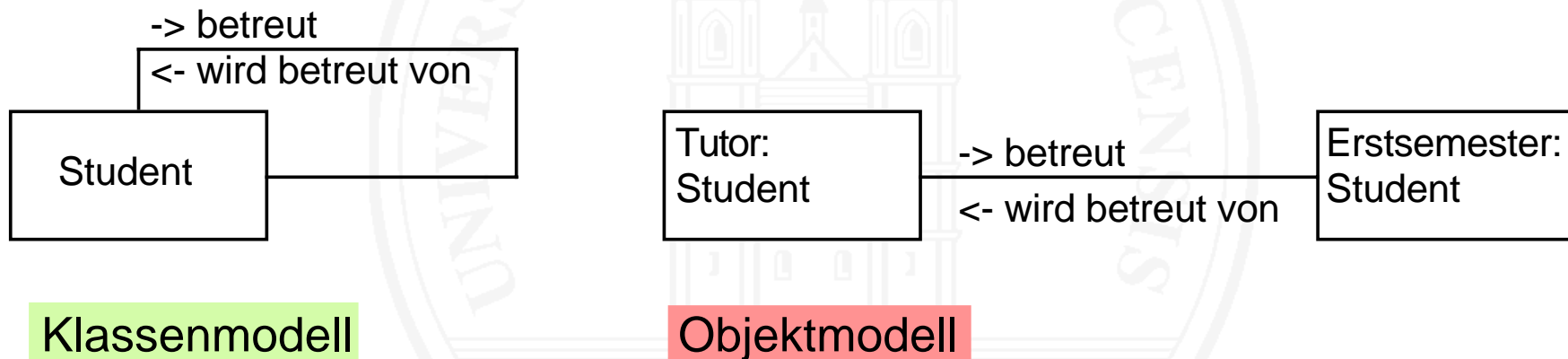
4.4 Objektmodelle

- **Objektmodelle** können als **Alternative** oder als **Ergänzung** zu Klassenmodellen verwendet werden.
- **Keine konkreten Objekte** („der Tutor Peter Meier“) sondern **abstrakte Objekte**, die als **Repräsentant** für konkrete Objekte stehen („ein Tutor“).
- Hintergrund: **Klassenmodelle versagen**, wenn
 - der **konkrete Verwendungskontext** eines Objekts modelliert werden soll
 - **verschiedene Objekte der gleichen Klasse** zu modellieren sind
 - ein **Modell hierarchisch** in Komponenten **zerlegt** werden soll

[Joos et al. (1997), Glinz, Berner, Joos (2002)]

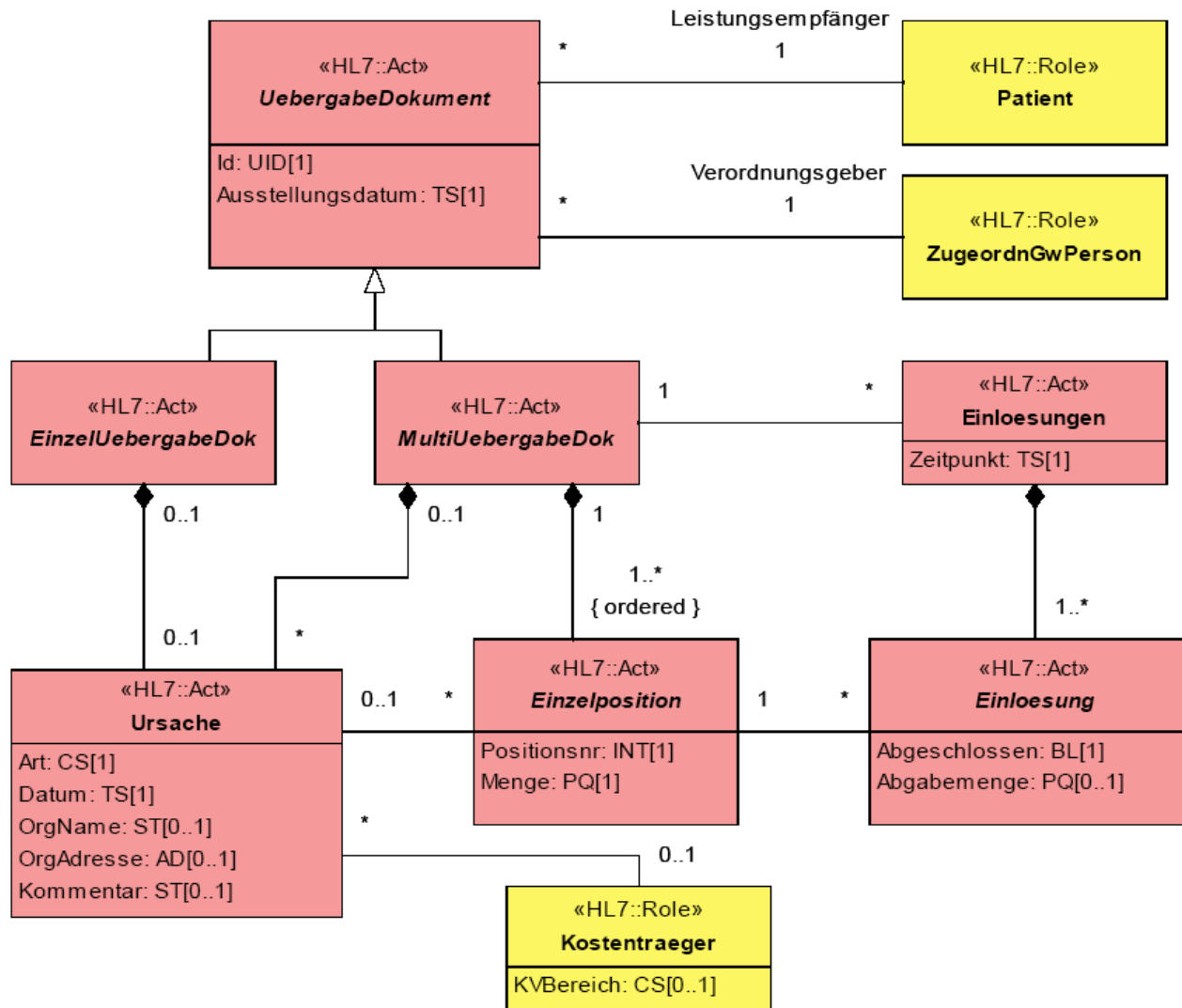
Klassenmodelle vs. Objektmodelle

- **Klassenmodelle** können **verschiedene Objekte** der **gleichen Klasse** **nicht modellieren**
- ⇒ **Objektmodelle** sind in solchen Situationen **überlegen**



- Ganz auf Klassenmodelle verzichten?
- ⇒ Geht nicht, weil Vererbungszusammenhänge auf Klassen, nicht auf Objekten definiert sind

Fallstudie Gesundheitskarte: e-Rezept



Hinweise

Rezept ist eine Unterklasse von MultiuebergabeDok

Rollen verweisen auf ein Referenzmodell im Gesundheitswesen [HL7 (2005)]

TS, CS, etc. sind spezielle Wertebereiche, zum Beispiel:

TS: Point in time

CS: Coded simple value

4.5 Methodik der Modellierung von Daten

- **Informationsquellen:** Aufgabenbeschreibungen, Gespräche oder Gesprächsnotizen, Formulare, Beschreibungen vorhandener oder verlangter Datenbestände, Glossare (Begriffslexika), Szenarien, Anwendungsfälle
- **Modellbildung (vgl. Kapitel 1)**
 - **Gewinnung:** Informationsquellen erschließen und auswerten
 - **Beschreibung:**
 - Gewonnene Information **klassifizieren**, **ordnen** und als Datenmodell-Fragmente **darstellen**
 - Fragmente abgleichen / zusammenbauen / vervollständigen
 - Rückkopplung durch ständiges **Validieren/Reflektieren** von Modellfragmenten
- **Mögliche Verfahren:** Objektanalyse, Ereignis-Reaktions-Analyse

Objektanalyse

Prinzip: **Textanalyse** des vorhandenen Materials

- Grammatisches **Subjekt**, grammatische **Objekte** → Kandidaten für Gegenstände, Gegenstandstypen, Attributwerte, Attribute oder Wertebereiche
- **Verben** beschreiben Zusammenhänge oder Handlungen:
 - **Zusammenhänge** → Beziehungen, Attribute
 - **Handlungen** → Funktionalität, Verhalten (wird in Entity-Relationship-Modellen nicht modelliert)
- **Adjektive präzisieren** Aussagen, **schränken sie ein** oder sind Kandidaten für **Attributwerte**
- Fragmente **klassifizieren, ordnen, vervollständigen**

Objektanalyse – 2

- Liefert nur bei **brauchbarem Ausgangsmaterial** brauchbare Ergebnisse
- Das Klassifizieren, Ordnen und Vervollständigen ist die **Hauptarbeit**
- Abgrenzung von Gegenständen/Gegenstandstypen gegen Attribute/Werte:
 - Gegenstände haben eine **Identität**
 - Attributwerte sind **Daten** ohne eigene Identität
 - Attribute von Attributen werden in der Regel **vermieden**: In solchen Situationen Gegenstandstypen und Beziehungstypen modellieren

Beispiel zur Objektanalyse

Zu erstellen sei ein **Informationssystem für Reisebüros**

Der Kunde verfügt über ein Glossar (Begriffslexikon) für seinen Anwendungsbereich. Hier findet sich folgende Information:

...

Buchung – Kauf eines ↑Arrangements (provisorisch oder fest) durch einen ↑Kunden. Gibt an, welches Arrangement von welchem Kunden gebucht wurde. Enthält ferner Buchungsdatum, Preis, ↑Buchungsstatus und Kurzzeichen des ↑Sachbearbeiters.

...

Vorgehen

Buchung - Kauf eines ↑Arrangements (provisorisch oder fest) durch einen ↑Kunden. Gibt an, welches Arrangement von welchem Kunden gebucht wurde. Enthält ferner Buchungsdatum, Preis, ↑Buchungsstatus und Kurzzeichen des ↑Sachbearbeiters.

Gegenstandstypen:

Buchung, ~~Kauf~~
Arrangement
Kunde

Attribute:

~~Buchungsdatum~~
Preis
~~Buchungsstatus~~
~~Kurzzeichen des~~
Sachbearbeiters
Verbindlichkeit

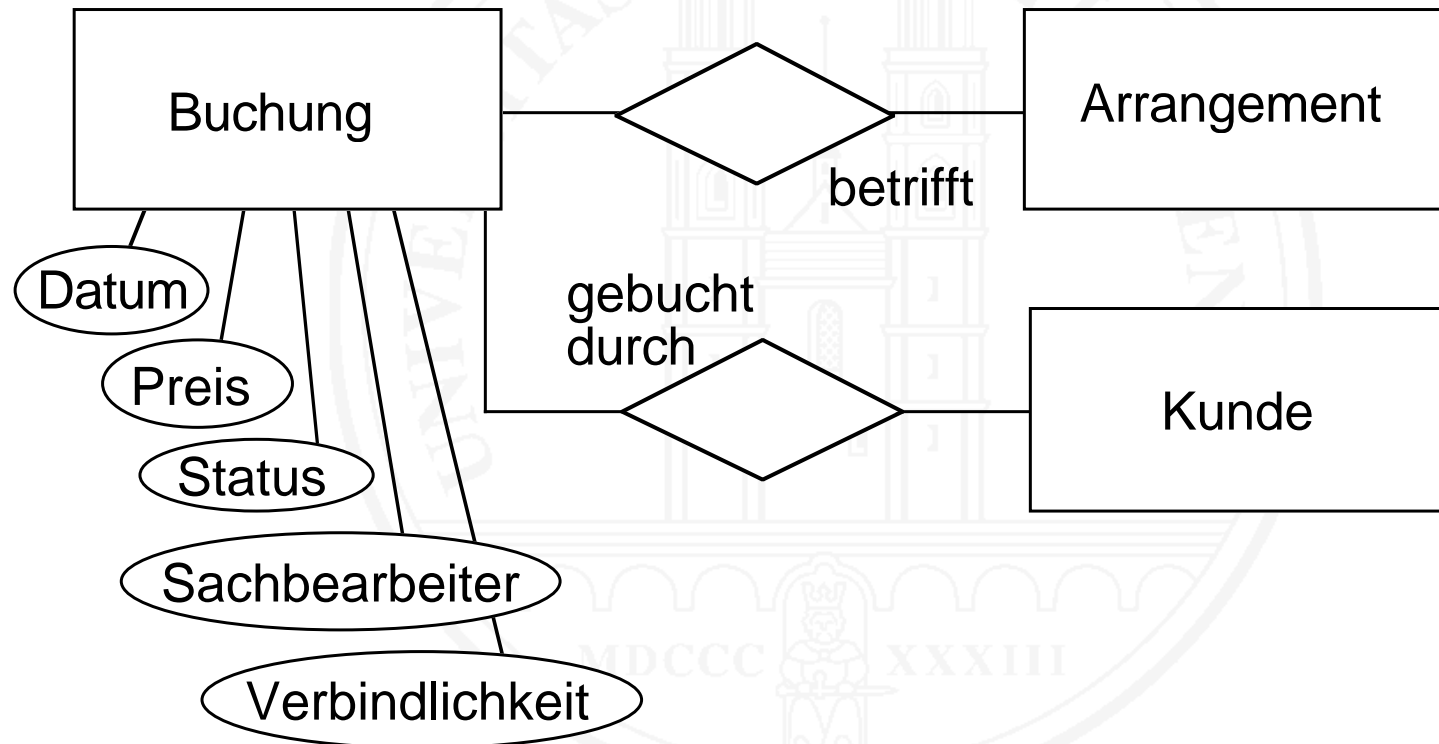
Attributwerte:

provisorisch
fest

gebucht wurde: Beziehungen Buchung – Arrangement, Buchung – Kunde
Enthält: Zuordnung von Buchungsdatum, etc. als Attribute von Buchung

Beispiel zur Objektanalyse – 2

Resultierendes Modellfragment:



Ereignis-Reaktions-Analyse

- Informatiksysteme werden durch **Ereignisse angestoßen** und müssen darauf **reagieren**
- Beispiel elektronischer Einkauf:
 - Ereignis: Benutzer will seine Bestellung bezahlen
 - Reaktion: Bezahlung abwickeln
- Prinzip:
 - Für jedes auf das Informatiksystem einwirkende **Ereignis** ...
 - ... die erwartete **Reaktion** des Informatiksystems bestimmen
- Informationsquellen:
 - **Prozessbeschreibungen**
 - **Interaktion der Benutzer mit dem System (Szenarien)**
 - **Beschreibungen von zu unterstützenden Funktionen und Abläufen**

Ereignis-Reaktions-Analyse – Analysefragen

- Auf welchen Objekten müssen welche Operationen ausgeführt werden, um die erwartete Reaktion zu erzeugen?
 - ⇒ Objekte
 - ⇒ Operationen auf Objekten
- Welche nicht mit dem Ereignis eintreffenden Daten werden für die Reaktion benötigt?
 - ⇒ Das Informatiksystem muss diese Daten kennen ⇒ Bestandteil des Entity-Relationship- / Klassen- / Objektmodells
- Welche erzeugten Ergebnisse werden nicht sofort, sondern in späteren Schritten benötigt?
 - ⇒ Das Informatiksystem muss diese Daten speichern ⇒ Bestandteil des Entity-Relationship- / Klassen- / Objektmodells

Beispiel zur Ereignis-Reaktions-Analyse

Zu erstellen sei ein **Informationssystem für Reisebüros**

In einem Szenario zu diesem System findet sich folgende Information:

Eine Person will ein Arrangement buchen. Zunächst wird geprüft, ob das gewünschte Arrangement existiert und ob es noch ausreichend freie Plätze hat. Falls ja, so wird das Arrangement gebucht. Hierzu erfasst der Berater die persönlichen Daten des Kunden (Name, Vorname, Adresse, ...). In der Buchung werden das Tagesdatum und das Zeichen des Sachbearbeiters vermerkt.

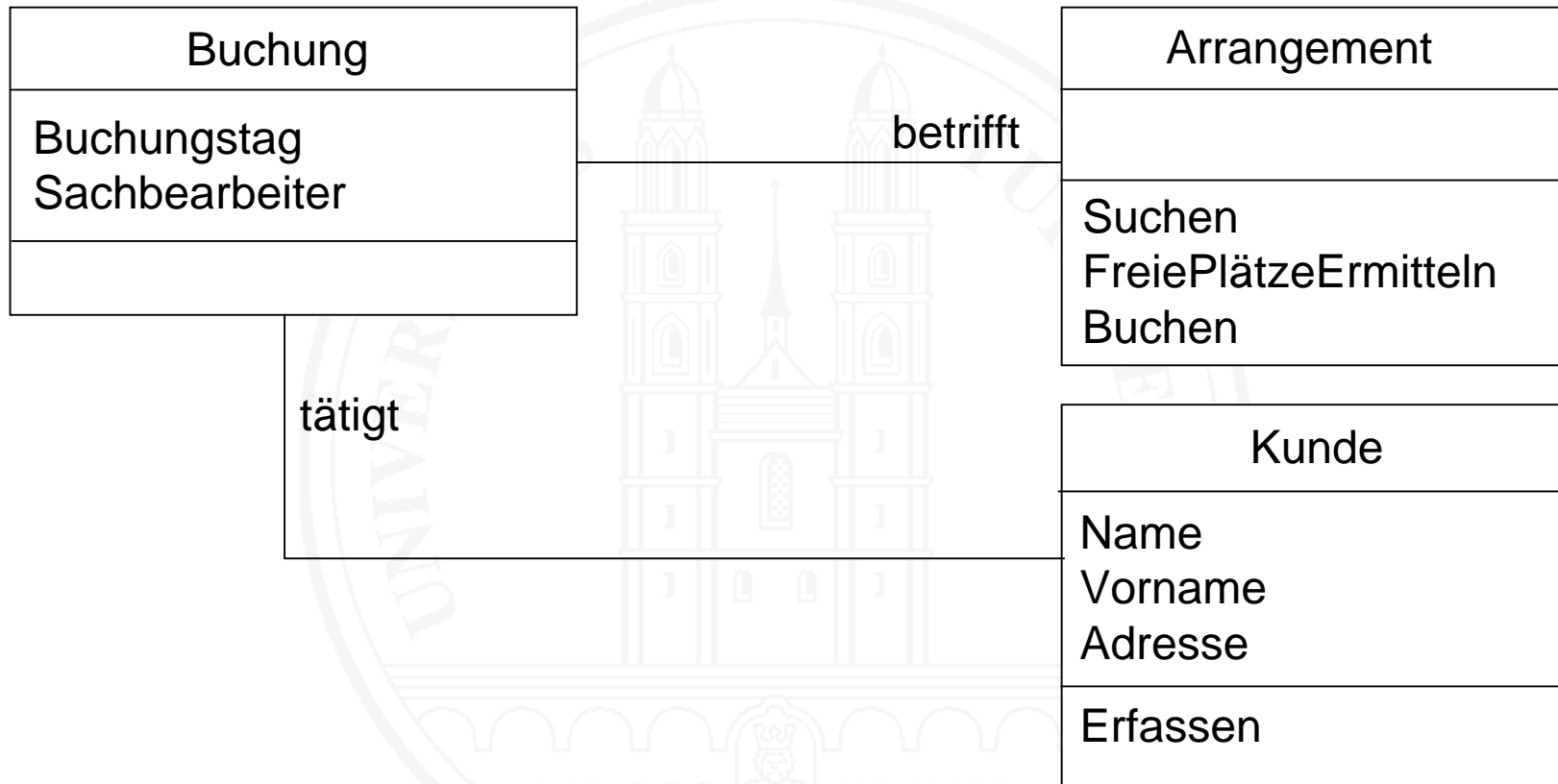
Ereignis

Objekt bzw. Objekt mit Operation

für Reaktion benötigte Daten

später benötigte Daten

Beispiel: resultierendes Modellfragment



Aufgabe 4.3: Vergleichen Sie die Modellfragmente dieses und des vorausgegangenen Beispiels. Was fällt Ihnen auf?

Aufgabe 4.4

Zu entwickeln sei ein Informationssystem für Pauschalreisen. Folgender Auszug aus der Problemstellung ist gegeben:

Das System kennt das Angebot an Pauschalreisen und die zugehörigen Anbieter. Zu jeder Reise im Angebot sind Hotelname, Kategorie, Ort, Reisedaten und Preis bekannt.

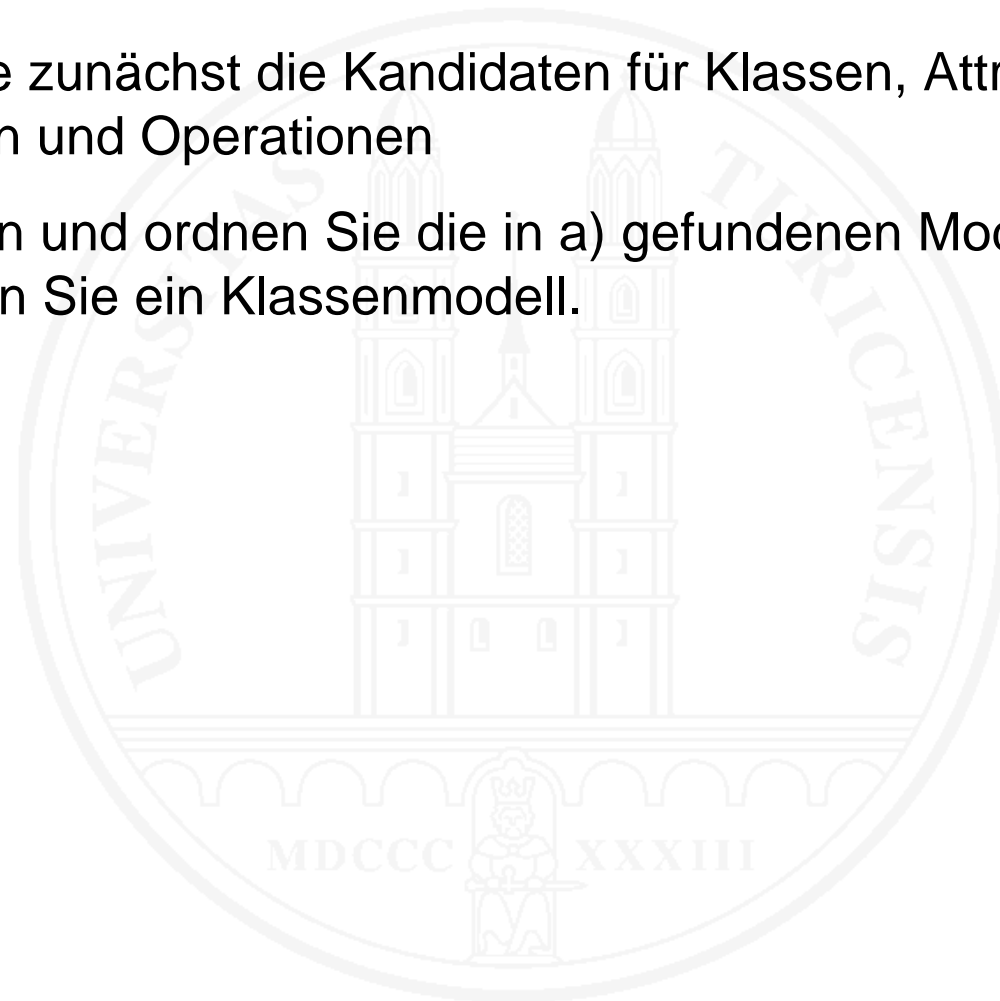
Ein Kunde kann das Angebot durchsehen und bei Interesse eine Reise provisorisch oder fest buchen. Zu jeder Buchung müssen Datum, Preis der Reise sowie Name, Vorname, Geburtsdatum und Adresse des Kunden vermerkt werden. Der Kunde kann eine Buchung stornieren.

Jede Buchung/Stornierung wird der Buchhaltung mitgeteilt, welche sich um Inkasso/Rückerstattungen kümmert.

Modellieren Sie die genannten Anforderungen in einem Klassenmodell. Lassen Sie Details der Buchhaltung beim Modellieren weg.

Aufgabe 4.4 (Fortsetzung)

- a) Ermitteln Sie zunächst die Kandidaten für Klassen, Attribute, Beziehungen und Operationen
- b) Klassifizieren und ordnen Sie die in a) gefundenen Modellelemente und zeichnen Sie ein Klassenmodell.



Literatur

bIT4health (Hrsg.) (2004b). *Erarbeitung einer Strategie zur Einführung der Gesundheitskarte: Informationsmodell, Version 1.1*

http://www.dimdi.de/dynamic/de/ehealth/karte/downloadcenter/technik/rahmenarchitektur/telematik_rahmenaktuell/b4h_informationsmodell_v1-1.pdf

Booch, G. (1994). *Object Oriented Analysis and Design with Applications*. Second Edition. Redwood City, Ca.: Benjamin/Cummings.

Chen, P.P. (1976). The Entity-Relationship Model - Toward a Unified View of Data. *ACM Transactions on Database Systems* **1**. 9-36.

Coad, P., E. Yourdon (1991a). *Object-Oriented Analysis*. 2nd edition. Englewood Cliffs, N.J.: Prentice Hall.

Coad, P., E. Yourdon (1991b). *Object-Oriented Design*. Englewood Cliffs, N.J.: Prentice Hall.

Glinz, M., S. Berner, S. Joos (2002). Object-Oriented Modeling With ADORA. *Information Systems* **27**, 6. 425-444.

HL7 (2005). *Health Level 7 Reference Information Model, Version 2.11*.

http://www.hl7.org/Library/data-model/RIM/modelpage_mem.htm

Literatur – 2

Jacobson, I., M. Christerson, P. Jonsson, G. Övergaard (1992). *Object-Oriented Software Engineering: A Use Case Driven Approach*. Amsterdam; Reading, Mass.: Addison-Wesley.

Joos, S., S. Berner, M. Arnold, M. Glinz (1997). Hierarchische Zerlegung in objektorientierten Spezifikationsmodellen. *Softwaretechnik-Trends*, **17**, 1 (Feb. 1997). 29-37.

Meyer, B. (1998). *Object Oriented Software Construction*. 2nd ed. Englewood Cliffs, N.J.: Prentice Hall.

Oestereich, B. (2004). *Objektorientierte Softwareentwicklung: Analyse und Design mit der UML 2.0*, 6. Auflage. München: Oldenbourg.

OMG (2007). *Unified Modeling Language: Superstructure, Version 2.1.1*. OMG document formal/2007-02-05. <http://www.omg.org/cgi-bin/doc?formal/07-02-05>

Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy, W. Lorensen (1991). *Object-Oriented Modeling and Design*. Englewood Cliffs, N.J.: Prentice Hall.

Rumbaugh, J., I. Jacobson, G. Booch (1999). *The Unified Modeling Language Reference Manual*. Reading, Mass.: Addison-Wesley.

Wirfs-Brock, R., B. Wilkerson, L. Wiener (1990). *Designing Object-Oriented Software*. Englewood Cliffs, N.J.: Prentice Hall.

[auf Deutsch: *Objektorientiertes Software Design*, München: Hanser, 1993]