LE 13
1

# Software-Technik

## 2 Die Definitionsphase
## Objektorientierte Analyse

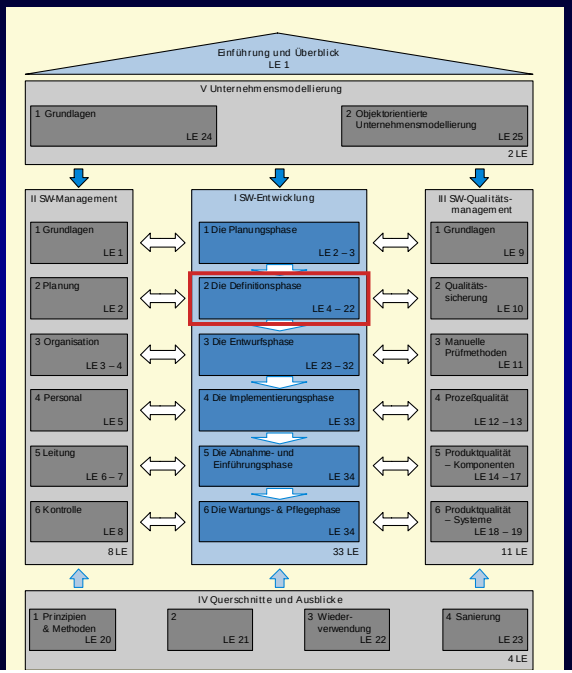
[leicht gekürzt]

Prof. Dr. Helmut Balzert
Lehrstuhl für Software-Technik
Ruhr-Universität Bochum

© Helmut Balzert 2001



LE 13
2






LE 13
3

# Inhalt

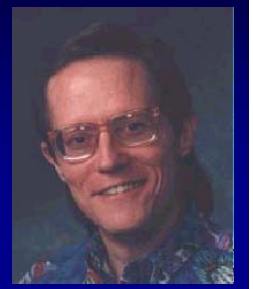
2.18 Objektorientierte Analyse
- 2.18.1 Zur Historie
- 2.18.2 OOA-Muster
- 2.18.3 Methodische Vorgehensweise mit Checklisten



LE 13
4

# 2.18 Objektorientierte Analyse

## Zur Historie



- **Peter Coad**
  *30.12.1953 in San Francisco
- Erzielte 1990 mit seinem Buch »Object-Oriented Analysis« (zusammen mit E. Yourdon) den Durchbruch in OOA (Integration des ER-Modells und des Zustandsautomaten in die Grundkonzepte der Objektorientierung)
- Wegbereiter von OOA-Mustern *(patterns)* (1995).

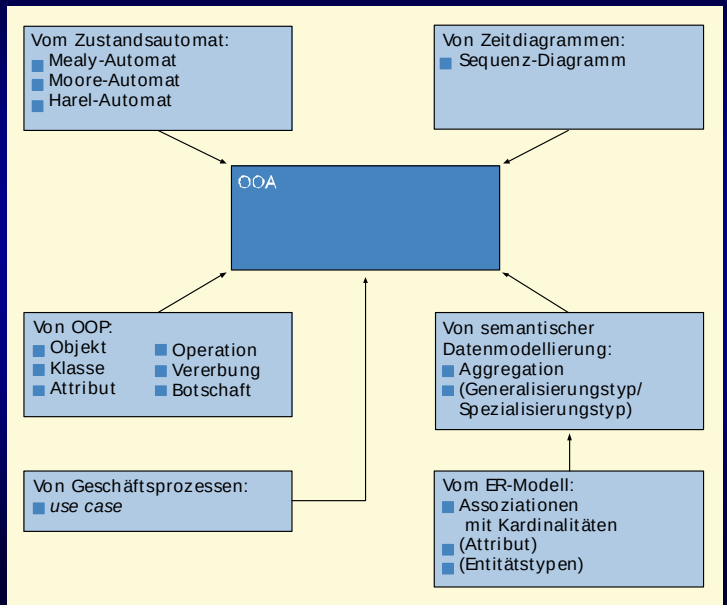
LE 13 5

# 2.18 Objektorientierte Analyse

- Zur Historie
  - **Dr. James Rumbaugh**
    ***22.8.1947 in Bethlehem, PA, USA**
    **Fellow, Rational Software Corporation**
  - **Chefentwickler der OMT-Methode *(Object Modeling Technique)*, die heute zu den führenden objektorientierten Entwicklungsmethoden zählt.**

LE 13 6

# 2.18.1 Zur Historie

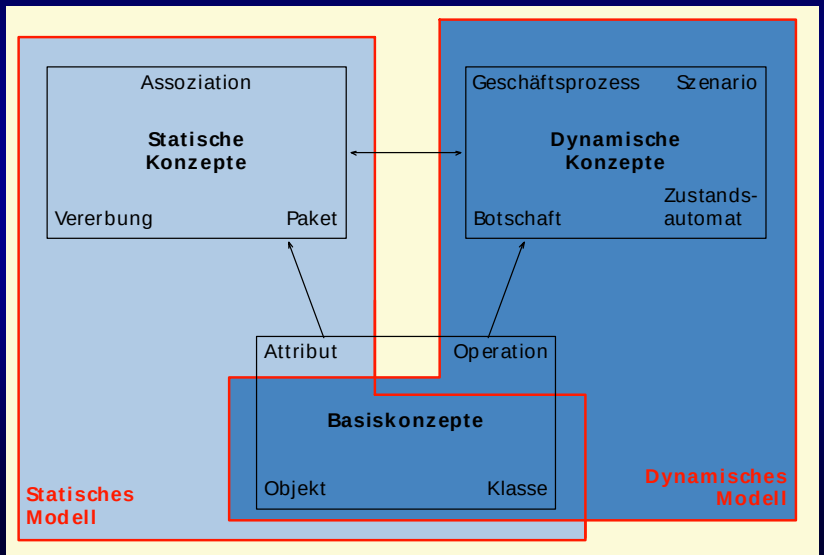





LE 13 7

# 2.18.1 Zur Historie

- Statisches und dynamisches Modell






LE 13 8

# 2.18.2 OOA-Muster

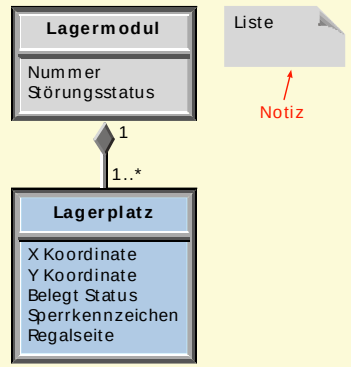
- **Muster 1: Liste**

**Lagerliste**

Nummer: 4

Störungsstatus: ok

| X Koordinate | Y Koordinate | Belegt Status | Sperrkenn-zeichen | Regal-seite |
|---|---|---|---|---|
| 50 | 100 | Frei | kein | A |
| 50 | 200 | Belegt | kein | A |
| 50 | 300 | Frei | kein | A |



## 2.18.2 OOA-Muster

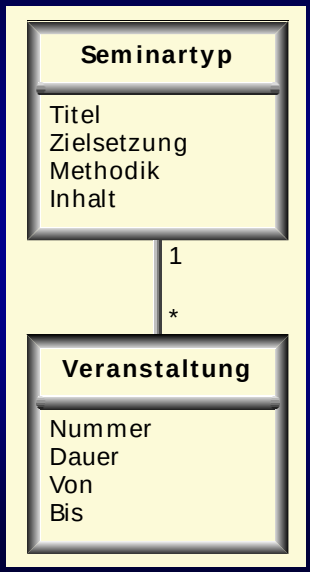
- **Muster 1: Liste**
  - Es liegt eine Komposition vor
  - Ein Ganzes besteht aus gleichartigen Teilen, d.h. es gibt nur eine Teil-Klasse
  - Teil-Objekte bleiben einem Aggregat-Objekt fest zugeordnet
    - Sie können jedoch gelöscht werden, bevor das Ganze gelöscht wird
  - Die Attributwerte des Aggregat-Objekts gelten auch für die zugehörigen Teil-Objekte
  - Das Aggregat-Objekt enthält mindestens ein Teil-Objekt, d.h. die Kardinalität ist meist 1..*.





## 2.18.2 OOA-Muster

- **Muster 2: Exemplartyp**








## 2.18.2 OOA-Muster

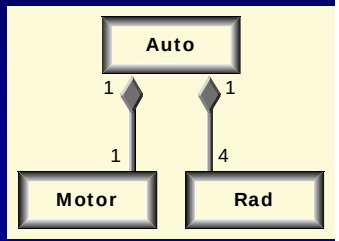
- **Muster 2: Exemplartyp**
  - Einfache Assoziation
  - Einmal erstellte Objektverbindungen werden i. Allg. nicht verändert
    - Sie werden nur gelöscht, wenn das betreffende Exemplar entfernt wird
  - Der Name der neuen Klasse enthält oft Begriffe wie Typ, Gruppe, Beschreibung, Spezifikation
  - Eine Beschreibung kann – zeitweise – unabhängig von konkreten Exemplaren existieren
    - Daher ist die Kardinalität i. Allg. *many* (0..m)
  - Würde auf die neue Klasse verzichtet, so würde als Nachteil lediglich die redundante »Speicherung« von Attributwerten auftreten.





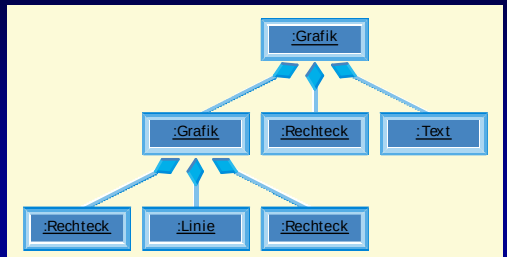
## 2.18.2 OOA-Muster

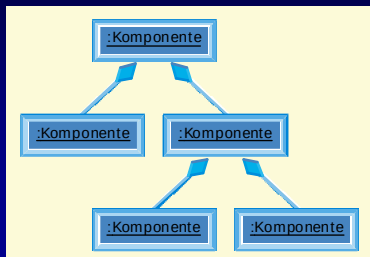
- **Muster 3: Baugruppe**
  - Es handelt sich um physische Objekte
  - Es liegt eine Komposition vor
  - Objektverbindungen bestehen über eine längere Zeit
    - Ein Teil-Objekt kann von seinem Aggregat-Objekt getrennt werden und einem anderen Ganzen zugeordnet werden
  - Ein Ganzes kann aus unterschiedlichen Teilen bestehen.

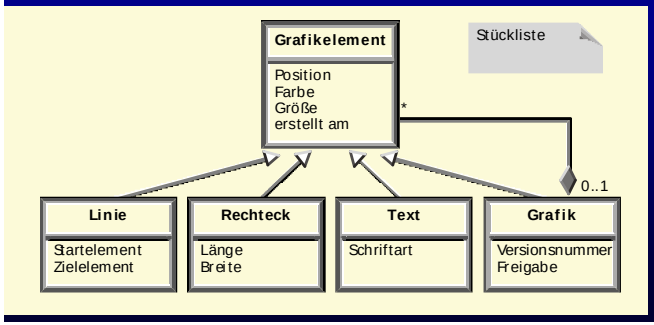


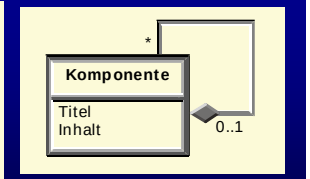
LE 13
13

## 2.18.2 OOA-Muster






LE 13
14

## 2.18.2 OOA-Muster

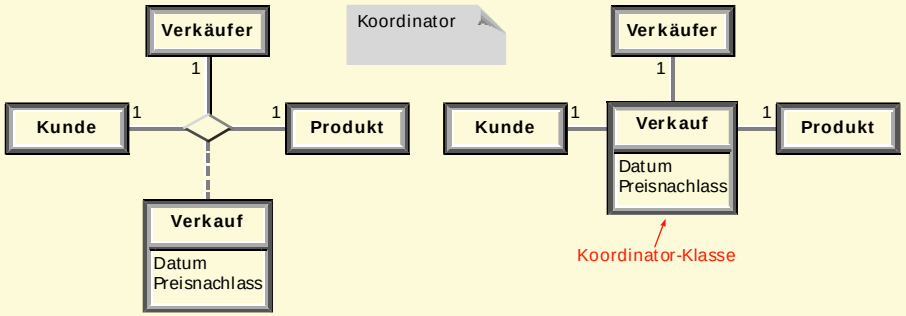
- **Muster 4: Stückliste**
  - Es liegt eine Komposition vor
  - Das Aggregat-Objekt und seine Teil-Objekte müssen sowohl als Einheit als auch einzeln behandelt werden können
  - Teil-Objekte können anderen Aggregat-Objekten zugeordnet werden
  - Die Kardinalität bei der Aggregat-Klasse ist **0..1**
  - Ein Objekt der Art A kann sich aus mehreren Objekten der Arten A, B und C zusammensetzen
  - Der Sonderfall der Stückliste ist, dass ein Stück nicht aus Objekten unterschiedlicher Art, sondern nur aus einer einzigen Art besteht.



LE 13
15

## 2.18.2 OOA-Muster

- **Muster 5: Koordinator**






LE 13
16

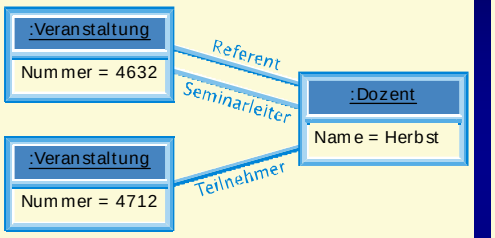
## 2.18.2 OOA-Muster

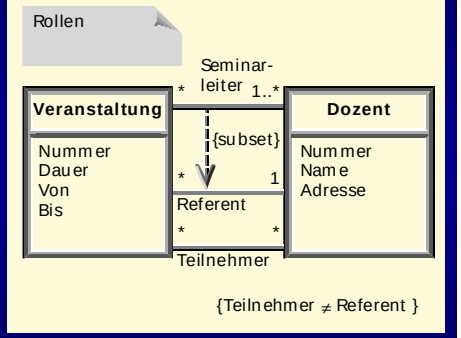
- **Muster 5: Koordinator**
  - Es liegen einfache Assoziationen vor
  - Die Koordinator-Klasse ersetzt eine n-äre (n >= 2) Assoziation durch binäre Assoziationen mit assoziativer Klasse
  - Die Koordinator-Klasse besitzt kaum Attribute/Operationen, sondern mehrere Assoziationen zu anderen Klassen, i. Allg. zu genau einem Objekt jeder Klasse.

LE 13
17

## 2.18.2 OOA-Muster

- **Muster 6: Rollen**






LE 13
18

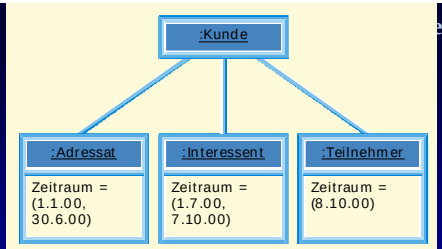
## 2.18.2 OOA-Muster

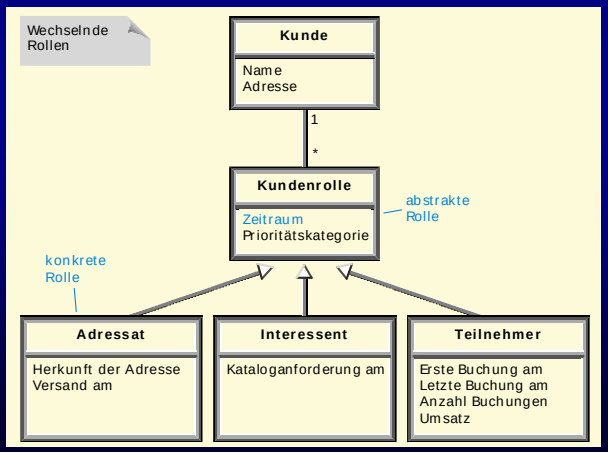
- **Muster 6: Rollen**
  - Zwischen 2 Klassen existieren 2 oder mehrere einfache Assoziationen
  - Ein Objekt kann – zu einem Zeitpunkt – in Bezug auf die Objekte der anderen Klasse verschiedene Rollen spielen
  - Objekte, die verschiedene Rollen spielen können, besitzen unabhängig von der jeweiligen Rolle die gleichen Eigenschaften und ggf. gleiche Operationen.



LE 13
19

## Rollen



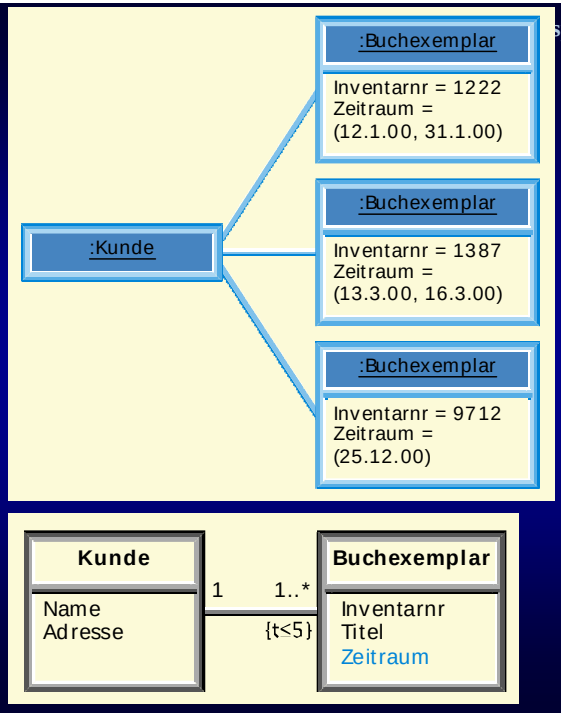






LE 13
20

## 2.18.2 OOA-Muster

- **Muster 7: Wechselnde Rollen**
  - Ein Objekt der realen Welt kann zu verschiedenen Zeiten verschiedene Rollen spielen
    - In jeder Rolle kann es unterschiedliche Eigenschaften (Attribute, Assoziationen) und Operationen besitzen
  - Die unterschiedlichen Rollen werden mittels Vererbung modelliert
  - Objektverbindungen zwischen dem Objekt und seinen Rollen werden **nur erweitert,** d.h. weder gelöscht noch zu anderen Objekten aufgebaut.








## 2.18.2 OOA-Muster

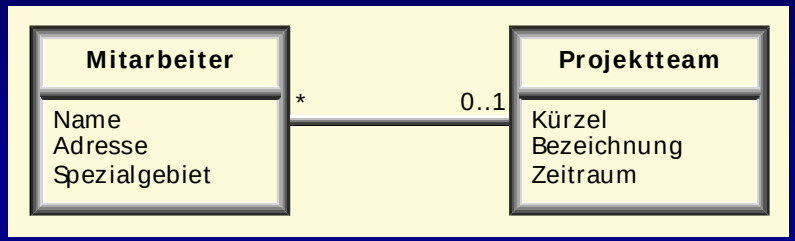
- **Muster 8: Historie**
  - Es liegt eine einfache Assoziation vor
  - Für ein Objekt sind mehrere Vorgänge bzw. Fakten zu dokumentieren
  - Für jeden Vorgang bzw. jedes Faktum ist der Zeitraum festzuhalten
  - Aufgebaute Objektverbindungen zu den Vorgängen bzw. Fakten werden nur erweitert
  - Die zeitliche Restriktion `{t#k}` (k = gültige Kardinalität) sagt aus, was zu einem Zeitpunkt gelten muss, wobei # für die Vergleichsoperationen `=, <, >, <=, >=` und ≠ steht.





## 2.18.2 OOA-Muster

- **Muster 9: Gruppe**





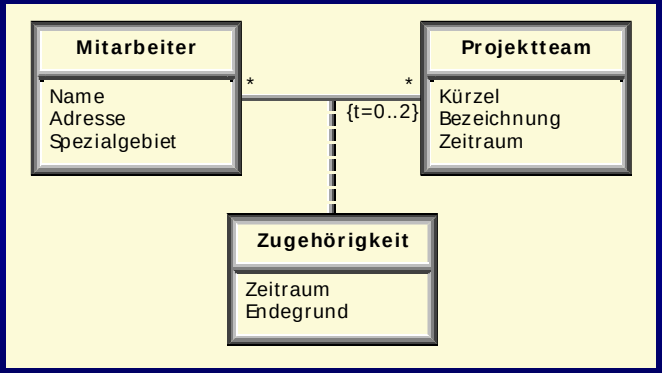



## 2.18.2 OOA-Muster

- **Muster 9: Gruppe**
  - Es liegt eine einfache Assoziation vor
  - Mehrere Einzel-Objekte gehören – zu einem Zeitpunkt – zum selben Gruppen-Objekt
  - Es ist jeweils zu prüfen, ob die Gruppe – zeitweise – ohne Einzel-Objekte existieren kann oder ob sie immer eine Mindestanzahl von Einzel-Objekten enthalten muss
  - Objektverbindungen können auf- und abgebaut werden.

## 2.18.2 OOA-Muster

- **Muster 10: Gruppenhistorie**








## 2.18.2 OOA-Muster

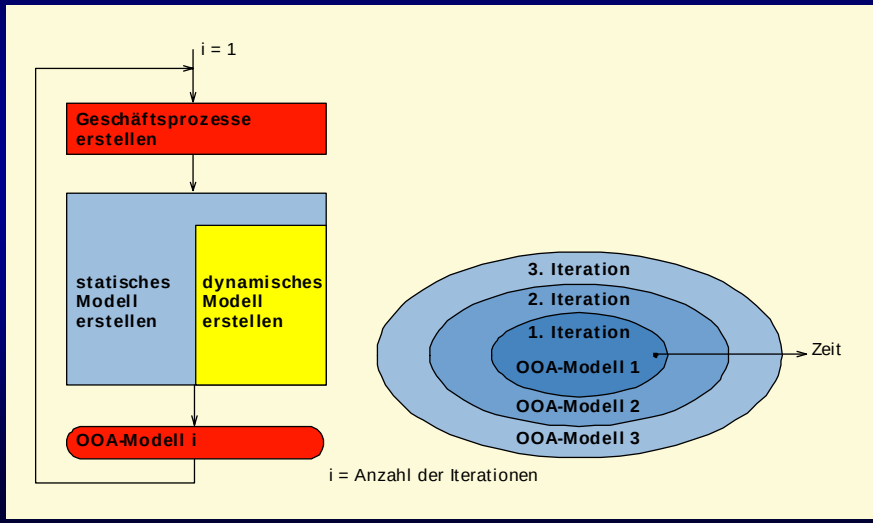
- **Muster 10: Gruppenhistorie**
  - Ein Einzel-Objekt gehört – im Laufe der Zeit – zu unterschiedlichen Gruppen-Objekten
  - Die Historie wird mittels einer assoziativen Klasse modelliert
    - Dadurch ist die Zuordnung zwischen Einzel-Objekten und Gruppen deutlich sichtbar
  - Die zeitliche Restriktion `{t=k}` (k = gültige Kardinalität) sagt aus, was zu einem Zeitpunkt gelten muss
  - Da Informationen über einen Zeitraum festzuhalten sind, bleiben erstellte Objektverbindungen bestehen und es werden nur Verbindungen hinzugefügt.





## 2.18.3 Methodische Vorgehensweise

- **Zur Modellbildung**








## 7 Schritte zum statischen Modell

### 1 Klassen identifizieren

- Für jede Klasse nur so viele Attribute und Operationen identifizieren, wie für das Problemverständnis und das einwandfreie Erkennen der Klasse notwendig sind
  - →Klassendiagramm
  - →Kurzbeschreibung Klassen

### 2 Assoziationen identifizieren

- Zunächst nur die reinen Verbindungen eintragen, d.h. noch *keine* genaueren Angaben, z.B. zur Kardinalität oder zur Art der Assoziation, machen
  - →Klassendiagramm.

## 7 Schritte zum statischen Modell

### 3 Attribute identifizieren

- Identifizieren aller Attribute des Fachkonzepts
  - →Klassendiagramm

### 4 Vererbungsstrukturen identifizieren

- ... aufgrund der identifizierten Attribute →Klassendiagramm

### 5 Assoziationen vervollständigen

- Endgültig festlegen, ob eine »normale« Assoziation, Aggregation oder Komposition vorliegt sowie Festlegung der Kardinalitäten, Rollen, Namen und Restriktionen
  - →Klassendiagramm
  - →Objektdiagramm.





## 7 Schritte zum statischen Modell

### 6 Attribute spezifizieren

- Für alle identifizierten Attribute eine vollständige Spezifikation erstellen
  - → Attributspezifikation

### 7 Muster identifizieren

- Das Klassendiagramm daraufhin überprüfen, ob Muster enthalten sind und diese richtig modelliert wurden.





## 3 Schritte zum dynamischen Modell

### 1 Szenarios erstellen

- Jeden Geschäftsprozess durch eine Menge von Szenarios präzisieren
  - →Sequenzdiagramm, Kollaborationsdiagramm
  - Alternativ oder ergänzend können auch Aktivitätsdiagramme verwendet werden

### 2 Zustandsautomat erstellen

- Für jede Klasse prüfen, ob ein nicht-trivialer Lebenszyklus erstellt werden kann und muss
  - →Zustandsdiagramm.





## 3 Schritte zum dynamischen Modell

### 3 Operationen beschreiben

- Überlegen, ob eine Beschreibung notwendig ist
  - Wenn ja, dann ist je nach Komplexitätsgrad die entsprechende Form zu wählen
    - →Klassendiagramm
    - →fachliche Beschreibung der Operationen
    - → Zustandsautomaten
    - → Aktivitätsdiagramme

- →SemOrg.

## Checkliste Klassen

### Ergebnisse

- Klassendiagramm
  - Jede Klasse – entweder nur mit Namen oder mit wenigen wichtigen Attributen/Operationen – in das Klassendiagramm eintragen
  - Kurzbeschreibung der Klassen

### Konstruktive Schritte

1 Welche Klassen lassen sich mittels Dokumentanalyse identifizieren *(bottom-up)*?

- Aus Formularen und Listen Attribute entnehmen und zu Klassen zusammenfassen.





## Checkliste Klassen

- *Reengineering* von Software-Altsystemen:
  - Arbeitsabläufe anhand von Benutzerhandbüchern, Bildschirmmasken, Dateibeschreibungen ermitteln
  - Anhand des laufenden Systems die Funktionen der Geschäftsprozesse ausführen
  - Klassen mithilfe der Bildschirmmasken ermitteln
- Bei technischen Systemen bieten sich die realen Objekte als Ausgangsbasis an, z.B. Lagermodul

2 Welche Klassen lassen sich aus der Beschreibung der Geschäftsprozesse identifizieren *(top-down)*?

- Beschreibung nach Klassen durchsuchen
  - Oft sind Substantive potenzielle Klassen
- Potenzielle Klassen auf Attribute überprüfen
- Akteure, über die man sich etwas »merken« muss, sind potenzielle Klassen.





## Checkliste Klassen

3 Zu welchen Kategorien gehören die Klassen?

- Konkrete Objekte bzw. Dinge, z.B. PKW
- Personen und deren Rollen, z.B. Kunde, Dozent
- Informationen über Aktionen, z.B. Banküberweisung durchführen
- Orte, z.B. Wartezimmer
- Organisationen, z.B. Filiale
- Behälter, z.B. Lagerplatz
- Dinge in einem Behälter, z.B. Reifen in einem Lagerplatz
- Ereignisse, z.B. Eheschließung
- Kataloge, z.B. Produktkatalog
- Verträge, z.B. Autokaufvertrag.





## Checkliste Klassen

### Analytische Schritte

4 Liegt ein aussagefähiger Klassenname vor?

- Der Klassenname soll
  - der Fachterminologie entsprechen
  - ein Substantiv im Singular sein
  - dasselbe ausdrücken wie die Gesamtheit der Attribute
  - nicht die Rolle dieser Klasse in einer Beziehung zu einer anderen Klasse beschreiben
  - eindeutig im Paket bzw. im System sein
  - nicht dasselbe ausdrücken wie der Name einer anderen Klasse

5 Ist das gewählte Abstraktionsniveau richtig?

- Die Ziele sind nicht
  - möglichst viele Klassen oder
  - Klassen geringer Komplexität zu identifizieren.

## Checkliste Klassen

6 Wann liegt keine Klasse vor?

- Keine Klassen bilden, um Mengen von Objekten zu verwalten

7 Fehlerquellen

- Zu kleine Klassen
- Aus jedem Report eine Klasse modellieren
- Klasse modelliert Benutzungsoberfläche
- Klasse modelliert Entwurfs- oder Implementierungsdetails

## Für klassische Entwickler

8 Identifizieren von Klassen für Datenmodellierer

- Eine potenzielle Klasse ist jeder Entitätstyp und jeder Beziehungstyp mit Attributen.





## 2.18.3.3 Checkliste Klasse

- 1 Dokumentanalyse
  - Fallstudie »Seminarorganisation«

Anmeldung zu TEACHWARE-Seminaren

Als Teilnehmer zu nachfolgenden TEACHWARE-Seminaren wird angemeldet:

Titel | Vorname | Name

Veranstaltungs-Nr. | Seminarbezeichnung | vom | bis

Anmeldebestätigung und Rechnung erbeten an:

Titel | Vorname | Name

Firma | Straße/Postfach | Land

PLZ | Ort | Telefon





## 2.18.3.3 Checkliste Klasse

- Klassen des Anmeldeformulars

| Teilnehmer |
|---|
| Titel<br>Vorname<br>Name |

| Seminarveranstaltung |
|---|
| Veranstaltungs-Nr<br>Seminarbezeichnung<br>Seminarbeginn<br>Seminarende |

| Rechnungsempfänger |
|---|
| Titel<br>Vorname<br>Name<br>Firma<br>Straße/Postfach<br>Land<br>PLZ<br>Ort<br>Telefon |



| Kunde |
|---|
| Name<br>Adresse<br>Kontakt<br>Geburtsdatum<br>Funktion<br>Umsatz<br>Kunde seit |
| erstelle Adressaufkleber()<br>erstelle Mitteilung() |

| Dozent |
|---|
| Nummer<br>Name<br>Adresse<br>Kontakt<br>Geburtsdatum<br>Biografie<br>Honorar pro Tag<br>Kurzmitteilung<br>Notizen<br>Dozent seit |

| Buchung |
|---|
| Angemeldet am<br>Bestätigung am<br>Rechnung am<br>Abgemeldet am<br>Mitteilung am |
| erstelle Mitteilung()<br>anmelden()<br>abmelden() |

| Firma |
|---|
| Kurzname<br>Name<br>Adresse<br>Kontakt<br>Ansprechpartner<br>Abteilung<br>Geburtsdatum<br>Funktion<br>Kurzmitteilung<br>Notizen<br>Umsatz<br>Kunde seit |

| Veranstaltung |
|---|
| Nummer<br>Dauer<br>Vom<br>Bis<br>TagesrasterAnfang<br>TagesrasterEnde<br>AnfangErsterTag<br>EndeLetzterTag<br>Ort<br>Adresse<br>TeilnehmerMax<br>Storniert |
| erstelle Teilnehmerliste()<br>erstelle Teilnehmerurkunde()<br>stornieren()<br>erstelle Honorarmitteilung() |

| Seminartyp |
|---|
| Kurztitel<br>Titel<br>Zielsetzung<br>Methodik<br>Inhaltsübersicht<br>Tagesablauf<br>Dauer<br>Unterlagen<br>Zielgruppe<br>Voraussetzungen<br>Gebühr ohne MWST<br>Teilnehmerzahl max<br>Teilnehmerzahl min |

| Firmenbuchung |
|---|
| erstelle Mitteilung()<br>anmelden()<br>abmelden() |

| Zahlungsverzug |
|---|
| Rechnungsdatum<br>Rechnungsbetrag |

| Mitarbeiter |
|---|
| Name<br>Berechtigung<br>Passwort<br>Tätigkeit |

## 2.18.3.3 Checkliste Klasse

- 3 Kategorien
  - Personen und deren Rollen
    - Kunde, Dozent, Mitarbeiter
  - Informationen über Aktionen
    - Buchung, Firmenbuchung, Zahlungsverzug
  - Organisation
    - Firma
  - Kataloge
    - Seminar, Veranstaltung
- →SemOrg.





## Checkliste Assoziationen

### Ergebnis

- Klassendiagramm
  - Assoziationen im ersten Schritt nur als Linie eintragen
  - Noch keine Kardinalitäten, Aggregationen, Kompositionen, Rollen, Namen, Restriktionen

### Konstruktive Schritte

1 Welche Assoziationen lassen sich mittels Dokumentanalyse ableiten?

- Aus Primär- und Fremdschlüsseln ermitteln.





## Checkliste Assoziationen

2 Welche Assoziationen lassen sich aus Beschreibungen der Geschäftsprozesse ermitteln?

- Nach Verben suchen, insbesondere:
  - räumliche Nähe (in der Nähe von)
  - Besitz (hat)
  - Aktionen (fährt)
  - Kommunikation (redet mit), (verheiratet mit)
  - allgemeine Beziehungen

3 Liegt eine Assoziation folgender Kategorien vor?

- A ist physische Komponente von B
- A ist logische Komponente von B
- A ist eine Beschreibung für B
- A ist eine Zeile einer Liste B
- A ist ein Mitglied von B.





## Checkliste Assoziationen

- A ist eine organisatorische Einheit von B
- A benutzt B
- A kommuniziert mit B
- A besitzt B

4 Welche Restriktionen muss die Assoziation erfüllen?

- Eine Assoziation: `{ordered}`
- Mehrere Assoziationen: `{or}, {subset}`

5 Welche Rollen spielen die beteiligten Klassen?

- Rollennamen angeben, wenn
  - Assoziation zwischen derselben Klasse existiert
  - eine Klasse in verschiedenen Assoziationen auch verschiedene Rollen spielt
  - durch den Rollennamen die Bedeutung der Klasse in der Assoziation genauer spezifiziert werden kann.

## Checkliste Assoziationen

### Analytische Schritte

**6 Ist eine Benennung notwendig oder sinnvoll?**

- Notwendig, wenn zwischen zwei Klassen mehrere Assoziationen bestehen
- Rollennamen (Substantive) bevorzugen
- Rollennamen sind bei reflexiven Assoziationen immer notwendig

**7 Liegt eine 1:1-Assoziation vor?**

- Zwei Klassen sind zu modellieren, wenn...
  - die Verbindung in einer oder beiden Richtungen optional ist und sich die Verbindung
  - zwischen beiden Objekten ändern kann
  - es sich um zwei umfangreiche Klassen handelt
  - die beiden Klassen eine unterschiedliche Semantik besitzen.





## Checkliste Assoziationen

**8 Gibt es zwischen 2 Klassen mehrere Assoziationen?**

- Prüfen, ob die Assoziationen
  - eine unterschiedliche Bedeutung besitzen oder/und
  - unterschiedliche Kardinalitäten haben

**9 Sind abgeleitete Assoziationen korrekt verwendet?**

- Fügen keine neue Information zum Modell hinzu
- Lassen sich mittels Objektdiagrammen erkennen

**10 Assoziative Klasse oder eigenständige Klasse?**

- Assoziative Klasse betont die Assoziation zwischen den beteiligten Klassen
- Assoziative Klassen lassen sich in eigenständige Klassen wandeln

**11 Fehlerquellen**

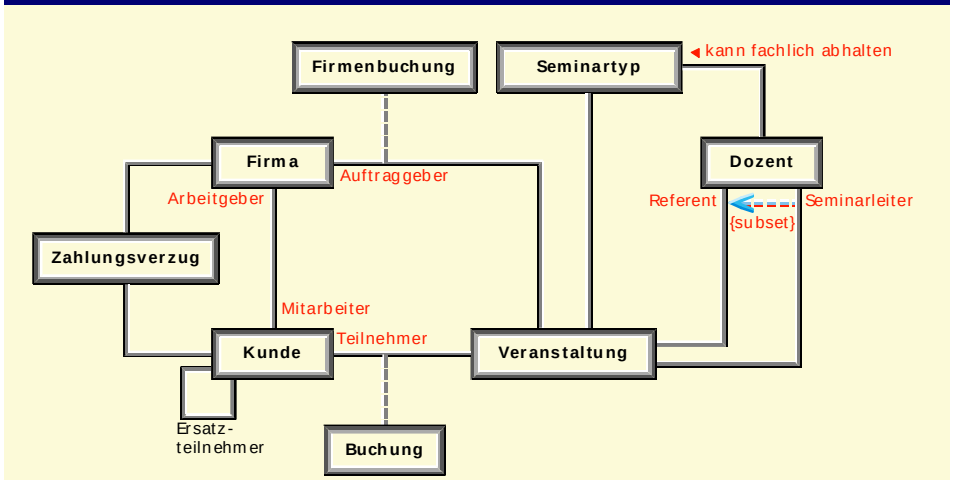
- Verwechseln von Assoziation mit Vererbung.





## 2.18.3.4 Checkliste Assoziation

- Identifizierte Assoziationen in der Fallstudie »Seminarorganisation«







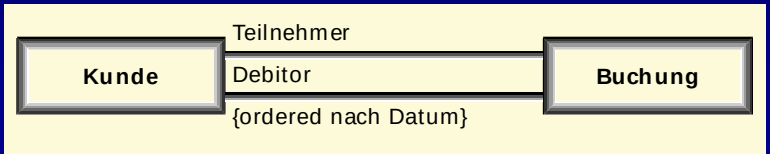

## 2.18.3.4 Checkliste Assoziation

- **3 Kategorien**
  - A ist logische Komponente von B:
    - Veranstaltung zu Seminartyp
  - A ist ein Mitglied von B
    - Kunde zu Firma
- **4 Restriktionen**
  - Aus dem Pflichtenheft ist *nicht* ersichtlich, ob ein Seminarleiter gleichzeitig auch Referent auf derselben Veranstaltung sein muss
  - Eine Rückfrage beim Auftraggeber ergibt, dass diese Restriktion gilt:
    - Der Seminarleiter muss Referent sein, aber außer ihm kann es noch andere Referenten geben.

LE 13 49

## 2.18.3.4 Checkliste Assoziation

- Klasse Zahlungsverzug gewandelt in eine Assoziation →SemOrg




- →SemOrg.



LE 13 50

## Checkliste Attribute

### Ergebnisse

- Klassendiagramm
  - Für jedes Attribut den Namen in das Klassendiagramm eintragen
  - Klassenattribute und abgeleitete Attribute kennzeichnen
- Attributspezifikation
  - Jedes Attribut spezifizieren
  - Für komplexe Attribute sind ggf. entsprechende Typen zu definieren.



LE 13 51

## Checkliste Attribute

### Konstruktive Schritte

1 Attribute durch Dokumentanalyse identifizieren
- Einfache Attribute sind ggf. zu Datenstrukturen zusammenzufassen
- Prüfen, ob alle Attribute wirklich notwendig sind
- Für jedes Attribut prüfen, ob es »im Laufe seines Lebens« einen Wert annehmen kann und ob diese Werte an der Benutzungsoberfläche sichtbar sind

2 Attribute durch Geschäftsprozesse identifizieren
- Benötigte Daten zur Ausführung der Aufgaben eines Geschäftsprozesses
- Benötigte Daten für Listenfunktionalität.



LE 13 52

## Checkliste Attribute

3 Wurden geeignete Attributtypen gewählt und u.U. als elementare Klasse beschrieben?
- Vorgegebene Typen *nur* verwenden, wenn problemadäquat
- Attribute beliebigen Typs definieren, um problemadäquate Modellierung auf ausreichendem Abstraktionsniveau zu erreichen
- Für komplexe Attribute zur Konstruktion der Typen das Klassenkonzept verwenden (elementare Klassen).

## Checkliste Attribute

## Analytische Schritte

**4 Ist der Attributname geeignet?**

- Der Attributname soll
  - kurz, eindeutig und verständlich sein
  - ein Substantiv oder Adjektiv-Substantiv sein
  - den Namen der Klasse nicht wiederholen
  - bei komplexen (strukturierten) Attributen der Gesamtheit der Komponenten entsprechen
  - nur fachspezifische oder allgemein übliche Abkürzungen enthalten, z.B. PLZ

**5 Klasse oder komplexes Attribut?**

- Klasse:
  - Objektidentität, gleichgewichtige Bedeutung im System, Existenz unabhängig von der Existenz anderer Objekte, Zugriff in beiden Richtungen grundsätzlich möglich.





## Checkliste Attribute

- Attribut: keine Objektidentität, Existenz abhängig von Existenz anderer Objekte, Zugriff immer über das Objekt, untergeordnete Bedeutung

**6 Wurde das richtige Abstraktionsniveau gewählt?**

- Wurden komplexe Attribute gebildet?
- Bilden Attribute geeignete Datenstrukturen?
- Anzahl der Attribute pro Klasse angemessen?

**7 Gehört Attribut zur Klasse oder einer Assoziation?**

- Test:
  - Muss Attribut auch dann zur Klasse gehören, wenn die Klasse isoliert von anderen Klassen betrachtet wird?
    - Wenn ja, dann gehört das Attribut zu dieser Klasse
    - Wenn nein, dann ist zu prüfen, ob es sich einer Assoziation zuordnen lässt
    - Sonst: Klasse oder Assoziation vergessen.





## Checkliste Attribute

**8 Liegen Klassenattribute vor?**

- Ein Klassenattribut liegt vor, wenn gilt:
  - Alle Objekte der Klasse besitzen für dieses Attribut denselben Attributwert
  - Es sollen Informationen über die Gesamtheit der Objekte modelliert werden

**9 Sind Schlüsselattribute fachlich notwendig?**

- Schlüsselattribute werden nur dann eingetragen, wenn sie Bestandteil des Fachkonzepts sind

**10 Werden abgeleitete Attribute korrekt verwendet?**

- Information ist für den Benutzer sichtbar
- Lesbarkeit wird verbessert.





## Checkliste Attribute

**11 Wann wird ein Attribut nicht eingetragen?**

- Dient nur zum Identifizieren der Objekte
- Dient lediglich dazu, eine andere Klasse zu referenzieren, d.h. es realisiert eine Assoziation
- Beschreibt den internen Zustand eines Lebenszyklus und ist außerhalb des Objekts nicht sichtbar
- Beschreibt Entwurfs- / Implementierungsdetails
- Ist abgeleitetes Attribut, das nur eingefügt wurde, um Berechnungszeit zu sparen

**12 Fehlerquellen**

- Verwenden atomarer Attribute anstelle von komplexen Datenstrukturen
- Formulieren von Assoziationen als Attribute.

## 2.18.3.5 Checkliste Attribut

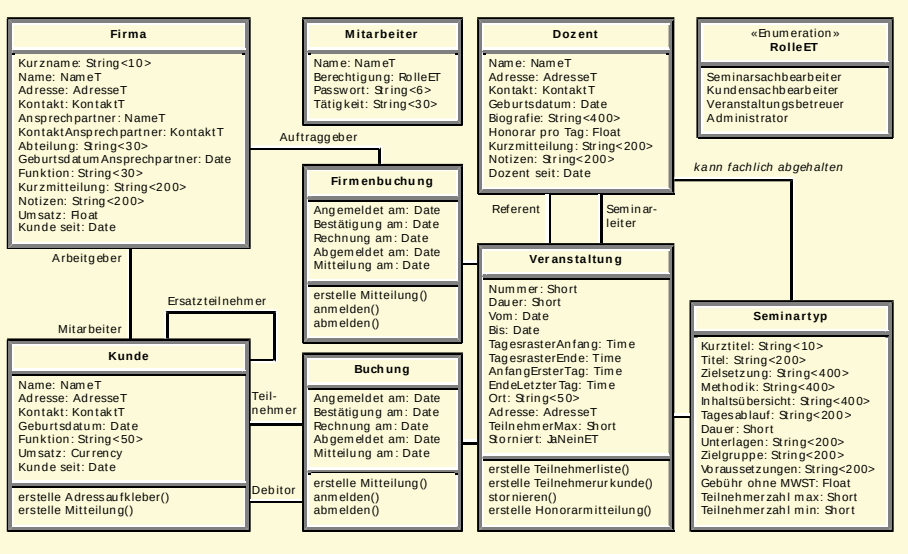
- 3 Selbstdefinierte Typen
  - Für die Seminarorganisation werden folgende elementare Klassen definiert:
    - Strukturierte Typen:
      - NameT, AdresseT, KontaktT, BankverbindungT
    - Aufzählungstypen:
      - AnredeET, TitelET, LandET, RolleET.





## 2.18.3.5 Checkliste Attribut

- Attribute der Seminarorganisation

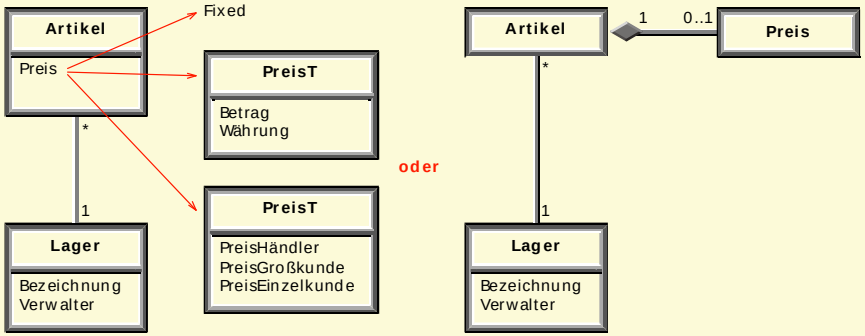







## 2.18.3.5 Checkliste Attribut

- Attribut oder Klasse




- →SemOrg.





## Checkliste Vererbung

### Ergebnis

- Klassendiagramm
  - Alle Vererbungsstrukturen eintragen
  - Abstrakte Klassen bilden

### Konstruktive Schritte

1 Ergibt Generalisierung eine Einfachvererbung?
- Neue Oberklasse aus gleichartigen Klassen?
- Vorhandene Klasse als Oberklasse geeignet?

2 Ergibt Spezialisierung eine Einfachvererbung?
- Kann jedes Objekt der Klasse für jedes Attribut einen Wert annehmen?
- Kann jede Operation auf jedes Objekt der Klasse angewendet werden?

# Checkliste Vererbung

## Analytische Schritte

**3 Liegt eine »gute« Vererbungsstruktur vor?**

- Verbessert die Vererbungsstruktur das Verständnis des Modells?
- Benötigt jede Unterklasse alle geerbten Attribute, Operationen und Assoziationen?
- Liegt eine Ist-ein-Beziehung vor?
- Entspricht die Vererbungsstruktur den »natürlichen« Strukturen des Problembereichs?
- Besitzt sie maximal 3 bis 5 Hierarchiestufen?

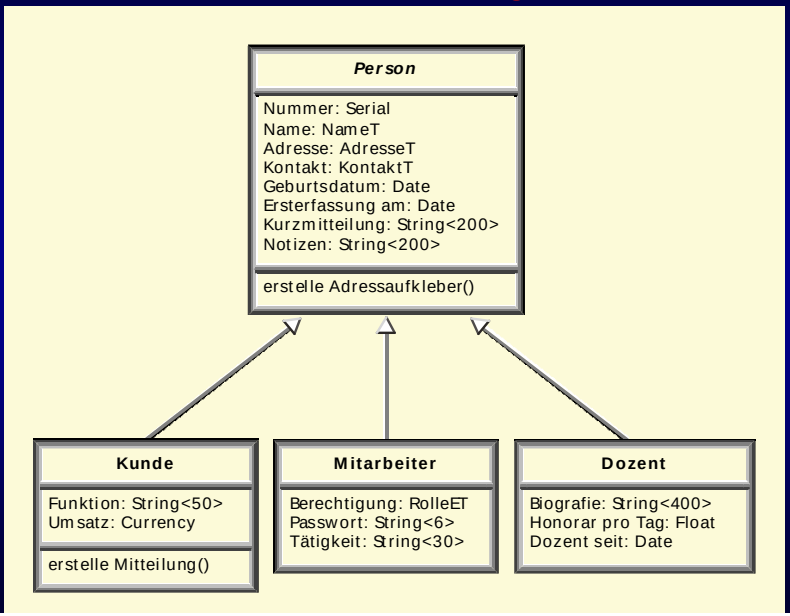
**4 Wann liegt keine Vererbung vor?**

- Die Unterklassen bezeichnen nur verschiedene Arten, unterscheiden sich aber weder in ihren Eigenschaften noch in ihrem Verhalten.





# 2.18.3.5 Checkliste Vererbung

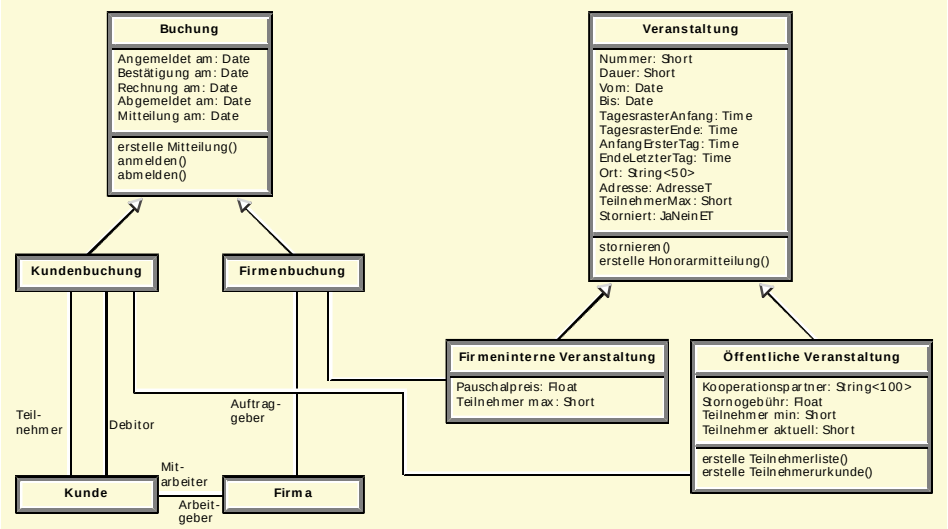







# 2.18.3.5 Checkliste Vererbung

- Ermittelte Vererbungsstrukturen →SemOrg








# Checkliste Kardinalitäten

## Ergebnis

- Klassendiagramm
  - Alle Kardinalitäten eintragen

## Konstruktive/analytische Schritte

**1 Schnappschuss oder Historie modellieren?**

- Aus den Anfragen an das System ergibt sich, ob
  - ein Schnappschuss (1- bzw. 0..1-Kardinalität) oder
  - die Historie (*many*-Kardinalität)
- zu modellieren ist
  - **Schnappschuss:** eine alte Verbindung wird gelöst wird, wenn eine neue aufgebaut wird
  - **Historie:** eine neue Verbindung wird zwischen den jeweiligen Objekten hinzugefügt.

## Checkliste Kardinalitäten

2 Liegt eine Muss- oder Kann-Assoziation vor?

3 Enthält die Kardinalität feste Werte?

- Ist eine Obergrenze vom Problembereich her (oft bei technischen Systemen) zwingend vorgegeben?
  - Im Zweifelsfall mit variablen Obergrenzen arbeiten
- Ist die Untergrenze vom Problembereich her zwingend vorgegeben?
  - Im Zweifelsfall mit »0« arbeiten
- Gelten besondere Restriktionen für die Kardinalitäten?

4 Fehlerquelle

- Oft werden Muss-Assoziationen verwendet, wo sie nicht benötigt werden.





## 2.18.3.7 Checkliste Kardinalitäten

- Seminarorganisation
  - Bei allen Assoziationen: Historie modellieren
  - Kardinalitäten aus dem Pflichtenheft:
    - Zwischen der Buchung und dem Kunden sowie der Veranstaltung liegt jeweils eine Muss-Beziehung vor, da eine Buchung für sich alleine nicht existieren kann
      - Ursprünglich: Assoziative Klasse
    - Ein Kunde kann mehrere Buchungen vornehmen (*), zu einer Veranstaltung kann es m Buchungen geben (*)
    - Jeder Zahlungsverzug gehört zu genau einem Kunden bzw. einer Firma, ein Kunde bzw. eine Firma kann mehrere Zahlungsverzüge besitzen.





## 2.18.3.7 Checkliste Kardinalitäten

- Zu jeder Veranstaltung gibt es einen oder mehrere Dozenten, die Seminarleiter sind
- Jede Veranstaltung gehört zu genau einem Seminartyp
- Eine Veranstaltung muss von mindestens einem Dozenten als Referent und einem Dozenten als Seminarleiter durchgeführt werden
- Ein gebuchter Kunde kann bei Abmeldung genau einen Ersatzteilnehmer melden
- Alle anderen Beziehungen sind Kann-Beziehungen mit variabler Obergrenze
  - Feste Grenzen
  - Durch Problemstellung: keine festen Grenzen
- →SemOrg.





## Checkliste Aggregation und Komposition

### Ergebnis

- Klassendiagramm
  - Alle Aggregationen und Kompositionen eintragen

### Konstruktive/Analytische Schritte

1 Für eine Komposition gilt:

- Es liegt eine Ist-Teil-von-Beziehung vor
- Kardinalität bei der Aggregatklasse: 0..1 oder 1
  - Lebensdauer der Teile ist an die des Ganzen gebunden
  - Ein Teil darf jedoch zuvor explizit entfernt werden
- Funktionen des Ganzen werden automatisch auf seine Teile angewendet
- Muster bilden eine gute Orientierungshilfe
  - Liste (Bestellung – Bestellposition), Baugruppe (Auto – Motor) und Stückliste mit physikalischem Enthaltensein.

## Checkliste Aggregation und Komposition

2 Für eine Aggregation gilt:
- Es liegt eine Ist-Teil-von-Beziehung mit *shared aggregation* (ein Teilobjekt kann mehreren Aggregatobjekten zugeordnet werden) vor
- Sie ist selten

3 Im Zweifelsfall: immer einfache Assoziation
- Bei dem geringsten Zweifel an einer Komposition oder Aggregation immer die Assoziation wählen

4 Fehlerquellen
- Prinzipiell ist es möglich, jedes Attribut als Klasse zu modellieren und mittels einer Komposition mit der ursprünglichen Klasse zu verbinden
  - Dies führt jedoch zu schlechten Modellen.





## 2.18.3.8 Checkliste Aggregation & Komposition

- Seminarorganisation
  - Eine Komposition liegt nicht vor
    - In erster Näherung könnte es sich bei Seminartyp – Veranstaltung um eine Komposition handeln
      - Nicht erfüllt: Funktionen des Ganzen werden automatisch auf seine Teile angewandt
  - Aggregation
    - Eine Aggregation liegt nicht vor
      - Bei Seminartyp – Veranstaltung liegt keine Ist-Teil-von-Beziehung vor
  - Im Zweifel Assoziation
    - Alle Beziehungen sind »einfache« Assoziationen.





## Makroprozess »Statisches Modell«

- 7 Muster identifizieren
  - Folgende Muster lassen sich identifizieren
    - Exemplartyp: Seminartyp – Veranstaltung
    - Koordinator: Kundenbuchung, Firmenbuchung
    - Rollen: Dozent
    - Gruppe: Firma - Kunde
  - Alle Muster sind korrekt angewendet
- →SemOrg.





## Checkliste Szenarios

### Ergebnisse

- Sequenzdiagramm, Kollaborationsdiagramm
  - Für jedes relevante Szenario: Sequenzdiagramm
  - Alternativ: Kollaborationsdiagramme

### Konstruktive Schritte

1 Aus jedem Geschäftsprozess (GP) mehrere Szenarios entwickeln
- Variationen von GPs ermitteln
- Standardausführung und Alternativen
- Positive und negative Fälle unterscheiden
- Prüfen, welche Szenarios wichtig sind
- Diagramme benennen und beschreiben.

## Checkliste Szenarios

2 Ablauf relevanter Szenarios durch Sequenz- oder Kollaborationsdiagramme beschreiben

- Beteiligte Klassen
- Aufgaben in Operationen zerlegen
- Reihenfolge der Operationen prüfen
- UML erlaubt es, im Sequenzdiagramm Bedingungen anzugeben
  - Damit können mehrere Variationen durch ein einziges Sequenzdiagramm beschrieben werden

3 Operationen den Klassen zuordnen

- Werden nur Attribute einer Klasse benötigt, dann ist die Operation dieser Klasse zuzuordnen
- Konstruktoren sind der jeweiligen Klasse selbst zuzuordnen.





## Checkliste Szenarios

### Analytische Schritte

4 Sind die Empfänger-Objekte erreichbar?

- Assoziation existiert (permanente Verbindung)
- Identität kann dynamisch ermittelt werden (temporäre Verbindung)

5 Sequenzdiagramm konsistent mit Klassendiagr.?

- Alle Klassen auch im Klassendiagramm enthalten
- Mit Ausnahme von impliziten Operationen werden nur Operationen aus dem Klassendiagramm eingetragen

6 Fehlerquellen

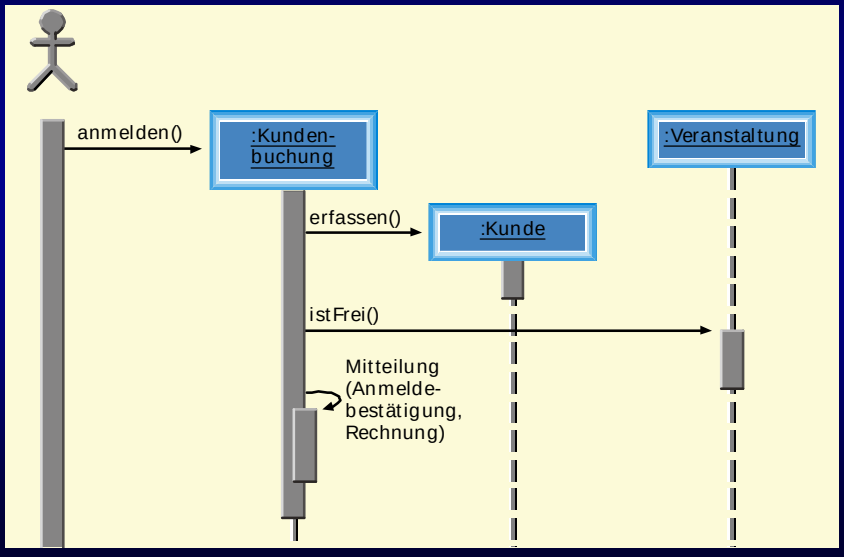
- Benutzungsoberfläche wird beschrieben
- Zu viele Details sind beschrieben.





## 2.18.3.9 Checkliste Szenario

- Seminarorganisation →SemOrg








## Checkliste Zustandsautomaten

### Ergebnis

- Zustandsdiagramm

### Konstruktive Schritte

1 Existiert ein nicht-trivialer Lebenszyklus?

2 Welche Zustände enthält der Automat?

- Ausgangsbasis ist der Anfangszustand
- Durch welche Ereignisse wird ein Zustand verlassen?
  - Ereignisse umgangssprachlich beschreiben, was »von außen auf das Objekt einwirkt«
- Welche Folgezustände treten auf?
- Wodurch wird der Zustand definiert (Attributwerte, Objektverbindungen)?

## Checkliste Zustandsautomaten

3 Existieren Endzustände?
- Nur bei linearen Lebenszyklen:
  - Das Objekt hört auf zu existieren
  - Das Objekt existiert weiterhin, aber sein dynamisches Verhalten ist nicht länger von Interesse (schlafendes Objekt)

4 Welche Operationen besitzt das Objekt?
- Jede zustandsabhängige Operation aus dem Klassendiagramm eintragen
- Operationen, die in jedem Zustand ausgeführt werden können, nicht eintragen
- Prüfen, ob weitere Operationen notwendig sind

5 Sind Operationen als Aktivitäten oder als Aktionen zu modellieren?





## Checkliste Zustandsautomaten

6 Welche Ereignisse sind zu modellieren?
- Externe Ereignisse:
  - vom Benutzer
  - von anderen Objekten
- Zeitliche Ereignisse:
  - Zeitdauer
  - Zeitpunkt
- Intern generierte Ereignisse des betrachteten Objekts
- Welche Fehlersituationen können auftreten und wie soll das Objekt darauf reagieren (Ausnahmebehandlung)
  - Zuerst immer das Normalverhalten und erst im zweiten Schritt das Fehlerverhalten betrachten.





## Checkliste Zustandsautomaten

### Analytische Schritte

7 Geeigneter Zustandsname
- Beschreibt eine bestimmte Zeitspanne
- Kein Verb
- Kann entfallen, wenn er keine zusätzliche Information enthält

8 Ist der Objekt-Lebenszyklus konsistent mit der Liste der Operationen?
- Gibt es für jede Operation mindestens einen Zustand, in dem das Objekt auf die entsprechende Botschaft reagieren kann?
- Sind alle Aktivitäten und Aktionen auch Operationen des Klassendiagramms?

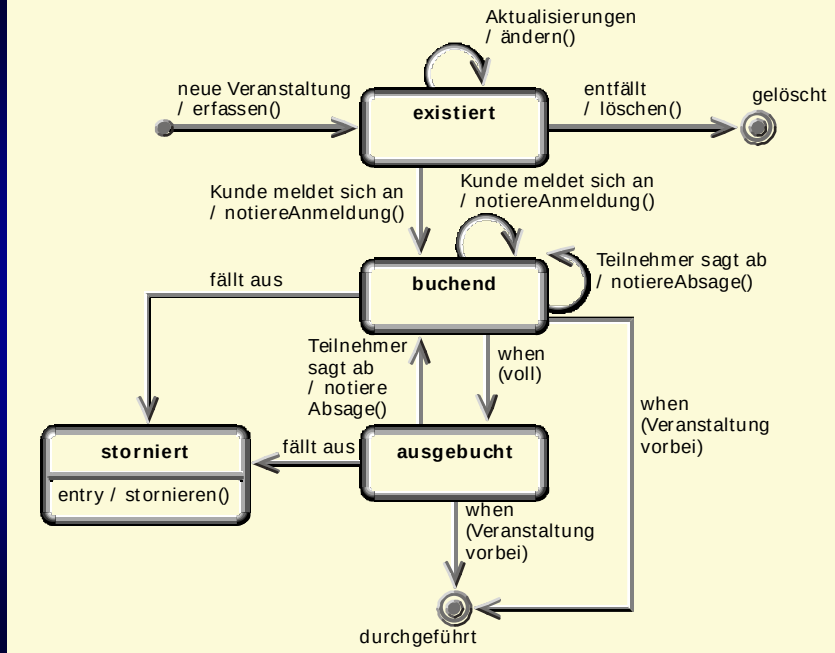




## Checkliste Zustandsautomaten

9 Sind alle Zustandsübergänge korrekt eingetragen?
- Ist jeder Zustand erreichbar?
- Kann jeder Zustand – mit Ausnahme der Endzustände – verlassen werden?
- Sind die Zustandsübergänge eindeutig?

10 Fehlerquellen
- Modellierung der Benutzungsoberfläche im Lebenszyklus
- Gedankengut aus den Programmablaufplänen übernommen
  - »Schleifen« dürfen nicht vorkommen
  - Bedingungen sind immer mit einem Ereignis verknüpft.








## 2.18.3.10 Checkliste Zustandsautomat

- Seminarorganisation →SemOrg
  - Zustandsmatrix der Veranstaltung

| Ereignis → / Zustand ↓ | Kunde meldet sich an | Teilnehmer sagt ab | ist voll | fällt aus | Veranstaltung vorbei |
|---|---|---|---|---|---|
| existiert | buchend | i | n.m. | i | i |
| buchend | buchend | buchend | ausgebucht | storniert | durchgeführt |
| ausgebucht | i | buchend | n.m. | storniert | durchgeführt |
| storniert | i | i | n.m. | i | i |
| durchgef. | i | i | n.m. | i | i |

Legende: i: Ereignis wird ignoriert  n.m.: Ereignis nicht möglich





## Checkliste Operationen

### Ergebnisse

- Klassendiagramm: Operationen eingetragen
- Beschreibung der Operationen

### Konstruktive Schritte

1 Operationen ins Klassendiagramm eintragen
- Aus Szenarios & Zustandsautomat. übernehmen
- Listenoperationen hinzufügen
- Keine Verwaltungsoperationen eintragen

2 Vererbung von Operationen berücksichtigen
- Operationen so hoch wie möglich in der Hierarchie eintragen

3 Beschreibungen erstellen (nur wenn nötig).





## Checkliste Operationen

### Analytische Schritte

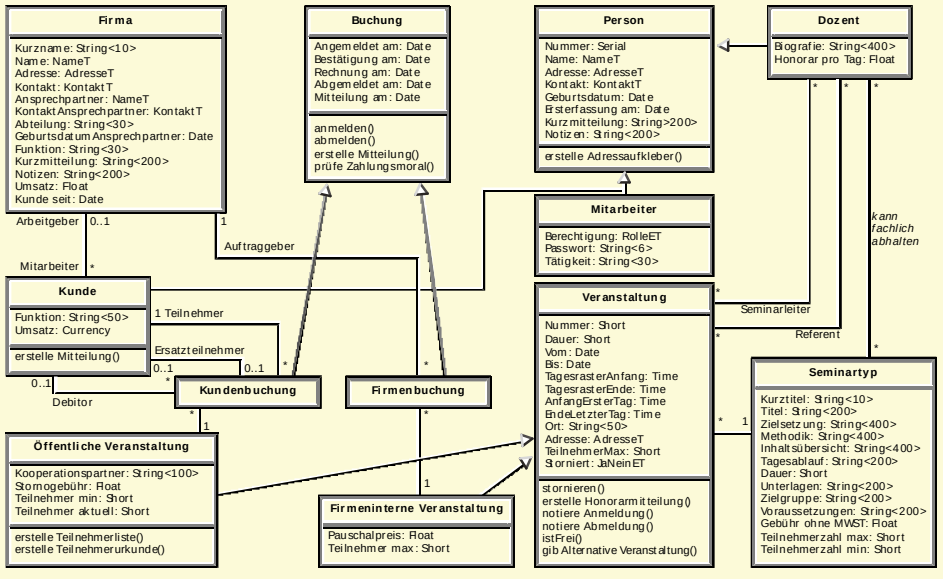
4 Besitzt die Operation einen geeigneten Namen?
- Beginnt mit einem Verb
- Beschreibt, was die Operation »tut«

5 Erfüllt jede Operation die Qualitätskriterien?
- Angemessener Umfang
- Funktionale Bindung, d.h. jede Operation realisiert eine in sich abgeschlossene Funktion

6 Ist das balancing erfüllt?
- Alle Attribute werden von den Operationen benötigt

7 Fehlerquellen
- Keine Benutzungsoberfläche.

# 2.18.3.11 Checkliste Operation








- Danke!
- Aufgaben

- Diese Präsentation bzw. Teile dieser Präsentation enthalten Inhalte und Grafiken des Lehrbuchs der Software-Technik (Band 1), 2. Auflage von Helmut Balzert, Spektrum Akademischer Verlag, Heidelberg 2001