19 Objektorientierter Entwurf

19.1 Grundlagen der Objektorientierung

Ein Vorgehen heißt objektorientiert, wenn es

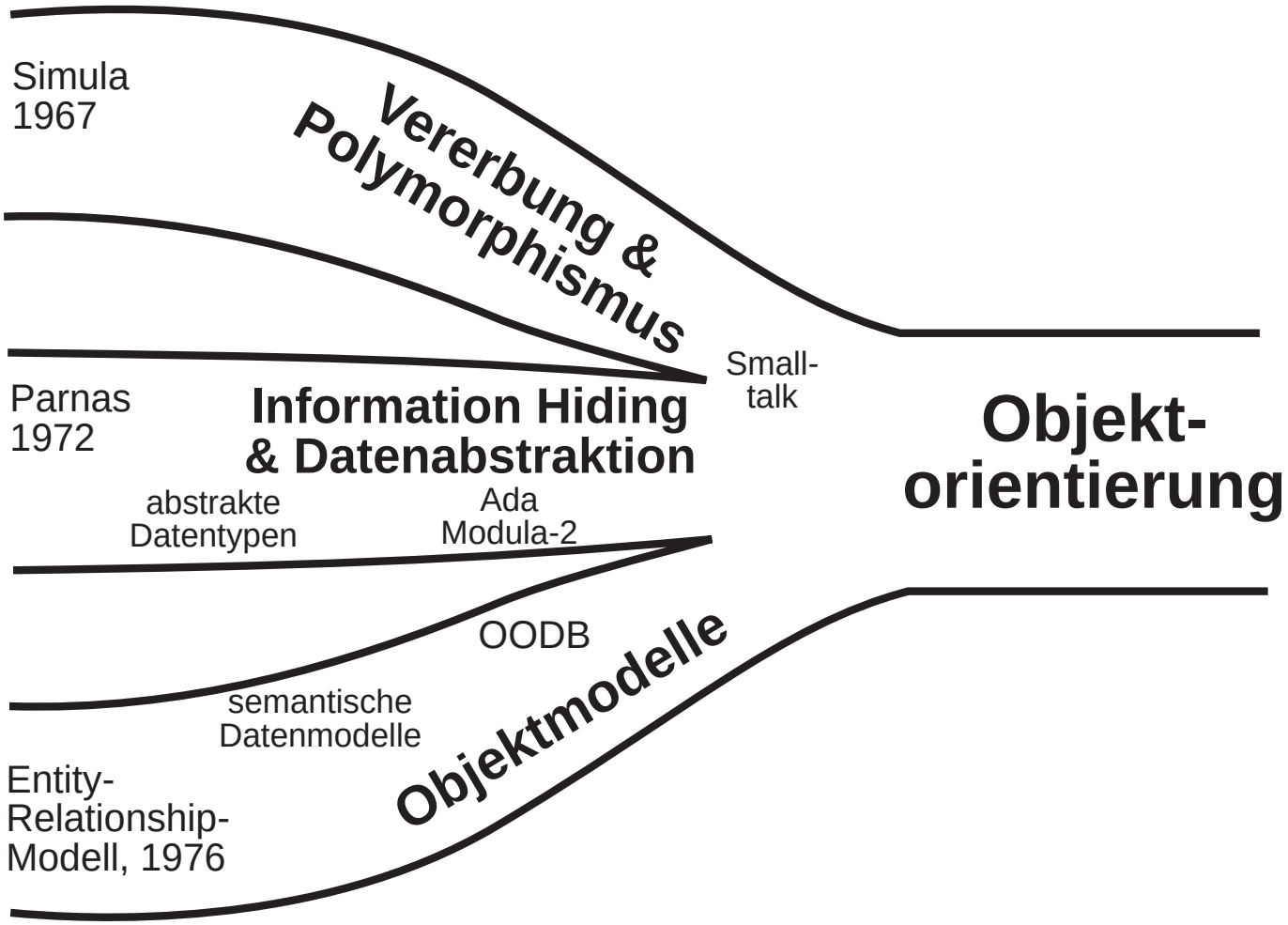
- ☆ sich bei der Modellbildung an den Gegenständen der Realität orientiert
- ☆ die Modelle auf der Grundlage von Information Hiding und Datenabstraktion aufbaut
- ☆ Vererbung und Polymorphismus unterstützt.

Objekt (object) – Ein individuell erkennbares, von anderen Objekten eindeutig unterscheidbares Element der Realität.

Gleichartige Objekte werden durch *Klassen* modelliert.

Klasse (class) – eine eindeutig benannte Einheit, welche eine Menge gleichartiger Objekte beschreibt.

Zur Geschichte:



19.2 Klassen- und Objektmodelle

Eine Klasse beschreibt den Aufbau, die Bearbeitungsmöglichkeiten und das mögliche Verhalten von Objekten dieser Klasse.

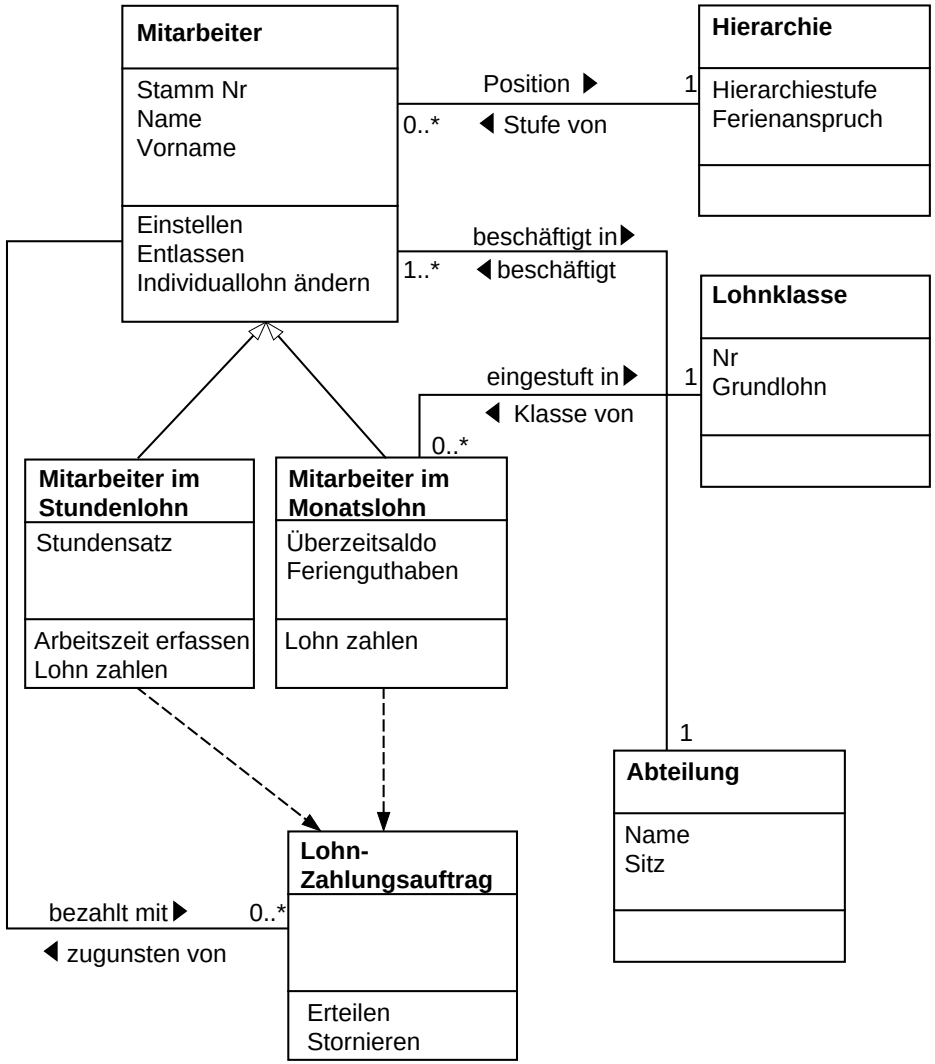
Eine Klassendefinition besteht aus

- Definition der Attribute der Klasse (lokale Merkmale)
- Definition der Beziehungen zu anderen Klassen
- Definition der Operationen, die auf Objekten der Klasse oder auf der Klasse selbst möglich sind

Klassen stehen in Beziehung zueinander:

- **Assoziation**: Die Objekte einer Klasse sind Merkmale von Objekten einer anderen Klasse. Gilt oft auch umgekehrt, d.h. Assoziationen sind häufig 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

Beispiel:



Klassenmodell

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

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

Klassendefinition

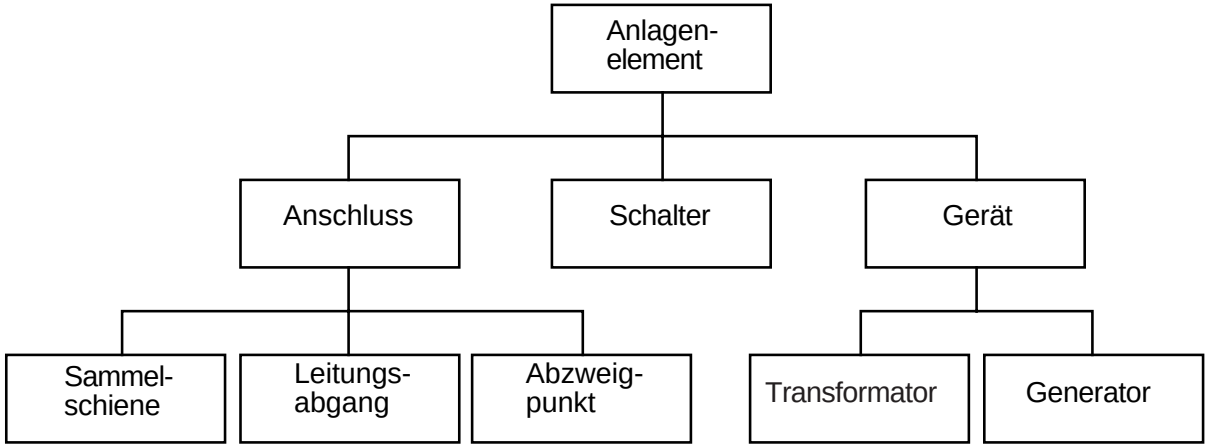
- **Klassenmodelle** bestehen aus einer Menge von Klassen und ihren wechselseitigen Beziehungen. Die Übersicht über die vorhandenen Klassen und ihre Beziehungen wird in der Regel mit graphischen Notationen modelliert, während für die Klassendefinitionen selbst strukturierter Text verwendet wird.

- **Objektmodelle** können als Alternative oder als Ergänzung zu Klassenmodellen verwendet werden. In beiden Fällen werden keine konkreten Objekte (d.h. solche mit bekannter Identität und bekannten Attributwerten) modelliert, sondern abstrakte Objekte, die als Repräsentant für konkrete Objekte stehen.

19.3 Vererbung und Polymorphismus

Vererbung (inheritance) – Relation, welche eine Menge von Klassen in eine Hierarchie von Ober- und Unterklassen gliedert, so dass jede Unterklasse neben ihren eigenen Eigenschaften auch alle Eigenschaften aller Oberklassen hat, sofern sie diese nicht explizit abändert oder ausschließt. Die Eigenschaften einer Klasse sind ihre Attribute, Beziehungen und Operationen.

Beispiel: Vererbungshierarchie von Anlagenelementen:



Polymorphismus (polymorphism) – Möglichkeit, eine Operation gleichen Namens für Objekte verschiedener Klassen mit möglicherweise verschiedener Wirkung zu definieren.
Wird eine solche Operation auf ein Objekt angewendet, so muss aus dem Kontext bestimmt werden, welche Operationsdefinition zu verwenden ist.

Polymorphismus und Vererbung

Wird eine Operation nur innerhalb einer Klassenhierarchie mehrfach definiert, so bietet die Vererbung eine einfache Möglichkeit zur Bestimmung der anzuwendenden Operationsdefinition:
Sei A der Name einer Operation, welche auf ein Objekt x der Klasse K anzuwenden ist. Ausgehend von der Klasse K wird die Klassenhierarchie aufsteigend nach Definitionen für A durchsucht. Die erste gefundene Definition wird verwendet.

Nutzen

Programme werden durch Polymorphismus kürzer, flexibler und leichter erweiterbar. Häufig kann auf komplizierte Fallunterscheidungen verzichtet werden.

Beispiel:

Zum Zeichnen der verschiedenen Symbole für Anlagenelemente in Anlagenschemata gibt es für jede der fünf untersten Klassen in obiger Klassenhierarchie eine eigene Operation Zeichne, welche das zugehörige Symbol erzeugt. Durch den Polymorphismus wird nun beispielsweise folgende Konstruktion möglich:

```
...
Anlagenelement []   bildsymbole;   // Vektor mit verschiedenen Bildsymbolen
...
for (int index = 0; index<anzahlSymbole; index++)   // Zeichne Symbole
    {bildsymbole[index].zeichne;}
...
```

Jedes Objekt des Vektors Bildsymbole entscheidet aufgrund seiner Klassenzugehörigkeit, welches die für sich passende Zeichne-Operation ist und wendet diese an.

Ad-hoc Verwendung von Vererbung und Polymorphismus

Vererbung in primitiver Form: Modell- bzw. Codesteinbruch: Brauchbares nutzen, nicht Brauchbares umdefinieren, Fehlendes ergänzen

Folge: Unerwünschte Nebenwirkungen (Seiteneffekte, side effects). Resultierende Software ist fehlerträchtig, schwierig zu verstehen und schwierig zu pflegen

Beispiel:

Klasse *Quadrat* mit Operationen *Verschiebe*, *Skaliere*, *Drehe*, *Zeichne*

Von *Quadrat* wird neue Klasse *Rechteck* abgeleitet mit zusätzlichen Operationen *SkaliereBreite* und *SkaliereHöhe*

$\rightarrow$ Klasse *Quadrat* verliert die Eigenschaft, dass alle ihre Objekte Quadrate sind: jedes Objekt der Klasse *Rechteck* ist durch die Vererbung auch ein Objekt der Klasse *Quadrat*.

Wehe dem Anwender, der sich darauf verlässt, dass Quadrate quadratisch sind...

Vererbung und Polymorphismus als Spezialisierungsabstraktion

Vererbung und Polymorphismus nur zur Realisierung einer Spezialisierungs-abstraktion verwenden: Jedes Objekt x einer Klasse S ist ein Spezialfall eines Objekts jeder Oberklasse von S.

→ Keine Operation der Klasse S darf eine Invariante einer ihrer Oberklassen verletzen

Invariante (invariant) – Eigenschaft eines Objekts, die durch die Anwendung beliebiger Operationen nicht verändert wird

→ Möglichkeiten der Polymorphie eingeschränkt: geerbte Operation darf nur noch spezialisiert, aber nicht mehr beliebig abgeändert werden

⇨ Ein großer Teil der gefährlichen Nebenwirkungen der Vererbung wird so ausgeschlossen

Subtypen

Auch bei Spezialisierung kann die Verwendung von Objekten einer Unterklasse an Stellen, wo Objekte einer ihrer Oberklassen verlangt sind, zu unvorhergesehenem Fehlverhalten führen.

Vermeidung durch noch stärkere Beschränkung von Vererbung und Polymorphie: Subtypen:

Eine Klasse S ist ein **Subtyp** (sub-type) einer Klasse K, wenn an jeder Stelle, wo ein Objekt der Klasse K verwendet wird, auch ein Objekt der Klasse S verwendet werden kann (Substituierbarkeit).

Dies ist dann der Fall, wenn

- für jede geerbte Operation gilt
  - Die Voraussetzungen der Operation werden höchstens abgeschwächt, aber nicht verstärkt
  - Die Ergebniszusicherung der Operation wird höchstens verstärkt, aber nicht abgeschwächt
- keine Operation der Klasse S verletzt eine Invariante der Klasse K

Bedeutung von Vererbung und Polymorphismus

- Vereinfachung der Modelle (weniger Schreib- und Darstellungsaufwand)
- Wesentlich beschleunigte Software-Entwicklung, wenn vorhandene Klassenbibliotheken genutzt werden können
- Einfache Erweiterbarkeit bestehender Systeme
- Erleichterte Wiederverwendung und Anpassung bestehender Komponenten

Aber:

- ☆ Vererbung und Polymorphismus sind modell- und softwaretechnisch nur dann sauber, wenn sie als Spezialisierungsabstraktion eingesetzt werden
- ☆ Spezialisierung ist eine zusätzliche Dimension in der Modellierung und stellt erhöhte Anforderungen an das Denk- und Abstraktionsvermögen der Modellierer
- ☆ Überall, wo Implementierungen vererbt werden, bricht die Vererbung das Geheimnisprinzip
- ☆ Die Möglichkeit, mittels Polymorphismus Objektreferenzen an beliebige Stellen in einem System weiterzureichen und dort zu verwenden, bricht bei undisziplinierter Verwendung so gut wie alle Abstraktionsprinzipien, insbesondere das Prinzip der Benutzungshierarchie (Metapher der geschichteten virtuellen Maschinen)

19.4 Entwurfstechnik

- Vorhandenes nutzen
  - Von vorhandenen Klassen neue Klassen so ableiten, dass diese das gestellte Problem lösen
  - Ziel: möglichst viel vorhandene Software wiederverwenden und diese lediglich problemspezifisch anpassen
  - Mittel: Klassenbibliotheken, Muster, Rahmen
- Klassenmodell aus Anforderungsspezifikation übernehmen und ergänzen/anpassen
- Wo nötig: Neues Klassenmodell erstellen

Erstellung neuer Klassenmodelle

Gleiche Verfahren wie in objektorientierter Spezifikation

- Objektanalyse: Statische Struktur der "Realität" abbilden. Texte, Glossare, vorhandene Informationsstrukturen heranziehen
- Szenarienanalyse: Szenarien erstellen und im Rollenspiel durchspielen. Leistungsträger für die Systemreaktionen festlegen.
  Hilfsmittel: Interaktionsdiagramme, Klassenkarten, CRC-Analyse

Vorhandene Klassenmodelle nutzen

- Black-Box-Verwendung: Nutzung vorhandener Operationen
  Nebenwirkungsfrei, Information Hiding-kompatibel, weniger flexibel
- White-Box-Verwendung: Nutzung durch Ableitung von Unterklassen
  Nebenwirkungen möglich, bricht Information Hiding, extrem flexibel

Problem des objektorientierten Entwerfens:

Drei Abstraktionen beherrschen

- ☆ Delegation: Sich auf vorhandene Dienstleistungen abstützen
- ☆ Generalisierung: Gemeinsamkeiten im Dienstleistungsangebot verwandter Klassen in gemeinsame Oberklasse verlagern
- ☆ Komposition: Zusammengehörende Klassen zu Einheiten zusammenfassen

19.5 Repräsentation von Modellen

- ☆ Vielfalt von Modellen und Notationen, die in die Kategorie „Objekt- und Klassenmodell“ fallen

- ☆ Zum Beispiel: Booch (1994), Coad und Yourdon (1991a, b), Jacobson et al. (1992), Rumbaugh et al. (1991), Wirfs-Brock et al. (1990)

- ☆ Derzeit dominiert ein Quasi-Standard: Die Unified Modeling Language UML (Rumbaugh, Jacobson und Booch 1999, Oestereich 1998).

19.6 Rahmen

Rahmen (framework) – Eine Menge kooperierender Module, die das Grundgerüst für die Lösung einer bestimmten Klasse von Problemen bilden.

- Konkrete Lösungen entstehen durch Ergänzung/Spezialisierung des Rahmens durch problemspezifische Module.
- Werden Rahmen verwendet, so bestimmen diese weitestgehend die Grundarchitektur der Lösung.
- Der Umfang eines Rahmens kann stark variieren:
  - Eng: Lösung eines Teilproblems, z.B. Realisierung einer Benutzerschnittstelle, Verwaltung von Dateien mit Vergabe und Prüfung von Zugriffsrechten oder die Grundfunktionalität eines Editors
  - Umfassend: Lösungsgerüst für ein vollständiges System, z.B. Rahmen für das Schaltergeschäft in einer Bank.
- Manchmal werden in einer Architektur mehrere Rahmen gleichzeitig verwendet.
  - ➪ Sehr anspruchsvoll: Unterschiedliche Architekturprinzipien müssen in gemeinsame, kohärente Gesamtarchitektur integriert werden.