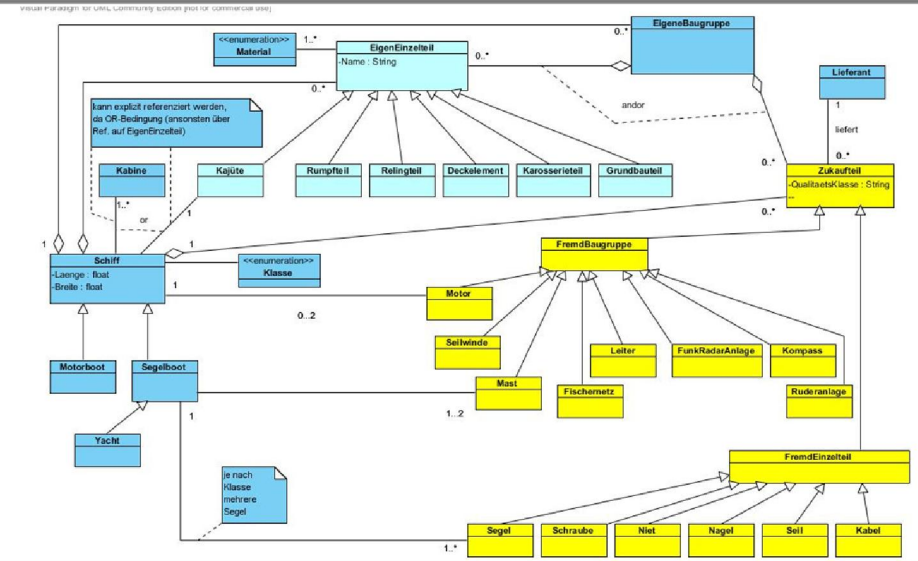
Karlsruher Institut für Technologie

DHBW
Duale Hochschule
Baden-Württemberg

Software Engineering I

Grundlagen der Objektorientierung
Einführung in die objektorientierte Datenmodellierung

Institut für Angewandte Informatik (IAI)



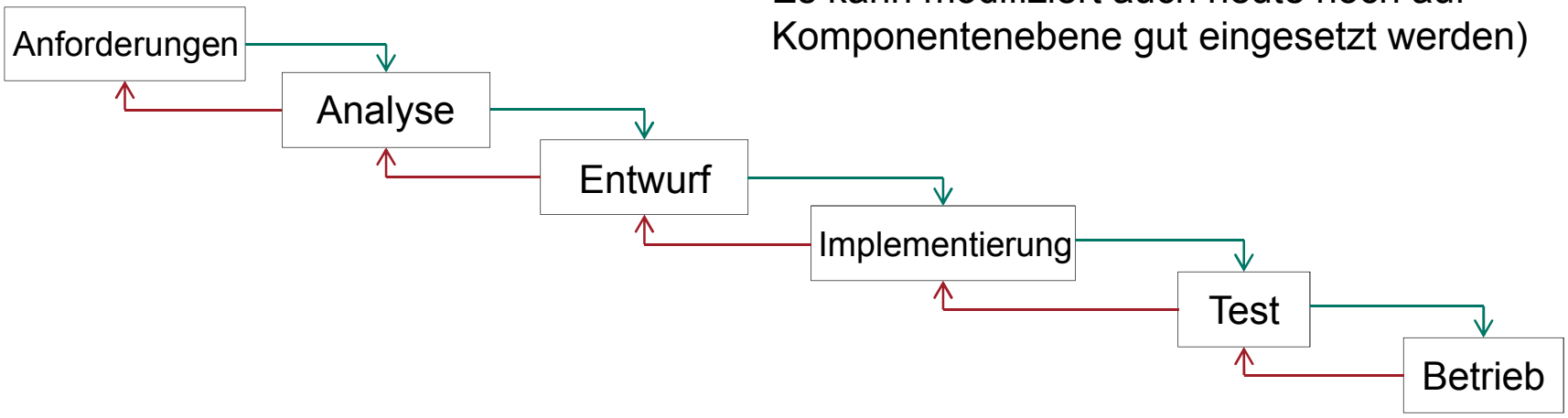
Historie

- Objektorientierte Programmierung
  - Seit den späten 50er bzw. frühen 60er Jahren bekannt
    (LISP, Simula 67, Smalltalk, später C++, C#, Java, …)
  - Beinhaltet Attribute, Funktionen sowie Verbindungen zu anderen Objekten

- Objektorientierte Modellierung
  - 1990: OOD (Object Oriented Design, Booch)
  - 1992: OMT (Object Modeling Technique, Rumbaugh)
    OOA (Object Oriented Analysis, Coad/Yourdon)
    OOSE (Object Oriented Software Engineering, Jacobsen)
  - 1997: UML (Unified Modeling Language, Booch/Jacobsen/Rumbaugh),
    standardisiert durch OMG (Object Management Group)
  - 2004: UML 2.0

Softwareentwicklungs-Prozess

- Jede Softwareentwicklung beinhaltet eine gründliche **Analyse** der Problemstellung!
  - Teilweise drastische Reduzierung von Zeit und Komplexität bei Entwurf, Implementierung und Test der späteren Software.
  - Analyse führt zur Identifikation der beteiligten Elemente und Zusammenhänge (Klassen, Objekte und Relationen)
- „Ur-Modell“ ist das Wasserfallmodell nach Royce und Böhm
  (bildet die Grundlage fast aller weiteren Vorgehensmodelle der Softwareentwicklung.
  Es kann modifiziert auch heute noch auf Komponentenebene gut eingesetzt werden)



Was ist ein Objekt?

- allgemein: eine Sache oder ein Gegenstand
- eine Einheit beim Programmieren
- eine Einheit beim Modellieren, d.h. Analysieren und Entwerfen
- ein Objekt kann konkret oder auch konzeptuell sein:

konkret

Datei in Dateisystem, Hund, Katze, Person, Haus, Computer, …

konzeptuell

Zuteilungsstrategie in einem Multiprozessor-Betriebssystem,
Buchung eines Flugs oder eines Zimmers,
Kauf eines Gegenstands, usw.

Was ist Objektorientierung?

- Methodik und Vorgehensweise beim Erstellen von Softwareprodukten, bei denen Objekte und deren Zusammenhänge im Mittelpunkt stehen
- Objekte vereinen Daten(-struktur) **und** Verhalten (Funktionalität) in sich
- Gegensatz:
  konventionelle, *prozedurale SW-Entwicklung (Programmierung)*
  - Funktionen und Strukturen stehen im Mittelpunkt
  - Datenstruktur und Verhalten sind nur lose miteinander verbunden

Was ist objektorientierte Modellierung?

- Möglichst genaue Abbildung eines Problembereichs der realen Welt in die Modellwelt.
- Wird für die Analyse des Problemumfelds und für den Entwurf der Software verwendet (A & E stehen im SW-Entwicklungsprozess grundsätzlich **vor** der Implementierung)
- Ab einer bestimmten Komplexität des Problemumfelds unerlässlich (vermeidet fehleranfälliges „happy hacking“)
- Ein Teil des Modells kann direkt in Quellcode umgewandelt werden.

Was ist der Vorteil von Objektorientierung?

- Aufgaben, Strukturen, Eigenschaften (Attribute) können klar getrennt und separat behandelt werden (Übersichtlichkeit)
- Gemeinsamkeiten können gebündelt werden (Vereinfachung)
- Ergebniselemente (Klassen) können leichter in anderen SW-Produkten verwendet werden

hohe Wiederverwendbarkeit!

Objekte, Klassen, Instanzen, Methoden, Operationen ???

Karlsruher Institut für Technologie

- Objekte:
  - reale Elemente in Alltag, Problemumfeld und Programm
- Instanzen:
  - Alternative Bezeichnung für Objekt (Programmierung, Modellierung)
  - Manchmal auch *Objektinstanzen* genannt
- Klassen:
  - Beschreibung (Definition, Schablone) eines Objekts.
  - Eine Klasse besteht aus Attributen (Eigenschaften bzw. Daten) und Methoden (Verhalten)
- Operation:
  - Beschreibung des Verhaltens von Objekten einer Klasse
- Methode:
  - Realisierung (Quellcode) einer Operation
  - Wird auch als alternativer Begriff für Operation verwendet

Herausforderung

- Wie kommt man von der realen Welt in die Modellwelt?
- Wie findet man die passenden Objekte, Klassen und Attribute?

???

Software

Klassen

Referenzen

Attribute

Funktionalität

Herausforderung 1: Identifizieren der Klassen

Laptop

Boot

Tasse

Herausforderung 2: Identifizieren der Attribute

- Attribute beschreiben die charakteristischen Eigenschaften einer Klasse

<u>Laptop:</u>
Bildschirmgröße, Farbe, Höhe, Breite, Tiefe, Ausstattung, …
an/aus, offen/zu, am Netz / nicht am Netz, …

<u>Boot:</u>
Form, Farbe, Höhe, Länge, Gewicht, Breite, Platzanzahl, Material, …
fahrend/stehend, leer/besetzt, …

<u>Tasse:</u>
Form, Farbe, Höhe, Aufdruck, Durchmesser, Material, …
leer/voll, heiß/kalt, bruchfest/zerbrechlich, …

Herausforderung 3: Identifizieren der Funktionalität

- Grundfrage: was macht das Objekt, wenn es „dazu beauftragt“ wird?

Laptop:
starten, herunterfahren,
Programm ausführen,
mit Netz verbinden, …

Boot:
losfahren, anhalten, beschleunigen,
wenden, …

Tasse:
???, …

Herausforderung 4: Identifizieren assoziierter Klassen

- Grundfrage: zu welchen anderen Objekten gibt es Verbindungen?

→ Laptop:
Tastatur, Festplatte, CPU, Touchpad, Bildschirm, Netzverbindung, Besitzer, …

→ Boot:
Motor, Sitze, Lenkrad, Kajüte, Fahrer, Besitzer, Mitfahrer, …

→ Tasse:
Besitzer, Deckel, Inhalt, …

Herausforderung 5: Identifizieren von Gemeinsamkeiten

- Grundfrage: welche Gemeinsamkeiten und Unterschiede gibt es für „gleichartige“ Objekte?

PKW:
Höhe, Breite, Länge, Leistung, Verbrauch, Gewicht,
Farbe, an/aus, offen/zu, …
fahren, stehen, tanken, beschleunigen, …
➔ Motor, Räder, Sitze, Besitzer, **Faltdach**, …

LKW:
Höhe, Breite, Länge, Leistung, Verbrauch, Gewicht,
Farbe, an/aus, offen/zu, …
fahren, stehen, tanken, beschleunigen, …
➔ Motor, Räder, Sitze, Besitzer, **Kran, Seilwinde,** …

Bus:
Höhe, Breite, Länge, Leistung, Verbrauch, Gewicht,
Farbe, an/aus, offen/zu, Platzanzahl, …
fahren, stehen, tanken, beschleunigen, …
➔ Motor, Räder, Sitze, Besitzer, **Mikrofon**, …

Wie werden die Objekte geeignet dargestellt?

- Objektdiagramm, Instanzendiagramm
  - Formale grafische Notation, um real vorhandene **Objekte** und deren Relationen zu anderen Objekten darzustellen.
  - Es können beliebig viele Objekte desselben Typs (Klasse) existieren.
  - Besonders nützlich, um Testfälle (vor allem Szenarien) zu dokumentieren und Beispiele zu diskutieren.

- Klassendiagramm
  - Grafische Notation, um **Klassen** und deren Relationen zu anderen Klassen darzustellen.
  - Schema, Muster oder Template (Schablone) zur Beschreibung vieler möglichen Objektinstanzen. Ein Klassendiagramm beschreibt Objektklassen.
  - Jede Klasse ist nur einmalig vorhanden.
  - Ein gegebenes Klassendiagramm entspricht einer unendlichen Menge von Instanzendiagrammen

Die 5 wichtigsten Aspekte der Objektorientierung

- Identität
- Klassifikation
- Vererbung
- Polymorphie (Polymorphismus)
- Kapselung

Die 5 wichtigsten Aspekte der OO: 1.) Identität

- Daten werden diskreten, unterscheidbaren Entitäten (Objekten) zugeordnet
- Jedes Objekt besitzt eine eigene inhärente ("ihm innewohnende") Identität, d.h. zwei Objekte sind klar voneinander unterscheidbar, selbst wenn alle ihre Attributwerte (wie Name oder Größe) identisch sind
- Jedes Objekt hat in einer Programmiersprache einen eindeutigen "*Anfasser*“ ➔ ermöglicht eindeutigen Zugriff
- Objektzugriffe
  - sind einheitlich und unabhängig vom Inhalt der Objekte
  - erlauben es, gemischte Objektkollektionen zu erzeugen (z.B. durch ein Dateisystem-Verzeichnis, das sowohl Dateien als auch Unterverzeichnisse enthält).

Die 5 wichtigsten Aspekte der OO: 2.) Klassifikation

- Objekte mit der gleichen Datenstruktur (***Attribute***) und dem gleichen Verhalten (***Operationen, Methoden***) können zu einer ***Klasse*** gruppiert werden
- Eine ***Klasse*** ist eine Abstraktion, die die Eigenschaften beschreibt, die für eine Anwendung wichtig sind, und den Rest ignoriert. Die Wahl von Klassen ist immer beliebig und hängt von der Anwendung ab
- Jede Klasse beschreibt eine möglicherweise unendliche Menge individueller Objekte.
- Jedes Objekt wird auch als eine Instanz seiner Klasse bezeichnet.
- Jede Instanz der Klasse besitzt eigene Werte für alle ihre Attribute, während sie die Attributnamen und Operationen mit anderen Instanzen der Klasse teilt.

Die 5 wichtigsten Aspekte der OO: 2.) Klassifikation

Beispiel:

Landfahrzeug

Wasserfahrzeug

Fahrzeug

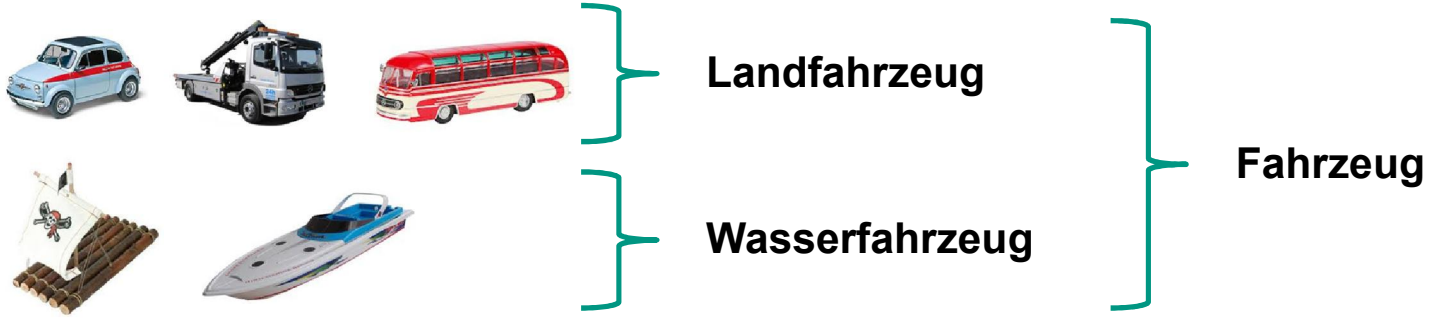
Die 5 wichtigsten Aspekte der OO: 3.) Vererbung

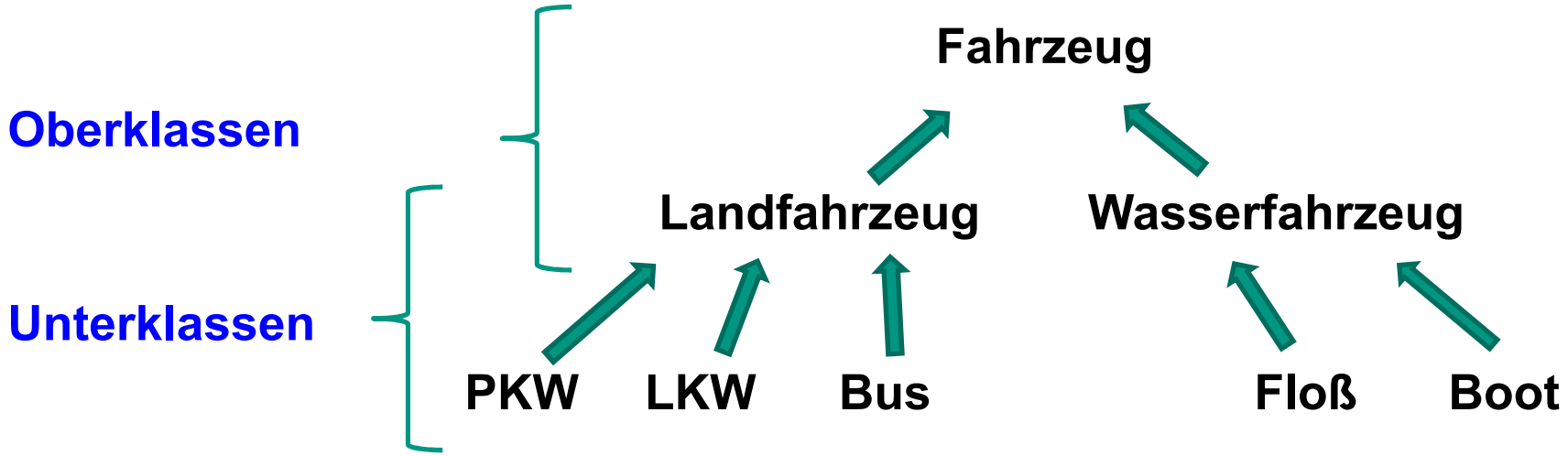
Karlsruher Institut für Technologie

- Mehrere Klassen mit denselben Attributen und/oder demselben Verhalten können durch eine gemeinsame Oberklasse repräsentiert werden (➔ *Generalisierung*, v.a. in Analysephase)
- Eine Klasse kann sehr allgemein definiert sein und dann in immer detailliertere Unterklassen verfeinert werden (➔ *Verfeinerung*, v.a. in Entwurfsphase)
- Jede Unterklasse übernimmt oder erbt alle Eigenschaften und Methoden ihrer Oberklasse und fügt ihre eigenen individuellen Eigenschaften und Methoden hinzu

Die 5 wichtigsten Aspekte der OO: 3.) Vererbung

Beispiel von vorhin:





Die 5 wichtigsten Aspekte der OO : 4.) Polymorphie

Polymorphie tritt immer im Zusammenhang mit Vererbung und Schnittstellen, d.h. mit Oberklassen und Interfaces auf

Die 2 wichtigsten Arten von Polymorphie:

- Polymorphie von Datentypen oder Klassen
- Polymorphie einer Methode bzw. Operation

Die 5 wichtigsten Aspekte der OO: 4.) Polymorphie

Polymorphie von Datentypen oder Klassen:

- besonderes Konzept der OOP. Es ist die Eigenschaft einer Variablen, für Objekte verschiedener Klassen stehen zu können („Platzhalter“)

Deklaration:

```
public class Oberklasse{
    . . .
    public void doSomething(){
        . . .
    }
}
```

```
public class UnterklasseEins
         extends Oberklasse{…}
```

```
public class UnterklasseZwei
         extends Oberklasse{…}
```

Erzeugung bzw. Verwendung:

```
public class IrgendeineKlasse{
    public Oberklasse okl;
    . . .
    okl = new UnterklasseEins();
    okl.doSomething();
    . . .
    okl = new UnterklasseZwei();
    okl.doSomething();
    . . .
}
```

Die 5 wichtigsten Aspekte der OO: 4.) Polymorphie

Polymorphie einer Methode bzw. Operation:

- Eine **Methode** ist **polymorph**, wenn sie in verschiedenen Klassen die gleiche Spezifikation hat, jedoch in der jeweiligen Klasse individuell implementiert ist (so genanntes Überschreiben)

```
public class Oberklasse{
    . . .
    public void doSomething(){
        . . .
    }
}
```

```
public class UK1 extends Oberklasse{
    . . .
    public void doSomething(){
      // spezifische Impl. für UK1
    }
    . . .
}
```

```
public class UK2 extends Oberklasse{
    . . .
    public void doSomething(){
      // spezifische Impl. für UK2
    }
    . . .
}
```

Die 5 wichtigsten Aspekte der OO: 5.) Kapselung

Karlsruher Institut für Technologie

- Unter Kapselung versteht man das Verbergen von Implementierungs-details, Daten oder Informationen vor dem Zugriff von außen.
- Der direkte Zugriff auf interne Daten(-struktur) wird unterbunden und erfolgt stattdessen über definierte Schnittstellen („Black-Box-Modell“)
- Objekte können den Zustand anderer Objekte nicht direkt unerlaubt lesen oder ändern. Das kann über so genannte Sichtbarkeitskriterien gesteuert werden (*private, public, protected,* etc.)

Zusammenfassung

- Erläuterung der Begriffe:
  - Objektorientierung: Objekt, Klasse, Instanz, Attribut, Operation, ..
  - Identifizieren von Objekten und Klassen
- Erklärung der 5 wichtigsten Aspekte der Objektorientierung
  - Identität
  - Klassifikation
  - Vererbung
  - Polymorphie (Polymorphismus)
  - Kapselung