Software-Engineering

Vorlesung 5 vom 15.11.2004

Sebastian Iwanowski
FH Wedel

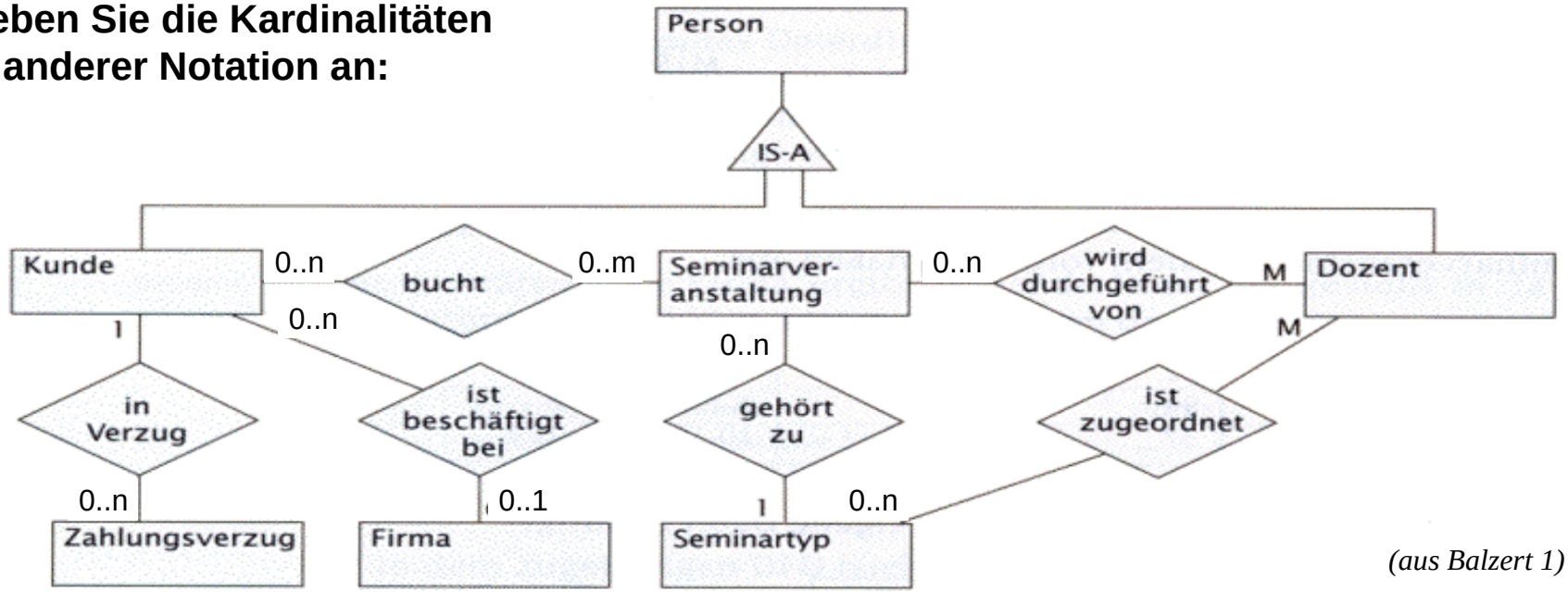
Software-Engineering

Vorlesungsthemen:

1. Überblick über das Thema und die Vorlesung
2. Grundlegende Prinzipien
3. Softwareplanung
4. Systemanalyse
5. Softwareentwurf
6. CASE-Tools
7. Aufwandsabschätzung
8. Qualitätsmanagement
9. Projektmanagement

Zum Einstieg: Lösung der ERM-Übung

1) Geben Sie die Kardinalitäten in anderer Notation an:



(aus Balzert 1)

2) Was bedeuten die folgenden Elemente:

- Rechteck: Entity / Datenobjekt
- Raute: Relationship / Beziehung
- Dreieck: Generalisierung
- Kante: Verbindungen
- Kreis: Attribut / Eigenschaft
- Entity-Schlüssel: Identifizierende Attributmenge

Zusammenfassung: Systemanalyse

Verschiedene Analysemethoden unterscheiden sich in den Sichten:

- Funktional-hierarchische Sicht
- Datenflussorientierte Sicht
- Algorithmische Sicht
- Ereignisorientierte Sicht
- Zustandsorientierte Sicht
- Regelbasierte Sicht
- Daten-Beziehungs-Sicht
- Objektorientierte Sicht
- Geschäftsprozessorientierte Sicht
- ...

2 grobe Sichteinteilungen:

- Prozessorientierte Sicht
- Datenorientierte Sicht

Zusammenfassung: Systemanalyse

Wesentliche prozessorientierte Sichten enthalten in:

- Strukturierte Analyse
- Geschäftsprozessmodellierung (z.B. ARIS)

Wesentliche datenorientierte Sichten enthalten in:

- Entity-Relationship-Modellierung
- Objektorientierte Modellierung

Systemanalyse: Datenorientierte Sicht

Rein datenorientierte Sicht: ERM

Datenorientierte Sicht mit funktionalen Aspekten: Objektorientierte Sicht (UML)

Konzeptionelle Gemeinsamkeiten:

- ***Entitäten*** **werden zu gleichartigen** ***Entitätstypen*** **zusamengefasst.**
- ***Instanzen*** **werden zu gleichartigen** ***Klassen*** **zusamengefasst.**

- ***Entitäten*** **werden durch** ***Attribute*** **näher beschrieben.**
- ***Instanzen*** **werden durch** ***Attribute*** **näher beschrieben.**

- ***Entitätstypen*** **sind durch** ***Assoziationen*** **miteinander verbunden, welche die Beziehungen der zugehörigen Entitäten beschreiben. Diese Assoziationen können von den Entitätstypen aus verschieden bezeichnet werden. Außerdem kann die Anzahl der beteiligten Instanzen eines Entitätstyps festgelegt werden.**
- ***Klassen*** **sind durch** ***Assoziationen*** **miteinander verbunden, welche die Beziehungen der zugehörigen Instanzen beschreiben. Diese Assoziationen werden von den Instanzen aus mit** ***Rollennamen*** **bezeichnet. Außerdem kann die Anzahl der beteiligten Instanzen eines Entitätstyps festgelegt werden.**

Systemanalyse: Datenorientierte Sicht

Rein datenorientierte Sicht: ERM

Datenorientierte Sicht mit funktionalen Aspekten: Objektorientierte Sicht (UML)

Konzeptionelle Unterschiede:

- ***Entitäten*** **werden durch** ***Schlüsselattribute*** **eindeutig bestimmt.**
- ***Instanzen*** **werden durch ihren** ***Namen*** **eindeutig bestimmt.**

- **Zugriff auf die Attribute nicht versteckt. Keine entitätsspezifischen Prozesse**
- **Methoden regeln den Zugriff auf die Instanzen sowie Operationen auf ihnen. Außerdem werden durch Methoden Operationen mit den Instanzen beschreiben. Methoden werden in den zugehörigen Klassen deklariert.**

- **Das Konzept der hierarchischen Ordnung von Entitätstypen wurde nachträglich in die ER-Modellierung aufgenommen (is-a). Die Vererbung bezieht sich auf die Attribute.**
- ***Klassen*** **können hierarchisch geordnet werden (Generalisierung und Spezialisierung). Die Instanzen einer spezialisierten Klasse erben alle Attribute und Methoden der generalisierten Klasse, es sei denn, ihre eigene Klasse hat selbst ein Attribut / eine Methode desselben Namens.**

Vertiefung: Objektorientierte Sicht

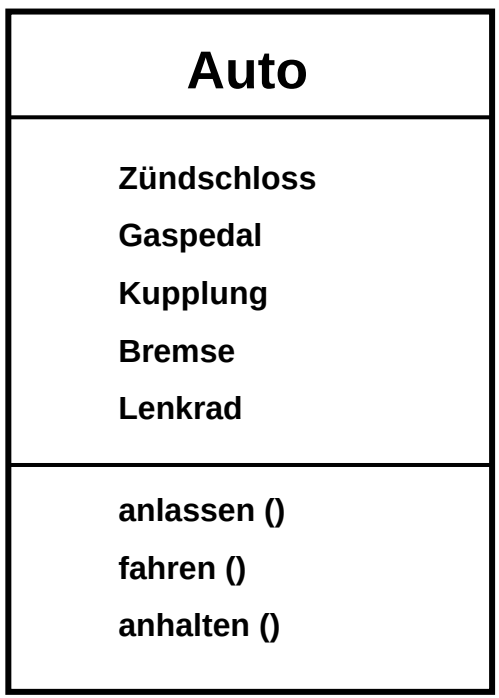
Methoden

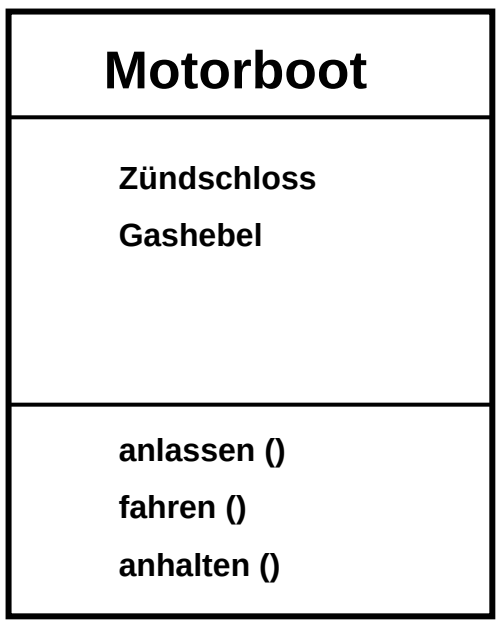
- **sind Funktionen, die auf eine Instanz der Klasse angewendet werden dürfen, in der sie beschrieben sind.**
- **„Anwendung“ heißt Ausführung der Funktion, d.h. des in der Funktion beschriebenen Algorithmus**
- **Methoden haben implizit immer die Instanz, auf die sie angewendet werden, als Parameter und optional noch weitere Parameter (die in der Beschreibung der Methode explizit festgelegt sein müssen)**

Bsp. für Methoden

Attribute

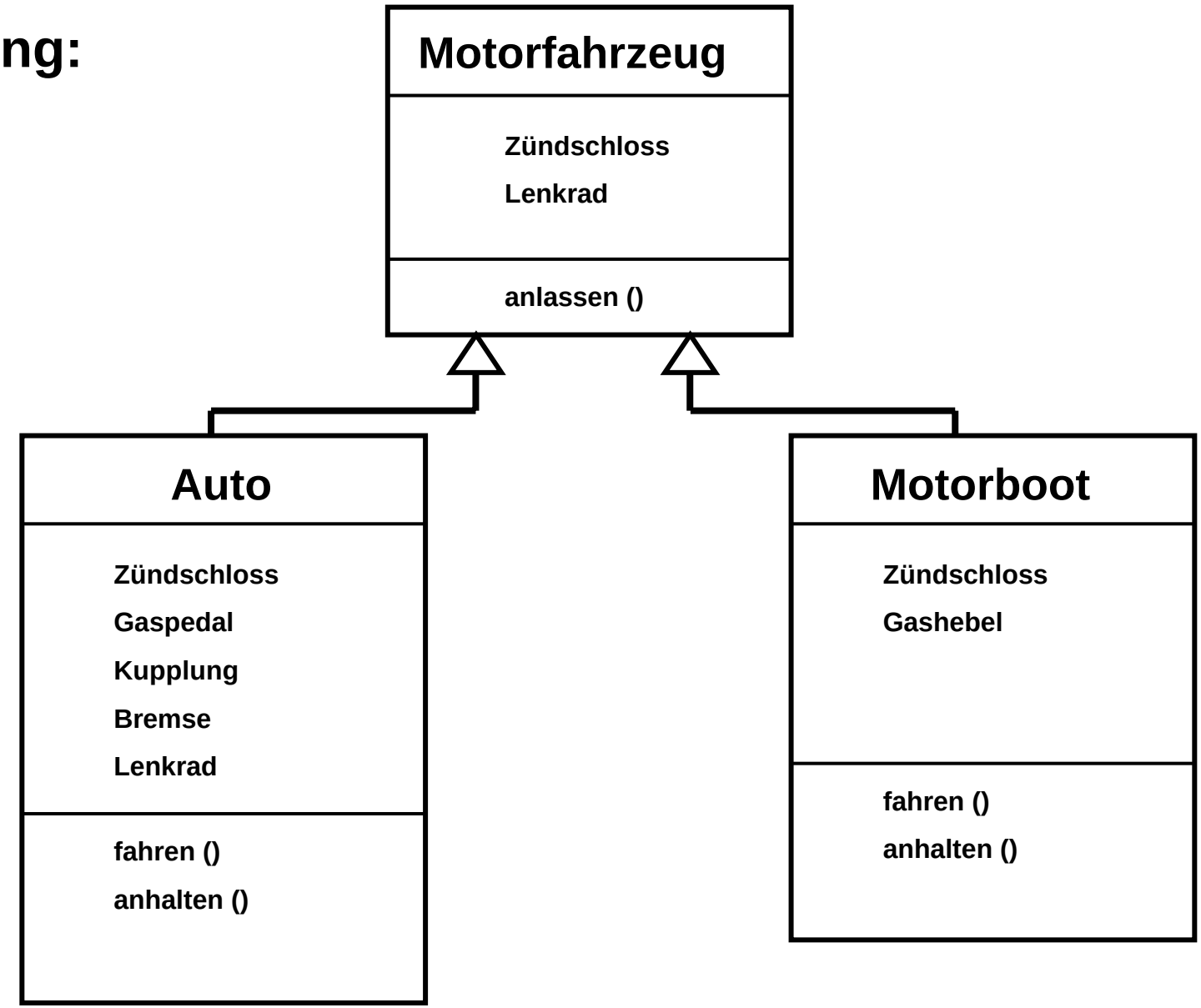
Methoden





Bsp. für Methoden

Vererbung:

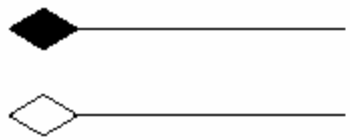


Attribute

Methoden

UML-Notation für Assoziationen

Was bedeuten diese beiden Pfeile ?

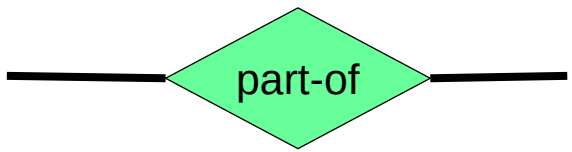


Komposition
Aggregation

Aggregation:

- Zerlegung eines Objekts in seine Bestandteile

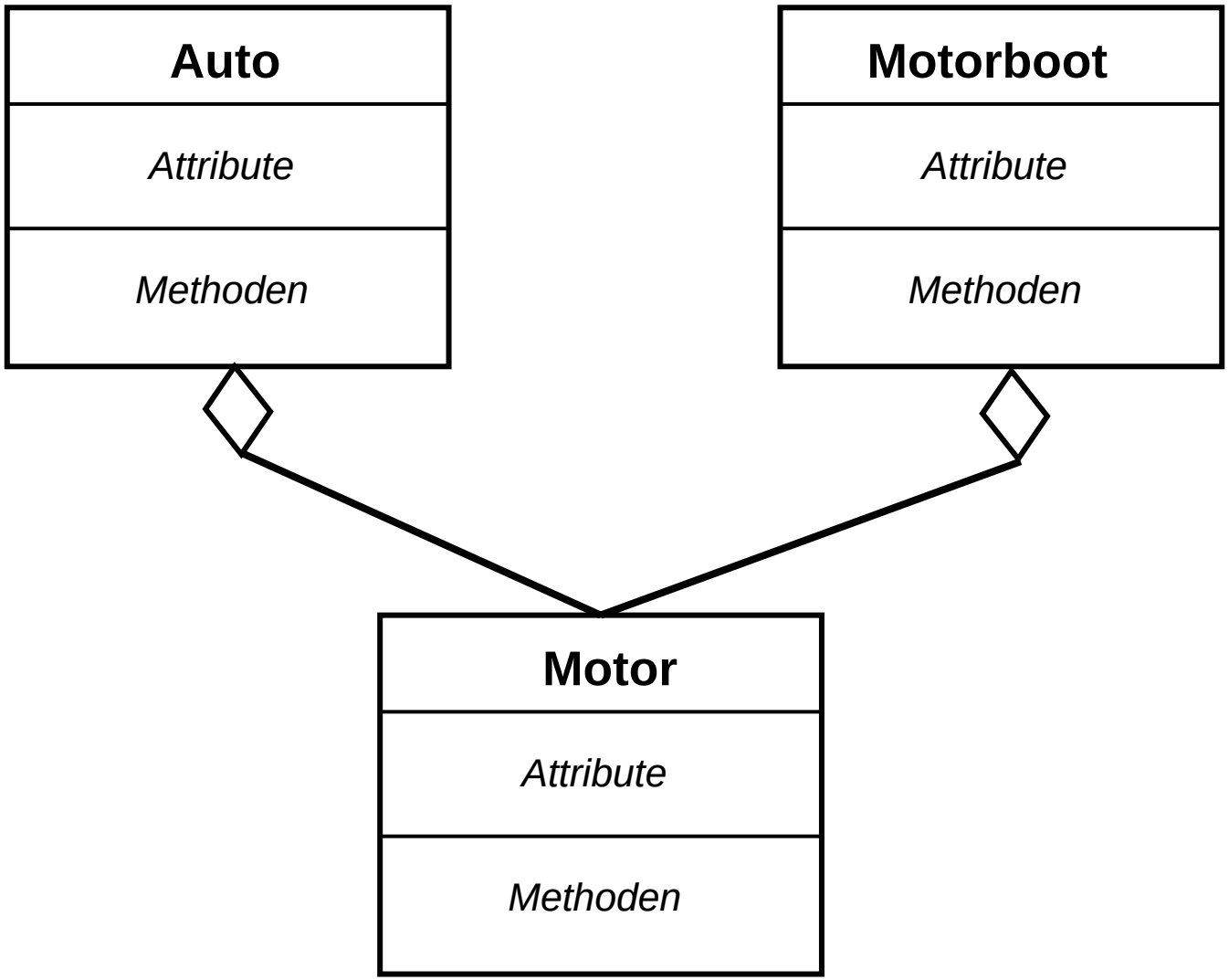
Aggregationen in ERM-Notation:



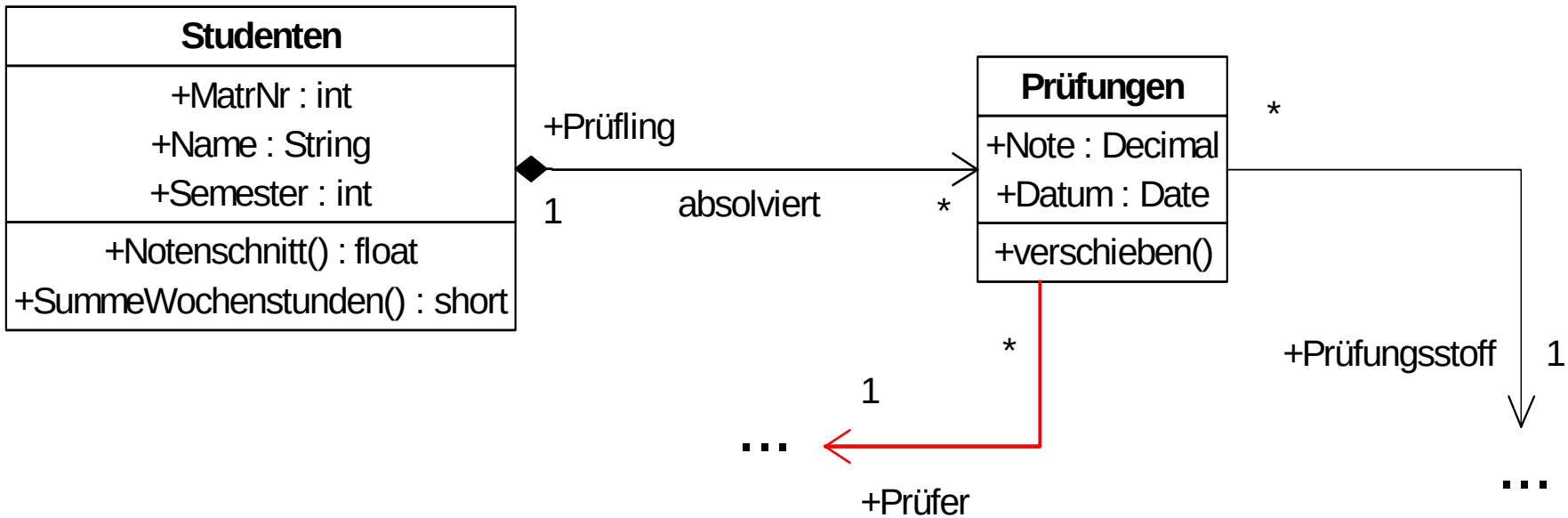
Komposition:

- Zerlegung eines Objekts in Bestandteile, die selbständig gar nicht existieren können

Bsp. für Aggregation



Bsp. für Komposition



Die Klasse „Assoziation“ in UML-Notation

Kann man in diesem Diagramm noch die Generalisierung unterbringen ?

Software-Engineering

Vorlesungsthemen:

1. Überblick über das Thema und die Vorlesung
2. Grundlegende Prinzipien
3. Softwareplanung
4. Systemanalyse
5. Softwareentwurf
6. **CASE-Tools**
7. Aufwandsabschätzung
8. Qualitätsmanagement
9. Projektmanagement

Systemanalyse vs. Softwareentwurf

Systemanalyse

- **beschreibt das System der Anwendung, für das eine Aufgabe gelöst werden soll**
- **Modellierung orientiert sich an der Realität**

Softwareentwurf

- **beschreibt den Aufbau der Software, mit der die Aufgabe gelöst werden soll**
- **Modellierung orientiert sich an den technischen Möglichkeiten**

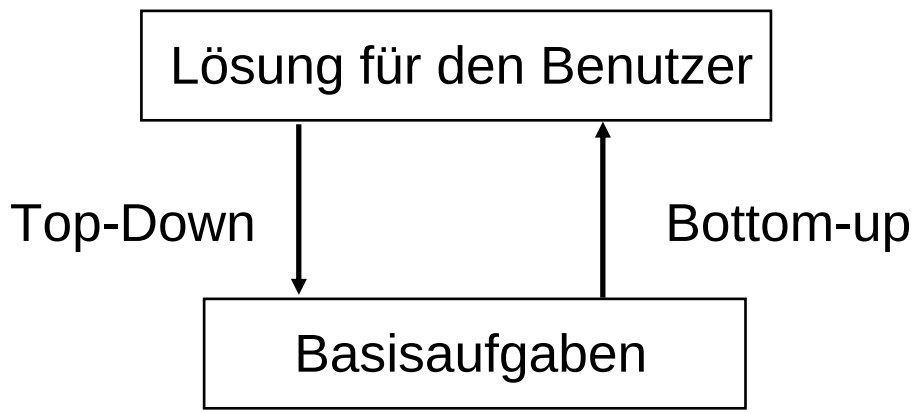
Die Modellierungshilfsmittel sind dieselben:

- **Abstraktion, verschiedene Sichten, hierarchische und modulare Zerlegung**
- **dieselben formalen Hilfsmittel: ERM, UML, ...**

Softwareentwurf: Allgemeine Prinzipien

Entwurfsrichtung

1) **Top-Down**
   - **vom Allgemeinen zum Konkreten**
2) **Bottom-Up**
   - **Vom Konkreten zum Allgemeinen**



Softwareentwurf: Top-Down

1. **Ausgangspunkt:**
   Abstrakte Zielvorgabe für das System als vollständige Beschreibung
2. **Jeder Entwicklungsschritt:**
   Zerlegung in Teilsysteme und Konkretisierung

Herausforderung:

Gewährleistung der Konsistenz zwischen den Abstraktionsstufen

Softwareentwurf: Top-Down

Vorteile

- Lösung steht im Mittelpunkt der Systementwicklung.
- Vollständige, exakte Spezifikationen führen genau zum gewünschten Ergebnis.
- Die Gesamtübersicht erleichtert die Bildung geeigneter Abstraktionsebenen und führt zu einer guten Systemstruktur.

Nachteile

- Gute Fähigkeiten zum abstrakten Denken erforderlich.
- Realisierungsprobleme werden erst spät erkannt.
- Die Wiederverwendung vorhandener Teillösungen wird erschwert.

Softwareentwurf: Bottom-Up

1. **Ausgangspunkt:**
   Konkrete Module für die Basislösung mit bekannter Funktionalität
2. **Jeder Entwicklungsschritt:**
   Zusammensetzung bekannter Module zu größeren Einheiten (Modulen)

Softwareentwurf: Bottom-Up

Vorteile:

- Realistischer Lösungsansatz durch sicheren Ausgangspunkt
- Suche nach vorhandenen Teillösungen wird begünstigt.

Nachteile:

- Schwierigkeiten, das Ziel im Auge zu behalten (und zu erreichen)
- Gefahr der Entwicklung von später nicht benötigten Softwarekomponenten auf unteren Ebenen des SW-Systems

Softwareentwurf: „Jo-Jo-Methode“

1. **Ausgangspunkt:**
   Zielvorgabe und Bestandsaufnahme der möglichen Teillösungen
2. **Jeder Entwicklungsschritt:**
   Zerlegung eines abstrakten (von oben entwickelten) Moduls in kleinere oder Zusammensetzung konkreter (von unten entwickelten) Modulen zu größerer Einheit

Softwareentwurf: „Jo-Jo-Methode“

Vorteile

- Bestmöglicher Kompromiss zwischen Wünschenswertem und Machbarem
- Erstellen einer möglichst ökonomischen Lösung
- Anforderungen an das Abstraktionsvermögen reduziert

Nachteile

- Wechsel der Bearbeitungsrichtung willkürlich, daher Gefahr mangelnder Systematik

Beim nächsten Mal:
Systementwurf (Fortsetzung)