Teil II
Objektorientierte Programmierung (OOP)
20. Objektnetze

Prof. Dr. rer. nat. Uwe Aßmann
Institut für Software- und Multimediatechnik
Lehrstuhl Softwaretechnologie
Fakultät für Informatik
TU Dresden

Version 13-1.0, 28.04.12

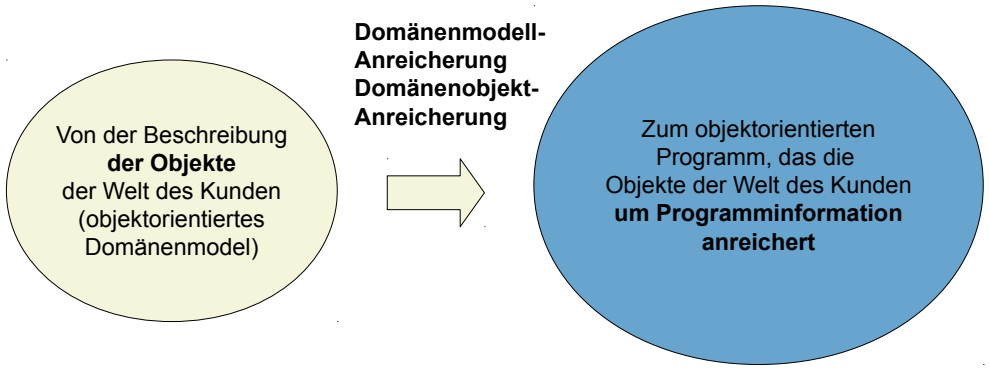
- 21) Verfeinern von Assoziationen mit dem Java-2 Collection Framework
- 24) Graphen in Java
- 25) Entwurfsmuster
- 26) Benutzerschnittstellen

Obligatorische Literatur

- JDK Tutorial für J2SE oder J2EE, www.java.sun.com

Die zentralen Fragen des objektorientierten Ansatzes

Wie kommen wir vom Problem des Kunden zum Programm (oder Produkt)?



Anreicherung/Verfettung: Anreicherung durch technische Programminformation
„object fattening“: Anreicherung von Objekten des Domänenmodells

Ziel von Teil II der Vorlesung

- Aufbau von Objektnetzen (Graphen, Dags, Bäumen, Listen) durch Datenverfeinerung von Assoziationen
  - Graphen, Iteratormethoden, Iteratoren, und Streams
  - Große Objekte (Bobs) mit internen Netzen
    - Endo- und Exoassoziationen
  - Wie man Graphen erweitert
- Verfeinerung

Grundlegende Begriffe

- Ein **Sprachkonstrukt (Sprachelement)** bezeichnet ein Konstrukt bzw. Konzept einer Sprache.
- Ein **Programm-/Modellelement** bezeichnet ein Element eines Programms/Modells
- Ein **Fragment** eines Programms oder Modells ist ein partieller Satz der Sprache.
- Ein **generisches Fragment (Fragmentformular)** eines Programms oder Modells ist ein partieller Satz der Sprache mit Platzhaltern (“Lücken”)
- Eine **Fragmentgruppe** ist eine Menge von Fragmenten und generischen Fragmenten
- Eine **Fragmentkomponente** ist eine Fragmentgruppe zur Wiederverwendung

Verfeinerung: Schritte von UML zur Implementierung

- **Verfeinerung von Sprachkonstrukten (Realisierung, Abflachen, lowering):** Viele der Sprachkonstrukte aus UML haben kein direktes Java-Äquivalent und müssen zu Java-Konstrukten *verfeinert* werden.

Ein ***Implementierungsmuster*** (*workaround, Idiom)* beschreibt die Verfeinerung eines Sprachkonstruktes einer Modellierungs- oder Spezifikationssprache durch ein Fragment einer Implementierungssprache

- Einziehen von Schnittstellen zur Sicherstellung von homogenem Verhalten (schnittstellenorientiertes Programmieren)
- Persistenz mit Objektrelationaler Abbildung (OR-Mapping)
- Netzentwurf:
  - Verfeinerung von Assoziationen zu Collections
  - Verfeinern von Assoziationen zu getypten Collections (mit Generics)
- Implementierung von Methoden (von Statecharts und Aktivitätsdiagrammen)

Was bedeutet Verfeinerung?

- Verfeinerungsschritte vom Analysemodell zum Entwurfsmodell
  - **Syntaktische Verfeinerung** ersetzt nur die Syntax
    - **Datenverfeinerung** verfeinert Datenstrukturen
    - **Kontrollverfeinerung** verfeinert Kontrollstrukturen (Verhalten)
  - **Semantische Verfeinerung** erhält die Semantik des Modells oder Programms
    - Dazu ist ein mathematischer Beweis nötig

Verfeinerungsoperationen

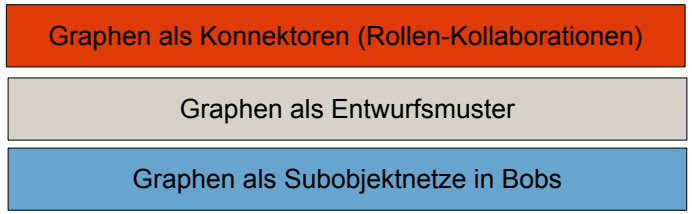
- Horizontale Operationen:
  - **Abstraktion:** Vernachlässigen von Details
  - **Detaillierung (Anreicherung)**: Ergänzung von (optionalen) Einzelheiten
  - **Vervollständigung** (**Elaboration**) von Fragmenten zu Sätzen der Modellierungssprache
  - **Erhöhung Zuverlässigkeit**: Ergänzung von qualitätssteigernden Fragmenten (Typisierung, Verträge)
  - Einführung des **Architektur-Aspektes** des Systems
  - **Strukturierung** und **Restrukturierung**
  - **Refaktorisierung (Refactoring)** ist semantische Restrukturierung
- Vertikale Operationen:
  - **Abflachen von Fragmenten** (Flachklopfen, Realisierung, lowering):
  - **Realisierung** ersetzt ausdrucksstarke Konstrukte durch weniger ausdrucksstarke, implementierungsnähere

Repräsentation von Objektnetzen als Datenstrukturen (Datenverfeinerung)

- Objektnetze (nicht-fixe Assoziationen) werden repräsentiert
  - durch Skriptsprachen, die Graphen als **Sprachkonstrukte** eingebaut haben
    - Rapid Application Development (RAD)
    - Falls es auf die Performanz nicht so ankommt
  - durch **Graphen** aus Java-Graph-Bibliotheken
  - durch **Collections**, nach dem Abflachen/Flachklopfen von bidirektionalen Assoziationen in gerichteten Links
  - durch **Datenstrukturen** fester Länge (Arrays, Matrizen) (speicher-bewusstes Programmieren)

Verfeinerung von fixen Assoziationen als Konnektoren (Kollaborationen)

- Fixe n:m-Assoziationen mit festem n, m (z.B. 1:1-Assoziationen) können durch **Konnektoren** (Kollaborationen, Teams) verfeinert werden
  - Assoziationen tragen Rollentypen als Assoziationsenden
- Einfache Objektnetze werden durch **Entwurfsmuster** beschrieben
  - Listen: Decorator, Chain
  - Bäume: Composite
  - Dags, Graphen: Modifiziertes Composite
- Große Objekte (Bobs) haben oft interne Subobjektnetze
  - Subobjektnetze beruhen auf Endo-Assoziationen



Softwareentwicklung im V-Modell [Boehm 1979]