Entwurfsstrategien

Funktionaler Entwurf:

System = Menge interagierender Fkt.

Systemzustand: global.

Was ist die benötigte Funktionalität?

Wie läßt sie sich verfeinern?

$\leadsto$ Datenstrukturen (Objekte?)

Objektorientierter Entwurf:

System = Menge interagierender Objekte.

Systemzustand: dezentral.

Was sind die beteiligten Objekte?

Wie sehen Beziehungen dazwischen aus?

Welches Objekt ist wofür verantwortlich?

$\leadsto$ Funktionalität

OO-Entwurf (I)

Modellierung:

Modell = Vereinfachung der Realität.

Verzicht auf *Unwichtiges*.

Modell $\leadsto$ besseres Verständnis
(da Komplexität reduziert).

Mehrere Perspektiven notwendig:

Klassenhierarchie

Objektstruktur

Paketstruktur

Prozeßstruktur

dynamische Struktur

...

OO-Entwurf (II)

Konstrukte einer typischen objektorientierten Programmiersprache:

Objekte

Klassen

Methoden

Pakete

Wie bilden wir die Realität sinnvoll auf diese Konstrukte ab?

Objekte (I)

Objekte repräsentieren individuelle, identifizierbare (reale oder abstrakte) Dinge.

Objekte haben eine wohldefinierte Rolle in dem Problembereich.

Objekte haben Zustand, Verhalten, Identität.

Objekte sind möglicherweise aktiv.

Objekte sind möglicherweise persistent.

Zum Beispiel:

Objekte der realen Welt,

aber auch: Objekte, die erst im Entwurfsprozeß entstehen und in Zusammenarbeit mit anderen eine bestimmte Funktionalität zur Verfügung stellen.

Objekte (II)

Konkrete Beispiele:

- Dinge der realen Welt
- Personen (oft: Person + Rolle)
- Organisationen
- Orte
- Ereignisse (Ereignisse als Schritt einer geordneten Folge; Ereignisse zu einem bestimmten Zeitpunkt)
- Konzepte

Beziehungen zwischen Objekten

Verbindungen:

- über die ein Objekt auf ein anderes zugreifen kann,
- entlang denen Nachrichten verschickt werden können (in einer oder in beiden Richtungen)
- Kardinalität: eins/eins, eins/viele, viele/viele.

Aggregationen:

- „hat-als-ein-Teil"-Beziehungen („has-a").

Klassen

Klasse = Menge von Objekten mit gleicher Struktur und gleichem Verhalten.

Beziehungen zwischen Klassen;

- wie zwischen Objekten,
- außerdem:
  - Vererbung: „ist-ein"-Beziehungen („is-a")

Vererbungshierarchien

Vererbungshierarchien sollen natürliches Abbild der modellierten Struktur sein.

Sorgfältig planen:

- nachträgliche Änderungen an komplexen Hierarchien können komplette Neustrukturierung nach sich ziehen.

Vererbungshierarchien müssen gelegentlich überarbeitet werden.

is-a und has-a

Manchmal ist nicht offensichtlich, ob eine „is-a“-Beziehung oder eine „has-a“-Beziehung vorliegt.

Beispiel: Listen/Mengen.

Was ist ein Objekt?

Einfache Methode:

Problem in natürlicher Sprache ausdrücken.

Substantive $\rightsquigarrow$ Objekte

Verben $\rightsquigarrow$ Operationen (Methoden)

CRC-Karten

Karteikarten mit

Component:
Klassenname.

Responsibility:
Aufgabe der Klasse.

Collaborators:
Komponenten, mit denen Objekte der Klasse kommunizieren.