Einführung in die Informatik

Modelle und Schichten

Meik Teßmer

Inhalte

Lernziele

- Erarbeitung eines geeigneten Modellbegriffs
- *Schichten* als grundlegendes Organisationskonzept
- Beispiele

Einstiegsfrage

Wie könnte man eine Kommunikationsplattform (Signal, WhatsApp) aufbauen?

Was wäre eine sinnvolle Herangehensweise?

Modellbegriff

Modell

Was ist ein Modell?

Modell

Abbildung 1: Modell? (Quelle: Wikipedia)

Modell

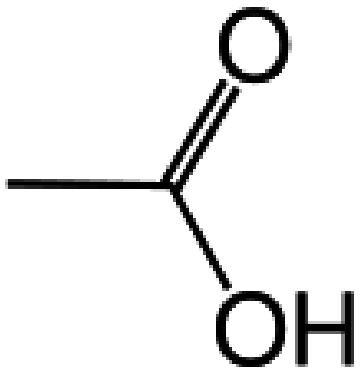

Abbildung 2: Auch Modell? (Quelle: Wikipedia)

Modell

Abbildung 3: Und das? (Quelle: Wikipedia)

Modell

Hauptgebäude
Bauteil
H
Bauteil
Q
S
T
U
V
W
R
RS
Hörsaal 16
Hörsaal 15
SKT
Toiletten
Infopunkt
Hörsaal 14
13
TLU
Hörsaal 12
11
Lebens-mittel
Hörsaal 10
9
8
Sparkasse
Schreib-waren
UMW
Restau-rant
Buch-laden
Hörsaal 7
VNW
Westend
Sporthalle
J
Auditorium Maximum
Audimin
K
L
Halle
M
N
Schwimmbad
P
Café
AKB
BLC
Hörsaal 1
Bäcker
Post
Hörsaal 2
3
CMD
Hörsaal 4
Vers.
Jobs
Hörsaal 5
6
DNE
EF
A
B
C
D
E
F
AA
Mensa
BB
CC
DD
EE
FF
Haupteingang
Stadtbahn
Parken
G

Modell

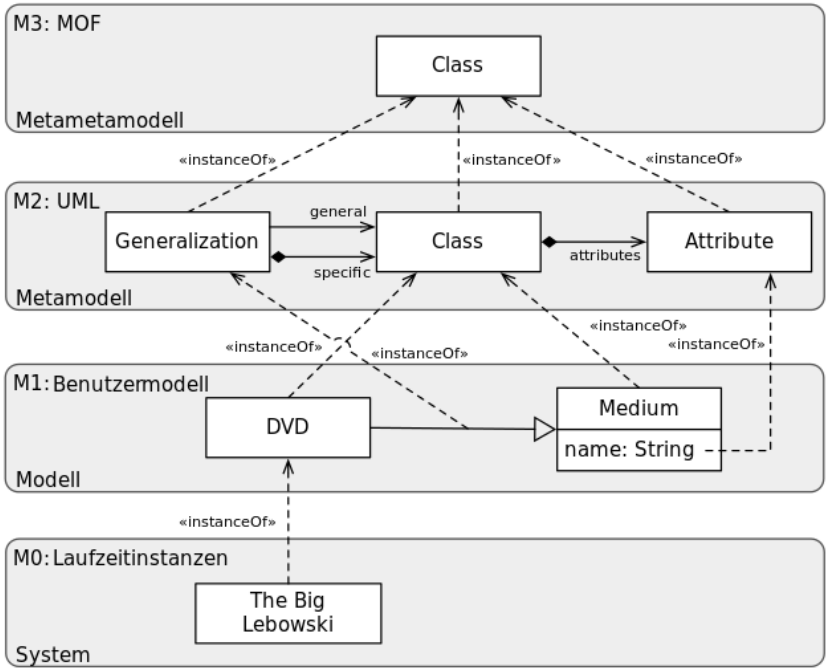

Abbildung 5: Und hier? (Quelle: Wikipedia)

Modell: Arbeitsdefinition

eine mögliche Definition:

Ein Modell ist ein Abbild eines Ausschnitts der Realität.

Sprachwelten

Beispiele von formalisierten Modellen

Beispiel: Mathematik

- Formeln:
  - eigene Zeichen
  - eigene Regeln
  - Fokus: Berechenbarkeit, Logik

Beispiel: Softwareentwicklung

- Unified Modeling Language (UML[1])
  - verschiedene Diagrammtypen
  - eigene Symbole und Verknüpfungsregeln
  - Fokus: Erfassen von Anforderungen, Übersicht über die Systemstruktur usw.

[1] zu UML siehe auch Rupp, Queins, and SOPHISTen (2012)

Beispiel

unser erstes eigenes Modell: unser Computer

Abbildung 6: Computer

Modell-Variante 1

Ein Computer ist ein Tupel C:

$C = (B, G, OS, A, D)$

mit

- B: Menge der Benutzer
- G: Gerät
- OS: Betriebssystem
- A: Menge der Anwendungsprogramme
- D: Menge von Daten

Modell-Variante 1

ein Laptop:

$C_1 = (\text{Peter Schmidt}, \text{Dell Latitude E6550}, \text{GNU/Linux}, \cdots, \cdots)$

Modell-Variante 2

„bessere“ Aufgabenstellung:

Wie werden Probleme mit Hilfe eines Computers gelöst?[2]

Im Detail:

- Welche Bestandteile unseres Rechners tragen zur Problemlösung bei?
- Wie hängen sie dabei voneinander ab?
- Darstellung: tabellarisch

[2] zu Problemlösen siehe auch Pólya (1957)

Modell-Variante 2

Schicht	Beispiel
Benutzer	wir
Anwendung	Firefox ...
...	...
Betriebssystem	GNU/Linux ...
...	...
Gerät	Dell ....

→ Schichtenmodell

Entwicklung

Arbeitsdefinition

bisher genutzte Definition:

Ein Modell ist ein Abbild eines Ausschnitts der Realität.

Wie kommt man dorthin?

Abgrenzung

- relevant: „mein“ Rechner
- nicht relevant: andere Rechner, Wandfarbe des Arbeitszimmers, Stuhlfabrikat, Geschwindigkeit der Internetanbindung

Reduktion

- relevant: Anwendungsprogramme, Betriebssystem ...
- nicht relevant: Farbe des Gehäuses, Geschwindigkeit der Festplatte, Anzahl USB-Anschlüsse

Dekomposition

- Reduktion der Komplexität
- Prinzip Teile-und-Herrsche
- kann hierarchisch strukturiert sein
- vergleichbar einem top-down-Vorgehen

wesentliche Schwierigkeit: geschickte Zerlegung finden

Aggregation

- detailreiche Teilsysteme zu größeren Teilsystemen kombinieren
- Gegenstück zur Dekomposition
- vergleichbar einem bottom-up-Vorgehen

Abstraktion

- Klassen-/Gruppenbildung: Einteilung von Objekten in homogene Mengen
- Eigenschaften: Objekte haben Eigenschaften
- Spezialisierung/Generalisierung

Zusammenfassung

- Abgrenzung: nicht relevante Objekte weglassen
- Reduktion: nicht relevante Details weglassen
- Dekomposition: Zerlegen in Teilsysteme
- Aggregation: Kombinieren von Teilsystemen
- Abstraktion: Klassen-/Gruppenbildung

Mögliche Risiken bei der Modellentwicklung

- Gleichsetzen von Modell und Realität

  konkretes Beispiel: Wirtschaftskrisen der letzten Jahre, Pandemien

  $\rightarrow$ BWL-/VWL-Modelle funktionieren offensichtlich nicht zuverlässig

  $\rightarrow$ Ausbreitung von Krankheiten sind schwer zu modellieren
- Rückübertragung von Ergebnissen/Optimierungen
- schlechtes Modell

Modelle

Zurück zur Einstiegsfrage

> *Wie könnte man eine Kommunikationsplattform entwickeln?*

Ideen?

Strukturierung unserer Plattform

- Zuständigkeiten: Separation of Concerns
- Hauptbestandteile:
  - Schnittstelle(n) zum Anwender (Oberfläche)
  - Anwendungslogik
  - Datenhaltung
- Regeln:
  - Kommunikation nur von oben nach unten bzw. umgekehrt
  - keine Umgehung von Schichten

Strukturierung unserer Plattform

unsere Plattform:

Schicht
Darstellung
Anwendungslogik
Datenhaltung

Charakteristika einer Schicht

- aktiv in nur einer Richtung (nach unten)
- keine Brüche
- Unabhängigkeit, solange Schnittstellen gewahrt bleiben (Kapselung)
- haben jeweils eine konkrete Aufgabe

Auswirkungen:

- geringe Kopplung
- Wiederverwendbarkeit
- Wartbarkeit und Systemverstehen

Probleme dieses Modells

- Performanceverlust (~10% pro Schicht)
- oftmals nur simples Durchreichen von Daten (Reports)

Beispiel für eine Zwei-Schicht-Architektur

Tabelle 4: Ein Office-Paket

Schicht	Ausprägung
Anwendungsschicht	Office
Datenschicht	Dateisystem

Beispiel für eine Drei-Schicht-Architektur

Tabelle 5: eKVV

Schicht	Ausprägung
Präsentationsschicht	Webbrowser
Logikschicht	Appl.-Server
Datenschicht	Datenbank

Beispiel für „übertriebene“ Schichtung: OSI

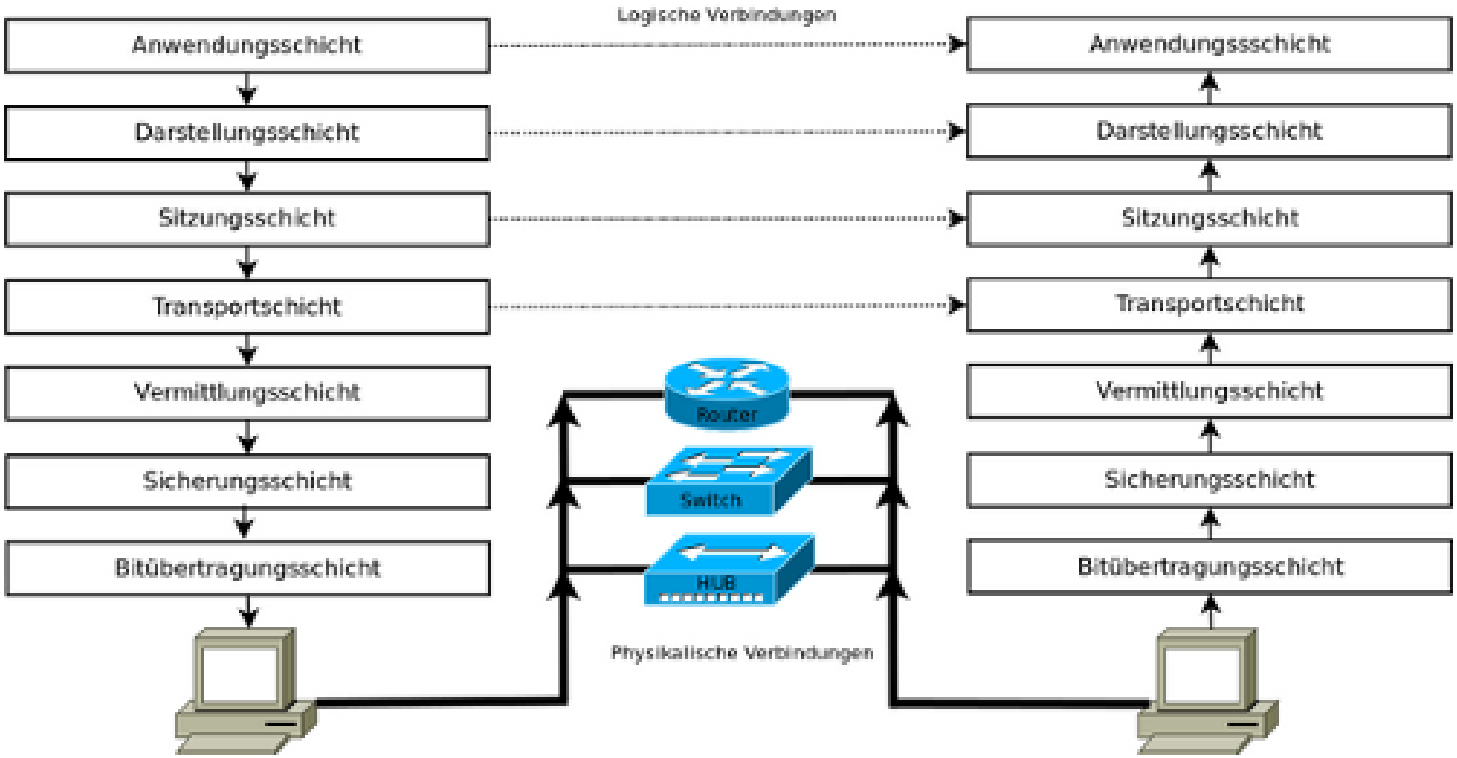

Abbildung 7: OSI-Referenzmodell

Anwendung der OSI-Schichten: TCP/IP

OSI-Schicht	TCP/IP-Schicht	Beispiel
Anwendungen (7)	Anwendungen	HTTP, UDS, FTP, SMTP, POP, Telnet, OPC UA
Darstellung (6)		
Sitzung (5)		
		SOCKS
Transport (4)	Transport	TCP, UDP, SCTP
Vermittlung (3)	Internet	IP (IPv4, IPv6)
Sicherung (2)	Netzzugang	Ethernet, Token Bus, Token Ring, FDDI, IPoAC
Bitübertragung (1)		

Abbildung 8: TCP/IP-Modell

Konzeptionelle Schichten in Programmiersprachen Python

- ganz unten: Anweisungen, Bedingungen, Schleifen
- darüber: Funktionen, die Anweisungen zusammenfassen
- darüber: Funktionen, die Funktionen aufrufen usw.

Code der Datenhaltungsschicht:

```
def lade_daten():
  ...

def speichere_daten(neue_daten):
  ...
```

→ Schnittstelle: `lade_daten` und `speichere_daten`

Schichten in Python

konkreter Code:

```
def lade_daten():
  handle = open("meine_daten.txt", "r")
  daten = []
  for zeile in handle:
    daten.append(zeile)
  handle.close()
  return daten

def speichere_daten(neue_daten):
  handle = open("meine_daten.txt", "w")
  for zeile in neue_daten:
    handle.write(zeile)
  handle.close()
```

Ende

Literatur:

Pólya, George. 1957. *How to Solve It: A New Aspect of Mathematical Method*. 2nd ed. Garden City, New York: Doubleday Anchor Books, Doubleday & Company, Inc.

Rupp, Chris, Stefan Queins, and die SOPHISTen. 2012. *UML 2 Glaskar: Praxiswissen Für Die UML-Modellierung*. 4. ed. München Wien: Carl Hanser Verlag.