18 Architekturstile und -metaphern

18.1 Architekturstile

Architekturstil (architectural style) – Leitlinien für die Gestaltung der Architektur

- ☆ Verwendete Modularten
- ☆ Verwendete Arten von Kooperation zwischen den Modulen
- ☆ Benutzung passender Architekturmuster
- ☆ Orientierung an einer Architekturmetapher

➜ Architekturstil

Was fällt hier auf?

Modularten

- bestimmt durch die Art der Modularisierung
- zum Beispiel Funktionen, abstrakte Datentypen oder Prozesse als Einheiten der Modularisierung

Kooperationsarten

- zum Beispiel: direkter Aufruf, indirekter Aufruf, Datenfluss
- müssen zur Art der Module passen

Architekturmuster → Kapitel 20

Architekturmetapher → Gestaltungs-Leitbild, siehe 18.2 unten

18.2 Architekturmetaphern

Metapher (metaphor) – sprachlicher Ausdruck, bei dem ein Wort aus seinem Bedeutungszusammenhang in einen anderen übertragen, als Bild verwendet wird.

Architekturmetaphern

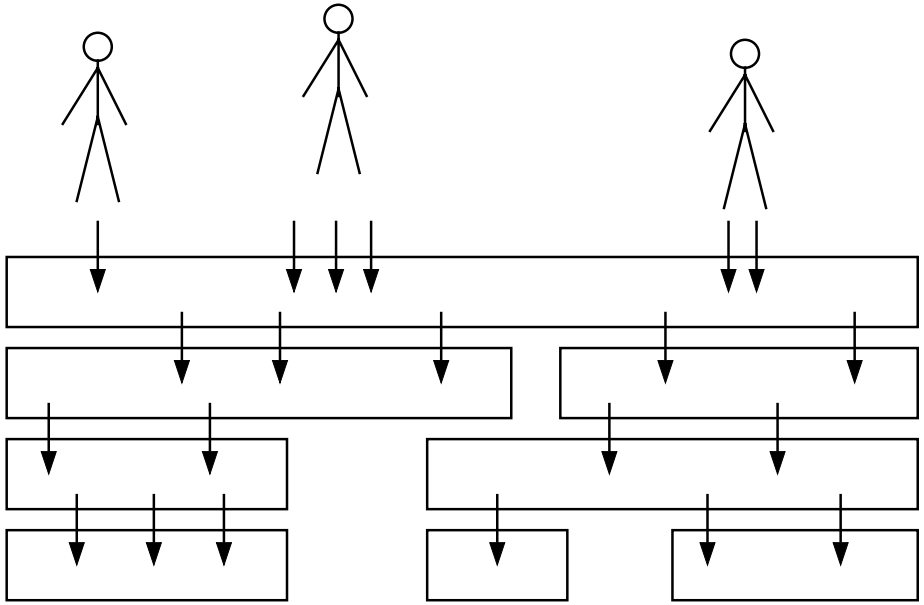
- sind Leitbilder für die Gliederung und das Verstehen einer Architektur
- erschließen das Verständnis über analoge, vertraute Bilder

Beispiele (siehe auch Vorlesung Informatik II, Systemmetaphern):

- Virtuelle Maschinen
- WAM (Werkzeug-Automat-Material)
- Steckersystem
- Organisationshierarchie

Die Virtuelle-Maschinen-Metapher

Leitgedanke: Das System besteht aus aufeinander aufbauenden Schichten realer oder künstlicher Maschinen.

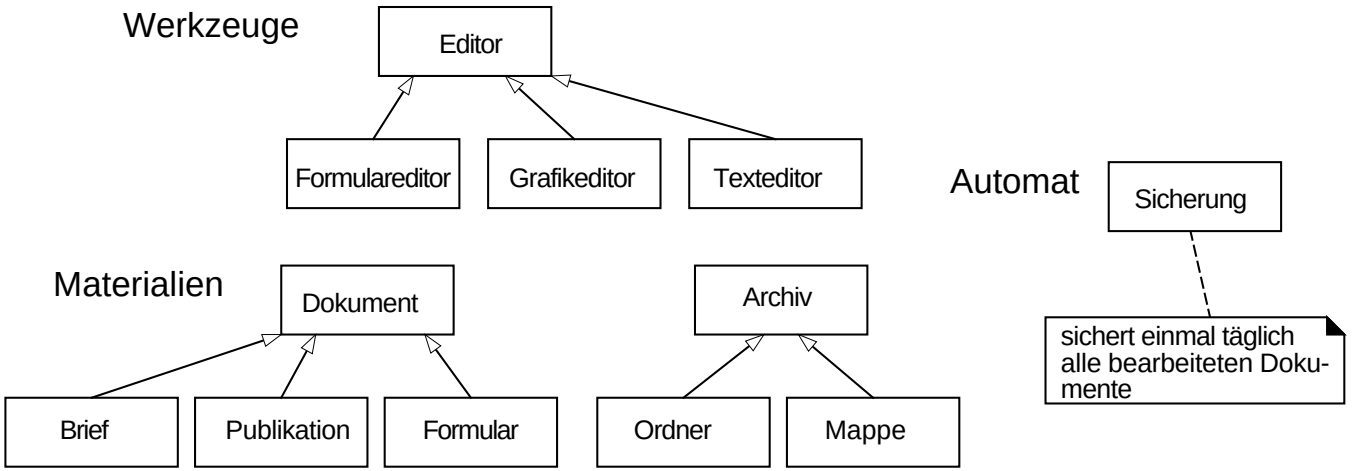


- Jede Schicht
  - besteht aus einer oder mehreren virtuellen Maschinen
  - erbringt Leistungen für die darüberliegende Schicht
  - benutzt Leistungen der darunter-liegenden Schicht
- Die unterste Schicht besteht aus realen Maschinen
- Die oberste Schicht erbringt die für Anwender zugänglichen Leistungen
- Typisches Beispiel: OSI-Referenzmodell für die Kommunikation von Rechnern

Die WAM (Werkzeug-Material-Automat) Metapher

Leitgedanke: Ein System besteht aus Materialien, deren verschiedene Aspekte durch aspektspezifische Werkzeuge bearbeitet werden.

- Werkzeug: – gegenüber Materialien aktiv: bearbeitet Materialien
  – gegenüber Menschen assistierend: Mensch bedient
- Material: – passiv, speichernd, wird bearbeitet
  – ist Arbeitsgegenstand oder Arbeitsergebnis
- Automat: aktiv, arbeitet vollautomatisch

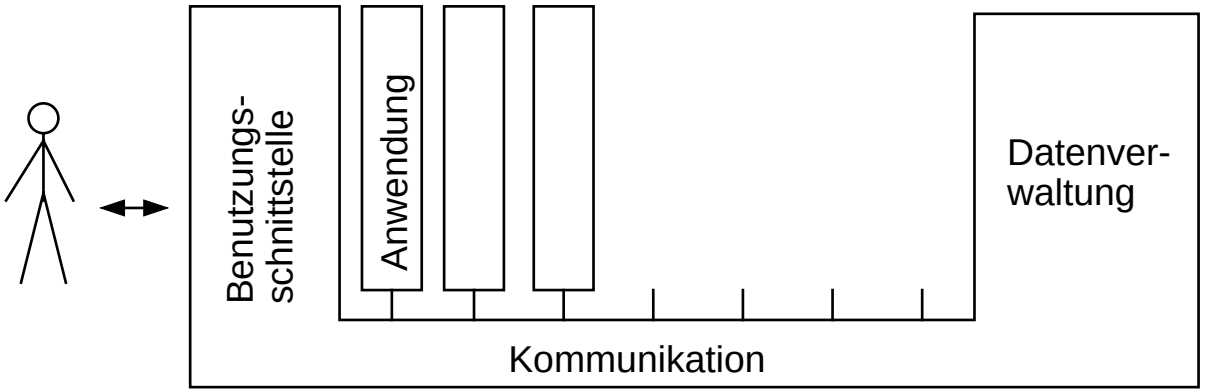


Die Steckersystem-Metapher

Leitgedanke: Komponenten werden flexibel in ein Grundsystem mit einer Reihe von Steckplätzen eingesteckt (analog zu Rechner-Hardware, Lichtschienen, etc.)

Ein System besteht typisch aus

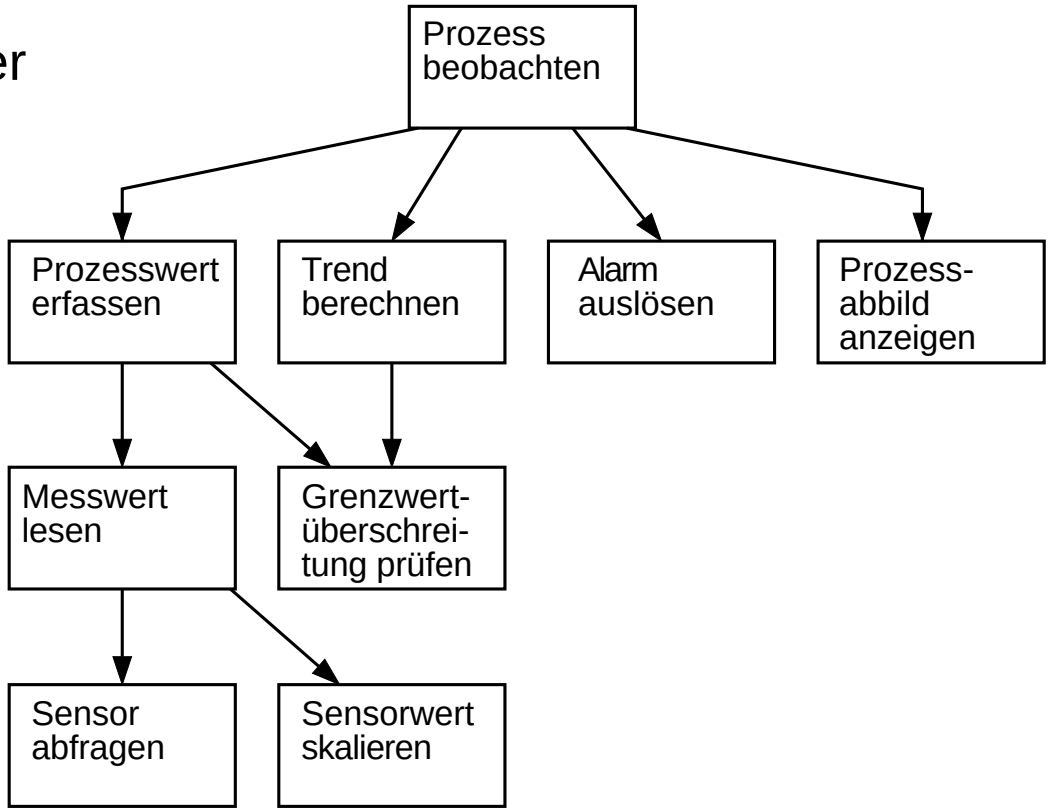
- dem Rahmen
  - Datenverwaltungsdienste, Kommunikationsdienste, Benutzerschnittstelle
  - Steckplätzen für Anwendungen
  - in der Regel vorgefertigt
- Anwendungen
  - in Steckplätze des Rahmens eingesteckt
  - nutzen Dienste und Benutzerschnittstelle des Rahmens
  - in der Regel problem- und kundenspezifisch angefertigt



18.3 Ausgewählte Architekturstile

18.3.1 Funktionsorientierte Architektur (Structured Design)

- Jeder Modul berechnet eine Funktion
- Zur Realisierung einer Funktion können andere, einfachere Funktionen aufgerufen werden
- Datenaustausch über Parameter oder gemeinsame Speicherbereiche
- Entwurf = Hierarchie aufeinander aufbauender Funktionen



- Meist nach dem Prinzip Eingabe – Verarbeitung – Ausgabe strukturiert
- Typischer Vertreter: “Structured Design”
- Problem: Daten und funktionsübergreifende Probleme können nicht gekapselt werden

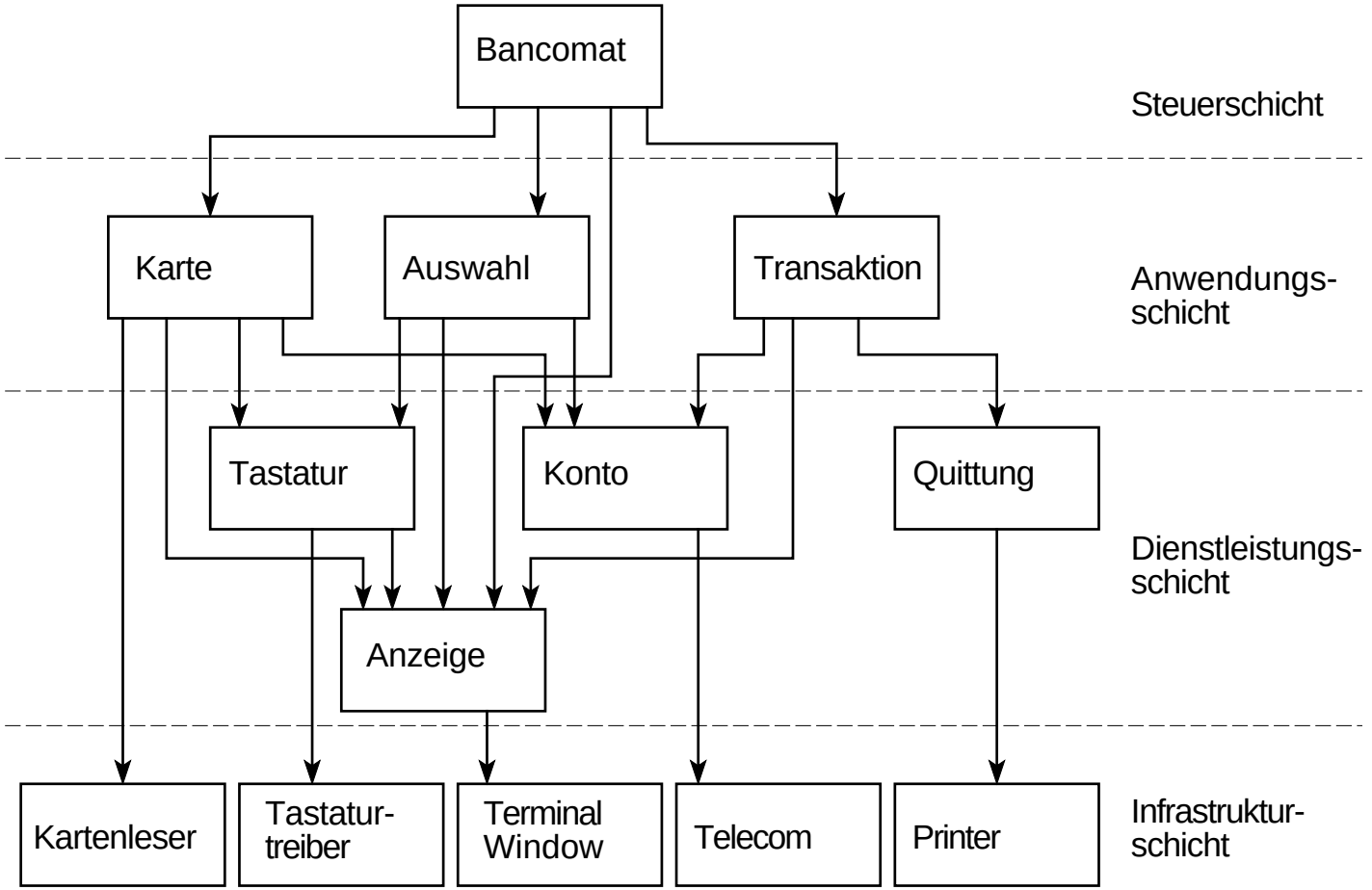
  $\rightarrow$ Ungenügende Abstraktion

  $\rightarrow$ Verständnis und Änderbarkeit erschwert
- ⇨ Liefert in vielen Fällen keine gute Modularisierung

18.3.2 Datenorientierte Architektur (Datenabstraktionen)

- Modularisierung nach dem Geheimnisprinzip
- Datenstruktur und alle darauf möglichen Operationen sind zusammengefasst: Datenabstraktion
  - notwendig zum Verbergen von Entwurfsentscheidungen
  - Realisierung durch abstrakte Datentypen
- System besteht aus Menge aufeinander aufbauender Datenabstraktionen
  - Jeder Modul bietet Leistungen an
  - Module benutzen die Leistungen anderer Module zur Realisierung der eigenen Leistungen
  - ⇨ Benutzungshierarchie
  - Regelung der Zusammenarbeit durch Verträge (Design by Contract)
- Zusätzlich häufig Gliederung in Schichten
- Gute Abstraktion → Entwürfe sind leicht verstehbar und änderbar

Beispiel: Datenorientierte Architektur eines Geldautomaten

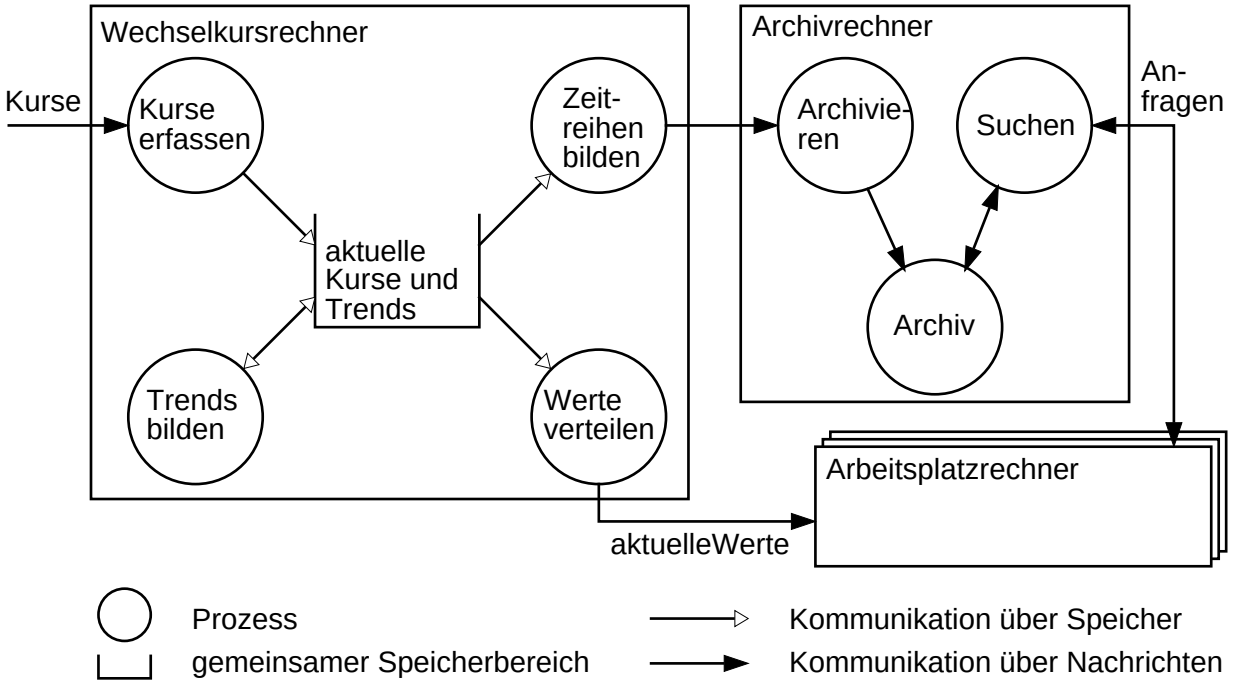


18.3.3 Objektorientierte Architektur (→ Kapitel 19)

- Modul repräsentiert Objekt des Problembereichs oder Informatik-Element
- System besteht aus Menge kooperierender Objekte, gruppiert zu Klassen
- Geeignet gebildete Klassen beachten das Geheimnisprinzip
  → gute Modularisierung
- Systematischer Zusammenhang zwischen allgemeinen und speziellen Objekten: Objekte von Spezialklassen erben alle Strukturen und Operationen der übergeordneten allgemeinen Klassen
  → Spezialisierungs- (Generalisierungs-) Hierarchie
  - ermöglicht problemnahe Modellierung
  - schränkt jedoch Anwendbarkeit des Geheimnisprinzips ein
- Durch Vererbung massiv erhöhte Flexibilität
  - **+** Software ist erweiterbar
  - **–** bei unsachgemäßer Verwendung starke Nebenwirkungen auf Verstehbarkeit und Änderbarkeit

18.3.4 Prozessorientierte Architektur (→ Kapitel 21)

- System besteht aus Menge unabhängig arbeitender, untereinander kooperierender Akteure
- Akteure sind typisch als Prozesse realisiert
- Prozesse kooperieren durch Austausch von Nachrichten oder durch Zugriff auf gemeinsame Speicherbereiche
- Prozesse sind die Module der obersten Stufe
- Jeder Prozess ist typisch ein sequentiell ablaufender Systemteil
- Prozesse sind selbst wieder modularisiert, z.B. in objekt-orientiertem Stil



18.3.5 Komponentenorientierte Architektur (→ Kapitel 23)

- Weiterentwicklung objektorientierter Architekturen

Komponente (component) [im engeren Sinn] – Stark gekapselte Menge zusammengehöriger Objekte / KIassen, die eine gemeinsame Aufgabe lösen

- System besteht aus einer Menge von (möglicherweise geographisch verteilten) Komponenten, die über einen Makler kommunizieren
- Komponenten kennen
  - Schnittstellen ihrer Partnerkomponenten
- Komponenten kennen nicht
  - Art der Realisierung der Kommunikation
  - Geografische Lokalisierung
  - Implementierung der Partnerkomponenten

18.3.6 Datenflussorientierte Architektur

- Die Software wird in Aktivitäten und Datenflüsse (sowie ggf. Speicher) gegliedert
- Aktivitäten arbeiten, wenn die von ihnen benötigten Daten vorliegen

  $\rightarrow$ Systemsteuerung durch Datenfluss

Typische Vertreter

- Leitung-Filter-Architekturen, z.B. UNIX pipes
- Regler
- Strukturierte Analyse