6. Architekturmuster und -metaphern

6.1 Motivation für Architekturmuster

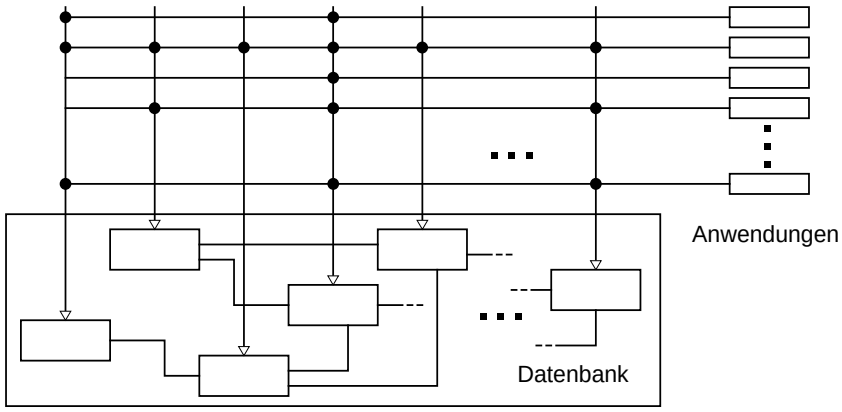
- Bereitstellung bewährter, vorgefertigter Grundstrukturen für wiederkehrende Architekturprobleme
- Abgrenzung zum Entwurfsmuster: Entwurfsmuster sind Muster für Komponenten, nicht für ganze Architekturen
- Abgrenzung zum Rahmen (framework): Rahmen enthalten fertig codierte Teile, während ein Muster nur eine Konstruktionsschablone darstellt.

Architekturmuster – Allgemeine, parametrierbare Architekturschablone für eine typische Problemklasse.

6.2 Einige typische Architekturmuster

Strukturmuster; Beispiel: Matrixmuster

- System besteht aus Menge von Daten- und Funktionsmodulen
- Jede Funktion kann auf jedes Datum zugreifen
- Funktionsmodule enthalten keine permanenten Daten
- Muster für die Klassische Architektur datenbankbasierter Systeme

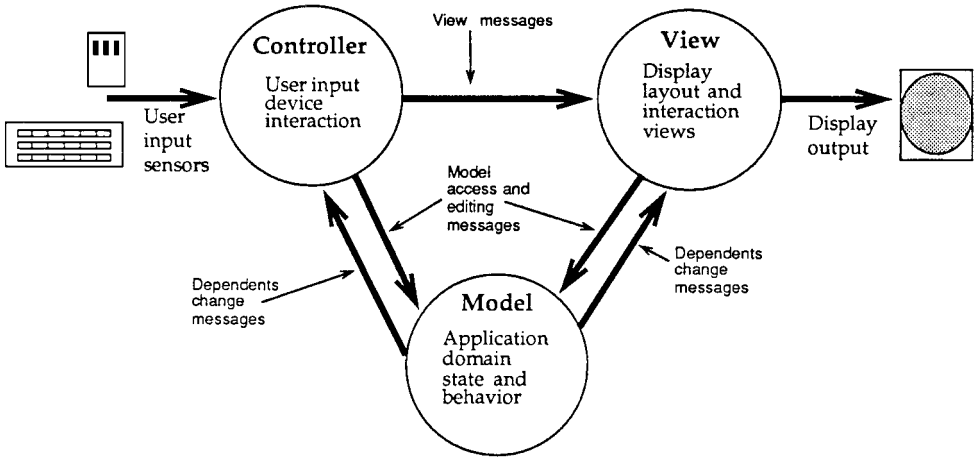


Steuermuster

- **EVA (Eingabe-Verarbeitung-Ausgabe)**
  Ein Steuermodul steuert nacheinander (in Sequenz oder iterativ) Eingabe-, Verarbeitungs- und Ausgabemodule an.
- **Hauptschleife**
  Ein Prozess misst, regelt und steuert, indem er in einer Endlosschleife zyklisch alle Datenquellen (Sensoren, etc.) abfragt und alle Datensenken (Anzeigen, Aktuatoren...) mit aktualisierten Werten versorgt.
- **Hollywood ("Don't call us, we call you")**
  Ein Ereignisverwalter registriert alle Eingabeereignisse und ruft die zugehörigen Dienste auf. Das Anwendungsprogramm enthält kein Hauptprogramm mehr.

Modularisierungs-/Entkopplungsmuster

- **Benutzungshierarchie**
  Ein System ist strikt nach dem Delegationsparadigma organisiert. Die Benutzungsbeziehungen bilden einen gerichteten, azyklischen Graph.
- **Model-View-Controller (MVC)**
  Gliederung eines Systems in ein "Model" (Anwendungslogik, Modell des Anwendungsbereichs), eine "View" (äußere, sichtbare Repräsentation) und einen "Controller" (Behandlung aller Benutzereingaben)



Aus: Krasner und Pope 1988

Verteilungsmuster

- **Client /Server**
  siehe Kapitel 5
- **TP-Monitor**
  Middleware-Architektur für datenbankbasierte Systeme mit einem Transaktionsverwalter als Hauptkomponente
- **Three-Tier**
  Middleware-Architektur für datenbankbasierte Systeme, bei der die Middleware einen Teil der Anwendungslogik enthält
- **Komponentenbus**
  siehe Kapitel 5

6.3 Architekturmetaphern

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

Architekturmetaphern

- sind *Leitbilder* für eine Architektur
- erschließen das Verständnis über analoge, vertraute Bilder

Abgrenzung: Stil - Muster - Metapher:

- Architekturmetapher – Leitbild für das Gliedern und Verstehen einer Architektur
- Architekturstil – eine bestimmte Art des Zusammenwirkens von Komponenten und Interaktionen
- Architekturmuster – Allgemeine, parametrierbare Architektur für eine typische Problemstellung

Einige typische Metaphern

- Die Virtuelle-Maschinen-Metapher
- Die WAM (Werkzeug-Material-Automat)-Metapher
- Die Lagerhaus-Metapher
- Die Steckersystem-Metapher
- Die Agenten-Metapher

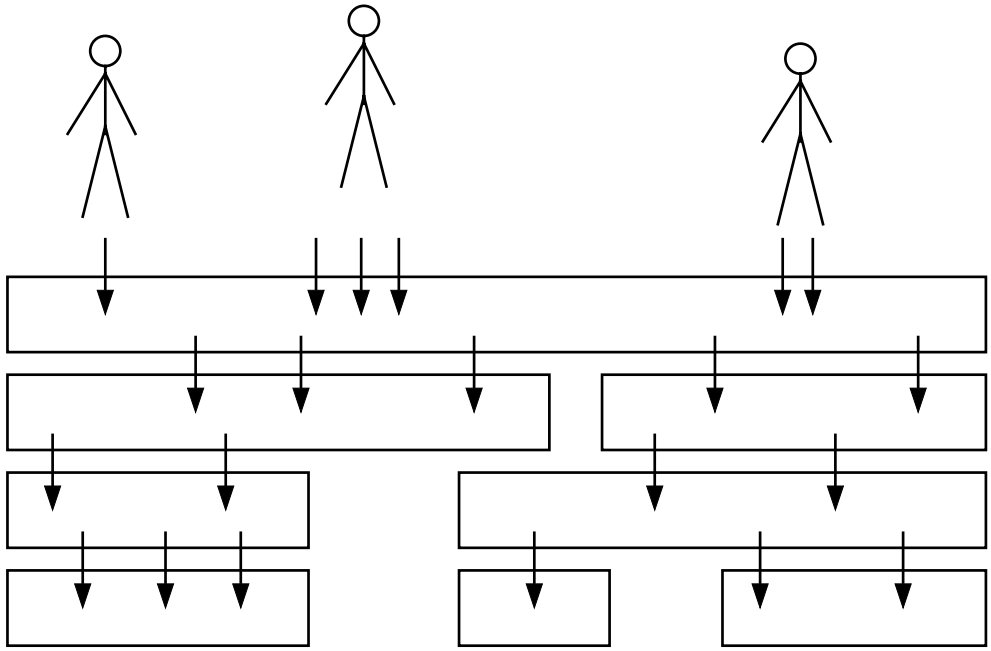
Die Virtuelle-Maschinen-Metapher

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 darunterliegenden Schicht

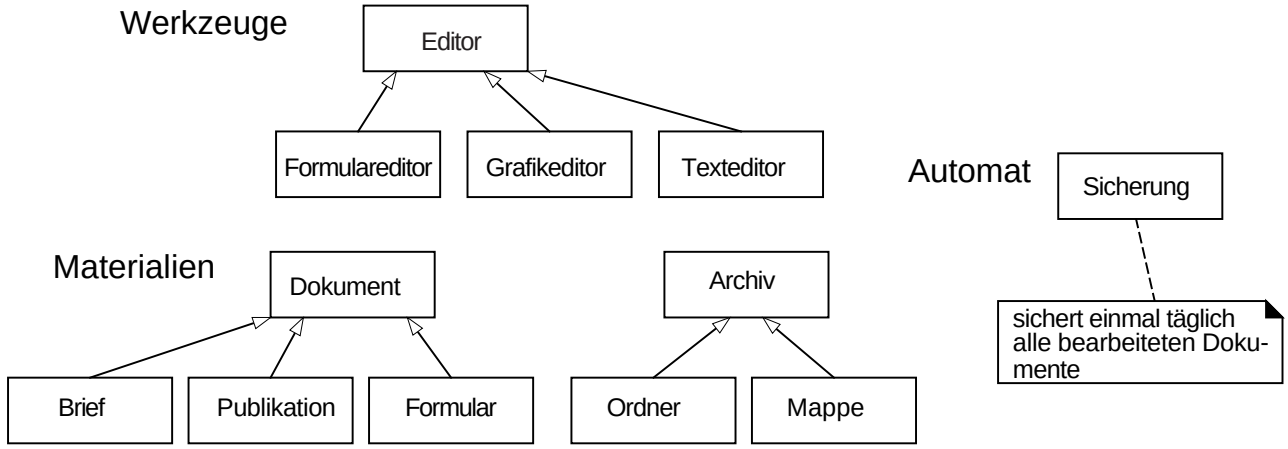
- Die unterste Schicht besteht aus realen Maschinen
- Die oberste Schicht erbringt die Leistungen, die für die Anwender zugänglich sind
- Typisches Beispiel: OSI-Referenzmodell für die Kommunikation von Rechnern



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



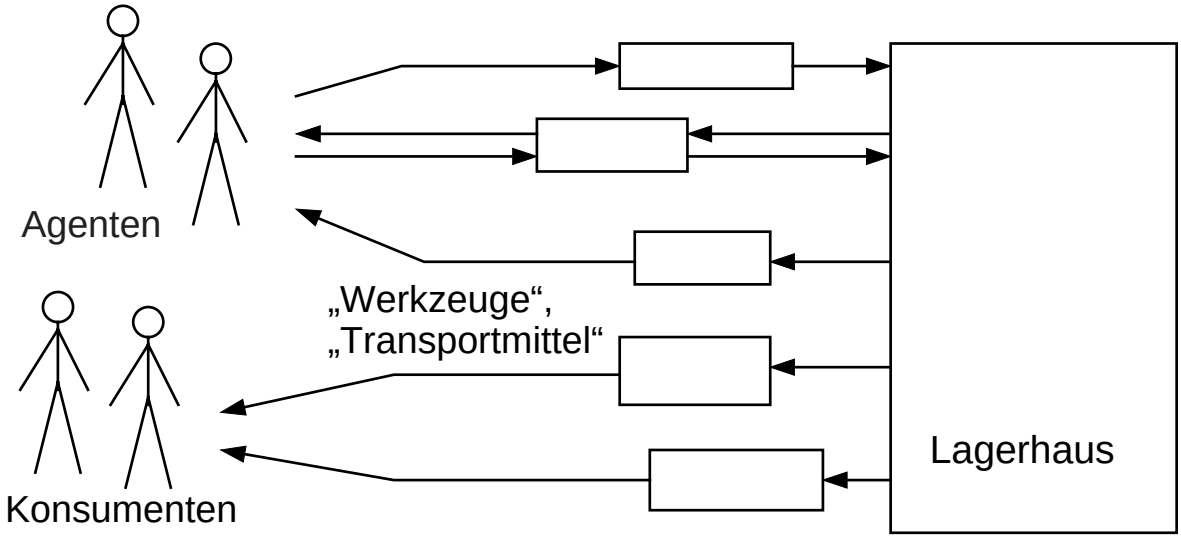
Die Lagerhaus-Metapher

Leitgedanke: Informationen werden wie Waren in einem Lagerhaus eingelagert und abgerufen.

Das System besteht aus

- einem „*Lagerhaus*“ (warehouse, repository), in dem Informationen gelagert sind
- *Agenten*, welche Informationen einlagern, ordnen und bearbeiten
- *Konsumenten*, welche auf Informationen aus dem Lagerhaus abrufen

- Alle Informationen sind zentral im Lagerhaus gelagert
- Agenten und Konsumenten tauschen untereinander keine Informationen aus
- Typisches Beispiel: Software-Entwicklungswerkzeuge



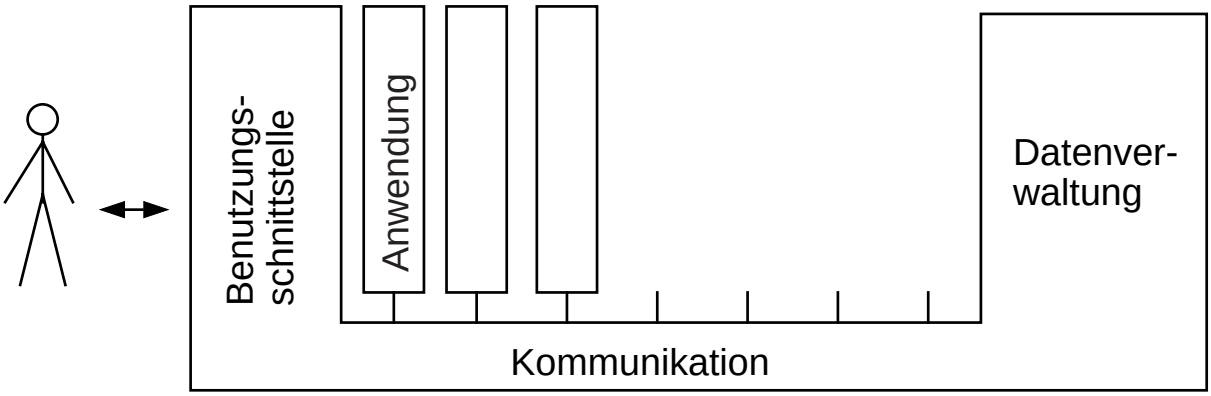
Die Steckersystem-Metapher

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

Ein System besteht typisch aus

- ☆ einem *Rahmen*
  - ✧ mit Datenverwaltungsdiensten, Kommunikationsdiensten und Benutzerschnittstelle
  - ✧ mit Steckplätzen für Anwendungen
  - ✧ in der Regel vorgefertigt
- ☆ *Anwendungen*,
  - ✧ in die vorhandenen Steckplätze des Rahmens eingesteckt

- ✧ nutzen die vorhandenen Dienste des Rahmens
- ✧ werden über vorhandene Benutzersschnittstelle angesprochen
- ✧ in der Regel problem- und kundenspezifisch angefertigt
- ✧ kann auch für Grundprobleme vorgefertigt sein



Die Agenten-Metapher

Leitgedanke: System besteht aus einer Menge kooperierender Agenten.

- ❍ Jeder Agent ist zuständig für eine bestimmte, weitgehend in sich geschlossene Aufgabe
- ❍ Agenten können ortsfest oder mobil sein