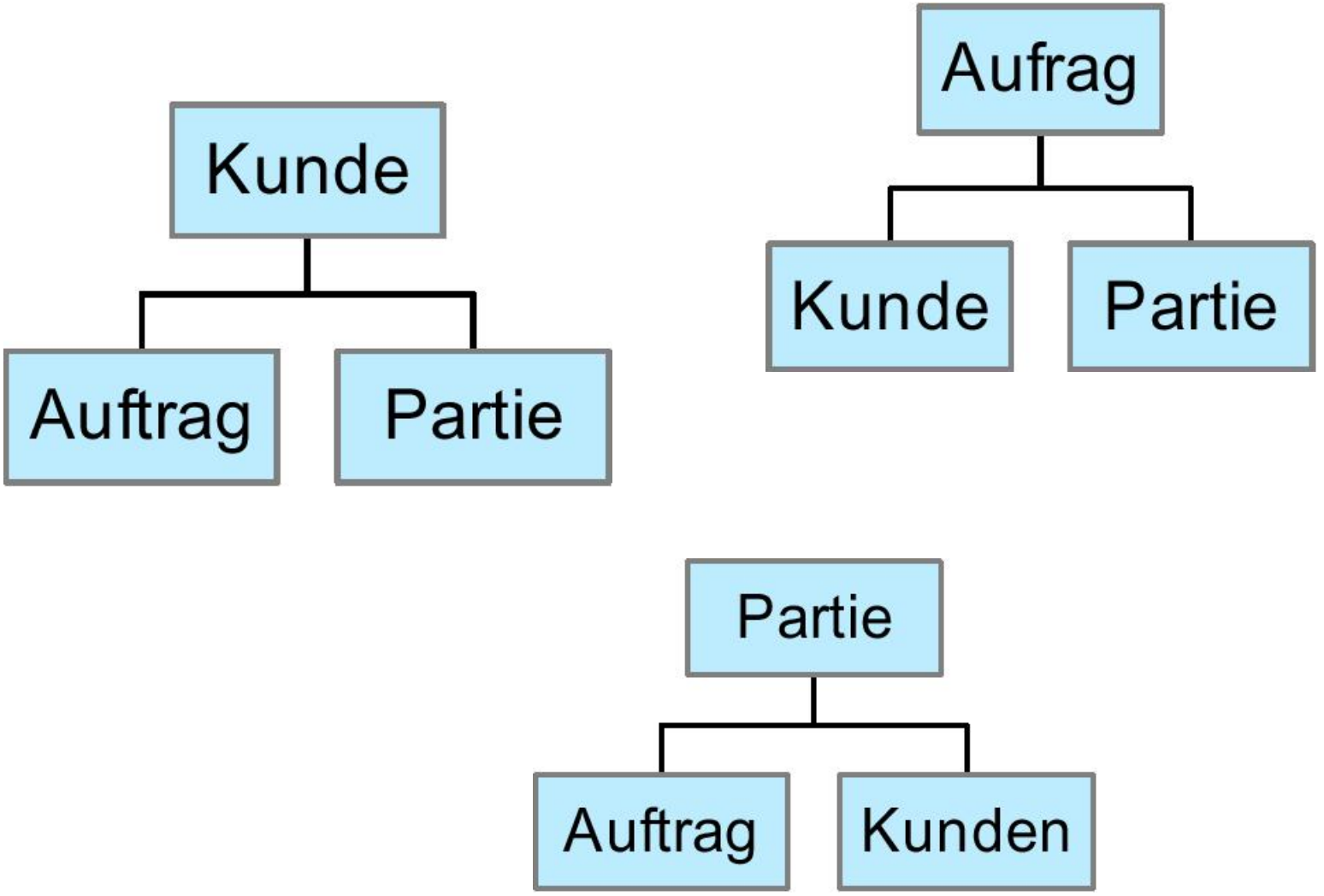
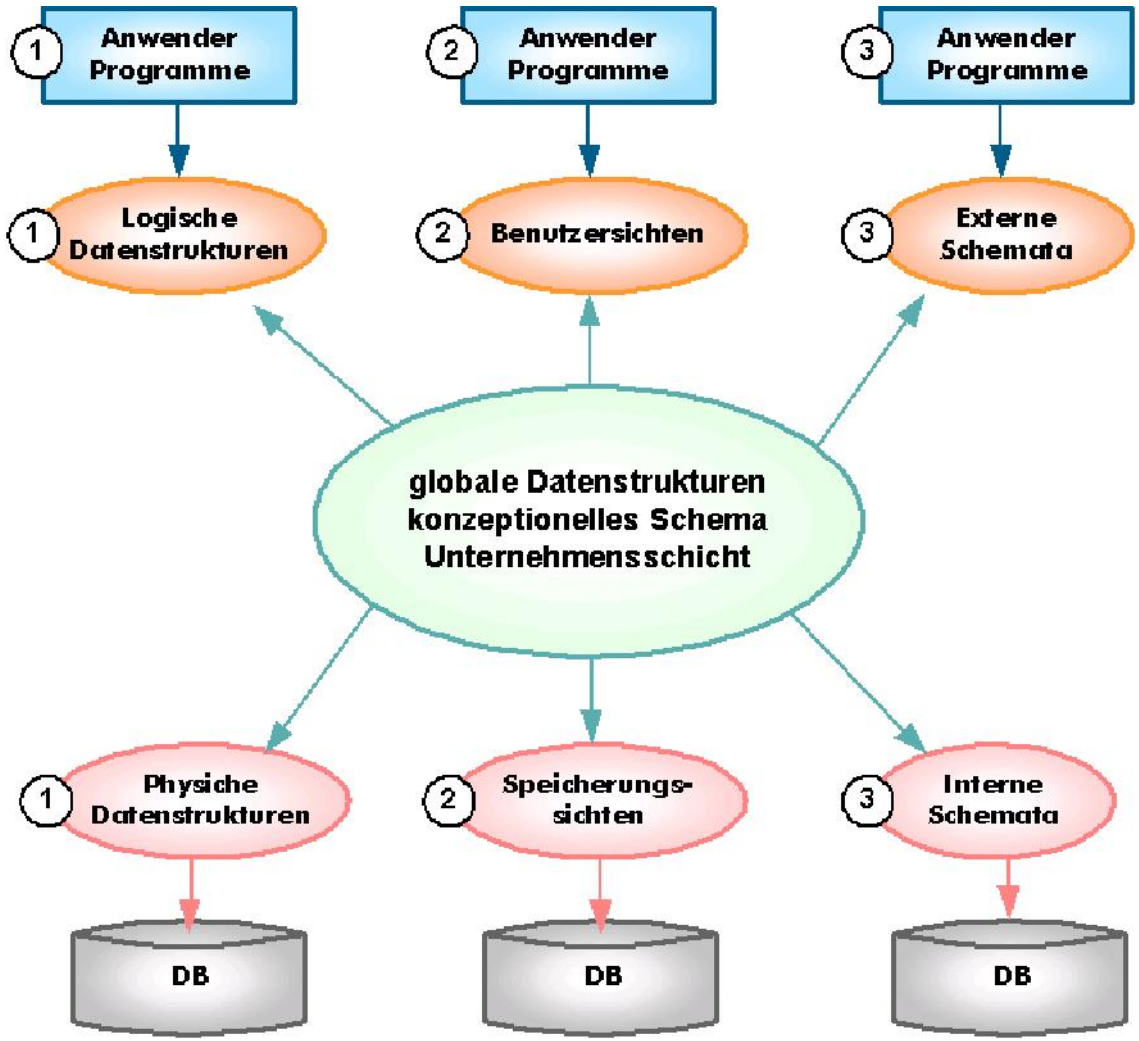
Modellierung

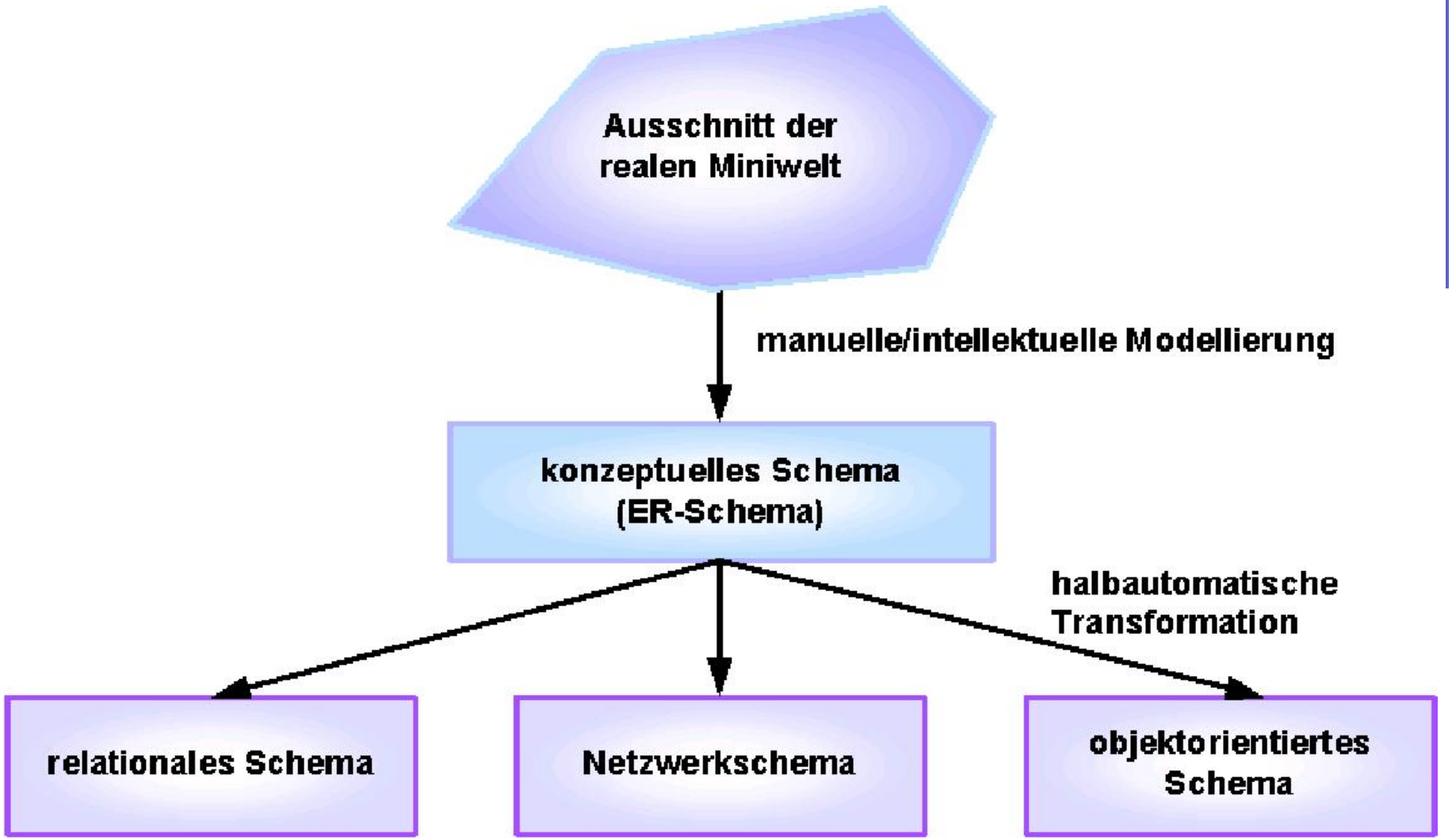
Hierarchisches Datenmodell



Ebenenarchitektur nach ANSI/SPARC



Übersicht Datenmodellierung



UML – Unified Modelling Language

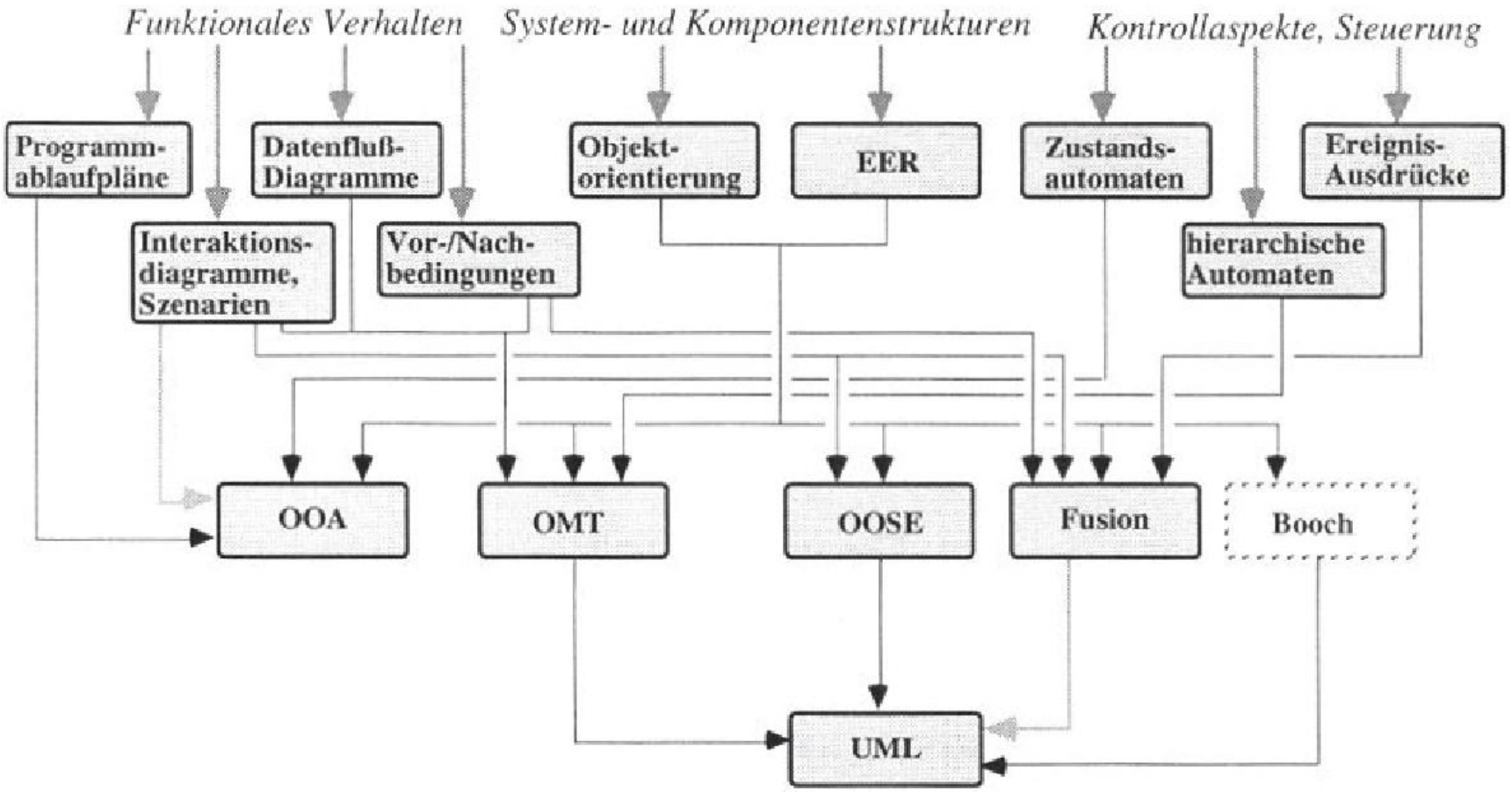
- Spezifikation, Visualisierung, Konstruktion und Dokumentation von Software
- De-facto-Standard für objektorientierte Softwareentwicklung
- Einheitliche Notation
- Offen für neue Konzepte

Viele notationelle Freiheiten

Prozessunabhängiger Formalismus

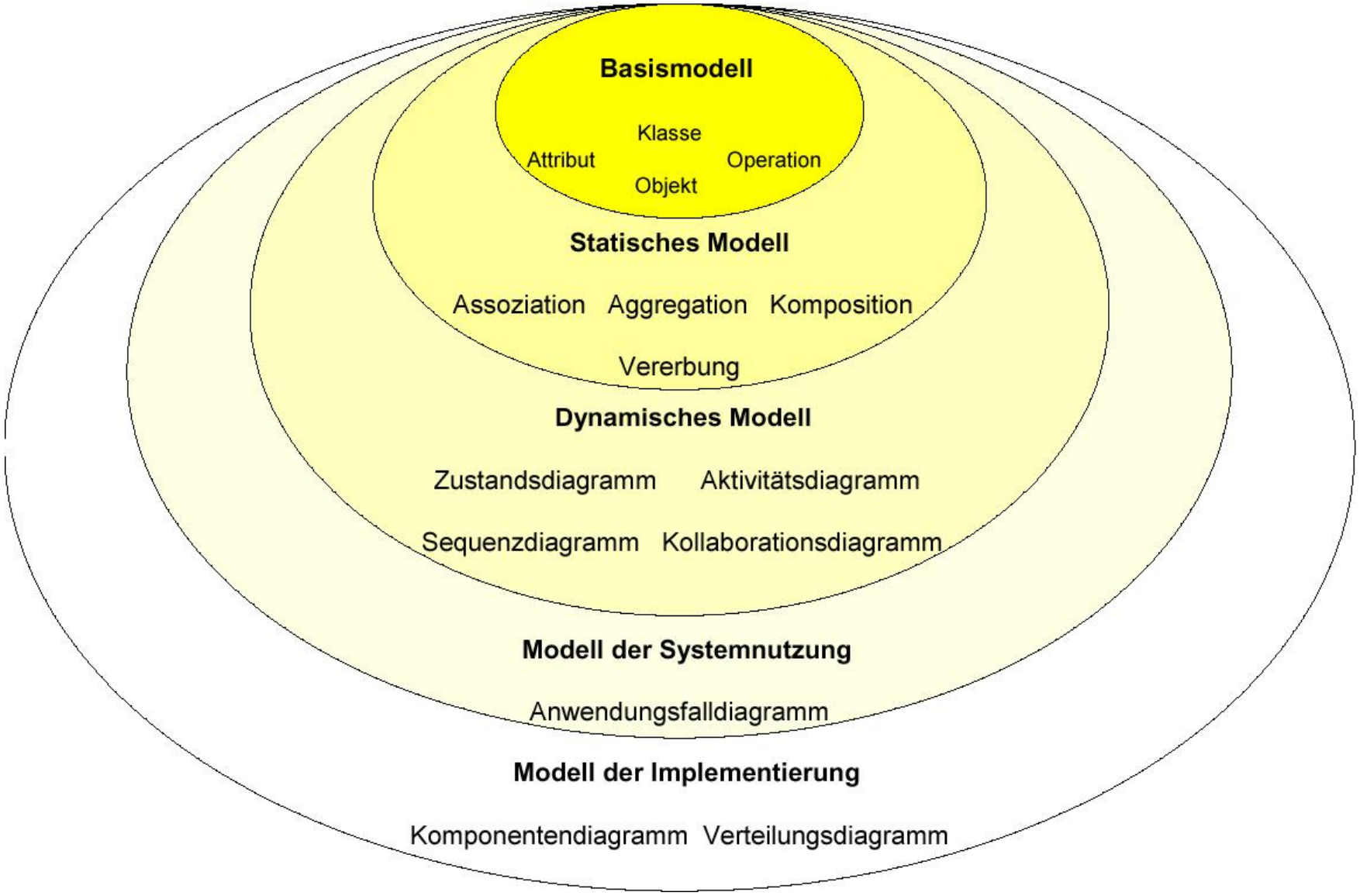
- Darstellung eines Sachverhalts in unterschiedlichen Varianten
- Verschiedene Abstraktionsstufen
- Werkzeugunterstützung durch CASE-Tools (Computer Aided Software Engineering)
  z.B. Rational Rose, iLogix Rhapsody, Together ControlCenter, Poseidon, ...

Entstehung der UML



UML ist seit 1997 Standard für objektorientierte Modellierung
UML entstammt der OMG (Object Management Group)

Modelltypen



UML Diagramme

Sichtweise	Diagrammtyp	Aufgaben
Anforderungen	1. Anwendungsfalldiagramm	• Benutzersicht darstellen • Systemsicht zur Umwelt definieren • Überblick über die Funktionalität des Systems geben
Struktur	2. Klassendiagramm 3. Objektdiagramm	• Systemstruktur darstellen
Dynamik	4. Zustandsdiagramm 5. Aktivitätsdiagramm 6. Sequenzdiagramm 7. Kollaborationsdiagramm	• Systemdynamik darstellen • Beziehungen zwischen Objekten und Aktivitäten definieren
Implementierung	8. Komponentendiagramm 9. Verteilungsdiagramm	• physikalische Architektur des Zielsystems definieren

Basismodell - Klasse (1/2)

Klasse: Kollektion von Objekten mit gleichen Eigenschaften (Attributen), gemeinsamer Funktionalität (Operationen) sowie gemeinsamen Beziehungen zu anderen Objekten und gleicher Semantik.

Attribute: charakteristische Daten bzw. Eigenschaften von Objekten bestehend aus Namen (Typangabe), Wert (bei Objekten) oder Initialwert (bei Klassen).

Klassenattribute: nur ein Attributwert für alle Objekte einer Klasse.
Beispiel: Der Attributwert von Symbol ist eine für alle Objekte der Klasse identische Graphik.

Operationen: ausführbare Tätigkeiten (Funktionen oder Algorithmen). Funktionalität eines Objektes.

Klassenoperationen: einer Klasse zugeordnet und nur auf sie anwendbar (nicht auf Objekte der Klasse), auch wenn keine Objekte der Klasse existieren.
Beispiel: anfragenSymbol() gibt das graphische Symbol für einen Widerstand zurück.

Basismodell - Klasse (2/2)

Verkapselung: Verbergen von Attributen (Daten) und Operationen (Verhalten, Implementierung) einer Klasse.

Geheimnisprinzip (information hiding): Änderung und Abfrage des Zustands und Eigenschaften ausschließlich über Operationen.

Zugriffsmodifikatoren:
- private
protected
+ public

<<...>> Stereotyp (gehört zu den **extensibility constructs**)**:** dient der Klassifikation von Elementen eines UML Modells auf Basis vordefinierter Elemente.

Basismodell - Objekt

:Widerstand

R100

R100:Widerstand

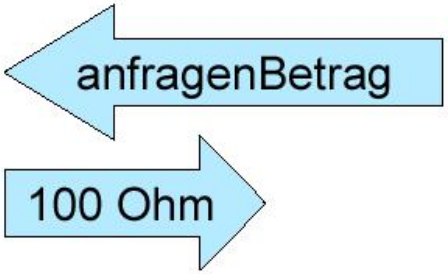
Betrag = 100 Ohm
Leistung = 100 Watt
Toleranz = 0,05

anfragenBetrag()
anfragenLeistung()
anfragenToleranz()

Objekt: Gegenstand, Person, Begriff mit eindeutiger, nicht veränderbarer Identität und charakteristischen Eigenschaften (Attribute mit Attributwerten).

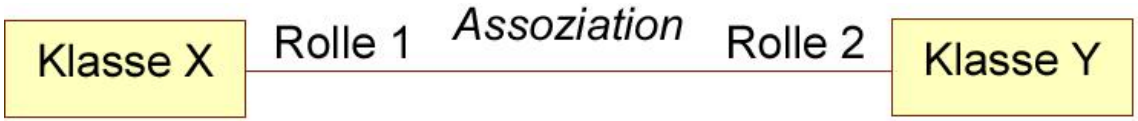
Instanz: Objekt einer Klasse.

Zustand: Kombination aller Attributwerte eines Objekts.



Botschaft: Aufforderung eines Objekts (Sender) an ein anderes Objekt (Empfänger) eine der Botschaft gleichnamige Operation auszuführen.

Statisches Modell - Assoziation



Assoziation: Menge von Beziehungen zwischen Objekten angegeben als Zusammenhang zwischen den zugehörigen Klassen
(entspricht konzeptuell einem Beziehungstyp im ER-Ansatz).

Rolle: Funktion eines Objekts in einer Assoziation.

Assoziation

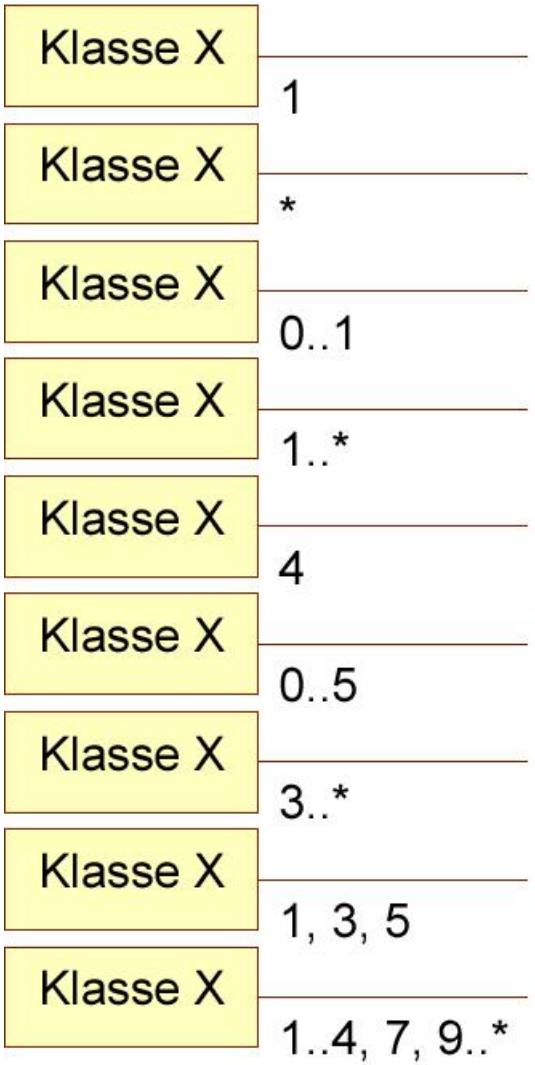
Aggregation
(„ist-Teil-von“-Assoziation)

Komposition
(starke Aggregationsform)

Link: Instanz einer Assoziation.

Kardinalität einer Assoziation

Kardinalität: zwischen einem Objekt der Klasse X und einem beliebigen Objekt besteht:



genau eine Beziehung (Muss-Beziehung)

null, eine oder mehrere Beziehungen (Kann-Beziehung)

null oder eine Beziehung (Kann-Beziehung)

mindestens eine Beziehung (Muss-Beziehung)

genau vier Beziehungen (Muss-Beziehung)

null bis fünf Beziehungen (Kann-Beziehung)

mindestens drei Beziehungen (Muss-Beziehung)

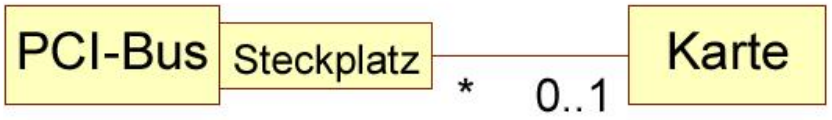
eine, drei oder fünf Beziehungen (Muss-Beziehung)

nicht keine, fünf, sechs oder acht Beziehungen (Muss-Beziehung)

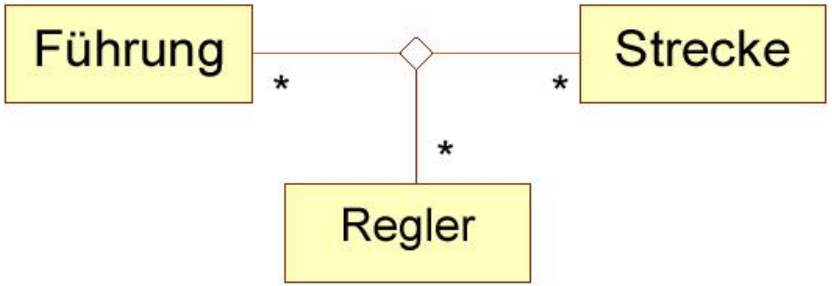
Assoziationen

Navigation

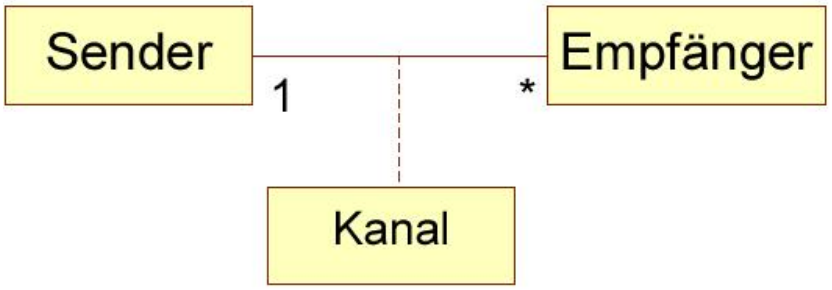
Qualifizierte Assoziation



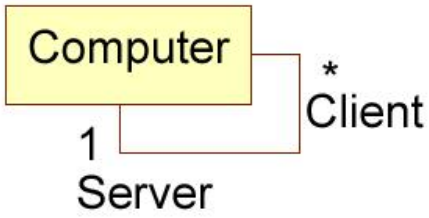
Tertiäre Assoziation



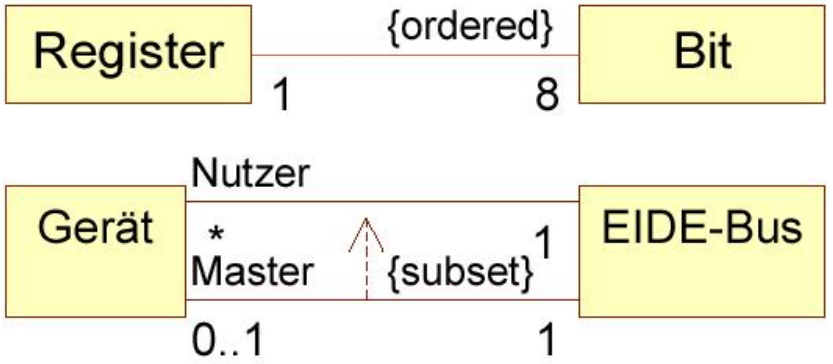
Assoziationsklasse

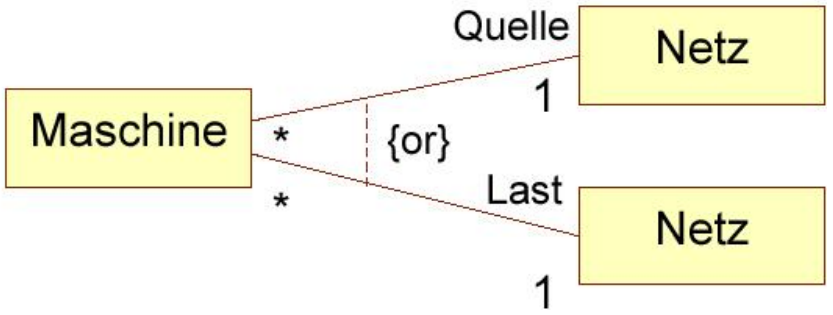


Autoassoziation

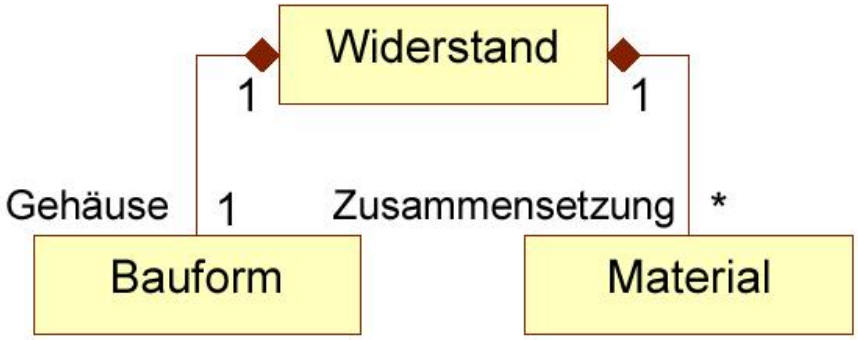


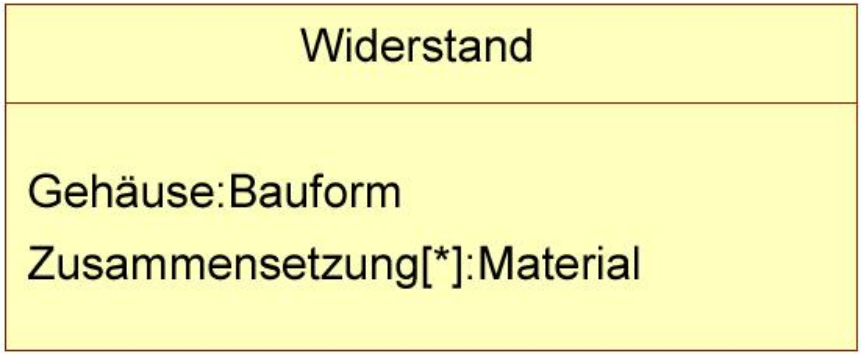
Assoziationen mit Einschränkung

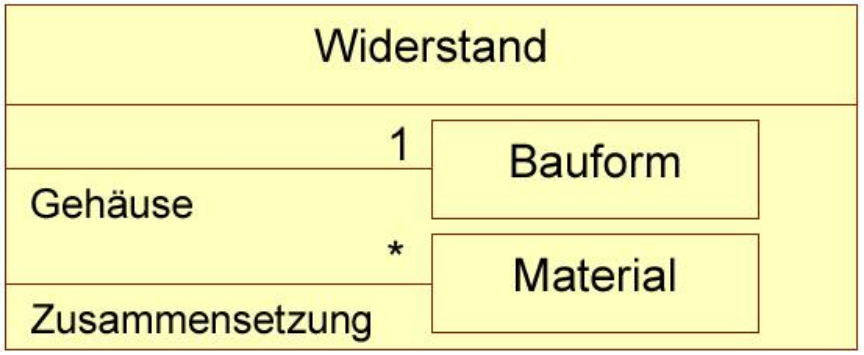




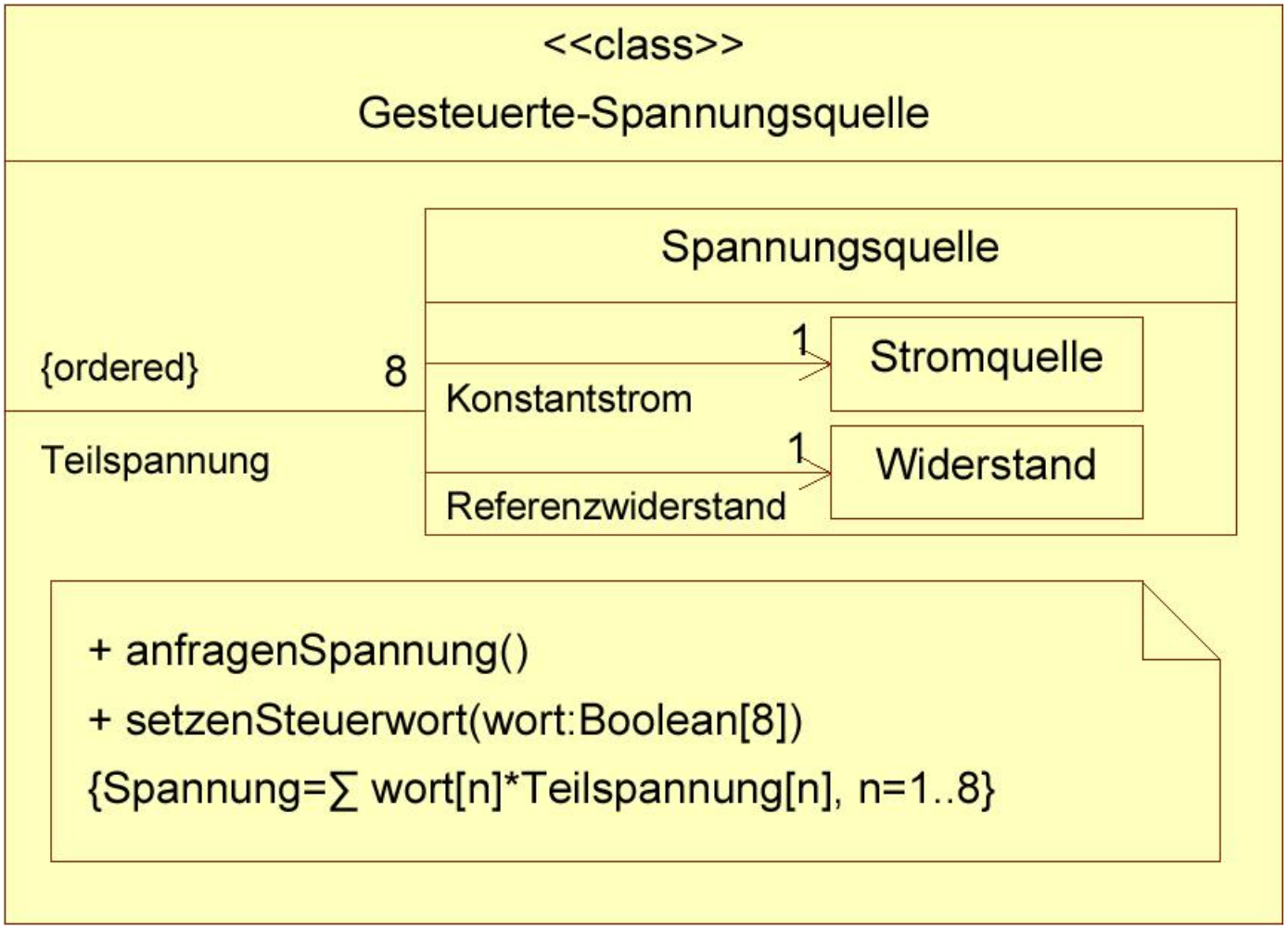
Komposition



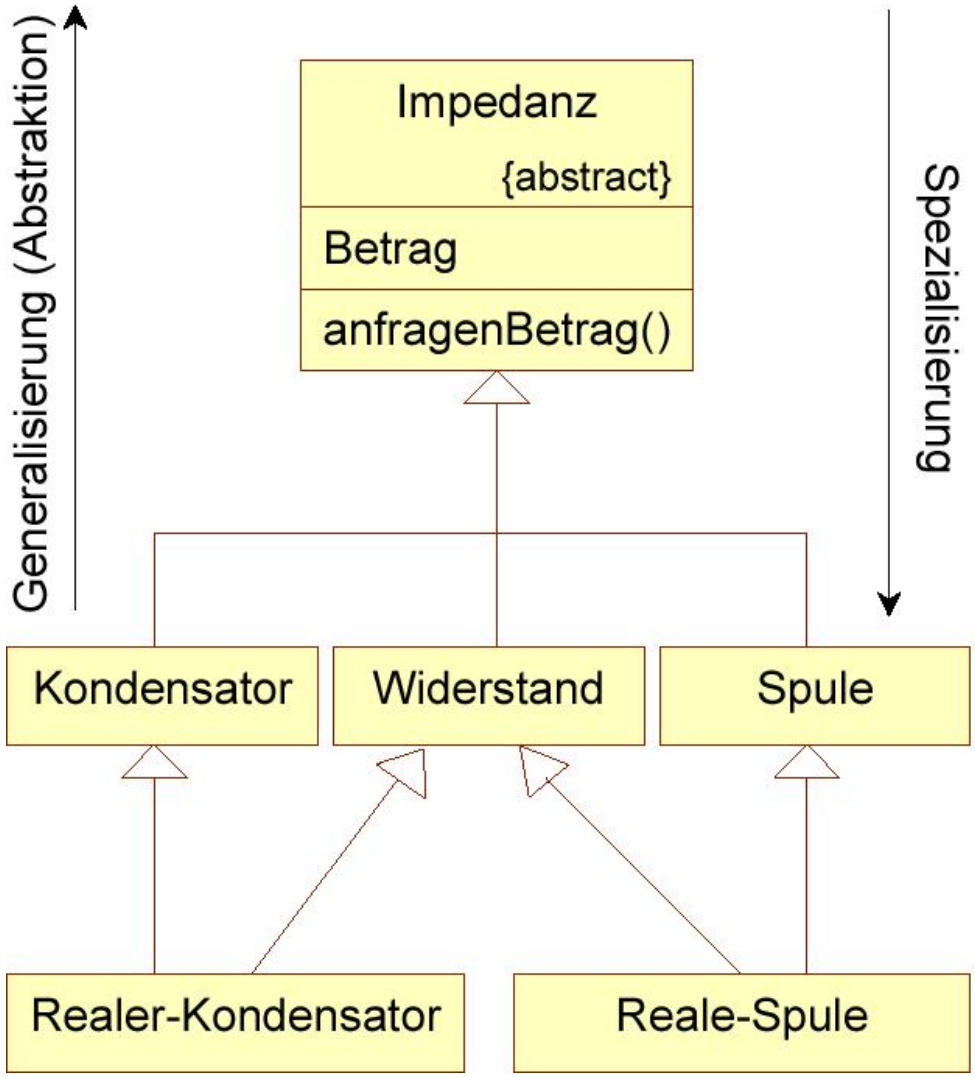




Beispielkomposition



Vererbung



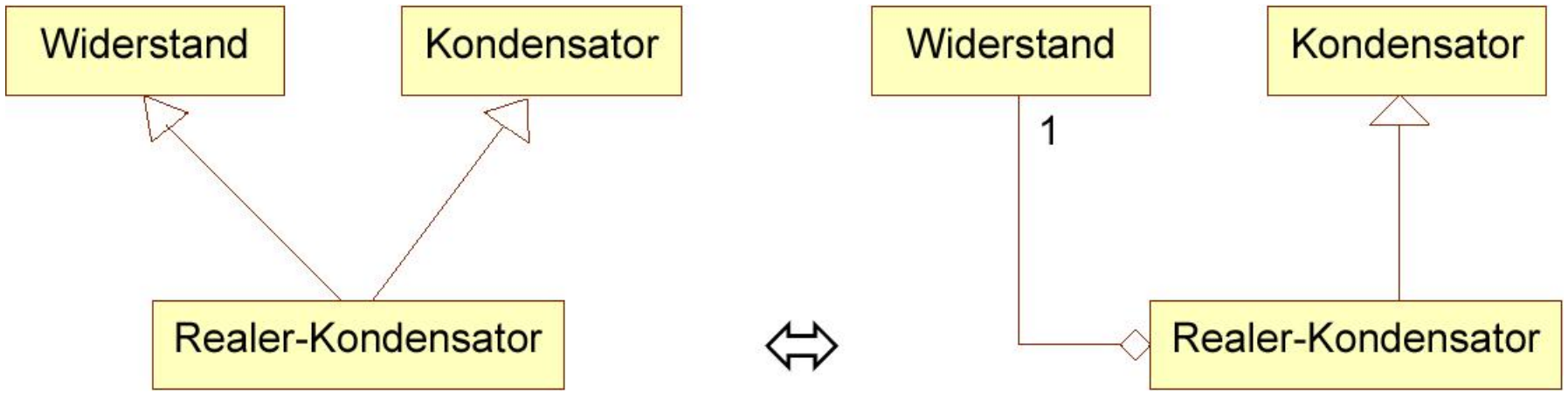
Vererbung: Spezialisierung von Klassen verbunden mit Weitergabe und Erweiterung von Eigenschaften und Operationen.
Ziele: z.B. Vermeidung von Redundanz, Widerverwendung.

Polymorphismus: Verwendung gleicher Namen für gleichartige aber semantisch verschiedene Operationen bei abgeleiteten Klassen (Unterschiedliche Implementierung der Operation anfragenBetrag() bei Spule und Reale-Spule).

Mehrfachvererbung: abgeleitete Klasse erbt direkt von mind. zwei Klassen.

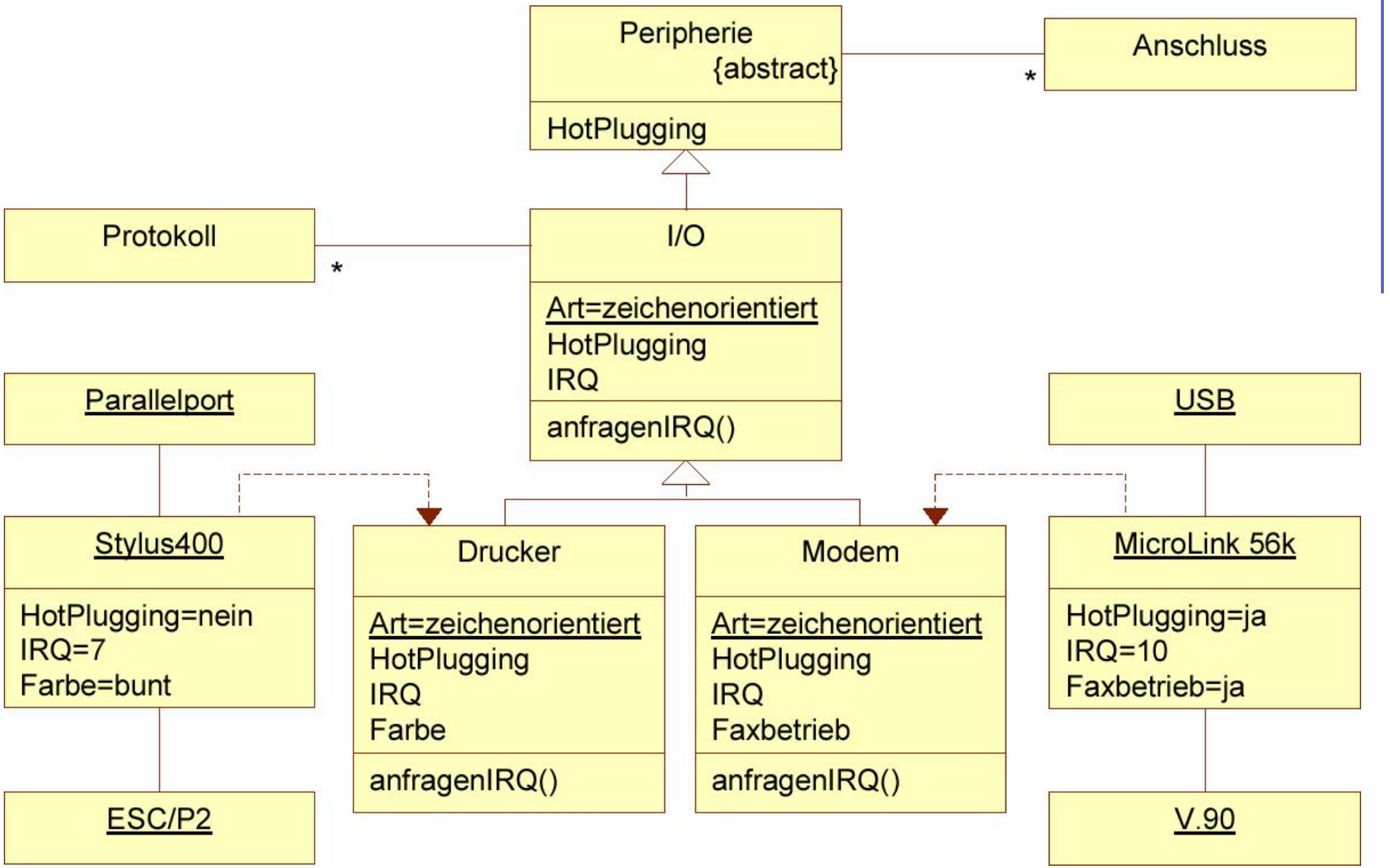
Mehrfachvererbung?

Vererbung <-> Aggregation



Mehrfachvererbungen sollten vermieden werden

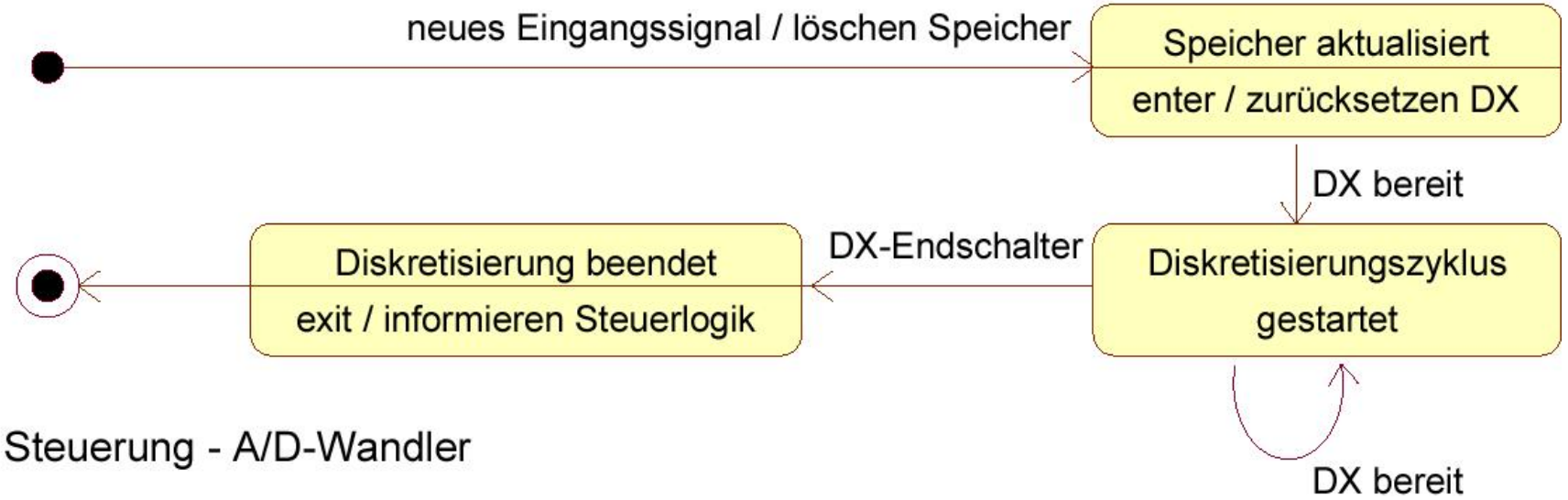
Klassen und Objekte



Zustandsdiagramm (1/2)

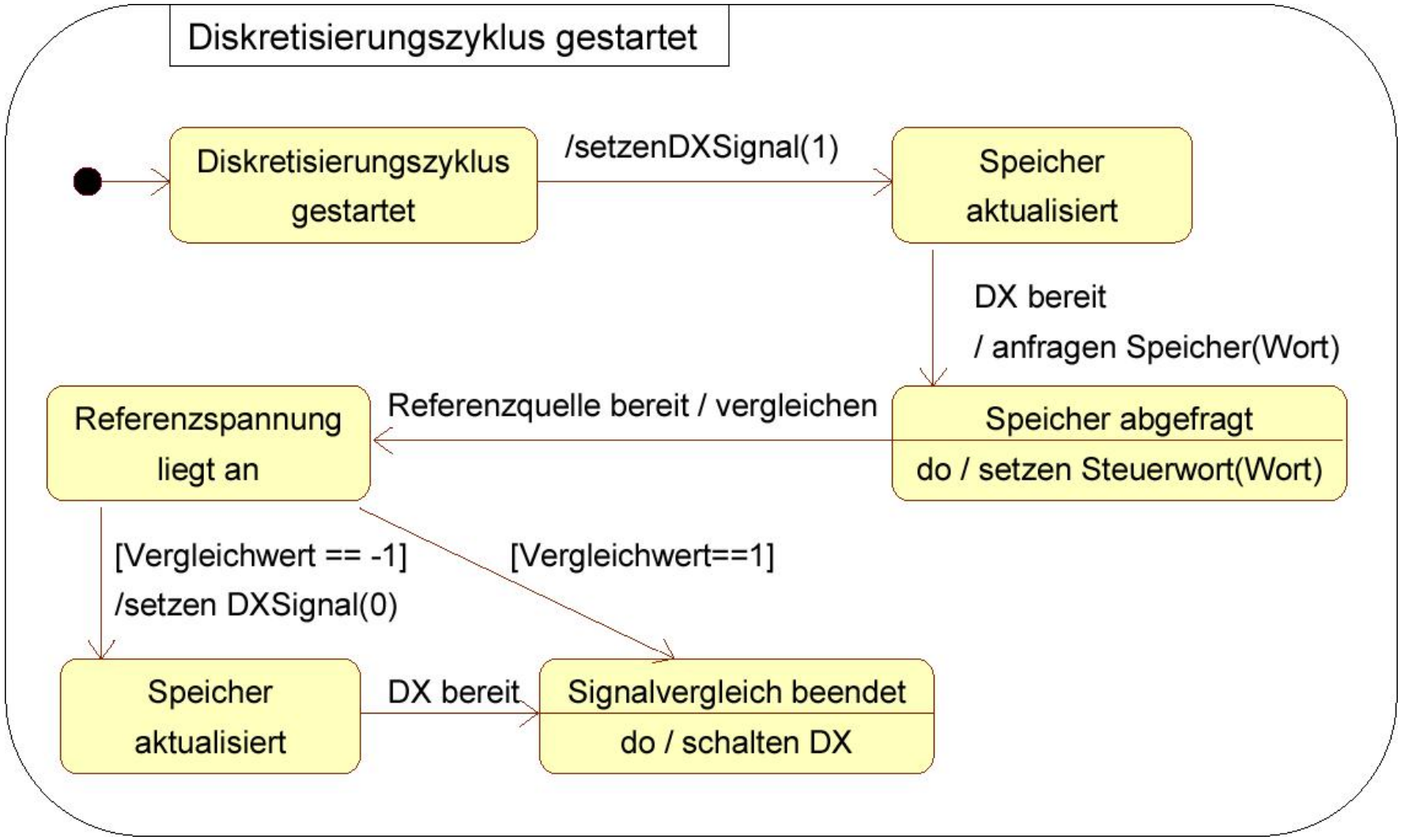
Zustandsdiagramm: beschreibt dynamisches Verhalten von Komponenten (Objekte oder Operationen):

- Interaktionen zwischen System und Umgebung, um Operationen auszuführen
- Berücksichtigt Zustandsabhängigkeiten von Operationen mit ihren Wechselwirkungen
- Ereignisbeschreibungen detaillieren Ereignisse
- Aktionsbeschreibungen präzisieren Aktionen
- Konzepte entsprechen den hierarchischen Automaten

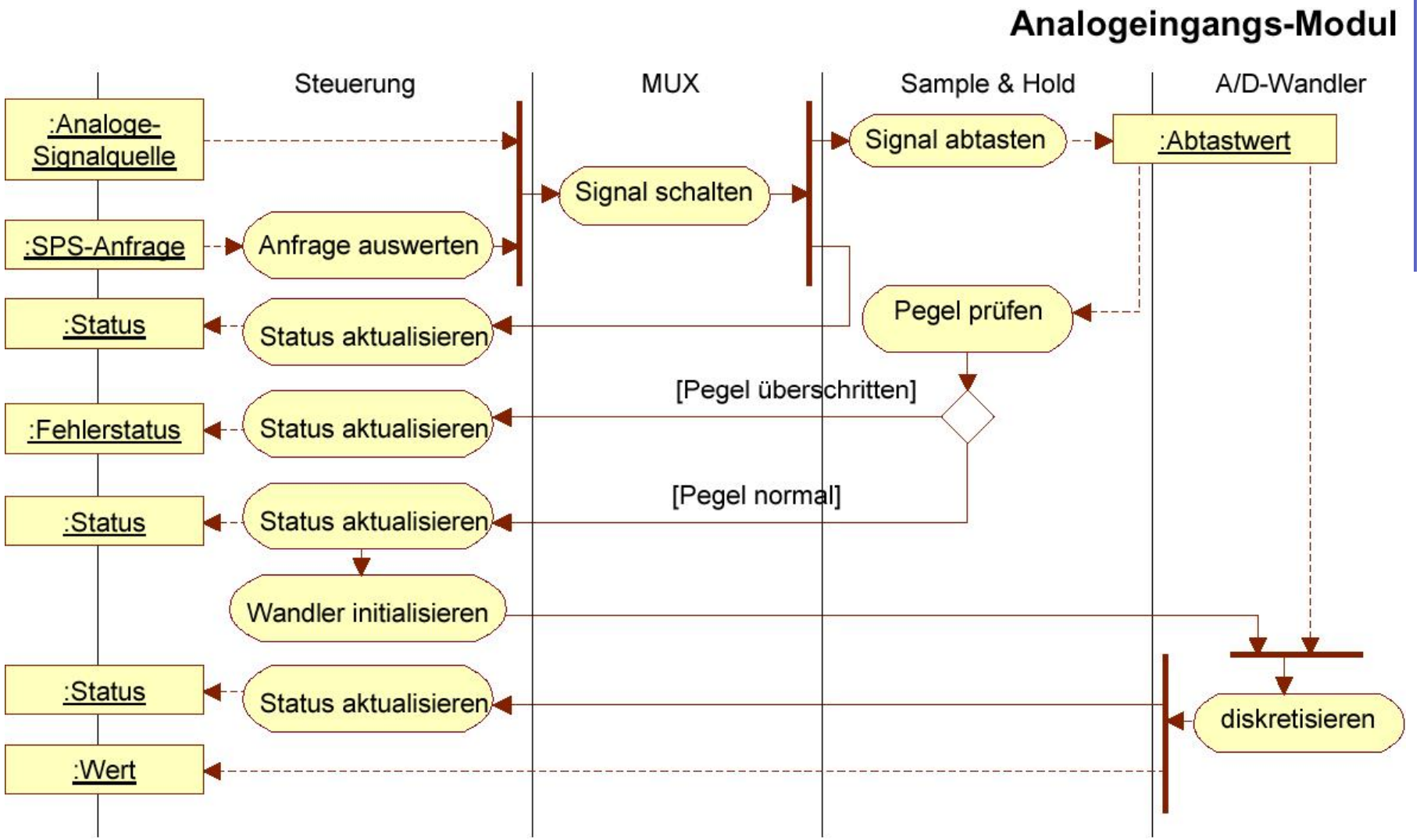


Steuerung - A/D-Wandler

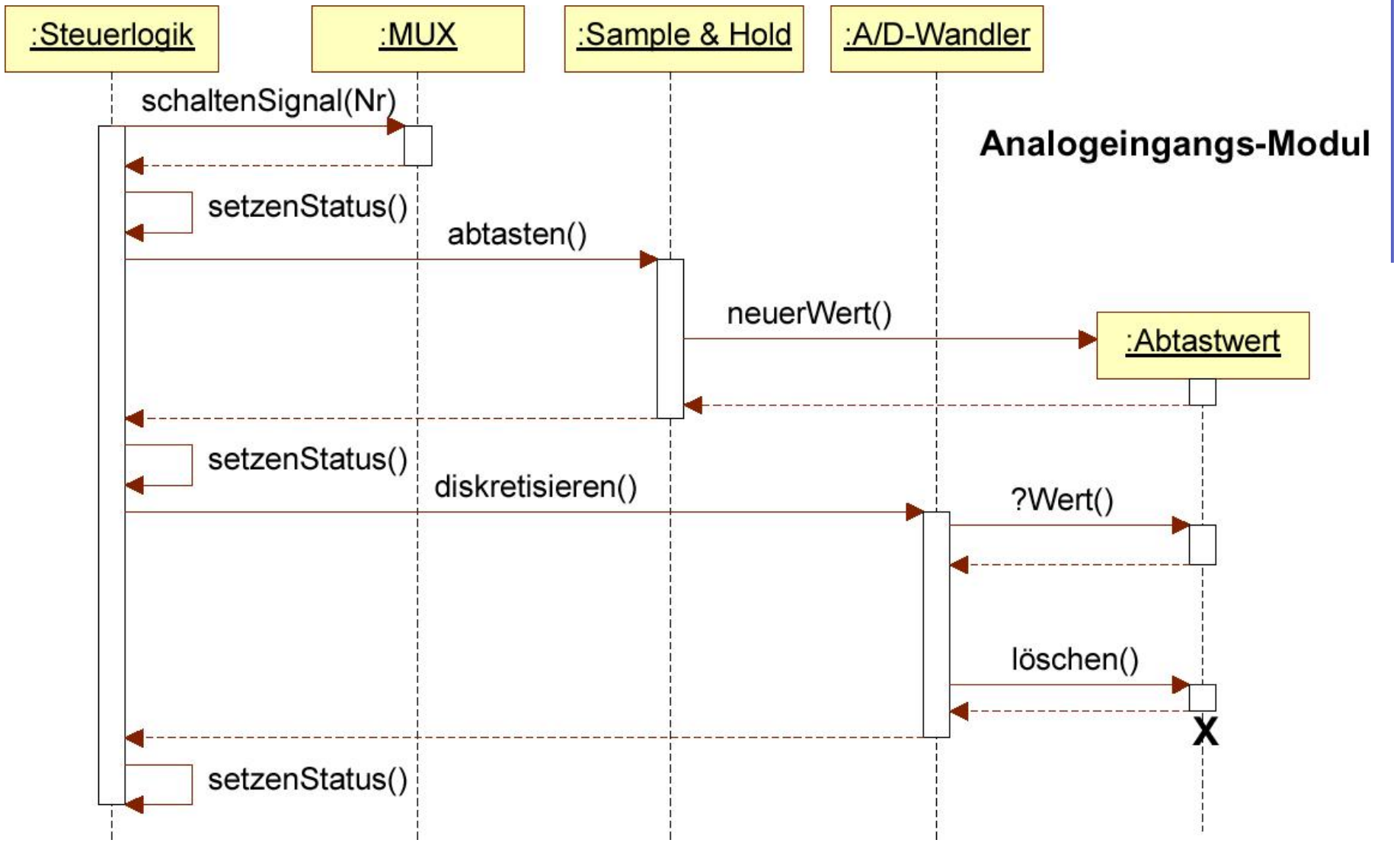
Zustandsdiagramm (2/2)



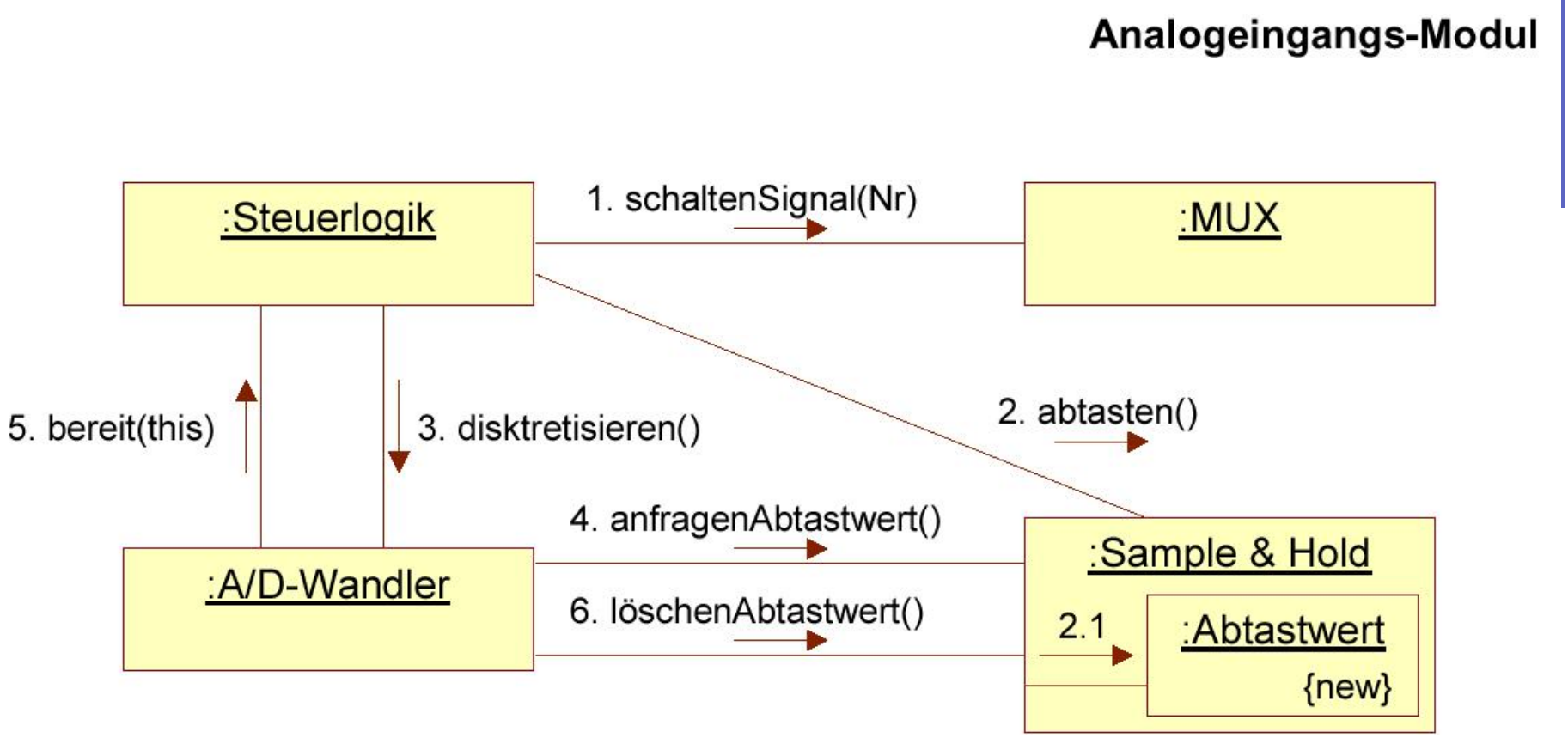
Aktivitätsdiagramm



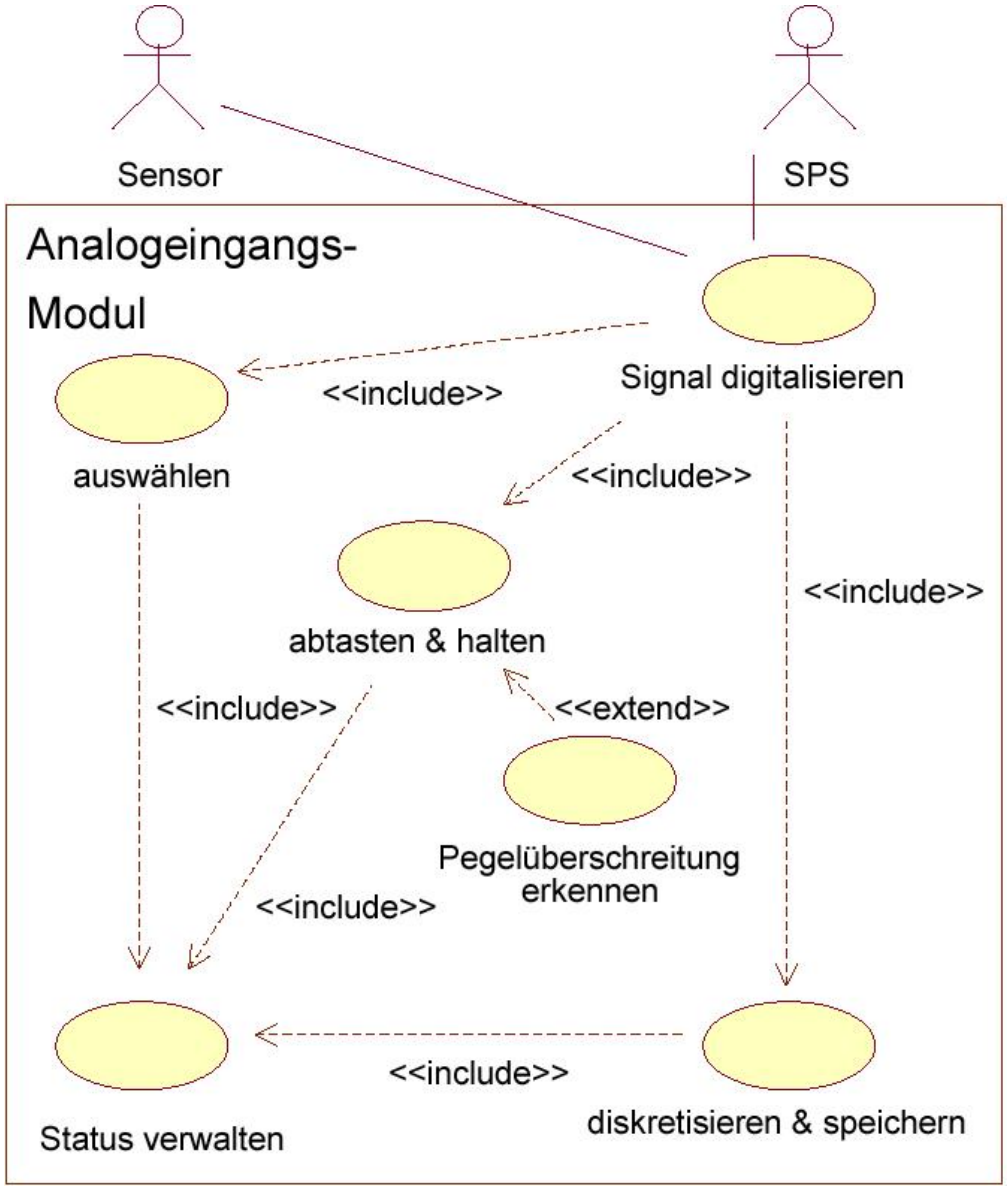
Sequenzdiagramm



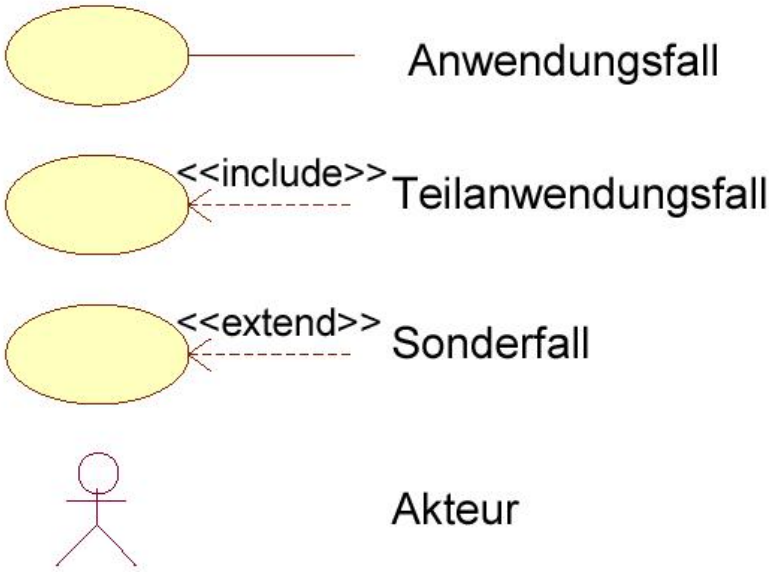
Kollaborationsdiagramm



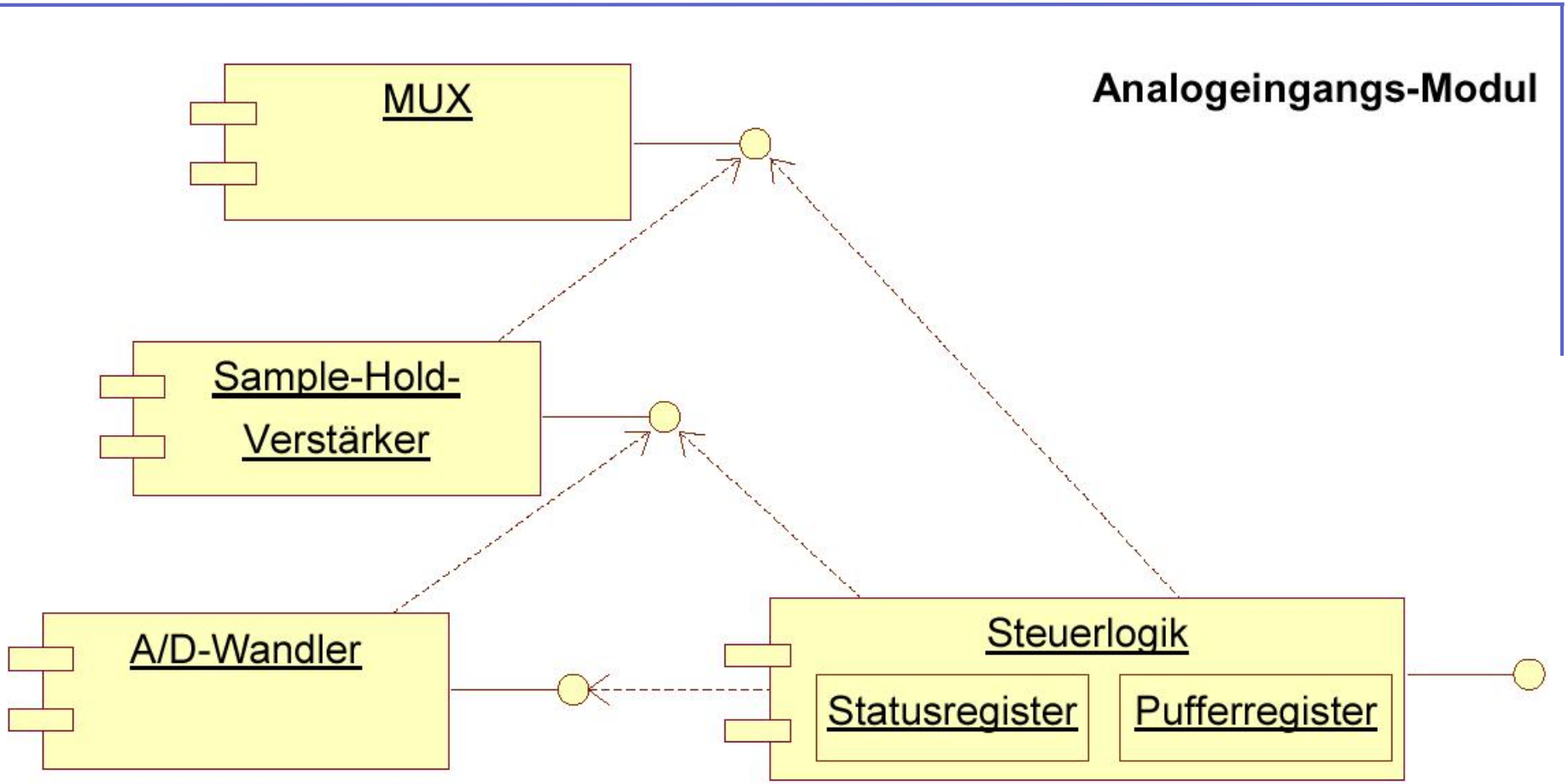
Anwendungsfalldiagramm



Anwendungsfalldiagramme zeigen die externen Schnittstellen eines Systems als Teil der externen Kommunikation und seine Hauptaufgaben (externe Funktionen) sowie ggf. ihre Zusammenhänge. Sie dienen zur Überprüfung des Modells, sind Grundlage für funktionale Test.

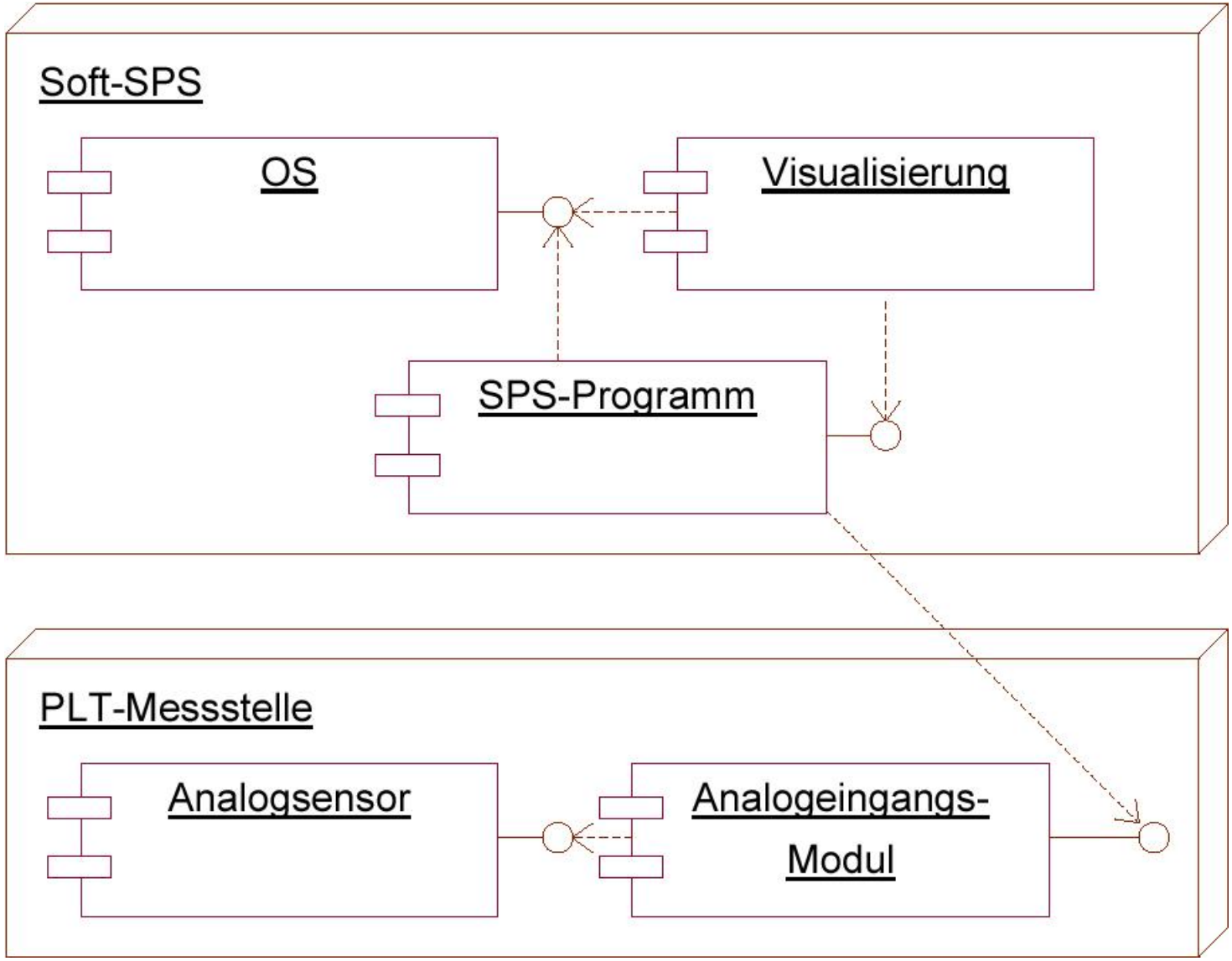


Komponentendiagramm

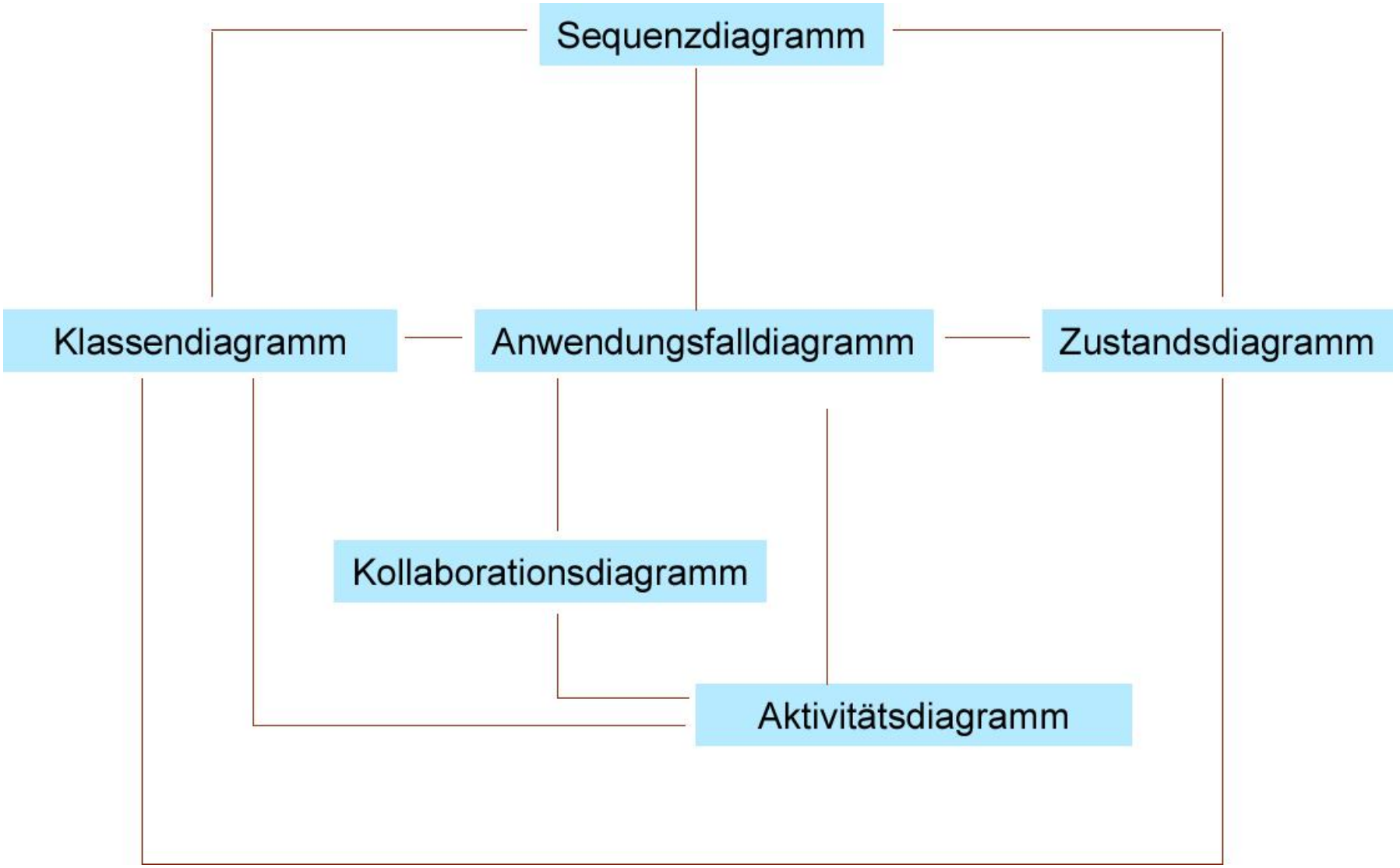


Schnittstelle (interface): Spezifikation des extern sichtbaren Verhaltens eines Modellelements.

Verteilungsdiagramm

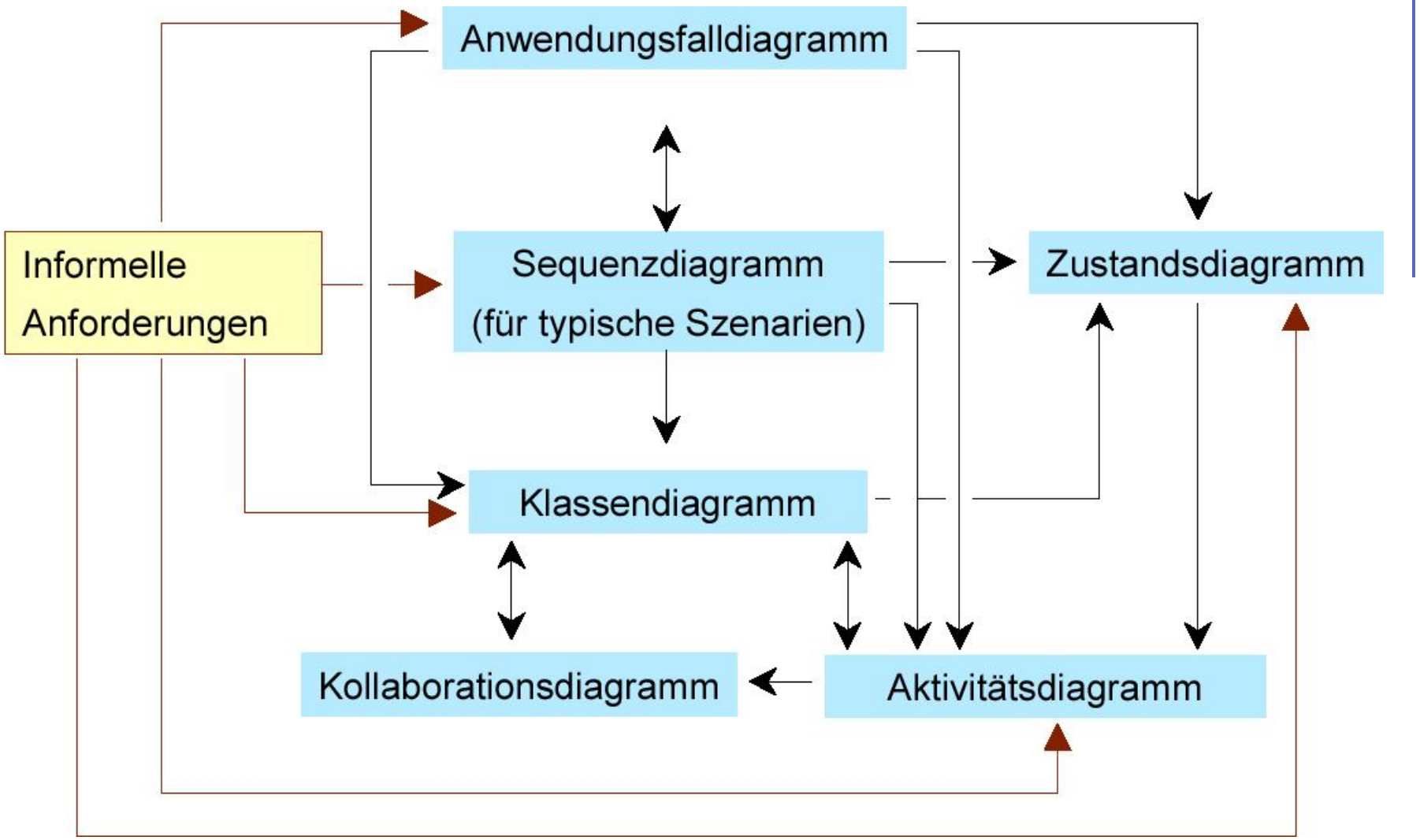


UML Beschreibungen und Zusammenhänge



In Anlehnung and Partsch

Struktur einer möglichen Vorgehensweise



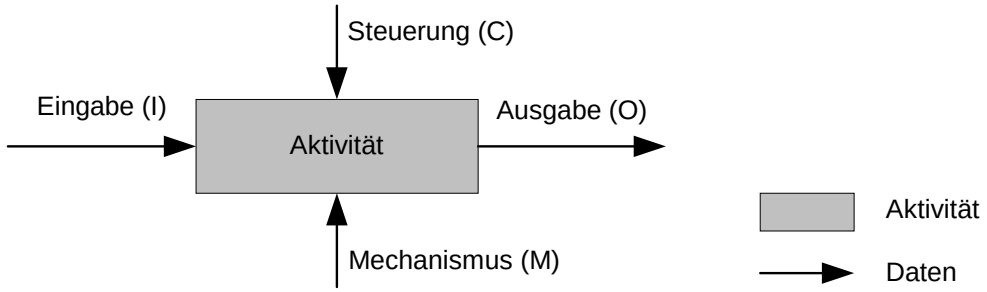
In Anlehnung and Partsch

Structured Analysis and Design Technique

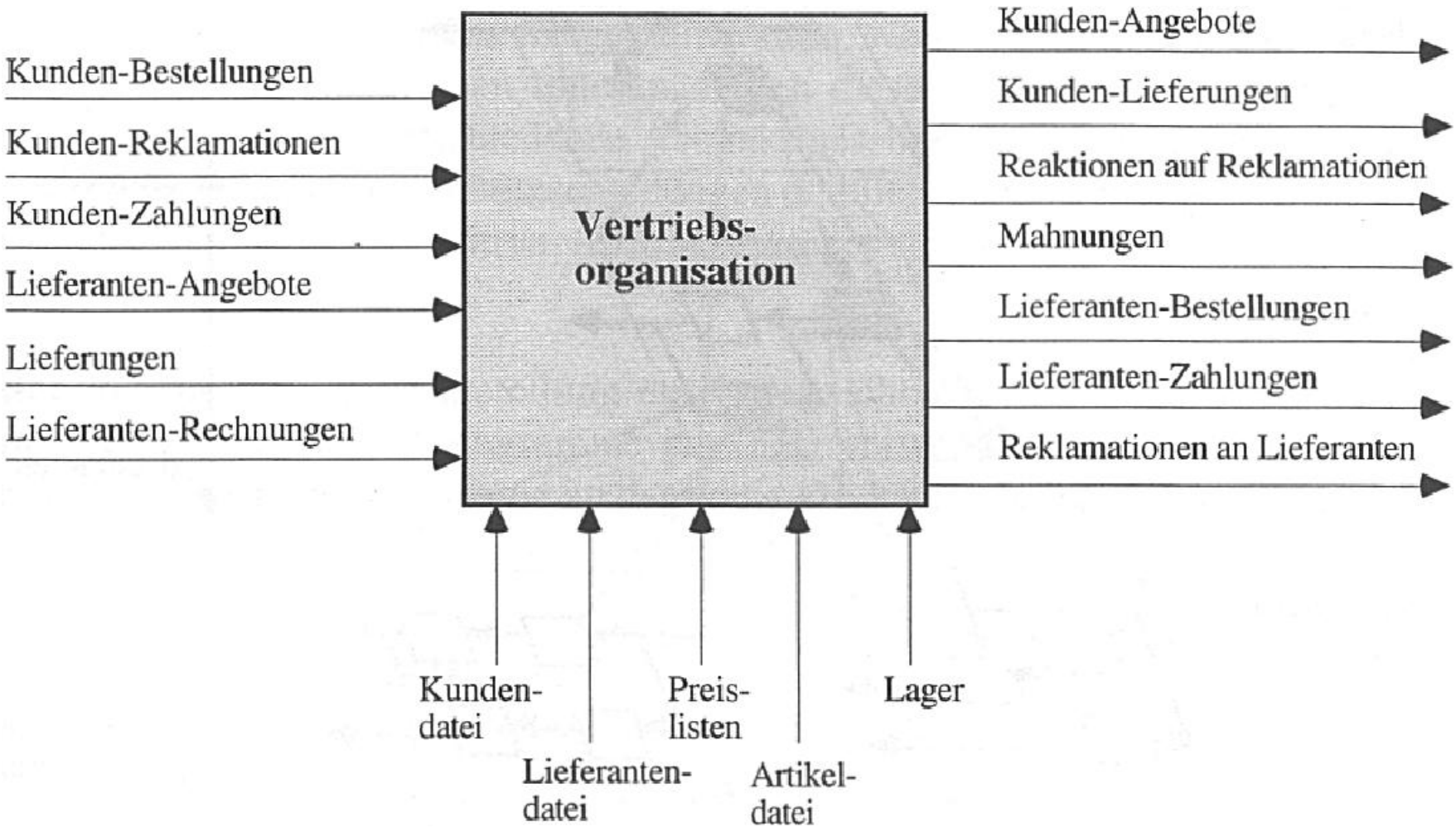
- SADT – Structured Analysis and Design Technique
  - graphische Sprache SA (structured Analysis) und
  - Methodik DT (Design Technique)
- Grundkonzept von SADT
  - Modellierung des Datenflusses ist im Vordergrund sowie Top-down- Vorgehensweise mit schrittweiser Verfeinerung
- SA – Structured Analysis
- ER – Entity Relationship
  - Diagramme zur Beschreibung von System- und Komponentenstrukturen
- SA/RT – Structured Anaysis / Real-Time
  - Zustandsautomaten zur Darstellung von Kontroll- und Steuerungsaspekten

Grundkonzept SADT

- Das System wird durch Dinge und Geschehen charakterisiert
  - Dinge:
    Passive Elemente wie Objekte, Daten oder Information
  - Geschehen:
    Aktive Elemente wie Operationen, Aktivitäten und Prozesse, die miteinander kommunizieren
- Aktivitätensichtweise
  - Aktivitäten (die durch Menschen, Maschinen, Institutionen, Rechner oder Algorithmen wahrgenommen werden) stehen im Vordergrund
- Datensichtweise
  - Daten, Objekte oder Gegenstände sind zentraler Aspekt
- Allgemeine Form der Bausteine:

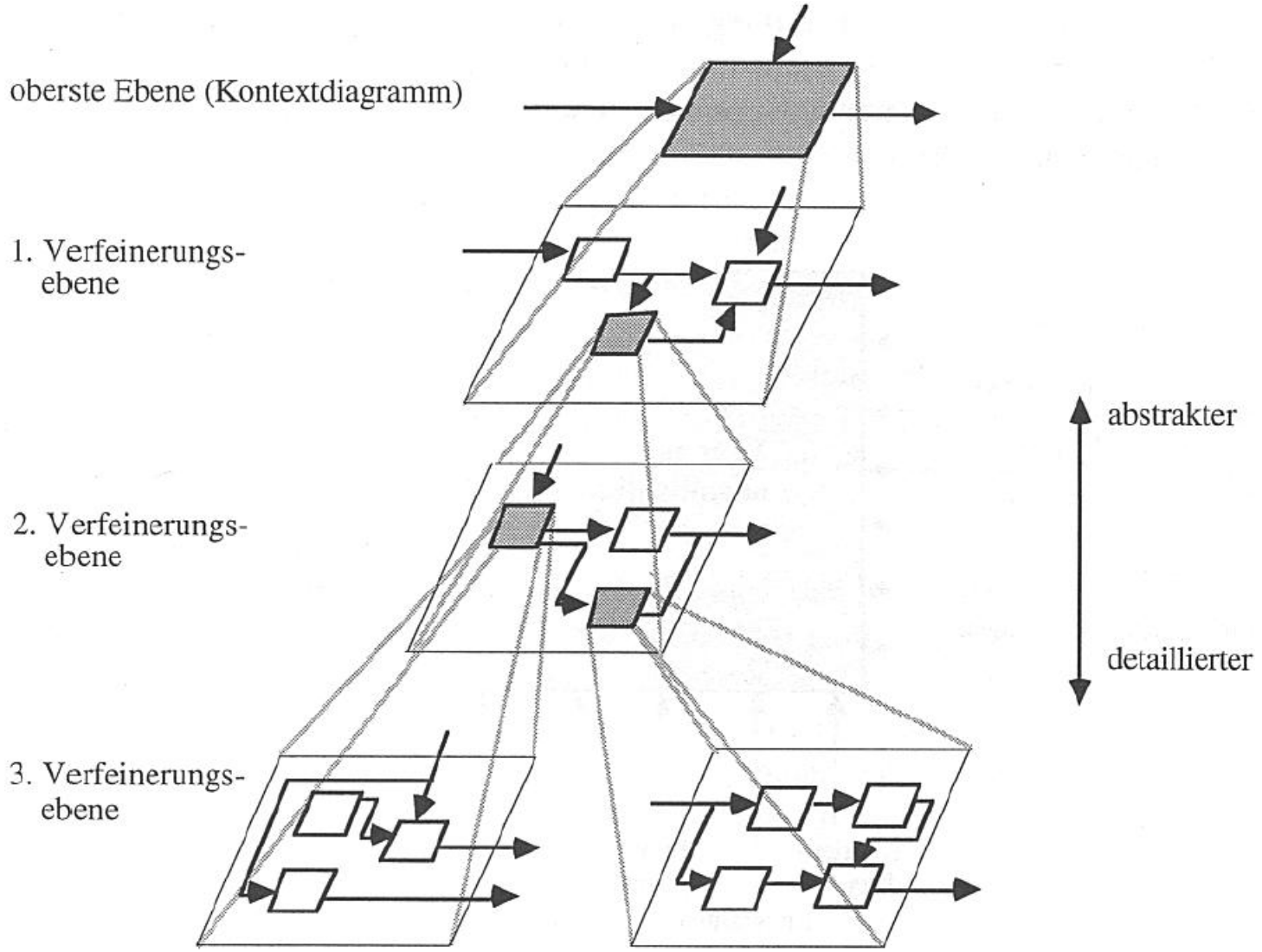


Aktivitätsdiagramm Vertriebsorganisation



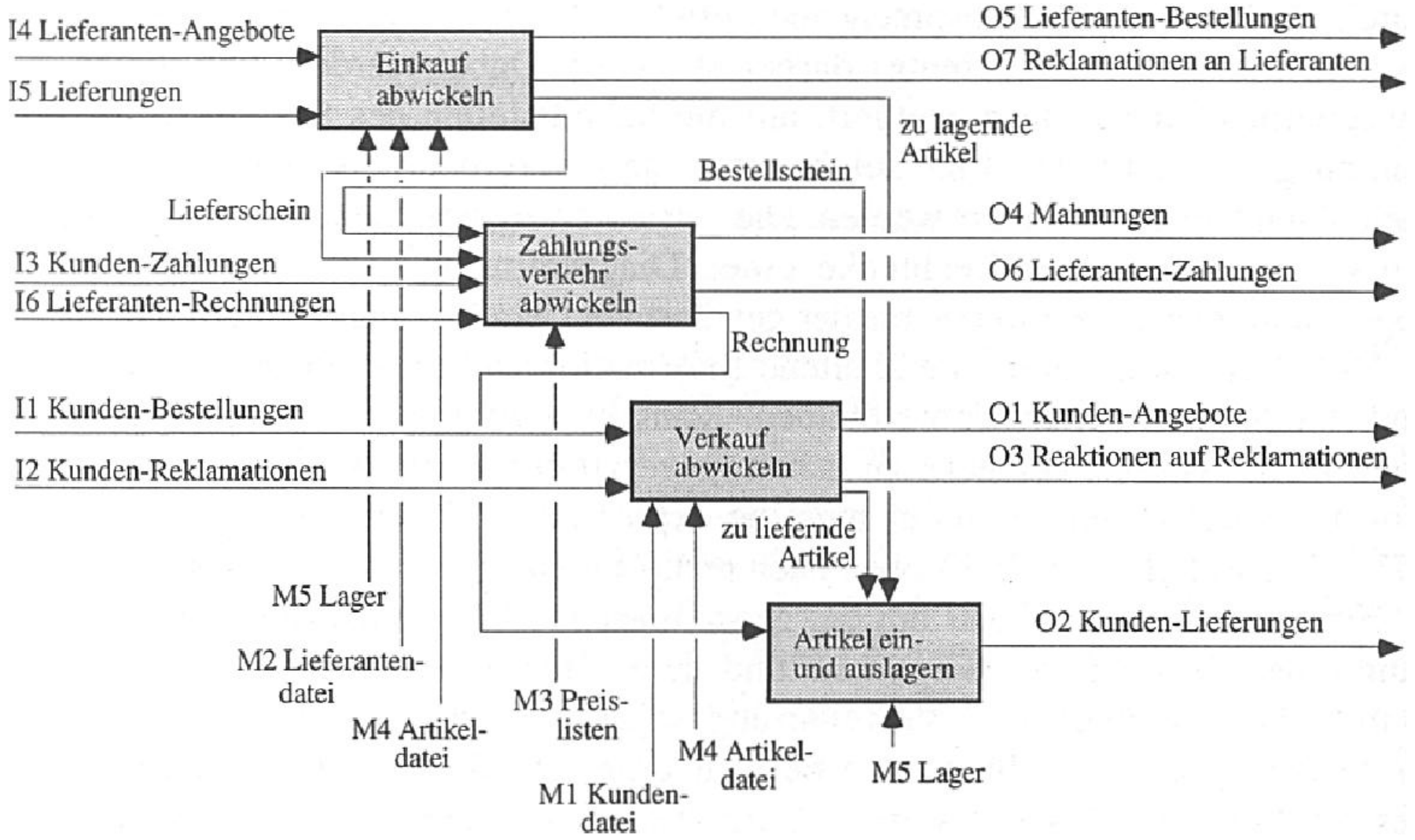
Partsch, 1988

Verfeinerungsprozess in SADT



Partsch, 1988

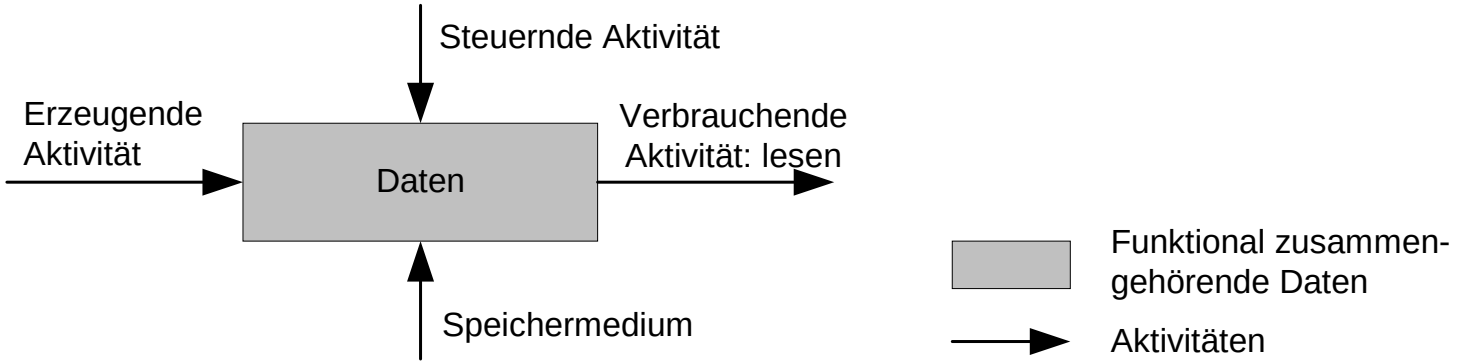
Aktivitätsdiagramm Vertriebsorganisation



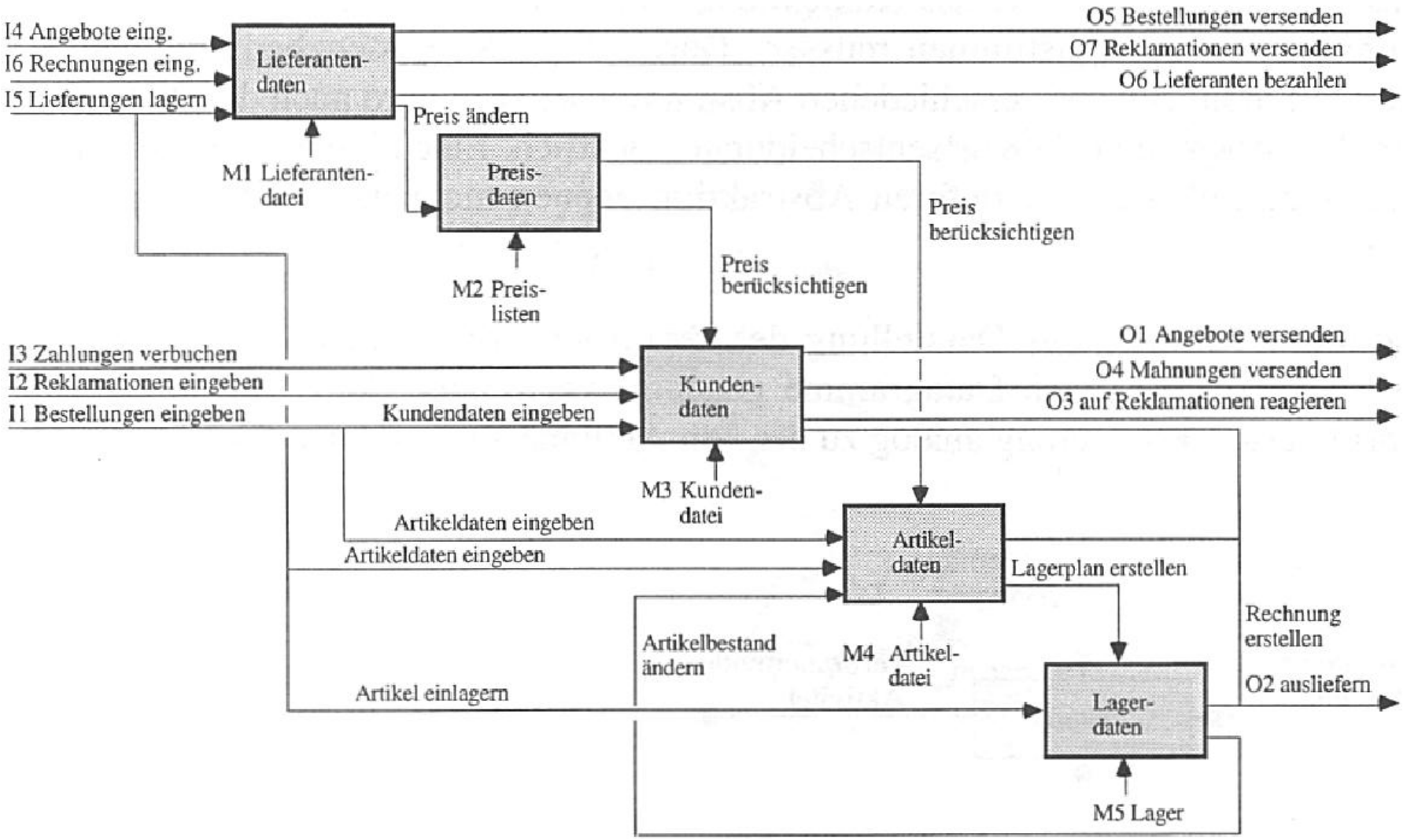
Partsch, 1988

Datendiagramme

- Zur Darstellung der Datensichtweise
  - Datendiagramme oder Datagramme
  - Schrittweise Verfeinerung
- Allgemeine Form der Bausteine eines Datendiagramms:



Datendiagramm Vertriebsorganisation



Partsch, 1988

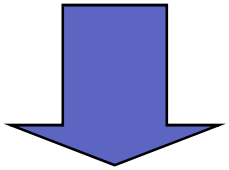
Aktivitäten- und Datendiagramme

- Aktivitäten- und Datensichtweise sind komplementär und ergänzend
- Verschiedene Detailinformation zur vollständigeren Beschreibung des Systems
- Wechselseitige Überprüfung auf Vollständigkeit und Konsistenz
- Aktivitätsdiagramme sind im wesentlichen Datenfluss-diagramme, sie erlauben keine Aussage über eine zeitliche Abfolge
- Datendiagramme sind eine spezielle Form von Strukturgraphen
- Formulare sind vorgesehen

SADT-Methodik

Erstellen einer Anforderungsdefinition

1. Aktivitätensichtweise mit Top- Down- Vorgehensweise entwickeln
2. Review
3. Datensichtweise mit Top- Down- Vorgehensweise
4. Vergleich Aktivitäten- und Datensichtweise
5. Modifikation falls notwendig
6. Aktivierungsfolgen falls notwendig anpassen
7. Eintragen auf Standard SADT- Formulare mit Knotenverzeichnis und Glossar als Anforderungsdokument



Entwurf
Systemspezifikation

Bewertung SADT

Vorteile	Nachteile
Einfach erlernbar wegen natürlicher Sprache und damit hoch flexibel	Semantische Ungenauigkeit wegen natürlicher Sprache
Graphische Beschreibung mit wenigen exakt festgelegten Bestandteilen	Keine formale Prüfung möglich
Manuell einsetzbar	Keine direkte methodische Anbindung an spätere Phasen
Unterstützt Top-Down-Zerlegung für klare Problemabgrenzung	Erstellung ohne Rechnerunterstützung aufwendig.
Vorgriffe auf Implementierung schwieriger	

Structured Analysis

- **Grundkonzept**
  - SA Structured Analysis
    Datenflussdiagramme, Datenlexikon und Prozessspezifikation.
  - SASS Structured Analysis and System Specification von Tom De Marco
- **Datenflussdiagramme (data flow diagram, bubble chart, Prozessmodell, Funktionsmodell):**
  - Ist ein endlicher, zusammenhängender knoten- und kantenmarkierter Graph.
    Er dient der Darstellung von Prozessen, Funktionen oder Aktionen (Knoten) und der Kommunikation zwischen Prozessen über Datenflüsse (Kanten)
- **Arten von Knoten:**

  - **Prozessknoten** (node, Prozess)
    Kreis, repräsentiert komplexe Aktionen oder Prozesse, die einen oder mehrere Eingangsdatenflüsse in einen oder mehrere Ausgangsdatenflüsse transformieren

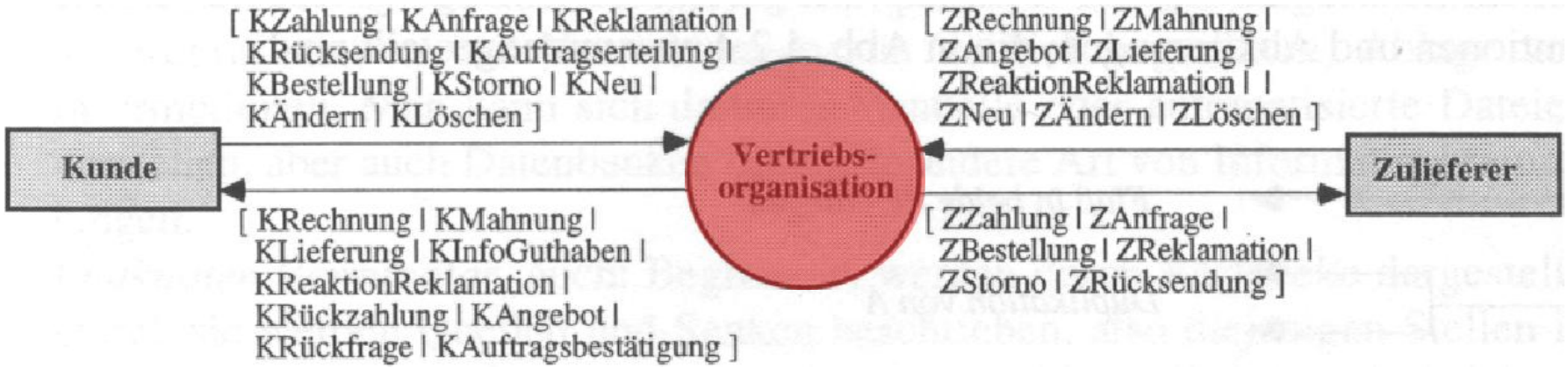
  - **Datenspeicher** (store, Lager)
    parallele Balken, dienen der Repräsentation von Vorrichtungen zur Ablage von Information (statischer oder abgelegter Daten)

  - **Endknoten** (terminator, Begrenzer)
    Rechteck, Quellen und Senken, Schnittstellen zur Umgebung. Black Boxes, da nicht Systembestandteil

Structured Analysis

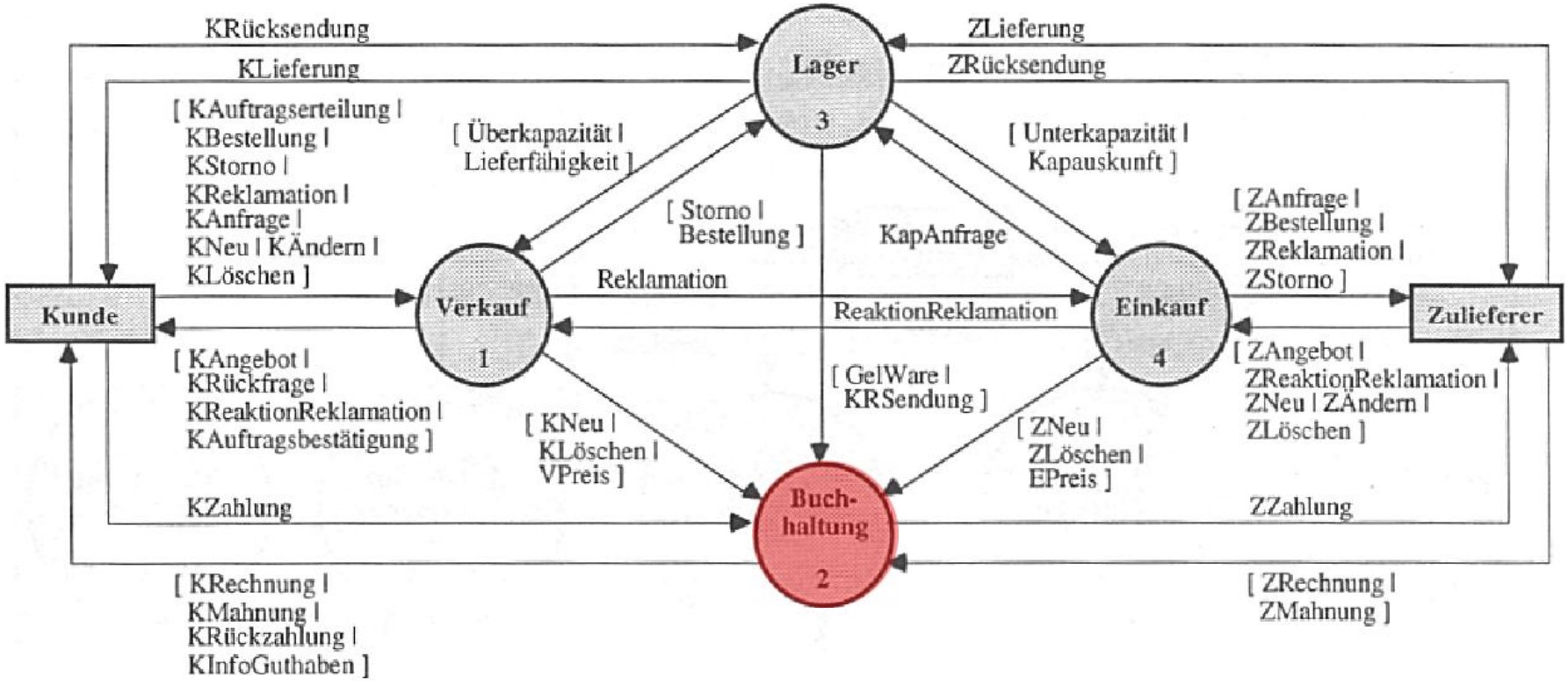
- Kanten im Graphen dienen der Darstellung des Datenflusses
  - beschreiben den Transport von Daten
  - haben immer eine Quelle und ein Ziel
  - sind stets benannt
    (mit der Bezeichnung der transportierten Daten)
- Regeln zur Benennung:
  - Datenspeicher – Substantiv im Plural
  - Begrenzer, Datenflüsse – Substantiv Singular
  - Prozess – Substantiv oder Verb- Objekt- Kombination

Datenflussdiagramm Vertriebsorganisation



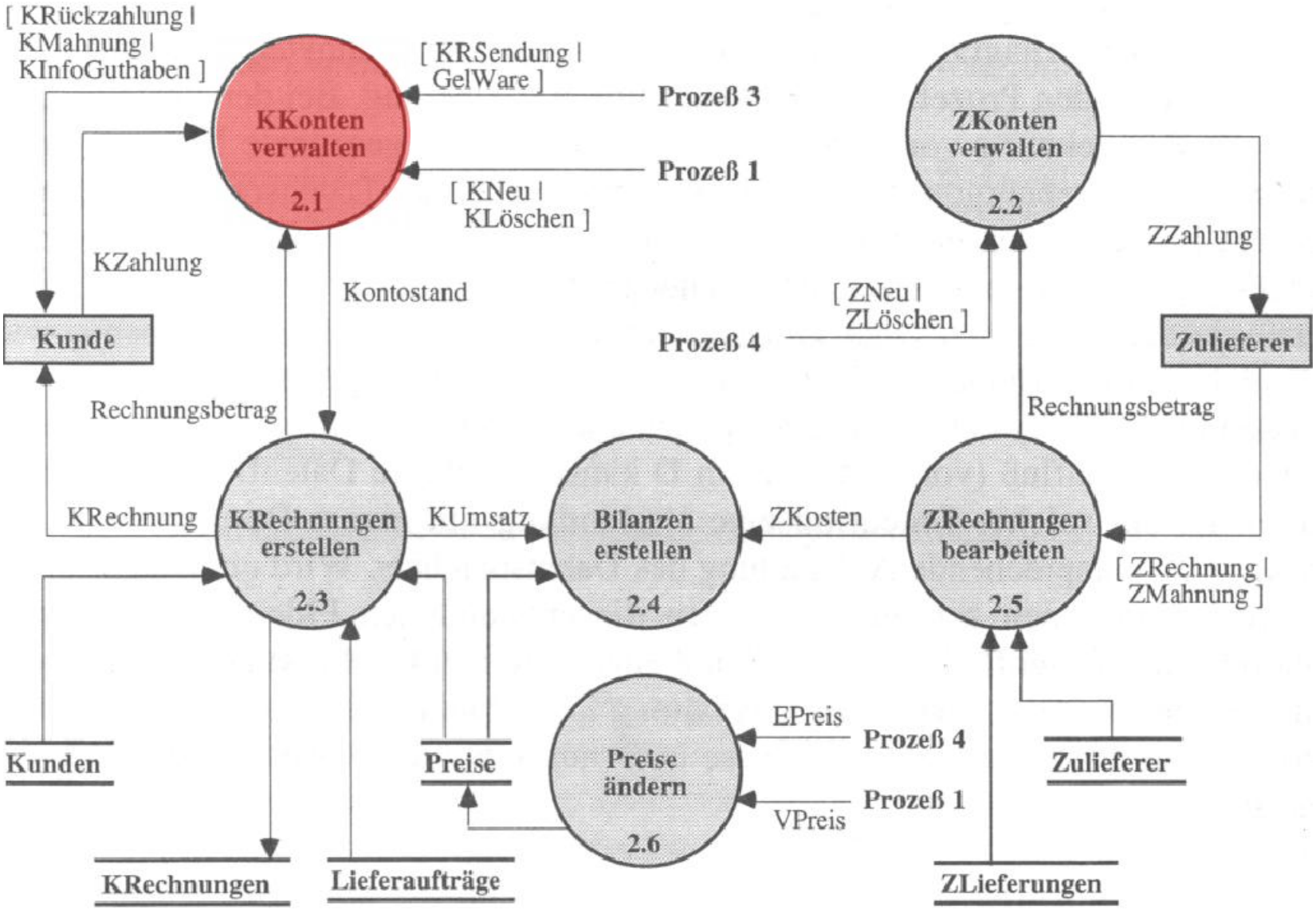
Partsch, 1988

Verfeinerung des Prozesses „Vertriebsorg.“



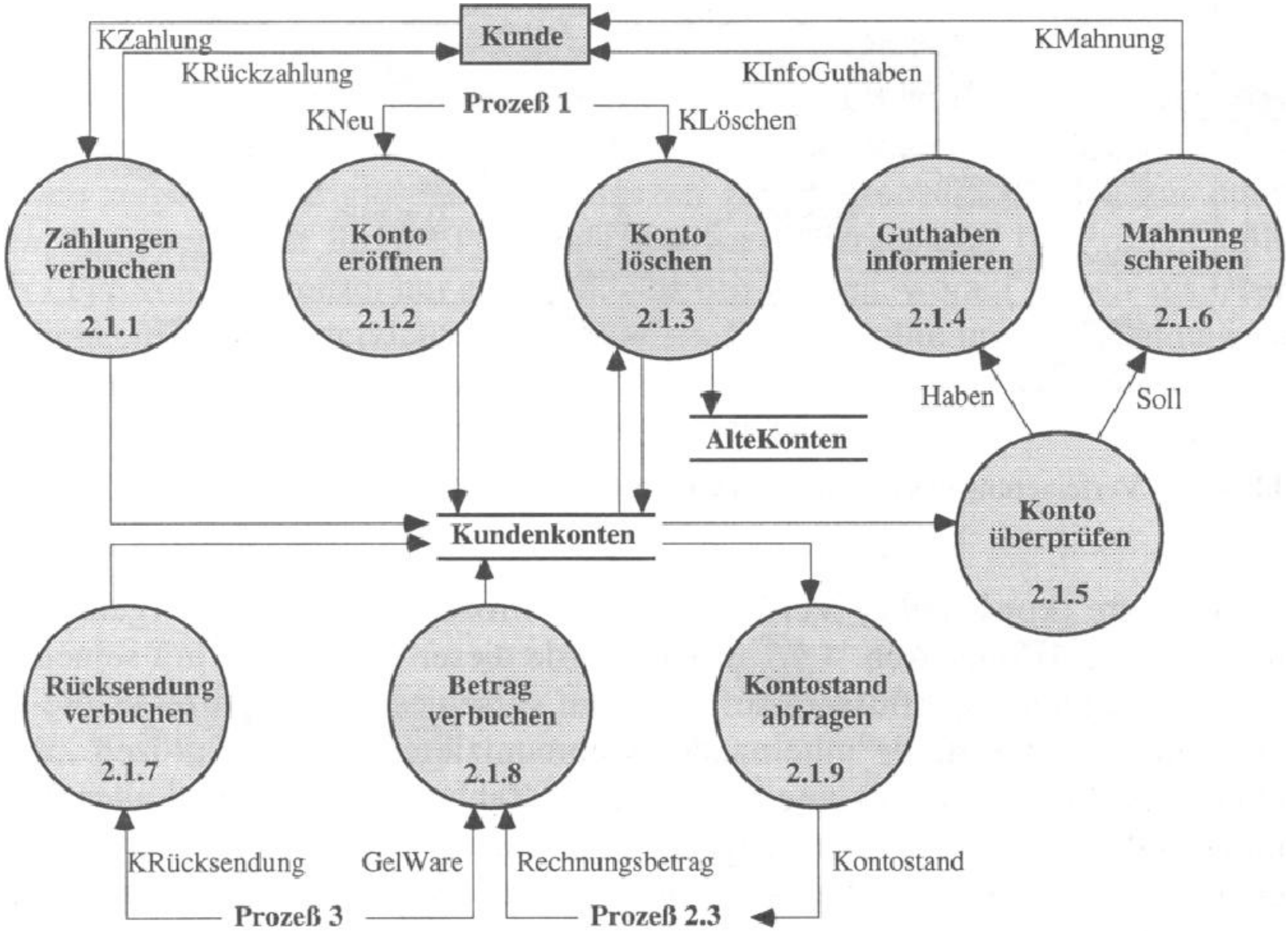
Partsch, 1988

Verfeinerung des Prozesses „Buchhaltung“



Partsch, 1988

Verfeinerung des Prozesses „Kontenverwalt.“



Partsch, 1988

Datenlexikon

- **Notation für reguläre Ausdrücke**

  = ist definiert als

  \+ Sequenz („ und“)

  | Auswahl („ oder“)

  [ ] Klammerung einer Auswahl

  { } beliebige Wiederholung

  { }n n-malige Wiederholung

  ( ) optionale Angabe

  @ Schlüssel

  ** Kommentar

Empfehlungen

- Eindeutige Namen, leicht verständlich
- Keine Redundanzen
- Je ein Eintag für
  - jeden Datenfluss,
  - jeden Datenspeicher und
  - jeden elementaren Prozess (aufgrund Konsistenz Datenflussdiagramm und Datenlexikon)

Datenlexikon Vertriebsorganisation

Adresse =	Straße + Hausnummer + (Länderkennung + -) + PLZ + Ort + (Land)
Anmerkungen =	{[Buchstabe | - | ' ']}^{100} ** Textfelder der Länge 100; ' ' symbolisiert das Leerzeichen ** Bemerkungen zum Zahlungsverhalten, Beschwerden u.ä.
Datum =	Tag + Monat + Jahr
Firma =	@Firmenname + Adresse
Firmenname =	Buchstabe + {[Buchstabe | - | . | & | ' ']}^{29}
Geburtsdatum =	Datum
Hausnummer =	[1 | 2 | ... | 999]
Jahr =	[1950 | 1951 | ... | 2000]
Kontostand =	[S | H] + {Ziffer}^{7} + . + Ziffer + Ziffer ** Soll oder Haben
Kunde =	[Person | Firma] + @Kundennummer + letzte Bestellung + Anmerkungen
Kundenkonto =	@Kundennummer + Kontostand
Kundennummer =	[1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9]^{8}
Länderkennung =	[D | F | NL | GB | AU | CH | | ...]
Land =	Name
letzte Bestellung =	Datum
Monat =	[Januar | Februar | ... | Dezember]
Nachname =	Name
Name =	Buchstabe + {[Buchstabe | – | ' ']}^{19}
Ort =	Name
Person =	@Nachname + @Vorname + Adresse + @Geburtsdatum
PLZ =	{Ziffer}^{5}
Straße =	Name
Tag =	[1 | 2 | ... | 31]
Vorname =	Name
:	

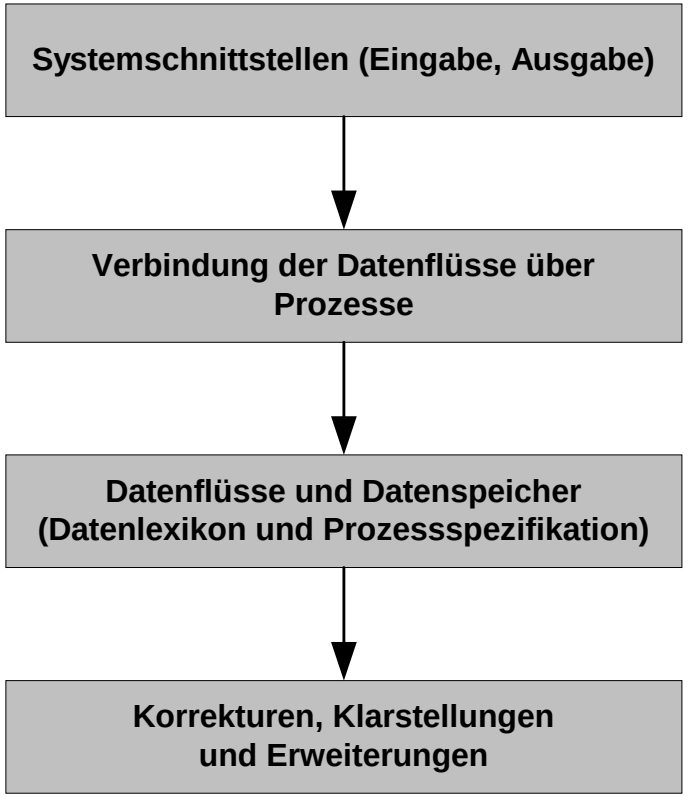
Partsch, 1988

Prozessspezifikationen

- Die Prozessspezifikation beschreibt, wie ein elementarer Prozess eingehende Datenflüsse in Ausgaben transformiert
  (ohne Vorwegnahme von Realisierungsaspekten)

- Beschreibungsformen:
  - Umgangssprache
  - Pseudocode
  - Graphen und Tabellen
  - Programmablaufpläne oder Struktogramme
  - Entscheidungstabellen

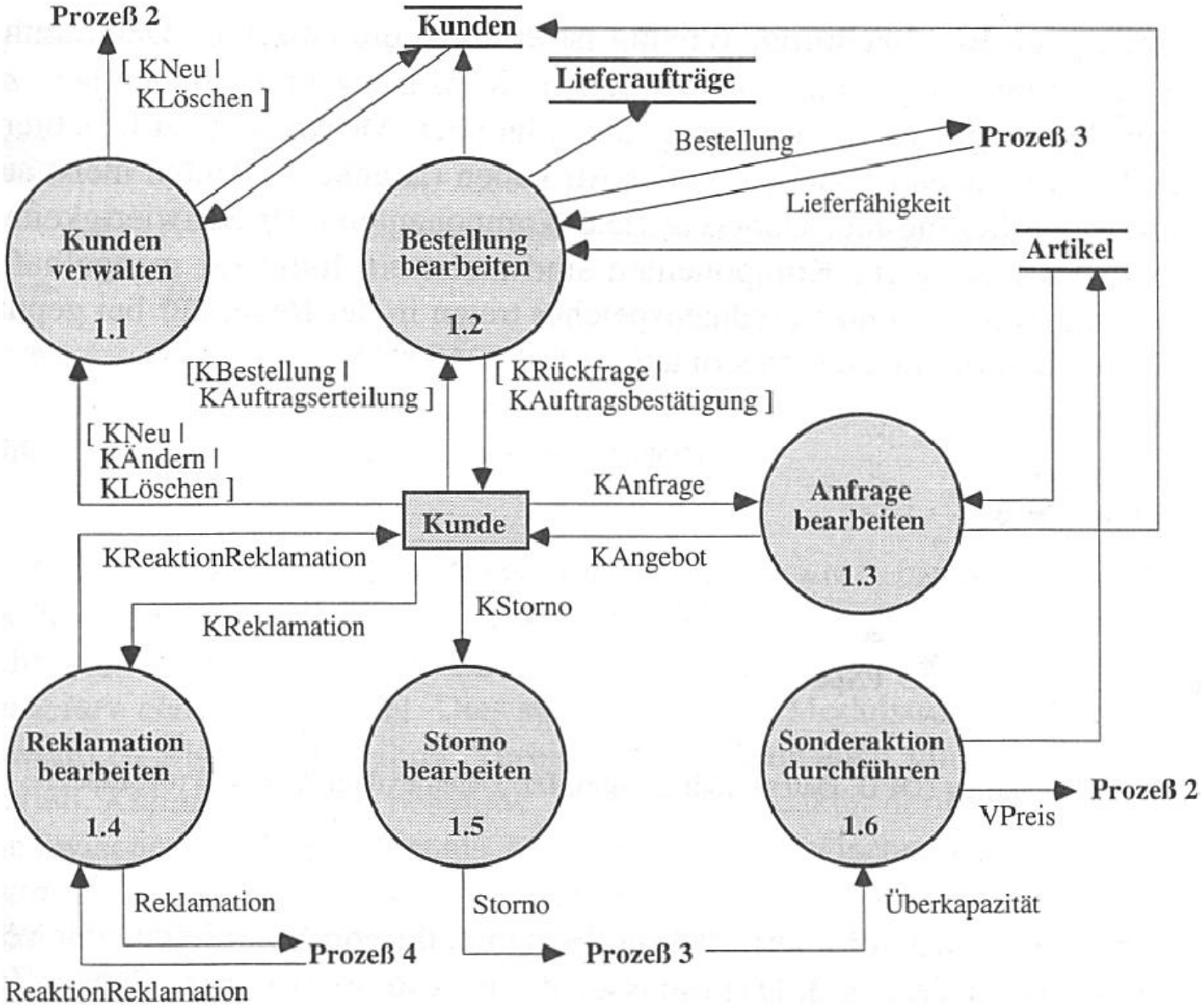
Methodik



- Systemanalytiker und Kunde
- Iterativer Verfeinerungsprozess

- Regel:
  - Nicht mehr als 8 Verfeinerungsebenen
  - Nicht mehr als 7 +/- 2 Prozesse pro Diagramm

Verfeinerung des Prozesses „Verkauf“

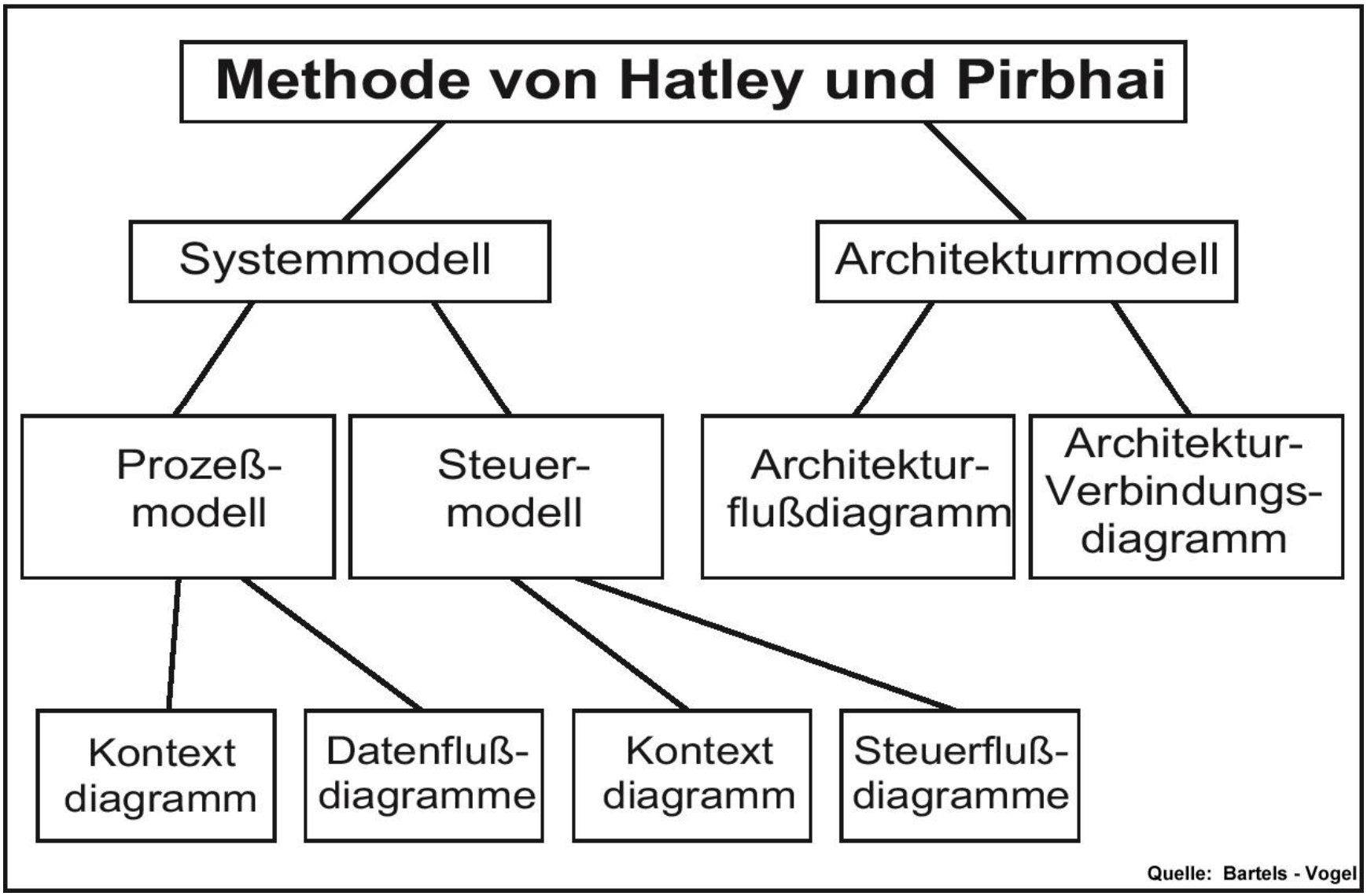


Partsch, 1988

Überprüfung der Beschreibung

1. Jeder Datenfluss und Datenspeicher, der in einem Diagramm vorkommt, muss im Datenlexikon definiert sein (und umgekehrt)
2. Jeder Prozess im Datenflussdiagramm (DFD) hat ein Verfeinerungsdiagramm oder eine Prozessspezifikation (aber nicht beides). Jeder Prozessspezifikation entspricht ein nicht weiter verfeinerter Prozess in einem Datenflussdiagramm. Ein- und Ausgabeflüsse (in DFD und Prozessspezifikation) müssen konsistent sein.
3. Jeder Datenlexikon-Eintrag muss referenziert sein (durch Prozessspezifikation, DFD oder Datenlexikon-Eintrag)
4. Für jeden Datenbezug in einer Prozessspezifikation muss eine der folgenden Bedingungen erfüllt sein:
   - Passt zu einem Datenfluss oder Datenspeicher, der mit demjenigen Prozess verbunden ist, mit dem die Prozesspezifikation assoziiert ist
   - Ist ein lokaler Term der Prozessspezifikation
   - Ist eine Komponente eines verbundenen Datenflusses oder Datenspeichers

Struktur der Systemanalyse nach Hatley und Pirbhai



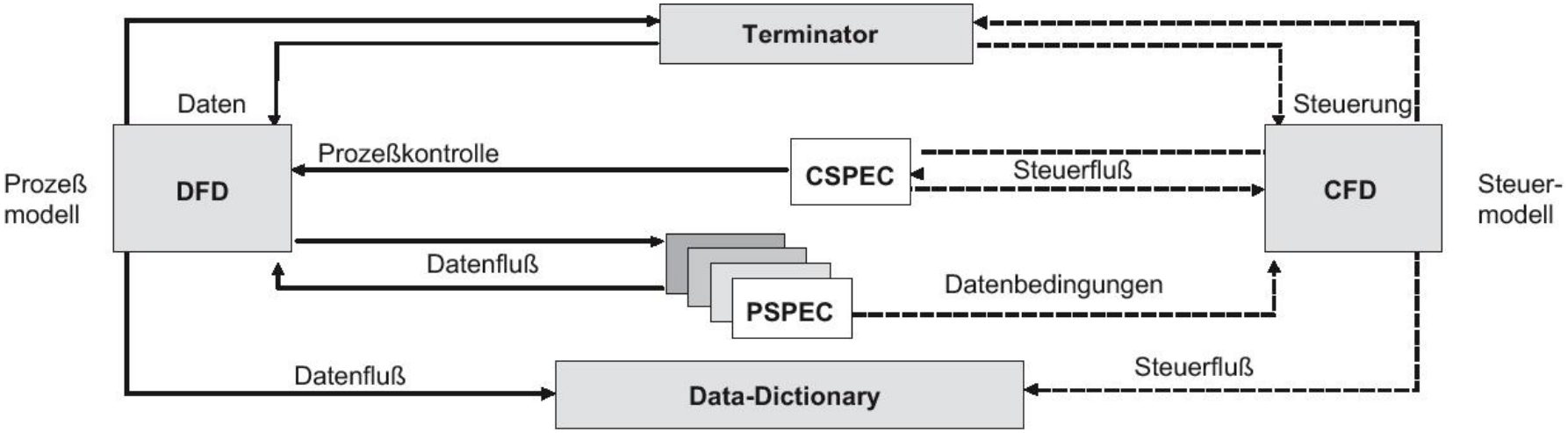
Verbale Aufgabenstellung „Presse“

Die verbale Funktionsbeschreibung lautet:

- ”Die Presse verfügt über mehrere Hydrauliksysteme. Diese Hydrauliksysteme bestehen aus einem Magnetventil zum Druckaufbau und einem Magnetventil zum Druckabbau. Als zugehörige Messwerte (Istwerte) stehen der Pressspalt (Istdistanz) sowie der Hydraulikdruck im jeweiligen System (Istdruck) zur Verfügung. Der Druck soll mit den Ventilen derart gesteuert werden, daß der Pressspalt eine vorgegebene Distanz nicht unterschreitet (Lageregelung). Dabei darf ein ebenfalls vorgegebener Maximaldruck keinesfalls überschritten werden (Druckregelung).”

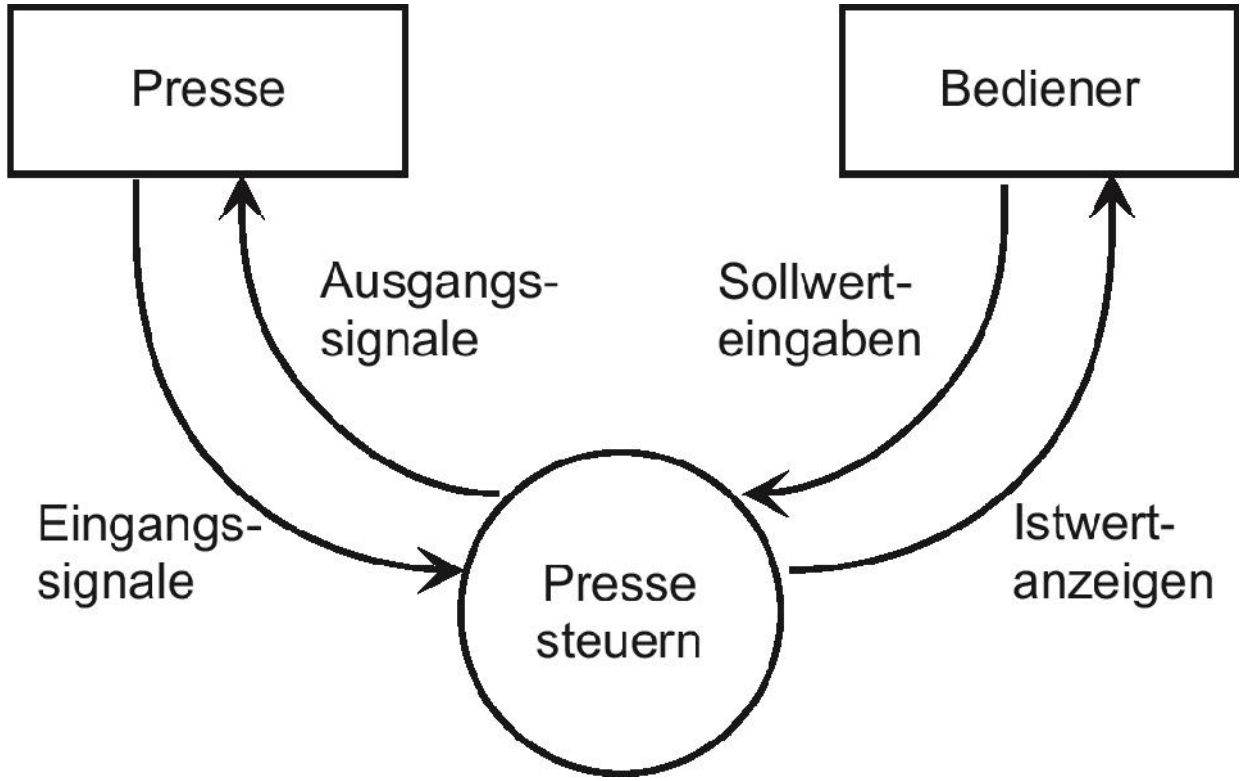
Systemmodell nach Hatley und Pirbhai

- Das Prozessmodell besteht aus den Datenflussdiagrammen (DFDs) und den Prozessspezifikationen (PSPECs)
- Das Steuermodell aus den Steuerflussdiagrammen (CFDs) und den Steuerspezifikationen (CSPECs).



Datenkontextdiagramm

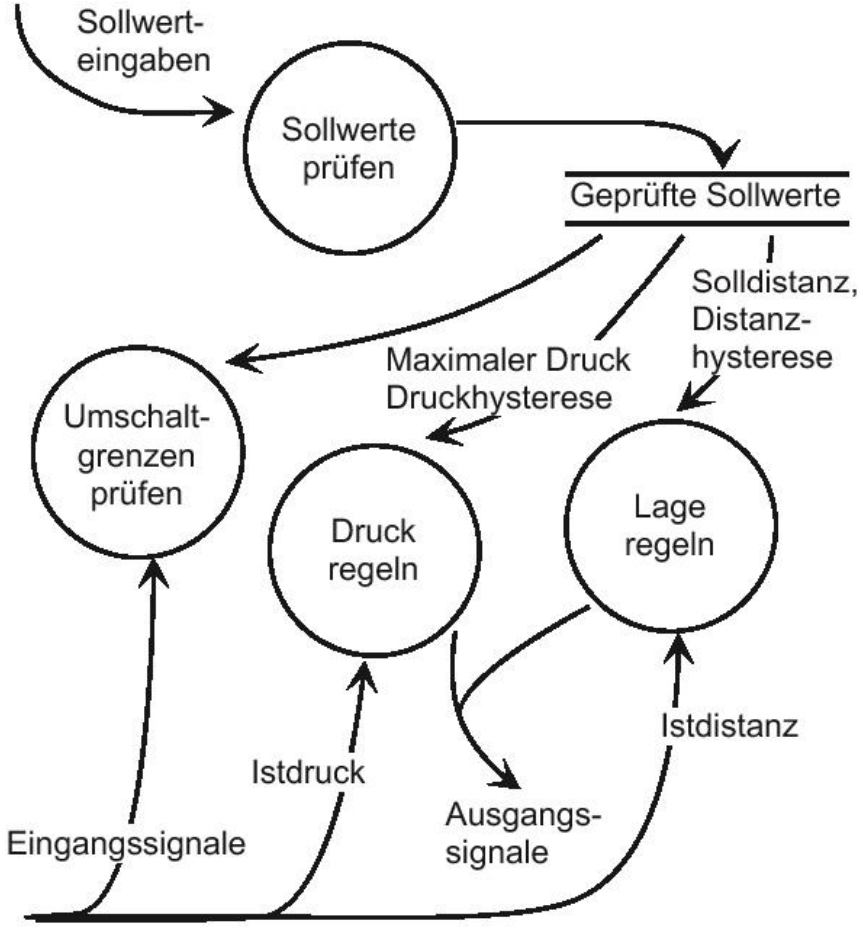
- Das Datenkontextdiagramm grenzt das Systemmodell zu den nicht modellierten Objekten (Terminatoren) ab



Quelle: Bartels-Vogel

Prozessmodell

- Das Prozessmodell wird mit hierarchischen Datenfluss-diagrammen und PSPECs beschrieben



PSPEC Lage regeln:

Drucksabbau := Istdistanz < Solldistanz – Distanzhysterese ;

Druckaufbau := Istdistanz > Solldistanz + Distanzhysterese ;

PSPEC Druck regeln:

Drucksabbau := Istdruck > Maxdruck –

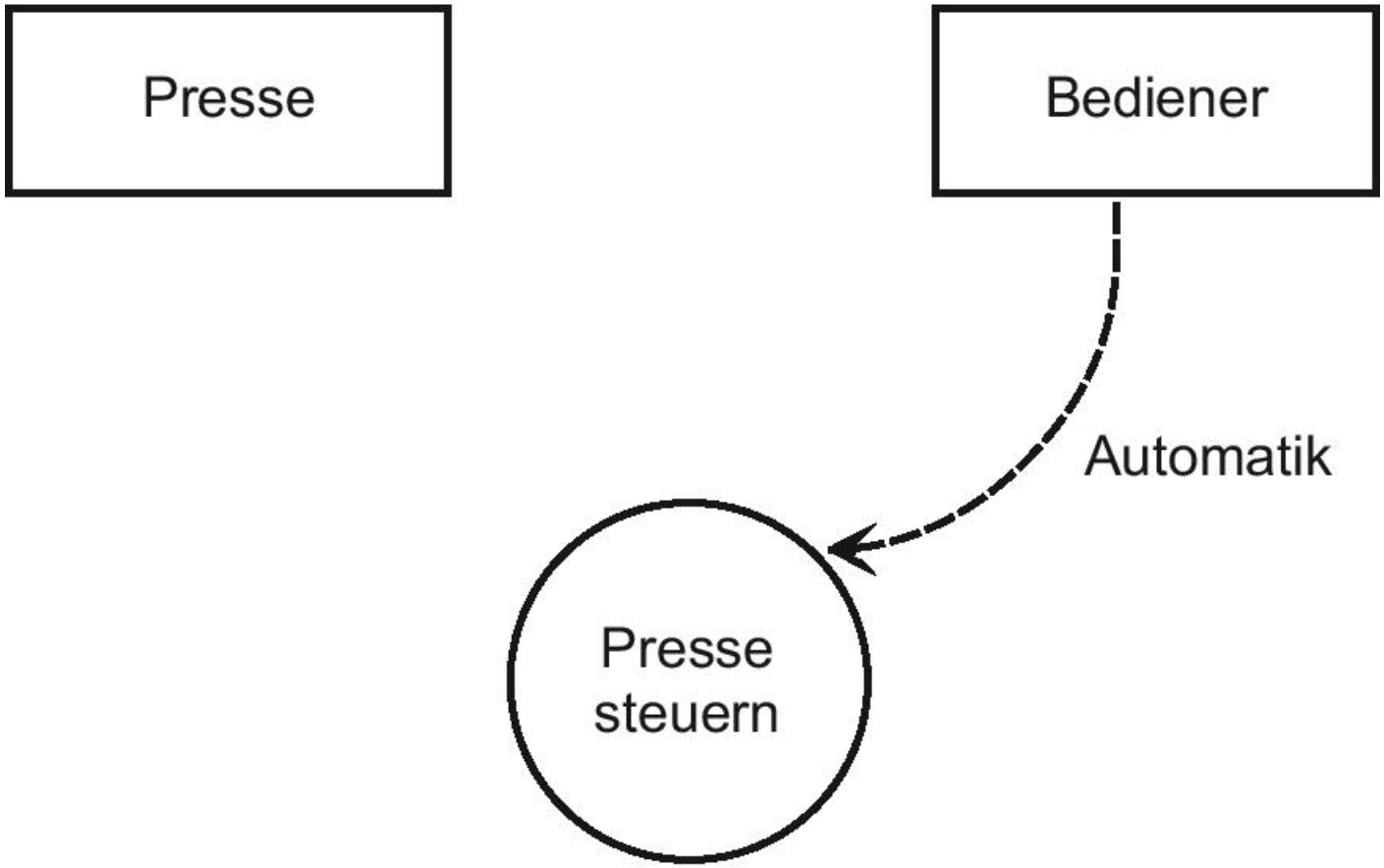
Druckhysterese ;

Druckaufbau := 0 ;

Quelle: Bartels-Vogel

Steuerkontextdiagramm

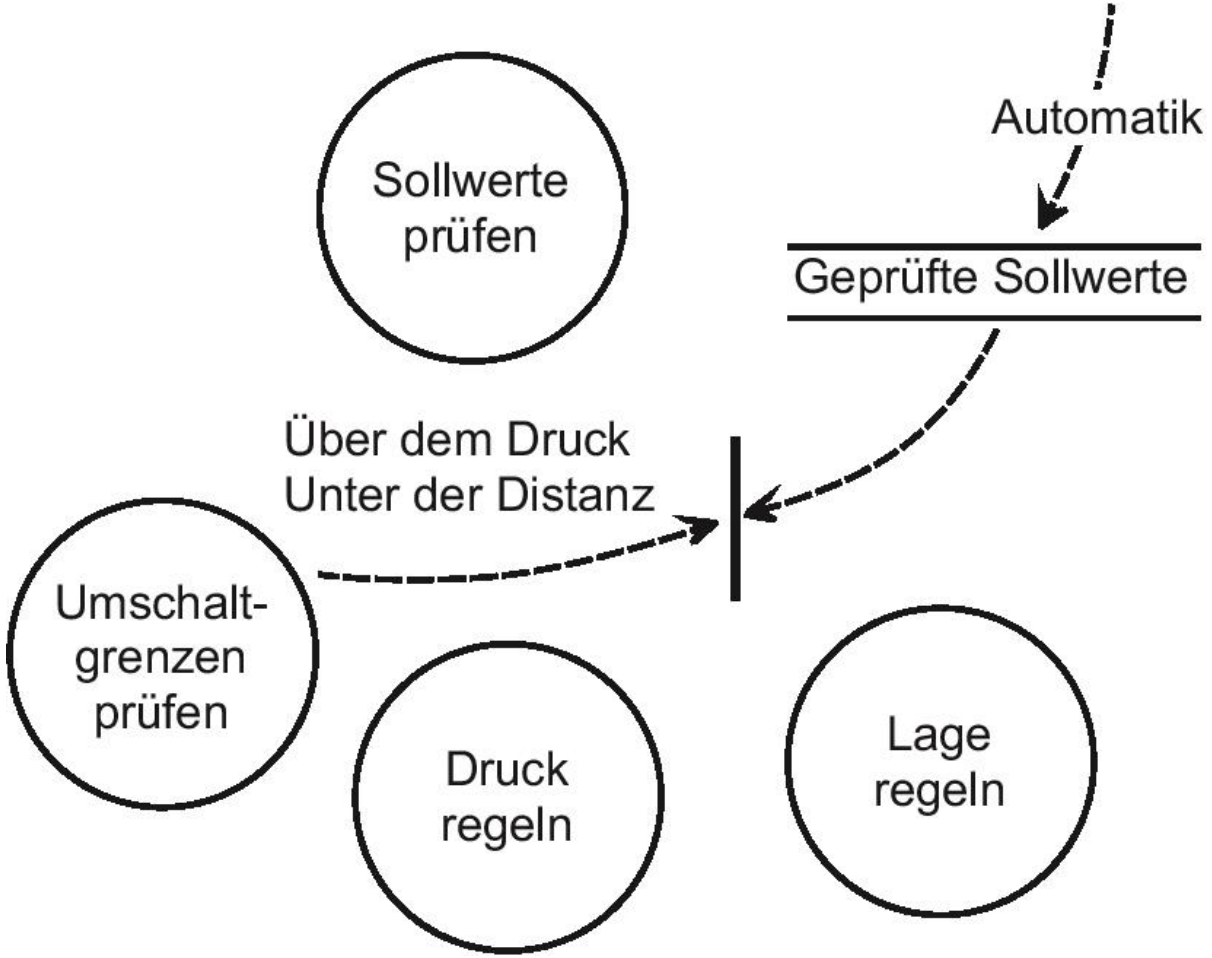
- Das Steuerkontextdiagramm enthält dieselben Terminatoren wie das Datenkontextdiagramm



Quelle: Bartels-Vogel

Steuerflussdiagramm

- Das Steuerflussdiagramm zeigt, welche Steuerflüsse verarbeitet werden



Quelle: Bartels-Vogel

Prozessspezifikation (PSPEC)

- Über sogenannte Daten- Bedingungen kann das Prozessmodell Steuerflüsse generieren.
- Die Beschreibung hierzu erfolgt im Prozessmodell als Prozessspezifikation (PSPEC).

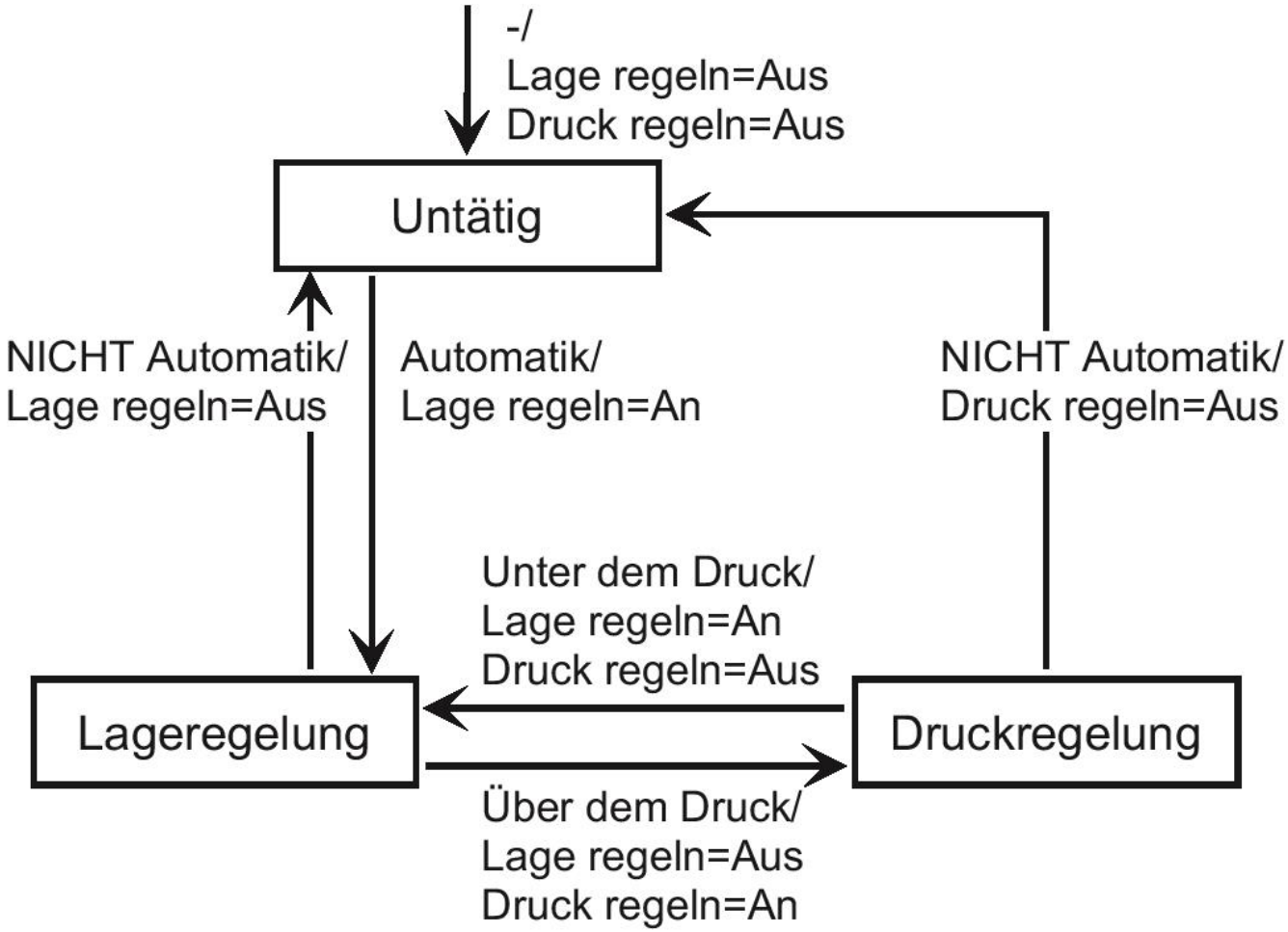
Über dem Druck := Istdruck > Maxdruck - Druckhysterese;
Unter dem Druck := Istdistanz < Solldistanz UND Istdruck < Maxdruck - Druckhysterese;

PSPEC "Umschaltgrenzen prüfen" zum Erzeugen der Datenbedingungen

Quelle: Bartels-Vogel

Steuerungsspezifikation (CSPEC)

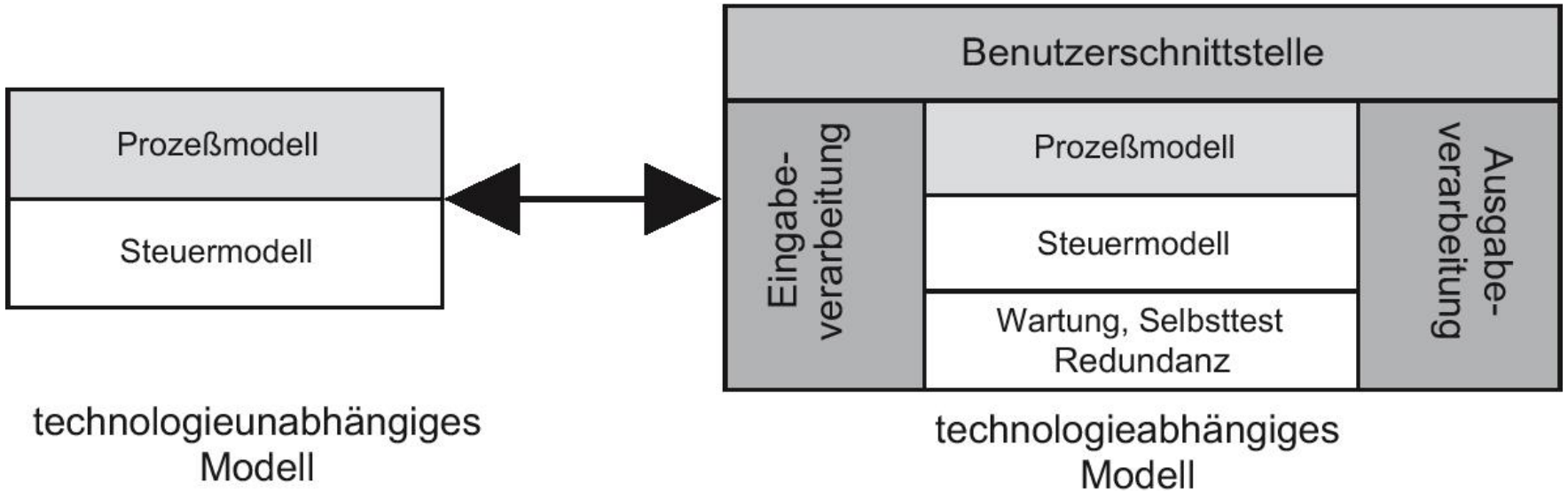
- Die Steuerspezifikation (CSPEC) ”Presse steuern” als Zustandsübergangsdiagramm



Quelle: Bartels-Vogel

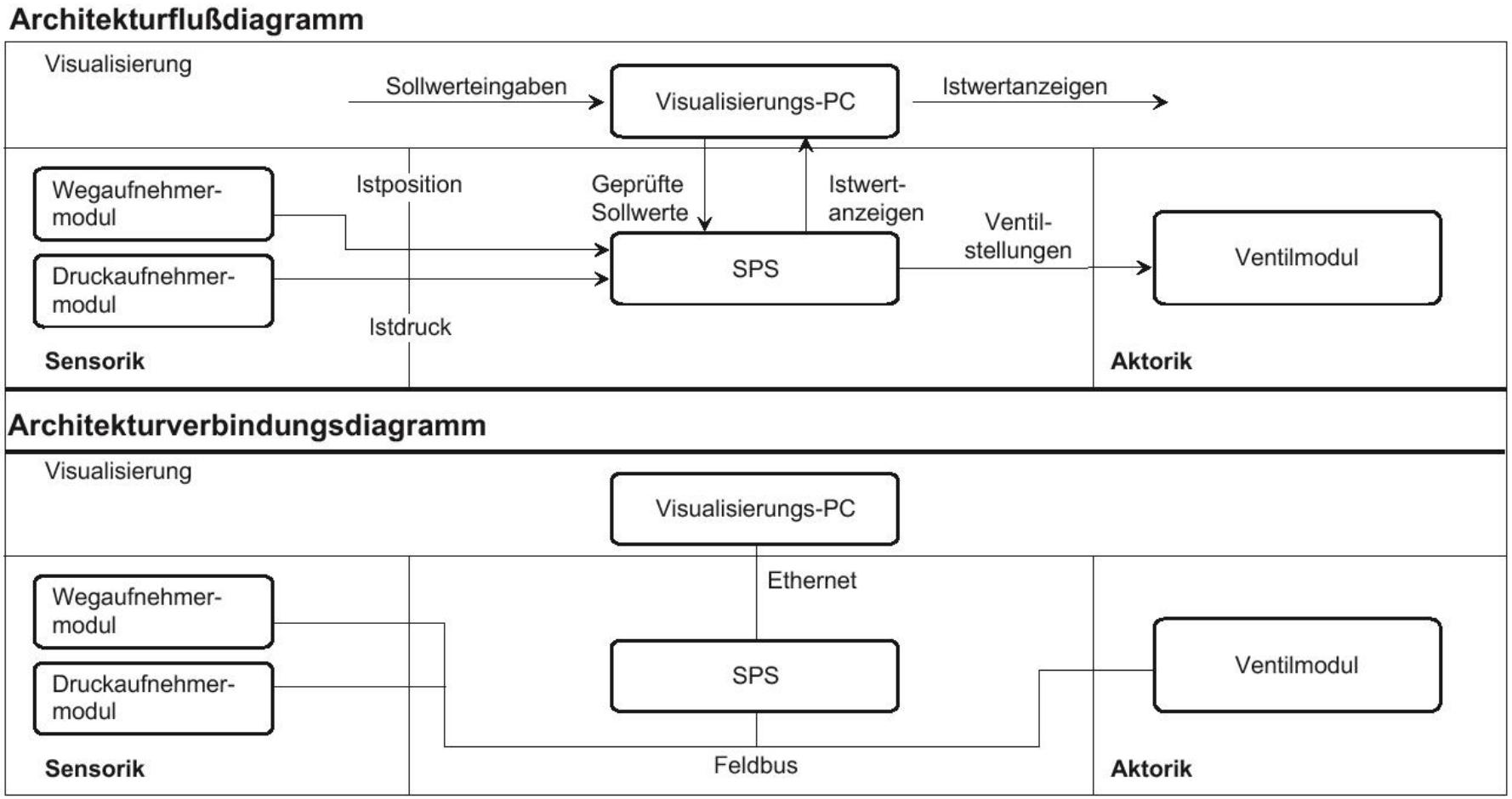
Architekturschablone

- Mit der Architekturschablone werden das Prozess- und Steuermodell um die Benutzerschnittstelle, die Eingabe- und - Ausgabeverarbeitung sowie um Wartungsaspekte ergänzt



Quelle: Bartels-Vogel

Architekturfluss- und Verbindungsdiagramm



Quelle: Bartels-Vogel

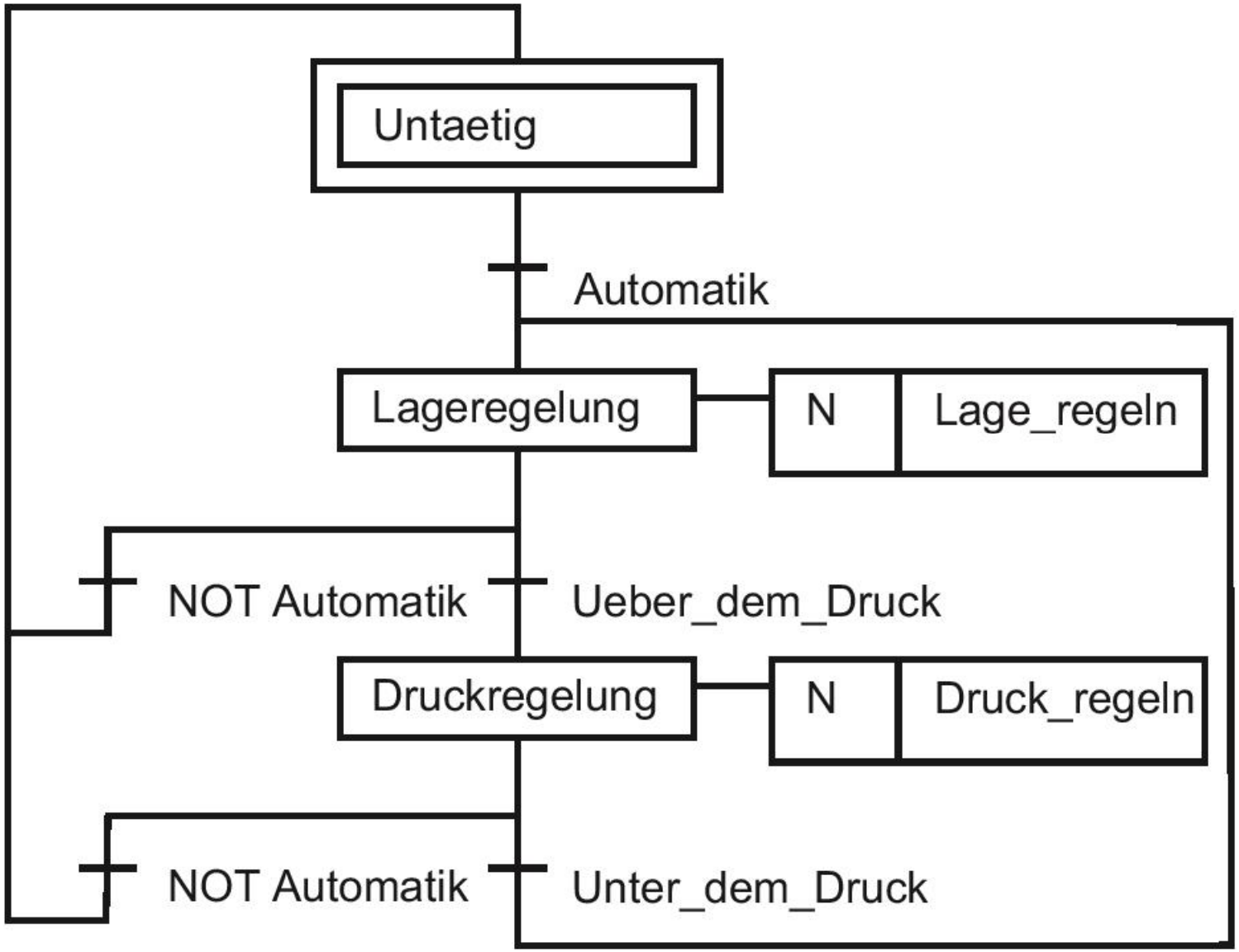
Portierbarkeit

- Im Hinblick auf eine einfache Portierbarkeit des Programmsystems auf verschiedene Automatisierungsumgebungen ist eine IEC 61131- 3- konforme Codierung der Module anzustreben, die als Programmorganisationseinheiten (POEs) in den IEC-Sprachen
  - Anweisungsliste (AWL),
  - Kontaktplan (KOP),
  - Funktionsbausteinsprache (FBS) oder
  - Strukturierter Text (ST) realisiert werden.
  - Die Automaten des Steuermodells werden in Ablaufsprache (AS) codiert.

  zu implementieren sind.
- Die Ablaufsprache besteht aus einer Menge von Schritten und Transitionen. Einem Schritt können beliebige Aktionen zugeordnet werden. Diese Aktionen werden abgearbeitet, wenn der zugeordnete Schritt aktiv ist (N- Aktionen). Der Schritt bleibt so lange aktiv, bis die folgende Transition erfüllt ist. Dieses Verhalten entspricht dem sequentieller Automaten. In AS können diese Automaten daher direkt abgebildet werden.
- Die Steuerflüsse aus dem Steuermodell sind als boolesche Bedingungen in AWL, KOP, FBS oder ST zu beschreiben, damit sie als Transitionen in AS eingesetzt werden können.
- Die Prozesse des Prozessmodells werden als Aktionen formuliert.

Quelle: Bartels-Vogel

Umsetzung der CSPEC



Quelle: Bartels-Vogel