

# **Grundlagen der Wirtschaftsinformatik**

Dresden, Sommersemester 2023

Prof. Dr. Torsten Munkelt

## Organisatorisches zur Lehrveranstaltung

Modulnummer	I-410
Studiengang	Bachelor/Diplom Wirtschaftsinformatik
Fachsemester	zweites
Vorlesung	zwei Semesterwochenstunden
Seminar	eine Semesterwochenstunde (zwei alle zwei Wochen)
Prüfung	schriftliche Klausur, 90 Minuten, mit Unterlagen
Lehrender	Prof. Dr. Torsten Munkelt
Telefon	+49 351 462 – 26 50
E-Mail	<a href="mailto:Torsten.Munkelt@HTW-Dresden.De">Torsten.Munkelt@HTW-Dresden.De</a>
Büro	Z 829
Sprechzeit	nach Vereinbarung

## GdWI in OPAL

The screenshot shows the OPAL - Online-Plattform interface. At the top, there is a menu bar with links to Datei, Bearbeiten, Ansicht, Chronik, Lesezeichen, Extras, and Hilfe. Below the menu is a browser-like header with a back button, a search bar containing 'Suchen', and a user profile for 'Torsten Munkelt'. The main content area has a teal header bar with buttons for Startseite, Lehren & Lernen, and Kursangebote. The 'Kursangebote' button is highlighted with a red rectangle. The page title is 'Wirtschaftsinformatik'. A navigation bar at the bottom includes links for 'OPAL / Hochschule für Technik und Wirtschaft Dresden / Fakultät Informatik/Mathematik / Wirtschaftsinformatik' and other site navigation icons. The main content area displays course cards. One card for 'Grundlagen der Wirtschaftsinformatik' is highlighted with a red rectangle. The card includes the course name, responsible teacher 'Torsten Munkelt', last viewed date '22.03.2017 um 10:04 Uhr', and view count '3925'. Another card for 'I-470: Allgemeine Betriebswirtschaftslehre' is partially visible below it.

OPAL - Online-Plattform f... + X

https://bildungssportal.sachsen.de/opal/auth | C Suchen X Torsten Munkelt ? E

Startseite Lehren & Lernen Kursangebote

Wirtschaftsinformatik

OPAL / Hochschule für Technik und Wirtschaft Dresden / Fakultät Informatik/Mathematik / Wirtschaftsinformatik

Geschaftsprozessmodellierung/Business Process Management  
Verantwortliche(r): Dirk Reichelt, Robert Ringel, Zuletzt angesehen: 22.03.2017 um 10:10 Uhr, Aufrufe: 7970

Grundlagen der Wirtschaftsinformatik  
Verantwortliche(r): Torsten Munkelt, Zuletzt angesehen: 22.03.2017 um 10:04 Uhr, Aufrufe: 3925

I-470: Allgemeine Betriebswirtschaftslehre  
Verantwortliche(r): Wolf-Eckart Gruening, Zuletzt angesehen: 21.03.2017 um 18:40 Uhr, Aufrufe: 4459

# Einschreiben für GdWI

The screenshot shows a web browser window for the OPAL Online-Plattform. The URL is https://bildungspotrait.sachsen.de/opal/auth/RepositoryEntry. The page title is "Grundlagen der Wirtschaftsinformatik". The left sidebar shows navigation links: "Grundlagen der Wirtschaftsinformatik" (with "Einschreibung" highlighted by a red box), "Vorlesungsunterlagen", and "E-Mail". Below that is "Gruppen / Teilnehmer" and "Teilnehmer GdWI". The main content area has a section titled "Einschreibung" with the subtext "Kursbaustein vom Typ Einschreibung". A table lists course information: Status (dropdown), Name (dropdown), Beschreibung (dropdown), Anzahl Plätze (dropdown), and Austragen Plätze (dropdown). A row for "Teilnehmer GdWI" shows the "Einschreiben" button (also highlighted by a red box) and a status message "0 / ∞ Nicht erlaubt". At the bottom, there is a pagination indicator "1 Eintrag" and page numbers "« 1 »".

# Herunterladen von Unterlagen zu GdWI

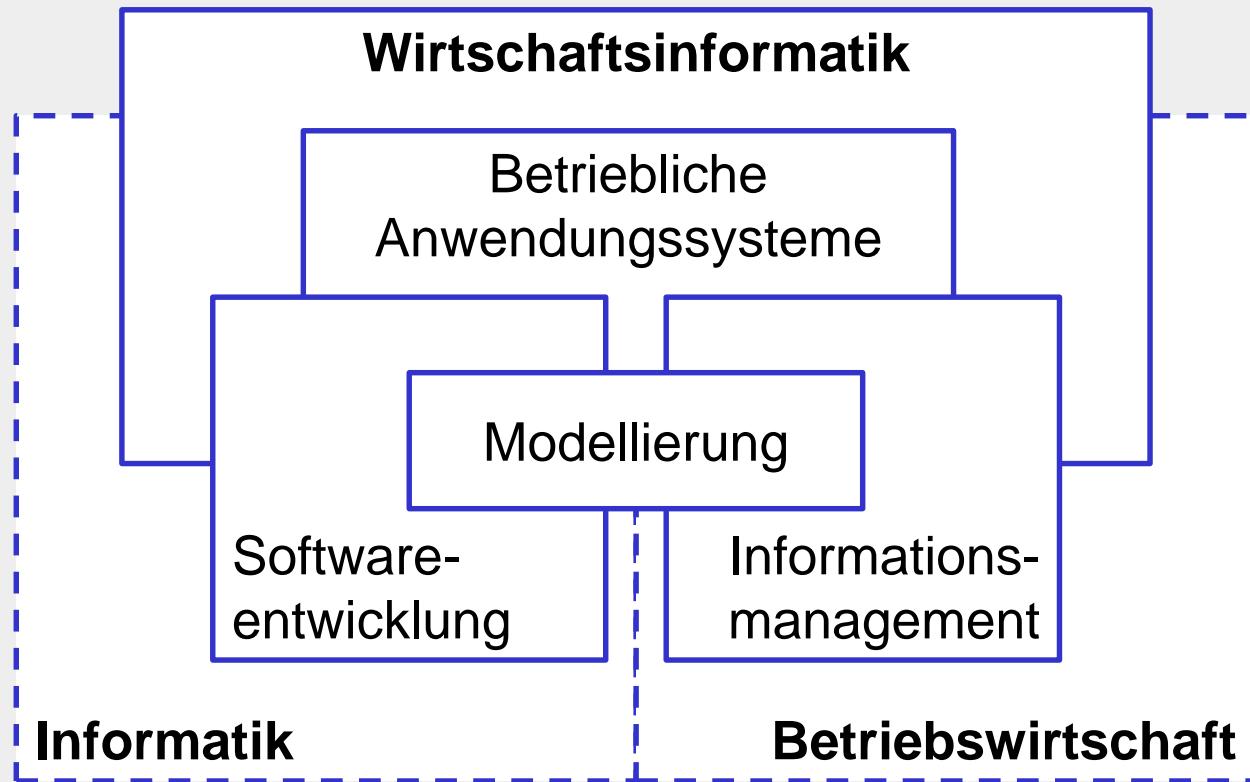
The screenshot shows a web browser window for the OPAL Online-Plattform. The URL is https://bildungspotrait.sachsen.de/opal/auth/RepositoryEntry/3131965446/C... . The page title is "Grundlagen der Wirtschaftsinformatik". The left sidebar shows navigation links: "Grundlagen der Wirtschaftsinformatik" (highlighted with a red box), "Einschreibung", "Vorlesungsunterlagen" (highlighted with a red box), and "E-Mail". Below these are "Gruppen / Teilnehmer" and "Teilnehmer GdWI". The main content area shows a section titled "Vorlesungsunterlagen" with a "Abonnieren" button. A note states: "Sie dürfen Dateien herunterladen, Dateien hochladen und Dateien löschen." It also says: "Sie können unter folgenden Bedingungen Dateien herunterladen:" followed by a bullet point: "Mitglieder der Gruppe: "Teilnehmer GdWI"". At the bottom, there is a table listing a file: "GdWI\_WInf\_SoSe2017\_01.pdf" (639K, 22.03.2017 um 10:42 Uhr). The "Aktionen" column for this file is also highlighted with a red box.

Dateityp	Name	Größe	Geändert am	Aktionen
	GdWI_WInf_SoSe2017_01.pdf	639K	22.03.2017 um 10:42 Uhr	

## Einordnung der Wirtschaftsinformatik

- „.... ein interdisziplinäres wissenschaftliches Arbeitsgebiet, das zwischen Betriebswirtschaftslehre und Informatik angesiedelt ist und auch die der Informationsverarbeitung zugrundeliegende Technik einbezieht.“ [Stahlknecht]
- „Durch ihre Interdisziplinarität hat sie [(die Wirtschaftsinformatik)] ihre Wurzeln in den Wirtschaftswissenschaften, insbesondere [in] der Betriebswirtschaftslehre[,] und [in] der Informatik.“ [Wikipedia]

## Einordnung der Wirtschaftsinformatik – grafisch



[Fink]

## Weitere zur Wirtschaftsinformatik beitragende Fachgebiete

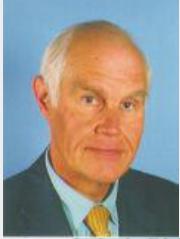
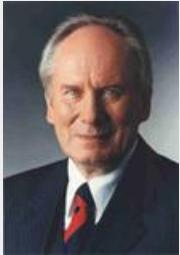
- Arbeitswissenschaften
- Ergonomie
- Psychologie
- Mathematik
- Operations-Research
- Statistik
- Jura
- ...

## Gliederung

---

- 1. Inhalte und Aufgaben der Wirtschaftsinformatik**
2. Grundlagen der Informatik und der Informationstechnik
3. Informationsmanagement
4. Modellierung
5. Datenbanken
6. Softwareentwicklung
7. Betriebliche Informationssysteme

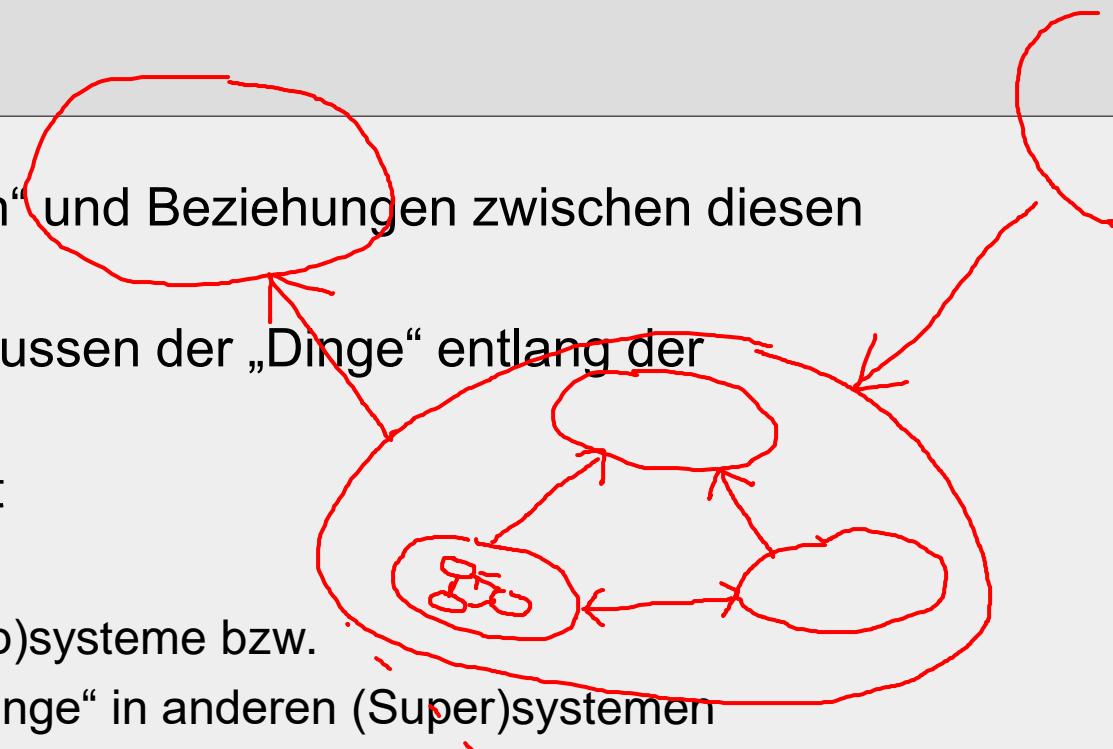
## Gegenstand der Wirtschaftsinformatik



- „[...] Wissenschaft von Entwurf, Entwicklung und Einsatz computergestützter **betriebswirtschaftlicher Informationssysteme**.“ [Scheer]
- „[...] befasst sich mit der Konzeption, Entwicklung, Einführung, Wartung und Nutzung von **Systemen der computergestützten Informationsverarbeitung im Betrieb**.“ [Mertens]
- „[...] Wissenschaft, die sich mit der Gestaltung **rechnergestützter Informationssysteme in der Wirtschaft** befasst.“ [Hansen]
- „[...] ist die Wissenschaft von [Konzeption,] Entwurf, Entwicklung[, Anpassung, Konfiguration (Customizing)] und Anwendung von **Informations- und Kommunikationssystemen [...] in Wirtschaftsunternehmen**.“ [Wikipedia]
- „[...] hat **Informations- und Kommunikationssysteme in Wirtschaft und Verwaltung** zum Forschungsgegenstand.“ [Alpar]

## System - informal

- Gesamtheit von „Dingen“ und Beziehungen zwischen diesen Dingen
- Wechselseitiges Beeinflussen der „Dinge“ entlang der Beziehungen
- Abgrenzbar von Umwelt
- Rekursion möglich:
  - „Dinge“ wiederum (Sub)systeme bzw.
  - Systeme wiederum „Dinge“ in anderen (Super)systemen
- Beziehungen zur Umwelt?
  - Ja → offenes System
  - Nein → geschlossenes System
- Veränderungen über der Zeit?
  - Ja → dynamisches System
  - Nein → statisches System

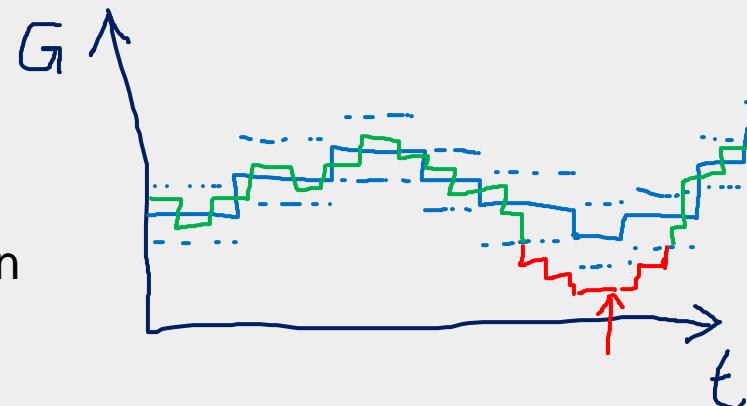


## Betriebliche Informationssysteme (BISe)

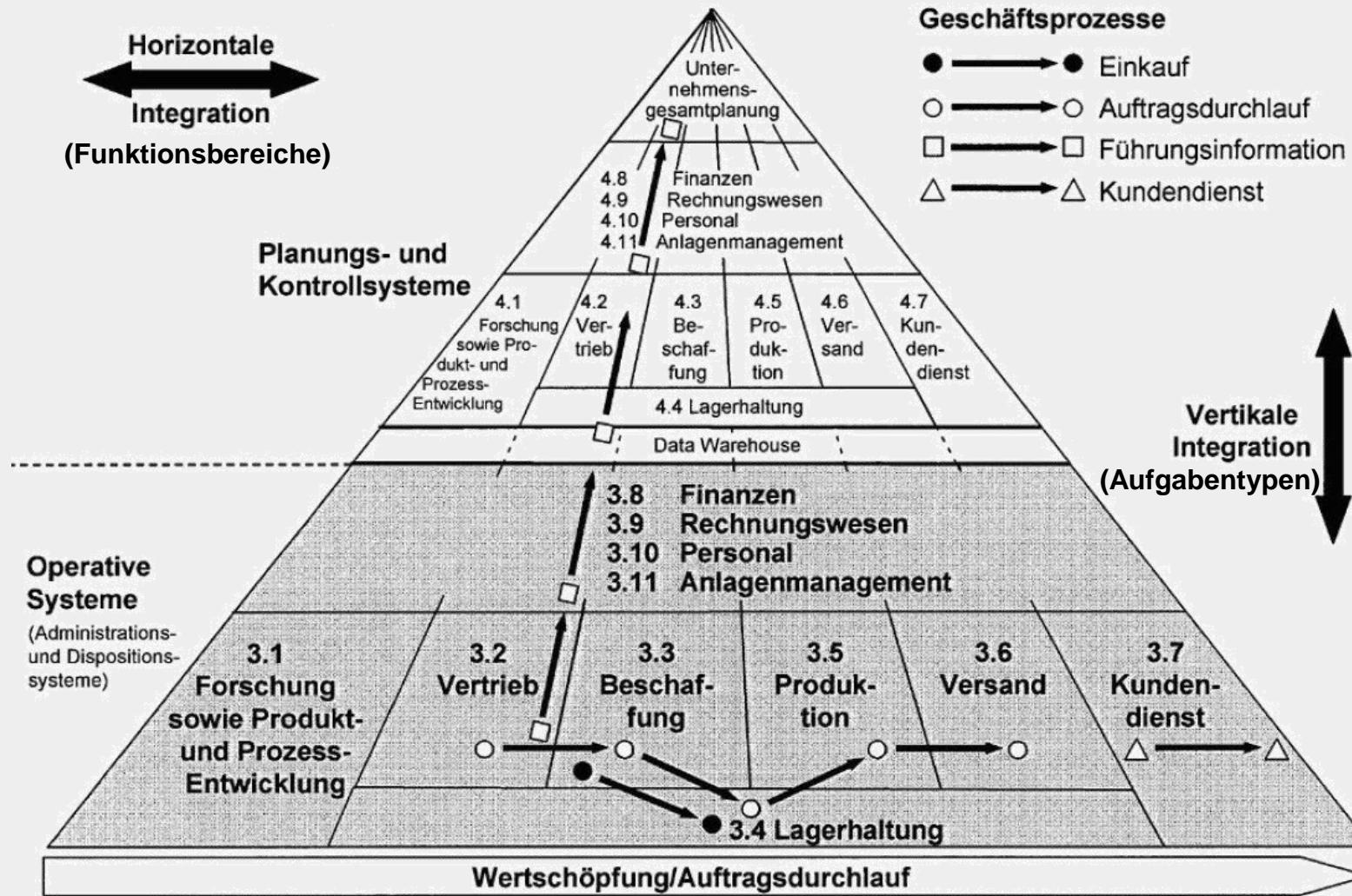
- vereinigen folgende Komponenten\*:
  - personelle (Qualifikation und Motivation),
  - organisatorische (Aufbau- und Ablauforganisation) und
  - technische (Hardware und Software)
- Struktur des jeweiligen BISs durch Kombination besagter Komponenten bestimmt\*
- dienen der Informationsversorgung von Akteuren\*
- verarbeiten Informationen
- erlauben das zielgerichtete Anlegen, Anzeigen, Ändern (und Löschen) von Daten (CRUD).
- sollten wirtschaftlich arbeiten
- \*[Gabler Wirtschaftslexikon]

## Arten betrieblicher Informationssysteme

- Administrations- und Dispositionssysteme
  - Administrationssysteme:  
Automatisieren einfacher, häufig wiederkehrender Routineaufgaben
  - Dispositonssysteme:  
Unterstützen (oder Automatisieren) einfacher, häufig wiederkehrender Entscheidungen
- Planungs- und Kontrollsystme
  - Planungssysteme:  
Lösen schlechtstrukturierter Probleme
  - Kontrollsystme:  
Überwachen des Einhaltens von Plänen



# Pyramide betrieblicher Informationssysteme [Mertens]



## Ausprägungen betrieblicher Informationssysteme

- ERP-System
- PPS-System
- CRM-System
- BDE-System
- MDE-System
- MIS/FIS
- CAQ-System
- MES
- PDM-System
- PLM-System
- Data-Warehouse
- OLAP-System
- CAD-System
- WFM-System
- DMS
- ...

## Ziel der Wirtschaftsinformatik

- Wirtschaftlicher (effizienter) Einsatz BISe

## Aufgaben der „angewandten“ Wirtschaftsinformatik

- Betriebliche Informationssysteme (BISe)
  - konzipieren (Lasten- und Pflichtenheft),
  - entwerfen (modellieren),
  - entwickeln,
  - anpassen,
  - erweitern,
  - konfigurieren (Customizing),
  - testen,
  - anwenden,
  - schulen,
  - warten,
  - einführen und
  - migrieren.
- „[automatisches] Generieren und Konfigurieren BISe aus formalisierten Anforderungen in Modellform“

## Aufgaben der Wirtschaftsinformatik als Wissenschaft

- Schaffen/Gewinnen von
  - Methoden,
  - Werkzeugen,
  - Modellen,
  - Terminologien,
  - Erkenntnissen,
  - ...
- zur besseren Gestaltung BISe

## „Wirtschaftsinformatik“ in englischer Sprache

- Business and Computer Science (0)
- Business Administration (-)
- Information Systems (+)
- Business Informatics (++)
- Commercial Information Technology (--)
- Information Management (--)



## Gliederung

---

1. Inhalte und Aufgaben der Wirtschaftsinformatik
2. **Grundlagen der Informatik und der Informationstechnik**
3. Informationsmanagement
4. Modellierung
5. Datenbanken
6. Softwareentwicklung
7. Betriebliche Informationssysteme

# Grundlagen der Informatik und der Informationstechnik

- **Berechenbarkeit**
- Komplexität
- Heuristiken
- Zahlensysteme
- Rechnerarchitektur
- Programmiersprachen
- Rechnernetze
- Softwarearchitektur
- Kryptographie

## Berechenbarkeit

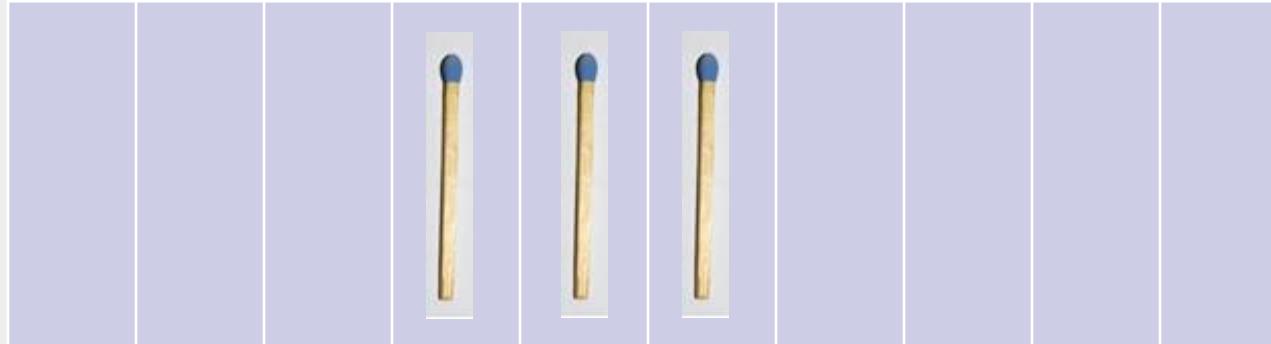
- Problem berechenbar, wenn Algorithmus existent, der für jeden Eingabewert nach endlich vielen Schritten anhält und Ergebnis liefert
- Algorithmus: „mit formalen Mitteln eindeutig beschreibbare , effektiv nachvollziehbare Verarbeitungsvorschrift zur Lösung einer Klasse von Problemen im Sinne der Transformation von Eingabe- in Ausgabedaten.“ [Schneidereit]

## Spiel zum Finden des Nachfolgers – Utensilien

- $1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 4, \dots$
- $i++, \text{System.inc}(), i = i + 1, \dots$
- Streichhölzer für natürliche Zahlen
  - 1 Streichholz  $\rightarrow 1$
  - 2 Streichhölzer  $\rightarrow 2$
  - 3 Streichhölzer  $\rightarrow 3$
  - ...
- Eine Münze für zwei Zustände
  - Kopf  $\rightarrow$  Zustand 0
  - Zahl  $\rightarrow$  Zustand 1
- Ein Bleistift als Markierer



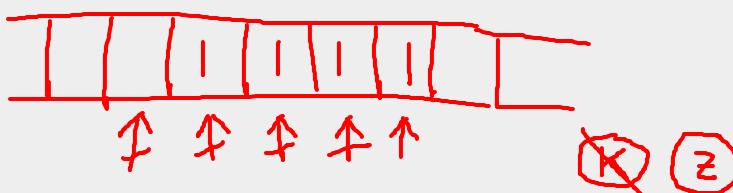
## Ausgangssituation des „Nachfolgerspiels“



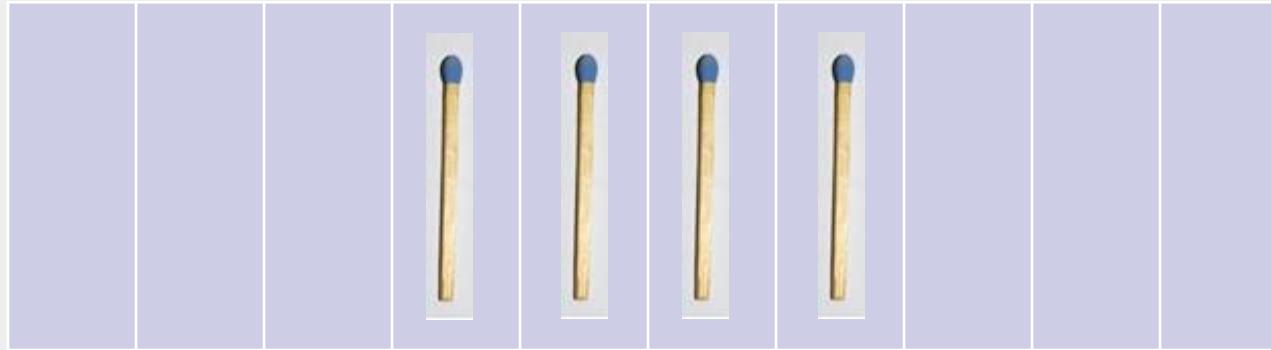
Gesucht: Nachfolger von 3

# Spielregeln des „Nachfolgerspiels“

Regel#	Wenn ...		..., dann		
	Münze zeigt	Bleistift zeigt auf	Münze drehen bzw. lassen auf	Aktuelles Feld	Bleistift
1	Kopf 	Leeres Feld	Kopf 	Leer lassen	Ein Feld nach rechts
2	Kopf 	Feld mit Streichholz 	Zahl 	Streichholz liegenlassen 	Ein Feld nach rechts
3	Zahl 	Feld mit Streichholz 	Zahl 	Streichholz liegenlassen 	Ein Feld nach rechts
4	Zahl 	Feld ohne Streichholz	Zahl 	Streichholz hineinlegen 	Hält an



## Endsituation des „Nachfolgerspiels“



Gefunden: Nachfolger von 3 → 4

## Ausprobieren des „Nachfolgerspiels“

- Anschreiben an der Tafel

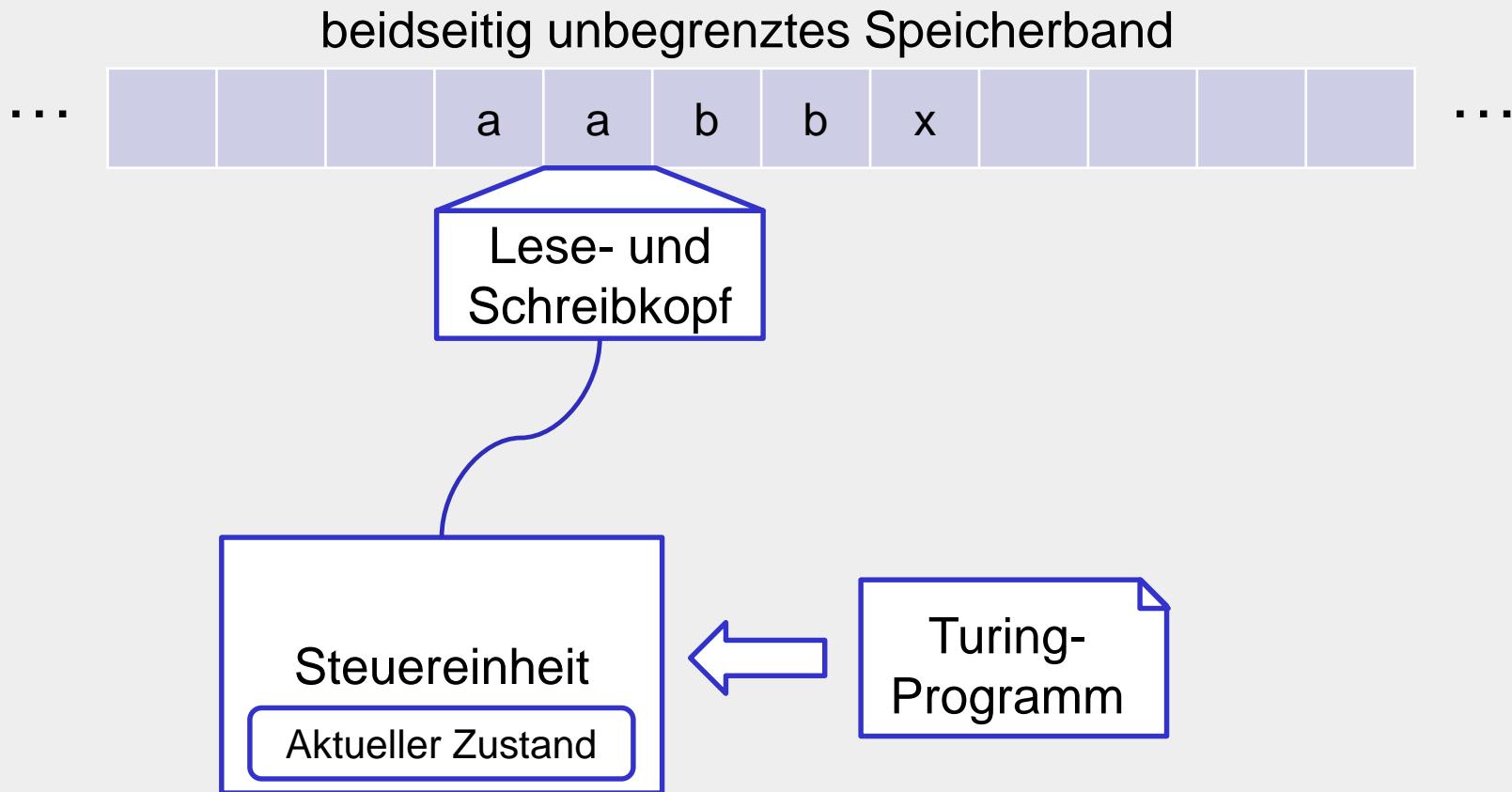
## Bisher verwendet

- Unbegrenztes Feld (Array), in dessen Elementen jeweils ein Zeichen steht (Ausgangssituation)
- Zeichen aus dem Alphabet {„ „, 1} (Streichholz und kein Streichholz)
- Zustände {Kopf, Zahl}
- Schreib- und Lesekopf (Markierer, Bleistift) mit Ausgangsposition
- Programm (Entscheidungstabelle, Produktionsregeln, Spielregeln)

## Bisher durchgeführt

- Input
  - Ausgangssituation (Problem, Aufgabe)
  - Programm (Regeln, Entscheidungstabelle)
- Abarbeiten des Programmes
- Output: Endsituation (Lösung)
- Transformatives System, kein reaktives:
  - einmaliger Input,
  - Abarbeiten des Programmes,
  - einmaliger Output,
  - Terminieren des Programmes
- Zustandsbasiert: Man weiß nicht, wie man in einen bestimmten Zustand gelangt, aber man weiß, was in einem bestimmten Zustand zu tun ist und was der Nachfolgezustand ist.

## Turing-Maschine – grafisch



## Turing-Maschine – informal (I)

- Unendliches(, eindimensionales) Speicherband
- Menge von Zeichen (Alphabet), die (das) das Leerzeichen enthält
- Lese-/Schreibkopf, der
  - Zeichen vom Speicherband liest und
  - Zeichen auf das Speicherband schreibt: WRITE(e)
  - über das Speicherband bewegt wird
    - ein Feld nach rechts (RIGHT, R)
    - ein Feld nach links (LEFT, L)
  - endgültig anhält (HALT, H)

## Turing-Maschine – informal (II)

- Steuereinheit, die den Lese- und Schreibkopf steuert
  - Enthält aktuellen Zustand aus einer Menge von Zuständen
- Überführungsfunktion (Turing-Programm):
  - ordnet einer Kombination aus einem aktuellen Zustand und einem gelesenen Zeichen eine Kombination aus einem (ggf. anderen) Zustand, einem zu schreibenden Zeichen und einer Bewegung des Lese- und Schreibkopfes (rechts, links, halt) zu

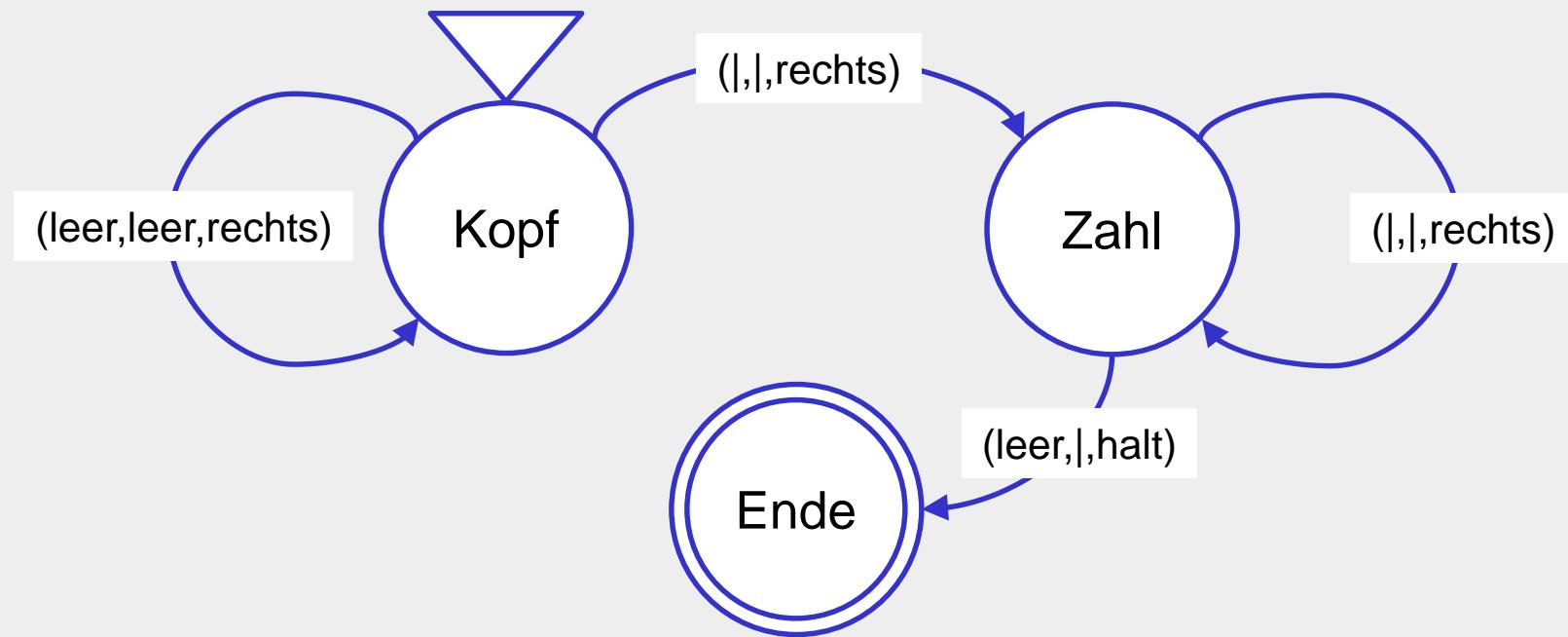
## Turing und seine Maschine

- 1900: Das Hilbertsche Entscheidungsproblem:  
„Gibt es ein Verfahren, das für jede ausreichend formalisierte Aussage der Mathematik entscheidet, ob diese [die Aussage] wahr oder falsch ist?“
- 1936: Turings (und Churchs) Beweis, dass obiges Entscheidungsproblem nicht lösbar ist
- 36 Jahre Differenz?
- Beweis, dass ein Verfahren zur Lösung eines Problems existiert?  
→ Verfahren angeben!
- Beweis, dass **kein** Verfahren zur Lösung eines Problems existiert?  
→ Wie ist ein Verfahren überhaupt definiert?  
→ Turing-Maschine!

## Church und seine These

- These:  
„Alles was überhaupt (intuitiv) berechenbar ist, ist schon mit der Turingmaschine berechenbar.“
- These, reformuliert:  
„Alles was mit der Turingmaschine nicht berechenbar ist, lässt sich überhaupt nicht berechnen.“
- Bisher kein berechenbares Problem, das nicht mit der Turingmaschine lösbar
- → These wird akzeptiert

## Zustandsdiagramm/-graph „Nachfolger“



# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- **Komplexität**
- Heuristiken
- Zahlensysteme
- Rechnerarchitektur
- Programmiersprachen
- Rechnernetze
- Softwarearchitektur
- Kryptographie

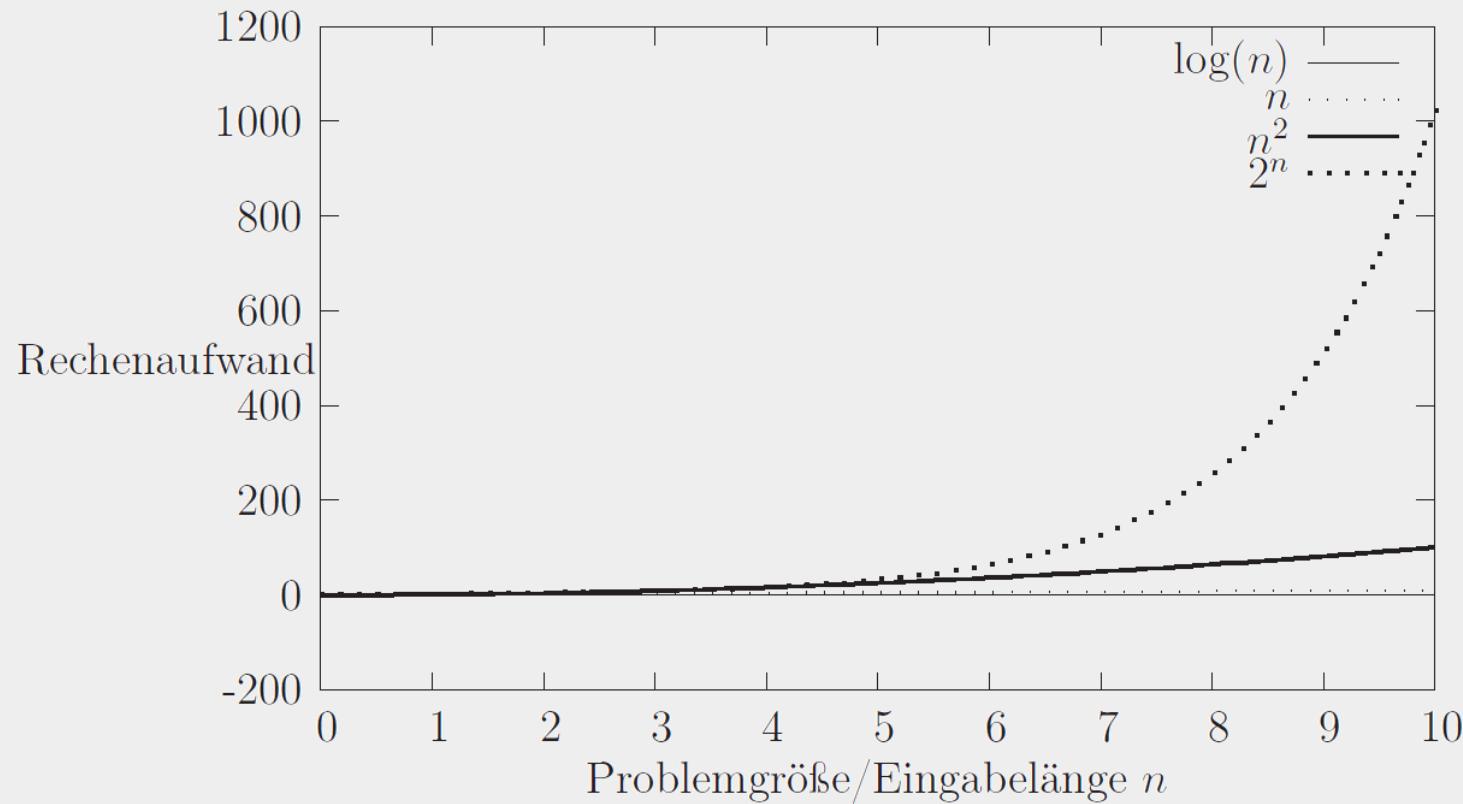
## Komplexität

- Nicht mehr Berechenbarkeit von Problemen (ist gegeben)
- Sondern Quantifizierung der Laufzeit von Algorithmen
- Asymptotische Laufzeitkomplexität (für große Problemgröße  $n$ )
- Worst-Case-Komplexität; Average-Case-Komplexität oft geringer
- Für Average-Case-Komplexität:
  - Verteilung der Eingaben nicht allgemein bekannt, z. B. Vorsortierung der Elemente beim Sortieren
  - Durchschnittlicher Berechnungsaufwand bei nicht-trivialen Algorithmen schwer zu bestimmen
- Eigentlich auch Speicherkomplexität relevant, aber
  - Laufzeit durch mehr Speicher reduzierbar
  - Benötigter Speicher durch längere Laufzeit reduzierbar

## Laufzeitklassen (O-Notation)

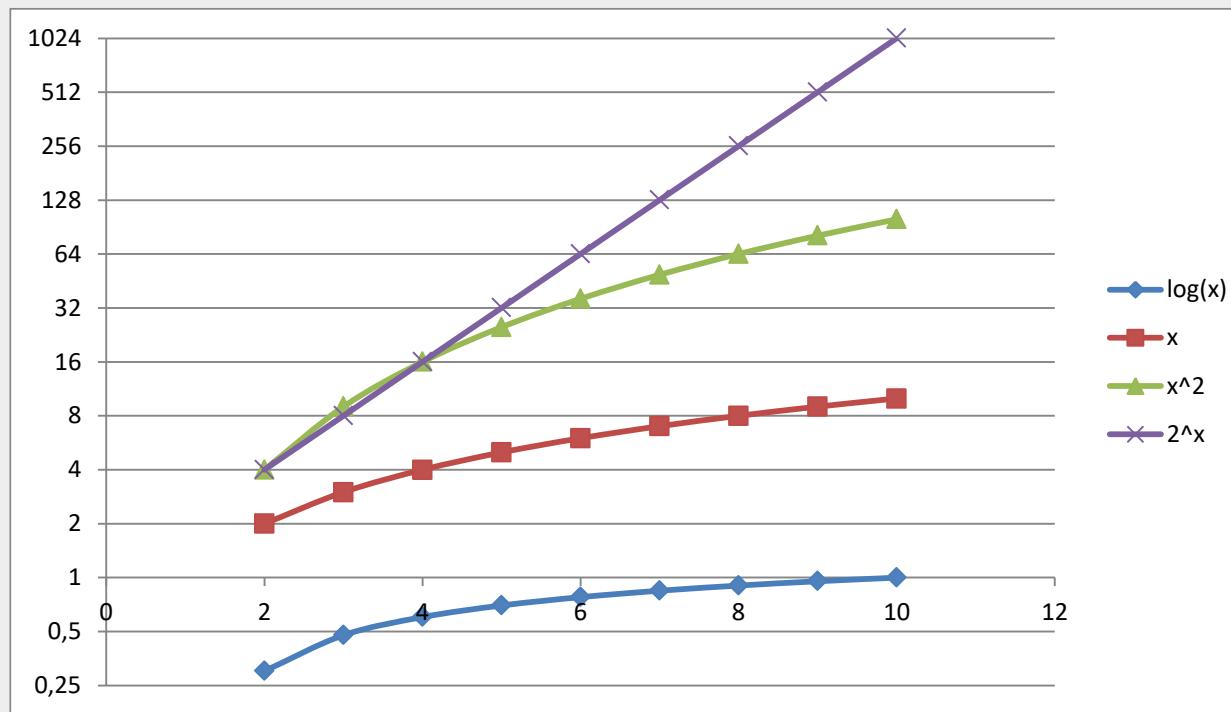
Laufzeitklasse	Komplexität	Bemerkung
$O(1)$	konstant	Von $n$ unabhängig
$O(\log n)$	logarithmisch	$2^n \rightarrow$ Laufzeit + k, binäre Suche
$O(n)$	linear	$\sim n$ , sequentielle Suche
$O(n \log n)$	-	Sortieren von $n$ Objekten
$O(n^k)$	polynomial	Algorithmen mit $k$ geschachtelten Schleifen, $n$ -fach durchlaufen
$O(k^n)$	exponentiell	Für großes $n$ nicht möglich

## Laufzeiten grafisch [Munkelt]



**Abbildung** : Beispiele für Verläufe von logarithmischem  $\log(n)$ , linearem  $n$ , polynomialem  $n^2$  und exponentiellem  $2^n$  Rechenaufwand in Abhängigkeit von der Problemgröße/Eingabelänge  $n$

# Laufzeiten, logarithmische Skalierung



## Komplexitätsanalyse am Beispiel

Quelltext	Operationen	Kommentar
r:=0; a:=0;	2	2 Zuweisungen
while a < n do	n+1+1	(n + 1) Vergleiche + 1 Sprung
a:=a+1;	2*n	n*(1 Addition + 1 Zuweisung)
b:=0;	n	n*(1 Zuweisung)
while b < a do	(1+2+...+n)+n+n=n(n+1)/2+2n =0,5n^2+2,5n	n(n+1)/2 erfolgreiche Vergleiche + n erfolglose Vergleiche + n Sprünge
r:=r+1;	2*n(n+1)/2=n(n+1)=n^2+n	2 Operationen * n(n+1)/2 Durchläufe
b:=b+1;	n^2+n	wie Zeile darüber
end;	=2+n+2+2n+n+0,5n^2+2,5n+n^2+n +n^2+n	Summe aller Zeilen
end;	=2,5n^2+8,5n+4	zusammengefasst
	→ O(n^2)	Konstanten und Terme niedrigerer Ordnung vernachlässigen

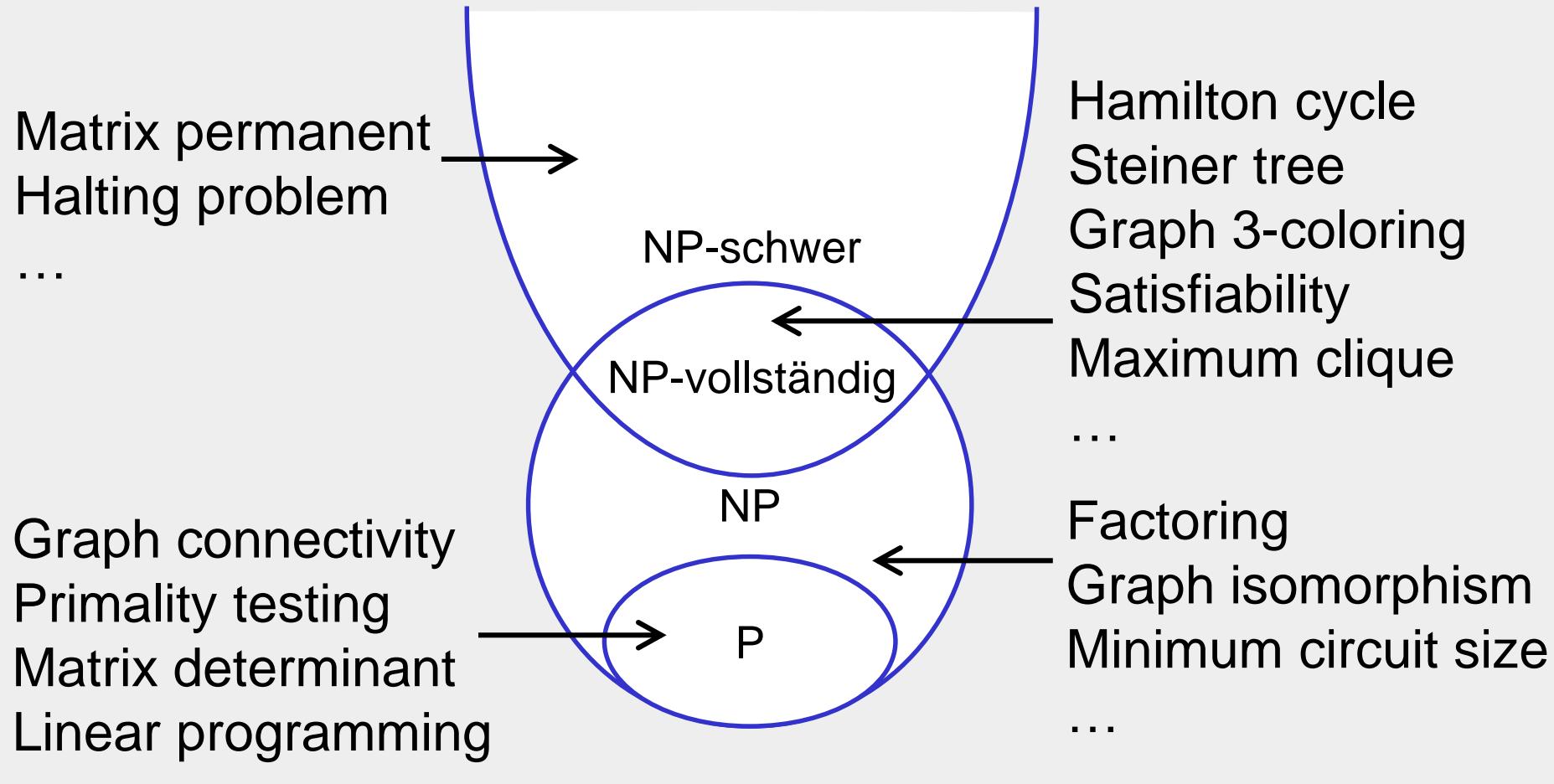
## Komplexitätsklassen P und NP – informal

- Problem in Komplexitätsklasse P:
  - Problem auf deterministischer Turingmaschine lösbar
  - Zeitkomplexität durch Polynom der Form  $n^k$  nach oben beschränkt
  - → Problem deterministisch in polynomialer Laufzeit lösbar
- Problem in Komplexitätsklasse NP:
  - Problem auf nicht-deterministischer Turingmaschine lösbar:
    - Lösung effizient nicht-deterministisch erzeugbar
    - Effizient überprüfbar, ob Zielfunktionswert unter gegebener Schranke
  - Zeitkomplexität durch Polynom der Form  $n^k$  nach oben beschränkt
  - → Problem nicht-deterministisch in polynomialer Laufzeit lösbar
- $P \subset NP$ , aber  $P = NP$ ?
  - Ja: Probleme aus NP effizient lösbar
  - Nein: Probleme aus NP nur in exponentieller Laufzeit lösbar

## NP-schwer und NP-vollständig

- Problem NP-schwer, wenn:
  - jedes Problem in NP in polynomialer Zeit auf Problem zurückzuführen
  - → Lösung jedes anderen Problems in NP durch Lösung dieses Problems ermöglicht
- Problem NP-vollständig:
  - Problem in NP  
**und**
  - Problem NP-schwer
- „Algorithmus gut für eines dieser Probleme, Algorithmus gut für alle anderen dieser Probleme“

## P, NP, NP-vollständig, NP-schwer



[Scott Aaronson 2004]

# Einige Probleme und ihre Komplexitätsklassen

Problem	Komplexi-tätsklasse	Beschreibung - informal
Kürzeste-Wege-Problem	P	Kürzesten Weg zwischen zwei Knoten in einem Graphen finden
Travelling-Salesman-Problem (Rundreiseproblem)	NP-voll-ständig	Rundreise durch mehrere Orte und zurück zum Ausgangspunkt mit kürzester Strecke
Rucksack-problem	NP-voll-ständig	Rucksack mit Dingen so packen, dass Gesamtgewicht nicht überschritten und Gesamtwert maximal
Graphenfärbung (chromatische Zahl)	NP-voll-ständig	Graphen mit so wenig wie möglich Farben so färben, dass keine adjazenten Knoten gleiche Farbe aufweisen
n-Puzzle	NP-voll-ständig	Ist eine bestimmte Ausgangsposition in einem 15er-Schiebepuzzle lösbar?
Cliquen-Problem	NP-voll-ständig	Clique mit mehr als k Knoten in einem Graphen existent?

# NP-vollständige Probleme in der PPS [Munkelt]

	Teilaufgabe der PPS			Komplexitätsklasse
			polyno- mial	NP- vollständig
<b>4</b>	Primärbedarfsplanung	Primärbedarfsprognose		X
		Grobplanung		X
	Mengenplanung	Materialbedarfsplanung	bedarfs- gesteuert	X
			verbrauchs- gesteuert	X
	Auftragsplanung			X
	Terminplanung	Arbeitsplanauflösung		X
<b>5</b>		Durchlaufterminierung		X
Kapazitätsplanung	Kapazitätsbedarfsrechnung		X	
	Kapazitätsterminierung		X	
Auftragsfreigabe	Verfügbarkeitsprüfung		X	
	Reihenfolgeplanung		X	
<b>6</b>	Produktionsregelung	Produktionsüberwachung		X
		Eingriffe in die Produktion		X

# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- **Heuristiken**
- Zahlensysteme
- Rechnerarchitektur
- Programmiersprachen
- Rechnernetze
- Softwarearchitektur
- Kryptographie

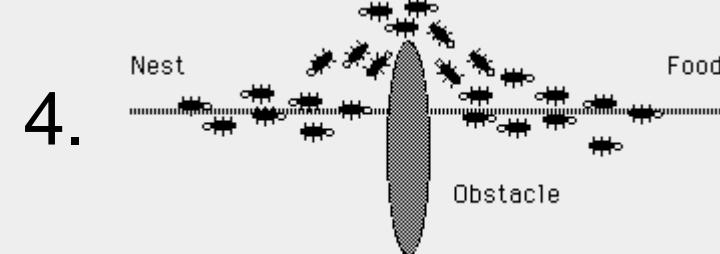
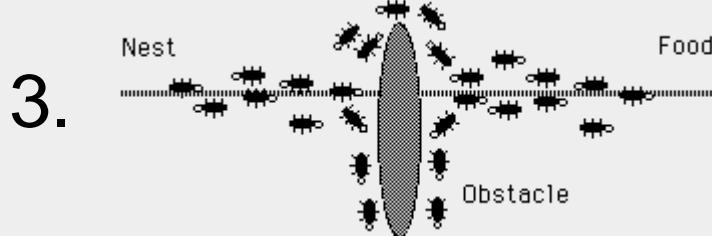
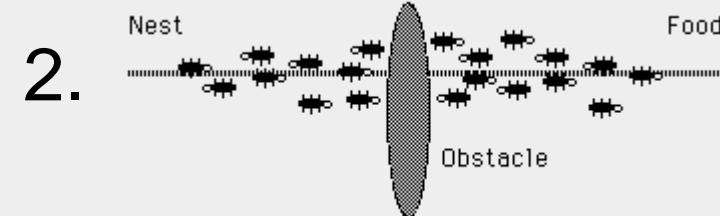
## Aussagen zur Sinnfälligkeit optimaler Lösungen

- „[...] the notation of an exact optimum does not appear to be very meaningful anyway in the light of the crudeness of data and concepts.“ [Hanssmann]
- „Eine schnelle Näherungslösung ist besser als ein langwierig gewonnenes Optimum, das ohnehin nur das Optimum des Modells und nicht das der erwarteten Realität ist.“ [Stahlknecht]
- Optimale Lösung nicht zwingend robust
- Kleine Änderung der Situation, dramatische Verschlechterung des Zielfunktionswertes
- Lieber gute, robuste Lösungen als optimale
- Optimale Lösungen durch falsche Eingabedaten und Änderung der Situation in der Realität sowieso suboptimal

## Heuristiken

- keine optimale Lösung von Problemen in akzeptabler Laufzeit
- → Heuristiken (approximative Verfahren)
- Systematisch möglichst gute, aber nicht zwingend optimale Lösungen erzeugt
- Laufzeit akzeptabel
- möglichst Anytime-Eigenschaft
- Meta-Heuristiken:
  - Simuliertes Abkühlen
  - Tabu-Suche
  - Genetische Algorithmen
  - Optimierung mit Ameisenkolonien
  - Ruin&Recreate (Optimieren mit Bomben)
  - ...

## Wie Ameisen den kürzesten Weg finden

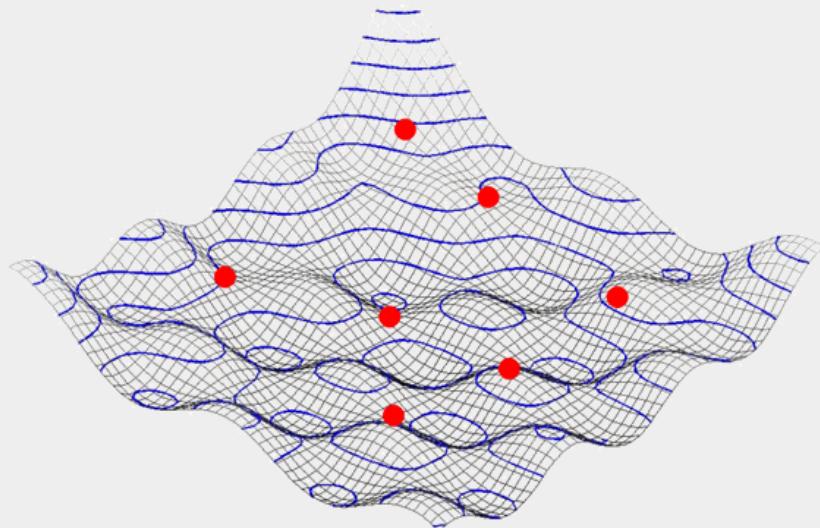


[<http://www.aco-metaheuristic.org>]

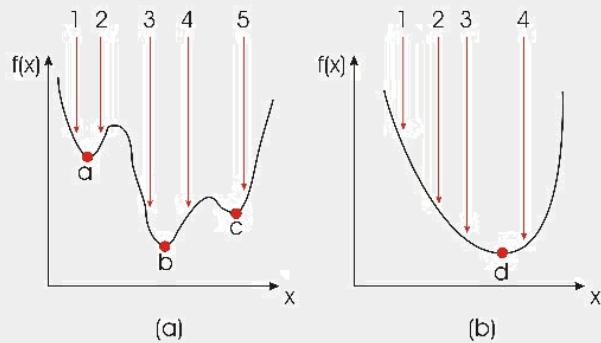
## Genetische Algorithmen

- Erzeuge Population von Individuen (potenzielle Lösungen)
- Bestimme Fitness (grob: Lösungsgüte) jedes Individuums
- Solange Abbruchkriterium (Zeit, Fitness, ...) nicht erreicht:
  - **Selektion** von Individuen (hohe Fitness wahrscheinlicher)
  - Erzeuge neue Individuen durch **Rekombination** (Kreuzen) selektierter
  - **Mutation** der neu entstandenen Individuen
  - Bestimmen der Fitness der neuen Individuen
  - Wähle Individuen für neue Generation (hohe Fitness wahrscheinlicher)
- Vorteile:
  - Schnell zu implementieren, gut parallelisierbar, gut anpassbar
- Nachteile:
  - Unbekannt, wie gut Ergebnis in Relation zum globalen Optimum
  - Operatoren für Rekombination und Mutation schwer zu finden

# Genetische Algorithmen - grafisch



[Demeulenaere]



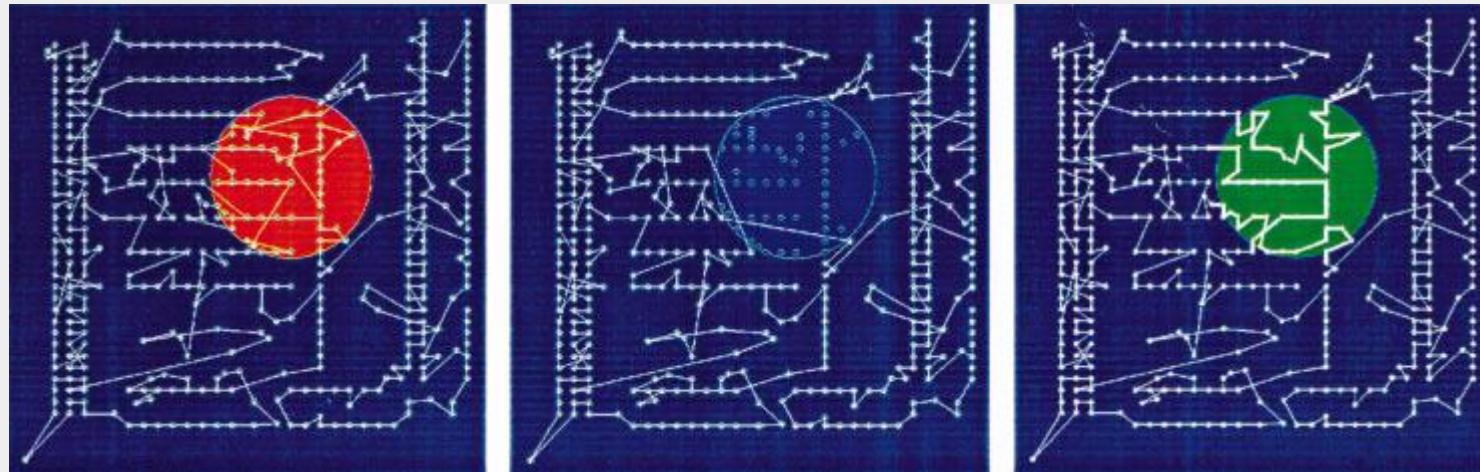
[Kurz et Tang]



## Ruin&Recreate

- „Optimieren mit Bomben“ [Spektrum der Wissenschaft 5/2000]:  
„Ein paar Bomben auf den Ballungsraum München/Nürnberg, und das Verkehrsproblem wäre gelöst.“ [Gedächtniszitat]
- Aufbau einer Initiallösung
- Solange Abbruchkriterium nicht erfüllt:
  - Zerstören zufällig ausgewählter Teile der Lösung
  - Wiederaufbau der zerstörten Teile

[Schrimpf et al.]



# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- Heuristiken
- **Zahlensysteme**
- Rechnerarchitektur
- Programmiersprachen
- Rechnernetze
- Softwarearchitektur
- Kryptographie

## Was ist der Unterschied zwischen Ingenieur und Informatiker?



Ein Kilobyte sind  
1.000 Byte.



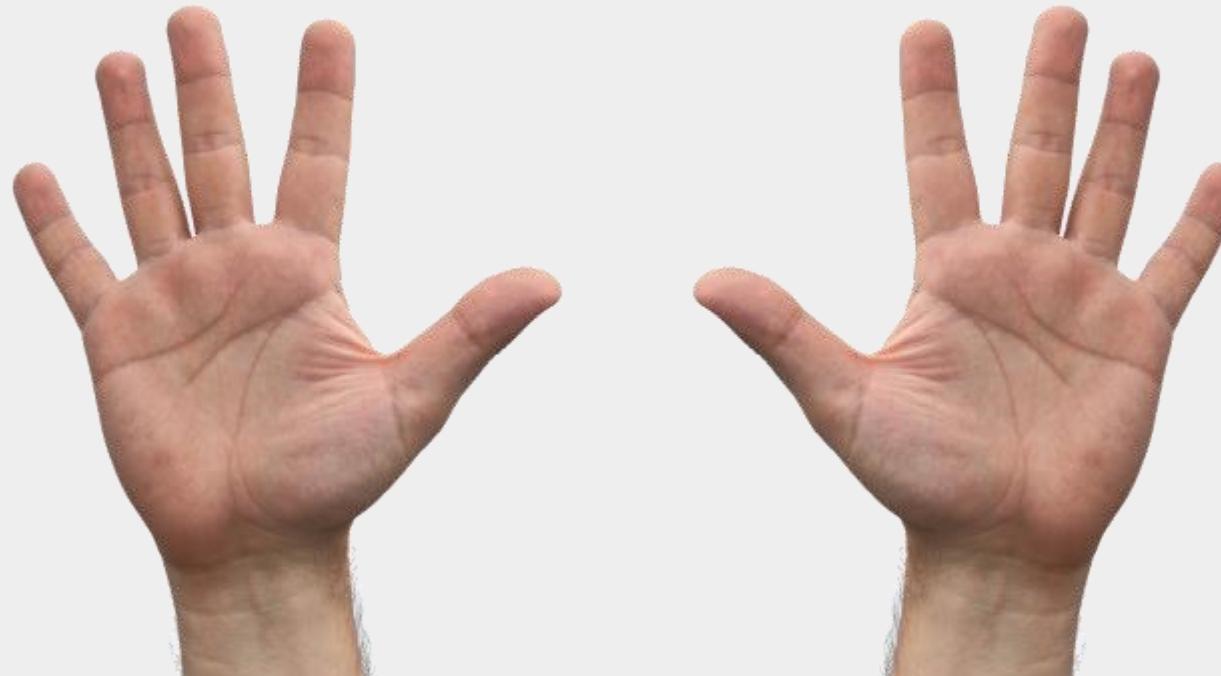
Ein Kilometer  
sind 1.024 Meter.



## Informationen als Daten im Rechner

- Rechnerinterne Darstellung von Informationen in elementaren Informationseinheiten Bits (binary digits)
- Alphabet: {0, 1}; Alternativen: {falsch, wahr}, {nein, ja}, ...
- Technisch: „Spannung liegt an (oder nicht).“
- Abbildung von beliebigen Symbolen und somit Informationen durch Binärzeichen, z. B.:
  - Schwarz → 00
  - Rot → 01
  - Grün → 10
  - Blau → 11
- Mit n Binärzeichen  $2^n$  Zustände abbildbar

## Wie weit kann man mit zehn Fingern zählen?



## Zahlensysteme

- Stellenwertsysteme: Wert einer einzelnen Ziffer entsprechend ihrer Position in der Zahl mit einer Basis<sup>Position</sup> gewichtet
- Gebräuchliche Basen zum Wichten in der (Wirtschafts)informatik:
  - Basis 2, Dualsystem, {0, 1}
  - Basis 8, Oktalsystem, {0, ..., 7}
  - Basis 16, Hexadezimalsystem, {0, ..., 9, A, ..., F}

## Zahl → Wert

- Berechnung des Wertes einer n-stelligen Zahl a in Stellenwertsystem mit Basis b:

$$a_{n-1}a_{n-2}\dots a_0 \rightarrow v(a) = \sum_{i=0}^{n-1} a_i \cdot b^i$$

## Zahl → Wert (Beispiel)

- Berechnung des Wertes einer Dualzahl:

Stelle	7	6	5	4	3	2	1	0
Basis	2	2	2	2	2	2	2	2
Basis <sup>^</sup> Stelle	128	64	32	16	8	4	2	1
Dualzahl	1	0	1	0	1	1	0	0
Dualziffer*(Basis <sup>^</sup> Stelle)	128	0	32	0	8	4	0	0
Summe	<b>172</b>							

## Wert → Zahl

- Berechnung einer n-stelligen Zahl a aus ihrem Wert v in Stellenwertsystem mit Basis 2:

```
private static String value2binary(int value) {  
    String result = "";  
    int ganzQuot = value;  
    while (ganzQuot > 0) {  
        int rest = ganzQuot % 2;  
        result = String.valueOf(rest) + result;  
        ganzQuot = (int) ganzQuot / 2;  
    }  
    return result.toString();  
}
```

## Wert → Zahl (Beispiel)

- Berechnung einer Dualzahl aus ihrem Wert 172:

Stelle	9	8	7	6	5	4	3	2	1	0
Dividend	0	0	1	2	5	10	21	43	86	172
Divisor (Basis)	2	2	2	2	2	2	2	2	2	2
Quotient (abgerundet)	0	0	0	1	2	5	10	21	43	86
Rest (Dualzahl)	0	0	1	0	1	0	1	1	0	0

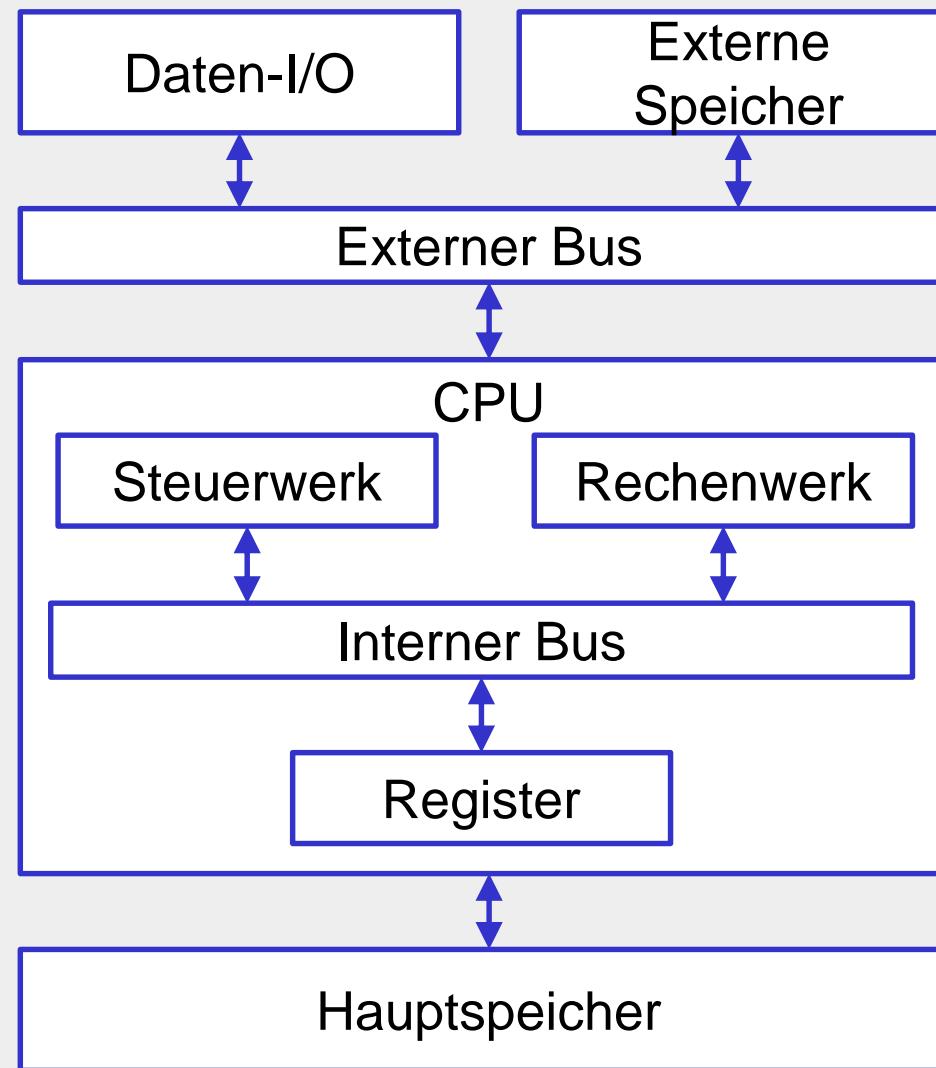
# Darstellung einer Gleitkommazahl

- Beispielzahl 2,7183:
    - $2,7183 = 0,27183 \cdot 10^1 = 0,27183 \text{ E}1$
  - Beispielzahl 0,0013:
    - $0,0013 = 0,13 \cdot 10^{-2} = 0,13 \text{ E}-2$
  - Mantisse in festem Format
  - Exponent mit Vorzeichen in festem Format
  - Basis (hier 10) durch Definition ebenfalls fest und implizit bekannt

# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- Heuristiken
- Zahlensysteme
- **Rechnerarchitektur**
- Programmiersprachen
- Rechnernetze
- Softwarearchitektur
- Kryptographie

# Hardware – Rechnerarchitektur (von-Neumann-Architektur)



## Pipeline-Prinzip (von-Neumann-Zyklus)

Phase	Aktion	Verantwortlicher
1	Befehl k aus HS holen	Steuerwerk
2	Operator, Adressen der Operanden und Sprungziel aus Befehl ermitteln	Steuerwerk
3	Operanden holen	Steuerwerk
4	Befehl ausführen: Operanden verarbeiten (und Ergebnis speichern)	Rechenwerk
5	$k := k + 1$ (Befehlszähler erhöhen)	Steuerwerk

(Register als Zwischenspeicher)

## Prozessor

- Auch Zentraleinheit oder Central Processing Unit (CPU) genannt
- Aufgabe: Befehle (Instructions) ausführen
- Taktfrequenz im GHz-Bereich
- Kommunikation mit anderen Komponenten über „Busse“ in niedrigerem Takt → Prozessortakt nicht entscheidend
- Arten:
  - CISC: Complex-Instruction-Set-Computer
  - RISC: Reduced-Instruction-Set-Computer

## Hauptspeicher

- Auch interner Speicher genannt
- Kapazität heute im GB-Bereich
- Durch virtuelle Speicherverwaltung (Paging) nur durch externen Speicher (Festplatte, SSD, ...) begrenzt
- Caching zwischen Hauptspeicher und Prozessor
- „Flüchtiger“ Hauptspeicher, aber ...

## Externer Speicher

- Festplatten
- Solid-State-Discs
- Derzeit mehrere hundert GB bis wenige TB
- Weitere Ausprägungen:
  - USB-Sticks
  - Magnetbänder
  - Disketten
  - CD-RW
  - DVD-RW
  - MO-Discs
  - ...

So lange hält ...	
Datenträger	Haltbarkeit
Steintafel	mehrere Tausend Jahre
Bücher aus säurefreiem Papier	mehrere Hundert Jahre
CD-ROM, Audio-CD, DVD (gepresst)	30 Jahre
CD-R, DVD-R	5 bis 10 Jahre
CD-RW, DVD-RW	2 bis 5 Jahre
Diskette	5 bis 10 Jahre
USB-Stick	3 bis 10 Jahre
Festplatte	3 bis 5 Jahre

Quelle: generelle Erfahrungswerte des Kompetenz-Netzwerks Langzeitarchivierung bei normaler Nutzung

Fotos: Corbis, ZB

## Zugriffszeiten

storage hierarchy		What does it mean in “our dimensions”?	
storage type	rel. access time	location	access time
register	1	my head	1 min
cache (on chip)	2	this room	2 min
cache (on board)	10	this building	10 min
main memory	100	Leipzig (by car)	~2 h
magnetic disk	$10^6\text{-}10^7$	Pluto ( $5910 * 10^6$ km)	>2 years
tape/opt. storage (automat. loading)	$10^9\text{-}10^{10}$	Andromeda	>2000 years

## Geräte an der Peripherie, I/O-Geräte

- Bildschirm
- Tastatur
- Maus
- Touch-Pad
- Mikrofon
- Kamera
- Barcode-Scanner
- RFID-Scanner
- Trackpoint
- Trackball
- ...

## Betriebssysteme

- Schnittstelle (Indirektionsebene) zwischen Anwendungssoftware und Hardware
- Anwendungssoftware abstrahiert von Hardware unterhalb des BS.
- Verwaltete Ressourcen:
  - Prozessor: Zuteilung von Prozessorzeit
  - Hauptspeicher: Verwaltung und Vergabe von Platz
  - Externe Speicher: Zugriff auf Dateisystem
  - I/O: Durchführung derselben mit Geräten an Peripherie
- Multitasking-Formen
- Derzeit noch weit verbreitet: Windows, Linux, Mac OS, Android

# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- Heuristiken
- Zahlensysteme
- Rechnerarchitektur
- **Programmiersprachen**
- Rechnernetze
- Softwarearchitektur
- Kryptographie

## Generationen der Programmiersprachen (I)

- 1. Generation: Maschinensprache
  - Form: Operation, Operand(en)
  - Alles binär (hexadezimal) dargestellt
  - Extrem schwer lesbar
  - Extrem schwer zu warten
  - Prozessorspezifisch

01D602  
2A0C40  
09  
54  
5D  
01B502  
2A0C40  
09  
EDB8  
C9

## Generationen der Programmiersprachen (II)

- 2. Generation: Assembler-Sprache
  - Selbe Abstraktionsebene wie Maschinensprache
  - Aber sprechende Abkürzungen für Befehle: jmp, mov, pop, push, ...
  - ... und textuelle Bezeichnung von Speicheradressen: wert, menge, summe, feld, ... möglich
  - Immer noch vom Prozessor abhängig
  - Leicht verbesserte Verständlich- und wartbarkeit

## Inline Assembler – Beispiel 1

```
procedure TForm1.btnGoClick(Sender: TObject);
var
  num, answer : integer;
begin
  num := StrToInt(edtInput.Text);
  //This is required with Lazarus on x86:
  {$ASMMODE intel}
  asm
    MOV EAX, num
    ADD EAX, 110B //add binary 110
    SUB EAX, 2      //subtract decimal 2
    MOV answer, EAX
  end;
  edtOutput.Text := IntToStr(answer);
end;
```

## Inline Assembler – Beispiel 2

```
function Less(  
    MemPtr1,MemPtr2:pointer;  
    MemWidth:integer):boolean; assembler;  
asm  
    PUSH ESI { Inhalt Register Extended-Source-Indexx auf Stack}  
    PUSH EDI { Inhalt Register E.-Destination-I. auf Stack}  
    MOV ESI,MemPtr1 { Inhalt MemPtr1 nach ESI}  
    MOV EDI,MemPtr2 { Inhalt MemPtr2 nach EDI}  
    XOR EAX,EAX { Register Extended-Accumulator auf 0 (falsch) }  
    REPE CMPSB { repeat string-comparsion bytewise while ESI=EDI }  
    JNB @@1 { jump (to @@1) if not below ( $\geq$ , carry-flag=0(?)) }  
    INC EAX { inkrementiere EAX (von 0) auf 1 (wahr) }  
@@1:POP EDI { Inhalt Stack zurueck auf EDI}  
    POP ESI { Inhalt Stack zurueck auf ESI}  
end;
```

## Generationen der Programmiersprachen (III)

- 3. Generation: Prozedurale Sprache
  - Problemorientiert
  - idealerweise maschinenunabhängig und portabel
  - WORA/WORE vs. WOCA
  - Sequenz, Verzweigung, Schleife, Unterprogrammaufruf
  - erlaubt oft Rekursion
  - Vertreter: C, Pascal, Fortran, Cobol, Basic, ...
  - Bessere Verständlichkeit
  - Spaghetti-Code möglich
  - Strukturierungsmöglichkeiten für große Anwendungssoftware ungenügend

## Prozedurale Sprache - Beispiele

```
function factorial(n:integer):integer;
begin
  if n = 0 then
    result := 1
  else
    result := n * factorial(n - 1);
end;
```

```
function factorial2(n:integer):integer;
var i:integer;
begin
  result := 1;
  for i := 2 to n do
    result := result * i;
end;
```

# International Obfuscated C Code Contest

```
X=1024; Y=768; A=3;
```

```
J=0;K=-10;L=-7;M=1296;N=36;O=255;P=9;_=1<<15;E;S;C;D;F(b){E="1""111886:6:??AAF"
"FHHMMOO55557799@>>>BBBGGIICK" [b]-64;C="C@==:C@==@=:C@=:C5""31/513/5131/"
"31/531/53" [b]-64;S=b<22?9:0;D=2; } I(x,Y,X){Y?(X^=Y,X*X>X?(X^=Y):0, I(x,Y/2,X
)):(E=X); } H(x){I(x, _,0); } p;q( c,x,y,z,k,l,m,a,
) ;x-=E*M ;y-=S*M ;z-=C*M ;b=x*x/M+ y*y/M+z
*z/M-D*D *M;a=-x *k/M -y*y1/M-z *m/M; p=( (b=a*a/M-
b)>0?(I (b*M,_,0),b =E, a+(a>b ?-b:b)): -1.0); } Z;W;o
(c,x,y, z,k,l, m,a){Z!= c? -1:z;c <44?(q(c,x ,y,z,k,
l,m,0,0 ),(p> 0&&c!= a&& (p<W ||z<0) )?(W=
p,Z=c): 0,o(c+ 1, x,y,z, k,l, m,a)):0 ; } Q;T;
U;u;v;w ;n(e,f,g, h,i,j,d,a, b,V){o(0 ,e,f,g,h,i,j,a);d>0
&&Z>=0? (e+=h*W/M, f+=i*W/M, g+=j*W/M, F(Z), u=e-E*M, v=f-S*M, w=g-C*M, b=(-2*u-2*v+w)
/3, H(u*u+v*v+w*w), b/=D, b*=b, b*=200, b/=(M*M), V=Z, E!=0? (u=-u*M/E, v=-v*M/E, w=-w*M/
E):0, E=(h*u+i*v+j*w)/M, h=-u*E/(M/2), i=-v*E/(M/2), j=-w*E/(M/2), n(e,f,g,h,i,j,d-1
,z,0,0),Q/=2,T/=2, U/=2, V=v<22?7: (V<30?1:(V<38?2:(V<44?4:(V==44?6:3)))) )
,Q+=V&1?b:0,T +=V&2?b :0,U+=V &4?b:0) :(d==P? (g+=2
,j=g>0?g/8:g/ 20):0,j >0?(U= j *j /M,Q =255- 250*U/M,T=255
-150*U/M,U=255 -100 *U/M):(U =j*j /M,U<M /5?(Q=255-210*U
/M,T=255-435*U /M,U=255 -720* U/M):(U =j*j /M,U<M /5?Q=213-110*U
/M,T=168-113*U /M,U=111 -85*U/M) ),d!=P? (Q/=2,T/=2
,U/=2):0);Q=Q< 0?0: Q>0? O: Q;T=T<0? 0:T>O?O:T;U=U<0?0:
U>O?O:U; } R;G;B ;t(x,y ,a, b){n(M*M+J*M *40*(A*x +a)/X/A-M*20,M*K,M
*L-M*30*(A*y+b)/Y/A+M*15,0,M,0,P, -1,0,0);R+=Q ;G+=T;B +=U;++a<A?t(x,y,a,
b): (++b<A?t(x,y,0,b):0); } r(x,y){R=G=B=0;t(x,y,0,0);x<X?(printf("%c%c%c",R/A/A,G
/A/A,B/A/A),r(x+1,y)):0; } s(y){r(0,--y,s(y),y:y); } main(){printf("P6\n%i %i\n255"
"\n",X,Y);s(Y); }
```

[Anders Gavare 2004]

## Generationen der Programmiersprachen (IV)

- 4. Generation: deklarative (deskriptive) Sprache
  - Nicht mehr programmiert, **wie** ein Problem gelöst werden soll
  - Sondern nur noch, **welches** Problem gelöst werden soll
  - Logische Sprache: Prolog
  - Funktionale Sprachen:
    - LISP
    - **Scheme**
    - Haskell
    - **XSLT**
    - ...
  - Weit verbreitet: SQL, neuerdings XSLT

## SQL - Beispiel

```
select
    c.customerid,
    c.name
from customers as c
    where c.country = 'DE'
        and c.turnover >= 2000000
order by
    c.name asc;
```

```
select
    o.customerid,
    sum(o.value)
from orders as o
group by o.customerid;
```

```
define liste := new Liste();
for each customer do
    if customer.country = 'DE'
        and customer.turnover > 2000000
    then
        liste.append(customer);
next customer;
liste.sort(customer.name);
```

// passt nicht  
// in dieses Kästchen ;-)

# XSLT-Fragment

```
<xsl:template match="employees">
  <xsl:element name="html">
    <xsl:element name="head">
      <xsl:element name="title">
        <xsl:value-of select="name()" />
      </xsl:element>
    </xsl:element>
    <xsl:element name="body">
      <xsl:element name="table">
        <xsl:attribute name="border">
          <xsl:value-of select="'1'" />
        </xsl:attribute>
        <xsl:apply-templates select="employee"/>
      </xsl:element>
    </xsl:element>
  </xsl:element>
</xsl:template>
```

# Logische Programmierung, Prolog – einfaches Beispiel

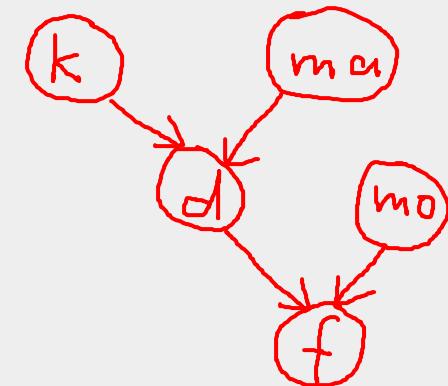
```
/* Fakten: */
elter(klaus,dieter).
elter(maria,dieter).
elter(dieter,franz).
elter(monika,franz).

/* Regeln: */
grosselter(X,Y) :-  
    elter(X,Z),  
    elter(Z,Y).

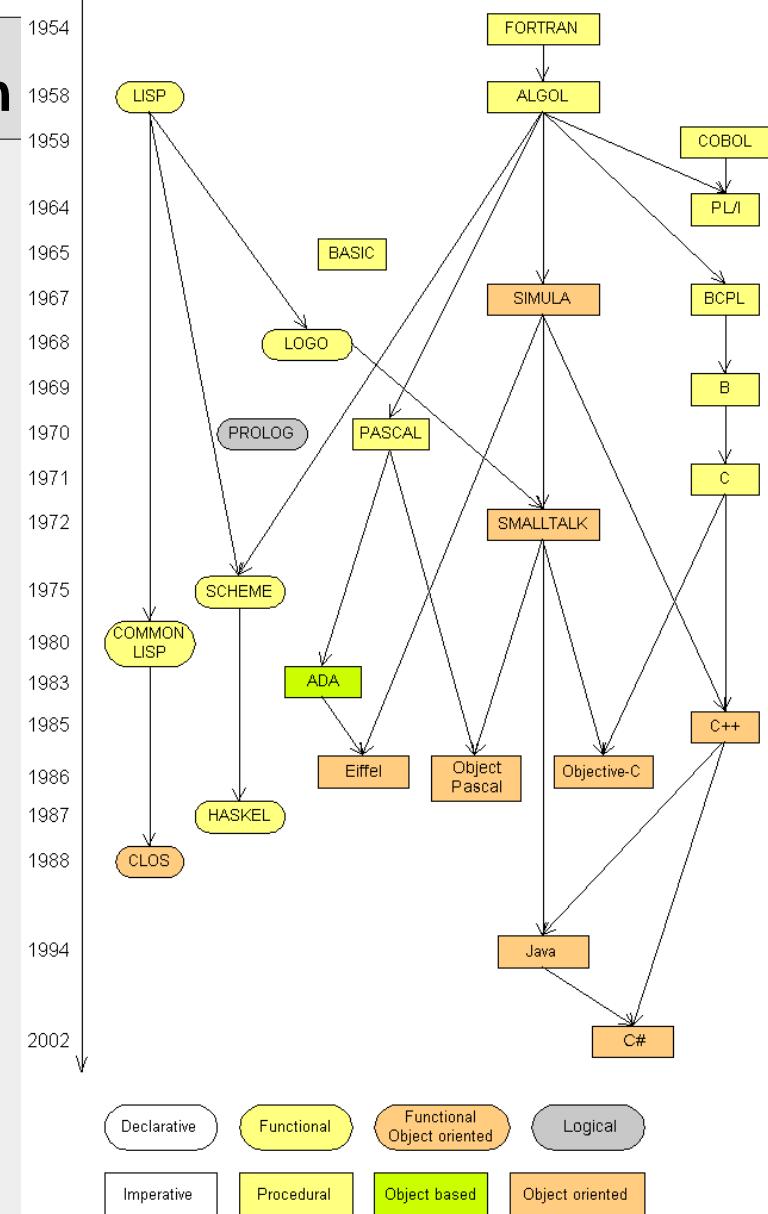
/* Fragen und Antworten: */
?- grosselter(monika,klaus).
no.

?- grosselter(maria,franz).
yes.

?- grosselter(X,franz).
X=klaus
X=maria
```



# Evolution der Programmiersprachen

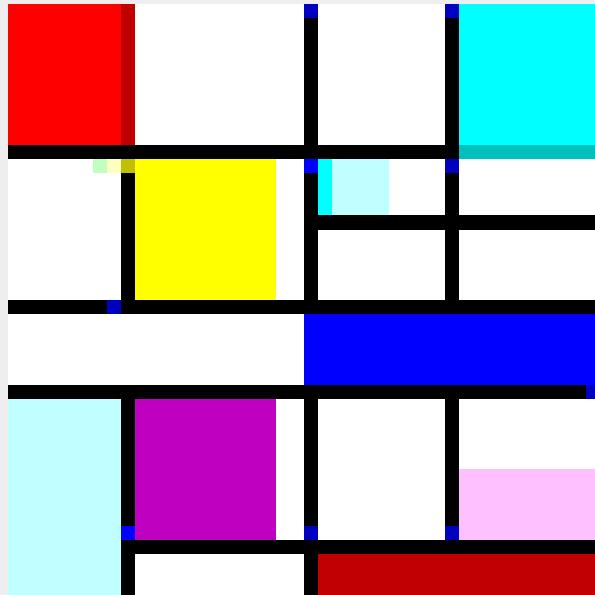


[Verkeyn]

## Esoterische Programmiersprachen

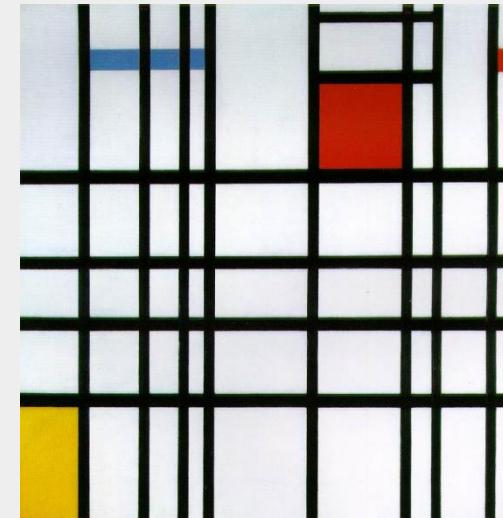
- Brainfuck: 8 Sonderzeichen für 8 Befehle
- [Küchen]Chef: englischsprachige Kochrezepte
- Malbolge: schwerste Sprache der Welt
- Ook!: für Orang-Utans (Ook. Ook? Ook!)
- **Piet: Quelle abstraktes Bild im GIFF-Format**
- Whitespace: Leerzeichen, Tab, Zeilenumbruch
- **Shakespeare Programming Language**

## Piet [David Morgan-Mar 2008]



„Piet“

- Programmcode abstraktem Bild ähnelnd
- Benannt nach Piet Mondrian, Pionier abstrakter Kunst
- Basiert auf Kellerspeicher-Operationen
- Gesteuert durch zwei Zeiger, die durch die Bild laufen
- Push, Pop, IO und Arithmetik durch z. B.:
  - Eintreten in schwarzes oder weißes Farbfeld,
  - Zahl zusammenhängender „Codels“,
  - Übergang zwischen zwei Farben



Composition with Red, Yellow and Blue  
1921; Oil on canvas, 39 x 35 cm

## Shakespeare Programming Language (SPL): Entwurfsziele

“The design goal was to make a **language with beautiful source code** that **resembled Shakespeare plays**. There are no fancy data or control structures, just basic arithmetic and gotos. You could say we have combined **the expressiveness of BASIC** with the **user-friendliness of assembly language**.”  
[Karl Hasselström und Jon Åslund 2001]

# SPL: From the Infamous Hello World Program.

Romeo, a young man with a remarkable patience.

Juliet, a likewise young woman of remarkable grace. [...]

Hamlet, the flatterer of Andersen Insulting A/S.

Act I: Hamlet's insults and flattery.

Scene I: The insulting of Romeo.

[Enter Hamlet and Romeo]

Hamlet:

**Romeo := 2 \* 2 \* 2 \* 2 \* 2 \* -1; /\* -64 \*/**

You lying stupid fatherless big smelly half-witted coward! You are as stupid as the difference between a handsome rich brave hero and thyself!

Speak your mind! [...]

**Romeo := 2 \* 2 \* 2 \* 1 – Romeo; /\* 72 \*/**

**Output(Ascii(ValueOf(Romeo))); /\* H \*/**

[Exit Romeo]

Scene II: The praising of Juliet.

[Enter Juliet]

Hamlet:

Thou art as sweet as the sum of the sum of Romeo and his horse and his black cat! Speak thy mind!

“As you see, this way of writing constants gives you much more poetic freedom than in other programming languages.”

[Karl Hasselström und Jon Åslund 2001]

## Compiler und Interpreter

### Compiler

- Grober Ablauf:
  - Prüfen auf syntaktische (und einfache semantische) Fehler
  - Übersetzen der Module in Maschinensprache
  - „Linking“ der Module
- Anschließend Ausführen des Programmes
- Schnelle Programme, aber plattformabhängig

### Interpreter

- Grober Ablauf:
  - Quelltext einlesen
  - Quelltext analysieren
  - Quelltext ausführen
- Alles zur Laufzeit des Programmes!
- Plattformunabhängig, aber relativ langsam
- JIT-Compiler ...
- Java-Byte-Code ...

# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- Heuristiken
- Zahlensysteme
- Rechnerarchitektur
- Programmiersprachen
- **Rechnernetze**
- Softwarearchitektur
- Kryptographie

## Ziele beim Einsatz von Rechnernetzen [Mertens/Stahlknecht]

Ziel	Realisierung	Verbundtyp
Gleichmäßige Auslastung mehrerer Rechner	Lastverteilung zwischen Rechnern	Lastverbund
Hohe Rechenleistung	Bündeln von Rechenressourcen	Leistungsverbund
Geringe Datenredundanz, wenige Anomalien	Paralleler Zugriff auf dieselben Daten	Datenverbund
Geringe Installations-, Wartungs- und Lizenzkosten	Gemeinsames Nutzen desselben Programmes	Programmverbund
Schneller Austausch von Nachrichten	Kommunikation zwischen Rechnern	Kommunikationsverbund
Geringe relative Kosten für periphere Geräte	Gemeinsames Nutzen von Geräten	Geräteverbund
Geringes Risiko des Datenverlustes	<i>Redundantes Speichern von Daten auf mehreren Rechnern</i>	Sicherheitsverbund

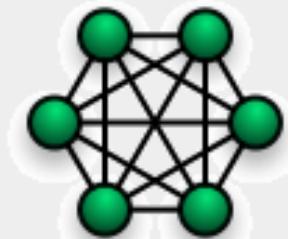
## Rechnernetze

- „.... räumlich verteiltes System von Rechnern, die durch Datenübertragungseinrichtungen und –wege miteinander verbunden sind.“ [Hansen]
- Local Area Network (LAN)
- Wide Area Network (WAN)
- Internet: Rechnernetz, das Rechnernetze vernetzt

## Bestandteile von Rechnernetzen

- Rechner mit Netzwerkanbindung
  - Netzwerkkarte/Modem
  - Netz- und Anwendungssoftware
- Netzwerkverbindungs- und –kommunikationskomponenten
  - Hub
  - Switch
  - Bridge
  - Router
- Datenübertragungswege
  - Verdrillte Kupferkabel
  - Koaxialkabel
  - Glasfaserkabel
  - Funk
  - Infrarot und Laser
- Protokolle

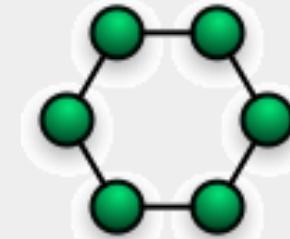
# Netzwerktopologien



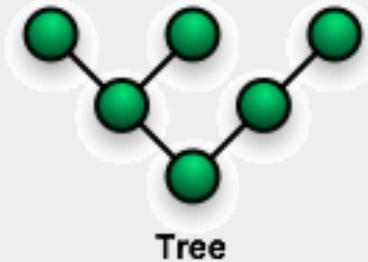
Fully Connected



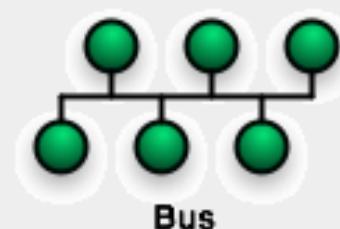
Star



Ring



Tree



Bus

[Bilder: wikipedia]

## Internet – einige Stationen der Entwicklung

Jahr	Ereignis
1969	Verbinden vierer Computer zum Advanced Research Projects Agency-NET (APRANET) durch US-Verteidigungsministerium
1972	Erstes E-Mail-Programm (Ray Tomlinson)
1974/75	TCP/IP (Vinton Cerf und Bob Kahn): Begriff „Internet“
1976	Ethernet-Technologie und Protokoll (Robert Metcalfe)
1978	Erste Spam-E-Mail (Gary Thuerk)
1989	Internetzugang per Modemeinwahl, erste kommerzielle ISPs
1992	WWW (Tim Berners-Lee, CERN)
1993	Erster Webbrowser Mosaic (Marc Andreessen)
1994	Erste Online-Bestellungen
2000	Platzen der .com-Blase

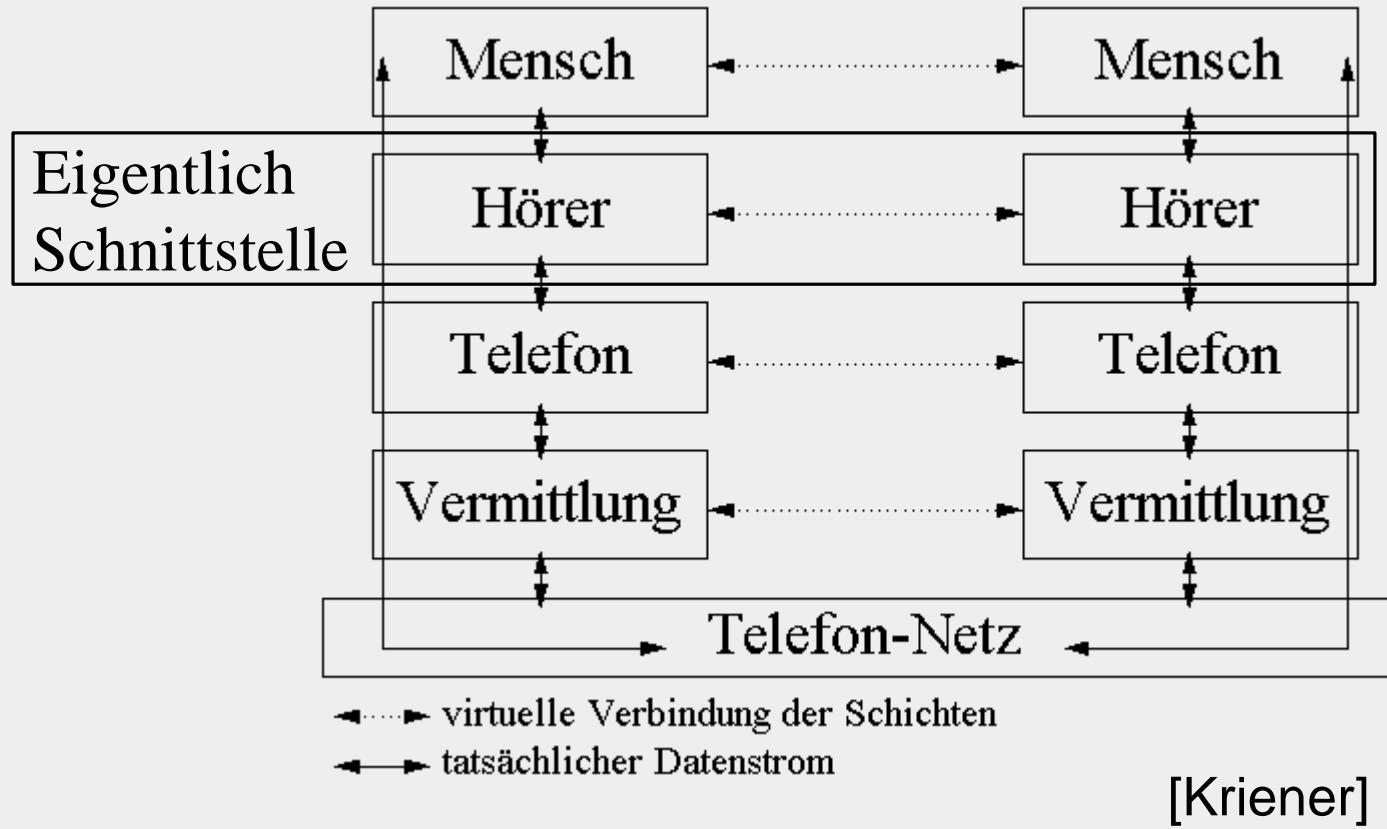
## Internet-Dienste – Auswahl

Dienst	Protokoll	Port
Telnet	TP	23
Dateiübertragung	FTP	20, 21
E-Mail	<b>SMTP, POP3, IMAP</b>	<b>25, 110, 143</b>
News	NNTP	119
Secure Shell	SSHP	22
VoIP	SIP	5060
WWW	<b>HTTP(S)</b>	<b>80, 443</b>
Namensauflösung	DNS	53
Talk	talk	517, 518
Internet-Relay-Chat	IRCP	194, 6667

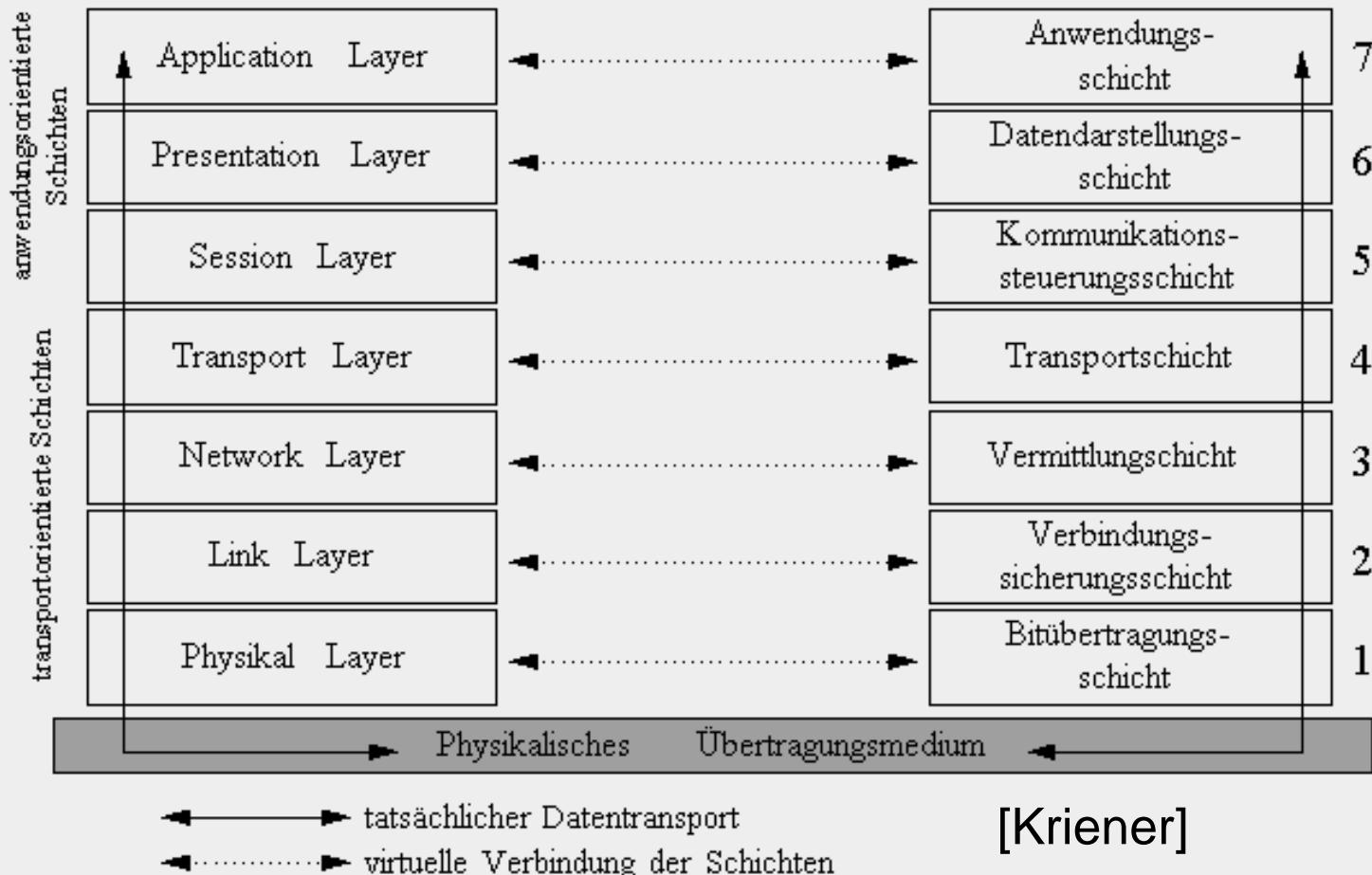
## Schichtenmodelle

- Logischer Anwendung nur auf Abstraktionsniveau der jeweiligen Schicht → Vereinfachung
- Verbergen der Implementierung der tiefer liegenden Schichten vor den höher liegenden
- Tiefer liegende Schichten logisch vernachlässigbar
- Kommunikation einer Schicht nur über Schnittstellen mit der direkt darunterliegenden Schicht (idealerweise)
- nur noch das WAS interessant, nicht mehr das WIE (OO:  
„Information Hiding“)
- Reduktion der Komplexität:
  - Geringe Kopplung
  - Hohe Kohäsion
  - Keine Zyklen → Austauschbarkeit der Implementierung der Schichten
- Eventuell bloßes „Durchreichen“ durch Schichten

## Schichtenmodell einer Kommunikation – Beispiel Telefon



# ISO/OSI-Referenzmodell Kommunikation in Rechnernetzen



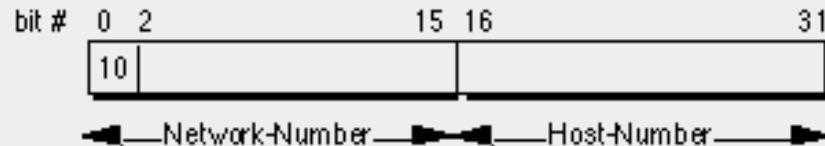
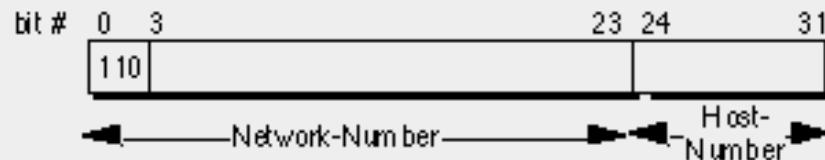
# TCP/IP-Kommunikationsprotokoll

ISO-Schicht	TCP/IP-Schicht	Beispielprotokolle
7 Anwendung	4 Anwendung	FTP SMTP HTTP
6 Darstellung		
5 Kommunikationssteuerung		
4 Transport	3 Transport	TCP, UDP
3 Vermittlung	2 Internet	IP
2 Sicherung	1 Netzzugang	Ethernet Token Ring
1 Bit-Übertragung		FDDI WLAN

[Stahlknecht]

# Adressierung im Internet - Adressklassen

**Class A**

**Class B**

**Class C**


[3Com]

Klasse	Präfix	Netze	Suffix	Hosts
A	7 Bit	$2^7 - 2 = 126$	24 Bit	$2^{24} - 2 = 16.777.214$
B	14 Bit	$2^{14} - 2 = 16.382$	16 Bit	$2^{16} - 2 = 65.534$
C	21 Bit	$2^{21} - 2 = 2.097.150$	8 Bit	$2^8 - 2 = 254$

# Leitungs- versus Speichervermittlung

## Circuit Switching

- Dauerhafte physische Verbindung zwischen Teilnehmern während gesamter Kommunikation
- Exklusiver Kanal für Teilnehmer während Kommunikation
- Adressinformation nur zum Verbindungsauflbau
- → schlechtere Netzauslastung, mehr „Besetzt“-Fälle, aber weniger Steuerinformation, keine Verzögerungen bei Verbindung, geringerer Aufwand

## Packet-Switching

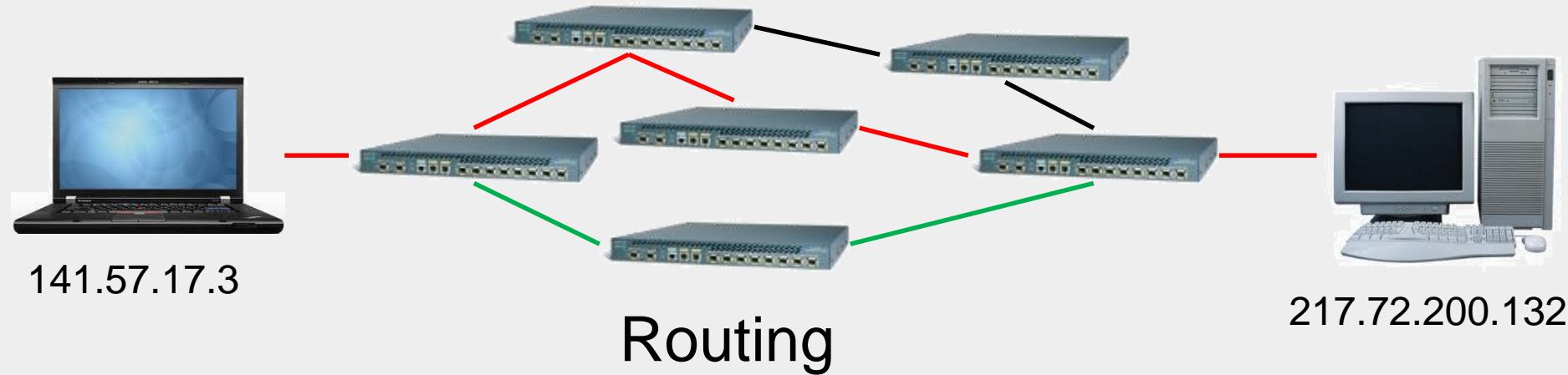
- Nur virtuell dauerhafte Verbindung zwischen Teilnehmern während Kommunikation
- Daten in Pakete zerlegt, auf verschiedenen Wegen durchs Netz transportiert (Routing) und wieder zusammengesetzt
- → bessere Netzauslastung, weniger „Besetzt“-Fälle, aber mehr Steuerinformation, Verzögerungen (bei hoher Last) und höherer Aufwand

## Paketvermittlung in Anlehnung an [Mertens]

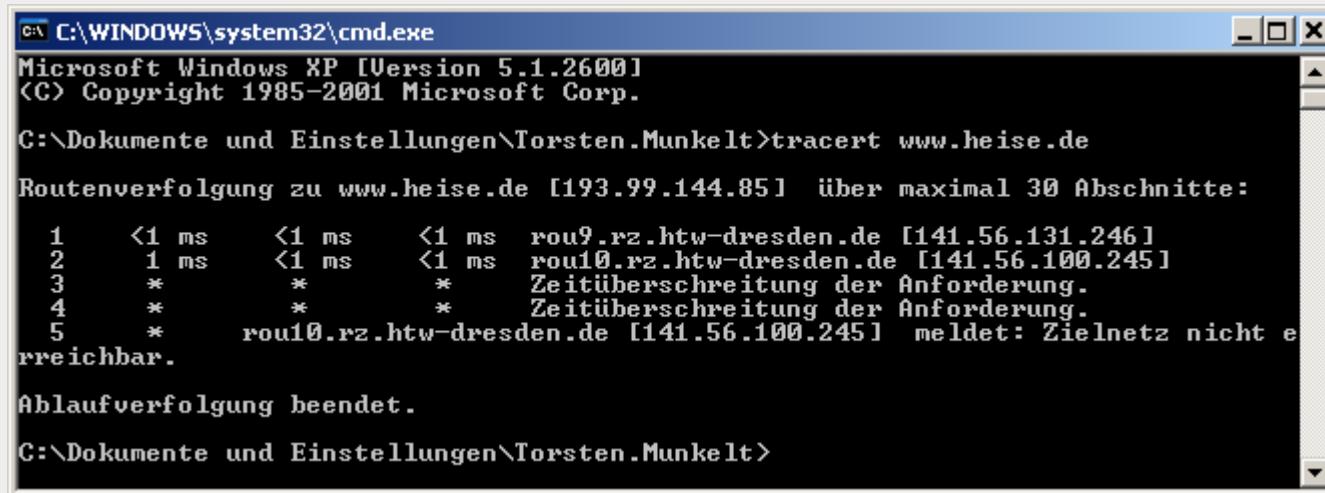
Zerlegen der Nachricht  
in Datenpakete mittels  
TCP

Bewegen der  
Datenpakete  
mittels IP

Sammeln, Ordnen und  
Zusammenfassen der  
Datenpakete durch TCP



## Beispiel für Rout(e)ing (amerikanisch/englisch) - tracert



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Dokumente und Einstellungen\Torsten.Munkelt>tracert www.heise.de

Routenverfolgung zu www.heise.de [193.99.144.85] über maximal 30 Abschnitte:

 1  <1 ms    <1 ms    <1 ms  rou9.rz.htw-dresden.de [141.56.131.246]
 2  1 ms    <1 ms    <1 ms  rou10.rz.htw-dresden.de [141.56.100.245]
 3  *        *        * Zeitüberschreitung der Anforderung.
 4  *        *        * Zeitüberschreitung der Anforderung.
 5  *        rou10.rz.htw-dresden.de [141.56.100.245] meldet: Zielnetz nicht erreichbar.

Ablaufverfolgung beendet.

C:\Dokumente und Einstellungen\Torsten.Munkelt>
```

## Beispiel für Routeing – WWW

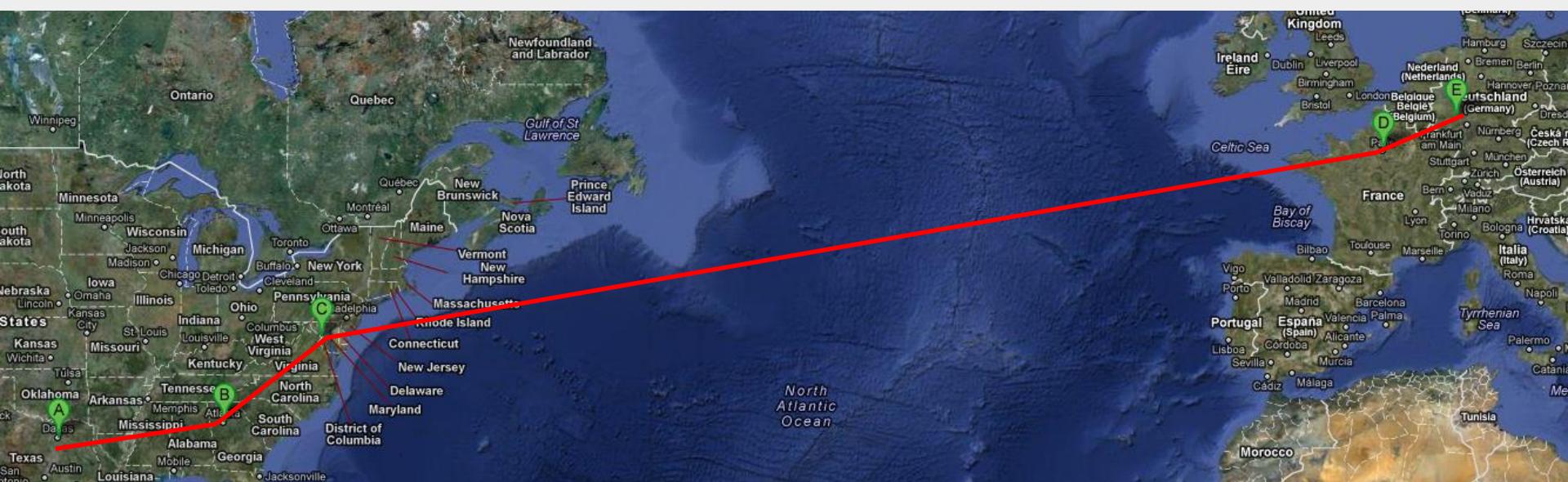
```
TraceRoute to 217.72.200.132 [www.web.de]
```

Hop	(ms)	(ms)	(ms)	IP Address	Host name
1	0	0	0	8.9.232.73	xe-5-3-0.edge3.dallas1.level3.net
2	0	0	0	4.69.145.254	vlan90.csw4.dallas1.level3.net
3	0	0	0	4.69.151.170	ae-93-93.ebr3.dallas1.level3.net
4	20	20	20	4.69.134.22	ae-7-7.ebr3.atlanta2.level3.net
5	33	33	34	4.69.132.86	ae-2-2.ebr1.washington1.level3.net
6	40	33	40	4.69.134.138	ae-81-81.csw3.washington1.level3.net
7	33	34	34	4.69.134.153	ae-82-82.ebr2.washington1.level3.net
8	112	113	114	4.69.137.53	ae-42-42.ebr2.paris1.level3.net
9	120	111	112	4.69.161.58	ae-57-222.csw2.paris1.level3.net
10	114	111	111	4.69.139.235	ae-2-52.edge5.paris1.level3.net
11	247	128	128	212.73.200.54	11-internet.edge5.paris1.level3.net
12	122	122	121	212.227.120.42	te-2-4.bb-d.bap.rhr.de.oneandone.net
13	131	125	124	212.227.121.167	ae-1.bb-c.tp.kae.de.oneandone.net
14	186	122	142	212.227.121.194	ae-4.gw-diste.bs.kae.de.oneandone.net
15	121	121	121	217.72.200.132	<b>web.de</b>

```
Trace complete
```

<http://network-tools.com/default.asp?prog=trace&host=www.web.de>

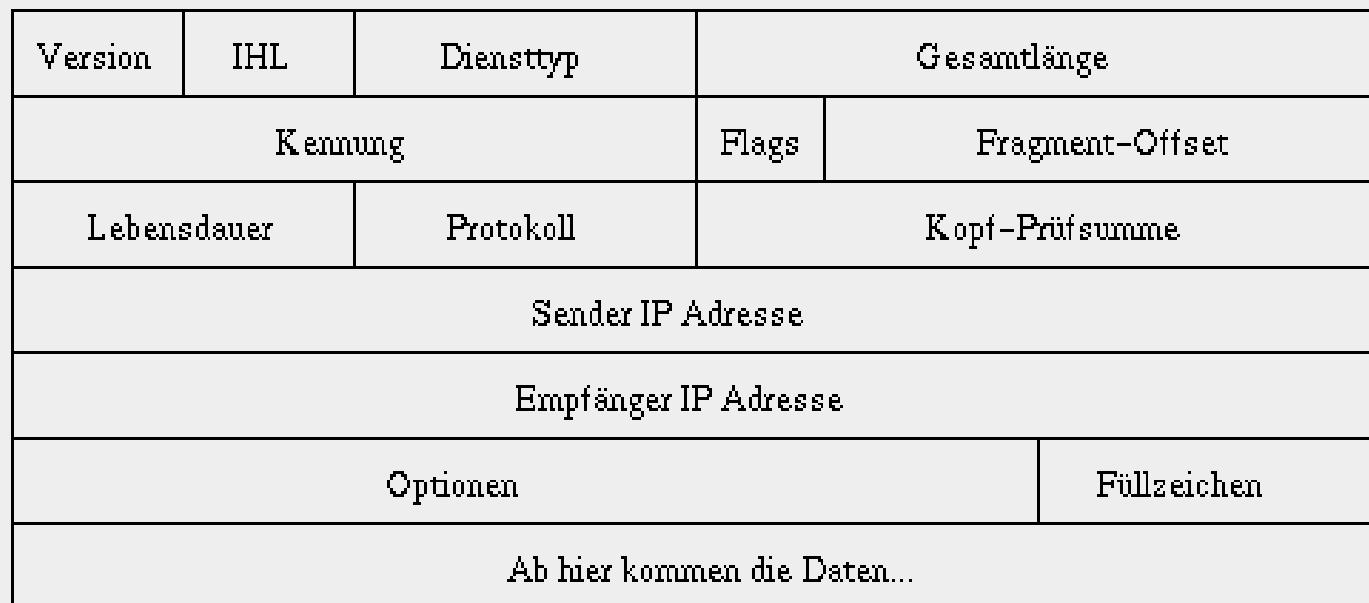
## Route in zirka einer Sekunde



# Aufbau eines IP-Paketes

## Das IP Datagram Format

Bits	0	4	8	12	16	20	24	28	31



[Kalhammer]

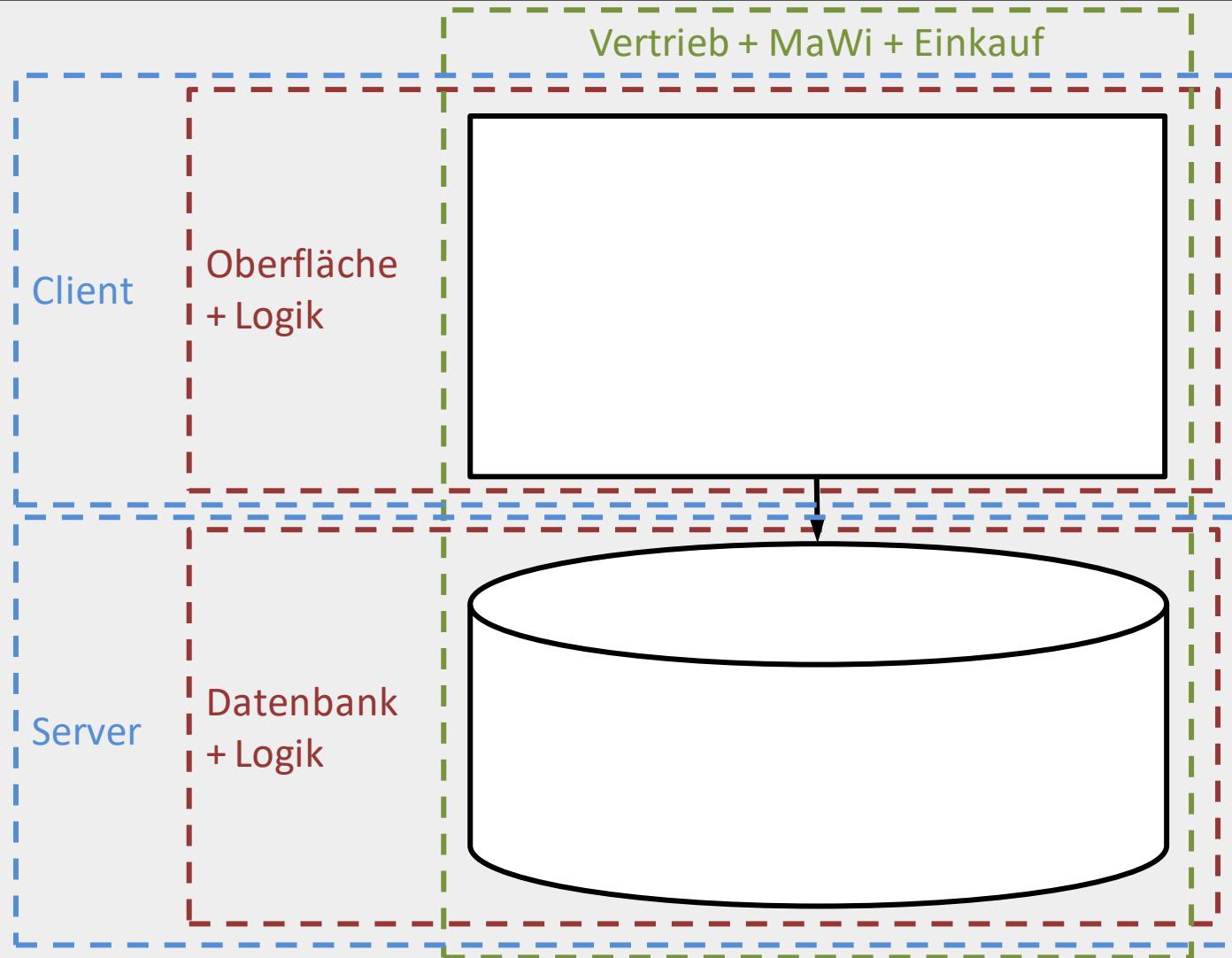
# World-Wide-Web

- Grundlegende Standards:
  - HTTP: Protokoll, wie Browser Informationen vom Webserver anfordert (HTTP-Request und Response)
  - HTML: Dokumentenbeschreibungssprache, die festlegt, wie die Dokumente gegliedert und verknüpft (Hyperlinks) sind
  - URLs: eindeutige Bezeichner von Ressourcen, die in Hyperlinks verwendet werden; Format:  
`<scheme>://<user>:<password>@<host>:<port>/<path>/<file/script>?<searchword1>=<word1>&<searchword2>=<word2>#<fragment>`
- Neuere Standards:
  - CSS: Stil der Elemente einer Webseite definiert
  - HTTPS: TCP  $\leftrightarrow$  SSL  $\leftrightarrow$  HTTP
  - DOM: API für JavaScript (im Webbrowser)
- JavaScript, dynamisches HTML, Java-Applets (application snippet), Server Side Scripting, AJAX, ...

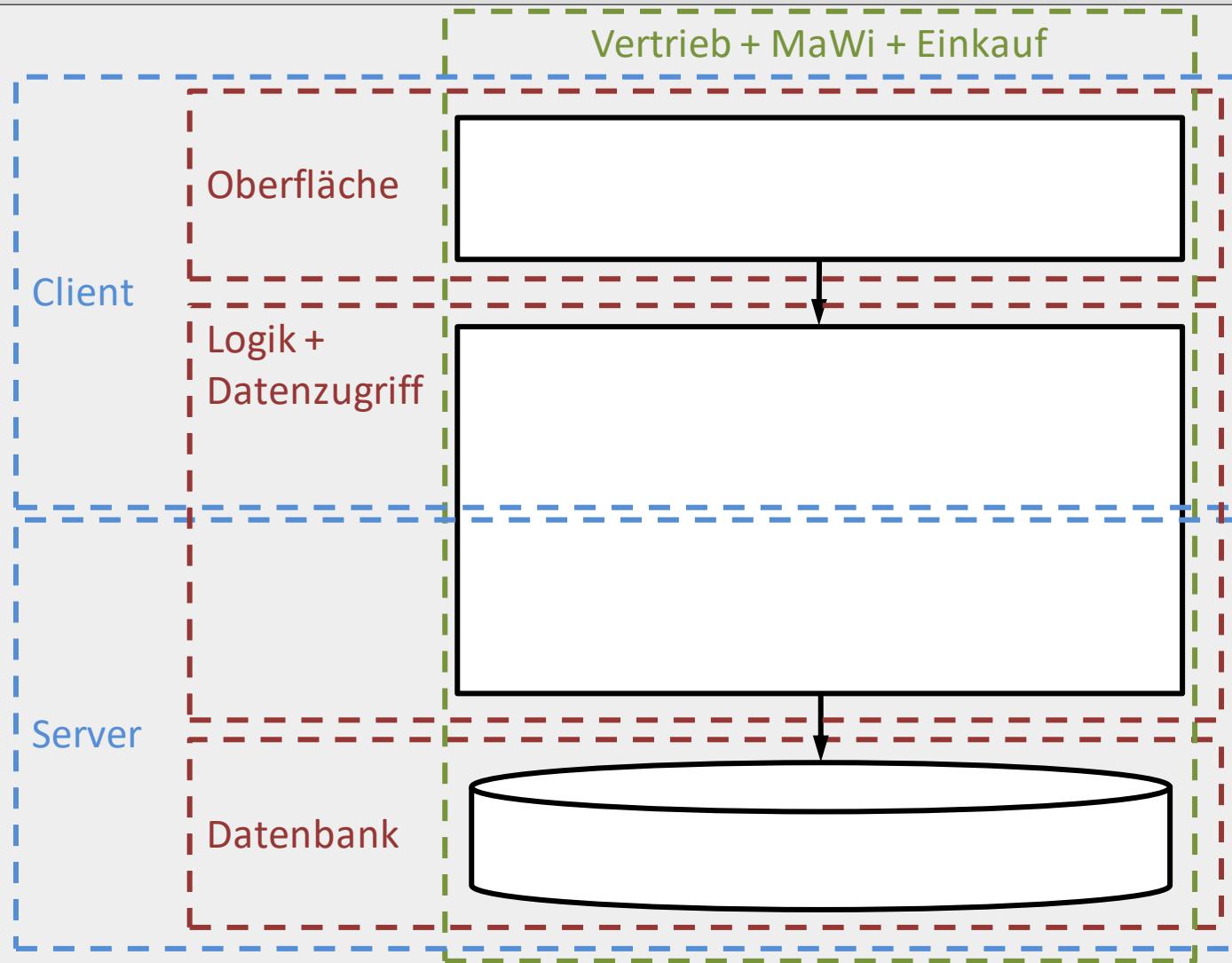
# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- Heuristiken
- Zahlensysteme
- Rechnerarchitektur
- Programmiersprachen
- Rechnernetze
- **Softwarearchitektur**
- Kryptographie

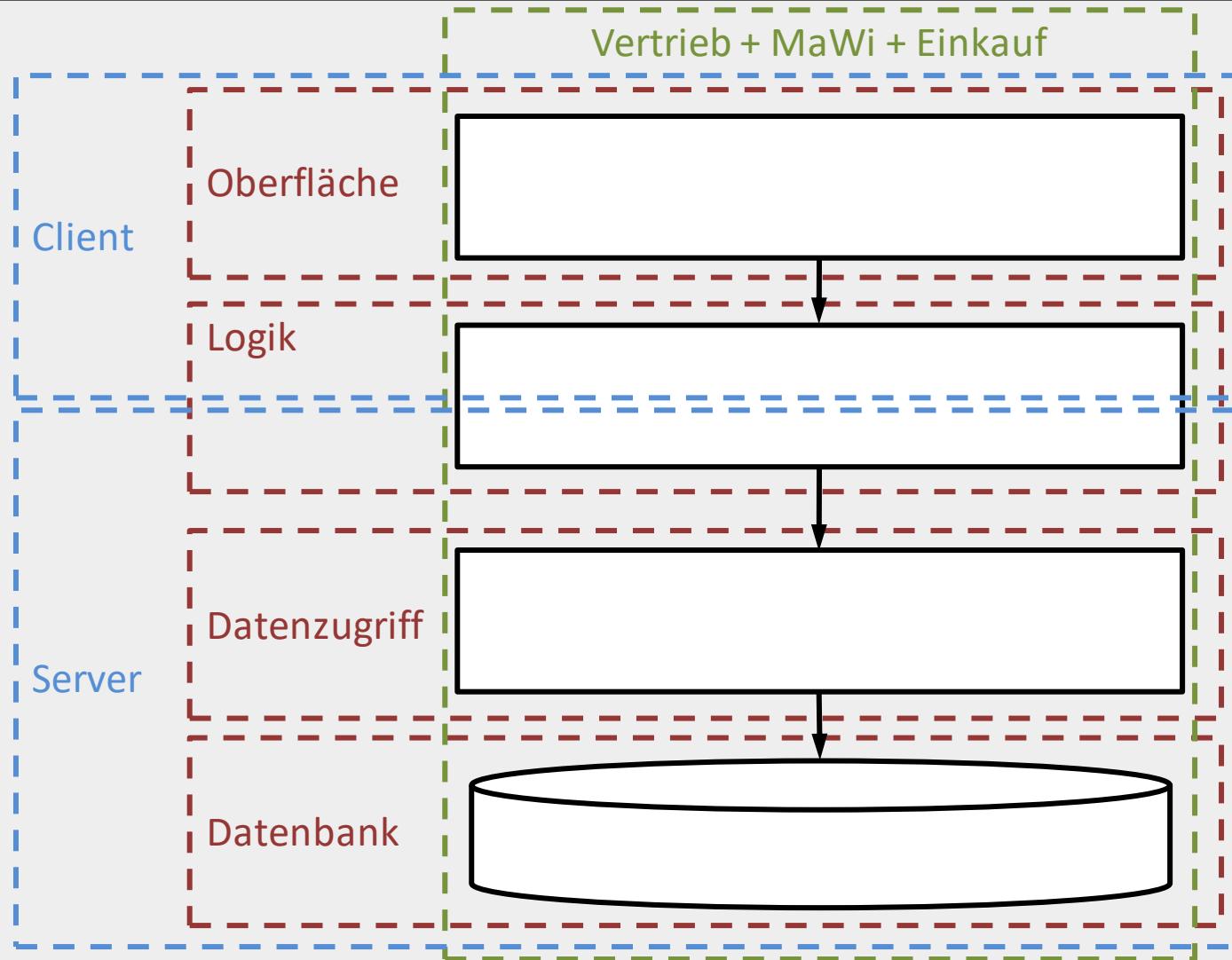
## Client-Server-Architektur – zwei Schichten



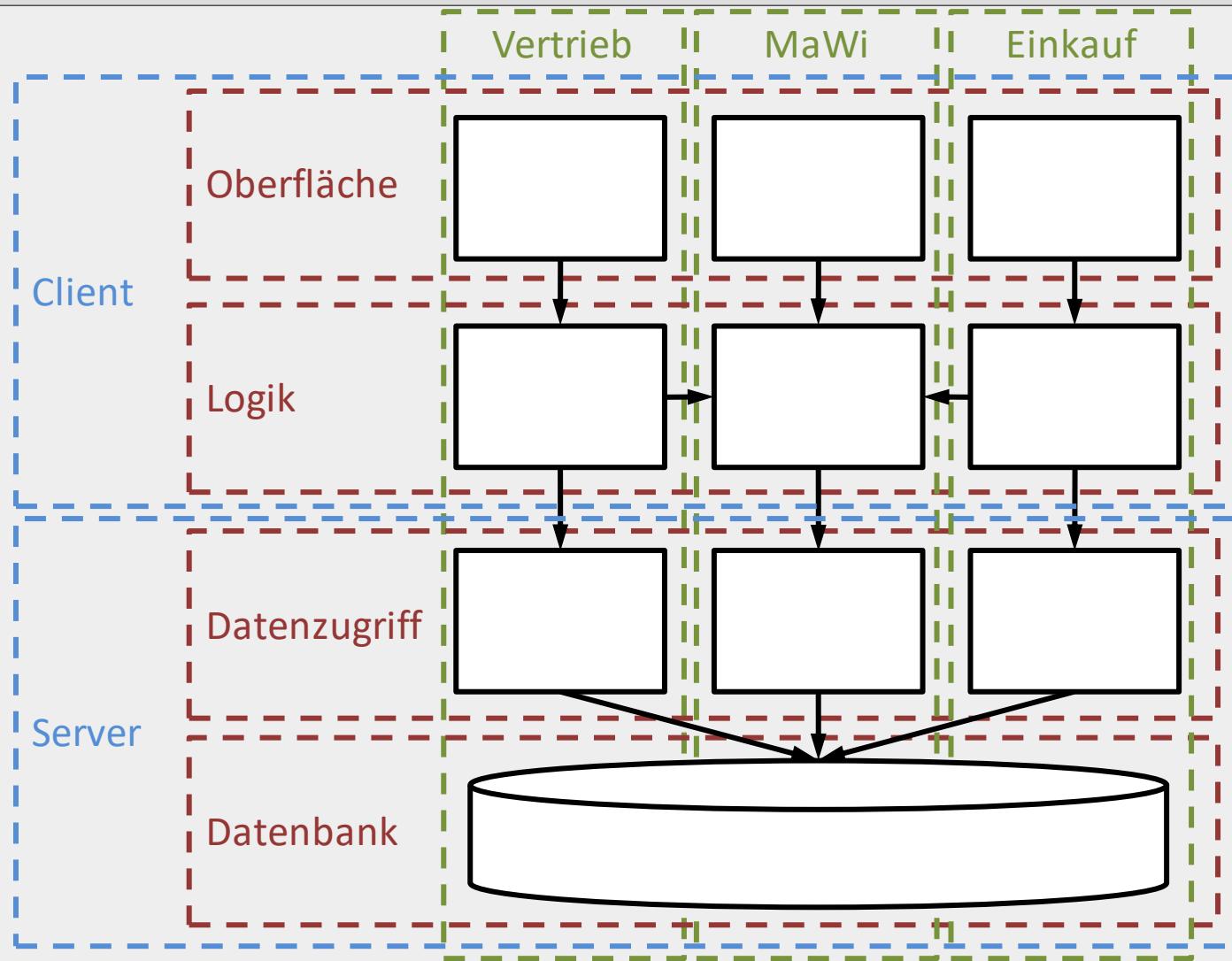
## Client-Server-Architektur – drei Schichten



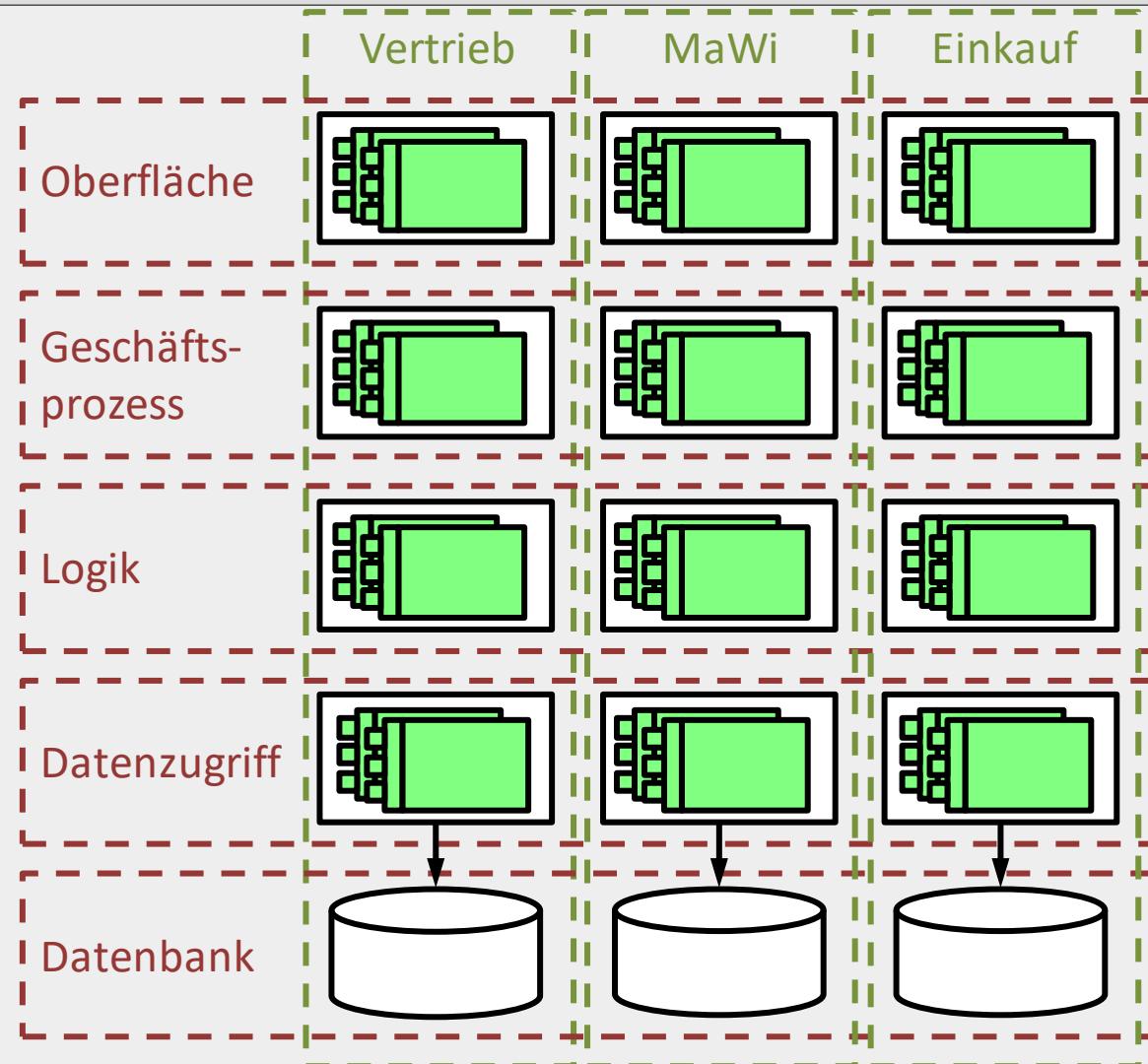
## Client-Server-Architektur – vier Schichten



# Client-Server-Architektur – modulare Untergliederung



# Service-Oriented Architecture (SOA)



# Grundlagen der Informatik und der Informationstechnik

- Berechenbarkeit
- Komplexität
- Heuristiken
- Zahlensysteme
- Rechnerarchitektur
- Programmiersprachen
- Rechnernetze
- Softwarearchitektur
- **Kryptographie**

## Ziele und Anwendungen der Kryptographie

- Ziele:
  - Vertrauliche Übermittlung von Nachrichten
  - Eindeutige Identifikation von Kommunikationsteilnehmern (Authentifizierung)
  - Eindeutige Identifikation von Nachrichtenerstellern (digitale Signatur)
  - Sicherstellen, dass Nachrichten bei der Übertragung nicht modifiziert werden
- Anwendungen:
  - Digitale Wahlen
  - Geheimhaltung der Daten auf Server/Festplatte
  - Vertrauliche Übermittlung der Kreditkartennummer
  - Überprüfen des Absenders einer Bestellung im Internet/WWW
  - ...

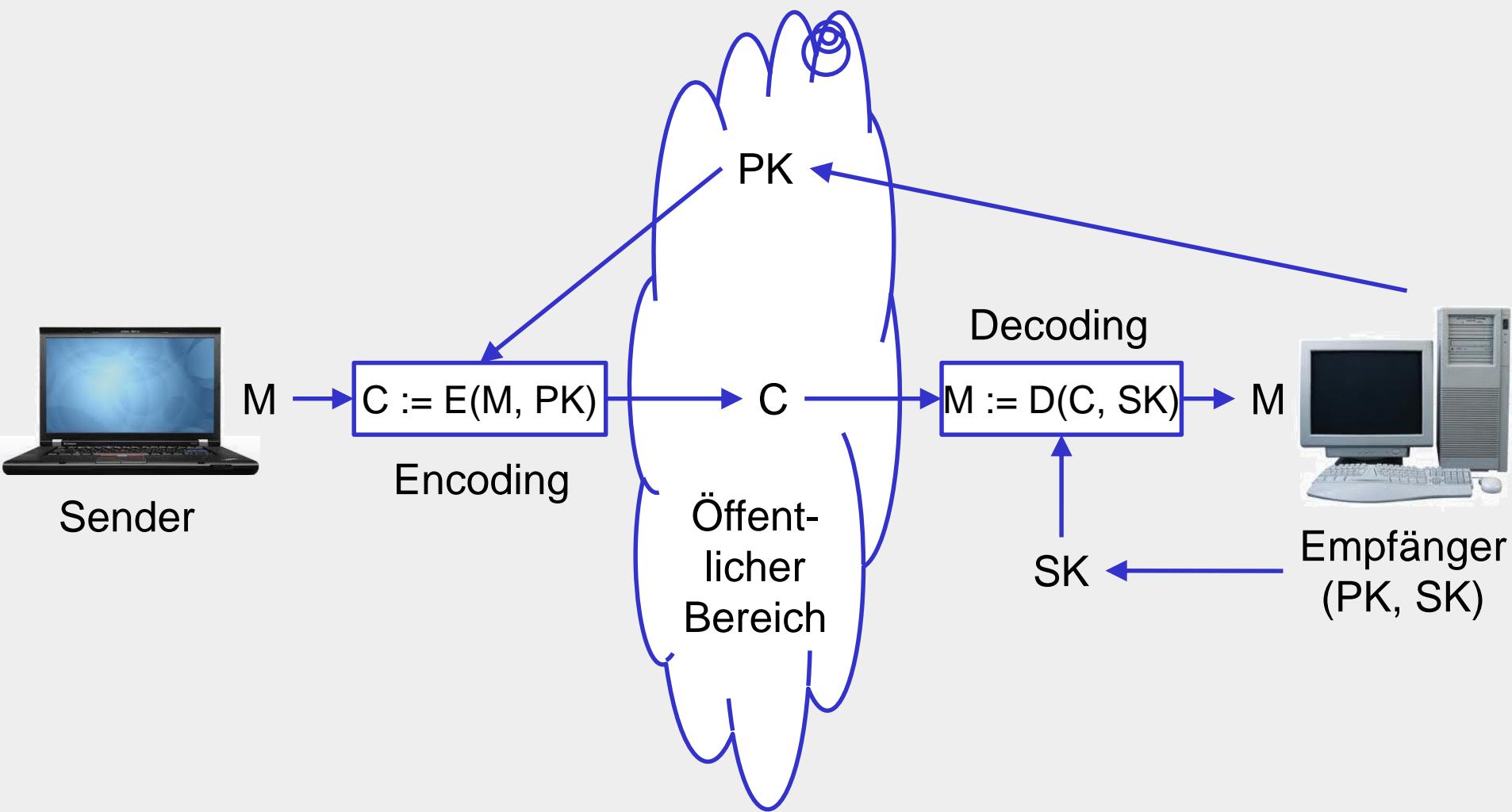
## Symmetrische Verschlüsselung – One-Time-Pad

- Schlüssel (One-Time-Pad): Zufallstext
- Verschlüsselung:  
auf jedes Zeichen (Bit) eines Klartextes modulo ein Zeichen (Bit)  
eines Zufallstextes addiert (XOR) → verschlüsselter Text
- Entschlüsselung:  
von jedem Zeichen (Bit) des verschlüsselten Textes modulo ein  
Zeichen des Schlüssels subtrahiert → Klartext
- Nicht entschlüsselbar, wenn:
  - Zufälliger Schlüssel (nicht Pseudo-Zufallszahl)
  - Schlüssel so lang wie oder länger als der Klartext
  - Schlüssel niemals wiederverwendet
  - Schlüssel für andere geheim

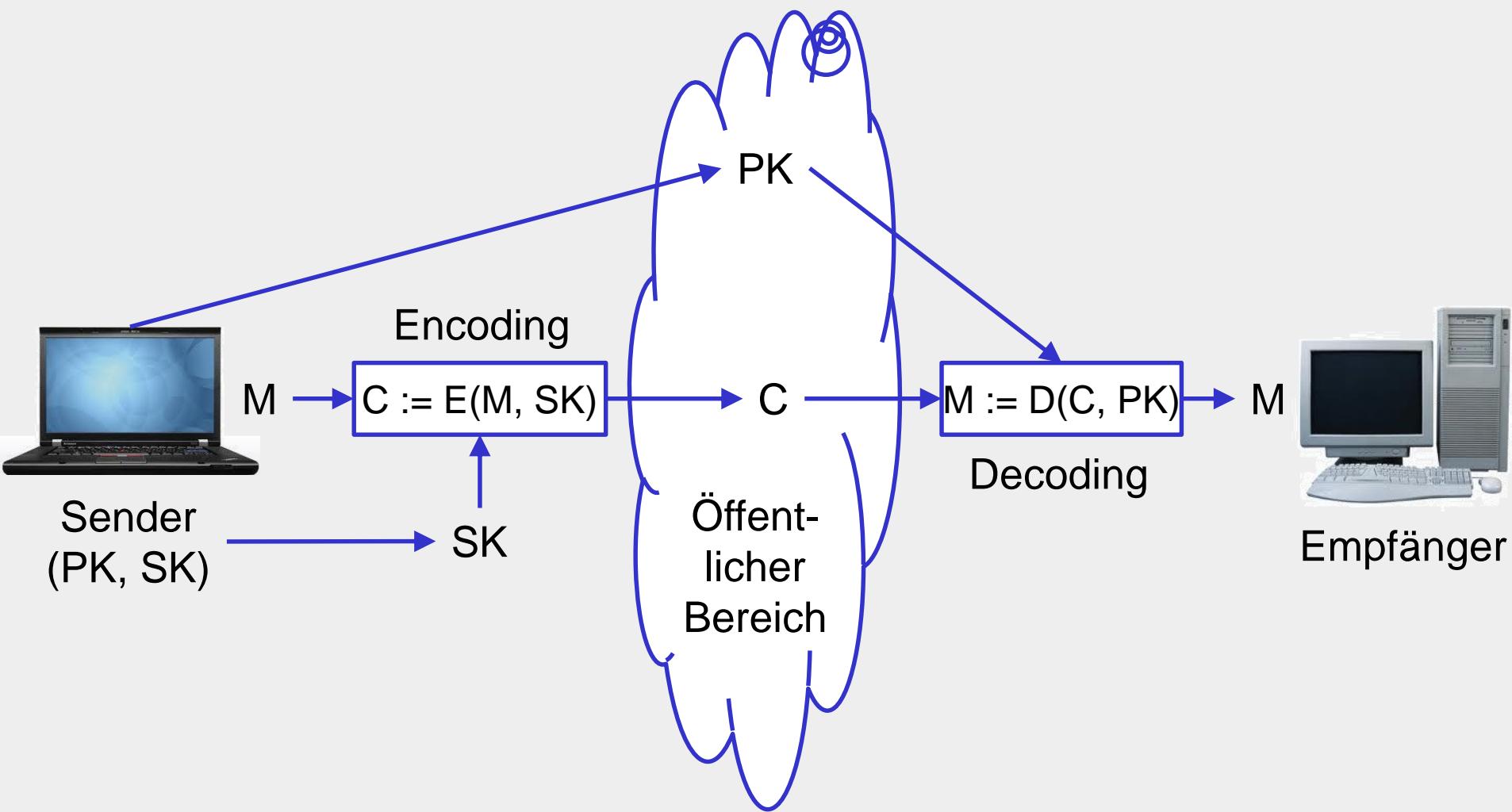
## Asymmetrische Verschlüsselung

- Grundlage: „Falltürfunktion“ (one-way-function)  $f$ :
  - $f$  schnell zu berechnen
  - $f^{-1}$  ohne Geheimnis („Falltür“) nicht oder nur sehr aufwendig zu berechnen
  - PK entspricht Funktion  $f$ ,
  - SK entspricht dem Geheimnis („Falltür“).
- Ablauf:
  - Empfänger verfügt über PK und SK.
  - Empfänger veröffentlicht PK.
  - (Sender verfügt für Empfang auch über PK1 und SK1.)
  - Sender verschlüsselt Nachricht M mit Verfahren E unter Zuhilfenahme des PKs des Empfängers:  $C := E(M, PK)$ .
  - Empfänger entschlüsselt die verschlüsselte Nachricht C unter Zuhilfenahme seines SKs:  $M = D(C, SK)$ .

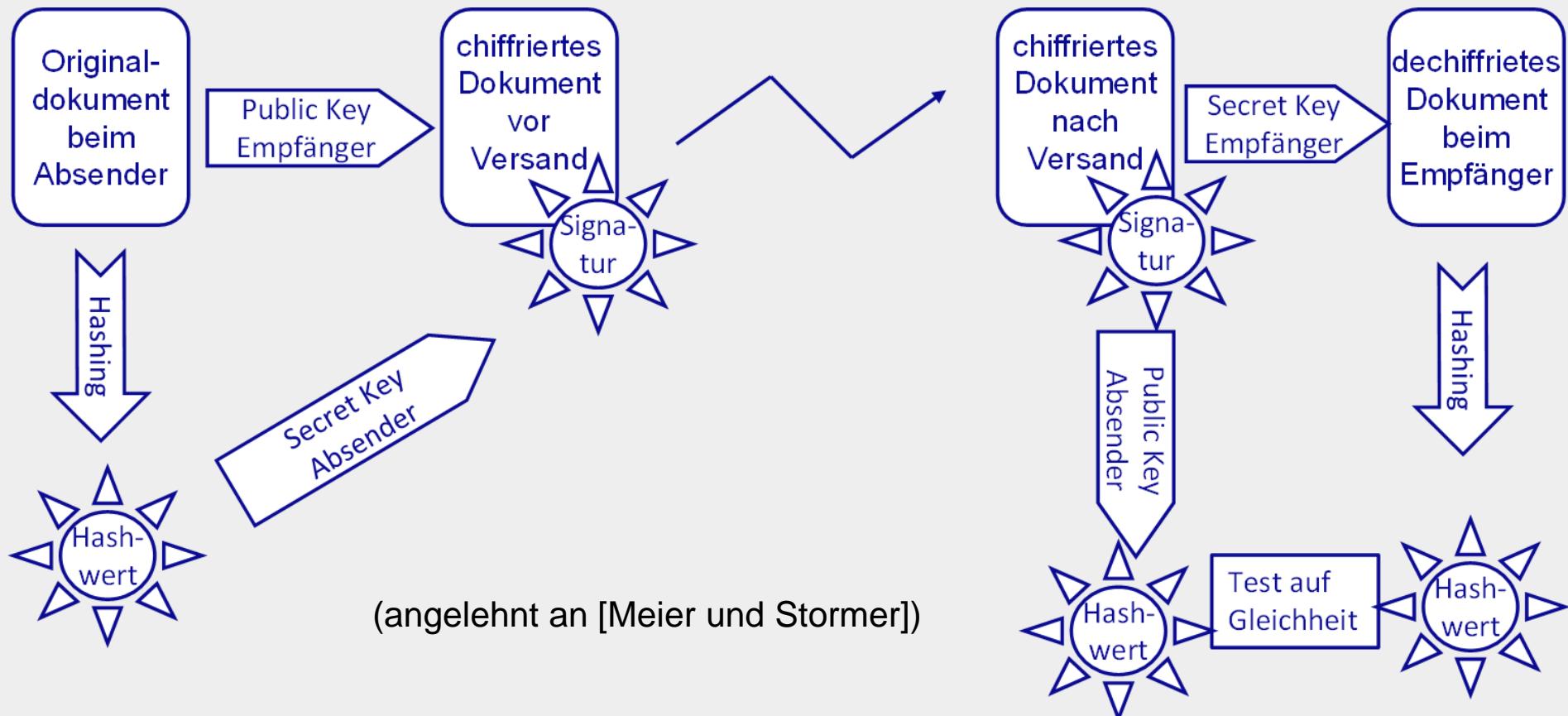
## Asymmetrische Verschlüsselung – grafisch



## Authentifizierung (asymmetrisch)



# Digitale Signatur



## RSA-Verfahren

1. zwei große Primzahlen  $p$  und  $q$  festlegen
2. Zahl  $e$  festlegen, die mit  $(p - 1) * (q - 1)$  keine gemeinsamen Faktoren besitzt („relativ prim“)
3.  $n := p * q$
4. Öffentlicher Schlüssel:  $(n, e)$
5.  $d$  ermitteln, für das gilt:  $(d * e) \bmod ((p - 1) * (q - 1)) = 1$
6. Privater Schlüssel:  $(n, d)$
7. Kodieren:  $c := m^e \bmod n$
8. Dekodieren:  $m := c^d \bmod n$



Heute ← 1977



## Gliederung

---

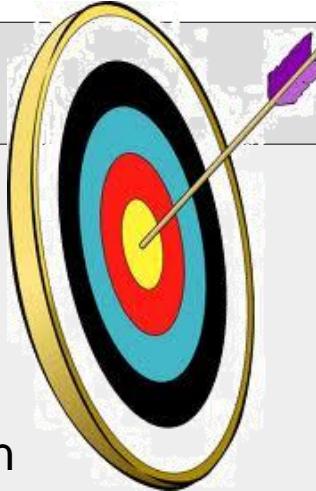
1. Inhalte und Aufgaben der Wirtschaftsinformatik
2. Grundlagen der Informatik und der Informationstechnik
- 3. Informationsmanagement**
4. Modellierung
5. Datenbanken
6. Softwareentwicklung
7. Betriebliche Informationssysteme

## Unternehmerische Fragestellungen

1. Sollte ich eine weitere Produktionsstätte/Vertriebsniederlassung gründen/eröffnen? Wo sollte ich sie gründen/eröffnen? Ist es sinnvoll, eine Produktionsstätte/Vertriebsniederlassung in China zu gründen/eröffnen?
2. Sollte ich meine Produktpalette um Produkte bzw. Konfigurationen erweitern? Um welche Produkte bzw. Konfigurationen sollte ich sie erweitern? Ist es sinnvoll, ein Taschenmesser auch mit Keramikklingen anzubieten?
3. Sollte ich die Verkaufspreise meiner Produkte verändern, um größeren Umsatz und höheren Gewinn zu erzielen? Bei welchen Produkten sollte ich den Preis wie verändern, um Umsatz und Gewinn zu erhöhen?



## Entscheidung(sprozess)



- Festlegen der Ziele
- Entscheidungsprozess
  1. Erkennen des Entscheidungsbedarfs
  2. Generieren von Handlungsalternativen
  3. Prognostizieren der Auswirkungen der Alternativen im Hinblick auf die zu erreichenden Ziele
  4. Auswählen der besten Alternative hinsichtlich der zu erreichenden Ziele (Entscheidung i. e. S.)
  5. Umsetzen dieser Alternative
  6. Kontrollieren der Auswirkungen der Alternative auf die Ziele
- Rücksprünge im Entscheidungsprozess möglich

## Aufgaben des Informationsmanagements i. e. S.

1. *Effizientes* Bereitstellen von Information für betriebliche Entscheidungen
  - Im richtigen Umfang
  - Im richtigen Aggregationsgrad
  - Zur richtigen Zeit
  - An der richtigen Stelle (dem richtigen Adressaten)
2. *Effizientes* Bereitstellen von Systemen zur Unterstützung betrieblicher Entscheidungen  
(Entscheidungsunterstützungssysteme)



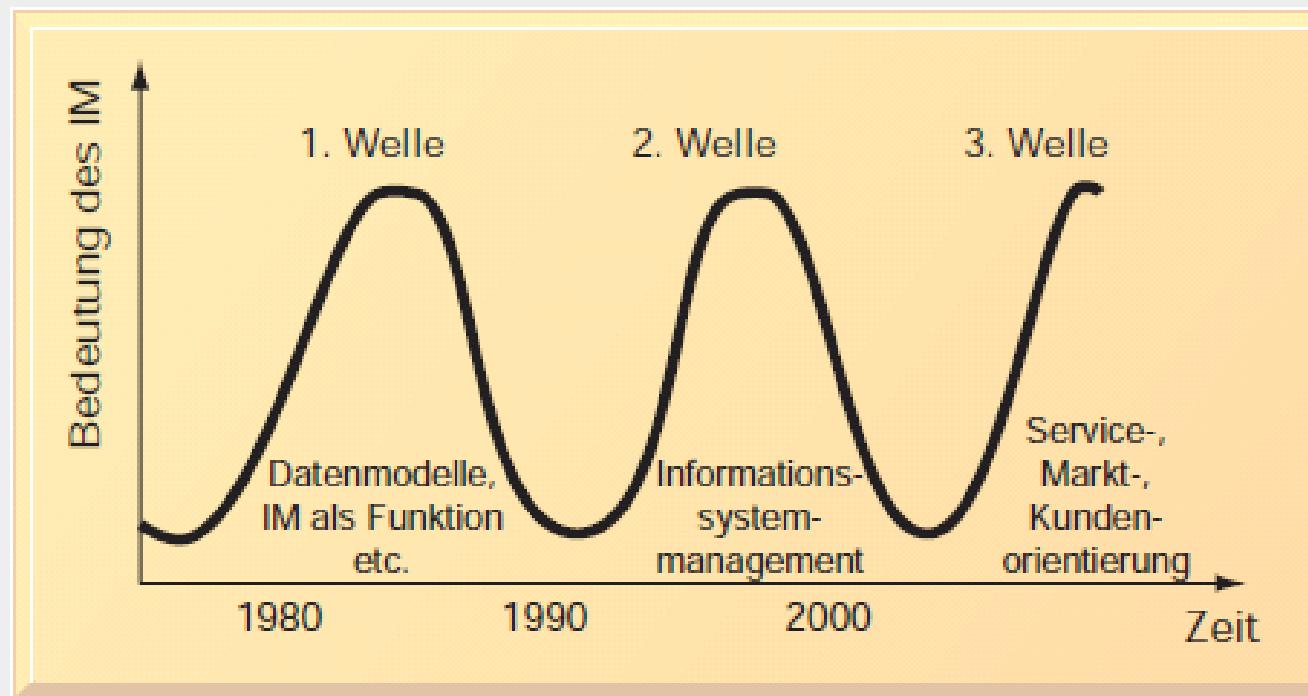
## Aufgaben des IM i. w. S. [Kudraß]

- Strategische Aufgaben
  - Festlegen der Ziele und strategischen Maßnahmen
  - Qualitätsmanagement
  - Controlling und Revision
- Administrative Aufgaben
  - **Projektmanagement**
  - **Geschäftsprozessmanagement**
  - **Daten- und Wissensmanagement**
  - Vertragsmanagement
  - **Sicherheitsmanagement**
- Operative Aufgaben
  - Produktionsmanagement
  - Problemmanagement

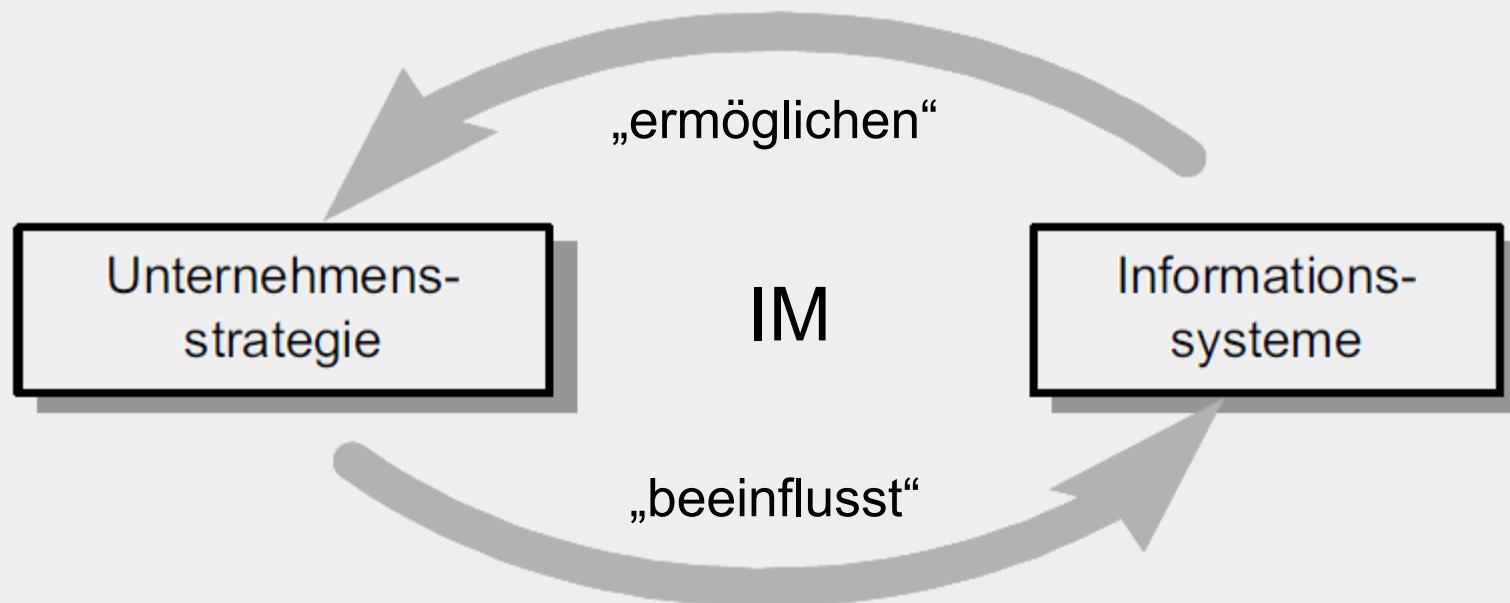
## Informationsmanagement – „fachliche“ Definition [Voß]

- „.... ist die wirtschaftliche [...]
  - Planung
  - Beschaffung,
  - Verarbeitung,
  - Allokation [Zuordnung zu den Nachfragern],
  - Distribution [Verteilung an die Nachfrager]
- [...] von Informationen [...] zur Unterstützung von [...] Entscheidungsprozessen [...].

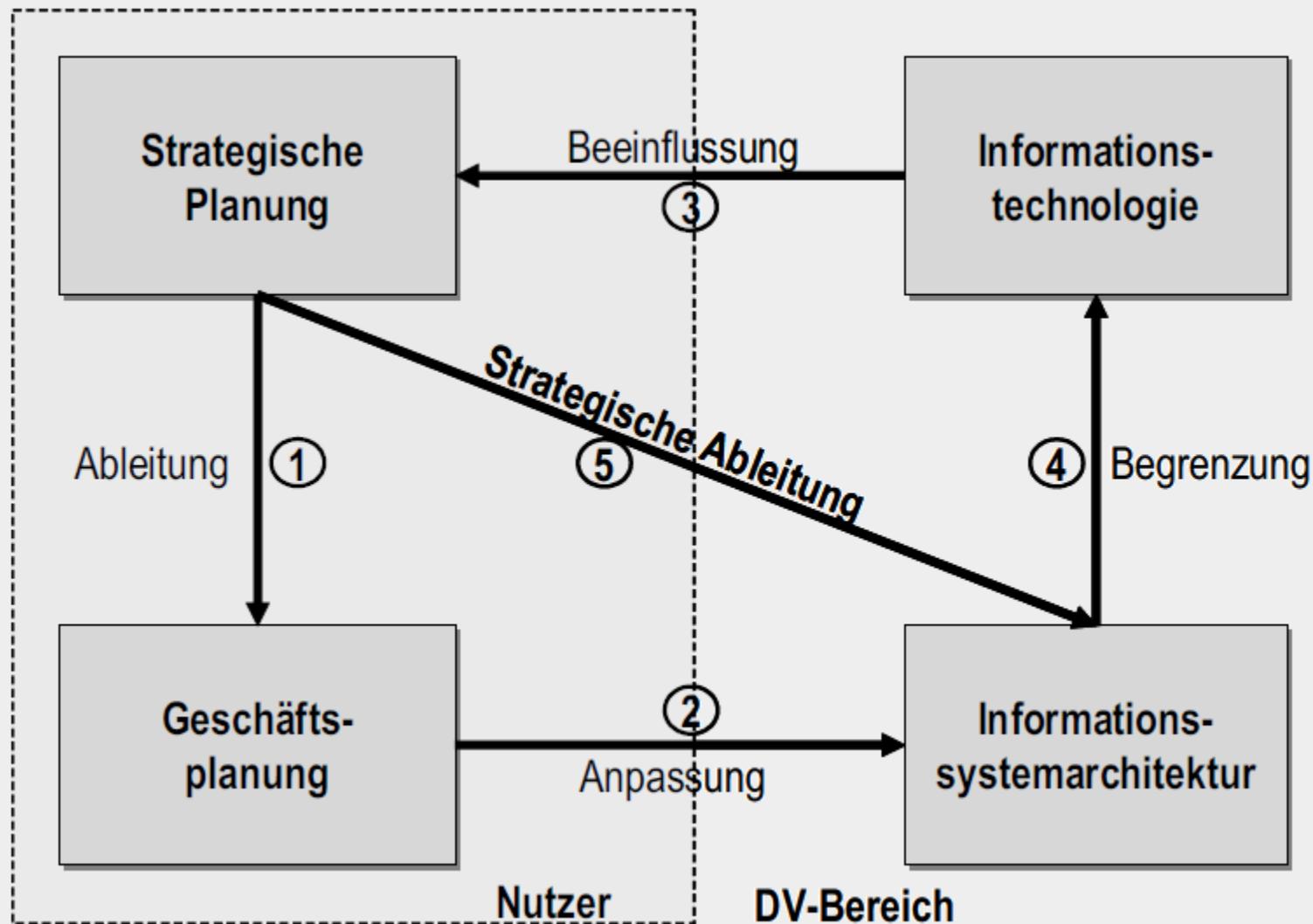
## Phasen des IMs [Laudon et al.; Zarnekow et al.]



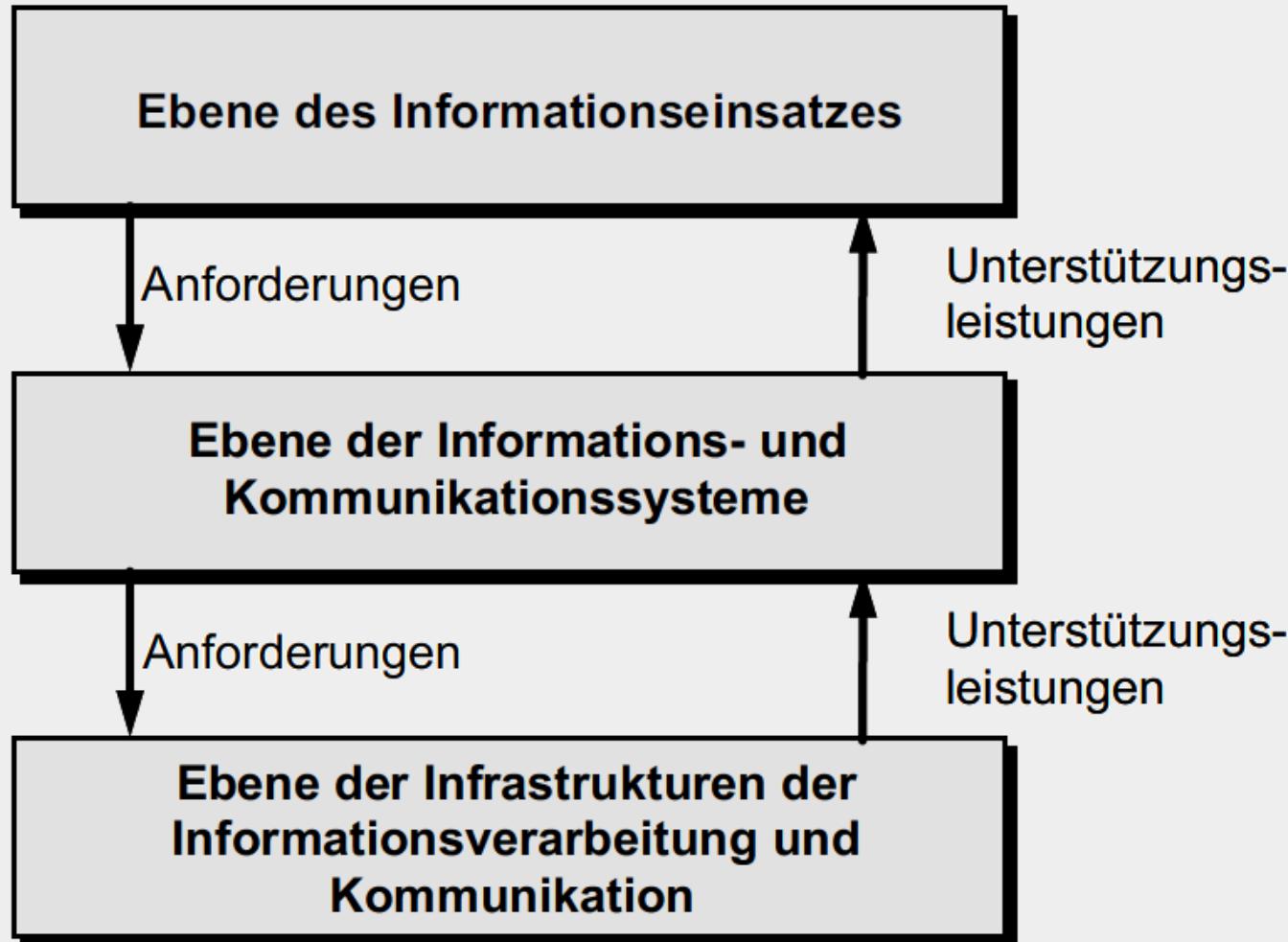
## IM zwischen Unternehmensstrategie und BISen [Krcmar]



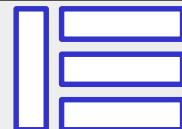
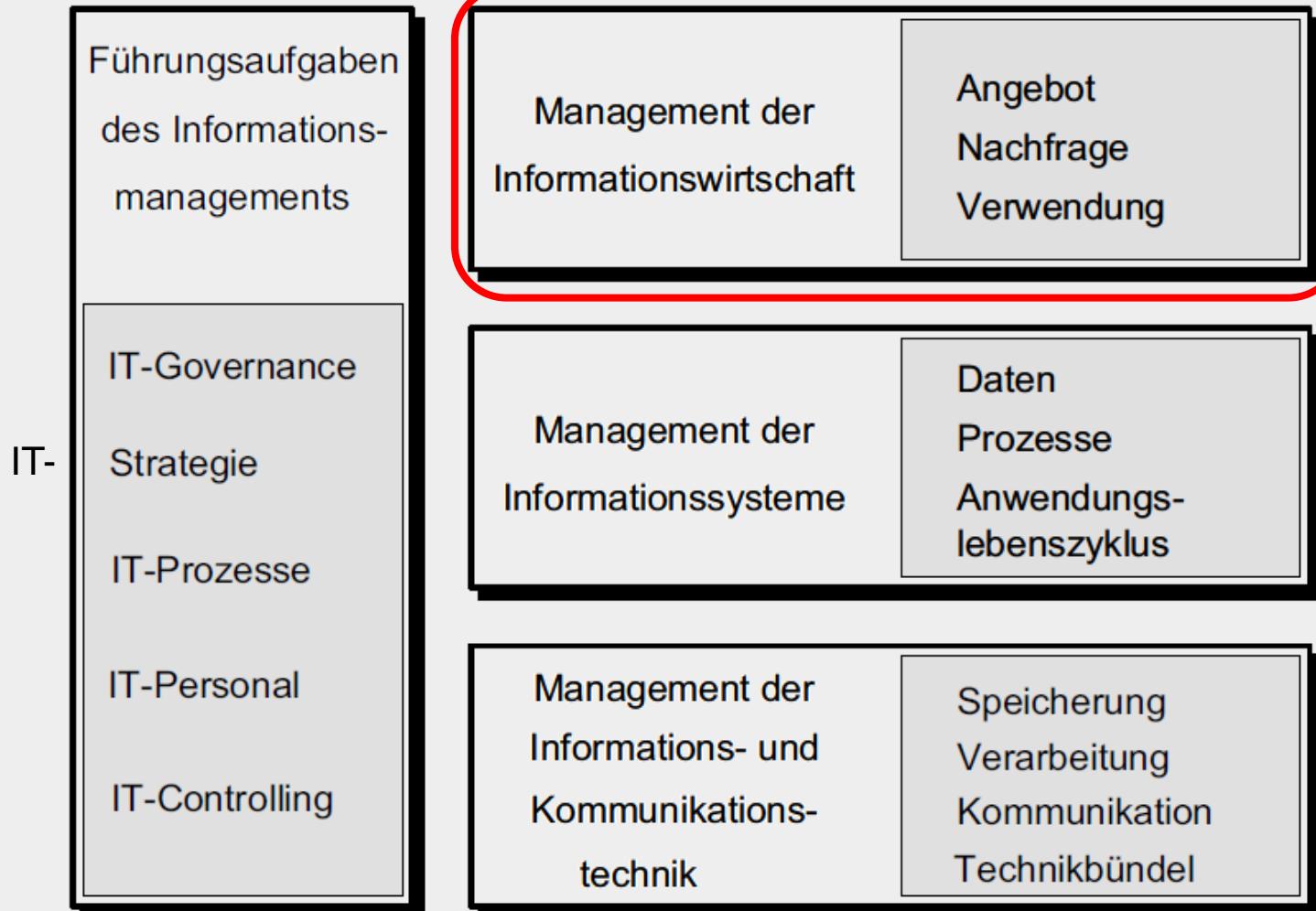
# Einflüsse zwischen Geschäft und Technologie [Parker et al.]



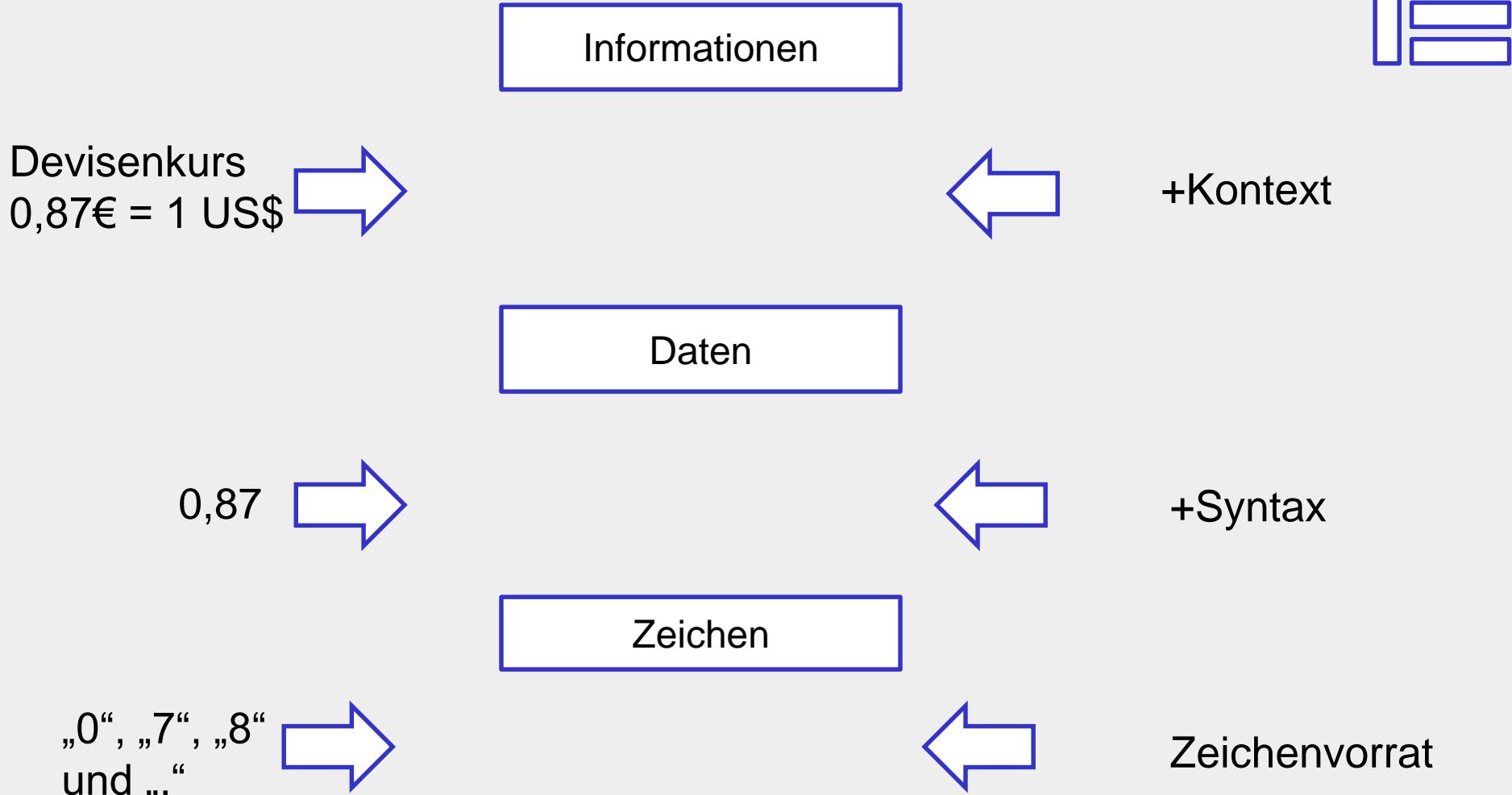
## Ebenen des IM [Wollnik]



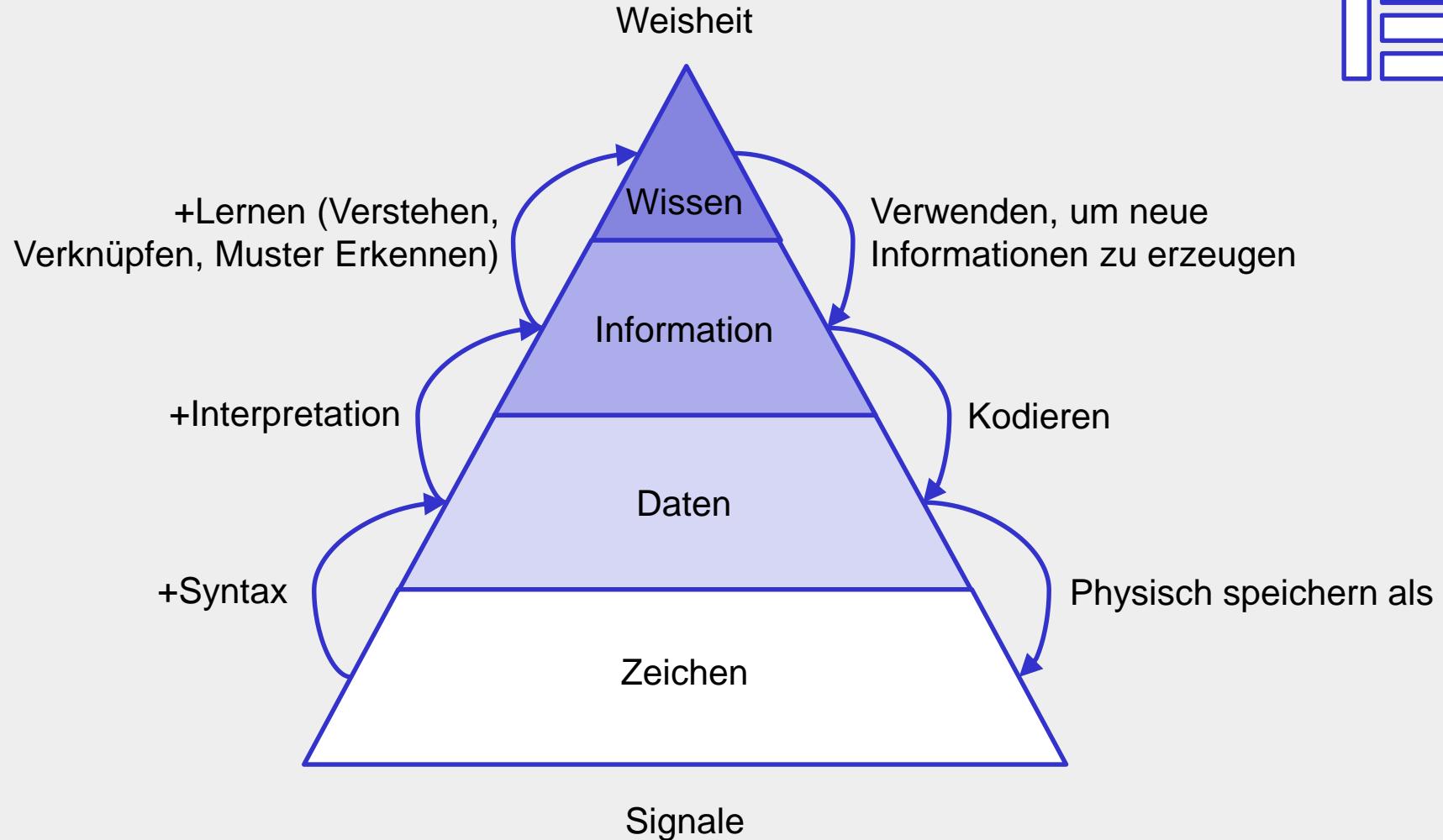
## „Referenzmodell des IM“ [Krcmar]



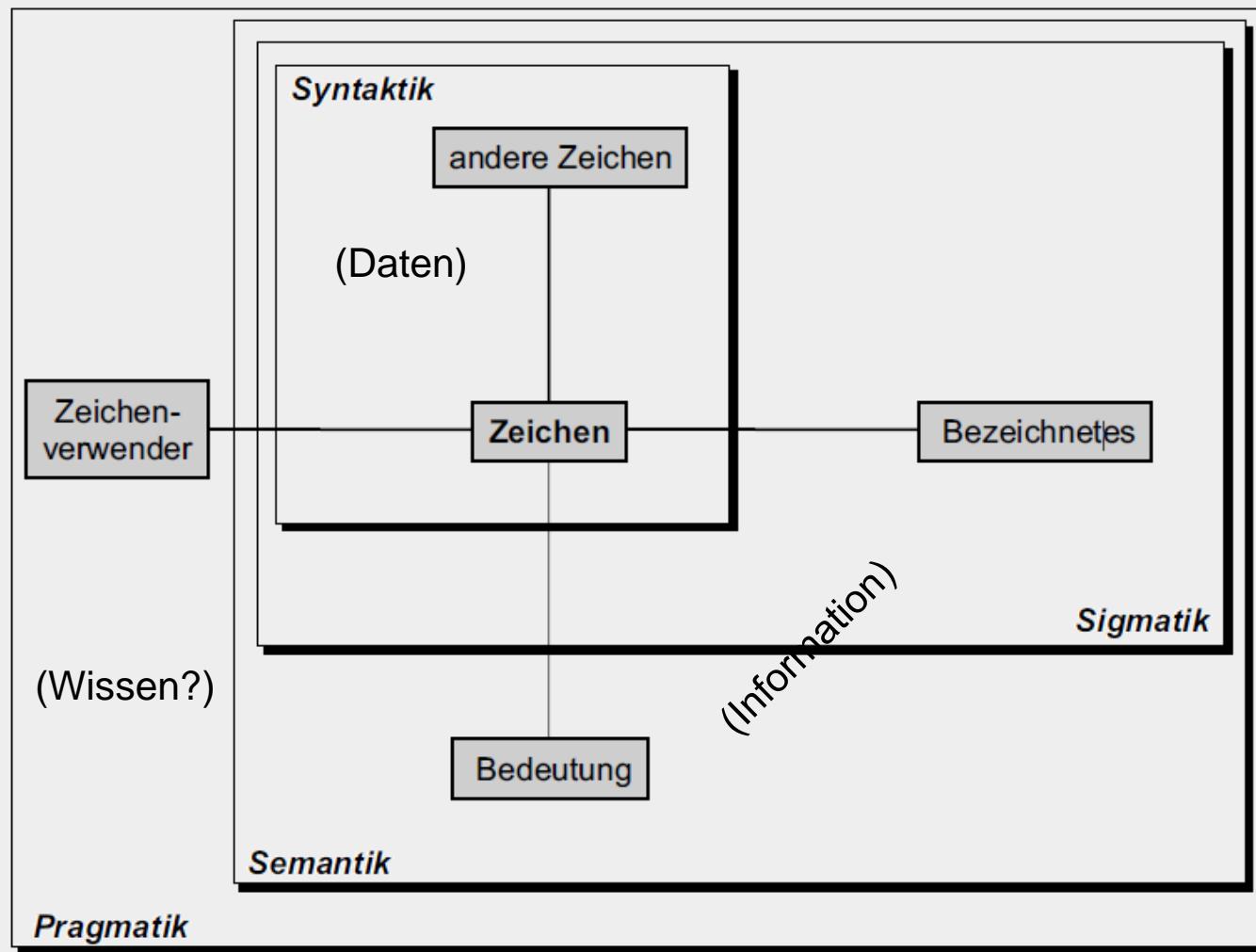
## Zeichen, Daten, Information [Rehhäuser et al.]



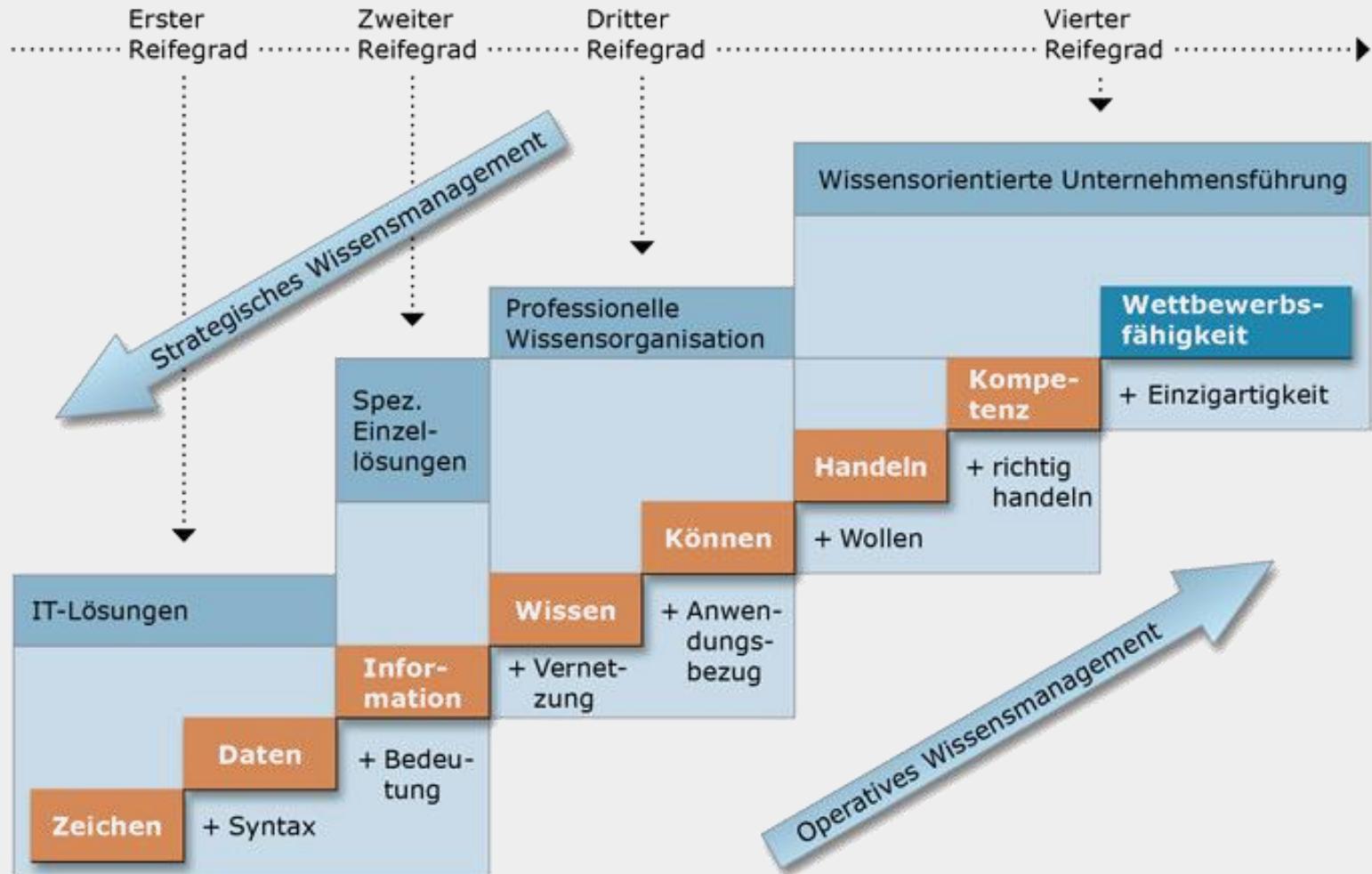
# Zeichen, Daten, Information, Wissen; in Anlehnung an [Fink]



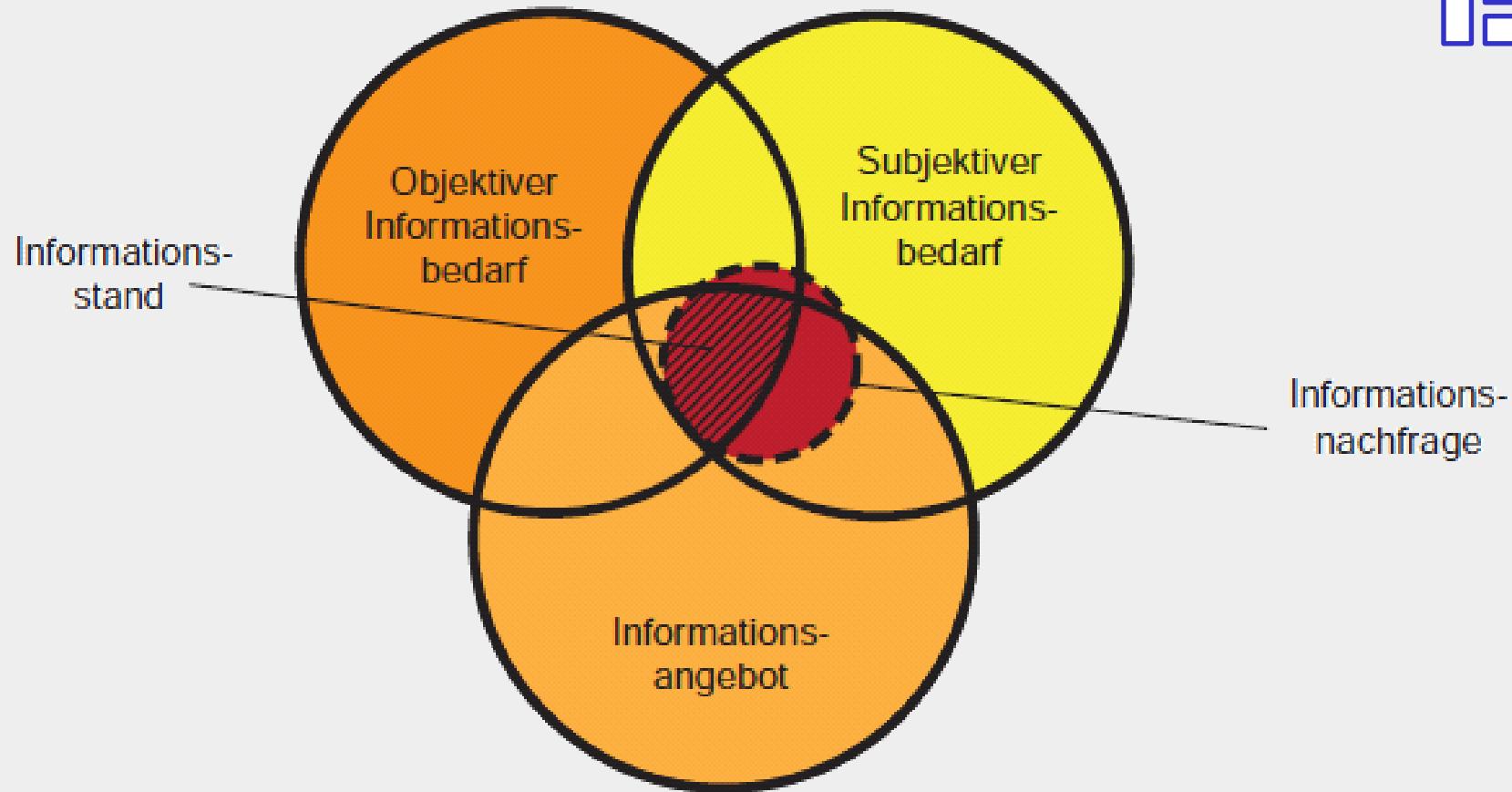
# Zeichen, Daten u. Information aus Sicht der Semiotik [Berthel]



# Wissenstreppe nach North

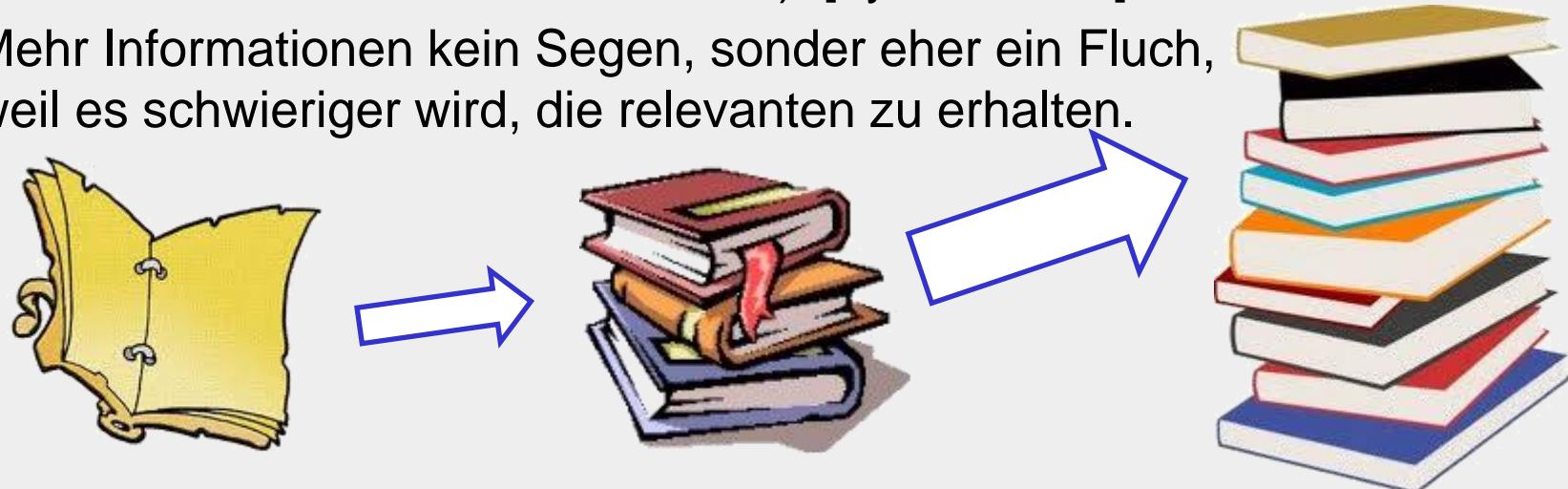


## Informationsangebot und –bedarf [Picot et al.; Laudon et al.]



## Zunehmende Informationsfülle – „Informationsflut“

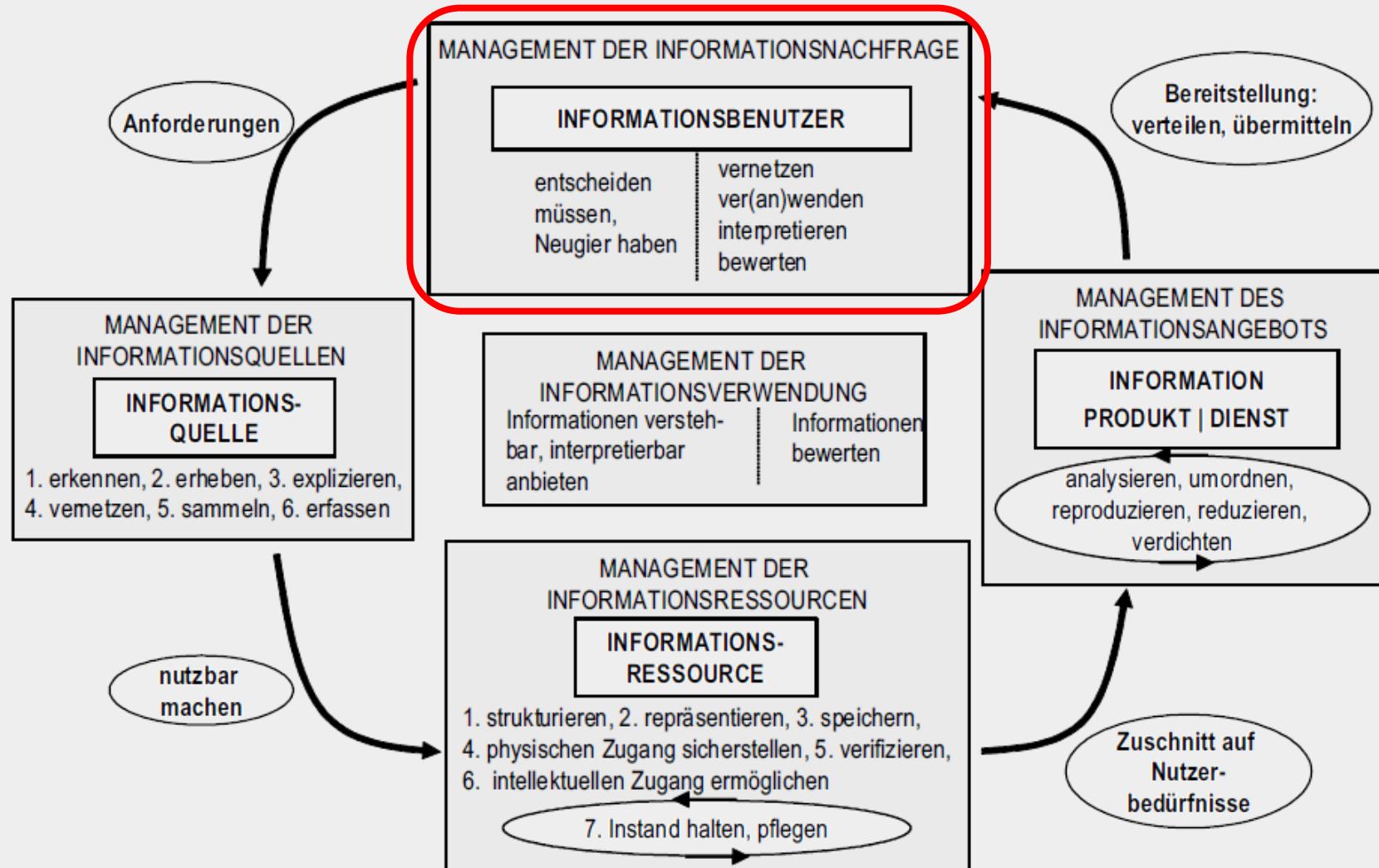
- „Der Mensch kann nur fünf bis neun Informationen gleichzeitig verarbeiten.“ [George A. Miller, 1956]
- Verdopplung des gedruckten Wissens alle acht Jahre (exponentielles Wachstum!)
- Die 2002 neu produzierte Menge an Information entspricht zirka 37.000 Bibliotheken in der Größe der „Library of Congress“ (größte Bibliothek der Welt mit zirka 136 TB). [Lyman et al.]
- Mehr Informationen kein Segen, sondern eher ein Fluch, weil es schwieriger wird, die relevanten zu erhalten.



## Ziele der Informationslogistik [Augustin]

#	Ziel: Bereitstellung	Erklärung
1	der richtigen Information	vom Empfänger [objektiv] benötigt und [objektiv] verstanden
2	zum richtigen Zeitpunkt	für die Entscheidung ausreichend; nicht zu spät, aber auch nicht viel zu früh
3	in der richtigen Menge	so viel wie nötig und so wenig wie möglich
4	am richtigen Ort	beim Empfänger verfügbar
5	in der erforderlichen Qualität	wahr, ausreichend detailliert/aggregiert, unmittelbar verwendbar

# Lebenszyklus der Informationswirtschaft [Krcmar; Rehäuser]

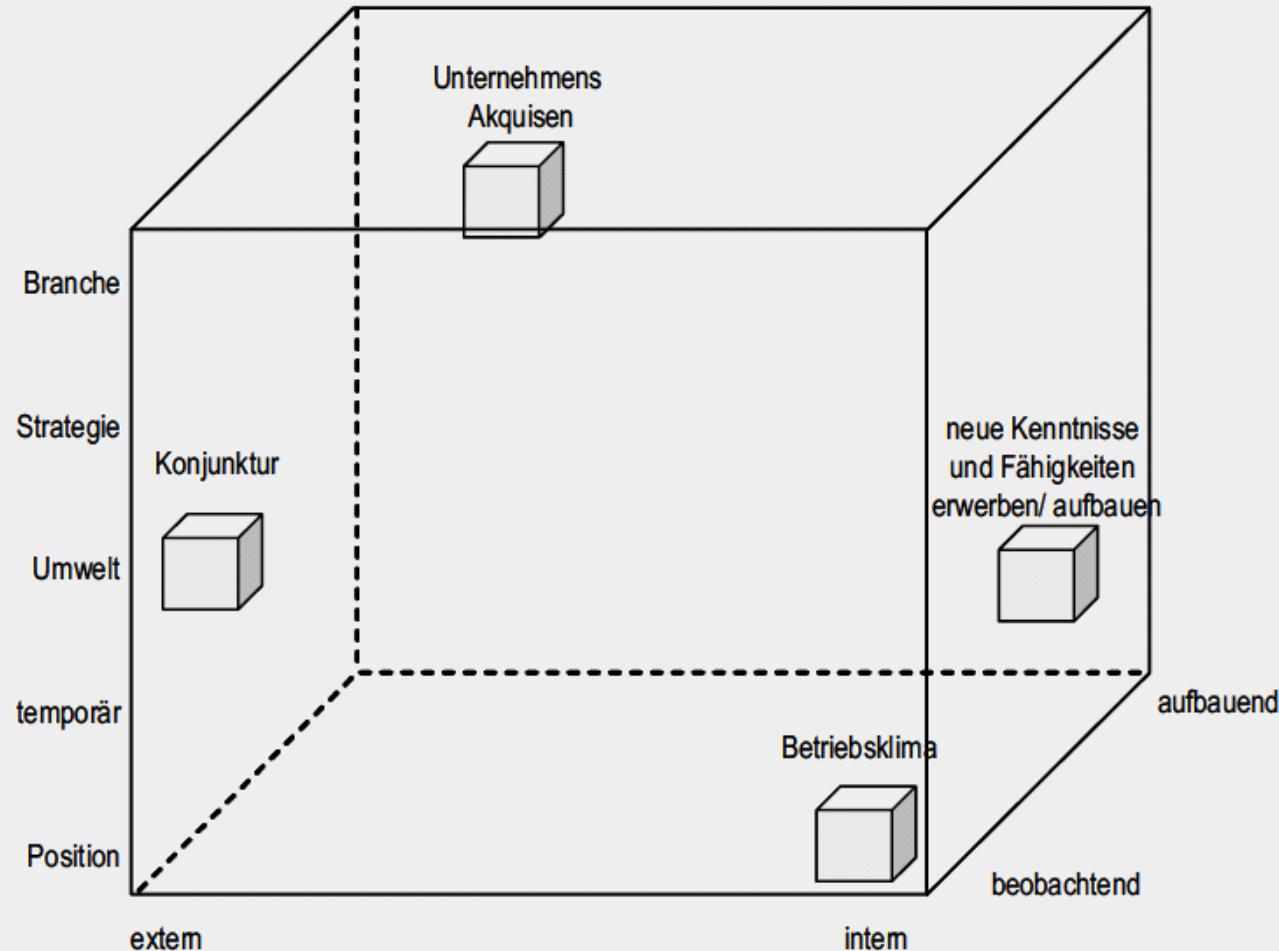


## Ermittlung des Informationsbedarfs [Schneider; Voß]

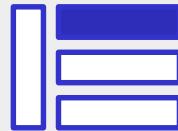


Subjektive Verfahren	Objektive Verfahren	Gemischte Verfahren
Ableitung aus subjektiver Interpretation der Aufgabe	Ableitung aus einer intersubjektiv validierten Interpretation der Aufgabe	Vorgabe theoretischer Raster, die subjektiv interpretiert werden
<ul style="list-style-type: none"><li>• Offene Befragung</li><li>• Wunschkataloge</li><li>• Befragung der Mitarbeiter im Tätigkeitsumfeld</li></ul>	<ul style="list-style-type: none"><li>• Prozessanalyse</li><li>• Entscheidungsanalyse</li><li>• ...</li></ul>	<ul style="list-style-type: none"><li>• Methode der kritischen Erfolgsfaktoren</li><li>• Balanced Scorecard</li><li>• ...</li></ul>

# Dimensionen der Kritischen Erfolgsfaktoren [Rockart]

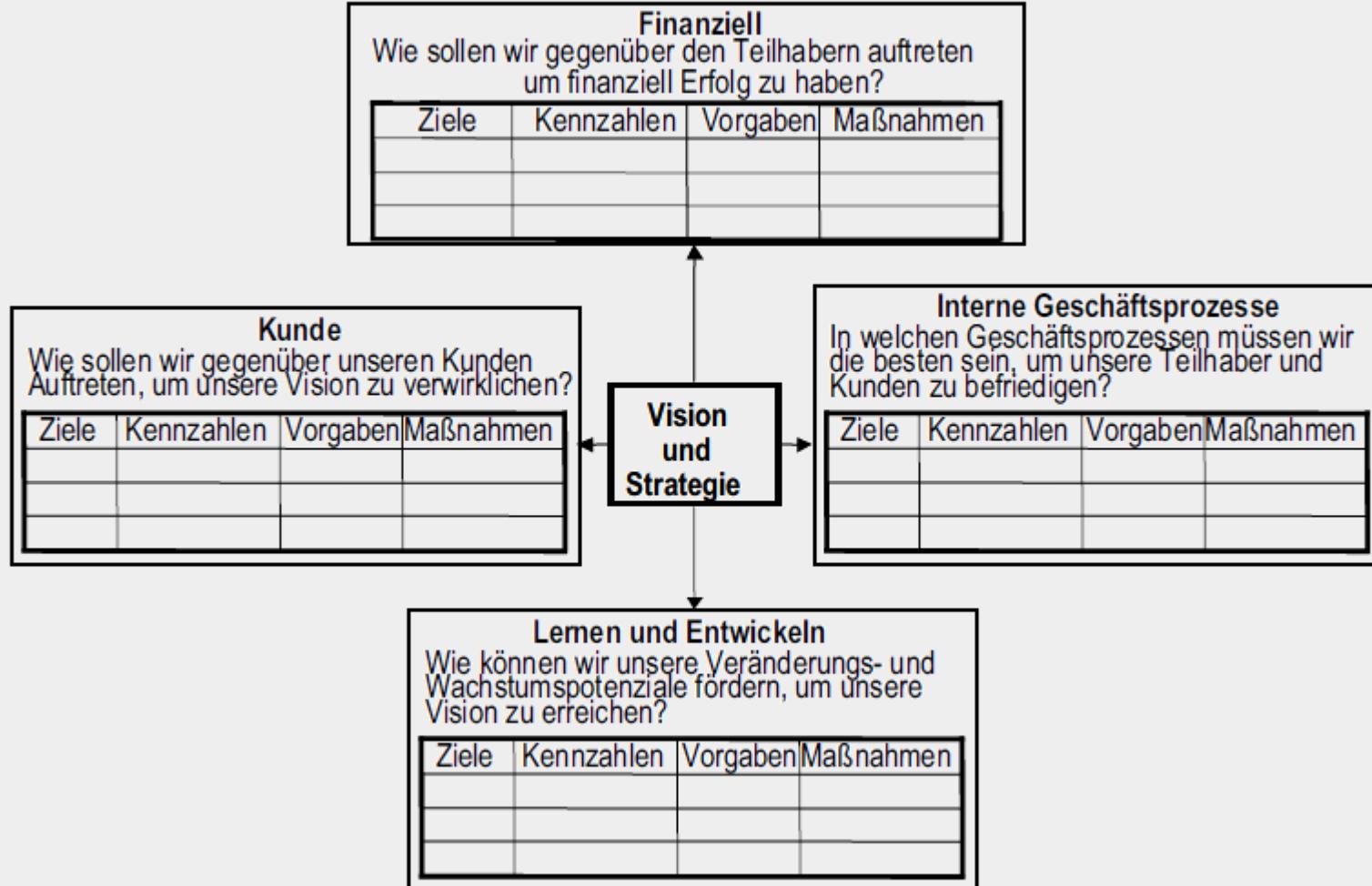


## Identifikation der KEF durch „Voting“ [Bullen]

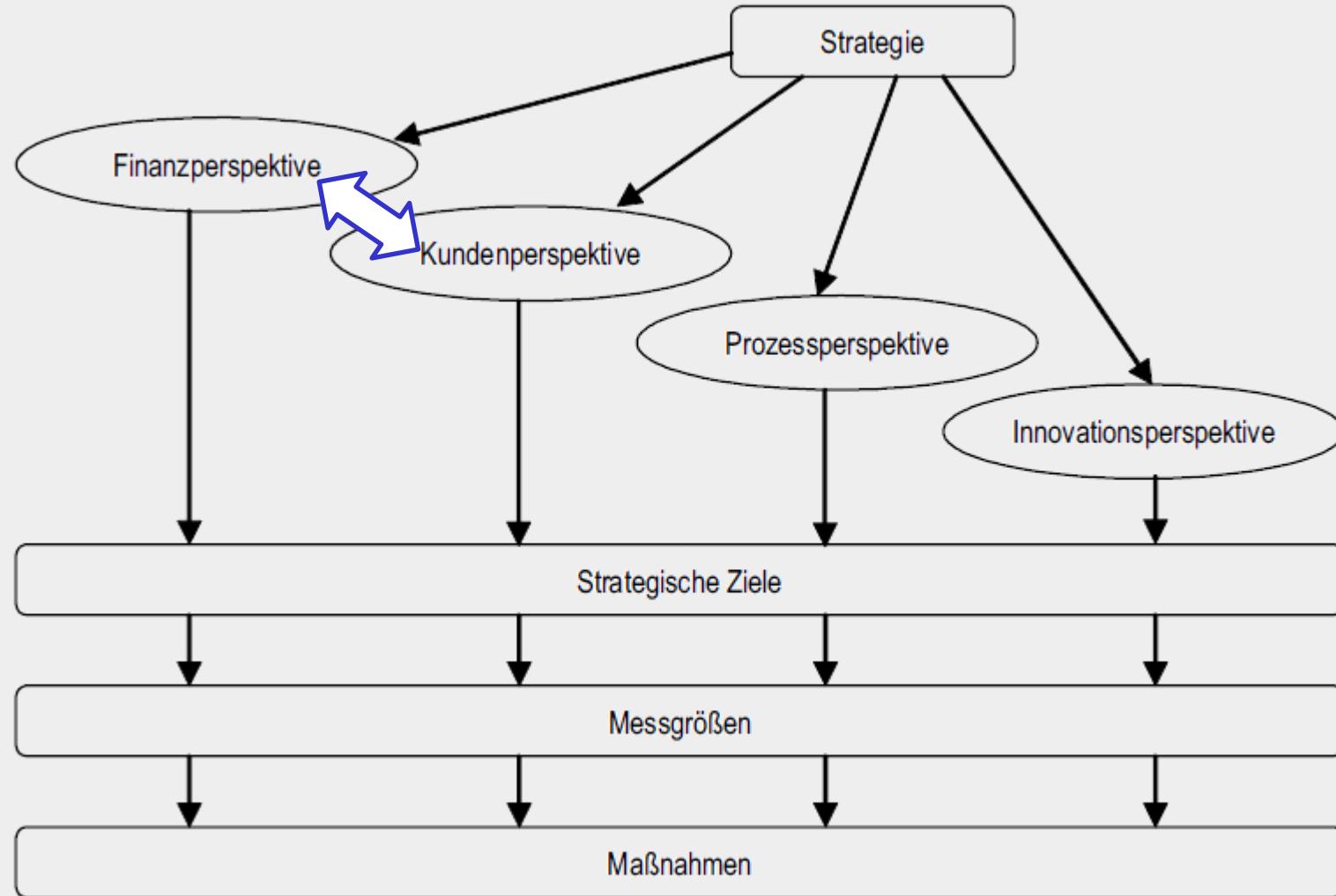
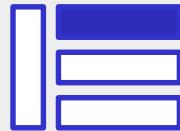


KEF Manager	Personal	Strategie	Arbeits- produktivität	Service- qualität	Pricing Strategie
<u>Division 1:</u> Maier Müller Schulz	(1) 5 2	(2) 1 3	(3) 2 1	4 - -	5 - 4
<u>Division 2:</u> Lutz Peters Schneider Fritz	(3) 1 2 5	1 4 - -	- - 5 1	(2) 3 1 2	- 2 - -
<u>Division 3:</u> Welker Reiter Heinrich Ahrend	3 3 4 -	(1) 4 1 1	2 - - -	1 3 - -	(4) 2 2 2

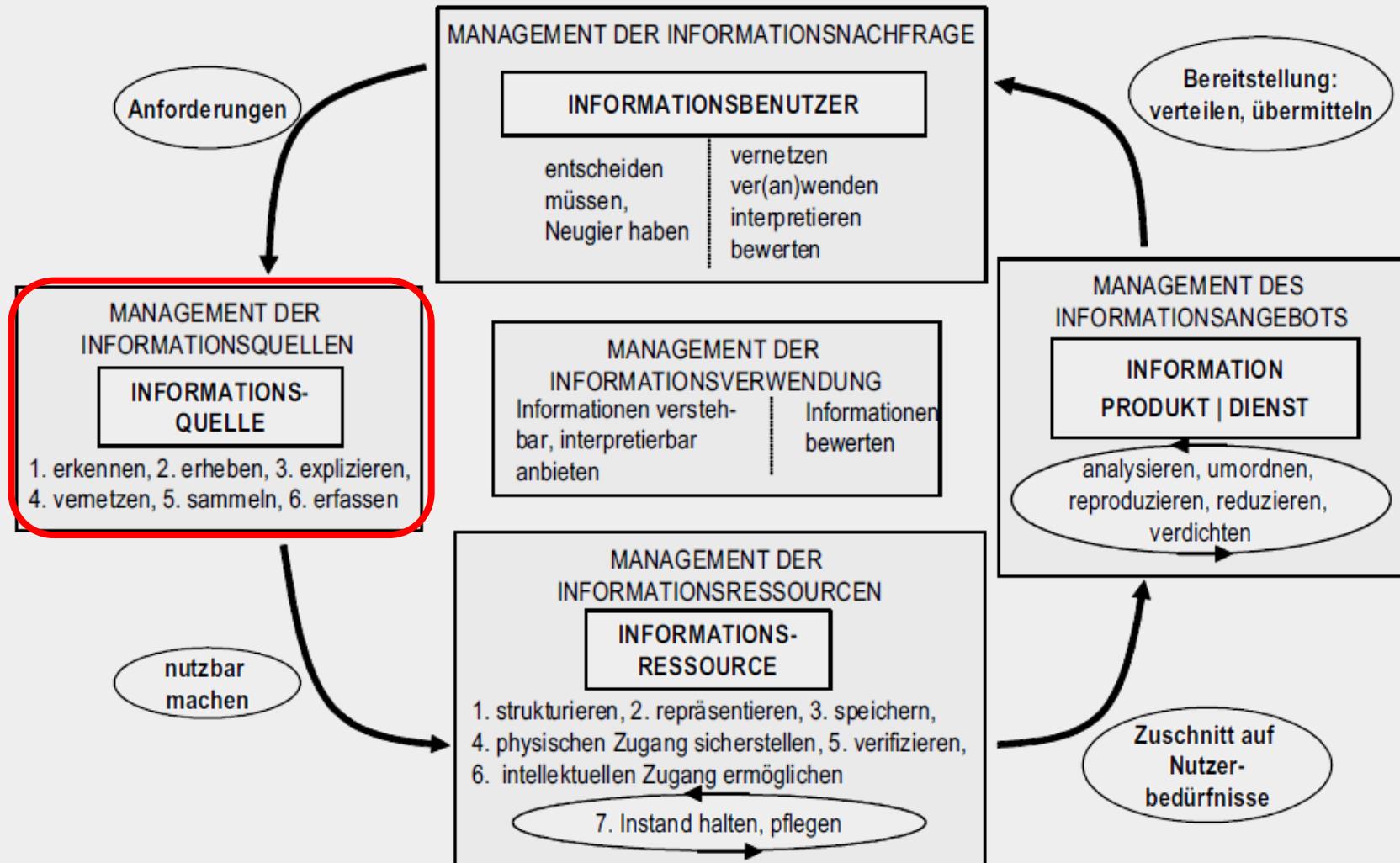
# Balanced Scorecard [Kaplan]



# Füllen einer Balanced Scorecard [Hensberg]



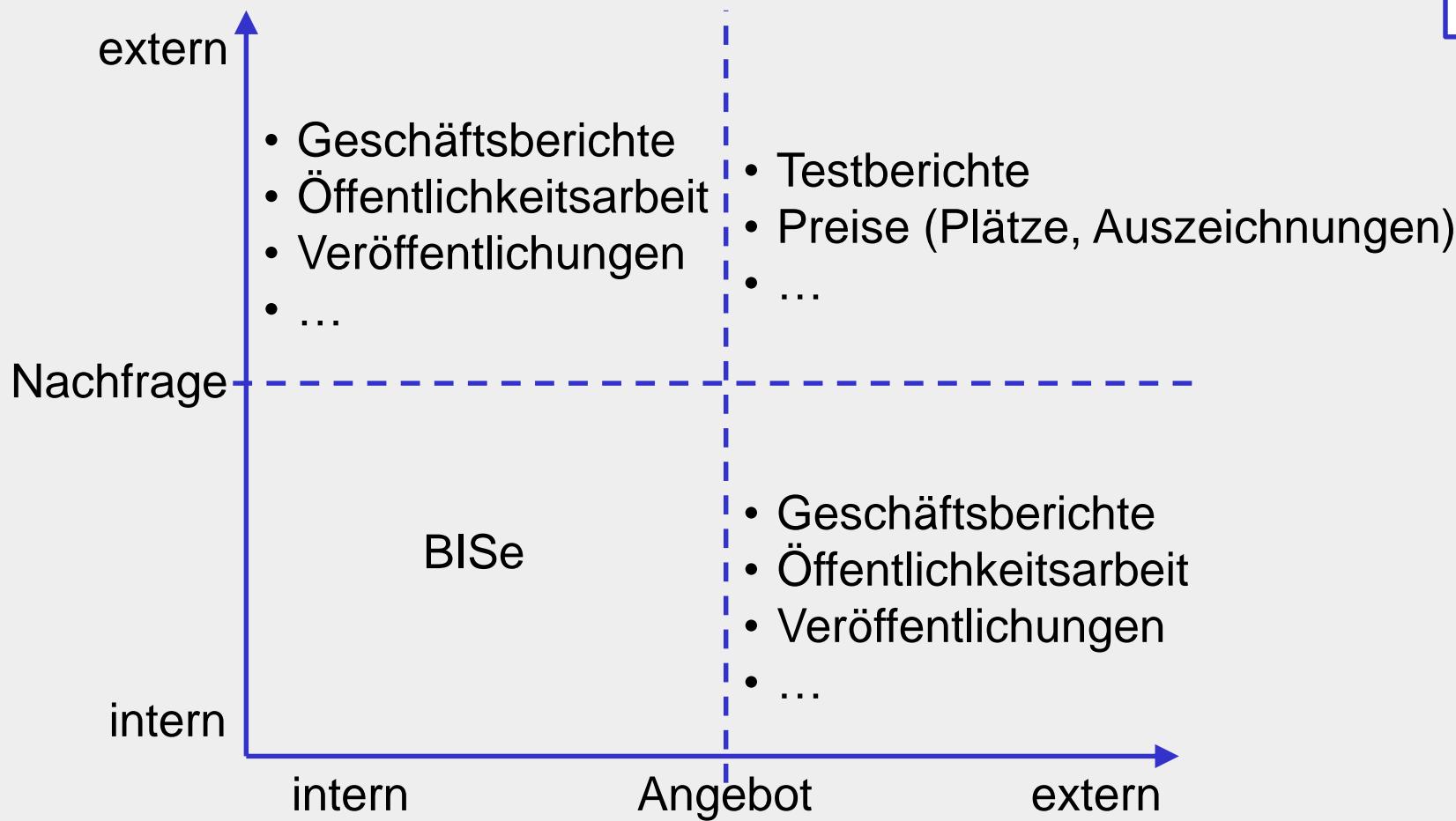
# Lebenszyklus der Informationswirtschaft [Krcmar; Rehäuser]



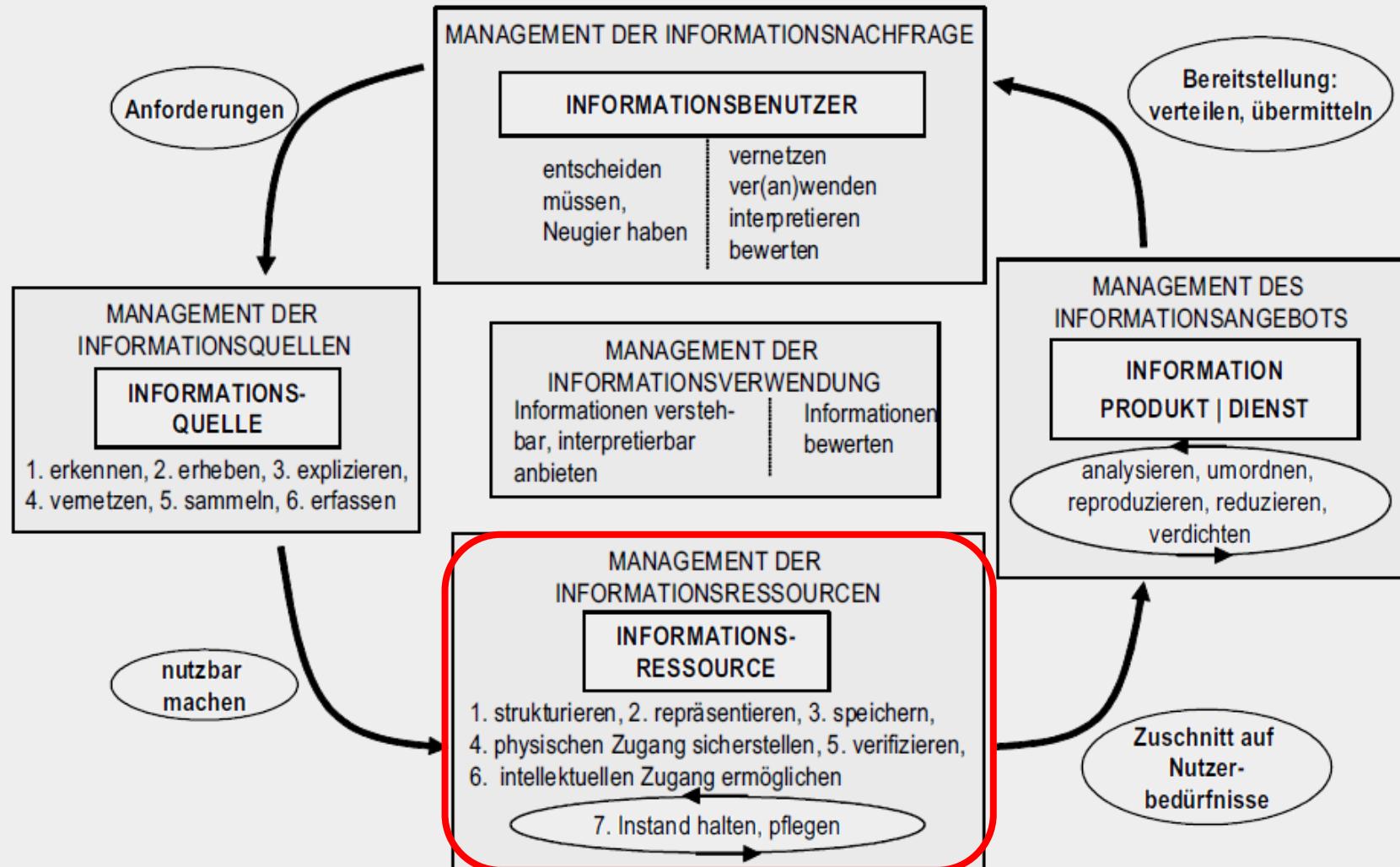
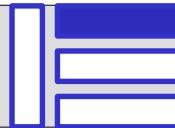
## Management der Informationsquellen

- Erkennen und Erheben von Informationen:
  - Neubewertung vorhandener oder
  - Schaffung neuer Informationen
- Sammeln und Erfassen von Informationen:
  - entstehungsnah
  - dezentral
- Informationsquellen (Beispiele):
  - Menschen
  - Online-Dienste
  - Patent-Datenbanken
  - Statistisches Bundesamt
  - Hochschulen
  - Information-Broker
  - ...

## Informationsangebot und Nachfrage



# Lebenszyklus der Informationswirtschaft [Krcmar; Rehäuser]

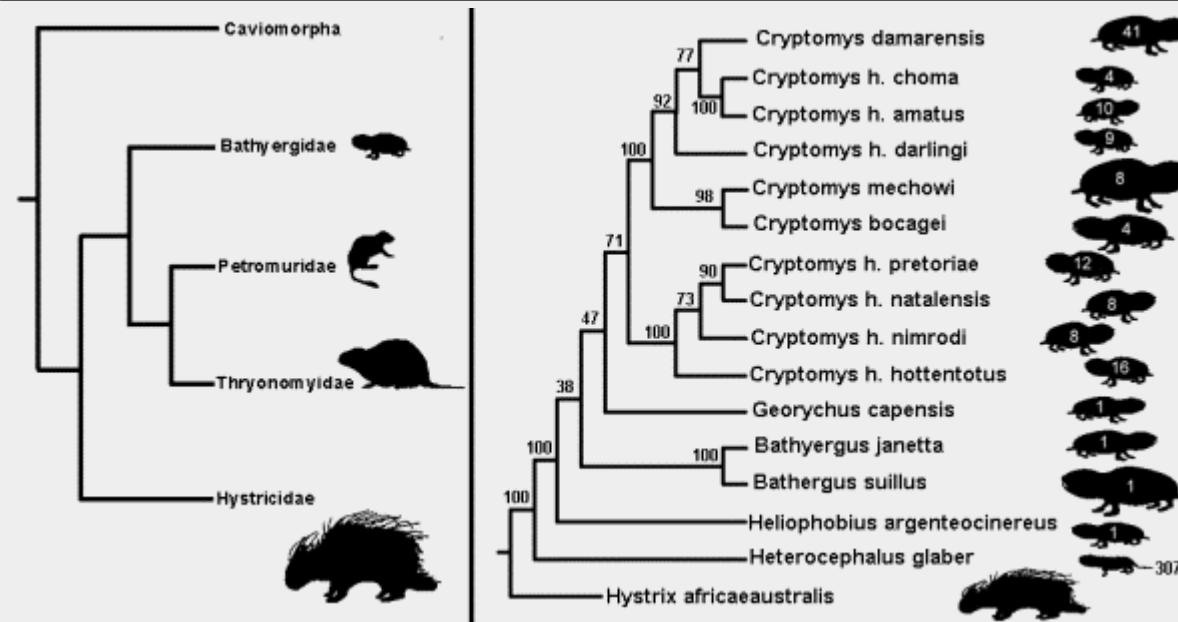


# Informationsstrukturierung – Taxonomie

[Kobuuun]



[Wikipedia]



Überordnung: Euarchontoglires

Ordnung: Nagetiere

Unterordnung: Stachelschweinverwandte

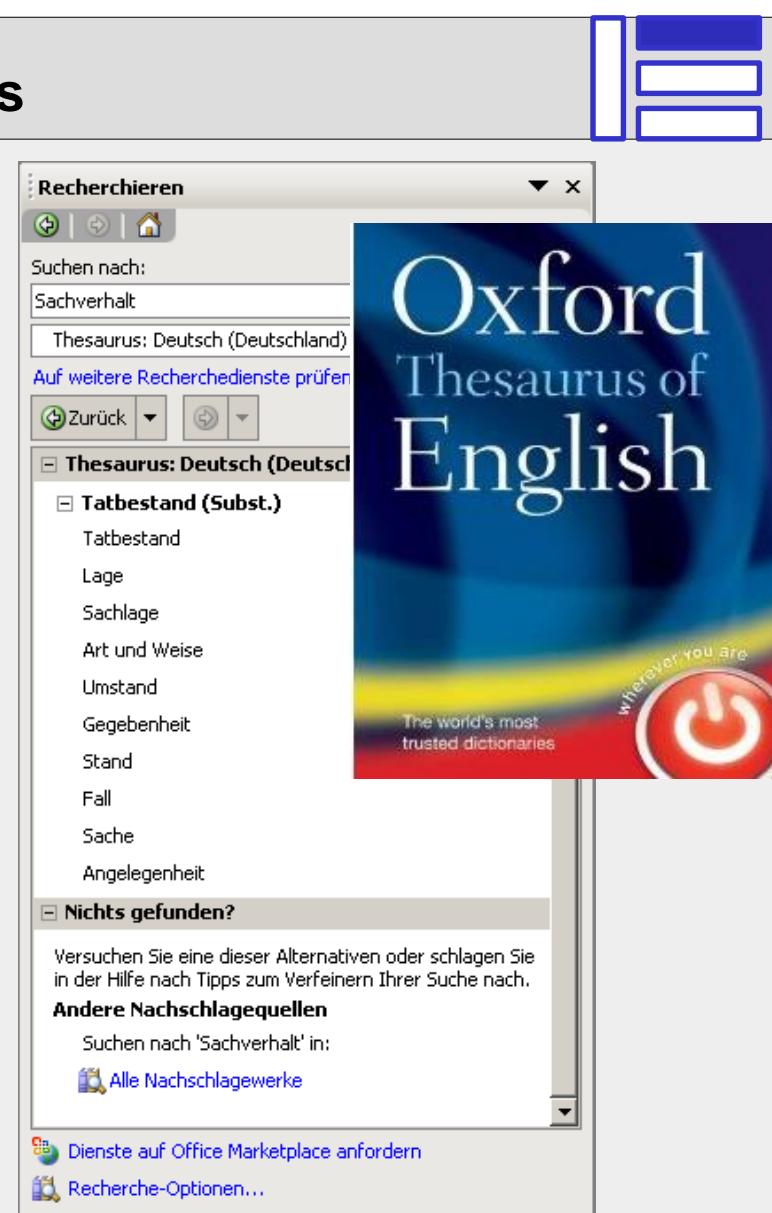
Familie: Sandgräber

Gattung: Nacktmulle

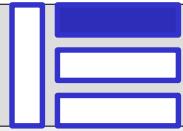
Art: Nacktmull

# Informationsstrukturierung – Thesaurus

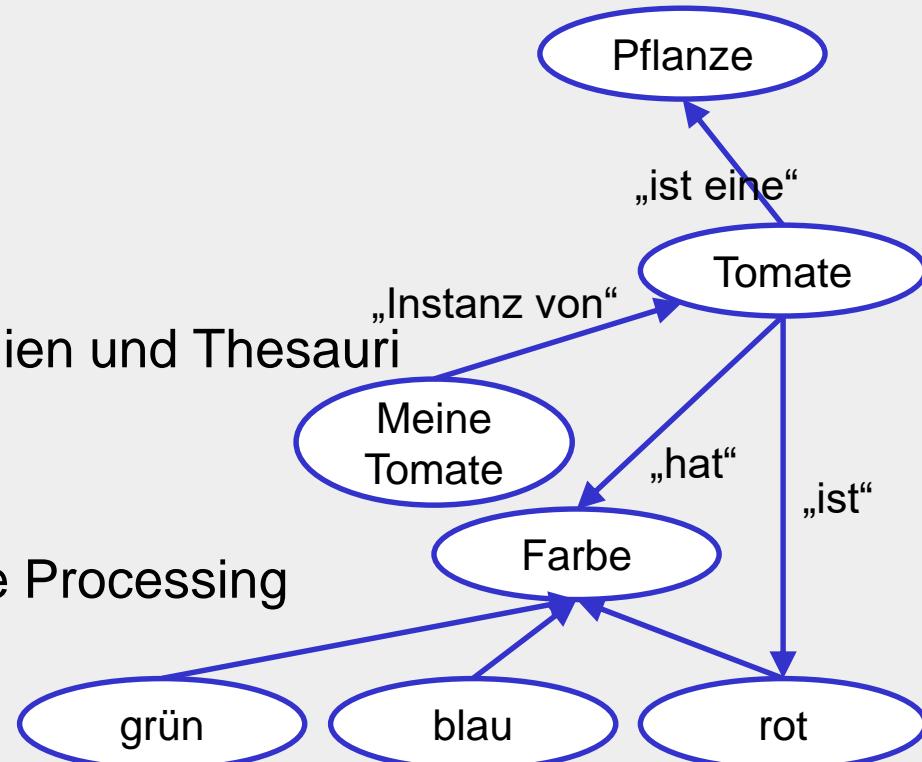
- Vokabeln, zwischen denen Beziehungen bestehen:
  - Synonym (Ersatzwort): Kartoffel ↔ Erdapfel
  - Homonym (gleichnamig): Tau → starkes Seil bei Schiffen, griechischer Buchstabe, Morgen~
  - Antonym (gegensätzliche Bedeutung): heiß → ← kalt
  - Generalisierung: Möbel ← Tisch, Bett, Schrank, ...
  - Spezialisierung: Fahrrad → Rennrad, Mountain-Bike, Radonneur, ...
  - ...



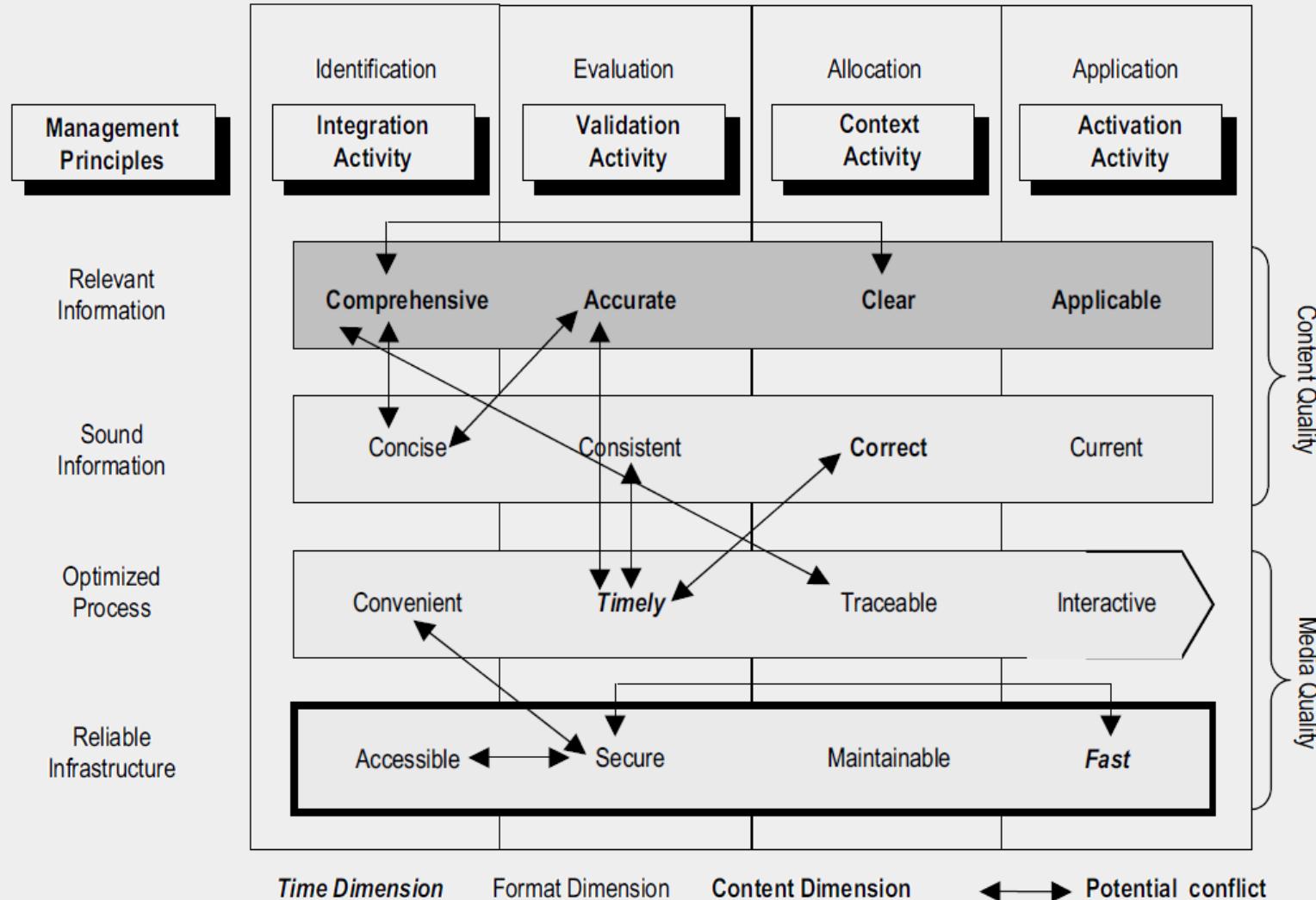
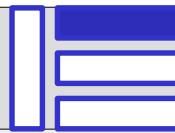
# Informationsstrukturierung – Semantisches Netz



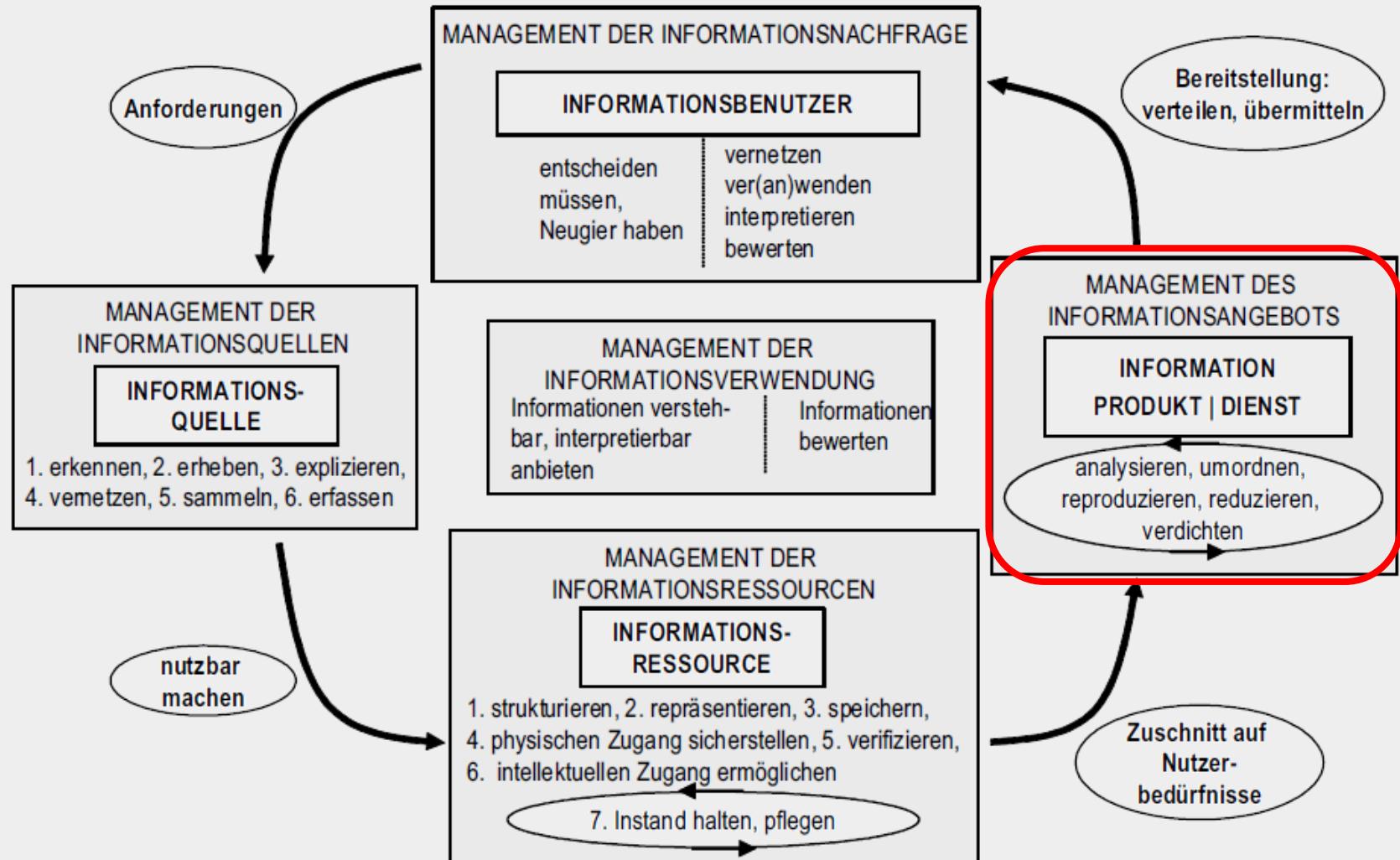
- Besteht aus Begriffen und Relationen:
  - Vererbung
  - Synonym
  - Antonym
  - Teil von
  - Eigenschaft
  - Kausalität
  - ...
- Verallgemeinerung von Taxonomien und Thesauri
- Wissensakquisition möglich
- Wissensanwendung möglich
- Anwendung im Natural Language Processing



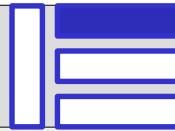
# Management der Informationsqualität [Eppler]



# Lebenszyklus der Informationswirtschaft [Krcmar; Rehäuser]

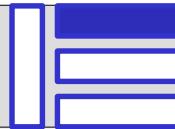


## Inhaltliche Anforderungen an Berichtswesen [Mertens]



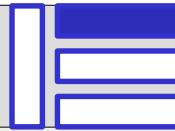
- Berichte formal einheitlich
- Keine isolierte Zahlendarstellung, sondern in Relation zu Vergleichsgrößen
- Relation zu Planwerten, Vergangenheitsdaten, Trends, ... → höhere Aussagekraft
- (Deutlich getrennte) Überblicks- und Detaildarstellung
- Außergewöhnliche Datenkonstellation hervorheben
- Dokumentation des Vorgehens zum Gewinnen der Daten im Bericht
- Grafische Darstellung besser als textuelle

## „Wirtschaftliche“ Anforderungen an Berichtswesen [Krcmar]



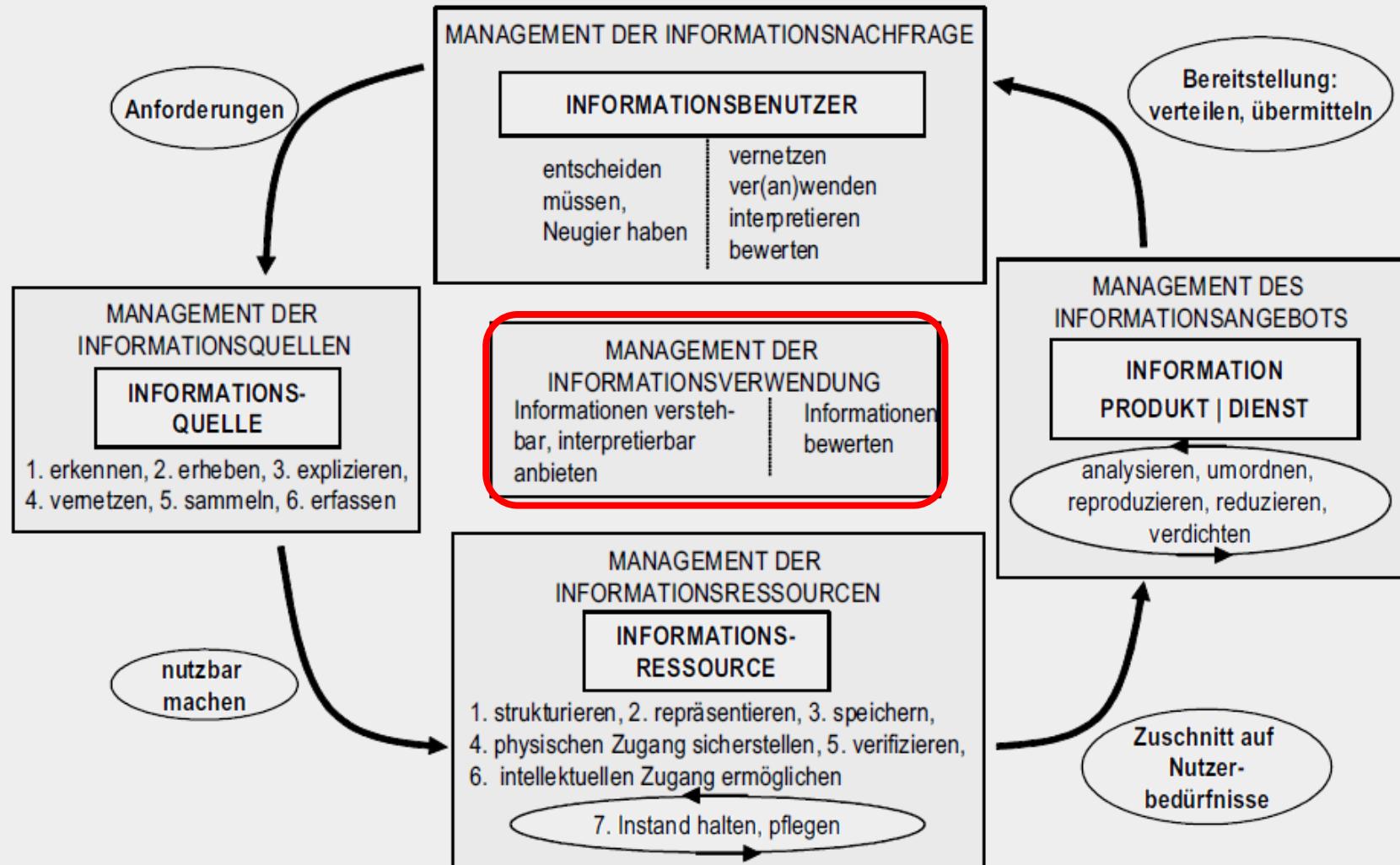
- **Nutzung des integrierten Funktionsvorrates der Standard-Software**
- **Hierarchiegerechte Aufbereitung der Berichtsinformationen**
- Ausrichtung der Berichtsmerkmale und -arten an den funktionsspezifischen Managementregelkreisen
- **Überschneidungsfreiheit der Berichte**
- Ausrichtung an Verbesserungspotenzialen (Kostensenkung) im Unternehmen

# Data-Warehousing

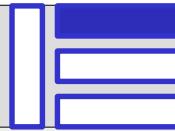


- Früher (ad-hoc-)Analysen über operativen (OLTP-)Systemen:
  - schwer zu erstellen
  - aufwendig auszuführen
  - Gefährdung der Verfügbarkeit des operativen Systems
- Lösung: Data-Warehousing:
  - vorverdichtete Daten
  - in separaten (OLAP-)Systemen
  - keine der früheren Nachteile mehr
- Eigenschaften:
  - subject-oriented: ausgerichtet an spezieller Unternehmensstruktur
  - integrated: auch externe Daten und im gleichen Format
  - nonvolatile: nur Einfüge und Leseoperationen, kein Ändern u. Löschen
  - time-variant: expliziter und impliziter Zeitraumbezug

# Lebenszyklus der Informationswirtschaft [Krcmar; Rehäuser]

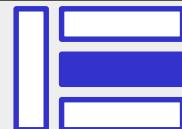
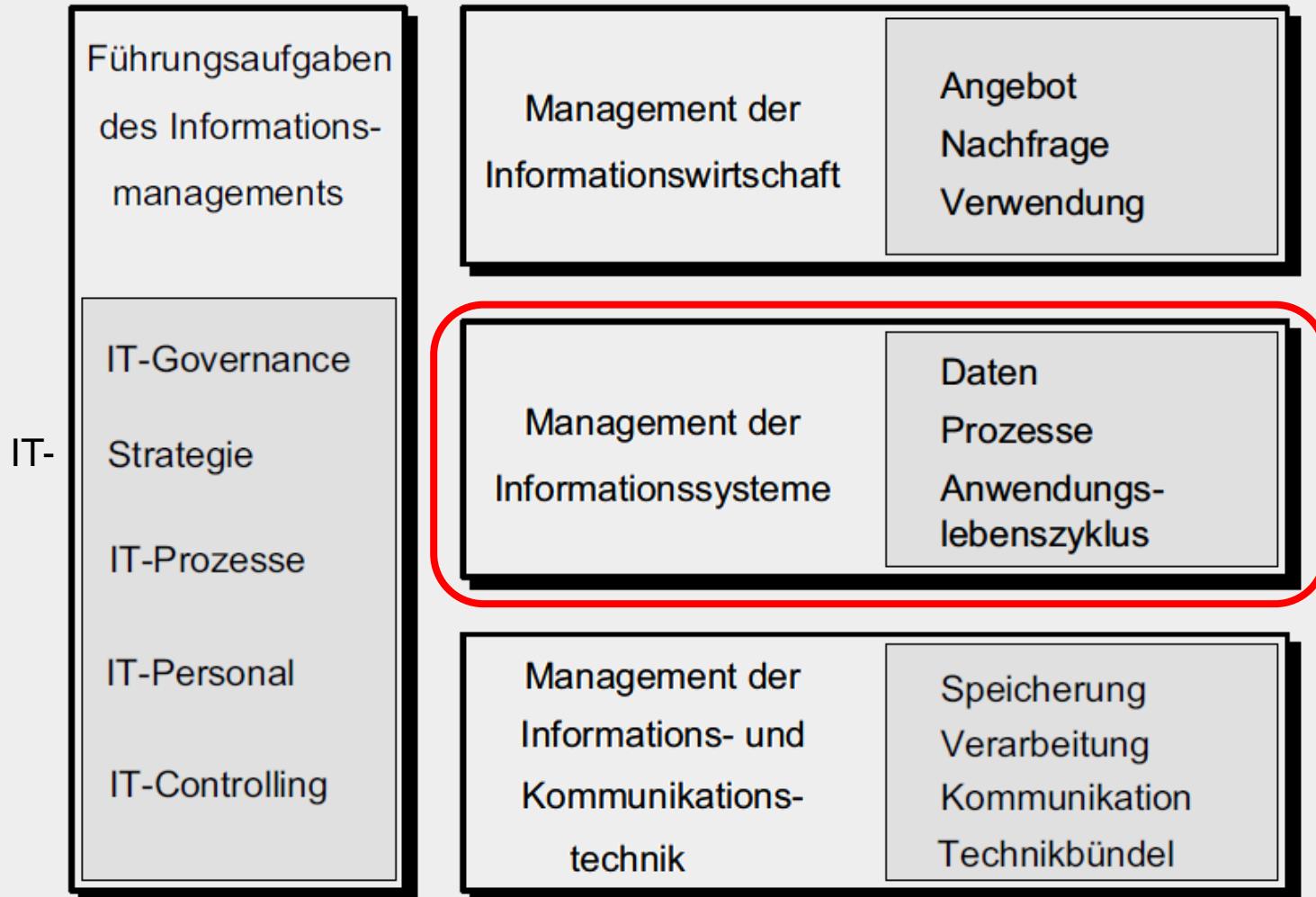


## Bewertung von Informationen [Zubaza]



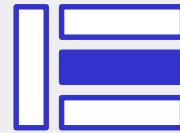
- Informationen haben keinen absoluten Wert.
- Normativer Wert:
  - Differenz zwischen den besten Alternativen vor und nach der Informationsbeschaffung (Opportunitätskosten)
  - Abzug der Kosten für die Informationsbeschaffung?
  - Ex post Betrachtung bei vollständiger Information
- Realistischer Wert:
  - Empirisch messbarer Gewinn aus durch Nutzung der Information veranlassten Handlungen
  - Nur durch Simulation oder Prototyping zu erheben
- Subjektiver Wert
  - An Individuum gebunden und daher nur schwer aus der Summe subjektiver Bewertungen herzuleiten
  - Ex post Betrachtung

## „Referenzmodell des IM“ [Krcmar]

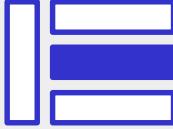


## Inhalte des Datenmanagements

- *Datenmodellierung*
- Datenadministration
- Datentechnik
- Datensicherheit (Schutz vor unbefugtem Zugriff)
- Datenkonsistenz
- Datenschutz (Sicherung von Daten, Backups, ...)
- Datenbereitstellung
- Datennutzung
- ...



## Arten von Datenbank(managementsystem)en

- Relationale Datenbanken bzw. Datenbankmanagementsysteme
  - Objektorientierte Datenbanken bzw.  
Datenbankmanagementsysteme
  - Verteilte Datenbanken bzw. Datenbankmanagementsysteme
  - Mischformen der obigen
  - Weitere und No-SQL-Datenbanken
- 

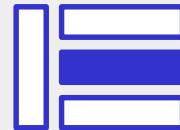
## Primäre Anforderungen an Datenbanken

- konsistent
- verfügbar
- performant (auch bei sehr vielen, gleichzeitigen Zugriffen)
- leichtes Reporting
- leichte und schnelle (ad hoc) Analyse
- sicher
- fähig, sehr große Datenmengen zu speichern
- kostengünstig
- ...



## Sekundäre/abgeleitete Anforderungen an Datenbanken

- skalierbar (vertikal? horizontal!), um sehr hohem und ständig wachsendem Datenaufkommen gerecht zu werden
- leicht verteilbar
- leicht partitionierbar und replizierbar
- *leichte/schnelle Strukturänderungen (ohne „Downtime“)*
- *Speichern wenig- oder unstrukturierter Daten*
- ...



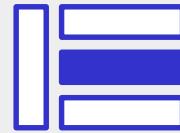
## Anforderungen BISe an Datenbanken [Munkelt]

Anforderung ↓\SW →	WWS	BSW	APS	BIA	WEB
konsistent	+	++	o	o	+
verfügbar	+	+	o	o	++
performant	+	o	++	+	+
leichtes Reporting	+	+	--	++	o
leichte/schnelle Analyse	+	+	--	++	-
sicher	+	++	-	o	+
sehr große Datenmengen	o	o	+	++	+
skalierbar (horizontal)	-	o	+	++	+

Legende: ++ .. äußerst wichtig, + .. sehr wichtig, o .. wichtig, - .. weniger wichtig, -- .. nahezu unwichtig

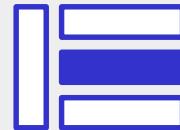
## Fähigkeiten relationaler DBen, bezogen auf BISe

- + konsistent
- +/- verfügbar
- + performant
- + leichtes Reporting
- + leichte/schnelle Analyse
- + sicher
- - **sehr große Datenmengen**
- - **skalierbar (horizontal)**



## Lösungsansatz: Verteilte Datenbanken

- große Datenmengen auf mehrere verbundene Datenbanken („Knoten“) verteilen (funktional oder „Sharding“)  
→ Bessere Lastverteilung, höhere Performance (FiBu xor MaWi; Sachbearbeiter A-M und N-Z)
- Daten (partiell) duplizieren bzw. replizieren  
→ Verfügbarkeit beim Ausfall einzelner Knoten gewährleistet
- mehrfaches Replizieren geschriebener Daten  
→ schnellere Lesezugriffe bei „Leseüberhang“
- ... und Kombinationen



## Welche Art DB für welche Art BIS? [Munkelt]

Anforderung ↓\SW →	WWS	BSW	APS	BIA	WEB
konsistent	+	++	o	o	+
verfügbar	+		o	c	+-
performant	+		+	-	-
leichtes Reporting	+	+		-	-
leichte/schnelle Analyse		+		-	-
sicher		++	-	-	-
sehr große Datenmengen	-	o	+	+	+
skalierbar (horizontal)	-	o	+	-	+

*monolithische  
relationale SQL-Datenbanken*

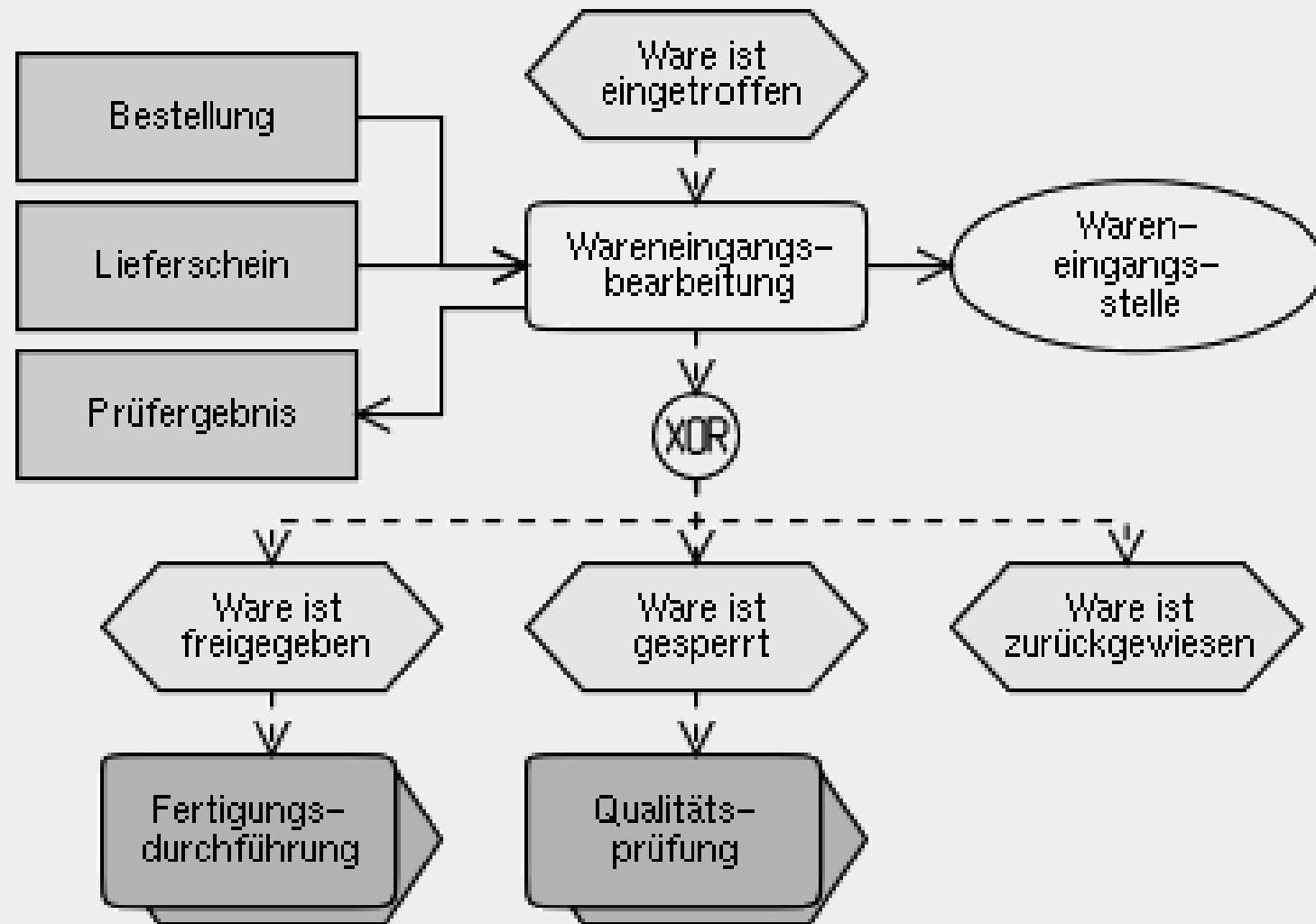
*Graphendatenbanken*

*spaltenorientierte Datenbanken*

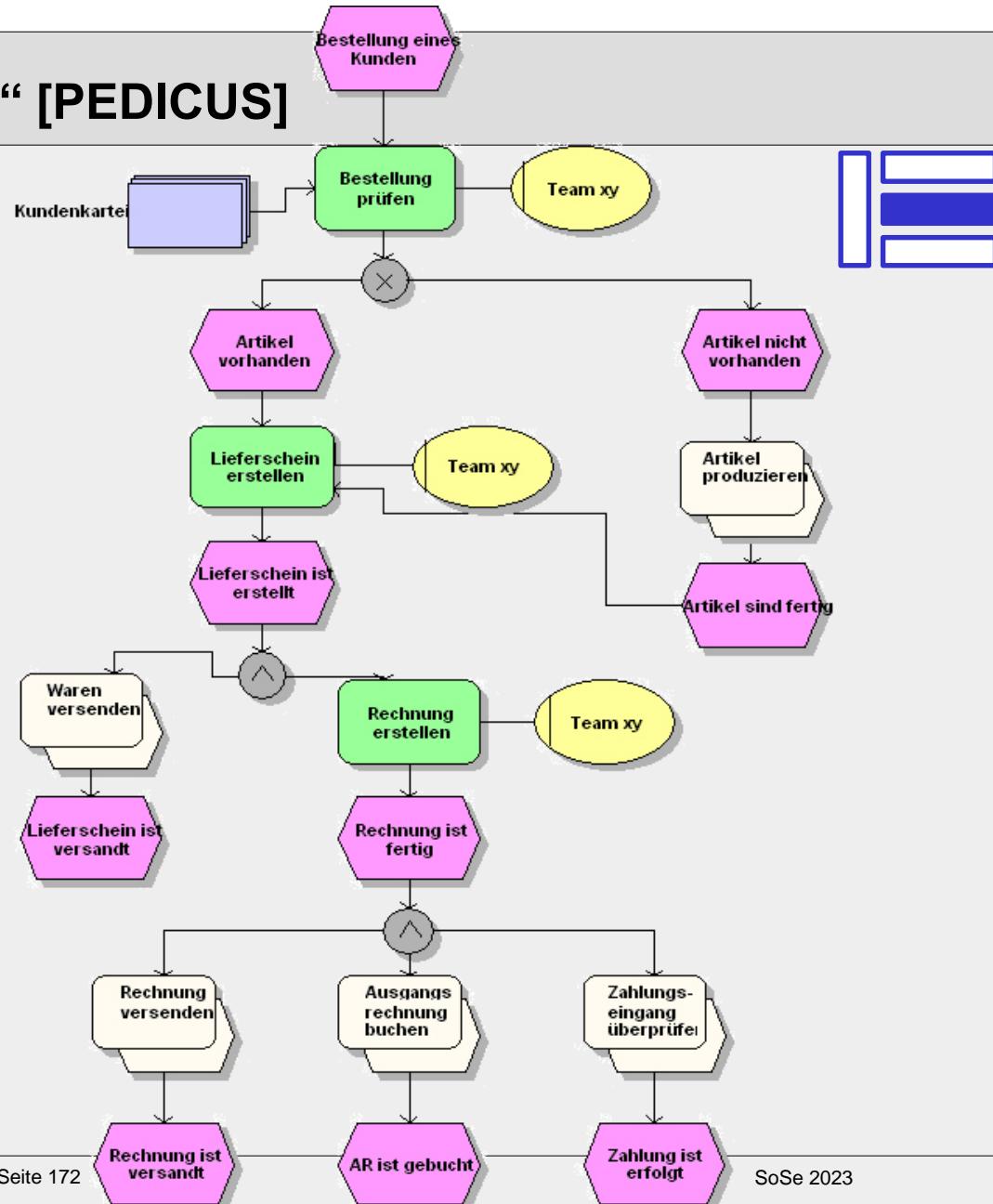
*Schlüssel-Wert-Datenbanken*

Legende: ++ .. äußerst wichtig, + .. sehr wichtig, o .. wichtig, - .. weniger wichtig, -- .. nahezu unwichtig

# Prozessmanagement mit EPKen – „Wareneingang“ [hawlisch.de]

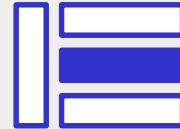


# EPK – „Auftragsabwicklung“ [PEDICUS]



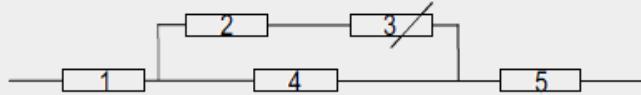
## Prozessbeurteilung

- Kriterien:
  - Ergebnisqualität
  - Prozesskosten
  - Termintreue
  - **Durchlaufzeit**
  - Kapazitätsauslastung
  - Kapitalbindung
- Kennzahlen z. B. aus einer Balanced Scorecard
- Messen der Kennzahlen in der Geschäftsprozessschicht des BISs



## Maßnahmen zur DLZ-Verkürzung [Bleicher]

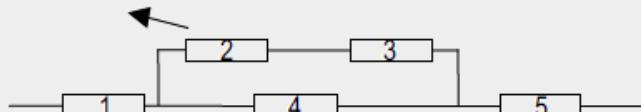
Weglassen



- Überprüfung der Notwendigkeit zur Funktionserfüllung
- Abschaffen von Medienbrüchen

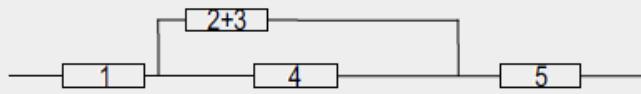


Auslagern



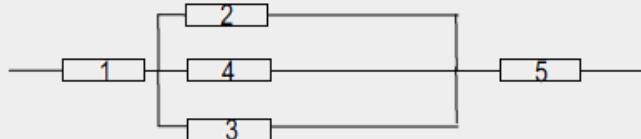
- „Vorfeld“-Aktivitäten verstärken
- Vergabe von Aktivitäten, z.B. extern

Zusammenfassen



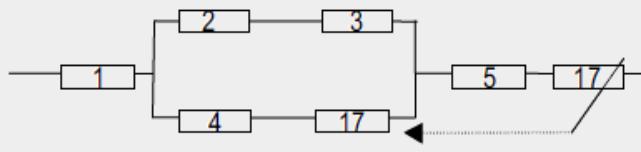
- Zusammenlegung von Aktivitäten

Parallelisieren



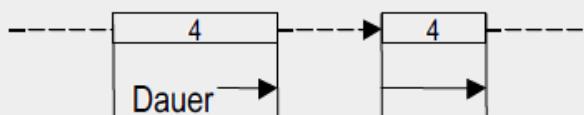
- Erhöhung der Arbeitsteilung

Verlagern



- Früherer Beginn von Aktivitäten

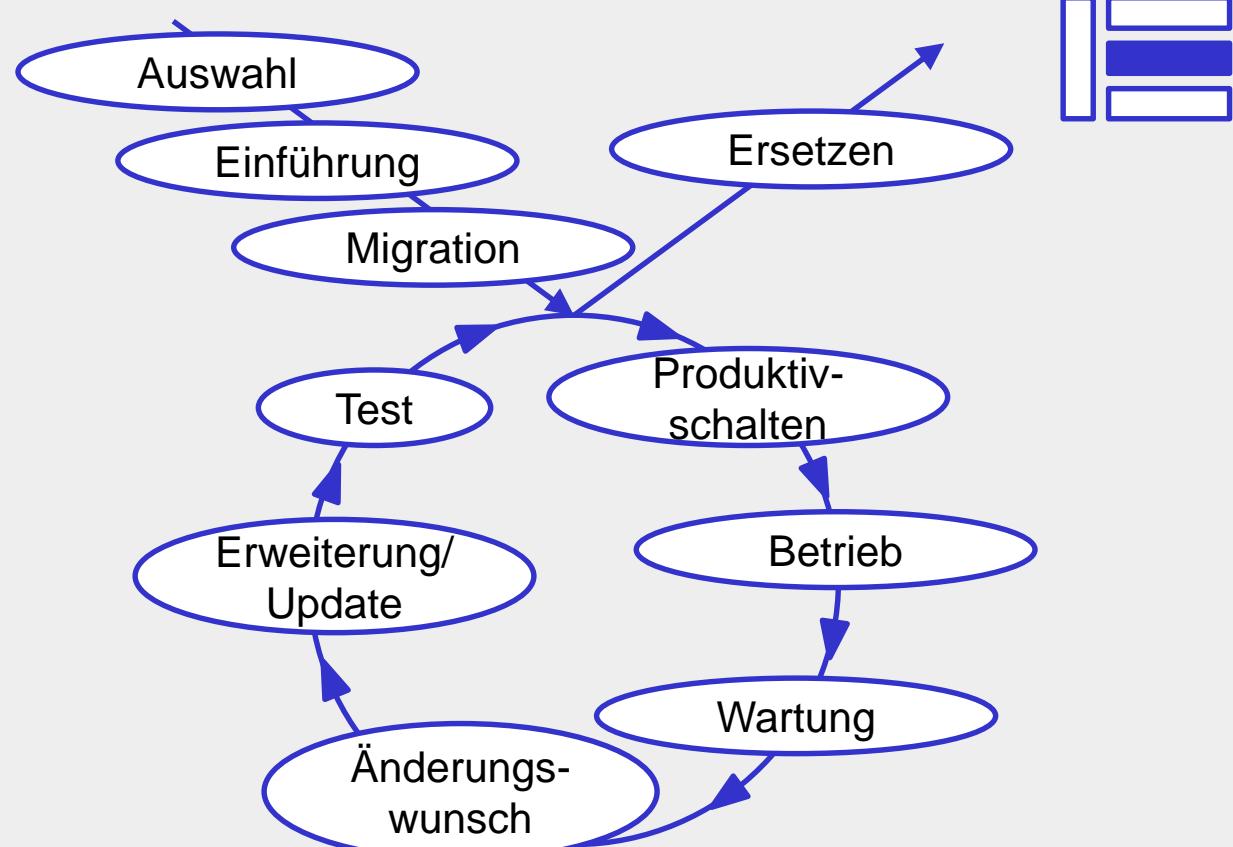
Beschleunigen



- Bereitstellung von Arbeitsmitteln zur effizienten Aufgabenerledigung
- Vermeidung von Warte- und Liegzeiten

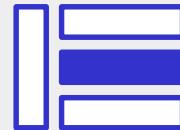
# Lebenszyklus BISe

- **(Make or Buy)**
  - **Auswahl**
  - Erweiterung
  - **Einführung**
  - **Datenmigration**
  - Produktivschaltung
  - Betrieb
  - **Wartung**
  - Erweiterung/Update
  - Test
  - ...
  - Ersetzen



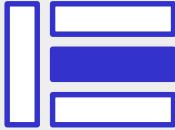
## Make or Buy? Buy!

- Früher: Make; heute: Buy
- Vorteile von „Buy“:
  - geringere Kosten durch Kostendegression
  - keine Entwicklungszeiten → sofortige Produktverfügbarkeit
  - hohe Programmqualität
  - Gewährleistung der Programmwartung und –weiterentwicklung
  - Unabhängig von internen IT-Ressourcen
  - Standardprozesse!
  - Hoffentlich integriertes Gesamtsystem, keine oder gute Schnittstellen



## BISe als Open-Source-Software (OSS)

- Eigenschaften von OSS:
  - Öffentlicher Quelltext, auch bei anderen BISen
  - Keine Lizenzgebühren → geringere Kosten
  - Freie Weitergabe – auch nach Anpassung/Erweiterung!
- Vorteile:
  - Keine Lizenzgebühren, aber andere Kosten
  - Bei Erweiterungen nicht von einem Hersteller abhängig
  - Geteilter Aufwand für die (Weiter)entwicklung
  - Einblick in SW-Struktur (→ Wart- und Erweiterbarkeit) und Reifegrad
- Nachteile:
  - Geringer *inhaltlicher* Reifegrad (derzeit) → Erweiterungen → Kosten
  - Erweiterungen/Anpassungen für jedermann einseh- und verwendbar → kein Wettbewerbsvorteil durch individuelle Weiterentwicklung
  - Gefahr, suboptimale Individualgeschäftsprozesse zu etablieren
  - Gefahr, Kapselung zu unterlaufen



## Verfügbare Module im ADempiere



## Installationsprobleme

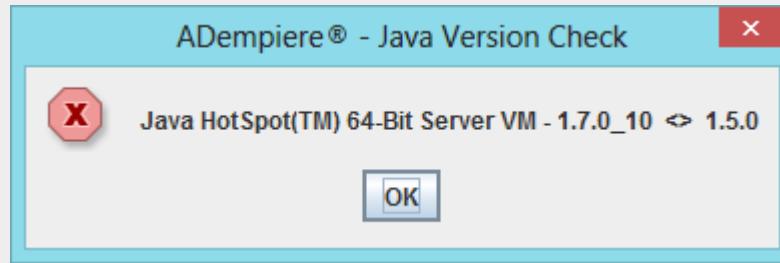
- unzureichende Installationsdokumentation



### Welche „back end“ Version?



- restriktive Softwarevoraussetzung



# mangelnder bzw. unzureichender Support

Von Werner Scharinger <werner.scharinger@catura.de>★

Betreff Re: ADempiere - Frage zur Rechnungserstellung

An s64633@htw-dresden.de★

Hallo,

es freut uns natürlich, wenn mit ADempiere gearbeitet wird. :-)

Zur Frage:

Um aus einem Auftrag einen Lieferschein oder einen Lieferschein / eine Rechnung zu erstellen, kann im Hauptmenü unter Punkt "Lieferschein manuell erstellen" oder "Lieferschein / Rechnung manuell erstellen" ausgewählt werden.

**Allgemeine „Standard“ Antwort**

Es öffnet sich dann eine Auswahlmaske, in der der Auftrag gewählt werden kann. Wird diese ausgewählt, so werden Lieferschein und Rechnung erstellt. Die jeweiligen Belegnummern werden zur Info angezeigt.

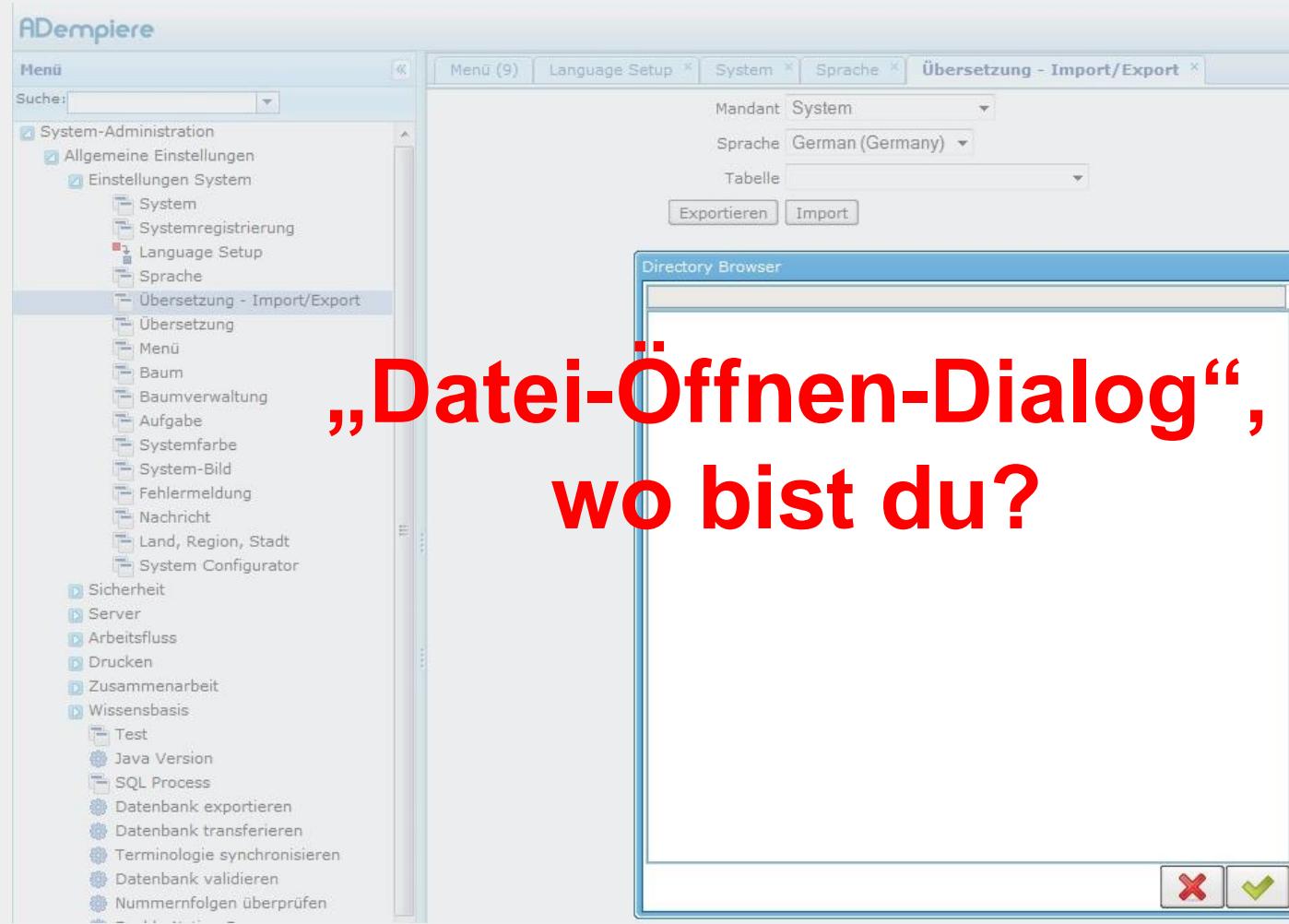
Im Auftrag kann dann über das "Lupensymbol mit Pfeil" (rechts neben den Druckersymbolen) direkt in den zugeordneten Lieferschein / Rechnung gewechselt werden. Über diese Referenz werden auch weitere Dokumente (Zahlungen usw.) verknüpft, soweit sie vorhanden sind.

Ich hoffe, dass hilft weiter. Viel Erfolg bei der weiteren Arbeit.

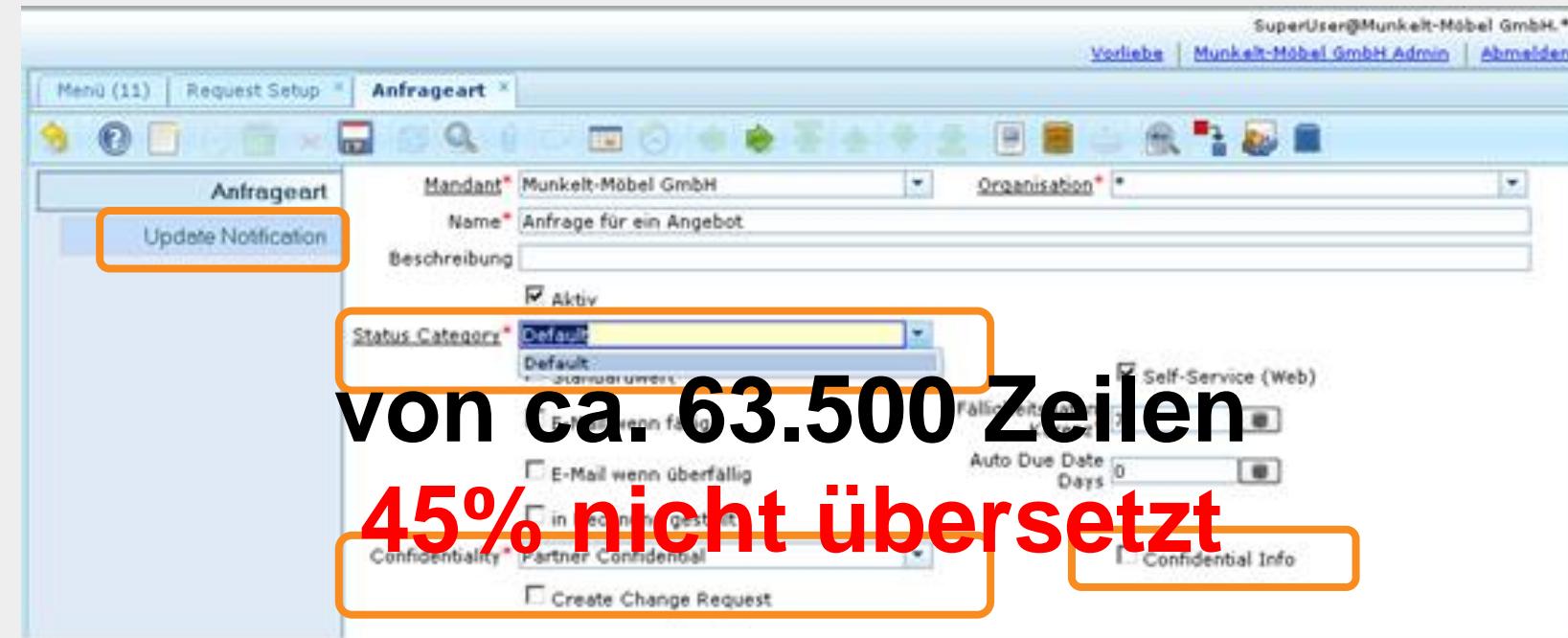
Beste Grüße,  
W. Scharinger



# Fehler



# unvollständige Übersetzung



The screenshot shows a software interface for managing sales orders. The top bar includes 'Menu (24)', 'Bestellung aus Auftrag gener...' (highlighted), 'Auftrag', 'Bestellung aus Auftrag gener...', and 'Bestellung aus Auftrag gener...'. Below is a section titled 'Bestellung aus Auftrag generieren'. The text explains that after completing sales orders, you can create one or more purchase orders for each sales order. It details the process of using the Sales Order Organization to create Purchase Orders, mentioning Vendor Price Lists and Unit of Measure. At the bottom, there are input fields for 'Auftragdatum' (17.12.2012), 'Geschäftspartner' (Schulte GmbH), 'Lieferant' (Meier GmbH), and 'Vertragsauftrag' (30004\_2012-12-20 00:00:00). The 'ADempiere' logo is in the bottom right corner.

## Funktionsfehler oder Zufall?

### Einzelblattansicht

Aktion

Handelsvertreter:

Nächste Aktion:

Standard Response:

Ergebnis:

Product Used:

Aktivität:

Task Status:

Start Plan:

Startdatum:

Funktion:

Entry Confidentiality: Partner Confidential

Brieftext:

Quantity Plan:

Complete Plan:

Close Date:

**MAGIC BUG?**

### Datenblattansicht

Anfrage	Self-Service (Web)	Handelsvertreter	Funktion	Nächste Aktion
		MunkeltMoebelAdmin MunkeltMoebelAdmin MunkeltMoebelUser		





# Sorry SAP

OpenERP 7.0

Is about to be released in 5 days

Share the news and get immediate access to the beta!

## Fakten und Gründe für OpenERP

- Fakten
  - OpenERP Unternehmen 2005 in Belgien gegründet
  - Niederlassungen in Europa (Brüssel), USA und Indien
  - 180+ Angestellte weltweit
  - Wachstum ~ 100% p.a.
- Gründe für OpenERP
  - 1.000 Installationen pro Tag
  - 400+ OpenERP-Partner in 70+ Ländern
  - kommerzieller Support
  - Entwicklung wird offen geführt auf <http://launchpad.net/>

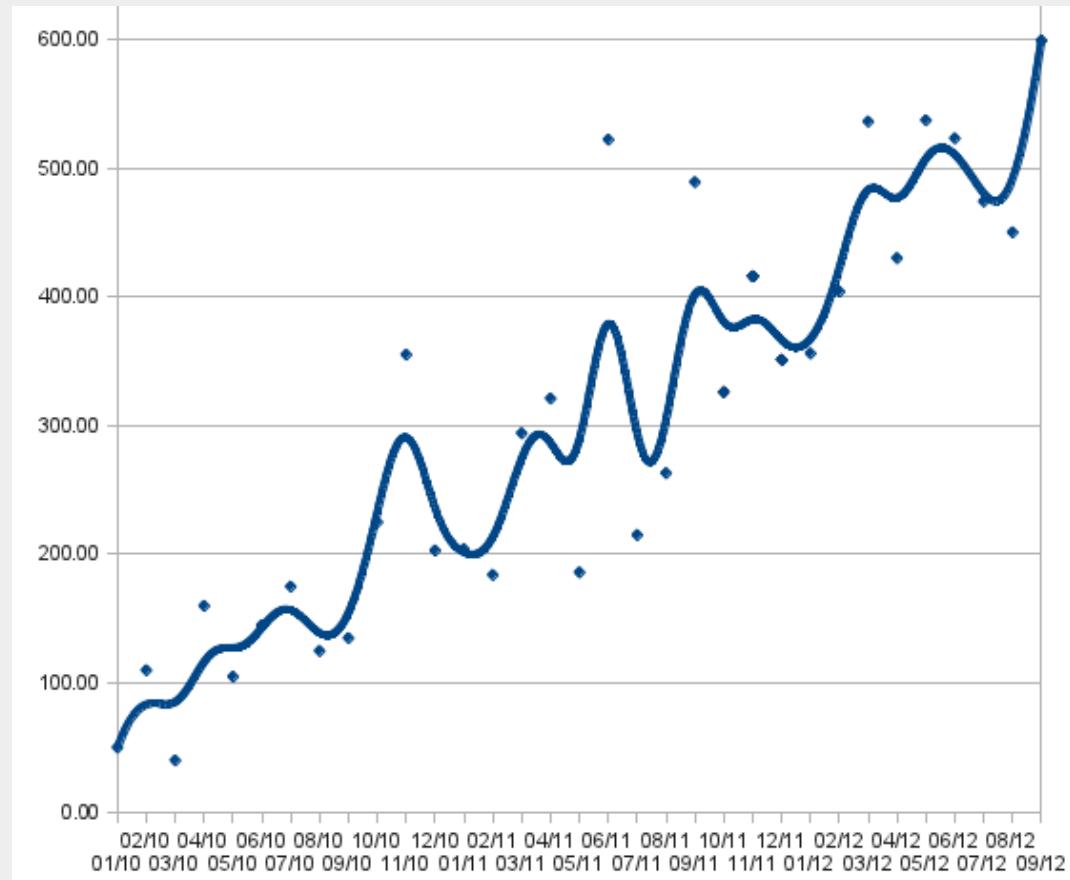
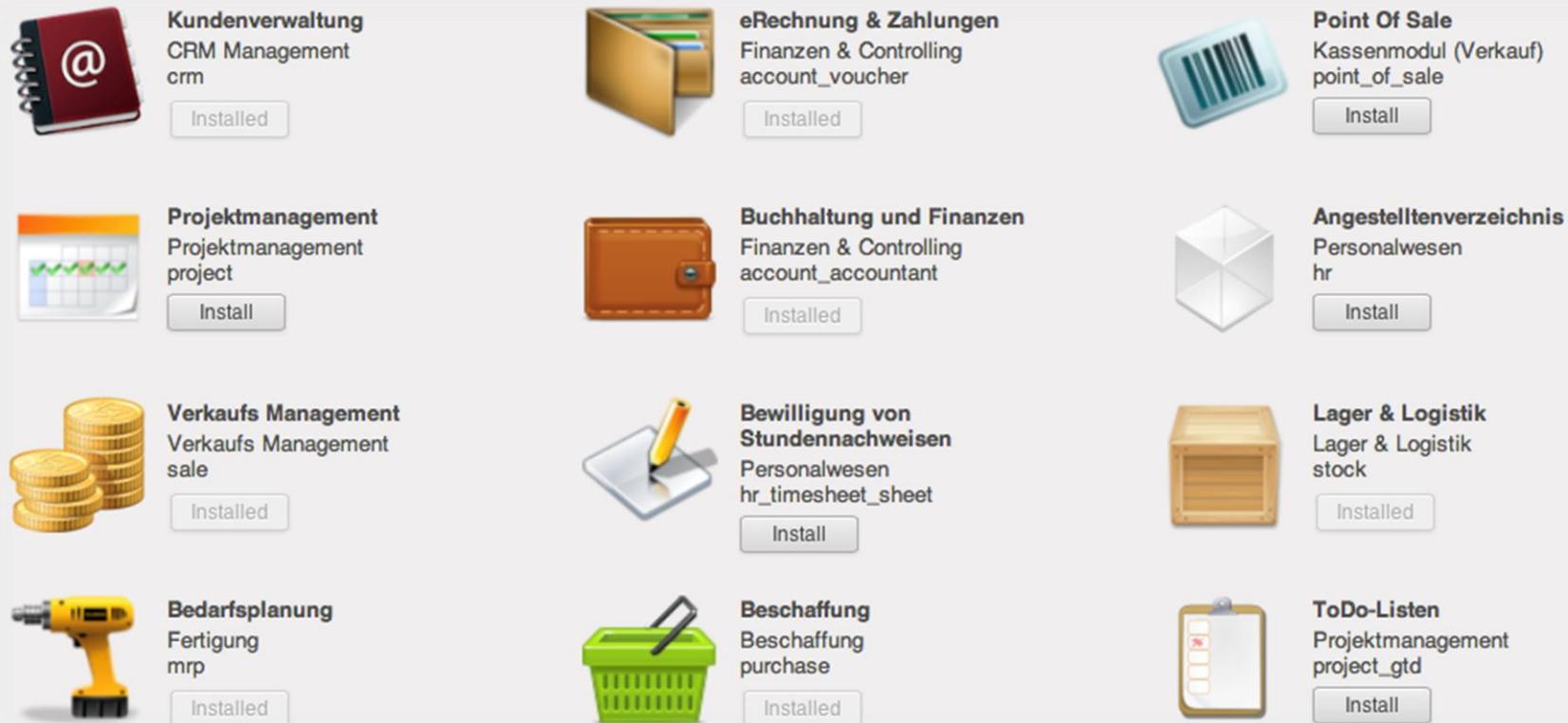


Abb.: monatlicher Umsatz/Absatz von OpenERP in T€ 0/2010 bis 09/2012

## Verfügbare Module in OpenERP (1)

- stark modularisiert: 160+ Module in OpenERP 6.1
- Bereiche: Buchhaltung, Rechnungslegung, Lager und Logistik, Personalwesen, Fertigung, Point of Sale, Projekt-Management, CRM, Marketing



## Verfügbare Module in OpenERP (2)



Kundenverwaltung  
CRM Management  
crm

Installed



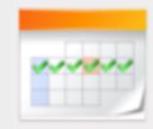
eRechnung & Zahlungen  
Finanzen & Controlling  
account\_voucher

Installed



Point Of Sale  
Kassenmodul (Verkauf)  
point\_of\_sale

Install



Projektmanagement  
Projektmanagement  
project

Install



Buchhaltung und Finanzen  
Finanzen & Controlling  
account\_accountant

Installed



Angestelltenverzeichnis  
Personalwesen  
hr

Install



Verkaufs Management  
Verkaufs Management  
sale

Installed



Bewilligung von  
Stundennachweisen  
Personalwesen  
hr\_timesheet\_sheet

Install



Lager & Logistik  
Lager & Logistik  
stock

Installed



Bedarfsplanung  
Fertigung  
mrp

Installed



Beschaffung  
Beschaffung  
purchase

Installed



ToDo-Listen  
Projektmanagement  
project\_gtd

Install



Problemverfolgung  
Projektmanagement  
project\_issue

Install



Personalbeschaffung  
Personalwesen  
hr\_recruitment

Install



Urlaubsmanagement  
Personalwesen  
hr\_holidays

Install

## ab Version 7.0 mit Market Place für Apps

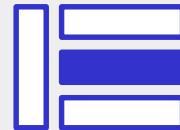
**Spesen-Verwaltung**Personalwesen  
hr\_expense[Install](#)**Anlagenverwaltung**Finanzen & Controlling  
account\_asset[Install](#)**Personalabrechnung**Personalwesen  
hr\_payroll[Install](#)**Personalbeurteilung**Personalwesen  
hr\_evaluation[Install](#)**Vertragsverwaltung**Verkaufs Management  
account\_analytic\_analysis[Install](#)**Vertragsarbeiten**Verkaufs Management  
analytic\_user\_function[Install](#)**Essensbestellungen**Extra Werkzeuge  
lunch[Install](#)**OpenOffice Report Designer**Berichtswesen  
base\_report\_designer[Install](#)**Bericht-Designer**Extra Werkzeuge  
report\_designer[Install](#)**Helpdesk**CRM Management  
crm\_helpdesk[Install](#)**Webkit Report Engine**Berichtswesen  
report\_webkit[Install](#)**Webkit Berichtsbeispiele**Berichtswesen  
report\_webkit\_sample[Install](#)

## Open-Source ERP-Systeme – Fazit

	OpenERP	ADempiere
Erlern- und Verständlichkeit	Sehr gut	Sehr schlecht
Ergonomie	Gut bis sehr gut	mittelmäßig
Programmunterstützung	Gut bis sehr gut	Eher schlecht
Erweiterbarkeit	Sehr gut	schlecht
Funktionale Anforderungen	86%	66%

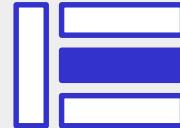
## Auswahl BISe – Kriterien (angelehnt an [Stahlknecht])

- Funktionalität des BISe? Prozesse im BIS!
- Integration in Anwendungslandschaft
- Aufwand für:
  - Parametrisierung
  - Nutzerschulung
  - Einführung
  - (Daten)migration
- Zukunftsorientierung
- Zuverlässigkeit
- Benutzerfreundlichkeit
- Kosten:
  - Kaufpreis (Lizenz)
  - Wartungskosten
  - Kosten für Erweiterungen
- Größe des Anbieters
- Branchenerfahrung
- geographische Nähe



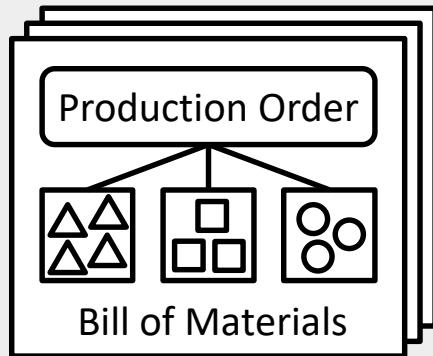
## Einführung BISe - Schritte

- Zieltermin und Zeitplan
- Auswahl der Key-User
- Schulung der Key-User
- Erweiterungen
- Testfälle
- Tests in (Integrations)workshops
- Datenmigration: Stamm und/oder Bewegungsdaten, manuell vs. automatisch
- Prozessdokumentation
- Schulung der Endbenutzer
- Produktivschalten: Stichtag oder peu à peu?

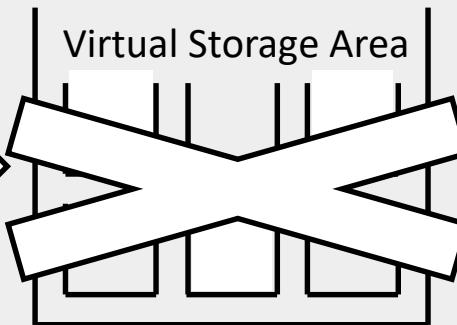


# Migration of Transaction Data – Production Orders [Munkelt]

Source System



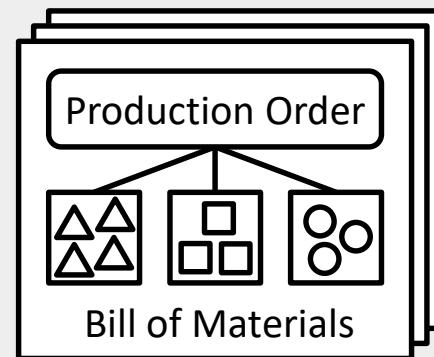
Target System



1.) Virtual Inventory Taking  
(Target Quantity BOM -  
Open quantity BOM)

3.) Pick Material  
(Pick List)

2.) Copying Production Orders  
without Material



**Result:** equivalent WIP and stock in source and target system

**Further problems:**

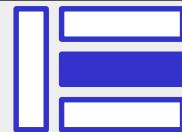
- Partially completed production orders
- Multi-level production orders
- DAGs to multitrees and vice versa
- Phantom parts which have to be broken down, ...

## Maintenance of ERP-Systems [Munkelt]

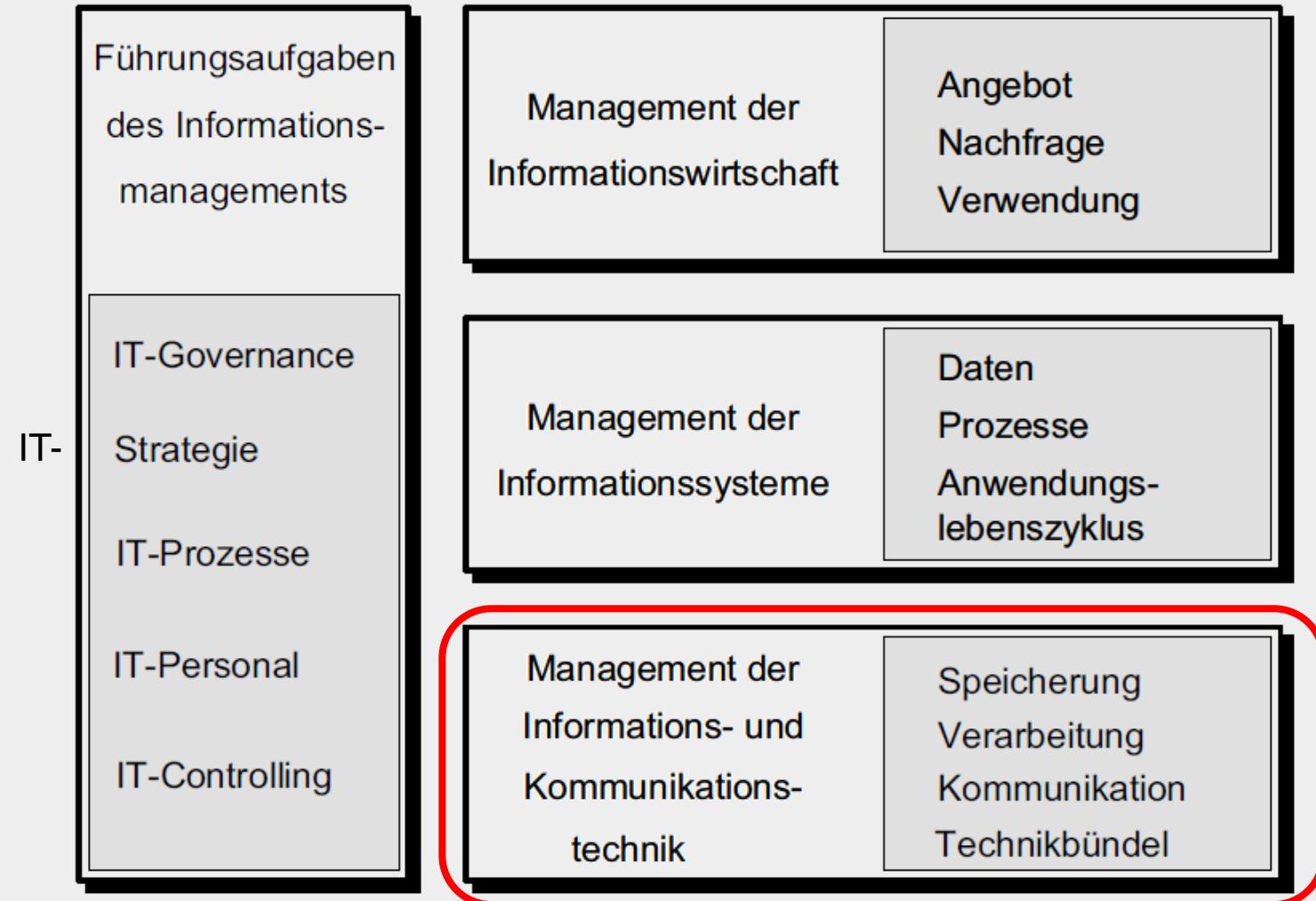
- Required manpower for system maintenance:

$$\text{number of ERP specialists} = \left\lceil \frac{\text{number of users}}{100} \right\rceil$$

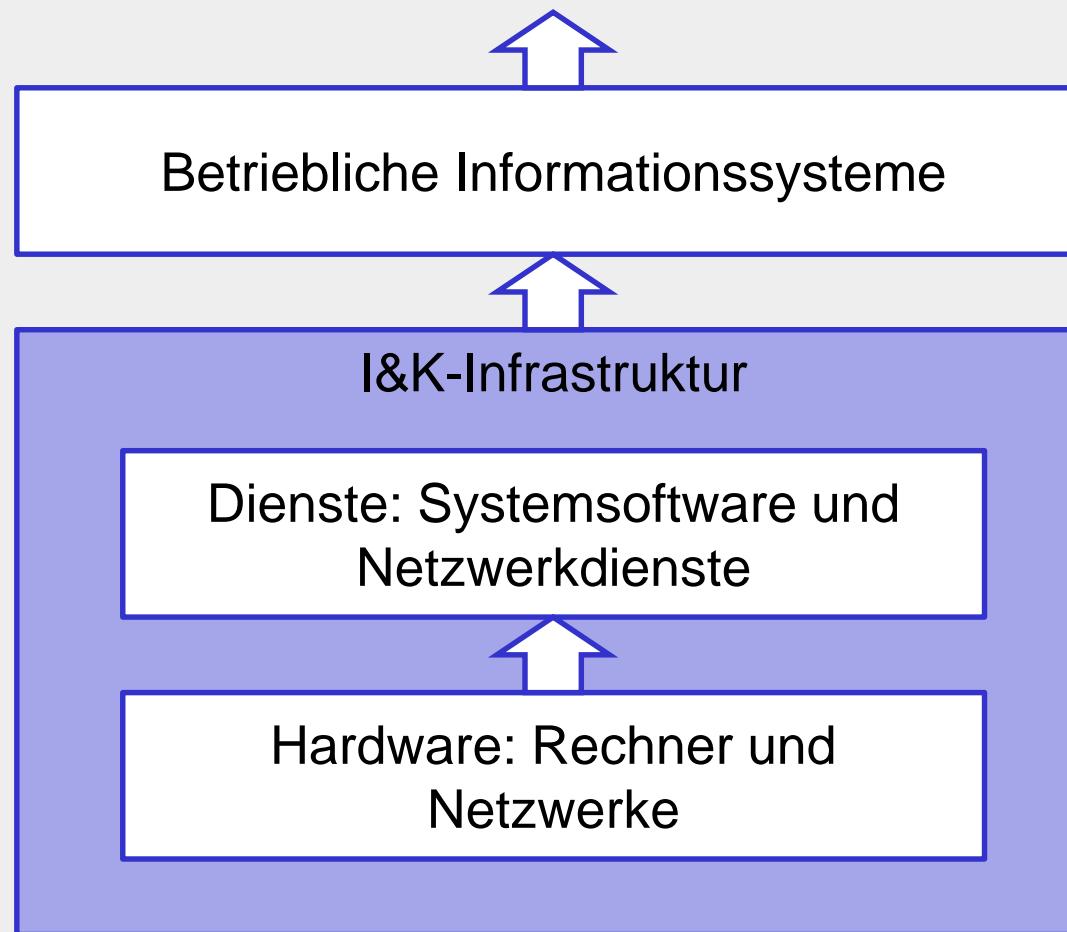
- Incident management and external support:
  - Incident classification and initial support: within 24 hours
  - Resolutions: within 2 work days for 85 % of the issues
- Annual maintenance fee: 15 – 25 % of license price
- Don't skip releases unless there are reasons!



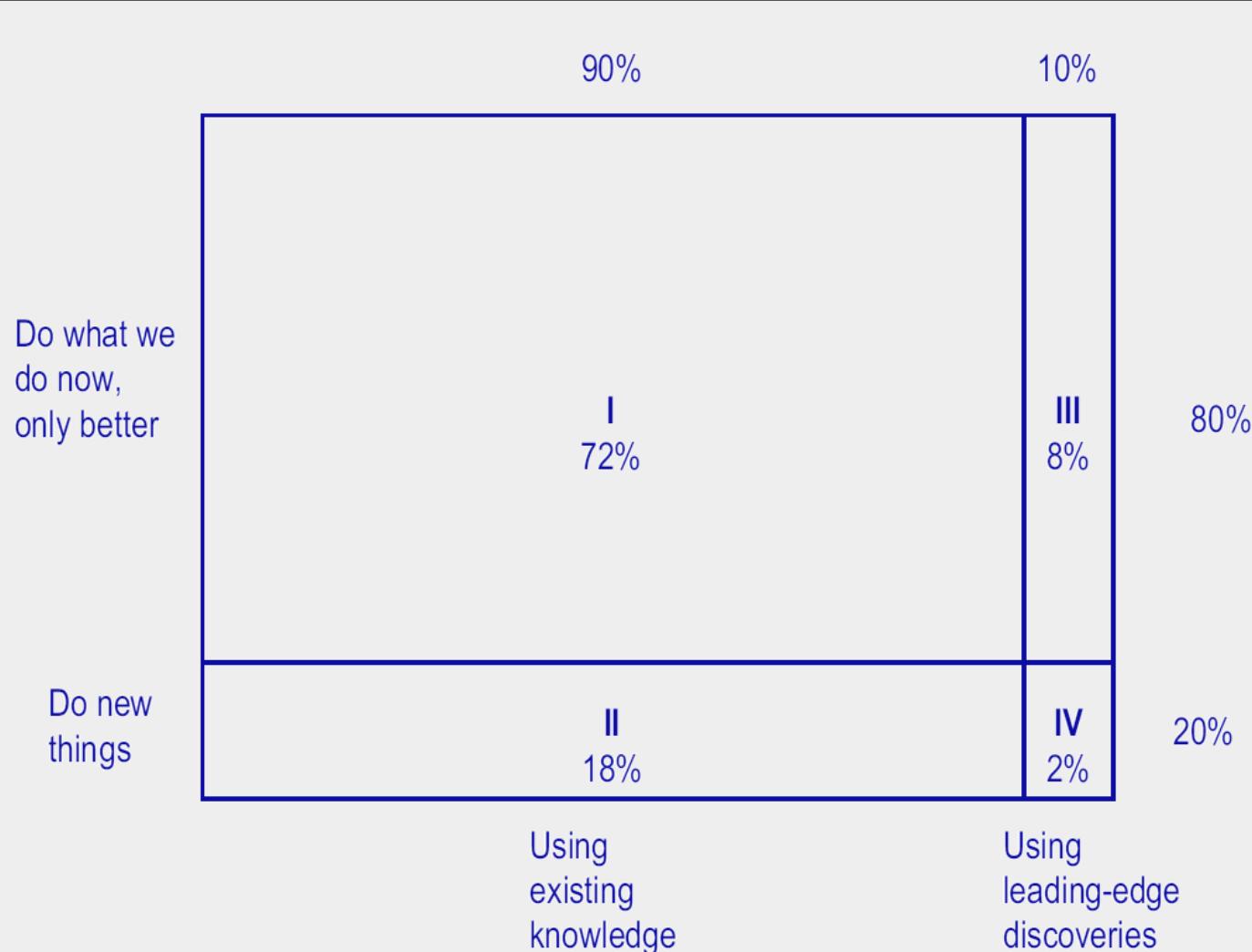
## „Referenzmodell des IM“ [Krcmar]



## I&K-Technik → I&K-Infrastruktur (angelehnt an [Minas])



## Neue oder alte I&K-Technik? [Phillips]



# Management der I&K-Technik

## Vorhandene I&K-Technik

- Installationsmanagement
- Reparatur
- Ersatz
- Helpdesk
- Bestandsführung
- Lizenzmanagement
- **Datenschutz**
- **Datensicherheit**
- **Katastrophenschutz**
- ...

## Neue I&K-Technik

- Marktbeobachtung
  - Technisch
  - Monetär
- Auswahl relevanter Technologie
- Bestimmen des optimalen Einsatz-/Ersatzzeitpunktes
- Beschaffen neuer I&K-Technik
- **Einführen und Durchsetzen von Standards**
- Entwurf einer Strategie
- ...



## Vorteile von Standards

- Geringere Kommunikationskosten
- Schutz von Investitionen
- Vielseitig einsetzbar
- Geringer Einarbeitungsaufwand
- Schnellere Kommunikation
- Weniger Medienbrüche
- Weniger Fehler
- Höhere Markttransparenz → besserer Wettbewerb
- ...



## Grundanforderungen an IT-Sicherheit [Voydock et al.]



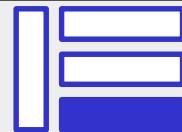
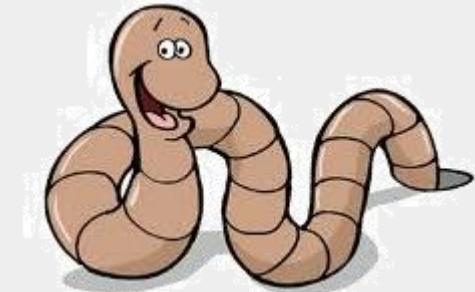
Anforderung	Inhalt
Vertraulichkeit	<ul style="list-style-type: none"><li>• Keine Einsichtnahme Dritter</li><li>• Anonymität</li><li>• Unbeobachtete Kommunikation</li><li>• Keine räumliche Nachverfolgbarkeit</li></ul>
Integrität	<ul style="list-style-type: none"><li>• Daten in sich und untereinander konsistent</li><li>• Nachrichten vollständig und unverändert</li></ul>
Verfügbarkeit	<ul style="list-style-type: none"><li>• Daten und Kommunikationsmedien verfügbar, wenn gewünscht</li><li>• Kein Verlust einmal vorhandener Daten</li></ul>
Zurechenbarkeit	<ul style="list-style-type: none"><li>• Nachweis, dass gesendet/empfangen</li><li>• Nachweis, dass tatsächlich Absender</li></ul>

## Gefährdung der IT-Sicherheit durch ...

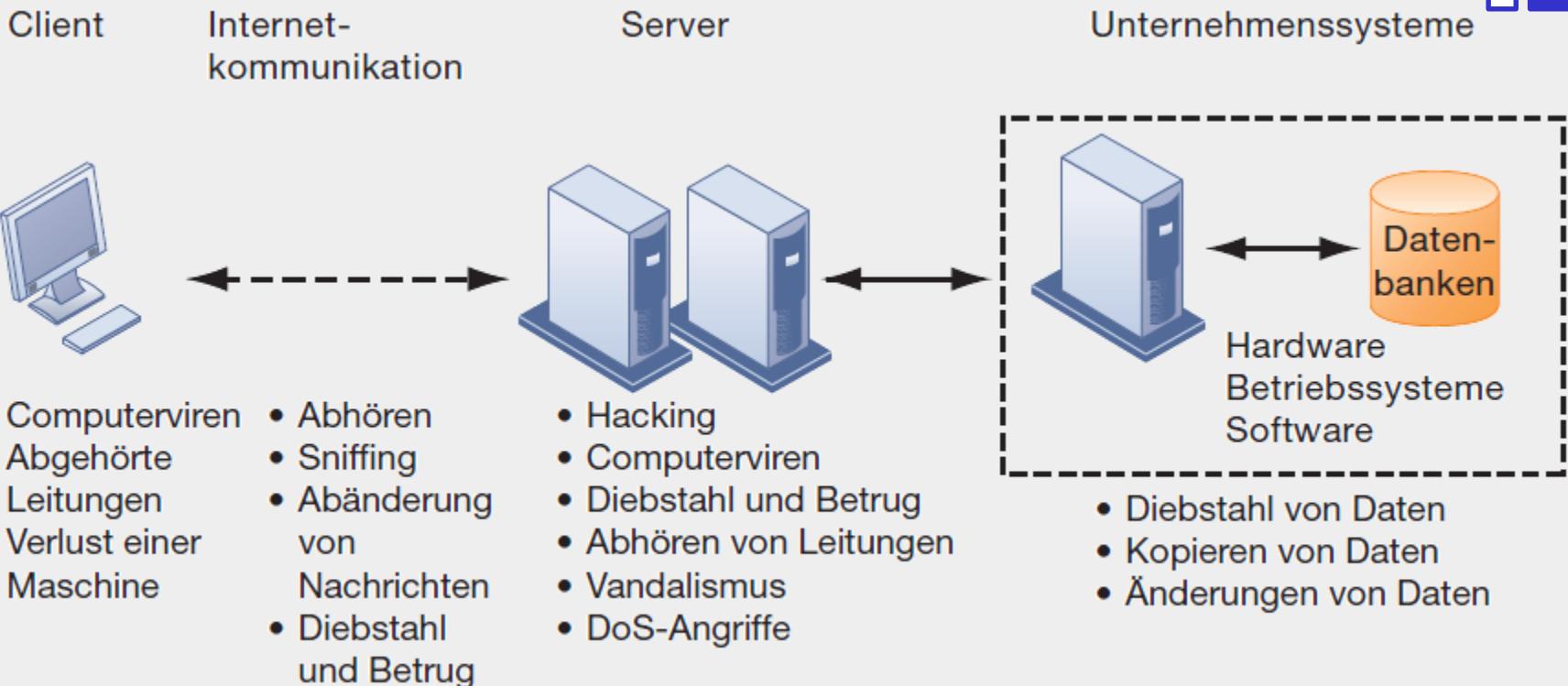
- Veraltete Hard- und Software
- Unzureichende Sicherheitskonzepte
- Schlecht konfigurierte Systeme
- Unsichere Vernetzung
- Sorgloser Umgang mit Passwörtern
- Schädliche Software
- ...
- Gefährdung = Bedrohung + Schwachstelle

## Malware (Malicious Software)

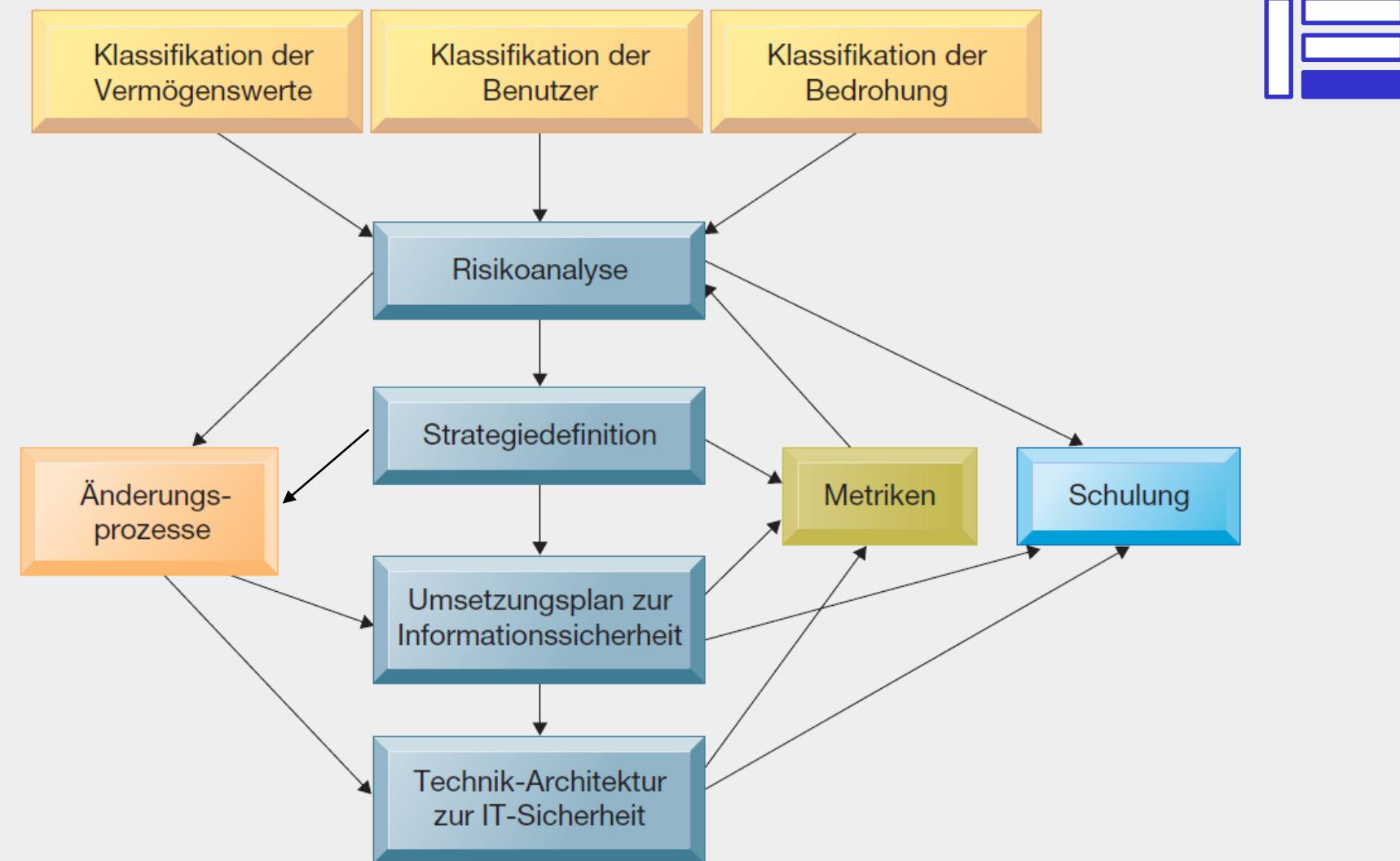
- Viren:
  - Behindern Arbeit mit Computer
  - Schädigen Computer
  - Passive Übertragung durch Wirtprogramm/-datei
- Würmer:
  - Verbreitet sich aktiv; infiziert nicht
  - Wesentlich schnellere Ausbreitung
- Trojanische Pferde:
  - Software mit geheimer Funktionalität
  - Öffnet Systeme mit Benutzerberechtigungen für Zugriff von außen
  - Installation von Spyware (Adware, Keylogger)



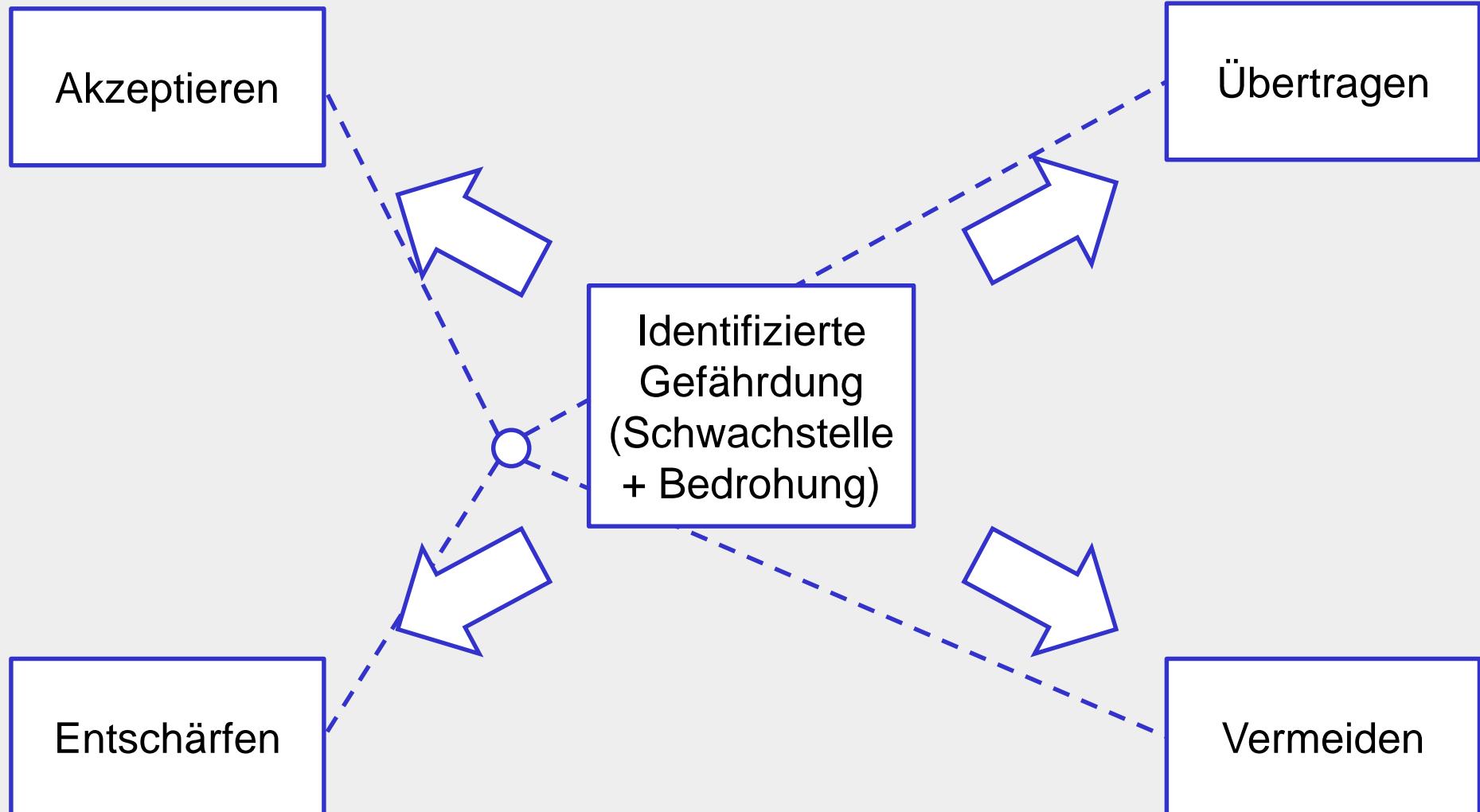
# Sicherheitsprobleme in IT-Systemen [Laudon]



# Informationssicherheitsprogramm [Burton-Group]



## IT-Sicherheitsmanagement



# IT-Grundschutzkataloge [BSI]



BSI: IT-Grundschutzkataloge - Mozilla Firefox

Datei Bearbeiten Ansicht Chronik Lesezeichen Extras Hilfe

BSI: IT-Grundschutzkataloge

BSI bund.de https://www.bsi.bund.de/DE/Themen/weitereThemen/ITGrundschutzKataloge/itgrundschutzkataloge\_node.html

Bundesamt für Sicherheit in der Informationstechnik

Das BSI Themen Aktuelles Presse Publikationen

Kontakt Impressum Service RSS Sitemap English

HTTPS TCP E G S H B S O V I WWW

IT-Grundschutzkataloge

Startseite IT-Grundschutz Inhalt Hilfsmittel Überblickspapiere Bezugssquellen FAQ Registrierung / Newsletter Download Kontakt

Suche Suchbegriff eingeben ▶

Startseite > Themen > IT-Grundschutzkataloge

## IT-Grundschutzkataloge

Die aktuelle 12. Ergänzungslieferung steht derzeit nur als PDF-Version zur Verfügung.  
Download: [IT-Grundschutzkataloge 12. Ergänzungslieferung \(Dokument ist nicht barrierefrei\) \(PDF, ca. 52,2 MB\)](#)

Die im Folgenden aufrufbare HTML-Version ist noch auf dem Stand der 11. Ergänzungslieferung.

### Einführung

- [Vorwort](#)
- [Neues in der 11. Ergänzungslieferung der IT-Grundschutzkataloge](#)
- [Dankesworte](#)

### IT-Grundschutz - Basis für Informationssicherheit

- [Warum ist Informationssicherheit wichtig](#)
- [IT-Grundschutz: Ziel, Idee und Konzeption](#)
- [Aufbau der IT-Grundschutzkataloge](#)
- [Anwendungsweisen der IT-Grundschutzkataloge](#)

### Schichtenmodell und Modellierung

- [Modellierung nach IT-Grundschutz](#)
- [Zuordnung anhand Schichtenmodell](#)

## Bausteine

- Übergreifende Aspekte
- Infrastruktur
- IT-Systeme
- Netze
- Anwendungen

## Gefährdungskataloge

- Höhere Gewalt
- Organisatorische Mängel
- Menschliche Fehlhandlungen
- Technisches Versagen
- Vorsätzliche Handlungen

## Maßnahmenkataloge

- Infrastruktur
- Organisation
- Personal
- Hardware und Software
- Kommunikation
- Notfallvorsorge

## Hilfsmittel

- Checklisten und Formulare
- Muster und Beispiele
- IT-Grundschutz-Beispielprofile
- Dokumentationen und Studien
- Informationen externer Anwender

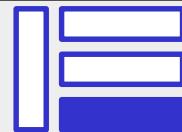
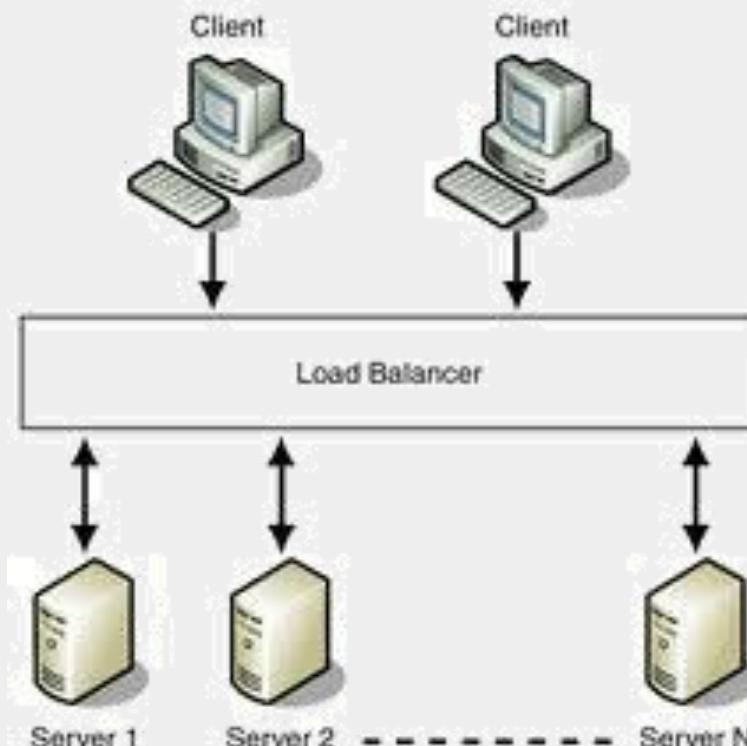
## Ausgewählte Maßnahmen zur Gewährleistung der IT-Sicherheit

- Regelmäßige Kontrollen
- Benutzerpflege
- Firewall, z. B.
- Intrusion Detection
- VirensScanner
- Sensibilisierung der Mitarbeiter
- Passwort-Management
- Verschlüsselung
- ...



## Hochverfügbarkeit

- Lastausgleich
- Spiegelung (Havariesystem)
- ...
- Fremdvergabe

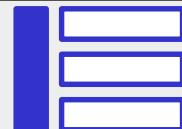
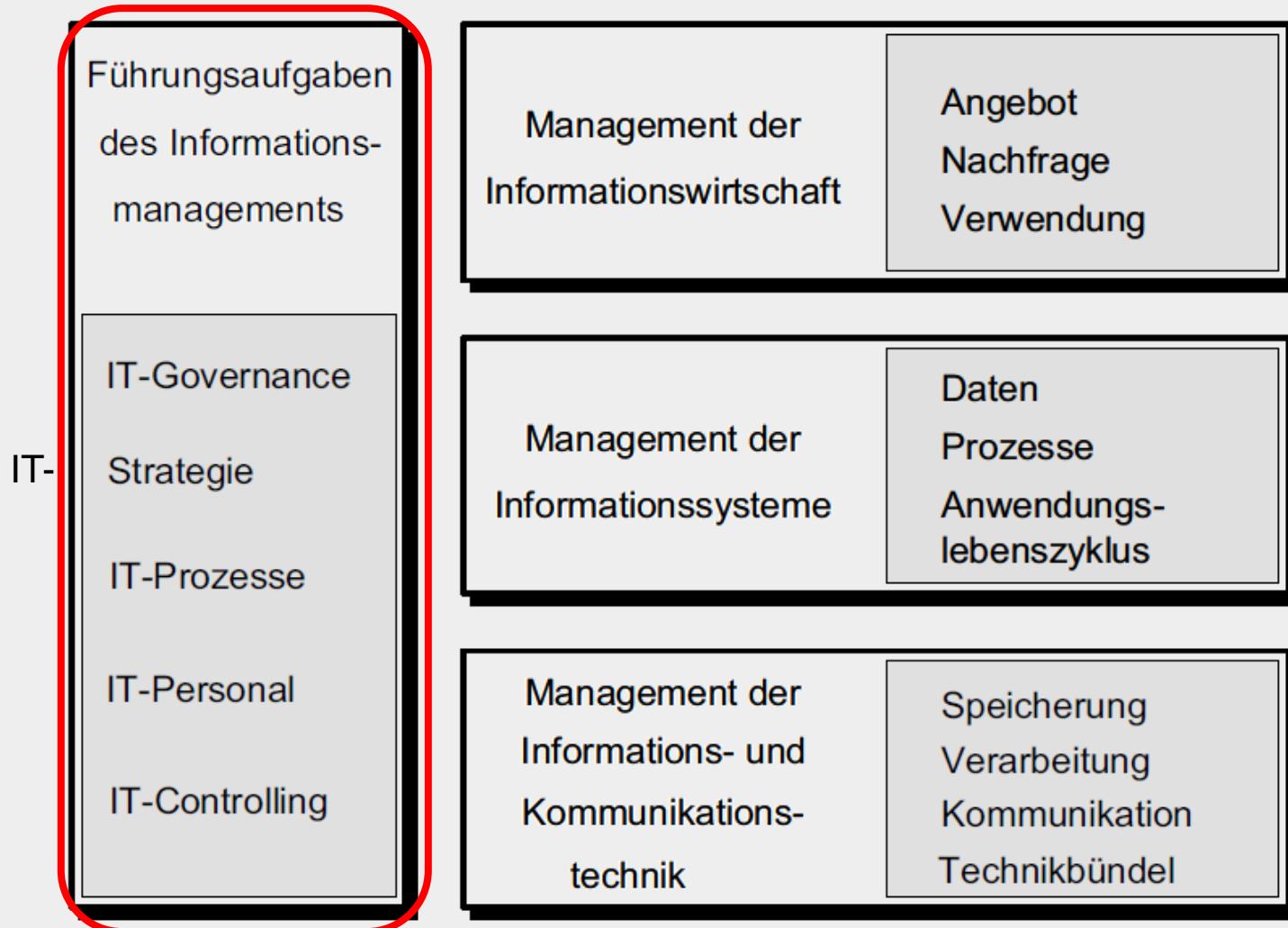


## Gutes Passwort?

- Geschützte Eingabe
- Sicher aufbewahren Passwort-Manager
- Bekannt geworden? → Sofort ändern!
- Je länger, desto besser (Länge wichtiger als Inhalt)
- Buchstaben, Zahlen **und** Sonderzeichen
- Keine erratbaren oder persönlichen Bestandteile
- Anfangsbuchstaben eines Satzes mit Zahlwörtern: „JDsi1/26Ua.“
- Passwort regelmäßig ändern
- Gleiches Passwort für mehrere Zugänge?



## „Referenzmodell des IM“ [Krcmar]



## IT-Governance

- „specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT“ [Weill et al.]
- Festlegen, wer was entscheiden darf und wer für die (Folgen der) Entscheidung verantwortlich ist, um ein erstrebenswertes Verhalten bei der Anwendung der IT anzuregen
- Effektive Steuerung (Regelung) des IT-Einsatzes
- Einklang zwischen Unternehmens- und IT-Strategie
- Beitrag der IT zum Unternehmenserfolg ausweisen
- ...

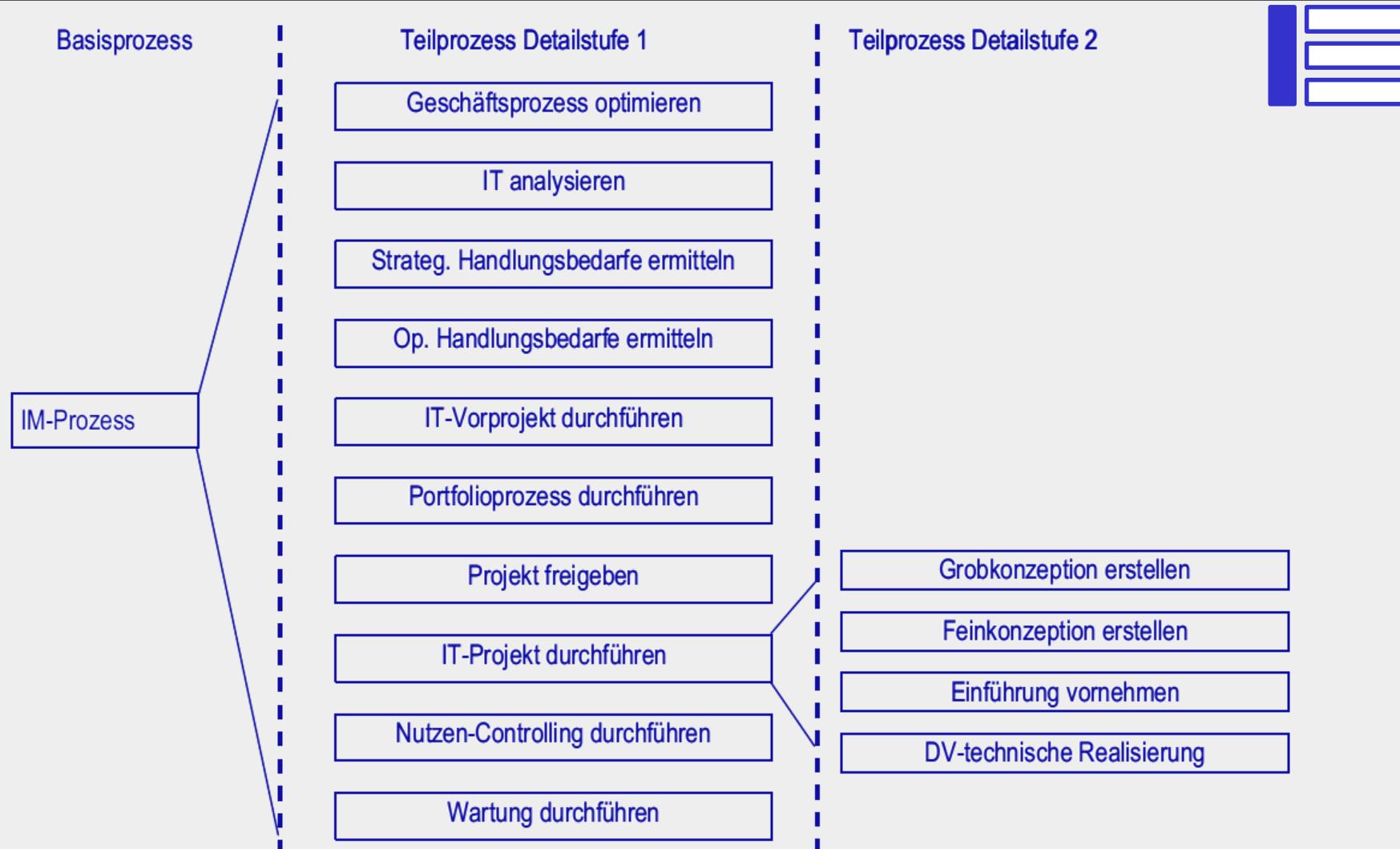


## IT-Strategien (Ziele) [Feeny] und Intensität [Krcmar]

- Bessere Geschäftsprozesse
- Mehr Geschäftsinnovation
- Stärkere Kunden(- und Lieferanten)bindung
- Höhere Markteintrittsbarrieren (für Marktbegleiter)
- Aufbau einer Informationsbasis/IT-Plattform
- ...



# IT-Prozess [Krcmar]



## IT-Personalmanagement

### Anforderungen an IT-Personal

- Abstraktions- und analytisches Denkvermögen
- überdurchschnittliche Lernbereitschaft
- Anpassungsfähigkeit
- Kommunikationsfähigkeit und soziale Kompetenz
- Schulungsfähigkeit
- Teamfähigkeit
- Organisationsgeschick
- Führungsfähigkeit („Führungs kraft und Schiedsrichter“)
- Integrierende Persönlichkeit
- Kenntnis der Betriebswirtschaft und der Informatik
- ...

### Aufgaben [Krcmar]

- Personalbestandsanalyse
- Personalbedarfsbestimmung
- Personalbeschaffung
- Personalentwicklung
- Personaleinsatzmanagement
- Personalführung
- Personalkostenmanagement
- „Personalfreisetzung“?

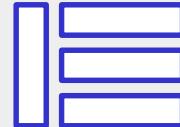
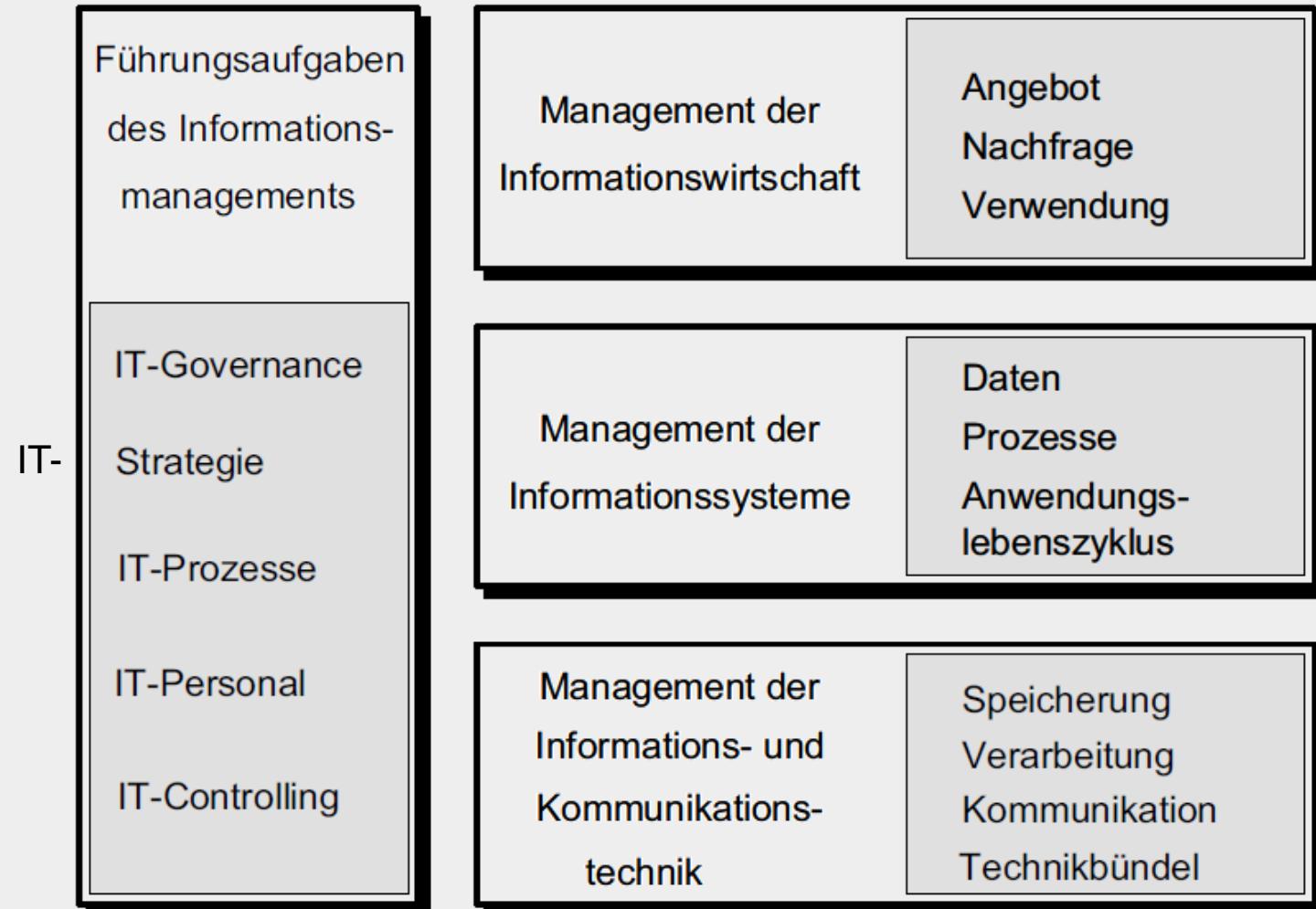


## IT-Controlling

- Projektcontrolling → Projektmanagement
- Controlling der IT-Infrastruktur
  - Kosten ← relativ gut zu bestimmen
  - Nutzen ← ex ante fast unmöglich, ex post schwierig zu bestimmen
- Beschränkung auf Kosten
- Nutzen über „technizitäre Ersatzziele der PPS“ bestimmen:
  - Termintreue
  - Durchlaufzeit
  - Kapazitätsauslastung
  - Kapitalbindung
- Ersatzziele trotzdem nur schwer vereinbar und schwer auf Kosten abzubilden
- Viele (Forschungs)ansätze, aber oft zu abstrakt



## „Referenzmodell des IM“ [Krcmar]

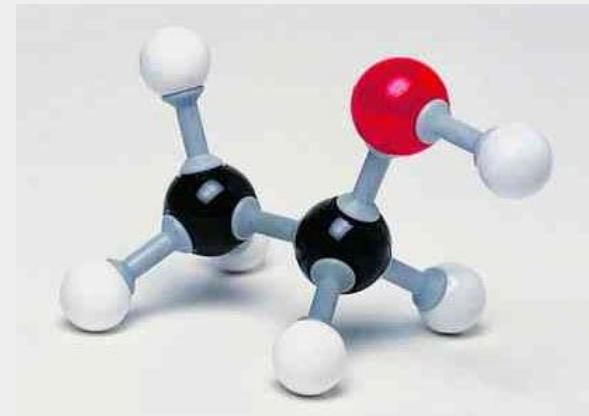


## Gliederung

1. Inhalte und Aufgaben der Wirtschaftsinformatik
2. Grundlagen der Informatik und der Informationstechnik
3. Informationsmanagement
- 4. Modellierung**
5. Datenbanken
6. Softwareentwicklung
7. Betriebliche Informationssysteme

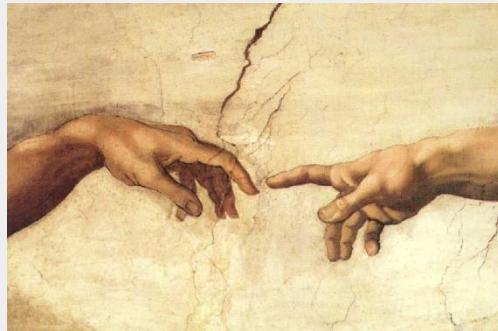
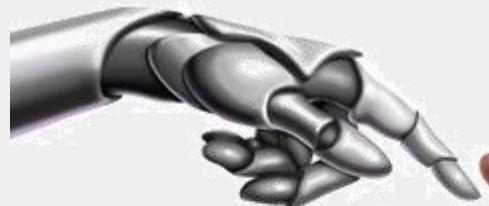
## Modell

- Abbild eines Realitätsausschnittes
- Merkmale (u. a.):
  - Abbildung – eines natürlichen oder eines künstlichen Originals
  - Vereinfachung – nur für Einsatzzweck relevante Eigenschaften abgebildet
  - Pragmatismus – ersetzt Realität für einen bestimmten Einsatzzweck
- Zweck (u. a.):
  - Erklärung und Demonstration
  - Projektierung/Konstruktion
  - Verifikation
  - Planung, Optimierung, Steuerung, Regelung
  - Erkenntnis
- Zielgerichtetes Experimentieren mit dem Modell
  - ➔ Erkenntnisse, die in die Realität übertragen werden können



## Modelltypen in der Wirtschaftsinformatik

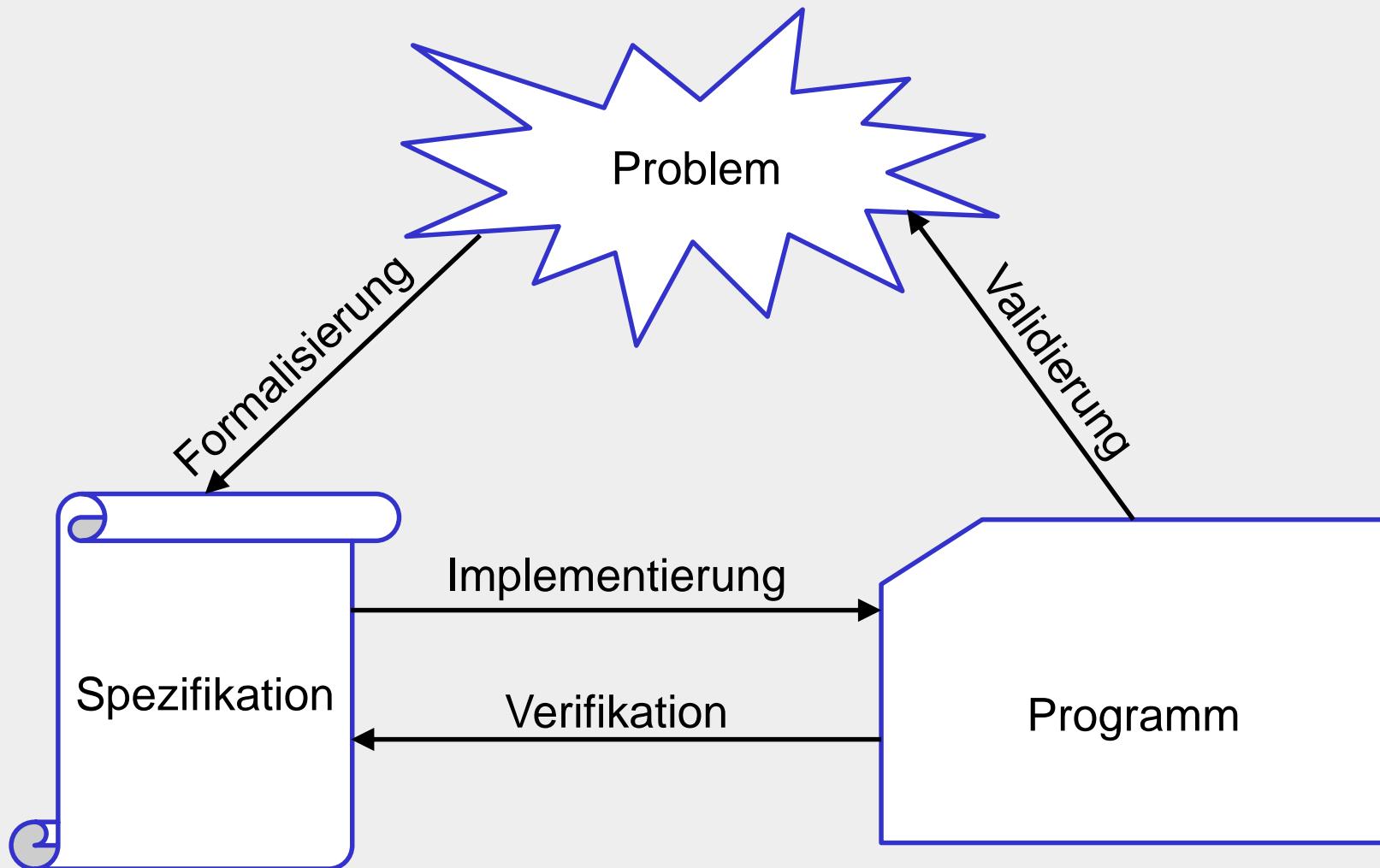
- Softwaremodelle
- Geschäftsprozessmodelle
- Modelle zur Mensch-Maschine-Interaktion
- ...



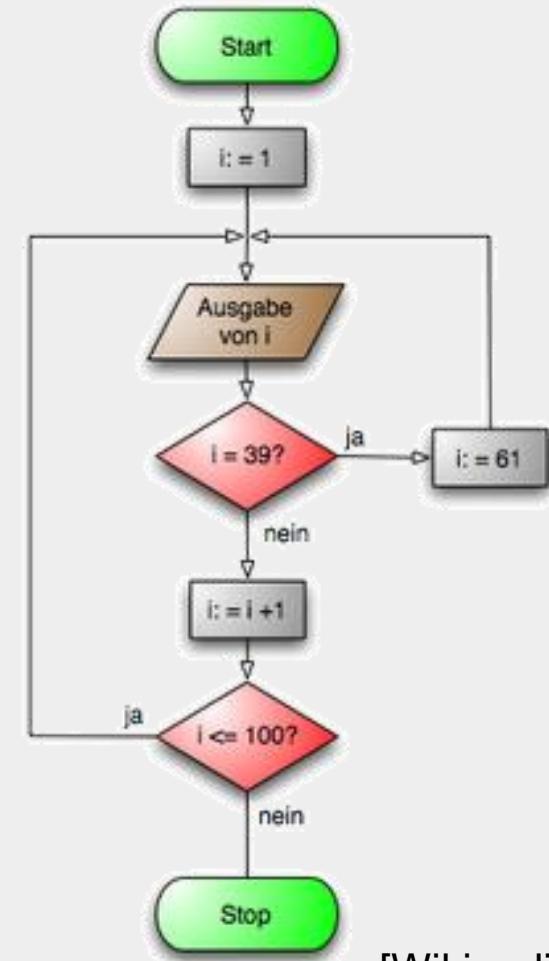
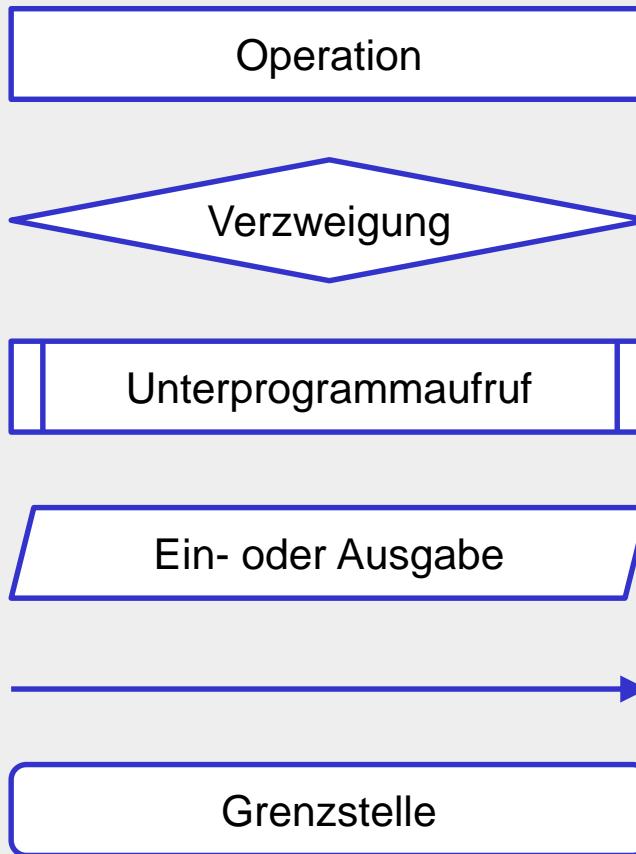
## Inhalte der SW-Modellierung

- SW-Modellierung: Erstellen eines SW-Modells
- SW-Modell
  - erlaubt das bessere Durchdringen der Programmieraufgabe und des Anwendungsbereichs,
  - bildet oft zukünftige SW ab (exploratives Modell),
  - umfasst nur relevante Aspekte der zukünftigen SW,
  - bedarf **Verifikation und Validierung**.
- einige Modelltypen:
  - Business Process Modelling Notation (BPMN),
  - Entity-Relationship-Model (ERM),
  - Unified Modelling Language (UML).

## Exkurs: Verifikation und Validierung

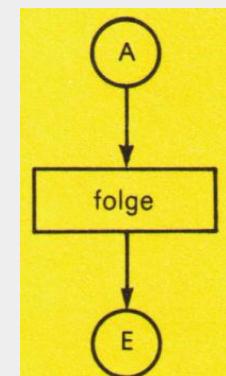
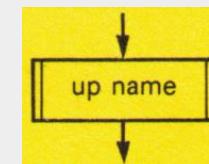
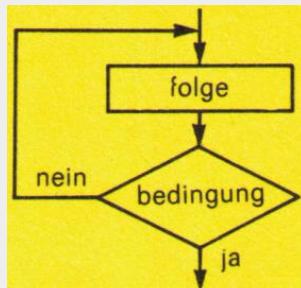
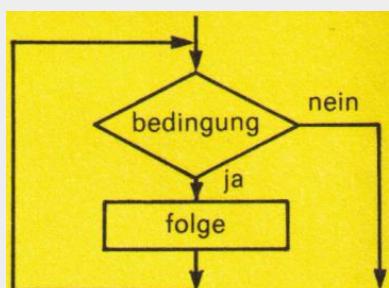
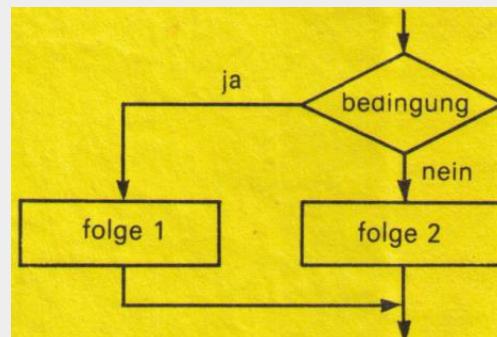
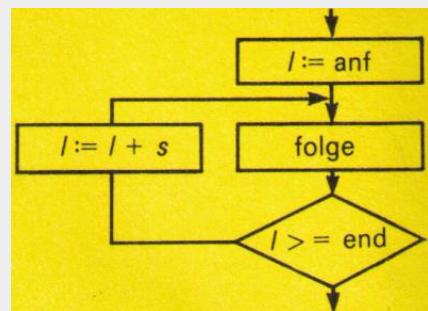
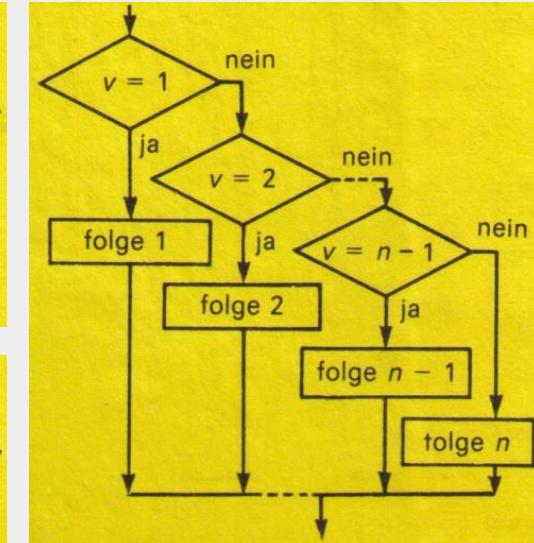
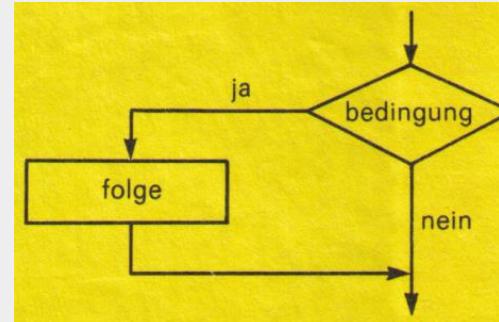
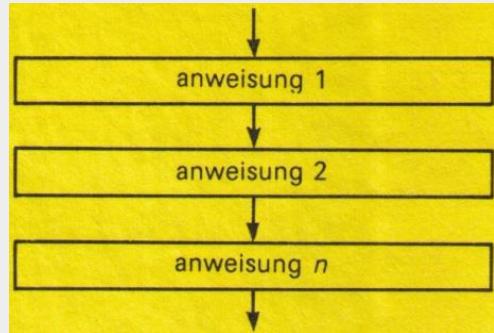


# Programmablaufplan [DIN 66001, ISO 5807]



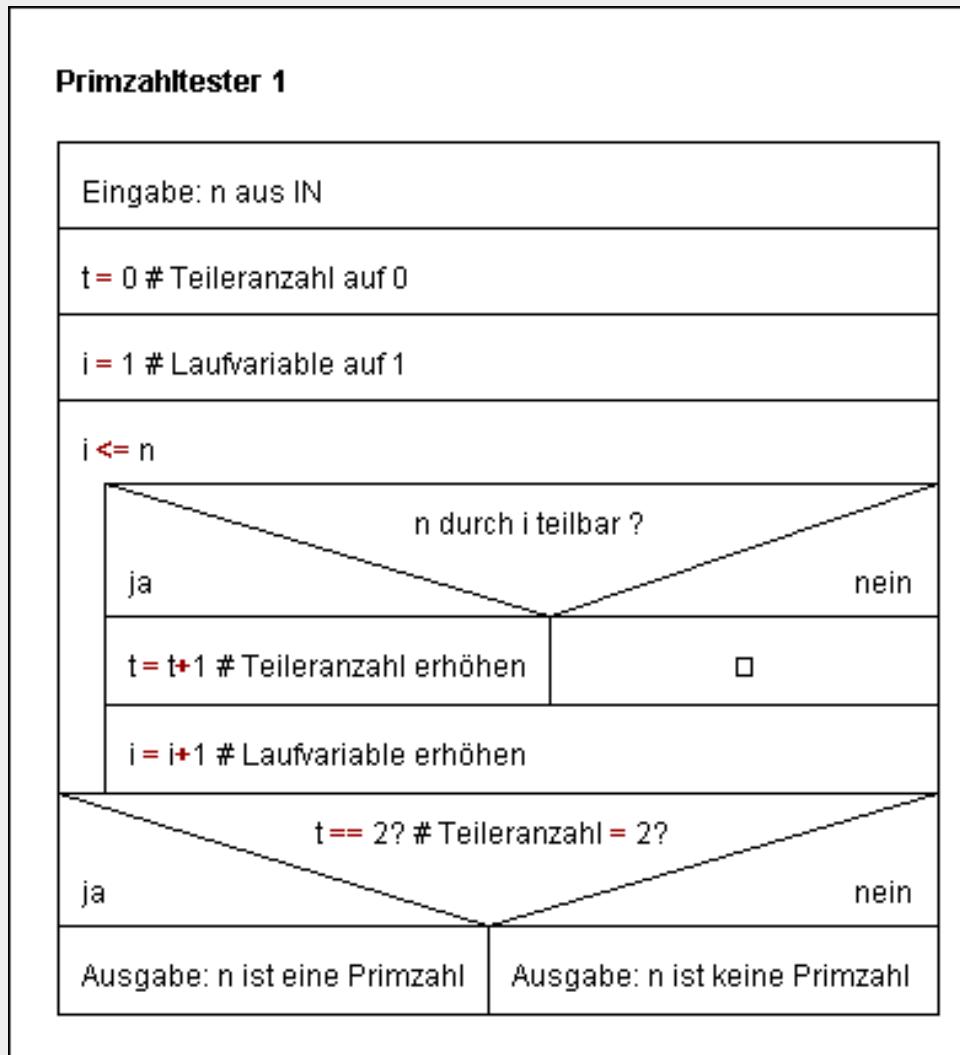
[Wikipedia]

# PAP – Sequenz, Verzweigungen, Schleifen, Unterprogramm

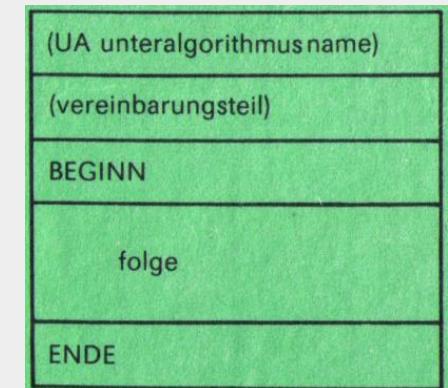
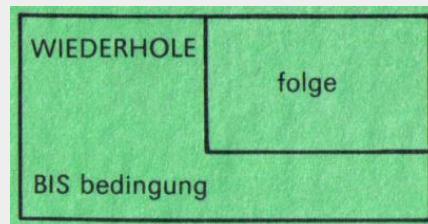
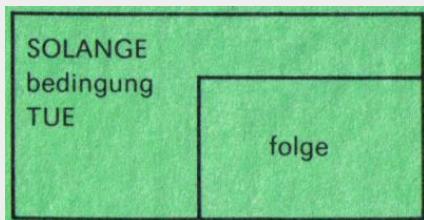
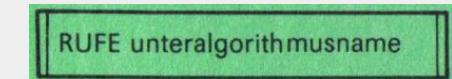
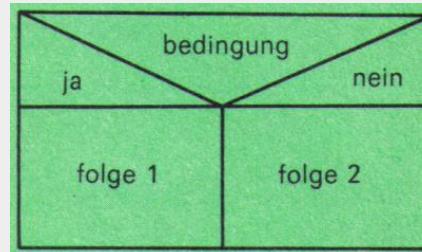
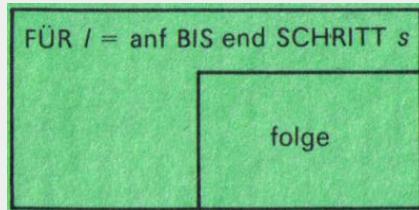
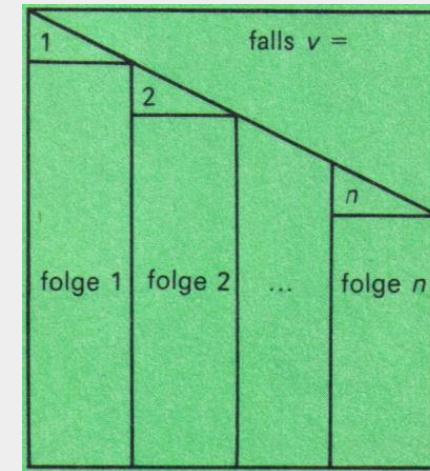
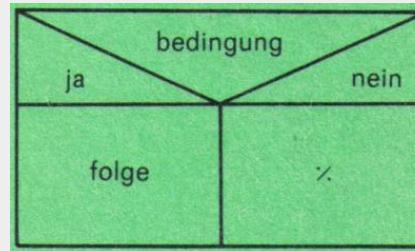
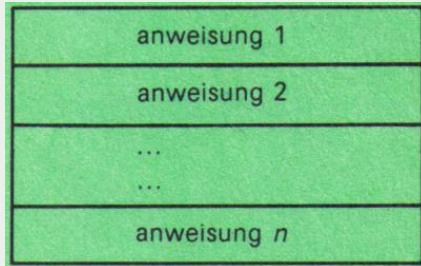


[Schilling, Töpfer]

# Struktogramm [DIN 66261]

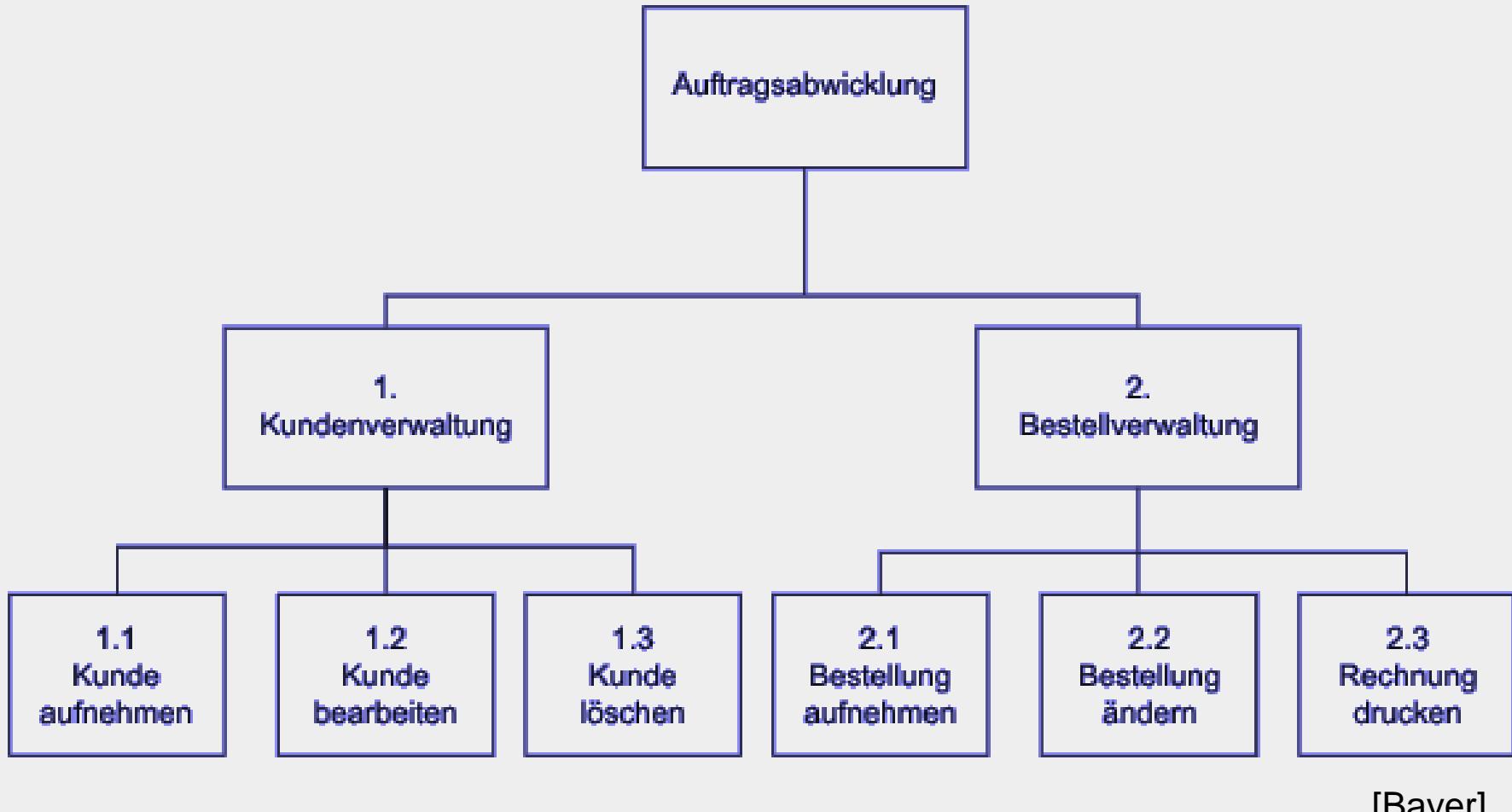


# Struktogramm – Sequenz, Verzweigung, Schleife, Unterprogramm



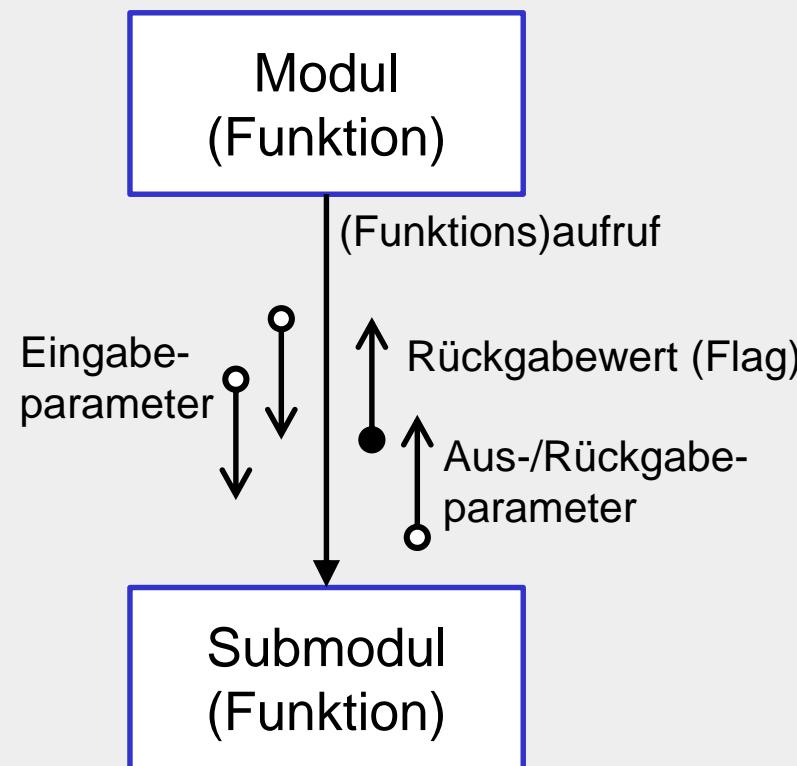
[Schilling, Töpfer]

# Funktionshierarchiediagramm bzw. Funktionsbaum

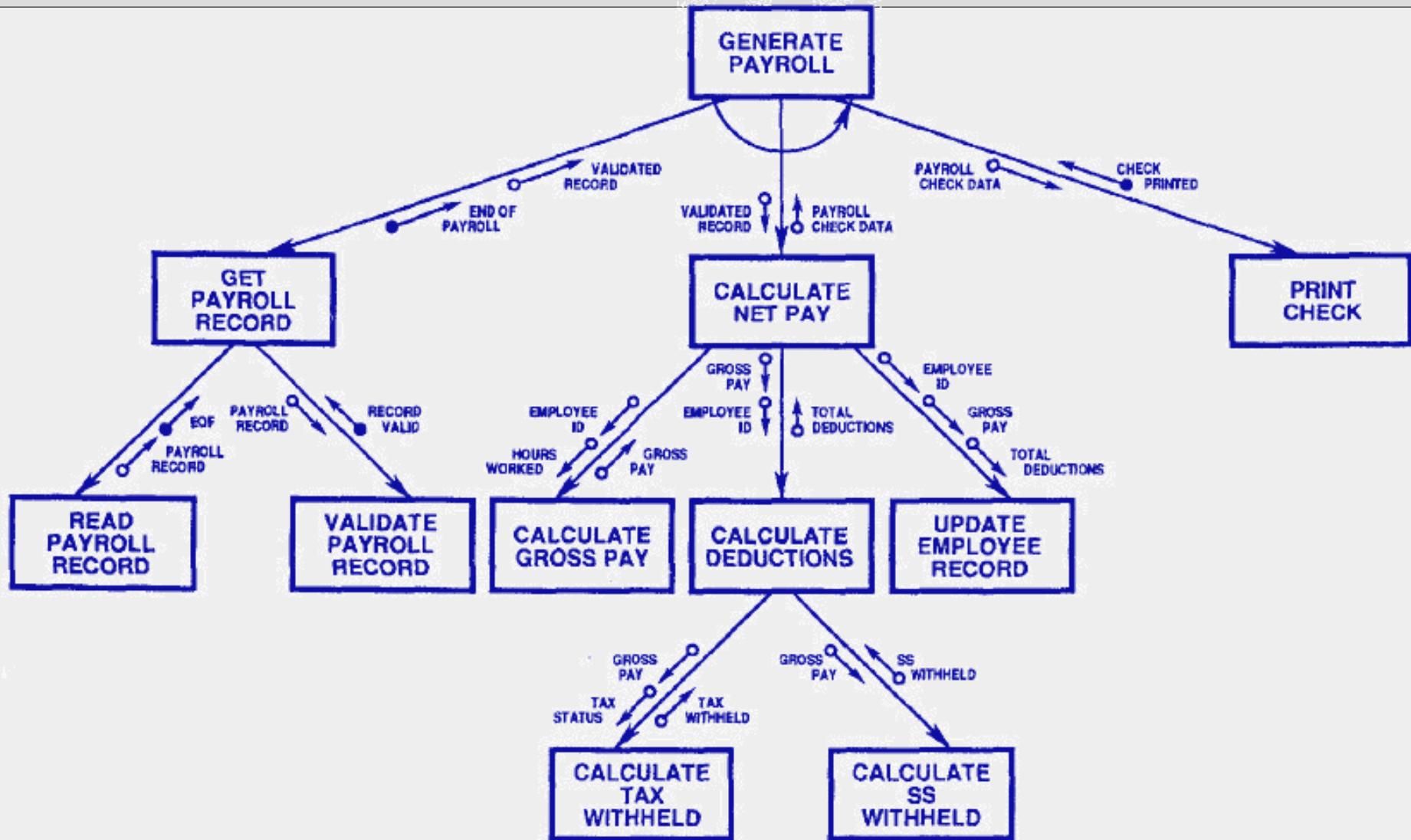


[Bayer]

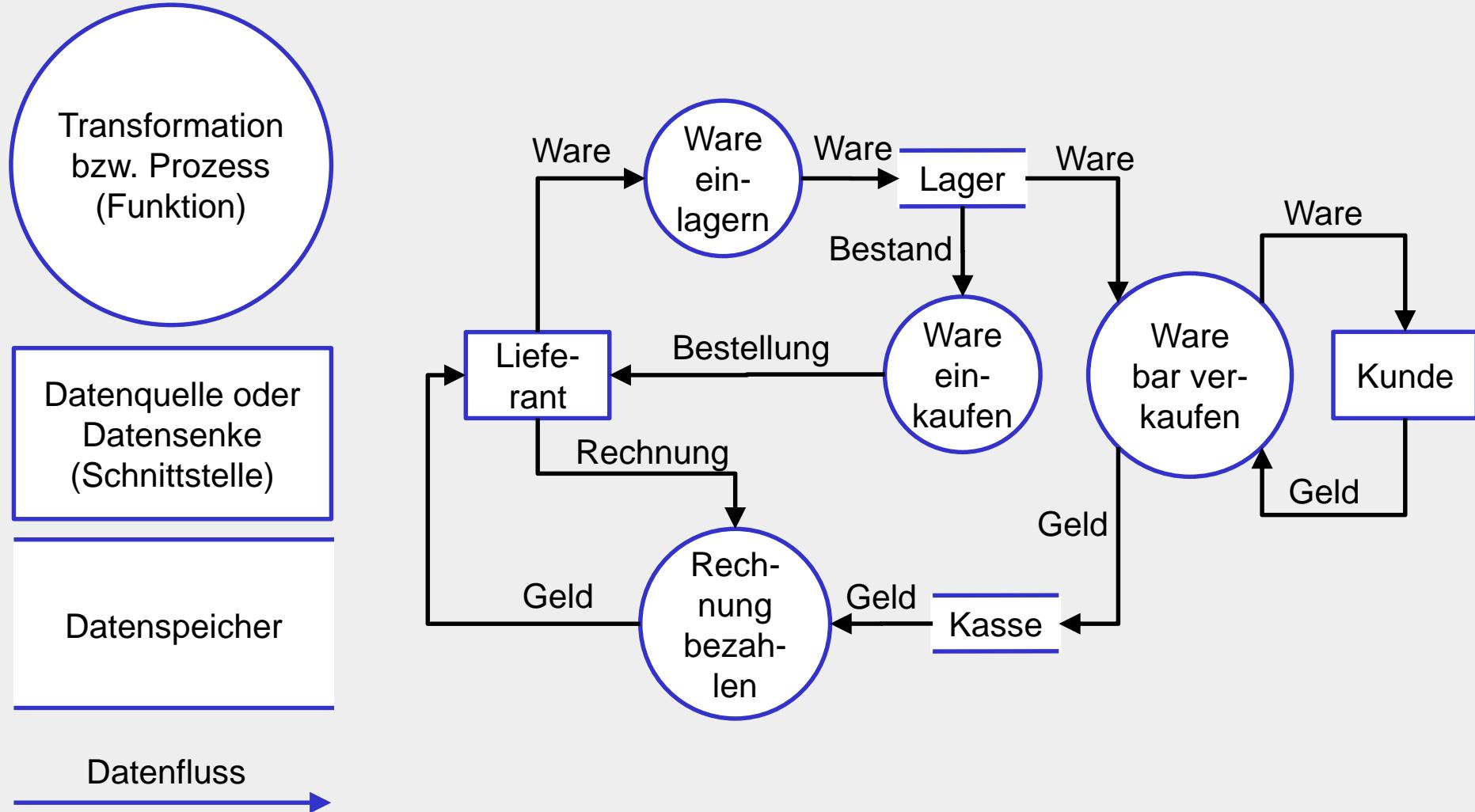
## Structured Chart (Aufrufdiagramm?)



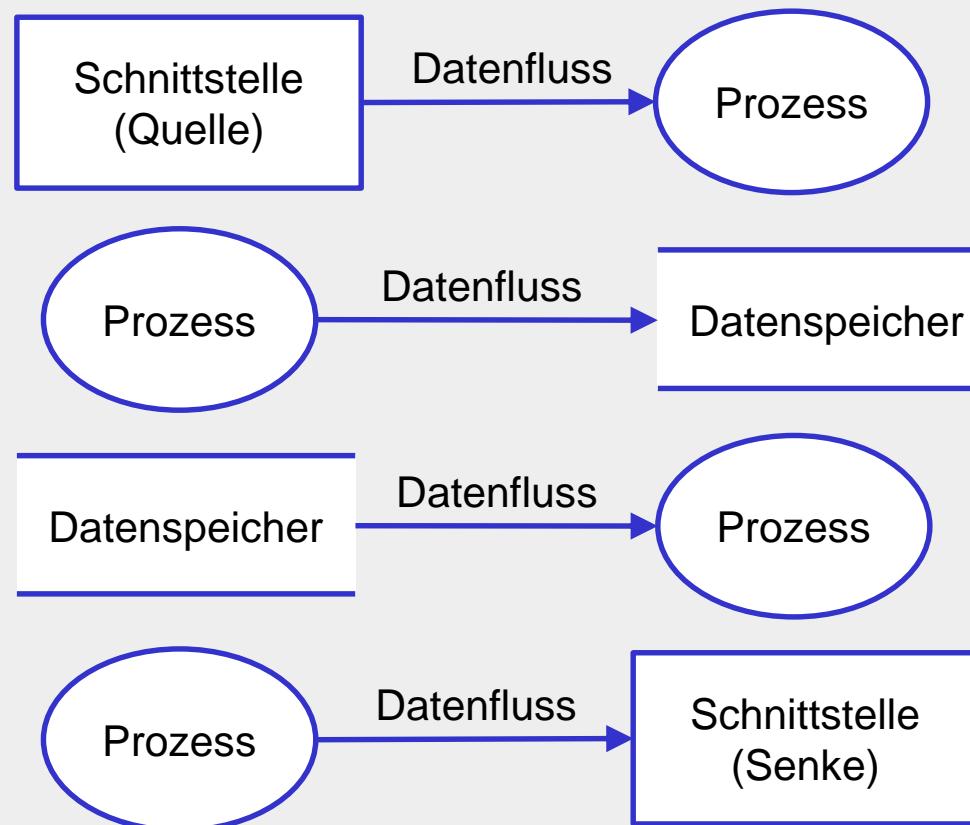
## Structured Chart – Beispiel [SANDIA]



## Datenflussdiagramm (DFD) [DIN 66001]



## DFD – erlaubte Datenflüsse



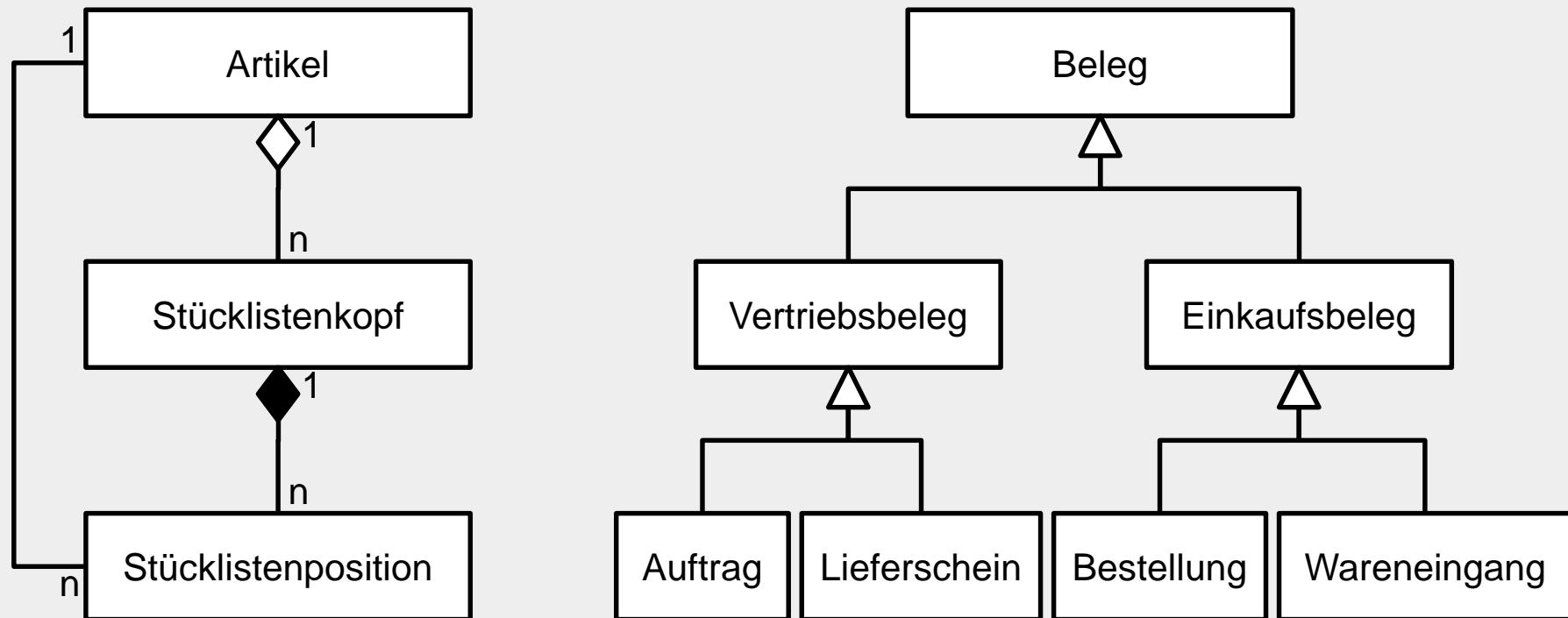
Alle anderen Datenflüsse sind verboten.

## Ausgewählte UML-Diagrammformen

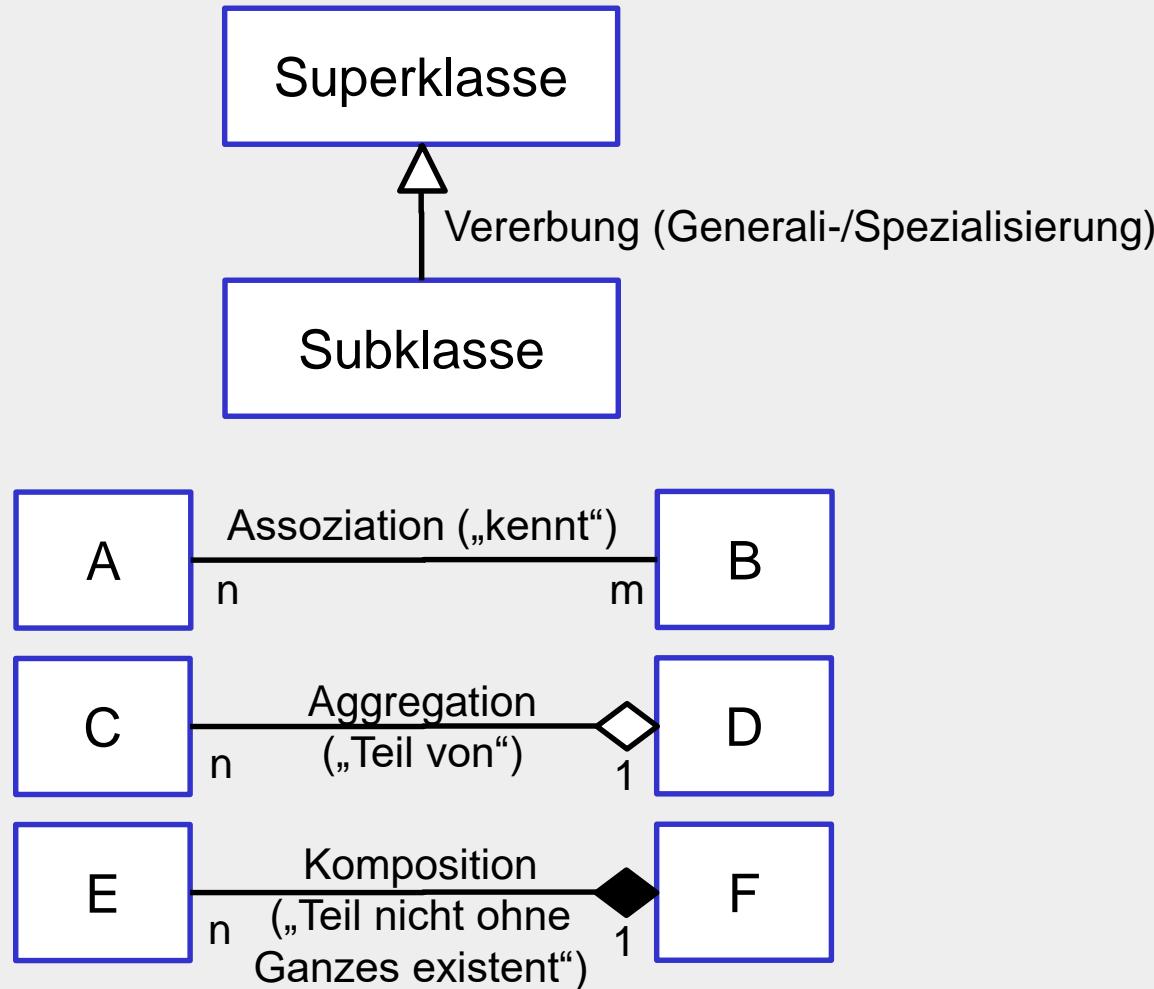
- Strukturdiagramme
  - Komponentendiagramm
  - Paketdiagramm
  - **Klassendiagramm**
  - **Objektdiagramm**
- Verhaltensdiagramme
  - Anwendungsfalldiagramm
  - Aktivitätsdiagramm
  - **Kommunikationsdiagramm**
  - **Sequenzdiagramm**
  - Zustandsdiagramm



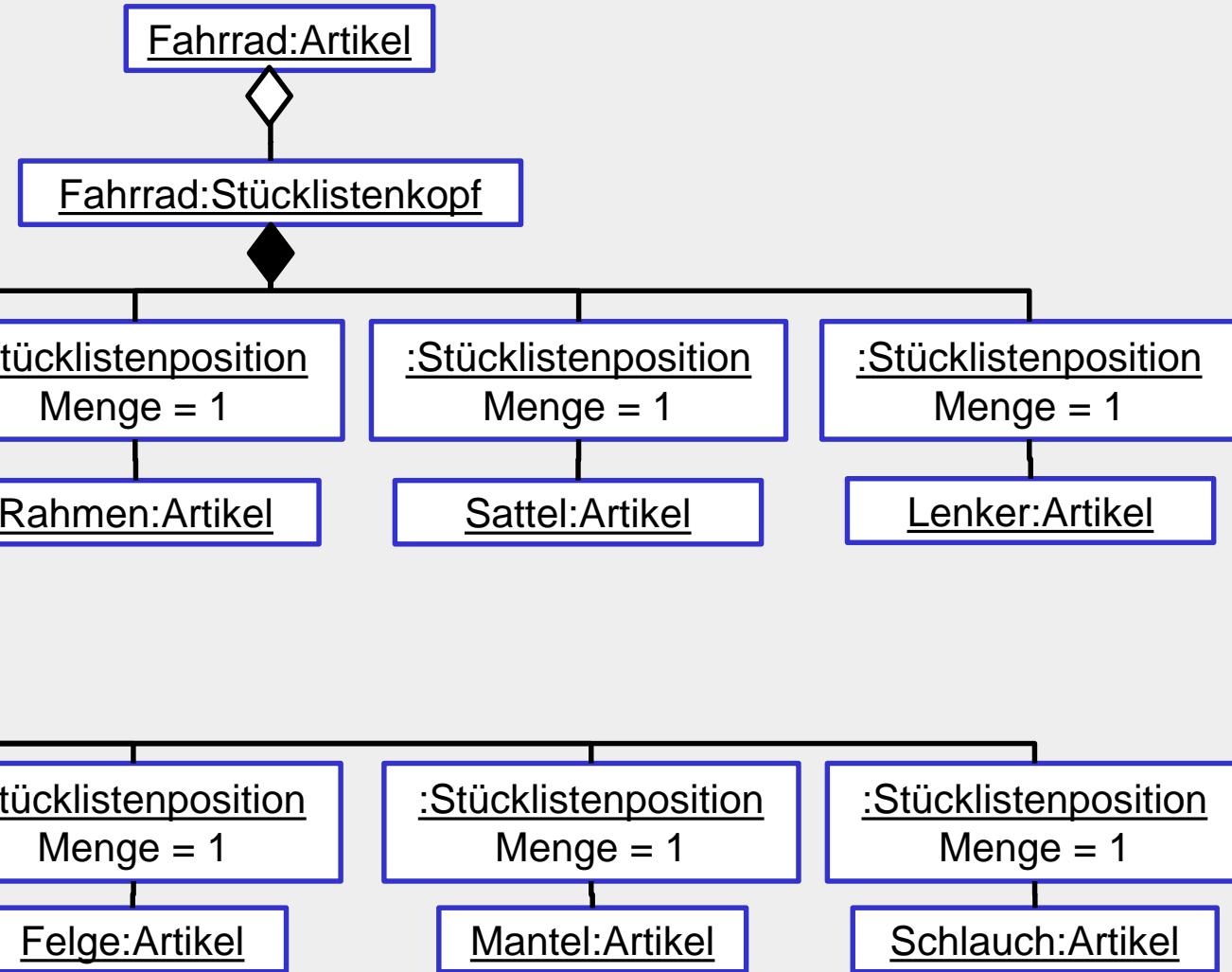
# Klassendiagramme, vereinfacht



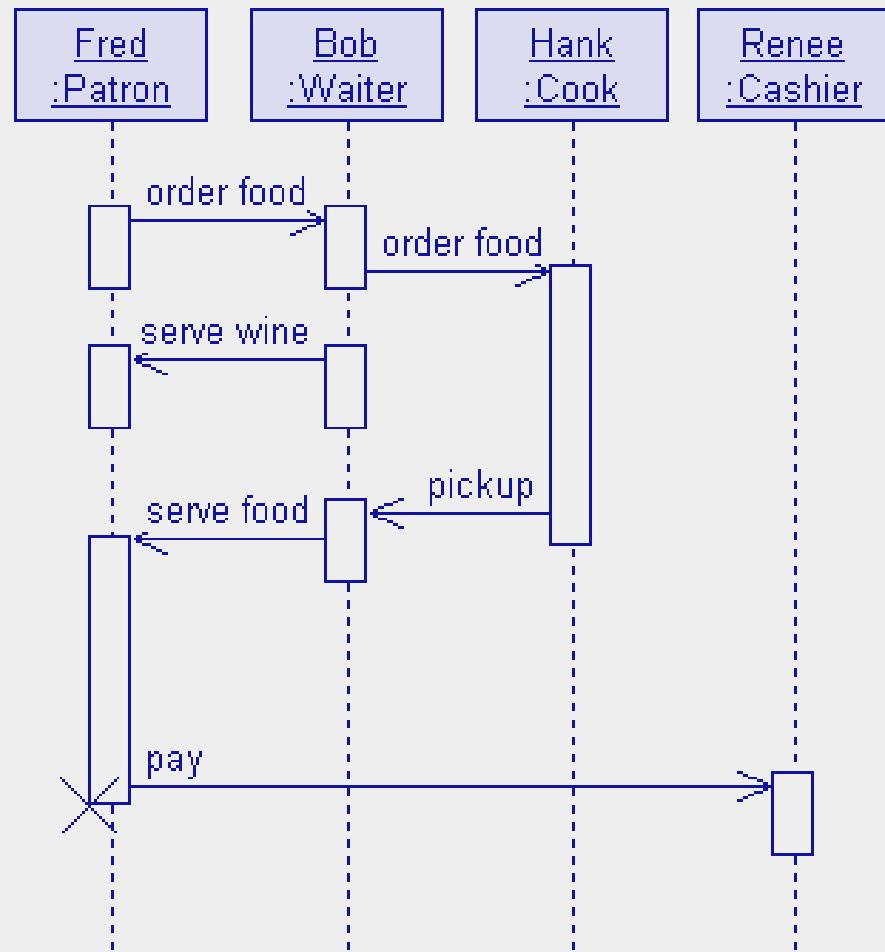
# Verbindungen in Klassendiagrammen – vereinfacht



## Objektdiagramm, vereinfacht

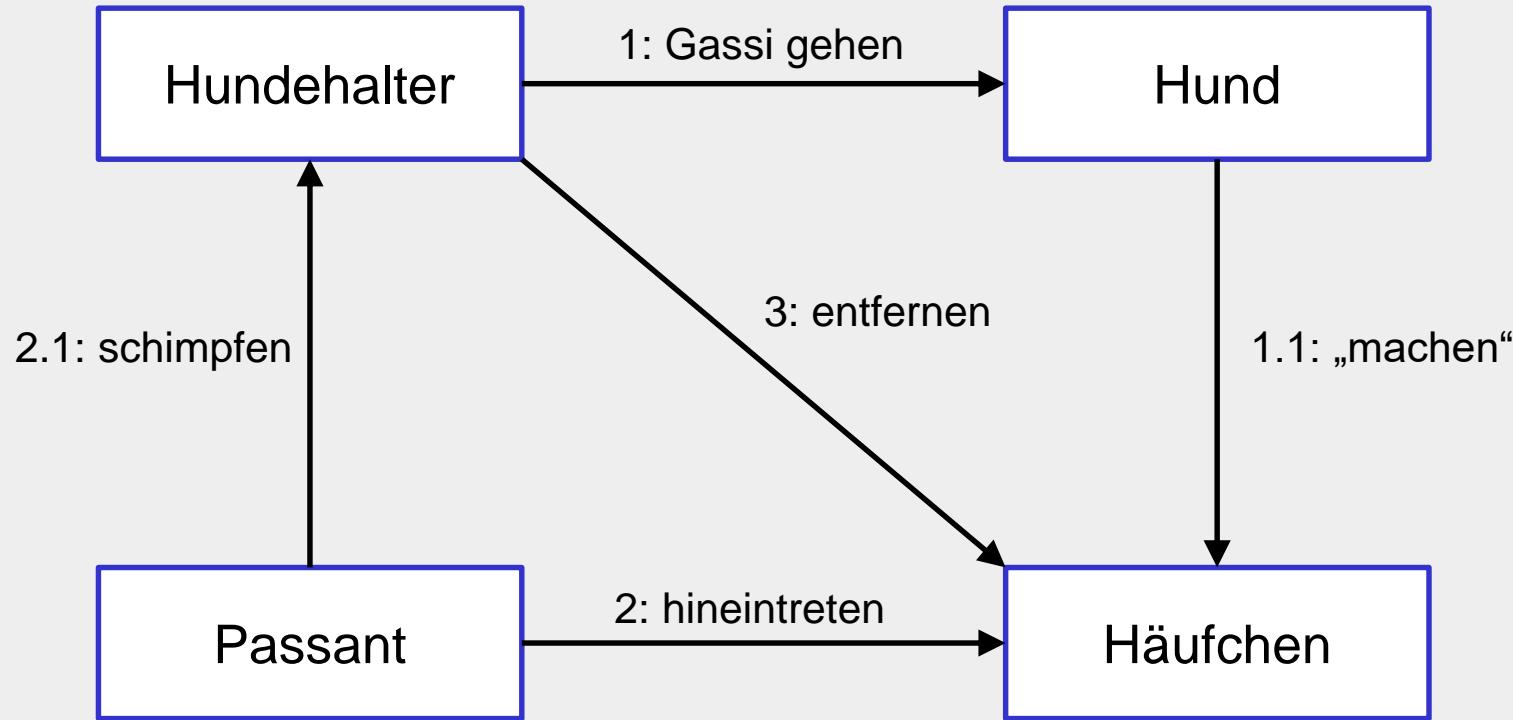


# Sequenzdiagramm, vereinfacht

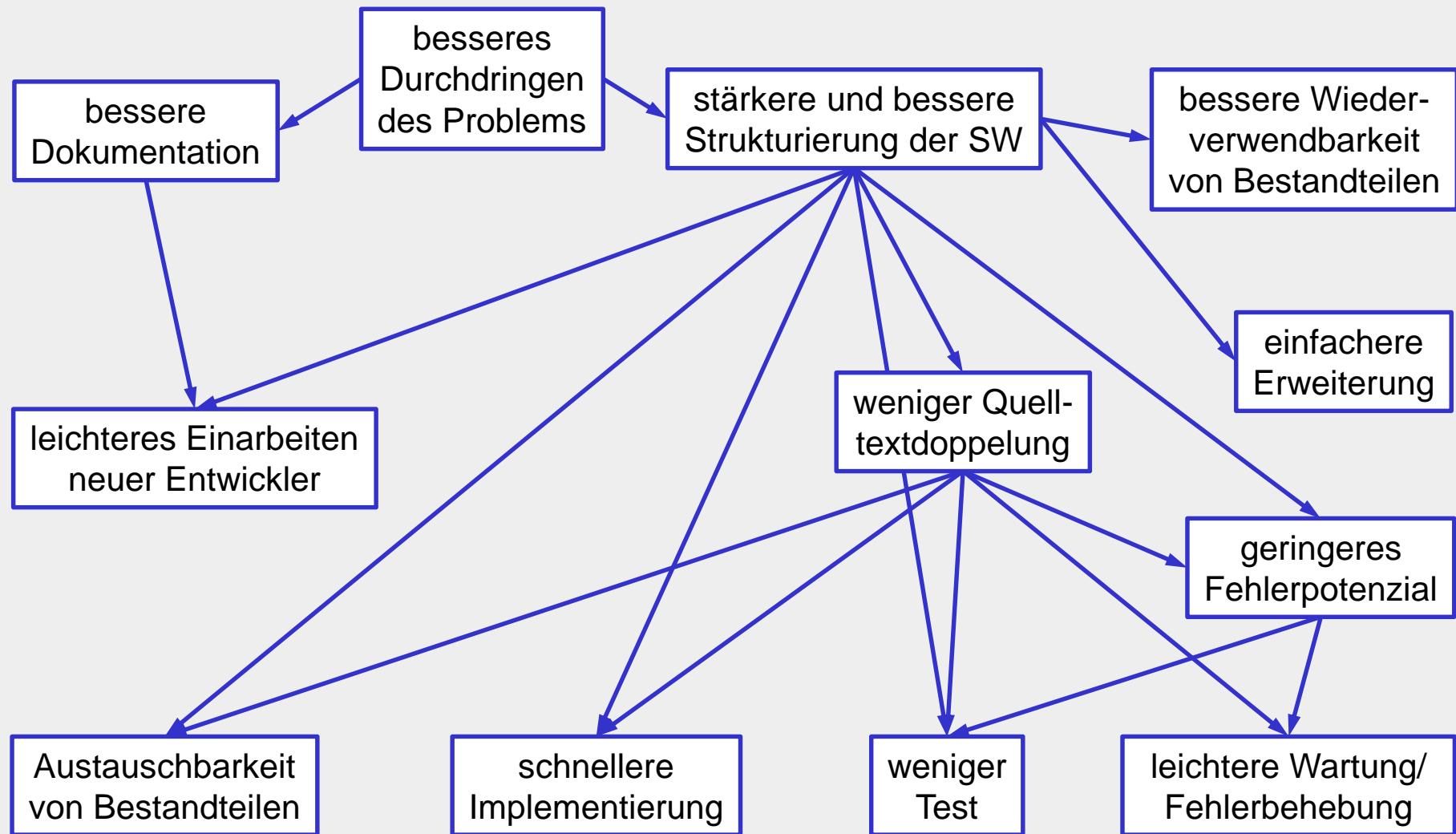


[Wikipedia]

# Kommunikationsdiagramm, vereinfacht – Beispiel [Jeckle]



## Einige Resultate der SW-Modellierung



## Wirtschaftlichkeit

- „langfristig mindestens kostendeckend“
- „kurzfristig mindestens fixkostendeckend“
- Maximieren der Wirtschaftlichkeit:  
Umsatz / Kosten → Max!
- Maximieren des Gewinns:  
Umsatz – Kosten → Max!
- Maximieren der Umsatzrendite:  
Gewinn / Umsatz → Max!
- höhere Verkaufspreise und/oder mehr verkaufte SW durch bessere Qualität und/oder höhere Leistung der SW
- geringere Kosten durch effiziente SW-Entwicklung

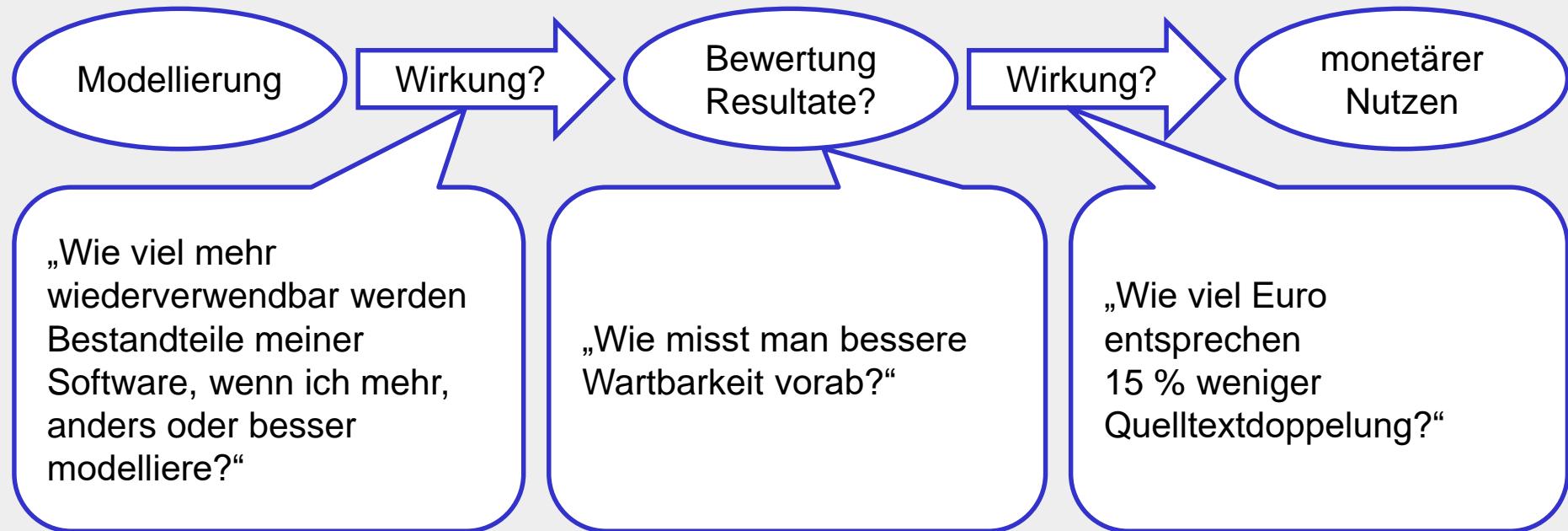


## Kosten der SW-Modellierung

- Kosten für SW-Modellierung leicht zu erfassen
- Kosten = Modellierungszeit \* Stundensatz für Entwickler + Fixkosten (z. B. für Modellierungswerkzeuge)
- Voraussetzung für das Erfassen der Kosten für die SW-Modellierung: Rückmeldungen der Zeiten für die Modellierung

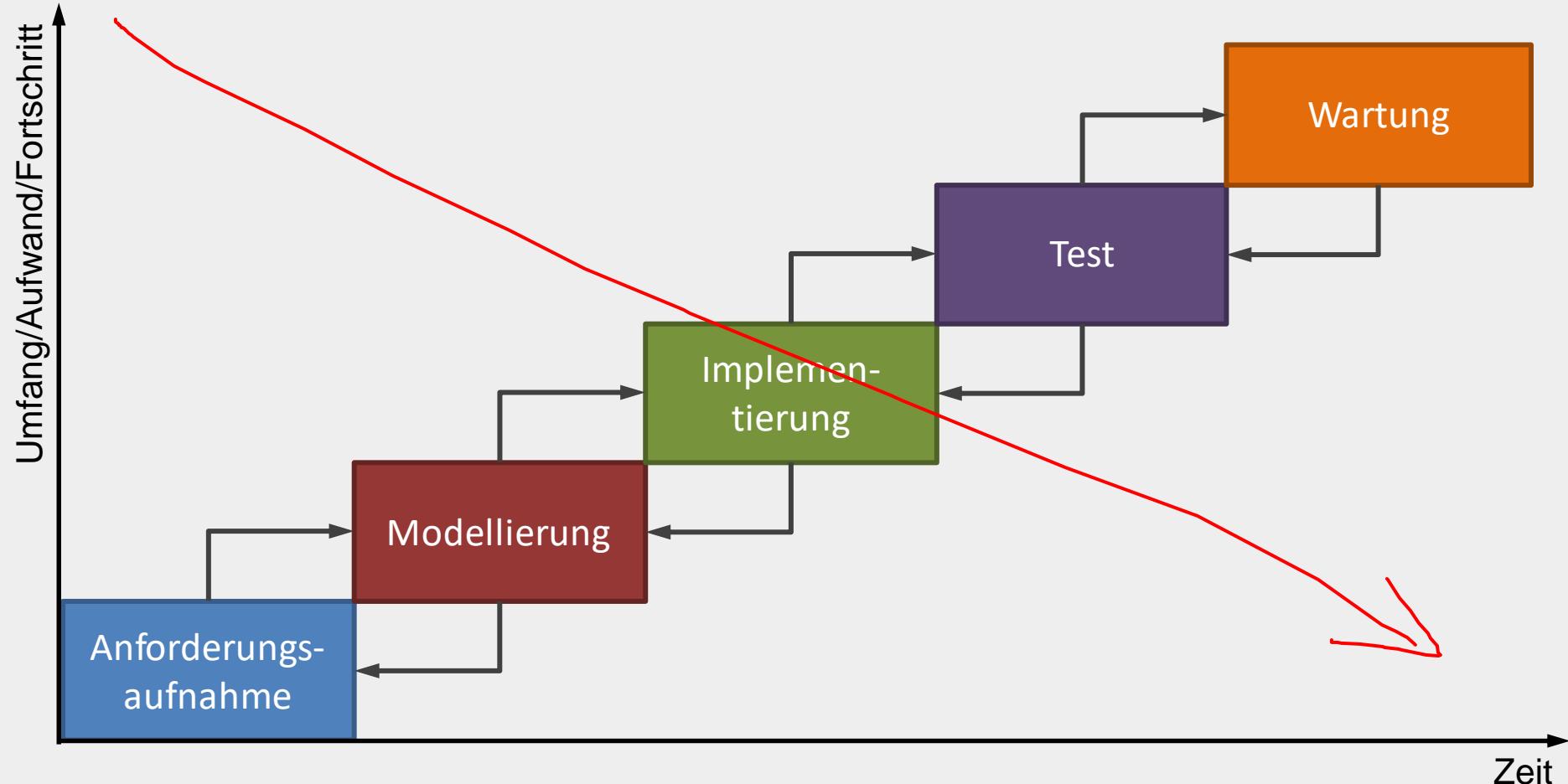


## Nutzen der SW-Modellierung?

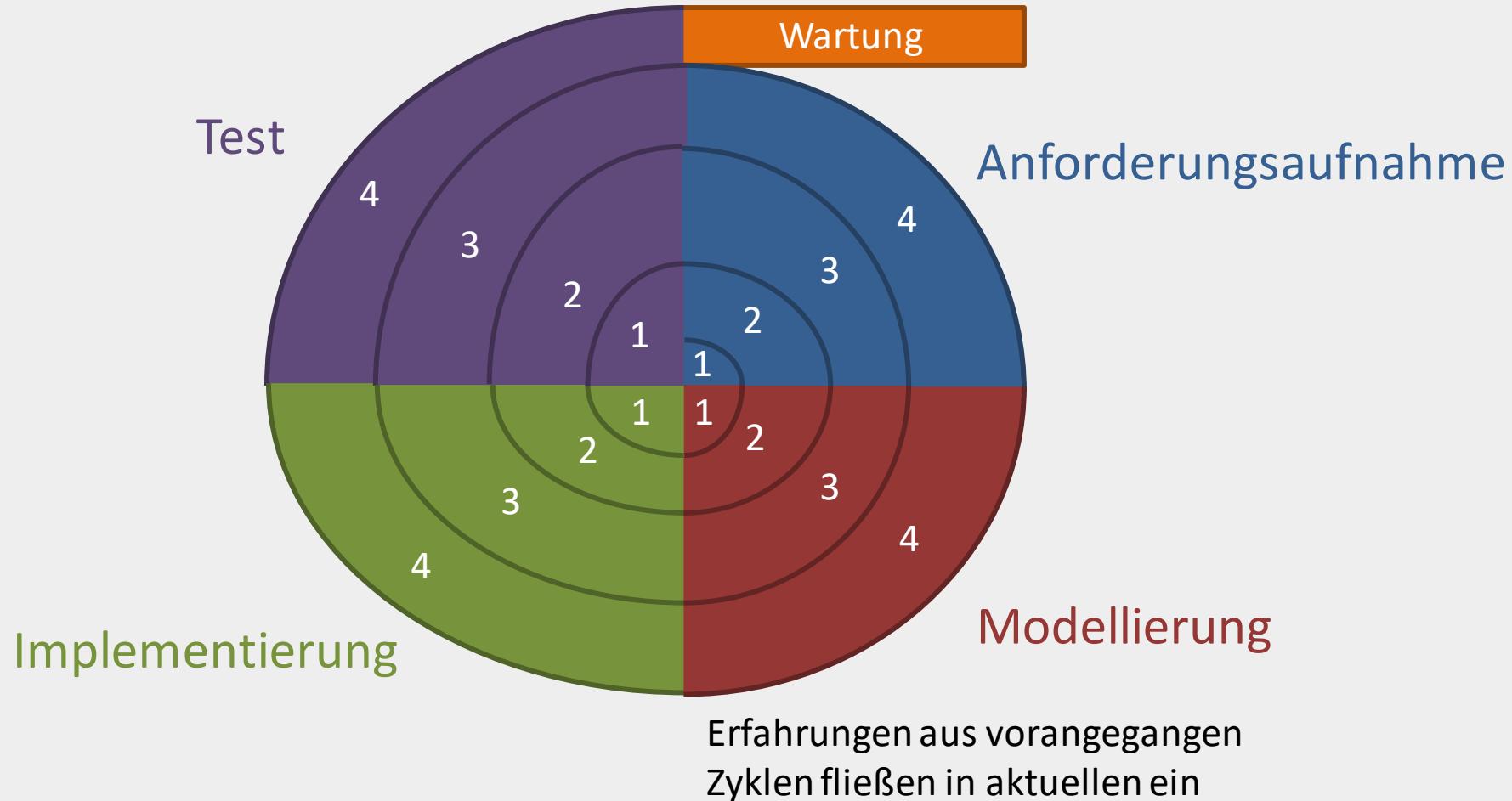


- Nutzen die SW-Modellierung schwer zu erfassen
- Art und Umfang der SW-Modellierung schwer betriebswirtschaftlich-analytisch zu bestimmen
- Art und Umfang der SW-Modellierung besser empirisch und besser **im Rahmen des übergeordneten Softwareentwicklungsprojektes** zu bestimmen

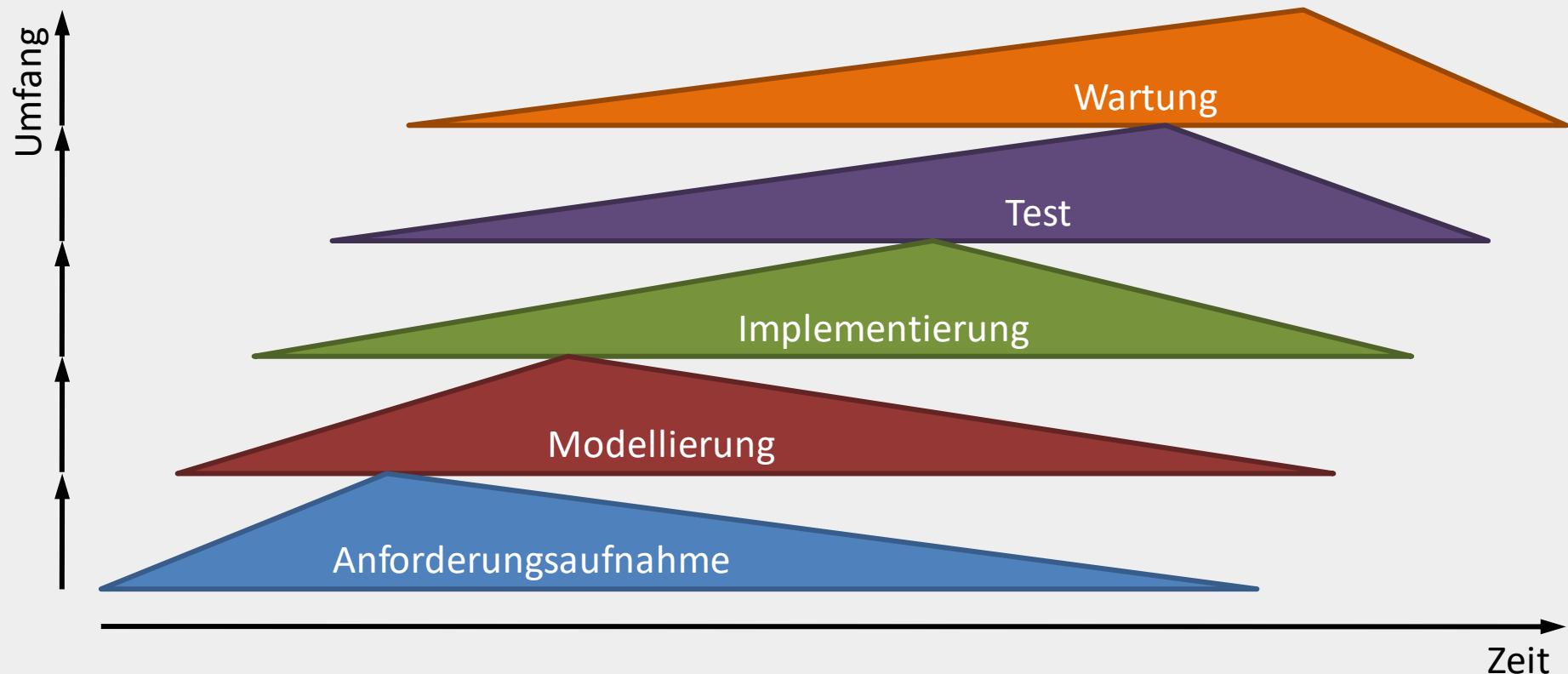
## „Wasserfallmodell“



## Spiralmodell, vereinfacht; evolutionäres Prototyping



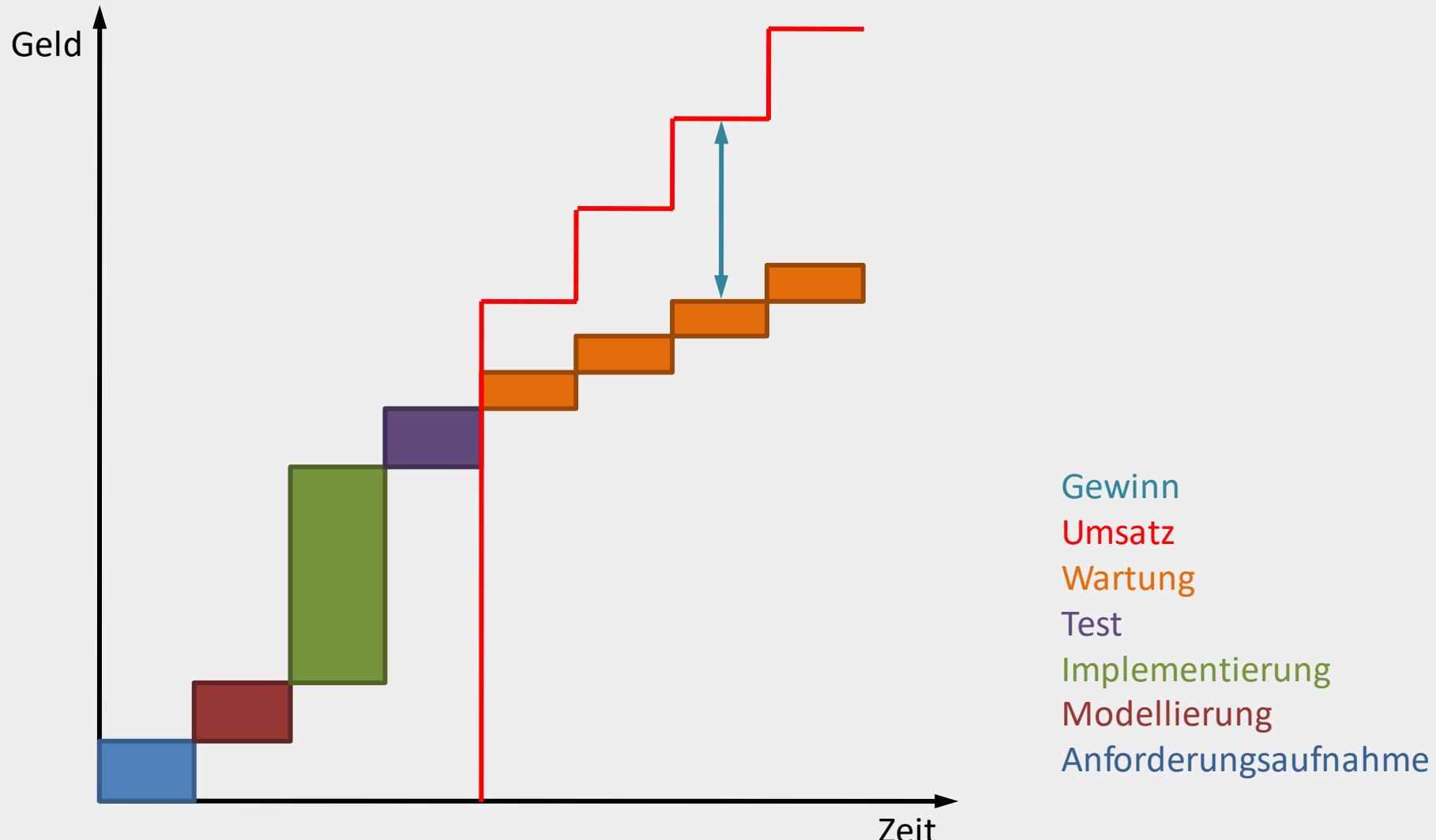
## Unified Process, vereinfacht



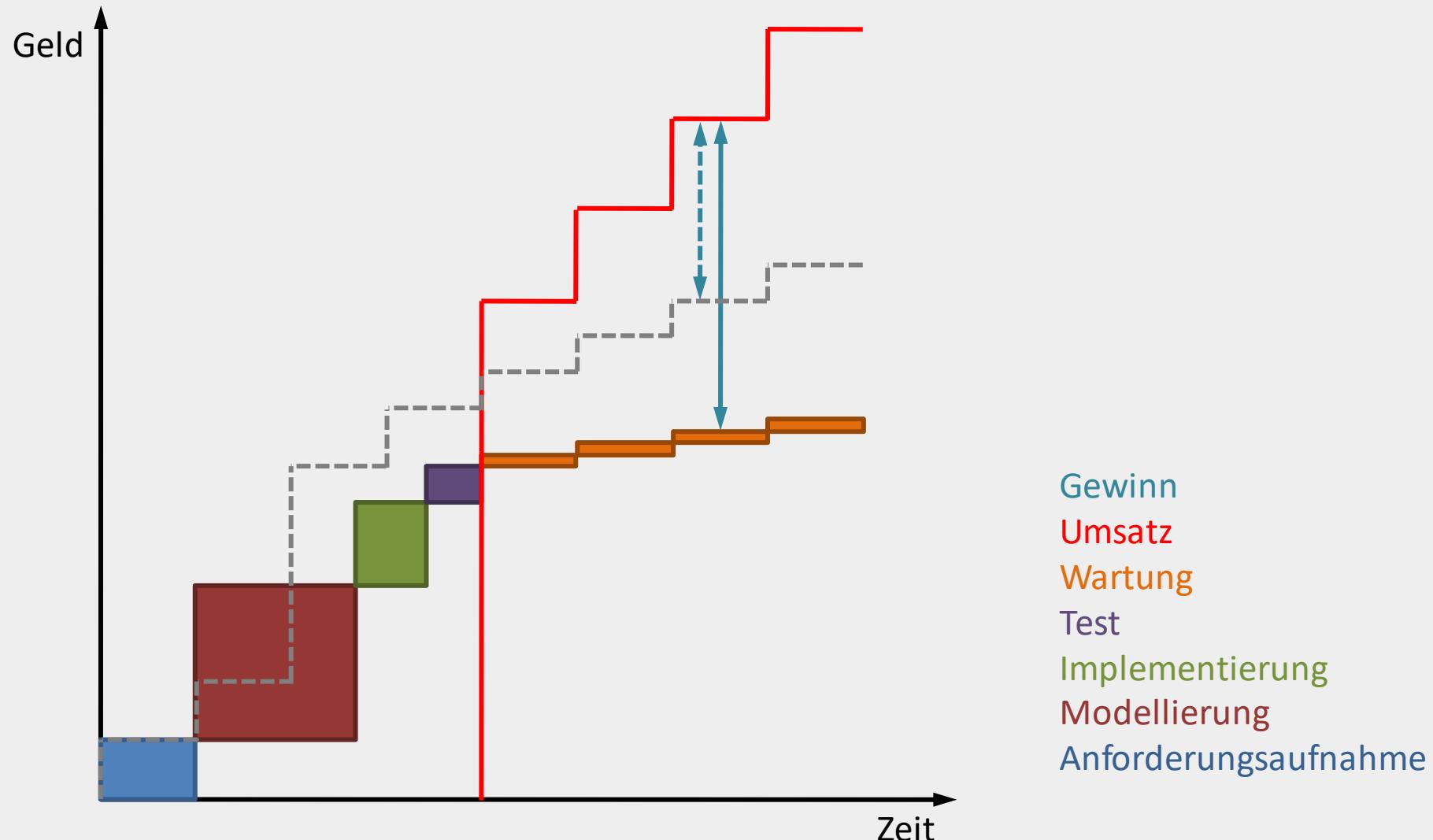
## Unified Process – Best Practices

- Iterative Entwicklung → Änderungen der Anforderungen auch später noch berücksichtigt
- Anforderungsmanagement (sich ändernde Anforderungen Grundlage der Software) → höhere Kundenzufriedenheit.
- Komponentenbasiert → Komponenten separat entwickelt und getestet → wiederverwendbar und austauschbar
- Visuelle Modellierung in UML → besseres Verständnis der Domäne und des Problems
- Qualitätsmanagement → frühzeitiges Erkennung von Fehlern
- Änderungsmanagement → Altstände reproduzierbar

## Wirtschaftlichkeit über der Zeit (I)



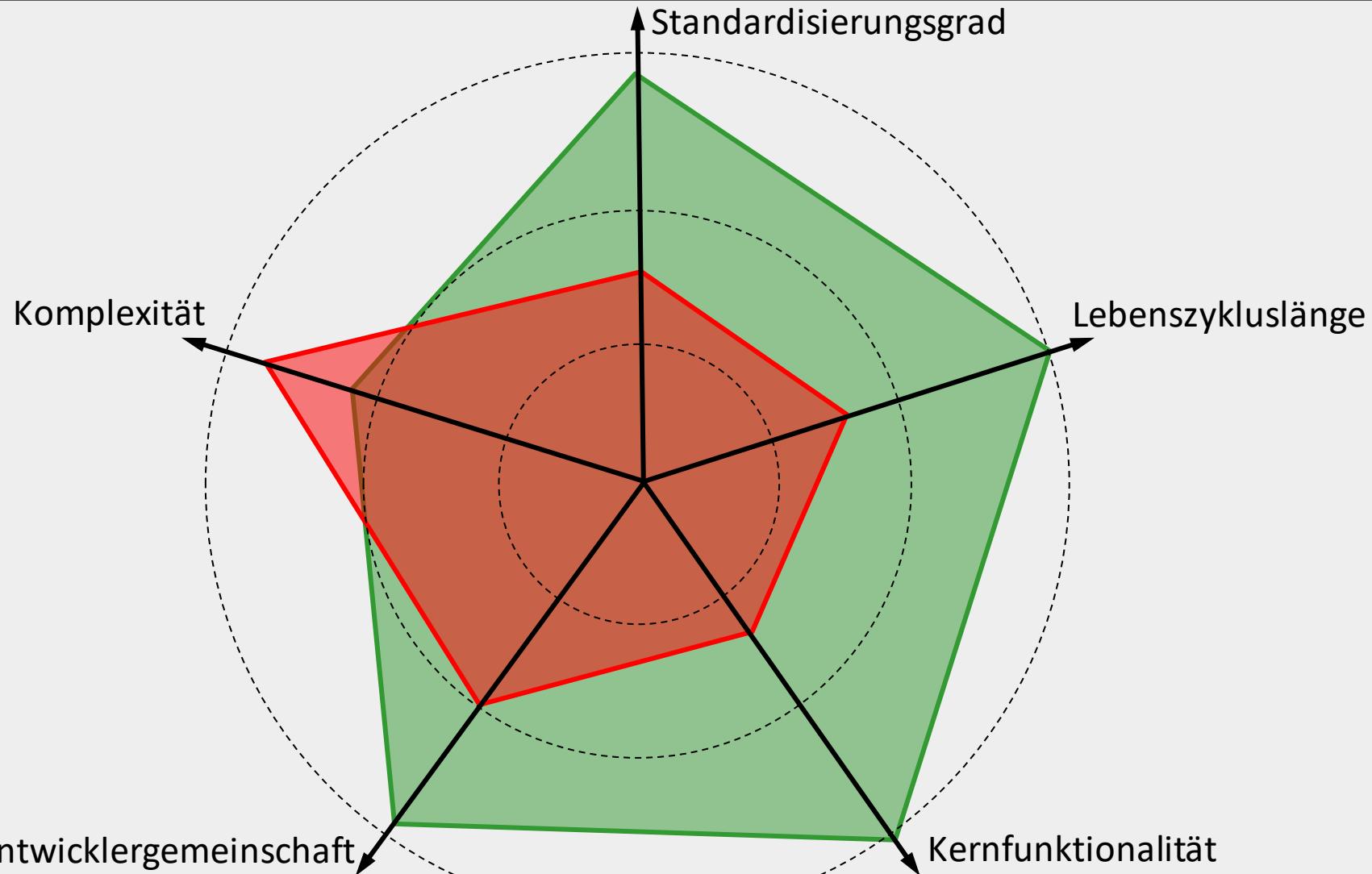
## Wirtschaftlichkeit über der Zeit (II)



## Controlling der Softwareentwicklung

- Planen der Entwicklung en detail (detaillierter Projektplan, Vorkalkulation)
- nach Projektstart zyklisch:
  - Rückmeldungen durch die am Projekt Beteiligten
  - Vergleich von Soll und Ist
  - Anpassen von Ist an Soll (oder umgekehrt) durch geeignete Maßnahmen
  - Verfeinern des Projektplanes
- Nachkalkulation nach Projektende
- Erfassen von Erfahrungswerten für Folgeprojekte
- → Art und Umfang der SW-Modellierung empirisch im Rahmen des übergeordneten SW-Entwicklungsprojektes bestimmt

## SW-Dimensionen → Modellierungsumfang



## Standardisierungsgrad, Extrema

- Standardsoftware
  - viele Kunden
  - verschiedene Versionen
  - eventuell Auswahl von Modulen
  - Konfiguration („Customizing“), aber keine/kaum Anpassungsprogrammierung
- Individualsoftware
  - ausschließlich ein Kunde
  - nur eine „aktuelle“ Version
  - meist nicht modular
  - quasi ausschließlich Anpassungsprogrammierung

Modellierungsumfang

## „Nähe zum Kern der Software“, Extrema

- Kernfunktionalität
  - anwenderfern
  - entscheidend für die Verfügbarkeit der Software
  - von allen anderen Bestandteilen genutzt
- „Randfunktionalität“
  - anwendernah
  - Ausfall führt nicht zu Totalausfall der Software
  - nutzt andere Bestandteile der Software, wird aber nicht/kaum von anderen benutzt

Modellierungsumfang

## Art des Softwarelebenzyklus, Extrema

- „Eintagsfliege“
  - einmaliges Softwareentwicklungsprojekt
  - keine Pflege und Weiterentwicklung
  - keine Wartung und kein Support
- „Dauerläufer bzw. –brenner“
  - Initialprojekt und Folgeprojekte
  - Pflege und Weiterentwicklung vorgesehen
  - Wartung und Support vorgesehen



Modellierungsumfang

## Typ der Entwicklergemeinschaft, Extrema

- ein und somit homogener Entwickler
  - Vermitteln des Modells nicht nötig
  - kein standardisiertes Modell nötig
- Viele, heterogene Entwickler
  - extrem verschiedene Qualifikationen
  - Entwicklung auch durch Fremdfirmen
  - Entwicklung auch durch Kunden selbst
  - Entwicklung über Zeitzonengrenzen hinweg
  - Vermitteln eines standardisierten Modells nötig



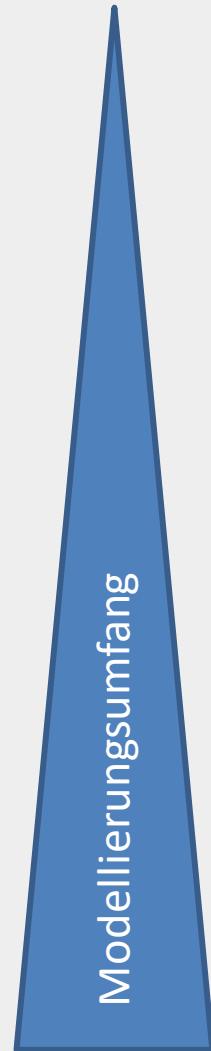
Modellierungsumfang

## Softwarekomplexität

- horizontale Komplexität
  - Anzahl der nebeneinander existierenden Bestandteile
  - Intensität der Kommunikation zwischen den Bestandteilen
  - Grad der Abdeckung des Anwendungsbereiches
- vertikale Komplexität
  - Anzahl und „Dicke“ der Schichten zwischen den vertikalen Systemgrenzen (z. B. GUI und Datenbank)
  - Intensität der Kommunikation zwischen den Schichten
- funktionale Komplexität
  - verschiedene Maße: LoC, Halstead, McCabe, ...

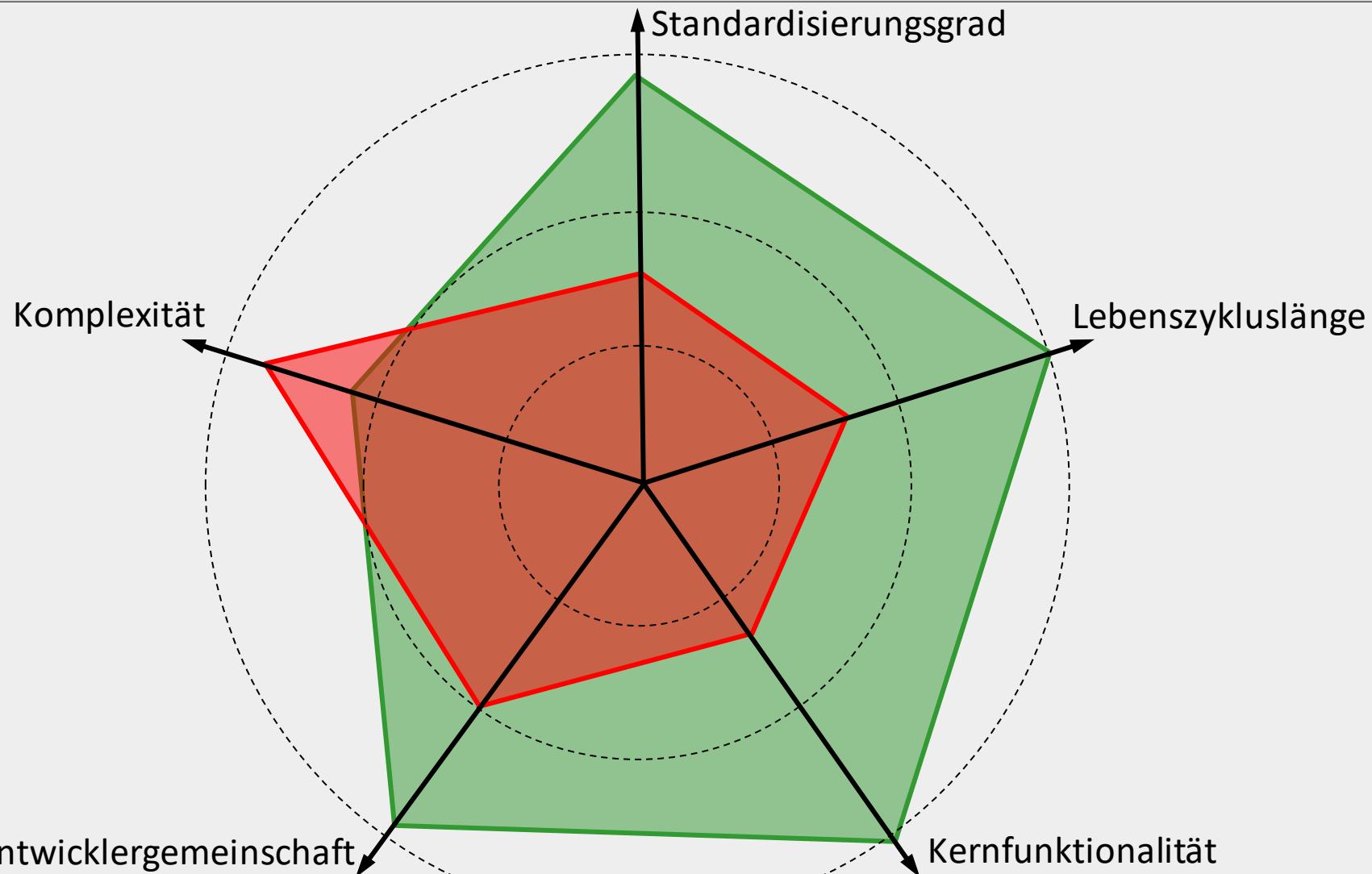
## Softwarekomplexität, Extrema

- einfach
  - eine Schicht oder zwei Schichten
  - stark eingeschränkter Anwendungsbereich
  - stark beschränkte Kommunikation
  - wenig und einfache Funktionalität
- komplex
  - drei und mehr Schichten
  - vollständig abgedeckter Anwendungsbereich
  - intensive Kommunikation
  - viel und komplizierte Funktionalität



Modellierungsumfang

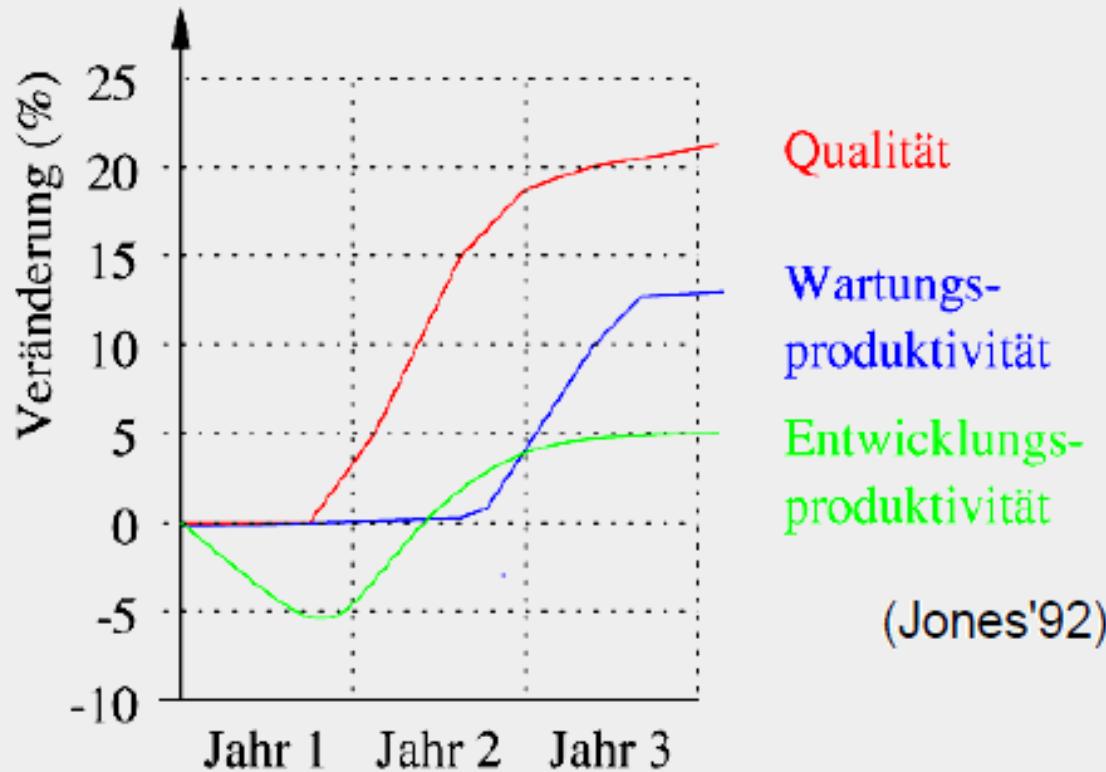
## SW-Dimensionen → Modellierungsumfang



## Empfehlungen für die Modellierung

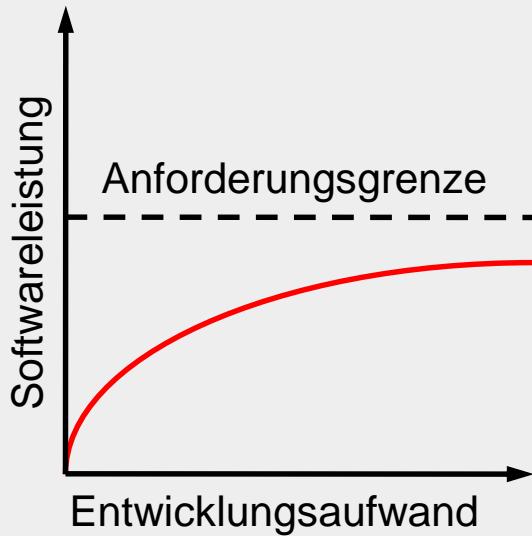
- Projektplan und Projektcontrolling
- **Nutzen von Modellierungswerkzeugen (CASE)**
- Reverse-Engineering
- Schulung der Entwickler in standardisierter Modellierung und Anwendung standardisierter Modellierungswerkzeuge und Modellierungssprachen
- Nutzen von Referenzmodellen
- Nutzen von Entwurfsmustern (GoF)
- **Tendenziell zu wenig Modellierung → lieber mehr modellieren und weniger programmieren**

## Einsatz von CASE-Werkzeugen

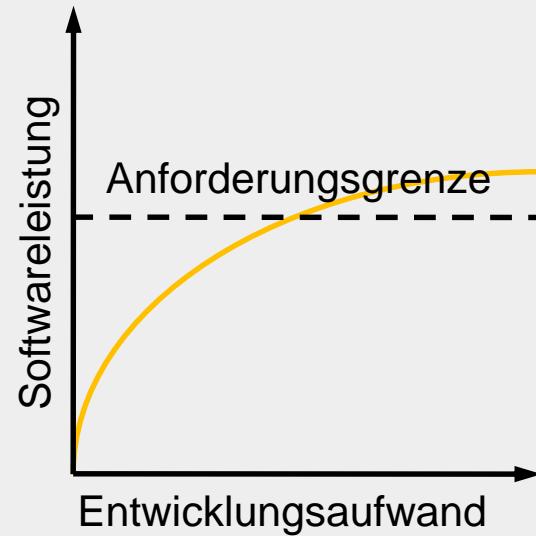


## Modellierungsumfang und Softwareleistung

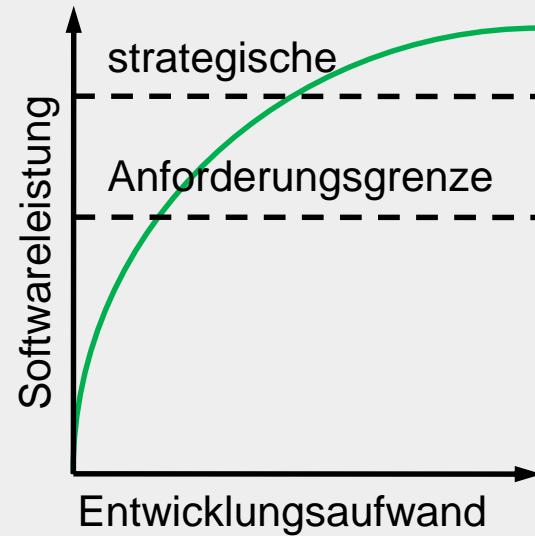
zu wenig  
Modellierung



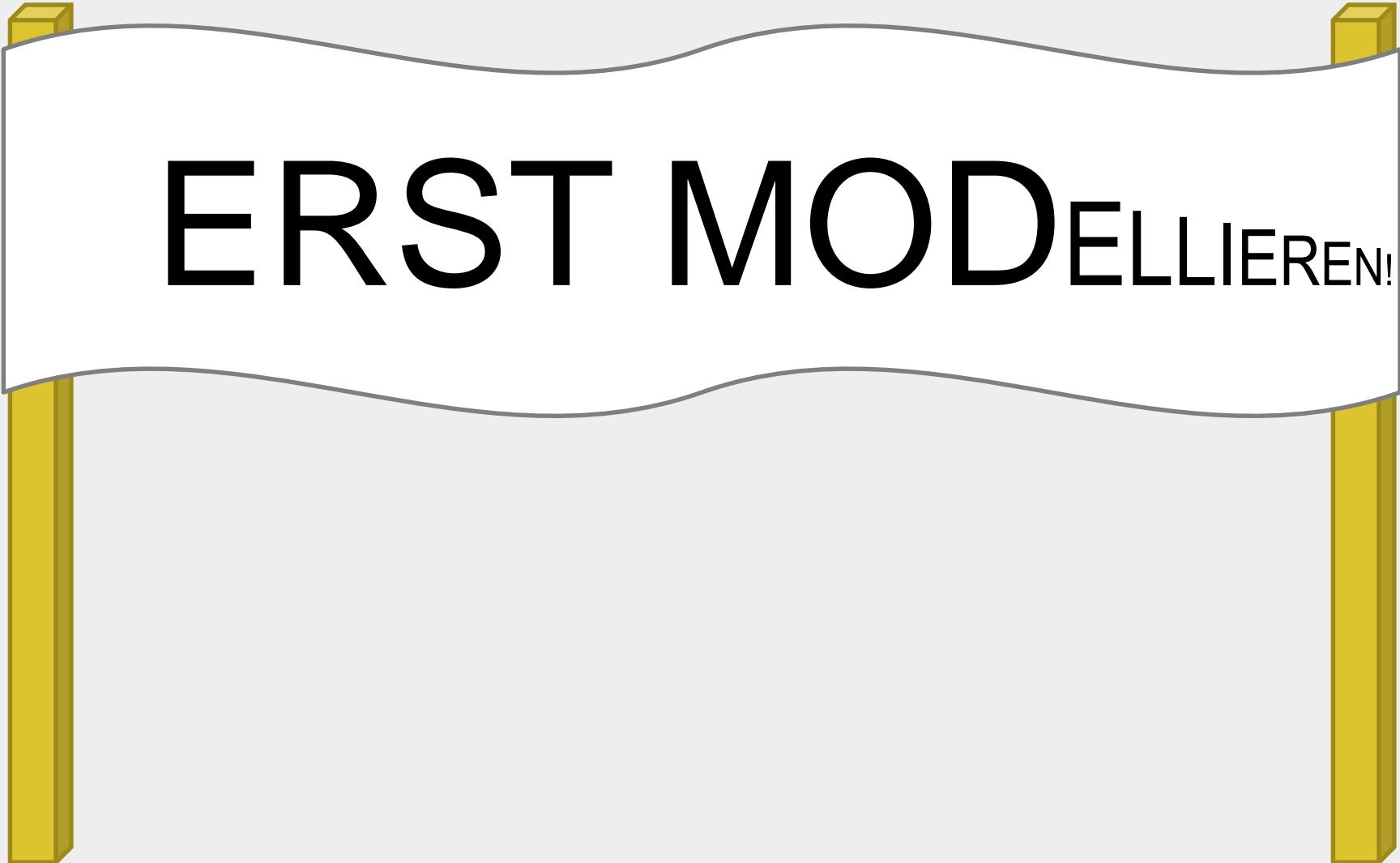
ausreichend  
Modellierung



zu viel  
Modellierung?



## Guter Rat



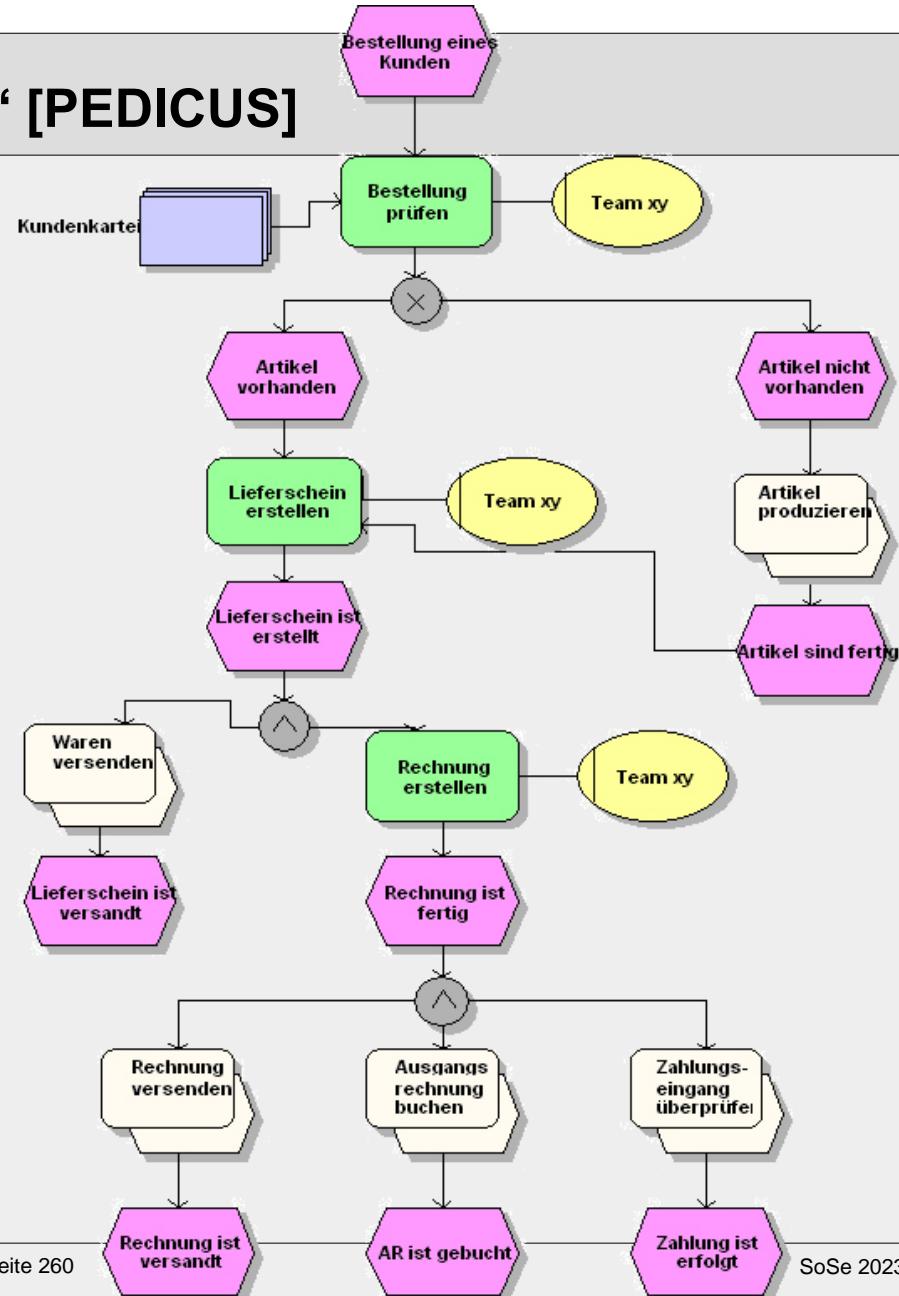
**ERST MODELLIEREN!**

## Ereignisgesteuerte Prozesskette (EPK) – vereinfacht

- grafische Sprache zur **Modellierung von Geschäftsprozessen** in und zwischen Organisationen
- „Ereignisse lösen Funktionen aus und sind deren Ergebnis.“ [Scheer]
- **Ereignisse** sind Zustandsänderungen des Systems und finden (idealisiert!) zu **Zeitpunkten** statt.
- **Funktionen** führen dazu, dass Ereignisse auftreten (sich Zustände des Systems ändern) und weisen **Dauern** auf.
- Logische **Operatoren** verknüpfen (eingehende oder ausgehende) Ereignisse (und Funktionen) und fallen bei 1:1-Verknüpfungen von Ereignissen und Funktionen weg.

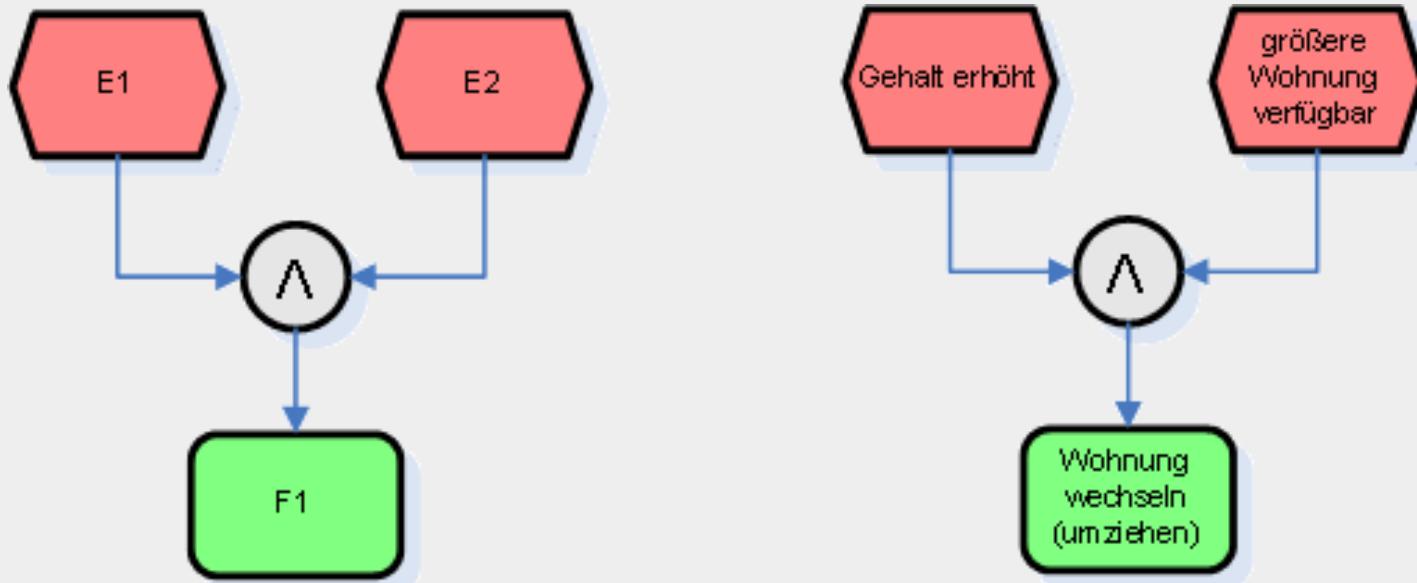


# **EPK – „Auftragsabwicklung“ [PEDICUS]**



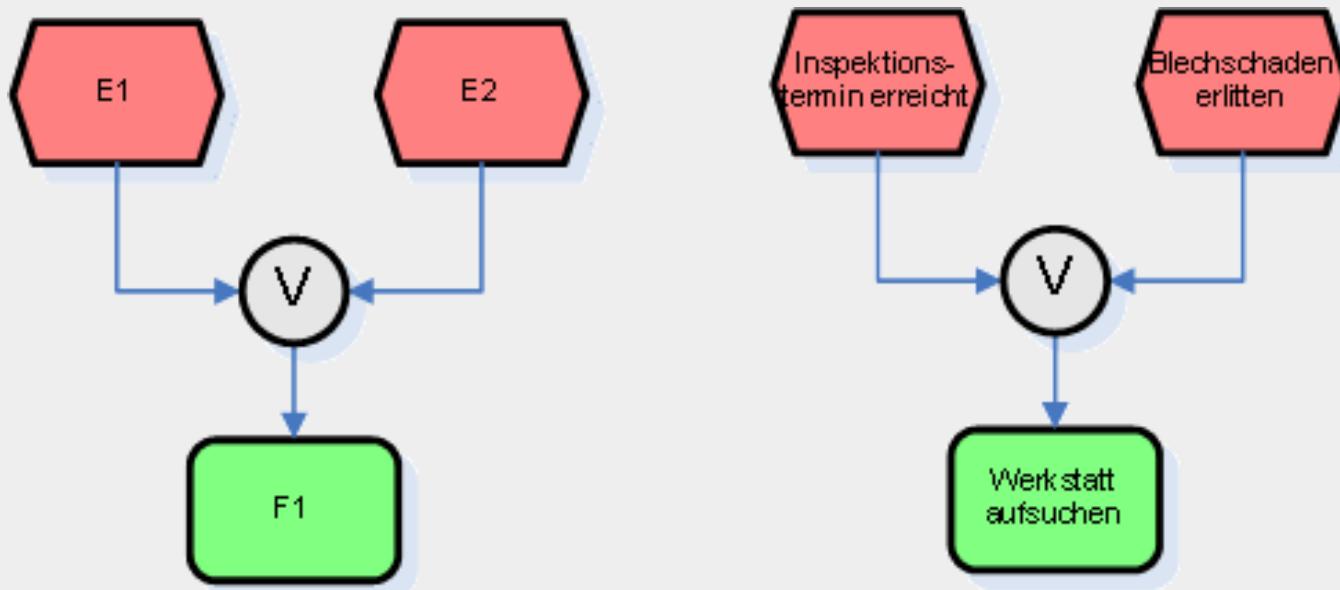
## UND-Verknüpfung ...

- ... zweier eingehender Ereignisse zu einer Funktion



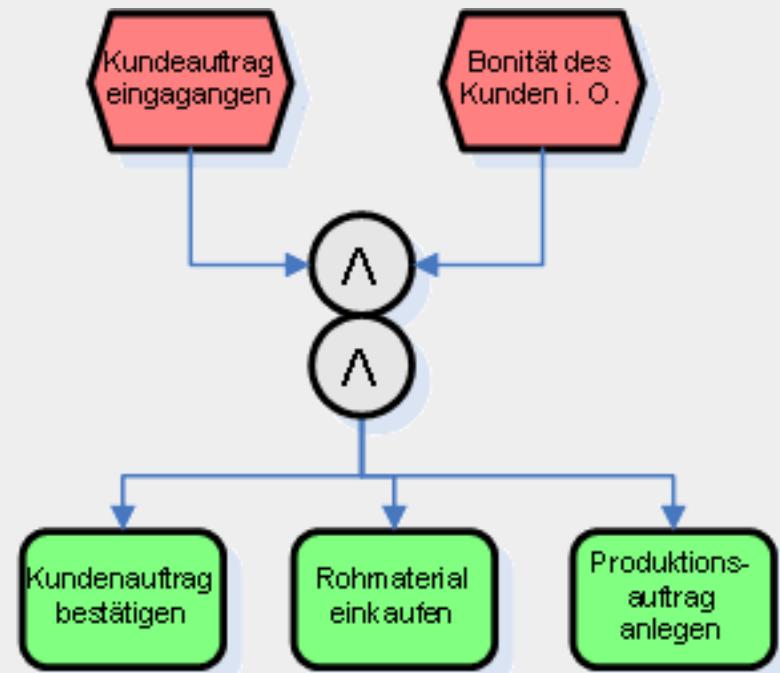
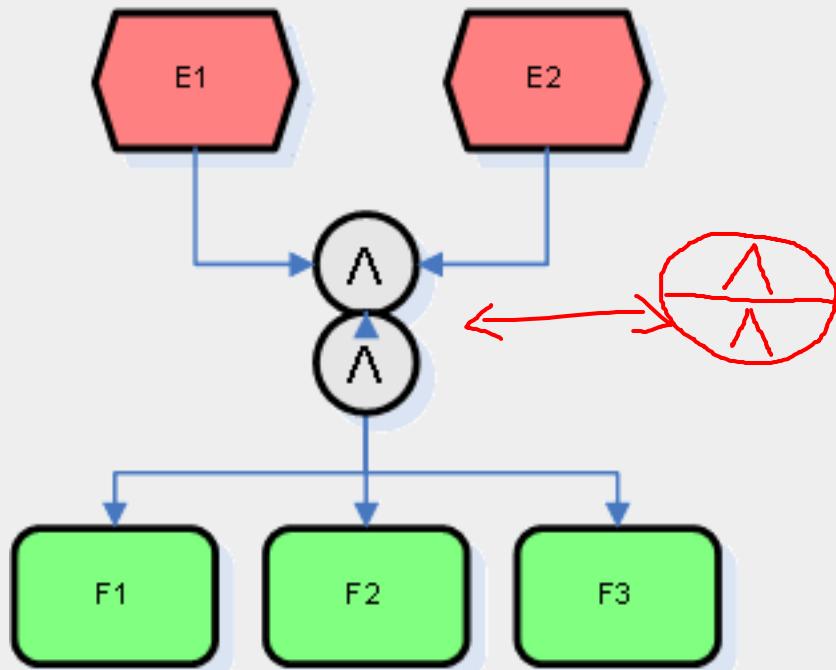
## ODER-Verknüpfung ...

- ... zweier eingehender Ereignisse zu einer Funktion



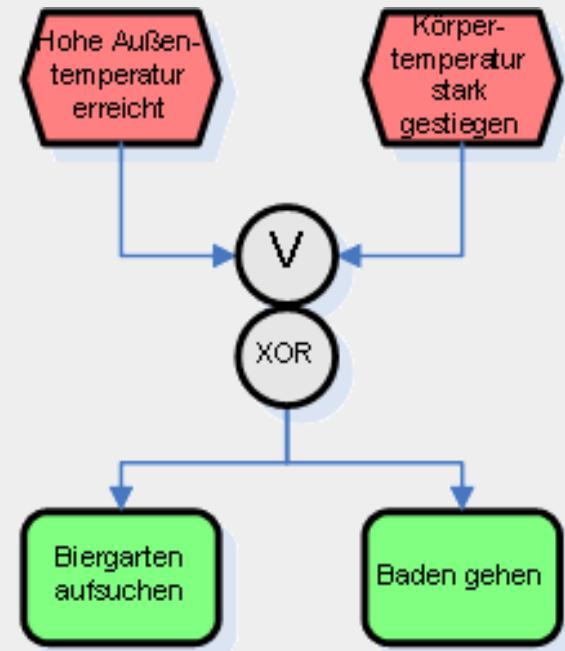
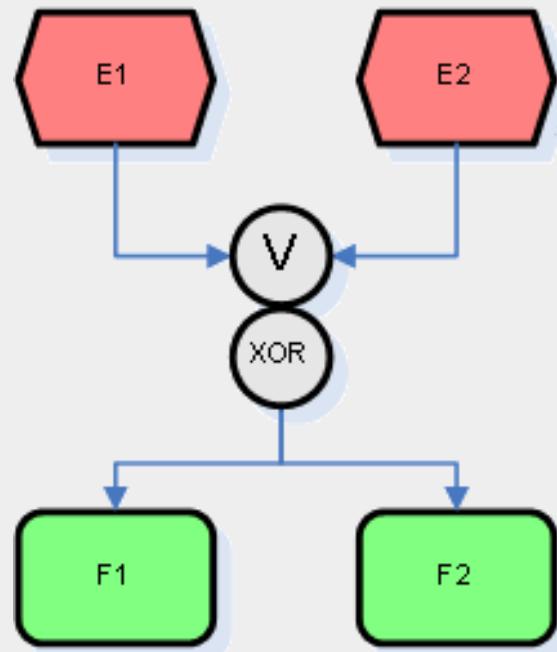
## UND-Verknüpfung ...

- ... zweier eingehender Ereignisse zu **drei** Funktionen



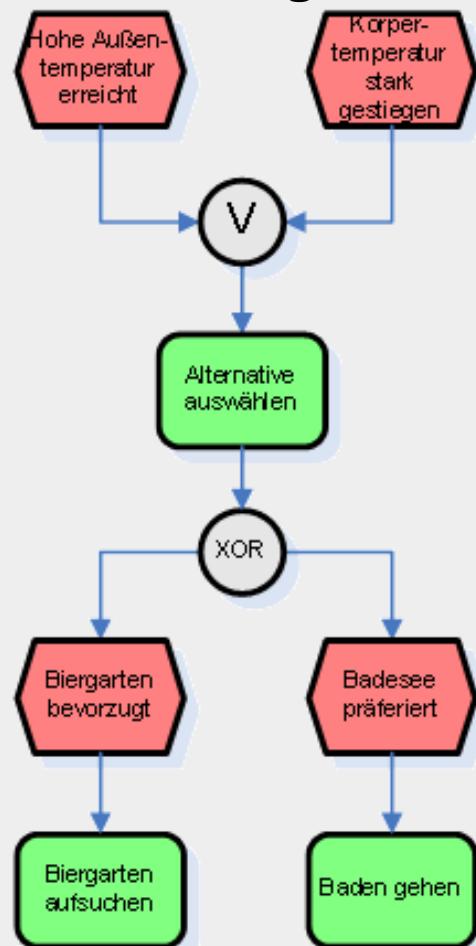
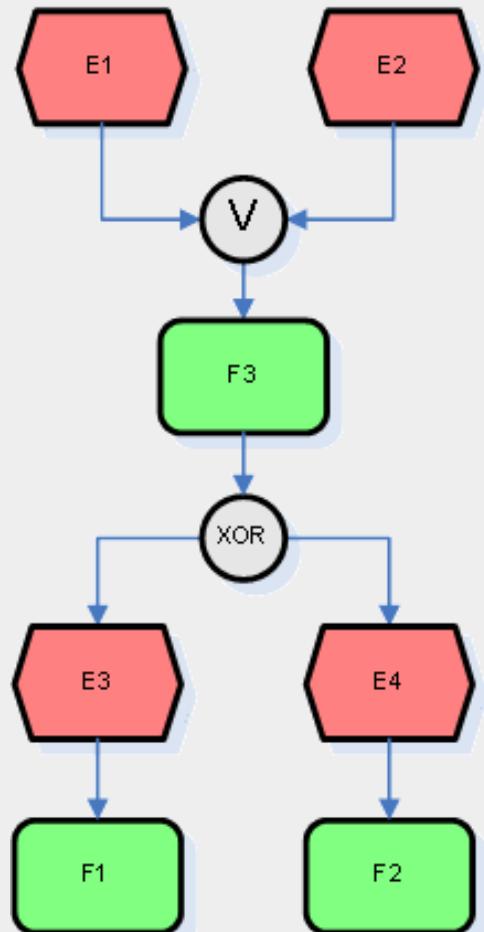
## ODER/XODER-Verknüpfung ...

- ... zweier Ereignisse zu zwei Funktionen

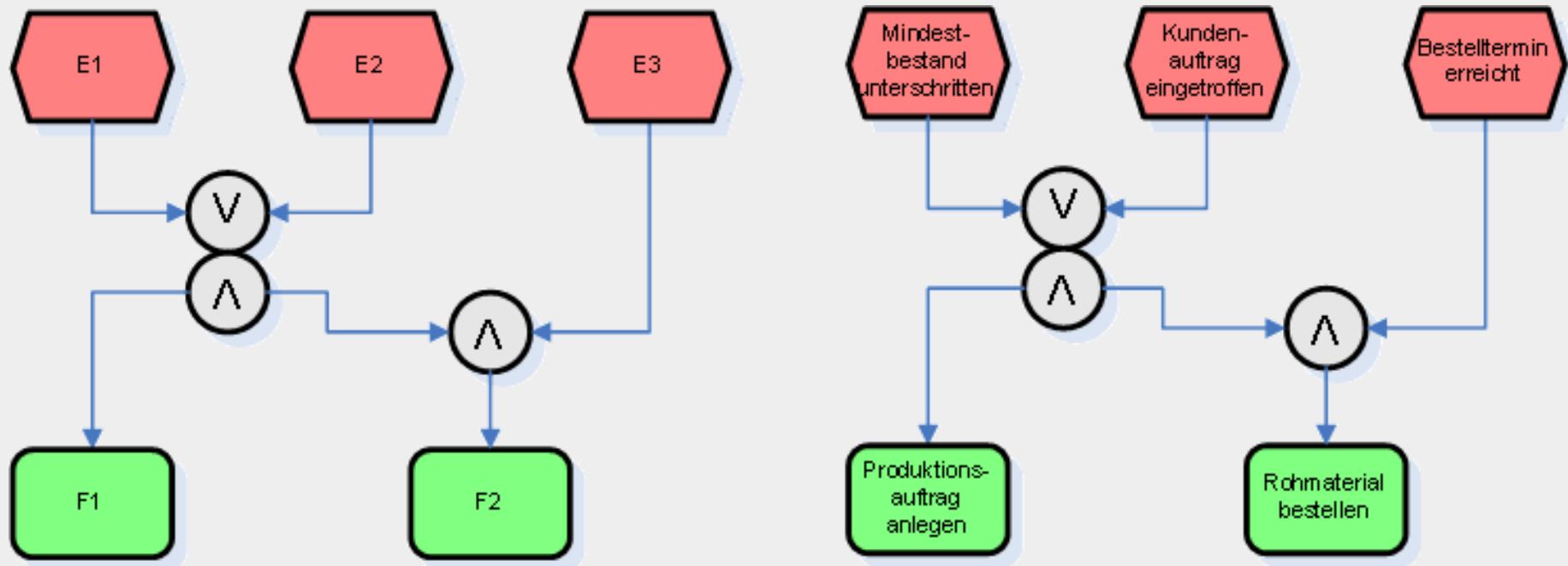


## ODER/XODER-Verknüpfung ...

- ... zweier Ereignisse zu zwei Funktionen – **wiederaufgenommen**

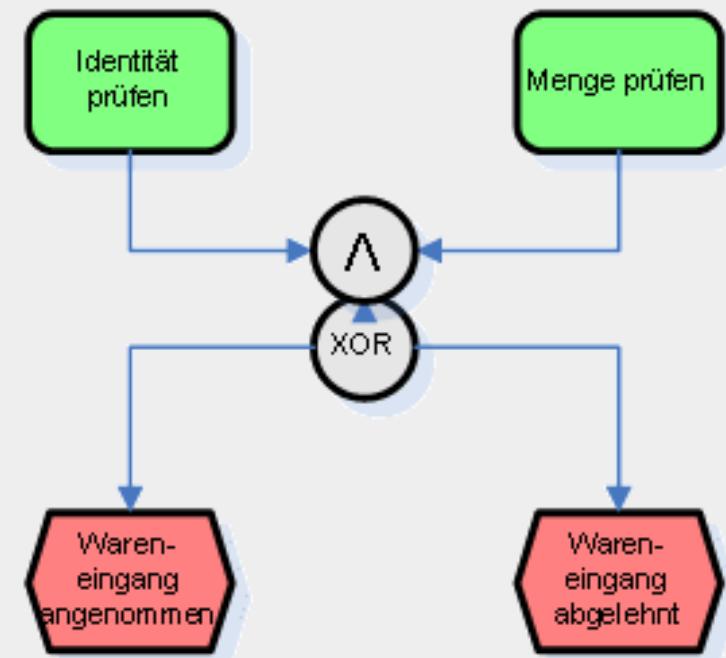
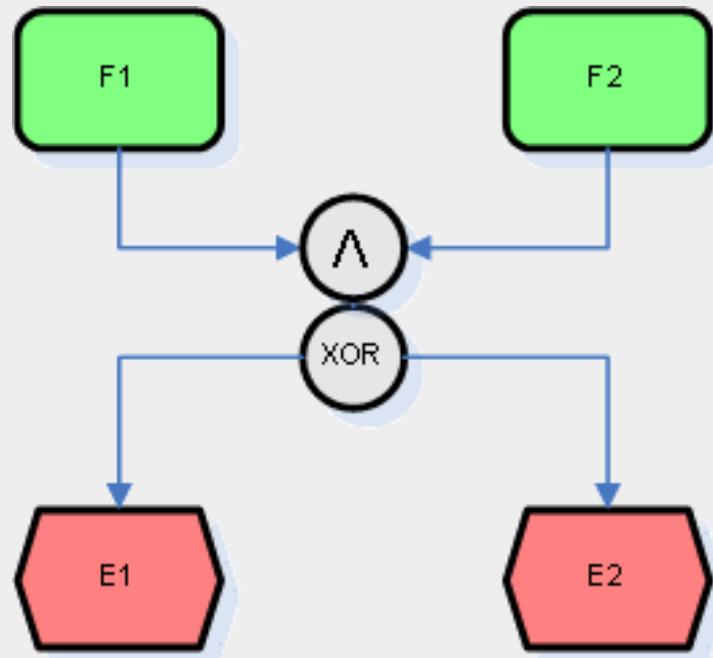


# Hierarchische Verknüpfung von Operatoren

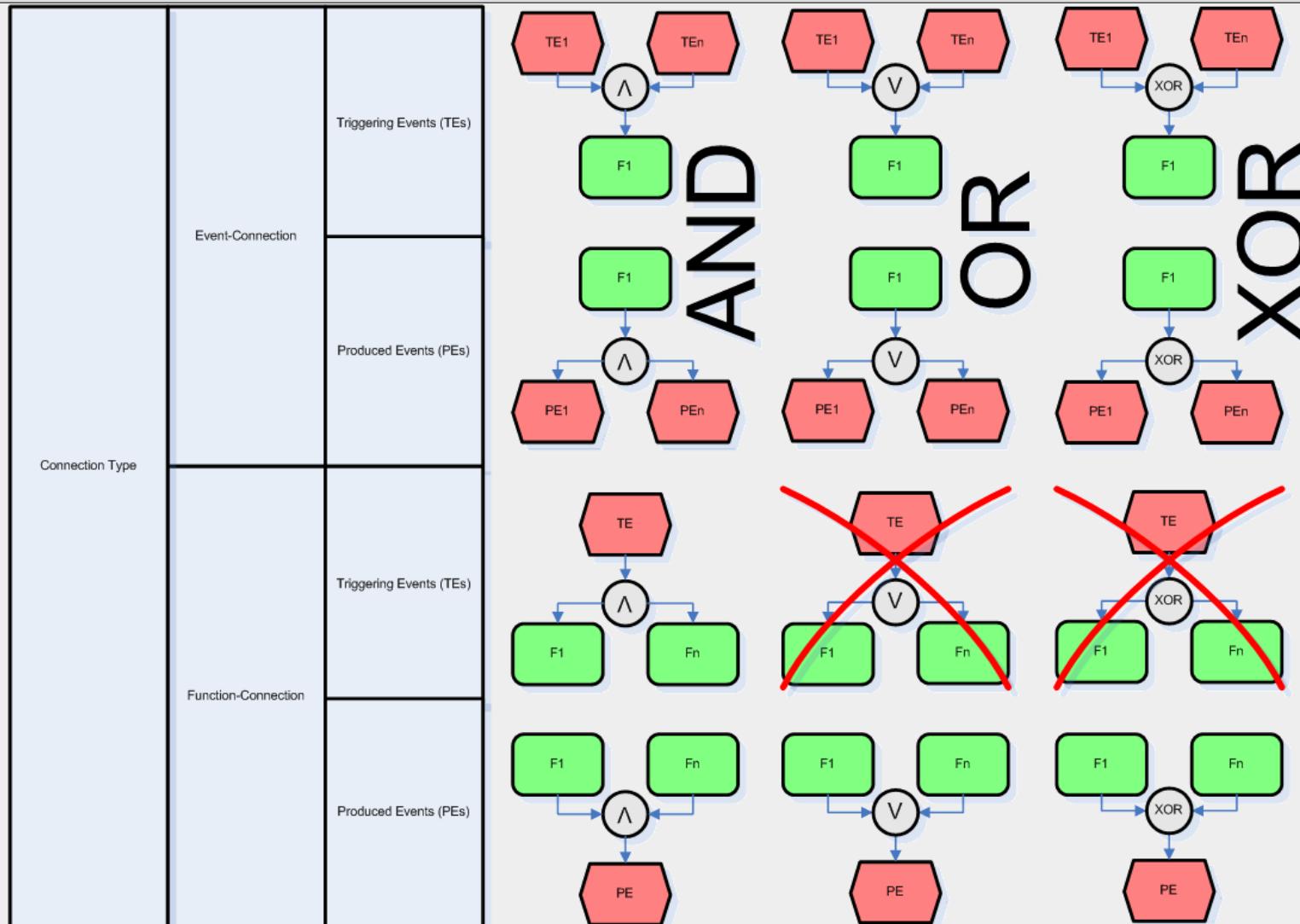


## UND/XODER-Verknüpfung ...

- ... zweier **Funktionen** zu zwei Ereignissen

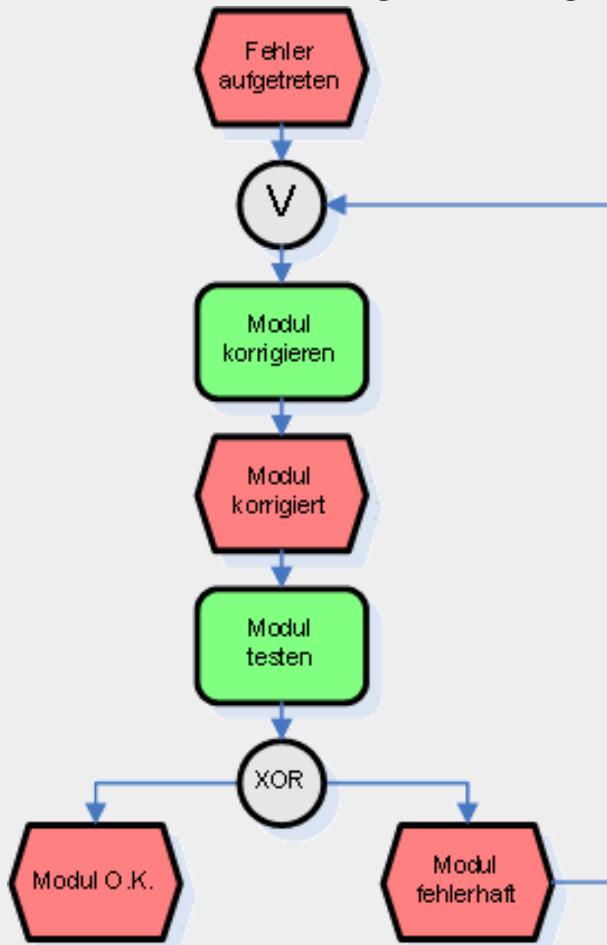


# Regeln zur Funktions- und Ereignisverknüpfung



## Rücksprünge in EPKn ...

- ... nur dann, wenn alle anderen Regeln eingehalten werden



## Namen für Ereignisse und Funktionen

- Funktion:
  - drückt eine **Tätigkeit** aus, die mit einem **Objekt** ausgeführt wird
  - Substantiv + Infinitiv (eines Verbs)
  - Beispiele: „Auftrag annehmen“, „Wartung durchführen“, „Bestellung versenden“, „Ware prüfen“, ...
- Ereignis:
  - drückt einen **Zustand** aus, den ein **Objekt** (gerade) erreicht hat
  - Substantiv + Partizip (II)
  - Beispiele: „Bestellung eingetroffen“, „Auftrag angenommen“, „Wartung durchgeführt“, „Bestellung versendet“, „Prüfung i. O.“, „Bonität O.K.“,  
...

## EPK – Beispielaufgabe „Auftragsabwicklung“

- Folgenden Prozess beschreibt Ihnen der COO einer mittelständischen AG, auf dass Sie ihn als Istprozess im Rahmen eines Business-Process-Reengineering aufnehmen:
  - Wenn ein Auftrag eingeht, wird die Kreditwürdigkeit des Kunden geprüft. Ist die Kreditwürdigkeit nicht gegeben, wird der Kundenauftrag abgelehnt. Ist die Kreditwürdigkeit gegeben, wird dreierlei veranlasst:
    - Erstens wird ein Liefertermin prognostiziert und danach eine Auftragsbestätigung an den Kunden gesendet,
    - zweitens wird Rohmaterial bestellt, und
    - drittens wird ein Produktionsauftrag angelegt.
  - Ist der Produktionsauftrag angelegt, der Starttermin für den Produktionsauftrag erreicht worden und das Rohmaterial für das Produkt eingegangen, wird das Produkt produziert.
  - Ist nach der Produktion das Produkt verfügbar, und ist der Liefertermin für das Produkt erreicht worden, werden nacheinander das Produkt an den Kunden geliefert und die Rechnung an den Kunden gestellt.

## Gliederung

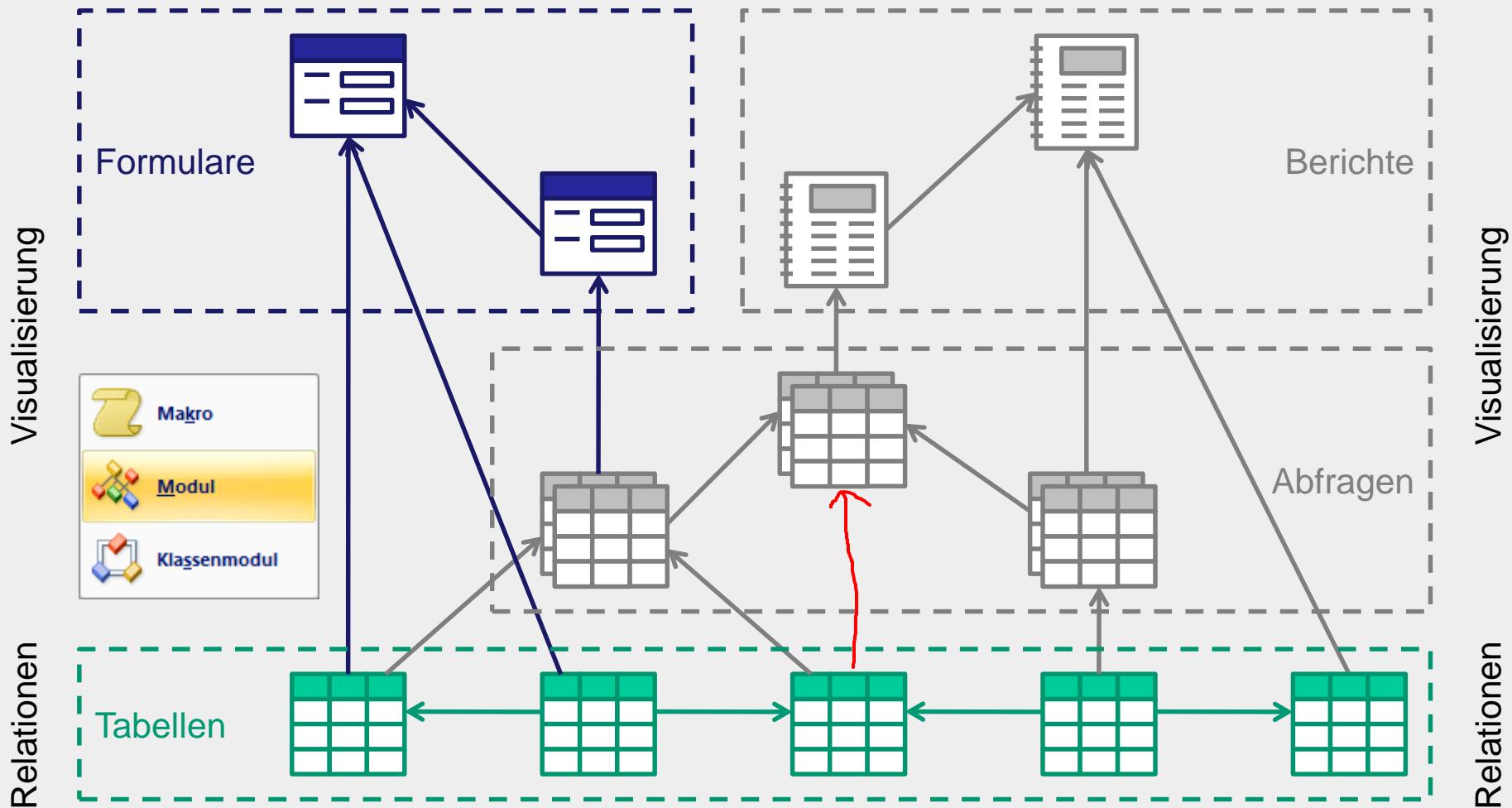
---

1. Inhalte und Aufgaben der Wirtschaftsinformatik
2. Grundlagen der Informatik und der Informationstechnik
3. Informationsmanagement
4. Modellierung
- 5. Datenbanken**
6. Softwareentwicklung
7. Betriebliche Informationssysteme

## „Komponenten“ in MS Access

- **Tabellen**: speichern strukturierte Daten tabellarisch und physisch.
- **Abfragen**: stellen Daten aus Tabellen und Abfragen zur Laufzeit zusammen.
- **Formulare**: repräsentieren Daten aus Tabellen, Abfragen und Formularen für CRUD-Operationen.
- **Berichte**: bereiten Daten aus Tabellen, Abfragen und Berichten für tabellarische Ausgabe auf.
- **Module** (und Makros): erledigen prozedural (objektorientiert), was nicht beim Entwurf anderer Komponenten deklarativ festgelegt werden kann.

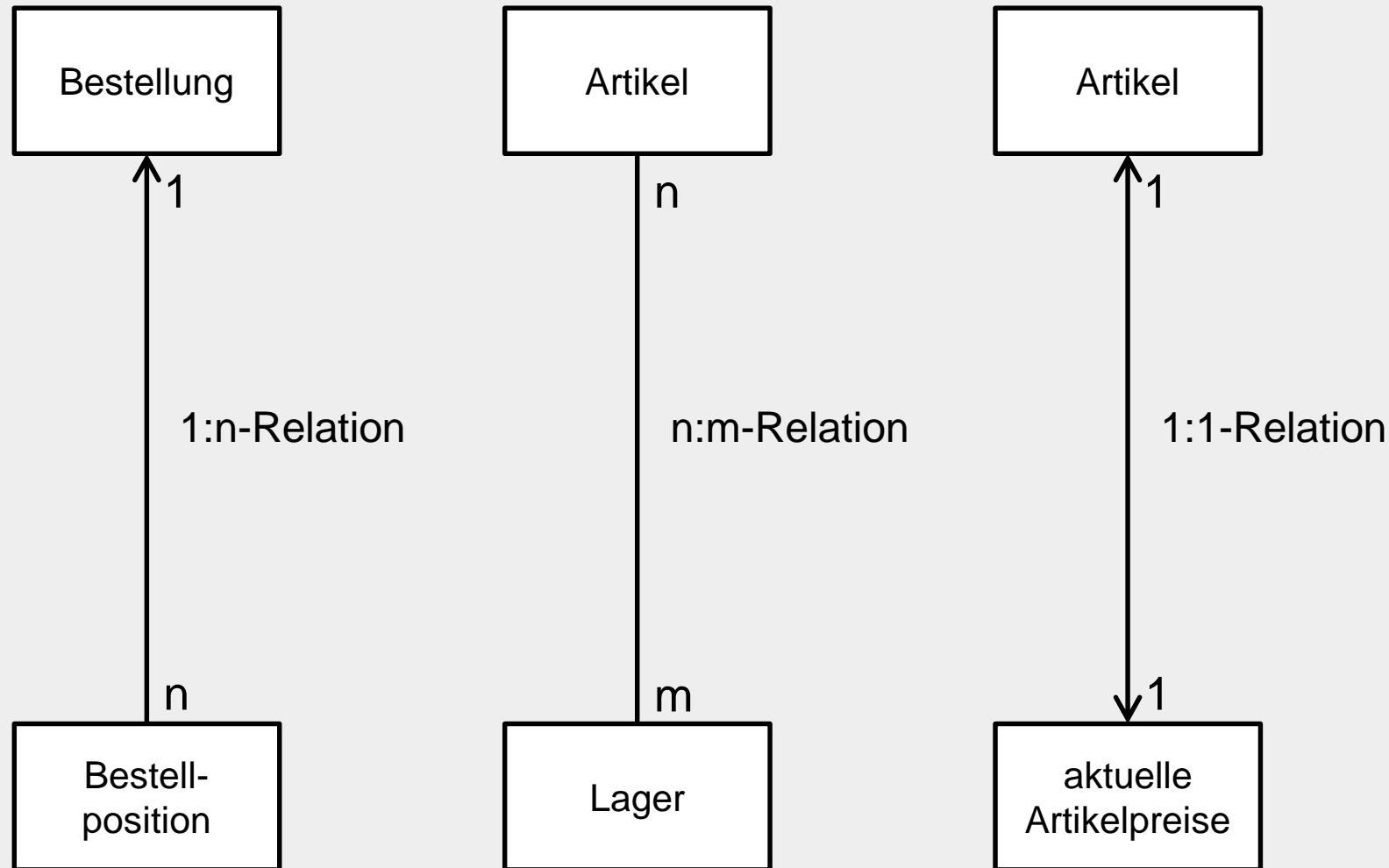
## Zusammenspiel der Komponenten



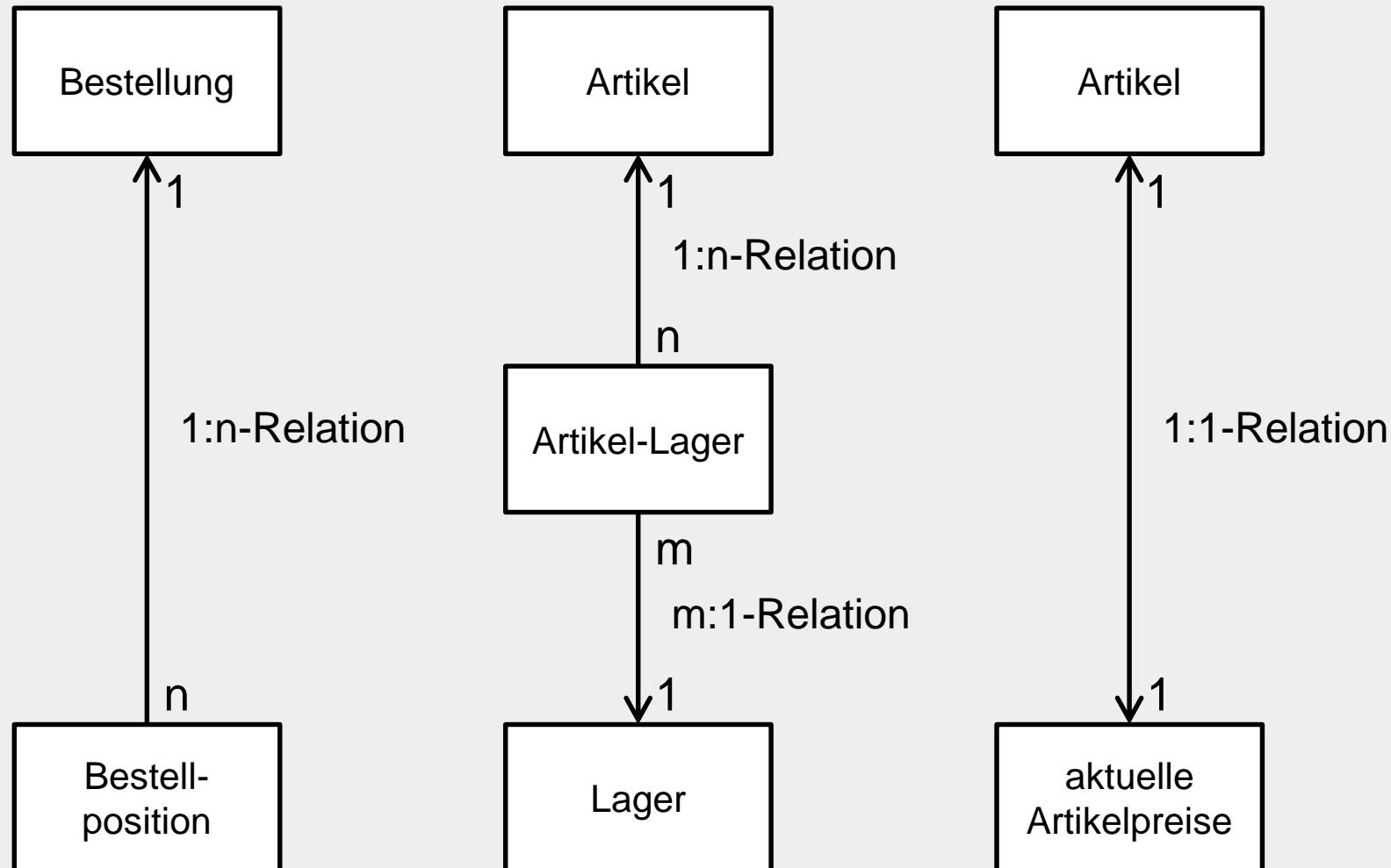
## Relationen und Kardinalität

- Datenbanktheorie: Relationen .. Tabellen
- Entity-Relationship-Model: Relationen .. Verknüpfungen zwischen Entitäten
- Relationen im hiesigen Sinne: Verknüpfungen zwischen Tabellen
- Kardinalität: geforderte/erlaubte Anzahl von Datensätzen auf beiden Seiten einer Relation

# Relationen und Kardinalität im logischen Datenmodell



# Relationen und Kardinalität im physischen Datenmodell



## 1:n-Relation

Primärschlüssel

1:n

Fremdschlüssel

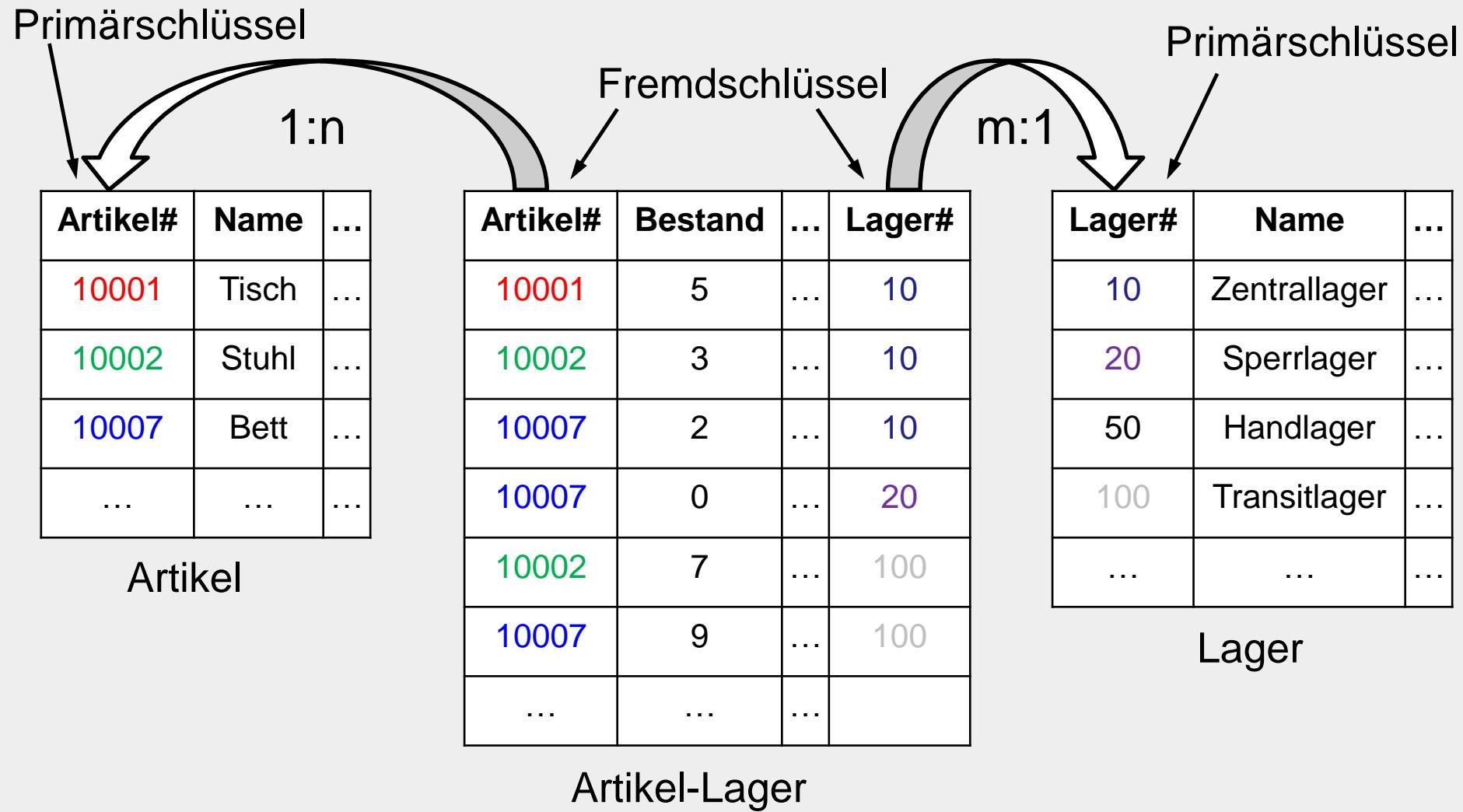
Bestell #	Lieferant	Termin	...
10001	Müller AG	15.06.2011	...
10002	Meyer GmbH	27.05.2011	...
10003	Schulz KG	(NULL)	...
10007	Schmidt GbR	17.02.2012	...
...	...	...	...

Bestellung

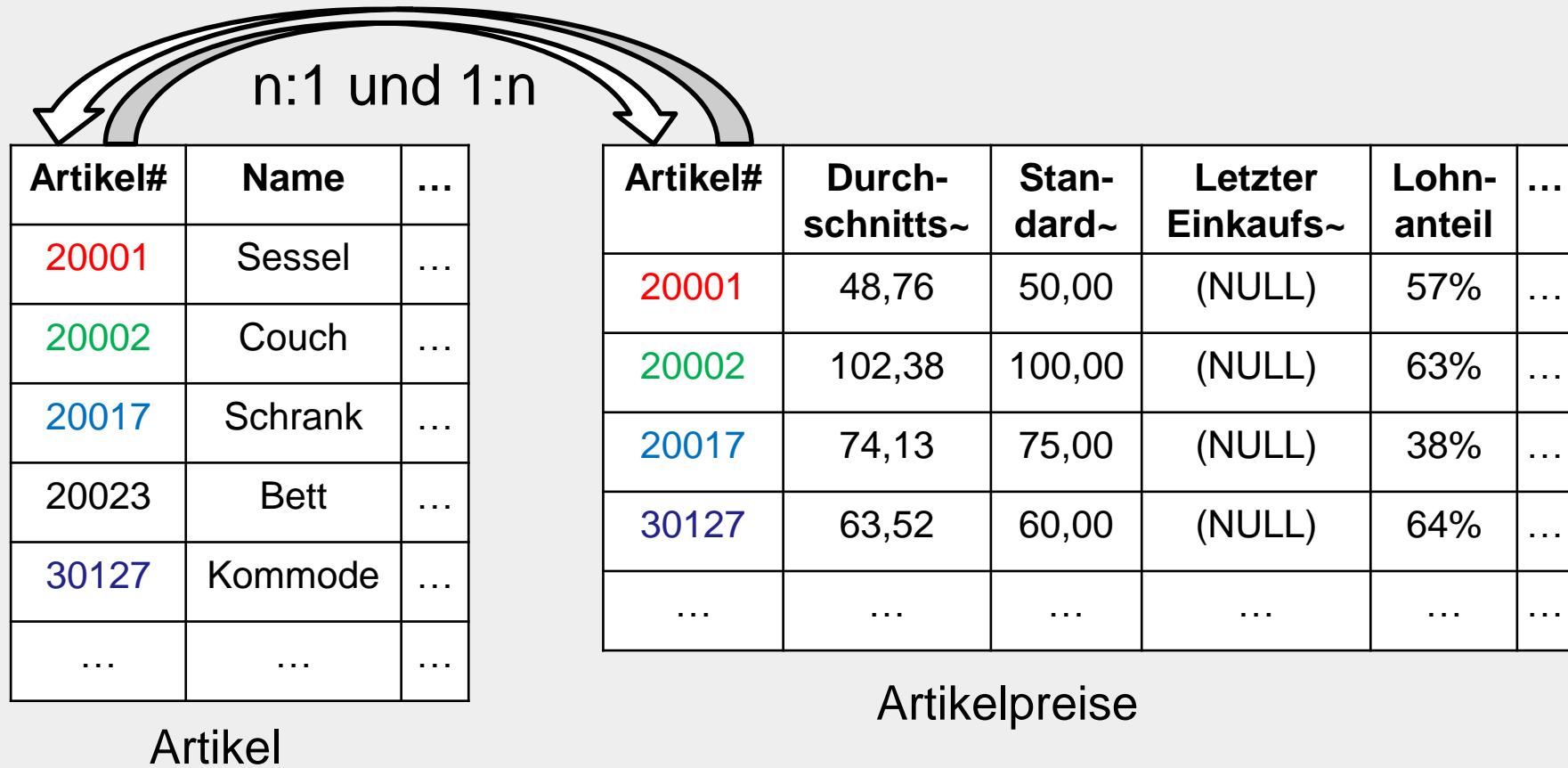
Bestell #	Positions#	Artikel	Menge	...
10001	1,0	Tisch	1	...
10001	2,0	Stuhl	4	...
10001	3,0	Bank	1	...
10002	1,0	Couch	1	...
10007	1,0	Bett	2	...
10007	2,0	Schrank	1	...
...	...			...

Bestellposition

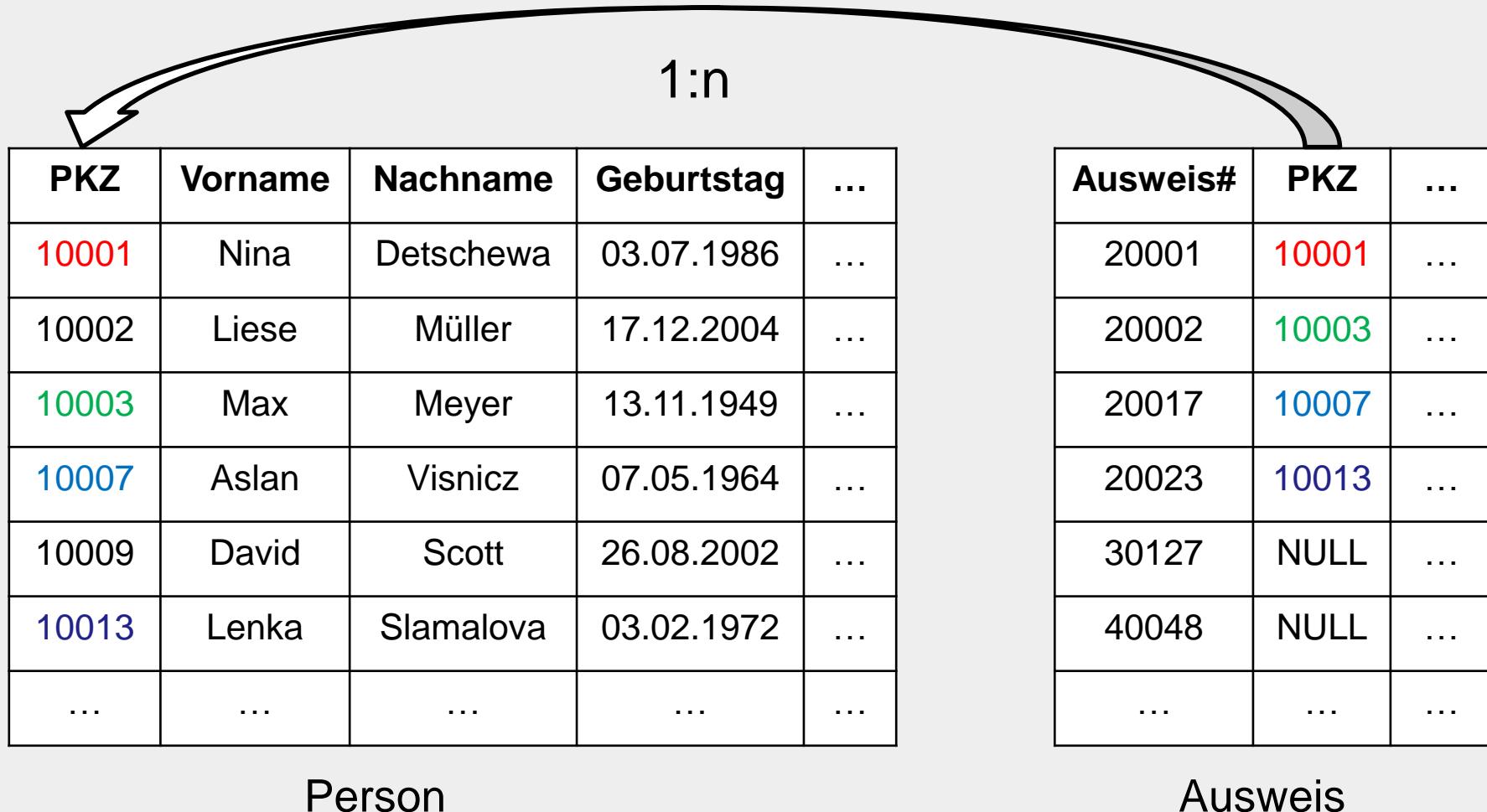
## Logische n:m- als physische 1:n- und m:1-Relation



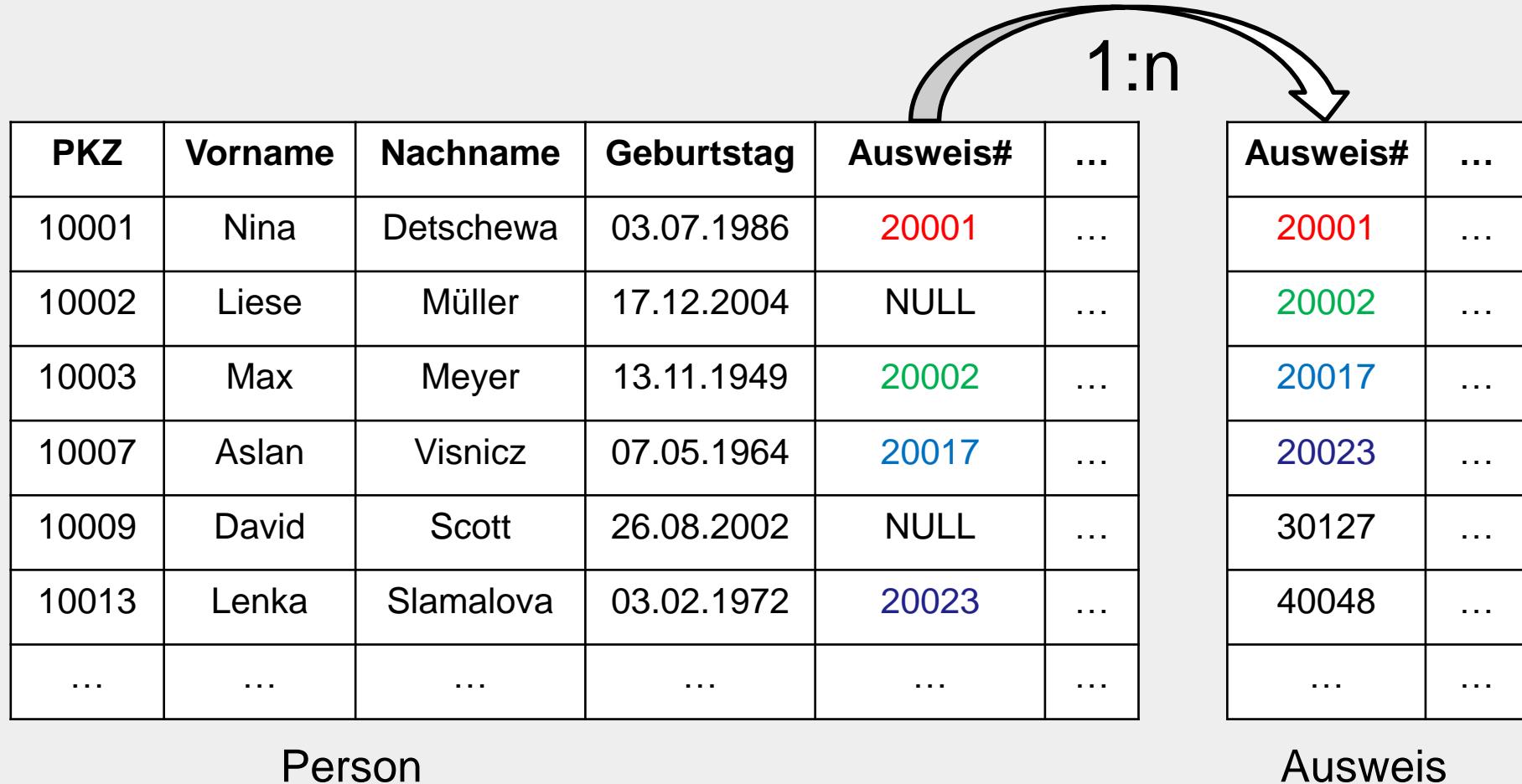
## 1:1-Relation als n:1- und 1:n-Relation



## 1:1-Relation als 1:n-Relation



# 1:1-Relation als n:1-Relation



# Verknüpfte Tabellen und Normalisierung gegen Redundanz, Lösch- und Änderungsanomalien

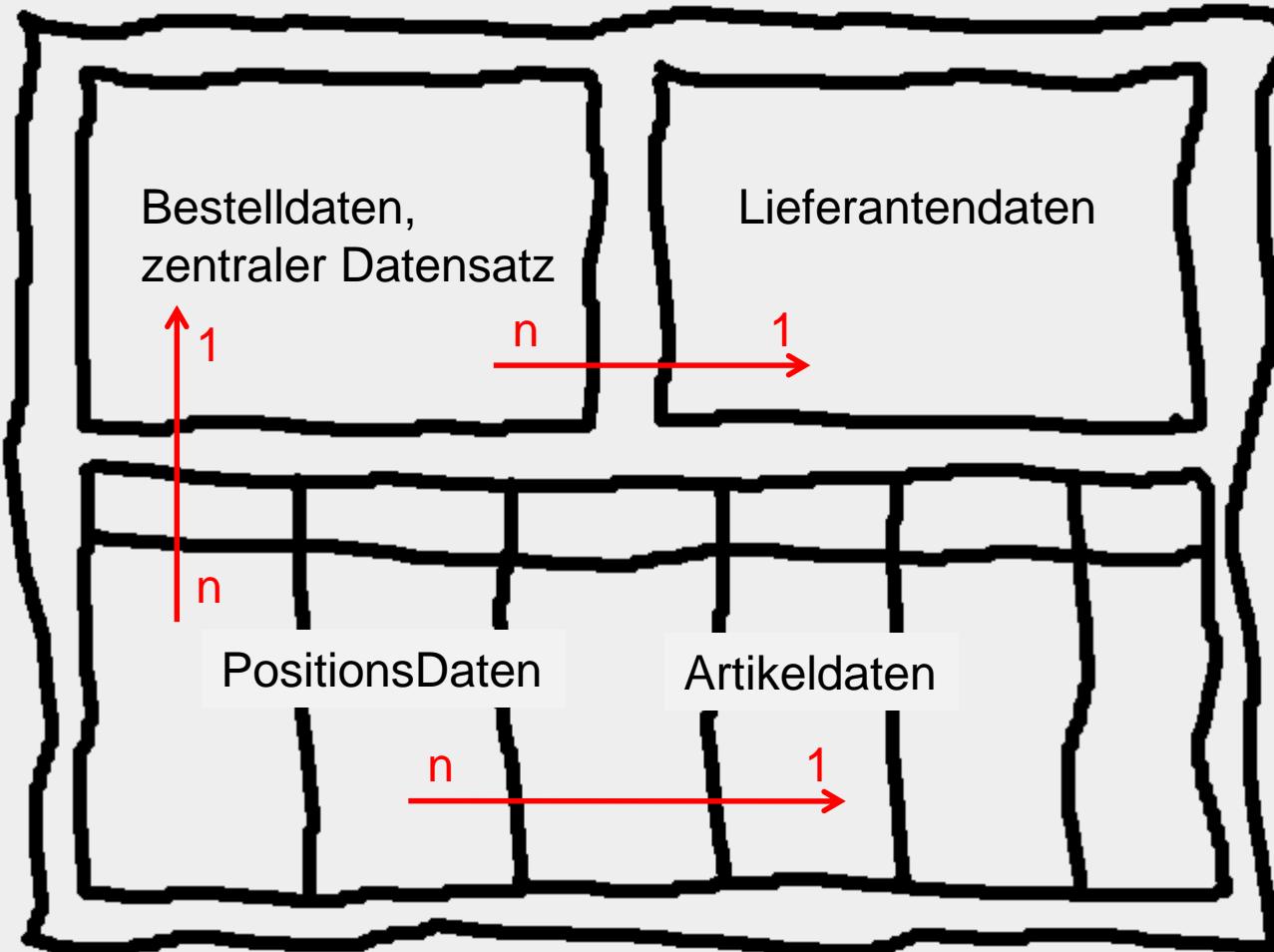
Bestell #	Firma	Termin	Positions#	Artikel	Menge	...
10001	Müller AG	15.06.2011	1,0	Tisch	1	...
10001	Müller AG	15.06.2011	2,0	Stuhl	4	...
10001	Müller AG	15.06.2011	3,0	Bank	1	...
10002	Meyer GmbH	27.05.2011	1,0	Couch	1	...
10003	Schulz KG	(NULL)	(NULL)	(NULL)	(NULL)	...
10007	Schmidt GbR	17.02.2012	1,0	Bett	2	...
10007	Schmidt GbR	17.02.2012	2,0	Schrank	1	...
...			...			...

Firma, Bestellung, Bestellposition und Artikel in einer Tabelle

## Datenkonsistenz mittels referenzieller Integrität

- **Löschweitergabe (ON DELETE CASCADE)**: Beim Löschen einer Bestellung müssen die Bestellpositionen automatisch gelöscht werden.
- **Löschrestriktion (ON DELETE RESTRICT/DENY)**: Ein Artikel darf nicht gelöscht werden, sofern er noch in einer Bestellposition vorkommt.
- **ON DELETE SET NULL**: Beim Löschen eines Projektes müssen die Fremdschlüssel NULL gesetzt werden, die von zugehörigen Bestellungen auf das Projekt zeigen.
- **Aktualisierungsweitergabe**: nicht verwenden

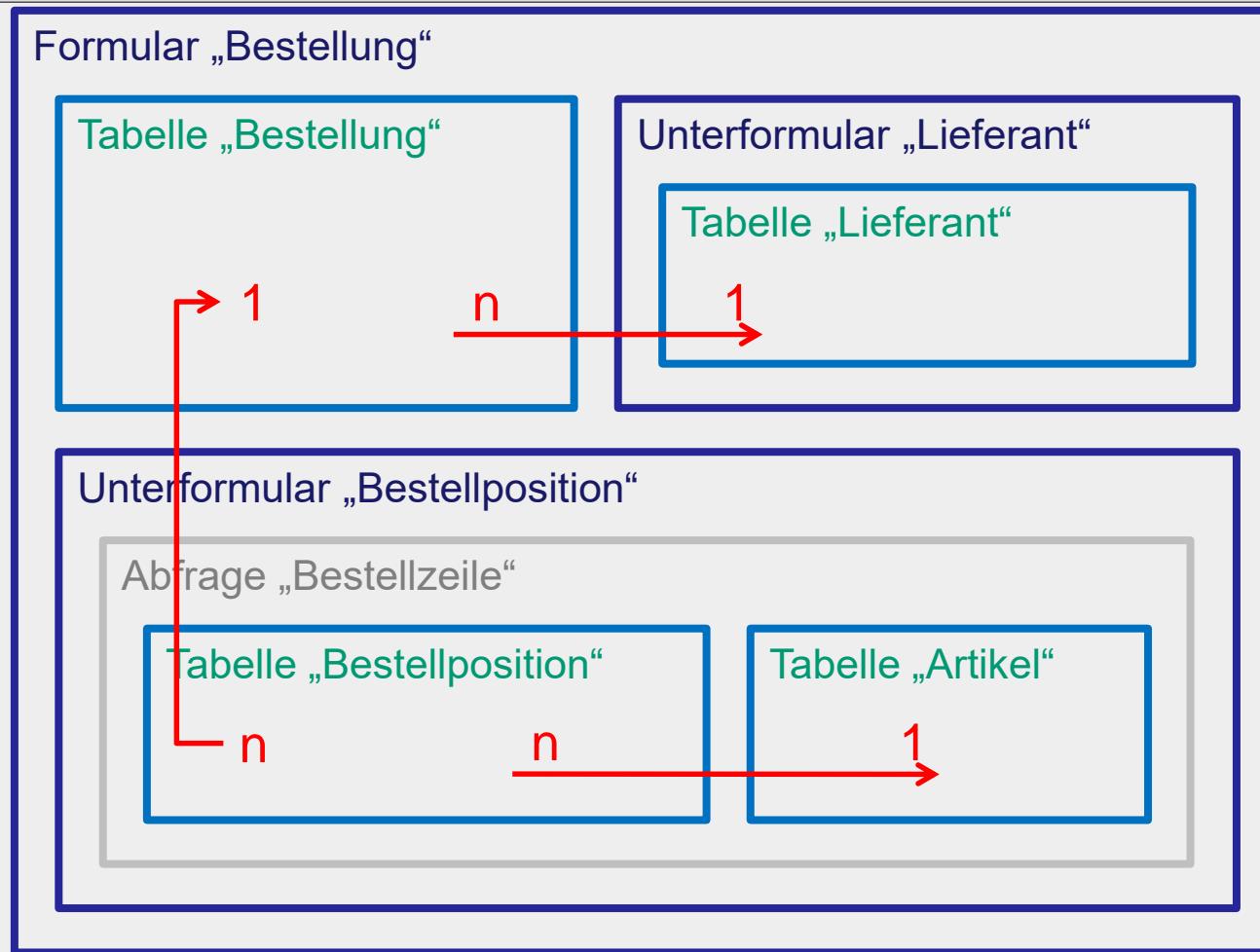
## Entwurf Bestellformular



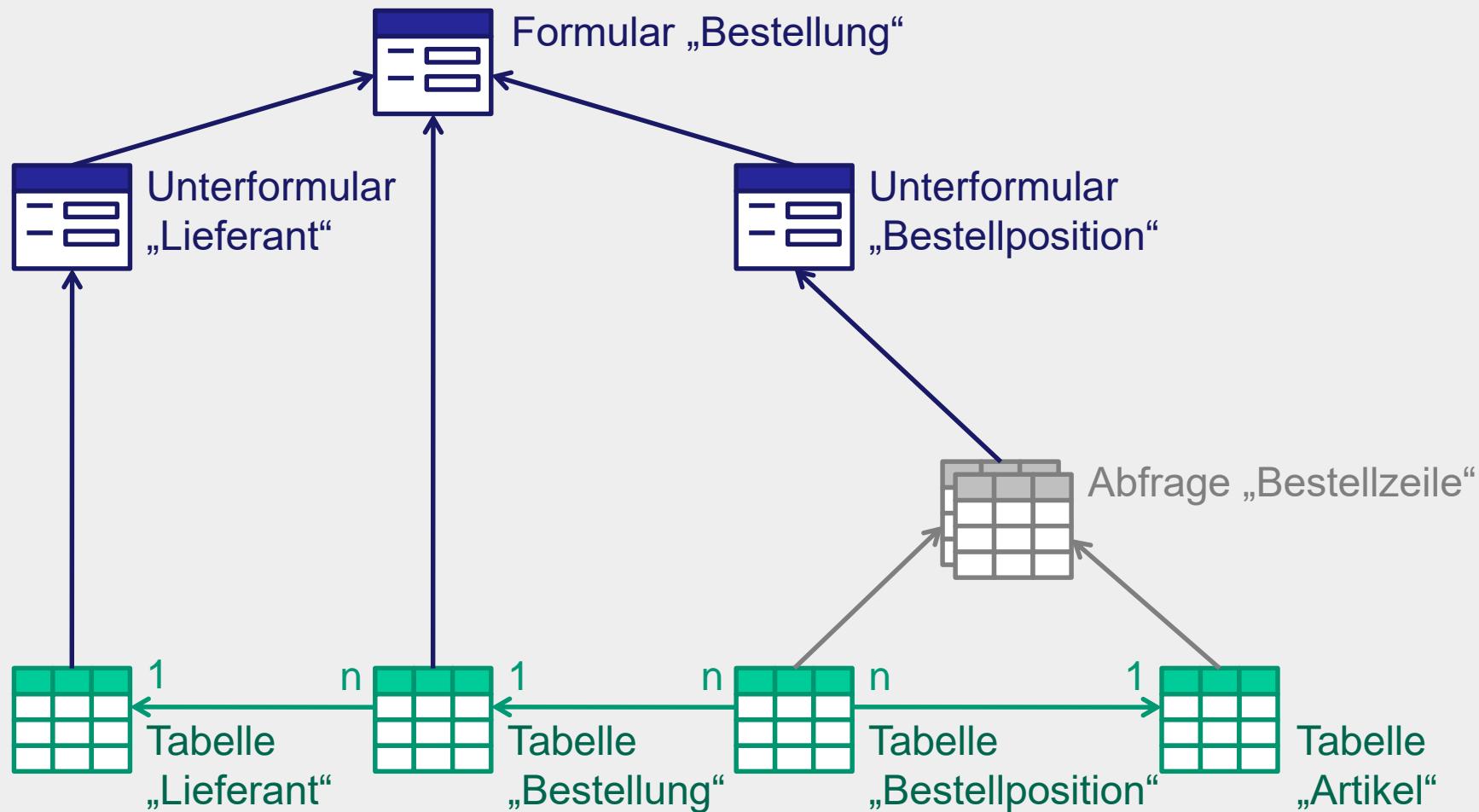
C  
R  
U  
D

... -Operationen für  
Bestellungen und  
Bestellpositionen,  
aber nicht für  
Lieferanten und  
Artikel in diesem  
Formular

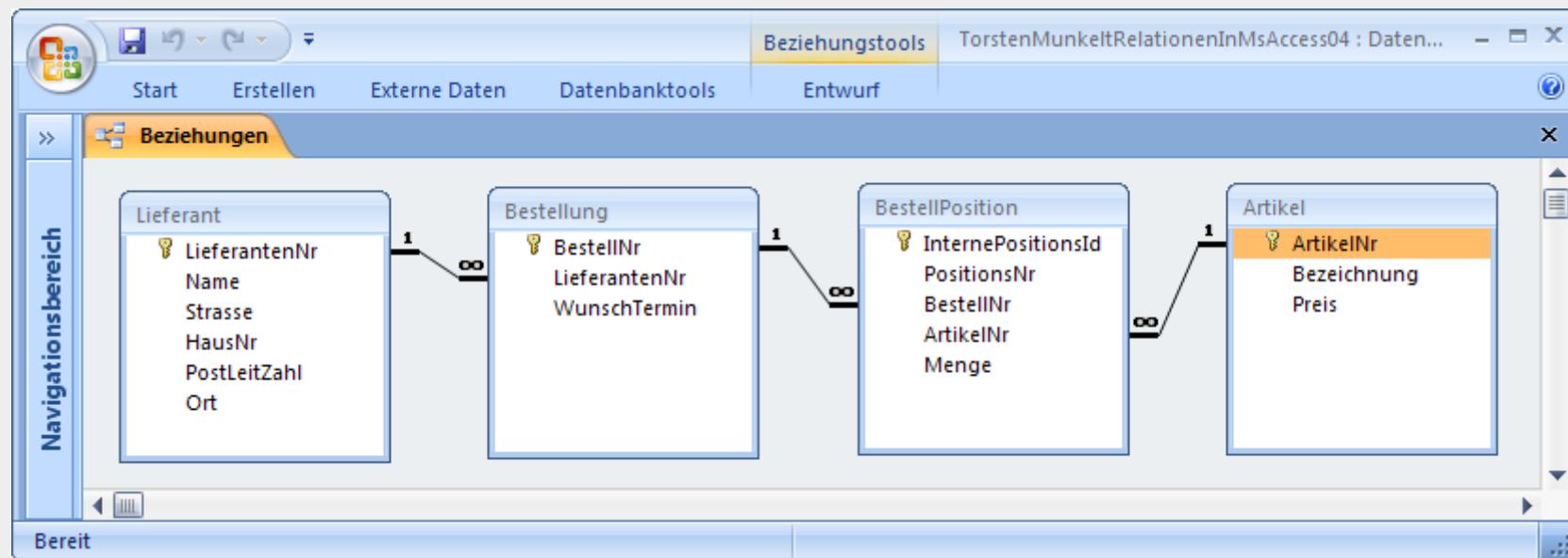
## „Block“-Architektur Bestellformular



## „Graph“-Architektur Bestellformular

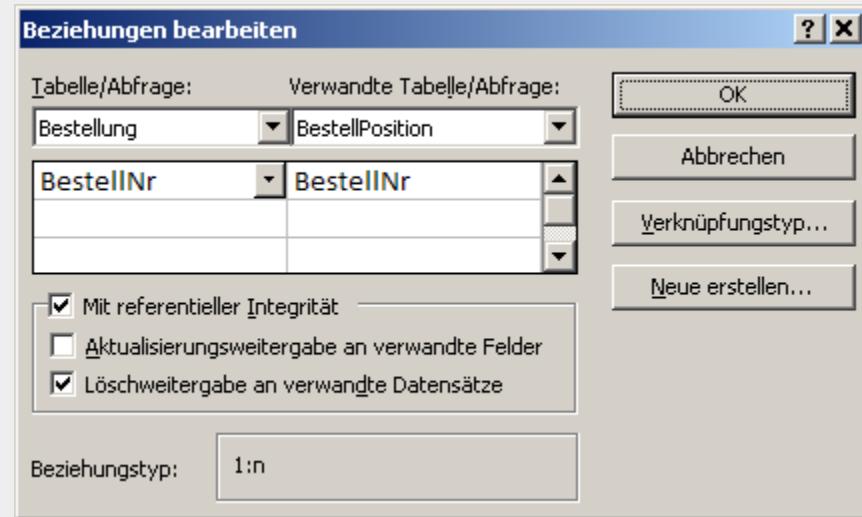
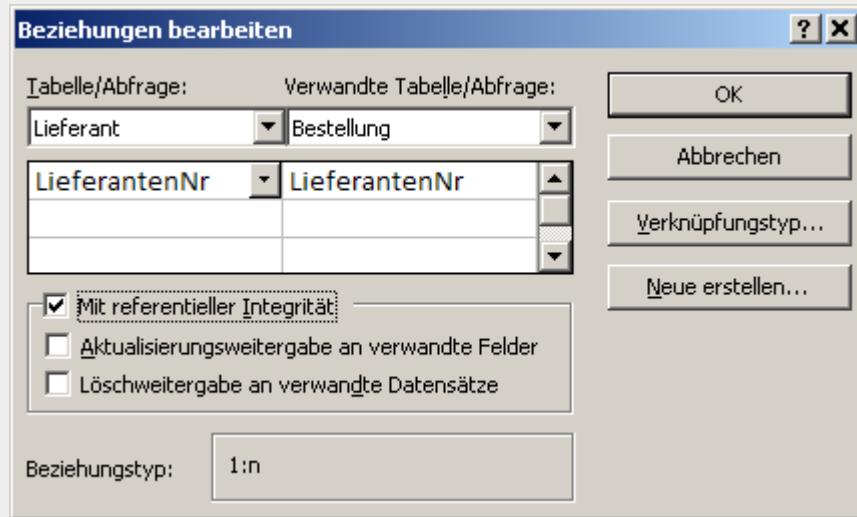


# Datenmodell „Bestellung“ (Tabellen und „Relationen“)



Doppelklick auf Relation ...

# Löschrrestriktion und -weitergabe



# „Nachschlagen“ von Werten aus verknüpften Tabellen

Bestellung

Feldname	Felddatentyp	Beschreibung
BestellNr	AutoWert	
LieferantenNr	Zahl	
WunschTermin	Datum/Uhrzeit	

Allgemein Nachschlagen

Steuerelement anzeigen Kombinationsfeld

Herkunftstyp Tabelle/Abfrage

Datensatzherkunft `SELECT [Lieferant].[LieferantenNr], [Lieferant].[Name] FROM Lieferant ORDER BY [Name], [LieferantenNr];`

BestellPosition

Feldname	Felddatentyp	Beschreibung
PositionsNr	Zahl	
BestellNr	Zahl	
ArtikelNr	Zahl	

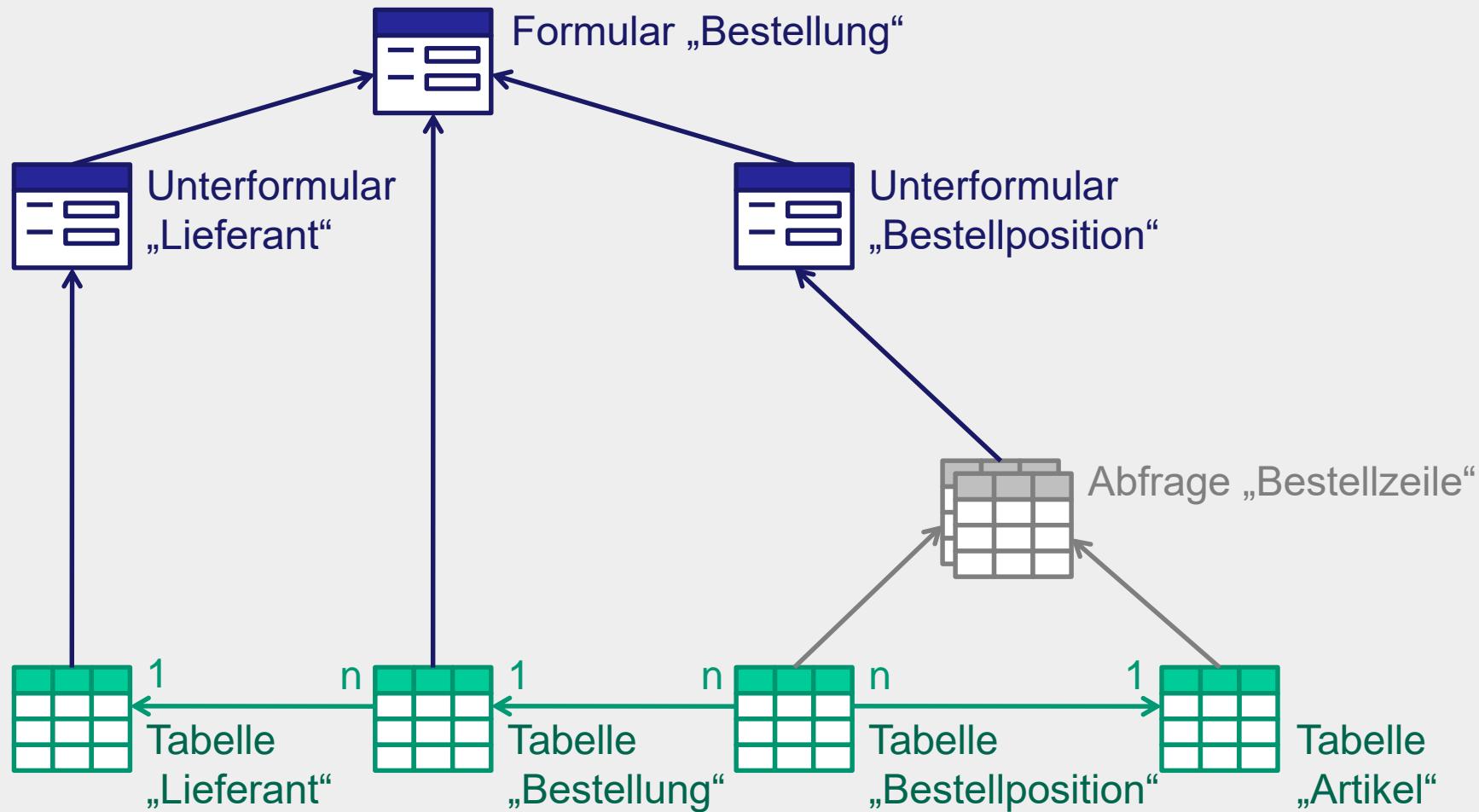
Allgemein Nachschlagen

Steuerelement anzeigen Kombinationsfeld

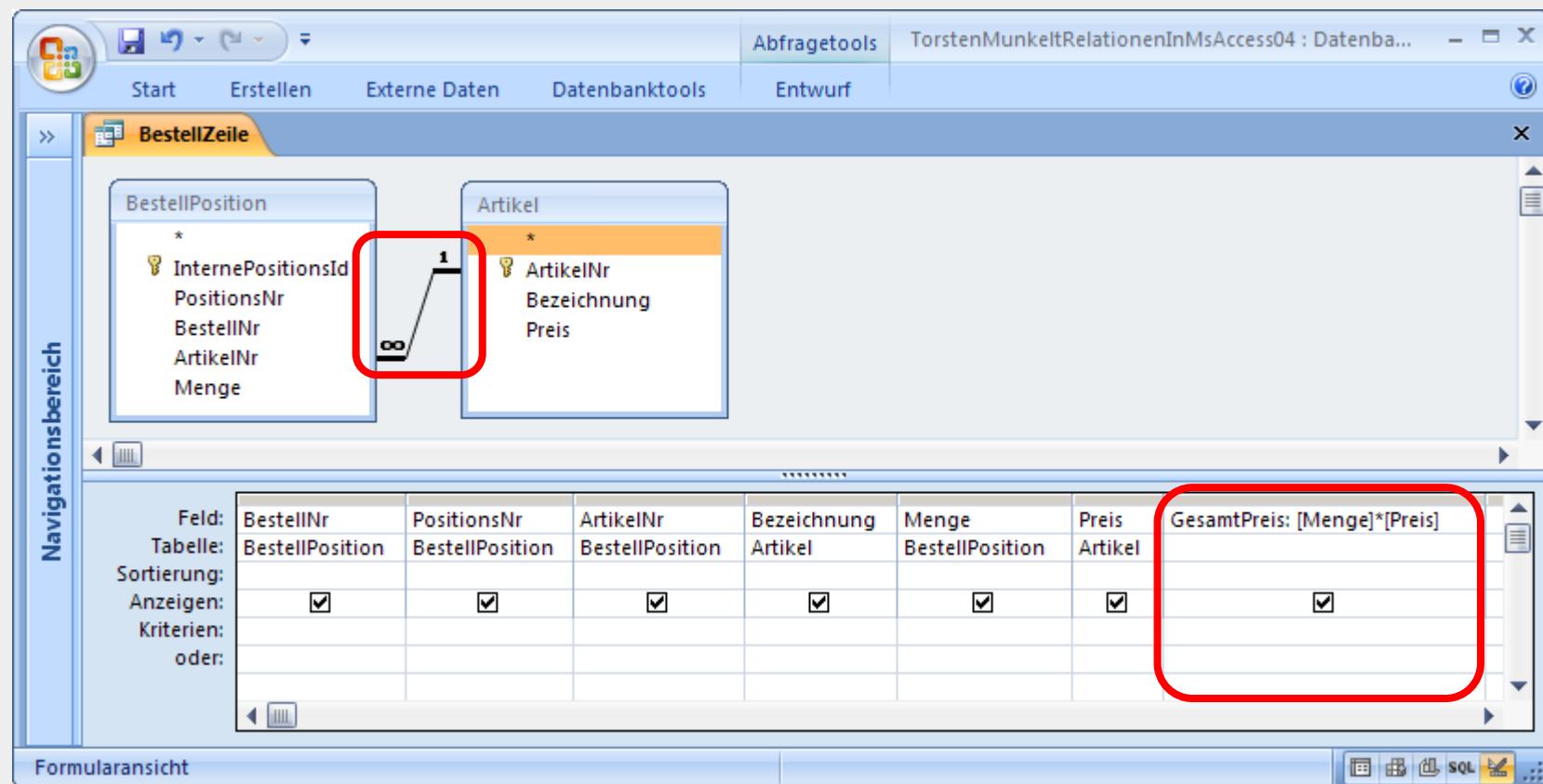
Herkunftstyp Tabelle/Abfrage

Datensatzherkunft `SELECT [Artikel].[ArtikelNr], [Artikel].[Bezeichnung] FROM Artikel ORDER BY [Bezeichnung], [ArtikelNr];`

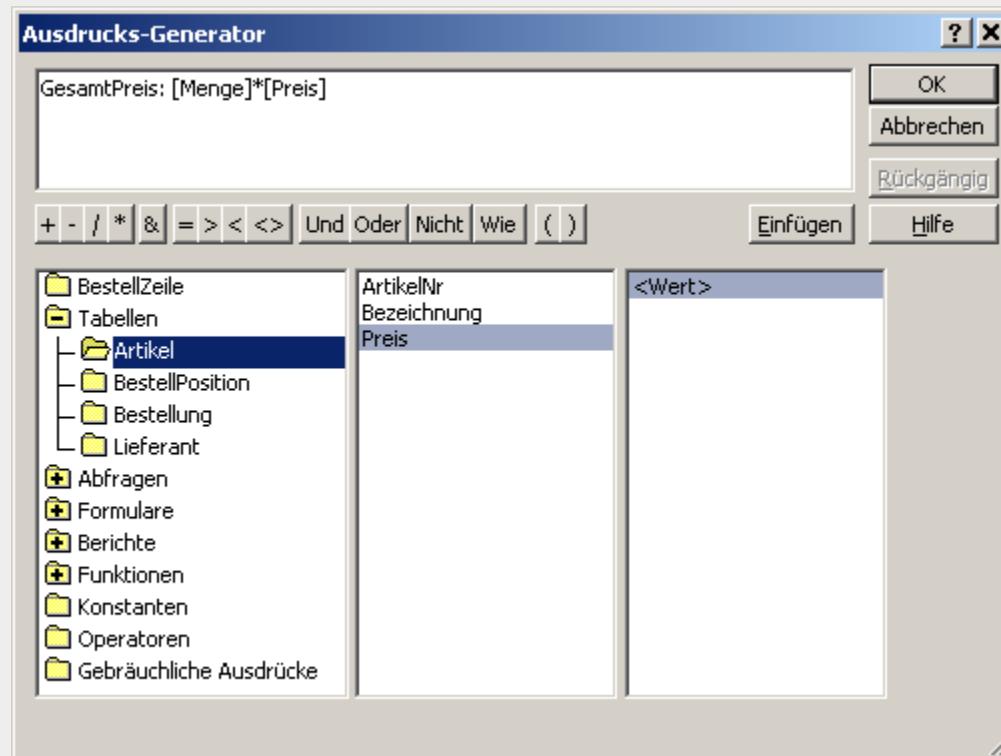
## „Graph“-Architektur Bestellformular



## Abfrage „Bestellzeile“



# Ausdrucksgenerator

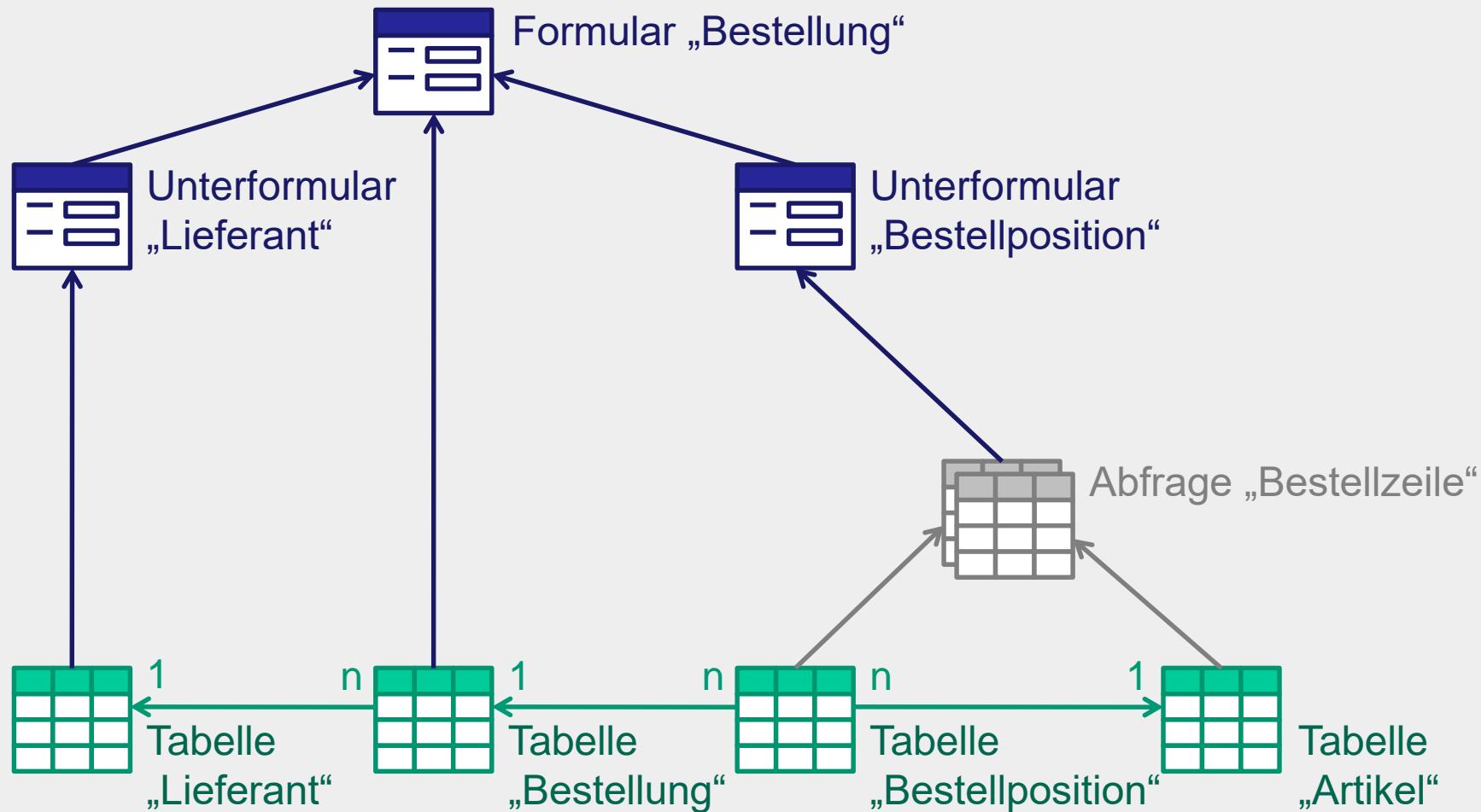


# Ergebnis der Abfrage

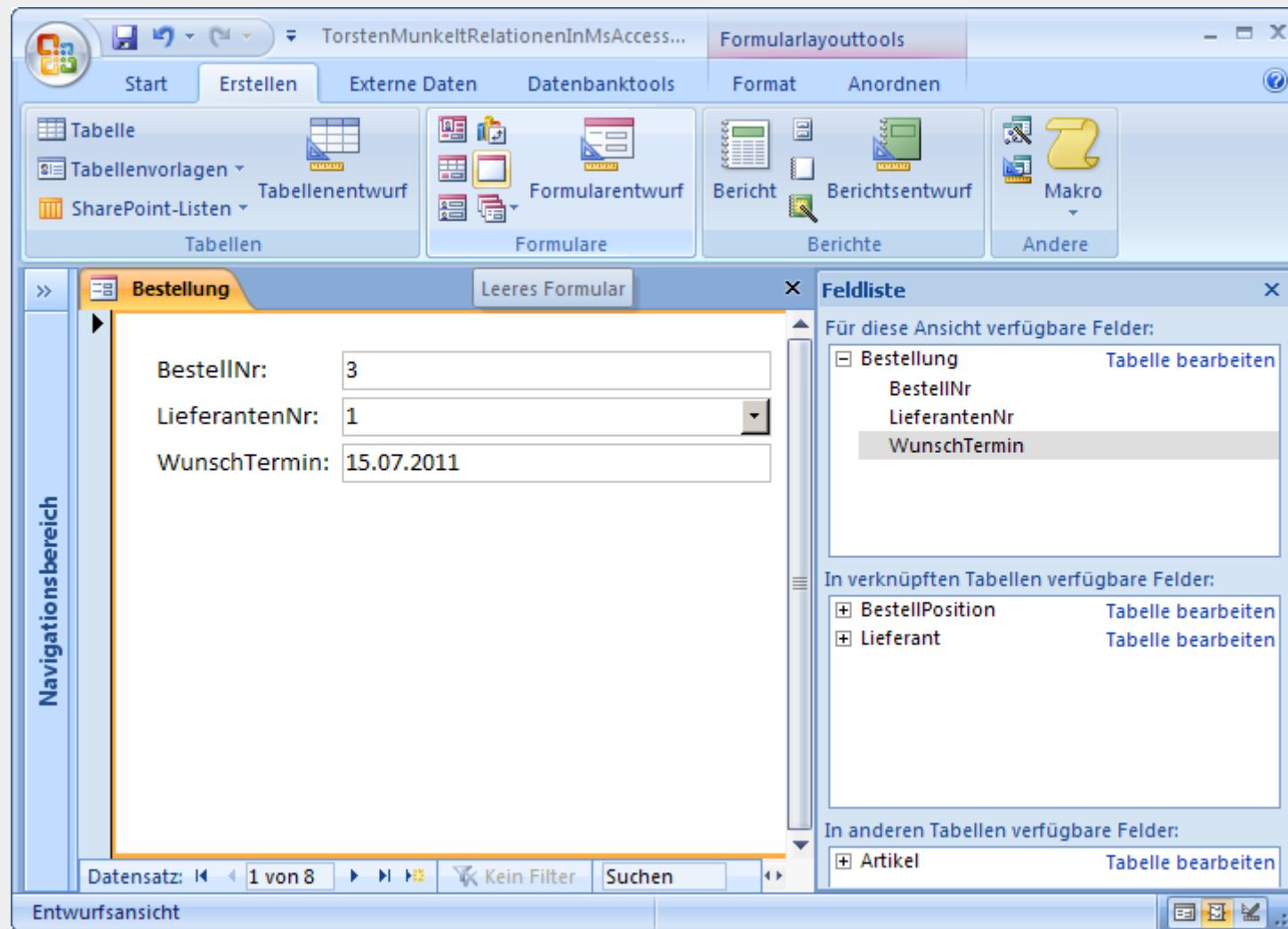
The screenshot shows a Microsoft Access application window titled "TorstenMunkeltRelationenInMsAccess05 : Datenbank (Access 2007) - Microsoft Access". The ribbon tabs are "Start", "Erstellen", "Externe Daten", and "Datenbanktools". On the left, the "Alle Tabellen" navigation pane lists tables: Bestellung, BestellPosition, BestellVolumenPro..., BestellZeile, Bestellposition, Lieferant, and Artikel. The "BestellZeile" table is selected and displayed in the main grid. The grid has columns: BestellNr, PositionsNr, ArtikelNr, Bezeichnung, Menge, Preis, and GesamtPreis. The data shows various items like Tisch, Bett, Couch, Hocker, Schrank, Sessel, and Stuhl with their respective quantities and prices. A red box highlights the last row for "Hocker" with values: PositionsNr 5, ArtikelNr 7, Bezeichnung "Hocker", Menge 4, Preis 25, and GesamtPreis 100.

BestellNr	PositionsNr	ArtikelNr	Bezeichnung	Menge	Preis	GesamtPreis
3	1	1	Tisch	2	100	200
3	4	6	Bett	1	100	100
3	2	4	Couch	8	35	280
4	3	7	Hocker	2	80	160
10	1	5	Schrank	2	80	160
3	2,5	3	Sessel	1	170	170
10	2	2	Stuhl	1	170	170
3	3	1	Tisch	1	290	290
3	5	7	Hocker	4	25	100
10	3	7	Hocker	1	25	25

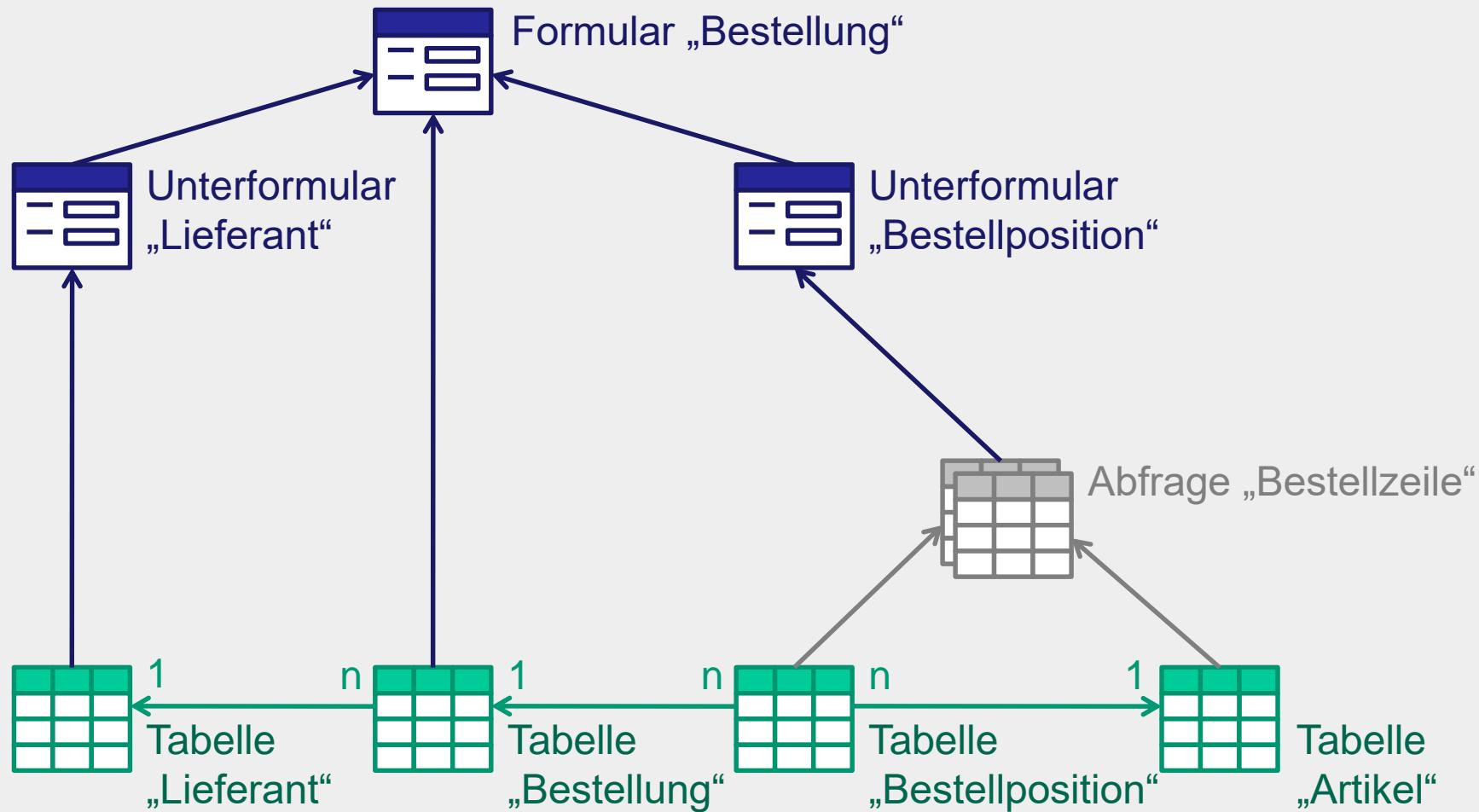
## „Graph“-Architektur Bestellformular



# Hauptformular „Bestellung“



## „Graph“-Architektur Bestellformular



# Entwurf Unterformular „Bestellposition“

The screenshot shows the Microsoft Access Form Design View. The main window displays a subform titled "Bestellposition" with a grid of six rows and two columns. The columns are labeled "BestellNr:" and "PositionsNr:". The rows contain the following data:

	BestellNr:	PositionsNr:
1		
2	BestellPosition : Tab...	PositionsNr
3	BestellZeile	ArtikelNr
4	Bestellposition	Bezeichnung
5	Menge	Menge
6	Preis	Preis
7	GesamtPreis	GesamtPreis

The "Formularentwurfstools" tab is selected in the ribbon. The "Formularentwurf" section of the ribbon is highlighted. The "Berichte" section of the ribbon is also visible. The "Eigenschaftenblatt" (Properties) pane is open on the right, showing properties for the "Formular" type. A red box highlights the "Format" tab of the properties pane, which contains settings for orientation, size, and layout.

Format	Daten	Ereignis	Andere	Alle
Beschaffung	Standardansicht	Datenblatt		
	Formularansicht zulassen	Nein		
	Datenblattansicht zulassen	Ja		
	PivotTable-Ansicht zulassen	Nein		
	PivotChart-Ansicht zulassen	Nein		
	Layoutansicht zulassen	Ja		
Bild	(keines)			
	Bild nebeneinander	Nein		
	Bildausrichtung	Mitte		
	Bildtyp	Eingebettet		
	Bildgrößenmodus	Abschneiden		
	Breite	8,79cm		
	Automatisch zentrieren	Nein		
	Größe anpassen	Ja		
	An Bildschirmgröße anpassen	Ja		
	Rahmenart	Veränderbar		

# Datenblatt Unterformular „Bestellposition“

The screenshot shows a Microsoft Access application window titled "TorstenMunkeltRelationenInMsAccess05 : Datenbank (Access 2007) - Microsoft Access". The ribbon tabs are "Start", "Erstellen", "Externe Daten", and "Datenbanktools". The "Ansichten" button is selected in the ribbon. The main area displays a datasheet for the "Bestellposition" table. A red box highlights the table name "Bestellposition" in the list of tables on the left. The table has columns: BestellNr, PositionsNr, ArtikelNr, Bezeichnung, Menge, Preis, and GesamtPreis. The data shows various items like Tisch, Stuhl, Sessel, Couch, Bett, and Hocker with their respective quantities and prices. The status bar at the bottom shows "Datensatz: 1 von 10" and "Kein Filter".

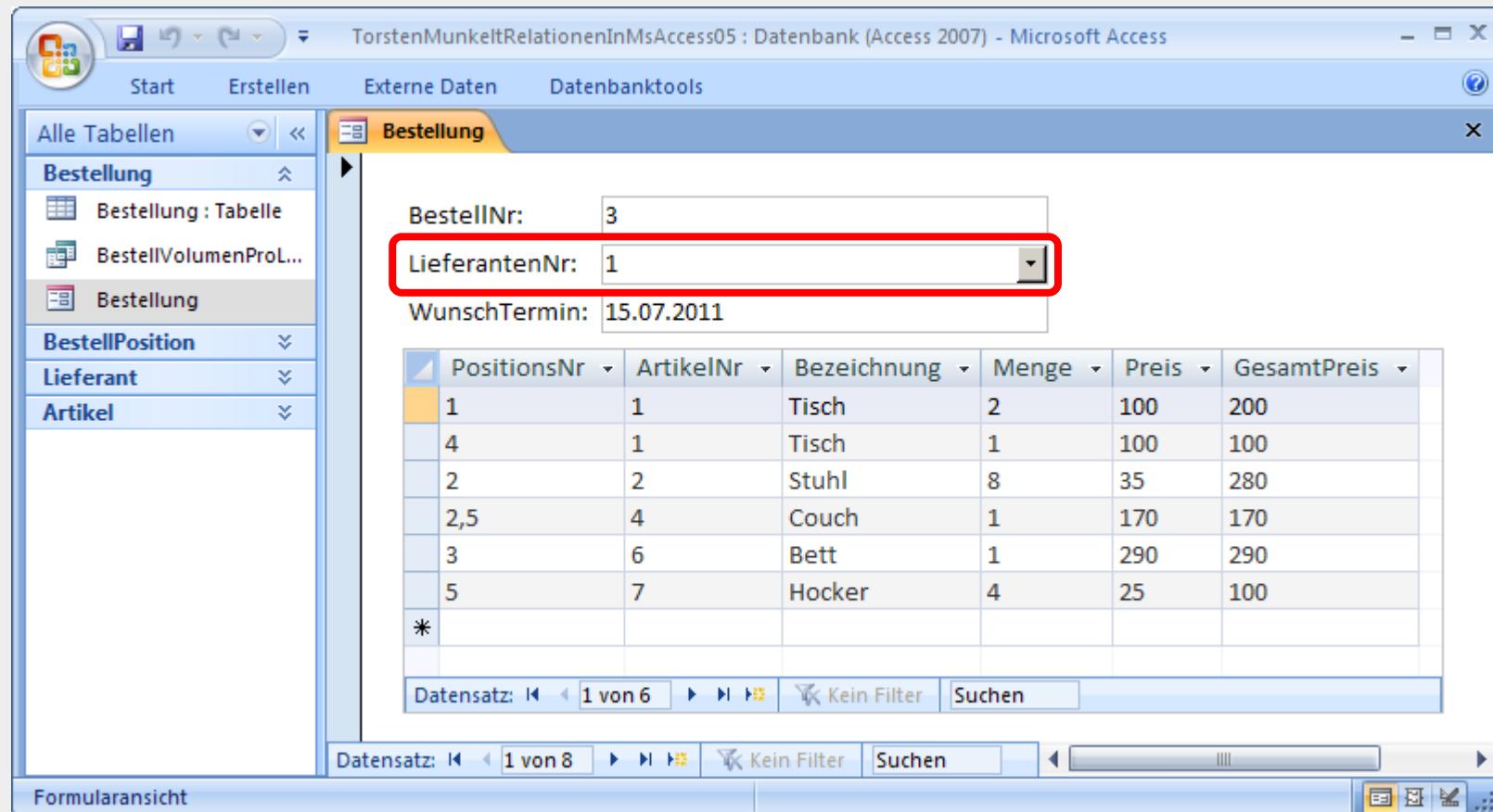
	BestellNr	PositionsNr	ArtikelNr	Bezeichnung	Menge	Preis	GesamtPreis
3	1	1	Tisch	2	100	200	
3	4	1	Tisch	1	100	100	
3	2	2	Stuhl	8	35	280	
4	3	3	Sessel	2	80	160	
10	1	3	Sessel	2	80	160	
3	2,5	4	Couch	1	170	170	
10	2	4	Couch	1	170	170	
3	3	6	Bett	1	290	290	
3	5	7	Hocker	4	25	100	
10	3	7	Hocker	1	25	25	
*							

# Entwurf „Bestellung“ mit „-positionen“

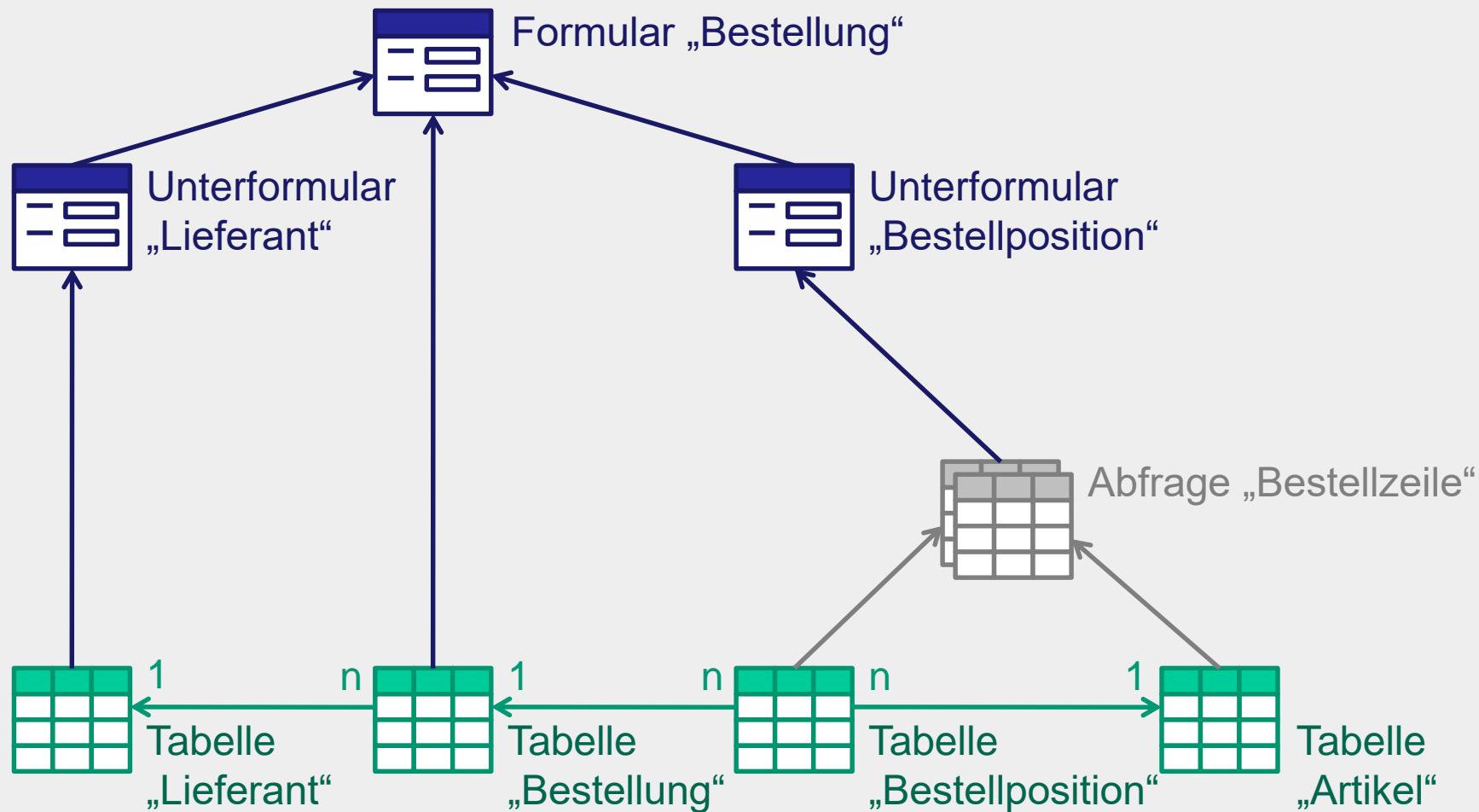
The screenshot shows the Microsoft Access Form Design View. The main window displays the 'Bestellung' form, which contains a subform titled 'Detailbereich' (Detail section) located at the bottom. The subform displays data from the 'BestellPosition' table. The properties window on the right is set to the 'Bestellposition' category under 'Auswahltyp: Unterformular/-bericht'. A red box highlights the 'Daten' tab of the properties window, which shows the following settings:

Herkunftsobjekt	Bestellposition
Verknüpfen nach	BestellNr
Verknüpfen von	BestellNr
Leeren Hauptentwurf	Ja
Aktiviert	Ja
Gesperrt	Nein

# Formular „Bestellung“ mit „-positionen“



## „Graph“-Architektur Bestellformular



# Unterformular „Lieferant“

The screenshot shows the Microsoft Access Formularentwurfstools interface. On the left, the navigation pane lists tables: Bestellung, BestellPosition, Lieferant, and Artikel. The 'Lieferant' table is selected. The main area displays a form titled 'Lieferant' with fields for LieferantenNr, Name, Strasse, HausNr, PostLeitZahl, and Ort. The 'Formularentwurfstools' tab is active in the ribbon. A floating 'Eigenschaftenblatt' (Propertiesheet) window is open, set to the 'Formular' type. It shows various properties, with the 'Datensatzmarkierer', 'Navigationsschaltflächen', and 'Navigationsbeschriftung' properties highlighted by a red rectangle.

Format	Daten	Ereignis	Andere	Alle
Bildtyp	Eingebettet			
Bildgrößenmodus	Abschneiden			
Breite	6,399cm			
Automatisch zentrieren	Nein			
Größe anpassen	Ja			
An Bildschirmgröße anpassen	Ja			
Reihenfolge	Veränderbar			
Datensatzmarkierer	Nein			
Navigationsschaltflächen	Nein			
Navigationsbeschriftung				
Trennlinien	Nein			
Bildlaufleisten	Nein			
Mit Systemmenüfeld	Ja			
Schließen Schaltfläche	Ja			
MinMaxSchaltflächen	Beide vorhanden			

# Entwurf „Bestellung“ mit „Lieferant“

The screenshot shows the Microsoft Access Formularentwurfstools (Form Design View). The main form is titled "Bestellung". It contains a "Detailbereich" (Detail section) with fields for "BestellNr:", "LieferantenNr:", and "WunschTermin:". A nested "Formularkopf" (Form header) and "Detailbereich" (Detail section) for the "Lieferant" table is embedded in the "Detailbereich" of the main form. This nested form has fields for "Name:" and "Strasse:". The "Lieferant" table is also listed in the "Alle Tabellen" (All Tables) list on the left.

**Eigenschaftenblatt (Propertiesheet)**

Aufklappbare Registerkarte: Lieferant

Herkunftsobjekt	Lieferant
Verknüpfen nach	LieferantenNr
Verknüpfen von	LieferantenNr
Leeren Hauptentwurf	Ja
Aktiviert	Ja
Gesperrt	Nein

Identifiziert Unterformular, -bericht oder Datei mit verknüpften Daten

# Formular „Bestellung“, vollständig

The screenshot shows a Microsoft Access application window titled "Bestellung". The main area displays a form with fields for "BestellNr" (3), "Name" (Müller GmbH), "LieferantenNr" (1), "Strasse" (Bahnhofsstr.), "WunschTermin" (15.07.2011), "HausNr" (23), "PostLeitzahl" (12345), and "Ort" (Berlin). Below the form is a data grid showing purchase details:

PositionsNr	ArtikelNr	Bezeichnung	Menge	Preis	GesamtPreis
1	1	Tisch	2	100	200
4	1	Tisch	1	100	100
2	2	Stuhl	8	35	280
2,5	4	Couch	1	170	170
3	6	Bett	1	290	290
5	7	Hocker	4	25	100
*					

Red annotations highlight relationships between the form fields and the data grid:

- An arrow points from the "LieferantenNr" field (containing "1") to the first row of the data grid (also containing "1").
- An arrow points from the "WunschTermin" field (containing "15.07.2011") to the second row of the data grid (containing "2").
- An arrow points from the "HausNr" field (containing "23") to the third row of the data grid (containing "8").
- An arrow points from the "PostLeitzahl" field (containing "12345") to the fourth row of the data grid (containing "1").
- An arrow points from the "Ort" field (containing "Berlin") to the fifth row of the data grid (containing "2").

At the bottom of the window, two sets of navigation buttons are highlighted with red boxes:

- The top set of buttons shows "1 von 6" (page 1 of 6).
- The bottom set of buttons shows "1 von 8" (page 1 of 8).

## Test der „Bestellung“

- Anlegen einer neuen Bestellung
- Auswahl eines Lieferanten
- Einfügen zweier Bestellpositionen
- jeweils Auswahl eines Artikels
- jeweils Eingabe der Menge
- Navigieren durch Bestellungen
- Navigieren durch Bestellpositionen
- ...

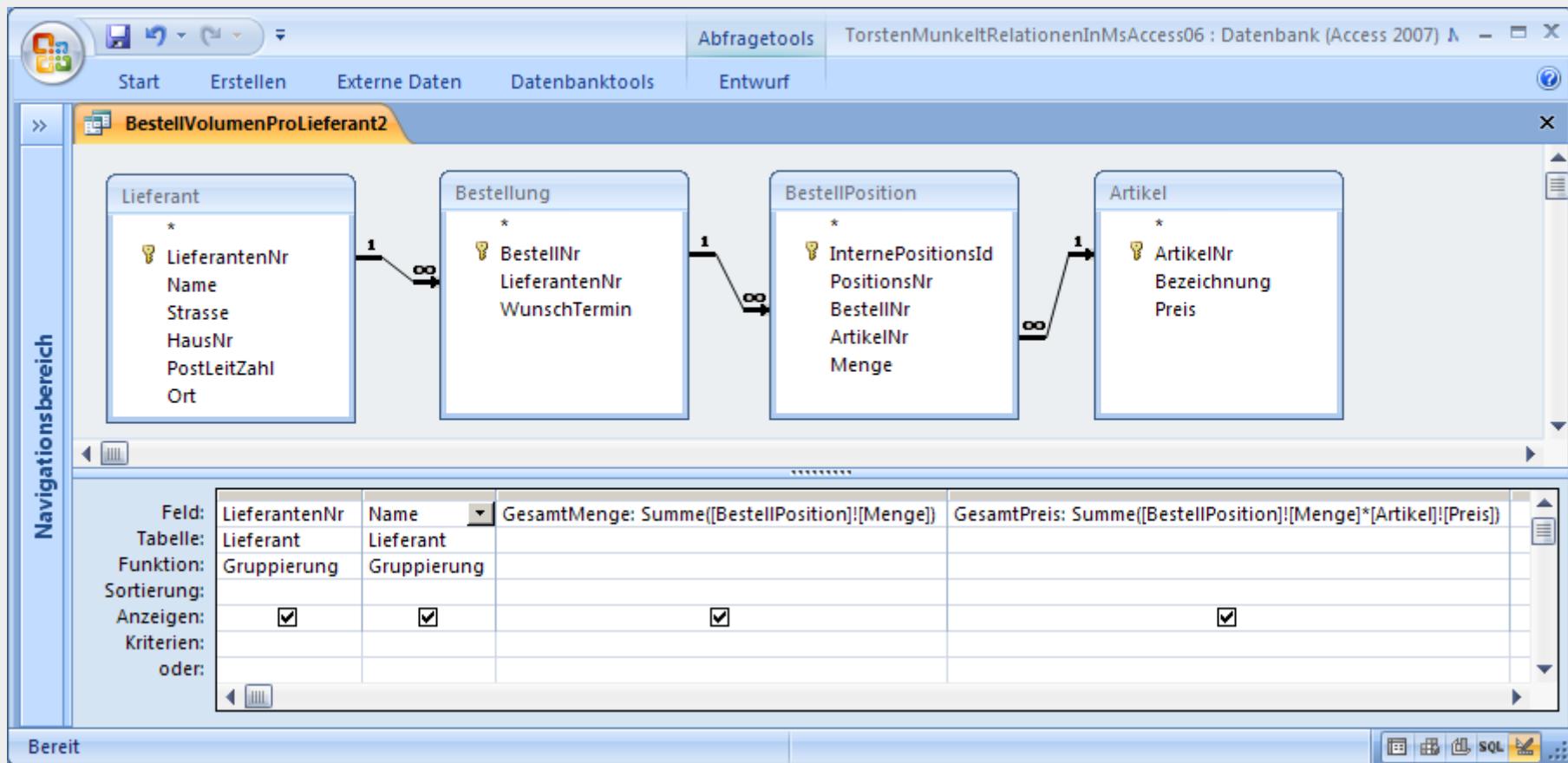
## Nächste Schritte

- Sortieren der Bestellpositionen nach PositionsNr. im Positionsformular bzw. in der Bestellzeilenabfrage
- Gesamtsumme pro Bestellung errechnen und anzeigen
- Sperren der Lieferantendaten gegen Änderungen im Bestellformular
- Sperren der Artikeldaten gegen Änderung im Positionsformular
- Separate Formulare für Lieferanten und Artikel (Stammdaten)
- Gestalten der vorhandenen Formulare
- Abfragen und Berichte über Bestellungen

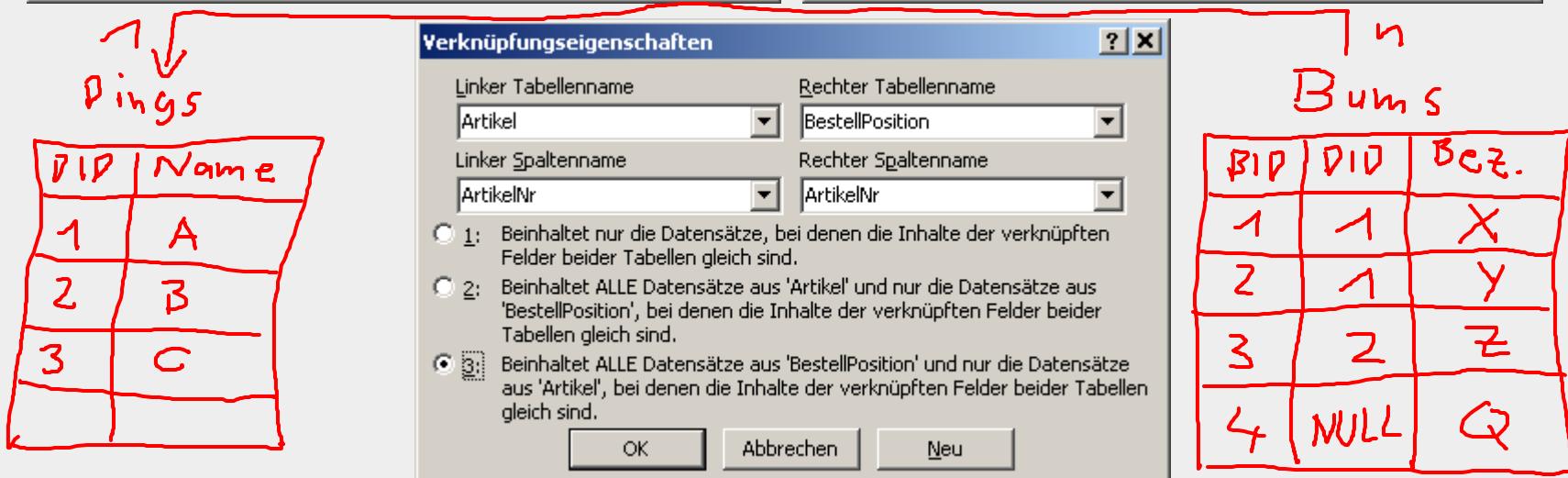
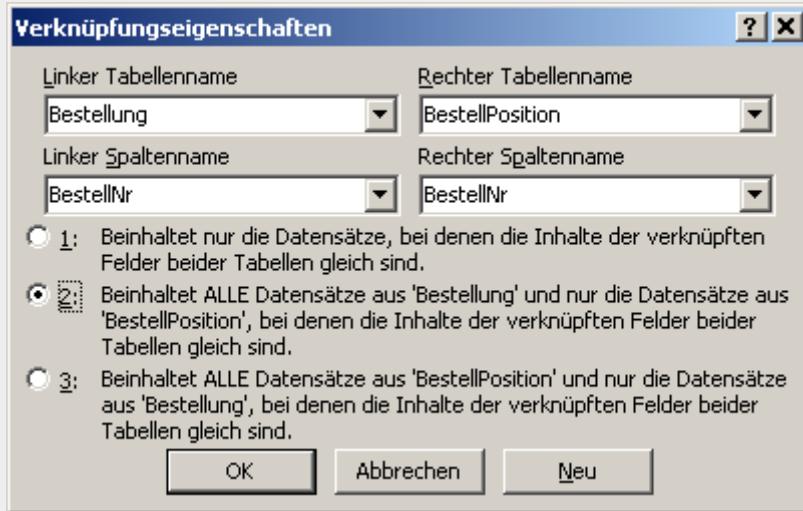
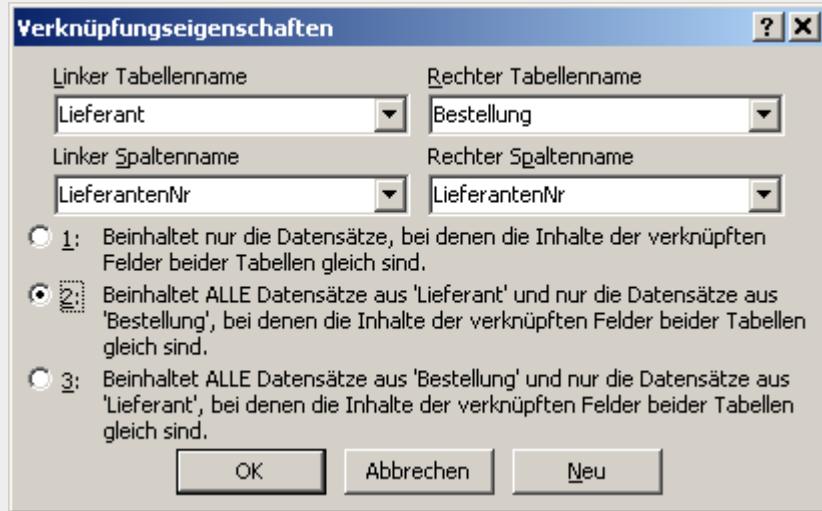
## Eine Auswertung

- Welches mengen- und wertmäßige Bestellvolumen liegt pro Lieferant vor?

# Abfrageentwurf



# Left and Right Outer Joins



## Abfrage in SQL

```
SELECT
```

```
    L.LieferantenNr,  
    L.Name,  
    Sum(P.Menge) AS GesamtMenge,  
    Sum(P.Menge * A.Preis) AS GesamtPreis
```

```
FROM Lieferant AS L
```

```
    LEFT OUTER JOIN (Bestellung AS B  
        LEFT OUTER JOIN (Artikel AS A  
            RIGHT OUTER JOIN BestellPosition AS P  
            ON A.ArtikelNr = P.ArtikelNr)  
            ON B.BestellNr = P.BestellNr)
```

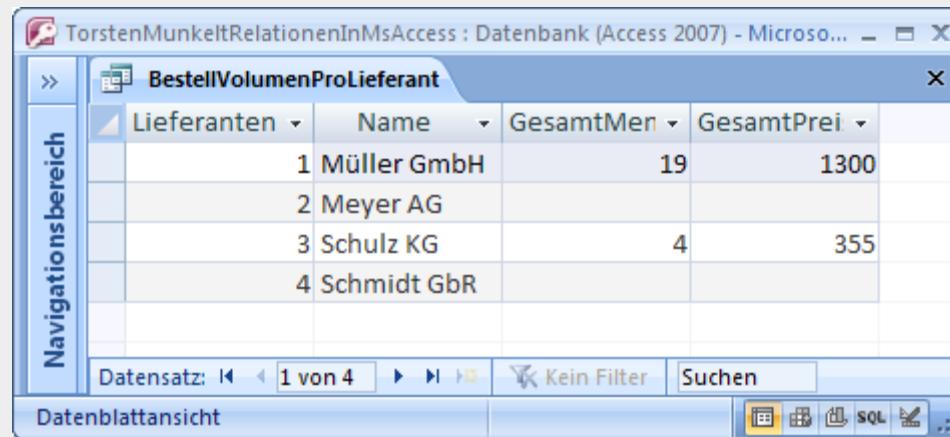
```
    ON L.LieferantenNr = B.LieferantenNr
```

```
GROUP BY
```

```
    L.LieferantenNr,  
    L.Name
```

```
;
```

# Abfrageergebnis in MS Access



The screenshot shows a Microsoft Access 2007 window titled "TorstenMunkeltRelationenInMsAccess : Datenbank (Access 2007) - Microsoft Access". The query results are displayed in a Datasheet view with the following columns: Lieferanten, Name, GesamtMen, and GesamtPrei. The data is as follows:

Lieferanten	Name	GesamtMen	GesamtPrei
1	Müller GmbH	19	1300
2	Meyer AG		
3	Schulz KG	4	355
4	Schmidt GbR		

# Zugriff auf MS-Access-DB mittels externem DB-Viewer

The screenshot shows the DbVisualizer Free 6.0.12 interface. On the left, the 'Connections' tree view lists several ODBC connections, with 'AccessDb' selected. The main window displays an SQL query in the 'SQL Commander' tab:

```
1 SELECT
2 L.LieferantenNr,
3 L.Name,
4 Sum(P.Menge) AS GesamtMenge,
5 Sum(P.Menge * A.Preis) AS GesamtPreis
6 FROM Lieferant AS L
7 LEFT OUTER JOIN (Bestellung AS B
8 LEFT OUTER JOIN (BestellPosition AS P
9 LEFT OUTER JOIN Artikel AS A
10 ON A.ArtikelNr = P.ArtikelNr)
11 ON P.BestellNr = B.BestellNr)
12 ON B.LieferantenNr = L.LieferantenNr
13 GROUP BY L.LieferantenNr, L.Name;
```

The results of this query are shown in a grid below:

	LieferantenNr	Name	GesamtMenge	GesamtPreis
1	1	Müller GmbH	19.0	1300.0
2	2	Meyer AG	(null)	(null)
3	3	Schulz KG	4.0	355.0
4	4	Schmidt GbR	(null)	(null)

## Zusammenfassung Exkurs DB-Anwendung in MS-Access

- Architektur von MS-Access-Anwendungen
- Relationstypen und Kardinalität
- Tabellen, Relationen und Abfragen
- Visualisierung von Relationen mittels Formularen und Unterformularen
- Komplexe Abfrage (mittels SQL)
- Vorteil: relativ schnell betriebliche SW-Anwendung gebaut ohne Programmierung und Programmierkenntnisse

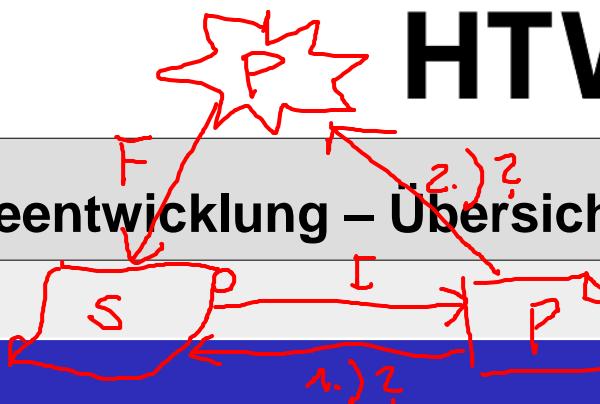
## Ausblick Exkurs DB-Anwendung in MS-Access

- MS-Access: Erweiterung des Anwendungsbeispiels „Bestellung“
- MS-Access: Berichte
- MS-Access: Makros und Module in VBA
- MS-Access: Trennung von Client-Anwendung und Datenbank
- MS-Access: Laufzeitumgebung

## Gliederung

1. Inhalte und Aufgaben der Wirtschaftsinformatik
2. Grundlagen der Informatik und der Informationstechnik
3. Informationsmanagement
4. Modellierung
5. Datenbanken
6. **Softwareentwicklung**
7. Betriebliche Informationssysteme

## Aktivitäten der Softwareentwicklung – Übersicht



Aktivität	Inhalt	Ergebnis
1.) „Planung“	Problemanalyse, Projektzielsetzung	Produktbeschreibung (Lastenheft)
2.) Anforderungsanalyse	Detaillierte Beschreibung der Anforderungen an die SW hinsichtlich Funktionalität und Leistungsumfang	Anforderungsspezifikation (Pflichtenheft)
3.) Entwurf	Grob- und Feinentwurf der SW-Architektur	Entwurfsspezifikation
4.) Implementierung	Umsetzung des Entwurfs	Software
5.) Test, Integration, Einführung	Test des Systems, Integration der implementierten Module zu einem Gesamtsystem	Dokumentation
6.) Wartung und Pflege	Fehlerkorrektur, Anpassung an veränderte Anforderungen	Neue Version

[Fink et al. 2001]

## Aktivitäten der SW-Entwicklung – 1.) Planung

- Problemanalyse → detaillierte Beschreibung des Problems!
- Ziele (für das SW-Entwicklungsprojekt) setzen
- Erheben von Nutzerwünschen und Anforderungen
- → Lastenheft mit wesentlichen funktionalen Anforderungen
  - Fachliche Sicht
  - DV-technische Sicht
- Bewerten der Umsetzbarkeit
  - Technisch
  - Wirtschaftlich
- Bewertungsmittel
  - Aufwandsschätzverfahren
  - Wirtschaftlichkeitsanalysen
- Grundlage für späteren **Test** der Software (Validierung)

## Aktivitäten der SW-Entwicklung – 2.) Anforderungsanalyse

- Ergebnis: verbindliche Anforderungen an das zukünftige Softwaresystem (Pflichtenheft):
  - **Prozesse**
  - Funktionen
  - Daten
  - Benutzeroberfläche
  - Schnittstellen
  - Dokumentation
- Grundlage für späteren **Test** der Software (Verifikation)

## Aktivitäten der SW-Entwicklung – 3.) Entwurf

- Ergebnis: Entwurfsspezifikation
- Grobentwurf (Komponentenarchitektur, ...) → Feinentwurf (Algorithmen und Datenstrukturen, ...)
- Fachlicher Entwurf (inhaltlich) → DV-technischer Entwurf (softwaretechnisch)
- Anwenden der Modellierungsmethoden aus dem vorangegangenen Kapitel
- Aspekte u. a.:
  - Erweiterbarkeit
  - Wiederverwendbarkeit
  - Effizienz
- Entwurf passender (Unit-)Tests?!

## Aktivitäten der SW-Entwicklung – 4.) Implementierung

- Programmierung: Umsetzen des Entwurfs in Quelltext
- Einsatz von CASE-Werkzeugen → 1:1-Umsetzung des Entwurfs
- Dokumentation des Quelltextes
- (Unit-)**Tests** schon während der Implementierung?!

## Aktivitäten der SW-Entwicklung – 5.) Test, Integration, Einführung

- **Test**, ob SW Spezifikation entspricht (Verifikation)
  - Analytisch ← Temporale Logik, Model Checking ← formale Spezifikation
  - Empirisch → kein Beweis!
- **Test** von Effizienz und Portabilität
- **Test**, ob SW das ursprüngliche Problem löst (Validierung)
  - Diskrepanz zwischen Anforderung und Spezifikation?
  - Diskrepanz zwischen Anforderungen damals und heute?
- Dokumentation und Schulungen für die Benutzer
- Migrationskonzept
  - Ersetzen von/Überführen aus Altsystemen
  - Eingliederung in Anwendungssystemumgebung

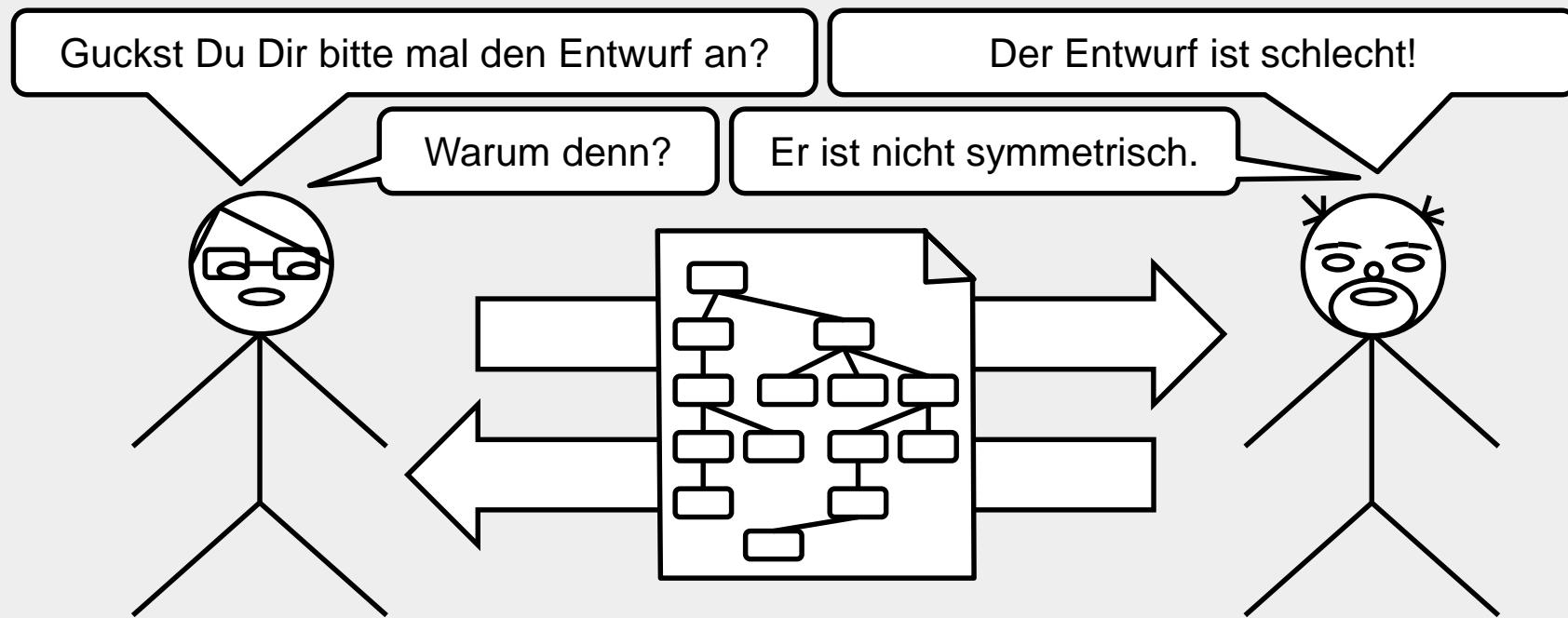
## Aktivitäten der SW-Entwicklung – 6.) Wartung, Pflege, Support

- Wartung: Fehlerkorrektur
- Pflege:
  - Anpassung an veränderte Anforderungen
  - Erweiterung, um zusätzliche Anforderungen zu erfüllen
- Ergebnis: neue Versionen, neue Releases
- Support: Unterstützung der Anwender bei auftretenden Problemen
- **Test** der neuen Version vor Inbetriebnahme
- Migration von einem Release auf ein folgendes

## Aktivitäten der SW-Entwicklung – Qualitätssicherung

- Querschnittsfunktion!
- Primäre Qualität:  
Funktionalität
- Sekundäre Qualität:
  - Erweiterbarkeit
  - Anpassbarkeit
  - Robustheit
  - Effizienz
  - Portabilität
  - Kompatibilität
  - Wiederverwendbarkeit
- Messen der Qualität  
eingeschränkt möglich →  
Werkzeuge
- Prinzipien der SW-QS [Balzer  
1998]:
  - Qualitätsmerkmale vorab  
bestimmen
  - Qualitätsmerkmale ständig  
messen und beurteilen
  - Viele Maßnahmen parallel  
anwenden, um Fehler a priori zu  
vermeiden
  - Fehlersuche (Test) während des  
gesamten  
Entwicklungsprozesses
  - Qualitätssicherung während des  
gesamten  
Entwicklungsprozesses
  - Programmierer ungleich Tester!

## Der Entwurf ist schlecht!



## Ravenous Bugblatter Beast of Traal

[Elisa Pavinato 2008]

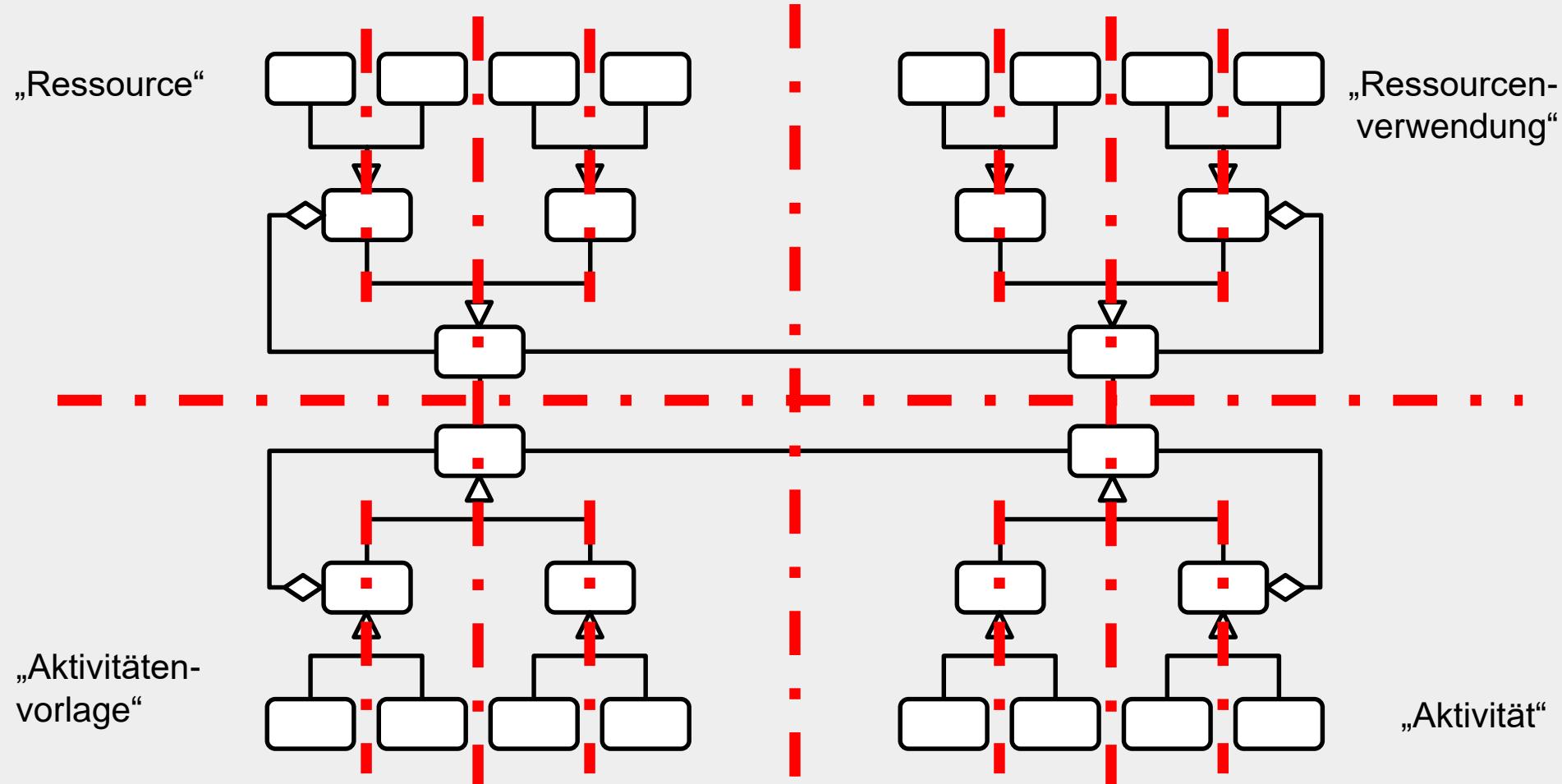


## „Schlussfolgerungen“ I

- Symmetrisch → schön → gut
- Asymmetrisch → hässlich → schlecht
- Asymmetrischer Entwurf ist schlecht!?
- Erfahrung/Eindruck: Symmetrie und Qualität eines Entwurfs oft konform
- Nach mehreren Überarbeitungszyklen Entwurf oft symmetrisch

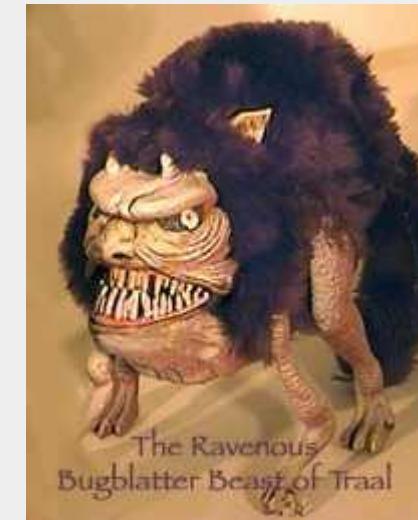


## Symmetrisches, schönes, gutes Design



## „Schlussfolgerungen“ II

- Schön → gut
- Gut → schön
- Guter Quelltext ist schön – und schöner ist gut.
- Konzentration auf schönen Quelltext, der dann automatisch gut ist



## Guter Quelltext ...

- Leicht zu modifizieren?
- **Leicht zu erweitern/fortzuschreiben!**
- **Einfach in Teilen wiederzuverwenden**
- Leicht zu testen
- Leicht zu warten
- **← gut strukturiert →**
- leicht zu lesen
- leicht zu verstehen
- leicht zu debuggen

## Effizienz

- Guter Quelltext nicht immer performantes Programm, aber:
- „Premature optimization is the root of all evil.“ [Donald E. Knuth]
- Programm nicht performant → Profiling → Optimieren
- Multiplikation hochdimensionaler Felder: inline Assembler oder Hochsprache?
- Hier keine Diskussion über Effizienz

## “Open-Closed Principle” (OCP) [Bertrand Meyer 1988]

- Sinngemäß: Software-Struktureinheiten (Komponenten, Module, Klassen, Methoden, Routinen, ...) offen für Erweiterungen, aber gesperrt für Veränderungen
- **Open** for Extension:  
Erweitern der Struktureinheiten (um neue) erlaubt
- **Closed** for Modification:  
Verändern der Struktureinheiten verboten
- Potenziell erwünschte Veränderungen des Verhaltens der Struktureinheit → Erweiterung
- Vorteile:
  - Unabhängige Struktureinheiten
  - Separat wiederverwendbar
  - Änderung (Erweiterung!) nur an einer Stelle
  - Robuster Entwurf im Hinblick auf Änderungen der Anforderungen

## Einfaches Beispiel für OCP (I)

```
public final class Dompteur {
    public void treibeAn(List<Tier> tiere) {
        for(Tier tier:tiere) tier.bewegeDich();
    }
}

public abstract class Tier {
    public abstract void bewegeDich();
}

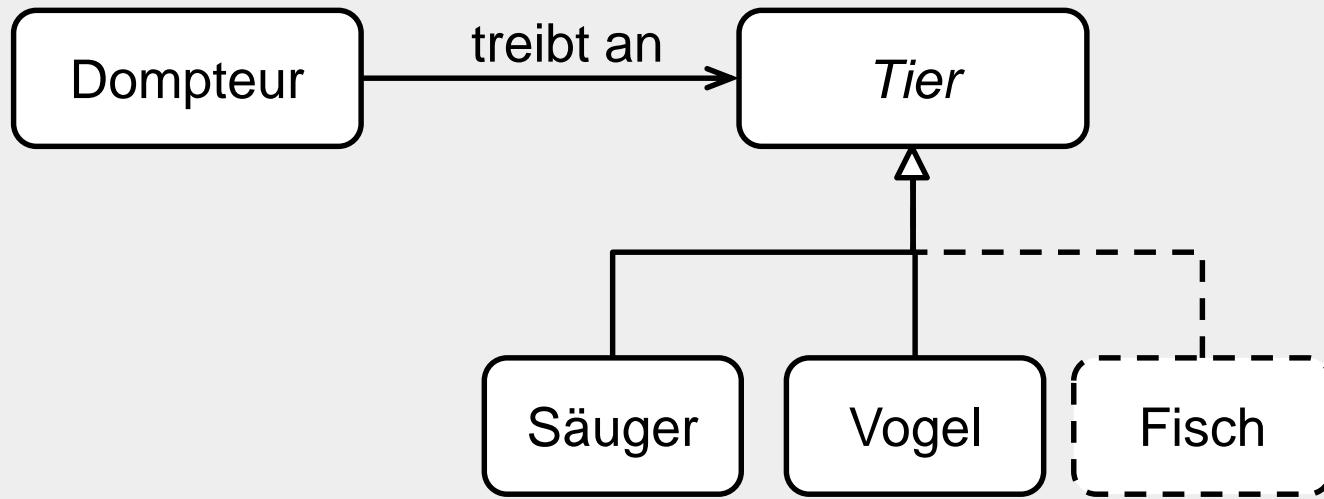
public class Saeuger extends Tier {
    @Override
    public void bewegeDich() {
        System.out.println("Ich laufe.");
    }
}

public class Vogel extends Tier {
    @Override
    public void bewegeDich() {
        System.out.println("Ich fliege.");
    }
}
```

## Einfaches Beispiel für OCP (II)

```
public class Fisch extends Tier {  
    @Override  
    public void bewegeDich() {  
        System.out.println("Ich schwimme.");  
    }  
}
```

## Einfaches Beispiel für OCP (grafisch)



Beim Hinzufügen einer neuen Tierklasse müssen weder das Tier noch der Dompteur modifiziert werden.

# Externe Kopplung und interne Kohäsion

- Externe Kopplung:
  - Abhängigkeiten und Interaktionen zwischen Struktureinheiten
  - Statisch, durch Importschnittstelle festgelegt
  - Dynamisch, durch Methodenaufrufe (beziehungsweise durch Zugriff auf Attribute) beeinflusst
  - Ziel: **minimale Kopplung**
- Interne Kohäsion:
  - Zusammenhalt einer Struktureinheit
  - Hohe Kohäsion: Struktureinheit logische Einheit aus Funktionalität und Daten
  - Ziel: **maximale Kohäsion**
- Vorteile minimaler Kopplung und maximaler Kohäsion:
  - Änderungen/Erweiterungen nur lokal, nicht struktureinheitenübergreifend
  - Wiederverwendbarkeit separater Struktureinheiten

## Minimale externe Kopplung versus maximale interne Kohäsion

- **Minimale externe Kopplung:**  
alle Daten und gesamte Funktionalität in einer einzigen Struktureinheit  
→ zwar minimale (keine) Kopplung zwischen Struktureinheiten, aber sehr **geringe Kohäsion**
- **Maximale interne Kohäsion:**  
jedes „Datum“ und kleinste Stücke Funktionalität in eigene, winzige Struktureinheiten  
→ zwar maximale Kohäsion innerhalb der Struktureinheiten, aber sehr **starke Kopplung**

## Reduzieren der externen Kopplung

- Information Hiding [David L. Parnas 1972]
- Einfache, kleine und stabile Schnittstellen zwischen Struktureinheiten
- „Indirektion“ über:
  - abstrakte Klassen
  - (Java-)Interfaces
  - Prozedural: Funktionszeiger und Ereignisbehandlungs Routinen
- Kommunikation über Nachrichten(Objekte)

## Starke externe Kopplung (Beispiel)

```
import coupling.high.funktionalitaet.Funktionalitaet;
public class OberFlaeche {
    public void zeigeFortschritt(int i) {
        System.out.println(i);
    }
    public void starteFunktionalitaet()
        throws InterruptedException {
        new Funktionalitaet().arbeite(this);
    }
}
```

```
import coupling.high.oberflaeche.OberFlaeche;
public class Funktionalitaet {
    public void arbeite(OberFlaeche of)
        throws InterruptedException {
        for (int i = 0; i < 10; ++i) {
            of.zeigeFortschritt(i);
        }
    }
}
```

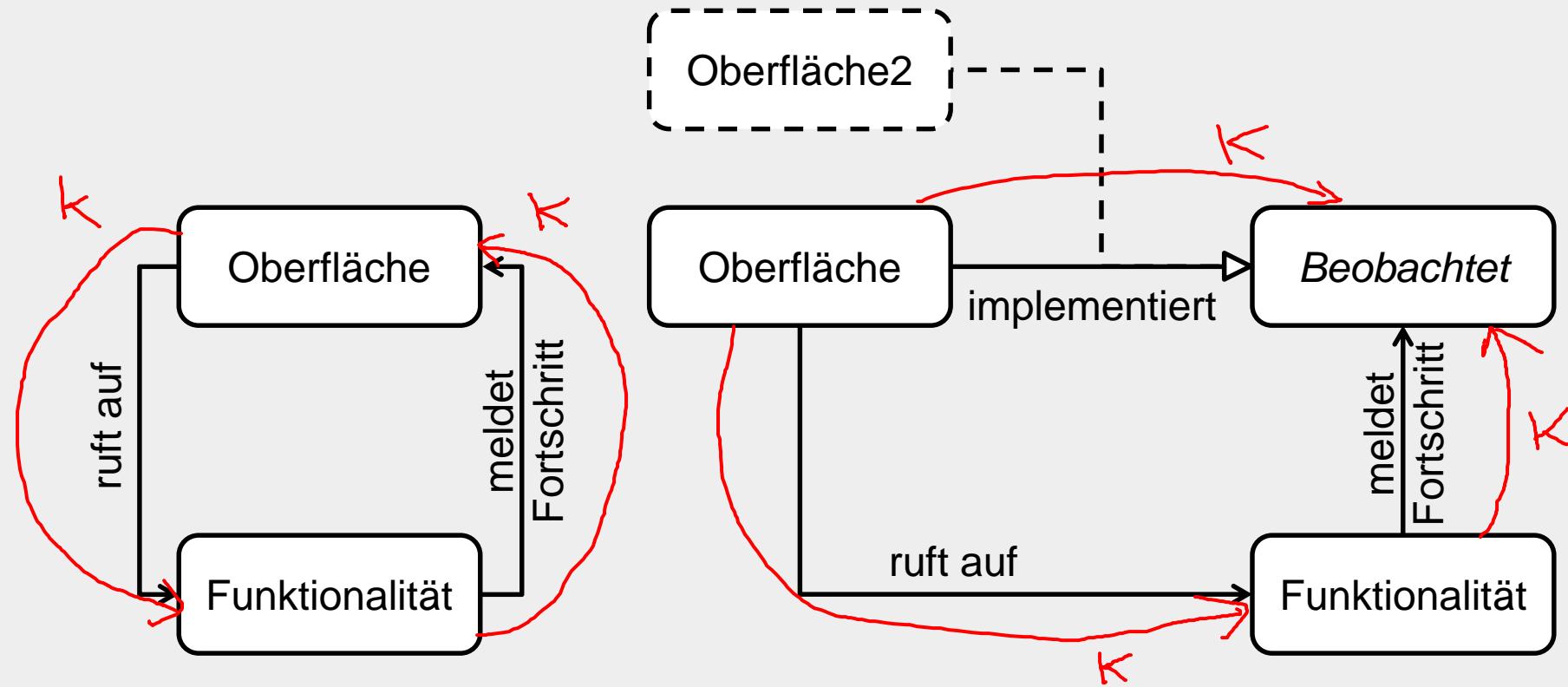
```
import coupling.low.funktionalitaet.Beobachtet;
import coupling.low.funktionalitaet.Funktionalitaet;
public class OberFlaeche implements Beobachtet {
    public void starteFunktionalitaet() throws InterruptedException {
        new Funktionalitaet().arbeite(this);
    }
    @Override
    public void benachrichtige(int fortSchritt) {
        System.out.println(fortSchritt);
    }
}

public interface Beobachtet {
    public void benachrichtige(int fortSchritt);
}

public class Funktionalitaet {
    public void arbeite(Beobachtet beobachter)
        throws InterruptedException {
        for (int i = 0; i < 10; ++i) {
            Thread.sleep(500);
            beobachter.benachrichtige(i);
        }
    }
}
```

## Schwache externe Kopplung (Beispiel)

## Externe Kopplung (Beispiele, grafisch)



Starke externe Kopplung

Schwache externe Kopplung

## Verstärken der internen Kohäsion

- Enge Beziehung zwischen (Instanz)variablen und die Routinen (Methoden) innerhalb einer Struktureinheit (Klasse) sicherstellen
- Indikatoren für zu geringe Kohäsion:
  - Gruppen von Routinen (Methoden), die nur auf bestimmten Gruppen von Daten arbeiten (innerhalb einer Struktureinheit)
  - Viele Parameter bei Routinen (Methoden)
  - Viele (Instanz)variablen und Routinen (Methoden) in einer Struktureinheit (Klasse)
  - Lange/große Routinen (Methoden, Klassen)
- Maßnahmen zum Verstärken der Kohäsion:
  - Struktureinheit geringer Kohäsion in mehrere Struktureinheiten aufteilen
  - A priori konsistente, treffende, ausdrucksstarke Bezeichnungen für Struktureinheiten (Klassen), Routinen (Methoden) und (Instanzvariablen) finden

## Geringe interne Kohäsion (Beispiel)

```
public class ZahlenTexte {
    private int zahl1 = 7;
    private int zahl2 = 3;
    private String text1 = "Schnick";
    private String text2 = "Schnack";
    public int summe() {
        return zahl1 + zahl2;
    }
    public int differenz() {
        return zahl1 - zahl2;
    }
    public String verkette() {
        return text1 + text2;
    }
    public String verketteUmgekehrt() {
        return text2 + text1;
    }
}
```

## Hohe interne Kohäsion (Beispiel)

```
public class Zahlen {  
    private int zahl1 = 7;  
    private int zahl2 = 3;  
    public int summe() {  
        return zahl1 + zahl2;  
    }  
    public int differenz() {  
        return zahl1 - zahl2;  
    }  
}
```

```
public class Texte {  
    private String text1 = "Schnick";  
    private String text2 = "Schnack";  
    public String verkette() {  
        return text1 + text2;  
    }  
    public String verketteUmgekehrt() {  
        return text2 + text1;  
    }  
}
```

# Zyklen automatisch finden und ausmerzen (Sonargraph, ©Hello2Tomorrow )

Cycle Analysis – Listing cycle groups with and without warnings

Cycle Groups Element

My Project (4 cycle group warnings)

- 4 directory cycle groups
- 4 physical build unit package cycle groups (4 cyc)
  - 14 elements (cycle group warning)
  - 6 elements (cycle group warning)
  - 2 elements (cycle group warning)
  - 2 elements (cycle group warning)
- 7 source file cycle groups

Cyclicity 196, 14 cyclic nodes in 1 cycle group

Always break cyclic dependencies with vi...  
Arcs Deps Refs CYC CycNodes Grps

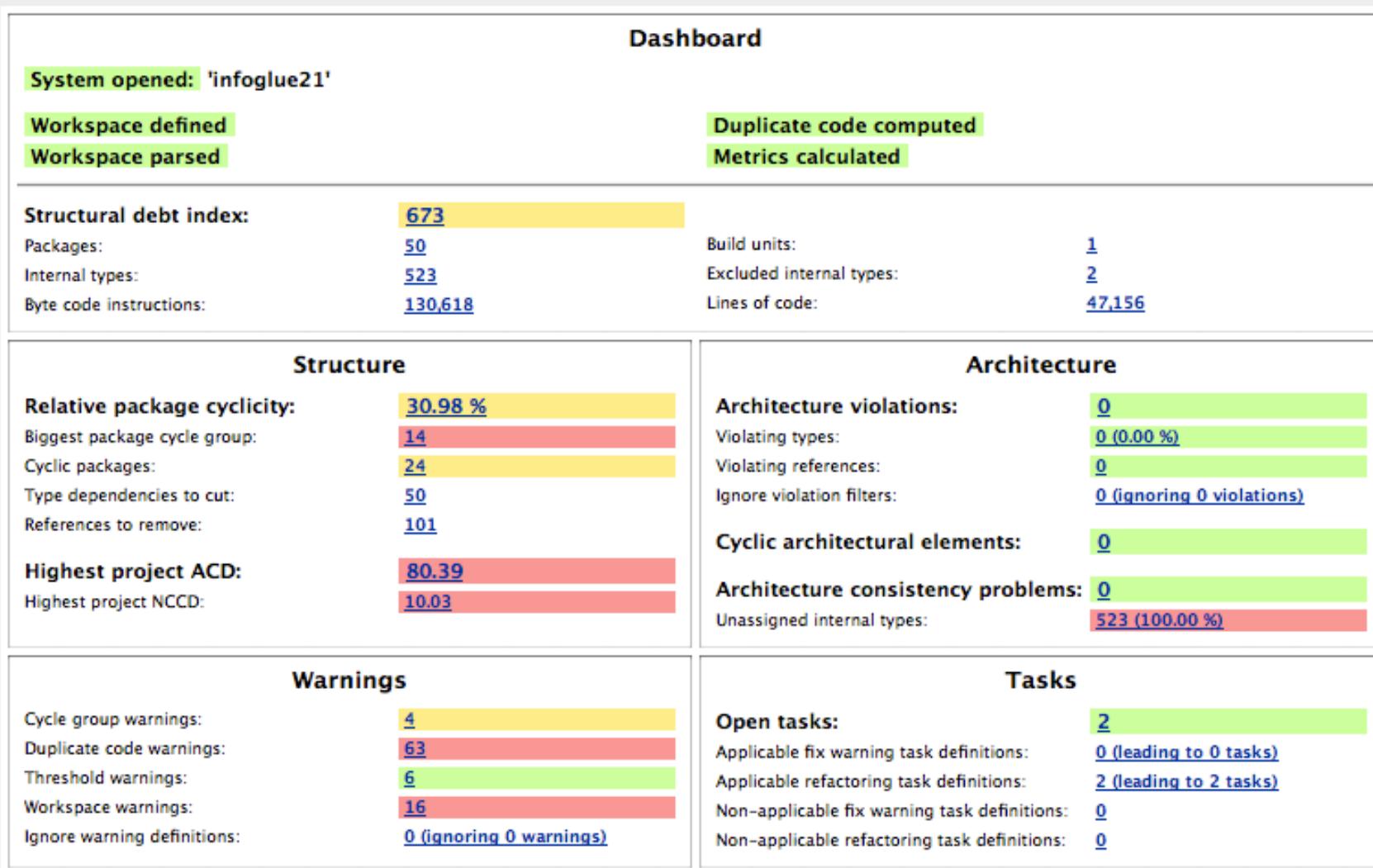
9	27	55	0	0	0
9	27	55	0	0	0
9	27	58	0	0	0

Found 1697 cycle break-up sets (pausing)  
Pause automatically after one minute

Dependency Graph (hiding non-cut non-cyclic dependencies, based on 14-element physical build unit package cycle group)

Cyclicity 196, 14 cyclic nodes in 1 cycle group (cutting marked dependencies: cyclicity 4, 2 cyclic nodes in 1 cycle group)

# Übersicht über Softwaregüte (Sonargraph, ©Hello2Tomorrow )



# Gleichen Quelltext nur einmal schreiben



Falls es doch passiert ist:

- Fragment in Methode auslagern
- Methode aussagekräftig benennen
- Unterschiede als Parameter übergeben
- Richtige Klasse für Methode finden
- Ggf. Methode in Superklasse ansiedeln
- Ggf. Methode in neue Klasse auslagern
- Ggf. „Schablonenmethode“ anwenden

FOR EACH M\_Aktivitaet

WHERE M\_Aktivitaet.Firma = '100'

EXCLUSIVE-LOCK

TRANSACTION

:

IF M\_Aktivitaet.MehrMaschFaktor = 0 THEN DO:

ASSIGN M\_Aktivitaet.MehrMaschFaktor = 100.

END.

END.

FOR EACH MA\_Aktivitaet

WHERE MA\_Aktivitaet.Firma = '100'

AND MA\_Aktivitaet.ProzessBereich = 'PPA'

EXCLUSIVE-LOCK

TRANSACTION

:

IF MA\_Aktivitaet.MehrMaschFaktor = 0 THEN DO:

ASSIGN MA\_Aktivitaet.MehrMaschFaktor = 100.

END.

END.

FOR EACH MB\_Aktivitaet

WHERE MB\_Aktivitaet.Firma = '100'

AND MB\_Aktivitaet.ProzessBereich = 'PPA'

AND (MB\_Aktivitaet.AuftragsStatus <> 'R'

OR MB\_Aktivitaet.fertig = FALSE)

EXCLUSIVE-LOCK

TRANSACTION

:

IF MB\_Aktivitaet.MehrMaschFaktor = 0 THEN DO:

ASSIGN MB\_Aktivitaet.MehrMaschFaktor = 100.

END.

END.

## „In der Kürze liegt die Würze“, VT100- oder 24\*80-Regel

- Kleine Pakete schnüren
- Klassen klein halten
- Kurze Methoden schreiben
- Vorteile:
  - Starke interne Kohäsion
  - Modifikation/Erweiterung nur an einer Stelle
  - Struktureinheiten separat verwendbar
  - Weniger Parameter/Instanzvariablen
  - Geringere Abhängigkeiten
  - Einfacher zu verstehen
  - Einfacher zu testen
  - Kaum Kommentare nötig

## „Objekt-Gymnastik“ [Jeff Bay 2008]

- 1. nur eine Ebene tief einrücken pro Methode
- ...
- 3. alle Zeichenketten und primitiven Datentypen als Klassen
- 4. nur “ein Punkt” pro Zeile
- 5. keine Namen abkürzen
- 6. kleine Struktureinheiten:
  - Nicht mehr als 50 Zeilen pro Klasse
  - Nicht mehr als 10 Klassen pro Paket
- 7. nicht mehr als zwei Instanzvariablen pro Klasse
- ...

```
static int dopcor(
    unsigned n,
    FcnEqDiff fcn,
    double x,
    double* y,
    double xend,
    double hmax,
    double h,
    double* rtoler,
    double* atol,
    int itoler,
    FILE* fileout,
    SoITrait solout,
    int iout,
    long nmax,
    double uround,
    int meth,
    long nstiff,
    double safe,
    double beta,
    double fac1,
    double fac2,
    unsigned* icont,
    double* duser,
    int* iuser)
```

## Wenige Parameter

Vorteile weniger Parameter:

- Starke interne Kohäsion
- Geringe externe Kopplung
- Kürzere Methode/Routine
- Methode/Routine an richtiger Stelle
- Einfache Methode/Routine
- Methode/Routine leicht zu verstehen
- Methode/Routine leicht zu testen

Falls doch zu viele Parameter:

- Parameterwerte errechnen
- Instanz anstelle ihrer Eigenschaften übergeben
- Parameterklasse einführen und verwenden
- Methode/Routine aufspalten → weniger Parameter, da nicht alle für gesamte Routine relevant
- Methode/Routine an andere Stelle verlagern, wo viele Parameter bereits bekannt

## Instanz vs. Eigenschaften; Parameterobjekt

```
public class PreisBerechner {  
    public double berechnePreis(  
        int breite,  
        int hoehe,  
        int tiefe,  
        Material material,  
        PreisListe preisListe,  
        Kunde kunde) {  
        double result = 0.0d;  
        /* Preis berechnen */  
        return result;}  
    public double berechnePreis(Artikel artikel, Kunde kunde) {  
        double result = 0.0d;  
        /* Preis berechnen */  
        return result;}  
    public double berechnePreis(Kontext kontext) {  
        double result = 0.0d;  
        /* Preis berechnen */  
        return result;}  
}
```

# Methode zur richtigen Klasse

```
public class Artikel {  
    public double berechnePreis(Kunde kunde) {  
        double result = 0.0d;  
        /* Preis berechnen */  
        return result;  
    }  
}
```

## Instanzen bestimmen selbst über ihr Verhalten

- Keine externe Unterscheidung, wie im Falle einer Instanz einer bestimmten Klasse zu verfahren ist, ...
- ..., sondern Aufruf der Methode (des Verhaltens) einer abgeleiteten Klasse A auf einer Variable, welche eine Instanz von A hält, aber vom Typ ihrer abstrakten Superklasse ist
- Vorteile:
  - Verhaltensänderung durch Erweiterung und ohne Modifikation möglich (OCP)
  - Verhaltensänderung nur an einer Stelle (nicht in vielen Fallunterscheidungen, die oft gemeinsam auftreten)

# Instanzen bestimmen selbst über ihr Verhalten, schlechtes Beispiel

```
Gericht[] gerichte = {new Fleisch(), new Suppe(), new Fisch()};
for (Gericht gericht:gerichte) {
    if (gericht instanceof Fleisch) {
        System.out.println("Messer und Gabel");
    } else if (gericht instanceof Suppe) {
        System.out.println("Löffel");
    } else if (gericht instanceof Fisch) {
        System.out.println("Fischmesser und Gabel");
    } else {
        throw new RuntimeException("Unbekanntes Gericht");
    }
}
```

Immer, wenn man ein neues Gericht einführt,  
muss man auch diese Fallunterscheidung ändern. ☺

## Instanzen bestimmen selbst über ihr Verhalten, gutes Beispiel

```
final Gericht[] gerichte = {new Fleisch(), new Suppe(), new Fisch()};
for (Gericht gericht:gerichte) {
    System.out.println(gericht.nenneBesteck());
}
```

```
public abstract class Gericht {
    public abstract String nenneBesteck();
}
```

```
public class Fleisch extends Gericht {
    @Override
    public String nenneBesteck() {
        return "Messer und Gabel";
    }
}
```

## Keine Verträge brechen

- in Subklassen Verträge einhalten, die in Superklassen geschlossen worden sind
- Erlaubt: Subklassen um Methoden und Attribute (Getter und Setter) zu erweitern
- Verboten: In Subklassen öffentliche Methoden und Attribute (Getter und Setter) der Superklassen zu verbergen
- Verboten: in übersteuernden Methoden ausschließlich Ausnahmen zu werfen oder Fehler zu melden
- Vorteil: Nutzende Klassen können sich darauf verlassen, dass eine Instanz funktioniert, auch wenn sie mittlerweile die Instanz einer abgeleiteten Klasse oder eine Geschwisterklasse ist.

## Keine Verträge brechen, gutes und schlechtes Beispiel (I)

```
public class Super {  
    public void macheEtwas() {  
        System.out.println("Ich mache etwas.");  
    }  
}
```

```
public class GuteSub extends Super {  
    @Override  
    public void macheEtwas() {  
        super.macheEtwas();  
        System.out.println("Ich mache noch etwas mehr.");  
    }  
}
```

```
public class SchlechteSub extends Super {  
    @Override  
    public void macheEtwas() {  
        throw new RuntimeException("Mit der Klasse \\""  
            + this.getClass().getName()  
            + "\" geht das nicht.");  
    }  
}
```

## Keine Verträge brechen, gutes und schlechtes Beispiel (II)

```
Super[] supers = {new Super(), new GuteSub(), new SchlechteSub()};  
for (Super s:supers) {  
    s.macheEtwas();  
}
```

Ich mache etwas.

Ich mache etwas.

Ich mache noch etwas mehr.

Exception in thread "main" java.lang.RuntimeException:

Mit der Klasse „SchlechteSub“ geht das nicht.

at SchlechteSub.macheEtwas(SchlechteSub.java:5)

at Main.main(Main.java:8)

## Vorsicht bei Methodenverkettung

- Mehrere miteinander verkettete Methoden:
  - → Indiz: Methode/Daten an falscher Stelle
  - → Abhängigkeiten zwischen vielen Objekten, die vielleicht gar nicht voneinander abhängen sollten
- Lösungsmöglichkeiten:
  - Methode/Daten an andere Klasse hängen
  - Methodenverkettung verbergen
- Vorteile einstufiger Methodenverkettung:
  - Geringere Kopplung
  - Austauschbarkeit separater Struktureinheiten
  - Wiederverwendbarkeit separater Struktureinheiten

## Methodenverkettung, Beispiele

Zehnfachverkettung; echter, produktiver Quelltext:

```
System.out.println(SArtikelArts.getInstance().getArtikelArts()
    .values().iterator().next().getArtikelArtSprs()
    .values().iterator().next().getBezeichnung());
```

Klassen und Methoden einführen, die HashMap und Iterator verstecken

→ Reduziert auf Sechsfachverkettung:

```
System.out.println(SArtikelArts.getInstance().getArtikelArts()
    .getFirst().getArtikelArtSprs()
    .getFirst().getBezeichnung());
```

Methode einführen, die erste Artikelart liefert, und Methode parametrisieren, so dass die Bezeichnung in einer bestimmten Sprache zurückgegeben wird

→ Reduziert auf Dreifachverkettung:

```
System.out.println(SArtikelArts.getInstance().getFirstArtikelArt()
    .getBezeichnung("D"));
```

→ Innenleben der Klassen besser versteckt (Information Hiding)

## Falls doch eine lange Methode vorliegt, ... (vorher)

```
public class Plan {  
    public double errechneRessourcenBedarf() {  
        double result = 0.0d;  
        /* initialisiere die Planschritte: */  
        /* 30 Zeilen Quelltext */  
        /* durchlaufe den Plan: */  
        /* 43 Zeilen Quelltext */  
        /* aggregiere den Ressourcenbedarf: */  
        /* 27 Zeilen Quelltext */  
        return result;  
    } /* ==> ueber 100 Zeilen Quelltext in Methode */  
}
```

## Falls doch eine lange Methode vorliegt, ... (nachher)

```
public class Plan {  
    public double errechneRessourcenBedarf() {  
        initialisierePlanschritte();  
        durchlaufePlan();  
        return aggregiereRessourcenBedarf();  
    }  
    private double aggregiereRessourcenBedarf() {  
        double result = 0.0d;  
        /* 27 Zeilen Quelltext */  
        return result;  
    }  
    private void durchlaufePlan() {  
        /* 43 Zeilen Quelltext */  
    }  
    private void initialisierePlanschritte() {  
        /* 30 Zeilen Quelltext */  
    }  
}
```

## Namenskonventionen

- Keine umgekehrte Notation; bei kleinem Gültigkeitsbereich nicht nötig
- Keine ungarische Notation; Compiler kennt den Typ
- Länge des Namens proportional zum Gültigkeitsbereich:
  - Lokale Variablen und private Methoden kurze Namen
  - Instanzvariablen längere Namen
  - Öffentliche Klassen und Methoden lange Namen
- Zu lange Bezeichner → zu große Struktureinheiten → aufbrechen!
- Variablen, Parameter und Klassen: Substantive; was sie sind, z. B.: „Graph“, „Knoten“, „Liste“, „Menge“, „WarteSchlange“, ...
- Methoden: Verben; was sie machen, z. B.: „gib...“, „berechne...“, „setze...“, „durchlaufe...“
- Schnittstellen: Fähigkeit ausdrücken, z. B.: „KannLesen“, „KannFahren“, „IstBehaelter“, „HatFluegel“, ...
- ...

## Sparsam kommentieren

- Guter/schöner Quelltext sollte selbsterklärend sein, so dass man keine/nur wenige Kommentare benötigt.
- Ehe man ein Quelltextfragment kommentiert, kann man es auch als Methode auslagern, die einen aussagekräftigen Namen aufweist.
- Ehe man eine Methode kommentiert, kann man sie auch aussagekräftig benennen oder aufspalten.
- Viel Kommentar muss immer auch mit geändert werden, wenn man den Quelltext ändert.  
→ Aufwand und Inkonsistenzen
- Erklären, **was** es macht (und vielleicht **warum**), aber **nicht**, **wie** es das macht

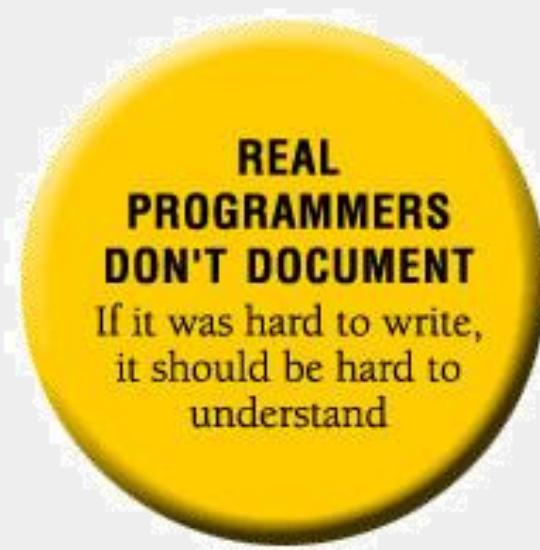
## Einrücken und Klammern

- Tiefes Einrücken für lange Methoden; keine langen Methoden → kein tiefes Einrücken nötig
- Egal, wie weit einrücken, aber nicht mehr als zwei Ebenen; falls mehr nötig → Quelltext auslagern
- {}: egal, ob auf neue Zeile oder nicht, aber einheitlich
- (): im Zweifelsfall immer, auch wenn Operatorrangfolge klar; muss auch verstanden werden

## Weitere Hinweise

- Auskommentierten oder nicht verwendeten Quelltext löschen
- Klassen bzw. Instanzen anstelle primitiver Datentypen verwenden
- Daten, die immer gemeinsam auftreten, zu Klasse kombinieren
- Klasse nur dann mit Instanzvariable versehen, wenn Zustand der Variable während gesamten Lebenszyklus‘ jeder Instanz der Klasse relevant ist
- Klassen nur dann anlegen, wenn sie Funktionalität **und** Daten aufweisen
- Globale Variablen vermeiden
- Nur benannte numerische Konstanten verwenden; niemals 86400; 0.628?
- ...

„If it was hard to write, it should be hard to read ...“



“... and even harder to modify.”

## How to Write Unmaintainable Code (I)

<http://mindprod.com/jgloss/unmain.html>

- “CapiTaliSaTion: Randomly capitalize the first letter of a syllable in the middle of a word.”
- “Åccented Letters: Use accented characters on variable names.”  
î, ï, í, Í
- “Use foreign language dictionaries as a source for variable names.”
- “marypoppins= (superman+starship) /god; This confuses the reader because they have difficulty disassociating the emotional connotations of the words from the logic they’re trying to think about.”
- “isValid(x) should as a side effect convert x to binary and store the result in a database.”
- “Hungarian Notation is the tactical nuclear weapon of source code obfuscation techniques; use it!”
- “Use constant names like LancelotsFavouriteColour instead of blue and assign it hex value of 0x0204FB [similar to blue only].”
- swimmer == swimmer == swimrnner?

## How to Write Unmaintainable Code (II)

<http://mindprod.com/jgloss/unmain.html>

- `008 == 008? parseInt == parseInt? DOCalc == DOCalc?`
- “Use Plural Forms From Other Languages: [...] Esperanto, Klingon and Hobbitese qualify as languages for these purposes. For pseudo-Esperanto pluraloj, add oj. You will be doing your part toward world peace.”
- “If God didn’t want us to use global variables, he wouldn’t have invented them.”
- “If you have a class with 10 properties in it, consider a base class with only one property and subclassing it 9 levels deep so that each descendant adds one property. [...] Make sure you create at least one instance of each subclass.“
- “Lie in the comments.” “Document the obvious.” “Document How Not Why.”
- “Monty Python Comments: On a method called `makeSnafucated insert` only the Javadoc `/*make snafucated*/`. Never define what snafucated means anywhere.”

## How to Write Unmaintainable Code (III)

<http://mindprod.com/jgloss/unmain.html>

- “LISP is a dream language for the writer of unmaintainable code.”
- ```
#define ONE 1
#define TWO 2
#define THREE 3
```
- “Use Assembler and Regular Expressions.”
- “Testing is for cowards. A brave coder will bypass testing.”
- “Ambiguity: Use the fuzziest, vaguest most general terminology you can come up with, especially for variable names. handle is a great example — a handle to what? processData is a great method name.”
- ```
byte[] rowVector, colVector, matrix[];
```
- „In the interests of efficiency, avoid encapsulation.”
- “No Secrets: Declare every method and variable public. After all, somebody, sometime might want to use it.”
- “Use octal.”

## Gewonnene Erkenntnisse

- Open-Closed Principle folgen
- Minimale externe Kopplung **und** maximale interne Kohäsion anstreben
- Aufgeführte Hinweise möglichst beherzigen
- → gute Programme

## Weiterführende Themen

- Programmierregeln und –gesetze
- Coding Conventions
- Style Guides
- Style Checker
- Software-Metriken
- Code Smells
- Refactoring
- Design-Patterns
- Anti-Patterns
- Cross-Cutting-Concerns (CCC)
- Inversion of Control (IoC)
- Dependency-Injection (DI)

## Weiterführende Literatur

---

- Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides: **Design Patterns: Elements of Reusable Object-Oriented Software**, Addison-Wesley Longman, Amsterdam, 1994.
- William J. Brown, Raphael C. Malveau, Thomas J. Mowbray: **AntiPatterns: Refactoring Software, Architectures, and Projects in Crisis**, John Wiley & Sons, 1998.
- Martin Fowler, Kent Beck, John Brant, William Opdyke, Don Roberts: **Refactoring: Improving the Design of Existing Code**, Addison-Wesley Longman, Amsterdam, 1999.

## Gliederung

1. Inhalte und Aufgaben der Wirtschaftsinformatik
2. Grundlagen der Informatik und der Informationstechnik
3. Informationsmanagement
4. Modellierung
5. Datenbanken
6. Softwareentwicklung
7. **Betriebliche Informationssysteme**