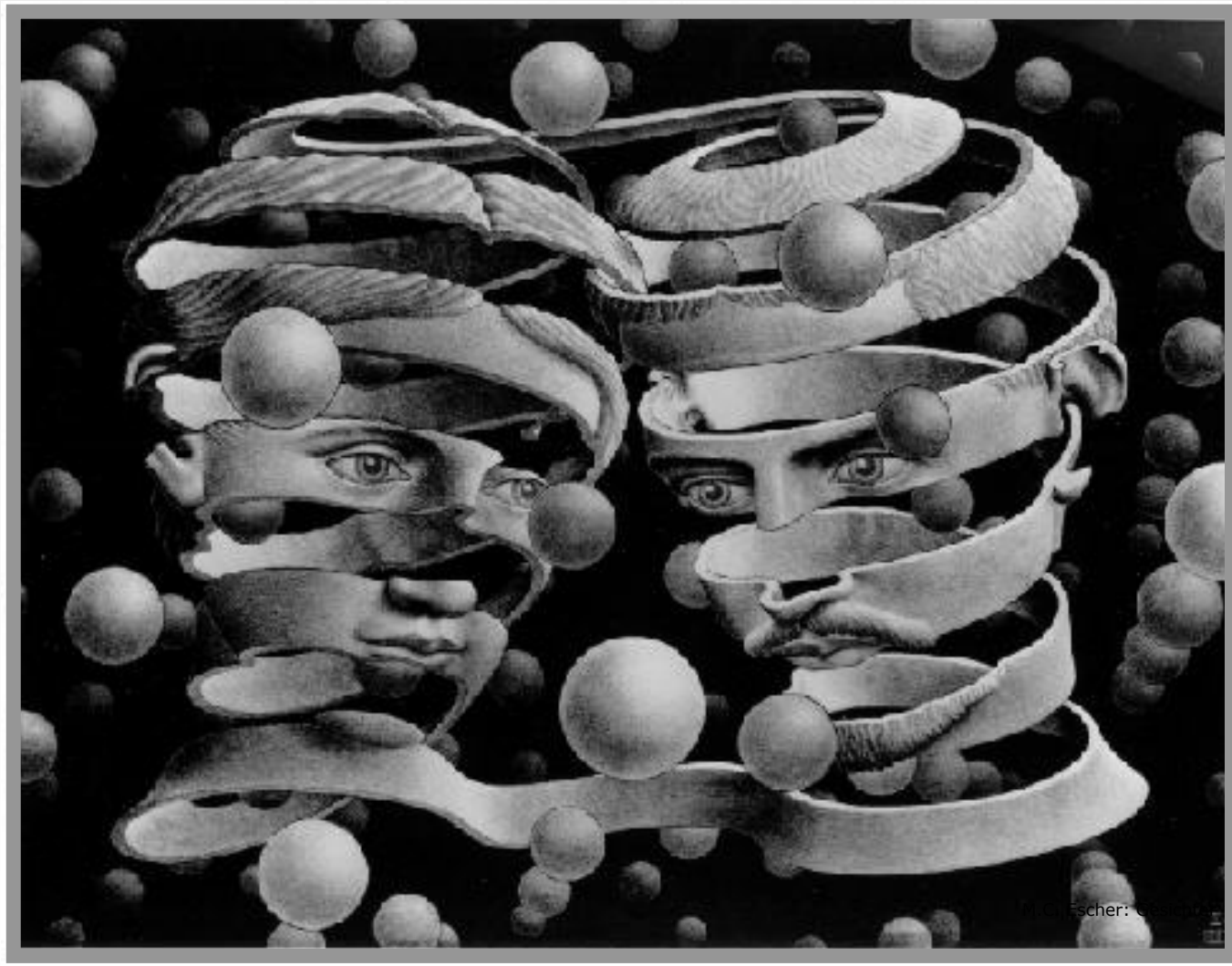
h_da

HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

Wissens-basierte Diagnostik



Kap.4: Wissensbasierte Systeme

4.0 Aufbau, Architektur, Eigenschaften
4.1 Wissensrepräsentation
4.2 Produktionsregeln & Regelverarbeitung

Dr. Norbert Waleschkowski
Semantis Information Builders GmbH
www.semantis-ib.de

Vorlesung Master-Studiengang
Wintersemester 2009/10

Kap. 4.0: Wissensbasierte Systeme

Aufbau, Architektur, Eigenschaften

Wissensbasierte Systeme

- Typische Anwendungsprogramme enthalten auch Wissen.
  Wann also spricht man von wissensbasierten Systemen?
  Und was zeichnet wissensbasierte Systeme aus?

- Wissensbasierte Systeme sind durch folgende Eigenschaften gekennzeichnet:
  - Wissen muss explizit (in Wissensbasen) erfasst sein und darf nicht nur im Programmcode stecken bzw. dort verborgen sein.
  - Die Wissensverarbeitung erfolgt durch eine Problemlösungskomponente, die das explizit vorhandene Wissen verarbeiten und auch eine Lösungsstrategie auswählen kann.

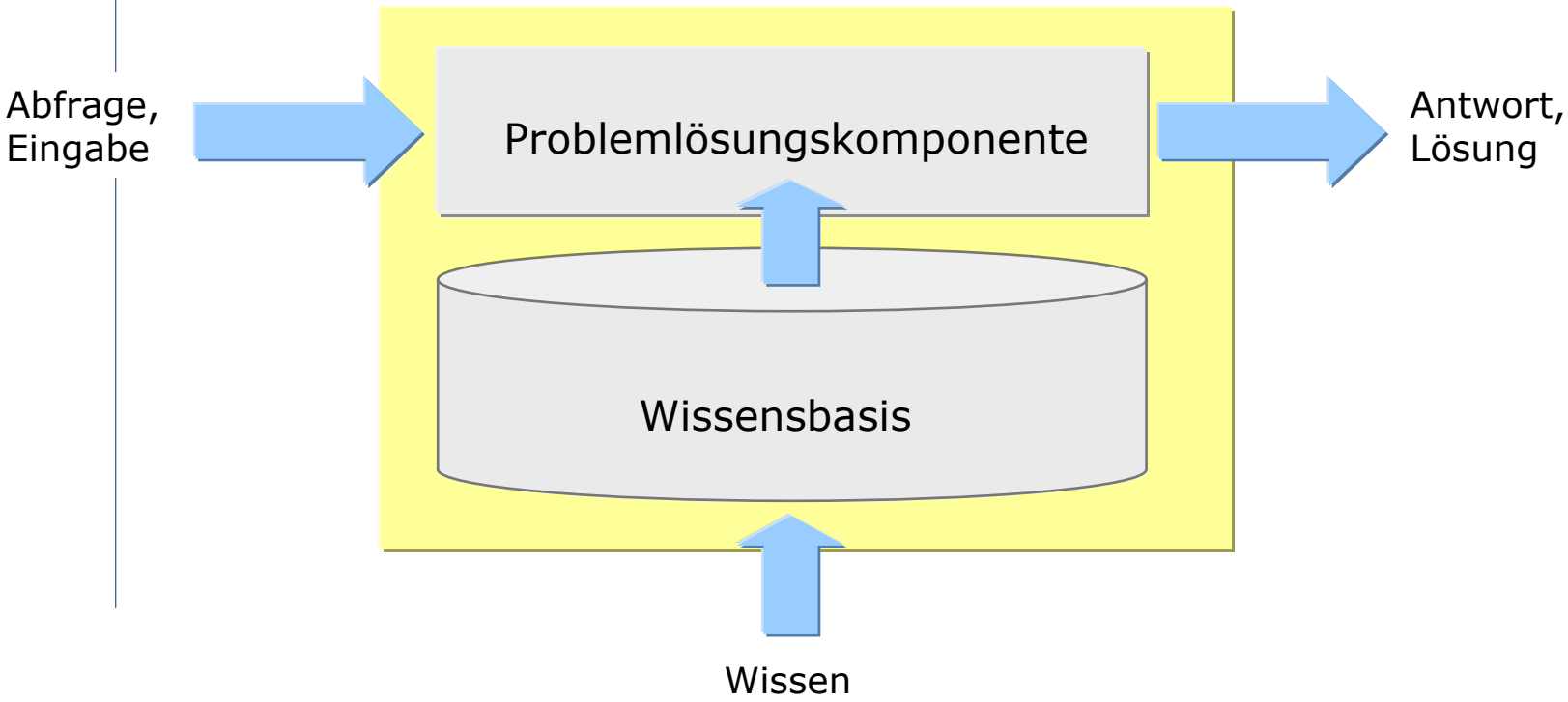
Was gehört zu einem wissensbasierten System?

- Ein WBS besteht aus explizitem Wissen und einer Problemlösungs-komponente.

WBS = Wissen + Problemlösungskomponente

Wissen = Wissens-atome + Verknüpfungs- bzw. Verarbeitungsregeln

Vereinfachte, verallgemeinerte Architektur eines WBS



Klassische Architektur wissensbasierter Systeme

Die Wissenserwerbs-komponente (WEK) dient zur Eingabe und Pflege der Wissensbasisinhalte. Dabei wird – sofern möglich - das Wis-sen auf Konsistenz und Plausibilität geprüft.

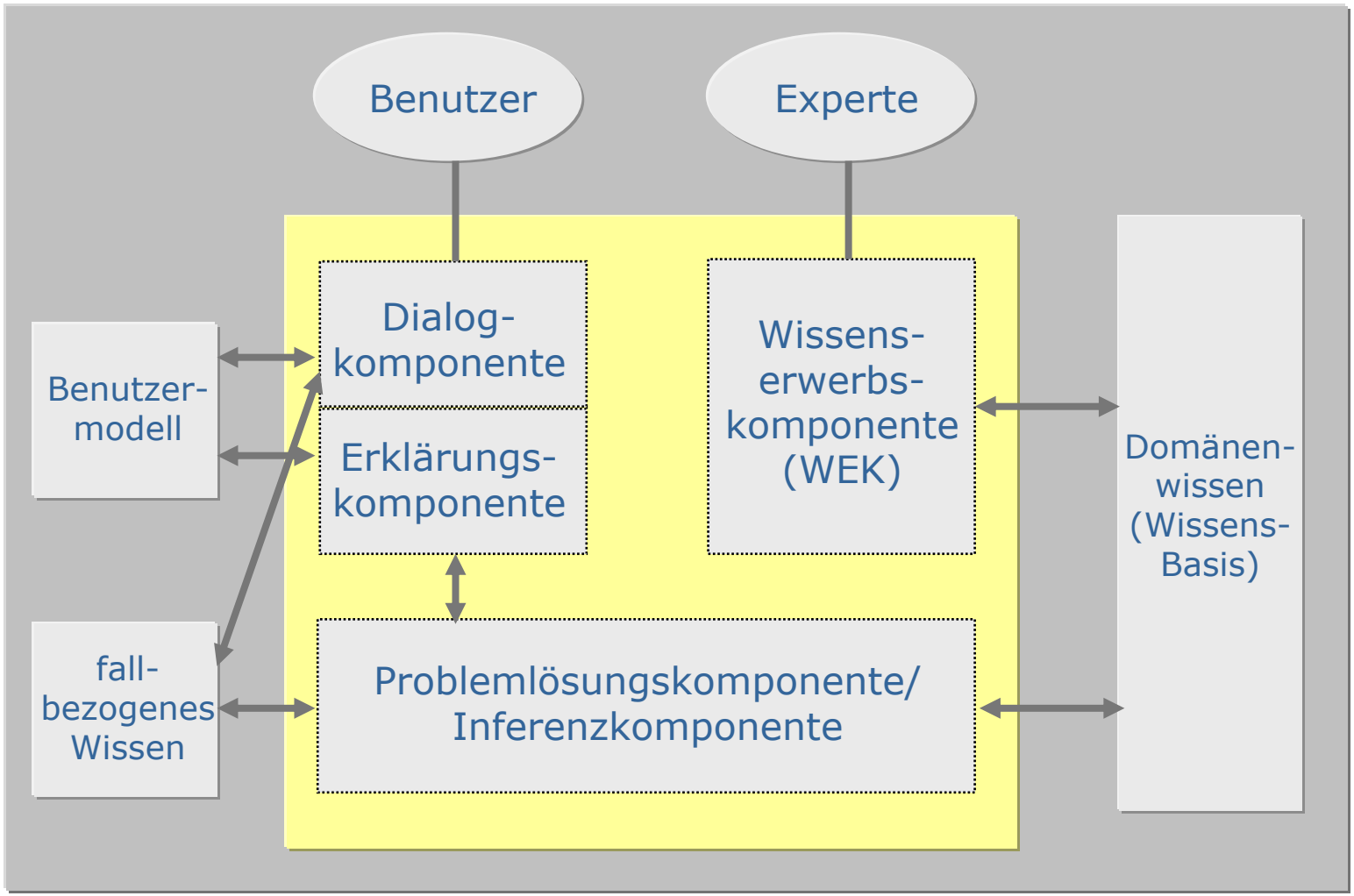
Die Wissensbasis (WB) enthält das über die WEK einge-gebene Domänen-wissen, ggf. ergänzt um Fall- und Meta-wissen.

Die Inferenzkompo-nente zieht die logi-schen Schlussfolge-rungen.

Die Erklärungskom-ponente versucht, dem Benutzer den Inferenzverlauf zu erklären und Fragen wie „Wie kam das Ergebnis zustande?“ zu beantworten.

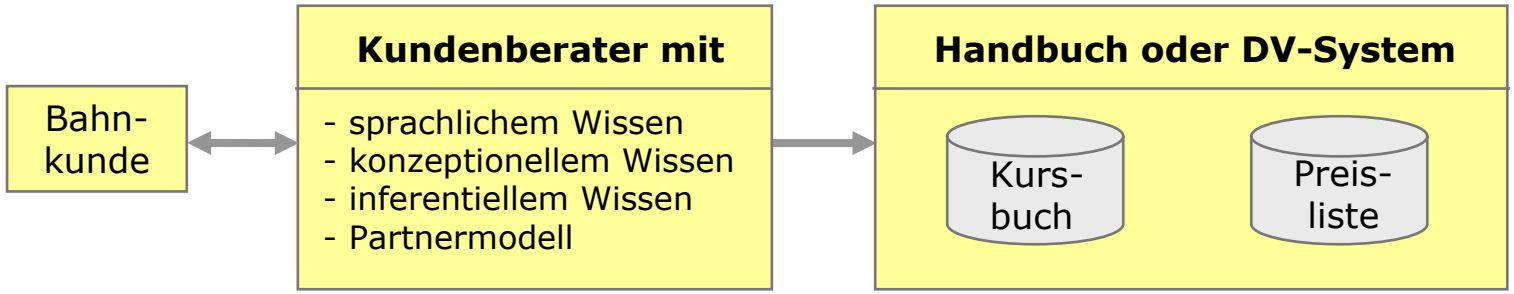
Die Dialogkompon-nente dient zur ad-äquaten Kommuni-kation mit dem Benutzer.

Der Prozess, das Wis-en zu erheben und adäquat abzu-bilden, heißt Know-ledge Engineering.

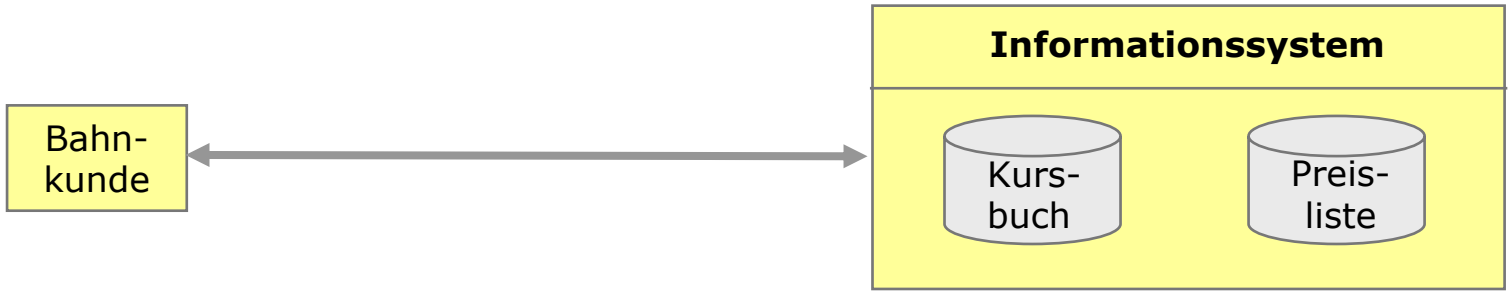


Datenverarbeitung vs. Wissensverarbeitung am Beispiel der Bahnauskunft

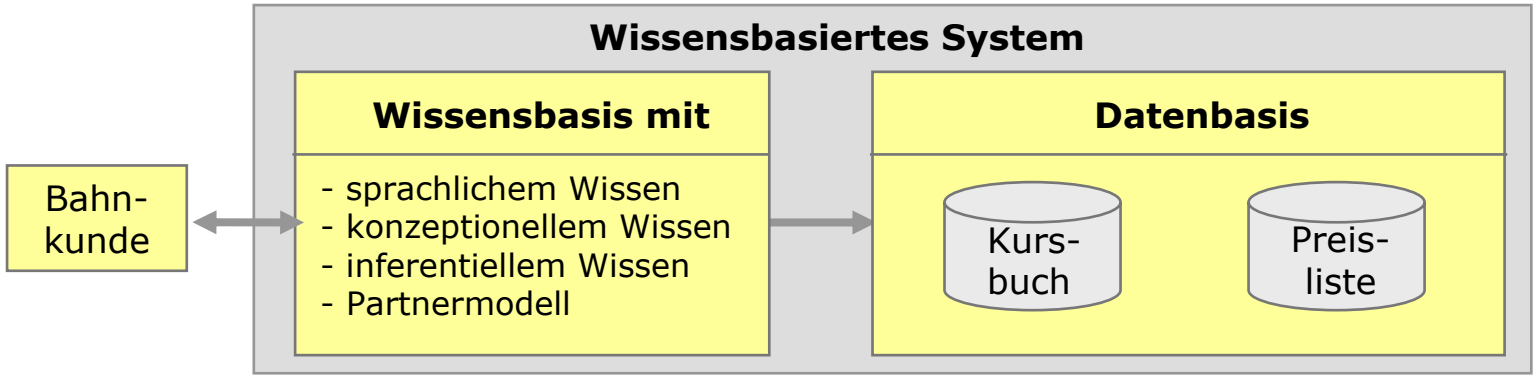
Klassische Auskunftssituation ggf. mit DV-Einsatz zur Unterstützung des Beraters



Minderung der Dienstleistungsqualität bei Rationalisierung durch konventionellen DV-Einsatz



Vervielfachung der Beratungskapazität ohne Qualitätsverlust durch kombinierte Wissens- & Datenverarbeitung



Typische Anwendungsklassen für WBS

- Diagnose
- Klassifikation
- Konfiguration
- Information & Beratung
- Rating & Scoring
- Kreditentscheidung
- Bilanzanalyse
- Produktionsplanung
- ...
- allgemein entscheidungsunterstützende Systeme

Nochmal: Was gehört zu einem wissensbasierten System?

- Ein WBS besteht aus Wissen und einer Problemlösungskomponente.

WBS = Wissen + Problemlösungskomponente

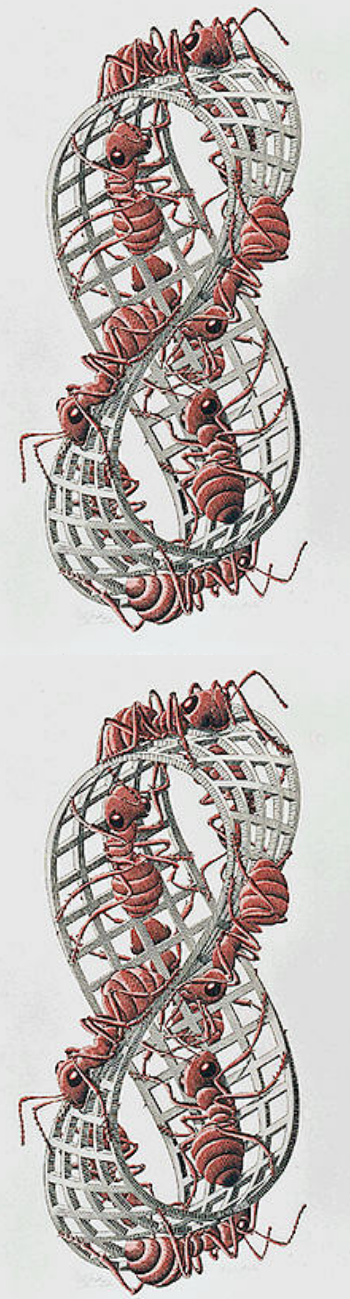
Wissen = Wissens-atome + Verknüpfungs- bzw. Verarbeitungsregeln

Logisches Wissen:		
Atome	Regeln	Ableitungsregeln
Fakten	wenn..., dann ...	Unifikation, Resolution

Funktionales Wissen:		
Daten	Funktionen	Funktionsauswertung

Objektorientiertes Wissen:		
Objekte	Methoden	Compiler / Interpreter

Regelhaftes Wissen:		
Fakten	Regeln	Compiler / Interpreter



Kap. 4.1: Wissensrepräsentation &-verarbeitung

Kategorien des Wissens (1)

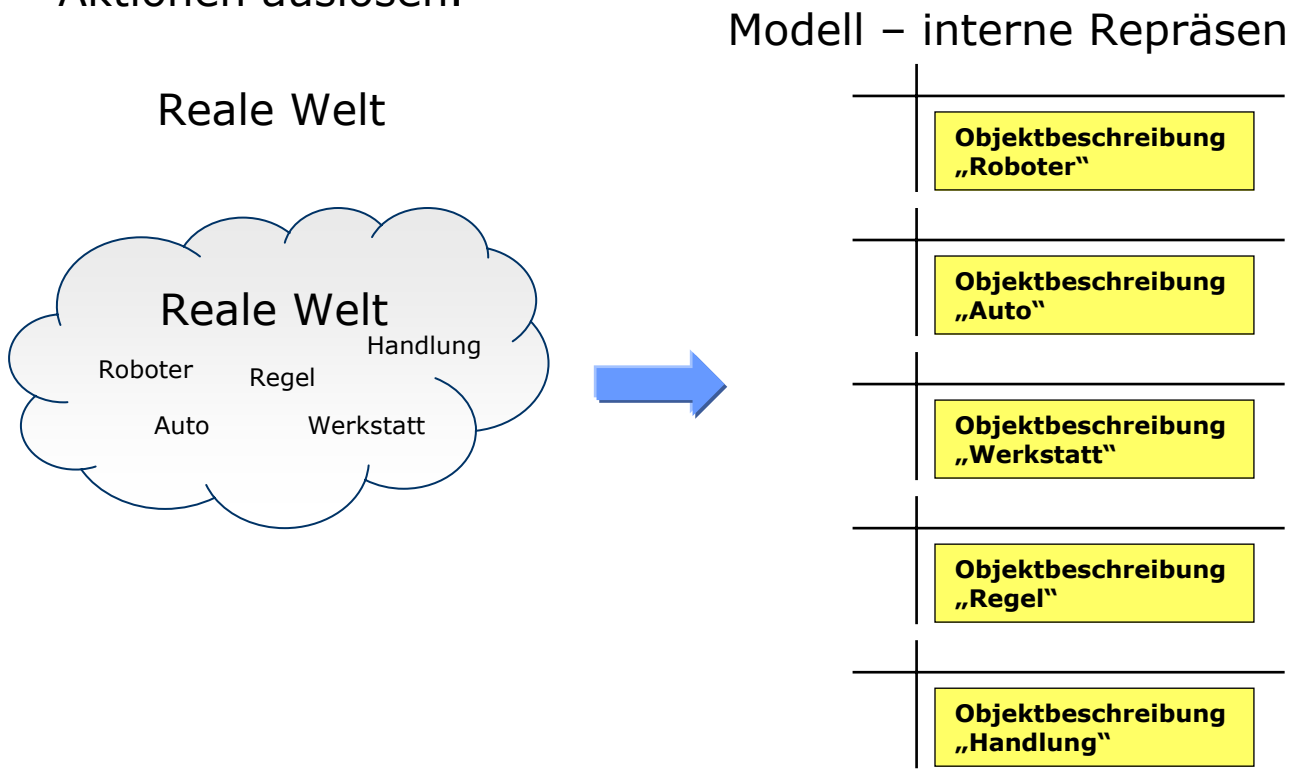
Wissen ist eine abstrakte Qualität. Es muss an eine symbolische Repräsentation gebunden sein, um einsatzfähig zu sein.
(A. Newell)

- Kategorien von Wissen. Es gibt Wissen über ...
  - Strukturen von Daten
  - Beziehungen zwischen Daten
  - Dateninhalte
  - Bedeutung von Daten
  - Regeln und Fakten
  - Wahrscheinlichkeiten
  - logisches Schließen
  - Wissen (Meta-Wissen) etc. ...
- Aus objektorientierter Sicht könnte man Wissen wie folgt gliedern:
  - Wissen über die Klassifizierung von Objekten
    - Klassenzugehörigkeit (is-a-Relation)
    - Aufbau der Klassenhierarchie
  - Wissen über Strukturen von Objekten
    - Liste der Attribute
    - Typ der Attribute
    - Meta-Attribute
  - Wissen über Beziehungen von Objekten
    - besteht-aus-Relation
    - andere Beziehungen
  - Wissen über das Verhalten von Objekten
    - Methoden
    - Regeln, nach denen sich Objekteigenschaften ändern
  - Wissen über die Manipulation von Objekten durch ein Regelsystem
    - Strategie der Auswahl von auf Objekte anwendbare Regeln
    - Anwendung von Regeln
    - Ausführung von Aktionsroutinen

Kategorien des Wissens (2)

Strukturierungs-kriterien:
- flach vs. tief
- sicher vs. unsicher
- qualitativ vs. quantitativ
- ...

- Die Auflistung ist sicher nicht vollständig; sie dient hier lediglich als Diskussionsgrundlage. Es gibt es zahlreiche alternative Mög-lichkeiten, Strukturierungskriterien für Wissen zu erstellen.
- Die Semantik, also die Bedeutung einer modellierten Welt, ist dem Menschen vorbehalten. Aber ein wissensbasiertes (z.B. regel-basiertes) System kann gemäß der in den Objekten und Methoden hinterlegten Semantik agieren, also Schlussfolgerungen ziehen und Aktionen auslösen.



Worin besteht der Unterschied zw. Daten und Wissen?

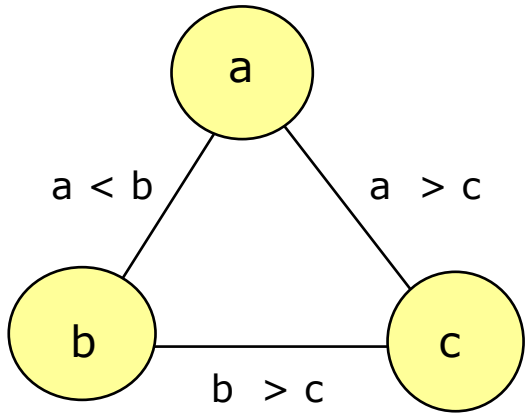
- Daten = Zeichen + Syntax
- Information = Daten + Semantik
- Wissen = Information + Verarbeitungsfähigkeit

- Wichtige Aufgaben der Wissensverarbeitung:
  - Suchen
  - Erkennen, Identifizieren
  - Informieren
  - Entscheiden
  - Verbessern
  - Schließen
  - ...

- Typische Aufgabenfelder im Kontext wissensbasierter Systeme:
  - Wissenserwerb
  - Wissensrepräsentation
  - Problemlösungs- bzw. Inferenzmechanismen
  - ...

Wissensrepräsentation

- Wie kann das Wissen formalisiert, strukturiert, organisiert werden?
  - Produktionsregeln
  - Frames & Scripts
  - Semantische Netze
  - Logik
  - Constraints
  - Fuzzy Logik
  - ...
- Ein Beispiel zur Wissensrepräsentation – Constraints
  - Constraints sind Neben- und Randbedingungen - typischerweise für Variablen.



a < b, b > c, a > c

a ∈ { 3, 4 }

b ∈ { 3, 4, 5 }

c ∈ { 3, 4, 5, 6 }

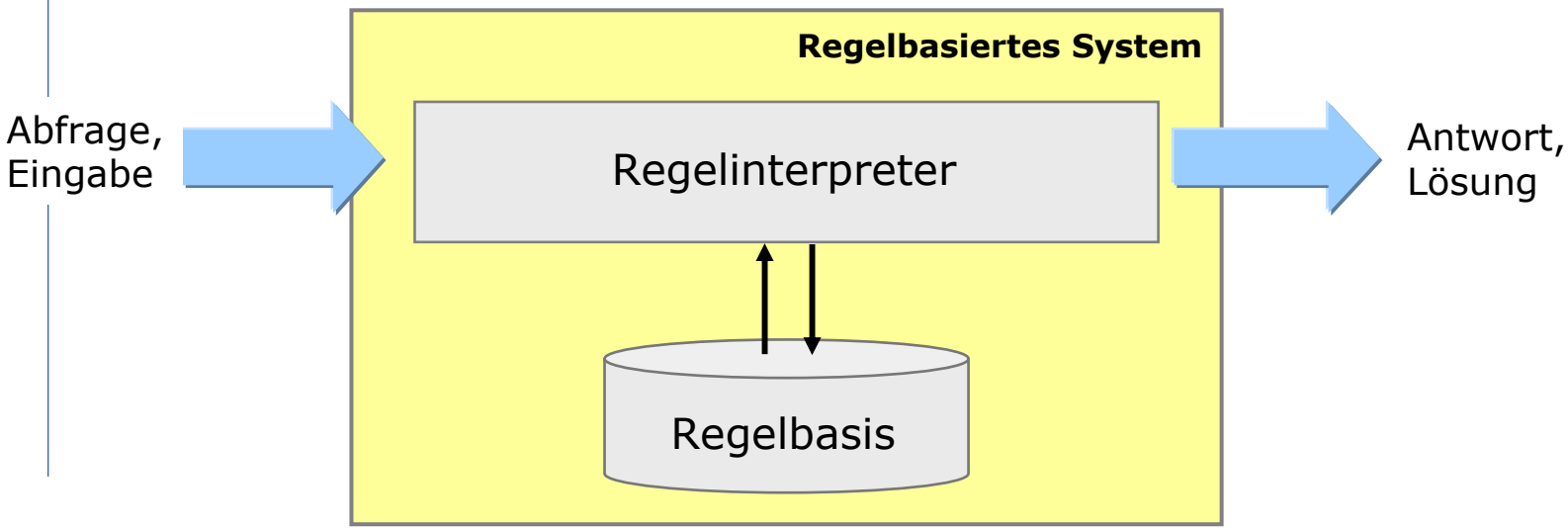
Lösung: a = 4, b = 5, c = 3

Kap. 4.2: Produktionsregeln & Regelverarbeitung

Regelbasierte Systeme – Das Prinzip

- In einem klassischen Programm, also einer algorithmischen Problemlösung, ist das Wissen direkt im Programmcode hinterlegt.

- In einem regelbasierten System wird das Wissen in Form von Regeln repräsentiert und von der Verarbeitungslogik isoliert.



Arbeitsweise regelbasierter Systeme

- Grundstruktur von Regeln: Eine einfache WENN-DANN-Regel hat den folgenden Aufbau:
  - WENN alle Voraussetzungen erfüllt sind, DANN ziehe Schlussfolgerungen (WENN Prämisse DANN Konklusion)

    bzw.
  - **IF** P_1, P_2, ..., P_n **THEN** K_1, K_2, ..., K_m
- Wenn alle Bedingungen P_i erfüllt sind, dann können alle Schlussfolgerungen K_j gezogen werden. Die P_i und K_j sind Wahrheitswerte, die nur WAHR (TRUE) oder FALSCH (FALSE) sein können.
- Neben Regeln gibt es auch noch Fakten. Das sind Aussagen, die immer erfüllt sind. Fakten können als Regeln ohne Prämisse betrachtet werden.
- Die Konklusion, also der THEN-Teil, kann auch Aktionen A enthalten, die bei Erfüllung aller Prämissen ebenfalls auszuführen sind. Es kann sogar sein, dass nur Aktionen auszuführen sind, also
  - **IF** P_1, P_2, ..., P_n **THEN** K_1, K_2, ..., K_m **DO** A_1, A_2, ..., A_l

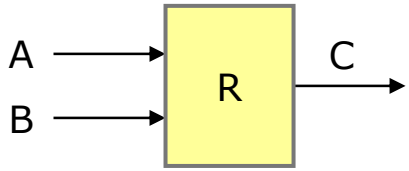
    bzw.
  - **IF** P_1, P_2, ..., P_n **THEN DO** A_1, A_2, ..., A_l

Aktionen sind eine Art Nebeneffekt. Wir nehmen im folgenden an, dass in den Aktionen, also im **DO**-Teil, keine Prämissen gesetzt werden können, damit das Schlussfolgerungsschema explizit sichtbar bleibt.

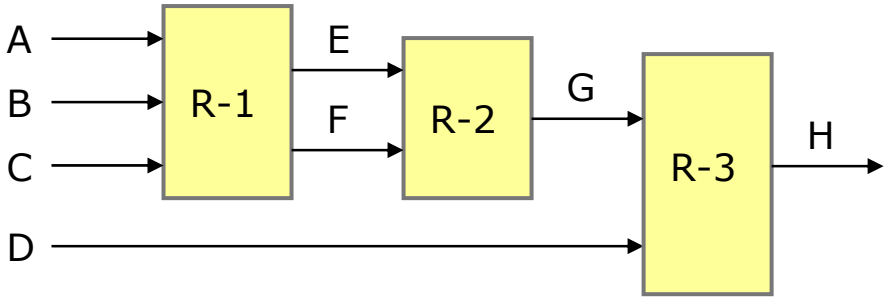
Vorwärtsverkettung von Regeln

- Die Konklusionen einer Regel können Prämissen einer anderen Regel sein. Dadurch ergeben sich viele potentielle Verkettungs-möglichkeiten von Regeln.
- Zur Verdeutlichung verwenden wir die folgende Notation:

  Regel R: **IF** A, B **THEN** C



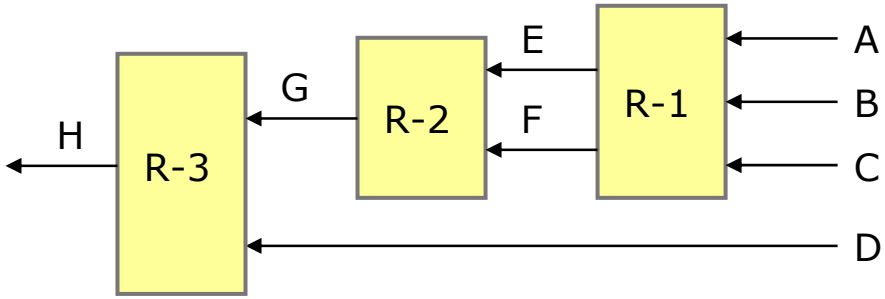
- Wir betrachten nun das folgende Regelsystem, bestehend aus den Regeln R-1, R-2 und R-3: Wir nehmen an, es gelten die Fakten A, B, C und D. Da A, B und C gelten, kann mittels R-1 auf E und F geschlossen werden. Mittels R-2 kann dann direkt auf G weiter-geschlossen werden. Wegen D und R-3 gilt dann auch H.



- Diese Art der Regelverkettung heißt Vorwärtsverkettung (Forward Chaining).

Rückwärtsverkettung von Regeln

- Wir betrachten dasselbe Regelsystem, um nun umgekehrt – rückwärts – zu schließen.



- Dabei besteht das Ziel darin, die Aussage H zu beweisen. H gilt gemäß Regel R-3, wenn G und D gelten. D gilt gemäß Voraussetzung. G lässt sich aus Regel R-2 ableiten, wenn E und F gelten. E und F lassen sich aus R-1 ableiten, da A, B und C gelten. H kann also bewiesen werden, weil A, B, C und D gelten.

- Diese Form der Regelverkettung heißt Rückwärtsverkettung (Backward Chaining).

Vergleich Vorwärts- / Rückwärtsverkettung

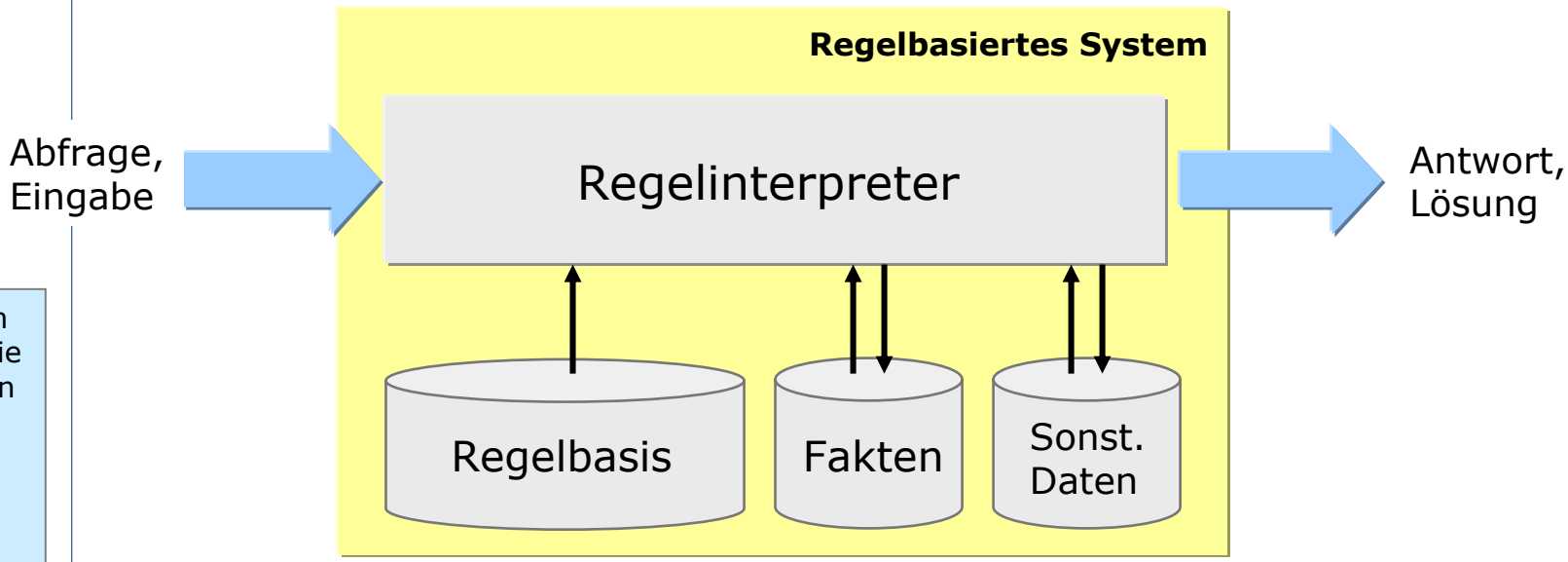
- Regelsysteme können sehr groß werden. Z.B. hatte das System R1/XCON (s. VWD_K1_Histo_KI, S. 41) zur Konfiguration von DEC-Mainframe-Computern, das erste kommerziell erfolgreiche wissensbasierte System, eine Regelbasis aus ca. 14.000 Regeln.
- Bei der Vorwärtsverkettung kann es sehr leicht passieren, dass Unmengen von Schlussfolgerungen gezogen werden, die gar nicht benötigt werden. Es kommt zur kombinatorischen Explosion von Regelausführungen. Es werden also Strategien benötigt, um das unkontrollierte Feuern der Regeln zu steuern.
- Bei der Rückwärtsverkettung geht man von dem zu beweisenden Satz aus und arbeitet sich schrittweise bis zu den gegebenen Voraussetzungen durch. Die Programmiersprache PROLOG basiert z.B. auf dem Prinzip der Rückwärtsverkettung. Die Rückwärts-verkettung arbeitet wesentlich zielgerichteter.
- <u>Vorwärtsverkettung:</u>
  - datengetrieben (data driven)
  - synthetisch, Ableiten aller Folgerungen
  - geeignet bei kleinen Regelmengen
  - Regeln sind anwendbar, wenn Prämissen erfüllt sind

<u>Rückwärtsverkettung:</u>

- zielgetrieben (goal driven)
- analytisch, Beweis der Hypothese
- nötig bei größeren Regelmengen
- Regeln sind anwendbar, wenn Folgerung zu beweisen ist.

Regelinterpreter (1)

- Zur Verarbeitung der Regeln wird noch eine Problemlösungs-komponente benötigt, der Regelinterpreter.



Aktionsroutinen manipulieren die sonstigen Daten des Systems und sorgen für die Interaktion mit dem Benutzer.

- Nach dem Start wählt der Regelinterpreter die anzuwendenden Regeln aus und führt einen ersten Schlussfolgerungszyklus durch.
- Dabei ausgeführte Aktionsroutinen manipulieren sonstige Daten und erfragen ggf. neue Fakten. Außerdem können Regelausführungen zu einer neuen Faktenlage führen. Ein neuer Zyklus beginnt.

Regelinterpreter (2)

- Die „sonstigen Daten" stehen für alle anderen Daten, die mit dem eigentlichen Schlussfolgerungsmechanismus des Regelinterpreters nichts zu tun haben. Sie sind Nebeneffekt der Aktionsroutinen.
- Es gibt verschiedene Strategien, um die Auswahl der anzuwendenden Regeln zu steuern:
  - Einfachste Form:
    Suche passende anwendbare Regel.
    Führe Regel aus.
  - Alternative 1:
    Suche alle anwendbaren Regeln.
    Feuere alle Regeln nacheinander.
  - Alternative 2:
    Suche alle anwendbaren Regeln.
    Wähle bestimmte Regeln nach einer gewissen Heuristik aus.
    Feuere die Regeln in der heuristisch bestimmten Reihenfolge.
  - Sonstige: ...
- Intelligentere Strategien verwenden Heuristiken, die durch Meta-Regeln gesteuert werden können. Meta-Regeln sind Regeln, die die Auswahl und Reihenfolge der anzuwendenden Regeln steuern.
  - Prioritäten von Regeln
  - Prioritäten von Fakten
  - Häufigkeit der früheren Verwendung
  - Spezialisierungsgrad der Regeln
  - Gruppenbildungen von Regeln etc.

Wissensrepräsentation & -verarbeitung

- Im folgenden sollen zwei weitere wichtige Wissensrepräsentations- und verarbeitungskonzepte intensiver untersucht werden.
- Hierzu werden zwei separate Dokumente bereitgestellt:

Siehe Dok. VWD_K4b_PredLogic.pdf

Kap. 4.3: Prädikatenlogik & logische Programmierung
(VWD_K4b_Logic.pdf)

Siehe Dok. VWD_K4c_Prolog.pdf

Kap. 4.4: Steilkurs Programmieren in Prolog
(VWD_K4c_Prolog.pdf)

Siehe Dok. VWD_K4d_FuzzyLogic.pdf

Kap. 4.5: Prinzipien der Fuzzy Logik
(VWD_K4d_FuzzyLogic.pdf)