Künstliche Intelligenz

Sebastian Iwanowski
FH Wedel

Kap. 1:
Einführung und Überblick

Vorlesungsüberblick

Inhaltliche Voraussetzungen:

GdP/GTI, Programmieren I und II

hilfreich: Objektorientierte Programmiertechnik

Lernziele dieser Vorlesung:

Anwendungsgebiete und -beispiele

Methoden, Techniken und Architekturen der KI

Theoretisches Verständnis der zugrundeliegenden Logik

Detailkenntnisse in: Modellbasierte Diagnosetechnik

Spiele-KI

Ameisenalgorithmen (pheromonbasierte Ansätze)

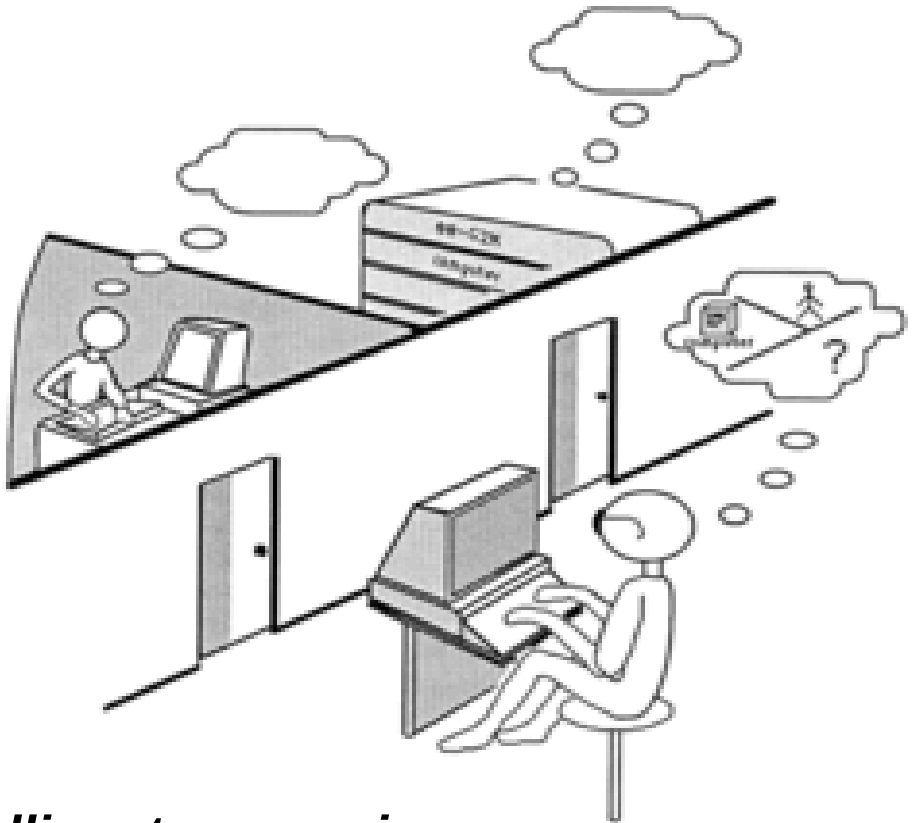
Vorlesungsüberblick

Vorlesungsgliederung:

1. Einführung und Überblick
2. Logische Grundlagen der KI
3. Algorithmische Grundlagen der KI
4. Verschiedene Wissensrepräsentationsarten und deren Verarbeitung
5. Anwendungsgebiet: Technische Diagnose
6. Anwendungsgebiet: Spiele-KI
7. Ameisenalgorithmen und ihre Anwendungen

Was ist KI ?

Der Turing-Test



Eine Software verhält sich intelligent, wenn ein Mensch ihr Verhalten nicht vom Verhalten eines Menschen unterscheiden kann.

Anwendungsgebiet: Spiele-KI

Verschiedene Spieletypen:

1. **Rundenbasierte Strategiespiele**
2. **Echtzeit-Strategiespiele**
3. **Mehrbenutzer-Strategiespiele**
4. **Sportsimulationen**
5. **Entwicklungssimulationen**

Nähere Infos: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1
http://www.fh-wedel.de/mitarbeiter/iw/lehrveranstaltungen-in-frueheren-semestern-ab-ss-2007/ss-2007/informatik-seminar-spiele-ki/

1. Rundenbasierte Strategiespiele

Problem:

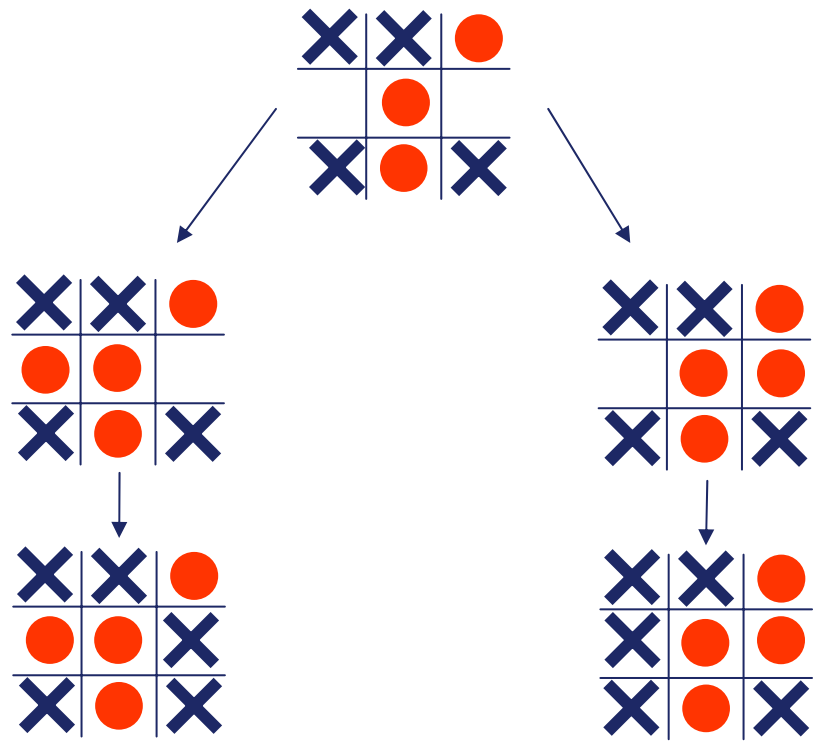
Finde eine Strategie von einem Startzustand zu einem Gewinnzustand

Bsp: Tic-Tac-Toe

Ausgangszustand

Folgezustand

Folgezustand



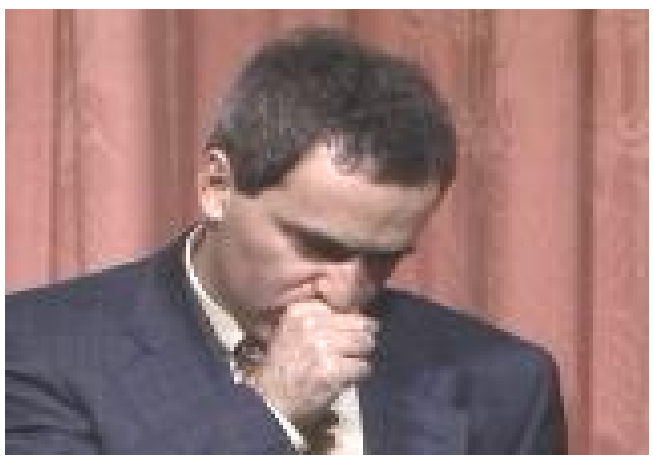
Quelle und nähere Infos: Seminarvortrag und Ausarbeitung von Nils Böckmann, SS 2005, Nr. 13
http://www.fh-wedel.de/Archiv/iw/Lehrveranstaltungen/SS2005/SeminarKI.html

1. Rundenbasierte Strategiespiele

Schachcomputer

Meilenstein 1997:

Kasparov 2.5 – Deep Blue 3.5



Weitere infos: http://www.research.ibm.com/deepblue

1. Rundenbasierte Strategiespiele

Schachcomputer

Unterschied 1

- Deep Blue kann über 200,000,000 Schach-Situationen pro Sekunde errechnen und bewerten.
- Kasparov nur 3.

Unterschied 2

- Deep Blue hat nur begrenztes Wissen über Schach, aber riesiges über Berechnungen.
- Bei Kasparov ist das umgekehrt.

Unterschied 3

- Deep Blue hat keine Emotionen oder Gefühle.
- Kasparov nutzt diese, um Weltklasseschach zu spielen.

1. Rundenbasierte Strategiespiele

Schachcomputer

Unterschied 4

- Deep Blue hat von vielen Leuten (Schachmeistern etc.) profitiert.
- Kasparov wird nur von seinem Coach Yuri Dokhoian und seiner eigenen Leidenschaft, das beste Schach der Welt zu spielen, gefördert.

Unterschied 5

- Deep Blue ist kein lernendes System. Es ist daher nicht fähig, von seinem Gegner zu „lernen“ oder über die momentane Situation des Schachspiels „nachzudenken“.
- Kasparov ist fähig, sehr schnell aus seinen Fehlern und Erfolgen zu lernen.

Unterschied 6

- Deep Blue kann durch nichts und niemanden von seinem Spiel abgebracht werden.
- Kasparov ist stets durch menschliche Schwächen wie: Langeweile, Verlust der Konzentration etc. beeinflussbar.

1. Rundenbasierte Strategiespiele

Schachcomputer

Unterschied 7

- Deep Blue kann sehr effektiv Schachprobleme zu lösen, aber nichts anderes.
- Gary Kasparov ist sehr vielseitig: Er schrieb mehrere Bücher, ist politisch aktiv, etc.

Unterschied 8

- Jeder Wechsel der Spielstrategie von Deep Blue muss von seinen Programmierern nach dem Spiel vorgenommen werden.
- Kasparov kann jederzeit seine Strategie ändern.

Unterschied 9

- Deep Blue kann nur Schachpositionen bewerten, nicht aber Schwächen seines Gegners.
- Kasparov kann die Schwächen seines Gegners herausfinden und dann ausnutzen.

Unterschied 10

- Deep Blue berechnet sehr viele Züge voraus, um die beste Position herauszufinden.
- Kasparov macht das genauso, schafft aber wesentlich weniger.

2. Echtzeit-Strategiespiele

Beispiele:

Half Life

Command + Conquer 3

2. Echtzeit-Strategiespiele

Typische KI-Anforderungen:

- Wegfindung und Terrainanalyse
- Ressourcen-Planung
- Taktiken und Strategien

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

2. Echtzeit-Strategiespiele

Anforderung Wegfindung und Terrainanalyse

(Beispiele aus Command + Conquer)

- Wie kommt gesamte Armee von meiner Basis zur gegnerischen Basis?
- Wie überquert gesamte Armee Brücke und behindert sich dabei nicht gegenseitig?
- Gibt es auf einer geplanten Route Hindernisse?
- Gibt es Terrain, welches ich besser nicht überquere?
- Wo baue ich was? – z.B. um nicht Ionenstürmen ausgesetzt zu sein

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

2. Echtzeit-Strategiespiele

Anforderung Resourcen-Planung

(Beispiele aus Command + Conquer)

- Ich habe 27.500$, wie gebe ich diese auf lange Sicht am sinnvollsten aus?
- In welchen Technologiezweig soll ich investieren? (z.B. Luft- oder Bodeneinheiten)

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

2. Echtzeit-Strategiespiele

Anforderung Taktiken und Strategien

(Beispiele aus Command + Conquer)

- Wie baue ich meine Basis auf, um mich gut verteidigen zu können?
- Welche Formation ist bei einem Angriff am sinnvollsten?
- Analyse des Gegnerverhaltens
- Viele kleine Einheiten bilden oder eine große?

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

3. Mehrbenutzer-Strategiespiele

Beispiel: World of Warcraft

Rollenspiel mit verschiedenen Charakteren, das im WWW gespielt wird

3. Mehrbenutzer-Strategiespiele

Beispiel: World of Warcraft

- Grundlegende Algorithmen, z.B. Wegfindung
- Waypoints (Patrouille gegnerischer Einheiten)
- Ansätze sozialen Verhaltens

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

3. Mehrbenutzer-Strategiespiele

Beispiel: World of Warcraft

- World Events als Scripte
- Denkbare Erweiterung von WoW:
  - WoW mit KI-Elementen
    wie in einem modernen Echtzeit-Strategiespiel
  - Vollkommen autonomes Verhalten künstlicher Gegner
    (Planung von Angriffen, etc.)

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

4. Sportsimulationen

Autorennen

Quelle und Details: Seminarvortrag und Ausarbeitung von Yannick Block, SS 2007, Nr. 8

4. Sportsimulationen

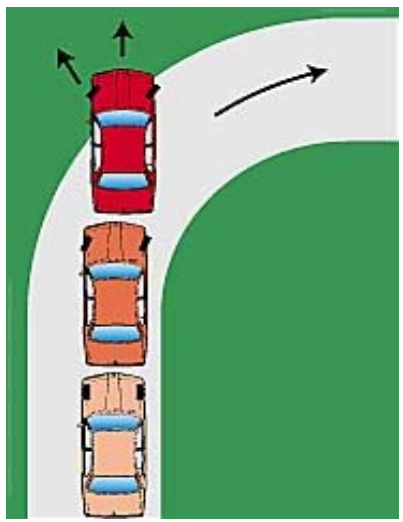
Autorennen

Quelle und Details: Seminarvortrag und Ausarbeitung von Yannick Block, SS 2007, Nr. 8

4. Sportsimulationen

Autorennen

Intelligente Fahrzeugsteuerung

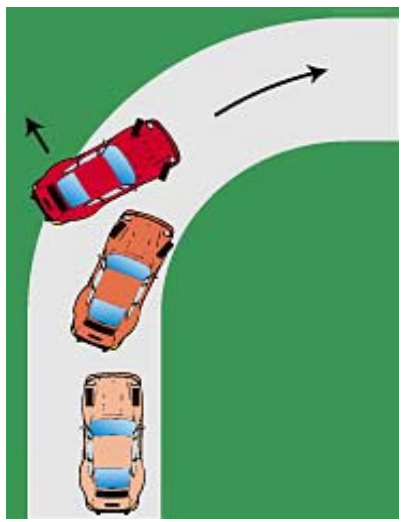


Quelle und Details: Seminarvortrag und Ausarbeitung von Yannick Block, SS 2007, Nr. 8

4. Sportsimulationen

Autorennen

Intelligente Fahrzeugsteuerung



Quelle und Details: Seminarvortrag und Ausarbeitung von Yannick Block, SS 2007, Nr. 8

4. Sportsimulationen

FIFA 2007

- Wegfindungsalgorithmen
- Taktische Vorgaben
- Verschiedene Standardsituationen:
  - Pässe
  - Eckstöße
  - Freistöße

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

4. Sportsimulationen

FIFA 2007

physikalische Anforderungen:

- Gegnerischen Spielern muss ausgewichen werden.
- Entwicklung des Spiels auf ein Ziel hin: Torschuss
- Torschüsse dürfen nicht zu perfekt sein.
- Verschiedene andere Situationen, wie Eckstöße, Freistöße, etc.

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

4. Sportsimulationen

FIFA 2007

Implementierung mit Zustandsautomaten:

- Jeder Zustand beschreibt aktuelle Aufgabe eines Spielers
- Zustände wie „Ich habe den Ball“, „Ich bin Ball-nächster-Spieler“, „Ich bin verletzt“

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

5. Entwicklungssimulationen

Bsp.: Sims

- Spieler modifiziert Prioritätenliste eines Sim-Charakters
- Wenn Spieler passiv:

  Handlung nach bestimmten Routinen

  (verschiedenen Bedürfnissen wird automatisch nachgekommen)

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

5. Entwicklungssimulationen

Bsp.: Sims

Smart terrain / Message-Passing:

Beispiel: Sim hat Durst

- Schrank in der Küche signalisiert: Ich hab was
- Sim trinkt und hat Bedürfnis zur Toilette zu gehen
- Toilette signalisiert von sich aus: Ich bin eine Toilette

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

5. Entwicklungssimulationen

Bsp.: Sims

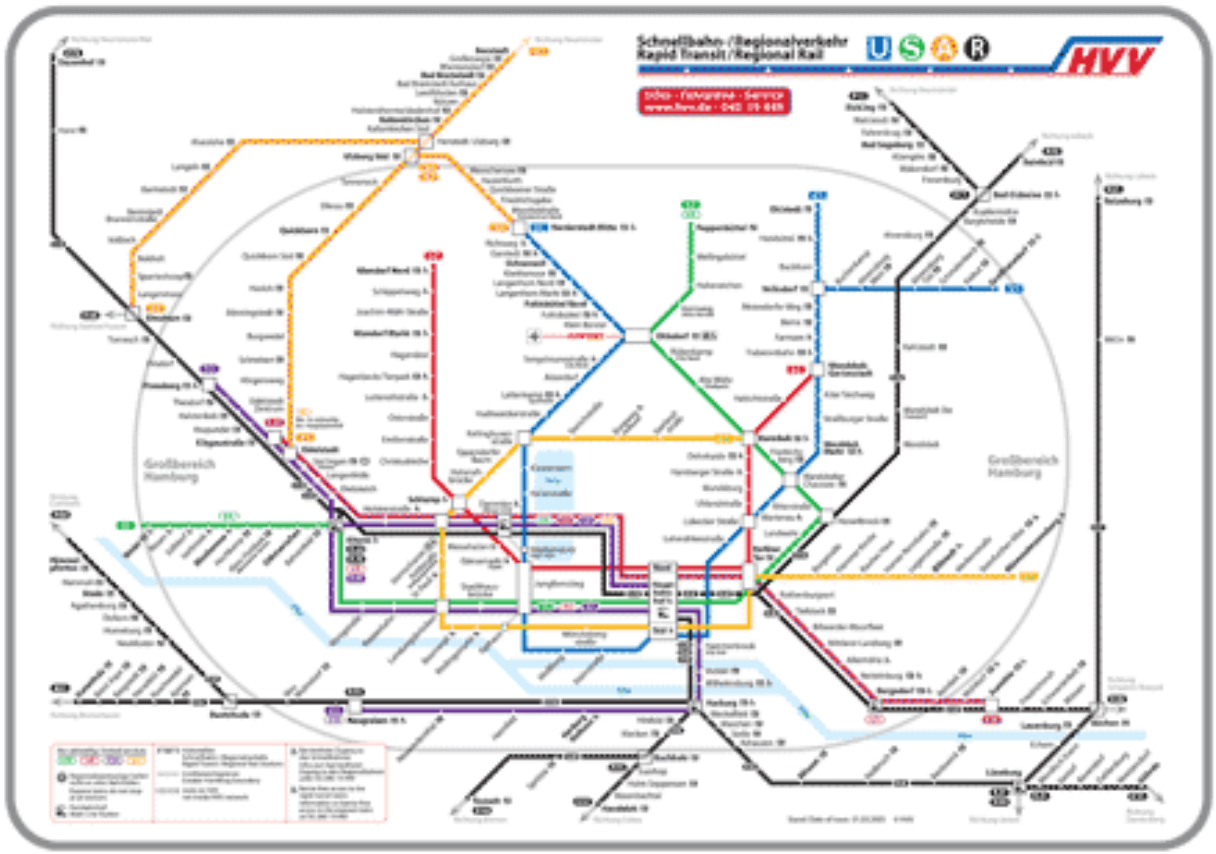
- Verwendung genetischer Algorithmen
- Weitergabe von Merkmalen phänotypischer und „seelischer“ Art an Nachkommen

Quelle: Seminarvortrag und Ausarbeitung von Julian Huppertz, SS 2007, Nr. 1

Basisproblem: Das Problem des kürzesten Weges

Problem:

Finde zu zwei Punkten A und B in einem gegebenen Verkehrsnetz den kürzesten Weg von A nach B, der ausschließlich Strecken dieses Verkehrsnetzes benutzt.



Weitere Infos: Seminarvortrag und Ausarbeitung von Stefan Görlich, SS 2005, Nr. 5,
http://www.fh-wedel.de/archiv/iw/Lehrveranstaltungen/SS2005/SeminarKI.html

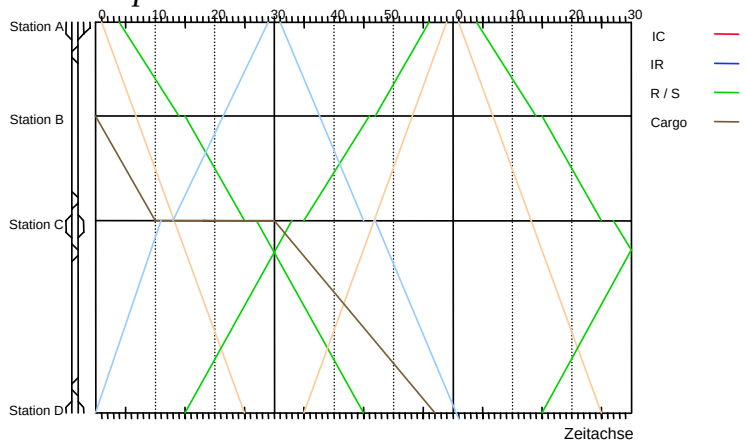
Schienenverkehr: Agentenorientierte Steuerung

Welcher Zug darf wann auf welchem Streckenelement fahren ?

Dispositionsebene:

Koordinierung aller Züge
über alle Streckenelemente

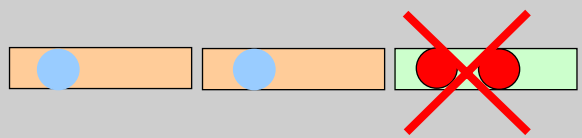
Graphische Fahrplankonstruktion



Sicherungssebene:

Absicherung eines Streckenelements:
Nur 1 Zug zur gleichen Zeit

wird hier nicht verändert



Blockungsprinzip

ortsfeste Signalisierung

Zugbeeinflussungssysteme

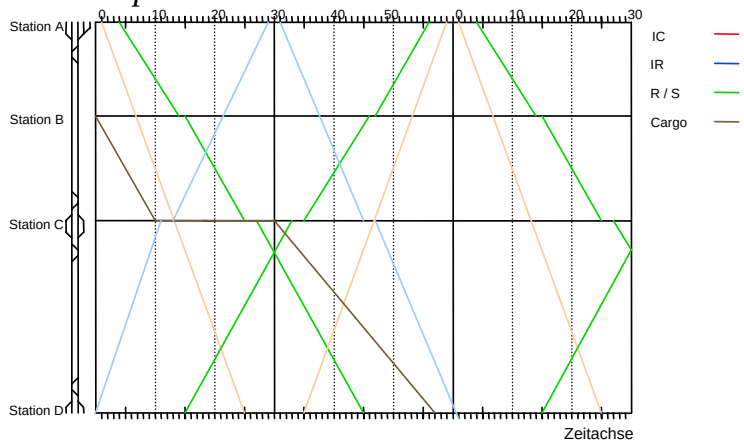
Schienenverkehr: Agentenorientierte Steuerung

Welcher Zug darf wann auf welchem Streckenelement fahren ?

Dispositionsebene:

Koordinierung aller Züge
über alle Streckenelemente

Graphische Fahrplankonstruktion



Stand der Technik: **Fahrpläne und Routenführungen werden in Zentrale vorausgeplant**

Weichen werden zentral gestellt

Problem: **unflexibel gegenüber**

- **unvorhersehbaren spontanen Änderungen (“Störungen”)**
- **spontane Bedarfsänderung**

Schienenverkehr: Agentenorientierte Steuerung

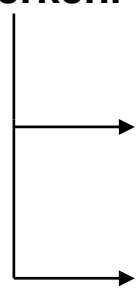
Weg von der zentralen Steuerung !

Alternative: **Züge stellen sich selbst alle Weichen**

Züge handeln sich untereinander freie Belegungen aus

bereits realisiert: **Straßenbahnen** **durch Menschen (mit elektronischer Unterstützung)**

neues Konzept für: **Fernverkehr** **durch autonome Softwareeinheiten**



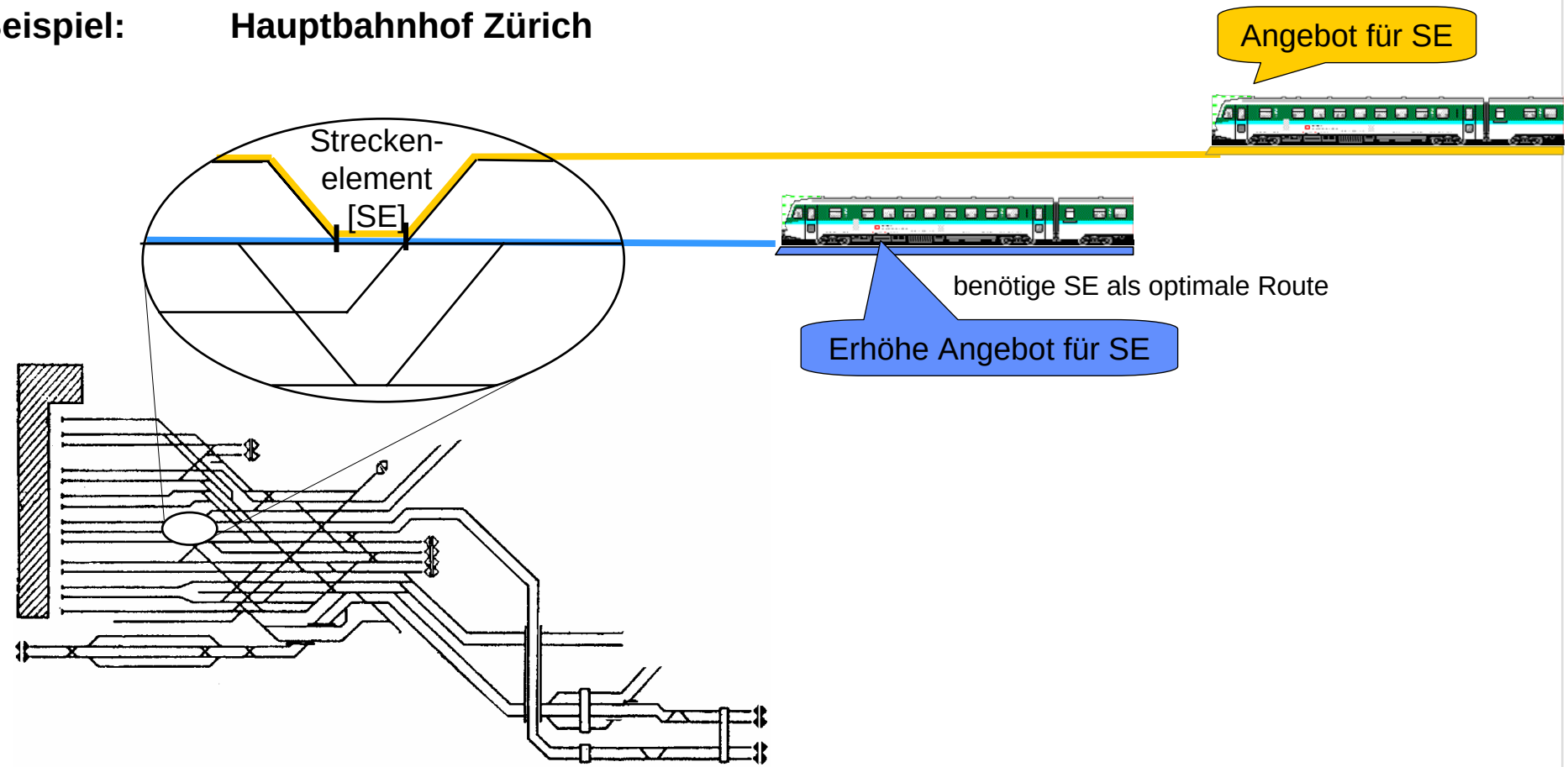
Züge mit Fahrplan **hauptsächlich Personenverkehr**

Züge ohne Fahrplan **hauptsächlich Güterverkehr**

Schienenverkehr: Agentenorientierte Steuerung

Verhandlungsmethode: Elektronische Auktion

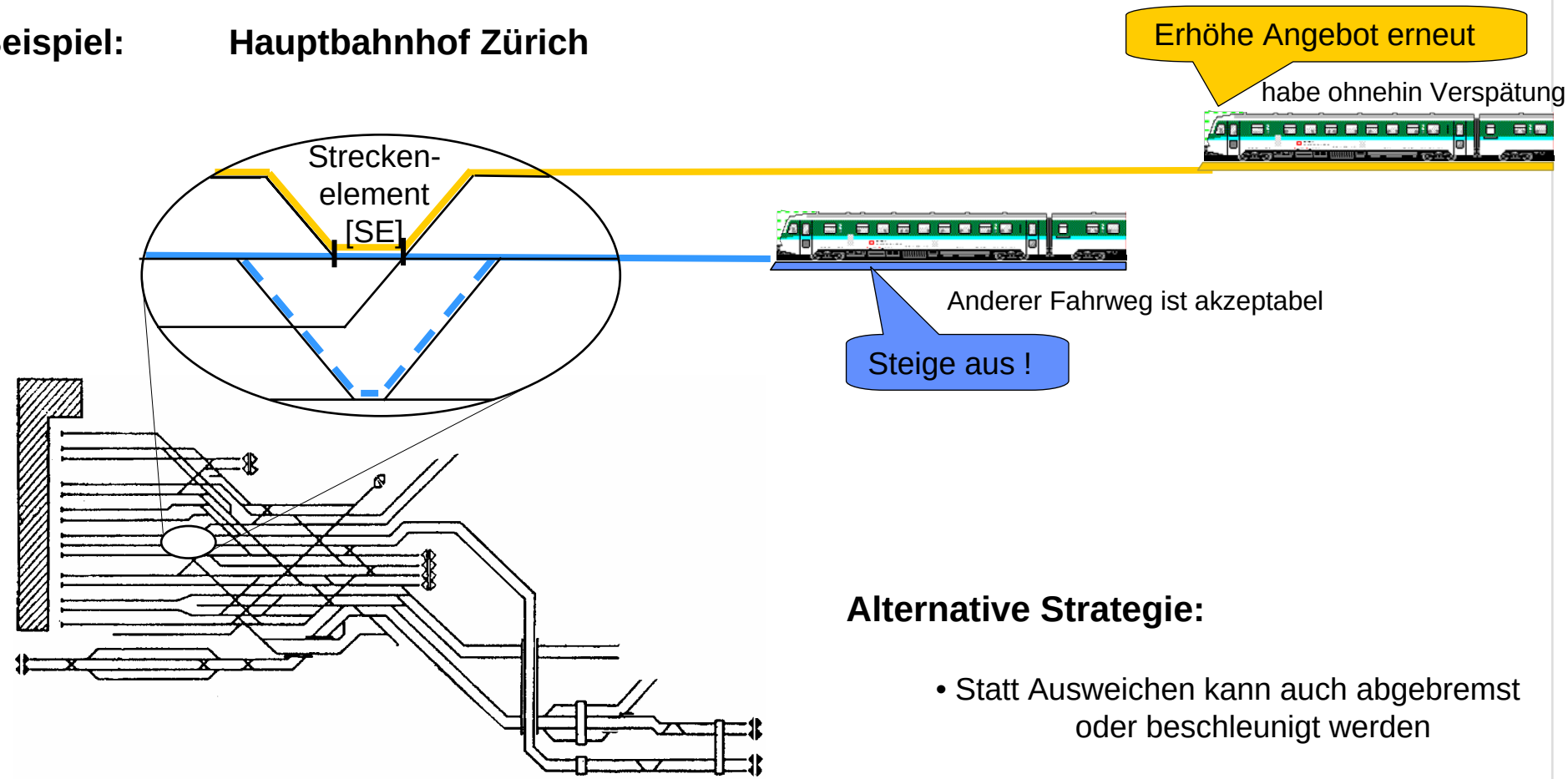
Beispiel: **Hauptbahnhof Zürich**



Schienenverkehr: Agentenorientierte Steuerung

Verhandlungsmethode: Elektronische Auktion

Beispiel: **Hauptbahnhof Zürich**



Alternative Strategie:

- Statt Ausweichen kann auch abgebremst oder beschleunigt werden

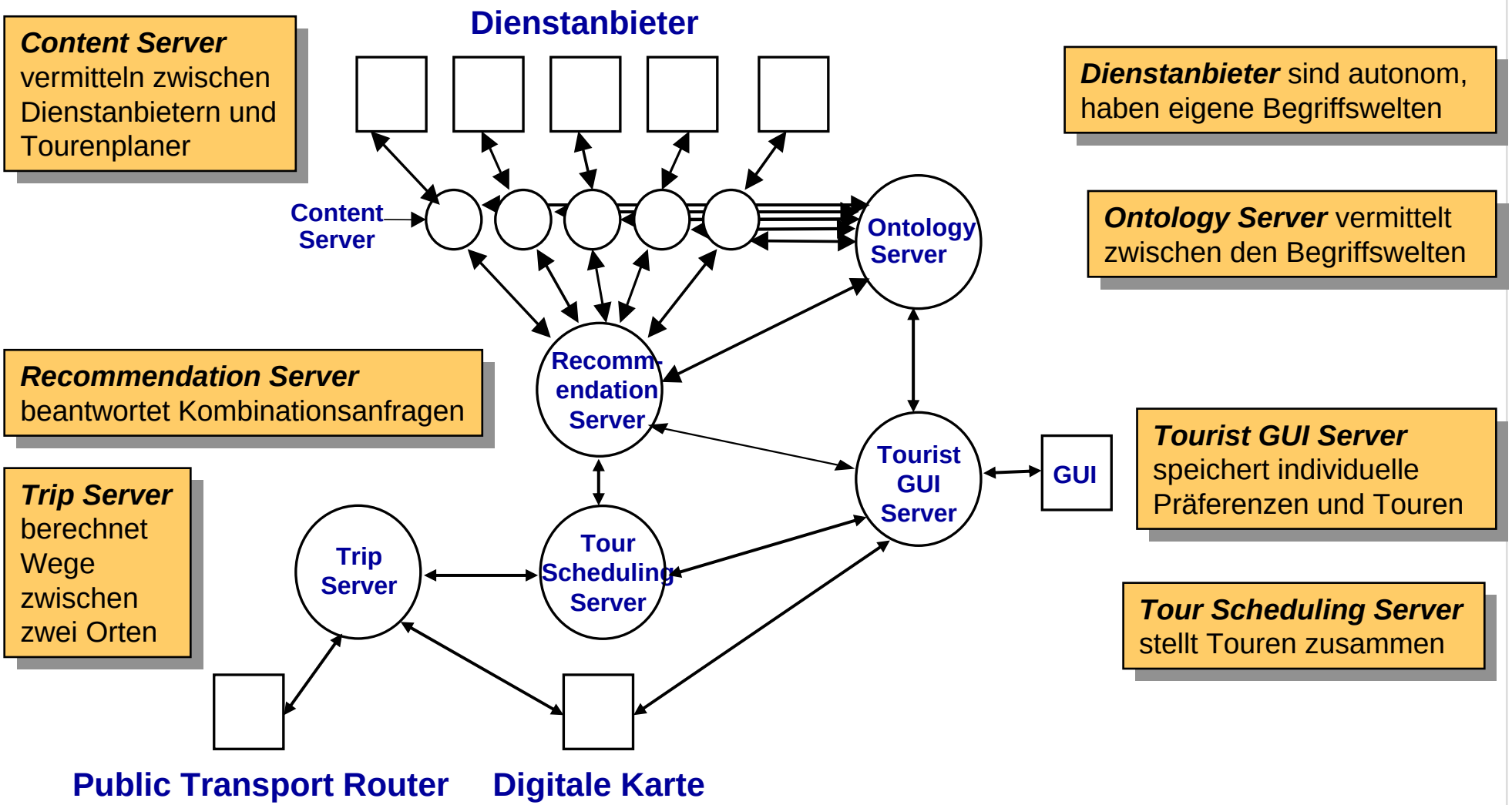
Verteiltes Touristeninformationsystem

Forderungen:

- Tourist hat die endgültige Kontrolle
- Dienstanbieter ist autonom und trägt die Verantwortung für die Informationen
- Unabhängiges Makeln zwischen verschiedenen Anbietern
- Flexibilität gegenüber Anforderungsänderungen, sogar während der Tour
- Fehlertoleranz gegenüber Ausfall von Dienstanbietern

Verteiltes Touristeninformationsystem

Architektur des Tourenplaners: Prototyp einer SOA



Content Server vermitteln zwischen Dienstanbietern und Tourenplaner

Dienstanbieter sind autonom, haben eigene Begriffswelten

Ontology Server vermittelt zwischen den Begriffswelten

Recommendation Server beantwortet Kombinationsanfragen

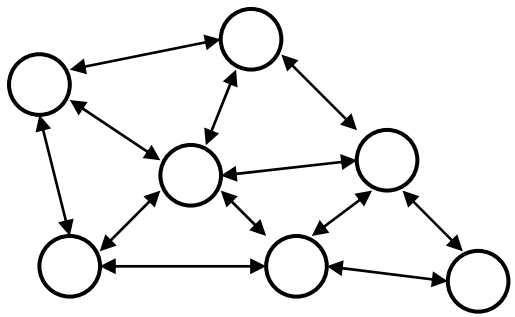
Tourist GUI Server speichert individuelle Präferenzen und Touren

Trip Server berechnet Wege zwischen zwei Orten

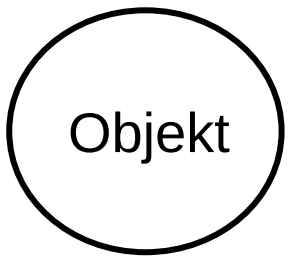
Tour Scheduling Server stellt Touren zusammen

Basistechnologie: Agentenorientierte Software

Multiagentensystem:



Softwareagent:



autonom

soziale Kompetenz

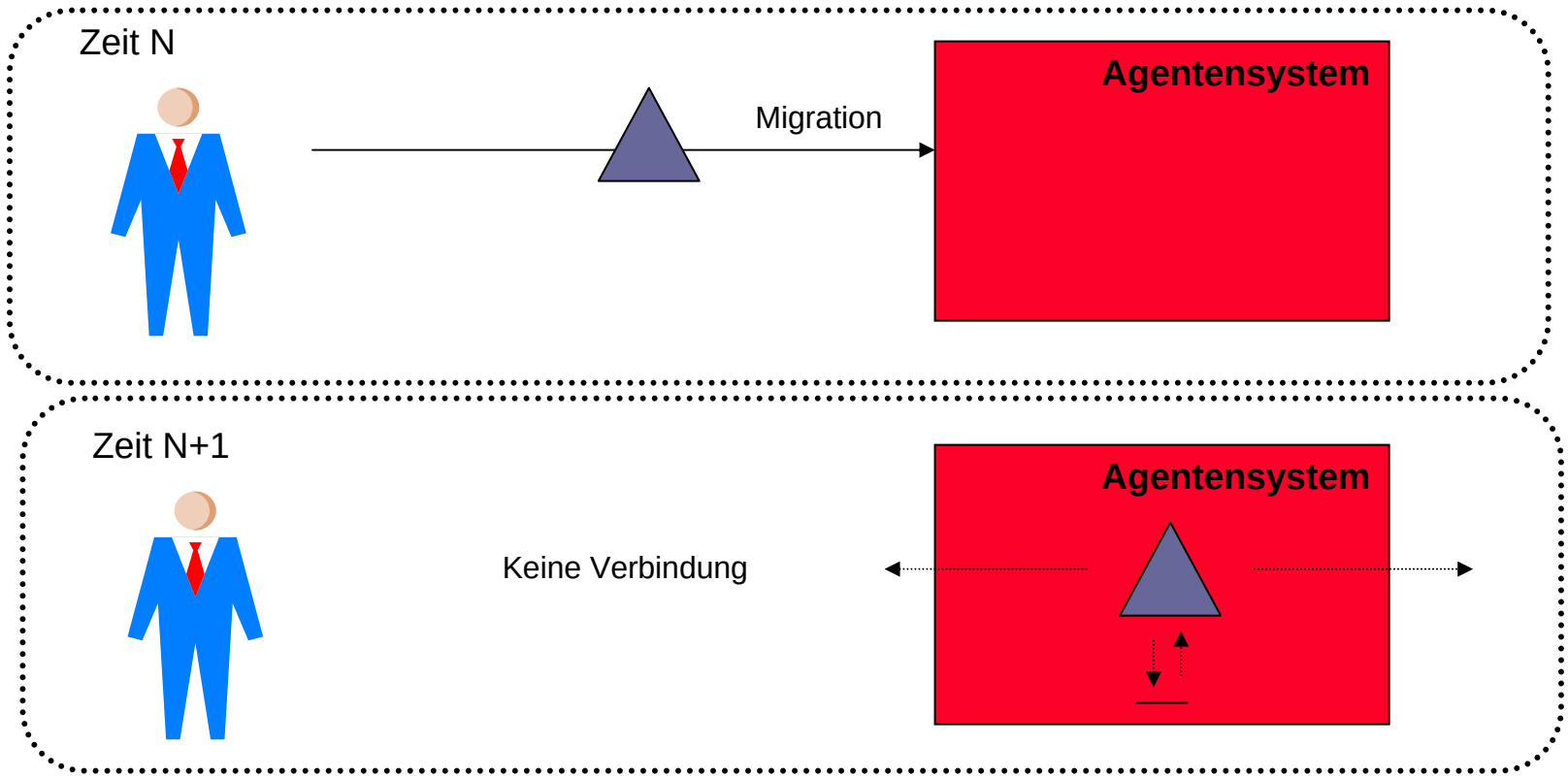
reaktiv

proaktiv

Weitere Infos: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4,
http://www.fh-wedel.de/*~si/seminare/ss04/Termine/Themen.html, erreichbar über* ***Archiv/iw***

Basistechnologie: Agentenorientierte Software

Agenteneigenschaft: Autonomie



Der Agent entscheidet selbstständig anhand bestimmter Kriterien über seine nächste Aktion (Problem: Kontrollmöglichkeiten).

Quelle: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4

Basistechnologie: Agentenorientierte Software

Agenteneigenschaft: Soziale Kompetenz

Agent kann mit anderen

- Agenten
- (Agenten-)Systemen
- Menschen

kommunizieren / kooperieren

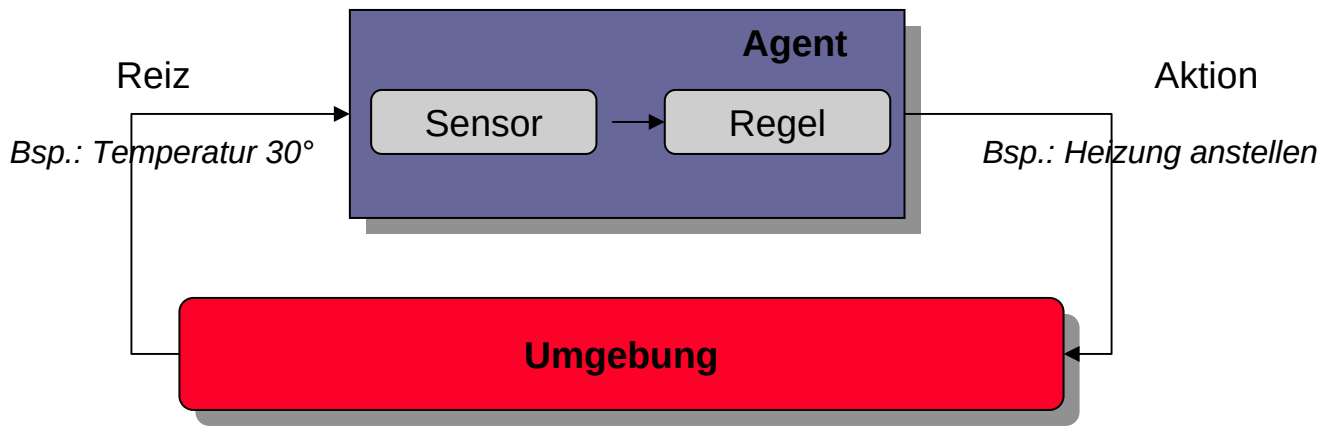
Vorraussetzungen:

- Gemeinsame Protokolle
- Gemeinsame Sprache / Schnittstellen
- Gemeinsame Ontologien

Quelle: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4

Basistechnologie: Agentenorientierte Software

Agenteneigenschaft: Reaktivität



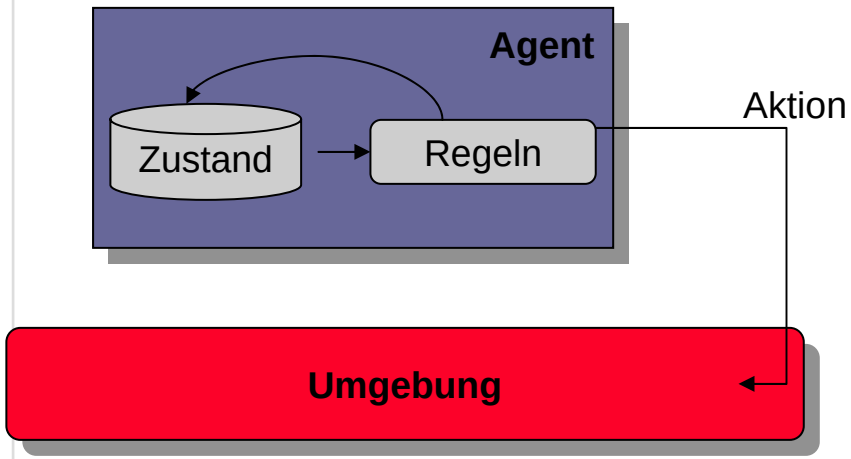
Der Agent reagiert auf Änderungen seiner Umgebung (engl. Environment)

Vorraussetzung: Existenz von Sensoren und Regeln

Quelle: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4

Basistechnologie: Agentenorientierte Software

Agenteigenschaft: Proaktivität (Zielgerichtetheit)



Agenten reagieren nicht nur auf Reize der Umgebung, sondern besitzen internen Zustand und sind zu zielgerichtetem Planung und Handeln fähig.

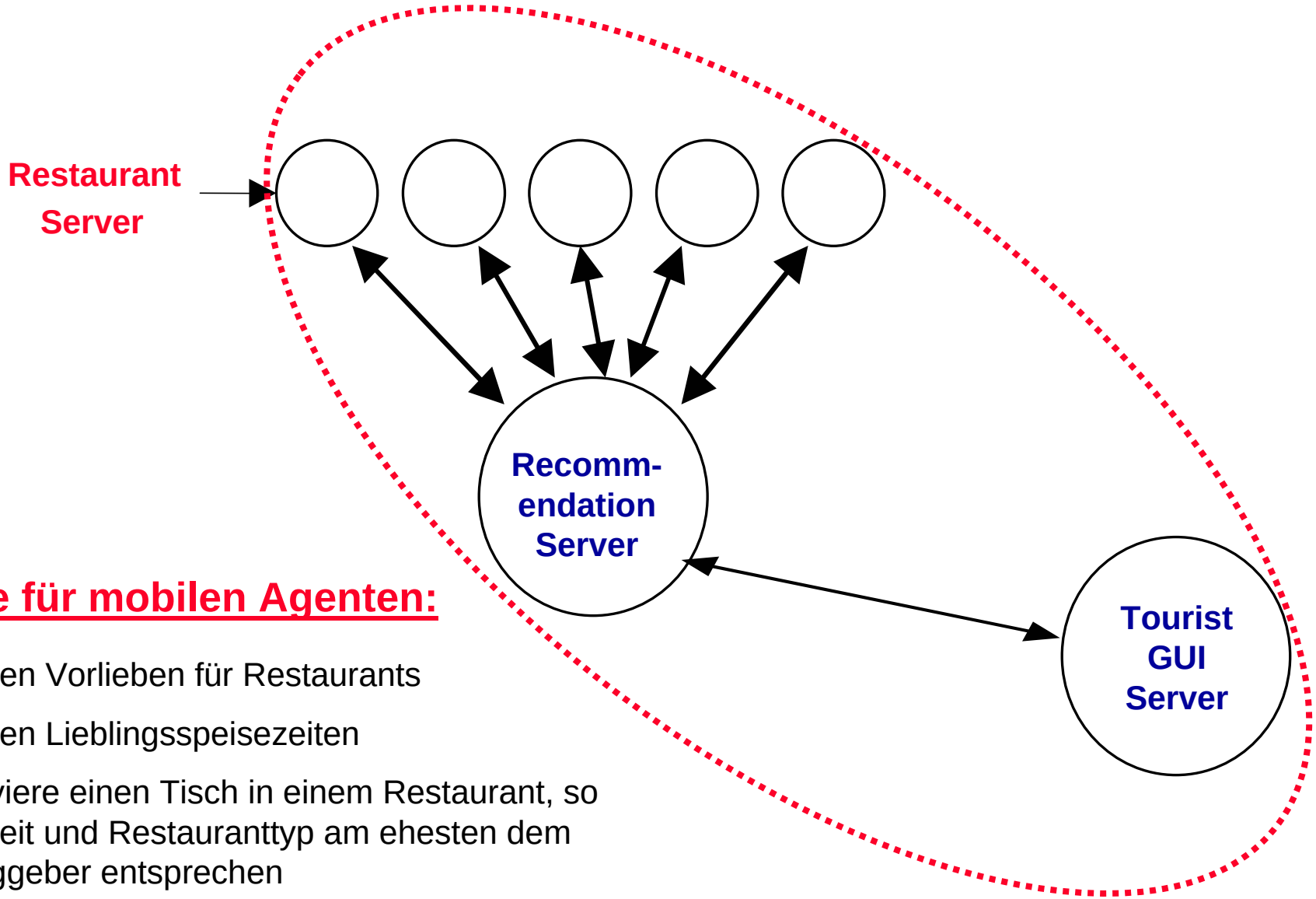
=> Sie ergreifen die **Initiative**

*„The difference between an automation and an agent is a somewhat like the difference between a dog and a butler. If you send your dog to buy a copy of the New York Times every morning, it will come back with its mouth empty if the news stand happens to have run out one day. In contrast, the butler will probably take the **initiative** to buy you a copy of the Washington Post, since he knows, that sometimes you read it instead."*

Le Du

Quelle: Seminarvorbereitung von Matthias Rohr, SS 2004, Nr. 4

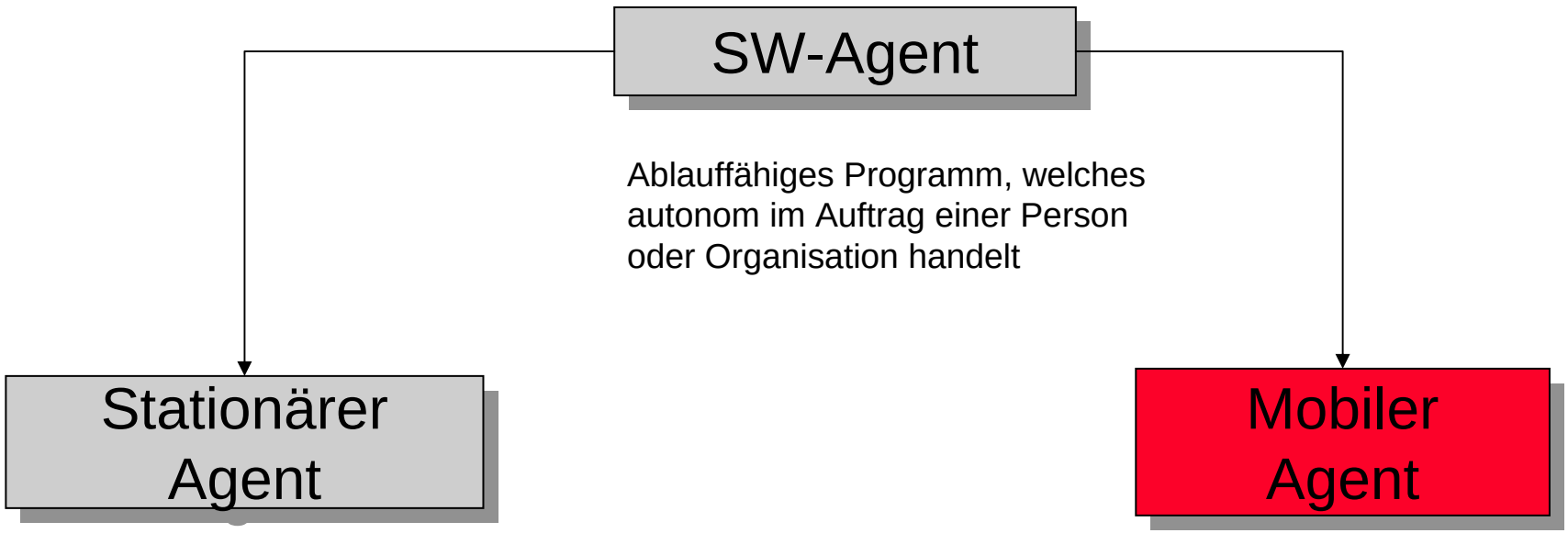
Verteiltes Touristeninformationsystem



Aufgabe für mobilen Agenten:

- Gegeben Vorlieben für Restaurants
- Gegeben Lieblingsspeisezeiten
- Reserviere einen Tisch in einem Restaurant, so dass Zeit und Restauranttyp am ehesten dem Auftraggeber entsprechen

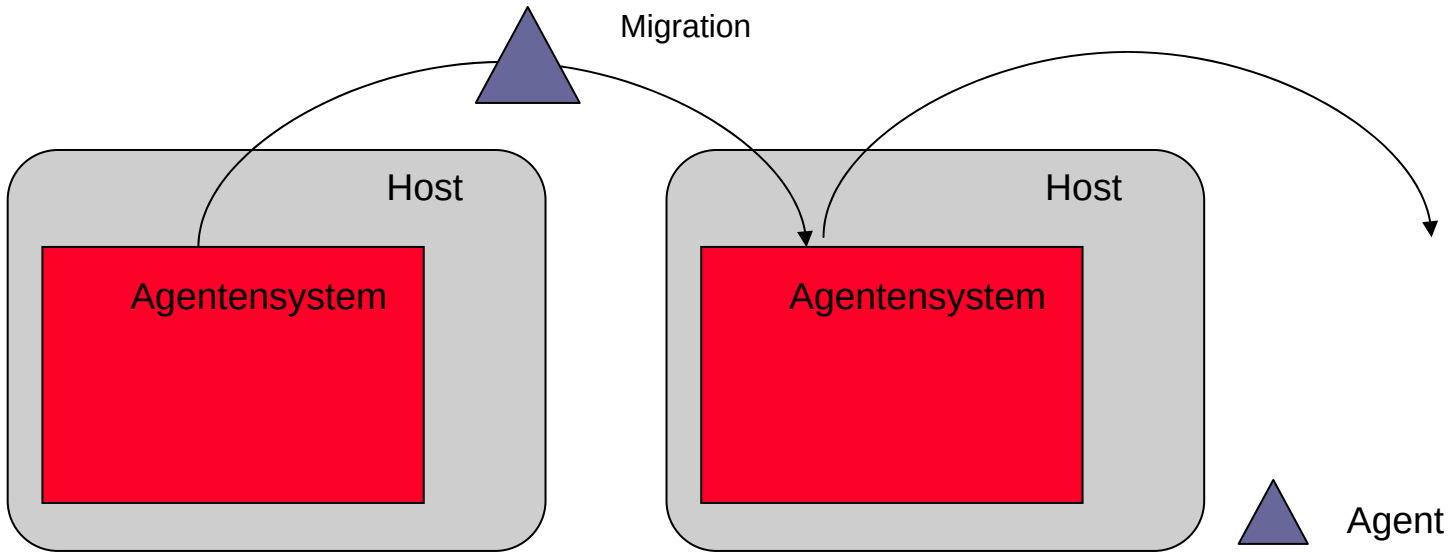
Basistechnologie: Agentenorientierte Software



Quelle: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4

Basistechnologie: Agentenorientierte Software

SW-Eigenschaft: Mobilität



Agent kann zwischen Agentensystemen wechseln (migrieren)

Voraussetzungen:

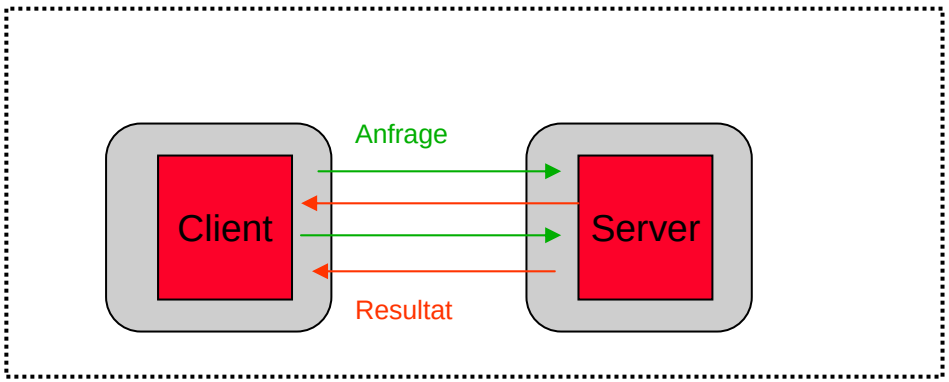
- Agenteninstanz kann in Datenfluss umgewandelt werden (Serialisierbarkeit)
- Die Zielsystem kann den Code des Agenten empfangen und interpretieren

Quelle: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4

Basistechnologie: Agentenorientierte Software

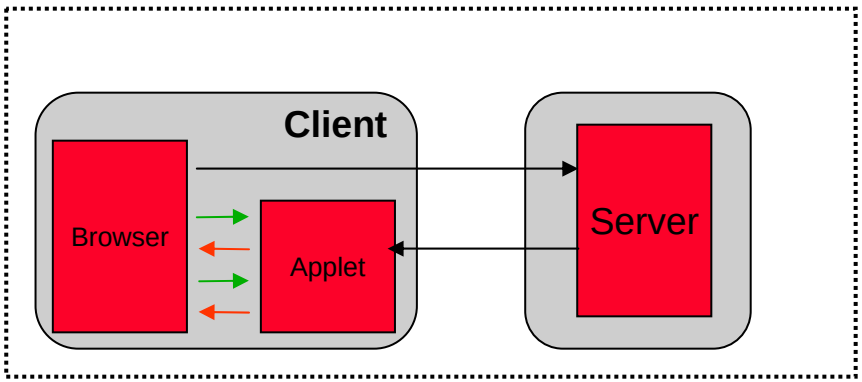
Mobile Agenten als Softwareparadigma: Abgrenzung

Client Server (CS)



- Funktionsweise wird vom Server festgelegt
- Ressourcenverbrauch beim Server

Code on Demand (CoD),
z.B.Java Applets



- Funktionsweise wird vom Server festgelegt
- Ressourcenverbrauch beim Client

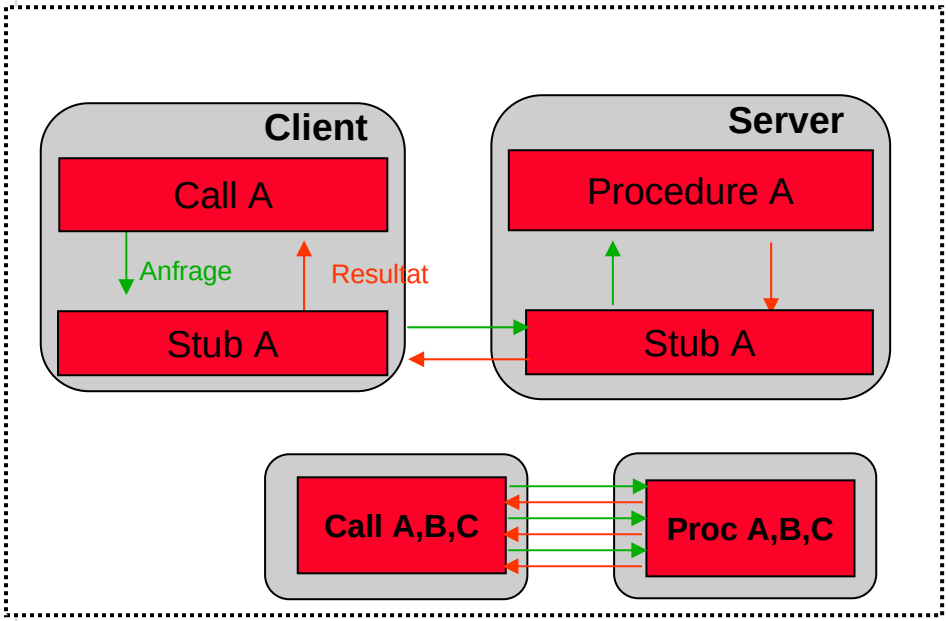
Quelle: Seminarvortrag und Ausarbeitung von Matthias Rohr, SS 2004, Nr. 4

Basistechnologie: Agentenorientierte Software

Mobile Agenten als Softwareparadigma: Abgrenzung

Remote Evaluation (REV)

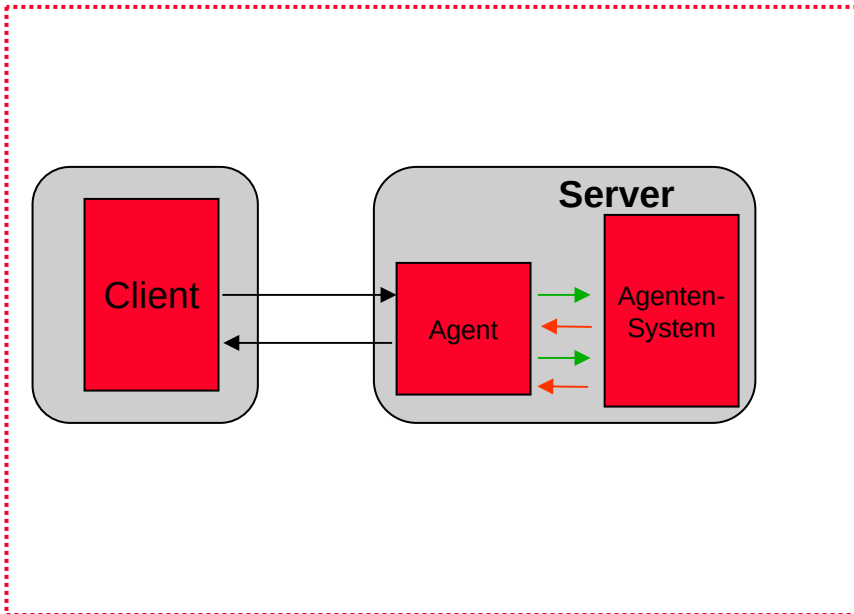
z.B. RPC, RMI, Corba



- Funktionsweise wird vom Server festgelegt
- Client behandelt Funktion wie eine eigene
- Ressourcenverbrauch hauptsächlich Server

Mobile Agent (MA)

z.B. Aglets, Voyager, Grasshopper

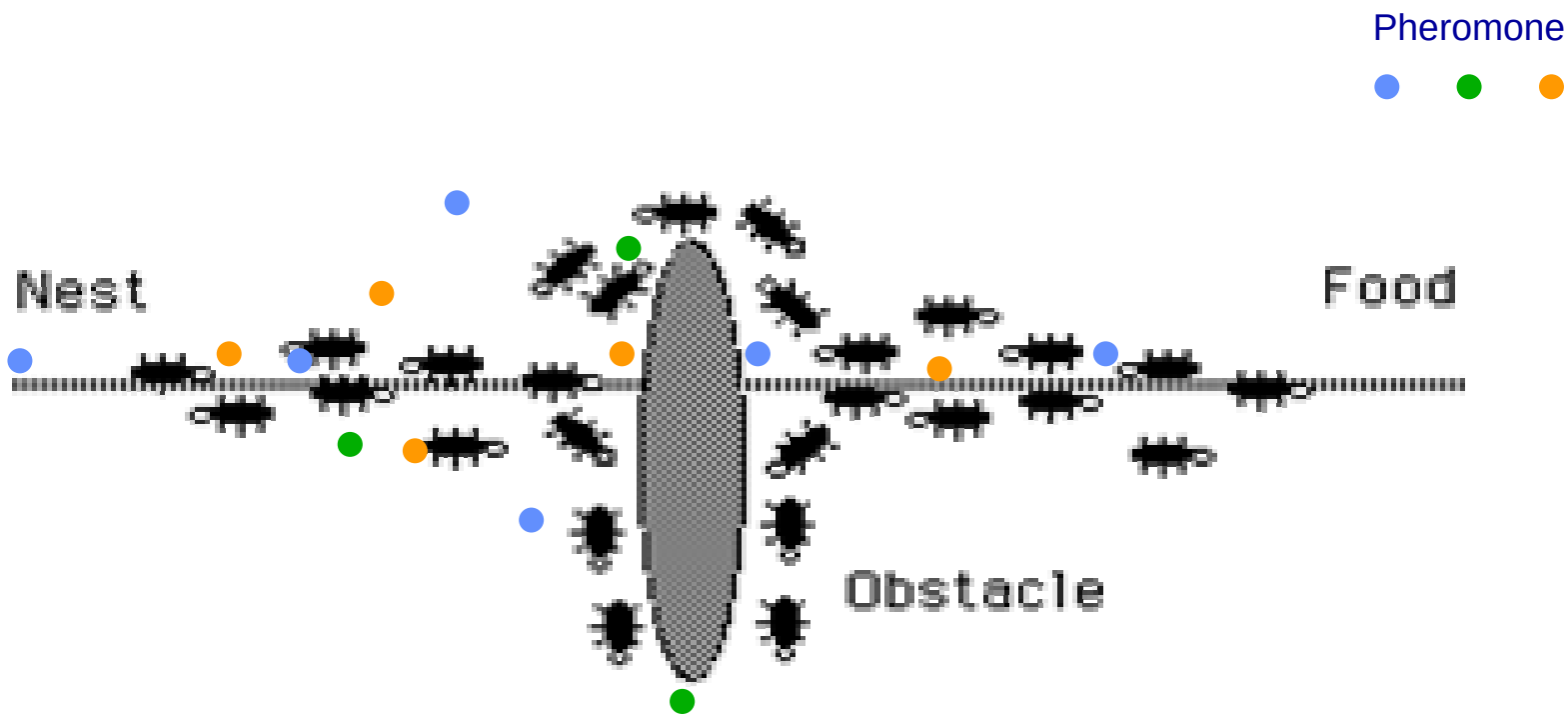


- Funktionsweise wird vom Client festgelegt
- Ressourcenverbrauch beim Server

Basistechnologie: Schwarmintelligenz

Pheromonbasierter Ansatz

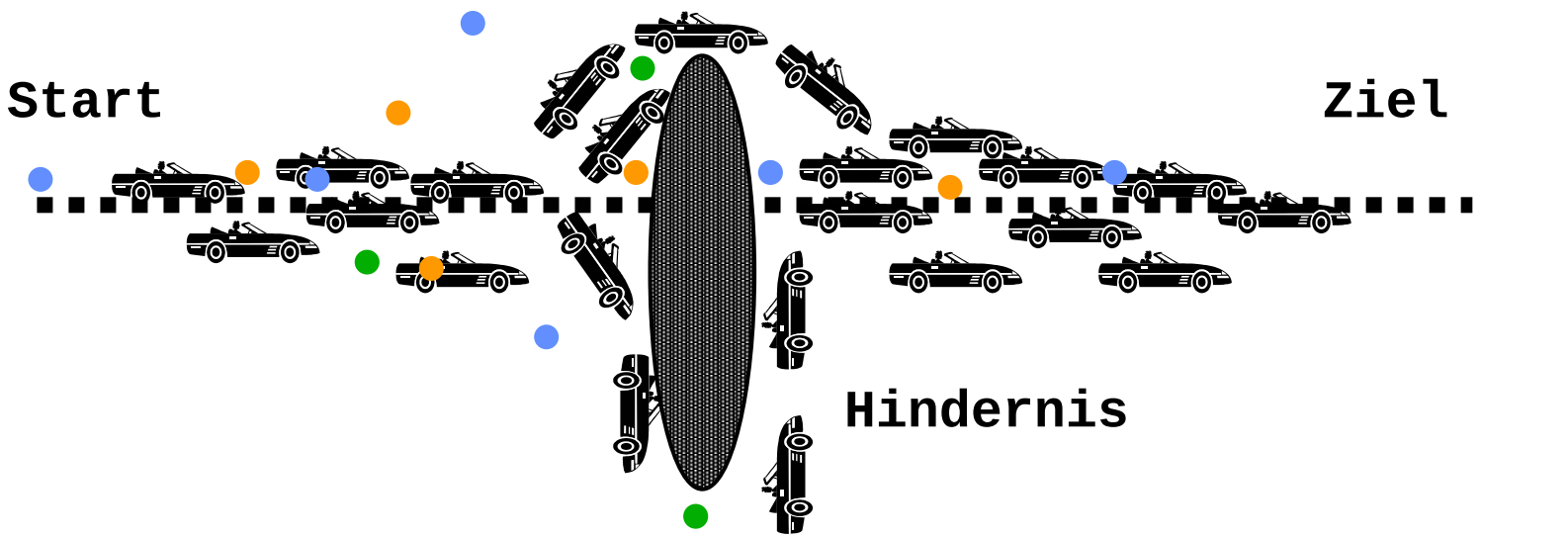
Ameisen auf Futtersuche



Basistechnologie: Schwarmintelligenz

Pheromonbasierter Ansatz

Analogon:
Autos auf Routensuche



Anwendung: Dynamische Verkehrsnavigation

Anwendungsgebiet: Technische Diagnose

Schilderung der historischen Entwicklung der KI-Technologien am Beispiel eines Anwendungsgebietes

Die folgenden Folien sind im Wesentlichen einem Vortrag entnommen zum Thema *Diagnose technischer Systeme* von Jakob Mauss (QTronic GmbH, Berlin)

Anwendungsgebiet: Technische Diagnose

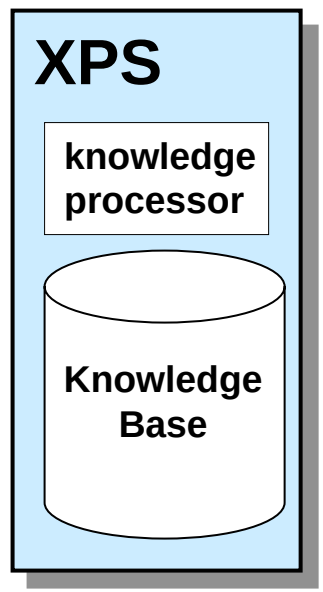
1970er: Diagnose = heuristische Klassifikation

Was ist Diagnose?

Diagnose ist der Schluss von beobachteten Symptomen auf klassifizierte Systemeigenschaften

Einsatzgebiete:

- Medizin
- industrielle Fertigungsprozesse
- Fahrzeugdiagnose
- Luft- und Raumfahrt



Anwendungsgebiet: Technische Diagnose

1970er: Diagnose = heuristische Klassifikation

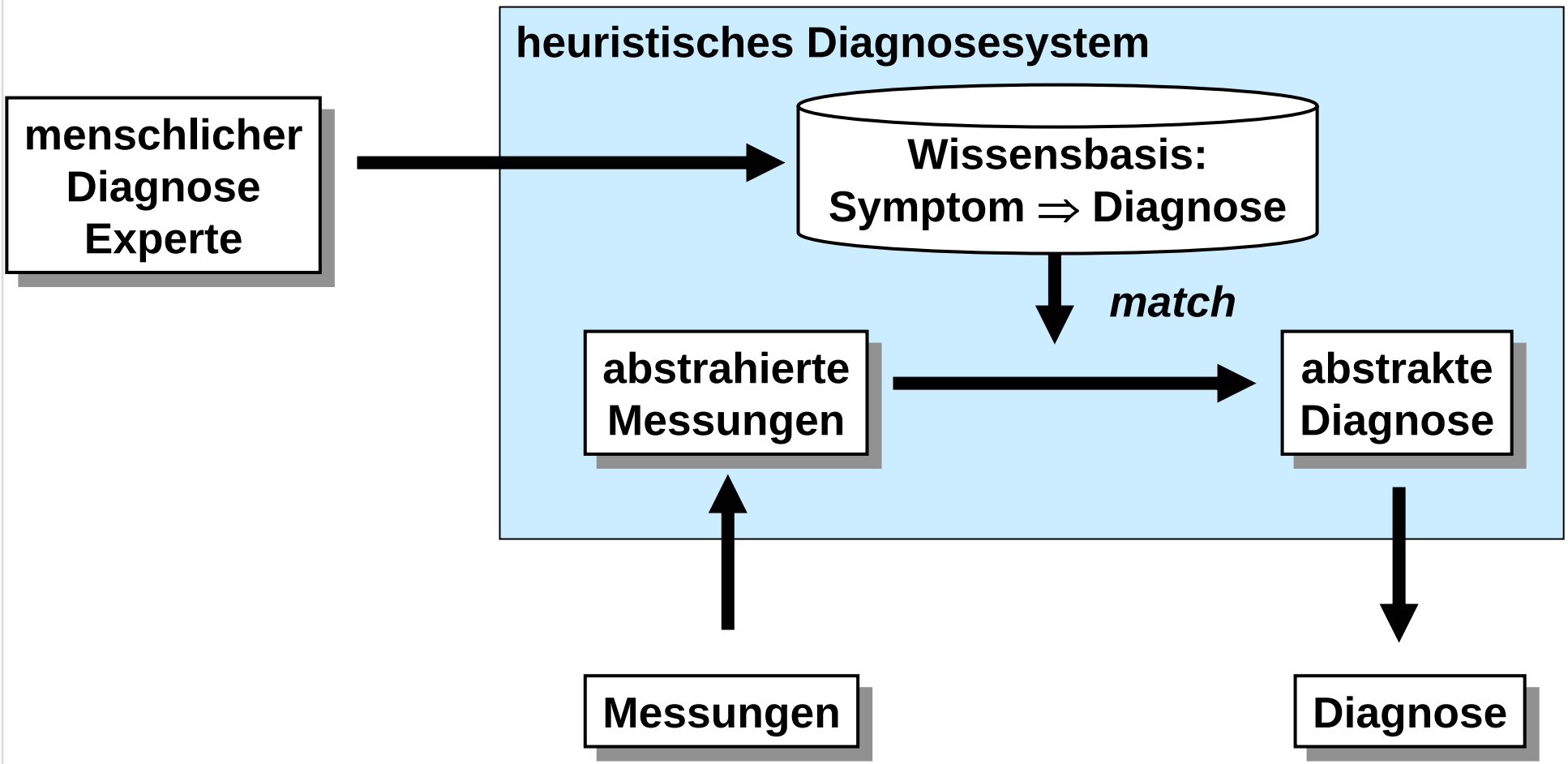
Was ist Technische Diagnose?

Gegeben:
- Ein technisches System (z.B. Auto, Zug)
- Beobachtungen (z.B. Messwerte, Fehlercodes, Fahrerbeanstandung), nicht nominal.

Aufgabe:
Bestimme,
- in welcher Weise das System defekt ist
- genau genug, dass das nominale Verhalten des Systems wiederhergestellt werden kann.

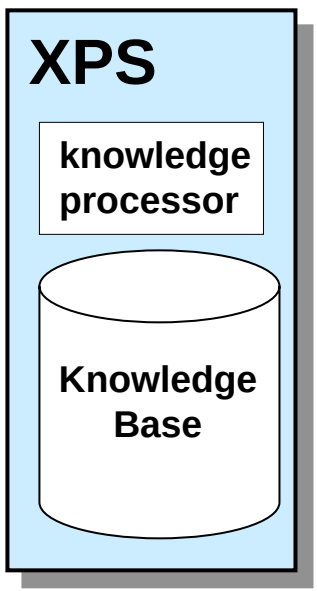
Anwendungsgebiet: Technische Diagnose

1970er: Diagnose = heuristische Klassifikation



Anwendungsgebiet: Technische Diagnose

1970er: Diagnose = heuristische Klassifikation



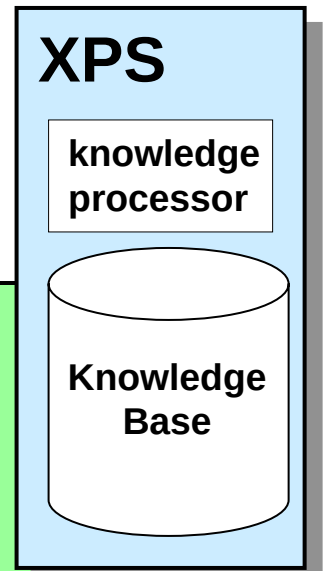
verschiedene Realisierungen des Konzeptes:

- regelbasierte Diagnose:
  sicher, probabilistisch, fuzzy, ...
- Diagnose mit Entscheidungsbäumen

- fallbasierte Diagnose
  (case-based reasoning CBR)
- Klassifikation mit neuronalen Netzen

Anwendungsgebiet: Technische Diagnose

1970er: Diagnose = heuristische Klassifikation



Stärken:

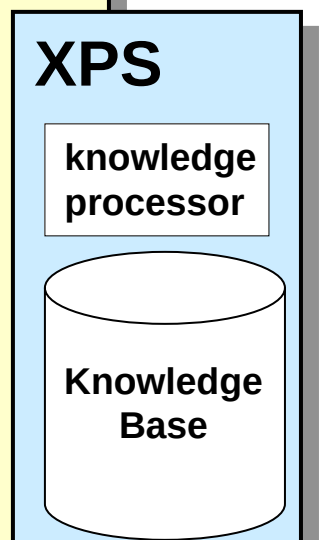
- breit anwendbar
- viele Werkzeuge verfügbar
- Erfahrungen mit vielen unterschiedlichen Anwendungen verfügbar

Anwendungsgebiet: Technische Diagnose

1970er: Diagnose = heuristische Klassifikation

Schwächen:

- Aufbau und Wartung der Wissensbasis:
  knowledge-acquisition bottleneck
  - Verfügbarkeit menschlicher Experten
  - Erweiterung der Wissensbasis um neue Fälle
  - Objektivität des erfassten Wissens
  - Vollständigkeit
  - Wiederverwendung einer Wissensbasis
- Wirtschaftlichkeit der Softwareerstellung
- Diagnose unvorhergesehener Fehler
  schwierig / unmöglich
- Diagnose von Mehrfachfehlern schwierig
- Erklärung generierter Diagnosen



Anwendungsgebiet: Technische Diagnose

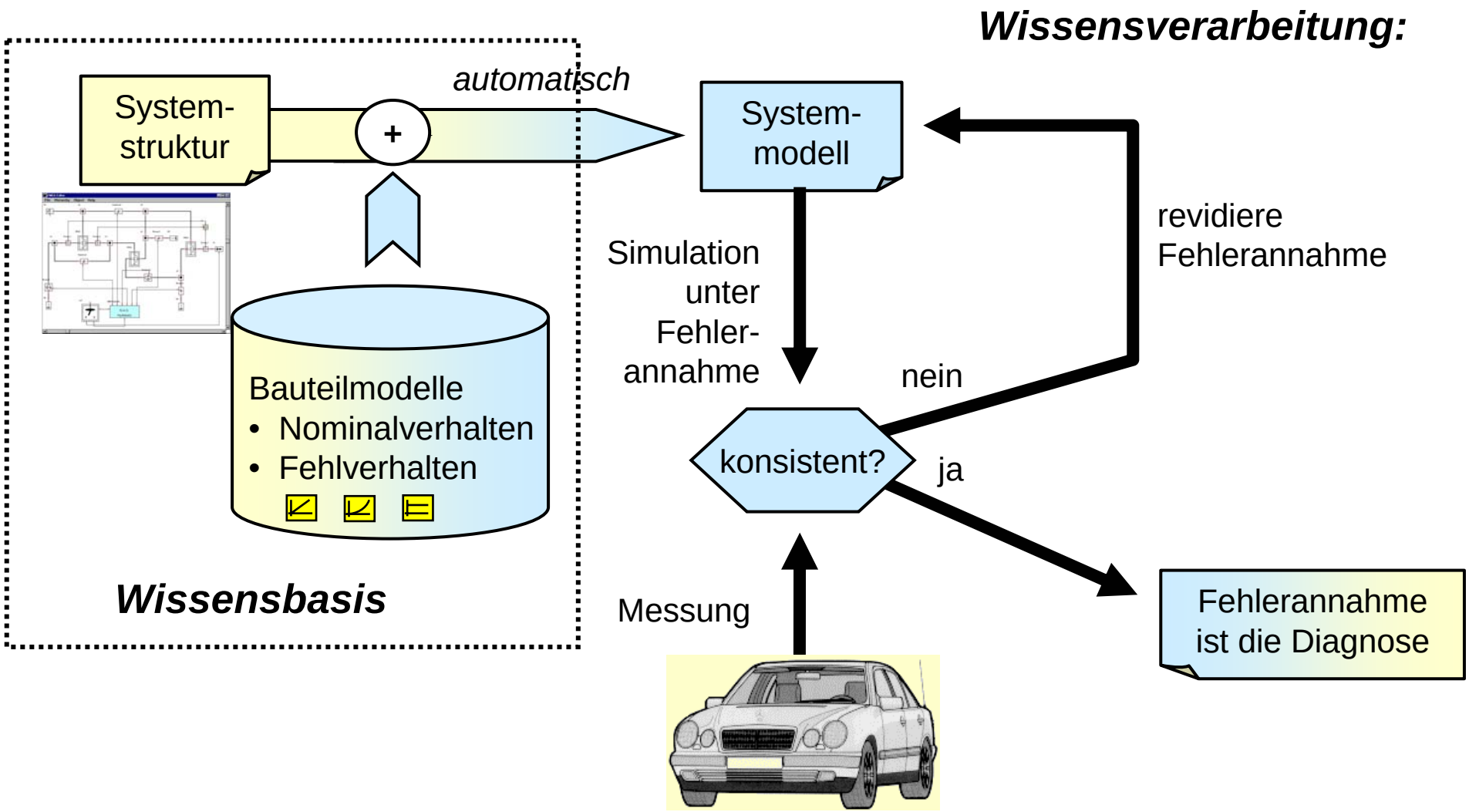
ca.1985: Diagnose = modellbasiertes Schließen

Ideen:

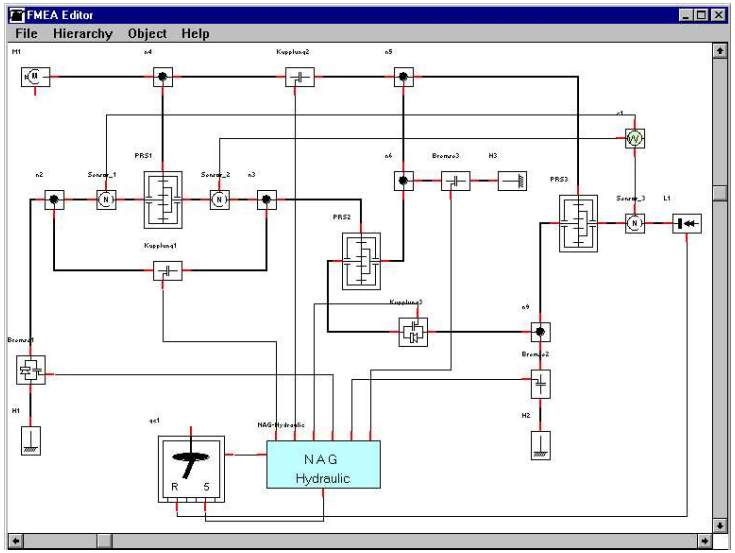
- Erfahrungswissen $\Rightarrow$ Modell des zu diagnostierenden Systems
- objektives Modell, unabhängig von Diagnoseaufgabe
- löst viele der Probleme des heuristischen Ansatzes
- Schlagworte
  - Second Generation XPS
  - tiefes Wissen
  - Schliessen ‘from first principles‘
  - modellbasiert

Anwendungsgebiet: Technische Diagnose

1980er: Diagnose = modellbasiertes Schließen



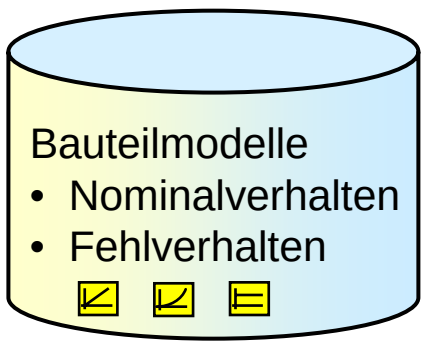
Wissenserwerb bei der modellbasierten Diagnose



<u>Systemstruktur</u>:

Welche Komponenten von welchem Typ sind wie miteinander verbunden ?

➔ ***erhältlich aus CAD-Daten***



<u>Komponentenmodelle</u>:

Wie ist die Abhängigkeit zwischen den Werten, die an den Verbindungs-punkten einer Komponente anliegen ?

➔ ***pro Komponententyp einmal zu modellieren***

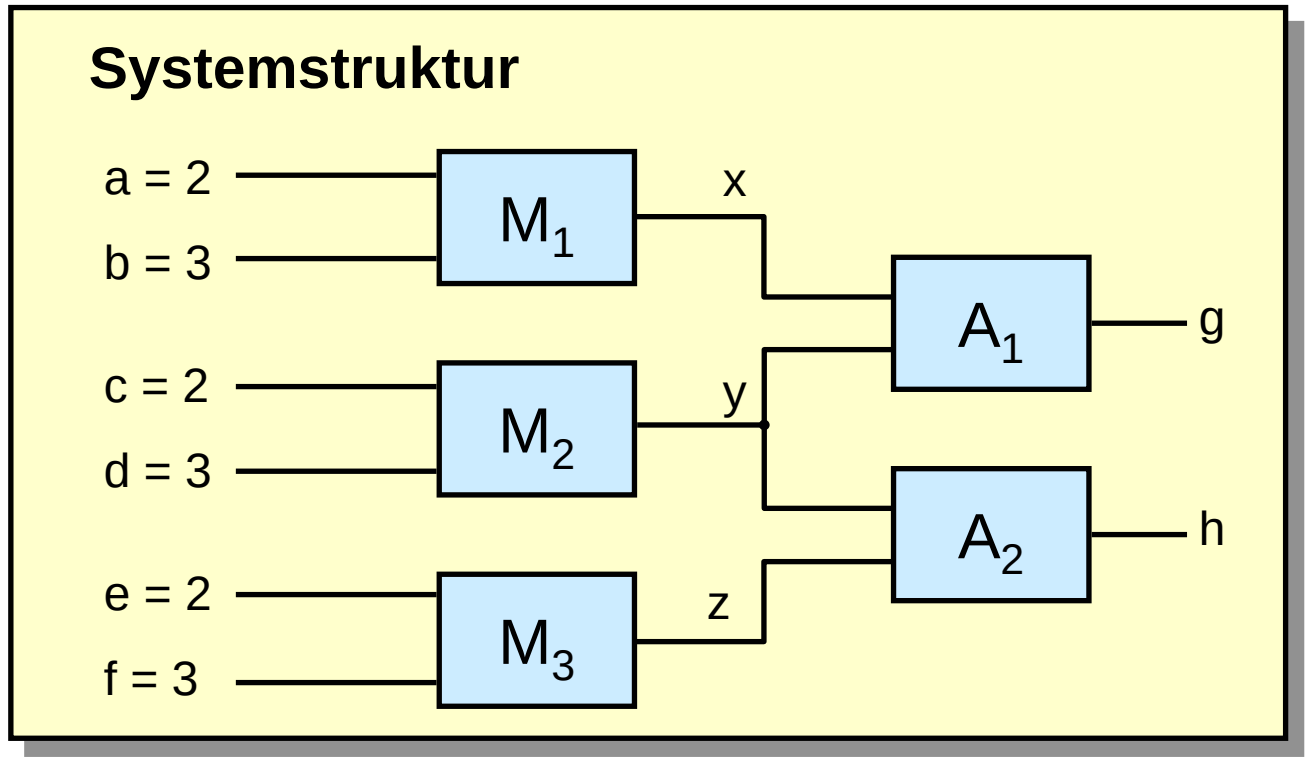
➔***Modell ist wiederverwendbar für alle Systeme, in denen Komponenten dieses Typs enthalten sind***

Die GDE:
1987: *Der* Prototyp für die modellbasierte Diagnose

Problem:
- ‚brute-force' Simulation ***aller*** Fehlerannahmen kombinatorisch nicht realisierbar

Idee: General Diagnostic Engine GDE, deKleer & Williams 1987
- intelligente Suche im Raum der möglichen Fehlerannahmen
- nutzt inkonsistente Annahmen zum Verkleinern des Suchraums
- Prinzip: konfliktgesteuerte Suche

GDE - Beispiel

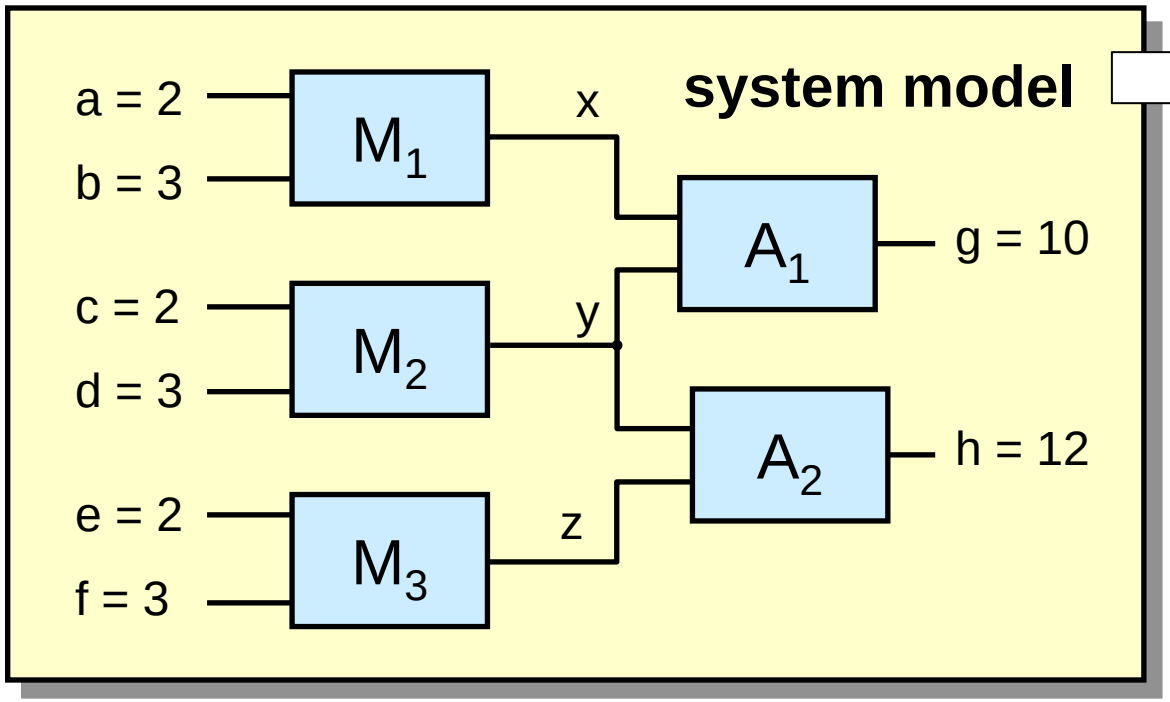


Komponentenmodelle

- Multiplizierer: mode=ok $\Rightarrow$ out = $in_1 * in_2$
- Addierer: mode=ok $\Rightarrow$ out = $in_1 + in_2$

Messungen $g = 10 \wedge h = 12$

GDE - Beispiel



simulation

x = 6 {M1}
y = 6 {M2}
z = 6 {M3}
g = 12 {M1 M2 A1}, g =10
y = 4 {M1 A1}
h = 10 {M1 A1 A2 M3}, h = 12
y = 6 {A2 M3}

two conflicts

M1	M2	M3	A1	A2
X	X		X	
X		X	X	X

diagnoses:
single-fault M1
single-fault A1
double fault M2 M3
:

Modellbasierte Diagnose: Entwicklung seit 1987

GDE - deKleer & Williams 87

Sherlock - deKleer & Williams 89
explicit fault models, probabilities

GDE+ - Struss & Dressler 89
explicit fault models, ordering

roδon
R.O.S.E. Informatik

many years of research ...
- speed-up: focusing techniques
- modeling assumptions, model switching
- dynamics, e.g. finite-state machine models
- real applications: power networks, vehicles, ...

Raz'r 1.0, 1997
commercial
product (OCC'M)

RA - Williams et al. 98
Remote Agent flight software
NASA AMES & JPL
CommonLisp

MDS 1.5, 2000
Daimler Research
Smalltalk

Modellbasierte Diagnose

Stärken (gegenüber dem heuristischem Ansatz):

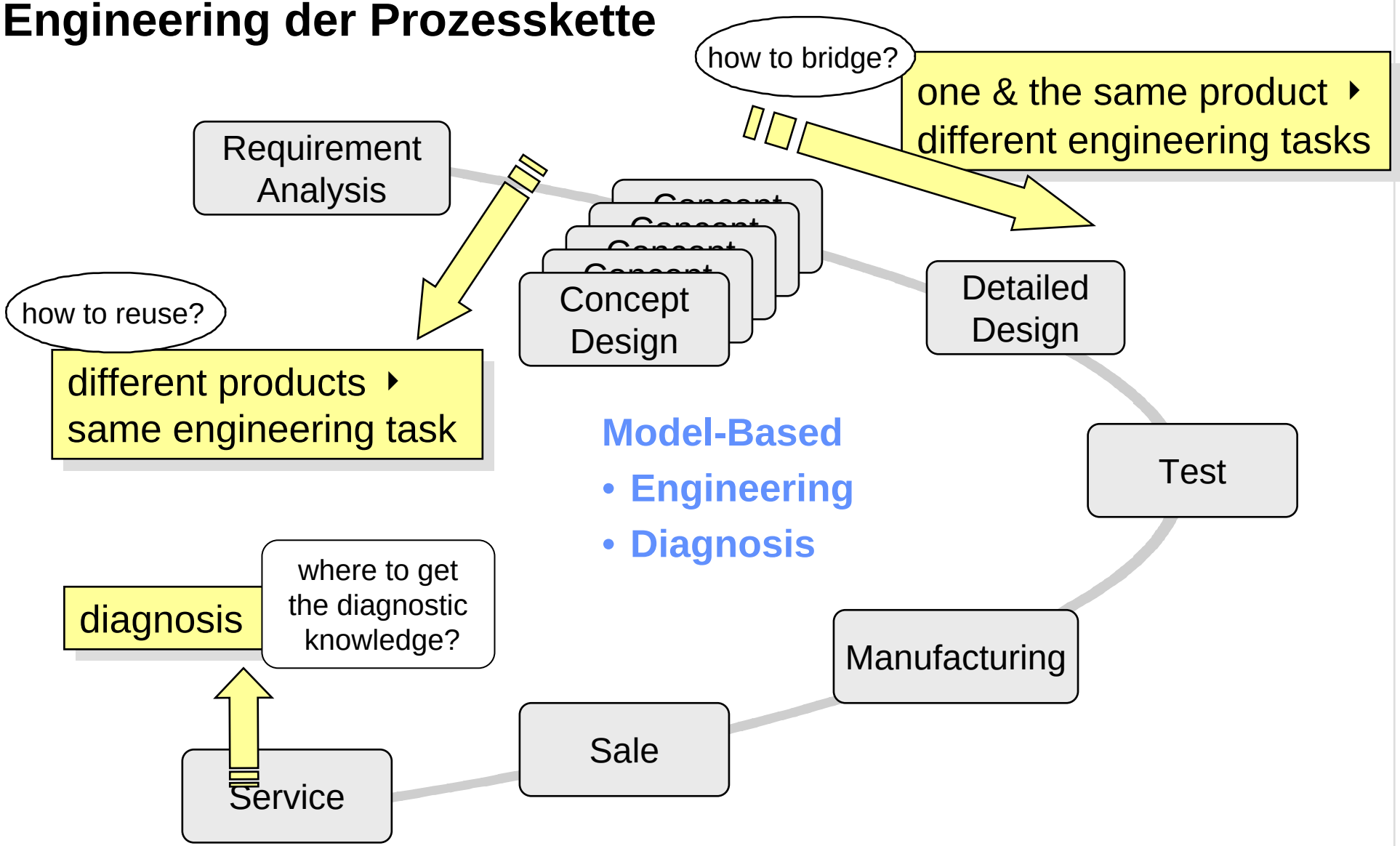
- wiederverwendbare Bauteilmodelle
- Aufbau und Wartung eines Systemmodells: einfach
- Systemmodell: objektiv, nachprüfbar, lokal modifizierbar
- neue Systeme auch ohne Erfahrung diagnostizierbar
- Mehrfachfehler und dynamische Systeme
- nachvollziehbare Erklärung von Diagnosen

Schwächen:

- Aufbau wiederverwendbarer Bauteilmodell-Bibliotheken: schwierig, verborgene Modellierungsannahmen
- rechenintensiv
- bisher wenige Werkzeuge und reale Anwendungen verfügbar

Wirtschaftliche Relevanz von KI-Techniken

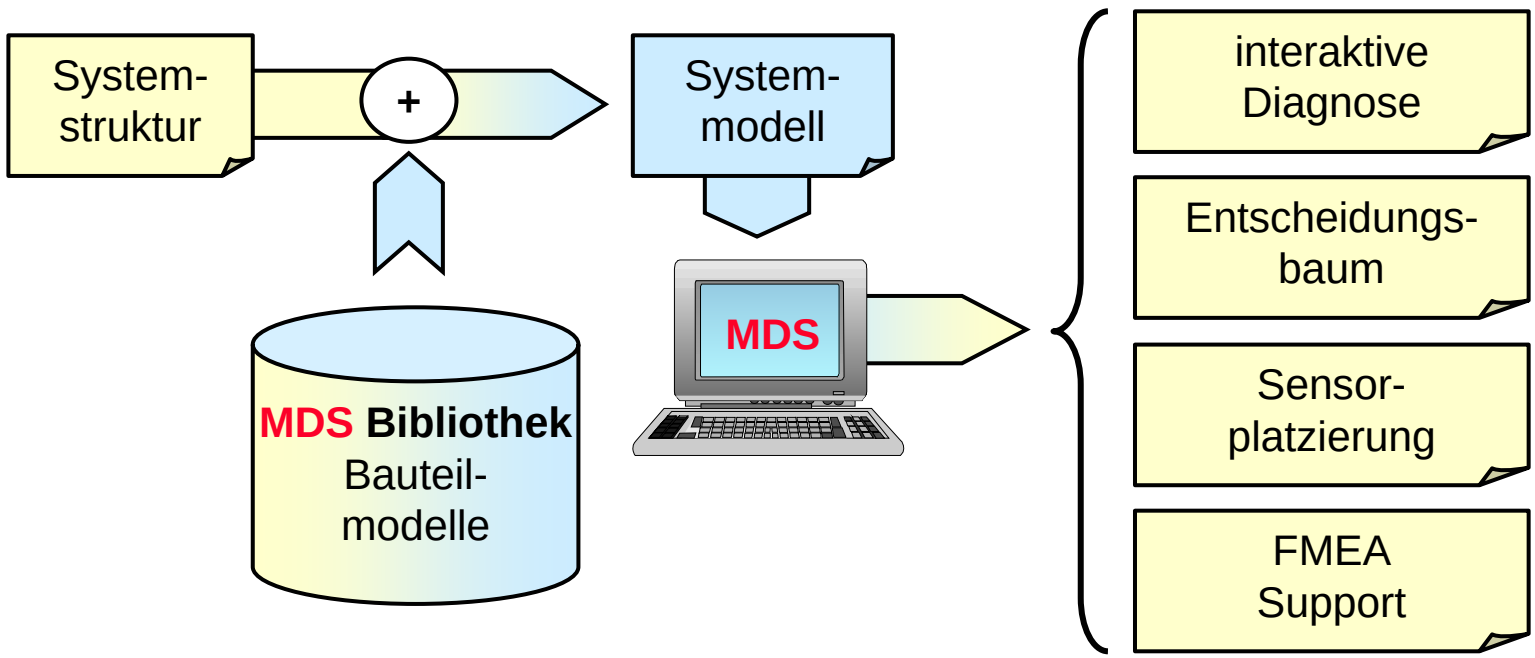
Engineering der Prozesskette



Wirtschaftliche Relevanz von KI-Techniken

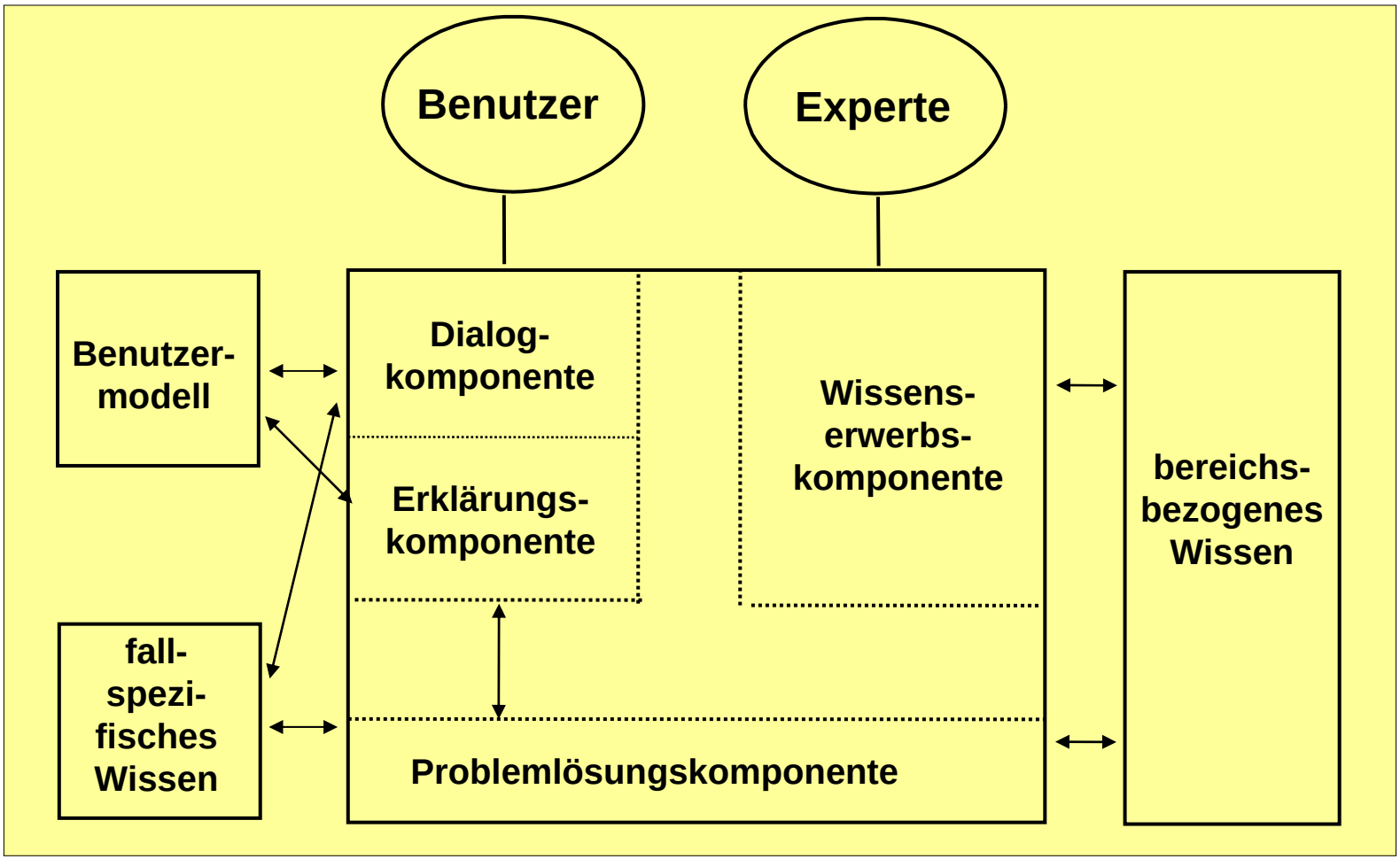
Wiederverwendbarkeit

MDS: Modellbasiertes Diagnose- und Analysesystem



Basistechnologie: Wissensbasiertes System

Architektur XPS (klassisch)



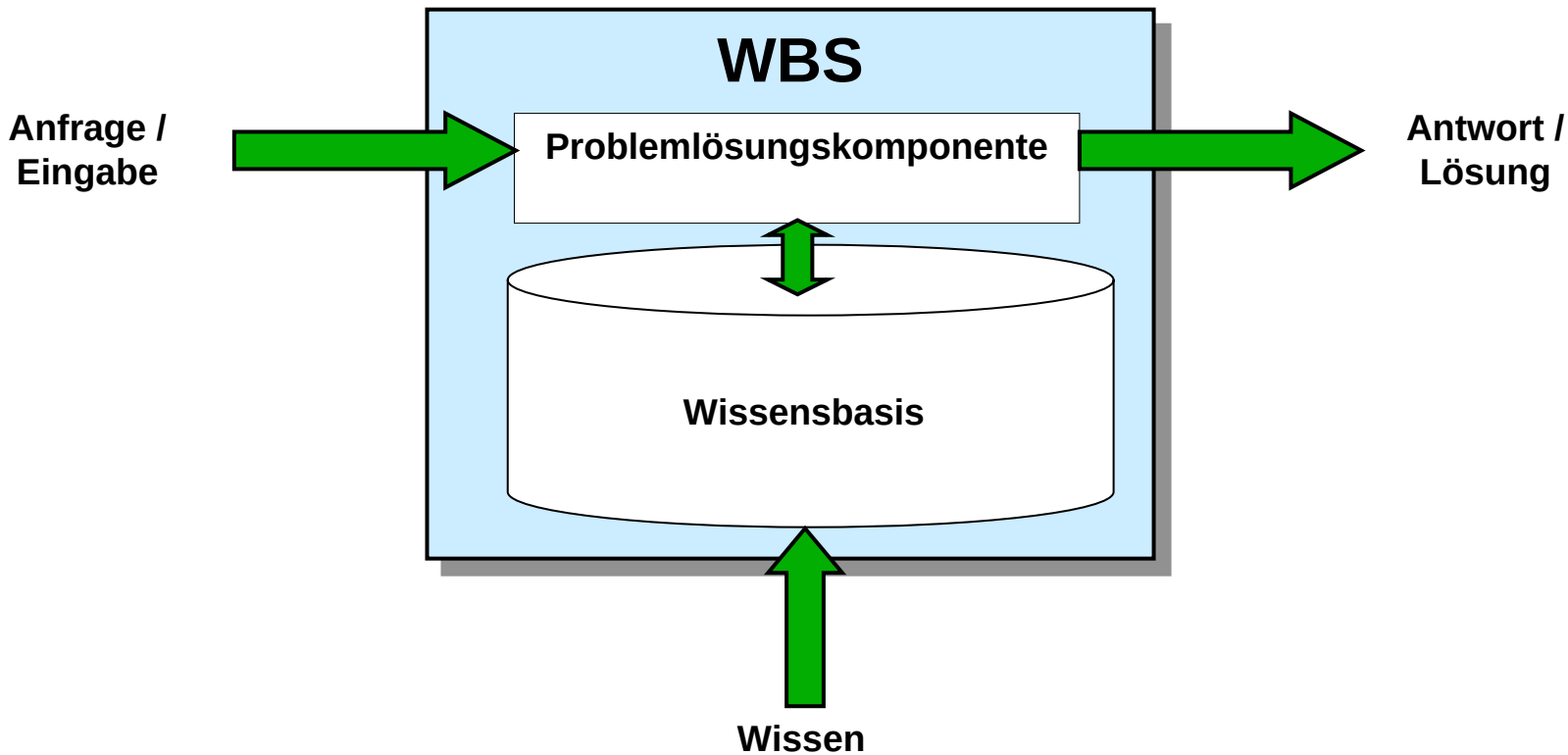
aus *Wissensbasierte Systeme*, Vorlesung 2, FH Deggendorf, 2004

Basistechnologie: Wissensbasiertes System

Wissen + Problemlösungskomponente = WBS

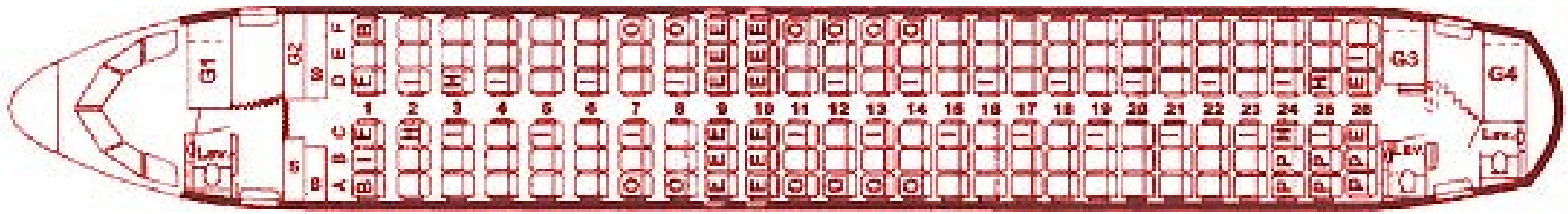
Daten **Verarbeitungsregeln**

Architektur WBS (allgemeiner)



Anwendungsgebiet: Konfiguration

Kabinenlayout für Passagierflugzeuge



Platzierung der Kabineneinrichtung (Sitze, Küchen, Toiletten, etc.) unter Berücksichtigung von:

- Kundenwünschen
- Technischen Möglichkeiten
- Legalen Beschränkungen
- Optimalitätskriterien

Grundlage der KI: Logik

Intelligente Lebewesen können auch sehr allgemeines Wissen verarbeiten:
Je allgemeiner, desto intelligenter

Allgemeine Verarbeitungsfähigkeiten benötigen allgemeine Beschreibungsmöglichkeiten für die Daten und Verarbeitungsregeln

Die allgemeinste objektive Beschreibungssprache
ist die Sprache der mathematischen Logik.

Daher arbeiten traditionelle KI-Verfahren mit logischen Beschreibungssprachen.

Probleme:

- **Aufgaben liegen häufig anders formuliert vor.**
- **Allgemeinheit geht auf Kosten der Effizienz.**

Ziele und Methoden der KI

Ziele für die SW-Lösungen

- **Allgemeinheit**
- **Flexibilität, Erweiterbarkeit**
- **Erklärbarkeit der Antworten**

Häufig eingesetzte Methoden

- **Logische Programmiersprachen (PROLOG)**
- **Objektorientierte Programmiersprachen (Smalltalk)**
- **Funktionale Programmiersprachen (Lisp)**
- **Verteilte Systeme (Neuronale Netze, Multiagentensysteme)**
- **Begriffswelten (Ontologien)**

Definitionen von KI

Systeme, die wie Menschen denken	Systeme, die rational denken
„Die aufregende und neuartige Anstrengung, Computern das Denken beizubringen, ... KI will die Sache selbst: Maschinen mit Verstand, im vollen und wörtlichen Sinne.“ (Haugeland, 1985) „[Die Automatisierung von] Aktivitäten, die wir dem menschlichen Denken zuordnen, Aktivitäten wie beispielsweise Entscheidungsfindung, Problemlösung, Lernen ...“ (Bellman, 1978)	„Die Studie mentaler Fähigkeiten durch die Nutzung programmiertechnischer Modelle.“ (Charniak und McDermott, 1985) „Die Studie der Programmtechniken, die es ermöglichen, wahrzunehmen, logisch zu schließen und zu agieren.“ (Winston, 1992)
Systeme, die wie Menschen agieren	**Systeme, die rational agieren**
„Die Kunst, Maschinen zu schaffen, die Funktionen erfüllen, die, werden sie von Menschen ausgeführt, der Intelligenz bedürfen.“ (Kurzweil, 1990) „Die Studie, wie man Computer dazu bringt, Dinge zu tun, bei denen ihnen momentan der Mensch noch überlegen ist.“ (Rich und Knight, 1991)	„Computerintelligenz ist die Studie des Entwurfs intelligenter Agenten.“ (Poole et al., 1998) „KI ... beschäftigt sich mit intelligentem Verhalten in künstlichen Maschinen.“ (Nilsson, 1998)

Definitionen aus Russell / Norvig

Definitionen von KI

KI beschäftigt sich mit Problemen, die

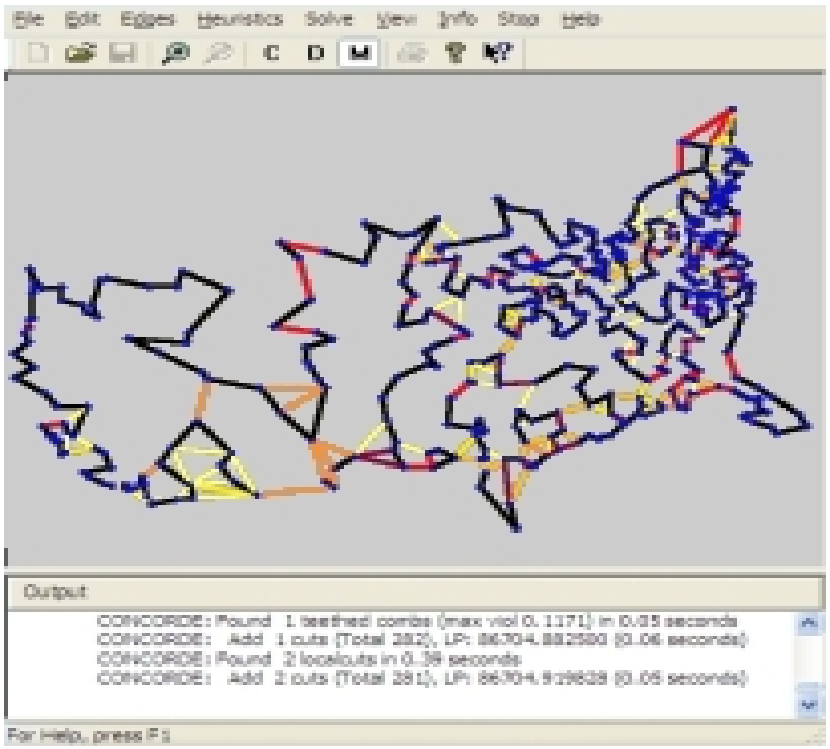
- **in der Praxis relevant sind.**
- **häufig nicht exakt spezifiziert werden können.**
- **NP-vollständig sind, wenn sie exakt spezifiziert werden können.**

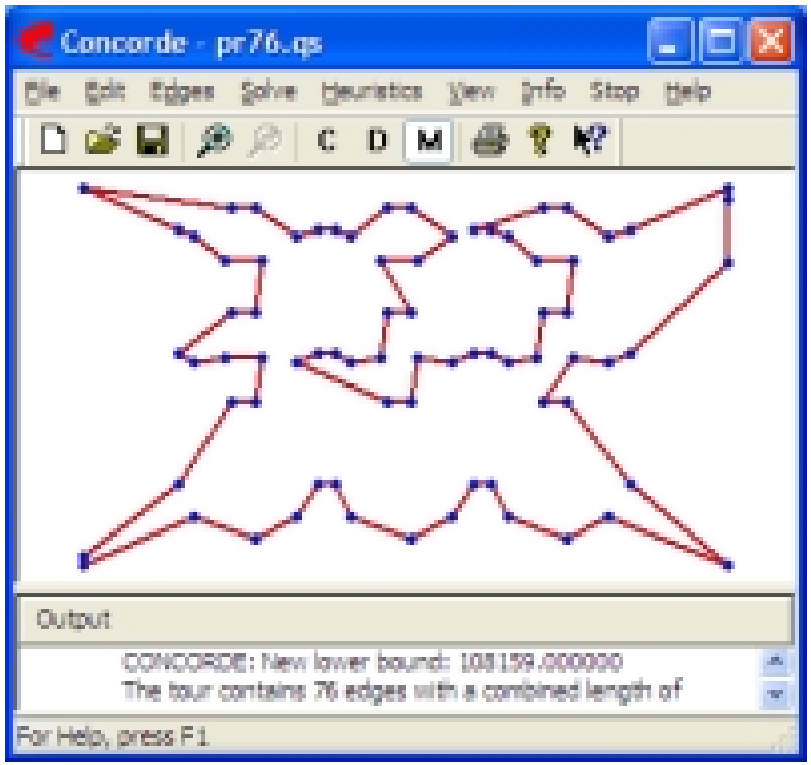
Definition iw

Das Traveling Salesman Problem (TSP)

Problem:

Finde zu einer gegebenen Menge von Städten mit gegebenen Entfernungen die kürzeste Rundreise, die jede Stadt genau einmal durchquert.





Quelle: http://www.tsp.gatech.edu//index.html

Zusammenfassung Kapitel 1

Anwendungsgebiete der KI:

- **Spiele, in denen die Maschine andere Spieler simuliert**
  - Rundenbasierte Strategiespiele
  - Echtzeit-Strategiespiele
  - Mehrbenutzer-Strategiespiele
  - Sportsimulationen
  - Entwicklungssimulationen
- **Resourcenverteilung**
  - Streckenbelegungsplan in Eisenbahnnetzen
- **Optimierungsprobleme mit dynamischen Parametern**
  - Verkehrsnavigation
- **Kundenberatung**
  - Touristeninformationssystem
- **Diagnose**
  - Medizinische Diagnose
  - Technische Diagnose

Zusammenfassung Kapitel 1

Basistechnologien der KI:

- **Agentenorientierte Software**
  - **verteilt**
  - **autonom**
  - **proaktiv**

- **Schwarmintelligenz**
  - **verteilt**
  - **statistisch**
  - **nebenläufige Aktualisierung**

- **Wissensbasierte Systeme**
  - **Trennung in Wissen und Inferenzmaschine**
  - **Intelligenter Wissenserwerb und Wissensrepräsentation**
  - **Hauptfokus: Wiederverwendung**

Zusammenfassung Kapitel 1

Der klassische Gegensatz von verschiedenen Forschergemeinden in der Informatik:

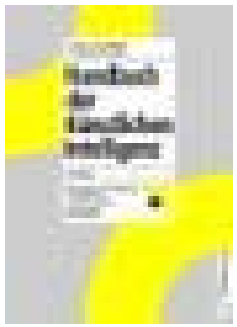
KI vs. Algorithmik

- **flexibel**
- **kundenorientiert**
- **exakt**
- **effizient**

Das muss kein Widerspruch sein!

Literatur

KI allgemein:



Günter Görz / Claus-Rainer Rollinger / Josef Schneeberger: *Handbuch der Künstlichen Intelligenz*
Oldenbourg 2000 (3. Auflage), ISBN 3-486-25049-3

Stuart Russell / Peter Norvig: *Artificial Intelligence: A Modern Approach*
Pearson 2003 (2. Auflage), ISBN 0-13-080302-2

spezielle Gebiete der KI:

Michael Wooldridge: *An Introduction to MultiAgent Systems*, Wiley 2002, ISBN 0-471-49691-X

Marco Dorigo / Thomas Stützle: *Ant Colony Optimization*, MIT Press 2004, ISBN 0-262-04219-3

Thomas Walther: *Dynamische Fahrzeugnavigation auf Basis von Ameisenkolonien*, Master-Thesis WS 2005/2006