

Wissensbasierte Systeme

Vorlesung 9 vom 15.12.2004
Sebastian Iwanowski
FH Wedel

Wissensbasierte Systeme

1. Motivation
2. Prinzipien und Anwendungen
3. Logische Grundlagen
4. Suchstrategien
5. Modellbasierte Diagnose
 - Kandidatengenerierung
 - Konfliktgenerierung
 - Zusammenspiel von Wertpropagierung, Konfliktgenerierung, Kandidatengenerierung
 - ➔ Weitere Funktionalitäten von MDS
 - Komponentenmodellierung
6. Andere Diagnosemethoden
7. Weitere Wissensrepräsentationsformen
8. Bewertung wissensbasierter Systeme

Vom ACS zur kompletten Diagnosesoftware

Was kann das ACS für den Anwender leisten?

Eingabe:

- Einstellung bestimmter Werte im System
- Beobachtung davon abhängiger Werte im System

Ausgabe:

- Mehrere Diagnosen folgender Art:
 - Jede Diagnose weist jeder Komponente einen Verhaltensmodus zu: entweder ok oder ein definierter Fehlermodus
 - Die Regeln aller zugewiesenen Verhaltensmodi sind konsistent mit allen eingestellten und beobachteten Werten)

Was braucht der Anwender ?

Eingabe: s.o.

Ausgabe:

- Eine eindeutige Anweisung, welche Komponenten wie repariert werden sollen

Vom ACS zur kompletten Diagnosesoftware

Was fehlt noch ?

1) Vorschlag von Testeinstellungen (control inputs)

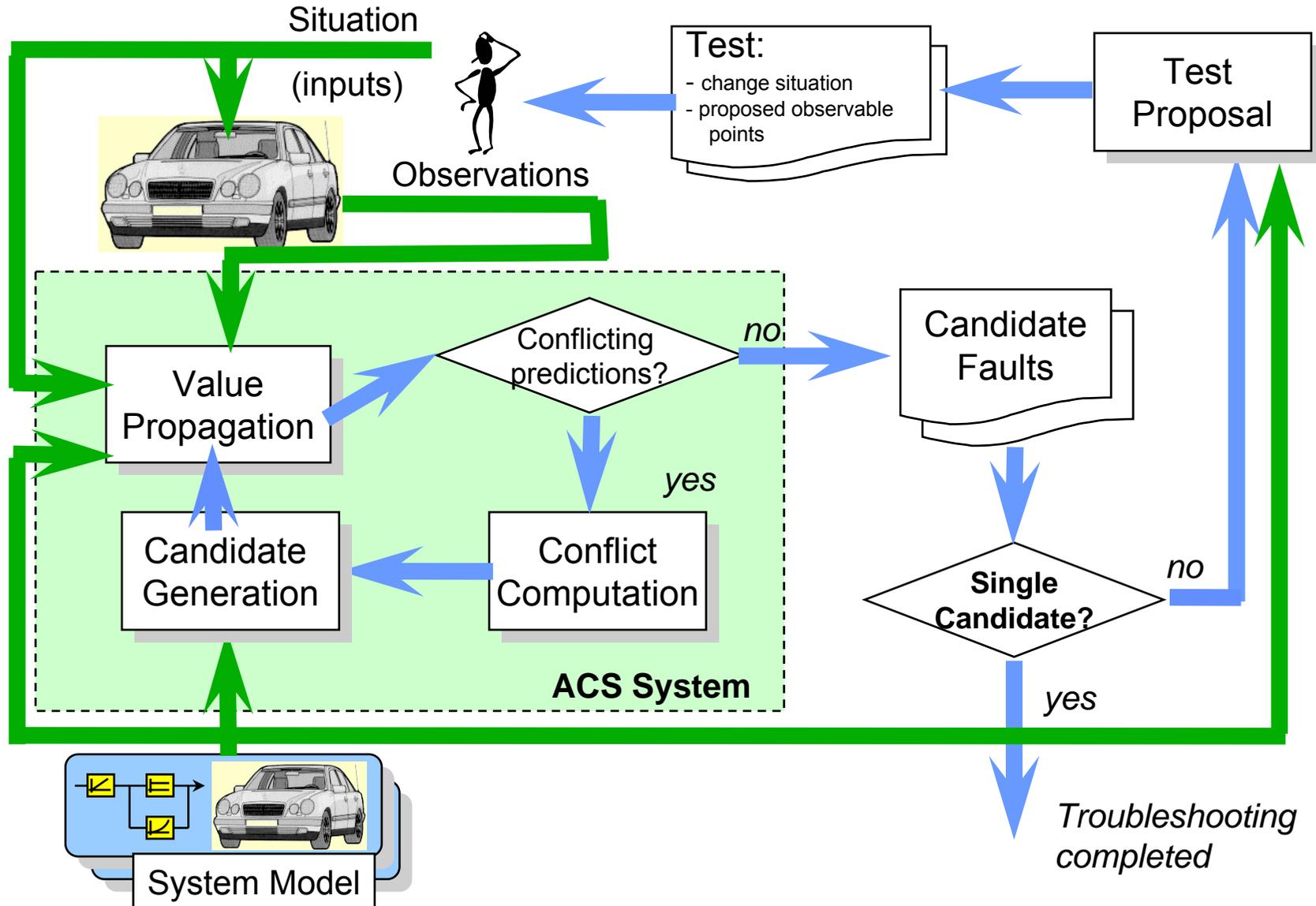
- Einstellung bestimmter Werte an bestimmten Stellen im System
(derart, dass die zu erwartenden Beobachtungen die bisher gültigen Diagnosen bestmöglich unterscheiden)

2) Vorschlag von Beobachtungspunkten

- Auswahl von Messstellen im System
(derart, dass die zu erwartenden Beobachtungen die bisher gültigen Diagnosen bestmöglich unterscheiden)

Test

Vom ACS zur kompletten Diagnosesoftware



Vom ACS zur kompletten Diagnosesoftware

Details:

1) Vorschlag von Testeinstellungen (control inputs)

- Die Control Inputs werden in der Wissensbasis gekennzeichnet und mit einer Bewertung versehen, welche die Schwierigkeit angibt, einen Wert einzugeben (Definition so genannter **Maßnahmen**).
- Die Eingabewerte an diesen Control Inputs werden als Annahmen im **selben** ATMS wie die Verhaltensmodi der Komponenten propagiert.
Die Menge aller Eingabewerte an Control Inputs heißt **Situation**.
- Das ATMS kann zwischen den Situationsannahmen und Verhaltensannahmen unterscheiden und fokussiert sowohl auf bestimmte Verhaltensannahmen als auch auf bestimmte Situationsannahmen.

➔ SIT-ATMS

- Für jede Situation wird der Informationsgewinn für den bestmöglichen Beobachtungspunkt berechnet.

Vom ACS zur kompletten Diagnosesoftware

Details:

2) Vorschlag von Beobachtungspunkten

- Gegeben sei eine fest vorgegebene Einstellung der Control Inputs (also eine Situation)
- Berechne die Wahrscheinlichkeit, dass an einem bestimmten Beobachtungspunkt in der ausgewählten Situation der Wert x vorliegt ($P(x)$).
- Berechne den Informationsgehalt (**Entropie**) für einen bestimmten Beobachtungspunkt B nach der Formel $I(B) = - \sum_x P(x) \cdot \log P(x)$, wobei x alle möglichen Werte für B sind.
- Suche den Beobachtungspunkt B , an dem $I(B)$ minimal ist.
 - ➔ Die Kenntnis des tatsächlichen Werts bringt den größten Informationsgewinn

Vom ACS zur kompletten Diagnosesoftware

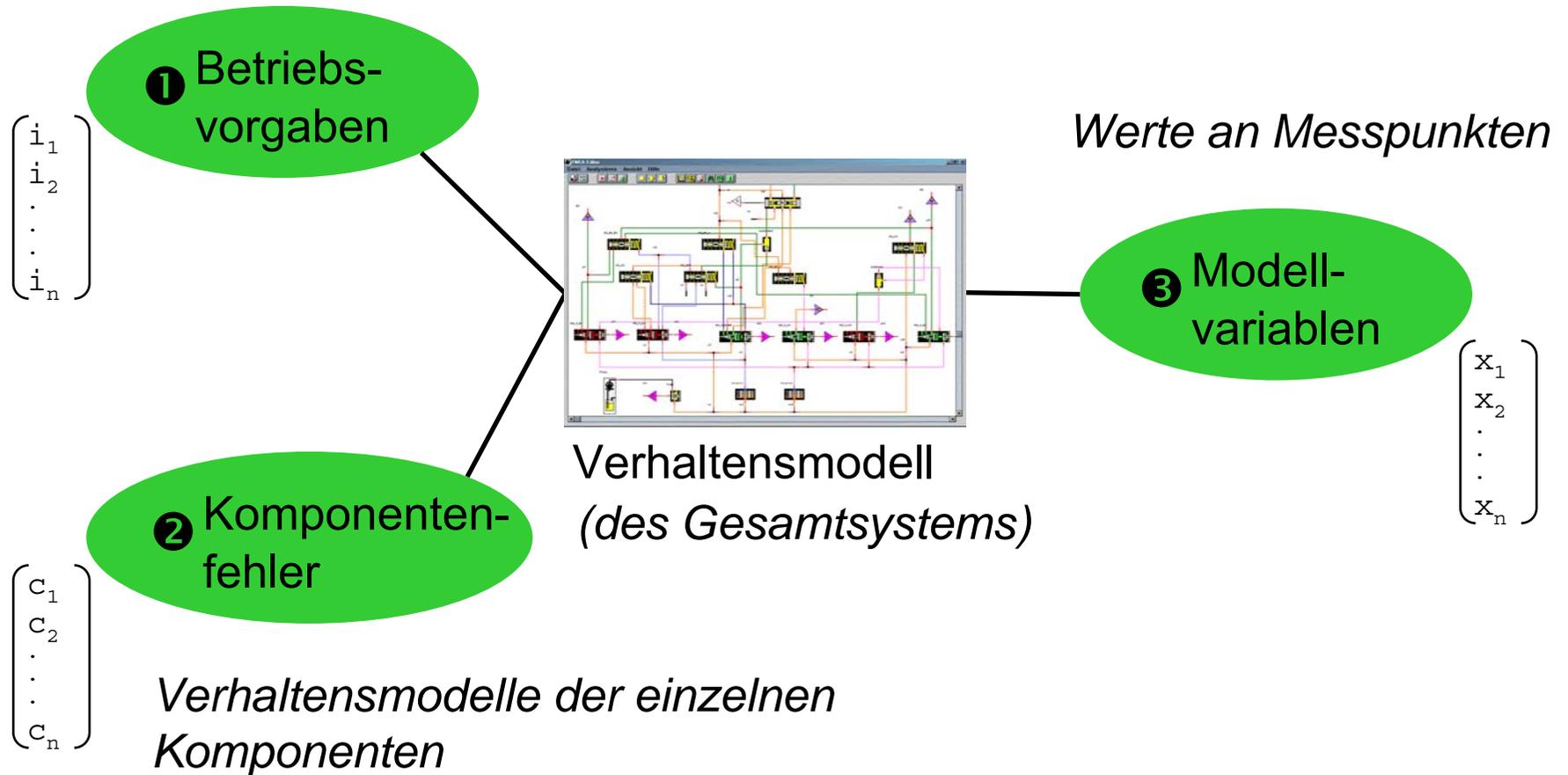
Zusammenfassung: Diagnose mit MDS

- MDS schlägt dem Benutzer systematisch Einstellungen im technischen System vor und gibt vor, wo er Werte messen soll.
- Dieser Vorgang wird solange wiederholt, bis eine eindeutige Diagnose mit hinreichend genauer Wahrscheinlichkeit gegeben werden kann.
- Die Diagnose gibt dem Benutzer explizit Hinweise, welche Komponenten defekt sind und wie er den Fehler beheben kann (wegen der Angabe des konkreten Fehlermodus)
- In der Wissensbasis kann für jeden Fehlermodus eine konkrete Abhilfemaßnahme hinterlegt werden.
 - ➔ Die Ausgabe von MDS ist eine konkrete Reparaturanleitung

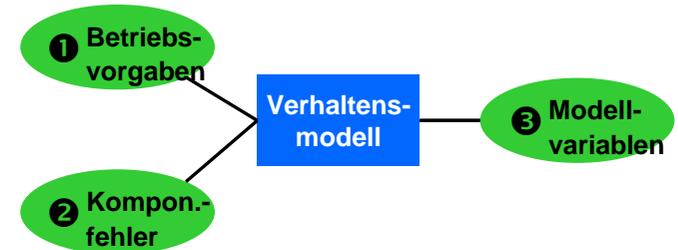
Model based troubleshooting

Weitere Aufgaben für MDS

Situationen (Werte an Control Inputs)

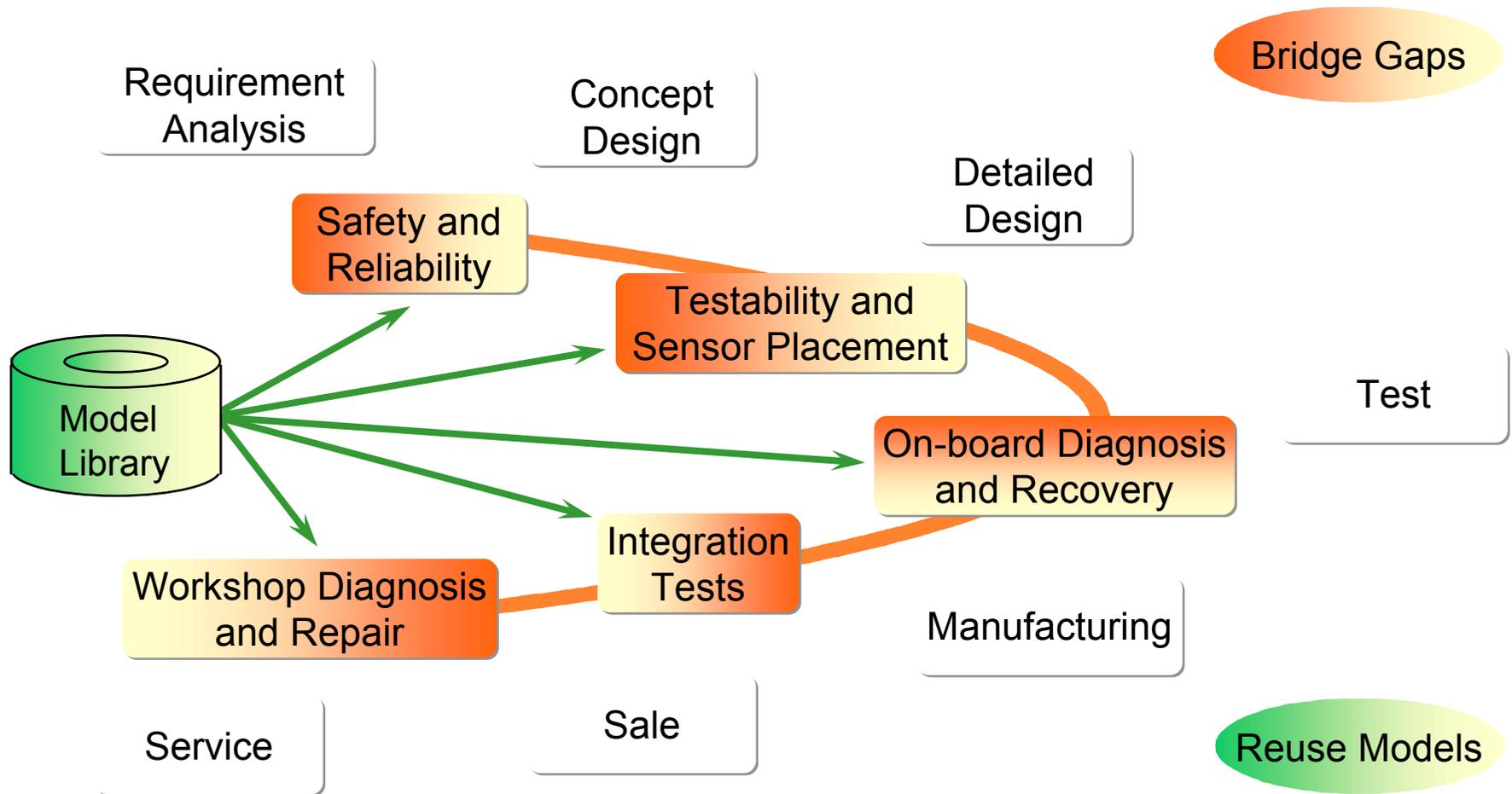


Weitere Aufgaben für MDS



| | | | |
|---|--|---------------------|---|
| A | | FMEA | 1 vorgeben, 2 vorgeben, 3 = sicherheitsrelevante Eigenschaften klassifizieren (FMEA Events) |
| B | | FTA | 3 sicherheitsrelevante Eigenschaften vorgeben, nach 1 und 2 suchen |
| C | | Testbarkeitsanalyse | 1 vorgeben, 2 vorgeben, 3 = vorhandene Sensoren bzw. Meßpunkte (Entscheidungsbaum generieren - eigenst. Fehlerkl.) |
| D | | Sensorplatzierung | 1 vorgeben, 2 vorgeben, 3 = alle potenziell mögliche Sensoren / Meßpunkte (Entscheidungsbaum generieren - Sensorplatzierung) |
| E | | Diagnose | 1 vorgeben, 3 messen, nach 2 suchen = Komponentenfehler |
| F | | Recovery | 2 vorgeben, 3 Ziele vorgeben, nach 1 suchen = mögliche Kompensationssteuerung |

MDS in der Geschäftsprozesskette



Zusammenfassung: Was kann MDS ?

System Design

Model-based Failure Analysis

Analysis Task

▶ *Result*

Reliability/Safety

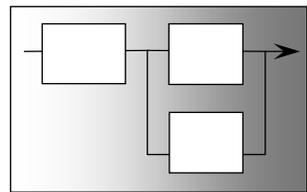
▶ *FMEA/FTA*

Testability

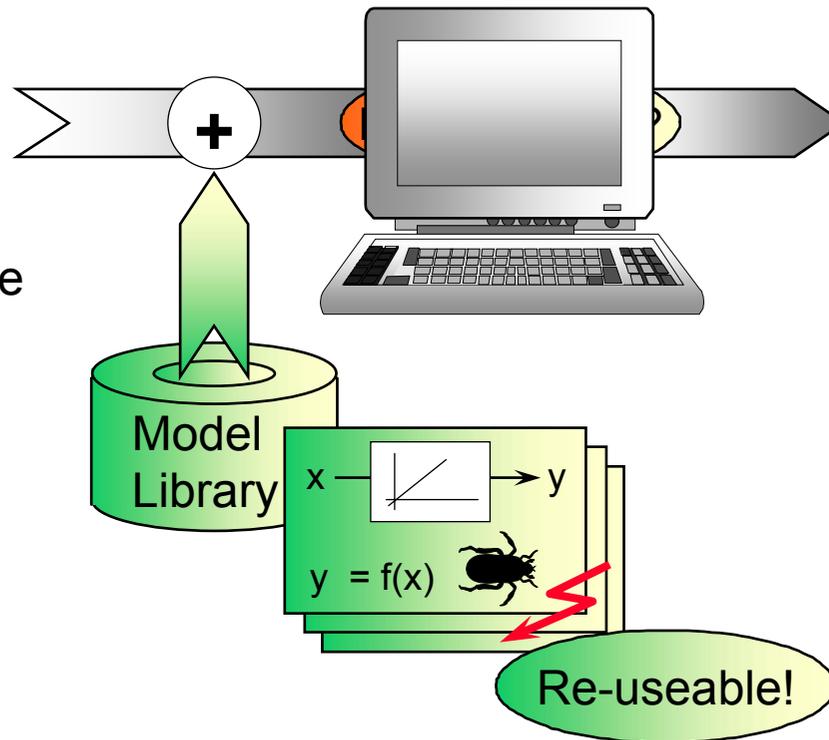
- ▶ *Sensor Placement*
- ▶ *Fault Classes*

Diagnosis

- ▶ *Decision Trees &*
- ▶ *Interactive Troubleshooting*



System Structure



Wissensbasierte Systeme

1. Motivation
2. Prinzipien und Anwendungen
3. Logische Grundlagen
4. Suchstrategien
5. Modellbasierte Diagnose
 - Kandidatengenerierung
 - Konfliktgenerierung
 - Zusammenspiel von Wertpropagierung, Konfliktgenerierung, Kandidatengenerierung
 - Weitere Funktionalitäten von MDS
 - ➔ Komponentenmodellierung
6. Andere Diagnosemethoden
7. Weitere Wissensrepräsentationsformen
8. Bewertung wissensbasierter Systeme

MDS Constraint Language

Value domains:

- finite / non-finite;
 - numeric (e.g. integers, floating point, ...);
 - numeric intervals (e.g. [-1, 1]);
 - boolean;
 - symbolic (e.g. #open, #close);
 - special structures (e.g. for electric);
- ⇒ **open to extensions** (domain properties must be specified at Smalltalk level);

MDS Constraint Language

Constraint types:

- **MDS Constraint:**
 - **set** of propagation rules;
- **MDS Propagation Rule:**
 - has a **procedural interpretation** (computes values given other values);
 - defined (at low level) using Smalltalk;
- **Generic Rule Set:**
 - feature to build higher-level constructs;

MDS Constraint Language

Examples

1) Constraint: “ $a = b$ ”

- MDS encoding using propagation rules:
 { R1: $\implies a := [b]$. “computes a from b ”
 R2: $\implies b := [a]$. “computes b from a ” };
- MDS encoding using the generic rule-set EQ:
 { R: $\implies EQ (a , b)$. };

2) Constraint: “ $a = b + c$ ”

- MDS encoding using propagation rules:
 { R1: $\implies a := [b + c]$. “computes a from b and c ”
 R2: $\implies b := [a - c]$. “computes b from a and c ”
 R3: $\implies c := [a - b]$. “computes c from a and b ” };
- MDS encoding using the generic rule-set SUM:
 { R: $\implies SUM (a , b , c)$. };

MDS Constraint Language

Examples

**3) Constraint: (ass=OK ==> out = in1 + in2) &
(ass=broken ==> out = 1)**

- MDS encoding using propagation rules :

{ R1: [ass == #ok] ==> out := [in1 + in2].

R2: [ass == #ok] ==> in1 := [out - in2].

R3: [ass == #ok] ==> in2 := [out - in1].

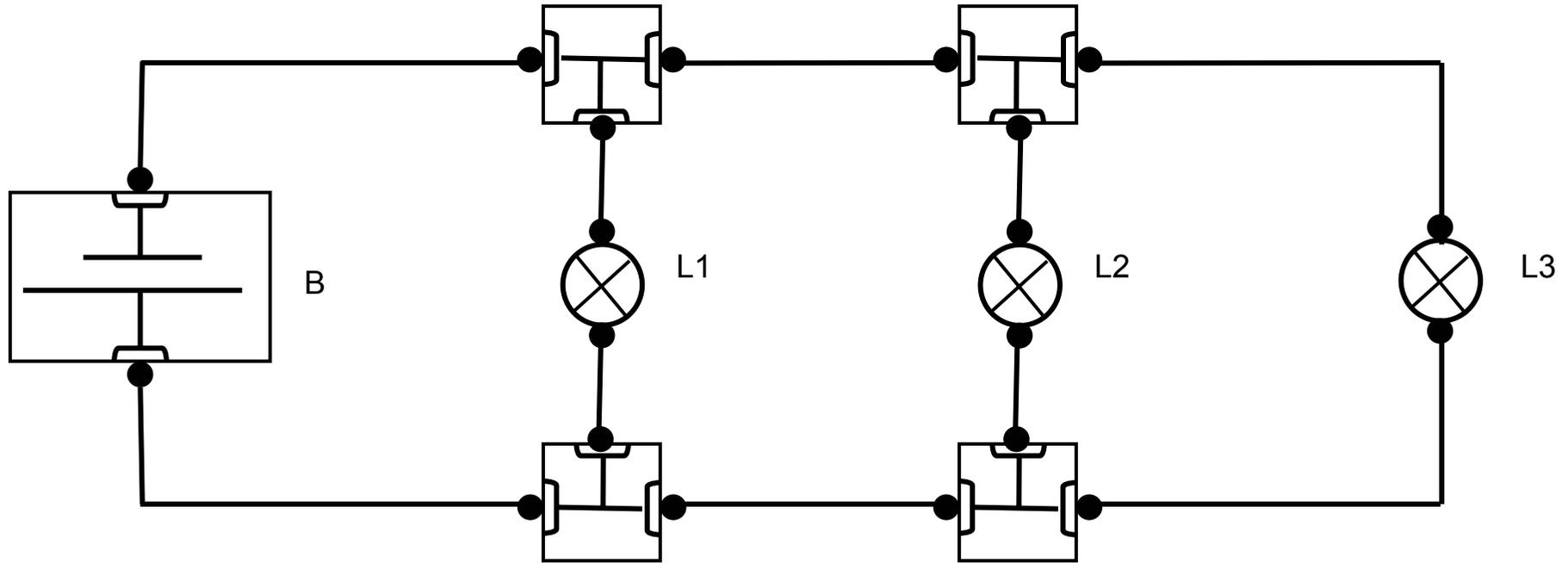
R4: [ass == #broken] ==> out := [1]. };

- MDS encoding using generic rule sets:

{ R1: [ass == #ok] ==> SUM (out, in1, in2).

R2: [ass == #broken] ==> EQ (out, [1]). };

Modellierung eines einfachen elektrischen Systems



Komponententypen:

Batterie

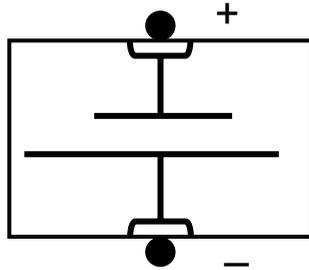
Lampe

Kabel

Steckverbindung (3)

Modellierung eines einfachen elektrischen Systems

Batterie



Fehlermodi:

entladen

Kontaktlücke bei +

Kontaktlücke bei -

Wackelkontakt bei +

Wackelkontakt bei -

korrodiert

Maßnahmen:

Klemme bei + lösen

Klemme bei - lösen

Klemme bei + befestigen

Klemme bei - befestigen

Beobachtungen:

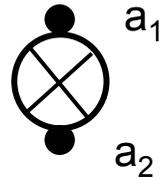
Klemmen ansehen

Spannung bei + messen

Spannung bei - messen

Modellierung eines einfachen elektrischen Systems

Lampe



Fehlermodi:

durchgebrannt
Lampe nicht eingesetzt
Wackelkontakt
korrodiert

Maßnahmen:

Lampe ausschrauben
Lampe einschrauben

Beobachtungen:

Lampe ansehen

Kabel



Fehlermodi:

unterbrochen
Kurzschluss gegen Masse
Kurzschluss gegen Spannung
korrodiert

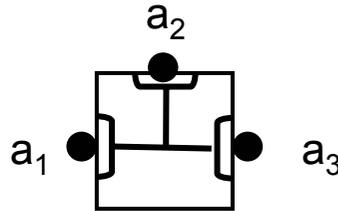
Maßnahmen:

Beobachtungen:

Spannung an a_1 messen
Spannung an a_2 messen
Kabel ansehen

Modellierung eines einfachen elektrischen Systems

Steckverbindung (3)



Fehlermodi:

Kontaktlücke bei a_1
Kontaktlücke bei a_2
Kontaktlücke bei a_3
Wackelkontakt bei a_1
Wackelkontakt bei a_2
Wackelkontakt bei a_3

Maßnahmen:

Kontakt bei a_1 schließen
Kontakt bei a_2 schließen
Kontakt bei a_3 schließen
Kontakt bei a_1 lösen
Kontakt bei a_2 lösen
Kontakt bei a_3 lösen

Beobachtungen: Kontakte ansehen

Beim nächsten Mal:

***Probleme bei der Komponentenmodellierung
Vergleich mit anderen wissensbasierten
Diagnosetechniken***