

# ***Künstliche Intelligenz***

Sebastian Iwanowski  
FH Wedel

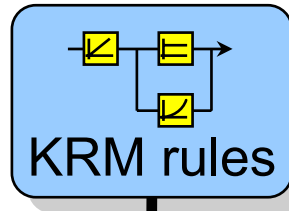
**Kap. 5:**  
Technische Diagnose

5.3: MDS: SW-Architektur und erweiterte Funktionalitäten

# Wertpropagierung und ATMS

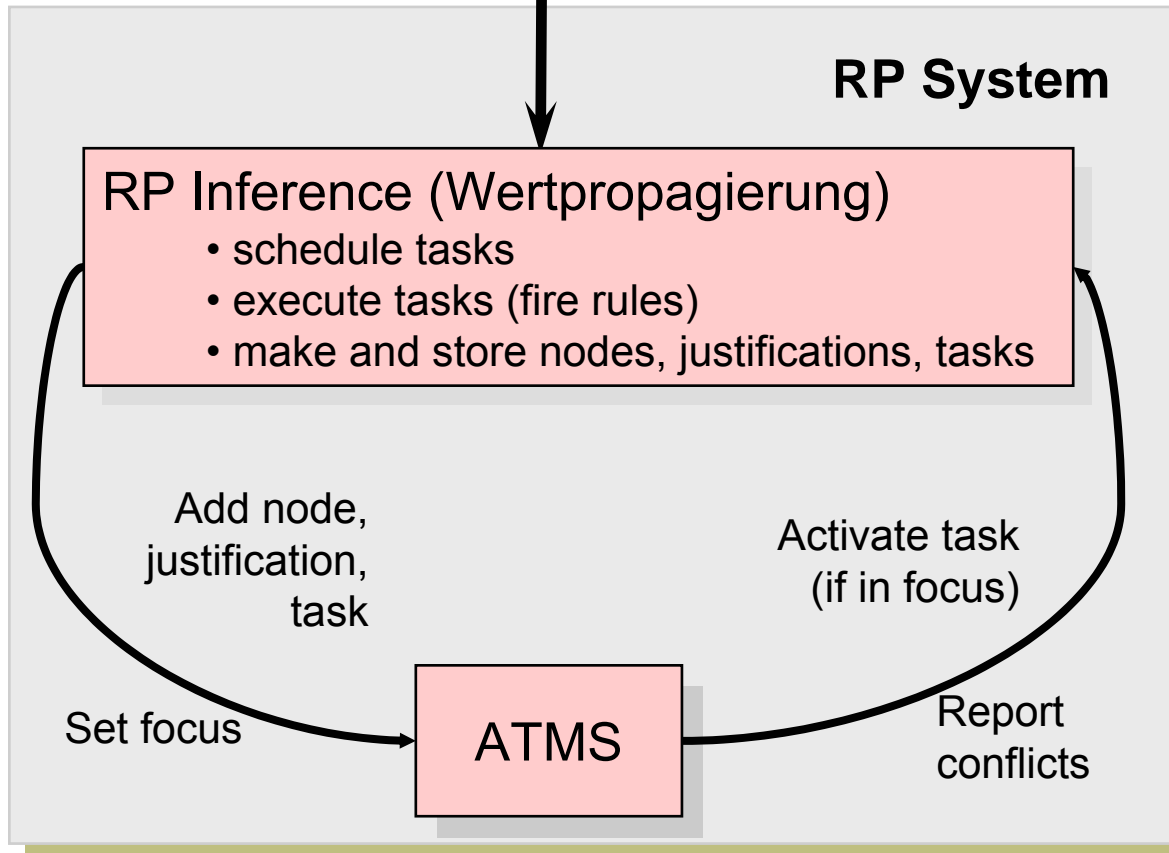
## Wissensbasis

(Komponentenmodellierung  
plus Systemzusammenhang)



## KRM:

Knowledge Representation Manager



# Wertpropagierung und ATMS

## Was versteht man unter Propagierung im MDS-Kontext ?

- Propagierung ist die Weiterleitung von Informationen über ein Netzwerk aus Kanten und Knoten.

## Trennung von Wertpropagierung und ATMS:

- Das **ATMS** ist verantwortlich für die Propagierung der Environments in einem gegebenen Netzwerk von Wertabhängigkeiten.
- Das Netzwerk von Wertabhängigkeiten wird in einem Rule Propagator (**RP**) hergestellt, der die Justifications aus den Regeln für die Verhaltensmodi der Komponenten zusammensetzt.
- Der RP ist genauso faul wie sein ATMS:
  - Werte werden nur weiterpropagiert, wenn sie von Environments unterstützt werden, die im gegenwärtigen Fokus stehen.
  - Das ATMS teilt dem RP unaufgefordert mit, welche Werte neu von Fokusenvironments unterstützt werden und initiiert die **(Propagation) Tasks**

# Wertpropagierung und ATMS

## Was bringt die Trennung von Wertpropagierung und ATMS ?

### 1. Antwort: Bessere Softwarearchitektur durch Modularisierung

- **Werte** entstehen meistens aus Beobachtungen (Messungen) und gezielten Eingaben. Diese sind spärlich, daher gibt es **nicht viele** resultierende Werte.
- **Environments** entstehen aus Annahmen über Verhaltensmodi. Von diesen gibt es **sehr viele** (selbst bei Einfachfehlern mindestens so viele wie Komponenten).
- Daher werden Fokusenvironments häufiger revidiert, als neue Werte berechnet werden. Diese Revision kann dann als ein ATMS-internes Problem behandelt werden.

*Anm.:* Die Aufteilung in ein RP- und ATMS-Modul fördert enge Modulbindung und lose Modulkopplung

# Wertpropagierung und ATMS

## Was bringt die Trennung von Wertpropagierung und ATMS ?

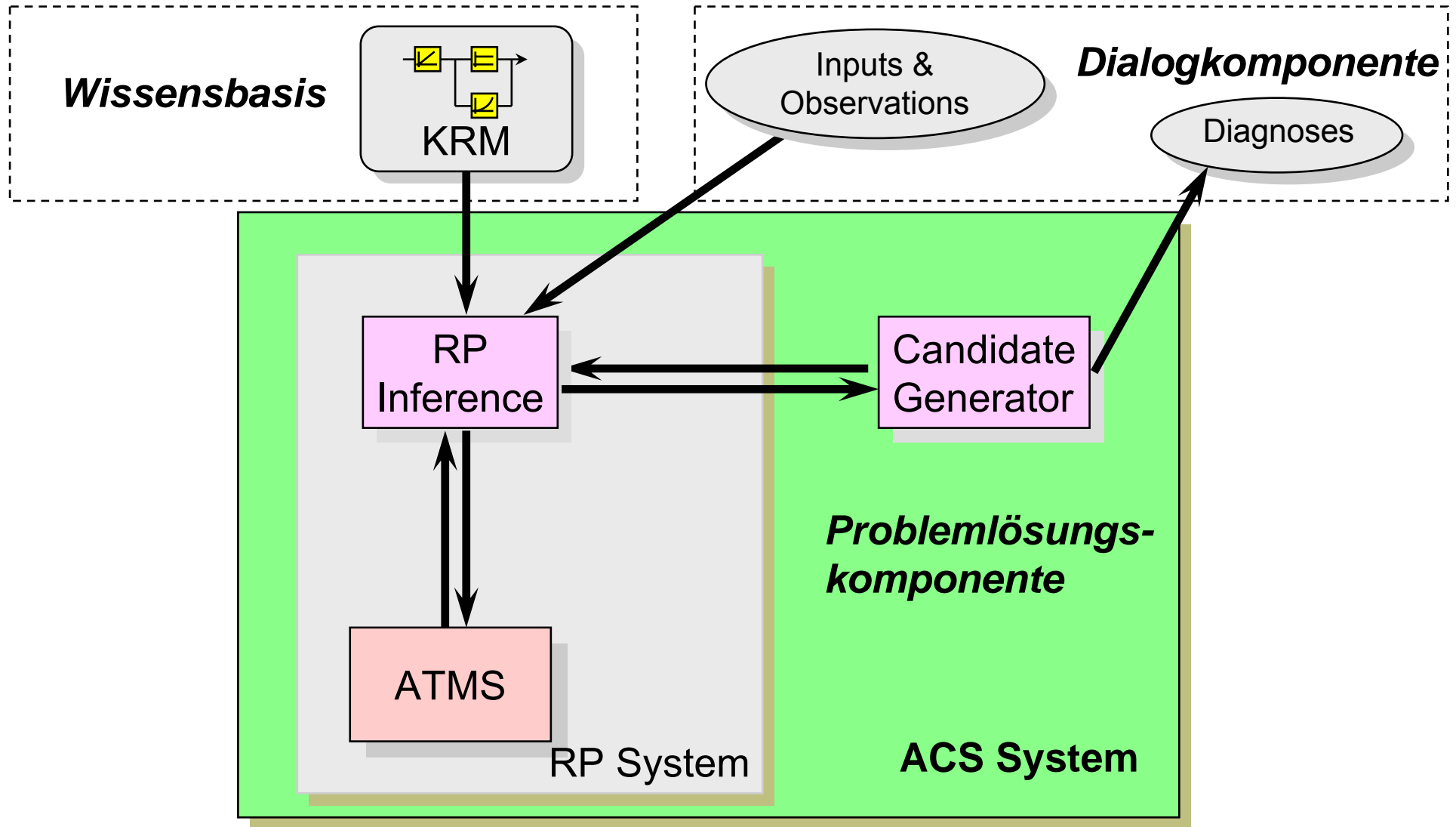
### 2. Antwort: Einsatz des ATMS für erweiterte Aufgaben

- Es können auch andere Annahmen als Verhaltensmodi für Komponenten untersucht werden:

#### Beispiele:

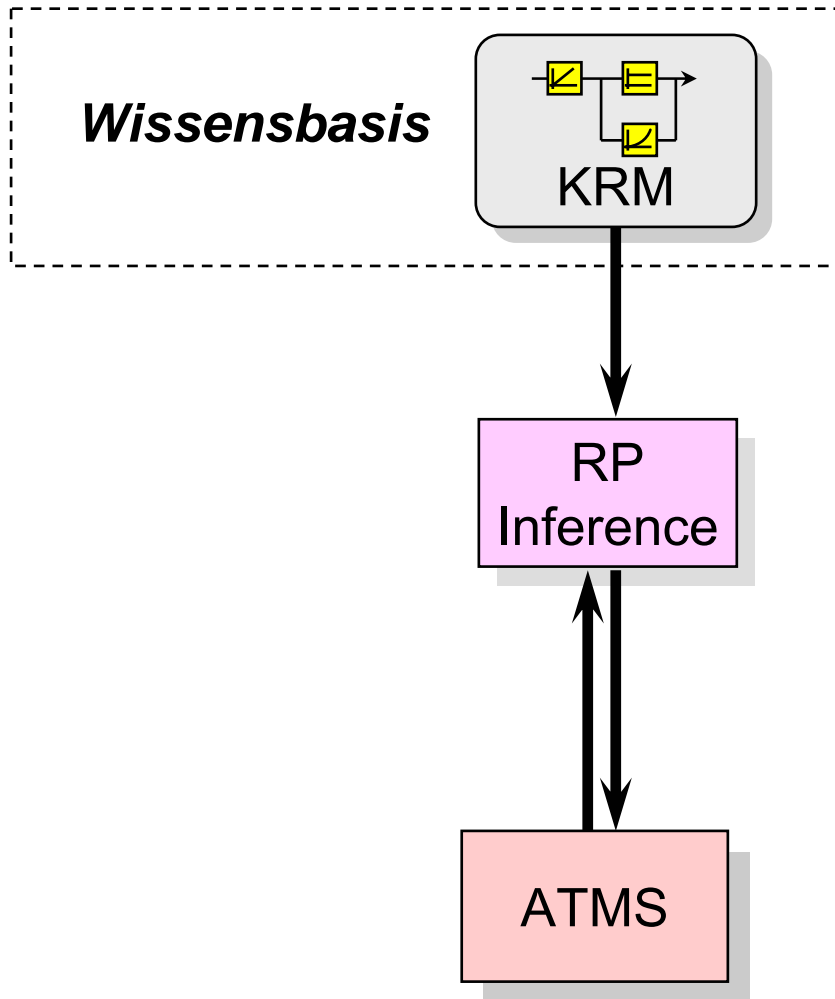
- Annahmen über Werteingaben (control inputs)  
*(für die Berechnung sinnvoller Testsituationen)*
- Annahmen über Komponentenzustände (bei dynamischen Komponenten)  
*(für dynamische Komponenten, deren Zustand unbekannt ist)*
- Annahmen über beliebige andere Werte  
*(könnte für Beobachtungspunkte interessant sein)*

# Zusammenspiel Kandidatengenerierer, RP und ATMS



**ACS: Assumption-based Constraint Solver**

# Anforderung an die Wissensbasis



## Was muss die Wissensbasis an die Inferenzkomponente liefern ?

- Regeln für die Wertzusammenhänge in den einzelnen Verhaltensmodi (*Komponentenmodellierung*)
- Kenntnis über die Wertdomänen: Wann gelten zwei Werte als widersprüchlich ?

***MDS löst diese Anforderungen durch das Anbieten einer Constraint-Sprache für die Komponentenmodellierung***

# 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 also noch ?

### 1) Vorschlag von Testeinstellungen (control inputs)

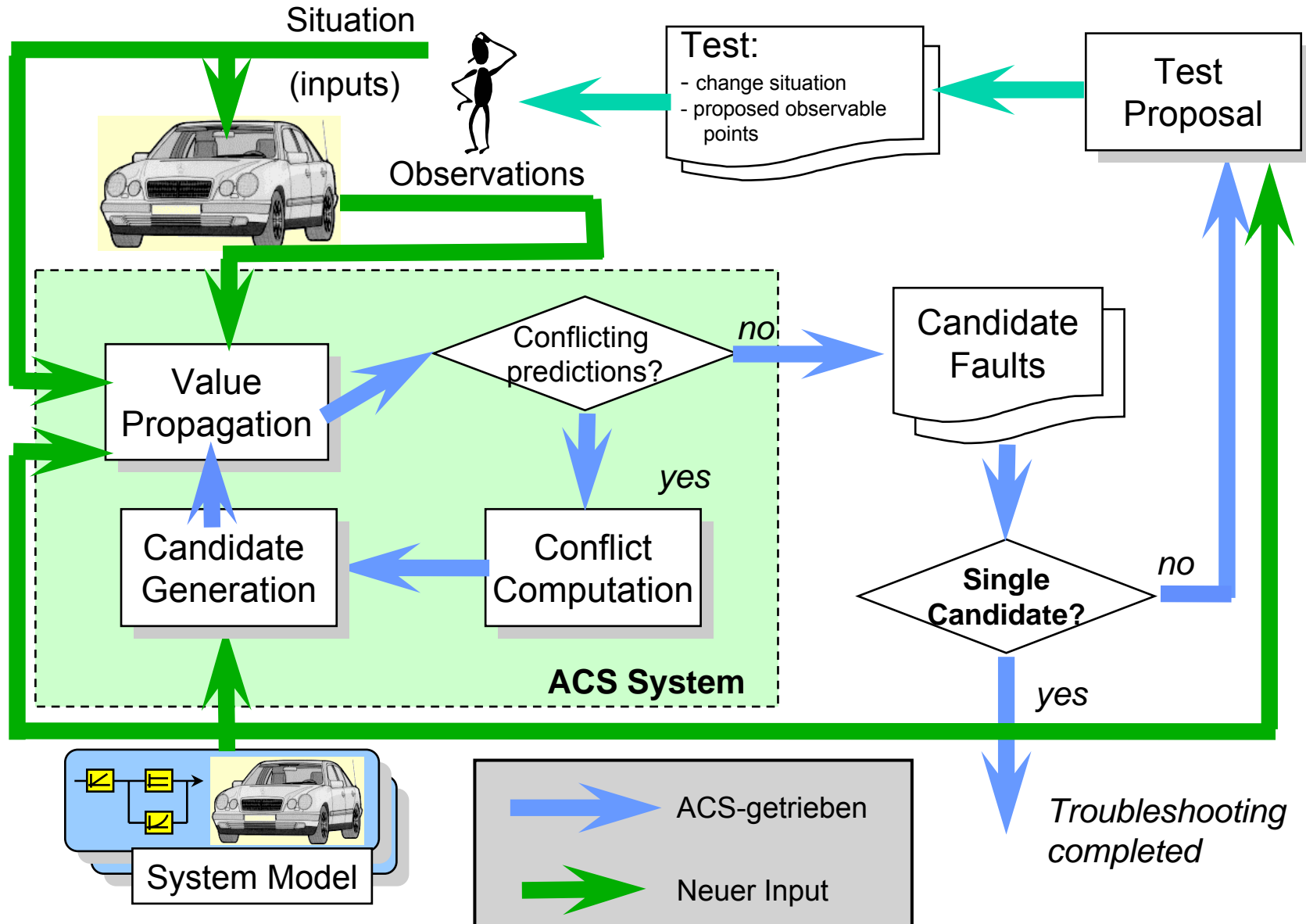
- Einstellung bestimmter Werte an bestimmten Stellen im System  
(derart, dass die zu erwartenden Beobachtungen die bisher gültigen Diagnosen bestmöglich unterscheiden)

### 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 **Des**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  $I(B)$  **maximal** ist.
  - ➔ Die Kenntnis des tatsächlichen Werts bringt den größten Informationsgewinn

# Vom ACS zur kompletten Diagnosesoftware

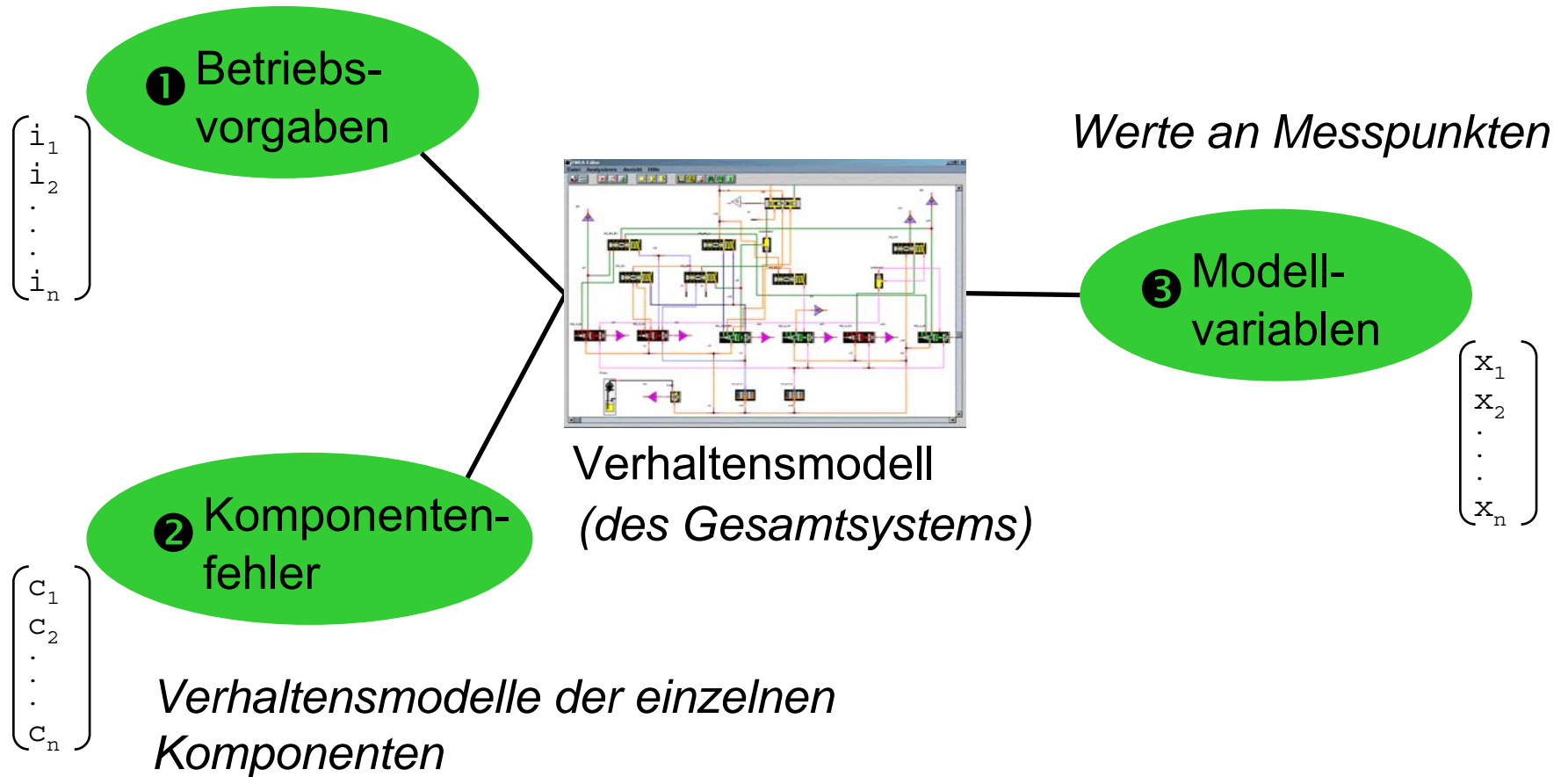
## Zusammenfassung: Diagnose mit MDS

- 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

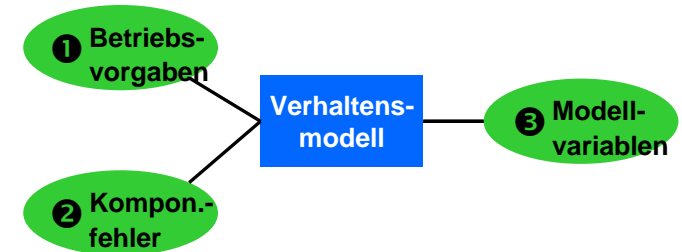
***Model based troubleshooting***

# Weitere Aufgaben für MDS

*Situationen (Werte an Control Inputs)*

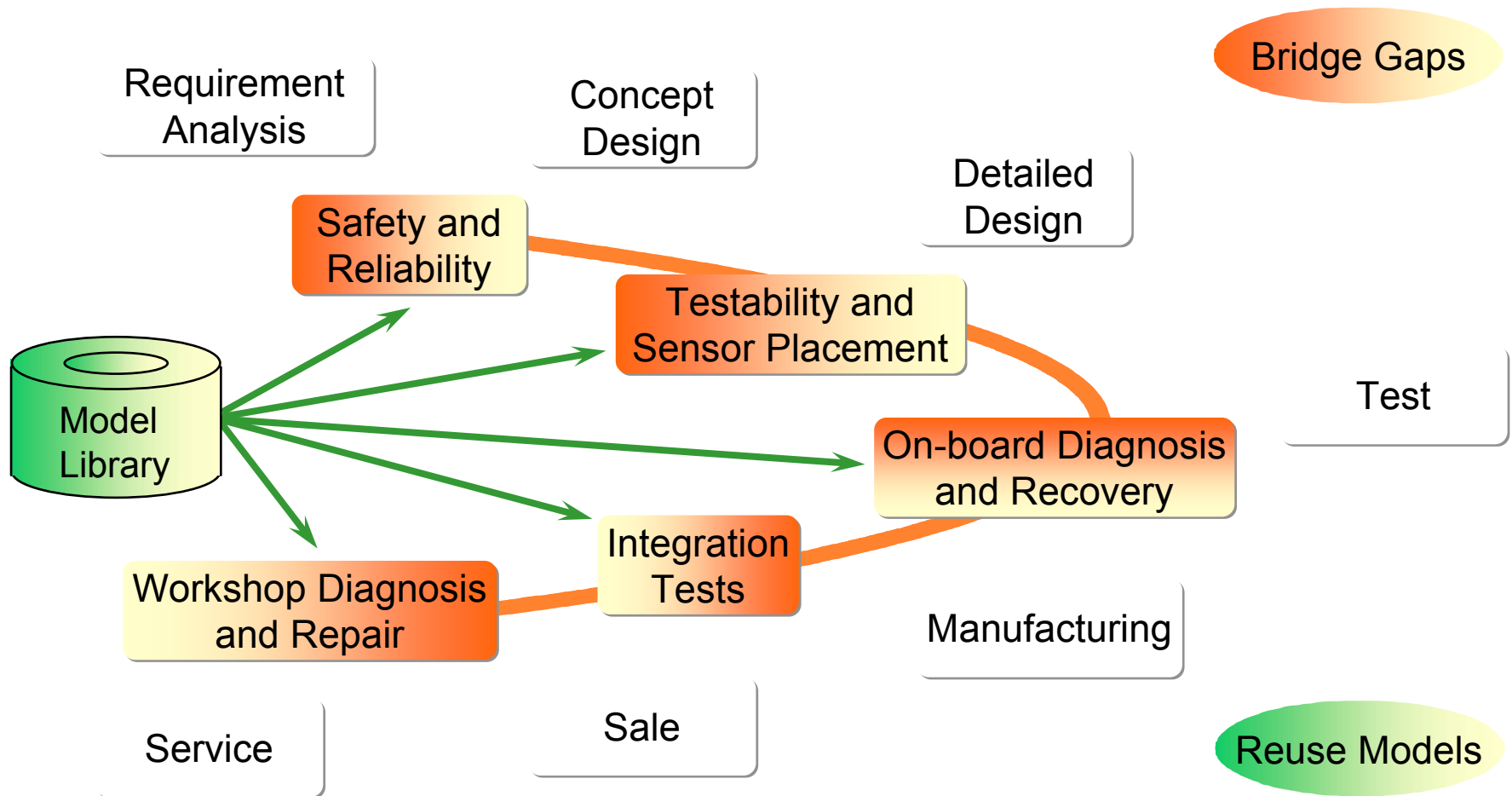


# Weitere Aufgaben für MDS



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

# MDS in der Geschäftsprozesskette





# Zusammenfassung: Was kann MDS ?

System Design

Model-based Failure Analysis

Analysis Task

▶ *Result*

Reliability/Safety

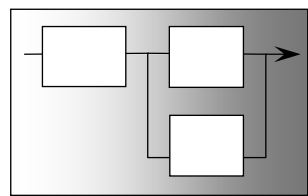
▶ *FMEA/FTA*

Testability

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

Diagnosis

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



System Structure

