Workshop on Artificial Intelligence in Practice

Part 1: AI Targets and Applications in Technics and Logistics

Sebastian Iwanowski
FH Wedel (D)
Erasmus Workshop at Fontys University, Eindhoven (NL)

Section 2: Applications in Technical Diagnosis

Technical Diagnosis

Run time system:

(in knowledge-based systems called **problem solver / inference engine**)

Input:

- Set of determined values in the system (control inputs)
- Measurement of resulting values in the system (observations)

Output:

- A unique prescription which components have to be repaired in which way

All different diagnostic techniques feature this specification !

Technical Diagnosis

Knowledge processing:

1) Knowledge acquisition

- rule-based (symptom-based)
- model-based
- case-based

} alternatively

2) Structure of knowledge base

- depends on knowledge acquisition

3) Knowledge processing in the inference engine

- depends on structure of knowledge base

This is where different diagnostic techniques differ !

Symptom-Based Diagnosis

Input to knowledge base:

- Causing or permanent faults in the complete system
- Possible symptoms (measurements)
- Dependencies between symptoms and faults (as rules)
    - Certain symptoms may confirm or even explain a fault
    - Certain symptoms may disconfirm or even exclude a fault.

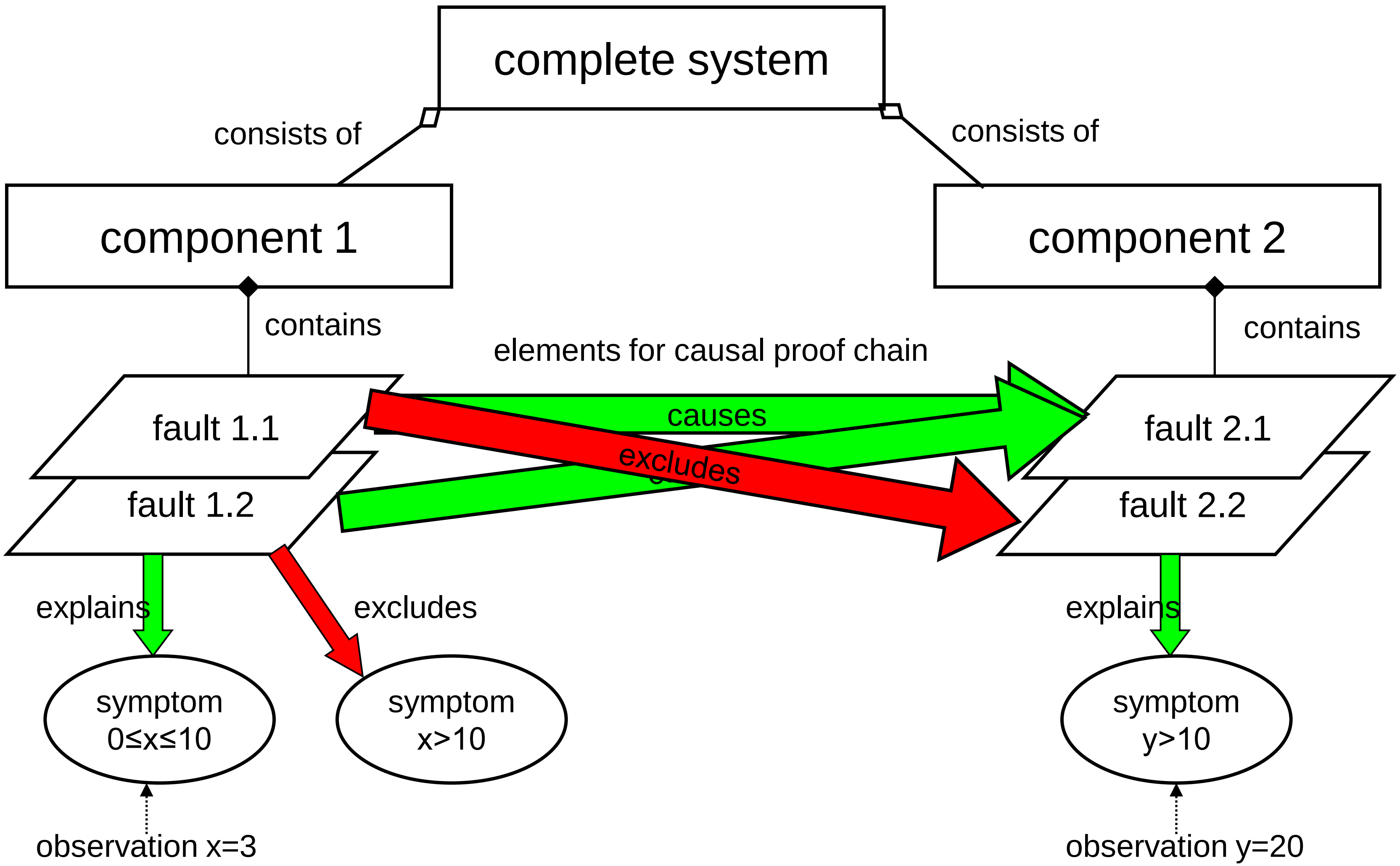
Structure of knowledge base:

- Semantic network of rules
- Possible instantiations:
    - Fault networks
    - Decision trees

This is the classical „expert system technology“

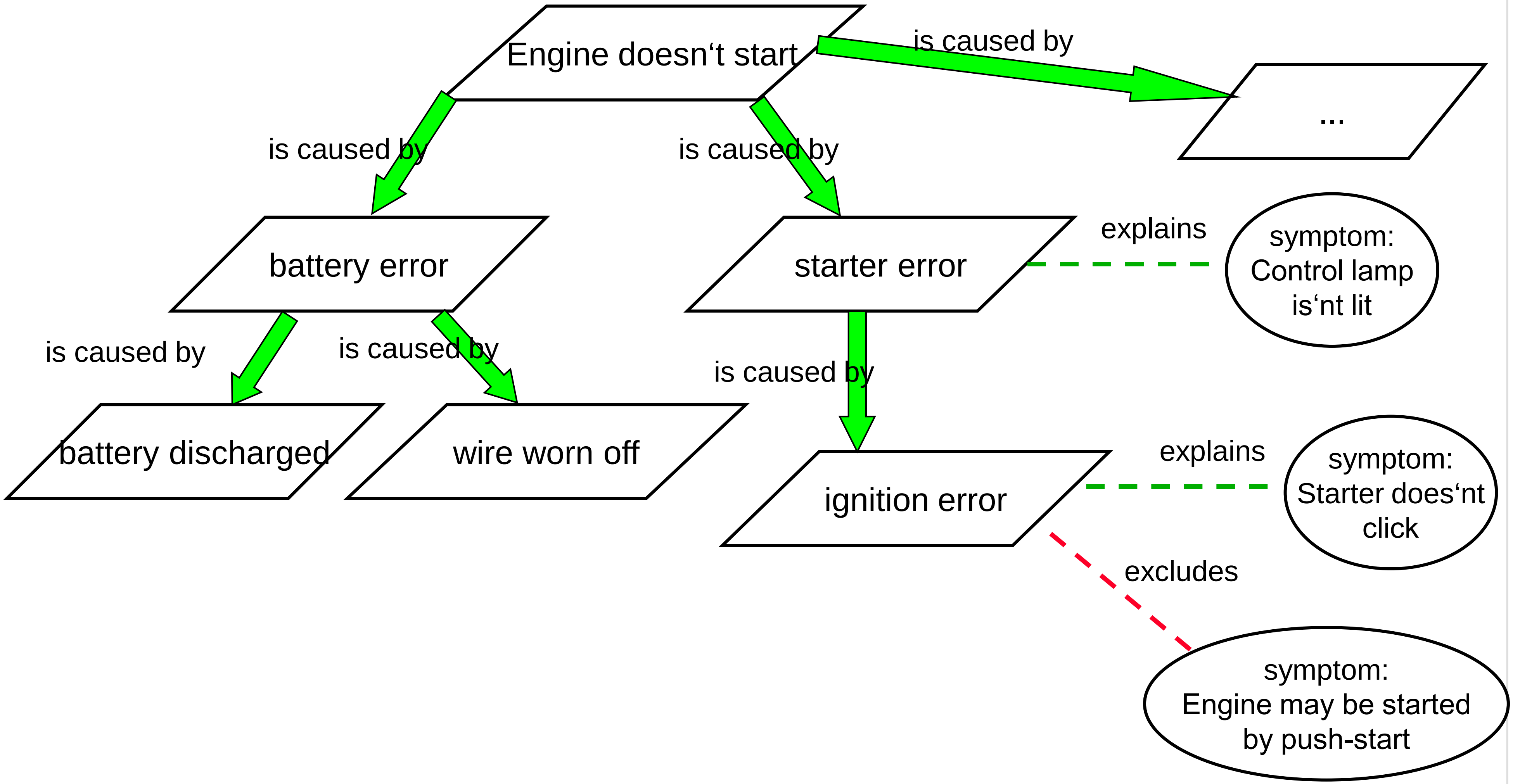
Symptom-Based Diagnosis

Example for elements of a symptom-based knowledge base:



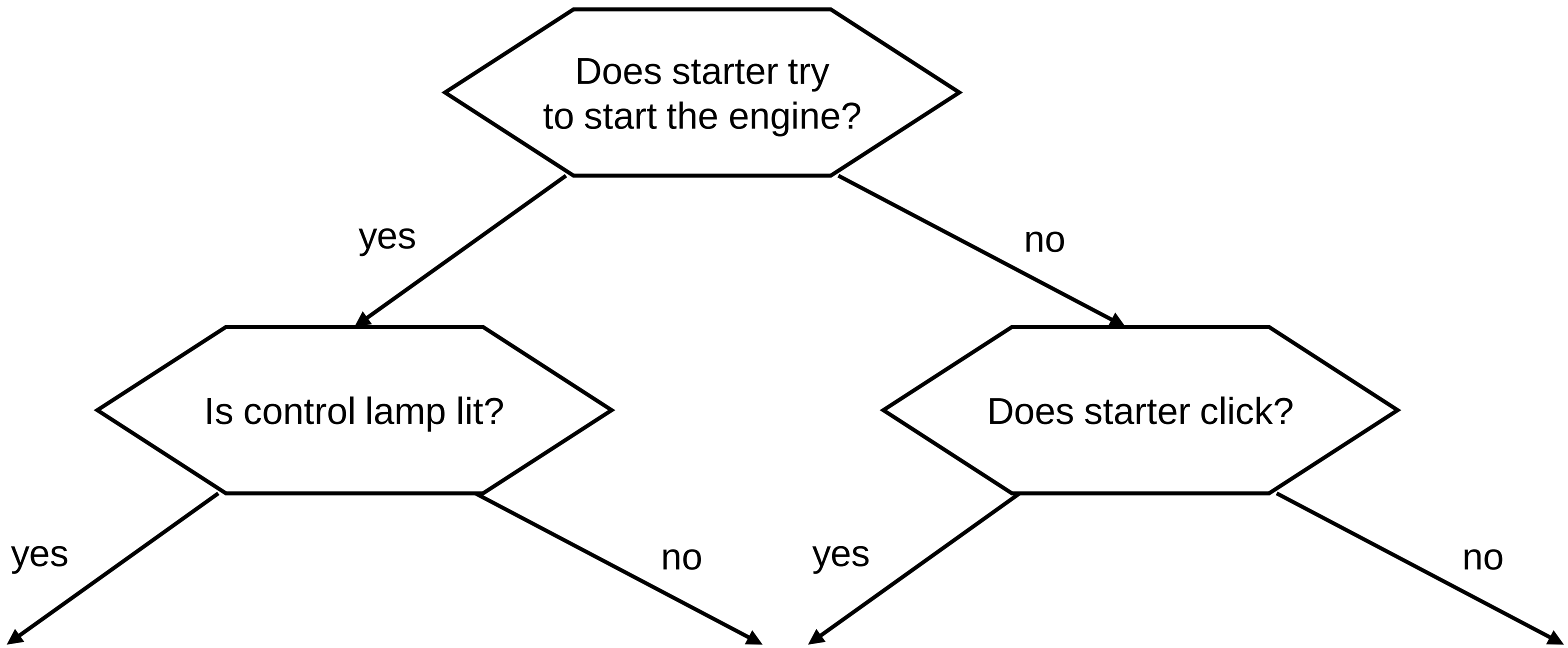
Symptom-Based Diagnosis

Exampe for a fault tree:



Symptom-Based Diagnosis

Example for a decision tree:



Symptom-Based Diagnosis

Functionality of inference engine:

- **Navigation in semantic network**
  (e.g., fault tree or decision tree)

- **Feasible starting points of navigation:**
  - Faults suspected
  - Symptoms observed

- **Main work consists of evaluating and firing rules:**
  - Conclusions of rules are inserted into antecedents of other rules.
  - Add-on: working with probabilities and fuzzy rules

Such input must be provided by knowledge acquisition component.

Symptom-Based Diagnosis

Advantages und Disadvantages:

- **Knowledge structure corresponds to expert knowledge.**
  - easy to use by an expert
  - Knowledge acquisition costs a lot of time.
- **Knowledge is represented very goal-oriented.**
  - quick run time component
  - difficult to alter the knowledge base
  - Reusability is a problem.
  - However, there are techniques to enable reusability.
- **Knowledge has shallow structure.**
  - All application domains are feasible in principle.
  - Knowledge base is likely incomplete.
  - Knowledge base is unstructured and difficult to be verified.

A lot of knowledge bases are faulty.

Model-Based Diagnosis

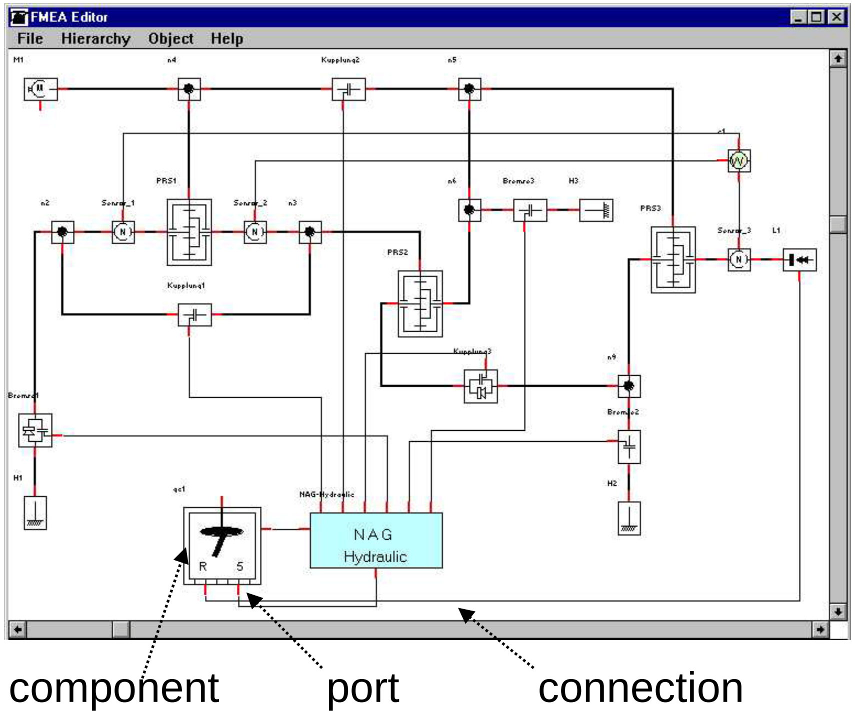
Targets:

- fast knowledge acquisition
- correct and easily verifyable results by inference engine

Challenging difficulty:

- Reasonably fast run-time component

Model-Based Diagnosis



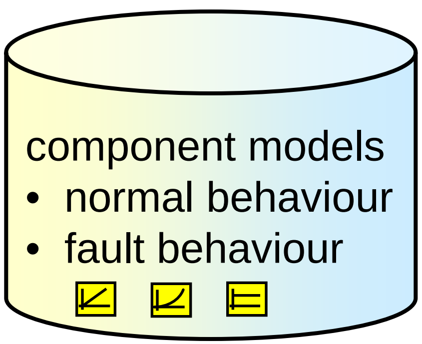
System structure:

Which components of what type are connected to each other in which way?

➔ ***available from CAD data***

component models:

How do values at different ports of the component depend on each other?

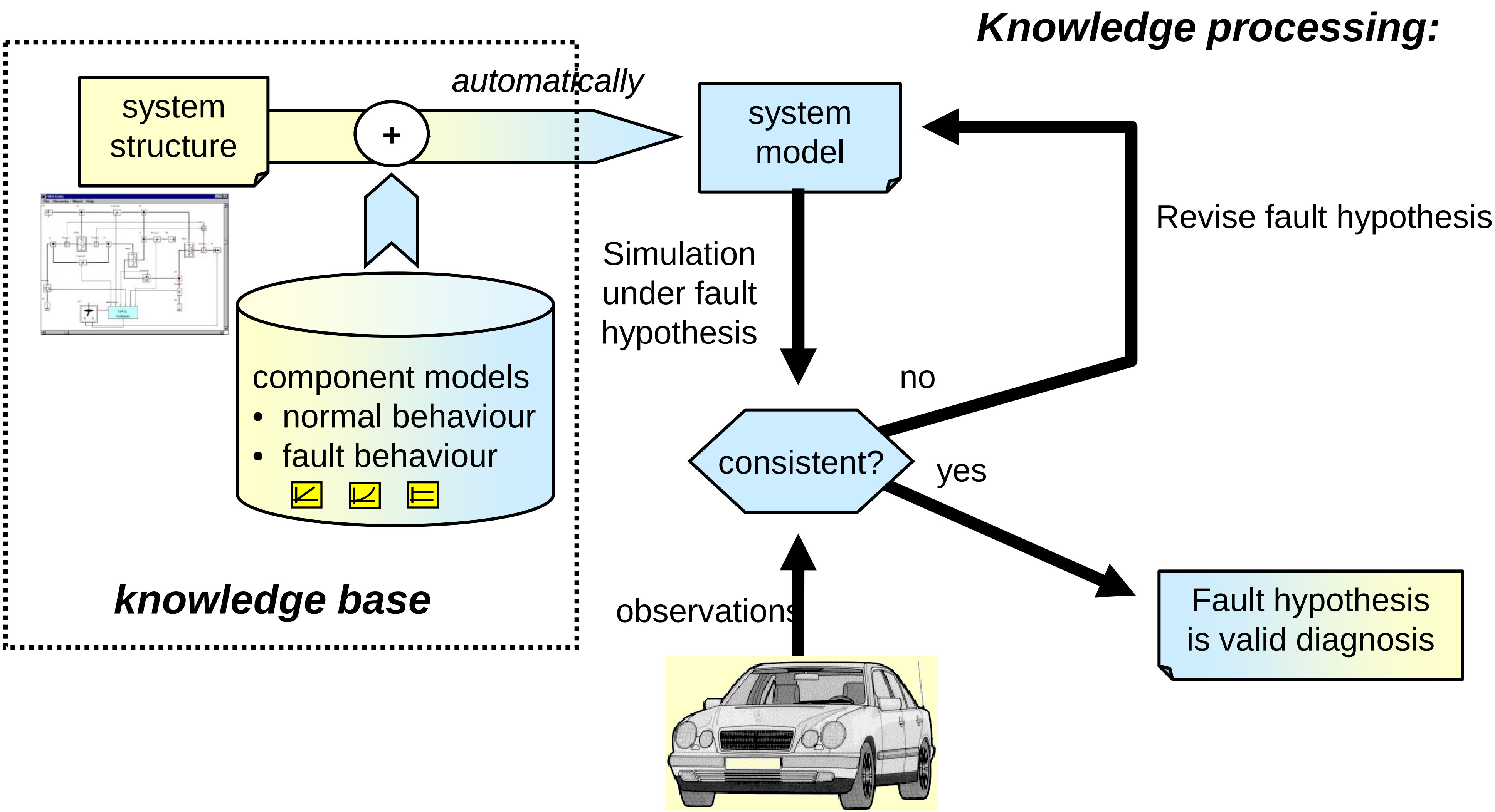


➔ ***is to be modelled once per component type***

➔***model may be reused for all systems containing the same type of component***

Model-Based Diagnosis

Basic functionality cycle



Model-Based Diagnosis

Basic functionality cycle

GDE 1987: *The* prototype for model-based diagnosis

original problem:

- brute force simulation of ***all*** fault hypotheses is not feasible due to combinatorial explosion

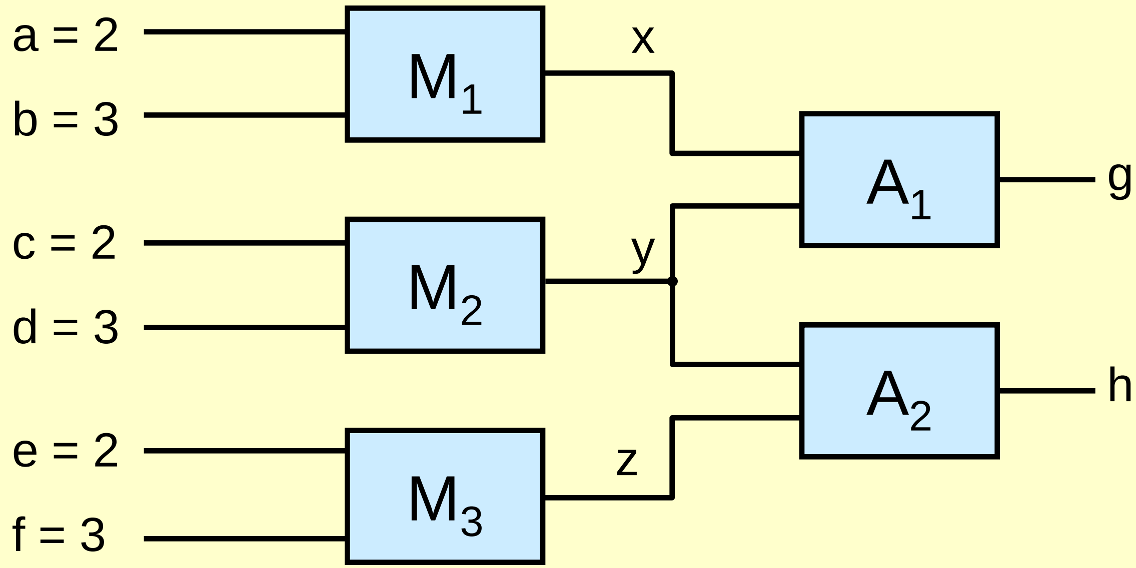
idea: General Diagnostic Engine GDE, deKleer & Williams 1987

- intelligent search in the space of all possible fault hypotheses
- uses inconsistencies among assumptions to cut the search space
- underlying principle: conflict-driven search

Model-Based Diagnosis

GDE: example

system structure



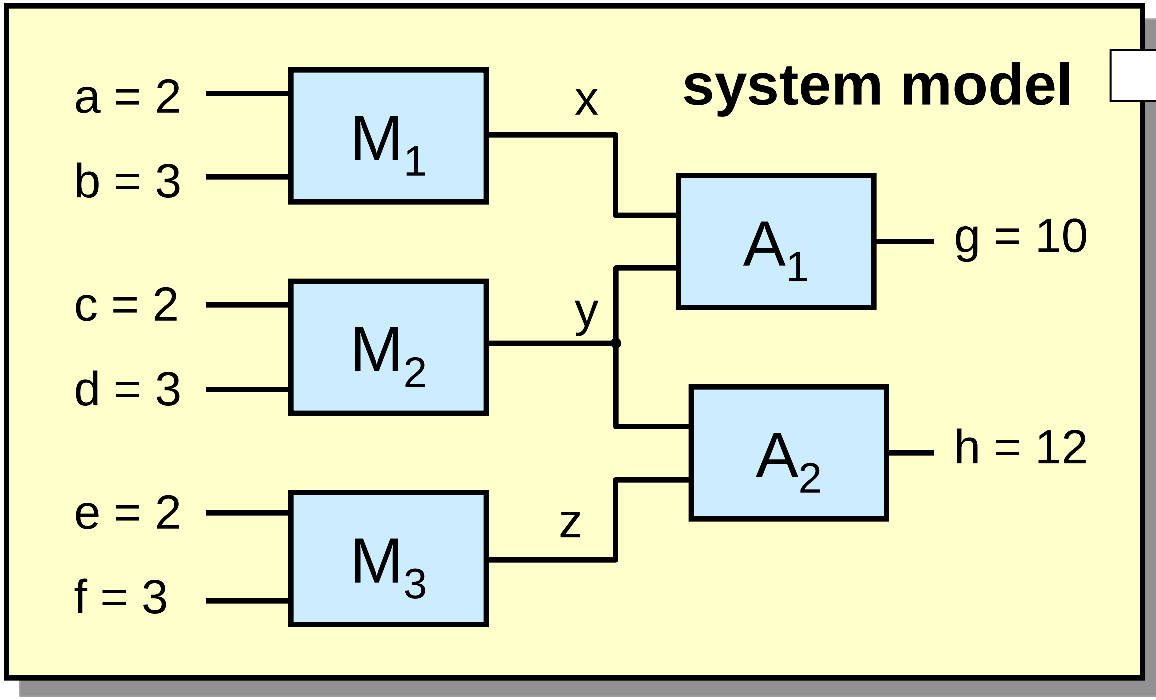
component models:

- multiplyers: mode=ok $\Rightarrow$ out = $in_1 * in_2$
- adders: mode=ok $\Rightarrow$ out = $in_1 + in_2$

measurements: $g = 10 \wedge h = 12$

Model-Based Diagnosis

GDE: example



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}

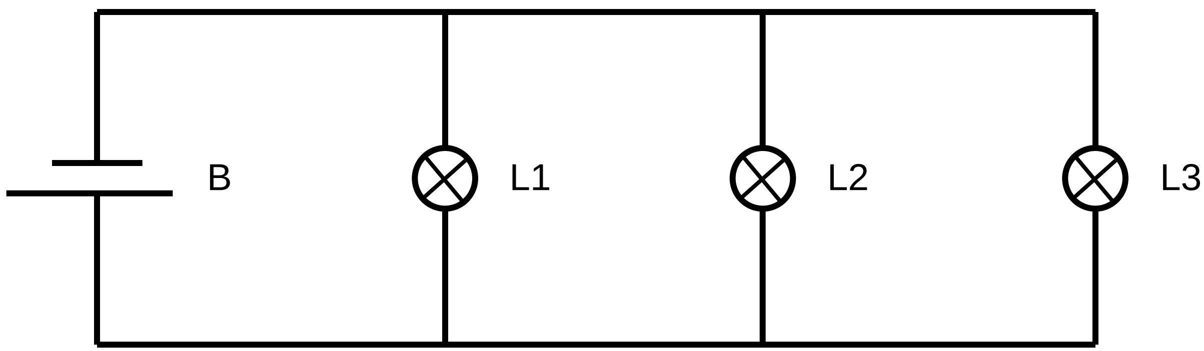
two conflicts

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

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

Model-Based Diagnosis

Example showing why the previous example does not cover all needs required in practice:



Observation:

L1, L2 are dark, L3 is lit

GDE diagnoses:

1. **(B ok, L1 defect, L2 defect, L3 ok)**
2. **(B defect, L1 ok, L2 ok, L3 defect)** ???
3. **(B defect, L1 ok, L2 ok, L3 ok)** ???

Model-Based Diagnosis

Models for electrical components:

Battery:

value domains: minus, plus $\in$ { `ground`, `supplyVoltage` }

rules:

ok $\Rightarrow$ (minus = `ground`)

ok $\Rightarrow$ (plus = `supplyVoltage`)

Wire:

value domains: a1, a2 $\in$ { `ground`, `supplyVoltage` }

rules:

ok $\wedge$ (a1 = `ground`) $\Rightarrow$ (a2 = `ground`)

ok $\wedge$ (a1 = `supplyVoltage`) $\Rightarrow$ (a2 = `supplyVoltage`)

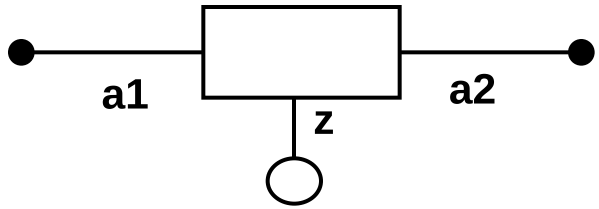
ok $\wedge$ (a2 = `ground`) $\Rightarrow$ (a1 = `ground`)

ok $\wedge$ (a2 = `supplyVoltage`) $\Rightarrow$ (a1 = `supplyVoltage`)

Model-Based Diagnosis

Models for electrical components:

bulb:



value domains:

a1, a2 ∈ { ground, supplyVoltage }

z ∈ { lit, dark }

rules:

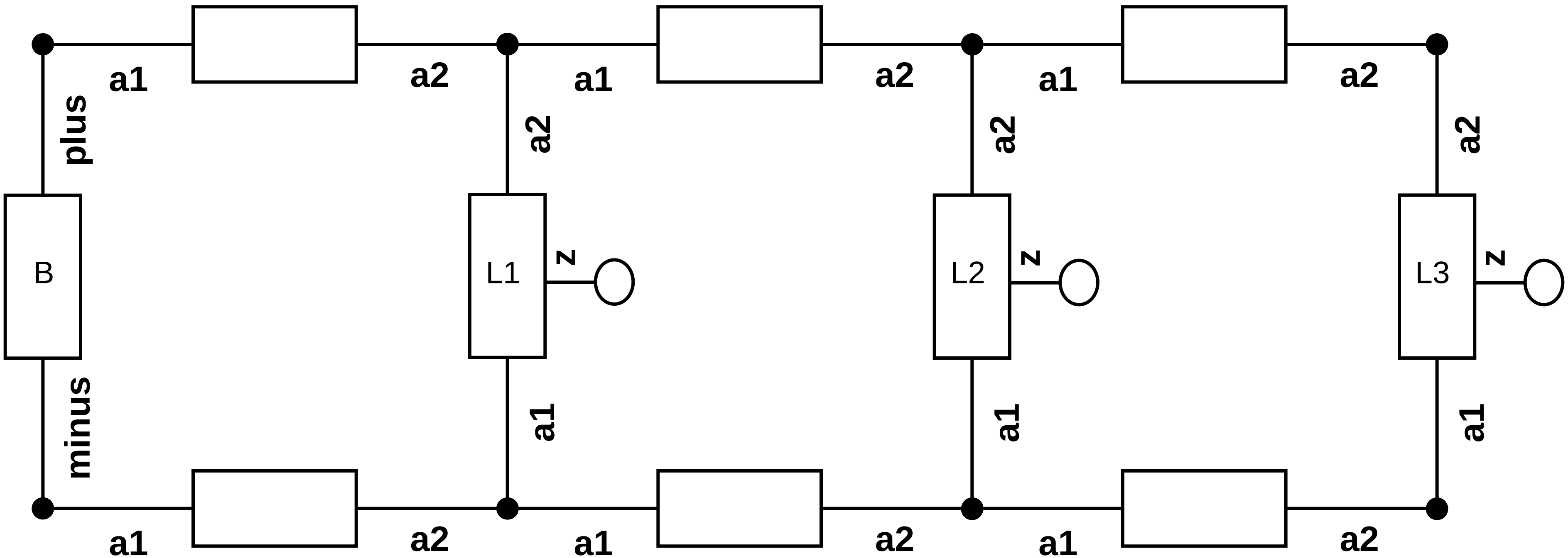
ok ∧ (a1 = supplyVoltage) ∧ (a2 = ground) ⇒ (z = lit)
ok ∧ (a2 = supplyVoltage) ∧ (a1 = ground) ⇒ (z = lit)
ok ∧ (a1 = supplyVoltage) ∧ (a2 = supplyVoltage) ⇒ (z = dark)
ok ∧ (a1 = ground) ∧ (a2 = ground) ⇒ (z = dark)

ok ∧ (a1 = ground) ∧ (z = lit) ⇒ (a2 = supplyVoltage)
ok ∧ (a1 = supplyVoltage) ∧ (z = lit) ⇒ (a2 = ground)
ok ∧ (a1 = ground) ∧ (z = dark) ⇒ (a2 = ground)
ok ∧ (a1 = supplyVoltage) ∧ (z = dark) ⇒ (a2 = supplyVoltage)

ok ∧ (a2 = ground) ∧ (z = lit) ⇒ (a1 = supplyVoltage)
ok ∧ (a2 = supplyVoltage) ∧ (z = lit) ⇒ (a1 = ground)
ok ∧ (a2 = ground) ∧ (z = dark) ⇒ (a1 = ground)
ok ∧ (a2 = supplyVoltage) ∧ (z = dark) ⇒ (a1 = supplyVoltage)

Model-Based Diagnosis

Composing the system model from the component models:



Values at connecting ports must be equal for both adjacent components

In case of contradictory values: conflict among the supporting behavioural model assumptions

Diagnoses are complete assignments of behavioural modes not containing any conflict

Model-Based Diagnosis

Conclusions from the previous modeling:

There is no logical contradiction to the following diagnosis:

2. (B defect, L1 ok, L2 ok, L3 defect)

Reason:

L3 may be lit in fault mode, even when there is no voltage difference.

Incompleteness of knowledge base !

Even worse:

If a rule is only applied when there are actual values for ist antecedents, then no contradiction can be found for the following diagnosis:

3. (B defect, L1 ok, L2 ok, L3 ok)

Reason:

No voltage values are computed at all.

Insufficient reasoning ability of inference component

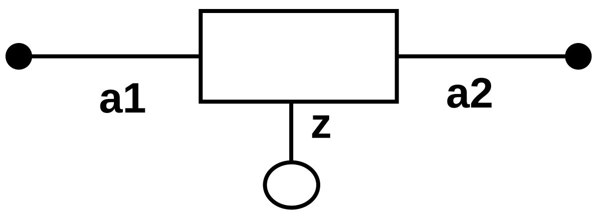
Model-Based Diagnosis

Solution of the previous dilemma:
Add additional rules in order to prevent diagnoses 2 / 3:

<u>battery:</u>

defect ⇒ (minus = `ground`)

<u>bulb:</u>



defect ∧ (a1 = `supplyVoltage`) ∧ (a2 = `supplyVoltage`) ⇒ (z = `dark`)
defect ∧ (a1 = `ground`) ∧ (a2 = `ground`) ⇒ (z = `dark`)

In order to exclude physically impossible predictions, there must be rules for faulty behaviour, too.

Model-Based diagnosis

Basic functionality:

Input:

- Set of determined values in the system (control inputs)
- Measurement of resulting values in the system (observations)

Output:

- Several diagnoses of the following kind:
  - Each diagnosis assigns exactly one behavioural mode to every component: For each component, the mode is either ok or a specified fault mode
  - The rules of all assigned modes are consistent with all input values (controlled or observed)

What do we want?

Input: see above

Output:

- A unique prescription which components have to be repaired in which way

Model-Based diagnosis

Extended functionality:

1) Proposal of control inputs

- Set certain values at certain places in the system

  (such that the expected obervations distinguish best among the currently valid diagnoses)

2) Proposal of observation points

- Select places in the system where to measure the value

  (such that the outcome distinguishes best among the currently valid diagnoses)

} **Test**

The component models must supply:

- Definition of test ports
- Definition of test values to be plugged in at test ports

} **Control actions**

- Definition of observation ports where to measure the values

Observations

Modeling components

Behavioural modes

- Feature of the component which is to be determined from the diagnostic process
- Value domain is finite (normally much less than 10 values possible)

Variables

- Value container
- The values are used in the constraints (see below).
- The constraints compute (predict) a new value for (another) variable.

Distinguish between internal and port variables!

Ports

- Contains the variables which have to be identified with the variables of the ports of another component which is connected through this port.

Constraints

- Set of behavioural rules establishing a logical connection between the variables of this component
- Typically, the antecedent of a constraint contains an assumption for the behavioural mode of this component.

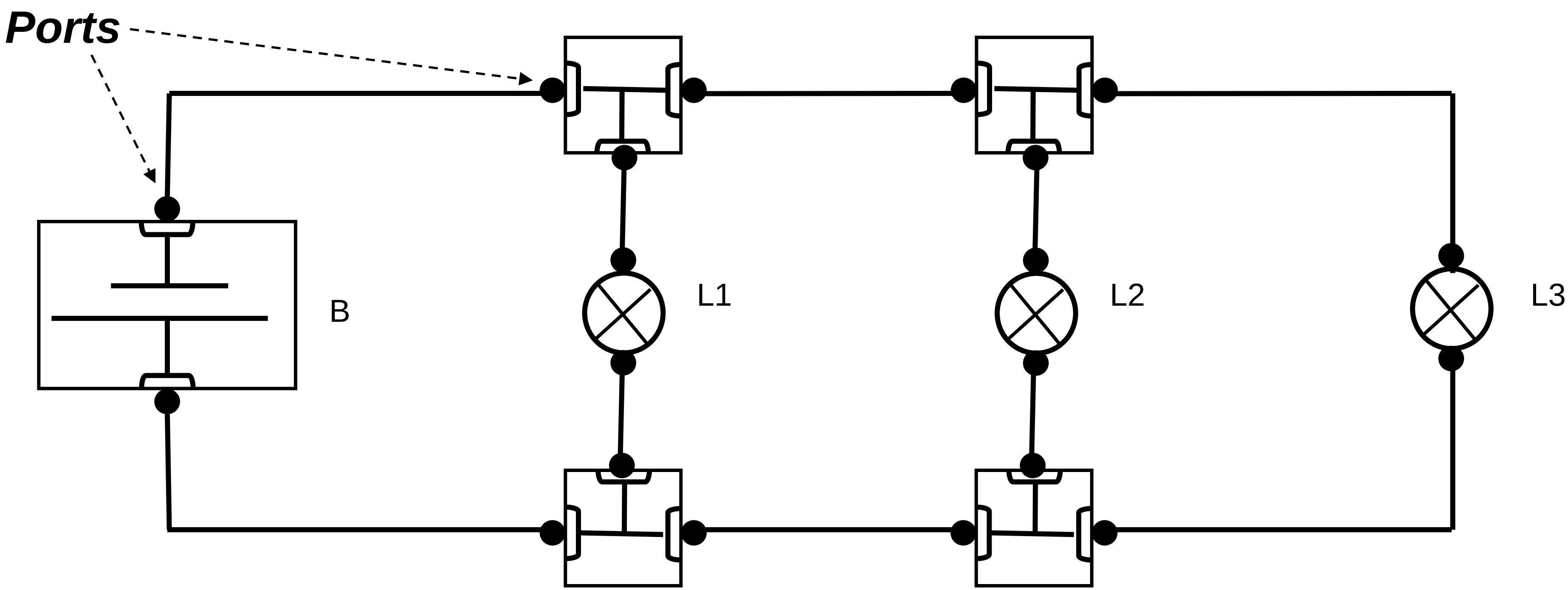
Control actions

- Variables and values to be assigned
- Level for accessibility and difficulty to set a certain value

Observations

- Variables
- Level for accessibility

Example: Model for a simple electrical system



Component types:

Battery

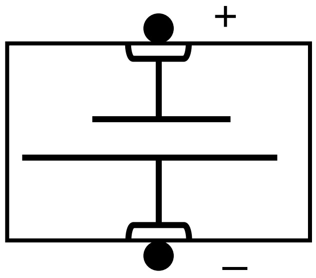
Bulb

Wire

connector (3)

Example: Model for a simple electrical system

<u>Battery</u>



Fault modes:

```
empty
no contact at +
no contact at -
loose contact at +
loose contact at -
corroded
```

Control actions:

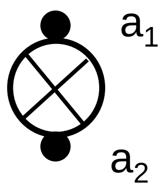
```
open terminal at +
open terminal at -
close terminal at +
close terminal at -
```

Observations:

```
inspect terminals
Measure voltage at +
Measure voltage at -
```

Example: Model for a simple electrical system

<u>Bulb</u>



Fault modes:

`blown`
`not screwd in properly`
`loose connection`
`corroded`

Control actions:

`screw off`
`screw in`

Observations:

`inspect`

<u>Wire</u>

Fault modes:

`disconnected`
`shorted to ground`
`shorted to supply voltage`
`corroded`

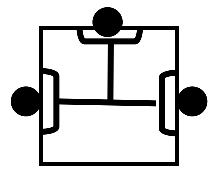
Control actions:

Observations:

measure voltage at a_1
measure voltage at a_2
inspect for disconnection

Example: Model for a simple electrical system

Connector (3)



Fault modes:

- no contact at a_1
- no contact at a_2
- no contact at a_3
- loose contact at a_1
- loose contact at a_2
- loose contact at a_3

Control actions:

- close contact at a_1
- close contact at a_2
- close contact at a_3
- open contact at a_1
- open contact at a_2
- open contact at a_3

Observations: inspect contacts