Fault Diagnosis in the Automotive Industry

Adj. Prof. Mattias Nyberg
Royal Institute of Technology (KTH), Stockholm, Sweden
Scania, Stockholm, Sweden

Keynote at SAFEPROCESS 18, Aug 29-31, 2018, Warsaw, Poland.

Mattias Nyberg at KTH

- **Adjunct professor in ”Dependable control systems”**
- **Division of ”Mechatronics and embedded control systems”**
- **Leading a research group in ”Rigorous Systems Engineering”**

The whole presentation is available on my KTH webpage:
https://www.kth.se/profile/matny

SCANIA

- Heavy trucks and buses
- Worldwide production and sales
- 50 000 employees
- 5000 engineers in total
- 2000 engineers in electronics and software
- 100 000 sold vehicles per year
- Vehicles in operation:
  - > 1 000 000
  - 300 000 connected

SCANIA

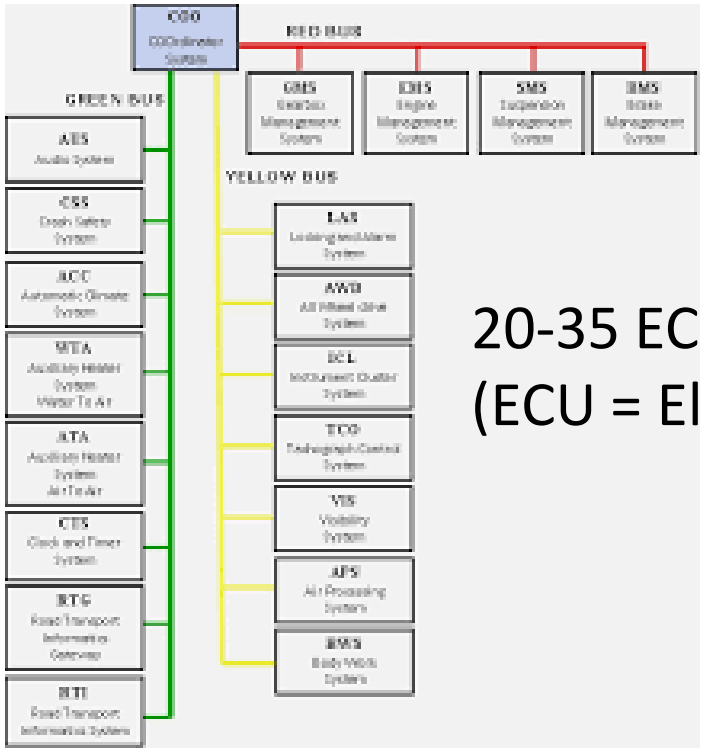
What's in the talk?

- Historical notes
- Overview – Perspectives - Principles
- State of practice
- Personal reflections and experiences from working 23 years with automotive diagnosis
- Future
- What are the challenges?

Automotive Electronics

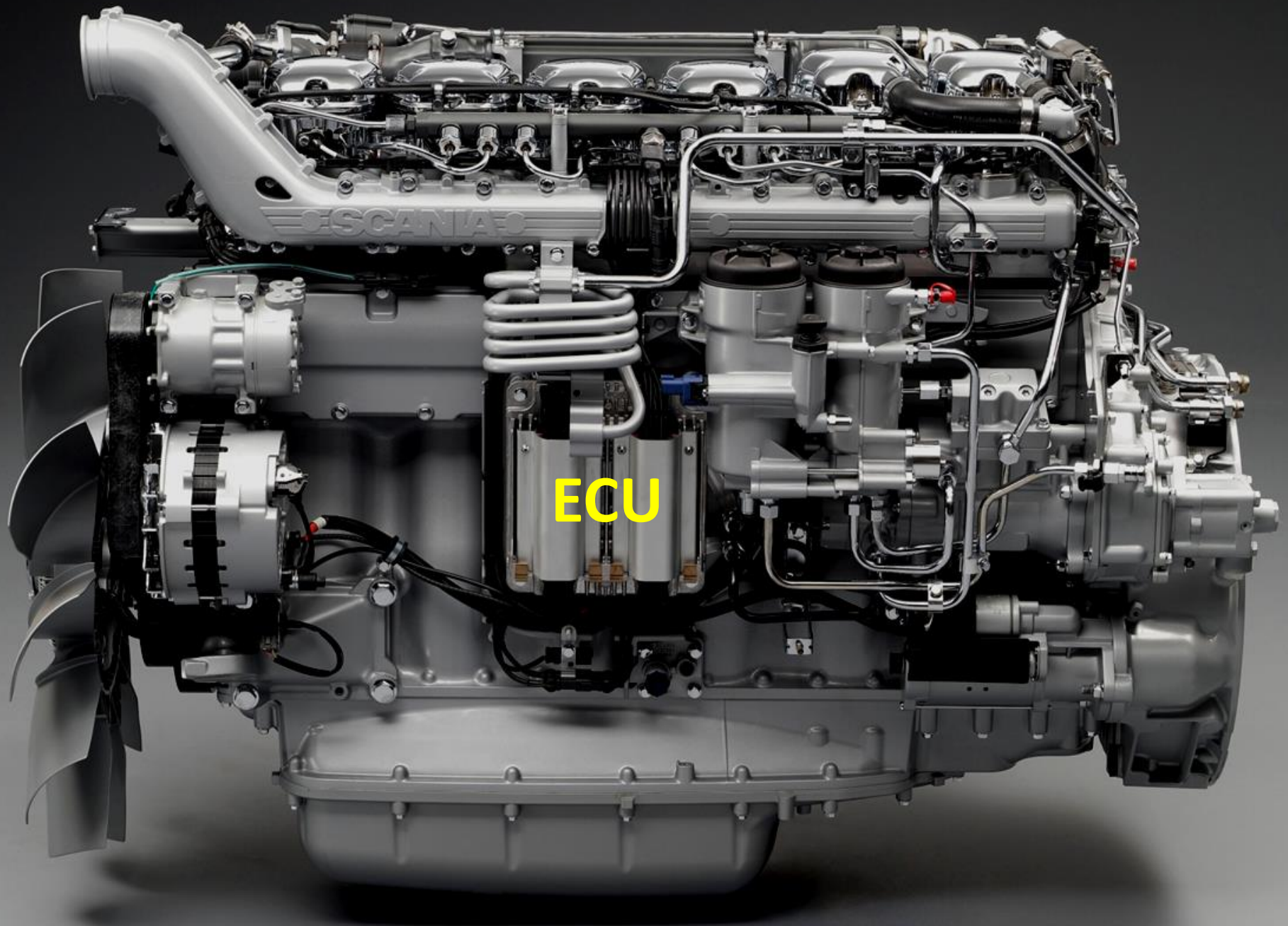
- 1897 Pope Manufacturing Company: first electrical car
- 1968 Volkswagen 1600 TE & LE: first electronic Engine Control Unit with 25 transistors.
- 1971 First microprocessor Intel 4004
- 1978 Cadillac Seville "trip computer“: first microprocessor in cars.
- 1980 GM’s Assembly Line Diagnostic Link (ALDL) **to read out fault codes**

Electric System of a Scania vehicle

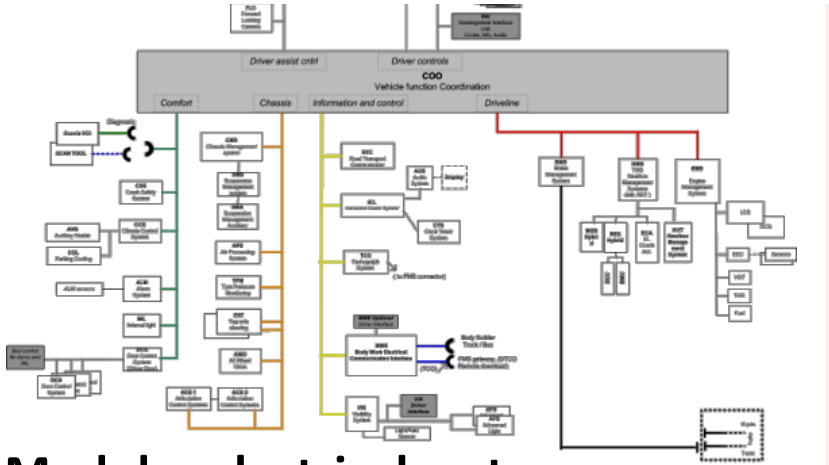


20-35 ECUs
(ECU = Electronic Control Unit)

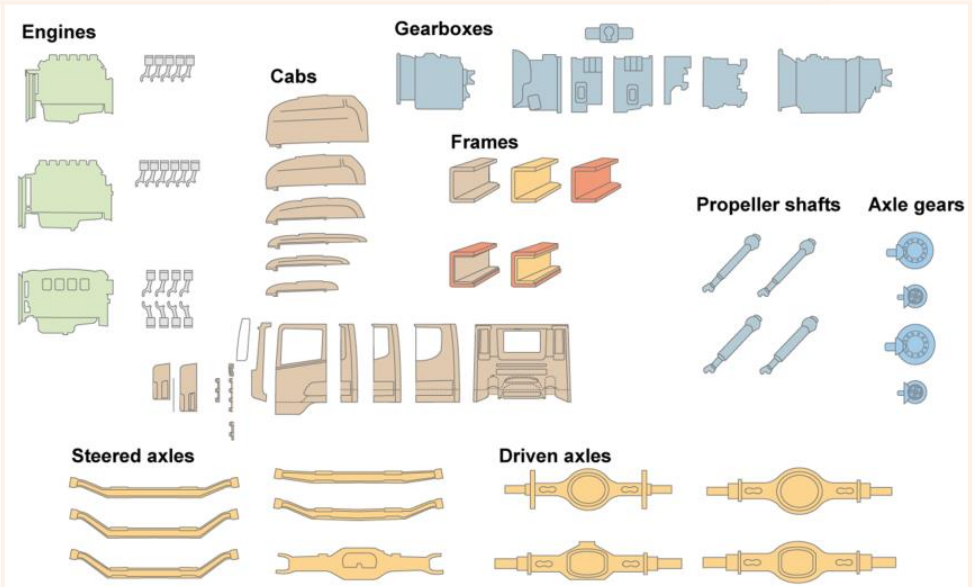
In passenger cars: up to **150** ECUs !


SCANIA
ECU

Evolving Product Lines



Modular electrical system



Modular chassi and drivetrain

Every week, some parts of the system are changed.

Automotive "Fault Diagnosis" Four Use-Cases !

	Mattias' Experience:
• Standardized legislative OBD (On-Board Diagnostics)	1995-2009
• Fault tolerant control	2001-2009
• Troubleshooting	2007-2011
• Safety Mechanisms for Functional Safety	2010-

Standardized Legislative OBD (On-Board Diagnostics)

OBD (On-Board Diagnostics)

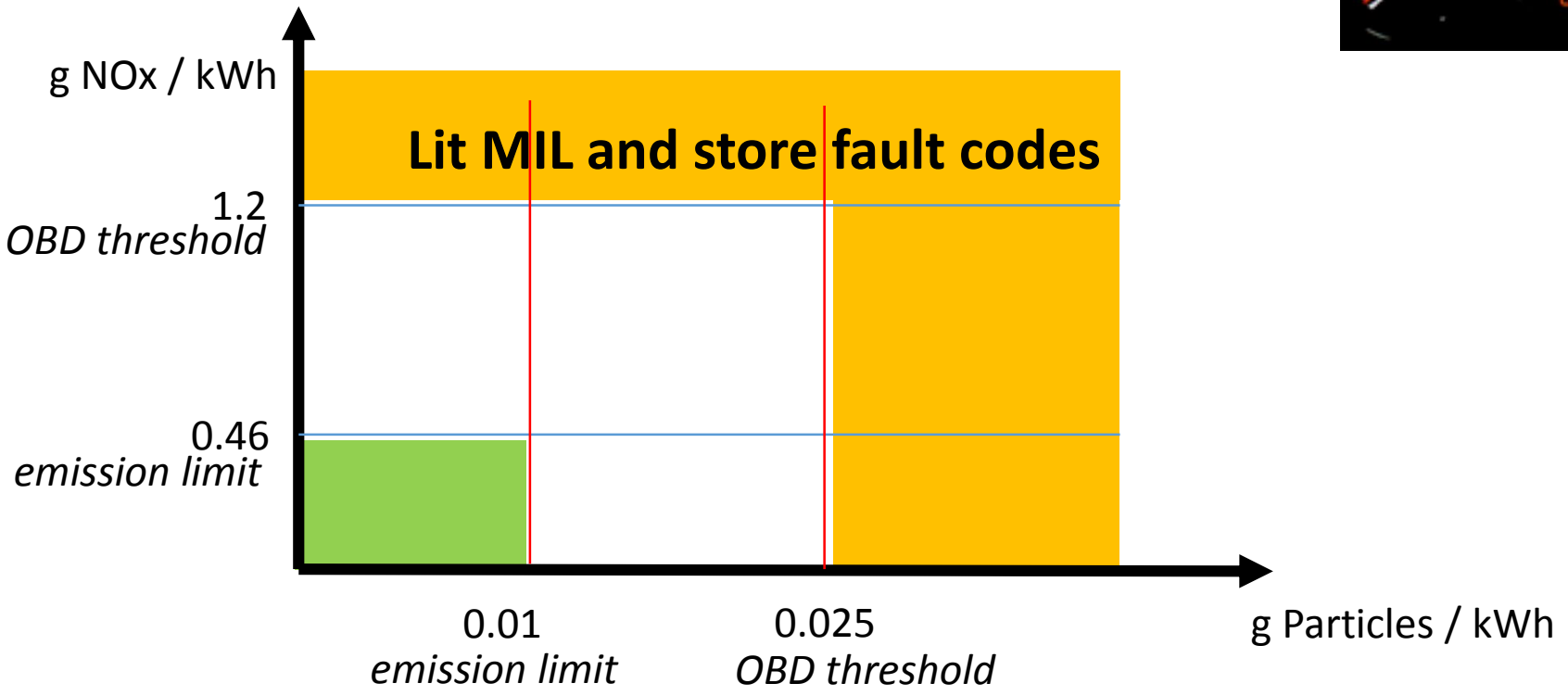
- Monitor reliability/availability of emission control systems with respect to random HW faults.
- Part of emission regulations
- Monitor “tampering” (security), and if detected, activate inducement (e.g. lower engine torque)

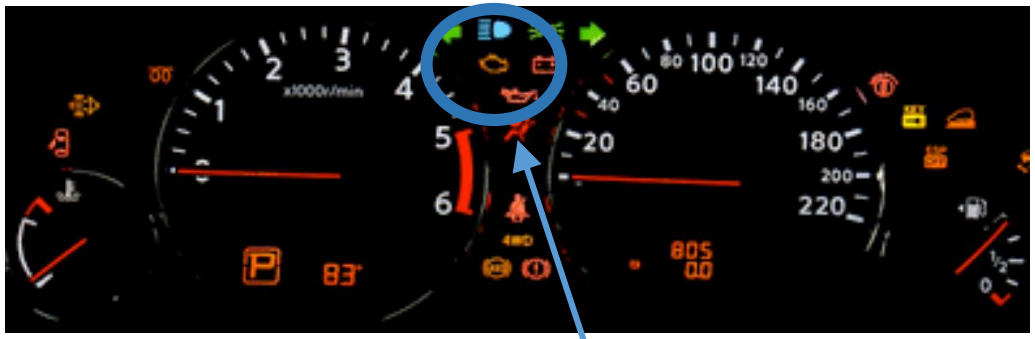
Timeline

- 1980 GM’s Assembly Line Diagnostic Link (ALDL)
- 1991 OBD for cars in California
- 1996 OBD II for cars in whole USA
- 2001 EOBD for passenger cars in Europe
- 2006 Euro IV -- OBD for heavy-duty trucks in Europé
- 2010 HD-OBD for heavy-duty trucks in USA
- …

OBD Principle

Example EuroVI HD-OBD:

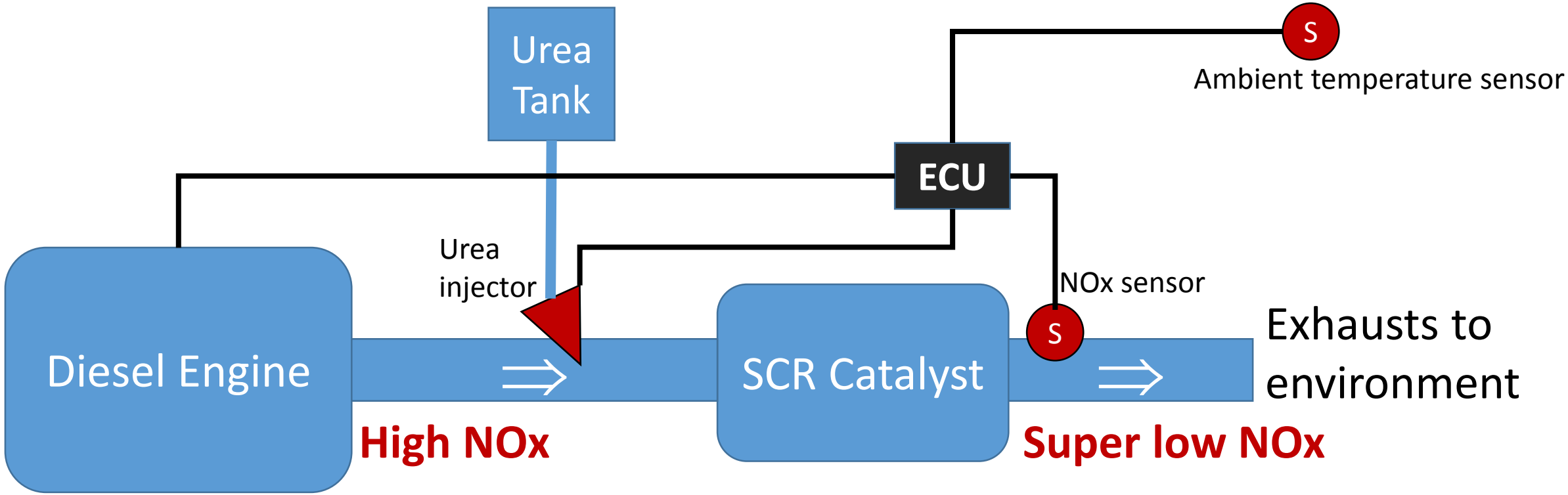




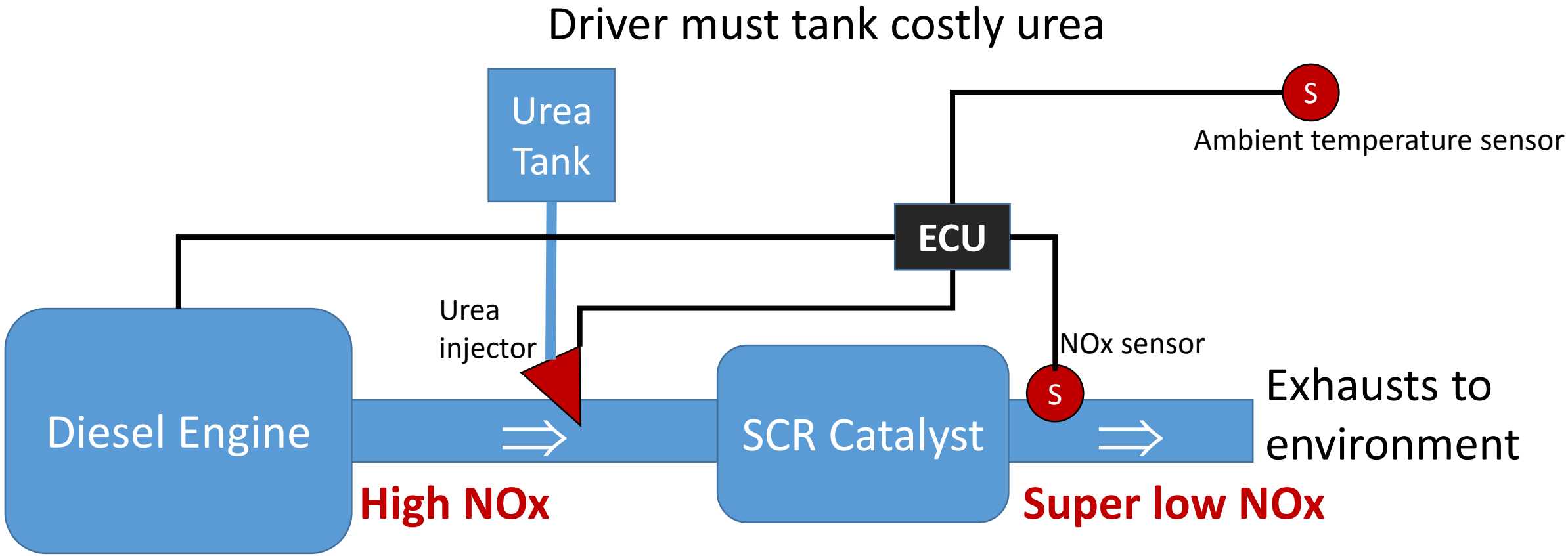
MIL = Malfunction Indicator Light

Nox Emission Control

$$2NO_2 + 4NH_3 + O_2 \rightarrow 3N_2 + 6H_2O$$



Tampering Monitoring



Tampering monitoring: the likely faults are the faults not monitored.

Reflections

- False detections must be avoided.
- What OBD used to be about:
  - follow regulations, and certification.
- After ”dieselgate” : Detect when **real-world** emissions are above thresholds

- Fault isolation is not important

- Next step: To make tampering monitoring to work…

Ad-Blue Off Kit:
1150 €

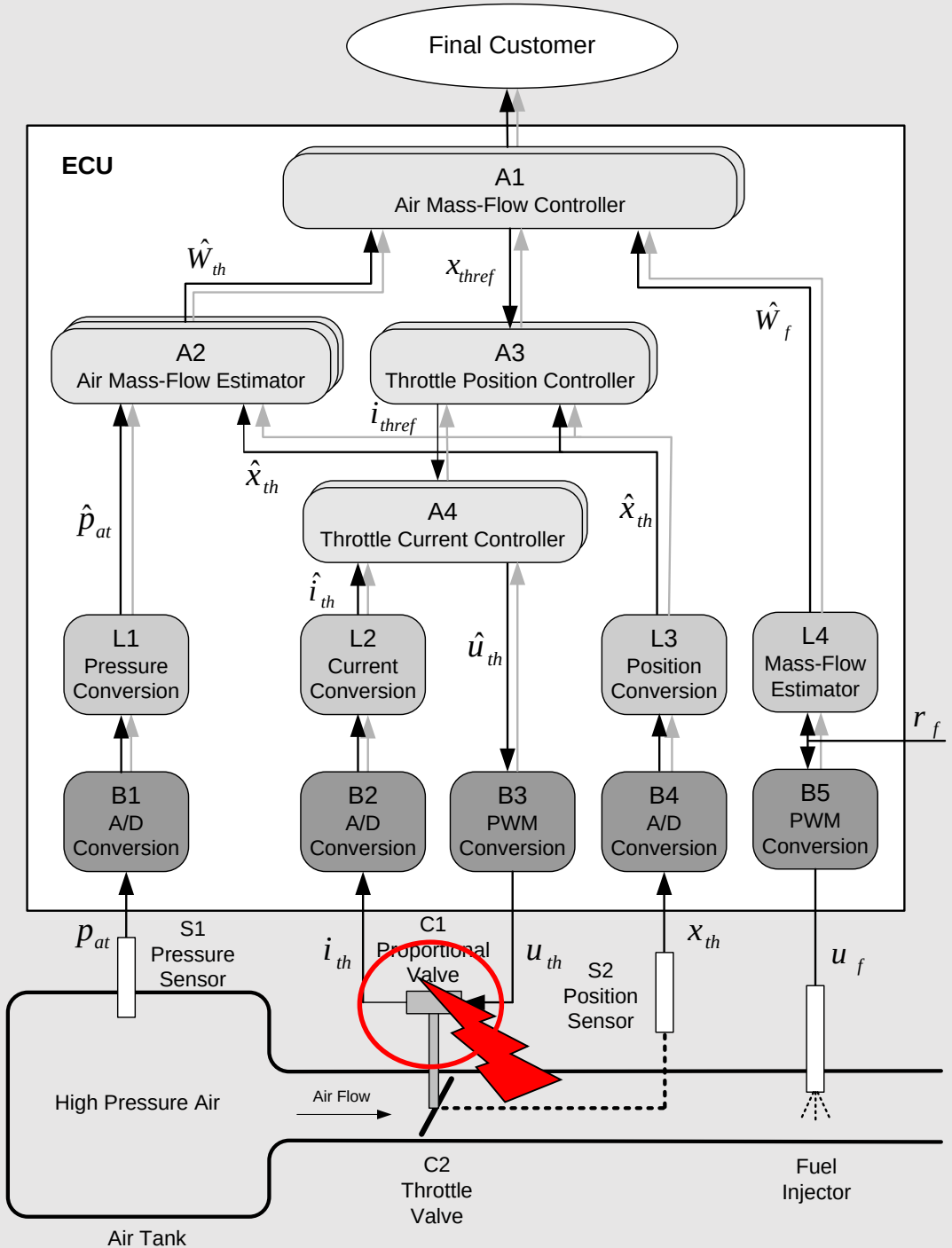
Fault Tolerant Control (FTC)

Fault tolerant control

- Applied from the beginning of automotive microprocessors ≈1978
- To ensure safety and availability;
  The purpose is to stop faults from propagating and develop into failures that:
  - cause accidents
  - damage the vehicle
  - stops the vehicle operation

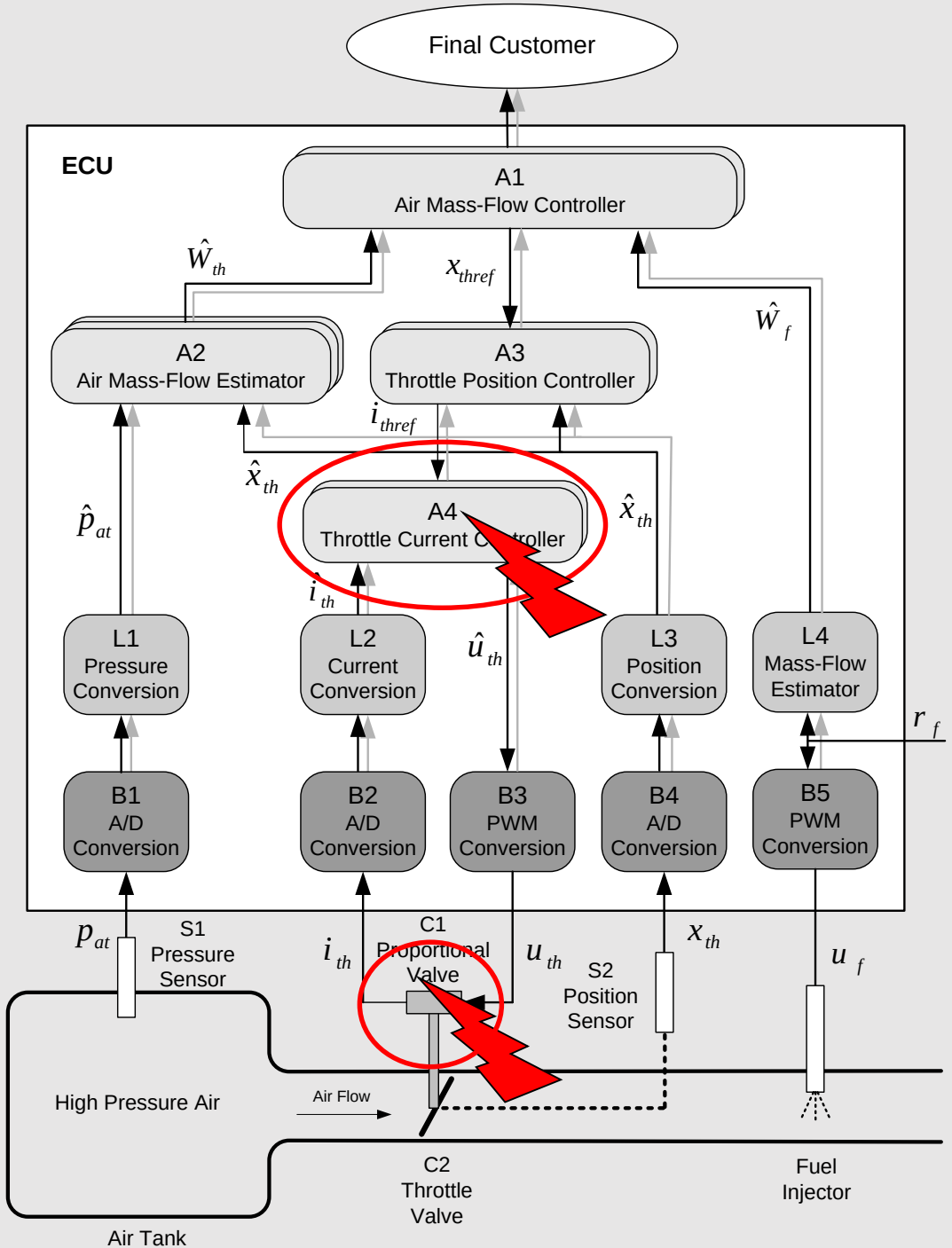
Failure Propagation

Note that ECU SW often has a hierchical structure following the ”cascade control” pattern.



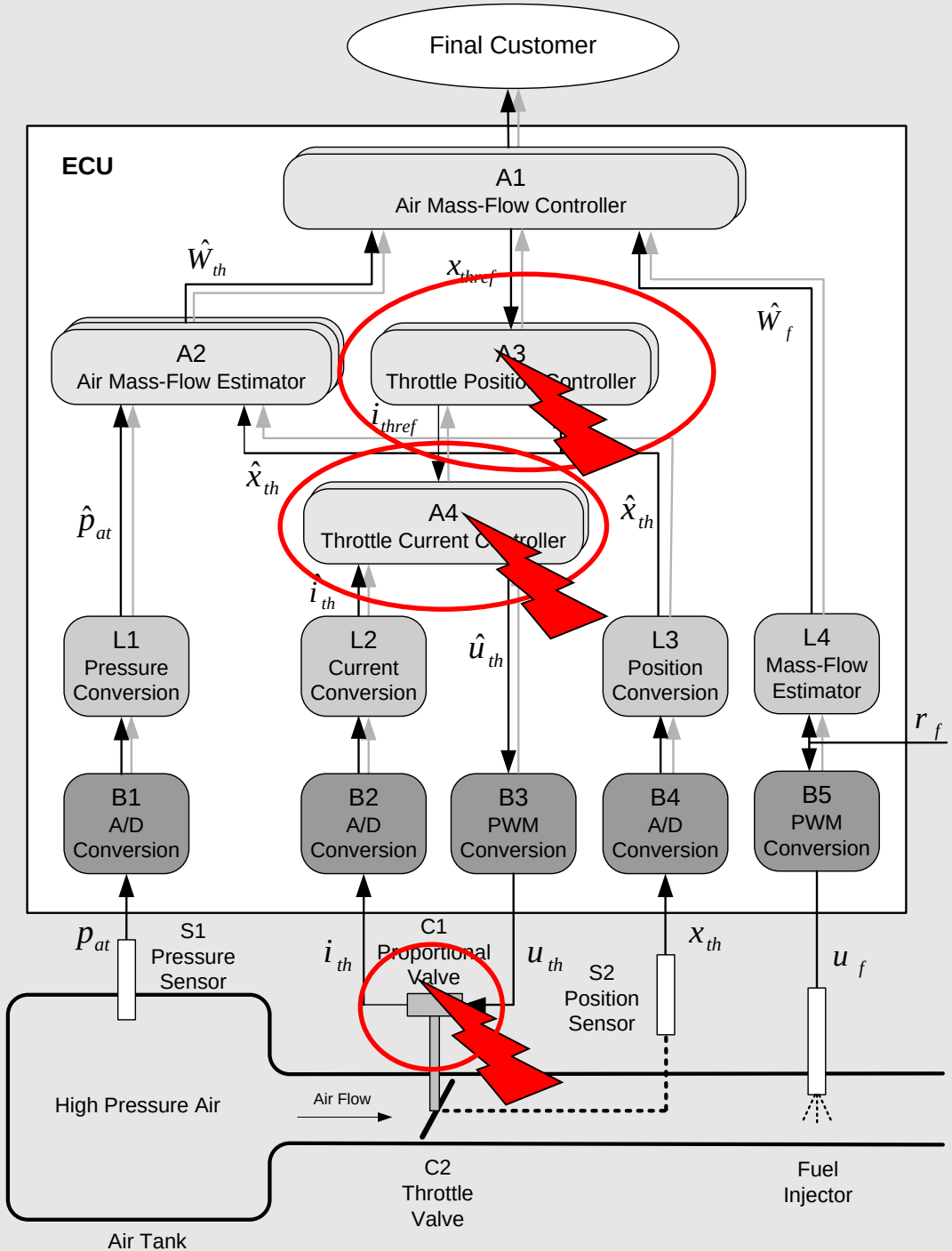
Failure Propagation

Note that ECU SW often has a hierchical structure following the ”cascade control” pattern.



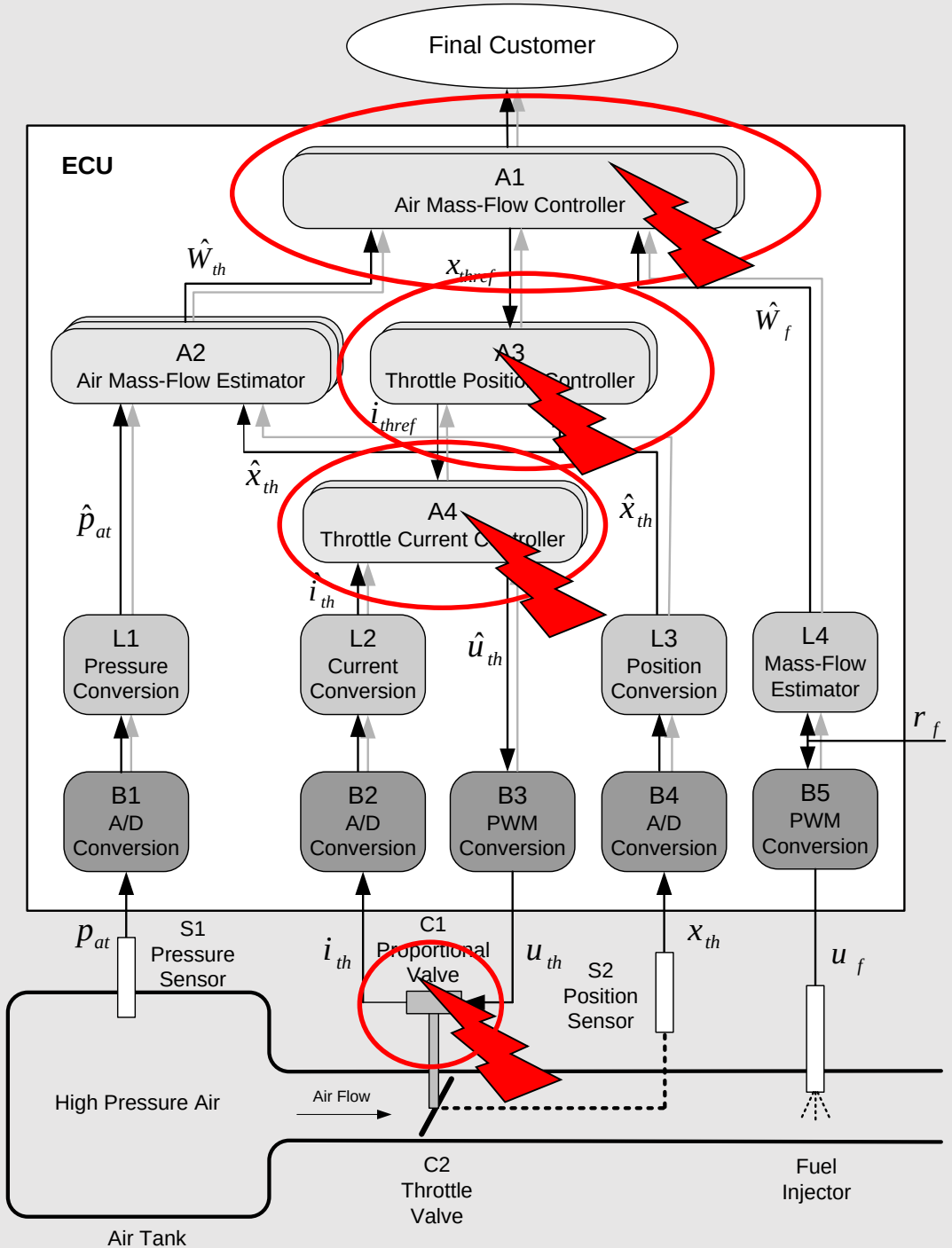
Failure Propagation

Note that ECU SW often has a hierchical structure following the ”cascade control” pattern.



Failure Propagation

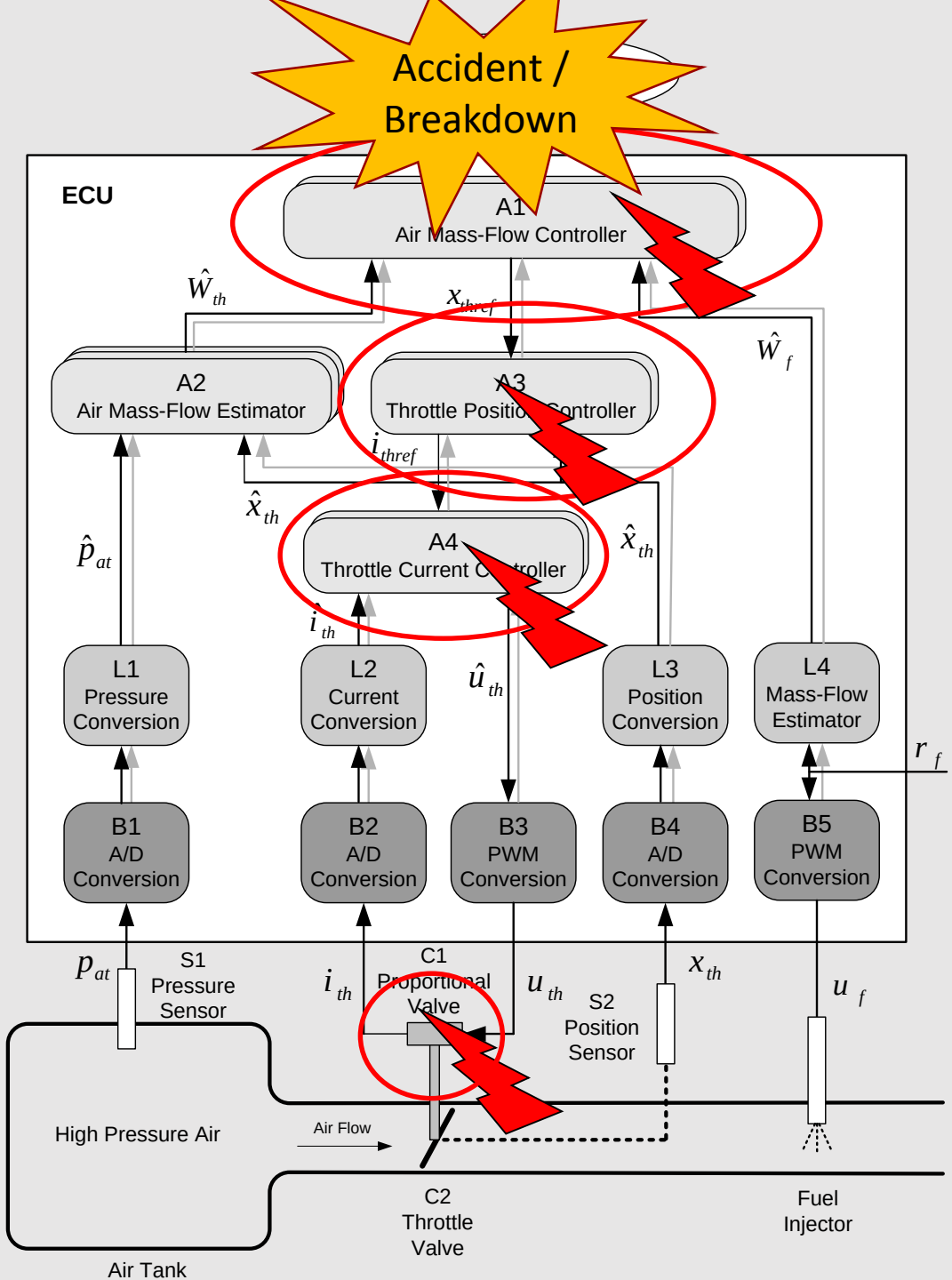
Note that ECU SW often has a hierchical structure following the ”cascade control” pattern.



Failure Propagation

Note that ECU SW often has a hierchical structure following the ”cascade control” pattern.

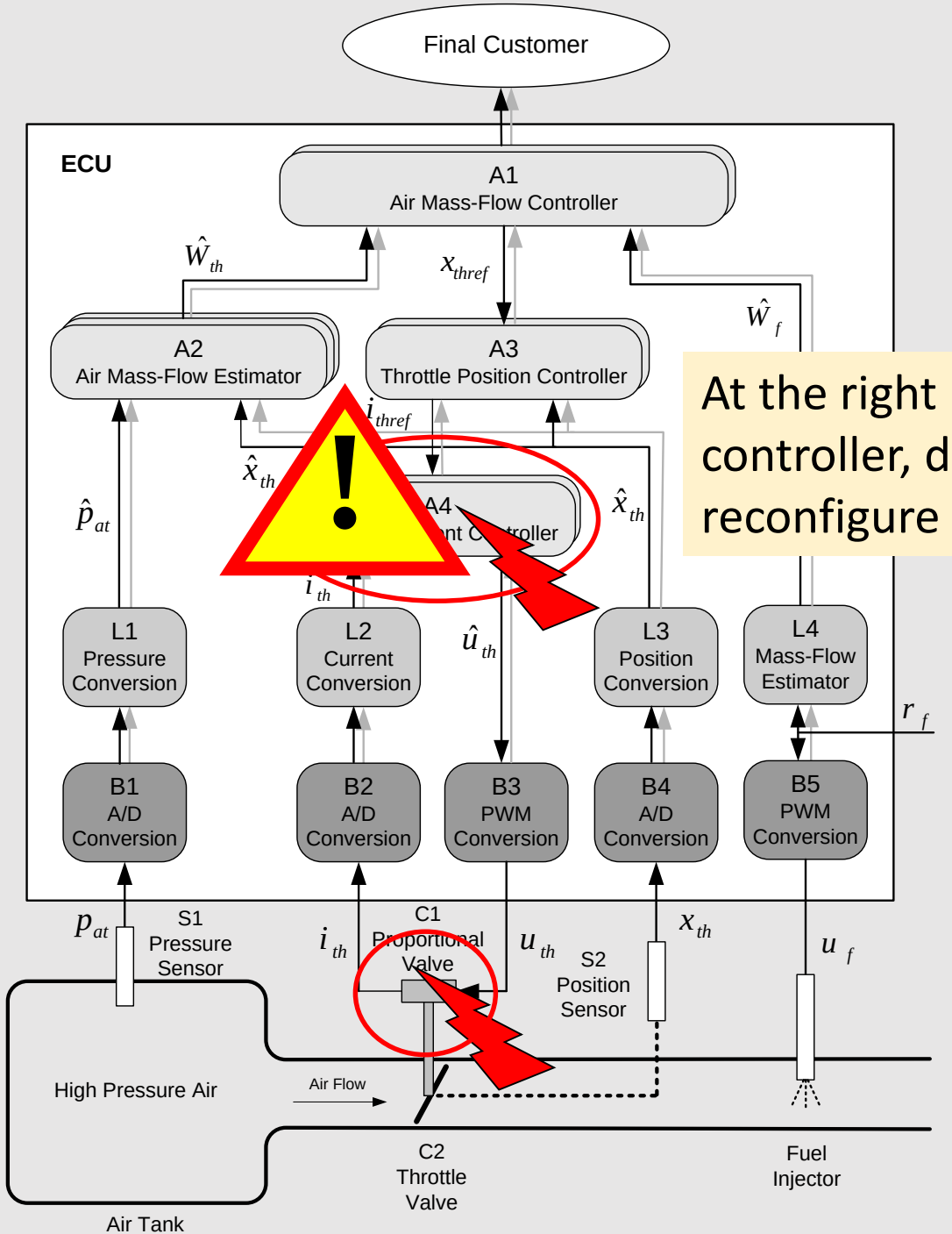
Note that the fault propagation follows the structure of cascade controllers rather than signals.



Failure Propagation

Note that ECU SW often has a hierchical structure following the ”cascade control” pattern.

Note that the fault propagation follows the structure of cascade controllers rather than signals.

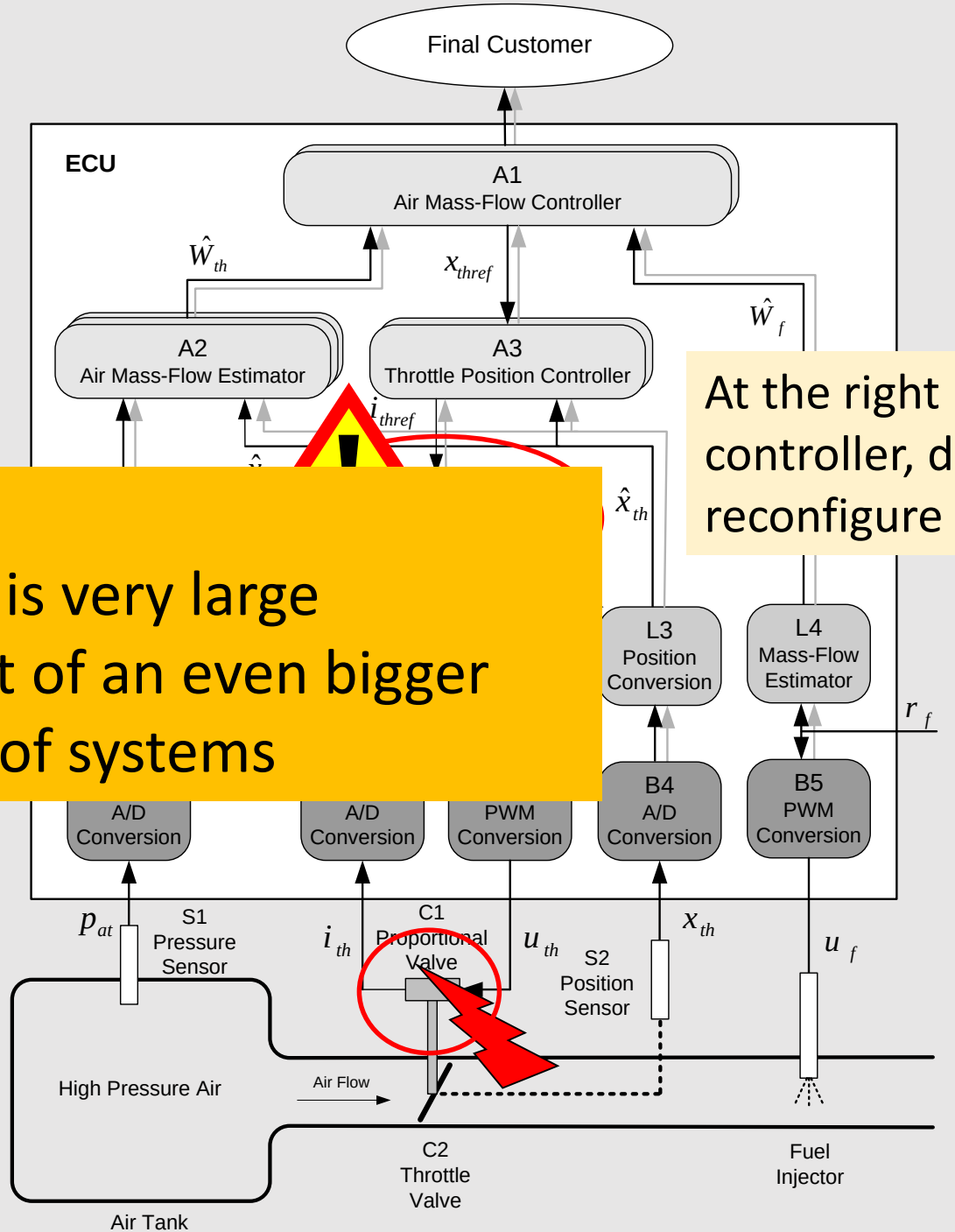


At the right level of controller, detect and reconfigure the controller.

Failure Propagation

Note that ECU SW often has a hierchical structure following the ”cascade control” patter

Note that the fault propagat follows the structure of casc controllers rather than signa

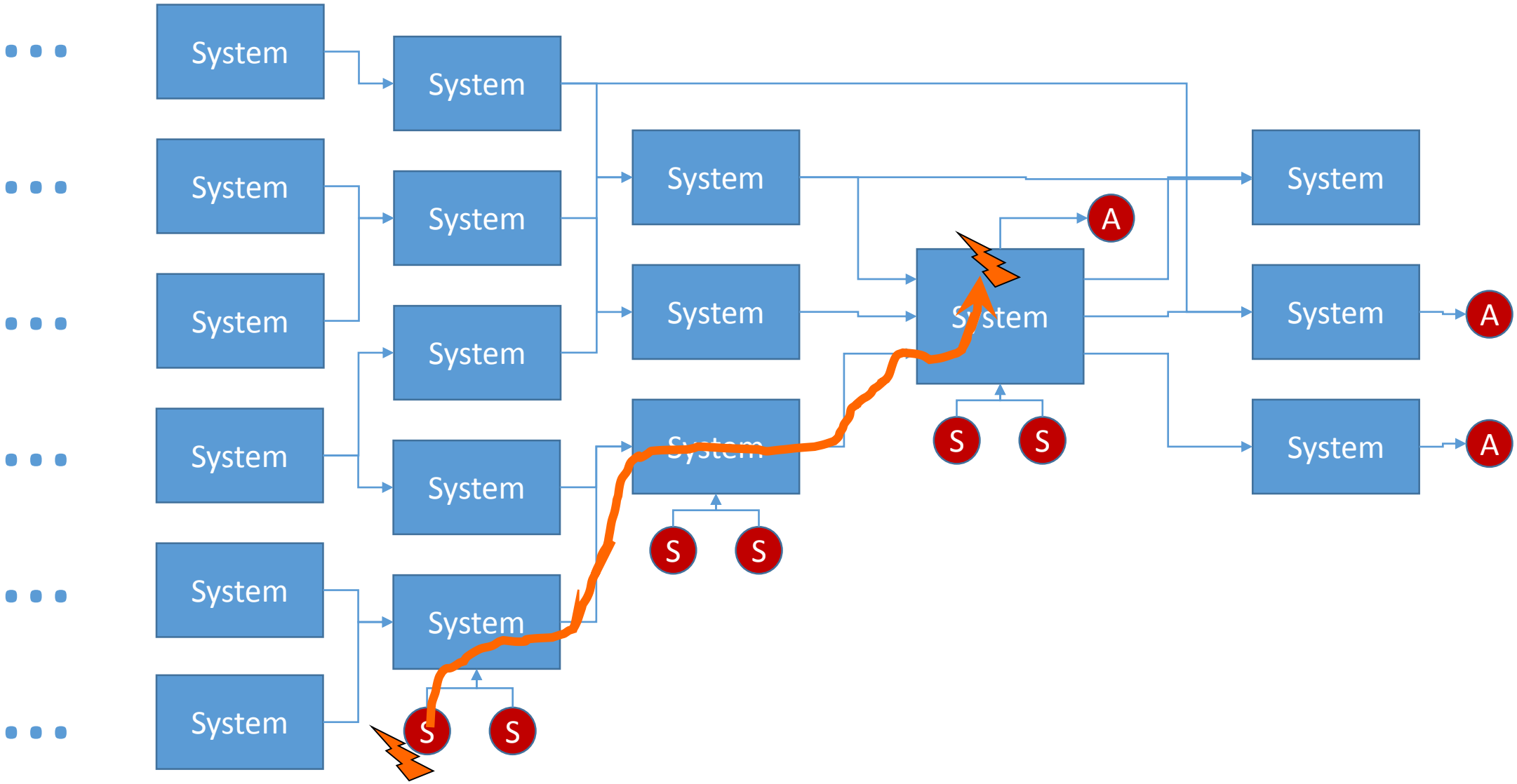


But:

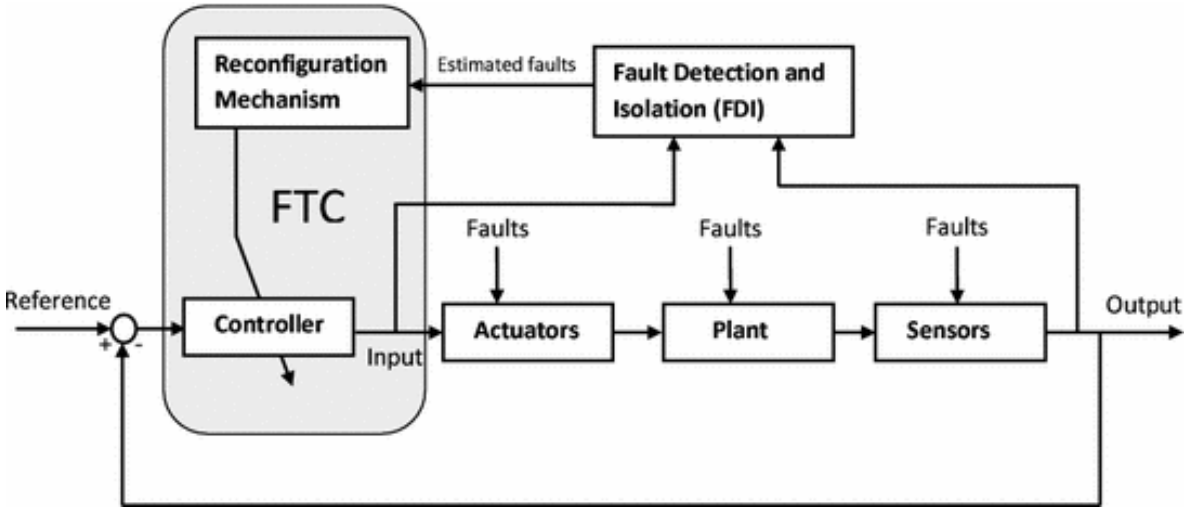
- the SW is very large
- it is part of an even bigger system of systems

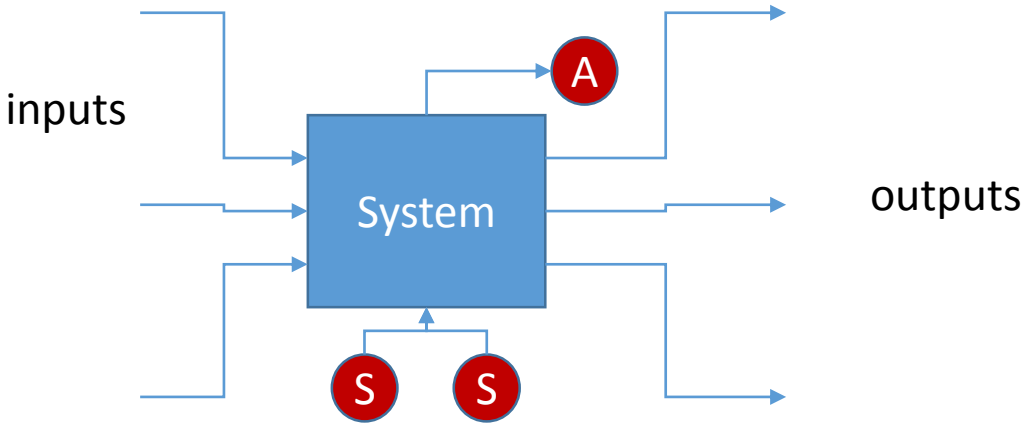
At the right level of controller, detect and reconfigure the controller.

System dependencies unfolded



Compared to ”traditional” FTC

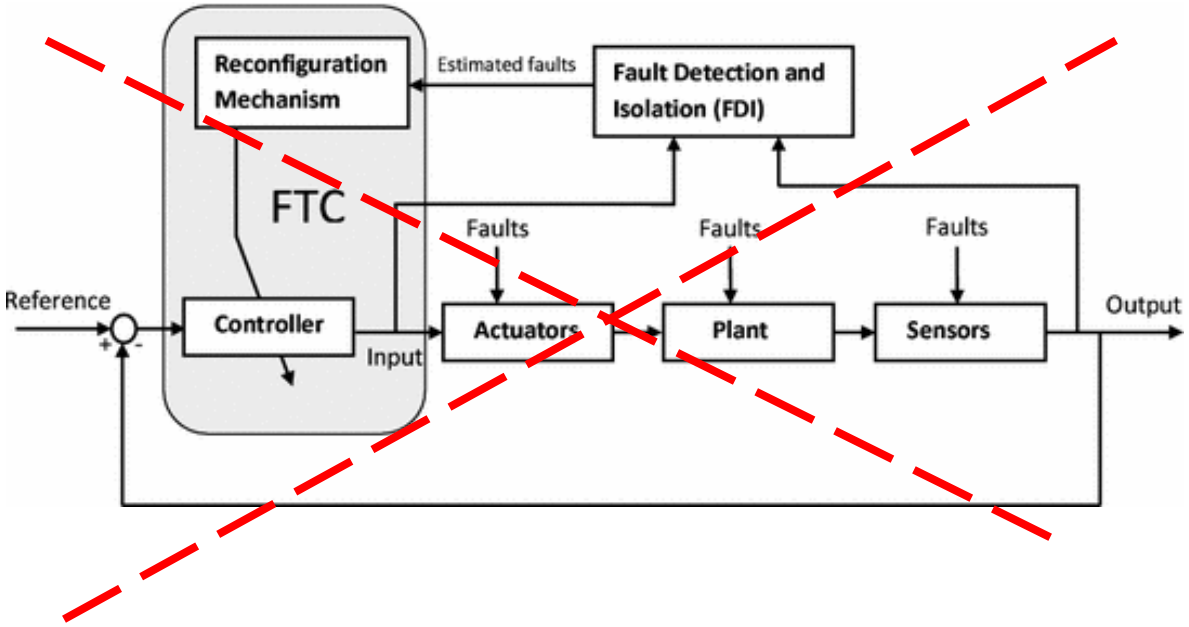


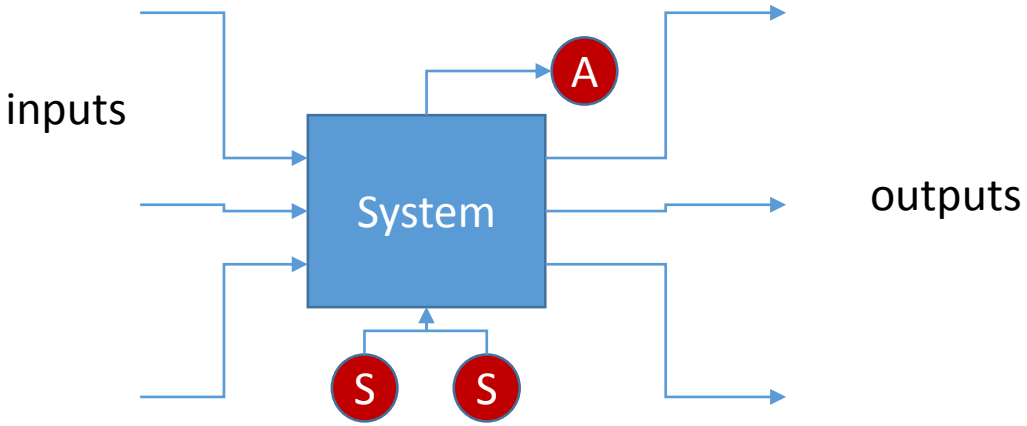


For the system:

- the set of sensors and actuators belonging to other systems are **unknown** and **differ** between different configurations
- the set of possible faults of other systems are **unknown** and **differ** between different configurations

FTC in a Modular Architecture



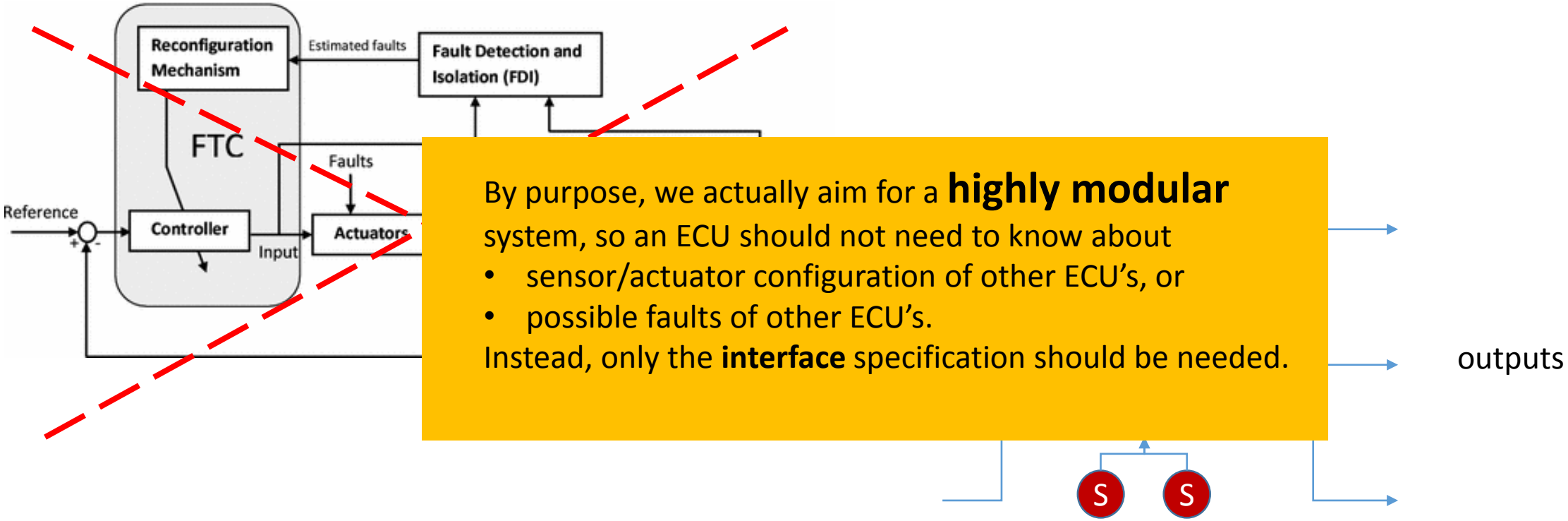


For the system:

- the set of sensors and actuators belonging to other systems are **unknown** and **differ** between different configurations
- the set of possible faults of other systems are **unknown** and **differ** between different configurations

⇒ A distributed view on fault-tolerant control (and FDI) is needed.

FTC in a Modular Architecture



By purpose, we actually aim for a **highly modular** system, so an ECU should not need to know about

- sensor/actuator configuration of other ECU's, or
- possible faults of other ECU's.

Instead, only the **interface** specification should be needed.

For the system:

- the set of sensors and actuators belonging to other systems are **unknown** and **differ** between different configurations
- the set of possible faults of other systems are **unknown** and **differ** between different configurations

⇒ A distributed view on fault-tolerant control (and FDI) is needed.

Faults are caused not only by HW problems

- Calibration errors
- SW Bugs
- Radiation causing bit-flips in the microprocessor

Example

vehicle weight parameter is incorrectly **calibrated**
$\Rightarrow$ vehicle pitch angle is incorrectly calculated
$\Rightarrow$ front-looking radar identifies a bridge as an obstacle in front of vehicle
$\Rightarrow$ emergency brake is activated
$\Rightarrow$ car behind crashes into the vehicle

Should the fault-tolerant control system deal with such ”faults”?

How, and where, to detect the fault?

How to isolate the cause of the fault?

How do we even know that the vehicle weight parameter needs to be monitored?

Troubleshooting

Troubleshooting

- Troubleshooting = trace and correct faults in a mechanical or electronic systems.

- The oldest form of automotive diagnosis. Carried out since the first car in 1769.

- Computer supported troubleshooting based on fault codes came with OBD around 1980.
- Troubleshooting is the main reason why fault codes are stored in ECUs.

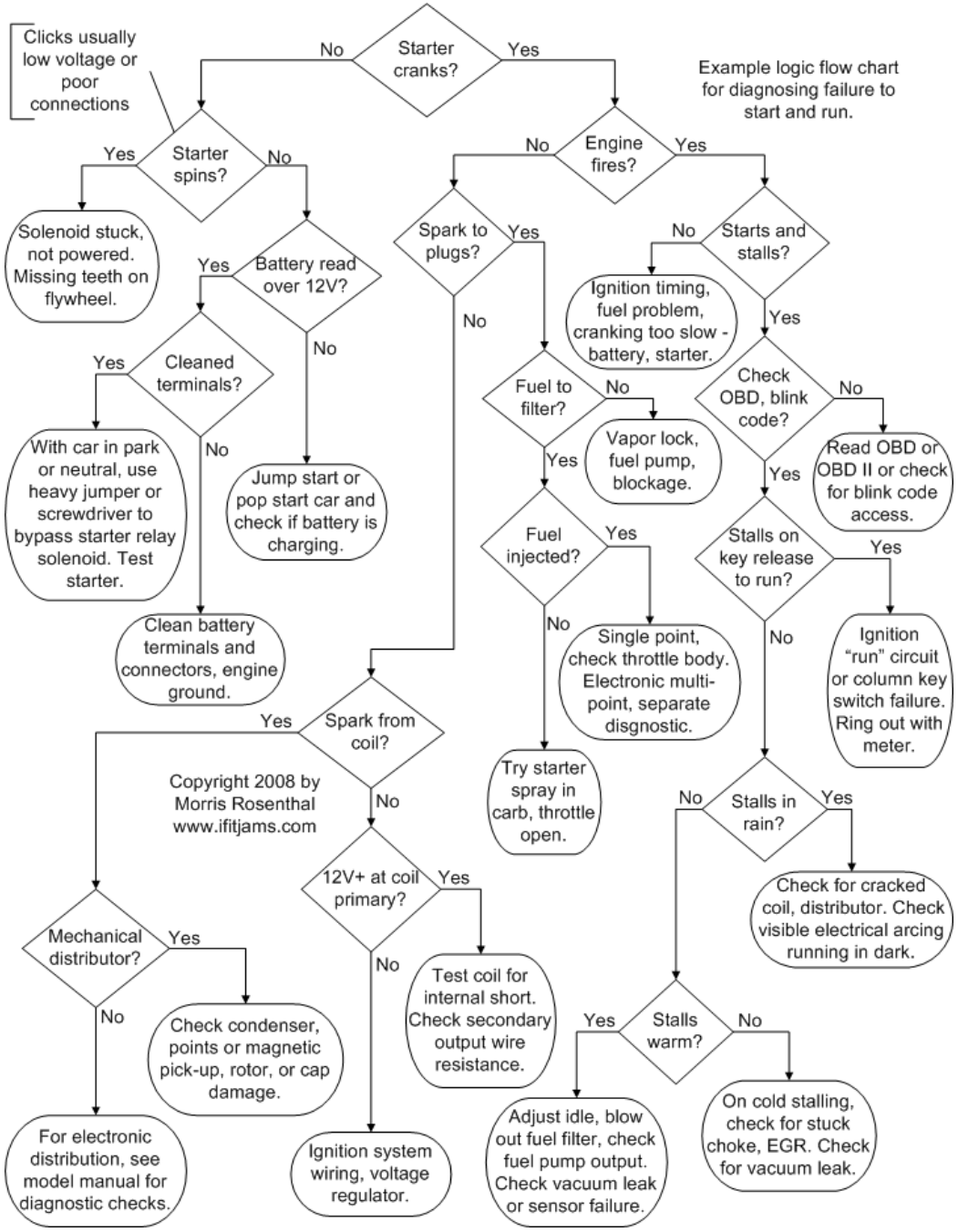
Static Decision Trees for Troubleshooting

State-of-practice

- computerized
- connection to vehicle enables:
  - filtered decision tree based upon fault codes
  - execution of built-in-tests

Problems

- creation
- maintenance



Trends

- Connectivity
- Remote diagnosis– diagnose the fault without visit to workshop
- Model based creation of static decision trees
- AI-search based troubleshooting
  - Bayesian networks
  - Failure propagation models
- Prognostics – predict fault before it occurs

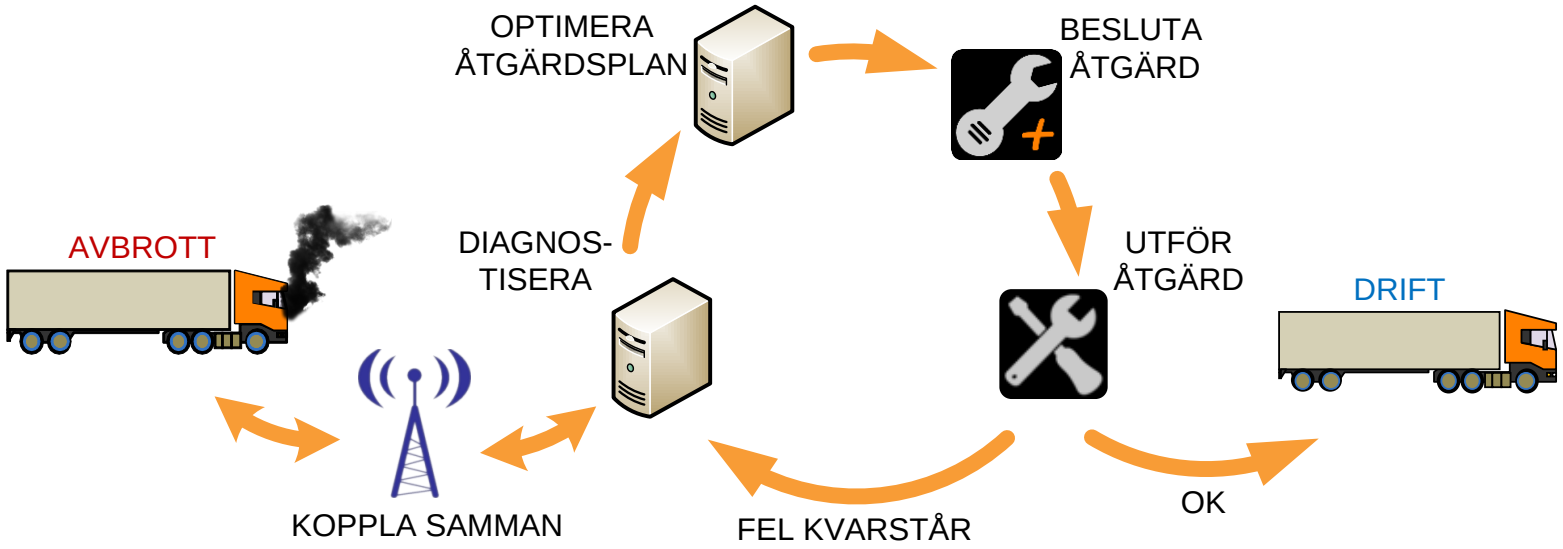
Commercial solutions exist already!

Remote Computer-Supported Diagnosis

At detection of problem, use AI-search to find the optimal plan including:

- actions by driver
- continue to drive or stop
- visit to workshop
- actions by the mechanic

in order to fix the vehicle with minimal interuption of operation.



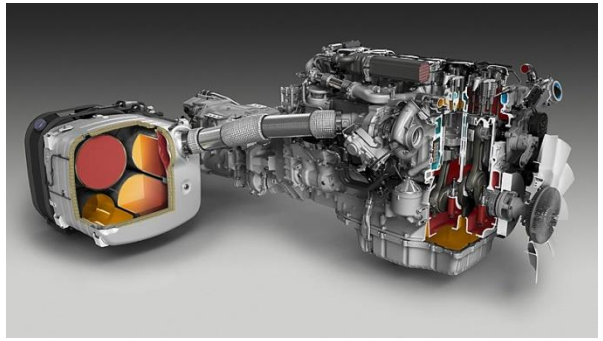
Troubleshooting in
1500 workshops worldwide

replaced by

5 remote troubleshooting centers worldwide with strong computer support

Diagnosis and troubleshooting without fault codes – Is there a fault present?

Scania Eu VI
- DTC 1049: ”The measured nitrogen oxide content after the catalytic converter is higher than it should be.”
- Mechanic: How to fix?



Volvo *Adaptive Cruise Control* with *Queue Assist*
- Driver: ”My vehicle drives too close to the vehicle in front!”
- Mechanic: How to fix?

Scania *Driver Support*
- Driver: ”The vehicle gives me too low scores!”
- Mechanic: How to fix?

Safety Mechanisms in Functional Safety

Functional Safety

ISO 26262

- Originates from the area of:
  - dependability (reliability, availability, safety, etc.)
  - critical software development
  - fault-tolerant computer systems

- Purpose is to provide **evidence** for that computerized functions of the vehicle are safe.
- If all functions are safe but there is not an evidence in a **standardized format**, then the system does not comply with ISO 26262.

- Diagnosis and fault tolerant control are fundamental parts.

A Common Ancestor !

1974: European Workshop on Industrial Computer Systems TC7

IEC61508
1998

ISO 26262
2011

SAFECOMP

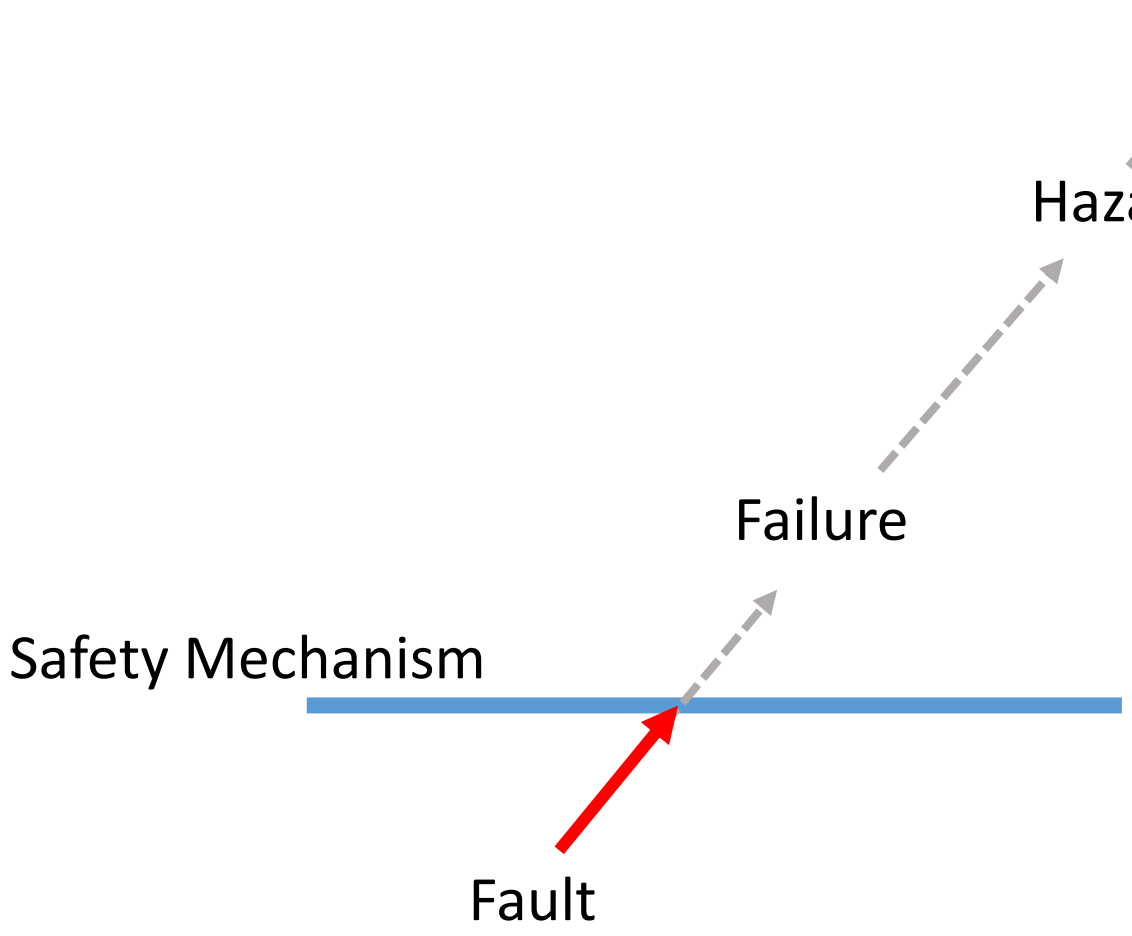
1979: IFAC/IFIP Workshop on Safety of Computer Control Systems

safeprocess

1991: Baden-Baden

IFIP=International Federation for Information Processing

"Safety Mechanism"



The safety mechanism should stop propagation from fault to failure by:

- detecting the fault
- transition to safe state

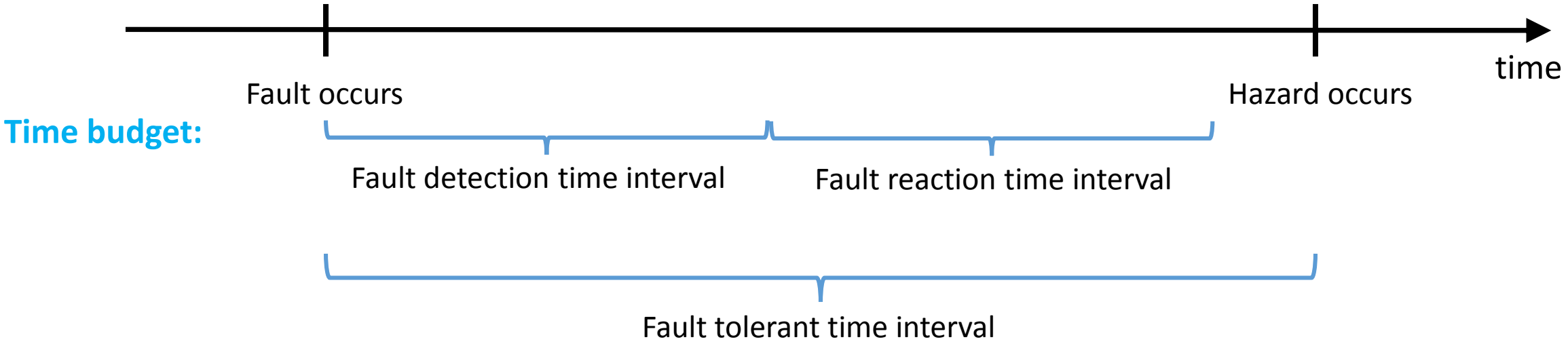
Evaluation method

ISO26262 provides a detailed method for evaluating if a **safety mechanism** of an element is **sufficiently efficient**.

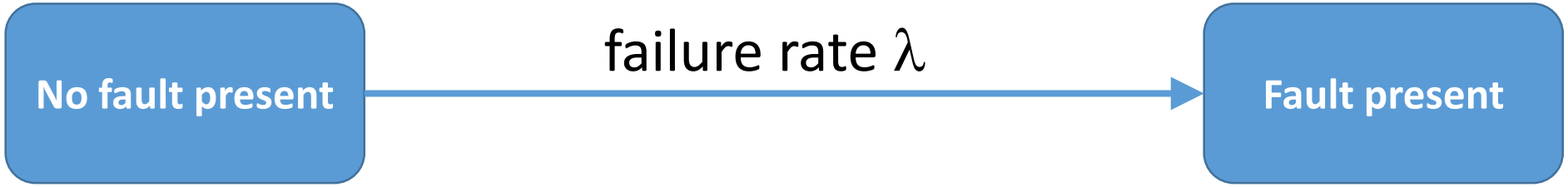
1. identify **maximum allowed failure rate** of the element to avoid hazard
2. identify and **classify** all faults of the element
3. identify actual **failure rates** of each fault
4. identify **diagnostic coverage** of each fault
5. use a formula to compute **actual failure rate** of the element
6. make sure **actual** is lower than **maximum allowed failure rate**

Diagnostic Coverage

DC(F) = P(detect fault within the maximum fault detection time interval | fault F present)



Continuous-Time Markov Chains



failure rate $\lambda \approx P(\text{fault within 1h} \mid \text{no fault present})$

$P(\text{fatality}) = \ldots = \Sigma i\ (1\text{-}P(\text{no failure of element} \mid \text{fault i}))P(\text{fault i}) = \Sigma i\ (1 - DC(Fi))\ \lambda i < 10\text{^-}9$

Current Issue

- How to make engineers change their way of working, to
  - become more rigorous
  - follow established patterns instead of being creative
  - write documents
  - write requirements

Change in engineering culture is needed.

The Future

[Semi-] Autonomous Vehicles

- The **functions of important sensor components** are inheritly unreliable.
  E.g. radars and cameras? *Can they be diagnosed?*

- How to troubleshoot the root-cause why a neural network took wrong decision?

- Not anymore only the vehicle; **transport systems** and **platooning** are new applications for automotive diagnosis.

- Current challenges:
  - Troubleshooting without detected faults
  - Correct and efficient troubleshooting
  - Provide evidence of safety
  - etc.

  will become more critical.

The challenges

Summary of Noted Challenges

- Standardized legislative OBD (On-Board Diagnostics)
  - Monitor tampering: Detect all faults that can disable urea injection
  - Detect **all** likely faults that can cause increased emissions
  - Avoid false detections
- Fault tolerant control
  - In a huge system of systems, how to design modular FTC for one system without knowledge of other systems?
  - How to detect faults not caused by HW problems.
- Troubleshooting
  - Troubleshooting without fault codes
  - Is there a fault present?
- Safety Mechanisms in Functional Safety
  - Change engineers towards more rigorous work

Summary of Noted Challenges

- Standardized legislative OBD (On-Board Diagnostics)
  - Monitor tampering: Detect all faults that can disable urea injection
  - Detect **all** I
  - Avoid false
- Fault tolerant
  - In a huge sy ystem without knowledge
  - How to detect faults not caused by HW problems.
- Troubleshooting
  - Troubleshooting without fault codes
  - Is there a fault present?
- Safety Mechanisms in Functional Safety
  - Change engineers towards more rigorous work

But,
there is something more…

Summary of Noted Challenges

- Standardized legislative OBD (On-Board Diagnostics)
  - Monitor tampering: Detect all faults that can disable urea injection
  - Detect **all** likely faults that can cause increased emissions
  - Avoid false detections
- Fault tolerant control
  - In a huge system of systems, how to design modular FTC for one system without knowledge of other systems?
  - How to detect faults not caused by HW problems.
- Troubleshooting
  - Troubleshooting wi
  - Is there a fault present?
- Safety Mechanisms in Functional Safety
  - Change engineers towards more rigorous work

How to find?

Knowledge about interfaces becomes very important.

How to identify?

How to derive correct information and models needed for manual and computer supported troubleshooting?

How to reach rigorous development without unrealistic burden on engineers?

Structured and machine-readable knowledge about the system is crucial !

We have to deal with:

- Huge size of the whole vehicle system
- Complexity
- Product line – not only one configuration
- Continuous integration and agility
- Development speed

For example:
Even with the best possible troubleshooting system, if the electrical architecture and components of the vehicle are not known to the system, troubleshooting will not be possible.

A very challenging information-management problem !

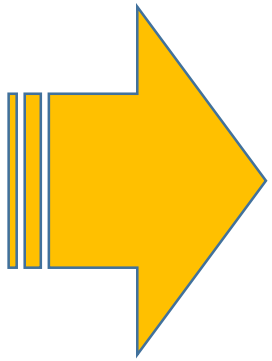
Digitalization: From Documents to Integrated Data

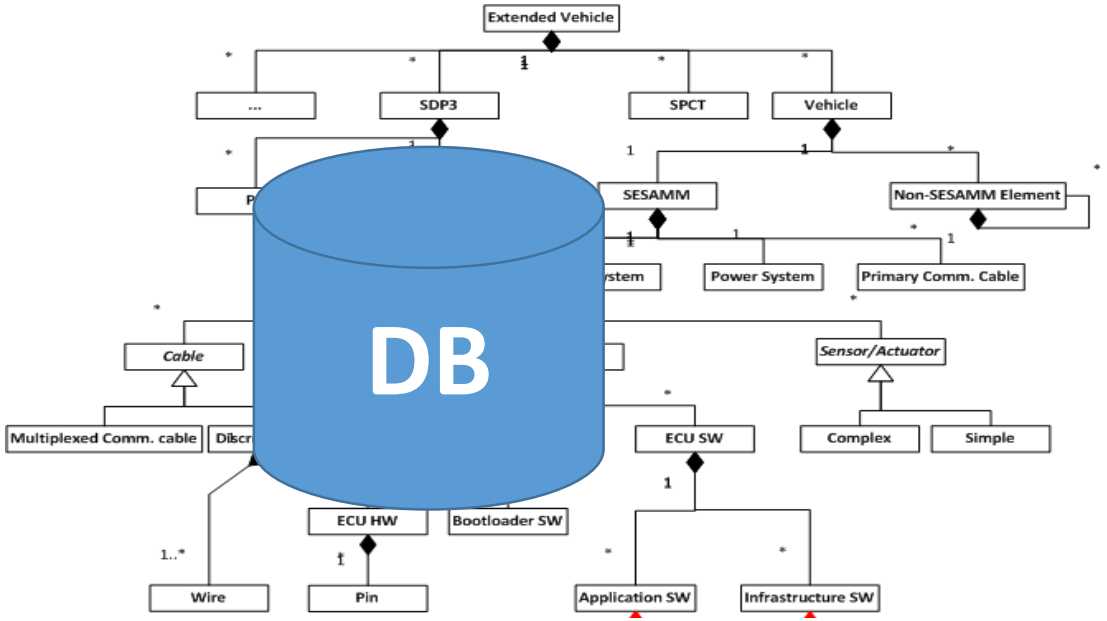
- Requirements
- Architectures
- Specifications
  - UF, AE, Appl SW
- *Hazard analyses, FMEA*
- *MSCs*
- *TMS*
- *Links to*
  - *diagnosis / workshop info*
  - *Issue tracking system*

.doc

.pdf

.xls

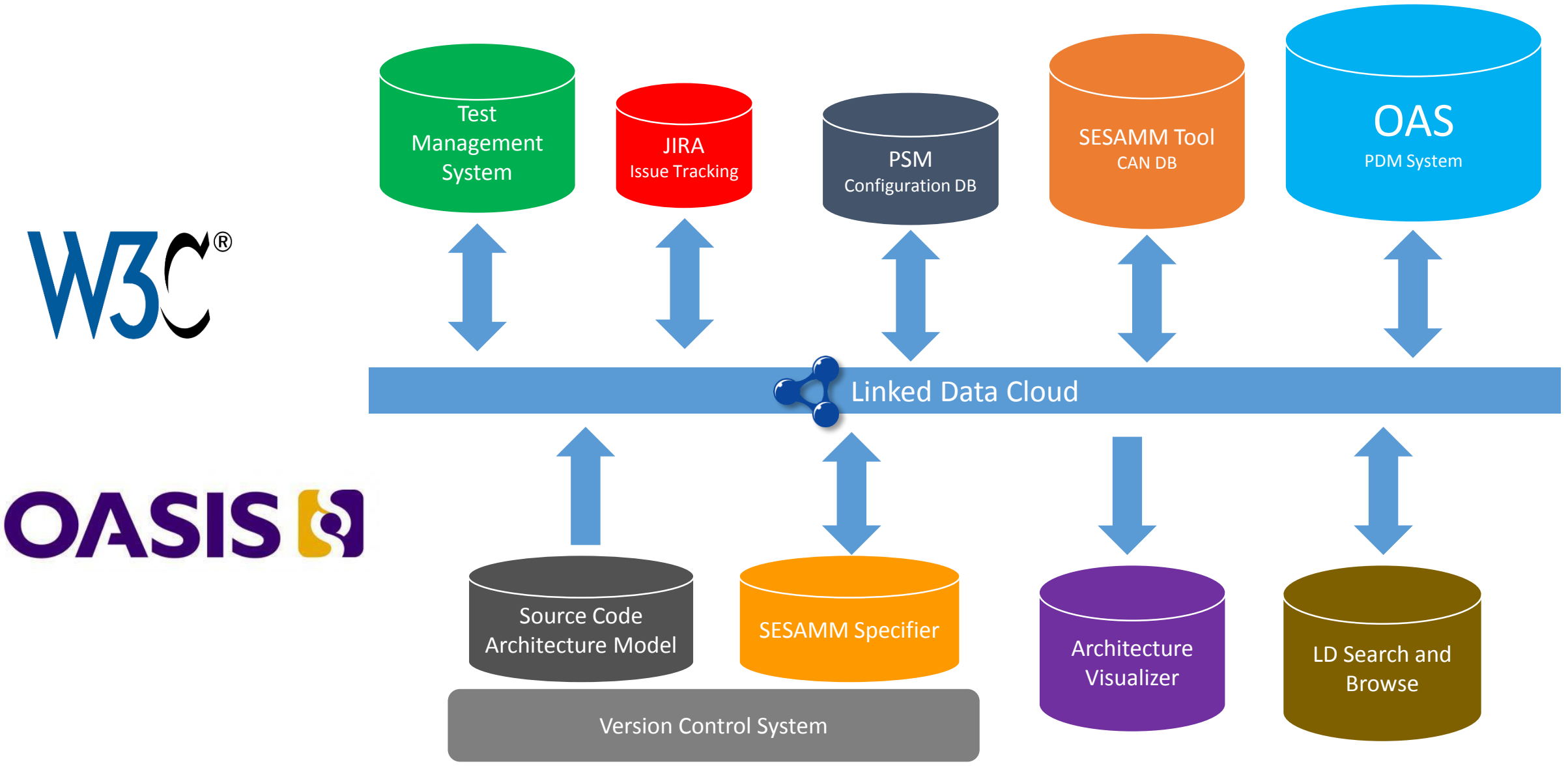




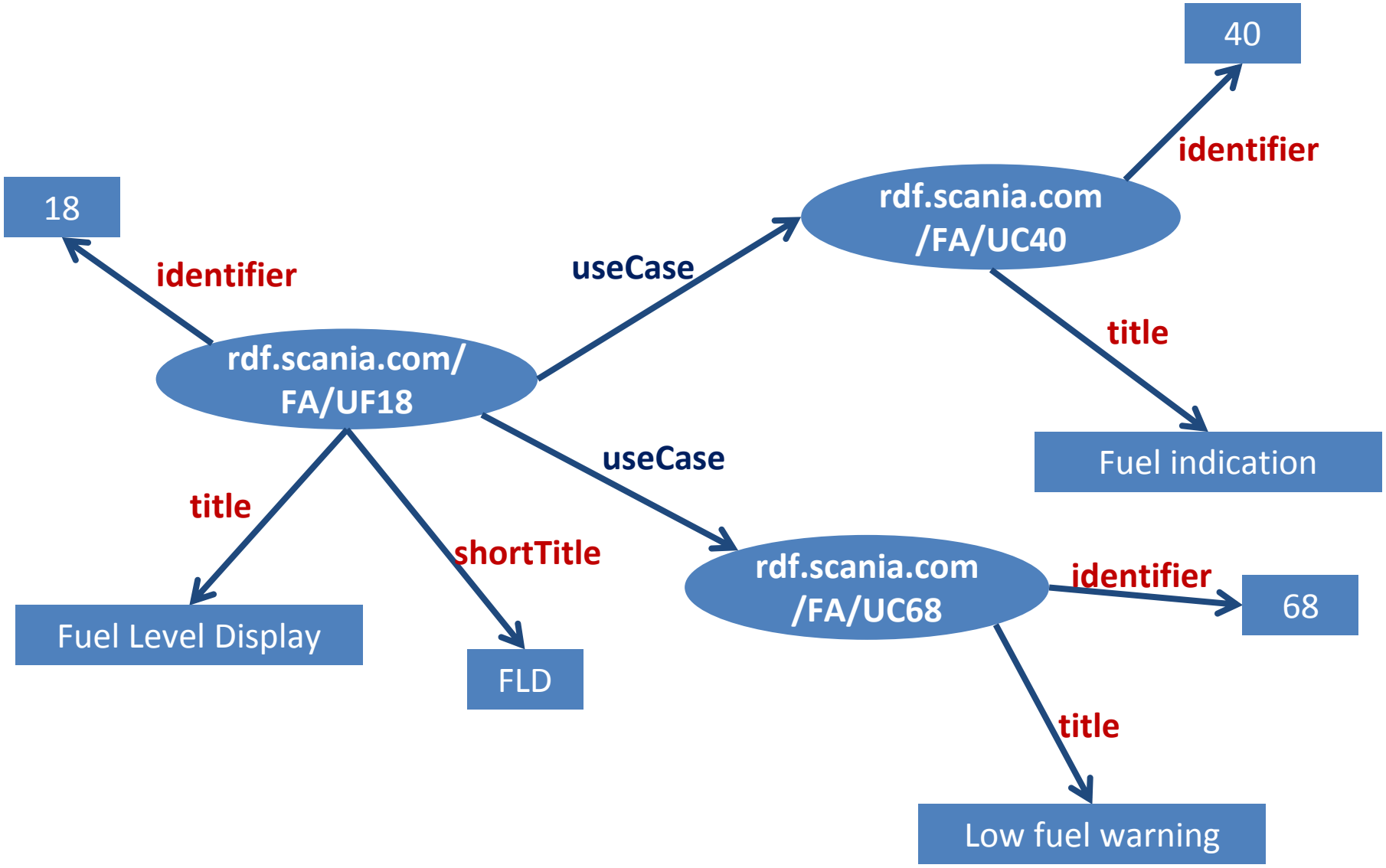
Digitalization Challenge

- It is not only digitization of information
  – method and processes need to change also
- Heavy management decisions
- Lack of competence; how to?
- Lack of technology
  - state-of-practice technologies do not support large scale digitalization

Tool Chain Architecture based on Linked Data (Semantic Web)



Linked Data provides a knowledge graph



Automotive Fault-Diagnosis - Summary

- Its **not only** ”on-board diagnostics”; **four** areas
  - OBD
  - FTC
  - Troubleshooting
  - Functional Safety
- Engineers and researchers from the different areas should meet

- Some really tricky problems remain: e.g.
  - How to diagnose a system without any detected faults?

- The main general problem is lack of structured and machine-readable information.
  Digitalization is needed.
  This is a focus of my current research: **Rigorous Systems Engineering**

END