



BMW Car IT GmbH, February 2014

RESEARCH ON AN OPEN-SOURCE SOFTWARE PLATFORM FOR AUTONOMOUS DRIVING SYSTEMS.

LUKAS BULWAHN, TILMANN OCHS, DANIEL WAGNER

**BMW
GROUP**
BMW Car IT GmbH



BMW CAR IT GMBH

Founded in 2001 as BMW affiliate

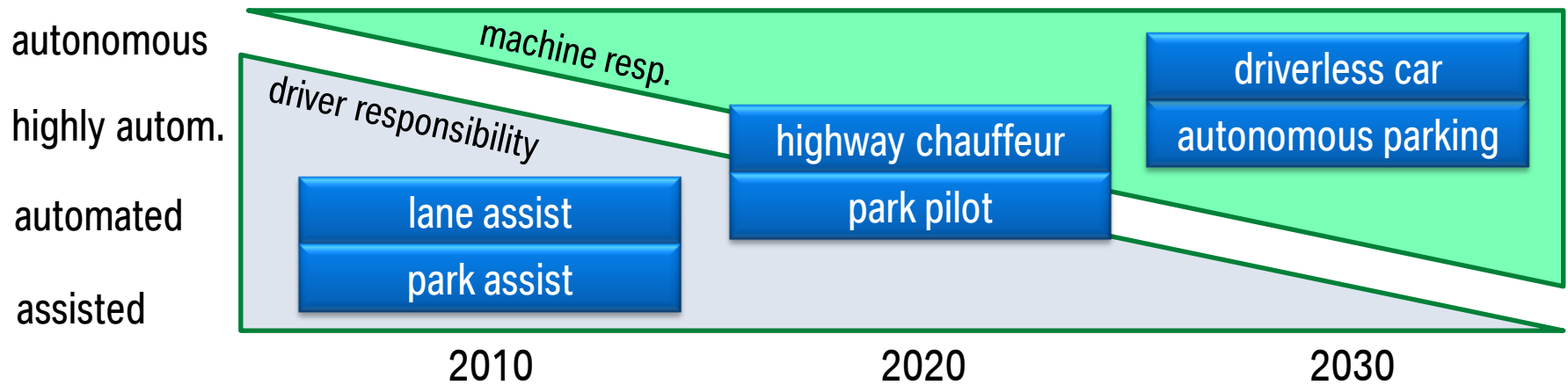
Strengthen BMW's software competence

- View vehicles as software systems
- Develop innovative software for future BMW Group vehicles
- Prototype solutions for early and reliable project decisions

Participate in several open-source communities and research projects



AUTONOMOUS DRIVING



Long-term trend: Pass tedious driving tasks to machine

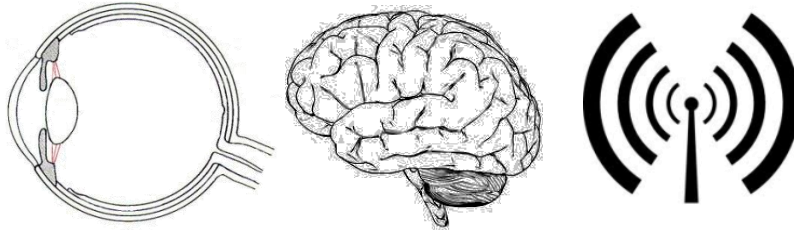
Active field of research: High rate of innovation for the foreseeable future

Competitive: All car manufacturers and others involved

Technology is now available: Sensors, Computers, AI-Algorithms

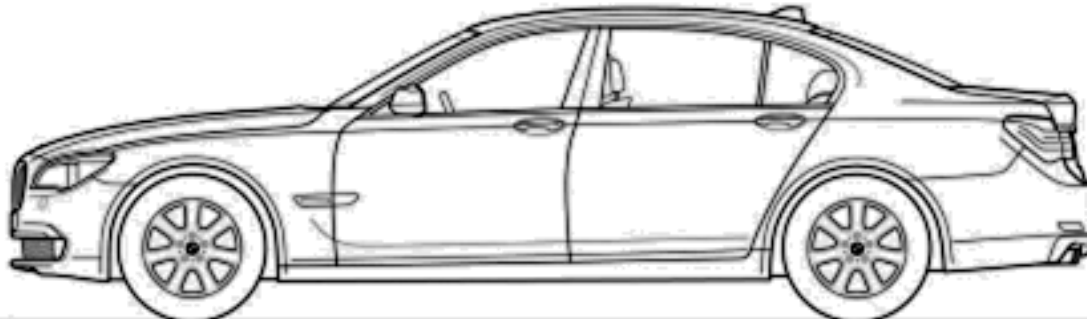
Main Challenge: Guaranteed reliability

DEPENDABLE POWER COMPUTING



Cognitive
System

advanced driver assistance, automated and autonomous driving, ...



Control
System

manual driving, driver assistance, active safety, ...

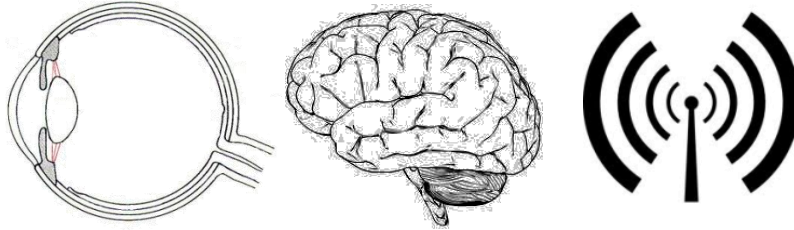
Control Software

- state machine + controller
- reliable microcontrollers
- deterministic software

Cognitive Software

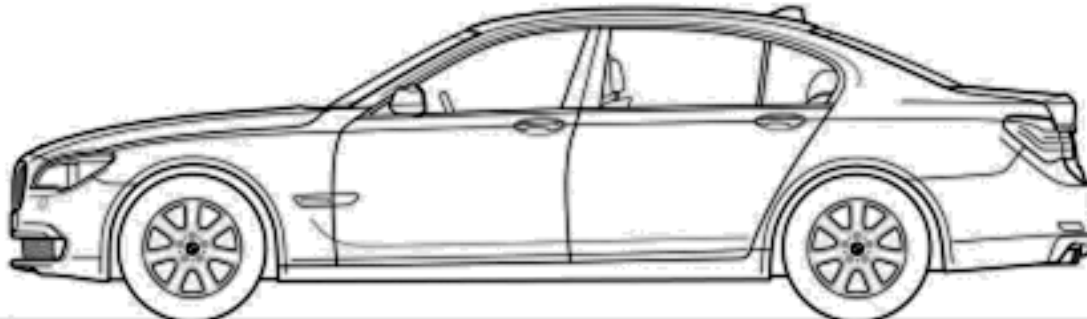
- dynamic models + AI
- peak performance SoCs
- dynamic software structure

DEPENDABLE POWER COMPUTING



Cognitive
System

advanced driver assistance, automated and autonomous driving, ...



Control
System

manual driving, driver assistance, active safety, ...

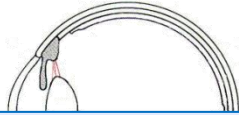
Control Software

- state machine + controller
- reliable microcontrollers
- deterministic software

Dependable Power Computing

- dynamic models + AI
- peak performance SoCs
- dynamic software structure

DEPENDABLE POWER COMPUTING



Cognitive

Claim:

differentiation through up-to-date **information** and **functional software**
vehicle E/E architecture and software platform is non-differentiating!



Control
System

manual driving, driver assistance, active safety, ...

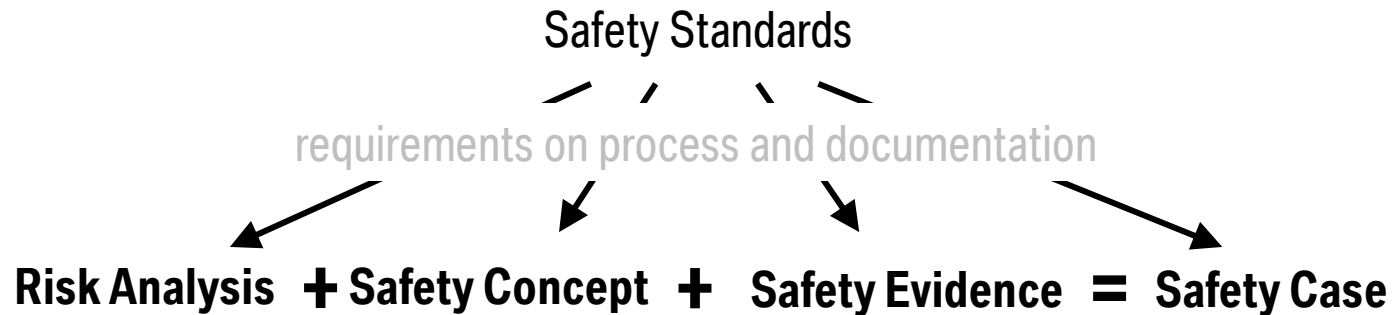
Control Software

- state machine + controller
- reliable microcontrollers
- deterministic software

Dependable Power Computing

- dynamic models + AI
- peak performance SoCs
- dynamic software structure

FUNCTIONAL SAFETY



What could go wrong, how bad?

How do we reduce the risk?

Evidence that concept is implemented!

Complete argument that system is safe.

Systematic faults => Careful design, Analysis, QM

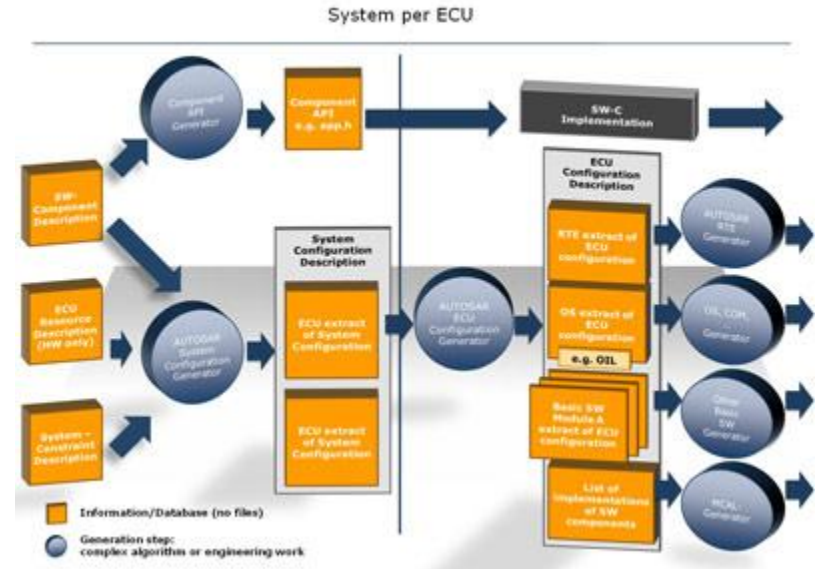
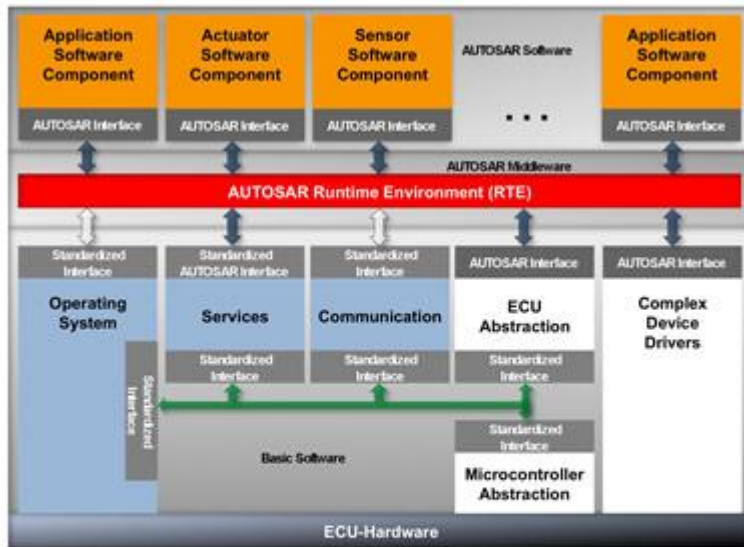
Random fault => Diagnosis, Redundancy and Fallback

„Human fault“ => Rigid process, „Safety culture“

Cognitive Software:

Systematic but non-deterministic fault => currently uncontrollable

CURRENT SOFTWARE PLATFORM



AUTOSAR Methodology

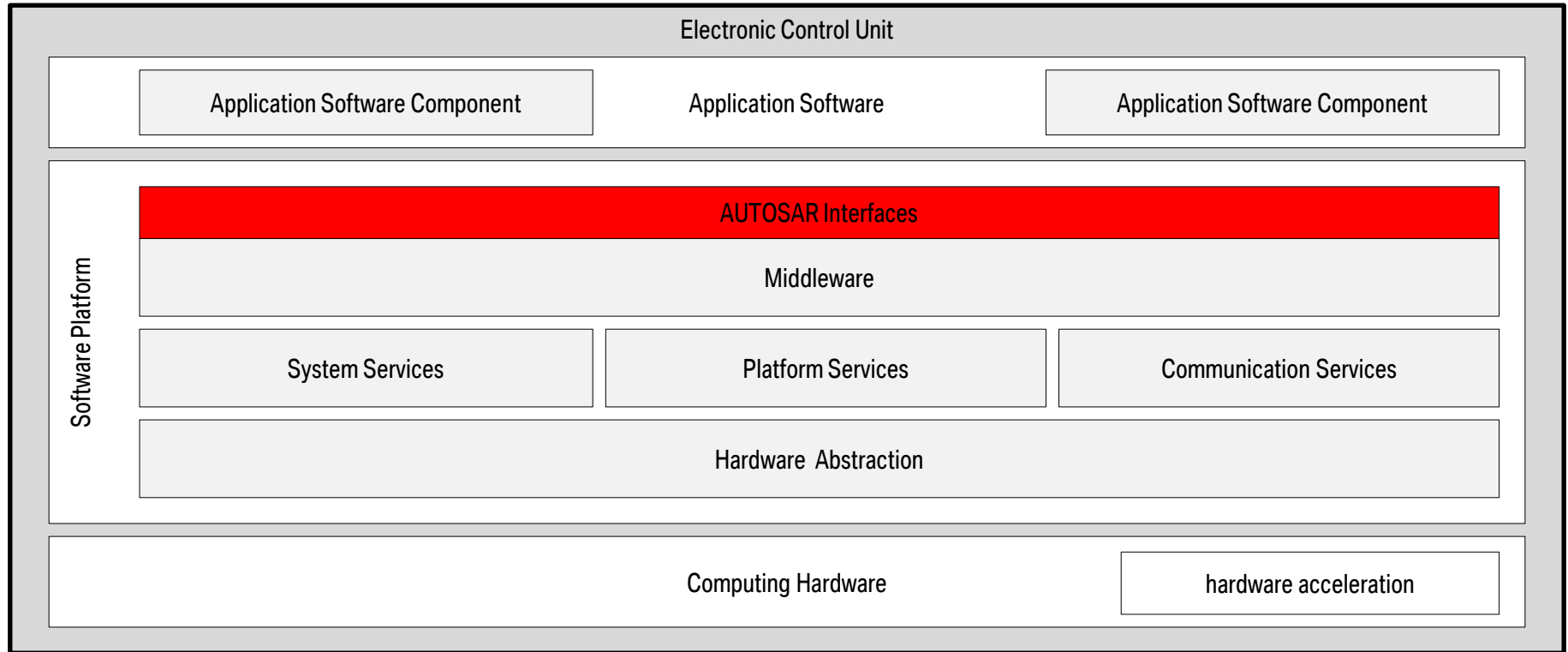
monolithic binary images
 „small“ microcontrollers

=> statically configured software stack
 => highly optimized using code generation

Cognitive Software:

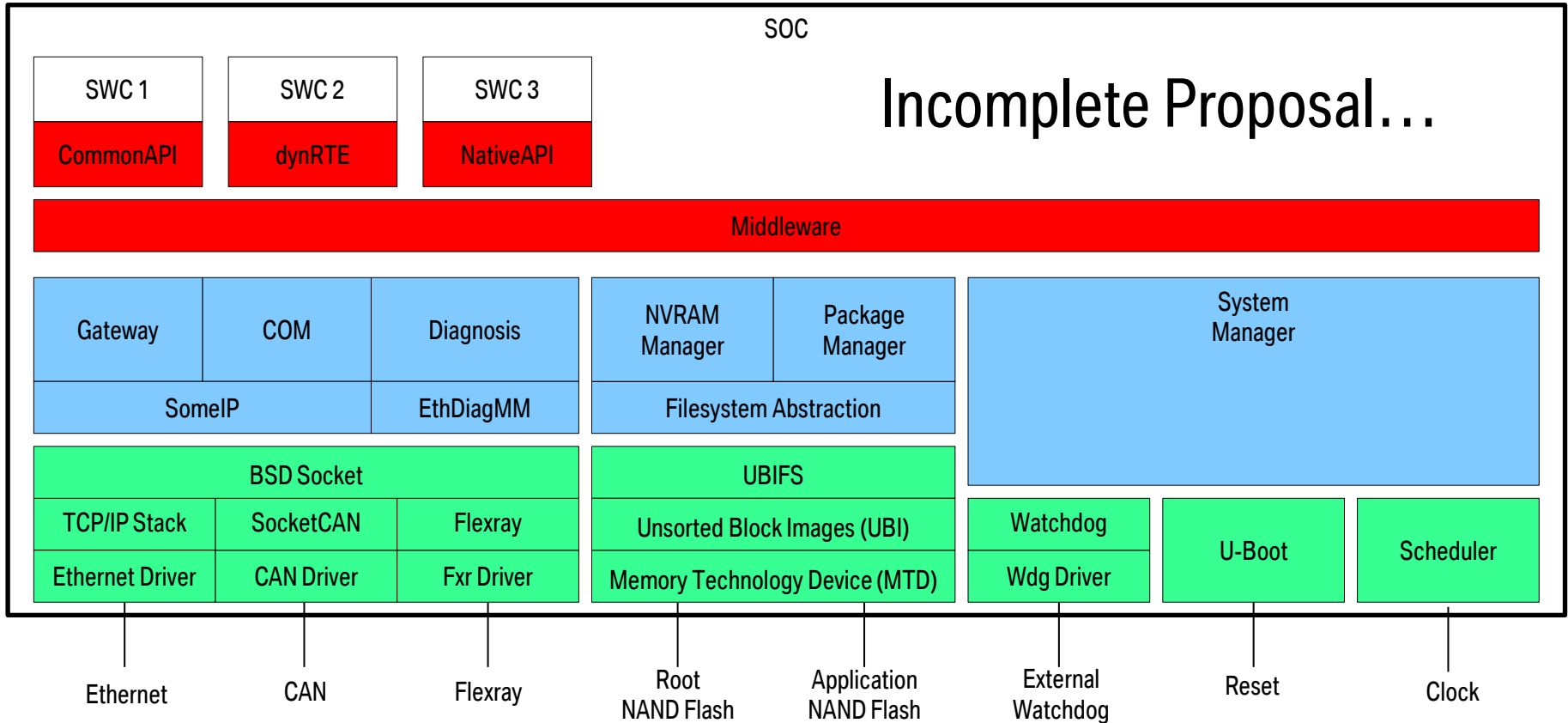
complex processing hardware => currently not supported
 large and complex sw-components => laborious to integrate and debug

DPC SOFTWARE PLATFORM



- System Services => Energy Mgmt, State Mgmt, Diagnosis, Update, ...
- Platform Services => Mass storage, Timebase, Monitoring, Isolation, ...
- Communication => Ethernet, CAN, Flexray, Network Management, ...
- Middleware => Standardized API for Portability and Reuse

LINUX AS DPC-OS



Linux...

- fulfills many of the requirements
- supports many architectures and is portable
- has large ecosystem and avoids vendor lock-in
- security is continuously monitored and improved

OSADL Foundation:

- Open Source Automation Development Lab
- Foundation (Genossenschaft)
- Funds projects of common interest and provides legal consulting
- Mission: Enable use of open-source software in automation industry

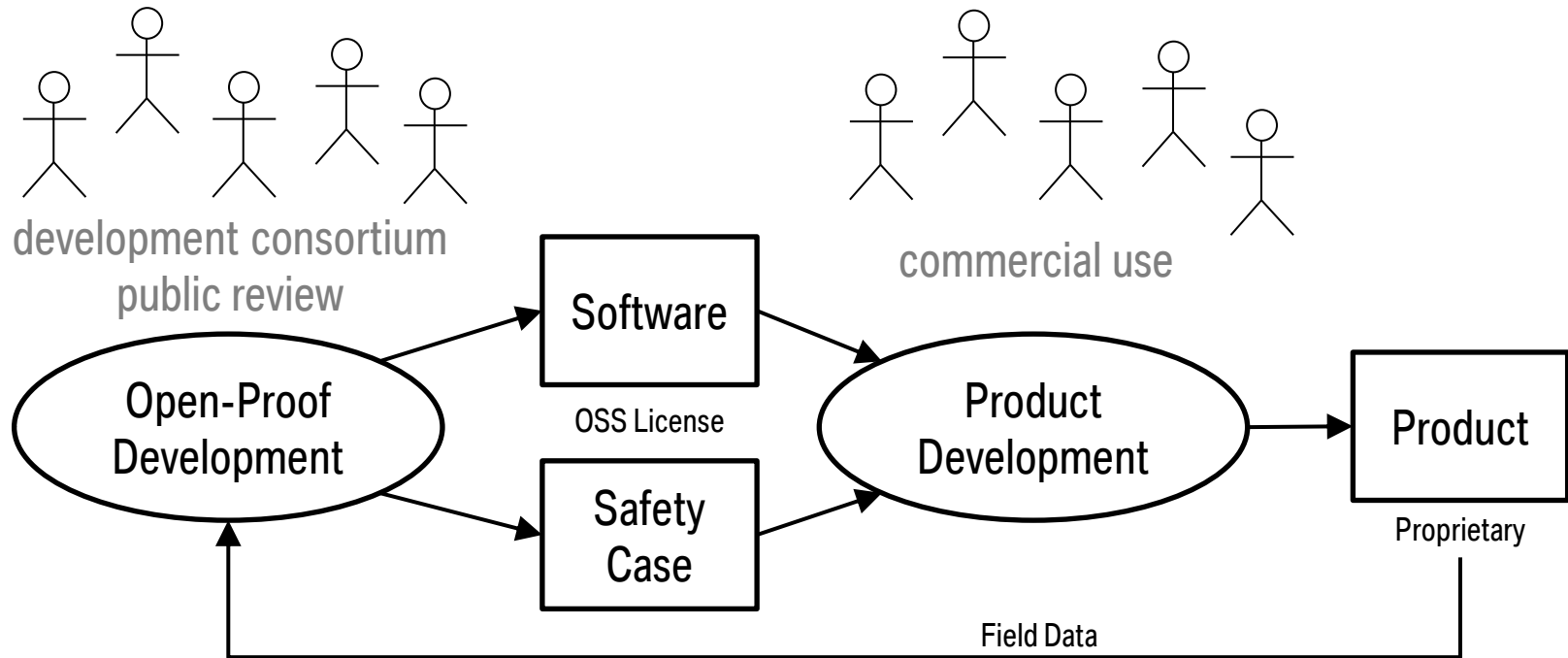
Realtime Linux :

- Support and funding of real-time kernel development (PREEMPT_RT)
- Develop and operate real-time testing lab
- Provide continuous feedback to real-time community

Safety Critical Linux:

- Qualify Linux for use in safety-relevant systems (up to ASIL B)
- Develop qualification packages for partner-provided use cases
- Enabling partners to qualify future GNU/Linux releases
- Results are open source, except use-case details

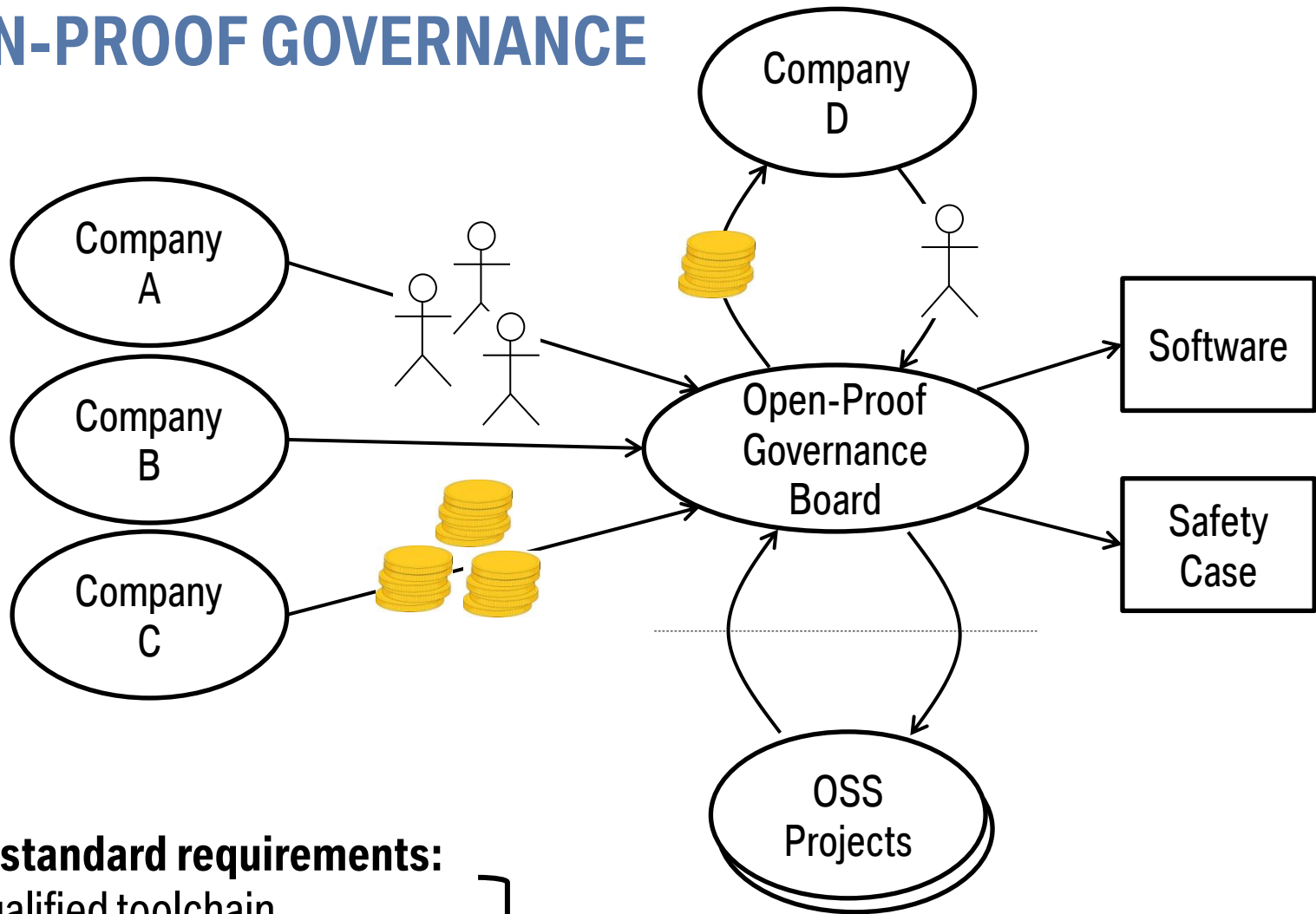
OPEN-PROOF DEVELOPMENT



- High development cost \Rightarrow Cost sharing through collaboration
- Beyond state of the art \Rightarrow Global agreement on safety case.
- Problem resolution \Rightarrow Publish safety-relevant field data.
- Complex platform software \Rightarrow Use existing building blocks, such as Linux.

Examples: eGAS, GNATpro, OpenETCS, ...

OPEN-PROOF GOVERNANCE



Safety standard requirements:

- Qualified toolchain
- Trained personnel
- Assigned roles
- Planned processes

Board with strict governance rules ensures **compliance** and **effectivity**.

CONCLUSION.

Open-proof development of an software platform for autonomous driving...

- is non-differentiating regarding future ADAS functionality
- provides solid base for application software
- is economically superior
- enables innovative approaches to safety

Proposal to initiate activities now:

- enable Linux as automotive operating system
- incorporate dynamic RTE in AUTOSAR standard
- initiate development of open-proof software platform
- harmonize vehicle and software architectures, where possible