Komponentenorientierte Automotive-Software-Entwicklung mit dem AUTOSAR-Standard

Dr. Heiko Dörr, Carmeq, Berlin

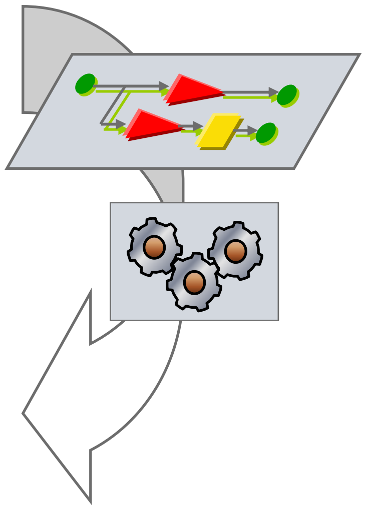
Kolloquium *Automotive Software Engineering*

TU Darmstadt, 12.05.2009

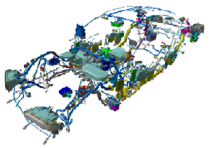
carmeq
software & systems

Content.

1. *Model-based Development*

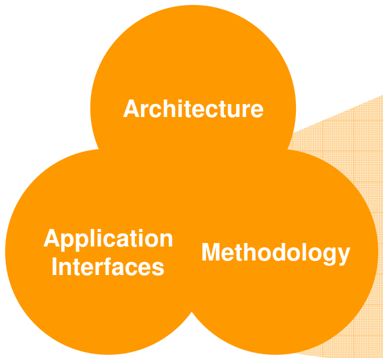


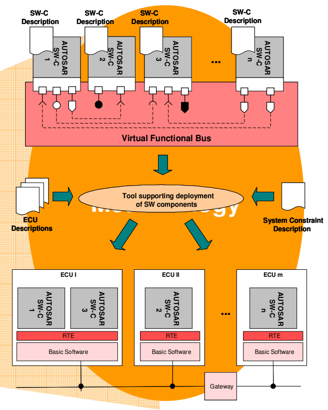
2. *Automotive System Engineering*



3. *AUTOSAR Standardization*

4. *Methodology of System Development*

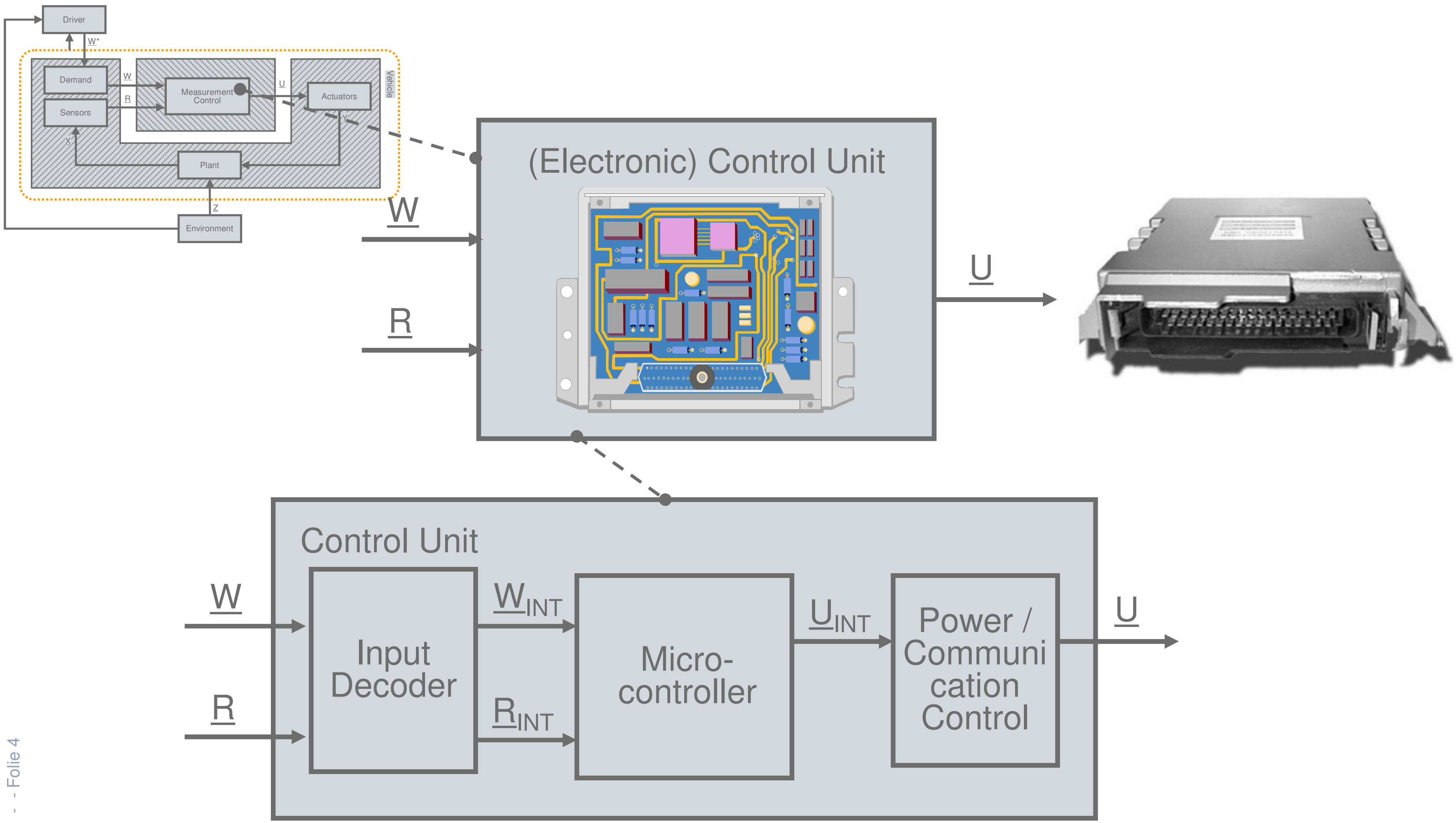




AUTOSAR

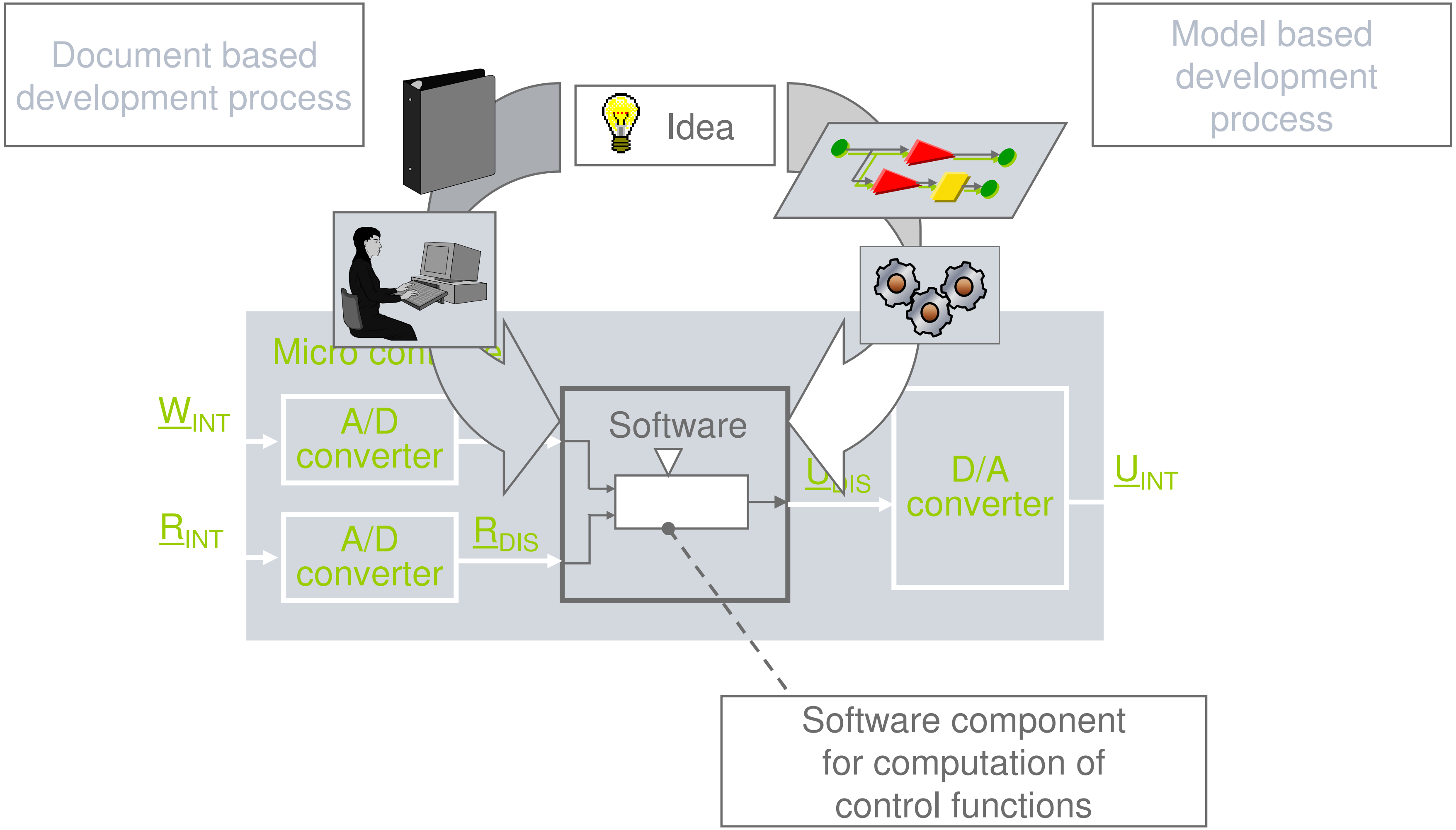
Model-based Development

Model-based Development.
General Task.

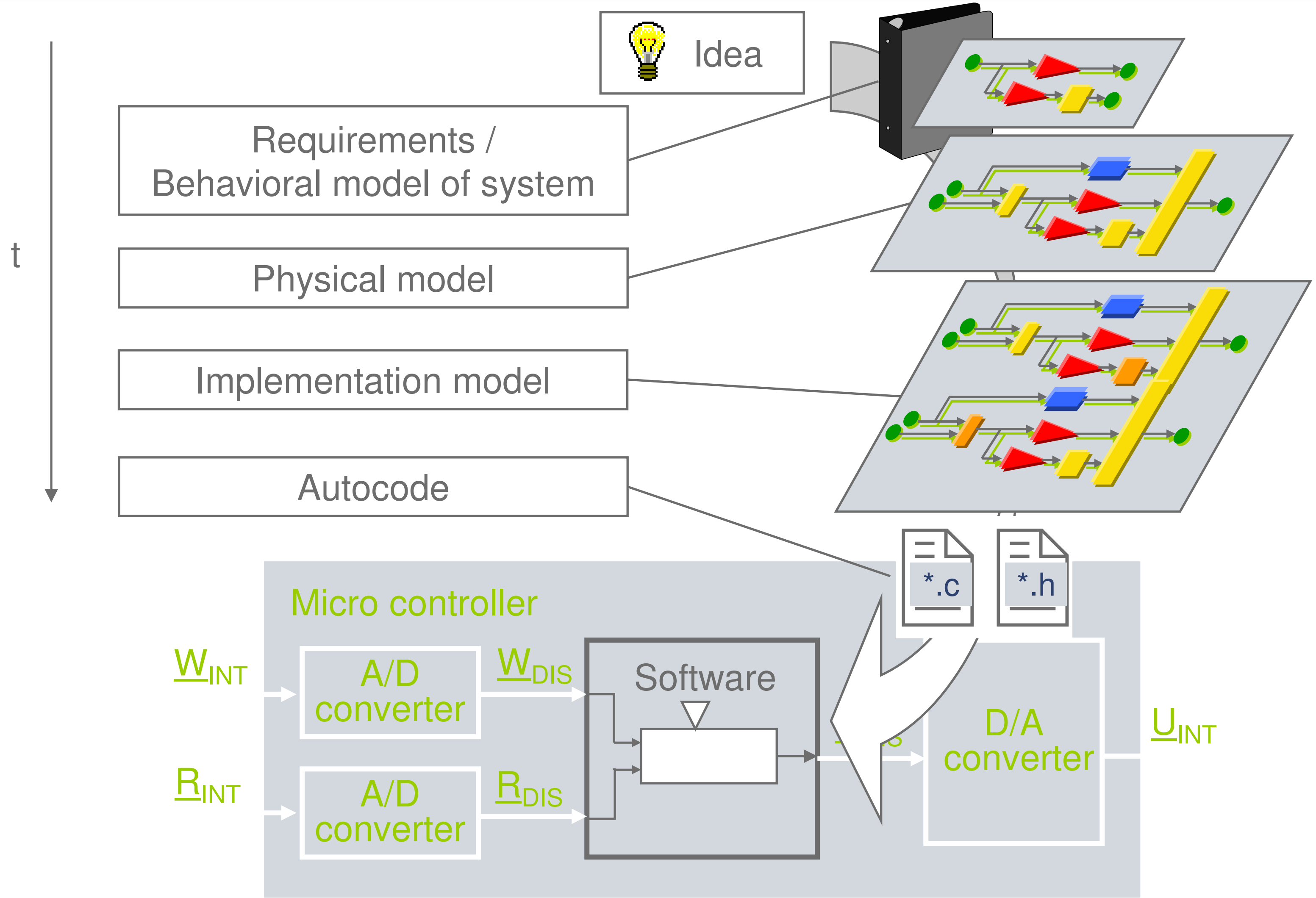


MbE-Slides by M. Conrad, Abb: Schäuffele/Zurawka: Automotive Software Engineering; Foto: www.bba-reman.com/ images/ecu.jpg

Model-based Development.
Paradigm Shift.



Model-based Development.
Evolution of Models.



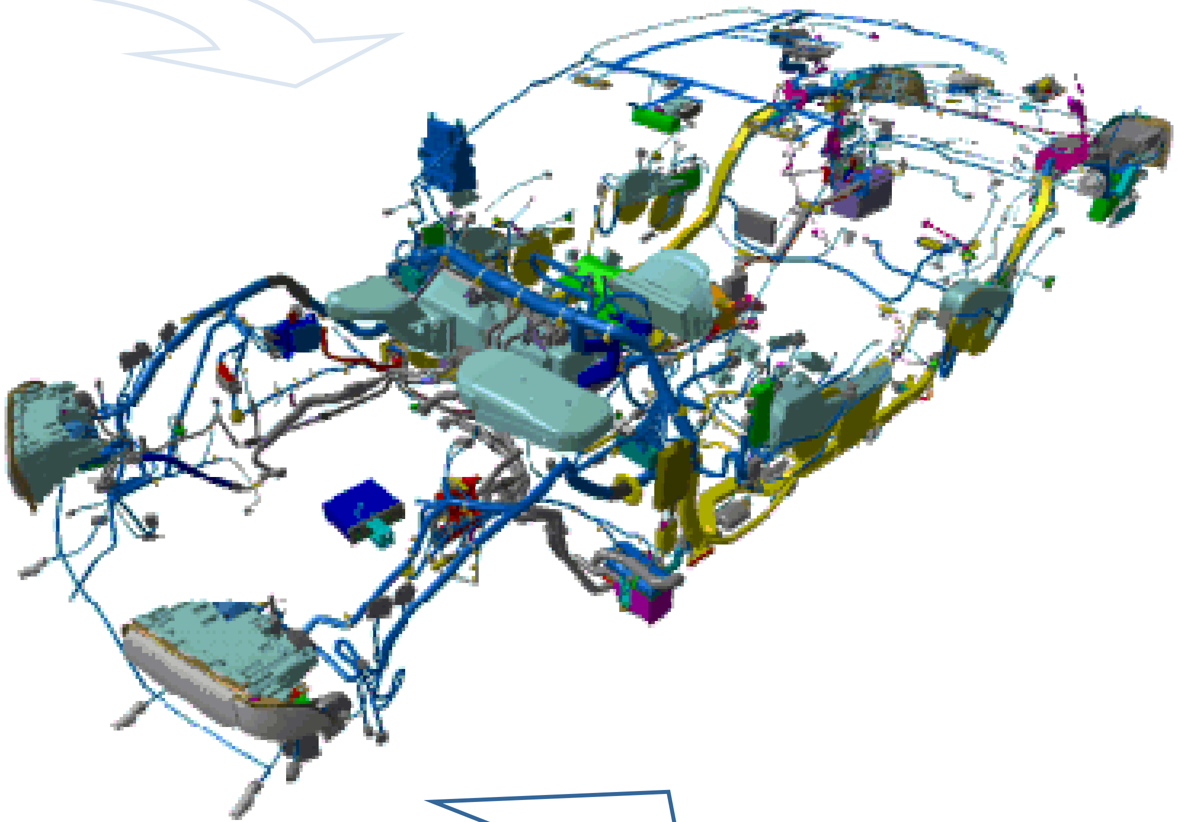
Automotive Systems and SW Engineering

Automotive Open System Architecture
Cooperate on standards – compete on implementation

Non functional legal requirements

Vehicle family management

Resource efficiency



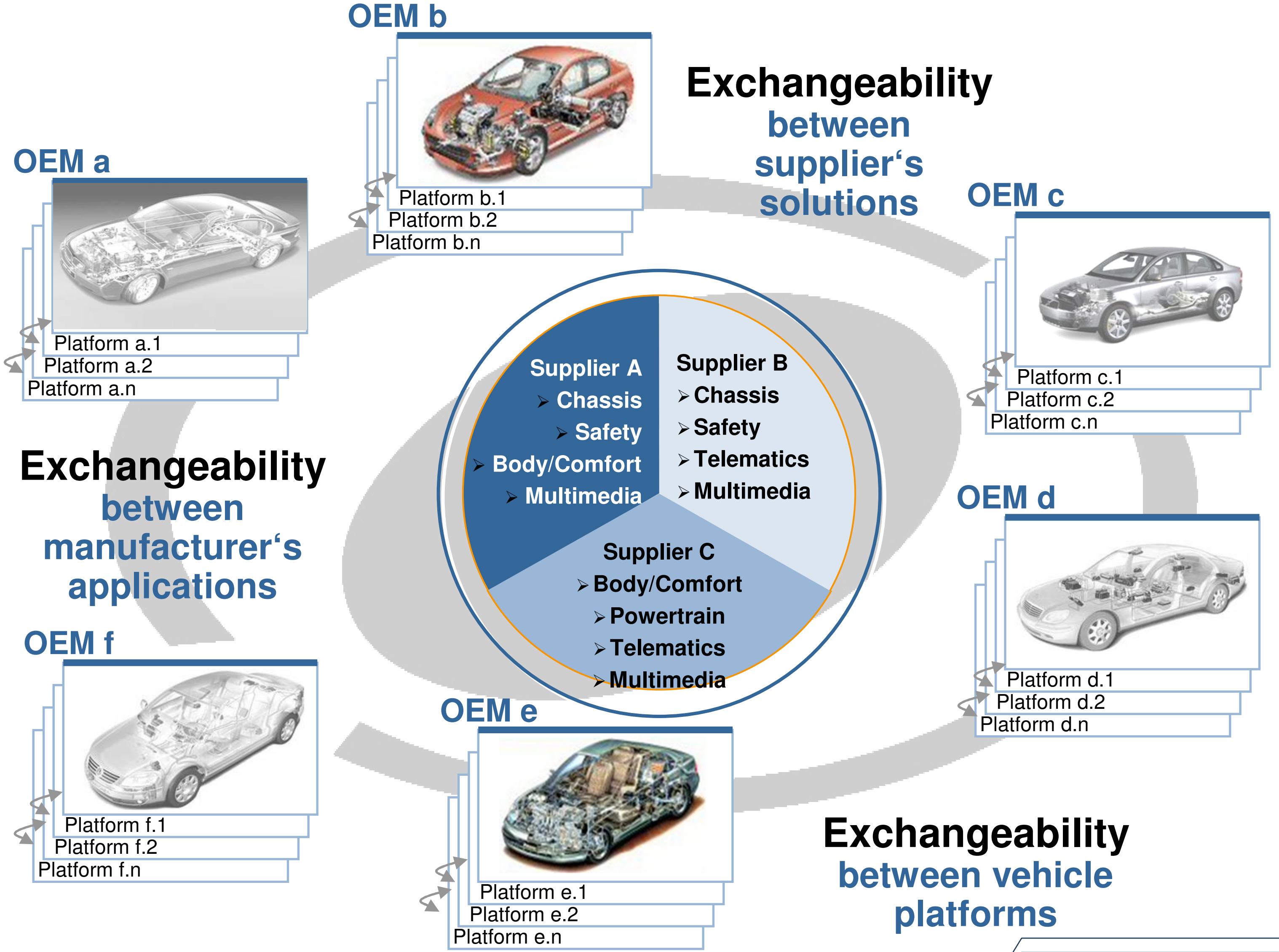
Driver assistance

Driving dynamics

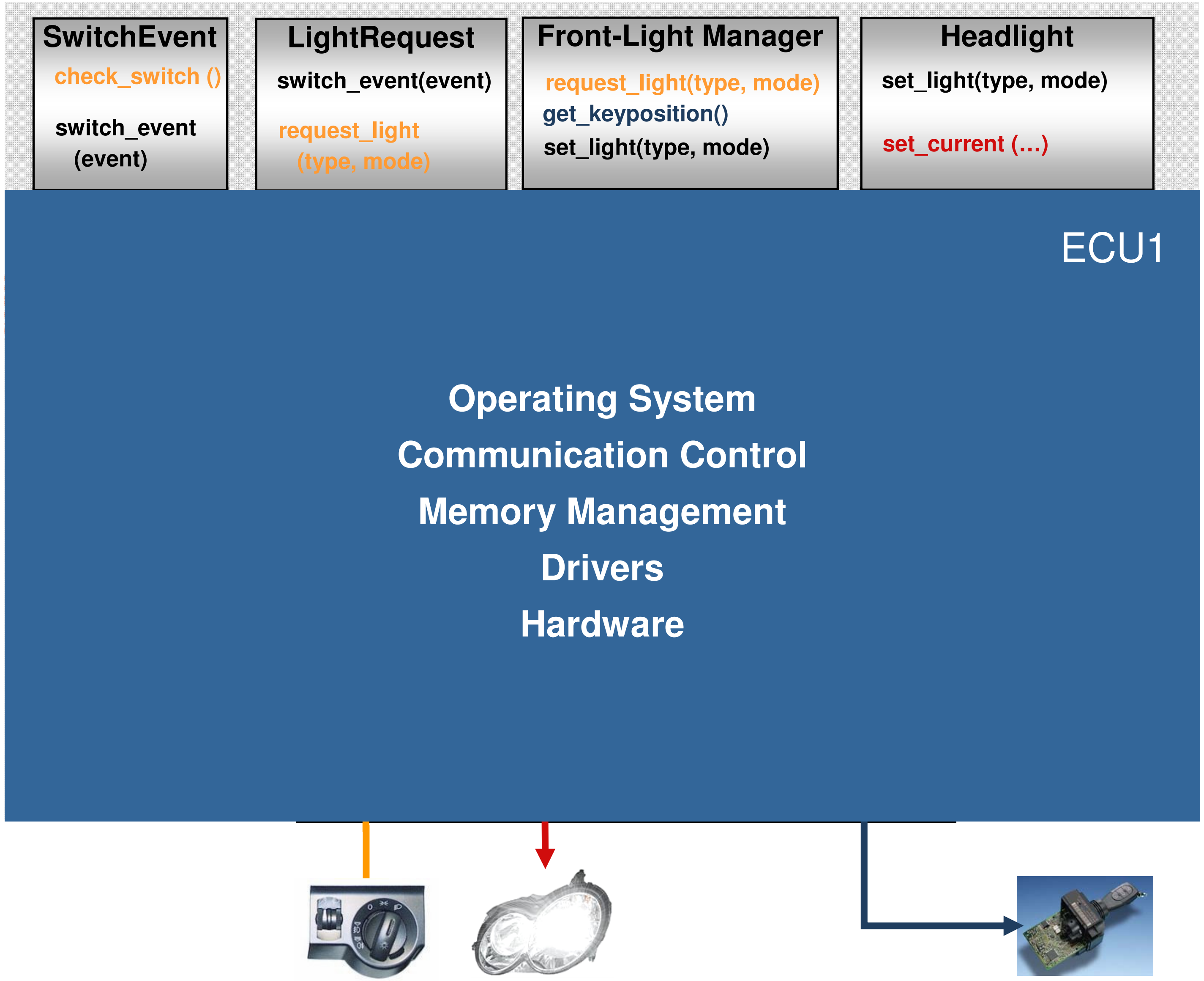
Safety functions (active/passive)

Comfort functions

AUTOSAR Managing Complexity by Exchangeability and Reuse of Software Components



Use case 'Front-Light Management'



Use case 'Front-Light Management'
Exchange of type of front-light

SwitchEvent	LightRequest	Front-Light Manager	Xenonlight
check_switch () switch_event (event)	switch_event(event) request_light (type, mode)	request_light(type, mode) get_keyposition() set_light(type, mode)	set_light(type, mode) set_current (…)

ECU2

Operating System

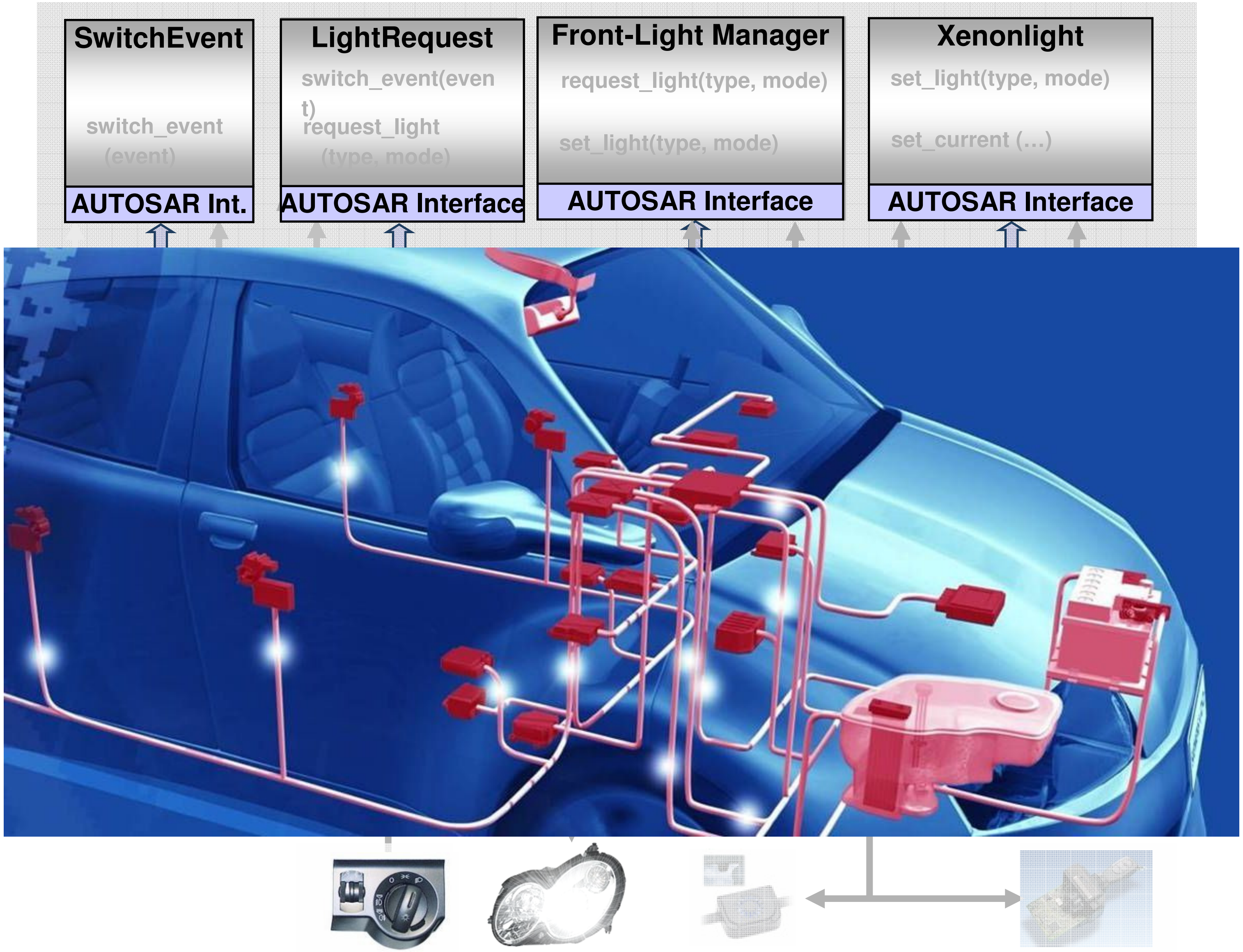
Communication Control

Memory Management

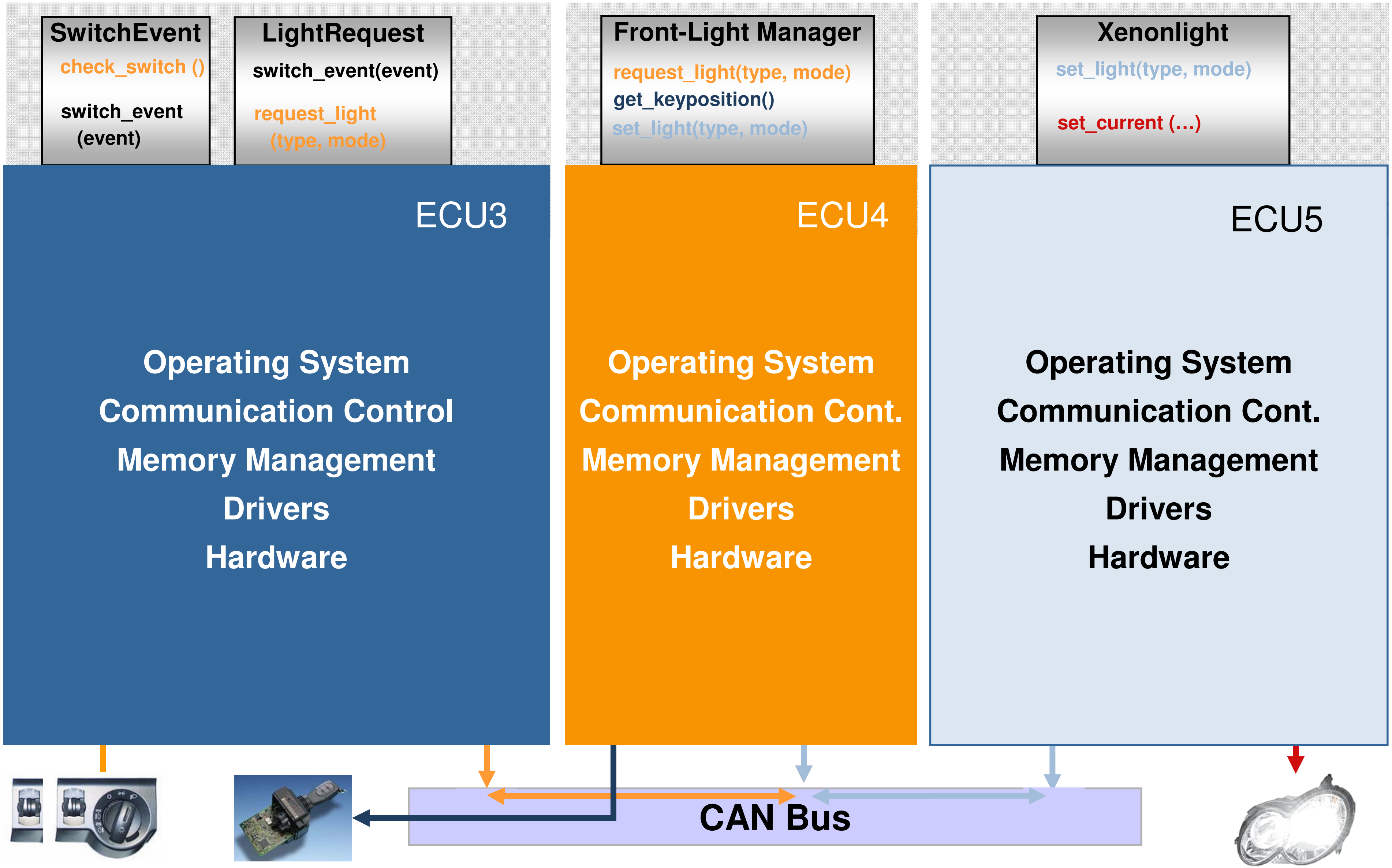
Drivers

Hardware

Distribution on ECUs



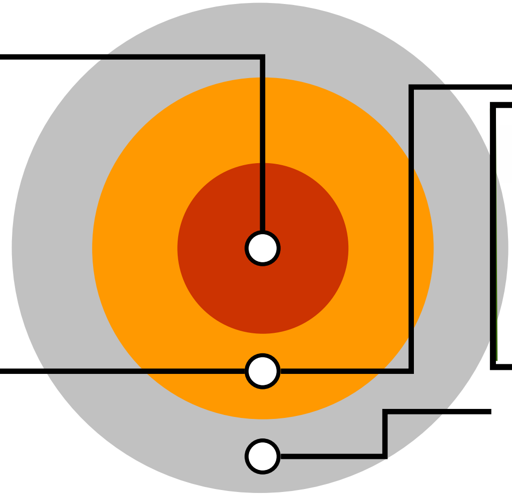
Use case 'Front-Light Management' – Multiple ECUs



AUTOSAR Standardization

AUTOSAR – Core Partners and Members

Status: 13th March 2008



9 Core Partner

BMW Group, Bosch, Continental, Daimler, Ford, Opel (A General Motors Company), PSA Peugeot Citroën, Toyota, Volkswagen AG

6 Development Member

Symta Vision, SGS, Kereval, Cosmic Software, TÜV Nord, C&S

67 Associate Member

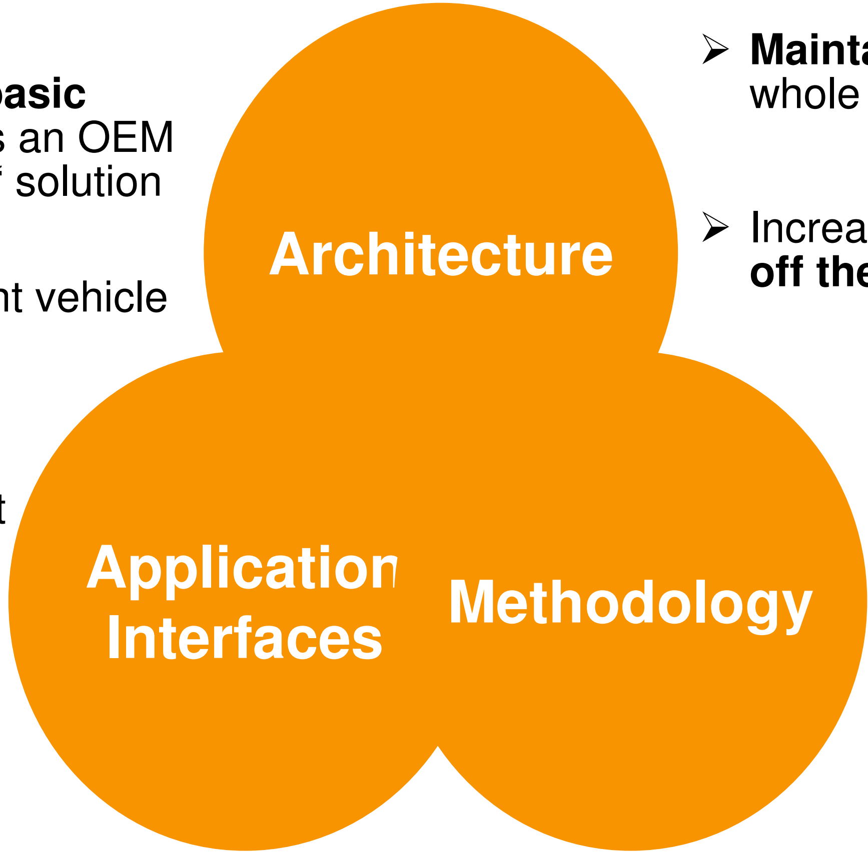
52 Premium Member

General OEM	Generic Tier 1	Standard Software	Tools and Services	Semi-conductors
Fiat Group Automobiles SpA, Honda, Nissan, Hyundai-Kia Motors, Mazda, Renault, Porsche, MAN, Volvo	Autoliv, Alpine, Delphi, Denso, Hella, Johnson Controls, Lear Corporation, Magna, TRW, Valeo, Magneti Marelli, Visteon, ZF, ZF Lenksysteme	Intecs, Hitachi, MBtech, softing, geensys, vector	Tata Elxsi Limited, KPIT Cummins Infosystems Limited, dSPACE, Infosys, Elektrobit, ETAS, Esterel Technologies, IAV GmbH, FEV, Mentor Graphics, TTTech, The MathWorks, Telelogic, TietoEnator, Wipro	ARM, freescale, patni, Infineon, Fujitsu, NEC Electronics Corporation, NXP, ST, Renesas

Up-to-date status see: http://www.autosar.org

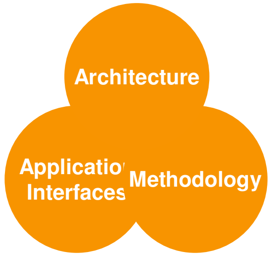
AUTOSAR
Project Objectives and Main Working Topics

- Implementation and **standardization of basic system functions** as an OEM wide “Standard Core“ solution
- **Scalability** to different vehicle and platform variants
- **Transferability of functions** throughout network
- **Integration** of functional modules from **multiple suppliers**

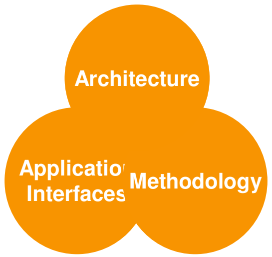


- **Maintainability** throughout the whole “Product Life Cycle“
- Increased use of **“Commercial off the shelf hardware“**
- **Software updates** and upgrades over vehicle lifetime
- Consideration of availability and **safety** requirements

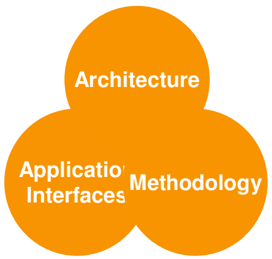
AUTOSAR
Main Working Topics



- Architecture:
  Software architecture including a complete basic or environmental software stack for ECUs – the so called AUTOSAR Basic Software – as an integration platform for hardware independent software applications.



- Methodology:
  Exchange formats or description templates to enable a seamless configuration process of the basic software stack and the integration of application software in ECUs and it includes even the methodology how to use this framework.

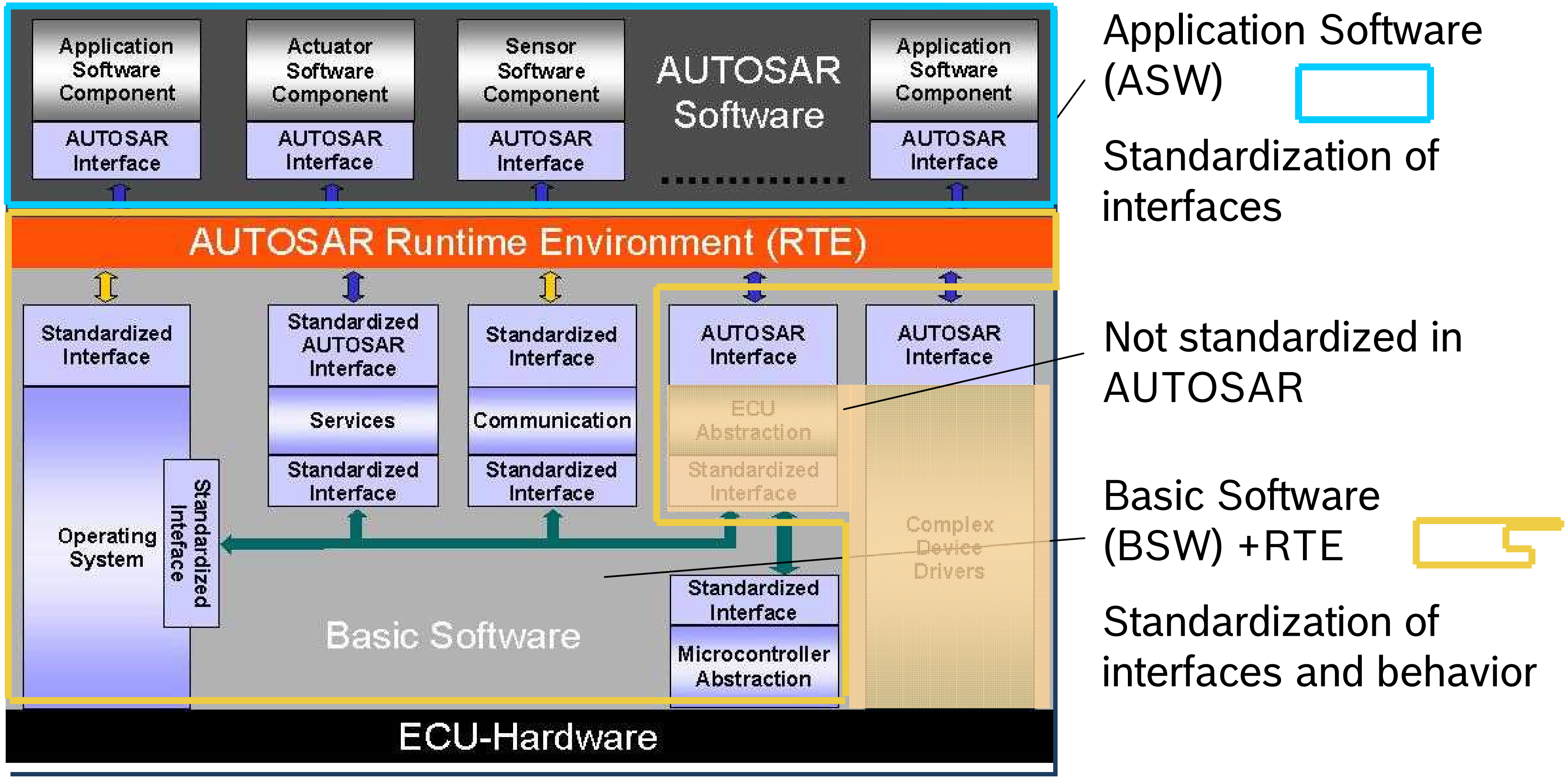


- Application Interfaces:
  Specification of interfaces of typical automotive applications from all domains in terms of syntax and semantics, which should serve as a standard for application software.

Main Concepts: Architecture

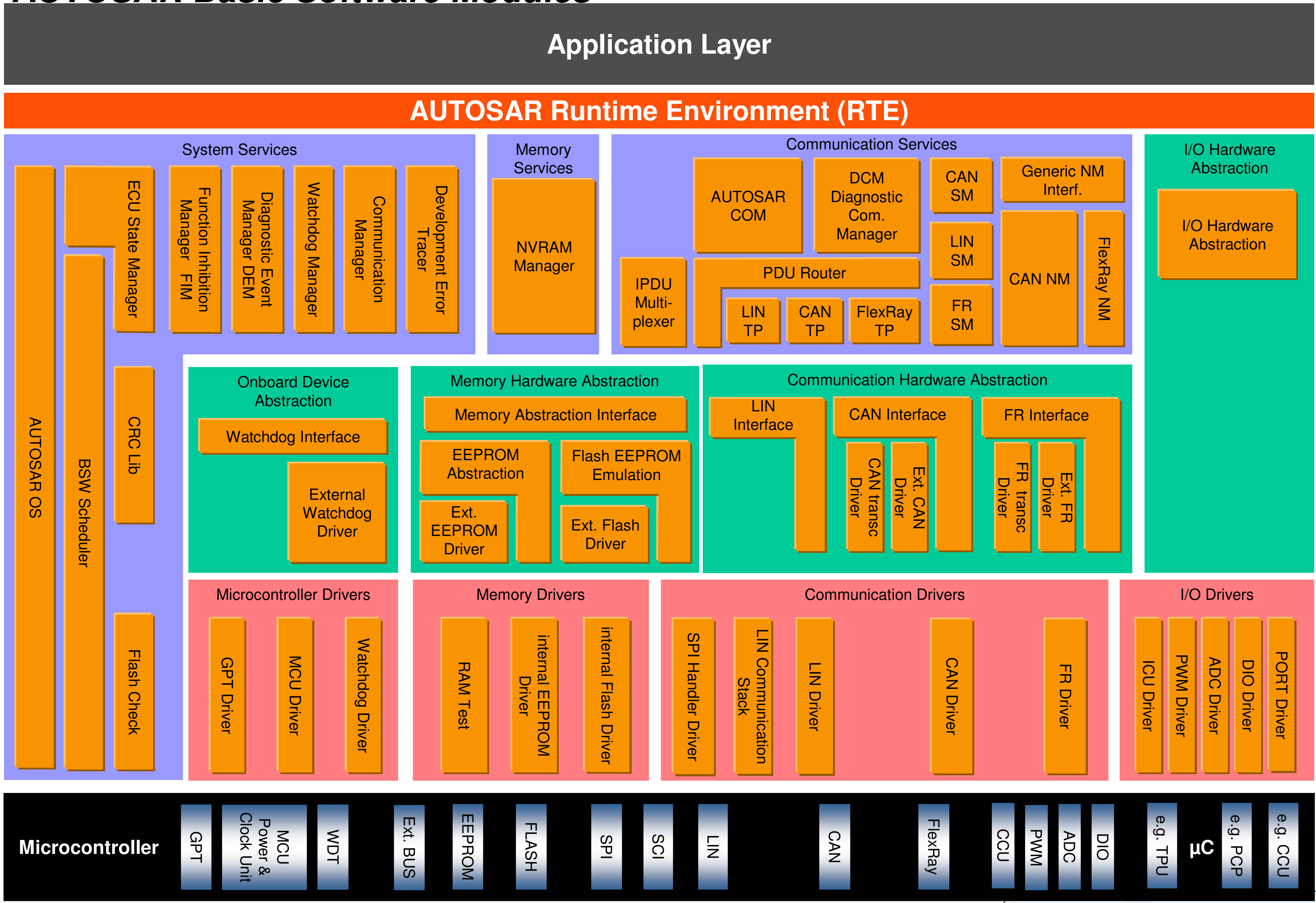
- Basic Software modules
- Run time environment and communication

AUTOSAR ECU Software Architecture



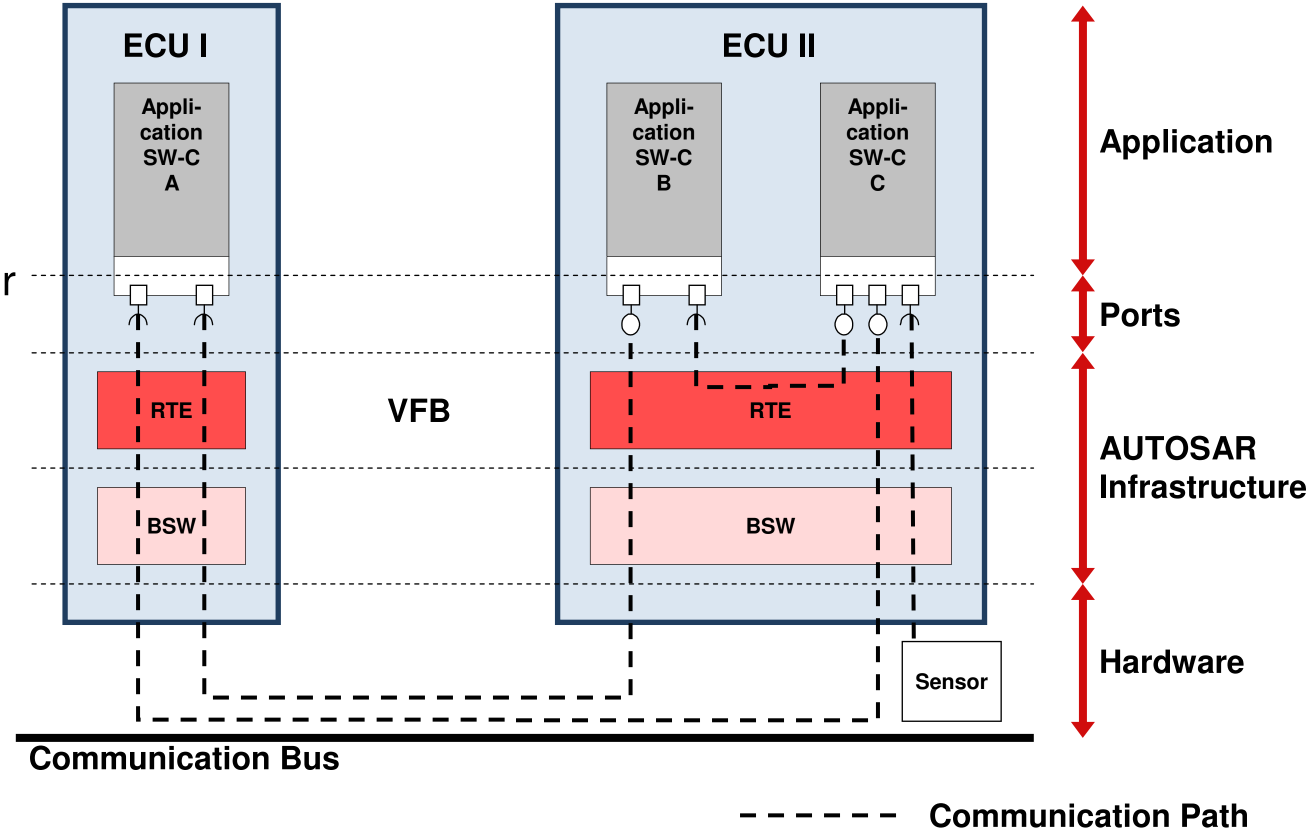
Objectives: Basic SW: Decoupling of Hardware and Application Software
Application SW: Relocation / Reuse of SW-Components between ECUs

AUTOSAR Basic Software Modules



Intra- and Inter-ECU Communication

- Ports implement the interface according to the communication paradigm (here client-server based).
- Ports are the interaction points of a component.
- The communication is channeled via the RTE.
- The communication layer in the basic software is encapsulated and not visible at the application layer.



AUTOSAR Architecture – Conclusion

1 AUTOSAR harmonizes already existing basic software solutions and closes gaps for a seamless basic software architecture.

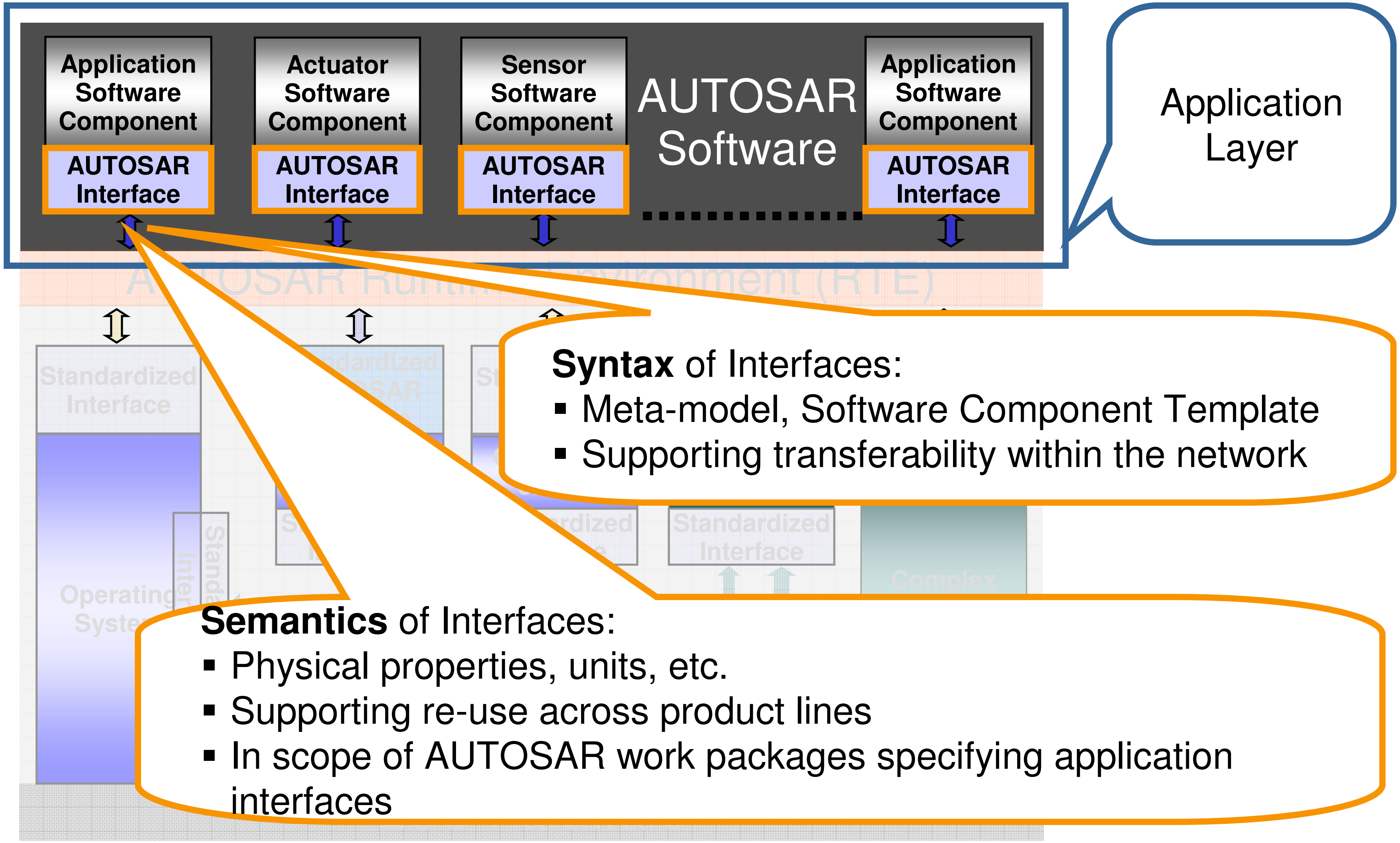
2 AUTOSAR aims at finding the best solution for each requirement and not finding the highest common multiple.

3 The decomposition of the AUTOSAR layered architecture into some 40 modules has proven to be functional and complete.

Main Concepts: Application Interfaces

- Standardization approach
- Current stage of standardization

AUTOSAR Application Interfaces



Syntax of Interfaces:

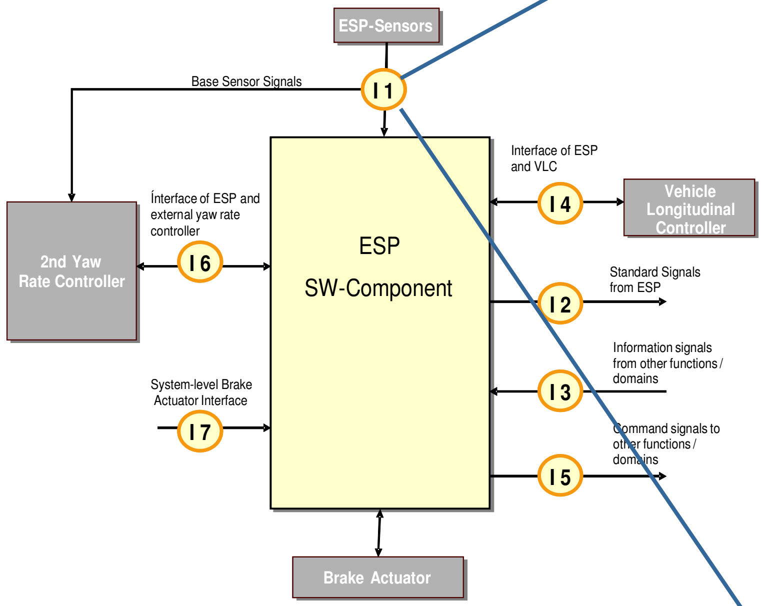
- Meta-model, Software Component Template
- Supporting transferability within the network

Semantics of Interfaces:

- Physical properties, units, etc.
- Supporting re-use across product lines
- In scope of AUTOSAR work packages specifying application interfaces

To ease the re-use of software components across several OEMs, AUTOSAR proceeds on the standardization of the application interfaces agreed among the partners.

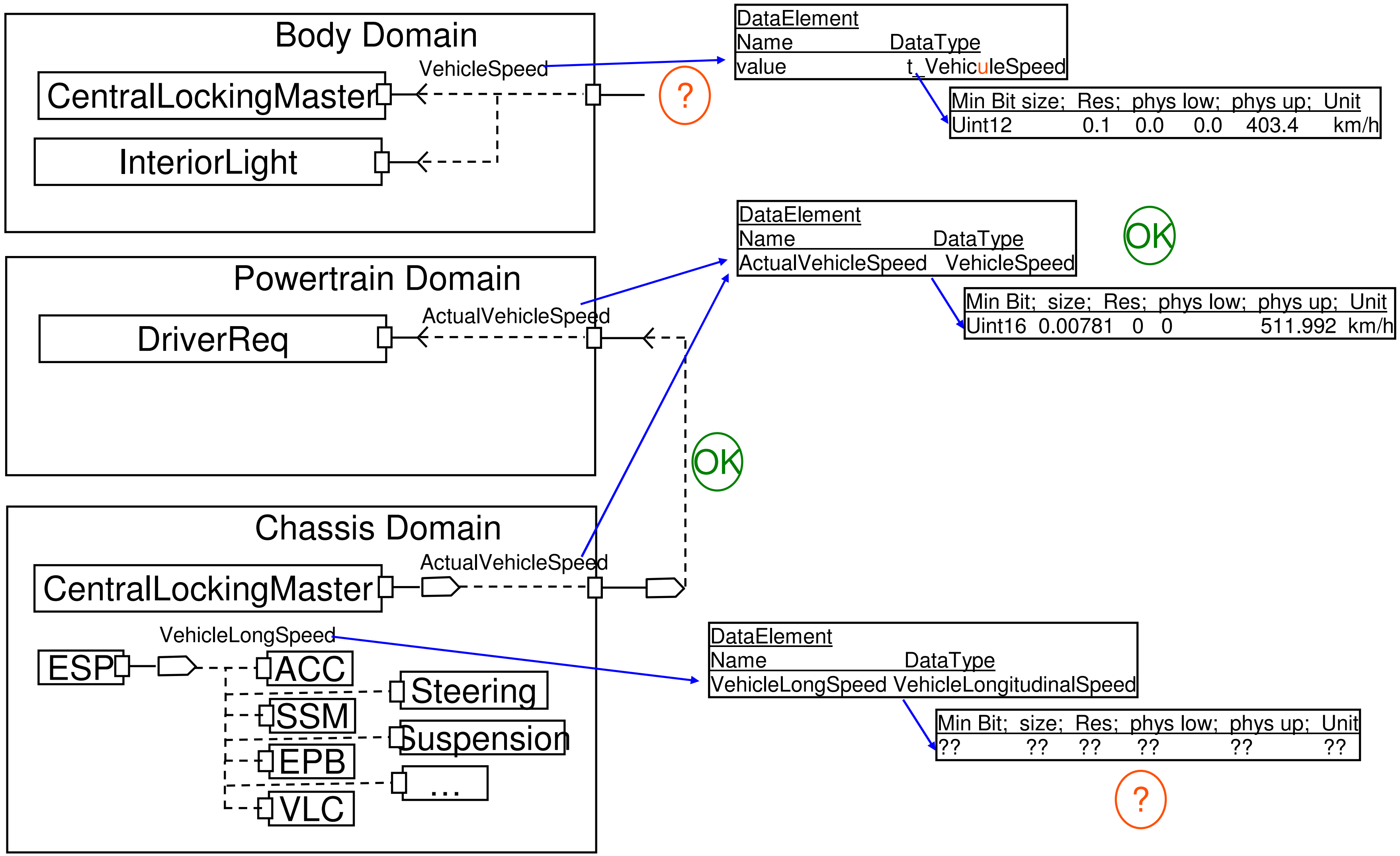
Example



Data Type Name	**LongAccBase**
…	
Data Type Name	**YawRateBase**
Description	Yaw rate measured along vehicle z- axis (i.e. compensated for orientation). Coordinate system according to ISO 8855
Data Type	S16
Integer Range	-32768..+32767
Physical Range	-2,8595..+2,8594
Physical Offset	0
Unit	rad/sec
…	….
Remarks	This data element can also be used to instantiate a redundant sensor interface. Range might have to be extended for future applications (passive safety).
…	
Data Type Name	**RollRateBase**

Standardized application interfaces on system level
(ESP-system, chassis domain)

Major task: Conflict Resolution – Example Vehicle Speed



AUTOSAR Application Interfaces – Conclusion

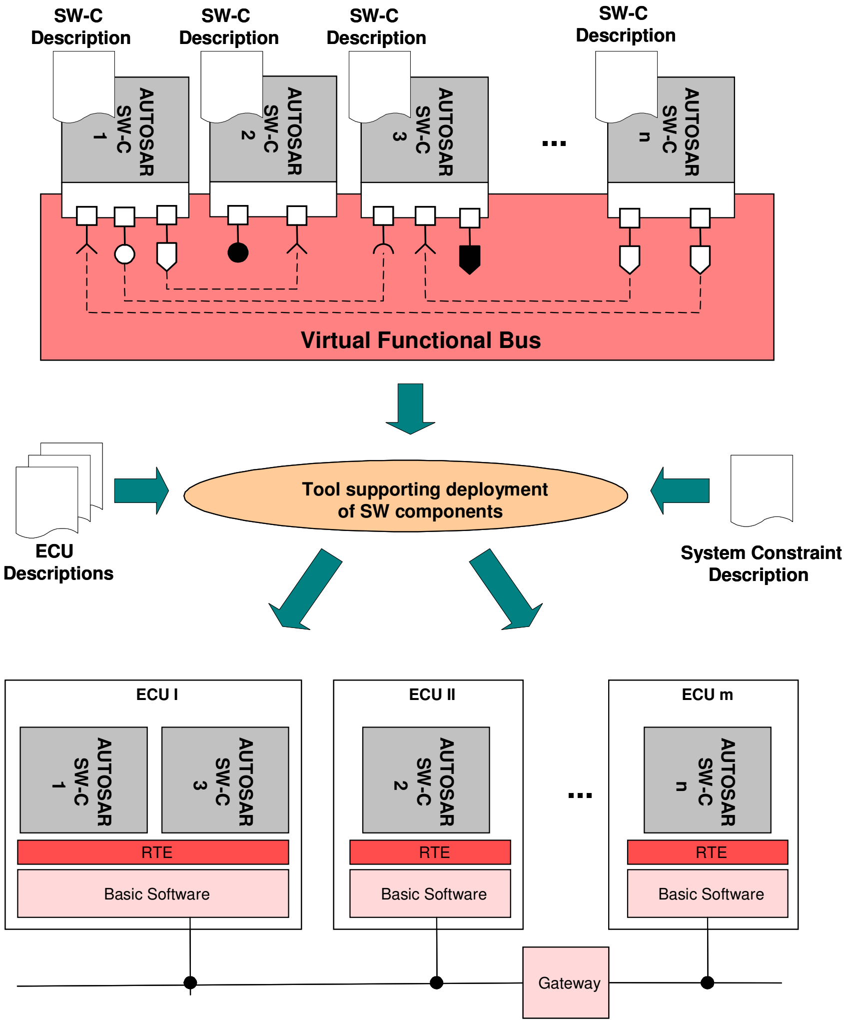
1. **For several domains a subset of application interfaces has been standardized to agreed levels.**

2. **It is a challenge to align standardization with the pace of application development.**

Main Concepts: Methodology

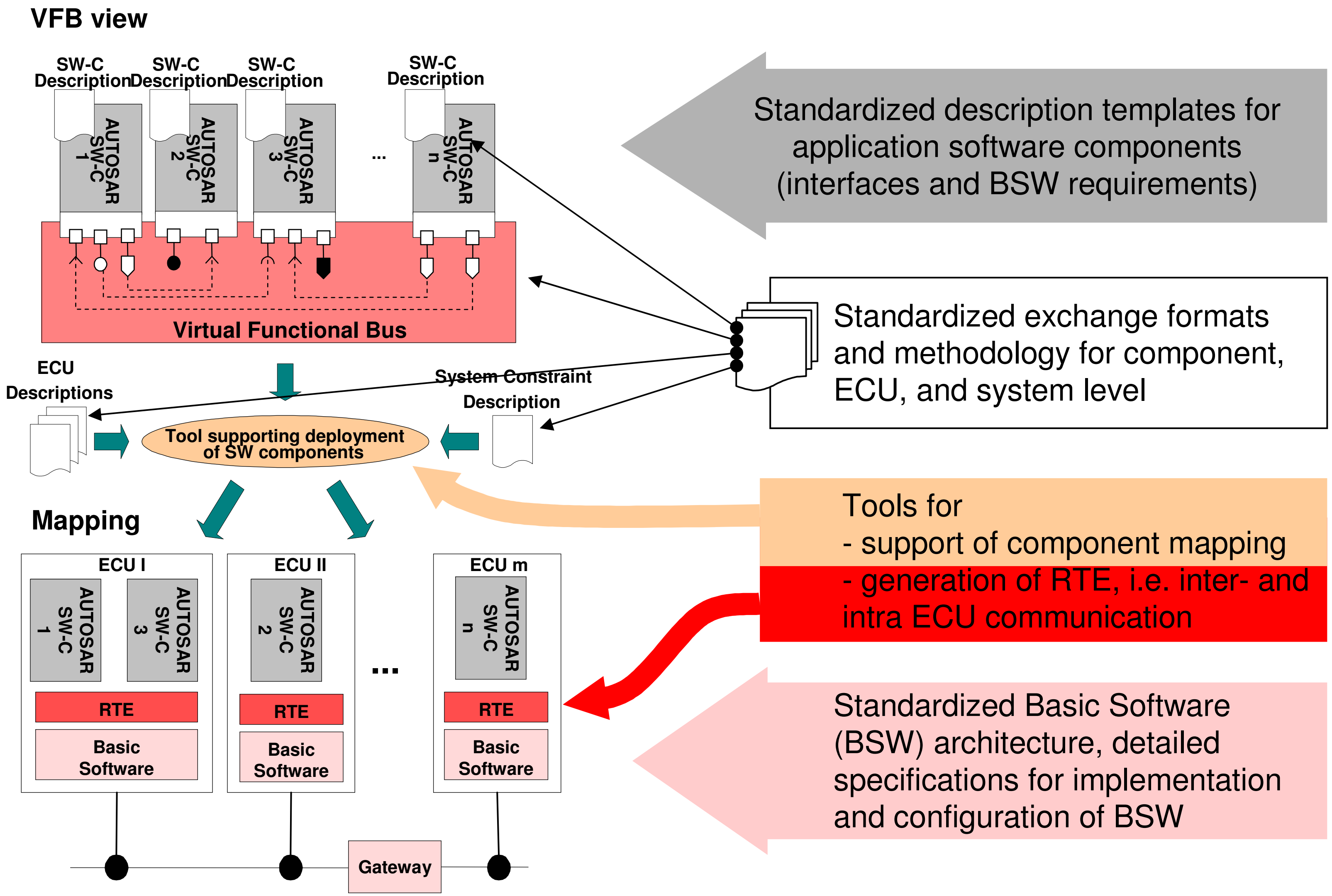
- Overall methodology
- Structure of configuration information
- System Design – Implementation Process

Following the AUTOSAR Methodology, the E/E architecture is derived from the formal description of software and hardware components.



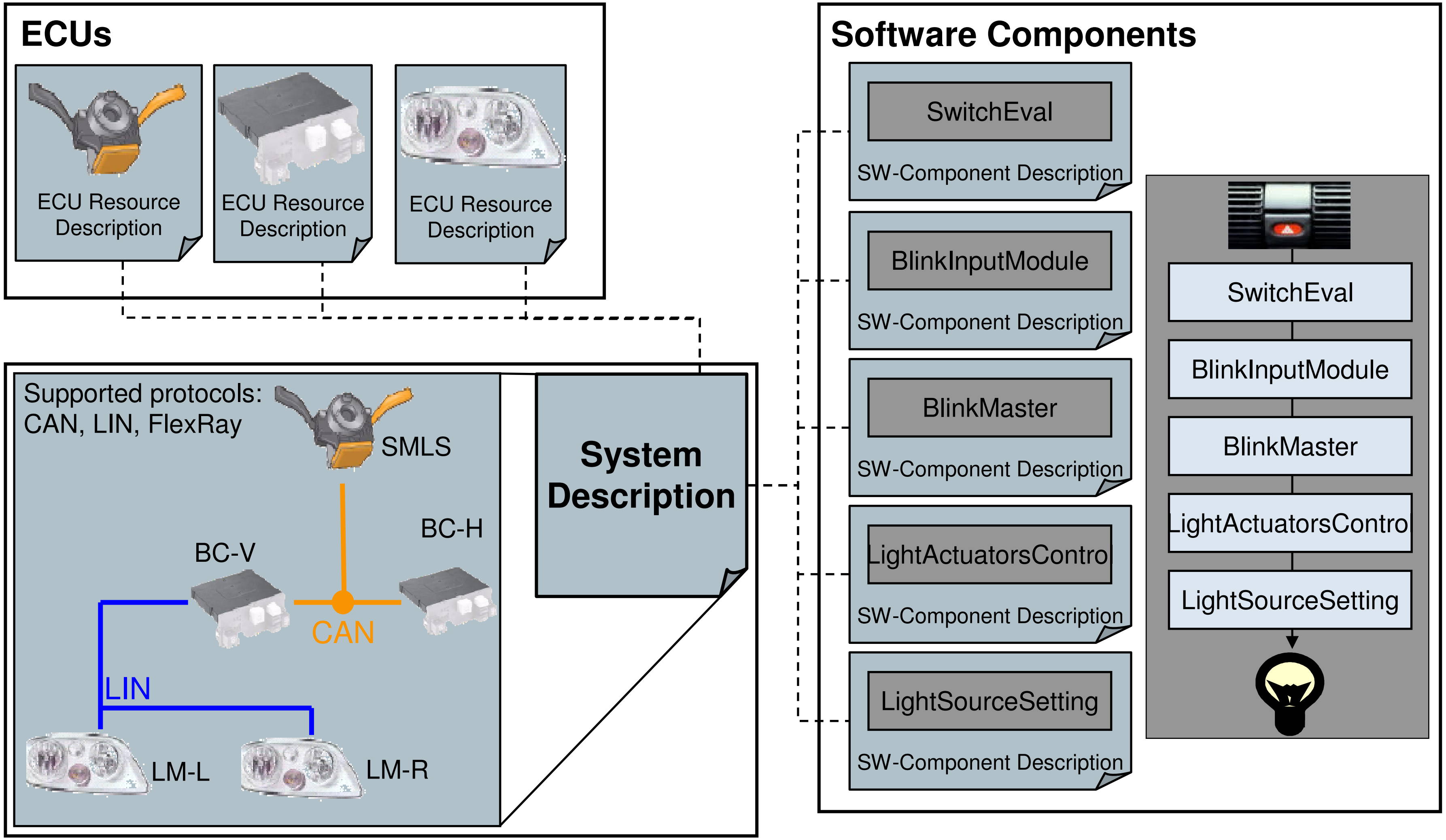
- Functional software is described formally in terms of “Software Components” (SW-C).
- Using “Software Component Descriptions“ as input, the „Virtual Functional Bus“ validates the interaction of all components and interfaces before software implementation.
- Mapping of “Software Components” to ECUs and configuration of basic software.
- The AUTOSAR Methodology supports the generation of an E/E architecture.

AUTOSAR Methodology
Derive E/E architecture from formal descriptions of soft- and hardware components



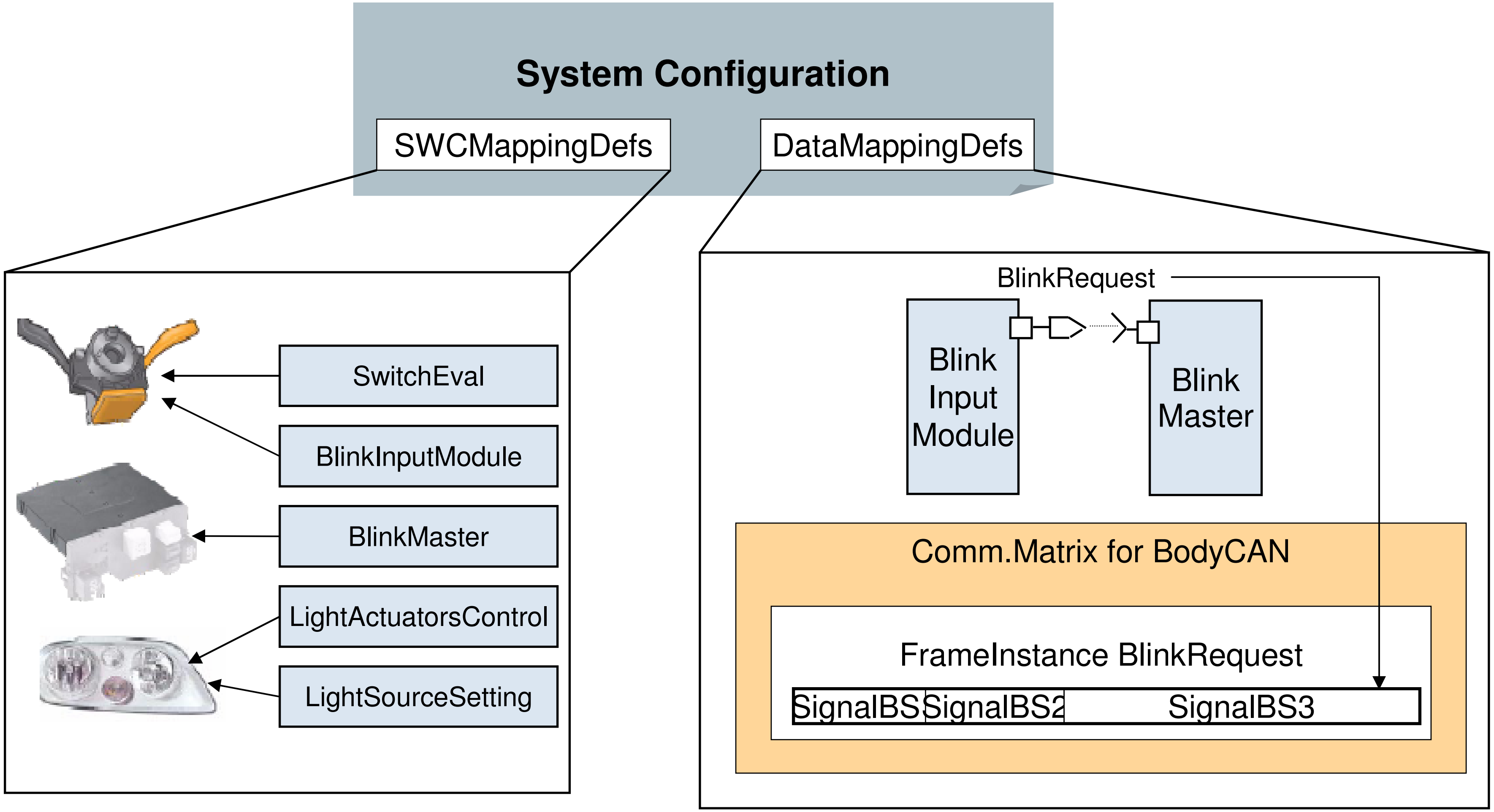
To configure the system, input descriptions of all software components, ECU resources and system constraints are necessary.

Example

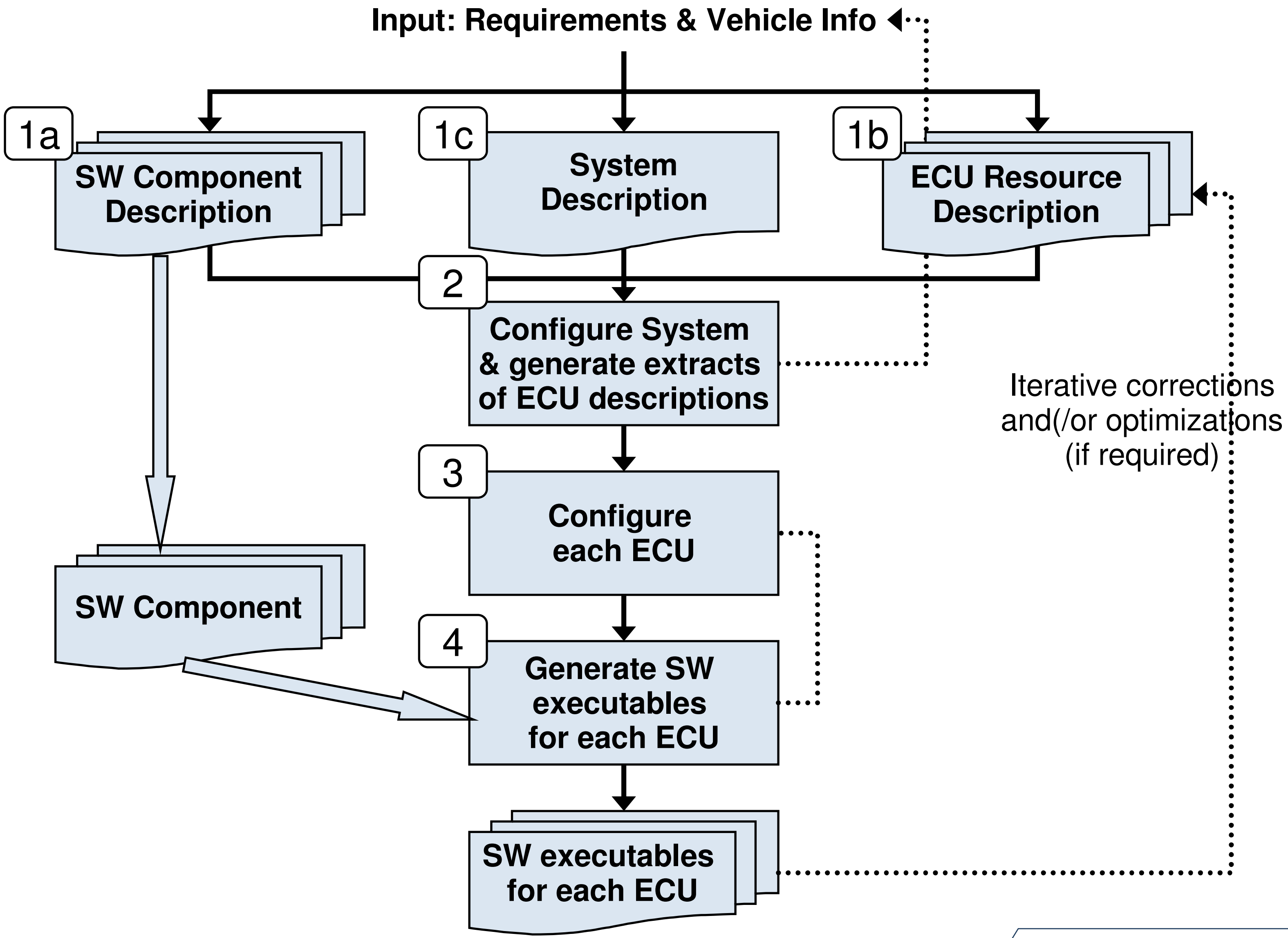


The system configuration maps software components to ECUs and links interface connections to bus signals.

Example

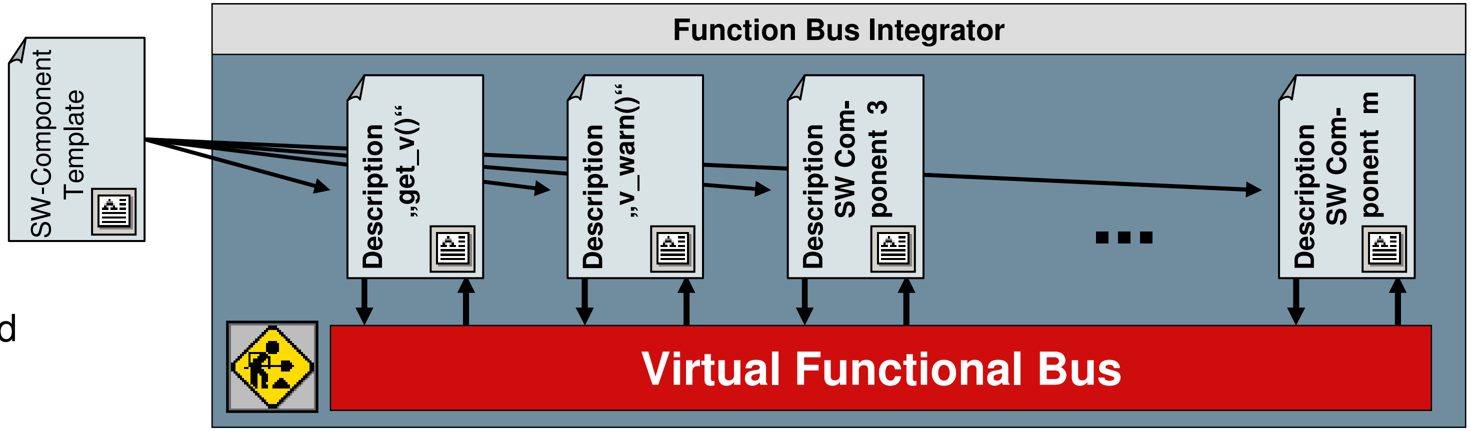


AUTOSAR – System Design – Implementation Process



AUTOSAR – The Virtual Functional Bus

Input to the System Design on an abstract level



Example: speed warning device

- SW-Component-Description „get_v()“ describes a function to acquire the current vehicle speed and defines the necessary resources (such as memory, run-time and computing power requirements, etc.)
- Function „v_warn()“ makes use of „get_v()“
- „Virtual Integration“ by check of
  - completeness of SW-Component-Descriptions (entirety of interconnections)
  - integrity/correctness of interfaces
- The Virtual Functional Bus is implemented by the AUTOSAR-Runtime-Environment (RTE) and underlying Basic-SW

= tool based

AUTOSAR – Input Descriptions (1 of 3)
Step 1a): Description of SW-Components independently of hardware

Information for each SWC
e.g. "get_v()"
- interfaces, behavior (repetition rate, ...)
- direct hardware interfaces (I/O)
- requirements on run-time performance (memory, computing power, throughput, timing/latency, …)
- ...

SW Component Description

- **General characteristics** (name, manufacturer, etc.)
- **Communication properties:**
  - p_ports
  - r_ports
  - interfaces

inner structure (composition)
  - sub-components
  - connections

- **required HW resources:**
  - processing time
  - scheduling
  - memory (size, type, etc.)

AUTOSAR-Description Editor

„get_v()"
Software Component Description

= tool based

AUTOSAR – Input Descriptions (2 of 3)
Step 1b): Description of hardware independently of application software

Information for each ECU
e.g. ECU1
- sensors and actuators
- hardware interfaces
- HW attributes (memory, processor, computing power, …)
- connections and bandwidths, etc.
- ...

ECU Resource Description

- **General characteristics** (name, manufacturer, etc.)
- **Temperature** (own, environment, cooling/heating)
- **Available signal processing methods**
- **Available programming capabilities**
- **Available HW:**
  - µC, architecture (e.g. multiprocessor)
  - memory
  - interfaces (CAN, LIN, MOST, FlexRay)
  - periphery (sensor / actuator)
  - connectors (i.e. number of pins)
- **SW below RTE for micro controller**
- **Signal path from Pin to ECU-abstraction**

AUTOSAR-Description Editor → ECU 1 Resource Description

= tool based

AUTOSAR – Input Descriptions (3 of 3)
Step 1c): Description of system

System Information overall system
- bus systems, protocols, communication matrix and attributes (e.g. data rates, timing, …)
- function clustering
- function deployment (distribution to ECU)
- ...

System Description

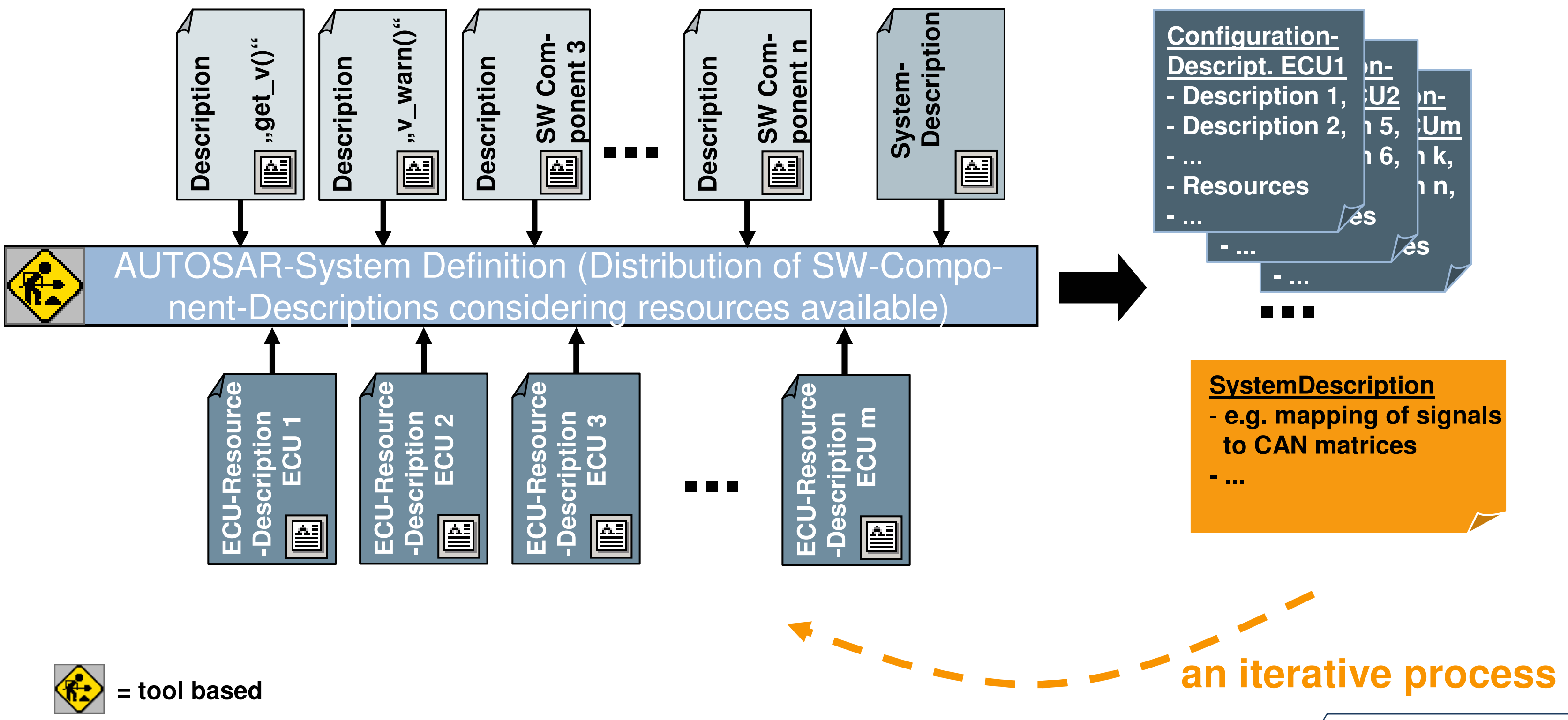
- **Network topology**
  - bus systems: CAN, LIN, FlexRay
  - connected ECUs, Gateways
  - power supply, system activation
- **Communication (for each channel)**
  - K-matrix
  - gateway table
- **Mapping / Clustering of SW components**

AUTOSAR-Description Editor → **System-Description**

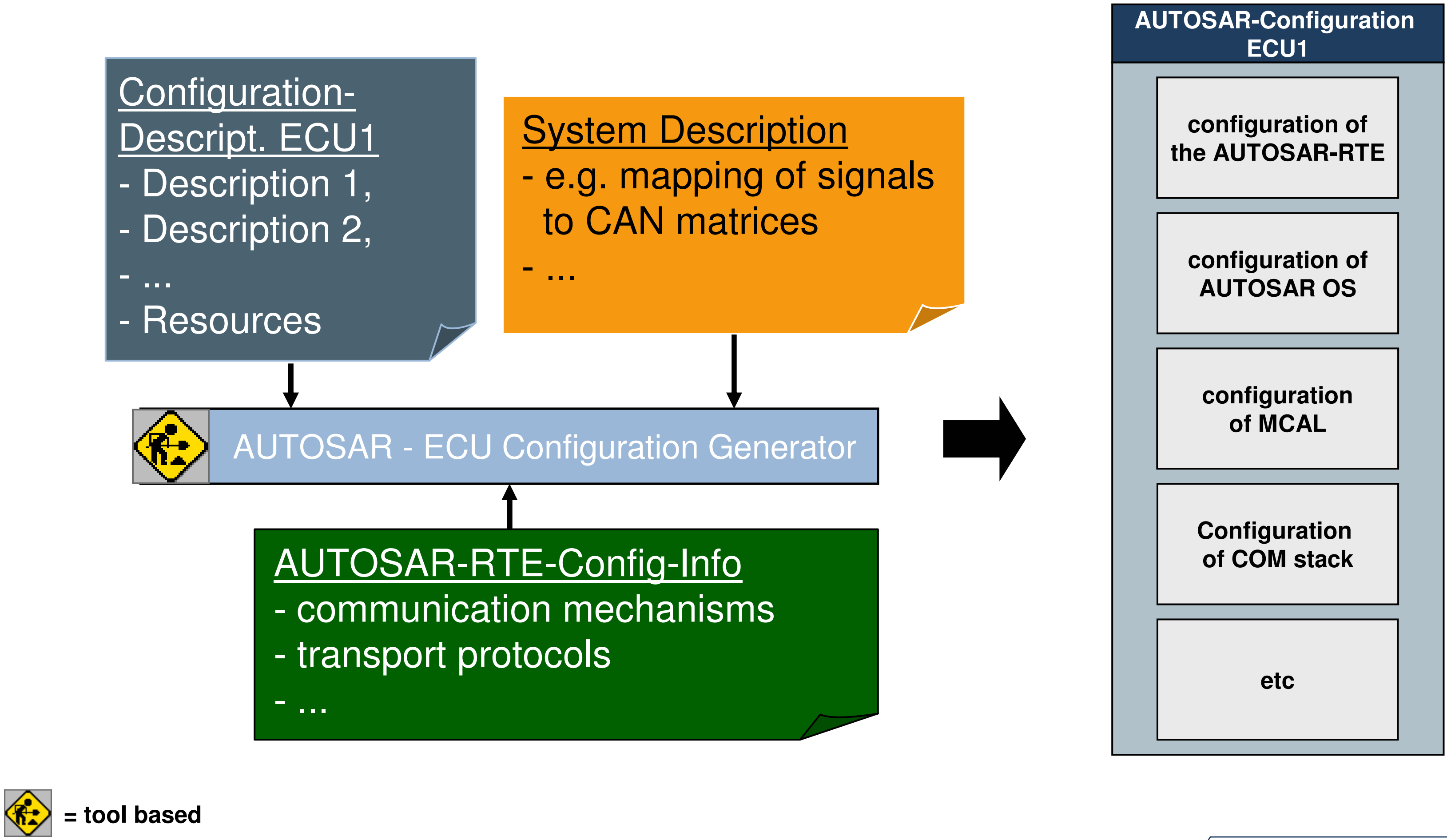
= **tool based**

AUTOSAR – System Configuration Step 2: Distribution of SW-Component-Descriptions to ECU

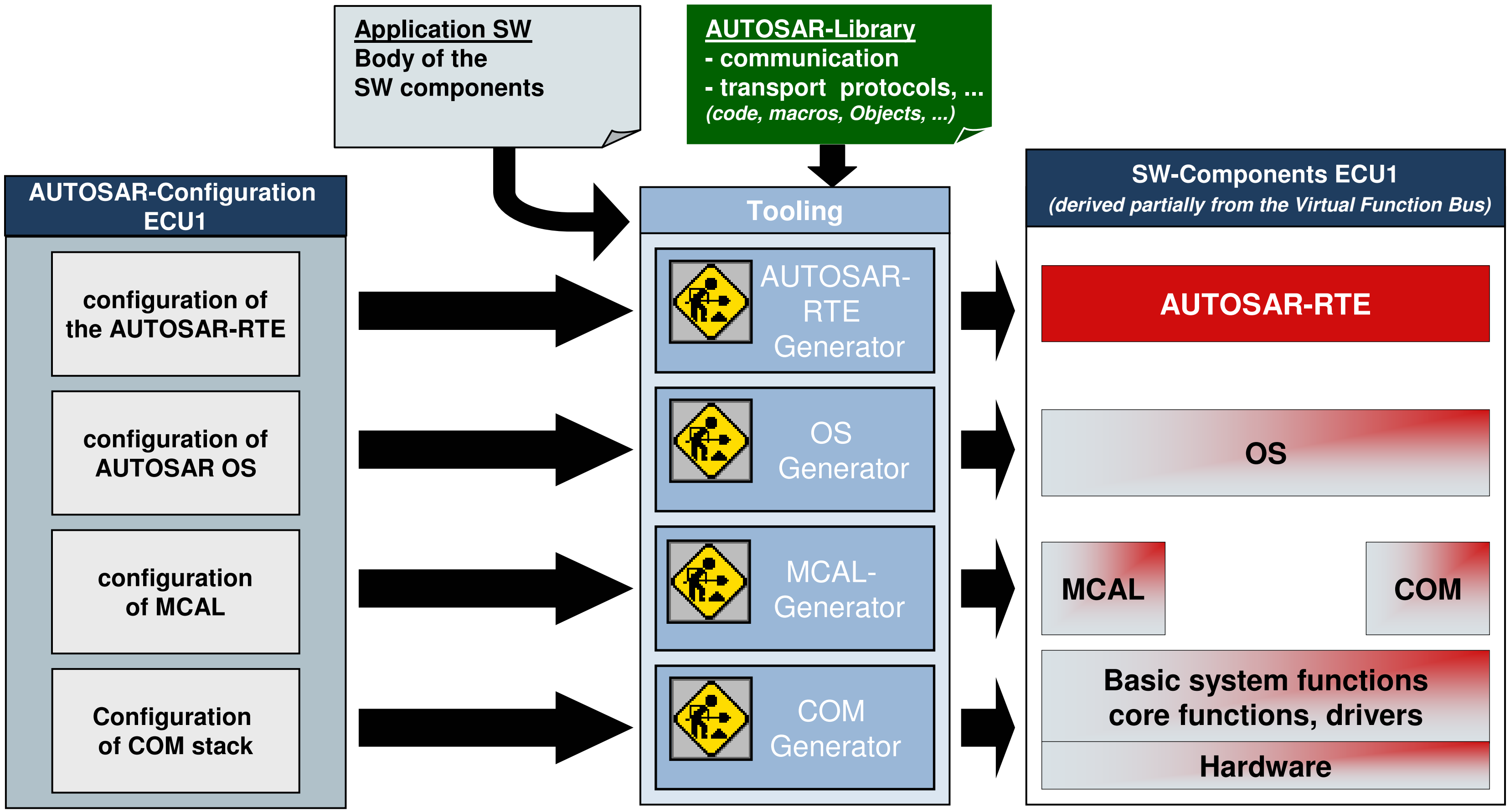
- Configuration on the basis of descriptions (not on the basis of implementations!) of SW-Components, ECU-Resources and System-Description
- Consideration of ECU-Resources available and constraints given in the System-Description



AUTOSAR – ECU-Configuration
Step 3: Generation of required configuration for AUTOSAR-Infrastructure per ECU



AUTOSAR – Generation of Software Executables
Step 4: Based on the configuration information for each ECU (example ECU1)



AUTOSAR Methodology – Conclusion

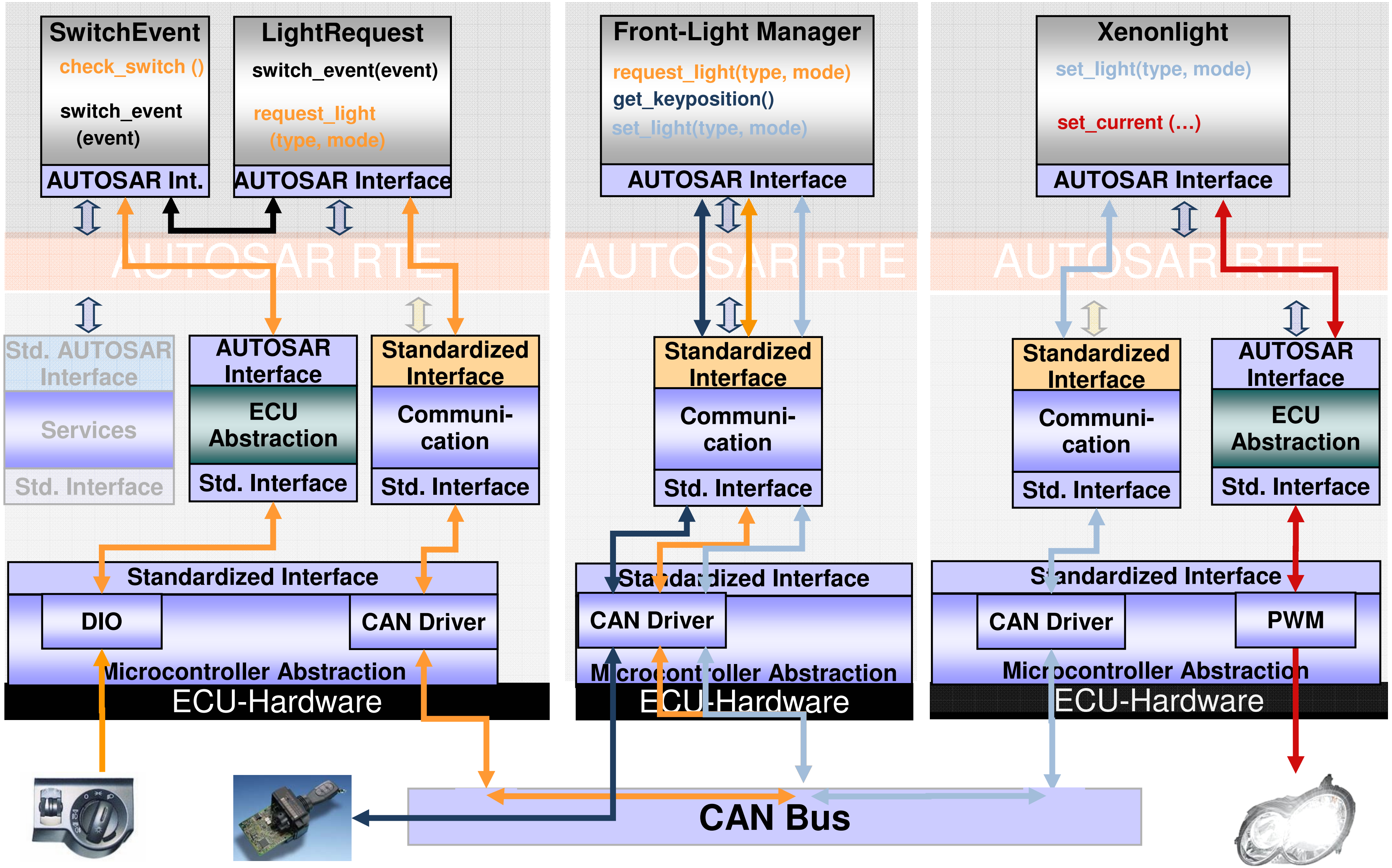
1. **The E/E system architecture can be described by means of AUTOSAR.**

2. **The meta model approach and the tool support for specifying the AUTOSAR information model allow working at the right level of abstraction.**

3. **A methodology to integrate AUTOSAR software modules has been designed.**

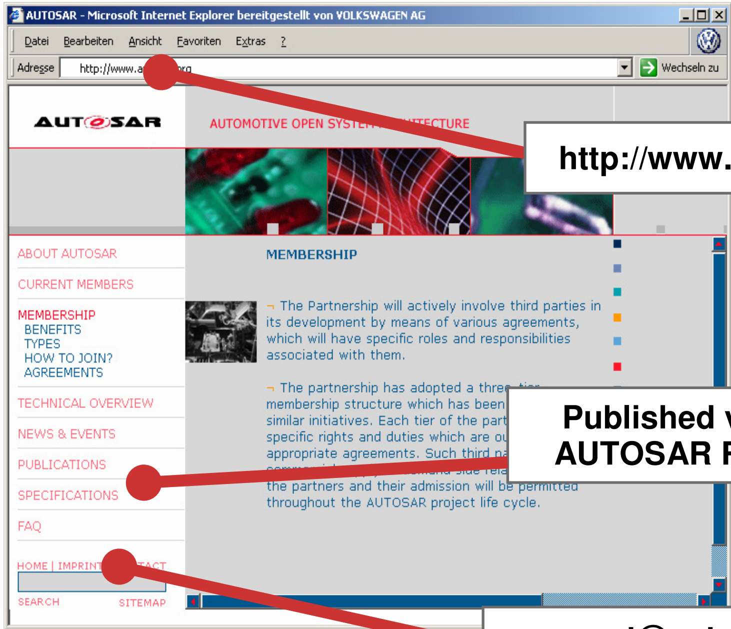
4. **AUTOSAR pushes the paradigm shift from an ECU based approach to a function based approach in automotive software development.**

Use case 'Front-Light Management' applying AUTOSAR



Further Information

http://www.carmeq.com



http://www.autosar.org

Published version of AUTOSAR Release 3.1

request@autosar.org