AUTOSAR

AUTOSAR – An open standardized software architecture for the automotive industry

Simon Fürst, BMW

1st AUTOSAR Open Conference &
8th AUTOSAR Premium Member Conference

October 23rd, 2008, Cobo Center, Detroit, MI, USA

BMW Group BOSCH Continental DAIMLER Ford OPEL PSA PEUGEOT CITROËN TOYOTA VOLKSWAGEN AG

Document Information and Change History

Document Owner	Heiko Dörr
Document Responsibility	SC
Document Version	1.5
Document filename	AUTOSAR_Tutorial
Document Status	<Draft | Ready for Approval | **Final**>

Document Change History			
Date	**Version**	**Changed by**	**Change Description**
30.05.2007	0.1	Heiko Dörr	Initial creation and proposal for content
10.08.2007	0.2	Heiko Dörr	Ready for approval
16.08.2007	0.3	Heiko Dörr	Comments by Mr. Bunzel entered
17.08.2007	0.4	Heiko Dörr	Section on Methodology updated after discussion with member of WPII-1.2; Section on Exploitation removed; Ready for approval by SC
23.08.07	1.0	Heiko Dörr	Tutorial approved by SC
31.08.07	1.1	Heiko Dörr	PM memberships updated according to slide provided by admin
07.04.08	1.2	Heiko Dörr	CP updated, readability improved
07.05.08	1.3	Heiko Dörr	Bus state managers added to BSW modules slide
03.07.2008	1.4	Heiko Dörr	Selected slides from ATI added (Animated use case for distributed scenario, Validator 2), Schedule updated
01.10.2008	1.5	Heiko Dörr	Selection of slides for presentation at 8th PM conference

Document Information and Change History

Document Change History			
Date	**Version**	**Changed by**	**Change Description**
22.10.2008	1.51	Simon Fürst	Review

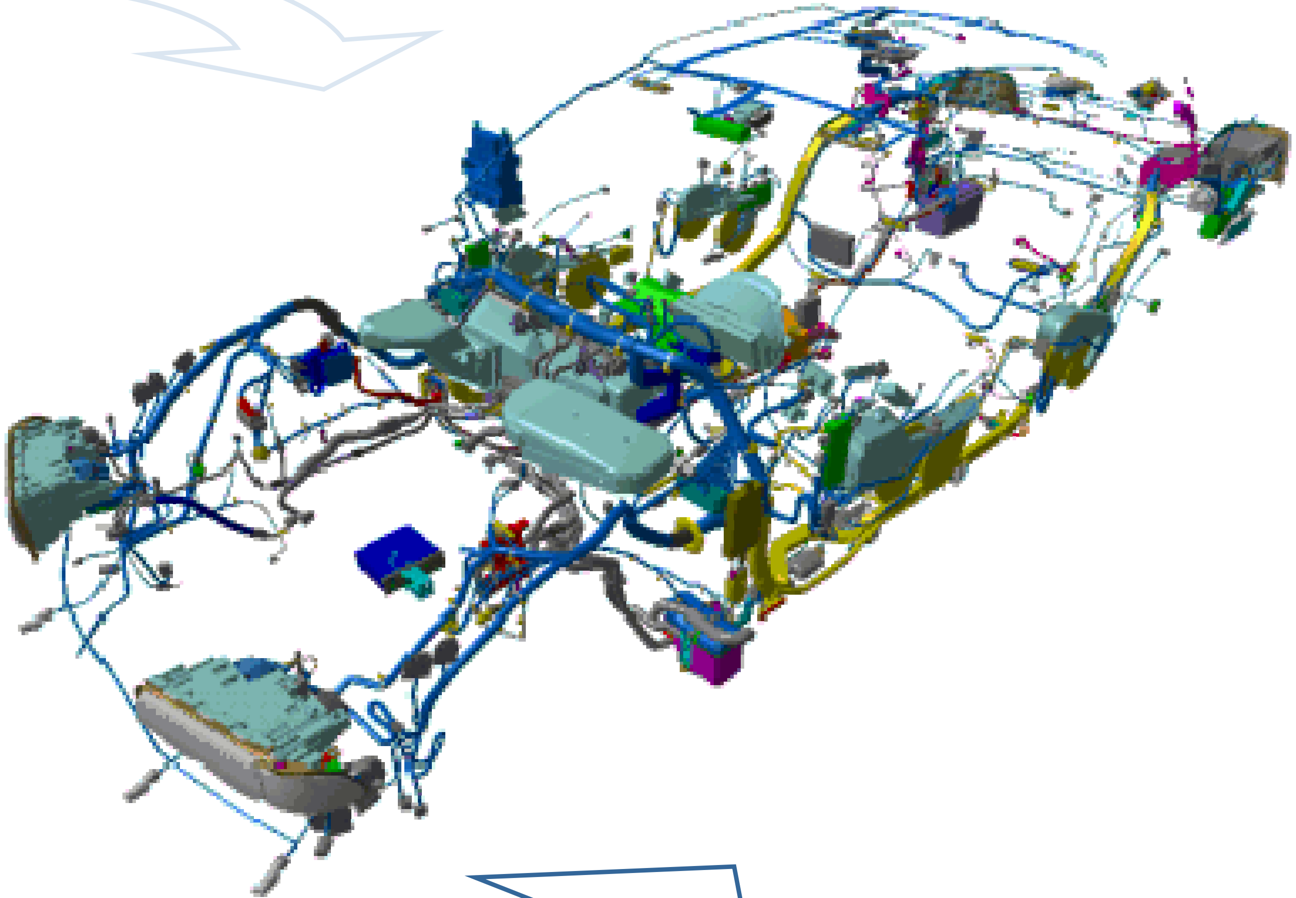
Automotive Systems and SW Engineering

Automotive Open System Architecture
Cooperate on standards – compete on implementation

Non functional legal requirements

Vehicle family management

Resource efficiency



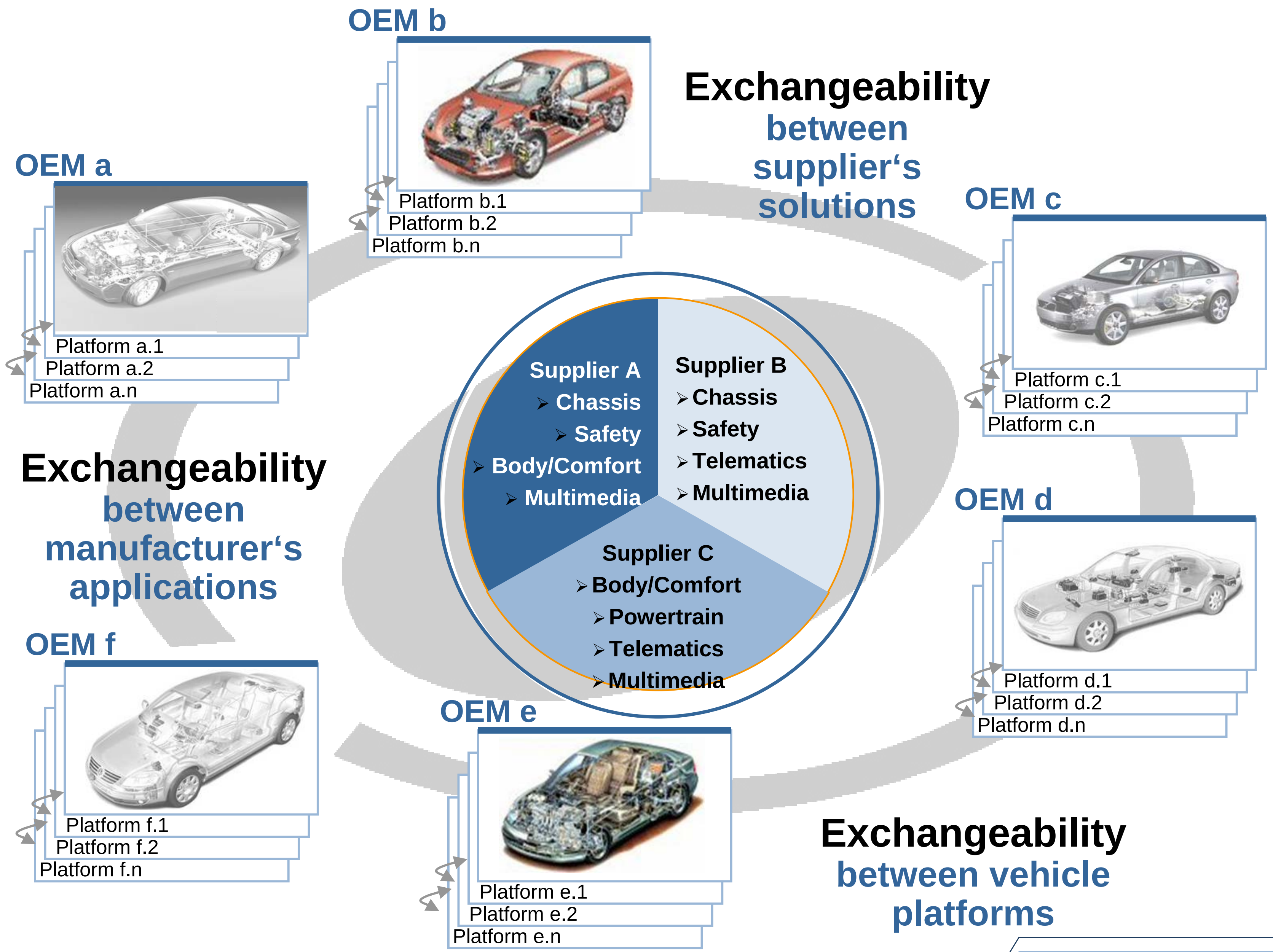
Driver assistance

Driving dynamics

Safety functions (active/passive)

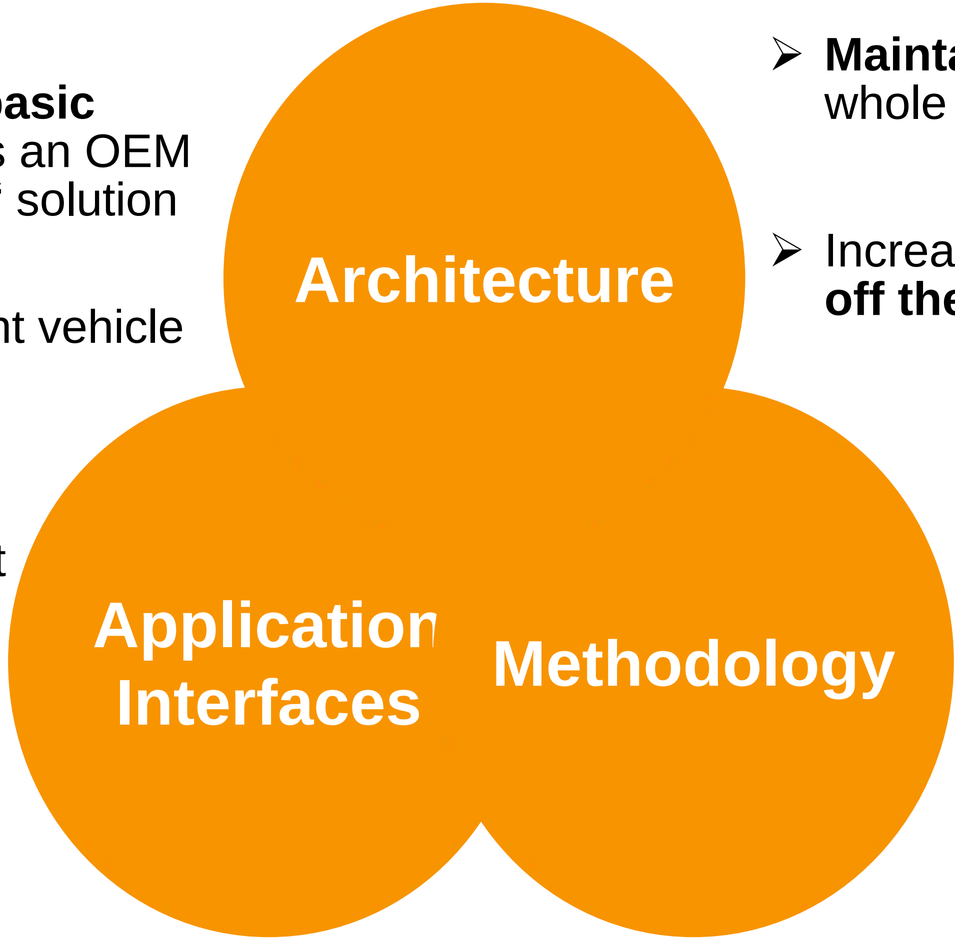
Comfort functions

AUTOSAR Managing Complexity by Exchangeability and Reuse of Software Components



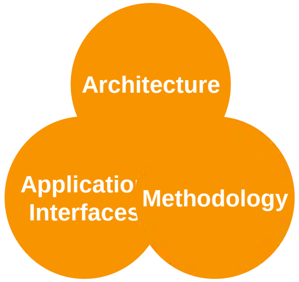
AUTOSAR
Project Objectives and Main Working Topics

- 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**

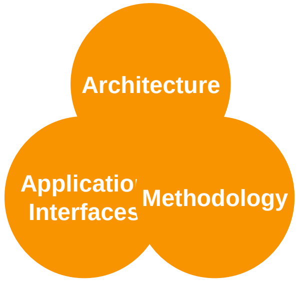


- **Maintainability** throughout the whole “Product Life Cycle“
- Increased use of **“Commercial off the shelf hardware“**
- **Software updates** and upgrades over vehicle lifetime
- Consideration of availability and **safety** requirements
- **Redundancy** activation

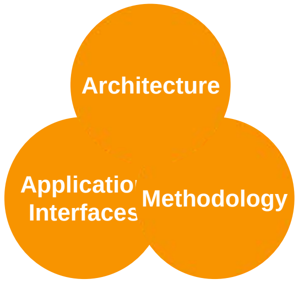
AUTOSAR
Main Working Topics



- **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.

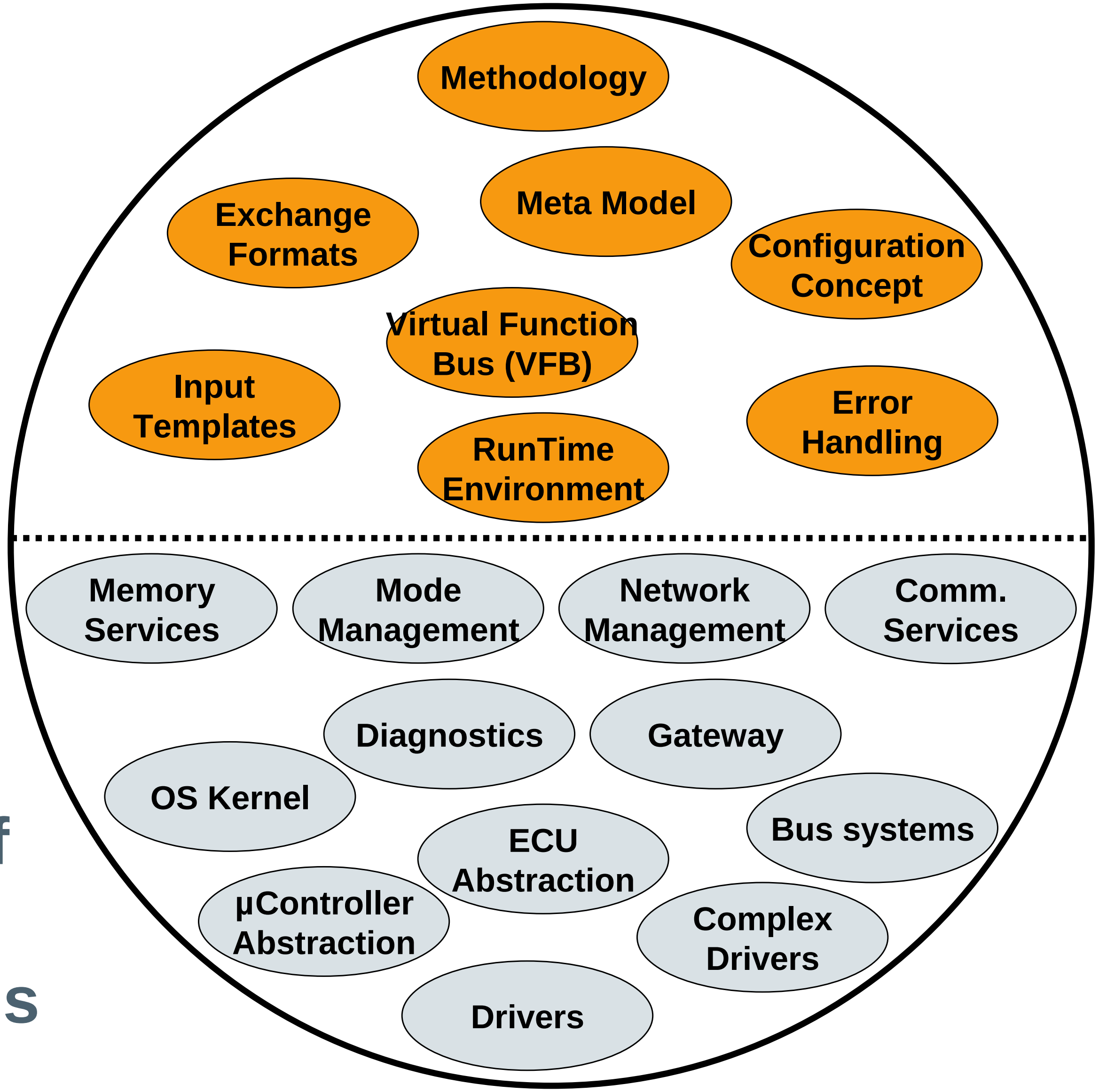


- **Application Interfaces:**
  Specification of interfaces of typical automotive applications from all domains in terms of syntax and semantics, which should serve as a standard for application software.

Technical scope of AUTOSAR

New concepts

Industry-wide consolidation of ‚existing‘ basic software designs



Benefits from AUTOSAR

OEM

- OEM overlapping reuse of software modules
- Maintaining ability to compete on innovative functions, enlarged design flexibility
- Simplification of the integration task
- Reduction of total SW development costs

Supplier

- Reduction of version proliferation
- Development partitioning among suppliers
- Increase of efficiency in functional development
- New business models possible

Tool provider

- Common interfaces with development processes
- Seamless, manageable, task optimized (time dependent) tool landscape

New market entrant

- Transparent and defined interfaces enable new business models

Project Setup Phase II

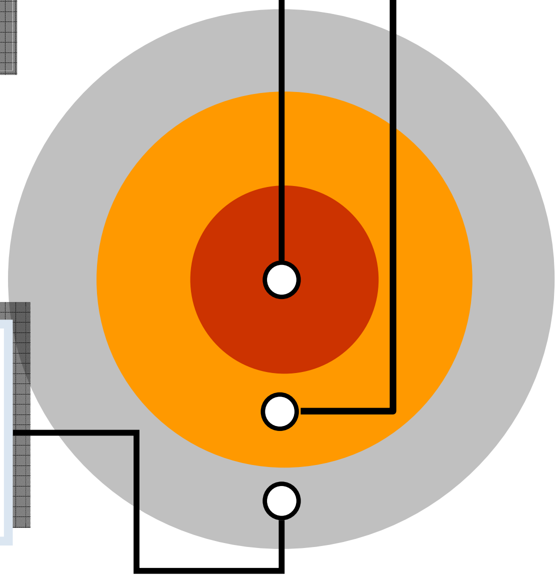
AUTOSAR – Partnership Structure

Core Partners
(OEM & Tier 1 Supplier)

- Organizational control
- Technical contributions
- Administrative control
- Definition of external Information (web-release, clearance, etc.)
- Leadership of Working Groups
- Involvement in Working Groups

Premium Members

- Leadership of Working Groups
- Involvement in Working Groups
- Technical contributions
- Access to current information



Support Roles

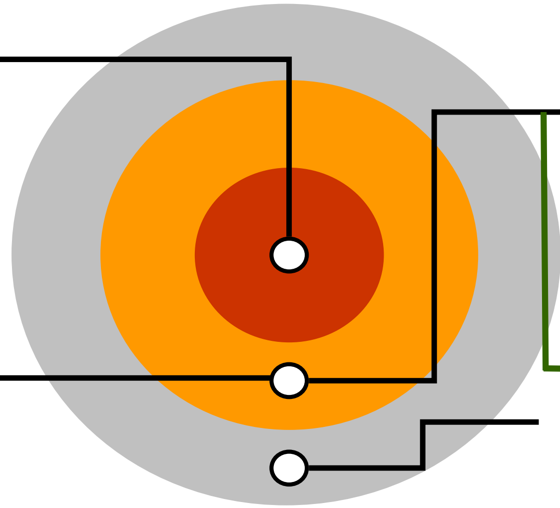
- *Development Members*
- *Attendees*

Associate Members

- Access to finalized documents
- Utilization of AUTOSAR standard

AUTOSAR – Core Partners and Members

Status: 10th October 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

85 Associate Member

55 Premium Member

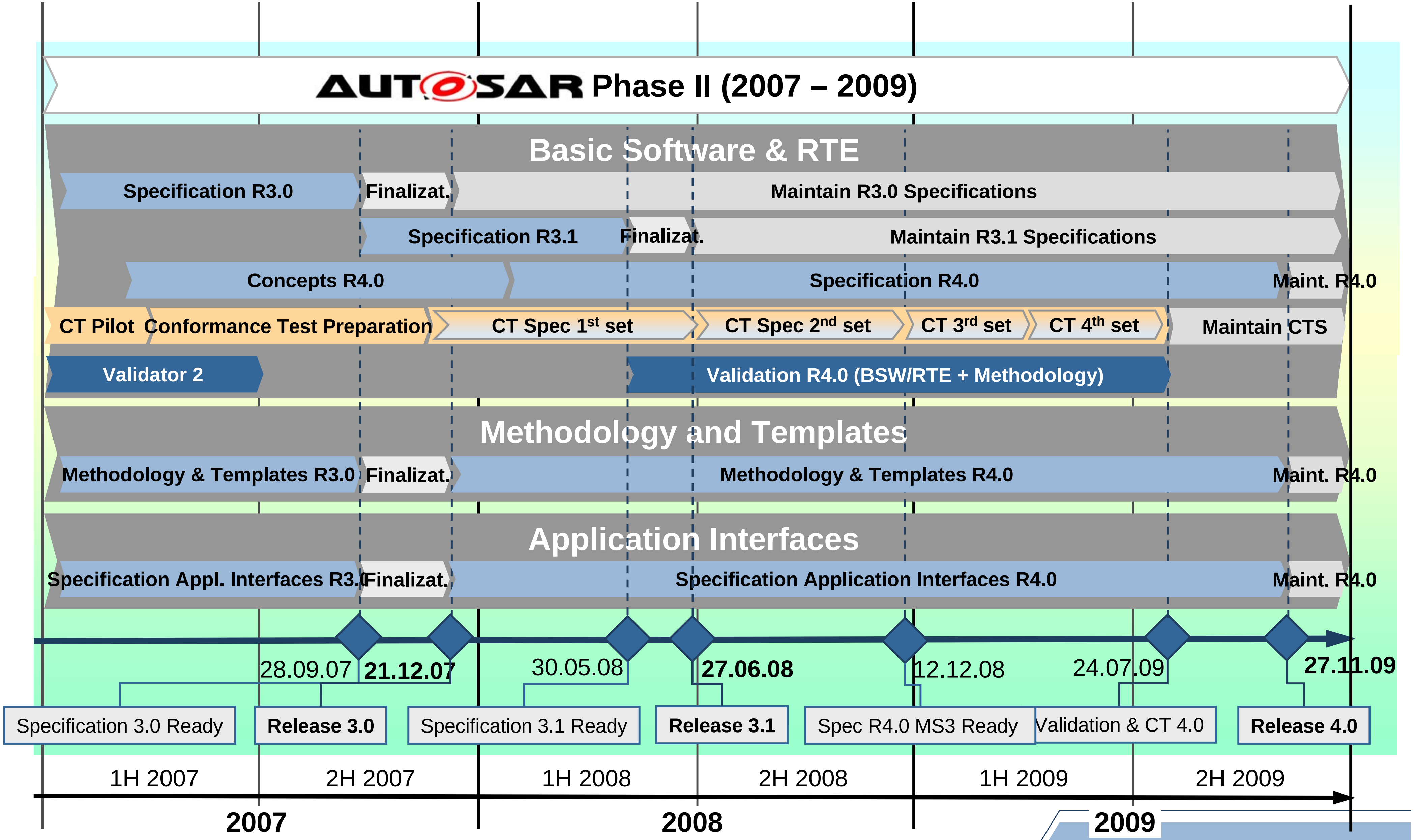
General OEM	Generic Tier 1	Standard Software	Tools and Services	Semi-conductors
Fiat Group Automobiles SpA, HONDA, HYUNDAI·KIA MOTORS, NISSAN, mazda, RENAULT, PORSCHE, MAN, VOLVO	Autoliv, ALPINE, DELPHI, DENSO, HELLA, Johnson Controls, LEAR CORPORATION, MAGNA, TRW, Valeo, Visteon, ZF, ZF Lenksysteme	intecs, HITACHI, bi2, MBtech, MAGNETI MARELLI, softing, geensys, vector	TATA ELXSI LIMITED, KPIT Cummins Infosystems Limited, dSPACE, EB, ETAS, IAV GmbH, ESTEREL Technologies, Infosys, FEV, Mentor Graphics, TTTech, The MathWorks, Telelogic, IBM, TietoEnator, WIPRO	ARM, freescale semiconductor, patni, Infineon, FUJITSU, NEC, NXP, ST, RENESAS

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

AUTOSAR Phase II 2007 – 2009

- AUTOSAR Development Partnership continues
- Identical Core Partners
- Exploitation and maintenance
  - Already in 2008 the first cars on the road with AUTOSAR technology inside
  - All Core Partners have planned the introduction of AUTOSAR products until 2010
  - Establish conformance test specifications and process
- Further development and amendment of the standard, e.g.
  - Functional safety features
  - Support for multi core microcontrollers
  - Vehicle & application mode management
  - Debugging and error handling
  - Variant handling
  - Timing model
  - Standardization of application interfaces

Top Level Schedule for AUTOSAR in phase II



AUTOSAR Phase II
Work Package Breakdown Structure

Work Packages

- **II-1 System Architecture**
  - II-1.1 Software Architecture
    - WPII-1.1.1 Software Architecture and OS
    - WPII-1.1.2 Vehicle and Application Mode Mgmt.
    - WPII-1.1.3 Debugging
    - WPII-1.1.4 Error Handling
    - WPII-1.1.5 VFB and RTE
  - WPII-1.2 Methodology and Configuration
  - WPII-1.3 Functional Safety
- **II-2 Software and Test Specification**
  - II-2.1 Basic Software
    - WPII-2.1.1 COM Stack
    - WPII-2.1.2 FlexRay
    - WPII-2.1.3 MCAL
    - WPII-2.1.4 Diagnostics
    - WPII-2.1.5 Libraries
  - WPII-2.2 Conformance Test Specification
- **II-3 Validation**
  - WPII-3.1 Basic Software Validation
  - WPII-3.2 Methodology Validation
- **II-4 Enabling Exploitation**
  - WPII-4.2 Communication and Marketing
  - WPII-4.3 Follow-up Organization
- **II-5 Maintenance of Releases**
  - WPII-5.1 Problem Management
  - WPII-5.2 Change and Release Mgmt.
  - WPII-5.3 Maintenance of Specifications
- **II-10 Application Interfaces**
  - WPII-10.0 Coordination of Appl. Interfaces
  - WPII-10.1 Body and Comfort
  - WPII-10.2 Powertrain
  - WPII-10.3 Chassis Control
  - WPII-10.4 Occupants and Pedest. Safety
  - WPII-10.5 MM / T / HMI

Use Cases of AUTOSAR Results

- Exchange of SW-Components
- Re-use of SW components for different platforms

… shown by uses cases

pedal management

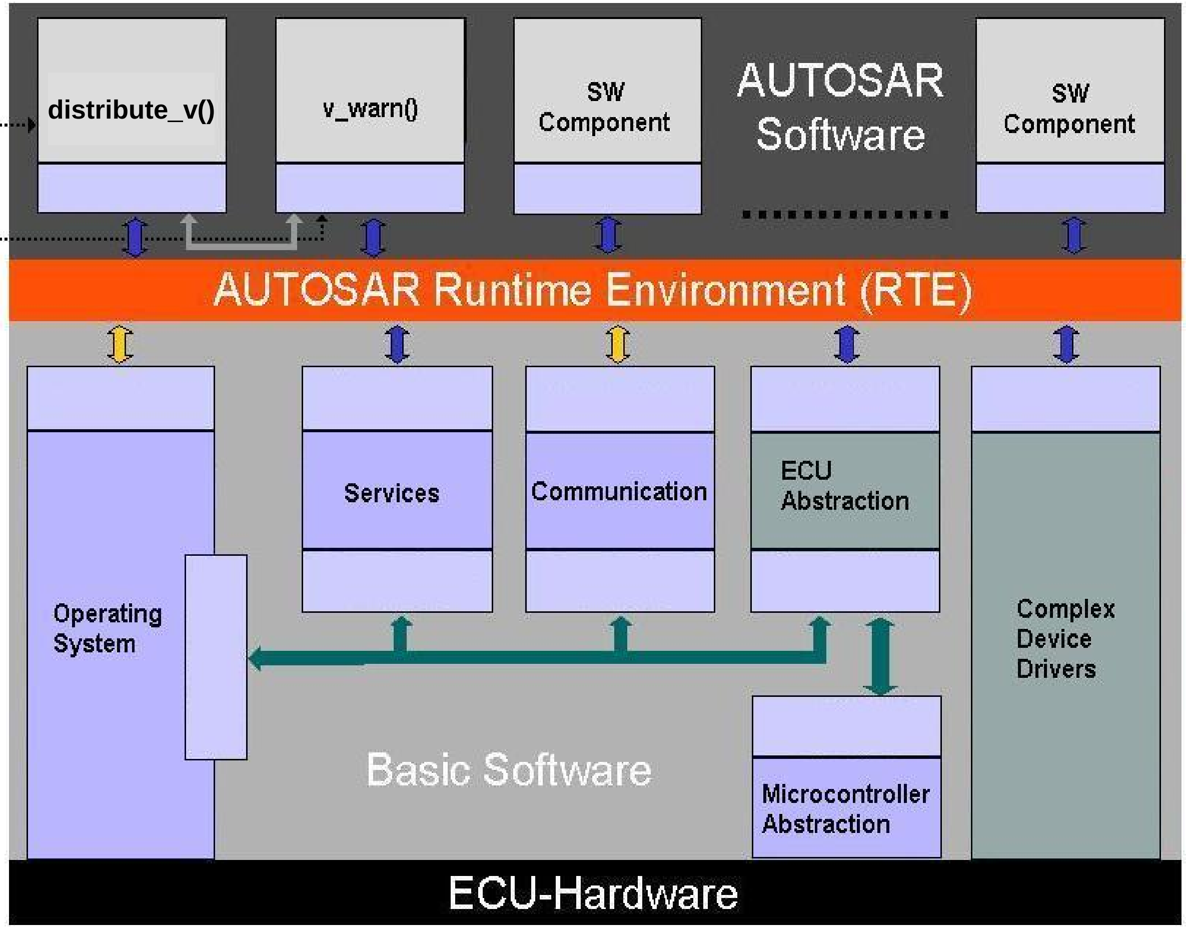
front light management

Use Case 'Pedal Management' view for one ECU

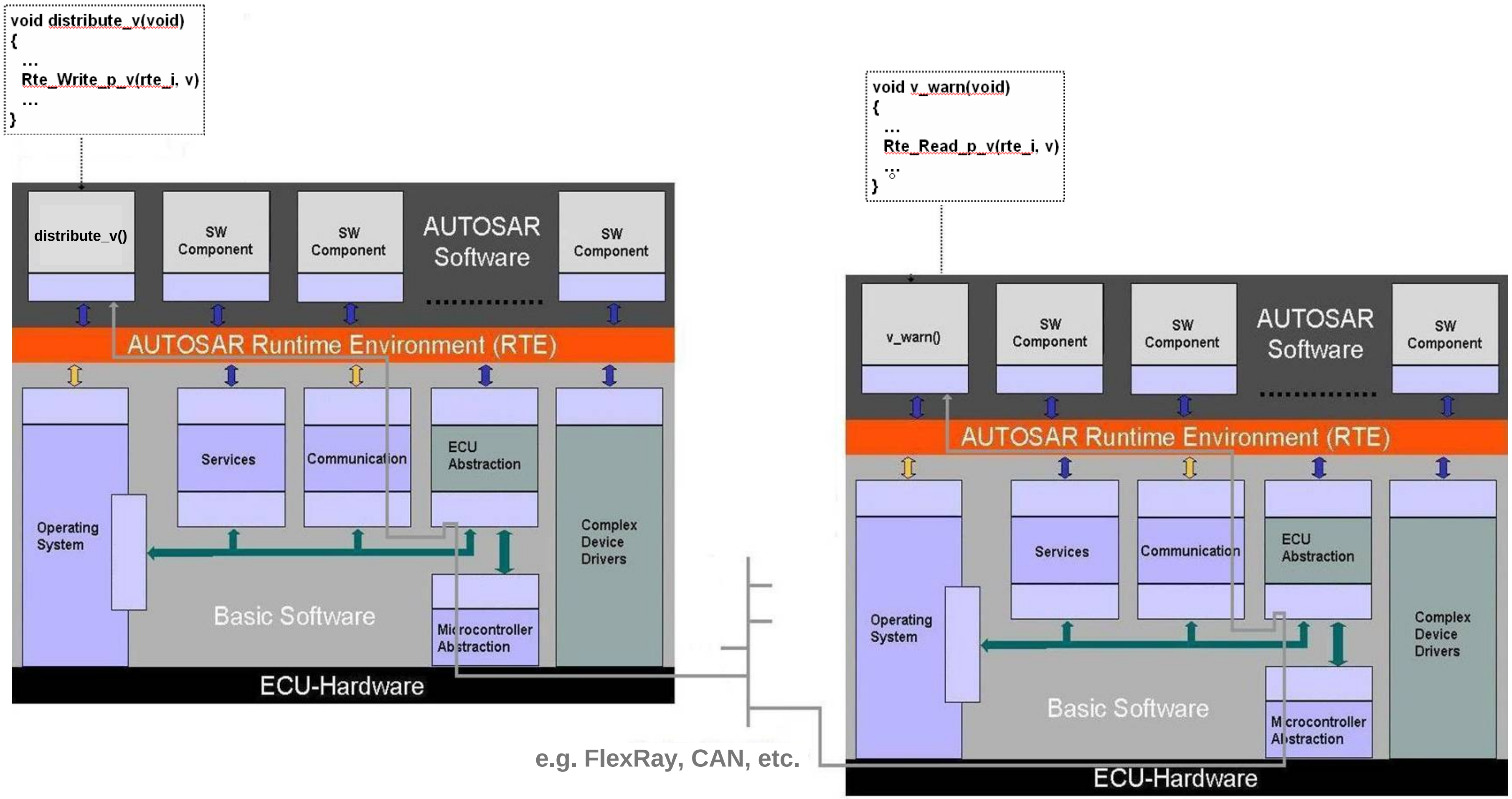
- Implementation of functions independent on distribution on different ECU as communication will be done via ECU-individual AUTOSAR-RTE exclusively

```
void distribute_v(void)
{
  …
  Rte_Write_p_v(rte_i, v)
  …
}
```

```
void v_warn(void)
{
  …
  Rte_Read_p_v(rte_i, v)
  …
}
```



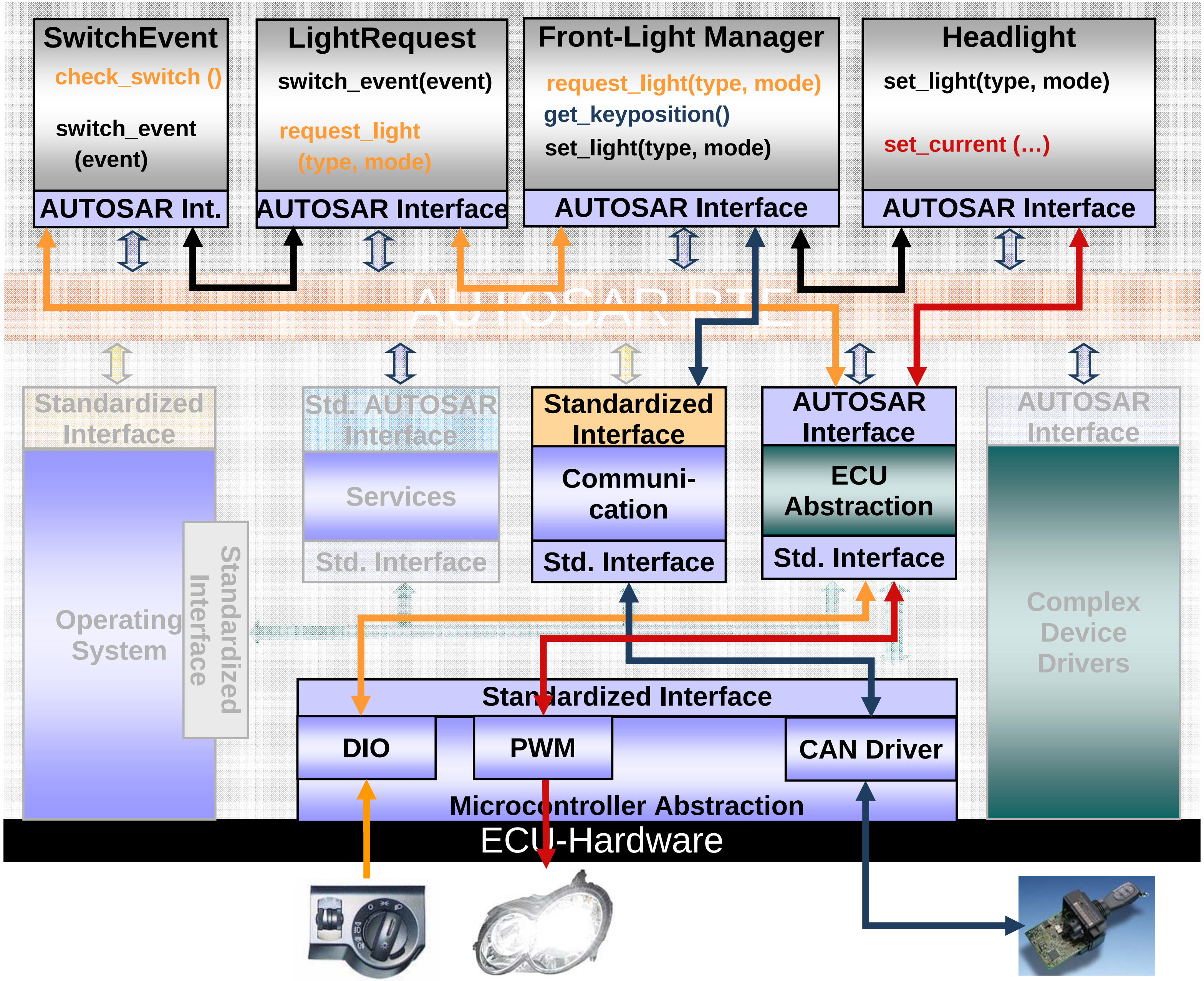
Use Case 'Pedal Management' view for two ECUs



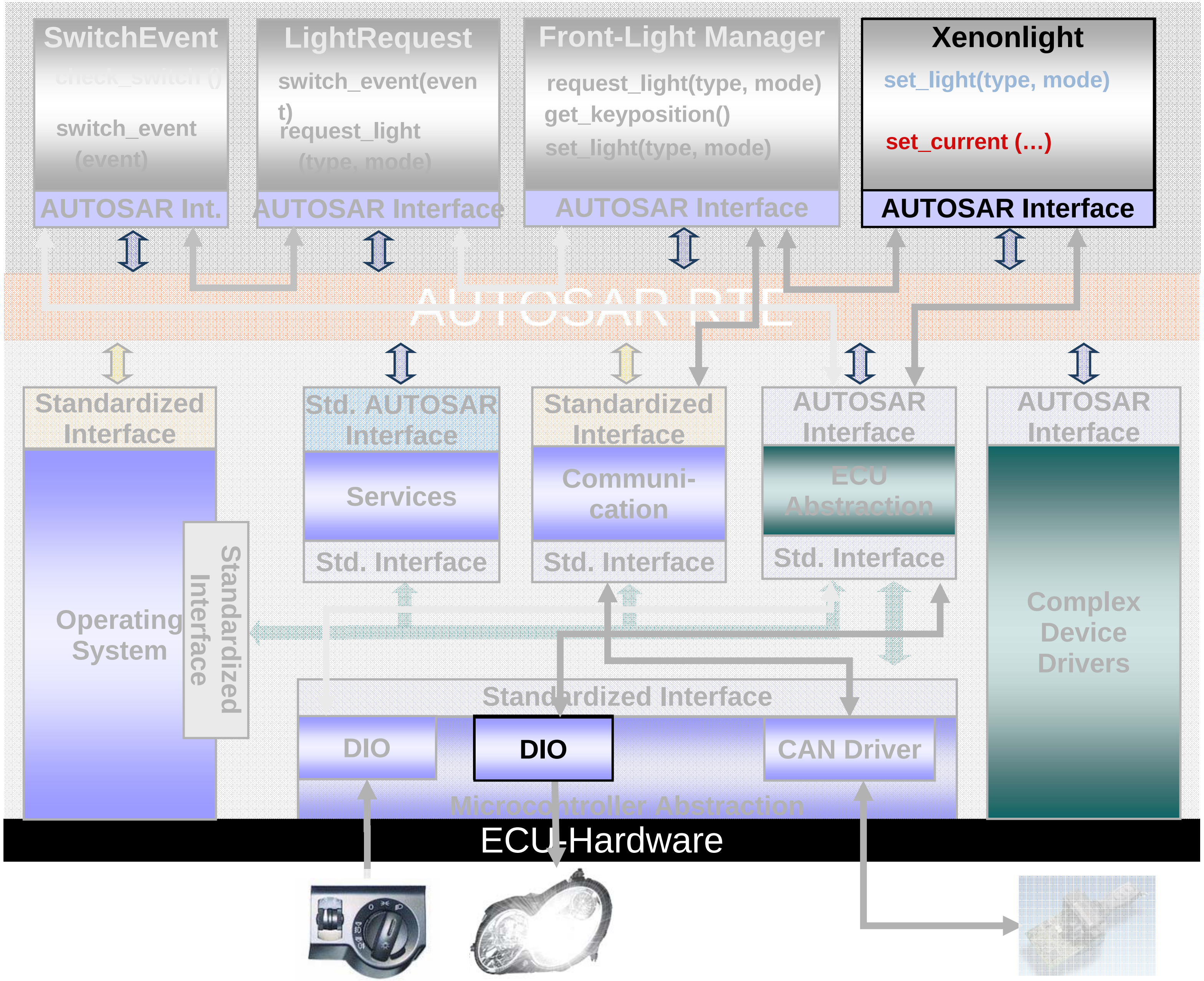
Technical benefits

- Reuse of Intellectual Property
- Increase in design flexibility
- Simplification of the integration task
- Reduction of SW development costs

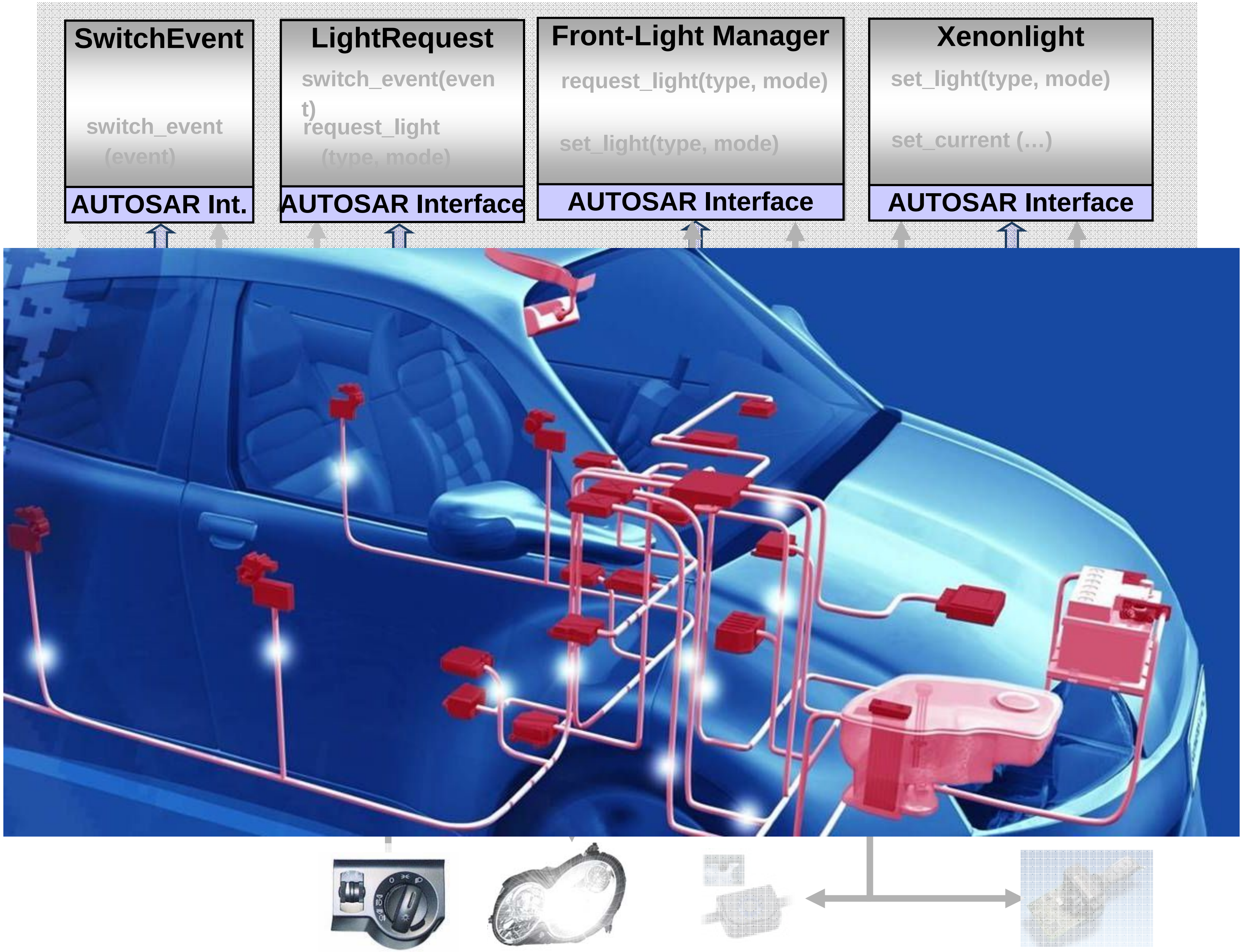
Use case 'Front-Light Management' in AUTOSAR



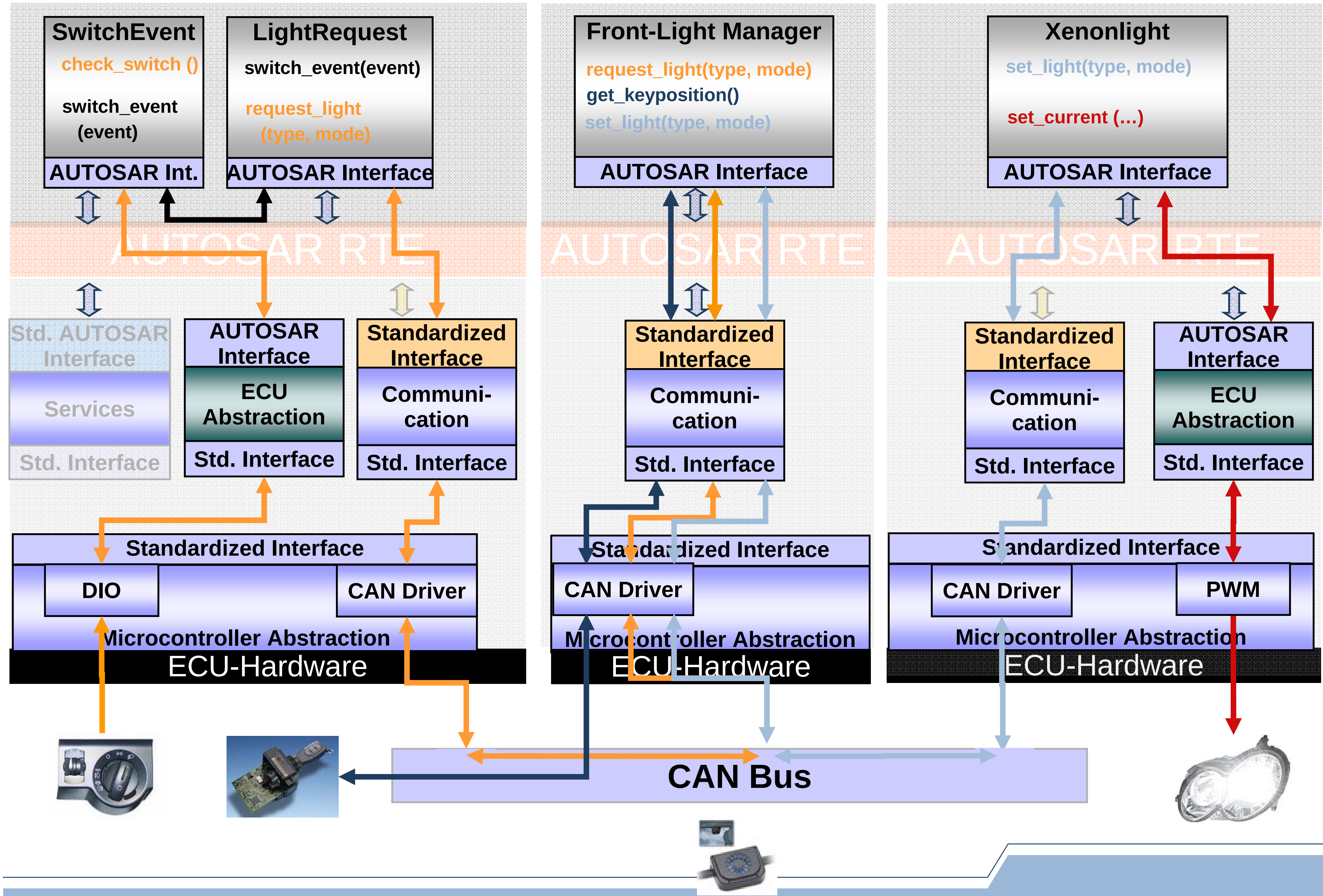
Exchange of type of front-light



Distribution on ECUs



Use case ‘Front-Light Management’ in AUTOSAR



AUTOSAR Key Topics

AUTOSAR provides three main areas of results:

- Architecture:
  Software architecture including a complete basic (environmental) software stack for an ECU as an integration platform for hardware independent SW applications

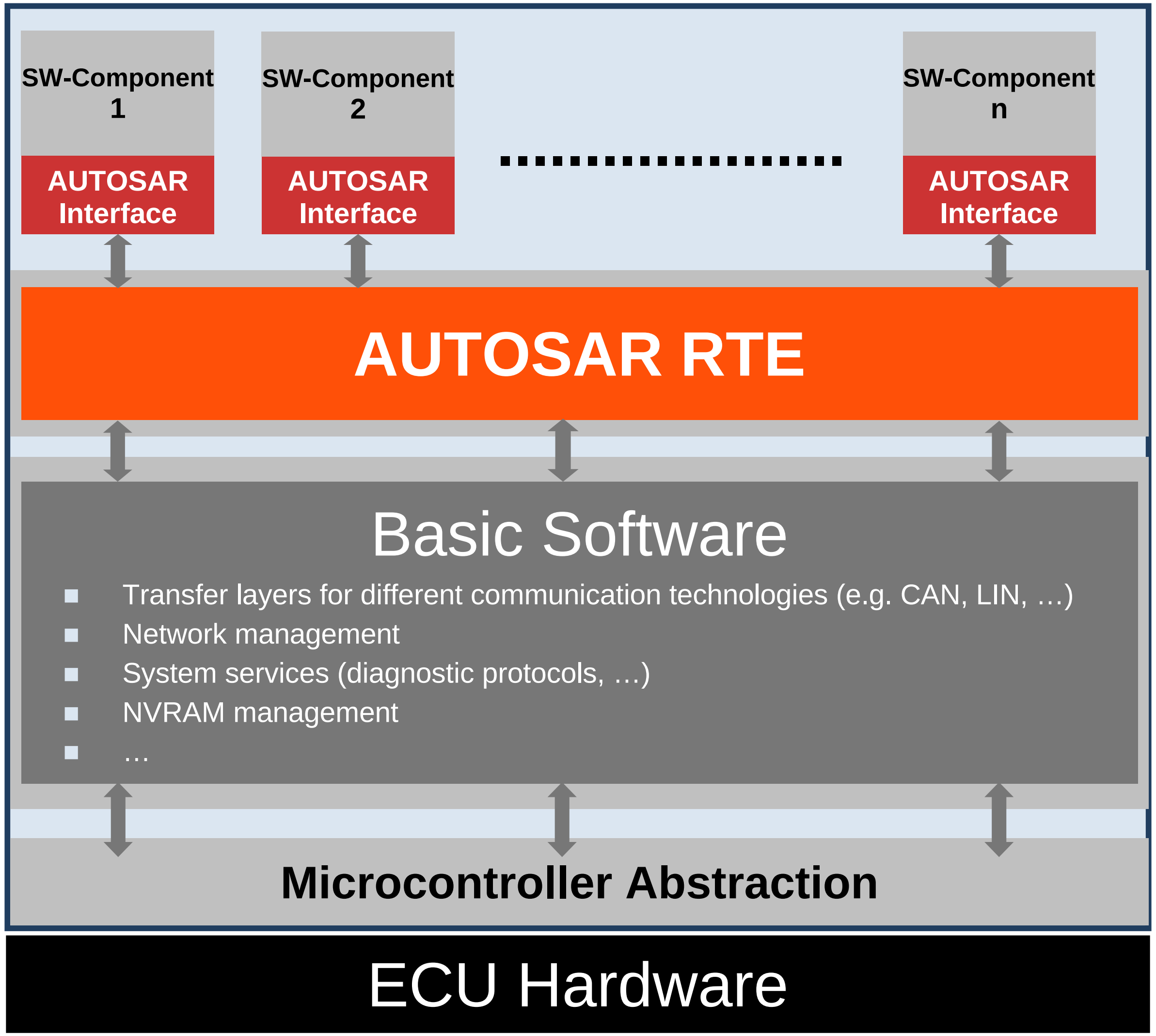
- Methodology:
  Exchange formats (templates) to enable a seamless configuration process of the basic software stack and the integration of application software in ECUs

- Application Interfaces:
  Specification of application interfaces as a standard for application software modules

Main Concepts: Architecture

- Basic Software modules
- Run time environment and communication
- Results of sample implementation in „Validator 2“

Standardized AUTOSAR interfaces will support HW independence and enable the standardization of SW components.



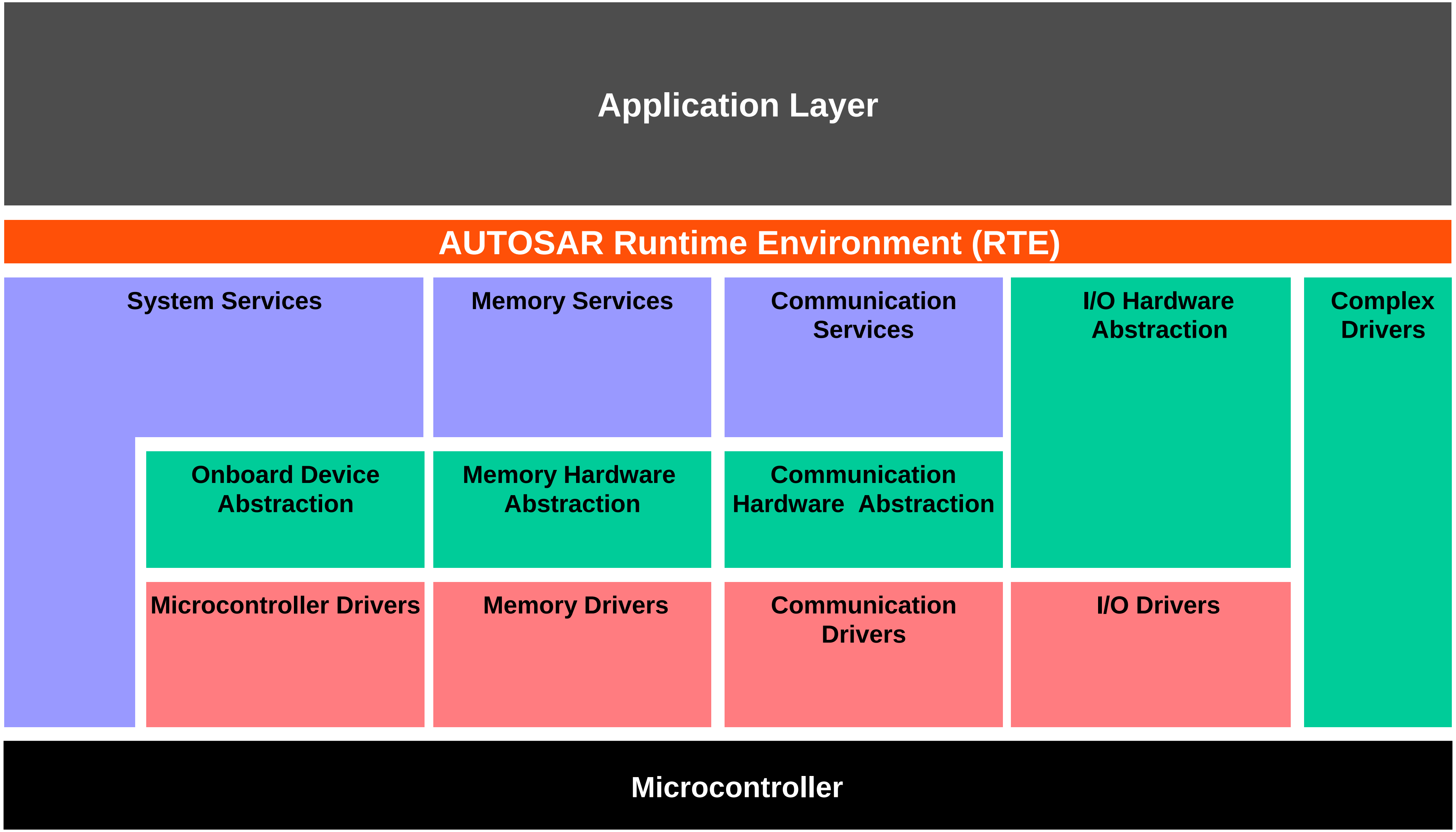
Automotive Open System Architecture (AUTOSAR):

- Standardized, openly disclosed interfaces
- HW independent SW layer
- Transferability of functions
- Redundancy activation

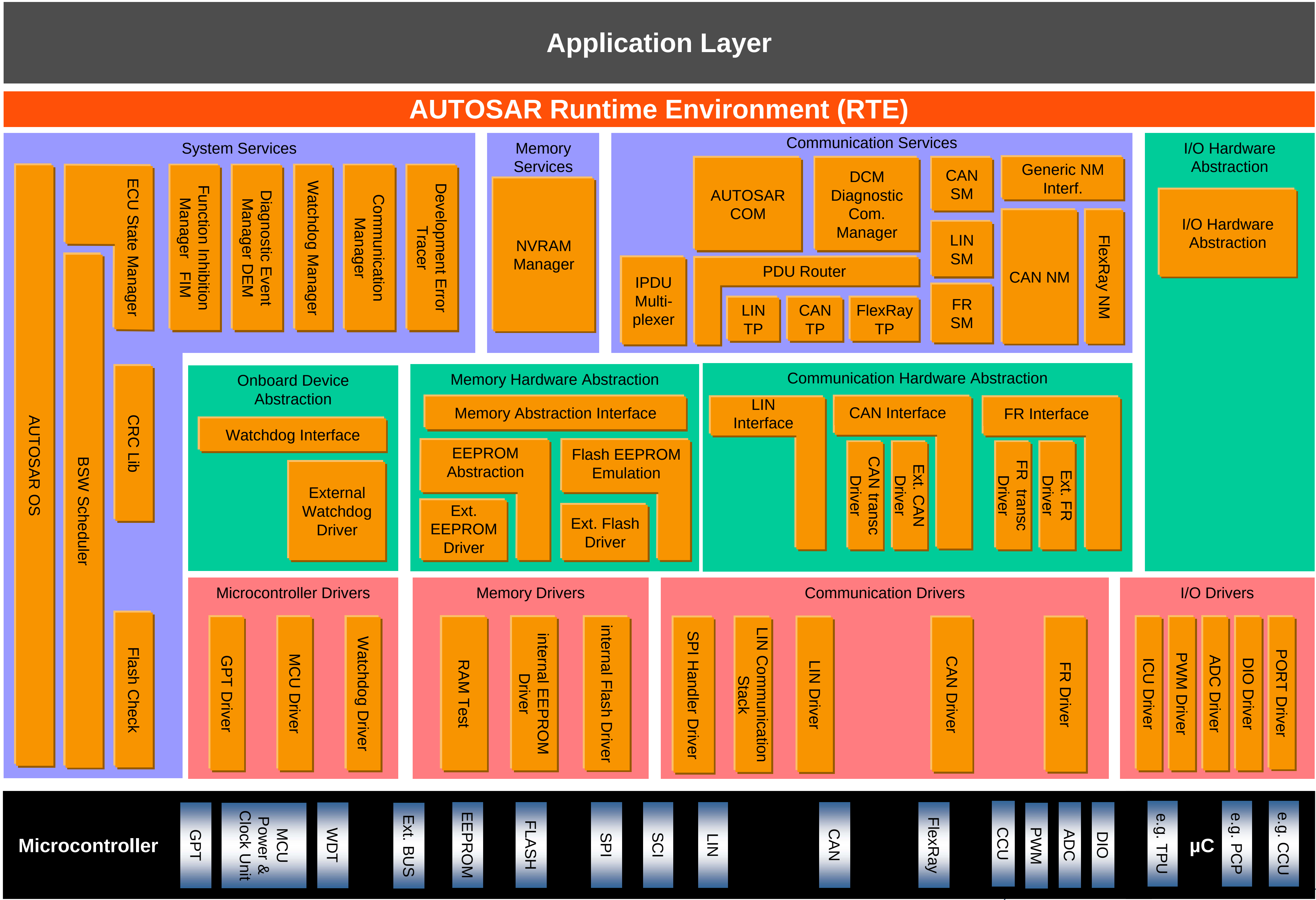
AUTOSAR RTE:

by specifying interfaces and their communication mechanisms, the applications are decoupled from the underlying HW and Basic SW, enabling the realization of Standard Library Functions.

AUTOSAR Basic Software

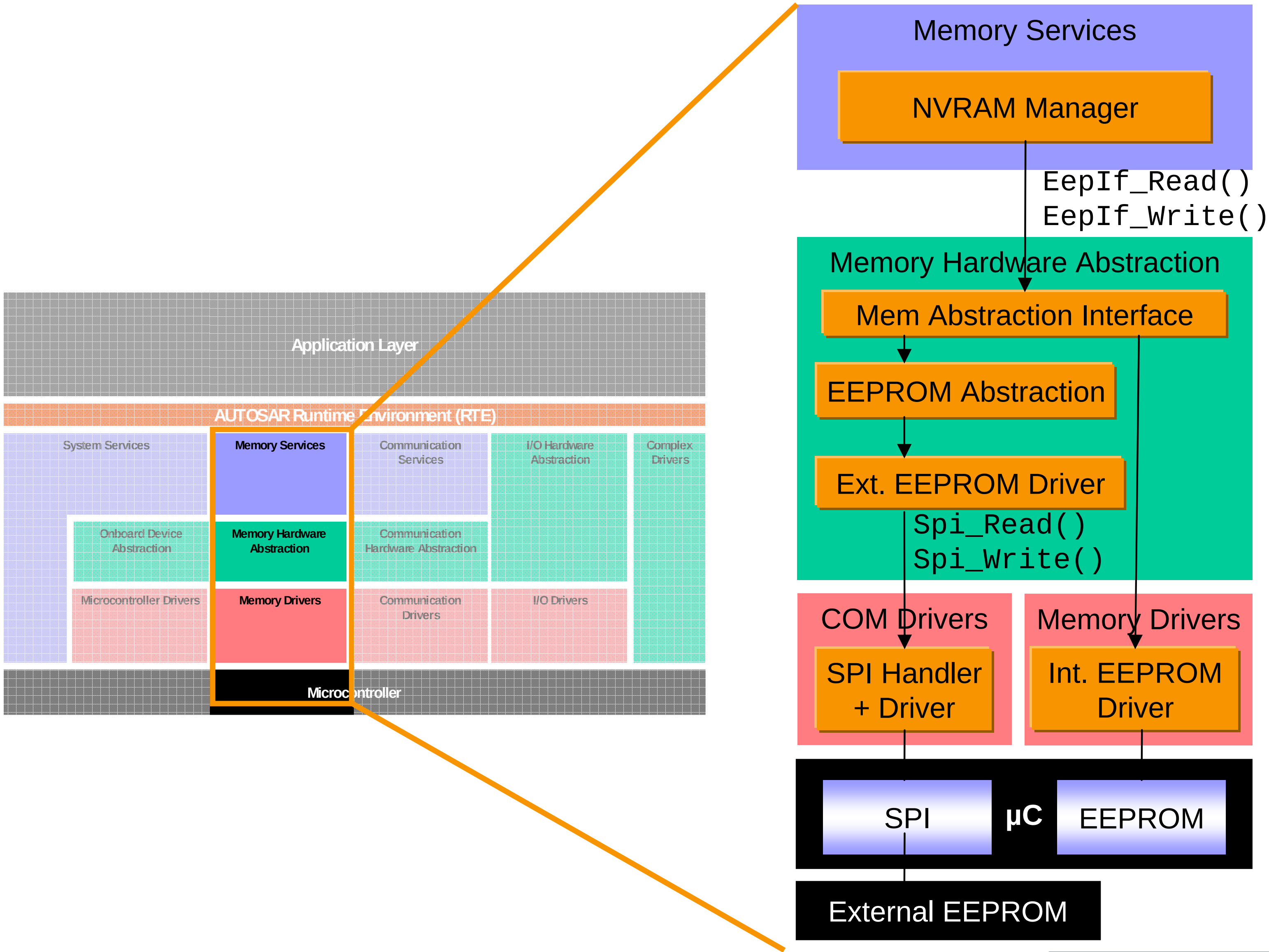


AUTOSAR Basic Software Modules



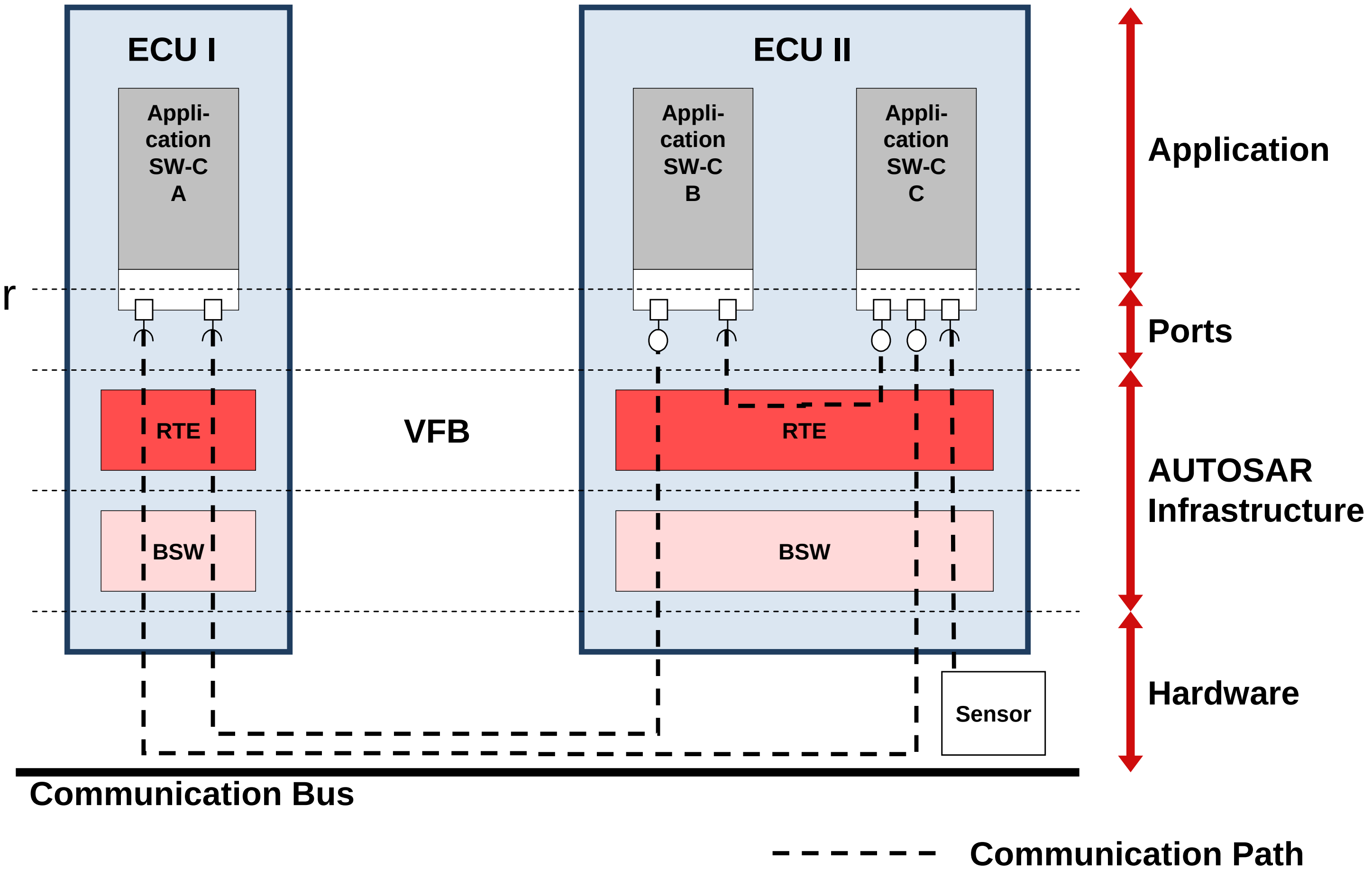
Example: "NVRAM Manager" ensures the storage and maintenance of non-volatile data and is independent of the design of the ECU.

Example



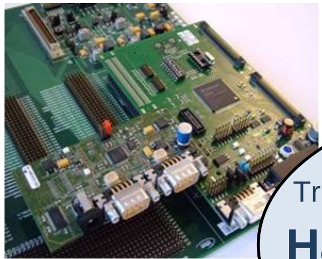
Intra- and Inter-ECU Communication

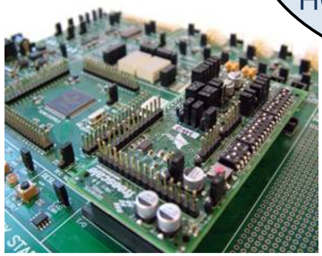
- Ports implement the interface according to the communication paradigm (here client-server based).
- Ports are the interaction points of a component.
- The communication is channeled via the RTE.
- The communication layer in the basic software is encapsulated and not visible at the application layer.



Validation of AUTOSAR Release 2.0

infineon





freescale™ semiconductor

AUTOSAR Specifications

AUTOSAR Concepts & Methodology

TriCore 1766 (32 Bit)

Hardware Platforms

HCS12X (16 Bit)

Validator 2 dealt with …

Integration Resource & Consumption Measurement

MBtech Mercedes-Benz technology

TRIALOG

Module Implementations & ECU Configuration Tools

BOSCH Invented for life

DECOMSYS Your FlexRay Development Partner

vector

tNi SOFTWARE

EB

freescale™ semiconductor

Infosys

Mecel

softing your connection to excellence

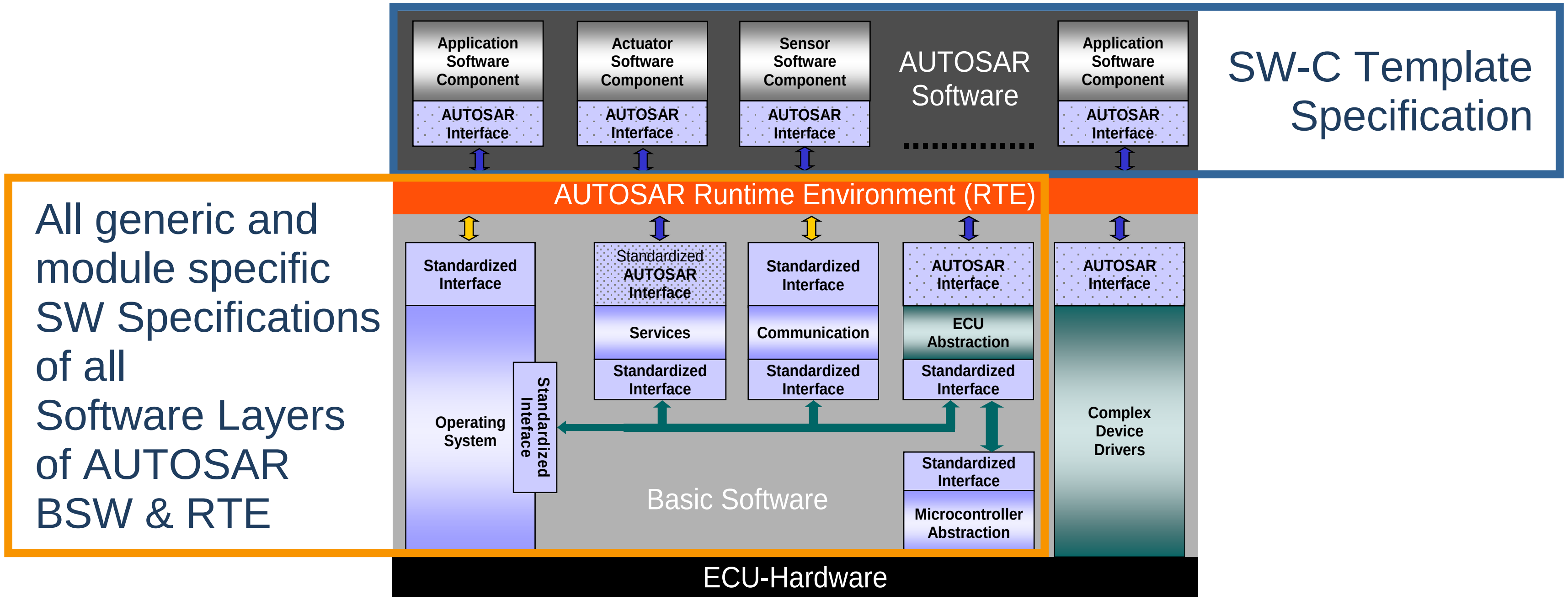
ETAS

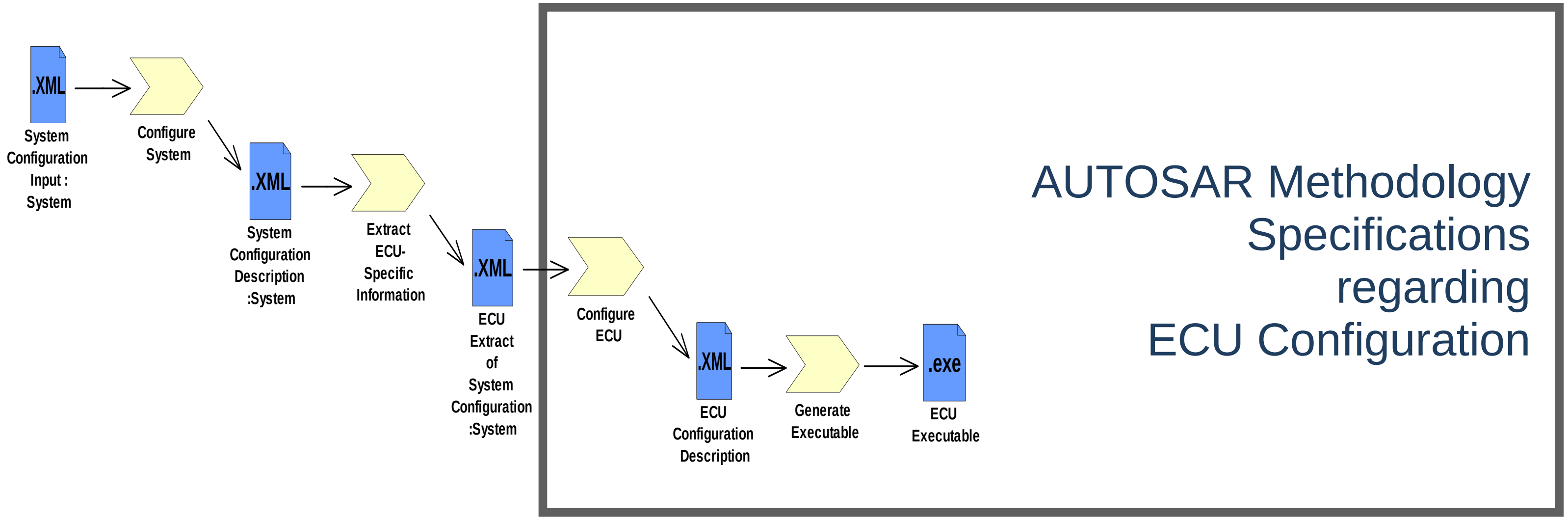
infineon

KPIT Cummins Infosystems Limited

Mentor Graphics

Used Release 2.0 AUTOSAR specifications

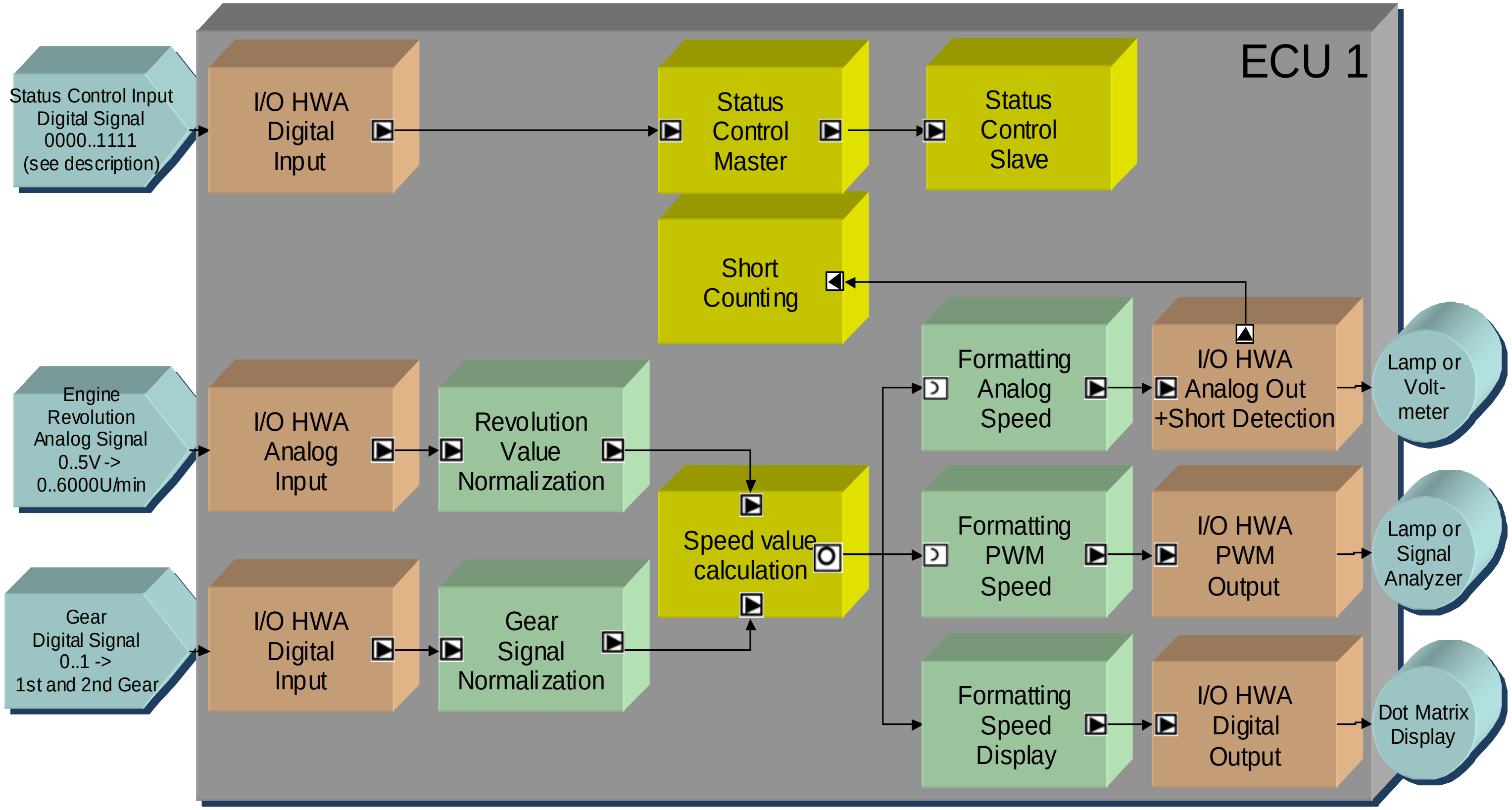




Validation of Standardized SW Specifications: Functionality & Scalability

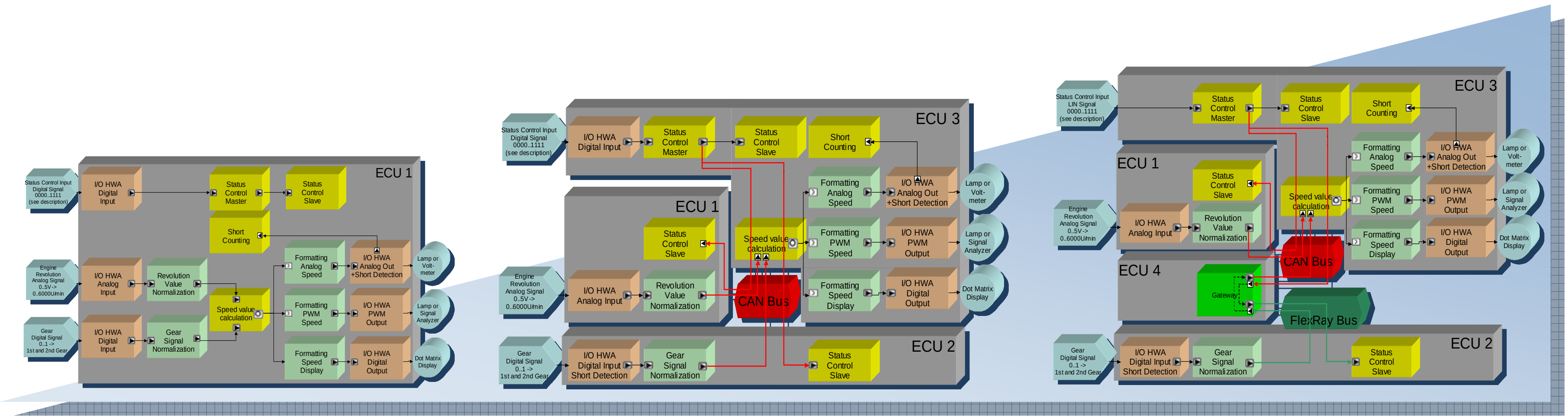
The specified application provides 'realistic' functionality:

- Calculating of the vehicle speed based on several inputs
- Displaying the calculated speed



System Test Approach: Functionality & Scalability

- **Scalability is divided into 3 aspects:**
  - Distribution of the given application on several nodes.
  - Using the appropriate communication bus technology.
  - Using the appropriate platform for each node.

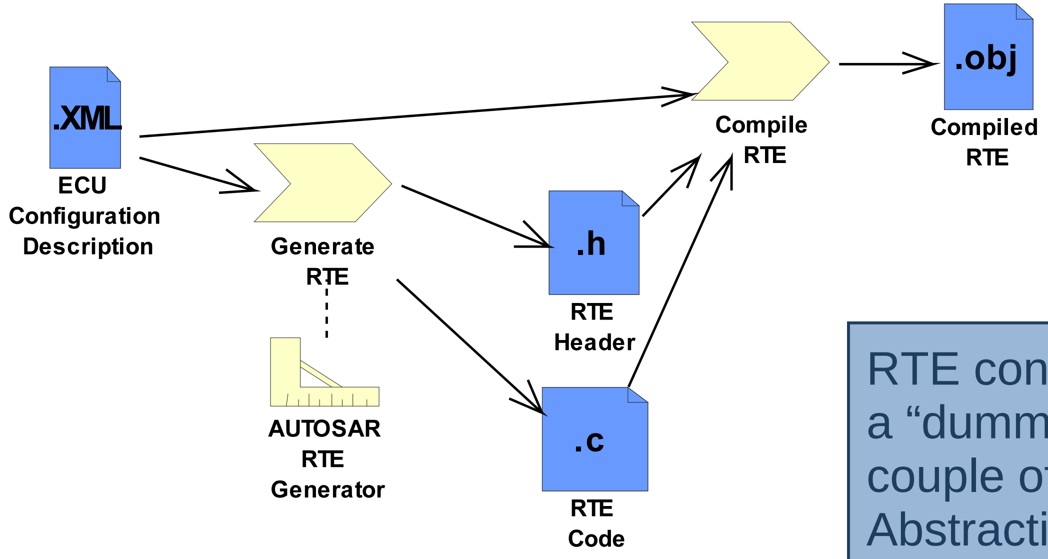


One ECU

3 ECU connected by CAN bus

4 ECU connected by CAN and FlexRay bus

Experience with AUTOSAR concepts and methodology: RTE



RTE concept was validated in the system test where a “dummy” real world application was created with a couple of AUTOSAR SW-C’s and IO Hardware Abstraction.

RTE overhead = low!

Lessons learned:

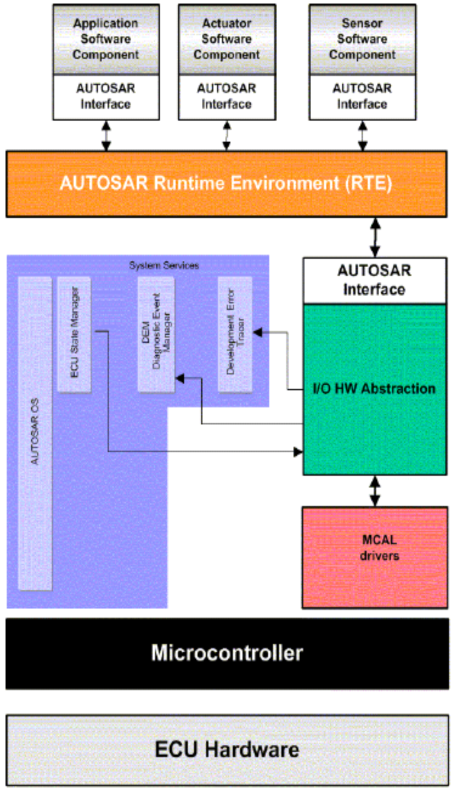
➔ Configuration of RTE might be very complex as long the requirements of the RTE and the OS are not optimized.

➔ Close linkage of RTE & OS requires close cooperation between implementers

Experience with AUTOSAR concepts and methodology: IO Hardware Abstraction

The IO HW Abstraction is a special kind of AUTOSAR SW-C.

It enables the integration of SW-C which use IOs from the ECU (e.g. SW-C for sensors and actors).



- ➢ The IO HW abstraction SWS implemented in validator 2 was handled as a SW-C (AUTOSAR interface) and as a BSW module (interface to BSW Scheduler and IO driver).
- ➢ It was needed to specify the AUTOSAR interface of the IO HW Abstraction at the beginning of the project, since it is not defined in the SWS of the IO HW Abstraction.
- ➢ The definition of the AUTOSAR interface was done by defining ports for IO HW Abstraction and SW-Cs.

➔ Port interfaces instead of ports should be defined first.

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 50 modules has proven to be functional and complete.**

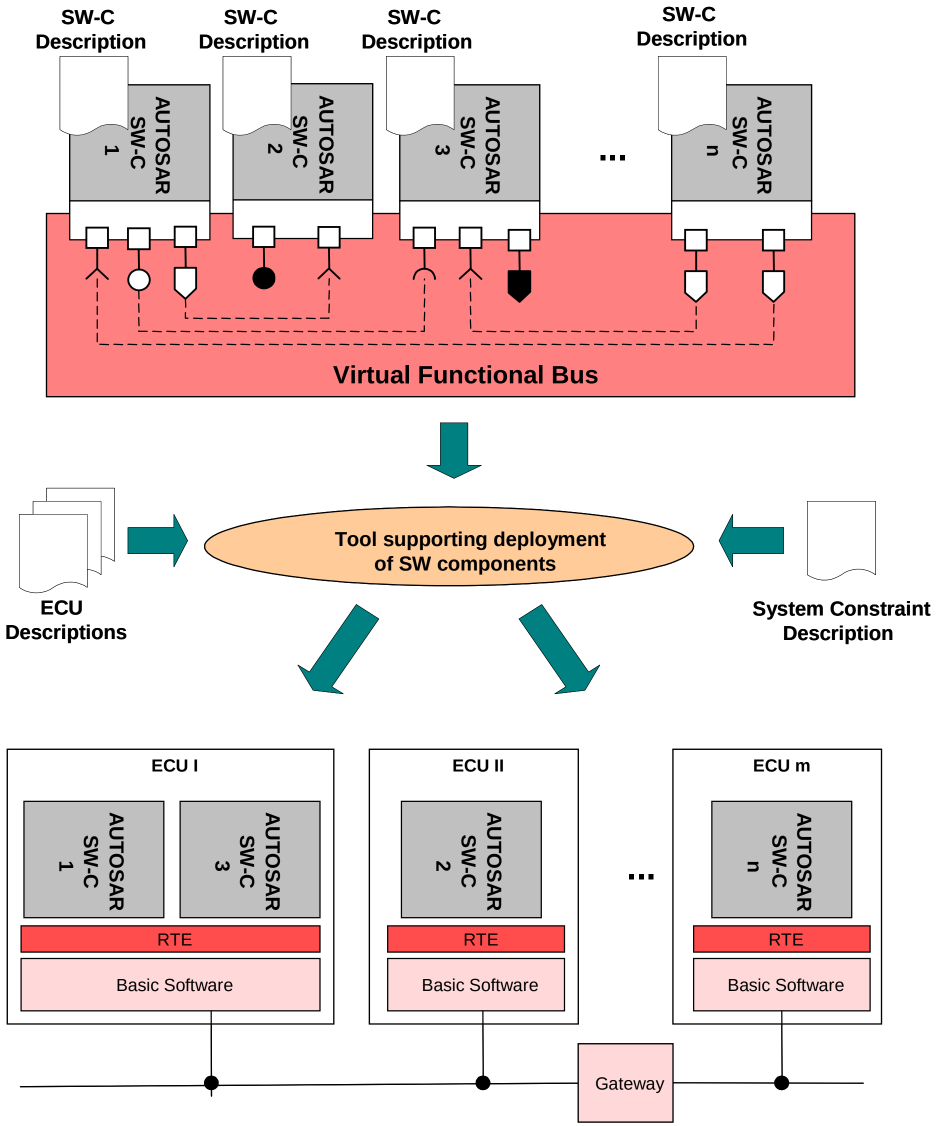
4. **The AUTOSAR 2.0 specifications for the modules of the layered architecture have been successfully implemented and integrated.**

5. **Conformance tests and processes are being prepared to ensure and to maintain a stable standard.**

Main Concepts: Methodology

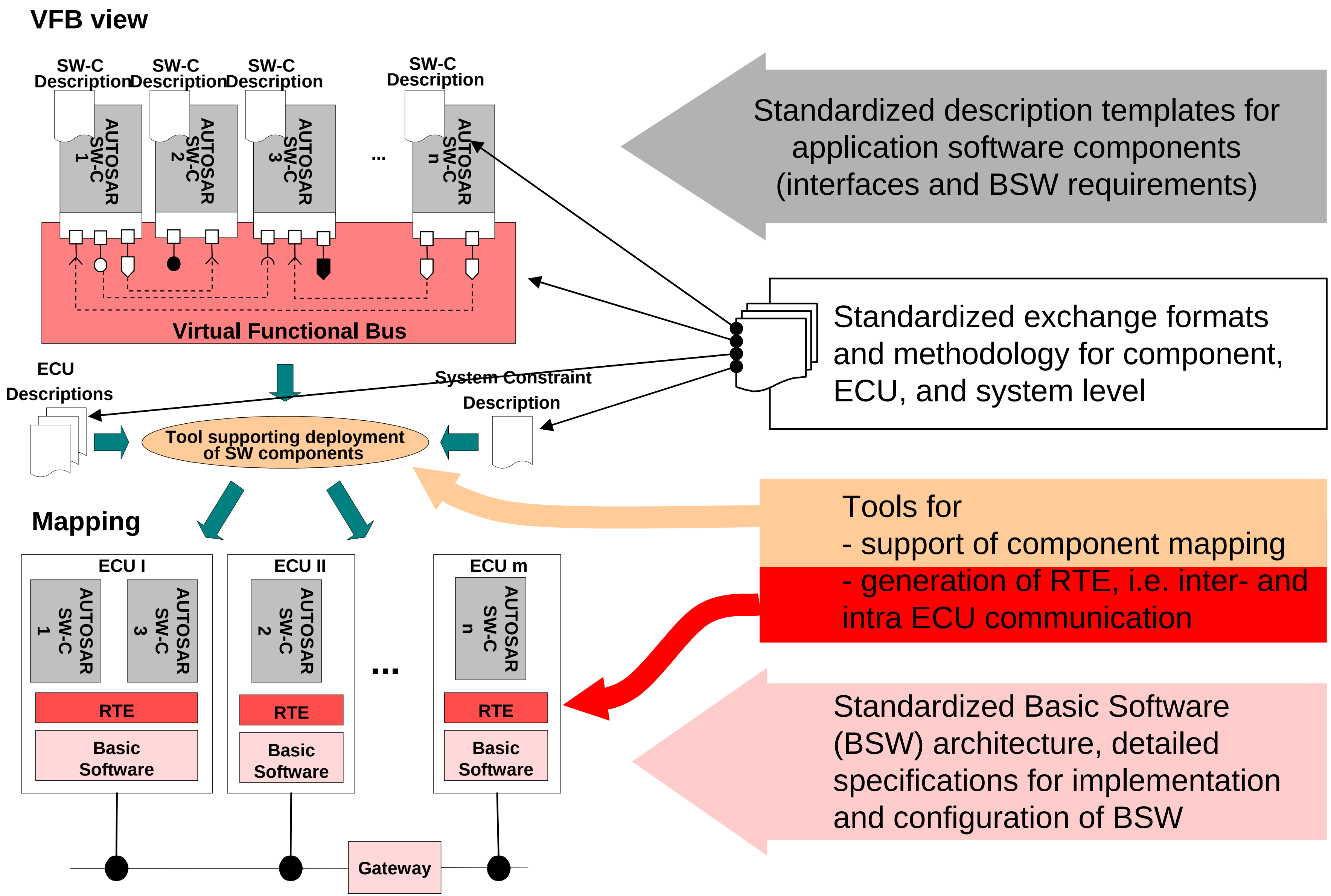
- Overall methodology
- Structure of configuration information
- System Design – Implementation Process
- Meta-model structure

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



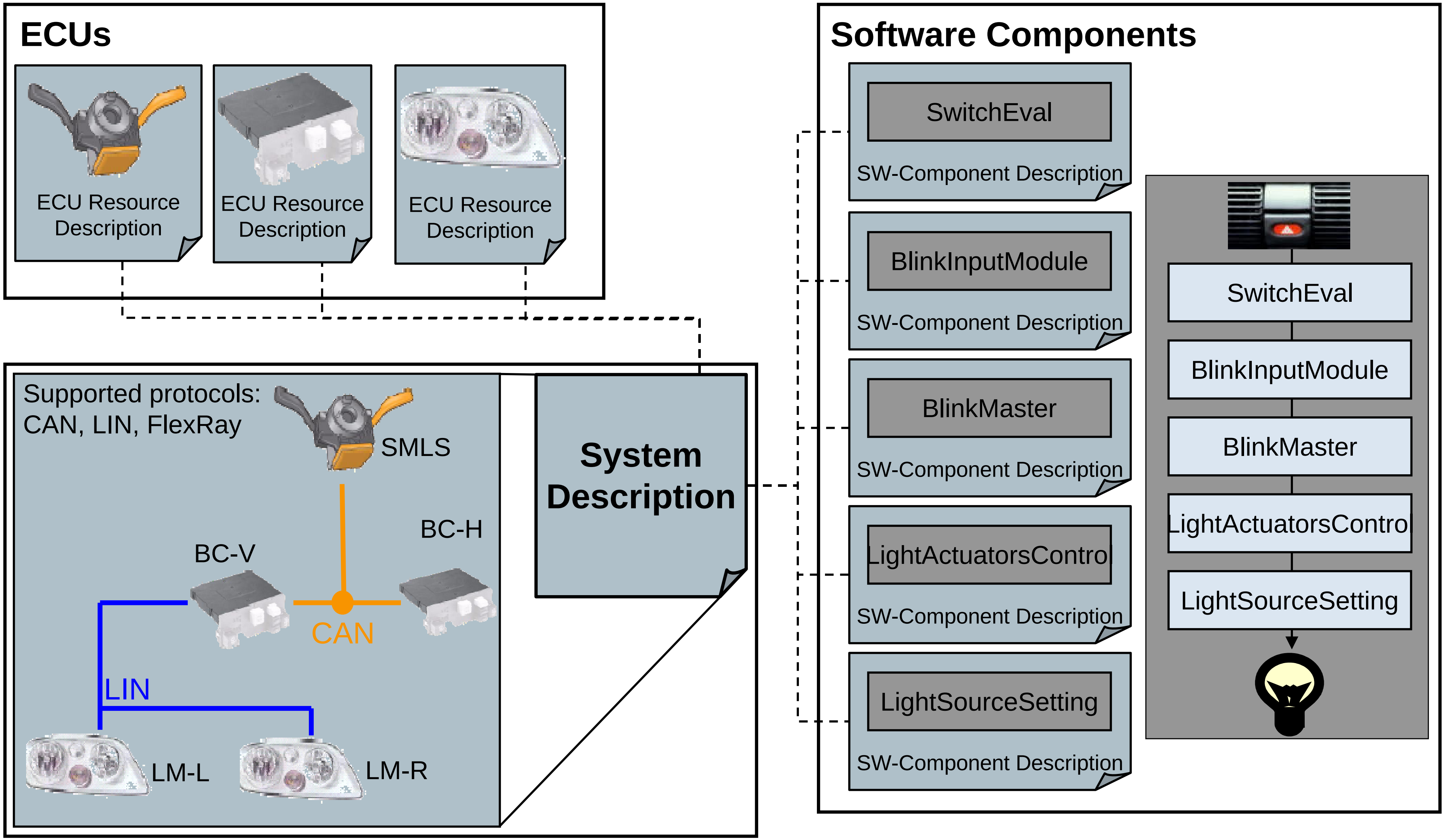
- Functional software is described formally in terms of "Software Components" (SW-C).
- Using "Software Component Descriptions" as input, the „Virtual Functional Bus" validates the interaction of all components and interfaces before software implementation.
- Mapping of "Software Components" to ECUs 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



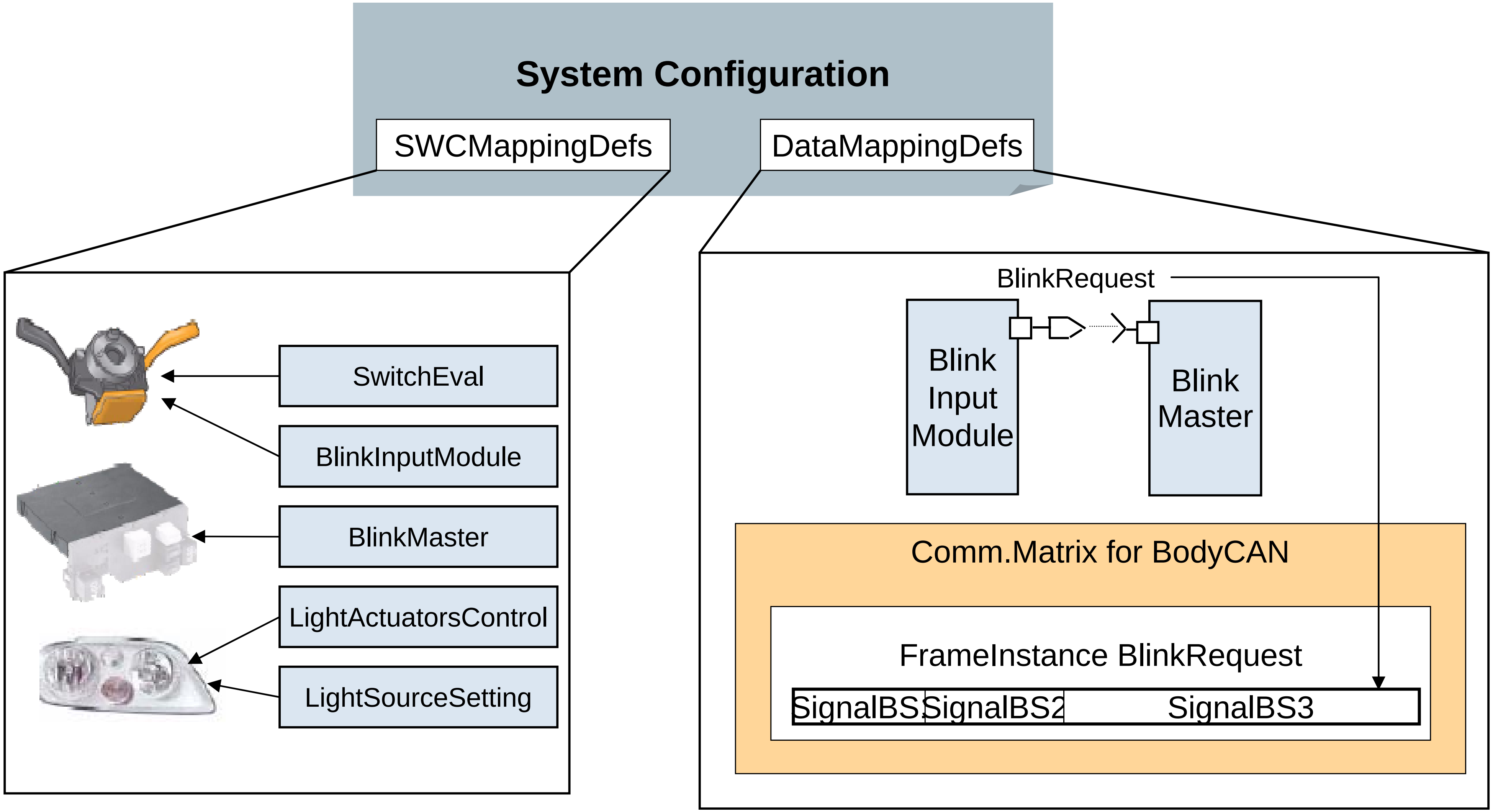
To configure the system, input descriptions of all software components, ECU resources and system constraints are necessary.

Example

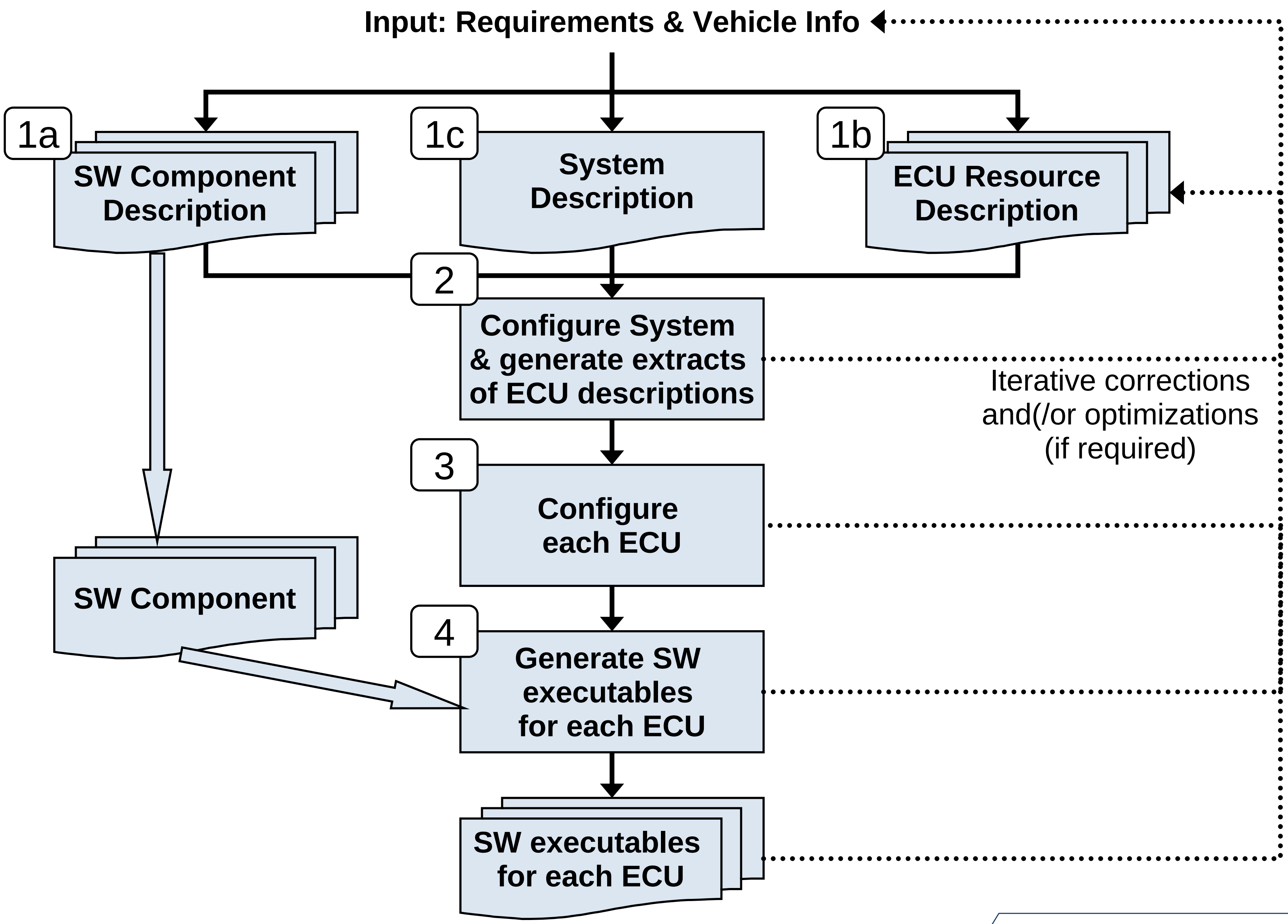


The system configuration maps software components to ECUs and links interface connections to bus signals.

Example

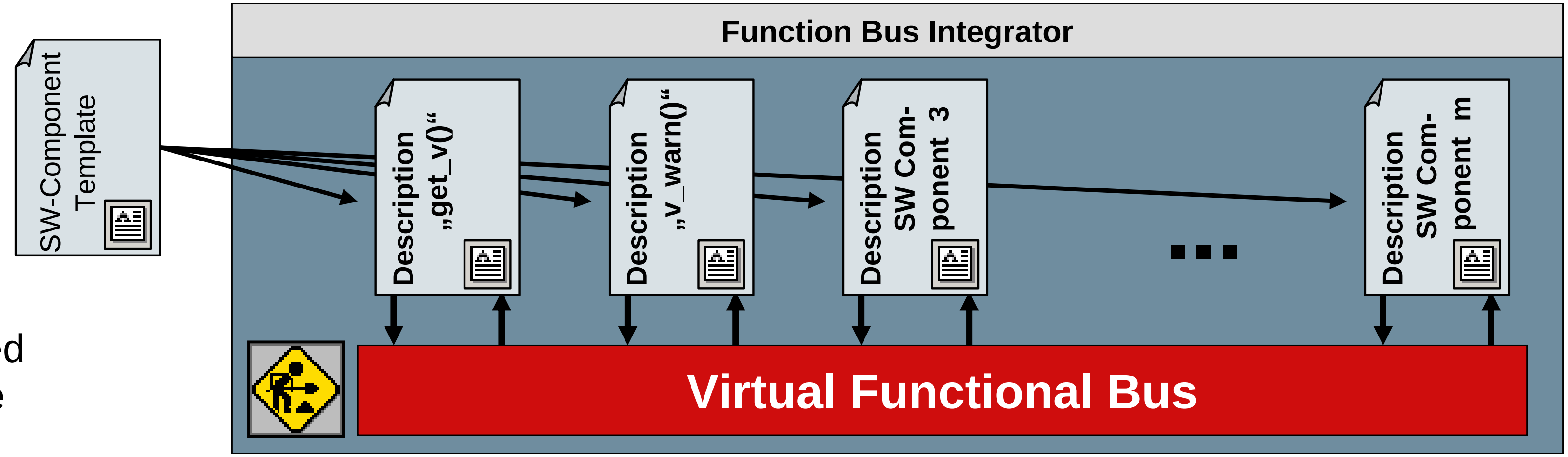


AUTOSAR – System Design – Implementation Process



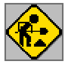
AUTOSAR – The Virtual Functional Bus

Input to the System Design on an abstract level

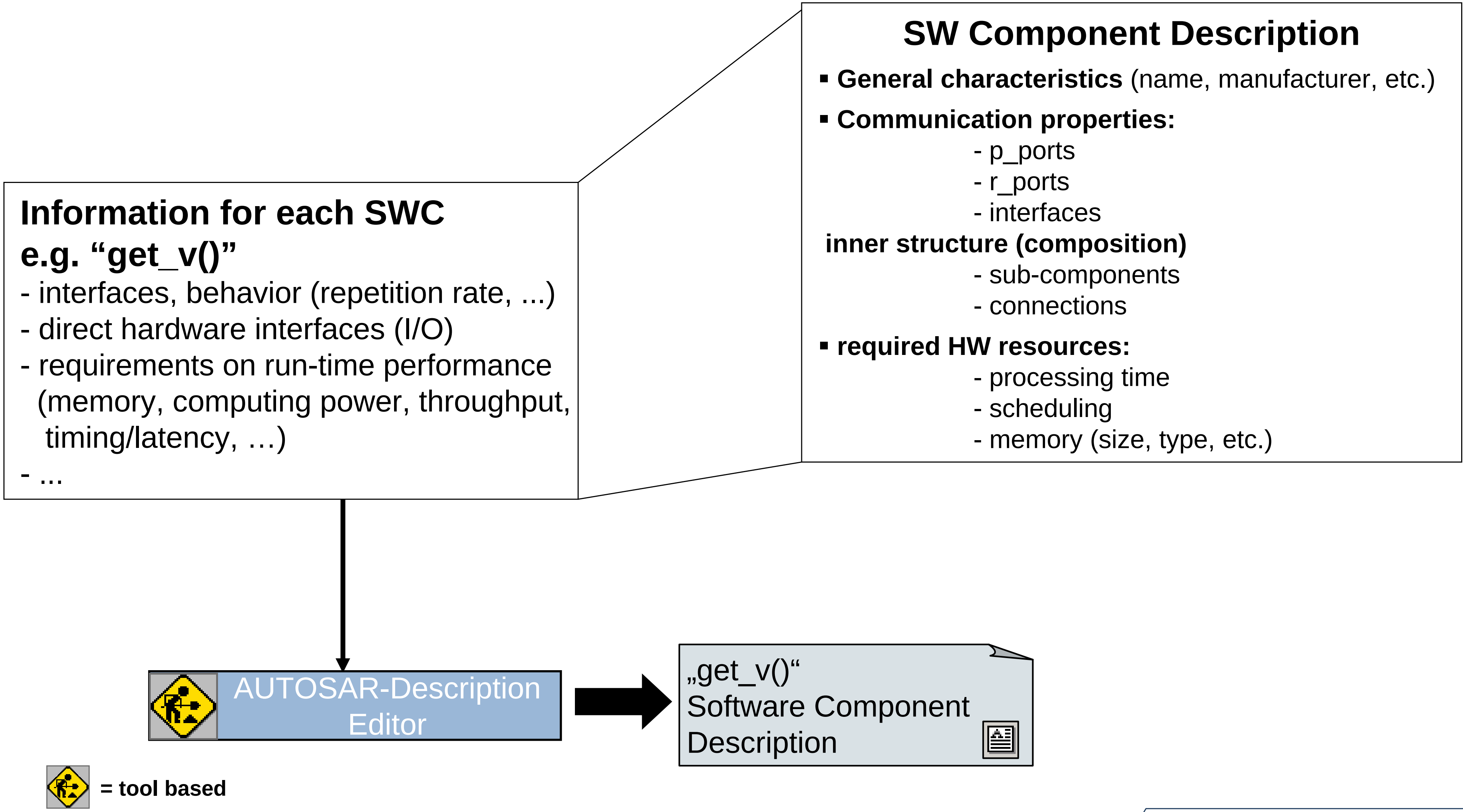


Example: speed warning device

- 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



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

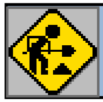
AUTOSAR – Input Descriptions (3 of 3)
Step 1c): Description of system

System 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**

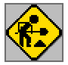
System Information overall system

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



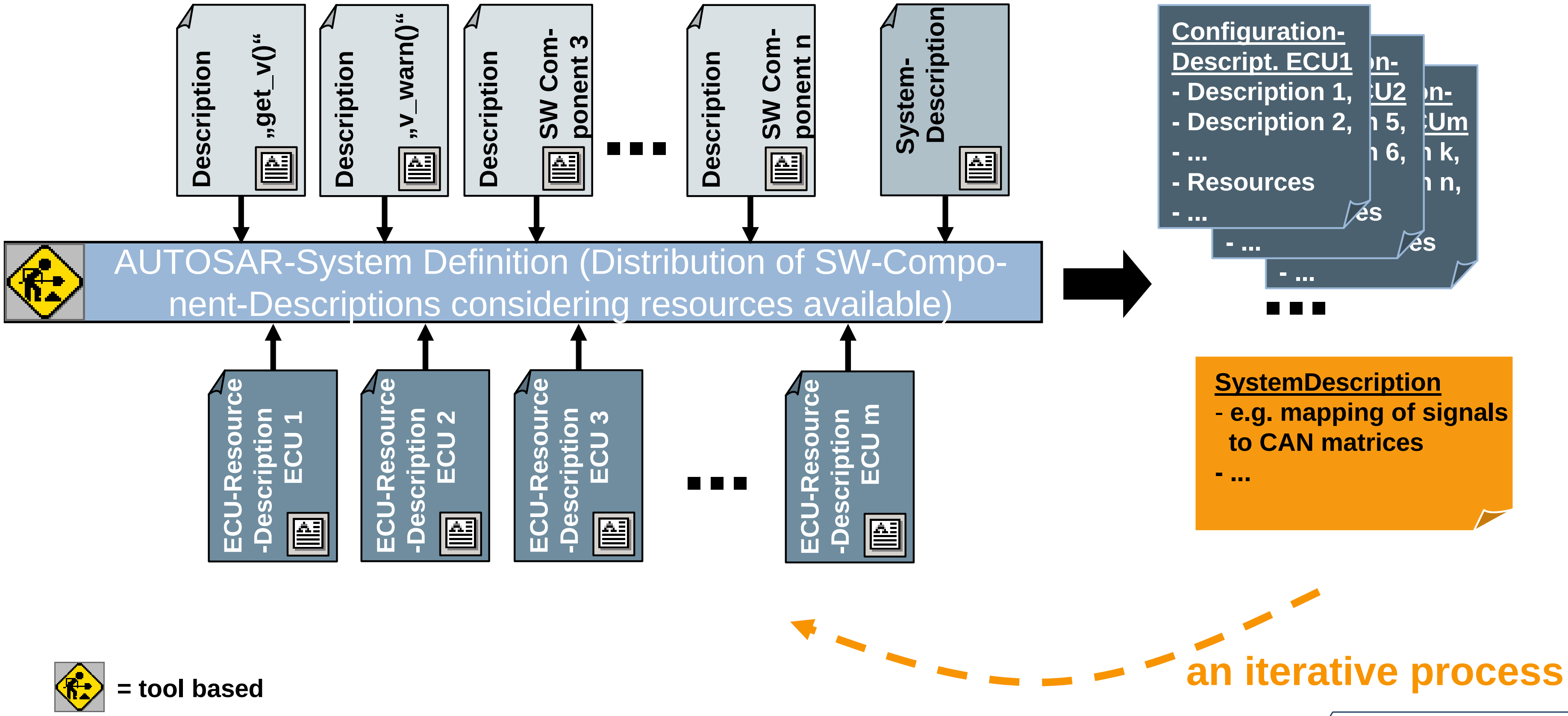
AUTOSAR-Description Editor

→ **System-Description**

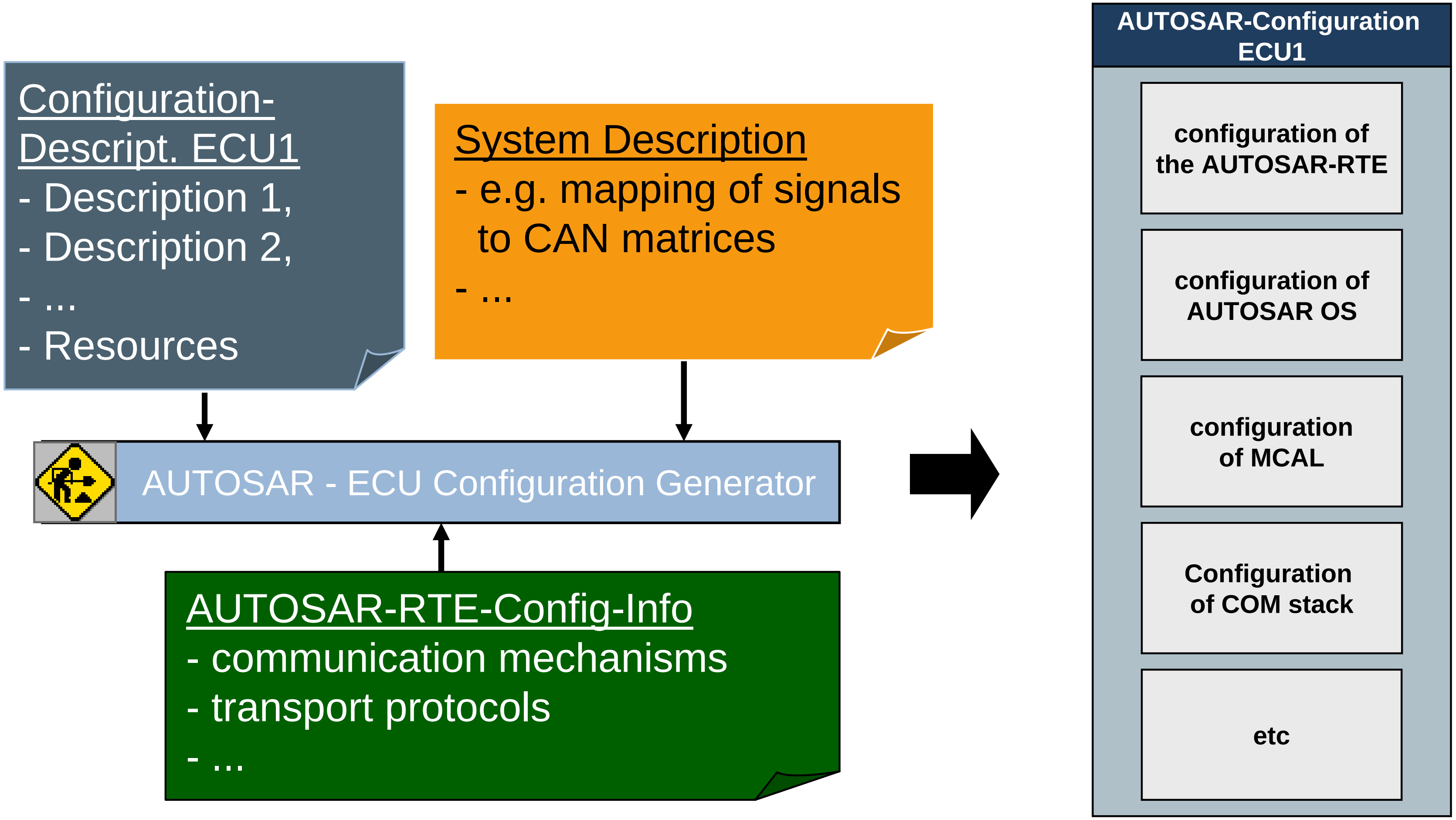
 **= tool based**

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

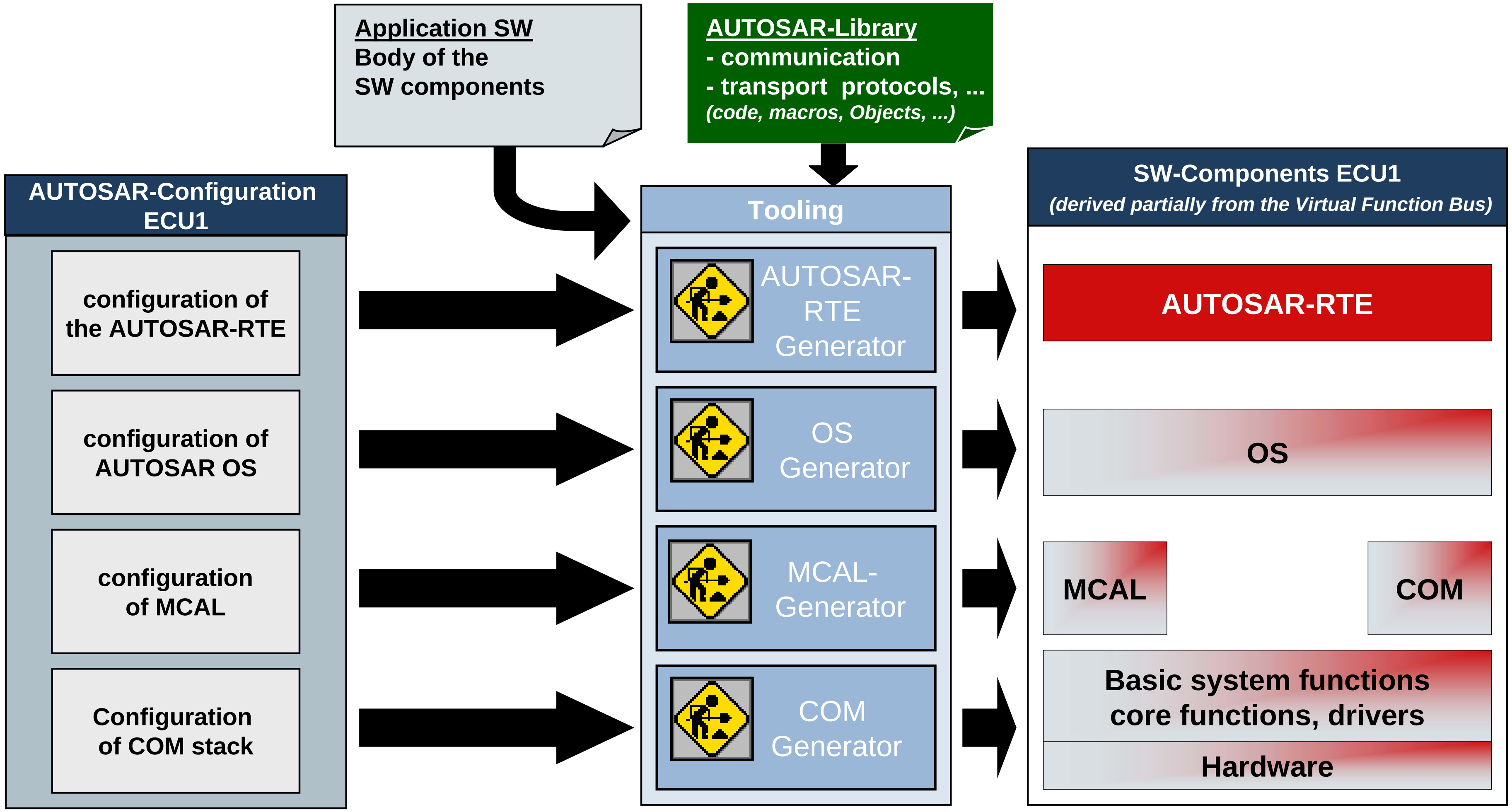


AUTOSAR – ECU-Configuration
Step 3: Generation of required configuration for AUTOSAR-Infrastructure per ECU



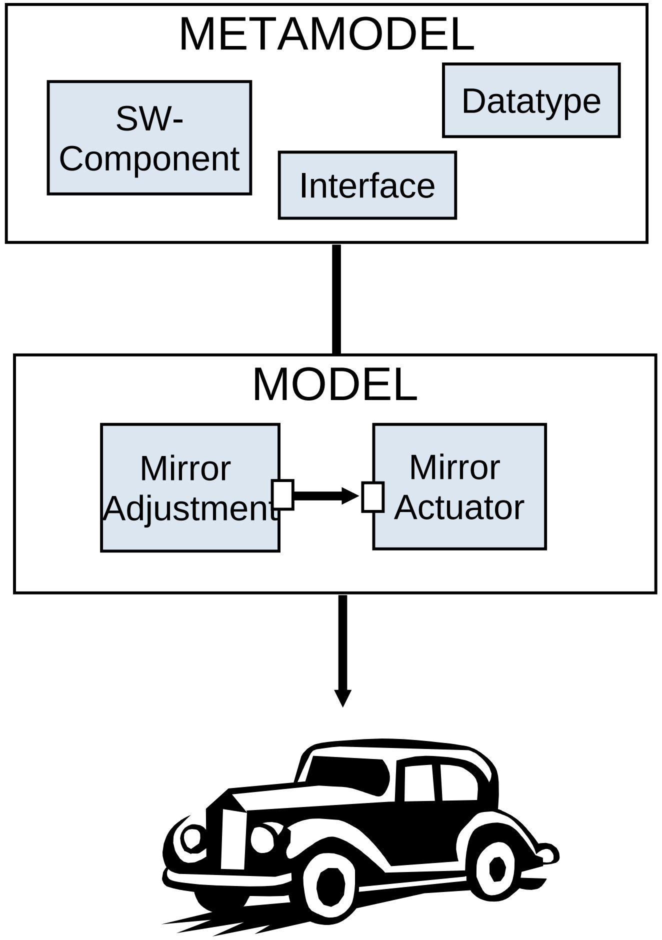
= tool based

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



AUTOSAR Metamodel
Formal description of all methodology related information

- The metamodel is modeled in UML
- The structure of the information can be clearly visualized
- The consistency of the information is guaranteed
- Using XML, a data exchange format can be generated automatically out of the metamodel



AUTOSAR Metamodel

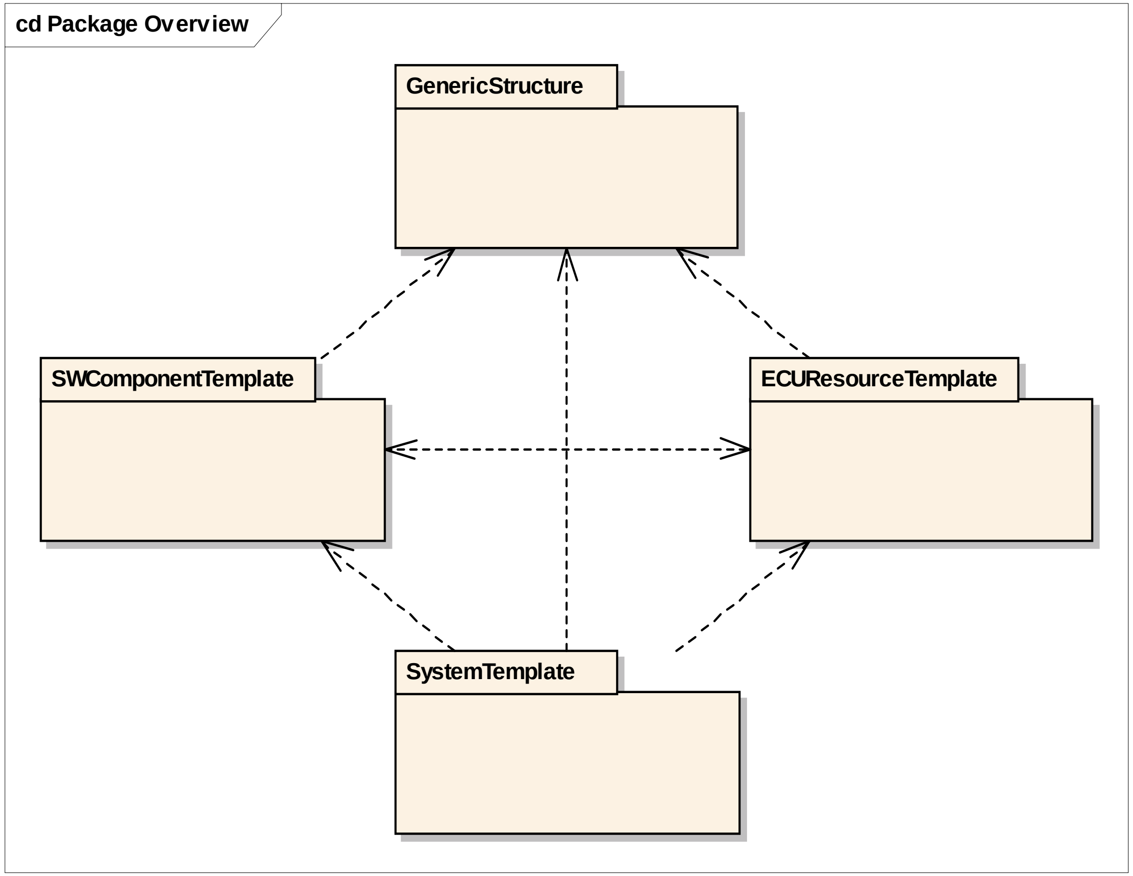
- The AUTOSAR Metamodel
  - is the backbone of the AUTOSAR architecture definition
  - contains complete specification, how to model AUTOSAR systems

M3: Model of the Metamodel
(Meta-Metamodel)
(Defines UML Modeling Elements)

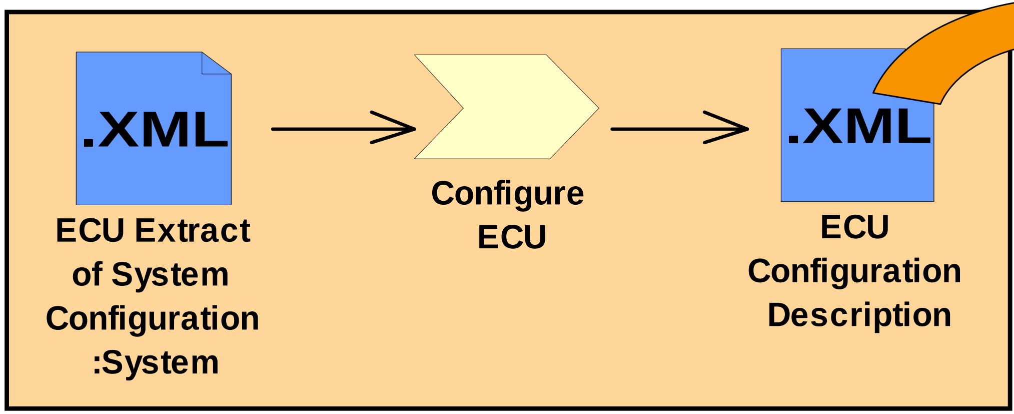
M2: Model of the model
(Metamodel)
(Defines AUTOSAR Modeling Elements)

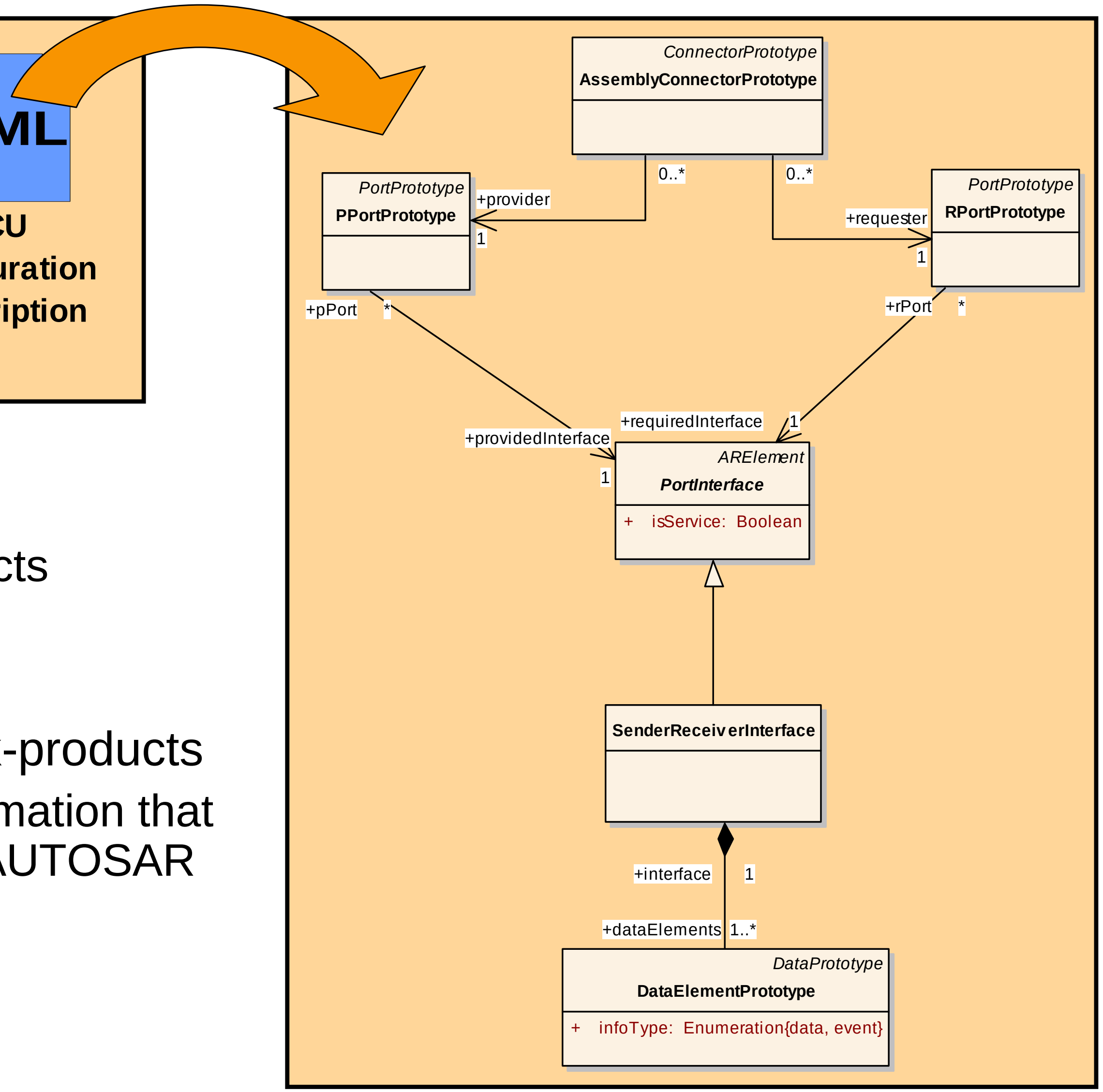
M1: Model of the system
(Defines a real system)

M0: Realized System in the car
(Implements a real system)



AUTOSAR Metamodel and Methodology





- Methodology
  - defines activities and work-products
  - is integrated in the metamodel

- Metamodel defines content of work-products
  - Formal description of all the information that is produced or consumed in the AUTOSAR methodology
  - Benefit of using the metamodel:
    - No inconsistencies
    - Easy maintenance
    - Consistent terminology

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.

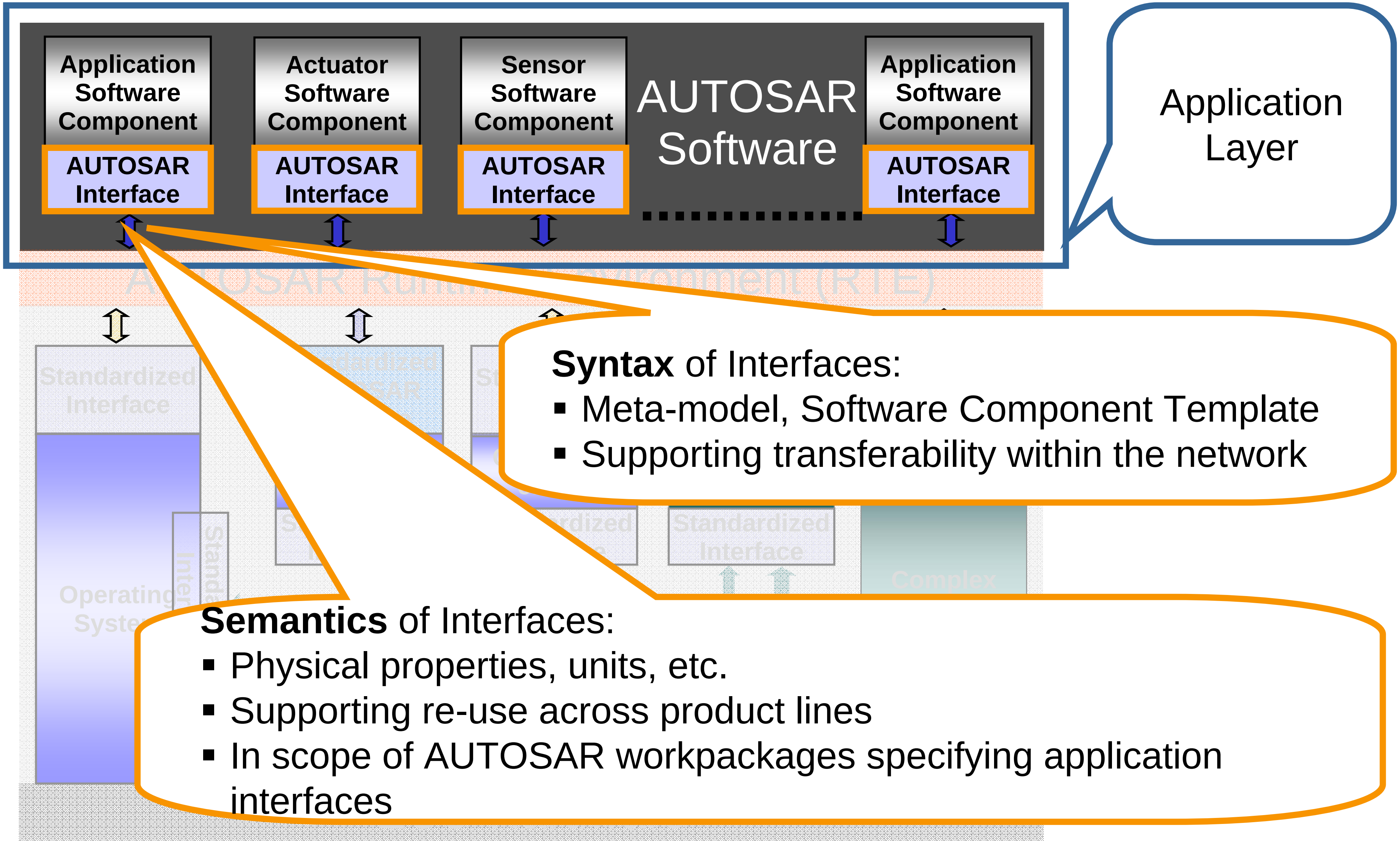
3. A methodology to integrate AUTOSAR software modules has been designed.

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

Main Concepts: Application Interfaces

- Standardization approach
- Current stage of standardization

AUTOSAR Application Interfaces

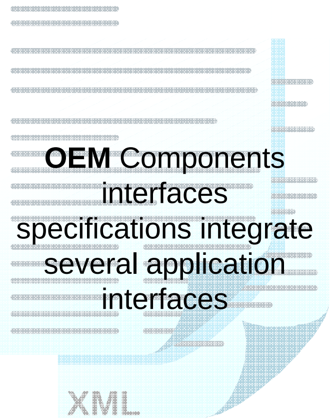


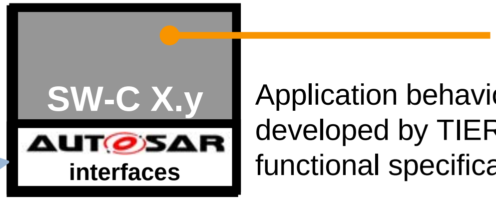
Syntax of Interfaces:
- Meta-model, Software Component Template
- Supporting transferability within the network

Semantics of Interfaces:
- Physical properties, units, etc.
- Supporting re-use across product lines
- In scope of AUTOSAR workpackages specifying application interfaces

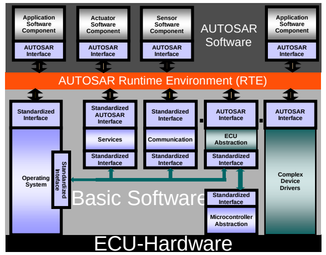
OEM Use case

- SHORT TERM: OEM is applying AUTOSAR Naming Convention more than 10.000 interfaces and calibrations data for industrial purposes after two years of intensive work on the specification of the naming convention
- Middle Term: Results are foreseen as an “AUTOSAR Application Interfaces Handbook” to support internal design & development of vehicle functions as much as support for exchange in project where suppliers are tied.

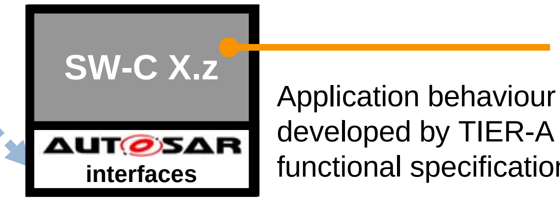




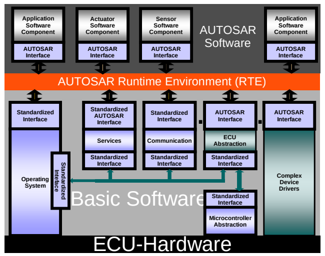
Application behaviour developed by TIER-B wrt OEM functional specifications



AUTOSAR software layered architecture integrated by **TIER- B**



Application behaviour developed by TIER-A wrt OEM functional specifications

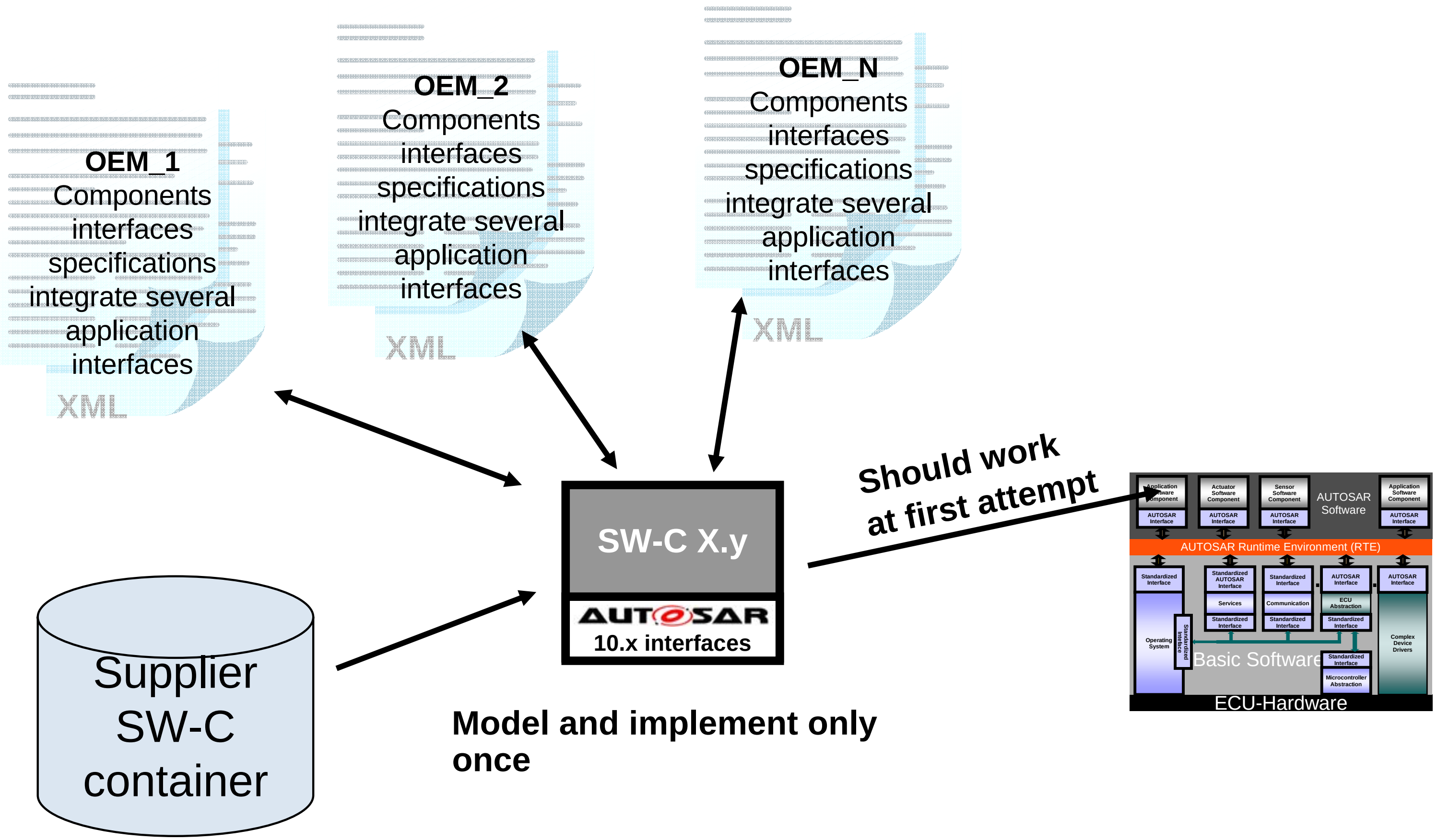


AUTOSAR software layered architecture integrated by **TIER-A**

Use of standardized application interfaces **increase quality on exchange** with suppliers and **improve software integration** from system standpoint.

Supplier Use case

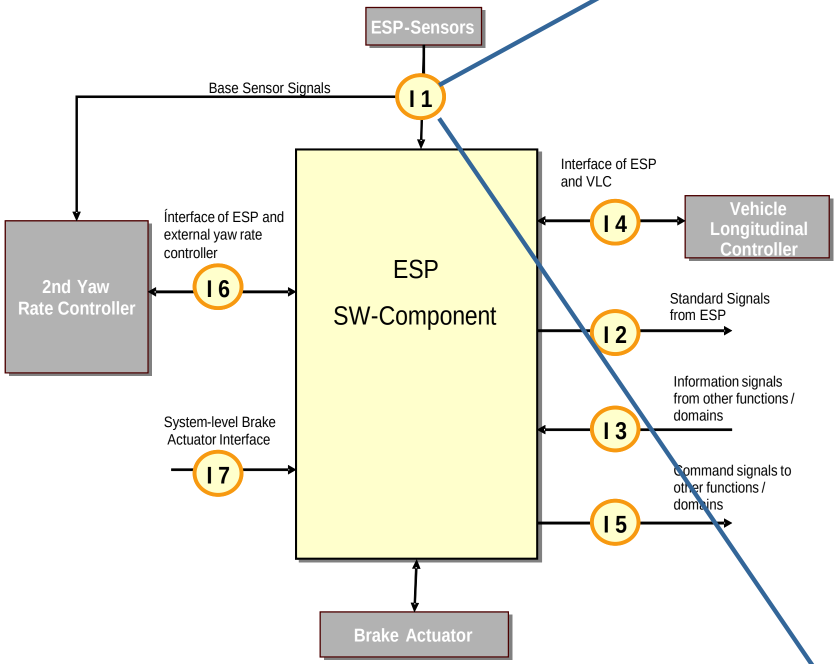
➢ Specification of application interfaces will support integration of SW-components.



Use of 10.x application interfaces increase **quality on integration**, i.e. they prevent from inconsistencies.

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

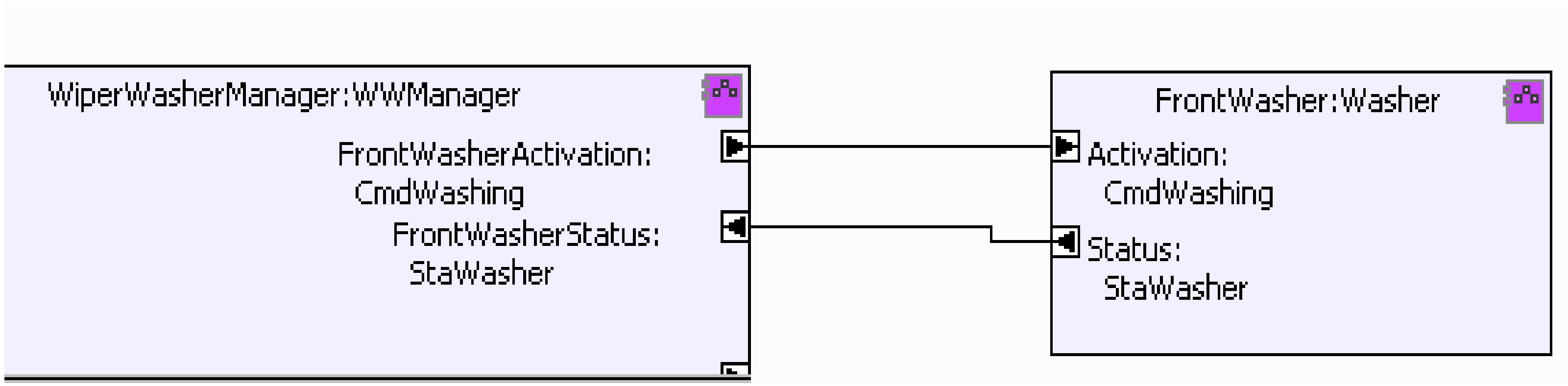
Example



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**

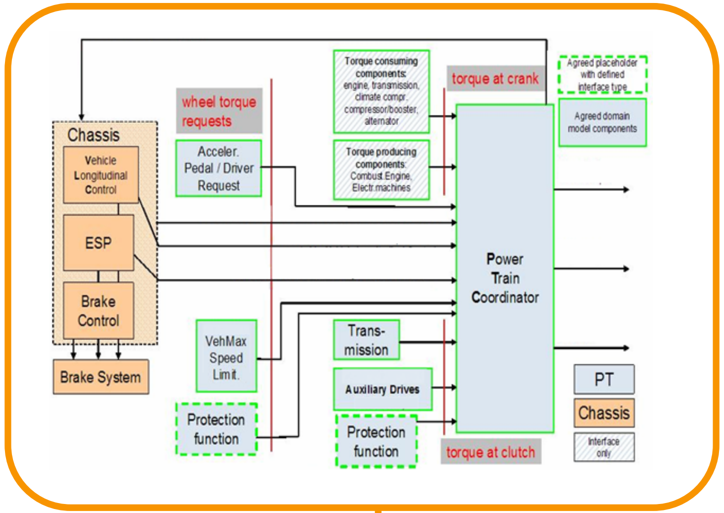
Standardized application interfaces on system level
(ESP-system, chassis domain)

Glance on Application Interfaces – Body Domain



- CmdWashing is the interface defined by following information:
  - It is provided by the WiperWasherManager component through the [Washer]Activation port
  - CmdWashing contains one data element command
  - Command is of type t_onoff
  - t_onoff is a RecordType, which describes a generic on/off information

AUTOSAR Application Interfaces Compositions under Consideration

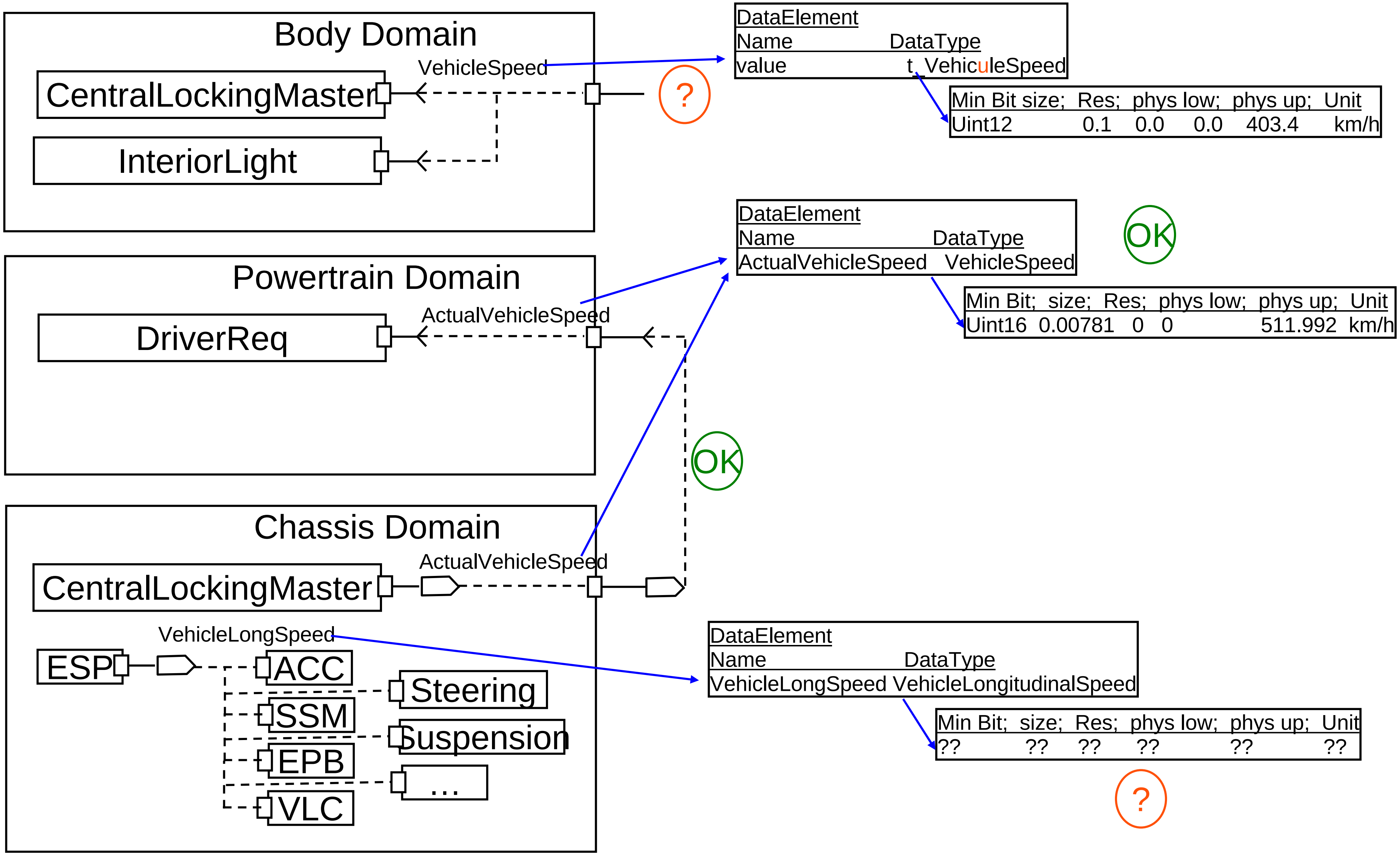


- **Body Domain**
  - Central Locking
  - Interior Light
  - Mirror Adjustment
  - Mirror Tinting
  - Seat Adjustment
  - Wiper/Washer
  - Anti Theft Warning System
  - Horn Control
  - Exterior Lights
  - Defrost Control
  - Seat climatization
  - Cabin climatization
  - Steering wheel climatization
  - Window Control
  - Sunroof/Convertible control
  - Steering column adjustment
  - Roller blind control

- **Chassis Control Domain**
  - Vehicle Longitudinal Control
  - Electronic Stability Program
  - Electronic Parking Brake
  - Adaptive Cruise Control
  - Roll Stability Control
  - Steering System
  - Suspension System
  - Stand Still Manager
  - High Level Steering
    - Vehicle Stability Steering
    - Driver Assistance Steering
  - All Wheel Drive/ Differential Lock

- **Powertrain Domain**
  - Powertrain Coordinator
  - Transmission System
  - Combustion Engine
    - Engine torque and mode management
    - Engine Speed And Position
    - Combustion Engine Misc.
  - Electric Machine
  - Vehicle Motion Powertrain
    - Driver Request
    - Accelerator Pedal Position
    - Safety Vehicle Speed Limitation

Major task: Conflict Resolution - Example Vehicle Speed



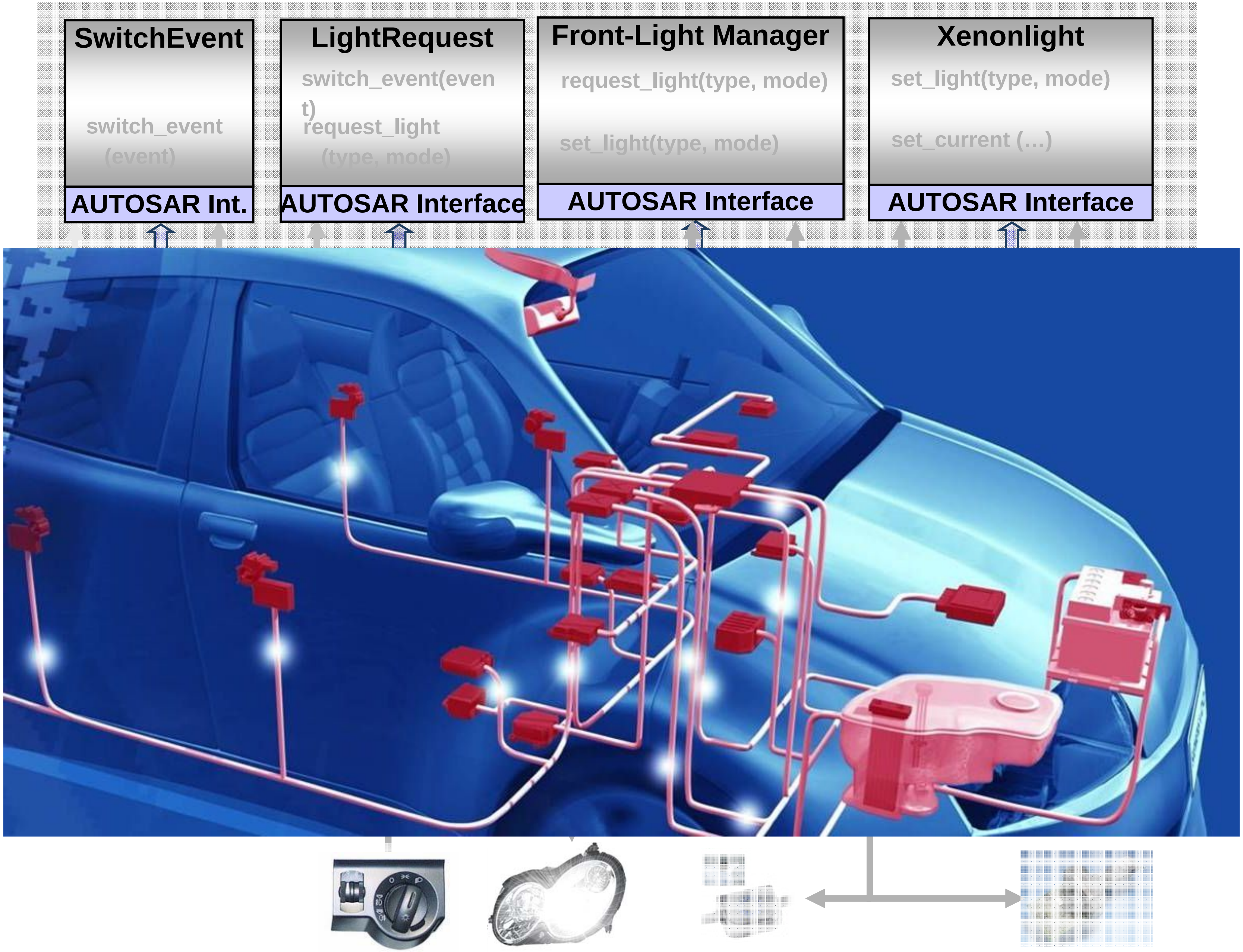
AUTOSAR Application Interfaces – Conclusion

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

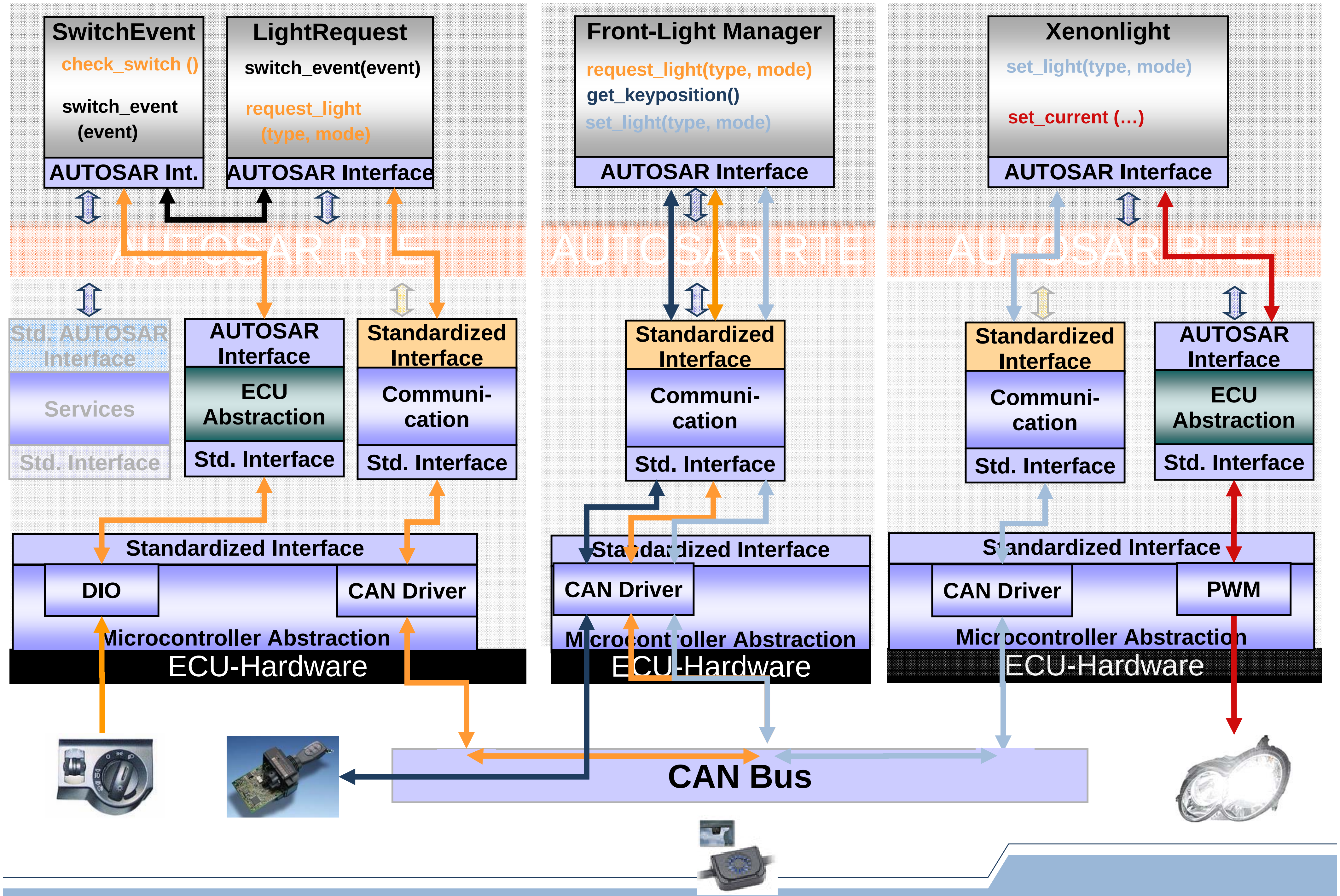
2. **It is a challenge to align standardization with the pace of application development.**

Wrap-up

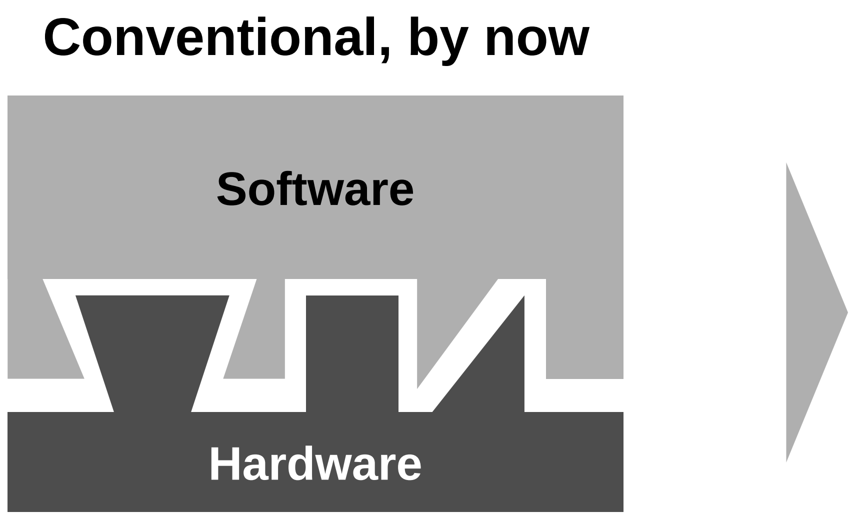
Distribution on ECUs

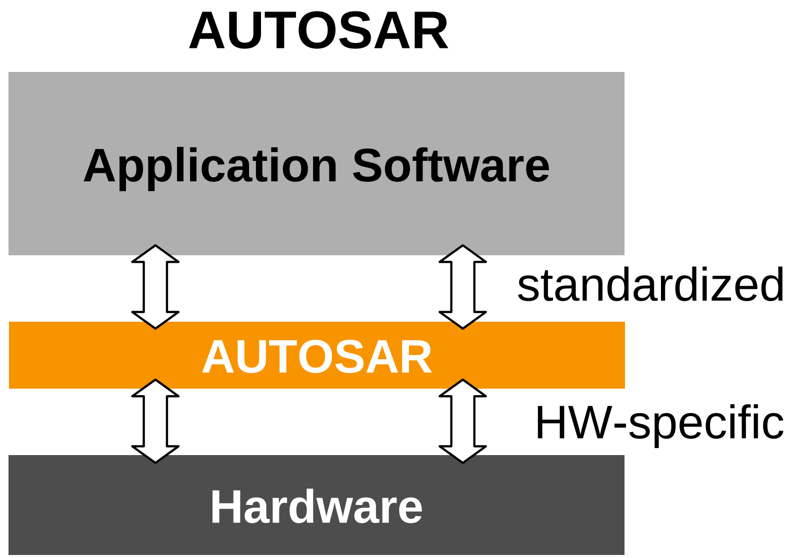


Use case ‘Front-Light Management’ in AUTOSAR



Automotive Software Development will change.

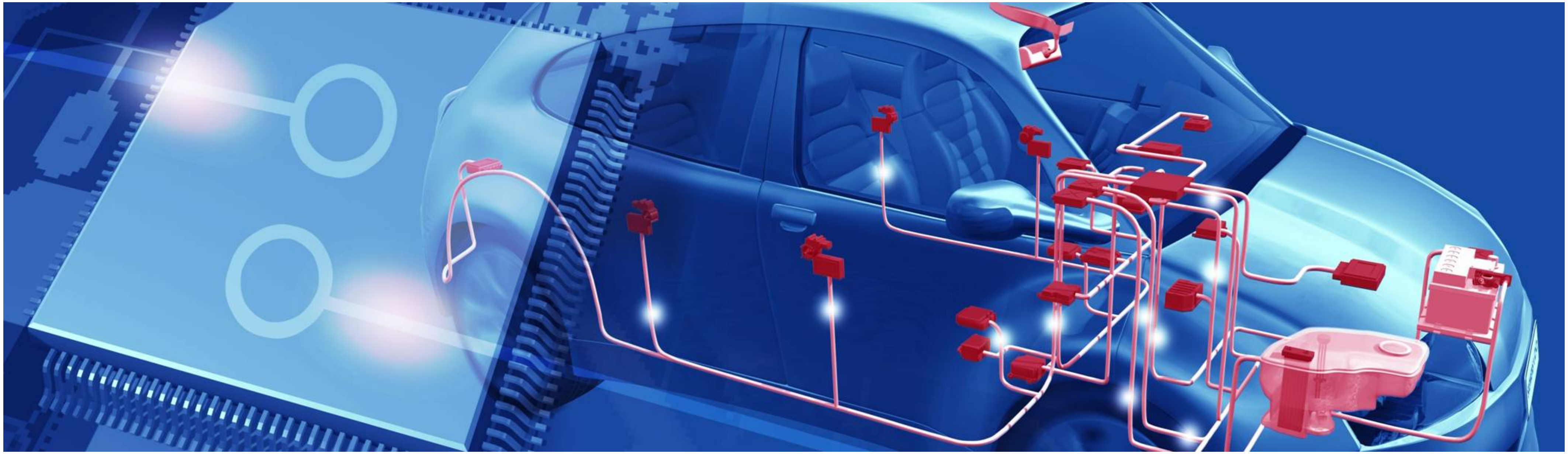




- **Hardware- and software** will be **widely independent** of each other.
- **Development processes** will be **simplified**.
  This **reduces development time** and **costs**.
- **Reuse** of **software increases** at OEM as well as at suppliers.
  This **enhances** also q**uality** and **efficiency**.

Automotive Software will become a product.

AUTOSAR Outlook



- AUTOSAR is ready to be used automotive product development
- Exploitation has already started
- AUTOSAR welcomes new members

- “Cooperate on standards, compete on implementation.”

Thank you for your attention!

http://www.autosar.org

request@autosar.org