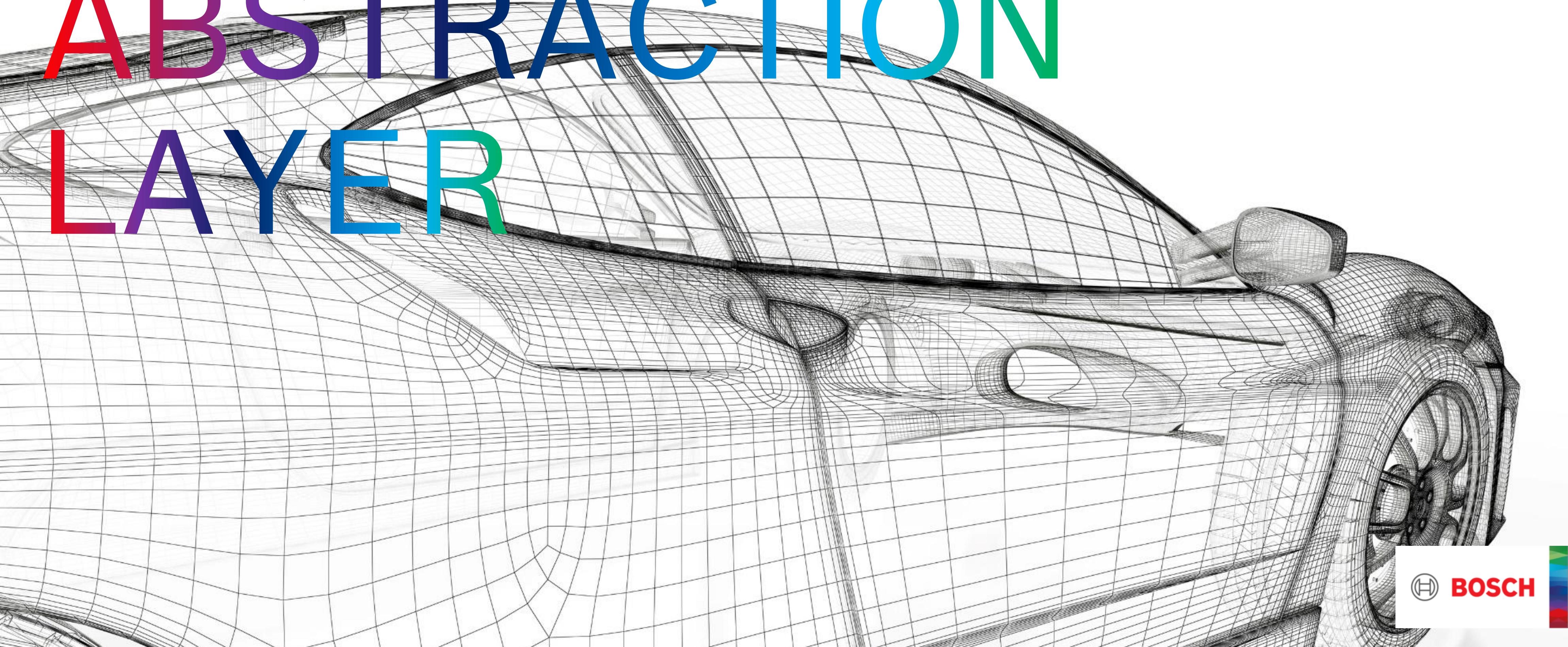
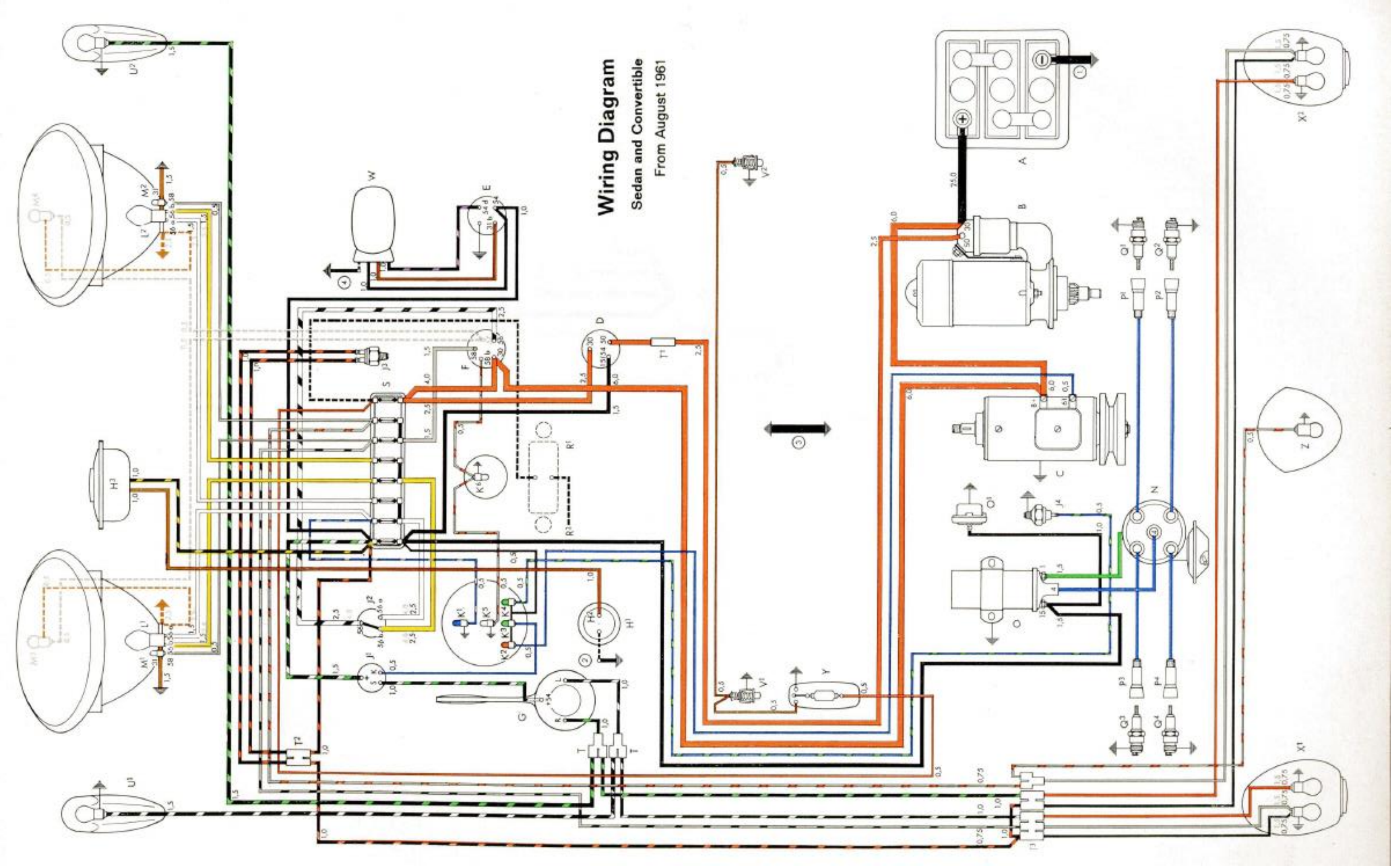
VEHICLE ABSTRACTION LAYER



BOSCH

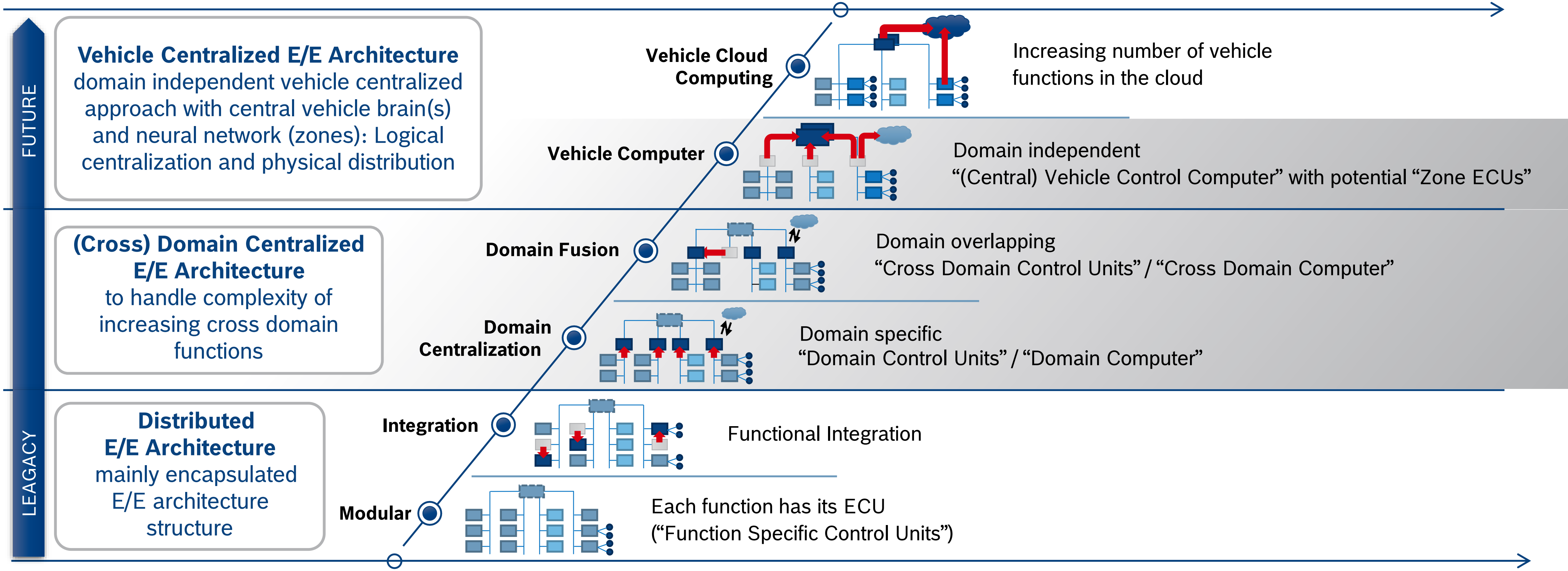
Vehicle Abstraction Layer
Automotive: 1962



Source: VW-Käferclub

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Trends for Future Mobility Systems
E/E Architecture Roadmap



FUTURE

Vehicle Centralized E/E Architecture
domain independent vehicle centralized approach with central vehicle brain(s) and neural network (zones): Logical centralization and physical distribution

(Cross) Domain Centralized E/E Architecture
to handle complexity of increasing cross domain functions

LEAGACY

Distributed E/E Architecture
mainly encapsulated E/E architecture structure

Vehicle Cloud Computing – Increasing number of vehicle functions in the cloud

Vehicle Computer – Domain independent "(Central) Vehicle Control Computer" with potential "Zone ECUs"

Domain Fusion – Domain overlapping "Cross Domain Control Units" / "Cross Domain Computer"

Domain Centralization – Domain specific "Domain Control Units" / "Domain Computer"

Integration – Functional Integration

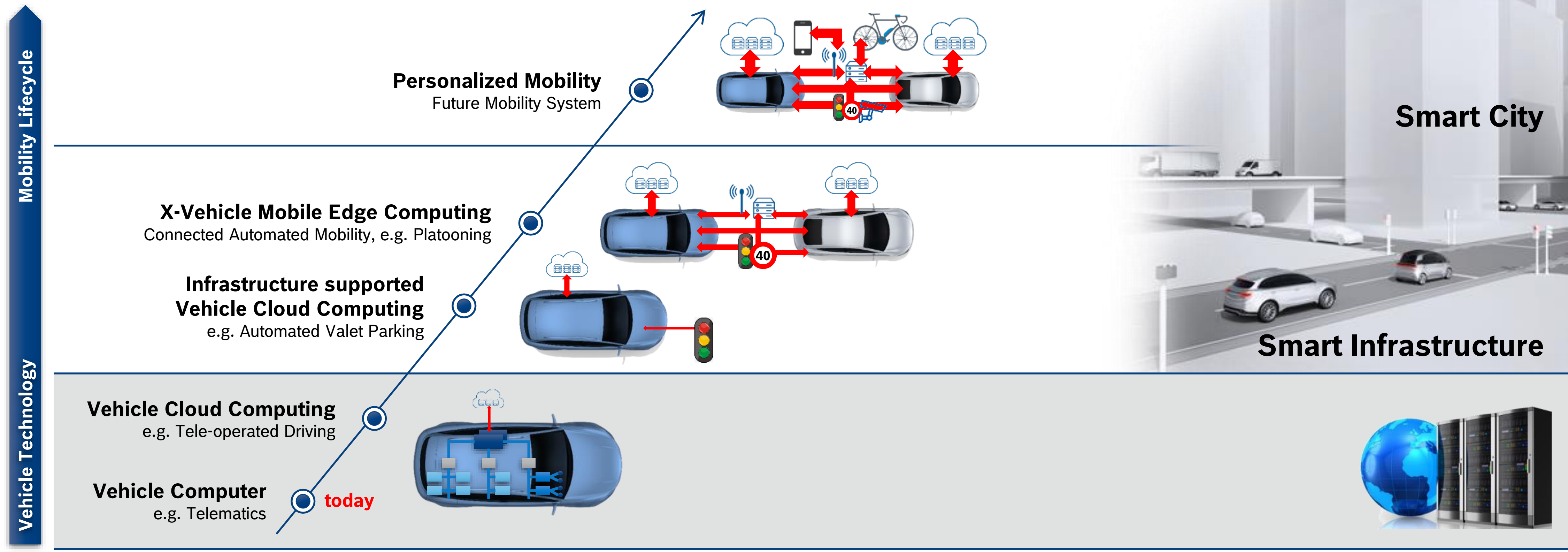
Modular – Each function has its ECU ("Function Specific Control Units")

increasing SW amount

▨ typ. state of the art automotive ECUs (function specific)	▨ Optional ECUs (e.g. Central Gateway)	⋖ Sensors/Actuators	ECU = Electronic Control Unit
■ Performance ECUs e.g. (Cross-)Domain Control Unit, (Cross-)Domain Computer, Vehicle Control Computer	■⋖ Domain independent Zone ECUs	▨ Domain specific Zone ECUs (e.g. todays Door ECU)	

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Trends for Future Mobility Systems
E/E Architecture Extension to Cloud Connectivity



Mobility Lifecycle

Vehicle Technology

Personalized Mobility
Future Mobility System

X-Vehicle Mobile Edge Computing
Connected Automated Mobility, e.g. Platooning

Infrastructure supported Vehicle Cloud Computing
e.g. Automated Valet Parking

Vehicle Cloud Computing
e.g. Tele-operated Driving

Vehicle Computer
e.g. Telematics

today

Smart City

Smart Infrastructure

State of the art ECU

Zone ECUs w/ sensors/actuators

Vehicle Computer

(Cross-)Domain ECU

Cloud

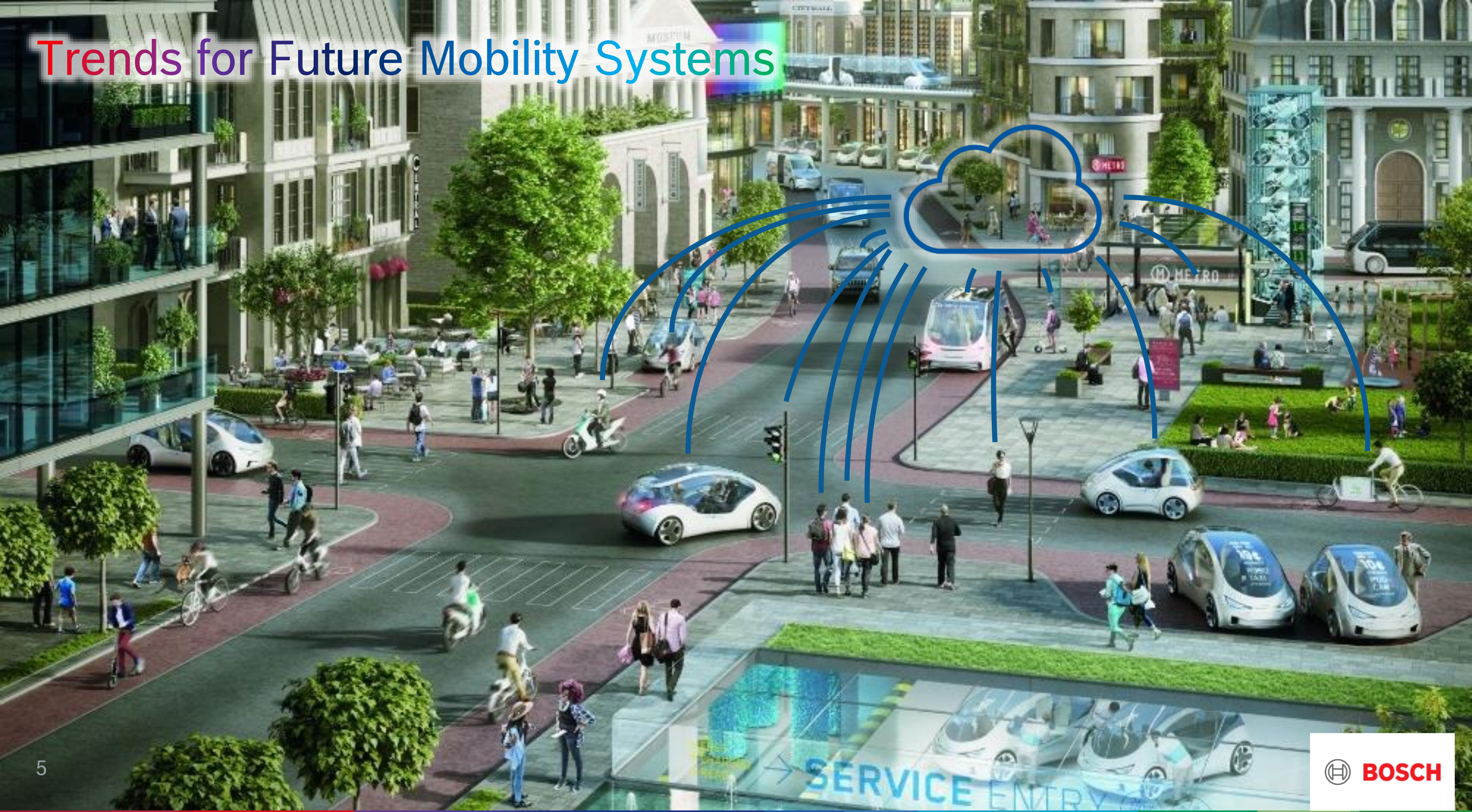
Infrastructure

Edge

Personal µMobility

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Trends for Future Mobility Systems

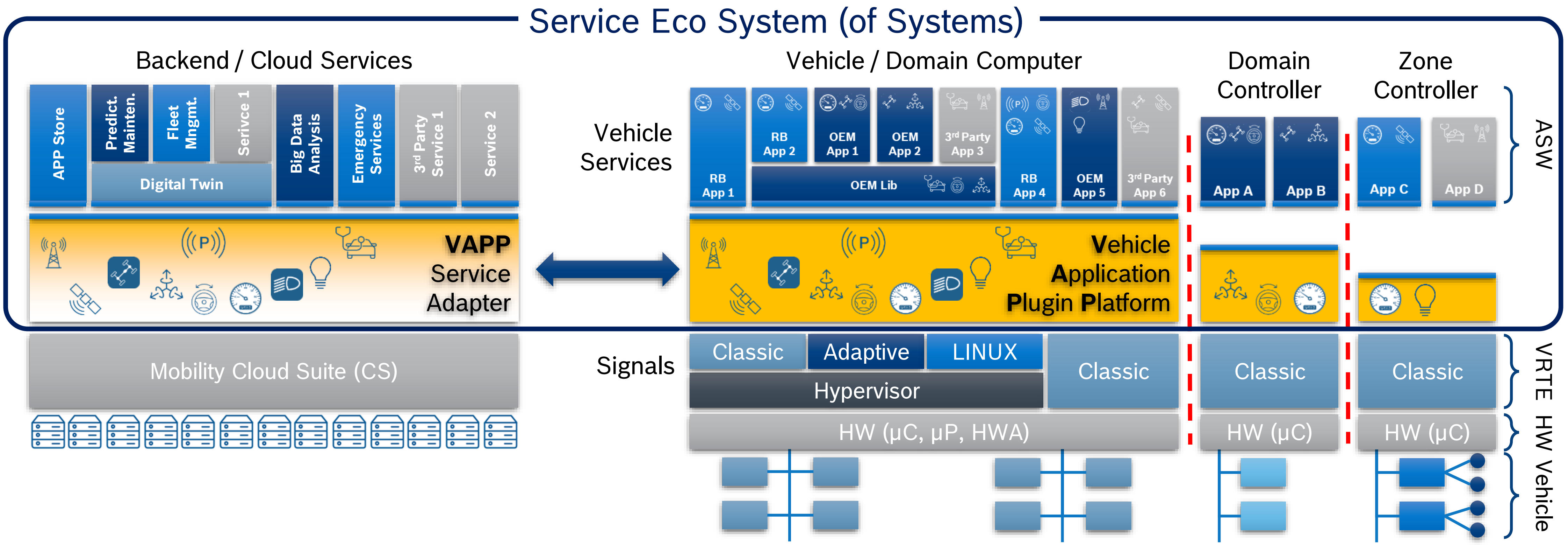


Vehicle Abstraction Layer

Point of View Cloud

BOSCH

Vehicle Abstraction Layer
Vehicle Application Architecture



Service Eco System (of Systems)

Backend / Cloud Services

APP Store

Predict. Mainten.

Fleet Mngmt.

Serivce 1

Digital Twin

Big Data Analysis

Emergency Services

3rd Party Service 1

Service 2

VAPP Service Adapter

Mobility Cloud Suite (CS)

Vehicle Services

Signals

Vehicle / Domain Computer

RB App 1

RB App 2

OEM App 1

OEM App 2

3rd Party App 3

OEM Lib

RB App 4

OEM App 5

3rd Party App 6

Vehicle Application Plugin Platform

Classic

Adaptive

LINUX

Classic

Hypervisor

HW (µC, µP, HWA)

Domain Controller

App A

App B

Classic

HW (µC)

Zone Controller

App C

App D

Classic

HW (µC)

ASW

VRTE

HW

Vehicle

Digital Twin as virtual representation of the Vehicle

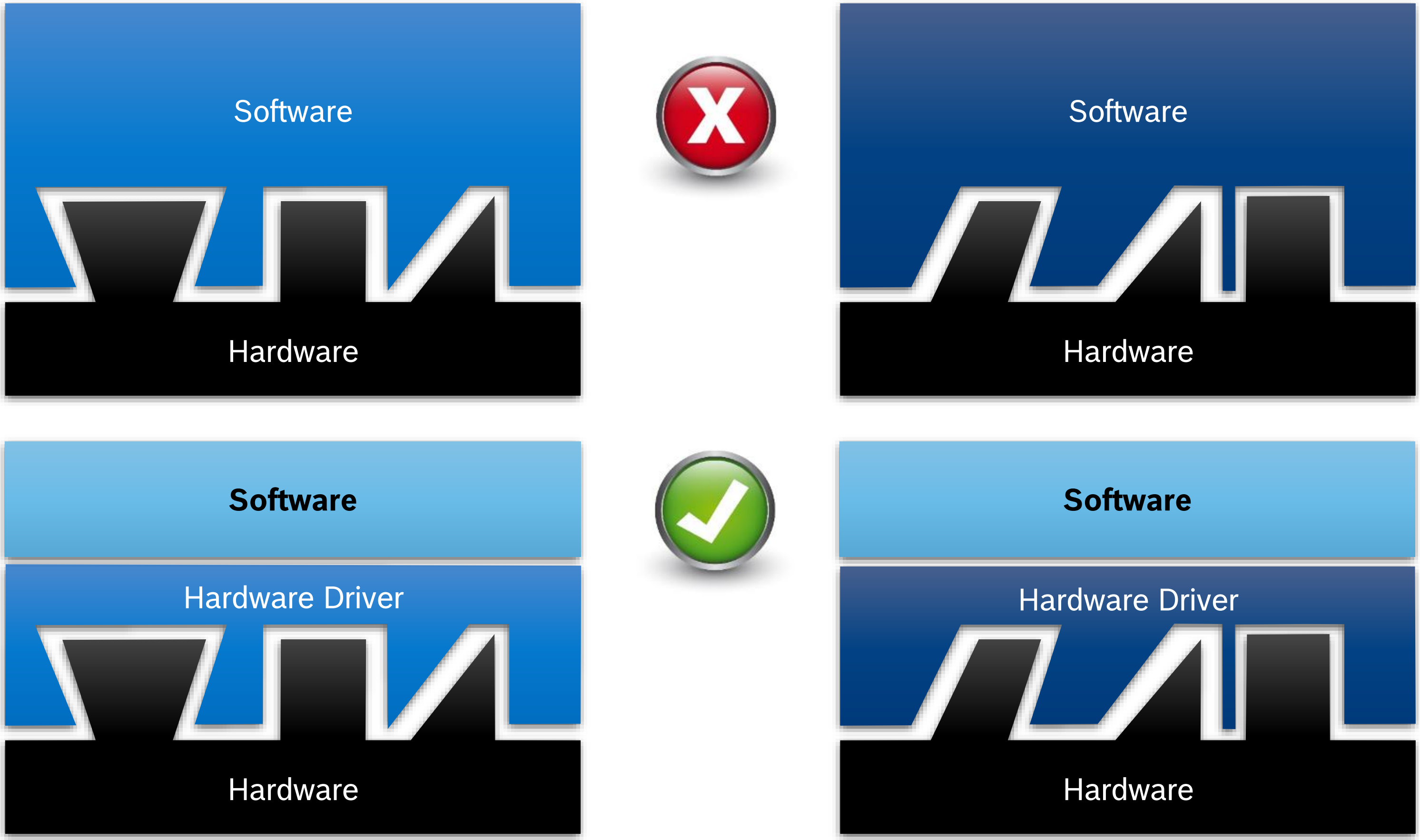
Vehicle Application and Service Interfaces are evolving as trend in automotive service area

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Vehicle Abstraction Layer
Abstraction and Freedom from Interference – ECU / Hardware

Introduction of HW-drivers allows independent development of hardware and software

- Reduction of dependencies and complexity
- Reduction of porting effort to different hardware
- Separation of driver and software development



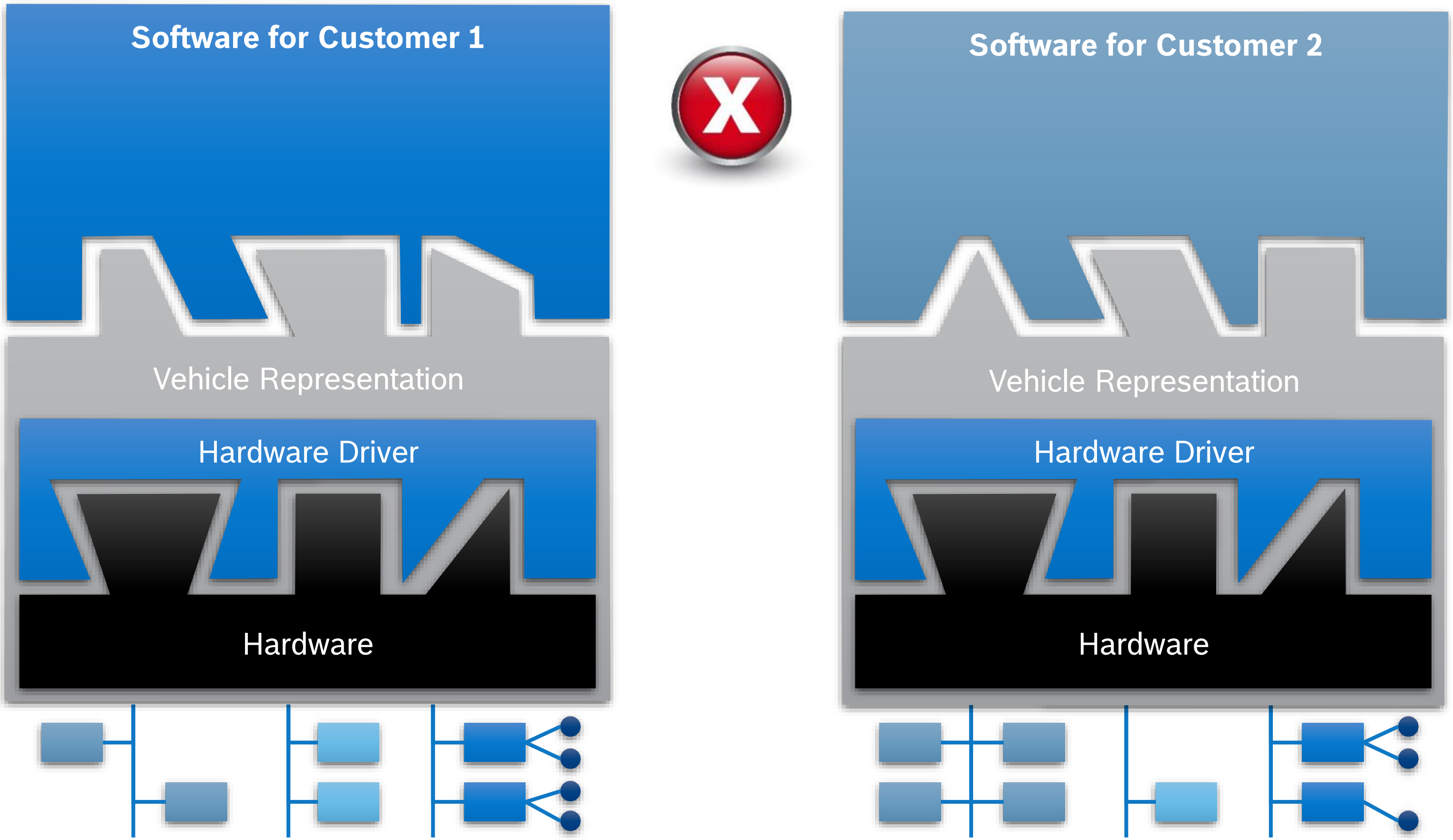
Source: based on AUTOSAR Guided Tour

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Vehicle Abstraction Layer
Abstraction and Freedom from Interference – E/E Architecture

Each Embedded System is reflected on implementation level due to communication, resources and specific component selection

- ▶ Porting software from a device depending on dedicated E/E architecture concept to another concept requires high adaptation efforts

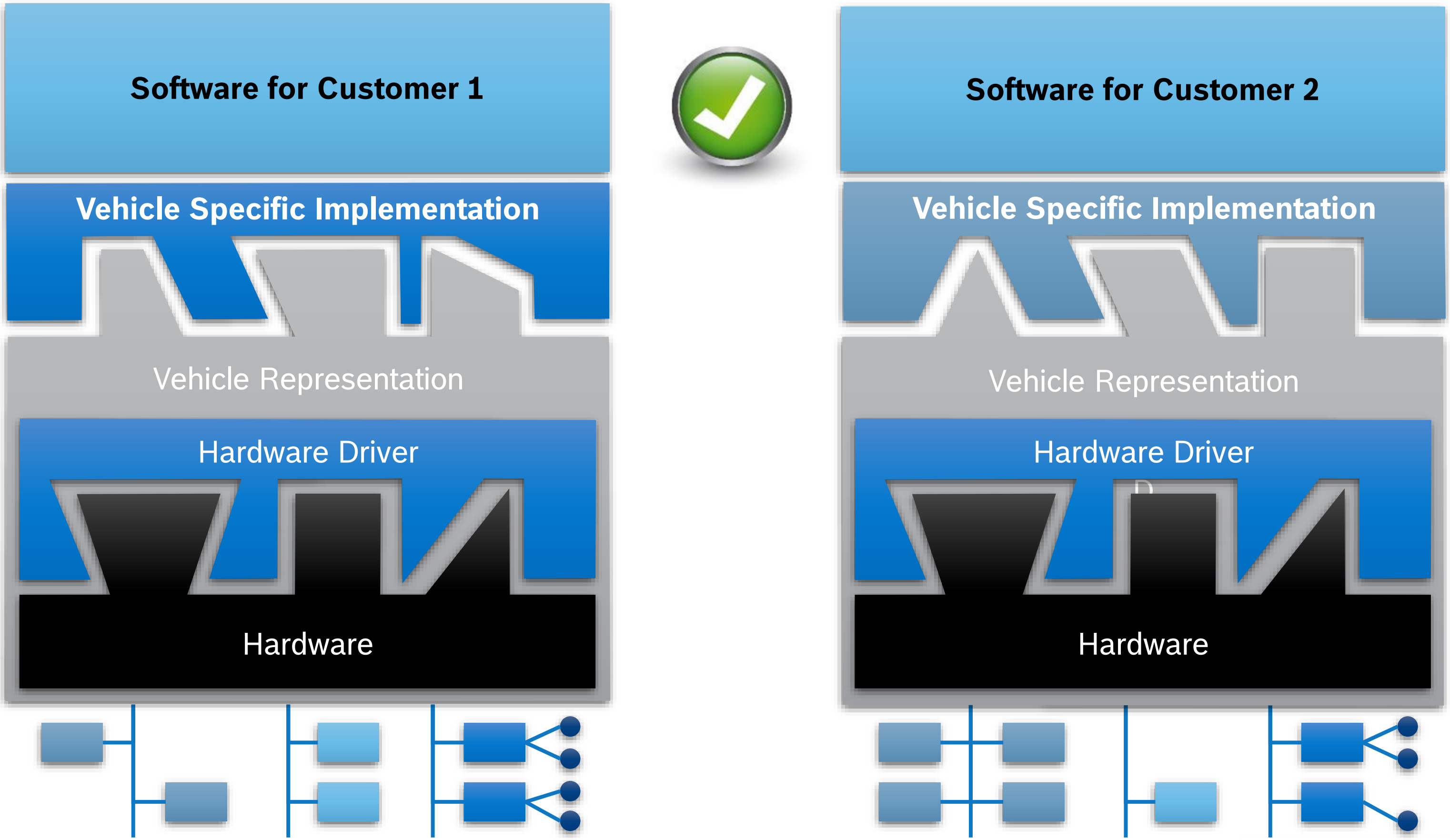


© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Vehicle Abstraction Layer
Abstraction and Freedom from Interference – E/E Architecture

A vehicle specific software layer allows independent development of E/E architecture and software

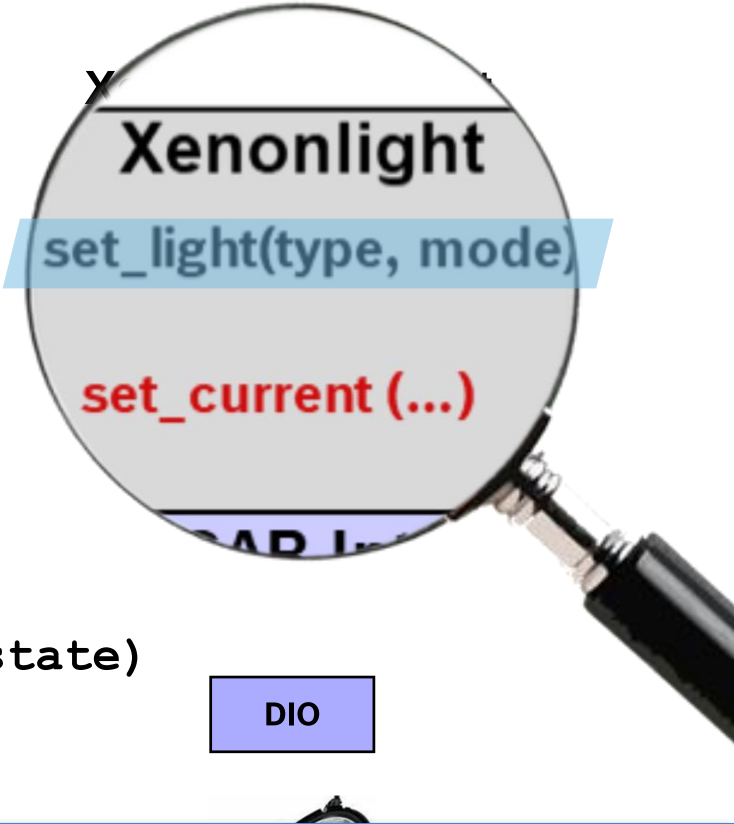
- Reduction of dependencies and complexity
- Reduction of porting effort in case of integration into new E/E architecture
- Separation of vehicle dependent and independent software development



© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Vehicle Abstraction Layer
Example: AUTOSAR: Exchange type of Front Light

```
setLight(enum state)
Set_Light(bool state)
switchHeadLight(enum type, enum mode)
lightOn()
setLight(bool state)
SetLight(bool state)
Set_Beam(enum range)
OP_MOD_Light_Func2(enum param1)
g_DrvReqHB(enum state)
lightSwitchEvent(enum state)
```



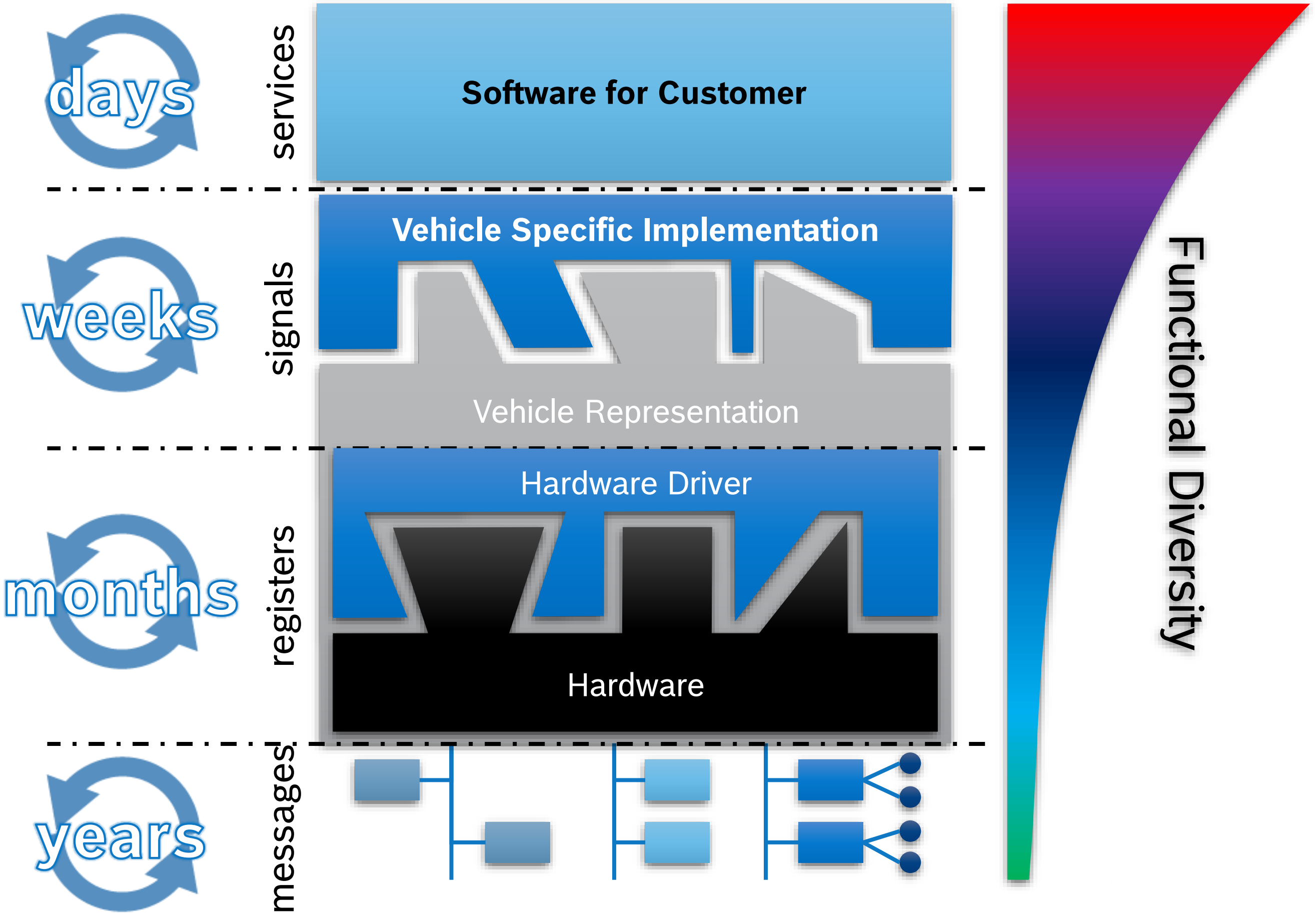
Remaining Challenge: NO Standardized Application Interface over OEMs / Project Borders

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Source: based on AUTOSAR Guided Tour

Vehicle Abstraction Layer
Decoupling of Development & Deployment Cycles

- ▶ Decoupling of implementation reduces effort and complexity
- ▶ Decoupling of deployment cycles allows fast updates for high level features and well-proven processes for embedded functionality
- ▶ Service development does not require knowledge of all future functionality
- ▶ New business models possible due to independent deployment



© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

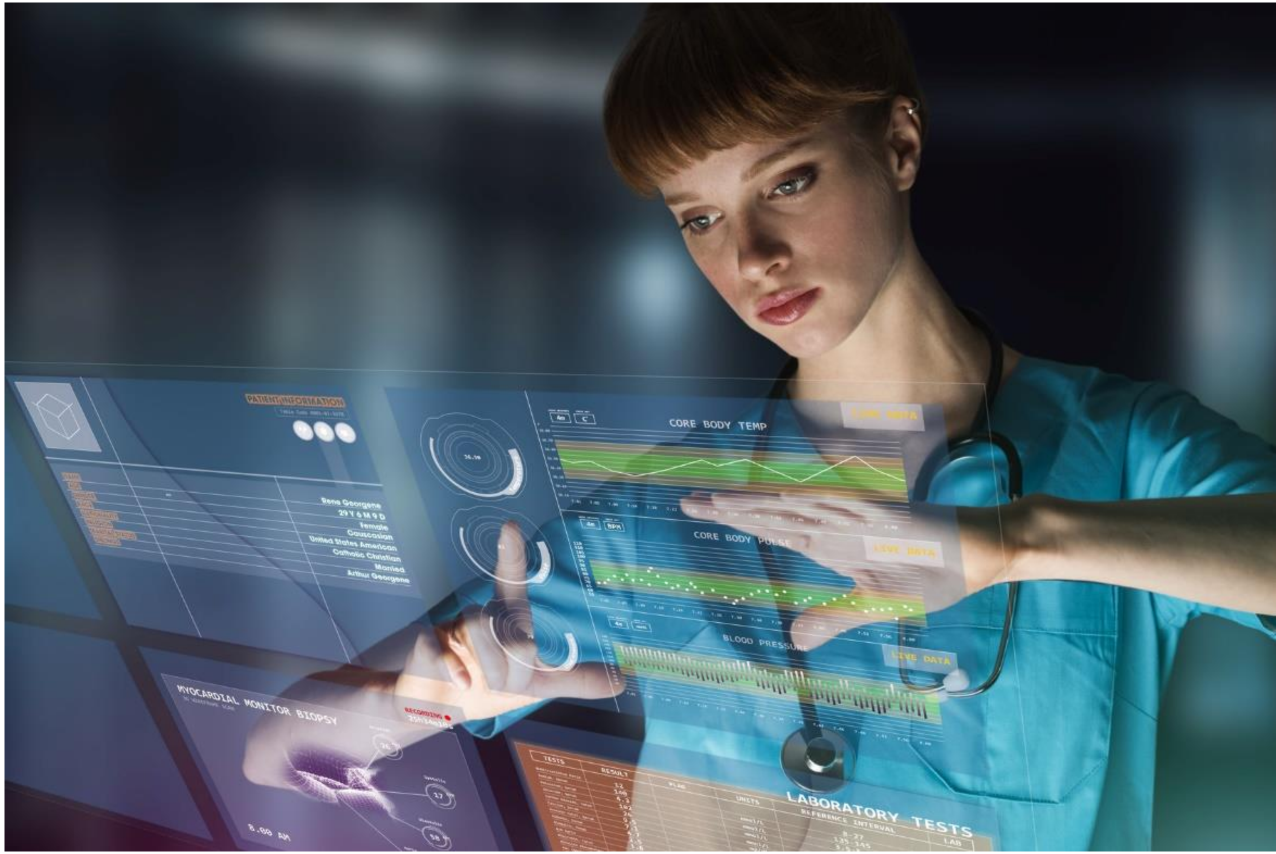
Vehicle Abstraction Layer
“Double Shift Left” utilizing Vehicle APIs and Service IFs

Simulation and Shared Models

- Simulated Vehicle Services for early agile software development
- Increased coverage
- Accelerated test cycles
- Enable early discovery of functional gaps
- Improved cost, time to market, quality

SW Development before HW

- SiL - test bench enables regression and high coverage even before hardware is available



Source: based on Synopsys presentation 2019

© Robert Bosch GmbH 2019. All rights reserved, also regarding any disposal, exploitation, reproduction, editing, distribution, as well as in the event of applications for industrial property rights.

Vehicle Abstraction Layer Conclusion

We need

- Standardized interfaces
- to easily develop functionalities for all kind of vehicles
- which can be distributed faster in a flexible way
- within the vehicle or in the digital twin

Let‘s define this together

Thank you!

Alexander Springer
Technology Strategy, Mobility Sector Software (M/NES)

☎ Mobile +49 160 702 88 98

✉ alexander.springer@de.bosch.com

BOSCH