Real-time Software Systems Engineering (2IN70)

Week2 - Automotive Software Engineering Autumn 2022

Automotive Software Development Challenges – Week2

dr.ir. Ion Barosan – i.barosan@tue.nl

Mathematics and Computer Science

Software Engineering Technology - SET

TU/e Technische Universiteit Eindhoven University of Technology

Contents

Smart Vehicle

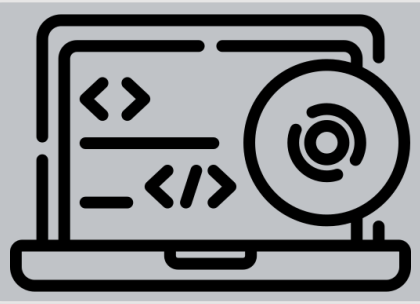
What is Software?

Programming, Programming Languages

What is Software ?

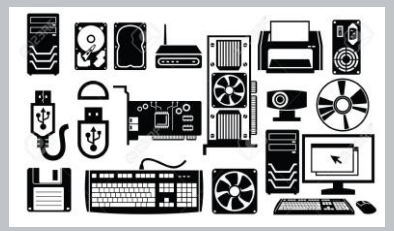
Computer Software

A collection of data or computer instructions that tell the computer how to work



Computer Software Components

Computer programs, libraries and related non-executable data



Computer Physical Hardware

Components which the system is built and performs the work

Computer hardware and software

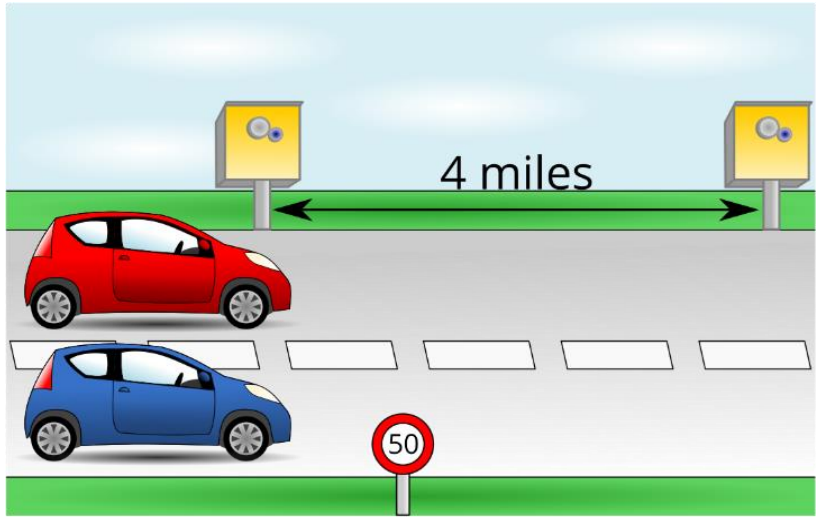
Require each other, and neither can be realistically used on its own

What is Programming?

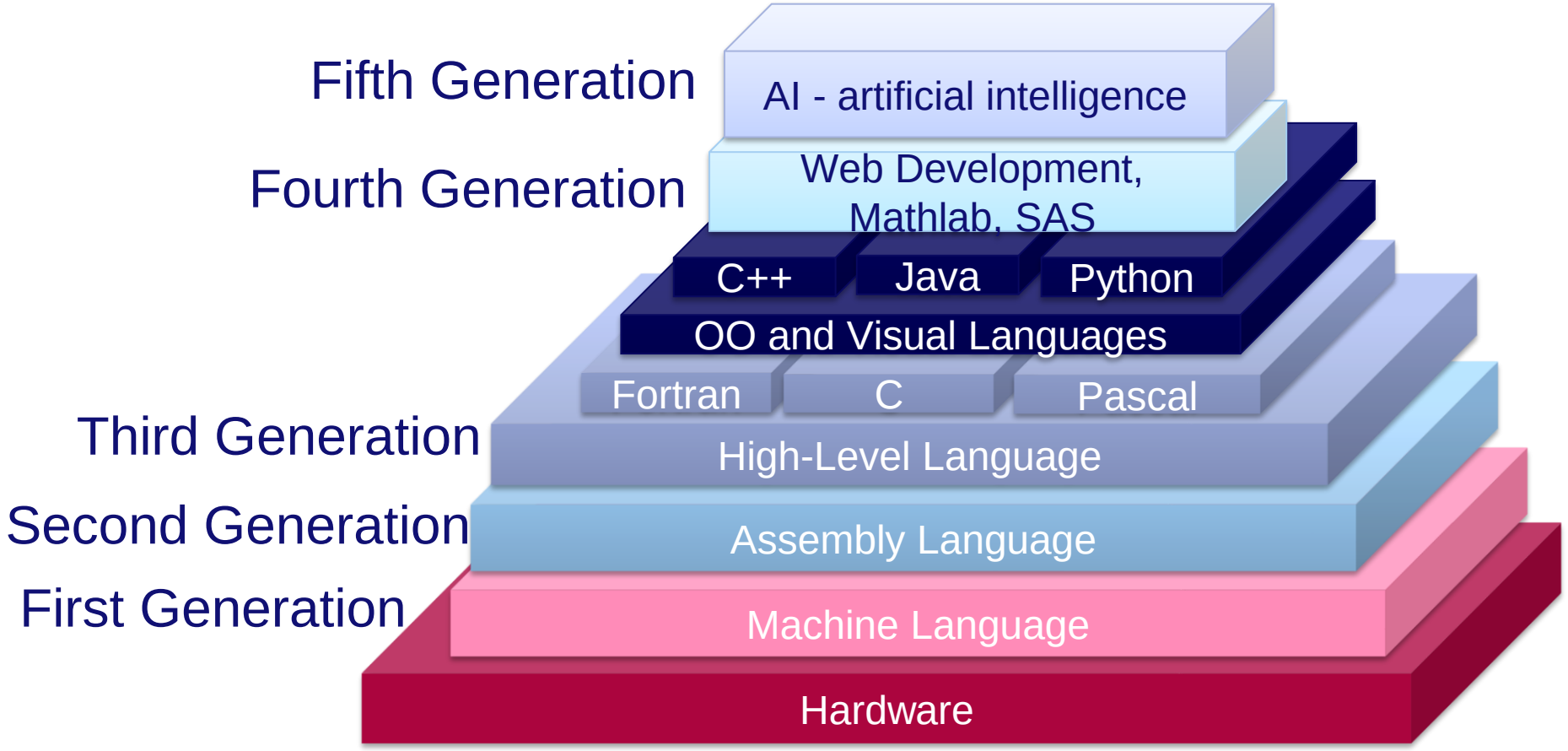
Programming is a way to "instruct the computer to perform various tasks".

"various tasks":

- *Simple* or *complex* tasks
- *Calculate the average speed of a vehicle in motion*
- *Calculating the path of an autonomous vehicle.*



Programming Language



Programming Languages – Top 2019

Rank	Language	Type	Score
1	Python	Web, Enterprise, Embedded	100.0
2	Java	Web, Mobile, Enterprise	96.3
3	C	Mobile, Enterprise, Embedded	94.4
4	C++	Mobile, Enterprise, Embedded	87.5
5	R	Enterprise	81.5
6	JavaScript	Web	79.4
7	C#	Web, Mobile, Enterprise, Embedded	74.5
8	Matlab	Enterprise	70.6
9	Swift	Mobile, Enterprise	69.1
10	Go	Web, Enterprise	68.0

Web | Enterprise | Mobile | Embedded

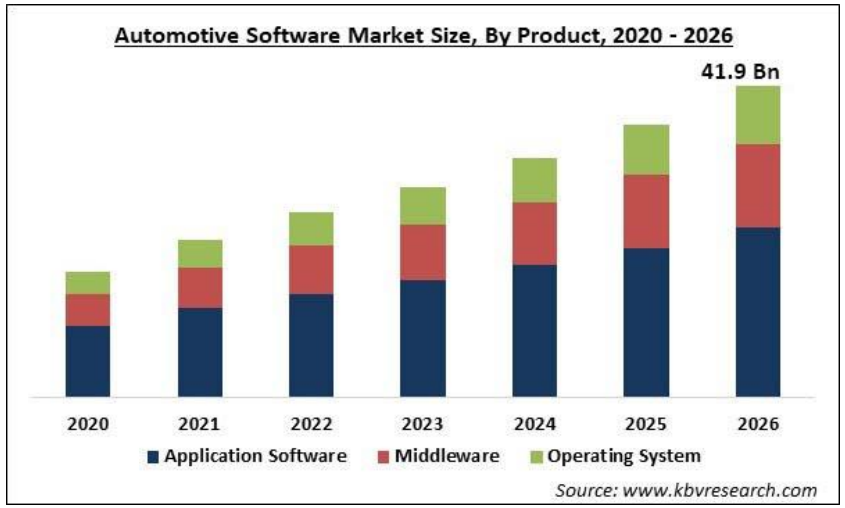
IEEE Spectrum *combines* data from multiple sources to rank the *popularity of the programming languages* that are used for the *type of coding you are interested in.*

Source:
https://spectrum.ieee.org/computing/software/the-top-programming-languages-2019

Industrial Context Automotive Software Engineering Past, Present, and Future

Challenges in Automotive Software Engineering

Automotive Software Engineering



The amount of software found in vehicles has *increased rapidly*

The first lines of code in a vehicle were introduced in the 1970s

Past

100 million lines of code in a premium cars

An increasing amount of functionality is realized in software

Software is the main innovator in the automotive industry today

Present

Automotive Software Engineering - Innovation

Automotive systems can be categorized into

Vehicle-centric functional domains

- Powertrain control
- Chassis control
- Active/passive Safety Systems

Passenger-centric functional domains covering

- Multimedia/telematics, body/-comfort
- Human-Machine Interface

Major areas of potential innovation:

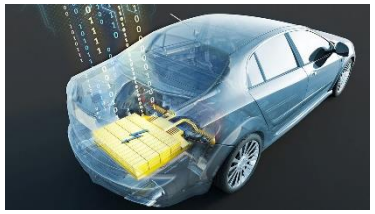
Powertrain	Connectivity	Active Safety	Assisted Driving

Automotive Software Engineering-ACES

Software is rapidly reprogram-ming the car industry

Autonomous Vehicles

Connectivity



Electrification

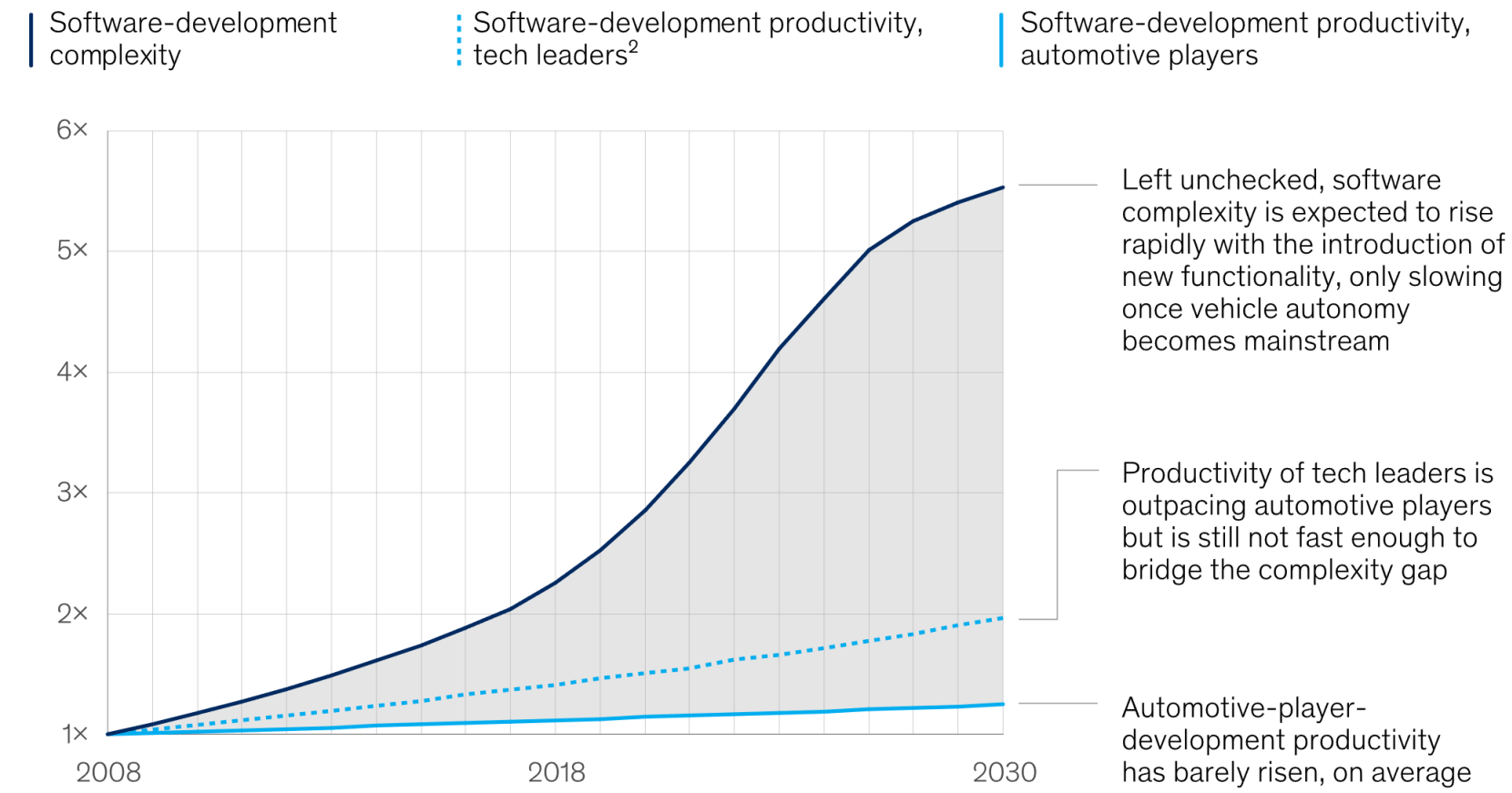
Shared Mobility

The four biggest disruptions in recent years, which rely heavily on leading-edge software - ACES

OEMs, suppliers, and new players try to control the new, software-driven value chain.

Compound Annual Growth Rate-CAGR

Relative growth over time, for automotive features,[1] indexed, 1 = 2008



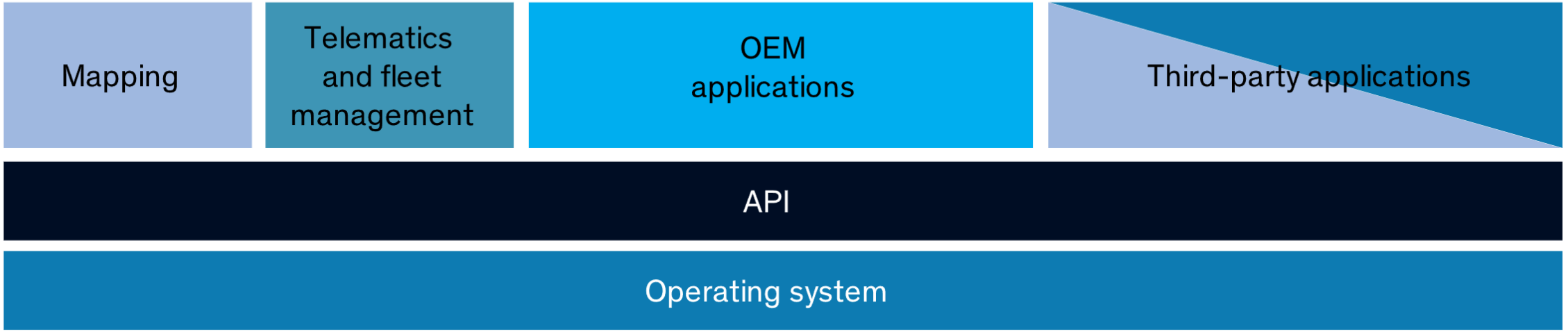
[1]Analysis of >200 software-development projects from OEMs and from tier-1 and tier-2 suppliers.
[2]Top-performing quartile of technology companies.
Source: Numetrics by McKinsey

Source - Mckinsey

OEMs are stitching disparate software components

- ADAS[1]-software provider
- Automotive middleware provider
- Automotive tier-1/ tier-2 support
- Application-specific provider
- Cloud-platform provider
- Diversified tech player
- Human–machine-interface provider
- Operating-system provider
- OEM
- Telecom provider

In cloud - Vehicle-Software Components

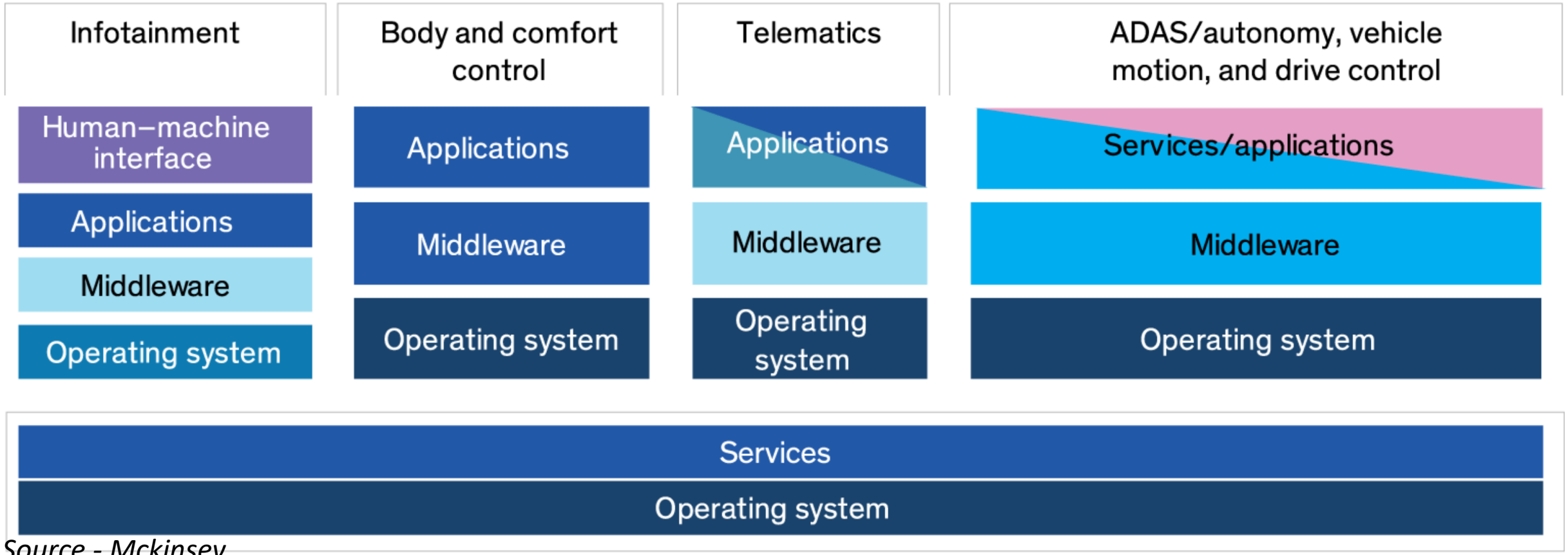


Source - Mckinsey

OEMs stitch disparate software components

- ADAS[1]-software provider
- Automotive middleware provider
- Automotive tier-1/ tier-2 support
- Application-specific provider
- Cloud-platform provider
- Diversified tech player
- Human–machine-interface provider
- Operating-system provider
- OEM
- Telecom provider

In car - Vehicle-Software Components



Source - Mckinsey

Slashing complexity while boosting efficiency – 4D

A new software operating model for OEMs

Four Critical Dimensions

What software is developed	Where is software developed within the organization	How is software developed	How is software development enabled
The architecture Design Requirements	Locations Talent Partnerships involved	Development methodologies - Agile-at scale, changes in development and testing processes	Performance Management Toolchain Infrastructure

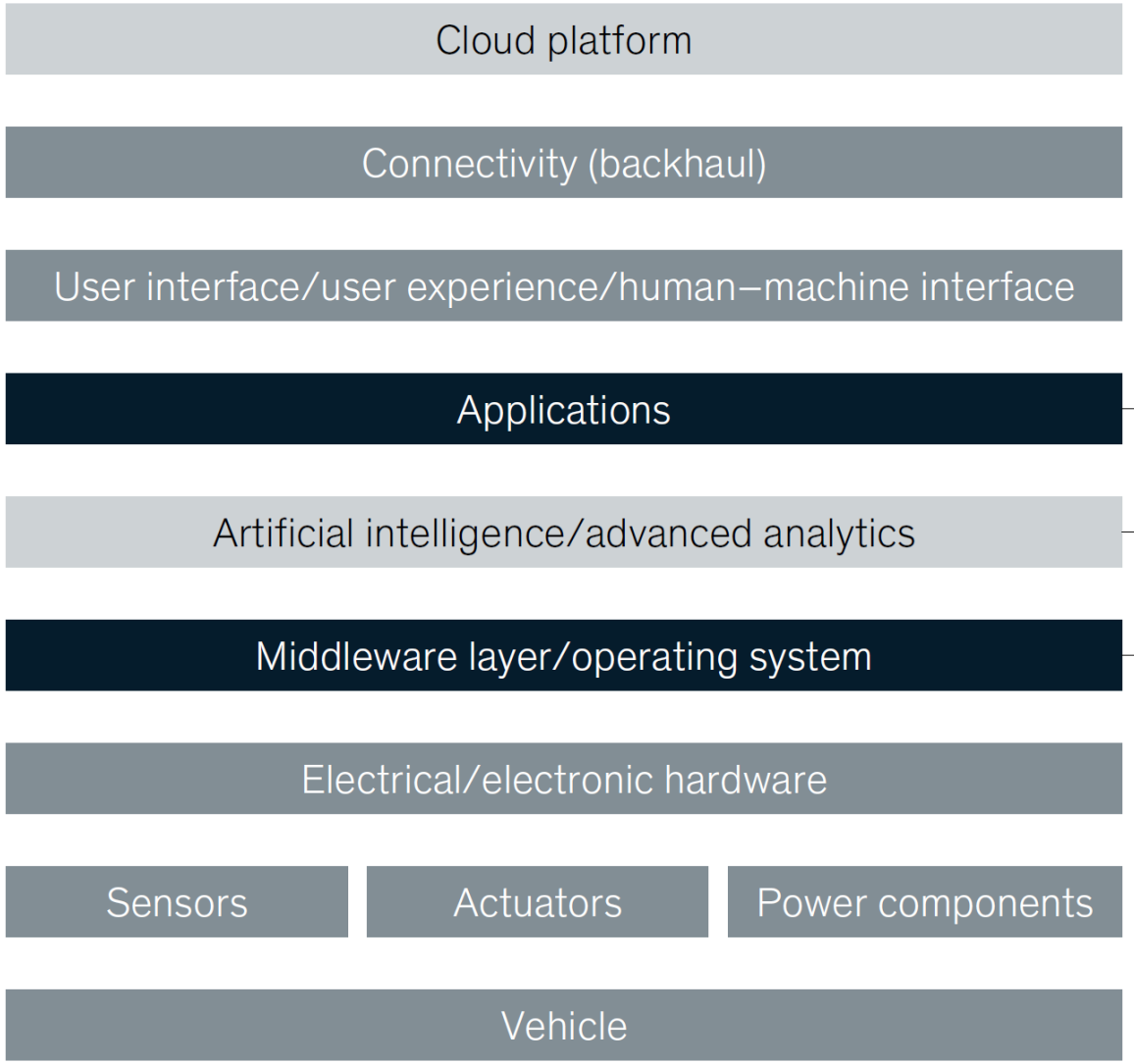
Best practice across dimensions

most important

Apply user-centric design	Adapt management of software requirements	Adapt the organization and establish global centers of excellence	Ensure access and attractiveness to top software-development talent
Reduce architecture complexity	**A** What software is developed	**B** Where software is developed	Define clear make-or-buy strategy and partnership ecosystem
Implement performance management	**D** How software development is enabled	**C** How software is developed	Implement agile at scale Decouple hardware and software development
Upgrade to a standardized, state-of-the-art software development toolchain		Increase test automation and mature continuous integration	

Source - Mckinsey

A target architecture that supports decoupling of hardware and software and features a strong middleware layer



Application layer to be restructured around services

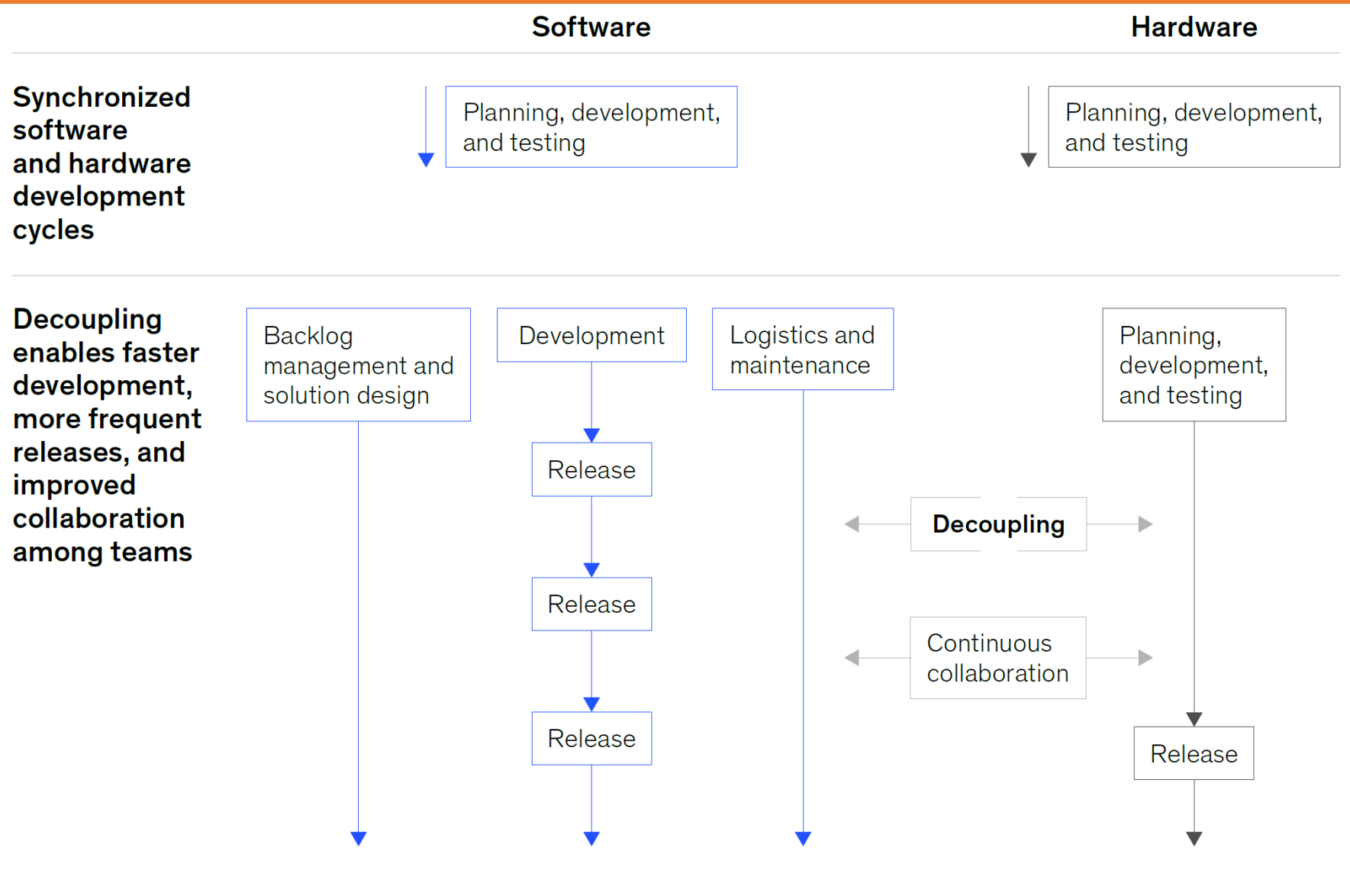
- Applications to be built according to modern software-development practices and without making assumptions about the underlying hardware
- AI/AA to use specialized hardware via middleware/libraries

Middleware layer to abstract hardware and expose APIs to upper layers

- Modular middleware structure
- APIs to define complex, standardized sets of hardware functions that can be used in upper layers
- Standardized operating systems to harmonize functions across multiple types of electronic and domain control units and sensors, ensuring interoperability

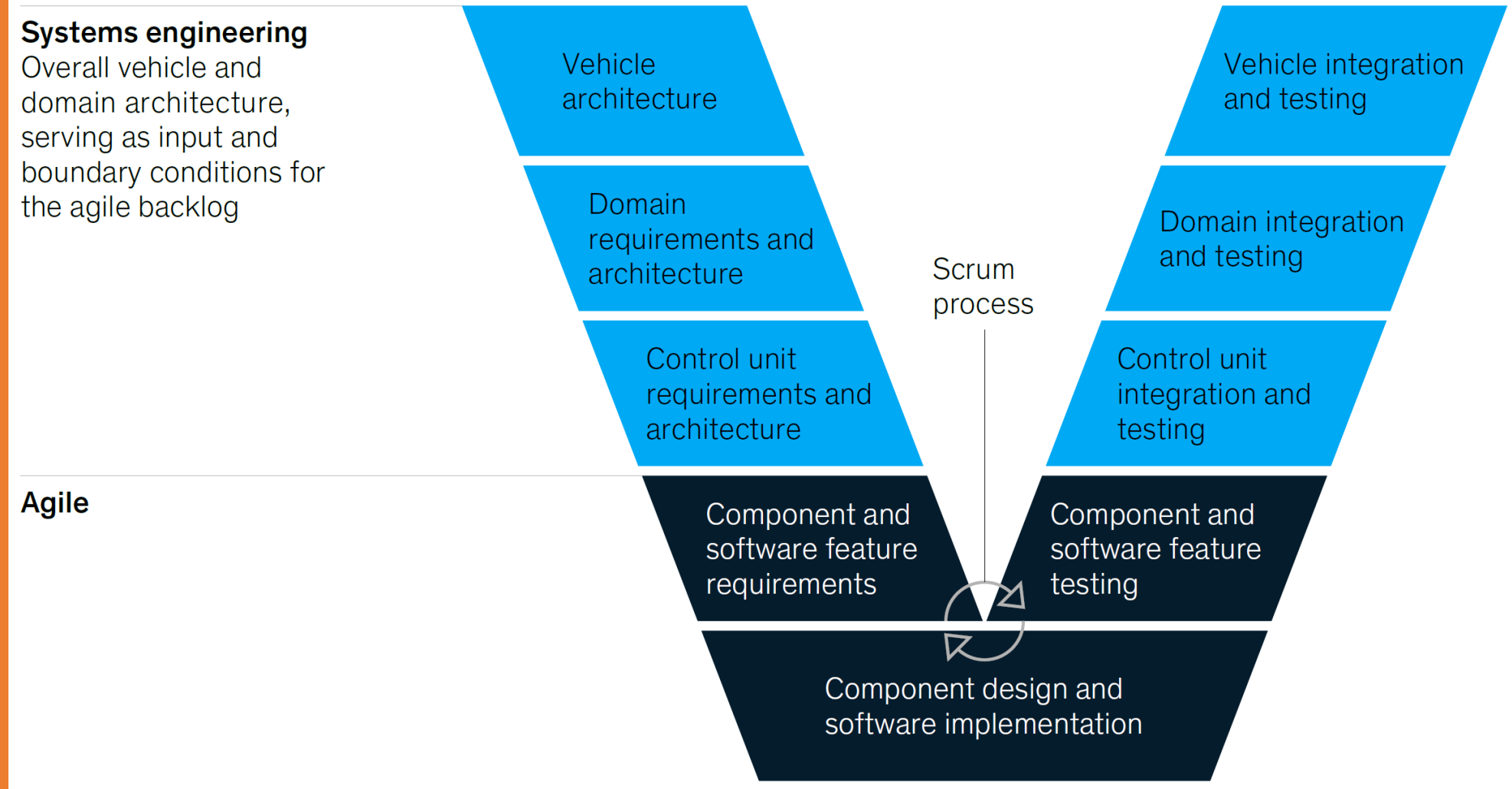
Source - Mckinsey

Decouple software and hardware developme nt cycles.



Source - Mckinsey

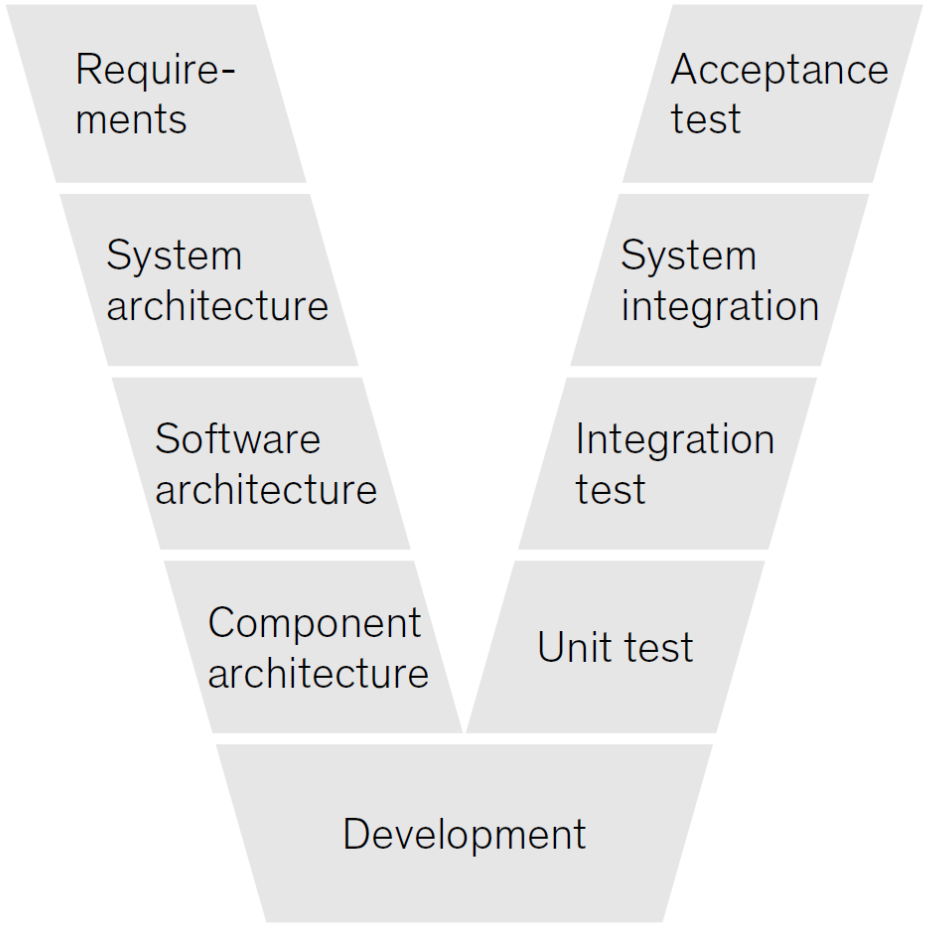
Automotive companies should combine overarching systems engineering with agile software developmen



Source - Mckinsey

Define clear make-or-buy strategies and develop a Partnership Ecosystem

Phase of the development process



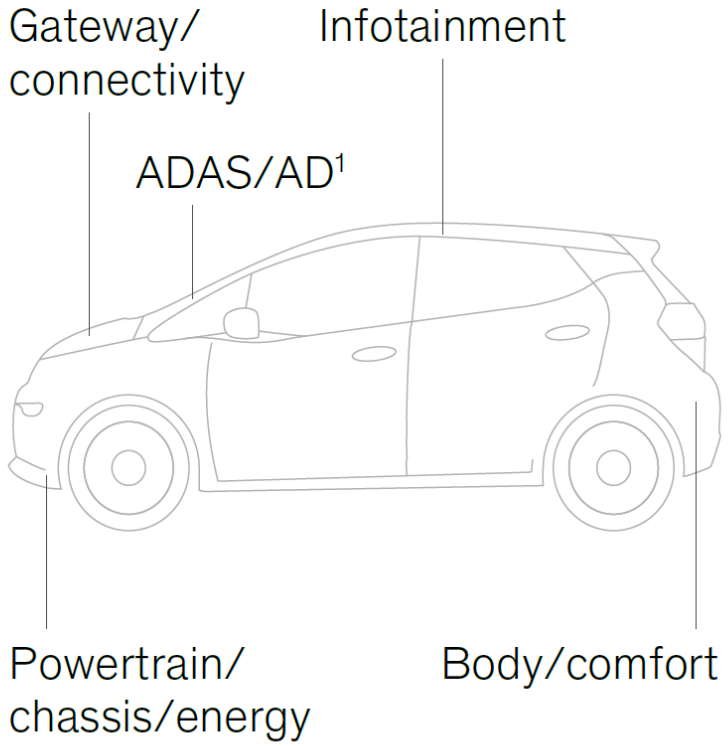
Software technology stack

- Applications and features
- Operating system
- Hardware abstraction
- Firmware and signal processing

Steering and management capabilities

- Infrastructure and tools
- Performance management
- Back end

Software domain/module

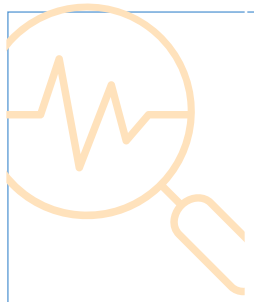


Conclusions – AT Software

To set up world-class software-development capabilities, companies should therefore take an end-to-end transformation view.

Companies may approach transformations in different ways

- Product architecture and make or buy
- Productivity boosts and software-development methodology
- Talent access at reasonable costs
- Organization and governance

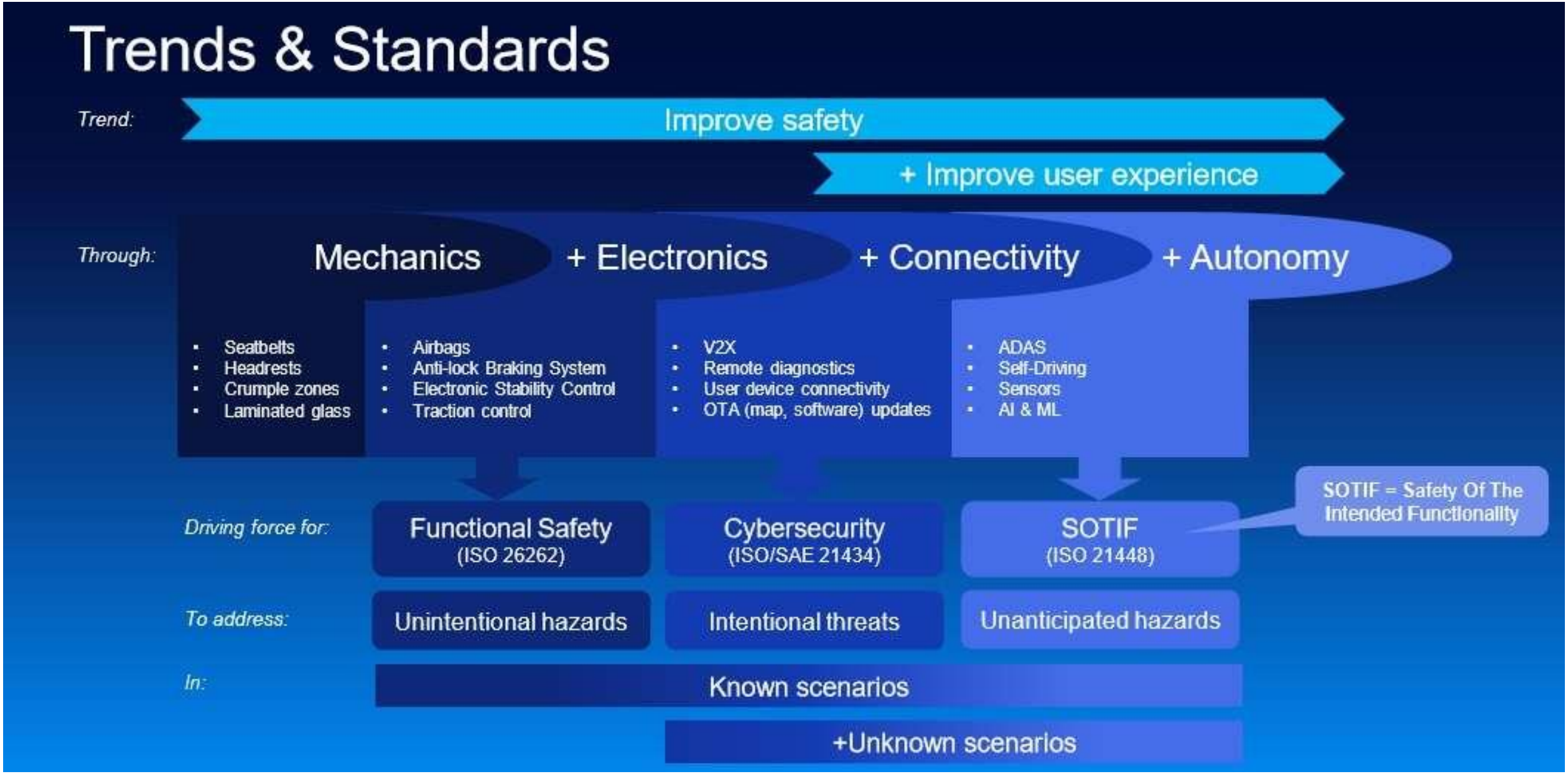


Overcoming the automotive industry's current software complexity and productivity conundrum requires a comprehensive transformation of automotive software R&D.

Automotive Standards

Automotive Standards - Functional Safety ISO26262, Cyber Security 21434, SOTIF 21448, MISRA C, C++

Trends & Standards



(Source: NXP Semiconductors)

Automotive Safety – ISO 26262

Safety - needs special attention during all the stages of the life cycle of a vehicle.

- Failures in software may be costly because of recalls
- May even be life threatening
- Safety requirements

Functional Safety Standards have been developed for safety-critical systems;

- *ISO26262 standard is the functional safety standard for the automotive domain, geared toward passenger cars.*
  - *A new version of the ISO26262 standard will cover trucks, buses, and motorcycles*

Hazard = edge cases

SOTIF - Safety of the Intended Functionality ISO/PAS 21448 [1]

You need to check for Hazard

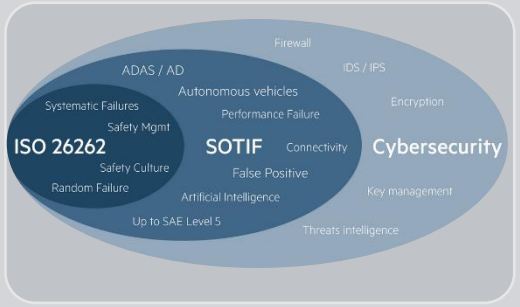
Safety of the intended functionality - for a road vehicle to achieve an acceptable level of safety, unreasonable risk must be avoided—not diluted, not minimized, but avoided entirely

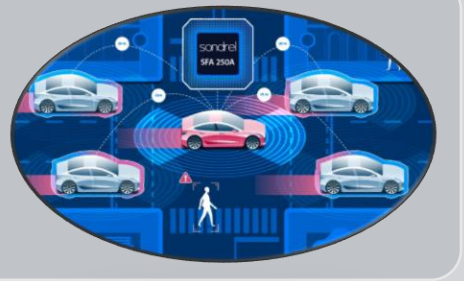
Avoid every hazard associated with both the **intended functionality** and any **unintended functionality** resulting from vehicle use in the real world

Realize the difference between **intended functionality** and **unintended functionality**

The scope of SOTIF is not defined by malfunctions

SOTIF - Safety of the Intended Functionality ISO/PAS 21448 [2]





The scope of SOTIF defines ***properly working equipment that has been designed, built, and tested to fulfill the equipment's given requirements***.

SOTIF is scoped to **specific intentions** ***manifested through thoughtful and deliberate engineering***.

We try to figure out ***whether the vetted and approved requirements for a given system are good enough for what it was designed to manage.***

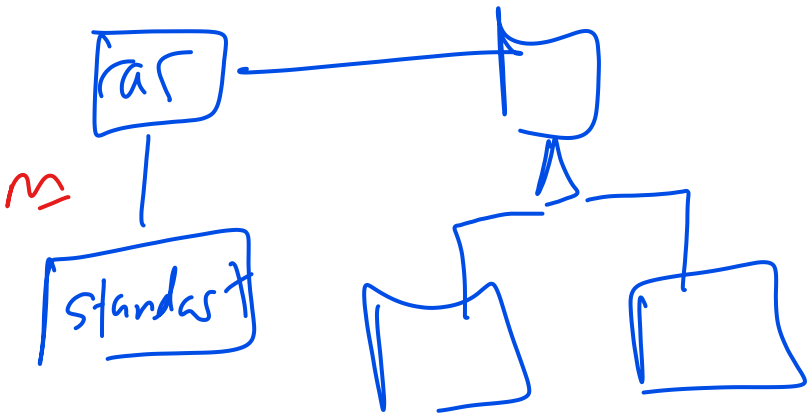
Were the requirements of that system properly specified?

Did we capture the intent of that system?

knowing the operating environment would treat the intent of the car

ISO 21434

functionality is not a problem



Standart

The Car is connected with “everything”

ISO 21434 works to protect vehicle and automotive security

The purpose of the standard is to define a structural process to ensure cybersecurity is “designed-in” from the start

The developers address cyber threats to the vehicle and its electrical and electronic components

car is connected to many system.

cybersecurity ⇒ designed from the start

ISO 21434 Scope

Define a structural process for cybersecurity in design phases.

Establish and **maintain** a consistent framework for cybersecurity globally.

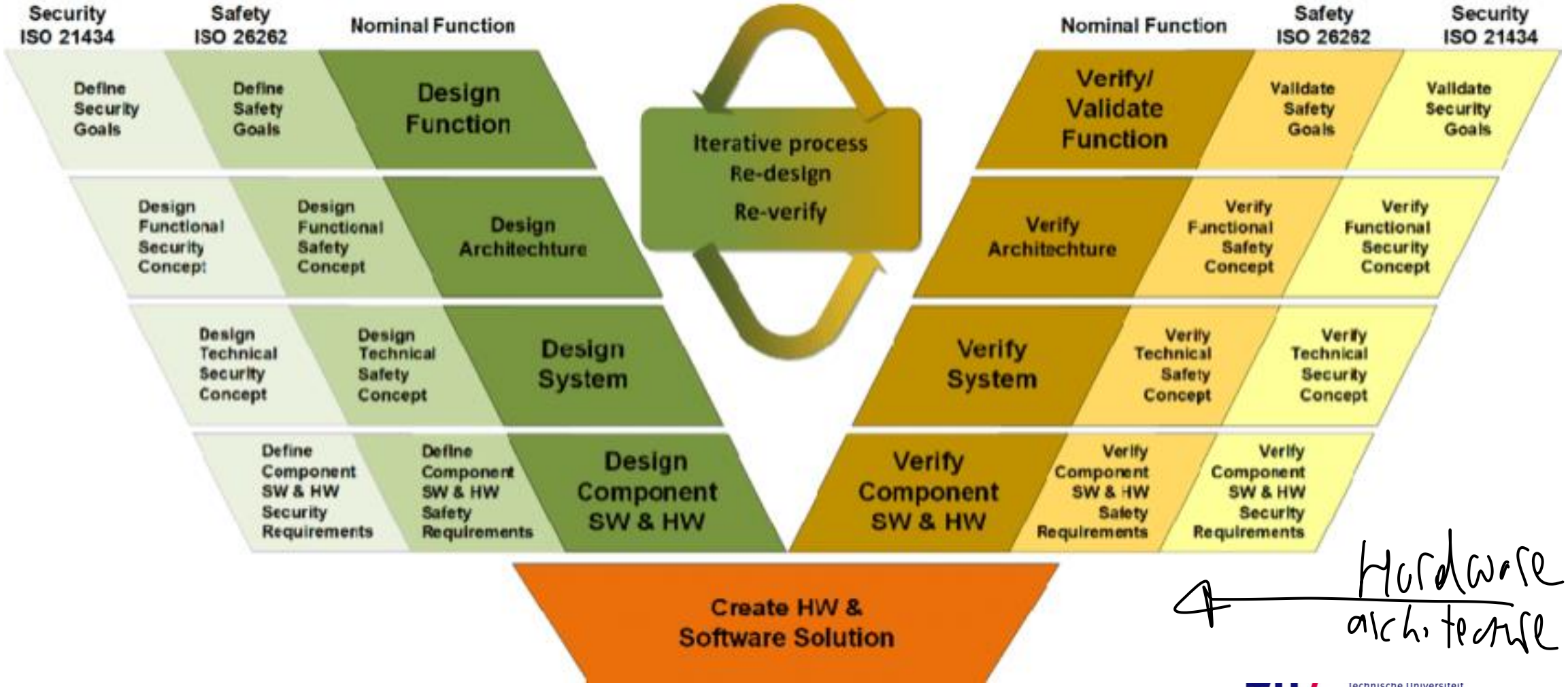
Provide a threat-informed approach to guide security controls.

Adopt and **apply** a risk-based approach.

Provide guidance for developing a CSMS (Cybersecurity Management System) for vehicles across the vehicle life cycle.

an Integrated Automotive Safety & Security Standard

logical architecture



Hardware architecture

MISRA

- doesn't come originally from automotive → often used with C, C++.
Has to comply with a safety standard

MISRA provides coding standards for developing safety-critical systems

The MISRA C coding standard was originally written for the automotive embedded software industry

MISRA standards for C and C++ are widely used by embedded industries

Most of industries have a compliance requirement to use a coding standard — such as ISO 26262

Automotive Standards - Takeaway

anything should comply with the standard →

The automotive industry is starting to apply these standards as guidelines for development projects.

Compliance with these standards is still very costly and time consuming due to huge amount of manual work

Complicated tools

Models of the Standards

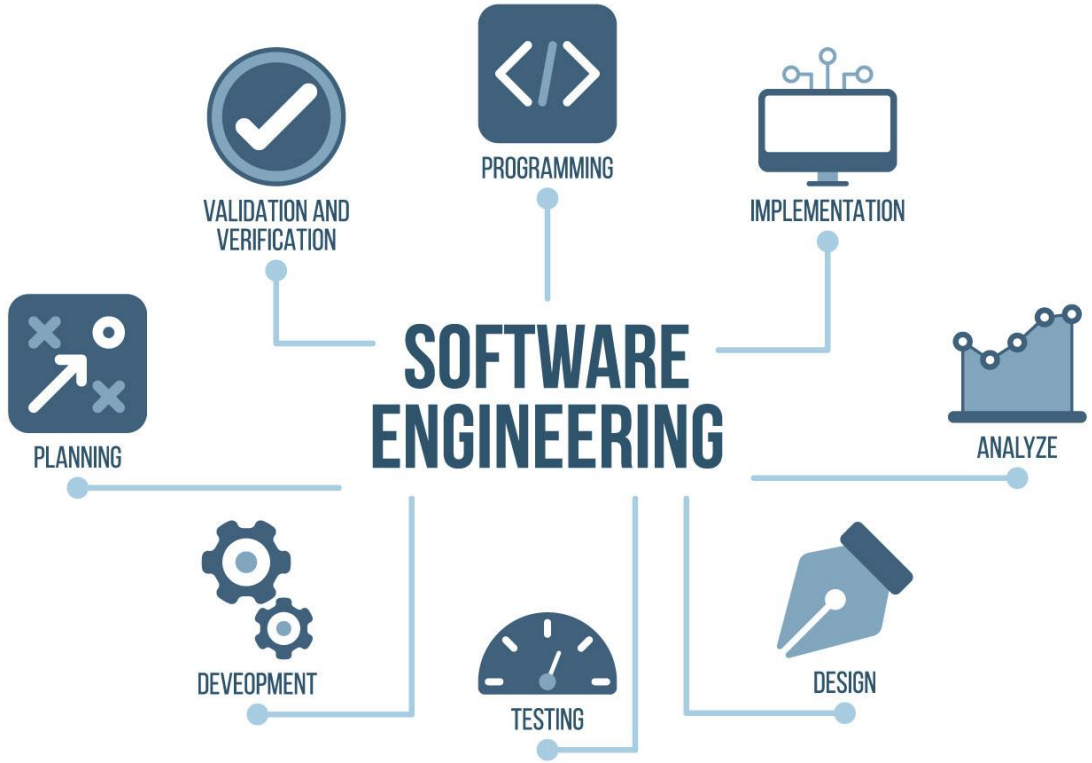
there are companies that verify your code and each tool you use!

Engineer's methodology.

Software Engineering - General

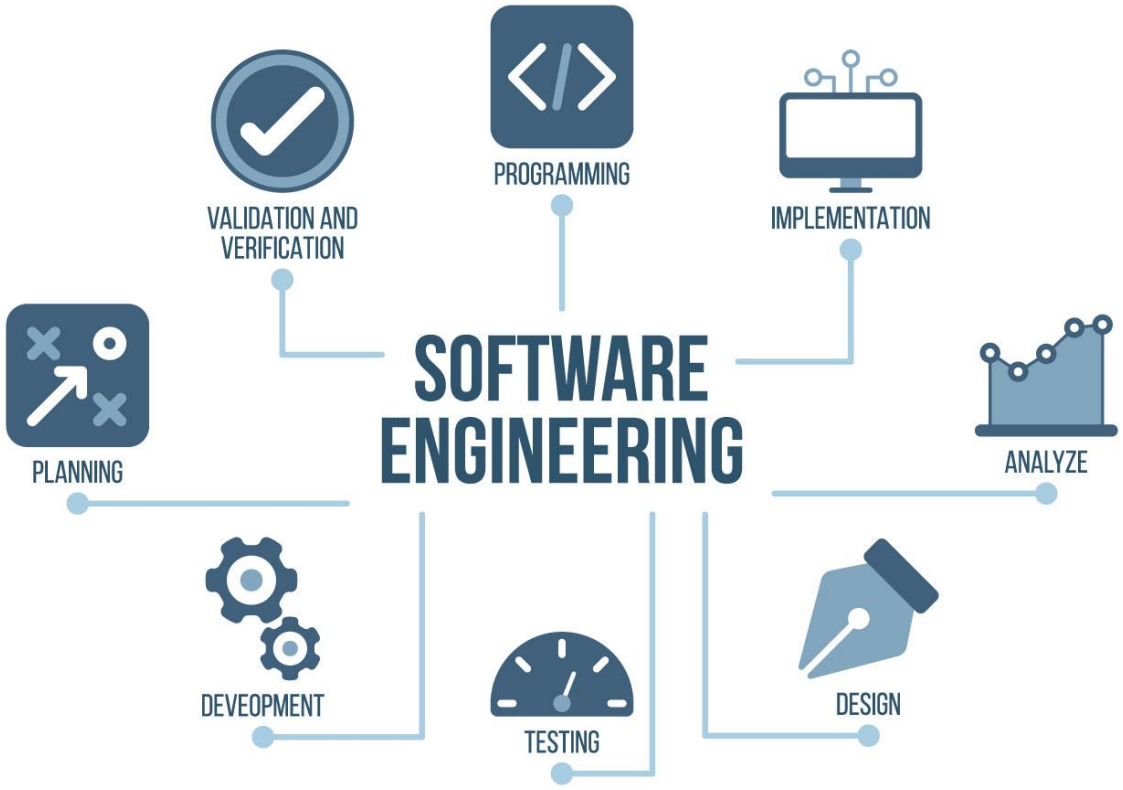
Why Software Engineering?

- The nature of software …
  - Software is everywhere
    - dependable
    - robust
  - Software is intangible
    - hard to understand development effort
  - Software is easy to reproduce
    - cost is in its *development*
      - in other engineering products, manufacturing is the costly stage
  - Software industry is labor-intensive
    - hard to automate



Why Software Engineering?

- The Nature Of Software …
  - Untrained people can hack something together
    - quality problems are hard to notice
  - Software is easy to modify
    - people make changes without fully understanding it
  - *Software does not 'wear out'*
    - it *deteriorates* by having its design changed:
      - erroneously, or
      - in ways that were not anticipated, thus making it complex



Why Software Engineering?

- The Nature Of Software …
- Conclusions
  - *a lot of software has a poor design, and it is getting worse, due to maintenance*
  - *demand for software is high and rising*
  - *we are in an ever lasting '**software crisis**'*
  - *we must learn to really '**engineer**' software*

Why Software Engineering?

- Types of software …
  - Custom - for a specific customer
  - Generic - sold on open market
    - often called COTS (Commercial Off The Shelf)
  - Embedded - built into hardware
    - *hard to change, used to be the case*
    - *car*

Automotive Software Engineering

Evolution of Automotive Software Engineering

Building a vehicle – by integrating components developed and built by suppliers.

Components' specification are defined by Original Equipment Manufacturers (OEMs) or car manufacturers,

Construction of the components is done by suppliers.

The components come with their own Electronic Control Units (ECUs) and software stack.

The individual software stacks add up to the huge amount of software in a modern vehicle.

Advantages:

A clear separation of concerns.

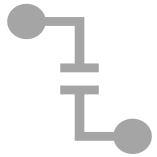
Integration and communication are performed via a Controller Area Network (CAN) bus or FlexRay.

The basis for a decentralized architecture

Evolution of Automotive Software Engineering

Components interacting with eachother

exchange messages via well-defined protocols.



Adding or removing functionality

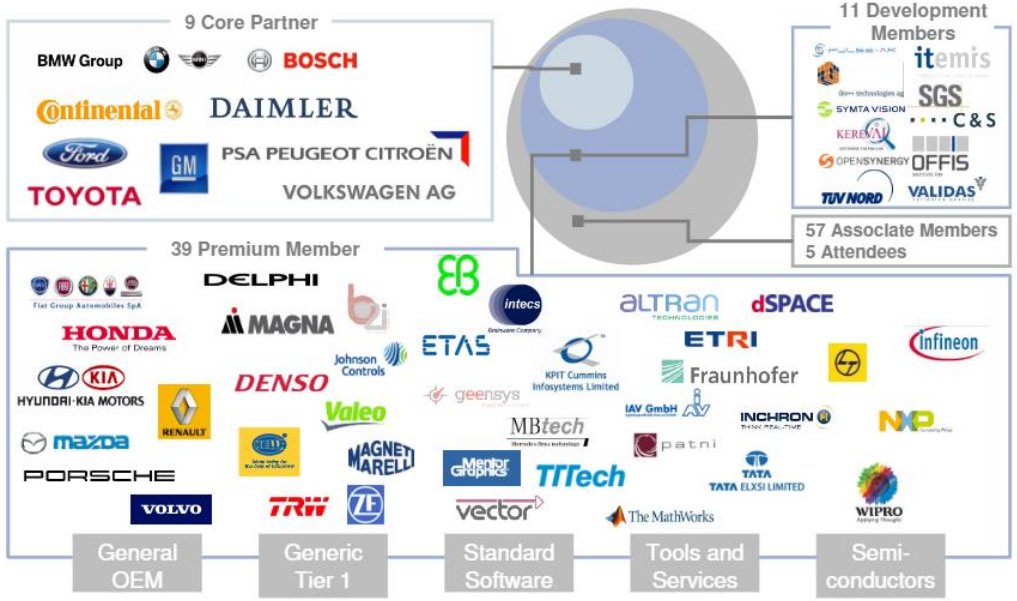
connect or disconnect components from the CAN bus or FlexRay.

The components can be designed in an isolated way

strict alignment with the interface protocols is respected.

The development leads to an explosion of ECUs and their corresponding software stacks.

AUTOSAR (AUTomotive Open System ARchitecture)



AUTOSAR provides a generic layered architecture

- Shields the basic infrastructure of ECUs
- Provides a rather high-level interface to develop functionality

AUTOSAR takes care of mapping functionality to the available ECUs.

AUTOSAR provides a basic software layer

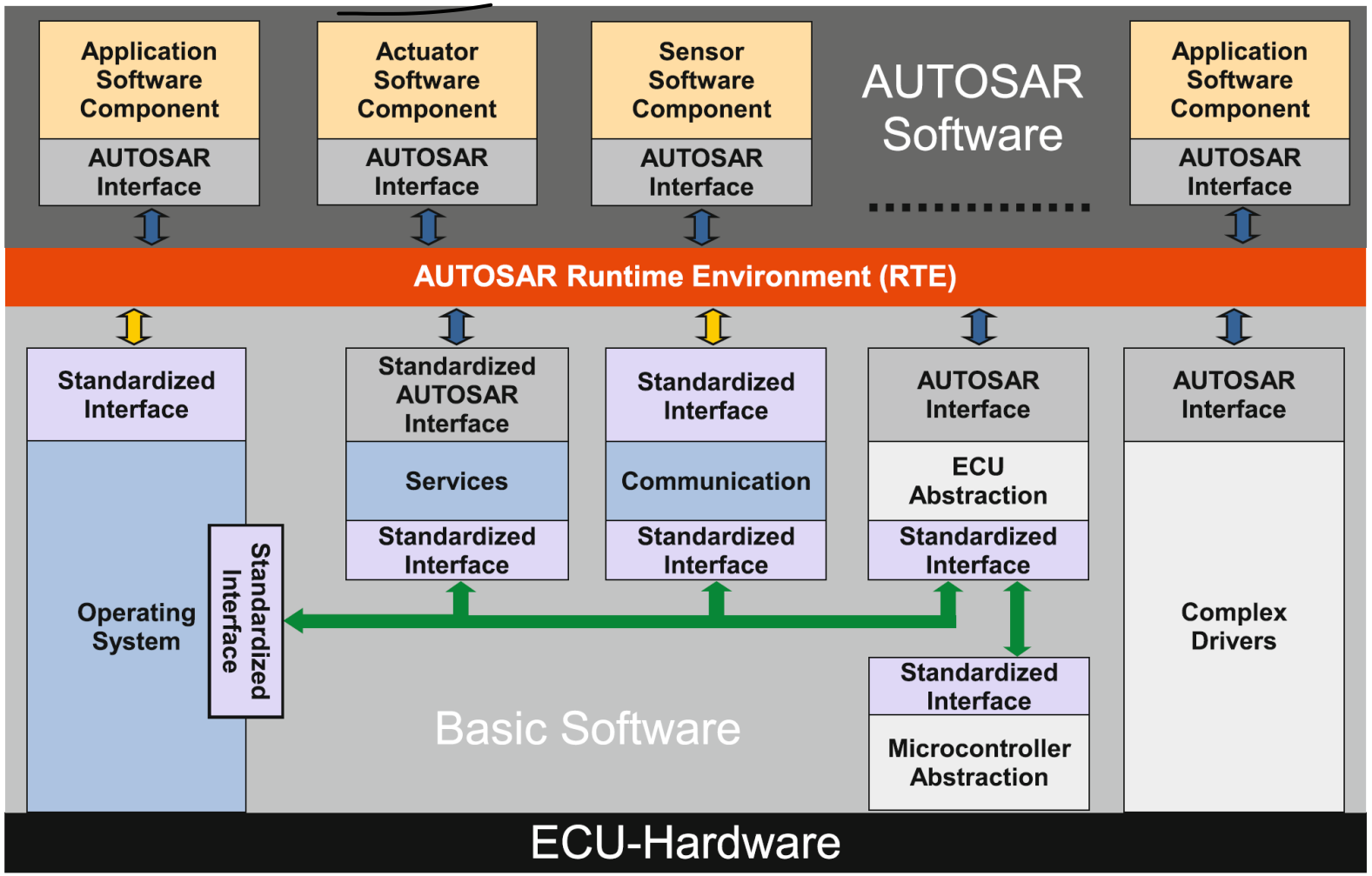
- Standardized software modules (mostly) without any specific functionality
- Offers services to implement the functional part of the application software.

AUTOSAR

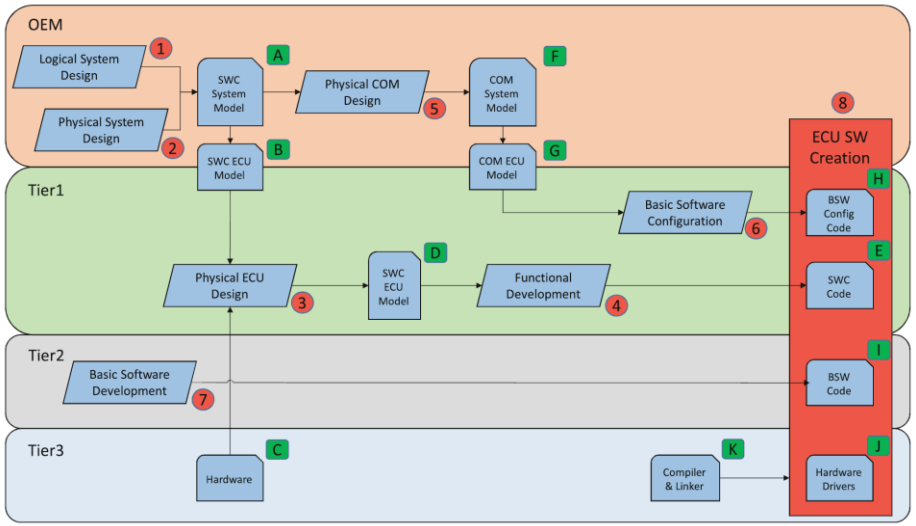
(many) companies are involved.

Its focus is the design, implementation, and realization of automotive systems

different operating system to interact with the hardware - autosar



AUTOSAR – Important to understand



AUTOSAR development process

Domain Application *AUTOSAR* is widely spread in the automotive domain

Rating As *AUTOSAR* does not include an implementation, a third-party tool (e.g., *IBM Rational Rhapsody*) is required to be able to create real projects.

AUTOSAR simplifies model sharing among different development teams and lowers the average initial learning effort.

Why Automotive Software Engineering?

Why Automotive Software Engineering?

Premium cars feature not less than **70 Electronic Control Units** (ECU) connected by more than 5 different bus systems

Up to **40 %** of the production costs of a car are due to electronics and software

On average: ***for 1 year software development about 5 years of testing is needed***

Why Automotive Software Engineering?

Within 30 years the amount of software in cars :

0 - 10,000,000 lines of code

More than 2000 functions are controlled by software in high-end cars

50/70% of the development costs of hard/software are software costs

Why Automotive Software Engineering?

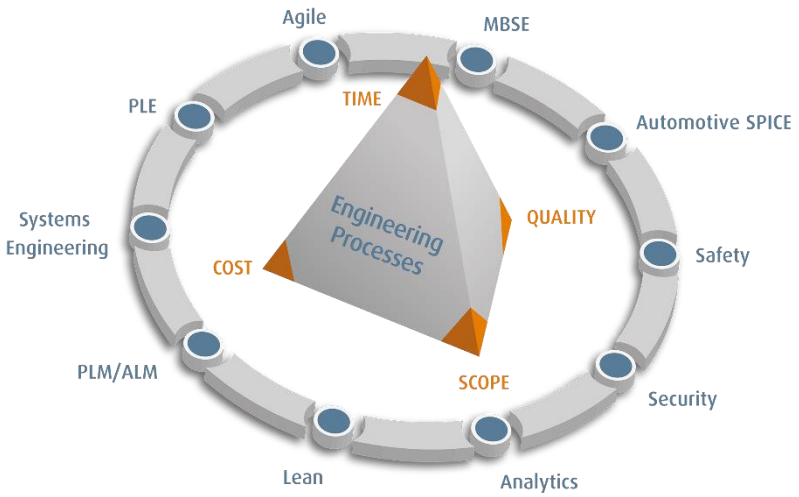
Embedded Software as Innovation Driver

Software is today the most crucial innovation driver for technical systems

By Software

- we realize innovative functions,
- we find new ways of implementing known functions with reduced costs, less weight, less emission, or higher quality,
- we save energy
- we combine functions and correlate them into multi-functional systems

What is Automotive Software Engineering?



Automotive Process Management

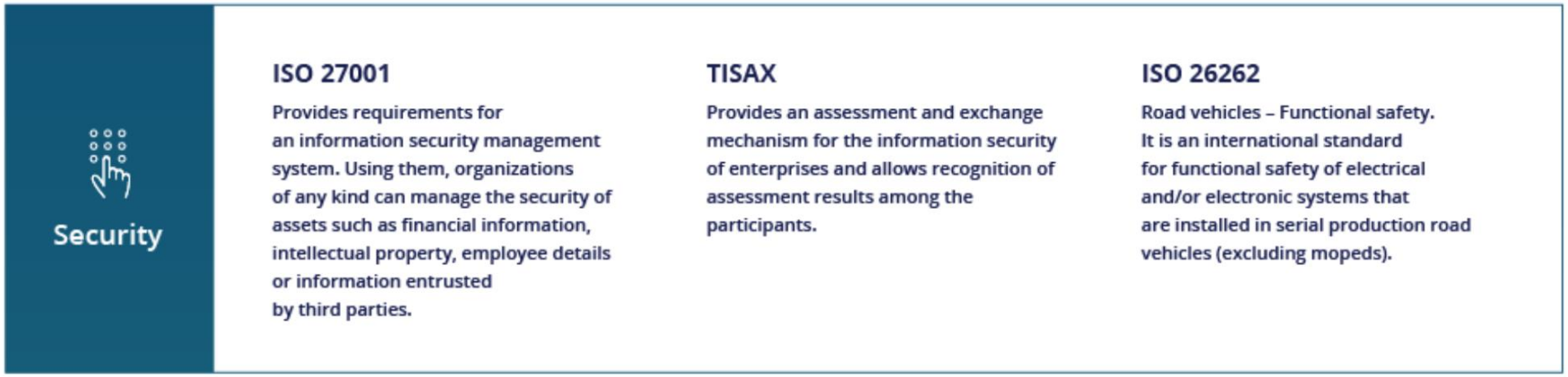
Process models:

- Capability Maturity Model Integration® (CMMI)
- Software Process Improvement and Capability Determination (SPICE)
- V-Model

Standards:

- IEC 61508 (Functional Safety standard)
- ISO26262 (Functional Safety standard)
- MISRA-C, C++
- AUTOSAR

Security Standards for Automotive Industry



- The automotive sector's long and complex supply chain requires a substantial IT security approach to encompass automotive suppliers, marketing companies, and other parties involved. Here are the basic automotive standards that help you build and maintain an ISMS for car manufacturers:
- **ISO 27001**
  - This ISO standard is the basis for creating an Information Security Management System, providing a set of requirements for companies to set up their data and information effectively.
- **TISAX**
  - A maturity-based information security assessment approach TISAX is tailored to the needs of the automotive industry. It applies to 1st and 2nd tier suppliers and extends to more complex supply chains. For certain OEMs, TISAX certification is a must.
- **ISO 26262**
  - It is a safety-related standard for vehicles that include one or more electrical and/or electronic systems and provides recommendations and regulations across the entire product development process: conceptualization, management, development, operation, production, service, and decommissioning.
  - ISO 26262 implementation ensures a step-by-step systematic approach for management of functional safety and regulating product development on different levels: system, hardware, and software.

Automotive Quality Standards

- **IATF 16949**
  - Developed by the International Automotive Task Force, IATF 16949 is the basic certification for automotive manufacturers. It comprises a set of methods for a common product and process development for automotive manufacturers worldwide. It is aimed to support the manufacturing of safe and reliable products and maintain continuous improvements of the product.
- **ISO 9001**
  - IATF 16949 works best with an implemented Quality Management System, so ISO 9001 certification is also obligatory for the automotive sector. ISO 9001 is the most popular standard for QMS, as it provides companies with the requirements that businesses can use to develop their own quality programs. Learn what is QMS and how you can maintain it effectively.
- **ASPICE**
  - ASPICE (Automotive Software Performance Improvement and Capability Determination) is designed to guide automotive companies through the quality matters of the software they use. ASPICE framework enables users to define, implement, and assess the process needed for software development in the automotive industry.
  - This standard was developed in 2005 by car manufacturers based on the ISO/IEC 15504 and relied on the V-Model, requiring testing of each phase on the development stages.

Standards Sustainability

- ISO 14001
  - The automotive industry is specifically required to maintain environmental management systems due to their high impact on nature. ISO 14001 is the primary EMS certification for car manufacturers worldwide to manage and control all aspects of its environmental footprint.
- ISO 45001
  - Health and Safety certification is a distinct sign that a company strives to provide their employees and customers with a safe and healthy environment.
  - ISO 45001 guides businesses to create a framework for controlling and eliminating factors that can lead to injuries or illness.
- **Holistic Compliance Management for Automotive**
  - With a vast collection of standards and regulations to align with, automotive companies do not underestimate the importance of a holistic and tool-driven approach that will enable their CISOs (Chief Information Security Officer) for comprehensive standard management.

Standards Data Protection

- **GDPR**
  - data from partners and *customers must be protected*.
  - businesses working inside or with the EU-member states *must be subject to GDPR and should align with the requirements to build strong DPMS* – Data Protection management Systems.