The essentials for SDV (Software Defined Vehicle)
Service modeling and a Multi-kernel OS

Masaki Gondo, CEO/CTO/President, eSOL

Agenda

- Introduction
- Service modeling
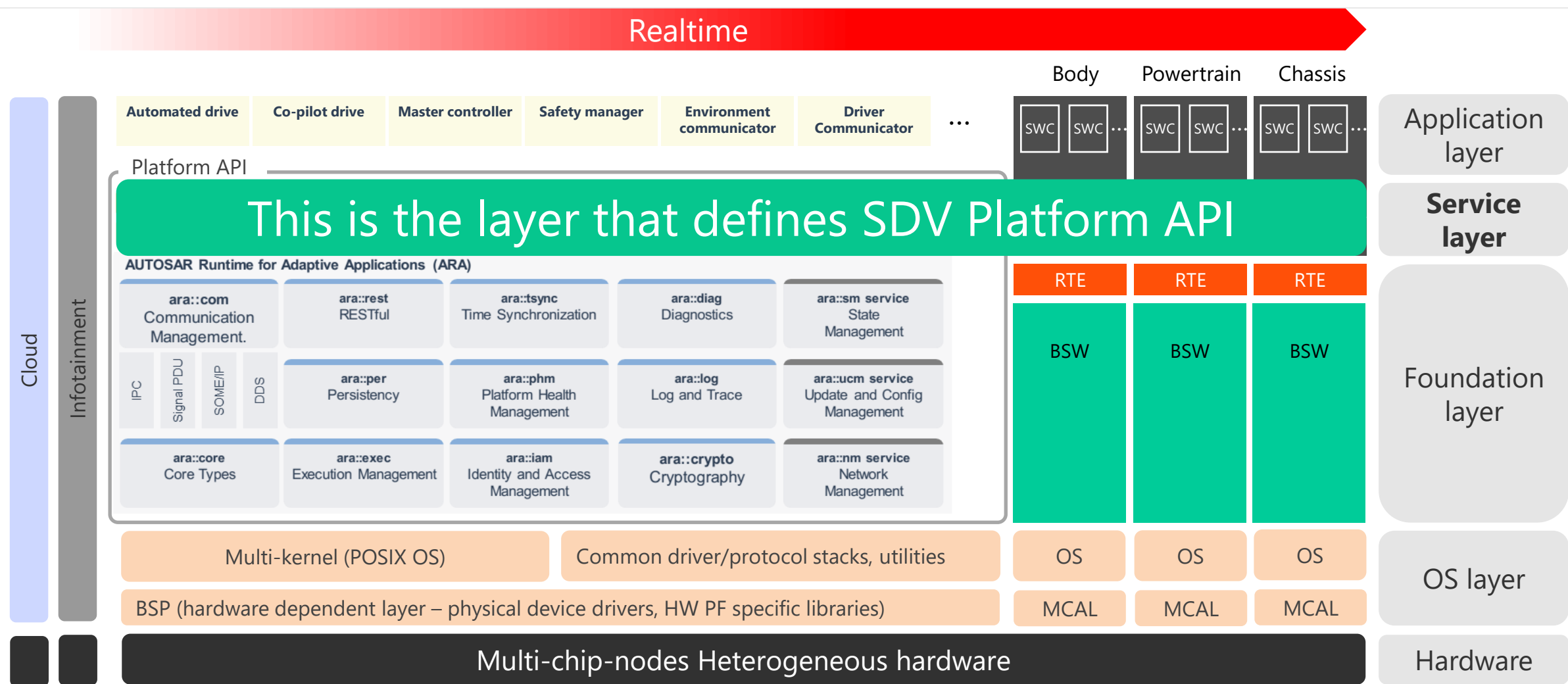- Multi-kernel OS

Introduction

Masaki Gondo, CEO/CTO/President, eSOL

- I was supposed to be a music player but graduated from SUNY ESF in Biochemistry and joined **eSOL** in 1996. **Tokyo-based, 550 people, specialize in embedded software**
- Development of original (RT) **OS**es and applying them to **distributed realtime systems in automotive/industrial/consumer/aerospace** projects with consultations
- Supported **Arm MPCore** in 2006 as the first commercial OS. Announced another **first manycore-capable OS** in 2012 (eT-Kernel Multi-Core Edition and eMCOS) MPSoC 2007 in Awaji, Japan
- Joined AUTOSAR to develop **AUTOSAR Adaptive Platform** in 2016 as one of the original architects, and still a member of architecture group
- **Chaired IEEE2804-2019**, **SHIM** (Software-Hardware Interface for Multi-manycore)
- Certified our OSes for **ISO26262 ASIL-D** and IEC61508 by SGS-TUV
- Introduced **Scrum** in 2015 and integrated it with **FuSa**
- Continuing R&D on **behavioral driver model** technology based on a network of DBNs
- Introduced **Product Management** to the company in 2003
- Executive Consultant for various Japanese automotive OEM/Tier1s
- **Become CEO/CTO/President on March 28th 2025**

Service modeling

The full SDV Software Platform based on SOA

Realtime →

Body   Powertrain   Chassis

| Automated drive | Co-pilot drive | Master controller | Safety manager | Environment communicator | Driver Communicator | ... | SWC SWC ... | SWC SWC ... | SWC SWC ... | **Application layer** |

Platform API

This is the layer that defines SDV Platform API

Service layer

AUTOSAR Runtime for Adaptive Applications (ARA)

ara::com Communication Management.	ara::rest RESTful	ara::tsync Time Synchronization	ara::diag Diagnostics	ara::sm service State Management
IPC / Signal PDU / SOME/IP / DDS	ara::per Persistency	ara::phm Platform Health Management	ara::log Log and Trace	ara::ucm service Update and Config Management
ara::core Core Types	ara::exec Execution Management	ara::iam Identity and Access Management	ara::crypto Cryptography	ara::nm service Network Management

RTE   RTE   RTE

BSW   BSW   BSW

Foundation layer

| Multi-kernel (POSIX OS) | Common driver/protocol stacks, utilities | OS | OS | OS | **OS layer** |

| BSP (hardware dependent layer – physical device drivers, HW PF specific libraries) | MCAL | MCAL | MCAL | |

| Multi-chip-nodes Heterogeneous hardware | **Hardware** |

Cloud

Infotainment

e♨SOL

The need for **non**-automotive domain modeling

- Because SDV is a **"vehicle defined by software"** and software <u>requires a 'model' to design and implement</u> it

- We need to <u>look at the vehicle '**system**' from the top view</u>
  - Fight to **keep distance from so-called "automotive domains"** – physical elements such as power-train, ADAS (sensors and compute), IVI, body, chasis – this <u>departure from the automotive domain thinking is essential in the right system modeling</u> here
  - We need to **decouple** it from hardware, provide non-automotive-friendly schema for the wider-ecosystem essential for both "delivering and capturing the value"

  - Our thinking is wired and driven by words – so <u>the terms/names are critically important</u>!

"Automotive domains"

Powertrain	Engine, Gearbox, Transmission, Battery-Manager...
Chassis	Braking, ADAS, Camera...
Body	Door, Airbag, Climate Control, Seat...
HMI-IVI	Dashboard, Headunit, E-call...
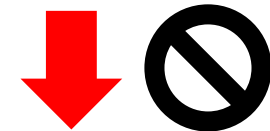
**Do not** start from here! It will force it to be **hardware-defined**

DVE: A holistic and driver-centric vehicle system model

- **DVE** (**Driver-Vehicle-Environment**) model
    - Car has driver, vehicle itself, and the environment that vehicle exist
    - This is the highest model, and services shall be defined beneath
    - Additionally, 'Utility' domain is added to handle the aspect oriented, system-
      becomes "**DVE+U**" model

- Plenty of prior studies based on DVE
    - Prof. Doi in 2006 – one of the oldest paper that discusses DVE
    - Has been used in various researches, spanning from **cognitive science**, **AD/ADAS**, s   evolves around
      "**human centric design**"
    - Effectively the same model is often used in the fields of **behavioral driver model st   d **adaptive
      HMI**
    - DVE model is a good candidate as the basis for the SDV requiring the **holistic top-level view**
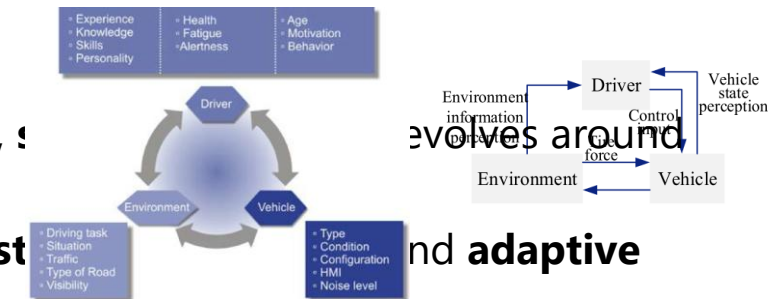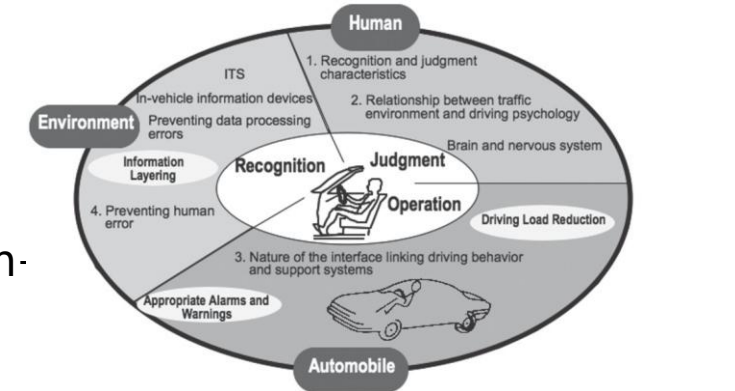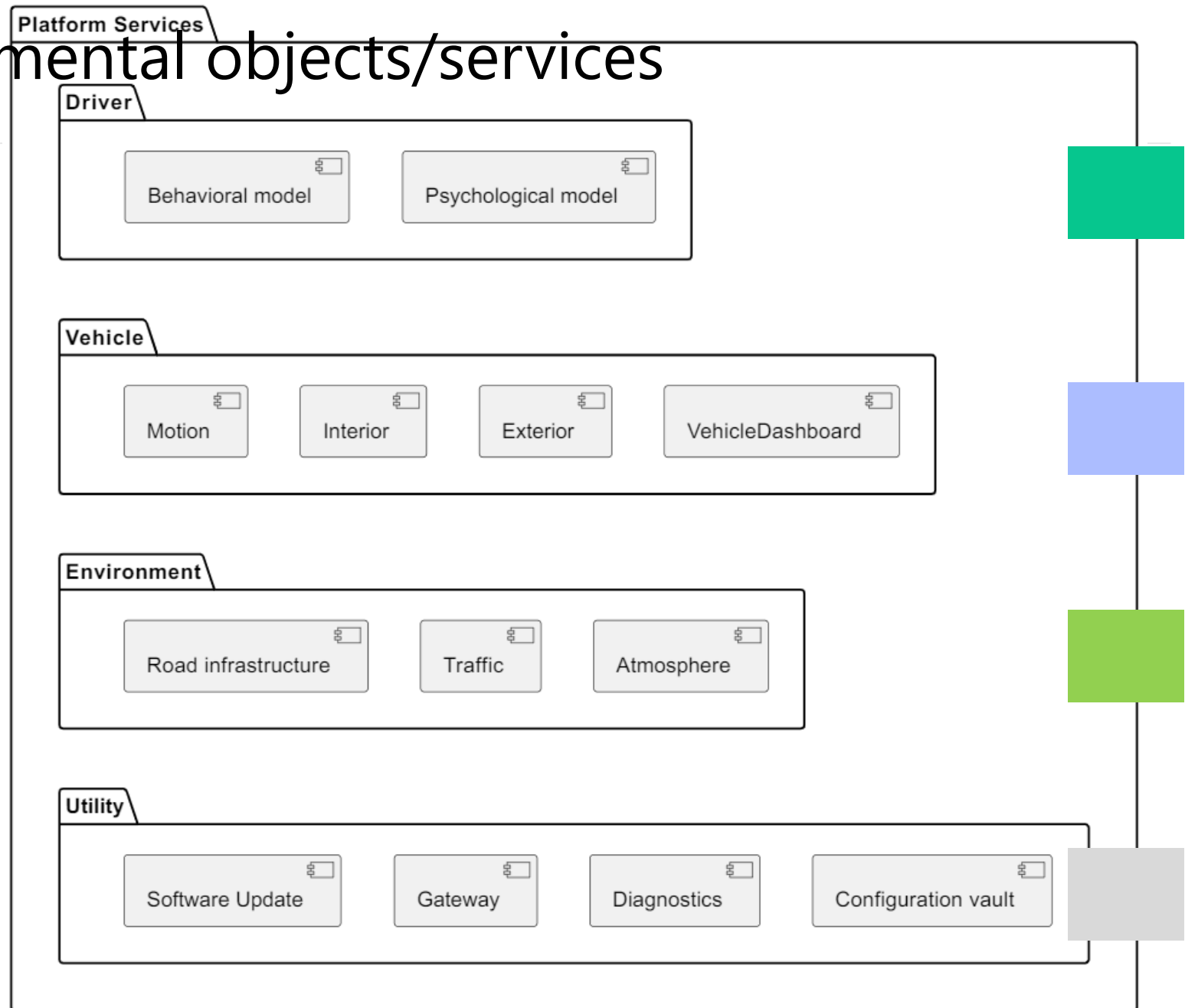
1. S. Doi, "TECHNOLOGICAL DEVELOPMENT OF DRIVING SUPPORT SYSTEMS BASED ON HUMAN BEHAVIORAL CHARACTERISTICS," *IATSS Research*, vol. 30, no. 2, pp. 19–28, 2006, doi: 10.1016/S0386-1112(14)60166-7.
2. S. Xie, S. Chen, N. Zheng, and J. Wang, "Modeling Methodology of Driver-Vehicle-Environment System Dynamics in Mixed Driving Situation," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, Las Vegas, NV, USA: IEEE, Oct. 2020, pp. 1984–1991. doi: 10.1109/IV47402.2020.9304850.
3. A. Capustiac, B. Hesse, D. Schramm, and D. Banabic, "A human centered control strategy for a driving simulator," vol. 11, no. 01, 2011.
4. M. Da Lio et al., "Artificial Co-Drivers as a Universal Enabling Technology for Future Intelligent Vehicles and Transportation Systems," IEEE Transactions on Intelligent Transportation Systems, vol. 16, no. 1, pp. 244–263, Feb. 2015, doi: 10.1109/TITS.2014.2330199.
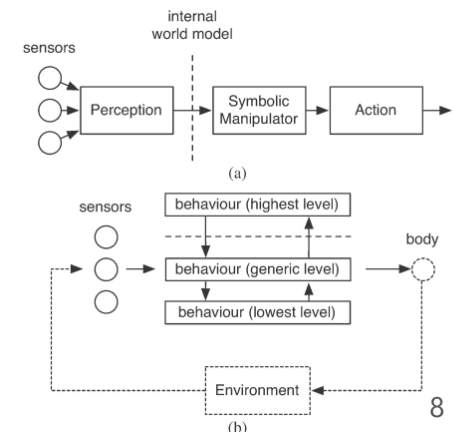
DVE+U model – fundamental objects/services

- The model is the **ontology** of the '**vehicle system**'

- Each service provides objects **not tied to the specifics of each vehicle**

- E.g., Traffic service provides "MovingObject" class, instead of "CameraData", or "LIDAR-point-cloud-data", to hide the specific physical sensors used

- This model provides the '**abstraction**' essential to enable another principal 'Decoupling'

Platform Services

Driver
- Behavioral model
- Psychological model

Vehicle
- Motion
- Interior
- Exterior
- VehicleDashboard

Environment
- Road infrastructure
- Traffic
- Atmosphere

Utility
- Software Update
- Gateway
- Diagnostics
- Configuration vault

Software platform architecture with DVE+U

Realtime →

Body   Powertrain   Chassis

Automated drive	Co-pilot drive	Master controller	Safety manager	Environment communicator	Driver Communicator	…				

Platform API

Behavioral model	Psychological model	Motion	Interior	Exterior	Road Infrastructure	Traffic	Aomosphere	Configuration Vault	Diagnostics	Software Update
									System Health	Gateway

AUTOSAR Runtime for Adaptive applications - ARA

POSIX PSE51 / C++ STL Operating System Interface	Platform Foundation Functional Clusters (FCs)				Platform Service FCs	Standardized App/Interface FCs	Vehicle Service FCs
	Communication Management ara::com	Execution Management ara::exec	Diagnostic Management ara::diag	Intrusion Detection System Manager ara::idsm	Update and Config Management ara::ucm service	Sensor Interfaces ara::adi::sensoritf service	Vehicle Update and Config Management ara::vucm service
	Network Management ara::nm	Persistency ara::per	Time Synchronization ara::tsync	Cryptography ara::crypto	State Management ara::sm service		
	Raw Data Stream ara::rds	Platform Health Management ara::phm	Log and Trace ara::log				
	Firewall ara::fw	Core ara::core		⋮	⋮	⋮	⋮

Multi-kernel (POSIX OS)	Common driver/protocol stacks, utilities	OS	OS	OS

BSP (hardware dependent layer – physical device drivers, HW PF specific libraries)	MCAL	MCAL	MCAL

Multi-chip-nodes Heterogeneous hardware

Cloud   **Infotainment**

RTE   RTE   RTE

BSW   BSW   BSW

Right-side layer labels:
- Application layer
- Service layer
- Foundation layer
- OS layer
- Hardware

eSOL

System modeling for software decoupling the hardware

- Applying the DVE model for the SDV Platform design to decouple the hardware (machines/electrics)

- A key factor in SDV is **"softwarization" of mechatronics hardware enabling simulation**/virtual hardware
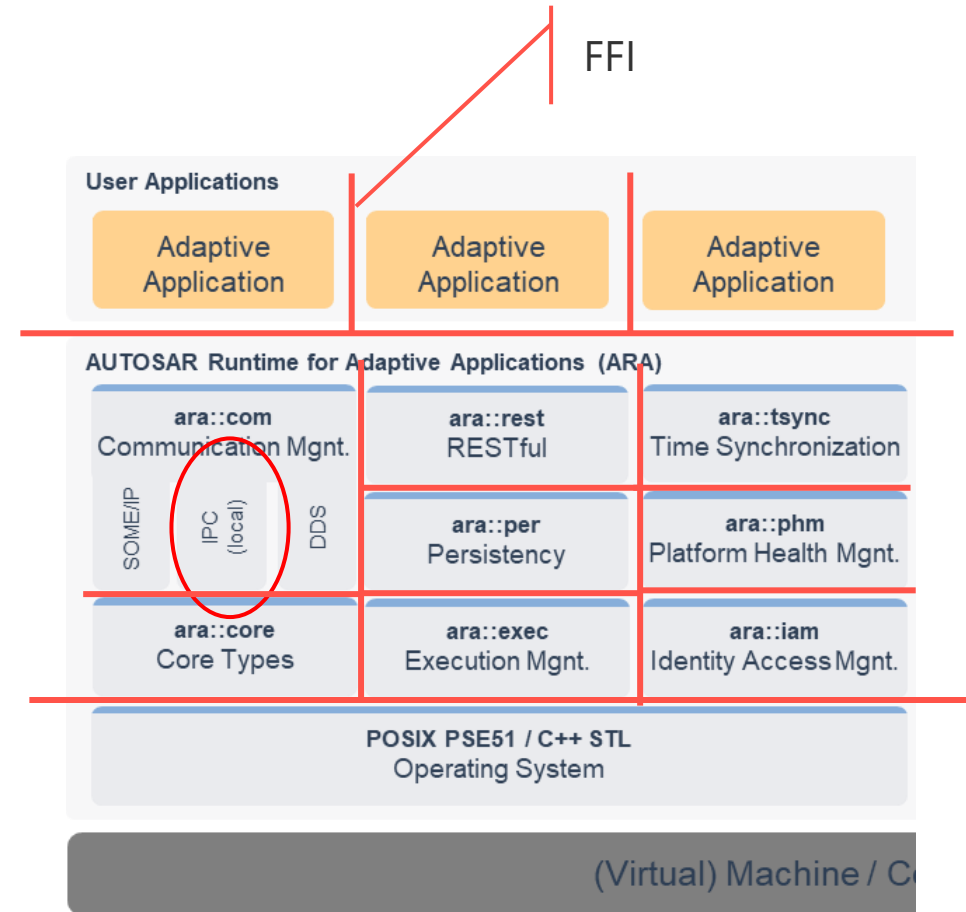
- **DVE allows encapsulating the hardware while serving as the API model** of SDV Platform
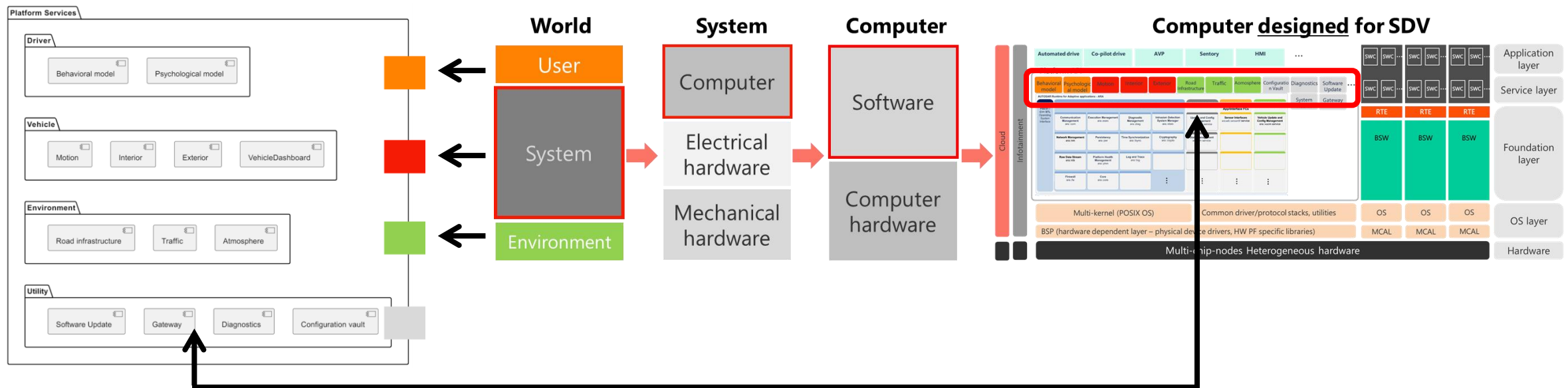
Multi-kernel OS

IPC is critical for AUTOSAR Adaptive Platform

- Adaptive Platform has the architecture with **FFI** in mind
    - **Adaptive Applications and Functional Clusters are implemented as OS processes, ensuring FFI**
    - The OS Interface (OSI) of ARA is POSIX PSE51
        - ➤ Single-process profile
        - ➤ PSE51 provides no IPC, no fork/exec/spawn
    - Only way to explicitly communicate to other processes is **via ara::com with its IPC binding**
    - Functional Clusters can use any OS interfaces
- **Performance and scalability of IPC is critical**
    - Platform heavily depends on the compartmentalization of software entities with OS processes

FFI

User Applications

| Adaptive Application | Adaptive Application | Adaptive Application |

AUTOSAR Runtime for Adaptive Applications (ARA)

ara::com Communication Mgnt.	ara::rest RESTful	ara::tsync Time Synchronization
SOME/IP · IPC (local) · DDS	ara::per Persistency	ara::phm Platform Health Mgnt.
ara::core Core Types	ara::exec Execution Mgnt.	ara::iam Identity Access Mgnt.

POSIX PSE51 / C++ STL
Operating System

(Virtual) Machine / C

IPC: Inter-Process Communication

The changes in the hardware architecture

- It has been **20 years since ARM 11 MPCore** is introduced to the market

- Now the clusters of cores (or more) totaling 10, 12, 16, 32, **even 256 cores are available**

- This trend is inevitable, due to **the end of Dennard Scaling** and **Pollack's rule** becoming more relevant the only way to scale up the performance is not by increasing the frequency but by the number of cores

- This will be coupled with programmable **accelerators/coprocessors**, resulting in **heterogenous-manycore compute to meet the HPC** challenge

- **Now** all these processor cores and accelerators will be interconnected by super-fast and high-bandwidth **NoC** or similar, and soon with multiple **chiplets, photonics, etc.**

- This whole set of **changes in processor/interconnect architecture calls for a new architecture of OS and IPC**

eSOL eMCOS® multi-kernel OS

- **The world first commercial multi-kernel OS** supporting **homogeneous and heterogeneous cores with one OS**

- **The independent kernel per core** provides **a strong architectural FFI foundation** and **fast and scalable communication** via inter-microkernel messaging

- Supports **both POSIX API** for AP and **AUTOSAR OS API** for CP
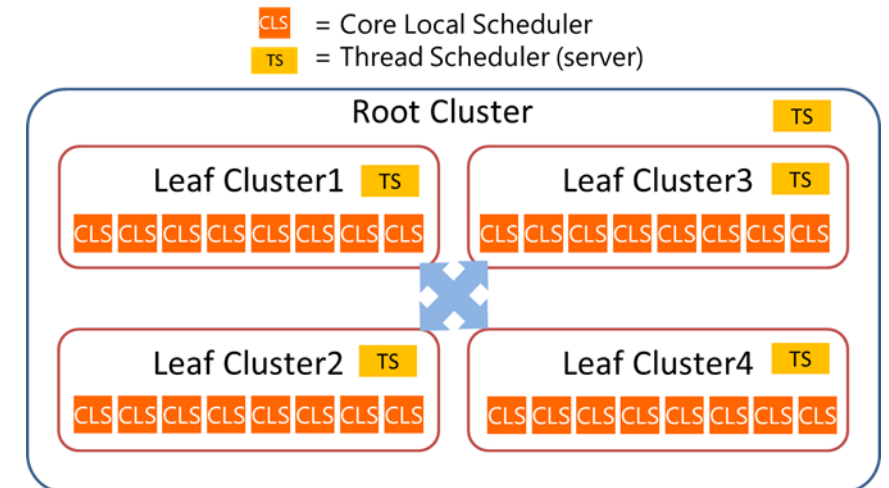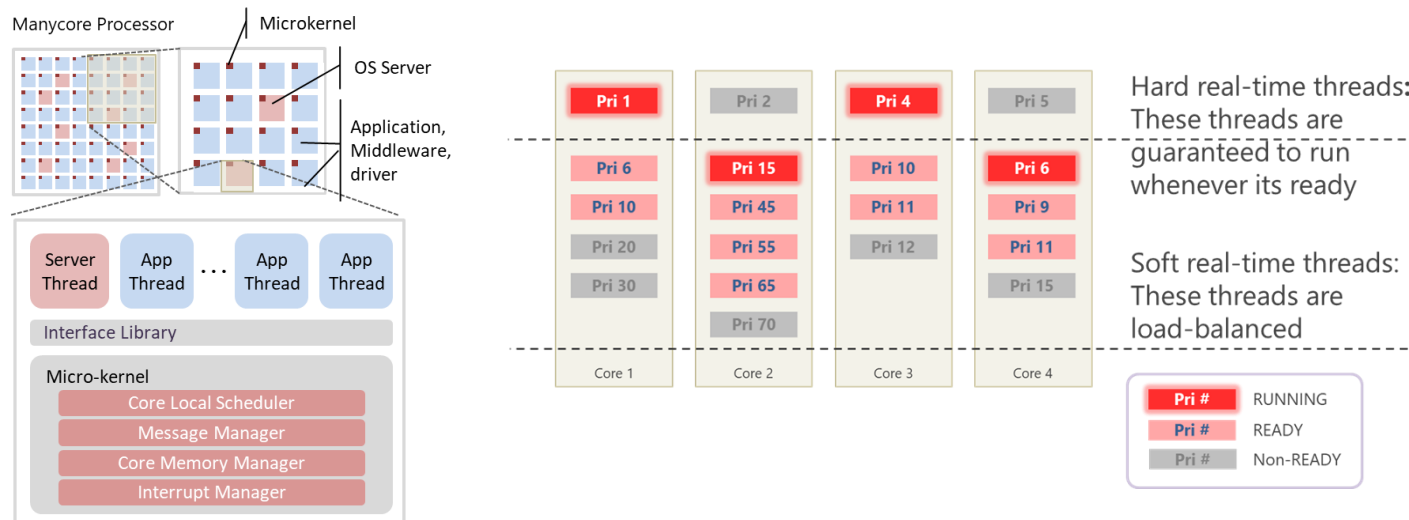
Realizing a good computer by designing the detail from a holistic view
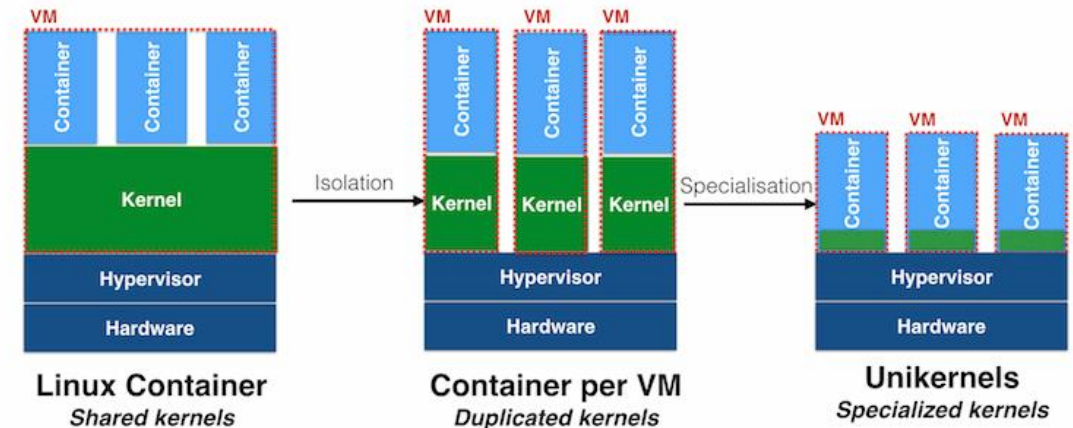
- ## Design it by going beyond the layers

Service Oriented Communication stack example

HV+OS+Container example
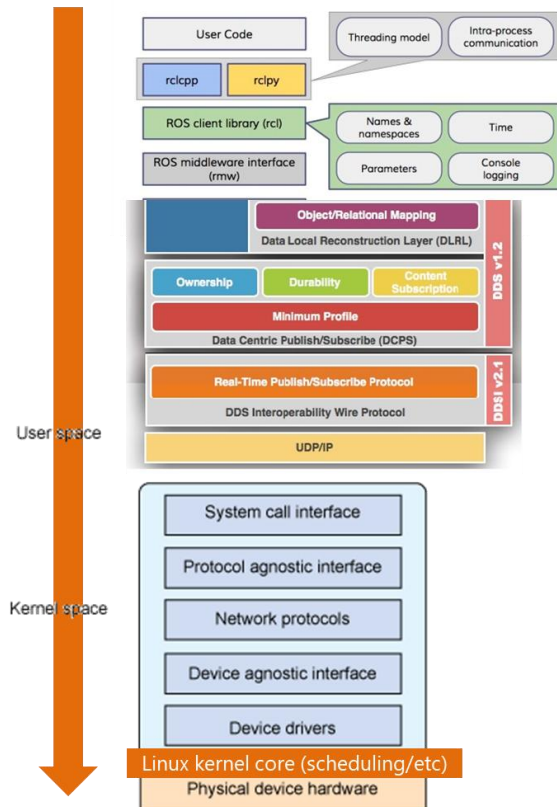
- **ROS 2** stack→**DDS** stack→**Linux kernel** networking stack – **totaling to 15 layers**

- If the kernel has the inter-core, inter-chip Service Oriented Comm., **this can be cut down into 5 layers**

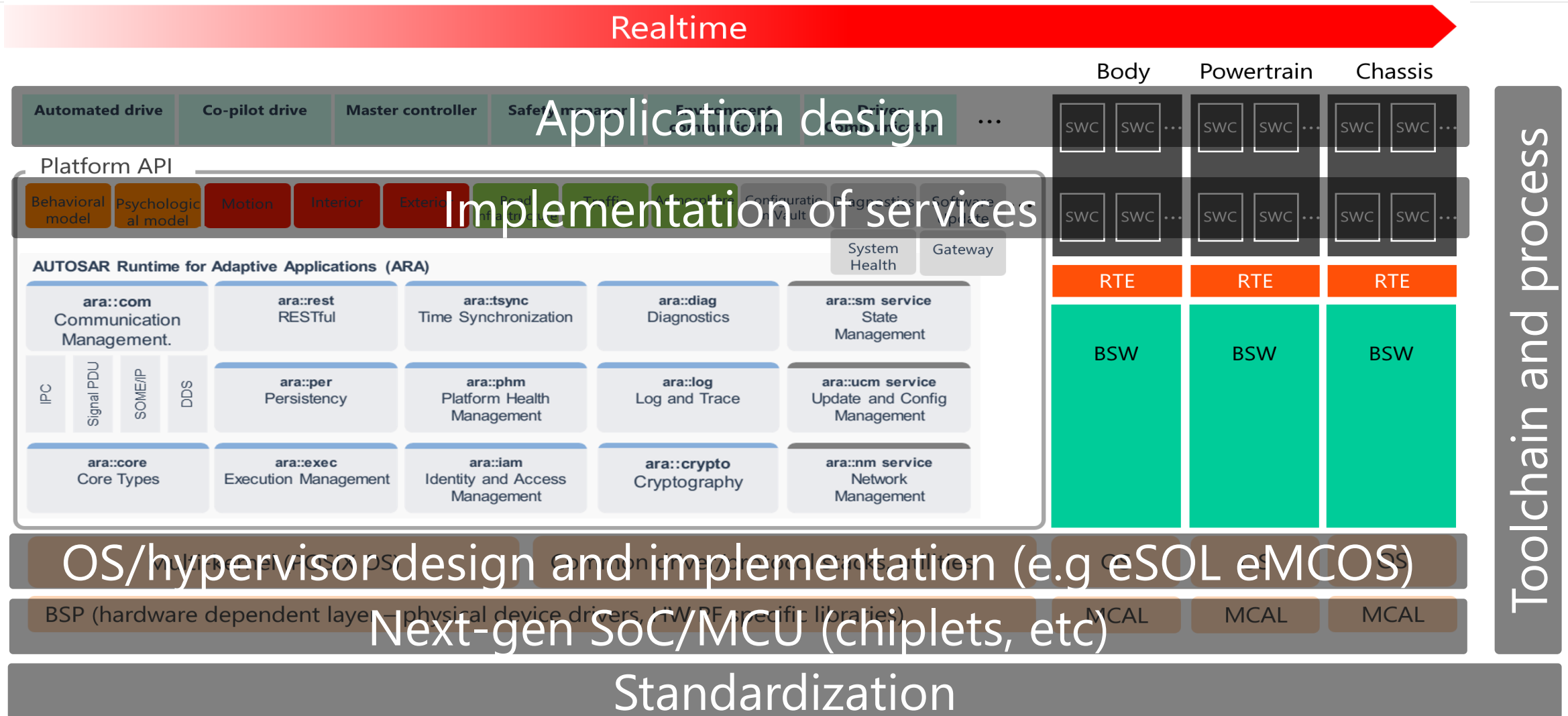- In the cloud HPC, the trend is **Hypervisor ⇒ Container+Linux ⇒ Unikernel/Library OS**

- Looking this architecture from a processor view, this is essentially user-land intensive multi-process OS

- ⇒ **A multi-kernel combined with unikernel**

Innovations in various layers shall come with SDV

Realtime

Body   Powertrain   Chassis

Application design

Implementation of services

Platform API

AUTOSAR Runtime for Adaptive Applications (ARA)

ara::com Communication Management.	ara::rest RESTful	ara::tsync Time Synchronization	ara::diag Diagnostics	ara::sm service State Management
IPC / Signal PDU / SOME/IP / DDS	ara::per Persistency	ara::phm Platform Health Management	ara::log Log and Trace	ara::ucm service Update and Config Management
ara::core Core Types	ara::exec Execution Management	ara::iam Identity and Access Management	ara::crypto Cryptography	ara::nm service Network Management

RTE   RTE   RTE

BSW   BSW   BSW

OS/hypervisor design and implementation (e.g eSOL eMCOS)

BSP (hardware dependent layer – physical device drivers, HW-DF specific libraries)

MCAL   MCAL   MCAL

Next-gen SoC/MCU (chiplets, etc)

Standardization

Toolchain and process
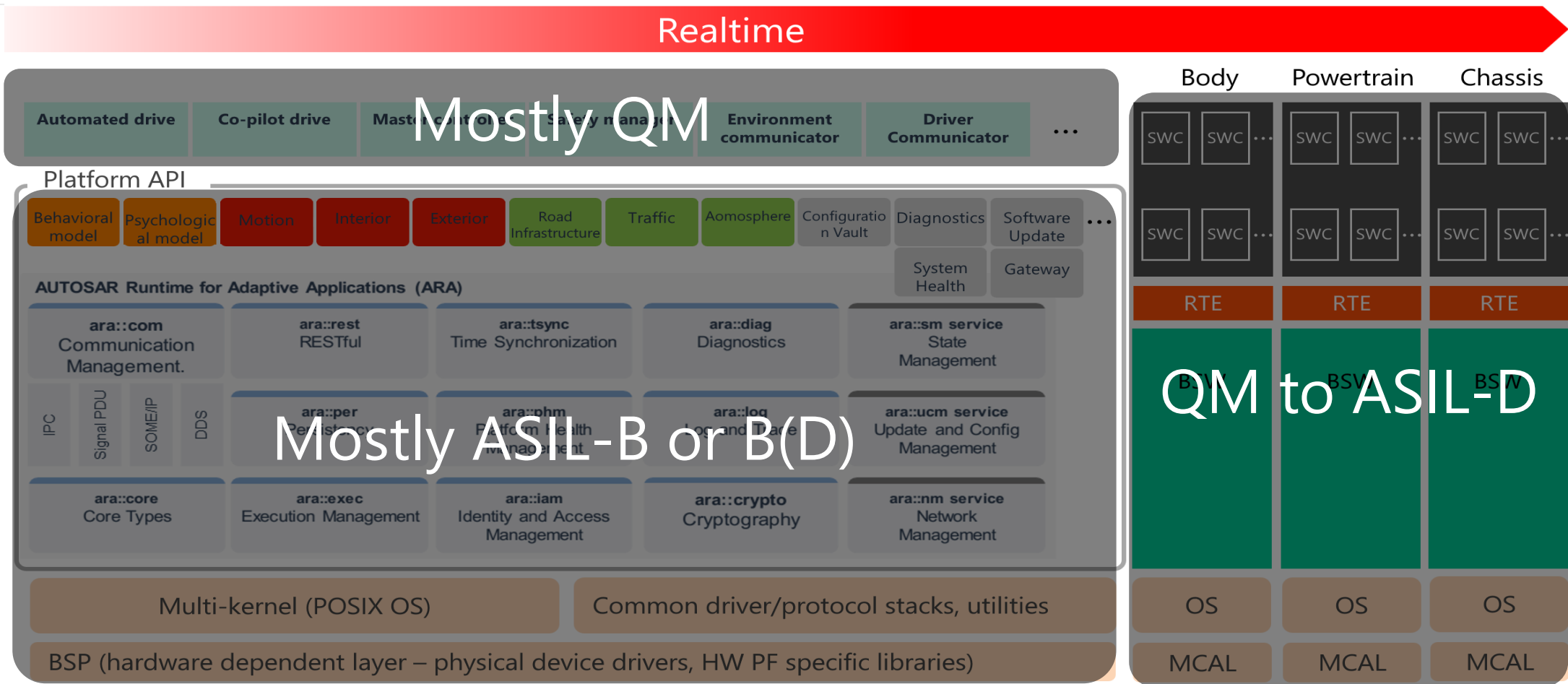
Challenge with Passion

Criticality mix and separations

Decoupling of mechanical/electrical and E/E arch

Realtime

Run on vehicle generic PF API (DVE + U)

Offer services (PPorts) over VFB and implements the hardware specifics

Mix of callouts to CP services and hardware specific implementation