SOAFEE

Panasonic AUTOMOTIVE

Transforming Automotive Edge to a Software-Defined Platform

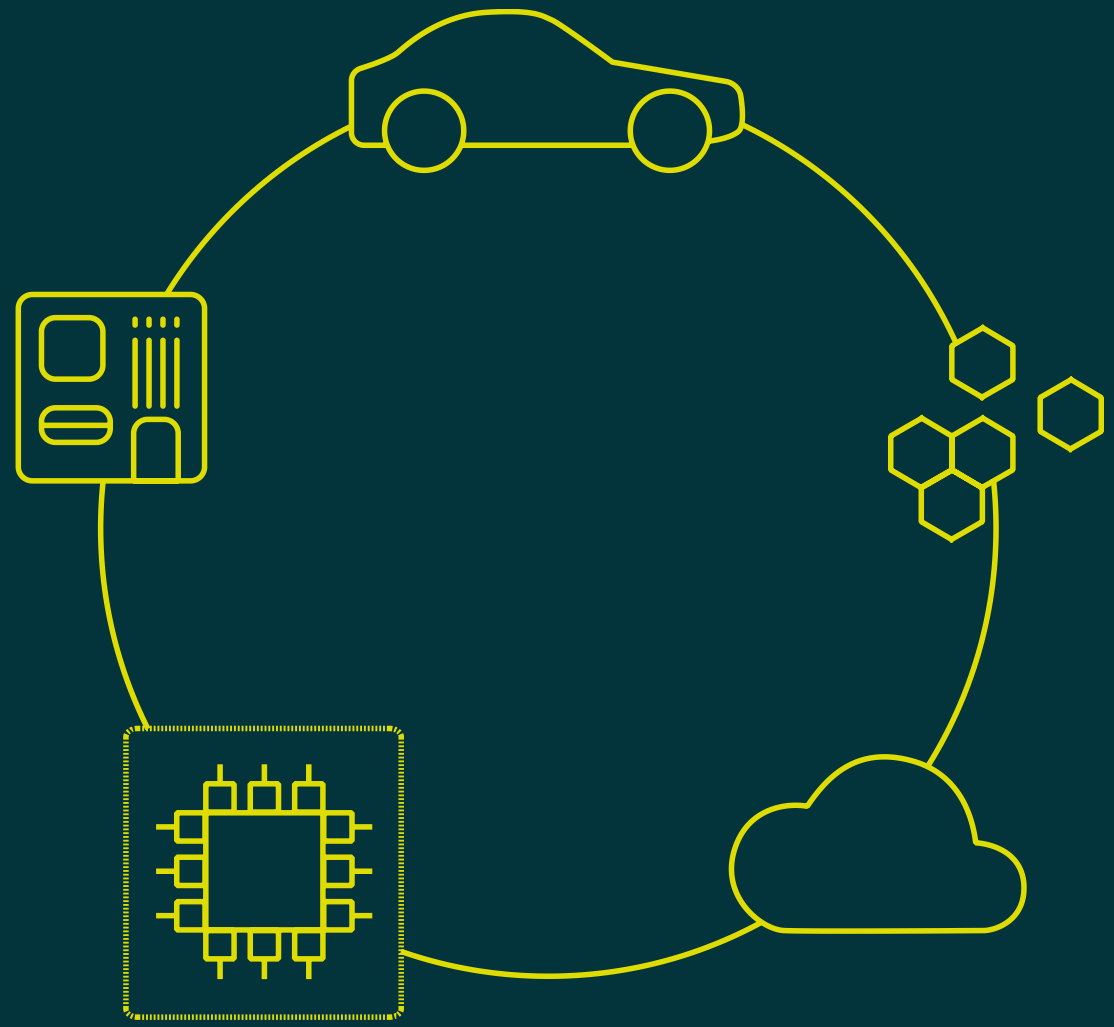
The Role of VirtIO Based Device Virtualization

Jerry Zhao, Chief Engineer at Panasonic Automotive Systems Co., Ltd

2024/5/9

SOAFEE

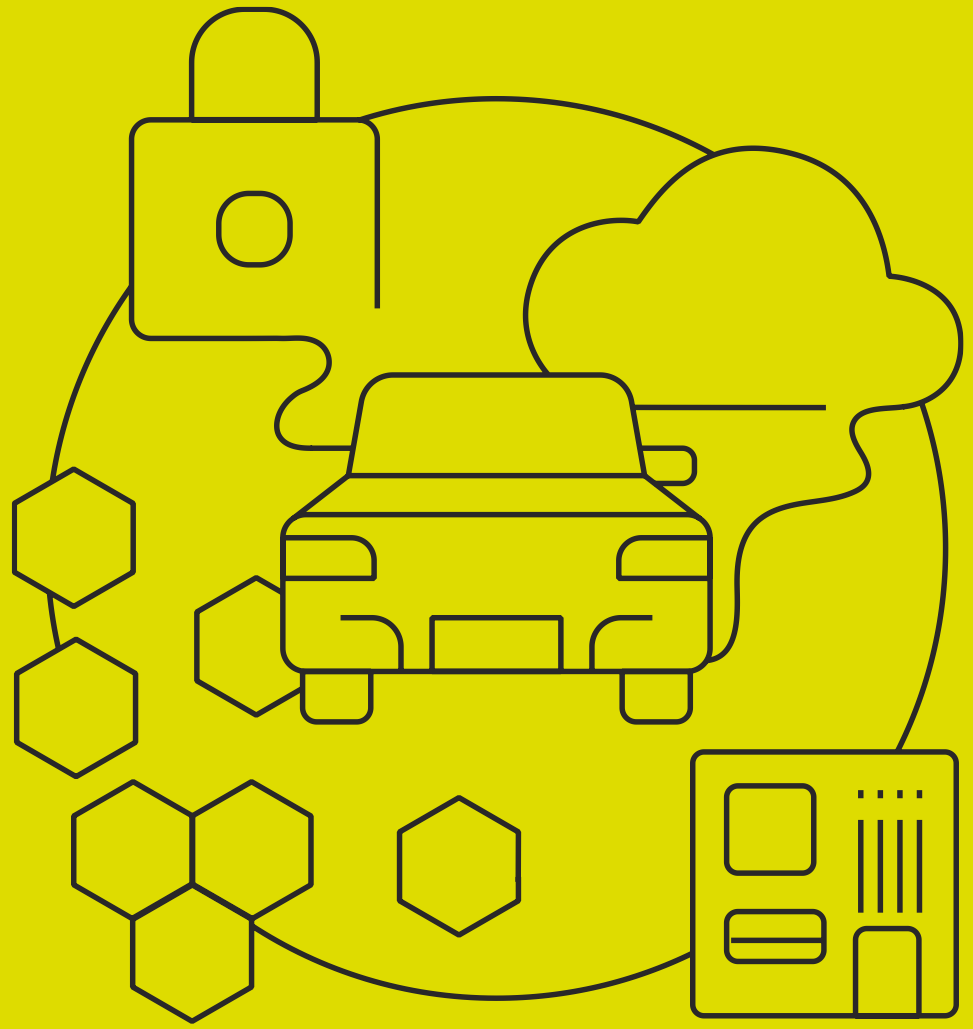
- Why: Industry Trends with Software-Defined Vehicles
- What: Architectural Changes in the Automotive World
- How: Decoupling Software from Hardware with Device Virtualization
- To where: Conclusion & Overlook - Constructing a bright and open future of SDV with SOAFEE



Panasonic AUTOMOTIVE

SOAFEE

Industry Trends with Software-Defined Vehicles (SDV)



Panasonic AUTOMOTIVE

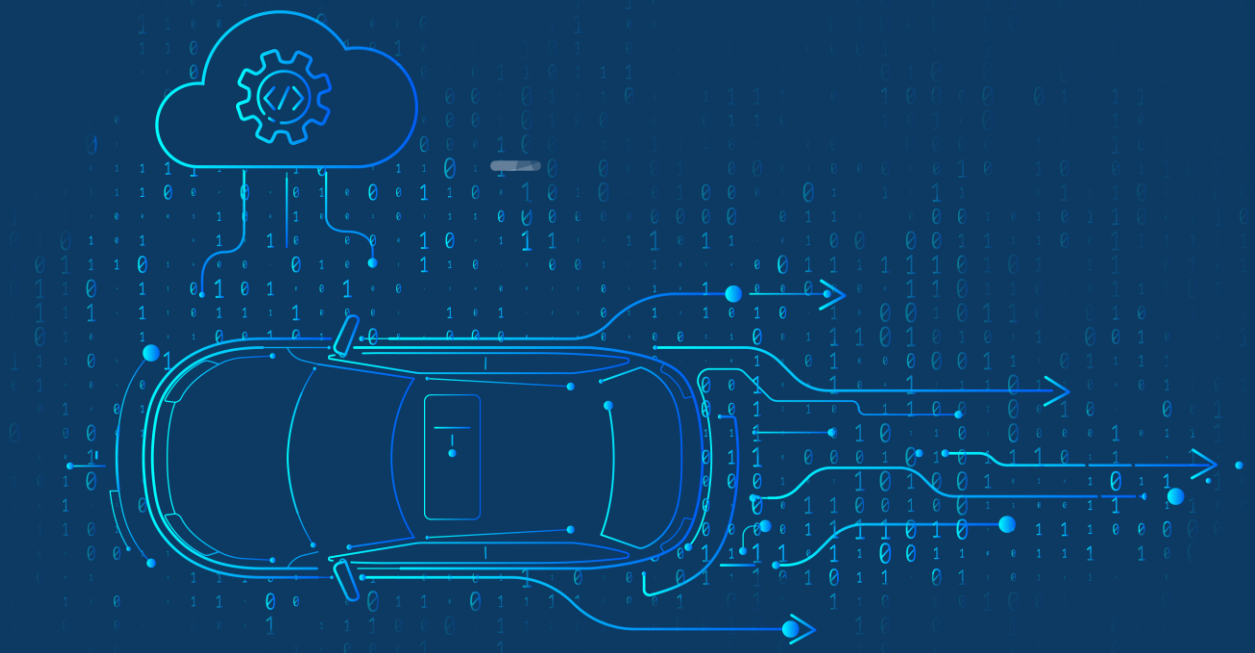
Industry Trend with SDV

Software-Defined Vehicles

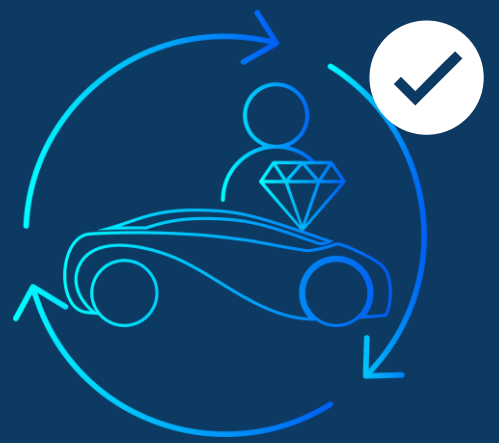
Drastic Changes
in EE Architecture

Open Source,
De Facto Standard

Connected
& AI-powered

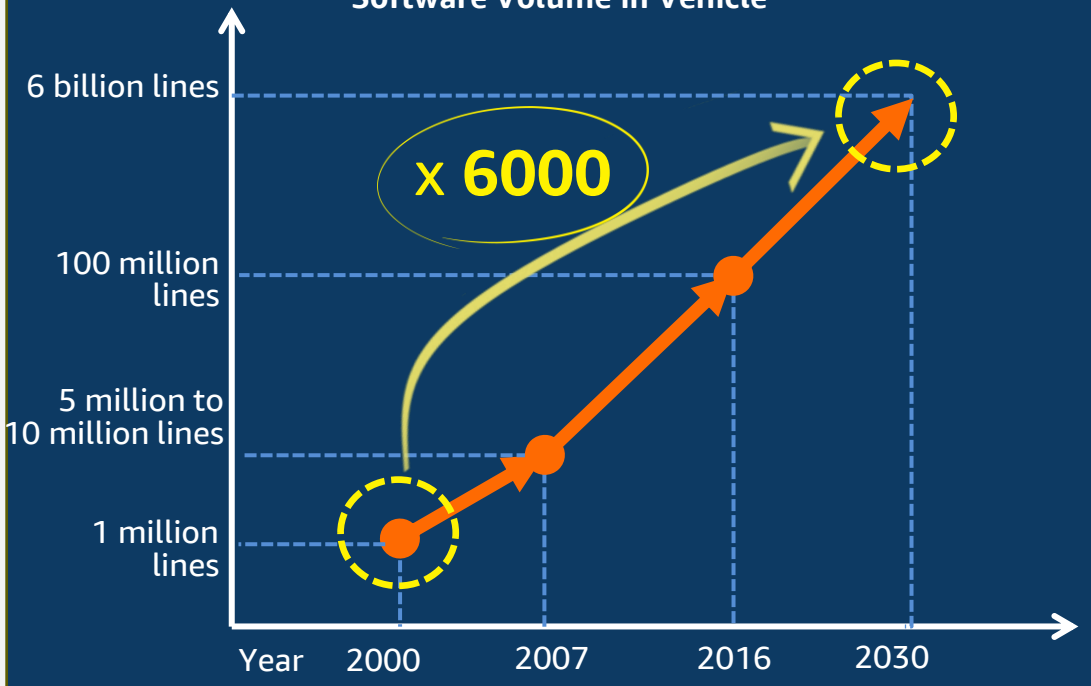


Speedily Delivered Values



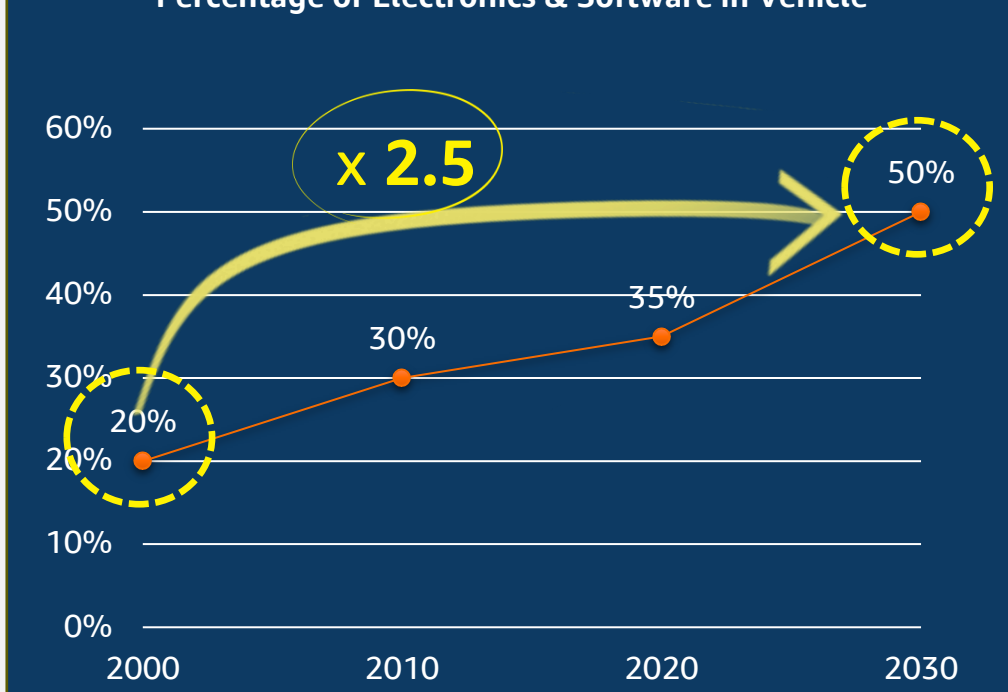
Shift to SDV Industry Trend

Increased Complexity of Vehicle Software (*1)



*1: Source: Ministry of Economy, NXP Semiconductors, Quora, Ignition in Action, NYC AVITAION, Trade and Industry "Toward acceleration of productivity improvement by IT" Mitsubishi UFJ Morgan Stanley Securities' materials, etc.

Increased Cost Contribution of Vehicle Software (*2)



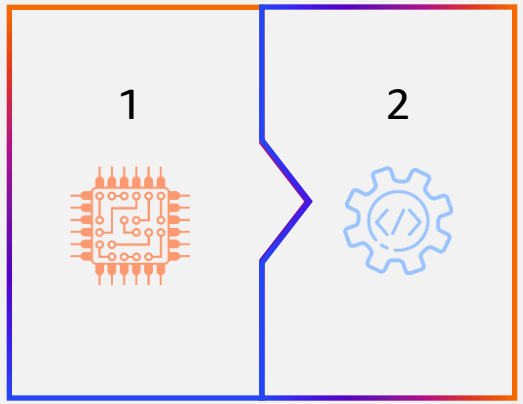
*2: Source: Lux Research

Panasonic AUTOMOTIVE

Automotive Industry Game Changer

Shift in Key Strategies

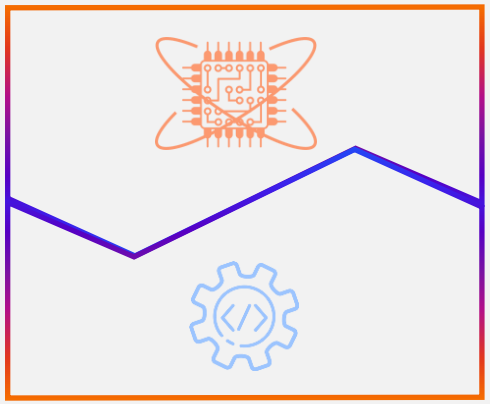
From Hardware First To Software First



Traditional

Manufacture HW prototype and develop SW

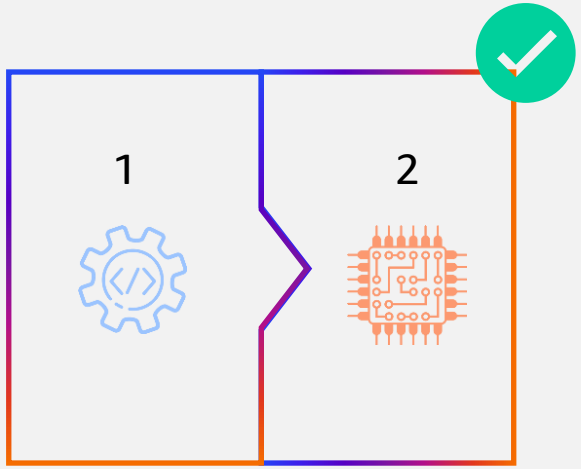
- Long wait time for limited HW
- High sample cost



HW Emulation

Emulate HW and develop SW simultaneously

- Limited to low-level SW & HW
- Costly & time-consuming



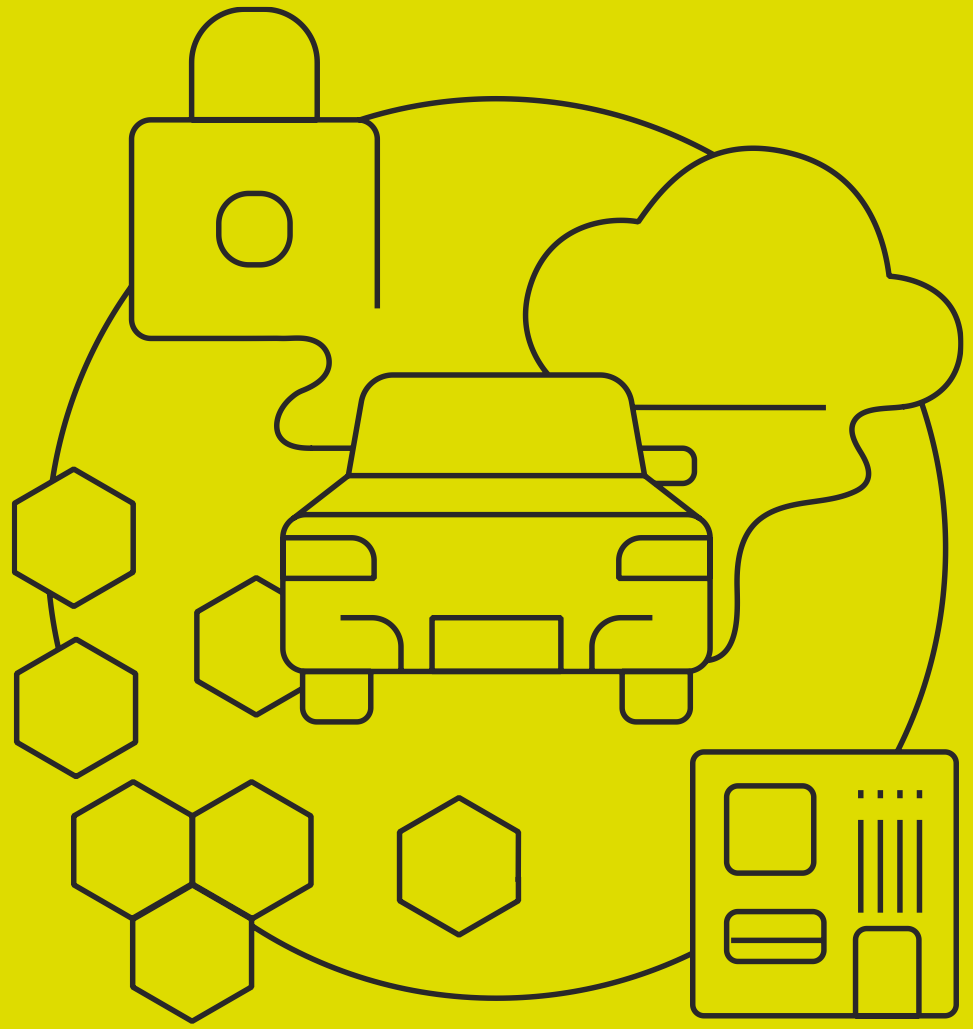
Cloud-Native

Develop SW on Cloud and select optimal HW

- Rapid function update
- Scalable for large-scale development

SOAFEE

Architectural Changes in the Automotive World

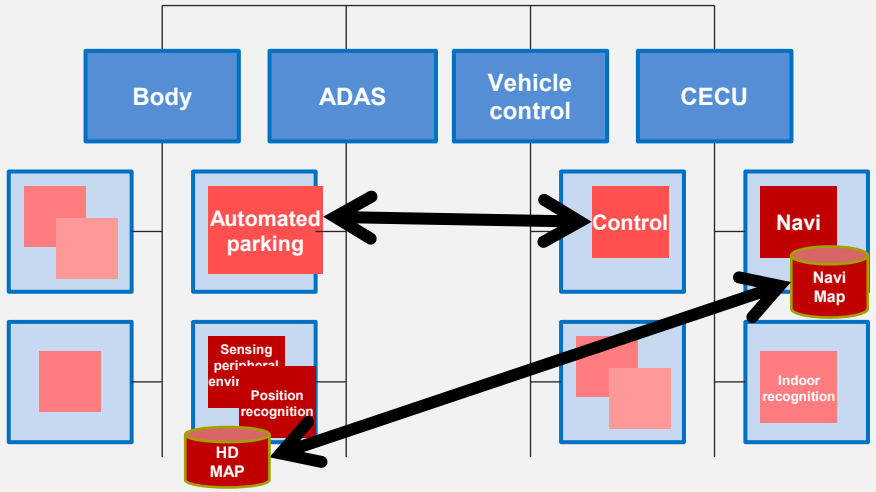


Panasonic AUTOMOTIVE

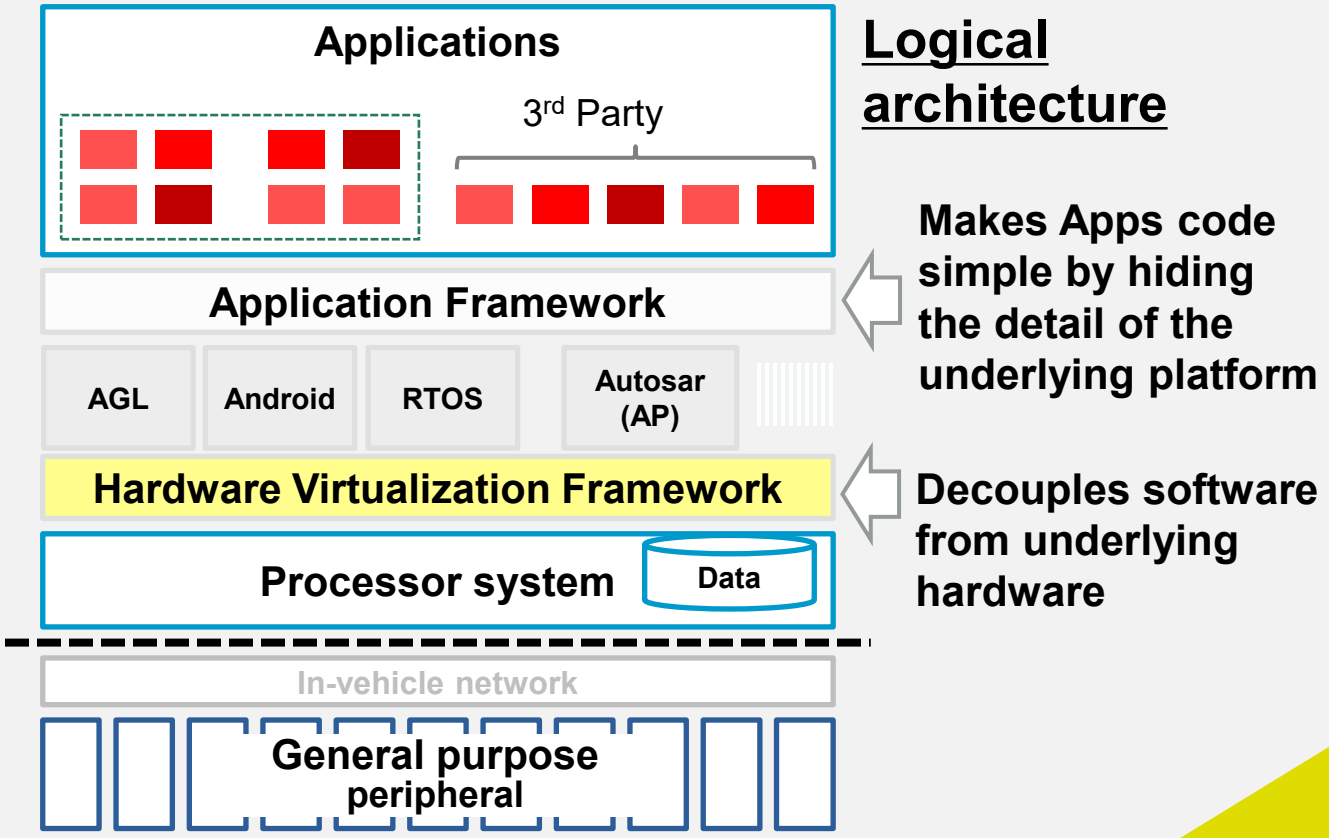
Desirable Direction of Automotive System Architecture

ECU consolidation is not a purpose but means --- The true purpose is to establish the optimal architecture for evolution of software.

"Those who can advance their software more rapidly will gain crucial competitive advantage."



Advancement of technology and updates are difficult.
Overlap of computing resources is an issue also.



Logical architecture

Makes Apps code simple by hiding the detail of the underlying platform

Decouples software from underlying hardware

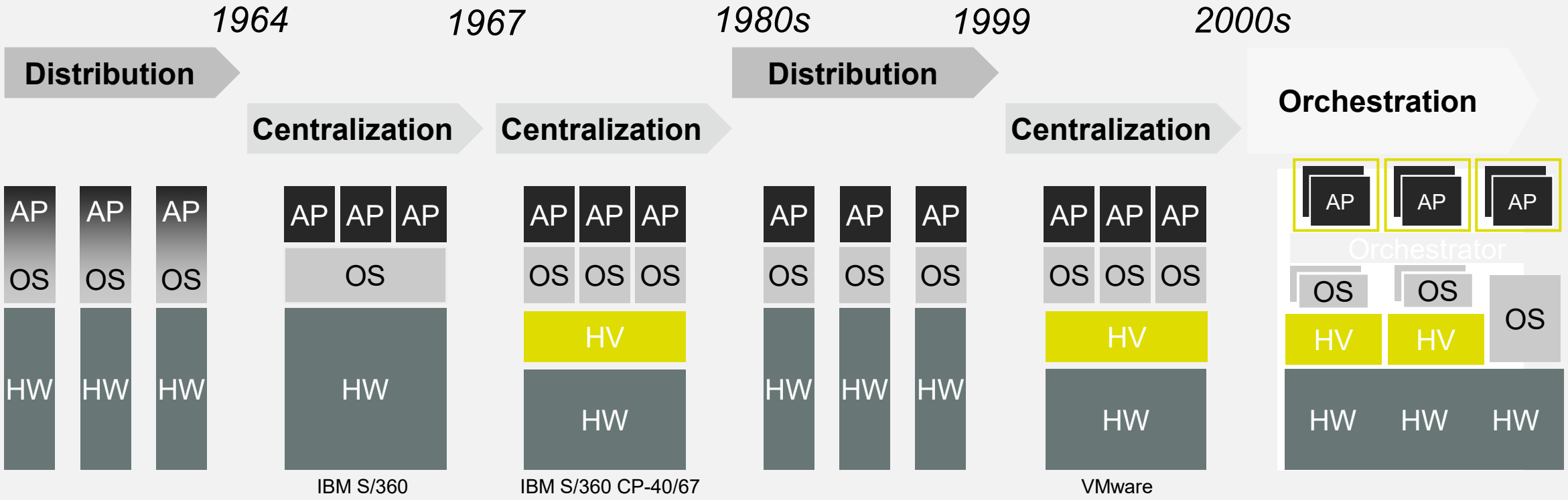
Domain Controllers

ECUs

Applications and Data

Historical Trend of General Computing Architecture (Distribution and Centralization)

The history of general computing architecture is **repeating the cycle between centralization and distribution**, and the automotive industry is following a similar path.

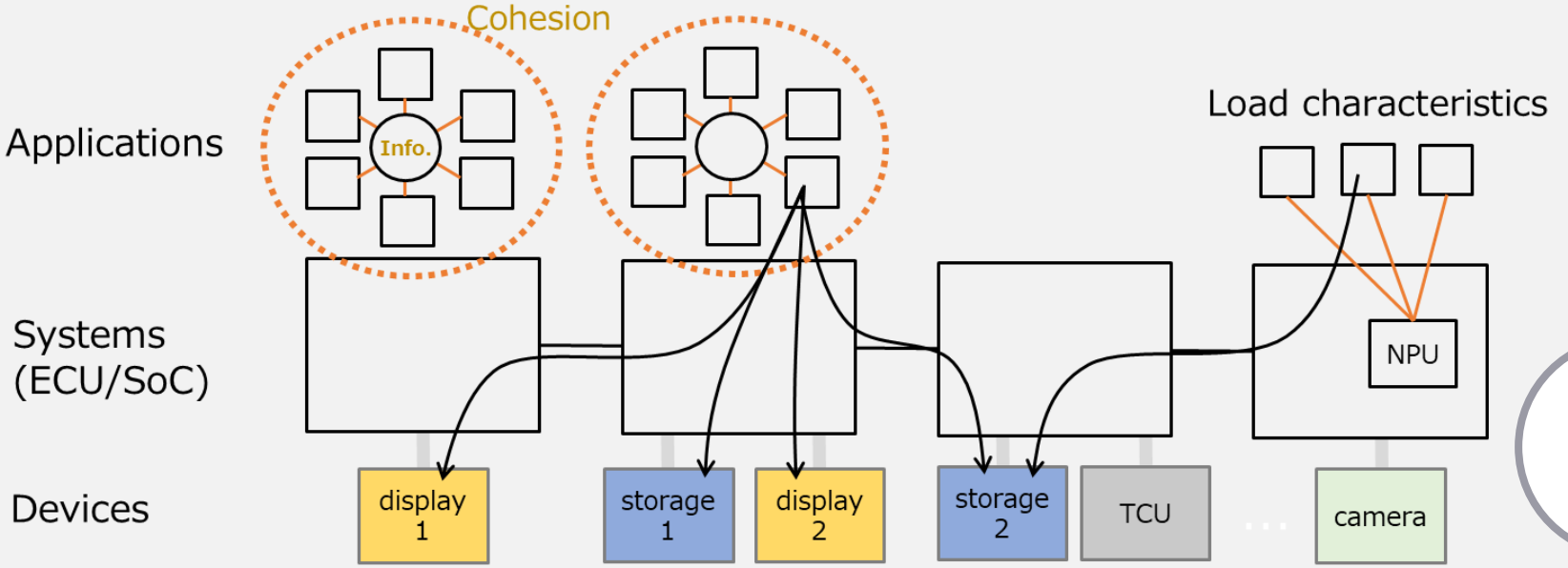


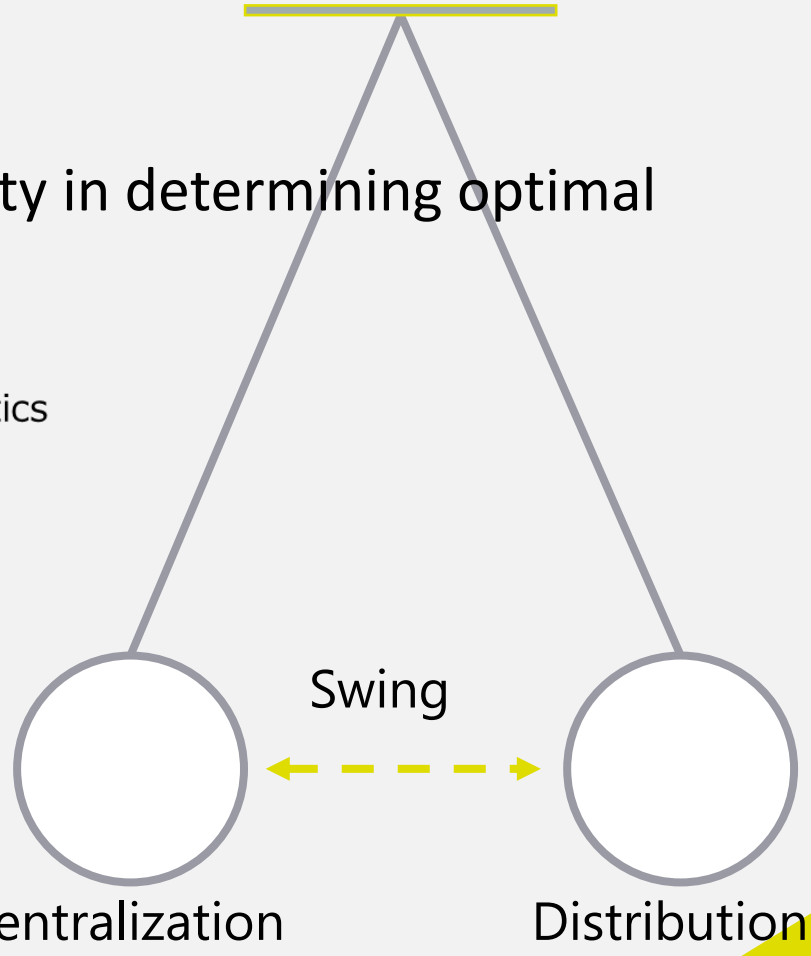
Created by Panasonic Automotive Systems referring to ITmedia IT solution cram school [Graphic explanation] History of virtualization on a single sheet
https://blogs.itmedia.co.jp/itsolutionjuku/2015/06/post_90.html

Greater Complexity in Automotive for Optimal Architecture

Complicated natures of both devices and applications make a greater complexity for automotive

- Diversity of Devices due to Various Car Models
- Allocation policies of applications and devices added difficulty in determining optimal system architecture whether distributed or centralized



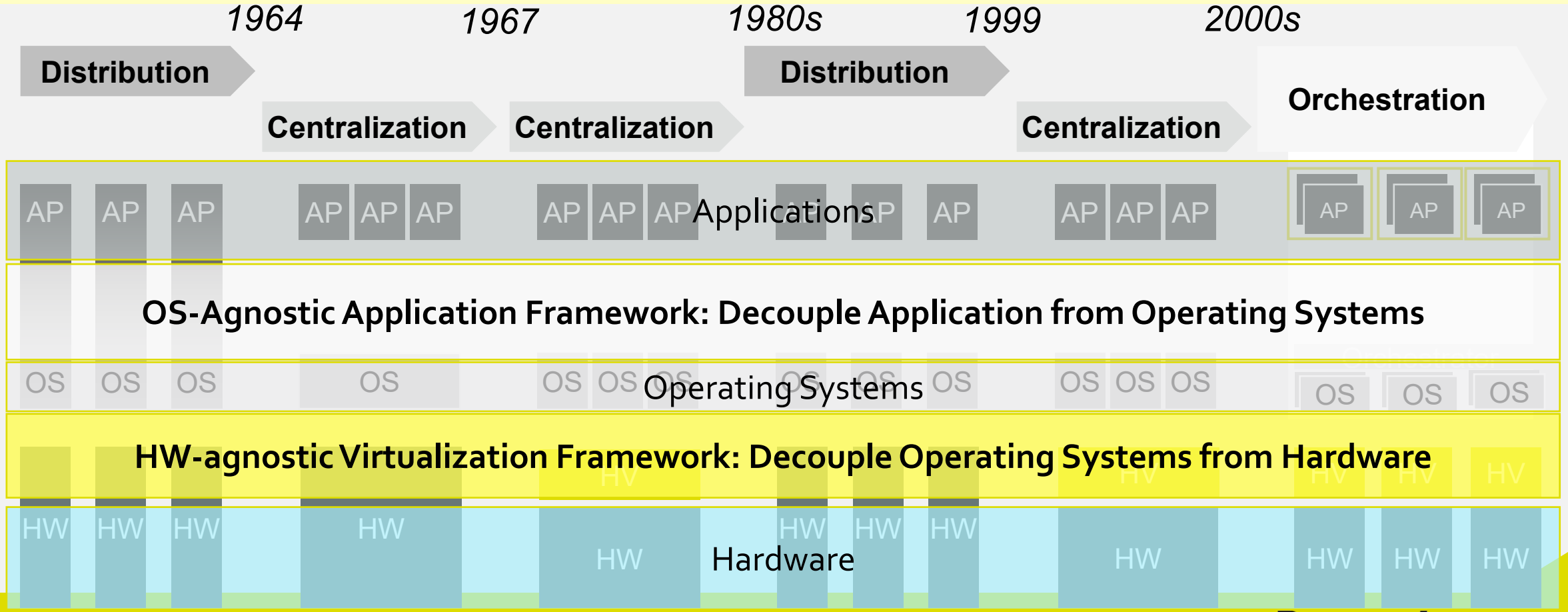


Panasonic AUTOMOTIVE

Historical Trend of General Computing Architecture (Distribution and Centralization)

No matter how the underlaying computing architecture has changed, a consistent objective is to decouple apps (directly contributed to user values) from underlying computing architecture

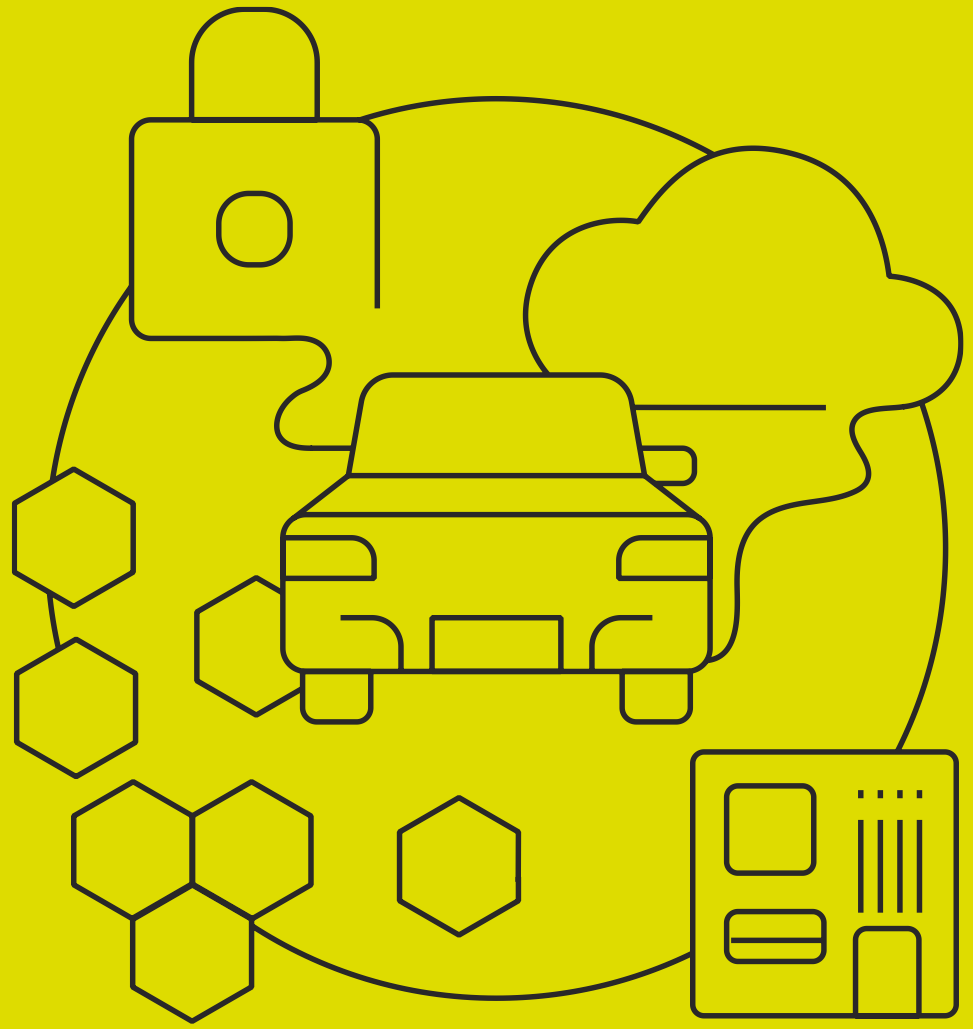
→ **An Operating-System-Agnostic Application Framework** and **a Hardware-Agnostic Abstraction Framework** are continuously to be the key to drive industry shift from hardware-centric to software-defined



Panasonic AUTOMOTIVE

SOAFEE

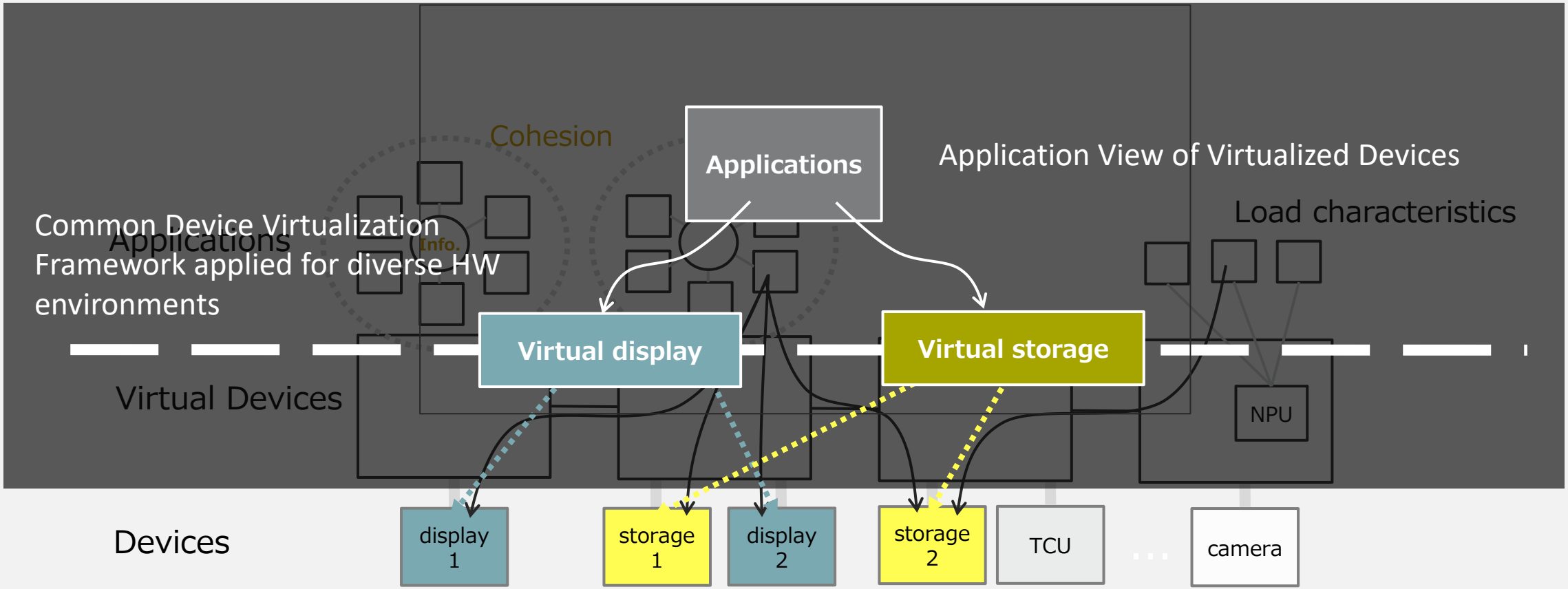
Decoupling Software from Hardware with Device Virtualization



Panasonic AUTOMOTIVE

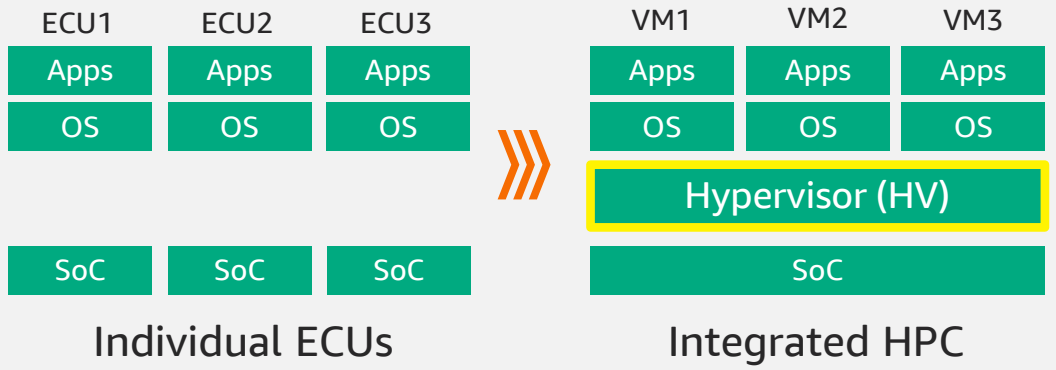
Device Virtualization: Key to Software Defined Vehicles

Software Defined Vehicle needs a common device virtualization framework to decouple software implementation from diverse hardware targets across vehicle variants/generations, architectures (single/multiple-ECU) and development environments (real/virtual ECU)

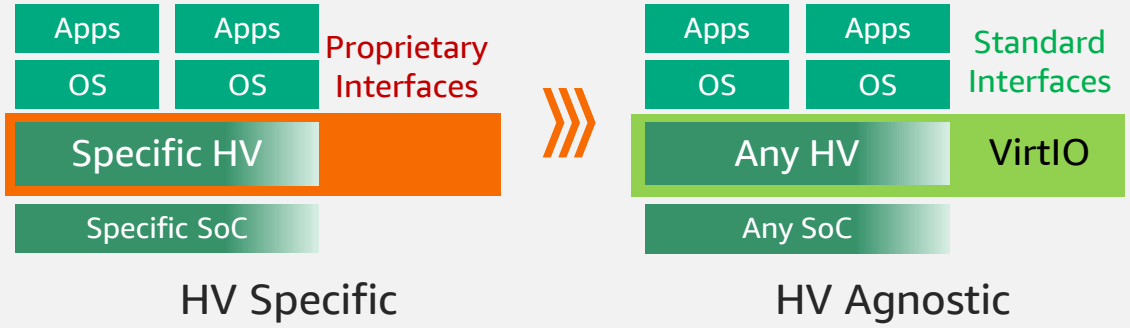


Decouple Hardware and Software

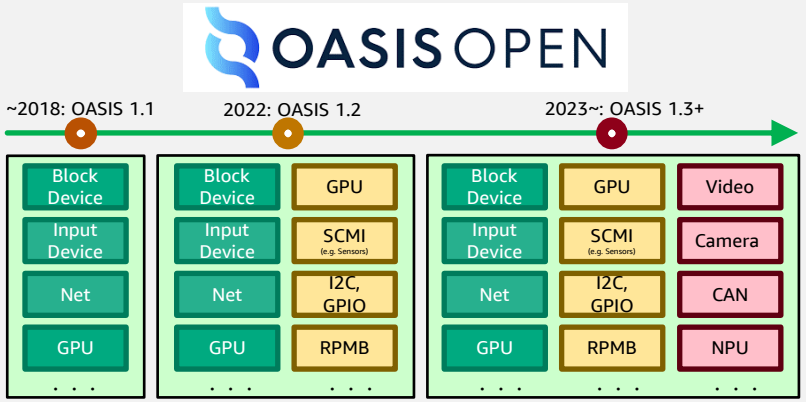
Consolidation requires Hypervisor



Hypervisor needs Standard Interfaces



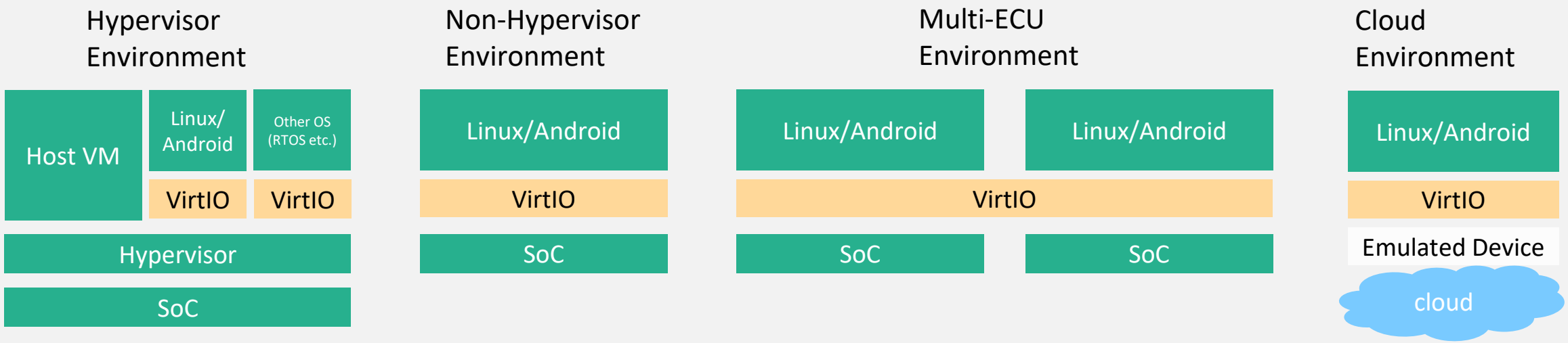
Standard Specification



Standard Implementation

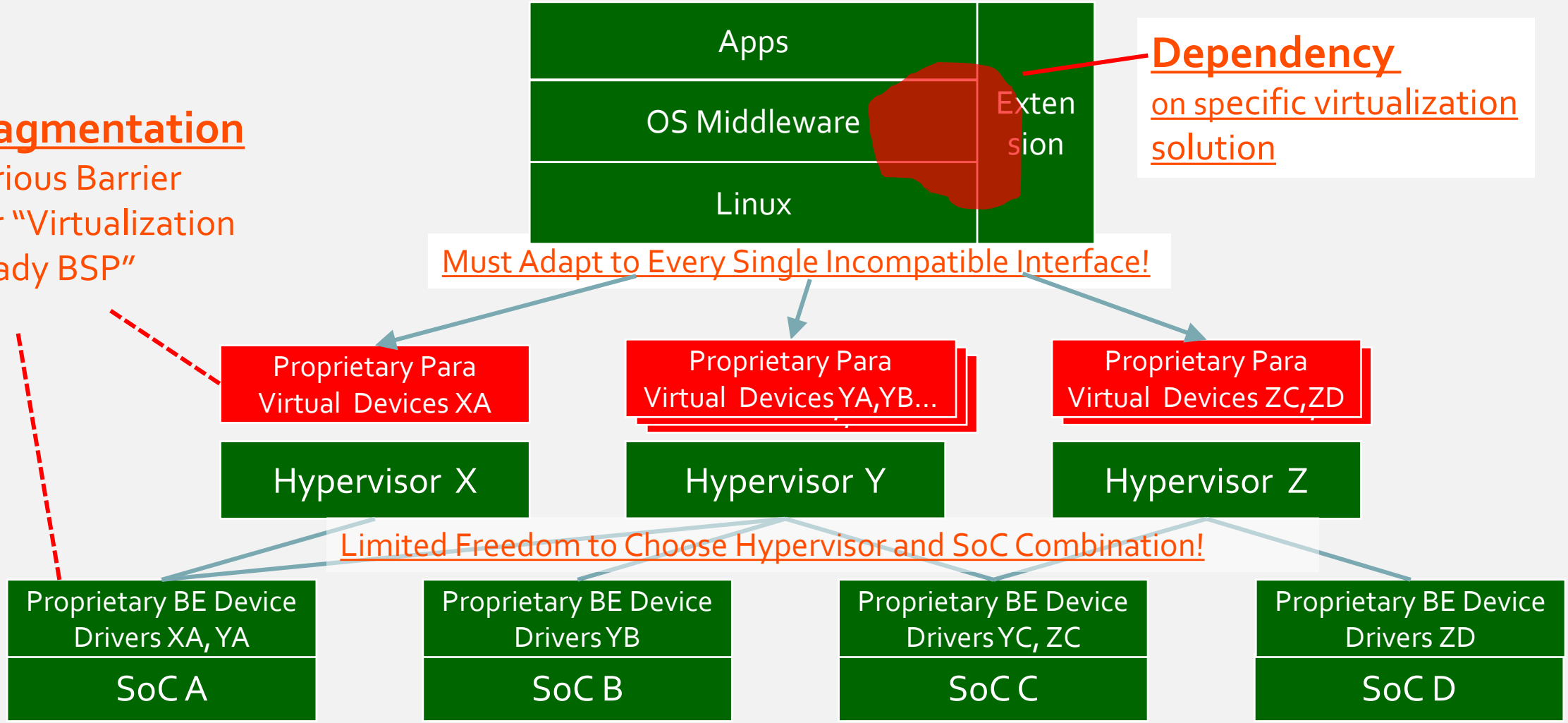
Overview of Device Virtualization - Concept

Device Virtualization with VirtIO benefits in establishing a complete and healthy ecosystem to enhance interchangeability and interoperability in various scenarios.



Pains around Peripheral Virtualization in the Past

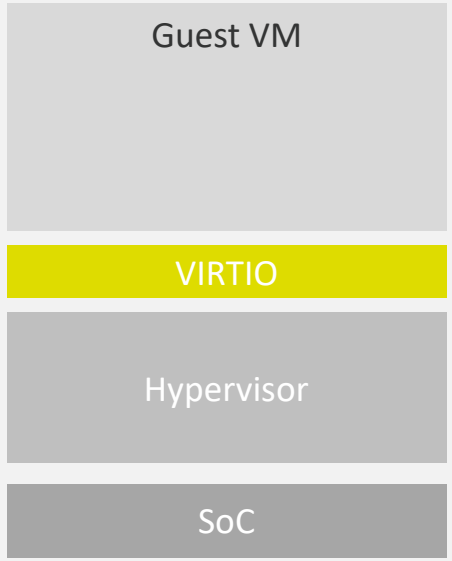
Fragmentation
Serious Barrier
For "Virtualization
Ready BSP"



*Excerpt from Panasonic's Keynote Presentation at the AGL AMM July 2020

Enter Standard Virtualization Framework - VirtIO

- Developed in 2008 as a hypervisor neutral way of accessing devices
- Provide virtual machines access to Input/Output
- A standardized interface for I/O between virtual machines and hypervisors
- Abstract device functionality instead of hardware
- Drivers are widely available in all major operating systems (Linux, Android, BSD, Windows, etc)
- Supported by all clouds and enterprise hypervisors

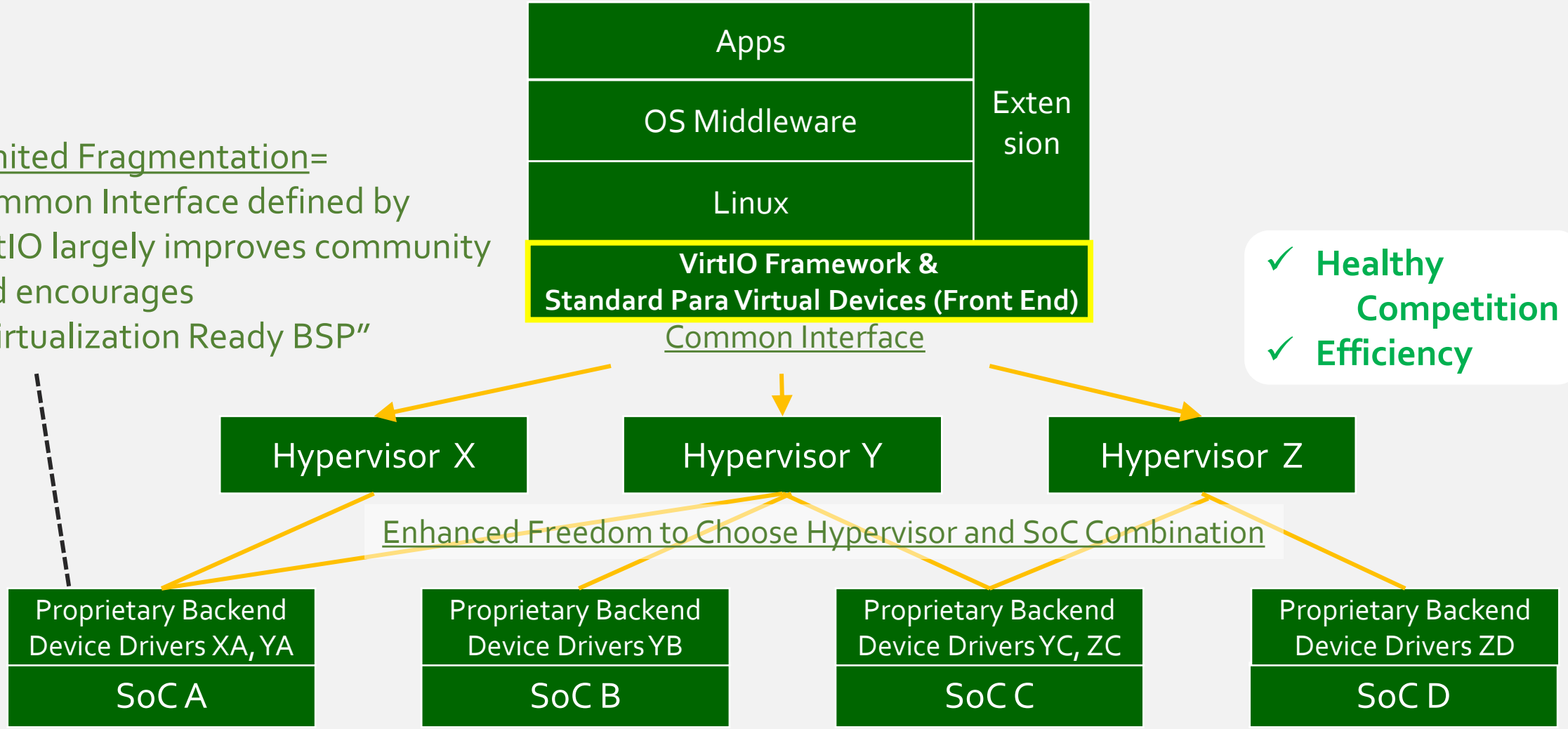


- Reliable and proven technology
- Versatile abstraction model
- Scalable and high performance
- Multiple interoperable implementations
- Broad ecosystem across multiple industries

*Excerpt from OpenSynergy Presentation at AGL F2F Oct 2022

VirtIO as a Common Framework for Virtualization

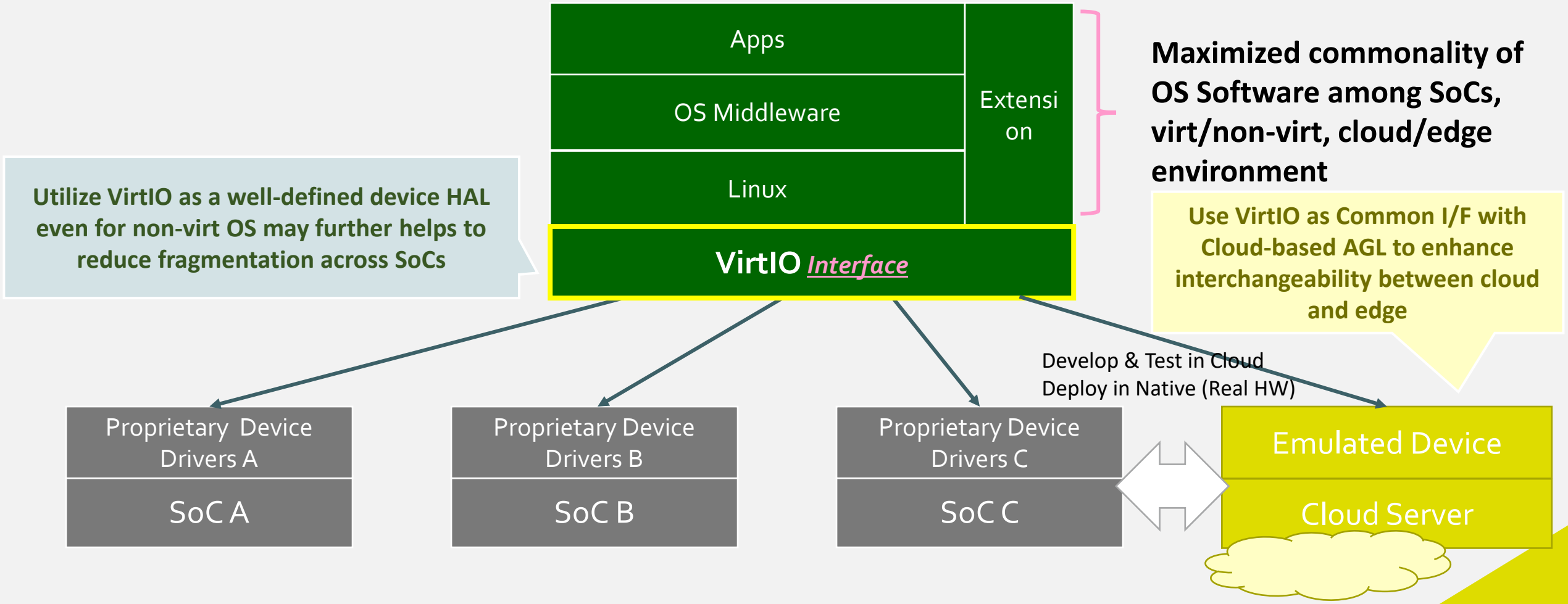
Limited Fragmentation=
Common Interface defined by VirtIO largely improves community and encourages
"Virtualization Ready BSP"



✓ Healthy Competition
✓ Efficiency

*Excerpt from Panasonic's Keynote Presentation at the AGL AMM July 2020

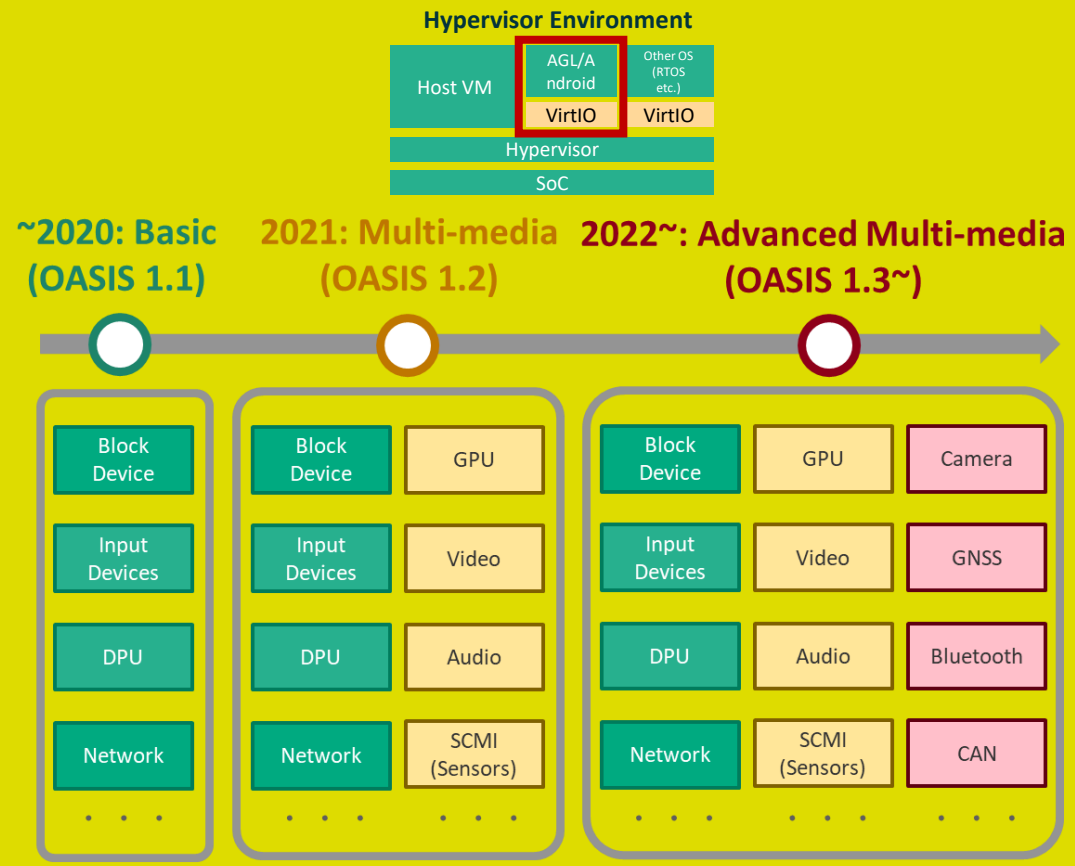
VirtIO Beyond Edge Hypervisor



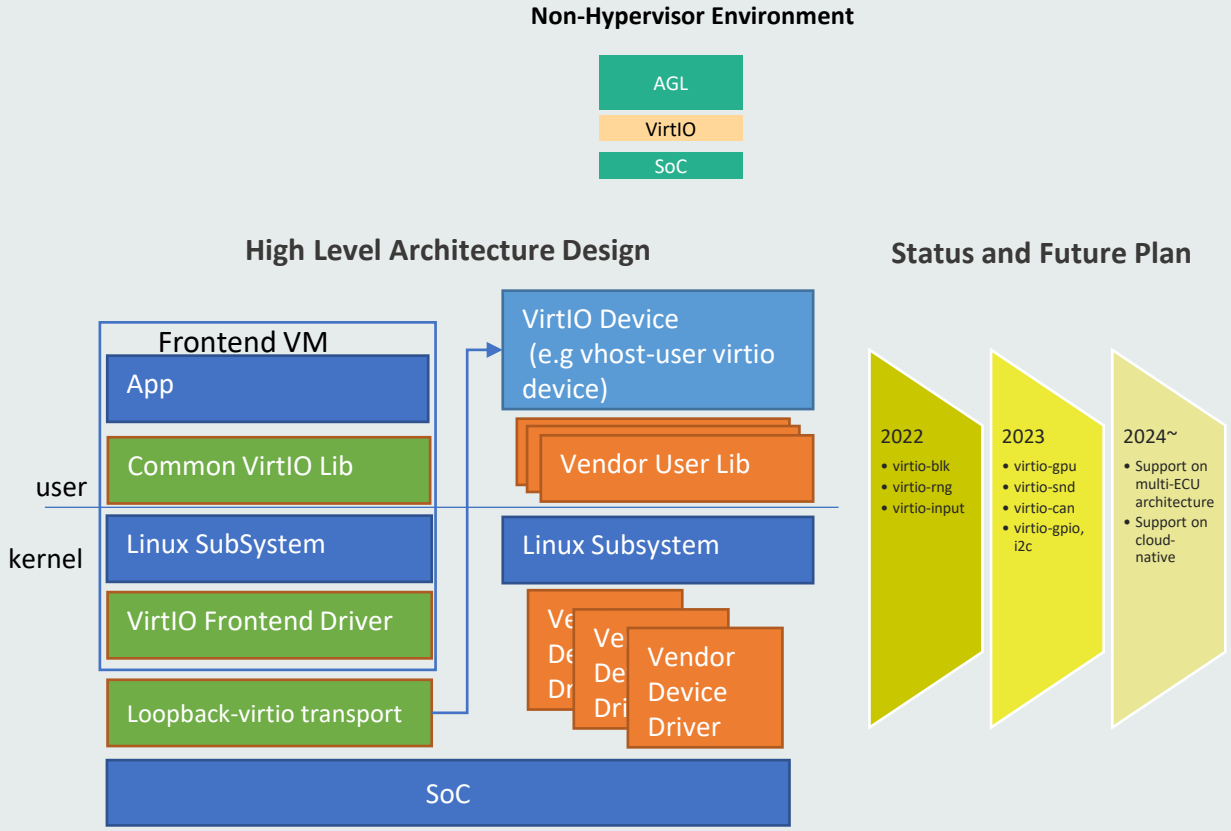
VirtIO for Edge

VirtIO for Hypervisor & Non-Hypervisor Environment

VirtIO frontend support for most of CDC use cases have been achieved in AGL & Android



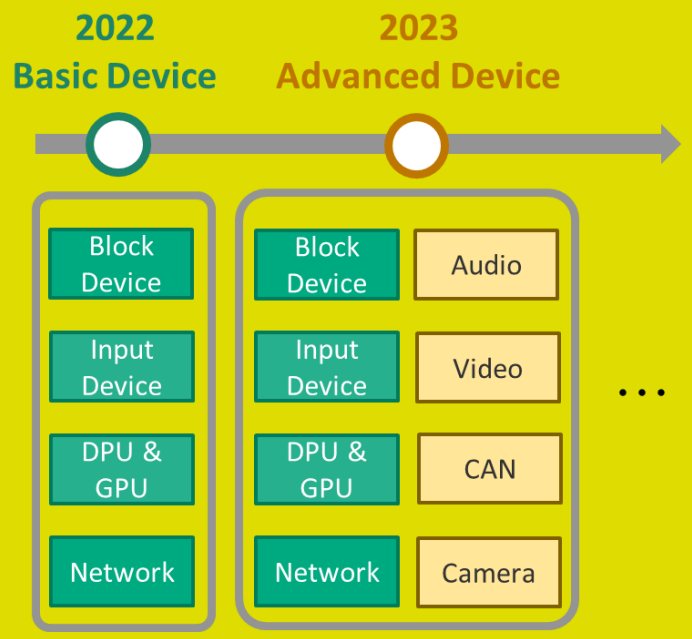
Common device HAL “virtio-loopback” portable to execute on both native and virtual environment

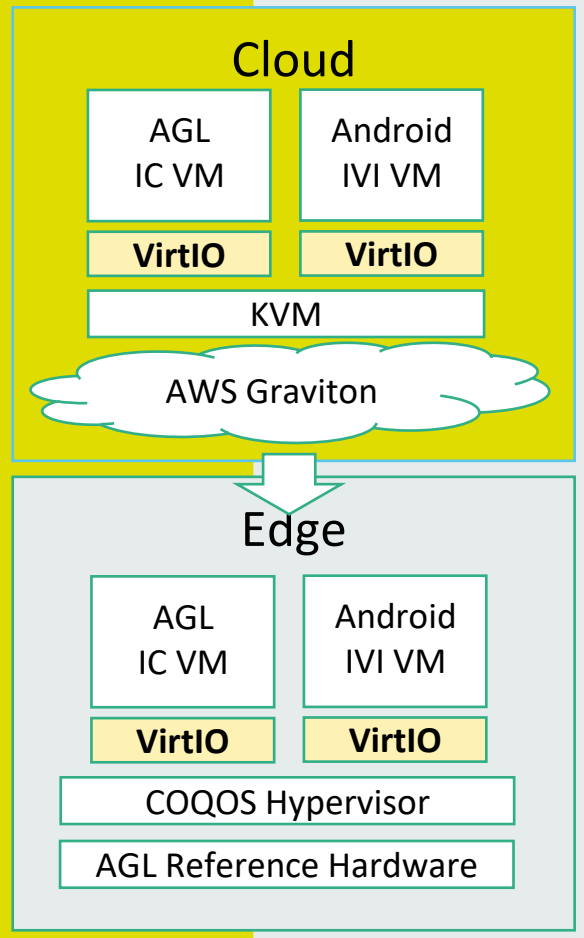


Panasonic AUTOMOTIVE

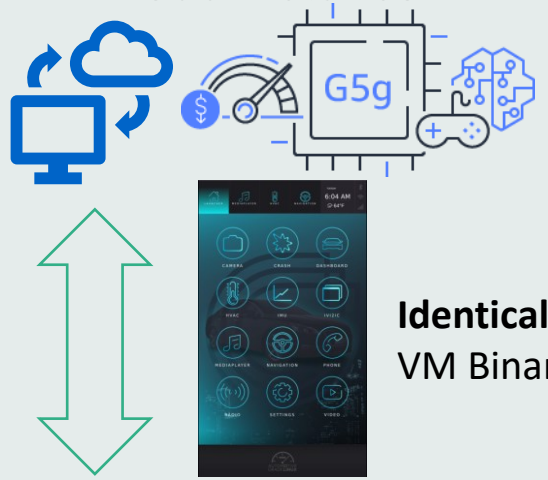
VirtIO for Cloud

Completely Identical IVI binary running on both cloud and edge



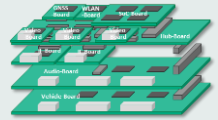


AWS Graviton G5g
Cloud Instance



Identical
VM Binary

Panasonic SkipGen-Flex



Panasonic AUTOMOTIVE

VirtIO for Cloud

Demo) Completely Identical IVI binary running on both cloud and edge

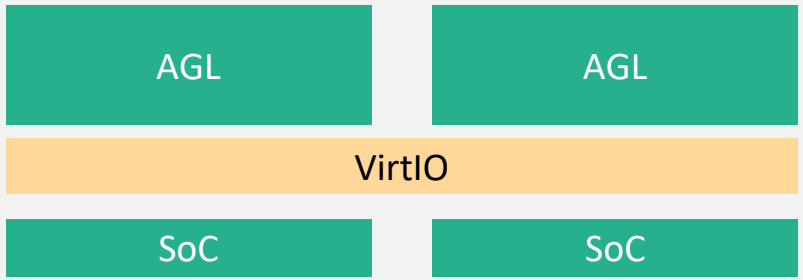
Panasonic AUTOMOTIVE

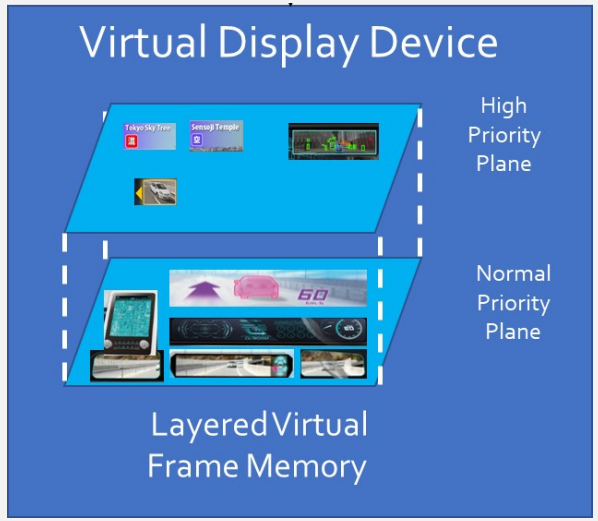
VirtIO Beyond Single SoC

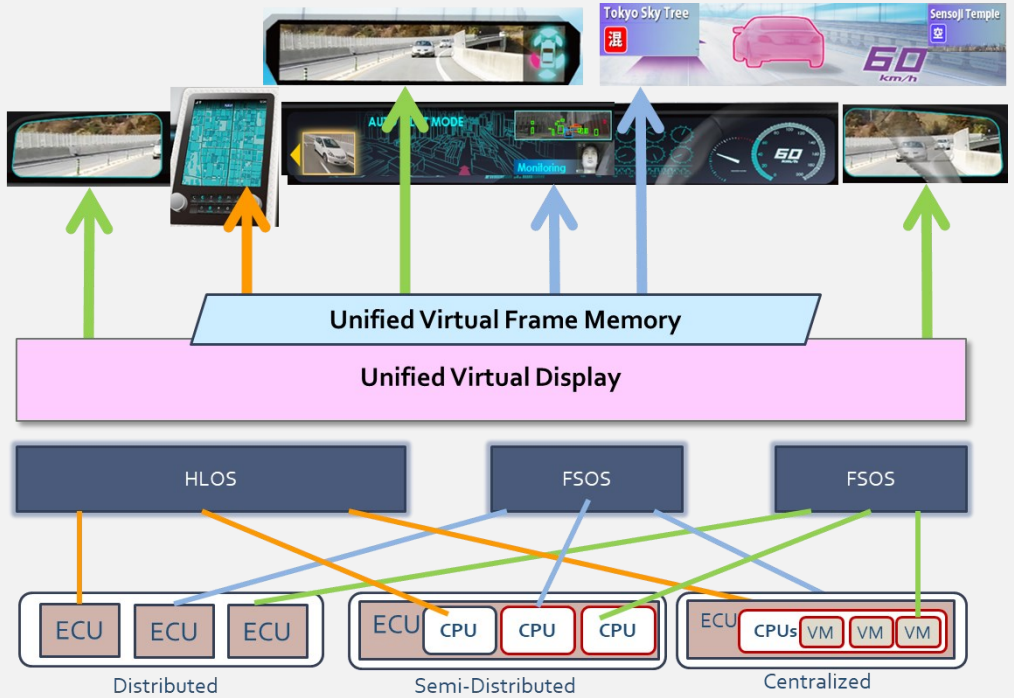
Integrated Cockpit Virtual Display System “Unified HMI”

A Unified Virtual Display based on VirtIO-GPU (“Unified HMI” technology) can be established to have Integrated control of multiple display on distributed SoC systems

Multi-ECU Environment







- **Mappig multiple physical displays** of cockpit & cabin into a **single large virtual display**
- **Rendering each application** to its **arbitrary region**

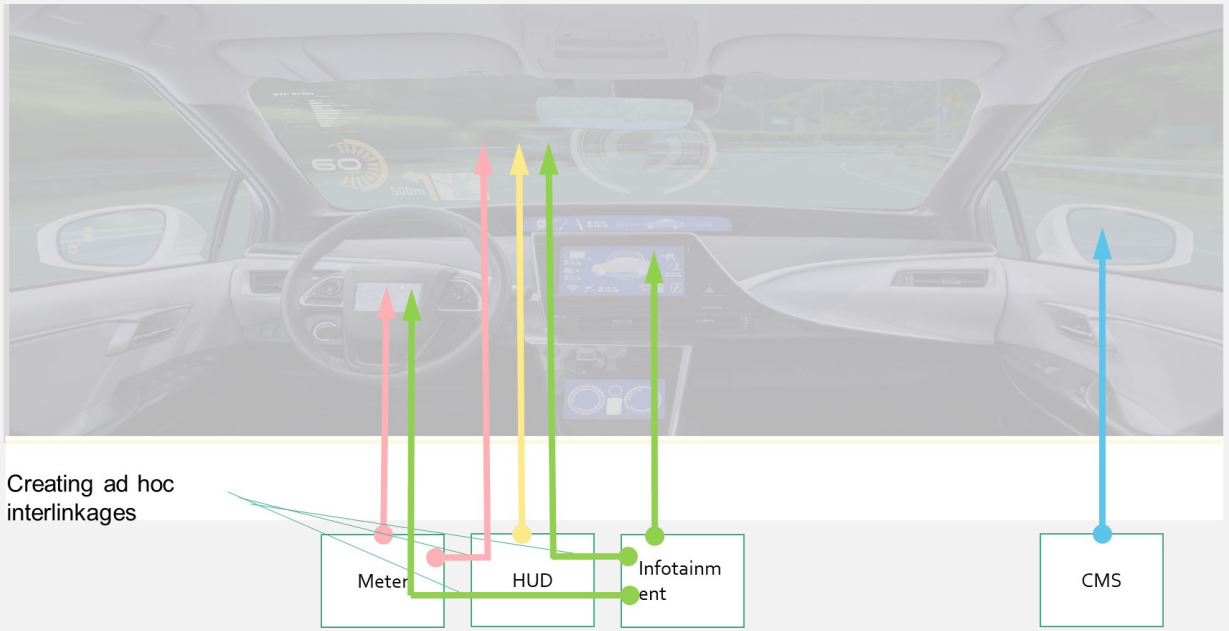
Panasonic AUTOMOTIVE

Integrated Cockpit Virtual Display System “Unified HMI”

VirtIO for Multi-ECU

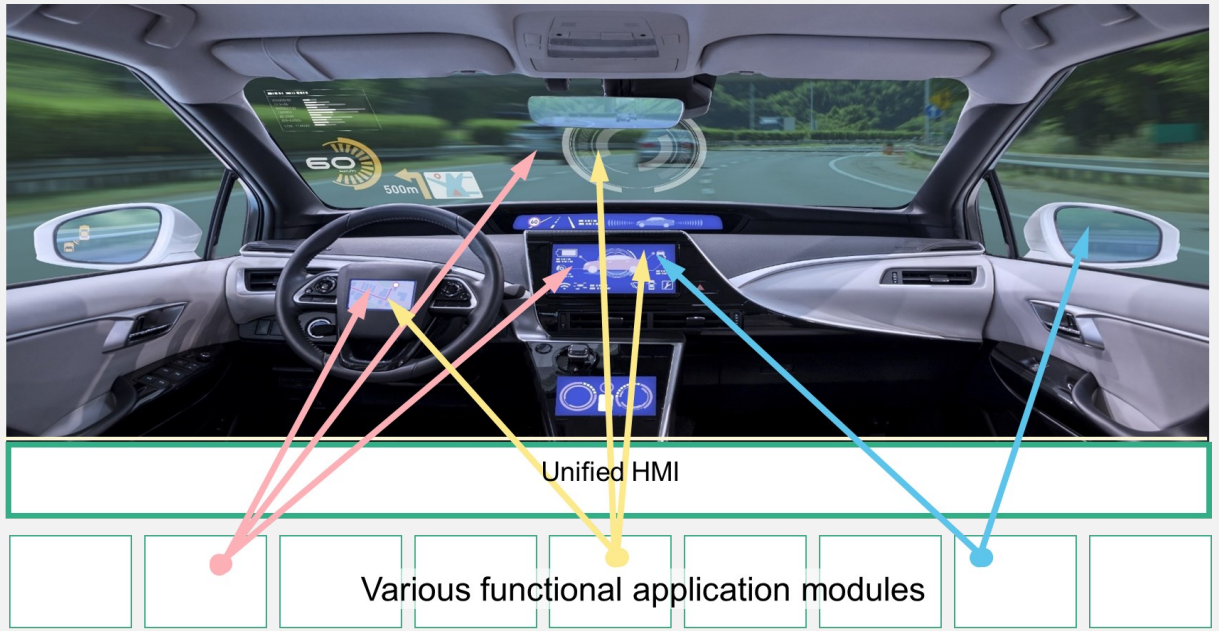
Legacy HMI System

Strict Restriction on ECU & Function-Display Relationship causing harmful Impediment for Cockpit UX



Unified HMI System

Full Flexibility on ECU & Function-Display Relationship for Cockpit UX Innovation

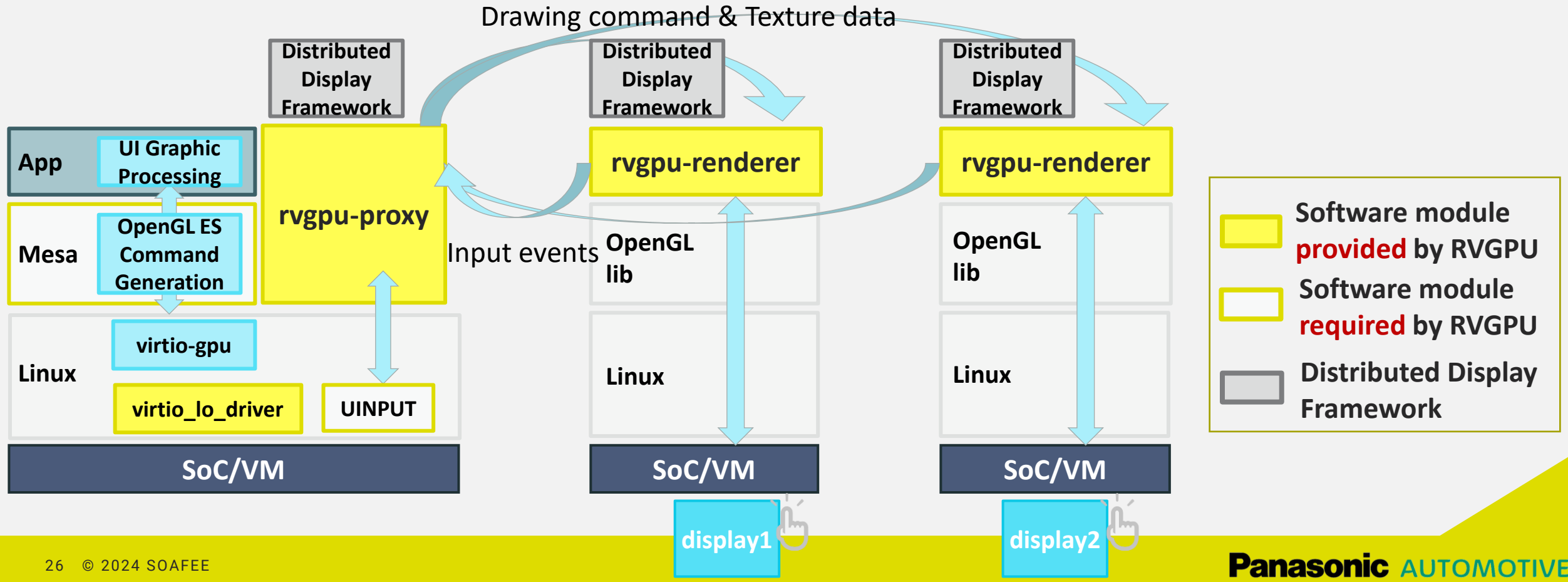


Panasonic AUTOMOTIVE

Unified HMI Architecture

Consists of two main components.

1. Remote Virtio GPU Device(RVGPU) : Render apps remotely in different SoCs/VMs.
2. Distributed Display Framework : Flexible layout control of apps across multiple displays.

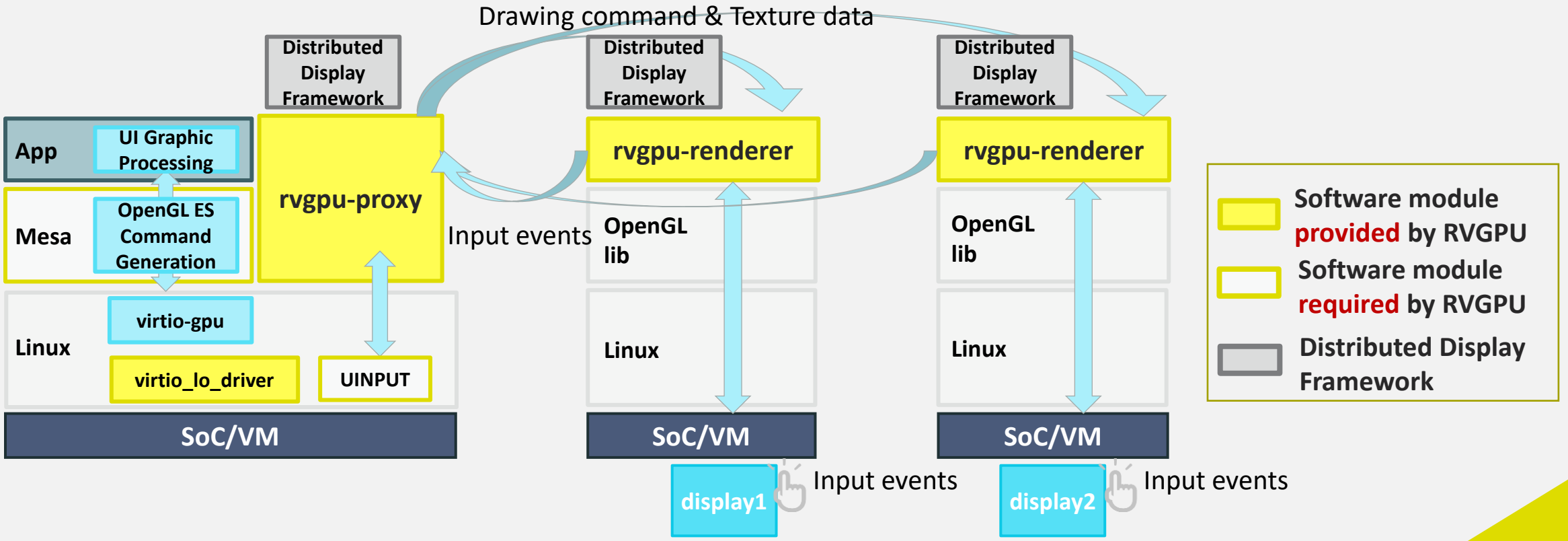


Panasonic AUTOMOTIVE

Unified HMI Architecture

Remote Virtio GPU Device (RVGPU)

- Network extension of **virtio-gpu** commonly used in GPU virtualization for VM.
- rvgpu-proxy : Transfer GPU commands generated by OpenGL ES to other SoCs/VMs.
- rvgpu-renderer : Receive GPU commands and draw graphics.



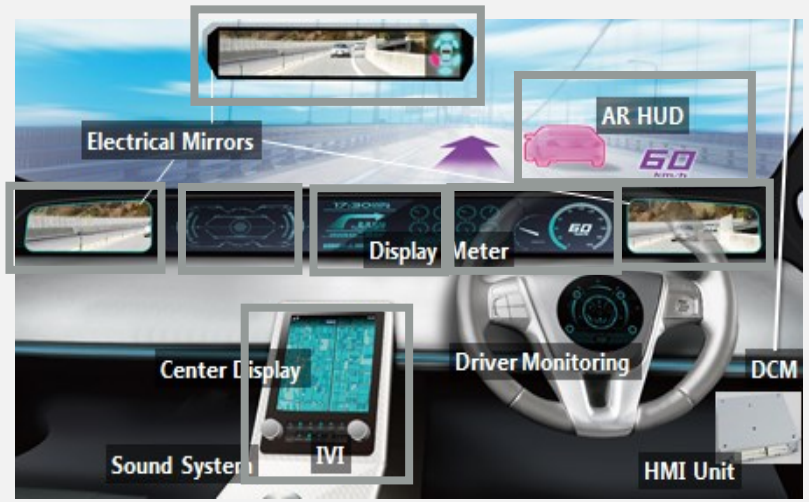
Panasonic AUTOMOTIVE

Unified HMI Architecture

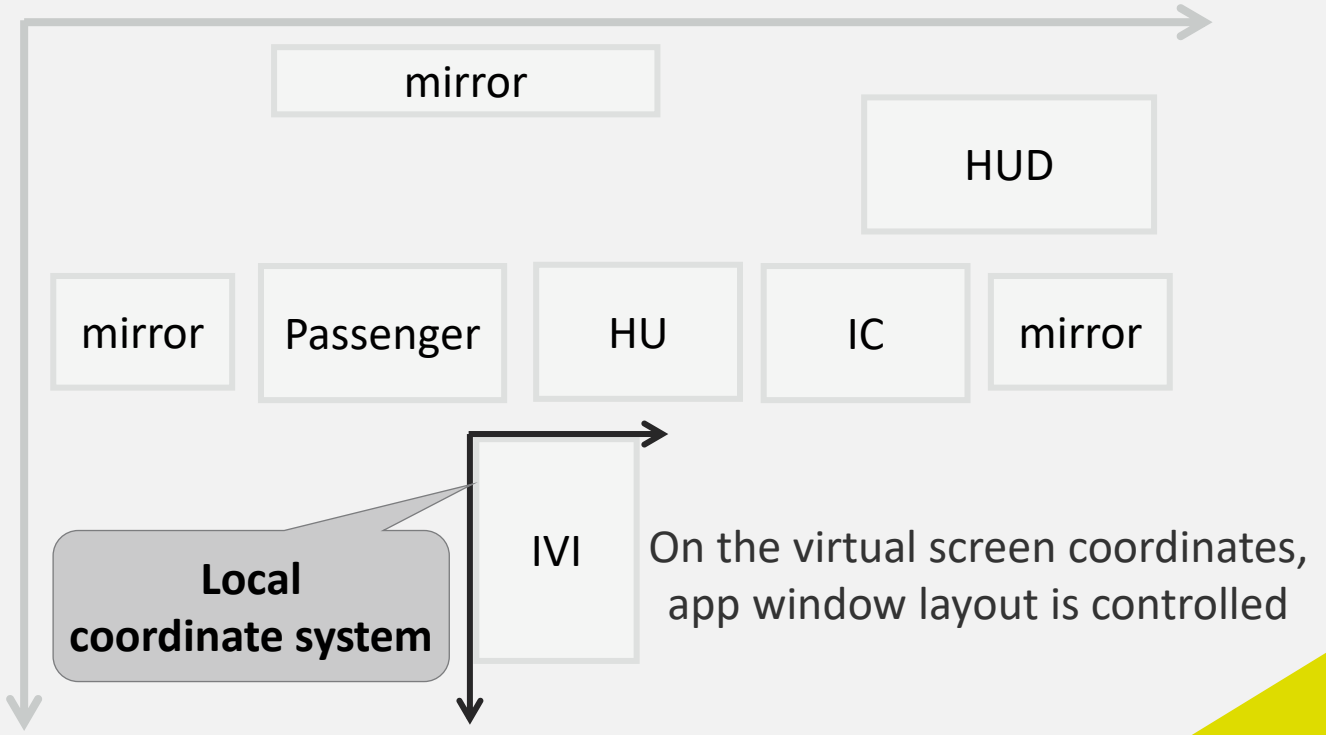
Distributed Display Framework

- Mapping multiple cockpit physical displays into a single large virtual screen.
- Control layout such as location, size, and display order of multiple apps.

Multiple cockpit physical displays



Virtual display

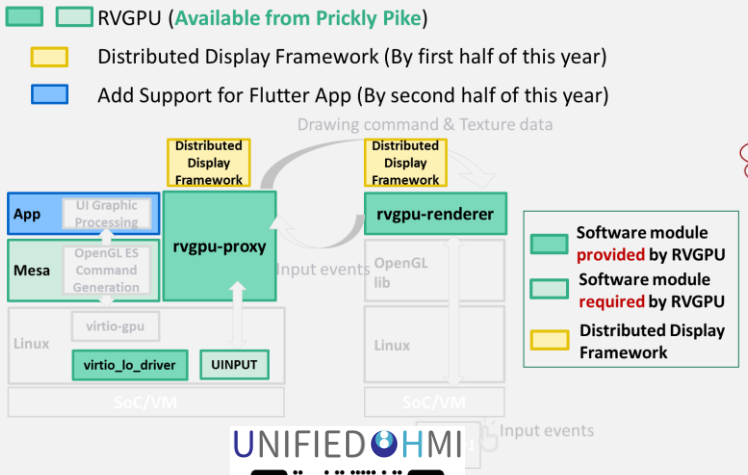


Unified HMI

Demo

Unified HMI Now and Beyond

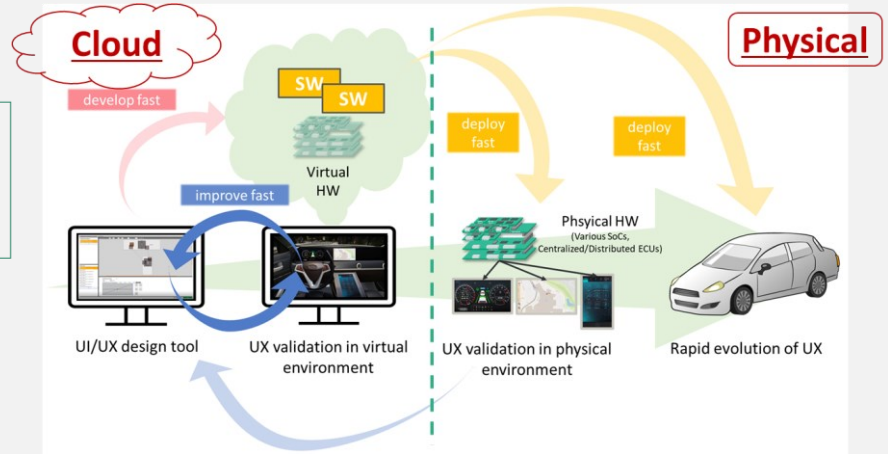
①Fundamental Unified HMI features have been **available Open-Source in Github and AGL UCB** and more features will be supported this year.



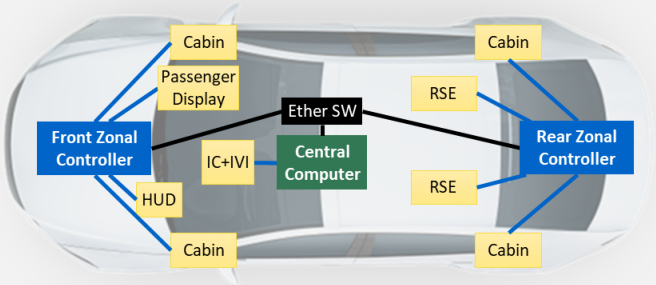
UNIFIED HMI

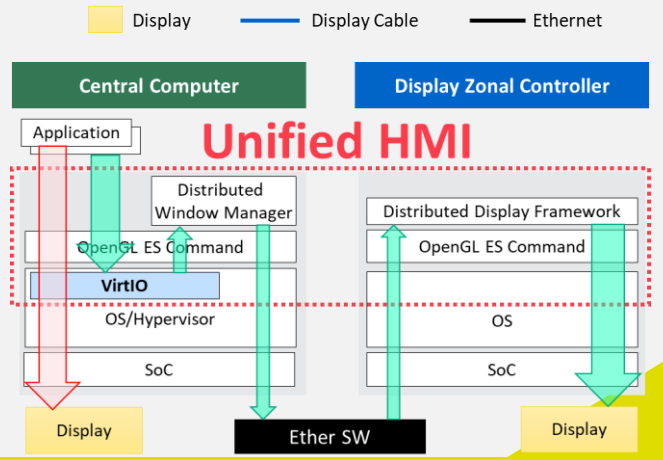
https://github.com/unified-hmi

②Collaborates with AWS to enable a **cloud native Unified HMI** environment with AGL able to develop UX/SW first and HW second.



③Collaborates with ARM to realize a “Display Zonal Architecture” with Unified HMI & AGL to achieve a **scalable zonal architecture**.

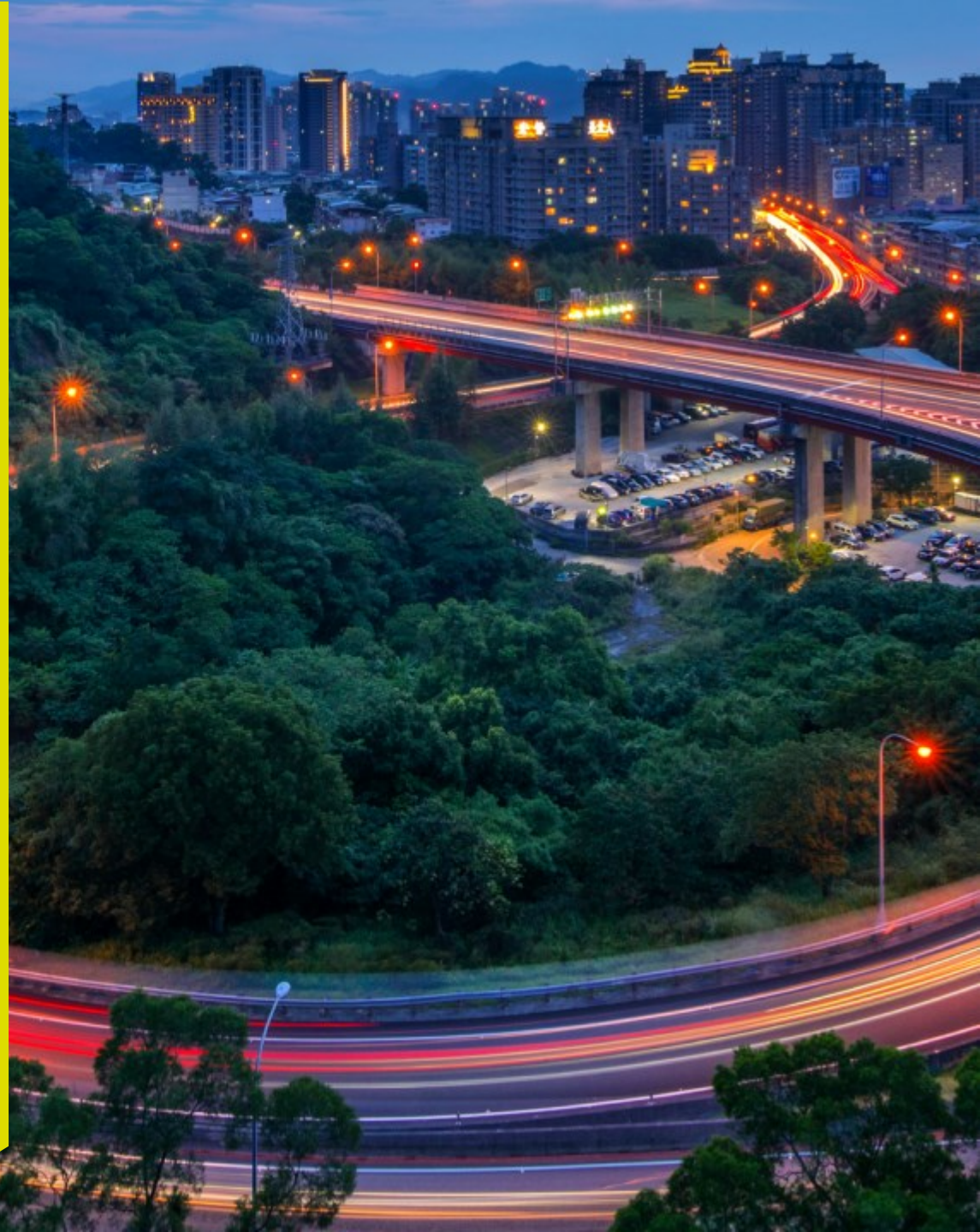




SOAFEE

Conclusion & Outlook

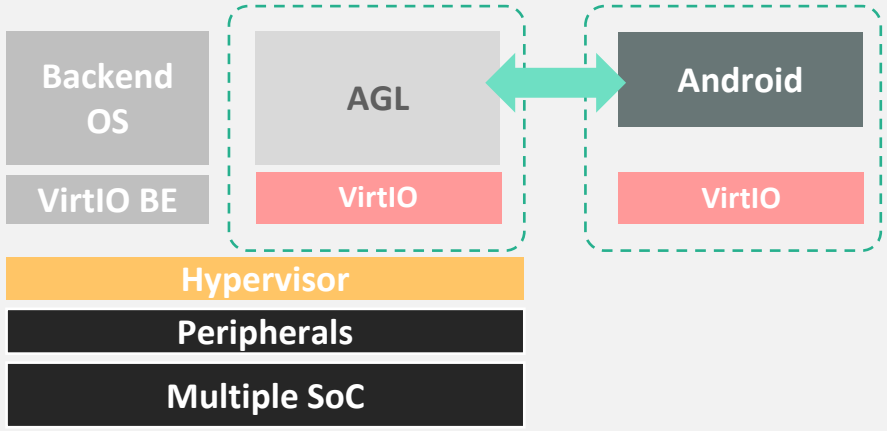
Constructing a bright and open future of SDV with SOAFEE



Advantages of Adopting VirtIO in CDC

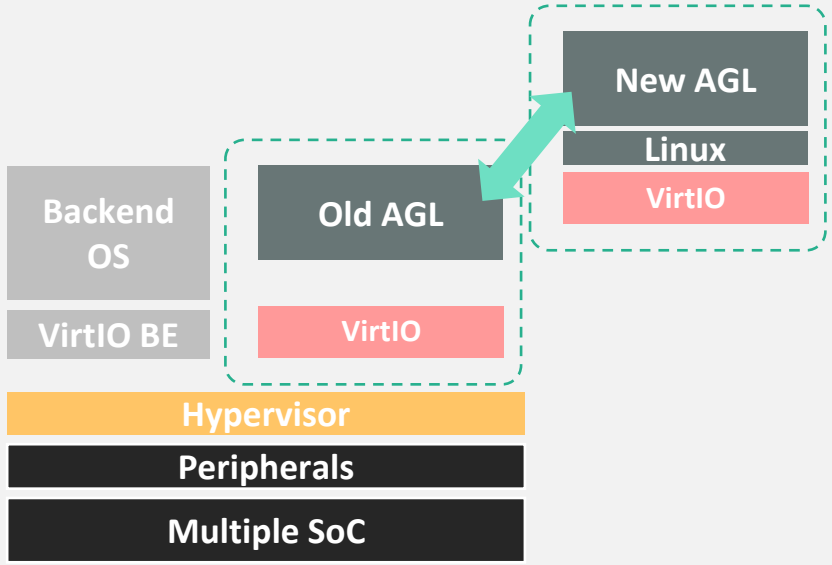
①Easy to Switch

VirtIO enables to easily replace other OS frontend to AGL, while keeping OEMs/Tier1s' existing backend and base SW&HW platform.



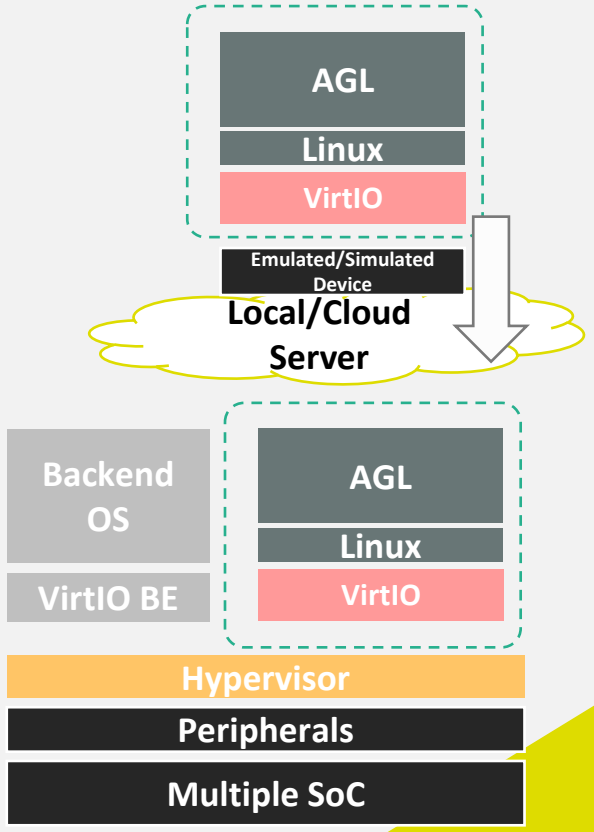
②Easy to Upgrade

VirtIO enables OS to easily upgrade without dependency on SoC Vendor



③Easy to Migrate

VirtIO enables OS to be developed on cloud and seamlessly deployed to edge ECU, which enables full OS-level binary parity



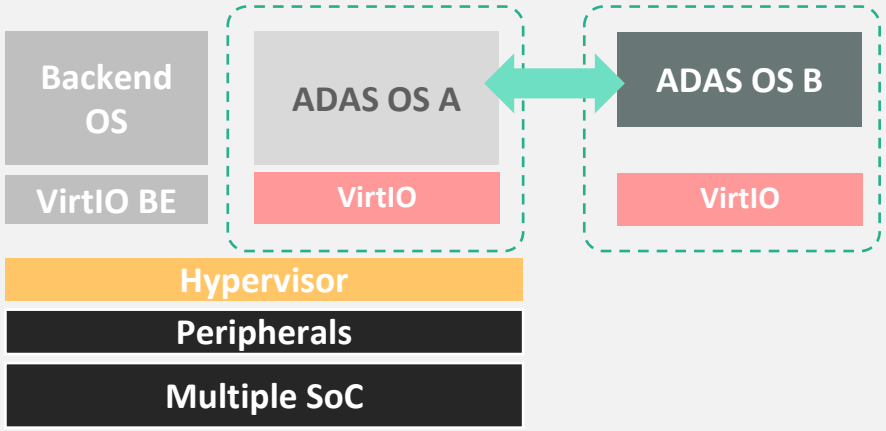
Advantages of Adopting VirtIO in ADAS

Increasing needs for functionality feature development & updates for ADAS and growing numbers of solutions from SoC/OS/App vendors for ADAS/AD domain.

→ Need to ensure environment parity between cloud and edge and across different edge SoCs

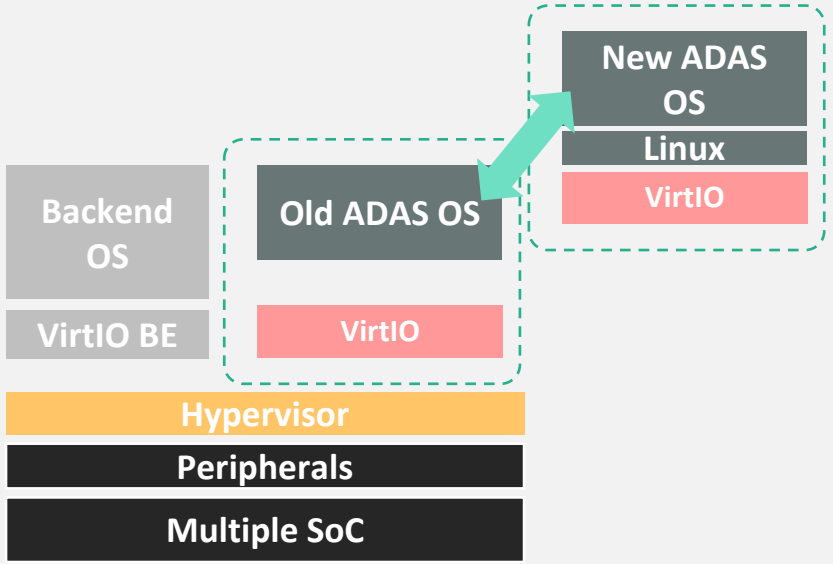
①Easy to Switch

Enable switch across different ADAS/AD Operating systems and different SoCs



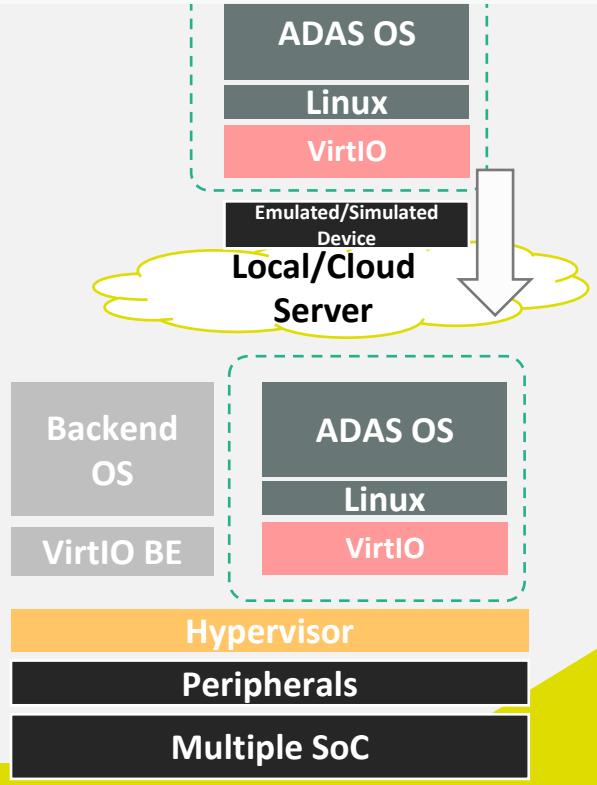
②Easy to Upgrade

Enable easy upgrade without dependency on SoC Vendor



③Easy to Migrate

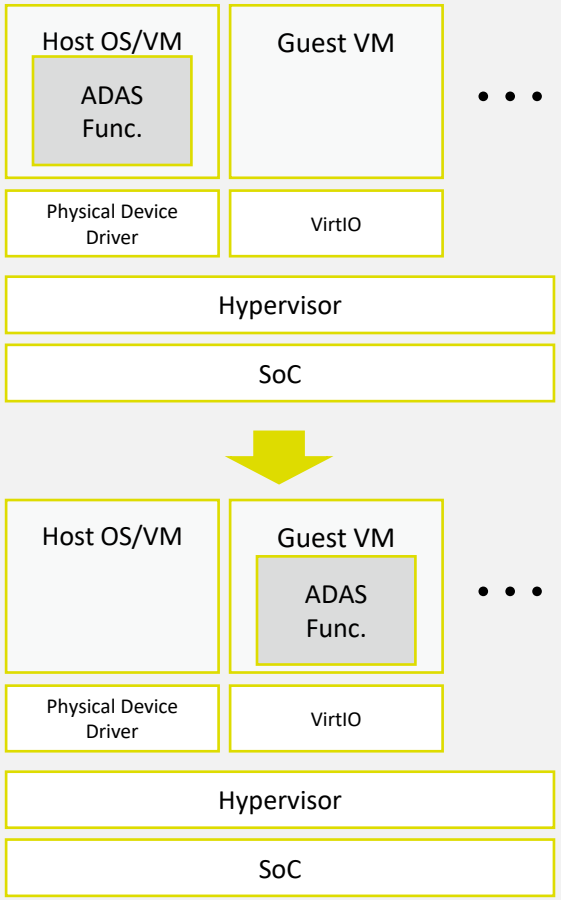
Enable cloud-native to reduce development cycle



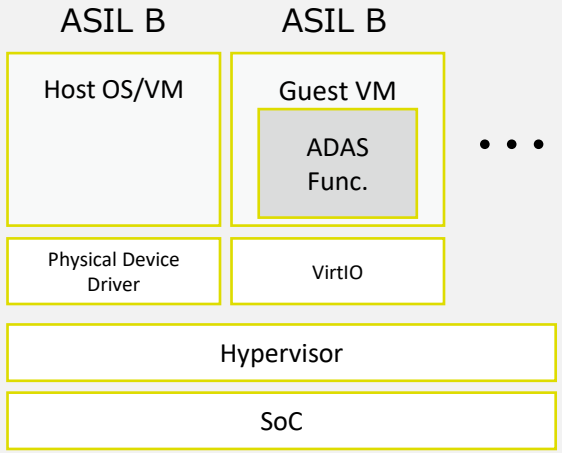
Challenges of Adopting VirtIO in ADAS

Necessary for more discussions and collaboration in Open Community just like SOAFEE!

①Existing architecture dependent on specific SoC/HV



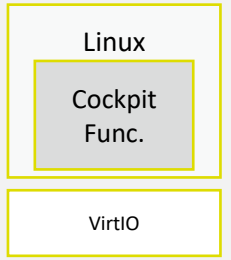
②Performance Function Safety



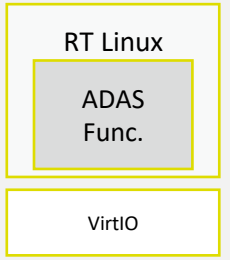
- How FuSa (ASIL-B) can be achieved with VirtIO
- How to reduce performance overhead for ADAS use cases

③No Available VirtIO Implementation for RTOS

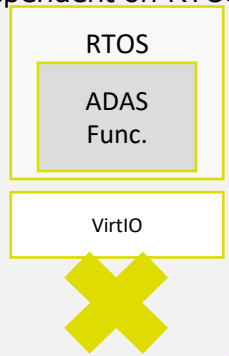
VirtIO Impl. Available and Standardized in Linux Kernel Upstream



Need to be integrated with RT Linux kernel

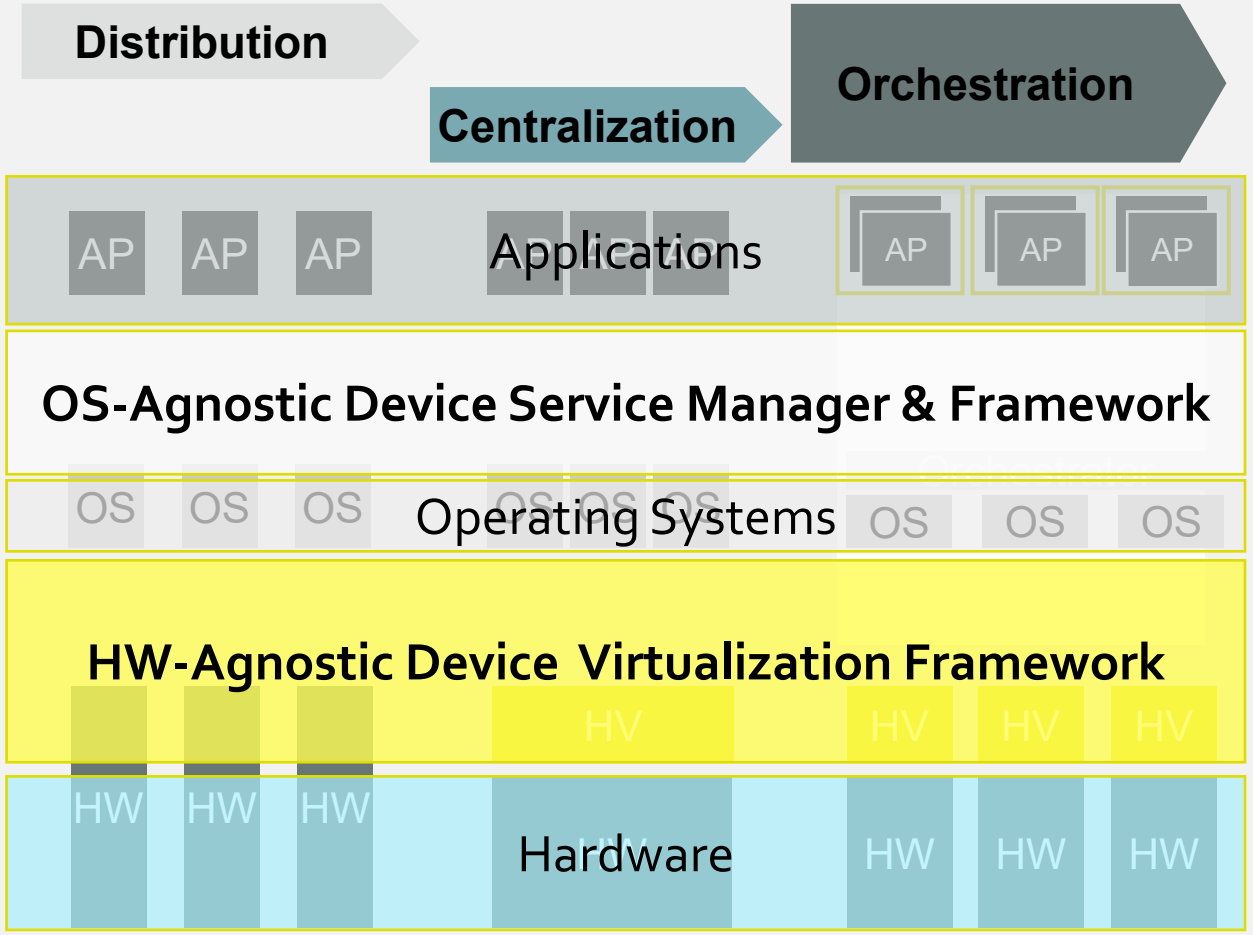


No Impl. for RTOS and dependent on RTOS vendor



Panasonic AUTOMOTIVE

Ideal Device Virtualization Framework for Software-Defined Vehicles (SDVs)



Scalable **O**pen **A**rchitecture **f**or **E**mbedded **E**dge

SOAFEE

Scalable, **O**pen, **A**utomotive, **F**lexible, **E**fficient, **E**ndurable

SOAFEE

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה

Panasonic AUTOMOTIVE