Panasonic
AUTOMOTIVE

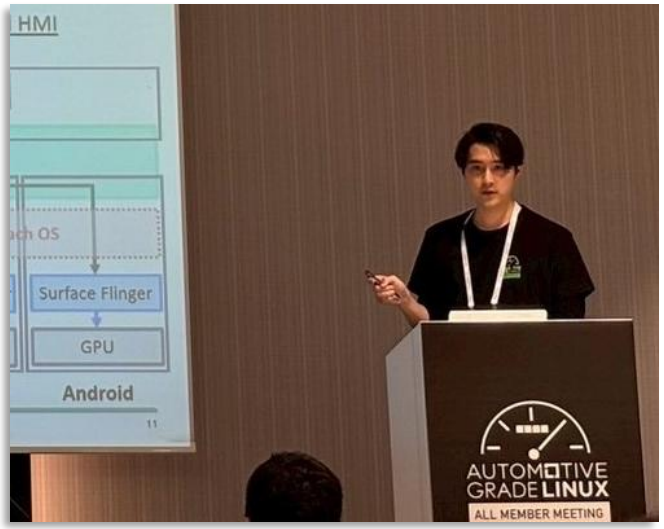
Unified HMI Updates:
Simplified APIs for Dynamic Multi-Display Layout Control

Panasonic Automotive Systems, Co., Ltd.
Kenta Murakami

Who am I ?

Panasonic AUTOMOTIVE

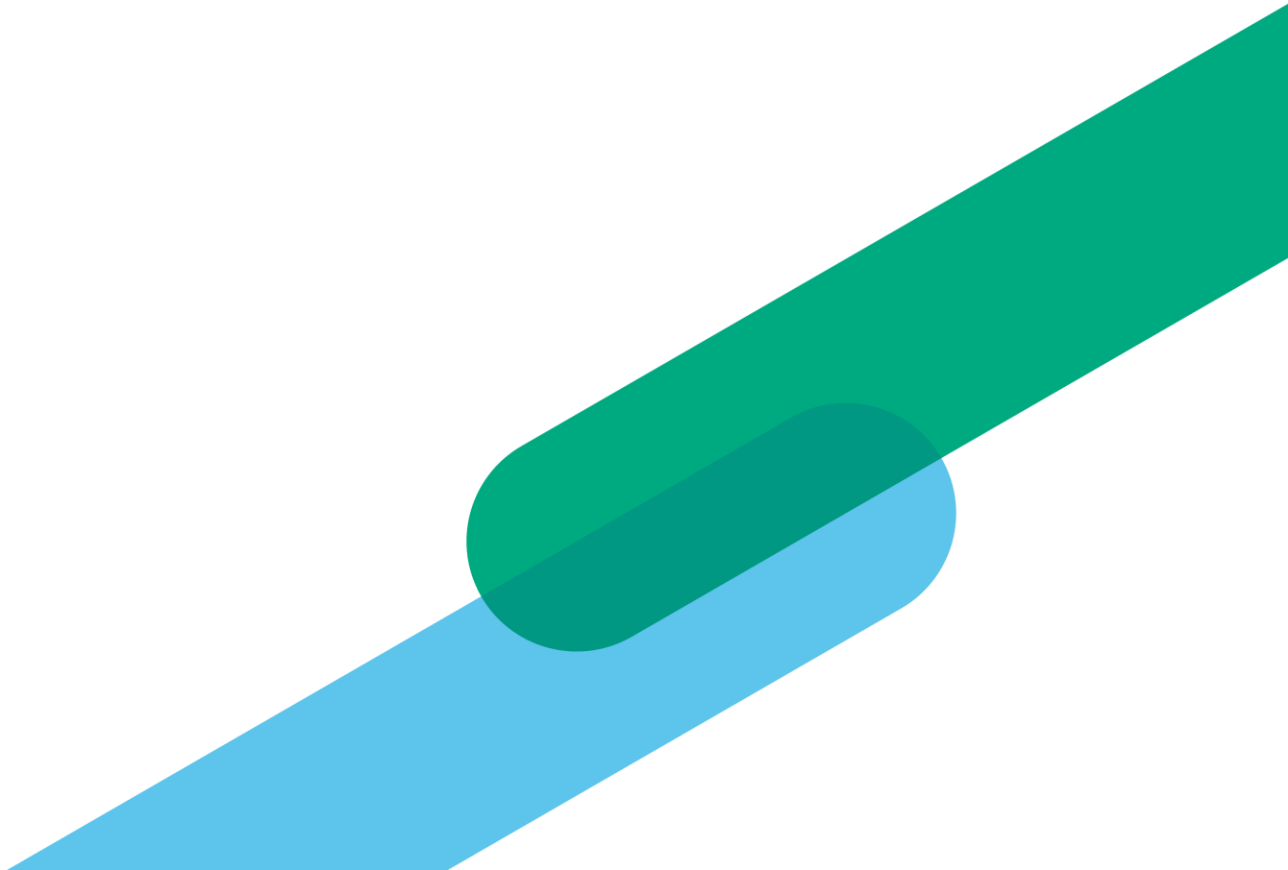
- ✓ Working at Panasonic Automotive Systems Co., Ltd.
- ✓ Focusing on the Unified HMI development.
- ✓ Contributing Unified HMI programs to AGL UCB since Oct. 2024.
- ✓ Hobby: Travel, Ramen, Coffee, Playing with my baby



Agenda

Panasonic AUTOMOTIVE

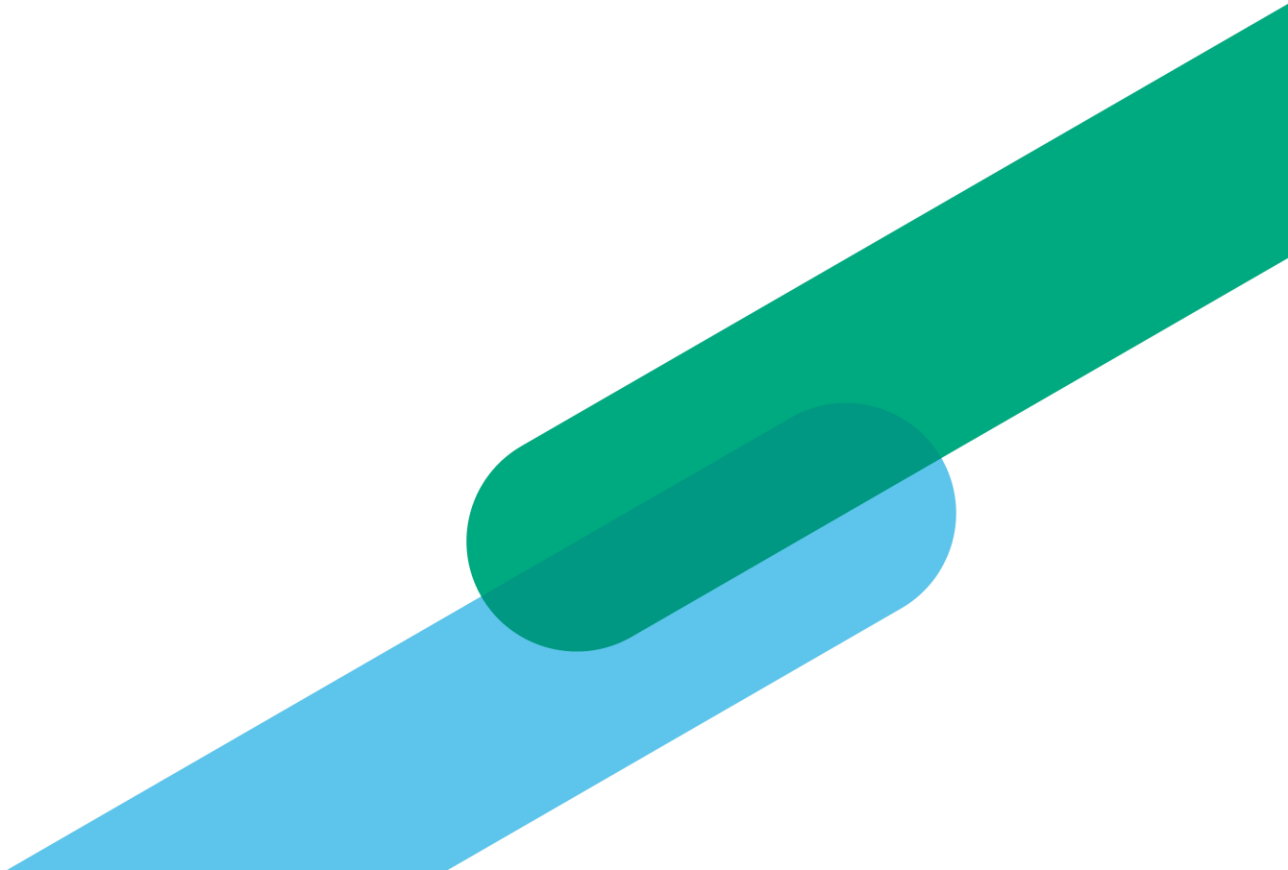
- ✓ What is Unified HMI
- ✓ Unified HMI Architecture
- ✓ Updates for Unified HMI v3.0
  - ✓ UHMI v3.0 architecture
  - ✓ Demo video
  - ✓ UHMI on AGL SoDeV



Agenda

Panasonic AUTOMOTIVE

- ✓ **What is Unified HMI**
- ✓ Unified HMI Architecture
- ✓ Updates for Unified HMI v3.0
  - ✓ UHMI v3.0 architecture
  - ✓ Demo video
  - ✓ UHMI on AGL SoDeV



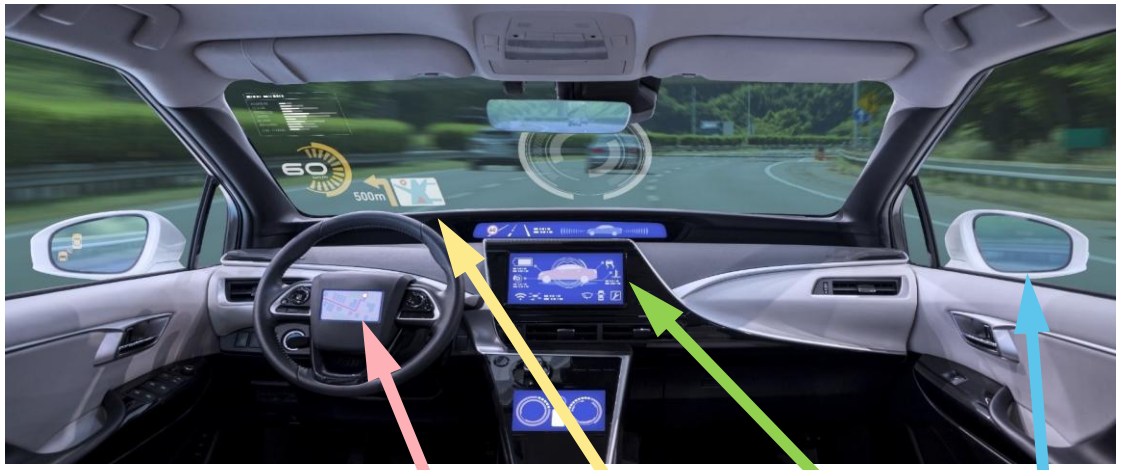
Trends in the Car Cockpit Systems

Panasonic AUTOMOTIVE

- ✓ The number of in-vehicle displays is increasing because of digitization of many devices.
  => Demands for flexible application display **across multiple displays**
- ✓ However, the development of **ad hoc coordination** for all car grades/models is required.

Future multi-display environment

Develop app layouts by using existing methods



What is Unified HMI

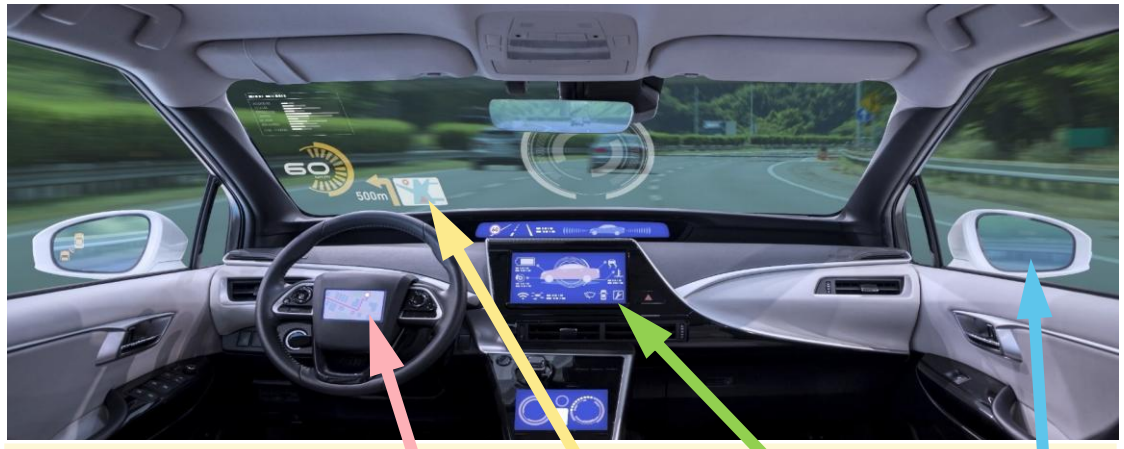
UNIFIED HMI

Panasonic AUTOMOTIVE

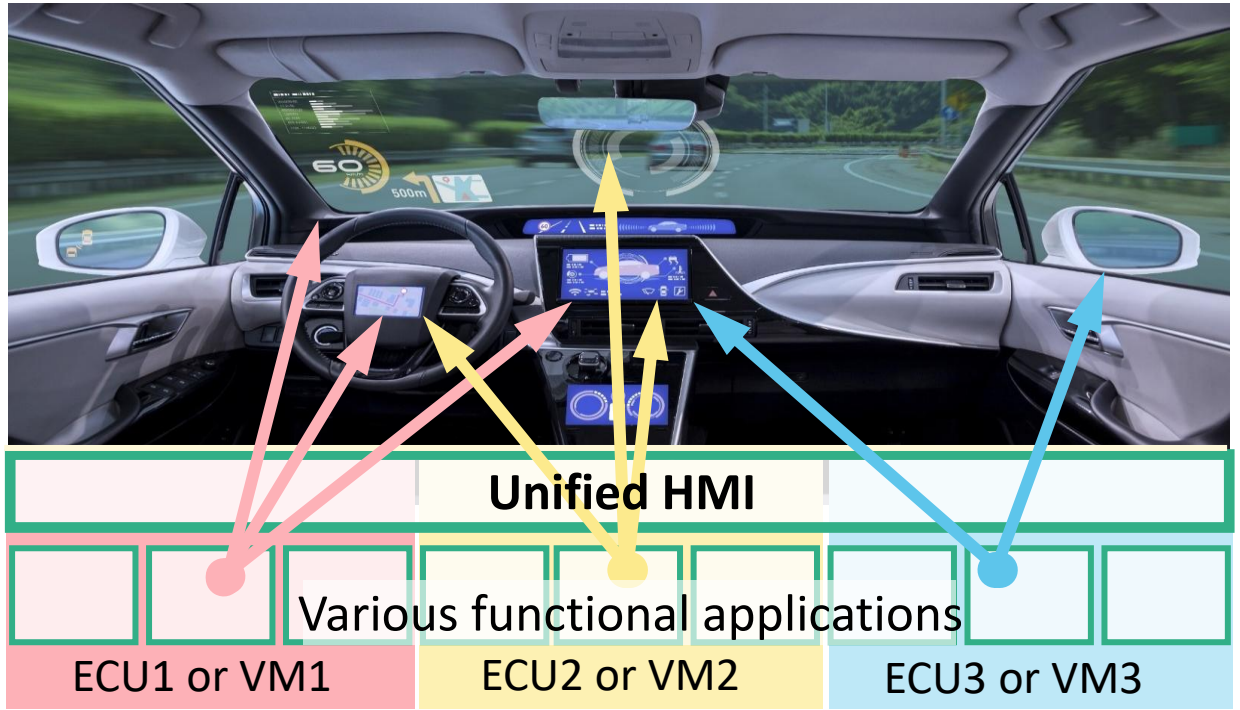
"**Software-Defined**" display virtualization platform based on VirtIO GPU.

- ✓ Flexible app layouts across multiple displays, **independent of hardware and OS configurations.**
- ✓ Specific modifications for each app are not required.

Without Unified HMI



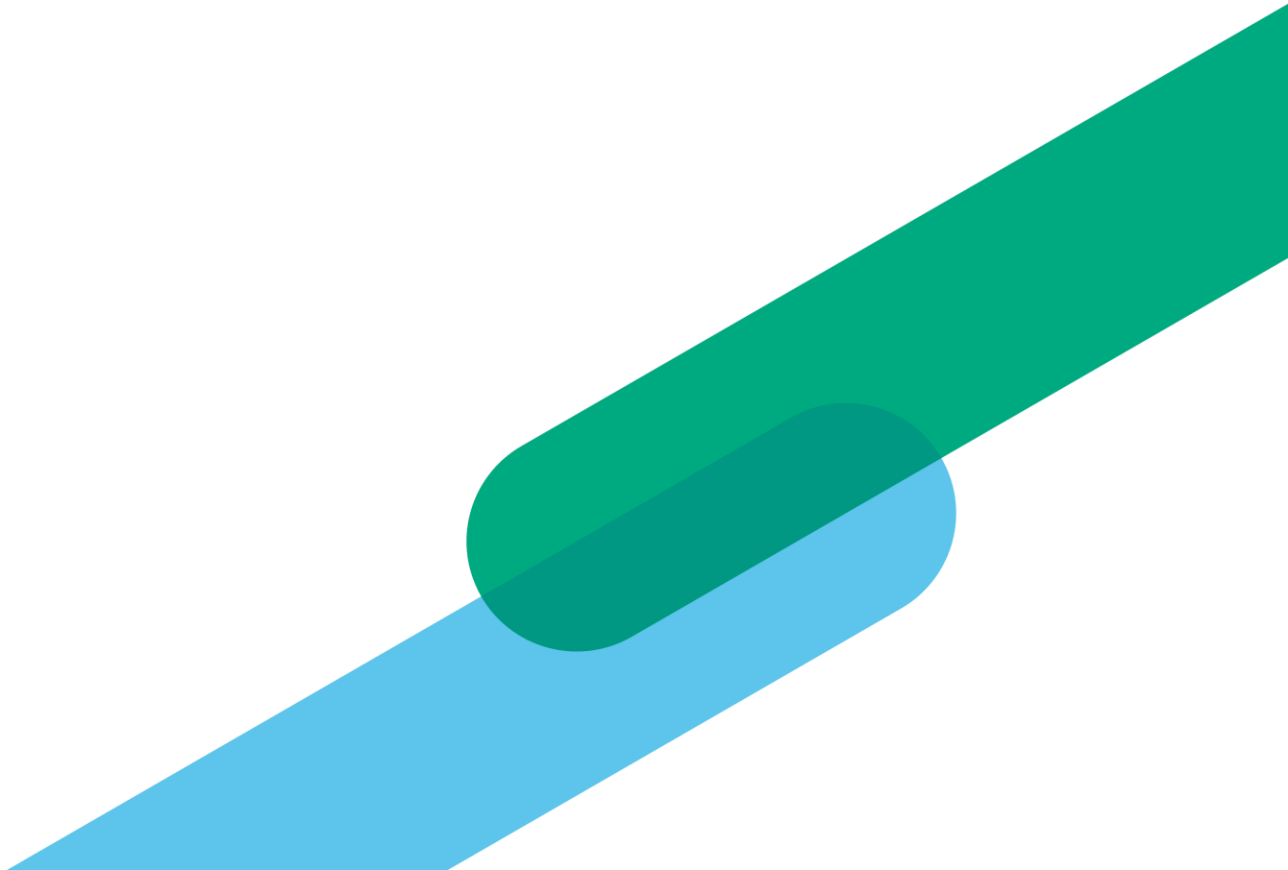
With Unified HMI



Agenda

Panasonic AUTOMOTIVE

- ✓ What is Unified HMI
- ✓ **Unified HMI Architecture**
- ✓ Updates for Unified HMI v3.0
  - ✓ UHMI v3.0 architecture
  - ✓ Demo video
  - ✓ UHMI on AGL SoDeV

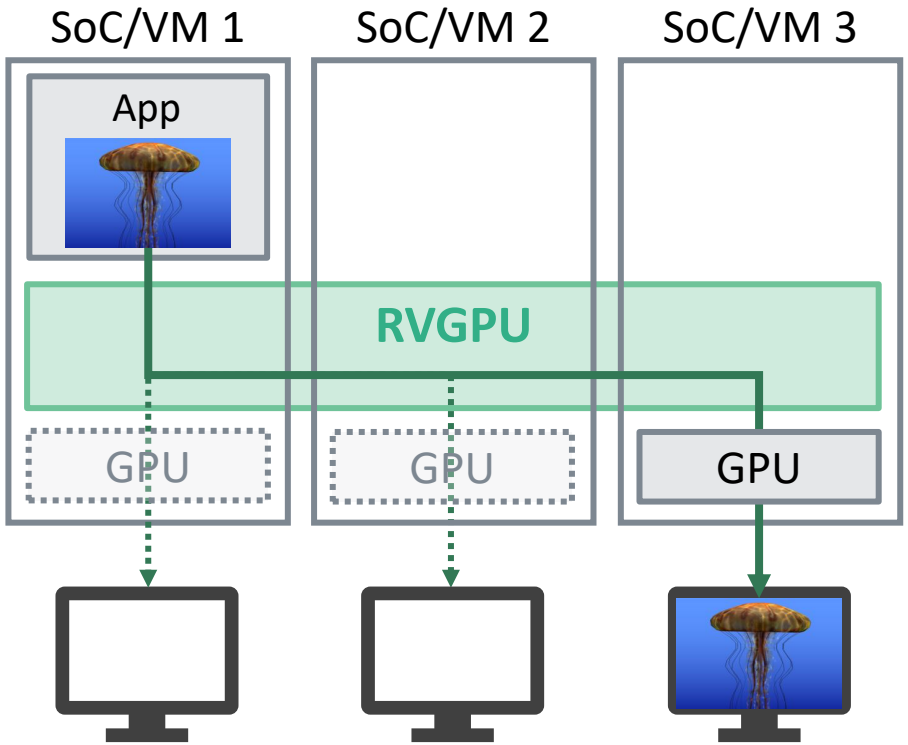


The two main components of Unified HMI

Panasonic AUTOMOTIVE

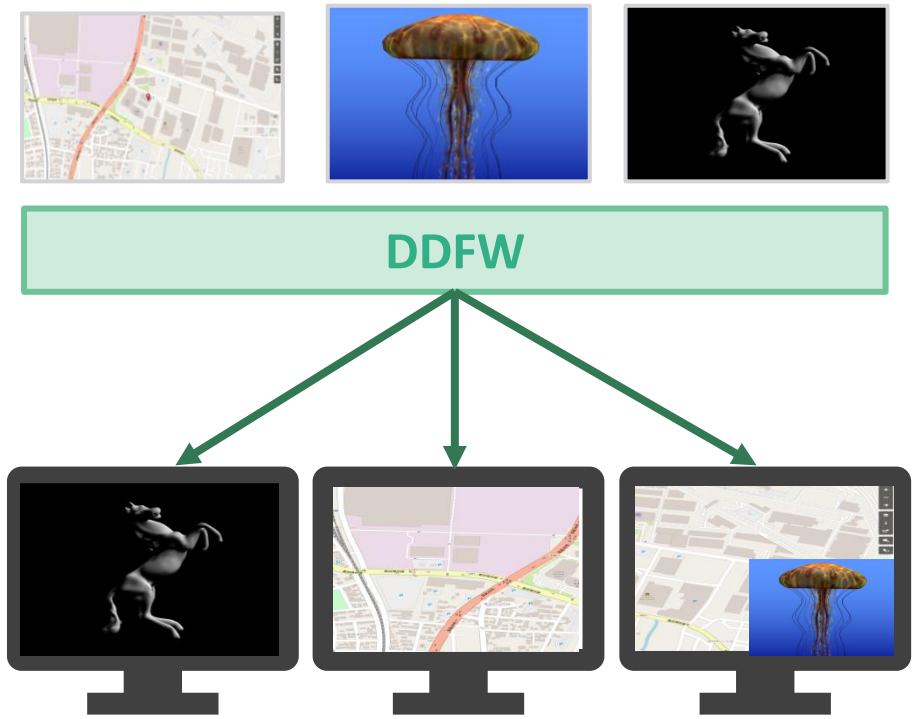
Remote VirtIO GPU Device(RVGPU)

✓ Render apps **remotely in different SoCs/VMs** even when **different OSes** are running.



Distributed Display Framework(DDFW)

✓ **Control layouts** across multiple displays and **manage the lifecycle** of apps.



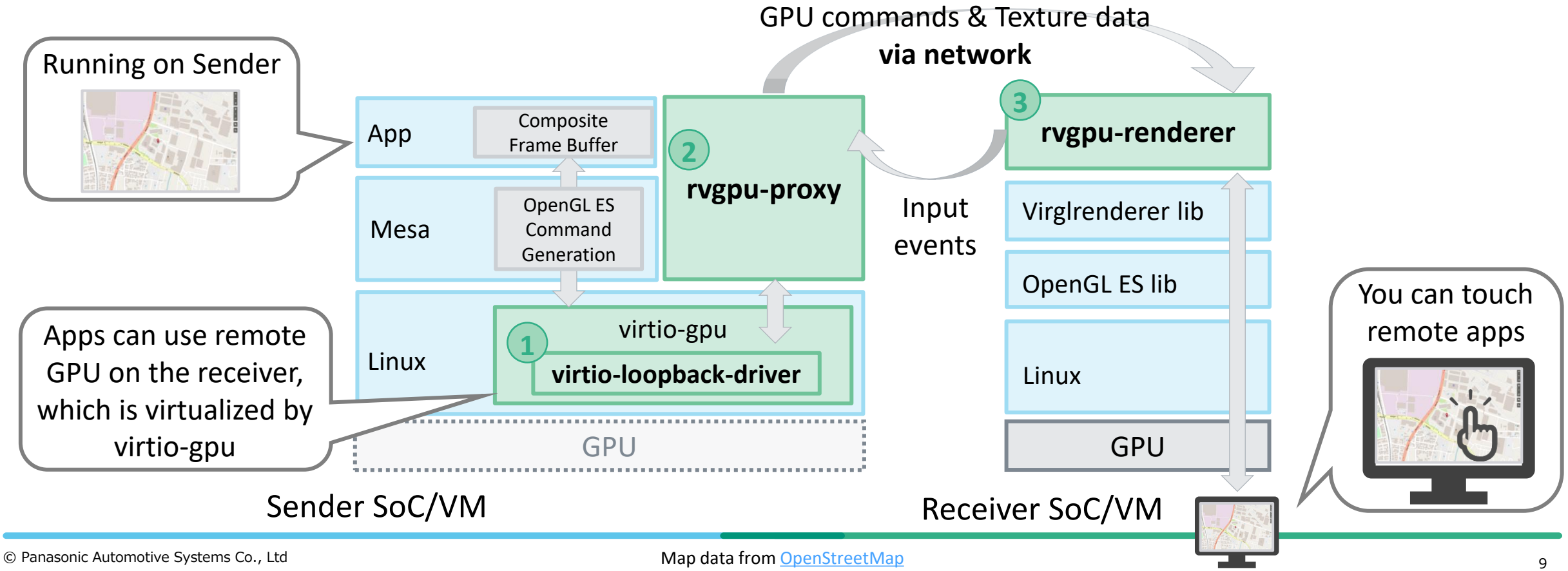
Map data from OpenStreetMap

Remote VirtIO GPU Device (RVGPU)

Panasonic AUTOMOTIVE

Network extension of **VirtIO GPU** commonly used in GPU virtualization for VM.

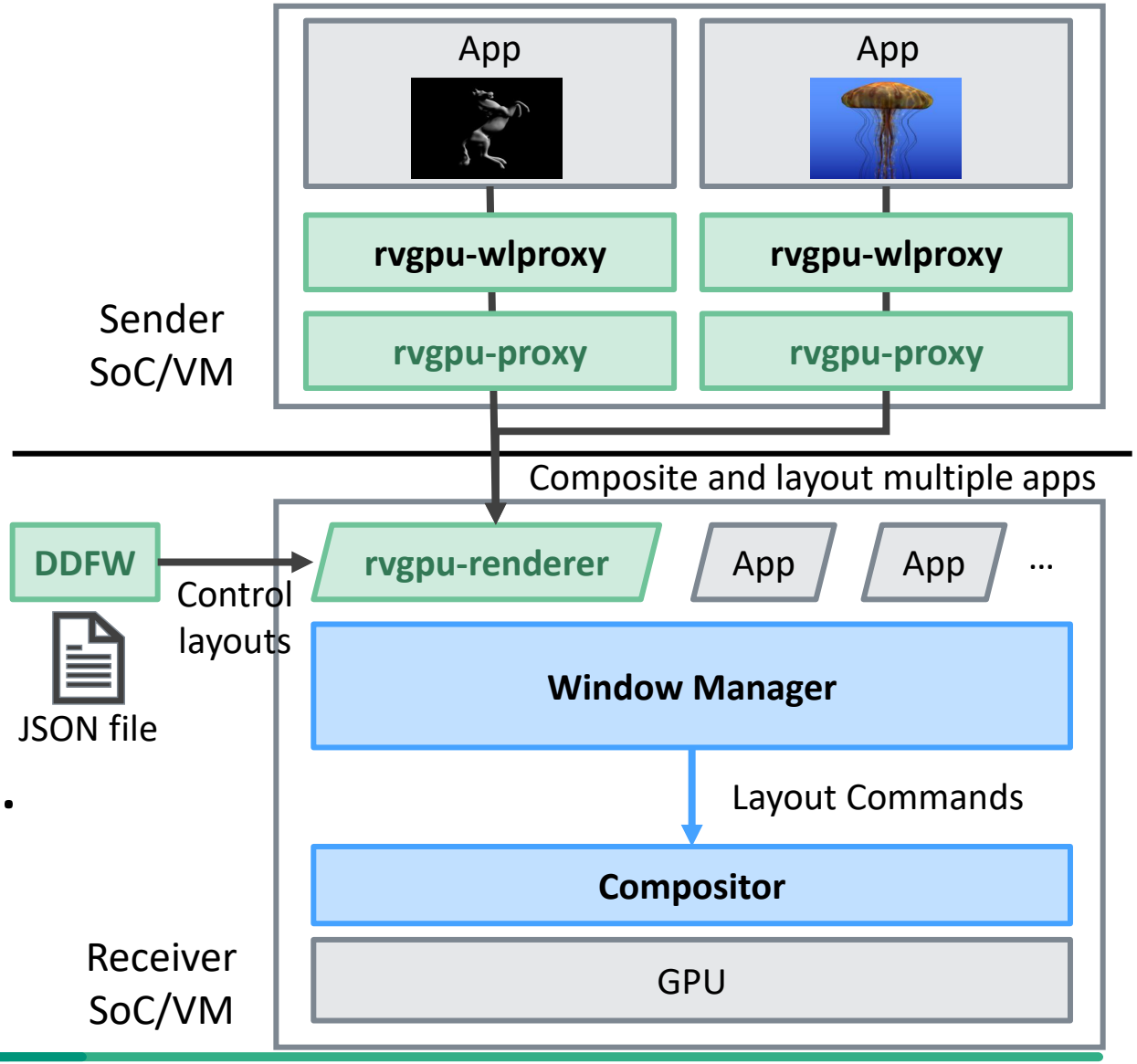
1. virtio-loopback-driver generate virtio-gpu device and receives **GPU commands** through it.
2. rvgpu-proxy sends commands to rvgpu-renderer on the receiver **via network**.
3. rvgpu-renderer render app on the receiver display **as usual data flow**.



Map data from OpenStreetMap

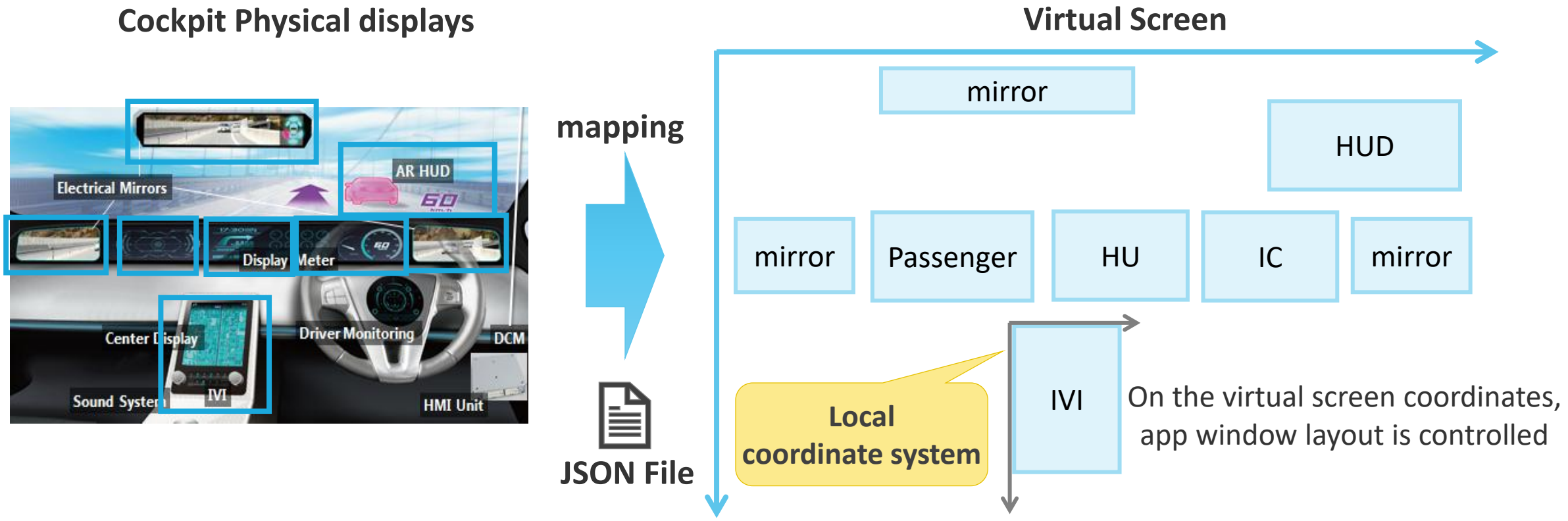
How Unified HMI control layouts

- ✓ rvgpu-renderer **composites** multiple app windows and controls layouts itself.
- ✓ It’s possible to support various OS, such as Linux/AGL/Android **without developing specific plugins for each.**
- ✓ The controlling layout functions can be developed on RVGPU, so it can easily achieve **flexible layouts across different OSes.**



Distributed Display Framework (DDFW)

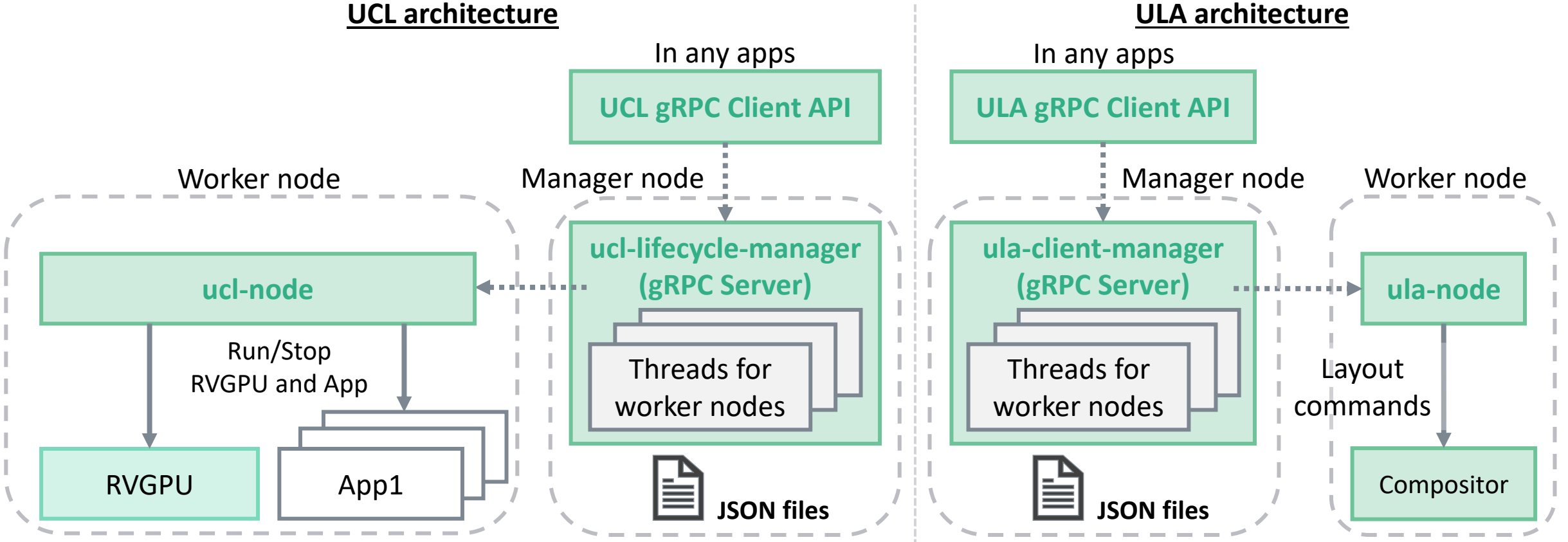
- ✓ Mapping cockpit physical displays into a **single large virtual screen**.
- ✓ Control app layouts such as location, size, and display order.
- ✓ **Manage the lifecycle** of apps and RVGPU modules.



DDFW architecture

Panasonic AUTOMOTIVE

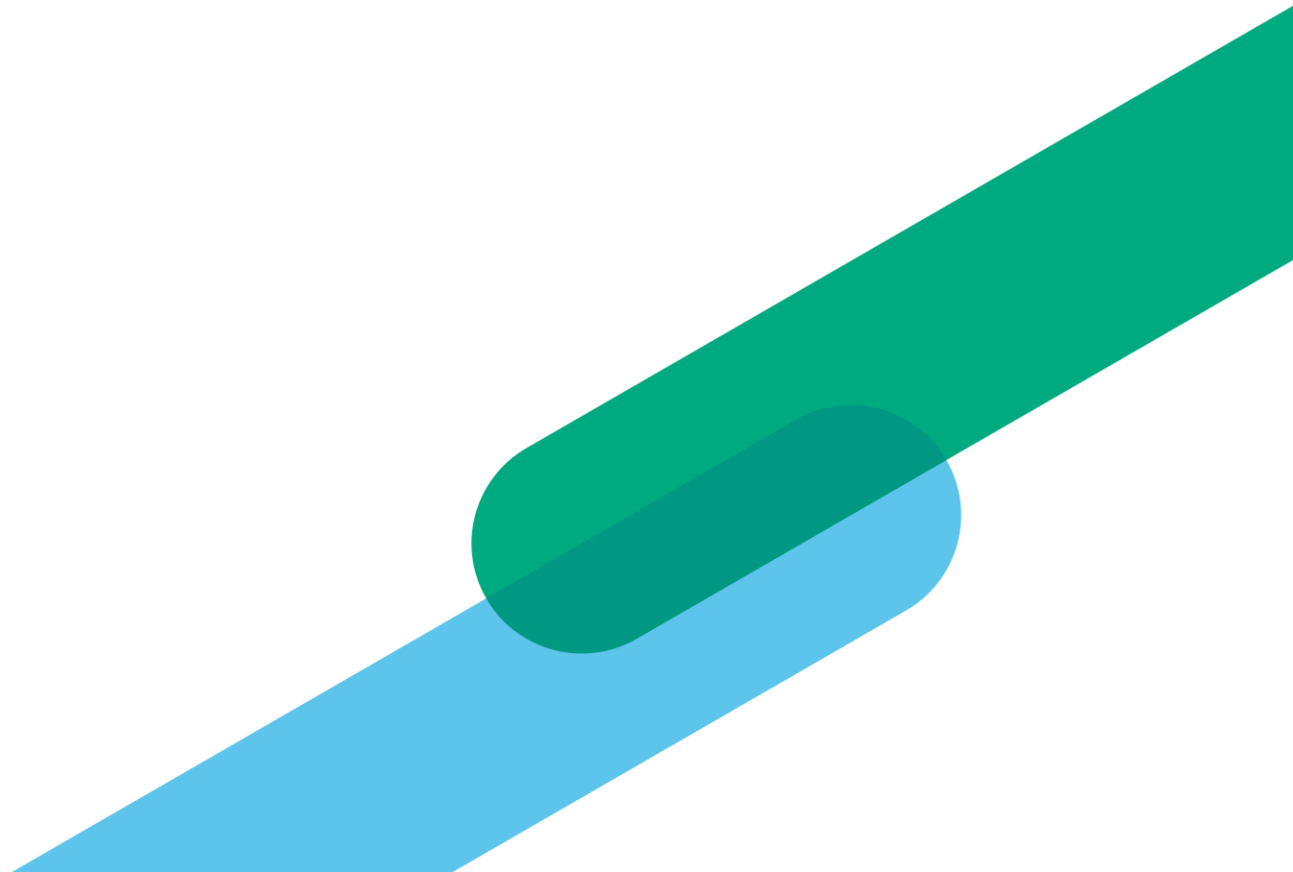
- ✓ Unified Clustering Tools(ucl-tools): Manages **overall lifecycle of apps and RVGPU modules.**
- ✓ Unified Layout Tools(ula-tools): Provides **unified control** of app layouts.
- ✓ UCL/ULA work as defined in **JSON files**.



Agenda

Panasonic AUTOMOTIVE

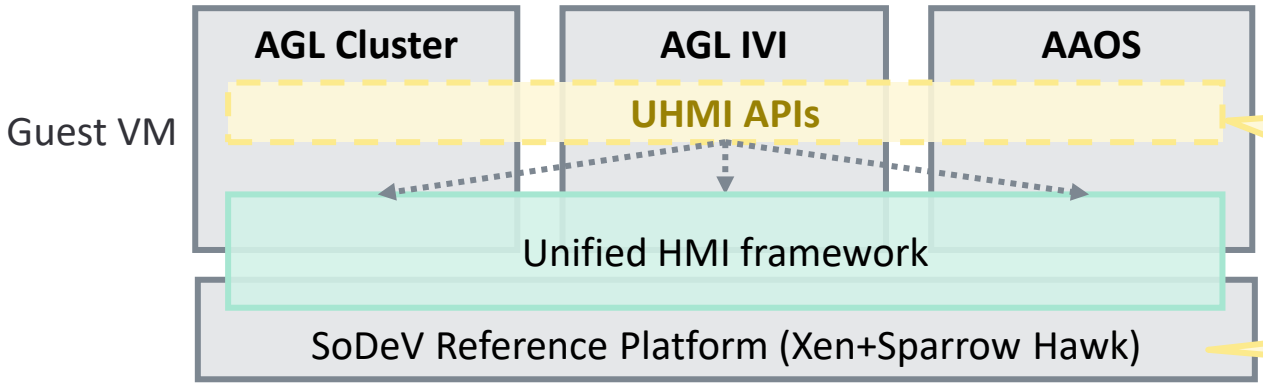
- ✓ What is Unified HMI
- ✓ Unified HMI Architecture
- ✓ **Updates for Unified HMI v3.0**
  - ✓ **UHMI v3.0 architecture**
  - ✓ **Demo video**
  - ✓ **UHMI on AGL SoDeV**



Updates for Unified HMI v3.0

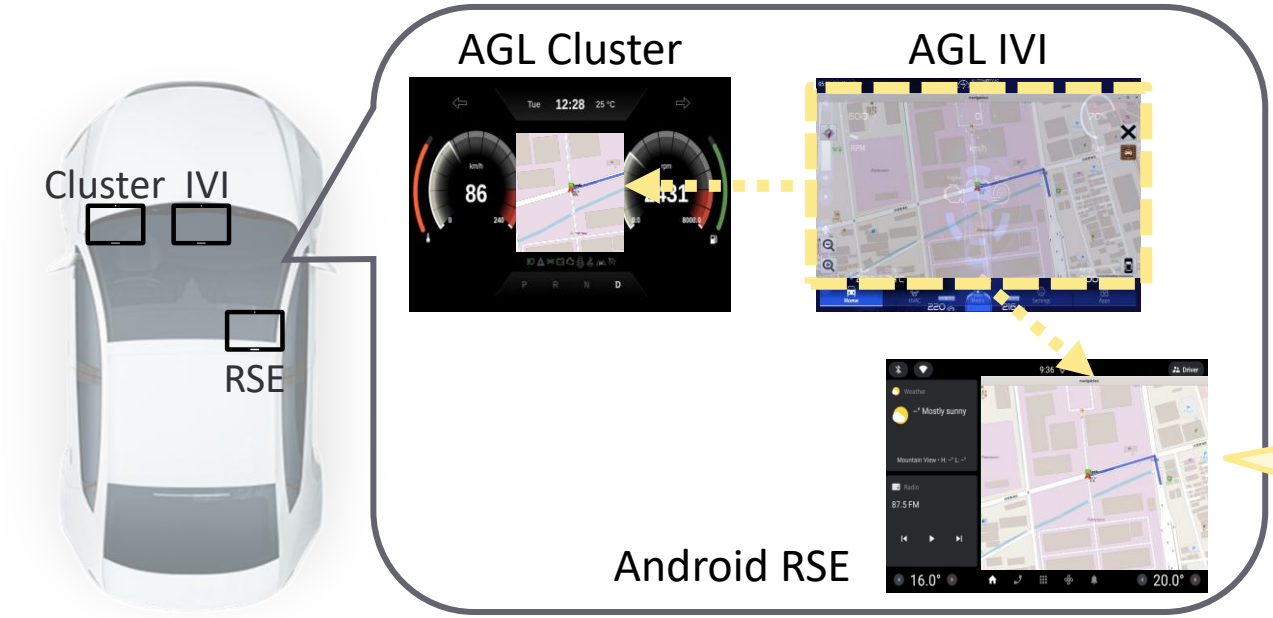
Panasonic AUTOMOTIVE

Unified HMI v3.0 focuses on **usability and developer experience**.



1. New simplified APIs to reduce manual steps and JSON handling for developers

2. Add UHMI support on AGL SoDeV

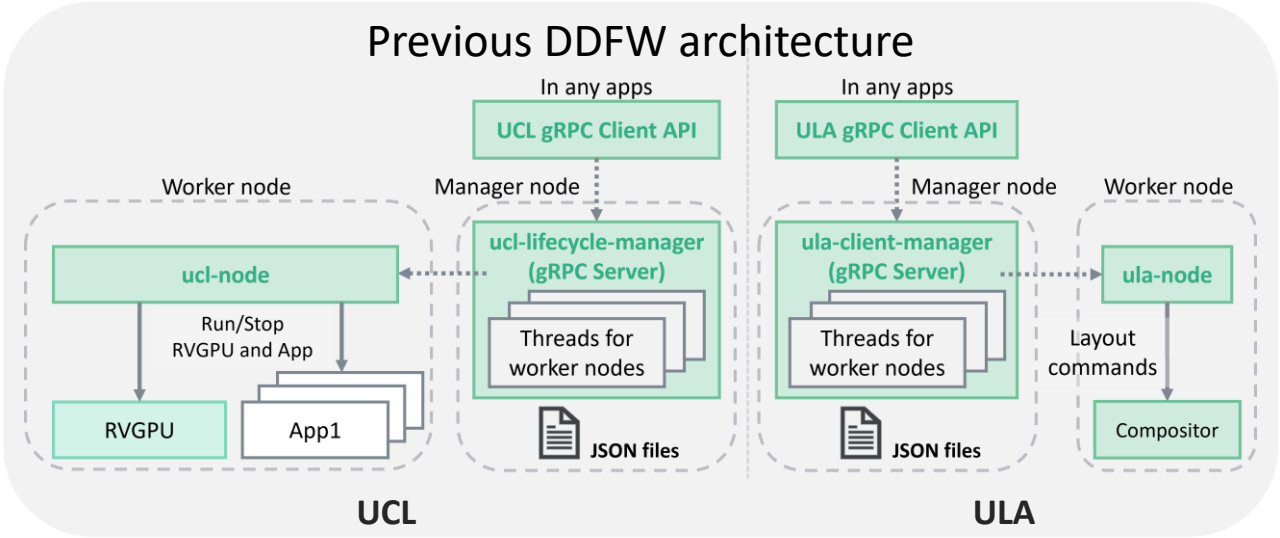


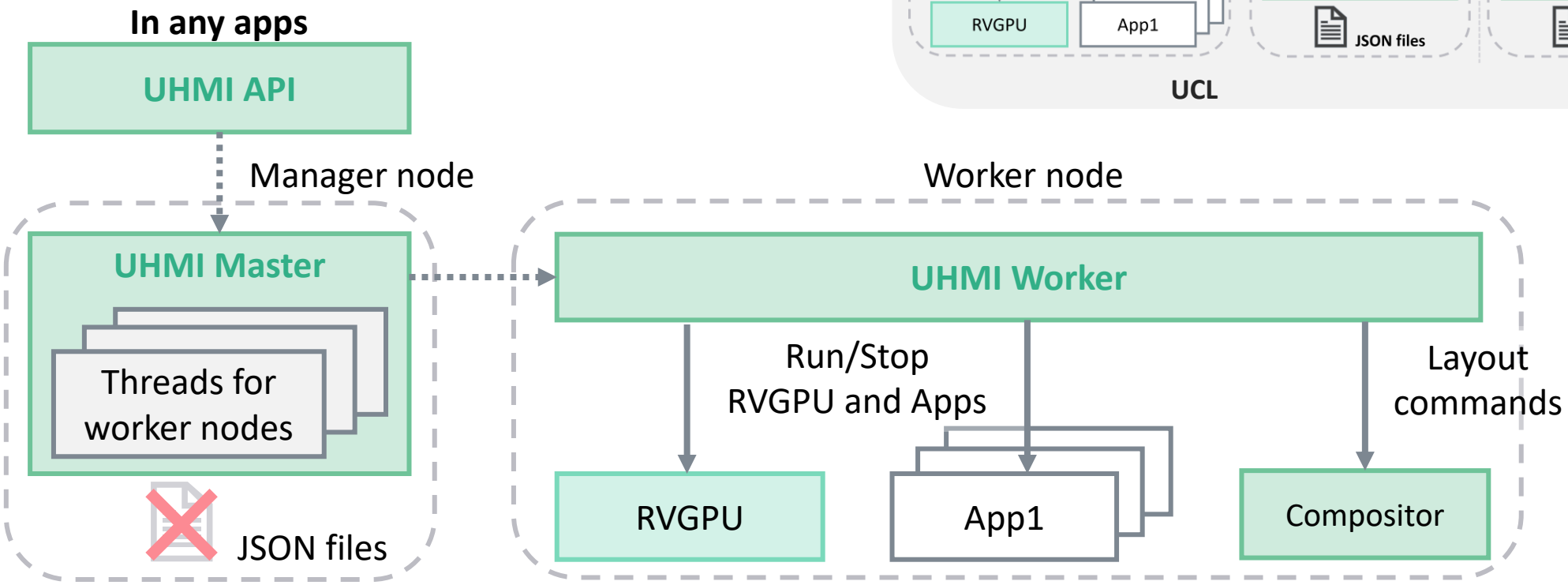
3. New use case: Render AGL apps to AAOS and control layouts to support SoDeV use cases

Map data from OpenStreetMap

Unified HMI v3.0: Simplified architecture

- ✓ Merge UCL/ULA functions
- ✓ **Reduce manual steps**
- ✓ **Dynamic** layout control **without JSON files** (JSON file defines display env is necessary)



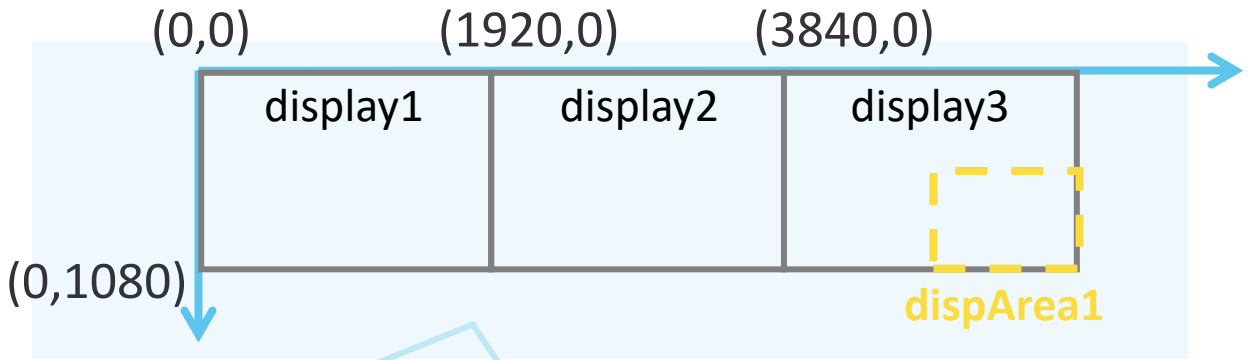


Dynamic layout control APIs without JSON

Panasonic AUTOMOTIVE

- ✓ Dynamically control layouts while apps are running **with some animation pattern.**
- ✓ **New customized display areas** can be defined in the JSON file.

Virtual screen



JSON file

display1 : x=0, y=0, w=1920, h=1080
display2 : x=1920, y=0, w=1920, h=1080
display3 : x=3840, y=0, w=1920, h=1080
dispArea1: x=4800, y=540, w=960, h=540

Real displays

```
$ moveApp horse display1
$ moveApp jerryfish dispArea1
$ wideApp navi display2 display3
```

Map data from OpenStreetMap

JSON Creation tool

Panasonic AUTOMOTIVE

- ✓ Web application that support creation of Json file **with simple GUI**.
- ✓ Input forms and arrange virtual displays visually using the mouse.

Input Forms

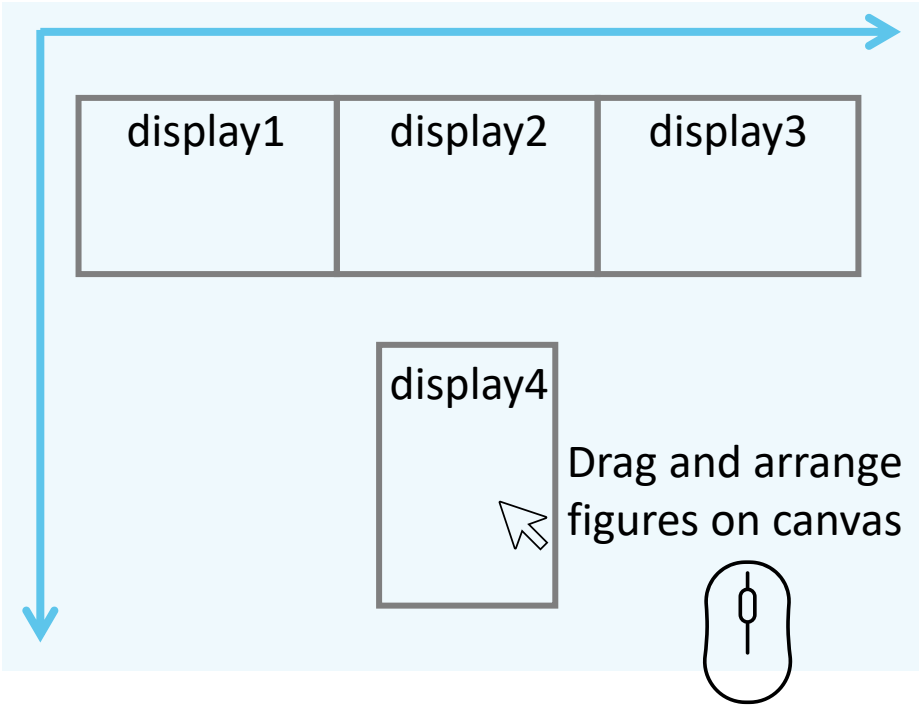
Node

Node Name	IP Address	OS	Environment Variables	Enable/Disable	Delete
Qualcomm SA8255 SoC	192.168.97.0	linux			
Reneses R-Car H3 SoC	192.168.97.0	agl			

Display

Phisical Display Information					Display Arrangement on Virtual Screen				Delete
Screen name	Connected node	Display number	Display resolution		Display Position		Display Size		
			Width	Height	x	y	Width	Height	
IVI display	Qualcomm SAi	0	1920	1080	2340	1080	1080	1920	
Cluster display	Qualcomm SAi	1	1920	1080	0	0	1920	1080	
Center display	Qualcomm SAi	2	1920	1080	1920	0	1920	1080	
Android tablet	Qualcomm SAi	3	1920	1080	3840	0	1920	1080	

Virtual display arrangement



UHMI API examples

Panasonic AUTOMOTIVE

With UHMI v2.0, launch app and control layouts **need three steps and JSON files**:

```
$ ucl-api-comm -c launch_compositor_async                    # launch RVGPU receivers
$ ucl-api-comm -c run_command <path to app.json>             # launch apps via RVGPU
$ ula-grpc-client -c DwmSetLayoutCommand <path to initial-vscreen.json>    # control layouts
```

With UHMI v3.0, it needs **only 1 step without JSON file**:

```
$ uhmi-api –c runApp AppName dispName        # launch apps via RVGPU and layout them
```

Advanced API examples:

```
$ uhmi-api –c
      - moveApp appName dispName          # move app to the indicated display
      - animationApp appName dispName     # move app to the indicated display with animation
      - wideApp appName disp1 disp2       # span app across disp1 and disp2
```

Demo video: How UHMI v3.0 APIs can be used

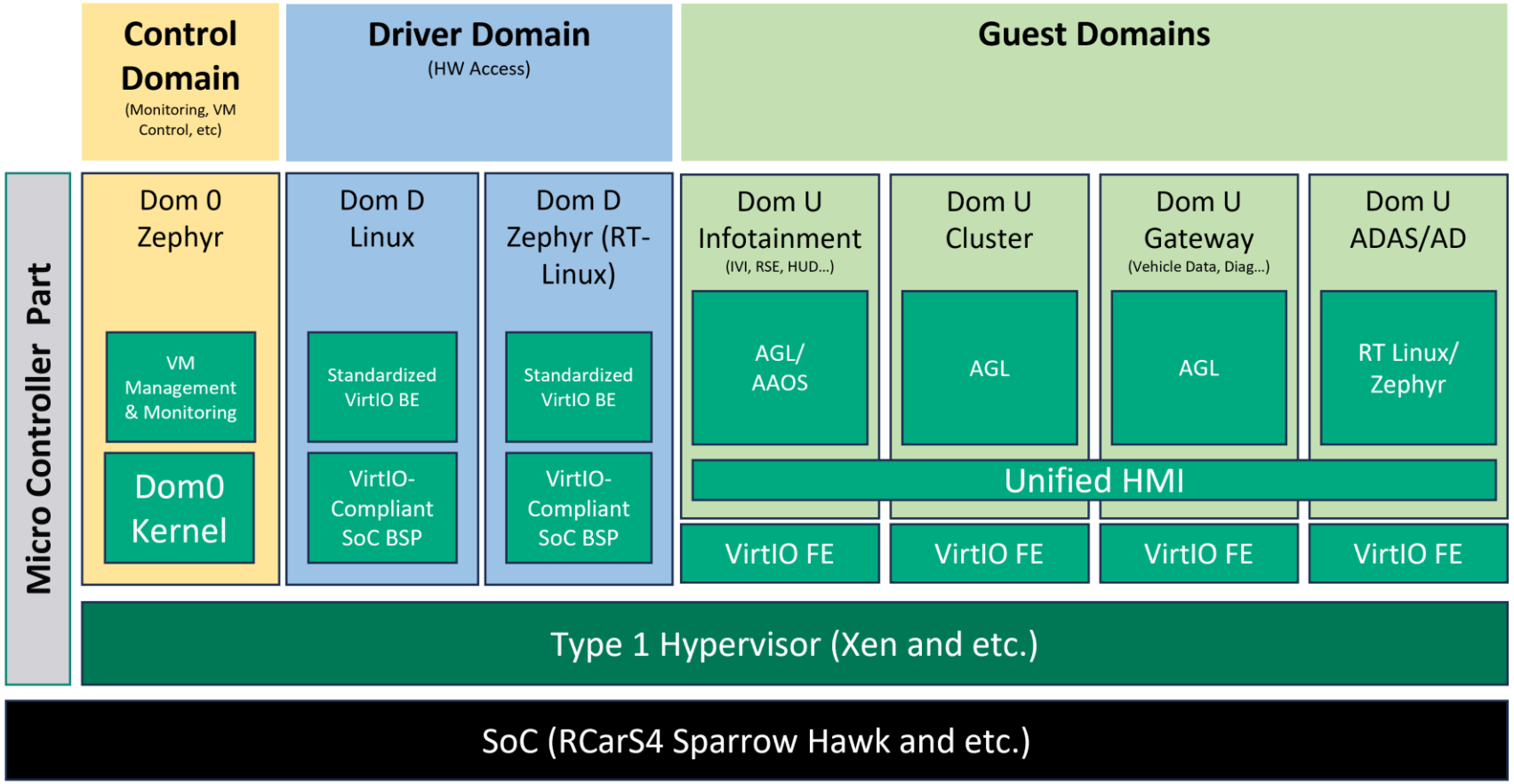
Panasonic AUTOMOTIVE

Map data from OpenStreetMap

Why is Unified HMI necessary for AGL SoDeV?

Panasonic AUTOMOTIVE

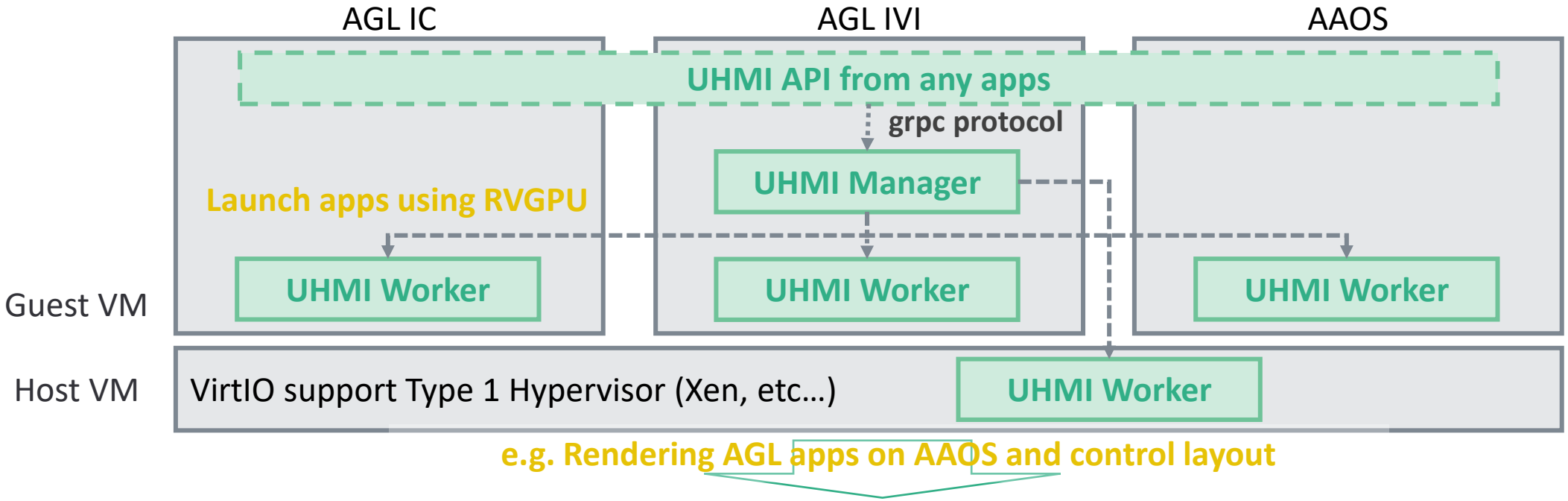
- ✓ Enable **a common and open-source** multi-display virtualization framework.
- ✓ Flexibly developed **with AGL community**



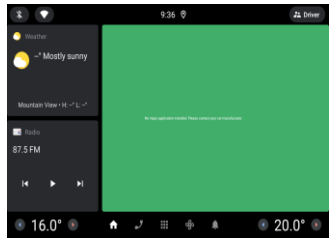
AGL SoDeV BlockDiagram

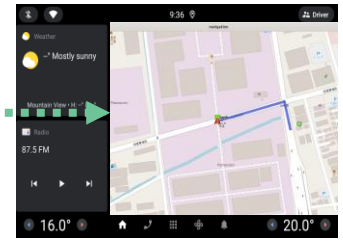
Unified HMI Reference Architecture on AGL SoDeV

Panasonic AUTOMOTIVE



e.g. Rendering AGL apps on AAOS and control layout

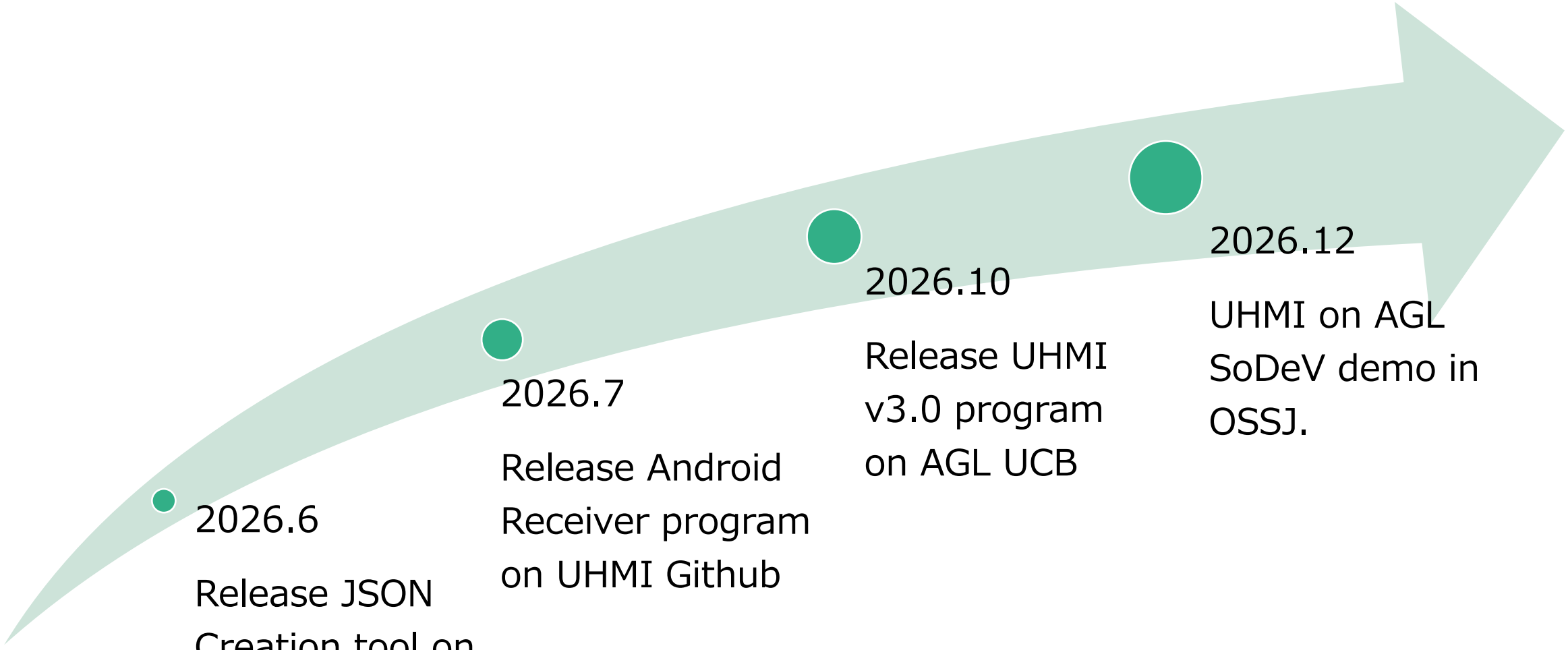




Map data from OpenStreetMap

Milestone for 2026

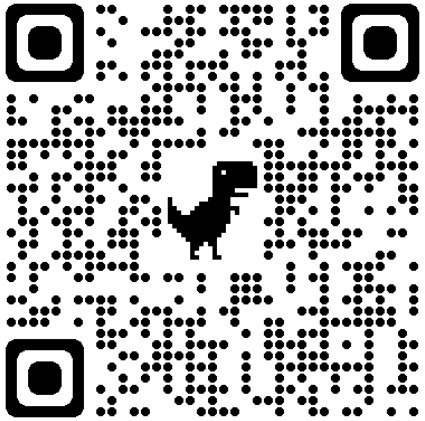
Panasonic AUTOMOTIVE



How to access Unified HMI

Panasonic AUTOMOTIVE

GitHub



- ✓ Access the source code for Unified HMI
- ✓ Compatible with Ubuntu for easy development
- ✓ **Please contribute to the project and report issues!**

AUTOMOTIVE GRADE LINUX

- ✓ Unified HMI is a **standard component of AGL**
- ✓ Includes **meta-uhmi Yocto recipe** for installing Unified HMI
- ✓ Supporting Platforms:
  QEMU, Raspberry Pi 4, and AGL Reference Hardware