

Tizen Developer Conference



Approach of In-Vehicle Infotainment
development on open source software

May22-24, 2013
@San Francisco
Takeshi Hoshina

Self-introduction



Agenda

- What is IVI system?
- How We collaborate d with Tizen IVI
- Introduction of Toyota source code
 - UI Manager
 - Vehicle Information Control (VIC)
 - Home screen
- Summary
- Q&A

What is IVI system?

Portable Device



IVI



Rear view camera

Other ECU



RSE

Example

In-Vehicle Infotainment (IVI) system i.e. Navigation, Multimedia, Telematics/Call Assistant
Portable Device and other ECUs are connected to IVI

How We collaborate d with Tizen

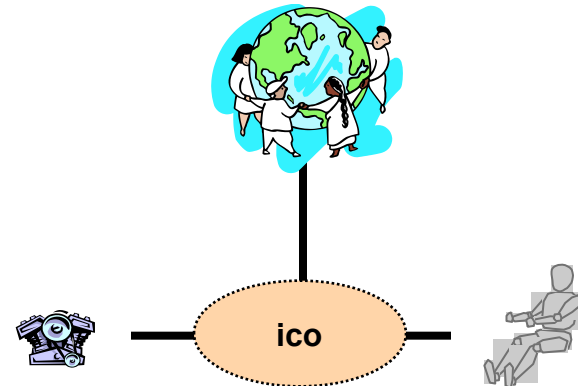
Toyota developed reference platform based on Tizen IVI

Released SW packages to tizen.org include

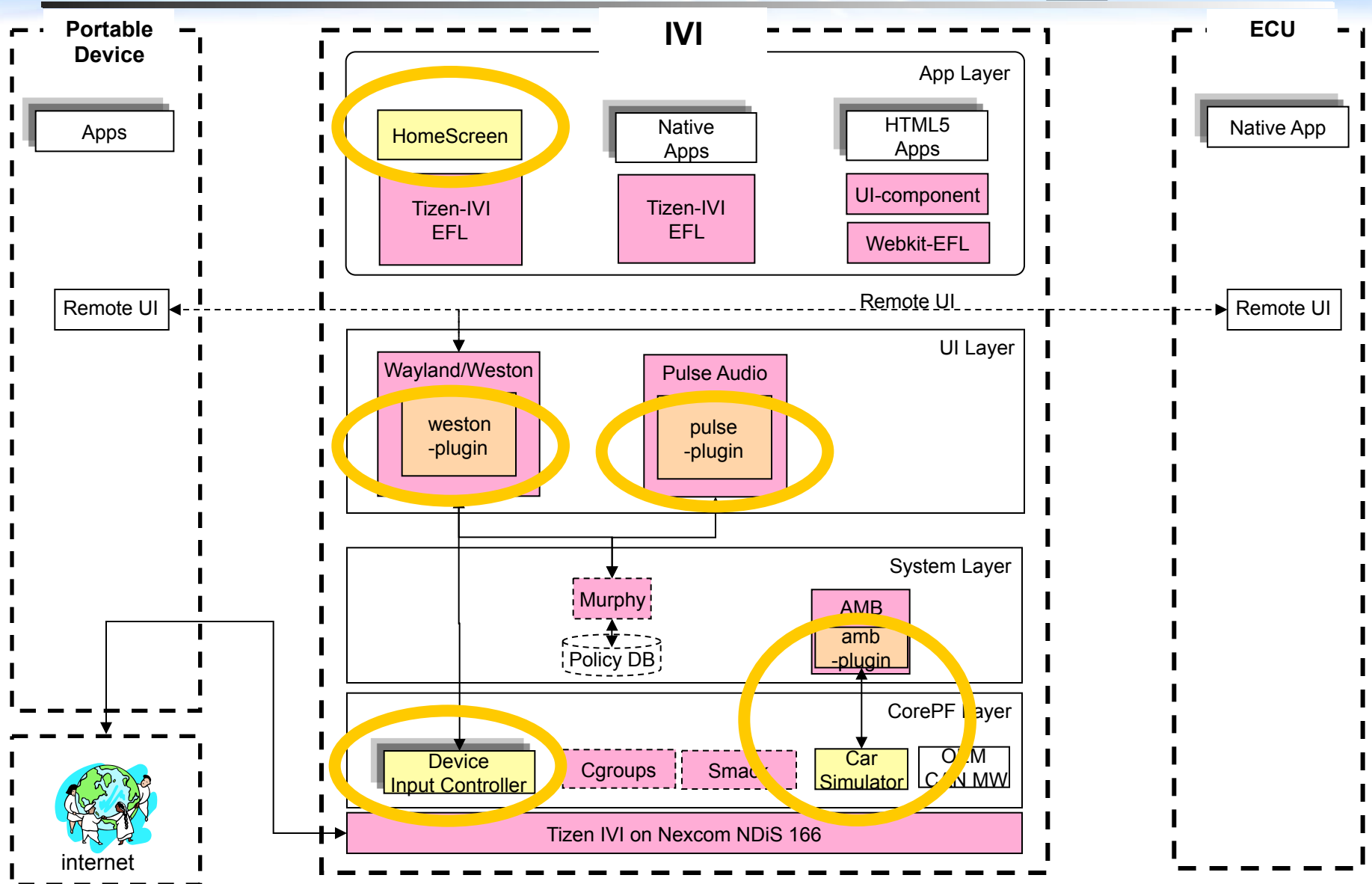
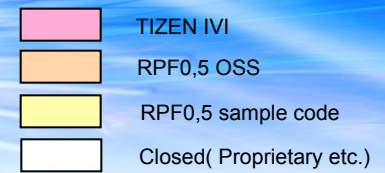
- Weston plugin
- Device input controller
- Pulse Audio plugin
- Vehicle information control (VIC) plugin for automotive message broker
- Car simulator
- Sample home screen
- Sample apps

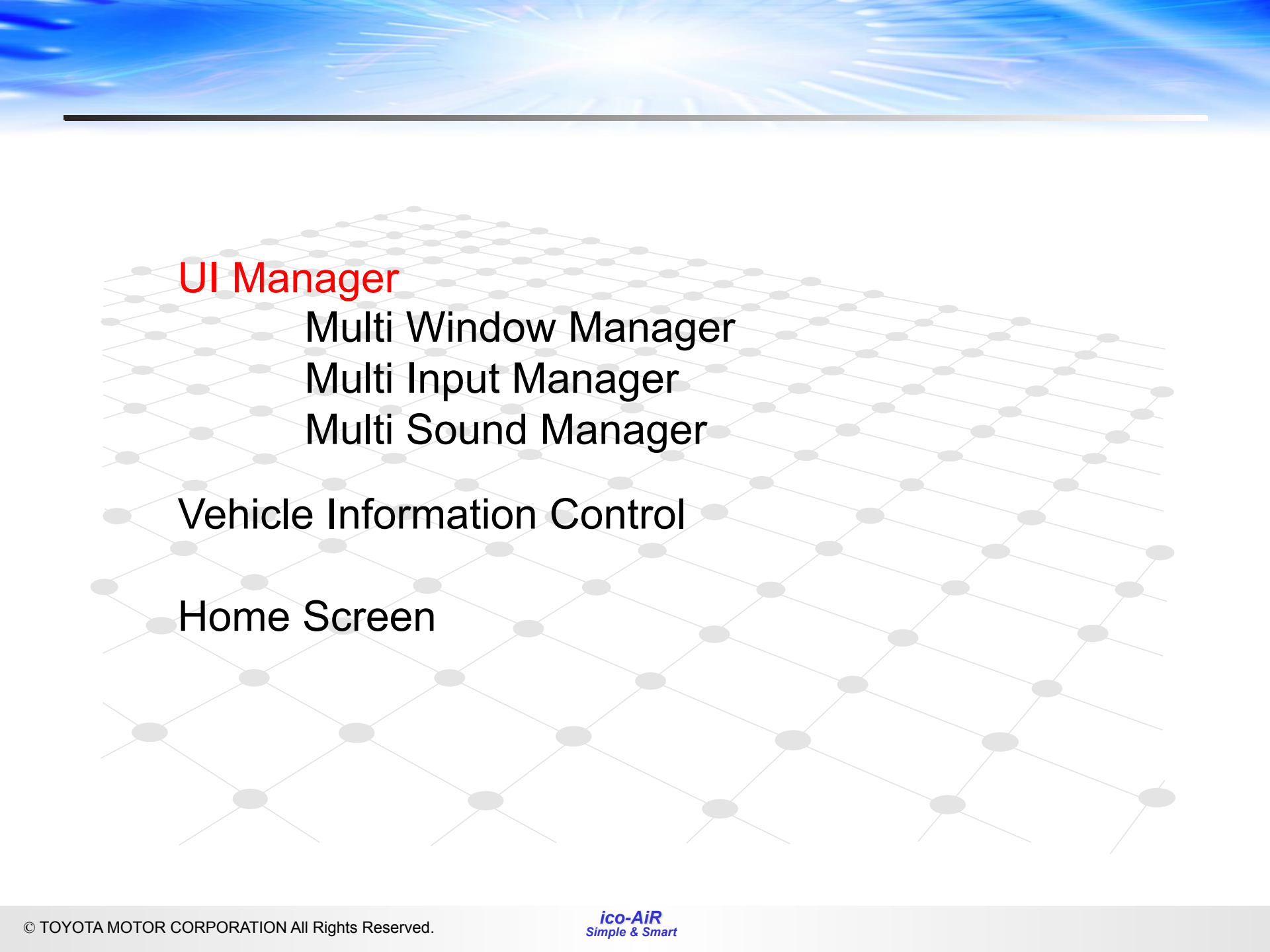
Reference Platform Ver0.5 (RPF0.5)

Connecting Car, Driver and Internet



RPF0.5 on Tizen IVI





UI Manager

Multi Window Manager

Multi Input Manager

Multi Sound Manager

Vehicle Information Control

Home Screen

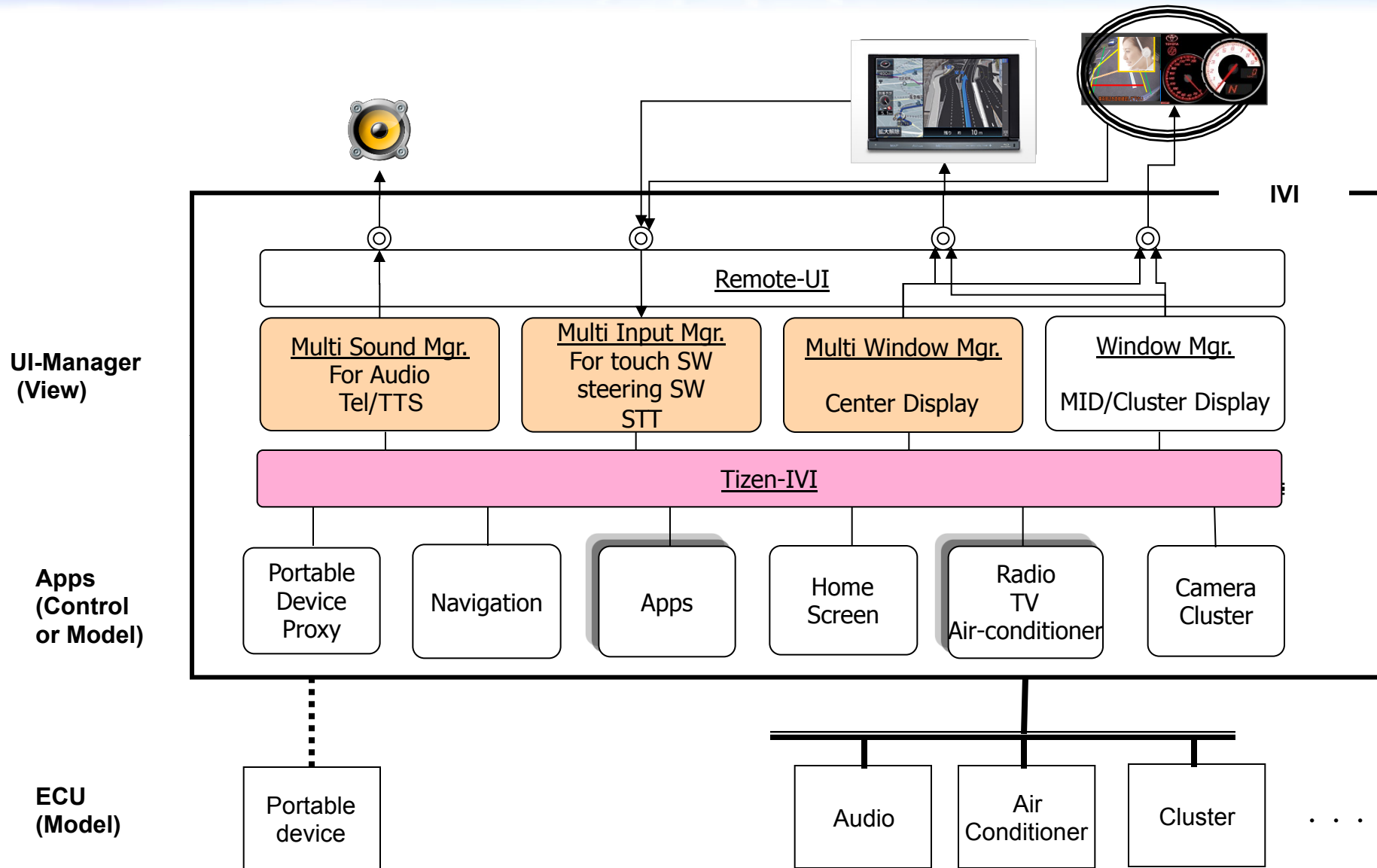
Requirements

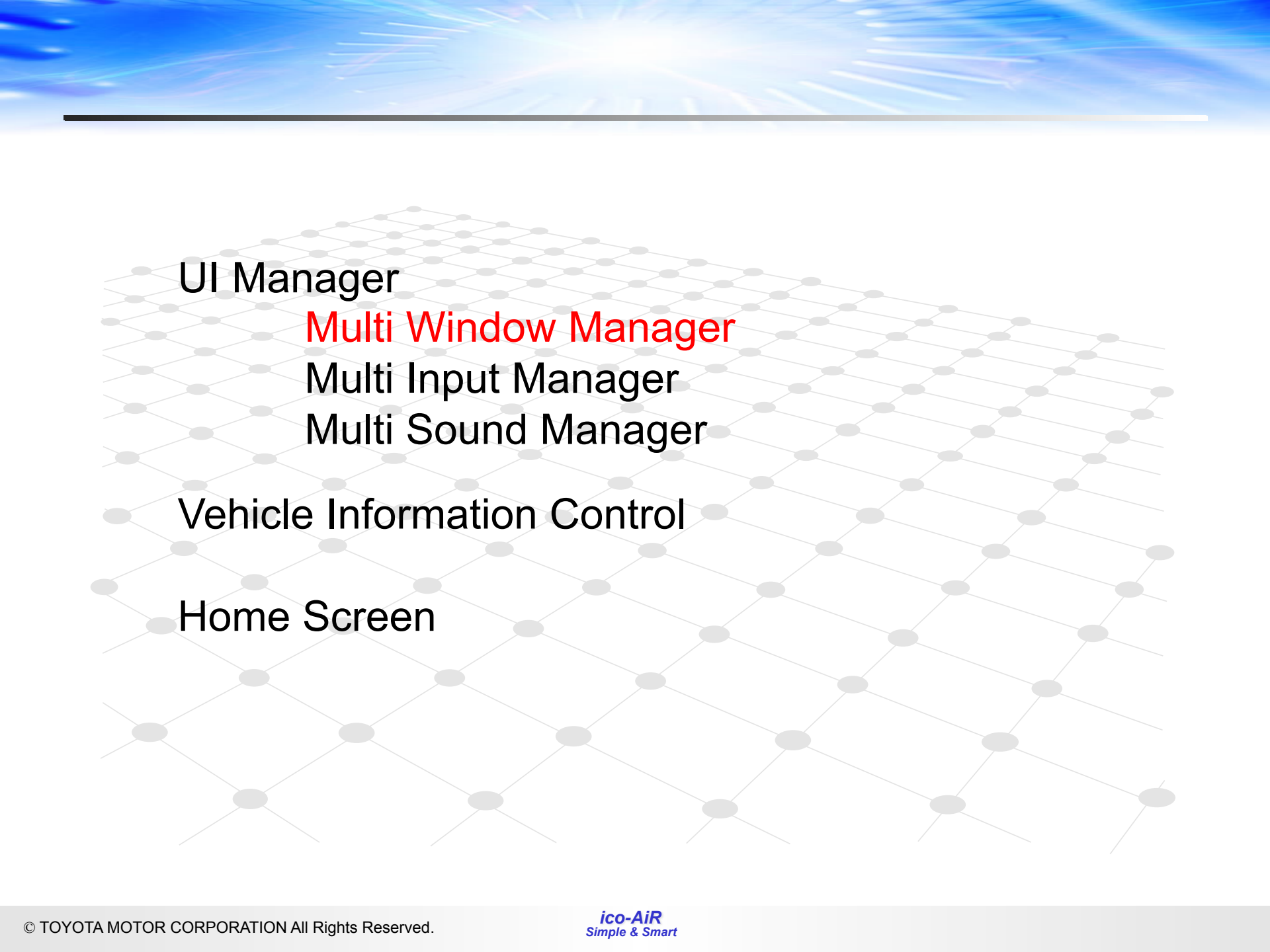
- Easy-to-use user interface
- Simple and Smart architecture to reduce SW complexity
- Robustness under limited memory resource on embedded CPU
- Automotive reliability
- Mobile Apps portability

Toyota UI-Manager

- Utilize OSS community efforts and common code base
- Adapt software layer architecture
- Use plugin and common IPC
- Support mechanism to avoid driver distraction

Role of UI-Manager





UI Manager

Multi Window Manager

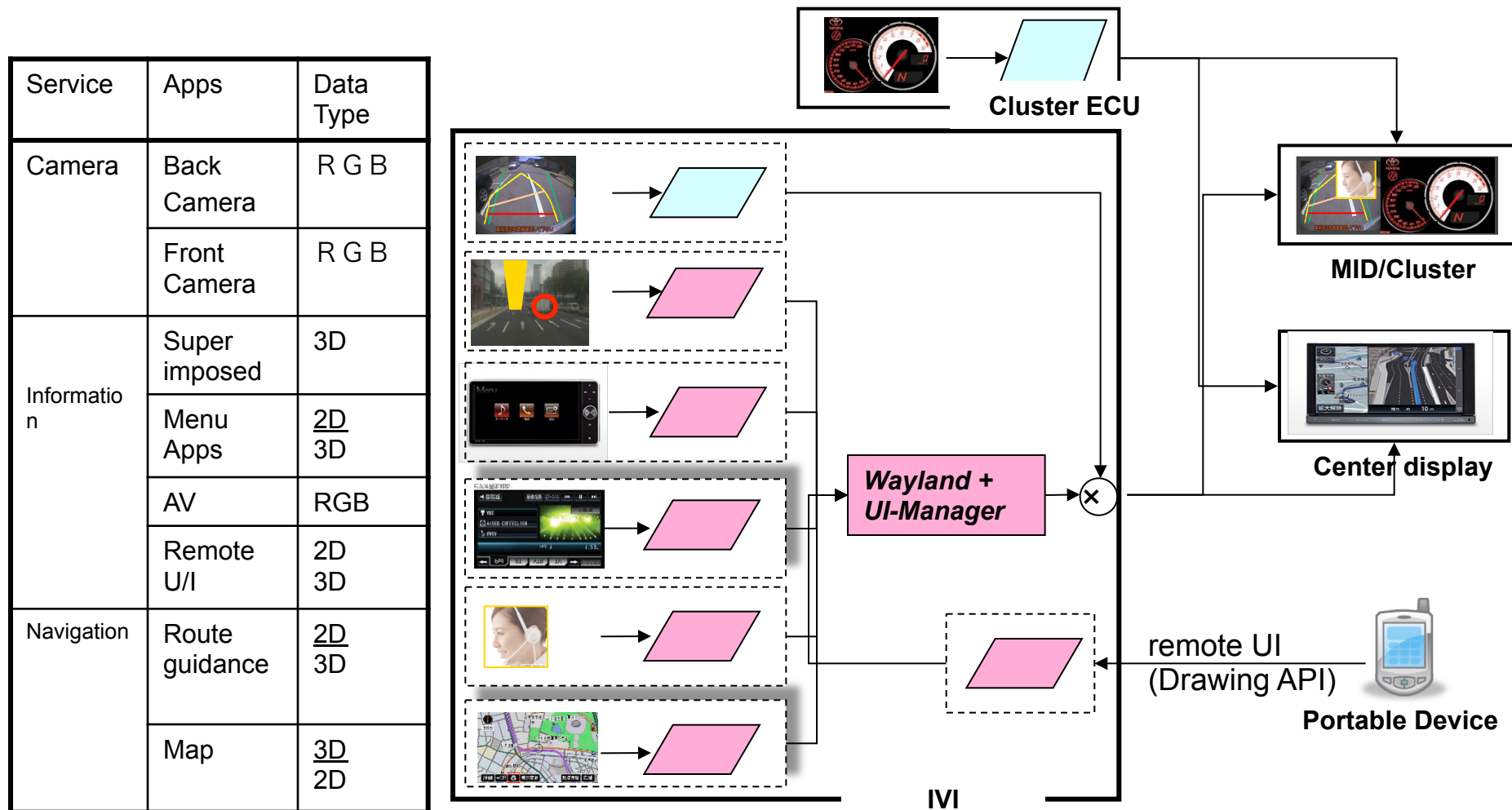
Multi Input Manager

Multi Sound Manager

Vehicle Information Control

Home Screen

MWM Usage example with multi layers



MWM functional requirements

- Sophisticated user interface
 - Multi layers for IVI
 - High performance
 - Wayland/Weston plugin
 - High scalability
 - Pluggable OEM specific drawing
- App coexistence with car OEM applications
 - Monitor Window operation of applications
 - Apps execution environment (Web, Native)

Why do we want to use Wayland/Weston?

- Simple and Smart

- Wayland/Weston = Display Server + Window Manager
- Direct rendering (No use of X11 rendering protocol)

- High scalability

- Easy to add OEM original features
 - Drawing method (3D animation, etc)
 - Car OEM original UI devices (Haptic device, etc)

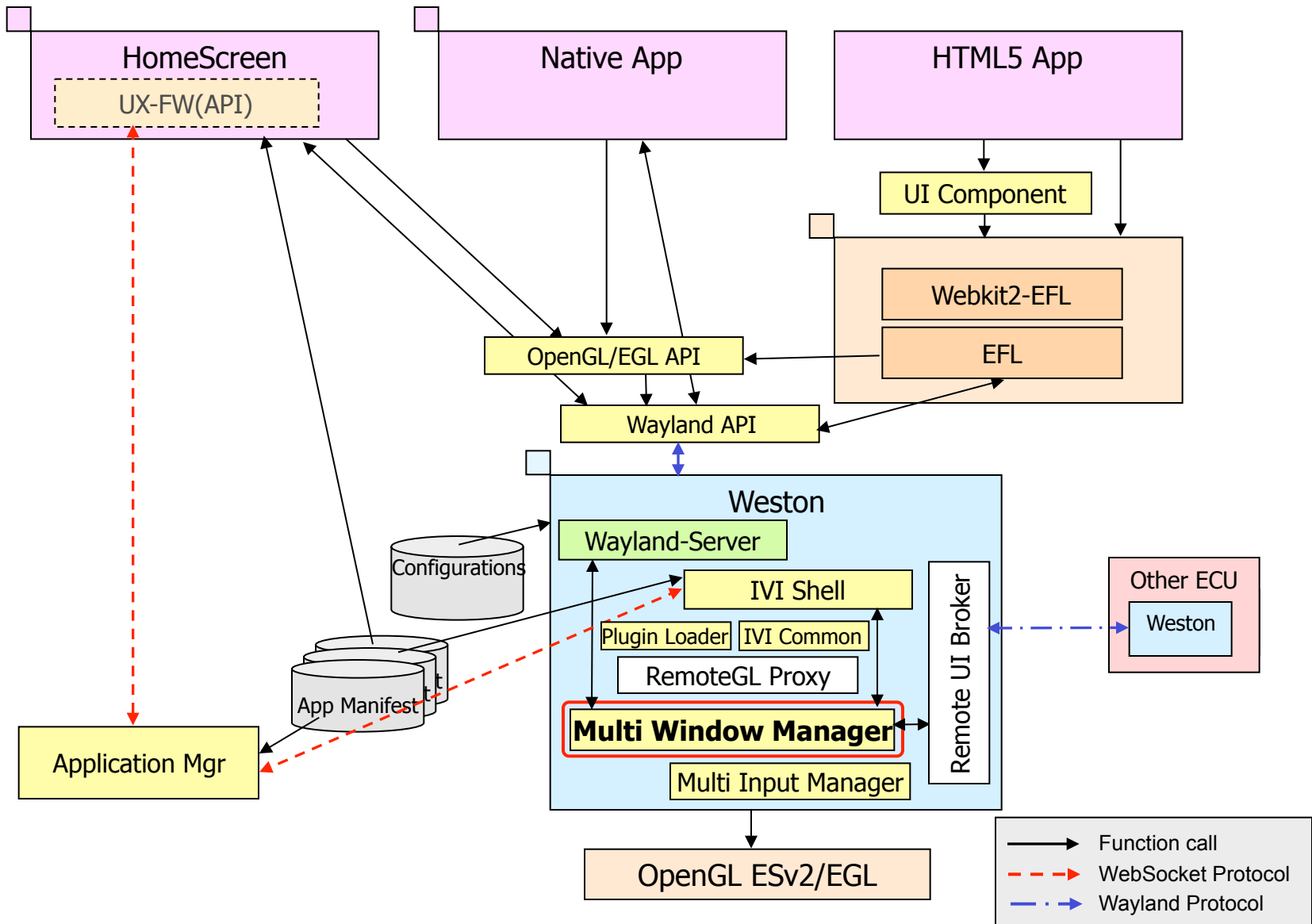
MWM feature(RPF0.5)

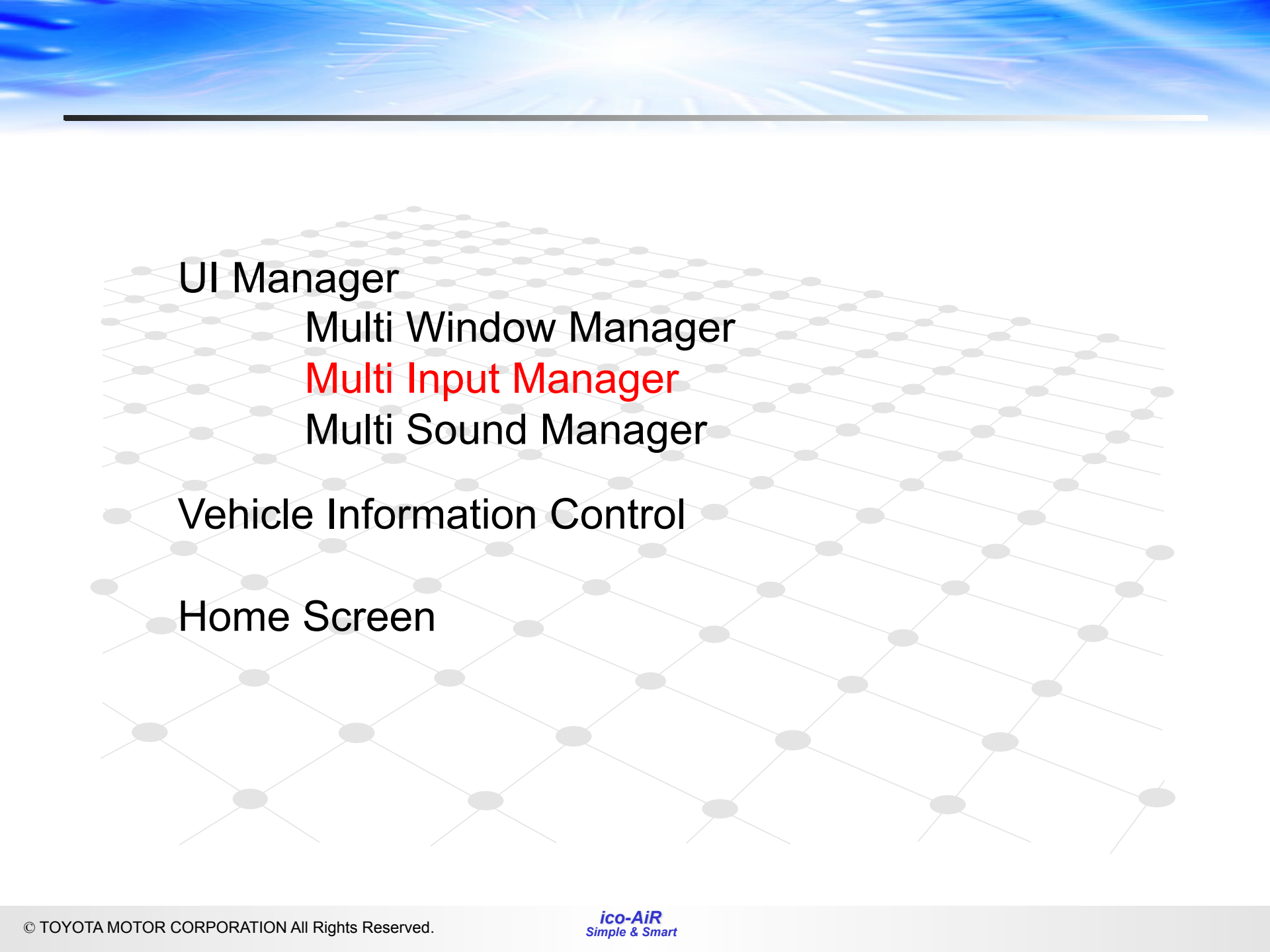
- Basic window functions
 - Create and delete two or more windows
 - Manage it's position, width and height, visible/invisible and effects
 - Manage two or more windows within one display device

- Window Layer Management
 - Free arrangement of the layers
 - Batch control of windows on the layers

- Window Layout Management (2 types)
 - Compositing window manager
using Normal Apps , Pop-Up and so on.
 - Tiling window manager
using Home-Screen and so on

MWM software block diagram (RPF0.5)





UI Manager

Multi Window Manager

Multi Input Manager

Multi Sound Manager

Vehicle Information Control

Home Screen

MIM functional requirements

- Sophisticated user interface
 - High efficiency
 - Weston plug-in
 - High scalability
 - Can add OEM specific input device

- Manage input events
 - Abstract Input Event
 - Map Input Event and each application
 - Arbitrate Input Event conflict
 - Suppress Input Event by vehicle condition

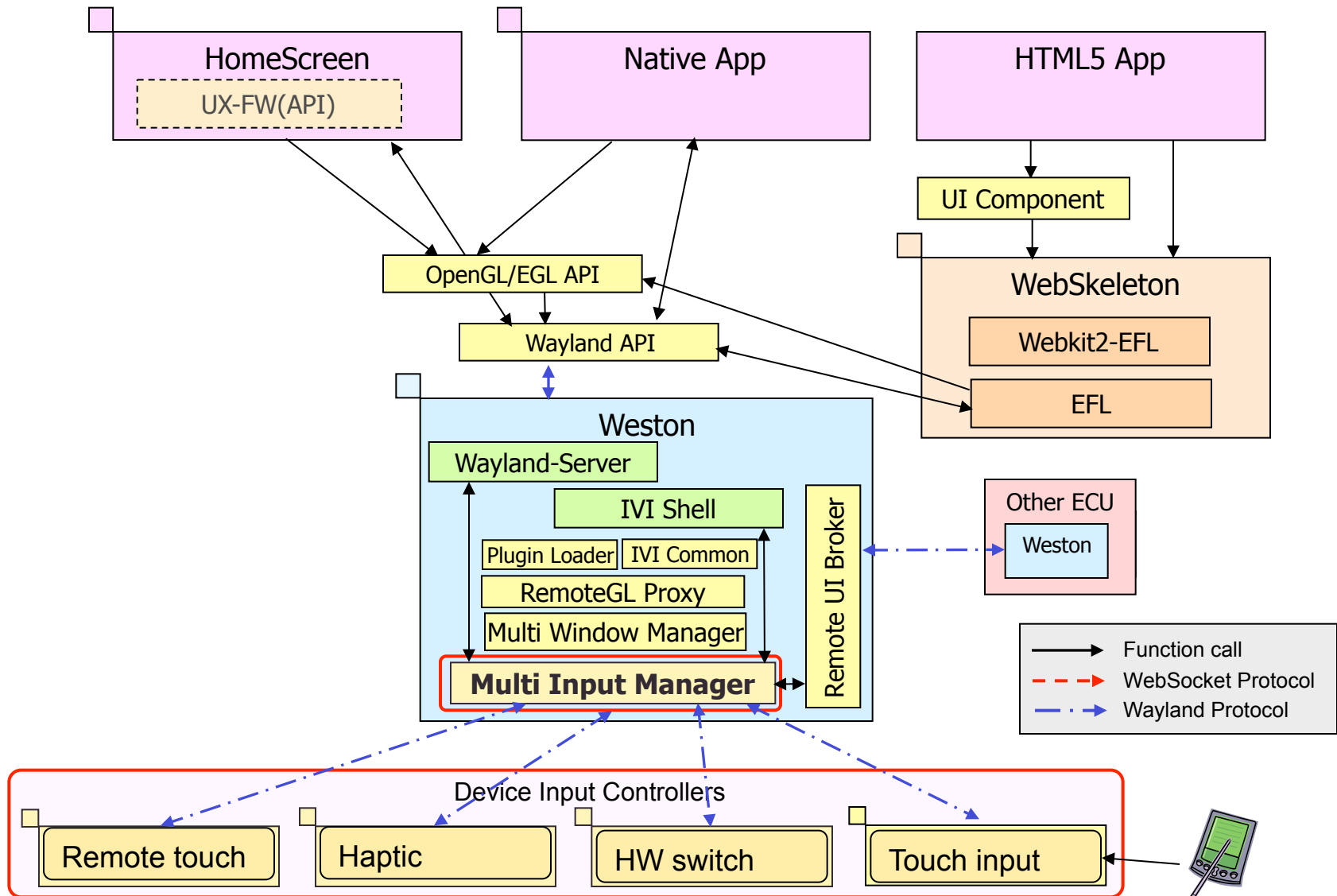
Near future,

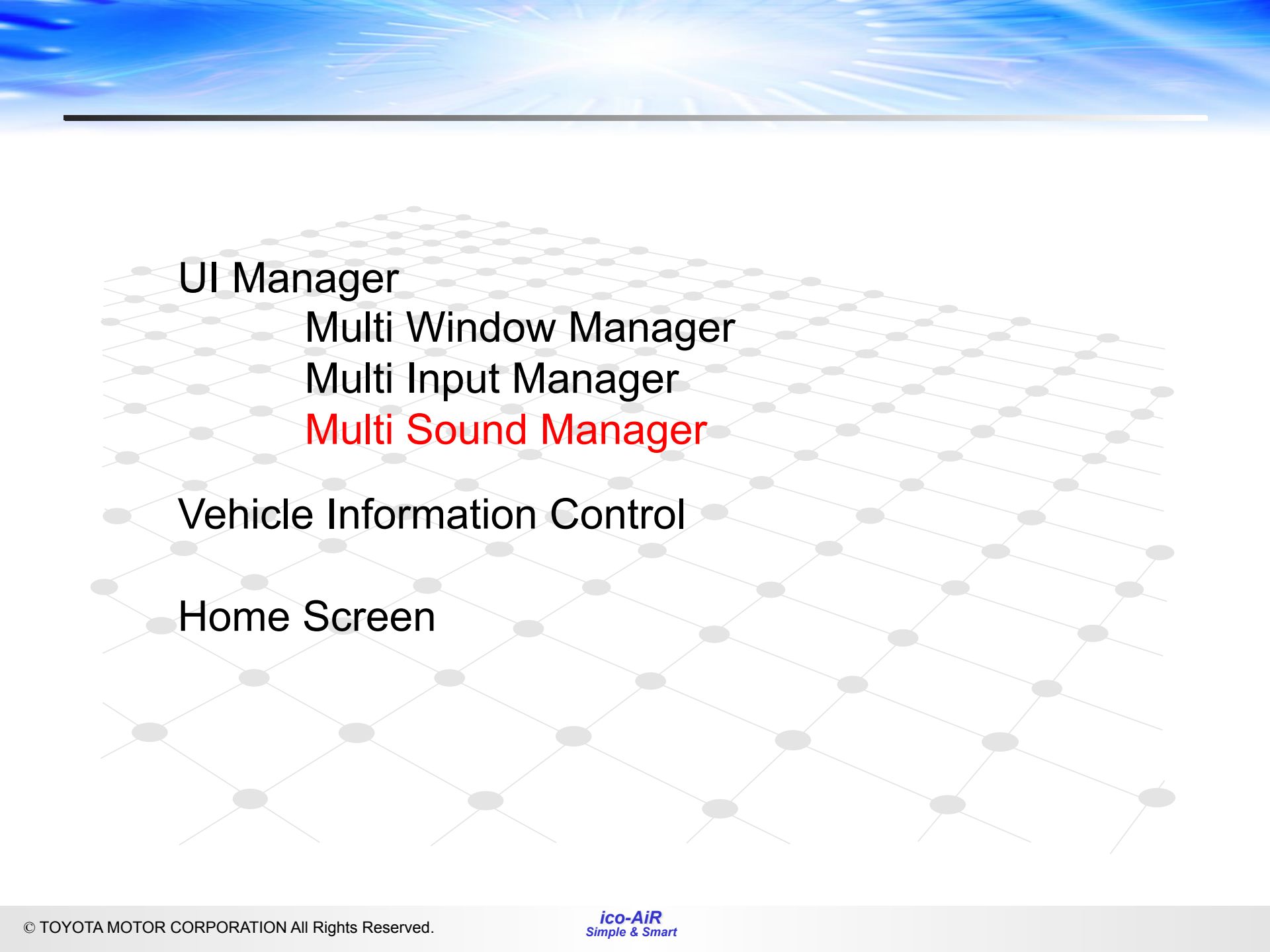
- MIM manages the input data even for other ECU

MIM feature (RPF0.5)

- Basic Input Device
 - Mouse (Touch-Screen), Keyboard, Joystick
- Mapping Apps and Input devices
 - Mapping by rule-based DB
 - Deliver the event to the apps
- Support Car OEM specific devices
 - Device Input Controller

MIM software block diagram (RPF0.5)





UI Manager

Multi Window Manager

Multi Input Manager

Multi Sound Manager

Vehicle Information Control

Home Screen

MSM functional requirements

- Sophisticated user interface
 - High efficiency
 - pulse audio plugin
 - High scalability
 - Pluggable Car OEM specific sound functions
- App Coexistence with car OEM apps
 - Apps execution environment (Web, Native)
 - Monitoring and Arbitration sound operation(*) of each application
 - (*) Mixing/attenuation/pause/Mute/Cut off etc.
- Zone-control
 - Multi sound zone (front and rear etc.)

Near future,

- Remote-UI
 - MSM manages the sound (in/out) even for other ECU

MSM feature (RPF0.5)

- Basic Audio functions

- Dynamic Volume Control
- Dynamic Mixing Control

User operation (Audio apps , Mute button etc.)

Other situation (Phone call etc.)

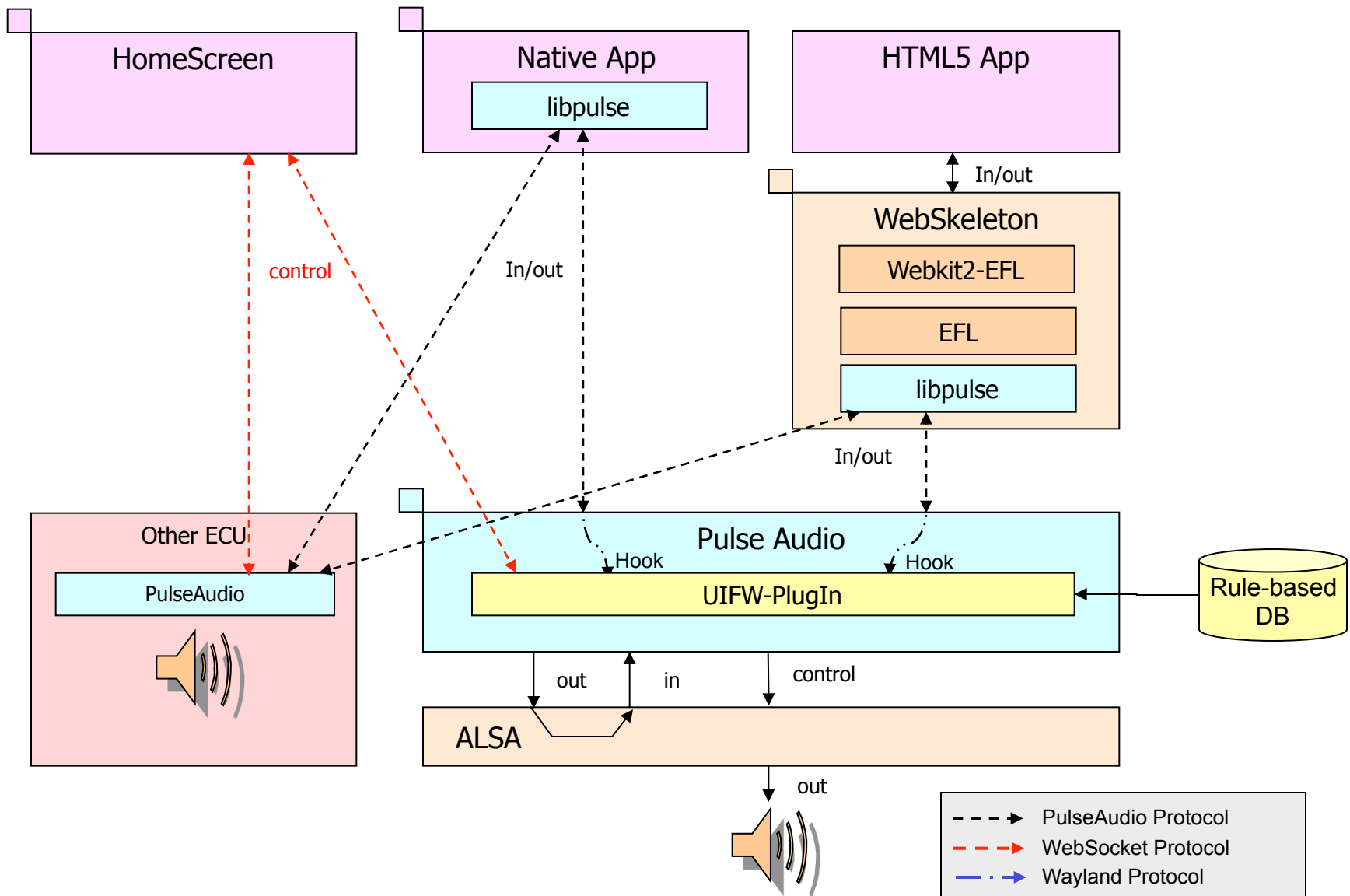
- Manage apps using Sound

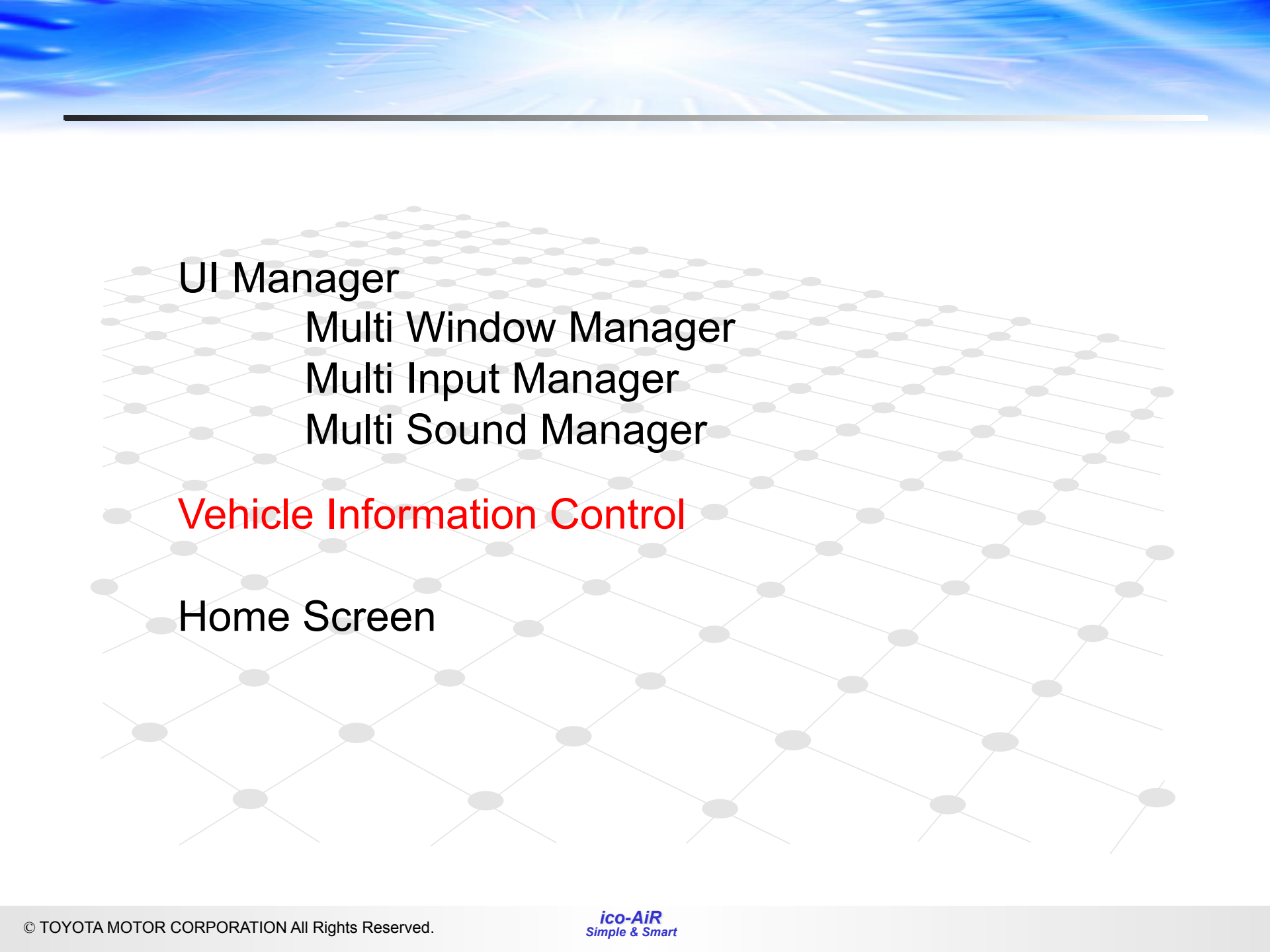
- Arbitrate sound operation from each application
(Priority scheduling by rule-based DB)

- Cooperation with the Home screen

- Mute , Cancel, get attribute
- notify status change to application

MSM software block diagram (RPF0.5)





UI Manager

Multi Window Manager

Multi Input Manager

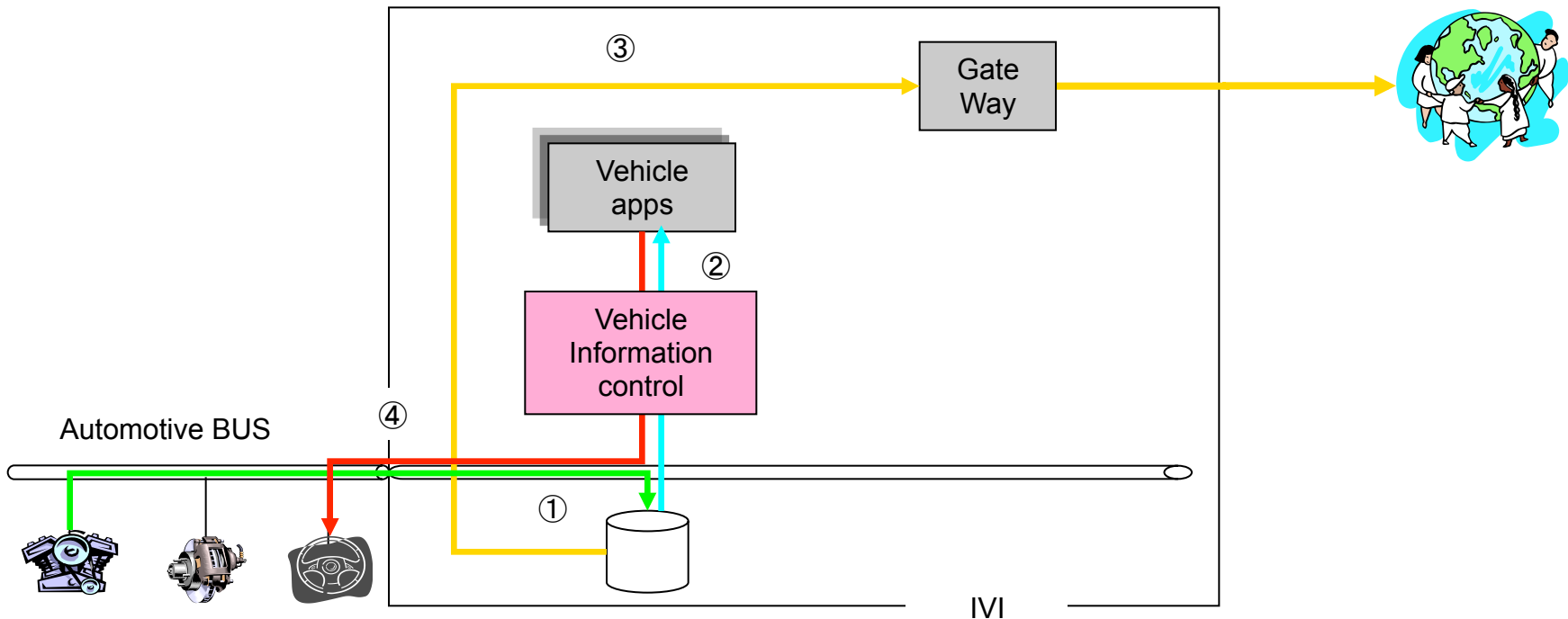
Multi Sound Manager

Vehicle Information Control

Home Screen

VIC functional requirement

- ① VIC caches vehicle data sent by the ECU.
- ② Vehicle data is offered by the demand from the Vehicle apps.
- ③ VIC uploads vehicle data to the center.
- ④ Vehicle apps controls each ECU.

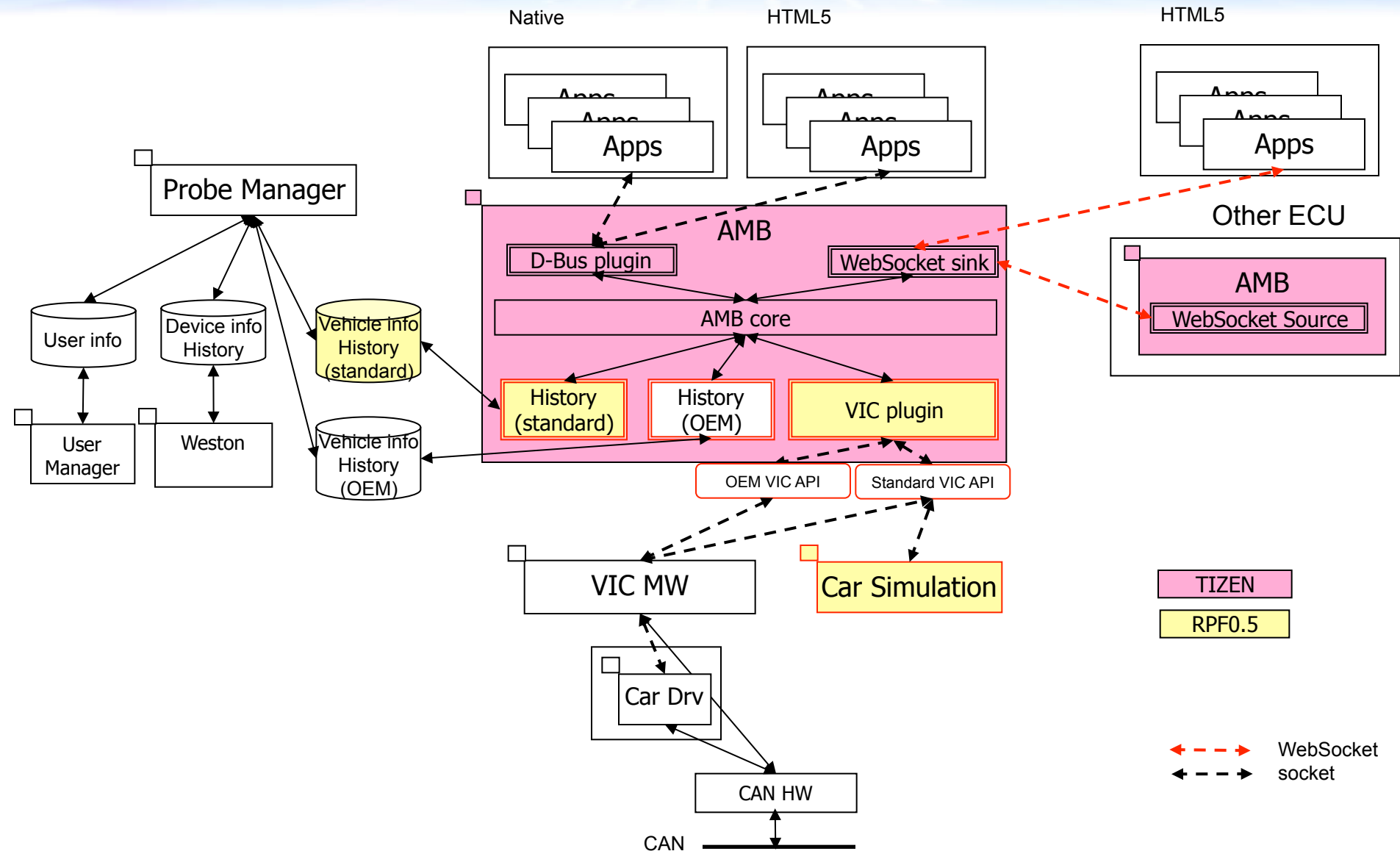


VIC feature (RPF0.5)

- High efficiency
 - Work as Automotive Message Broker plug-in
- Manage Vehicle Data
 - Abstract vehicle data
 - Data format for in-vehicle LANs may change
 - Two Data format type
 - Real-time data in the cache
 - History data in the vehicle-information-DB
 - Two API property type
 - Common API
 - OEM specific API (Non Disclosure)
- Car simulator for software debug
 - Can simulate the running state of the vehicle
Steering, Acceleration pedal, Brake Pedal etc.
 - Increase efficiency for debugging during vehicle test
API for car simulator and vehicle middleware are the same



VIC Software Block Diagram (RPF0.5)



【VIC topics】 Policy of API Definitions

Contents Provider needs open standard vehicle API

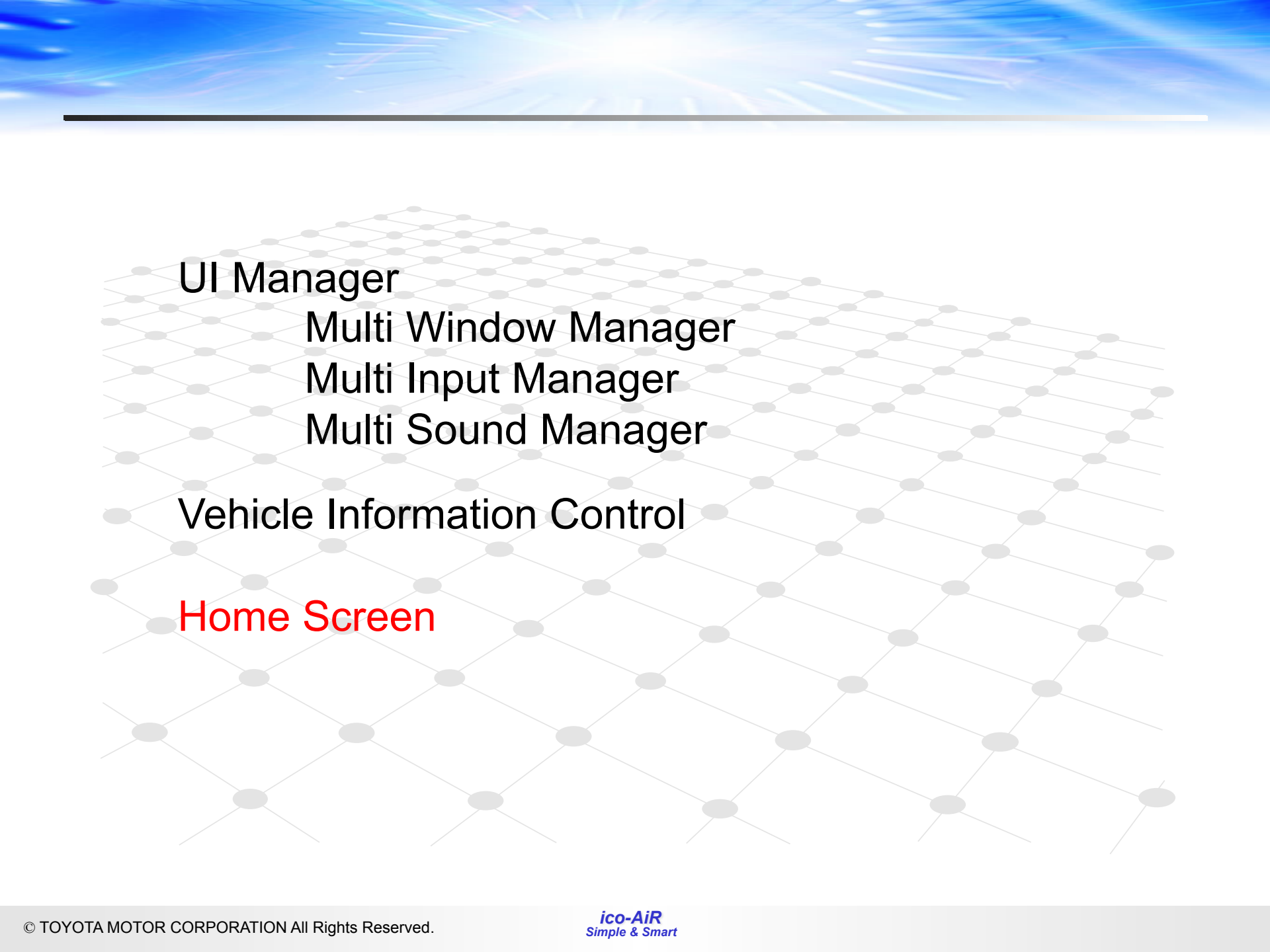
Car OEM can not support all standard API

- Depend on each car OEM
 - Personal information protection policy
 - Safety, Security policy
 - Business strategy (probe data property etc.)
- Depend on each vehicle model
 - Equipment of vehicle package option (Navigation, ADAS, etc)
 - Automotive LAN grade (e.g. not connected some ECU)

We need to define open standard API!

But,

We should be provided with the access control every API (Property)



UI Manager

Multi Window Manager

Multi Input Manager

Multi Sound Manager

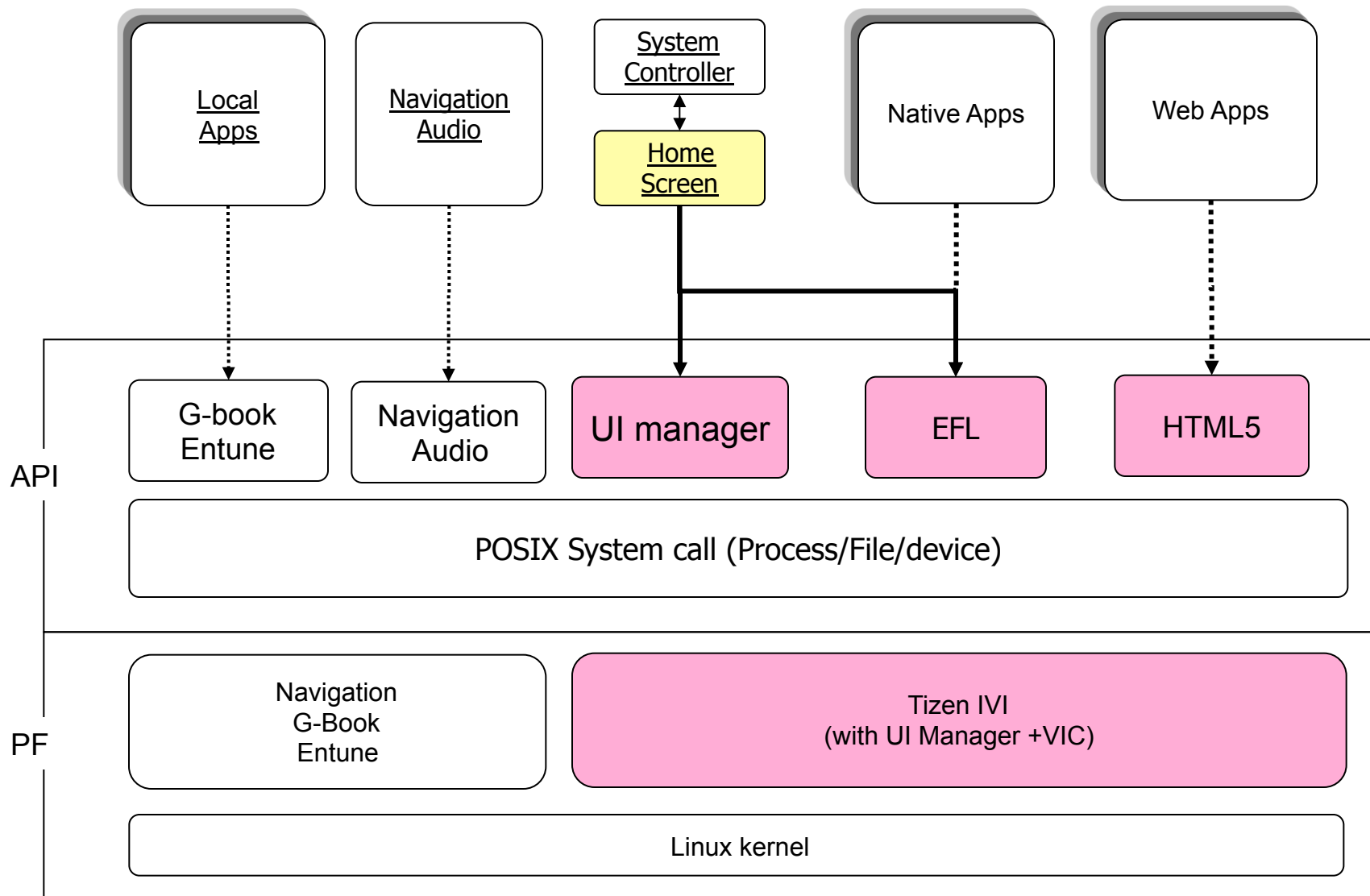
Vehicle Information Control

Home Screen

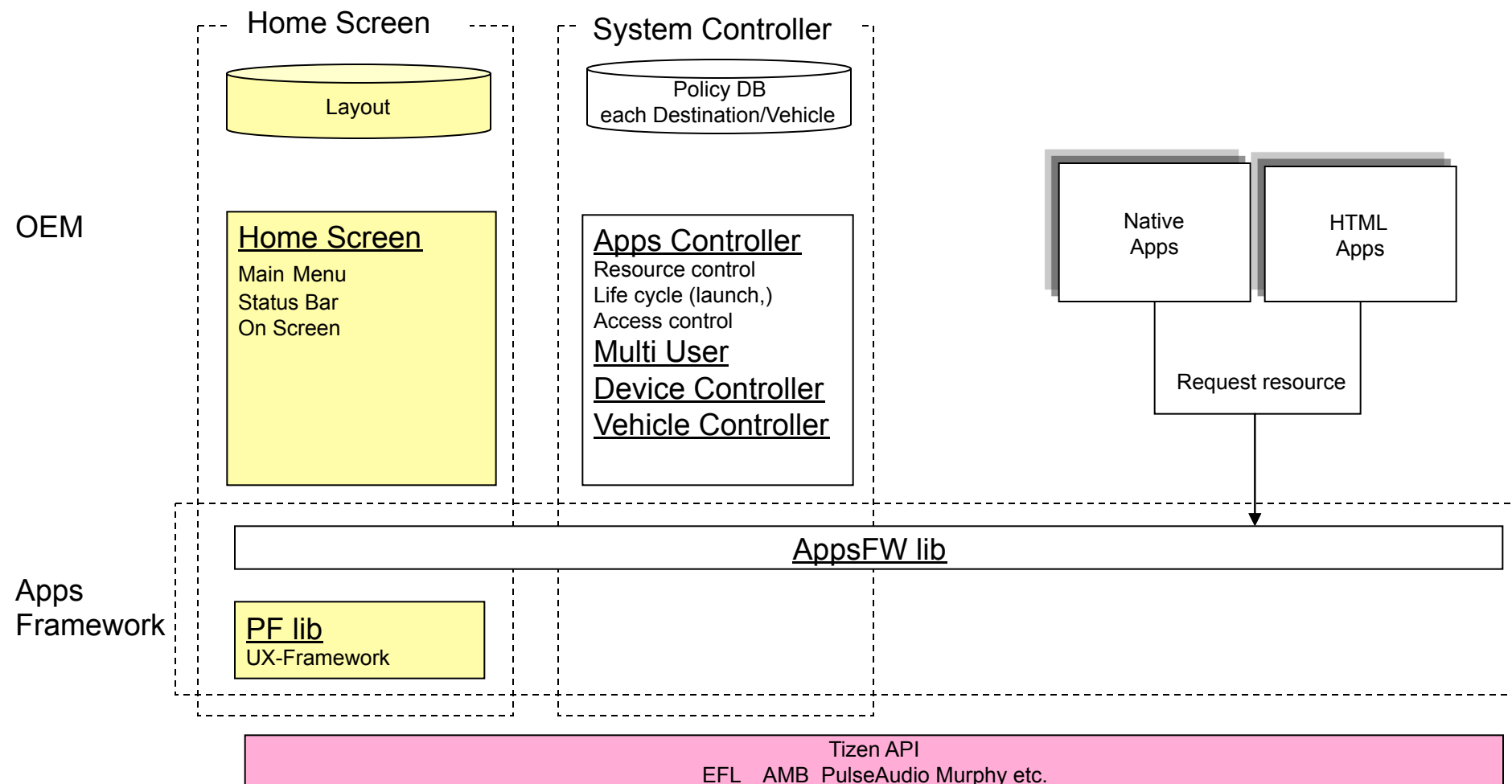
Home Screen feature (RPF0.5)

- User Menu
 - Simple Tile Menu (using Window Manager)
 - Status Bar
 - Launcher (with sample Apps)
 - On Screen
- Application control
 - regarding window, input event, sound
- Interrupt Action
 - Change View for Driver Distraction
(interrupted by vehicle inf. via Automotive Message Broker)

Overview of Sample Home Screen

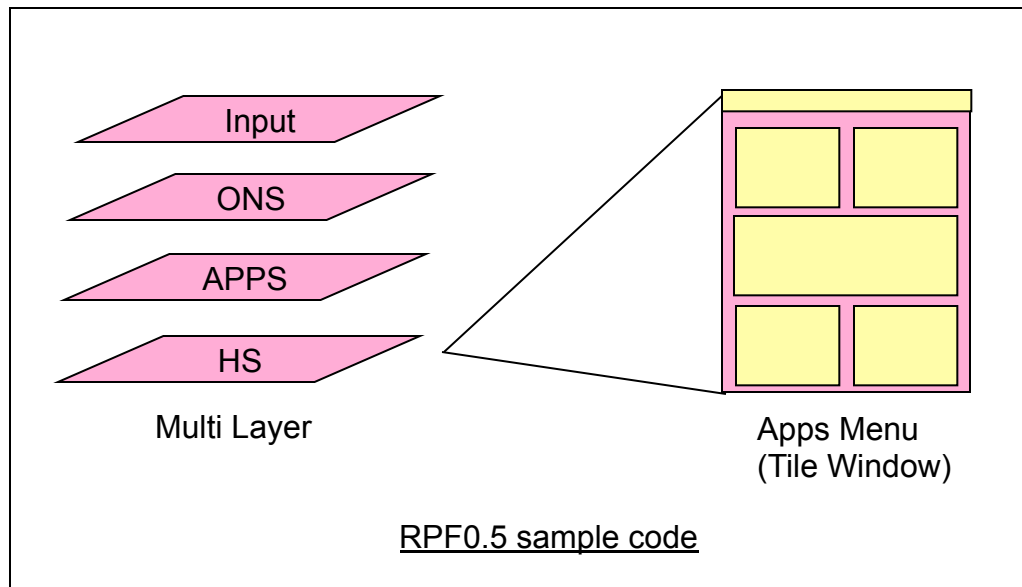


Home Screen & System Controller



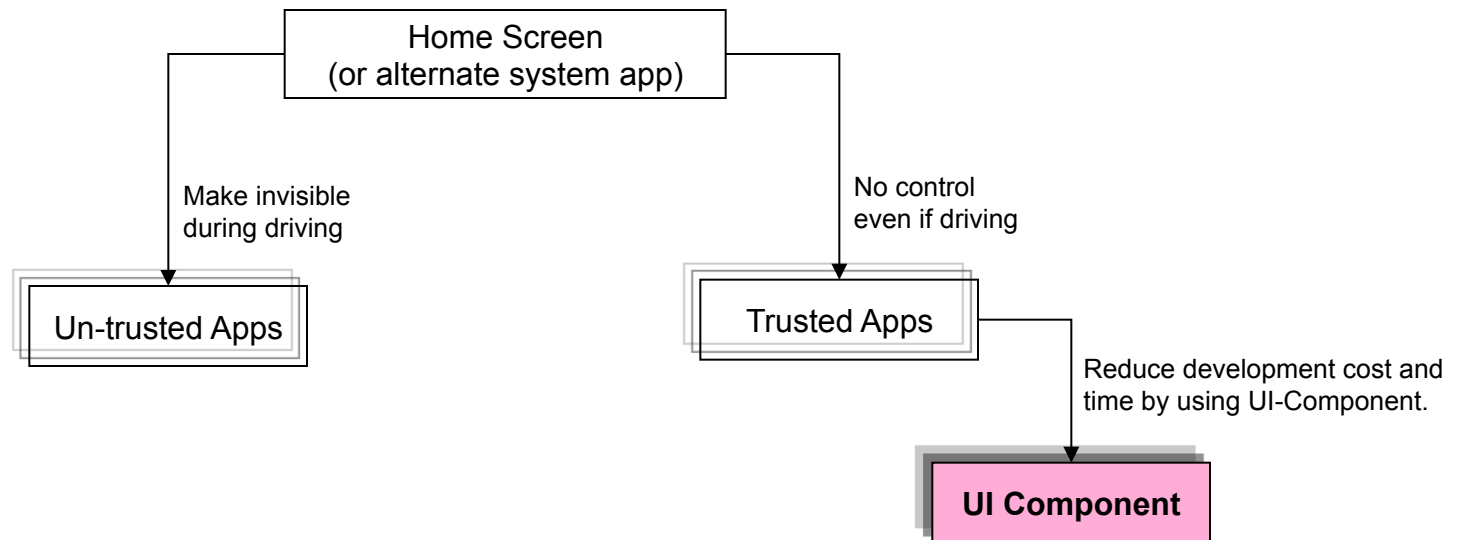
【HS topics】 Sample Screen Layers (RPF0.5)

- Event handling (Input Layer)
- On Screen Message (ONS layer)
- Apps Display (Apps Layer)
- Apps Menu (Home Screen Layer)
status bar, live thumbnail, Apps launcher etc.



【HS topics】 Change View for Driver Distraction

- In IVI system, application must follow the rule and the law.
- Only trusted applications are allowed to display during driving.
 - UI-Component is libraries for trusted application.



UI Manager

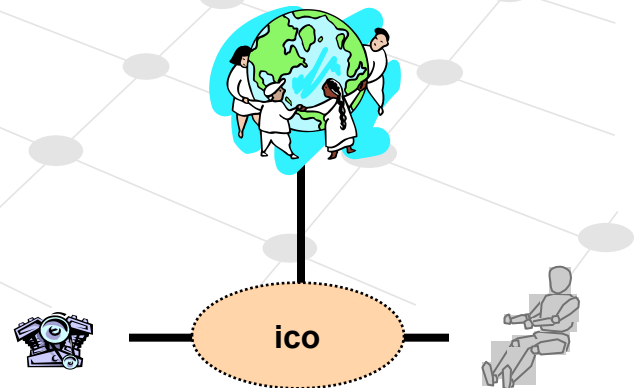
Multi Window Manager

Multi Input Manager

Multi Sound Manager

Vehicle Information Control

Home Screen

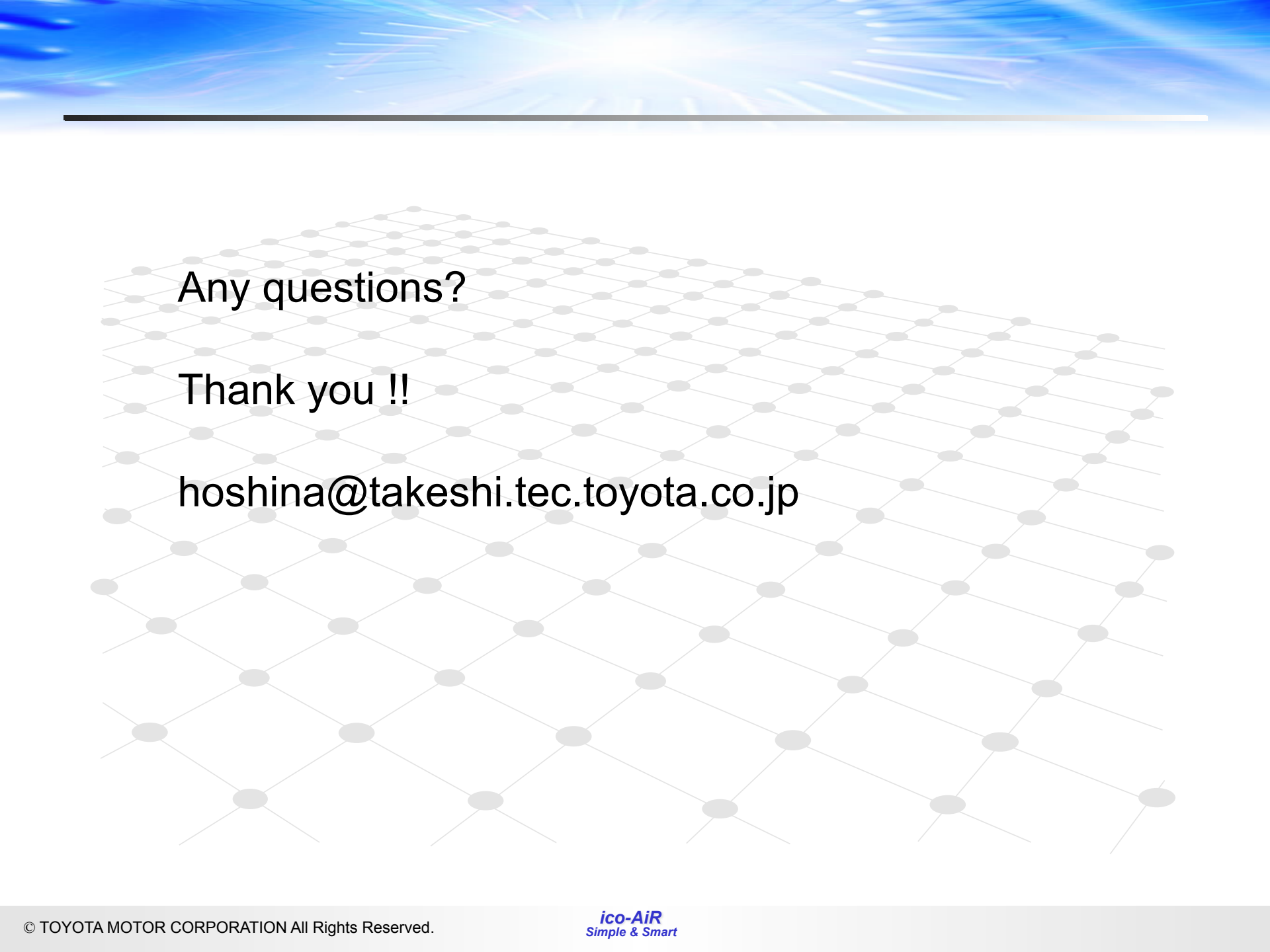


Download

- <http://download.tizen.org/snapshots/2.0alpha/ivi-wayland/latest/images/ivi-wayland/>
- **Packages**
 - **ico-vic-amb-plugin:** Automotive Message Broker (AMB) plugin for delivering vehicle information to applications.
 - **ico-uxf-weston-plugin:** Weston compositor plugin for UI management.
 - **ico-uxf-device-input-controller:** Touch screen and USB steering wheel input device calibration.
 - **ico-uxf-HomeScreen-sample-apps:** a set of sample applications from Toyota including AR navigation, sound and vehicle information controller applications. These apps are examples of how applications behave when the vehicle state changes. i.e. running, stop
 - **ico-vic-carsimulator:** CAN bus emulator
 - **ico-uxf-pulse-plugin:** a policy enforcement plugin to Pulse Audio
 - **ico-uxf-HomeScreen:** sample UI for application launcher and manager.

Summary

- Toyota delivered OSS packages
 - UI-Manager
 - Vehicle Information Control
 - Sample home screen
- Keep up to date on Tizen IVI
- Let's collaborate in OSS community



Any questions?

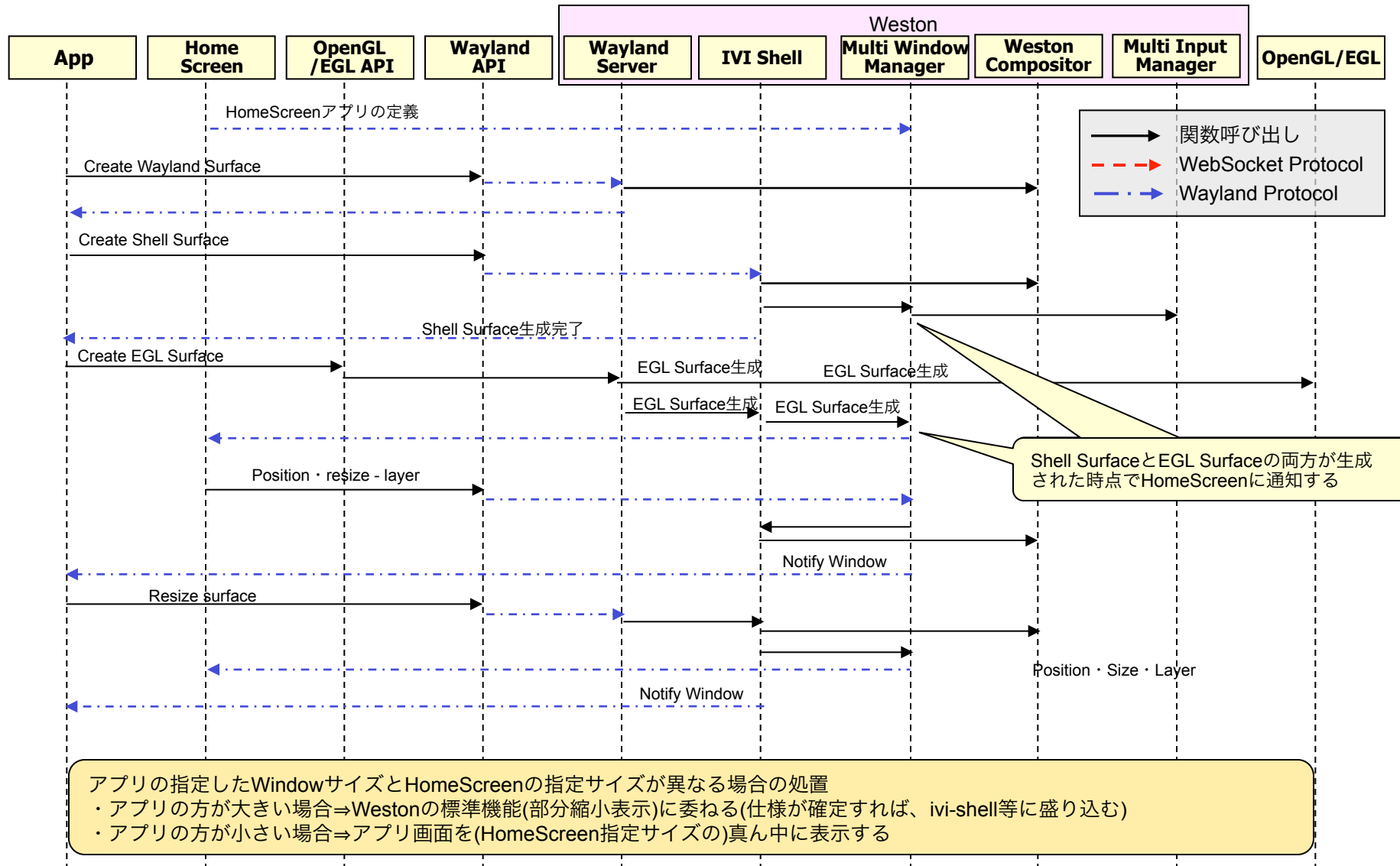
Thank you !!

hoshina@takeshi.tec.toyota.co.jp



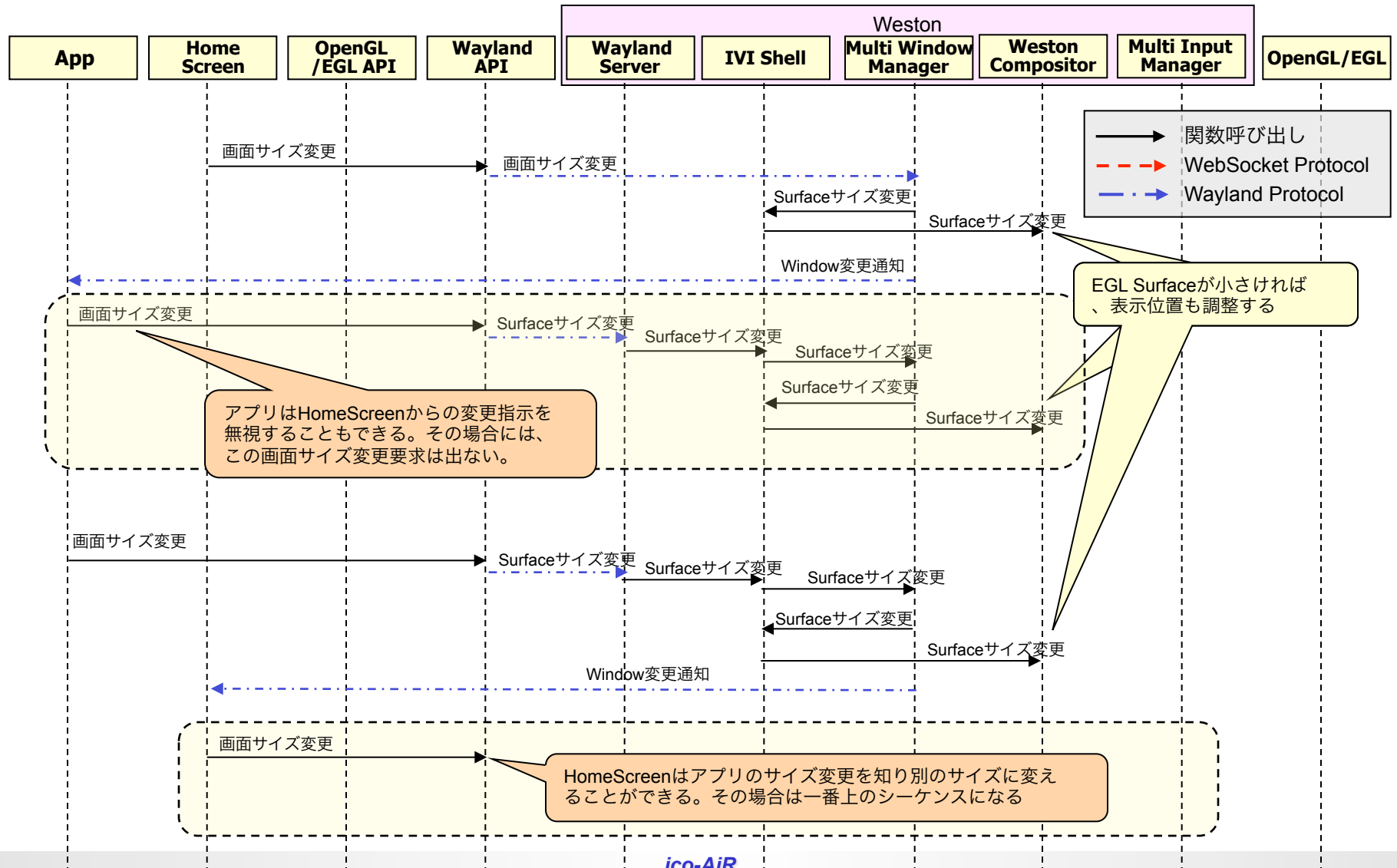
Backup

【MWM TOPICS】 sequence chart

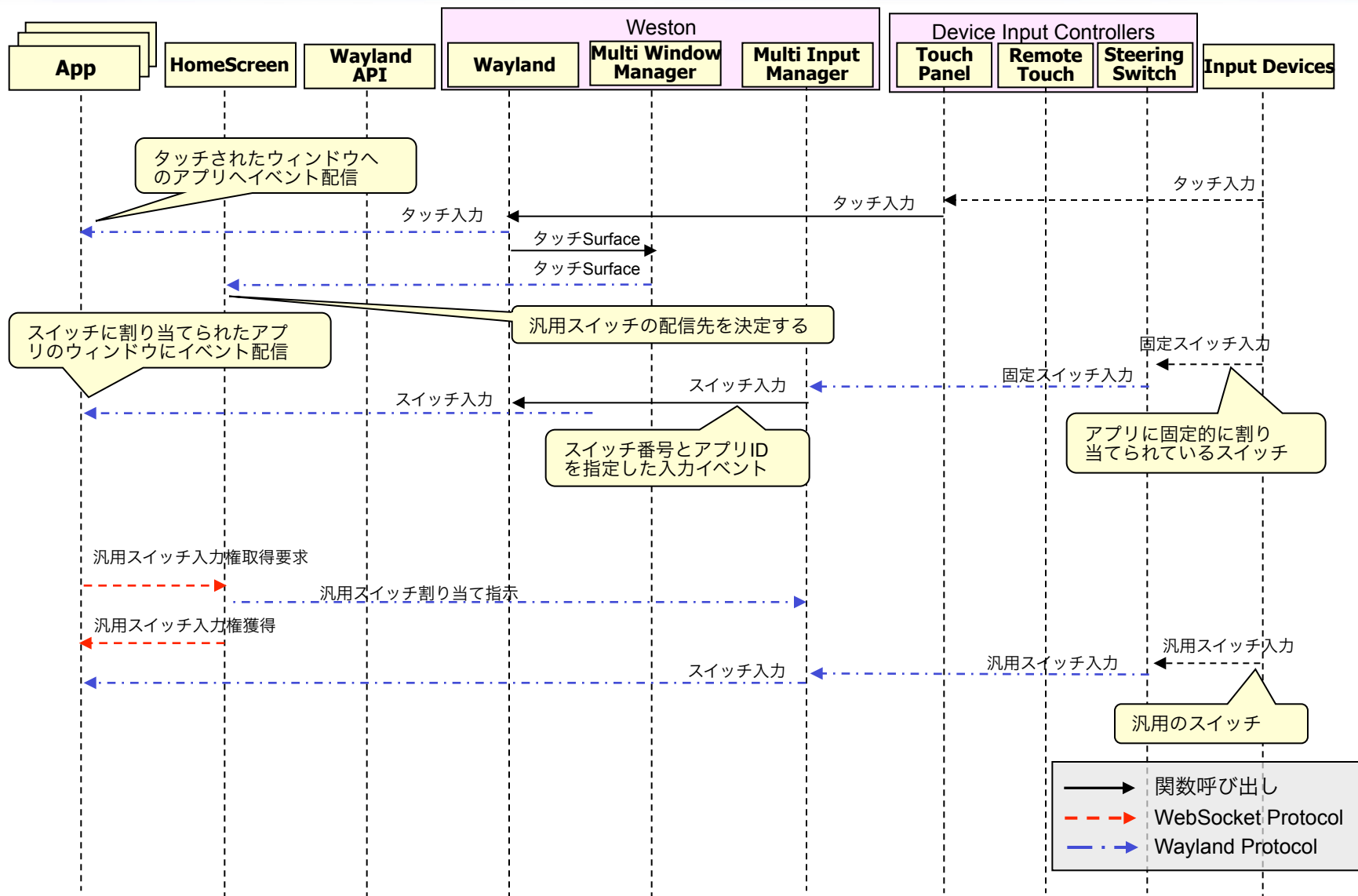


【MWM TOPICS】 sequence chart

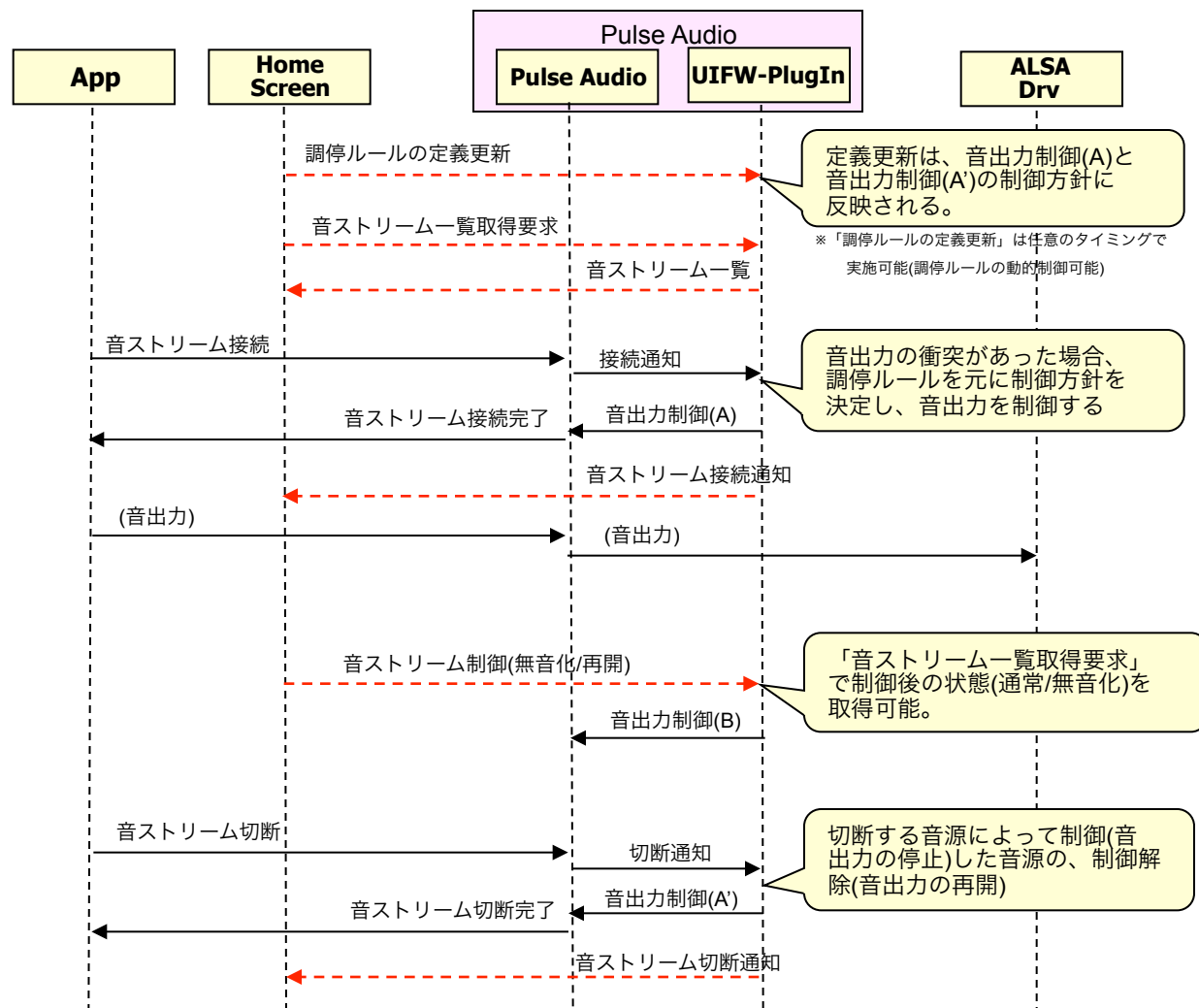
画面サイズ変更シーケンス



MIM sequence chart (RPF0.5)



MSM sequence chart (RPF0.5)



◆音出力制御の種類

A : ミキシング / 減衰ミキシング / ボーイズ / 無音化 / 切断

A' : 切断を除くAの制御解除

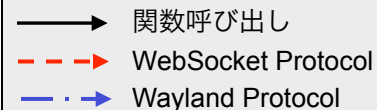
B : 無音化 / 無音化解除

◆音出力制御対象

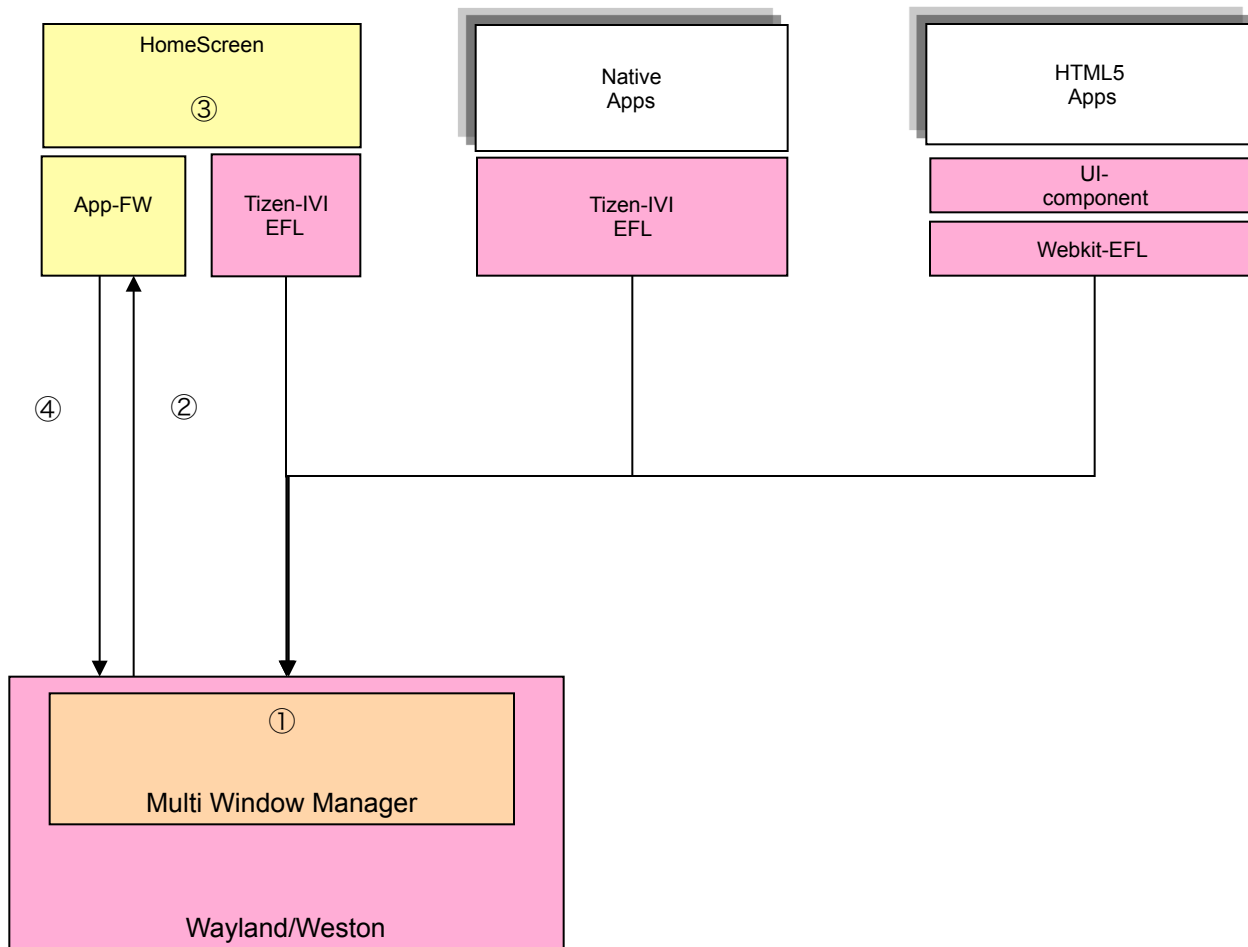
A : 出力デバイスが競合するストリーム

A' : Aで制御したストリーム(切断除く)

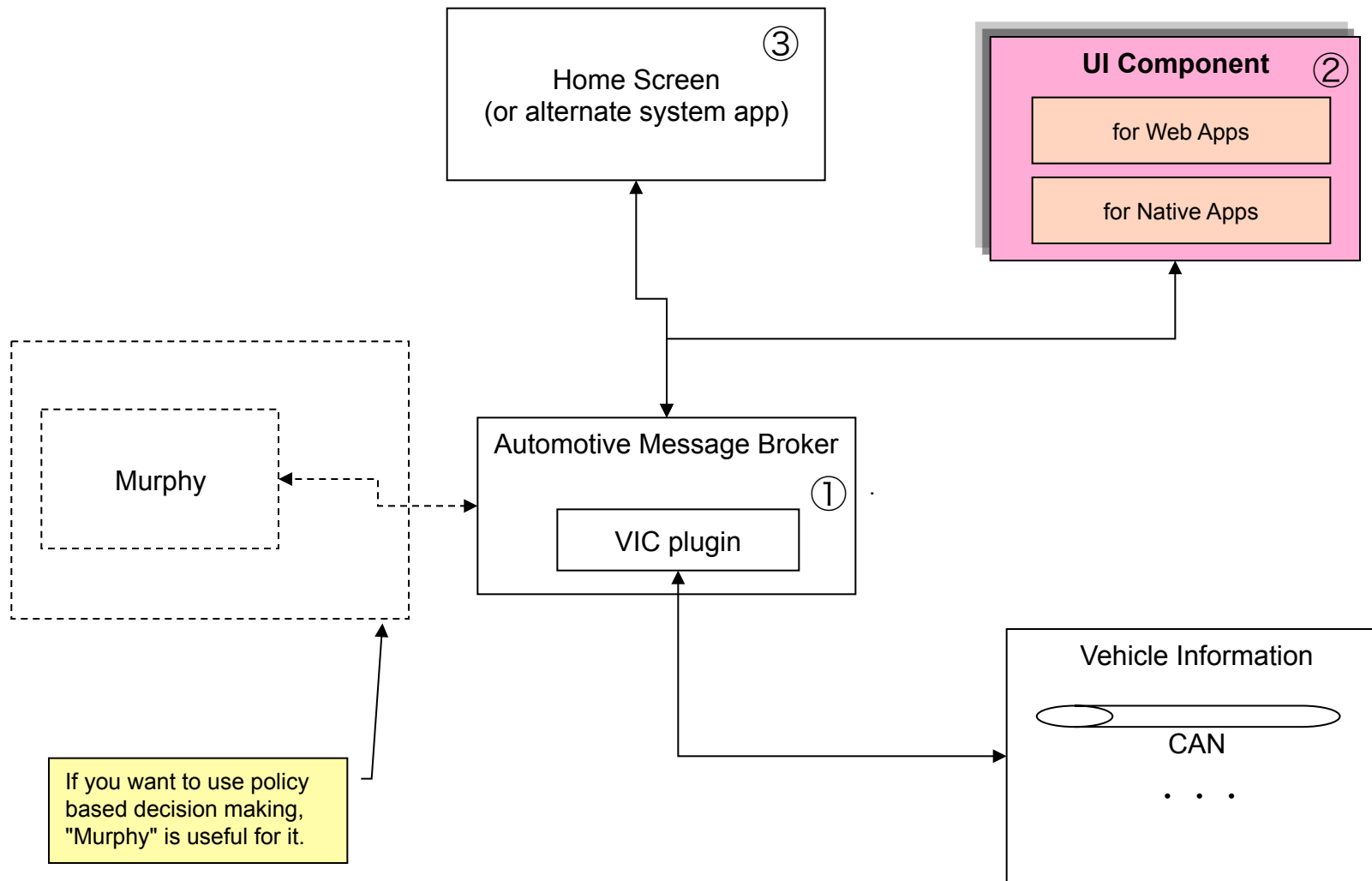
B : 全ストリーム、または指定PIDのストリーム



【MWM TOPICS】 Application monitoring



【HS topics】 Change View for Driver Distraction



【HS topics】 UI Components functional overview

Element type	Driving mode visibility
■ Text labels ■ Radio button ■ Check box ■ Tab panel ■ Audio setting panel	Allowed with restriction below: - (Font size > limit) && (Text string length < limit) - If Text label supports text-scrolling, then scrolling is not allowed. (should stop scrolling)
■ Animation (MNG, GIFanim...)	Allowed with restriction below: - Animation should stop. (only still image is allowed)
■ Slider ■ Accordion list	Allowed with restriction below: - 2 value slider is not allowed, only 1 value. - Accordion should not be animated.
■ Anchor and mode transition ■ Text Input form ■ Dropdown list box ■ Date picker	Not allowed.