AGL HMI Framework
Design Document

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.20</td>
<td>2017/XX/XX</td>
</tr>
</tbody>
</table>
Index

1. HMI Framework overview ................................................................. 4
   1.1. HMI-FW Related components .................................................... 4
       1.1.1. Related components .......................................................... 5
       1.1.2. HMI-FW Components ......................................................... 6
   1.2. Considerations on implementation ............................................. 7
2. HMI-Apps (HMI-FW Related components) ....................................... 8
   2.1. Overview ................................................................................... 8
       2.1.1. Related external components ............................................... 8
       2.1.2. HMI-Apps Life Cycle .......................................................... 9
3. GUI-library ..................................................................................... 10
   3.1. Overview ................................................................................... 10
       3.1.1. Related external components ............................................... 10
       3.1.2. Internal Components .......................................................... 11
   3.2. Graphics functions .................................................................... 12
       3.2.1. Procedure necessary for HMI-Apps ...................................... 12
       3.2.2. Software configuration of GUI-lib ....................................... 13
   3.3. Sound functions .......................................................................... 15
   3.4. Input functions ........................................................................... 16
   3.5. GUI-lib Standard Functions List (Reference material) .................. 17
4. Window Manager ............................................................................... 18
   4.1. Overview ................................................................................... 18
       4.1.1. Related external components ............................................... 18
       4.1.2. Internal Components .......................................................... 19
       4.1.3. Window Resources ............................................................. 20
   4.2. Window Manager Client (API) .................................................... 26
   4.3. Window Resources Manager ....................................................... 27
       4.3.1. Recover Window Resources (Boot Sequence) ....................... 27
       4.3.2. Window Resource Control (Window Manager API) ............... 28
       4.3.3. Register My Application (Window Manager API) .................. 29
       4.3.4. Allocate/Release Window Resources (Window Manager API) .... 30
4.3.5. Notify Window Resources Status (Window Manager API: EVENT) ........................................... 34
4.4. Window Policy Manager ........................................................................................................ 35
  4.4.1. Window Layout Decision .................................................................................................. 36
  4.4.2. Message Signaling Client ................................................................................................. 36
  4.4.3. Window Policy DB Control (Window Manager API) ...................................................... 37
  4.4.4. Window Policy DB use cases ......................................................................................... 37
4.5. Window Layout Manager ........................................................................................................ 40
  4.5.1. Change Window Layout ................................................................................................. 40
  4.5.2. Window Layout DB Control (Window Manager API) .................................................... 41
  4.5.3. Window Layout Pattern Data (DB) sample ..................................................................... 41
5. Home Screen ........................................................................................................................................ 43
  5.1. Overview ................................................................................................................................... 43
    5.1.1. Related external components .......................................................................................... 43
    5.1.2. Internal Components .................................................................................................... 44
  5.2. Home Screen CORE API ........................................................................................................ 45
  5.3. HomeScreen initial processing ............................................................................................... 46
    5.3.1. Initial setting of Window Manager .................................................................................. 46
    5.3.2. Initial setting of Sound Manager .................................................................................. 47
    5.3.3. Initial setting of Input Manager .................................................................................... 47
  5.4. HomeScreen CORE functions ............................................................................................... 48
    5.4.1. Shortcut key ..................................................................................................................... 48
    5.4.2. Status Bar ........................................................................................................................ 48
    5.4.3. Apps Area ....................................................................................................................... 49
    5.4.4. Home Key ........................................................................................................................ 51
    5.4.5. On Screen Area (Floating Area) .................................................................................... 52
  5.5. Apps shipped with HomeScreen ............................................................................................. 53
    5.5.1. Apps launcher ................................................................................................................... 53
    5.5.2. Input Method Editor ........................................................................................................ 54
6. Sound Manager (T.B.D) .................................................................................................................. 55
7. Input Manager (T.B.D) .................................................................................................................... 56
1. HMI Framework overview

1.1. HMI-FW Related components

The related components are shown below.
(Orange box components included in HMI-FW)
1.1.1. Related components

It is not included in HMI-FW, but related components are shown below.

**HMI-Apps**

An application including HMI (drawing, voice, input) processing is called HMI-Apps. HMI-Apps expresses HMI by calling components of HMI-FW.

HMI-Apps has the following responsibilities
- HMI-Apps is used after requesting the HMI resource required for HMI-Manager
- HMI-Apps will do the appropriate processing when the HMI rights are deprived from Manager

**HMI-Services**

It belongs to AGL Service Layer by HMI (drawing, voice, input) control software.

- Graphics Subsystem : Weston/Graphics Device Driver
- Sound Subsystem : Audio Manager/ALS
- Input Subsystem : T.B.D
1.1.2. HMI-FW Components

Components of the HMI-FW are shown below.

GUI-library
You can select the GUI-library (e.g. Qt, HTML5, JavaFX, EB) suitable for HMI with the software necessary for representing HMI.
- 2D/3D Graphics，Image Output
- Sound Output
- Input Event

HMI-Manager
HMI-Manager located between upper GUI-library and lower HMI-Service and has the following components for each HMI.
- Window Manager
- Sound Manager
- Input Manager

Home Screen
Home Screen have an auxiliary screen other than the application screen and interact with the user.
There are various Home screens, but the following representative auxiliary screens are shown below.
- Short Cut Bar
- Status Bar
- Onscreen Bar
1.2. Considerations on implementation

Since HMI-Manager often has different functions depending on OEM and system, it should be separated from HMI-Service. However, if implemented according to this specification, the application calls HMI-Service twice, and performance and sequence issues remain. Therefore, it is also possible to implement the integration of HMI-Manager and HMI-Service modules.
2. HMI-Apps (HMI-FW Related components)

2.1. Overview

2.1.1. Related external components
2.1.2. HMI-Apps Life Cycle

HMI-Apps receives events from each component and performs optimum processing.

Life Cycle

CREATE

VISIBLE

ACTIVE

DRAW

InACTIVE

InVISIBLE

DESTROY

Apps Manager

Window Manager

Window Manager

Window Manager (syncDRAW) GUI-library

Window Manager

Window Manager

Apps Manager
3. GUI-library

3.1. Overview

GUI-library is a library that provides HMI functions to applications, and mainly has HMI functions related to graphics, sound, and input.

3.1.1. Related external components

The application developer selects the GUI-library (e.g. Qt, HTML5, JavaFX, EB) according to the required HMI expression, and issues Upper API depending on each GUI-Library.

(As API functions depend on each GUI-library, refer to each specification.)
3.1.2. Internal Components

HMI-Apps

GUI-Library
CORE

GUI-Library
PF(AGL) adaptation Layer

HMI-Service
3.2. Graphics functions

Graphics provides rendering functions to the application.

3.2.1. Procedure necessary for HMI-Apps

Graphics draws with the following procedure.
1. The application requests Weston to acquire Surface
2. The application makes Area request to Window Manager (OEM options)
3. The application inputs and outputs Graphics data with the Graphics Device Driver.
3.2.2. Software configuration of GUI-lib

GUI-lib has an API specific to AGL besides the standard drawing API. Software vendors providing GUI-lib do not modify GUI-lib CORE, but need to delete functions other than GUI prescribed in AGL. Software vendors need to remodel PAL(*) according to AGL.

(*) PAL = PF Adaptation Layer

```
AGL API                                    Standard API

GUI-lib CORE                               

GUI-lib PAL for Window Manager              GUI-lib PAL for AGL PF

Window Manager                              Weston

OpenGL /EGL

HMI Apps
```
3.3. Sound functions

Sound provides sounding functions to the application with the following procedure.
① The application requests ALSA to acquire Stream.
② The application makes Zone request to Sound Manager (OEM options)
③ The application inputs and outputs Sound data with the Sound Device Driver.
3.4. Input functions

Input provides Input data functions to the application with the following procedure.
① Input-Manager collects input data from each service.
② Input-Manager determines applications to distribute data based on policy (OEM options)
### 3.5. GUI-lib Standard Functions List (Reference material)

<table>
<thead>
<tr>
<th>Fuctions</th>
<th>Qt</th>
<th>JavaFX</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2D</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Window</td>
<td>Qt GUI</td>
<td>○ Stage</td>
<td>POP-UP-WINDOW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Canvas</td>
<td>Painter2D</td>
<td>○ Canvas2D</td>
<td></td>
</tr>
<tr>
<td></td>
<td>WebView</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>3D</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SceneGraph</td>
<td>Material</td>
<td>○ Camera/Light</td>
<td>POP-UP-WINDOW</td>
</tr>
<tr>
<td></td>
<td>Transform</td>
<td>Transform</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Animation</td>
<td>Visual Effect</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Clip-Node</td>
<td>Pick</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Opacity</td>
<td>Sub-Scene</td>
<td></td>
</tr>
<tr>
<td>Graphics</td>
<td>OpenGL/ES</td>
<td>○ 2D Share</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Canvas 3D</td>
<td>3D Share</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(WebGL)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ML</td>
<td>QML</td>
<td>○ FXML</td>
<td></td>
</tr>
<tr>
<td><strong>ETC</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Package</td>
<td>Qt package</td>
<td>△ Java OSGI</td>
<td></td>
</tr>
<tr>
<td>MultiMedia</td>
<td>Audio</td>
<td>○ Audio</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Video</td>
<td>Video</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Camera</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Radio</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input</td>
<td>Mouse</td>
<td>○ Mouse</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gesture</td>
<td>Gesture</td>
<td></td>
</tr>
</tbody>
</table>
4. Window Manager

4.1. Overview

Window Manager determines the optimum screen layout and controls the screen based on the request from the HMI-Apps.

4.1.1. Related external components
4.1.2. Internal Components

<table>
<thead>
<tr>
<th>No</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Window Manager Client API</td>
<td>API</td>
</tr>
<tr>
<td>2</td>
<td>Window Resource Manager</td>
<td>Window Resource Management</td>
</tr>
<tr>
<td>3</td>
<td>Window Policy Manager</td>
<td>Mediation of Window Resources</td>
</tr>
<tr>
<td>4</td>
<td>Window Layout Manager</td>
<td>Window Layout Management</td>
</tr>
</tbody>
</table>

```
WindowManager
Client

Message Signaling
Server

Window
Resource

Window
Policy

Window
Layout

Window
Resources

Message Signaling
Client

Window
Policy

Window
Layout
```
4.1.3. Window Resources

Window Resources are resource information related to the screen managed by the Window Manager and varies depending on the in-vehicle unit configuration (e.g. display). The data items included in Window Resources are shown below.
Display
Display has information on the display device.
A Display can have multiple Layers.

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Information Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ID</td>
<td>Graphics Subsystem</td>
<td>Display ID</td>
</tr>
<tr>
<td>2</td>
<td>Name</td>
<td>-</td>
<td>Display Name</td>
</tr>
<tr>
<td>3</td>
<td>Size</td>
<td>Graphics Subsystem</td>
<td>Display Width and Height</td>
</tr>
</tbody>
</table>

For vehicles with multiple displays it is assumed to have multiple Window Manager.

The Window Manager maintain the link state of the window resources.
Layer

「Layer」 is the information representing the depth of display.

A layer can map multiple areas.
<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Information Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ID</td>
<td>Home Screen</td>
<td>Layer ID</td>
</tr>
<tr>
<td>2</td>
<td>Name</td>
<td>Home Screen</td>
<td>Layer Name</td>
</tr>
<tr>
<td>3</td>
<td>Z order</td>
<td>Home Screen</td>
<td>Layer Zorder</td>
</tr>
<tr>
<td>4</td>
<td>Visibility</td>
<td>Home Screen</td>
<td>Layer Visibility Status</td>
</tr>
<tr>
<td>5</td>
<td>Alpha Blend</td>
<td>Home Screen</td>
<td>Layer Transparent Ratio (α Blend)</td>
</tr>
</tbody>
</table>
Area

The Area is the information of the area where the application draws. A Area maps one Parent Surface. Parent Surface can have multiple child Surfaces.

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Information Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ID</td>
<td>Application</td>
<td>Area ID</td>
</tr>
<tr>
<td>2</td>
<td>Name</td>
<td>Application</td>
<td>Area Name</td>
</tr>
<tr>
<td>3</td>
<td>AppID</td>
<td>Application</td>
<td>Application uniqu ID</td>
</tr>
<tr>
<td>4</td>
<td>Pid</td>
<td>Application</td>
<td>Application Process ID</td>
</tr>
<tr>
<td>5</td>
<td>Parent ID</td>
<td>Application</td>
<td>Parent Surface ID</td>
</tr>
<tr>
<td>6</td>
<td>Child ID</td>
<td>Application</td>
<td>Next Child Surface ID</td>
</tr>
<tr>
<td>7</td>
<td>Position</td>
<td>—</td>
<td>Area Position</td>
</tr>
<tr>
<td>8</td>
<td>Size</td>
<td>—</td>
<td>Area Width and Hight</td>
</tr>
<tr>
<td>9</td>
<td>Z order</td>
<td>—</td>
<td>Area Z order</td>
</tr>
<tr>
<td>10</td>
<td>Visibility</td>
<td>—</td>
<td>Area Visibility Status</td>
</tr>
</tbody>
</table>
Surface
Surface is information of display material frame buffer managed by Graphics Subsystem (Weston).

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Information Source</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ID</td>
<td>Application</td>
<td>Surface ID</td>
</tr>
<tr>
<td>2</td>
<td>SourceSize</td>
<td>Application</td>
<td>FrameBuffer Size</td>
</tr>
<tr>
<td>3</td>
<td>Position</td>
<td>—</td>
<td>Surface position</td>
</tr>
<tr>
<td>4</td>
<td>Size</td>
<td>—</td>
<td>Width and Hight</td>
</tr>
<tr>
<td>5</td>
<td>Z order</td>
<td>—</td>
<td>Surface Z order</td>
</tr>
<tr>
<td>6</td>
<td>Visibility</td>
<td>—</td>
<td>Visibility Status</td>
</tr>
<tr>
<td>7</td>
<td>Alpha Blend</td>
<td>—</td>
<td>Transparent Ratio (α Blend)</td>
</tr>
</tbody>
</table>

How to create Surface ID

|------------------|--------------------------------------|-------------------|

How to create Zorder

|----------------------|---------------------|-----------------------|
4.2. Window Manager Client (API)

The API is shown below.

<table>
<thead>
<tr>
<th>No</th>
<th>Function</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Window Resources Control</td>
<td>R/W</td>
<td>Get/Set Window Resources</td>
</tr>
<tr>
<td>2</td>
<td>Register My Application</td>
<td>W</td>
<td>Registration Own process(PID, SurfaceID)</td>
</tr>
<tr>
<td>3</td>
<td>Allocate Window Resources</td>
<td>W</td>
<td>Request Allocate Area/AreaName</td>
</tr>
<tr>
<td>4</td>
<td>Release Window Resources</td>
<td>W</td>
<td>Request Release Area/AreaName</td>
</tr>
<tr>
<td>5</td>
<td>Notify Window Resources Status</td>
<td>R</td>
<td>Post Window Resources Status to Apps</td>
</tr>
<tr>
<td>6</td>
<td>Window Policy DB Control</td>
<td>R/W</td>
<td>Get/Set Policy DB</td>
</tr>
<tr>
<td>7</td>
<td>Window Layout DB Control</td>
<td>R/W</td>
<td>Get/Set Layout DB</td>
</tr>
</tbody>
</table>
4.3. Window Resources Manager

4.3.1. Recover Window Resources (Boot Sequence)
The Window Manager always holds current window resources. After reboot, Window Manager recovers the Last Window resources.
4.3.2. Window Resource Control (Window Manager API)

HMI Apps can Get/Set Window Resources.

- HomeScreen Get/Set Display and Layer Info.
- HMI-Apps Set/Set Area Info.

Sample: Set Change Layer by Home Screen
4.3.3. Register My Application  (Window Manager API)

When an application uses WindowManager, registration of the application is necessary.

![Diagram of register my application process](image)
4.3.4. Allocate/Release Window Resources (Window Manager API)

When the application starts drawing, it is necessary to acquire Window Resources.

Use Case of Allocate Window Resources

The use cases in which the MediaPlayer displays the song list during Navi map display are shown below.

![Diagram showing use cases of Allocate Window Resources]
The internal sequence of WindowManage during 「AllocateWindowResources」 execution is shown below.

① The app issues 「AllocateWindowResources」 to Window Policy Manager.
② Window Resources Manager responds to App. The App draws its own area.
③ Window Resources Manager responds to another App. The App draws its own area.
④ Window Resources Manager receives 「endDRAW」 from Apps.
⑤ Window Resources Manager issues 「ChangeLayout」 to Window Layout Manager.
Use Case of inVISIBLE Window Resources

Before

Home Screen
ShortCutIcon
Navi Map
HomeButton

MediaPlayer
Song list

After

Media Player
Navi Map
Window Manager
Weston

Home Screen
ShortCutIcon
Navi Map
HomeButton

Request Apps
AllocateWindowResources
SyncDRAW
Layout Decision
DRAW
inVISIBLE
endDRAW
flushDRAW
Swap Buffer

Change Surface
4.3.5. Notify Window Resources Status (Window Manager API:EVENT)

Window Manager notifies the application at the event when the situation of Window Resources changes.

<table>
<thead>
<tr>
<th>No</th>
<th>EVENT</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VISIBLE</td>
<td>R</td>
<td>When own Area becomes Visible</td>
</tr>
<tr>
<td>2</td>
<td>inVISIBLE</td>
<td>R</td>
<td>When own Area becomes InVisible</td>
</tr>
<tr>
<td>3</td>
<td>ACTIVE</td>
<td>R</td>
<td>When own Area becomes Active</td>
</tr>
<tr>
<td>4</td>
<td>inACTIVE</td>
<td>R</td>
<td>When own Area becomes InActive</td>
</tr>
<tr>
<td>5</td>
<td>syncDRAW</td>
<td>R</td>
<td>The app must draw own Area. After the end, response 「endDRAW」.</td>
</tr>
<tr>
<td>6</td>
<td>flushDRAW</td>
<td>R</td>
<td>The app must flush own Area.</td>
</tr>
</tbody>
</table>
4.4. Window Policy Manager

When there is a screen request from the application due to a user operation or a state change of the system, it is common to erase the old screen and display a new screen. But, setting an optimum screen layout in consideration of the following conditions is an important requirement of an in-vehicle HMI.

- Application Priority
- Driving restrictions

This requirement is called "HMI Policy". However, HMI Policy is often different for each OEM and each in-vehicle device. So, Window Policy Manager have policy logic based on PolicyDB prepared in advance.
4.4.1. Window Layout Decision

According to a request from "Window Resource Manager", Window Policy Manager decides Layout based on Window Policy DB and responds to Window Resource Manager.

4.4.2. Message Signaling Client

Policy Manager acquires latest vehicle information from Message Signaling.
4.4.3. Window Policy DB Control (Window Manager API)

Update the Window Policy DB with the following timing.

✓ Hardware
   in-vehicle unit setting
✓ Software
   Software update, Application delivery

4.4.4. Window Policy DB use cases

Precondition

✓ Window Layer Pattern (have 2 Layers)
   ① Base Layer (HomeScreen)
   ② Apps Layer

---

Control Bar

<table>
<thead>
<tr>
<th>Apps Area</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base Layer</td>
</tr>
<tr>
<td>Apps Layer</td>
</tr>
</tbody>
</table>

---

HomeButton
 ✓ Apps Layout Pattern (have 2 types)
   ① FULL Full Apps Area
   ② HALF MAIN: Upper Apps Area
       SUB: Lower Apps Area

<table>
<thead>
<tr>
<th>ControlBar</th>
<th>ControlBar</th>
</tr>
</thead>
<tbody>
<tr>
<td>① FULL</td>
<td>② HALF</td>
</tr>
<tr>
<td></td>
<td>MAIN</td>
</tr>
<tr>
<td></td>
<td>SUB</td>
</tr>
<tr>
<td>HomeButton</td>
<td>HomeButton</td>
</tr>
</tbody>
</table>

 ✓ Displayable area of application (3 Apps)
   ① HOMESCREEN: FULL
   ② NAVI: FULL or MAIN
   ③ BASE (General Apps): FULL or MAIN or SUB
**Policy DB (State Machine at stopping)**

The state transition table during STOP is shown below.
In the case of driving start, save the current state and shift to the RUN state.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>SUB</th>
<th>HOME</th>
<th>NAVI</th>
<th>BASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hs</td>
<td>HOME</td>
<td>HOME</td>
<td>–</td>
<td>To n1</td>
<td>To b1</td>
</tr>
<tr>
<td>n1</td>
<td>NAVI</td>
<td>NAVI</td>
<td>To Hs</td>
<td>–</td>
<td>To n2</td>
</tr>
<tr>
<td>n2</td>
<td>NAVI</td>
<td>*</td>
<td>To Hs</td>
<td>To n1</td>
<td>MAIN:NAVI SUB:BASE</td>
</tr>
<tr>
<td>b1</td>
<td>BASE</td>
<td>BASE</td>
<td>To Hs</td>
<td>To n1</td>
<td>–</td>
</tr>
<tr>
<td>b2</td>
<td>BASE</td>
<td>*</td>
<td>To Hs</td>
<td>To n1</td>
<td>To b1</td>
</tr>
</tbody>
</table>

First Low : State Name
Second Low、Third low : Area Name
(When MAIN and SUB are the same, it is regarded as FULL screen)
The action after the 4th column is for the application request（State Machine Table）
* : Other applications different from MAIN

**Policy DB (State Machine at running)**

The state transition table during RUN is shown below.
In the case of stopping, restore the current state and shift to the STOP state.

<table>
<thead>
<tr>
<th></th>
<th>MAIN</th>
<th>SUB</th>
<th>HOME</th>
<th>NAVI</th>
<th>MM</th>
<th>BASE</th>
</tr>
</thead>
<tbody>
<tr>
<td>n1</td>
<td>NAVI</td>
<td>NAVI</td>
<td>–</td>
<td>–</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>
4.5. Window Layout Manager

The Window Layout Manager has the following functions related to Layout.

4.5.1. Change Window Layout

If Window Layout Manager receive 「Change Window Layout」
They need update Window Resources and send 「Change Surface」 to Weston.
4.5.2. Window Layout DB Control (Window Manager API)

Update the Window Layout DB with the following timing.
- Hardware
  in-vehicle unit setting
- Software
  Software update, Application deliver

4.5.3. Window Layout Pattern Data (DB) sample

The layout Pattern Data is shown below together with data description (JSON).

Precondition

Layout Pattern (2 patterns)
1. HomeScreen Basic
2. Apps Half Basic

```
Control Bar

Apps A

Apps B

Apps Half Basic
```

```
HomeScreen Basic
```

```
HomeButton
```
Pattern Description (JSON)

Display_height=1920
Display_width =1024

ControleBar_height =200
HomeController_height=200

Apps_height=(Display_height- ControleBar_height- HomeController_height)/2
Main_y = ControleBar_height
Sub_y= Main_y+ Apps_height
HomeController_y= Sub_y+ HomeController_height

① Home Screen Basic (Home Screen Layer)

“Layout”
{
   "name":"HomeScreenBasic"
   "areas":
   [  
      "name":"ControleBar", "x":0, "y":0, "width":"display_width",  
       "height": "display_height", "zorder":0  
   ],
   [  
      "name":"HomeController", "x":0,"y":HomeController_y,"width":"HomeController_width",  
       "height": "HomeController_height", "zorder":0  
   ]
}

② Apps Half Basic (Apps Layer)

“Layout”
{
   "name":"AppsHalfBasic"
   "areas":
   [  
      "name":"Main", "x":0, "y": Main_y,  
       "width":"Display_width", "height": "Apps_height", "zorder":0  
   ],
   [  
      "name":"Sub", "x":0", "y":Sub_y,  
       "width":"Display_width", "height": "Apps_height", "zorder":0  
   ]
}
5. Home Screen

5.1. Overview

Home Screen is a component for performing user operation. It is possible to have different Home Screen for each in-vehicle device.

5.1.1. Related external components
## 5.1.2. Internal Components

<table>
<thead>
<tr>
<th>No</th>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Home Screen Client</td>
<td>API</td>
</tr>
<tr>
<td>2</td>
<td>Home Screen Core</td>
<td>Home Screen CORE functions</td>
</tr>
<tr>
<td>3</td>
<td>Apps Launcher</td>
<td>HMI-Apps</td>
</tr>
<tr>
<td>4</td>
<td>Input Method Editor</td>
<td>HMI-Apps</td>
</tr>
</tbody>
</table>
5.2. Home Screen CORE API

<table>
<thead>
<tr>
<th>No</th>
<th>Function</th>
<th>R/W</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Home Screen Resources Control</td>
<td>R/W</td>
<td>Get/Set Resources</td>
</tr>
<tr>
<td>2</td>
<td>Short Cut key</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Status Bar</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Home Key</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>On Screen</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Notify</td>
<td>R</td>
<td></td>
</tr>
</tbody>
</table>
5.3. **HomeScreen initial processing**

The HomeScreen performs the following processing at startup. For details, see the chapter of each manager.

5.3.1. **Initial setting of 「Window Manager」**

**Setting Layer**

HomeScreen needs to set up multiple layers including applications at startup. An example of setting of multiple layers is shown below.

An application may monopolize one layer or share it.
Setting Area
HomeScreen needs to set multiple areas for 「OnScreen」 and 「HomeScreen」 at startup.

5.3.2. Initial setting of 「Sound Manager」 (T.B.D)
5.3.3. Initial setting of 「Input Manager」 (T.B.D)
5.4. HomeScreen CORE functions

The standard Home Screen sample is shown below. (Sample)

5.4.1. Shortcut key
The user selects an application to use with apps menu.

5.4.2. Status Bar
The Status Bar shows status information by notification command from each application.
5.4.3. Apps Area
Apps Area is the area for the application to display.

Share Apps Layer
Each application needs to acquire screen rights to Window Manager. Applications that lose screen rights are set to hide the area.

Depending on the screen size, multiple applications can share the Apps Layer.
**Stack Display**

When an executing application calls another application, another application occupies the screen.
5.4.4. Home Key
HomeScreen hides the current application and returns to the home screen window. In the following example, the displayed application (app A) is not displayed and HomeScreen is displayed.
5.4.5. On Screen Area (Floating Area)
On Screen displays on the screen when notification from each application is received.

On Screen Sample
5.5. Apps shipped with HomeScreen

5.5.1. Apps launcher

The user can select necessary applications from the application menu. The HomeScreen informs the application that it is selected. If the selected application is not activated, the HomeScreen requests the application management to start the application.
5.5.2. Input Method Editor

The application can call IME with user operation.
6. Sound Manager (T.B.D)
7. Input Manager (T.B.D)