

## **Waltham transmitter upstream**

From AGL to Wayland community

# Agenda

---

- **Motivation**
- **Requirements**
- **Current solutions**
  - ◆ Remoting plugin
  - ◆ Waltham
- **Ideas**
- **Discussion points**

# Motivation

## Future cockpit system

- Multiple displays
- Different hardware, different companies
- Seamless integration of content
- Content not fixed to one display



Advanced Driver  
Information Technology

A joint venture company of

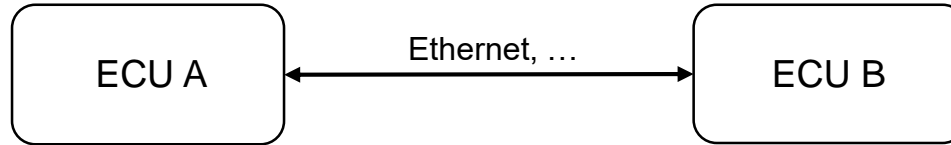
Robert Bosch GmbH / Robert Bosch Car Multimedia GmbH and DENSO Corporation

# Motivation

## Realization method of sharing

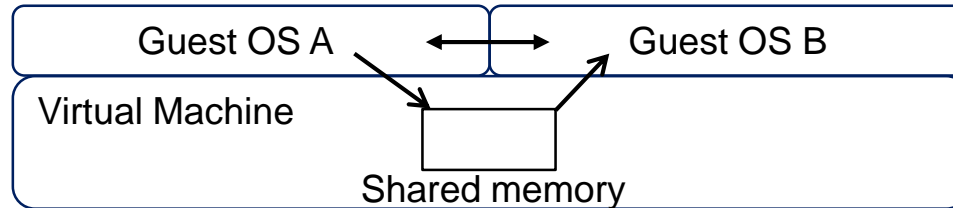
### ■ Sharing between multi ECUs

- ◆ Data are shared over network



### ■ Sharing between guest OSs on virtualization environment

- ◆ Data are shared on shared memory or over virtual network



**Both graphic sharing mechanism and protocol are needed for sharing**

# High level Requirements

- **A buffer shall be shared per application surface.  
(per display is not flexible enough.)**
- **A buffer shall be shared between multi ECUs.**
- **A buffer shall be shared between multi OSs on hypervisor.**
- **A buffer shall be shared for multiple receivers.**
- **Input events shall be handled at the same time of buffer sharing.**
- **Connection recovering mechanism shall be enabled.**
- **It shall be 60 fps.**
- **It shall work on OS other than Linux. (e.g. Android)**

- **Virtual display sharing**

- ◆ Remoting plugin

- **Surface sharing**

- ◆ Waltham

- **For hypervisor**

- ◆ GPU sharing, Display sharing
  - SoC Specific

# Current solutions

## Remoting plugin

- **The remoting plugin can only send per virtual display. It cannot send per surface.**
- **The remoting plugin just send virtual output and don't handle input.**

### ■ ADIT has two implementations

#### ◆ For AGL

- Waltham is just used for starting gst streaming.
- A buffer is shared on RTP.
- Only touch event is tested. (keyboard and pointer events are not tested.)
- Weston v2.0 based.

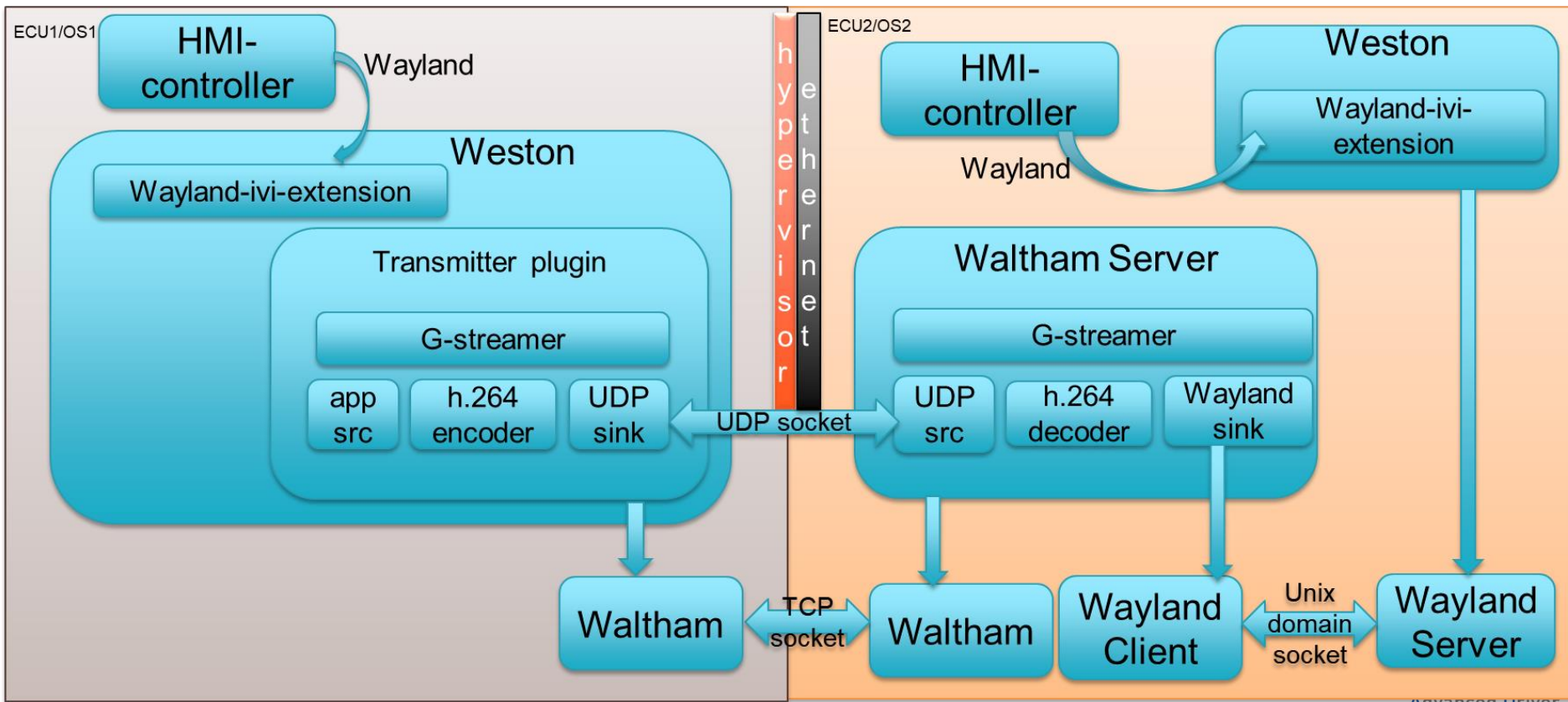
#### ◆ For customer project

- Waltham is just used for touch forwarding on hypervisor.
- A buffer is NOT shared. Gst streaming is also NOT used.
- Buffers from both OSs are scanned out by drm directly. (GPU sharing and Display sharing are used.)
- Weston v2.0 based.



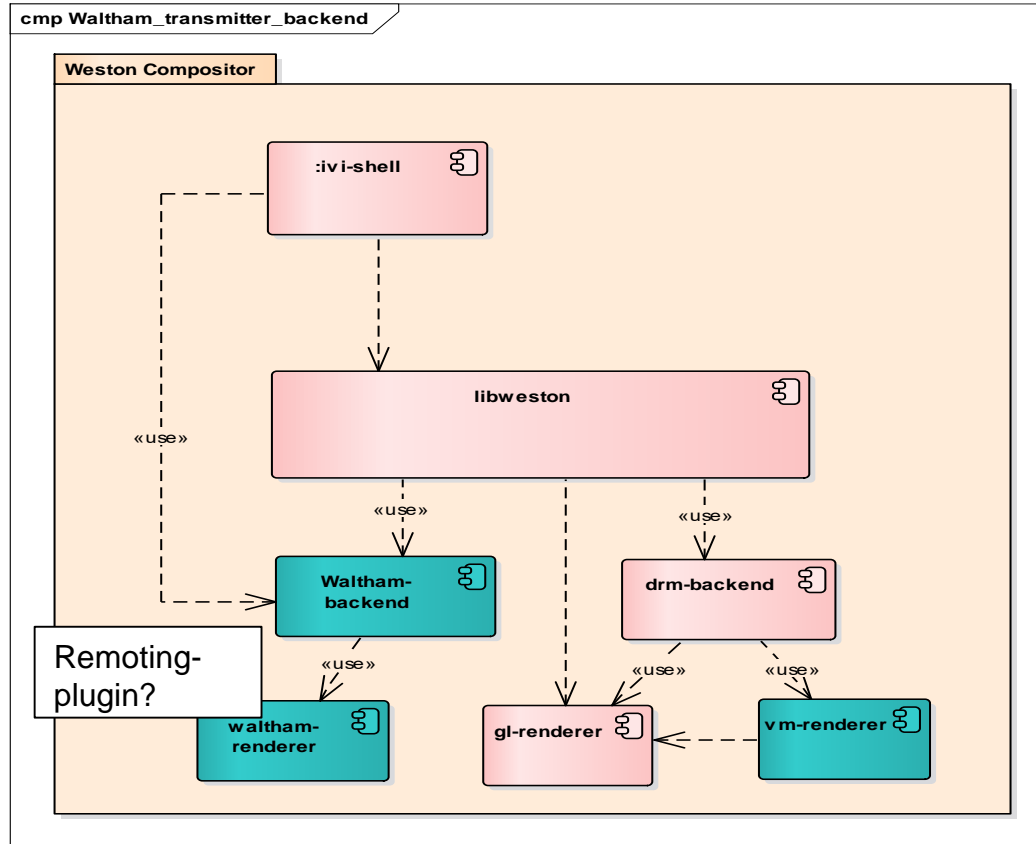
# Current solutions

## Waltham in practice at ADIT



# Ideas

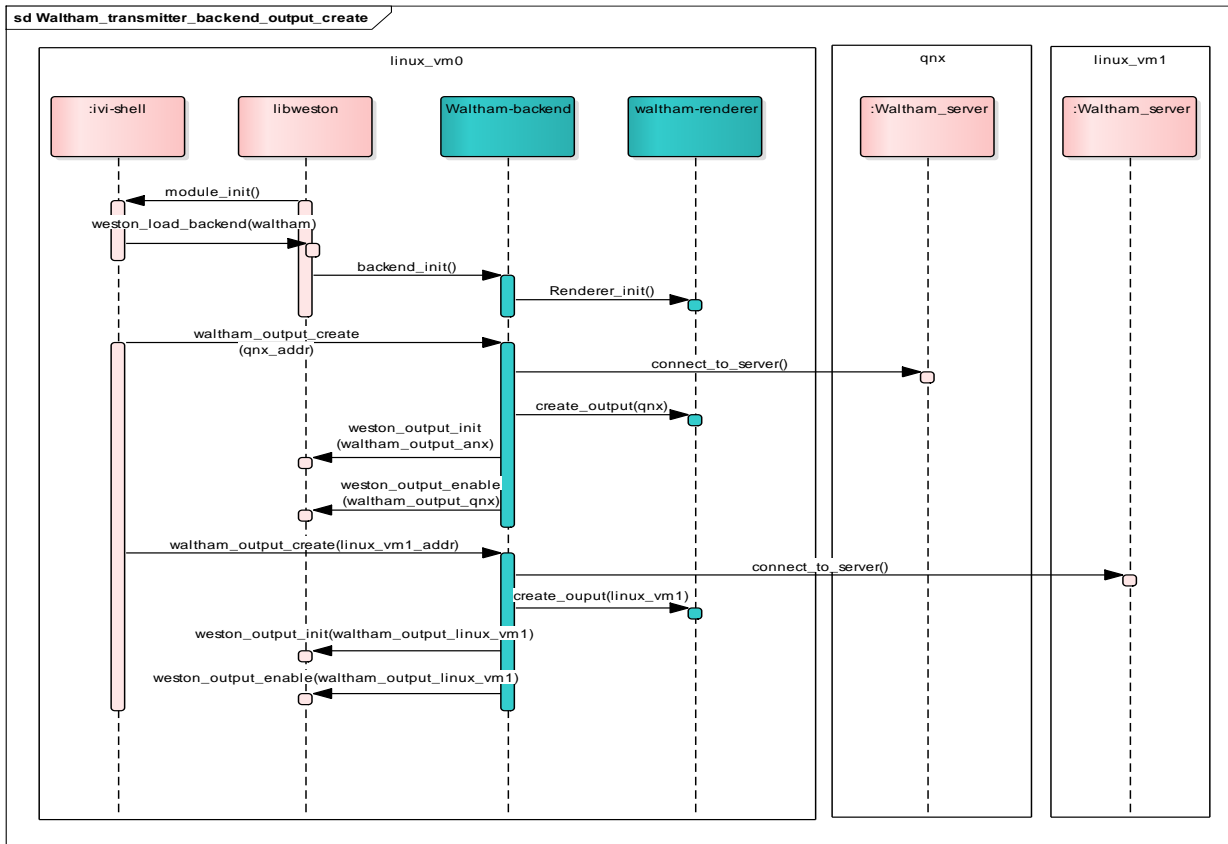
## Waltham transmitter as Weston backend



- **Libweston should implement support for multiple backends and multiple renderers**
- **A Waltham backend similar to Weston backend can be implemented. Differences lie in terms of handling frame callback and creation of multiple connections.**
- **Waltham backend should implement input handling. It is also the right place to handle input as per Weston architecture.**
- **Waltham renderer implements the rendering part. There is not rendering as such but it takes care of sending buffers to Waltham server.**
- **Multiple server connections can be realized by creating multiple outputs in Waltham backend.**
- **Surfaces can be associated to outputs by means of `ilm_screenSetRenderOrder` interface.**

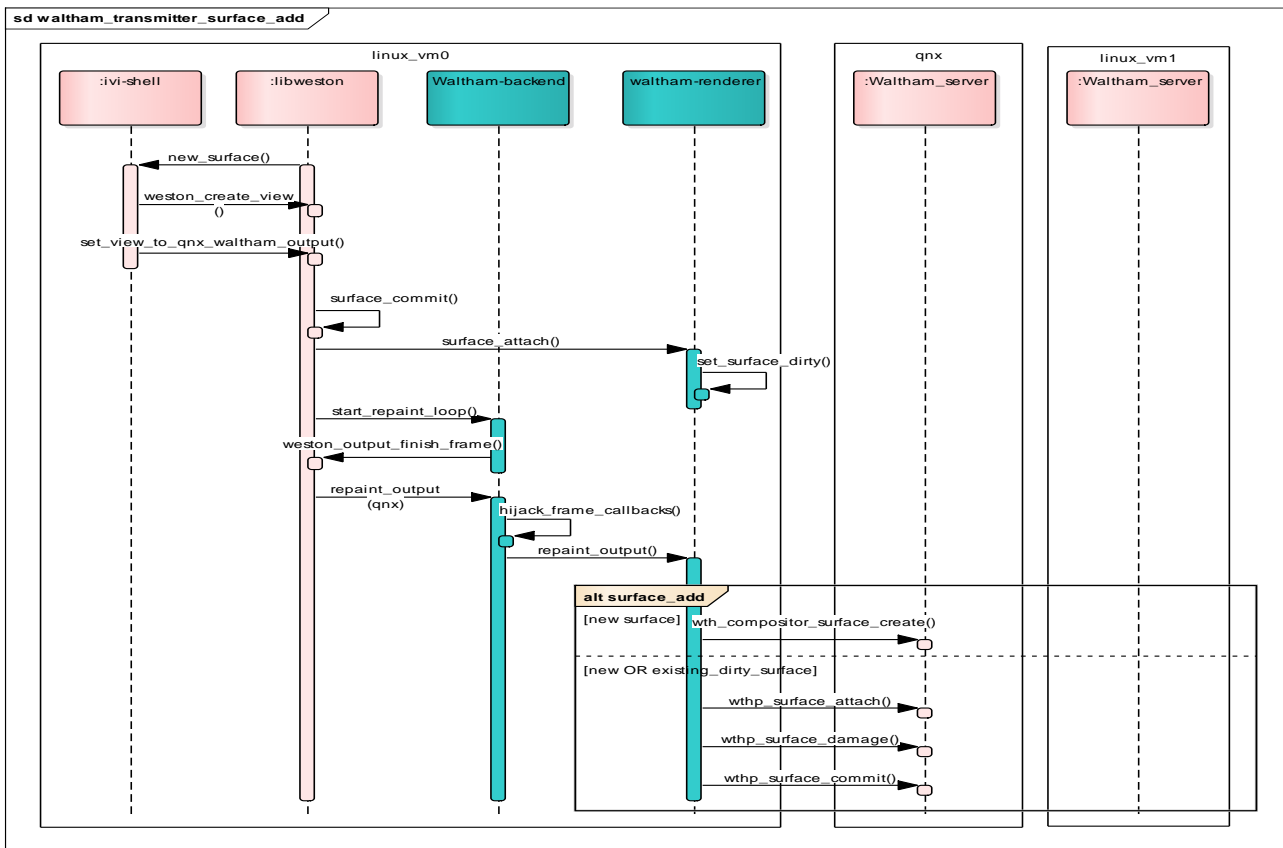
# Ideas

## Waltham transmitter as Weston backend



# Ideas

## Waltham transmitter as Weston backend



- Provides provision for input handling as per Weston architecture.
- Waltham output is treated as any other output. This means there are no additional changes in ivi-shell apart from creation of Waltham output.

- **Requires changes to core Weston (libweston) to handle multiple backends and multiple renderers.**
- **Libweston obtains properties of a surface(width, height, buffer) from renderer. So libweston is coupled to use a single renderer. Present solution requires a change to take “surface properties” part out of renderer. This means changes and tests to multiple renderers and backends are needed. This could be quite too much.**

## ■ What is AGL's strategy?

- ◆ Use Weston for the future as well? Or create own compositor?
- ◆ Does Waltham still fit to the strategy?

## ■ How to harmonize with remoting plugin?

## ■ How to abstract buffer sharing technologies?

- ◆ Gstreamer
- ◆ Raw buffer sharing via Waltham
- ◆ SoC specific technology like Hyper DMA-BUF
- ◆ ...



## ■ AGL

- ◆ All Member Meeting Winter 2017
  - <https://wiki.automotivelinux.org/eg-ui-graphics>
    - ◆ [20170209 ui and graphics eg waltham.pdf](https://wiki.automotivelinux.org/eg-ui-graphics)
- ◆ All Member Meeting Europe 2017
  - <https://sched.co/C9rQ>

## ■ Genivi

- ◆ GENIVI Technical Summit
  - <https://at.projects.genivi.org/wiki/display/WIK4/GENIVI+Technical+Summit+Session+Content+2018>
    - ◆ ADIT/Bosch Implementing Waltham in practice
    - ◆ Waltham in Practice - working session (Harsha Manjula Mallikarjun)
- ◆ GENIVI 18th All Member Meeting
  - <https://at.projects.genivi.org/wiki/display/WIK4/18th+GENIVI+AMM+Presentations>
    - ◆ Domain Interaction - Wayland-IVI-extension / Waltham Usage in Shared Graphics Environment

# Appendix

### ■ <https://at.projects.genivi.org/wiki/display/DIRO/RAMES>

#### Solution ideas:

#### 4. Scene-based distribution



```
glBindFramebuffer(...)  
glClearColor(...)  
glClear(...)  
...  
glUseProgram(...)  
glBindBuffer(...)  
glActiveTexture(...)  
glUniform1f(x)  
glDrawElements(...)
```



ECU 1

ECU 2

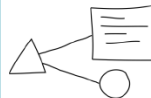
- ✦ Low network bandwidth needed especially after initial transfer
- ✦ No compression artifacts
- ✦ Easier scaling to higher resolutions
- ✦ No GPU needed on sending side
- ✦ Graphical interaction possible between scenes from different ECUs
- ✦ Application has to provide content with special API



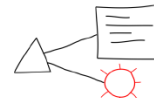
#### Update of frames: Scene-based distribution

Frame A

Frame B



changeValue -> y



Scene + state

```
glBindFramebuffer(...)  
glClearColor(...)  
glClear(...)  
...  
glUseProgram(...)  
glBindBuffer(...)  
glActiveTexture(...)  
glUniform1f(x)  
glDrawElements(...)
```

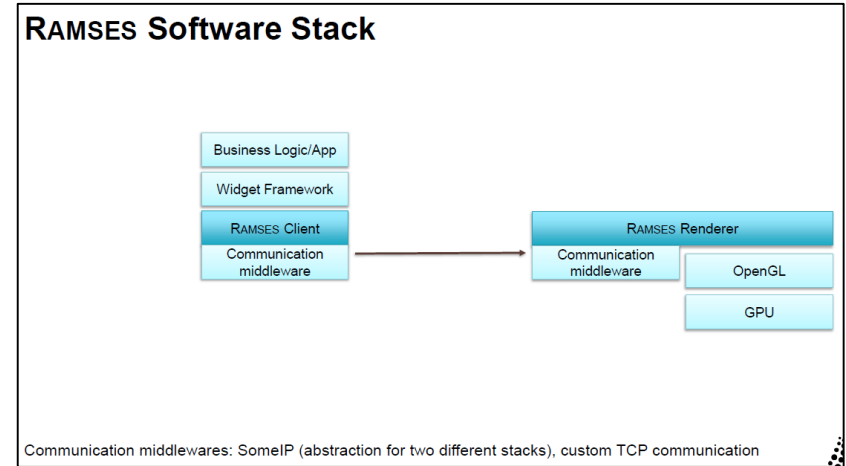
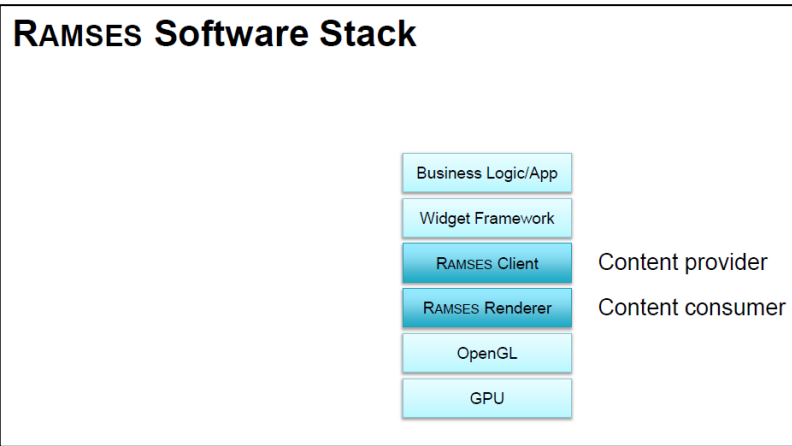


```
glBindFramebuffer(...)  
glClearColor(...)  
glClear(...)  
...  
glUseProgram(...)  
glBindBuffer(...)  
glActiveTexture(...)  
glUniform1f(y)  
glDrawElements(...)
```



# Current solutions

## API remoting: RAMSES



- **RAMSES requires quite big modifications in the existing components.**
- **RAMSES became OSS but it is not general as of now.**