

Wayland IVI Extension

May 10, 2017 | Updates in front of us

Eugen Friedrich, Emre Ucan

Graphics engineers, ADIT

Wayland IVI Extension

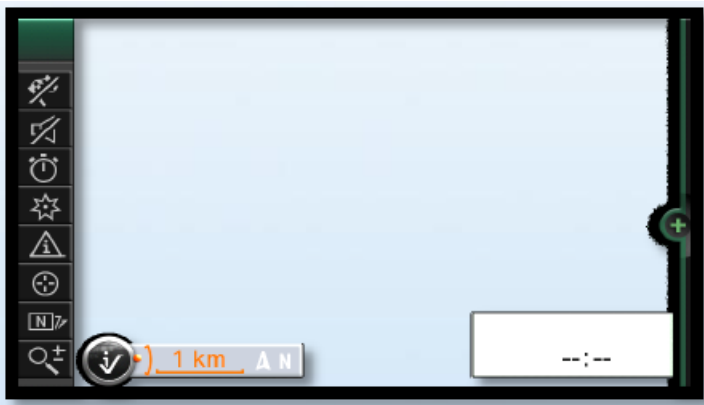


Agenda

- Short introduction of wayland-ivi-extension
- Current status of wayland-ivi-extension repository
- ivi-controller protocol update
- New challenges
 - Xdg-shell support
 - Using libweston
 - Virtualization
 - Resource management: not really new but still not realized

Introduction of wayland-ivi-extension

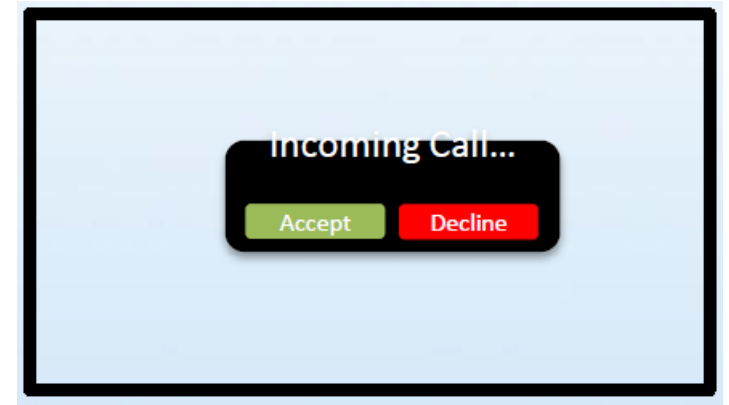
- Graphical content is produced from different applications
- Compositor is in charge to combine it to a single view



HMI



Navigation



Popup

Introduction of wayland-ivi-extension

- Wayland-ivi-extension provides an interface to control the composition and input routing
- Control is possible from an separate application which is not a compositor
 - allows flexible implementation of window managers
 - Allows using the same compositor across projects



Current status of wayland-ivi-extension repository

- Github project
 - 24 closed, 3 open pull request in last year
 - 6 contributors
- Most important updates
 - deprecated APS's
 - Complete ilmClient API: use ivi-application protocol direct
 - ilm_surfaceGetPixelFormat: doesn't make sense for ilm surface pixel format is provided by the attached buffer
 - Adaptation for weston 2.0
 - Renesas GPU stack has a problem, buffers are inverted
 - Workaround reverts commit 319397e050e2b4833e10093ccef8ad77a6ef78d in weston
 - Correct fix should be done in the GPU stack

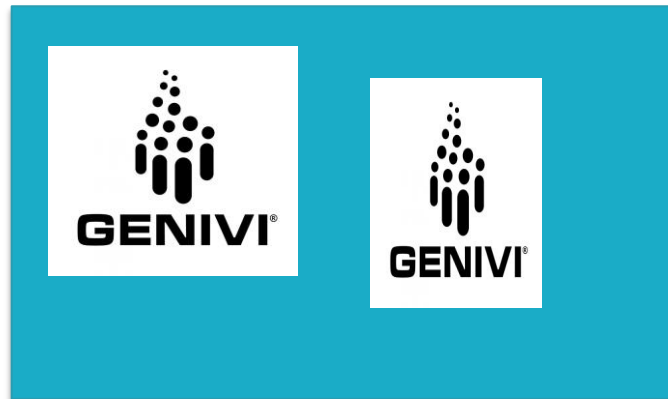
ivi-controller protocol update

- Motivation for the update
 - Reduce round trips between ivi-controller and compositor
 - Simplify implementation of some ilm API's
 - Remove unused parts of the protocol
 - Simplify wayland object lifecycle
- Better error handling
- Introduce surface types
 - Desktop compatible surfaces
 - Restricted surfaces

ivi-controller protocol update (cont.)

- New features
 - Visualize one surface on two displays
 - Visualize one surface on one display in two different views
 - Display hotplugging
 - Predefined display IDs

Display 1



Display 2



Discussion on the new protocol update

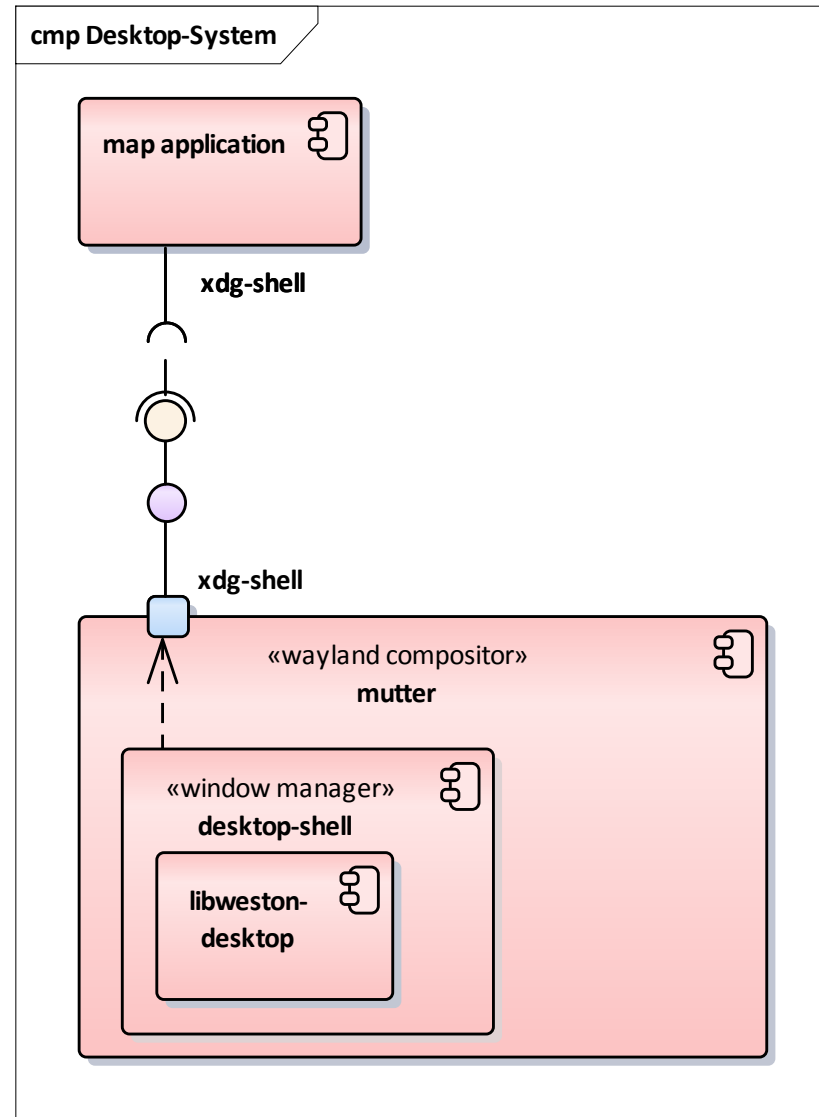
New challenges: xdg shell support

- xdg-shell is supported by freedesktop.org
- Freedesktop is building a base platform for desktop software
- Backend for application visible APIs as such:



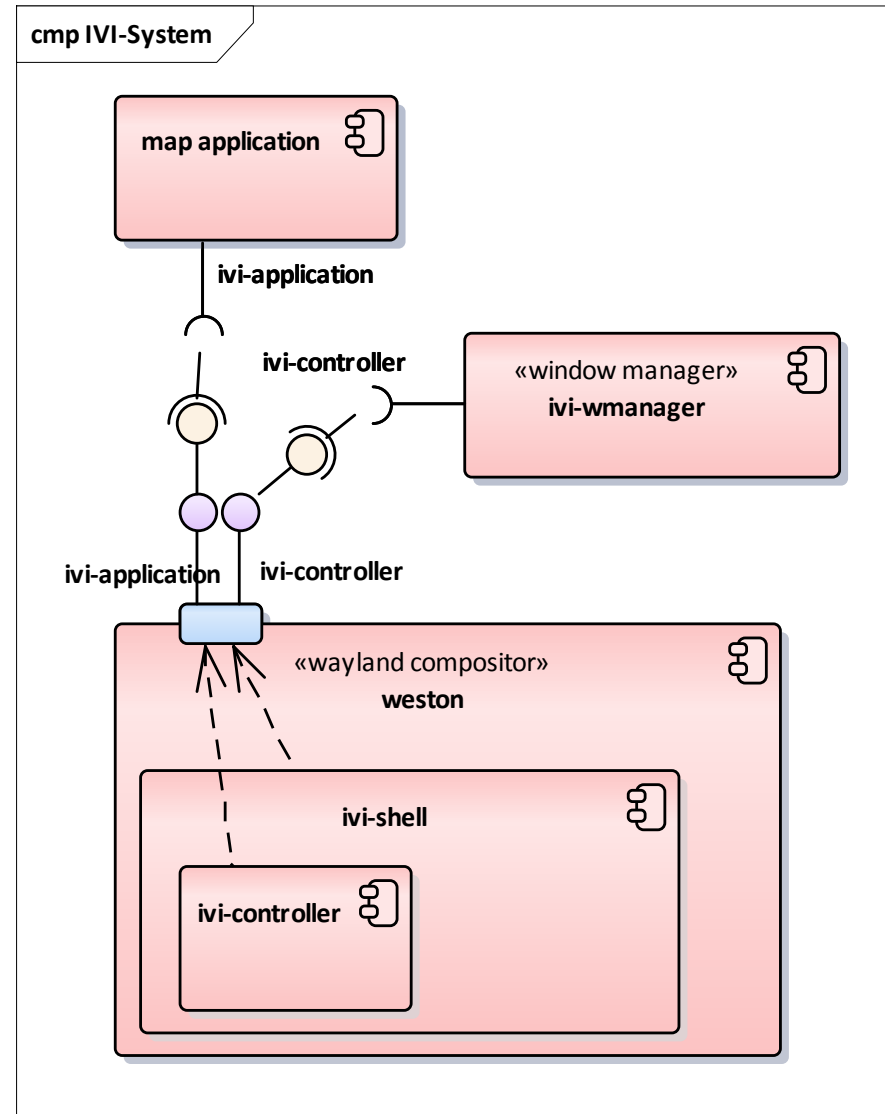
New challenges: xdg shell support

Desktop setup



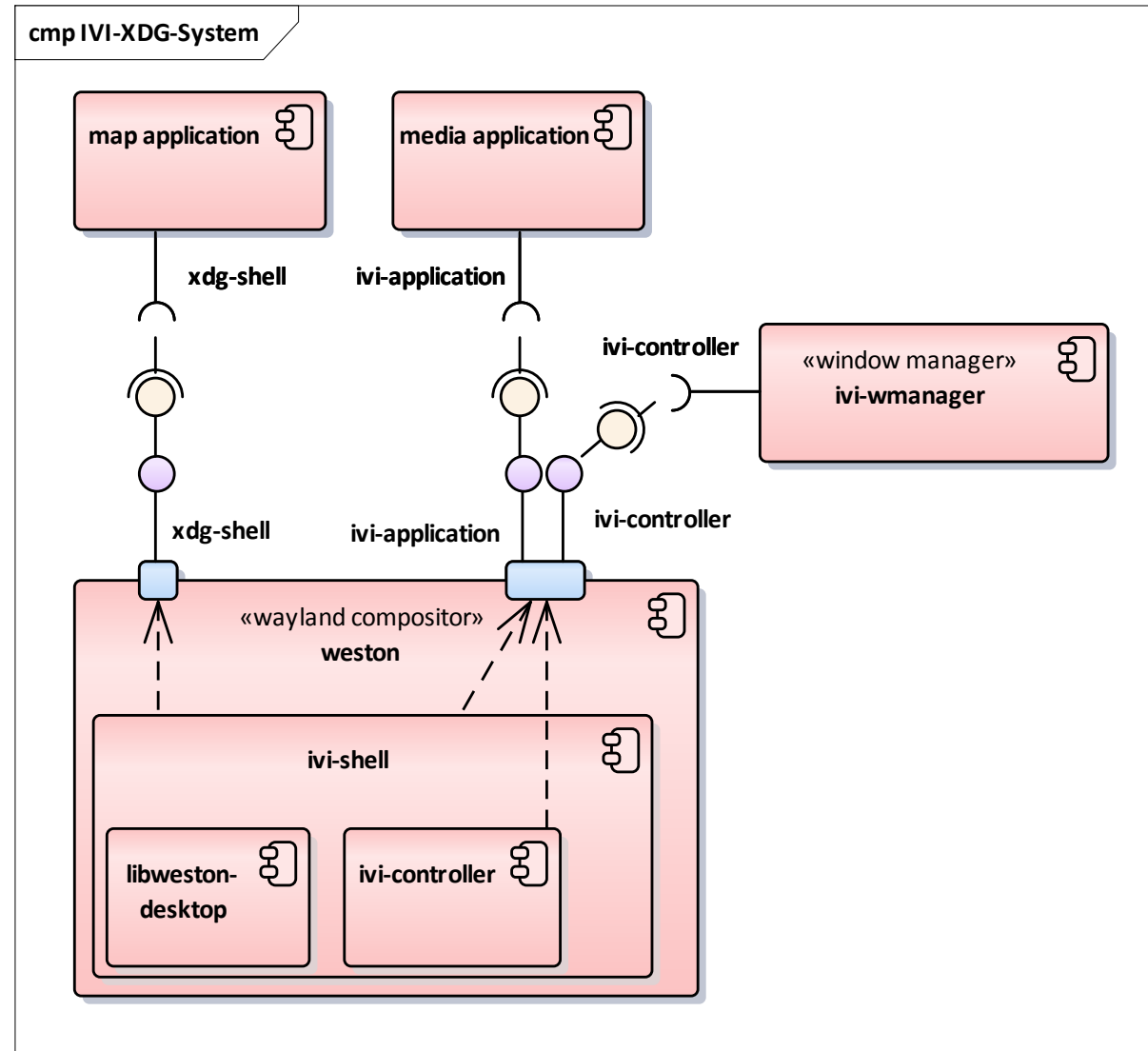
New challenges: xdg shell support

IVI setup



New challenges: xdg shell support

IVI with xdg setup



New challenges: xdg shell support

- ivi-application protocol does not offer much benefits:
 - It is only used to assign surface IDs for application surfaces
 - Every application has to be modified to include unique Ids
 - IDs are not portable across projects
- xdg-shell offers:
 - Easier integration of desktop applications in ivi system
 - Support open source frameworks out of the box
 - Existing desktop applications could be use as it is

New challenges: xdg shell support

- xdg shell is designed for desktop environment
 - Some of the defined events and request are not applicable in ivi systems
- We still need to have surface IDs to be able to control surfaces from a HMI controller.
 - Application will use xdg shell and will provide “name”
 - ivi-shell can generate surface IDs
 - ivi-shell can read IDs from a database
- Startup critical applications need special handling:
 - default minimal scene
 - Defined id and behavior for early apps

Discussion on the xdg shell support

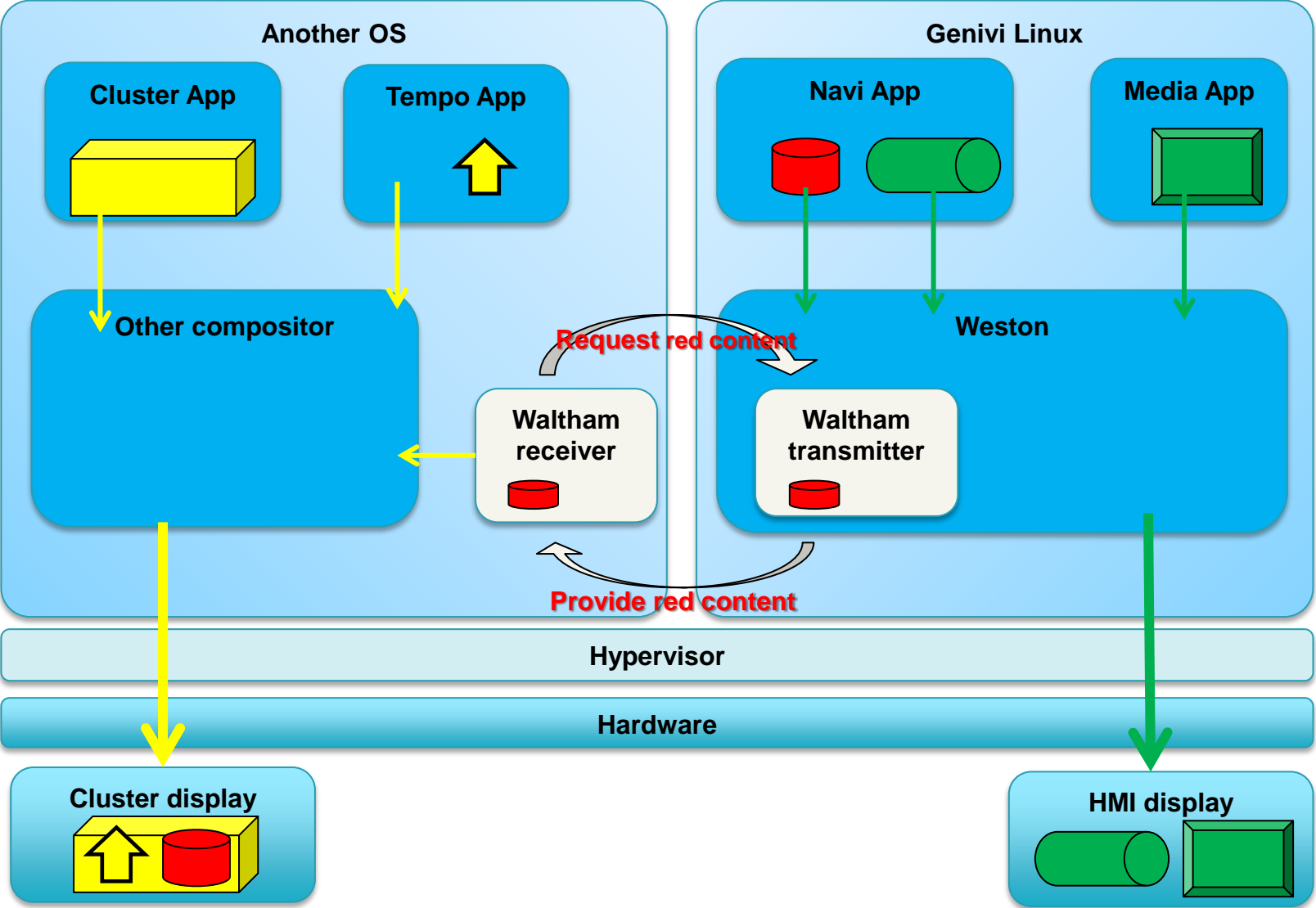
New challenges: Using libweston

- We can use libweston instead of weston compositor
 - We can create our own compositor with greater flexibility
 - We can still reuse upstream core weston code
- Different versions of libweston could be installed on parallel
- Easier to maintain and update wayland-ivi-extension
 - We can move ivi-shell code to wayland-ivi-extension repository
 - We can stick with one version of wayland/weston for longer time

New challenges: Virtualization

- Complex systems
 - Complexity affects a lot of hardware and software layers in the system

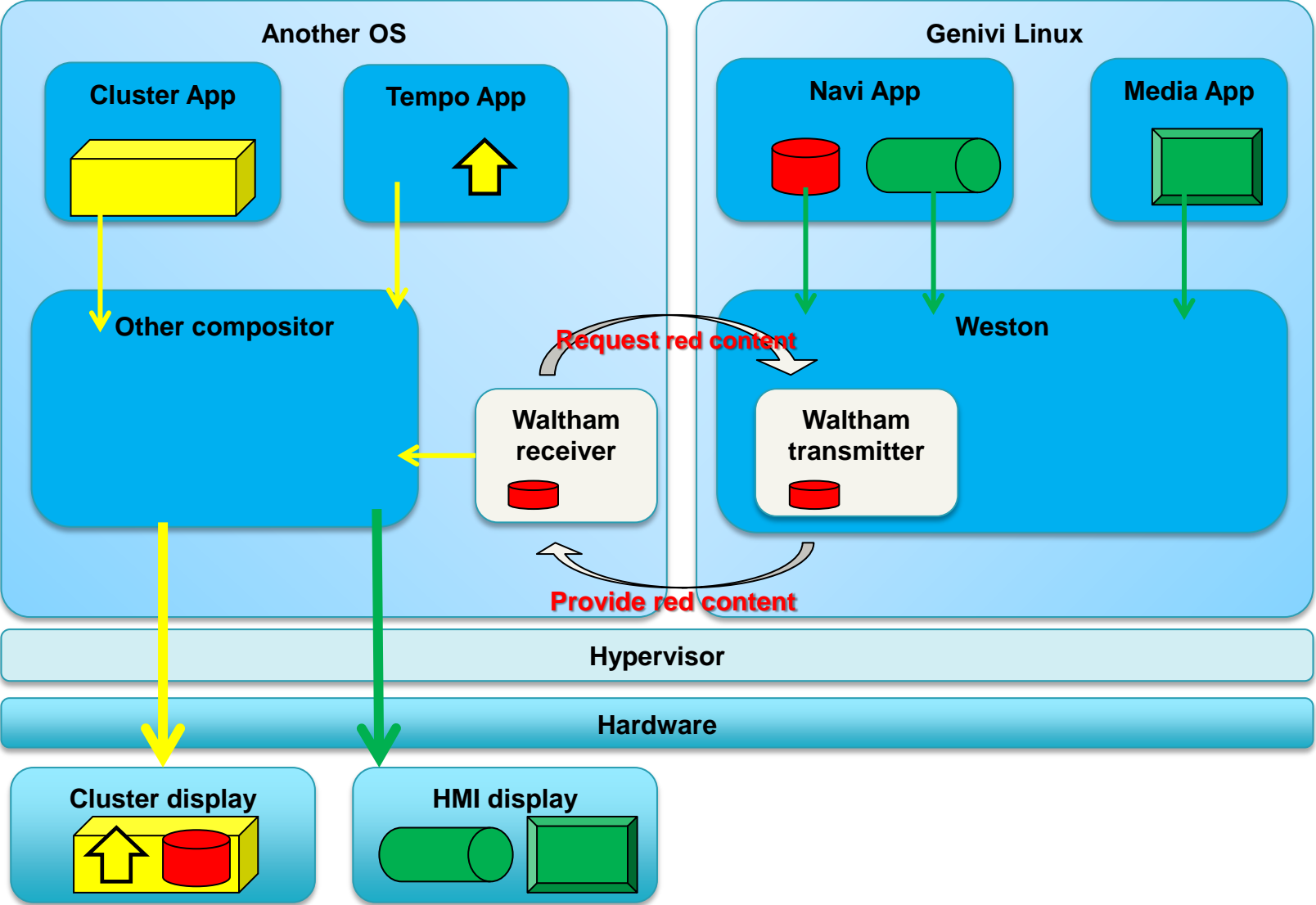
New challenges: Virtualization



- Each OS has its own display
- One or several application content is shared from one OS to another
- Bidirectional sharing of content could be required



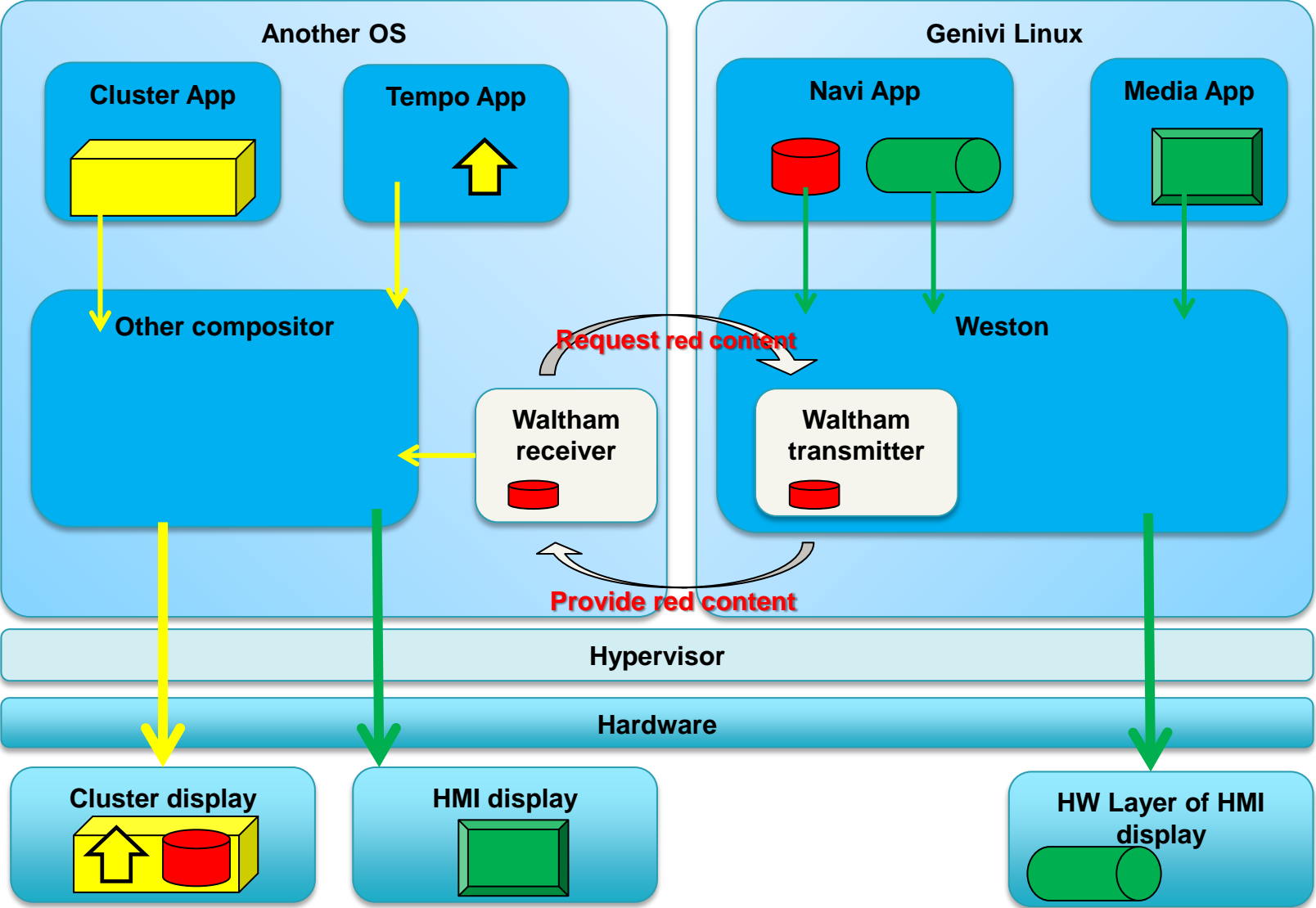
New challenges: Virtualization



- All displays are connected to one OS
- All application content is shared from one OS to another, here from Genivi Linux -> Another OS



New challenges: Virtualization



- All display are connected to one OS
- But Genivi Linux has controll of a hardware plane on HMI display
- Each OS is able to display content on the display directly
- Handling of application content is complex



New challenges: Virtualization

- How virtualization affects compositor?
 - We have to send and received content to a different OS and used it in the composition
- How virtualization affects ilm interface?
 - HMI controller could need a way to find out if the surface is a remote or local surface
 - This information is also required to forward the input event correctly

Discussion on virtualization

New challenges: Resource management

- Are there enough resources in the system to display or render particular use-case?
- Do we need clear separation of resource and policy management?
- Support for sending and receiving content to a from different OS in virtualization environment

Thank you!

We definitely expect some questions!!!

Visit us <https://at.projects.genivi.org/wiki/display/WIE/Wayland+IVI+Extension+Home>

Contact us: genivi-ivi-layer-management@lists.genivi.org

This work is licensed under a Creative Commons Attribution-Share Alike 4.0 (CC BY-SA 4.0)
GENIVI is a registered trademark of the GENIVI Alliance in the USA and other countries.
Copyright © GENIVI Alliance 2017.

