



Technology Consulting Company
Research, Development &
Global Standard

Porting Tizen-IVI 3.0 to an ARM based SoC Platform

Damian Hobson-Garcia

Automotive Linux Summit
July 1-2, 2014
Tokyo, Japan

Tizen IVI support

Until recently...

■ Intel architecture (x86) system

- Tizen IVI 2.0alpha, Tizen IVI 3.0

■ ARM architecture based system

- Tizen IVI 2.0alpha (ivi-panda)


Need to port Tizen IVI 3.0 to ARM ourselves

Current State of Affairs

■ Intel architecture (x86) system

- Tizen IVI 2.0alpha, Tizen IVI 3.0
- Tizen Common 

■ ARM architecture based system

- Tizen IVI 2.0alpha (ivi-panda)
- Tizen Common 

Tizen IVI now based on Tizen Common

- Lots of reuse

Target Platform

- **Renesas R-Car Gen2 series platform**
- **R-Car M2**
 - ARM Cortex A15 x2
- **R-Car H2**
 - ARM Cortex A15 x4, + ARM Cortex A7 x4 (option)
- **3D Graphics System**
 - Imagination Technologies PowerVR series
- **On board IP blocks**
 - H/W video decode/encode
 - image processing

Agenda

- **Objective**
- **Methodology**
- **Porting Tasks**
 - Weston/Wayland Integration
 - WebKit Integration
 - GStreamer Integration

■ Tizen IVI 3.0 on R-Car M2/H2

1. Standard Native Applications

- Terminal program
- Open GL/ES applications

2. Web

- Browser and web applications

3. Multimedia

- Video playback (1080p @ 30fps)

■ Tizen IVI 3.0 milestone releases we used:

- M2-Sep (released Oct 11, 2013)
- M2-EOY (released Jan 15, 2014)
- M2-March2014 (released April 11, 2014)

■ Non-hardware dependant packages

- Rebuild for ARM instruction set

■ Hardware dependant packages

- Replace/update with R-Car M2/H2 support

Tizen Common/IVI Rebase Methodology

- **Reuse Tizen Common ARM support for Tizen IVI 3.0**
 - Most Tizen IVI packages now based on Tizen Common
- **Non-hardware dependant packages**
 - Use prebuilt packages
- **Hardware dependant packages**
 - Replace/update with R-Car M2/H2 support

Workflow and Source Code Download

Full local build



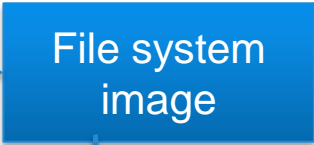
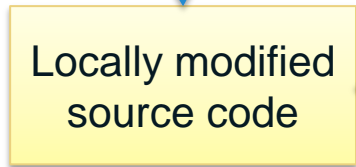
review.tizen.org



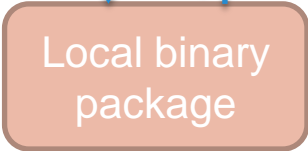
git source code repo

<https://source.tizen.org>
Building tizen from scratch

Image creation



Package Compilation



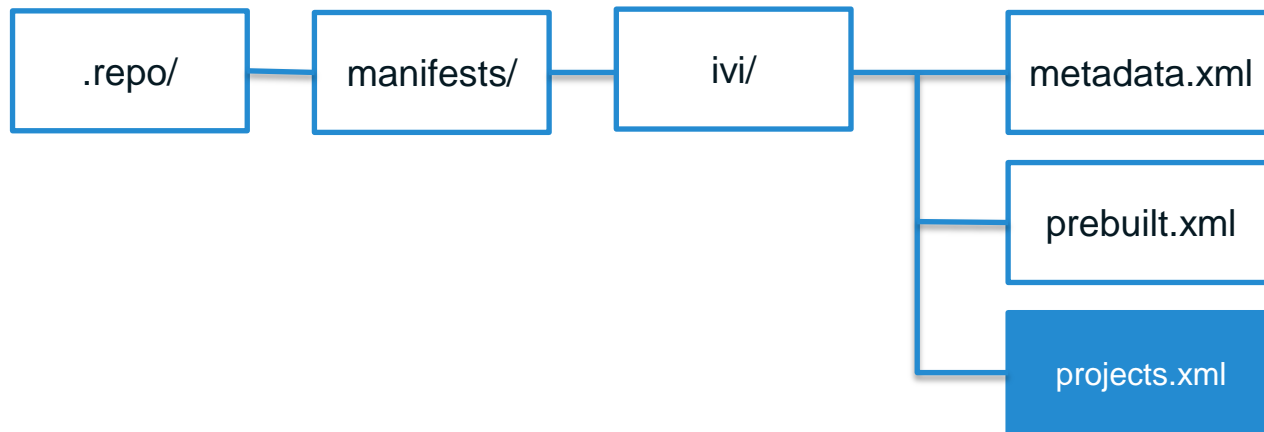
Flash onto target system



Source Code and Build Preparation

■ Get source code

```
$ repo init -u review.tizen.org:scm/manifest -b tizen -m ivi.xml
```

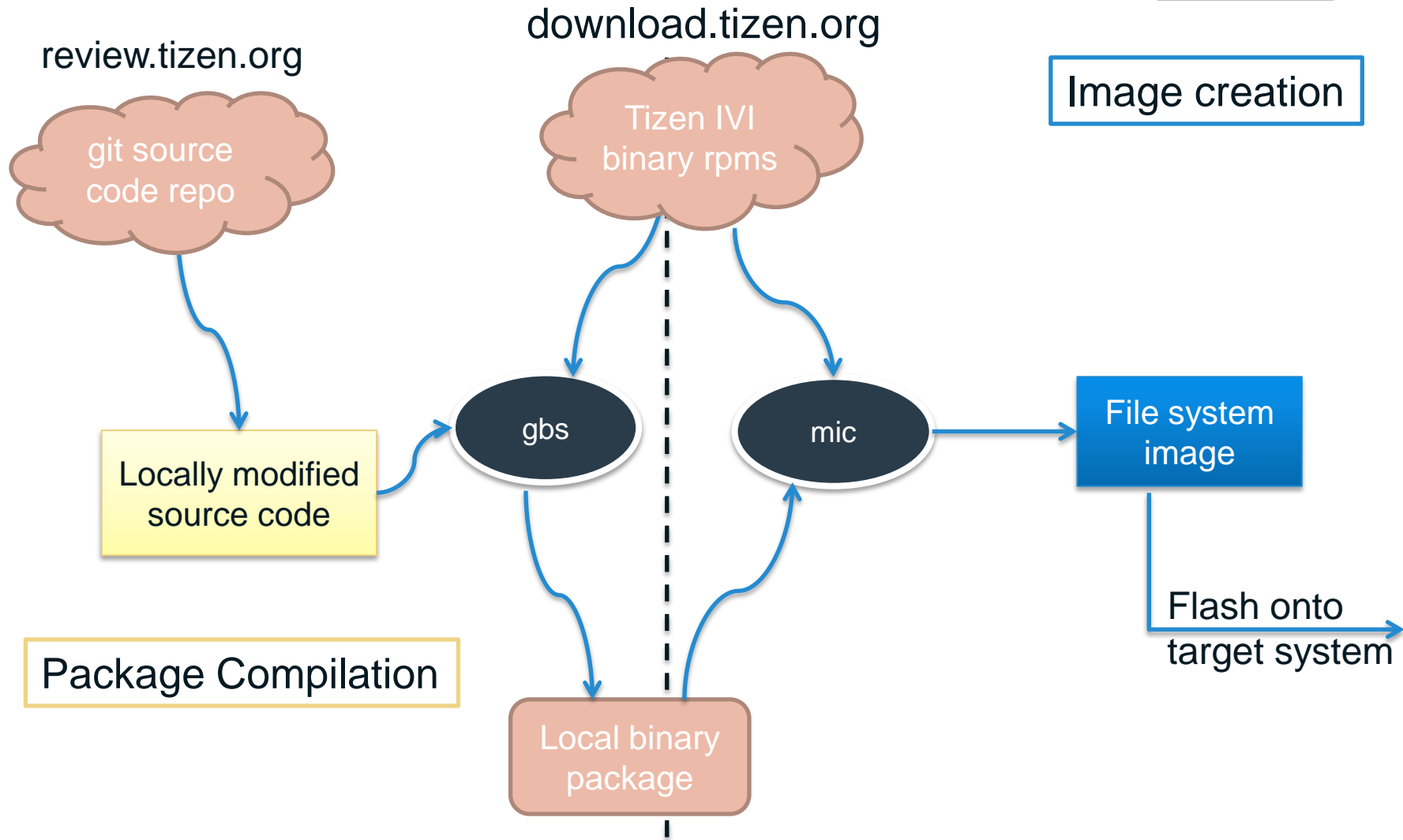


■ Overwrite `projects.xml` with milestone manifest file

[http://download.tizen.org/\\${RELEASE_PATH}/builddata/manifest/xxx.xml](http://download.tizen.org/${RELEASE_PATH}/builddata/manifest/xxx.xml)

■ Customize `projects.xml`

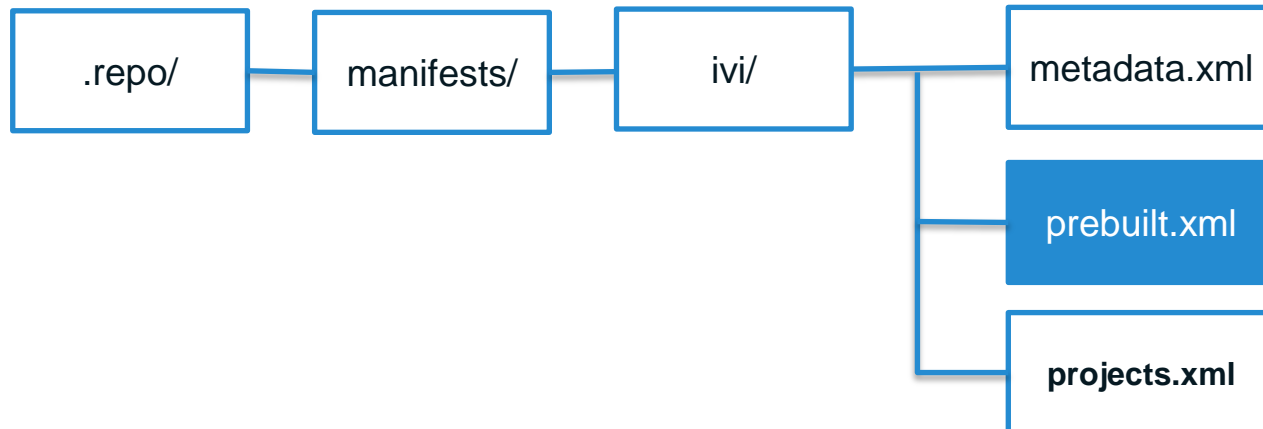
Using Tizen IVI Repos



Build Preparation (cont.)

- Use prebuilt ARM toolchain from **tizen branch**

```
-<project name="pre-built/toolchain-arm" ... revision="tizen-ivi"/>  
+<project name="pre-built/toolchain-arm" ... revision="tizen"/>
```



\$ repo sync

■ Wayland/Weston (windows system) backend

- Use PowerVR driver instead of Mesa

■ Web Applications

- Implement WaylandBufferManager (for WebKit)

■ Multimedia Acceleration Video Playback

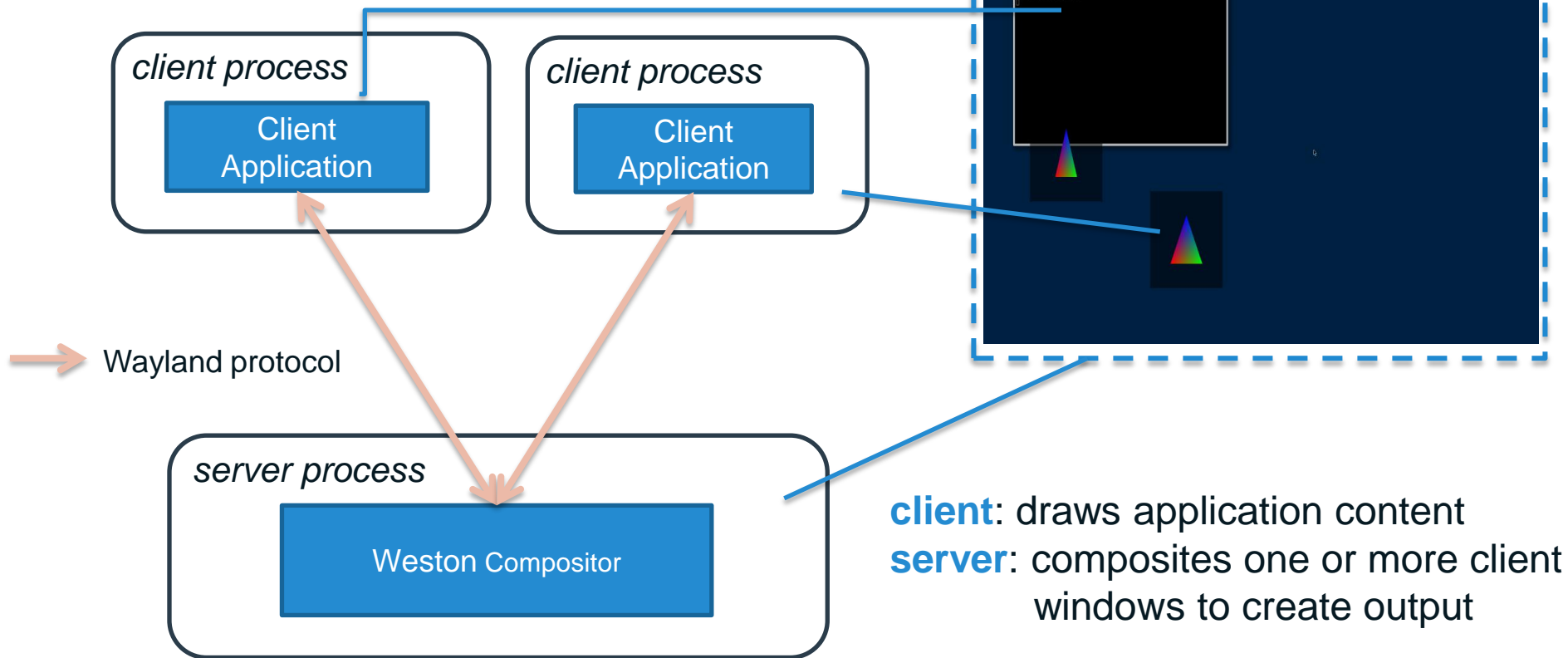
- 0 – copy video stream processing (1080p @ 30fps)

Replacing the Mesa driver for Wayland/Weston

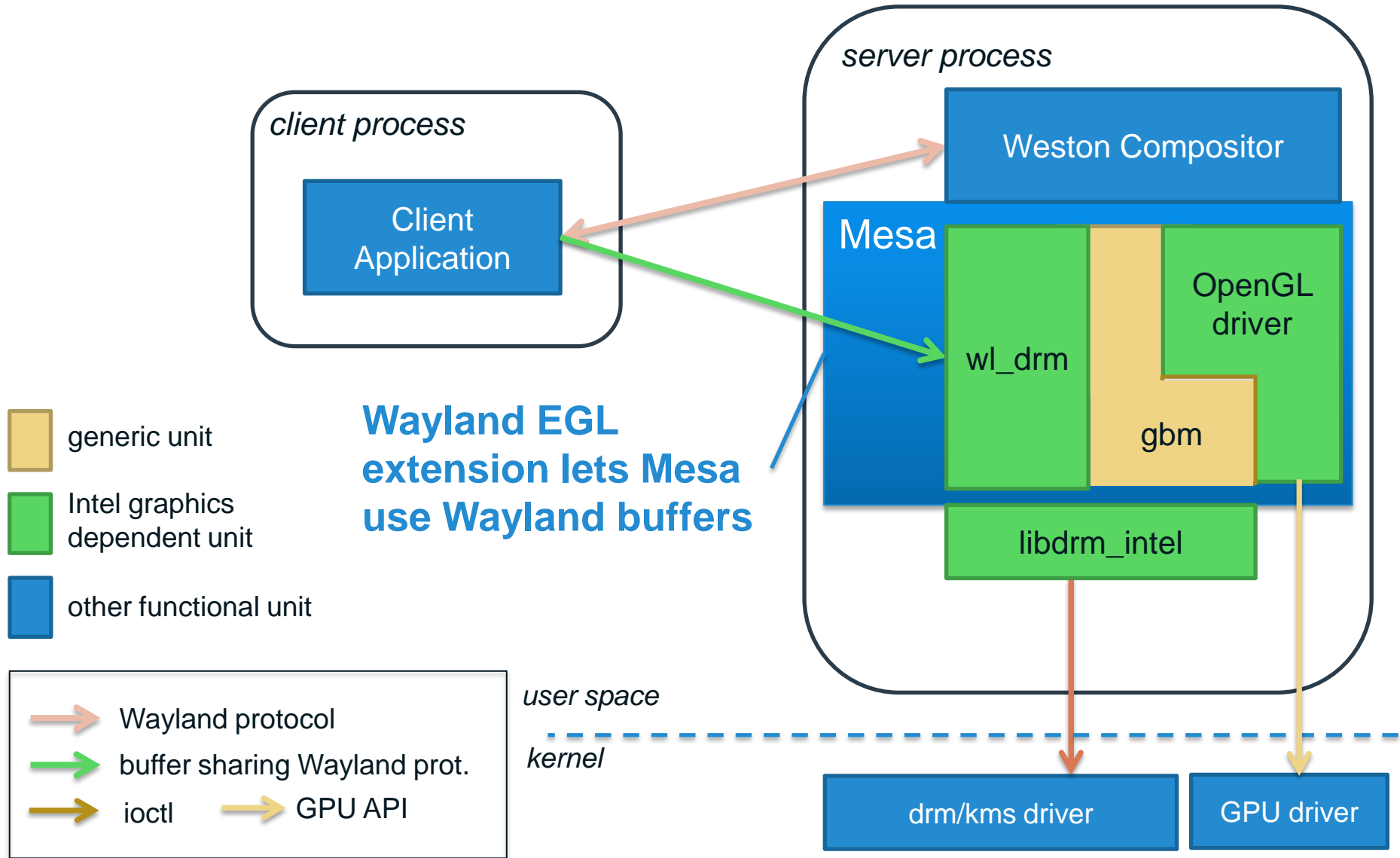
Wayland/Weston Overview



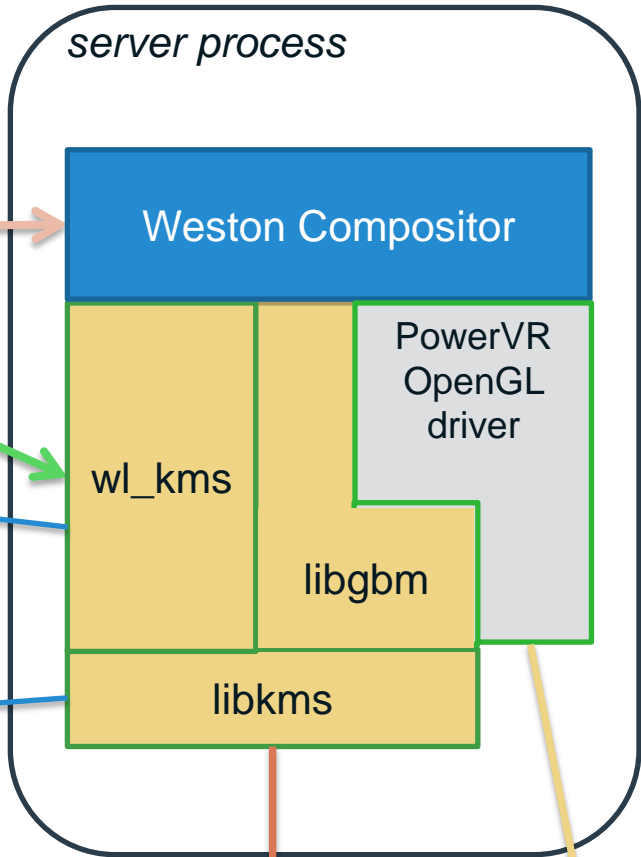
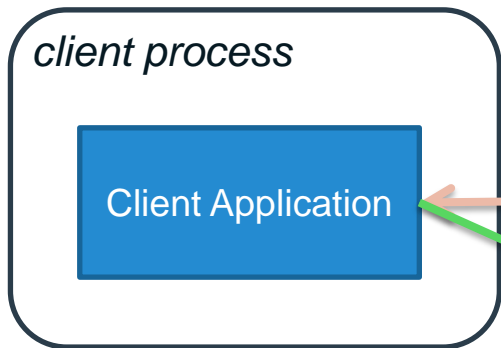
client/server based windowing system



Wayland/Weston with Mesa



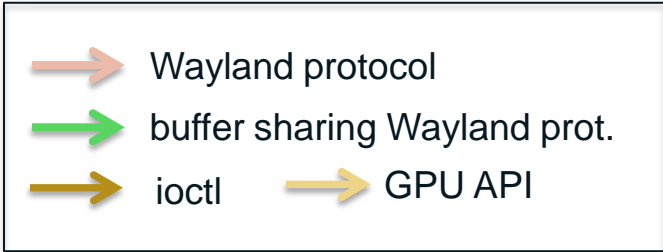
Wayland/Weston on R-Car M2/H2



**almost same as wl_drm
but with libkms back end**

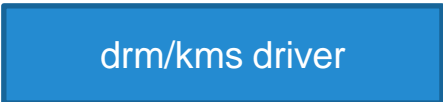
**uses generic dumb
buffer backend**

- generic library
- proprietary library
- other functional unit



user space

kernel



Replacement libraries must

- Implement EGL_WL_bind_wayland_display EGL extension for Open GL/ES driver

http://cgit.freedesktop.org/mesa/mesa/tree/docs/specs/WL_bind_wayland_display.spec

- Provide
 - libgbm – Access to drm backend (<https://github.com/robclark/libgbm>)
 - libdrm/libkms – for access to memory buffers (provided in Tizen release)
 - buffer sharing interface – (similar to Mesa's wl_drm)
- libgbm backend should match buffer sharing interface

Replacing Mesa on Tizen

1. replace mesa library

```
$ rm -r <build directory>/platform/upstream/mesa  
$ cp my_libraries <build directory>
```

2. edit build.conf (build settings file)

```
-%define with_mesa=1  
...  
+Substitute: pkgconfig(gl)  
+Substitute: mesa-devel pkgconfig(gles20)  
...  
Macros  
-%with_mesa=1
```

3. build the system

```
$ gbs build -A armv7l
```

(for full build command line see <http://source.tizen.org>
“building Tizen from scratch”)

Objective

■ Tizen IVI 3.0 on R-Car M2/H2

1. Native Applications

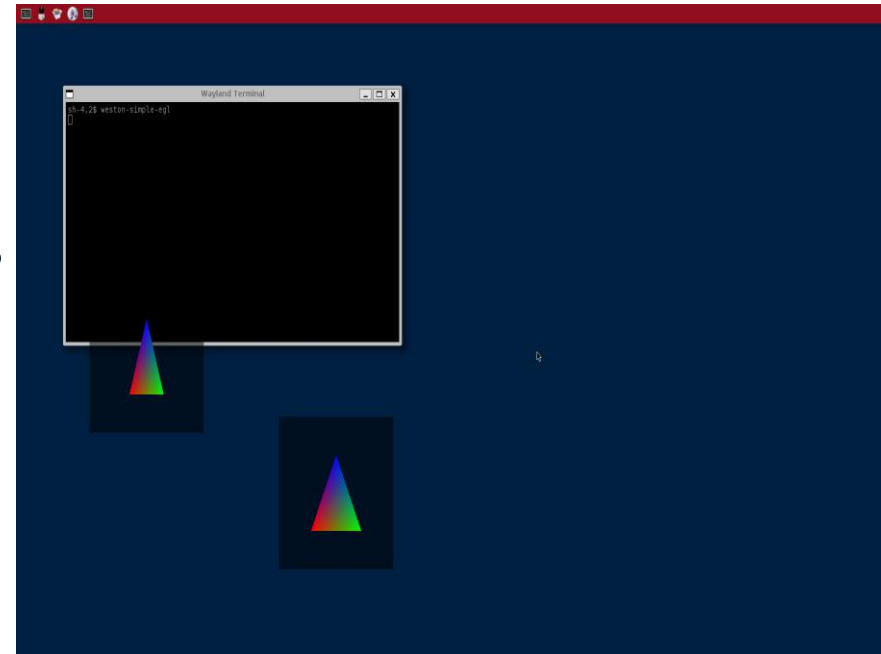
- Terminal program
- Open GLES applications

2. Web

- Browser and web applications

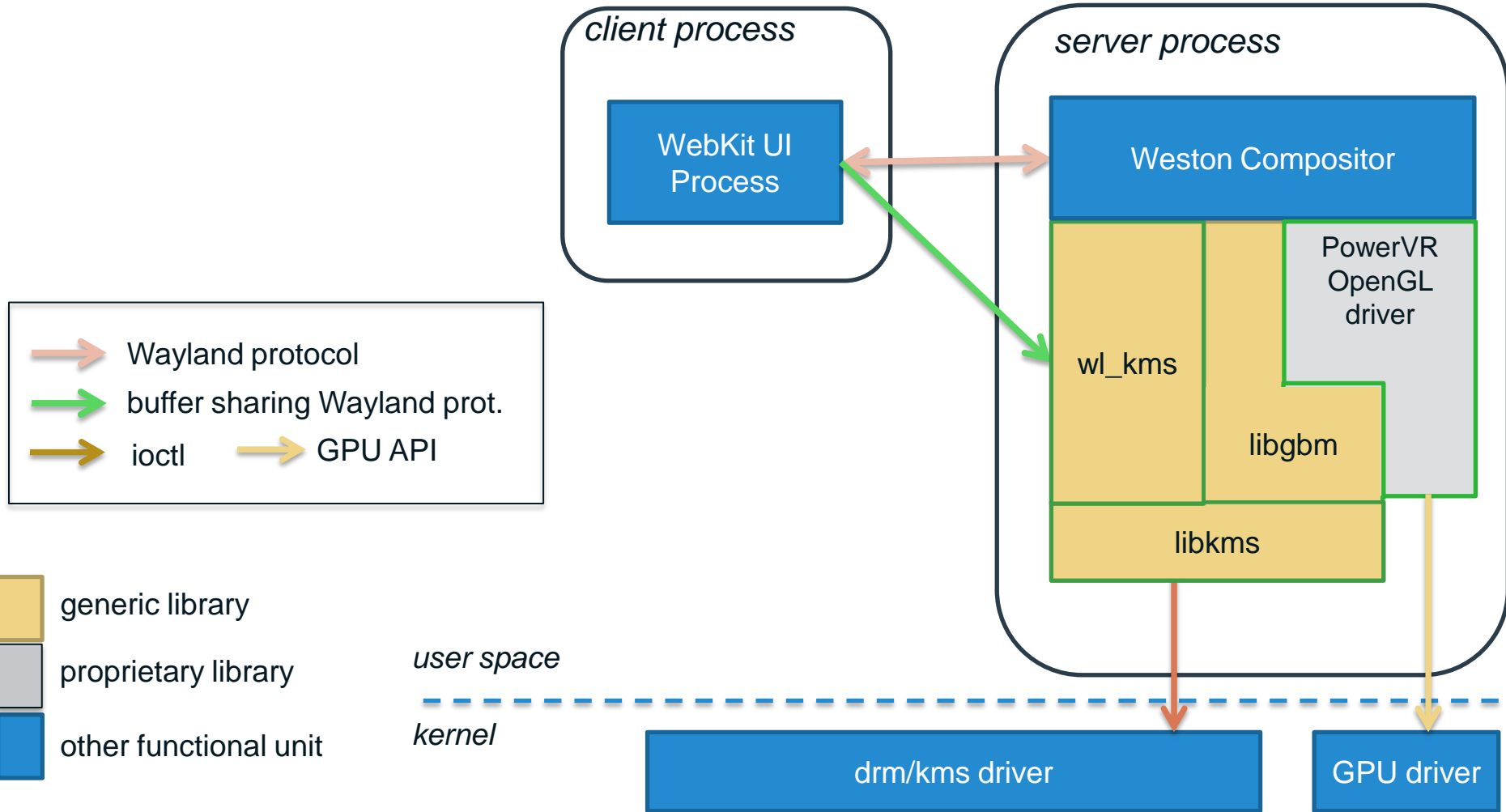
3. Multimedia

- Video playback (1080p @ 30fps)

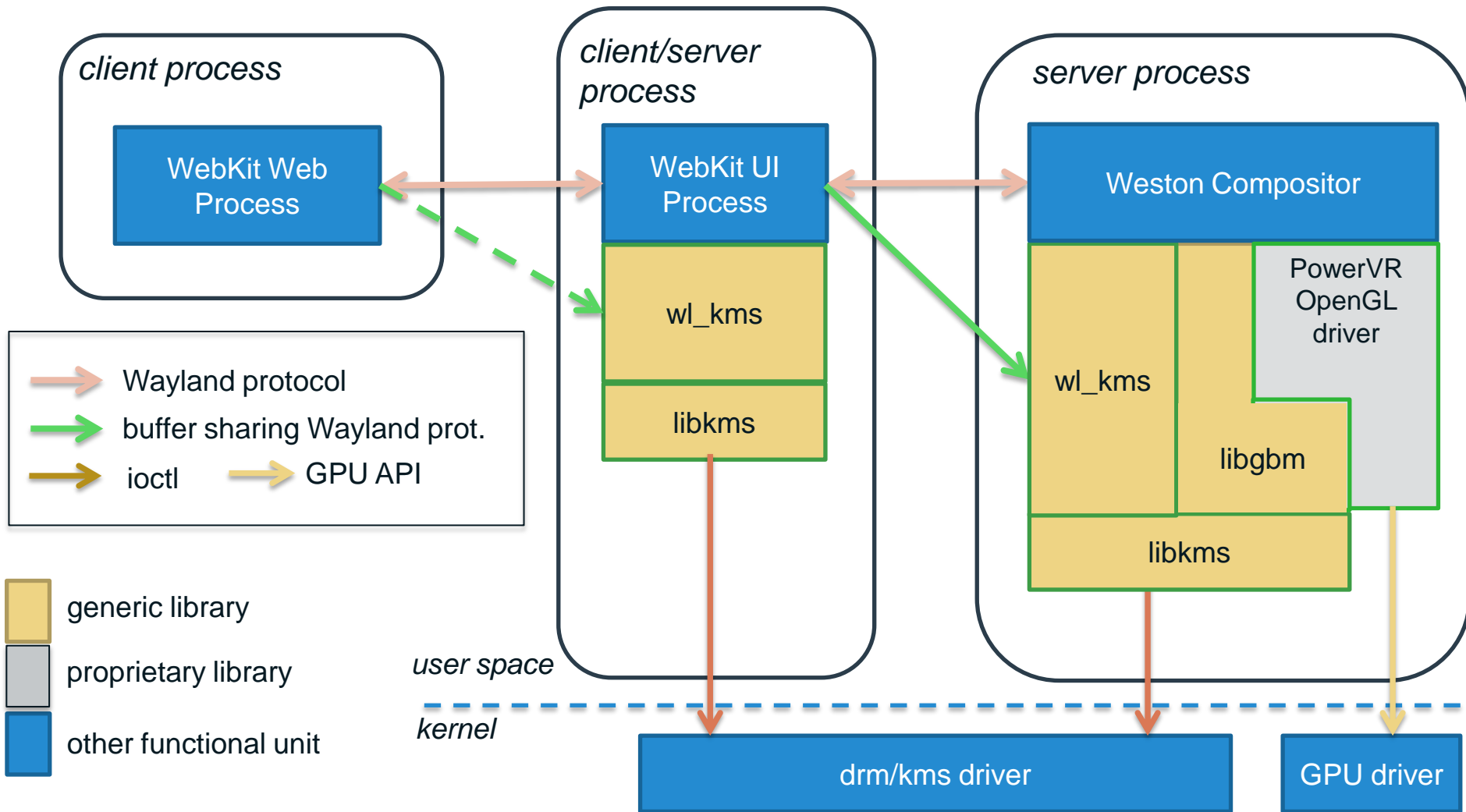


Webkit2 and WaylandBuffer Manager

Simple client-server configuration



Webkit2 client-client/server-server configuration



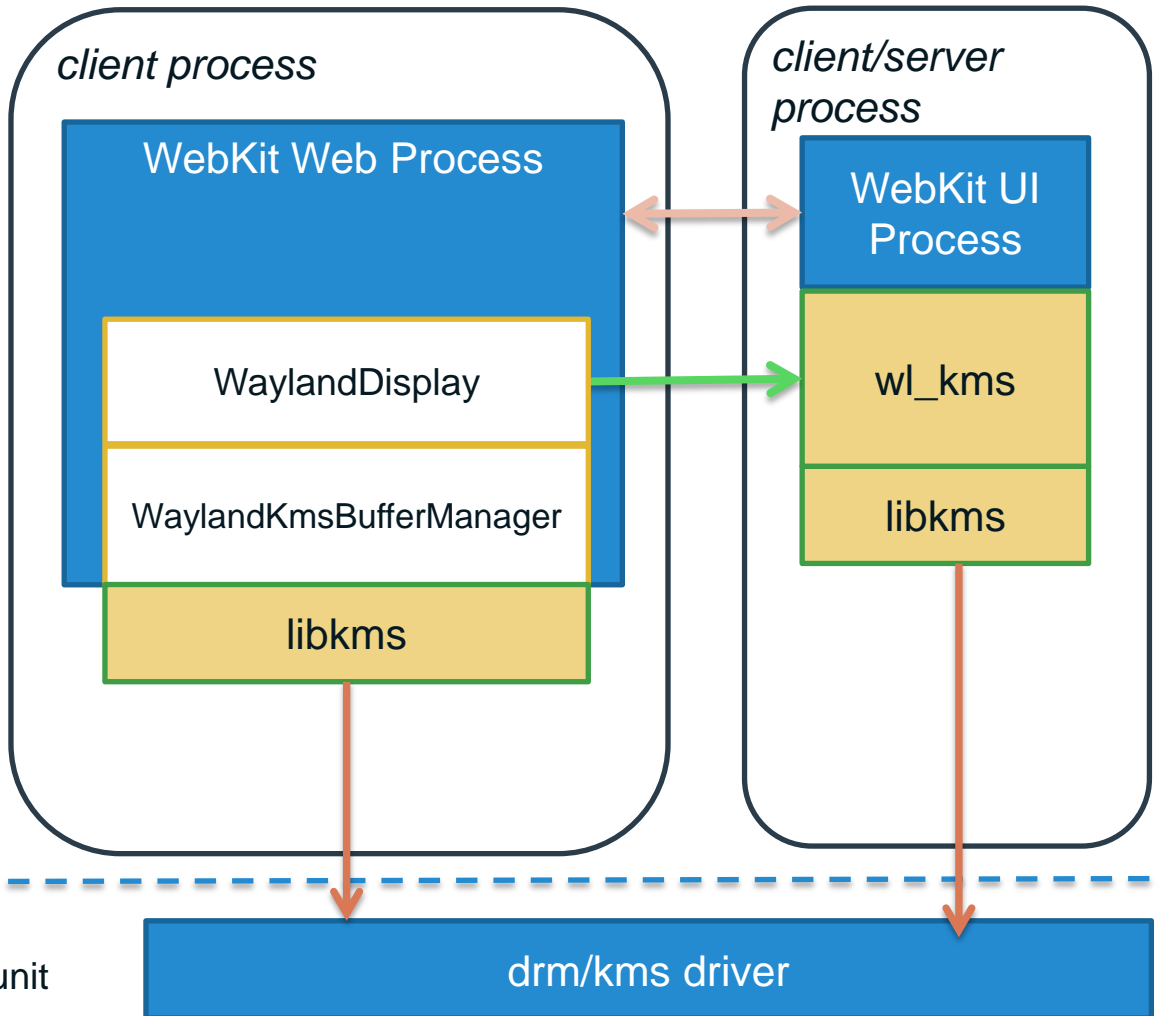
Webkit2 Buffer Allocation

WaylandDisplay (class):
Update to use wl_kms instead of wl_drm

WaylandKmsBufferManager (class):
Implementation of WaylandBufferManager interface

- Wayland protocol
- buffer sharing Wayland prot.
- ioctl

generic library other functional unit



WaylandBufferManager Interface

■ WaylandBufferManager and WaylandDisplay source:

webkit-efl/Source/WebCore/platform/graphics/surfaces/wayland/

■ Interface for allocating/locking shareable buffers (e.g.. kms_bo)

- allocateBO returns `handleId`.
- `*handle` is pointer to shareable fd (ie. flinked fd, or DMABuf handle)
- `query` to get buffer virtual address

```
class WaylandBufferManager {
    allocateBO(w, h, stride, size, align, *handle);
    lockSurface(handleId);
    unlockSurface(handleId);
    freeBO(handleId);
    query(handleId, **addr);
}
```

Objective

■ Tizen IVI 3.0 on R-Car M2/H2

1. Native Applications

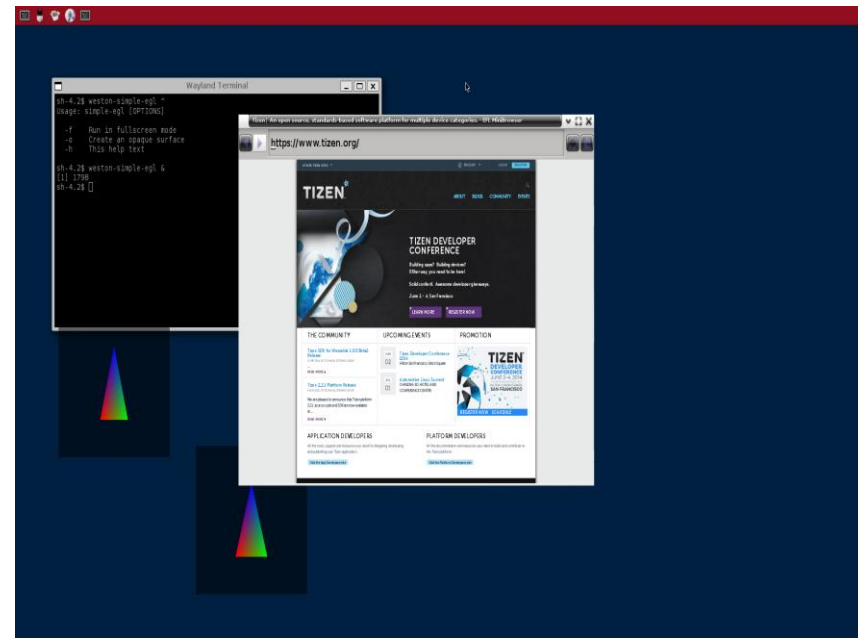
- Terminal program
- Open GLES applications

2. Web

- Browser and web applications

3. Multimedia

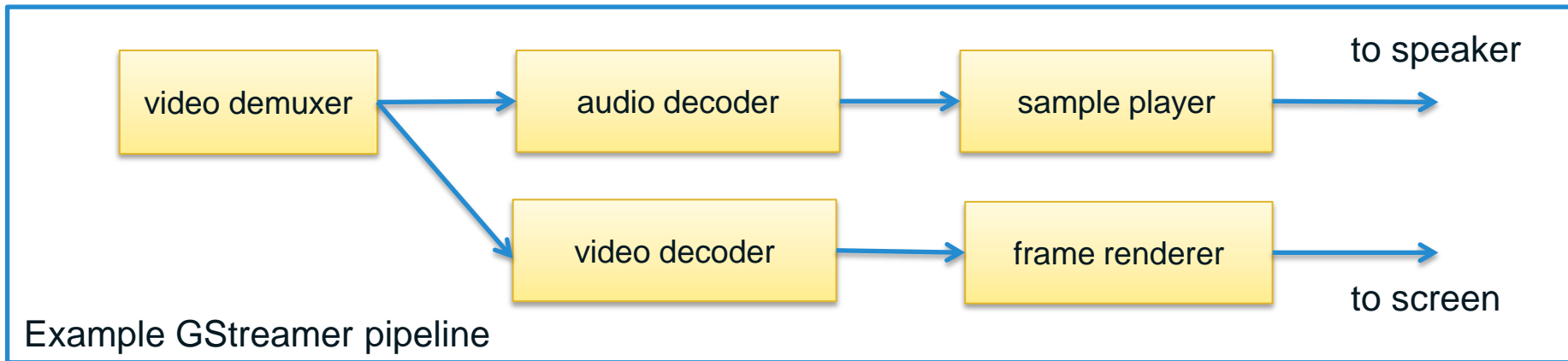
- Video playback (1080p @ 30fps)



Using GStreamer with Tizen IVI 3.0

GStreamer

- Encode, decode, capture and display multimedia data
- Make a pipeline of components to do what you want



Video Decode on R-Car M2/H2 on Tizen IVI 3.0

■ Audio pipeline

- Software decode for now

■ Video decode

- Use gst-omx to bridge GStreamer to OpenMAX IL component

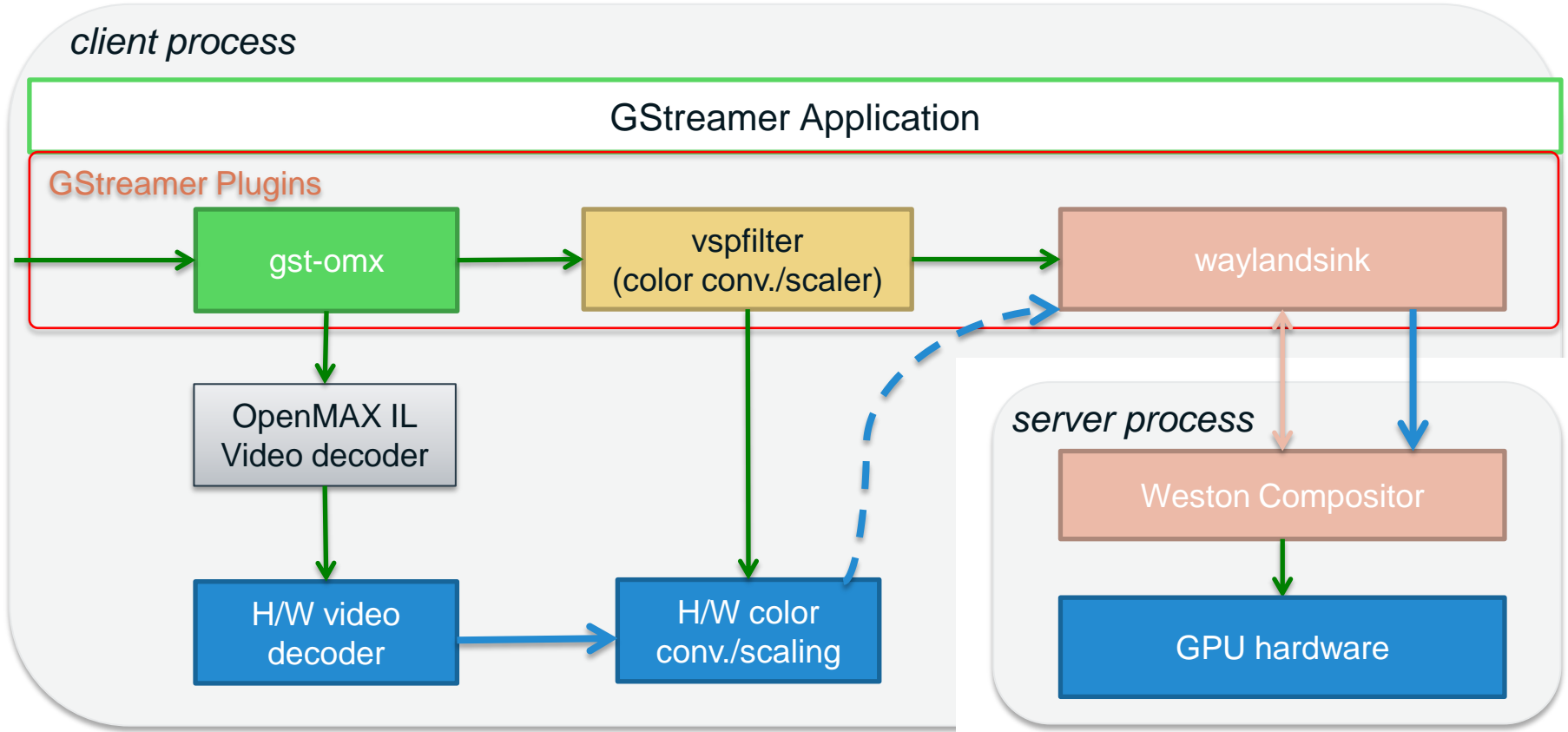
■ Color conversion/scaling

- Use hardware accelerated color conversion/scaling module

■ Display

- Use waylandsink to display via Weston compositor

GStreamer H/W accelerated video decode



Wayland protocol API call data flow memcpy()

full custom as-is upstream component customized component Reneas proprietary library

■ **H/W color conversion requires physically contiguous buffers**

- Waylandsink allocates non-contiguous shared memory buffers
- Add extra memcpy()s into pipeline.

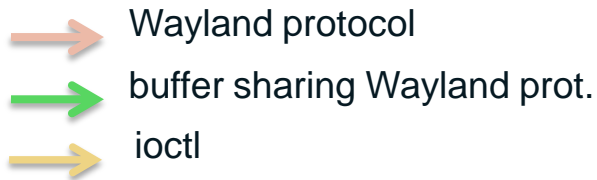
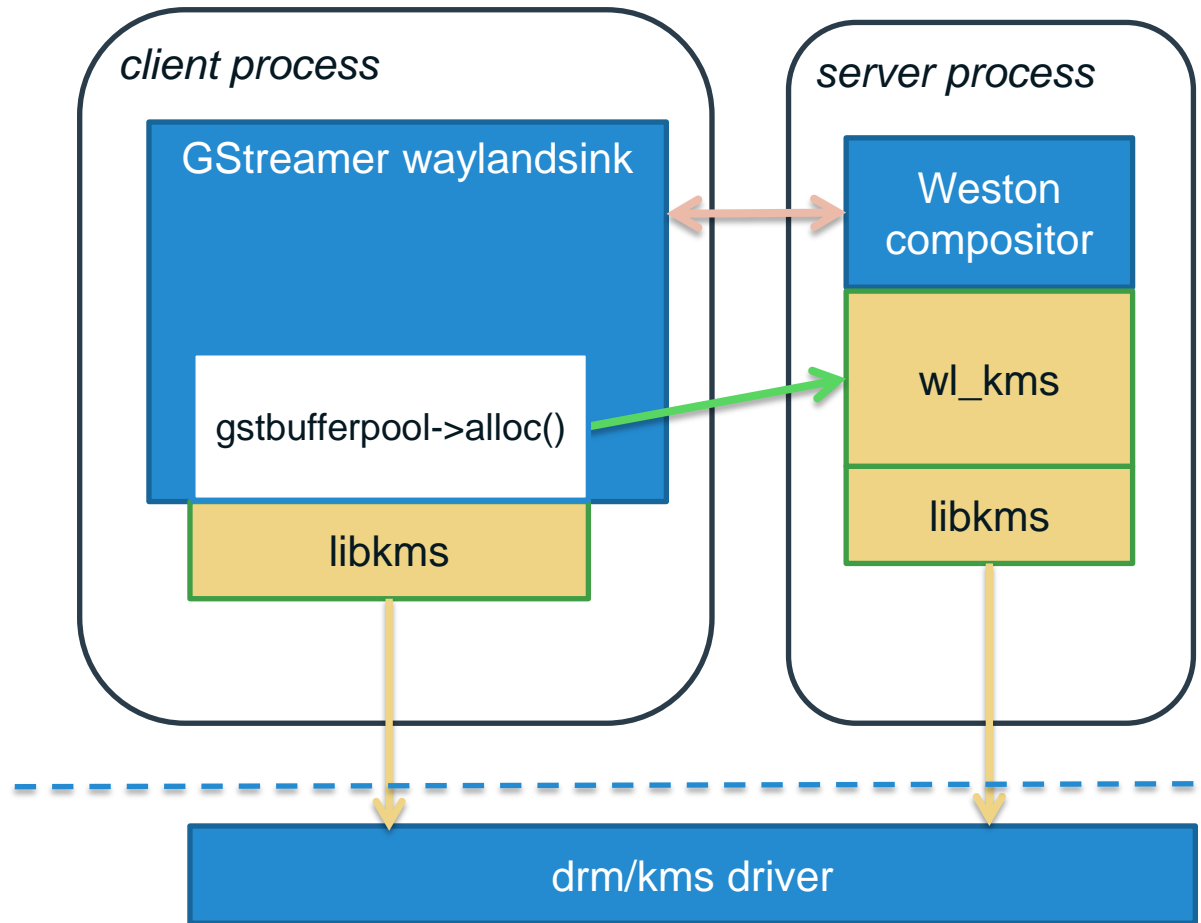
■ **Buffers allocated from kms bo are physically contiguous (on our system)**

- Use the same method as with WebKit to allocate and share graphics buffers

Waylandsink customized for libkms usage

Allocated kms dumb buffers used for H/W color conversion.

No memcopy() required between video decode and screen display.



GStreamer H/W accelerated video decode no memcpy()

client process

GStreamer Application

GStreamer Plugins

gst-omx

vspfilter
(color conv./scaler)

waylandsink

OpenMAX IL
Video decoder

H/W video
decoder

H/W color
conv./scaling

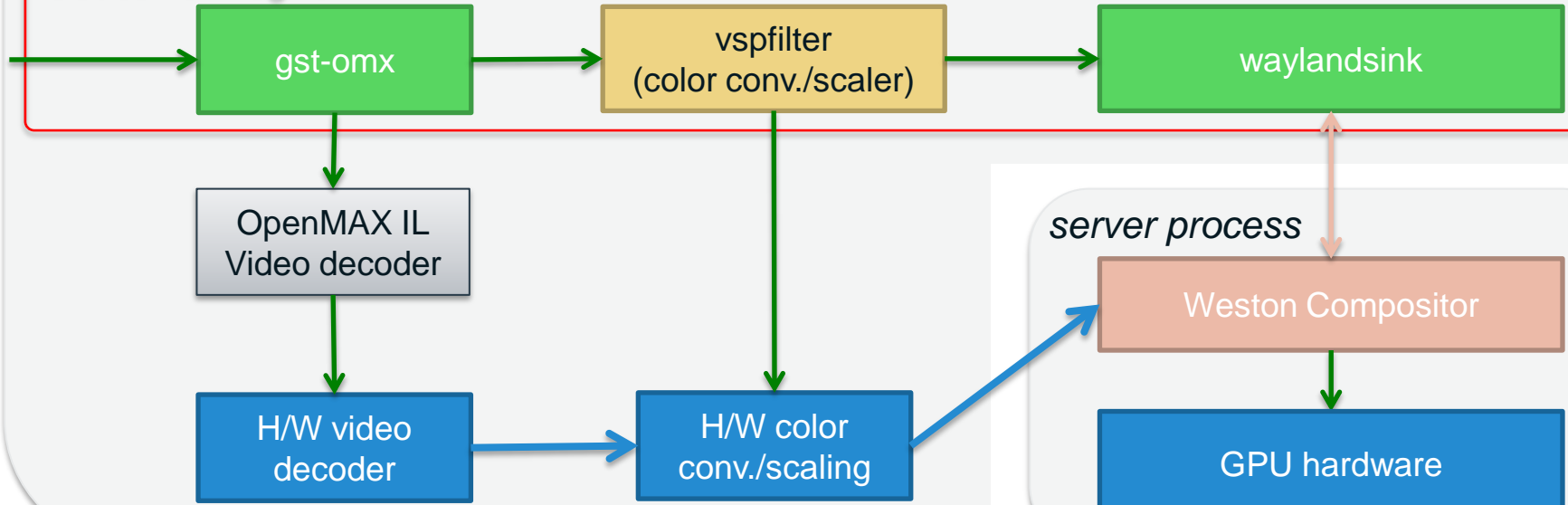
server process

Weston Compositor

GPU hardware

Wayland protocol API call data flow

full custom as-is upstream component customized component Reneas proprietary library



Objective

■ Tizen IVI 3.0 on R-Car M2/H2

1. Native Applications

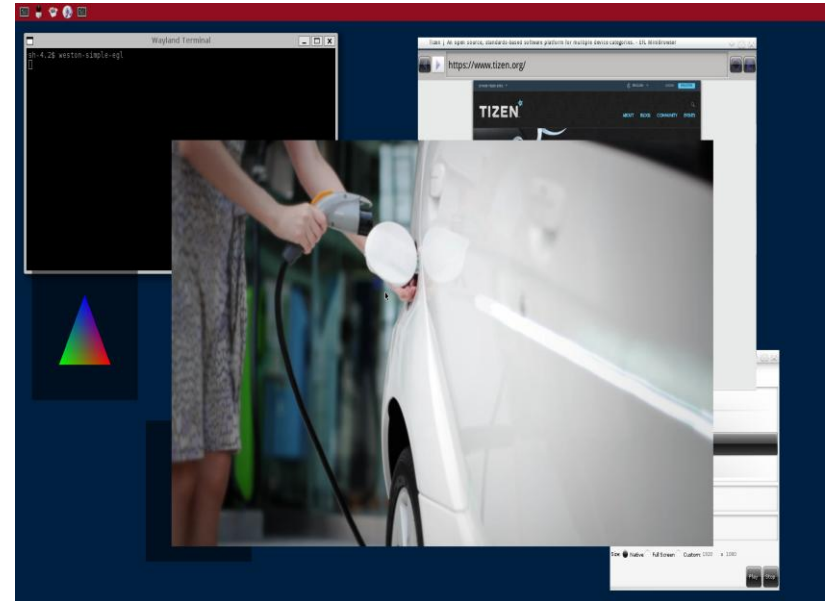
- Terminal program
- Open GLES applications

2. Web

- Browser and web applications

3. Multimedia

- Video playback (1080p @ 30fps)



What we learned - review

■ Building

- Use manifest xml file from milestone release on <http://download.tizen.org>
- Use mobile toolchain for ARM

■ Weston/Wayland

- Need support for EGL_WL_bind_wayland_display in Open GL/ES driver
- Can use libkms dumb buffers

■ WebKit

- Implement WaylandBufferManager; update WaylandDisplay

■ Multimedia playback

- Use libkms and Wayland buffer sharing to implement 0-copy processing with physically contiguous memory buffers

Thank you.

Questions?

■ **Building Tizen from scratch**

<https://source.tizen.org/documentation/developer-guide/all-one-instructions/creating-tizen-images-scratch-one-page>

■ **EGL_WL_bind_wayland_display EGL extension**

http://cgit.freedesktop.org/mesa/mesa/tree/docs/specs/WL_bind_wayland_display.spec

■ **libgbm**

<https://github.com/robclark/libgbm>

■ **Renesas R-Car series platforms**

http://am.renesas.com/applications/automotive/cis/cis_highend/