

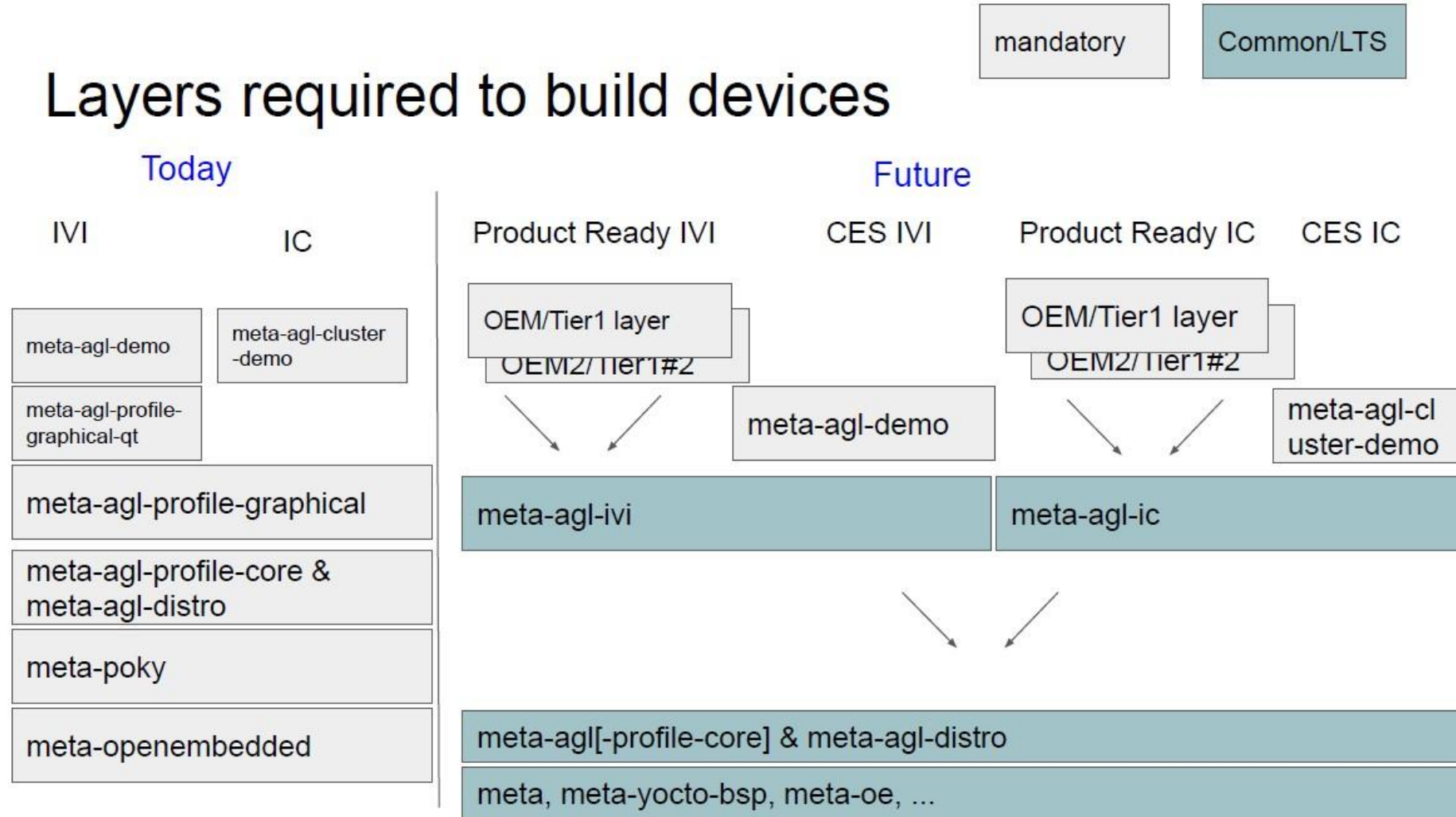


# AGL Layer Design - Rethink

Tadao Tanikawa @ AGL Developer Community

# Basically, I agree with

## Layers required to build devices

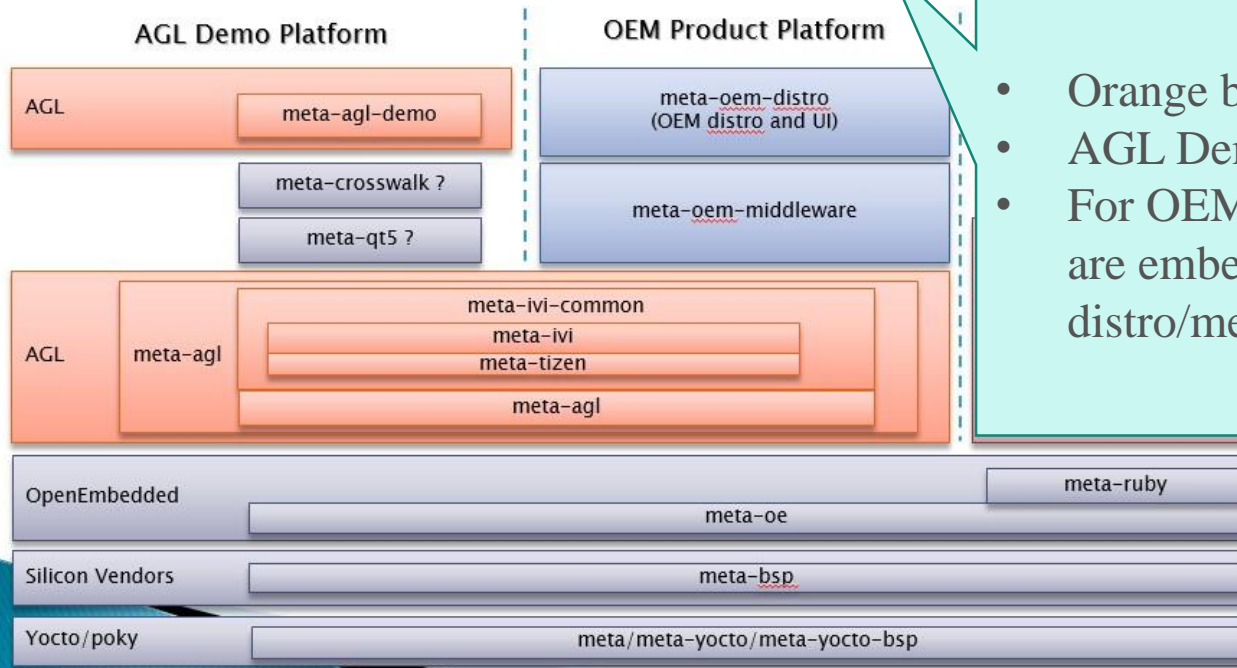


# Originally, it was very simple

## AGL Layer Design

Slide created at 2015/06/22

- オレンジの箱の範囲が、AGL Distribution
- AGL Demo Platformがリファレンス実装
- OEM製品では、独自のフレームワークやproprietaryなソフトウェアが搭載される想定



- Orange boxes are included in AGL Distribution
- AGL Demo Platform is a reference implementation of AGL
- For OEM products, own framework or proprietary software are embedded to own meta-layers (e.g. meta-oem-distro/meta-oem-middleware)

**Proposed layers are similar to this**

# AGL Activity for IVI Product Readiness (Proposal)

## Activity for IVI Product Readiness (proposal)

- Define what is Product Readiness
- Define system requirements
- Define system architecture
- Complete system design
- etc

To be discussed  
in Special SAT (so far)

**IVI-Profile** (original) continued as meta-agl-demo

Refer

The goal is to challenge  
new technologies.

Develop DEMO / PoC based on  
meta-agl-ivi (if possible)

migrate

We discuss in EG and SAT for each Item before migration

**meta-agl-ivi** (NEW) started from empty

The goal is to be able to  
develop products based on  
AGL

We discuss in EG and SAT for each Item before merging

merge

**OEM / Tier 1 layer**

Develop products  
based on meta-agl-ivi (GOAL of IVI PR)

**Product development (NON\_AGL\_UCB)**

# A study: What is product readiness of AGL?

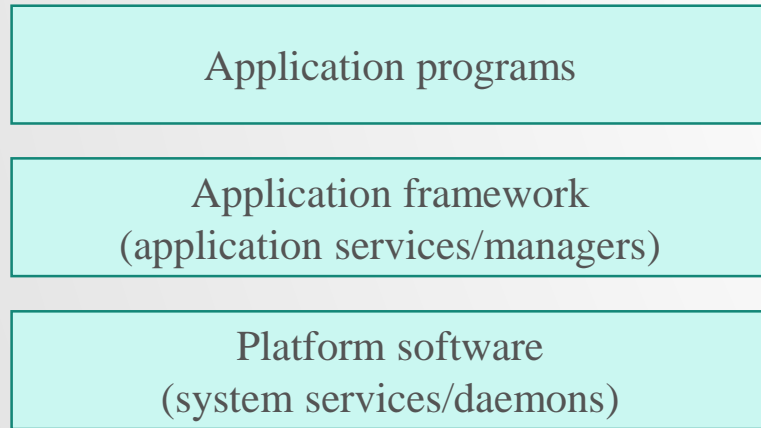
---

- Goal: reducing cost of delivering OSS to products
  - What is major issues?
    - Functionality
    - Quality
  - GAP from functionality point of view
    - Missing features for automotive use cases
    - Missing functions to implements those missing features.
  - To fill these gaps, roughly 2 ways
    - Extend existing software → Re-use existing OSS and modify it
    - Develop new software → Create OSS projects inside/outside AGL UCB

# Basic software integration architecture

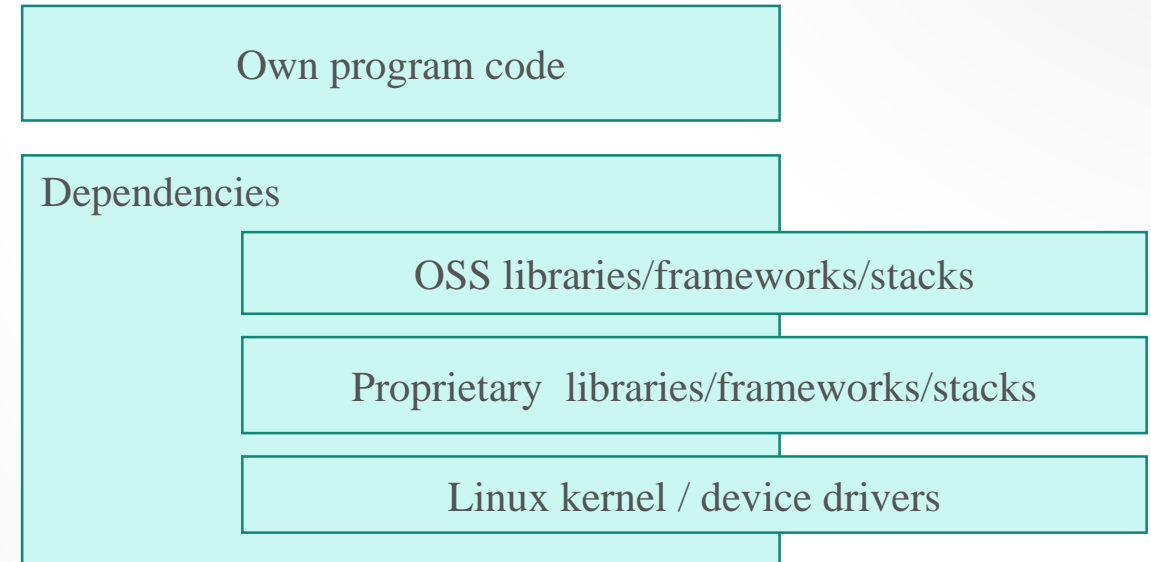
---

Software stack



Defined by AGL layers

Software structure



Defined by AGL (bitbake) recipes  
Selected by AGL features

# Today's AGL layers, meta-agl-xxx

---

- bsp
- build
- external
- meta-agl
- meta-agl-cluster-demo
- meta-agl-demo
- meta-agl-devel
- meta-agl-extra
- meta-agl-telematics-demo

- meta-agl
- meta-agl-bsp
- meta-agl-distro
- meta-agl-profile-cluster
- meta-agl-profile-cluster-qt5
- meta-agl-profile-core
- meta-agl-profile-graphical
- meta-agl-profile-graphical-html5
- meta-agl-profile-graphical-qt5
- meta-agl-profile-hud
- meta-agl-profile-telematics
- meta-app-framework
- meta-netboot
- meta-pipewire
- meta-security

**Difficult to understand**

# Today's AGL features

---

Available features:

```
[meta-agl]
agl-all-features :( agl-demo agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-pipewire agl-speech-framework agl-netboot )
agl-appfw-smack
agl-archiver
agl-buildstats
agl-ci
agl-ci-change-features :( agl-demo agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-pipewire agl-speech-framework agl-devel agl-netboot agl-pipewire agl-buildstats agl-ptest )
agl-ci-change-features-nogfx :( agl-demo agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-pipewire agl-speech-framework agl-devel agl-netboot agl-pipewire agl-buildstats agl-ptest )
agl-ci-snapshot-features :( agl-demo agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-pipewire agl-speech-framework agl-devel agl-netboot agl-archiver agl-pipewire agl-buildstats agl-ptest )
agl-ci-snapshot-features-nogfx :( agl-demo agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-pipewire agl-speech-framework agl-devel agl-netboot agl-archiver agl-pipewire agl-buildstats agl-ptest )
agl-devel
agl-egvirt
agl-fosdriver
agl-gplv2
agl-hmi-framework
agl-netboot
agl-pipewire
agl-profile-cluster :( agl-profile-graphical )
agl-profile-cluster-qt5 :( agl-profile-graphical-qt5 agl-profile-graphical )
agl-profile-graphical
agl-profile-graphical-html5 :( agl-profile-graphical )
agl-profile-graphical-qt5 :( agl-profile-graphical )
agl-profile-hud
agl-profile-telematics
agl-ptest
agl-sign-wgts
agl-sota
agl-virt
agl-virt-guest-xen
agl-virt-xen :( agl-virt )
agl-weston-remoting :( agl-profile-graphical )
[meta-agl-cluster-demo]
agl-cluster-demo :( agl-profile-cluster-qt5 agl-profile-graphical-qt5 agl-profile-graphical agl-hmi-framework )
agl-cluster-demo-preload
[meta-agl-demo]
agl-cloudproxy
agl-cluster-demo-support :( agl-weston-remoting agl-profile-graphical )
agl-demo :( agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-pipewire agl-speech-framework )
agl-demo-preload
agl-demo-soundmanager :( agl-appfw-smack agl-hmi-framework agl-profile-graphical-qt5 agl-profile-graphical agl-audio-soundmanager-framework )
[meta-agl-devel]
agl-jailhouse
agl-speech-framework
agl-voiceagent-alexa :( agl-speech-framework )
agl-voiceagent-alexa-wakeword :( agl-voiceagent-alexa agl-speech-framework )
[meta-agl-extra]
agl-localdev
[meta-agl-telematics-demo]
agl-telematics-demo :( agl-profile-telematics )
```

**Difficult to understand**



# Goals of integration of IVI ProductReady Trial/1<sup>st</sup> release

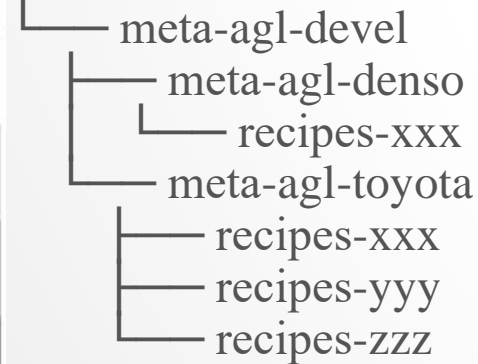
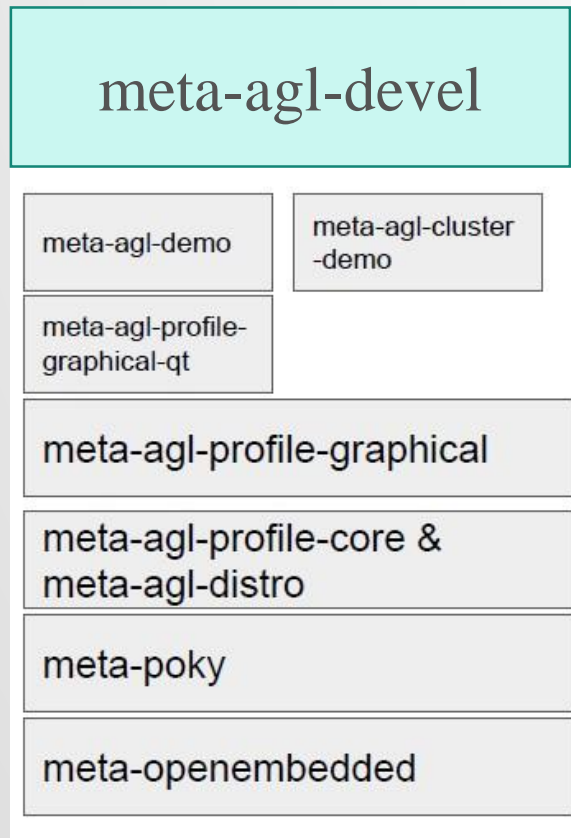
---

- IVI ProductReady Trial
  - Re-use existing AGL UCB (means keeping Today's layers)
    - Adding RBA from DENSO
    - Adding BaseSystem's components from TOYOTA
    - Adding Test applications/services to evaluate several features (e.g. basesystem components/RBA/...)
- 1<sup>st</sup> Release
  - Improve / Relace exising AGL components (Assuming new design)
    - AppFw?
    - App Services? (e.g. BT / Radio)
    - ...

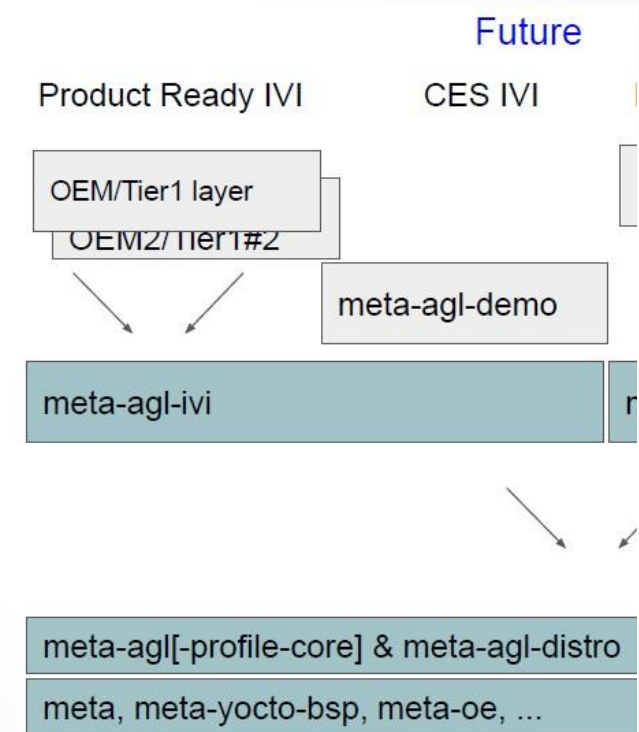
# How to integration of Trial

- All new components are integrated into meta-agl-devel once
  - After new layers come, migrates all recipes of IVI in meta-agl-devel to proper layers (e.g. meta-agl-ivi and OEM layer)

## Trial



## 1<sup>st</sup> Release



# How to integration of Trial (cont.)

---

- Issues still remain: how to add AGL features for new components

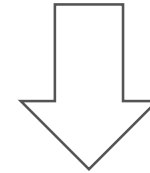
E.g. today's basesystem in staging

Available features:

[meta-agl]

ns-backupmanager  
ns-commonlibrary  
ns-frameworkunified  
ns-loglibrary  
ns-notificationpersistent  
os-eventlibrary  
os-posixbasedos001legacylibrary  
os-rpclibrary  
os-vehicleparameterlibrary  
ps-communication  
ss-config  
ss-interfaceunified  
ss-loggerservice  
ss-powerservice  
ss-resourcemanager  
ss-romaccesslibrary  
ss-systemmanager  
ss-taskmanager  
ss-versionlibrary  
vs-positioning  
vs-positioningbaselibrary

**Too complicated (too detailed)**



**Need to organize into  
Dependency of packages (recipes) and  
AGL Features (templates)**

# Reworking: staging/toyota.git

---

- Today's staging/toyota.git has lots of issues if followed current meta-agl's rule and policy of integration
  - All recipes merged into meta-agl already and commit whole recipes by 1 supermassive commit (not broken down into patches for individual elements/components)
  - Unclear integration policy for AGL AppFw non-compatible services
  - File structure of the recipes that violate the unspoken rule (e.g. complete source code files should not be included)
  - Packages of unknown purpose (e.g. kernel-module-evklib)
  - Non-HMI/GUI related packages are included into recipe-graphics
  - Multiple elements combined into 1 massive recipe (e.g. recipes-core/agl-systemd)
  - Unclear rule/policy of the choice between replacement and expansion existing packages (e.g. recipes-core/agl-basefiles)
  - And so on
- If ignoring these issues at Trial period, it would be a bigger challenge in the next 1<sup>st</sup> Release
  - E.g. When plan to move from an OEM-specific layer (meta-agl-devel/meta-agl-toyota) to a more generic meta-agl-ivi.

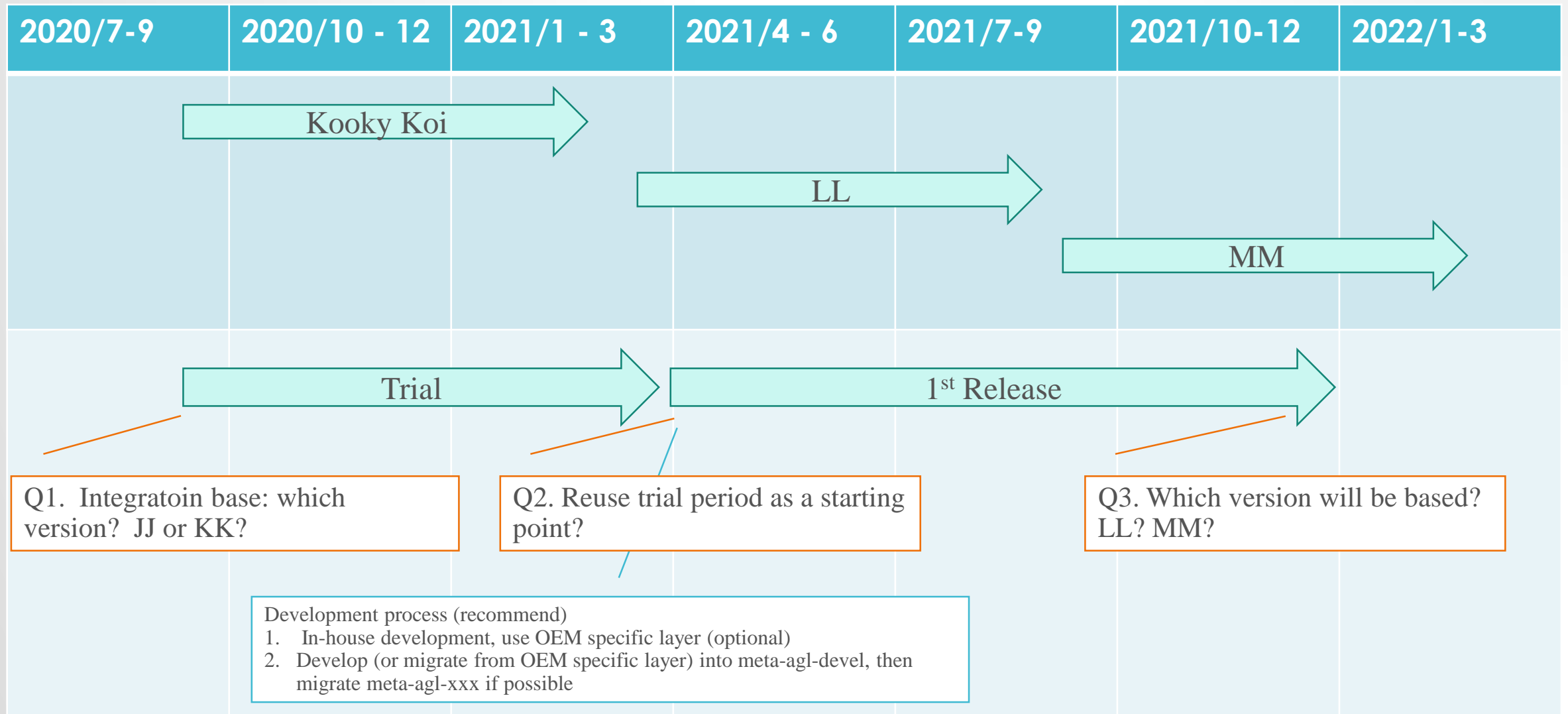
# Reworking: staging/toyota.git (cont.)

---

- Plan A
  - Trial Period
    - Keep the current recipes of staging/toyota as it is and integrate it into meta-agl-devel/meta-agl-toyota
  - 1<sup>st</sup> Release
    - Reorganize the package structure and recipes, carefully reintegrate them one by one into meta-agl-toyota and meta-agl-ivi
  - Pros
    - The amount of development during the Trial Period is minimal
  - Cons
    - Lots of work put off to the 1<sup>st</sup> Release
- Plan B
  - Trail Period
    - Reorganize the package structure and recipes, integrate them into meta-agl-devel/meta-agl-toyota once
  - 1<sup>st</sup> Release
    - Discuss and review whether it is appropriate to integrate into meta-agl-toyota or meta-agl-ivi for each package
  - Pros
    - As OSS development methodology, Plan B is better because we can focus on discussion and review for the 1<sup>st</sup> Release
  - Cons
    - The amount of development during the Trial Period would be huge

The choice is up to Toyota, because the AGL developers can only respond to reviews and, due to resource, can't really help with refactoring and development recipes (which Toyota should do voluntarily)

# How to align with UCB Release



# Sample integration how to use meta-agl-devel

- CES2020 container integration demo

Manifest branch: refs/tags/halibut/8.0.3  
Manifest merge branch: refs/heads/halibut  
Manifest groups: all, -notdefault

---

## Projects Overview

### meta-agl

\* sandbox/ruke47/ces2020\_803 ( 3 commits, Tue Jan 7 02:07:11 2020 +0900)  
- 320f3744 [RCAR] Update graphics driver  
- 5e20dda4 weston: Disable wayland backend frame  
- 576d49b3 Fix wayland-backend bug with more than 2 outputs

### meta-agl-demo

\* sandbox/ruke47/ces2020\_803 ( 1 commit, Wed Dec 25 21:15:32 2019 +0900)  
- 2c41a10f agl-container-demo: split recipe of sllin driver and service

} Only bug fixes for CES2020 demo use case

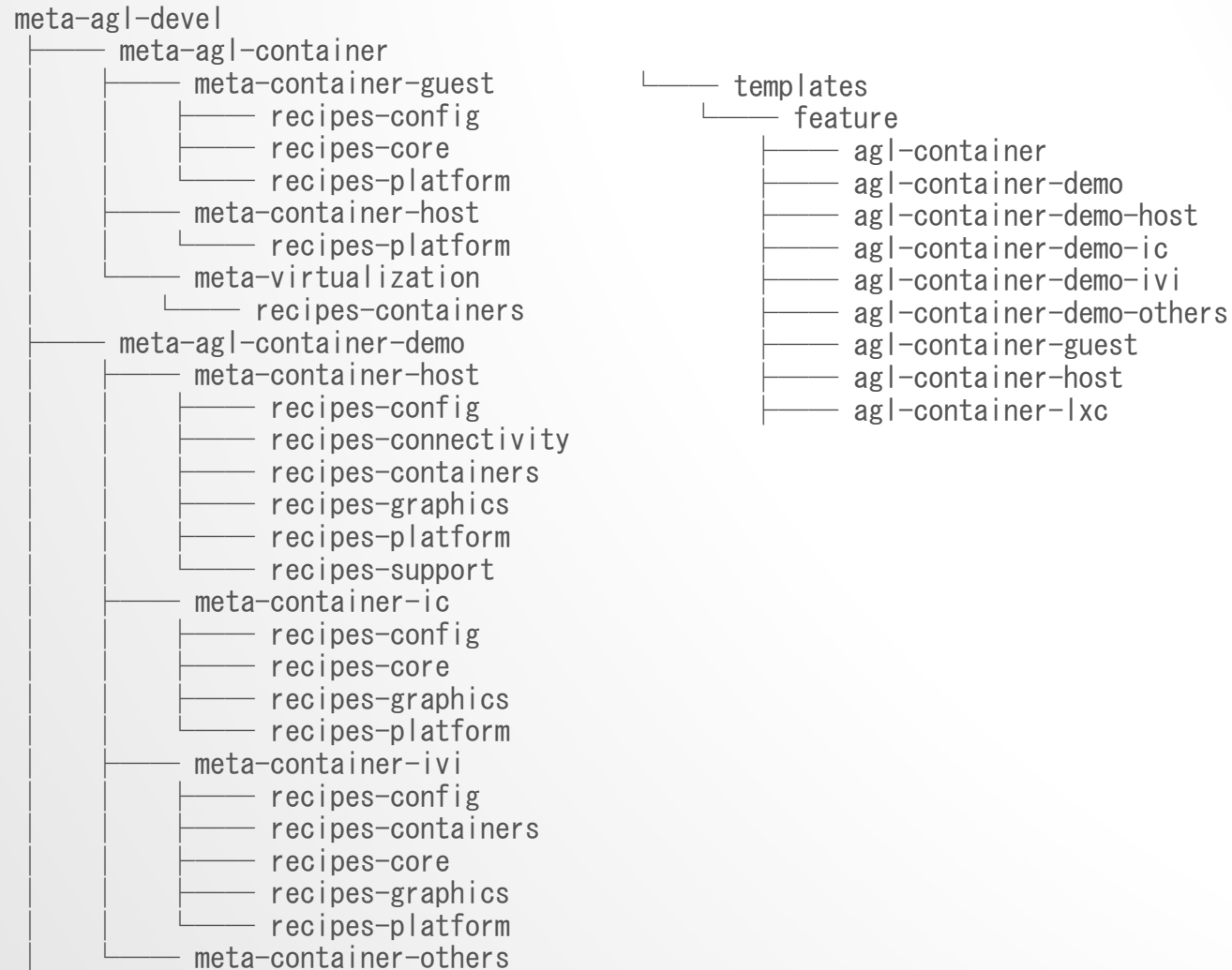
### meta-agl-devel

\* sandbox/ruke47/ces2020\_803 (24 commits, Thu Jan 30 12:06:54 2020 +0900)  
- 34fdd427 agl-container: initial import into meta-agl-devel  
- 1c617a7e agl-container: base config of LXC container  
- 7247ea0d agl-container: new packagegroup for agl-container-host  
- f5c9474c agl-container: new packagegroup for agl-container-lxc-guest  
- 9b097f00 agl-container: backport lxc 3.2.1  
- 6ea7578d agl-container-demo: initial import into meta-agl-devel  
- caa01996 agl-container-demo: new host image of agl container demo  
- 9e669054 agl-container-demo: config of guest's rootfs  
- 051d0598 agl-container-demo: setup lxc-net for host  
- 04c91ec2 agl-container-demo: setup wayland compositor of host  
- 2bc01410 agl-container-demo: lxc-net would fail if run before eth0 up  
- 971580b1 agl-container-demo: enable LIN, Radio and Audio  
- 25401314 agl-container-demo: enable gstrecord on container  
- 393e88c0 agl-container-demo: agl-demo-platform for lxc container  
- dd5adcc3 agl-container-demo: config of IVI demo container  
- 84275562 agl-container-demo: setup child weston for guest IVI  
- 73bc477b agl-container-demo: agl-cluster-demo-platform for LXC container  
- 166b7414 agl-container-demo: config of IC demo container  
- 97a321c7 agl-container-demo: setup child weston for guest IC  
- 541cb025 agl-container-demo: enable Radio of Kingfisher  
- 89ecdcfe agl-container-demo: [HACK] most/unicens (host)  
- 12bf2345 agl-container-demo: simple container manager, lxc-launcher  
- 74829e2c agl-container-demo: support demo of rebooting container/system

} All new features and components are added here

# Sample integration how to use meta-agl-devel (cont.)

- meta-agl-devel for CES2020 container integration demo







km/h  
Thank you