Behind the scenes of AGL v3.0 "CC" and towards

Charming Chinook

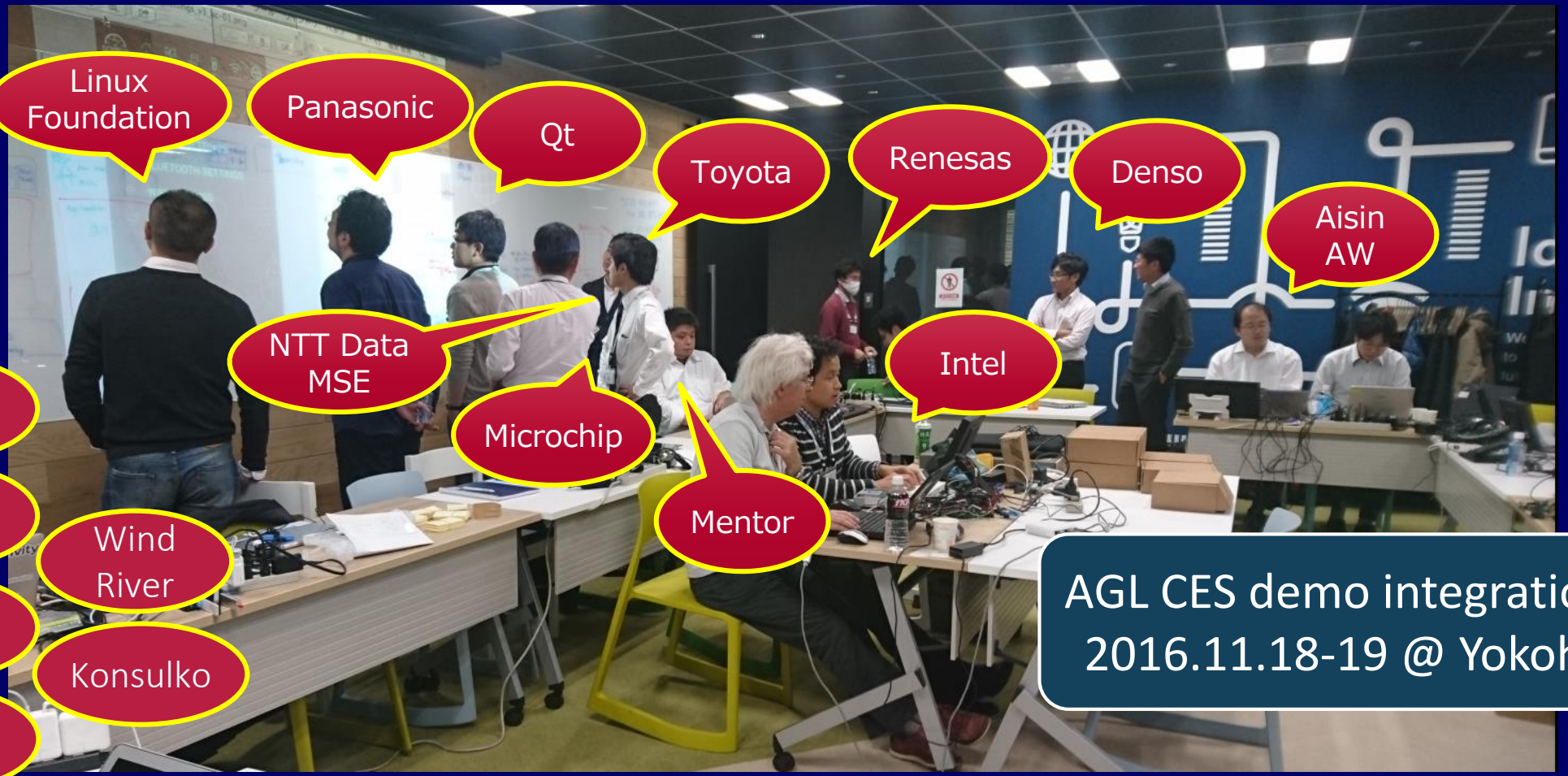
Hisao Munakata

Automotive Grade Linux, Advisory board
Senior Director, Renesas Electronics Corp

Today's topic

- Introduction of AGL v3.0 “CC” development (behind the scene)
- AGL development process review
- Proposal for AGL v4.0 “DD” and later

Essence of AGL = “working together”

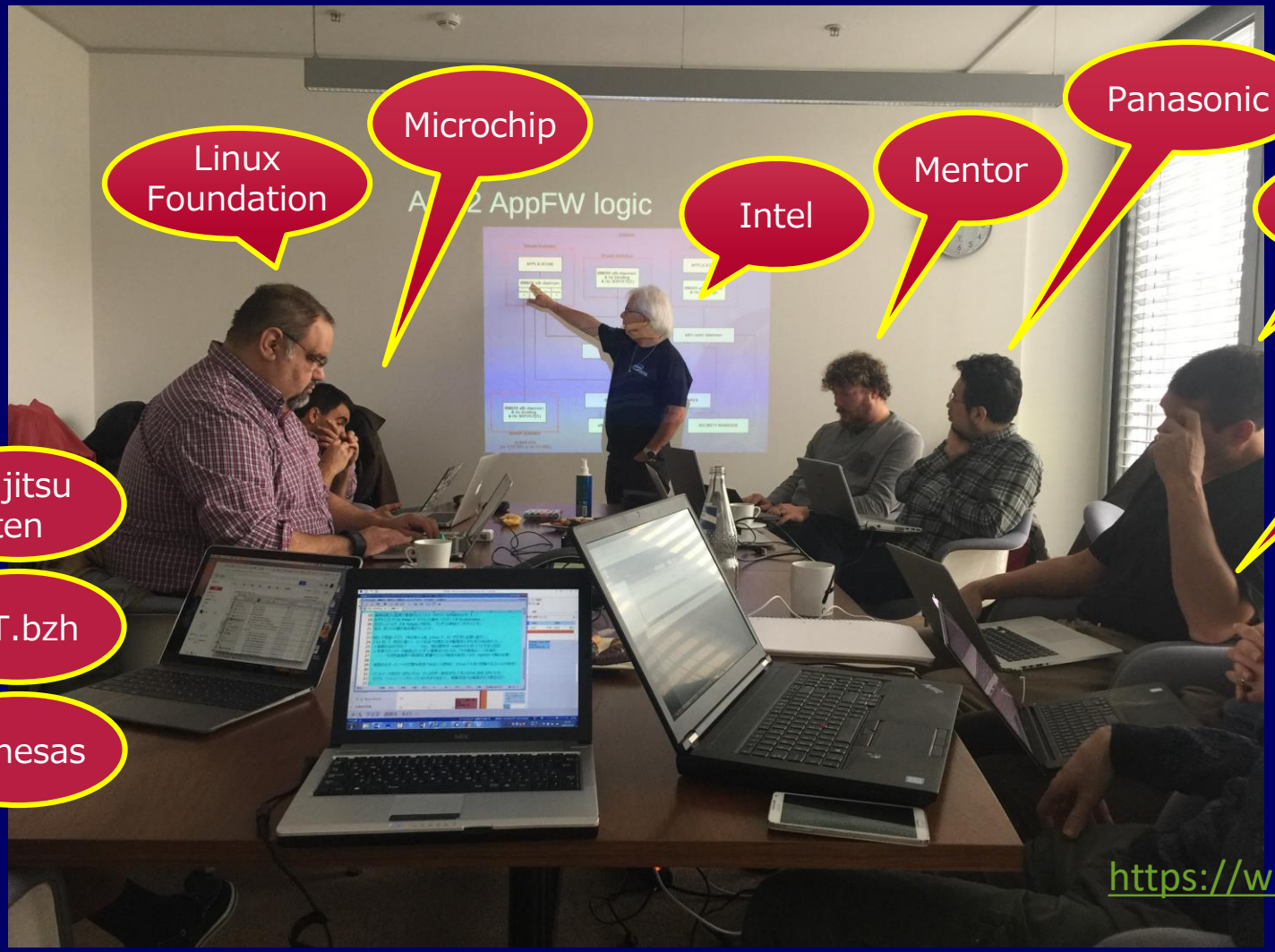


AGL CES demo integration WS
2016.11.18-19 @ Yokohama

https://wiki.automotivelinux.org/agl-distro/nov2016-f2f

Developers from various member company work together for the single target

Essence of AGL = “working together”



AGL Europe F2F meeting
2016.10.14 @ATS Berlin

https://wiki.automotivelinux.org/agl-distro/oct2016-f2f

AGL v3.0 “CC” unveiled at CES2017

Microchip

Qualcomm

NTT Data MSE

Intel

AGL master demo

+11 demo

Texas Instruments

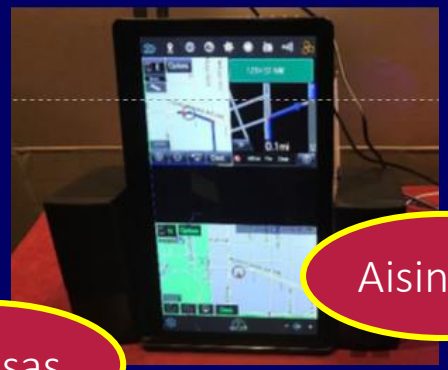
Advanced Telematics

Forge rock

Renesas

Panaonic

Denso



Aisin AW

AGL roadmap : every 6 month update

ver.	AGL 1.0 “Agile Albacore”	AGL 2.0 “Brilliant Blowfish ”	AGL 3.0 “Charming Chinook ”
release	2015.12.1	2016.7.11	2017.1.6
demo	2016 CES	2016 ALS	2017 CES

AGL v3.0 Features	● UCB updated to use Yocto Project version 2.1 (Krogoth) ● A brand new home screen and window manager ● An improved application framework and application launcher ● A new SDK for rapid application development ● Reference applications including media player, tuner, navigation, Bluetooth, WiFi, HVAC control, audio mixer, ECU connection and vehicle controls ● Wide range of hardware board support including Renesas, Qualcomm Technologies, Intel, Texas Instruments, NXP and Raspberry Pi

Behind the scene of “CC” development

Structural insight of “CC”

N = F2F work shop

= Integration & debug

number of attendee

November

December

January

2015

11-13

12

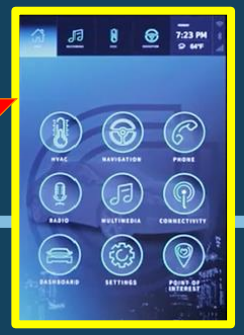
9 company

Renesas

2016

CES

App FW replaced



2016

4

14

17-18

39

25 company

new home screen

14-15

40

19 company

new demo housing

28

4-5

Qualcomm

Ti

NXP

Raspberry

Intel

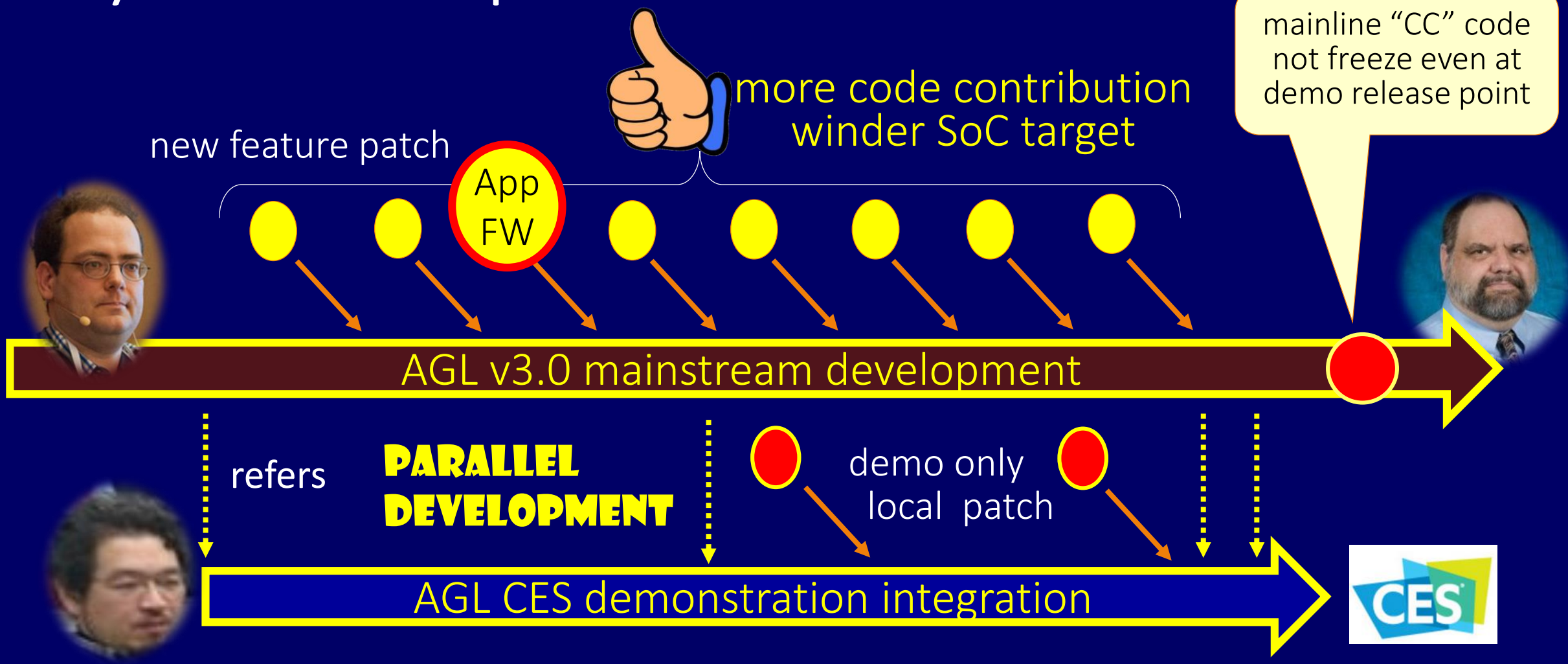
Renesas

2017

CES

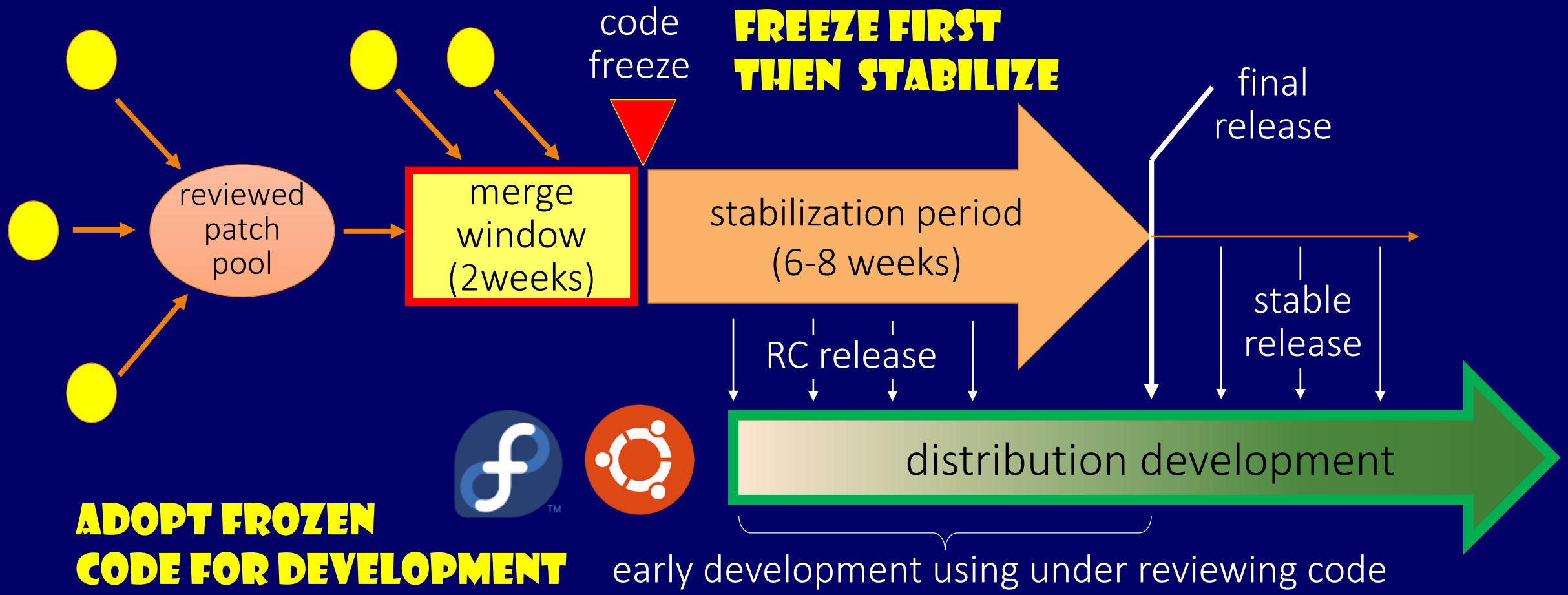
AGL v3.0 “CC” development was much more complicated work than ever

Why “CC” development confused a lot?



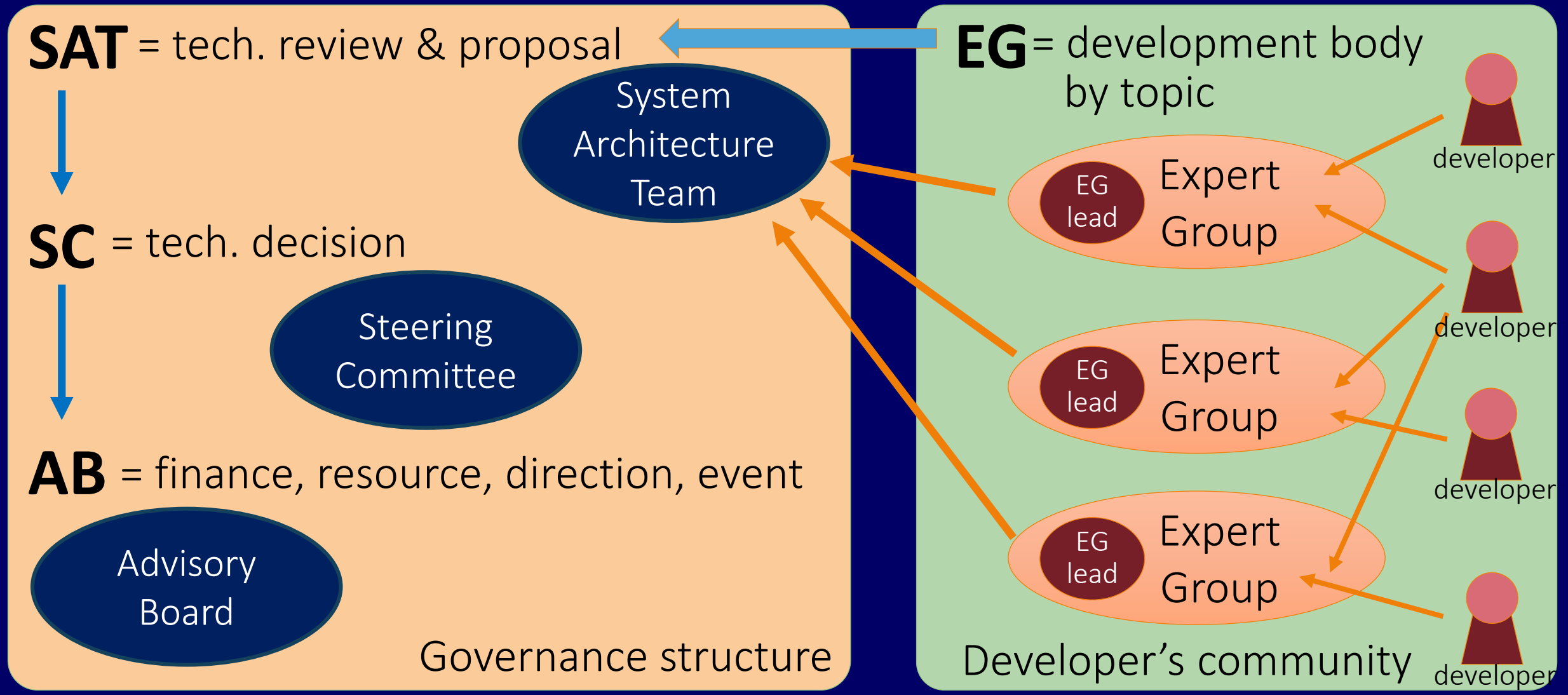
Existing AGL demo integration procedure no more works for the new scale

Proven reference : Linux kernel development model



Linux kernel update every around 65 days, following strict patch merge period

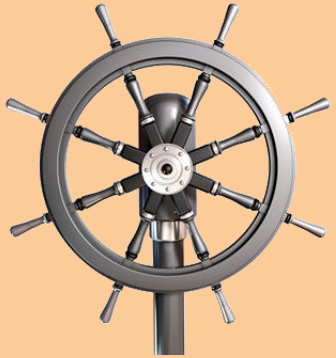
AGL “governance” and “developer community”



AGL model = transparent community style

private ML

loose control



Who Steers?

member zone

pubic ML

AGL wiki

AGL web

public zone

proposal patch

public review

gerrit

validation

Auto Test

gate keeper

checking a sanity

bug fix patch

bi-week dev/EG call

AGL master

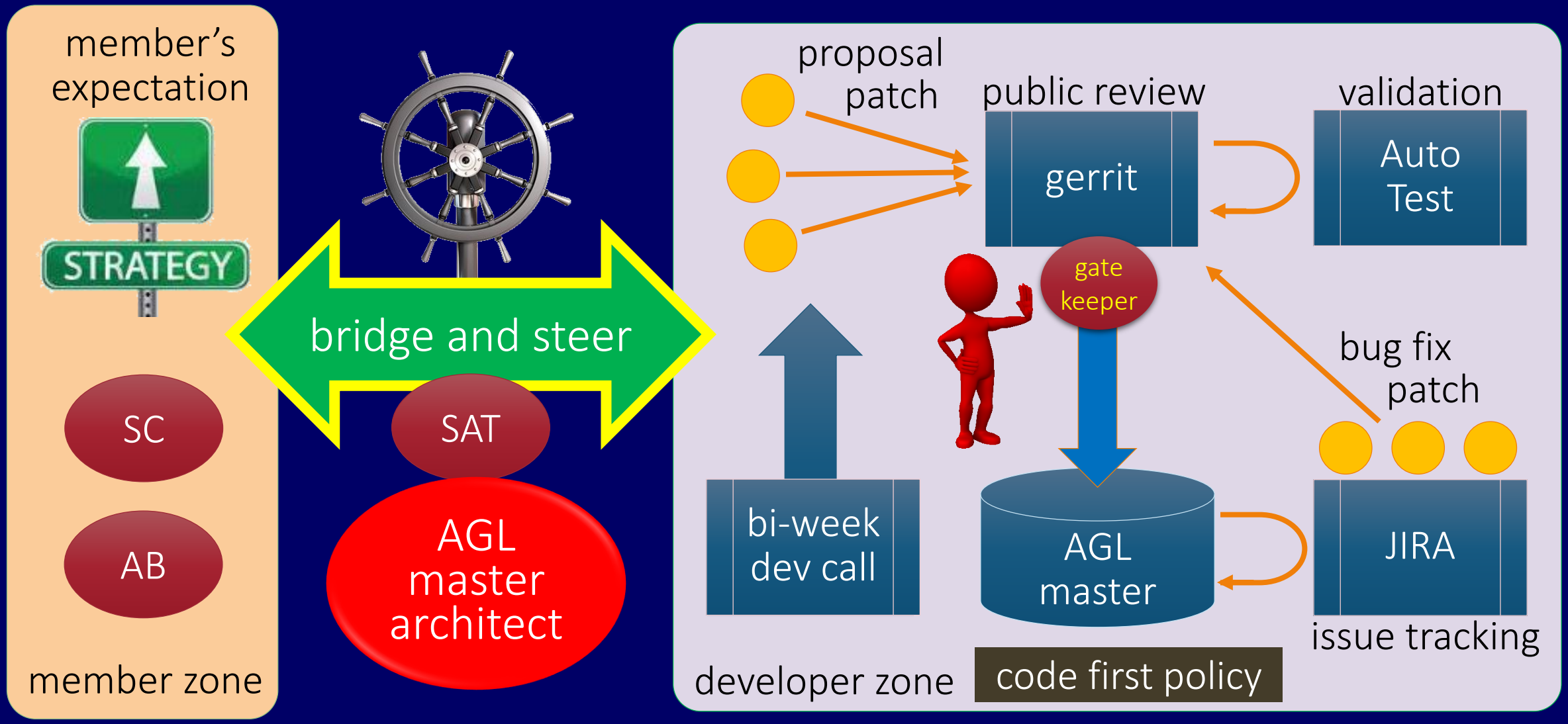
JIRA

issue tracking

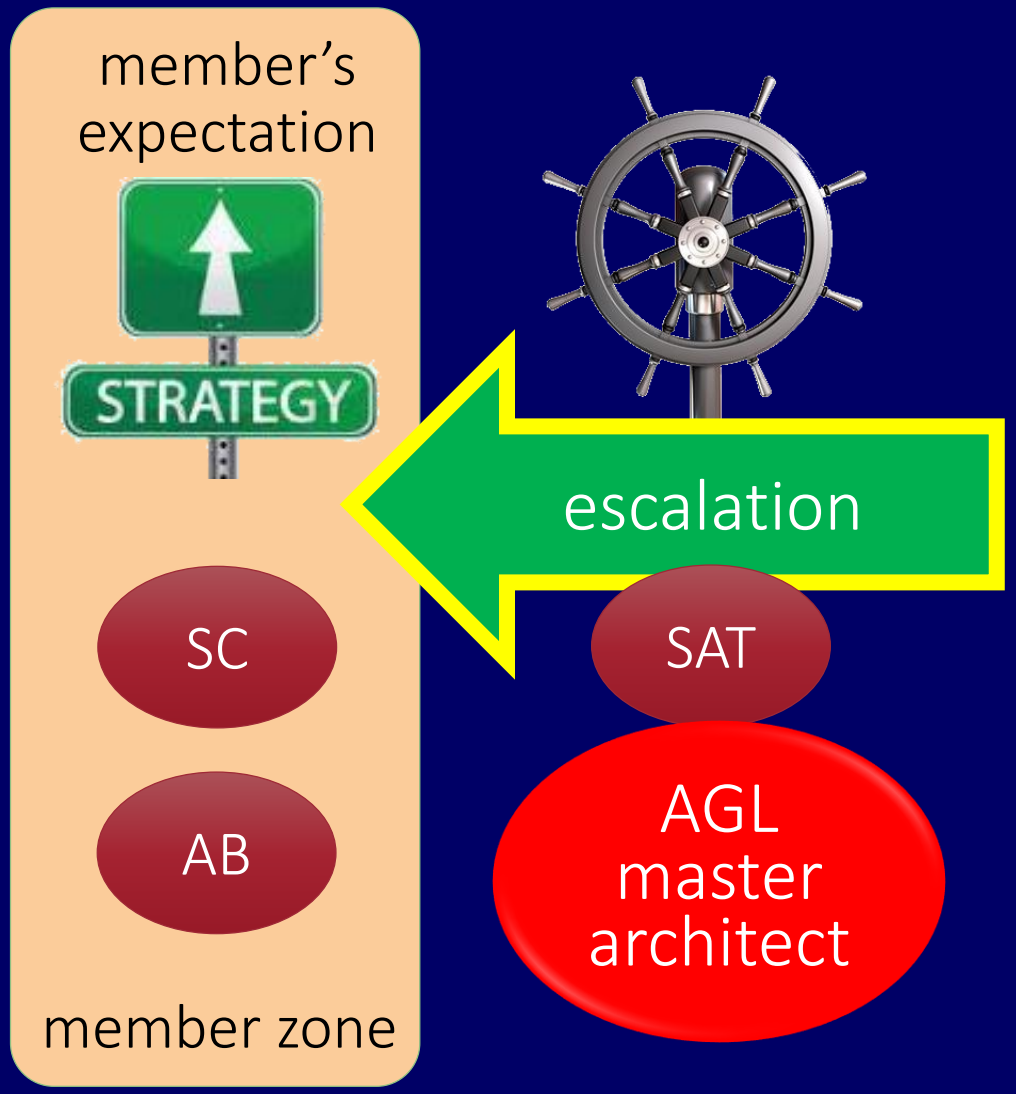
developer zone

code first policy

We need a middle man who steers the development

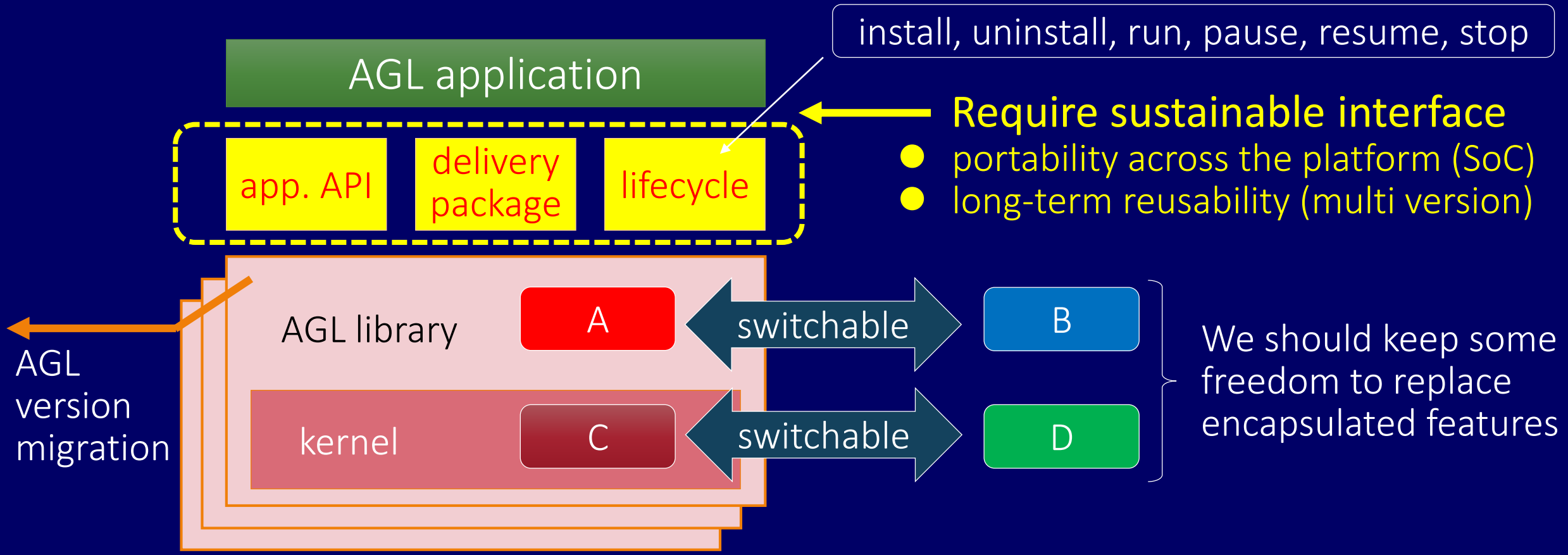


We need to settle an official review rules



- Initially, we do not want to loose a speed and passion of community style development.
- Not all topics (=patch submission) require audit by SC (& AB). Master architect has a privilege of daily code migrations.
- However, if anyone want to add completely new scheme that affect existing asset and/or future scope of AGL, it must be escalated to the SC review through SAT.
- We should have crystal clear and fair rules for the review and decision process.

AGL compliance = abstraction for application portability

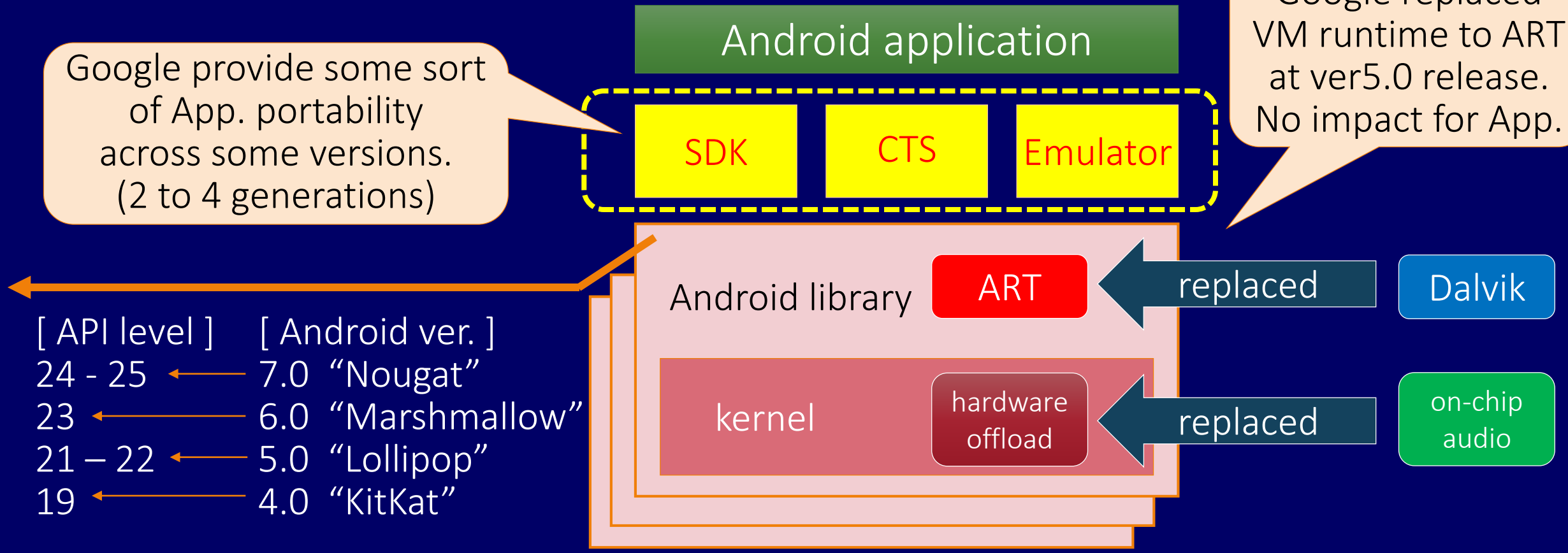


1) Solid and sustainable external API for application portability, reusability
2) Flexibility to select OS core mechanism to allow each OEM’s preference

Example : Google Android “sustainable” SDK

Google provide some sort of App. portability across some versions. (2 to 4 generations)

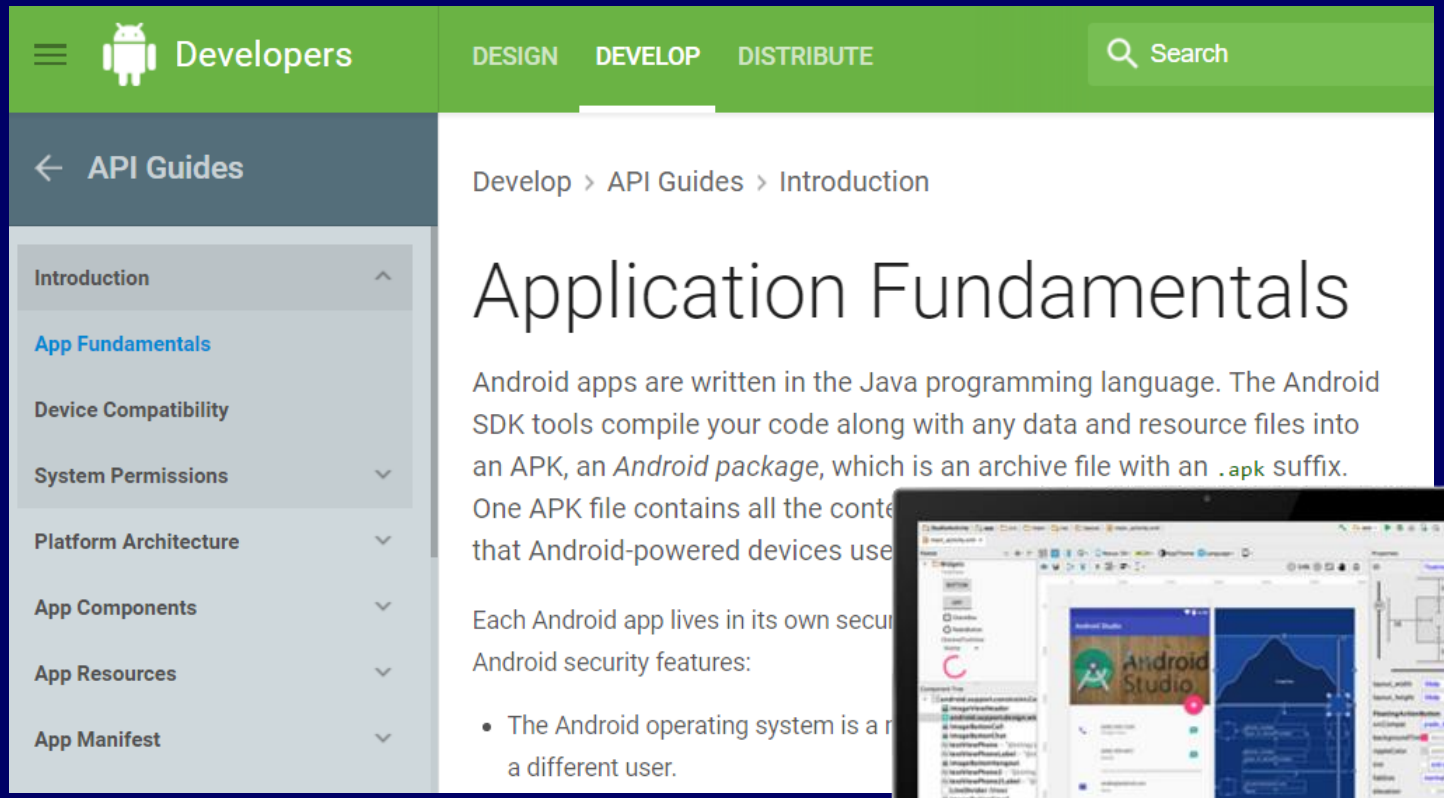
Google replaced VM runtime to ART at ver5.0 release. No impact for App.



[API level]	[Android ver.]
24 - 25	7.0 “Nougat”
23	6.0 “Marshmallow”
21 – 22	5.0 “Lollipop”
19	4.0 “KitKat”

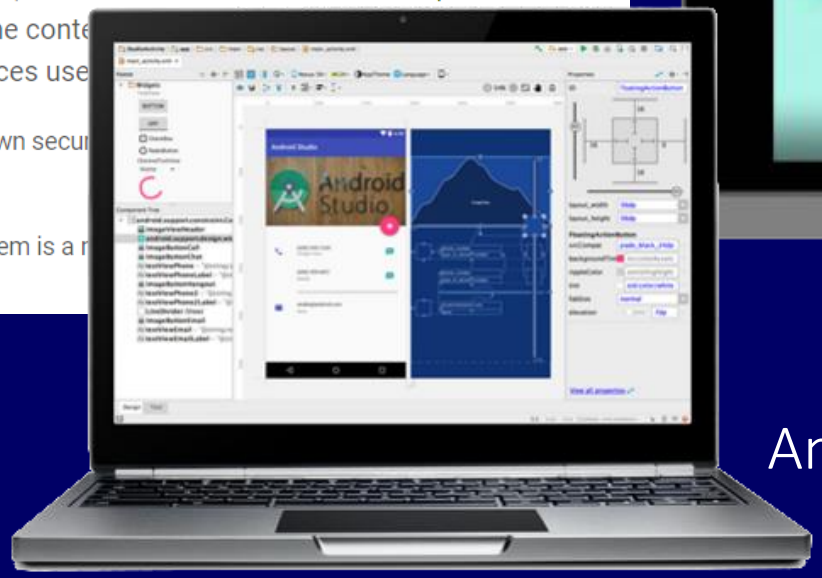
Definitely we need more long-term, but now Android offers better option

Example 2 : Android application developer support



on-line document for Android app. developer

emulator runs on PC



Android studio (IDE)

AGL kick start document : in progress

http://docs.iot.bzh/docs/getting_started/en/dev/reference/machines/R-Car-Starter-Kit-gen3.html

IOT BZH Getting Started Architecture Guides Hardware Support Developer Guides APIs & Services

TechPost Contribute Search 'dev' docs...

Build for R Car Starter Kit gen3

board

- Building the AGL Demo Platform for R-Car Starter Kit Gen3
- Setting up the build environment:
- Build your image:
- Booting AGL Image on R-Car Starter Kit Gen3 boards
- Prepare
- Bootin
- Serial Co
- Install a serial

Draft version of documentation under development may change!

AGL Kickstart on Renesas R-Car Starter Kit Gen3 (h3ulcb, m3ulcb)

re is a non exhaustive list of hardware parts that could be u
rter Kit Gen3 board development environment:

- Starter Kit Gen3 board with its power supply
- micro USB-A cable for serial console
- USB 3.0 Hub

Most of existing AGL documents, wiki are for BSP developer. (Not for app.)

Conclusion

- AGL community is growing day by day, and existing development model has not been functioning as well as it used to. Due to this, AGL “CC” development was a bit messy; even its result was great, much more than expected.

- We need to improve our process and rules to comply with current AGL circumstances. Demo development need to be isolated from upstream development and allocate dedicated maintainer for that.

- Community style flexible collaboration and “code first” policy are the beauty of AGL, and we should retain them. At the same time, we need to improve governance like compliance and technology selection. TBD at SC/AB level.