

Why AGL should collaborate with the community and how?

Aiming **long-term reusable common asset** for automotive industry

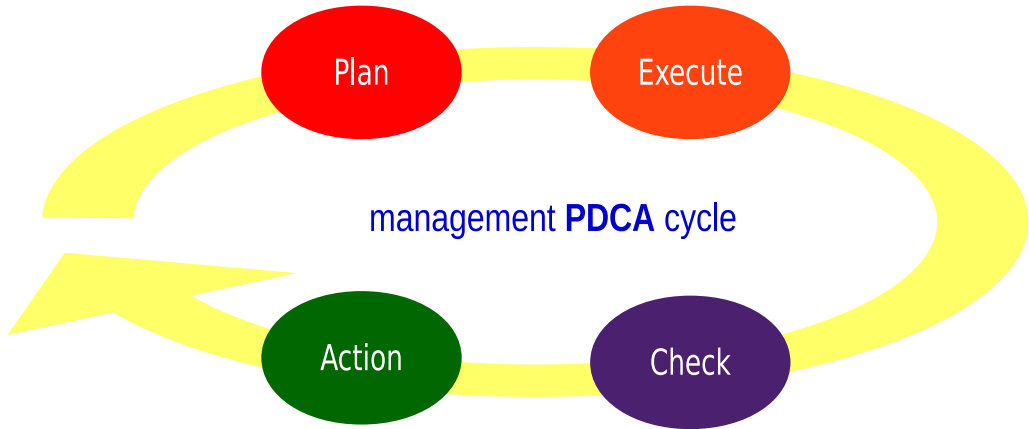
Hisao Munakata

Linux Foundation, Automotive Grade Linux

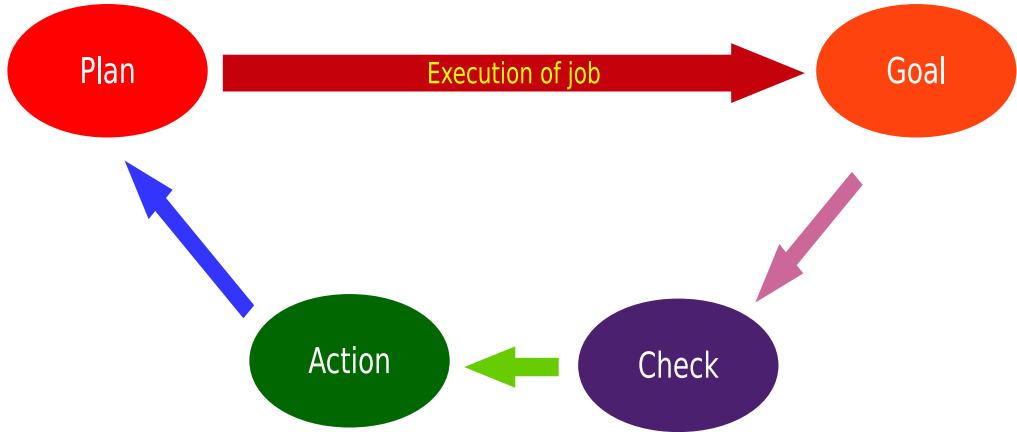
February 25th 2016

How we deal with the issue

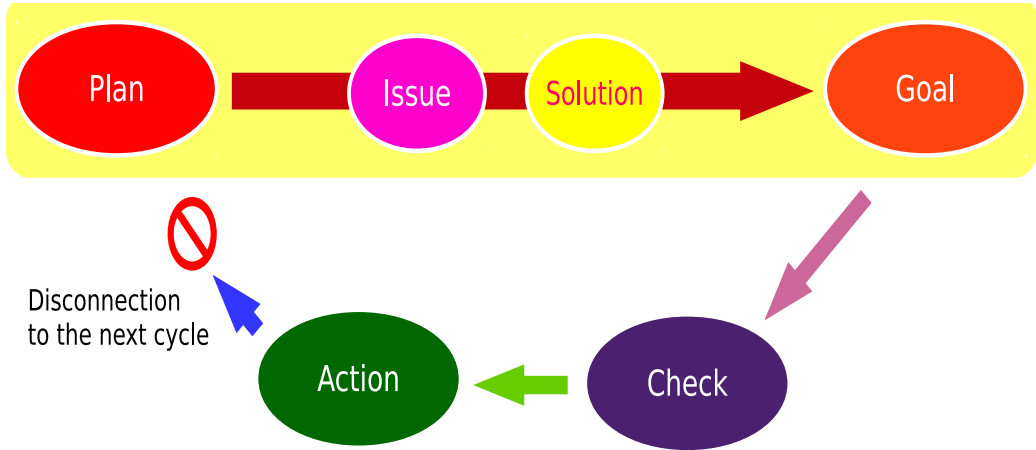
PDCA is theoretical process management procedure, **but..**



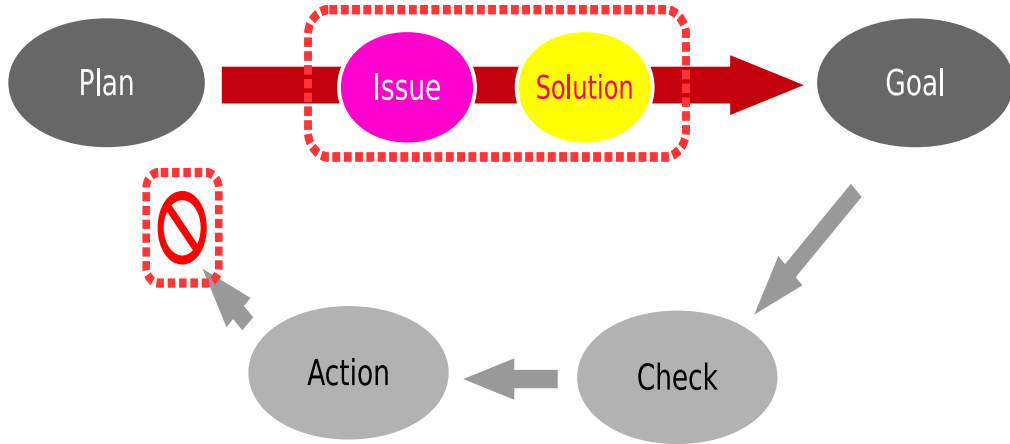
In real life, **EXECUTION** consumes most of the time



While EXECUTION, various **ISSUES** hit you and require **SOLUTION**



Today I focus on this ISSUES and SOLUTION



Case 1 : Stuck at the train station in the snow morning

ISSUES

- You want to reach office by train as usual
- As forecast predicted, slight snowfalls
- Then you notice **commute train comes only every 30 min.** rather than regular 5 min.



Case 1 : Stuck at the train station in the snow morning

ISSUES

- You want to reach office by train as usual
- As forecast predicted, slight snowfalls
- Then you notice **commute train comes only every 30 min.** rather than regular 5 min.

SOLUTION

- stack at the station
- go back to the home
- seek for the alternative train route
- use taxi or walk to the office



Case 2 : 24bit/192k hi-resolution audio playback on WindowsPC

ISSUES

- You want play 24bit/192k hi-reso contents
- You purchase 32bit/384k support USB-DAC
- Then you notice **WindowsOS does not contain required USB Audio Class2.0** feature



USB Audio Class 2.0 compatible DAC

- M2TECH HiFace DAC
- up to 32bit / 384k support
- **asynchronous 2.0 Audio Class USB**

Case 2 : 24bit/192k hi-resolution audio playback on WindowsPC

ISSUES

- You want play 24bit/192k hi-reso contents
- You purchase 32bit/384k support USB-DAC
- Then you notice **WindowsOS does not contain required USB Audio Class2.0** feature

SOLUTION

- return purchased USB-DAC
- seek for the proprietary driver
- replace to Windows compatible USB DAC
- use as USB Audio 1.0 compatible DAC



USB Audio Class 2.0 compatible DAC

- M2TECH HiFace DAC
- up to 32bit / 384k support
- **asynchronous 2.0 Audio Class USB**

Available **SOLUTION** options for the **ISSUE**



Available **SOLUTION** options for the **ISSUE**



Available **SOLUTION** options for the **ISSUE**



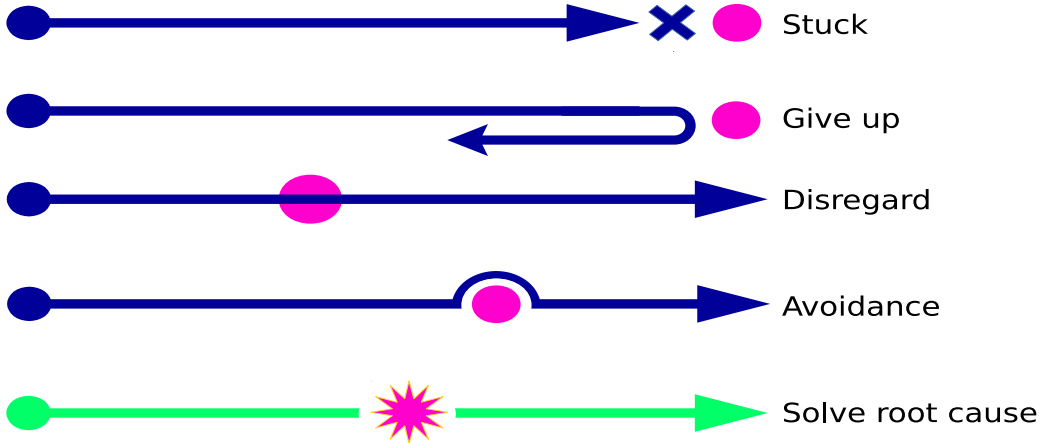
Available **SOLUTION** options for the **ISSUE**








Available **SOLUTION** options for the **ISSUE**








Available **SOLUTION** options for the **ISSUE**



Categorization of Case1/2 SOLUTION

Case1 (snow train)	Reaction pattern	Case2 (USB DAC)
Stack at the station		Give up to use USB DAC
Back to the home		Return to the shop
		Use as low spec DAC
		
Seek alternative train route use taxi, or walk		Seek for proprietary driver

Categorization of Case1/2 SOLUTION

Case1 (snow train)	Reaction pattern	Case2 (USB DAC)
Stack at the station		Give up to use USB DAC
Back to the home		Return to the shop
		Use as low spec DAC
		
Seek alternative train route use taxi, or walk		Seek for proprietary driver

Avoidance might be the best possible option you can choose for case1/2

Why AVOIDANCE does not work for long-term solution

Avoidance approach works only for **one time solution**



Avoidance approach works only for **one time solution**



As original issue remains, you may hit exact same problem in the future

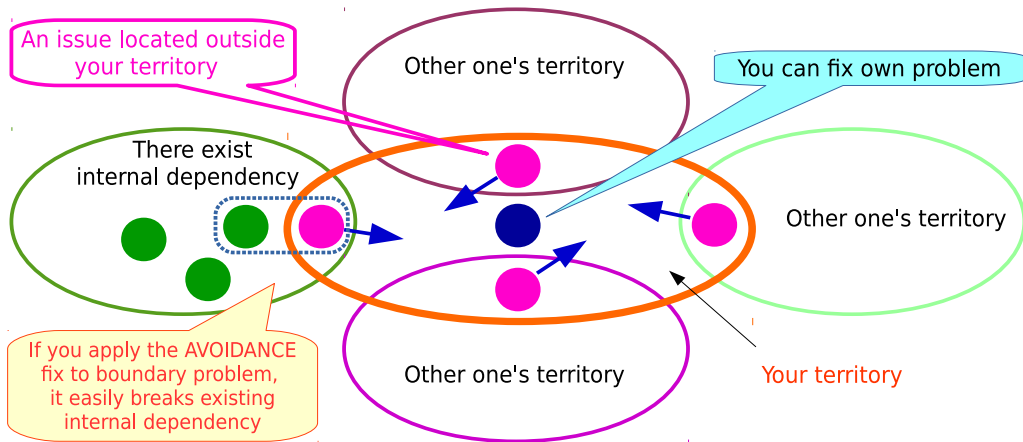
Case 1 was a human disaster, not a simple natural disaster

- Train crash accident occurred on a snow day (2014-2-15)
- Train break system did not work properly due to the snow
- Train operator wanted to avoid the similar accident
- Then, **intentionally added excessive operation interval**



Tokyu train snow slip & crash accident

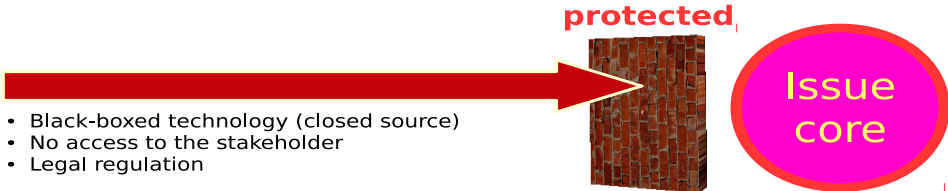
Avoidance structure : pull other one's issue to own place, then fix



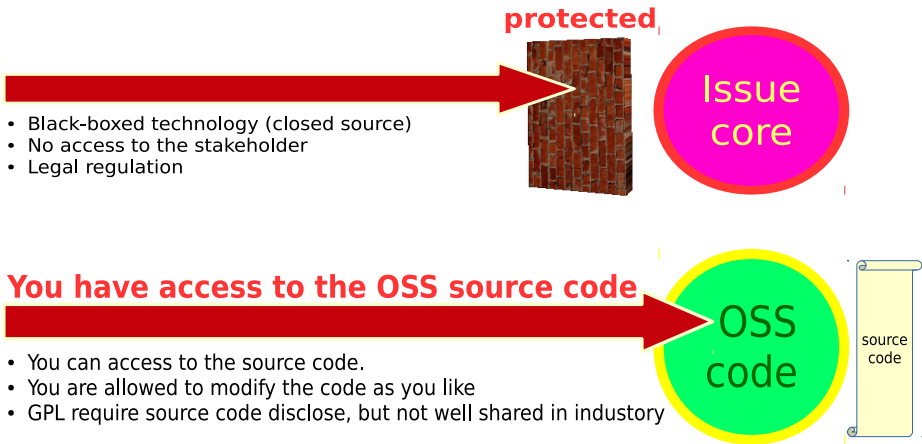
Once goal achieved, **you really do not care for the root cause fix**

In general, you have no (or very limited) access to the root cause

- Proprietary system (no source code access, closed system)
- Operated as large scale infrastructure (transportation, electricity, gas,...)
- Restricted by the regulation
- A natural disaster



Avoidance for OSS easily create fragmentation (negative aspect)



GPL allow code modification, redistribution as you like, **however..**

The Open Source Definition (Annotated)

3. Derived Works

The license must **allow modifications and derived works**, and must allow them to be distributed under the same terms as the license of the original software.

<https://opensource.org/osd-annotated>

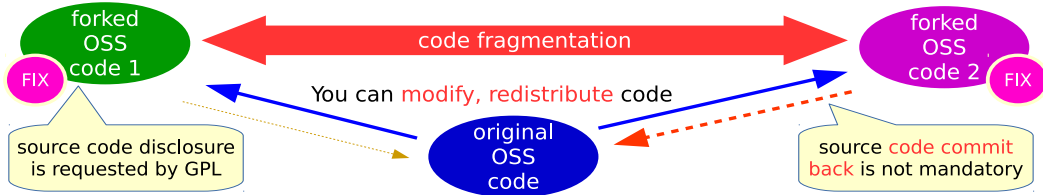
GPL allow code modification, redistribution as you like, **however..**

The Open Source Definition (Annotated)

3. Derived Works

The license must **allow modifications and derived works**, and must allow them to be distributed under the same terms as the license of the original software.

<https://opensource.org/osd-annotated>



Case 3 : Add new sound device configuration to your Linux BSP

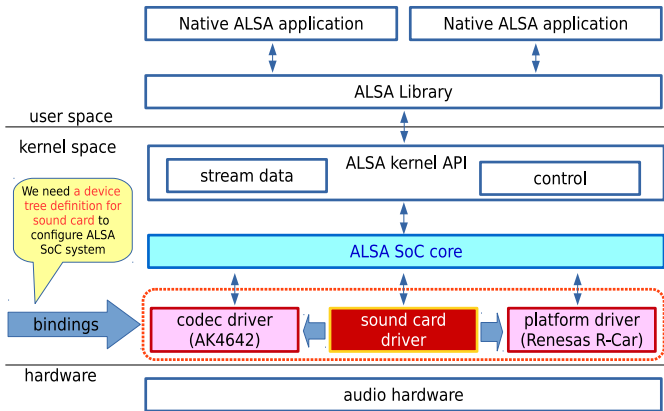
ISSUE

- You develop Linux BSP
- plan to use AK4642 codec
- Then, noticed **sound-card config needed for ALSA-SoC**

Case 3 : Add new sound device configuration to your Linux BSP

ISSUE

- You develop Linux BSP
- plan to use AK4642 codec
- Then, noticed **sound-card config needed for ALSA-SoC**



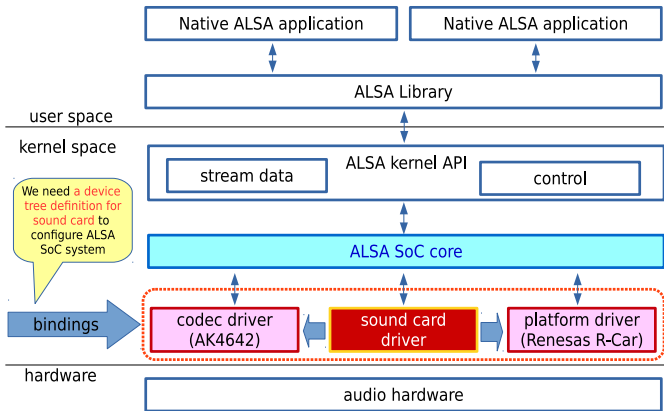
Case 3 : Add new sound device configuration to your Linux BSP

ISSUE

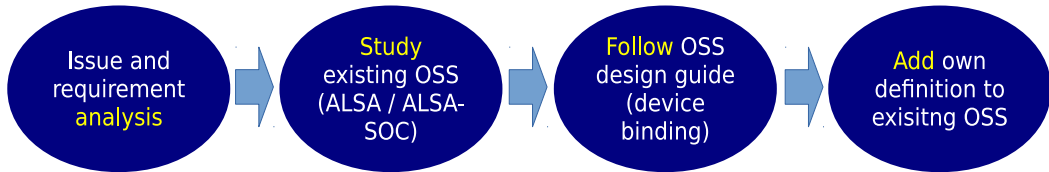
- You develop Linux BSP
- plan to use AK4642 codec
- Then, noticed **sound-card config needed for ALSA-SoC**

SOLUTION

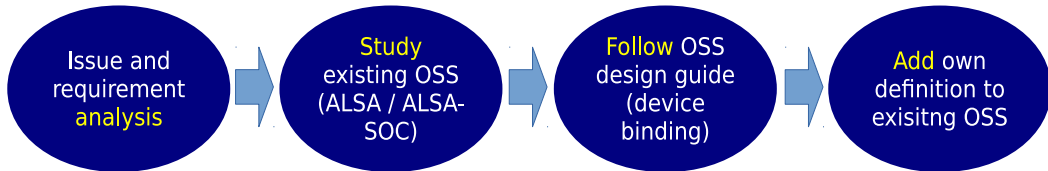
- Follow ALSA/ALSA-SoC
- Write “simple-card” config
- AK4642 start working
- **Submit file to the upstream**



Case 3 assessment : Appropriate OSS utilization and contribution



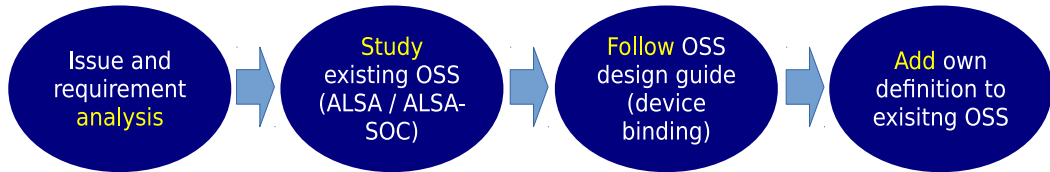
Case 3 assessment : Appropriate OSS utilization and contribution



positive (your achievement)

- Follow standards (ALSA, ALSA-SoC)
- Own issue fixed (AK4642 works)
- Submit code to the upstream

Case 3 assessment : Appropriate OSS utilization and contribution



positive (your achievement)

- Follow standards (ALSA, ALSA-SoC)
- Own issue fixed (AK4642 works)
- Submit code to the upstream

potentially negative (for the community)

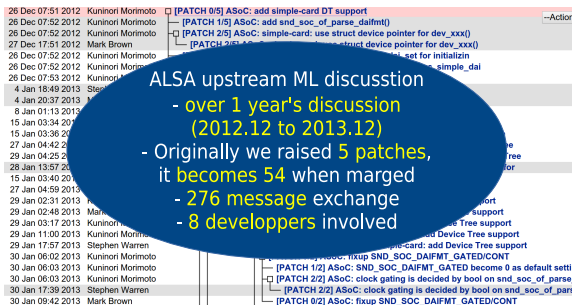
- Single purpose (**not sharable**) code
- Cause similar config **code flooding**
- Increase code complexity (**diffusion**)

Your code submission to the community might cause further confusion

Reality : simple-card patch raised big argument in the community

Feedback from the ALSA community

- Seek for the best practice
- Add reasonable device abstraction
- Write sharable code
- Do not flood the config definition
- Care for long-term maintenance
- If needed request core code enhance

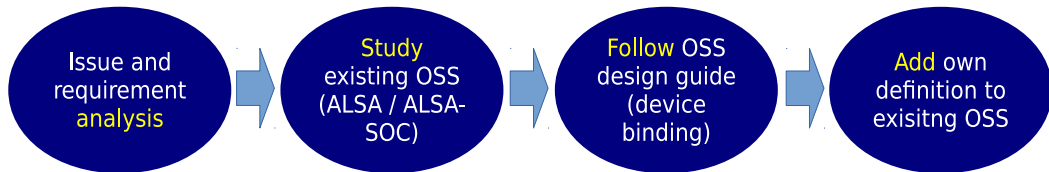


Date	Time	From	Subject
26 Dec	07:51 2012	Kuninori Morimoto	[PATCH 0/5] ASoC: add simple-card DT support
26 Dec	07:52 2012	Kuninori Morimoto	[PATCH 1/5] ASoC: add snd_soc_of_parse_daifmt()
26 Dec	07:52 2012	Kuninori Morimoto	[PATCH 2/5] ASoC: simple-card: use struct device pointer for dev_xxxx()
27 Dec	17:51 2012	Mark Brown	[PATCH 2/5] ASoC: simple-card: use struct device pointer for dev_xxxx()
26 Dec	07:52 2012	Kuninori Morimoto	[PATCH 3/5] ASoC: simple-card: use struct device pointer for dev_xxxx()
26 Dec	07:52 2012	Kuninori Morimoto	[PATCH 4/5] ASoC: simple-card: use struct device pointer for dev_xxxx()
26 Dec	07:53 2012	Kuninori Morimoto	[PATCH 5/5] ASoC: simple-card: use struct device pointer for dev_xxxx()
4 Jan	18:49 2013	Stephen Warren	Re: [PATCH 0/5] ASoC: add simple-card DT support
4 Jan	20:37 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
8 Jan	01:13 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
15 Jan	03:34 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
15 Jan	03:36 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
27 Jan	04:42 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
29 Jan	04:25 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
28 Jan	13:57 2013	Stephen Warren	Re: [PATCH 0/5] ASoC: add simple-card DT support
15 Jan	03:40 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
27 Jan	04:59 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
29 Jan	02:31 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
29 Jan	02:48 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support
29 Jan	03:17 2013	Kuninori Morimoto	Re: [PATCH 0/5] ASoC: add simple-card DT support
29 Jan	11:00 2013	Kuninori Morimoto	Re: [PATCH 0/5] ASoC: add simple-card DT support
29 Jan	17:57 2013	Stephen Warren	Re: [PATCH 0/5] ASoC: add simple-card DT support
30 Jan	06:02 2013	Kuninori Morimoto	Re: [PATCH 0/5] ASoC: add simple-card DT support
30 Jan	06:03 2013	Kuninori Morimoto	Re: [PATCH 0/5] ASoC: add simple-card DT support
30 Jan	06:03 2013	Kuninori Morimoto	Re: [PATCH 0/5] ASoC: add simple-card DT support
30 Jan	17:39 2013	Stephen Warren	Re: [PATCH 0/5] ASoC: add simple-card DT support
30 Jan	09:42 2013	Mark Brown	Re: [PATCH 0/5] ASoC: add simple-card DT support

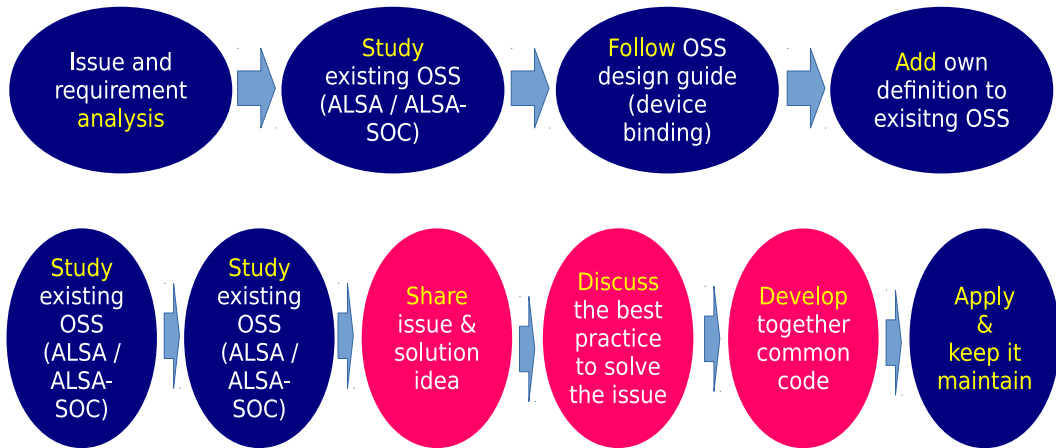
<http://thread.gmane.org/gmane.linux.alsa.devel/104057>

We strived to develop reusable, common ALSA SoC binding framework

Community expect more than patch submission, but coordination



Community expect more than patch submission, but coordination

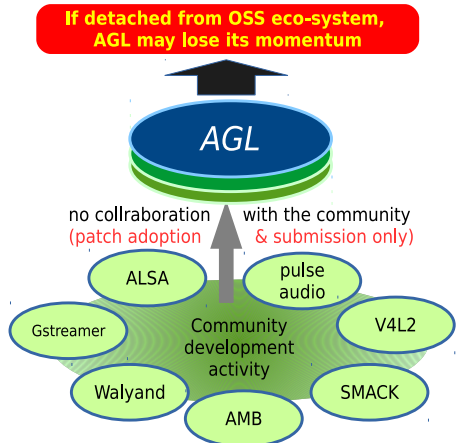


Why AGL need to collaborate with the community and how?

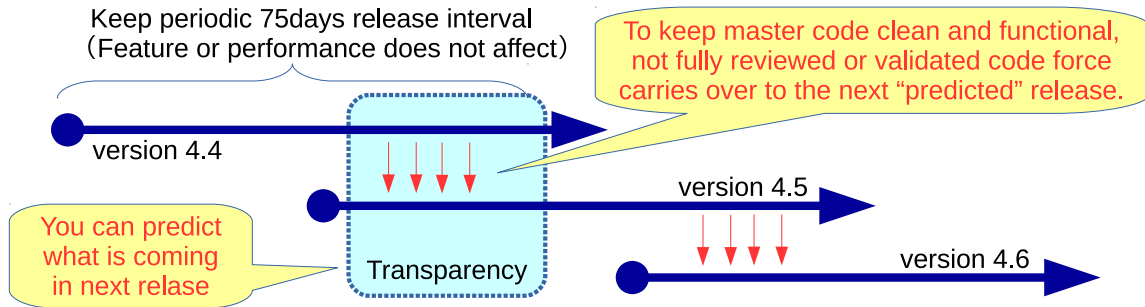
Not let the AGL to detached from the OSS mainstream momentum

Challenge

- collect automotive domain specific demands
- investigate existing OSS code before start writing own code (=AVOIDANCE approach)
- find and establish a good relation with existing reference OSS project
- collaborate with existing project and enhance code
- commit long-term maintenance for domain specific code
- create software CEO-system to encourage people to write an application code

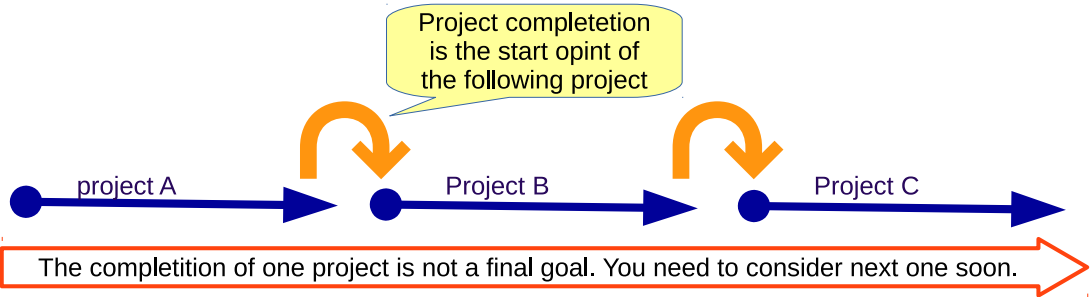


OSS development methodology = continuous integration (CI)



As you can predict mod, long-term release plan, you can concentrate essential issue fix.

Aiming to develop 3S (Sustainable, Sharable and Safe) asset



OSS CI enables long-term reusable / sustainable asset creation

5C effort enables long-term reusable asset creation

5C effort for long-term sustainable solution

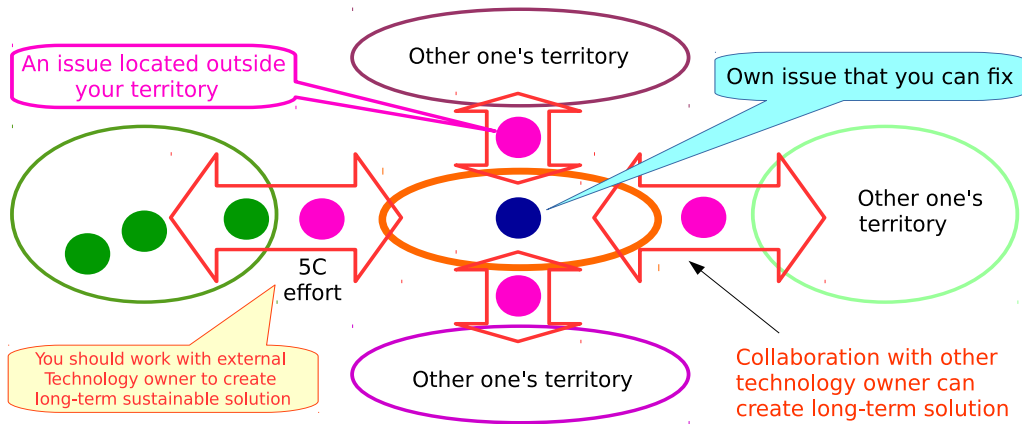
- 1 **C**onnection
- 2 **C**ommunication
- 3 **C**ollaboration
- 4 **C**oordination
- 5 **C**ontenuous

5C effort proof in many existing OSS project



fosdem 2016 restaurant place

structure of essential fix effort



conclusion

Conclusion

- An **avoidance approach** is a common way of development as well as everyday life. However, it **works only for a short-term solution**.
- The big advantage of OSS (Open Source Software) adoption is **3S (Sustainable, Sharable and Safe)** solution. It requires **5C (Connection, Communication, Collaboration, coordination and continuous)** effort.
- AGL need to care for 5C effort before start writing own code (is Avoidance). Without such effort, AGL will detach from OSS community and lose momentums. We need to recognize **community connection is key success factor of AGL success**.

