(LHC) Controls – an OP perspective

K. Fuchsberger

on behalf of the LHC-OP team

Many, many thanks to the whole OP team for the input!

OP Perspective on Controls

Content

- Users view
- Developers View
- Summary

Content

First of all ...

We have a very good control system!!!!

- Very stable
- All operational scenarios managable (e.g. Ramp & squeeze)
- Big cleanup and a lot of improvements done of APIs during LS1.

So, ... Do we have to change at all?

- Avoiding Mistakes → Increase Availability (cf Andreas Presentation)
- Evolve (carefully!) → Be prepared!
  - The world around us changes (e.g. processors, java, …)
  - E.g. Future accelerators
- Be Compatible with the 21st Century ;-)

What we did not ask for ...

Breaking API changes

- LSA Refactoring
- Logging Service API (x2)
- Changes in many FESA classes

We understand that such changes might be necessary..

Transparency & Communication (in advance)

→ See Marines Presentation

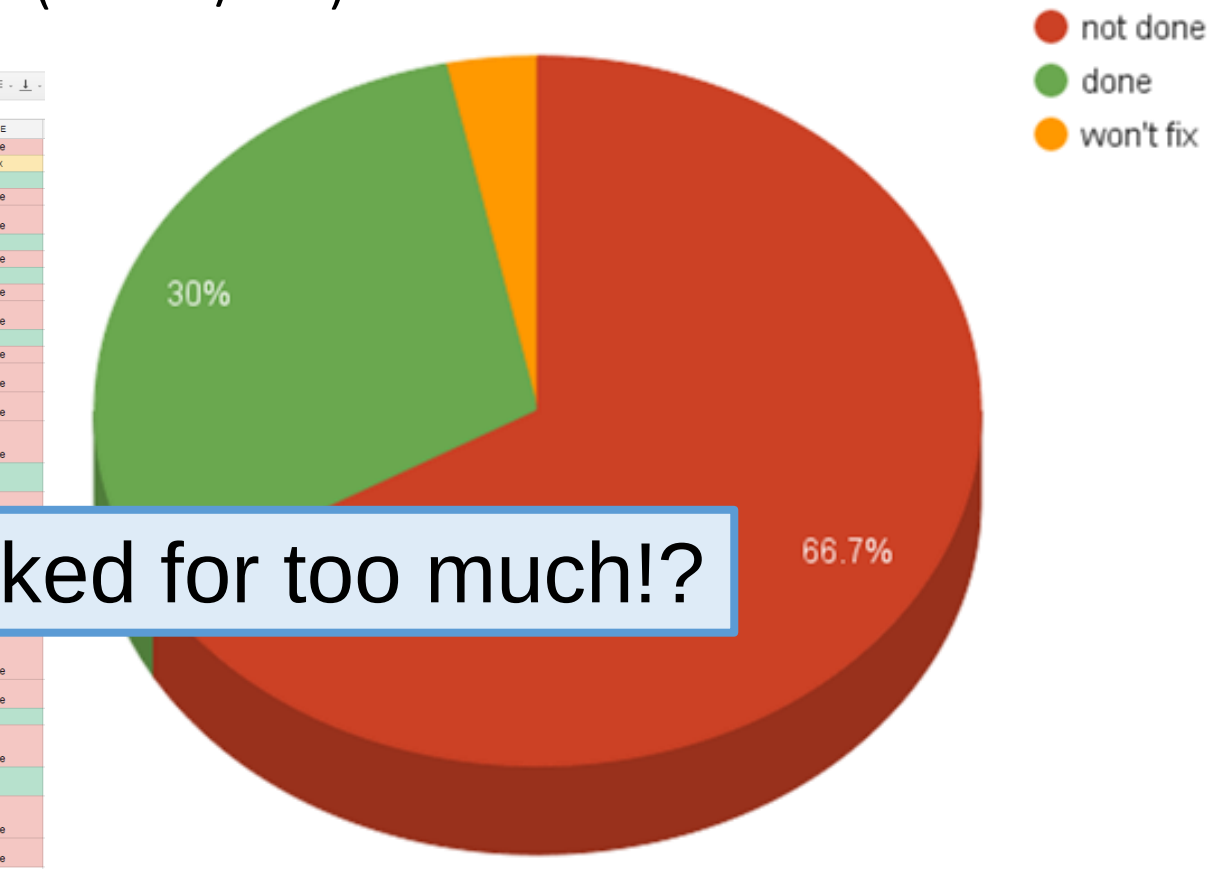
What we asked for (2010/11)…

old-requests

File Edit View Insert Format Data Tools Add-ons Help All changes saved in Drive

Name

	A	B	C	D	E
4	IQC does not latch	normal conditions should be green	requested	not done	not done
5	Sequencer Checklist panel		requested	won't fix	won't fix
6	Sequencer Parameter		requested	not done	done
7	Sequencer parallel subsequences		requested	not done	not done
8	State Machine influences workflow		requested	not done	not done
9	Lsa compare settings		requested		done
10	Easy Lsa rollback trims	e.g. all parameters of same trim	requested	not done	not done
11	Lsa history	of driven params and resident BPs	requested		done
12		history of resident BP	requested	not done	not done
13	Lsa incorporation improvement	More flexible (e.g. one rule per BP type not enough, snapback incorporation)	requested	not done	not done
14	Knob application		requested		done
15	Hypercycle change	clear procedure and sequence	requested	not done	not done
16	Alarm Configuration	No Alarm when everything ok, mode dependance	requested	not done	not done
17	Diamon Configuration	See at one glance when a server has a problem	requested	not done	not done
18	Clear information of the hierarchy between application, middleteears, proxy, frontends		requested	not done	not done
19	Sequence editor improvements	Task copy, cut paste/ Subsequences, change tracking	requested		done
20	Injection Interlocks	Summary display with all the involved system[illegible] come?			
21	Fixed Displays	Fixed d[illegible]			
22	Software releases	Well te[illegible]			
23	Timing/sequences in injectors	sequen[illegible] time, 3[illegible]			
24	Fidel fixed display				
25	State Machine flexibility	Disting[illegible] require[illegible] help efficiency; "After LS1 forcing a state should be exceptional, even in MDs"		requested	not done
26	FESA	Not easy to know on which version a device is running on; review Navigator.		requested	not done
27	Fill by fill data analysis			requested	done
28	Data storage	b/b data; avoid ad-hoc storage solution; Common System to log large amount of data.		requested	not done
29	Fidel hypercycle change	hypercycle change after fidel started -> precycle needed		requested	done
30	Console manager	more user friendly tools to edit the menues, automatic periodic refresh of menu configuration		requested	not done
31	MCS problem	Regeneration needs expert signature, complicated workaround	requested	not done	not done



→ We asked for too much!?

https://drive.google.com/open?id=1AOXREGR6X7VmoizAldJFS6gPCtACeTP2Mmo6P-o9AUA

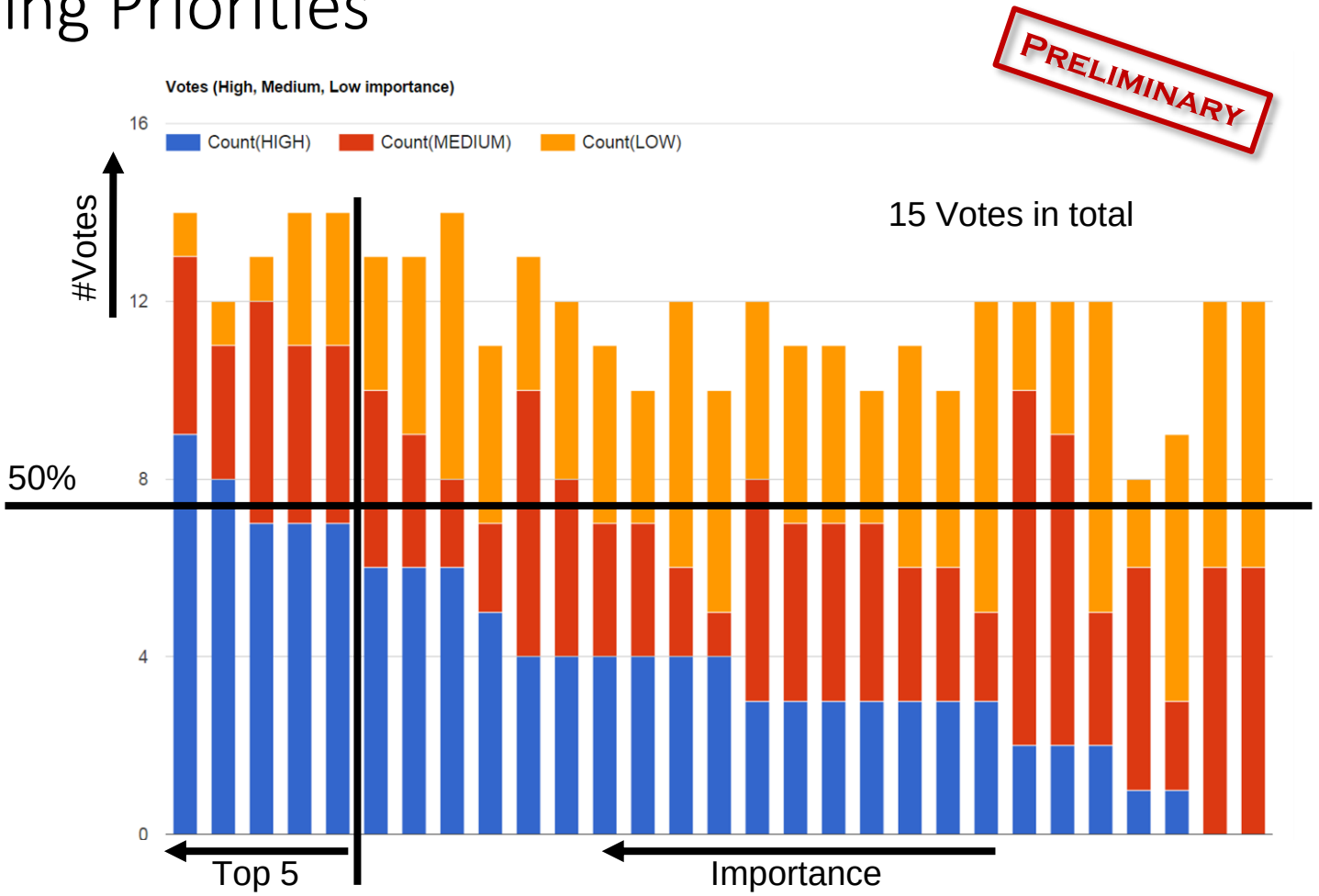
What does OP want?

What do you consider as the …

- … most important features/properties/components that an ideal control system should have?
  (Long term)
- … most important improvements that should be made to the available tools?
- … most important issues that should have been fixed as soon as possible?

→ Not easy to find out what we want ☹

Finding Priorities



Top 5 of LHC-OP‘s most wanted improvements

PRELIMINARY

5 – Improve Window Management on Consoles

SINCE 2011

- Rationalize the space
- Many overlapping windows
- ‘Perspectives’ per activity: e.g. Injection, Ramp, Lossmaps, Powering
- Beam mode / State machine deciding what gets displayed and where?
- Different window manager?
- Autostart of predefined set of applications on consoles (e.g. after a reboot)
- Shortcuts (e.g. new logbook entry, search for an application)

4 - Easily usable tool to move collimators

- Currently: complicated acrobatics of making some BP resident plus pressing the right combinations of buttons in equip state.
- Certain Scenarios
  - Move all to parking
  - Symmetrize TCTs
- Nice display where to see the collimators (probably w.r.t. the beam)

3 – Improved Filling Diagnostics

- A lot of time lost
- See at one glance why the beam did not come (Missing Fixed display)
- E.g. Why did the CBCM reject the request?
- Show in advance what has to be done before to request an injection.

- → Availability (Delphine)

2 – QPS/PIC/Equip State

- PIC permit loss-> immediately display root cause
- PM knows it - can this be faster?
- Replacement for QPS macros
- Better integration in other controls system
  - Simple overview
  - Dig down in case of problems
  - Easy tool for Circuit (+QPS) resets

→ See also: Mirkos presentation

1 – Improved Automation/Sequencer/Scripting

Execution:

- (Automated) Parallelism
- Better overview (several sequences running)
- Small things (e.g. Quirky Windows behaviour, Better error presentation)

Editing:

- Long time/chain from idea to operation
  (every change in a task needs a sequencer release)
- Tools that make it easier to refactor between different layers (Sequence <-> Task).

Scripting (More flexible (and more dynamic) way to create scripts/macros/sequences). E.g. for

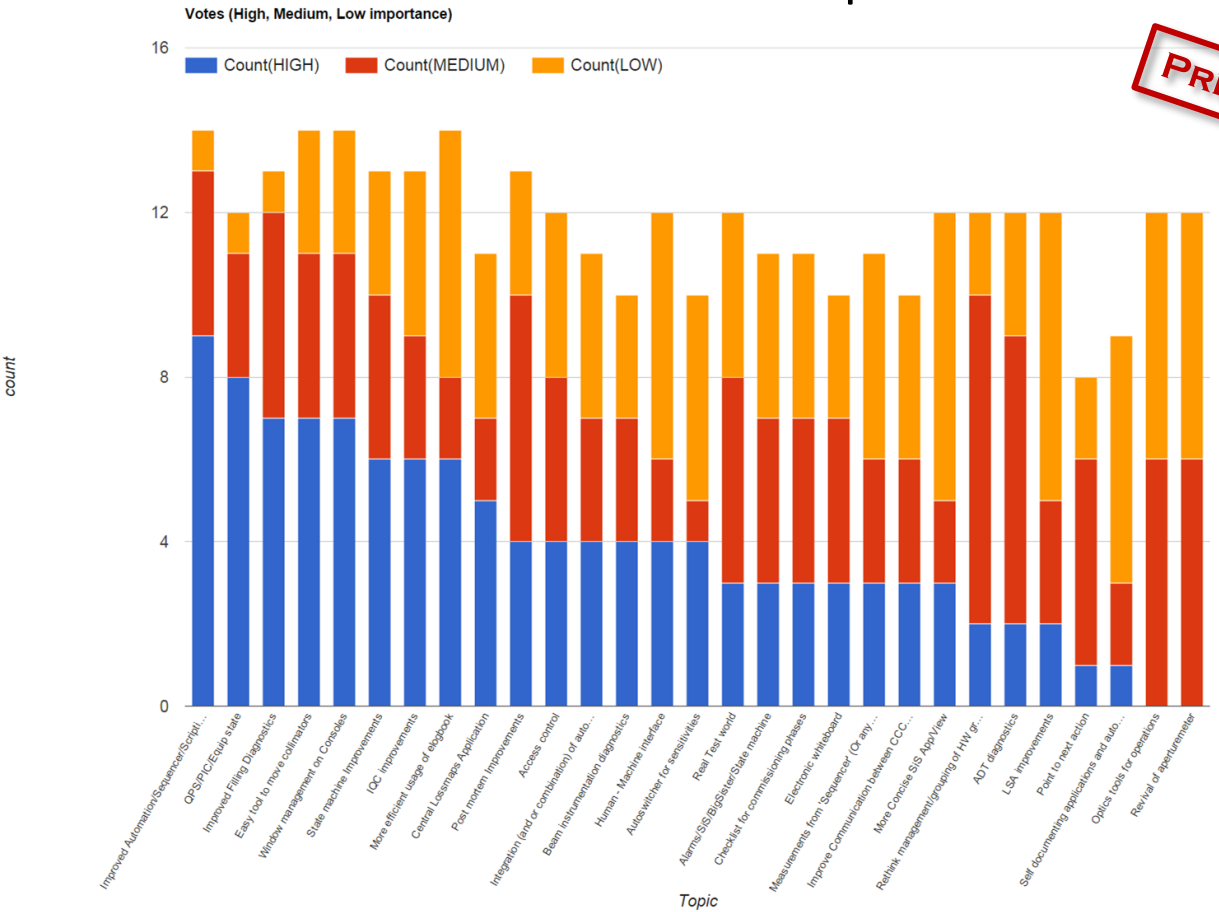
- Easy way to formulate commissioning tests
- MDs

1 – Improved Automation II

Settings vs. Sequences

- Clear separation between settings and sequence
  → Avoid hardcoded values in sequence (devices, contexts, values…)
- Templates/Operational Scenarios (certain modes of operation (e.g. Proton Physics, Lossmaps, VdM scans)
  - Avoid e.g. different sequences for different particle types.
    (Now Copies with slight changes)
  - Clearly see the discrepancies when something is wrong.

Just for the sake of completeness …



PRELIMINARY

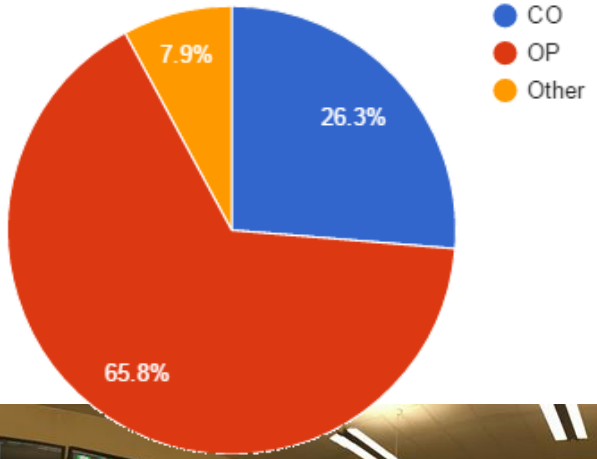
→ To be refined in January

https://docs.google.com/spreadsheets/d/1aLT2UMOGL9T4o2hQSj1VHOH0CVQmZvGr5DraD0u9Sgo/pubchart?oid=751201601&format=interactive

Content

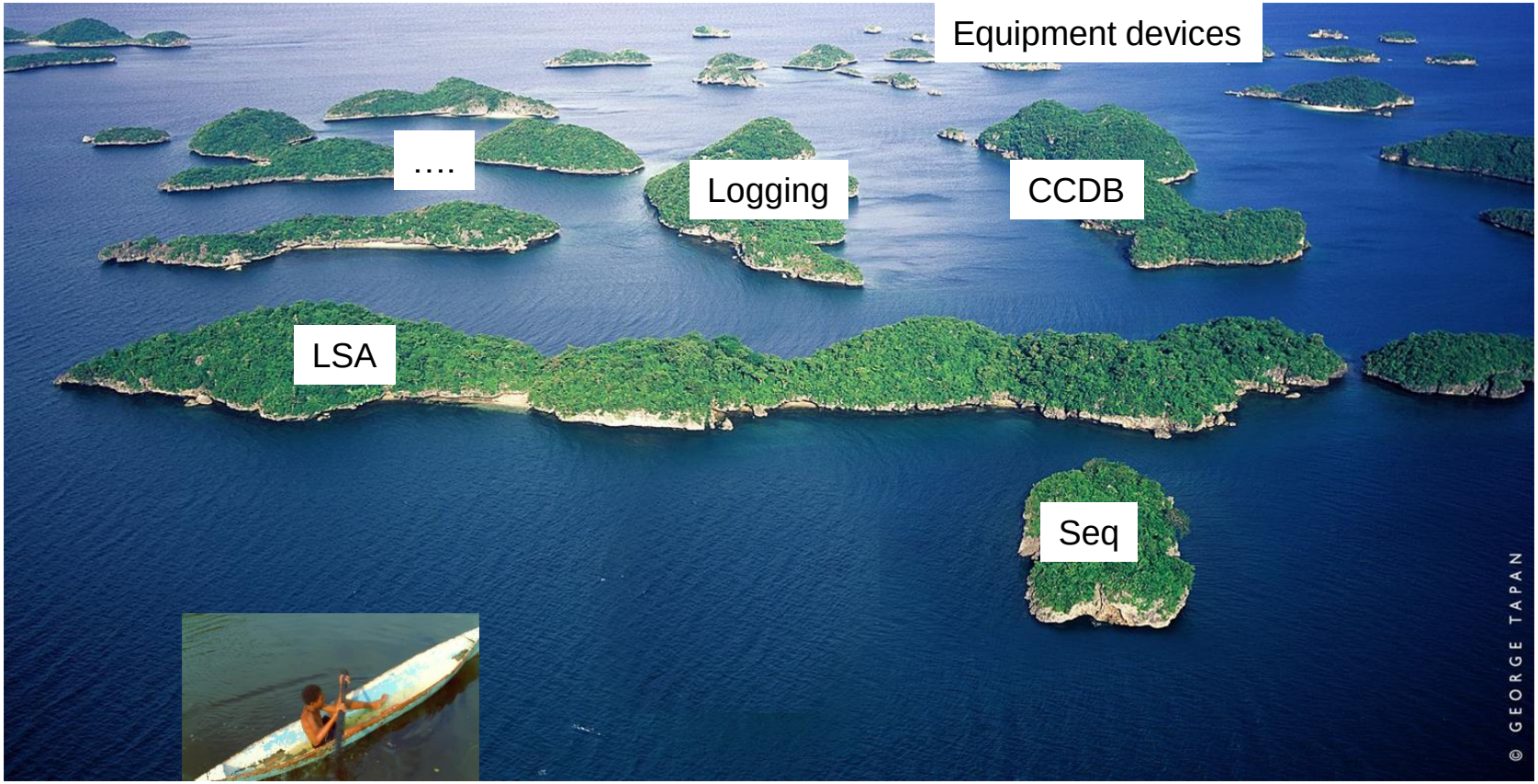
- Users view
- Developers View
- Summary

The uppermost layer

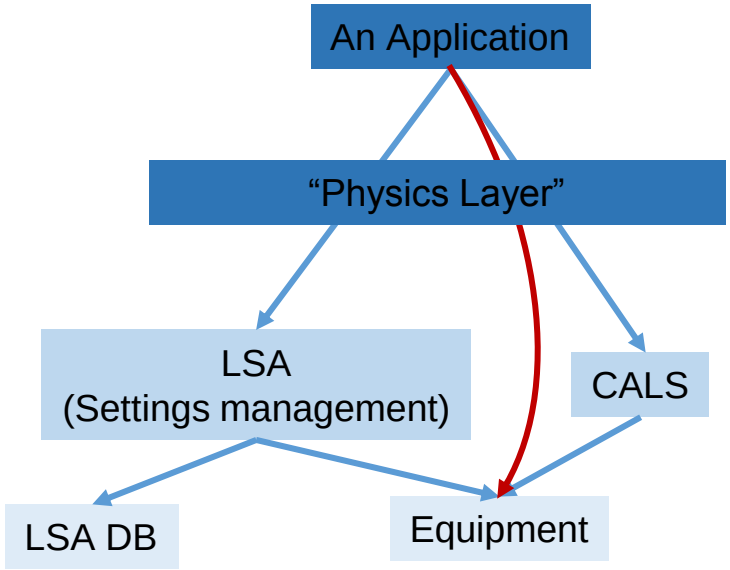


Courtesy: D. Valuch

Application developers perspective



The layered control system …

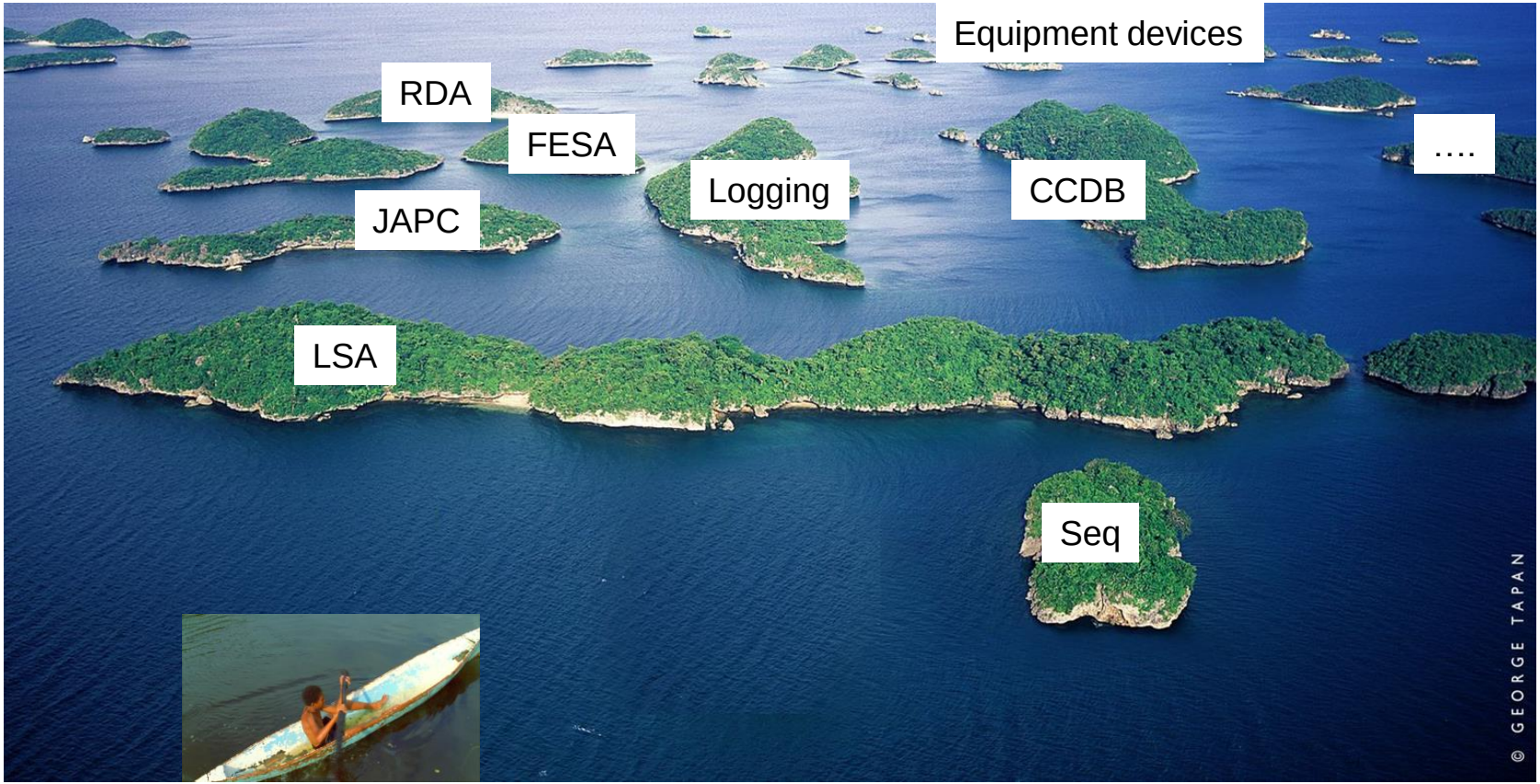


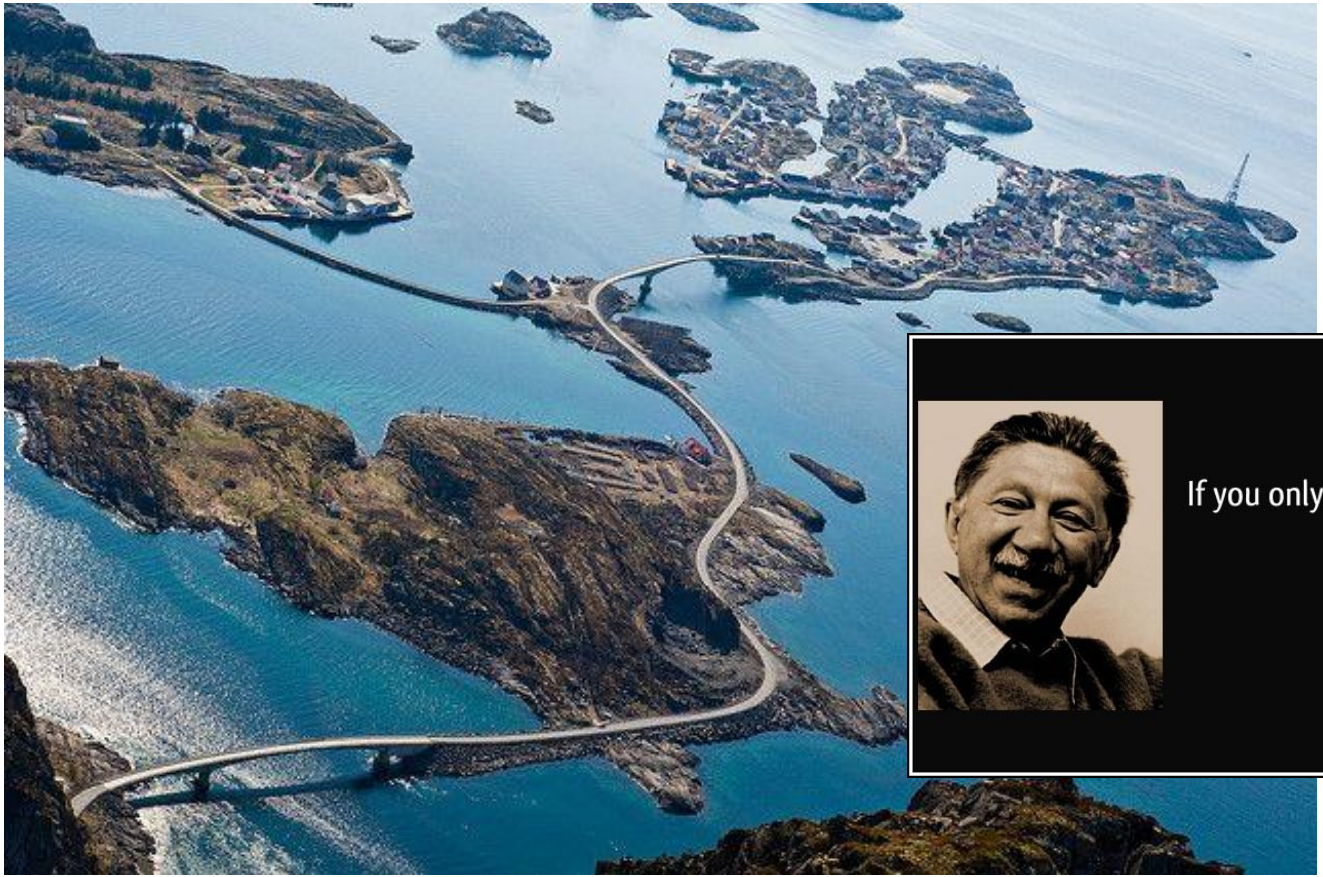
- Domain Driven Design
- Ease testing
- Making writing Applications easy!

```
Tune tune = get(HORIZONTAL, TUNE).of(LHC);

on(HORIZONTAL, TUNE).of(LHC).subscribe((tune) -> System.out.println(tune));
```

OP SW developers perspective





If you only have a hammer, you tend to see every problem as a nail.

(Abraham Maslow)

izquotes.com

Who has the global view?

Global long term vision

Based on some principles e.g.

- Intuitive
- Consistent
- Simple on top, easy to dig down.
- Built to be tested
- Built to grow
- Right dose of Automation
- ...

Do not be afraid of validation testing …. ?

Procedure for software tests during operational periods:

Option A - Do not test

Option B - Be brave and do the following:

1. Check that you are connected to LSA NEXT and your test device
2. Open LHC Page 1
3. Check that you are really connected to LSA NEXT and the test device (better restart the application once more, just to be sure)
4. Close your eyes and and mumble the mantra: “No worries, RBAC will save me if I do something wrong!”.
5. press the button …
6. Open your eyes and check on page 1 if the beam is still in …
7. (optional) – if the beam was dumped, call the CCC and apologize

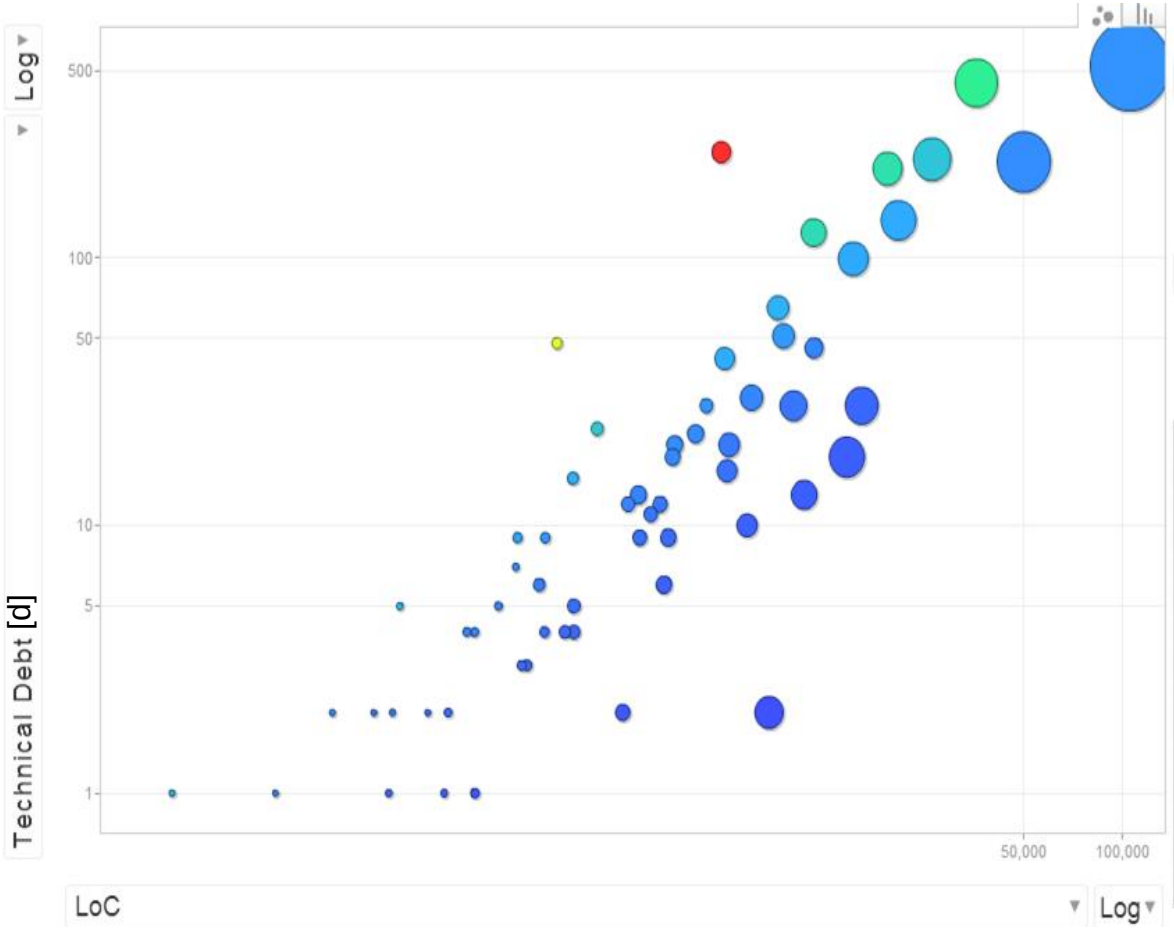
→ Would appreciate a comforting development and testing environment outside the TN ☺

LHC OP Software - Status

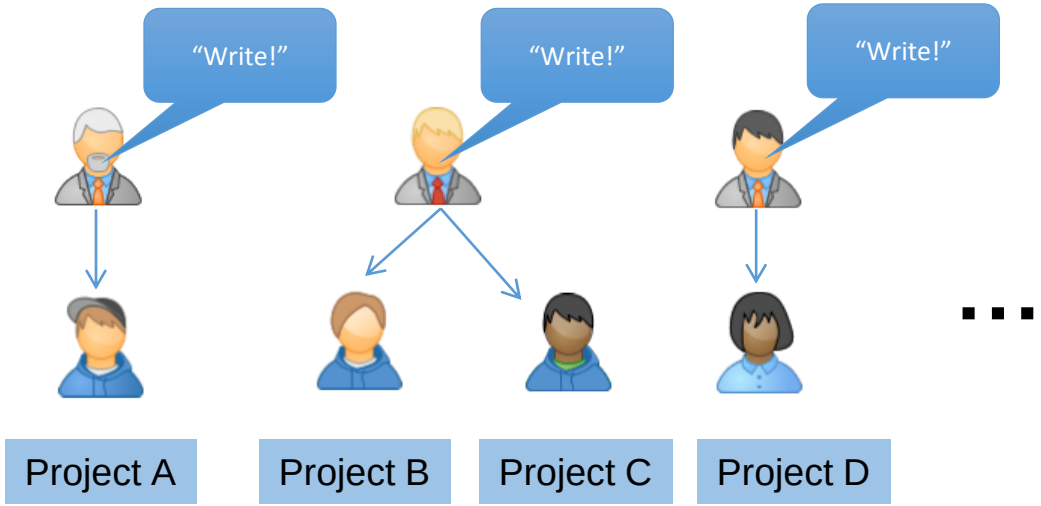
- ~500.000 lines of code (800.000 lines in total)
- 13.6 manyears of technical dept

- 6+2(SPS) people writing software

→ 8.4 minutes of tech dept per line of code!!!!!



How come?

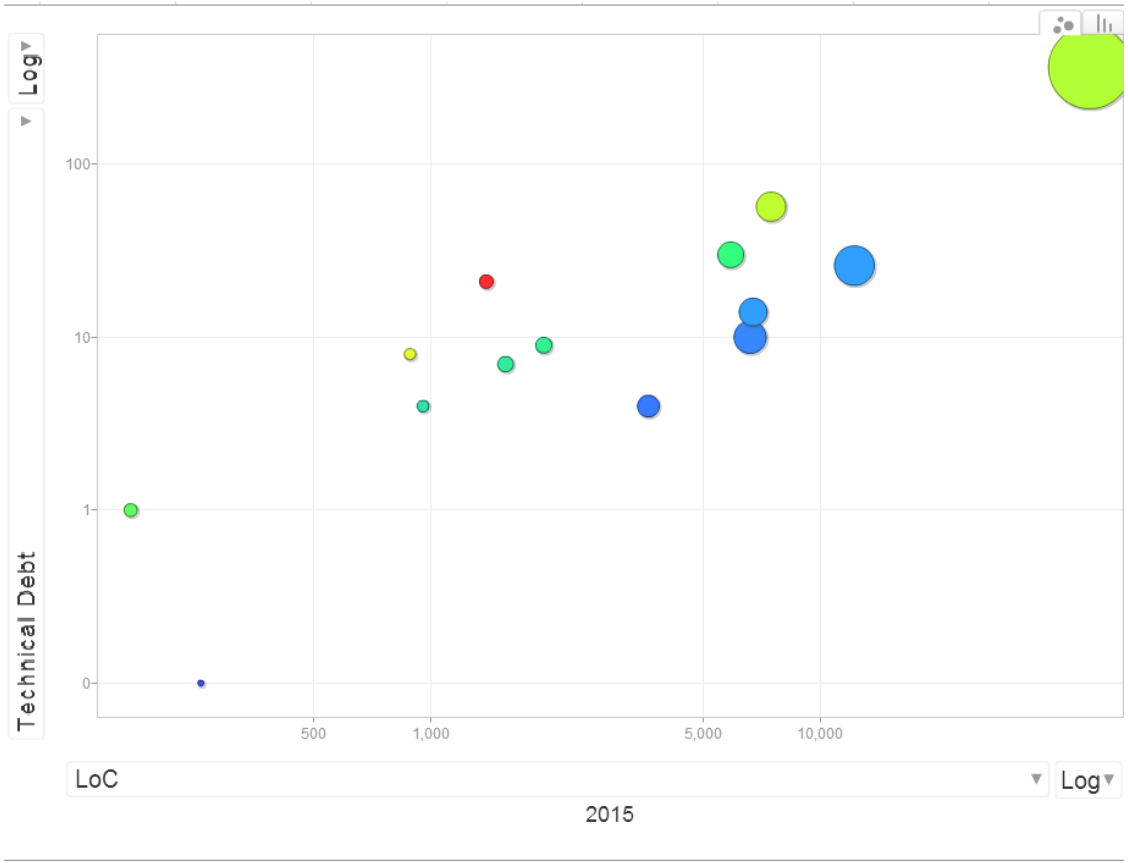


Potential Incoming

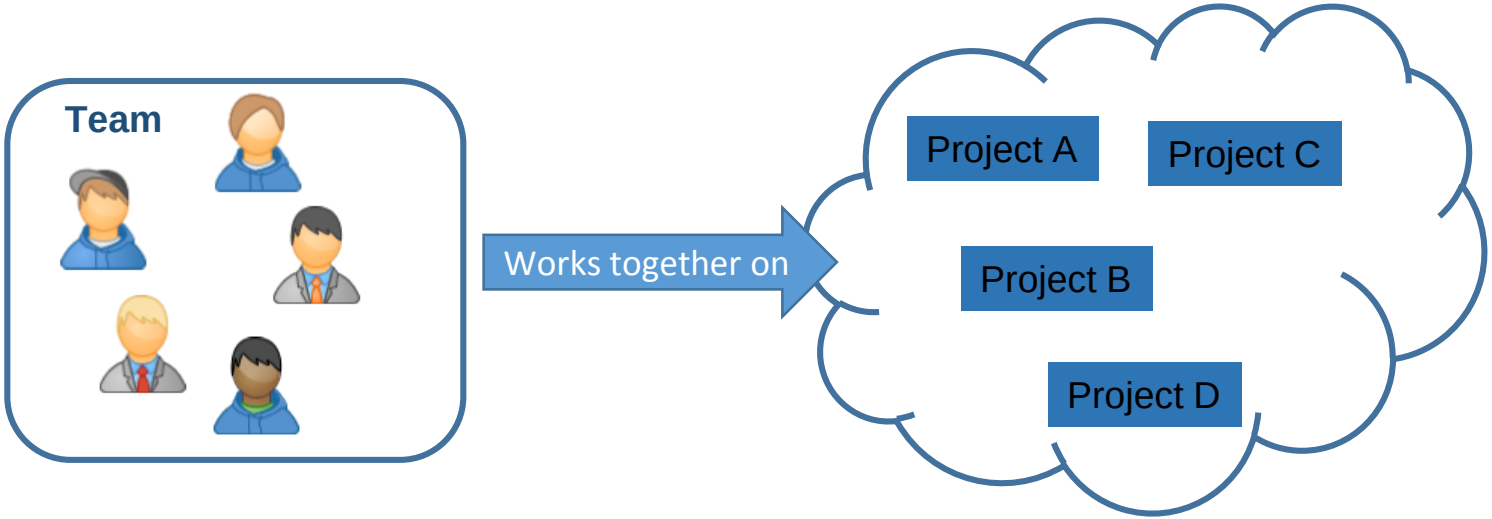
- ~100.000 Lines of code (20% of actual codebase) (~160.000 lines in total)
- ~2.5 man-years of tech debt
- → ~ 8 min/LoC

- ~2 students projects + 1 external SW stack.

- Lack of continuity



How we are trying to tackle this problem



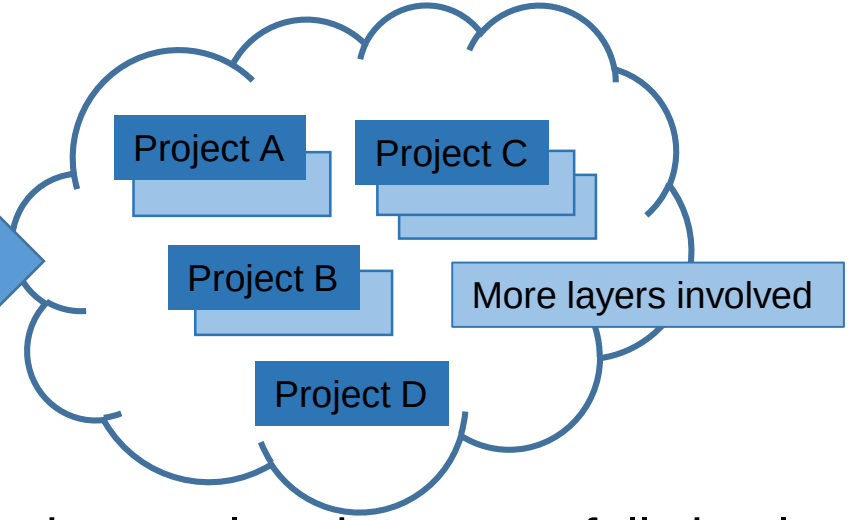
- + Knowledge exchange
- →better SW!
- + Focus (more done)
- + More fun!!

Because of shiftwork, difficult to ensure

- … continuous progress
- … continuous supervision
- … continuous support

Collaborate more with other sections

Works together on



- Approach was already successfully implemented for development of FB-testing framework.
- Some iterations “in preparation” together with TE-MPE for Jan/Feb
- CO-APS is willing to give it a try together with us next year (“good old LSA collaboration” ;-)

Content

Summary

Everything working well! Time to evolve (carefully)!

As Control System Users, we would like
- a Control system that
  - behaves as we expect it to
  - prevents us from making mistakes
- Evolution according to our priorities

… while we know that we have to improve in defining our priorities.

As LHC-OP SW developers, we would like to
- develop in teams together with CO developers.
- have intuitive domain specific layers to program against and
- be closely involved in evolving the control system in design and implementation as well as strategic decisions.

… while we know that we have to improve our SW-development skills


Careful evolution to drive CERN’s
accelerators safely into the future