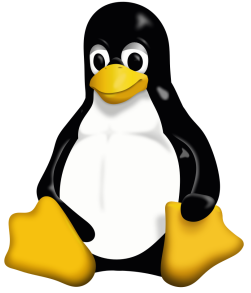
ObsBox

A Linux-based real-time system for LHC Beam Monitoring

Kernel Recipes 2018

Miguel Ojeda

miguel@ojeda.io – https://ojeda.io



CERN

The European Organization for Nuclear Research

CERN – What is it?

***C**onseil **E**uropéen pour la **R**echerche **N**ucléaire*

- acronym

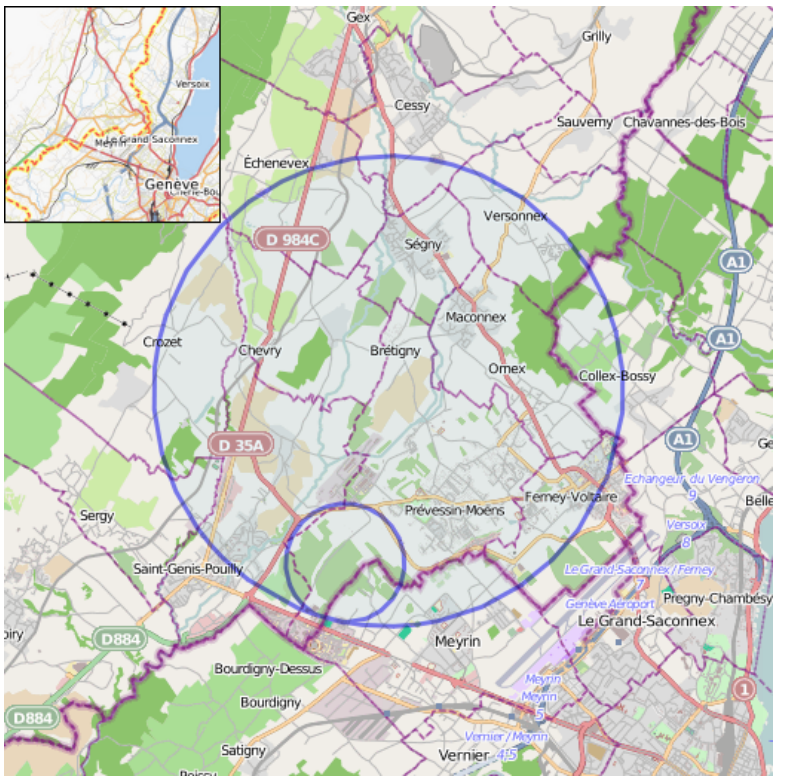
European research organization

- operating the largest particle physics lab in the world

Function: to provide the **particle accelerators for high-energy physics** (HEP) research

CERN – Where is it?

Near Geneva, on the Franco–Swiss border



CERN – Origins

Founded in **1952** by a handful of Europe's leading scientists

- mandate: establishing a world-class fundamental physics research organization in Europe.

Originally, **pure physics research**

- concentrated on understanding the inside of the atom (⇒"nuclear").


FRANCE
ITALIE

CERN – Nowadays

Research: **particle physics**

- the study of the fundamental constituents of matter and the forces acting between them

International effort

- **22** Member States
- **5** + **3** Associate Members States
- observers: **Japan, Russia, USA, EU, JINR, UNESCO**...
- **36** Non-member States
- **18** States with scientific contacts
- **600+** institutes and universities using CERN's facilities


190 m
SPIE

CERN – ./MAINTAINERS

2500+ Staff Members

- scientific, technical & administrative

12000+ Visiting Scientists

- half of the world’s particle physicists!
- **70+** countries
- **100+** nationalities

Me – Work

Software Engineer, worked at CERN for 7 years

Physics Department – CMS detector

- redesign of a VCS on top of CVS
- redesign beam data automatic serialization & storage systems
- co-manager of the Data Quality Monitoring team

Beams Department

- real-time RF control systems of particle accelerators

Me – Fun

First kernel patch in 2006

Few months later, a couple of simple drivers

- Greg, update the LDD book!

Maintainer of the `drivers/auxdisplay` tree

C++ aficionado

- do not tell Linus 🙏

CERN – Particle physics in 9 steps

Real easy ☺

- 1.
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2.
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3.
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4.
- 5.
- 6.
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4. **Filter** in real-time ungodly amounts of data
- 5.
- 6.
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4. **Filter** in real-time ungodly amounts of data
- 5. **Store** whatever is deemed important, lose everything else
- 6.
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4. **Filter** in real-time ungodly amounts of data
- 5. **Store** whatever is deemed important, lose everything else
- 6. **Repeat** `24/7` for months, every `25 ns`
- 7.
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4. **Filter** in real-time ungodly amounts of data
- 5. **Store** whatever is deemed important, lose everything else
- 6. **Repeat** `24/7` for months, every `25 ns`
- 7. **Analyze** the resulting data
- 8.
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4. **Filter** in real-time ungodly amounts of data
- 5. **Store** whatever is deemed important, lose everything else
- 6. **Repeat** `24/7` for months, every `25 ns`
- 7. **Analyze** the resulting data
- 8. **Test** if the theoretical expections match reality
- 9.

CERN – Particle physics in 9 steps

Real easy ☺

- 1. **Accelerate** particles to the speed of light (almost!)
- 2. **Smash** them in opposite directions
- 3. **Detect** the byproducts of these collisions
- 4. **Filter** in real-time ungodly amounts of data
- 5. **Store** whatever is deemed important, lose everything else
- 6. **Repeat** `24/7` for months, every `25 ns`
- 7. **Analyze** the resulting data
- 8. **Test** if the theoretical expections match reality
- 9. **Get** a Nobel prize!

CERN – Accelerators

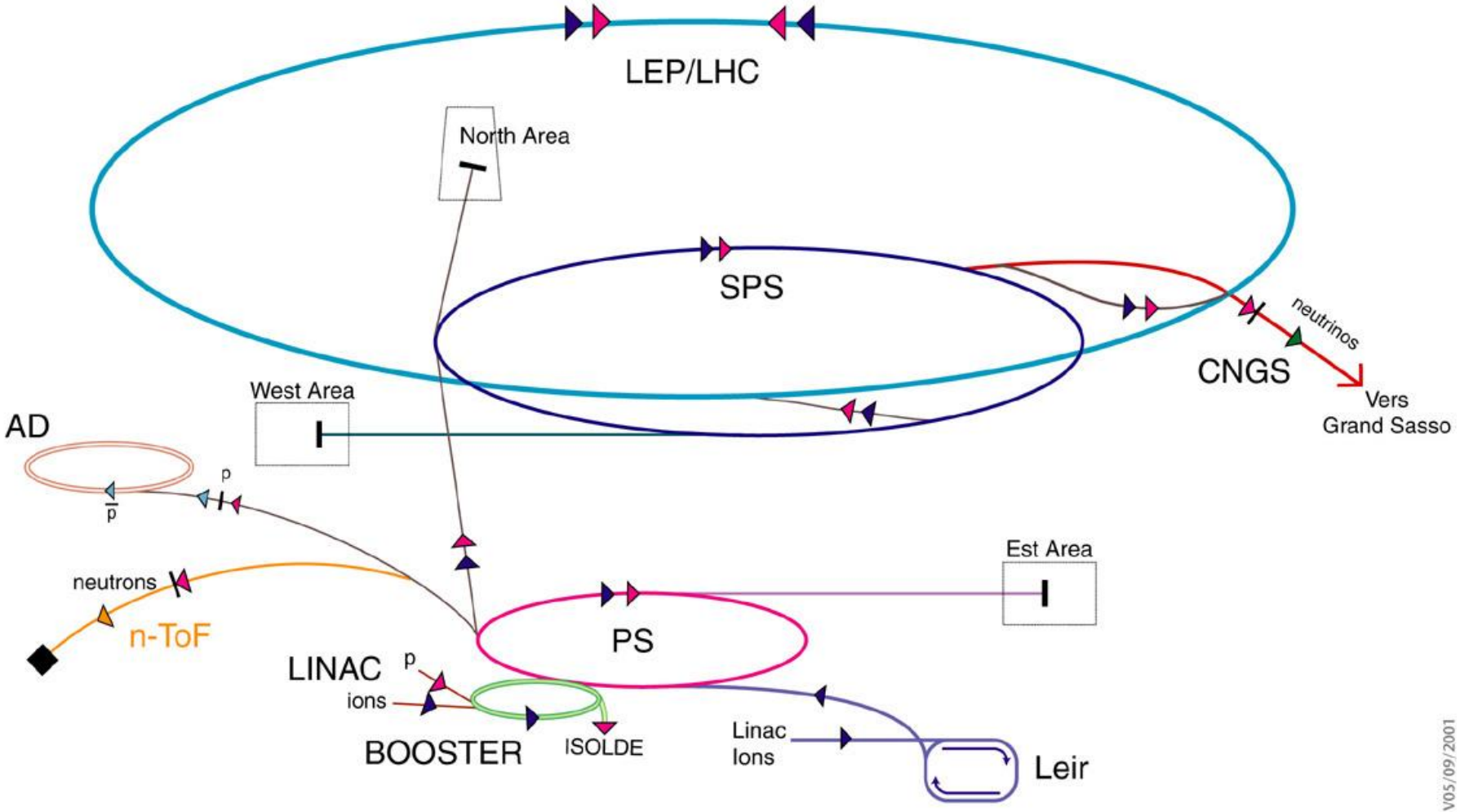
LHC – *Large Hadron Collider*

- the famous one
- actually the last piece on a kilometric chain!

All starts in a bottle of hydrogen

- each accelerators boosts the beam energy...
- ...then its injected into the next one
- Linac 2 ⇒ PSB ⇒ PS ⇒ SPS ⇒ LHC


PREINJECTEUR
LINAC 2
HAUTE
HIGH
100 KV
H2
A PROTON SOURCE


LEP/LHC
North Area
SPS
neutrinos
CNGS
Vers
Grand Sasso
West Area
AD
p
p̄
neutrons
n-ToF
Est Area
PS
LINAC
p
ions
BOOSTER
ISOLDE
Linac
Ions
Leir
p (proton)
ion
neutrons
p̄ (antiproton)
proton/antiproton conversion
neutrinos
AD Antiproton Decelerator
PS Proton Synchrotron
SPS Super Proton Synchrotron
LHC Large Hadron Collider
n-ToF Neutrons Time of Flight
CNGS Cern Neutrinos Grand Sasso
CERN AC_HF205_V05/09/2001

CERN – Accelerators

Beams accelerated up to `6.5 TeV`

- ⇒ roughly `1 μJ`

For a proton ⇒ `~0.99999999c`

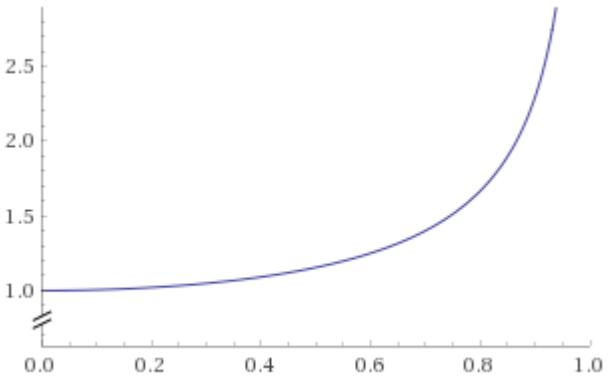
- just `11 km/h` less than `c`!
- `~11230` revolutions per second (LHC: `26.7 km`)

CERN – Accelerators

$$E = \frac{1}{2}mv^2$$

$$E = \frac{mc^2}{\sqrt{1 - \frac{v^2}{c^2}}}$$

$$E = mc^2\left(1 + \frac{1}{2}\frac{v^2}{c^2} + O(\frac{v^4}{c^4})\right)$$



CERN – Accelerators

Radio-Frequency (RF) cavities

- to actually accelerate the particles

Ultrahigh Vacuum

- otherwise, they would interact with the medium

Superconducting Magnets

- to keep them in track!
- `8+ T`

Cryogenics

- Largest in the world
- Colder than outer space (`1.9 K` vs. `2.7 K`)

RF Video

CERN – Experiments

So, we smash a beam of protons against another beam in the opposite direction ⇒ A lot of debris happens

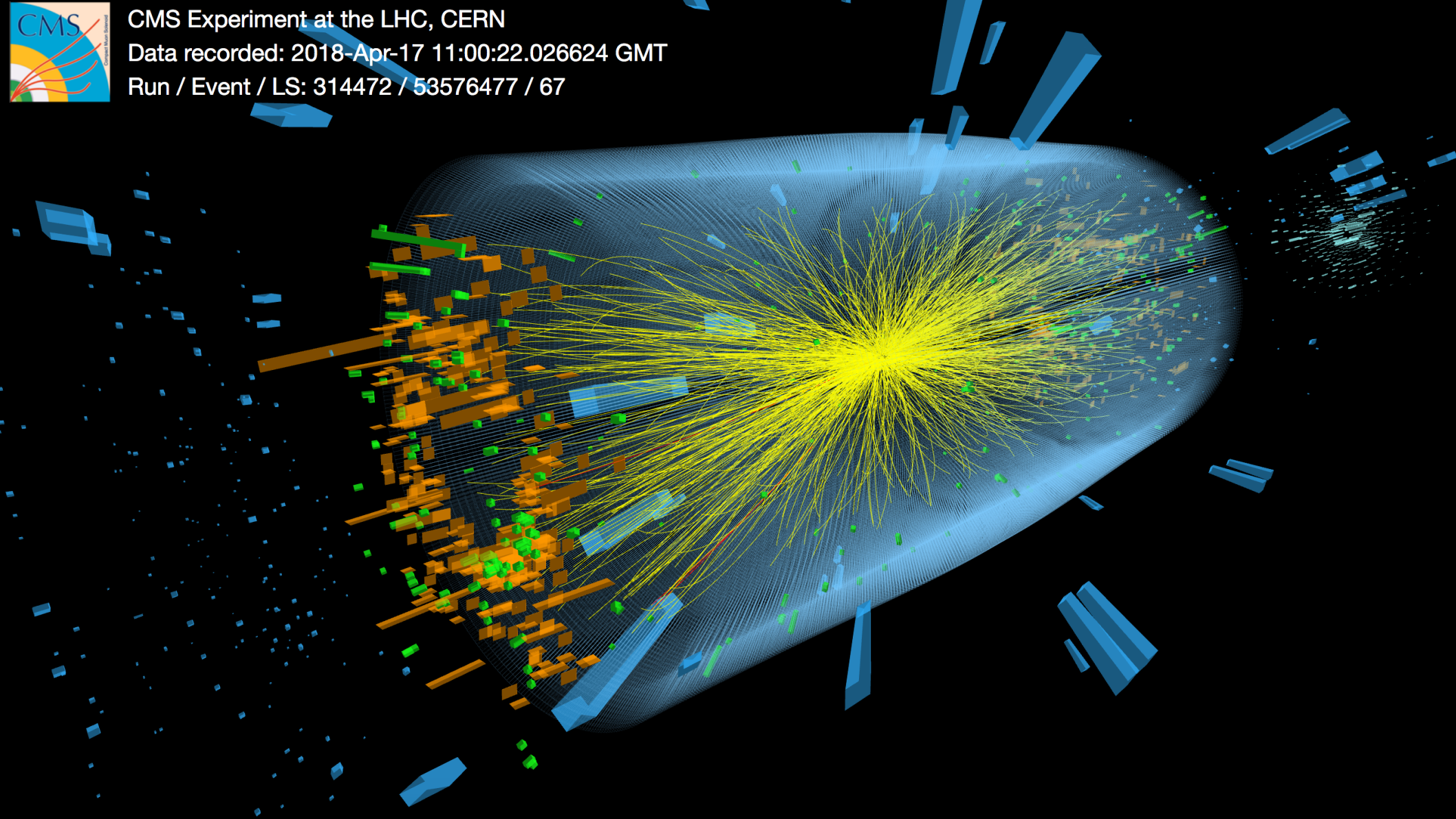
- a collision every `25 ns`
- this is called **an event**

Many layers of **subdetectors** setup around the collision site

- ⇒ hints about the particle properties (speed, mass, charge…)

Sadly, we cannot store everything

- ⇒ we have to **filter** in real-time, saving only promising ones


CMS Experiment at the LHC, CERN
Data recorded: 2018-Apr-17 11:00:22.026624 GMT
Run / Event / LS: 314472 / 53576477 / 67

CERN – Experiments

With all that, we can “backtrack” in time

- infering what particles originated in the collision
- ...with some degree of confidence!

Using enough events, we can get enough **confidence** about a theory

- e.g. the existance of the Higgs boson
- $5\ \sigma$ is required to announce a discovery

Experiments Video

CERN Overview Video

CERN – Much more!

There are many, many more experiments and technical stuff happening at CERN!

In **Physics**

- quark-gluon plasma (a “fluid” form of matter) in ALICE
- antimatter creation, storage and research in AD and LHCb
- researching forward particles in TOTEM and LHCf
- searching for a magnetic monopole particle in MoEDAL
- neutrino research in nTOF
- solar axions detection in CAST
- ...

CERN – Much more!

In **Computer Science**

- LHC Computing Grid
- World Wide Web
- EOS distributed filesystem
- Collaboration with companies – e.g. CERN openlab (Intel, Oracle, Huawei, Siemens…)
- ...

CERN – Much more!

A lot of C++

- 50 million lines of code (2011 estimate)
- Member of the ISO C++ Standards Committee

A lot of Python

- e.g. used as a configuration language for simulations run in C++
- Web applications galore

Dozens of data analysis/simulation packages

- ROOT (Data Analysis Framework)
- Geant4 (Simulation of passage of particles through matter)
- Monte Carlo simulations (many)
- ...

CERN – Much more!

Testing bleeding edge compiler releases

- Quite a lot of bugs submitted!

e.g. migration to Git for the CMS detector software

- ~5 million lines of code
- 700+ contributors
- 200k commits (Linux: 800k)
- 2010: 1000+ subfolders of "projects" under CVS
- 2011: started recording git-like commits across all the projects
- 2013: GitHub migration completed

CERN – Learn more!

https://home.cern

ObsBox

The Observation Box

ObsBox – What is it?

In the LHC, we have acquisitions systems to digitize beam phase and position

- RF and transverse damper feedback

Originally, they were slow

- few ms of bunch-by-bunch data
- bunch = a cluster of particles in a beam, 3564 bunches in the LHC
- over a VME bus

During LHC Run I (2009-2013), experience showed that we needed more ⇒ the project was born

ObsBox – What is it?

Streams captured data directly out of the feedback hardware

- into a Linux server
- through an optical fiber link

Allows processing and buffering of at least 1 minute of data

- in the cheapest configuration
- online Fourier analysis of transverse position data

Connected to an LHC-wide trigger network

- for detection of beam instabilities

Data made available for analysis by client applications

- as CERN-standard control systems

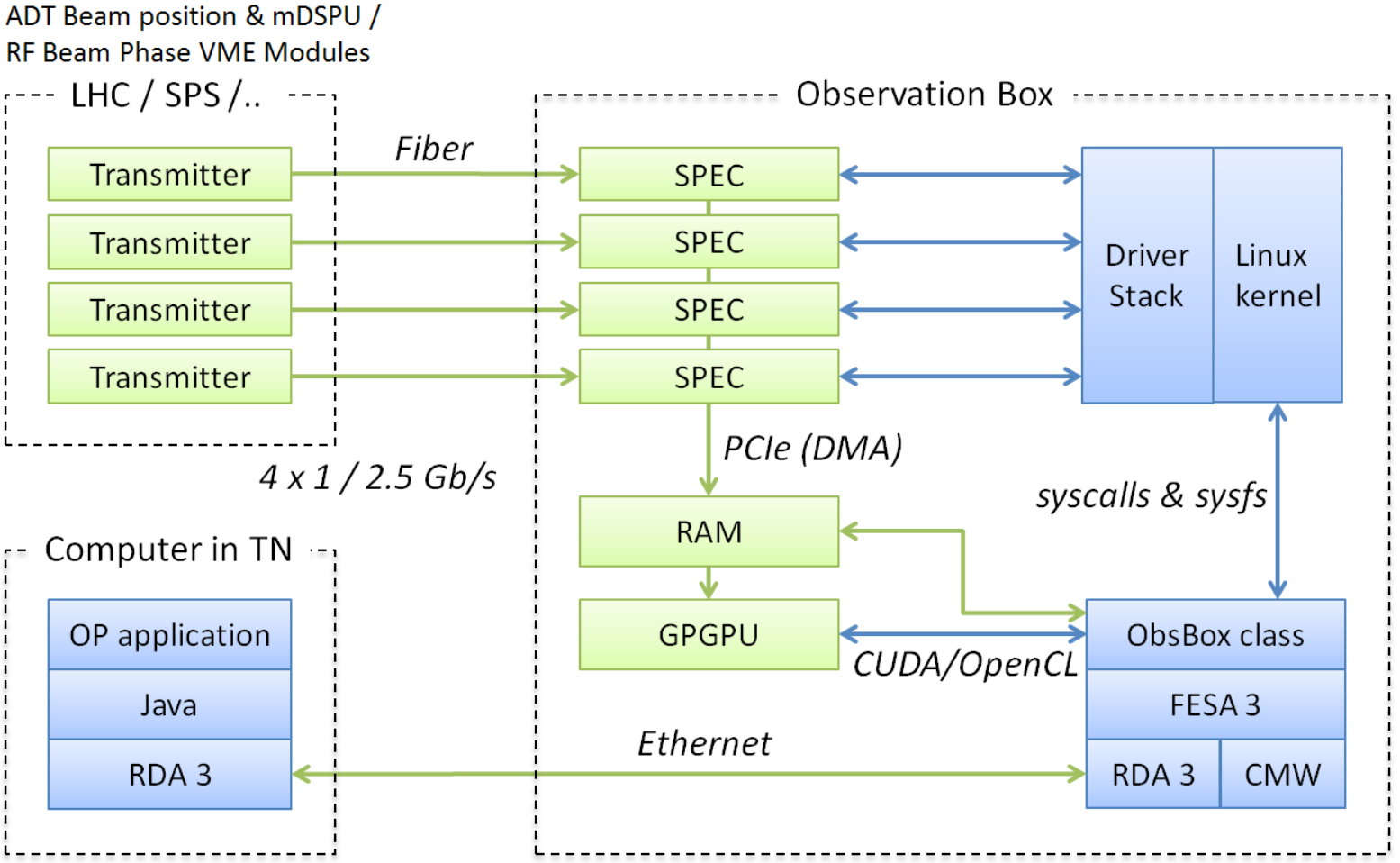
ObsBox – `./CREDITS`

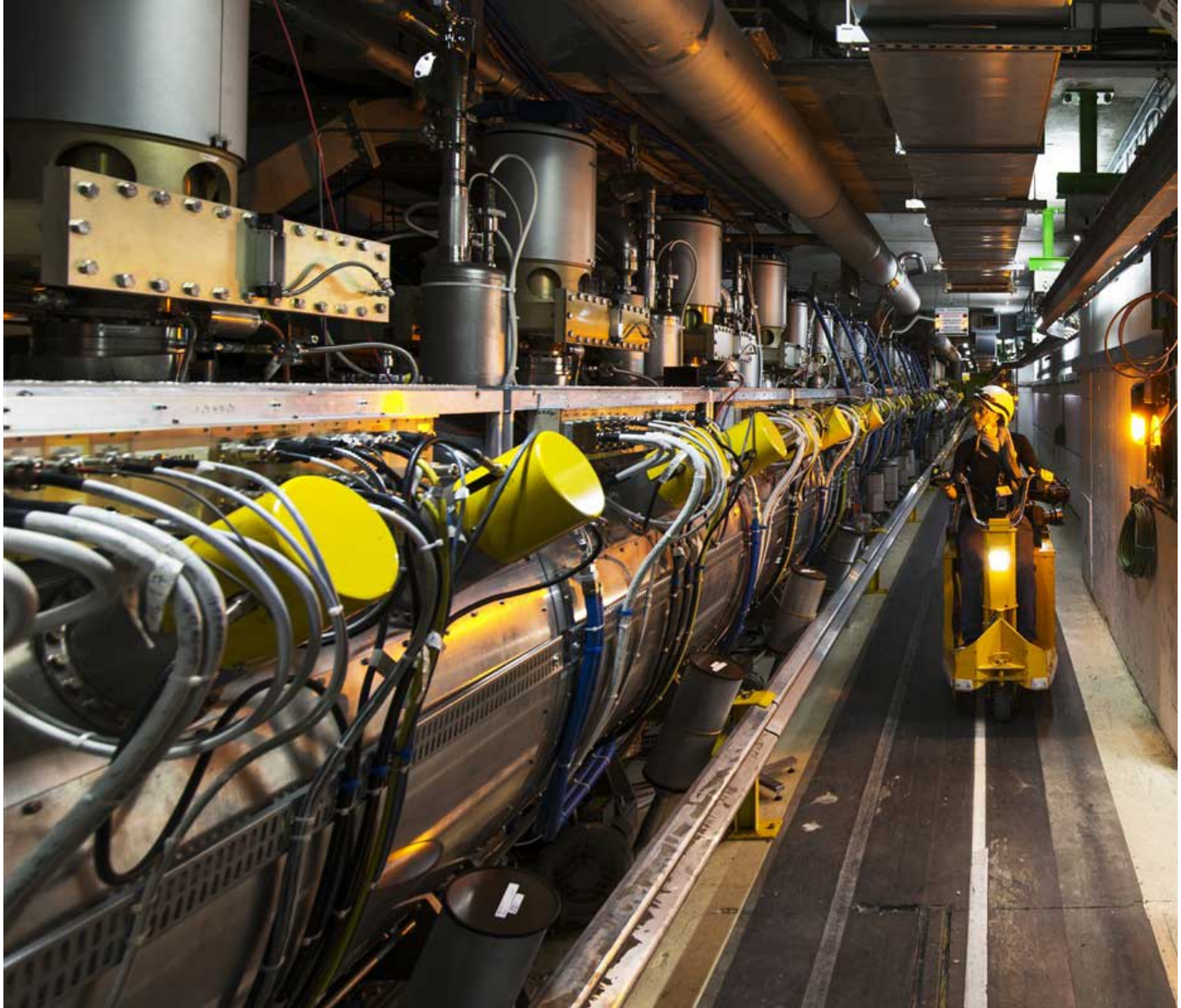
The design and implementation – HW and SW – was done by many people:

- Philippe Baudrenghien
- Andrew Butterworth
- Javier Galindo
- Gerd Kotzian
- Wolfgang Höfle
- Tom Levens
- John Molendijk
- Miguel Ojeda
- Martin Söderén
- Federico Vaga
- Daniel Valuch

...and many others in different departments! Thanks a lot!

ObsBox – System Overview





ObsBox – Transmitters

VME modules

- in underground caverns close to the beam

With embedded **FPGA**s

- that sample different aspects of the beam at a high frequency

For the LHC transverse damper system (ADT)

- 2 or 4 pickups per plane per beam
- to measure beam position at 40 MS/s

ObsBox – Links

Optical **fiber link**

- to a receiver in the surface

Unidirectional ⇒ no synchronization

- ⇒ protocol is trivial
- CRC included

1 Gbps per fiber link

- 2.5 Gbps designed top speed
- For LHC use case, running with 4 links

ObsBox – Receiver

Fiber connected to FPGA Mezzanine Card (VITA 57)

Mezzanine installed in Simple PCI Express Carrier (SPEC)

- from the Open Hardware Repository

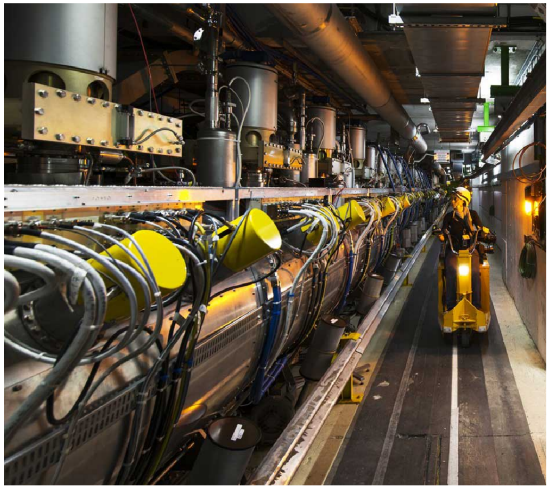
SPEC plugged into a Supermicro server

- SuperServer 6028U-TR4+
- 128 GB RAM in the first use case, up to 3 TB

Running customized Linux kernel

- real-time patches
- custom driver

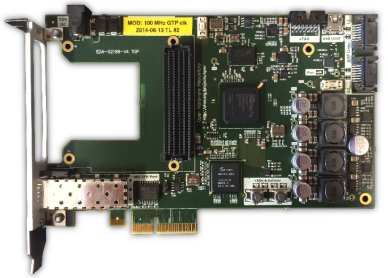
RF cavities



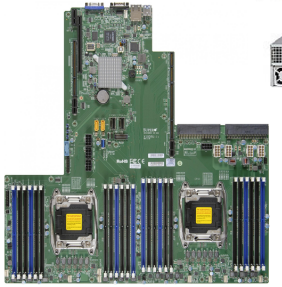
RF VME modules

Fiber link

Simple PCIe FMC carrier (SPEC)



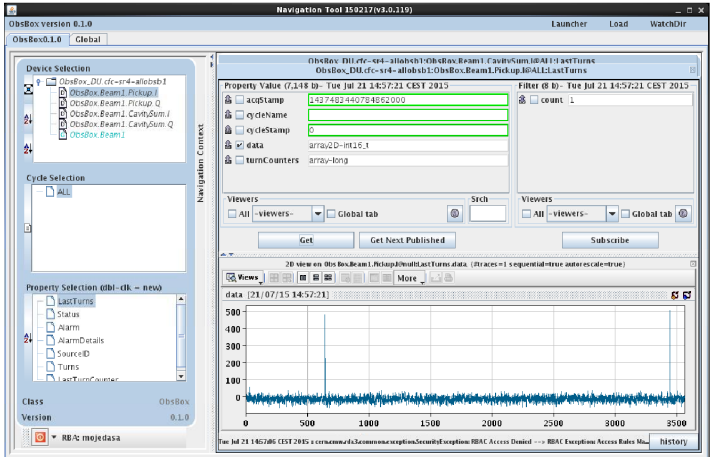
SuperMicro chassis



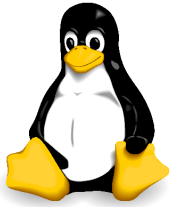
SuperMicro motherboard

ObsBox

Analysis and monitoring software (clients)



Ethernet TCP/IP



Custom Linux kernel driver

The Ultimate I/O framework (ZIO)

Front-End Software Architecture (FESA)

ObsBox – Driver

Implemented on top of **ZIO**

- “the Ultimate Linux I/O Framework”
- intended for physics labs
- started in 2011 to improve on Comdedi and IIO

Customized driver to provide a simple interface

- blocking `read()` syscalls
- ...plus a few control configuration values

ObsBox – Userspace

Real-time process

- Continously acquiring raw data from the driver
- while “synchronizing” & validating it

Buffers circularly the data in huge areas of memory

- the reason behind the `128 GB` of RAM
- for clients to require any of it at any time
- supports snapshots by triggering

Performs online analysis on the data

- for cases where offline is not possible

Serves the data to network clients

- with possible pre-filtering

ObsBox – Userspace

Acquisition threads

- real-time
- fetch data as quickly as possible from the kernel
- avoids saturating small kernel buffers
- simply doing blocking `read()` syscalls ⇒ trivial
- 1 per link

Worker threads

- real-time, slightly lower priority
- interpreting the data, validating and buffering it
- 1 per link

Other threads

- deal with supplying the buffered data, interrupts, configuration, control, networking...

ObsBox – Userspace

A lot of the **infrastructure** provided by the FESA framework

- “**F**ront-**E**nd **S**oftware **A**rchitecture”
- front-end = computers running RT control systems

Abstracts control systems behind a common interface

Supplies networking

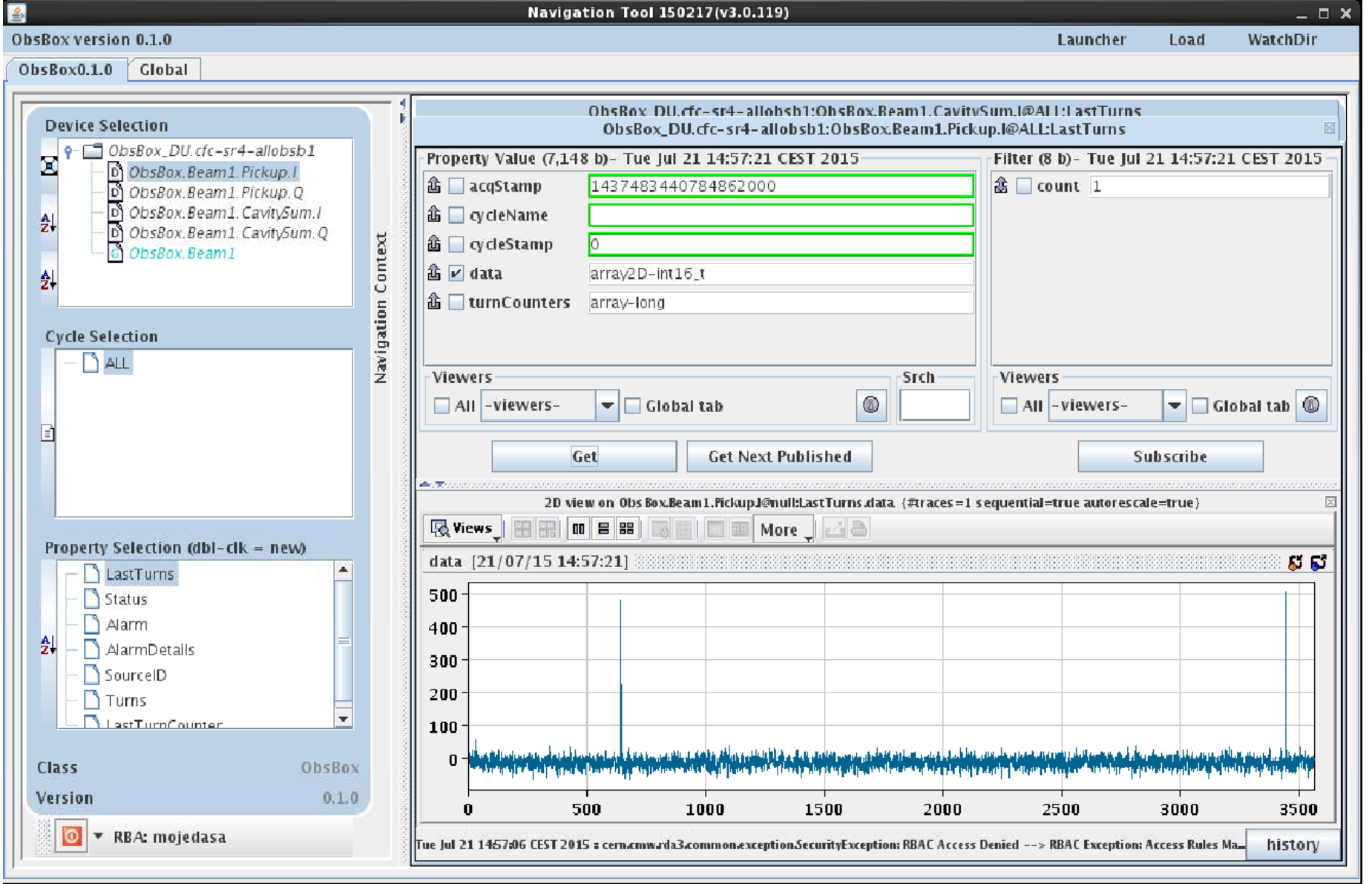
- leveraging ZeroMQ

Provides a library, CLI and GUI

- very useful for control experts and operators

Written in C++

- both FESA and the control systems on top


Navigation Tool 150217(v3.0.119)
ObsBox version 0.1.0
Launcher
Load
WatchDir
ObsBox0.1.0
Global
Device Selection
ObsBox_DU.cfc-sr4-allobsb1
ObsBox.Beam1.Pickup.I
ObsBox.Beam1.Pickup.Q
ObsBox.Beam1.CavitySum.I
ObsBox.Beam1.CavitySum.Q
ObsBox.Beam1
Cycle Selection
ALL
Property Selection (dbl-clk = new)
LastTurns
Status
Alarm
AlarmDetails
SourceID
Turns
LastTurnCounter
Class
ObsBox
Version
0.1.0
RBA: mojedasa
Navigation Context
ObsBox_DU.cfc-sr4-allobsb1:ObsBox.Beam1.CavitySum.I@ALL:LastTurns
ObsBox_DU.cfc-sr4-allobsb1:ObsBox.Beam1.Pickup.I@ALL:LastTurns
Property Value (7,148 b)- Tue Jul 21 14:57:21 CEST 2015
acqStamp
1437483440784862000
cycleName
cycleStamp
0
data
array2D-int16_t
turnCounters
array-long
Filter (8 b)- Tue Jul 21 14:57:21 CEST 2015
count
1
Viewers
All
-viewers-
Global tab
Srch
Viewers
All
-viewers-
Global tab
Get
Get Next Published
Subscribe
2D view on ObsBox.Beam1.PickupI@null:LastTurns.data {#traces=1 sequential=true autorescale=true}
Views
More
data [21/07/15 14:57:21]
500
400
300
200
100
0
0
500
1000
1500
2000
2500
3000
3500
Tue Jul 21 14:57:06 CEST 2015 : cern.cmw.rda3.common.exception.SecurityException: RBAC Access Denied --> RBAC Exception: Access Rules Ma...
history

Conclusion

Conclusion

CERN needs to develop exotic HW and SW

- custom systems are, many times, unavoidable

Linux is extremely useful for that

- **Free** – budget is a constraint
- **Customizable** – e.g. real-time patches
- **Extendable** – e.g. for writing drivers for exotic HW

Thank you❗

Questions❓