Accelerator Controls

Stephane Deghaye (BE-CSS)

14-03-2024

Agenda

1. Control System Requirements
2. A bit of History
3. Hardware & Software Architecture
4. CERN examples & Key Components

Why a Control System?

- The physicists and operators need to be able to remotely control and monitor the elements the particle accelerators are made off

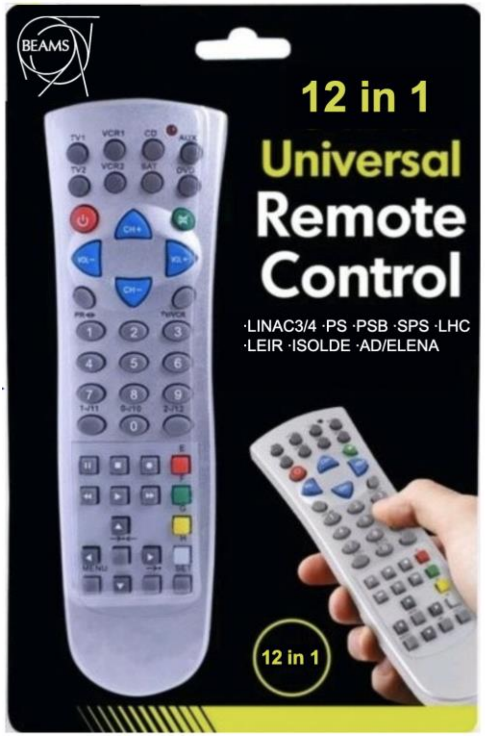
  ➔ this is the role of the **Control System**.

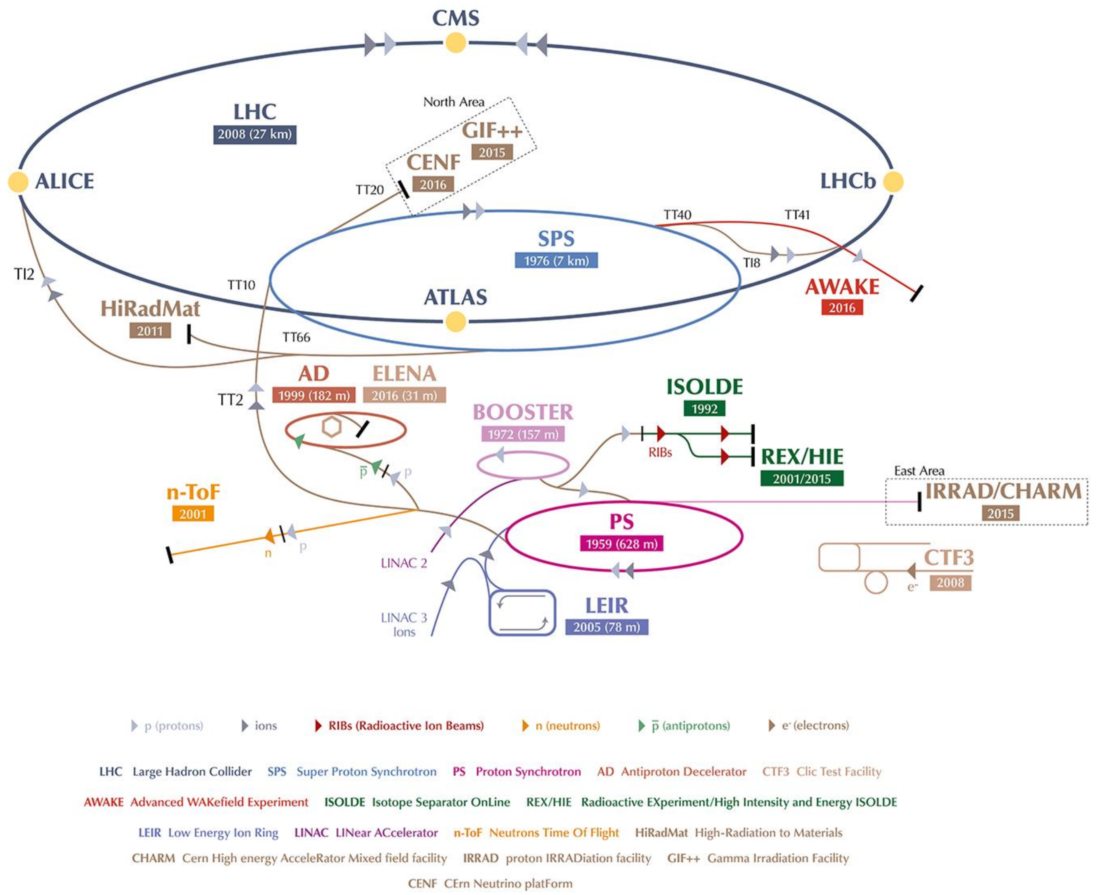
- The Control System's job is to provide to the physicists and operators a means to:
  - set reference values (aka setting) and states in active elements (e.g. power converters),
  - read instruments,
  - monitor the health of sub-systems,
  - diagnose faults,
  - etc.

Controls Complexity

- **Many requirements from physicists and operators**
- **Accelerators made of many elements**
  - Early accelerators, e.g. CERN Proton Synchrotron (PS), were small (< 5'000 devices)
  - Latest accelerators, e.g. LHC, are much more complex to operate (30'000+ elements)

The Control System's job is to hide the complexity and help you to do your job as efficiently as possible.



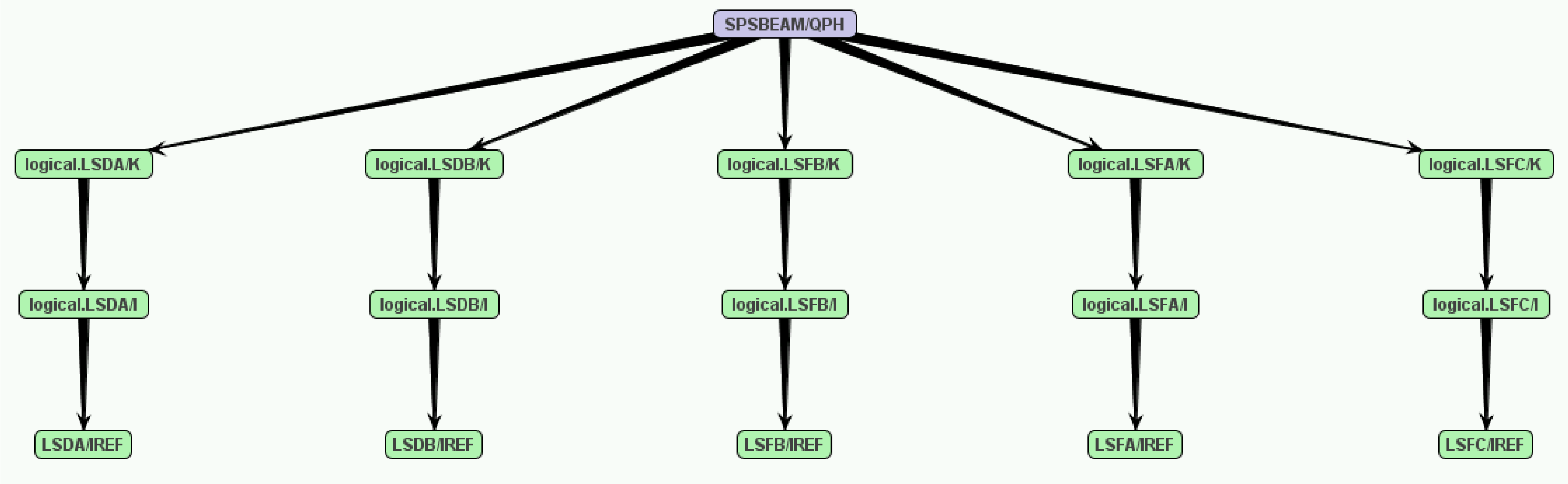


Control System Requirements

Control System Key Requirements

Act on accelerator elements (settings & states)

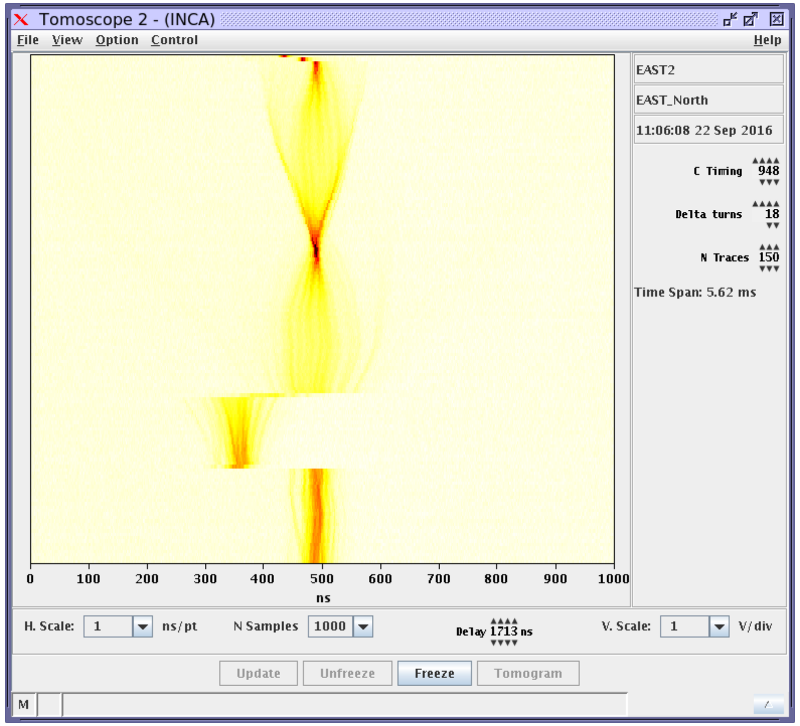
- Minimum: direct access to the hardware values
- Ideal:
  - Model-driven control to work at a higher level
  - Global transactional synchronisation



Control System Key Requirements

Monitor the elements (instruments & actuators)

- Minimum: Display raw acquisitions
- Ideal: Time-tagged, coherent acquisition, post-processing for quick detection of abnormal situations

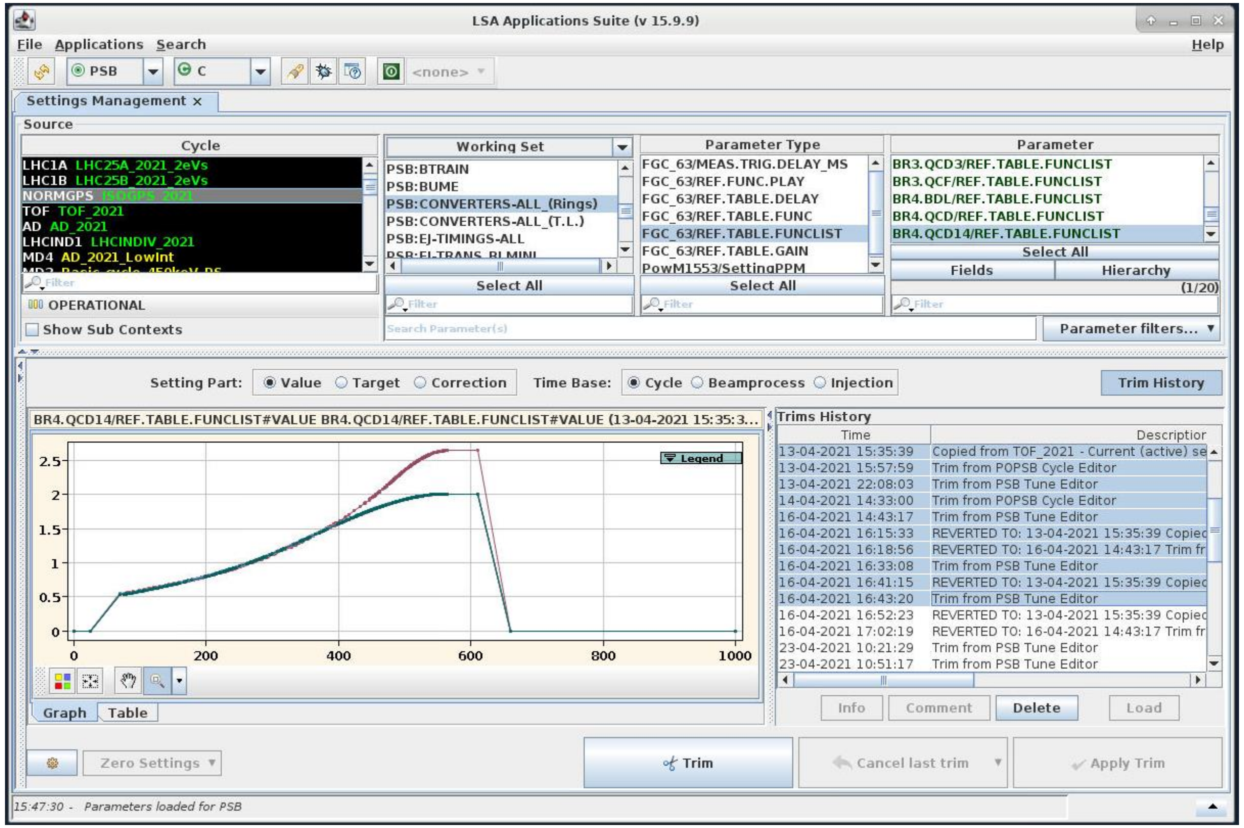


Control System Key Requirements

The CERN Accelerator School

Long-Term Memory of Settings & Acquisitions

- AKA Logging
- Accelerator performance & post-mortem analysis, fine tuning of the machines
- Minimum: structured time-series in a simple format (CSV, SDDS, etc.)
- Ideal: Years of data (settings & acquisitions) with performant data extraction & analysis tools

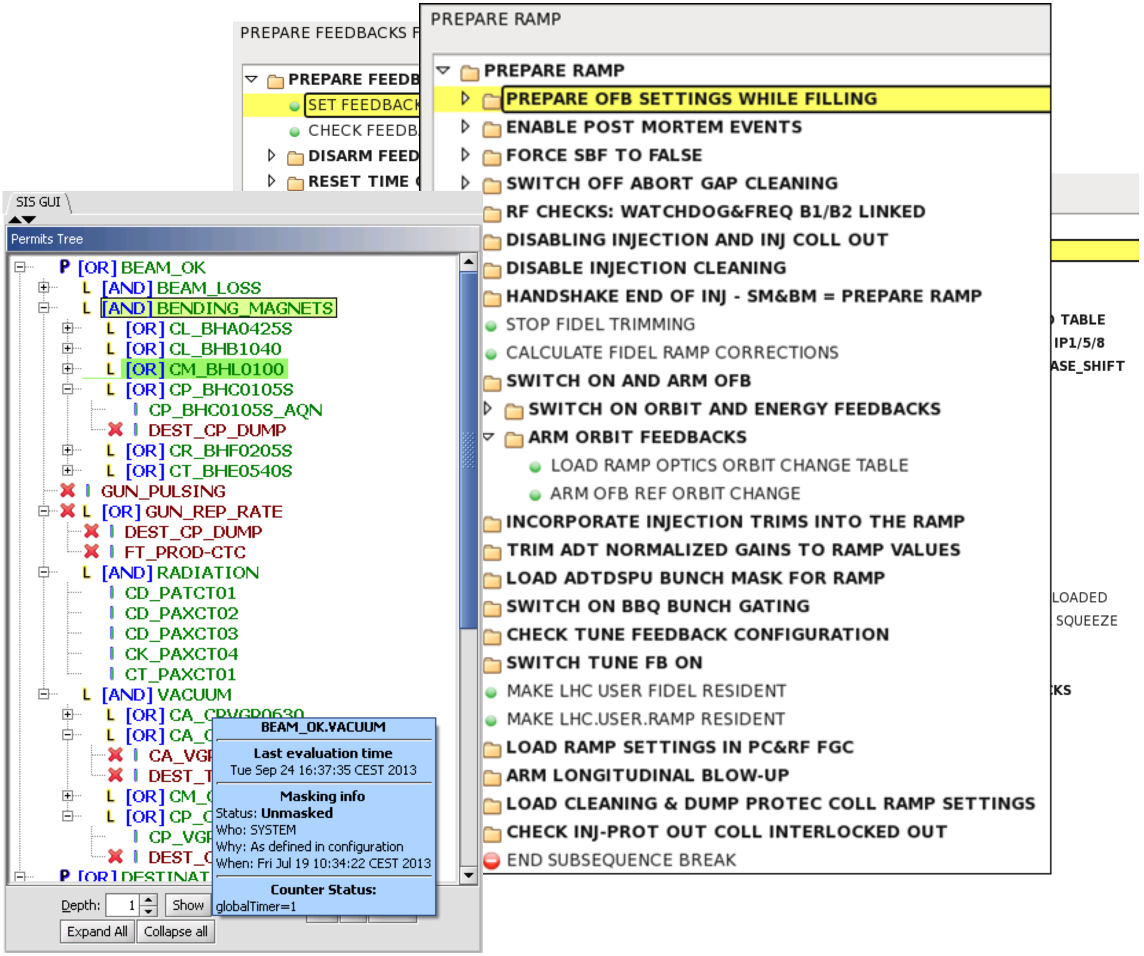


More Control System Requirements

- **Safety for machine protection & operational availability**
  - **Minimum: Machine interlock to protect the hardware**
  - **Ideal: High-level fast-reaction interlocks and role-based access to prevent the wrong action at the wrong time**

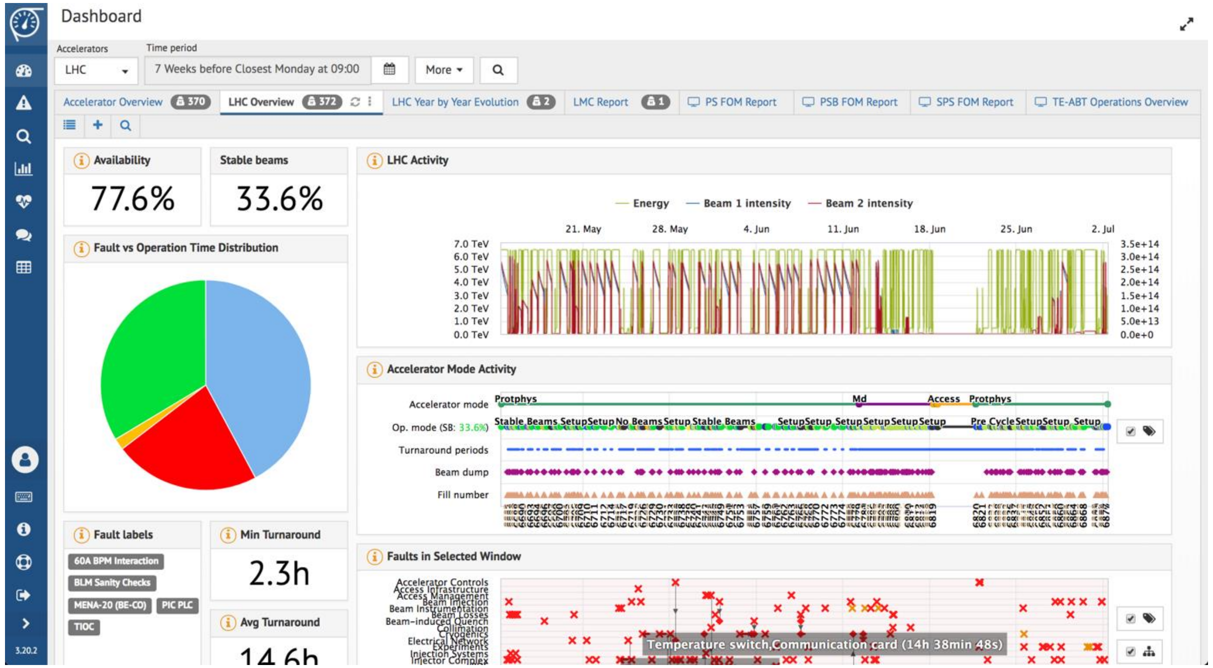
More Control System Requirements

- Automation
  - Generate initial values, play sequences, feedback loops, etc.
  - Minimum: Non-interactive scripts
  - Ideal: Model-driven generation, flexible sequencer (almost like a debugger), automated actions (decision tree, machine learning)



More Control System Requirements

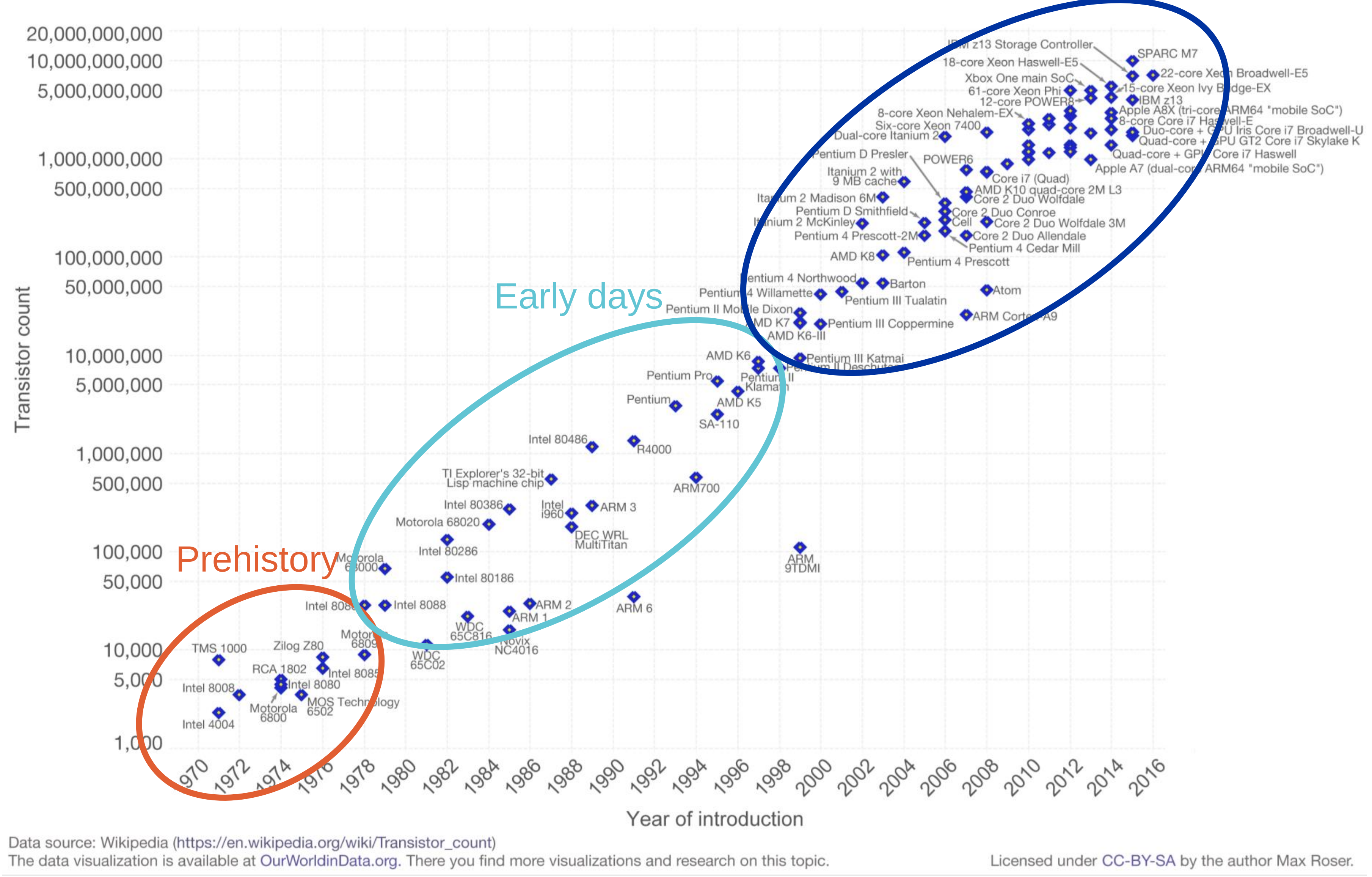
- **Diagnostics**
  - **Detection, identification, and follow-up of problems in the controls infrastructure**
  - **Minimum: Non-interactive status screens**
  - **Ideal: Online monitoring, remote interventions (e.g. power cycle), failure prediction (Machine Learning), analysis tools**



And many more…

A bit of History

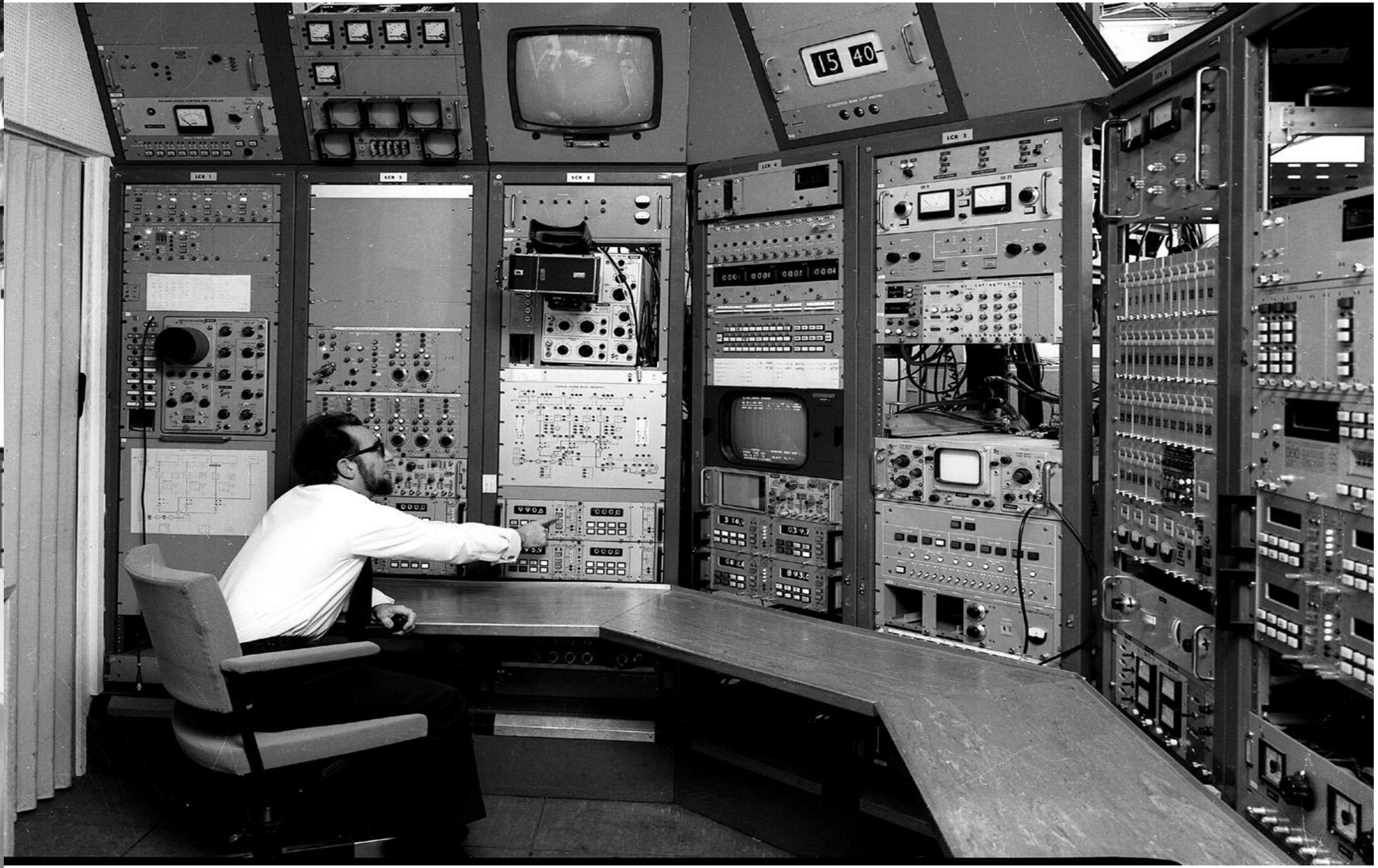
Moore's Law


Modern days
Early days
Prehistory
Transistor count
Year of introduction
Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.
Licensed under CC-BY-SA by the author Max Roser.

Control System Prehistory

- **Accelerators are small and overall less complex (e.g. no superconducting magnets)**
  - No more than a few thousands of devices to control
- **No computing infrastructure and limited possibility to model**
- **Actuator and monitors are physically in the local control rooms (e.g. buttons, knobs, analogue oscilloscopes, etc.)**

Control System Prehistory



The Early Days

Beginning of remote controls

- **Still limited by the available performance**
- **Lack of standards and common frameworks → more DIY and custom solutions**
- **Emergence of several controls solutions, aiming at different types of accelerators (at first)**
  - EPICS (driven by US labs),
  - Tango (driven by ESRF (Fr) – synchrotron light sources)
  - CERN[1]

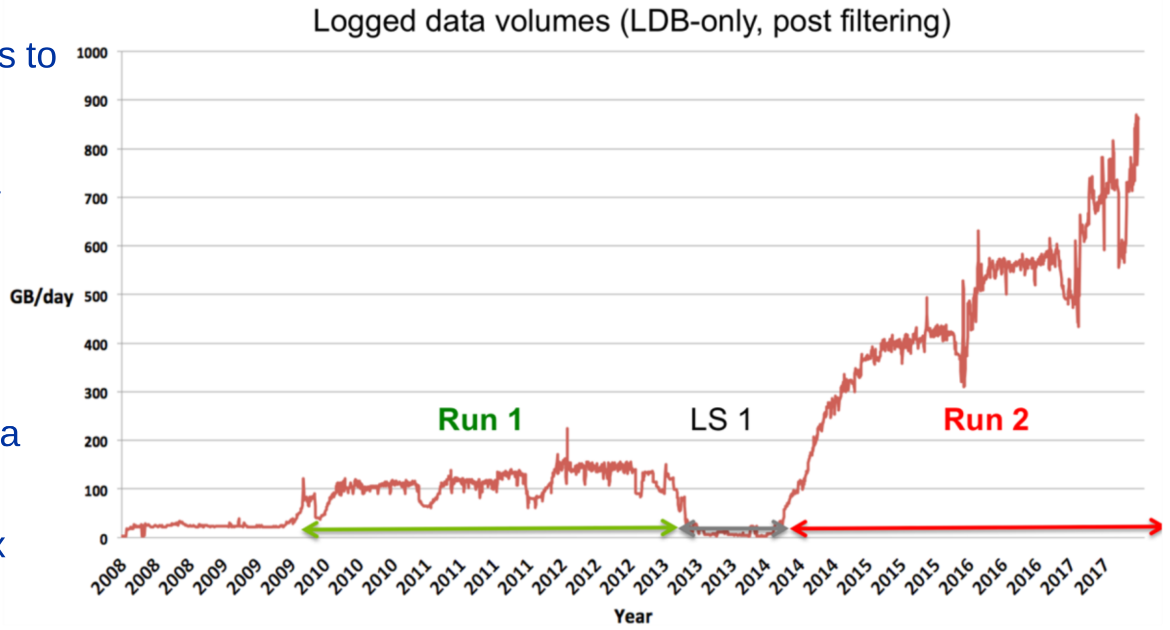
[1] non-exhaustive list

Modern days

- **Hardware has become powerful**
  - E.g. embedded systems at CERN in late 90s had **64 MB of RAM**; Nowadays, they have **8 GB**
  - Most of the needs are covered
  - Yet, users want more and more data (turn-by-turn acquisitions, big-data solution for the long-term storage, etc.)

- **Software industry has become a major actor worldwide.**
  - We can rely on many readily available technologies that open the doors to much more powerful systems

- **We still need to integrate and customise them to the very specific domain of particle accelerator controls**
  - Not all solutions are appropriate;
    Need to remember accelerator controls ≠ selling plane tickets
  - Mastering the different solutions with their evolution, limitations, etc. is a major challenge
  - The rhythm of updates is no longer under our control. E.g. recent Linux CentOS changes

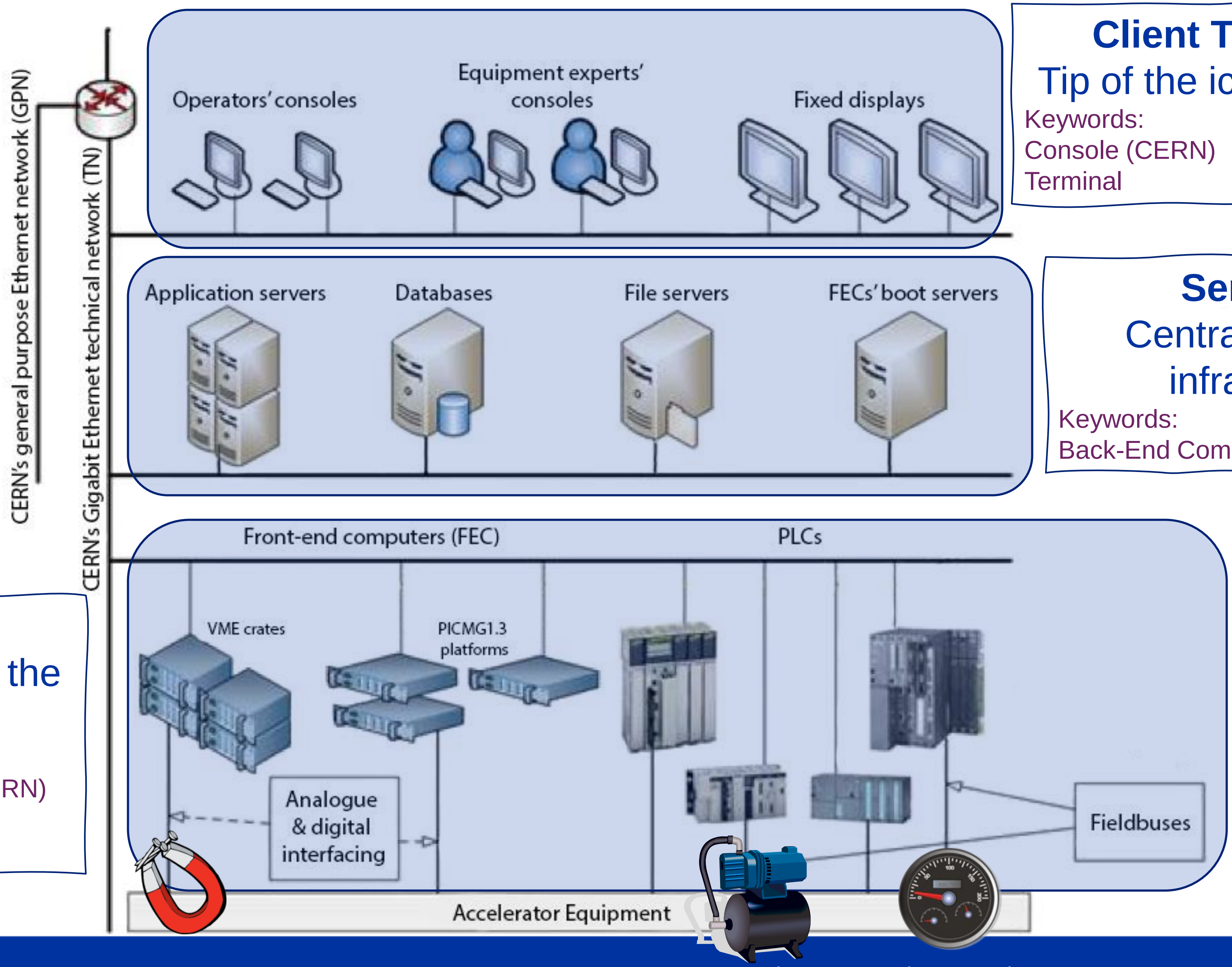


50 years of technology evolution

High-Level Architecture

of a modern control system

High-Level Hardware Architecture



Client Tier
Tip of the iceberg
Keywords:
Console (CERN)
Terminal

Server Tier
Central computing infrastructure
Keywords:
Back-End Computer (BEC) (CERN)

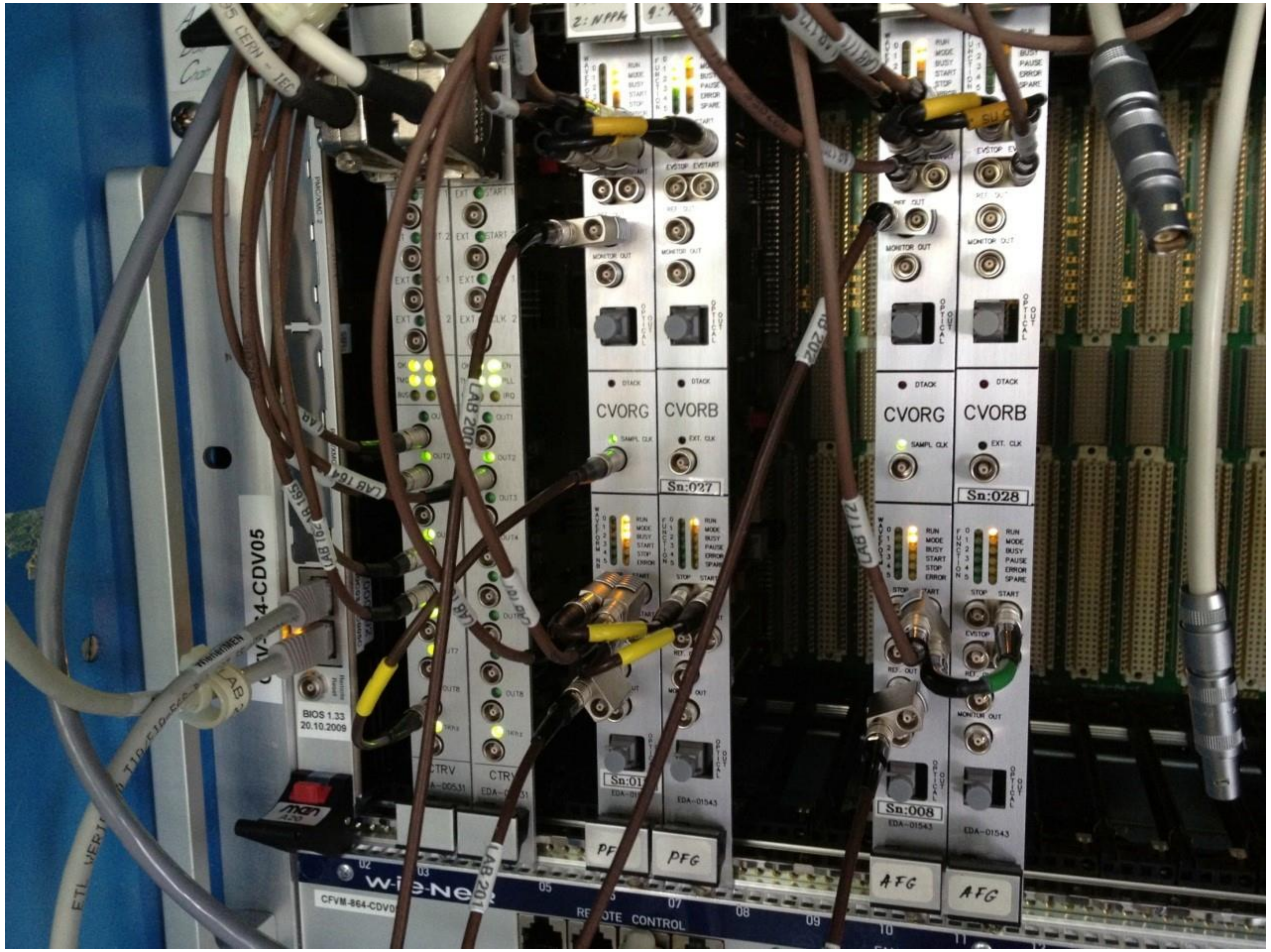
Resource Tier
Electronics close to the accelerator
Keywords:
Front-End Computer (FEC) (CERN)
I/O Controller (IOC) (EPICS)
Device Server (Tango)

High-Level Hardware Architecture

The CERN Accelerator School

Resource Tier

- **Open enclosures**
  - Easy access
  - Better cooling and power available
  - But expensive
- **Closed enclosures**
  - Possible for simple functions (e.g. fieldbus control)
  - Cost effective; when deployed in big number, e.g. LHC power converter control gateways

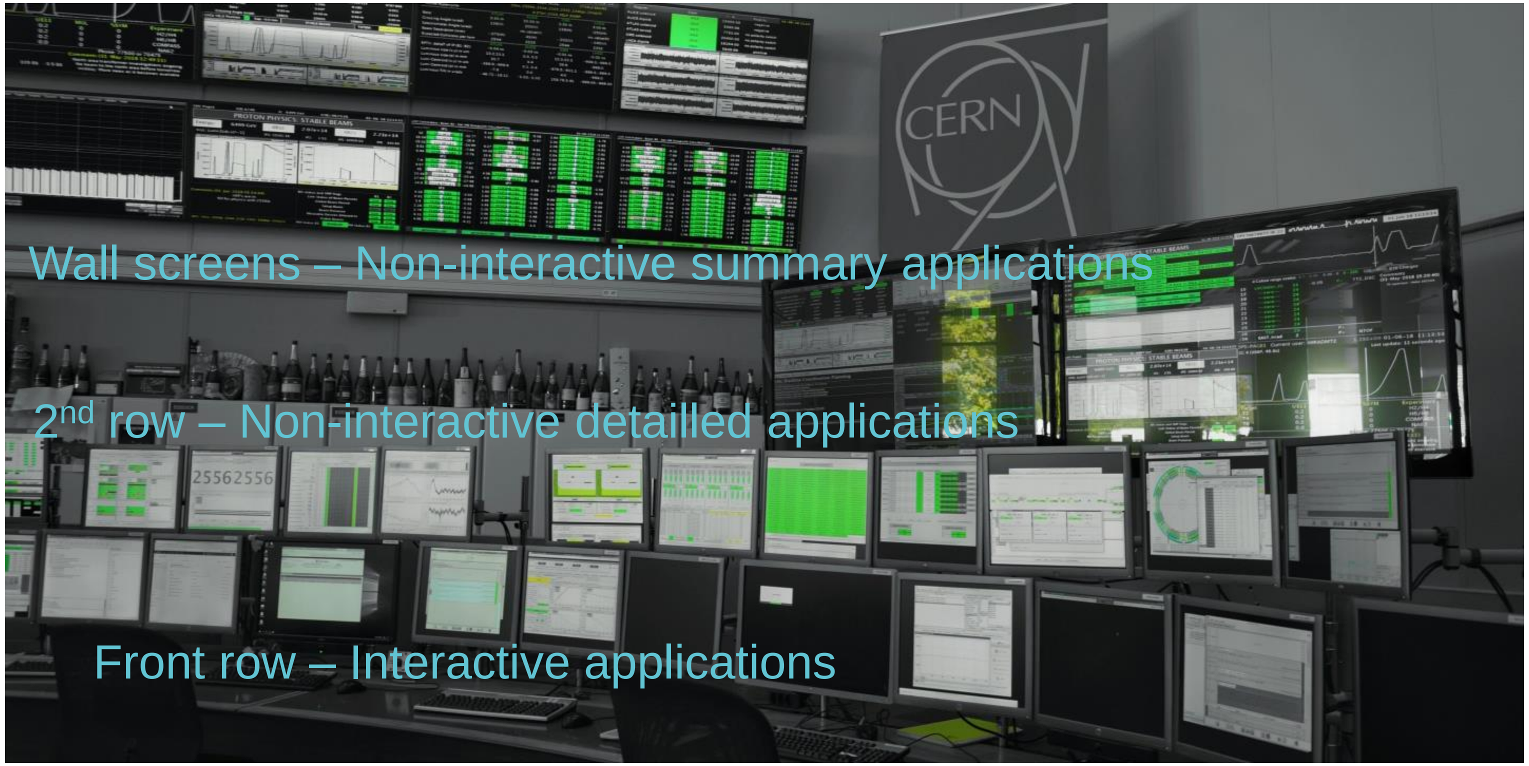


High-Level Hardware Architecture

Middle Tier

- IT Computer centre type of hardware
- High-density
- Highly available (redundancy and hot-swap)

High-Level Hardware Architecture



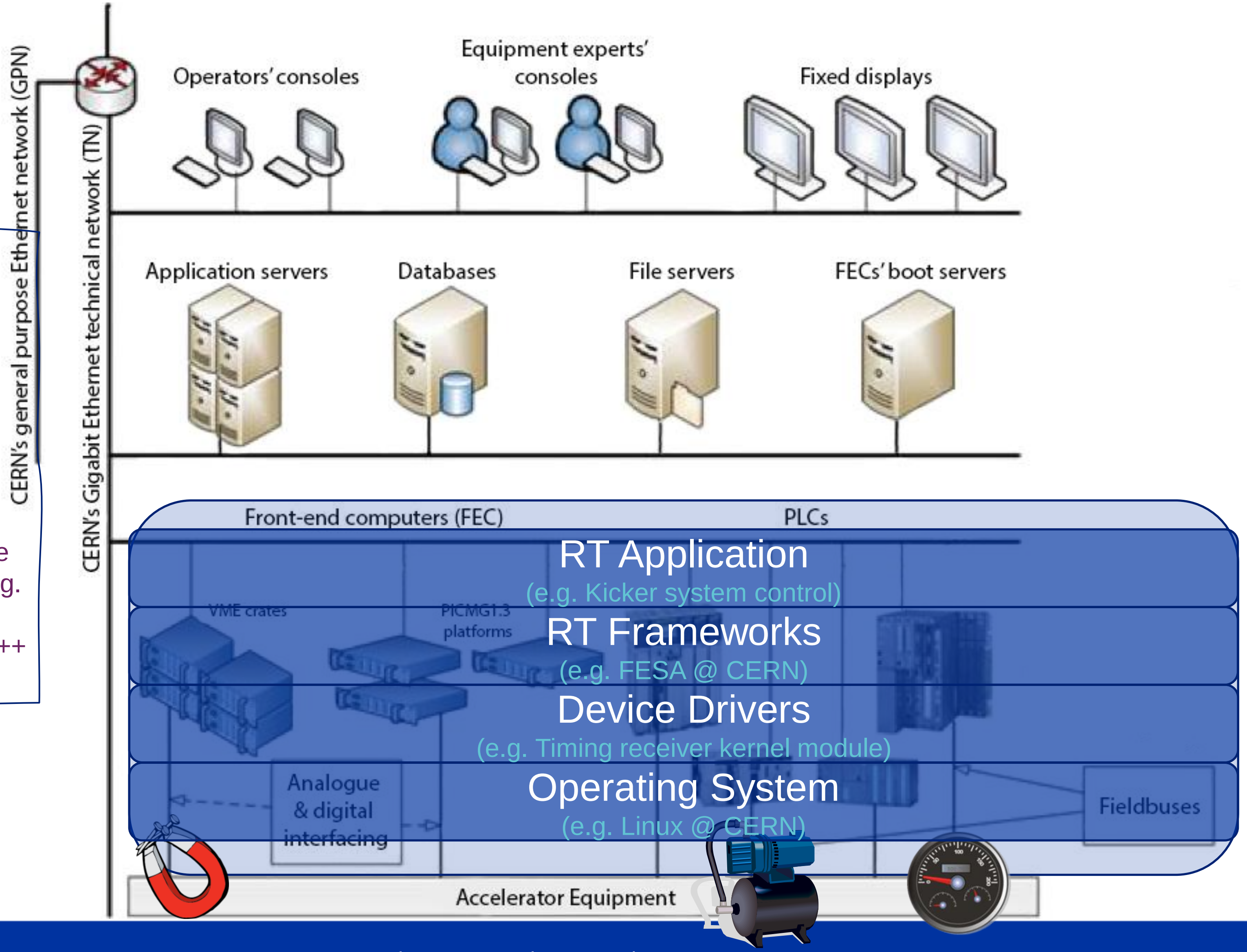
Control Room Computers

- As much as possible COTS desktop PCs but MTBF requirements might be difficult to satisfy
- Users expect modern reactive GUIs
- Several layers of screens to have as much data as possible available

High-Level Software Architecture

Front-end Tier
Real-time control and acquisition

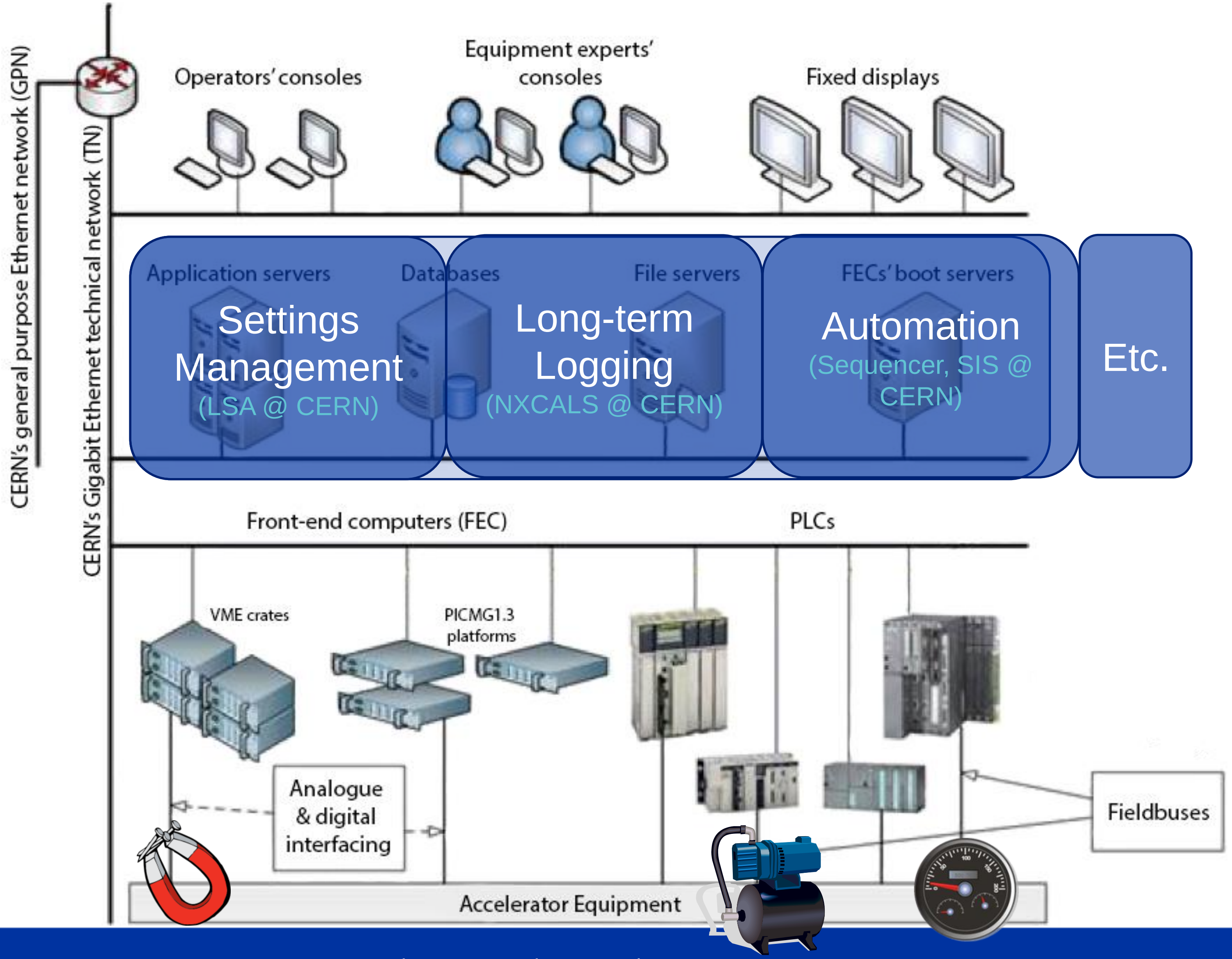
- Limited, local scope
- Fast reaction possible (interrupts)
- Limited computing power (compared to other tiers)
- Equipment processing to provides a high-level view of the hardware
- Real-time (RT) applications relies on frameworks, which capture the recuring aspects (react to events, publish new data, etc.) E.g. FESA @ CERN, POGO with Tango
- Based on technologies closed to the hardware (C for drivers, C++ for RT, etc.)



High-Level Software Architecture

Business Tier
General purpose services & Specific business logic

- Broader scope; able to coordinate the entire accelerator
- Powerful computers
- Less reactive (network) and at a higher-level of abstraction
- Based on technologies that are better suited for high-level business logic (e.g. Java)



High-Level Software Architecture

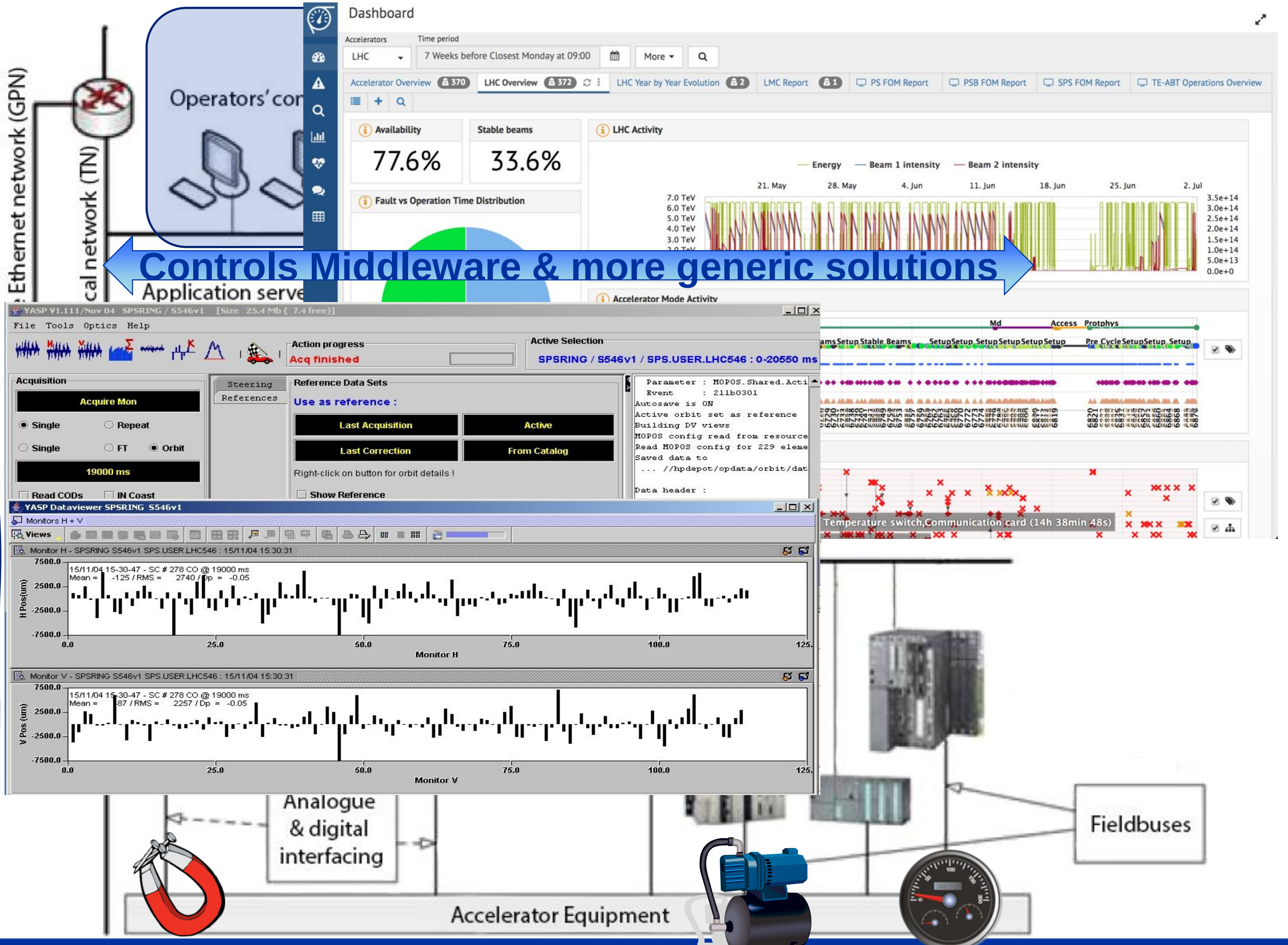
Presentation Tier
Graphical applications

Different technologies available
- Python with PyQt
- Web ecosystem (**Angular**, View.js, etc.)
- Java Swing, Java FX

Keywords:
Graphical User Interface (GUI)
Human-Machine Interface (HMI)
Command-line interface (CLI)

Communication

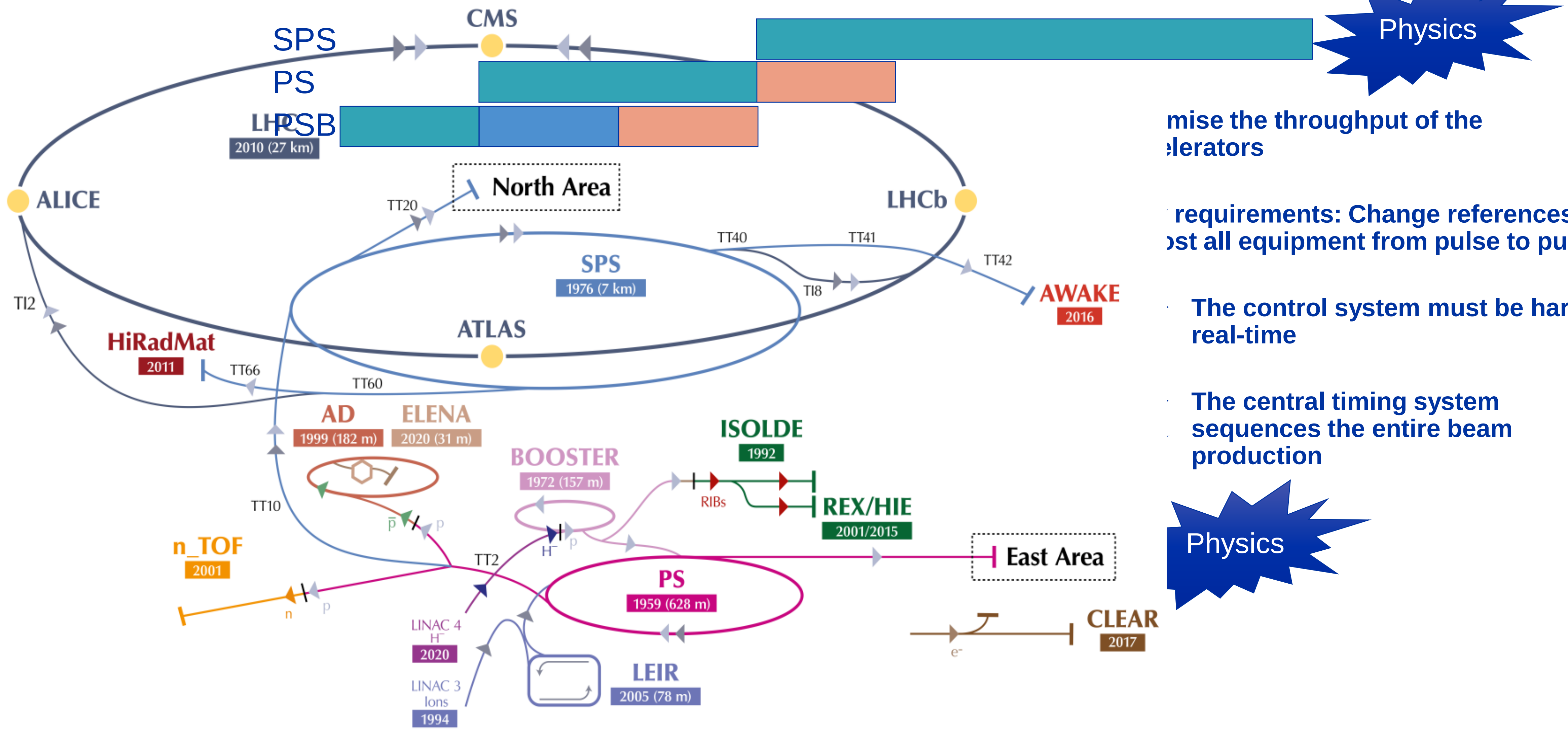
- Accelerator-specific protocols for the lower layers
  - Channel Access (EPICS)
  - CMW (CERN)
- Potentially, more generic technologies for the higher layers
  - RMI/JMS
  - REST API
  - gRPC
  - …



A few examples

And the key components used…

Accelerators Optimisation



mise the throughput of the
lerators

requirements: Change references of
st all equipment from pulse to pulse

- The control system must be hard real-time
- The central timing system sequences the entire beam production

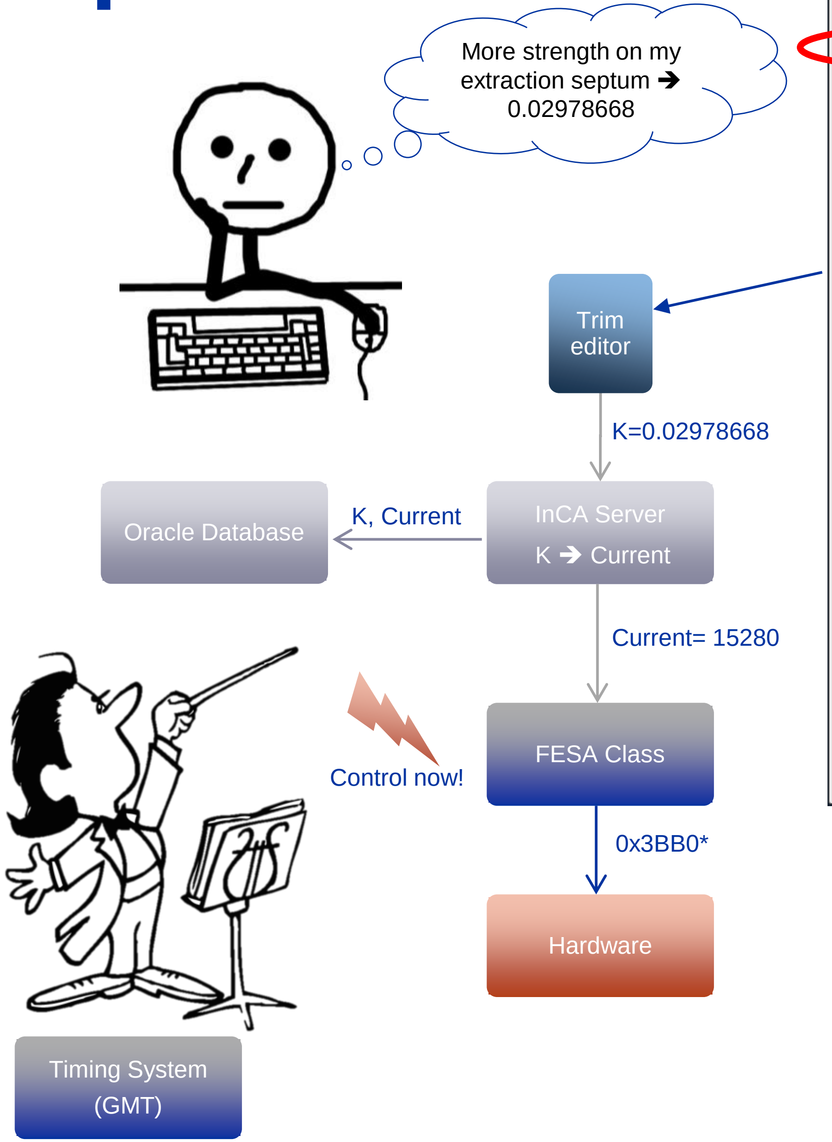
Physics

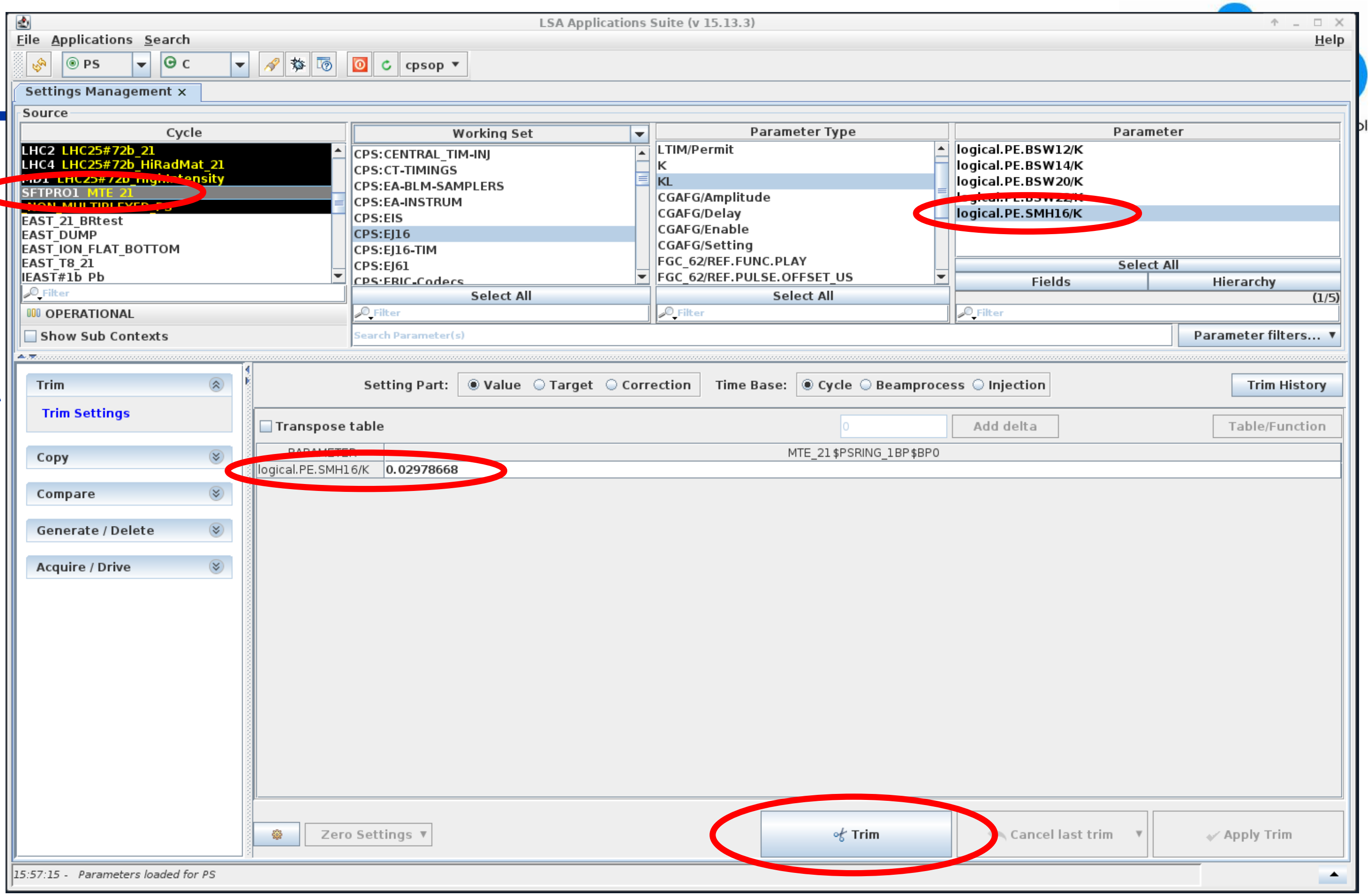
Use Case 1: Control

- **Who:** PS operator
- **What:** Change the strength of the extraction septum towards the SPS for the Fixed Target beam
- **Involved controls components:**
  - InCA/LSA (Setting management)
  - CMW (Controls Middleware)
  - FESA (Real-time hardware control)
  - Timing (Synchronisation)

In all the examples, we assume all the configuration is already done and we focus on the run-time aspects

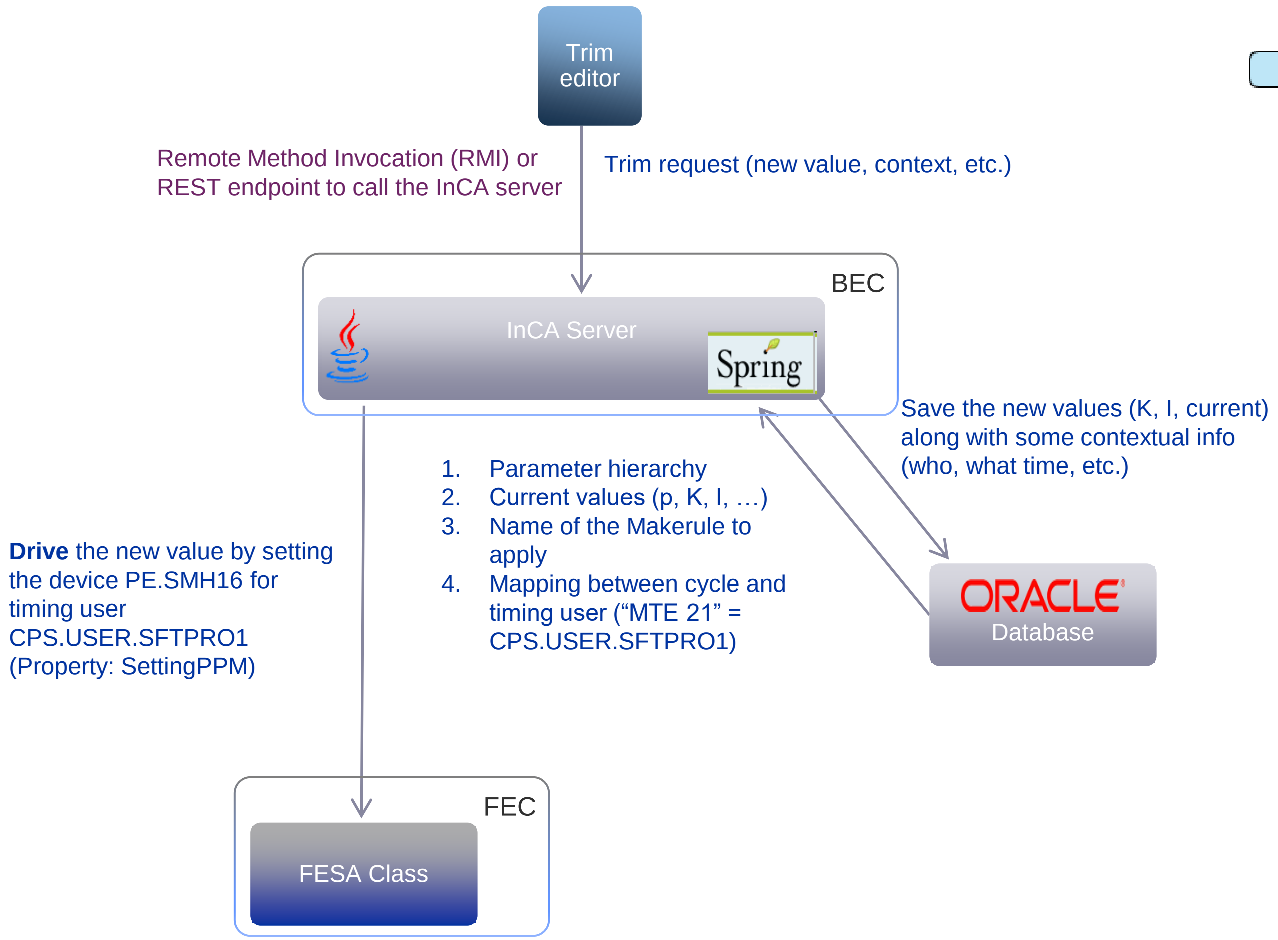
Example 1 – Control –

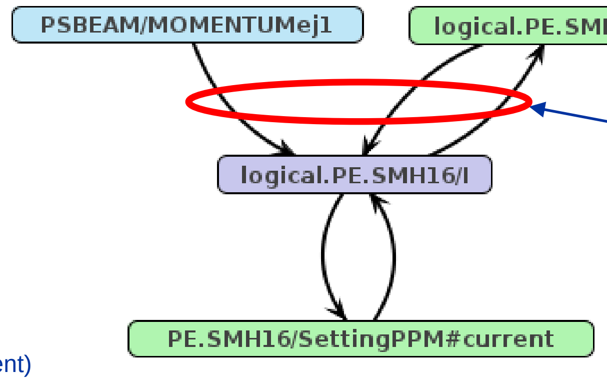




*not a real value

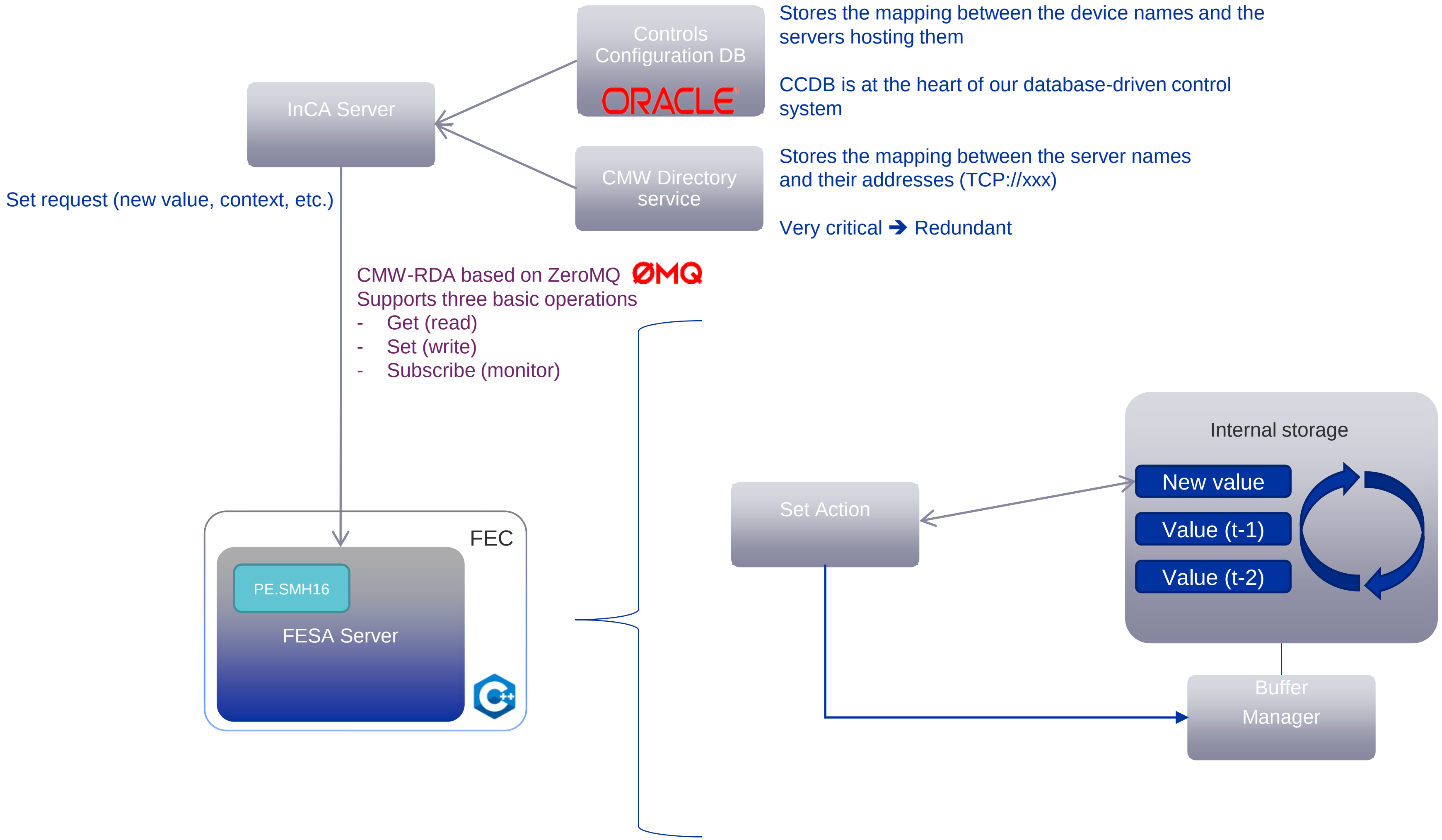
Example 1 – Control - Communication



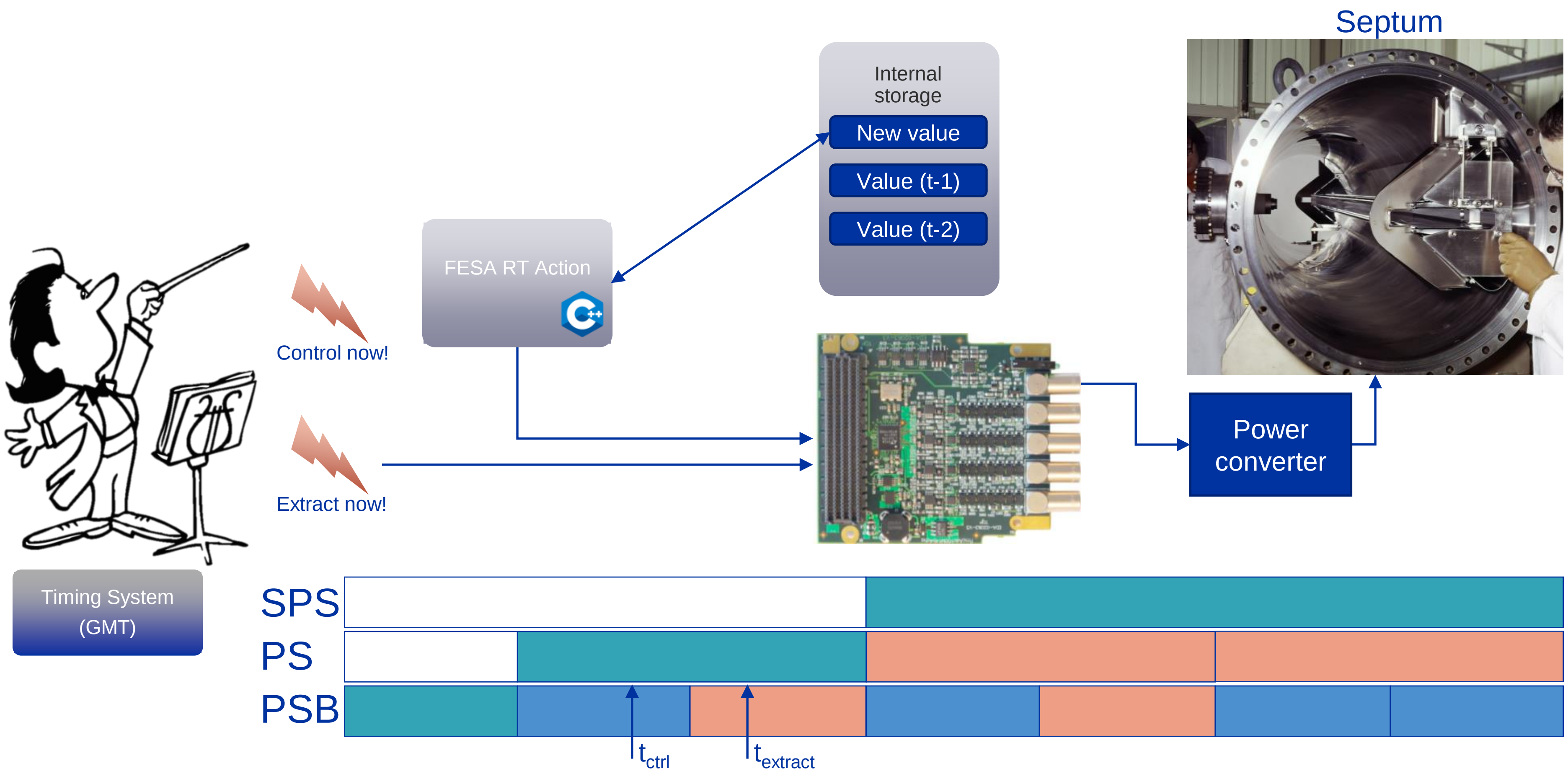


Makerule: Java class to compute parameters based on other parameters, hardware characteristics (calibration curve) and cycle specific values

Example 1 – Control – Low-level



Example 1 – Control – Beam production

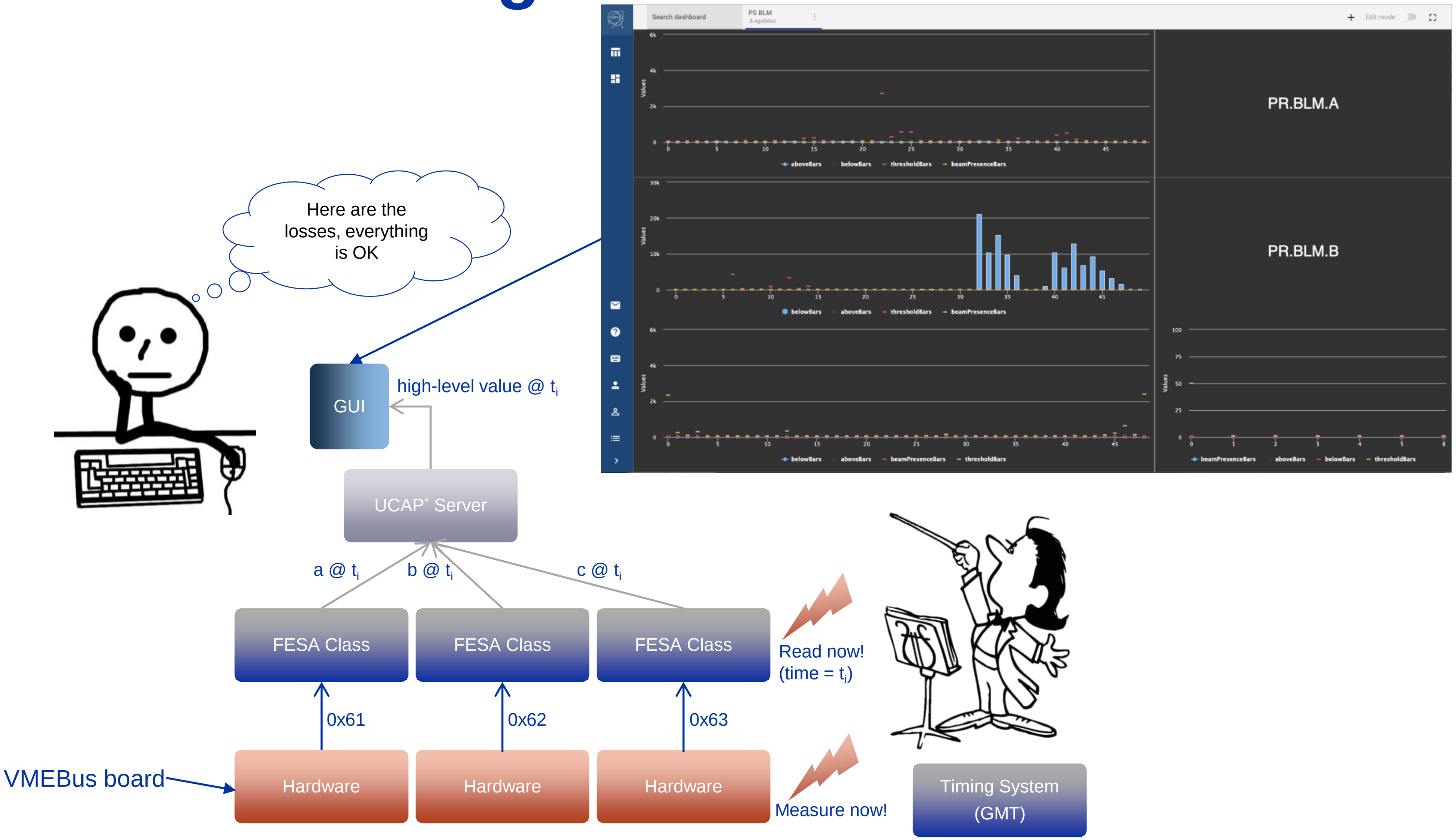


Use Case 2: Acquisition

- **Who:** PS operator
- **What:** Keep an eye on acquisition values of many control devices. The low-level data needs post-processing and will be displayed as a graph in a web page
- **Involved controls components:**
  - Timing (Synchronisation)
  - FESA (Real-time hardware control)
  - CMW (Controls Middleware)
  - UCAP (Unified Controls Acquisition & Processing)
  - WRAP (Web Rapid Application Platform)

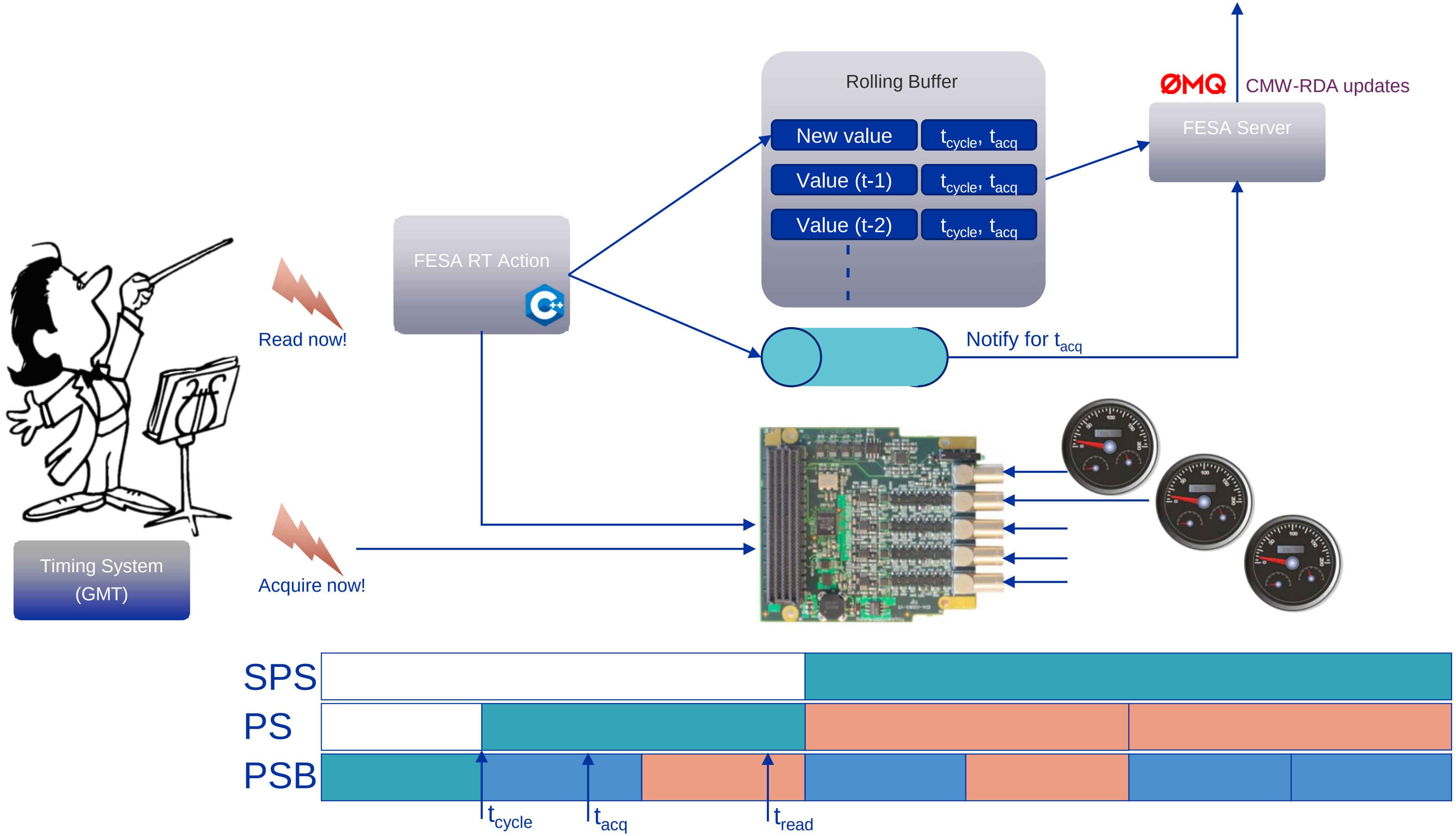
In all the examples, we assume all the configuration is already done and we focus on the run-time aspects

Example 2 – Monitoring

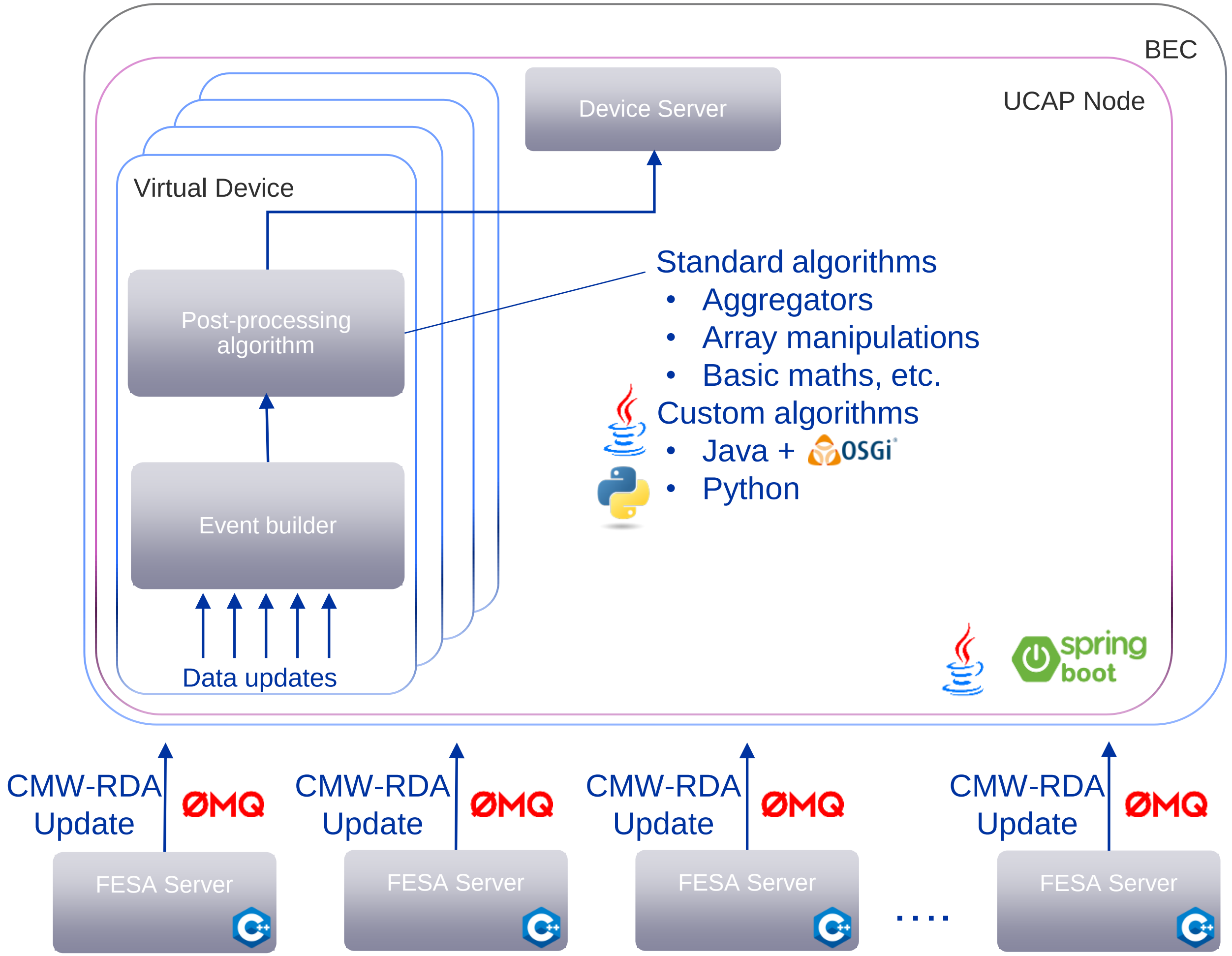


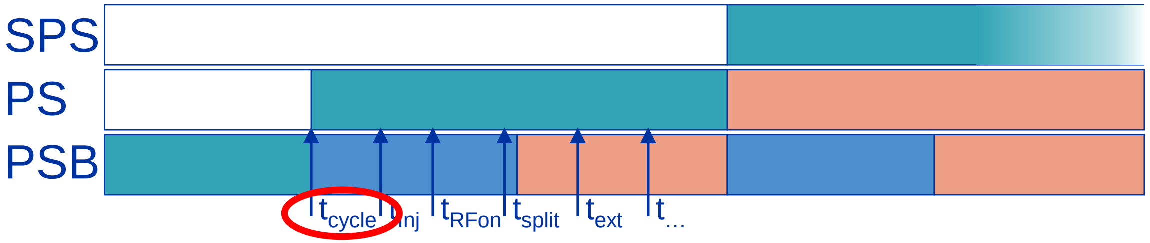
*UCAP: Unified Controls Acquisition & Processing framework

Example 2 – Monitoring – Low-level acquisition



Example 2 – Monitoring – Post-processing

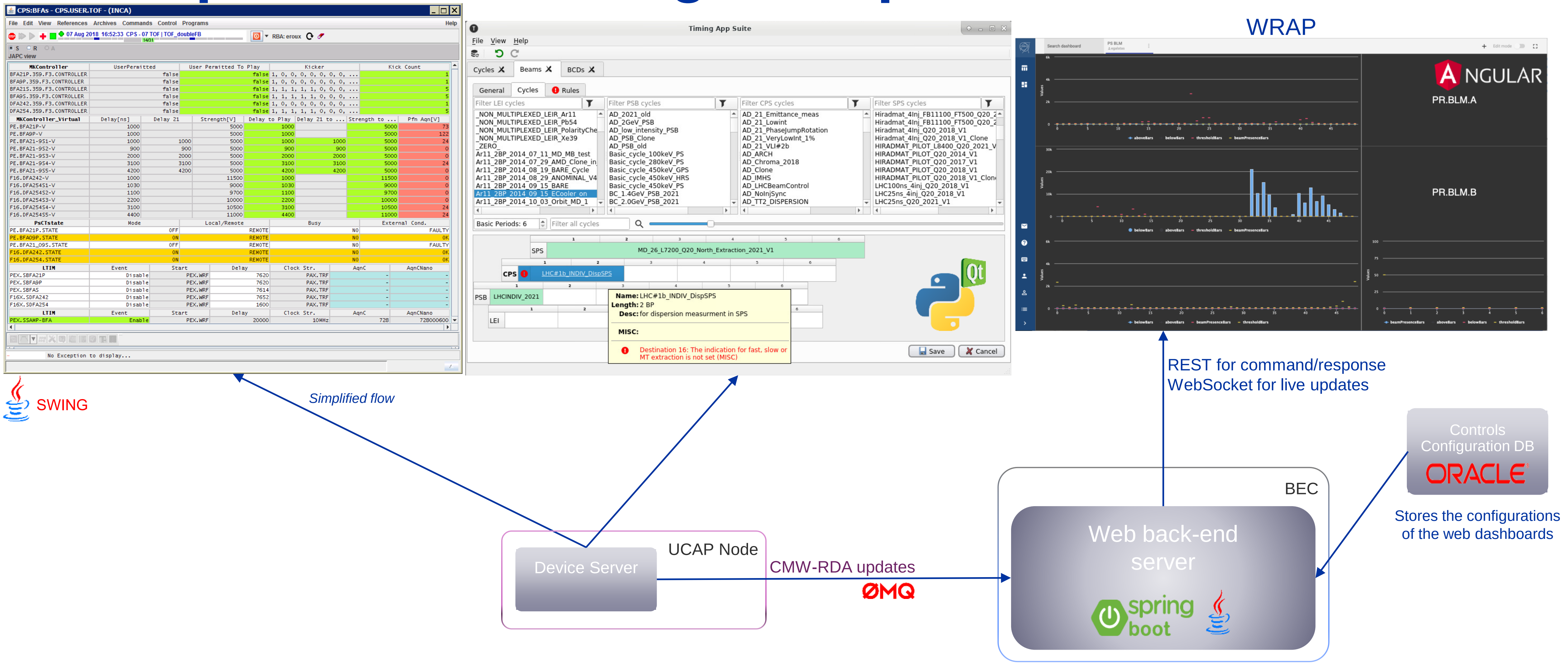




- How to group the incoming data?
  ➔ Start Cycle timestamp (AKA cyclestamp)
- When to trigger the post-processing?
  ➔ Once all the data is there or after a time-out
- How long to wait for late comers?
  ➔ Configurable time-out
- What to do if no data is published?
  ➔ Out-of-the-box monitoring

Example 2 – Monitoring – Diagnostics

Example 2 – Monitoring – Graphical User Interface

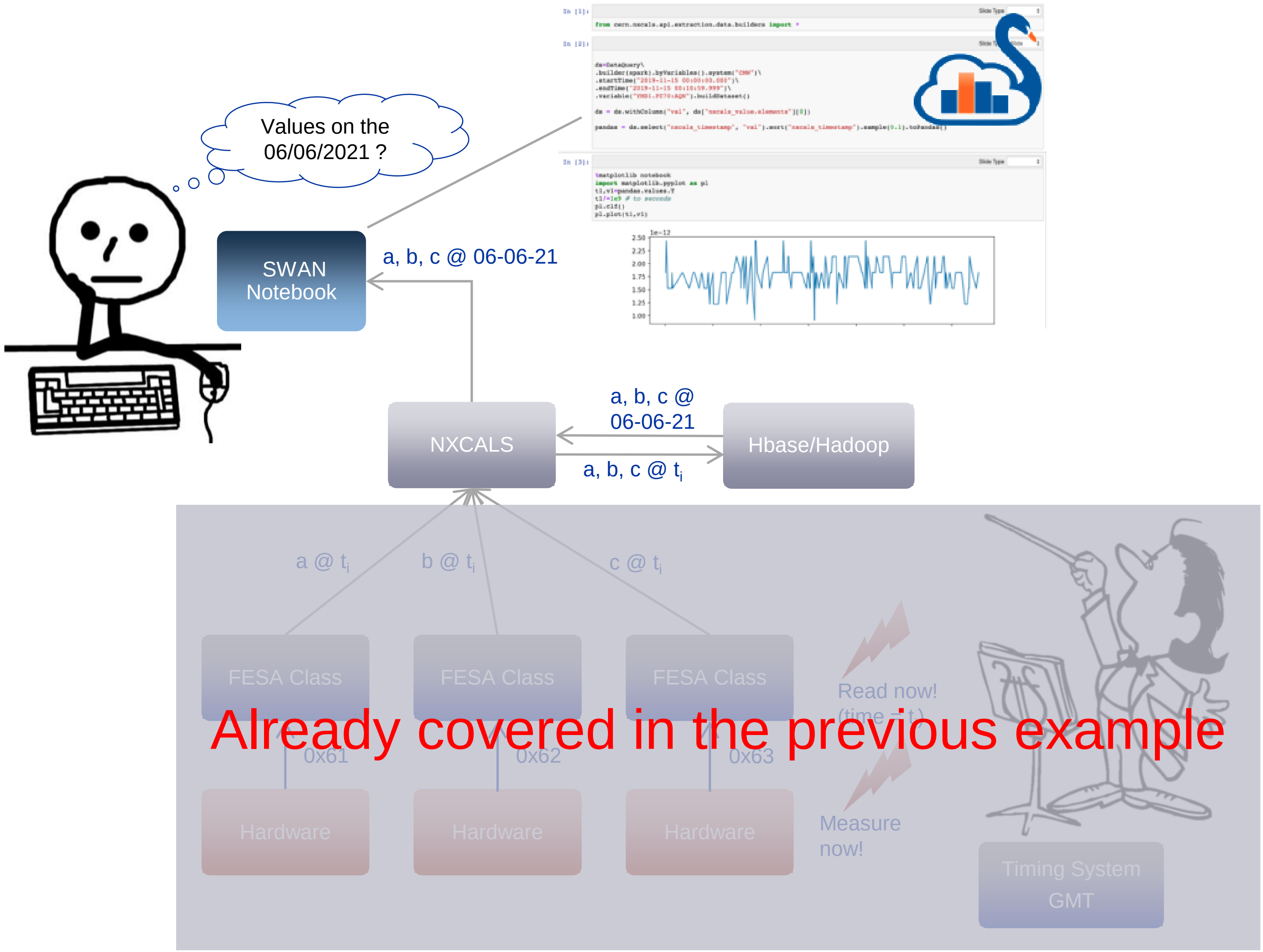


Use Case 3: Logging

- **Who:** Accelerator physicist
- **What:** Store acquisition values of many control devices long-term and perform analysis
- **Involved controls components:**
  - Timing (Synchronisation)
  - FESA (Real-time hardware control)
  - CMW (Controls Middleware)
  - NXCALS (Data Logging)
  - SWAN (Web-based analysis tool)

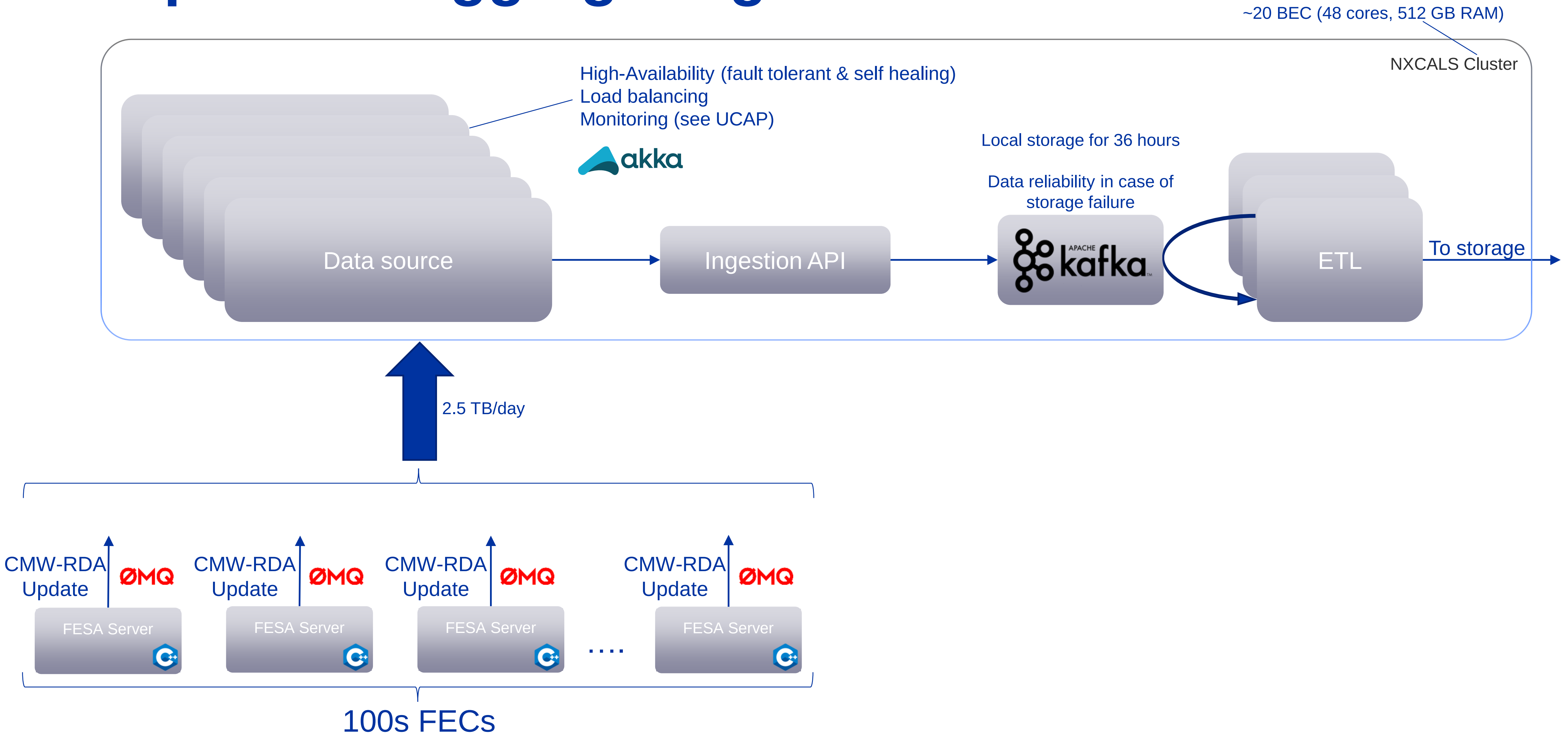
In all the examples, we assume all the configuration is already done and we focus on the run-time aspects

Example 3 – Logging



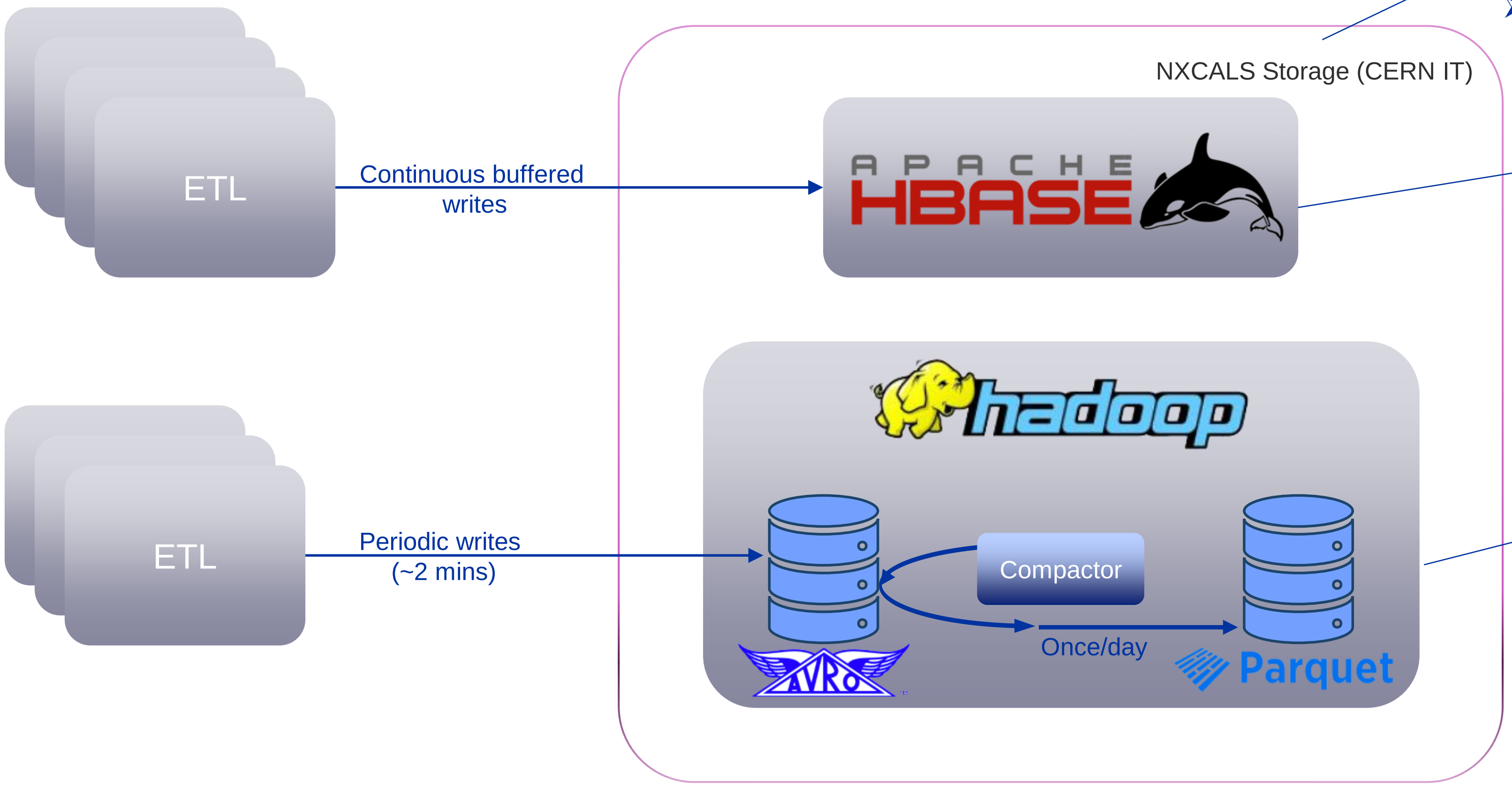
*SWAN: Service for Web-based ANalysis

Example 3 – Logging - Ingestion



Example 3 – Logging - Storage

Nov-2021:
- ~40 servers
- 1.1 PB logged data

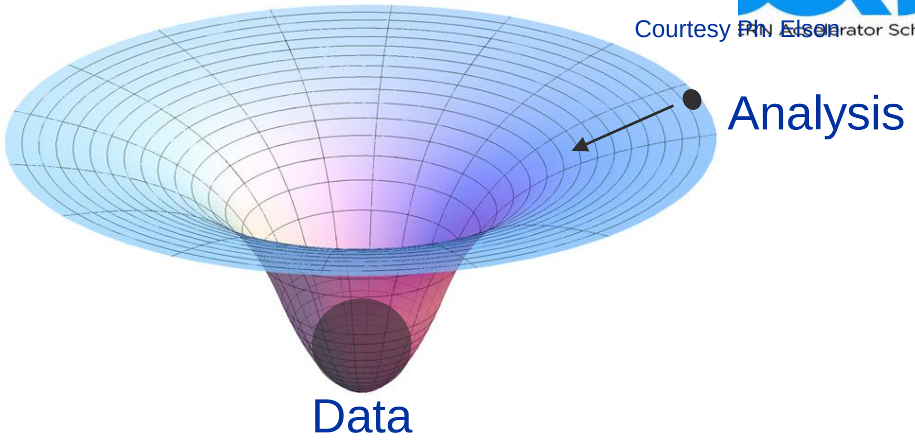


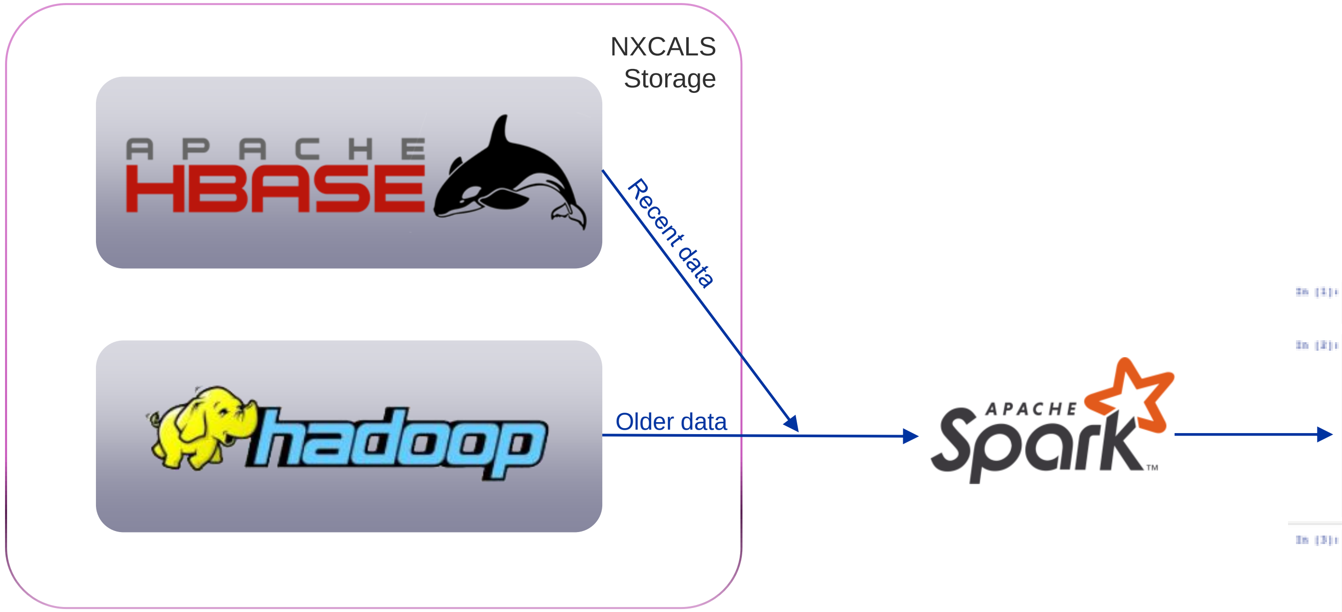
Recent data
~ last 4-5 days

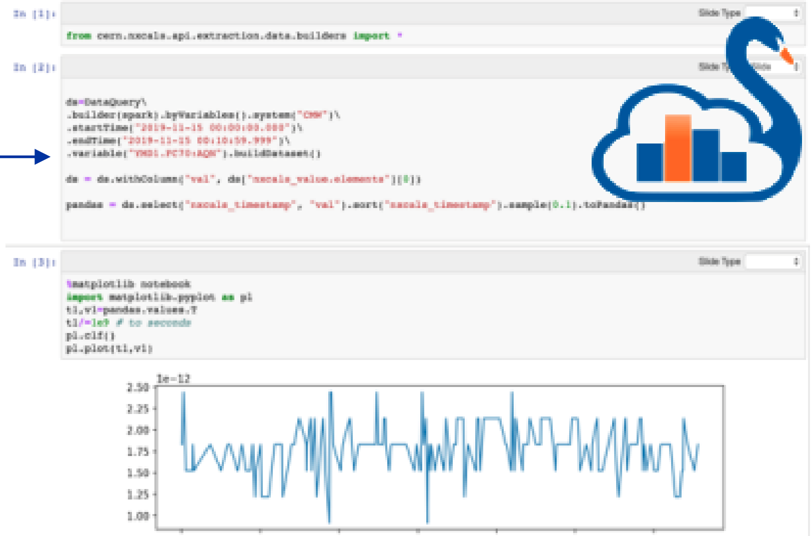
All of the data
after compaction

Example 3 – Logging - extraction

Courtesy Ph. Elson







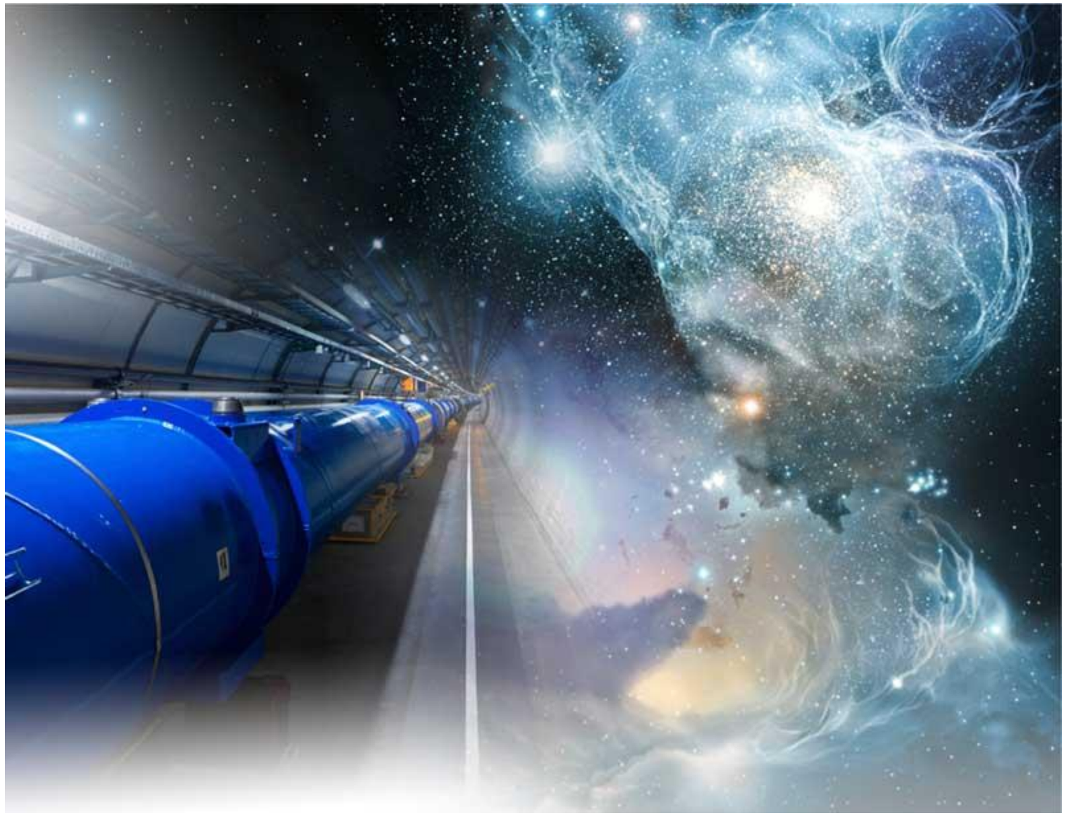
Want to know more?

CERN Beams Department (https://beams.cern/)

Introduction to BE-CO Control System, 2019 Edition, S. Deghaye & E. Fortescue, CERN, 2020. (https://cds.cern.ch/record/2748122)

Tango Controls (https://www.tango-controls.org/)

EPICS (https://epics-controls.org/)



home.cern