Controls changes overview

Philip Elson (BE-CSS)

With thanks to:
R. Alemany Fernandez (BE-OP), V. Baggiolini (BE-CSS), M. Braeger (BE-CSS), L. Burdzanowski (BE-CSS), S. Deghaye (BE-CSS), F. Ehm (BE-CSS), J-C. Garnier (TE-MPE), M. Gourber-Pace (BE-CSS), V. Kain (BE-OP), S. Page (BE-CSS), C. Roderick (BE-CSS), I. Sinkarenko (BE-CSS), A. Stanisz (TE-MPE), G. Trad (BE-OP), M. Vanden Eynden (BE-CSS), J. Wenninger (BE-OP), J. Wozniak (BE-CSS)

24th November 2021

Introduction

Controls has seen significant change throughout.

Wherever practical, it has been backwards compatible.

Based on workshop session chair requests, topics covered:

- Development tools
- GUI strategy
- Logging service
- Post-mortem service

Changes in controls

Control system must be:

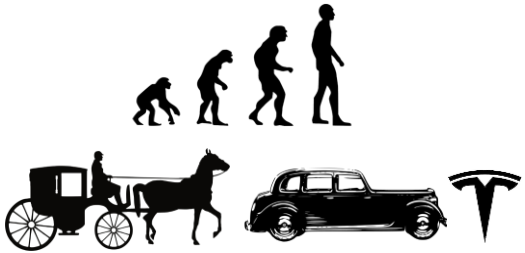
- Fit for purpose
- Adaptable
- Reliable
- Robust

Some key values:

- User friendly
- Backwards compatible where possible
- External technology exposure managed
- Maintainable

Drivers of change:

- Response to the evolving needs of **operations / equipment experts**
- Adapting to **external technology changes**
- **Consolidating** technical debt



Always mindful:
Changes in controls can have a major impact across the sector

Development languages & tools

LS2: Major migration from Java 8 to Java 11 (breaking change)

Scope: most applications and services from Run 2

→ **Smooth transition for operations in 2021**

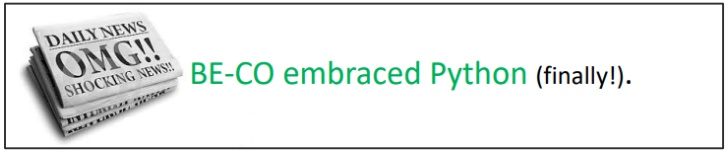
Development modernisation:

- SVN to Git (and Gitlab)
- Devtools (CBNG 4) consolidation
- Acc-Java user meetings

Coordinated controls updates of third-party dependencies.
Ready *very* early in the LS, offering stronger JAR compatibility.

Response to feedback from D. Jaquet & D. Cotte "BE-CO LS1 review: View from BE-OP"

Development languages & tools



C. Roderick 9th LHC Evian Workshop

We've come a long way!

- Fully supported since 2019 in A&T sector
- Foundations, infrastructure and rationalisation
- Development of controls libraries adopted from across ATS
- Creation of new controls libraries to meet user needs

PyJapc
Fundamental device access

PjLSA
API for LSA settings management

PyCCDA
API for Controls Configuration

accwidgets
Common PyQt CO widgets

PyTimber
Access to logged data

PyLogbook
Read and write from eLogbook

PyRBAC
Access control token handling

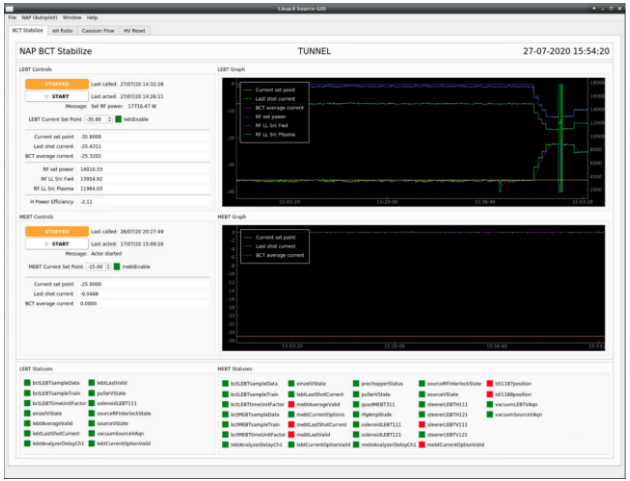
Even more...

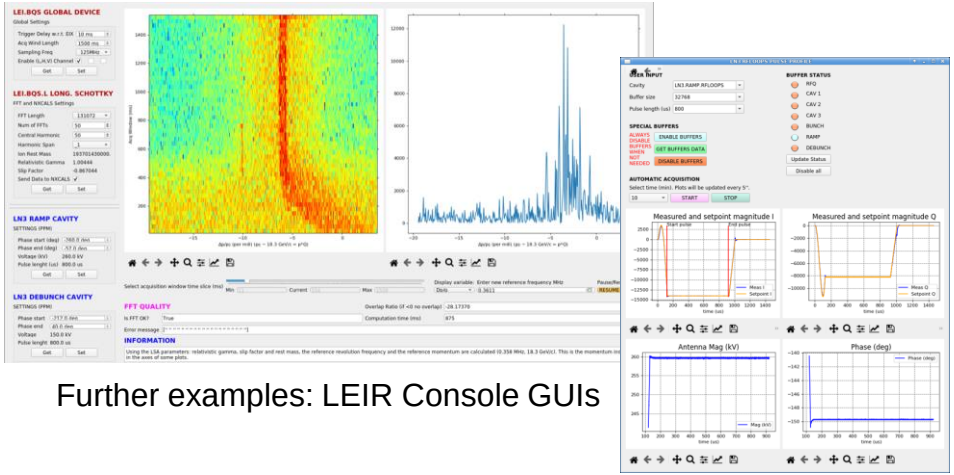
P. Elson et al., Introducing Python as a supported language for accelerator controls, ICALEPCS'21

Python apps in operations

An example is Linac4 Source Autopilot*:

- Python GUI based on PyQt
- UCAP processing in Python





Further examples: LEIR Console GUIs

Regular Acc-Py user meetings

Over 400 users across the sector in 2021

Deployment tools

2019: Python app launch temperamental

2020: Provision of app deployment tool:
common approach, integrated into Console Manager, robust startup

* Collaboration between BE-APB & BE-CSS

GUI Strategy

Number of applications ↑ (*well* over 500), and accelerating

Each application needs to be adapted to change manually (potentially during operation)

Technology evolving at increased pace ⟳

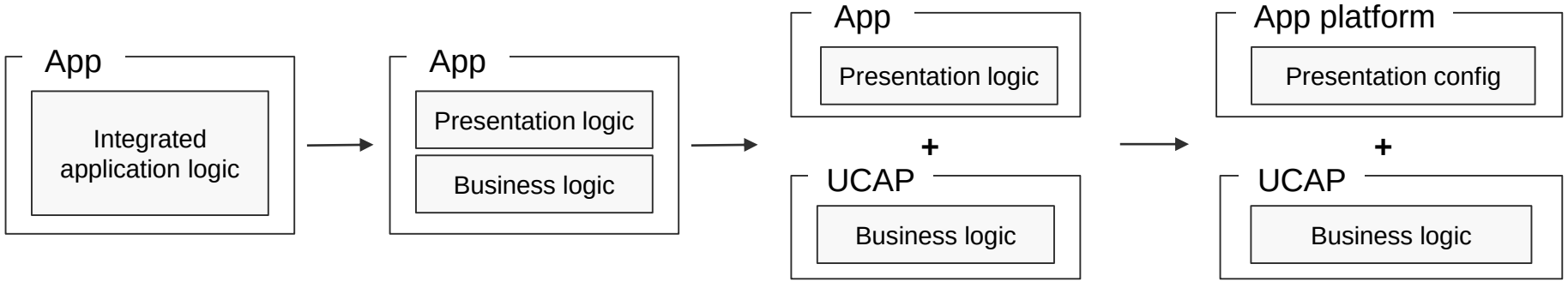
Goal:

Minimise the total maintenance cost of applications across the sector.

GUI Strategy

Strategy:

Reduce the *number* and *complexity* of manually maintained applications as much as possible.



GUI Strategy: Application platform

Key features:

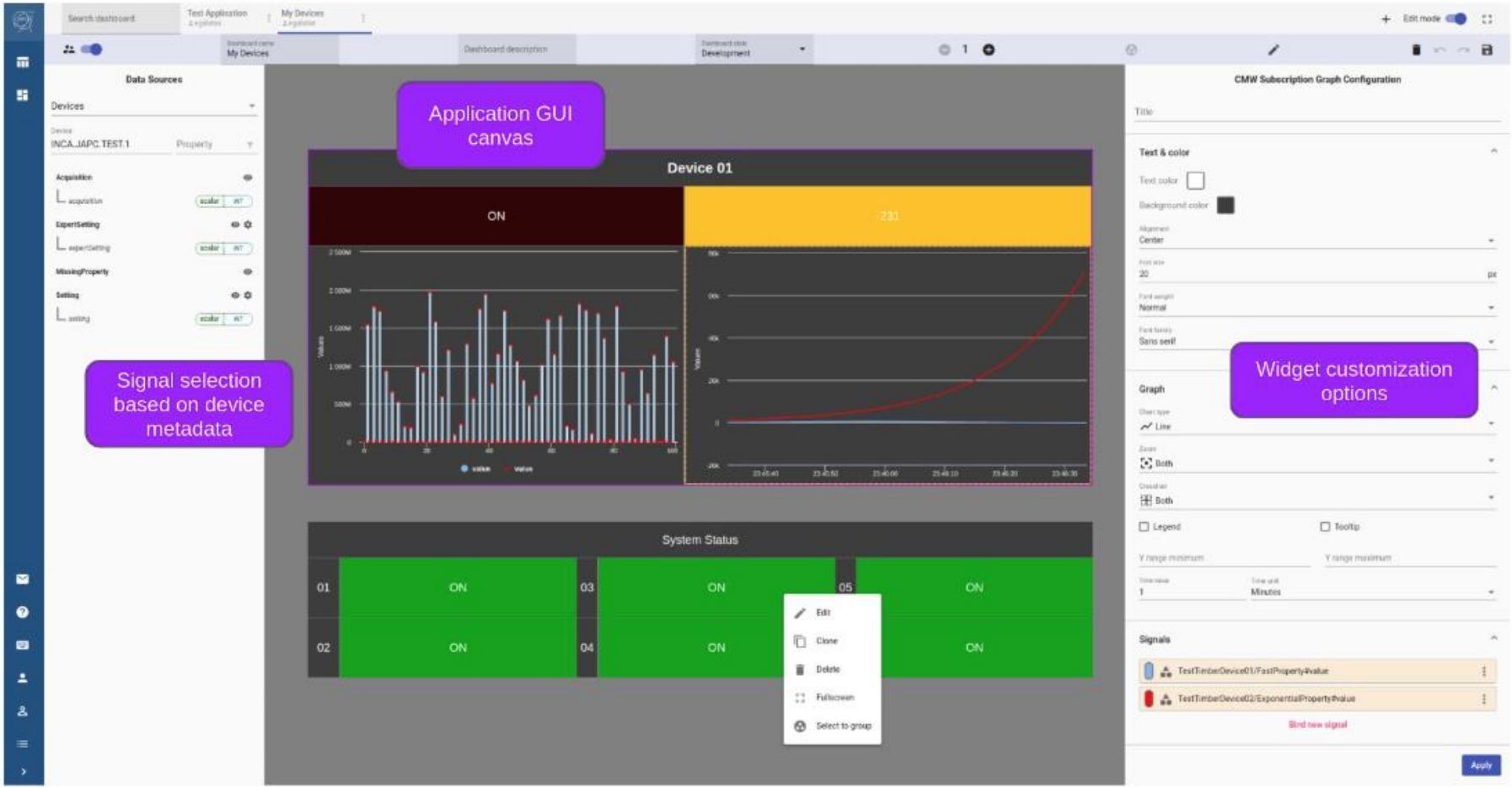
- Integrated with Control System lifecycle, with automatic migrations
- Zero-code applications, shielding users from inevitable GUI change
- Leveraging UCAP for data processing

Web Rapid Application Platform (WRAP):

- Centralisation is essential for maintainability
- Web being embraced at many labs (as seen at ICALEPCS'21)
- Improved recruitment prospects
- Working on this as quickly as possible

WRAP over the next 12 months (and beyond):

Targeting the needs of a significant portion of the applications which are manually maintained today.



An early example of the application editor

GUI tools and strategy

Reduce the *number* and *complexity* of manually maintained applications as much as possible.

Still a need for bespoke (code-based) applications:

Swing

PyQt

Applications will continue to need to be maintained and adapted manually

Landscape is evolving:

- WRAP (in development)
- FESA Navigator replacement (in development)
- Bespoke Applications

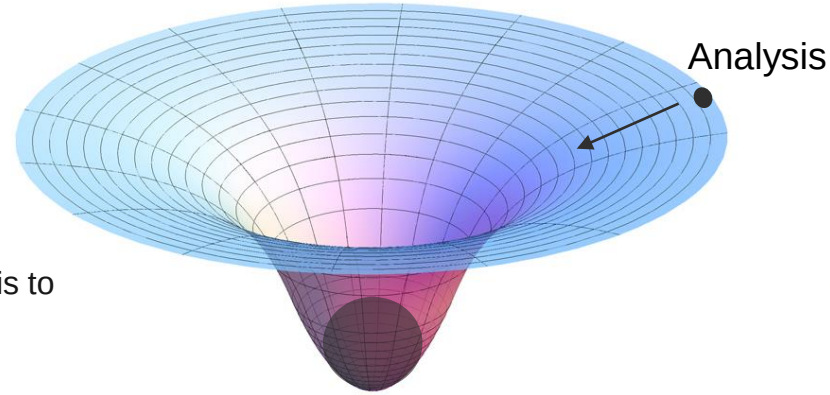
Stay informed through GUI strategy user meetings and CTTB forums

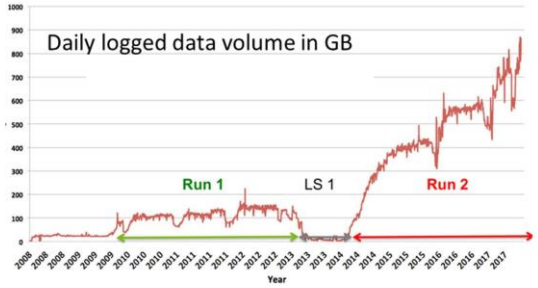
Let us know about your requirements: acc-gui-support@cern.ch

Logging service: Motivation for change

Data growth makes it increasingly difficult to move data around in order to do analysis.

Instead, we need to take the analysis to the data (***data gravity***).





Evolution in volume of CALS logged data reaching over 800GB/day by end of Run 2

NXCALS infrastructure

Event streaming (Device data) → Distributed storage → Data processing engine

→ **NXCALS interfaces**

(Industry standard “big data” platform)

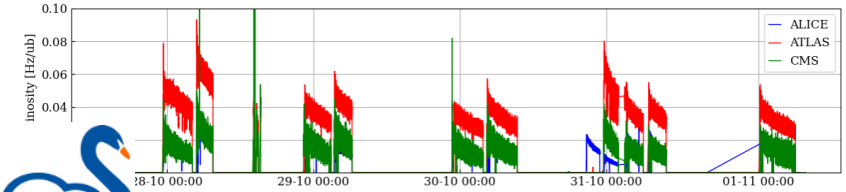
NXCALS interfaces

- Programmatic APIs in Java and Python
- 1.3PB migrated and validated. CALS switched off 2021

User survey review:

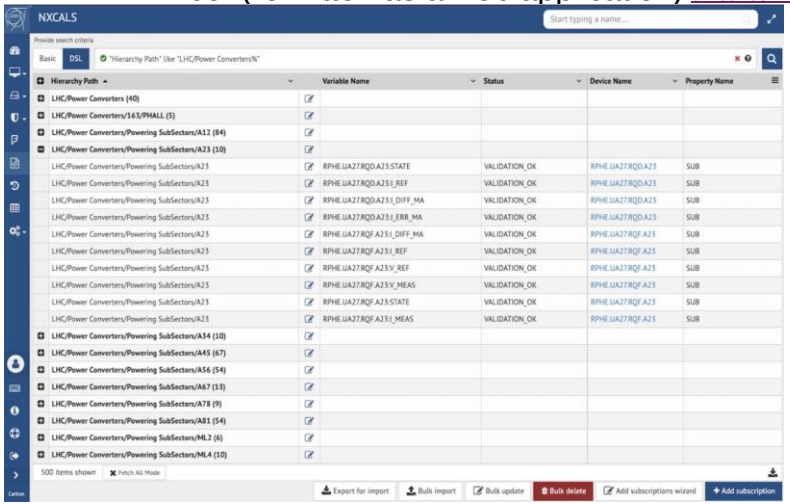
- Extraction performance
  - Significant improvement coming for recent (T-48h) data. Will require NXCALS update in YETS.
- RBAC authentication
- Data reduction (downsampling, on-demand logging)

- (new) Logged LSA settings



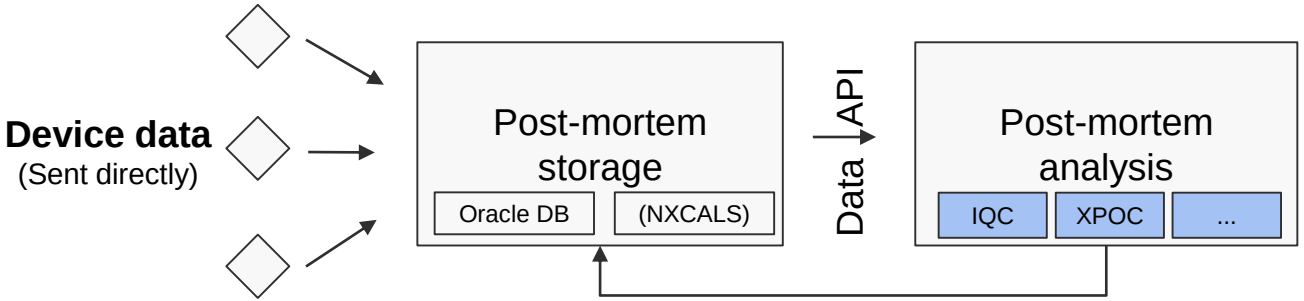
Notebook based analysis using NXCALS on SWAN

Timber (rewritten as a web application) timber.cern.ch



Self-service configuration via ccde.cern.ch

Post-mortem refresh (TE-MPE)



- Dual instances: Core PM & SPSQC
- Improved scalability of data collection
- Replacement of underlying storage
- Introduction of new Data API (REST)
- Backwards compatibility for PM analysis

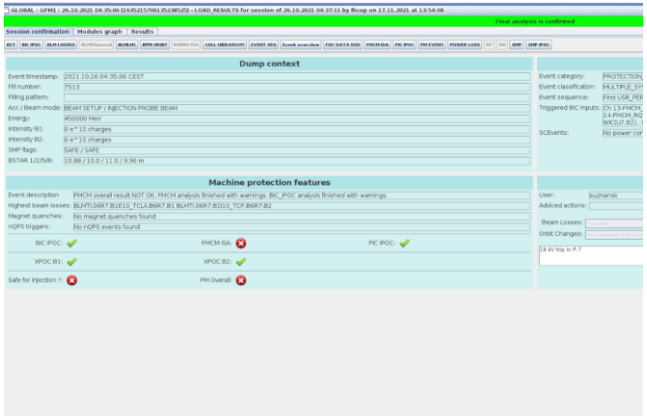
Challenging requirements

- Large volumes of data
- Must be processed quickly and reliably
- Difficult to fully test without beam

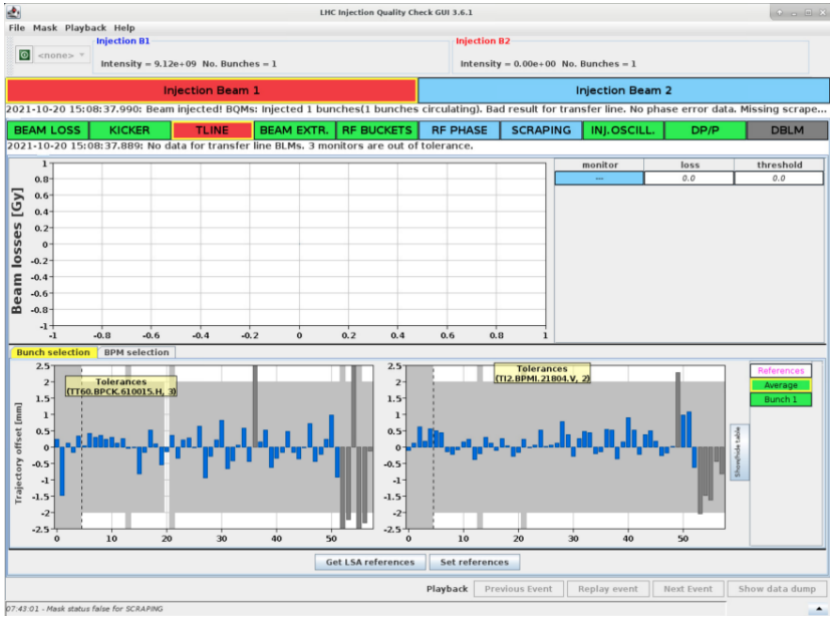
Post-mortem refresh (TE-MPE)

Outcomes of 2021

- Validated during operations, incl. LHC beam test
- Identification and resolution of performance issues
- Unable to dump to both PM storage instances
  (as required for SPSQC & IQC)



PM example from beam test 2021



Injector Quality Check (from Y. Dutheil's talk) with incomplete analysis possibly due to the need for data in both instances (Core PM & SPSQC)

Plans

- Addressing remaining issues identified in beam test
- Long-term storage of PM data on NXCALS
- Full offline analysis functionality (incl. Pre LS2)

Software lifecycle management

External technology evolution is ***accelerating***

More frequent releases, shorter lifetime (Java, Python, Linux OS, ...)

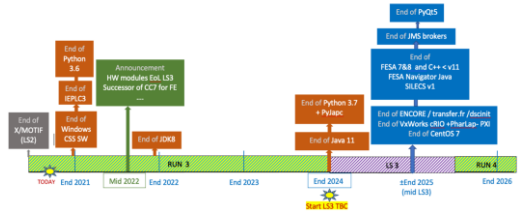
External releases don't necessarily align with the Run schedule/duration

Software end-of-life is an essential part of maintaining a healthy control system

LS3 will be a major milestone for software end-of-life

- Linux CentOS7 officially end of life during Run 3:
  - FECs remain CC7 until end of run
  - Consoles and servers to be upgraded mid-Run
- Java upgrade ~2023

The longer the run, the more external change we will be exposed to.



Changes and EoL being coordinated. Details at https://wikis.cern.ch/display/codp/Controls+Legacy+End+of+Life

Summary

A lot of change during LS2, huge effort and mostly backwards compatible.

Successfully validated by operations in 2021.

Change is essential, but can be disruptive. We try to mitigate impact as much as possible.

Regularly engaging with user communities to:

- Understand and follow-up on user needs
- Raise awareness of changes as early as possible

We’re looking forwards to a successful Run 3, and to continuing our fruitful collaborations throughout the sector!