LHCb Online Monitoring

CHEP 2023, May 8-12

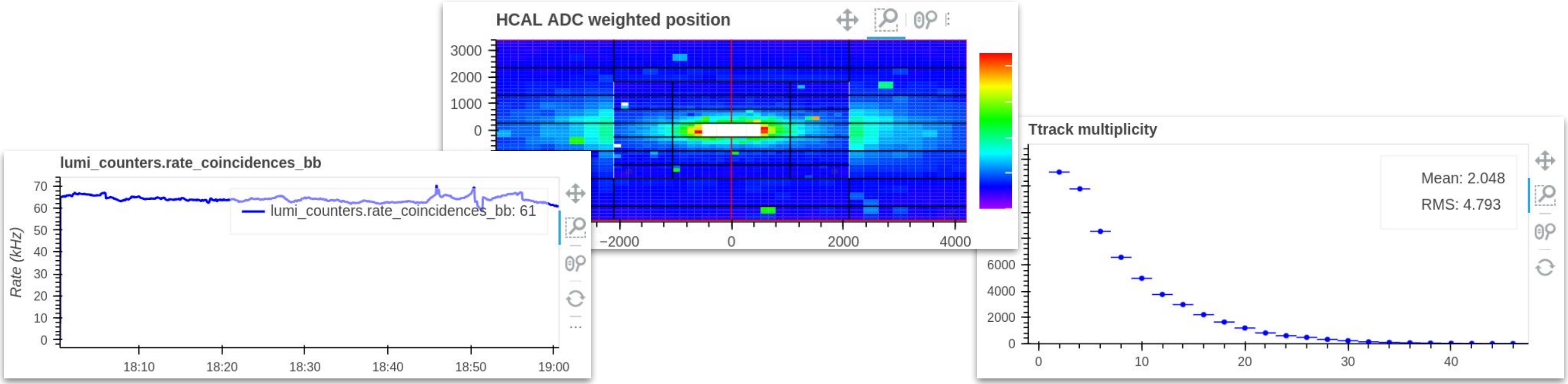
R. Matev (CERN) and P. Robbe (IJCLab)

Online monitoring

- Several reasons why we need monitoring
  - quickly react to problems with the hardware / data taking conditions
  - quality control (e.g. validation of trigger configuration)
  - flag the data quality for physics analysis
  - record conditions (e.g. inputs for simulation)
  - real-time feedback to LHC
  - debug problems
- Main parts
  - data sources
  - moving and storing data
  - analysis and visualisation

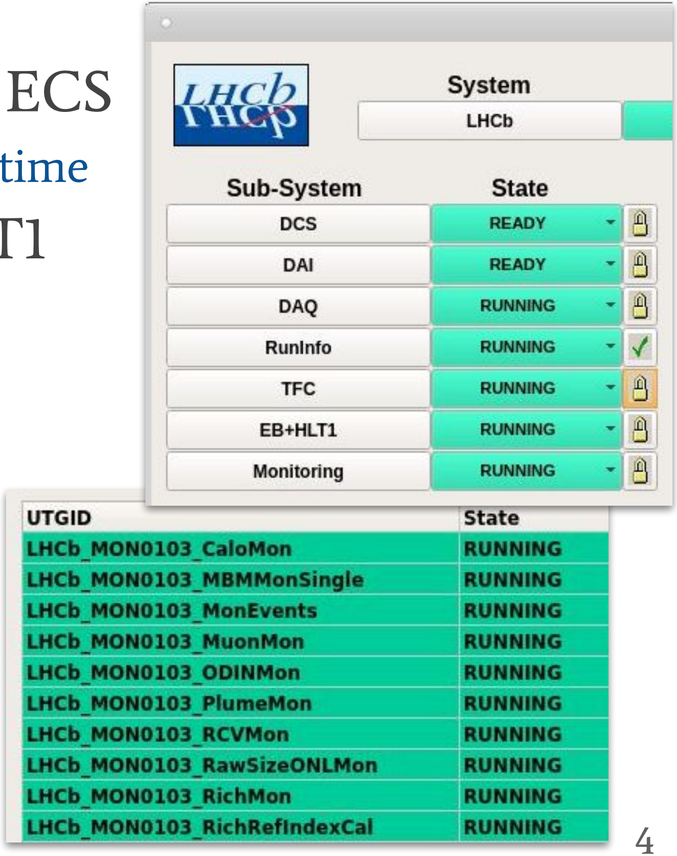
Data sources

- Environmental quantities: produced by electronics, HV, readout, ...
- Detector raw data (aggregated into histograms)
- Higher-level quantities: output of HLT reconstruction/selections
- Automatic analyses (e.g. combined histograms, fits to shapes)



Subsystem monitoring tasks

- Managed by subsystems, controlled by the ECS
  - part of the system that must be running all the time
- 10 kHz of dedicated events selected by HLT1
  - activity-unbiased events,
  - events with some activity,
  - events for time alignment, “NZS” events, etc.
- Output:
  - Δ histograms, saved every 1-10 min to .root files
  - live histograms published via DIM
- Gaudi-based tasks
  - using standard LHCb physics software



Gaudi counters

- Until a few years ago we only had `StatEntity`
  - no need to declare anything, owned by a global registry
    `counter(“# skipped”) += skipped.size()`
  - mixed many purposes: pure counter, binomial counter, statistics
  - **prohibitive cost**: name lookup (synchronized) and a lock for every increment
- New counters (`Gaudi::Accumulators::`)
  - members of the algorithms/tools, no lookup involved
  - minimal counter backends (e.g. 1 uint for a counter, 2 uints for a binomial)
  - thread safety by using atomic increments (⇒ eventual consistency)
  - **fast!**
- New warning/error counters: emit log messages, back by a counter

Gaudi histograms

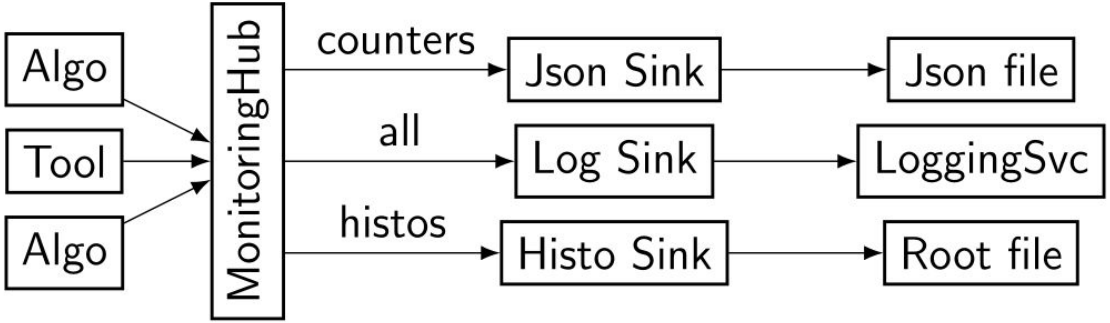
- Up until recently we only had:
  - histograms “owned” centrally by HistogramSvc
  - backed by the non-thread-safe ROOT6 histograms
  - same issues as for the StatEntity counters
- Histograms as (new) counters

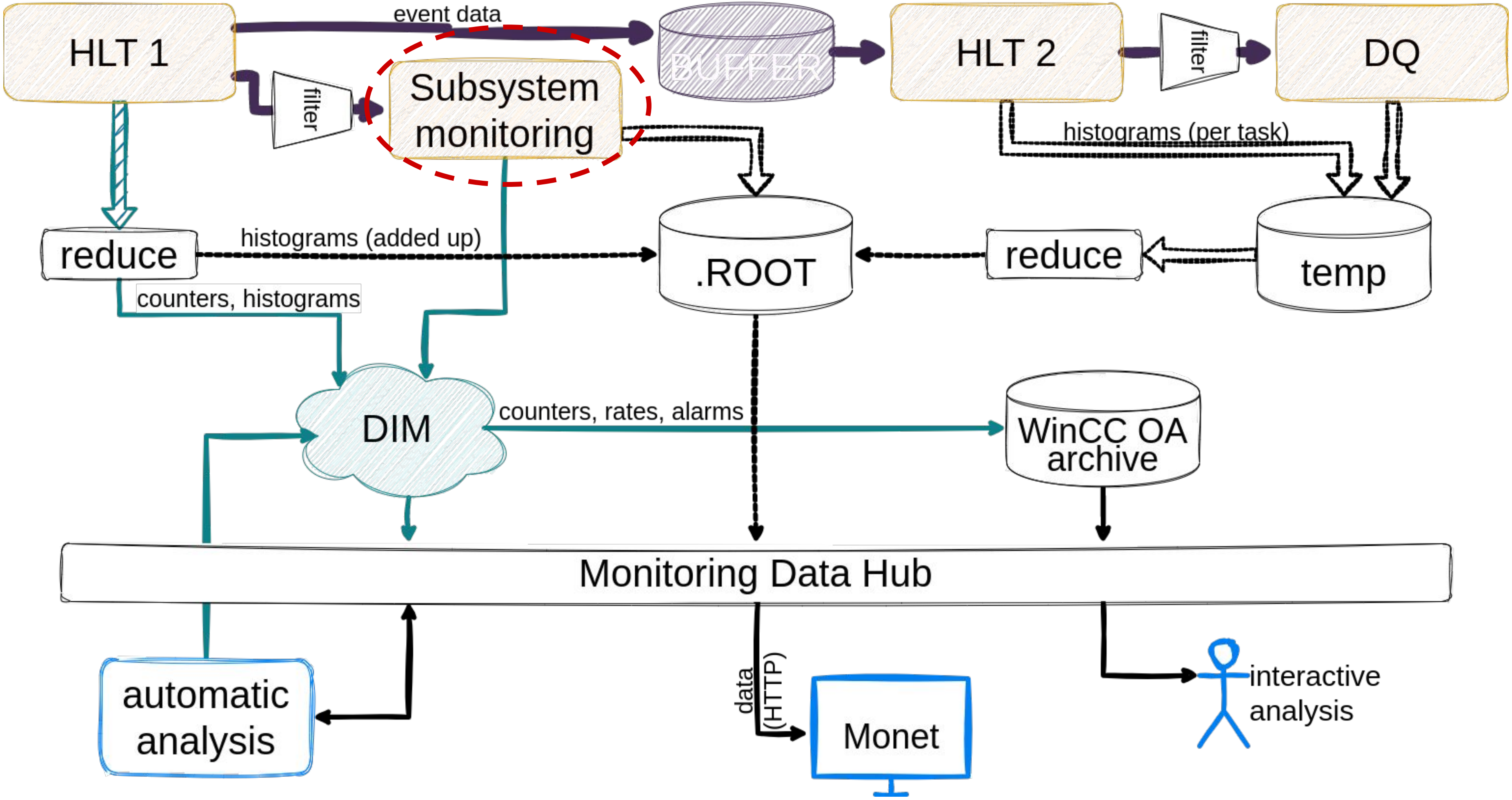
```
WeightedHistogram<1> wh{this, "WH", "Title", {nBins, min, max}};
hw[val] += weight;
```

  - backend is an array of (counter) accumulators
  - support weighted, profiles, N dimensions, custom counter storage type

Generalizing non-event-data output

- Decouple histo/counter producers & publishers
  - allows new types of non-event data and new ways of publishing
  - use a generic schemaless serialised exchange format (JSON)
- Send all data to unique service
  - dispatching all data to all sinks
  - with selection of data by “type” in each Sink
- Each sink can have its own output
  - e.g. structured formats (JSON, XML, ...)
  - or network forwarding for online case




HLT 1
event data
BUFFER
HLT 2
filter
DQ
filter
Subsystem monitoring
histograms (per task)
reduce
histograms (added up)
.ROOT
reduce
temp
counters, histograms
DIM
counters, rates, alarms
WinCC OA archive
Monitoring Data Hub
automatic analysis
data (HTTP)
Monet
interactive analysis

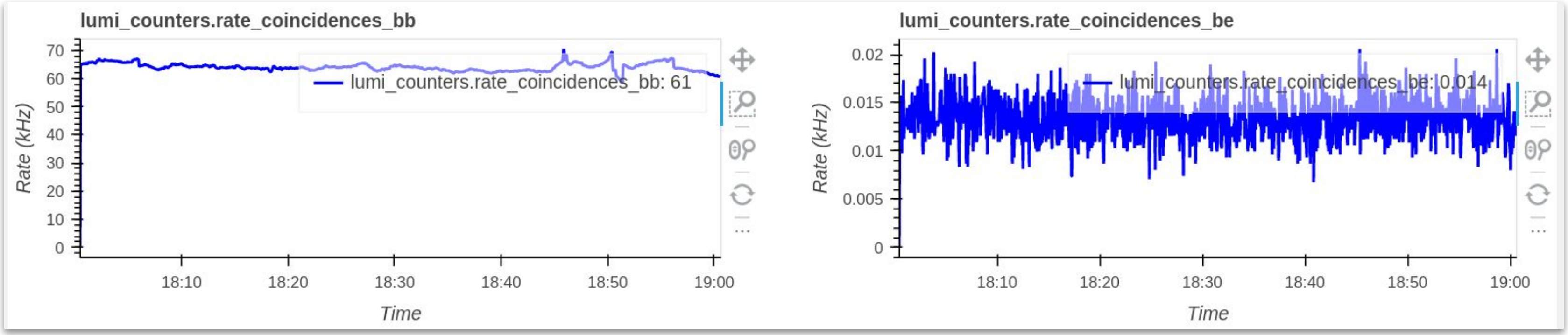
HLT and DQ monitoring

- HLT 1 monitoring
  - HLT 1 is a Gaudi task that wraps the GPU processing
  - only processing step that sees the full 30 MHz collision rate
  - real-time reconstruction/selection histograms
- “RecoMon” (a subsystem monitoring task)
  - real-time histograms for (HLT 2) reconstruction and some high-rate signals
- HLT 2 and DQ monitoring
  - input to the physics data quality flagging
  - low level (detector) and high-level (reconstruction/selection) histograms
  - DQ task runs synchronously, on the output of HLT 2
  - separation is for operational reasons (easier to update DQ than HLT 2)

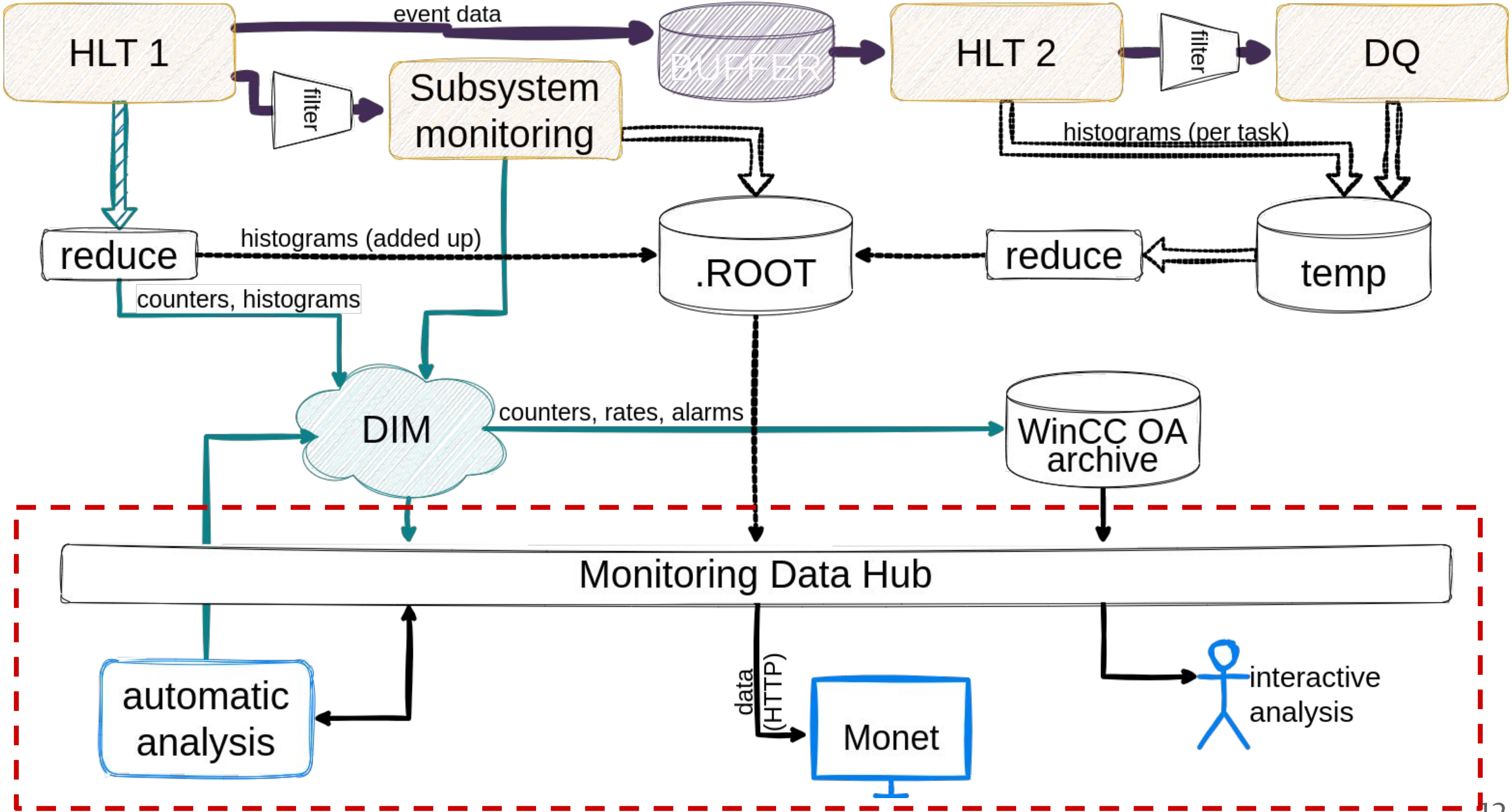
“Reducers”

- HLT 1/2 processing is distributed over 100s/1000s processes
  - data from all processes must be added up
- HLT 1 needs “live” aggregation
  - data is published via DIM in regular intervals
  - data is consumed by a dedicated process and added up
  - aggregated data is republished via DIM and saved to ROOT files
- HLT 2 and DQ monitoring output can have some latency
  - data is saved to files (usually) once per run
  - data is added up asynchronously and saved to ROOT files

WinCC OA archive DB



- WinCC OA (RDB) archiving is the standard data store in the ECS
  - some important quantities must be monitored by the shift crew
- Web API exists to query archived data points
- Reuse infrastructure for time-series data (counters, rates) from tasks
  - counters published via DIM, which is well integrated in WinCC
  - archive storage+query latency is low enough



HLT 1
event data
BUFFER
HLT 2
filter
DQ
filter
Subsystem monitoring
histograms (per task)
reduce
histograms (added up)
.ROOT
reduce
temp
counters, histograms
DIM
counters, rates, alarms
WinCC OA archive
Monitoring Data Hub
automatic analysis
data (HTTP)
Monet
interactive analysis

Analysis and visualisation services

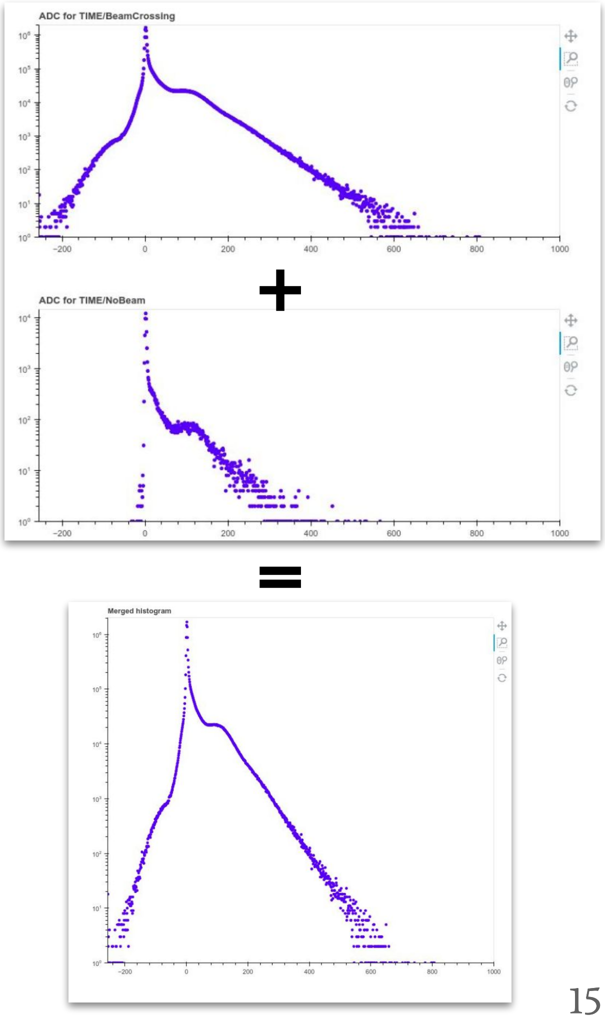
- Lightweight web services using the Flask Python framework
    - Monitoring Data Hub, Automatic Analysis, Monet (visualisation)
- Run in containers, started and managed by the Online K8s cluster
- Automated deployment
    - When a new version is tagged, e.g. in the Monet GitLab repository, a new Docker image is automatically created and can be deployed
    - All past versions are kept and available
    - Deploying a new version (bug fix for ex.) takes 5 minutes
    - Reverting to an old version or restarting the server takes about 2 minutes

Monitoring data hub

- Common interface for all monitoring sources to Monet
  - DIM
  - files (on the filesystem, via xrootd)
  - WinCC archive DB (slow control data, counters, alarms)
  - “automatic analysis”, i.e. monitoring data transformations
  - new sources could be added when needed
- Uniform way to request and obtain data
- Output are JSON objects that Monet can interpret and display
- Exposed API for easy interactive analysis

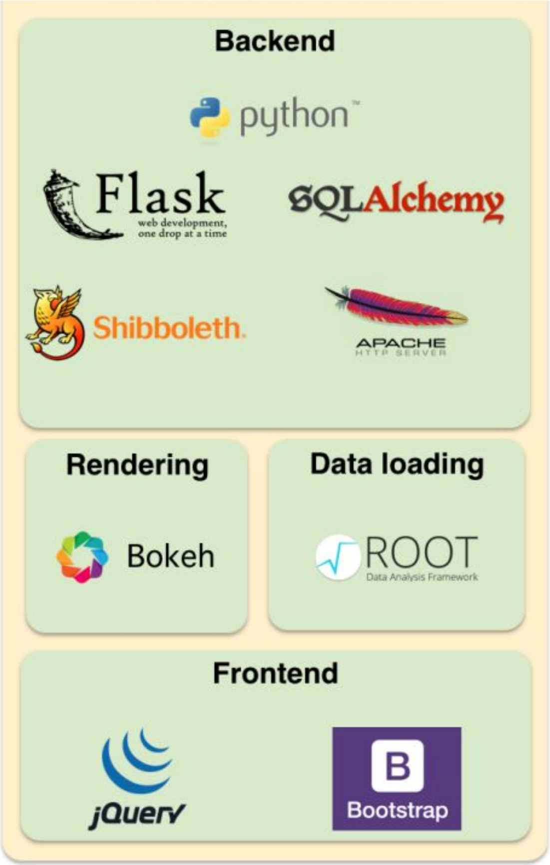
Automatic analyses

- Web service to transform monitoring data
  - receives the data to operate on; returns new data
  - triggered regularly to check for alarm conditions
  - used for custom visualisations
- Preset transformations and checks
  - merge, divide, project, detect holes, spikes, ...
- User-defined analysis

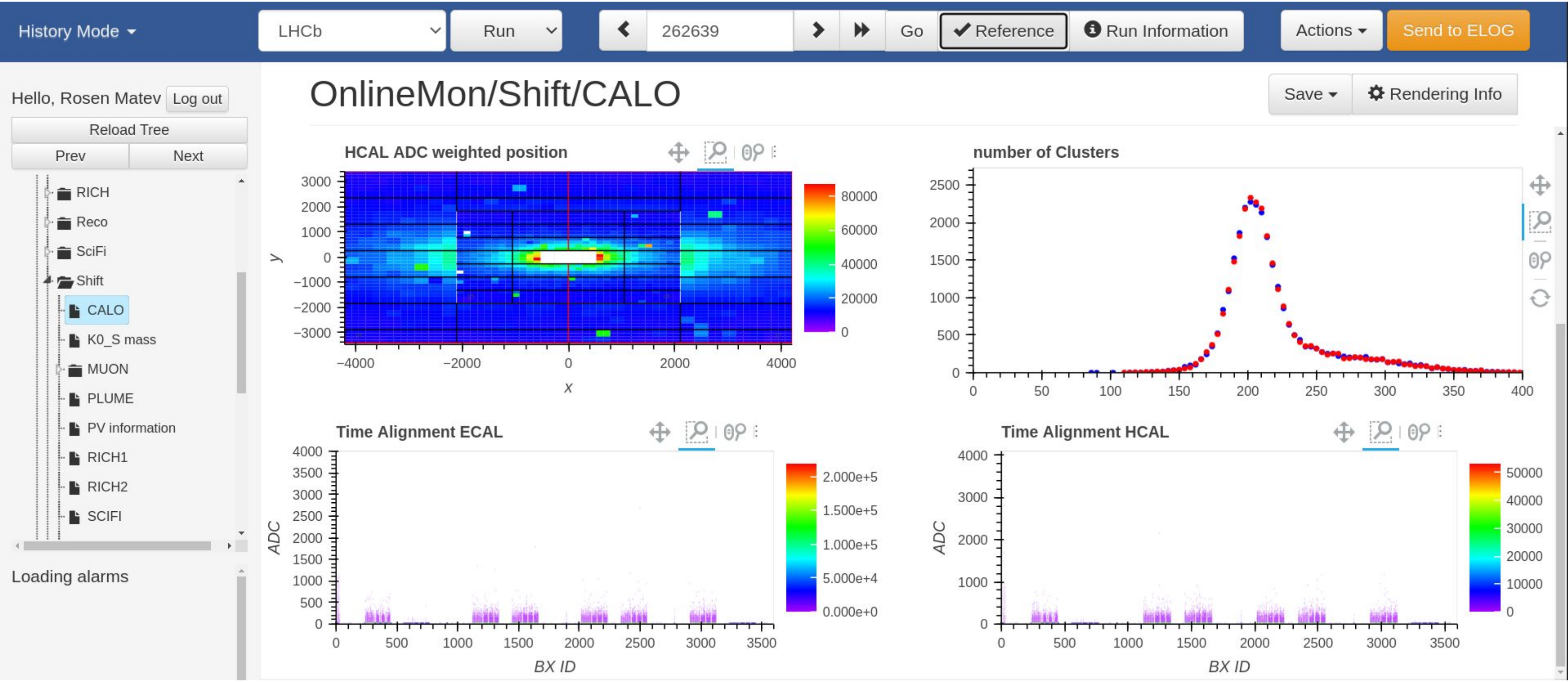


Monet: visualisation tool

- Based on standard web technologies
- Display monitoring algorithm output and time series
- Display organised in pages
  - layout of display configurable by subsystem experts
  - configuration in a GitLab repo of yaml files
- Some features
  - overlay several histograms in one plot
  - customise drawing attributes
  - overlay references
  - annotate plots
  - draw profile histograms
  - send plots to ELOG
  - display alarms



J. Phys.: Conf. Ser. 898 092027

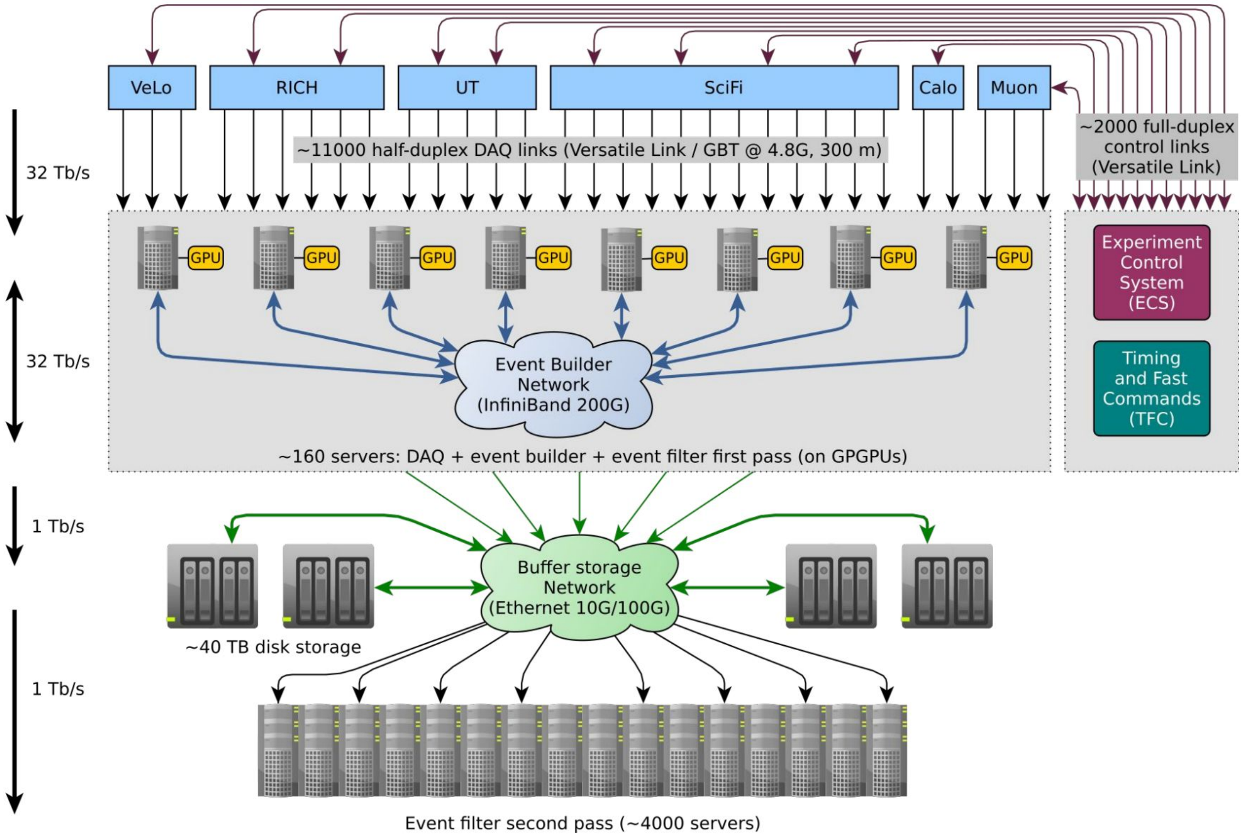

History Mode
LHCb
Run
262639
Go
Reference
Run Information
Actions
Send to ELOG
Hello, Rosen Matev
Log out
Reload Tree
Prev
Next
RICH
Reco
SciFi
Shift
CALO
K0_S mass
MUON
PLUME
PV information
RICH1
RICH2
SCIFI
Loading alarms
OnlineMon/Shift/CALO
Save
Rendering Info
HCAL ADC weighted position
number of Clusters
Time Alignment ECAL
Time Alignment HCAL
ADC
BX ID

Conclusion

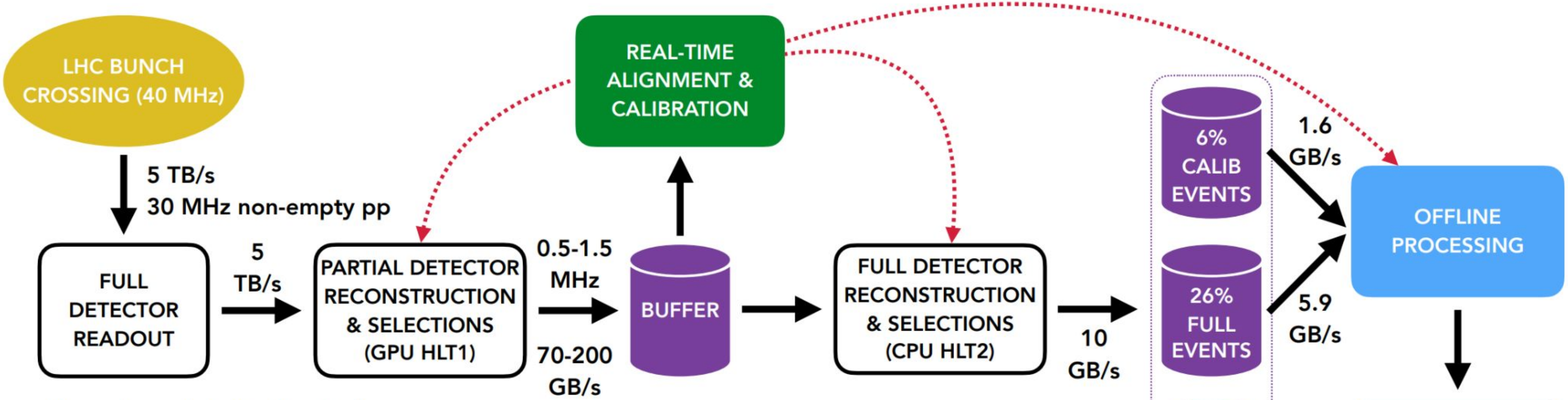
- LHCb is taking data with the upgraded detector in the Run 3
- Live data monitoring is crucial
  - data with malfunctioning detector will be discarded for offline analysis
- Combine data from the control system and processed event data
- Important plots are delivered to the shift crew and experts
  - cover all subsystems (subdetectors, trigger, alignment)
  - quickly diagnose problems
  - qualify the data for physics analysis

Thank you

DAQ architecture



LHCb upgrade dataflow



All numbers related to the dataflow are taken from the LHCb

Upgrade Trigger and Online TDR

Upgrade Computing Model TDR

Trigger output

- Dedicated raw data format in the online system
  - trivial and concatenable (after HLT1)
- “Routing bits” are set by HLT1 / HLT2 per event
  - used to decide which event goes where (e.g. input of monitoring tasks or data for tracker alignment, etc.)
- HLT 2 needs to stream internally (different content per stream)