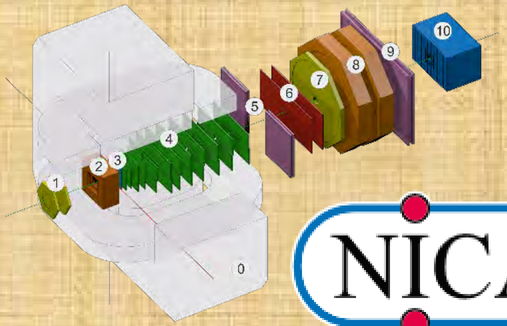
Design of the Data Quality Monitoring system for the BM@N experiment

Igor Alexandrov, Evgeny Alexandrov,Alexander Chebotov, Konstantin Gertsenberger

JINR, MLIT, LHEP



NICA

BM@N

13-15 May 2025
14th Collaboration Meeting of the
BM@N Experiment at NICA

Joint Institute for Nuclear Research

Data Quality Monitoring (DQM) and Quality Assurance (QA) : Goals of the systems

- **histogram creation tool**
  - **interface for histograms creation**
  - **library with simple implementations of this interface**
- **create flexible tool for online creation, filling, transport and archival of histogram and other monitor elements**
- **create flexible online tool to perform algorithms for automated quality and validity tests**
- **keep the results of the DQM process**
- **create GUI for DQM user**
  - **visualization of the histograms and quality test results**
  - **alarms in case of bad quality data**

Main goals

❑ **online and offline data quality check and results visualization**

❑ **fast automation reaction on data quality problems**

DQM & QA systems in LHC experiments: ATLAS

- **ATLAS**: light, flexible (input-output-configuration interfaces, algorithms as plugins)
  - DQM Core: executes DQ Algorithms (any common operation like histogram comparison, histogram fitting, thresholds application, etc.); has three abstract interfaces for the communication with the external systems
    - DQM Input (receives histograms, messages, counters)
    - DQM Output (way of publishing DQ Results produced by the DQ algorithms)
    - DQM Configuration interface (way of reading configuration info which defines behavior of the DQM Core in a specific environment)
  - DQ Configuration is described as a hierarchical tree of objects of two different types: DQ Regions and DQ Parameters.
  - DQ Region
    - Children (DQ Region or DQ Parameter)
    - DQ Summary Maker
  - DQ Parameter
    - location of the monitoring information (represents the state of a particular detector element)
    - weight
    - DQ Algorithm that has to be used
    - specific parameters and thresholds
    - reference values or histograms
    - the actions which have to be taken depending on the results

DQM & QA systems in LHC experiments : ATLAS (2)

DQ Algorithm

- DQM Framework (DQMF) provides a number of predefined DQ Algorithms
- DQ Algorithms are integrated into the DQMF in a dynamic plug-in manner
    - allows adding new algorithms on the fly without modifying the core software
- each DQ Parameter has at least one DQ Algorithm associated with it
- executed whenever a piece of info which is associated with that DQ Parameter becomes available

DQ Summary Maker

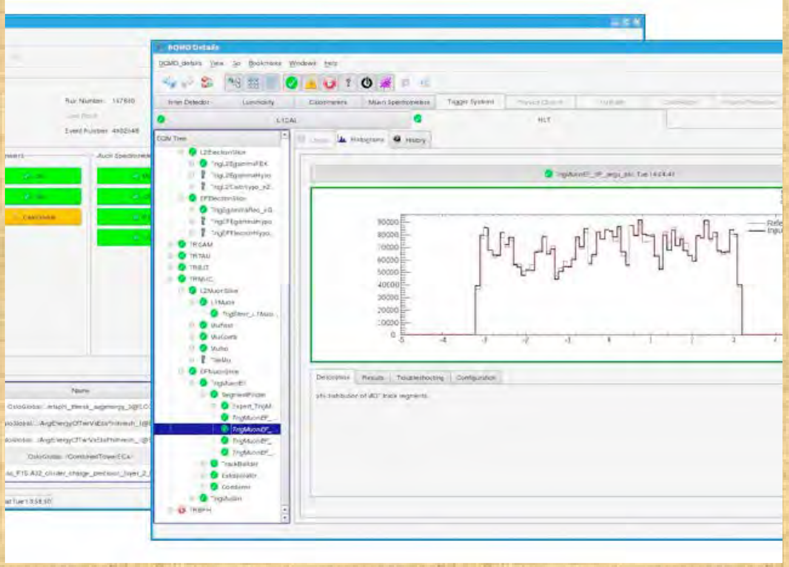
- special implementation of the DQ Algorithm interface that evaluates the DQ Result for a given DQ Region

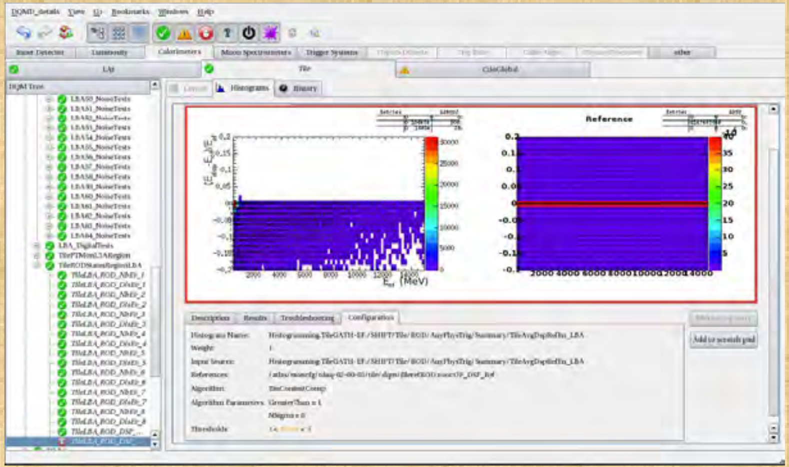
DQMF Agent

- instantiates appropriate implementations of the DQMF generic interfaces (i.e. DQM Input, DQM Output and DQM Configuration)
- takes care of starting and stopping the DQM Core engine in appropriate moments
    - In the online environment the DQ assessment has to be started at start of run event and stopped when the run is finished
- DQMF may contain one or more DQMF Agents with each of them responsible for a well defined subset of the whole ATLAS system.

DQ Result

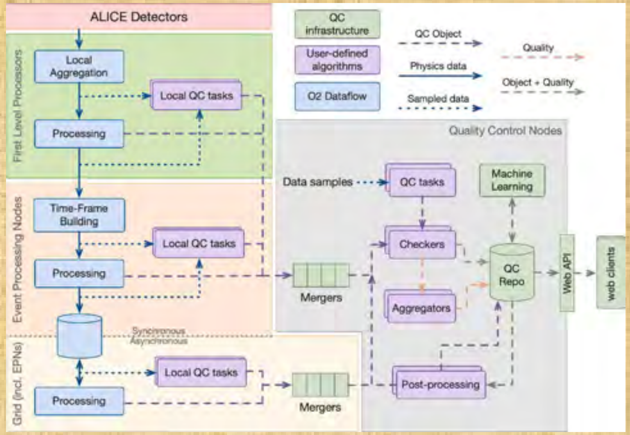
- consist of a colored tag and any output that the algorithms might want to attach
    - If some areas of the detector are disabled, then the corresponding dq results will be **black**
    - otherwise, results might be
        - green (good), yellow (warning), red (bad) or gray (undefined)





Histogram Name: Histogramming.TileGATH-EF./SHIFT/Tile/ROD/AnyPhysTrig/Summary/TileAvgDspRefEn_LBA
Weight: 1
Input Source: Histogramming.TileGATH-EF./SHIFT/Tile/ROD/AnyPhysTrig/Summary/TileAvgDspRefEn_LBA
References: [illegible]
Algorithm: BinContentComp
Algorithm Parameters: GreaterThan = 1
NSigma = 0
Thresholds: [illegible]

DQM & QA systems in LHC experiments : ALICE



- **ALICE**: DQM + QA = Data Quality Control (DQC)
  - largest DQC systems worldwide
  - first in the high energy physics community to leverage the **message passing technique** and the actor model to such an extent
  - high-level quality assessment of the 3.5 TB/s data produced by the detector
- QC system
  - multi-step process
    - sampling the data, usually at a rate of 1%
    - QC Tasks will then execute user-defined algorithms to process it and generate a QC Object, often a histogram
    - Given the parallel nature of this processing, with a copy of the task running on each of the hundreds of nodes, these **histograms** are then **merged**
    - merged results are evaluated by a series of Checks to determine one or several Qualities, which can themselves be aggregated to give a general assessment of the health of the data
  - based on a **message passing paradigm** where data flows asynchronously through a set of devices connected via buffered channels
  - channels use ZeroMQ by passing either the whole message payloads or just pointers to the shared memory region
  - QC Objects and Qualities are stored in Conditions Database

DQM & QA systems in LHC experiments : LHCb

- **LHCb**: collected data are grouped together in runs
  - LHCb DQ workflow
    - small subset of the data selected by the trigger is fully-reconstructed on the LHCb Online computing farm
    - reconstruction produces sets of histograms which allow the (sub-)detector performance to be assessed
    - these histograms are presented by the Data Quality Monitoring(DQM) software to the DQ shifter
    - shifter compares whether the run is suitable for physics analysis or not, by comparing it to a reference run previously set by experts.
    - software package previously used in DQM shifts, was based on dedicated custom C++ code and X Window System. Now implemented "Monet" which is a python based web application that supersedes the Presenter.
  - Update
    - using Python as the primary language allows the usage of rich set of libraries provided in the large ecosystem of third-party Python packages. This simplifies both development, as common functionality has already been implemented elsewhere, and maintainability, as the size of the required LHCb specific code is reduced.
    - for plotting Bokeh libraries provide interactive plots in web browsers, has pythonic interface
  - RoboShifter: automatic problem detection
    - predicts probability of given run being bad
    - decisions made by each tree are summed with weights, representing the importance of each tree
    - each tree corresponds to a single histogram
    - possible to compute, for each histogram, its contribution to the probability of the run being bad
    - histograms with the highest contributions can be presented to DQ shifter as potentially problematic ones.
  - Machine learning at LHCb: vector Kolmogorov-Smirnov distances btw histograms and their references; AdaBoost algorithm,
    - track pattern recognition
    - long track reconstruction
    - downstream Track Reconstruction (reconstruction of the daughters of long-lived particles)
    - fake track rejection
    - topological trigger (HLT2)
    - jet tagging
    - charged particle identification

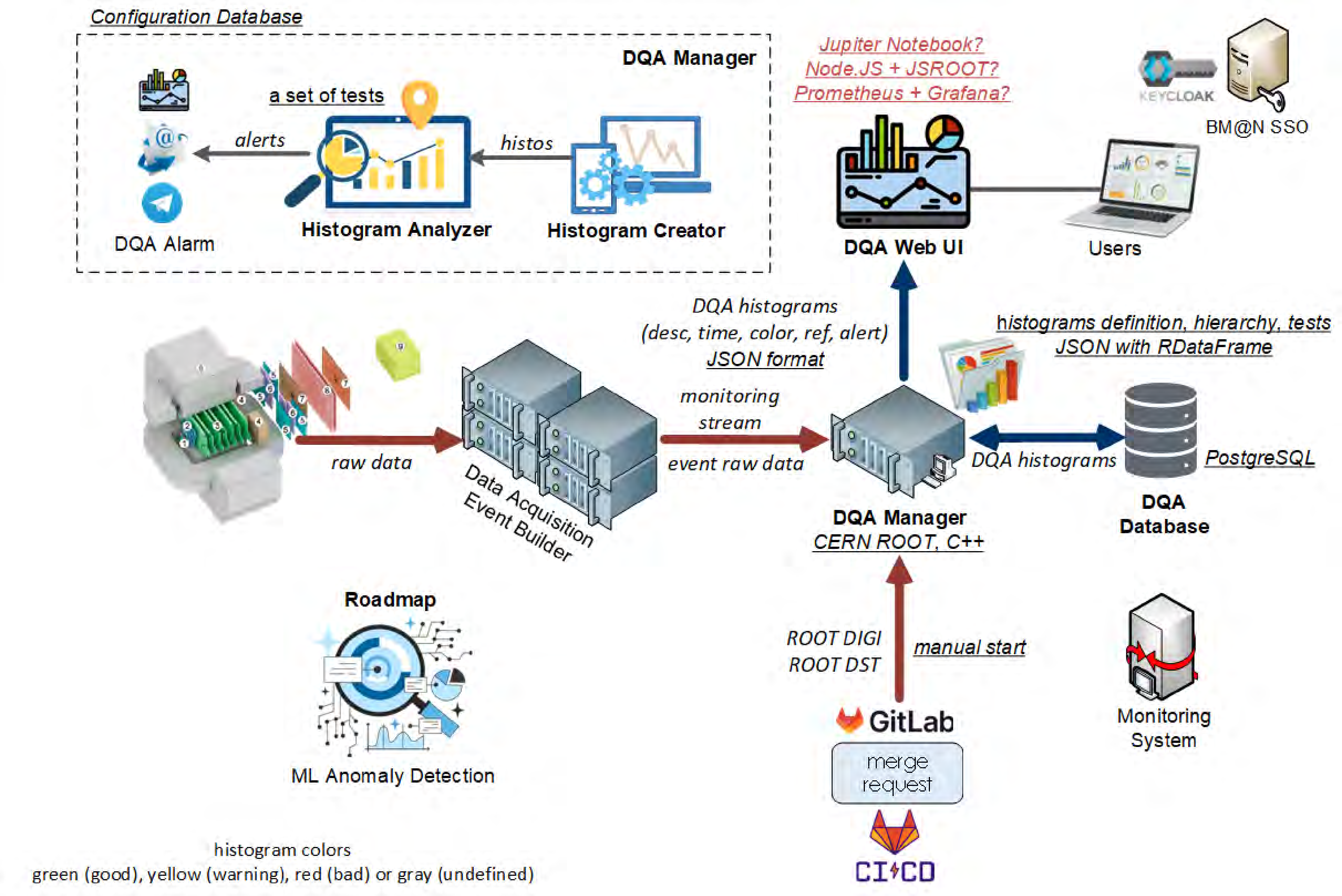
DQM & QA systems in LHC experiments : CMS

- **CMS: The DQM software is a central tool in the CMS experiment. High-level goal of the system is to discover and pin-point errors - problems occurring in detector hardware or reconstruction software**
  - tools for
    - creation, filling, transport and archival of histogram and scalar monitor elements
    - standardized algorithms for performing automated quality and validity tests on value distributions
  - monitoring systems live online for
    - the detector, the trigger, and the DAQ hardware status and data throughput,
    - the online reconstruction
    - validating calibration results, software releases and simulated data
  - visualization of the monitoring results
  - certification of datasets and subsets thereof for physics analyses
  - retrieval of DQM quantities from the conditions database
  - standardization and integration of DQM components in CMS software releases
  - organization and operation of the activities, including shifts and tutorials

DQM & QA systems in LHC experiments : what interesting, some conclusions, considirations on using expirience

- All do it or interesting in LHC experiments:
  - produce some sampling (events go to DQM depending on their frequency after trigger system)
    - message passing technique (ALICE)
    - set the rate for DQM input
    - produce histograms as main input for DQM
      - but not only histograms can be used for quality check
  - moving as much as possible to automation of quality assurance
  - flexibility in check algorithms use (keep in database source of algorithms, their names and parameters, what to be on output, destination etc)
    - load from library the check algorithm as implementation of interface (ATLAS)
    - some interfaces have thresholds as parameters in order to create result: bad or good event
  - some of check algorithms used
    - empty histogram (threshold what does it mean empty)
    - counters of subdetectors responses by sectors
    - Kolmogorov-Smirnov test
  - moving to rich shifter GUI, mostly web GUI
    - different base tools/languages for implementation (python as example in ALICE, C++ in ATLAS)
    - alarms in case of big probability of events bad quality
  - ALL these experiments suppose to use in RUN4 ML-based quality assessment

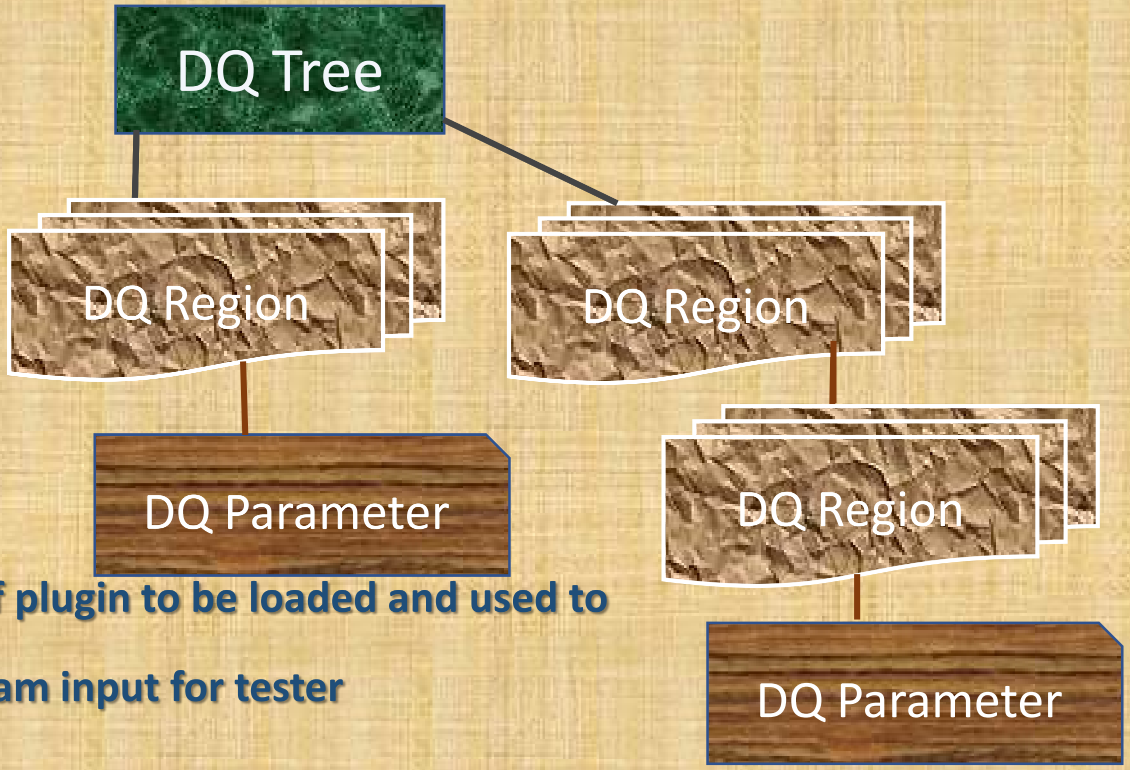
General architecture of the system



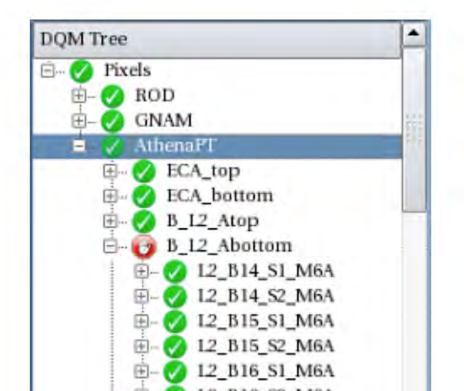
histogram colors
green (good), yellow (warning), red (bad) or gray (undefined)

The DQM tree

DQM Configuration keeps set of DQM Tree in JSON format.



- **DQ Tree**
  - **Histogram producer library (plugins)**
  - **Analizer library (plugins)**
  - **DQ Region … DQ Region**
    - **DQ summary maker**
    - **DQ Parameter**
      - **Input from sampler**
      - **Histogram producer**
        - **Algorithm for histogram creation (name of plugin to be loaded and used to produce histograms)**
        - **Output with histogram produced - histogram input for tester**
      - **Analyzer (quality tester)**
        - **histogram input (from previous output)**
        - **Algorithm (quality checker algorithm)**
        - **Threshold (numbers to understand if histogram good or bad)**
        - **Output**
          - **result (good, bad, undefined, disable)**
          - **DQ Tree path**
          - **histogram**

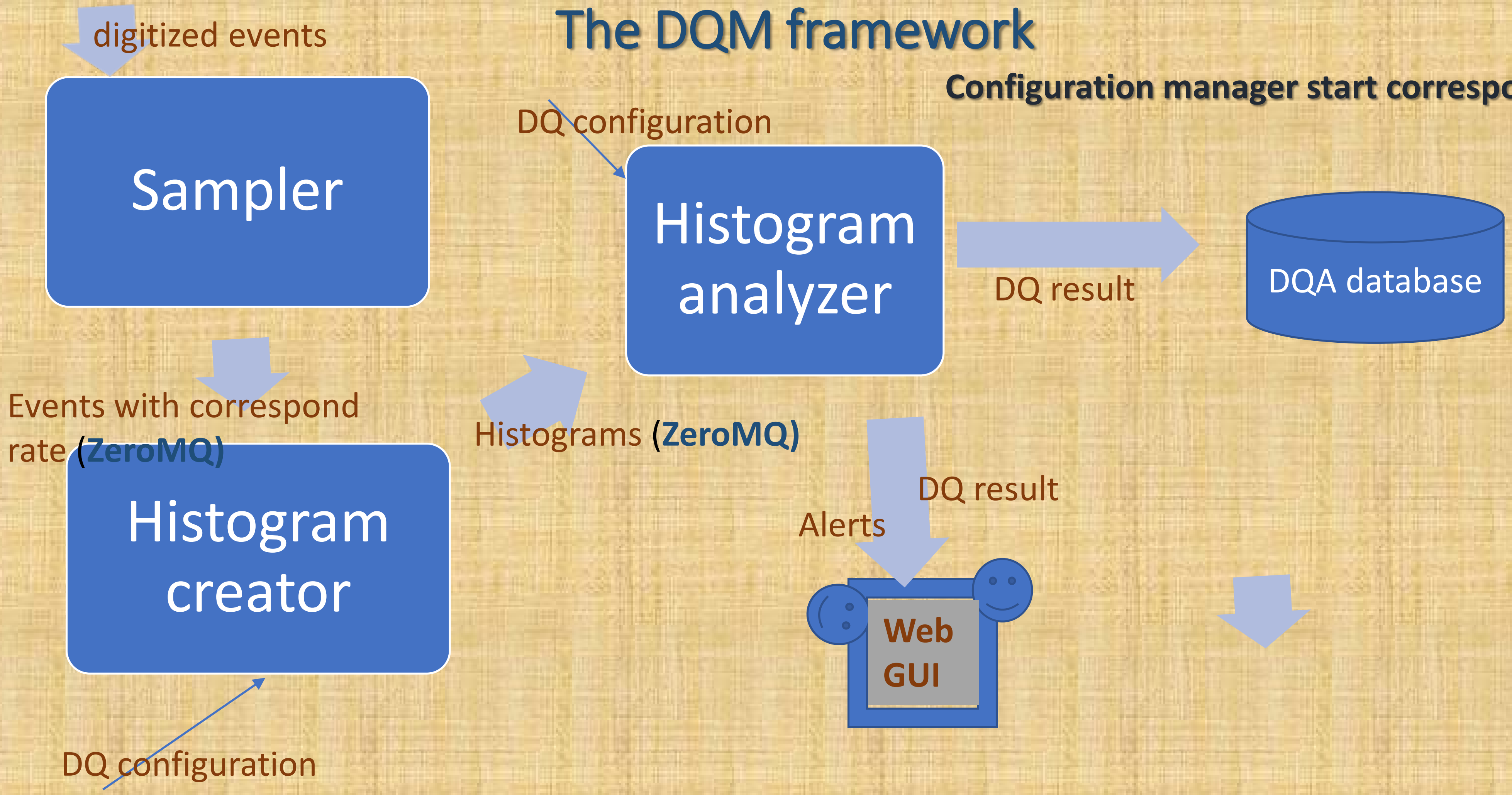


DQM Tree

- ✓ Pixels
  - ✓ ROD
  - ✓ GNAM
  - ✓ AthenaPT
    - ✓ ECA_top
    - ✓ ECA_bottom
    - ✓ B_L2_Atop
    - B_L2_Abottom
      - ✓ L2_B14_S1_M6A
      - ✓ L2_B14_S2_M6A
      - ✓ L2_B15_S1_M6A
      - ✓ L2_B15_S2_M6A
      - ✓ L2_B16_S1_M6A

The DQM framework

Configuration manager start correspond DQ setup



digitized events

Sampler

Events with correspond rate (ZeroMQ)

Histogram creator

DQ configuration

DQ configuration

Histogram analyzer

Histograms (ZeroMQ)

DQ result

DQA database

DQ result

Alerts

Web GUI

Next steps and conclusion

- To be finalized
  - detailed DQ configuration structure
  - database scheme
- Start the work on
  - DQM web GUI design
  - DQM system implementation

Conclusion

- **Review of LHC experiments DQM and DQA systems is presented**
- **DQM system general architecture and system framework is shown**
- **The structures of the DQA configuration and the result structure are discussed.**

BACKUP