DQM4HEP : a generic data quality monitoring framework for HEP

Ete, R. (DESY) *et al*

18 January 2018

The AIDA-2020 Advanced European Infrastructures for Detectors at Accelerators project has received funding from the European Union's Horizon 2020 Research and Innovation programme under Grant Agreement no. 654168.

This work is part of AIDA-2020 Work Package **5: Data acquisition system for beam tests**.

DQM4HEP
A data quality monitoring framework.

BTTB6 2018 - Zurich

R. Ete, A. Pingault, T. Coates

DESY

January 17, 2018

Summary

- Introduction
- Framework presentation
- Experiments running with DQM4HEP
- Current status
- Ongoing and future work

DQM systems in a nutshell

DQM systems in HEP domain:

- Automated data quality assessement
- Alert users when anomalies are observed
- Provide for online/offline analysis
  - Automatic data quality tests, possibly with reference histograms
  - Distributed system for online analysis (data collectors)
  - Dedicated visualization interfaces for shifters
- Must be scalable: from prototypes to collider-like detectors

General goal of using a DQM framework in testbeams:

- Having a better understanding of your DUT
- Understand your setup and run settings
- Avoid starting bad runs
- Discard bad/unexpected data

DQM systems for testbeams

Typical use cases:

DQM systems for testbeams

Typical use cases:

- Environmental/slow control monitoring
  - Gas flow ? Current/HV ? Temperature ? Pressure ? B field ?
    $\rightarrow$ Avoid to start bad runs, discard bad runs

DQM systems for testbeams

Typical use cases:

- Environmental/slow control monitoring
  - Gas flow ? Current/HV ? Temperature ? Pressure ? B field ?
    $\rightarrow$ Avoid to start bad runs, discard bad runs
- Hit maps (e.g calorimeters or trackers)
  - Detect inefficient areas
    $\rightarrow$ Discard bad data, understand your DUT

DQM systems for testbeams

Typical use cases:

- Environmental/slow control monitoring
  - Gas flow ? Current/HV ? Temperature ? Pressure ? B field ?
    $\rightarrow$ Avoid to start bad runs, discard bad runs
- Hit maps (e.g calorimeters or trackers)
  - Detect inefficient areas
    $\rightarrow$ Discard bad data, understand your DUT
- Beam structure analysis
  - Check particle properties: type, momentum/energy ...
    $\rightarrow$ Avoid starting bad runs

DQM systems for testbeams

Typical use cases:

- Environmental/slow control monitoring
  - Gas flow ? Current/HV ? Temperature ? Pressure ? B field ?
    $\rightarrow$ Avoid to start bad runs, discard bad runs
- Hit maps (e.g calorimeters or trackers)
  - Detect inefficient areas
    $\rightarrow$ Discard bad data, understand your DUT
- Beam structure analysis
  - Check particle properties: type, momentum/energy ...
    $\rightarrow$ Avoid starting bad runs
- Combine telescope + DUT
  - Run tracking algorithm, quickly detect mis-alignment
    $\rightarrow$ Understand your setup, discard unexpected data

DQM systems for testbeams

Typical use cases:

- Environmental/slow control monitoring
  - Gas flow ? Current/HV ? Temperature ? Pressure ? B field ?
    → Avoid to start bad runs, discard bad runs
- Hit maps (e.g calorimeters or trackers)
  - Detect inefficient areas
    → Discard bad data, understand your DUT
- Beam structure analysis
  - Check particle properties: type, momentum/energy ...
    → Avoid starting bad runs
- Combine telescope + DUT
  - Run tracking algorithm, quickly detect mis-alignment
    → Understand your setup, discard unexpected data

Problem: One experiment = one EDM = one framework !

- Detector algorithm (DA) not re-usable by other experiments
- Leads to duplicated software and efforts
- EDM dependency: custom prototype EDM make use of these framework complicated → Each new prototype comes with its ad-hoc solution

DQM systems for testbeams

Typical use cases:

- Environmental/slow control monitoring
  - Gas flow ? Current/HV ? Temperature ? Pressure ? B field ?
    $\rightarrow$ Avoid to start bad runs, discard bad runs
- Hit maps (e.g calorimeters or trackers)
  - Detect inefficient areas
    $\rightarrow$ Discard bad data, understand your DUT
- Beam structure analysis
  - Check particle properties: type, momentum/energy ...
    $\rightarrow$ Avoid starting bad runs
- Combine telescope + DUT
  - Run tracking algorithm, quickly detect mis-alignment
    $\rightarrow$ Understand your setup, discard unexpected data

Problem: One experiment = one EDM = one framework !

- Detector algorithm (DA) not re-usable by other experiments
- Leads to duplicated software and efforts
- EDM dependency: custom prototype EDM make use of these framework complicated $\rightarrow$ Each new prototype comes with its ad-hoc solution

Need for a more generic framework

DQM4HEP

Data Quality Monitoring for High Energy Physics

Philosophy:

- Encapsulate changes in (abstract) interfaces
  - No EDM, just a handler for your data
  - Data streaming: how should we read/write your data
- Make user code *plugable*
  - Plugins in shared library: plug and play
  - Make the framework easily extensible

Features:

- Core:
  - Streaming tools for reading/writing event
  - Quality test tools : interface + many templates
- Online:
  - Online analysis plugin (API)
  - Distributed system (TCP/IP)
  - Data collectors : event and histogram collector servers
  - Remote process management

DQM4HEP

Quality test API

Monitor element

- Wrap a ROOT TObject
- Optionally hold a ROOT TObject as reference

Quality test

- Implement the logic for monitor element testing
- Output a quality report (quality flag, success, etc)

DQM4HEP

Quality test API

Monitor element

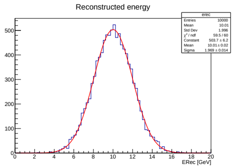
- Wrap a ROOT TObject
- Optionally hold a ROOT TObject as reference

Quality test

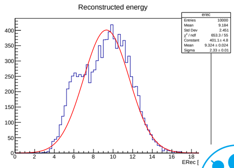
- Implement the logic for monitor element testing
- Output a quality report (quality flag, success, etc)

Concrete example:

- π^+ beam in a calorimeter
- Plot the total energy distribution.
- Assess quality :
  - Fit distribution with gaussian function
  - Extract χ^2 and mean value
  - Check for any deviation



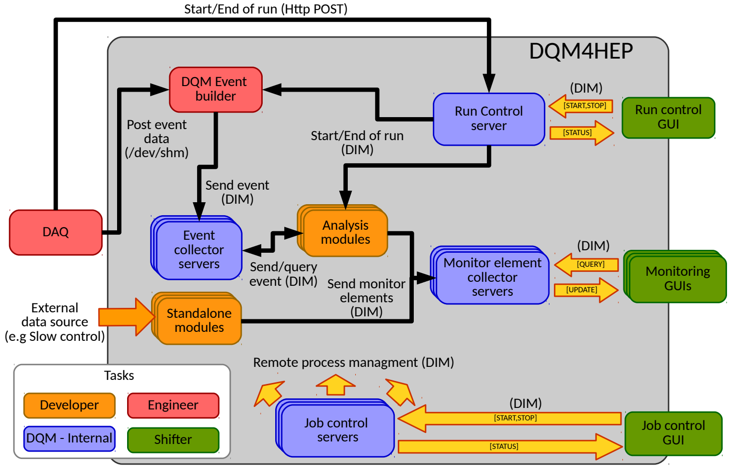
Normal run



Beam conversion ?

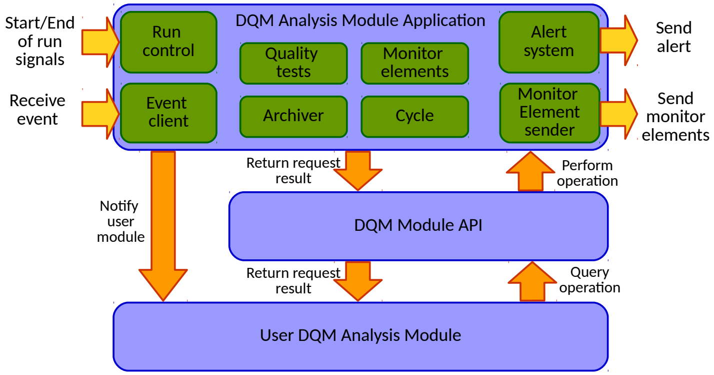
DQM4HEP

Online architecture



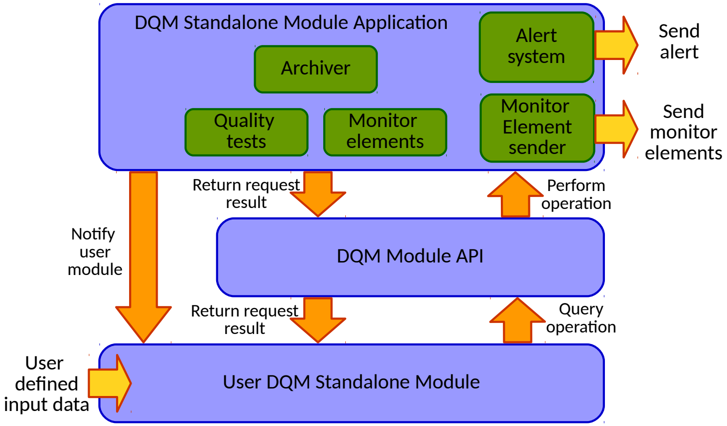
DQM4HEP

Data analysis module



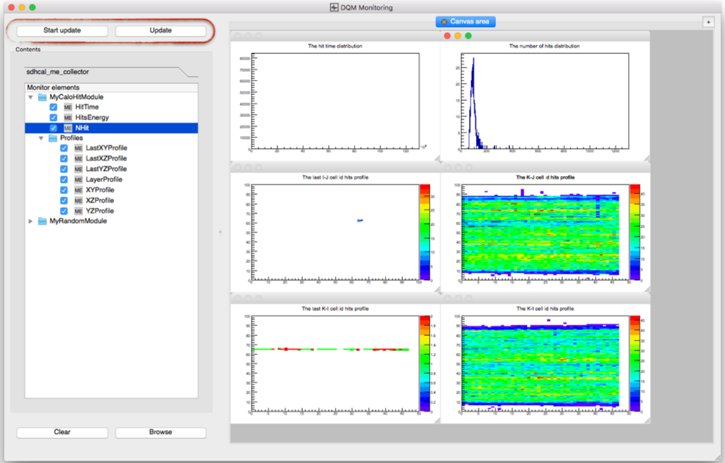
DQM4HEP

Slow control module



DQM4HEP

Online monitoring interface (Qt Gui)



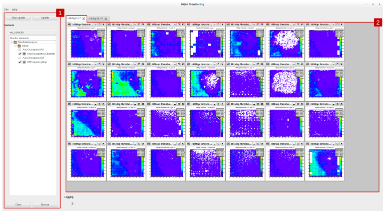
DQM4HEP

Detectors using DQM4HEP

DQM4HEP used by different detectors in the CALICE collaboration.

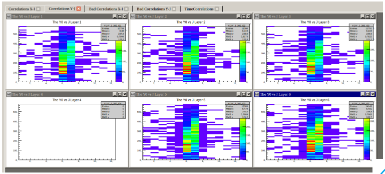
SDHCal online monitoring

- Hit map
- Electronics rate
- Slow control : I, HV, LW, T, P
- GRPC efficiency, multiplicity



AHCal online monitoring

- Hit map
- Correlation with Telescope hits
- Electronics rate



DQM4HEP

AIDA2020 WP5 / WP15

DQM4HEP developed within AIDA2020 WP5 (see MS67):

Task 5.4 Development of data quality and slow control monitoring

EUDAQ also developed within AIDA2020 WP5 as the DAQ solution (see MS46).

Plan an integration in the EUDAQ event builder

- Replace current EUDAQ monitoring
- Send event to DQM4HEP event collector before writing to disk

Once this is achieved, the two frameworks will provide a rather complete and robust suite for test beam data taking.

DESY slow control monitoring developped within AIDA2020 WP15.

Plan also to develop a DQM4HEP generic slow control module for the DESY test beam area, based on the SC software (see next talk by M. Wu).

DQM4HEP

Status - Ongoing work

- Current available version is `v01-04-04`:
  - Fully working version, used as proof of principle
  - EUDAQ-DQM4HEP interface not feasable (run control)
  - Module configuration (xml files) messy in case of a multiple host deployment
  - No clear seperation between online and offline tools
  - No documentation available for users ...

DQM4HEP

Status - Ongoing work

- Current available version is `v01-04-04`:
  - Fully working version, used as proof of principle
  - EUDAQ-DQM4HEP interface not feasable (run control)
  - Module configuration (xml files) messy in case of a multiple host deployment
  - No clear seperation between online and offline tools
  - No documentation available for users ...
- Refactoring on-going:
  - ✓ Separation of the framework into Core / Net / Online / Vis packages
  - ✓ Make the classes more C++11 like and re-usable
  - Necessary refactoring to allow for EUDAQ binding
    - ✓ Run control re-implemented
  - ✓ Core and Net packages have been fully re-implemented
  - ✚ Online package in development
  - ✗ Vis package not yet re-implemented

DQM4HEP

Ongoing work - More functionalities and projects

<u>Framework functionalities:</u>

- ✓ Custom interface to any DAQ run control (SOR/EOR/Status)
- Quality assessement in offline mode:
  - ✓ Configure your quality tests in an xml file
  - ✓ Run them on a ROOT file and output results (✓ file ✓console ✗ db)
  - ✚ Strong effort to develop built-in qtests for users (extensible)
- ✓ Database config: XML parser allows to fetch parameters from MySQL db
- ✚ Javascript interface: visualization and steering through web pages
- Documentation
  - ✚ User documentation (manual) written in parallel of ongoing development
  - ✓ Technical documentation (doxygen) generated/pushed online when a PR is merged
- ✓ Travis CI added for all packages

DQM4HEP

Ongoing work - More functionalities and projects

Framework functionalities:

- ✓ Custom interface to any DAQ run control (SOR/EOR/Status)
- Quality assessement in offline mode:
  - ✓ Configure your quality tests in an xml file
  - ✓ Run them on a ROOT file and output results (✓ file ✓console ✗ db)
  - ✚ Strong effort to develop built-in qtests for users (extensible)
- ✓ Database config: XML parser allows to fetch parameters from MySQL db
- ✚ Javascript interface: visualization and steering through web pages
- Documentation
  - ✚ User documentation (manual) written in parallel of ongoing development
  - ✓ Technical documentation (doxygen) generated/pushed online when a PR is merged
- ✓ Travis CI added for all packages

More projects:

- Development of DESY slow control monitoring with DQM4HEP
  - Can run continuousely and provide information to users at any time
- DESY beam line uses EUDAQ → DQM4HEP will comes for free on DESY beam line
- Looking for integration in other experiments ...

DQM4HEP

Ongoing work - More functionalities and projects

Framework functionalities:

- ✓ Custom interface to any DAQ run control (SOR/EOR/Status)
- Quality assessement in offline mode:
  - ✓ Configure your quality tests in an xml file
  - ✓ Run them on a ROOT file and output results (✓ file ✓console ✗ db)
  - ✚ Strong effort to develop built-in qtests for users (extensible)
- ✓ Database config: XML parser allows to fetch parameters from MySQL db
- ✚ Javascript interface: visualization and steering through web pages
- Documentation
  - ✚ User documentation (manual) written in parallel of ongoing development
  - ✓ Technical documentation (doxygen) generated/pushed online when a PR is merged
- ✓ Travis CI added for all packages

More projects:

- Development of DESY slow control monitoring with DQM4HEP
  - Can run continuousely and provide information to users at any time
- DESY beam line uses EUDAQ → DQM4HEP will comes for free on DESY beam line
- Looking for integration in other experiments ...

Timescale for the new version: ∼ June 2018 !

DQM4HEP

URLs and contact

GitHub collaboration (contributing, issues)

`https://github.com/dqm4hep`

Installation package (`v01-04-04`)

`https://github.com/DQM4HEP/dqm4hep/releases/tag/v01-04-04`

Slack channel (Announcements, forum, management)

`https://dqm4hep.slack.com`

Documentation (ongoing, be patient !)

Read the docs : `http://dqm4hep.readthedocs.io`
Doxygen : `https://dqm4hep.github.io/dqm4hep-doxygen/`

Contact us!

- R. Ete (`remi.ete@desy.de`)
- A. Pingault (`antoine.pingault@ugent.be`)
- T. Coates (`tc297@sussex.ac.uk`)