Overview

- DQM4hep v01-04-04 (Milestone 67)

- Ongoing and future work
  - Framework refactoring
  - Framework extension
  - Experiment integration
Online monitoring framework for HEP experiments

- Online data analysis (Module)
- Data quality assessment (QualityTest, MonitorElement)
- Interact with the DAQ system (RunControl, EventCollector)
- Distributed system (TCP/IP)
- GUI for shifters (process management, data visualization, run control)

- Not experiment specific

Central concepts

- Plugin system
  - User C++ class loaded at runtime via shared libraries
  - Example: analysis module, event streamer, etc ...

- Event streaming
  - Event type is abstract
  - Event serialisation as a plugin
Online architecture

- **DAQ**
- **Event collector servers**
- **Analysis modules**
- **Monitor element collector servers**
- **Run Control server**
- **Run control GUI**
- **Monitoring GUIs**
- **Send event (DIM)**
- **Send monitor elements (DIM)**
- **Send/query event (DIM)**
- **Post event data (/dev/shm)**
- **Start/End of run (Http POST)**
- **Start/End of run (DIM)**
- **Remote process management (DIM)**
- **External data source (e.g. Slow control)**

**Tasks**
- Developer
- Engineer
- Shifter
- DQM - Internal

**DQM4HEP**
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:

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

![Reconstructed energy](image)
Online monitoring interface (Qt Gui)
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
Main limitations:

- Difficult/impossible linking with DAQ system
- Qt vis requires ROOT compiled enable-qt

Secondary limitations:

- Clumsy deployment of analysis module (steering file)
- No separation between offline/online tools
- No documentation
- Heavy logging library (log4cxx)

A refactoring process of the framework has been triggered.

- Fix the limitations
- Add a few more important features
- Online architecture remains unchanged!
DQM4hep incoming version
Ongoing work - refactoring

Split packages into different pieces

- DQMCore → dqm4hep-core + dqm4hep-net + dqm4hep-online
- DQMViz → dqm4hep-qt-viz
- New packages:
  - dqm4hep-doc: Online user documentation (readthedocs)
  - dqm4hep-doxygen: Online code documentation (Doxygen)

Packages status:

- dqm4hep-core: Fully re-implemented + extended
- dqm4hep-net: Fully re-implemented
- dqm4hep-online: Re-implementation started + extended
- dqm4hep-qt-vis: Not yet created
DQM4hep incoming version
Ongoing work - dqm4hep-core

What’s new?

- Development of builtin quality tests (ongoing)
  \[\rightarrow v01-04-04: \text{no builtin QTest!}\]
- Database (MySQL) interface
- Smart XML parser: can pull parameters from database
  \[\rightarrow \text{All analysis modules can use central settings from database}\]
- New logging interface (spdlog): lighter, faster, header only
  \[\rightarrow \text{Nothing to compile. Save 30 minutes of compilation!}\]
- New builtin event streamer: RootEventStreamer
  - Event = TObject type
  - Use ROOT streaming facility to read/write event
  - Any experiment using ROOT for their event definition can use this streamer!
New executable: dqm4hep-run-qtests

- Input arguments
  - ROOT file
  - XML file describing quality tests to run
  - Quality report output file name

- Process all QTests in the XML file
- Can output a json file with QReports

Possible console output:

<table>
<thead>
<tr>
<th>NAME</th>
<th>QTEST</th>
<th>STATUS</th>
<th>QUALITY</th>
<th>MESSAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>DblGaus_Mean15.RMS2.RMS5</td>
<td>MeanAround15Short</td>
<td>SUCCESS</td>
<td>0.998484</td>
<td>Expected 15, got 15.0019</td>
</tr>
<tr>
<td>Gaus_Mean10.RMS2</td>
<td>MeanAround10Long</td>
<td>SUCCESS</td>
<td>0.997348</td>
<td>Expected 10, got 10.0133</td>
</tr>
<tr>
<td>Gaus_Mean10.RMS2.bck</td>
<td>MeanAround10Short</td>
<td>FAILURE</td>
<td>0.388153</td>
<td>Expected 10, got 5.6458</td>
</tr>
</tbody>
</table>

Adjustable behavior:

- Fail command if at least 1 WARNING
- Print only or sort output by quality value
- Print only the tests with status FAILURE and WARNING
Define a top level interface to DIM:

- Domain name servers (DNS)
  - No connection to host/port
  - Connection to service name
  - DIM DNS process:
    - map (host,port) ↔ service name

- 3 networking patterns
  - Command
  - Service
  - Request
Re-implementation ongoing:

- ✓ Event source / collector / client
- ✓ Run control + extension:
  → Custom connection to DAQ run control
  → Possible link with EUDAQ
- X Analysis/standalone module
- X Module Api
- X Monitor element source / collector / client
- X Remote process management

New features:

- Application are monitored (CPU/bandwidth/memory)
- New process: Online Manager
  - Collect application logs and statistics
  - Single instance per deployment → simplify setups
DQM4hep incoming version

Ongoing work - visualization

Re-implementation not yet started

- Drop ROOT-Qt vis
  - Avoid re-compiling ROOT with enable-qt flag!
  - Move to Qt 5.5 (QtCharts)

- Monitoring interface
  - Keep the same interface
  - Implement mapping: ROOT class → QtChart class

- Other interfaces to implement
  - Process manager client
  - Logging client
  - Application performance client (CPU/bandwidth/memory)
Event entry point:
DQM4hep incoming version
EUDAQ integration (not started)

Event entry point:
  ● Modify base class eudaq::DataCollector

---

To DQM4hep

Rémi Ete — DESY — April 24, 2018 — Page 15
Event entry point:
- Modify base class `eudaq::DataCollector`?
- Inherit from `eudaq::OnlineMonitor`?
Run control interface:
- Implement `eudaq::CommandReceiver` and `dqm4hep::RunControlInterface`
- Run in DQM4hep central run control to receive sor/eor/config from EUDAQ

Code integration:
- Download (github), configure and compile DQM4hep from EUDAQ
- Compile EUDAQ and link libs/bins against DQM4hep

Package for EUDAQ tools: `dqm4hep-eudaq-tools`
- interface to run control
- event streamer + interface to collector
- builtin analysis modules (current EUDAQ monitor)
Current integration with v01-04-04: SDHCAL and AHCAL
→ Calorimeters and LCIO so far ...

DQM4hep always advertised as a generic framework
→ Need various experiment integration as proof of principle (detector/data)

Possible integrations with incoming version:
• DESY slow control system (AIDA WP15)
• DREAM collaboration (EDM test, combined)
• AHCal + HGCal (combined, EUDAQ test)

→ Looking for new experiment integration
→ EUDAQ integration means new DQM4hep users
GitHub collaboration

- https://github.com/dqm4hep

Installation package (v01-04-04)

- https://github.com/dqm4hep/dqm4hep

Slack channel (Announcements, issues, management)

- https://dqm4hep.slack.com

Contact us!

- R. Ete (remi.ete@desy.de)
- A. Pingault (antoine.pingault@ugent.be)
- T. Coates (tc297@sussex.ac.uk)
- General mailing list: dqm4hep@desy.de
DQM4hep incoming version
ROOT Eve online event display

- Display run status
- Online event display
  - Events from collector
- Display geometry
  - DD4hep
  - GEAR
  - ROOT
  - GDML
- Scene projection