Quasi-online integration with DQM4HEP

Tom Coates, Remi Ete,
Antoine Pingault, Fabrizio Salvatore

University of Sussex

RD_FA Collaboration Meeting

2018-07-06

Data quality monitoring software

in a nutshell...

Main goals of DQM systems in high-energy physics:

- Evaluate data quality and alert users of possible anomalies
  - Is the data how you expect?
  - Is the data comparable to previous data?
  - Quick feedback from detector(s)
- Online and offline monitoring
  - Distributed system (TCP/IP)
  - Quality testing automation
  - Event display
  - Visualisation and interface

Data quality monitoring software

in a nutshell...

Data is the core of such systems. But…

- Existing frameworks are highly dependent on event data model
- This leads to duplicated software
- Test-beam setups lead to *ad hoc* software solutions

The DQM4HEP framework

Central ideas

Plugin system

- User-specific logic is encapsulated in **plugins**
- Plugins loaded at runtime
- Plugins are non-intrusive

Abstract event data model (EDM)

- No assumed event data model → abstracted and user-defined
- Event streamers or file readers implemented as **plugins**

The DQM4HEP framework

Central ideas

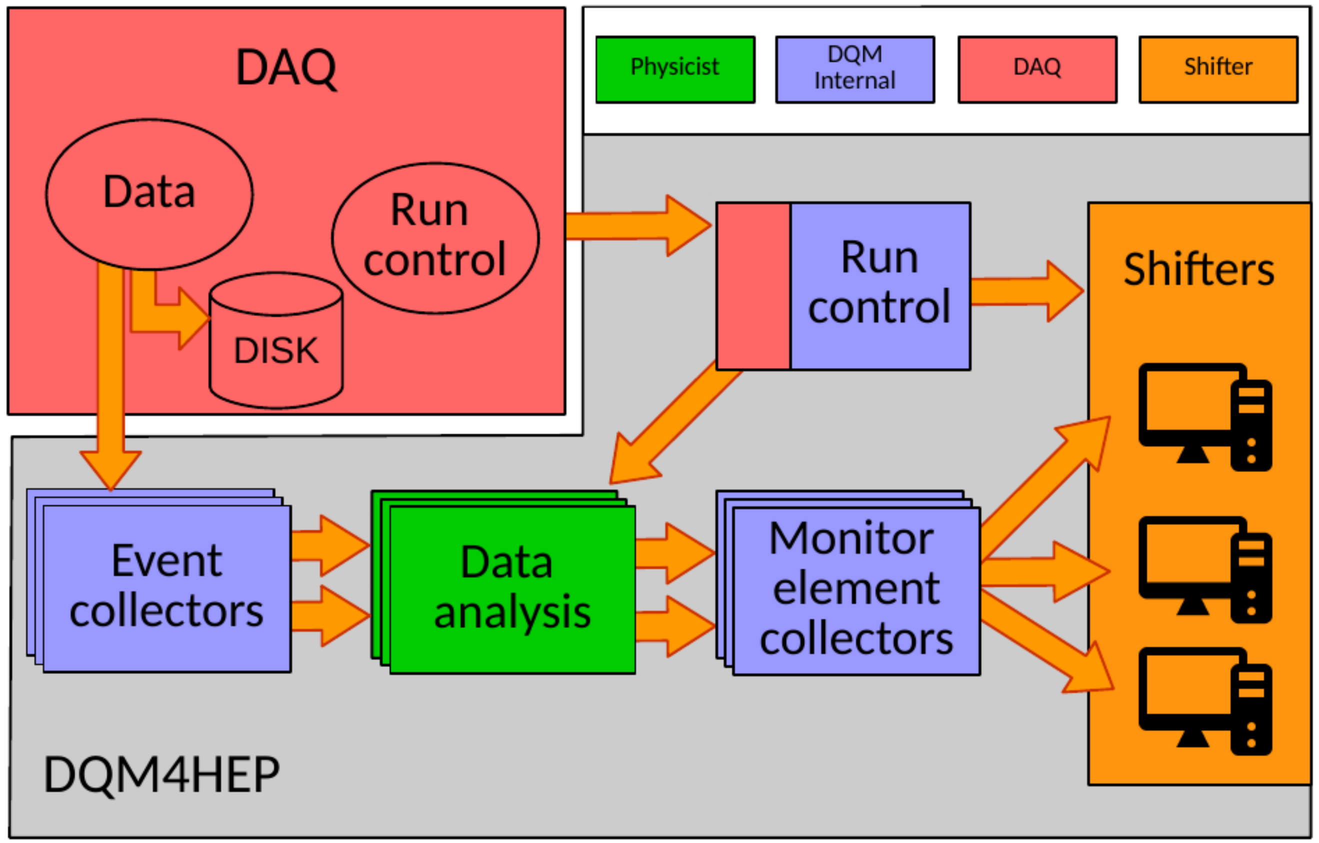
Online:

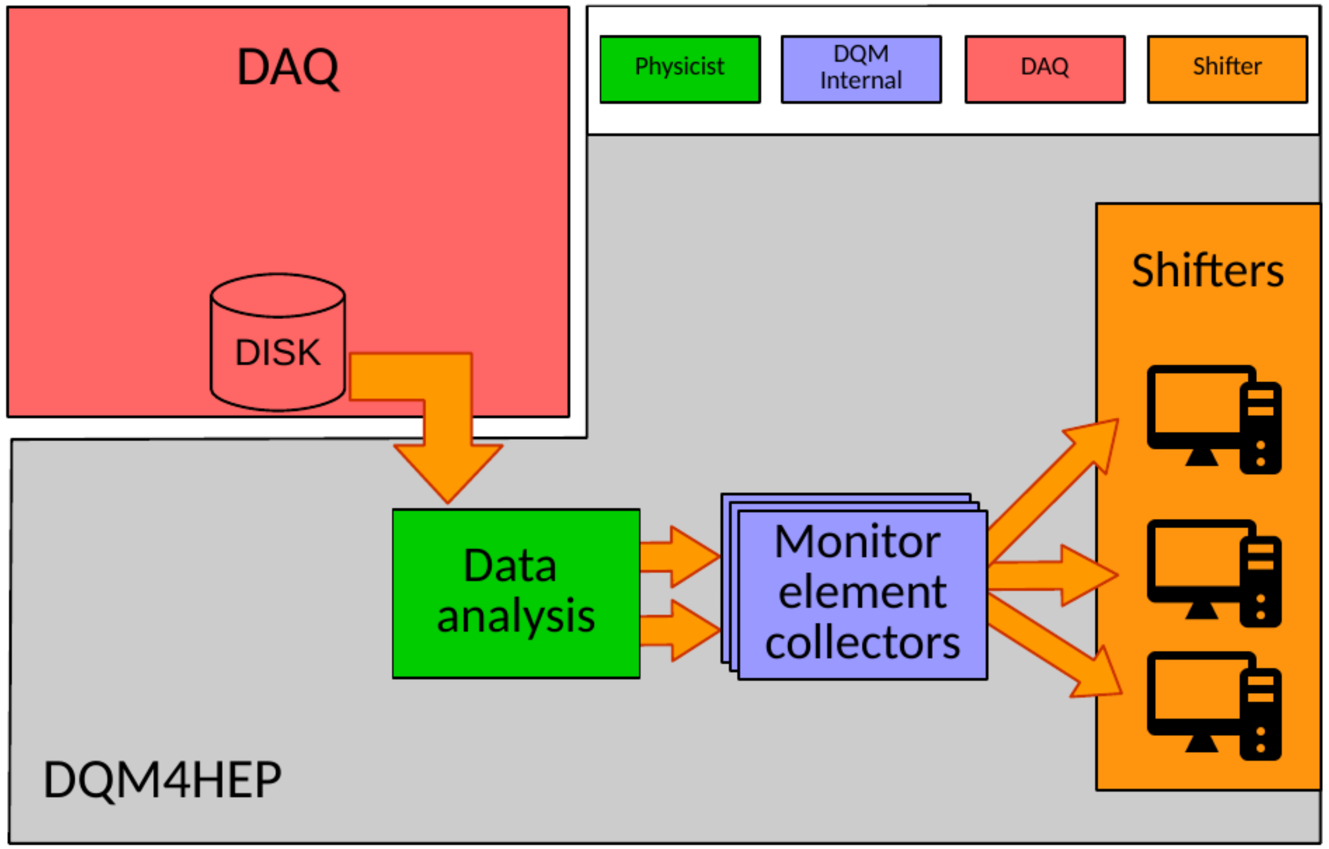
- Interface to DAQ systems
- Online data processing
  - Data monitoring from DAQ (analysis module)
  - Slow control monitoring (standalone module)
  - Data re-processing from file (event reader)

Offline:

- General purpose data monitoring
  - Data quality assertion and reporting
  - Comparison with reference data

Online/offline architecture





Components for offline monitoring

- **File reader for each data type** – Data must be decoded from the storage file format to be readable by DQM4HEP. We implement a file reader (a type of plugin) to open files, parse data into C++ objects, and send them through the network. Each different file type requires its own file reader.
- **Analysis module for each set of graphs/plots** – Plugins which perform the required data analysis, e.g. creating histograms or hitmaps. The "meat" of the analysis part of the framework.
- **Quality tests** – If required, DQM4HEP can perform quality tests on monitored data. There is a suite of existing quality tests to get mean, RMS, median of data, compare data to references, perform fits, etc. Additional quality tests can be written if needs are not covered by the existing tools.

University of Sussex

DESY UNIVERSITEIT GENT IPNL AIDA 2020

Progress so far

```
StatusCode SiPMFileReader::readNextEvent() {
  EventPtr pEvent = GenericEvent::make_shared();
  GenericEvent *pGenericEvent = pEvent->getEvent<GenericEvent>();

  std::string eventDelimiter = ";";
  std::string currentEventString;
  std::vector<float> eventContainer;

  while (eventContainer.size() == 0) {
    if (inputFile.eof() == true) {
      dqm_info("Reached end of file");
      return STATUS_CODE_OUT_OF_RANGE;
    }

    std::getline(inputFile, currentEventString);
    dqm4hep::core::tokenize(currentEventString, eventContainer, eventDelimiter);
  }

  if (eventContainer.size() != 66) {
    dqm_error("Event has wrong number of members");
    return STATUS_CODE_FAILURE;
  }

  std::vector<int> ev_eventNum = {eventContainer[0]};
  pEvent->setEventNumber(ev_eventNum[0]);

  std::vector<int> ev_time = {eventContainer[1]};
  pEvent->setTimeStamp(core::time::asPoint(ev_time[0]));

  eventContainer.erase(eventContainer.begin(),eventContainer.begin()+2);
  pGenericEvent->setValues("Channels", eventContainer);

  onEventRead().emit(pEvent);
  return STATUS_CODE_SUCCESS;
}
```

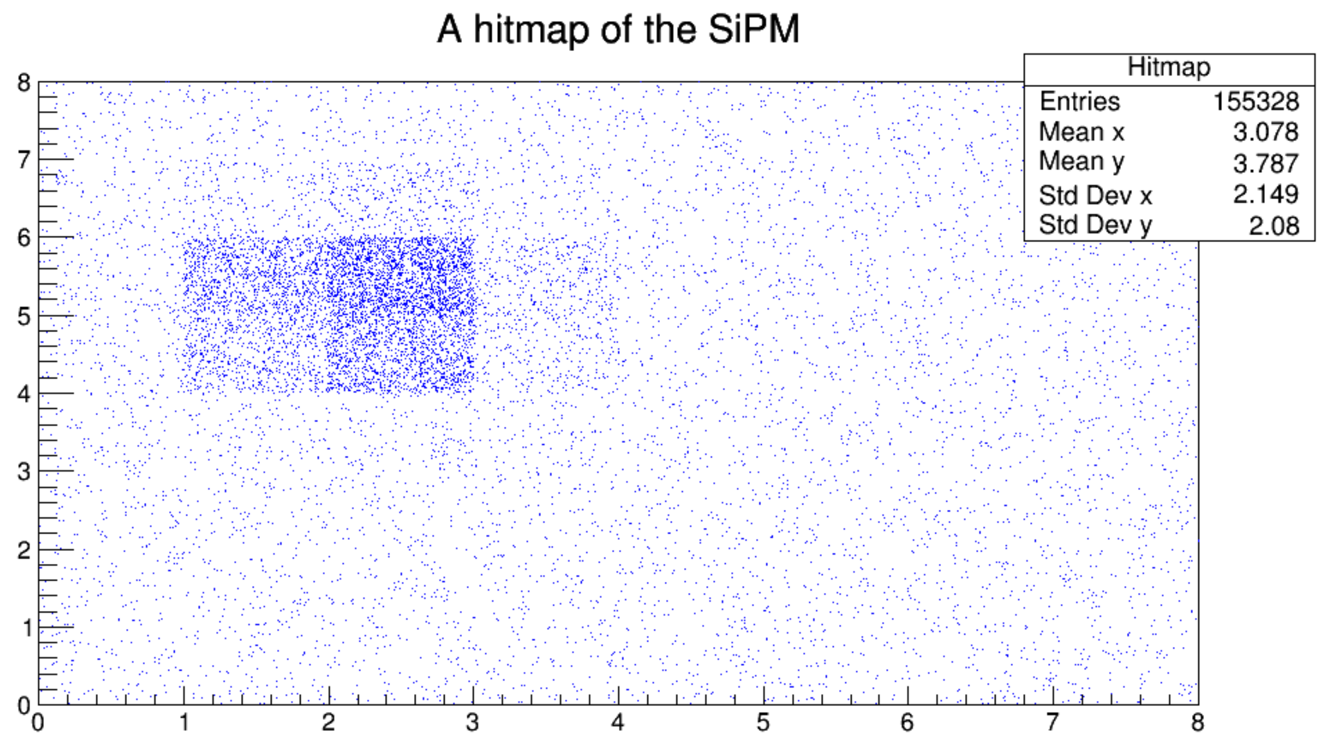
```
void SiPMHitmap::process(core::EventPtr pEvent) {

  if(nullptr == pEvent) {
    dqm_warning("Event pointer is invalid - skipping this event");
    return;
  }

  std::vector<float> eventChannels;
  core::GenericEvent *pGenericEvent = pEvent->getEvent<core::GenericEvent>();
  pGenericEvent->getValues("Channels", eventChannels);

  int i = 0;
  int j = 0;
  int channelNum = 0;

  for(i=0; i<8; i++) {
    for(j=0;j<8; j++) {
      m_pHitmap->objectTo<TH2I>()->Fill(i,j,eventChannels[channelNum]);
      channelNum++;
    }
  }

}
```

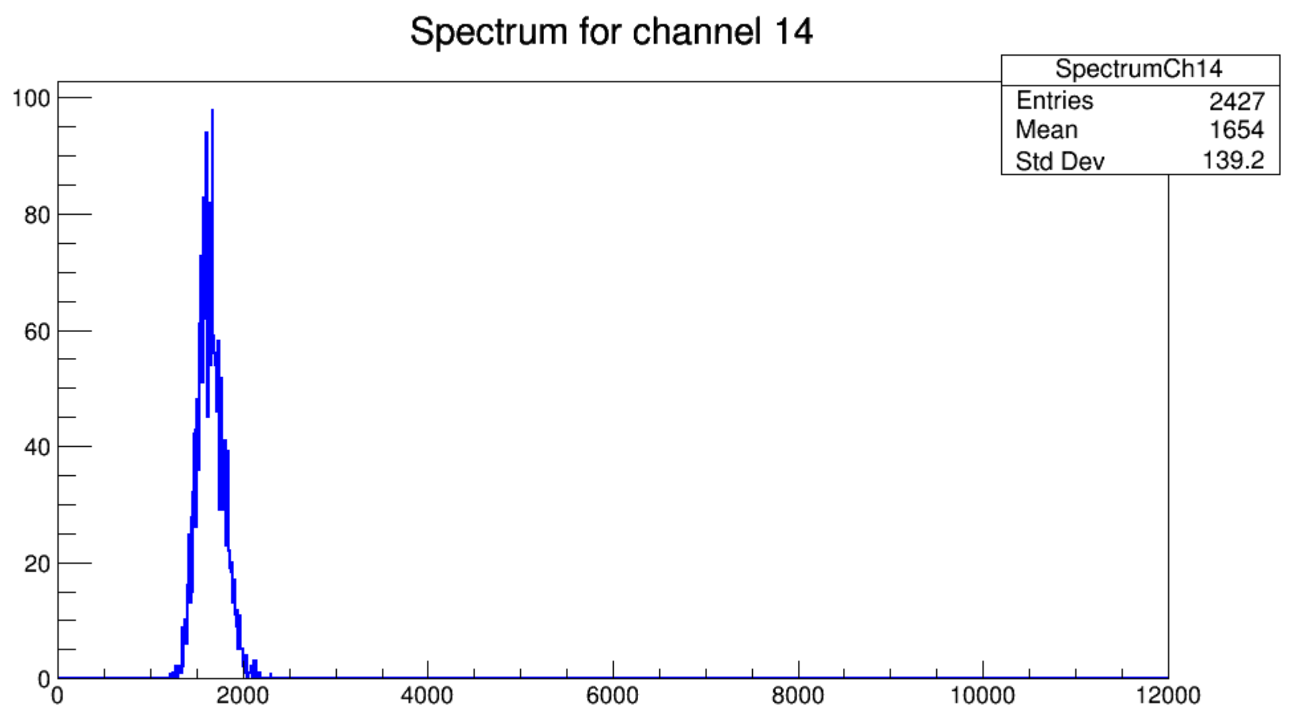
```
void SiPMChannelSpectra::process(core::EventPtr pEvent) {

  if(nullptr == pEvent) {
    dqm_warning("Event pointer is invalid - skipping this event");
    return;
  }

  std::vector<float> eventChannels;
  core::GenericEvent *pGenericEvent = pEvent->getEvent<core::GenericEvent>();
  pGenericEvent->getValues("Channels", eventChannels);

  for (int i=0; i<64; i++) {
    m_pSpectrum[i]->objectTo<TH1>()->Fill(eventChannels[i]);
  }

}
```

University of Sussex

DESY UNIVERSITEIT GENT IPNL AIDA 2020

Progress so far

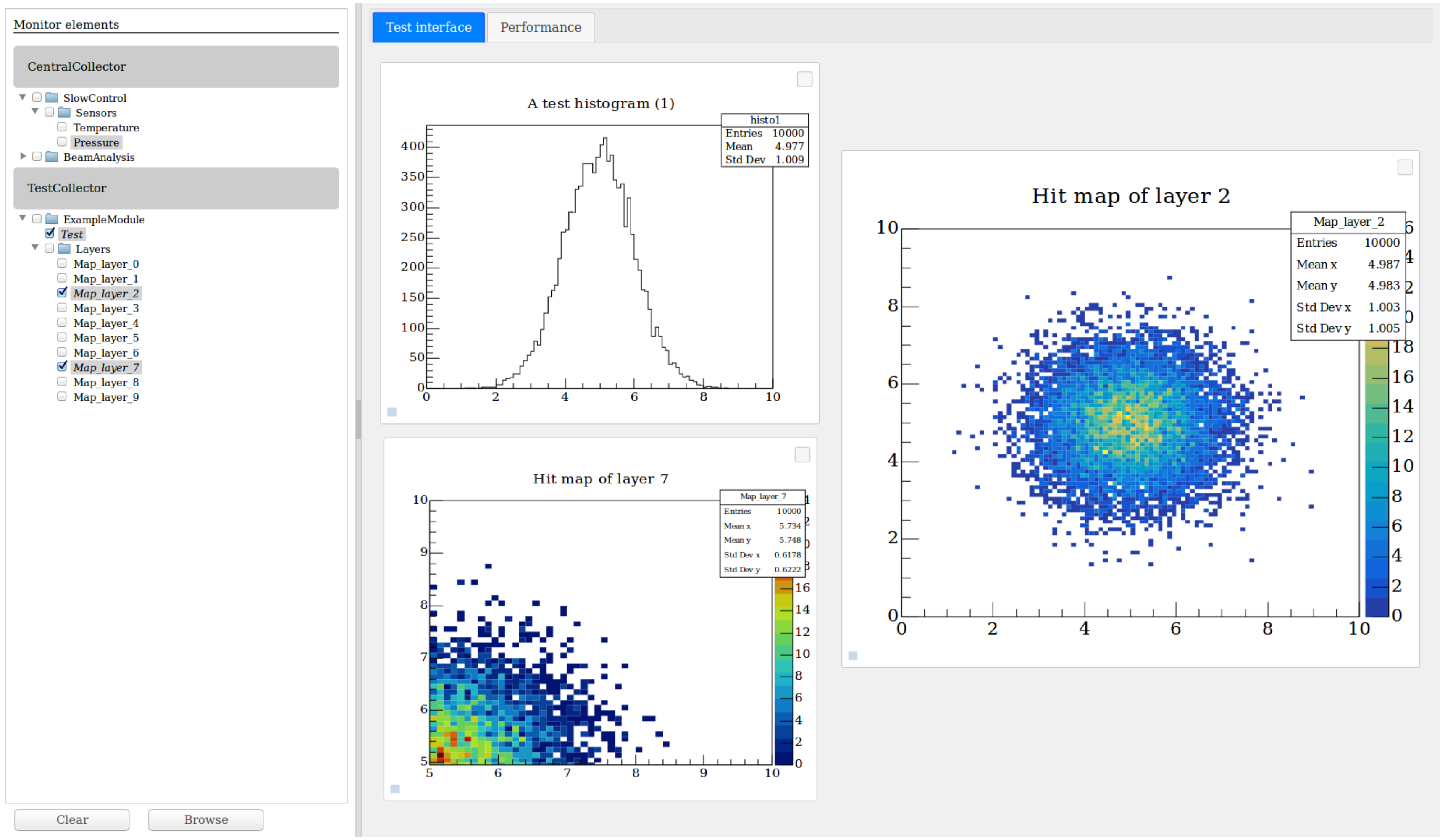




University of Sussex

Web monitoring interface

Ongoing work…



histo1	
Entries	10000
Mean	4.977
Std Dev	1.009

Map_layer_2	
Entries	10000
Mean x	4.987
Mean y	4.983
Std Dev x	1.003
Std Dev y	1.005

Map_layer_7	
Entries	10000
Mean x	5.734
Mean y	5.748
Std Dev x	0.6178
Std Dev y	0.6222

University of Sussex

Summary

- Integration of the various subdetectors has started, and is going well
- SiPM finished! Two basic analysis modules, writing more is easy and quick (can even be done while testbeam is in progress, if needed)
- Calorimeter file reader underway, will be ready to go soon
- Development of DQM4HEP is going quickly, on-target for readiness for CERN testbeam
- Still need information on data type and decoding for drift chamber, preshower, and muon
- I will be attending the testbeam to help ease integration and teach shifters
- Current code is available at https://github.com/tcoates3/dqm4hep-dream/tree/dragonfruit/

Thank you

University of Sussex

RD_FA Collaboration Meeting

2018-07-06