PROCEEDINGS OF SCIENCE

Data Quality Monitoring of the CMS Pixel Detector

Petra Merkel[1]*
Purdue University
Department of Physics, 525 Northwestern Ave, West Lafayette, IN 47906 USA
E-mail: petra.merkel@cern.ch

We present the CMS Pixel Data Quality Monitoring (DQM) system. The concept and architecture are discussed, as well as first experience with the system during global CMS cosmic data taking.

17th International Workshop on Vertex detectors
Utö Island, Sweden
July 28th - 1st August, 2008

[1] Speaker
* On behalf of the CMS Pixel Group

PoS(VERTEX 2008)041

1.Introduction

The DQM is a very important tool to ensure high quality data taking. It needs to be robust, reliable, and fast. The DQM tools in CMS [1] are centralized across sub-systems (sub-detectors, trigger, and physics analysis objects), and are implemented in the CMS software framework (CMSSW) [2,3]. The DQM is performed by evaluating histograms, or Monitoring Elements (ME's), which are filled with information from raw as well as reconstructed data. It runs both online in real time on a fraction of events during data taking, and offline on the full statistics, but reduced granularity, during data reprocessing.

2.DQM architecture in CMS

In the online case almost all sub-systems' applications are run on computers in the CMS surface control room, accessing events from the Storage Manager Proxy Server, which delivers a configurable rate of events of up to 25 Hz. This data stream can be filtered for individual DQM applications with bits set by the High Level Trigger (HLT), which allows for example the Pixel DQM application to look only at events with at least one track candidate in order to enhance the physics content in the histograms. For a few histograms, where the full data statistics is needed, such as the monitoring of data corruption and errors in the front-end readout crates, DQM applications are implemented directly in Filter Units of the HLT filter farm [4].

For every sub-system a set of C++ programs, the so-called DQM sources fill histograms, which are then picked up and further processed by the DQM client, a dedicated C++ program. These are the sub-system applications in Figure 1; they are launched and monitored by the central CMS data acquisition run control application. The client picks up the histograms from the sources and analyzes them by creating summary histograms, and by defining and applying quality tests. It can write test results into the online data base, and it writes out all results and histograms in root files, which are centrally archived and can be viewed by the central web based CMS DQM GUI [5]. More detailed information on the CMS DQM system can be found in [1].

In the offline DQM case, in which the same applications as in the online case are run with partially modified parameters, the main purpose is to verify the quality of the data during offline processing at Tier-0 at CERN, and to finalize the data certification bits on which all physics analyses will subsequently rely. Here the main advantages are full statistics, and access to fully calibrated and reconstructed objects.

3.Pixel specific DQM

The CMS Pixel detector contains ~67 million pixels that need to be monitored. The basic monitoring unit for the DQM is a module, which comprises between 8,000 and 66,000 pixels each. The entire Pixel detector consists of 1440 modules [6]. Such a finely segmented detector calls for a highly automized system of quality assurance. The DQM application monitors the detector performance for physics data taking. Due to the

extremely fine granularity of the Pixel detector, a single pixel cell measures 100 x 150 μm^2, a large statistic of charged particles crossing the detector is needed, and therefore a fast application that keeps the CPU and memory usage of the online computing system low.

The Pixel DQM has proven to be a versatile and powerful tool during the commissioning of the detector, as well as during early cosmic data taking together with the rest of CMS. It is used both in an online application in real time during data taking, as well as as an offline tool running on the full statistics just after full processing of the data at the Tier-0 computing centre at CERN. Many of the tools of the Pixel DQM are in common with the SiStrip applications and therefore a close collaboration is maintained [7].

Within the pixel DQM source code a range of different programming languages and protocols is used. Figure 2 is illustrating the interplay between the various languages in order to assure quasi deadtime free operation and user interaction. The html GUI communicates with the C++ client code via java script functions. An ajax engine[4] ensures asynchronous communication, which minimizes the dead time in the browser following a user request. Only modified parts in the browser get updated, thus minimizing the response delay. The ajax engine communicates with the client code, which is embedded in a web server architecture, via xml http requests.

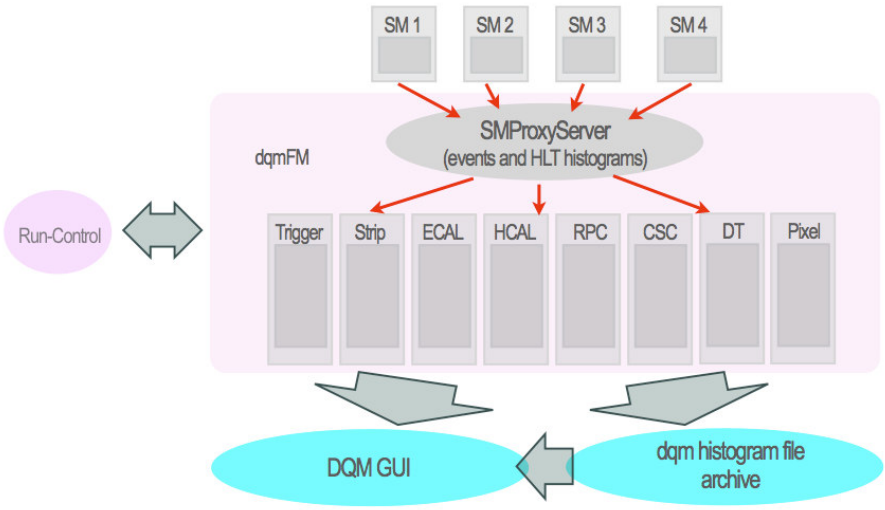

Figure 1: Scheme of the CMS DQM architecture

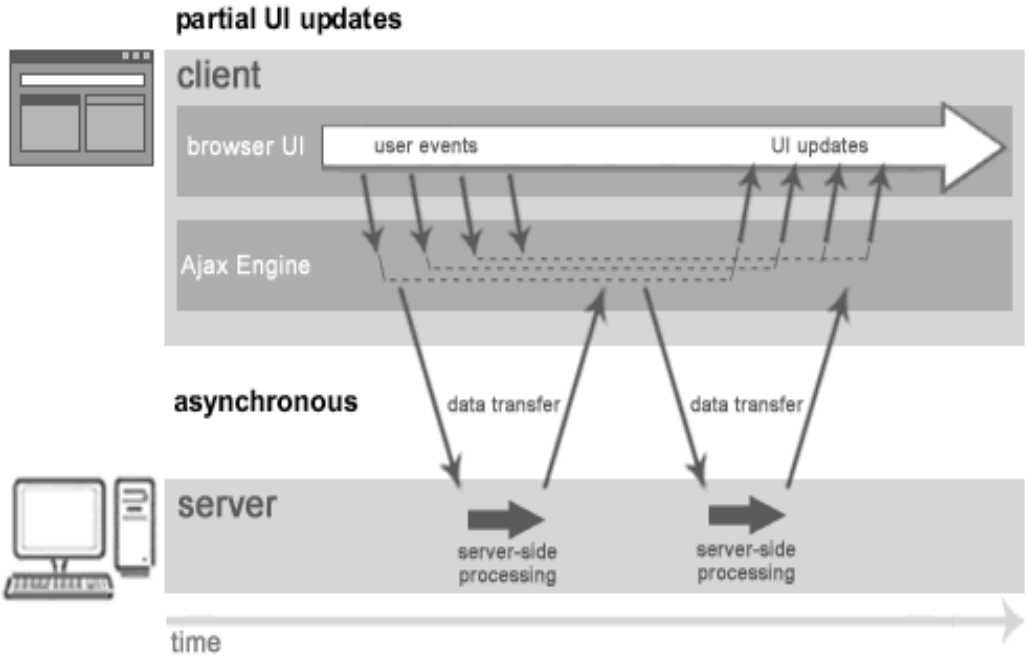

Figure 2: Scheme of asynchronous communication between DQM client and Pixel GUI

4.Pixel commissioning via DQM

The following DQM source applications are available for the Pixel detector:

- RawData error source: monitor data corruption and other readout errors from the front-end data boards (FED's) and the data unpacking.
- Digi source: monitor multiplicity, raw charge and position of digis (uncalibrated single pixel hits above hardware threshold).
- Cluster source: monitor multiplicity, gain corrected charge, position, and size of clusters (gain corrected and clustered digis).
- RecHit source: monitor multiplicity, charge, position, and size of reconstructed hits (Lorentz angle corrected clusters).
- Track source: monitor residuals of reconstructed pixel hits associated to tracks, multiplicity, charge, position, and size of reconstructed pixel hits on and off tracks.
- PixelAlive calibration: detect alive and dead pixel cells in front-end readout chips (ROC's) via internal charge injection.
- Scurve calibration: measure charge response threshold for each pixel cell in ROC's via internal charge injection scan.
- Gain calibration: measure the gain of each pixel cell in the ROC's via internal charge injection scan. The information is used to translate the measured raw charge in ADC counts into a more uniform charge in electrons.

The DQM pixel client is then running in sequence with the sources, both calibration as well as data sources are possible, depending on the type of the run. It performs frequent

analyses at the end of every lumi section (typically every few minutes) and at the end of each run, such as application of quality tests, comparison with reference histograms, and a compilation of key distributions in summary ME's. In addition, a logic is applied that defines the goodness of the data for future analysis from the view of the Pixel detector. These global data quality flags are based on information from the data acquisition system, the detector slow control system, and on the DQM results.

The DQM data are visualized with the help of the central CMS DQM GUI [5], which is used by a central shifter to monitor the quality of the data taking, as well as sub-system specific shifters and experts. In there all the ME's can be viewed and overlaid to reference histograms. This can be done both for the ongoing data taking run as well as for archived runs, and for the processed DQM data at Tier-0. In addition, a pixel specific GUI was developed for the use of pixel detector experts, which is able to interact with the pixel client at run time, see Figure 3. This allows for more detailed plots, tree browsing of histograms, list of bad modules (flagged by the automized quality tests), and the use of a so-called TrackerMap, a graphic display of the status of the entire Pixel detector based on a monitoring variable, which can be selected by the user, see Figure 4. In the future, different topological views will be provided in the TrackerMap, so besides the graphical detector representation also views by power supply and readout channel will be accessible, allowing for a fast debugging tool for macroscopic problems.

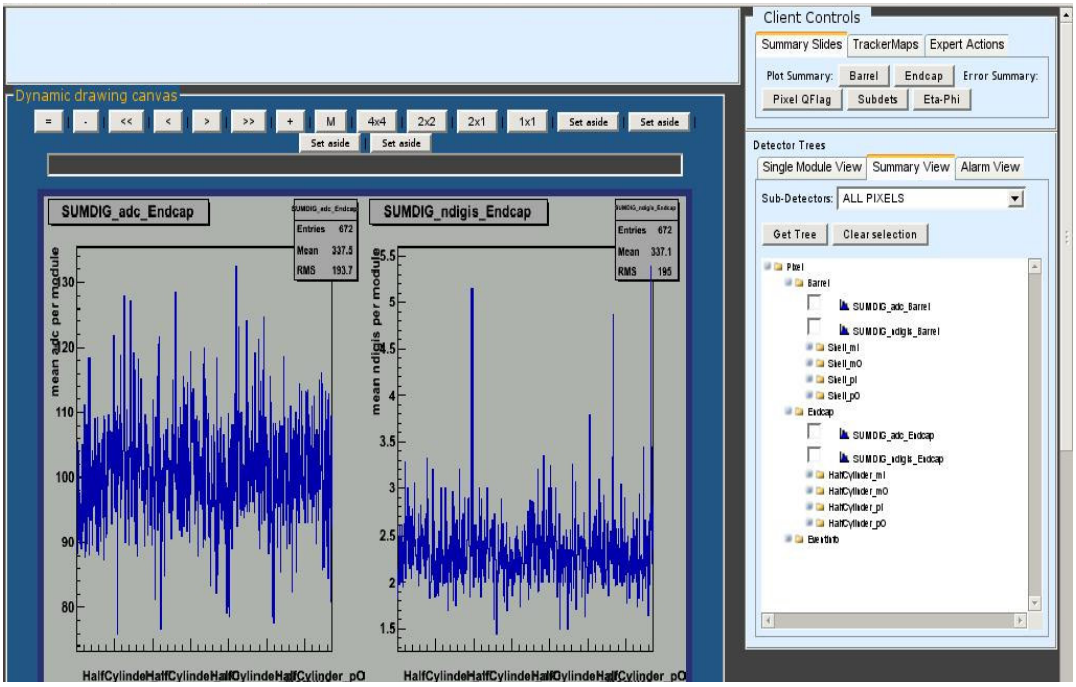

Figure 3: Snapshot of the Pixel expert DQM GUI showing digi summary plots for all modules in the endcaps. Expert control buttons and a tree structure for browing the detector hierarchy on the right hand side allow fast navigation and problem debugging.

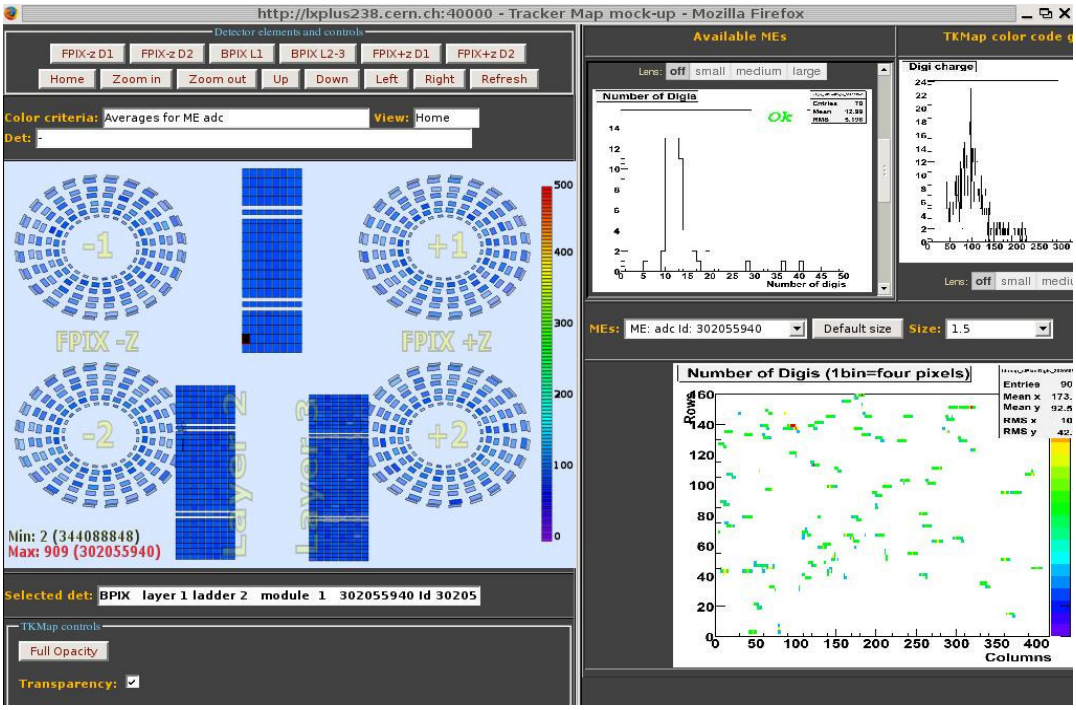

Figure 4: A snapshot of the TrackerMap, a synoptic geometric overview of the entire Pixel detector allowing for fast access to problematic modules and their distributions (on the right).

In addition to the main DQM client, the Pixel DQM framework also provides the so-called historic information client, which is a C++ application that processes ME's from the DQM sources, analyzes the distributions, and writes some key numbers into the data base. A subsequent application then retrieves these numbers from the data base and fills historic trend plots, with which the Pixel experts can monitor the stability of the detector's behaviour over longer periods of time.

5.First experience with cosmic data taking

The Pixel detector was included in the global cosmic data taking runs of CMS starting from July 2008. From the start the online DQM was available to ensure that basic problems with the data out of the pixel detector could be spotted in real time. Due to the lack of statistics during cosmic data taking in the Pixel detector, especially in the endcaps, only a few fundamental quantities could be monitored in a reasonable way, namely the digi and cluster charge and occupancy, as well as the rate of noisy modules, and errors in the readout stream and data unpacking. In the offline DQM processing at Tier-0 a few more quantities such as residuals and hit multiplicities from pixel hits on and off tracks were added thanks to the availability of the full statistics.

CMS has taken cosmic data during a few weeks with no magnetic field, as well as four weeks with full magnetic field of 3.8 T. During these data taking periods the surface control room as well as the offline operation centre were fully manned with shifters around the clock. Besides a central CMS DQM shifter, who monitored a small, well-

defined subset of histograms for each sub-system, mostly summarizing the global status of each sub-system, the Pixel group also had a Pixel expert on shift at all times. Part of this person's tasks was to periodically study a pre-defined subset of Pixel DQM histograms in more detail than the central shifter. The goal was to be able to spot problems in the data taking, such as data corruption, and newly broken or noisy modules. Interspersed with the cosmic data taking was the calibration program with which to optimize the Pixel detector for physics data taking. The DQM applications were used to analyze data taken during stand-alone pixel calibration runs such as PixelAlive, Scurve and Gain calibrations, which were briefly explained in section 5 of this paper. In addition they assisted with prompt feedback during dedicated timing scans, in which the relative timing of the Pixel detector's data readout with respect to the rest of CMS was optimized.

The goodness of the data are judged both, from the online DQM results as well as from the more detailed offline processing. The final good run flag will be based on information from these different sources: data acquisition, detector slow control for power supplies and cooling, as well as online and offline DQM processes. These flags will then be loaded into the data base, and used to define data-sets for physics analyses.

6.Summary

An overview of the general CMS DQM architecture has been given, followed by the dedicated Pixel DQM system. The highlights of these applications are the modular architecture, the centralized, web-based GUI and histogram archiving. The system has been successfully operated during the several weeks of cosmic data taking in CMS in 2008. Automization and shift operation have been already proven to work, and are continuously optimized. In particular, the Pixel DQM has played an invaluable role in the commissioning of the Pixel detector, as well as the early detection of data corruption problems and noisy channels during cosmic data taking.

References

[1] CMS Collaboration, *The CMS Experiment at the CERN LHC*, *JINST* **3-S8004-2008**

[2] CMS Collaboration, *CMS Physics Technical Design Report Volume I, Detector Performance and Software*, CERN/LHCC/2006-001, CMS TDR 8.1, 2 February 2006

[3] C. Leonidopoulos, E. Meschi, I. Segoni, G. Eulisse, D. Tsirigkas, *Physics and Data Quality Monitoring at CMS*, Proceedings of XV International Conference on Computing in High Energy Physics Conference (CHEP06), Volume I (106)

[4] CMS Collaboration, *The Trigger and Data Acquisition project, Volume II*, CERN/LHCC/2002-26, CMS TDR 6.2, 15 December 2002

[5] CMS Collaboration, *CMS Data Management Webtools,* https://twiki.cern.ch/twiki/bin/view/CMS/DMWebTools

[6] CMS Collaboration, *CMS Tracker Technical Design Report*, CERN/LHCC 98-6, CMS TDR 5, 15 April 1998; CMS Collaboration, *Addendum to the CMS Tracker TDR*, CERN/LHCC 2000-016, CMS TDR 5 Addendum 1, 21 February 2000

[7] S. Dutta, *The Data Quality Monitoring of the CMS Experiment: the Tracker Case*, proceedings for the 11th Topical Seminar on Innovative Particle and Radiation Detectors (IPRD08) to be published in Nuclear Physics **B** (Proceedings Supplement)