EPJ Web of Conferences **214**, 05030 (2019) https://doi.org/10.1051/epjconf/201921405030
CHEP 2018

A Historic Data Quality Monitor (HDQM) tool for the CMS Tracker Detector

Aristotelis Kyriakis[1,*] on behalf of the CMS Collaboration

[1]Institute of Nuclear and Particle Physics, NCSR DEMOKRITOS, 15310 Agia Paraskevi, Greece

Abstract. Monitoring the time evolution of data related observables is important for the successful operation of the LHC experiments. It permits keeping control on data quality during LHC running and also effectively checking the influence on data of any detector calibration performed during the year. The Historic Data Quality Monitor (HDQM) of the CMS experiment is a framework developed by the Tracker group of the CMS collaboration that permits a web-based monitoring of the time evolution of interesting quantities (i.e. signal to noise ratio, cluster size) in the Tracker Silicon micro-strip and pixel.

1 Objectives and Architecture

The main goal of the Historic Data Quality Monitor (HDQM) framework of the CMS experiment [1] is to provide web-based, dynamic and interactive, trend plots to facilitate the monitoring of the evolution of various CMS Tracker [2][3][4] DQM quantities (i.e. cluster size, signal to noise ratio) by experts. The visualization is performed through a distributed three-tier architectural model. In the lower database tier the data (various quantities describing the performance of the CMS tracking system) are stored in the form of ROOT objects [5]. The middle logic tier is based on a set of python [6] scripts (data harvesting) that read the data and generate JSON [7] files for the trend of each input quantity. The higher presentation client tier is a web portal which consists of a front-end HTML [8] module and a JavaScript [9] module (Figure 1). This last tier is responsible for fetching JSON data from the middle logic tier to dynamically create the trend plots.

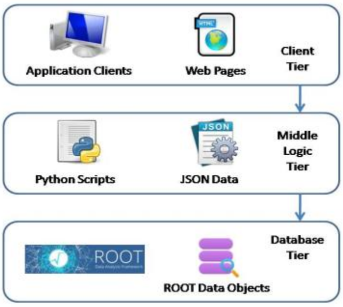


Fig. 1. The three-tier architectural model of the HDQM framework that consists of the higher client presentation tier, the middle logic tier and a lower database tier.

---

[*] e-mail: aristoteles.kyriakis@cern.ch

2 Data Harvesting

There is a main python script, which performs the data harvesting and generates the trend values as a function of the CMS run number. This script retrieves the requested information from the Data Quality Monitor (DQM) system [10] of the CMS experiment and saves a set of JSON files that are used for the trend visualization. Figure 2 gives an example of such a JSON file.

```
{
    “clusterStoN_ONTk_TIB_mpv”: [
        {
            “run”: 295315,
            “y”: 18.1175,
            “x”: 1.0,
            “htitle”: “S/N (mpv)”,
           “yErr”: 0.0032,
           “yTitle”: “S/N (mpv)”
        },
    …………………
    ]
}
```

Fig. 2. Example of a JSON file containing the trend information for the object: “clusterStoN_ ONTk _TIB_mpv”, which contains the signal to noise ratio of clusters associated to reconstructed particle tracks for the inner barrel part of the strip tracker.

The harvesting script accepts a set of options:

- Data type (proton - proton collision data or data taken in off-beam periods triggering on cosmic muons),
- Data reconstruction type (raw data or reconstructed data),
- Data quality filter (data that fulfill specific quality criteria) and
- Configuration files defining the information need to be extracted from the data (calling predefined metrics methods) and the names of the plots to be created.

More than one configuration files can be passed as argument to the script. The available metrics are defined within two python files: the ”basic” metrics (i.e. mean value, bin content, integral) and the more complex metrics with fitting capabilities (i.e. Gaussian, Landau-distribution). The most common metrics methods and their description are presented in Table 1.

When the script is executed, the configuration files are parsed and from them a list of trend objects is created. Then, for each trend object a check is performed in CMS DQM in order to verify if all the requested runs are valid and a final list of runs is created. Once this is done the harvesting can start. For each run the corresponding ROOT file on the CMS DQM server is opened (via web protocol, without download) and a loop over the different trend objects is executed. For each trend object, the histogram associated to it, is loaded into the memory and the corresponding metric is executed. When all the runs have been processed, the script builds the output files and for each trend object the results are saved in a JSON file.

Table 1. The most frequently used methods for extracting information from the CMS DQM.

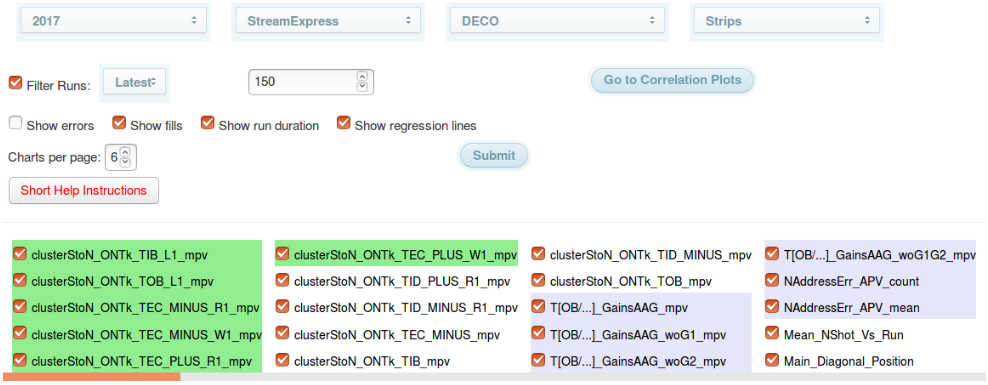
File Name	Method Name	Method Description

basic.py	basic.Mean()
	basic.Count()
	basic.BinCount(i_{th})
	basic.BinsCount(i_{th})
	basic.MaxBin()
	basic.MeanY()
	basic.FractionInBinArray([x],[y])
		Mean value of 1D histogram
		Total number of entries of 1D histogram
		Single bin (i_{th}) content
		Sum of all bins starting from the i_{th}
		Bin with the highest value
		Mean value in the vertical axis
		Ratio between x_{th} and y_{th} bin

fits.py	fits.FlatLine(...)
	fits.Gaussian(...)
	fits.Landau(...)
		Flat Line fit
		Gaussian fit
		Landau fit

3 Web Interface

The JSON files that describe the trend objects names and also those that contain the actual trend object data are fed to the web interface tool for visualization. This tool provides controls for dataset selection, filtering of data and navigation. It uses drop down menus where the user can select: the year of the data taking period, e.g. 2017, the Data type, the CMS tracker data taking mode related to the Silicon Strip APV25 readout chip [11] working mode and the Tracker Subsystem e.g. Pixel, Silicon Strip. A set of check boxes help the user to apply various filters concerning the number of runs that will be displayed, e.g. all runs from the beginning of the run period or the latest runs or specific run sets. Figure 3 shows a possible user selection.


2017
StreamExpress
DECO
Strips

☑ Filter Runs: Latest 150

☐ Show errors ☑ Show fills ☑ Show run duration ☑ Show regression lines

Charts per page: 6

☑ clusterStoN_ONTk_TIB_L1_mpv
☑ clusterStoN_ONTk_TOB_L1_mpv
☑ clusterStoN_ONTk_TEC_MINUS_R1_mpv
☑ clusterStoN_ONTk_TEC_MINUS_W1_mpv
☑ clusterStoN_ONTk_TEC_PLUS_R1_mpv
☑ clusterStoN_ONTk_TEC_PLUS_W1_mpv
☑ clusterStoN_ONTk_TID_PLUS_R1_mpv
☑ clusterStoN_ONTk_TID_MINUS_R1_mpv
☑ clusterStoN_ONTk_TEC_MINUS_mpv
☑ clusterStoN_ONTk_TIB_mpv
☑ clusterStoN_ONTk_TID_MINUS_mpv
☑ clusterStoN_ONTk_TOB_mpv
☑ T[OB/...]_GainsAAG_mpv
☑ T[OB/...]_GainsAAG_woG1_mpv
☑ T[OB/...]_GainsAAG_woG2_mpv
☑ T[OB/...]_GainsAAG_woG1G2_mpv
☑ NAddressErr_APV_count
☑ NAddressErr_APV_mean
☑ Mean_NShot_Vs_Run
☑ Main_Diagonal_Position

Fig. 3. Main page of the HDQM Web Interface with a possible user selection related to the CMS Silicon Strip Tracker Detector.

The graphical representation of the data is done using the Highcharts [12] API. Figure 4 shows an example of a plot displaying the signal to noise ratio of the Tracker Inner Barrel layers. The white and grey zones of the runs that belong to the same LHC fill number and the run duration in the right vertical axis are also displayed. For each point a Highcharts pop up box with all the relevant information is also activated together with the relevant plot names. Figure 5 shows a 2D correlation plot displaying the total number of clusters in the

Silicon Strip Tracker versus the total number of clusters in the Pixel detector. The colors indicate the run range.

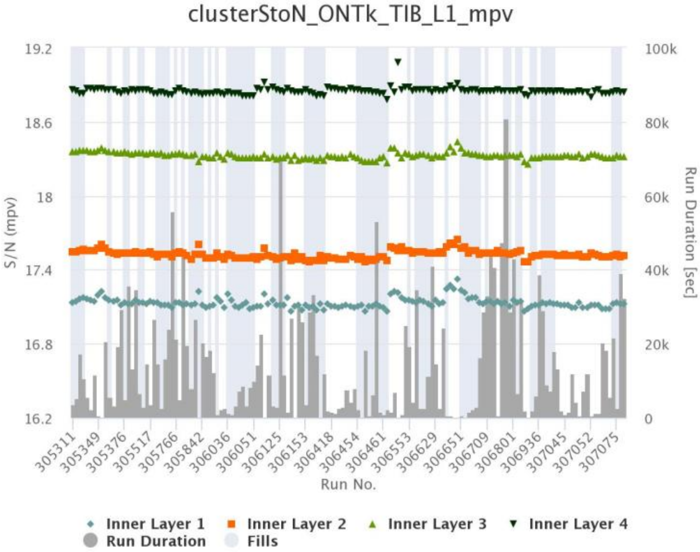


Fig. 4. Example of a plot displaying the signal to noise of the Tracker Inner Barrel layers. Runs that belong to the same LHC fill are shown as grey zones as well, while the run duration in seconds is displayed in the right vertical axis.

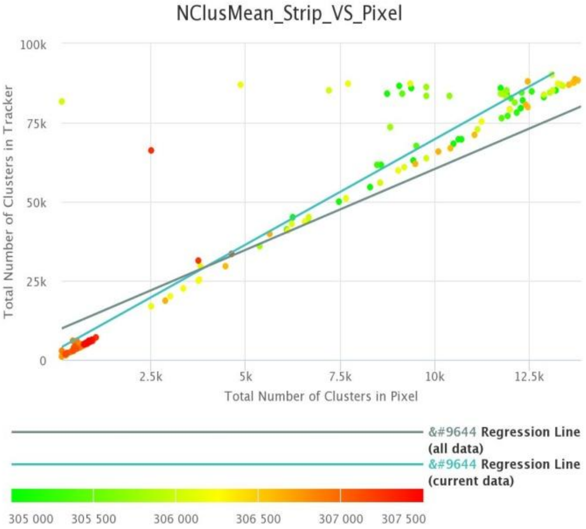


Fig. 5. Example of a 2D correlation plot displaying the total number of clusters in the Silicon Strip Tracker versus the total number of clusters in the Pixel detector. The colors indicate the run range.

The Web interface tool is based on a static HTML document structure while all the dynamic content is created with JavaScript. Due to the amount of data to be displayed on a single page and to avoid overloading problems, data loading and rendering is done asynchronously. Figure 6 shows the flow chart of the Javascript based HDQM Web Interface tool that is explaining the navigation through the various classes.

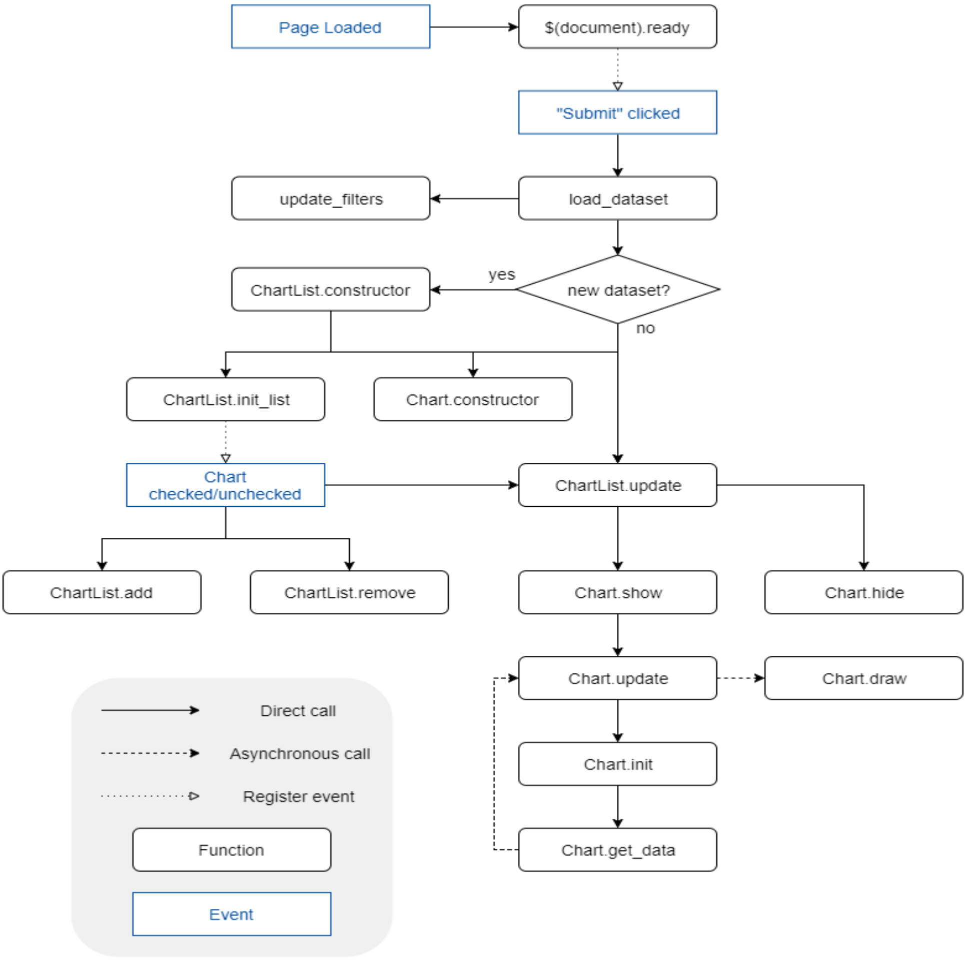


Fig. 6. Flow Chart graph of the HDQM Web Interface explaining the navigation through the various classes.

4 Conclusions

The HDQM is a framework developed by the Tracker group of the CMS collaboration to monitor the time evolution of sensitive quantities of the Tracker systems. It runs in servers provided by the central CMS DQM group and the development of new features is under consideration. Such new features may include the use of a database scheme instead of a collection of JSON files, the addition of extra information by adopting a bin width proportional to the run duration and the possibility to fit interactively the trends plots. This framework is flexible enough to permit the extension of its use to the other CMS sub-detectors since the core software will remain unchanged while new observables under consideration can be to the web interface via JSON files.

5 References

1. CMS Collaboration, The CMS experiment at CERN LHC, JINST **3 S08004** (2008), doi:10.1088/1748-0221/3/08/S08004
2. CMS Collaboration, “CMS Tracker Technical Design Report”, CERN/LHCC **98-6**
3. CMS Collaboration, “The CMS tracker : addendum to the Technical Design Report”, CERN-LHCC-2000-016, CMS-TDR-5-add-1
4. CMS Collaboration, “CMS Technical Design Report for the Pixel Detector Upgrade”, CERN-LHCC-2012-016, CMS-TDR-011,FERMILAB-DESIGN-2012-02
5. ROOT project, ”Root” version 6.14.0 (2018), https://github.com/root-project/root/releases/tag/v6-14-00
6. Python project, “Python” version 3.6.2, (2017), https://www.python.org/downloads/releae/python-362/
7. H. Andrews, A. Wright, “JSON schema: A media type for describing JSON documents”, http://json-schema.org/latest/json-schema-core.html
8. L. Hunt, “Html5 reference: The syntax, vocabulary and apis of html5”, https://dev.w3.org /html5/html-AUTHOR/
9. J. Duckett, “Javascript and jquery: Interactive front-end web development”. http://javascriptbook.com/
10. Federico De Guio for the CMS Collaboration, “The CMS data quality monitoring software: experience and future prospects” J. Phys.: Conf. Ser. **513** 032024 (2014), doi:10.1088/1742-6596/513/3/032024
11. L.L.Jones *et al.*, “The APV25 Deep Submicron Chip for CMS Detectors”, CERN/LHCC 99-09, p.162-166 (1999), doi :10.5170/CERN-1999-009.162
12. Highsoft AS, “Highcharts, highstock and highmaps documentation”, https://www.highcharts.com/