Web System for Data Quality Assessment of Tile Calorimeter During the ATLAS Operation

C. Maidantchik, F. Ferreira, F. Grael, A. Sivolella & L. Balabram
on Behalf of ATLAS TILE Calorimeter Community

1 COPPE, Federal University of Rio de Janeiro (UFRJ), Brazil
2 Escola Politécnica, Federal University of Rio de Janeiro (UFRJ), Brazil
E-mail: Tile.Commissioning@cern.ch

Abstract. TileCal, the barrel hadronic calorimeter of the ATLAS experiment, gathers almost about 10,000 electronic channels. The supervision of the detector behavior is very important in order to ensure proper operation. Collaborators perform analysis over reconstructed data of calibration runs for giving detailed considerations about the equipment status.

During the commissioning period, our group has developed seven web systems to support the data quality (DQ) assessment task. Each system covers a part of the process by providing information on the latest runs, displaying the DQ status from the monitoring framework, giving details about power supplies operation, presenting the generated plots and storing the validation outcomes, assisting to write logbook entries, creating and submitting the bad channels list to the conditions database and publishing the equipment performance history.

The ATLAS operation increases amount of data that are retrieved, processed and stored by the web systems. In order to accomplish the new requirements, an optimized data model was designed to reduce the number of needed queries. The web systems were reassembled in a unique system in order to provide an integrated view of the validating process. The server load was minimized by using asynchronous requests from the browser.

1. Introduction
ATLAS [1] is one of the particle physics experiments at the Large Hadron Collider at CERN, designed to measure proton-proton and heavy ions collisions. ATLAS is composed by several detectors with particular features.

Tile Calorimeter [2] (TileCal) is the central section of the hadronic calorimeter. It is a sampling calorimeter using iron as absorber and scintillating tiles as active medium, that is readout by wavelength shifting fibers and photomultipliers [3]. TileCal is composed by four sections divided in two Long Barrels (central section) and two Extended Barrels. Each barrel is divided azimuthally into 64 wedges, so-called modules. A module contains a number of cells and each one has two PMTs. Hence, there are roughly 10,000 readout channels in the calorimeter.

The number of electronic channels and the complexity of the experiment demand a continuous monitoring of the acquired data quality. Collaborators perform analysis over reconstructed data of calibration runs. In order to assert detector’s status [4], hundreds of plots and histograms are analyzed,

1 In 2010, the proton-proton collisions reached 7 TeV center of mass energy.
the output from Data Quality Monitoring (DQM) Framework [2] is considered and complete reports are composed. Findings (i.e. noisy channels, tripping power supplies, etc.) are presented at the weekly meetings and discussed among: TileCal community. The data quality responsible is on charge of updating the list of problematic channels that should not be considered for the future physics analyses.

During the commissioning period, seven web systems were developed for guiding the collaborators through the laborious data quality assessment process. The event rate during the ATLAS operation increased the amount of produced data. Over ninety million plots are stored in the databases, slowing the systems performance. Thus, an optimized data model was designed to reduce the number of needed queries. The web systems were reassembled in a unique system to provide an integrated view of the validating process.

Hence, a unique web system, so-called Dashboard, was created. It presents a set of views, each one representing a different aspect of the duty. For example, one view displays information about the latest runs and whether they were reconstructed. Another one shows the available plots and histograms and allows the user to register comments. The system will be presented by detailing its functionalities in the Section 2. The used technologies and the software architecture are discussed in the Section 3. Its data model is presented in the Section 4.

2. Data validation using the Dashboard
The Dashboard guides the data validator during the offline shift through the different steps of the task. First, the shifter checks whether there is a new run and whether it has been reconstructed. The system presents the latest three hundred runs in a tabular format, as can be seen in Figure 1. The table presents details about the run, such as the run type, the run date and the number of events. Under the Plots column, a hyperlink is made available whenever the respective run is reconstructed. That’s the way validators know when the plots and histograms are done. Also, through this link they can access the plots and perform their analysis.

![Figure 1. List of the latest runs. The run details, like date and number of events are shown.](image)

The system shows an overview of the runs. This overview reports the proportion of good and bad modules based on the status retrieved from the software model. It also displays the name of the responsible for the analyses, i.e., the validators. It is important to notice that a run can have more than one validator.
There is also the possibility of following the detector operation over time, through another perspective. The user can access a timeline through the main page. Through a search interface, it is possible to select the modules, the run type and the time period of interest. The system then displays the performance of the selected modules for the runs taken during the specified period of time, in chronological order. The statuses given by the validators are displayed according the following color code: OK is green, Bad is red, Some Problems is yellow, Not to Be Analyzed is black and the Not Analyzed is white. Runs with detailed comments are highlighted. The Figure 2 shows the Timeline view.

![Timeline View](image)

**Figure 2.** Performance of the modules over the runs.

Back on the overview display, the view for analyzing runs can be accessed through the icon on the Plots column. This view presents all modules for a given run, as it can be seen in the Figure 3.

For a given run, a table is then displayed with the following columns: Run Number, Module, Run Date, Run Type, Plots, Detailed Comments, Status Comment, DQM Status, Shifter Name. The first five columns are technical details about the run and are automatically filled into the database, while the others present data from the analysis. The user can toggle the visibility of any column. The columns Status Comment and DQM Status display icons with the same color code presented in the Timeline view. The icons under the Status Comment column are bigger than the DQM Status, as they are considered more relevant for the collaboration. By clicking on the “+” button, it is also possible to check the status and DQM statuses of the last five previous runs of the same type, e.g. Pedestal, for the corresponding module.

Moreover, it is possible to filter rows by the module’s name by writing in the search field located on the top-left part of the page. For example, if the user writes “LB”, just the rows corresponding to the long barrel modules. If an “A” is added to the query, just modules in the A side will appear. The number of rows in the same page can be configured as well. Sometimes, different barrels are analyzed by different validators and this feature allows them to focus on the modules of interest.

Validators can give the same status for a whole run, for convenience, i.e., before starting an analysis, is common to attribute the green status for all modules since this will be, or at least it is expected to be, the most common diagnosis. The DQM tool logic is pessimistic, that is, it will flag a module as problematic even if it is a minor problem that would not disturb the data acquisition. Therefore, most users prefer to hide the modules marked as good by the DQM tool, so they can focus only on those that could have problems.

Plots and histograms are available at the select boxes on the “Plots” column. By selecting a plot, a new view is shown. This new view shows the selected plot in its normal size and the other available plots...
Figure 3. Information and access to plots for a reconstructed run. View for an anonymous user.

as a thumbnails. This way the user can select another plot without leaving this screen. It is also possible to open all plots at a once, as is preferred by some users. The Figure 4 shows this view in details.

Figure 4. Available plots.

After logging into the system, the validator is capable of adding/editing comments. “Detailed Comments” can be written or modified by clicking on the corresponding cell, which causes a text-area field to appear. For selecting or changing the status comment value is just needed one click over the small icons.

An toolbar is also made available. The top toolbar presents four functionalities, each one represented by a different button. The first one summarizes all the shifter’s notes and post them to another view of the process, shown in the Figure 6.
At this point, all the information about the analyzed run is consolidated for creating an entry in the collaboration’s official electronic log (e-log). All the well behaved modules are suppressed in this step, because only problems are to be reported. Also, the same overview displayed in the run list in the first view is submitted. The validator has to write down the final comments about the whole run. By submitting the information, the data quality validation is done for a run.

It also is possible to access the web system that displays data from the Tile Detector Control System (DCS) [5]. This system is responsible for providing information for the monitoring of the detector’s power supplies behavior. For this specific case, the mean and standard deviation values for voltages and currents during the time period of the given run. The interface present a graphic representation of the barrels. Each wedge is divided in two so that the mean and standard deviation
analysis can be displayed. Colors are given to each part of the wedge, according to the result of the comparison of these statistics to the nominal values. Figure 7 shows the described functionality. The integration between the Tile DCS and the main system important because many problems during the data acquisition can be correlated to power supplies’ misbehavior.

![Figure 7. Detector Control Web System. High Voltage status during the Run 169555.](image)

3. Technologies
The Dashboard is based on the Model-View-Control (MVC) software architecture and uses a JavaScript framework for rendering the user interface and performing requests to the server asynchronously. It uses a Python based back-end for serving these requests. The MVC pattern isolates the application logic from the user interface, allowing them to be developed, tested and maintained separately. The model represents the system’s data structure and how the application will access it. It manages the behavior from the requests of the end user. The view renders the models making it suitable for the user’s interaction. The user’s actions are processed and requests are sent to the model whenever it is necessary. The controller is the set of scripts that activates the model according to the requests from the client. Figure 8 describes how the different layers communicate among each other.

![Figure 8. Model-View-Controller Architecture](image)

All views are based on dynamic HTML (DHTML), this is, combining HTML, JavaScript and CSS elements [6].
Requests to the server and their response are made using messages in XML format, retrieved asynchronously. This allows for a more elaborate client-side processing and, due to this, an action from the user does not necessarily reflects on a request to the server.

The AJAX method was used for handling this communication [7] and the Jquery library was the chosen one for developing the AJAX engine [8]. By doing this, the server load was decreased and users get the results faster.

The back-end system, which comprehends the controller and models, is implemented mainly in Python. The model for all the views discussed are implemented in a single file, which is used by the controller for each action. Although the layout for the views may seem dissimilar, they all have been made to use this common back-end. The dissimilarities on the layout have been kept in order to reflect the different steps of the validation process. The system store its data into ORACLE database server, managed by the CERN’s IT department.

4. Data Model
The proper functioning of the system depends on how the data is modeled. Therefore, after the beginning of the ATLAS operation, the database was carefully redesigned for minimizing the database load.

The web system uses a relational database for data persistence. The model presents two major entities – run and module – which are associated by a many-to-many relationship. One run has the following attributes: date, type, id number and the number of events. For guaranteeing the normalization, the entity run type was created since they are limited. The module is represented by the name of the barrel where it is located and by its drawer number, e.g., LBA25. The Figure 9 shows the relations.

![Data model after the beginning of ATLAS operation.](image)

After the run reconstruction, several plots and histograms are created and stored into public directories of a local server at CERN. For each module, a summary flag is set according to outcomes of the DQM Framework tests.

The data quality validator assigns status for each module in a run according to its performance after analyzing the plots and understanding the DQM flag. There are five possible status: “OK”, “Bad”, “Some Problems”, “Not Analyzed” and “Not to be Analyzed”. The validator is also able to provide detailed comments if needed. The validator names and their CERN ids are stored.

Thus, the new data schema represents the entities from the data validation task. The software’s model pattern uses this schema and takes care how the attribute values are filled and how the system responds for different requests.

5. Conclusion
During the ATLAS commissioning period several prototypes were purposed for supporting the offline analysis job. These prototypes became a set of web systems extensively used by the whole collaboration. The amount of data at that point allowed the use of an unofficial database without an accurate model.
It was necessary to redesign the system for dealing with the large amount of data produced during the ATLAS operation. The commissioning experience was very important for purpose the creation of an unique system for the whole data analysis process, capable of guiding the validator through the laborious job.

This way, a web system using MVC software architecture was developed. This system provides information on the latest runs, displays status from the automatic monitoring framework, presents the generated plots and stores the validation outcomes, assists to write logbook entries, creating and submitting the bad channels list to the conditions database and publishes the equipment performance history.

The system is accessible for the entire collaboration and runs under the Scientific Linux 5 environment in the AFS CERN’s node.

6. Bibliography
[1] ATLAS Collaboration. ATLAS: Technical Proposal for a General-Purpose pp Experiment at the Large Hadron Collider at CERN. Technical report, CERN, 1994. CERN/LHCC 94–43.
[2] J M Abdallah, U Blumenschein, T Davidek, A Dotti, L Fiorini, C Maidantchik, C S Sanchez, and A Solodkov. Atlas tile calorimeter data quality assessment with commissioning data. Journal of Physics: Conference Series, 119(3):032001, 2008.
[3] Anderson, K., Berglund, S., Blanchot, G., et all. Front-end electronics for the atlas tile calorimeter. Technical report, CERN, 1998.
[4] R Calkins. The atlas tile calorimeter performance at the lhc. Technical Report ATL-TILECAL-PROC-2011-004, CERN, Geneva, May 2011.
[5] C Maidantchik, F Ferreira, F Grael, and Atlas Tile Calorimeter Community. Tiledcs web system. Journal of Physics: Conference Series, 219(2):022029, 2010.
[6] Jason Cranford Teague. DHTML and CSS for the World Wide Web. Peachpit Pres, 2001.
[7] James Garret. Ajax: A new approach to web applications. article Adaptive Path http://adaptivepath.com/ideas/essays/archives/000385.php, 2005.
[8] Bear Bibeault and Yehuda Katz. Jquery in action. Manning Publications Co., Greenwich, CT, USA, 2008.