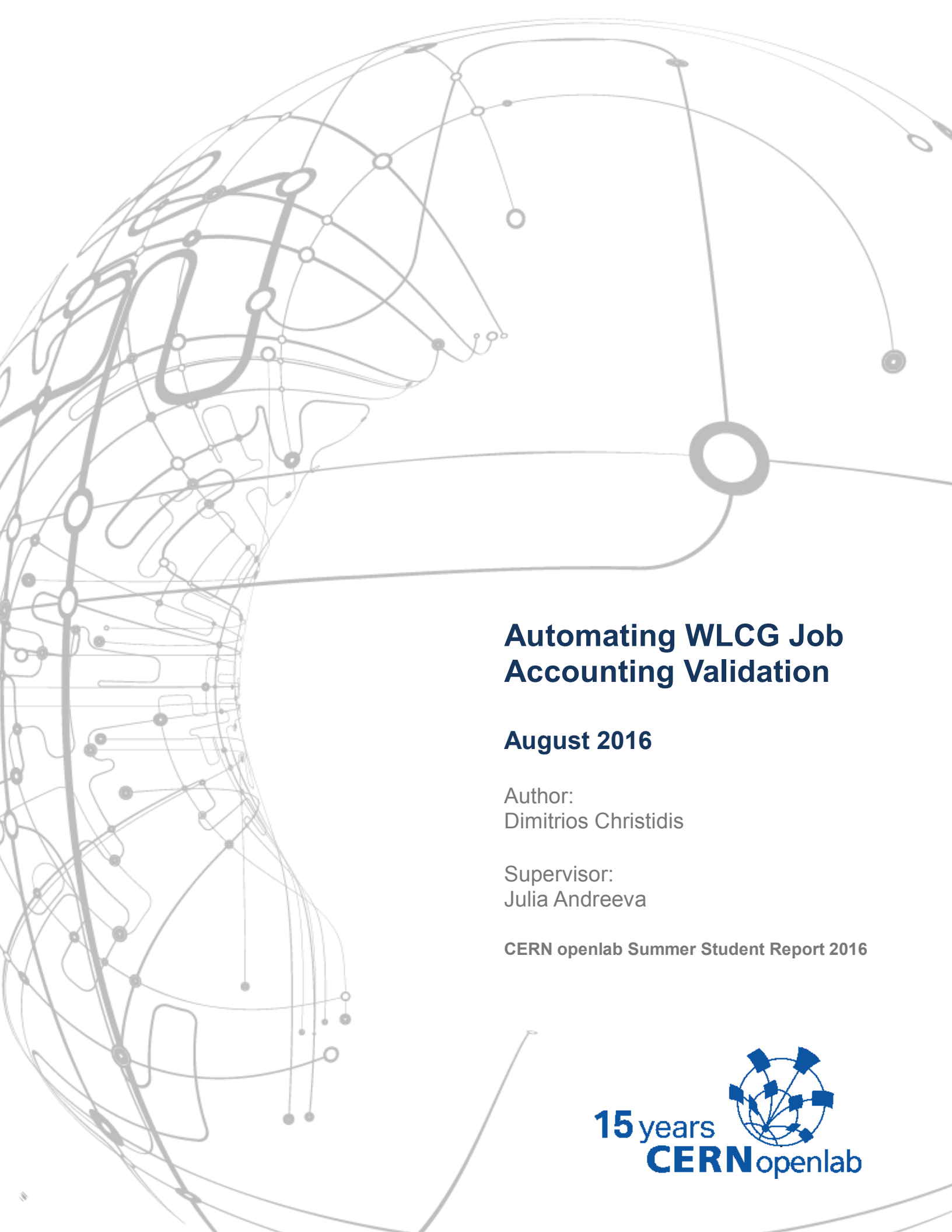

Automating WLCG Job Accounting Validation

August 2016

Author:
Dimitrios Christidis

Supervisor:
Julia Andreeva

CERN openlab Summer Student Report 2016

15 years CERN openlab

Project Specification

The WLCG infrastructure provides resources to store, distribute and analyse the data generated by the Large Hadron Collider (LHC). The LHC data is analysed at multiple computing facilities at the geographically dispersed LHC Grid sites. Accounting data are produced for each job executed on the Grid and collected to a central accounting repository through APEL. One of the goals of the WLCG accounting project is to prove the trustworthiness of the data presented on the new accounting portal. This entails performing automated, frequent comparisons against the data from the experiment-specific accounting systems and visualising the results. The student will work on an existing implementation with the goal of completing the necessary features, providing improvements and creating an infrastructure for its automated execution.

Abstract

The WLCG Accounting Task Force is overseeing the creation of a new accounting portal. One of the areas of work is to check the trustworthiness of the provided data. This is accomplished by making comparisons against the data from the accounting systems used by the LHC experiments. To that end, a script was created to perform these checks and publish the results online. Additional emphasis was given on making sure it is fully automated and on improving the visualisation of the results. The initial outcome has been positive and some accounting issues were discovered and fixed.

Table of Contents

1 Introduction

The Worldwide LHC Computing Grid (WLCG) [1] is the biggest distributed computing production infrastructure in the world. It is the result of a global effort, built for the purpose of analysing the vast amount of data produced by the Large Hadron Collider (LHC) [2] at CERN. It is comprised of more than 170 sites, spread through 42 countries. Each day, more than 2,000,000 jobs are executed on its systems.

As a result, it is of utmost importance to be able to account computing resources used for data processing and data storage. The current project is focussed on the CPU accounting.

2 WLCG Accounting

Before becoming a part of the WLCG, aspiring sites must sign a Memorandum of Understanding. Essentially, it is a form of contract where the site pledges to provide a certain amount of computing resources for the needs of the LHC experiments. Of course, there is a necessity for a mechanism to verify that those resources were provided and to determine how they were utilised. The WLCG Accounting Task Force [3] is overseeing the creation of a new accounting portal that presents accounting data and generates accounting reports.

The portal uses measurements produced by the batch systems of the sites, which are collected and processed by APEL [4], the EGI [5] accounting system. On the other hand, the experiments have their own internal accounting systems. In order to prove the trustworthiness of the data provided by the accounting portal, a comparison of the APEL data is performed against the data coming from the experiment-specific accounting systems. The results of the comparison are to be published on a development instance of the Site Status Board (SSB) [6].

Measuring processor utilisation is not a trivial matter. Firstly, all stakeholders must agree on a specific measurement. Then, it has to be proven that the measurements are accurate. Currently, the focus is drawn to the two metrics described below.

Wall Clock Time as reported by the batch system. Raw wall clock time represents the elapsed time between the start and the end of a job. It should be multiplied by the number of processor cores that were used. In an environment like the WLCG, jobs are usually long in duration and context switches are not frequent. However, this metric is not sufficient to estimate how much actual work has been performed since it does not take into account the benchmarked HEP-SPEC06 [7] power of a given CPU resource. Moreover, some of the batch systems do not report raw wall clock time, but rather report wall clock time scaled by a batch system. Scaling allows jobs to run smoothly in a heterogeneous resource because the batch system automatically extends the CPU time and wall clock time on slower nodes proportionally to their power. Therefore, the first metric which is taken into account is Wall clock time as reported by the batch system.

HEP-SPEC06 Wall Clock Work: The wall clock time as reported by the batch system, multiplied by benchmarked HEP-SPEC06 power of the reference node and multiplied by the number of processor cores. HEP-SPEC06 is a benchmark that evaluates the performance of a processor. The resulting value should be representative of the work that was completed. Still, some questions remain on whether the measurement is done accurately on all levels.

Even if the measurements were entirely accurate, it would have still been impossible to have a perfect match between the two sources. This is due to the fact that experiments and sites measure different things. Experiments have moved to a mechanism known as pilot jobs [8]. Submitted jobs are no longer directly pushed to the sites' batch systems. They are placed in an experiment-specific queue and pilot jobs are pushed to the sites instead. When a pilot job is executed, it fetches a payload job from the central queue. As depicted in Figure 1, sites measure the entire pilot job execution, while experiments measure only its payload. In theory, the overhead of the pilot should be small compared to its actual payload.

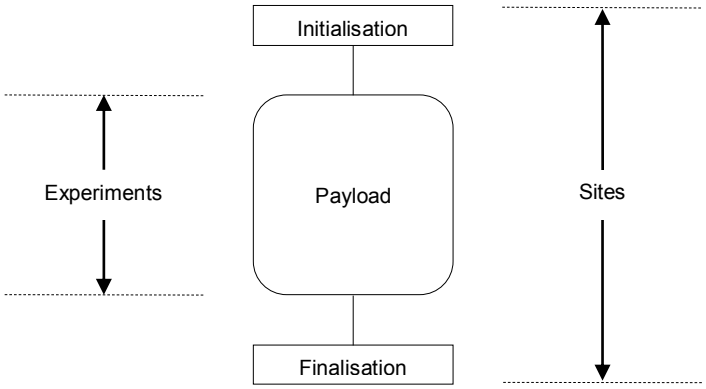

Figure 1: Simplistic model of a pilot job.

3 Implementation Details

A collection of scripts was already available and it had proven very useful in providing some preliminary results. However, the scripts were not designed to be executed in an automated manner. Furthermore, there was a lot of code repetition, which created maintenance burden. The initial task was to refactor the source code and port it from the C Shell to the POSIX specification.

3.1 Architecture

In the new implementation, everything is consolidated into a single file. Multiple other scripts written in AWK, Python and Sed are embedded into the main script and they are extracted to a temporary directory at run-time.

3.2 Data Flow

The operations performed over the data can be organised into seven well-defined stages. They are presented graphically in Figure 2. This approach creates clear segregation inside the source code. Changes made to a stage are completely isolated. Further descriptions of each stage are provided below.

Download: The raw accounting data are downloaded from the experiment-specific accounting systems and from the EGI Accounting Portal via available APIs. The file format is CSV for the former and JSON for the latter.

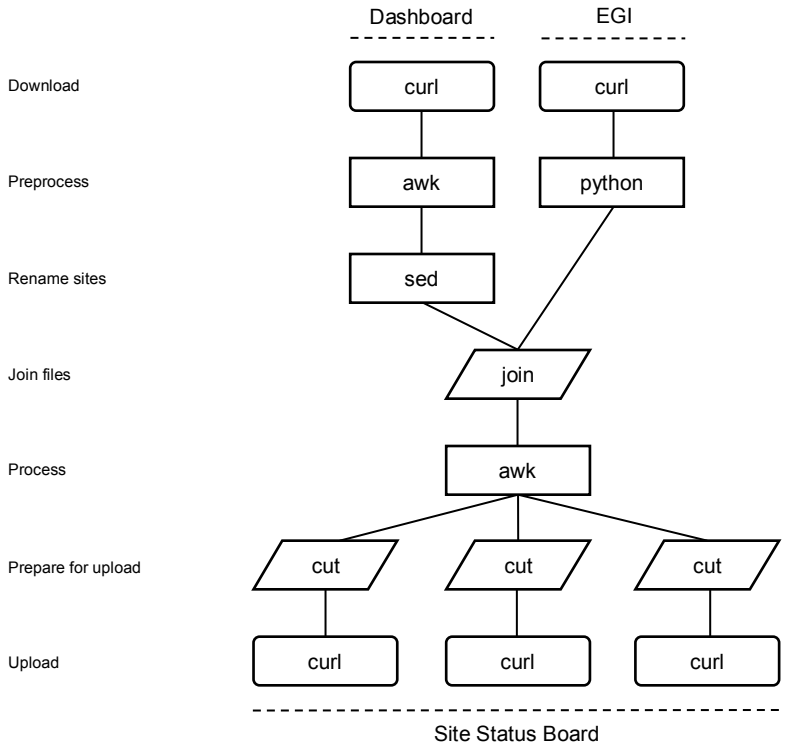

Figure 2: The flow of data through the stages of the script.

Preprocess: The files are processed so that they have the same two-column comma-separated structure. Depending on the experiment, some entries and/or values are discarded. In addition, the data is converted to the same time unit. Finally, sites that belong in the same logical entry are merged into one.

Rename: The entries from the experiment-specific systems are renamed to conform to the official WLCG naming specification. Internally, experiments have their own convention. The conversion is done using the topology information from the ATP VO feeds [9], which are XML files that describe, among other things, the match between the internal and the official names.

Join: The two files from the previous stage are merged. In database terms, it is an outer join operation. Thus, entries are maintained even if they appear in only one of the files.

Process: This is the core stage where actual computations are performed. The ratios are calculated and the colour is determined. Some additional values that are required by the SSB API are appended, such as the numerical ID of the metric and the location to open on click. The format is changed to tab-separated values.

Prepare: The output from the previous stage is split in order for each SSB metric to use its own file.

Upload: The results are published on SSB.

3.3 Colour Coding

The improvement of the data visualisation was a noteworthy objective. When viewing a graph, the user's attention should only be drawn to sites that show high discrepancy in their data. However, it is also important to be able to understand how the data evolved over time without having to look at the actual numbers. The original scripts used three colours to signify the divergence of a ratio from 100 %. Instead, this implementation raises that number to thirty-two.

The colour is calculated based on the ratio and some thresholds. In order to allow for some flexibility, the thresholds can be defined on a per-experiment basis. Thus, the results can be adapted based on what is considered acceptable by the experiment, which in turn depends on factors like the overhead of the pilot framework that is used.

Figure 3: The implemented colour gradient.

4 Deployment

The source code, along with documentation aimed at future maintainers, were made available online on the CERN GitLab service*. Git is a distributed version control system. It keeps track of the changes made to a project and it provides features that ease the development process, such as the ability to easily maintain multiple branches. In turn, GitLab is a web-based application to manage Git repositories. The choice of licence was made following the recommendation from the Open Source Licence Task Force [10]. Thus, the code developed for this project is Free Software.

The script does not require any input from the user during execution. But, in order to make the process fully automated, it has to be arranged to run periodically on some remote machine. This ensures the dependable and timely publication of the latest accounting data. CERN has a large OpenStack infrastructure. OpenStack is a cloud computing platform that allows users to create and manage virtual machines, while transparently handling the allocation of the actual hardware resources.

Setting up a virtual machine to execute the script can be done manually, but it is preferable to do the configuration in a structured and automated way. For this purpose, CERN has a Puppet infrastructure. Puppet is configuration management tool that can administer and orchestrate changes across thousands of servers.

Configuring the virtual machine requires writing a manifest file in Puppet's declarative language. The purpose of the manifest is to describe the desired state of the machine. A Puppet agent running on the machine will periodically communicate with the Puppet master server, receive the configuration and perform any steps required in order for the machine to reach its desired state. The manifest itself is maintained in a Git repository.

Below are the required steps to set up the virtual machine after its initial creation:

* The source code is available at https://gitlab.cern.ch/WLCG-Ops-Services/accounting_validation.

1. Create a local dedicated user.
2. Install the certificate and the key in the user's home directory.
3. Configure the script to be executed once a month using Cron.

Cron is a Linux software utility that allows users to schedule the execution of other software at specified time intervals. Cron was set to download the script from the Git repository and execute it on the fourth day of every month. The benefit of this approach is that it eliminates the need to redeploy with every revision. At each invocation of the script by Cron, the latest version of the code is used.

Running the script and publishing the accounting data should be done as early as possible. This grants quick detection of any issues with the sites. However, it should not be done before accurate data are available on the experiment-specific accounting systems and on the EGI Accounting Portal. The choice of the fourth of each month was made arbitrarily. Consequently, an evaluation is being made to see if there are alterations of the data past the chosen day, and if so, to move the day or to add a second invocation per month.

5 Results

The Site Status Board offers an intuitive user interface to view the accounting data. Figure 4 shows the main view, where the data are presented in a table. It is possible to sort the data based on any column or search for a specific site. However, this view shows only the latest values. The user needs to able to also see how the data evolved over time.

Figure 5 showcases the type of graphs that can be produced by SSB. Here it is possible to choose an arbitrary period of time. As a result of the colour coding, it is easy to locate sites with high discrepancy. Furthermore, due to the use of a gradient, the user can quickly comprehend if the status is getting better or worse. In this particular graph, one specific metric is visualised for all sites. Likewise, it is possible to visualise all metrics for one specific site.

Lastly, Figure 6 shows the display of the raw data when selecting an individual cell in one of the graphs. This feature provides a better understanding of the site status. SSB itself does not provide any mechanism to achieve this. Thus, the hidden text metric was added to address this limitation.

The initial results that were published during the development of the script provided valuable data. They showed good consistency overall, but drew attention to problematic cases. For example, two sites had very high discrepancy. The appropriate teams were notified and the issues that were causing the invalid data were found and addressed.

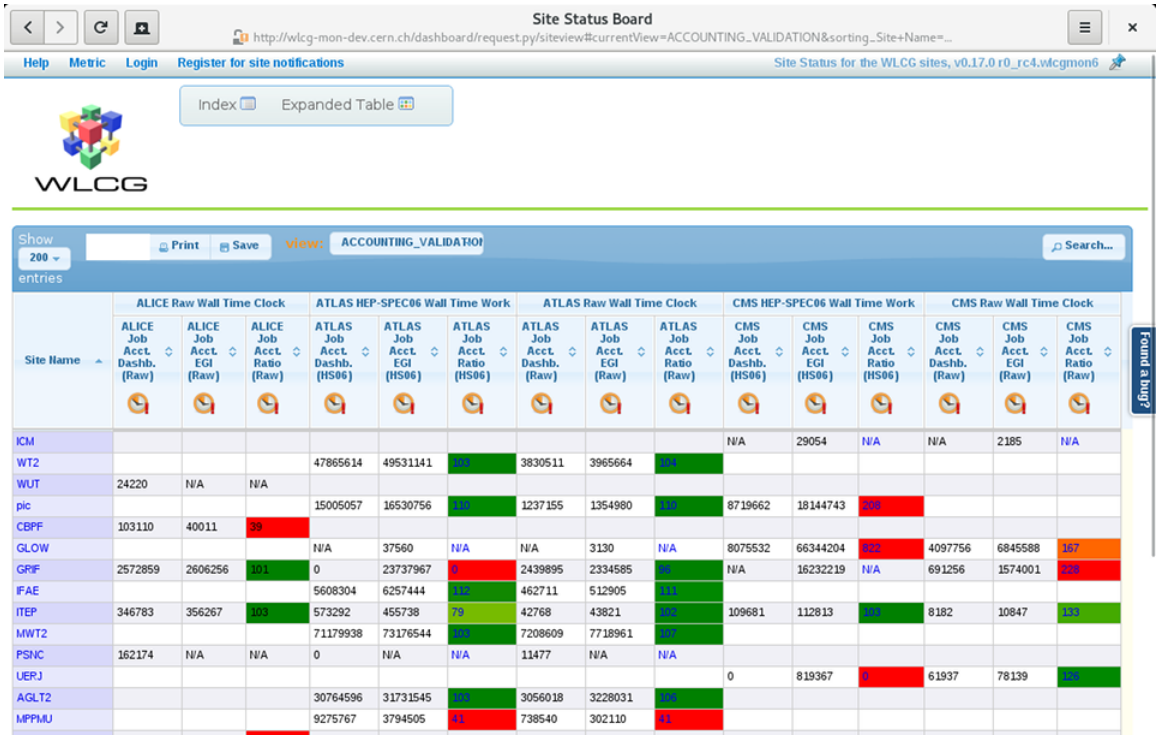

Figure 4: The main view of the Site Status Board.

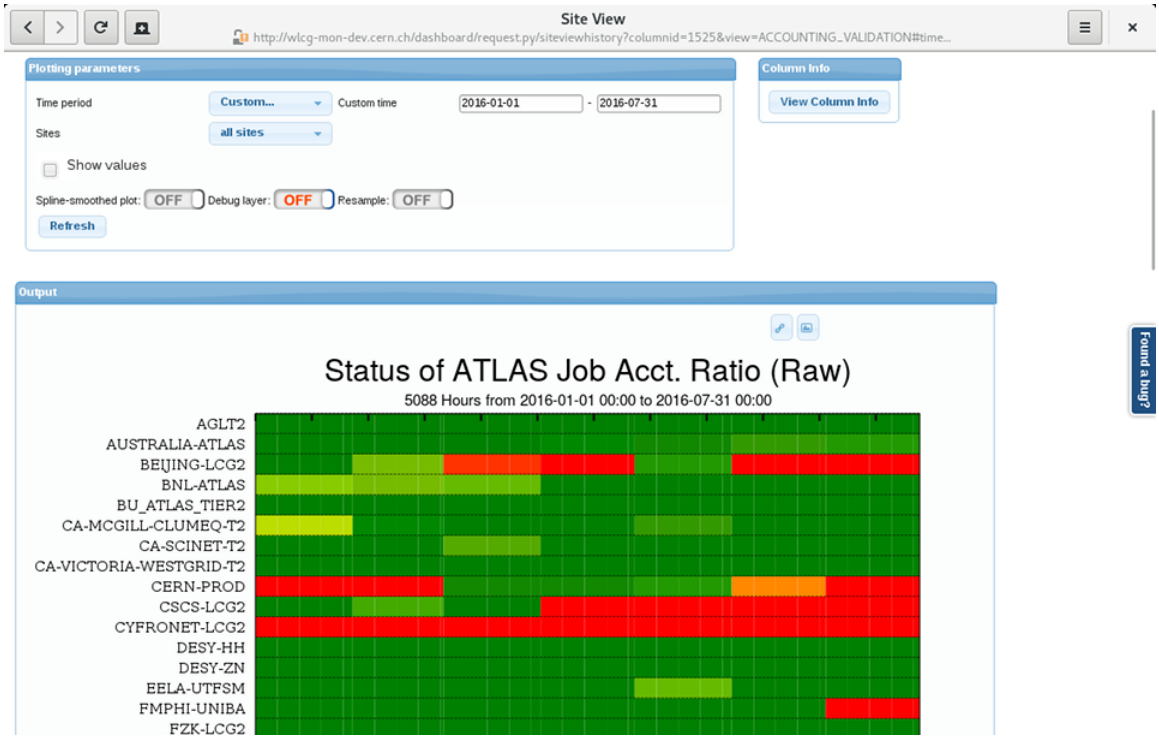

Figure 5: One of the graphs produced by the Site Status Board.

Figure 6: The raw data inside a JSON file.

6 Conclusion and Future Work

This project was successful in accomplishing its goals. It provided a tool that assists in locating and understanding disparities between the accounting data from the sites and the experiments. As these issues are addressed, the validity of the data on the EGI Accounting Portal is reinforced. This, in turn, constitutes one of the goals of the WLCG Accounting Task Force. The deployment removes the need for interaction from the user and the initial results have proven constructive.

There is some room for further development and improvement. The most notable omission is the lack of support for ALICE and LHCb. This was not possible during the time span of the project because these experiments did not provide an interface to acquire the accounting data on demand. For ALICE, a limited set of offline data dumps was used to manually publish some initial results.

Another area that can be refined is the handling of the topology information and the renaming of the sites. In the current implementation, the conversion rules are hardcoded in an embedded script. This is not optimal and it will create additional maintenance burden in the future. Instead, a much better approach would be to download the XML files, parse them and produce the conversion script at run time.

Finally, some of the code can be restructured. Reducing the verboseness, refactoring duplicate logic into functions and consolidating different stages will decrease the total lines of code and improve the script's maintainability.

References

[1] "Worldwide LHC Computing Grid." wlcg.web.cern.ch. Accessed 22 August 2016.

[2] "The Large Hadron Collider." *CERN*, home.cern/topics/large-hadron-collider. Accessed 22 August 2016.

[3] Andreeva, Julia. "WLCG Accounting Task Force." *CERN TWiki*, twiki.cern.ch/twiki/bin/view/LCG/AccountingTaskForce. Accessed 22 August 2016.

[4] "APEL."apel.github.io. Accessed 22 August 2016.

[5] "EGI." www.egi.eu. Accessed 22 August 2016.

[6] Andreeva, Julia et al. "Site Status Board: a flexible monitoring system developed in close collaboration with user communities." *In proceedings of "EGI Community Forum 2012 / EMI Second Technical Conference,"* PoS(EGICF12-EMITC2)111, pos.sissa.it/archive/conferences/162/111/EGICF12-EMITC2_111.pdf.

[7] "HEP-SPEC06 Benchmark." w3.hepix.org/benchmarks/doku.php. Accessed 22 August 2016.

[8] Casajus, Adrian, Ricardo Graciani, Stuart Paterson, Andrei Tsaregorodtsev and the LHCb DIRAC Team. "DIRAC pilot framework and the DIRAC Workload Management System." *Journal of Physics: Conference Series*, vol. 219, no. 6, pp. 062049, iopscience.iop.org/article/10.1088/1742-6596/219/6/062049

[9] Collados Polidura, David et al. "ATP – VO TOPOLOGY FEEDS." *CERN Twiki*, twiki.cern.ch/twiki/bin/view/Main/ATPVOFeeds. Accessed 22 August 2016.

[10] Fluckiger, François. "Final report from Task Force on Open Source licensing at CERN." 10 January 2012, ep-dep-sft.web.cern.ch/document/final-report/task-force-open-source-licensing-cern.