ATLAS BigPanDA monitoring

A Alekseev[1], A Klimentov[2], T Korchuganova[1], S Padolski[2*], T Wenaus[2]

on behalf of the ATLAS Collaboration

[1]Tomsk Polytechnic University, 30 Lenin av., Tomsk, 634050, Russia
[2]Brookhaven National Laboratory, P.O. Box 5000, Upton, NY 11973-5000, USA

*Presenter, primary author

E-mail: siarhei.padolski@cern.ch

Abstract. BigPanDA monitoring is a web application that provides various processing and representation of the Production and Distributed Analysis (PanDA) system objects states. Analysing hundreds of millions of computation entities, such as an event or a job, BigPanDA monitoring builds different scales and levels of abstraction reports in real time mode. Provided information allows users to drill down into the reason of a concrete event failure or observe the broad picture such as tracking the computation nucleus and satellites performance or the progress of a whole production campaign. PanDA system was originally developed for the ATLAS experiment. Currently, it manages execution of more than 2 million jobs distributed over 170 computing centers worldwide on daily basis. BigPanDA is its core component commissioned in the middle of 2014 and now is the primary source of information for ATLAS users about the state of their computations and the source of decision support information for shifters, operators and managers. In this work, we describe the evolution of the architecture, current status and plans for the development of the BigPanDA monitoring.

1. Introduction

The ATLAS experiment [1] uses the PanDA [2] (Production and Data Analysis) Workload Management System for managing workflows for all data and Monte Carlo processing on over 170 WLCG [3] data centers and opportunistically used resources such as commercial clouds, supercomputers, volunteer machines. As of the middle of 2017, it manages executing of 2M jobs daily and run 300k simultaneous jobs.

To perform monitoring and effectively debug described computation payload a dedicated Web application was developed. The BigPanDA monitoring produces a set of aggregated reports, dashboards and representations of entities manipulated by the PanDA Workload Management system. There are four functional roles distinguished among the BigPanDA monitoring users: physicists, production managers, shifters and developers. Despite there being some common requirements all groups focus on different views provided by the system. For example, production managers need to have high-level aggregates which report the progress of particular payloads while developers should

have an access to job logs to trace possible issues. All of this information is accessible in the BigPanDA monitoring in real-time mode along with two years' history of job execution.

While the development of the monitoring system is primarily driven by the ATLAS requirements, other experiments, for example, COMPASS [4], also utilize their instances of BigPanDA to monitor custom computations.

2. Architecture

Basically, two mutually exclusive requirements defined the BigPanDA architecture: the system should work as a Web application analyzing data in a real-time mode and on the other hand it is expected to perform some accounting with aggregates of historical data. To find an appropriate balance between them the following approach was applied:

- Hotspots in Python source code were identified and these code fragments were reimplemented in PL/SQL in order to improve the processing-over-data performance.
- Web pages containing few independent data sources were separated into asynchronous fragments loaded into a web page by AJAX [5] or AngularJS [6] libraries.
- The system contains many SQL queries which involve large amounts of data in the processing algorithm and therefore they could not be executed in real-time mode. To cover such cases, we developed a proactive caching approach which is described later.

The general architecture diagram is presented in figure 1.

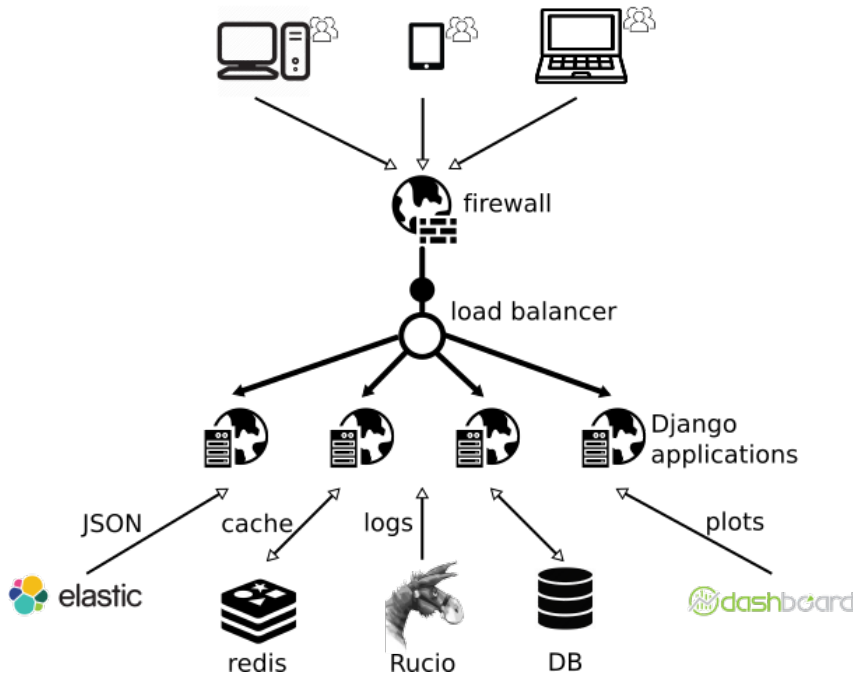

Figure 1. BigPanDA system architecture diagram

The Load Balancer (LB) distributes incoming traffic across the Apache Server instances which handle user requests. Currently, we are using the Nginx HTTP server [7] for this purpose.

Continuous development following the growing community needs requires the efficient utilization of frameworks. We are using Django [8] as a basic framework for BigPanDA monitoring because of its versatility, flexibility, and number of ready to use components. The dynamic data load provided by AJAX and AngularJS and Web page design is based on ZURB Foundation [9] components.

BigPanDA collects data from following sources as shown in Figure 1:

- ElasticSearch [10] instance providing access to parsed and indexed log records of PanDA server [11], JEDI [12] and Pilot [13].
- Rucio [14] provides access to log files for individual jobs from the distributed file storage to BigPanDA web servers which make the files accessible via ordinary URLs.
- CERN dashboard [15] service provides summary plots incorporated into the main page of the BigPanDA monitoring.
- Oracle DB [16] used by PanDA to keep all transactions is the main sources of data for BigPanDA monitor. Thanks to the Django DB Middleware [17] BigPanDA also supports MySQL [18] backend for installations beyond ATLAS.

As it was mentioned above a cache system is actively utilized in the monitoring. It has two principal purposes:

- To reduce the extra load on the Django part of the BigPanDA application and to the database. When a user requests particular information the response from the monitoring system is kept in the Redis cache [19] for 20 minutes and can be reused for the same requests in the future.
- Another advantage of data caching is the proactive preparation of aggregated data in order to reduce the waiting time and rise up the system's responsiveness. Some views require to process large volumes of data and these requests cannot be completed within a few seconds. For example, searching for jobs using a wildcard filter pattern may require processing almost 900 million database rows. To make such aggregates always available for display, a special crawler passes through a dynamically generated list of views with different filter conditions and then stores the outputs in the Redis cache.

As of the middle of 2017 BigPanDA monitoring servers receive 20k requests daily, with 35% of them conimg from automatic scripts or other third-party systems developed by users. Such custom applications can interact with the monitoring system using the JSON data format.

3. User-oriented content

BigPanDA monitoring generates around 70 different views providing information for the specific user roles described above. The distribution of requests for the most demanded views is shown in Figure 2.

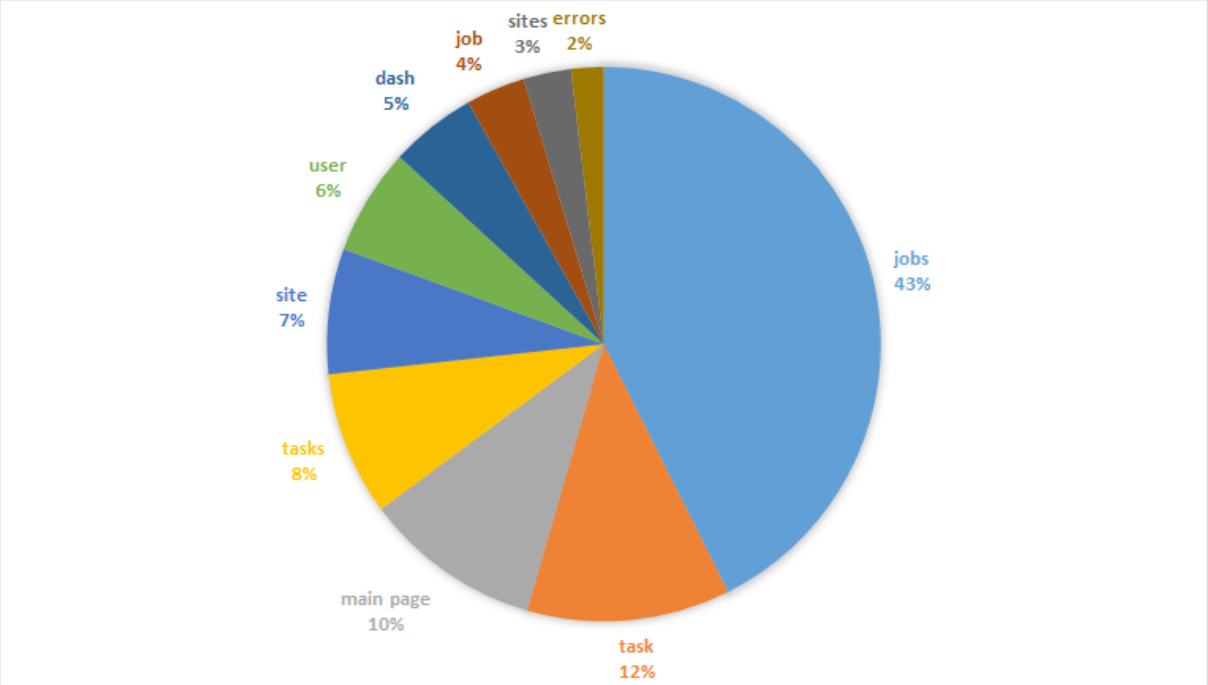

Figure 2. Distribution of the most frequently accessed views

The most important objects managed by the PanDA workload system are presented in the BigPanDA monitoring in two forms – as an individual (e.g. job, task, site) and as a part of a summary (e.g. jobs, tasks, sites). The second representation provides aggregates for key indicators in a compact form such as computation accounting, payload distribution over sites, or its internal structure etc. We will describe some of the most frequently accessed views below.

3.1. Jobs view

The jobs view provides three summaries on the top level: Job attribute summary, Overall error summary and Job list. In the first view the 26 most informative jobs attributes are aggregated by values and are available for the following drilling-down. Overall error summary accounts the errors codes of selected jobs and also provides some textual description of them. The Job list is the table containing a subset of selected jobs. There are several different filters available for this view: jobs modification time, assigned computing site, the name of the user who submitted the job, simulation software version, etc.

3.2. Tasks view

The tasks view has a similar structure as the one described above with the only difference being that the aggregated objects are the tasks.

3.3. User view

This view is the starting point used by most of ATLAS physicists to monitor execution of custom payloads. It contains tasks and jobs summaries filtered by a particular user, as well as the totals of the CPU time consumed. A recommender system is also integrated into this view. It analyses prior visits of BigPanDA monitoring and suggests links to views and objects that can probably be the focus of the user's attention.

3.4. Failures analysis

BigPanDA monitoring has advanced capabilities for error analysis. There are four basic inputs processed by the system: individual job logs, DB records, PanDA server logs and other infrastructure software logs. The last one produces nearly 400 GB of records daily. In order to manage and aggregate this data flow in an efficient way a special setup based on ELK stack [20] was developed [21]. BigPanDA Monitoring also has a special querying interface that allows one to compose custom requests to Kibana and deliver a response to the corresponding views.

4. Conclusion

BigPanDA monitoring is a Web application developed to cover the various information needs of PanDA users such as access to large scope analytics dashboards, jobs and tasks summaries, results of logs processing, and payload objects state. To provide a smooth user experience of working with BigPanDA, a different acceleration of the following approaches has been introduced: proactive caching, processing-over-data and dynamic data loading. The described system serves not only human requests, but is also used as a data processing engine with almost 7,000 JSON requests per day. The future plans for BigPanDA monitoring include further acceleration of processing speed, following the growing needs of the users and a deeper content personalization.

Acknowledgements

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award No. DE-SC-0012704. This work is partly funded by the Russian Federal budget (Government Task "SCIENCE") project No. 2.9223.2017/8.9

References

[1] ATLAS Collaboration 2008 The ATLAS Experiment at the CERN Large Hadron Collider, JINST 3 S08003

[2] Maeno T for the ATLAS Collaboration 2008 PanDA: Distributed production and distributed analysis system for ATLAS, *J. Phys.: Conf. Series* 119 062036

[3] Bird I 2011 Computing for the Large Hadron Collider *Annual Review of Nuclear and Particle Science* vol 61 pp 99-118

[4] Abbon P et al. 2007 The COMPASS experiment at CERN *Nucl. Instrum.Meth.* A577 455

[5] AJAX project, "AJAX" [software], version 3.2, 2017. Available from http://api.jquery.com/category/ajax/ [accessed 2017-09-30]

[6] AngularJS project, "AngularJS" [software], version 1.2, 2017. Available from https://angularjs.org/ [accessed 2017-09-30]

[7] Nginx project, "nginx" [software], version 1.12.1, 2017. Available from https://nginx.org/en/download.html [accessed 2017-09-30]

[8] Django project, "Django" [software], version 1.11.5, 2017. Available from https://docs.djangoproject.com/en/1.11/ [accessed 2017-09-30]

[9] ZURB Foundation project, "ZURB Foundation" [software], version 6, 2017. Available from http://foundation.zurb.com/sites/docs/ [accessed 2017-09-30]

[10] ElasticSearch project, "Elastic Search" [software], version 5.0, 2017. Available from https://www.elastic.co/products/elasticsearch [accessed 2017-09-30]

[11] Maeno T et al. 2014 Evolution of the ATLAS PanDA workload management system for exascale computational science *J. Phys.: Conf. Ser.* 513 032062

[12] De K, Golubkov D, Klimentov A et al. 2014 Task management in the new ATLAS production system. *J. Phys.: Conf. Ser.* 513 032078

[13] Nilsson P et al 2011 *J. Phys.: Conf. Ser.* 331 062040

[14] Garonne V et al 2014 *J. Phys.: Conf. Ser.* 513 042021

[15] Andreeva J, Campana S, Karavakis E et al. 2012 ATLAS job monitoring in the Dashboard Framework *J. Phys.Conf. Ser.* 396

[16] Oracle DB project, "Oracle" [software], version 11g, 2017. Available from http://www.oracle.com/technetwork/database/database-technologies/express-edition/downloads/index.html [accessed 2017-09-30]

[17] Django Databases Middleware project, "Django Databases" [software], version 1.11, 2017. Available from https://docs.djangoproject.com/en/1.11/ref/databases/ [accessed 2017-09-30]

[18] MySQL project, "MySQL" [software], version 5.7, 2017. Available from https://www.mysql.com/ [accessed 2017-09-30]

[19] Nelson J 2016 Mastering Redis. *Birmingham:Packt Publishing* p 340

[20] ELK stack project, "ELK" [software], version, 2017. Available from https://www.elastic.co/products [accessed 2017-09-30]

[21] Alekseev A, Barreiro Megino F, Klimentov A, Korchuganova T, Maeno T and Padolski S 2017 Applying Big Data solutions for log analytics in the PanDA infrastructure *Symp. on Nuclear Electronics and Computing* (Montenegro: Budva) https://indico.jinr.ru/getFile.py/access?contribId=199&sessionId=18&resId=0&materialId=slides&confId=151 [accessed 2017-10-02]