EPJ Web of Conferences **251**, 02031 (2021) https://doi.org/10.1051/epjconf/202125102031
CHEP 2021

Prototype of the Russian Scientific Data Lake

Aleksandr Alekseev[1,2,3], *Xavier* Espinal[10,*], *Stephane* Jezequel[9,**], *Andrey* Kiryanov[1,4,***], *Alexei* Klimentov[5,****], *Tatiana* Korchuganova[1,2,3], *Valeri* Mitsyn[6], *Danila* Oleynik[1,6], *Alexander* Smirnov[1,7], *Sergei* Smirnov[7], and *Andrey* Zarochentsev[1,8,†]

[1]Plekhanov Russian University of Economics, Moscow, 117997, Russia
[2]Ivannikov Institute for System Programming of the RAS, Moscow, 109004, Russia
[3]Universidad Andrés Bello, Santiago, 7550196, Chile
[4]NRC "Kurchatov Institute" - PNPI, Gatchina, 188300, Russia
[5]Brookhaven National Laboratory, Upton, NY 11973, USA
[6]Joint Institute for Nuclear Research, Dubna, 141980, Russia
[7]National Research Nuclear University MEPhI, Moscow, 115409, Russia
[8]Saint Petersburg State University, Saint Petersburg, 198504, Russia
[9]Laboratoire d'Annecy de Physique des Particules (LAPP), Annecy le Vieux, 74941, France
[10]European Organization for Nuclear Research (CERN), Geneva, 1211, Switzerland

Abstract. The High Luminosity phase of the LHC, which aims for a tenfold increase in the luminosity of proton-proton collisions is expected to start operation in eight years. An unprecedented scientific data volume at the multi-exabyte scale will be delivered to particle physics experiments at CERN. This amount of data has to be stored and the corresponding technology must ensure fast and reliable data delivery for processing by the scientific community all over the world. The present LHC computing model will not be able to provide the required infrastructure growth even taking into account the expected hardware evolution. To address this challenge the Data Lake R&D project has been launched by the DOMA community in the fall of 2019. State-of-the-art data handling technologies are under active development, and their current status for the Russian Scientific Data Lake prototype is presented here.

1 Introduction

Modern high energy and nuclear physics experiments, which are being carried out at the accelerator complexes of the LHC (CERN, Switzerland) [1], SuperKEKB (KEK, Japan), RHIC (BNL, USA), and in the coming years at NICA (JINR, Russia) and FAIR (GSI, Germany), deal with hundreds of petabytes of scientific data located in billions of files. Starting from 2014 the LHC experiments analyze annually more than two exabytes of physics data processing millions of files per day, and the total permanent storage capacity for the experiments has exceeded the exabyte level. The next project at CERN, the High Luminosity LHC (HL-LHC), aims to increase the intensity of proton-proton collisions tenfold, which will lead to

*e-mail: xavier.espinal@cern.ch
**e-mail: stephane.jezequel@cern.ch
***e-mail: kiryanov@cern.ch
****e-mail: aak@bnl.gov
†e-mail: andrey.zarochencev@cern.ch

an even greater amount of incoming scientific information. Such enormous data volumes and the amount of files require new techniques for their storage and processing [2][3], and one of the proposed approaches is to use the "data lake" concept [4][5]. The architecture development and prototyping of "scientific data lakes" is being carried out as a part of the DOMA [6] project (WLCG [7], CERN). The work on the project is closely related to:

- the development of data management systems and data streams for processing and analyzing information in the exabyte range;
- the development and testing of scenarios for connecting computer centers to the "data lake" infrastructure depending on the level of their resources and the quality of communication channels (including supercomputer centers);
- the development of methods for determining the popularity (demand) of particular scientific data sets and data management methods.

In this paper, we will mainly discuss the development of the Russian Scientific Data Lake prototype. This work is carried out by Russian research centers (NRC KI - PNPI, ISP RAS) and universities (SPbSU, MEPhI, RUE) in cooperation with international scientific centers (CERN, JINR, LAPP, UNAB).

2 Problem description and challenges

Taking into account the new requirements for data storage and data processing systems for the HL-LHC project, the R&D activities are currently underway on technologies for building such systems using the "data lake" concept which would work as a part of distributed heterogeneous high-throughput global computing system. To obtain relevant research results, each of the possible technologies must be tested using the uniform methodology.

It is also necessary to demonstrate and describe a methodology for connecting relatively small computing systems without associated storage elements to already existing distributed computing infrastructure. This methodology should significantly decrease the "entry threshold" for the small sites (analysis facilities), which for some reason cannot provide deployment and support of dedicated data storage systems. Provided methodology needs to reduce the complexity of deployment and maintenance requirements of components being installed, as well as bring down the overhead costs, which would allow even smaller sites to work effectively with the data lake.

To reduce the complexity of installing and deploying of software components, the modern container technologies can be used, which allow for almost instantaneous deployment of many identical instances of a prepared software stack that requires just a few parameters to configure.

Concerning the reduction of overhead costs, the authors see an increase in the efficiency of using the CPU resources of computational nodes by minimizing the time the CPU is idle due to data I/O operations. In the standard workflow for distributed data processing of scientific experiments a task allocated to a computational node must first receive input data from the storage system, and then write output data into it. Both operations can take a significant amount of time compared to the whole task processing time, although they practically do not occupy the CPU. Minimizing the time for data reading and writing at the beginning and at the end of a computational task can significantly increase the efficiency of using the CPU resources.

Combining these two directions of increasing the data processing efficiency discussed above, the development of a data lake prototype with one central storage, which has high reliability, capacity and availability, and several satellite computing sites with small storage

resources used as a short-term data storage (cache or buffer) will be shown in this paper. At the same time the data caching for reading and writing will be considered separately, since the expediency of its use, as well as the technologies used for this, depend on the types of computational tasks.

3 The prototype architecture

To build the data lake prototype a test site will be considered with dedicated computing resources, consisting of several identical computing nodes that can execute both synthetic test tasks and tests based on real tasks of scientific experiments. Also, a monitoring system will be considered, which allows for evaluating the efficiency of using the computing power accessing the data lake, and on the basis of this assessment, a conclusion will be made about the feasibility and efficiency of using certain caching schemes.

3.1 Data caching

Three main configurations of the caching system were evaluated and studied (Fig. 1):

1. a dedicated caching server — central control of caching, possibility to quickly change the storage hardware without touching the computational nodes;
2. a local caching server on each computational node with no data exchange between the nodes — good for fast deployment of multiple independent instances;
3. a local caching server on each computational node with data exchange between the nodes forming a distributed cache (one of the computational nodes additionally acts as a control node) — a somewhat more complicated configuration than 2 but with better LAN utilisation and higher cache hit rates.

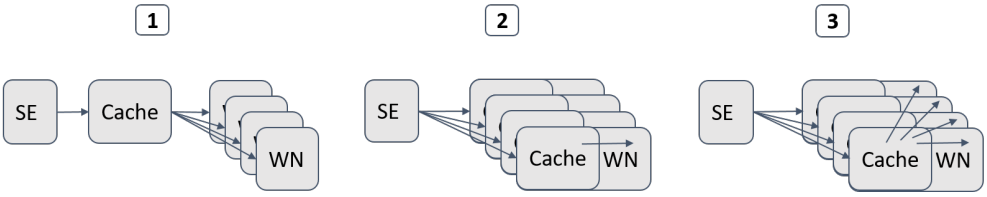


Figure 1. The caching system configurations

Configuration 2 has been used for primary evaluation tests only. This is due to the fact that the number of repeated requests to the same input data file in LHC workflows is low, and is less than the average number of computational nodes even on relatively small sites, which leads to the nearly zero hit ratio for an isolated node-local cache, rendering it practically useless.

It is more interesting to compare configurations 1 and 3. In configuration 1 all the burden of storing, searching and copying files from the main storage lies on a dedicated server, almost completely freeing up worker nodes. In the case of configuration 3 with a distributed cache there are savings due to the lack of a dedicated server, but there is an additional load on the worker nodes. Also, configuration 3 allows to create a scalable hybrid CE + SE system (compute plus storage) based on one unified configuration containing all the necessary

components. Such a system can be scaled horizontally by adding more instances, which is extremely convenient when deploying to cloud resources. In addition to this, there appears, although not very high, the probability of reading the input file from the cache located on the same worker node as the requesting task, which further optimizes the data transfer. This effect is unlikely to have any significant impact on large sites, but can have a visible impact on configurations with a few multicore worker nodes.

3.2 Data buffering

In order to further optimize the usage of CPU resources for write-oriented workloads the development of a buffering system for output data was started in 2020. In the case when the analysis task generates a significant amount of output data, the time required to transfer this data from a worker node to a remote storage can be accounted for as a wasted CPU time. If the generated data is temporarily stored on a fast local server (a buffer) then the CPU time will be saved and the efficiency of the site will increase. This can be expressed, among other things, in the average processing time of a single task. The proposed scheme is shown in Figure 2.

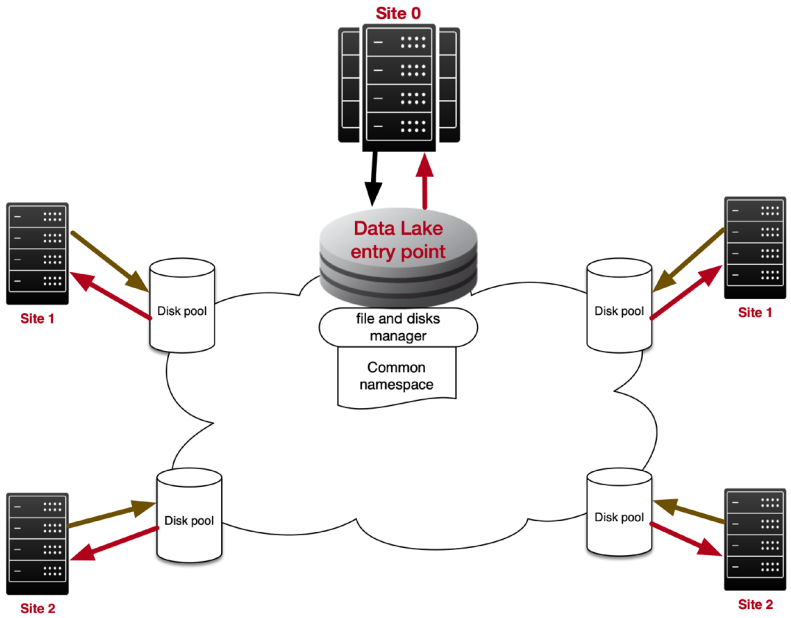


Figure 2. Proposed scheme of data transfers in the data lake prototype with the buffering system

4 The prototype implementation

4.1 Data caching

The testbed for the data caching system was deployed at the resource centers at JINR (Dubna), PNPI (Gatchina), MEPhI (Moscow), and RUE (Moscow). The central storage located at JINR is operated under the control of the dCache storage system. The local storage systems on sites

were implemented using the XCache package, which was configured, depending on scenario, either as a dedicated cache on a separate server or as a distributed cache on worker nodes. The GSI authorization system with WLCG X.509 certificates was configured, which made it possible among other things to work with WLCG resources and to use HammerCloud testing system.

In addition the registration and configuration of infrastructure elements in the AGIS information system (now migrated to CRIC [9]) was carried out before testing. The following combinations of computing components and storage systems (task queues) have been defined to test various scenarios of caching:

1. PNPI_XCache-TEST. Computing cluster at PNPI with a dedicated caching server based on XCache connected to the dCache storage system located at JINR (JINR-LCG2_DATADISK)

2. PNPI_XCache_NODES. Computing cluster at PNPI with a caching service based on XCache using local disk resources of worker nodes combined into a distributed cache, connected to the dCache storage system located at JINR (JINR-LCG2_DATADISK)

3. PNPI-TEST. Computing cluster at PNPI connected to the dCache storage system located at JINR (JINR-LCG2_DATADISK)

4. MEPHI-TEST. Computing cluster at MEPhI connected to the dCache storage system located at JINR (JINR-LCG2_DATADISK)

5. MEPHI_XCache. Computing cluster at MEPhI with a dedicated caching server based on XCache connected to the dCache storage system located at JINR (JINR-LCG2_DATADISK)

4.2 Data buffering

The testing of the data buffering is carried out on the same testbed, but the EOS [8] system with a control server at JINR and storage servers at all sites is used as a distributed storage. In the future, it is planned to configure queues in the CRIC system and to develop special test templates in the HammerCloud system in order to test the operability of the distributed storage.

EOS was chosen as the storage system which is supposed to serve as a basis for testing the buffering mechanism because it meets all the storage requirements for the data lake and, at the same time, has two handy built-in features: an LRU Engine and a Converter Engine. The former is a mechanism that automatically scans virtual namespace and applies the prescribed policy to files depending on their name, type, size, creation time, etc, while the latter can automatically convert file layouts (striping, location, etc) based on the policy. Combination of these two engines was used to implement buffering. During the tests some technical inconsistencies were discovered between LRU and Converter engines, which were reported to the EOS developers. A temporary tweak was applied, which made it possible to continue testing, and the issue was fully resolved in the later EOS release.

4.3 Testing methodology

By the time of testing the configuration 3 with a distributed cache, synthetic tests have been improved. If in 2019 and early 2020 it was just a sequential copying of the same file to a working node, then by the end of 2020 the synthetic tests were modified to implement a more realistic load:

- A set of different files is taken, in which one file is repeated no more than 20 times (the limitation is taken from the evaluation of the data popularity in the ATLAS experiment)
- Each iteration starts on a separate, randomly selected worker node. The main measured parameter is not the file copy time (since file sizes are different this parameter does not make sense anymore), but the transfer speed, that is, the the file size over the time required to copy it in GB/s.

4.3.1 Testing the Russian Scientific Data Lake computing infrastructure with the ATLAS payloads

To test the computing infrastructure of the Russian Scientific Data Lake, the HammerCloud [10] system is used, which creates computational testing tasks based on the defined templates and sends them to the ATLAS experiment [11] distributed production and analysis system (PanDA) [12]. In this system, five new test patterns were developed, depending on the methods of accessing the input data for processing:

- Copy2Scratch - copy the input files to the worker node and read the file from the local disk (as an input file root://local path/file)
- Directaccess - direct reading of a remote file from tasks (root://remote path/file is specified as an input file)
- TTreeCache - direct reading of individual trees of a remote file (root://remote path/file is written as input)
- Copy2Scratch (for long-running tasks) - similar to the Copy2Scratch template, but with the number of physics events increased to 8 thousand per input file.
- TTreeCache (for long-running tasks) is similar to the TTreeCache pattern, but with the number of physics events increased to 8 thousand per input file.

The data set used by each template consists of 52 input files for processing in the AOD (Analysis Object Data) format, which has a large number of physics events per input file. At each stage, a set of tests is carried out for each test template, and according to their results the errors in the operation of the infrastructure components can be corrected and the results obtained can be analyzed.

5 The monitoring system

To estimate the efficiency of computer resources usage and to monitor the software and hardware infrastructure in the process of testing the Russian Scientific Data Lake prototype, a unified monitoring system was developed using modern web and database technologies. At the current stage of the prototype development the following distributed data sources are used to monitor the operation of the deployed software infrastructure:

- XRootD log files, which contain complete information about the XCache service operations;
- Billing relational database (dCache) based on PostgreSQL DBMS. The database contains information about operations (billinginfo) and requests for an operation (doorinfo)
- A special API provided by the monitoring system for distributed production and analysis of ATLAS experiment data - BigPanDA [13], to obtain information and necessary metrics describing test jobs. In particular, the job type, the queue of the computing resource and the node where the job was completed, the final job status, error messages, the list of

input files, the amount of consumed RAM, time metrics: total job execution time; time of job initialization; loading time of input files; time of uploading output files; time of transformation of input files into output ones, etc.

- Accounting database based on MySQL DBMS. The database contains information about the local jobs that describe the current state of computing elements.

ElasticSearch was chosen as a unified data storage, which is part of the ELK stack. It provides efficient access to aggregated information from distributed sources due to the flexibility of storage schemes. The general architecture diagram of the unified monitoring system is shown in Figure 3.

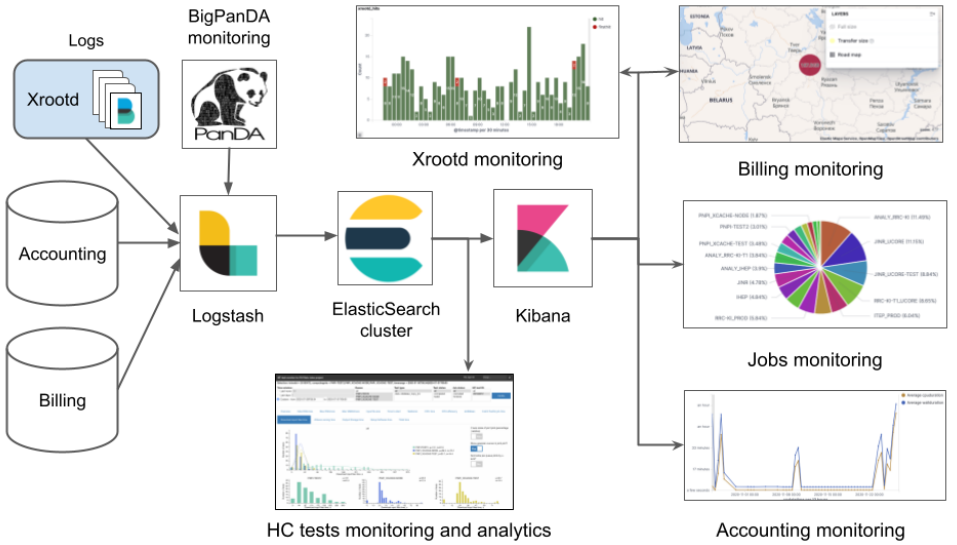


Figure 3. The unified monitoring system architecture

Using the PNPI computing infrastructure, all the necessary components included in the monitoring system were deployed, in particular, the ElasticSearch storage, the Logstash input data flow collector, the Kibana visualization system and a specially developed web application for monitoring and analysis of the test jobs execution results, which implements specific data visualizations which are not natively available in the Kibana. Thus, data streams from all sources pass through the Logstash collector, which converts heterogeneous data into key-value format and stores it in specially dedicated indexes of ElasticSearch document-oriented non-relational storage. In the expanded Kibana visualization system, dashboards were created for the software components of the Russian Scientific Data Lake prototype, which consist of graphs, tables and other available types of visualizations. The following monitoring dashboards were implemented in the Kibana system using the collected data:

- XRootD monitoring. Dashboard is designed to monitor the XRootD component and is based on processed data from log files;
- Billing monitoring. This monitoring section allows to get information about the work of dCache, which is extracted from two separate tables of the Billing database combined and exported to the ElasticSearch index;
- Jobs monitoring. Based on information about all test jobs executing on the testbed;

- Accounting monitoring. The "data lake" includes sites with a computing element based on CREAM CE, an outdated system at the moment, and computing elements based on ARC CE, which is currently recommended for use in the WLCG.

Due to the limited number of available visualisation types in the Kibana system and the lack of customizing functionality, we developed a special web application for monitoring and analyzing the results of test jobs. The application is based on Django framework. Nginx is used as a web server together with uWSGI to interpret the application code. The Angular is responsible for interactive communication and data transfer between the server and client sides. The ZURB Foundation style library is used to display the structural blocks of a web page. The C3.js library is used to build visualizations. The DataTables plugin is used to build interactive tables. An example of a web application interface is shown in Figure 4.

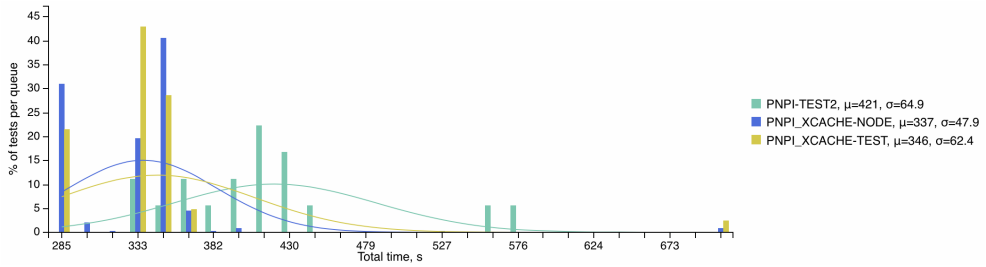
Figure 4. The monitoring web application interface

The interface of the web application was designed in such a way that allows the user to interactively select a set of test jobs by the most convenient attributes, in particular the time when the test task was executed, the queue of the computing resource, the type of tasks, the status and the test ID in the HammerCloud system. The results are displayed on 16 tabs, the first of them ("Overview") contains general information about the number of selected tests per queue and a list in tabular form. The rest of the tabs contain bar diagrams with the most useful metrics of test performance, for each of the metrics a general diagram is built and separate ones for each computational queue (Fig. 4). In addition for each queue the average value of the metric and its standard deviation are displayed. It is also possible to plot the curves of the normal distribution functions (Gaussians) for a better visual comparison of the metric value distributions between queues. Due to the fact that each queue represents a certain configuration of the architecture of the Russian Scientific Data Lake prototype, this type of data visualization makes it easy to estimate the efficiency of resource usage and the advantage of one architectural option over another.

6 Results and Conclusions

The data lake prototype has been implemented and the synthetic tests were carried out for three data caching configurations. The results have demonstrated almost no gain of data access time for a configuration with an isolated local cache in comparison to a configuration without a cache. For the scenarios with dedicated and distributed caches (configurations 1 and 3) a significant gain has been observed in comparison with a system without a cache. Somewhat unexpectedly, tests for a system with a distributed cache showed a noticeably worse result compared to a dedicated cache: the average speed for configuration (3) is 1.8 ± 1.5 GB/s, while the average speed for configuration (1) is 2.6 ± 1.5 GB/s. Speed without a cache is 0.5 ± 0.5 GB/s. The large error in the results is due to the relatively small number of duplicate files in the tests - no more than 20. This leads to the fact that in the case of caching at least every 20th request to the file is a cache miss, which requires the cache to fetch the file from the data lake.

ATLAS real-life payloads have been launched via Hammercloud after evaluating the synthetic tests results and system tuning. Copy2scratch test results are presented on Fig. 5.



PNPI-TEST2 - no cache; PNPI_XCache-NODE - the distributed cache;
PNPI_XCache-TEST - the dedicated cache; μ - the average value; σ - standard deviation.

Figure 5. Total time distribution of copy2scratch tests per queue

The results showed the minimal, within the margin of error, difference between the distributed and dedicated caches and the gain of these two systems in comparison to the case without a cache. Preliminary tests have been carried out for the data buffering: a policy has been set up according to which files are copied to the nearest storage pool by Geo Tag. The LRU policy is tied to the directory where the file is copied, according to which each file with a lifetime of more than 10 minutes is moved to the central storage server. The files copied from the worker nodes at PNPI were processed properly, according to the defined policy. A work has begun on the creation of synthetic tests that are similar in characteristics to real payloads, as well as a work on adaptation of real payloads from ALICE and ATLAS experiments for the HammerCloud system. Based on the results at this stage of R&D, we have demonstrated a benefit of using data caching for the ATLAS data processing payloads (for the cases when the same input files are read 20 times or more). Also, the effectiveness of creating a distributed cache on worker nodes for such tasks has been confirmed, in the case when the installation of a dedicated caching server seems difficult. Assessing the advantages of a dedicated cache over a distributed one remains at the moment an unresolved task, planned for the next year of work. The necessary monitoring tools for this task have been created, but it is necessary to accumulate some representative statistical data. For the buffering system, a basic concept is presented, its practical implementation and testing is expected in 2021.

As our results look quite promising we will continue to exploit the presented ideas, in the future transforming them into practical, easily deployable software configurations possibly in the form of ready-made containers.

7 Acknowledgements

We would like to thank our ATLAS colleagues for a valuable discussion about monitoring and HammerCloud tests implementation. The research and development of the Russian Scientific Data Lake in Plekhanov University is financially supported by the Russian Science Foundation award #19-71-30008.

References

[1] L. Evans, P. Bryant, LHC Machine, 2008, Journ. of Instrum. 3, S08003

[2] A. Kiryanov et al, Federated data storage system prototype for LHC experiments and data intensive science, 2016, J. Phys.: Conf. Ser. 898 062016

[3] A.Klimentov, Exascale Data Processing in Heterogeneous Distributed Computing Infrastructure for Applications in High Energy Physics, Physics of Particles and Nuclei, 51(6), 995-1068, DOI 10.1134/S1063779620060052

[4] A. Klimentov et al, Russian scientific data lake, Open Science Platforms, 2018, vol. 4.

[5] A. Alekseev et al, On the road to a scientific data lake for the High Luminosity LHC era, 2020, International Journal of Modern Physics A, Vol. 35, No. 33, 2030022

[6] D.Berzano et al, HEP Software Foundation Community White Paper Working Group - Data Organization, Management and Access (DOMA), 2018, arXiv:1812.00761

[7] J. Shiers, 2007, Computer Physics Communications 177, 219–223

[8] A.Peters et al EOS as the present and future solution for data storage at CERN, 2015 J. Phys.: Conf. Ser. 664 042042

[9] A. Anisenkov et al, CRIC: Computing Resource Information Catalogue as a unified topology system for a large scale, heterogeneous and dynamic computing infrastructure, 2020, EPJ Web Conf. 245 03032

[10] J. Elmsheuser et al, Grid sites testing for ATLAS with Hammercloud, 2014, J. Phys.: Conf. Ser. 513 032030

[11] ATLAS Collaboration, The ATLAS Experiment at the CERN Large Hadron Collider, 2008, Journ. of Instrum. 3, S08003

[12] F H Barreiro Megino et al, PanDA for ATLAS distributed computing in the next decade, 2017, J. Phys.: Conf. Ser. 898 052002

[13] A. Alekseev et al, ATLAS BigPanDA monitoring, 2018, J. Phys.: Conf. Ser. 1085 032043