Integration of Rucio Metadata in Belle II

Cédric Serfon[1,*], *Anil* Panta[2], *Hironori* Ito[1], *John Steven* De Stefano Jr[1], *Michel* Hernández Villanueva[3], *Paul* Laycock[1], *Ruslan* Mashinistov[1], *Hideki* Miyake[4,5], and *Ikuo* Ueda[4,5]

[1]Brookhaven National Laboratory, Upton, NY, USA
[2]University of Mississippi, MS, USA
[3]Deutsches Elektronen-Synchrotron DESY, Hamburg, Germany
[4]High Energy Accelerator Research Organization (KEK), Tsukuba 305-0801, Japan
[5]The Graduate University for Advanced Studies (Sokendai), Tsukuba 305-0801, Japan

Abstract. Rucio is a Data Management software that has become a de-facto standard in the HEP community and beyond. It allows the management of large volumes of data over their full lifecycle. The Belle II experiment located at KEK (Japan) recently moved to Rucio to manage its data over the coming decade (O(10) PB/year). In addition to its Data Management functionalities, Rucio also provides support for storing generic metadata. Rucio metadata already provides accurate accounting of the data stored all over the sites serving Belle II. Annotating files with generic metadata opens up possibilities for finer-grained metadata query support. We will first introduce some of the new developments aimed at providing good performance that were done to cover Belle II use-cases like bulk insert methods, metadata inheritance, etc. We will then describe the various tests performed to validate Rucio generic metadata at Belle II scale (O(100M) files), detailing the import and performance tests that were made.

1 Introduction

The Belle II experiment [1] is a B-factory experiment located on the SuperKEKB [2] accelerator complex at KEK (Japan). Belle II has established its computing system based on distributed computing technologies[3] and it recently changed its Data Management system to use Rucio [4]. This migration [5], [6] opened new possibilities regarding computing, in particular the possibility to use Rucio as a Metadata Service in addition to the Data Management functionalities already in use. Rucio metadata functionalities are briefly introduced in section 2. Developments done in Rucio and BelleDIRAC [7], the extension of DIRAC [8], in order to accommodate Belle II needs are described in section 3. To validate them and check that Rucio metadata functionalities fit the purposes and use for Belle II, different tests were performed as well as a test import of all metadata into the Rucio production instance. All these tests are described in section 4.

[*]e-mail: cedric.serfon@cern.ch

2 Introduction to Rucio metadata

Rucio was initially designed to support a list of fixed metadata for all DIDs[1] it supports. These metadata are recorded as columns in the table defining the DIDs and therefore the type of metadata is fixed (string or numeric value). These fixed metadata, that will be also named "column metadata" hereafter, are very suitable to perform data accounting.

Because of the limited use-case of the column metadata, support to more metadata possibilities was added to Rucio :

- Internal generic metadata stored under json format in Rucio database. This allows to store any kind of metadata as a key, value pair inside Rucio. This type of metadata will be named as "json metadata" in the following.
- Support of external metadata database backend (e.g. PostGreSQL or MongoDB) or service such as metacat [9]. This type of external metadata is not considered in this paper.

Taking into account these different possibilities, we decided to use column metadata to store "accounting metadata" (i.e. metadata that can be used to generate space accounting reports) and json metadata to store all the rest.

3 Developments

In order to be able to serve all Belle II use-cases regarding metadata, different development were performed in Rucio and BelleDIRAC.

3.1 Developments in Rucio

Belle II uses a hierarchical namespace similar to a file system with directories and files which maps nicely with the concept of DIDs in Rucio. In Belle II, different metadata can be applied to files or to directories and for some workflows, we are interested in getting the metadata of the files and from their parents. To achieve this, the possibility for files to "inherit" the metadata of their parents was added. In addition to this, different bulk methods to add or get metadata were added.

3.2 Developments in BelleDIRAC

Developments were also performed in BelleDIRAC. New metadata methods were implemented into the RucioFileCatalog client of BelleDirac. These methods are the same as the methods implemented for the DiracFileCatalog[10] client, allowing to use Rucio in a similar way as the Dirac File Catalog. This code is not Belle II specific, leaving the possibility to be merged into "standard" Dirac.

Other changes were applied into the code for pilot jobs in order to register the metadata for the output files. The metadata are extracted from the so called JDL (Job Description Language) file that contains the information relative to the job to run on the grid. Metadata (both accounting and generic metadata) are set into Rucio in parallel to the current metadata service called AMGA[11] as shown on figure1.

Finally, the production system was modified to allow the registration of metadata at the directory level.

[1]Data in Rucio are organized using Data IDentifiers (DIDs). There are three levels of granularity for DIDs: files, datasets, and containers

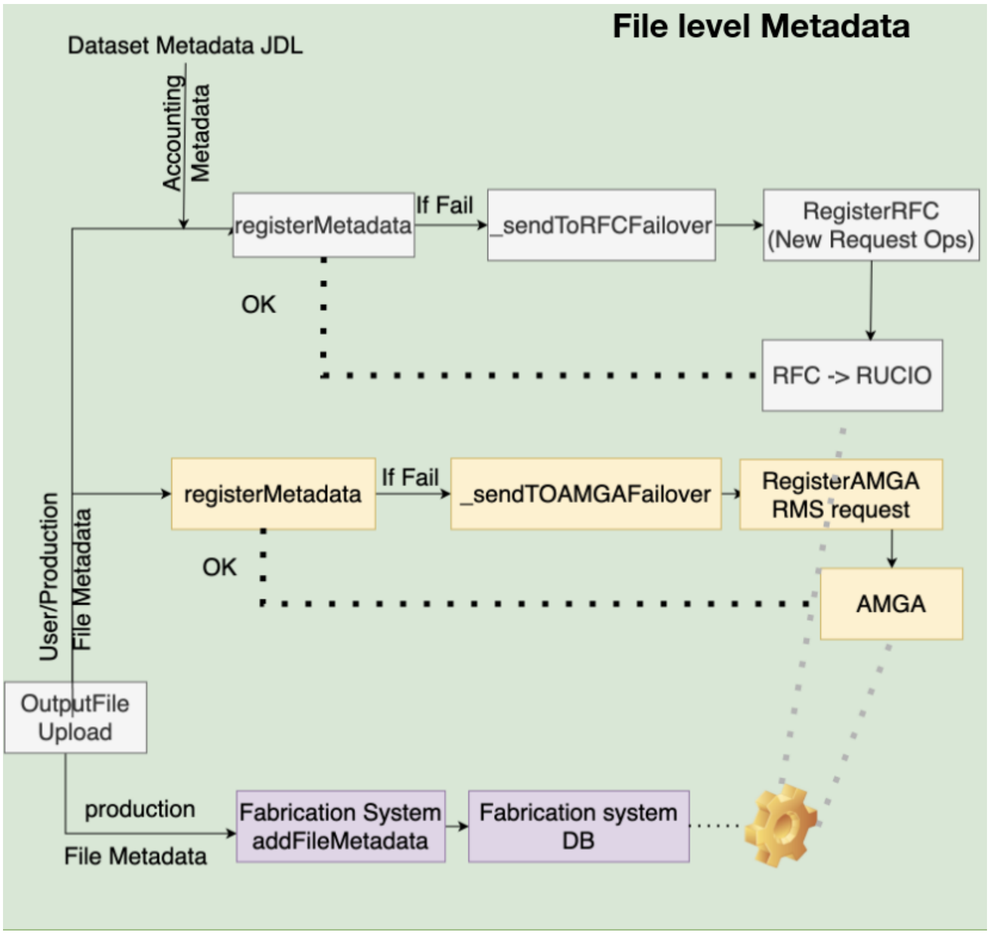

Figure 1. Schema detailing the setting of file metadata. The metadata are registered into Rucio by the pilot calling the registerMetadata method. If the registration fails, a request is inserted in the Dirac RMS Service to register the metadata on the server side. RFC is the acronym for Rucio File Catalog. Fabrication System is a subsystem of the Production System.

4 Tests and transition

4.1 Scaling tests

In order to validate that Rucio is able to handle the Belle II load, stress tests were conducted on a test Rucio instance. The database of this instance is a snapshot of the Belle II Rucio production one and run on a similar hardware. The frontend is only made of one node versus two on the production instance.

One of the tests consisted in inserting, then reading the inserted metadata continuously. The test takes a random file already registered into Rucio, inserts 7 json metadata in a single bulk operation to this file, then reads all metadata back. This operation is reiterated 5000 times over different files without interruption. So in total 35000 metadata are inserted and 5000 get metadata calls are done for one test. To increase the load even more, the same test was sent to a batch system. During the batch test, 360 tests were run in parallel during 3 hours. The number of metadata inserted reached then $360 \times 35000 = 12.6 \cdot 10^6$ in 3 hours, which means 1.16 kHz which is orders of magnitude higher than what is required in Belle II workflows. Similarly, the number of get metadata calls is $360 \times 5000 = 1.8 \cdot 10^6$ in 3 hours, which means 166 Hz. Figure 2 and 3 show the load on the database and Rucio frontend server respectively. The load on the database stays very low ∼3% whereas the load on the Rucio frontend reaches almost 70%. This load could be easily brought down by increasing the number of frontends.

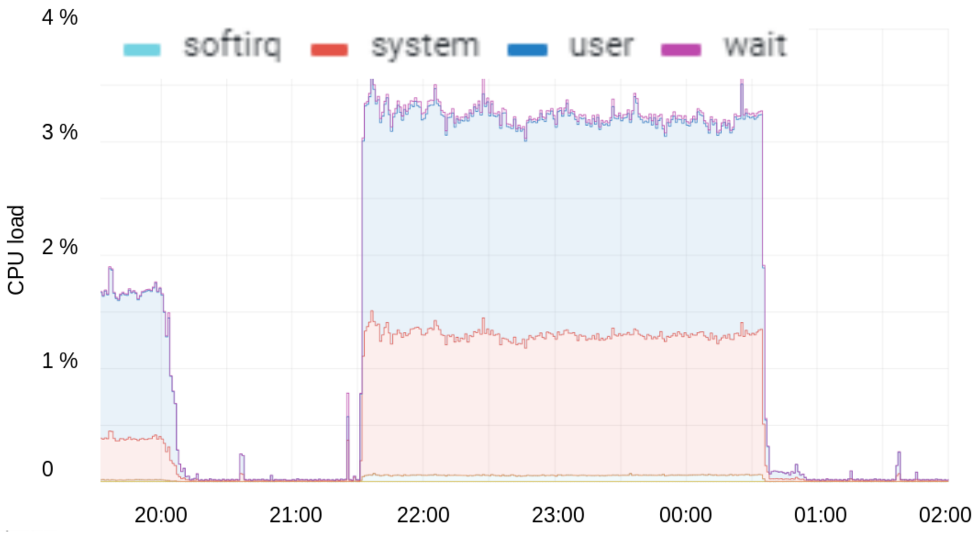

Figure 2. CPU usage of the Database node during the stress test. The highest plateau corresponds to the maximum load of the test (360 parallel jobs) with the load reaching only 3%

4.2 Metadata import

After the tests were run successfully, an import of metadata from AMGA to Rucio was conducted. Since the Rucio production instance and the AMGA one are located far away (Rucio at Brookhaven National Lab (BNL) and AMGA at KEK, which represents a Round Trip Time of 180 ms), a snapshot of the AMGA database was taken and reinstalled at BNL. For scalabity reason, the import procedure runs an SQL query on the AMGA database and populates the Rucio production database. The import process had little impact on the Rucio service and was able to run in background mode. In order to check the impact on the database regarding the disk usage, the import was done gradually over a three months period (see figure 4). At the end of the import, the space used on the database by the metadata represented roughly 1 kB per file including table space + indices.

5 Conclusion

After the successful migration of Belle II to Rucio as Data Management service, work was performed to adapt it to Belle II computing in order to use Rucio's metadata functionalities. The new developments allowed to cover all the Belle II use cases already covered by Belle II's current metadata service. Stress tests have shown that the service is able to handle the

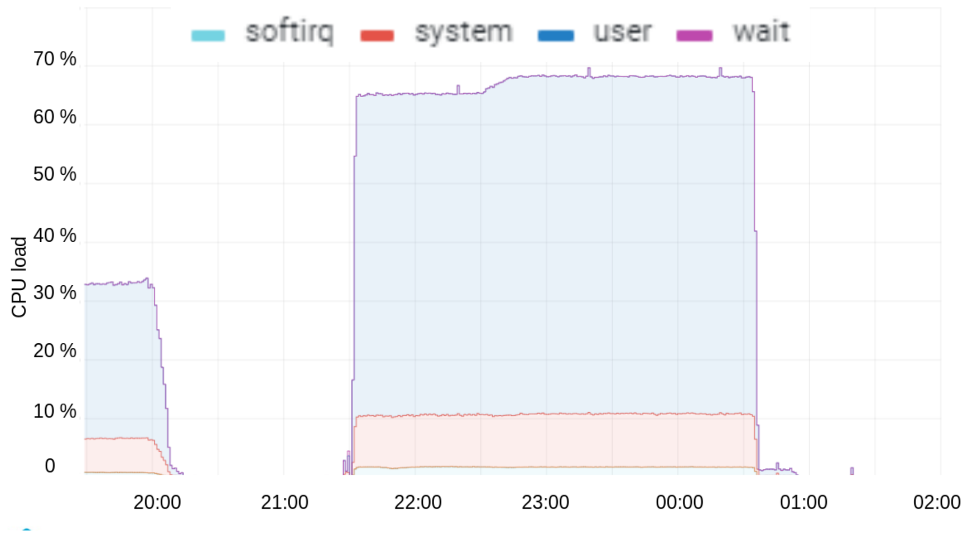

Figure 3. CPU usage of the Rucio frontend server during the stress test. The highest plateau corresponds to the maximum load of the test (360 parallel jobs) with the load reaching about 70%

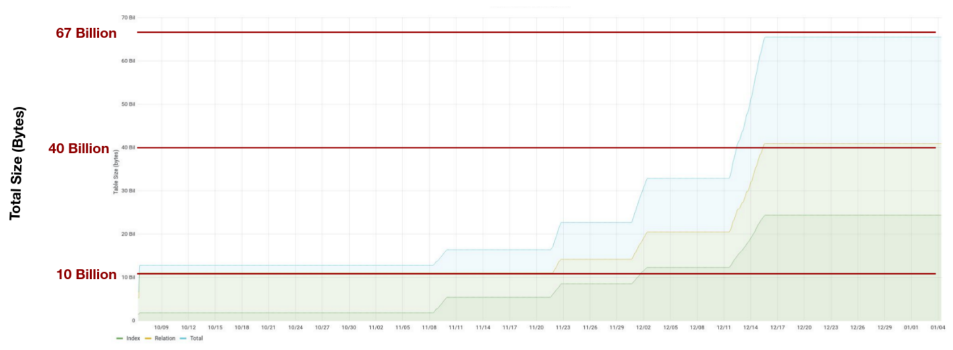

Figure 4. Evolution of the size of the metadata table vs time. The import of metadata was done gradually on the Belle II production instance over a 3 months period to limit the impact on Rucio and to monitor any potential issue.

Belle II load and well beyond. Import of metadata into the production Rucio database was performed and didn't cause any service degradation. For all these reasons, Rucio is a good long term solution to become Belle II metadata service. New developments can even improve the performances, e.g. providing an atomic method to register file replicas and metadata would allow to halve the number of requests to Rucio.

Acknowledgements

The work at Brookhaven National Laboratory is funded by the U.S. Department of Energy, Office of Science, High Energy Physics contract No. DE-SC0012704.

References

[1] T. Abe, I. Adachi, K. Adamczyk, S. Ahn, H. Aihara, K. Akai, M. Aloi, L. Andricek, K. Aoki, Y. Arai et al., *Belle ii technical design report* (2010), 1011.0352

[2] K. Akai, K. Furukawa, H. Koiso (SuperKEKB), Nucl. Instrum. Meth. A **907**, 188 (2018), 1809.01958

[3] T. HARA, on behalf of the Belle II computing group, Journal of Physics: Conference Series **664**, 012002 (2015)

[4] M. Barisits et al., Comput. Softw. Big Sci. **3**, 11 (2019), 1902.09857

[5] C. Serfon, J.S. De Stefano, M. Hernández Villanueva, H. Ito, Y. Kato, P. Laycock, R. Mashinistov, H. Miyake, I. Ueda, EPJ Web Conf. **251**, 02026 (2021)

[6] C. Serfon, R. Mashinistov, J.S. De Stefano, M. Hernández Villanueva, H. Ito, Y. Kato, P. Laycock, H. Miyake, I. Ueda, EPJ Web Conf. **251**, 02057 (2021)

[7] H. Miyake, R. Grzymkowski, R. Ludacka, M. Schram, on behalf of the Belle II Computing Group, Journal of Physics: Conference Series **664**, 052028 (2015)

[8] Stagni, Federico, Tsaregorodtsev, Andrei, Sailer, André, Haen, Christophe, EPJ Web Conf. **245**, 03035 (2020)

[9] Igor, Mandrichenko, EPJ Web Conf. **251**, 02048 (2021)

[10] A. Tsaregorodtsev, S. Poss, Journal of Physics: Conference Series **396**, 032108 (2012)

[11] J.H. Kwak, G. Park, T. Huh, S. Hwang, on behalf of the Belle II Computing Group, Journal of Physics: Conference Series **664**, 042041 (2015)