Письма в ЭЧАЯ. 2016. Т. 13, № 5(203). С. 962–968



# EVOLUTION OF THE USE OF RELATIONAL AND NoSQL DATABASES IN THE ATLAS EXPERIMENT

*D. Barberis*[1] *on behalf of the ATLAS Collaboration*

University of Genova and INFN, Sezione di Genova, Italy

The ATLAS experiment used for many years a large database infrastructure based on Oracle to store several different types of non-event data: time-dependent detector configuration and conditions data, calibrations and alignments, configurations of Grid sites, catalogues for data management tools, job records for distributed workload management tools, run and event metadata. The rapid development of "NoSQL" databases (structured storage services) in the last five years allowed an extended and complementary usage of traditional relational databases and new structured storage tools in order to improve the performance of existing applications and to extend their functionalities using the possibilities offered by the modern storage systems. The trend is towards using the best tool for each kind of data, separating, for example, the intrinsically relational metadata from payload storage, and records that are frequently updated and benefit from transactions from archived information. Access to all components has to be orchestrated by specialised services that run on front-end machines and shield the user from the complexity of data storage infrastructure. This paper describes this technology evolution in the ATLAS database infrastructure and presents a few examples of large database applications that benefit from it.

PACS: 29.50.+v; 29.85.-c

## INTRODUCTION

The ATLAS experiment [1] at the LHC accelerator at CERN records each year several billion $p$–$p$, Pb–Pb and $p$–Pb interactions at a centre-of-mass energy that increased from 7 TeV in 2010 to 8 TeV in 2012 and 13 TeV in 2015. The event records, of order 1 MB for raw data and 300–400 MB for Analysis Object Data (AOD) after reconstruction, are grouped into files that contain a few thousand events (a few GB per file). Files containing statistically equivalent events taken under the same conditions are grouped into "datasets". In addition to event data, ATLAS records a wealth of "metadata", i.e., data about the conditions of data taking. Other metadata consist of the information about file and dataset contents, or their locations, and the jobs that produced, or are producing, reconstructed data in the various stages of processing.

All this information, of very diverse nature, needs to be stored "somewhere" and be made readily available to people who analyse ATLAS data. The natural choices are databases, organised differently according to the kind of data to be stored and the insertion rates and

[1]E-mail: dario.barberis@ge.infn.it

access patterns. Until a few years ago only SQL databases were available, and CERN provided a well-supported Oracle service, therefore all ATLAS conditions data and metadata were stored in Oracle. In the last five years there was a number of developments of the so-called "NoSQL databases", better called "structured storage systems", which offer more flexibility in the data storage architecture and ease of use for the developers, at the expense of reduced functionality for real-time applications.

ATLAS started exploring the use of structured storage systems for accounting applications for Grid applications (the data management system Rucio [2] and the job management system ProdSys/PanDA [3]) and later also for the global catalogue of all events, the EventIndex [4].

This paper summarises the different types of data ATLAS stores in databases or other structured storage systems and explains the choices of technologies and implementations for the current LHC run (Run 2) and the future evolution of these systems.

## CONDITIONS DATA AND METADATA

Conditions data are all non-event data that contain information about the data-taking conditions. There are several kinds of conditions data, all sharing the property that they vary with time, and their validity is defined by time ranges:

- Detector read-out and trigger configurations are uploaded to online processors and are valid for the duration of one run;
- Detector operational conditions, such as temperatures, gas pressures and flow rates, high- and low-voltage settings and currents in the read-out electronics are sampled frequently and recorded for offline use;
- Calibration and alignment constants are computed for each time interval within a run and used for event reconstruction;
- LHC luminosity measurements are performed continuously and the "best knowledge" is stored as a function of time, as the luminosity decreases exponentially during each run.

Except for the detector and trigger configurations, which cannot change once they are applied, all other conditions data can be recomputed at any later stage if the understanding of the detector behaviour improves or the quality of the input data increases. Conditions data therefore need a database infrastructure that supports time intervals of validity and also versioning for each data type.

The COOL API and the CORAL interface packages [5] used by ATLAS provide respectively the database structures needed to store conditions data and the interface to several back-end technologies, including the ones used by ATLAS to store the databases (Oracle) or extract subsets (SQLite [6]) and the web service to provide access to the data from the Grid (Frontier [7]).

Several kinds of metadata are used for physics analyses. They include lists of runs that can be analysed together, as taken under the same conditions, accelerator luminosities as a function of time (or run number), data quality and related information. These metadata are created after data taking from the aggregation and processing of information extracted from many different sources. ATLAS uses the AMI (ATLAS Metadata Interface [8]) and COMA (COnditions MetadatA [9]) databases, which work together holding respectively high-level dataset metadata and conditions metadata in a coordinated way, and are both implemented in

Oracle tables; in this way they can share information and the user can seamlessly navigate through the system.

The distributed production/analysis and data management systems produce and need to keep a wealth of metadata about the data that are processed and stored.

• Rucio [2], the distributed data management system, keeps several catalogues at the dataset level (with the list of files, total size, ownership, provenance, lifetime, status, replica locations and ownership, etc.) and at the file level (with size, checksum, number of events, etc.). In addition, there are data transfer tools (with the queue of transferring datasets, status, etc.) and deletion tools (with lists of datasets (or replicas) to be deleted, status, etc.). Finally, there are lists of storage resources and their properties.

• PanDA and related tools [3], the distributed workload management system, keep lists of requested tasks and their input and output datasets, software versions, etc., lists of jobs with their status, location, etc., and processing resource lists with their properties.

Both systems use a combination of quasi-static and rapidly changing information, given that ATLAS uses about 130 Grid sites to run over 1M jobs/day using 200k job slots and move 600 TB/day around the world. Oracle satisfies supports very well both systems if the tables don't grow indefinitely; therefore "old" information is copied to an archive Oracle database and removed from the primary one periodically. Accounting information, as well as selected information to be used for analytics studies, is extracted from the back-up Oracle database and stored in Hadoop [10] for further processing.

Metadata pertaining to event data can also be usefully employed if readily available, for example, to answer the question "Which file contains this event, in which format, and which is the internal pointer to retrieve it?" The EventIndex [4] was developed to hold event-level metadata and uses Hadoop to store the necessary information.

## DATABASE TECHNOLOGIES AND APPLICATIONS FOR ATLAS IN LHC RUN 2

ATLAS relies heavily on Oracle for all major database applications, for several technical but also historical and practical reasons. When ATLAS started in the early 1990s, only SQL databases were available and tested to work at the necessary scale. CERN provides Oracle licences and Oracle RACs (Real Application Clusters), including the system level support, for the use of the experiments; in addition, ATLAS employed since 2006 two experienced Oracle database administrators to help the application development teams to optimise their tools that use Oracle databases. The Oracle setup consists of three main production RACs, each one consisting of four servers and complemented by an active stand-by and a separate back-up replica, plus an archive database for old, but not obsolete, data that are not accessed frequently.

• The ATONR ("online") RAC is used by online applications and contains the configuration data for detector, trigger and DAQ operation, the detector conditions data (DCS), and calibration and alignment constants for online event reconstruction. DCS data are copied as time series to a COOL schema in the offline database ATLR.

• The ATLR ("offline") RAC contains the detector geometry and trigger databases, the detector conditions data imported from the online database, the calibration and alignment constants for offline event reconstruction, the COMA metadata database and a replica of the AMI database, whose master is for historical reasons at the IN2P3 Computer Centre in Lyon.

COOL data are replicated to IN2P3-CC (France), RAL (UK) and TRIUMF (Canada) in order to guarantee an optimal level of accessibility by Grid jobs that run all over the world.

• The ADCR ("distributed computing") RAC contains the configuration data for ATLAS Grid sites, the tables with the file and dataset catalogues for Rucio and the back-end databases for PanDA.

• The ATLARC ("archive") RAC contains old, but not obsolete, data that are not accessed frequently.

In addition, two smaller RACs (INTR and INT8R) are used by developers of database applications to test new versions of applications as well as new versions of the Oracle database software. On all clusters selected users and processes have write access; all ATLAS members have read access. Read access normally takes place through front-end web services (no direct access to Oracle to avoid overloading the servers):

• Frontier [7] for access from production and analysis jobs;

• DDM and PanDA servers for access to dataset and production/analysis task information;

• The AMI and COMA front-end servers for access to metadata.

As the amount of data kept in Oracle databases increased with time and with the complexity of the database applications through LHC Run 1 (2009–2013), it became apparent that some additional technology had to be used to process and summarise these large amounts of data, reserving the Oracle databases for real-time applications. The WLCG Database Technical Evaluation Group, active in 2011–2012, reached the conclusion that out of the (then) existing structured storage tools, Hadoop had a good chance of being useful for LHC experiments [11]. The CERN IT department started providing initially a test Hadoop cluster and later a production service.

The first application that was ported to Hadoop was the data management accounting tool; it extracts daily information from the back-up replica of the ADCR Oracle database (not to overload the main server) and stores it in HDFS (the Hadoop file system). A separate application runs in Hadoop to aggregate and analyse the data and provide summary information used for storage resource management and accounting. A similar application runs now extracting information from the PanDA database and storing it for further analysis in a Cassandra [12] system; more details are available in [13].

Logs of jobs run on the Grid provide a lot of useful information on dataset access pattern and frequencies as a function of time. This information is now extracted from the PanDA database and transferred to a dedicated Hadoop cluster where it is analysed to find ways to optimise data structures in event records, as well as dataset distribution and replication (or deletion) on Grid sites.

The first ATLAS application that was developed directly for the CERN Hadoop cluster is the EventIndex [4]. It contains a logical record for each event (including Run 1) in each processing stage; the event information consists of its identifier (run and event number), trigger selection chain, data format and the pointer to the logical file that contains it and the internal pointer to the event itself within the file. With this information it is possible to retrieve the events of interest, just having the run and event numbers, with the automatic help of Rucio to get the file and PanDA to run a Grid job to extract the wanted events and return them to the analyst. The events are stored in Hadoop "mapfiles", which are in turn catalogued internally in HBase; more details can be found in [14].

## EVOLUTION TOWARDS LHC RUN 3

The continued usage of Oracle databases is good for the time being but we are warned by CERN that the licence conditions may change at the end of the current agreement (2018), so some diversification may be needed. Some types of data and metadata fit naturally into the relational database model, other data much less, like, for example, the large amounts of useful but static data on DDM datasets (accounting), completed PanDA production and analysis tasks, and event metadata.

Modern structured storage systems (”NoSQL databases”) are already in use in addition to Oracle to store large chunks of read-only data:

- Hadoop is in operation for Rucio accounting and EventIndex [14];
- Hadoop is also under study for Distributed Computing monitoring and accounting;
- Cassandra is under study for PanDA production and analysis tasks archive [13].

As long as access to the data is always done through an interface server, the users won’t actually see the underlying storage technology; the back-end technology only has to be chosen considering the performance, cost of the infrastructure, support for it from CERN-IT and ease of use for the developers of the applications.

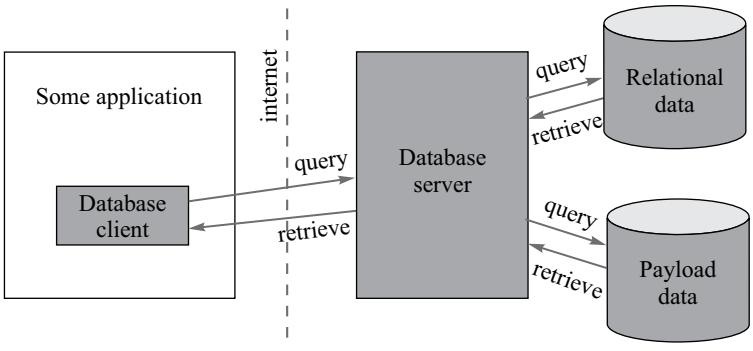
A modern generic architecture to design new applications consists of four main components:

- A data store to hold all data records;
- A relational (SQL) database to hold all transactional data and metadata about the data records, including the pointers to the actual data in the data store;
- A server that runs all necessary code to store, update and retrieve all data;
- A very thin client that contacts the server to retrieve the needed data and unpacks them for the application that requests the data.

The figure shows the relations between these building blocks.

In this architecture very little data is kept in the relational database, which at this point can be implemented in Oracle or any other SQL database. Keeping only the “live” data in Oracle now means that at some point in the future we could change technology for the SQL database without too much trouble (only in case of need of course).

This design schema is applied for the first time for the new conditions database that is currently under development together with CMS [15] and should be ready for LHC Run 3.




Schematic view of a genetic database storage and client-server architecture

This project just started but is building on the experience collected by both large LHC experiments during Run 1; so far the work packages have been defined and studies have started to identify the best technology for each component. As explained previously, the current choice for the relational database, which will hold the metadata associated to each data block (data type, interval of validity, version and pointer to the data block in the payload store), is Oracle, but keeping compatibility with other SQL databases such as PostgreSQL [16] and SQLite [6]. For this small amount of data no particular optimisation is needed in Oracle, so it is possible to maintain the compatibility with other SQL databases.

The payload store is under study. One option is to keep everything in Oracle as BLOBs or CLOBs, which have to be packed and decoded by the client code. Another possibility is to use a NoSQL store, keeping data either packed or unpacked, or maybe both for ease of access by different clients. Unpacked data stored as key-value pairs can then be searched or analysed to extract trends or other information.

The server has to provide write access to people and processes that add data to the system, and read access through a web service to the clients that request the data. All the client has to do is to communicate with the server, normally through a web proxy that has a cache, sending a simple "curl http://. . . " request specifying the data type to retrieve, with time interval and version; the response will contain the payload. The server will interpret the request, get the pointer to the payload from the SQL database, retrieve the payload and send it back to the client. Several caches are built into the system, in the server, at each processing site (site Squid [17]) and in the client, so that repeated queries for the same data will be satisfied by the first process that receives them and has the relevant data in the cache, in the same way as the existing Frontier system [6].

## CONCLUSIONS

ATLAS has a large number of database applications that use both SQL (Oracle) and NoSQL (Hadoop and Cassandra) technologies. The present infrastructure fundamentally works: data can be inserted and accessed fast and people can promptly use them to analyse the event data. The system design and applications are continuously adapted to the evolution of computing technologies.

**Acknowledgements.** This work would not have been possible without the continuous support of the managements of the ATLAS Collaboration, and the financial support of the participating institutions and funding agencies.

## REFERENCES

1. *ATLAS Collab.* The ATLAS Experiment at the CERN Large Hadron Collider // JINST. 2008. V. 3. P. S08003; doi:10.1088/1748-0221/3/08/S08003.
2. *Garonne V. on behalf of the ATLAS Collab.* Rucio — The Next Generation of Large Scale Distributed System for ATLAS Data Management // J. Phys. Conf. Ser. 2014. V. 513. P. 042021; doi:10.1088/1742-6596/513/4/042021.
3. *Maeno T. on behalf of the ATLAS Collab.* Evolution of the ATLAS PanDA Workload Management System for Exascale Computational Science // J. Phys. Conf. Ser. 2014. V. 513. P. 032062; doi:10.1088/1742-6596/513/3/032062.

4. *Barberis D. on behalf of the ATLAS Collab.* The ATLAS EventIndex: Architecture, Design Choices, Deployment and First Operation Experience // J. Phys. Conf. Ser. 2015. V. 664. P. 042003; doi:10.1088/1742-6596/664/4/042003.
5. *Valassi A.* 2014. CORAL and COOL during the LHC Long Shutdown // J. Phys. Conf. Ser. 2014. V. 513. P. 042045; doi:10.1088/1742-6596/513/4/042045.
6. SQLite. https://www.sqlite.org.
7. *Barberis D. on behalf of the ATLAS Collab.* Evolution of Grid-Wide Access to Database Resident Information in ATLAS Using Frontier // J. Phys. Conf. Ser. 2012. V. 396. P. 052025; doi:10.1088/1742-6596/396/5/052025.
8. *Albrand S. on behalf of the ATLAS Collab.* The ATLAS Metadata Interface // J. Phys. Conf. Ser. 2010. V. 219. P. 042030; doi:10.1088/1742-6596/219/4/042030.
9. *Gallas E. J. on behalf of the ATLAS Collab.* Conditions and Configuration Metadata for the ATLAS Experiment // J. Phys. Conf. Ser. 2012. V. 396. P. 052033; doi:10.1088/1742-6596/396/5/052033.
10. Hadoop. https://hadoop.apache.org.
11. *Barberis D.* Report of the WLCG Database Technical Evolution Group // CERN Report. 2012.
12. Cassandra. http://cassandra.apache.org.
13. *Grigorieva M.* The Development of Hybrid Metadata Storage for PanDA Workload Management System // Phys. Part. Nucl. Lett. 2016. V. 13, No. 4 (in press).
14. *Favareto A. on behalf of the ATLAS Collab.* Use of the Hadoop Structured Storage Tools for the ATLAS EventIndex Event Catalogue // Part. Nucl., Lett. 2016. V. 13, No. 5(203). P. 968.
15. *Barberis D. on behalf of the ATLAS Collab.* Designing a Future Conditions Database Based on LHC Experience // Proc. of the 21st Intern. Conf. on Computing in High Energy and Nuclear Physics (CHEP2015), Okinawa, Japan, April 2015. J. Phys. Conf. Ser. (submitted).
16. PostgreSQL. http://www.postgresql.org.
17. Squid. http://www.squid-cache.org.