EI3 – THE ATLAS EVENTINDEX FOR LHC RUN 3

F.V. Prokoshin[1 a], I.N. Aleksandrov[1], E.I. Alexandrov[1], Z. Baranowski[2], D. Barberis[3], G. Dimitrov[4], A. Fernandez Casani[5], E. Gallas[6], C. Garcia Montoro[5], S. Gonzalez de la Hoz[5], J. Hrivnac[7], A.I. Kazymov[1], M.A. Mineev[1], G. Rybkin[7], J. Sanchez[5], J. Salt[5], M. Villaplana Perez[8]

[1] *Joint Institute for Nuclear Research, 6 Joliot-Curie St., Dubna, Moscow Region, 141980, Russia*

[2] *CERN-IT, Geneva, Switzerland*

[3] *INFN Genova and Universita' di Genova, Dipartimento di Fisica, Genova, Italy*

[4] *CERN-ATLAS, Geneva, Switzerland*

[5] *Instituto de Fisica Corpuscular (IFIC), Centro Mixto Universidad de Valencia - CSIC, Valencia, Spain*

[6] *Department of Physics, Oxford University, Oxford, United Kingdom*

[7] *LAL, Universite Paris-Sud, CNRS/IN2P3, Universite Paris-Saclay, Orsay, France*

[8] *Department of Physics, University of Alberta, Edmonton AB, Canada*

E-mail: [a] fedor.prokoshin@cern.ch

Since 2015 the ATLAS Event Index provides a good and reliable service for the initial use cases (mainly event picking) and several additional ones, such as production consistency checks, duplicate event detection and measurements of the overlaps of trigger chains and derivation datasets. LHC Run 3 will meet increased data-taking and simulation production rates, which the current infrastructure would still cope but may be pushed to its limits by the end of Run 3. This talk describes a new implementation of the front- and back-end services that will be able to provide at least the same functionality as the current one for increased data ingestion and search rates and with increasing volumes of stored data. It is based on a set of HBase tables, with schemas derived from the current Oracle implementation, coupled to Apache Phoenix for data access; in this way the possibility of SQL as well as NoSQL data access will be added to the advantages of a BigData based storage system, allowing re-use of most of the existing code for metadata integration.

Keywords: Event Index, ATLAS computing, Database, BigData

Fedor Prokoshin, Igor Aleksandrov, Evgeny Alexandrov, Zbigniew Baranowski, Dario Barberis, Gancho Dimitrov, Alvaro Fernandez Casani, Elizabeth Gallas, Carlos Garcia Montoro, Santiago Gonzalez de la Hoz, Julius Hrivnac, Andrei Kazymov, Mikhail Mineev, Grigori Rybkin, Javier Sánchez, José Salt, Miguel Villaplana Perez

1. Introduction

The ATLAS Experiment [1] produces large amount of data: several billion events per year. In addition, similar number of simulated Monte Carlo (MC) events is being produced. This data is stored on the hundreds of sites all around the world. A database containing references to the events is necessary in order to efficiently access them from the distributed data storage. The ATLAS EventIndex [2, 3] provides a way to collect and store event information using modern technologies. It also provides various tools to access this information through command line, GUI and RESTful API interfaces.

An infrastructure was created that allows fast and efficient selection of events of interest from the billions of events recorded, based on various criteria. The EventIndex provides an indexing system that points to these events in millions of files spread through a worldwide distributed computing system. It automatically detects and indexes new data that was produced in this system from events collected from the detector or simulated using Monte-Carlo technique.

2. The ATLAS EventIndex

The ATLAS EventIndex is a system designed to be a complete catalog of ATLAS events, experimental and simulated data.

2.1 ATLAS Events

Event is the basic unit of ATLAS data. Each event contains result of a single triggered interactions, plus eventually piled-up interaction. Event information includes signals from the detector, reconstructed particles with their parameters and trigger decisions.

Event is uniquely identified by the run number and the event number. Event information is stored in many instances, that are spread among the hundreds of GRID sites. Event information has different formats and contents to fit analyses needs.

Each event record contains the following fields:

- Event identifiers: run and event number, trigger stream, luminosity block, bunch crossing ID (BCID).
- Trigger decisions: Trigger masks for each trigger level, and decoded trigger chains, specifying trigger condition passed.
- References to the events at each processing stage in all permanent files generated by central productions. They can be used to retrieve specific events of interest (event picking).

2.2 ATLAS Datasets

ATLAS event data is written in files that are organized in datasets. Datasets can have different format depending on the processing stage: Detector data is first written in the RAW format, then Analysis Object Data (AOD, i.e. the reconstruction output) datasets are produced after reconstruction. Derived AOD datasets (DAOD, i.e. selected events with reduced information for specific analyses) are produced for use in the specific analyses. In addition to datasets produced for experimental data, MC datasets are produced on the GRID, to be used in various analyses and system calibration. The EVNT datasets contain particles information and RDO datasets contain simulated detector signals. There are various versions of the datasets originating from the same RAW data or simulated events, with different reconstruction settings and software version. Datasets are being reprocessed roughly yearly.

2.3 Use cases

Originally the EventIndex was intended to be used mostly for Event Picking: user may ask for event in specific format and processing version. Several use cases were added based on operation experience and user requests. A trigger information stored in the event record allows to count and select events based on trigger decisions. Trigger chains overlaps are also being counted for trigger tables optimization. The EventIndex appears to be useful for production completeness and consistency

checks, looking for data corruption, missing and/or duplicated events, derivation overlaps counting. The EventIndex is used for dataset browsing: finding datasets of interest, dataset report and inspection.

Summary of use cases can be found on the project page:
https://twiki.cern.ch/twiki/bin/view/AtlasComputing/EventIndexUseCases

2.4 The EventIndex architecture

Figure 1 shows the partitioned architecture of the EventIndex.

The Data Production system extracts the EventIndex information from ATLAS files produced at CERN or on the GRID [4]. The process starts automatically as soon as a new dataset is completed. The information about new datasets on the GRID is obtained from the ATLAS Metadata Interface database (AMI, [5, 6]). These indexing jobs also provide a data integrity test, as they are the first to run on new data files. All new datasets containing events in the AOD format are indexed by default, while other datasets derived from the AOD are indexed on demand only. Since the extracted metadata contains also the references to the corresponding raw data, it is always possible to extract events in RAW data format too. For simulated data, all event generator files are indexed as well.

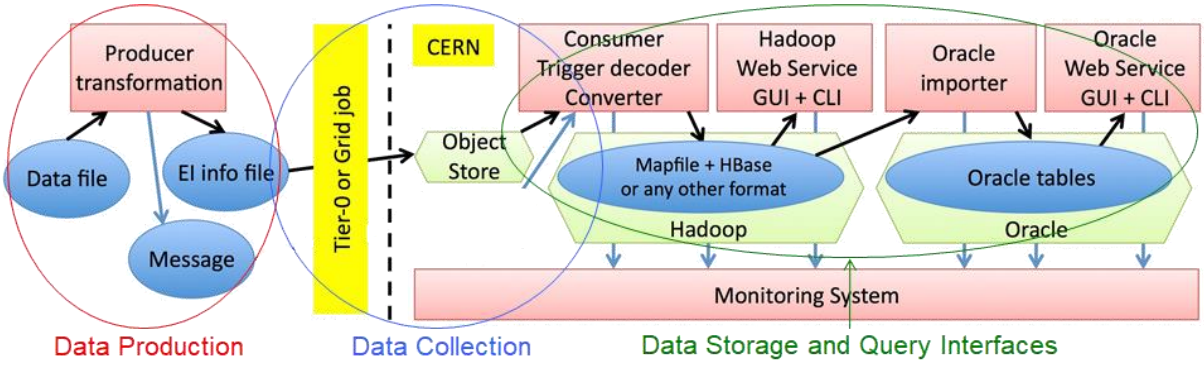

Figure 1. The EventIndex architecture

The extracted event metadata is packed into small files and transferred to the CERN Object Store by the Data Collection system [7, 8], that runs on servers at CERN. A Data Collection Supervisor controls the transfers and validates indexed data. When a new dataset has been indexed, the Supervisor informs the Consumer services, which then fetch data from the Object Store and store it in the Hadoop [9,10] cluster at CERN. The system is now in routine operation, with low number of failures mostly originated from site problems or corrupted files.

Data Storage and Query system provide permanent storage for the EventIndex data, fast access for the most common queries, and reasonable-time response for complex queries. It uses Hadoop as a baseline storage technology. The full information is stored in compressed MapFile format [11]. An internal catalogue in HBase [12] keeps track of the status of each dataset and holds dataset metadata. All event records are also added to a HBase table, that is used for fast event lookup for the event picking use case. The CLI, RESTful API and GUI interfaces are available for data inspection, search and retrieval. As of October 2019, the system holds 24 TB of data for experimental data and 11 TB for simulated ones.

Reduced information from experimental data records (only event identification information and pointers to event locations, no trigger information) are also copied to an Oracle database. It is related with other ATLAS databases stored in the same Oracle cluster, this is especially useful for checking of the dataset processing completeness and duplicates detection. The Oracle is also much faster in event lookup operations, if the data schema has been well designed [13]. It is also being used for easier calculation of dataset overlaps. Information on 77 k datasets with185 billion event records is stored there.

A monitoring system [14, 15] keeps tracks for health of the system components and the data flow, providing a visualization of the system status and of the stored data volume. Current

implementation of the monitoring system uses InfluxDB [16] to store the data and Grafana [17] to display information.

3. The EventIndex evolution towards The LHC Run 3

The current design of the EventIndex, based on Hadoop and implemented using MapFiles and HBase, has been in continuous operation since 2015 with satisfactory results. Nevertheless, the current implementation of the EventIndex started showing scalability issues due to new use cases and the increasing amount of stored data (the event rate increased steadily throughout Run 2 beyond the initial expectation). The fast data querying for experimental events based on Oracle is no longer sufficient to cover all requests. Therefore, an R&D programme was started to explore new technologies that would allow the EventIndex to include new functionalities and to keep up with the future demanding event rates.

3.1 The next generation EventIndex

The EventIndex technologies must evolve to future demanding rates of data. Currently all ATLAS processes produce ~30 billion events records/year (up to 350 Hz on average). This is update rate through the whole system, all years, experimental and simulated data. The EventIndex reads 8M files/day and produces 3M files. In the future due to expected trigger rates, there is need to scale for next ATLAS runs. For Run 3 (2021-2023) an expected increase is at least half an order of magnitude: 35 B new experimental events/year and 100 B new MC events/year. It is expected an order of magnitude increase for Run 4 (2026-2029): 100 B new experimental events and 300 B new MC events per year. In addition, sum up replicas and reprocessing.

An evolution of the EventIndex concepts is being planned. Currently the same event across each processing step (RAW, ESD, AOD, DAOD, NTUP) is physically stored at different Hadoop HDFS files. In the future EventIndex there will be one logical record per event, that will contain event identification, inmutable information (trigger, lumiblock, ...), and for each processing step there will be additional information: link to algorithm (processing task configuration), pointer(s) to output(s) and possibly flags for offline selections (derivations).

The EventIndex evolution includes support of virtual datasets. A virtual dataset is a logical collection of events, created either explicitly (giving a collection of Event Ids) or implicitly (selection based on some other collection or event attributes).

3.2 New Use Cases

A number of new use cases were proposed:

- Massive event picking: Selection of large number of events, touching a large fraction (or all) of the files in a dataset. Possible need of the dedicated service, especially if input is on tape (RAW data).
- Adding "offline trigger" information: Store results of selections that can be used to form derived datasets. This requires the ability to add info to a part of event record.
- Using offline trigger information: select events using online and offline trigger information to build a "Virtual dataset". Use massive event picking to physically retrieve events belonging to a virtual dataset (probably in the AOD format, but also RAW format in case of a small number of events) and continue the analysis with more info on reduced size datasets. This is useful if selecting <1% of the events.
- Partial processing for production tests. May skip some input checks and then assign a finite lifetime to the information (delete once the test is done).

3.3 EI Evolution: SQL on HBase

Apache HBase is the Hadoop database, a distributed, scalable, big data store. It is open-source, distributed, versioned, non-relational database modeled after the Google BigTable paper. HBase was built on top of HDFS and provides fast record lookups (and updates) for large tables.

HBase organizes data into tables. Tables have rows and columns, which store values (like a spreadsheet). Rows are identified uniquely by their row key. Each row can have a different schema. Data within a row is grouped by column family, that must be defined up front and not easily modified.

HBase belongs to noSQL database family. NoSQL databases allow to deal with scalability problems that relational databases are traditionally suffering. If data model is trivial - simple key-value store could satisfy it. On the other hand, SQL/structured schemas provide their own advantages: structured data are easy to understand and maintain, and standard declarative query logic is 'optimized' for complex queries.

Various possibilities for SQL on HBase were considered: Apache Impala, Hive (handling of a row key mapping should be on the application side) and Spark (mainly for batch jobs). A series of tests were performed with prototype based on Apache Kudu [18, 19]. Finally, the Apache Phoenix [20] was chosen as the most promising platform for use in the new EventIndex. It creates SQL layer on top of HBase, providing structured schema of the tables instead of schemaless freeride, mapping of columns to HBase cells and serialization of data types to bytes. It also has SQL planner and optimizer with built-in HBase related optimizations, server-side (optimized) executions and provides convenient access via JDBC (Java DB Connector).

The Apache Phoenix provides online transaction processing and operational analytics for HBase through SQL. It takes SQL query, compiles it into a series of HBase scans. It allows direct use of the HBase API, along with coprocessors and custom filters and produces regular JDBC result sets. The HBase RowKey design has to be adapted to the Phoenix's types and sizes, losing "some" performance. The Apache Phoenix allows using of RowKey fields in queries, but they are stored as one entity in HBase.

Several tests has been performed: loading the ATLAS EventIndex data to HBase via the Phoenix and Phoenix queries on loaded data. Results are encouraging: single event picking in 30 ms, l full dataset queries in 6-10 seconds. Some basic functions are ready, and further work on performance and user interfaces is ongoing.

4. Conclusion

The EventIndex project started in 2012 at the end of the LHC Run 1 driven by the need of having a functional event picking system for ATLAS data. The data storage and search technology selected in the first phase of the project (Hadoop MapFiles and HBase, in 2013-2014) was the most advanced available at that time in the fast-growing field of Big Data and indeed after a couple of initial hiccups it proved reliable and performed satisfactorily. Part of the data are replicated also to Oracle for faster access but mainly to have a uniform environment between event and dataset metadata.

Nevertheless, the current implementation of the EventIndex started showing scalability issues as the amount of stored data increases: slower queries, high storage usage (now eased by compression). Also, significant increase in the data rates expected in the future LHC runs demands transition to a new technology

Phoenix queries and HBase new event table prototypes have been tested and show encouraging results. There is table schema candidate, basic functionality is ready, working towards improved performance and better interfaces. Need to keep testing with more data and get performance metrics. The plan is to have the new system operational by the middle of 2020 in parallel with the old one and phase out the old system at the end of 2020 (well in advance of the start of LHC Run 3).

References

[1] ATLAS Collaboration. The ATLAS Experiment at the CERN Large Hadron Collider // Journal of Instrumentation, Volume 3, Issue 08, pp. S08003 (2008) - DOI: 10.1088/1748-0221/3/08/S08003.

[2] Barberis D. et al. The ATLAS EventIndex: an event catalogue for experiments collecting large amounts of data // Journal of Physics: Conference Series, Volume 513, Issue 4, article id. 042002 (2014) - DOI: 10.1088/1742-6596/513/4/042002.

[3] Barberis D. et al. The ATLAS EventIndex: architecture, design choices, deployment and first operation experience // Journal of Physics: Conference Series, Volume 664, Issue 4, article id. 042003 (2015) - DOI: 10.1088/1742-6596/664/4/042003.

[4] Bird I. et al. LHC Computing Grid. Technical Design Report // CERN-LHCC-2005-024 LCG-TDR-001. - Geneva : CERN, 2005. - 149 p.
Available at: https://cds.cern.ch/record/840543/files/lhcc-2005-024.pdf (accessed 19.11.2019).

[5] Albrand S., Fulachier J. and Lambert F. The ATLAS metadata interface // Journal of Physics: Conference Series, Volume 219, Issue 4, article id. 042030 (2010) - DOI:10.1088/1742-6596/219/4/042030.

[6] Fulachier J., Odier J. and Lambert F. ATLAS Metadata Interface (AMI), a generic metadata framework // Journal of Physics: Conference Series, Volume 898, Issue 6, article id. 062001 (2017) - DOI: 10.1088/1742-6596/898/6/062001.

[7] Sanchez J., Fernandez Casani J., González de la Hoz S. on behalf of the ATLAS Collaboration. Distributed Data Collection for the ATLAS EventIndex, // Journal of Physics: Conference Series, Volume 664, Issue 4, article id. 042046 (2015) - DOI: 10.1088/1742-6596/664/4/042046.

[8] Fernandez Casani A. et al. Distributed Data Collection for the Next Generation ATLAS EventIndex Project // The European Physical Journal Conferences Volume 214, article id. 04010 (2019) - DOI: 10.1051/epjconf/201921404010.

[9] K Shvachko et al. The Hadoop Distributed File System, // Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST), MSST'10, pages 1-10 - DOI: 10.1109/MSST.2010.5496972.

[10] Apache Hadoop. Available at http://hadoop.apache.org (accessed 19.11.2019).

[11] Favareto A. et al. Use of the Hadoop structured storage tools for the ATLAS EventIndex event catalogue // Physics of Particles and Nuclei Letters, September 2016, Volume 13, Issue 5, pp 621–624 - DOI:10.1134/S1547477116050198.

[12] Apache HBase. Available at: https://hbase.apache.org (accessed 19.11.2019).

[13] Gallas E. et al. An Oracle-based EventIndex for ATLAS // Journal of Physics: Conference Series, Volume 898, Issue 4, article id. 042033 (2017) - DOI: 10.1088/1742-6596/898/4/042033.

[14] Barberis D. et al. ATLAS EventIndex monitoring system using the Kibana analytics and visualization platform // Journal of Physics: Conference Series, Volume 762, Issue 1, article id. 012004 (2016) - DOI: 10.1088/1742-6596/762/1/012004.

[15] Alexandrov E. et al. BigData tools for the monitoring of the ATLAS EventIndex // Proceedings of the VIII International Conference "Distributed Computing and Grid-technologies in Science and Education" (GRID 2018), Dubna, Moscow region, Russia, September 10 -14, 2018, Available at: http://ceur-ws.org/Vol-2267/91-94-paper-15.pdf (accessed 19.11.2019).

[16] InfluxDB: Purpose-Built Open Source Time Series Database. Available at: https://www.influxdata.com (accessed 19.11.2019).

[17] Grafana: The open observability platform. Available at: https://grafana.com (accessed 19.11.2019).

[18] Baranowski Z. et al. A study of data representation in Hadoop to optimize data storage and search performance for the ATLAS EventIndex // Journal of Physics: Conference Series, Volume 898, Issue 6, article id. 062020 (2016) - DOI: /10.1088/1742-6596/898/6/062020.

[19] Fernandez Casani A. et al. A prototype for the evolution of ATLAS EventIndex based on Apache Kudu storage // The European Physical Journal Conferences Volume 214, article id. 04057 (2019) - DOI: 10.1051/epjconf/201921404057.

[20] Apache Phoenix: https://phoenix.apache.org (accessed 19.11.2019).