The ATLAS EventIndex: data flow and inclusion of other metadata

D. Barberis¹, S.E. Cárdenas Zárate², A. Favareto³, A. Fernandez Casani⁴, E.J. Gallas⁵, C. García Montoro⁶, S. Gonzalez de la Hoz⁷, J. Hrivnac⁸, D. Malon⁹, F. Prokoshin⁵, J. Salt⁴, J. Sanchez⁴, R. Toebbicke⁸, R. Yuan⁶

on behalf of the ATLAS Collaboration

¹ Università di Genova and INFN, Genova, Italy
² Universidad Técnica Federico Santa María, Valparaiso, Chile
³ Università degli Studi and INFN, Genova, Italy
⁴ Instituto de Física Corpuscular, Valencia, Spain
⁵ Department of Physics, Oxford University, Oxford, United Kingdom
⁶ Laboratoire de l’Accélérateur Linéaire, Orsay, France
⁷ Argonne National Laboratory, Argonne, USA
⁸ CERN, Europe

*Corresponding author: Fedor.Prokoshin@cern.ch

Abstract. The ATLAS EventIndex is the catalogue of the event-related metadata for the information collected from the ATLAS detector. The basic unit of this information is the event record, containing the event identification parameters, pointers to the files containing this event as well as trigger decision information. The main use case for the EventIndex is event picking, as well as data consistency checks for large production campaigns. The EventIndex employs the Hadoop platform for data storage and handling, as well as a messaging system for the collection of information. The information for the EventIndex is collected both at Tier-0, when the data are first produced, and from the Grid, when various types of derived data are produced. The EventIndex uses various types of auxiliary information from other ATLAS sources for data collection and processing: trigger tables from the condition metadata database (COMA), dataset information from the data catalogue AMI and the Rucio data management system and information on production jobs from the ATLAS production system. The ATLAS production system is also used for the collection of event information from the Grid jobs. EventIndex developments started in 2012 and in the middle of 2015 the system was commissioned and started collecting event metadata, as a part of ATLAS Distributed Computing operations.

1. Introduction

The ATLAS experiment [1] runs a complex system that employs detectors with millions of readout channels, and gathers billions of events in data taking periods that may extend to several years. It produces several petabytes of data per year, with different physics conditions and data formats. Different types of metadata (data about data) make possible the effective usage of this tremendous amount of information. Dataset metadata deal with data storage organization. Run-wise metadata deal with slowly changing parameters and links to the data-taking conditions. It is useful to have a
catalogue containing information about the basic properties of event data and references to their storage locations, and also providing a means of rapid search through the event metadata.

The EventIndex [2] is a system designed to be a complete catalogue of ATLAS events, real and simulated data at all processing stages. However, experience obtained in the process of commissioning and first period of use, input from the physics groups and performance constraints resulted in the exclusion of some of the short-living data (like derived data for workgroups - DAODs) from being processed by the EventIndex. We also try to avoid repeated processing of the same information (like trigger decisions) from different production stages and software tags.

2. The EventIndex Data
The event record contains event identifiers: (run number, event number, trigger stream, luminosity block), trigger decisions for different levels and references (GUID + internal pointer) to the events at different processing stages in permanent files, generated by central production. For the Level-1 trigger we store the final trigger decision and also additional information on events rejected by veto or prescaling. For High-Level Trigger in addition to “physics” triggers, “pass-through” and “resurrected” triggers are also stored.

The event record has size of approximately 350 bytes per event. This gives about 350 gigabytes of uncompressed information per billion \((10^9)\) events, multiplied by number of processings and formats. The real amount of storage space needed depends on the chosen technology, replication multiplicity and internal data format and indexing.

There are a number of use cases for the EventIndex:
- Event picking: a request for reference to the specific event in given format and processing type.
- Event counting: counting the number of times that a specific trigger decisions were passed. The information is collected by the EventIndex using trigger tables from the Condition Metadata Store (COMA) [3]. Results are provided to users also by COMA.
- Duplicate finding: detecting duplicated data within the scope of a dataset or physics collection.

The information is supplied to the users through the ATLAS Metadata Interface (AMI) [4].

Different data types [5] are processed by the EventIndex and may be treated differently. Top priority for the EventIndex are Analysis Oriented Data (AOD), both for real and simulated (MC) data. The indexing of AODs provides also indexing of RAW data through the provenance information that is picked up. MC datasets containing generated events (EVNT) are indexed for duplicate event detection. Routine indexing of derived data (DAODs) is currently not performed, because of a short useful lifetime and large amount of data (10x AODs). Still the capability to index DAOD’s is preserved; a very small subsample is indexed for that purpose. We avoid repeated processing of the trigger data for the same events, so we have trigger processing enabled only for real data AODs from Tier-0 (including spill-over) and for MC AODs. The trigger processing is disabled for real data AOD from reprocessing and for DAODs.

3. EventIndex structure and dataflow
The EventIndex infrastructure consists of a few autonomous components, providing the collection and processing of EventIndex data. The modular approach allowed sharing the developments between different institutions. The main components and dataflow are presented in Figure 1.

The information is collected by event processing tasks (“producers”) which can run at Tier-0 (initial reconstruction at CERN) or on Grid sites (downstream processing). Event metadata collected by the producers is sent via the ActiveMQ [6] message broker to the Hadoop [7] store at CERN. At CERN messages are read by Consumer program and stored as Hadoop Mapfile objects. Data are validated for completeness and duplicated events, and decoding of the trigger information performed using trigger tables from COMA. Users can access EventIndex information through the query services: command line interface and web services with graphical and text/plain output suitable for the ATLAS PanDA [8] (Production and Distributed Analysis) system [9].
Datasets produced at the Tier-0 computing center at CERN are indexed by tasks running EventIndex Producer transformations.

On the Grid the EventIndex uses ProdSys2 [10], a second generation ATLAS production system, to index datasets as soon as they appear in the AMI catalogue and are marked as valid and completed. Information on data produced/modified daily on the Grid is gathered by automatic scripts by querying the AMI metadata catalog for VALID datasets from mc_15 and data_15 projects, and soon mc_16 and data_16 too. Then a catalogue in the ATLAS Distributed Data Management system [11] (Rucio) is queried for the identifiers of the datasets to feed the EventIndex processing jobs. Collected datasets are attached to the special “technical containers” by cron script, to be picked up by the EventIndex production tasks. There is one technical container for each type of data to be indexed and trigger information processing setting. A new set of technical containers is being created when a new version of the EventIndex producer appears. For each dataset that appears in the technical container, ProdSys2 creates a Grid task that extracts event data and sends information to the “consumers” running on servers at CERN [12].

This operation is done in the following steps (see Figure 2):

- The producer process extracts EventIndex information from each event in a file and fills an intermediate file on the worker node. Event data are then read from the intermediate file, trigger masks are compressed, duplicate information is dropped and records are encoded into JSON to build the text message.
- EventIndex uses the Streaming Text Oriented Messaging Protocol [13] (STOMP) to transport information between producer and consumers. The message size is set small (1-10 kB).
- Messages are received by the ActiveMQ message broker and queued for the consumers.
- Messages are finally read by EventIndex consumers, decoded and organized into Hadoop MapFile objects.
- Consumers also run a validation step, verifying that the EventIndex for a collection (dataset) contains all the files (GUIDs) and all the events, and detecting multiple processing of the same file and job failures.

The consumers send back statistical information, containing the counts of messages, files and events to the broker to report the status. This is used, in particular, by the monitoring system.

Measuring the system performance for sending real events shows that it is able to process 200k events per second or 60 MB of data per second for the typical configuration: 1 broker, 6 producers, 4 consumers and 50k events/job.

The Hadoop data management platform was chosen as the storage technology for the EventIndex. This decision was driven by the following reasons:

- This platform is provided and supported by CERN-IT
- The DDM (Distributed Data Management) project also uses Hadoop
- There are plenty of tools to organize the data, index them internally and search them
- It showed a satisfactory performance in a prototype populated with a year of ATLAS data
Data are stored as Hadoop mapfiles (files based key-value maps) in HDFS (Hadoop File System). The trigger information receives additional processing: trigger decisions are decoded using tables from COMA and the resulting lists of trigger chain names are added to the mapfiles.

Data are internally catalogued by HBase [14], creating metadata about HDFS files as keyed indexes, thus enhancing the performance: searches based on keys give almost immediate results (seconds). Complex searches instead use MapReduce [15] and require 1-2 minutes for typical event collections.

Since the beginning of 2015 we store approximately 50 TB of EventIndex data: 46 TB corresponding to the real data and 4 TB to the MC data. We currently also keep the consumer data, that takes approximately the same amount of space and will eventually be removed.

Users can access EventIndex information through the query services:

- Command line interface, available on the Hadoop cluster and remotely from AFS, CVMFS and a downloadable tar file;
- Web services with graphical and text/plain output suitable for the Panda system.

Remote access goes via a proxy server to a Tomcat server at the Hadoop cluster to assure the Hadoop cluster isolation from the outside environment and fallback & load-balancing.

Searches are executed in two steps. First, the HBase catalogue finds the mapfiles to search for. The execution time depends on the search criteria and may take from less than 1 s (exact key-based search) to 20 s (full scan search). Then Hadoop executes the search in the selected files. It can be a key-search (almost immediate) if searching directly on RunNumber-EventNumber or MapReduce jobs if searching with other attributes, like trigger decisions.

In the Table 1 the event lookup results for the search for events from a list are presented. Events are specified by run-event pairs, and in this case only AODs were searched. Requests were done from a remote machine for general ATLAS use, not from the restricted EventIndex cluster.

<table>
<thead>
<tr>
<th>Number of events</th>
<th>Found GUIDs</th>
<th>Searched runs</th>
<th>Searched tag Files</th>
<th>Time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>10</td>
<td>39</td>
<td>1</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>50</td>
<td>185</td>
<td>1</td>
<td>6</td>
<td>3</td>
</tr>
<tr>
<td>100</td>
<td>381</td>
<td>2</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>500</td>
<td>1803</td>
<td>9</td>
<td>54</td>
<td>11</td>
</tr>
<tr>
<td>1000</td>
<td>3808</td>
<td>22</td>
<td>146</td>
<td>31</td>
</tr>
<tr>
<td>5000</td>
<td>9957</td>
<td>50</td>
<td>319</td>
<td>120</td>
</tr>
<tr>
<td>8879</td>
<td>24179</td>
<td>66</td>
<td>426</td>
<td>193</td>
</tr>
</tbody>
</table>

Table 1 Event lookup benchmark results

It should be noted that HBase, Hadoop and Tomcat keeps some results in cache/memory so the second (overlapping) search is usually faster than the first one.
4. Data exchange with other systems

The EventIndex uses information from other systems in order to find and correctly process data. In particular, trigger information is decoded using information from COMA [16]. The trigger decoding data flow is presented in Figure 3.

Trigger decisions are stored in data files as trigger masks. If a bit of the mask is set, this means that one of the sets of trigger conditions (trigger chains) has been passed in this event. The position of this bit in mask (chain counter) indicates which chain has been passed. Depending on the trigger configuration, the same chain counter may correspond to a different chain name. The relation is uniquely defined by the trigger table corresponding to the trigger key, or SMK. Trigger masks from event record are decoded, and chain counters are converted to chain names. The list of trigger chain names, obtained after decoding are then stored in updated event records.

The EventIndex uses trigger tables replicated from COMA to HBase for the trigger decision decoding. The trigger tables use SMK as key and ChainName:ChainCounter pairs as columns. Additionally there is a Run:SMK table for verification. As of the beginning of 2016 the trigger tables contain information on 616 SMKs and 10526 runs. Trigger table replication is done by a cron job running daily. Automatic replication is in progress since March 2015.

The EventIndex is a source of data on trigger event counts for COMA. The EventIndex program does the counting on a server within the CERN network providing information through a RESTful service that can be accessed from outside CERN.

We use AMI information for identifying new and changed datasets that have to be processed by the EventIndex. We query AMI through a python interface for a list of complete dataset containers of certain types with last modification time on the previous day and flagged as “valid”. As large datasets in ATLAS can be produced by a number of tasks, containers are made of one or few task ID (TID) datasets that have to be specified as an input for the EventIndex production job. Information on such datasets is obtained ATLAS data management system (Rucio).

The current procedure based on dataset modification times has a number of drawbacks. It is difficult to detect datasets that became corrupted or obsolete. Dataset modification times may change for reasons not related to their contents, requiring unnecessary re-importing of data. There is no feedback to AMI that may be useful to tag corrupted data based on the result of the EventIndex verification.

It was proposed to introduce a special table in the AMI for information exchange between the two systems. Information on the new, changed and invalid datasets will be placed by AMI to a special field, AMI_DS_STATE, with a timestamp. The EventIndex will put dataset import status into another field, EI_DS_STATE, providing to AMI information on data consistency problems detected by the EventIndex. This machinery is currently in the process of implementation by the AMI Team.

5. Conclusions and outlook
The development of the EventIndex started in 2012 and at the beginning of 2015 all its components were operative. The import of data from the previous years of LHC operation started in February 2015 and now is complete. Run 2 data import started when it became available in summer 2015 and now it flows continuously from the Tier-0 and Grid and is available to users almost in real time.

All major components exist and work:

- Data Collection: Producer transforms run at Tier-0 and on the Grid. Consumers read data from the ActiveMQ servers, validate them and store them to HDFS.
- Storage System: Data organization in Hadoop and indexing in the catalogue is running smoothly, with recent significant stability improvement. Trigger decoding services have been developed, tested and are currently used in data import.
- Query System: Command line and web interface are fully operative and used. We constantly improve its functionality and adding new features based on users’ feedback.
- Monitoring: System level monitoring in the new Kibana [17] environment at CERN is ready and running, providing information for the EventIndex operation team and for ATLAS shifters.

The EventIndex development team continues its work on further improving the system. This includes optimization of the internal data representation, further automation of the data flow, system interconnections and monitoring, and automatic checks of production completeness.

References