Query Service for New ATLAS EventIndex System*

Grigory Rybkin[1,**]
on behalf of the ATLAS Collaboration

[1]Université Paris-Saclay, CNRS/IN2P3, IJCLab, 91405 Orsay, France

Abstract. The ATLAS EventIndex system consists of the catalogue of all events collected, processed or generated by the ATLAS experiment at the CERN LHC accelerator, and all associated software tools. The new system, developed for LHC Run 3, makes use of Apache HBase—the Hadoop database—and Apache Phoenix—an SQL/relational database layer for HBase—to store and access all the event metadata. The Query Service allows searches for and retrieval of information of interest. Its design is based on a server-client architecture with the sides communicating via a RESTful API. The OpenAPI Specification is used to describe the API and generate parts of the source code of the server and client. Based on selection criteria received from the client, the server executes SQL queries against the Phoenix tables, retrieves the data, and returns it to the client. The client—a command line utility following the UNIX/POSIX-conforming program design style—can output results in various formats including plain text and JSON. The server and the client are developed in Java 8, with the use of the Jersey 2 JAX-RS API implementation and the Spring Framework. The service performance is suitable for interactive use with queries for up to 1000 events at a time.

1 Introduction

The ATLAS experiment [1] records several billion particle interactions every year of operation and produces even larger simulated data samples. The ATLAS EventIndex [2] is a Big Data catalogue for all ATLAS events recorded or generated, and processed, and all associated software tools to collect, store and query this information. Each EventIndex record includes summary information on the event itself, and pointers to the files containing the full event. The global catalogue allows searches for and retrieval of specific events for in-depth investigations. EventIndex is an important tool, necessary to the smooth operation of the ATLAS collaboration.

2 New EventIndex

A new major version of EventIndex was developed and put in production for Run 3 of the LHC [3]. The components of the EventIndex system are implemented using Big Data technologies, open-source and free software projects and tools: Apache Hadoop — a framework

**e-mail: Grigori.Rybkine@cern.ch

that allows for distributed processing of large data sets [4], Apache HBase — the Hadoop database [5], Apache Phoenix — an SQL/relational database layer for HBase [6]. The Query Service is a key component of EventIndex.

3 Architecture and Technologies

Query Service's design is based on a server-client architecture in which the server and client communicate via a RESTful API, as shown in Figure 1.

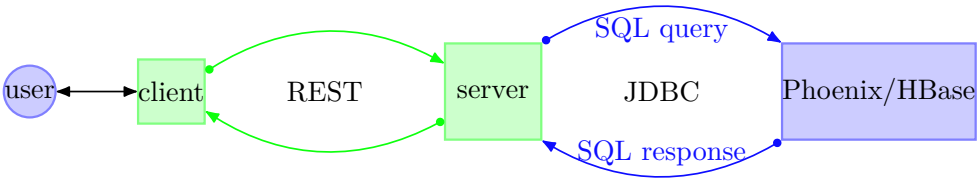

Figure 1. Architecture and Technologies.

The client

1. sends user specified selection criteria to the server,
2. returns the server response to the user.

The server

1. executes SQL queries against the Phoenix tables — EventIndex' database,
2. retrieves the data, and returns it to the client.

The OpenAPI Specification [7] is used to (a) comprehensively describe the API, (b) generate parts of the source code of the server and client.

Both the server and client are written in Java 8, making use of the Jersey 2 JAX-RS API implementation [8].

The server implementation makes use of the Spring Framework [9] providing support for working with SQL databases, in particular, via direct JDBC (Java DataBase Connectivity) access. The JDBC driver is delivered by Phoenix. Configuration is also done with the Spring Framework, using a configuration file, environment variables, and command line options. Parallelism is implemented (a) by adapting the SQL queries, then it is done by Phoenix; (b) by means of Java `Stream`s, then it is done by Java runtime. Thanks to parallelization the service performance is suitable for interactive use with queries for up to 1000 events at a time.

4 Client Command Line Interface

The client Command Line Interface (CLI) comprises a set of command line utilities that follow UNIX/POSIX-conforming program design style [10] and are convenient to use with and by other programs. There are several output representation formats, e.g., plain — lines of space separated fields, for processing with UNIX tools; json — object-oriented representation, to use programmatically; pretty — more human-readable json. Input can also be read directly from the utility standard input.

The convenience features include: (a) two forms for each option, short and long. The long form, giving a human readable name to the option [11], originated in the GNU project [12] of the Free Software Foundation (FSF) [13] and has become common. Most long options are named after fields of ATLAS dataset names, e.g., `--project`, `--data-type`, `--ami-tag`; (b) options and operands (also referred to as ordinary or positional arguments) can appear in any order (be interspersed).

5 Client CLI Utilities

5.1 event-lookup

The so-called "event picking" is the most used functionality of EventIndex. It consists of identifying the files containing events of interest in order to perform in-depth investigations or additional processing. An `event-lookup` utility implements "event picking". Here is an excerpt from its help message:

```
usage: event-lookup [-h] [-v] [-p PROJECT] [-s STREAMNAME] [-S PRODSTEP]
                    [-d DATATYPE] [-a VERSION] [-D GUIDDATATYPE] [-F FILE]
                    [-c plain,name,short,pretty,json] [-o OUTPUT]
                    [event [event ...]]

Search (indexed datasets) for GUIDs  of  files  with events. The files with
found GUIDs may be part of the searched datasets or provenance datasets.
```

An event is specified by a run number and an event number. Additional parameters select the dataset(s) of which the file containing the event is part, e.g., project name, data type. The utility returns the so-called Globally Unique IDentifier(s) [14] or GUIDs of the file(s).

5.1.1 Usage examples

- Look up event number 3613825 in run 348895 (i.e., event 348895:3613825) and output dataset name:

```
  $ event-lookup -c name 348895:3613825
  348895 3613825 4aa369dc-fc28-3444-aab5-c2fdf27a9c69 DAOD_HDBS3↩
↪ data18_13TeV.00348895.physics_Main.deriv.DAOD_HDBS3.f937_m1972_p4928
  348895 3613825 78c6837f-f448-e811-ae3c-44a8420a7621 RAW↩
↪ data18_13TeV.00348895.physics_Main.deriv.DAOD_HDBS3.f937_m1972_p4928
  348895 3613825 0d1c27ff-c5b2-6b46-a0fa-edbdc51c6b77 AOD↩
↪ data18_13TeV.00348895.physics_Main.deriv.DAOD_HDBS3.f937_m1972_p4928
```

  For each line of the output above the third field contains the file GUID, the fourth the file data type, the last the searched dataset name.

- Look up event 348895:3613825, print the result in human readable json format:

```
  $ event-lookup -c pretty 348895:3613825
  [ {
    "eventNumber" : 3613825,
    "guid" : "4aa369dc-fc28-3444-aab5-c2fdf27a9c69",
    "prov" : [ {
      "dataType" : "RAW",
      "guid" : "78c6837f-f448-e811-ae3c-44a8420a7621"
```

```
  }, {
    "dataType" : "AOD",
    "guid" : "0d1c27ff-c5b2-6b46-a0fa-edbdc51c6b77"
  } ],
  "dataset" : {
    "runNumber" : 348895,
    "project" : "data18_13TeV",
    "streamName" : "physics_Main",
    "prodStep" : "deriv",
    "dataType" : "DAOD_HDBS3",
    "version" : "f937_m1972_p4928",
    "name" : "data18_13TeV.00348895.physics_Main.deriv.DAOD_HDBS3.f937_m1972_p4928"
  }
} ]
```

As we can see, an event representation consists of the event number, the GUID of the file with the event, a list of GUIDs of the provenance files of the file with the event and their respective data types — RAW and AOD in this example, and a reference to the dataset to which the file with the event belongs. The reference includes the run number, project name, data type — DAOD_HDBS3 in this example, version, name. This is the dataset with the event that has been indexed by EventIndex and searched in this example. The lines of the output of the previous example (5.1.1) correspond to the file with the event in the searched dataset of DAOD_HDBS3 data type and to its provenance files of RAW and AOD data types, respectively.

- Look up event 348895:3613825, select GUIDs of files with RAW data type, output format `short` — suitable for use with PanDA [15] jobs that are given `--eventPickWithGUID` option:

```
$ event-lookup -c short 348895:3613825 --guid-data-type RAW
348895 3613825 78c6837f-f448-e811-ae3c-44a8420a7621
```

- Arguments of options related to dataset name fields may contain wildcard characters ? (to match any single character) and * (to match any string of zero or more characters). E.g., look up event 348895:3613825 in datasets with the project and version matching the patterns `da*` and `*1972_?4928` respectively:

```
$ event-lookup 348895:3613825 -a '*1972_?4928' -p 'da*'
348895 3613825 4aa369dc-fc28-3444-aab5-c2fdf27a9c69 DAOD_HDBS3 data18_13TeV↩
↪ physics_Main deriv DAOD_HDBS3 f937_m1972_p4928
348895 3613825 78c6837f-f448-e811-ae3c-44a8420a7621 RAW data18_13TeV↩
↪ physics_Main deriv DAOD_HDBS3 f937_m1972_p4928
348895 3613825 0d1c27ff-c5b2-6b46-a0fa-edbdc51c6b77 AOD data18_13TeV↩
↪ physics_Main deriv DAOD_HDBS3 f937_m1972_p4928
```

For the files found, project `data18_13TeV` and version `f937_m1972_p4928` match the patterns specified in the query.

- Look up events from `FILE` (one event per line), 2 events at a time, print each command line prior to invocation:

```
$ <FILE xargs -r -d'\n' -n 2 -t event-lookup --data-type AOD
event-lookup --data-type AOD 300800:5211371 283155:908758
```

Here `event-lookup` is used in combination with the popular `xargs` program. Such an invocation is possible since `event-lookup` accepts (an arbitrary number of) events as ordinary arguments *as most UNIX programs do*.

5.2 dataset-list, event-list

The `dataset-list` utility allows queries concerning datasets for specified runs:

```
usage: dataset-list [-h] [-v] [-p PROJECT] [-s STREAMNAME] [-S PRODSTEP]
                    [-d DATATYPE] [-a VERSION] [-F FILE]
                    [-c plain,name,short,pretty,json] [-o OUTPUT]
                    [run [run ...]]

List (indexed) datasets (metadata) for the runs (all by default).
```

Apart from the dataset name, it returns the number of events, date and time at which the dataset was updated, and whether or not it contains trigger information.

The `event-list` utility returns either event counts or event numbers along with event-related metadata in dataset(s) specified by name(s):

```
usage: event-list [-h] [-v] [-B LB] [-C BCID] [--lpsk LPSK] [--id ID]
                  [--lb1 LB1] [--bcid1 BCID1] [--hpsk HPSK] [-l LIMIT]
                  [-S OFFSET] [-r] [-N] [-F FILE]
                  [-c {plain,long,short,pretty,json}] [-o OUTPUT]
                  [name [name ...]]

List metadata for or count  events  satisfying  the selection criteria from
the datasets.
```

5.2.1 Usage examples

- Read dataset names from standard input, count the events in luminosity block 155, bunch-crossing 284:

```
$ cat names | event-list --count --lumi-block 155 --bcid 284
1 data17_13TeV.00325790.physics_Main.deriv.DAOD_HDBS3.r10250_p3399_p4928
4 data18_13TeV.00348895.physics_Main.deriv.DAOD_SUSY20.f937_m1972_p4928
```

- Output two events, having skipped one, (i.e., list the 2nd and 3rd events) in luminosity block 155:

```
$ event-list -F names -B 155 --limit 2 --offset 1
138323950 155 2049 0 1524711389↩
↪ data18_13TeV.00348895.physics_Main.deriv.DAOD_SUSY20.f937_m1972_p4928
138327037 155 1236 0 1524711389↩
↪ data18_13TeV.00348895.physics_Main.deriv.DAOD_SUSY20.f937_m1972_p4928
7159115 155 2870 0 1496710736↩
↪ data17_13TeV.00325790.physics_Main.deriv.DAOD_HDBS3.r10250_p3399_p4928
7159217 155 2092 0 1496710736↩
↪ data17_13TeV.00325790.physics_Main.deriv.DAOD_HDBS3.r10250_p3399_p4928
```

For each line of the output the first field contains event number, the second the luminosity block number, the third the bunch-crossing id, the fourth the L1 prescaler key, the fifth the time at which the event was recorded (in seconds from the epoch of 1970-01-01T00:00:00Z).

6 Summary

The Query Service — part of the new ATLAS EventIndex system — was developed and deployed for LHC Run 3. The OpenAPI Specification for RESTful APIs was used in the service design and implementation along with standard Java technologies. The required service performance was achieved by parallelization. The service CLI utilities conform to POSIX and are convenient to use in combination with other programs.

The author is grateful to A. Formica for his interest in the work, the fruitful discussions of the results obtained and of his article [16].

References

[1] The ATLAS Collaboration et al., "The ATLAS Experiment at the CERN Large Hadron Collider," *JINST* **3**, S08003 (2008). https://doi.org/10.1088/1748-0221/3/08/S08003

[2] Barberis, D., Alexandrov, I., Alexandrov, E. et al., "The ATLAS EventIndex: A BigData Catalogue for All ATLAS Experiment Events," *Comput Softw Big Sci* **7**, 2 (2023). https://doi.org/10.1007/s41781-023-00096-8

[3] Gallas, E., Alexandrov, E., Alexandrov, I. et al., "Deployment and Operation of the ATLAS EventIndex for LHC Run 3," *Proceedings of the 26th International Conference on Computing in High Energy & Nuclear Physics (CHEP2023)* (2023)

[4] Apache Hadoop: https://hadoop.apache.org

[5] Apache HBase: https://hbase.apache.org

[6] Apache Phoenix: https://phoenix.apache.org

[7] OpenAPI Specification: https://swagger.io/specification

[8] Jersey 2 JAX-RS API implementation: https://eclipse-ee4j.github.io/jersey

[9] Spring Framework: https://spring.io

[10] POSIX.1-2017: IEEE Std 1003.1™-2017 and The Open Group Technical Standard Base Specifications, Issue 7, 2018 edition, Utility Conventions: https://pubs.opengroup.org/onlinepubs/9699919799/basedefs/V1_chap12.html

[11] GNU Standards for Command Line Interfaces: https://www.gnu.org/prep/standards/html_node/Command_002dLine-Interfaces.html#Command_002dLine-Interfaces

[12] GNU Project: https://www.gnu.org

[13] Free Software Foundation: https://www.fsf.org

[14] RFC 4122: A Universally Unique IDentifier (UUID) URN Namespace http://www.ietf.org/rfc/rfc4122.txt

[15] Barreiro Megino, F. H., De, K., Klimentov, A. et al., "PanDA for ATLAS distributed computing in the next decade," *J. Phys.: Conf. Ser.* **898**, 052002 (2017). https://doi.org/10.1088/1742-6596/898/5/052002

[16] Rinaldi, L., Formica, A., Gallas, E. et al., "Conditions evolution of an experiment in mid-life, without the crisis (in ATLAS)," *EPJ Web Conf.* **214**, 04052 (2019). https://doi.org/10.1051/epjconf/201921404052