Testing framework and monitoring system for the ATLAS EventIndex

Elizaveta Cherepanova[1,*], *Elizabeth J.* Gallas[2,**], *Fedor* Prokoshin[3], and *Miguel* Villaplana Pérez[4]

[1]University of Amsterdam and NIKHEF, Amsterdam, Netherlands
[2]University of Oxford, Oxford, UK
[3]Joint Institute for Nuclear Physics, Dubna, Russia
[4]IFIC, University of Valencia and CSIC, Valencia, Spain

Abstract. The ATLAS EventIndex is a global catalogue of the events collected, processed or generated by the ATLAS experiment. The system was upgraded in advance of LHC Run 3, with a migration of the Run 1 and Run 2 data from HDFS MapFiles to HBase tables with a Phoenix interface. Two frameworks for testing functionality and performance of the new system have been developed. There are two types of tests running. First, the functional test that must check the correct functioning of the import chain. These tests run event picking over a random set of recently imported data to see if the data have been imported correctly, and can be accessed by both the CLI and the PanDA client. The second, the performance test, generates event lookup queries on sets of the EventIndex data and measures the response times. These tests enable studies of the response time dependence on the amount of requested data, and data sample type and size. Both types of tests run regularly on the existing system. The results of the regular tests as well as the statuses of the main EventIndex subsystems (services health, loaders status, filesystem usage, etc.) are sent to InfluxDB in JSON format via HTTP requests and are displayed on Grafana monitoring dashboards. In case (part of) the system misbehaves or becomes unresponsive, alarms are raised by the monitoring system.

1 Introduction

The ATLAS EventIndex [1] is the tool that collects, checks and stores information about the main properties of all real or simulated events that were collected, processed or generated by the ATLAS experiment [2], and points to the files that contain them. The system provides:

- a way to collect and store event information using modern technologies;
- various tools to access this information through command line and web services;
- an indexing system that points to these events in millions of files scattered through a worldwide distributed computing system.

*e-mail: Elizaveta.Cherepanova@cern.ch
**e-mail: Elizabeth.Gallas@physics.ox.ac.uk

The system comprises multiple subsystems, each dedicated to performing a specific task in accordance with the data flow. The implementation is based on the BigData tools that are available in the Apache Hadoop ecosystem [3]. A schema of the EventIndex architecture implemented for LHC Run 3 is shown in Figure 1 and a description of every component can be found in Ref. [4].

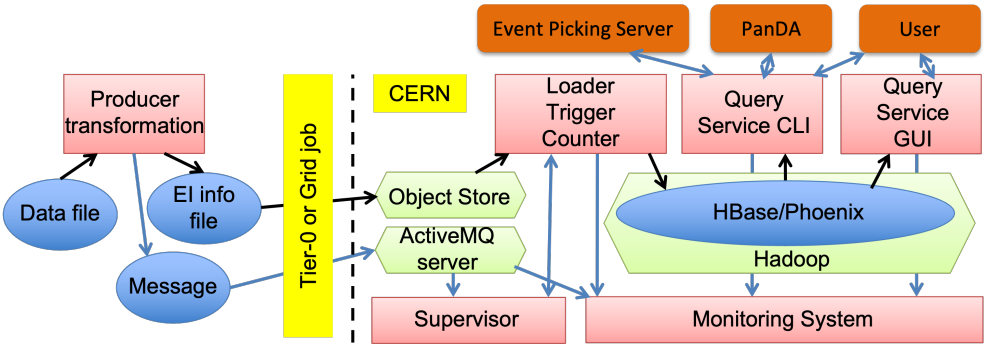

Figure 1: Architecture of the EventIndex system as implemented for LHC Run 3. The black arrows indicate the indexing data flow; the blue arrows indicate the information flow or exchange between the different tools. A description of every component can be found in Ref. [4].

Ideally, the system should be available and accessible under various conditions and give a response in the appropriate time, depending on the request. To monitor the system's current state, functionality, and performance, a specialized framework has been created. The Monitoring System is responsible for tracking the test results and monitoring the statuses of the main subsystems. This paper describes the current implementation of the Monitoring System for the ATLAS EventIndex and the testing framework.

2 Monitoring system

The multi-component structure of the EventIndex system requires monitoring of each of its subsystems and processes. To accomplish this, the monitoring system has two general parts called the "producer" and the "viewer".

The producer part collects and stores data about the status and the performance of the system. It includes a scheduler, a number of Python scripts, and the database (storing each result with a timestamp). The scheduler uses a cron utility to run the Python scripts at fixed times. The Python scripts collect data from the servers at CERN and insert them into the database. Several different modules monitor different system sub-components. Each of these requires a different approach for data collection and processing: scanning log files and Hadoop HDFS space, sending REST requests [5], analysis of the web pages. Thus every module has its own Python script and scheduler to run it. The producer transfers the data using HTTP requests to InfluxDB [6], a database system well suited for time-series data.

The viewer part is responsible for the graphical presentation of data. A Grafana [7] dashboard is used for the visualisation. Every EventIndex sub-component has a general status dashboard along with additional dashboards for monitoring more detailed parameters of that component. The status of critical EventIndex services is also fed into the global ATLAS service monitoring view that is checked by computing operation shifters.

Figure 2 shows the functional schema of the EventIndex Monitoring System and the data flow.

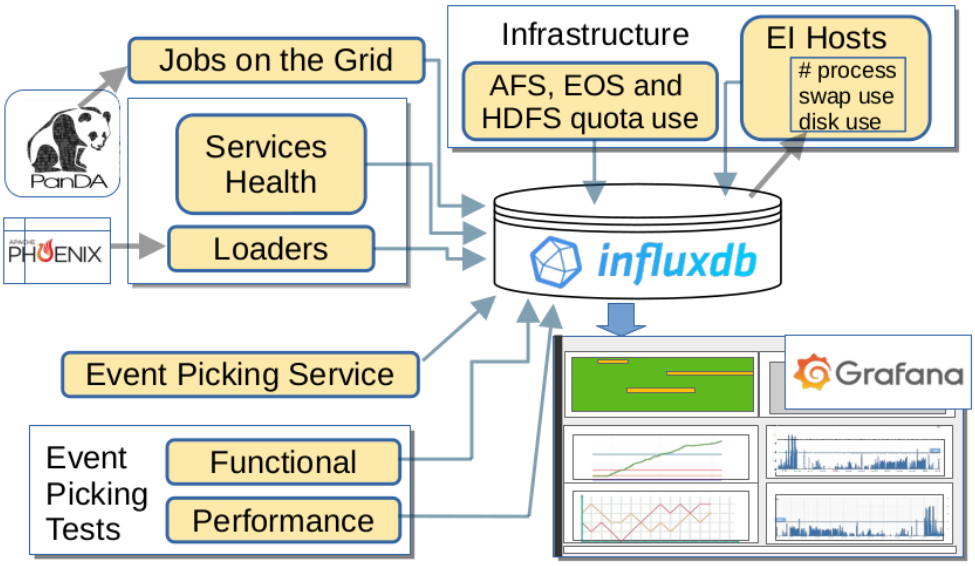

Figure 2: Diagram for the monitoring system of the EventIndex components. Several modules collect information from multiple sources about the status of EventIndex processes (top left), functional and performance tests (bottom left) and the computing infrastructure (top right) and store this information in an InfluxDB database; the data are then displayed in Grafana dashboards.

3 Functional tests

When a dataset is marked as complete in the ATLAS metadata database AMI [8], dataset indexation is triggered automatically. Functional tests aim to check the functionality of the performance chain using newly indexed datasets. The tests run twice per week, on Mondays and Thursdays. Each test procedure takes the following steps:

- Scan the EventIndex catalog to get a list of datasets indexed between 24 hours and 4 days ago;
- For each sample found:

  – Get a file reference (Globally Unique IDentifier, GUID [9]) and the runNumber-eventNumber pair of a random event in the dataset;

  – Locally (at CERN) run the EventIndex CLI (command line interface) [10] event lookup utility with the runNumber-eventNumber pair as input.

  - If unsuccessful (i.e. the correct GUID is not returned), log the incident for later inspection.
  - If successful, submit two event picking jobs to PanDA [11] which launch the tasks on the WLCG (Grid) [12] to retrieve the event(s).

    1. using the runNumber-eventNumber pair and the GUID;
    2. using the runNumber-eventNumber pair (in this case, PanDA will query the EventIndex for the GUID before launching the Grid job).

- Send the complete list of samples that failed event lookup to the EventIndex operations mailing list.

The execution status of the functional test Grid jobs can be checked in the PanDA monitor web interface as shown in Figure 3.

Task ID	Request ID	Task name	N files total	N files done	N files failed	%	Status (JEDI)	Duration, days	Task logged status	Jobs failure, %
32889647	706168	group.proj-evind.EPRAW.NG.v0.data22_13p6TeV.00440613.physics_ZeroBias.merge.AOD.f1321_m2153.20230327/	1	1	0	100	done	0.04	-	0
32889646	706167	group.proj-evind.EPRAW.YG.v0.data22_13p6TeV.00440613.physics_ZeroBias.merge.AOD.f1321_m2153.20230327/	1	1	0	100	done	0.13	-	0
32889645	706166	group.proj-evind.EPRAW.NG.v0.data22_13p6TeV.00440613.physics_Main.merge.AOD.f1321_m2153.20230327/	1	1	0	100	done	0.16	-	16.67
32889644	706165	group.proj-evind.EPRAW.YG.v0.data22_13p6TeV.00440613.physics_Main.merge.AOD.f1321_m2153.20230327/	1	1	0	100	done	0.03	-	0
32889643	706156	group.proj-evind.EPRAW.NG.v0.data22_13p6TeV.00440613.physics_Main.deriv.DAOD_PHYS.f1321_m2153_p5442.20230327/	1	1	0	100	done	0.16	-	0

Figure 3: Status of the functional test jobs on the Grid displayed on PanDA monitor.

The results of the functional tests are collected by the Monitoring System and displayed on the Grafana monitor. Figure 4 shows the Grafana dashboard dedicated to the functional tests. If any local test finishes with an error it will be indicated on the main EventIndex Grafana dashboard.

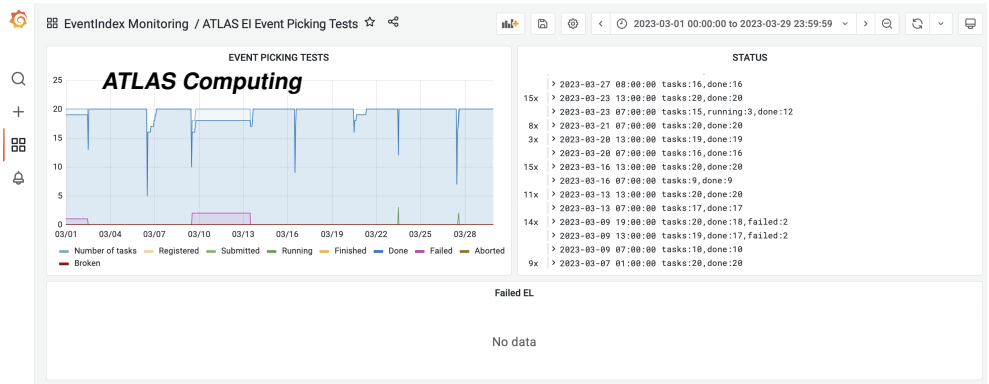

Figure 4: Overall view of test jobs' status over time in Grafana. The top pads show the different statuses of tests in time, graphical (left) and textual (right) representations. The bottom pad lists only jobs that failed locally (at CERN).

4 Performance tests

The performance tests are designed to check the responsiveness of the EventIndex system and measure the performance time. A suite of event lookup tests was created, reflecting the range of capabilities of the system and based on queries that a user might try using the CLI. The tests were formed from a list of about 50000 events spanning real data samples recorded in 2015-2018 during LHC Run 2. All tests execute the "event-lookup" CLI client command and the response includes information about all the events requested: the event GUIDs along with the full dataset names and data types.

A summary of the performance test configurations and their average response times is shown in Table 1. The content is described as follows:

Table 1: Performance test configurations and results (averaged over all tests executed during one week).

Data	Type	Number of events	Average performance time, s
data15 (1M)	random	1000	11.8
data15 (Multi DS)	random	1000	3.3
data17 (Multi DS)	random	1000	15.3
data18 (1M)	random	10	1.48
data18 (1M)	random	100	1.71
data18 (1M)	random	1000	2.98
data18 (1M)	random	10000	11.6
data17 (1M)	fixed	1000	2.07
data17 (Multi DS)	fixed	1000	14.8

- The **"Data"** column shows the year of data-taking (for example 'data15' tests used data taken in 2015, the first year of Run 2 operations)
  - the note in parenthesis indicates the nature of the data sample:
    - (1M) indicates that the dataset queried is a large dataset with over 1 million events.
    - (Multi DS) indicates the test is over multiple datasets (such as more than one run in the same year of data-taking).
- The **"Type"** column indicates the test type:
  - value 'random' means the test used a randomly generated event list.
  - value 'fixed' means the test used the same event list every time.
- The **"Number of events"** column shows the number of events requested (which ranges from 10 events to 10000 events).
- The **"Average performance time"** column shows the measured time required to get the response (averaged over results obtained during one week of the tests running).

A cron scheduler launches the tests: The 'random' type tests are run every hour while the 'fixed' type tests are executed every two hours. The results show that:

- The response times are very fast (just a few seconds) even for many thousand events. This is demonstrated over all tests.
- The response time grows with the number of events requested. This is most clearly shown by comparing the response of the tests on the data18 samples (where only the number of requested events varies).
- To understand the variation in response of the 'Multi DS' tests, an in-depth analysis comparing the configurations (not shown in the table) concluded that the response time increases with:
  1. the number of datasets in the 'Multi DS' test (there are more datasets in the data17 case than in the data15 case)
  2. the number of events per dataset in those tests (there are 50% more events in data17 datasets than in data15 datasets of the tests).

Figure 5 shows the time-series data over the week (from which the averages are computed) to show the variation in response times of the individual tests over the days of the example week period. The average performance time varies a little depending on the number of events to search and the number of datasets from which the events were taken. However it remains within a few seconds, as is desirable for these types of queries. The occasional peaks in the performance are due to interference with other activities on the Hadoop cluster which are not related to the EventIndex.

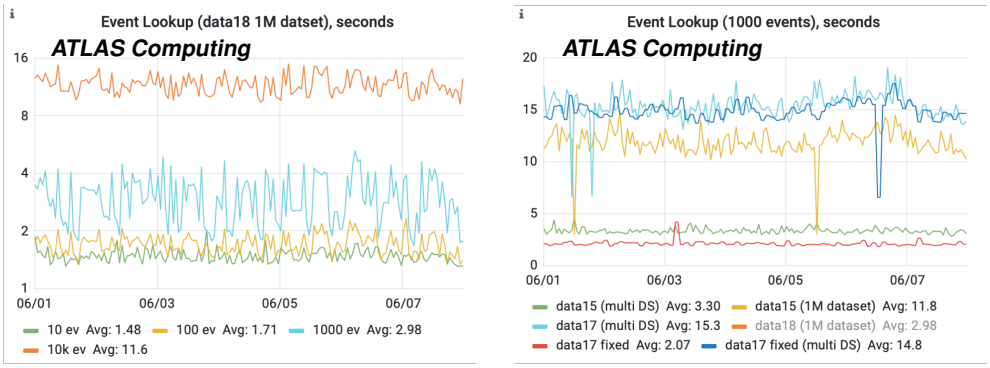

Figure 5: Response times of the event lookup queries selecting events out of a dataset with 1 million records or a mixture of data18 dataset (left) and data15, data17 datasets (right) as a function of time.

The test results are written to JSON files and then displayed on the Grafana Monitor. The dedicated dashboard for the performance tests is shown in Figure 6. If a test query finishes with an error it will be indicated on the main EventIndex Grafana dashboard.

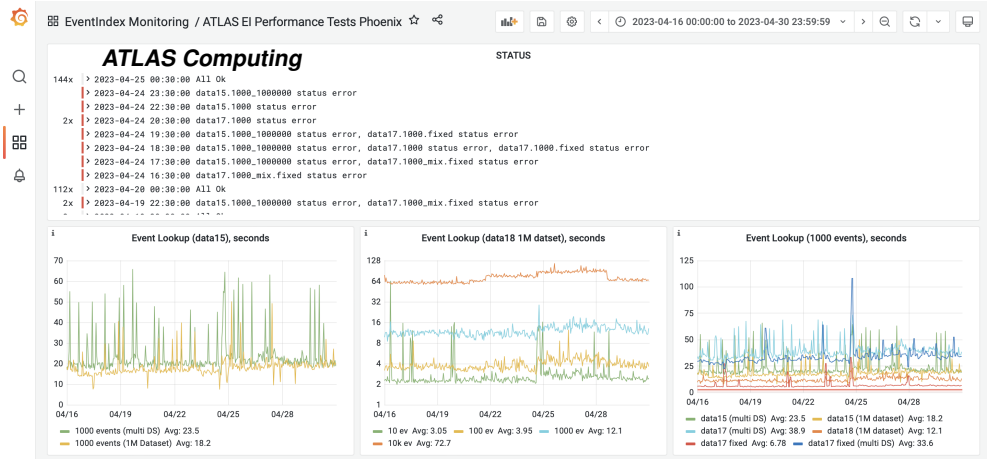

Figure 6: Overall view of tests' status over time in Grafana. The top pad shows the different statuses of tests in time with information about errors if there are any. The bottom pad shows the results of the performance tests: only data15 (left), only data18 (middle), only searches for 1000 events (right).

5 Conclusions

The Monitoring System is a crucial element of the EventIndex system, providing a comprehensive representation of the status of all EventIndex subsystems and key parameters. Its effectiveness in detecting system issues has been proven, enabling prompt resolution and ensuring the delivery of services to end users with optimal functionality and reliability.

The functional and performance test frameworks were developed to control the well-being and the responsiveness of the EventIndex tool. The tests run on a regular basis, both locally (at CERN) and remotely on the Grid. They are quick indicators of technical problems not necessarily caused by EventIndex malfunctions. These tests are essential for conducting performance studies and optimizing the system, particularly during infrastructure and system upgrades. They were used extensively during preparations for LHC Run 3 in early 2022.

References

[1] Barberis, D. et al., *The ATLAS EventIndex: A BigData Catalogue for All ATLAS Experiment Events, Comput.Softw.Big Sci.* **7** (2023) 1,2. https://doi.org/10.1007/s41781-023-00096-8

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

[3] Apache Hadoop and associated tools: https://hadoop.apache.org

[4] Barberis, D. et al., *Deployment and Operation of the ATLAS EventIndex for LHC Run 3*, these proceedings (2023).

[5] REST: https://restfulapi.net

[6] InfluxDB: https://www.influxdata.com

[7] Grafana: https://grafana.com

[8] Fulachier, J. et al. *ATLAS Metadata Interface (AMI), a generic metadata framework. J. Phys. Conf. Ser.* **898** (2017) 062001. https://doi.org/10.1088/1742-6596/898/6/062001

[9] GUID: https://www.rfc-editor.org/rfc/pdfrfc/rfc4122.txt.pdf

[10] Rybkine, G., *Query Service for the new ATLAS EventIndex system*, these proceedings (2023).

[11] Barreiro Megino, F.H. 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

[12] Bird, I. et al., *LHC Computing Grid: Technical Design Report.* Document LCG-TDR-001, CERN-LHCC-2005-024. ISBN 978-92-9083-253-9 (2005)