The ATLAS EventIndex: data flow and inclusion of other metadata

ACAT2016 – 17th International workshop on Advanced Computing and Analysis Techniques in physics research, 18-22 January 2016, Valparaíso, Chile

D. Barberis[1], S.E. Cárdenas Zárate[2], A. Favareto[3], A. Fernandez Casani[4], E.J. Gallas[5], C. Garcia Montoro[4], S. Gonzalez de la Hoz[4], J. Hrivnac[6], D. Malon[7], F. Prokoshin[2*], J. Salt[4], J. Sanchez[4], R. Toebbicke[8], R. Yuan[6]
on behalf of the ATLAS Collaboration

[1] Università di Genova and INFN, Genova, Italy, [2] Universidad Técnica Federico Santa Maria, Valparaíso, Chile, [3] Università degli Studi and INFN, Genova, Italy, [4] Instituto de Fisica Corpuscular, Valencia, Spain, [5] Department of Physics, Oxford University, Oxford, United Kingdom, [6] Laboratoire de l'Accelerateur Lineaire, Orsay, France, [7] Argonne National Laboratory, Argonne, USA, [7] CERN, Europe

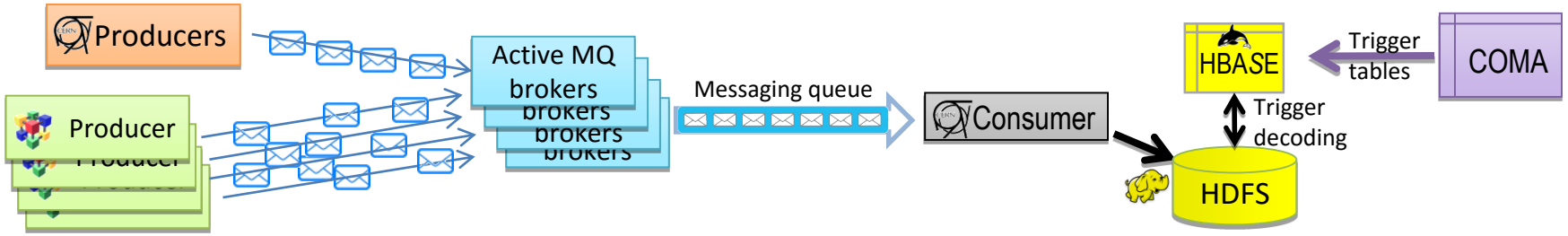
*Presenter

The EventIndex

- ATLAS is a complex system that employ detectors with millions of read-out channels, and gather billions of events in data taking periods that may extend to several years.
- It producing several petabytes of data per year, with different physics conditions and data formats. Using of different types of *metadata* (data about data) makes possible effective usage of these tremendous amounts of information
  - Dataset metadata deals with data storage organization
  - Run-wise metadata deals with slowly changing parameters and links to the conditions
- It is useful to have a catalogue containing information about the basic properties of event data and references to its storage location, and also providing a means of rapid search through this event metadata.
- The EventIndex is a system designed to be a complete catalogue of ATLAS events:
  - All events, real and simulated data at all processing stages
  - Experience obtained in the process of commissioning and first period of use, input from the physics groups and performance constrain exclusion some of the short-living data from being processed by the EventIndex.
  - We also trying to avoid repeated processing of the same information (like trigger decisions) from different production stages and tags.

The EventIndex

- The EventIndex consist of number of autonomous components, providing collection and processing of EventIndex data. Modular approach allowed share development between different institutions.
- 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 producers is sent via ActiveMQ message broker to the Hadoop store at CERN
- At CERN messages are read by Consumer program and stored as Hadoop Map file objects.
- Data is being validated for data completeness and duplicated events, and decoding of the trigger information performed using trigger tables from COMA – Condition Metadata Store
- Users can access EventIndex information through the query services:
  - Command line interface
  - Web Services with graphical and text/plain output suitable for the ATLAS *PanDA* (Production And Distributed Analysis) Event Service.



The event record contains

- Event idetifiers:
  - Run Number, Event Number, Trigger Stream, Luminosity block
- Trigger decisions for different levels
  - L1 (Level 1) - After veto, after prescale, before prescale.
  - L2 (Level 2) - Physics, pass through, resurrected (for Run 1 only)
  - EF (Level 3) - Physics, pass through, resurrected (for Run 2 replaced with HLT, using the same record fields)
- References (GUID + internal pointer) to the events at different processing stages in permanent files, generated by central production
  - Due to performance and space constrains data with short active life expectancy (derived data for workgroups - *DAOD*) is no longer indexed

Size of the event record:

- ~**350** B/event → **350** GB of uncompressed information per billion (10^9) events × number of processings and formats.
  - The real storage needed depends on the chosen technology, replication multiplicity and internal data format and indexing

EventIndex use cases

- Event Picking
  - Request reference to the specific event in given format and processing type
- Event Counting.
  - Trigger counting, collected by EventIndex and provided to users by COMA
- Duplicate finding
  - Detecting duplicated data with the scope of dataset or physics collection

Data types processed by the EventIndex

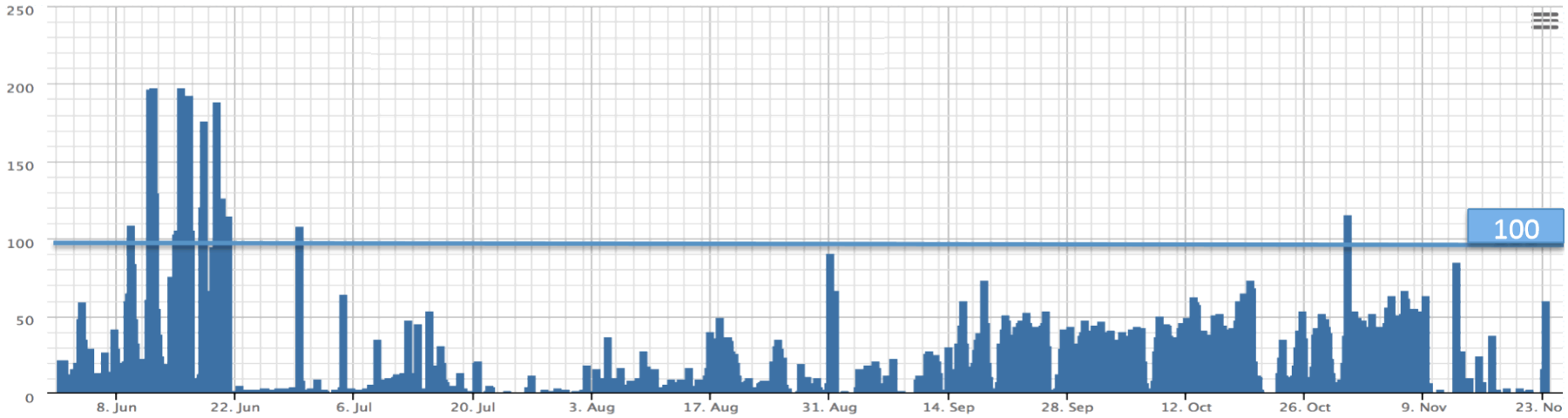
- Top priority for EI is Analysis oriented data (**AOD**), real data and MC.
  - Note that indexing the AODs provides also indexing of RAW through the provenance information that is picked up.
- MC datasets containing generated events (**EVNT**) are being indexed for duplicate event detection
- Routine 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 of it is indexed for that purpose.
- We try to avoid repeated processing of the trigger data for the same events, so we have
  - Trigger processing enabled for real data AOD from Tier0 (including spill-over) and for MC AOD
  - Trigger processing disabled for real data AOD from reprocessing and for DAOD

Information collection

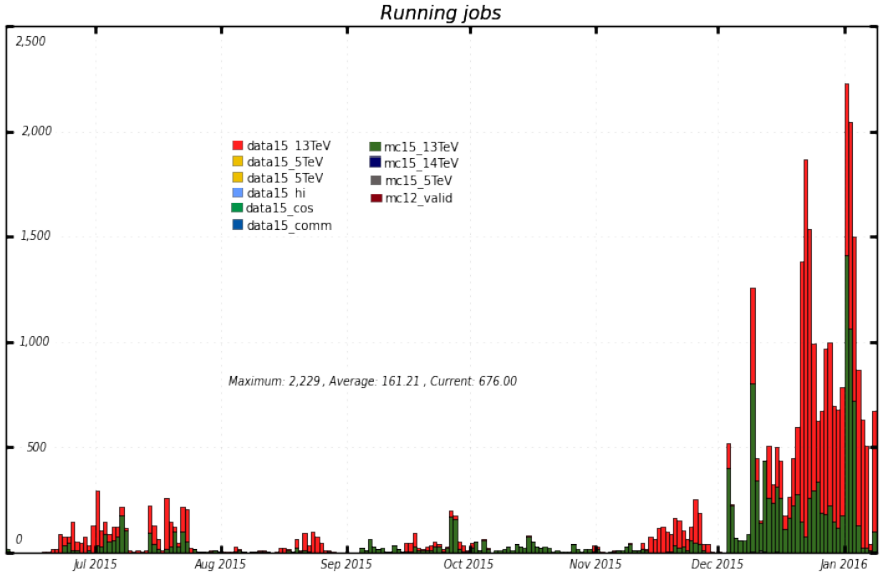
- Datasets produced on **Tier 0** centre at CERN are indexed by tasks running EventIndex Producer transformations
- On the **Grid** EventIndex uses second generation ATLAS production system (ProdSys2) to index datasets as soon as they appear in the AMI catalogue as valid and completed
- Information is gathered on data produced/modified daily on the Grid by querying ATLAS AMI metadata catalog.
  - We select VALID datasets from mc15 and data15 projects
  - A catalog in ATLAS Distributed Data Management system (RUCIO) is queried for IDs of the datasets to feed the EventIndex processing jobs
  - All tasks are managed by automatic scripts running as a cron jobs
- Datasets from the list are attached to the “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.
- EventIndex production tasks are automatically run on the Grid by ProdSys2. It creates a task for each dataset that was added to the technical container

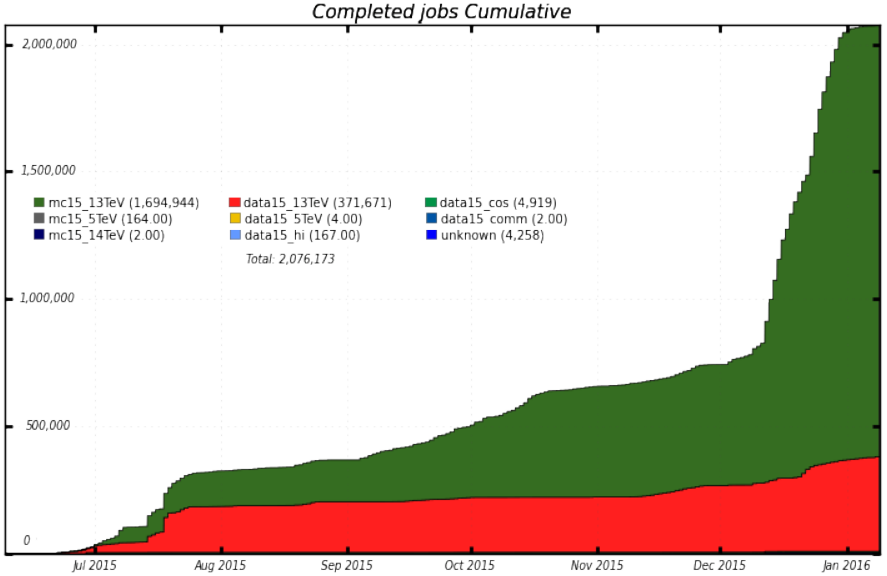
Jobs run on Tier 0 and on the Grid

EI jobs on the Tier-0



EI jobs on the Grid





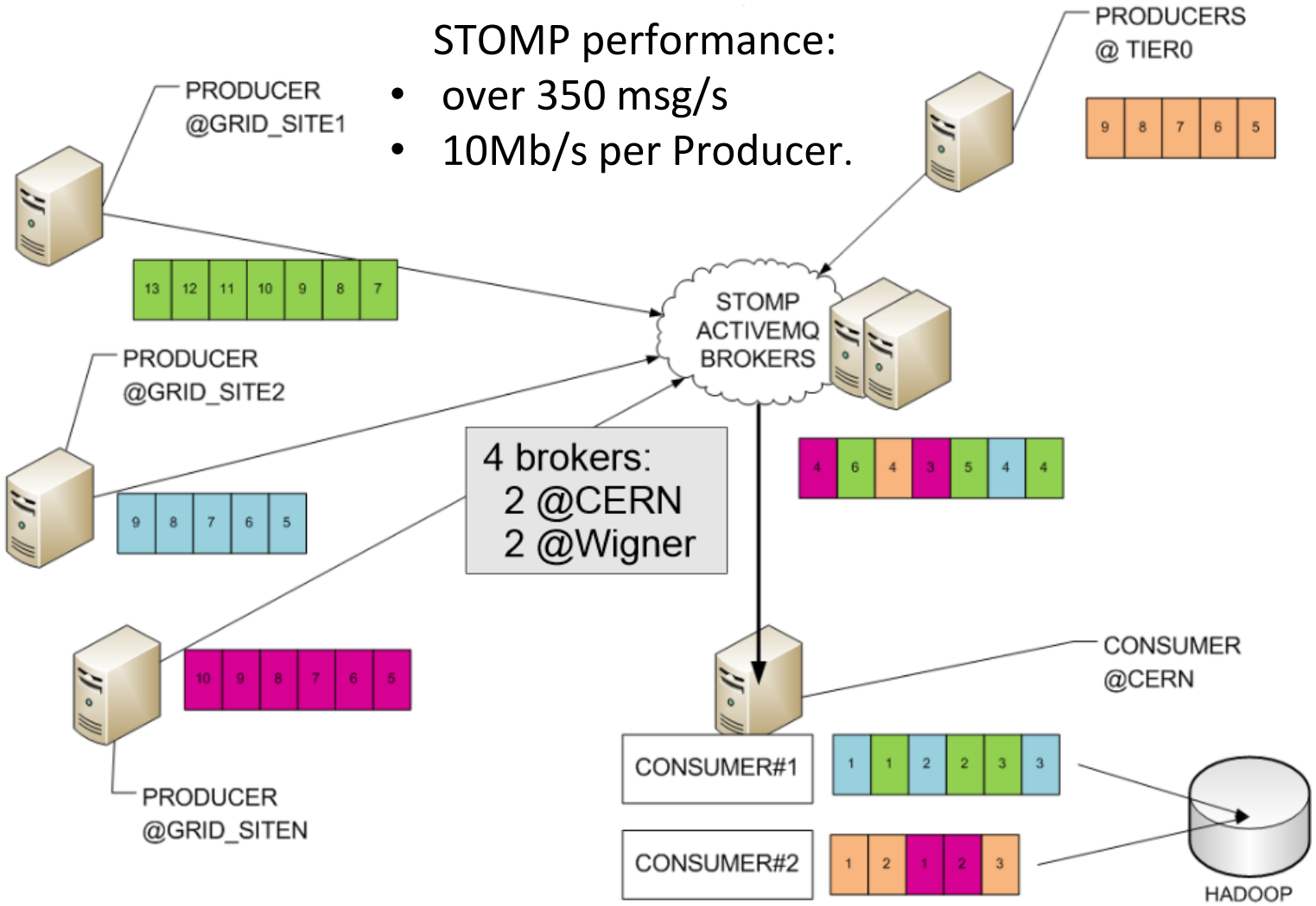
EventIndex Producers extract event data and send information to consumers running at servers at CERN

- Event data is **read** from intermediate file, trigger masks are compressed, duplicate information is dropped and records are encoded into JSON to build the text message
- EventIndex uses a Streaming Text Oriented Messaging Protocol (STOMP) to **transport** information between producer and consumers. Message size is set small (1-10kB)
- Messages are **received** via Active MQ message broker, decoded and queued for the consumers
- At CERN messages are **read** by EventIndex consumers, decoded and organized into Hadoop MapFile objects.
- Consumers also do **validation** that verifies that all EventIndex for a collection (dataset) contains all the files (GUIDs) and all the events, detect multiple processing of the same file and job failures.
- Consumer sends back **statistic** messages (# messages, # files, # events, #uniq_events) to broker to report status. It is used, in particular, by monitoring system

The messaging system

STOMP performance:

- over 350 msg/s
- 10Mb/s per Producer.



Measured performance for sending real events:

- **200K event/s**
- **60Mb/s**

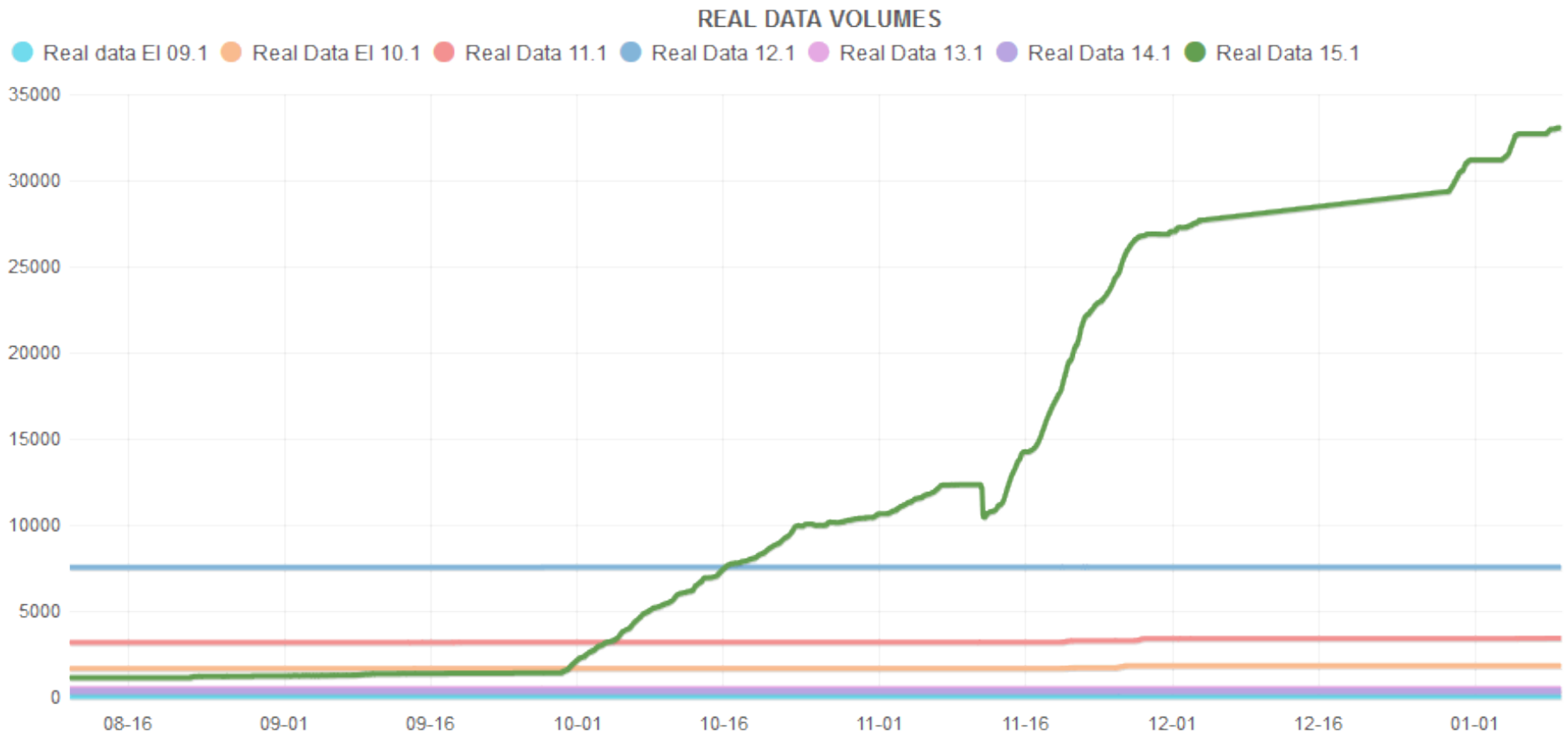
(1Broker, 6 Producers, 4 Consumers, 50K events/job)

Data storage and handling

- **Hadoop** was chosen as the storage technology for the EventIndex:
  - Platform is **provided** and **supported** by CERN-IT
  - DDM (Distributed Data Management) project also uses Hadoop
  - **Plenty** of tools to organise the data, index them internally and search them
  - Showed satisfactory **performance** in prototype populated with a year of ATLAS data
- Storage Structure:
  - Data are stored as **mapfiles** in HDFS (Hadoop File System)
  - Data are internally catalogued Hadoop HBase: metadata about HDFS files.
  - Triggers are decoded and added to the mapfiles
- Search performance enhanced using keyed indexes based on use cases:
  - Searches based on a key give **immediate** results (seconds)
  - Complex searches use MapReduce (MR) and require **1-2 minutes** for typical event collections

Data volume in Hadoop

- **50 TB** of EI data (+62 TB of transient data)
  - **46 TB** of real data + **3.3 TB** of MC data
- 17 TB of consumer data (transient)
- 33 TB of consumer archive data

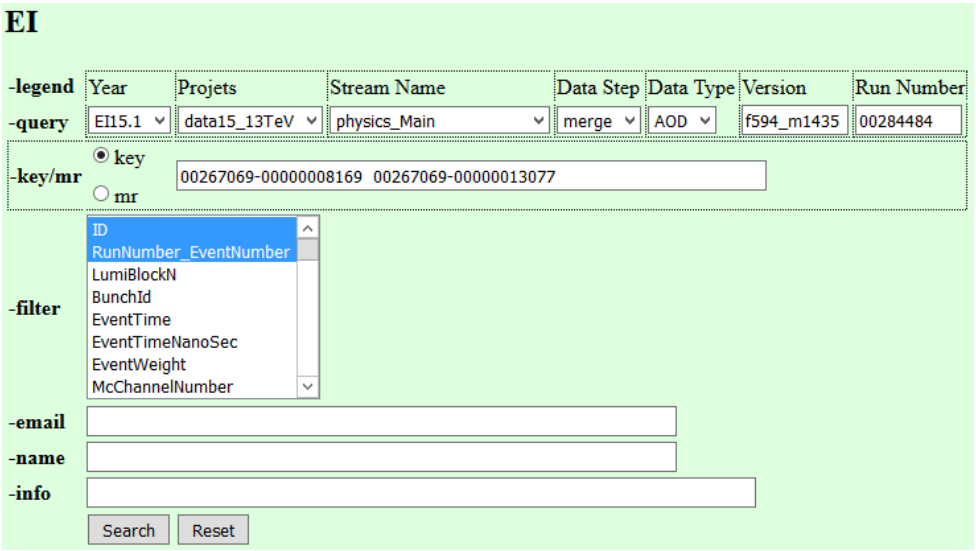


Query Services

- Users can access EventIndex information through the query services:
  - Command line interface, available on the cluster and remotely from AFS, CVMFS and downloadable tar file
  - Web Services with graphical and text/plain output suitable for the Panda Event Service.
    - `https://atlas-event-index.cern.ch/EIHadoop/`

Search executed in two steps:

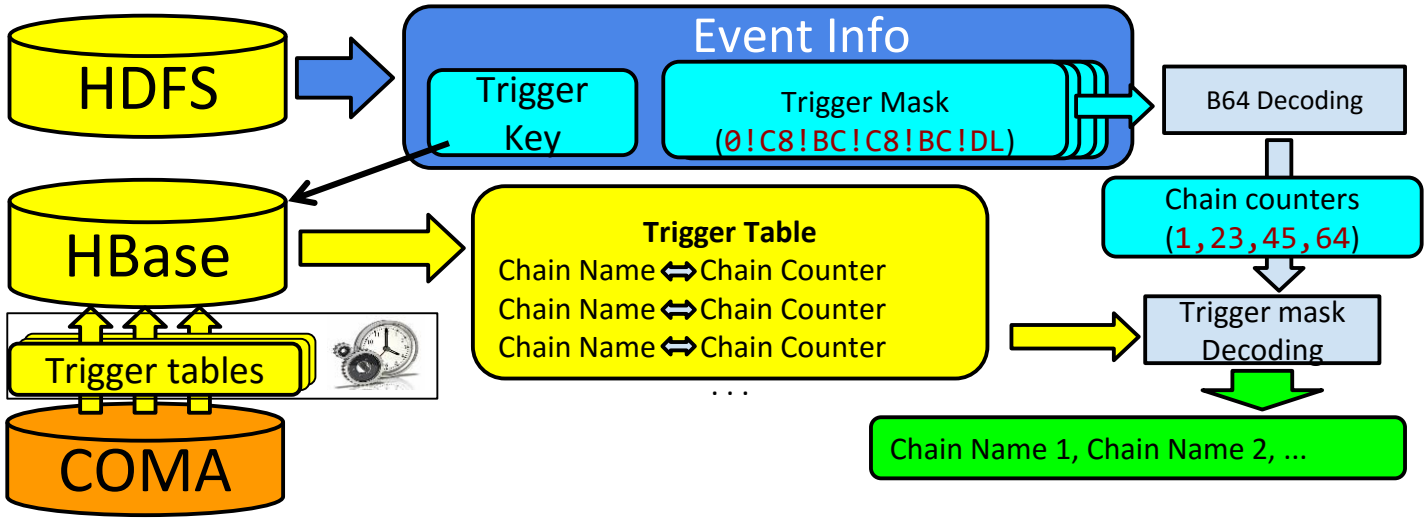
1. HBase Catalog finds TagFiles to search for
   - Speed depends on search criteria
     - from < 1s (exact key based search)
     - to 20s (full scan search)
2. Hadoop executes search in those files:
   - key-search (almost immediate) if searching directly on RunNumber-EventNumber
   - Map/Reduce job if searching with other attributes (Trigger,...)



- Remote access goes via proxy server to Tomcat @ Hadoop cluster to assure
  - Hadoop isolation from outside environment
  - fallback & load-balancing (not yet)

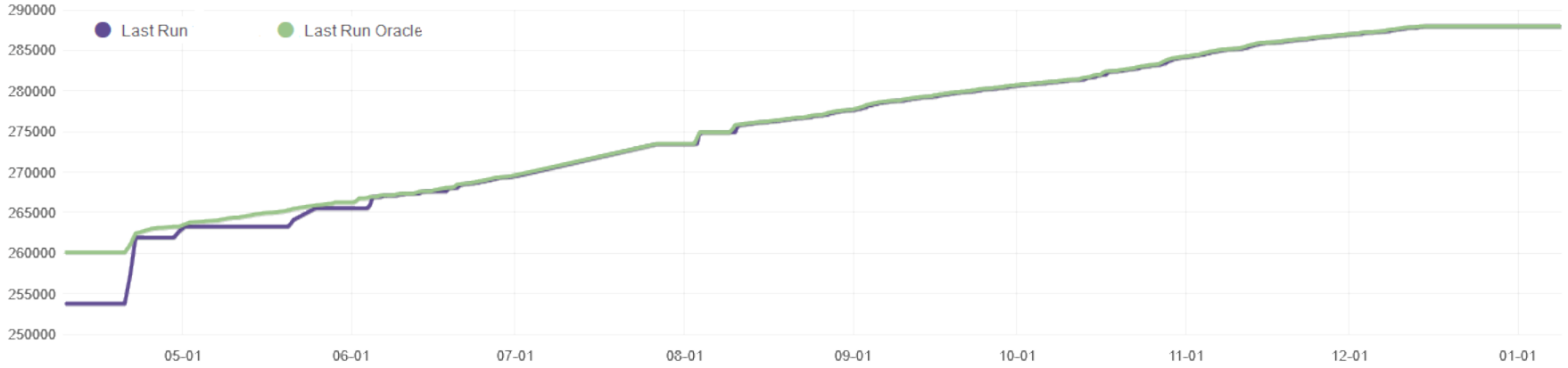
Trigger information Decoding

- Trigger decision is 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.
  - A position of this bit in mask (chain counter) indicates which chains have 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
- EventIndex uses trigger tables replicated from COMA for the trigger decision decoding. Trigger masks from event record are being 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.



Trigger information exchange with COMA

- Trigger tables use SMK as key and ChainName:ChainCounter pairs as columns
- Additionally there is Run:SMK table for verification
- Currently trigger table contain information on 616 SMKs and 10526 runs
  - Includes data for 2015 HI runs
- Trigger table replication is done by a cron job running daily
- Automatic replication is in progress since Mar 2015



- EventIndex Serves as a source of data on trigger event counts for COMA
  - The EventIndex program does counting on server within a CERN network providing information through RESTful service that can be run outside CERN

Dataset information

- We use AMI information for identifying new and changed datasets that have to be processed by the EventIndex
  - We query AMI through python interface for a list of complete dataset containers of certain types with lastModificationTime inside previous day and flagged “Valid”
  - As large datasets in ATLAS can be produced by a number of tasks, containers are made of one or few **“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**).
- Current procedure based on dataset modification times has a number of drawbacks:
  - Difficult to detect datasets that became **corrupted/obsolete**, as we look for VALID datasets
  - Dataset modification time may **change** for reasons not related to event content
  - No feedback to AMI
- It was proposed to introduce special table in the AMI for information exchange between 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.
  - EventIndex will put dataset import status into other field, `EI_DS_STATE`, providing to AMI information on data consistency problems detected by EventIndex.
  - This machinery is currently in the process of implementation by AMI Team

Event lookup performance

- Benchmark results for the searching for events from the list
- Events specified by run-event pairs
- Only AODs we searched
- Requests were done from “remote” machine

Number of events	Found GUIDs	Searched runs	Searched tag Files	Time (s)
1	3	1	6	2
10	39	1	6	2
50	185	1	6	3
100	381	2	12	4
500	1803	9	54	11
1000	3808	22	146	31
5000	9957	50	319	120
8879	24179	66	426	193

- HBase, Hadoop and Tomcat keeps some results in cache/memory so the second (overlapping) search is usually much faster then the first one.
- Performance depends on load on the EI gateway machine at CERN, so timing may vary a lot

Development and status

- The EventIndex development started in 2012 and its main components were ready at the end of 2014.
- Run 1 data loading started at February 2015
- Run2 data now flowing continuously from Tier-0/Grid and available almost in real time
- All major components exist and work:
  - Data Collection: Producer transform runs at Tier-0 and on the Grid
  - Data Collection: Consumer reads data from the ActiveMQ servers, validates them and stores to HDFS
  - Storage System: Data organization in Hadoop and indexing in catalogue
  - Storage System: Trigger decoding interface
  - Query System: CLI and web interfaces. Also EventLookup for event picking
  - Monitoring: System level monitoring in the new CERN Kibana environment
- Currently working on
  - Optimization of the internal data representation
  - Further automation of the data flow
  - System interconnections and monitoring
  - Automatic checks of production completeness