The ATLAS EventIndex: Full chain deployment and first operation

Álvaro Fernández Casaní

Instituto de Física Corpuscular (IFIC)
Universitat de València – CSIC
On behalf of the ATLAS Collaboration

ICHEP 2014 VALENCIA

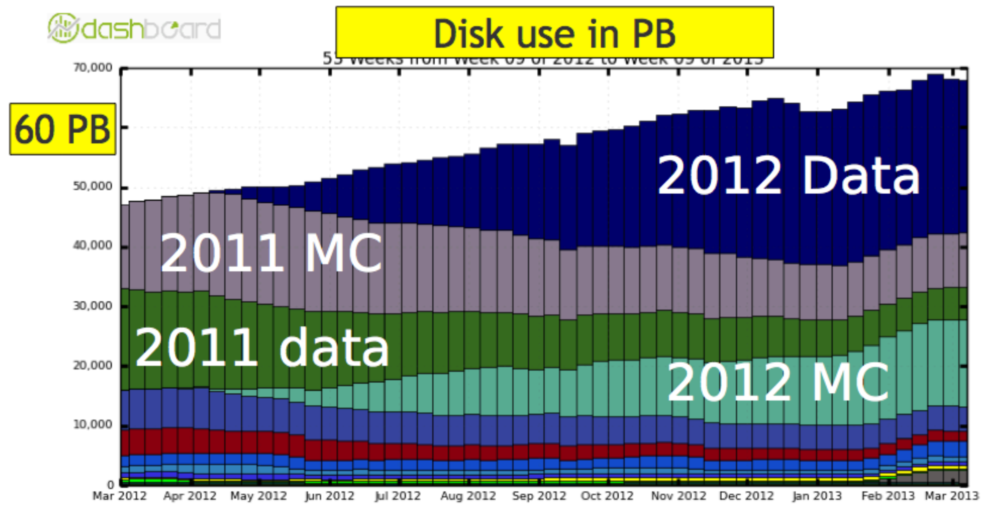
37th INTERNATIONAL CONFERENCE ON HIGH ENERGY PHYSICS

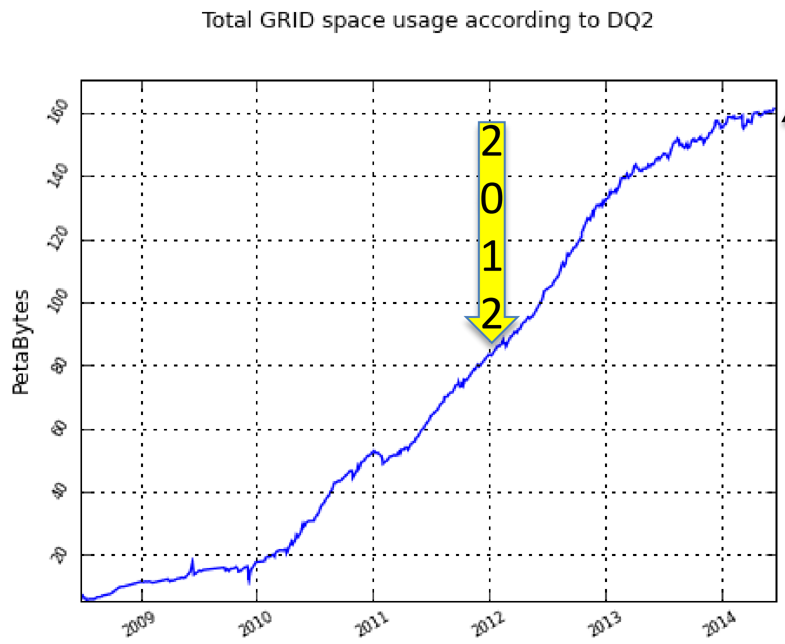
2 - 9 - JULY - 2014 - VALENCIA

Outline

- ATLAS Experiment
- Motivation for an event index
- Prototype and first operation
  - Distributed data collection
  - Hadoop Core
  - Query Services
- Summary and Conclusions

Computing Challenges





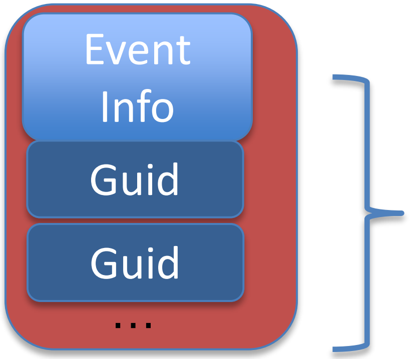
- Experiments like ATLAS produce large amounts of data:
  - 2 billion real events and 4 billion simulated events were produced in 2011 and the same in 2012.
  - These numbers increase if we take into account all the reprocessing.
- After current Long Shutdown 1, in 2015:
  - Increasing rate to 1 kHz . Event size 0.8/1MB.
  - RAW DATA 800/1000 MB/s
  - Translates globally in CPU +20%, DISK +15%, TAPE +15% each year.

Event Cataloguing

- All this information needs cataloguing according to different points of view to meet multiple use cases and search criteria.
- A system that contains the reference to the file that includes every event at every stage of processing is necessary to recall selected events from data storage systems.
- ATLAS already has a database called *EventTag*, with some issues:
  - Using traditional DBMS Oracle, expensive to maintain
  - Scaling up in hardware is expensive.
  - Not all use cases implemented.
  - Not completely automatized: asynchronous, error prone
  - Consistency issues
  - Performance not satisfactory
- *EventTag* is potentially very useful but has to improve.

EventIndex

- **We need another approach: EVENTINDEX PROJECT, using Structured Storage (ie: NoSQL) technologies:**
  - Appropriate to scale for increased data rates (Commodity HW, Fixed cost/unit), scalable.
  - Cheaper, easier to use and faster for sparse searches over large amounts of data.
- A complete catalogue of all ATLAS events:
  – All events, real and simulated data.
  – All processing stages



Needed EventIndex information ~= 300bytes to 1Kbyte per event:

•Event identifiers
•Online trigger pattern and hit counts
•References (pointers) to the events at each processing stage (RAW, ESD, AOD, NTUP) in all permanent files on storage

Use Cases

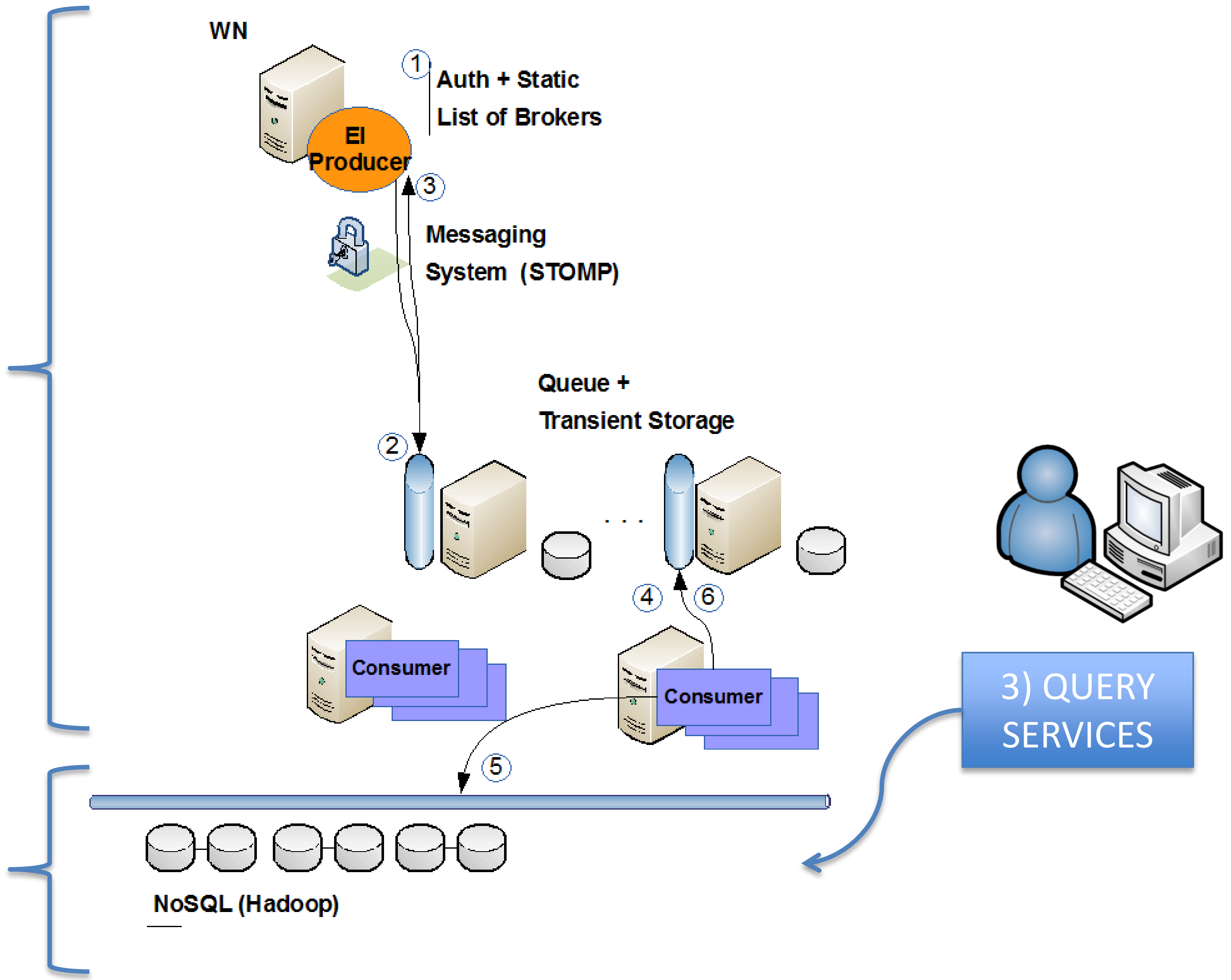
- **Event picking:** Give me the reference (pointer) to "this" event in "that" format for a given processing cycle.
- **Event skimming:** Give me the list of events passing "this" selection and their references.
- **Production consistency checks:** Technical checks that processing cycles are complete.
- **Panda event service (see talk in this session):** Give me the references for the events on this GUID file (to be distributed to HPC or cloud clusters for processing)

Prototype Overview

1) DATA COLLECTION

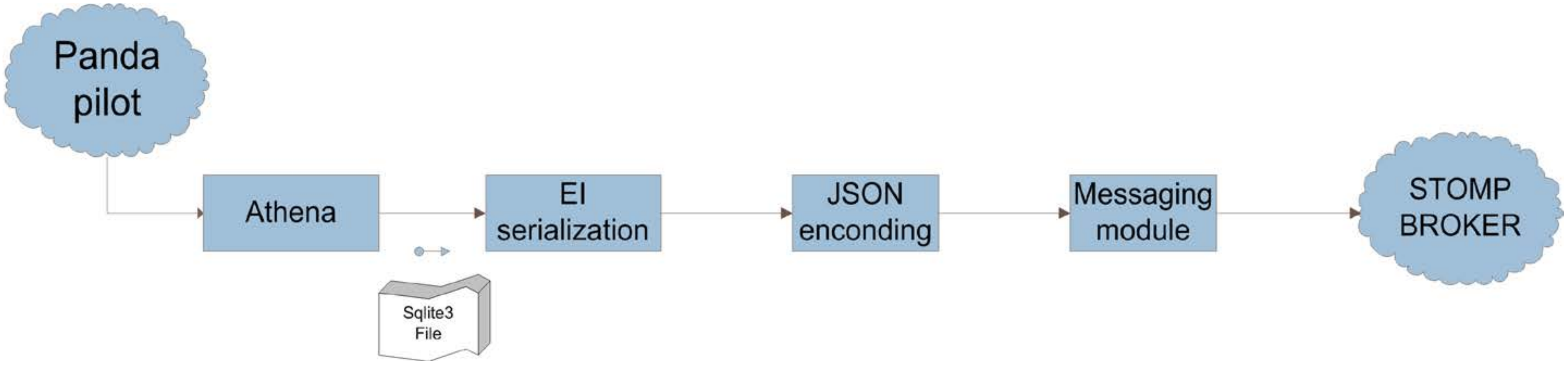
2) HADOOP CORE

3) QUERY SERVICES



Data Collection - Producer

- EventIndex Information is produced in Athena (Atlas software framework) with python scripts, running in Tier0 and GRID sites.



- Current tools include POOL, RAW, TAG formats, and produce the information in a intermediate file to be sent afterwards.
- Production jobs measured last May'2013 give us the daily data to be loaded as 265 K gridjobs/day -> 300 Million events/day -> 1Kb/event. **Means around 300 GB/day.**
- Rate of the Producers is therefore measured as **20Hz** of file records containing **~3.4 kHz** of event records, to be increased the following years up to **80Hz** and **30kHz** of event records.

Data Collection - Messaging

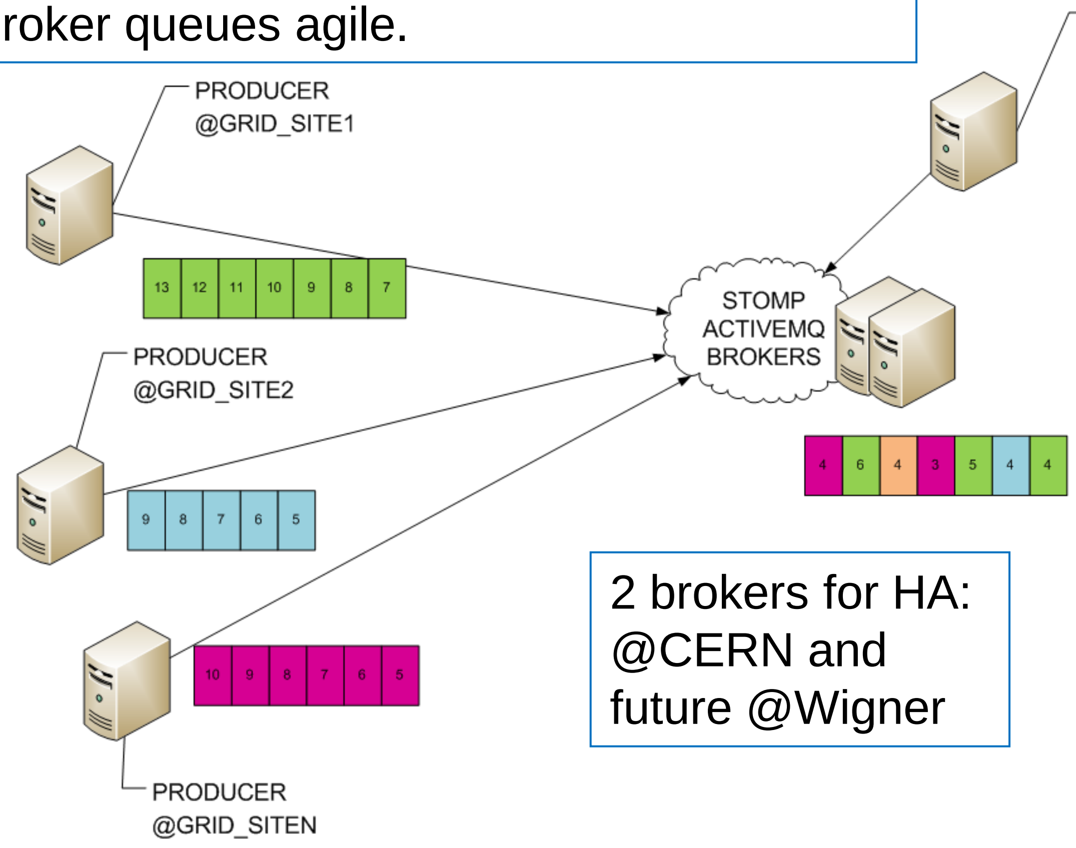
- Collection of the produced Event Info is done with a messaging system based on the STOMP protocol, and ActiveMQ brokers.

Message size is set small 1-10kB to keep broker queues agile.

Atomic transactions on Producers: if connections breaks no partial processing occurs.

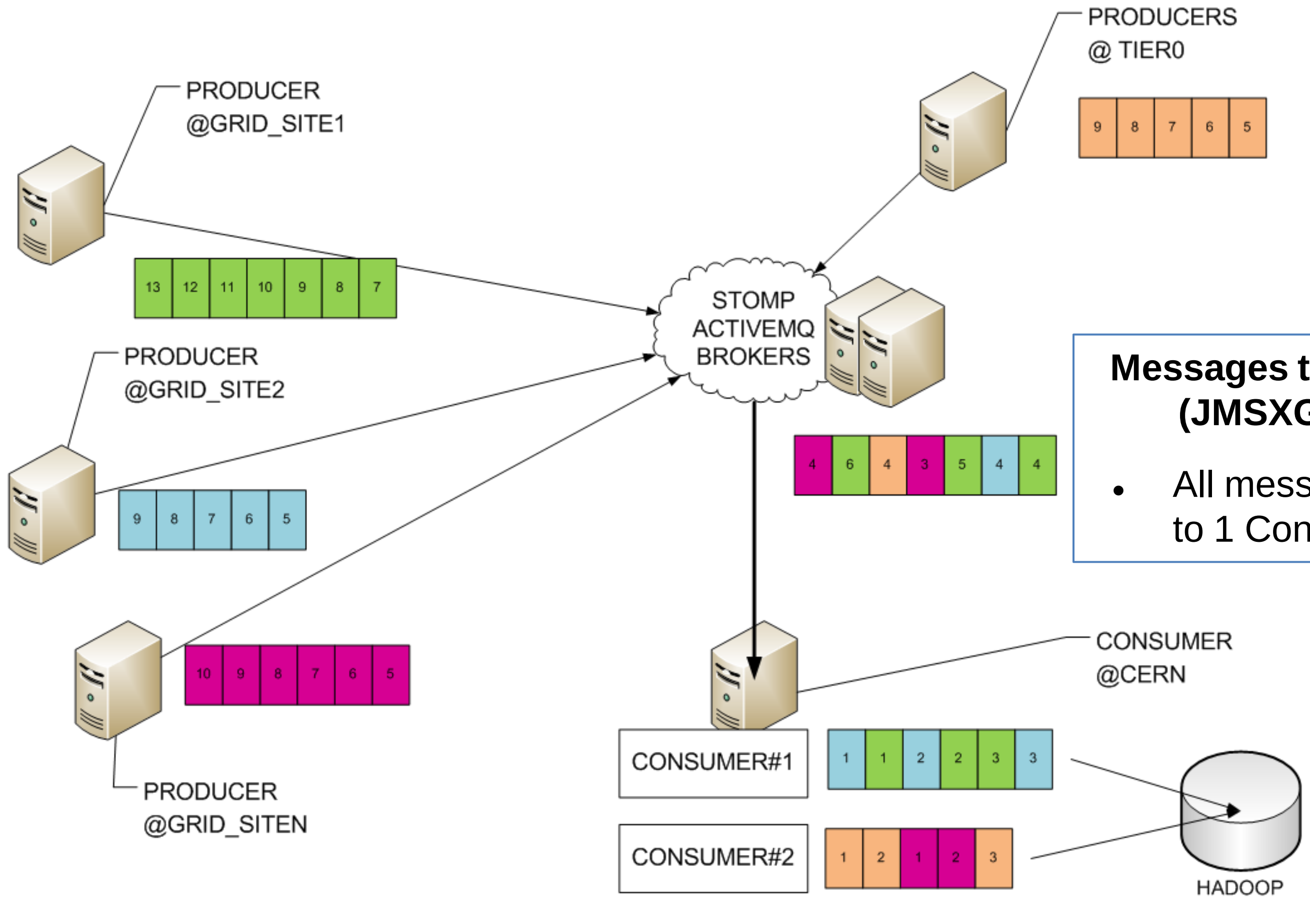
Stomp performance in our benchmarks reaches enough 350 msg/s and 10Mb/s per Producer**.** Measured performance for sending real events reached 200K event/s and 60Mb/s (1Broker, 6 Prod/s, 4 Cons, 50K events/job)

2 brokers for HA: @CERN and future @Wigner



Data Collection - Consumer

- **Consumers** are in charge of retrieving the information from the brokers and insert it into Hadoop.

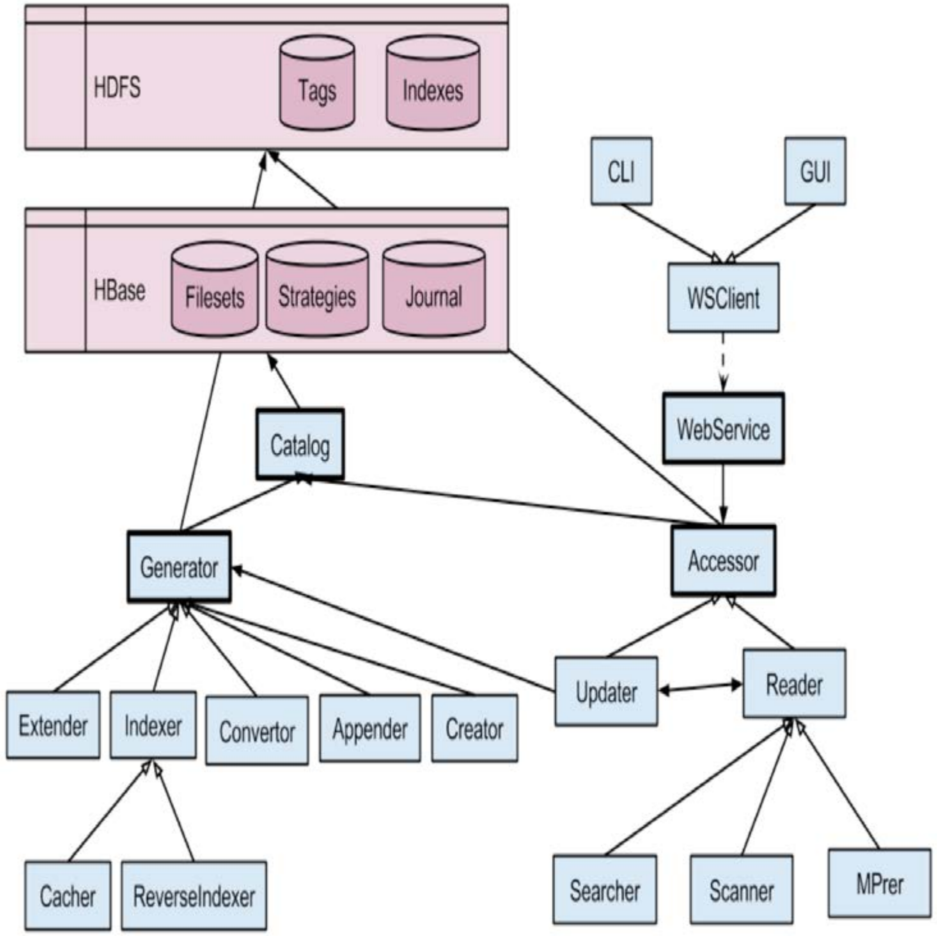


Messages tagged by group (JMSXGroupID)

- All messages from 1 Producer will go to 1 Consumer (while is alive)

Files stored on Mapfile format usable by Hadoop Core Services

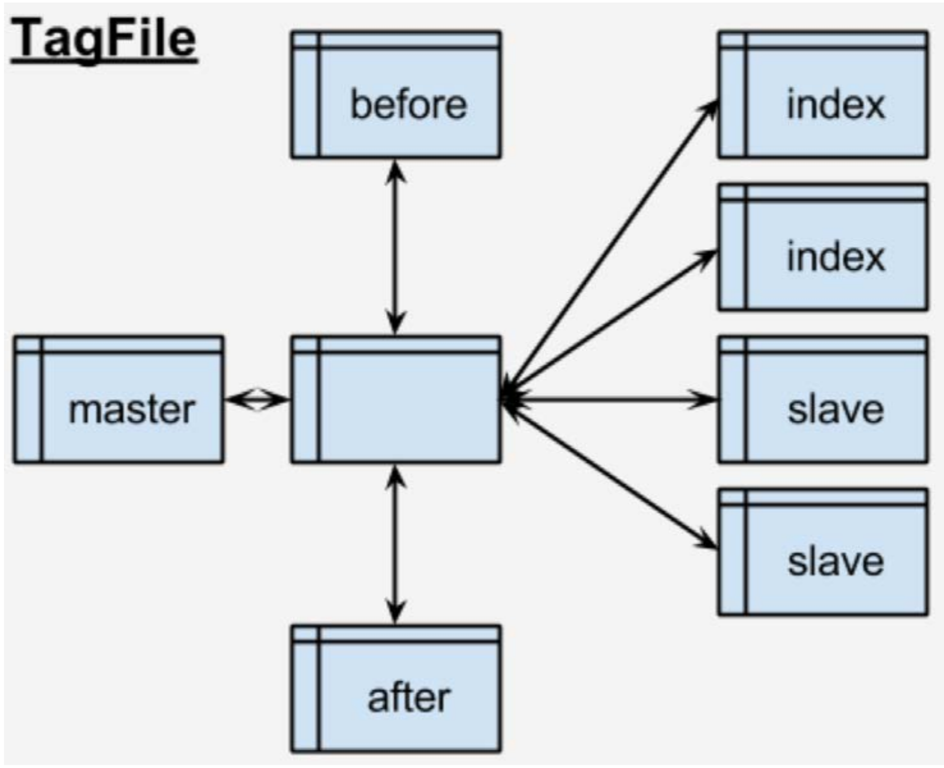
Hadoop Core (1)



- Data uploaded from Consumers is stored in MapFiles files in HDFS. Catalog is stored in Hbase.
- Data can be indexed (e.g. Using inverted indices for trigger info). Index files just have key + references to data files.
  Some index files created at upload, others added. Searches can be performed using keys (immediate results), or on data files.
- Searches can be done with full scans or using MapReduce jobs.

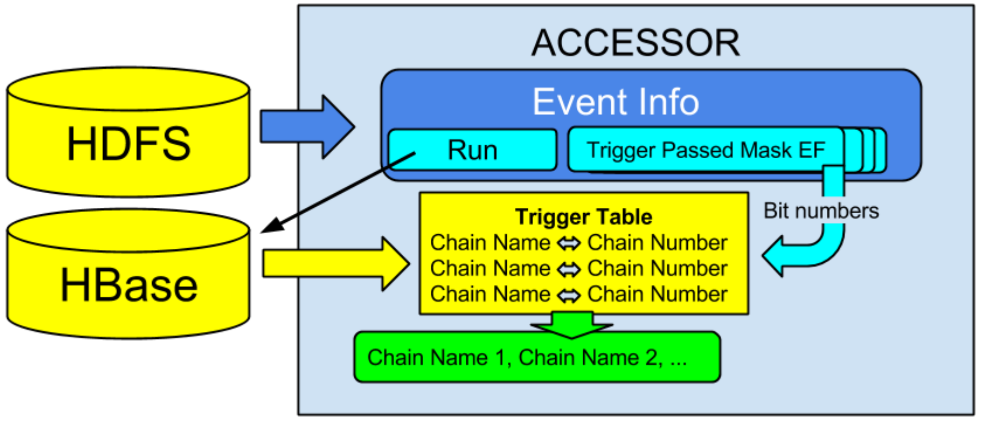
Hadoop Core (2)

- Files in HDFS are logically grouped in collections of filesets, representing event collections.
- Each Fileset contains: a master record, pointers to before /after filesets (vertical partitions), slave filesets (horizontal partitions), and indexes.
- Each file is represented by one entry in a HBASE table. The Catalog checks presence and consistency of all components.



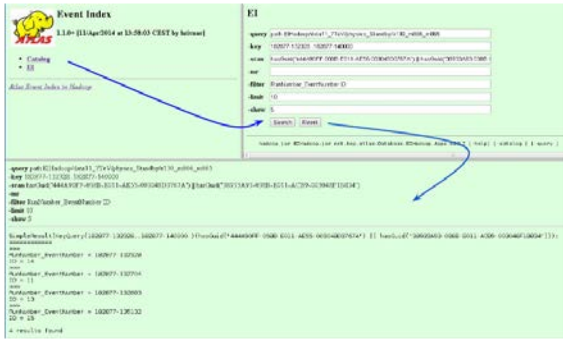
Hadoop Core (3)

- Important Use case: query/count the events that satisfy a given trigger condition, or a combination.
- Trigger chains need to be first decoded by knowing the trigger menu via the run number and finding in the Conditions Metadata (COMA) database the corresponding trigger info for each bit in the trigger words.
- Trigger decoding (get chain names from trigger bits and run number) is also cached to improve the future searches.



ACCESSOR
Event Info
Run
Trigger Passed Mask EF
HDFS
HBase
Bit numbers
Trigger Table
Chain Name ⇔ Chain Number
Chain Name ⇔ Chain Number
Chain Name ⇔ Chain Number
Chain Name 1, Chain Name 2, ...

Query Services



Event Index

1.1.0+ [11-Apr-2014 at 13:58:03 CEST by hrivnac]

- Catalog
- EI

Atlas Event Index in Hadoop

EI

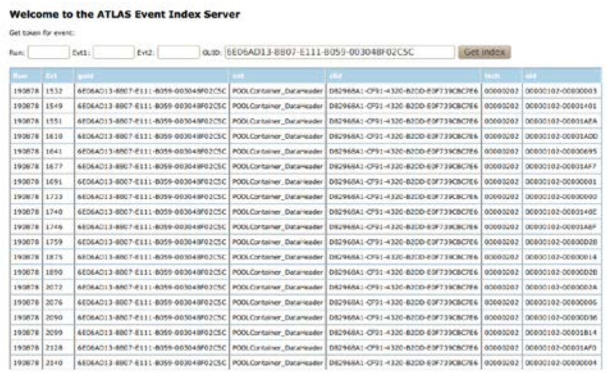
```
-query  path:EI/hadoop/data11_7TeV/physics_Standby/r130_m066_m088
-key    182077-132328..182077-140000
-scan   hasGuid("4444..."
-ar
-filter RunNumber_EventNumber ID
-limit  10
-show   5
```

Search Reset

```
-query path:EI/hadoop/data11_7TeV/physics_Standby/r130_m066_m083
-key 182077-132328..182077-140000
-scan hasGuid("4444A90FF-0500-E011-AE55-003048D37674") || hasGuid("3003A93-0000-E011-AC39-003048F10034")
-ar
-filter RunNumber_EventNumber ID
-limit 10
-show 5

SimpleResult(hepQuery[182077-132328..182077-140000](hasGuid('4444A90FF-0500-E011-AE55-003048D37674') || hasGuid('3003A93-0000-E011-AC39-003048F10034')]);
=============
===
RunNumber_EventNumber = 182077-132328
ID = 14
===
RunNumber_EventNumber = 182077-132704
ID = 11
===
RunNumber_EventNumber = 182077-132003
ID = 13
===
RunNumber_EventNumber = 182077-136132
ID = 15

4 results found
```



Welcome to the ATLAS Event Index Server

Get token for event:

Run: ____ Evt1: ____ Evt2: ____ GUID: 6ED6AD13-8807-E111-B059-003048F02C5C Get index

Run	Evt	guid	cnt	clid	tech	oid
190878	1532	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000003
190878	1549	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001401
190878	1551	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001AEA
190878	1610	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001ADD
190878	1641	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000695
190878	1677	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001AF7
190878	1691	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000001
190878	1733	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000000
190878	1740	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-0000140E
190878	1746	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001A8F
190878	1759	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000D28
190878	1875	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000014
190878	1890	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000D2B
190878	2072	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-0000002A
190878	2076	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000006
190878	2090	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000D36
190878	2099	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001B14
190878	2128	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00001AF0
190878	2140	6ED6AD13-8807-E111-B059-003048F02C5C	POOLContainer_DataHeader	D82968A1-CF91-4320-B2DD-E0F739CBC7E6	00000202	00000102-00000004

A Typical query from the Panda Event Service for all events from a GUID takes ~3-10 secs

- Query services are the interface for users to the Core Hadoop:
  - Command line interface,
  - Web Services with graphical and text/plain output suitable for the Panda Event Service.
- Simple data scan operations on the test dataset (1 TB of 2011 data) using the available Hadoop cluster of 20 nodes take about **15 minutes.**
  - worst case, reads all data
- Queries that give the run number and event number (event picking use case) and use the so-far unoptimised “accessor” method return their result in **3.7 seconds**

Summary and Conclusions

- **The ATLAS experiment needs an EventIndex** to catalogue billions of events taken yearly.
  - Several TBs of catalogue data are needed to index several PBs of event data.
- **Full chain tested successfully** with first prototype:
  - EventIndex production worldwide at different Grid Sites. Messaging system can amply cope with the required rates for the following years.
  - Hadoop core tested with previously loaded 1TB of 2011 data, and data coming from the data collection.
  - Query services to satisfy the use cases. Results are encouraging, and performance suitable to improve.
- **Next steps are:**
  - Automate procedures.
  - Load all Run1 data.
  - Tune the system.
- According to the original plan and the current experience the system will be fully working at the start of 2015, ready for the new LHC data.