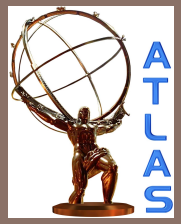


THE ATLAS DISTRIBUTED DATA MANAGEMENT SYSTEM & DATABASES

Vincent Garonne, Mario Lassnig, Martin Barisits, Thomas Beermann, Ralph Vigne, Cedric Serfon

Vincent.Garonne@cern.ch
ph-adp-ddm-lab@cern.ch

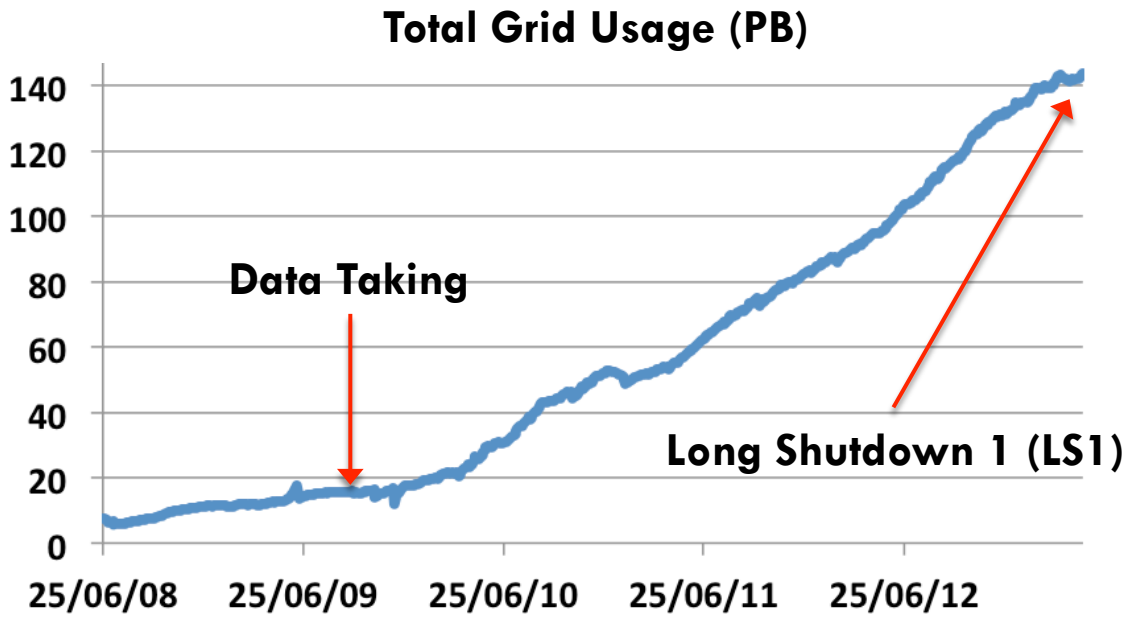
XLDB 2013 Workshop Europe

Overview

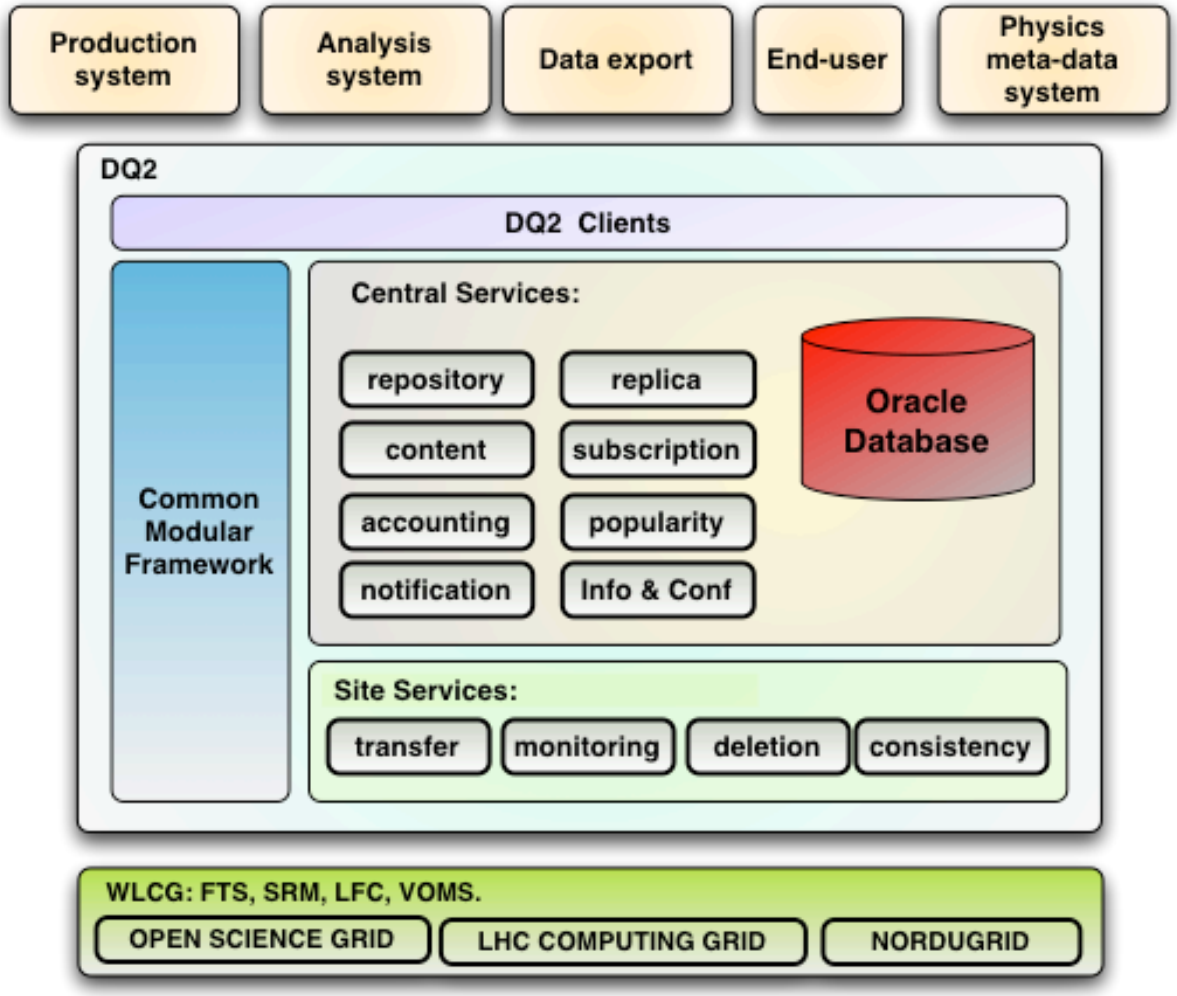
- Distributed Data Management (DDM)
  - Overview
  - Architecture
- Relational Database Management Systems (RDBMS) & DDM
  - Use cases & Experiences
- NoSQL & DDM
  - Use cases & (operational) Experiences
- Present & Future: Rucio
- Conclusions

DDM Background

- The Distributed Data Management project manages ATLAS data on the grid
- The current system is Don Quijote 2 (DQ2)
  - 144 Petabytes
  - 600k datasets
  - 450 million files
  - 800 active users
  - 130 sites
  - + History



DDM Implementation: DQ2



DQ2

- In production since 2004
- 2007-2011: Many updates

RDBMS – Oracle

- Critical dependency
- Proven technology
- Expertise@cern
- Great for enforcing data integrity
- Tool of choice for Online transaction processing applications (OLTP)

Oracle & DDM

- Constant optimization over the years has allowed us to scale
  - avg. 300 Hz, physical write 7 MB/s, physical read 30 MB/s
- Improvement with I/O, concurrency, front-end deployment, time partitioning, table space management, de-normalization for performance, etc.
- Developing and maintaining an high availability service with Oracle requires some good knowledge and expertise
  - Bind variables, index hint, oracle optimizer, pl/sql, execution plan, etc.

Concepts

- RDBMS
  - Vertical scalability ("scale up")
  - Few powerful nodes
  - Shared state
  - Explicit partitioning
  - Resistant hardware
  - ACID
  - Implicit queries (WHAT)
- NoSQL (Structured storage)
  - Horizontal scalability ("scale out")
  - Lots of interconnected low cost nodes
  - Shared nothing architecture
  - Implicit partitioning
  - Reliability in software
  - BASE
  - Explicit data pipeline (HOW)

NoSQL & DDM

- Data warehousing use cases & applications relevant for NoSQL
  - Lot of data
  - No transactions and relaxed consistency
  - Multi dimensional queries
- Three technologies evaluated: **MongoDB**, **Cassandra** & **Hadoop**
  - Many more available, but these were chosen with the following things in mind
  - Large community available and widely installed
  - In production in several larger companies with respectable data sizes
  - Potential commercial support
- 12 node cluster located in CERN IT data center to evaluate technologies
  - 96 CPU cores (Intel Xeon, 2.27GHz, 8/node)
  - 288 GB RAM (24/node)
  - 24 TB space (1 TB SATA each, 2/node)
  - 1 GigE network

Data Models


```
{
 _id: 'Main Account User',
 groups: ['group_a',
          'group_b',
          'group_c'],
 selections: {
              'select_a': 123,
              'select_b': abc
             }
}
```

- Explicit row-key
- Native datatypes
- Everything indexable


```
'Groups' : {
    'Main Account User' : {
        'groups' : ['group_a',
                    'group_b',
                    'group_c']
    }
}

'Selections' : {
    'Main Account User' : {
         'select_a' : 123,
         'select_b' : abc
    }
}
```

- Implicit row-keys
- Data is byte streams
- Column Families group row-keys


```
'Main Account User' : {
    'Groups' : {
         'groups' : ['group_a',
                     'group_b',
                     'group_c']
     },
     'Selections' : {
         'select_a' : 123,
         'select_b' : abc
     }
}
```

- Implicit row-key
- Data is byte streams
- Row-keys group Column Families
- Row-keys are sorted

Architectures

- Master/Slave
  - Smart client implements failover
- Write-ahead log
- Limited MapReduce
  - interleaved
  - bound to single thread
- Keyed binary storage
- Indexes
- Table locking
- Replica sets
- Explicit partitioning

- No single point of failure
  - ring of nodes
  - forwarding of requests
- Write-ahead log
- No MapReduce
  - can use Hadoop
- No file storage
- Bloom filter
- Row locking
- Snapshotting
- Implicit partitioning

- No single point of failure
  - multiple masters
- Write-ahead log
- MapReduce
- File storage
  - Data on HDFS
  - Can be used as a source and sink within Hadoop
- Bloom filter
- Row locking
- HDFS-backed redundancy
- Implicit partitioning

Technology Selection

	MongoDB	Cassandra	Hadoop/HBase
Installation/ Configuration	Download, unpack, run	Download, unpack, configure, run	Distribution, Complex config
Buffered read 256	250'000/sec	180'000/sec	150'000/sec
Random read 256	20'000/sec	20'000/sec	20'000/sec
Relaxed write 256	10'000/sec	19'000/sec	9'000/sec
Durable Write 256	2'500/sec	9'000/sec	6'000/sec
Analytics	Limited MapReduce	Hadoop MapReduce	MapReduce, Pig, Hive
Durability support	Full	Full	Full
Native API	Binary JSON	Java	Java
Generic API	None	Thrift	Thrift, REST

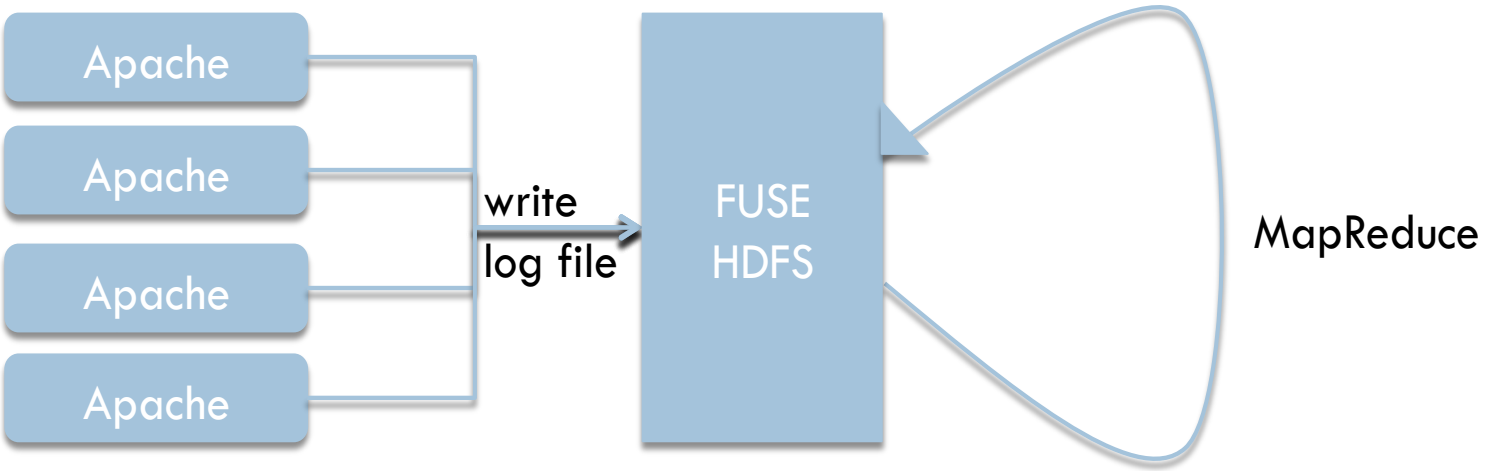
Technology: 

	MongoDB	Cassandra	Hadoop/HBase
Installation/ Configuration	Download, unpack, run	Download, unpack, configure, run	Distribution, Complex config
Buffered read 256	250'000/sec	180'000/sec	150'000/sec
Random read 256	20'000/sec	20'000/sec	20'000/sec
Relaxed write 256	10'000/sec	19'000/sec	9'000/sec
Durable Write 256	2'500/sec	9'000/sec	6'000/sec
Analytics	Limited MapReduce	Hadoop MapReduce	MapReduce, Pig, Hive
Durability support	Full	Full	Full
Native API	Binary JSON	Java	Java
Generic API	None	Thrift	Thrift, REST

Hadoop is a framework for distributed data processing (not only a database) with many components: **HDFS** (distributed filesystem), **MapReduce** (distributed processing of large data sets), **HBase** (distributed data base for structured storage), **Hive**(SQL frontend), **Pig**: data-flow language for parallel execution, …

Use Cases : Log File Aggregation

- HDFS is mounted as a POSIX filesystem via FUSE
  - Daily copies of all the ATLAS DDM log files are aggregated in a single place
  - 8 months of logs accumulated ~ 3 TB of space on HDFS
- Python MapReduce jobs analyse the log files
  - Streaming API: read from stdin, write to stdout
- Processing the data takes about 70 minutes
  - Average IO at 70MB/s
  - Potential for 15% performance increase if re-written in pure Java
    - Better read patterns and reducing temporary network usag



Use Cases : Trace Mining

- Client interaction with ATLAS DDM generates traces
  - E.g., downloading a dataset/file from a remote site
  - Lots of information (25 attributes), time-based
  - One month of traces uncompressed 80GB, compressed 25GB
    - Can be mapreduced in under 2 minutes
- Implemented in HBase as distributed atomic counters
  - Previously developed in Cassandra
  - At various granularities (minutes, hours, days)
  - Size of HBase tables negligible
  - Average rate at 300 insertions/s
- Migrated from Cassandra within 2 days
  - Almost the same column-based data model
  - Get extra Hadoop benefits for free (mature ecosystem with many tools)
  - The single Cassandra benefit, HA, was implemented in Hadoop

Use Cases : DQ2Share

- HTTP cache for dataset/file downloads
  - Downloads via ATLAS DDM tools to HDFS, serves via Apache
  - Get all the features of HDFS for free, i.e., one large reliable disk pool

DQ2Share ddo.000001.Atlas.Ideal.DBRelease.v0701010 Get dataset and files

Search results: **ddo.000001.Atlas.Ideal.DBRelease.v07010104** Displaying 20 on 84 files

Type	Name	Size of all files (Bytes)	Number of files	Date	In the Cache
Dataset	ddo.000001.Atlas.Ideal.DBRelease.v07010104	1 754 037 441	84	2009-07-02 02:18:11	.tar: ✗

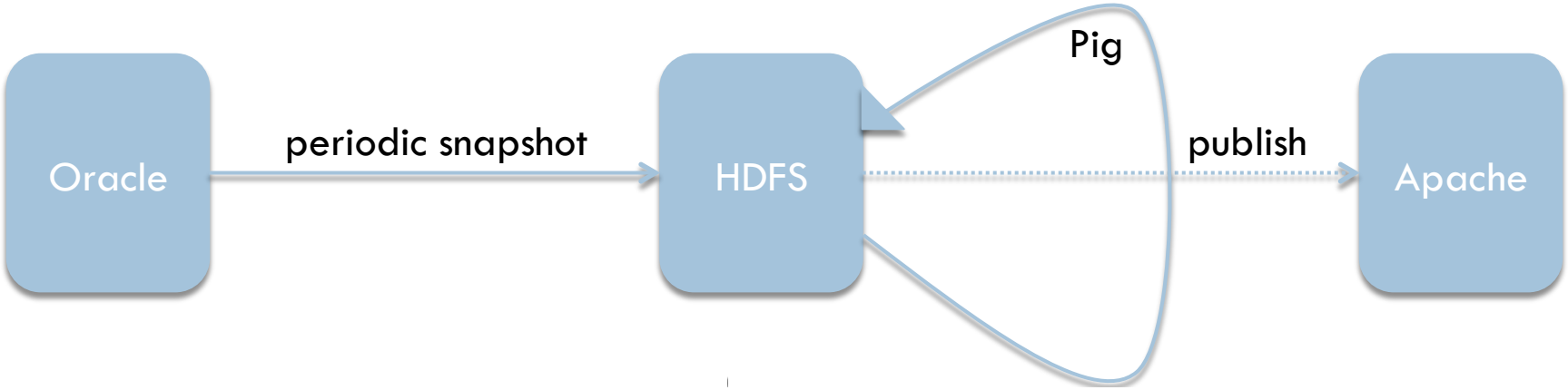
wget 19%...	Type	Name	Size (Bytes)	Checksum	In the Cache
☐	File	07010104._0121364.tar.gz	12 446 534	ad:b05542bb	✗
☐	File	07010104._0120760.tar.gz	12 183 186	ad:8bd6b6d3	✓
☐	File	07010104._0121457.tar.gz 100%	13 654 310	ad:2aee4fcc	✓
☐	File	07010104._0120884.tar.gz	11 927 416	ad:654e6578	✓
☐	File	07010104._0121416.tar.gz	18 179 853	ad:ed7f5d07	✗
☐	File	07010104._0121064.tar.gz	12 591 388	ad:a5464922	✓
☐	File	07010104._0120808.tar.gz	12 181 040	ad:203c2a15	✓
☑	File	07010104._0121198.tar.gz	13 648 670	ad:1afc7773	✓
☑	File	DBRelease-7.1.1.4.tar.gz	693 005 997	ad:3db1645e	✓
☑	File	07010104._0120713.tar.gz	11 958 239	ad:fa3b23a6	✗
☐	File	07010104._0120852.tar.gz	12 246 473	ad:a0d72443	✓
☐	File	07010104._0121412.tar.gz	12 455 380	ad:32a180f0	✗
☐	File	07010104._0121226.tar.gz	12 559 921	ad:d06936f5	✗
☐	File	07010104._0121414.tar.gz	12 461 831	ad:b8e44e10	✗

Use Cases : Accounting

- Break down usage of ATLAS data contents
  - Historical free-form meta data queries

```
{site, nbfiles, bytes} := {project=data10*, datatype=ESD, location=CERN*}
```

- Non-relational periodic summaries
- A full accounting run takes about 8 minutes
  - Pig data pipeline creates MapReduce jobs
  - 7 GB of input data, 100 MB of output data



Operational experiences: Hadoop

- Cloudera distribution
  - Tests and packages the Hadoop ecosystem
  - Straightforward installation via Puppet/YUM
  - But the configuration was ... not so obvious with many parameters, extensive documentation, but bad default performance
- Disk failure is common and cannot be ignored
  - Data centre annual disk replacement rate up to 13% (Google & CMU, 2011)
  - Within one year we had:
    - 5 disk failures (20% failure rate!) Out of which 3 happened at the same time
    - 1 Mainboard failure together with the disk failure, but another node
- Worst case scenario experienced up to now: 4 nodes out of 12 dead within a few minutes
  - Hadoop Reported erroneous nodes, blacklisted them and re-synced the remaining ones
  - No manual intervention necessary
  - Nothing was lost (but one node is still causing trouble, RAID controller?, cluster not affected)

The Next DDM Version: Rucio

- DQ2 will simply not continue to scale
  - Order(s) of magnitude more data with significant higher trigger rates and event pileup (after LS1)
  - Extensibility issues
  - Computing model and middleware changes
- Rucio is the new major version, anticipated for 2014, to ensure system scalability, remove unused concepts and support new use cases
  - Rucio exploits commonalities between experiments (not just LHC) and other data intensive sciences (e.g., Cloud/Big data computing)
- Rucio & Databases
  - Relational database management system
    - SQLAlchemy Object Relational Mapper(ORM) supporting Oracle, sqlite, mysql, …
    - Use cases: Real-time data and transactional consistency
    - Test with representative data volume is ongoing
    - Evaluation of synchronization strategies between DQ2 and Rucio
  - Non relational structured storage (Hadoop)
    - Use cases: Search functionality, realtime stats, monitoring, meta-data, complex analytical reports over large volumes

Conclusions

- ATLAS Distributed Data Management delivered a working system to the collaboration in time for LHC data taking relying heavily on Oracle
- NoSQL: DDM use cases well covered
  - Hadoop proved to be the correct choice: Stable – reliable – fast – easy to work with
  - We see Hadoop complementary to RDBMS, not as a replacement
- DDM team is currently developing and (stress-)testing a new DDM version called Rucio anticipated for 2014
  - Using both SQL and NoSQL

Thanks !

http://rucio.cern.ch