CHEP 2018
23RD INTERNATIONAL CONFERENCE ON
COMPUTING IN HIGH ENERGY AND NUCLEAR PHYSICS
9-13 July 2018
National Palace of Culture
Sofia, Bulgaria

The challenges of mining logging data in ATLAS

The University of Texas ARLINGTON

Elizabeth Gallas, Nurcan Ozturk
on behalf of the ATLAS Collaboration

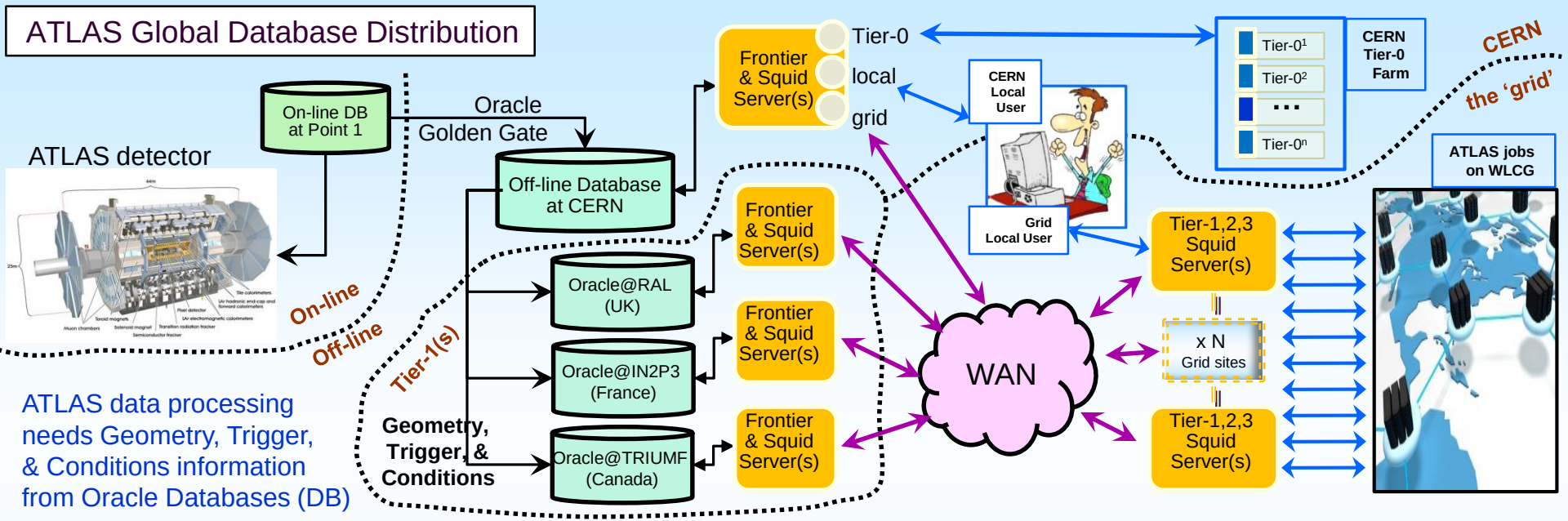
UNIVERSITY OF OXFORD
DOMINVS ILLVMINATIO MEA

CHEP 2018, Sofia Bulgaria
July 9-13, 2018

Outline

- Introduction(s):
  - The ATLAS experiment @ the LHC/CERN
    and the current database deployment strategy
  - Overview of ATLAS processing,
    focussing on its need for Database-resident data.
- Project goals and scope
  - Overview of clients and servers in global database distribution
    - How the data are provided and how that usage is logged and monitored
  - Big Data sources, their connections, and limitations
    - Plans for consolidating key aspects in critical areas
  - Features & findings so far
  - Open questions and Future plans
- Summary and Conclusions

ATLAS Global Database Distribution



ATLAS data processing needs Geometry, Trigger, & Conditions information from Oracle Databases (DB)

ATLAS Database distribution (above) includes Frontier & Squid servers, offering a buffer between client tasks and primary DB sources

- Known benefits of Frontier/Squid deployment:
  - Queuing of requests (distributing and levelling the load) and
  - Caching results of identical queries from clients (reducing the direct load on the databases)

The problem:

With an increasing number and diversity of clients, components of the system are strained, occasionally to the breaking point.

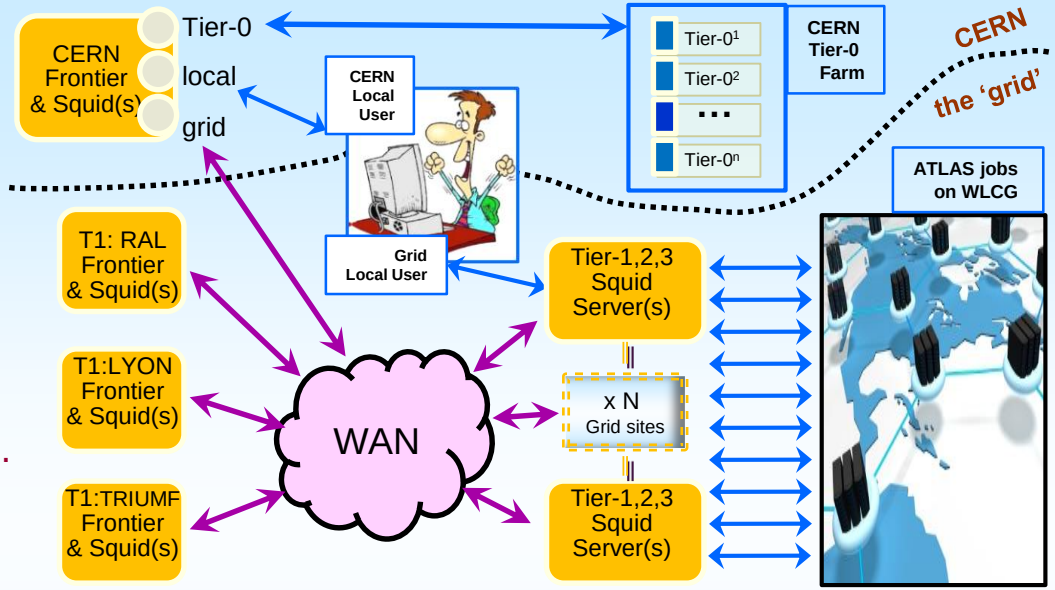
The goal:

Understanding the DB usage patterns of clients will help us globally tune the distribution infrastructure and help us work with clients to refine their requests.

Project Scope

We focus on:

- Offline processing,
  both at CERN and on ‘the grid’
  The right hand side of the previous diagram
- “Conditions DB” access:
  where “**Conditions**”: calibration, alignment, etc. specific to the time range of the data processed
  - It dominates overall DB usage and has found to be associated with degradation and failures.



We have a diversity of clients:

- Mainly event processing: clients are deploying jobs based on **Athena**, the ATLAS software framework
  - CERN-based Tier-0 farm
  - Grid-wide event processing
  - Users running jobs on local machines
- But also: Users running non-Athena-based systems or development which also require information from the database

- To understand client DB usage, we need to assess
  - the request and response between the client (a job) and server (Frontier/Squid): double-headed arrows (above)
    → Over the WAN ⟷ & Over local networks ⟷
    - This is recorded in Frontier and Squid logs
  - but also: what the client is requesting (and why)
    - This is recorded in job log files or
    - Communication with individual users

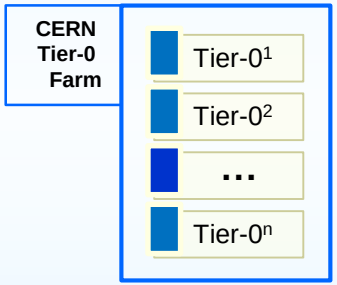
"Client-side" (jobs requiring DB access)

Tasks generally deploy many jobs running over the event files of the dataset being processed

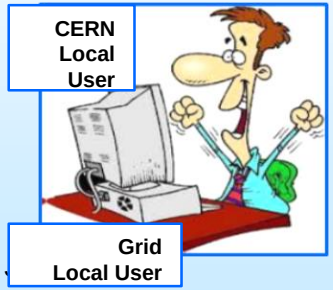
- Each job typically executes a few sequential stages of processing, each needing DB information
  - Each stage will write a separate Athena log file

Client DB access is evaluated by analysing these **Athena log files** (when available) to look for anomalies

- **Grid-wide event processing**
  - Athena-based tasks deployed via **PanDA** (ATLAS system for distributed event processing)
    - MC simulation, data reprocessing, and most user and group processing
  - **Athena logs**
    - **readily available via PanDA**



- **CERN-based Tier-0 farm**
  - Always Athena-based tasks:
    - Mainly performing prompt reconstruction and first-pass processing of newly recorded data
    - Exceptionally also MC production
  - **Athena logs**
    - **Initially kept on EOS**
    - **Eventually archived on CASTOR**



- **Users running jobs on local machines**
  - Generally Athena-based tasks but also non-Athena-based development which may require DB information
    - has greater potential for anomalous data requests (not regulated by Athena-based methods)
  - **Athena logs or use cases must be collected from the user**

A job's DB access and associated metrics are recorded in its Athena log file by the Athena-based "**IOVDbSvc**"

which negotiates database access and makes the data available to the job (prompted by the job's requests for data)

The information includes summaries of volume, rows, channels, and latency

- per schema (detector subsystem) and
- per folder (tables of conditions data)

and warnings about which data were retrieved but never used by the job

- i.e. wasted queries

But there > 100,000 jobs running per day

Which log files should be analysed ?

"Server-side" (Frontier/Squid)

Grid or CERN-based tasks need DB information

They never make direct Oracle DB connections, but make requests to their local Squid server

→ If results are not cached, the request is passed along a chain of Squids to a **Frontier Launchpad**

→ The Launchpad queries the DB if needed

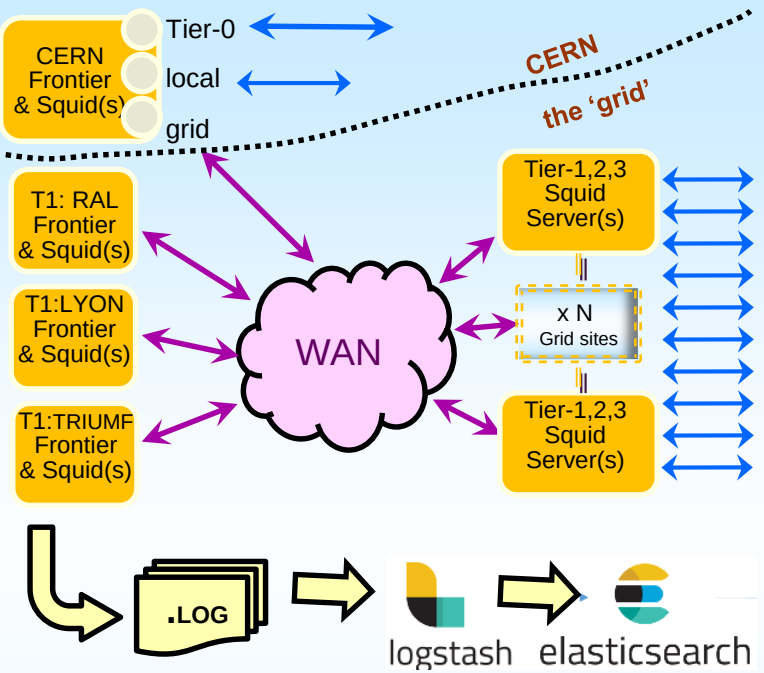
Globally, launchpads handle >10M actions/day

- They return results back to clients via squids

Both launchpads and squids log transactions

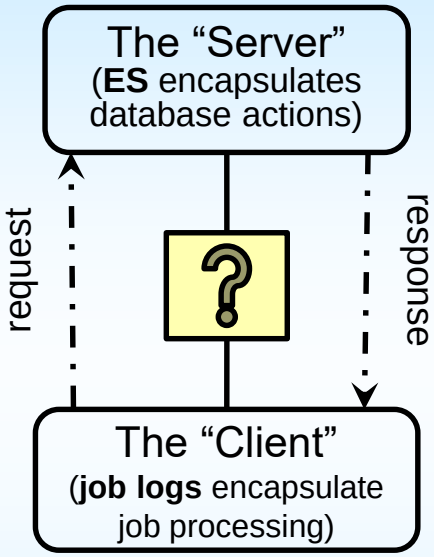
- But these logs are huge !
  Previously: logs were kept by sites for only a few days
  → Accessible only at the site … and difficult to interpret
- But thanks to the efforts of experts, essential information
  → Task and Job identifiers, SQL query, rows, data volume, transaction and query time, various status flags, etc.

  is now extracted into Elastic Search (ES) repositories !
  - ES and associated visualization tools (Kibana, …) :
    Give us a wide angle view of throughput as well as the ability to look at individual transactions



See CHEP **Poster #191** for details !
Elastic Search cluster:
@University of Chicago
Data from Frontier Launchpads
@INFN Roma
Data from Squids@Roma, Tier-1 Lyon

“Client” and “Server” in DB access studies



These ‘Big Data’ sources have been extremely useful:

1. Frontier and Squid logs:
   - A. Elastic Search cluster at the University of Chicago
     - logging from primary Frontier Launchpads (CHEP poster #191)

     Also ES@Chicago email alerts: relay anomalous states for investigation.
   - B. Elastic Search cluster at INFN Roma
     - logging from select Squid sites: Roma and the Tier-1 site at LYON

   Information may include job and task identifiers (if using PanDA)

   → If so, this helps to directly find jobs logs via PanDA

   → If not, it may contain sufficient client connection information to help to find users operating locally
2. Athena job logs
   - These are generally obtained via PanDA, but may also be collected from individual users or groups running Athena outside PanDA (usually locally at CERN) in our attempts to understand and improve their database access workflow

But neither source tells all … what more do we need …

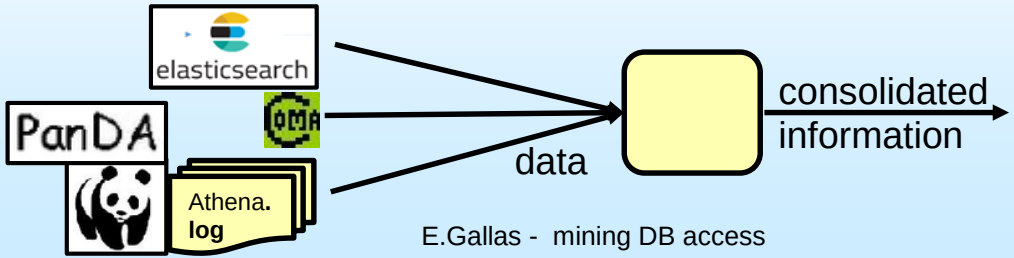
Pros and Cons: filling gaps with auxiliary information

- Each of these sources operates in relative isolation
  - Elastic Search repositories
    - contain individual queries, performance metrics, client Task and Job ID (if PanDA),
    - but they may not contain all the queries of any job and has no knowledge of what the job is doing
  - Athena job log IOVDbSvc messages
    - record DB connections and summaries of metrics for folders accessed
    - but they don't contain the queries themselves, know if slow performance is due to queries themselves, nor is it aware that some of its queries were unnecessary until the end of the job.
  - → and neither source will retain logs forever

These "Big Data" sources are complementary, with considerable unique and essential information to form micro- and macroscopic views of usage patterns and causes of degradation in the infrastructure.

To further enhance the data, we are developing

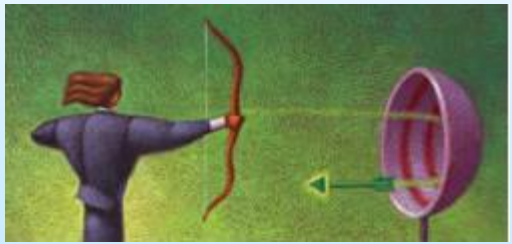
- methods for automated job log collection from PanDA for usage patterns of interest
- methods using COMA (collected metadata about the Conditions DB) to decode elements of SQL queries into the subsystems and folders targeted by the job and supplement GUI reports



Progress and Findings

Studies to date have found:
In general, at least some DB access by most jobs is completely unnecessary … remnants of schema design, software evolution, complacency, and habit:

- Schema design tends to be optimized for storage rather than read-access
  - Some of this is inherent in the design & flexibility of the LCG Conditions DB infrastructure (COOL)
  - Subsystem experts are needed here for folder redesign.
- The same jobs deployed in different ways where DB access should be identical are not
  - Such as standard Athena and AthenaMP production (using single or multi-core processing)
    - Core software expertise is needed to debug in these cases.
- Both users and experts tend to build job configurations based on something that works
  - e.g. Simpler forms of processing do not need the same information as full reconstruction
- Users are happy when jobs produce the expected output in good time
  - Many are unaware of whether their DB access is optimal, or not.
  - The absence of oversight and assessment tools contribute to this.
- In the past, the database deployment infrastructure was operating well below capacity
  - Queries in excess of requirements were not noticed.



Current status

Studies so far have focussed on investigations during key incidents

- Dataflow and cache efficiency during dedicated stress tests,
- Looking at details of queries and response based on alerts from ES monitoring
  - Times of high load and
  - Particular tasks noted in alerts

In general, these studies have exposed issues in workflow or configuration problems:
Things which are not expected / supposed to happen, but do !

- Considerable time is required to collect sufficient information and work with clients on how best to eliminate such usage.
- It is useful to find solutions to fix issues found, which eliminate future occurrences in the logs (and the associated load).

These studies have only scratched the surface
but showed the ES (Elastic Search) repositories are key to gain:

- A microscopic view of components of that data flow
  - With information to trace it to specific client processes
- A macroscopic view of global DB network data flow

In progress: Automated tools for consolidating data
and reporting information about DB access patterns.

- Design: based on the experience gained with manual probes of the data so far.

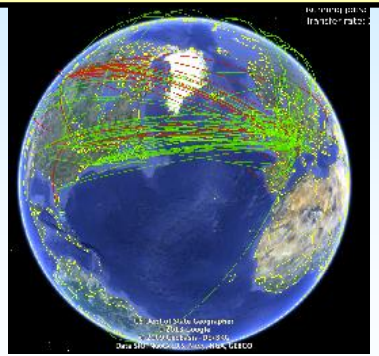
Open questions

- Diversity in types of ATLAS jobs has grown over the years.
  - Collecting DB access patterns by job type will give insight into
    - which job types are most expensive in terms of DB access ?
    - which schemas and tables are accessed most often (or not at all) ?
- Job deployment at sites is also variable:
  - single vs multi-core processing
  - single vs multi-threaded processing

  DB access patterns can differ in these scenarios:
    - Understanding how and why will help to optimize their DB access
- On Conditions DB access
  - How effective is the caching in various scenarios ?
  - Is the IOVDbSvc alignment of queries working as expected ?
  - What is the variation in DB access load with
    - Conditions folder structure, channels, rows, blobs ?
    - Data volume (both large and small) ?
- What are the implications for future
  - Conditions DB design (CHEP#192) ? and
  - ATLAS Software design (CHEP#117) ?

Summary

A happy and healthy grid needs a better regulated diet of data from databases.



Collecting Frontier/Squid logs into ES enables the monitoring of global DB access throughput and load (CHEP #191) to

- identify failing sites and
- inform when tasks should be throttled or killed.

This project (CHEP #189) studies these problematic (and other) tasks using information in ES along with other data sources to probe more deeply into the associated clients to

- refine the access by clients,
- tune for coherence between clients (improve cache efficiency), and
- improve the overall deployment of the database distribution infrastructure

Studies using this data so far have shown we must

- **log**, **study**, and **report** our findings in coherent ways to form the motivation to influence clients and tasks for future operations:
  - Run 2 optimization of conditions data access (CHEP #124) and
  - Run 3 evolution of the Conditions DB infrastructure (CHEP #192) and conditions data handling in the multithreaded ATLAS framework (CHEP #117).