ATLAS developed tools for OI

Ilija Vukotic

University of Chicago

HOW, JLab, March 21st 2019

Overview

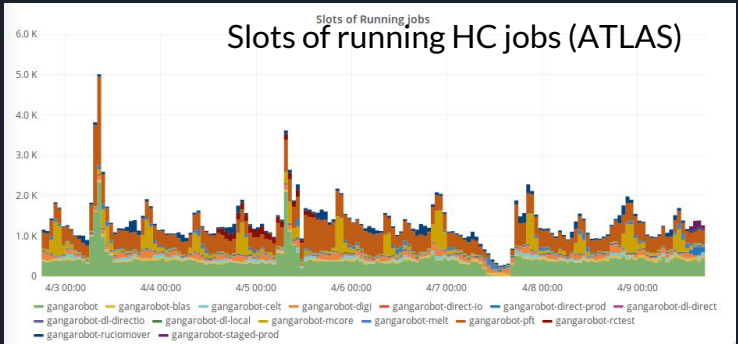
- Frameworks
  - HammerCloud
  - Alarm & Alert
  - ML platform
  - GATES (in dev)
- Tools
  - Anomaly detection in time series (for PerfSONAR)
  - C3P0 - dataset popularity prediction
  - CPU performance measurement
  - Frontier analysis
  - Error classification (in dev)
  - FTS - transfer completion time prediction
  - Panda Time-To-Complete prediction

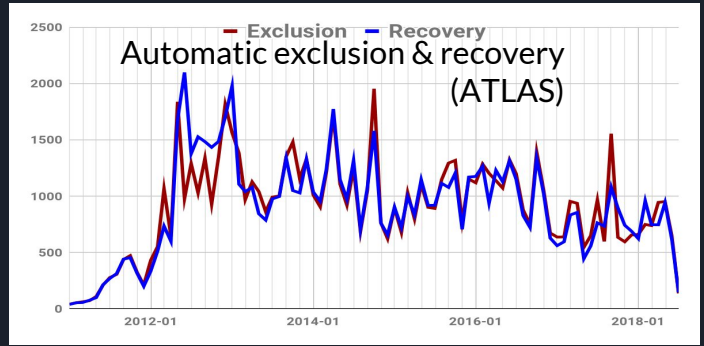
Frameworks - HammerCloud

- A WLCG testing service and framework to ***commission***, run ***continuous tests*** or on-demand large-scale ***stress tests***, and ***benchmark*** computing resources and components of various distributed systems & ***prototype ideas to evolve*** our systems, with realistic ***full-chain experiment workflows***
- **ATLAS:** functional testing & ***auto-exclusion of resources***; EventService blacklisting; commissioning of new resources; commissioning of new components of distributed computing systems (Rucio, Pilot, new data access protocols, ...); functional testing of services (ObjectStore testing); smoke testing of SW config tools (ALRBdevel testing)
- **CMS:** functional testing; commissioning of new resources; commissioning of new components of distributed computing systems
- **CERN IT Batch:** commissioning of resources & CI/CD, containers usability, ...
- **Future computing and R&Ds:** WLCG Data Lake & WLCG DOMA, ESCAPE, ...

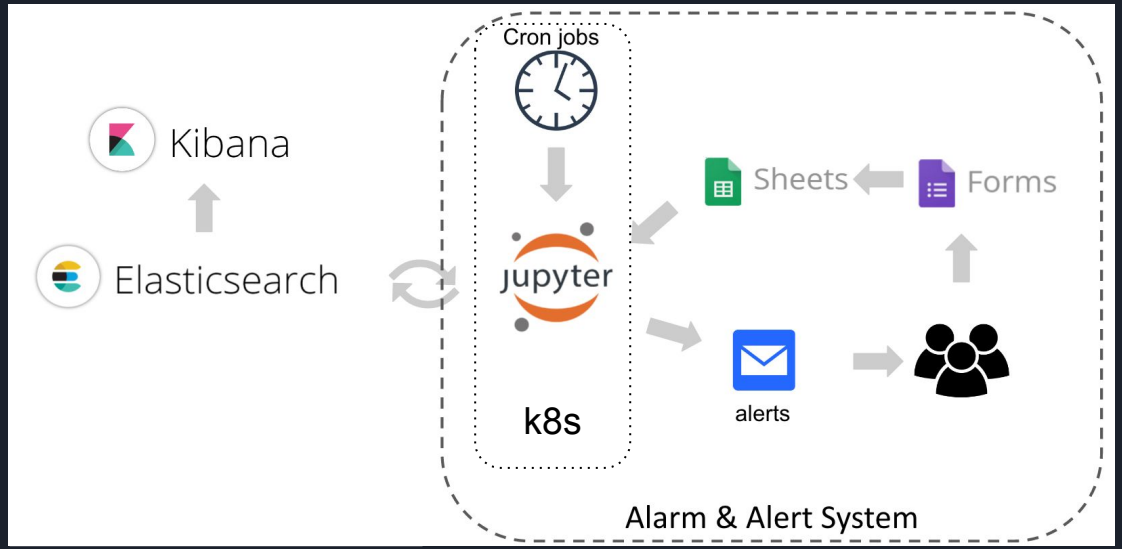
Frameworks - HammerCloud #2

- ***HammerCloud is at the core*** of ***ATLAS*** & ***CMS*** & ***Batch*** distributed computing ***operations***
- ATLAS: ***automatic exclusion & recovery*** of compute resources, based on job failures count
- ***Evolution***
  ⇒ how can we improve automatic blacklisting with ML?
  ⇒ enlarge family of ***tested & commissioned resources***, ***commission new components*** of / ***prototype ideas*** to evolve our tools, already now ***capable to test*** with ***workflows of experiments beyond HEP***!





Frameworks - Alarm & Alert



Good

- Flexible
- Easy to change
- Easy to (un)subscribe :)
- 2.5-3k mails per month
- Almost zero support

Could improve

- Make it experiment agnostic
- Nicer API
- Support much more complex evaluations - from Spark or any other type of Jobs
- Have a library of frequently used algorithms for anomaly detection/classification etc.

ML platform

https://www.atlas-ml.org/

- Nodejs, expressjs site running on k8s. Has full control of k8s deployments.
- Uses Globus authentication.
- Can limit resources per instance.
- Extensible
  - Currently offers: private and shared JupyterLab instances and Spark Job submissions.
  - To come: TFAAS
- Easy to deploy elsewhere.
  - ML workshops
  - Teaching computational physics courses

Could improve:

- shared storage
- federate backend resources
- more options - CVMFS

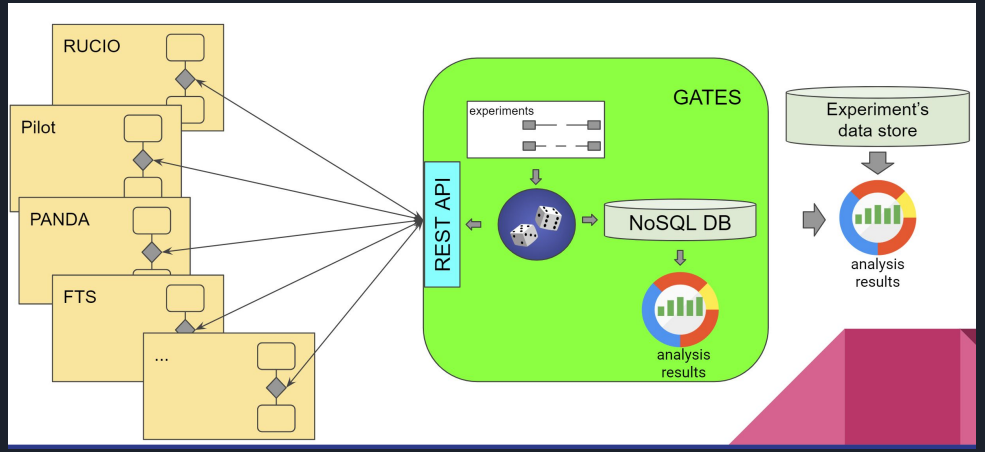
GATES

Experiments have large data stores collecting data from different computing systems: job scheduling, data distribution, FTS, PerfSONAR, etc.

While that is great for monitoring, accounting, and finding issues, it is not sufficient for the system optimization.

One can try to guess what kind of effect a change will made, but without validation it does not mean much.

We need a way to quickly test different options and get actionable answers.



In development.
Completely experiment agnostic.
Contributions welcome.

Tools - Anomaly detection in timeseries

Probably the most important tool:

- Data quality
- Network issues
- Site performance

Not having well annotated data is severely limiting us in tools we can use.

Not at all easy to find optimal sensitivity/specificity.

Despite importance, not many off-the-shelf tools. We use or tried:

- Simple hwm/lwm limits
- ARIMA
- plato detection
- Bayesian simultaneous change point detection
- ANN/BDT in time bins

Tools - classification

Needed in at least several different systems:

- Data popularity prediction (based on dataset names, thanks to our encoding metadata in dataset names).
- Error classification:
  - Jobs - almost free style text - NLP?
  - FTS
  - Rucio
  - Frontier - parsing sql queries
  - HC
- Significance of issue - can be defined in all kind of different ways. Not at all trivial.
- Attribution - we very often don’t have a way to find out who is responsible to fix the problem.

Time-To-Complete Predictions

Several attempts:

- FTS - predicting file transfer times (src, dest, ..)
- Panda - task TTC

Nice to have but of dubious value:

- If predictions used by humans not sure humans will know how to optimize based on predictions.
- Probably too slow feedback for usage in automatic feedback connected systems.

Needs - An annotation tool

Currently we only have tickets as a history of things that happened.

Not classified in any way that can be used to train any model.

Going through historical data and doing it manually is not at all easy - nobody wants to do it. Postfact automatic annotation does not help.

We need a simple, quick, experiment agnostic way of annotating any kind of event in any kind of system, collecting all the info needed to correlate events with other data in the system.

Needs - MODELS

Models for our biggest, most important systems

e.g. Panda, Rucio, FTS

Once we have reliable models:

- it becomes easy to find if they are running nominally or not.
- it gets easy to test in advance what kind of effect a change would have

Needs - specialization

We have a lot of different services.

Distributed computing experts and specially site admins don’t have time/knowledge required to:

- Run all the services
- Understand how performant are they
- Tune them
- Understand when there is a problem

Optimally we would want one/two people - service experts - that run/manage them for all sites or maybe even experiments.