

Survey of Workload Management and Resource Provisioning Systems

Eric Vaandering — Fermilab

Executive Summary

An overview of workflow and resource provisioning observations from CHEP 2016¹ in San Francisco, October 2016.

Each of the LHC experiments has been extending their workflow and resource provisioning systems to be able to access clouds, super computers, and other opportunistic resources. None of the experiments seem to feel their workflow systems are clearly inadequate for the Run 3 or HL-LHC challenges. ATLAS, in particular, feels they can scale up by 5–10x. Both CMS and ATLAS are currently operating at the 200-250K core level routinely. Workflow systems and experiment frameworks are being pushed in new directions to operate on preemptable resources.

While all the workflow systems rely on pilots for resource provisioning, both PanDA and DIRAC use pilots tightly coupled to the workflow system and are in the process of reworking their pilots. Both PanDA and DIRAC are multi-VO: PanDA supports ATLAS, AMS, and COMPASS. DIRAC seems to have momentum and supports LHCb, Belle II, BES III, CTA, ILC, and a number of small VOs through the UK's GridPP project. DIRAC has recently took the step of having community governance.

Both DIRAC and PanDA have been extended by building additional layers around them for workflow chaining, workflow request management systems, etc. All experiments are breaking the old MONARC “Tier”ed model which assumed resources at sites were specialized and networks were unreliable and expensive. Instead, experiments are relying on remote reading (xrootd) and/or data transfers as part of the workflow system to make workflows more agnostic about where they run.

Most experiments have had good success with analysis trains or producing analysis-group specific datasets with the production system.

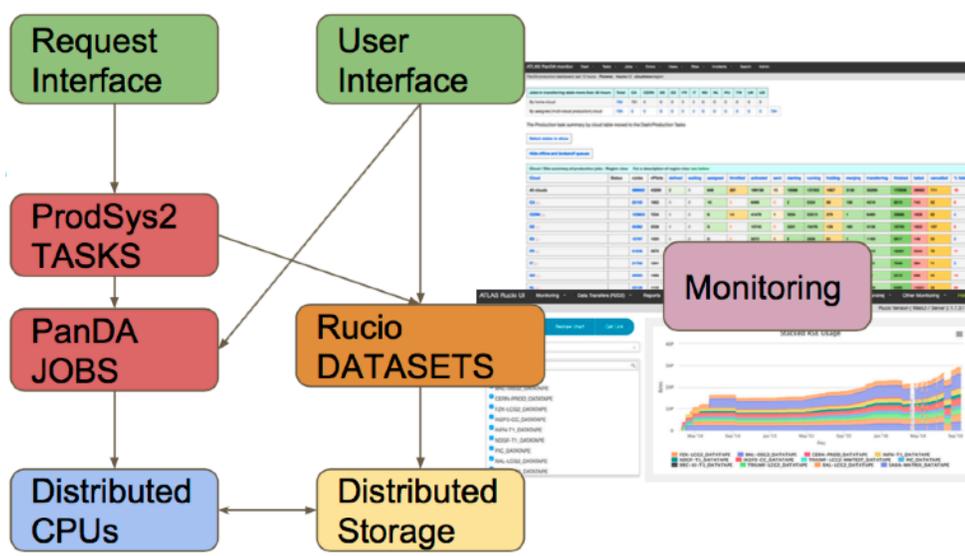
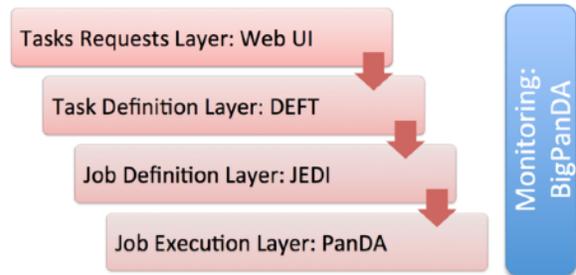
Personally I was disappointed as I was looking for ways in which the LHC experiments could collaborate on the flexible definition of workloads (such as the chained workloads mentioned above) and on resource provisioning. With each experiment firmly on their own path, the opportunities for such commonality seem limited, especially with such tightly coupled systems.

Contents

- ATLAS & PanDA/BigPanDA2
 - Optimizations.....3
 - ATLAS Event service.....3
- DIRAC (LHCb and others)4
 - DIRAC Pilots5
- CMS & GlideinWMS.....5
- Opportunistic and HPC strategies.....6
- Other technologies.....6
- References.....6

ATLAS & PanDA/BigPanDA

PanDA is part of the larger ATLAS system ProdSys2² which they revamped for Run2. PanDA provides the job execution and resource provisioning layers. They also use JEDI for job definition, DEFT for bookkeeping and workflow definition, and a Web UI, which all sit on top of PanDA analogous to how CMS uses McM, ReqMgr, and parts of WMAgent on top of GlideinWMS. They mention BigPanDA as the monitoring layer, which is interesting since my understanding was that BigPanDA was supposed to be the multi-VO version of PanDA. (See diagrams to the right and below.) PanDA is also used by AMS and COMPASS, but all the presentations I saw seemed to be from ATLAS people.



Each of these layers communicates with each other through databases. I presume this is the PanDA database that they also use for all their job matching. They have been able to scale the system up to managing 250k cores of computing.

Their largest change to the computing model for Run2 regards their “cloud” model³ (not to be confused with cloud computing). This is how they determine where workflows run. They have network quality measurements between each of their sites. For a given workflow, one site is picked as the nucleus of the workflow and the destination site for the data. Well connected sites are also given jobs from this workflow, read data remotely, and write data back to the host site. This allows them to use reliable Tier2s to help complete the work assigned to Tier1s. This is similar in concept to what CMS accomplishes with “overflow” in GlideinWMS and allowing remote reads with AAA/xrootd.

ATLAS is in the process of rewriting their pilot and pilot factories.⁴ The Pilot 2.0 is a nearly complete rewrite since the older pilot was getting difficult to maintain. They need the ability to submit just one or two pilots to a supercomputer site and be able to claim large amounts of resources with that one pilot. They are also replacing the APF pilot factory with a new component now called Harvester. Harvester no longer uses HTCondor for submission to the sites. This also helps them with getting the burst capability in HPC resources, because marrying HTCondor with MPI is a difficult problem for them.

There was an illuminating talk about how ATLAS deals with memory issues.⁵ Unlike GlideinWMS, they don't schedule memory within the pilot and apparently job memory requirements are not passed into the underlying batch systems on their grid nodes. So they have a number of different queues with different memory requirements going up to 64 GB/core at some of their large Tier1 and Tier2 sites. Memory requirements for a particular workflow are determined by a few scouting jobs and the workflow is submitted to the right queues. The few jobs that are killed for exceeding these limits are resubmitted to higher memory queues. Additionally some sites don't kill on memory, so the pilot has to kill the job at 2x the requested memory.

Optimizations

ATLAS described a number of optimizations to push their work through faster,⁶ sometimes at the expense of the amount of resources used. They give a priority boost to almost done tasks. They also have the ability to submit multiple identical jobs (to different sites) to guarantee completion of the highest priority workflows and discard output from the duplicates which may finish. They are introducing global fair share between analysis and production which can then use the same batch account. Finally, they say that they have cut analysis usage (as a fraction of the total) about in half by moving the first step of analysis workflows into the production system. I don't think this is strictly an analysis train model, just a definition of analysis workflows in the chain of workflows.

ATLAS Event service

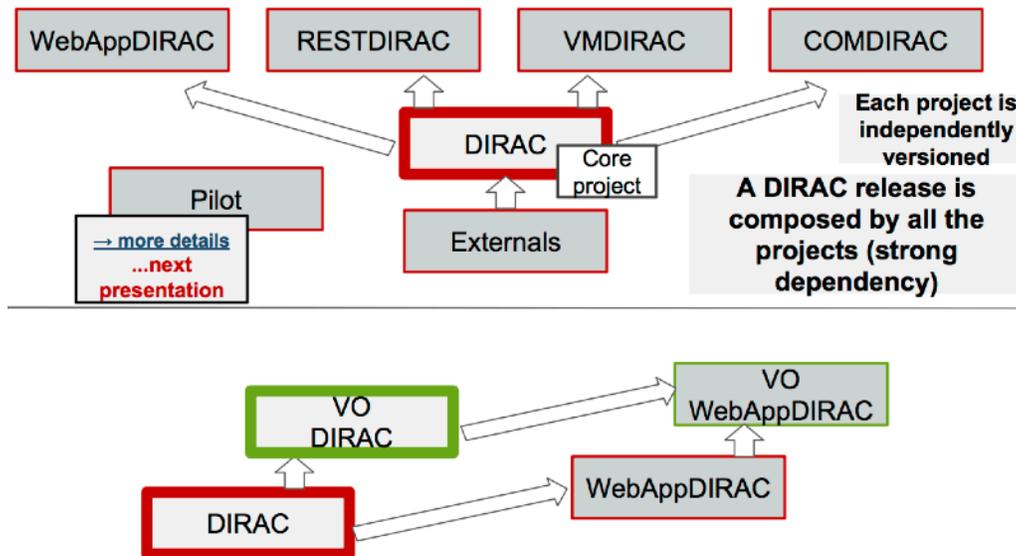
ATLAS has found that strict job-based workloads are not good for non-traditional resources (HPC, clouds, Boinc). The amount of time to run something doesn't match well to a predictable job, so they've been working on an event service of sorts.⁷ They have two flavors of this depending on if they have outbound connectivity. In both cases they assign some amount of work

to a job which runs as the service. The first version saves events in an object store every 10 minutes as its working. The second version (for HPCs with no outbound access) writes the events and how far it got to the disk of the WN; the ARC CE they use on the HPC does the data transfer and fakes a finished job. In both cases, if the worker is preempted, the PanDA system (which this is tightly integrated with) reassigns the undone work to another event service. By adopting this approach they recover about 10% of the data processing (about 20% of the jobs are preempted). They appear to have plans to extend this into what I would consider a real event service where specialized processes are responsible for streaming input and output from a number of worker nodes.⁸

DIRAC (LHCb and others)

DIRAC⁹ appears to be gaining mindshare and users. It's an open source effort with about 5 core developers and since 2014 there has been a consortium with governance agreement, etc. (It's been multi-VO since 2009.) Various experiments have signed on to use it: LHCb (the originators), Belle, CTA (Cherenkov Telescope), ILC, and BES III. There is also an effort in the UK to be able to use a single instance for a number of smaller VOs.¹⁰

The software consists of a lot of interrelated products which are each versioned and released separately, but you need all of them for a distribution. Each component can also be extended by each VO for their own needs while maintaining the core.¹¹ (See below.) DIRAC includes its own data catalog and data movement service. However, it can also be used with external data catalogs as some of the stakeholders do this.



CTA is building in the possibility to pre-define file locations to avoid DB queries for chained workflows.¹²

DIRAC Pilots

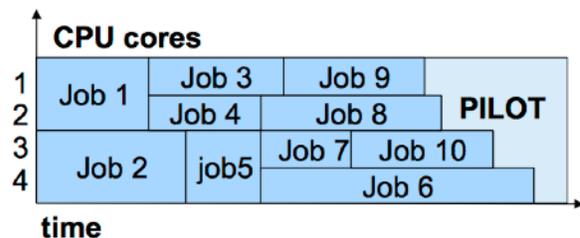
LHCb is very interested in benchmarking and understanding which CPUs are faster than others to determine how much work to give them (CMS just targets some average), so they do benchmarking as part of the pilot. They have been doing other work on their pilot, including recently adding multicore support. However, they do not support mixed multi/single core jobs in a pilot and don't have plans to add this without a strong use case. (I assume this means they run one job/pilot.) They've been hard at work with their pilot code making HPC and other resources look the same as Grid resources.

One surprise is that only recently, with work from the ILC group, has DIRAC been able to run on OSG resources using the Globus toolkit and the ability to connect to HTCondor-CEs. It's unclear if this code is now part of the common DIRAC distribution.

CMS & GlideinWMS

There were no talks or presentations on WMAgent, the workflow system for CMS. There was a talk on Unified,¹³ which is a set of scripts that sits outside of WMAgent and ReqMgr assigning workflows, moving data, and double checking results before datasets are announced. This has reduced operator effort as the number of workflows has grown and also increased efficiencies by allowing more sites to contribute to individual workflows.

The CMS global GlideinWMS pool was discussed in a pair of talks. The first¹⁴ concentrated on CMS efforts to move to multicore partitionable pilots scheduling a mix of multi and single core work. Both types of work will co-exist during Run2 since the multi-threaded CMSSW framework is not used universally for all types and steps of workflows. During 2015 and 2016, all Tier1 and nearly all Tier2 sites began accepting multicore pilots for CMS. Since the pilots run HTCondor internally, multi and single core work can be scheduled inside the pilot, although with some loss in efficiency near the end of the pilot lifetime (shown to the right). However, multicore pilots reduce the total number of pilots (and jobs) that must be tracked in CMS and give them a flexible system where the work being scheduled need not match the size of the pilots.



The CMS Global Pool is now scaling to 160k cores and beyond.¹⁵ This has been a constant effort working with both GlideinWMS and HTCondor developers to identify and fix the bottlenecks. The current scale is limited by I/O, combinatorics of job matching, and the speed of components. Multicore pilots and workflows have reduced some scaling problems and exacerbated others. Reaching higher scales will require identifying issues with GlideinWMS and HTCondor and fixing them one by one; there is no dominant issue. A new round of scale tests, with the goal of testing the provisioning system to 500k cores before it's needed, is contemplated using 32 startds/glidein pilot.

Opportunistic and HPC strategies

All experiments and facilities that are exploring clouds are working, as best as possible, to integrate these resources in a seamless way, hiding the complexities and uniqueness, especially of HPC systems, from the underlying parts of the workflow system and the applications. The pilot systems of the experiments are the key to this; starting the pilots may vary greatly, but once started they often integrate seamlessly.

However workflows must be chosen carefully to run on sites with limited data access or where removing data from the provider (e.g. commercial clouds) incurs real expense.

Other technologies

There was an increasing amount of discussion around lightweight containers such as Docker, Shifter, etc. as being very useful tools to solve the environment issues we have and for being an easy way to bring your environment along with you, especially combined with CVMFS. We in HEP were admonished in being behind in this, especially in the container orchestration game with things like Mesos, Docker Swarm, and Kubernetes.

Networks are the last technology we have that is still doubling every 18–24 months, but we were warned this may not continue as the data able to be carried by a single fiber appears to be reaching a plateau. Should we find ourselves in a position where we have storage geographically separated from our CPU, LHC experiments will require Tb/s networks (and peering with commercial providers). As usual there were discussions about needing to schedule network bandwidth as a resource and claims that SDN was the way to do this. This could be useful for both scheduled data transfers or on-demand reading (xrootd).

One brief mention that interested me was of the EOS multi-site cluster.¹⁶ There is now a single instance of EOS that spans three sites across Australia. This is an extension of the Meyrin & Wigner work CERN has put in place already. Could this be the start of a global file system with appropriate replication and caching that could possibly remove the need to actively manage data locations?

References

¹ CHEP 2016 homepage: <https://indico.cern.ch/event/505613/timetable/>

² Borodin, M. <https://indico.cern.ch/event/505613/contributions/2230440/>

³ Megino, F. <https://indico.cern.ch/event/505613/contributions/2230706/>

⁴ Maeno, T. <https://indico.cern.ch/event/505613/contributions/2230704/>

⁵ Forti, A. <https://indico.cern.ch/event/505613/contributions/2230705/>

⁶ Pagés, A. <https://indico.cern.ch/event/505613/contributions/2230712/>

- 7 Cameron, D. <https://indico.cern.ch/event/505613/contributions/2230710/>
- 8 Tsulaia, V. <https://indico.cern.ch/event/505613/contributions/2230947/>
- 9 DIRAC homepage: <http://diracgrid.org>
- 10 Bauer, D. <http://indico.cern.ch/event/505613/contributions/2230725/>
- 11 Stagni, F. <http://indico.cern.ch/event/505613/contributions/2227928/>
- 12 Arrabito, L. <https://indico.cern.ch/event/505613/contributions/2230708/>
- 13 Vlimant, J.R. <https://indico.cern.ch/event/505613/contributions/2230726/>
- 14 Yzquierdo, A.P. <https://indico.cern.ch/event/505613/contributions/2230723/>
- 15 Yzquierdo, A.P. <https://indico.cern.ch/event/505613/contributions/2230730/>
- 16 Curull, X. <https://indico.cern.ch/event/505613/contributions/2230703/>