

Middleware Trends in offline computing summary (with a bit of monitoring too)

Several themes were evident at CHEP 2016 in middleware developments. Generally speaking, trends are moving towards simplifying things for users, and trying to make intelligent decisions regarding job submission options and job site steering.

We can roughly divide the presentations into user-facing tools and non-user-facing tools. The overall theme of the user-facing tools was to continue shielding the user from the complexities of grid environments and creating job requirements. There were some interesting ideas from CMS about taking a job submission and running a single test job with some small piece of the dataset, and then using the result of the test job to try to automatically submit a proper number of jobs, and to automatically re-split the dataset and add additional jobs to process the smaller remaining piece of data dataset if the system detected that it was taking too long to complete the processing of a dataset. Lobster (built on CCTools) also had some interesting ideas in that it contains the idea of “categories” of jobs, each with its own limits on concurrency, and runtime, which can be set by admins appropriately for each category. That eliminates the need for each individual user to profile his/her jobs, at least if running one of the predefined categories. It also has the ability to feed back information from jobs in the task that have already finished, allowing later jobs to have more appropriate resource limits tailored to the task (as opposed to the user guessing at the beginning.) The drawback there is that the “early” jobs in the task may end up requesting a much larger resource allocation than they need (then it gets “tuned” later), so some kind of happy medium between the two approaches (the other being 100% user-specified) might be an interesting way to go in the future.

Non-user facing middleware saw concerted efforts towards interoperability between different types of resources, and improving experiments’ ability to more rapidly provision opportunistic resources. That includes things like a BOSCO-style interface, where remote resources can be reached via a single ssh connection, reducing the burden on administrators who might otherwise be reluctant to embrace a complex system such as HTCondor. There are also some interesting ideas in the DIRAC universal pilot, which is simply a Python script, making it very adaptable to a wide variety of remote resources.

Monitoring talks were legion at CHEP, and while everyone recognizes the need for robust monitoring that satisfies both admins and end users, to date no single software package has emerged as standard. There is heavy use of off the shelf tools such as the ELK stack, Grafana, etc., but FIFEMON has been very well-received by the larger community, and the tools behind it are slowly being adopted by other parts of the field. It has as good a chance as anything to emerge as a standard toolset in the field. Nonetheless we should study other setups within the field, as there is always something new that we could possibly incorporate into our own systems.

Selected User-facing middleware talks:

DAGMan in CRAB3 (automatic task splitting, etc.):

<https://indico.cern.ch/event/505613/contributions/2230736/attachments/1347290/2041501/Oral-494.pdf>

Lessons learned with Lobster:

<https://indico.cern.ch/event/505613/contributions/2230715/attachments/1347614/2041496/Oral-224.pdf>

ATLAS resource monitoring in software releases and CI (similar to what we are now doing wrt the ART memory tracker, new CI tools w/physics validation, etc.):

<https://indico.cern.ch/event/505613/contributions/2227923/attachments/1350855/2043253/Oral-127.pdf>

Derek Weitzel et al., HTCondorCE+ BOSCO:

<https://indico.cern.ch/event/505613/contributions/2227941/attachments/1347462/2044936/Oral-503.pdf>

Experience with DIRAC in large VOs:

<https://indico.cern.ch/event/505613/contributions/2227928/attachments/1345987/2047255/Oral-v2-217.pdf>

DIRAC Universal pilots:

<https://indico.cern.ch/event/505613/contributions/2227927/attachments/1345988/2047261/Oral-v2-211.pdf>

Selected Non-user-facing middleware talks

Qiulan Huang, Elastic computing resource management based on HTCondor:

<https://indico.cern.ch/event/505613/contributions/2227931/attachments/1346850/2043531/Oral-v3-288.pdf>

Dirk Hufnagel, CMS use of HPC resources:

<https://indico.cern.ch/event/505613/contributions/2227945/attachments/1346692/2043224/Oral-563.pdf>

HTCondorCE and OpenStack:

<https://indico.cern.ch/event/505613/contributions/2227921/attachments/1340266/2043473/Oral-98.pdf>

Selected monitoring talks:

Edward Karavakis, Unified monitoring for IT and grid services (uses some FIFEMON-like tools):
<https://indico.cern.ch/event/505613/contributions/2227920/attachments/1347026/2041426/Oral-v3-83.pdf>

PhEdEx monitoring in CMS:
<https://indico.cern.ch/event/505613/contributions/2227935/attachments/1347639/2041231/Oral-369.pdf>

Distributed system monitoring with MonaLisa:
https://indico.cern.ch/event/505613/contributions/2227938/attachments/1347516/2041640/Oral-463_v5.pdf

Software performance monitoring in ATLAS:
<https://indico.cern.ch/event/505613/contributions/2227923/attachments/1350855/2043253/Oral-127.pdf>

Monitoring at INFN (some pieces similar to FIFEMON):
<https://indico.cern.ch/event/505613/contributions/2227917/attachments/1346728/2041386/Oral-v1-018.pdf>