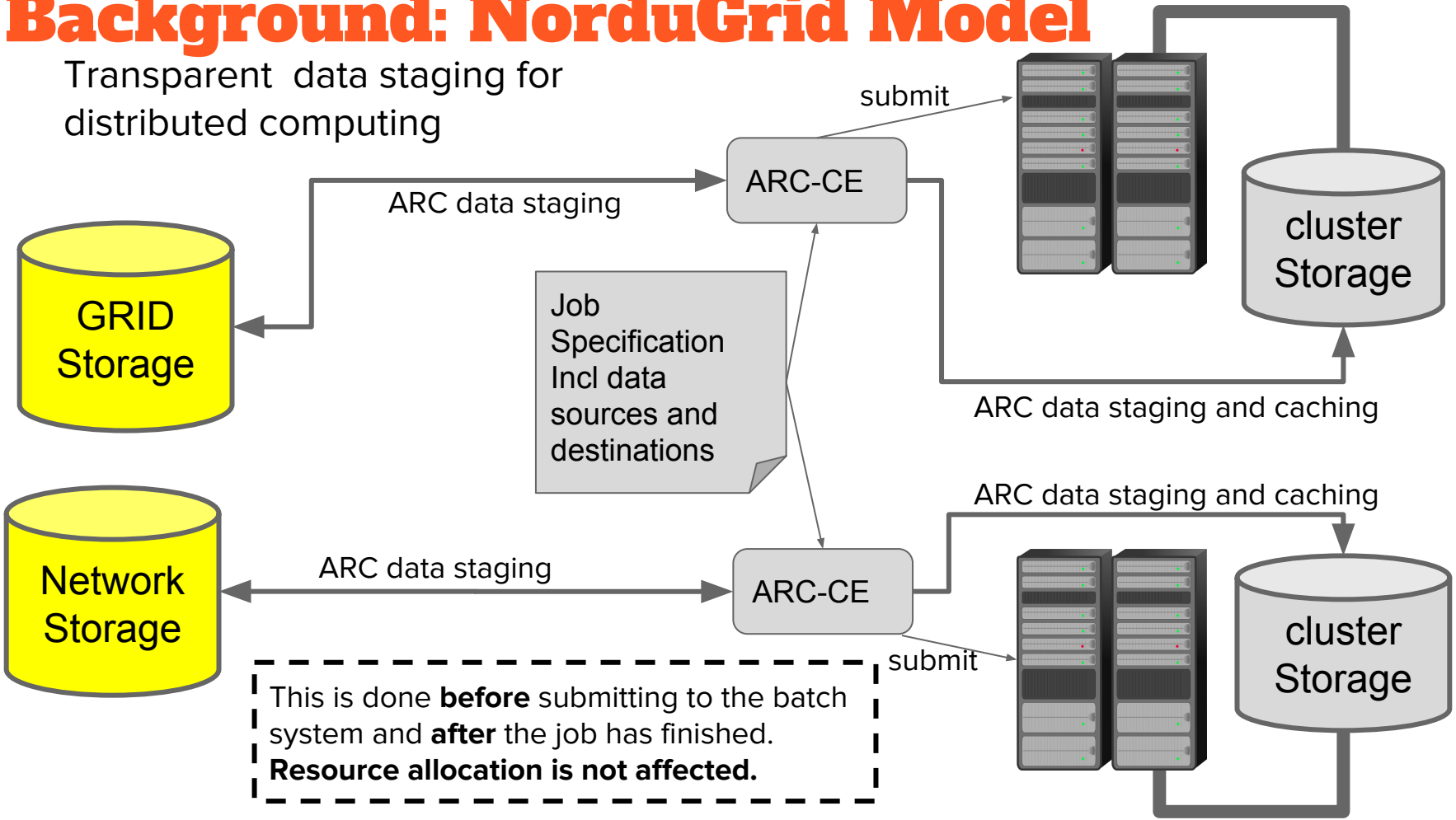
aCT: an introduction

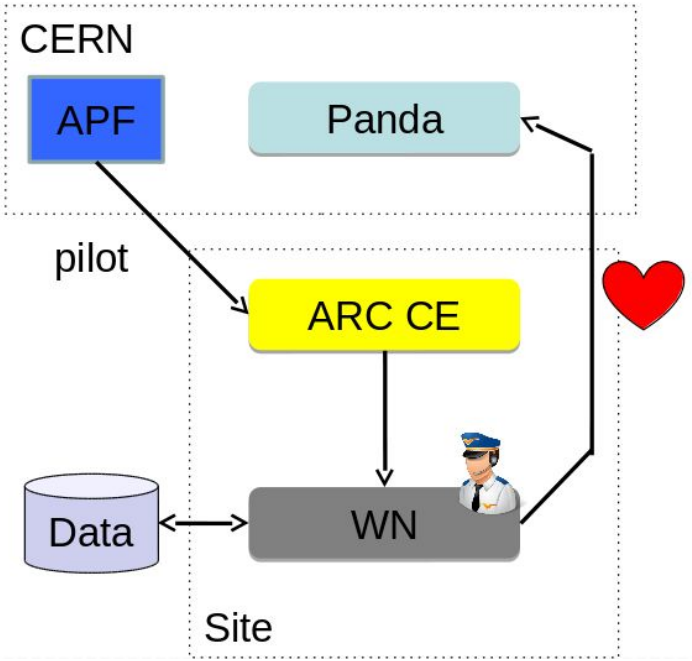
Slides courtesy of David Cameron, Oslo University
Addenda by Florido Paganelli, Lund University
ISGC 2019, Academia Sinica in Taipei,
Taiwan 31 March - 5 April 2019

Background: NorduGrid Model

Transparent data staging for distributed computing



Background:
ATLAS Panda "Pilot Factory" Model



- ATLAS Pilot Factory: A special Pilot job coordinates the work on a worker node, e.g. pulls subjobs from PANDA
- Each pulled job downloads data to work on, or works on data already staged.

Issues with NG model:

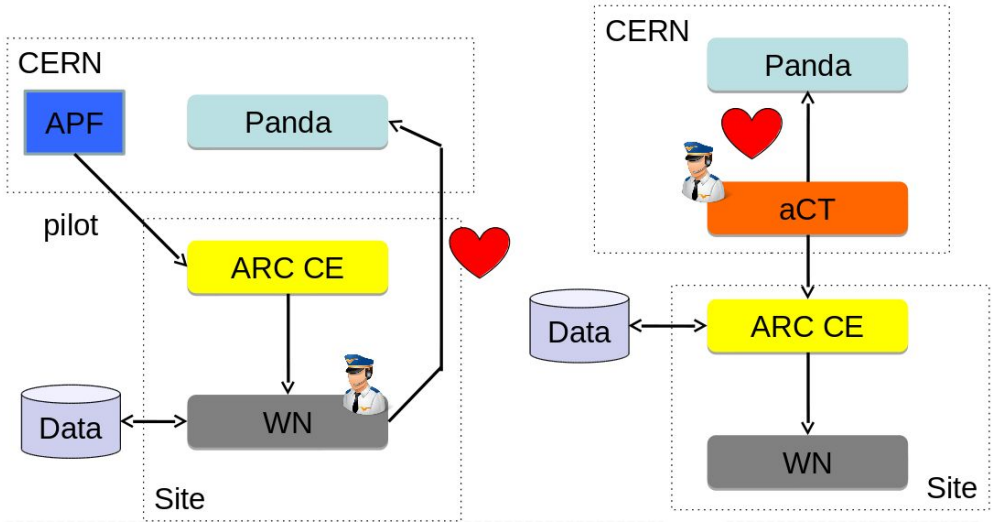
A. The resource requirements for each single job is NOT known at submission time
B. data requirements NOT known at submission time => load of huge datasets

History

- NorduGrid model was built on philosophy of ARC-CE and distributed storage
  - No local storage - data staging on WN is too inefficient
  - No middleware or network connectivity on WN
  - Everything grid-related was delegated to ARC-CE
- The pilot model implemented in Panda for **ATLAS** did not fit easily
  - An intermediate service was needed to fake the pilot and submit to ARC behind the scenes
  - ~2008 ARC Control Tower (aCT) was written (by Andrej Filipcic) and ran in Ljubljana
  - 2013-14 aCT2 was written and the service moved to CERN
    - Multi-process instead of multi-thread, MySQL instead of sqlite
  - 2015-17 Many sites moved from CREAM CE to ARC CE
    - Creation of truepilot mode
  - 2017: Condor support added for submission to HTCondor CE, CREAM, ...

Differences between aCT and pilot factory

- Pilot factories submit a generic wrapper with fixed batch system requirements, which then pulls a real job
- aCT pulls the job from Panda, then submits to the CE with the exact requirements of that job
  - Including no of cores, memory, walltime, priority
  - Useful if a wide range of workloads run on the same batch system
- What about late-binding?
  - If the CE and batch system properly propagate the priority set in the job description, the highest priority jobs will always run first

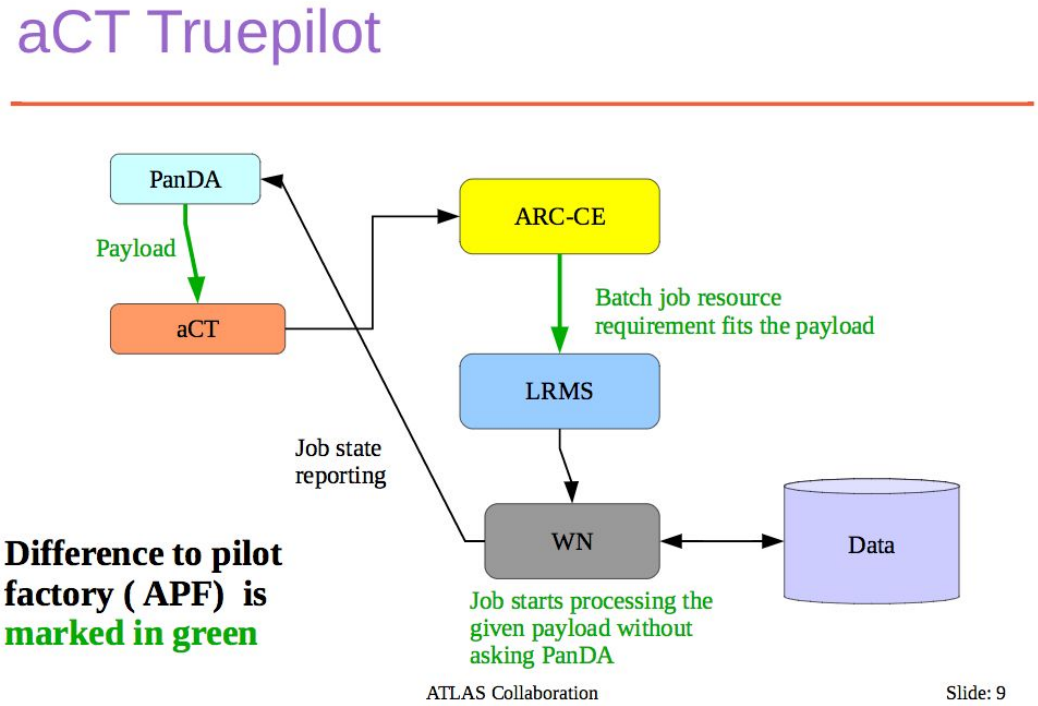


NorduGrid mode

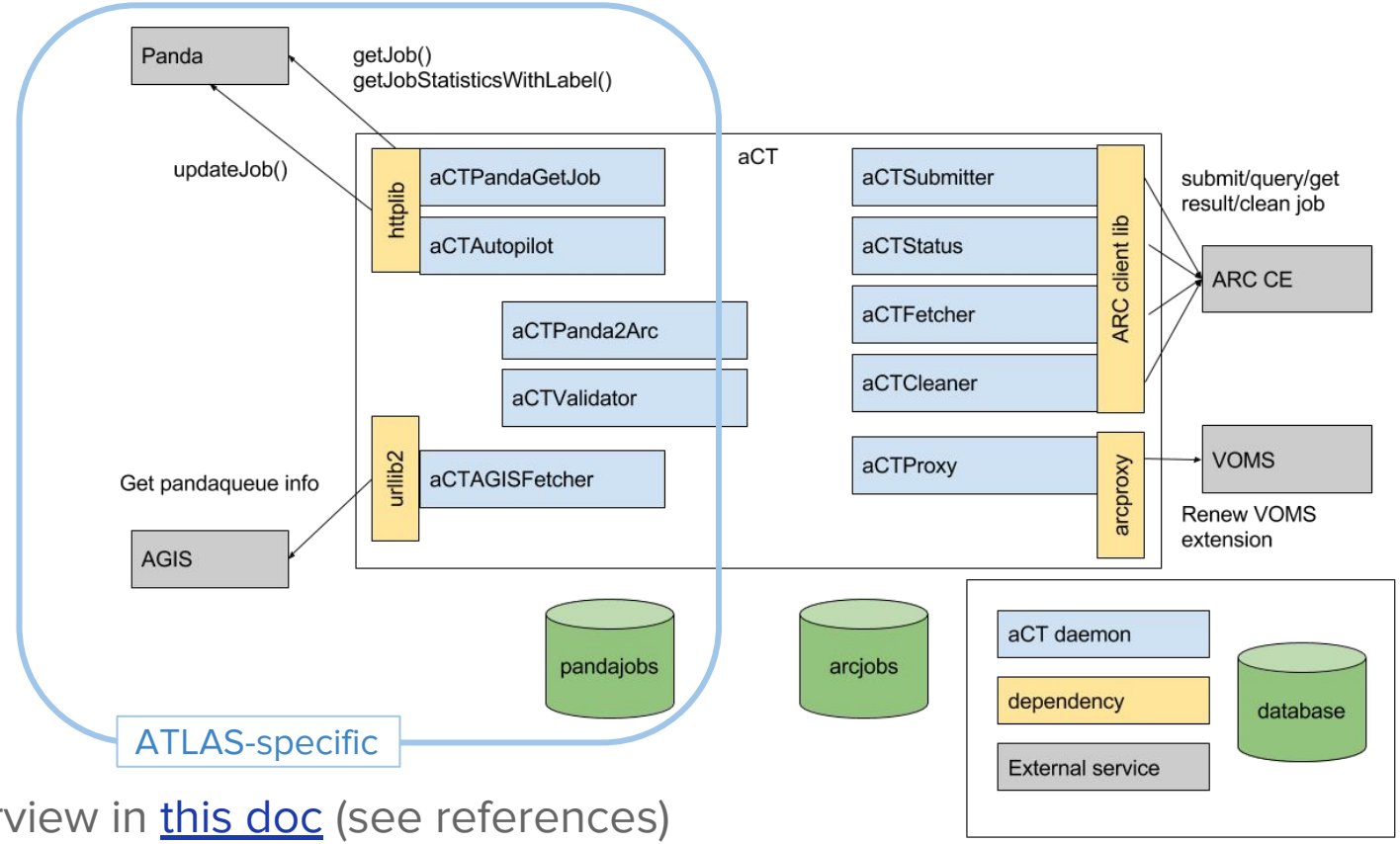
- aCT has to emulate certain parts of the pilot
  - Get new jobs, send heartbeats
- Post-processing
  - Once job finishes, it leaves its log and some meta-information for aCT to download through the CE
  - Log is copied to web area for access via monitoring pages
  - Output files are validated (check size and checksum on storage vs pilot meta-info)
  - Job meta-info is sent to panda along with final heartbeat
- If job fails badly (pilot crash or batch system kill) and no pilot info is available
  - aCT sends what it can to Panda
  - Error info and timings from the CE

True pilot mode

- For sites running ARC CE who do not need the full “native” mode with staging etc
- aCT fetches the payload and submits it to the ARC-CE
- ARC-CE submits the batch job with predefined payload and requirements
- Pilot on the worker node does the same as on the conventional pilot sites including downloading of data, but skips the fetching of payload from PanDA
- aCT sends heartbeats to Panda up until job is running, then leaves it to pilot



General Architecture



Overview in [this doc](#) (see references)

aCT Daemons

ATLAS Daemons:

- aCTPandaGetJob: Queries panda for activated jobs and if there are any, gets jobs
- aCTAutopilot: Sends heartbeats every 30 mins for each job and final heartbeats
- aCTAGISFetcher: Downloads panda queue info from AGIS every 10 minutes. This info is used to know which queues to serve and the CE endpoints.

ARC Daemons (use python ARC client library):

- aCTSubmitter: Submits jobs to ARC CE
- aCTStatus: Queries status of jobs on ARC CE
- aCTFetcher: Downloads output of finished jobs from ARC CE (pilot log file to put on web area, pickle/metadata files used in final heartbeat report to panda)
- aCTCleaner: Cleans finished jobs on ARC CE
- aCTProxy: Periodically generates a new VOMS proxy with 96h lifetime

Internal Daemons:

- aCTPanda2Arc: Converts panda job descriptions to ARC job descriptions and configures ARC job parameters from AGIS queue and panda job info
- aCTValidator: Validates finished jobs (checksum of output files on storage etc) and processes pilot info for final heartbeat report

Service setup and configuration

- 2 prod machines and 2 dev machines at CERN
- MySQL DBonDemand as DB
- Code and dependencies installed via pip and virtualenv
- Configuration is via 2 xml files, one for ARC and one for ATLAS
- Service is started by “actmain start”
- One prod machine runs almost all jobs
- One prod machine is warm spare running one job per queue
    - <maxjobs>1</maxjobs> can be changed if main machine goes down

Current status

- Average 200k jobs per day from one machine
- Peaks up to 300k/day

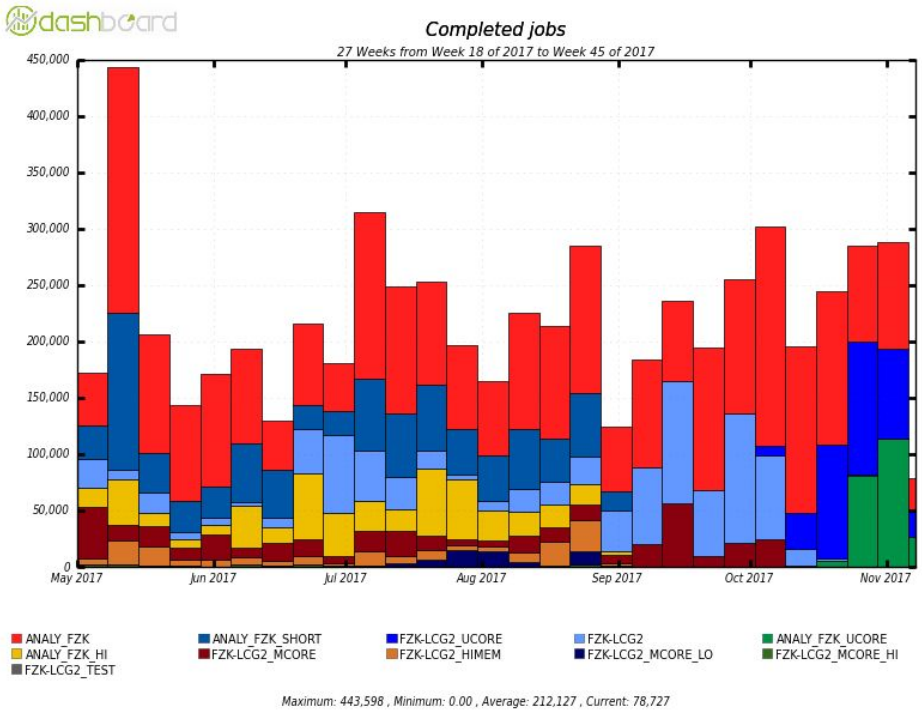
ATLAS sites served

NorduGrid
Truepilot

- T1: FZK, NDGF (4 sites), RAL, TAIWAN, TRIUMF
- T2: Australia-ATLAS, BERN, CA-SFU, CA-WATERLOO, CSCS, DESY-HH, LRZ, LUNARC, MPPMU, SiGNET, TOKYO, WUPPERTAL
- T3: ARNES, SiGNET-NSC
- HPC: CSCS (PizDaint), LRZ (SuperMUC), IN2P3-CC (IDRIS, in testing), MPPMU (Draco + Hydra), DESY-HH (Maxwell), Prague (IT4I)
- Others: BOINC
- Full list at https://aipanda404.cern.ch/data/aCTReport.html

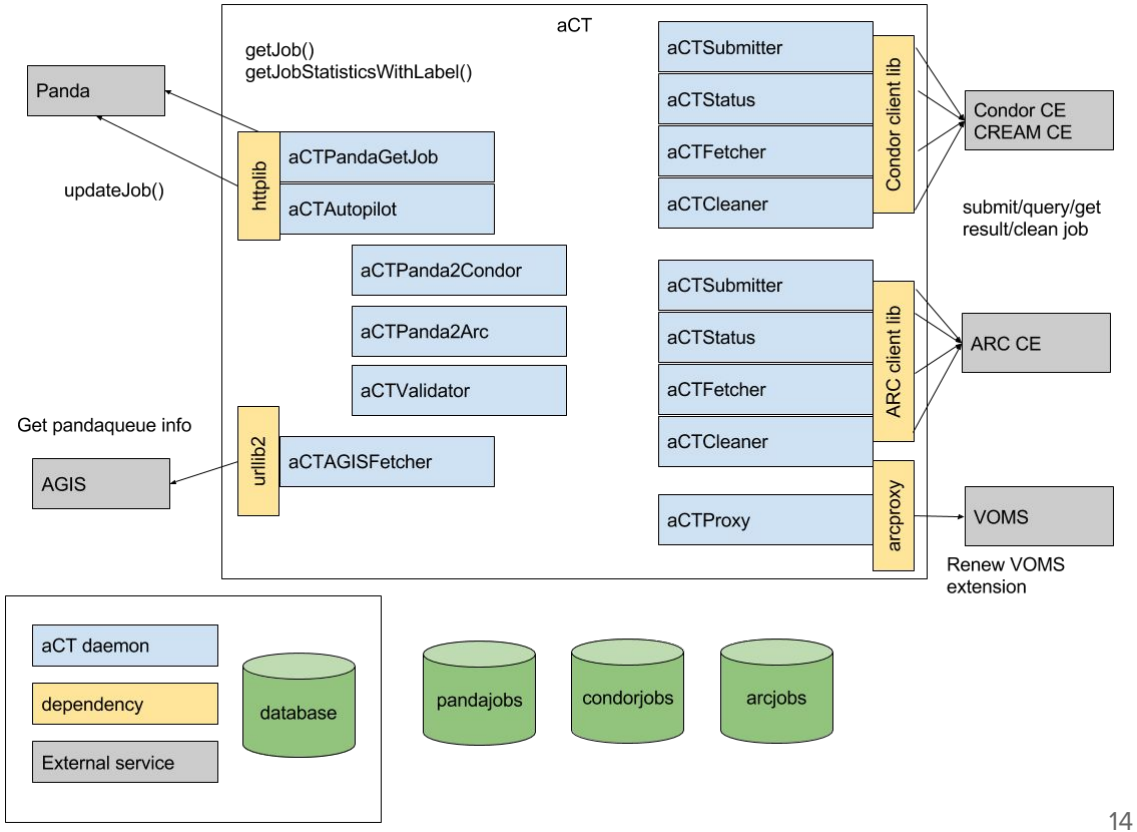
Unified queues

- Traditional pull-mode requires defining a “queue” for each type of pilot submitted, eg
  - Analysis single-core with low memory
  - Analysis single-core with high memory
  - Production multi-core with high memory and short walltime
  - etc.
- In push-mode only a single queue is required because the job requirements are dynamically set
  - However in ATLAS we still need separate queues for production and analysis because the credentials are different
- FZK went from 7 to 3 (now 2) queues



Condor submission

- Submission to Condor-G was implemented in 2017
- Separate DB table for condor jobs
- Submitter/Status/Fetcher/Cleaner for Condor using Condor python bindings
- Panda2Condor to convert pandajob to ClassAd
- Requires running local schedd with out of the box config
- Truepilot only



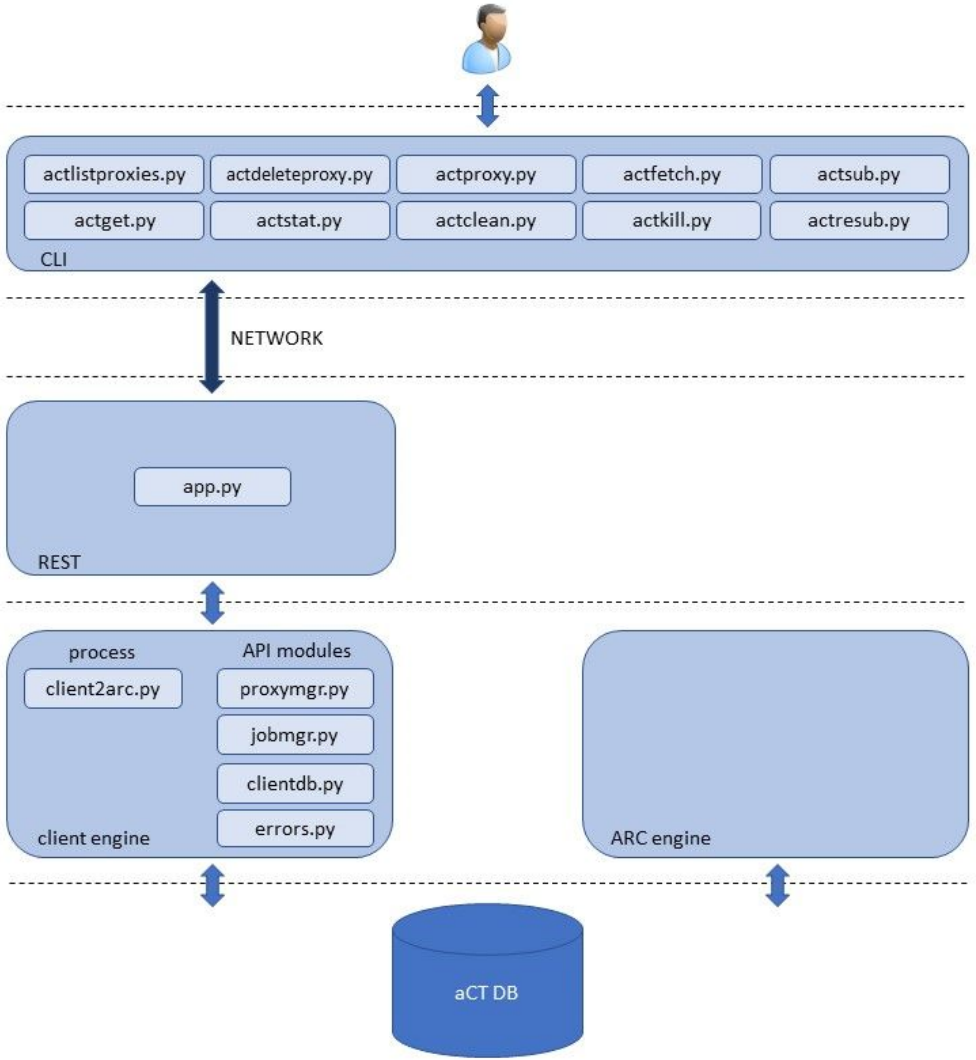
Condor submission

- Example classad
- “GridResource” is added at the time the submitter picks the job for a specific CE

```
{'Arguments': '-h IN2P3-LAPP-TEST -s IN2P3-LAPP-TEST -f false -p 25443 -w https://pandaserver.cern.ch',
 'Cmd': 'runpilot3-wrapper.sh',
 'Environment':
'PANDA_JSID=aCT-atlact1-2;GTAG=http://pcoslo5.cern.ch/jobs/IN2P3-LAPP-TEST/2017-11-07/$(Cluster).$(Process).out;APFCID=$(
Cluster).$(Process);APFFID=aCT-atlact1-2;APFMON=http://apfmon.lancs.ac.uk/api;FACTORYQUEUE=IN2P3-LAPP-TEST',
 'Error': '/var/www/html/jobs/IN2P3-LAPP-TEST/2017-11-07/$(Cluster).$(Process).err',
 'JobPrio': '100', <--- taken from job description
 'MaxRuntime': '172800', <--- taken from job description
 'Output': '/var/www/html/jobs/IN2P3-LAPP-TEST/2017-11-07/$(Cluster).$(Process).out',
 'RequestCpus': '1', <--- taken from job description
 'RequestMemory': '2000', <--- taken from job description
 'TransferInputFiles': '/home/dcameron/dev/aCT/tmp/inputfiles/3697087936/pandaJobData.out', <--- (real) job description
 'Universe': '9',
 'UserLog': '/var/www/html/jobs/IN2P3-LAPP-TEST/2017-11-07/$(Cluster).$(Process).log',
 'X509UserProxy': '/home/dcameron/dev/aCT/proxies/proxiesid5'}
```

REST Interface and user-friendly CLI

- Support for a REST interface and CLI tools to submit to aCT has been added
  - MSc. “User Interfaces for arcControlTower” by J. Merljak
- Allows aCT to be used as generic job management system without the application-specific part



Summary

- aCT is a stable mature product handling a large fraction of ATLAS workload
- The split between application and infrastructure allows easy extension to support new instances of either
- Code: https://github.com/ATLASControlTower/aCT
- Future plans:
  - Python 3 support
  - Re-design of database layer to support different database implementations
  - Move REST interface into production
  - Improve packaging and configuration to allow lightweight setups

References

- ARC Control Tower architecture: https://docs.google.com/document/d/15UvlpDTkLeYK38ornc27zGDI7Dg369hoYY1XIA2x0dk