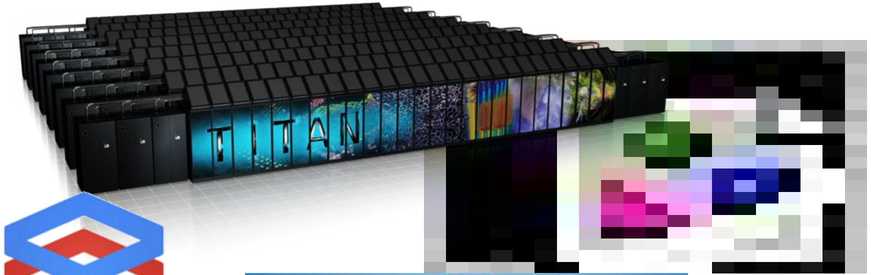
Benefits and performance of ATLAS approaches to utilizing opportunistic resources

Andrej Filipčič on behalf of the ATLAS Collaboration

Opportunistic Resources

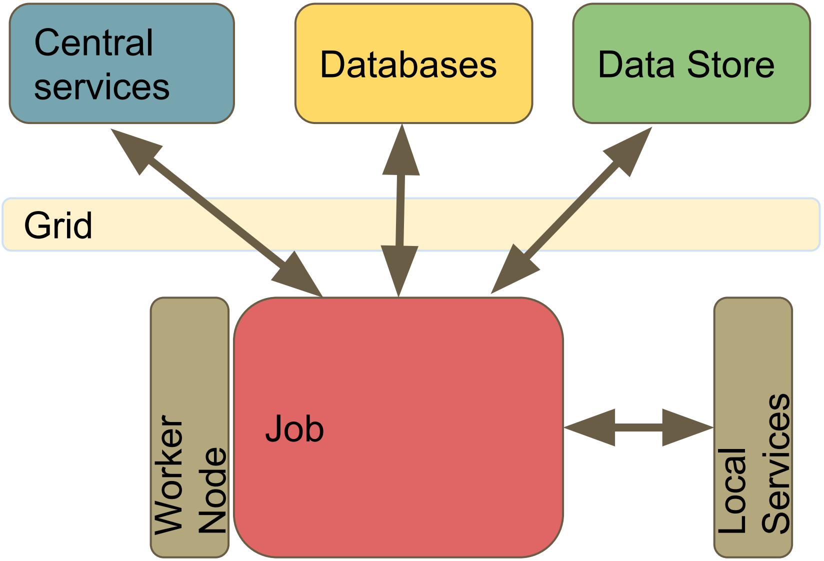
- Many centers have computing resources which are willing to contribute to ATLAS but they are not part of WLCG
- Some examples:
  - High-Performance Computing centers in EU, US and China (~10)
  - Shared academic clusters
  - Cloud resources, academic or commercial, private or public
  - Volunteers with home or office PCs
- ATLAS is exploring all those resources and including them in the production system



Job Execution in WLCG

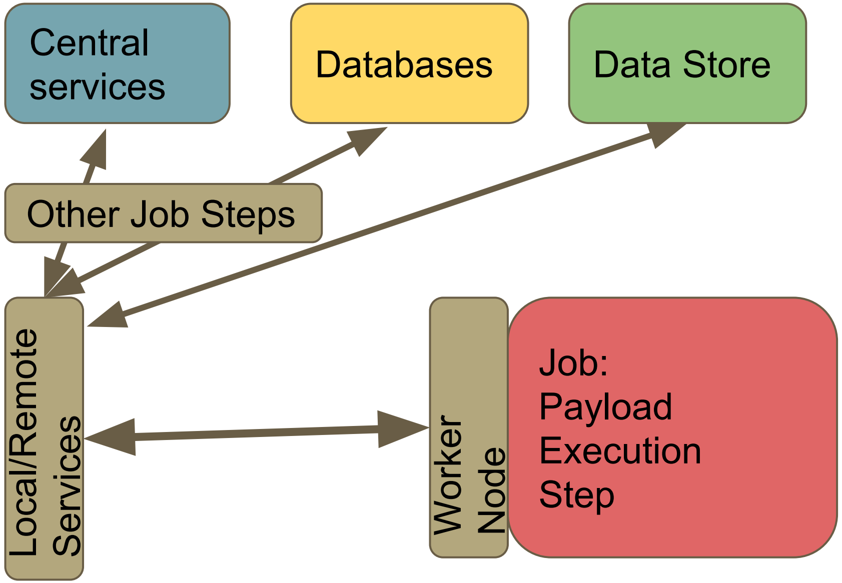
- Historically, grid infrastructure was not reliable - the jobs running on WNs took over all the workload
    - Communication with central services
    - Data staging, transfers
    - Access to databases and software repositories
- The WLCG cluster architecture differs significantly to a typical non-standard resource setup:
    - Local disk is typically shared
    - WNs do not always have outbound connectivity
    - Local storage is not useful as Storage Element
    - WAN traffic is limited

WLCG Job Execution Architecture



Job execution on opportunistic resources

- WLCG jobflow not applicable
- Separation of jobs steps
  - Payload distribution
  - Input data staging
  - Software distribution
  - Database prefetching
  - Payload execution
  - Output data transfers
  - Communication with central services
- All the steps but the payload execution can be offloaded to an external service, either local or remote

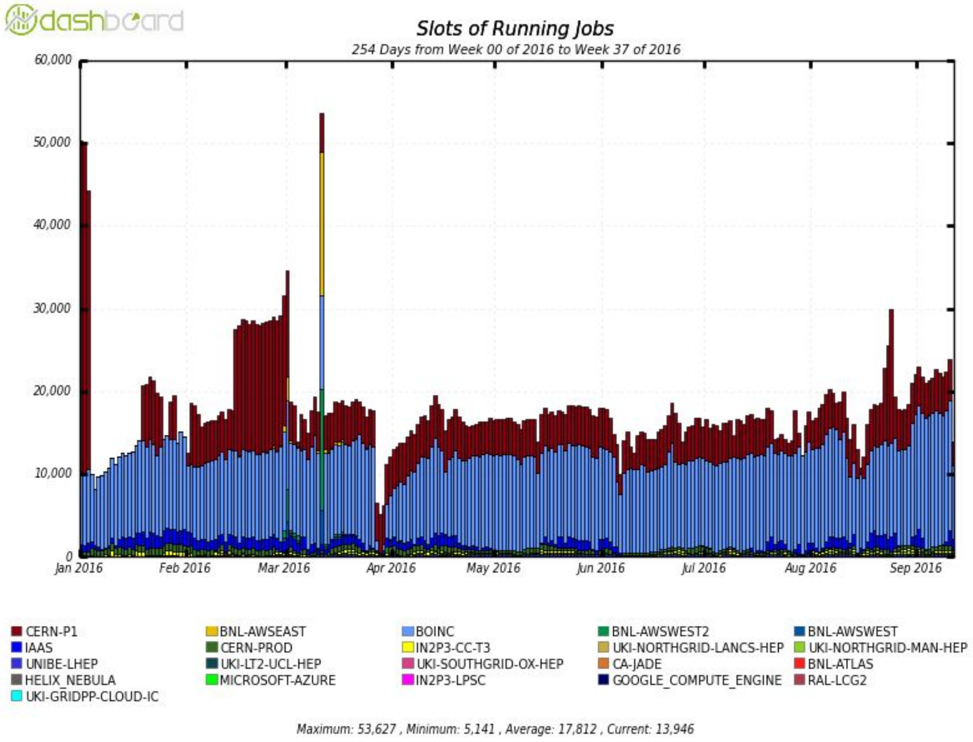


Offloaded job step execution in ATLAS

- Software Distribution:
  - Partial cvmfs copy and relocation to local shared filesystem
  - Parrot for sites with outbound connectivity but without fuse
  - Docker images in evaluation
  - Prefilling cvmfs cache within CERN VM with selected releases to minimize network traffic (ATLAS@Home)
- Database prefetching:
  - Using DBRelease sqlite local files with limited content, mostly for MC Simulation
- Clouds are flexible resources
  - ATLAS typically installs grid middleware on the infrastructure
  - Jobs execute in full WLCG pilot mode
  - Submission either through ATLAS Pilot Factory or embedded pilot in VM instance

Cloud Integration in production system

- CERN-P1
  - With cream-CE + HTCondor, on spare HLT slots, and full farm (e.g. Jan 2016) when HLT idle
- Academic clouds:
  - VAC, CloudScheduler, HTCondor, Federated cloud, ...
- Commercial clouds
  - Amazon EC2 test
  - Evaluating Google Compute Engine
- CERN
  - Evaluating commercial providers to possibly outsource a fraction of pledges



Clouds are easy to integrate and many techniques exist.
The best approaches will be consolidated in the future

Implementations of offloaded job step execution

arcControlTower + ARC-CE

- aCT
  - Payload distribution
  - Communication
- ARC-CE
  - Data staging
- WN job
  - Payload execution only

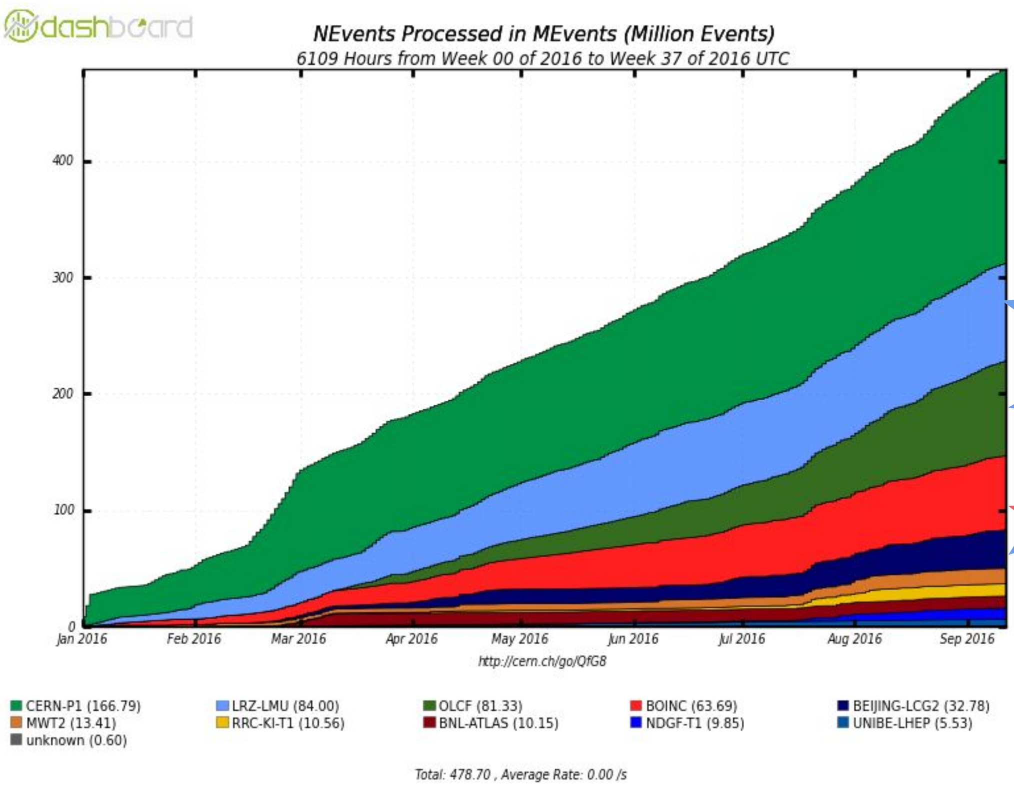
pilot for HPCs

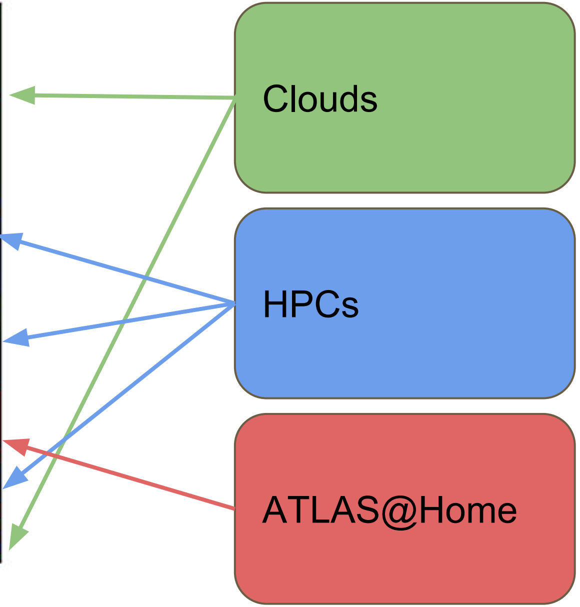
- Edge service part
  - Payload distribution
  - Communication
  - Data staging
- WN job
  - Payload execution only

ATLAS@Home

- aCT+ARC-CE
  - Payload distribution and data staging
  - Communication
  - Submission to Boinc
- Boinc
  - Payload and data distribution to PCs
- PC job
  - Payload execution
  - Communication to Boinc

ATLAS Monte-Carlo Simulation Contribution

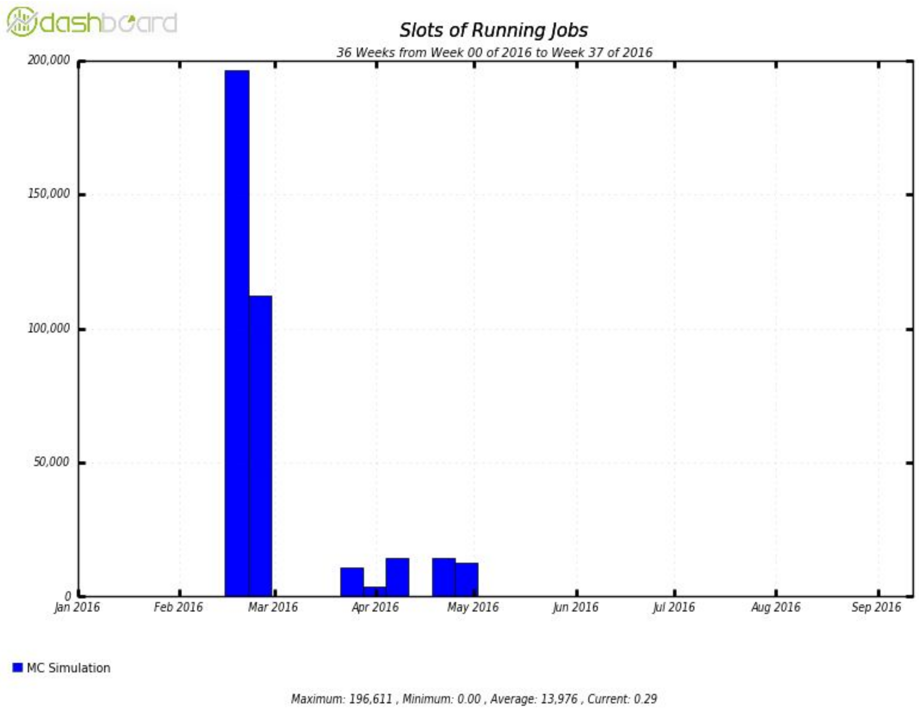




10% of MC events were simulated on opportunistic resources in 2016

MC Event generation on HPCs

- MC event generation is the only application running on non-x86 architecture for now
- Mira PowerPC HPC produced 25B events
  - Not yet automated through the production system
- Future ATLAS software aims to execute as well on
  - PowerPC
  - arm64



Evolution of ATLAS production system

- Already during LHC Run-1, ATLAS was able to exploit opportunistic resources and included them partially in the production system
- For Run-2, PanDA, JEDI, ProdSys2 and Rucio enabled even tighter integration:
  - Job execution steps present in PanDA since the initial design - different clients can update the status of the same payload.
  - Opportunistic resource descriptions (walltime, memory, installed software) match automatically the payload requirements
  - Dynamic job sizing and ATLAS Event Service are in development to even better explore opportunistic resources
- Further functionality (Harvester) is being developed to extend the production system to cover all possible mainstream site architectures

Requirements for the future distributed computing

- Transparent distributed job steps and control
  - Local and remote job services
- Common middleware platform
  - Reusing components (libraries) in job services and payload execution
- Efficient data caching
  - ARC-CE cache
  - XrootD cache
- Transparent usage of site services
  - Offload the data staging, transfers to eg. site gridftp doors or ARC-CE caching service
  - Extend the Compute Element to other systems (cloud schedulers, web-service submission services, SCEAPI ...)
- Optimization of the job workflows to
  - Maximize cpu efficiency and minimize the memory usage
  - Minimize the I/O and the WAN transfers
- Port the ATLAS software and middleware stack to non-intel platforms

Conclusions

- ATLAS Distributed Computing has demonstrated to be able to transparently integrate any kind of opportunistic resources into its production system to participate in the production in automated way due to:
    - Well designed production and data management system
    - Flexible separation and distribution of job execution steps
    - Extending the middleware backends and job execution wrappers to various computing site architectures
- In 2016, the opportunistic resources contributed modestly (~10%) to ATLAS production, but the future is bright:
    - Many of the HPCs and Clouds are still in early development stages, ATLAS did not request yet a significant CPU allocation on them
    - The number of volunteers in ATLAS@Home is growing steadily
- A significant increase in usage of opportunistic resources is expected in 2017 and 2018.