An OSG-based distributed Campus Computing Infrastructure

Christoph Paus, MIT
CHEP Meeting, San Francisco
Oct 10-14, 2016

Ray an Maria Stata Center at MIT

Introduction Campus Computing

Description of generic campus situation (*ex.* MIT)

- Large variety of research areas: engineering, maths, sciences, social sciences ....
- Most need large computing *at times* and have some computing resources
  - some resources shared but not widely, usage not 100% for all of them
  - no accurate inventory of all existing resources
- Most resources use some linux variant of similar versions

Some Issues

- Researches have peak demands that exceed their resources
- Often though the resources are not fully used
- Account management is work intensive

Introduction Campus Computing

Why not have one big computing center?

- Could work, but it does/did not: ownership, funding, different interests, management ....
- Existing resources would need to be moved
- Difficult to change Status Quo

Requirements for a new model

- Minimally expensive: money and human resources
- Technically feasible and attractive for most research efforts
- Use all existing computing resources
- Leave computing resource owners maximal control
- Reach beyond campus as needed/wanted

The Virtual Computing Center

Pretending to be a big computing center

- Create a common login pool, big enough for tests
- Connect this pool to each campus computing resource
- Also connect to external resources: *ex.* OSG

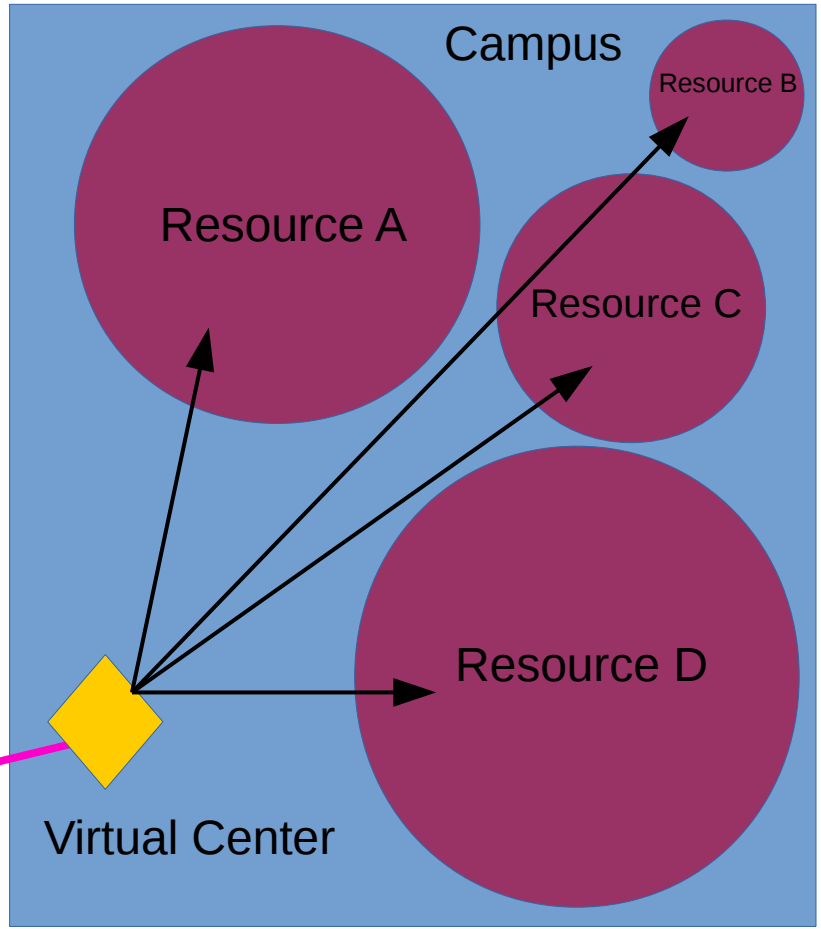
Who would need to change?

- Resource owners, administrators, users
- In short … everybody :-(
- But by how much and to what advantage?

The Virtual Computing Center

Conceptually simple

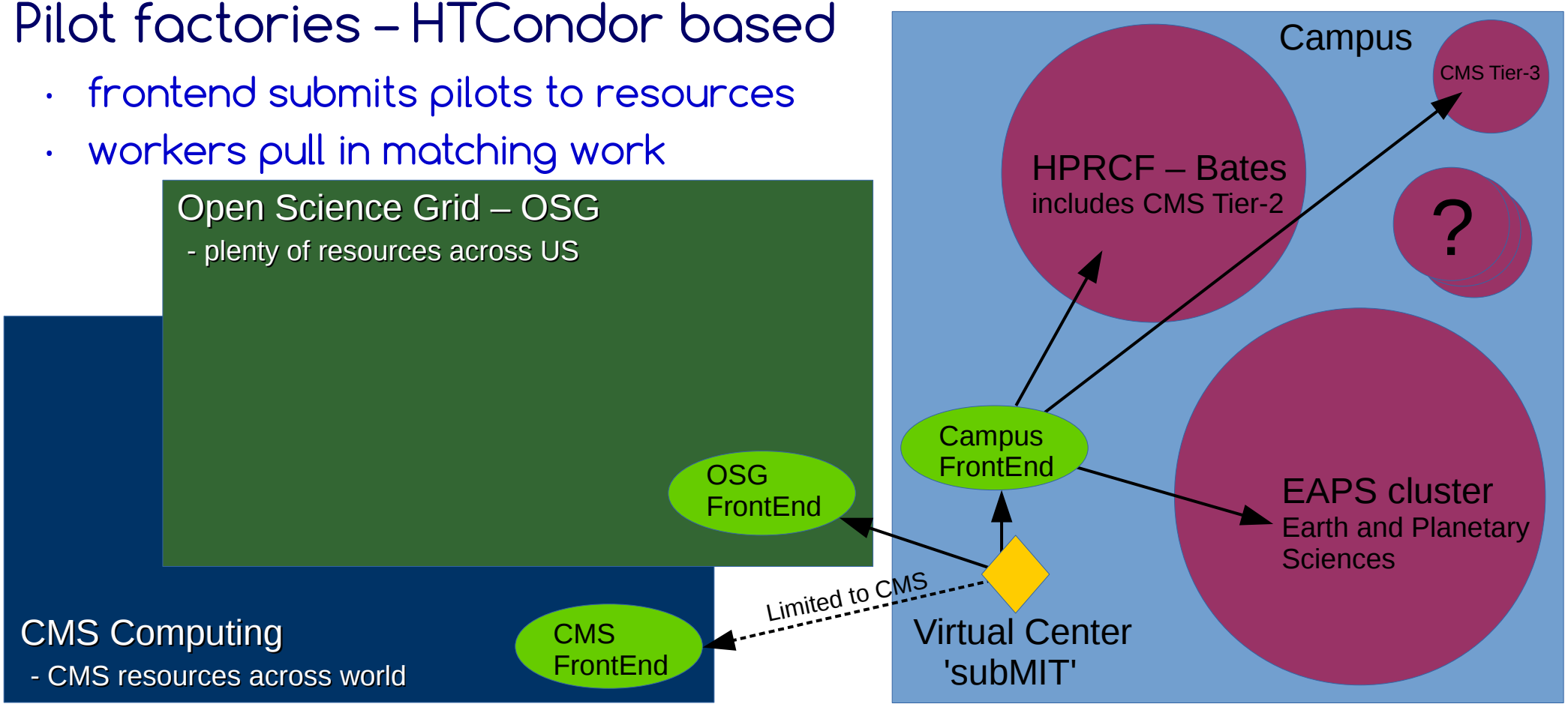
- Users logon to virtual computing center
- Setup their task
- Launch the task
- System distributes task to resources as requested
- Work with the output



Implemetation Details – MIT

Pilot factories – HTCondor based

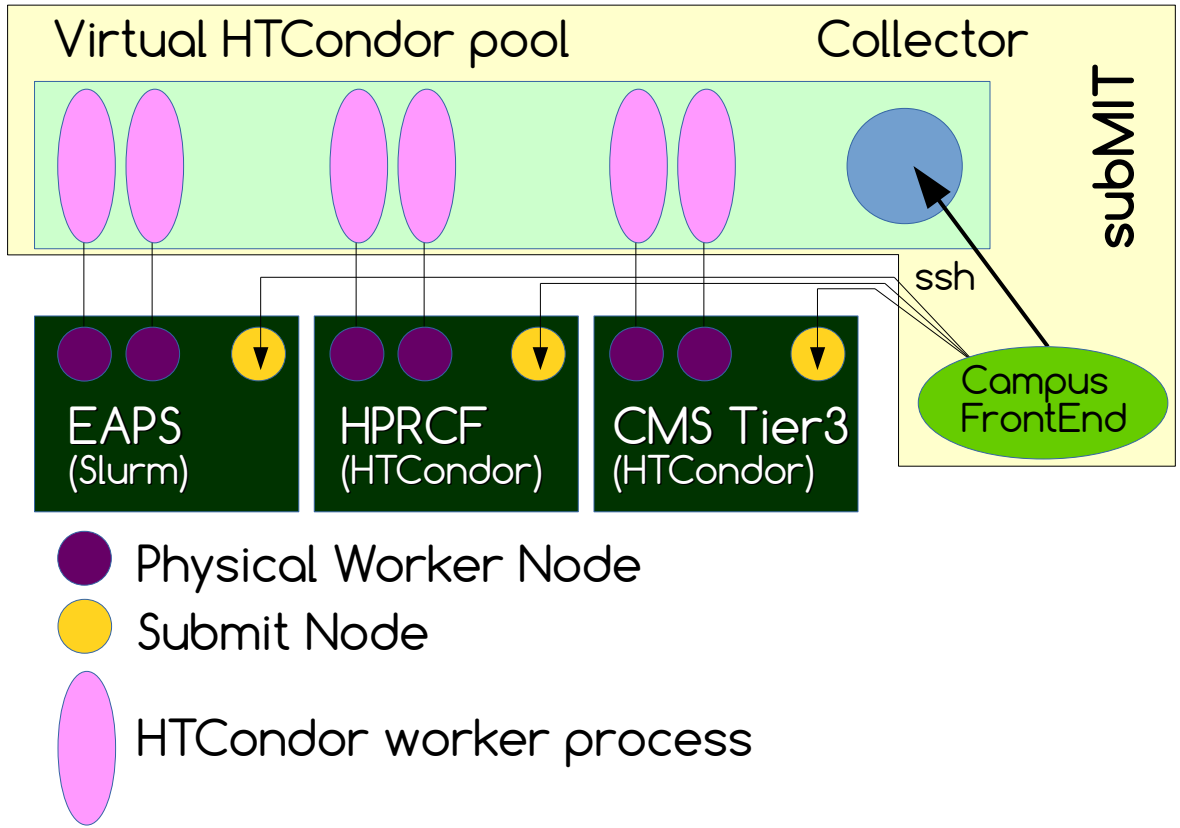
- frontend submits pilots to resources
- workers pull in matching work



Implementation Details – MIT

How are jobs running?

- FrontEnd submits glideIn pilots through BOSCO to the various resources submit nodes (local flavor)
- On subMIT user jobs get submitted to a HTCondor collector
- Physical workers are matched at subMIT and pull down their work
- subMIT becomes a huge virtual resource, real work is done at the physical worker
- At completion output is shipped as specified



Implementation Details – MIT

User perspective

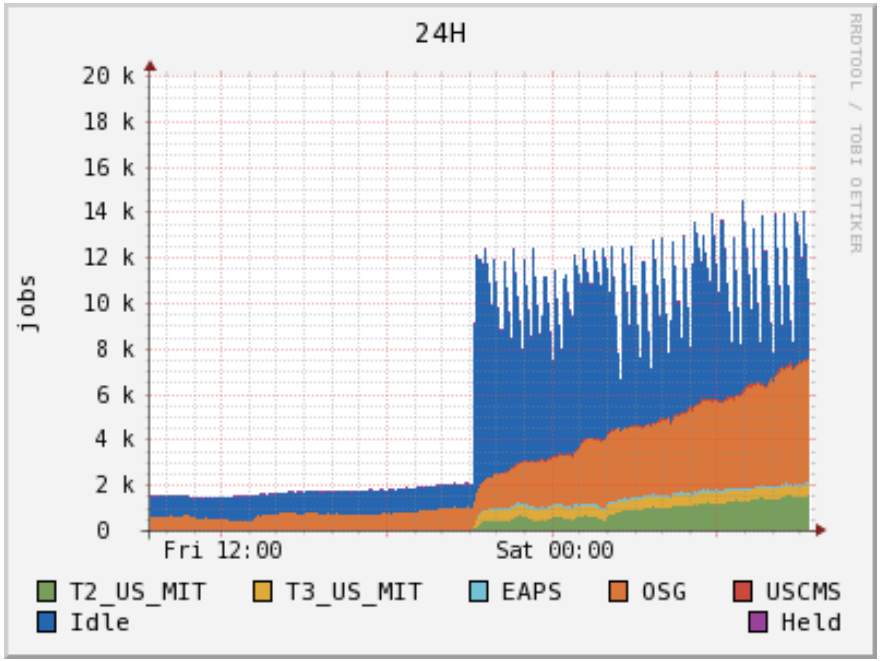
- Request access to campus computing
- One beefy machine: ssh subMIT.mit.edu (could be a bunch of machines)
- At login people land in their afs home
- Local work area provided: O(10 GB) /user
  - Can mount dropbox *etc.*
- HTCondor submission as usual with all basic monitor/debug
- testBed runs on machine itself (HtCondor short slots, fast turnaround)
- Specify running location / requirements in condor job configuration as usual: details to use various resources are documented
- To match to the outside (OSG) a project will have to be declared with OSG to allow for some monitoring/accounting (approval is simple)

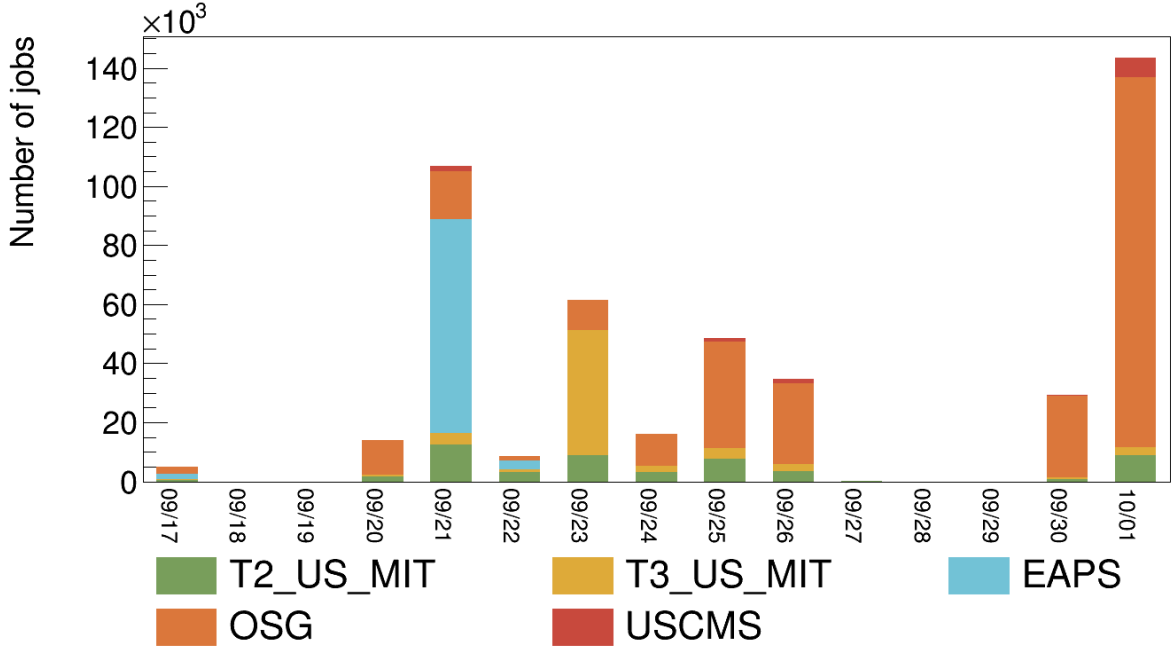
Implementation Details – MIT

Resource Owner's/Administrator's perspective

- Access to the resources are granted through service accounts
- Service account submits in local flavor to the batch system and can be managed: privileges/priorities *etc.*
- Typical service accounts:
  - for resource owners (get full access)
  - for visitors from MIT (opportunistic access)
  - for visitors from off-campus (opportunistic access, pre-emption)
  - there are many options possible depending on the wishes of the owners
- CVMFS is used to distribute bigger software

Prototype – Campus Computing

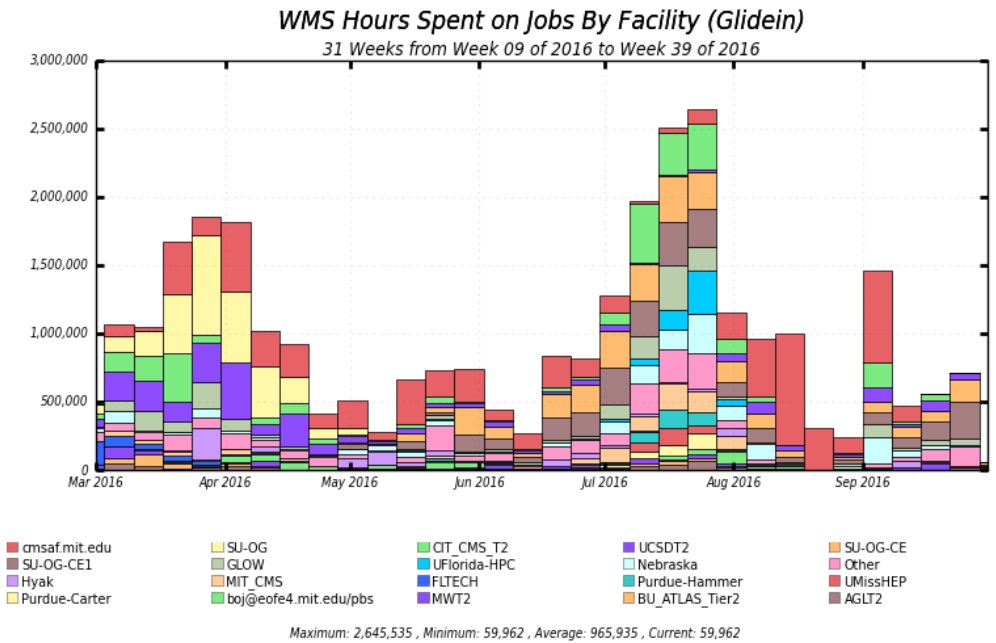


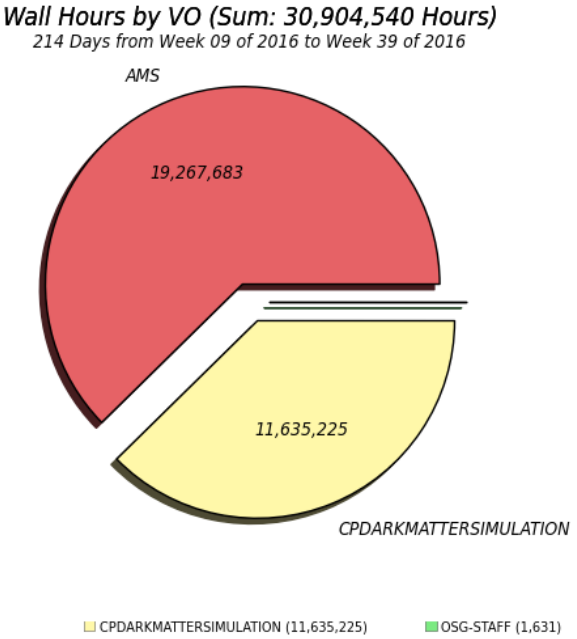


Three campus resources connected (6 prototype users)

- T2 at Bates, T3 in B24, and EAPS at the Green Center in Holyoke
- Also seamlessly integrated the OSG access

Prototype – OSG Impact





Uses of our virtual computing center (Campus + OSG)

- About 1 million CPU hours per week for 31 weeks
- 19 M computing hours for cosmic particle simulations (AMS) and 12M for Dark Matter simulation in pp collisions (CMS)

Conclusions

The Virtual Computing Center

- Viable, pragmatic solution for generic campus computing
- Covers most use cases, *but not all*
- Allows maximal flexibility: all resources can be separately registered and used, but also controlled by owners
- Specific fully functional prototype implemented at MIT using OSG based tools: HTCondor, bosco, glideInWMS pilots

What next?

- Some investment needed to establish infrastructure and support
- Users need to re-learn some, but win big
- Resource owners need to be convinced and have to adjust
- Need to find all resources on campus and connect them