Condor in the CMS Experiment

Burt Holzman
Condor Week 2009
April 21, 2009

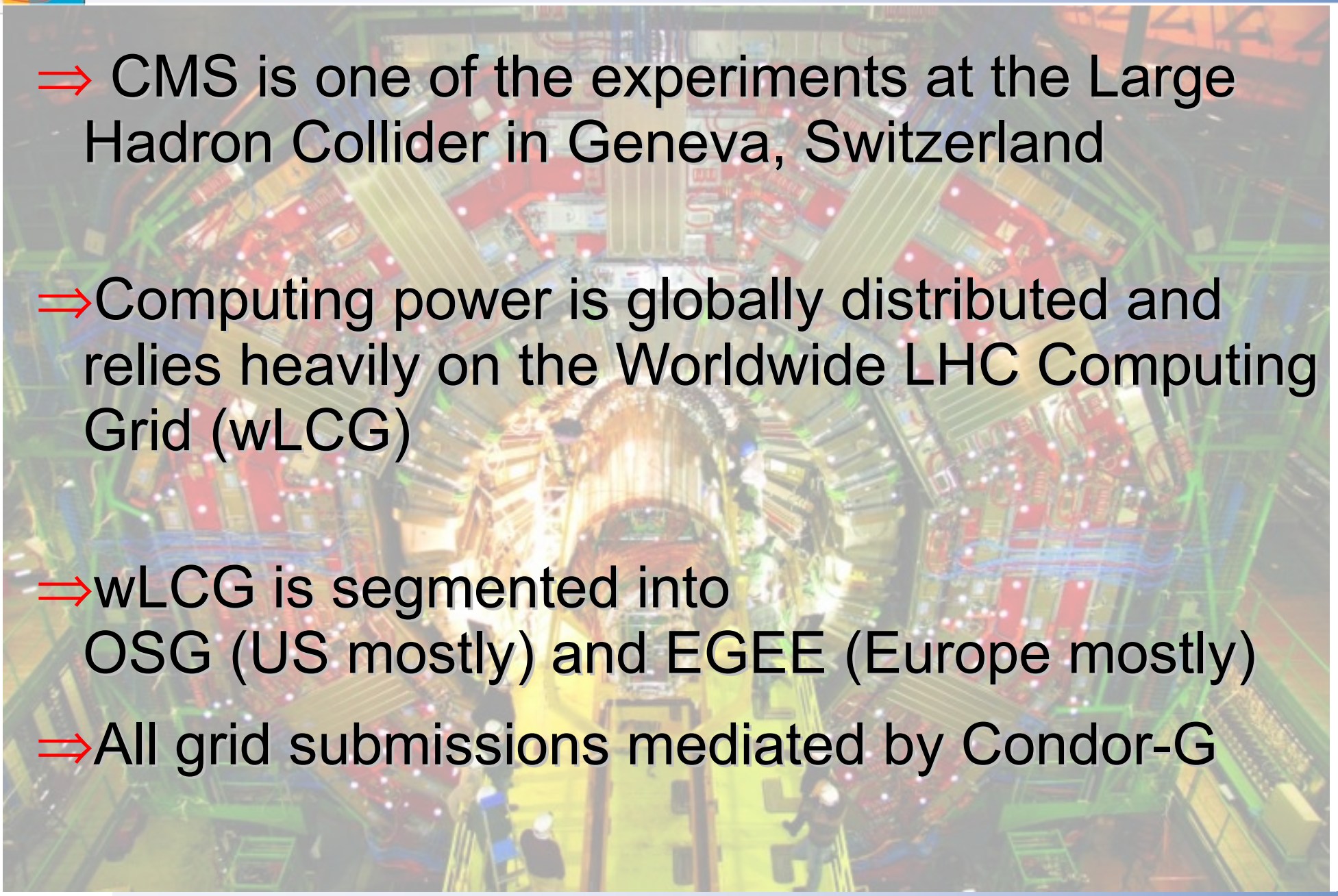
Preface (courtesy of Miron)

Condor-G
Glide-ins
Matchmaking

carry the load of
EGEE and OSG

www.cs.wisc.edu/Condor

CMS: Compact Muon Solenoid



⇒ CMS is one of the experiments at the Large Hadron Collider in Geneva, Switzerland

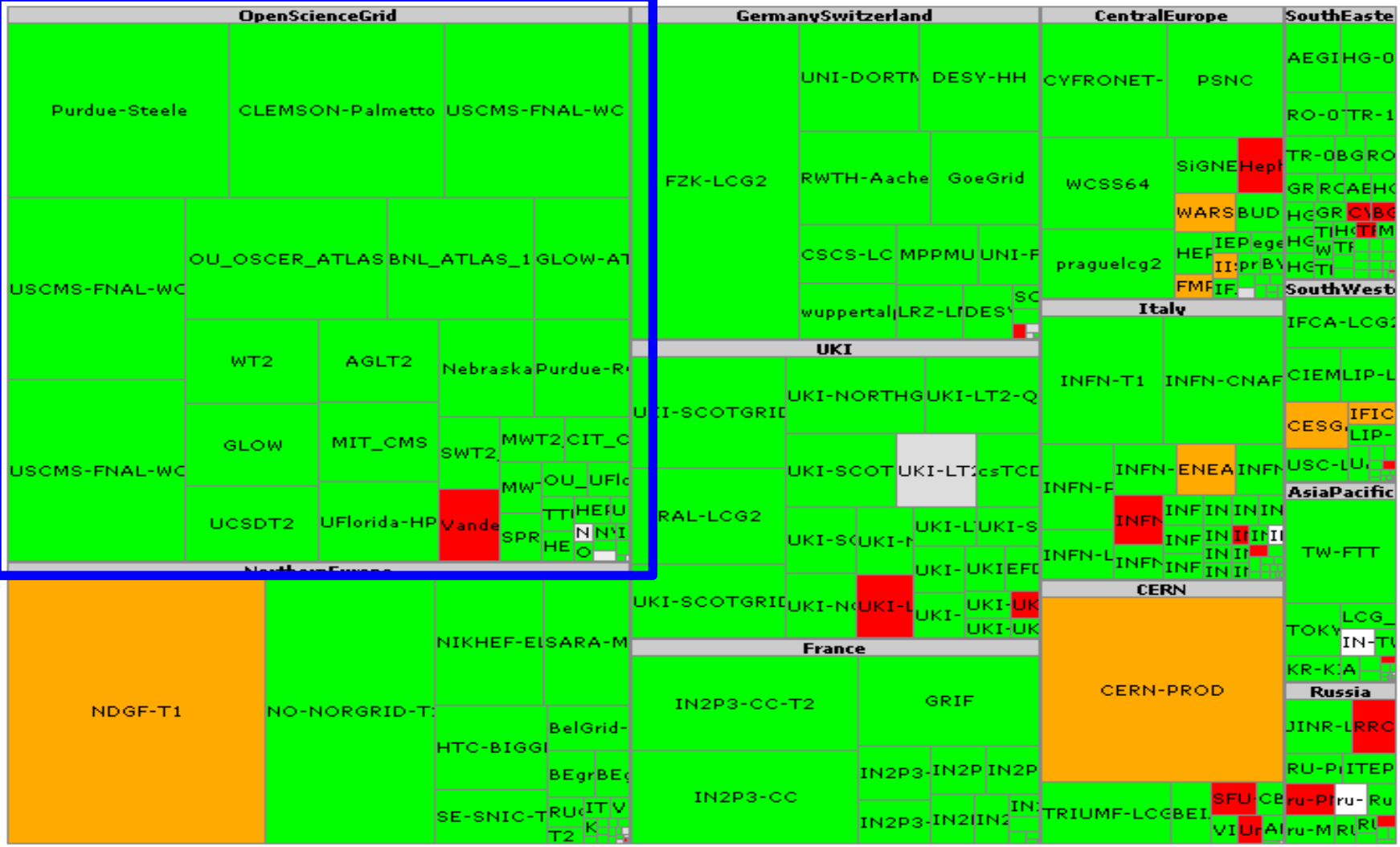
⇒Computing power is globally distributed and relies heavily on the Worldwide LHC Computing Grid (wLCG)

⇒wLCG is segmented into OSG (US mostly) and EGEE (Europe mostly)

⇒All grid submissions mediated by Condor-G

WLCG Cartogram (number of slots)

OSG



Latest SAM results, CE Status, for 'OPS' VO, 20 Apr 2009 15:17 GMT.
Size of site rectangles is number of CPUs from BDII.
Certified Production sites, grouped by regions.

Maint Down Degraded Ok

Condor at three different scales

⇒ "Small"-scale (local)

⇒ CMS Tier 1 at Fermilab

⇒ thousands of nodes, ~6800 batch slots

⇒ Mid-scale (regional)

⇒ American CMS Tier 2s (7 US + 2 Brazil)

⇒ Plus the Tier 1 above; ~12k batch slots

⇒ Large-scale (global)

⇒ Worldwide CMS facilities

⇒ Seven Tier 1s, many Tier 2s; > 30k slots*

CMS @ “Small”-scale: FNAL Tier 1

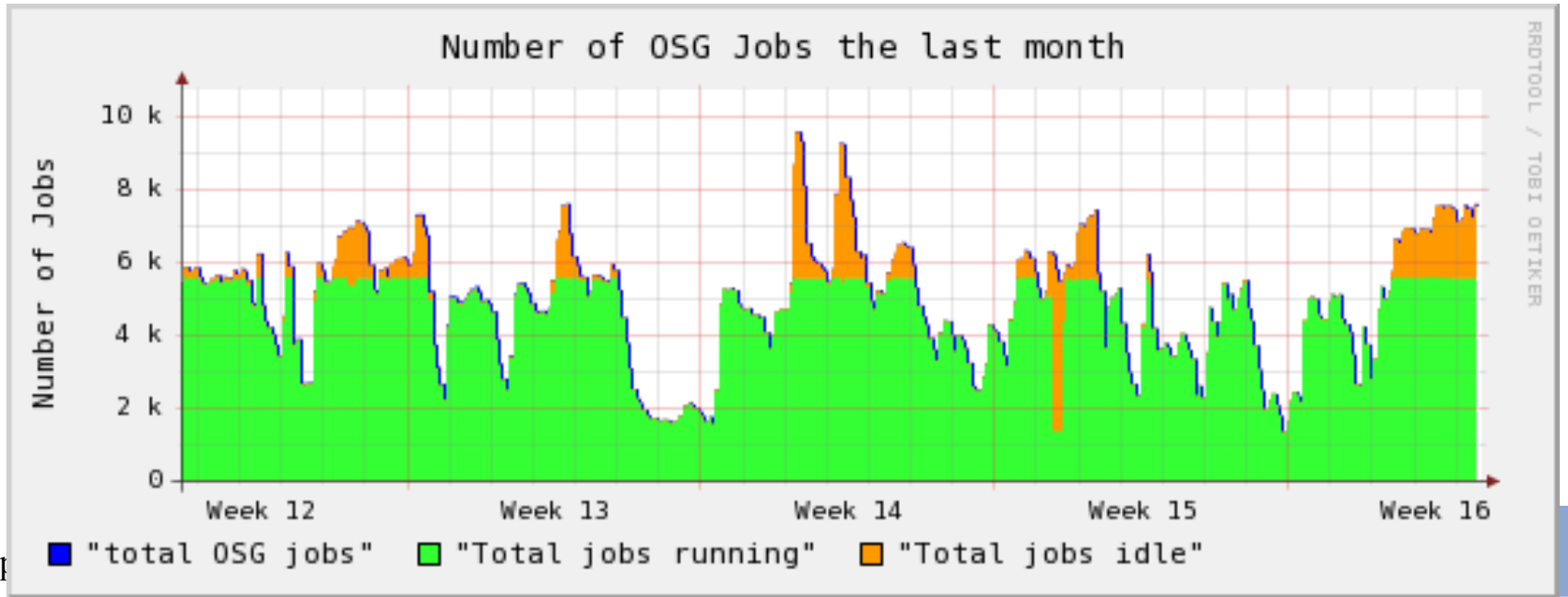
CMS @ “Small”-scale: FNAL Tier 1

⇒ One of the largest single Condor pools in the world

⇒ Four schedulers running Open Science Grid gatekeeper software

⇒ Over 1000 worker nodes and 6800 slots*

⇒ One negotiator/collector machine (of course!)



CMS @ "Small"-scale: FNAL Tier 1

- ⇒ Schedd / Shadow communication is high priority
- ⇒ Newer Condor is much better at this (non-blocking I/O, child processes)
- ⇒ Condor daemons **must** have generous timeslice in kernel land
  - ⇒ Renice +20 for xinetd, globus gatekeeper (makes heavy use of system CPU)
  - ⇒ Otherwise, shadows and schedds perform the "dance of death"
    1. Shadows timeout and die
    2. Schedd spawns new shadows, increase load
    3. New shadows timeout and die

CMS @ “Small”-scale: FNAL Tier 1

⇒ Busy cluster, thousands of idle jobs: negotiator cycles still short due to auto-clustering

⇒ Jobs coming from grid have homogeneous requirements in general

⇒ Job or node failures can lead to thousands of idle jobs with different ImageSize and DiskUsage

⇒ No more auto-clustering -- every job for himself!

⇒ Solution: script/cronjob to condor_qedit these attributes so auto-clustering can resume

CMS @ “Small”-scale: FNAL Tier 1

⇒ UDP updates still preferred

⇒ Easy to lose simultaneous updates

⇒ Spread out worker updates

⇒ UPDATE_INTERVAL = $RANDOM_CHOICE(490, 500, ... , 510)

⇒ Increase tolerance for missed updates

⇒ CLASSAD_LIFETIME = 1800

⇒ Use job leases in submit JDF (even if we drop a machine's ClassAd, we still have a chance!)

⇒ JobLeaseDuration = 7200

CMS @ “Small”-scale: FNAL Tier 1

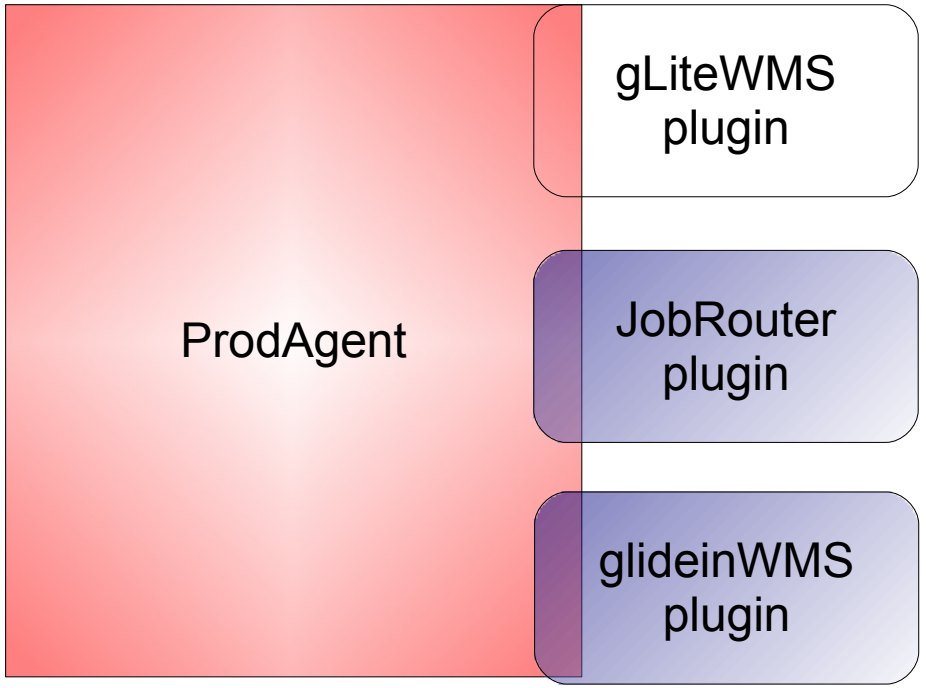
⇒ Avoid NFS whenever possible

- ⇒ On OSG, that means “condor-nfslite” and no shared disk
- ⇒ On our other pools (with shared disk), we still have different FILESYSTEM_DOMAIN on the schedds and startds

⇒ Does your monitoring scale?

- ⇒ Monitoring that works for 100 nodes may perform terribly with 1000

CMS Workload Management Systems



gLite UI → gLite WMS (with **Condor**)

Condor job router

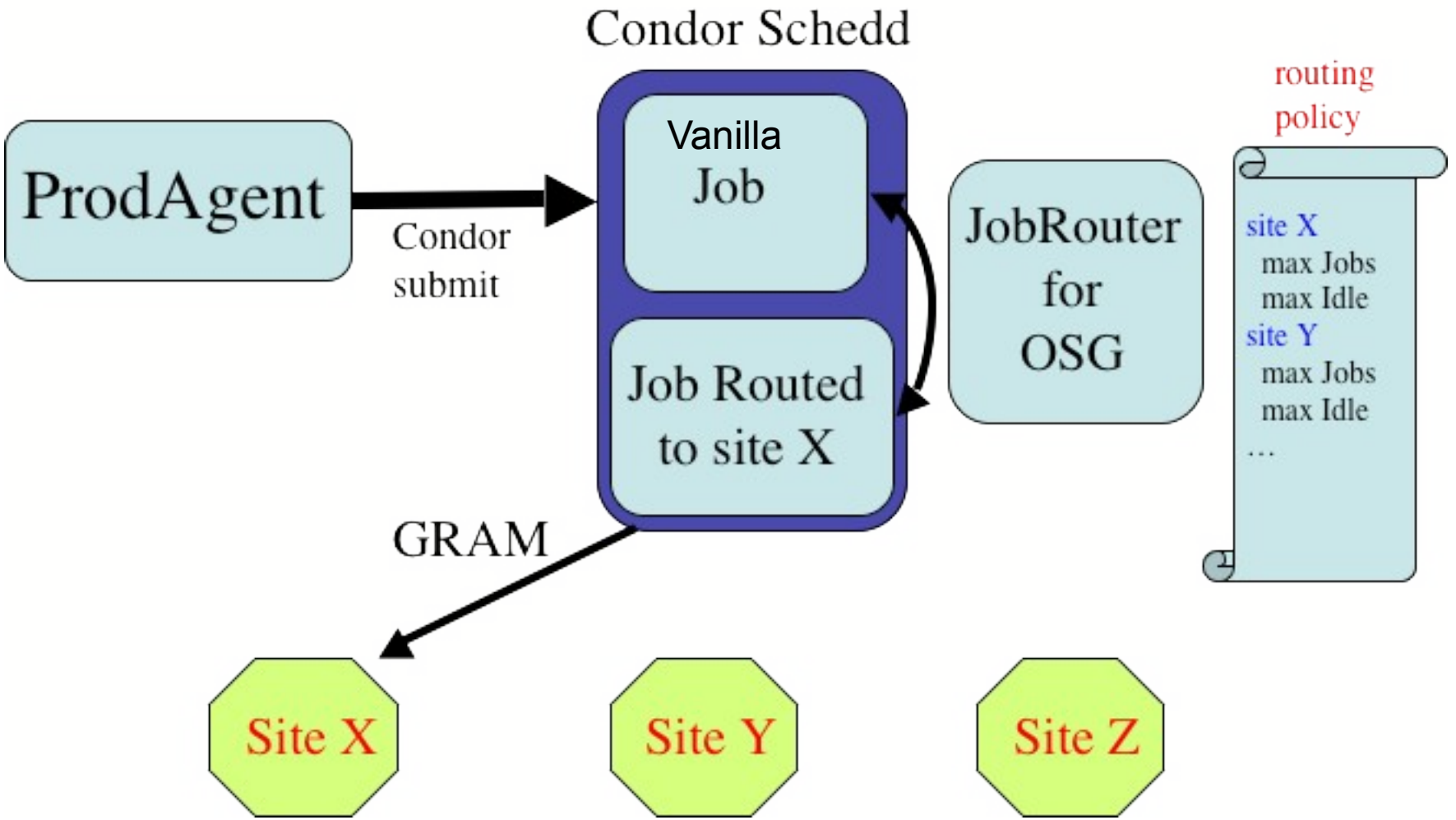
Condor glide-ins

⇒ CMS has not blessed a chosen WMS

⇒ Different solutions at different scales

⇒ Sometimes different solutions at the same scale also

CMS @ Mid-scale: JobRouter



CMS @ Mid-scale: JobRouter

⇒ Transforms vanilla jobs into grid jobs

⇒ Destination site list is fixed (but could be dynamically discovered)

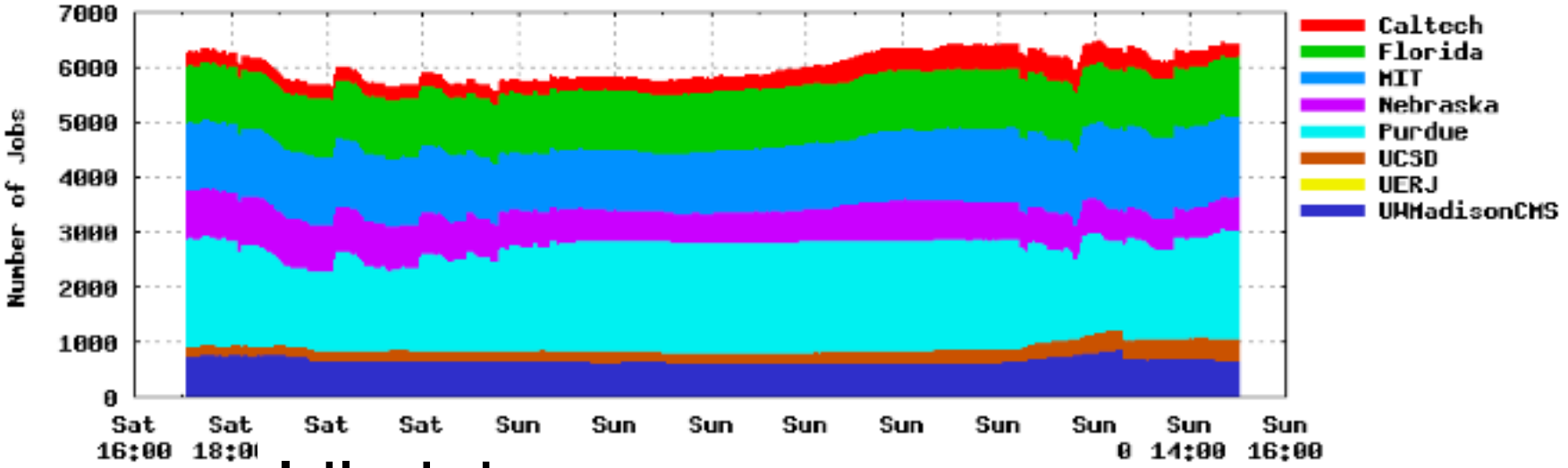
⇒ Can potentially avoid blackhole sites (see D. Bradley CHEP 09 talk)

⇒ Submits only to OSG (unable to submit to EGEE)

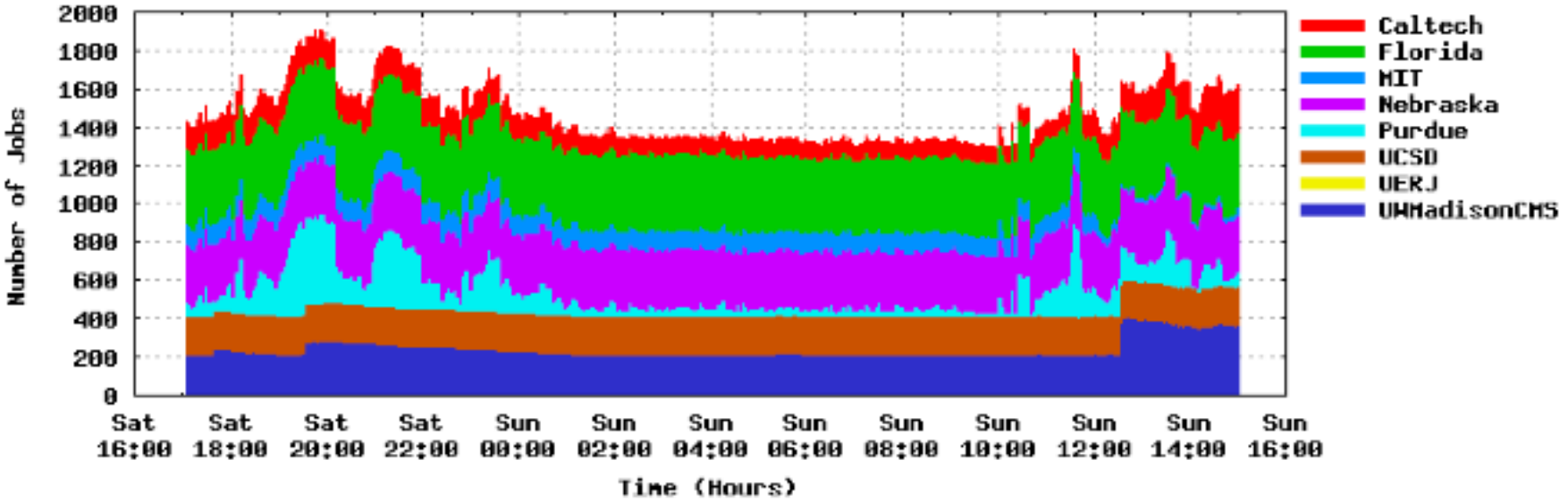
⇒ Should scale up to Condor-G limits

CMS @ Mid-scale: JobRouter

Running jobs

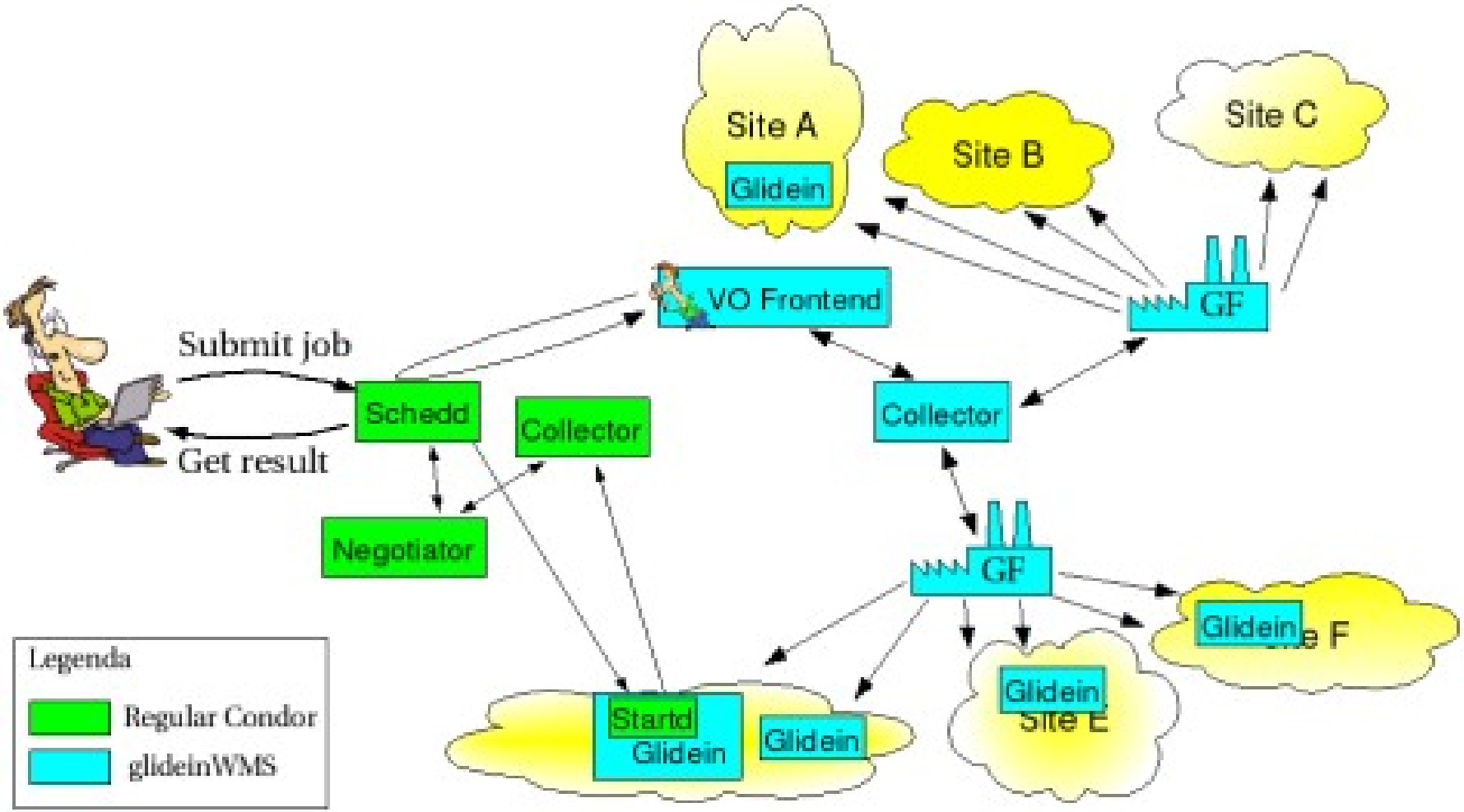


Idle jobs



CMS @ Large-scale: glideinWMS

⇒ glideinWMS: workload management system based on condor glideins



CMS @ Large-scale: glideinWMS

⇒ glideinWMS: workload management system based on condor glideins

⇒ Pilot jobs land on grid sites and form a “virtual condor pool”

⇒ Pilots can run multiple jobs serially: reduces grid latency

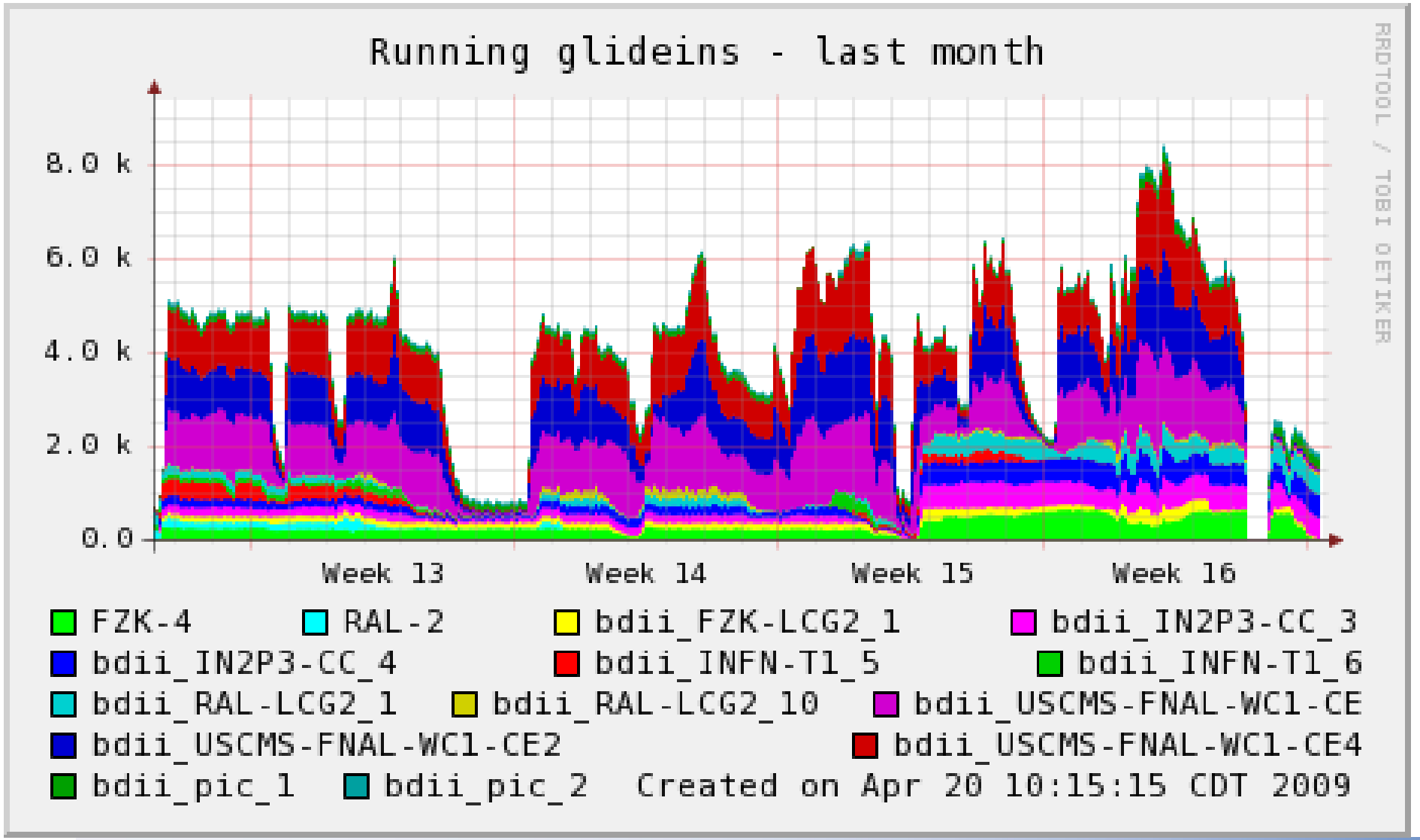
⇒ Workload in distributed fashion instead of through the grid gatekeeper choke-point

⇒ glideinWMS can submit to both OSG and EGEE

⇒ glideinWMS supports pseudo-interactive monitoring

CMS @ Large-scale: glideinWMS

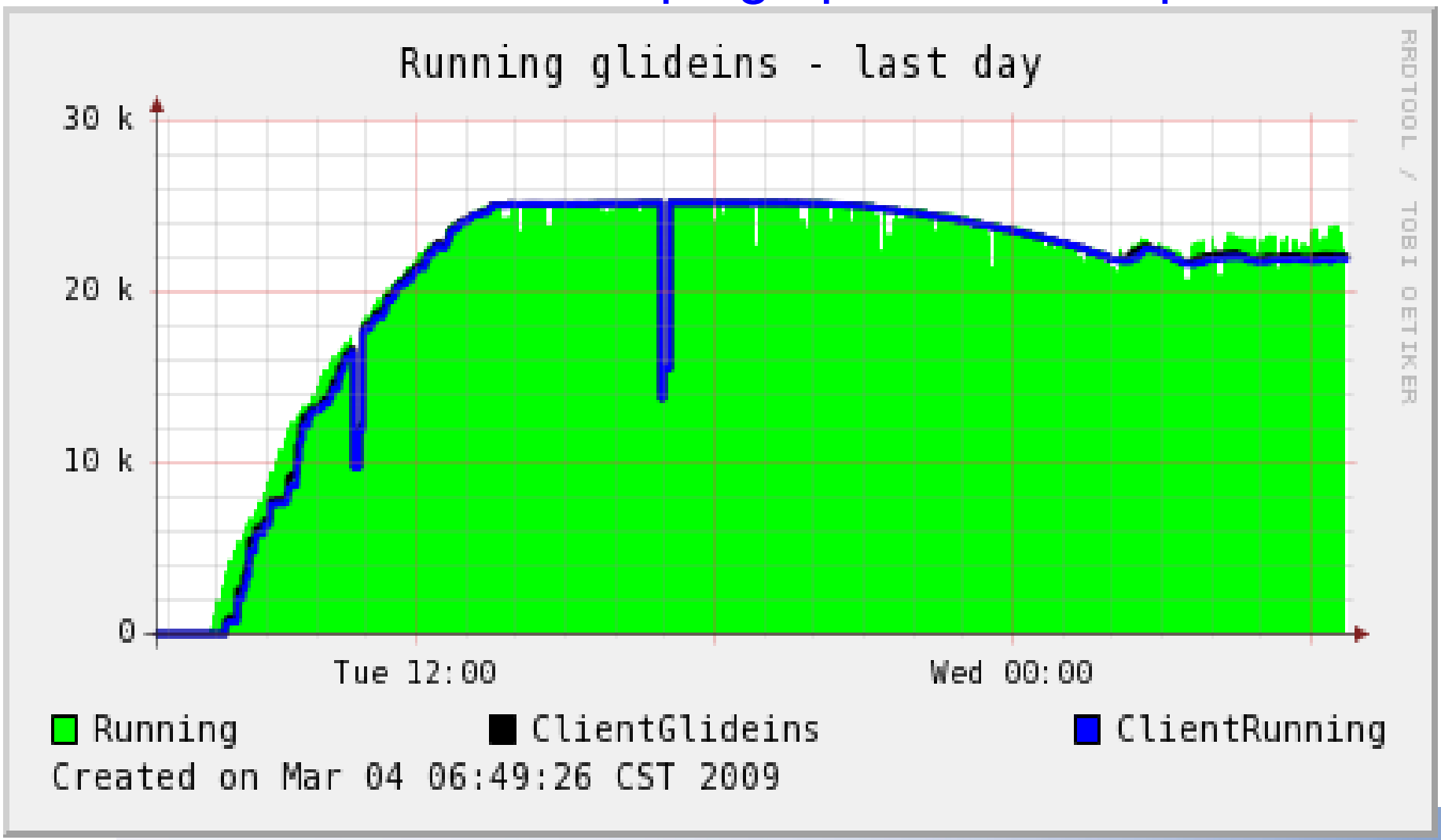
⇒ Currently in production



CMS @ Large-scale: glideinWMS

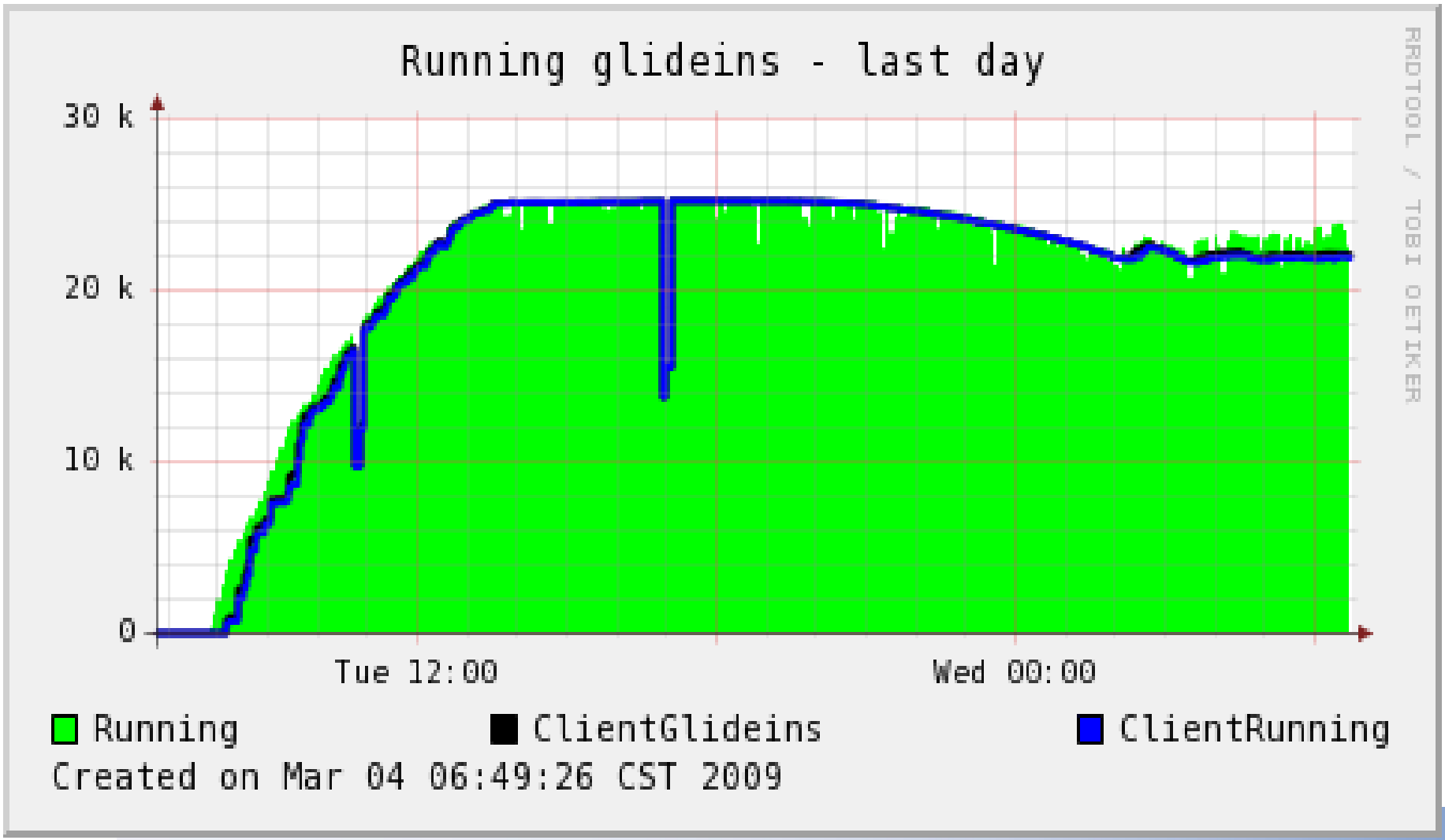
⇒ Proof-of-principle instances scale to 25k

⇒ We will be ramping up to > 10k in production



CMS @ Large-scale: glideinWMS

⇒For the details, see Dan Bradley's talk on Wed.



The End

⇒ Questions?

⇒ CMS: http://cms.cern.ch

⇒ glideinWMS: http://tinyurl.com/glideinwms