CDF's experience with Condor in a large-scale grid environment

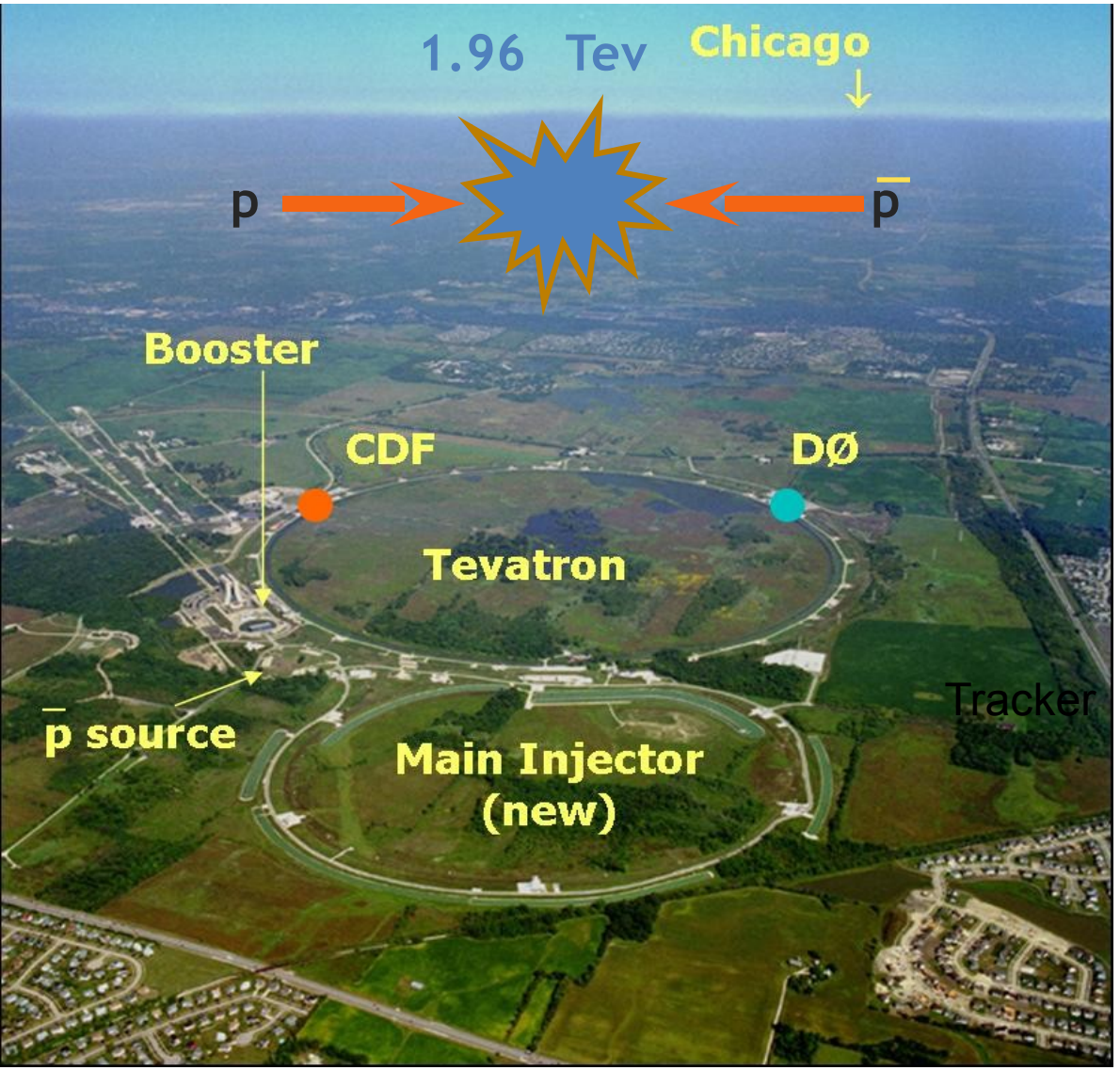
Dennis Box
Fermilab

April 21 2009

Outline

- Introduction
- CDF, Data flow, Computing Model
- The CAF
- Early CAF Implementations
- Transition to GRID
- Glideins
- GlideCAF
- GlideinWMS
- General Observations
- Conclusions

Collider Detector at Fermilab (CDF)



CDF :

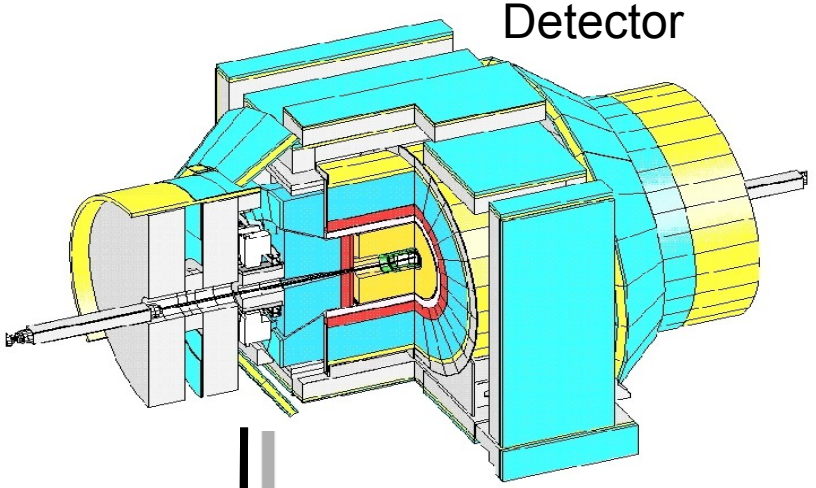
- Particle Physics Experiment at Fermi National Accelerator Lab (Fermilab)

-Started collecting data in 1988

•Data taking will continue until at least 2010 – likely through 2011

•Heavy condor user since 2004

Data Flow



Detector

Trigger
Filtering

Tape Robots
~10 PByte

Disk Staging
Area

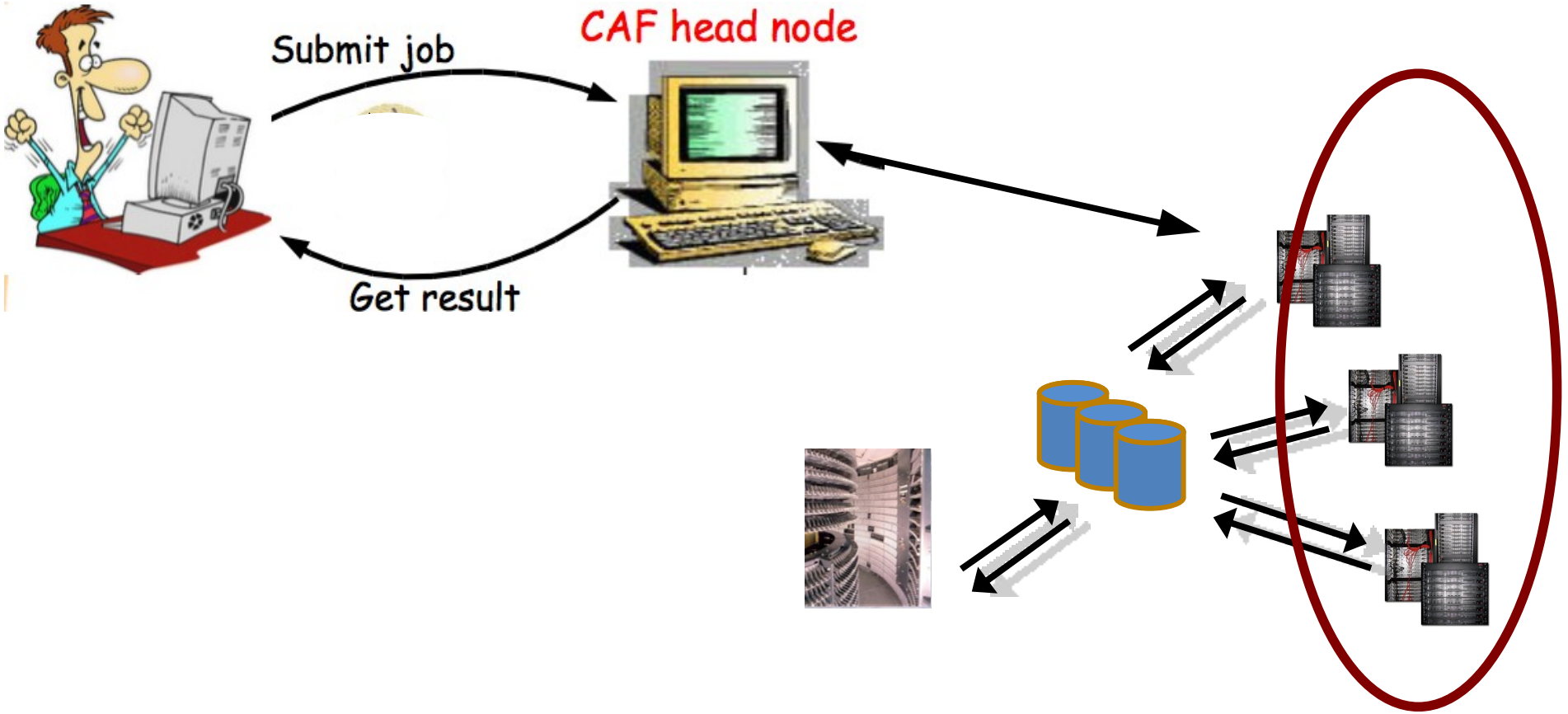
~10TByte/Day

~30TByte/Day

Batch Cluster
Data Analysis

Computing Model

- Users develop on linux workstations, which have same interface to data handling as batch nodes
- Users then submit to batch systems using a head node or portal known as the CAF

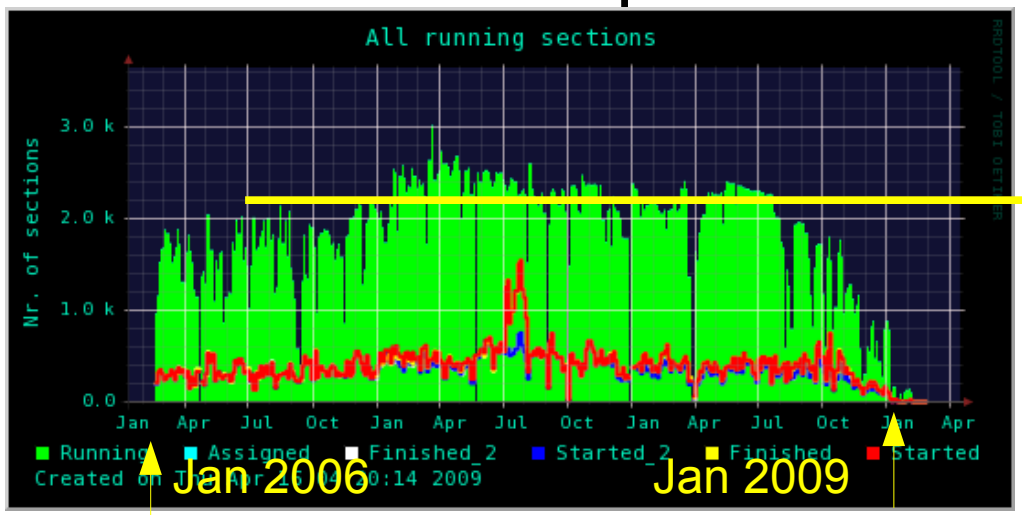


The CAF

- CDF Analysis Farm
- Submission to head node hides differences between batch systems
  - Kerberos authentication to head node only, submission deamons take over authentication from there (BIG WIN on the GRID!)
  - Uniform way to monitor jobs progress
    - Through web pages
    - Or command line tools
- Delivery of results directly to users desktop or to large output pool if desired
- Email notification when jobs complete

Early CAF Implementations

- Early implementations FBNG batch system
- First Condor implementation 2004-2009, using dedicated condor pool



2K slots

The CAF model was successful and proliferated over many CDF affiliated sites

Transition To The GRID

- Motivation:
  - There are idle resources out there that we can use!
  - The people who pay for this made us do it
- The good:
  - There are idle resources out there that we can use!
- The bad:
  - We have to share our toys
  - It is much more complicated to make work reliably

Transition To The GRID

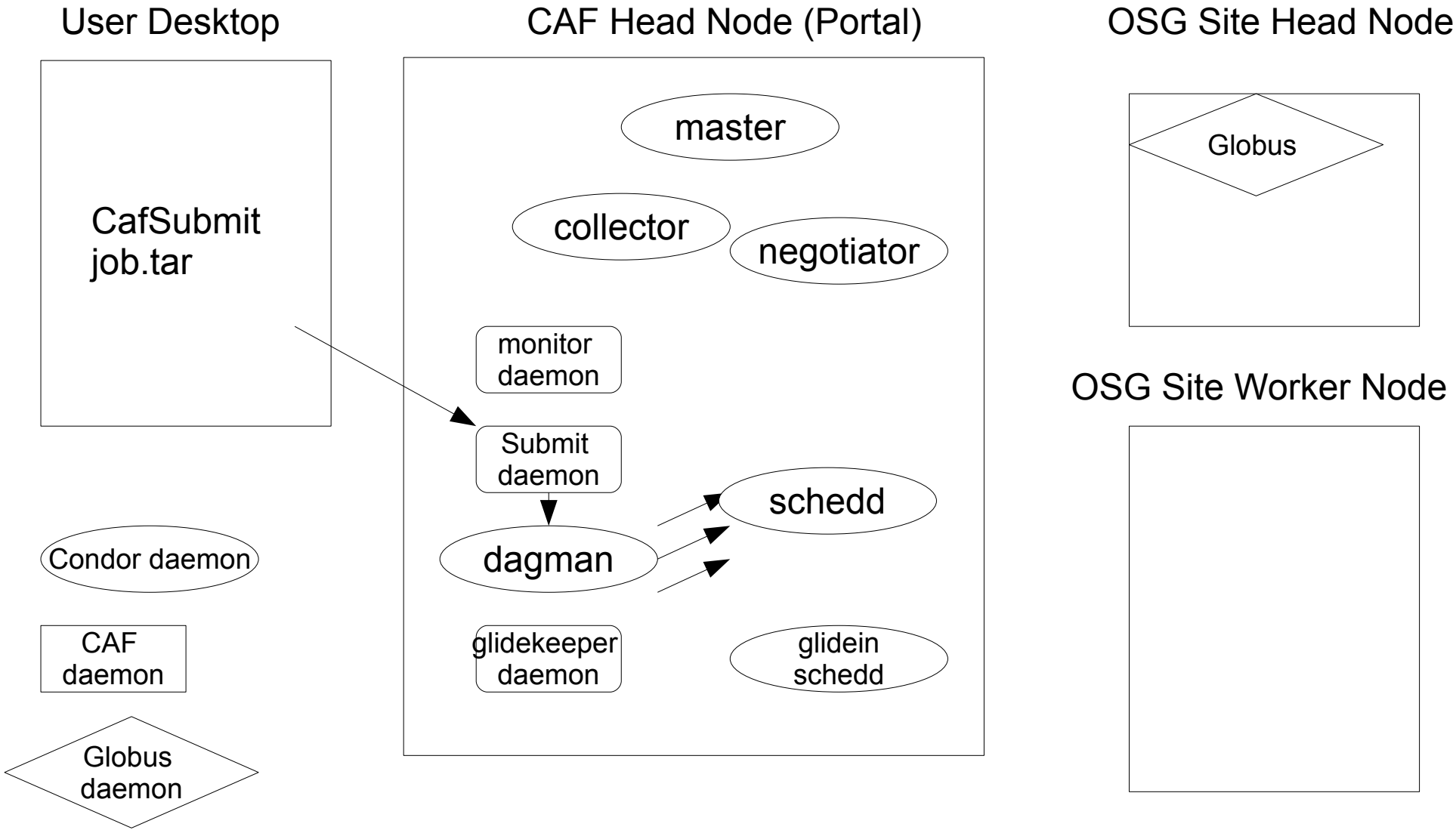
- Requirements:
  - Minimize disruption to users so they can concentrate on their jobs (physics)
- Strategy:
  - Adapt the CAF model and user toolset
  - Use kerberos authentication for users as before
    - Kx509 authentication across the grid
- Implemenation:
  - Modify the CAF headnode to interface to GRID middleware (OSG or gLite)
  - Use glideins

Glideins

- Condor startd, ClassAd, and Globus software configured to 'phone home' to our collector
- We submit glideins instead of user jobs to batch system
- Batch system runs it like any other job
- Starter on worker node receives job tarball via squid
- Globus GLExec authenticates user on remote site, starts job execution
- Looks like 'normal' condor job to most of our daemons

GlideCAF

- User submits job (CafSubmit – options)
- Submit daemon queues job ,creates job.dag authenticates, submits
- Dagman orders sections, condor_submits to schedd



GlideCAF

- Schedd notifies collector of jobs
- Glidekeeper polls schedd and notices pending jobs

User Desktop

Condor daemon

CAF daemon

Globus daemon

CAF Head Node (Portal)

master

collector

negotiator

monitor daemon

Submit daemon

schedd

dagman

glidekeeper daemon

glidein schedd

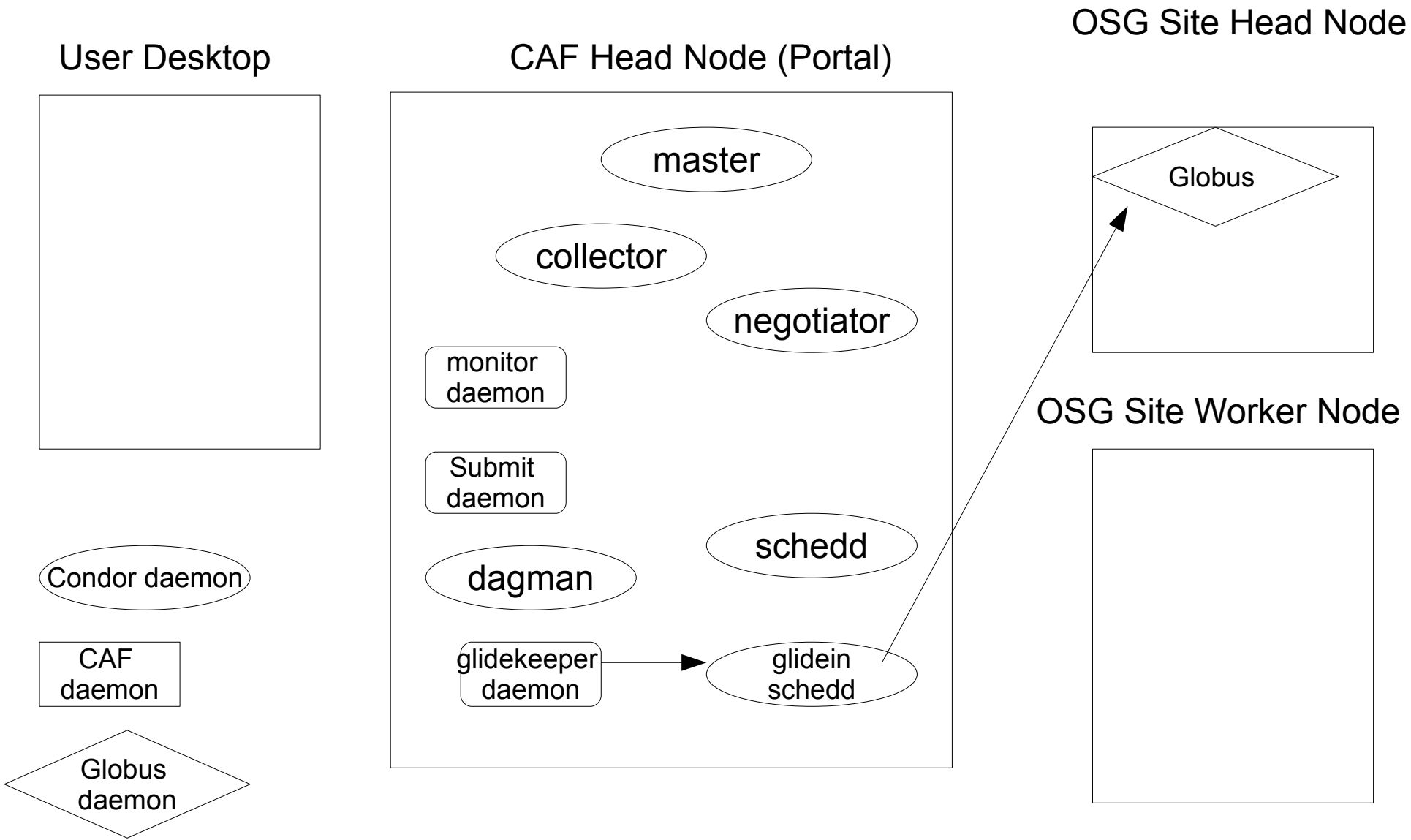
OSG Site Head Node

Globus

OSG Site Worker Node

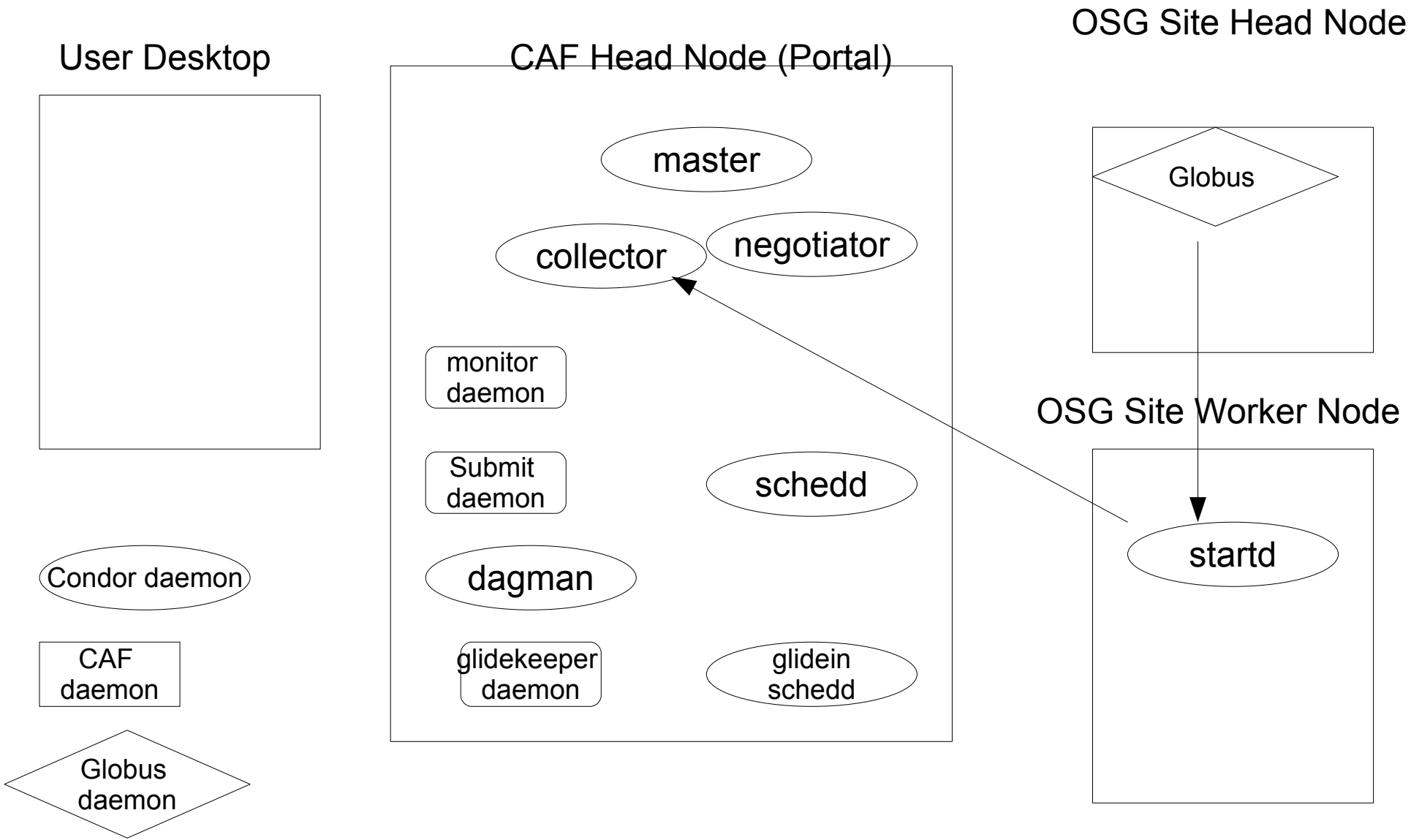
GlideCAF

- Glidekeeper condor_submits to glidein schedd
- Glidein schedd uses globus-job-submit to send glidein to GRID Head Node



GlideCAF

- Globus notifies GRID batch system, which starts glidein
- Glidein startd notifies colllector its ready to start a job



GlideCAF

- Negotiator polls collector, sees a match
- introduces startd and schedd
- Startd and schedd agree to run a job

User Desktop

CAF Head Node (Portal)

OSG Site Head Node

Globus

OSG Site Worker Node

master

collector

negotiator

monitor daemon

Submit daemon

schedd

dagman

startd

glidekeeper daemon

glidein schedd

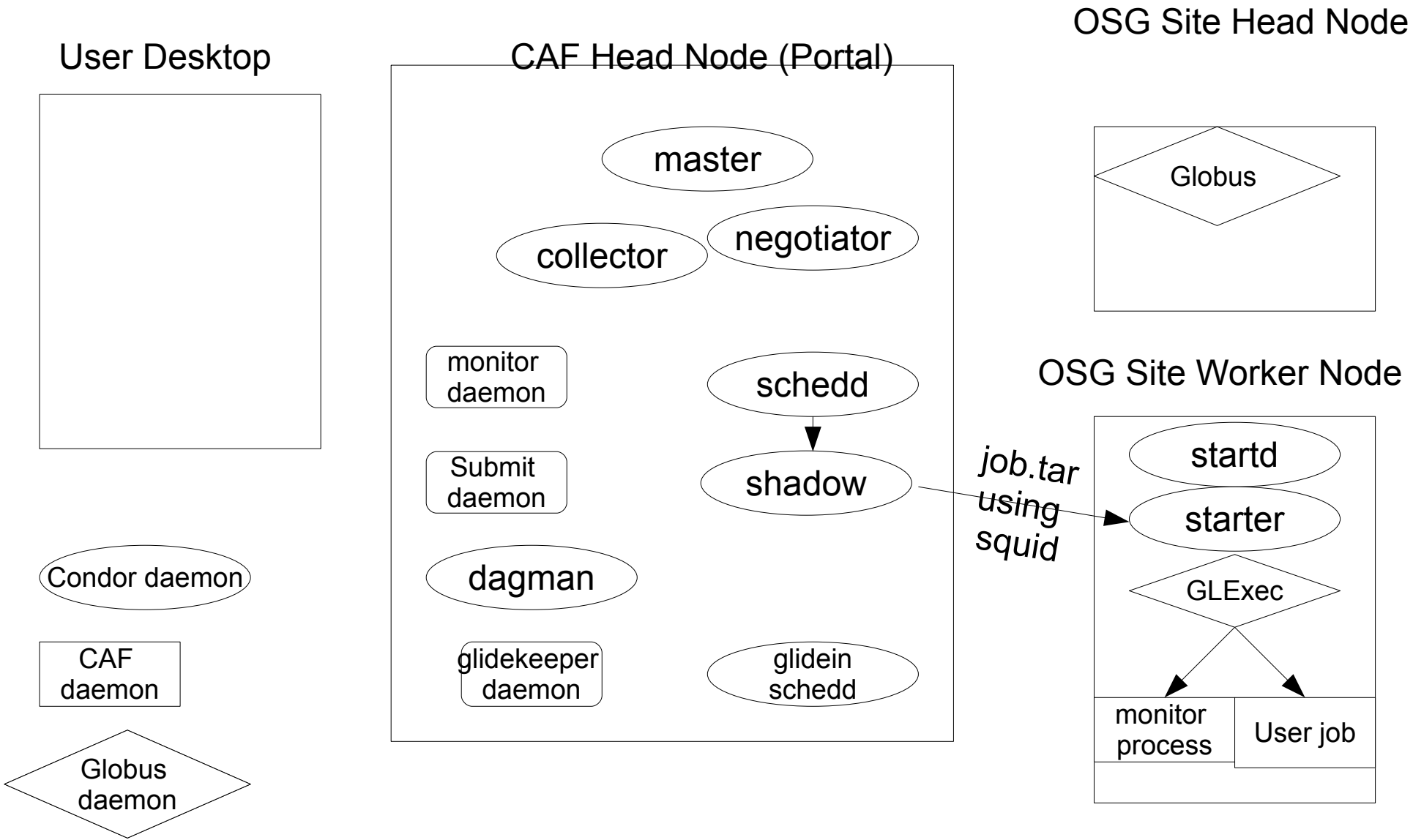
Condor daemon

CAF daemon

Globus daemon

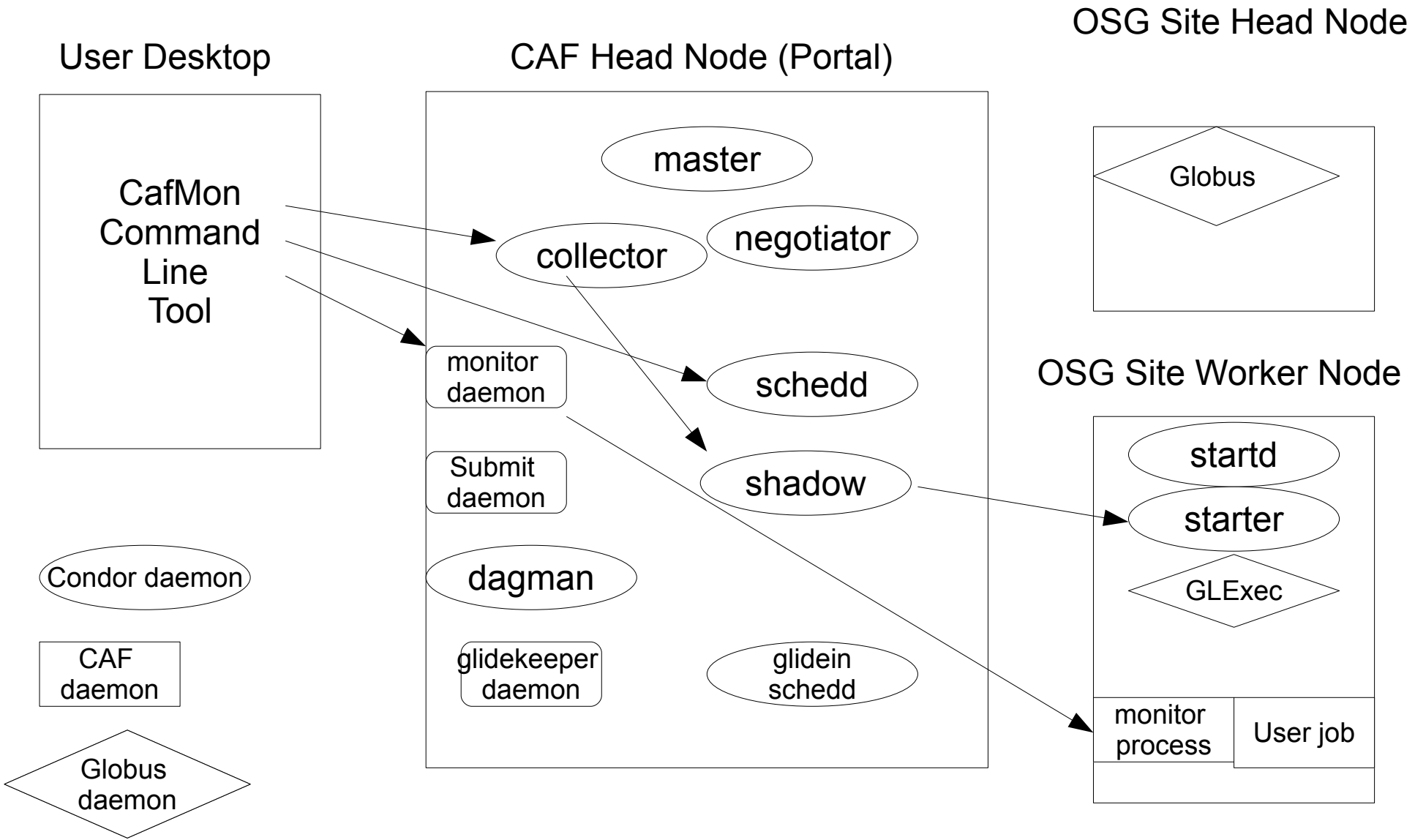
GlideCAF

- Startd spawns starter spawns gleexec
- Schedd spawns shadow process
- Shadow passes job.tar to starter using squid
- Glexec authenticates and starts monitor process & (finally!) User Job



GlideCAF

- Monitoring with command line tool CafMon
  - CafMon talks to collector using condor_status
  - Talks to schedd using condor_q
  - Talks to worker node using condor COD and monitor daemon



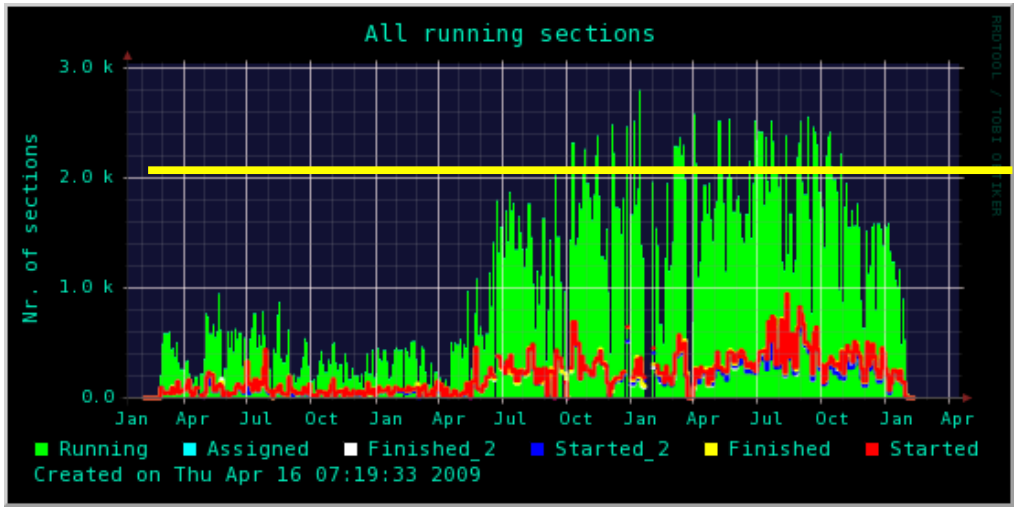
GlideCAF Notes

- This description is simplified
  - 3 monitor daemons, other CAF daemons
  - Multiple schedds, glidein schedds, collectors
  - Ignored authentication
  - Ignored GCB/firewall issues

- All of these daemons create log files, which are sometimes the only way to figure out what happened when something goes wrong.

First GlideCAF Implementation

- CAF middleware, schedds, glidein schedds on head node
- Collector, Negotiator on second head node
- Worker class machines - drives 2k slots sustainable
- Worker nodes actually on site, NFS mounted home area, libraries (but other sites can use them)

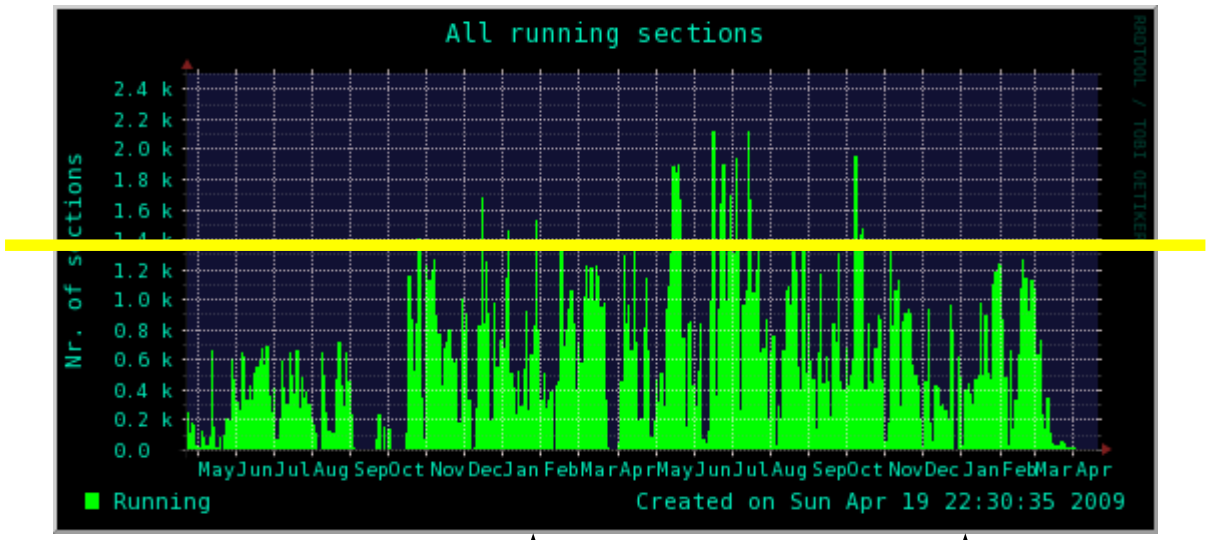


2K slots

2006 2009

Second Production GlideCAF

- NamCAF (North American CAF)
- True GRID implementation
- both on and off site worker nodes



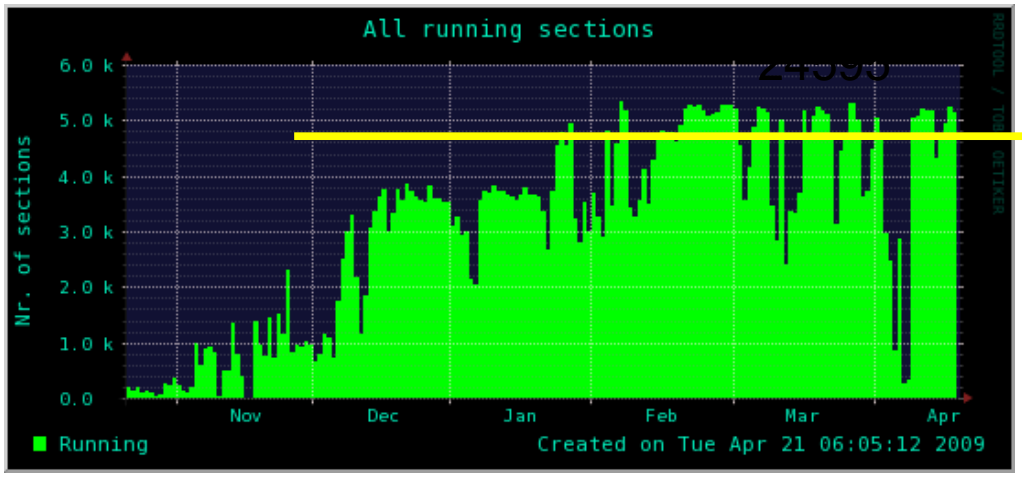
1.5k slots

Jan 2008

Jan 2009

Our Latest GlideCAF

- 8 Processors, 40G Memory
- Driving 5.5k slots
- Collector/Negotiator back on head node

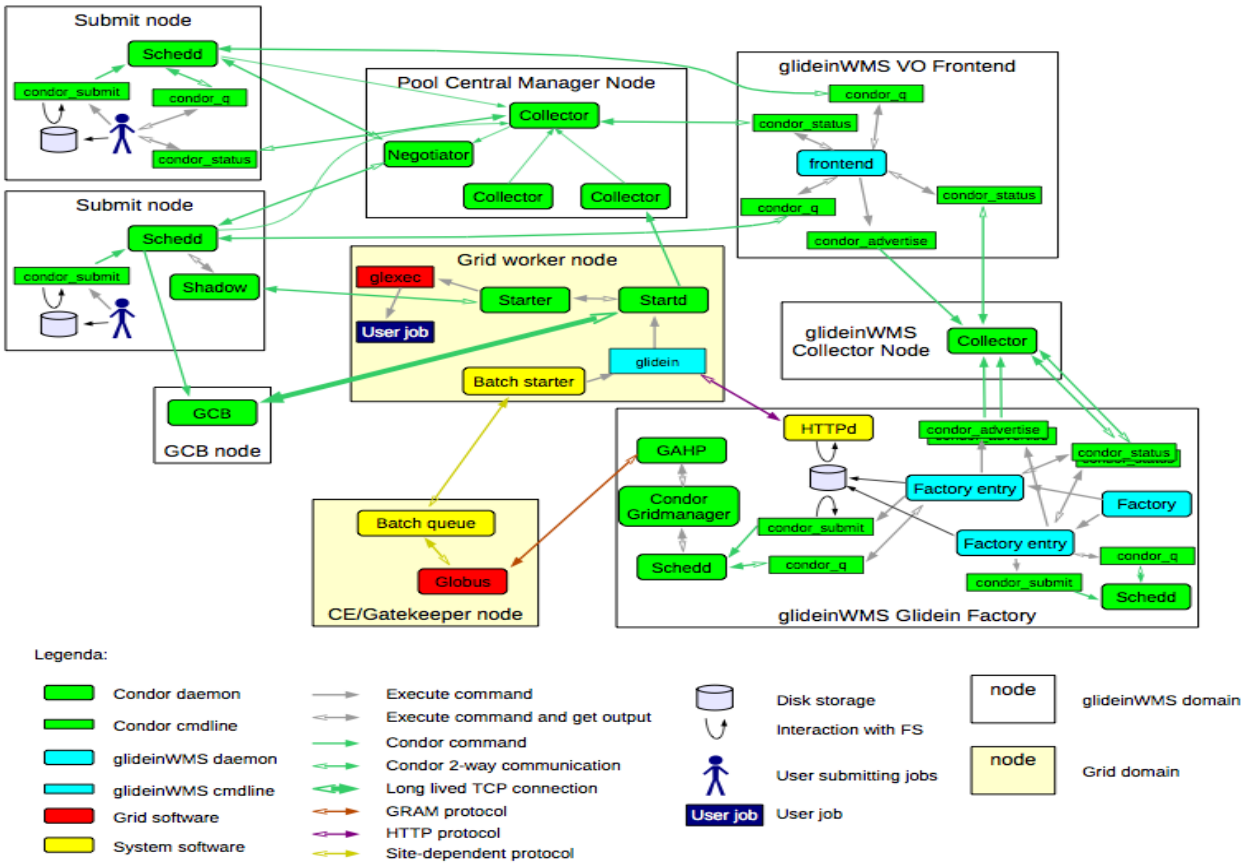


5K slots

Last 6 months

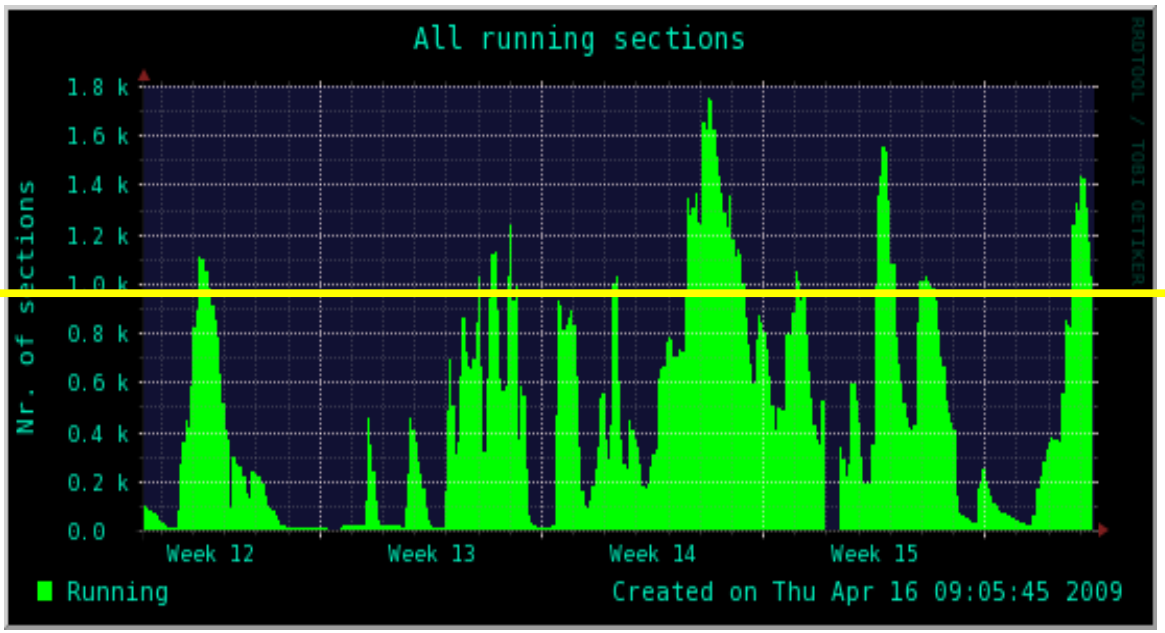
GlideinWMS

- Next Generation Glidein System
- More Condor daemons, more nodes increase throughput/scalability
- Better monitoring of glideins
- See Igor Sfilgois talk tomorrow



Production GlideinWMS!

- NAMCaf replacement
- ~1 month operational experience
- Better monitoring, easier to maintain despite increased complexity



1k slots

Last month running jobs

General Observations

- Configurations are always changing
- Machines are always breaking/being replaced
- We could still use better monitoring
  - ~40 trouble tickets / week
  - My jobs keep restarting - why?
  - My jobs won't start - why?
  - My jobs finish but don't come back - why?
  - Some other experiment is claiming our slots when we know that we have pending jobs & priority - why?
- Often log files are the only way to answer these - there are a lot of them!
  - Pegasus uses netlogger, maybe we should too

General Observations

- Bugs
  - 7.2.0, 7.2.1 collector needs to be restarted once a week or so (memory leaks?) when running 5k slots
  - condor_rm results in held slots – some timing issue
- We need better reliability
  - Too many critical parts 99% reliable
    - Chained together in ways a lot less than 99% reliable
  - Need better fail/over – retry behavior
  - It would be nice if the failovers also let us know what needs fixing
    - example: can startd HOOK_JOB_EXIT help us understand our restarts?

Conclusions

- Condor has been essential to CDFs computing efforts
- Glideins have allowed us to transition to the GRID with with very little impact to our physicists and their work
- Thanks to the Condor team for such a useful and well supported product!

- Acknowledgements: Current Fearless Leaders: Rick Snider, Rick St. Denis Current CAF team: Federica Moscato, Marian Zvada, Joe Boyd, moi, Previous Leader: Donatella Lucchesi Special thanks to previous developers Igor Sfilgoi and team members Doug Benjamin, Krzysztof Genser, also FermiGrid operations contact Steve Timm