Look What I Can Do: Unorthodox Uses of HTCondor in the Open Science Grid

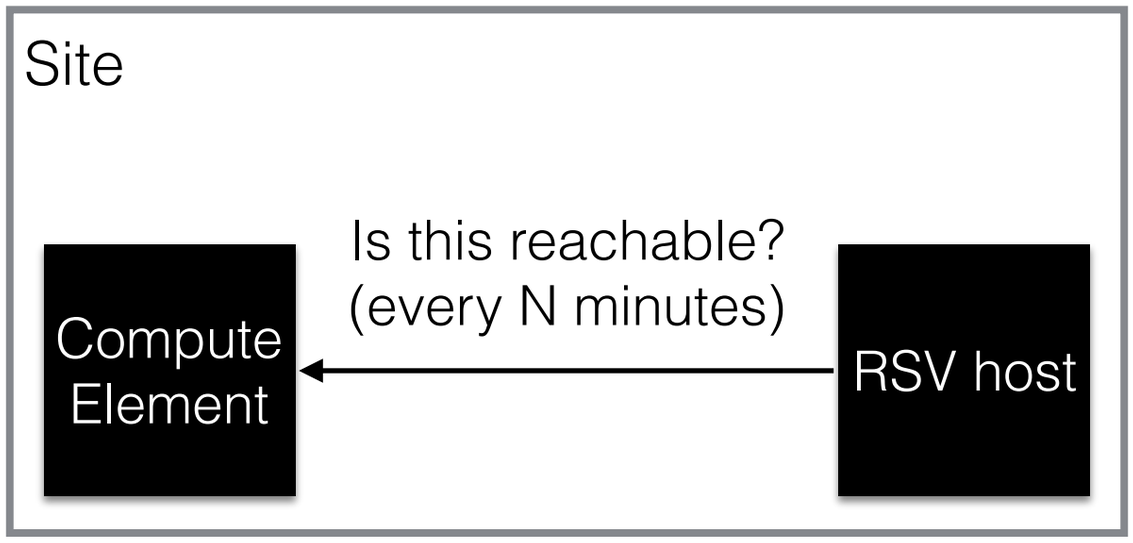
Mátyás Selmeci
Open Science Grid Software Team / Center for High-Throughput Computing
HTCondor Week 2015

More Than a Batch System

- HTCondor has many features besides queuing up jobs to run on execute nodes
- We use these features in several ways to create a wider grid infrastructure
  - Periodic execution of jobs
  - Job router
  - Classads data format and the collector daemon
  - Master daemon
  - Virtual Machine universe

Using Periodic Execution for Monitoring with RSV

Resource and Service Validation (RSV)



- RSV consists of scripts that test part of a system
- These scripts need to run at regular intervals

- Cron is the standard Unix way of running scripts at regular intervals, but has some drawbacks:
  - Jobs may be missed (if the machine is not running when the job is scheduled)
  - No process management: duplicates may run concurrently (if a job is not finished by the time the next run is scheduled) and jobs may pile up

- Instead, submit scripts via HTCondor
- Special attributes in the submit file to cause periodic execution
  - OnExitRemove keeps the job in the queue
  - Cron* attribute values look and act exactly like cron
    - CronWindow lets jobs be late – prevents missed jobs
- HTCondor won't run a second job until the first one has finished – prevents duplicate jobs

```
OnExitRemove = false

# 7,27,47 * * * *
CronMinute = 7,27,47
CronHour = *
CronDayOfMonth = *
CronMonth = *
CronDayOfWeek = *

CronWindow = 99999999

Executable = ping-host
Arguments = ce.example.com
Queue
```

- No need to run all of HTCondor just for cron features
- "condor-cron" package – an HTCondor installation with a separate init script, config files, and wrapper scripts
  - Starts up minimal set of daemons
  - Only listens to localhost
- Does not conflict with running HTCondor (if there is one)

```
root# condor_cron_q
-- Submitter: fermicloud025.fnal.gov : <127.0.0.1:49388> : fermicloud025.fnal.gov
 ID      OWNER            SUBMITTED     RUN_TIME ST PRI SIZE CMD
   1.0   rsv             5/12 10:58   0+00:00:00 I  0   0.0  rsv-control -v 3 -
   2.0   rsv             5/12 10:58   0+00:02:39 R  0   0.0  html-consumer
   3.0   rsv             5/12 10:58   0+00:02:33 R  0   0.0  gratia-consumer

3 jobs; 0 completed, 0 removed, 1 idle, 2 running, 0 held, 0 suspended
```

- For more info on periodic execution of jobs, see section 2.12 of the HTCondor manual: https://research.cs.wisc.edu/htcondor/manual/v8.2/2_12Time_Scheduling.html
- For more info on RSV, see the overview page in the OSG documentation wiki: https://www.opensciencegrid.org/bin/view/Documentation/Release3/RsvOverview

Using the Job Router to Create a Job Gateway with HTCondor CE

- A job gateway, e.g. the Globus GRAM Gatekeeper is the "front door" to jobs wanting to run on a site; it provides:
  - Authentication/authorization
  - Remote submission and monitoring
  - Translation and job routing
- The job gateway is the primary service running on a Compute Element (CE)
- HTCondor CE is HTCondor acting as a complete job gateway

- Like Condor-Cron, it's a separate instance of HTCondor, running alongside whatever batch system is installed on the machine
- Special config with collector, schedd, and job router

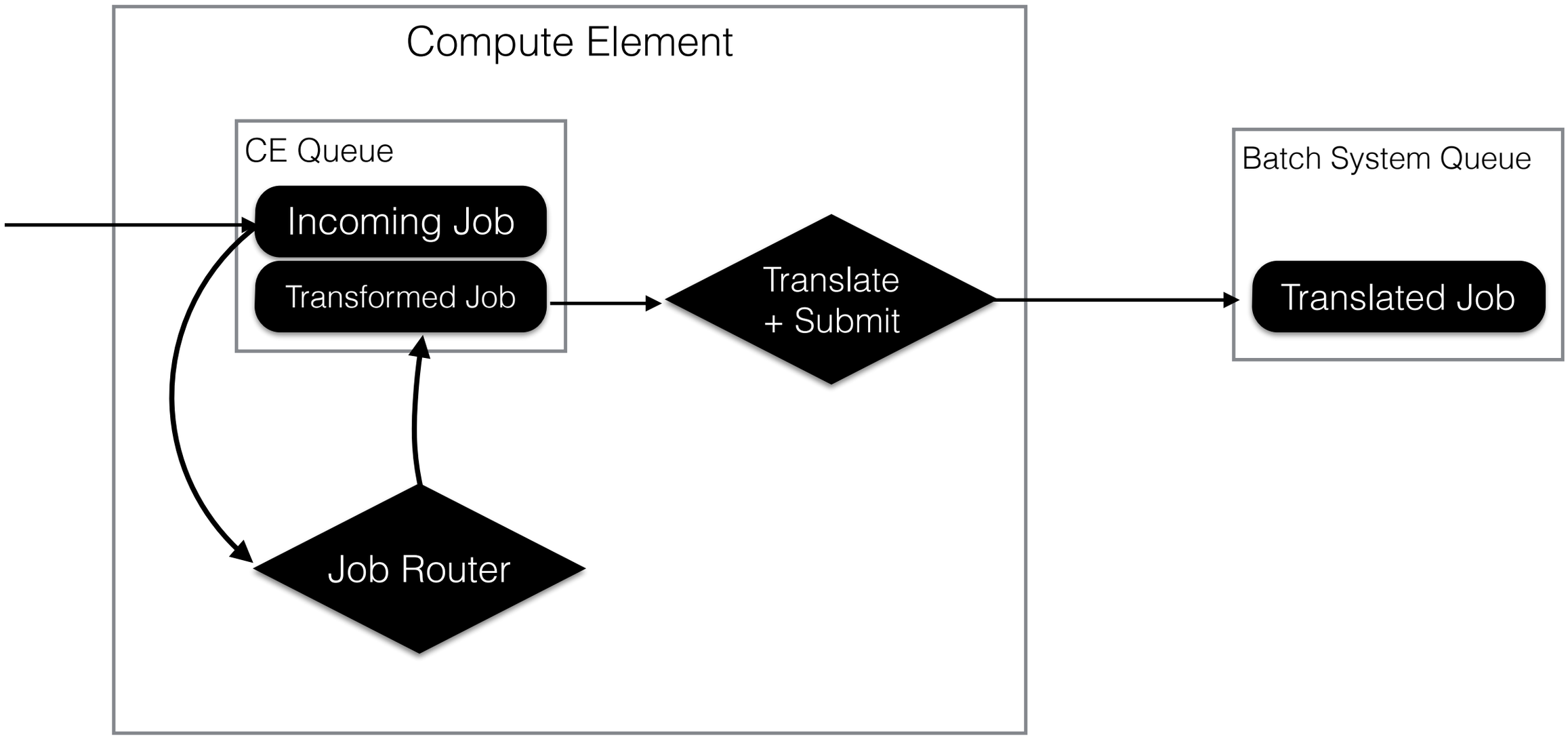
```
condor    15335 condor_master -pidfile /var/run/condor-ce/condor_master.pid
root      15339  \_ condor_procd -A /var/lock/condor-ce/procd_pipe -L /var/
condor    15340  \_ condor_shared_port -f -p 9620
condor    15342  \_ condor_collector -f -port 9619
condor    15347  \_ condor_schedd -f
condor    15348  \_ condor_job_router -f
```

- The job router filters and transforms jobs according to "job routes" defined in the config
- Job classads matched against a classad expression (Requirements)
- Classad attributes added, edited, or removed
- Special attributes also control "next hop", e.g.:
  - TargetUniverse
  - GridResource

Job Route

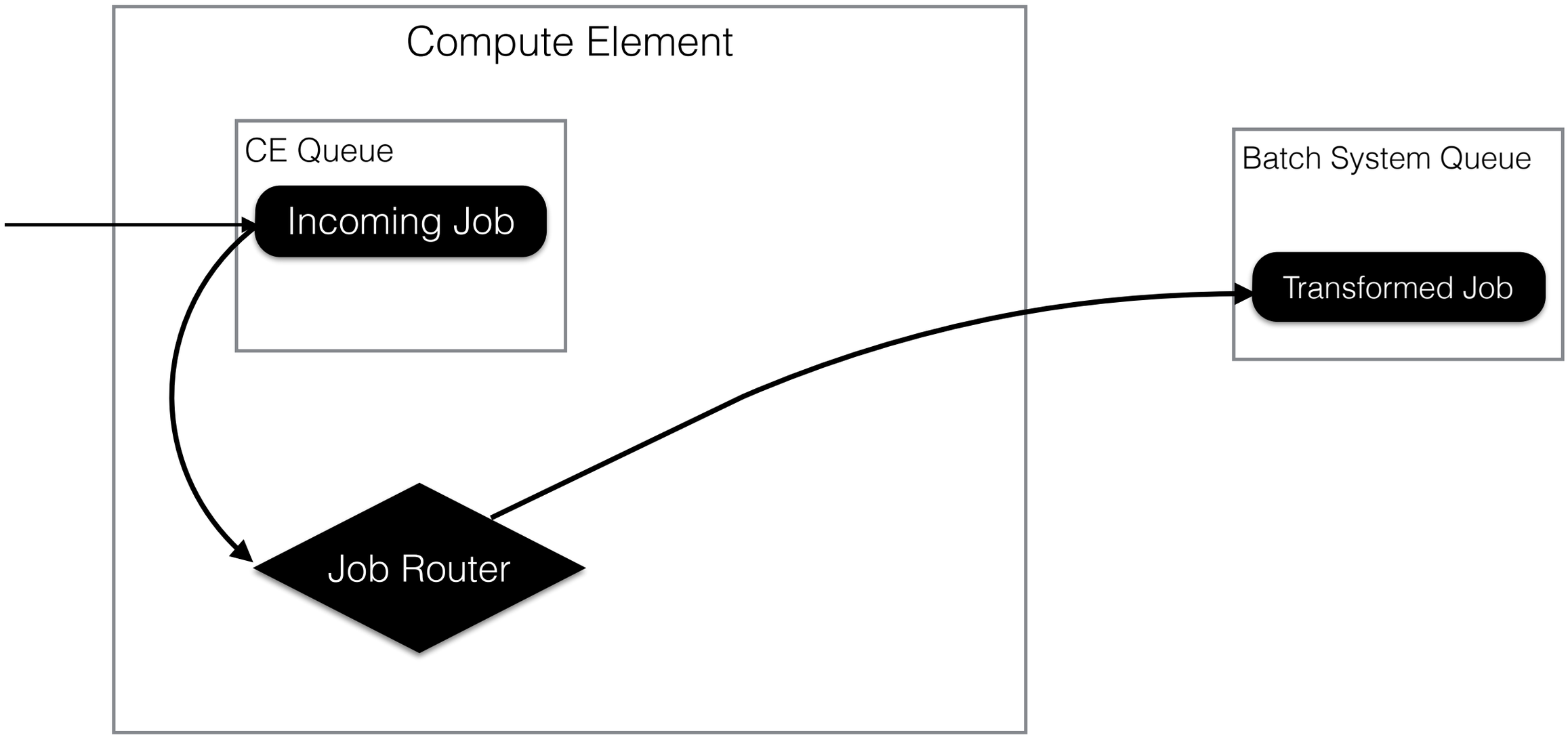
```
[ Requirements = (TARGET.Owner =?= "matyas"); \
  set_Queue = "blue"; \
  delete_Environment = True; \
  ... \
]
```

What Happens on a CE (non-HTCondor batch system)



TargetUniverse = 9 (grid)
GridResource determines next hop

What Happens on a CE (HTCondor batch system)



TargetUniverse = 5 (vanilla)
JOB_ROUTER_SCHEDD2_* config options determine next hop

- For more info on the job router, see section 5.4 of the HTCondor manual: http://research.cs.wisc.edu/htcondor/manual/v8.2/5_4HTCondor_Job.html
- For more info on HTCondor CE, see the overview in the OSG documentation wiki: https://www.opensciencegrid.org/bin/view/Documentation/Release3/HTCondorCEOverview
- For examples of job routes, see the OSG documentation wiki: https://www.opensciencegrid.org/bin/view/Documentation/Release3/JobRouterRecipes

Using Classads and the Collector for Data Exchange

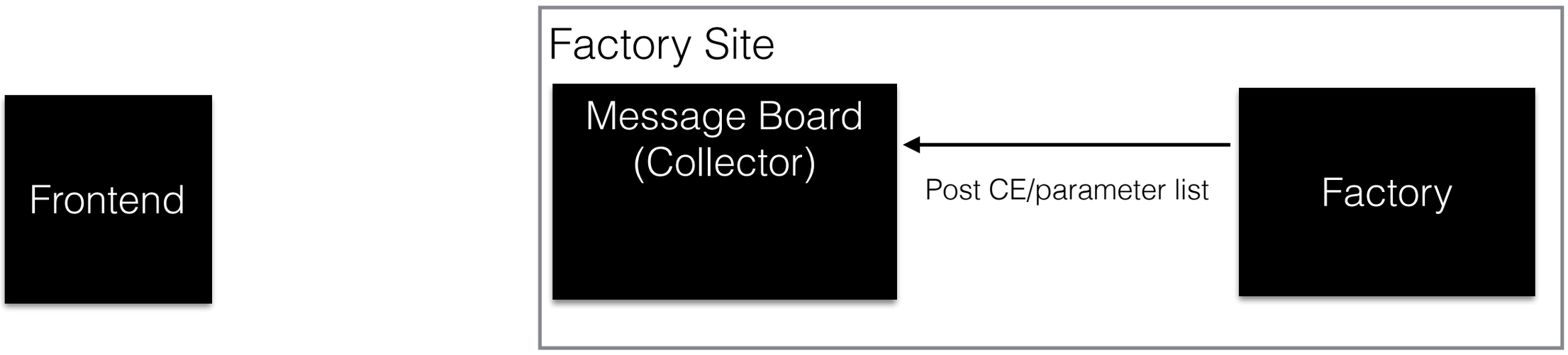
- Classads feature:
  - Arbitrary key/value pairs and ordered lists; may be nested
  - At least as expressive as JSON
  - Also includes expressions; a value can reference other parts of the same classad, or even other classads

- The HTCondor collector:
  - Stores classads
  - Allows retrieval of classads
  - Can forward classads
- condor_status -long retrieves classads
- HTCondor daemons send classads related to their functionality
- condor_advertise sends arbitrary classads

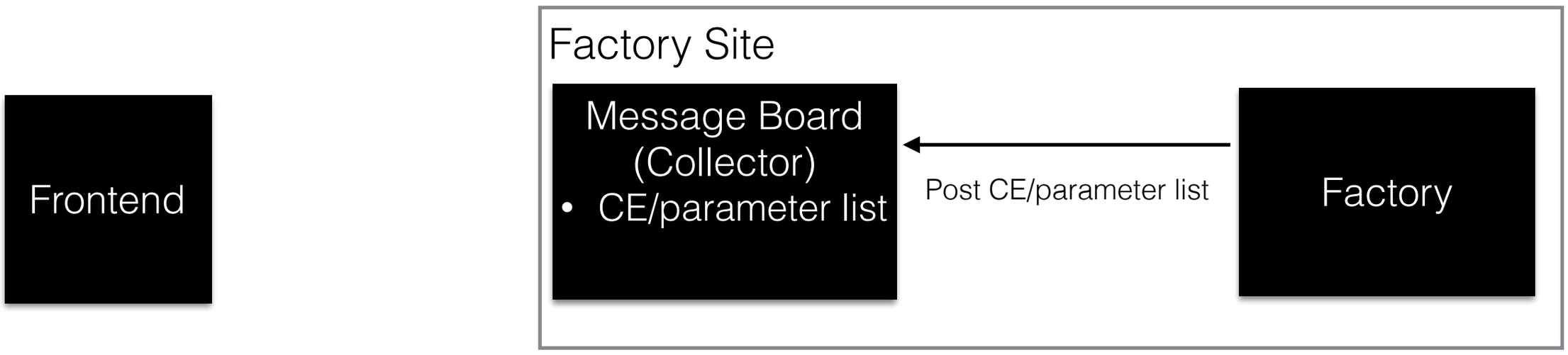
Example 1: GlideinWMS

Example 1: GlideinWMS

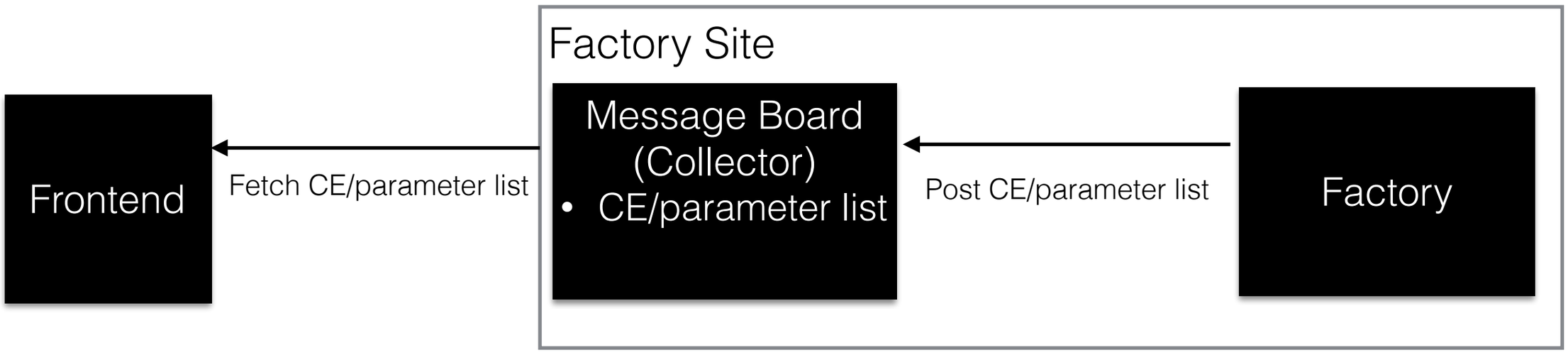
Example 1: GlideinWMS



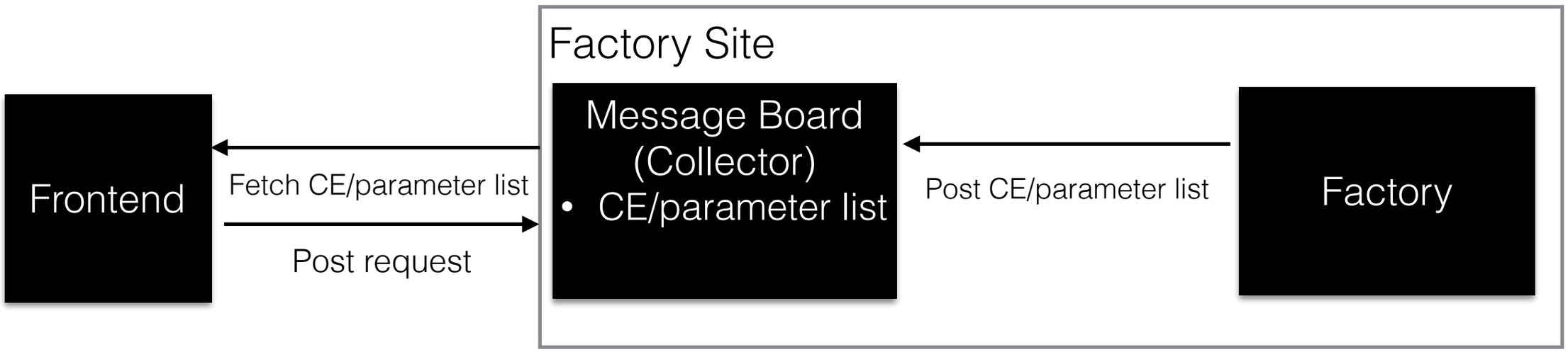
Example 1: GlideinWMS



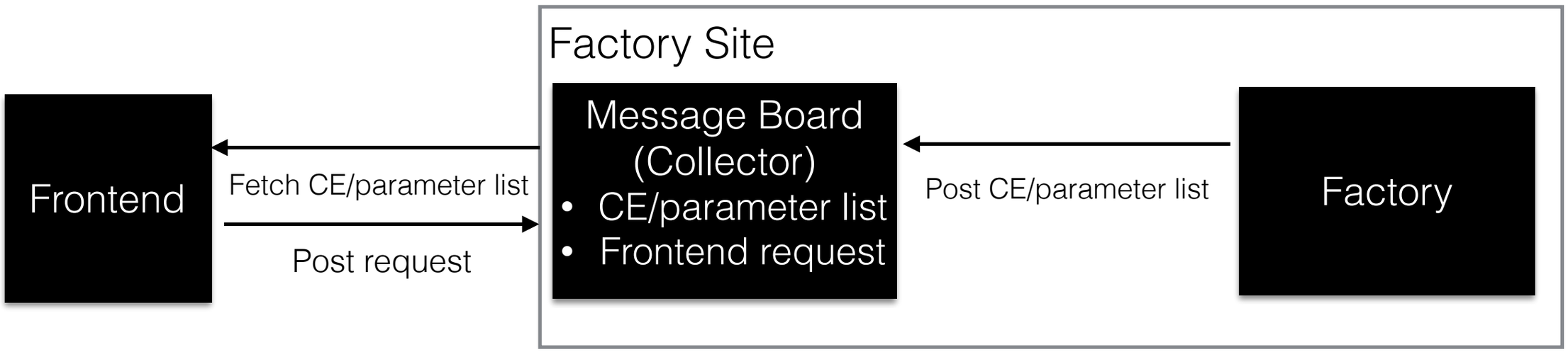
Example 1: GlideinWMS



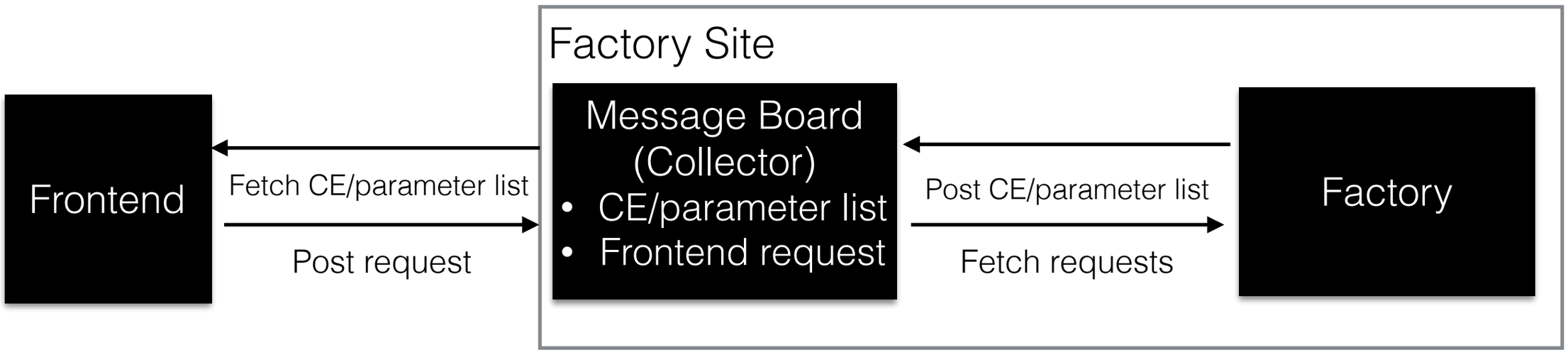
Example 1: GlideinWMS



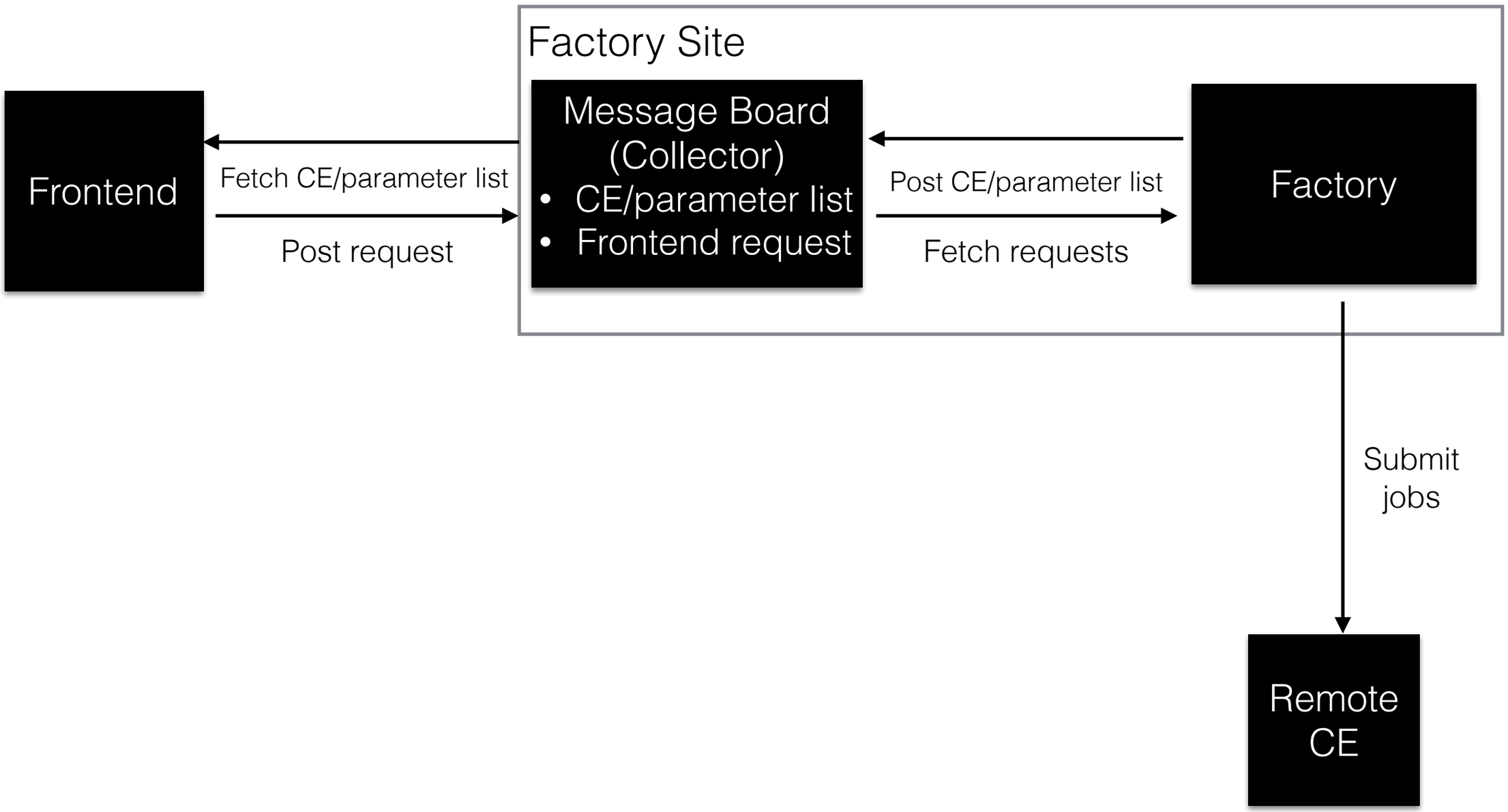
Example 1: GlideinWMS



Example 1: GlideinWMS



Example 1: GlideinWMS



Example 2: CE Collector

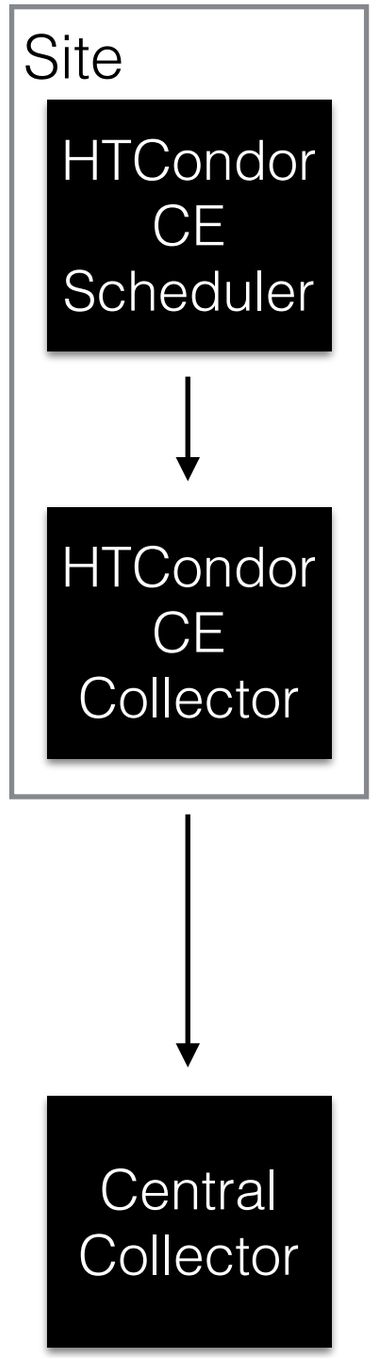
- Work-in-progress system, made in conjunction with HTCondor CE
- (Will be) used to populate Glidein factories' CE/parameter lists
- Uses collector as a database

- Catalog with entries describing sets of machines like this:
  - "GLOW g9"
  - Accessible from cmsgrid01.hep.wisc.edu
  - 2 cores each
  - 4GB memory each

Catalog added to classad of scheduler daemon

Site HTCondor daemons send their classads to the site collector

CONDOR_VIEW options cause collector to forward its classads to a central collector



- Data from multiple HTCondor CE collectors is aggregated and stored for later lookup
- Central collectors queried by a script using the HTCondor Python bindings
- Classad expressions used to narrow down the data we want to see, like the WHERE clause in a SQL query

```
% ./condor_ce_info_status --constrain '(Name == "GLOW g9") && (regexp("cmsgrid01", grid_resource))' --long

[
    OSG_BatchSystems = "Condor";
    Name = "GLOW g9";
    CPUs = 2;
    Memory = 4096;
    OSG_Resource = "GLOW";
    Transform =
        [
            set_MaxMemory = RequestMemory;
            set_xcount = RequestCPUs
        ];
    grid_resource = "condor cmsgrid01.hep.wisc.edu cmsgrid01.hep.wisc.edu:9619";
    Requirements = TARGET.RequestCPUs <= CPUs && TARGET.RequestMemory <= Memory;
    OSG_ResourceGroup = "GLOW"
]
```

- For more info on GlideinWMS, see the documentation at:
  http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.prd/index.html
- For more info on the CE Collector, see the following slides from the OSG All Hands Meeting:
  https://indico.fnal.gov/contributionDisplay.py?contribId=19&sessionId=8&confId=8580

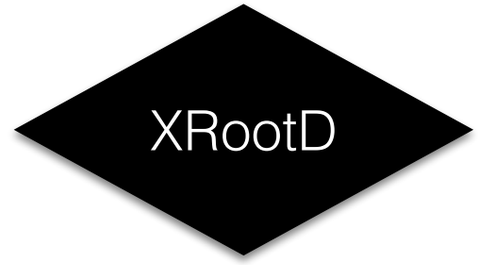
Using the Master for Custom Daemon Management in StashCache

- StashCache is a new infrastructure that will be deployed across OSG to provide efficient staging for large input files
- XRootD daemons – not HTCondor-related at all – provide storage and transfer
- Cache servers distributed across several locations all over the OSG but should be controlled by one organization
- Want to remotely control XRootD daemons on cache servers

- condor_master manages HTCondor daemons
  - Start, stop, reconfigure daemons – locally or remotely
  - Restart dead daemons
  - Restart upgraded daemons
  - Notify admins
- How to reuse some of this for XRootD?

- Add the XRootD daemon to the HTCondor config file
  - XROOTD = /usr/sbin/xrootd
  - DAEMON_LIST = MASTER, COLLECTOR, XROOTD
- Restart the master
- Why won't this work?

XRootD does not implement Daemon Protocol



XRootD does not implement Daemon Protocol

XRootD does not implement Daemon Protocol

XRootD does not implement Daemon Protocol

XRootD does not implement Daemon Protocol

Solution: add a shim script between Master and XRootD

Solution: add a shim script between Master and XRootD

Solution: add a shim script between Master and XRootD

Solution: add a shim script between Master and XRootD

Solution: add a shim script between Master and XRootD

- Add the shim script instead of the XRootD daemon to the HTCondor config file
  - XROOTD = /usr/sbin/stashcache
  - DAEMON_LIST = MASTER, COLLECTOR, XROOTD
- Restart the master

- For more info on StashCache, see the following slides from the OSG All Hands Meeting:
  https://indico.fnal.gov/contributionDisplay.py?contribId=47&sessionId=5&confId=8580
- For more info on XRootD, see the documentation at:
  http://www.xrootd.org/docs.html

Virtual Machine Universe

- HTCondor can also launch virtual machines as jobs
- Perfect for automated tests of installs
- See Tim Cartwright's talk

Conclusion

- Thank you to the following people for their help with this presentation:
  - Brian Bockelman
  - Brian Lin
  - Tim Cartwright
- Send questions to osg-software@opensciencegrid.org

Backup slides

(Simplified) HTCondor Daemon Protocol

Master Behavior	Daemon Behavior
Send SIGHUP	Reload configuration
Send SIGTERM	Shut down after completing work
Send SIGQUIT	Shut down quickly
Kill & restart daemon if hung	Send heartbeat packet as proof of life

Shim script's implementation of HTCondor Daemon Protocol

Master Behavior	Shim Script Behavior
Send SIGHUP	Run "service xrootd restart"
Send SIGTERM	Run "service xrootd stop" then exit
Send SIGQUIT	Same as SIGTERM
Kill & restart daemon if hung	Send heartbeat packet on behalf of xrootd