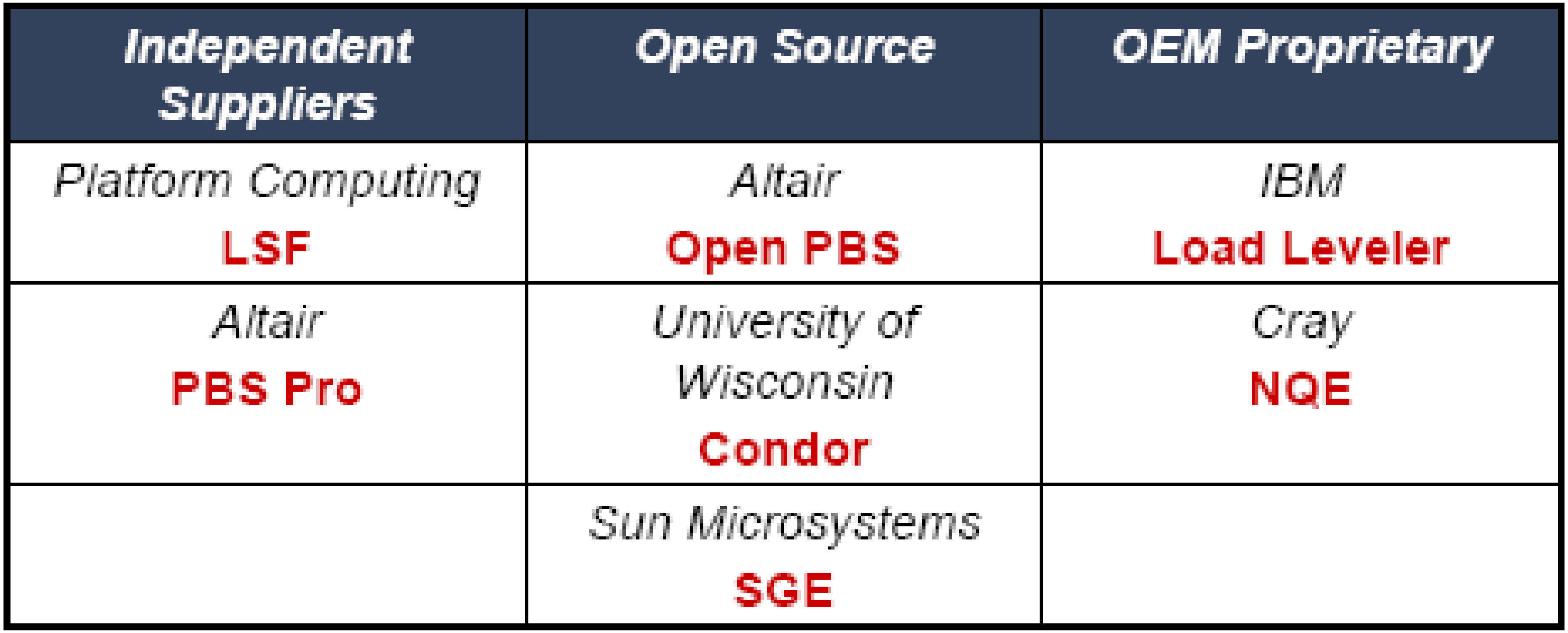
Resource Management Systems

RMS



Independent Suppliers	*Open Source*	*OEM Proprietary*
Platform Computing **LSF**	*Altair* **Open PBS**	*IBM* **Load Leveler**
Altair **PBS Pro**	*University of Wisconsin* **Condor**	*Cray* **NQE**
	Sun Microsystems **SGE**	

NQE
(Network Queue Environment)

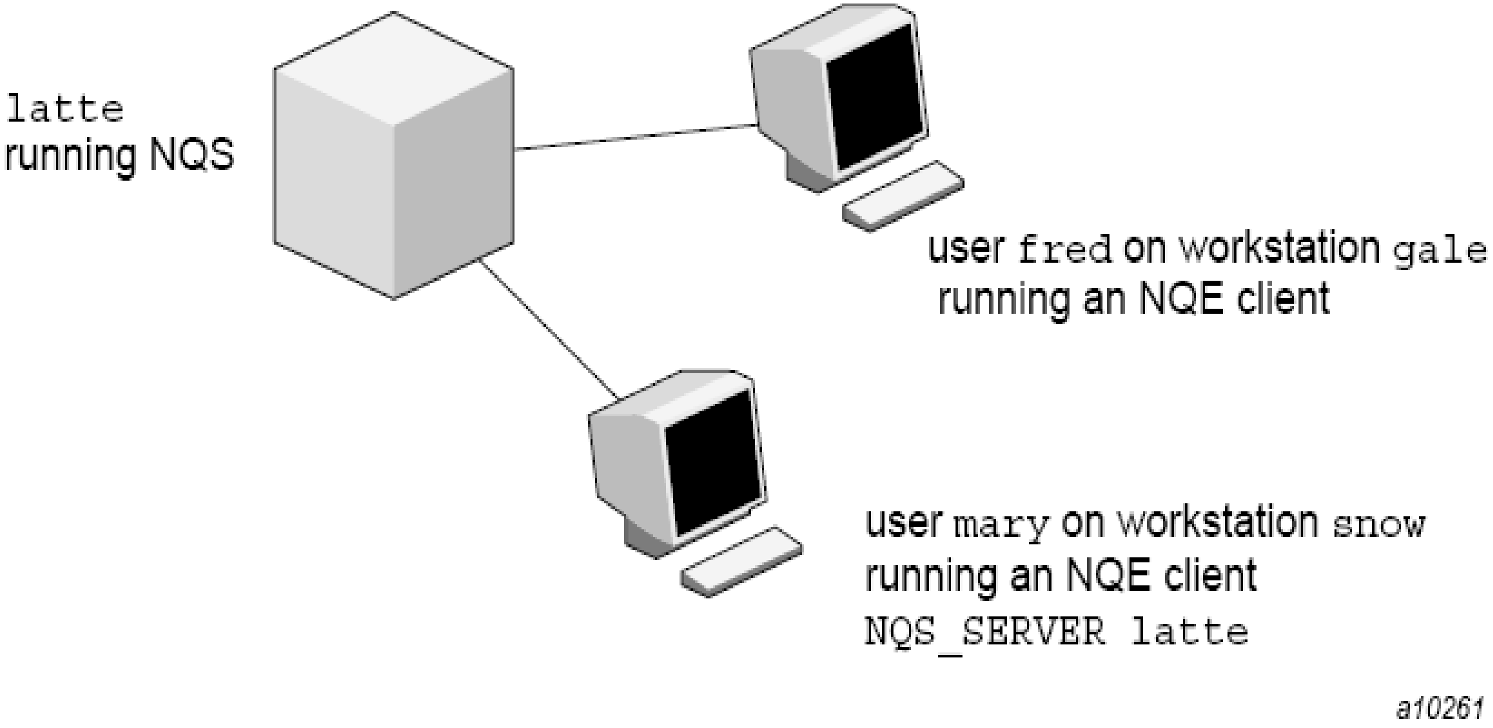


Figure 1. Sample NQE Configuration

NQE

```
#QSUB –eo                                                          # merge stdout and stderr
#QSUB –J m                                                         # append NQS job log to stdout
#QSUB –o "%fred@gale/nppa_latte:/home/gale/fred/mary.jjob.output"  # returns stdout to fred@gale
#QSUB –me                                                          # sends mail to submitter at
                                                                   completion
#QSUB                                                              # optional delimiter
date                                                               # prints date
rft –user mary –host snow –domain nppa_latte –nopassword –function get jan.data   # use FTA to transfer jan.data
nqs.data                                                           from latte to the NQS server
cc loop.c –o prog.out                                              # compile loop.c
./prog.out                                                         # execute
rm –f loop.c prog.out jan.data nqs.data                            # delete files
echo job complete
```

FTA: File Transfer Agent   NQS: Networking Queueing System

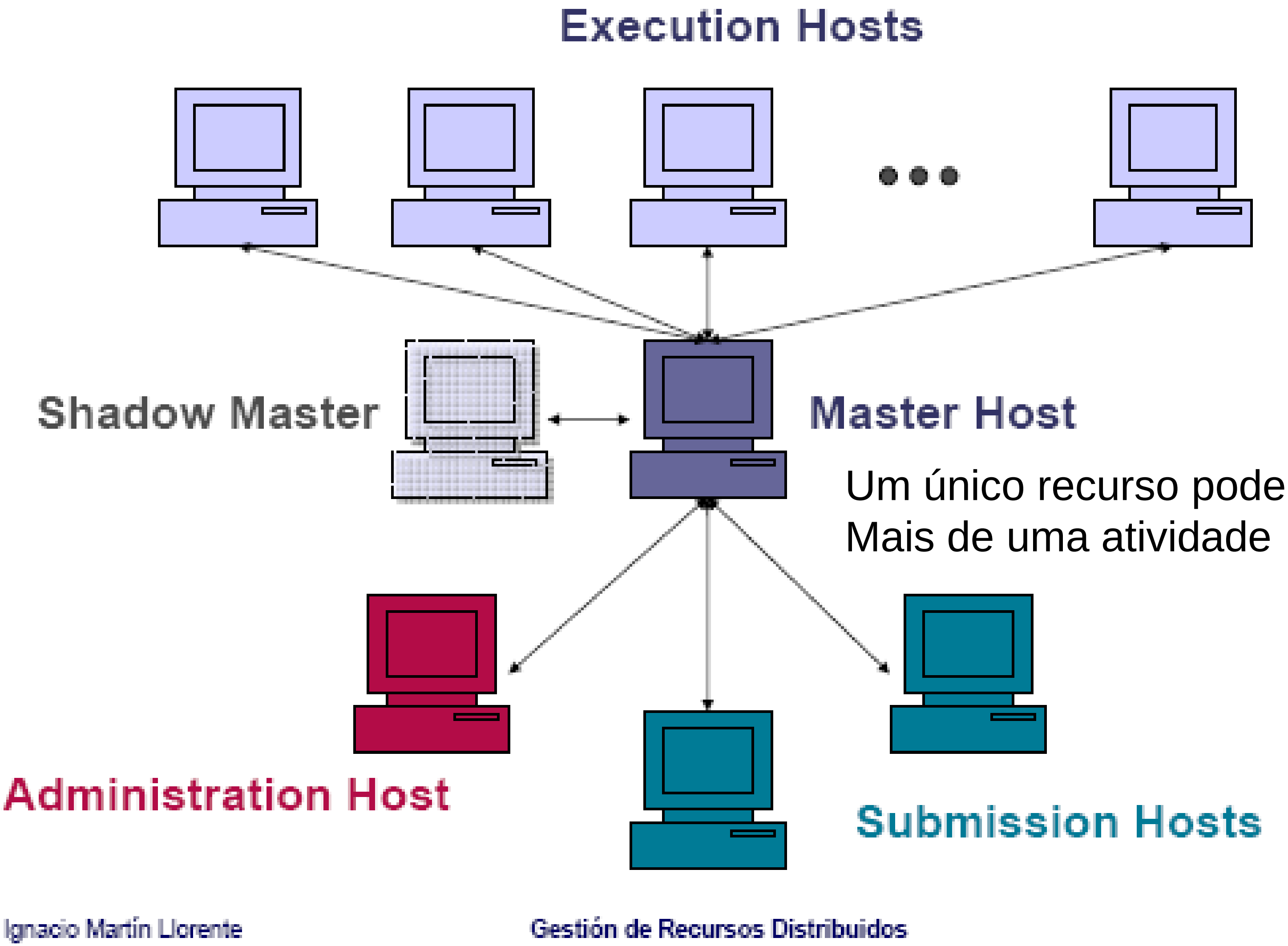
NQE user commands

cevent Posts, reads, and deletes job-dependency event information.
cqdel Deletes or signals to a specified batch request.
cqstatl Provides a line-mode display of requests and queues on a specified host
cqsub Submits a batch request to NQE.
ftua Transfers a file interactively (this command is issued on an NQE server only).
ilb Executes a load-balanced interactive command.
nqe Provides a graphical user interface (GUI) to NQE functionality.

Commands issued on an NQE server only:
qalter Alters the attributes of one or more NQS requests
qchkpnt Checkpoints an NQS request on a UNICOS, UNICOS/mk, or IRIX system
qdel Deletes or signals NQS requests
qlimit Displays NQS batch limits for the local host
qmsg Writes messages to stderr, stdout, or the job log file of an NQS batch request
qping Determines whether the local NQS daemon is running and responding to requests
qstat Displays the status of NQS queues, requests, and queue complexes
qsub Submits a batch request to NQS
rft Transfers a file in a batch request

Fonte: http://techpubs.sgi.com/library/tpl/cgi-bin/getdoc.cgi?coll=0650&db=bks&fname=/SGI_Admin/NQE_AG/apa.html

SGE
(Sun Grid Engine)



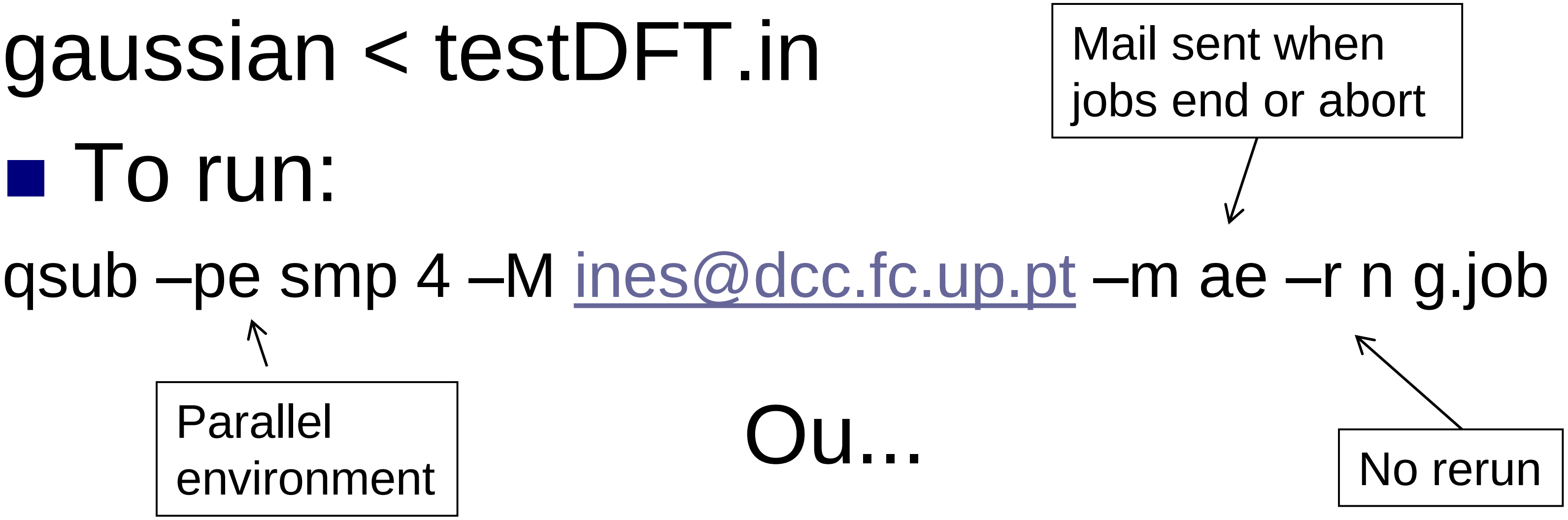
Um único recurso pode desempenhar
Mais de uma atividade

SGE

- Commands similar to NQE
- Example: g.job

```
#!/bin/csh
gaussian < testDFT.in
```

- To run:



Mail sent when jobs end or abort

```
qsub –pe smp 4 –M ines@dcc.fc.up.pt –m ae –r n g.job
```

Parallel environment

Ou...

No rerun

SGE

- File g.job

```
#!/bin/csh
#$ -pe smp 4    # parallel environment
#$ -M ines@dcc.c.up.pt
#$ -m ae        # mail sent at end/abort
#$ -r n         # no rerun
gaussian < testDFT.in
```

- To run: qsub g.job

SGE: other example

```
#$ -pe openmpi* 32
#$ -q short*
#$ -l dedicated=4
```

SGE: another example

#$ -V	# Inherit the submission environment
#$ -cwd	# Start job in submission directory
#$ -N myMPI	# Job Name
#$ -j y	# Combine stderr and stdout
#$ -o $JOB_NAME.o$JOB_ID	# Name of the output file (eg. myMPI.oJobID)
#$ -pe 12way 24	# Requests 12 tasks/node, 24 cores total
#$ -q normal	# Queue name normal
#$ -l h_rt=01:30:00	# Run time (hh:mm:ss) - 1.5 hours
#$ -M	# Use email notification address
#$ -m be	# Email at Begin and End of job

SGE

- User can specify requirements (cpu type, disk space, memory etc)
- SGE registers a task, requirements and control information (user, group, dept, date/time of submission etc)
- SGE produces plans to execute tasks
- As soon as a resource becomes available, SGE launches the execution of one of the waiting tasks
  - The task with greater priority or greater waiting time, according to the task planner configuration
  - If there are several queues available, choose the one that has the least loaded host
  - There can be several queues per cluster

SGE

- Planning Policies:
  - Ticket-based (User)
    - + tickets → + priority
    - Tickets are assigned statically according to the queue policy and priorities assiged to each user
  - Urgency-based (tasks)
    - time limit for the task (can be given by the user)
    - Queue waiting time
    - Requested resources
  - Personalized: allows arbitrary assignment of priorities to tasks (similar to Unix nice)

SGE

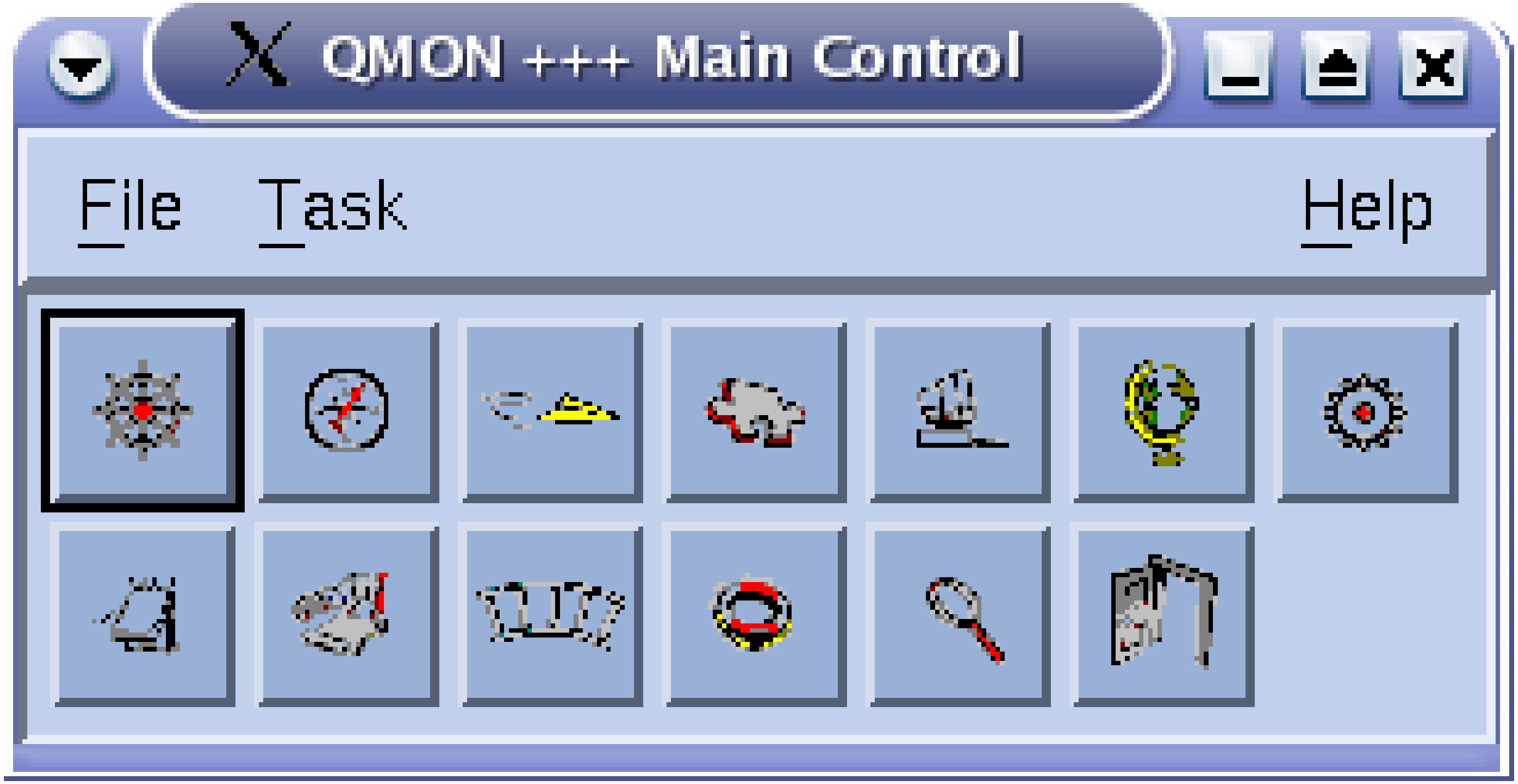
- Lyfe cycle of a task:
  - □ Submission
  - □ Master stores task and informs planner
  - □ Planner inserts task in the suitable queue
  - □ Master sends task to corresponding host
  - □ Before executing, the execution daemon:
    - Changes the directory to the task's dir
    - Initializes the environment (variables)
    - Initializes the set of processors
    - Change the uid to the uid of the task's owner
    - Initialize resource limits to that process
    - Collects accounting information
    - Stores the task to a database and waits for it to finish
    - As soon as the task terminates, sends a message to the master and eiliminates the task entry from the database

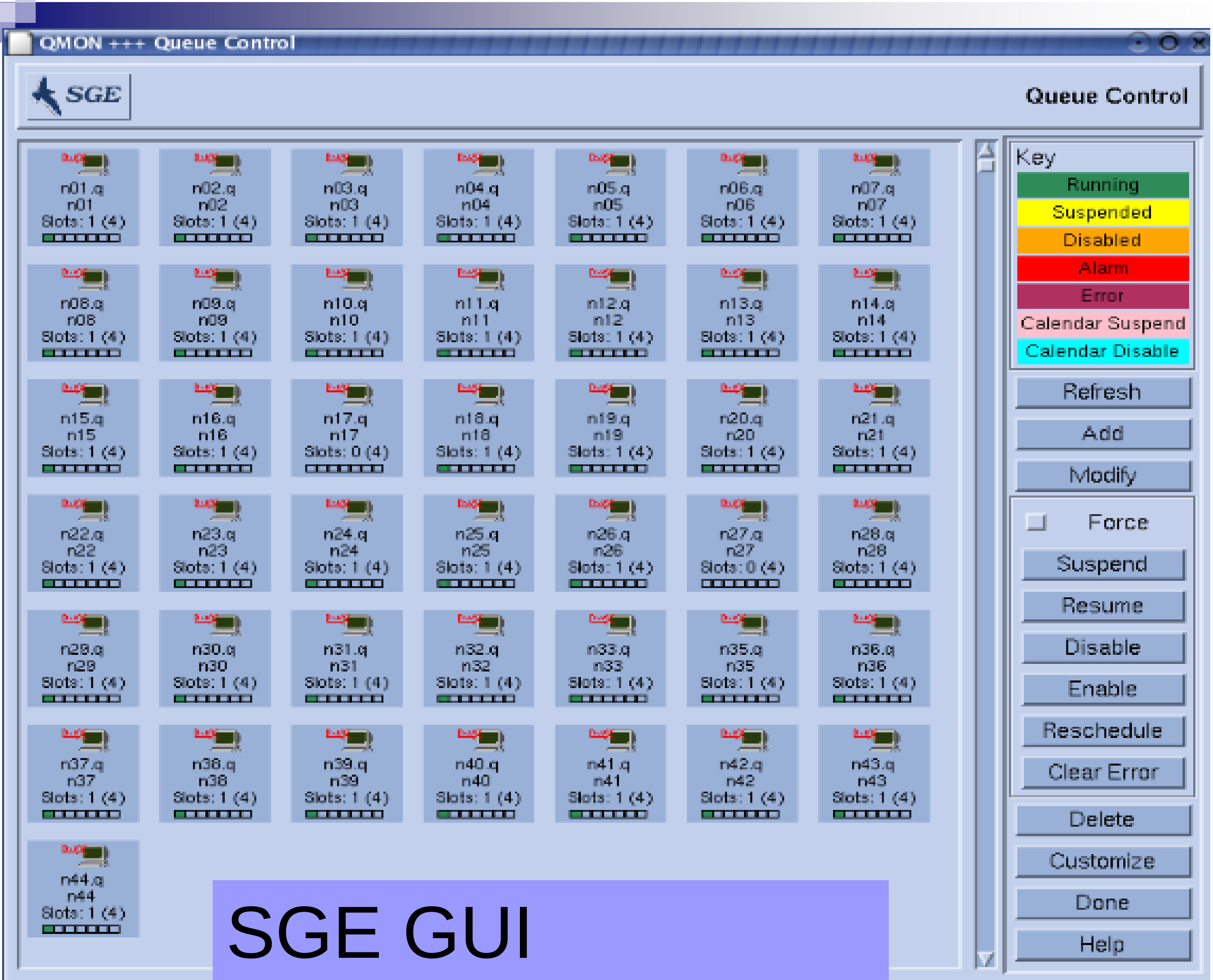
SGE

- Some commands:
  - qconf: cluster config
  - qsub: task submission
  - qdel: eliminate tasks
  - qacct: accounting
  - qhost: inspect hosts status
  - qstat: inspect queue status

SGE

- GUI





n01.q n01 Slots: 1 (4)	n02.q n02 Slots: 1 (4)	n03.q n03 Slots: 1 (4)	n04.q n04 Slots: 1 (4)	n05.q n05 Slots: 1 (4)	n06.q n06 Slots: 1 (4)	n07.q n07 Slots: 1 (4)
n08.q n08 Slots: 1 (4)	n09.q n09 Slots: 1 (4)	n10.q n10 Slots: 1 (4)	n11.q n11 Slots: 1 (4)	n12.q n12 Slots: 1 (4)	n13.q n13 Slots: 1 (4)	n14.q n14 Slots: 1 (4)
n15.q n15 Slots: 1 (4)	n16.q n16 Slots: 1 (4)	n17.q n17 Slots: 0 (4)	n18.q n18 Slots: 1 (4)	n19.q n19 Slots: 1 (4)	n20.q n20 Slots: 1 (4)	n21.q n21 Slots: 1 (4)
n22.q n22 Slots: 1 (4)	n23.q n23 Slots: 1 (4)	n24.q n24 Slots: 1 (4)	n25.q n25 Slots: 1 (4)	n26.q n26 Slots: 1 (4)	n27.q n27 Slots: 0 (4)	n28.q n28 Slots: 1 (4)
n29.q n29 Slots: 1 (4)	n30.q n30 Slots: 1 (4)	n31.q n31 Slots: 1 (4)	n32.q n32 Slots: 1 (4)	n33.q n33 Slots: 1 (4)	n35.q n35 Slots: 1 (4)	n36.q n36 Slots: 1 (4)
n37.q n37 Slots: 1 (4)	n38.q n38 Slots: 1 (4)	n39.q n39 Slots: 1 (4)	n40.q n40 Slots: 1 (4)	n41.q n41 Slots: 1 (4)	n42.q n42 Slots: 1 (4)	n43.q n43 Slots: 1 (4)
n44.q n44 Slots: 1 (4)						

SGE GUI

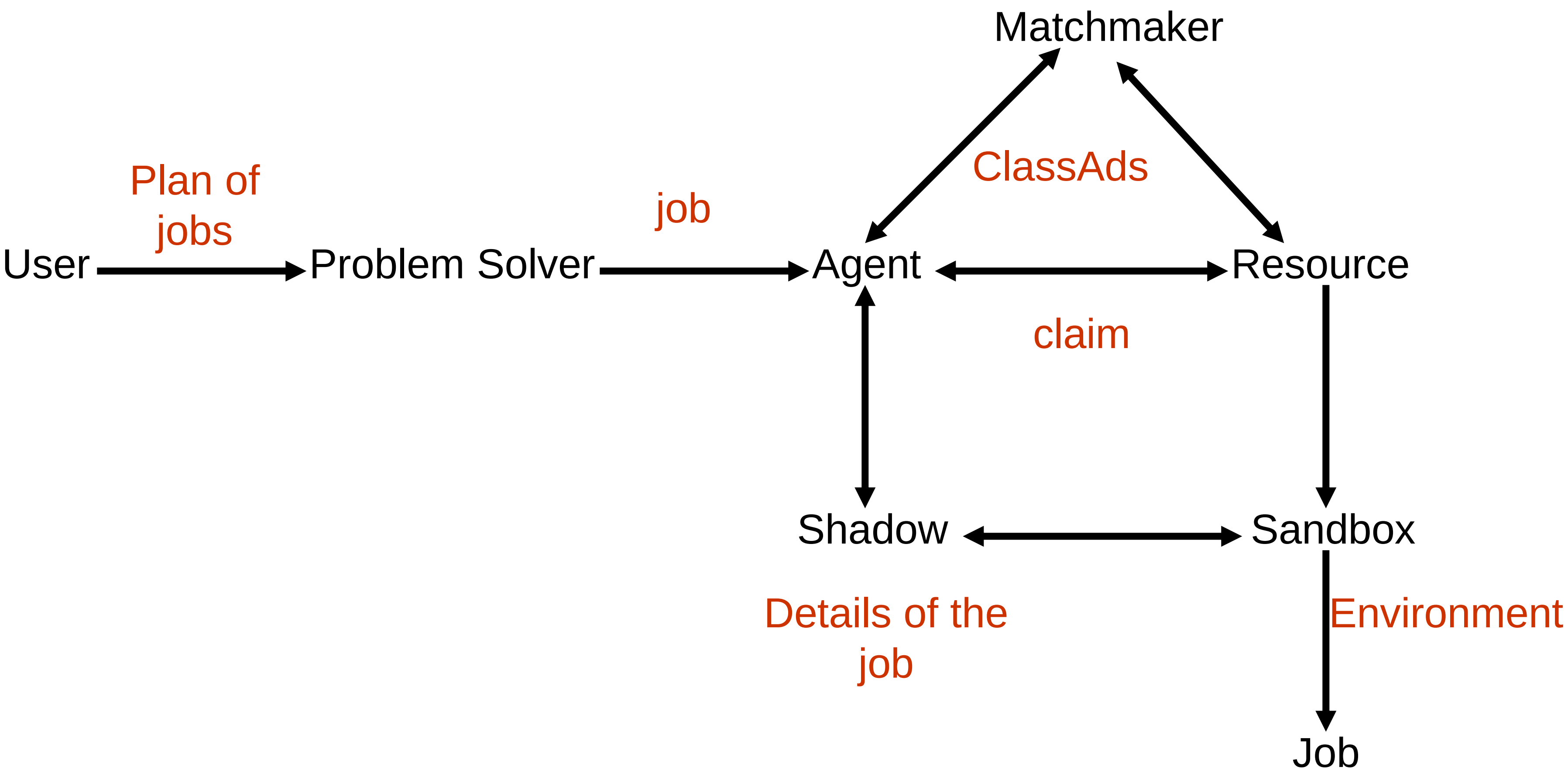
Condor

- It is a specialized job and resource management system. It provides:
  - Job management mechanism
  - Scheduling
  - Priority scheme
  - Resource monitoring
  - Resource management

Condor

- The user submits a job to an agent.
- The agent is responsible for remembering jobs in a persistent storage while finding resources willing to run them.
- Agents and resources advertise themselves to a **matchmaker**, which is responsible for introducing potentially compatible agents and resources.
- At the agent, a shadow is responsible for providing all the details necessary to execute a job.
- At the resource, a **sandbox** is responsible for creating a safe execution environment for the job and protecting the resource from any mischief.

Condor


Matchmaker

ClassAds

Plan of jobs

job

User

Problem Solver

Agent

Resource

claim

Shadow

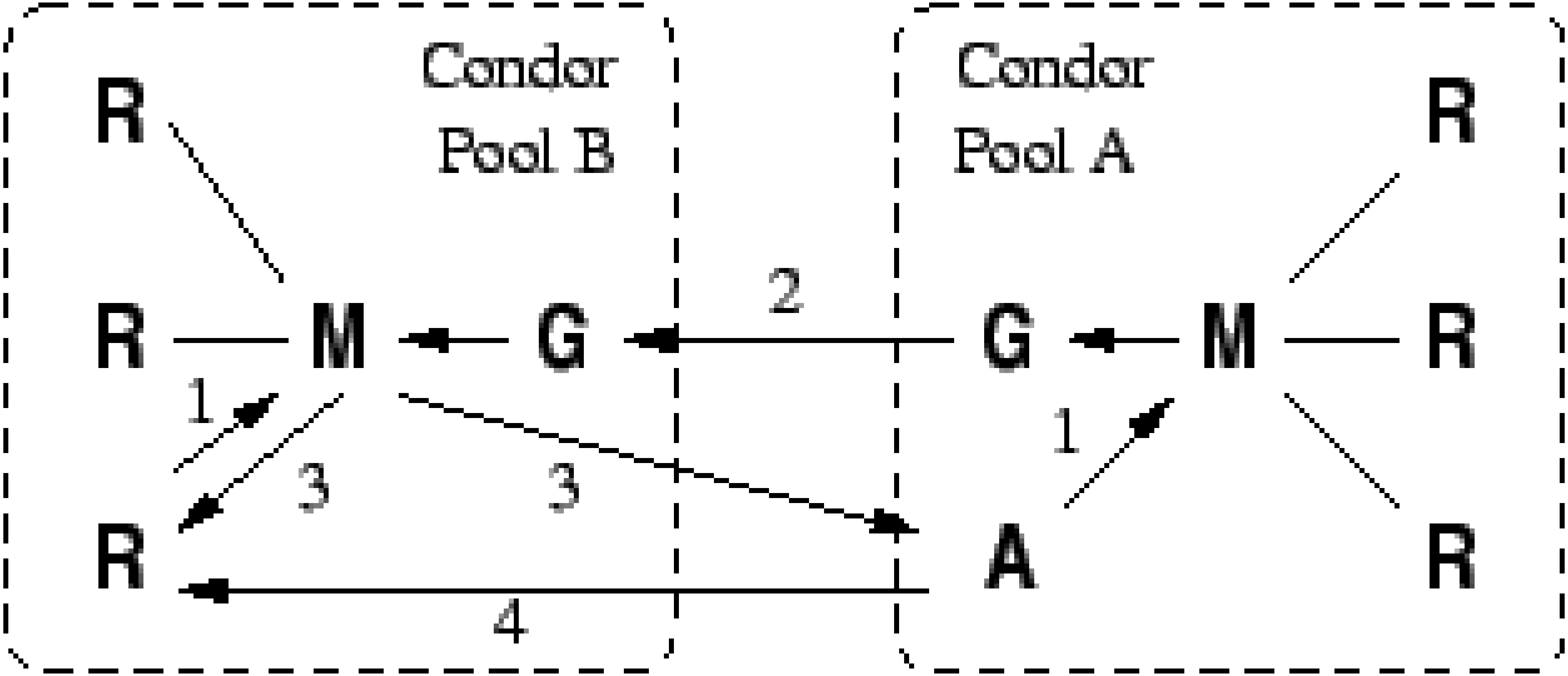
Sandbox

Details of the job

Environment

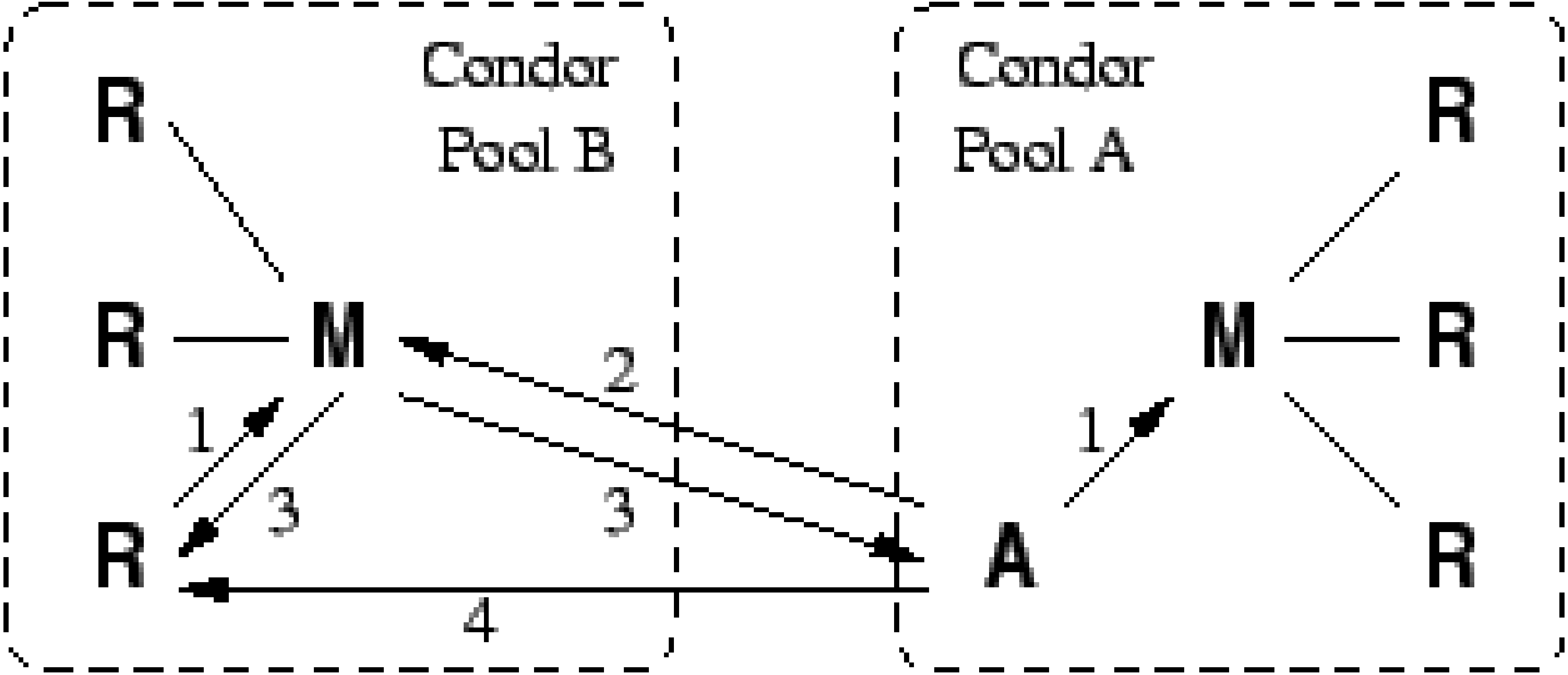
Job

Condor: Gateway Flocking



- Gateway passes information about participants between pools,
- M(A) sends request to M(B) through gateways,
- M(B) returns a match

Condor
Direct Flocking



A also advertises to Condor Pool B

Condor

- **Strongly recommended**:
  - Condor week 2012/2013 materials:
  - http://research.cs.wisc.edu/htcondor/HTCondorWeek201*/
  - Basic Condor Administration (DeSmet)
  - Basic Introduction to using Condor (Miller)
  - Condor and Workflows: an Introduction (Panike)

RMS

- Each has its own interface
- Do not provide integration
- No interoperability
- Require specific administrative capabilities
- Increment operational costs
- Generate over-provisioning and global load imbalance