Grid Compute Resources and Job Management

Job and compute resource management

- This module is about running jobs on remote compute resources

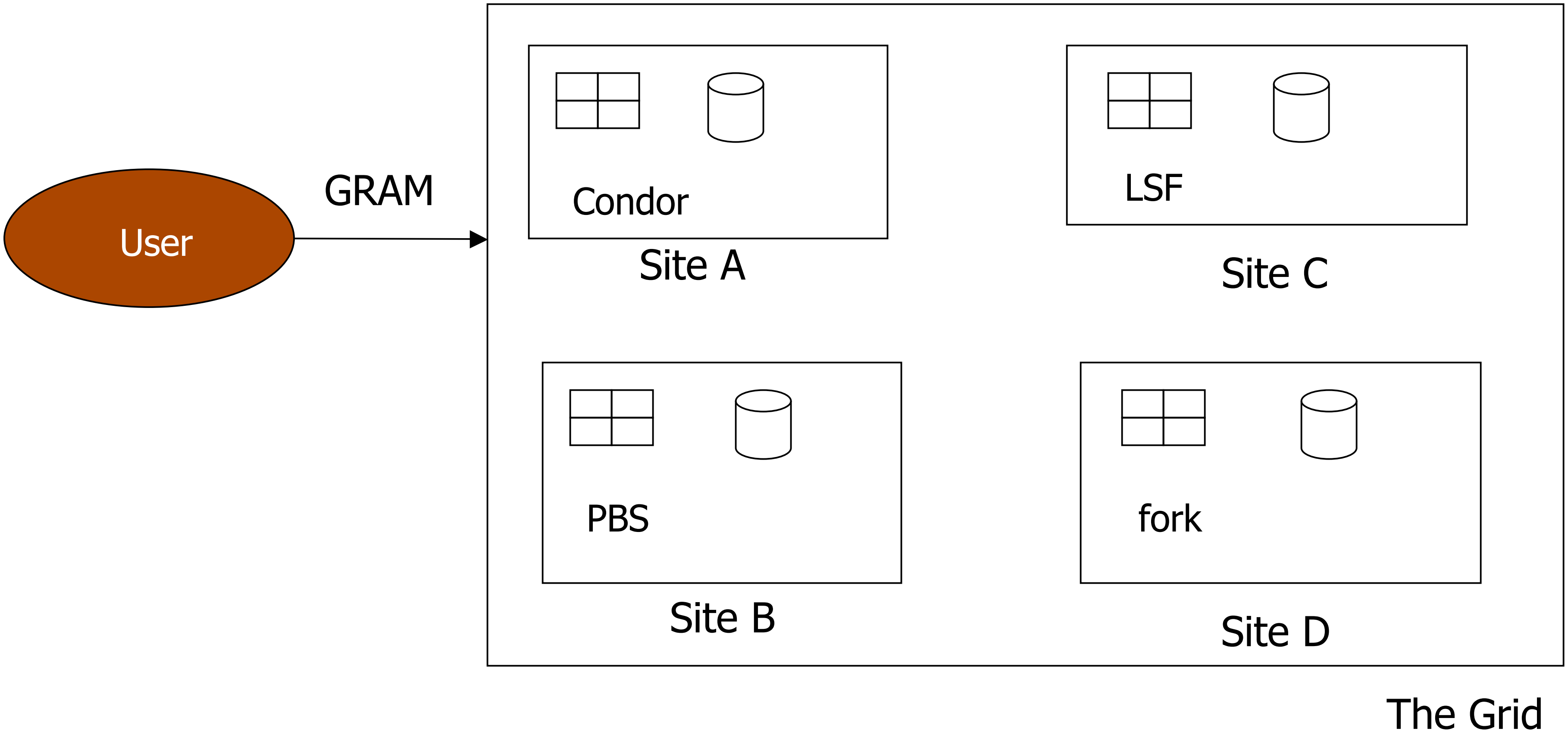
Job and resource management

- Compute resources have a local resource manager
    - This controls who is allowed to run jobs and how they run, on a resource
- GRAM
    - Helps us run a job on a remote resource
- Condor
    - Manages jobs

Local Resource Managers

- Local Resource Managers (LRMs) – software on a compute resource such a multi-node cluster.
- Control which jobs run, when they run and on which processor they run
- Example policies:
  - Each cluster node can run one job. If there are more jobs, then the other jobs must wait in a queue
  - Reservations – maybe some nodes in cluster reserved for a specific person
- eg. PBS, LSF, Condor

Job Management on a Grid



GRAM

- *Globus Resource Allocation Manager*
- Provides a standardised interface to submit jobs to different types of LRM
- Clients submit a job request to GRAM
- GRAM translates into something the LRM can understand
- Same job request can be used for many different kinds of LRM

GRAM

- Given a job specification:
  - Create an environment for a job
  - Stage files to and from the environment
  - Submit a job to a local resource manager
  - Monitor a job
  - Send notifications of the job state change
  - Stream a job’s stdout/err during execution

Two versions of GRAM

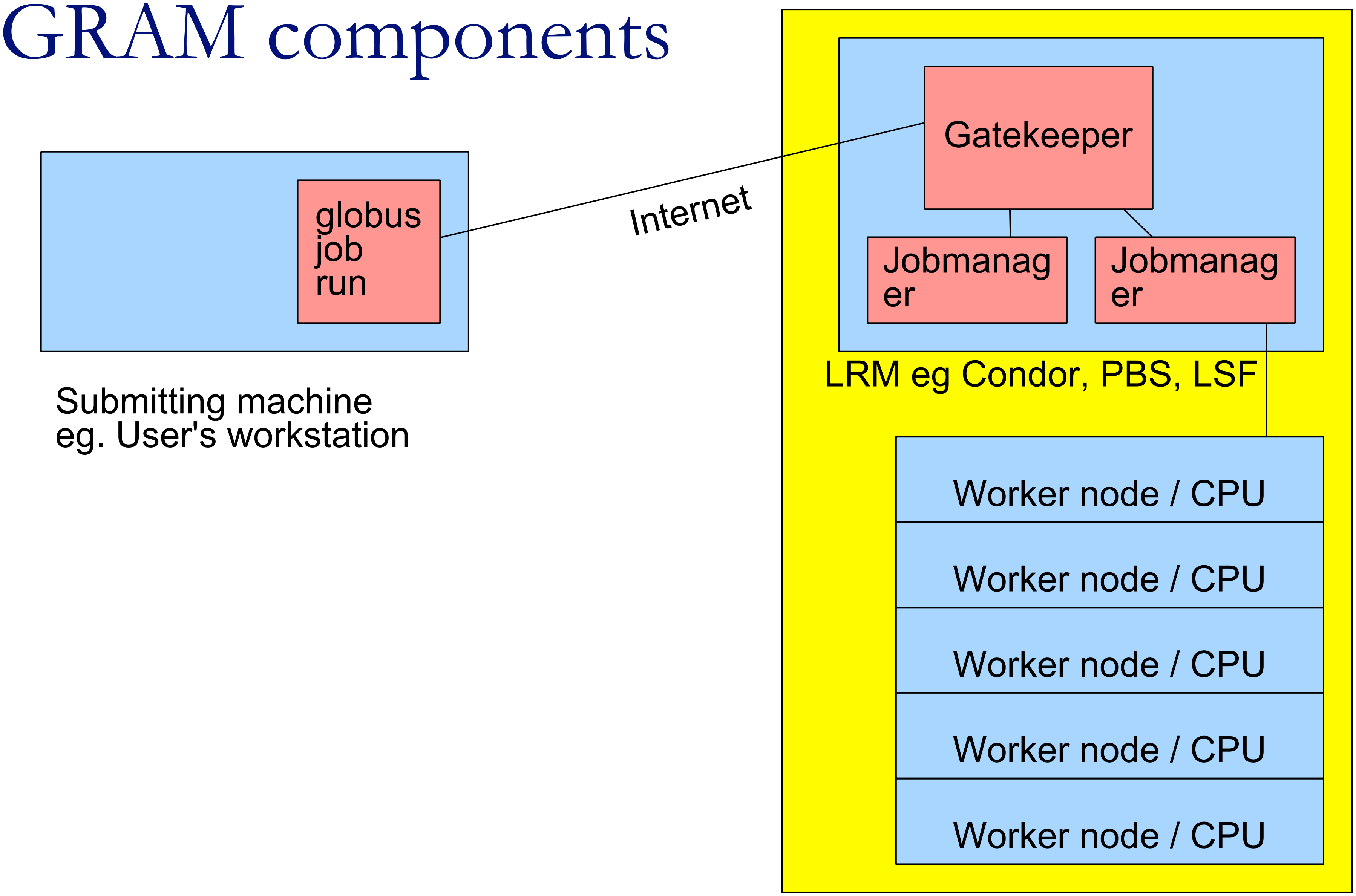
- There are two versions of GRAM
  - GRAM2
    - Own protocols
    - Older
    - More widely used
    - No longer actively developed
  - GRAM4
    - Web services
    - Newer
    - New features go into GRAM4

- In this module, will be using GRAM2

GRAM components

- Clients – eg. Globus-job-submit, globusrun
- Gatekeeper
  - Server
  - Accepts job submissions
  - Handles security
- Jobmanager
  - Knows how to send a job into the local resource manager
  - Different job managers for different LRMs

GRAM components



Submitting a job with GRAM

- Globus-job-run command

-
  ```
  globus-job-run rookery.uchicago.edu /bin/hostname
  rook11
  ```
- Run '/bin/hostname' on the resource rookery.uchicago.edu
- We don't care what LRM is used on 'rookery'. This command works with any LRM.

The client can describe the job with GRAM’s Resource Specification Language (RSL)

- Example:

```
&(executable = a.out)
(directory = /home/nobody )
  (arguments = arg1 "arg 2")
```

- Submit with:

```
 globusrun -f spec.rsl -r
rookery.uchicago.edu
```

Use other programs to generate RSL

- RSL job descriptions can become very complicated
- We can use other programs to generate RSL for us
- Example: Condor-G – next section

Condor

- Globus-job-run submits jobs, but...
  - No job tracking: what happens when something goes wrong?
- Condor:
  - Many features, but in this module:
  - Condor-G for reliable job management

Condor can manage a large number of jobs

- Managing a large number of jobs
  - You specify the jobs in a file and submit them to Condor, which runs them all and keeps you notified on their progress
  - Mechanisms to help you manage huge numbers of jobs (1000's), all the data, etc.
  - Condor can handle inter-job dependencies (DAGMan)
  - Condor users can set job priorities
  - Condor administrators can set user priorities
- Can do this as:
  - a local resource manager on a compute resource
  - a grid client submitting to GRAM (Condor-G)

Condor can manage compute resource

- Dedicated Resources
  - Compute Clusters
- Non-dedicated Resources
  - Desktop workstations in offices and labs
    - Often idle 70% of time
- Condor acts as a Local Resource Manager

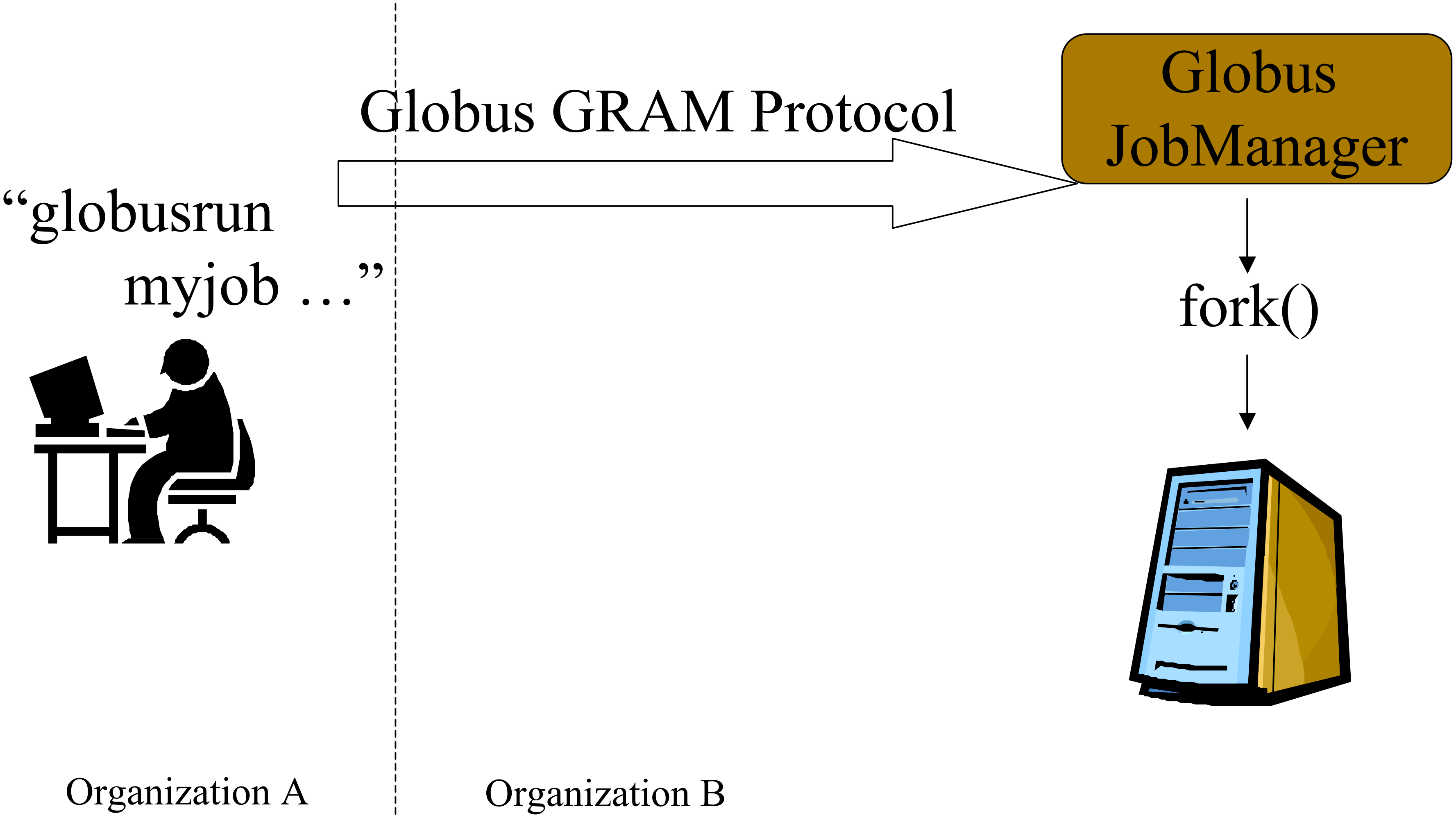
… and Condor Can Manage Grid jobs

- Condor-G is a specialization of Condor. It is also known as the “Grid universe”.
- Condor-G can submit jobs to Globus resources, just like globus-job-run.
- Condor-G benefits from Condor features, like a job queue.

Some Grid Challenges

- Condor-G does whatever it takes to run your jobs, even if …
  - The gatekeeper is temporarily unavailable
  - The job manager crashes
  - Your local machine crashes
  - The network goes down

Remote Resource Access: Globus



“globusrun myjob …”

Globus GRAM Protocol

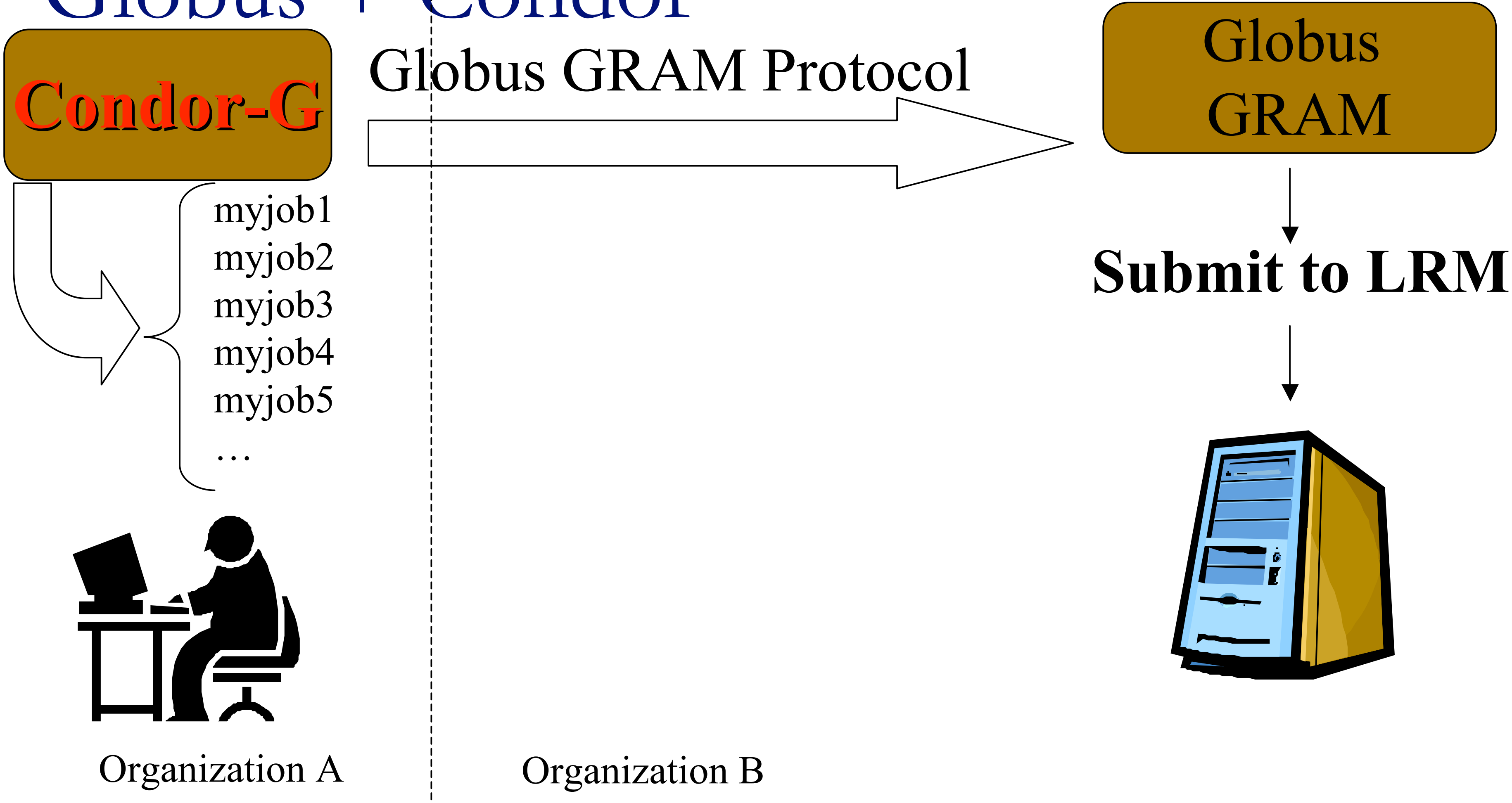
Globus JobManager

fork()

Organization A

Organization B

Remote Resource Access: Condor-G + Globus + Condor



Condor-G

myjob1
myjob2
myjob3
myjob4
myjob5
…

Globus GRAM Protocol

Globus GRAM

Submit to LRM

Organization A

Organization B

Example Application …

Simulate the behavior of **F**(x,y,z) for 20 values of x, 10 values of y and 3 values of z (20*10*3 = 600 combinations)

- **F** takes on the average 3 hours to compute on a "typical" workstation (total = 1800 hours)
- **F** requires a "moderate" (128MB) amount of memory
- **F** performs "moderate" I/O - (x,y,z) is 5 MB and **F**(x,y,z) is 50 MB

- 600 jobs

Creating a Submit Description File

- A plain ASCII text file
- Tells Condor about your job:
  - Which executable, universe, input, output and error files to use, command-line arguments, environment variables, any special requirements or preferences (more on this later)
- Can describe many jobs at once (a "cluster") each with different input, arguments, output, etc.

Simple Submit Description File

```
# Simple condor_submit input file
# (Lines beginning with # are comments)
# NOTE: the words on the left side are not
#       case sensitive, but filenames are!
Universe   = vanilla
Executable = my_job
Queue
```

```
$ condor_submit myjob.sub
```

Other Condor commands

- condor_q – show status of job queue
- condor_status – show status of compute nodes
- condor_rm – remove a job
- condor_hold – hold a job temporarily
- condor_release – release a job from hold

Condor-G: Access non-Condor Grid resources

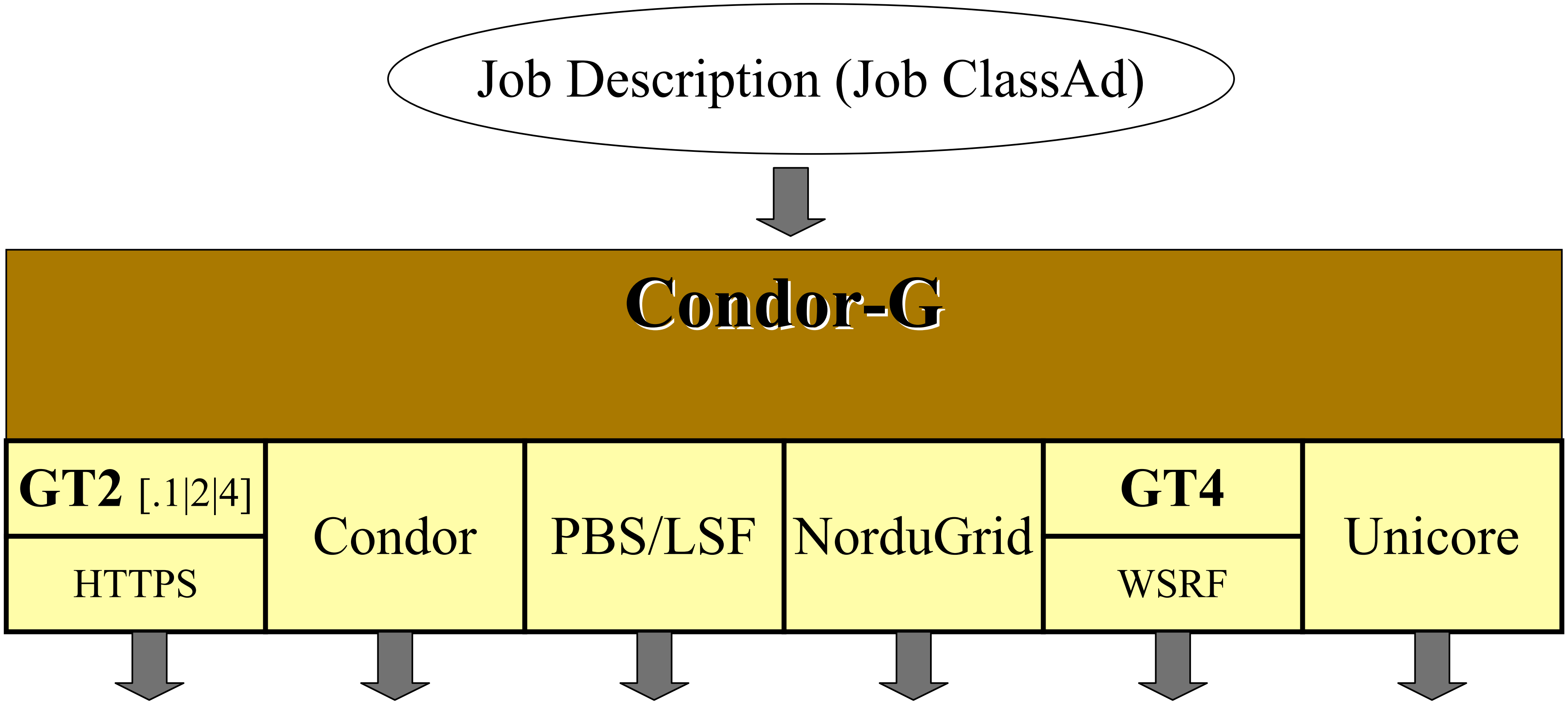
the globus project
www.globus.org

- middleware deployed across entire Grid
- remote access to computational resources
- dependable, robust data transfer

Condor
High Throughput Computing

- job scheduling across multiple resources
- strong fault tolerance with checkpointing and migration
- layered over Globus as "personal batch system" for the Grid

Condor-G



Job Description (Job ClassAd)

Condor-G

GT2 [.1|2|4]

HTTPS

Condor

PBS/LSF

NorduGrid

GT4

WSRF

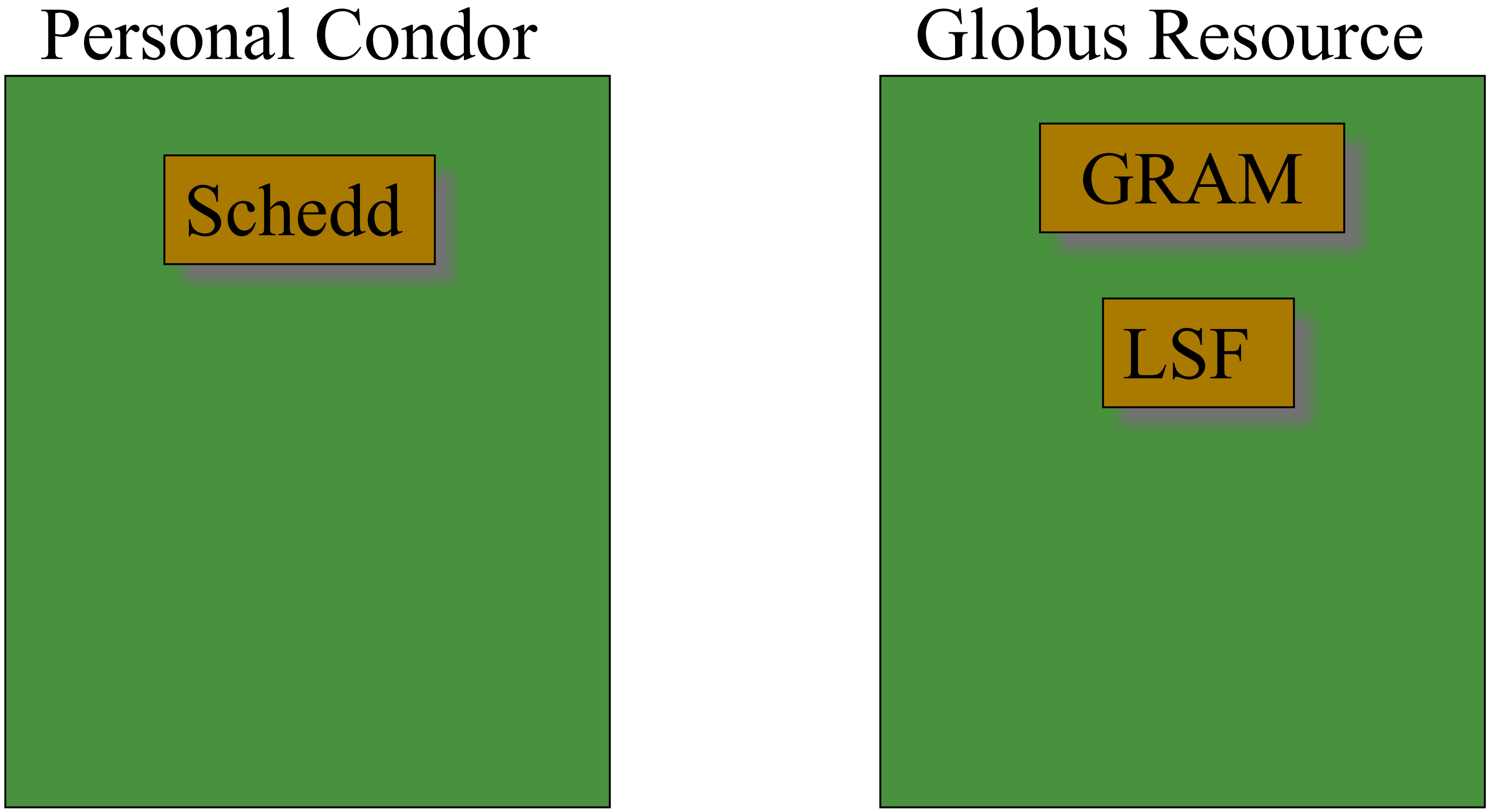
Unicore

Submitting a GRAM Job

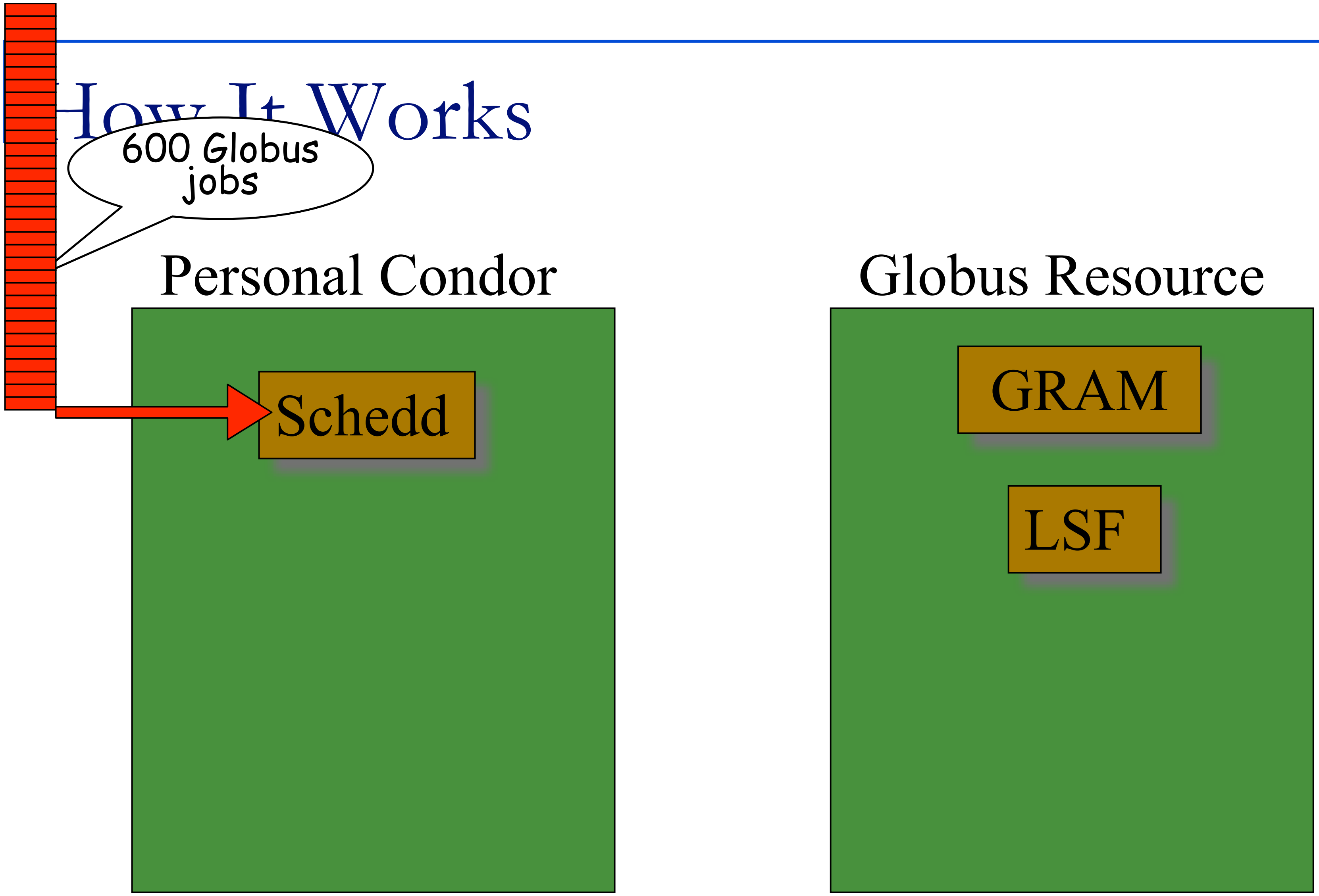
- In submit description file, specify:
  - Universe = grid
  - Grid_Resource = gt2 <gatekeeper host>
    - ‘gt2’ means GRAM2
  - Optional: Location of file containing your X509 proxy

```
universe      = grid
grid_resource = gt2 beak.cs.wisc.edu/jobmanager-pbs
executable    = progname
queue
```

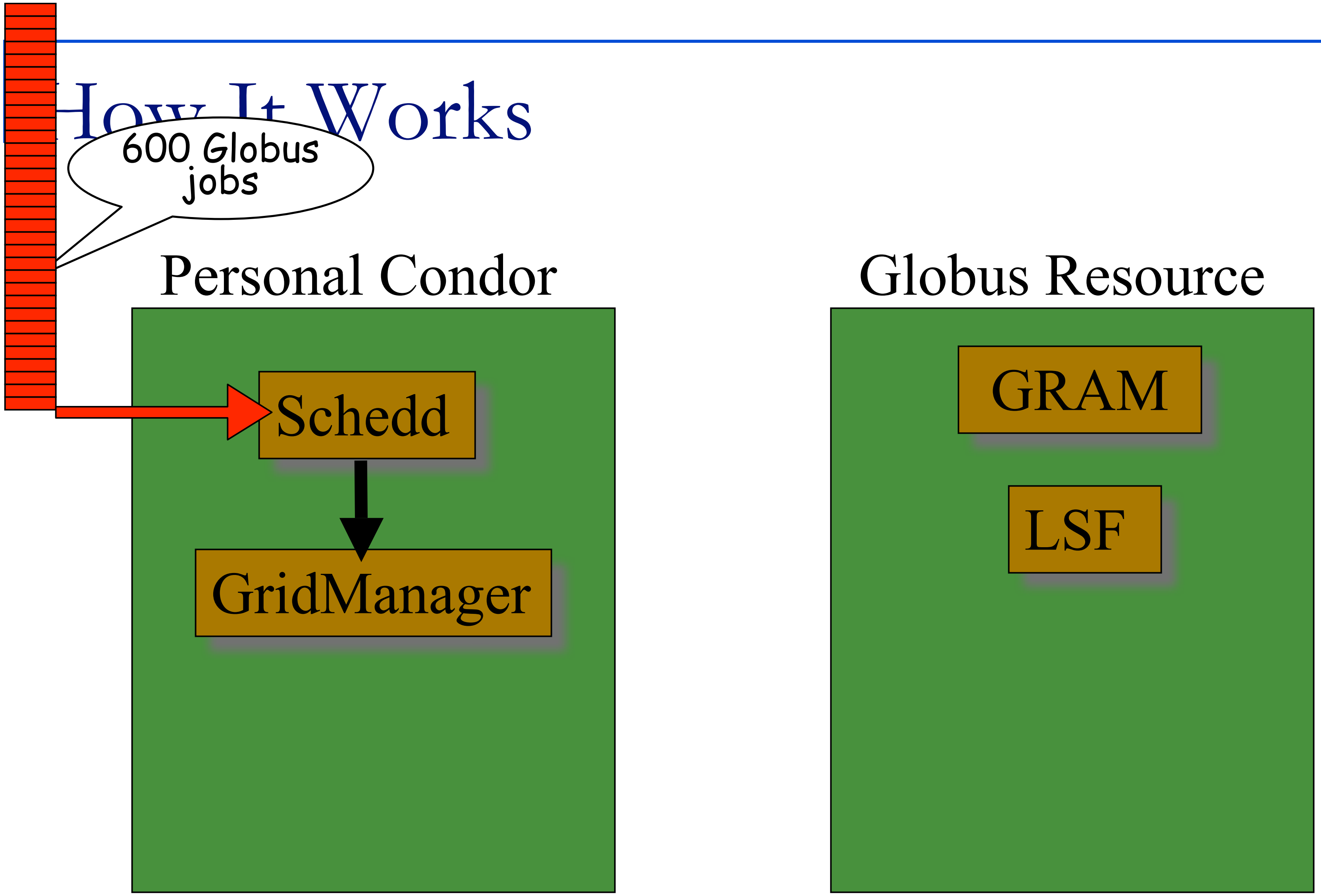
How It Works



How It Works


600 Globus jobs
Personal Condor
Schedd
Globus Resource
GRAM
LSF

How It Works



Personal Condor

Schedd

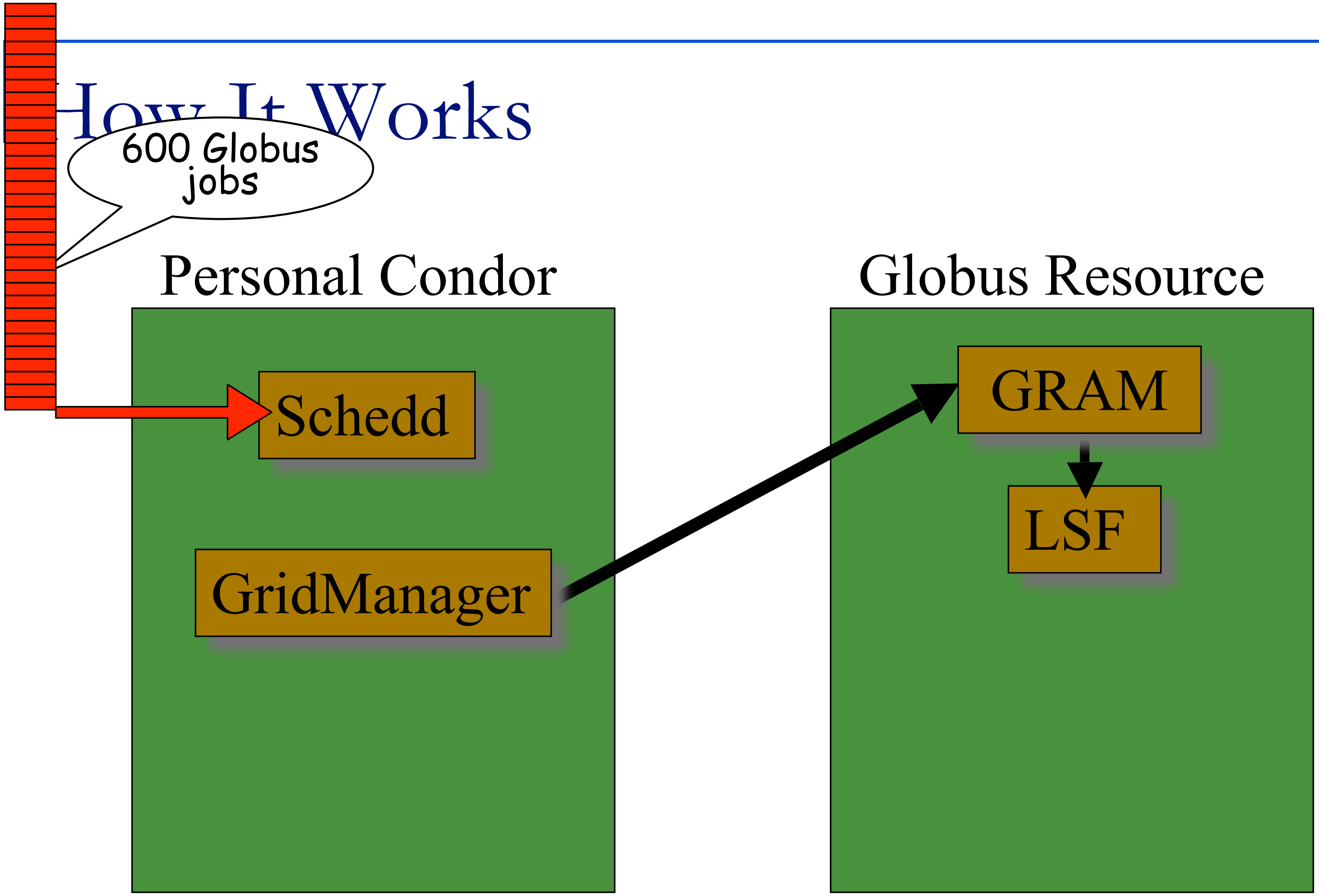
GridManager

Globus Resource

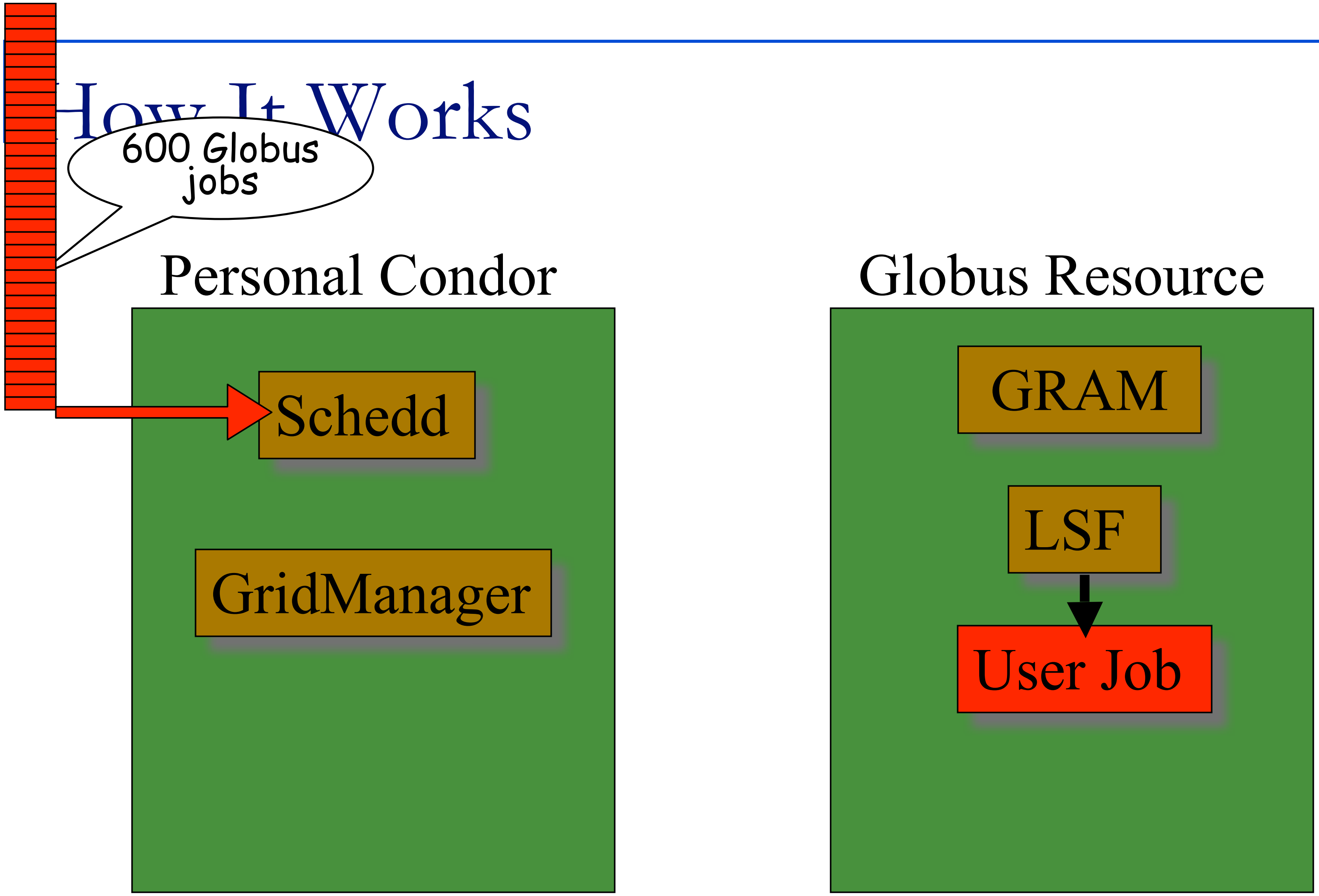
GRAM

LSF

How It Works


600 Globus jobs
Personal Condor
Schedd
GridManager
Globus Resource
GRAM
LSF

How It Works


600 Globus jobs
Personal Condor
Schedd
GridManager
Globus Resource
GRAM
LSF
User Job

Grid Universe Concerns

- What about Fault Tolerance?
  - Local Crashes
    - What if the submit machine goes down?
  - Network Outages
    - What if the connection to the remote Globus jobmanager is lost?
  - Remote Crashes
    - What if the remote Globus jobmanager crashes?
    - What if the remote machine goes down?
- Condor-G's persistent job queue lets it recover from all of these failures
- If a JobManager fails to respond…

This presentation based on: Grid Resources and Job Management

Jaime Frey
Condor Project,
University of Wisconsin-Madison
jfrey@cs.wisc.edu

Grid Summer Workshop
June 26-30, 2006