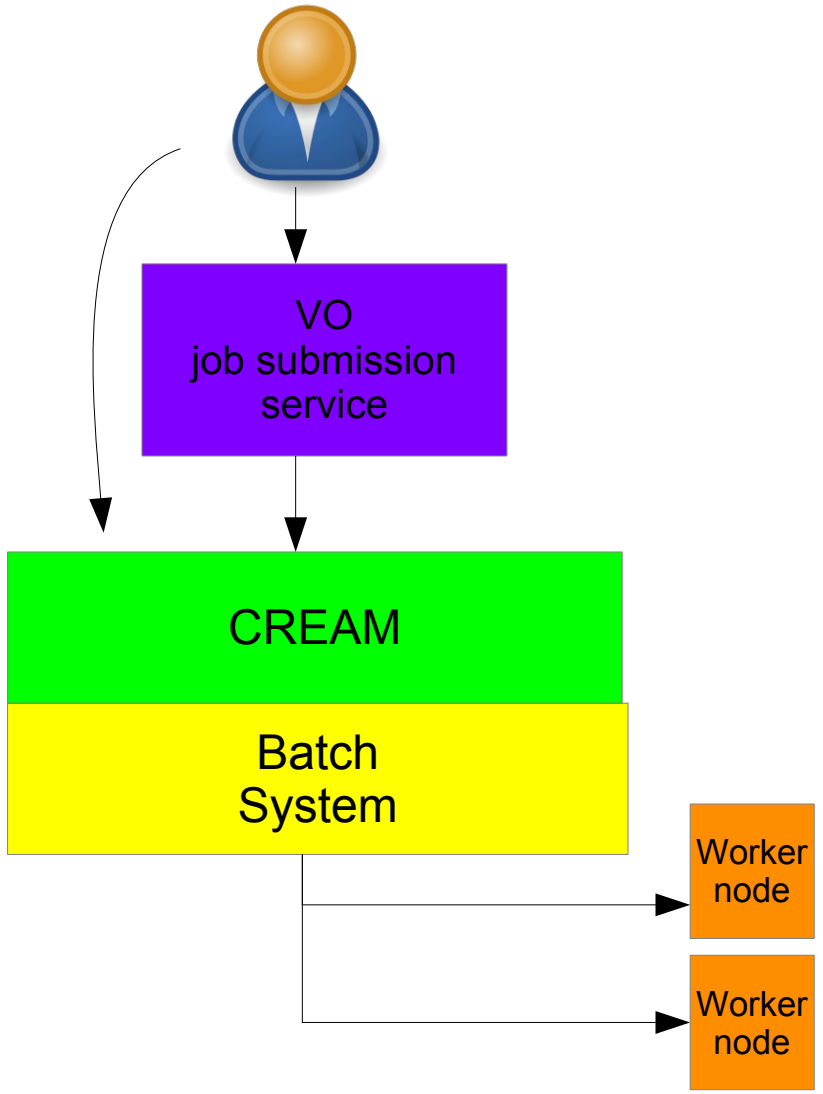
Massimo Sgaravatto
INFN Padova

The CREAM Computing Element

What is CREAM ?

- CREAM (Computing Resource Execution AND Management) service is a Compute Element
- Grid front-end(s) for the computational resources of a site
- Exposes a web service interface
- Users can interact directly with CREAM or through a higher level job submission framework (CondorG, DIRAC, etc.)
- Implemented by INFN Padova group



Batch system

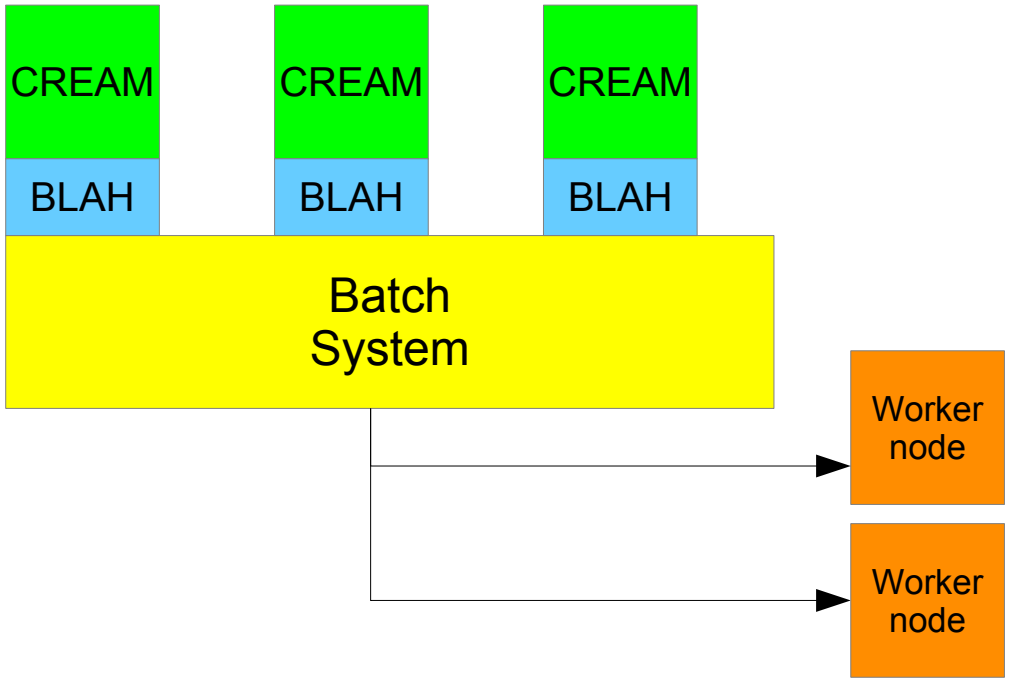
Site resources are managed by a local resource management system (LRMS) that implements the policies imposed by the site administrator, e.g.

- Fair share in resource usage among the VOs
- Usage of some resources only by some VOs
- Priorities
- ...

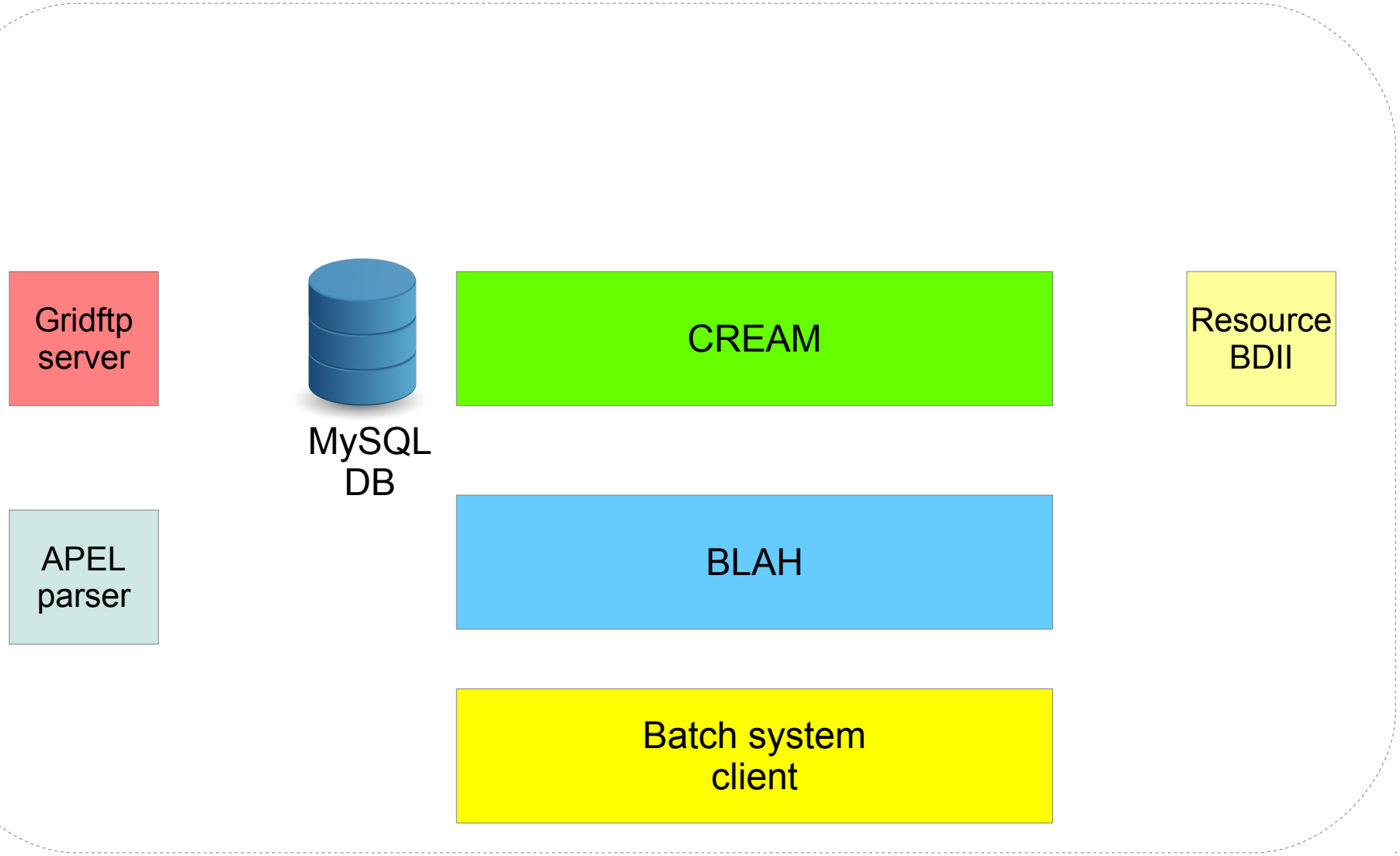
CREAM supports LSF, PBS, HTCondor, SLURM, SGE

Interaction with batch system is managed by BLAH

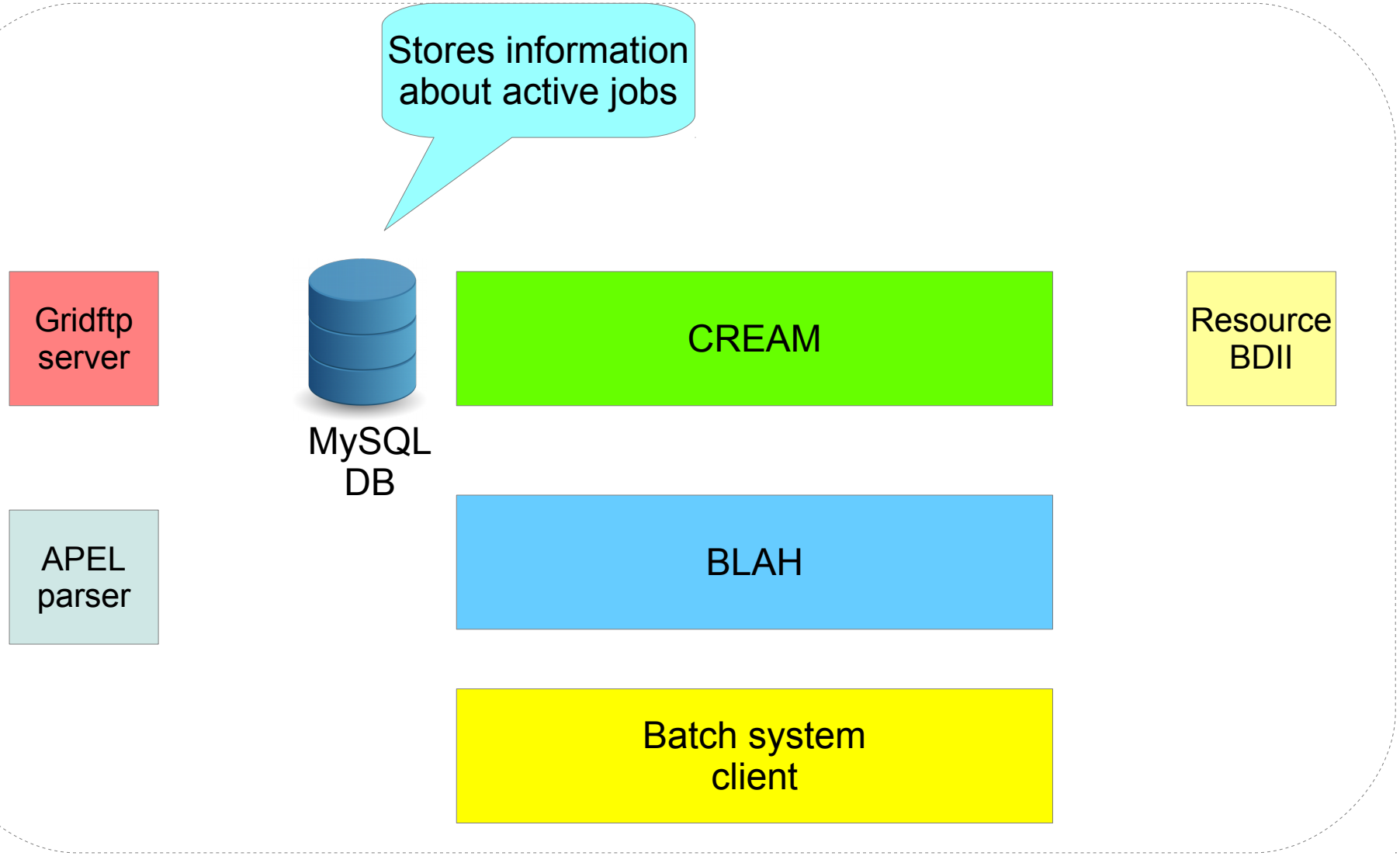
For fault tolerance and load balancing the site can deploy multiple Compute Elements on top of the same batch system



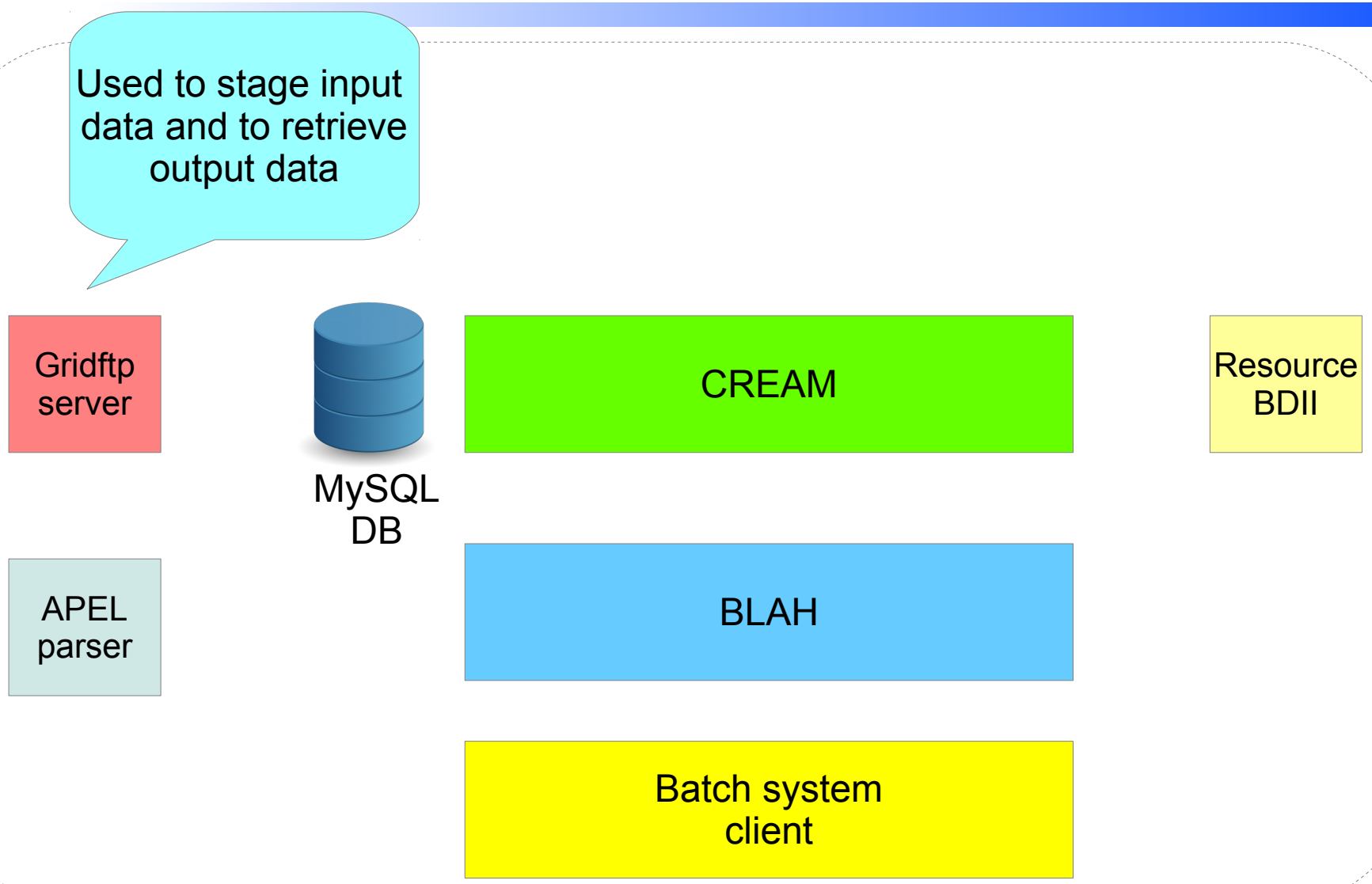
Within a CREAM CE node



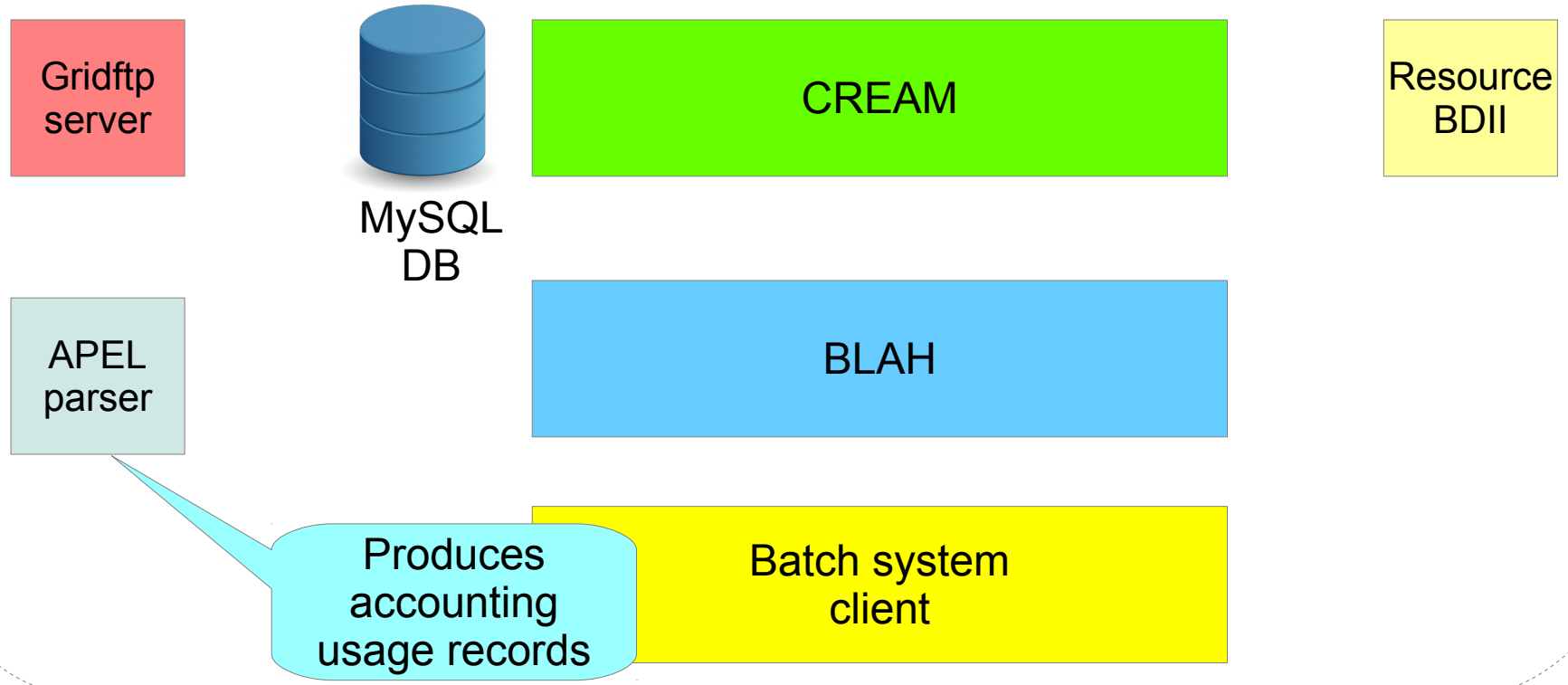
Within a CREAM CE node



Within a CREAM CE node



Within a CREAM CE node



CREAM capabilities

- Job submission
- Staging of input files (InputSandbox, ISB)
  - Can be staged from User Interface machine and/or from remote gridftp servers
- Management of output files (OutputSandbox, OSB)
  - Can be retrieved on the User Interface or stored on remote gridftp servers
- Proxy delegation
  - Can be done automatically with job submission

    glite-ce-job-submit -a -r t2-ce-01.lnl.infn.it:8443/cream-lsf-cms my.jdl

  - Or you can delegate a proxy, and use it for multiple jobs (recommended for performance reasons)

    glite-ce-delegate-proxy -e t2-ce-01.lnl.infn.it:8443 mydel

    glite-ce-job-submit -D mydel -r t2-ce-01.lnl.infn.it:8443/cream-lsf-cms my.jdl

VIRGO CREAM capabilities (cont.ed)

- Proxy renewal
- Job cancel
- Job suspend and resume
- Job status
- Job list
  - Give me the list of all active jobs I submitted on a CREAM CE
- Job purge
  - Remove any data and information related to the job from the CE
  - Explicitly asked by the user, and/or done automatically according a policy specified by the CE admin (e.g. purge jobs finished more than 10 days ago)

VIRGO CREAM capabilities (cont.ed)

- Possibility to forward some requirements to the batch system
  - Require some setting at the site level
- Possibility (for admin) to disable new job submissions
- Self limiting CREAM behavior
  - No new jobs are accepted if the system is overloaded
- Job lease
  - A job can be given a “lease” that can be renewed
  - A job is automatically killed if its lease expires
  - Used to avoid “orphan jobs”
- Support for GPU

CREAM CLI

Functionalities accessible through a CREAM CLI available on all standard User Interfaces

- glite-ce-delegate-proxy
- glite-ce-job-submit
- glite-ce-job-cancel
- glite-ce-job-status
- glite-ce-job-output
- glite-ce-job-list
- glite-ce-job-suspend
- glite-ce-job-resume
- glite-ce-job-lease
- glite-ce-job-purge
- glite-ce-service-info
- glite-ce-allowed-submission
- glite-ce-disable-submission
- glite-ce-enable-submission

CREAM JDL

Job to be submitted is described through a Job Description Language (JDL), based on Condor classads

JDL file is the needed argument of the glite-ce-job-submit command

```
[
# executable is the only mandatory
# attribute
Executable = myexe;
#
# Other attributes
attribute-1 = expression-1;
attribute-2 = expression-2;


attribute-n = expression-n;
]
```

CREAM JDL example 1

```
[
executable="myscript.sh";
arguments="200";
inputsandbox={"file:///home/VIRGO/sgaravattovirgo/myscript.sh",
"file:///home/VIRGO/sgaravattovirgo/inputfile"};
stdoutput="std.out";
stderror="std.err";
outputsandbox={"std.out", "std.err", "outputfile"};
outputsandboxbasedesturi="gsiftp://t2-gftp-01.lnl.infn.it/data/cms/store/user/sgaravat";
]
```

Input files staged from User Interface
Output files staged in a gridftp server

CREAM JDL example 2

```
[
executable="myscript.sh";
arguments="200";
inputsandbox={"file:///home/VIRGO/sgaravattovirgo/myscript.sh",
"gsiftp://t2-gftp-01.lnl.infn.it/data/cms/store/user/sgaravat/inputfile"};
stdoutput="std.out";
stderror="std.err";
outputsandbox={"std.out", "std.err", "outputfile"};
outputsandboxbasedesturi="gsiftp://localhost";
]
```

One input file staged from User Interface
The other input file staged from a gridftp server

Output files must stay in the CE (they will be retrieved using *glite-ce-job-output*)

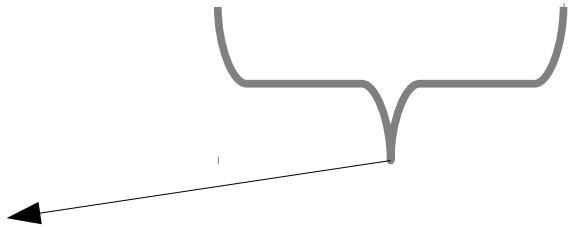
CREAM JDL example 3

```
[
Executable= "glidein_startup.sh";
Arguments = "-v std -name v3_2 -entry CMSHTPC_T2_IT_Legnaro_ce01";
StdOutput = "job.2346338.0.out";
StdError = "job.2346338.0.err";
InputSandbox = {"glidein_startup.sh"} ;
OutputSandbox = {"job.2346338.0.out","job.2346338.0.err"} ;
outputsandboxbasedesturi = "gsiftp://localhost";
HostNumber = 1;
CPUNumber = 8;
]
```

Job requires 8 cores
They must be in the same host

Job submission

glite-ce-job-submit [-a | -D <delegation-id>] -r <CREAM-CE-ID> <jdl-file>

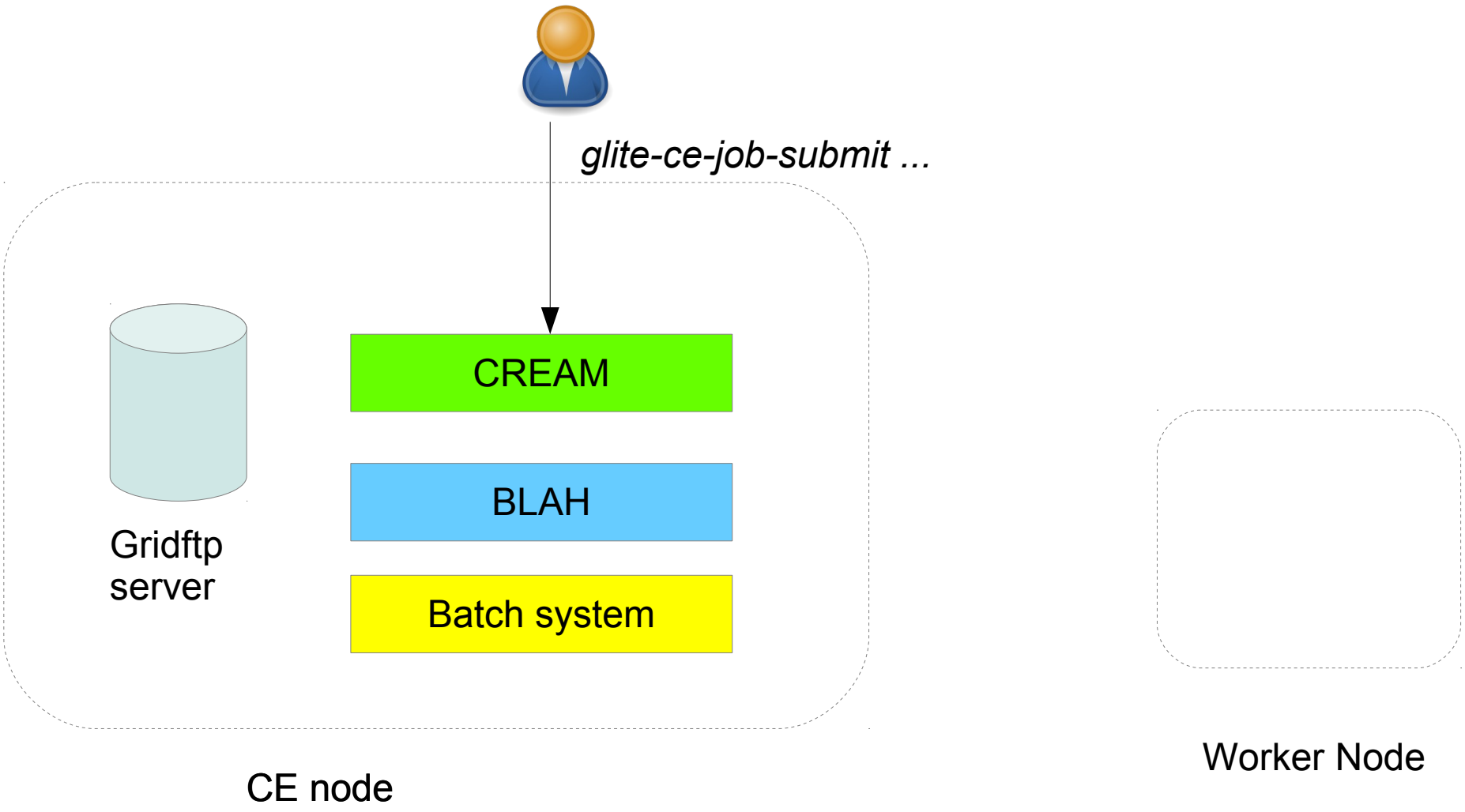


<hostname>:<port>/cream-<batchsystem>-<queuename>

E.g.

glite-ce-job-submit -a -r ce04-lcg.cr.cnaf.infn.it:8443/cream-lsf-virgo_h12 my.jdl

Job submission

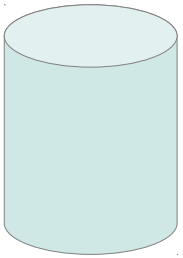


Job submission

CREAM **registers** the job and prepares the JobWrapper

JobWrapper is a script which basically:

- stages in input data
- runs the user payload
- stages out output data



Gridftp server

CREAM

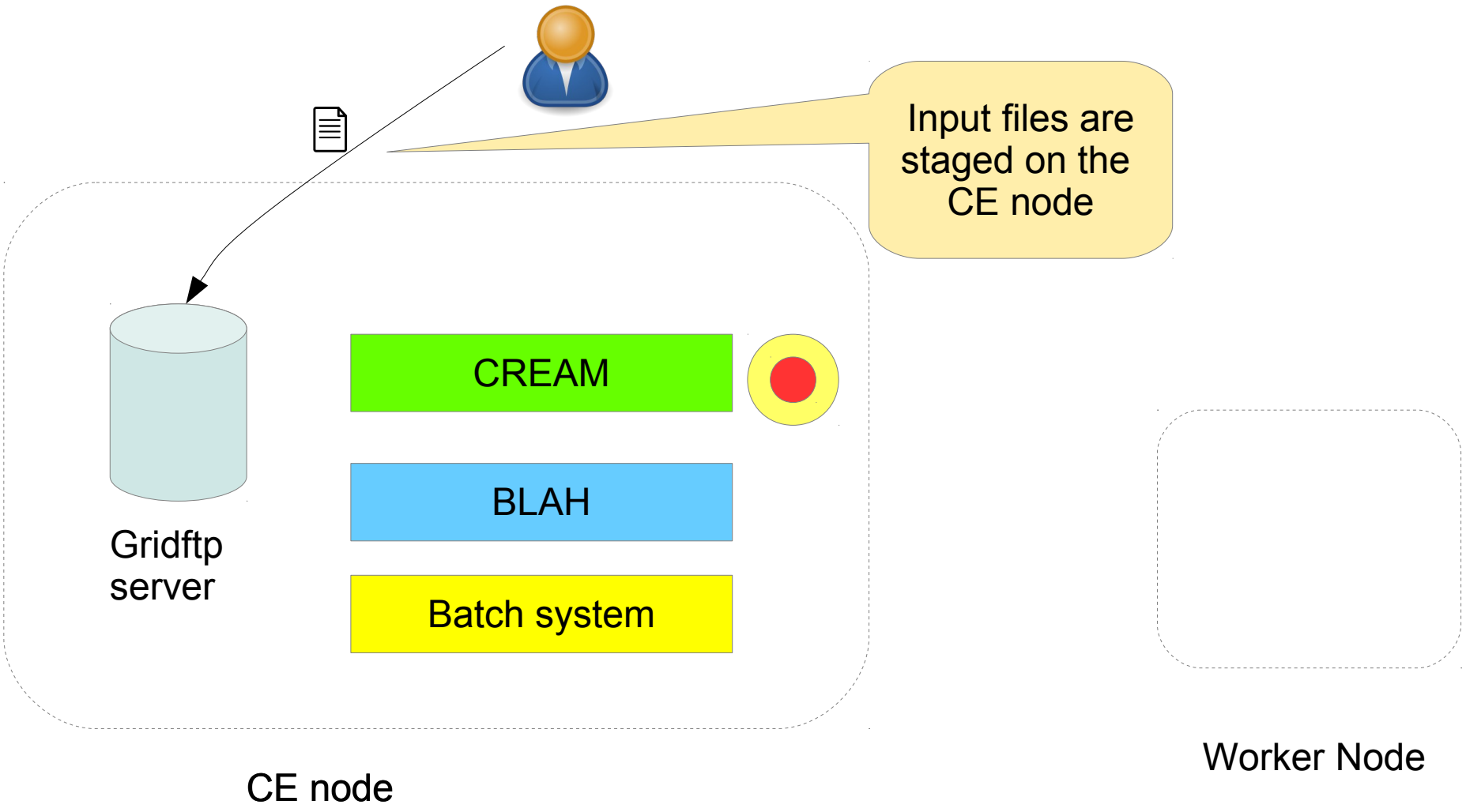
BLAH

Batch system

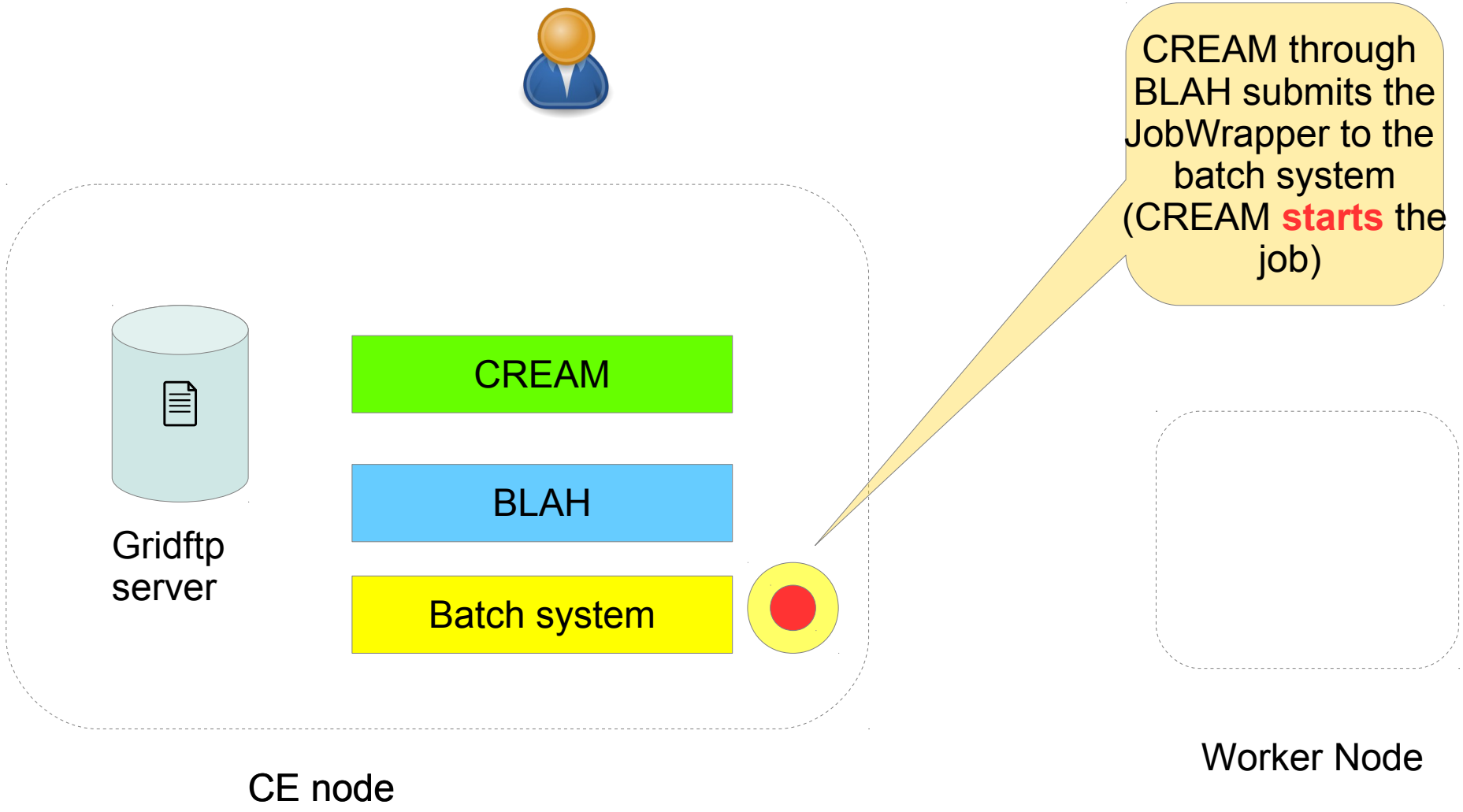
CE node

Worker Node

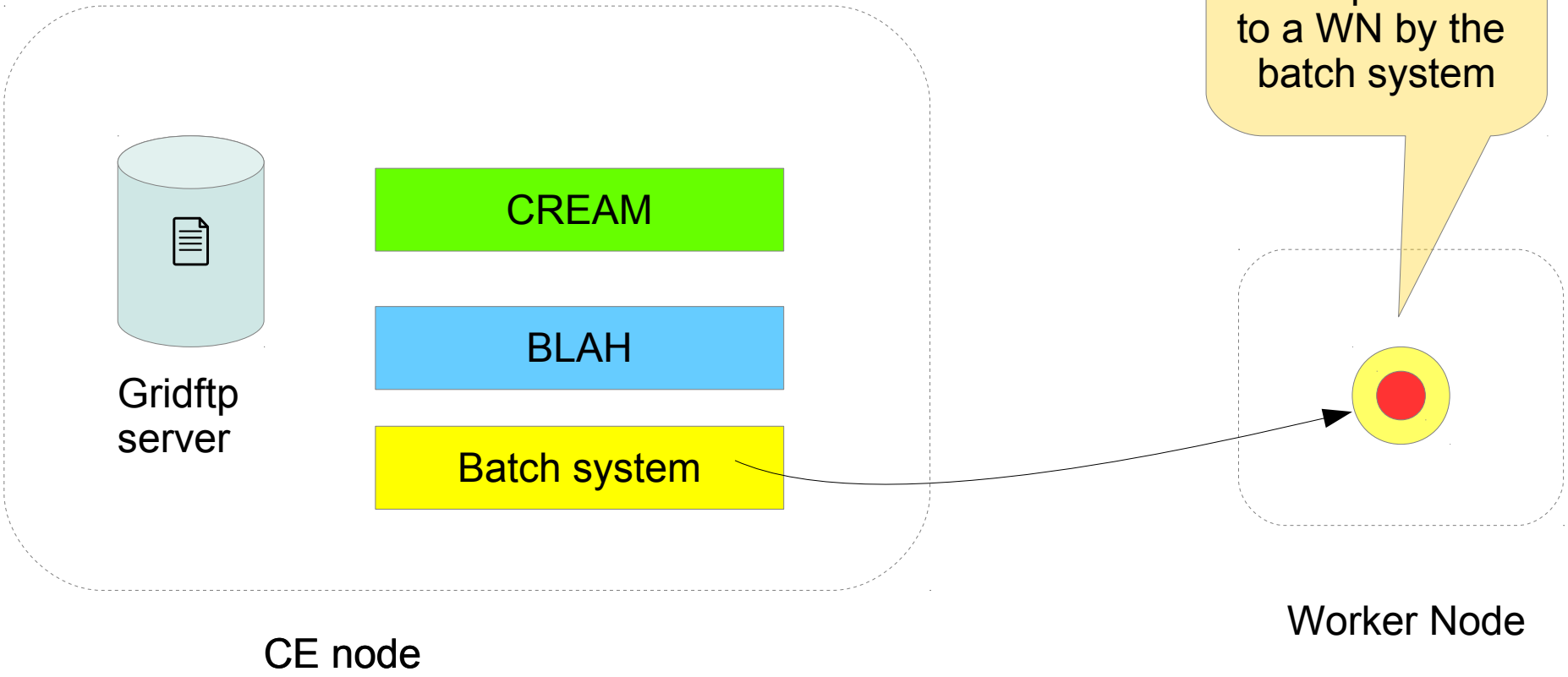
Job submission



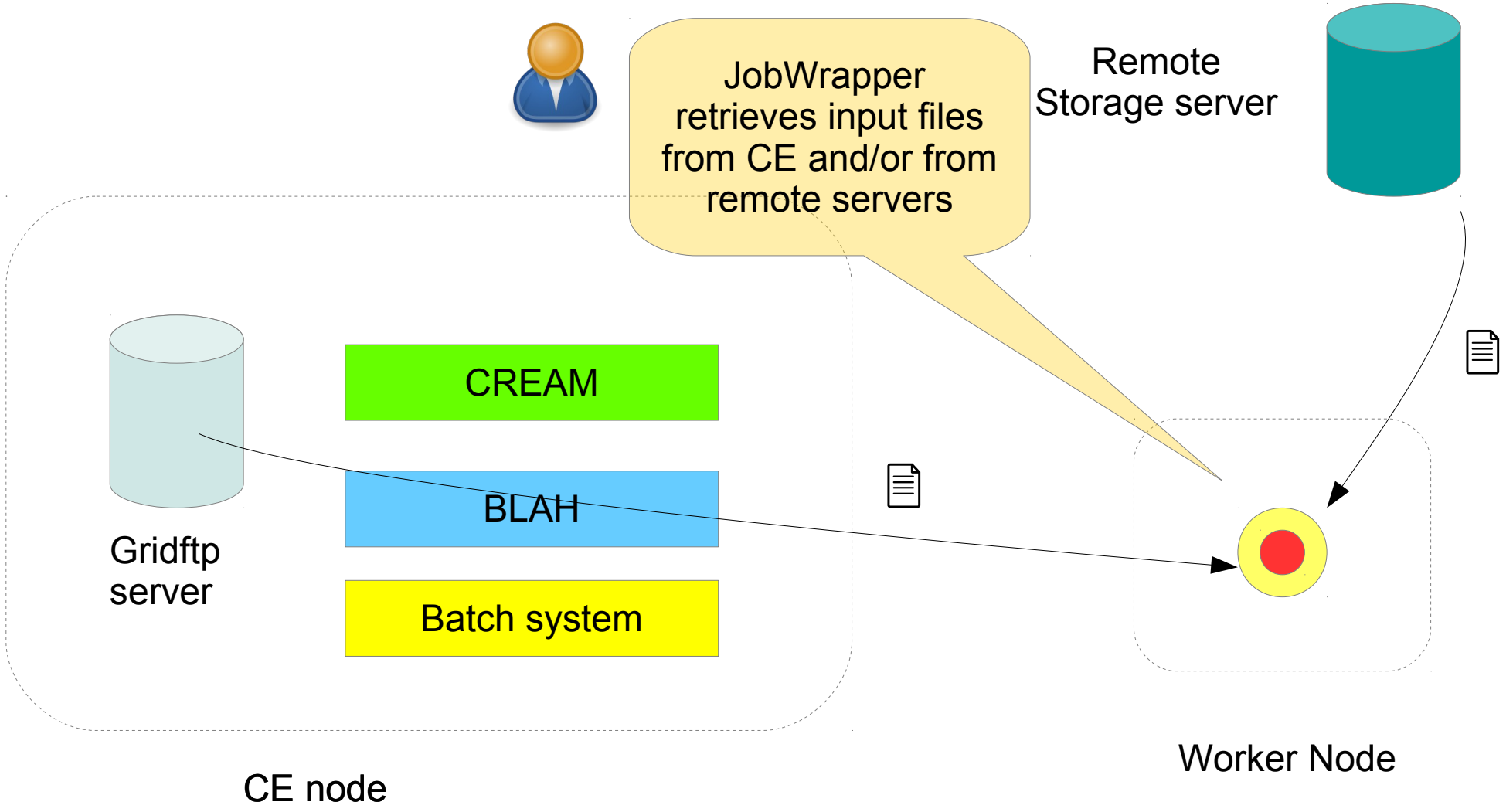
Job submission



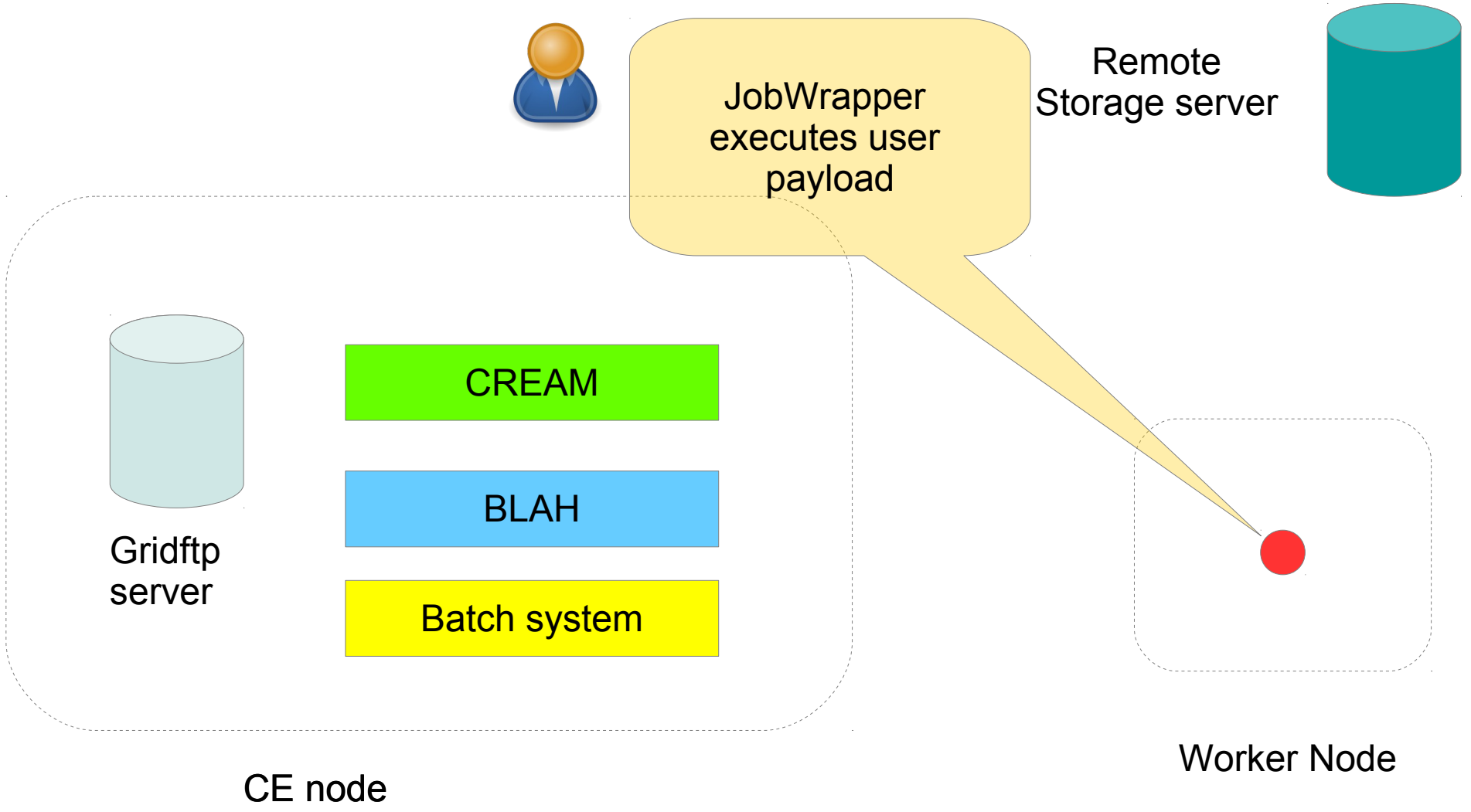
Job submission



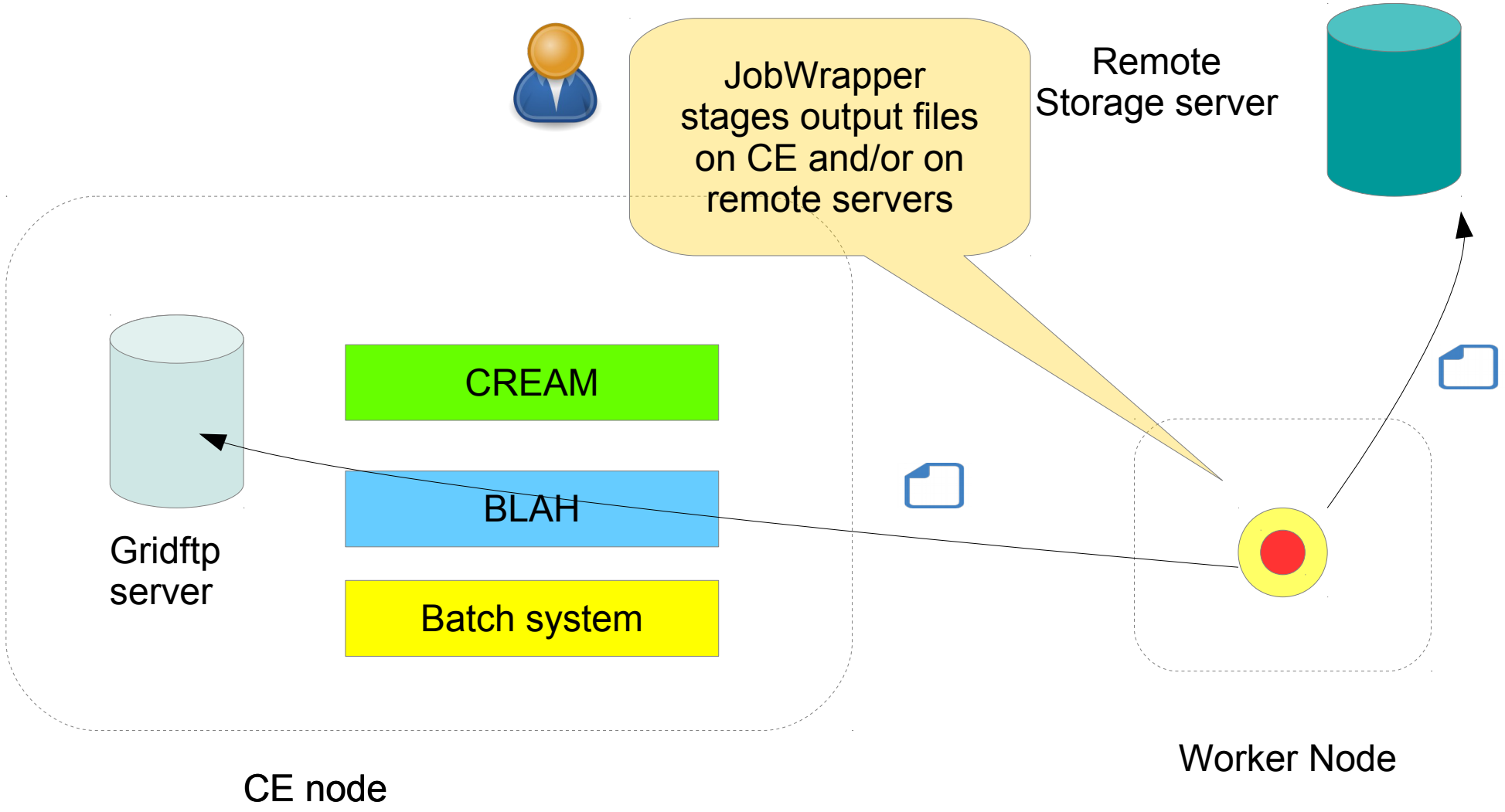
Job submission



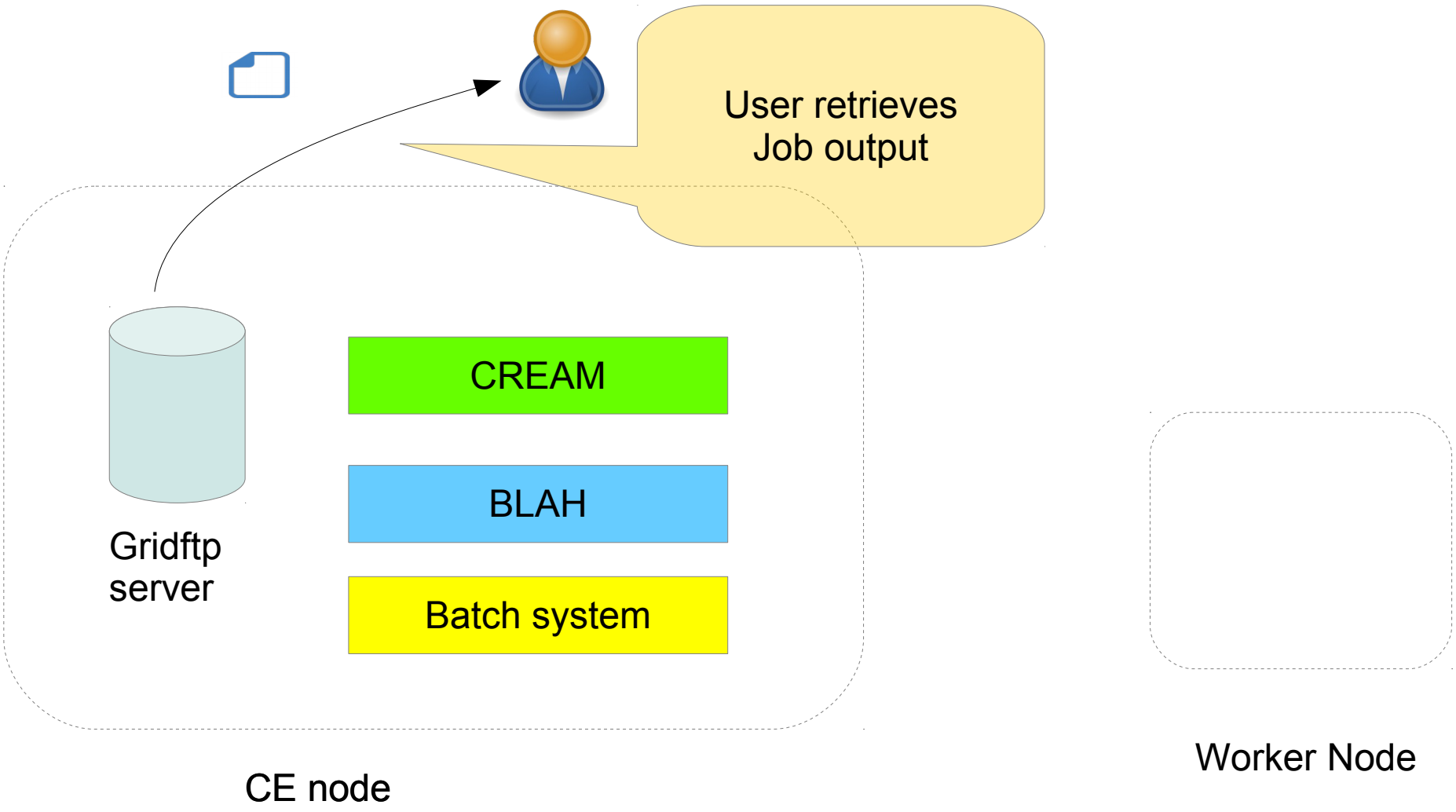
Job submission



Job submission

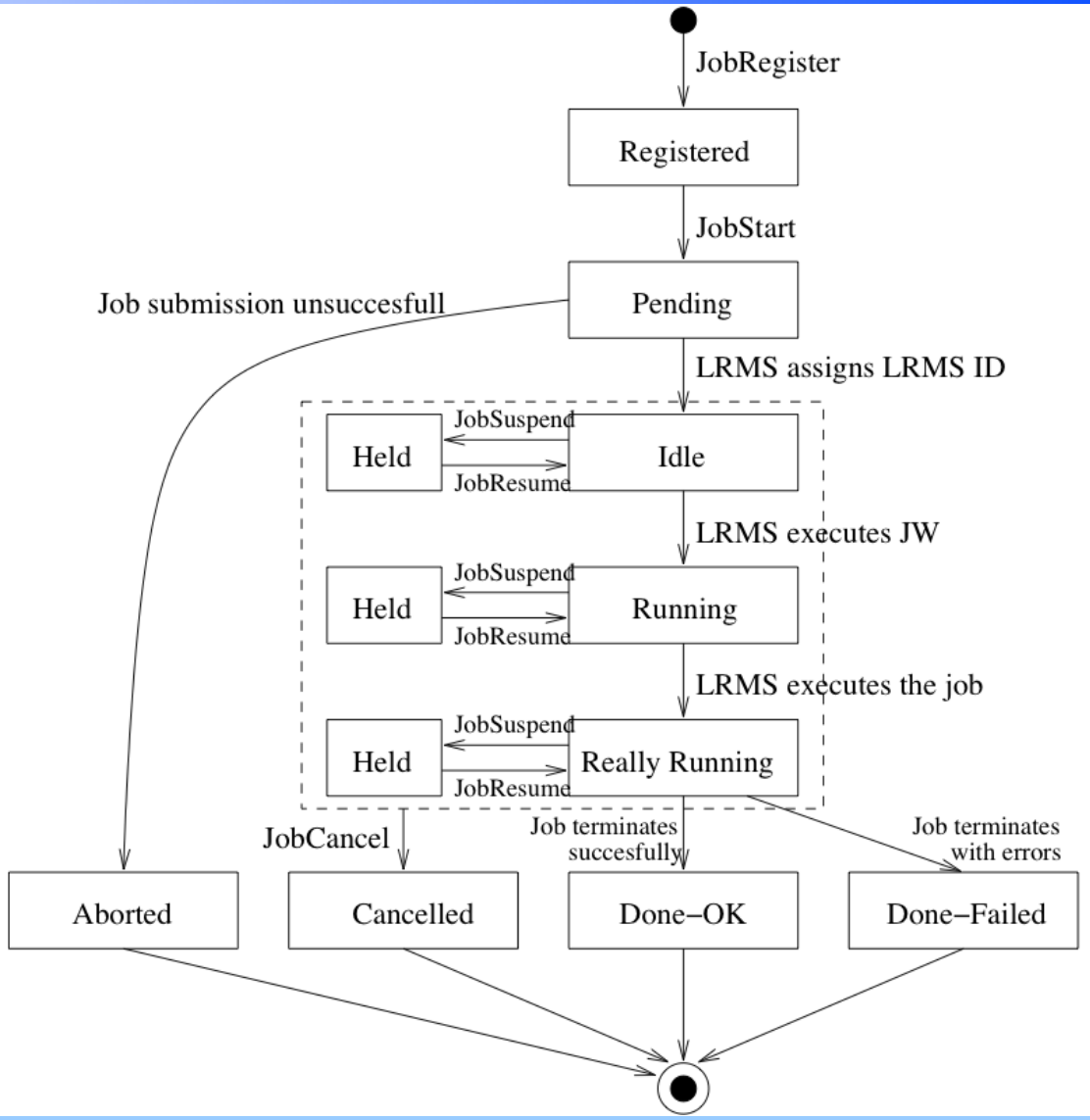


Job submission



CREAM job states

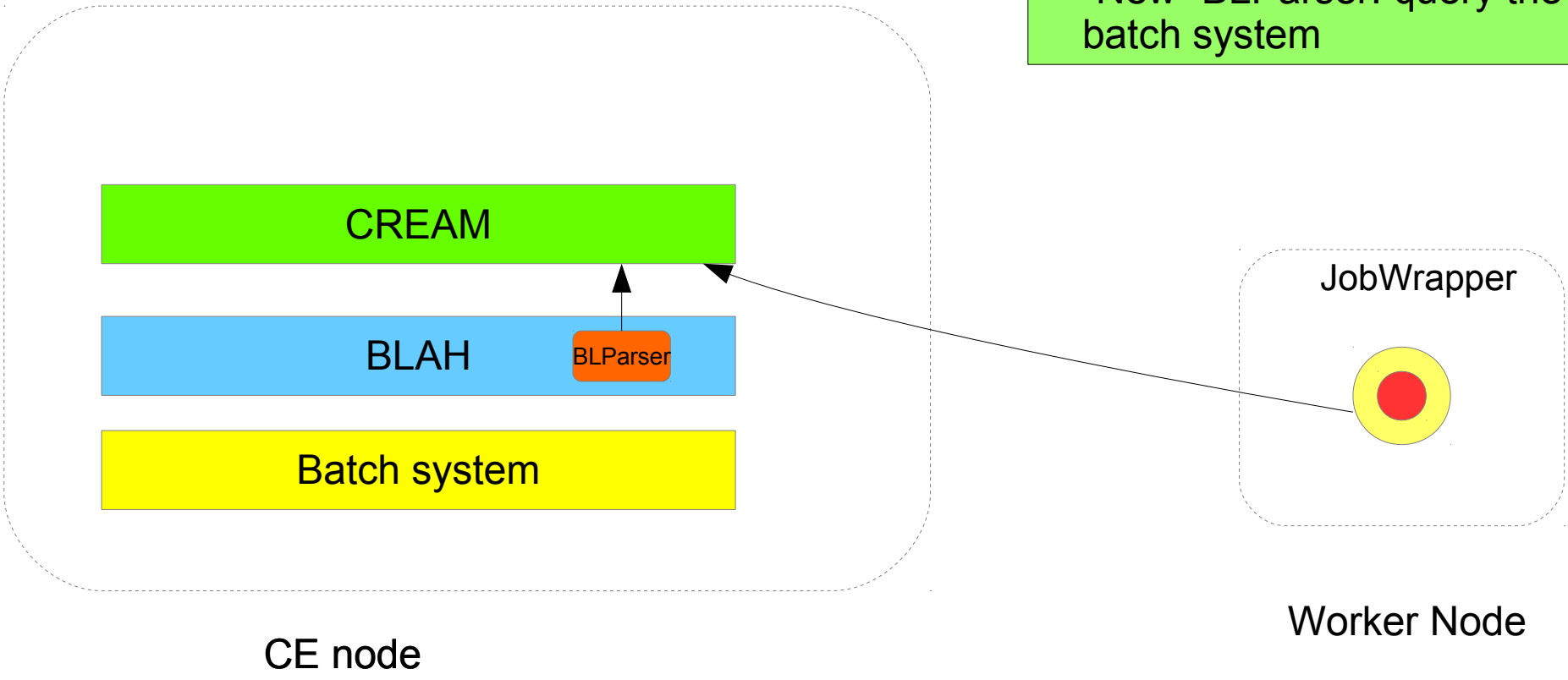
REGISTERED: job in CREAM but not yet submitted to batch system
PENDING: job being submitted to batch system
IDLE: job queued in the batch system
RUNNING: The JobWrapper is running
REALLY-RUNNING: the user payload is running
HELD: job is suspended
CANCELLED: job was cancelled
DONE-OK: job successfully executed
DONE-FAILED: job executed but with errors
ABORTED: some errors happened during the management of the job



How does CREAM detect job status changes ?

CREAM receives notifications about job status changes by BLParser and JobWrapper
Some events are detected by both entities, some other events by only BLParser or only by JobWrapper

Two possible configurations for Blparser
- “Old” Blparser: parses batch system logs
- “New” BLParser: query the batch system



Authorization

CREAM usually configured to implement VO based authorization

CREAM can manage itself the authorization or can rely on a site Authorization system (ARGUS)

Managing authorization means:

- Deciding if the user is allowed to use the resource
- Credential mapping: map the Grid user to a specific local (pool) account

```
"/virgo/virgo/Role=NULL/Capability=NULL" .virgo
"/virgo/virgo" .virgo
"/virgo/ligo/Role=NULL/Capability=NULL" .vligo
"/virgo/ligo" .vligo
```

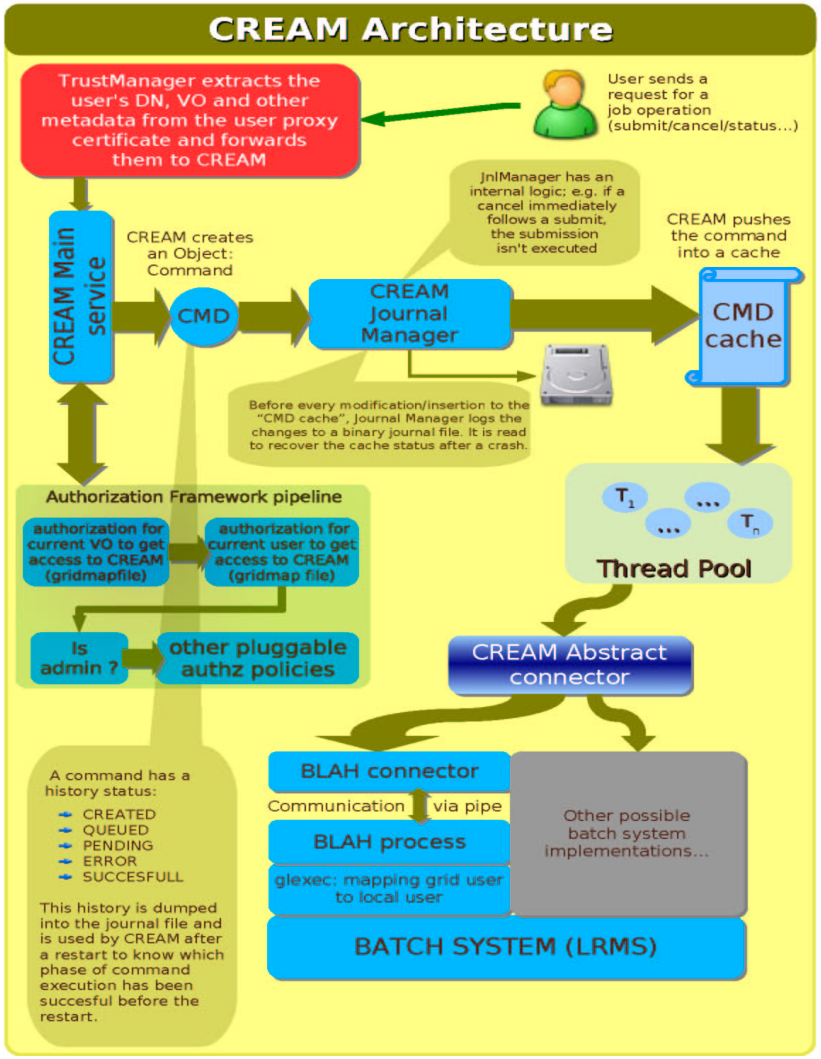
In this example users belonging to /virgo/virgo group are mapped to different pool accounts wrt users belonging to /virgo/ligo group.

Site can then manage the 2 user groups in different ways (e.g. different shares)

THE END

Questions?

CREAM architecture



“Simplified” view of CREAM architecture