Active/Active HA Job Scheduler and Resource Management

a proof-of-concept implementation

Kai Uhlemann

Oak Ridge National Laboratory, USA
University of Reading, UK
University of Applied Sciences Berlin, Germany

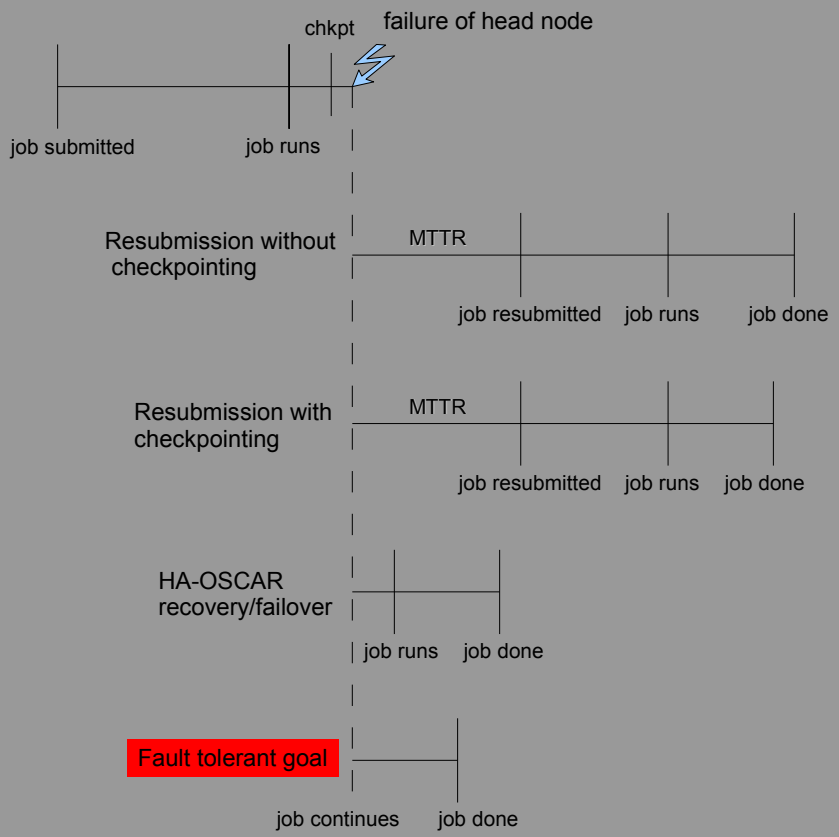
March 15, 2006

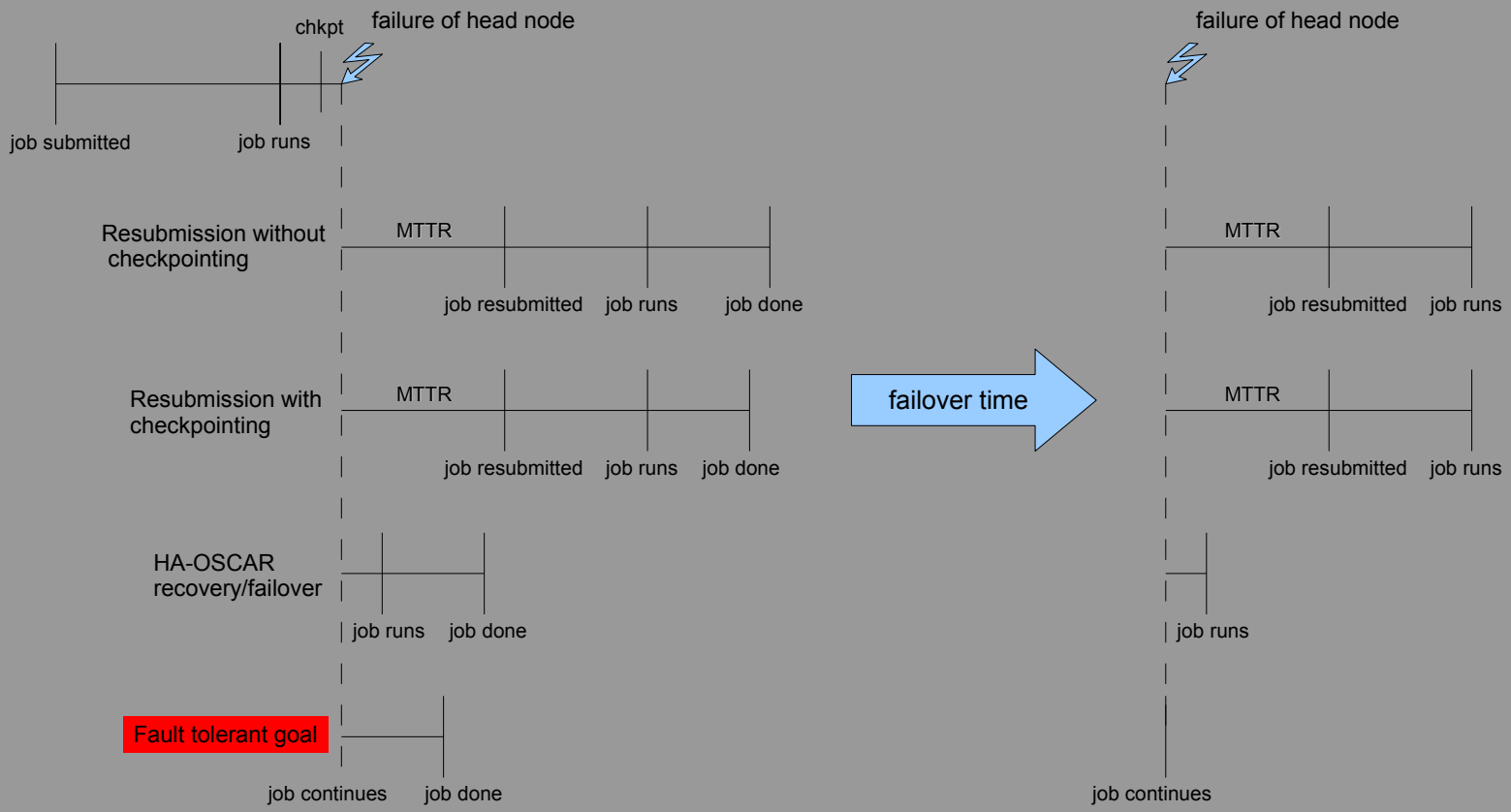
Content

Content

Content

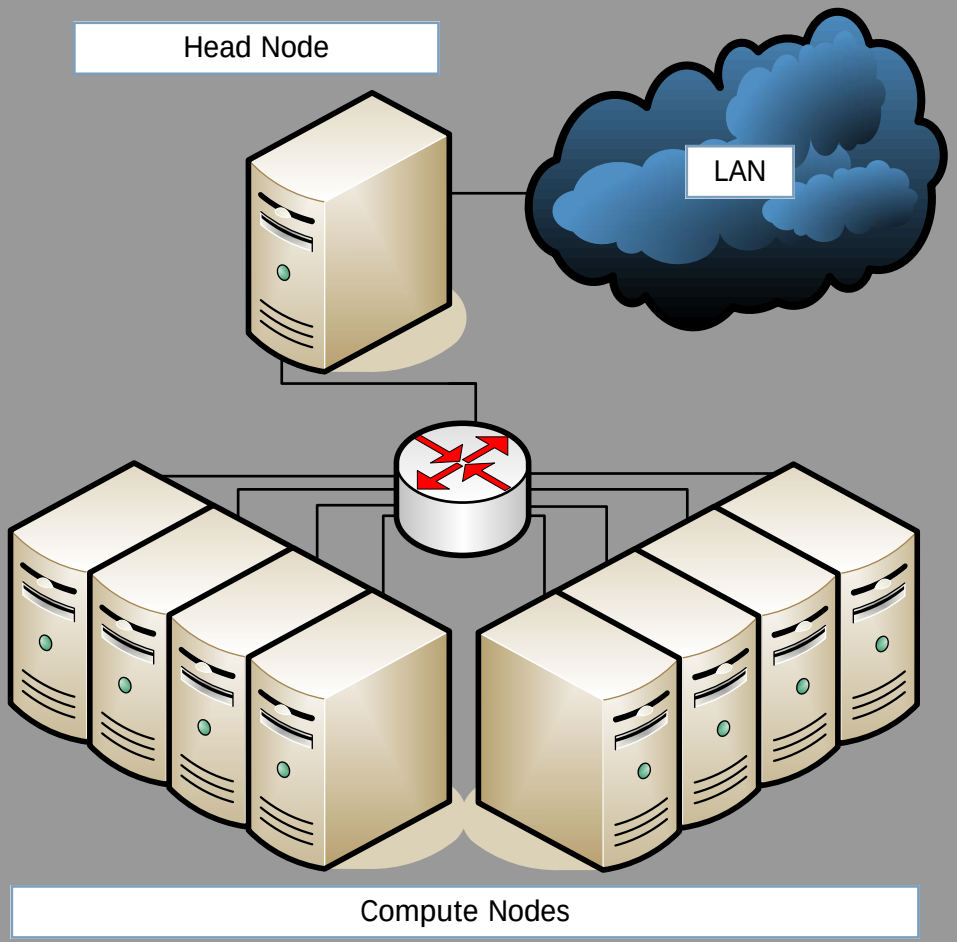
- Research area of high availability for job scheduler
- Efforts focused on the head node
- Proposed solution uses the virtual synchrony paradigm
- Proof-of-concept application shows possible design solution


chkpt
failure of head node
job submitted
job runs
Resubmission without checkpointing
MTTR
job resubmitted
job runs
job done
Resubmission with checkpointing
MTTR
job resubmitted
job runs
job done
HA-OSCAR recovery/failover
job runs
job done
Fault tolerant goal
job continues
job done

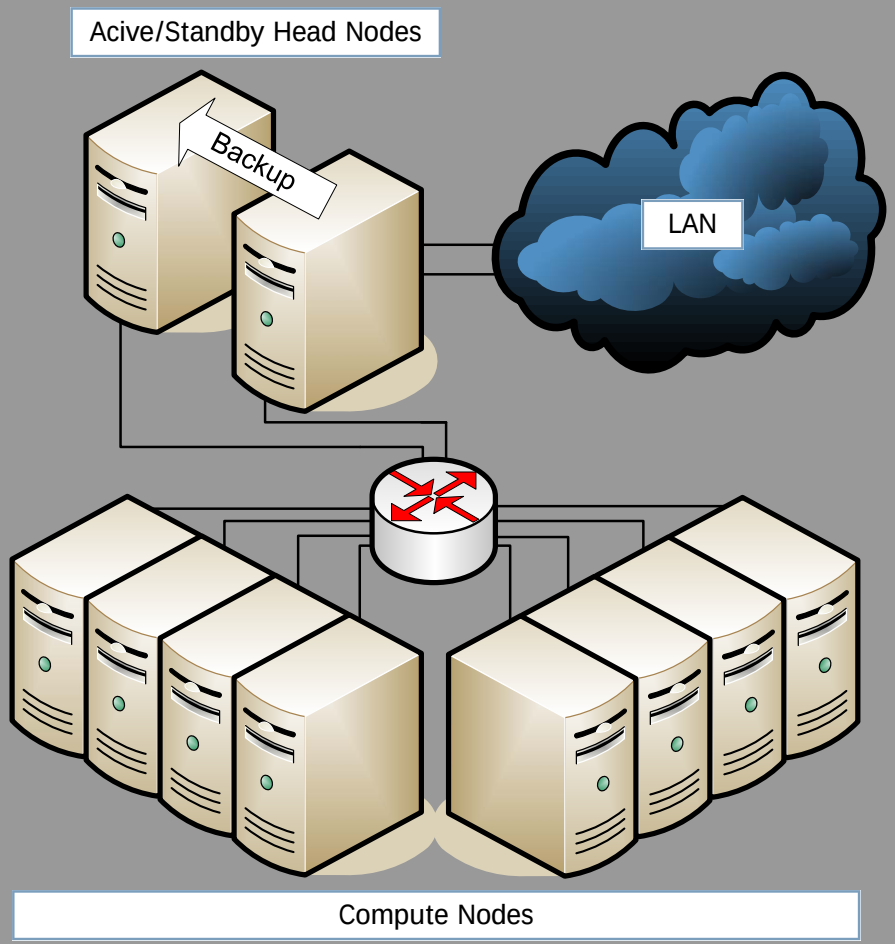

chkpt
failure of head node
job submitted
job runs
Resubmission without checkpointing
MTTR
job resubmitted
job runs
job done
Resubmission with checkpointing
MTTR
job resubmitted
job runs
job done
HA-OSCAR recovery/failover
job runs
job done
Fault tolerant goal
job continues
job done
failover time
failure of head node
MTTR
job resubmitted
job runs
MTTR
job resubmitted
job runs
job runs
job continues

- Job scheduler service providing Active/Active HA
  - No loss of scheduled job
  - No loss of running job
  - No restart of running jobs
- Leading to uninterruptible service availability
- Seamless failover (actually none)
- Dynamic reconfiguration and recovery are completely masked
- High transparency of HA capabilities from the user

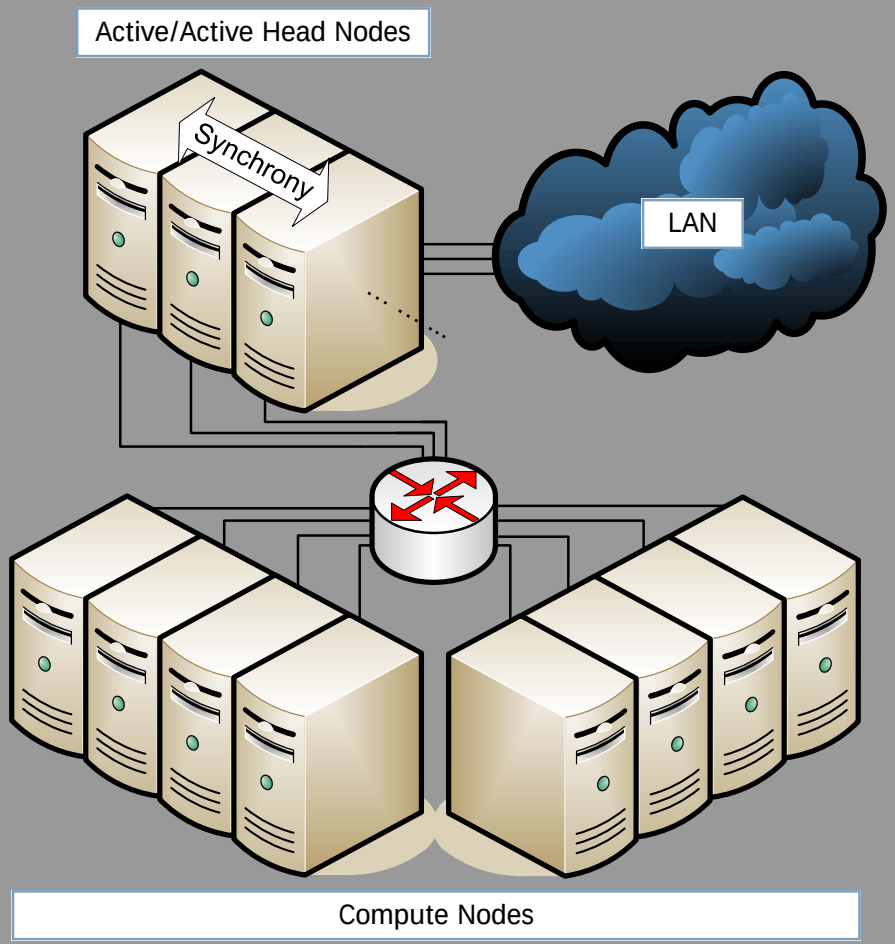
Beowulf cluster architecture



HA-OSCAR cluster architecture



Symmetric Active/Active HA cluster architecture



- Active state replication using virtual synchrony
- All messages are processed as uninterruptible events
- Global process state provided by external group communication system
- External replication used for resource and job management
- External components should be replaceable with new or similar ones
- Introduction of symmetric multi headed cluster architecture
- Any amount of head nodes possible $\Rightarrow$ scalable high availability

MTBF 5000 hours MTTR 72 hours

$$t_{down} = 8760 \cdot (1 - A)$$

$$A_{component} = \frac{MTBF}{MTBF + MTTR}$$

$$A = 1 - (1 - A_{component})^n$$

No. HN	Availability	Est. downtime
1	*98,580441640%*	**5d 4h 21min**
2		
3		
4		
5		

MTBF 5000 hours MTTR 72 hours

$$t_{down} = 8760 \cdot (1 - A)$$

$$A_{component} = \frac{MTBF}{MTBF + MTTR}$$

$$A = 1 - (1 - A_{component})^n$$

No. HN	Availability	Est. downtime
1	*98,580441640%*	*5d 4h 21min*
2	*99,979848540%*	**1h 45min**
3		
4		
5		

MTBF 5000 hours MTTR 72 hours

$$t_{down} = 8760 \cdot (1 - A)$$

$$A_{component} = \frac{MTBF}{MTBF + MTTR}$$

$$A = 1 - (1 - A_{component})^n$$

No. HN	Availability	Est. downtime
1	*98,580441640%*	*5d 4h 21min*
2	*99,979848540%*	*1h 45min*
3	*99,999713938%*	**1min 30s**
4		
5		

MTBF 5000 hours MTTR 72 hours

$$t_{down} = 8760 \cdot (1 - A)$$

$$A_{component} = \frac{MTBF}{MTBF + MTTR}$$

$$A = 1 - (1 - A_{component})^n$$

No. HN	Availability	Est. downtime
1	*98,580441640%*	*5d 4h 21min*
2	*99,979848540%*	*1h 45min*
3	*99,999713938%*	*1min 30s*
4	*99,999995939%*	**1s**
5		

MTBF 5000 hours MTTR 72 hours

$$t_{down} = 8760 \cdot (1 - A)$$

$$A_{component} = \frac{MTBF}{MTBF + MTTR}$$

$$A = 1 - (1 - A_{component})^n$$

No. HN	Availability	Est. downtime
1	*98,580441640%*	*5d 4h 21min*
2	*99,979848540%*	*1h 45min*
3	*99,999713938%*	*1min 30s*
4	*99,999995939%*	*1s*
5	**99,999999942%**	**18ms**

Group communication system transis

- Communication facilities for virtual synchrony
- Handles group membership changes

Resource management PBS TORQUE

- Enhanced version (>= version 2.0p1)
- Enabled multiserver architecture support

Job scheduler maui

- Commonly used in conjunction with PBS

Group communication system transis

- Communication facilities for virtual synchrony
- Handles group membership changes

Resource management PBS TORQUE

- Enhanced version (>= version 2.0p1)
- Enabled multiserver architecture support

Job scheduler maui

- Commonly used in conjunction with PBS

Group communication system transis

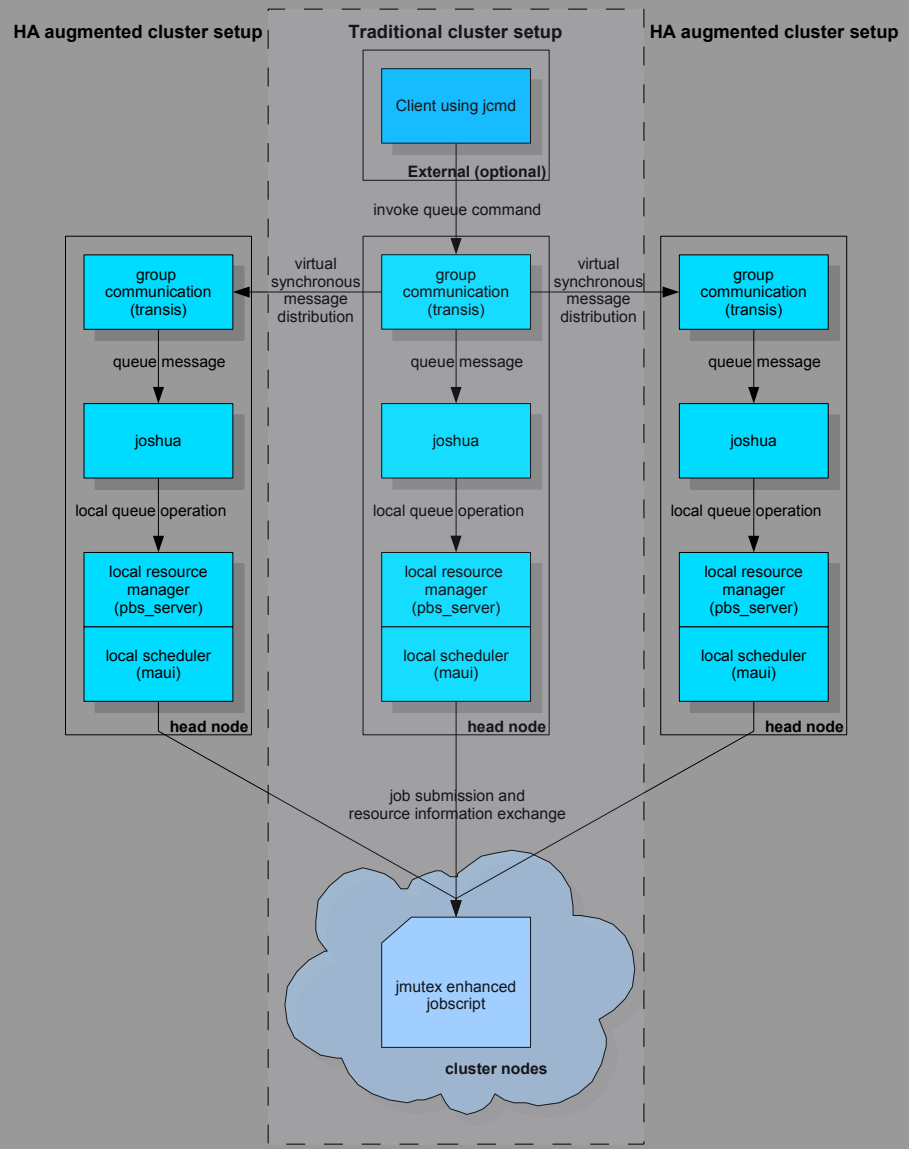
- Communication facilities for virtual synchrony
- Handles group membership changes

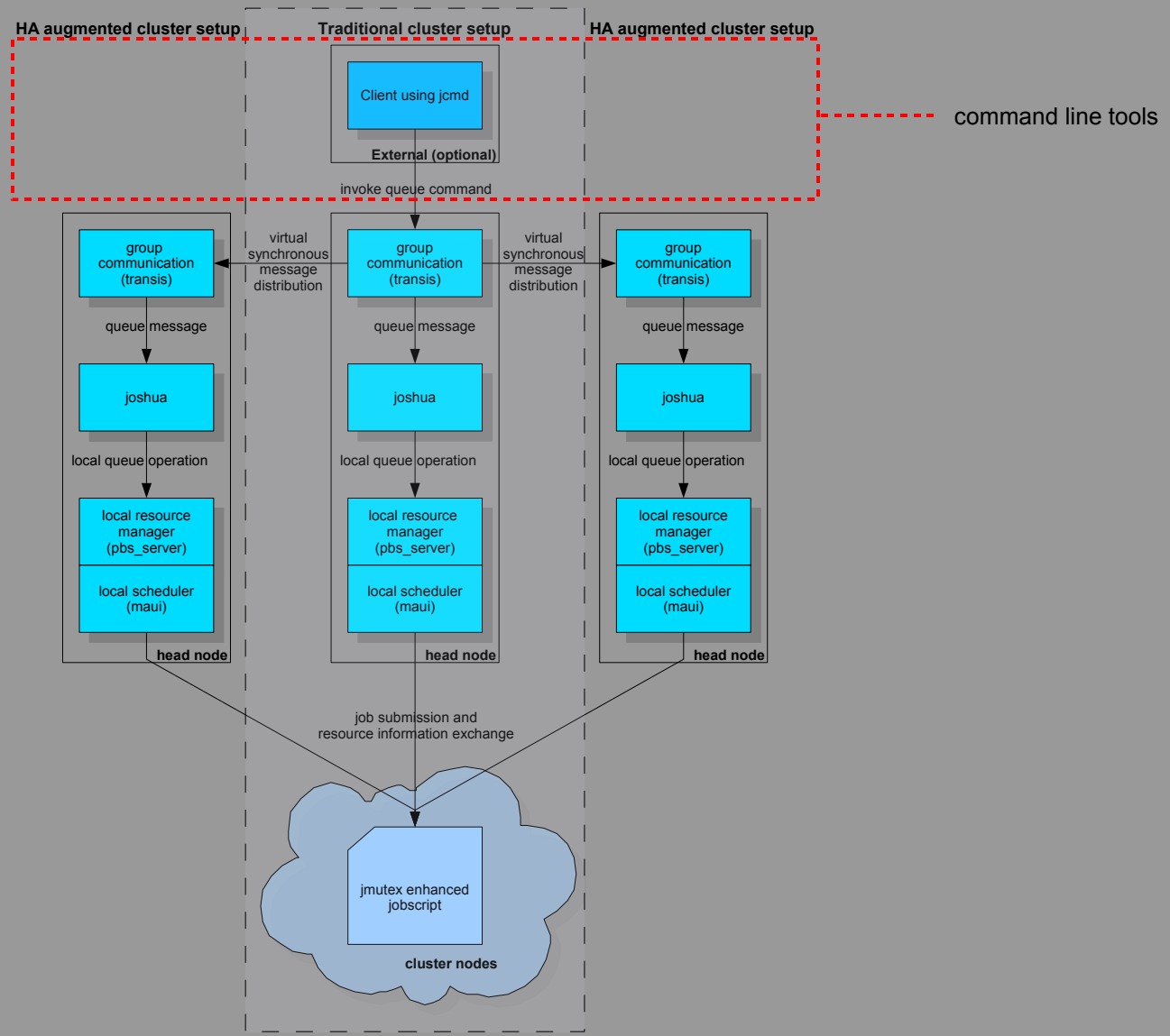
Resource management PBS TORQUE

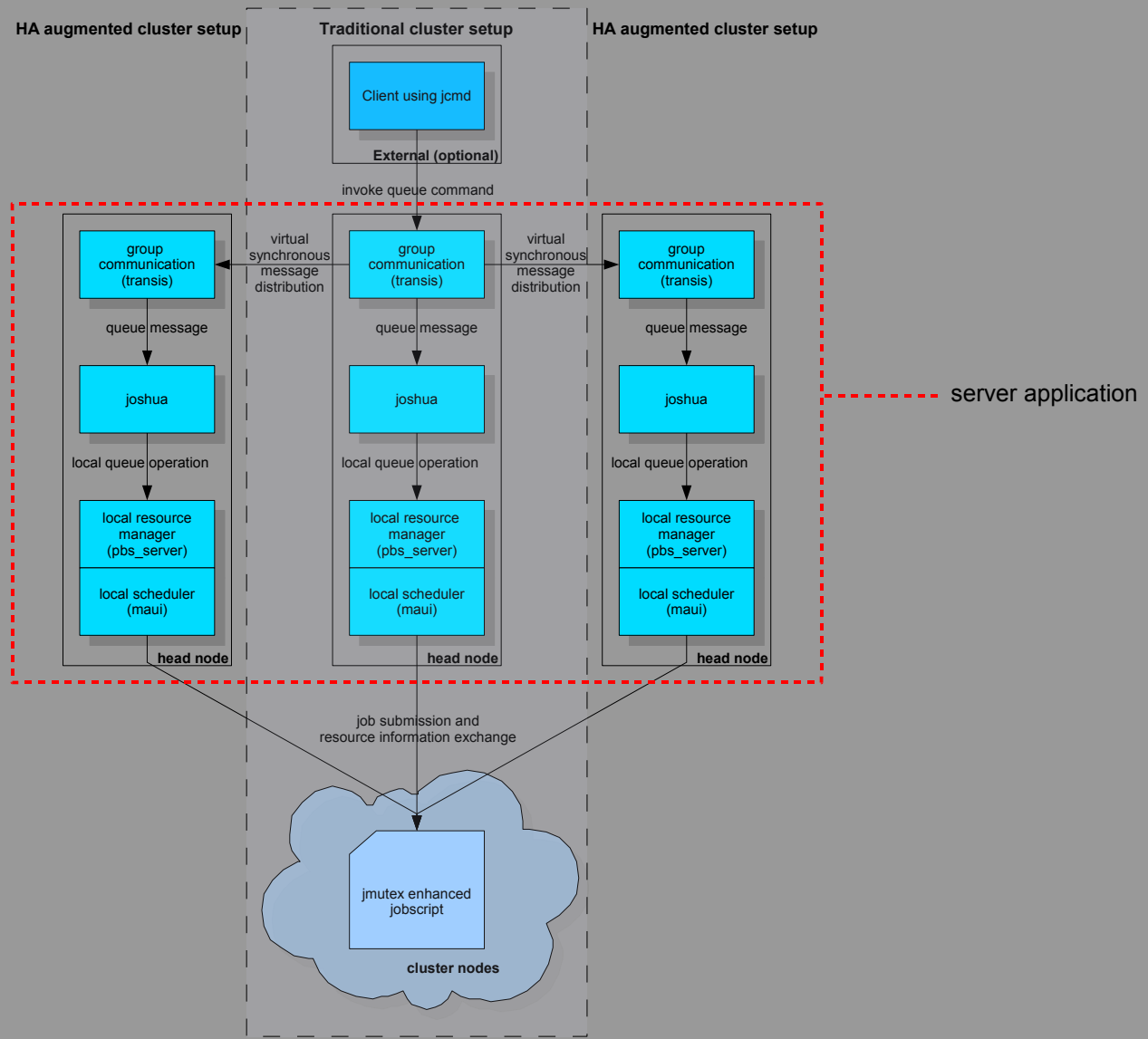
- Enhanced version (>= version 2.0p1)
- Enabled multiserver architecture support

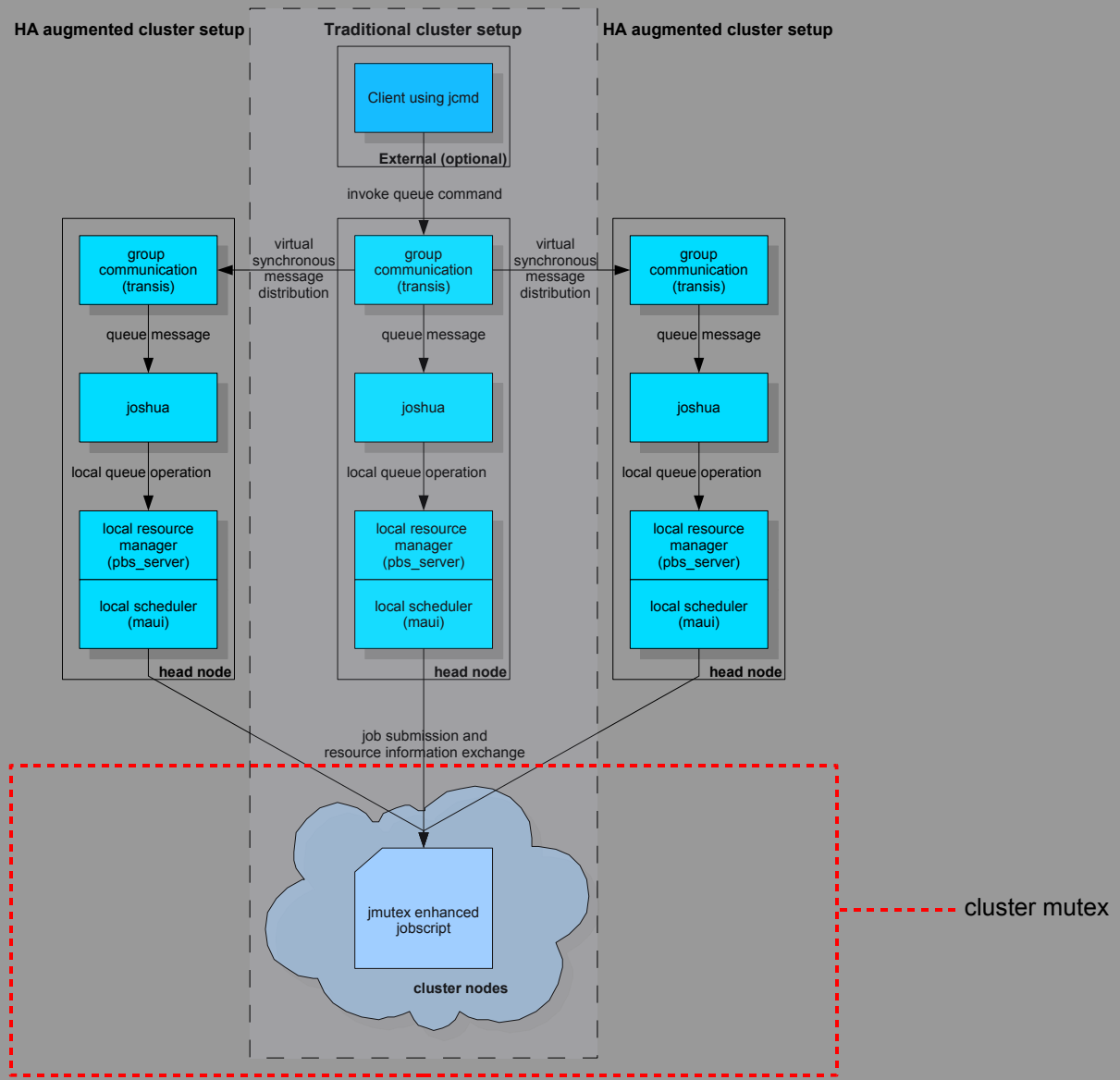
Job scheduler maui

- Commonly used in conjunction with PBS

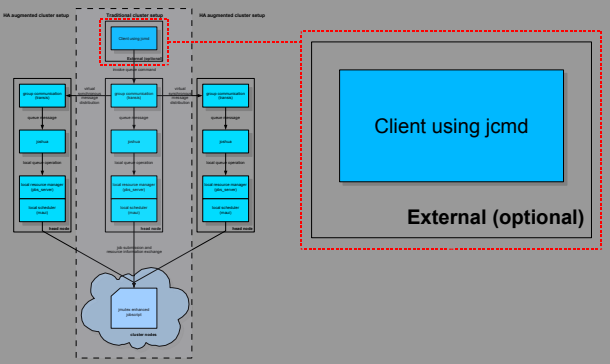

HA augmented cluster setup
Traditional cluster setup
HA augmented cluster setup
Client using jcmd
External (optional)
invoke queue command
virtual synchronous message distribution
virtual synchronous message distribution
group communication (transis)
group communication (transis)
group communication (transis)
queue message
queue message
queue message
joshua
joshua
joshua
local queue operation
local queue operation
local queue operation
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
head node
head node
head node
job submission and resource information exchange
jmutex enhanced jobscript
cluster nodes


HA augmented cluster setup
Traditional cluster setup
HA augmented cluster setup
Client using jcmd
External (optional)
command line tools
invoke queue command
group communication (transis)
virtual synchronous message distribution
group communication (transis)
virtual synchronous message distribution
group communication (transis)
queue message
queue message
queue message
joshua
joshua
joshua
local queue operation
local queue operation
local queue operation
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
head node
head node
head node
job submission and resource information exchange
jmutex enhanced jobscript
cluster nodes


HA augmented cluster setup
Traditional cluster setup
HA augmented cluster setup
Client using jcmd
External (optional)
invoke queue command
group communication (transis)
virtual synchronous message distribution
group communication (transis)
virtual synchronous message distribution
group communication (transis)
queue message
queue message
queue message
joshua
joshua
joshua
local queue operation
local queue operation
local queue operation
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
head node
head node
head node
server application
job submission and resource information exchange
jmutex enhanced jobscript
cluster nodes

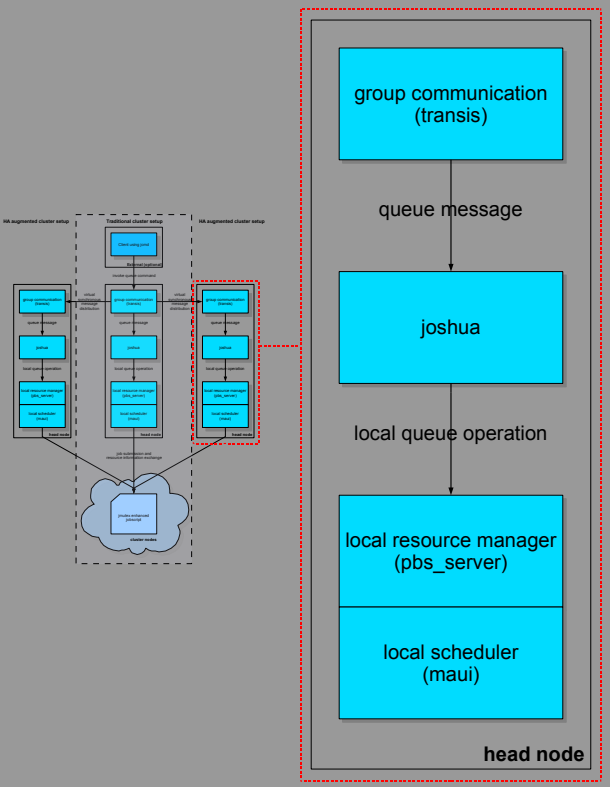

HA augmented cluster setup
Traditional cluster setup
HA augmented cluster setup
Client using jcmd
External (optional)
invoke queue command
virtual synchronous message distribution
virtual synchronous message distribution
group communication (transis)
group communication (transis)
group communication (transis)
queue message
queue message
queue message
joshua
joshua
joshua
local queue operation
local queue operation
local queue operation
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
local resource manager (pbs_server)
local scheduler (maui)
head node
head node
head node
job submission and resource information exchange
jmutex enhanced jobscript
cluster nodes
cluster mutex

Command line tools



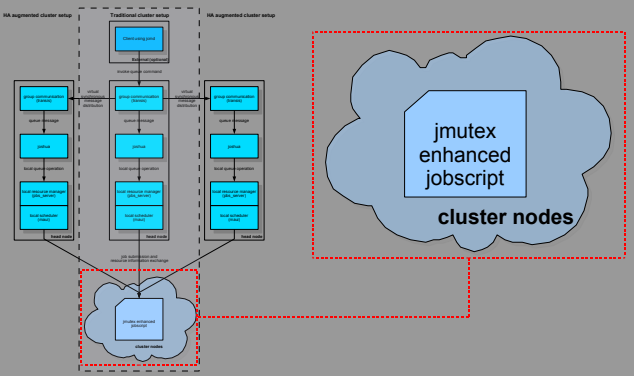
- Perform queue operation such as
  - Job submission
  - Job status information
  - Job removal from the queue
- Designed for high transparency to the user
- Enables simple replacement for former job queue commands
- Achieved by message encapsulation of argument vector, stdin and environment
- Effective command is redirected to the joshua server daemon

Server application

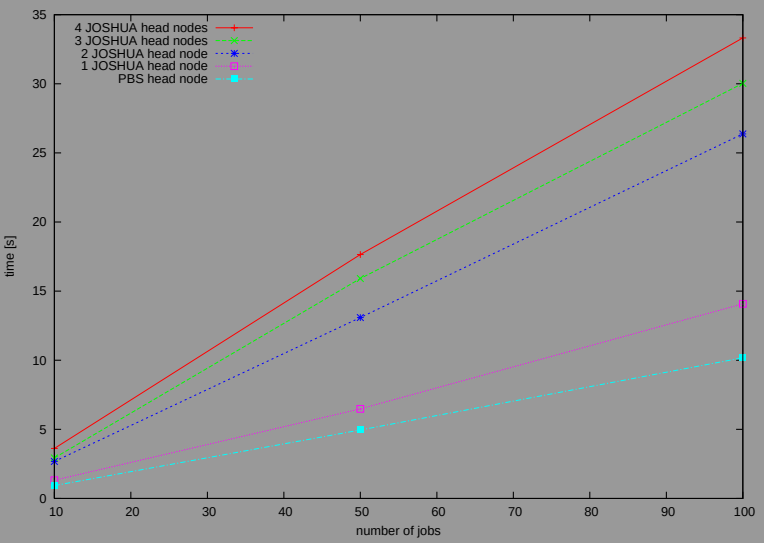


- Daemons act as transparent gateway for user commands
- Queue commands are received globally, executed locally
- Output is redirected to the client
- All joshua daemons run in virtual synchrony provided by transis
- Daemons form a group of head nodes
- Using group communication the group reconfigures itself in cases such as
  - Daemon joins
  - Daemon failure

Cluster mutex



- Distributed mutual exclusion across all cluster nodes
- Realized in connection with virtual synchrony
- Executor sends request for execution
- First gets the permission to run the job
- All other are put on hold
- When job is finished the pending executors will be released
- Solves the single instance problem



- Reasonable cost for HA depends from number of head nodes
- Submission performance in linear scale
- Joshua handles parallel job submission, join and fail events as proposed
- Jmutex handles single instance execution problem
- Transis has some stability issues
- PSB is still focused on client-server model

- Possible merge with HA-OSCAR project
- Integration and development of more reliable group communication system
- Use of more suitable external resource management system
- Add missing job manipulation features (hold, release)

- Take advantage of multi-headed architecture and the symmetric Active/Active HA design
- Prototype implementation could prove design to improve the cluster HA
- System availability is scalable as proposed
- Approach leads to a significant increase of availability of cluster head nodes
- First step toward non-stop computation by introducing a fault tolerant job scheduler

Research Paper

K. Uhlemann, C. Engelmann and S. L. Scott. "'JOSHUA: Active/Active high availability for HPC job and resource management"'. *To appear in proceedings of IEEE International Conference on Cluster Computing*, Barcelona September 25th-27th, 2006.