

Changing the batch system in a Tier 1 computing center: why and how



Andrea Chierici, Stefano Dal Pra
INFN-CNAF

andrea.chierici@cnaif.infn.it, stefano.dalpra@cnaif.infn.it

Introduction

Batch system is a key element of INFN Tier1, allowing flexible and fine grained control on computing resources. Our center adopted LSF since the very beginning, a very robust and reliable solution widely adopted in HEPiX community. Lately new software (free and commercial) gained our attention, thanks to interesting features as well as attractive pricing. INFN has a nation-wide contract in place to use LSF that will expire by 2014: during this time we would like to evaluate alternatives. In this poster we explain our investigation on UNIVA Grid Engine, a solution that has gained consensus in our community.

Another INFN site evaluated SLURM in a very similar way and discussion is ongoing inside INFN on what could be a suitable alternative to be adopted by our computing centers.



Grid Engine



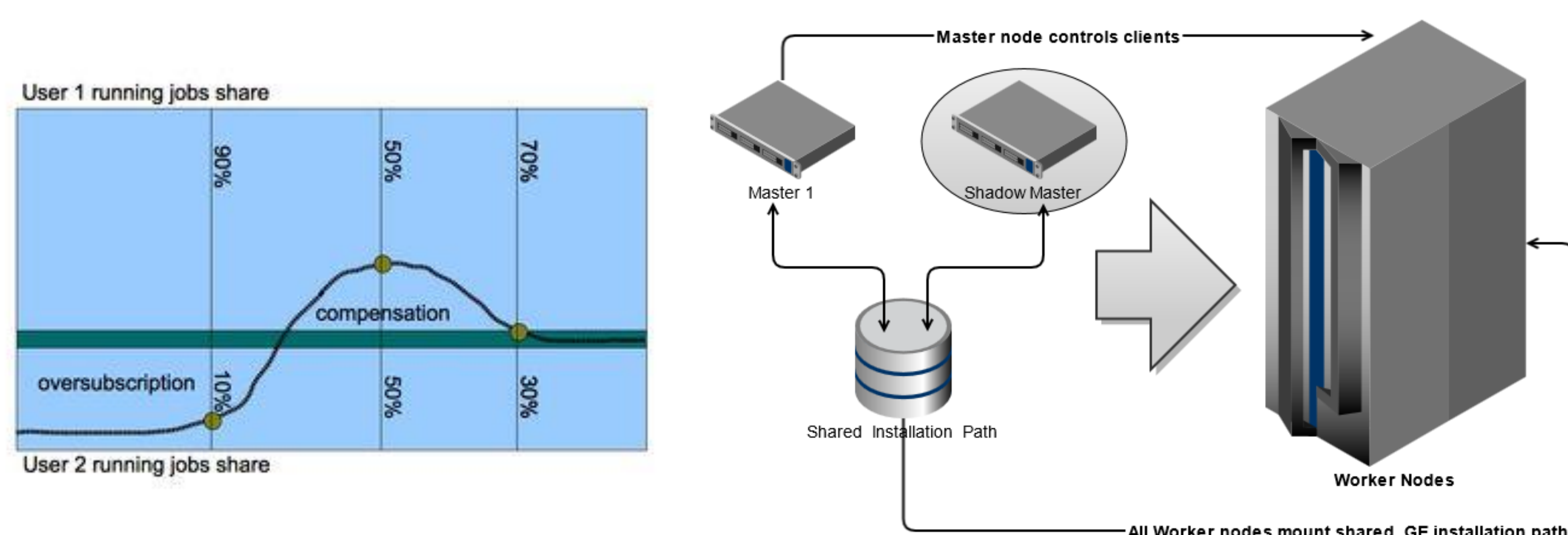
Grid Engine is an open source batch-queuing system, originally developed and supported by Sun Microsystems. Univa Corporation provides a commercial and supported evolution of Grid Engine, which recently has been adopted by many computing centers in HEP community. We evaluated this product in order to understand if it can fit our requirements.

Evaluation Goals

In order to be able to fit our needs the new batch system must be compatible with our use-cases:

- Effective Integration with grid middleware
- Support for hierarchical fair-share
- Support for multicore jobs
- Allow for implementation of priorities and advance reservation
- Current LSF configuration should be translated to GE without loss of features
- Master node must support high availability
 - Quick and safe recover after hardware failure is **not** an option
- Scalability
 - On ~18K cores we currently run 80 to 130 Kjobs/day
 - Jobs are heterogeneous (submission rate, lifetime, CPU intensive or I/O intensive or both, etc.)

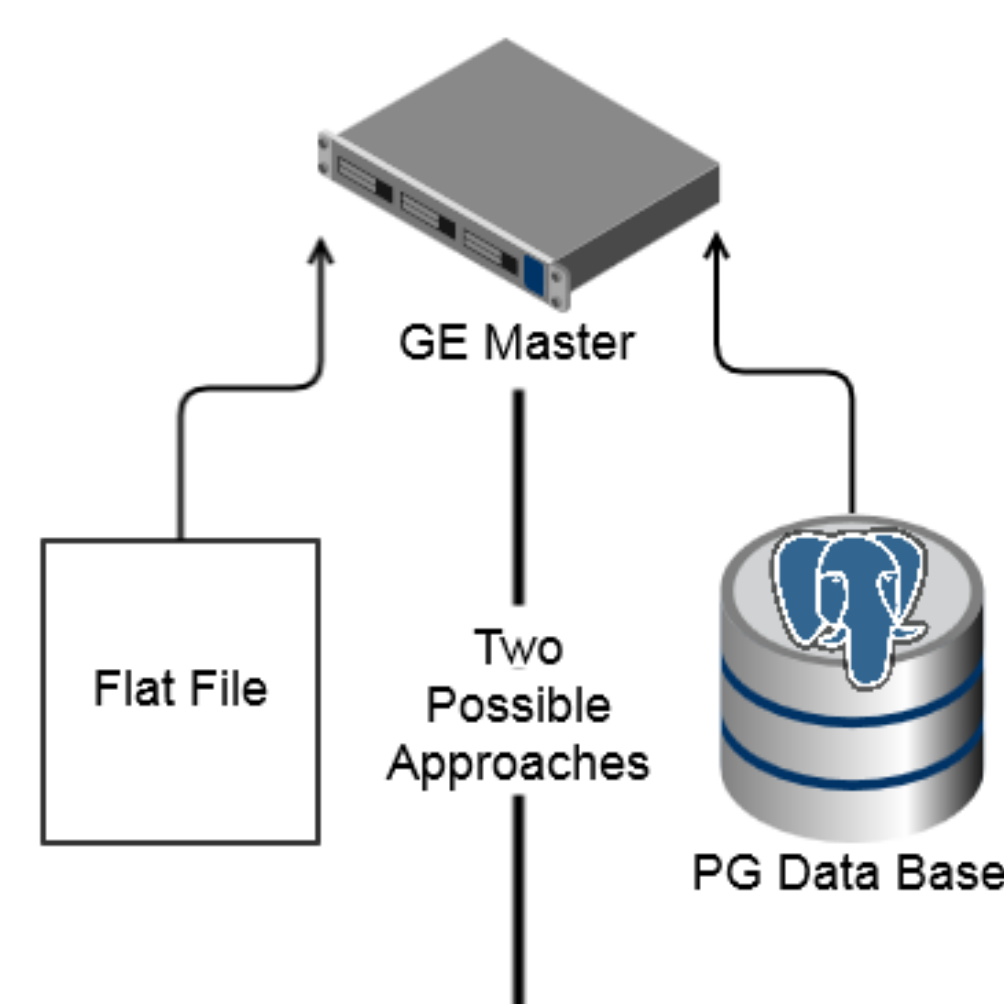
During the tests in our lab we were able to verify all these requirements are met.



Evaluation Timeline

Following is the timeline we adopted for the evaluation of GE:

1. Setup a mini cluster for self-training
2. Get confident with GE admin & maintenance tasks
3. Integrate GE with EMI middleware (cream CE) and run test jobs on it
4. Evaluate different setups for performance and HA:
 - Classic spooling (flat-files on shared directory)
 - DB spooling
 - Master with shadow
5. Evaluate integration with accounting and monitoring:
 - Data collection for our local custom accounting & monitoring system **must** be adapted
 - dgas and apel compatibility
6. Conversion and emulation of real use cases
 - adaptation of LSF "pre_exec" and "post_exec" scripts
7. Test using production farm
 1. Install one Cream CE with GE extensions as an access point from the grid to our farm
 2. Share computing nodes with LSF
 3. Gradually migrate computing resources from LSF to GE. Two ways to achieve this:
 - **By nodes:** a node is entirely managed either by GE or by LSF
 - **By slots:** x slots in a node are assigned to GE and N-x to LSF. Gradually increasing x from 0 to N realizes a complete switch.

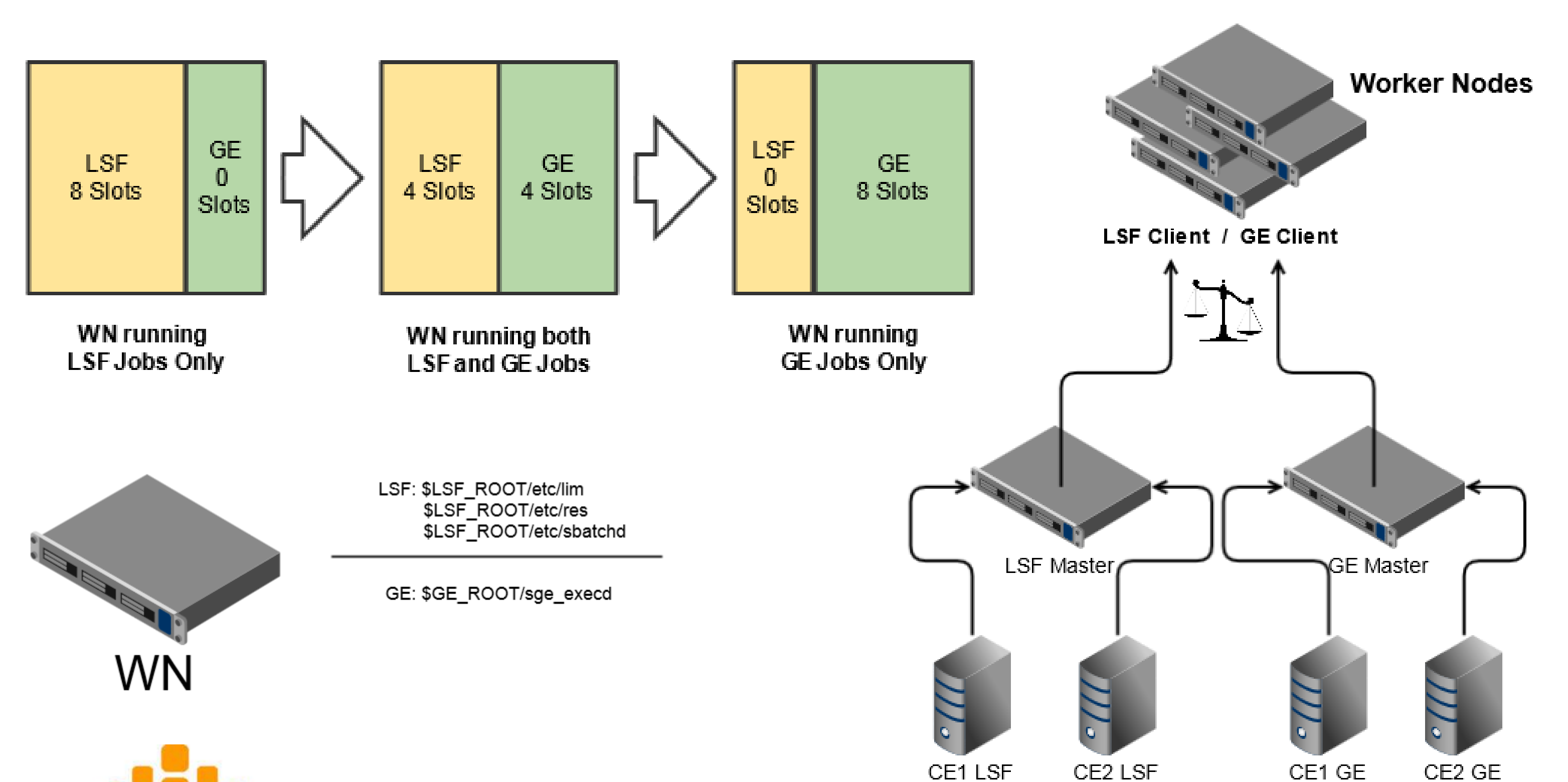


The Switch

Switching from LSF to GE is not really difficult, because both systems have common requirements and do not need many specific interventions on the worker nodes. The switch must be implemented at two different levels: "grid" level and "worker node" level.

At the grid level, we install Cream Computing Elements with GE extensions and point them to our Master Node, that shares the same WNs set. When all the WNs have turned to GE only job execution, it's safe to remove the LSF client and turn the LSF CEs off.

At worker node level, both LSF and GE clients run at the same time on the node and the number of slots is modified in order for GE to take over the LSF slots. An alternative approach is to allocate a subset of WNs on the farm to GE, once all the LSF jobs on them are terminated.

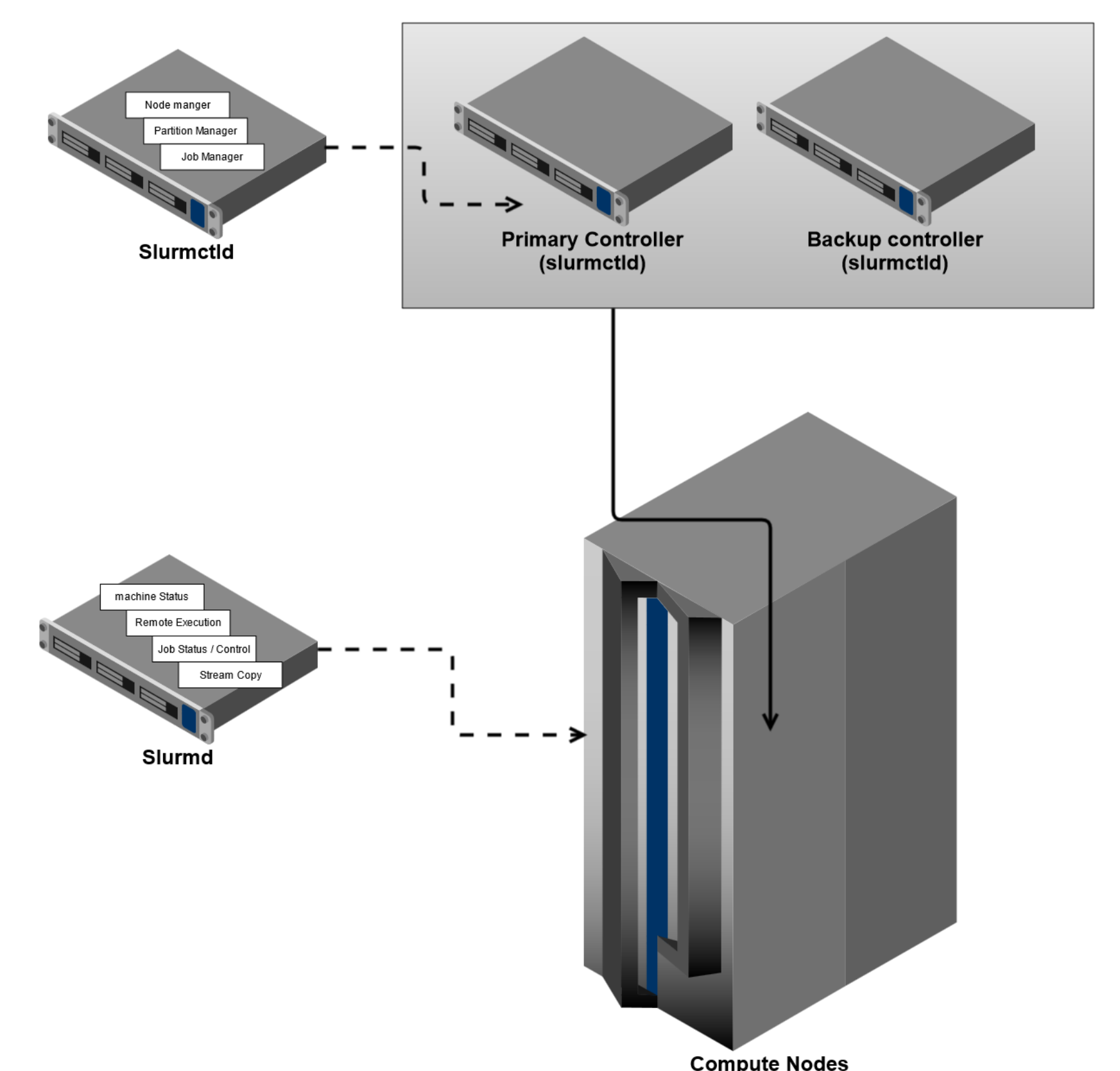


SLURM

INFN Bari tested SLURM, the Simple Linux Utility for Resource Management, a free and open-source job scheduler for the Linux kernel. SLURM implements a fairly traditional cluster management architecture, rather similar to the one we found in LSF and GE.

SLURM passed all the tests that were made, in particular:

- Hierarchical fair-share
- Multicore jobs
- Priorities, resource reservation
- LSF-like Tier1 configuration
- Master node High availability
- Scalability test



Conclusions

Grid Engine (UNIVA flavor) proved to be a very reliable and stable batch system. We verified that it can quite easily replace of our current batch system in our environment, thanks to its scalability and since all the software that rely on it is at production quality level (or can with small effort).

INFN Bari proved SLURM to be a good alternative too, giving us another software to choose from. Our current batch system contract will expire next year and now we know that switching it is possible in a safe way.

References

- HEPIX: <http://www.hepix.org>
 LSF: <http://goo.gl/t1dK13>
 UNIVA Grid Engine: <http://www.univa.com>
 EMI middleware: <http://www.eu-emi.eu>
 SLURM: <http://slurm.net>