Wishlist and Questions for Vendors

Below please find the wishlist and questions the main SLAC batch system users and administrators posed.

Followup Questions

- What kind of support (transition/longterm) do we get for the quoted price?
- How much downtime do we incur if we need to "restart the system"
- What sort of activities require such a "restart" (e.g. creating a new queue?)
- Do large numbers of short jobs (about 1 minute) cause problems?
- Please provide additional information about support for virtualization, and any plans for future enhancements to that support.
- What would be involved in adding support for Mac OSX or Windows? Would that be considered normal "support"?
- Where are the bottlenecks in the system likely to occur? For example, can a user stress the system by repeatedly and frequently querying job stats?

Original Wishlist

User Group 1

- Automatic job preemption/suspend/resume?
- Support for multiple-levels of job preemption (e.g. 3-queue hierarchy)?
- Job environment propagation (including limits like "stacksize")?
- Subgroup-specific priority calculation (queue-specific priority formula)?
- Capability to delegate subgroup administration privileges (adjust job priorities, suspend, resume, kill) to subgroup administrators?
- Cross-queue fairshare (with cpu-speed weighting)?
- CPU advanced reservations for MPI?
- GPU support?
- Ability to submit jobs to hosts where we don't have accounts/home-directories?
- Avoid bad behavior when MPI head node reboots: slave node processes get "forgotten" ?
- How well does the system scale?
    - Number of cores, queues, queued and running jobs?

User Group 2

- Please list supported operating systems (for submission hosts and for execution hosts)
- Explicit, site-specific-naming resource specification (e.g., rhel5-64, amount of /scratch space, etc.) at job submission time?
- Is there an API for "time remaining" query (to save state near end of job)?
- Do submission / management hosts require a license?
    - ability to submit/monitor jobs from any machine at SLAC (not just those with licenses as is the case with LSF)

User Group 3

- Resource arbitration (i.e. ability to easily control number of running jobs to prevent overloading of disks and other resouces, such as memory)
- Reliable notification to job when CPU, other resources, about to be exhausted
- Built-in support for virtualization?
    - Ability to suspend/resume/move running jobs (MPI and single-core)
- API (preferably rest/xml/json) for submitting, monitoring, controlling jobs, in addition to command-line control
- Can resource allocation within a subgroup be delegated to subgroup administrators?
    - e.g. today I want 90% of resources dedicated to one subgroup within my allocation, tomorrow only 10%
- Hooks for adding extensions (e.g. kerberos tickets or afs token support) without introducing maintainance problems

User Group 4

- Knowledge of cluster topology to assign large jobs across nodes in optimized way?
    - For example an MPI job with 256 cores would best be assigned on as few nodes as possible and those nodes should span as few Infiniband/ethernet switches as possible.
- Ability to request exclusive access to nodes: this is to support using the full node's memory on a subset of the nodes cores?
- Ability to schedule using node properties and resources: memory, gpu cores, cpu cores, i/o connectivity, other user defined properties?
- Job priority schemes that support coexistence of small and large jobs?
    - This likely includes issues of pre-emption, backfill, checkpointing, migration and restarting. Also includes aggregation of single core jobs onto nodes with other similar jobs.
- Ability to partition the cluster / support multiple cluster (by host capabilities, such as IB-capable, GPU nodes, other per-node attributes)?
- Ability to automatically enable preemption at certain times of day?

User Group 5

- Is the batch system supported by Open Science Grid?

User Group 6

- Software should be installed via RPM or similar, both in terms of reducing the reliance on a shared file system, and in terms of simplification of system install process
- Per-node configuration should be fairly simple, and easily manageable via a configuration management tool (e.g. chef).
- Servers must support semi-high-availability

  - at least as much as our existing LSF setup (at a minimum 2 redundant batch-manager nodes).
- Can there be multiple simultaneous sources for authorization information that are applied in a well-defined order?
  - e.g. NIS, LDAP, flat files as source where a host-local file can override central LDAP
- Where possible, we want to be able to perform meaningful queries regarding current queue and job states and get pretty-and-useful reports back.
  - This is true both for command-line operations *and* web operations.
- From the Unix side, interaction should occur over well-documented and consistent ports (so that we can handle firewalls properly).

User Group 7

- Integration with AFS (i.e. it must be possible for single-node and MPI jobs to acquire the user's kerberos and AFS identities)?
  - Does the system (at a minimum) provide hooks to implement such a feature?
- Real-time and historical monitoring , accounting and analytics (performance, utilization, resource consumption, etc) available through web interface, command line, other APIs (e.g. for embedding in other web applications).
  - It is fine if this is an add-on capability
- Short list of resource allocation / scheduling schemes (such as fairshare, backfill, etc.) with brief explanations
- Job submission and management through a web portal?
- Duration and limitations of trial licenses?

User Group 8

- How are licenses managed (i.e. is there a requirement for a license server, if yes, does it allow for redundancy)?
- Ability to schedule jobs based on file server load?