Evaluation of Job Queuing/Scheduling Software: Phase 1 Report

James Patton Jones[1]

NAS Technical Report NAS-96-009 July 96

jjones@nas.nasa.gov
NAS High Performance Processing Group
NASA Ames Research Center
Mail Stop 258-6
Moffett Field, CA 94035-1000

Abstract

The recent proliferation of high performance workstations and the increased reliability of parallel systems have illustrated the need for robust job management systems to support parallel applications. To address this issue, NAS compiled a requirements checklist for job queuing/scheduling software [Jon96]. Next, NAS began an evaluation of the leading job management system (JMS) software packages against the checklist. This report describes the three-phase evaluation process, and presents the results of *Phase 1: Capabilities versus Requirements*. We show that JMS support for running parallel applications on clusters of workstations and parallel systems is still insufficient, even in the leading JMSs. However, by ranking each JMS evaluated against the requirements, we provide data that will be useful to other sites in selecting a JMS.

---

1. MRJ, Inc., NASA Contract NAS 2-14303, Moffett Field, CA 94035-1000

1.0 Introduction

The Numerical Aerodynamic Simulation (NAS) supercomputer facility, located at NASA Ames Research Center, has been working for the last few years to bring parallel systems and clusters of workstations into a true production environment. One of the primary difficulties has been identifying a robust job management system (JMS) capable of completely supporting parallel jobs. For a complete discussion of the role and need of a JMS, see [Sap95].

Many JMS software packages exist that cover a wide range of needs, from traditional queuing/batch systems to "load-balancing" and "cycle-stealing" software for workstations. While many exist, few attempt to support parallel jobs. It was to address this deficiency that NAS produced the *NAS Requirements Checklist for Job Queuing/Scheduling Software* [Jon96] (with input from the NASA Cooperative Agreement (CAN) NCC3-413 project members: NAS, NASA Ames, NASA Langley, NASA Lewis, Pratt Whitney, Platform Computing, PBS group; as well as input from Cray Research, Inc. (CRI), and IBM). (For a complete description of the cooperative agreement see [CAN95].) This list of requirements focuses on the needs of a site which runs parallel applications (e.g. message-passing codes) across clusters of workstations and parallel systems. However, the requirements attempt to cover the gamut from clusters of PCs to MPPs and clusters of Crays. The intent was twofold: to provide a baseline set of requirements against which to measure and track various JMSs over time; and to provide direction to JMS vendors as they plan product improvements. Therefore, the requirements list was published separately from this evaluation paper in order to allow vendors the maximum amount of time to address the requirements. A condensed summary of the requirements is reproduced herein; refer to the original document for a complete description of each requirement.

Recently, there have been several excellent comparisons of job queueing/batch software systems, e.g. [Bak95 and Kap94]. The two comparisons cited cover most of the vast array of available JMS products. The NAS evaluation differs from these in two primary ways. First, NAS chose to evaluate only the four leading JMS systems. Second, NAS chose to perform a more in-depth comparison with more than twice the number of criteria as the cited evaluations.

2.0 Evaluation Description

This paper discusses an evaluation of the leading job management systems in order to identify the one(s) that best meet(s) the needs and requirements of NAS. The evaluation will proceed in three phases, as shown in Tables 1 and 2.

After the evaluation plan was written, we identified which JMS software packages to evaluate. Table 3 lists the four packages identified, and the versions selected for evaluation.

TABLE 1. Phases of Comparison

Phase 1	Capabilities versus requirements
Phase 2	Staff and selected user testing
Phase 3	Full deployment, production use

TABLE 2. Steps in Evaluation

Phase 1:

1. Obtain most recent production release (non-beta) of JMS from each vendor (see Table 3 below).
2. Review vendor-supplied documentation for JMS system.
3. Perform pencil-paper comparison of JMS requirements against stated capabilities, assigning "points" according to SCALE (see below).
4. Provide each vendor an opportunity to review and correct any technical errors in the evaluation of their product.
5. Rank all JMS systems against METRIC (see below) of capabilities against requirements.
6. Any JMS falling below MININUM THRESHOLD (see below) will be eliminated from comparison; all remaining will continue to Phase 2.
7. Summarize and publish results.

Phase 2: (for each JMS meeting minimum requirements)

A. For each test platform (see Table 4 below)
   1. Install software in test configuration.
   2. Configure and/or write basic job scheduler.
   3. Verify capabilities claimed in vendor-supplied documentation.
   4. Re-score as necessary.
   5. Configure and/or write complex job scheduler.
   6. Run simulated TEST SUITE (see Section 4 below) against JMS.
   7. Open system for staff testing.
   8. Open system for selected user testing.
   9. Solicit feedback from testing.

B. Test inter-platform JMS capabilities.

C. Summarize and publish results.

D. Optionally perform Phase 3 evaluation at this time.

E. Archive JMS configuration.

F. Deinstall JMS.

TABLE 2. Steps in Evaluation

Phase 3: (Optional) 1. Install software in production configuration. 2. Configure and/or write complete job scheduler with all NAS policies. 3. Produce all necessary documentation and guides to educate users on JMS. 4. Evaluate under normal user workload for several months.
Conclusion: 1. Produce summary report of findings.

.

TABLE 3. JMS Software Selected for Evaluation

JMS	Version	Vendor	Released
LoadLeveler (LL)	v.1.2.1	IBM	Aug 95
Load Sharing Facility (LSF)	v.2.2	Platform	28 Feb 96
Network Queueing Env (NQE)	v.2.0	CRI	31 Mar 95
Portable Batch System (PBS)	v.1.1.5	NASA	18 Jan 96

A general description of each of these products is given in the *Phase 1 Results* section below.

Next, we generated a rough timeline for the evaluation. Table 4 shows the portion of the timeline covered by this paper. (Table 11 in Section 5 below gives the revised timeline for the conclusion of the project.).

TABLE 4. Timeline of JMS Evaluation, Phase 1

Time Period	Activity
1 March 1996:	Cut-off date for vendor release of production software.
1 March - 15 April:	Phase 1 comparison.
15 April -15 May:	Summarize and publish Phase 1 results.

Choosing a cut-off date was necessary to set a fixed window of time for the evaluation. The original proposed date was revised to March 1st in order to include the latest versions of LSF and NQE, both of which were scheduled for a major release at the end of February 1996. Unfortunately, the NQE release 3.0 slipped three months, so the current version 2.0 was evaluated. The next release of LoadLeveler is scheduled for Fall 1996.

We then determined which computer systems would be used for the second phase of the evaluation. The three testbed systems at NAS, listed in Table 5, were selected for the diversity and flexibility they provide. Because they are not true production systems, we have more latitude with regard to software changes and providing staff with dedicated-system time. The three systems differ in their workload and job mix, but all three give priority to supporting parallel and message-passing applications.

TABLE 5. Phase 2 Comparison Platforms

Architecture	NAS Hostname	Configuration
SGI PowerChallenge	davinci	8-node (40 CPU) workstation cluster, 1 front end
CRI J90	newton	4-node (20 CPU) cluster, 1 front end
IBM SP2	babbage	160-node (160 CPU) SP2, 2 front ends

In addition, we determined that the test suite to be used in Phase 2 for evaluating each JMS will consist of a combination of the following:

- A suite of applications including the NAS Parallel Benchmarks (NPBs)
- Jobs or scripts testing particular features of the JMS
- Simulated job stream (based on past job accounting data from the SP2)

The details of the test suite will be determined prior to beginning Phase 2.

While the main focus of Phase 1 was to compare capabilities of the selected products, we also wanted a way to eliminate from Phase 2 any JMS that did not meet a minimum number of our requirements; it would not be worthwhile to perform the level of evaluation required in Phase 2 on products that did not meet enough of our needs.

Since the list of requirements was divided into three main categories: absolute requirements, recommended capabilities, and future requirements, we decided to use the absolute requirements (those listed in the requirements checklist in section 3 below) for the elimination metric. Each of those requirements was further ranked as high or medium priority. From this we generated the following simple metric, a percentage index for the number of section 3 criteria met, taking the priority into consideration:

[sum (“score” * “priority”)] / max possible * 100

We next determined what the “minimum threshold” would be: any JMS ranking below 90 percent on the above metric will be eliminated from the Phase 2 comparison as not meeting enough of the base requirements.With these details decided, we proceeded with the Phase 1 evaluation.

The following section gives an abbreviated list of the requirements used in the evaluation. Again, we suggest a review of the evaluation data with a copy of the complete requirements.

3.0 Condensed Requirements List

Job Management System

High Priority

3.1.1 Must operate in a heterogeneous multi-computer environment...
3.1.2 Must integrate with frequently used distributed file systems...
3.1.3 Must possess a command line interface to all modules of the JMS...
3.1.4 Must include a published application programming interface (API) to every component of the JMS...
3.1.5 Must be able to enforce resource allocations and limits...
3.1.6 Software must permit multiple versions on same system...
3.1.7 Source code must be available for complete JMS...
3.1.8 Must bee able to define more than one user id as JMS administrator...

Medium Priority

3.1.9 Must provide a means of user identification outside the password file...
3.1.10 Must be scalable...
3.1.11 Must meet all requirements of appropriate standards...

Resource Manager Requirements

High Priority

3.2.1 Must be “parallel aware,” i.e. understand the concept of a parallel job and maintain complete control over that job...
3.2.2 Must be able to support and interact with MPI, PVM, HPF...
3.2.3 Must provide file “stage-in” and “stage-out” capabilities...
3.2.4 Must provide user-level checkpointing/restart...

Medium Priority

3.2.5 Must provide a history log of all jobs...
3.2.6 Must provide asynchronous communication between application and Job Manager via a published API...
3.2.7 Must be integrated with authentication/security system...
3.2.8 Interactive-batch jobs must run with standard input, output, and error file streams connected to a terminal...

Scheduler Requirements

High Priority

3.3.1 Must be highly configurable...
3.3.2 Must provide simple, out-of-the-box scheduling policies...
3.3.3 Must schedule multiple resources simultaneously...
3.3.4 Must be able to change the priority, privileges, run order, and resource limits of all jobs, regardless of the job state...
3.3.5 Must provide coordinated scheduling...

Medium Priority

3.3.6 Must provide mechanism to implement any arbitrary policy...
3.3.7 Must support unsynchronized timesharing of jobs...
3.3.8 Sites need to be able to define specifics on time-sharing...

Queuing System Requirements

High Priority

3.4.1 Must support both interactive and batch jobs with a common set of commands...
3.4.2 User Interface must provide specific information...
3.4.3 Must provide for restricting access to the batch system using a variety of site-configurable methods...
3.4.4 Must be able to sustain hardware or system failure...
3.4.5 Must be able to configure and manage one or more queues...
3.4.6 Administrator must be able to create, delete, and modify resources and resource types...
3.4.7 Administrator must be able to change a job's state...
3.4.8 Must allow dynamic system reconfiguration by administrator with minimal impact on running jobs...
3.4.9 Must provide centralized administration...
3.4.10 Users must be able to reliably kill their own job... See 3.2.1 above.

Medium Priority

3.4.11 Must provide administrator-configurable programs to be run by JMS before and after a job...
3.4.12 Must include user specifiable job interdependency...
3.4.13 Must allow jobs to be submitted from one cluster and run on another...
3.4.14 Must provide a site-configurable mechanism...to permit users to have access to information about jobs from other submitters...

Requested Capabilities

High Priority

4.1.1 Job scheduler should support dynamic policy changes...
4.1.2 Possess a Graphical User Interface (GUI) to JMS...
4.1.3 Provide a graphical representation of the configuration and usage of the resources under the JMS...

Medium Priority

4.1.4 The time-sharing configuration information should be available to the job scheduler for optimizing job scheduling...
4.1.5 Provide a graphical monitoring tool with the specified capabilities...
4.1.6 Support both hard and soft limits when appropriate...
4.1.7 Should be readily available with full, complete support...
4.1.8 Should supply some kind of a proxy account optional setup...
4.1.9 Should provide specified accounting capabilities...

Low Priority

4.1.10 Should allow a site to choose to run separate resource managers for each system (or cluster), as well as a single resource manager for all systems...
4.1.11 Should allow owner of interactive jobs to “detach” from the job...
4.1.12 Should provide a mechanism to allow reservations of any resource...
4.1.13 Should provide specific attributes for jobs...
4.1.14 Should be able to define and modify a separate access control list for each supported resource....
4.1.15 Should provide wide area network support...
4.1.16 Should allow an interactive user on a workstation console to instruct the JMS to suspend or migrate a job to a different workstation...
4.1.17 Should provide both client and server capabilities for Windows NT...

Future Requirements

High Priority

5.1.1 Should provide gang-scheduling...
5.1.2 Should provide dynamic load balancing...
5.1.3 Should provide job migration...

Medium Priority

5.1.4 Should inter-operate with OS level checkpointing, providing the ability for the JMS to restart a job from where it left off and not simply from the beginning....

4.0 Phase I Results

The results of *Phase 1: Capabilities versus Requirements* for the products evaluated are provided below. A description of each product is provided followed by its evaluation. As indicated in Table 2 above, each vendor was given the opportunity to review and correct any technical inaccuracies in the evaluation of their product. It should be noted that CRI did not accept this opportunity.

Table 6 lists the definitions of "scores" for each requirement. Note that instead of performing a "yes/no" or "has/has not" comparison, we attempt to determine how much of each requirement the JMS meets. The result for each requirement is presented in a single "score" accompanied by a short explanatory note. The notes are not intended to replace the description of the requirements. A copy of *NAS Requirements Checklist for Job Queuing/Scheduling Software* [Jon96] is required to interpret the evaluation data.

Table 6: Score Definitions

Score	Explanation
●	Meets requirement
◕	Meets most of requirement
◑	Meets roughly half of requirement
◔	Meets little of requirement
○	Does not meet any of requirement

4.1 LoadLeveler (LL)

Loadleveler, from IBM, is a commercially available, general-purpose JMS software package. Emphasis is currently on clusters of workstations running single serial jobs. Some support for parallel jobs is provided, but is limited to SP systems where the Parallel Operating Environment (POE) is available. Extensive support for parallel jobs (include non-SPs) is scheduled for the Fall 1996 release. Information for this evaluation is based on [IBM95a, IBM95b]. Additional information is online: (http://spud-web.tc.cornell.edu/hn/frame/LL.html).

Table 7: Loadleveler 1.2.1

Requirement	Score	Notes
3.1.1	◑	SP2, RS/6000, SUN, SGI, HP; no support for CRI UNICOS (one of the evaluation platforms)

Table 7: Loadleveler 1.2.1

Requirement	Score	Notes
3.1.2	◑	NFS and AFS only; DFS/DCE due 1Q97
3.1.3	●	has command line interface
3.1.4	◑	API for accounting, prologue, epilogue, checkpoint (serial), submit, monitor; scheduler API due 3Q96
3.1.5	◕	not provided: wall-clock time (due 3Q96) provides per-process, not per-job: memory utilization; swap, dedicate/shared access
3.1.6	●	via different port numbers and file tree
3.1.7	●	source-code available for a price
3.1.8	●	multiple managers, no operators
3.1.9	◔	insufficient user identification mechanisms
3.1.10	●	in use at Cornell: 512 nodes; another site: 800+ nodes
3.1.11	❍	does not meet POSIX 1003.2d, “Batch Queueing Extensions” standard
3.2.1	◔	does not track all subprocesses, forward signals, or provide job-JMS communication for job-start accounting is questionable; tracks parent-wait3-child processes only
3.2.2	◔	“supports” but does not interact with MPI, PVM, HPF
3.2.3	◑	suggests use of prologue/epilogue to copy files, but no automatic file staging as required
3.2.4	◑	system-level check-point/restart where supported by OS; JMS assisted user-level checkpointing for serial jobs only
3.2.5	◕	combination of UNIX accounting data and LL generated data (no suspended execution data)
3.2.6	❍	application-JMS communication not available
3.2.7	◔	UNIX-level security only; DCE support in 1Q97
3.2.8	❍	does not support batch-scheduled interactive jobs

Table 7: Loadleveler 1.2.1

Requirement	Score	Notes
3.3.1	◔	does not support dynamic & pre-emptive resource allocation; only distinguishes batch and interactive jobs
3.3.2	◕	capable of all except “fair-share”; need to be configured before use
3.3.3	◕	scheduler supports all listed, except supports only one file-system (execution directory)
3.3.4	◕	cannot change running jobs
3.3.5	●	supports space-sharing
3.3.6	❍	scheduler not separable from JMS; no API for scheduler (due 3Q96)
3.3.7	●	supports unsynchronized timesharing
3.3.8	●	via local configuration in MACHINE stanza
3.4.1	●	handles both interactive and batch
3.4.2	◕	does not provide resources consumed for running jobs or for subprocesses of parallel jobs; no status of system resources
3.4.3	●	provided
3.4.4	◕	jobs (except interactive) are automatically requeued/resumed/rerun in event of system failure.
3.4.5	●	provided
3.4.6	●	provided
3.4.7	●	provided
3.4.8	●	can add/delete nodes; can request each daemon re-read its configuration files
3.4.9	◕	commands are centralized, log and accounting files are distributed, but tools are provided to combine remote logs into single log
3.4.10	❍	if subprocesses of parallel jobs are not controlled, then JMS cannot guarantee to kill processes

Table 7: Loadleveler 1.2.1

Requirement	Score	Notes
3.4.11	●	provided
3.4.12	◔	job dependencies limited to “job-steps” (steps/statements within a job) rather than “jobs”
3.4.13	●	provided
3.4.14	●	provided
4.1.1	●	allows reconfiguration of JMS scheduler without affecting rest of JMS
4.1.2	●	has GUI “to all functions” (LL. Summary p.4)
4.1.3	❍	no graphical system configuration tool
4.1.4	❍	no MACHINE stanza for this (due ‘97)
4.1.5	❍	no graphical monitoring tool (suggests using separate product, “Performance Toolbox/6000”)
4.1.6	◑	supports hard limits (wall-clock); allows user-specified simple soft limit; limits do not take into consideration multi-node parallel jobs; focused on “job steps”
4.1.7	●	supported by large software company
4.1.8	●	via USERS stanza
4.1.9	◑	JMS accounting provides some of the data and some tools to process it
4.1.10	●	provided
4.1.11	❍	cannot detach/reattach; plus no concept of “interactive-batch”
4.1.12	❍	no resource reservations
4.1.13	◑	doesn’t accurately track all parallel job resource consumption or limits
4.1.14	◔	ACL only for selected resources (e.g. hosts)
4.1.15	●	distance not an issue as long as network is stable and reliable
4.1.16	❍	no workstation owner-JMS interaction

Table 7: Loadleveler 1.2.1

Requirement	Score	Notes
4.1.17	○	no Windows NT support
5.1.1	○	no gang-scheduling
5.1.2	○	no dynamic load-balancing
5.1.3	◔	only for serial jobs
5.1.4	◔	only for serial jobs

4.2 Load Sharing Facility (LSF)

LSF, the Load Sharing Facility, from Platform Computing Corporation., is a commercially available, general-purpose JMS software package. Emphasis is on providing a single package for all needs, but focuses on load balancing and "cycle-stealing". Only limited parallel job support is provided. Extensive support for parallel jobs is due in a late 1996 release. Information for this evaluation is based on [Pla96a, Pla96b, Pla96c]. Additional information is available online: (http://www.platform.com).

Table 8: LSF 2.2

Requirement	Score	Notes
3.1.1	●	Currently: ConvexOS, UNICOS, OSF/1, HP-UX, AIX, Linux, NEC EWS OS, Solaris, SunOS, Sony NEWS
3.1.2	●	provided
3.1.3	●	commands well documented
3.1.4	◕	general API provided (not for scheduler)
3.1.5	◑	no support for disk usage, swap, network
3.1.6	●	via different port numbers
3.1.7	●	available on specific-case basis
3.1.8	●	provides primary administration, and queue-level administration
3.1.9	●	provides site-configurable authentication on per-queue level

Table 8: LSF 2.2

Requirement	Score	Notes
3.1.10	◕	claims scalability to above 200 hosts
3.1.11	○	does not meet POSIX 1003.2d “Batch Queueing Extensions” standard
3.2.1	◔	aware of needs, but all tools directed at sequential, serial jobs
3.2.2	◔	supports, but does not interact
3.2.3	◕	users can do file-staging via user-level pre-execution capability; includes tests for check/requeue
3.2.4	◑	system-level check-point/restart where supported by OS; JMS-assisted, user-level checkpointing for serial jobs only
3.2.5	◕	meets all except those listed in 3.1.5 above
3.2.6	○	no published job-JMS API
3.2.7	◑	has some DCE support; site configurable
3.2.8	○	no support for batch-scheduled interactive sessions
3.3.1	◑	not highly configurable (must use provided scheduling algorithms); no concept of interactive-batch
3.3.2	◑	has many of those listed
3.3.3	●	can configure via HOST stanza
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	supports space-sharing
3.3.6	○	scheduler not separable; no scheduler API
3.3.7	●	provided
3.3.8	●	via job limits per host
3.4.1	◔	handles both, but does not provide common command set
3.4.2	◕	no remaining resource tracking

Table 8: LSF 2.2

Requirement	Score	Notes
3.4.3	●	provided
3.4.4	◕	jobs (except interactive jobs) are automatically requeued/resumed/rerun in event of system failure
3.4.5	●	provided
3.4.6	●	provided
3.4.7	●	provided
3.4.8	●	provided
3.4.9	●	administration and logs can be centralized (via shared filesystem)
3.4.10	❍	does not have full parallel awareness, therefore cannot “reliably kill” job subprocesses
3.4.11	●	provided
3.4.12	◕	meets all “status of other computer system”
3.4.13	●	provided
3.4.14	❍	not configurable; default is “all users can see all other users jobs”
4.1.1	●	allows reconfiguration of JMS scheduler without affecting rest of JMS
4.1.2	●	GUI for all modules
4.1.3	◕	one window per cluster
4.1.4	●	via HOSTS stanza
4.1.5	◕	captures snapshot via external program such as xv
4.1.6	◑	supports hard limits only
4.1.7	◕	very popular package for cycle stealing and load balancing
4.1.8	●	Create shared account(s) for LSF jobs to run under, restrict access via configuration file

Table 8: LSF 2.2

Requirement	Score	Notes
4.1.9	◑	JMS provides some requested data in ascii format, and simple tool to process records
4.1.10	◔	cannot schedule multiple “clusters” with single server; vendor suggests putting all machines to be scheduled into single “cluster”
4.1.11	❍	cannot detach/reattach; plus no concept of “interactive-batch”
4.1.12	❍	no resource reservation
4.1.13	◑	no resource consumption counters
4.1.14	◔	controls access to JMS, specific hosts, classes of hosts, and queues only
4.1.15	●	distance not an issue as long as network is stable and reliable
4.1.16	◔	only indirectly; if load on system goes up, JMS may reallocate resources
4.1.17	❍	no Windows NT support
5.1.1	❍	no gang-scheduling
5.1.2	❍	no dynamic load-balancing
5.1.3	◑	provides only for serial jobs where supported by OS
5.1.4	●	provided

4.3 Network Queueing Environment (NQE)

NQE, the Network Queueing Environment, from the CraySoft division of Cray Research Inc., is a commercially available, general-purpose JMS software package. Emphasis is currently on JMS support of large CRI machines, but also provides batch queueing for clusters of workstations running single serial jobs. Initial support for parallel jobs arrived with July 1996 release, too late to be included in this evaluation. Information for this evaluation is based on [Cra95a, Cra95b, Cra95c]. Additional information on the latest release is available online: (http://www.cray.com/PUBLIC/product-info/sw/nqe/nqe30.html).

Table 9: NQE 2.0

Requirement	Score	Notes
3.1.1	●	Solaris, SunOS, IRIX, AIX, HP-UX, DEC OSF/1, UNICOS
3.1.2	◔	NFS support only
3.1.3	●	has command-line interface
3.1.4	◕	API to “all” components
3.1.5	◑	supports: number CPUs, CPU time, memory, disk
3.1.6	●	via different port numbers
3.1.7	●	source code available for a negotiable price
3.1.8	●	provided
3.1.9	●	provided
3.1.10	◔	no explanation of extent of scalability
3.1.11	○	does not meet POSIX 1003.2d, “Batch Queueing Extensions” standard
3.2.1	○	due in v.3.0 (July 96)
3.2.2	◔	supports PVM
3.2.3	◕	provides a “file-transfer agent” to move data from system to system, with fault tolerance
3.2.4	◔	system-level checkpoint/restart where supported by OS; no JMS-assisted user-level checkpointing
3.2.5	◔	very limited accounting logs, appears to rely on UNIX accounting for most data
3.2.6	○	no application-JMS communication available
3.2.7	◔	no indication of AFS/DFS/DCE support
3.2.8	○	no concept of “interactive-batch”

Table 9: NQE 2.0

Requirement	Score	Notes
3.3.1	◔	doesn't support dynamic & preemptive resource allocation; only distinguishes batch and interactive jobs
3.3.2	◑	limited
3.3.3	◑	scheduler (and underlying NQS) can support some listed
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	supports space-sharing
3.3.6	❍	scheduler not separable from JMS; no API for scheduler - due 3Q96
3.3.7	●	supports unsynchronized time-sharing
3.3.8	◔	limited
3.4.1	●	handles both interactive and batch jobs
3.4.2	◑	does not provide the following: why not running, consumed/ remaining resources, allocated/requested resources, state of all
3.4.3	◕	not all restrictions
3.4.4	●	provided
3.4.5	●	provided
3.4.6	◕	limited
3.4.7	◑	only before job is started
3.4.8	◑	limited
3.4.9	◑	limited
3.4.10	❍	no parallel awareness
3.4.11	❍	no prologue/epilogue support
3.4.12	◑	no status of other computer systems

Table 9: NQE 2.0

Requirement	Score	Notes
3.4.13	●	access restrictions apply
3.4.14	◑	all or nothing configurable
4.1.1	◑	limited
4.1.2	●	motif/X and WWW
4.1.3	❍	no graphical system configuration tool
4.1.4	❍	none
4.1.5	❍	no graphical monitoring tool
4.1.6	◑	hard limit: yes; soft limit: no
4.1.7	◕	based on NQS—old *de facto* standard
4.1.8	◕	via shared account and ACLs
4.1.9	◑	much of necessary data provided, no tools to process data however
4.1.10	◕	limited
4.1.11	❍	cannot detach/reattach; plus no concept of “interactive-batch”
4.1.12	◑	has SRFS support, but no other
4.1.13	◑	no computation counters
4.1.14	❍	no ACLs
4.1.15	●	distance not an issue as long as network is stable and reliable
4.1.16	❍	no workstation owner-JMS interaction
4.1.17	❍	no Windows NT support
5.1.1	❍	no gang-scheduling
5.1.2	❍	no dynamic load-balancing
5.1.3	❍	no job migration support

Table 9: NQE 2.0

Requirement	Score	Notes
5.1.4	●	where supported by OS

4.4 Portable Batch System (PBS)

PBS, the Portable Batch System, developed and maintained by the NAS Facility at NASA Ames Research Center, is a freely available, general-purpose JMS software package. Emphasis is on providing a single package for all needs, but focuses on support for high-performance computing (e.g. supercomputers and clusters of workstations). Extensive support for parallel jobs is due in a September 1996 release, with support for dynamic resource management due in January 1997 release. Information for this evaluation is based on [Hen96a, Hen96b]. Additional information is available online: (http://www.nas.nasa.gov/NAS/PBS).

Table 10: PBS 1.1.5

Requirement	Score	Notes
3.1.1	●	Currently: IRIX, AIX, UNICOS, SunOS, Solaris, CM5, SP2, CRAY C90, J90
3.1.2	◔	NFS support only; DCE/DFS support (due 4Q96)
3.1.3	●	commands well documented and explained
3.1.4	●	API well-documented and explained
3.1.5	●	network adapter access enforcement only if OS makes it observable
3.1.6	●	implemented via different port numbers and directories
3.1.7	●	source freely available
3.1.8	●	provides both manager and operator IDs
3.1.9	●	provides ACL in addition to /etc/passwd; could use a single generic account and control all user access via ACLs
3.1.10	●	in production use on a 160-node SP2 at NAS
3.1.11	●	provided

Table 10: PBS 1.1.5

Requirement	Score	Notes
3.2.1	❍	capability will be included in “full parallel awareness” (due 4Q96)
3.2.2	◑	“supports” but does not “interact”; capability will be included in “dynamic parallel awareness” (due 1Q97)
3.2.3	●	provided
3.2.4	◑	system-level checkpoint/restart where supported by OS; no JMS assisted user-level checkpointing; will be included in “full parallel awareness” (due 4Q96)
3.2.5	◕	meets all except a couple of the resources specified in 3.1.5 expect complete resource accounting; with “full parallel awareness” (due 4Q96)
3.2.6	❍	capability will be included in “dynamic parallel awareness” (due 1Q97)
3.2.7	◔	UNIX-level security only; DCE support (due 4Q96)
3.2.8	●	provided
3.3.1	●	administrator must write scheduler specific to site, or use/modify one provided
3.3.2	◑	several complex schedulers included, but not all listed
3.3.3	●	scheduler can support all listed
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	supports space-sharing
3.3.6	●	scheduler can be written in tcl, C, or PBS scripting language
3.3.7	●	provided
3.3.8	●	via PBS nodefile
3.4.1	●	“qsub -I” indicated interactive, all other options are the same as for batch jobs
3.4.2	◕	meets all except CPU consumption of subprocesses of parallel jobs not currently provided; (due with “full parallel awareness” 4Q96)

Table 10: PBS 1.1.5

Requirement	Score	Notes
3.4.3	●	provided
3.4.4	◕	jobs (except interactive jobs) are automatically requeued/resumed/rerun in event of system failure
3.4.5	●	provided
3.4.6	●	provided
3.4.7	●	provided
3.4.8	◑	can add/delete nodes from defined pool; cannot redefine pool without JMS stop/restart
3.4.9	●	all logs are located on server host
3.4.10	❍	capability will be included in “full parallel awareness” (due 4Q96)
3.4.11	●	provided
3.4.12	◕	meets all except “status of other computer systems”
3.4.13	●	provided
3.4.14	●	provided
4.1.1	●	provided
4.1.2	❍	user and operator GUI due 4Q96
4.1.3	❍	no graphical system configuration tool
4.1.4	●	via PBS nodefile
4.1.5	❍	no graphical monitoring tool
4.1.6	◑	supports hard limits only
4.1.7	◑	public domain
4.1.8	●	create shared account(s) for PBS jobs to run under, and restrict access via ACLs
4.1.9	◑	JMS accounting provides much of the necessary data, but no tools to process the data
4.1.10	●	provided

Table 10: PBS 1.1.5

Requirement	Score	Notes
4.1.11	❍	cannot detach/reattach
4.1.12	●	via scheduler; currently doing node reservation on SP2, and disk reservation via SRFS on C90
4.1.13	●	provided
4.1.14	●	server provides ACLs for restricting/allowing access to PBS; scheduler can provide ACLs for any other resources
4.1.15	●	distance not an issue as long as network is stable and reliable
4.1.16	❍	no workstation-owner interaction
4.1.17	❍	no Windows NT support
5.1.1	❍	no gang-scheduling support
5.1.2	❍	first part will be “full parallel awareness” (due 4Q96)
5.1.3	❍	first part will be “full parallel awareness” (due 4Q96)
5.1.4	●	where supported by OS (e.g. UNICOS)

5.0 Conclusions

Now that the first phase of the evaluation is complete, we feel the information and data contained in this report will prove useful to both JMS customers and vendors.

The method of the evaluation proved successful, as did allowing each vendor to review the evaluation results of their product for technical accuracy. The documentation review illustrated to at least one vender that their documentation needed serious attention before the next release. This will benefit existing and future customers alike.

In analyzing the data collected from the evaluation, we found that none of the leading JMS packages yet meet enough of our requirements. Both from the evaluation experience and from actually applying the metric described in section 2 we found that none of the JMSs evaluated meet our minimum number of criteria threshold. In fact, if we were to drop the threshold from 90 percent to 80 percent,

only one JMS would meet the criteria. The four JMS were ranked, highest to lowest: PBS, LSF, LL, and NQE.

Note that this threshold metric was intended only to eliminate less capable JMSs from the Phase 2 evaluation. We needed a metric to draw a line between “pass” and “fail”. It should not be used as an overall comparison of the products, because not all sites have the same needs. Site who use this data are encouraged to select only the criteria important to them, in order to better understand how each product compares against their needs.

While the bad news is the confirmation of a continuing lack of JMS support for parallel applications, parallel systems, and clusters of workstations, the good news is that this year will be an interesting one for JMS functionality. All the major players will be releasing JMS versions with some amount of parallel support by the end of 1996. It is anticipated that by late fall 1996 all four products evaluated will have responded to this evaluation with increased support for parallel applications—even beyond what they have currently planned.

However, due to the current lack of capability across the market, we have decided to postpone Phase 2 of the evaluation until the products are more mature. When we feel the market has matured sufficiently, we will perform the Phase 1 evaluation again, and then continue through the complete evaluation as described in Table 2 above. Assuming the product release schedules announced by the various vendors hold firm, Table 11 shows the revised timeline.

.

TABLE 11. Revised Timeline of JMS Evaluation

Time Period	Activity
1 Sept - 1 Oct	Repeat Phase 1 comparison
1 Oct - 1 Nov	Summarize and publish Phase 1 results
1 Nov - 31 Dec	Phase 2 comparison
1 Jan - 15 Jan	Summarize and publish Phase 2 results
15 Jan - 31 May	Optional Phase 3 comparison; assumes two month evaluation of each product selected for Phase 3

The entire evaluation process is expected to be repeated until the market successfully produces a product that meets the needs of sites around the world.

6.0 Acknowledgments

The requirements used in this evaluation were the result of many discussions between the NAS Parallel Systems group members and organizations participating in the NASA Lewis CAN.

The evaluation itself was made more complete and accurate through the participation of the JMS development teams of each vendor involved.

7.0 References

[Bak95] "Cluster Computing Review," Mark A. Baker, Geoffrey C. Fox, and Hon W. Yau, Northeast Parallel Architectures Center, Syracuse University, November 1995.

[CAN95] NASA Cooperative Agreement NCC-413.
URL: http://www.lerc.nasa.gov/Other_Groups/NPSS/html/can95.html

[Cra95a] "Introducing NQE," CraySoft, Cray Research Inc., Document Number IN-2153 2.0, 1995.

[Cra95b] "NQE Administration," CraySoft, Cray Research Inc., Document Number IN-2150 2.0, 1995.

[Cra95c] "NQE User's Guide," CraySoft, Cray Research Inc., Document Number IN-2148 2.0, 1995.

[Hen95] "Portable Batch System: Requirements Specification", Robert Henderson and Dave Tweten, NAS, NASA Ames Research Center, April 1995.

[IBM95a] "IBM Loadleveler Administration Guide, Release 2.1," IBM, Document Number SH26-7220-03, August 1995.

[IBM95b] "IBM Loadleveler User's Guide, Release 2.1," IBM, Document Number SH26-7226-03, August 1995.

[Jon96] "NAS Requirements Checklist for Job Queuing/Scheduling Software," James Patton Jones, NAS Technical Report NAS-96-003, NAS, NASA Ames Research Center, April 1996.

[Kap94] "A Comparison of Queueing, Cluster and Distributed Computing Systems," Joseph A. Kaplan and Michael L. Nelson, NASA Langley Research Center, June 1994.

[Pla96a] "LSF Administrator's Guide," Platform Computing, February 1996.

[Pla96b] "LSF User's Guide," Platform Computing, February 1996.

[Pla96c] "LSF Programmer's Guide," Platform Computing, February 1996.

[Sap95] "Job Management Requirements for NAS Parallel Systems and Clusters", William Saphir, Leigh Ann Tanner, and Bernard Traversat, NAS Technical Report NAS-95-006, NAS, NASA Ames Research Center, February 1995.



NAS TECHNICAL REPORT

Title: Evaluation of Job Queuing/ Scheduling Software: Phase 1 Report

Author(s): James Patton Jones

Reviewers:
"I have carefully and thoroughly reviewed this technical report. I have worked with the author(s) to ensure clarity of presentation and technical accuracy. I take personal responsibility for the quality of this document."

Two reviewers must sign.

Signed: _____

Name: _____

Signed: _____

Name: _____

After approval, assign NAS Report number.

Branch Chief:

Approved: _____

Date:

12 July 96

NAS ReportNumber:

NAS-96-009