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

James Patton Jones[1] and Cristy Brickell[1]

NAS Technical Report NAS-97-013 June 1997

jjones@nas.nasa.gov
brickell@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 [Jon96a]. Next, NAS evaluated the leading job management system (JMS) software packages against the checklist [Jon96b]. A year has now elapsed since the first comparison was published, and NAS has repeated the evaluation. This report describes this second evaluation, 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 lacking, however, definite progress has been made by the vendors to correct the deficiencies. This report is supplemented by a WWW interface to the data collected, to aid other sites in extracting the evaluation information on specific requirements of interest.

1. MRJ Technology Solutions, 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 completely support parallel jobs and parallel systems. It was to address this deficiency that NAS produced the *NAS Requirements Checklist for Job Queuing/Scheduling Software* [Jon96a] (with input from the NAS, NASA Ames, NASA Langley, NASA Lewis, Pratt Whitney, Platform Computing, PBS group; as well as input from Cray Research, Inc., and IBM). 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 to 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 leading JMS systems identified in recent reviews. Second, NAS chose to perform a more in-depth comparison with more than twice the number of criteria as the cited evaluations.

NAS also realizes that the data collected for such an evaluation is often more useful to other sites than the conclusions NAS draws from the data. This is because few sites will have the exact same requirements as the NAS. Recognizing this, the data collected for this evaluation has been placed on-line with a WWW interface to allow sites to query the data for the specific criteria and requirements important to their site. In this report we present both the evaluation data and our conclusions. The WWW interface to the data itself is available at: (**http://parallel.nas.nasa.gov/Parallel/JMS**).

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 NASA supercomputing facilities. The evaluation will proceed in three phases, as shown in Table 1. After the evaluation plan was written, we identified which JMS software packages to evaluate. Table 2 lists the six packages identified, and the versions selected for evaluation.

TABLE 1. Steps in Evaluation

Phase 1: Capabilities versus requirements

1. Obtain current production JMS release (see Table 2 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 Table 5 below).
4. Provide each vendor an opportunity to review and correct any technical errors in the evaluation of their product.
5. Rank all JMS system capabilities against requirements (see page 5).
6. Any JMS falling below MININUM THRESHOLD (90%) will be eliminated from comparison; all remaining will continue to Phase 2.
7. Summarize and publish results.

Phase 2: Staff / user testing (for each JMS meeting minimum requirements)

A. For each test platform (see Table 3 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 page 5) 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. Collect staff and user experiences from other sites already running JMS.

D. Summarize and publish results.

E. Optionally perform Phase 3 evaluation at this time.

F. Archive JMS configuration.

G. Deinstall JMS.

TABLE 1. Steps in Evaluation

Phase 3: (Optional) Full deployment, production use 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 2. JMS Software Selected for Evaluation

JMS	Version	Vendor	Released
Computing in Distributed Networked Environments (CODINE)	4.0.2b	GENIAS	November '96
Distributed Queueing System (DQS)	3.1.4.1.1	SCRI	28 August '96
LoadLeveler (LL)	1.3.0	IBM	30 August '96
Load Sharing Facility (LSF)	3.0	Platform	December '96
Network Queueing Env (NQE)	3.2	CRI	5 February '97
Portable Batch System (PBS)	1.1.9	NASA	23 December '96

A general description of each of these products is given in the *Second Phase 1 Results* section below. Two other packages that had been suggested by readers of the first report, that were not included in this evaluation were: Hector (currently only supports workstations) and GRD (did not have a non-beta release by the March 1 deadline).

Next, we generated a rough timeline for the evaluation. Table 3 shows the portion of the timeline covered by this paper. (Table 13 in Section 5 below gives the revised timeline for the conclusion of the project.).

TABLE 3. Timeline of JMS Evaluation, Phase 1

Time Period	Activity
1 March 1997:	Cut-off date for vendor release of production software.
1 March - 15 May:	Phase 1 comparison.
15 May -30 May:	Summarize and publish Phase 1 results.

Choosing a cut-off date was necessary to set a fixed window of time for the evaluation. This eliminated the perpetual waiting for the next release of each product to arrive.

We then determined which computer systems would be used for the second phase of the evaluation. The three testbed systems, and two production systems at NAS, listed in Table 4, were selected for the diversity and flexibility they provide. This list has grown since the Phase 2 prediction due to increased requirements. The five systems differ in their workload and job mix, supporting serial, vector, parallel and message-passing applications.

TABLE 4. Phase 2 Comparison Platforms

Architecture	NAS Hostname	Configuration
SGI PowerChallenge	davinci	4-node (32 CPU) workstation cluster, 1 front end
IBM SP2	babbage	160-node (160 CPU) SP2, 2 front ends
SGI Origin 2000	turing	64 CPU, 8 GB memory system
CRI J90	newton	4-node (36 CPU) cluster
CRI C90	vn	16 CPU, 1 GW memory system

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 needed a method 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) for the elimination metric. Each of those requirements was further ranked as high or medium priority. These priorities held a weight of 5 and 3 respectfully. From this we generated the following simple metric, a percentage index for the number of section 3 criteria met, taking the priority into consideration:

METRIC = [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 would 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, available online:
(http://science.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-96-003).

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 be 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 Second Phase I Results

The results of the *Second 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 1 above, each vendor was given the opportunity to review and correct any technical inaccuracies in the evaluation of their product.

Table 5 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 intended to show how closely the product met the requirement. A copy of *NAS Requirements Checklist for Job Queuing/Scheduling Software* [Jon96a] is required to interpret the evaluation data. This report is now available online at: (**http://science.nas.nasa.gov/Pubs/TechReports/NASreports/NAS-96-003**).

Table 5: Score Definitions

Score	Explanation	Weight Used in Ranking
●	Meets requirement	4
◕	Meets most of requirement	3
◑	Meets roughly half of requirement	2
◔	Meets little of requirement	1
○	Does not meet any of requirement	0

4.1 Computing in Distributed Network Environments (CODINE)

Computing in Distributed Network Environments (CODINE) is a commercially available job-management software package released by GENIAS Software GmbH, Germany. Emphasis is currently on providing JMS support across heterogeneous environments. Information for this evaluation is based on [GEN97]. Additional information is available online: (**http://www.genias.de**).

Table 6: CODINE 4.0.2b

Number	New Score	Notes
3.1.1	◕	Currently: AIX, IRIX, Solaris, SunOS, Linux, HP-UX, Digital UNIX and OSF. UNICOS is supported in CODINE 4.1.1 (April 1997).
3.1.2	◔	AFS and DCE/DFS are not yet provided
3.1.3	●	provided
3.1.4	◔	an API library is provided for some components
3.1.5	◕	# of nodes per job, system time, dedicated/shared access, and network adapter access are not enforced
3.1.6	●	implemented via environment variables
3.1.7	●	available on specific-case basis
3.1.8	●	provided
3.1.9	●	ACLs are provided
3.1.10	●	claims scalability to above 500 nodes
3.1.11	◕	meets most POSIX 1003.2d, "Batch Queueing Extensions" standards
3.2.1	◑	limited support for parallel jobs and no job-JMS communication
3.2.2	●	provided
3.2.3	○	file "stage-in" and "stage-out" are not provided
3.2.4	◕	when linked against checkpoint library
3.2.5	●	provided
3.2.6	○	no application-JMS communication available
3.2.7	◑	NFS: yes
3.2.8	○	not provided

Table 6: CODINE 4.0.2b

Number	New Score	Notes
3.3.1	◑	configurable; and through the "joint project program", scheduler can be modified to fit site needs
3.3.2	◑	only one scheduler is provided with support for FIFO, user priority and load balancing
3.3.3	●	can configure via complex lists
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	space sharing requires scheduler modifications
3.3.6	●	C framework for scheduler is provided through the "joint project program"
3.3.7	●	provided
3.3.8	◑	limited
3.4.1	●	provided
3.4.2	●	provided
3.4.3	◑	restrictions on past resource consumption and job origin are not provided
3.4.4	◕	jobs (except interactive) are requeued/resumed/rerun upon user request in the event of a system failure
3.4.5	●	provided
3.4.6	●	provided
3.4.7	●	provided
3.4.8	●	provided
3.4.9	●	provided
3.4.10	●	provided
3.4.11	●	provided
3.4.12	◕	most user specified job inter-dependency is provided
3.4.13	●	provided

Table 6: CODINE 4.0.2b

Number	New Score	Notes
3.4.14	●	provided
4.1.1	●	provided
4.1.2	●	GUI provided
4.1.3	●	GUI provided
4.1.4	◕	provided by external scheduler
4.1.5	○	graphical monitoring tool is not provided
4.1.6	◑	some hard and soft resource limits are supported
4.1.7	◕	popular package for load balancing and cycle stealing
4.1.8	○	proxy account option is not provided
4.1.9	◔	limited account information
4.1.10	●	provided
4.1.11	●	provided
4.1.12	○	resource reservation is not supported
4.1.13	◑	resource consumption counters are not supported
4.1.14	●	provided
4.1.15	●	distance is no problem as long as network works
4.1.16	●	provided
4.1.17	◑	Windows NT server available via Queuing System Interface; client facilities will be supported in CODINE 4.2
5.1.1	○	gang-scheduling is not supported
5.1.2	◔	migration of running applications to other nodes is supported
5.1.3	●	provided
5.1.4	●	where supported by O/S

4.2 Distributed Queueing System (DQS)

The Distributed Queueing System (DQS) is a freely available batch queuing system which has been under development at the Supercomputer Computations Research Institute (SCRI) at Florida State University. Emphasis is currently on providing JMS support across a heterogeneous environment. Information for this evaluation is based on source distribution documentation: [SCR96a, SCR96b, SCR96c]. Additional information is available online: (**http://www.scri.fsu.edu**).

Table 7: DQS 3.1.4.1.1

Number	New Score	Notes
3.1.1	●	Currently: UNICOS, AIX, IRIX, Solaris, Linux, HP-UX, and multiple UNIX versions
3.1.2	◑	limited DFS support, DCE is supported in DQS 4.0 (April 1997)
3.1.3	●	provided
3.1.4	◔	source distribution provides API but not separately documented
3.1.5	◔	minimal resource enforcement
3.1.6	●	implemented via different port numbers and directories
3.1.7	●	source code is freely available
3.1.8	●	provided
3.1.9	●	ACLs are provided
3.1.10	◑	limited experience with sufficiently large clusters
3.1.11	◕	meets most POSIX 1003.2d, "Batch Queueing Extensions" standards
3.2.1	◑	limited support for parallel jobs and no job-JMS communication
3.2.2	◕	no HPF support
3.2.3	○	file "stage-in" and "stage-out" are not provided
3.2.4	◕	when linked against checkpoint library
3.2.5	◕	history log of time job entered (each) queue, time job suspended or restarted execution, and resource usage such as memory are not provided

Table 7: DQS 3.1.4.1.1

Number	New Score	Notes
3.2.6	○	no application-JMS communication available
3.2.7	◑	AFS: yes, DCE is supported in DQS 4.0 (April 1997)
3.2.8	○	interactive-batch jobs are not supported
3.3.1	◑	scheduler can be modified to fit site needs
3.3.2	◔	only one scheduler is provided
3.3.3	●	can configure via complex lists
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	space sharing requires scheduler modifications
3.3.6	●	C framework for scheduler is provided
3.3.7	●	provided
3.3.8	◑	limited
3.4.1	◑	interactive jobs are not supported
3.4.2	●	provided
3.4.3	◑	restrictions on past resource consumption and job origin are not provided
3.4.4	◕	user must override default of not re-queuing job
3.4.5	●	provided
3.4.6	●	provided
3.4.7	●	provided
3.4.8	●	provided
3.4.9	●	provided
3.4.10	●	provided
3.4.11	◑	after only

Table 7: DQS 3.1.4.1.1

Number	New Score	Notes
3.4.12	○	user specified job inter-dependency is not provided
3.4.13	●	provided
3.4.14	●	provided
4.1.1	●	provided
4.1.2	●	GUI provided
4.1.3	●	GUI provided
4.1.4	◕	provided by external scheduler
4.1.5	○	graphical monitoring tool is not provided
4.1.6	◑	some soft and hard limits are supported
4.1.7	◑	public domain
4.1.8	●	provided
4.1.9	◔	limited account information
4.1.10	●	provided
4.1.11	○	interactive jobs are not supported
4.1.12	◕	scheduler can be modified to fit site needs
4.1.13	◑	resource consumption counters are not supported
4.1.14	●	provided
4.1.15	●	distance is no problem as long as network works
4.1.16	●	provided
4.1.17	○	Windows NT is not supported
5.1.1	○	gang-scheduling is not supported
5.1.2	○	dynamic load balancing is not supported
5.1.3	◔	serial only, no API

Table 7: DQS 3.1.4.1.1

Number	New Score	Notes
5.1.4	●	where supported by O/S

4.3 LoadLeveler (LL)

Loadleveler, from IBM, is a commercially available, general-purpose JMS software package. Support is provided for clusters of workstations running serial jobs and parallel jobs, as well as for the IBM SP supercomputer. Information for this evaluation is based on [IBM95a, IBM95b]. Additional information is online: (**http://www.austin.ibm.com/software/sp_products/loadlev.html**).

Table 8: Loadleveler 1.3.0

Number	New Score	Notes
3.1.1	◑	SP2, RS/6000, SUN, SGI, HP; no support for any CRI / UNICOS systems or SGI Origin 2000
3.1.2	◑	NFS and AFS only; DFS/DCE expected 3Q97
3.1.3	●	has command line interface
3.1.4	●	API for accounting, prologue, epilogue, checkpoint (serial); basic API for submit, monitor, query, and scheduler
3.1.5	◕	per-job: CPU-time and wall-clock time; per-process: memory utilization, CPU time, stack, core, file; 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	◔	limited 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

Table 8: Loadleveler 1.3.0

Number	New Score	Notes
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 provided
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
3.2.6	○	application-JMS communication not available
3.2.7	◔	UNIX-level security only; DCE support 3Q97
3.2.8	○	does not support batch-scheduled interactive jobs
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	●	allows a separate scheduler via published API
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	●	specified restrictions provided
3.4.4	◕	jobs (except interactive) are automatically requeued/resumed/rerun in event of system failure.
3.4.5	●	provided

Table 8: Loadleveler 1.3.0

Number	New Score	Notes
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
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	●	GUI provides
4.1.3	○	no graphical system configuration tool
4.1.4	○	no MACHINE stanza for this
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

Table 8: Loadleveler 1.3.0

Number	New Score	Notes
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
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

4.4 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". Supports both serial and parallel jobs, on clusters of workstations and supercomputers. Information for this evaluation is based on [Pla96a, Pla96b, Pla96c]. Additional information is available online:
(**http://www.platform.com**).

Table 9: LSF 3.0

Number	New Score	Notes
3.1.1	●	Currently: ConvexOS, UNICOS, Digital Unix, HP-UX, AIX, Linux, NEC EWS OS, Solaris, SunOS, Sony NEWS, SGI IRIX, SPP-UX
3.1.2	●	provided

Table 9: LSF 3.0

Number	New Score	Notes
3.1.3	●	commands well documented
3.1.4	◕	general API provided (not for scheduler)
3.1.5	●	no direct support for disk and network usage; but provide hooks for a site to provide such info
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
3.1.10	●	Clusters in existence of >500 hosts.
3.1.11	○	does not meet POSIX 1003.2d "Batch Queueing" standard
3.2.1	◑	limited support for parallel jobs; no job-JMS communication.
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 when linked with checkpoint library
3.2.5	◕	meets all except those listed in 3.1.5 above
3.2.6	○	no published job-JMS API
3.2.7	◕	supports NFS and AFS; has DCE support for some systems; site configurable; requires DCE 1.1
3.2.8	●	provides batch-scheduled interactive sessions
3.3.1	◕	configurable (must use provided scheduling algorithms)
3.3.2	◕	has many of those listed
3.3.3	●	can configure via HOST stanza

Table 9: LSF 3.0

Number	New Score	Notes
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	supports space-sharing (dedicated access)
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
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 except "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

Table 9: LSF 3.0

Number	New Score	Notes
4.1.4	●	via HOSTS stanza
4.1.5	◕	has monitoring tool, suggests capture 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
4.1.9	◑	JMS provides some requested data in ascii format, and simple tool to process records
4.1.10	●	suggests using separate LSF-add-on which provides "multi-cluster" support.
4.1.11	○	cannot detach/reattach; plus no concept of "interactive-batch"
4.1.12	●	resource reservations provided
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; job owner can force migration, but not workstation owner
4.1.17	◔	runs on Windows NT with a long list of restriction and missing features
5.1.1	○	no gang-scheduling
5.1.2	◑	provides auto migration of serial jobs; limited support for parallel jobs
5.1.3	◕	provided for serial jobs and some parallel jobs, if linked with checkpoint library
5.1.4	●	provided

4.5 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 queuing for clusters of workstations running serial and parallel jobs. Information for this evaluation is based on [Cra95a, Cra95b, Cra95c]. Additional information on the latest release is available online:

(http://www.cray.com/products/software/nqe/).

Table 10: NQE 3.2

Number	Score	Notes
3.1.1	●	Solaris, SunOS, IRIX, AIX, HP-UX, DEC UNIX, UNICOS, UNICOS/mk
3.1.2	◕	DCE/DFS only; on all except IRIX, SunOS, UNICOS/mk
3.1.3	●	has command-line interface
3.1.4	◕	API to most components
3.1.5	◑	supports all NQS resource limits; no "node" or equivalent support
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	◑	manages T3E systems with hundreds of CPUs and IRIX PCA with large numbers of Nodes and CPUs.
3.1.11	◔	little compliance with POSIX 1003.2d, "Batch Queueing" standard
3.2.1	◑	not provided for multi-node jobs; expected in future release
3.2.2	◔	supports PVM; no mention of MPI or HPF; expected in future release
3.2.3	◕	provides a "file-transfer agent" to move data from system to system, with fault tolerance

Table 10: NQE 3.2

Number	Score	Notes
3.2.4	◑	system-level checkpoint/restart where supported by OS; no JMS-assisted user-level checkpointing, but user can specify system checkpoint interval.
3.2.5	◕	provides ascii accounting logs with most info; is integrated with UNICOS system accounting
3.2.6	○	no application-JMS communication available
3.2.7	◕	DFS/DCE support on DCE supported systems only
3.2.8	○	not provided; suggest launching xterm from batch job
3.3.1	◕	configurable (via TCL interface)
3.3.2	◑	provides FIFO, load balancing, fair share on UNICOS, URM, job ordering by time and size.
3.3.3	●	network load balancer manages most requirements; scheduler could be extended to handle rest.
3.3.4	◕	once running, observable resources only; other job states: yes
3.3.5	●	supports space-sharing
3.3.6	◕	scheduler can be replaced; tcl-based scheduler interface available for site customizations
3.3.7	●	supports unsynchronized time-sharing
3.3.8	◔	limited
3.4.1	◑	handles only 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	●	provided
3.4.7	●	provided

Table 10: NQE 3.2

Number	Score	Notes
3.4.8	◑	limited
3.4.9	◑	limited
3.4.10	◔	limited parallel awareness
3.4.11	○	no prologue/epilogue support
3.4.12	◕	no status of other computer systems
3.4.13	●	access restrictions apply
3.4.14	◑	configurable: user can view either their jobs or all jobs
4.1.1	●	provided
4.1.2	●	motif/X and WWW
4.1.3	◑	limited configuration via GUI
4.1.4	◑	limited to TCL interface
4.1.5	◑	basic graphical monitoring tool
4.1.6	◑	hard limit: yes; soft limit: no
4.1.7	◕	supported by large software company
4.1.8	●	via ACLs and "administrative domain" features
4.1.9	◑	much of necessary data provided, no tools to process data however
4.1.10	●	provided via "network based scheduler"
4.1.11	○	cannot detach/reattach; plus no concept of "interactive-batch"
4.1.12	○	no resource reservations
4.1.13	◑	no computation counters
4.1.14	◔	limited ACLs
4.1.15	●	distance not an issue as long as network is stable and reliable

Table 10: NQE 3.2

Number	Score	Notes
4.1.16	●	provides workstation owner-JMS interaction
4.1.17	◔	no direct Windows NT support (has web access only)
5.1.1	◔	gang-scheduling only under UNICOS/mk
5.1.2	○	no dynamic load-balancing
5.1.3	○	no job migration support
5.1.4	●	where supported by OS

4.6 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 1997 release, with support for dynamic resource management to follow. Information for this evaluation is based on [Hen95, Hen96]. Additional information is available online: (**http://science.nas.nasa.gov/Software/PBS**).

Table 11: PBS 1.1.9

Number	New Score	Notes
3.1.1	●	Currently: IRIX, AIX, UNICOS, SunOS, Solaris, CM5, SP2, CRAY C90, J90
3.1.2	◔	has NFS support; AFS, DCE/DFS support due 4Q97
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

Table 11: PBS 1.1.9

Number	New Score	Notes
3.1.8	●	provides both manager and operator IDs, as well as flexible restrictions on "root" jobs and connections.
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
3.1.11	●	Fully compliant with POSIX 1003.2d
3.2.1	○	capability will be included in "full parallel awareness" (due 3Q97)
3.2.2	◑	"supports" but does not "interact"; capability will be included in "dynamic parallel awareness"
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 "dynamic parallel awareness"
3.2.5	◕	meets all except a couple of the resources specified in 3.1.5 except complete resource accounting, provided with "full parallel awareness" (due 3Q97)
3.2.6	○	capability will be included in "dynamic parallel awareness"
3.2.7	◑	UNIX-level security only; allows site to replace security mechanism; DCE support due 4Q97
3.2.8	●	provided
3.3.1	●	administrator can 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

Table 11: PBS 1.1.9

Number	New Score	Notes
3.3.8	●	via PBS nodefile
3.4.1	●	"qsub -I" indicates 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" 3Q97)
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	●	all logs are located on server host
3.4.10	○	capability to be included in "full parallel awareness" (due 3Q97)
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	●	GUI provided
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

Table 11: PBS 1.1.9

Number	New Score	Notes
4.1.7	◕	public domain with support promised by NASA for 5 years past last feature release
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; suggests using ACCT++ accounting package, also available free from NAS
4.1.10	●	provided
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	◑	PBS can detect keyboard/mouse activity and respond accordingly; does not yet provide additional workstation owner-JMS 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 3Q97)
5.1.3	○	first part will be "full parallel awareness" (due 3Q97)
5.1.4	●	where supported by OS (e.g. UNICOS)

5.0 Conclusions

In analyzing the data collected from the evaluation, we found that once again none of the leading JMS packages meet enough of our requirements. Both from the evaluation experience and from actually applying the metric described on page 5 we found that none of the JMSs evaluated meet our minimum criteria threshold. In fact, if we were to drop the threshold from 90 percent to 75 percent, only three of the six JMSs would meet the minimum. Table 12 shows the ranking that each JMS received on the threshold metric (see page 5 for details of the metric formula).

TABLE 12. Ranking based on Threshold Metric

JMS Package	Score (Weighted Percentage of Section 3 Requirements Met)
PBS	82.49
LSF	78.53
CODINE	77.82
LL	73.87
DQS	71.61
NQE	69.21

Please note that this threshold metric was intended only to eliminate less capable JMSs from the Phase 2 evaluation, and is not intended to reflect how each product would meet the needs of any site other than NAS. 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. Sites 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. The online "Job Management System Evaluation Station" was created so that these comparisons could be generated dynamically. (See **http://parallel.nas.nasa.gov/Parallel/JMS**).

Again this year, the bad news is the confirmation of a continuing lack of JMS support for parallel applications, parallel systems, and clusters of workstations. However, the four products reviewed last year showed growth in critical areas.

Once again, 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 1 above. Assuming the product release schedules announced by the various vendors hold firm, Table 13 shows the revised timeline for the next evaluation.

TABLE 13. Revised Timeline of JMS Evaluation

Time Period	Activity
1 March - 1 April 1998	Repeat Phase 1 comparison
1 April - 1 May 1998	Summarize and publish Phase 1 results
1 May - 31 June 1998	Phase 2 comparison

TABLE 13. **Revised Timeline of JMS Evaluation**

Time Period	Activity
1 July - 15 July 1998	Summarize and publish Phase 2 results
15 July - 31 Nov 1998	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 NAS JMS Requirements document was reviewed by members of the NASA Consolidated Supercomputing Management Office (CoSMO) software committee. 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.

[Cra95a] "Introducing NQE," CraySoft, Cray Research Inc., Document Number IN-2153 2/97, 1997.

[Cra95b] "NQE Administration," CraySoft, Cray Research Inc., Document Number SG-2150 3.2, 1997.

[Cra95c] "NQE User's Guide," CraySoft, Cray Research Inc., Document Number SG-2148 3.2, 1997.

[GEN97] "Installation and Administration Guide 4.0," GENIAS Software GmbH, Neutraubling, Germany, March 1997.

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

[Hen96] "Portable Batch System: External Reference Specification," Robert Henderson and Dave Tweten, NAS, NASA Ames Research Center, December 1996.

[IBM95a] "IBM Loadleveler Administration Guide, Release 3.0," IBM, Document Number SC23-3989, August 1996.

[Jon96a] "NAS Requirements Checklist for Job Queuing/Scheduling Software," James Patton Jones, NAS Technical Report NAS-96-003, NAS, NASA Ames Research Center, April 1996.

[Jon96b] "Evaluation of Job Queuing/Scheduling Software: Phase 1 Report," James Patton Jones, NAS Technical Report NAS-96-009, NAS, NASA Ames Research Center, July 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.

[SCR96a] "DQS 3.1.3 User Guide," Supercomputer Computations Research Institute, Florida State University, March 1996.

[SCR96b] "DQS 3.1.3 Reference Manual," Supercomputer Computations Research Institute, Florida State University, March 1996.

[SCR96c] "DQS 3.1.3 Installation and Maintenance Manual," Supercomputer Computations Research Institute, Florida State University, August 1996.

NAS	NAS TECHNICAL REPORT
	Title: Second Evaluation of Job Queuing / Scheduling Software: Phase 1 Report
	Author(s): James Patton Jones, Cristy Brickell
Two reviewers must sign.	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." Signed: ______________________ Name: ______________________ Signed: ______________________ Name: ______________________
After approval, assign NAS Report number.	Branch Chief: Approved: ______________________
Date: 11 June 97	NAS Report Number: NAS-97-013