# Implementation of the NASA Metacenter: Phase 1 Report

James Patton Jones[1]

NAS Technical Report NAS-97-027 October 1997

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

**Abstract**

This paper discusses the efforts to create, using two NASA SP2 supercomputers, a "Metacenter" which includes the capability to transparently and dynamically distribute the SP2 workload across the geographically separated systems. Functional components of the Phase 1 Metacenter are identified, outstanding issues are discussed, and the plan for the second phase of the project is outlined.

## 1.0 Introduction

The NASA Metacenter is a joint exploratory project between the NAS parallel systems group at NASA Ames Research Center (ARC) and the parallel systems staff at NASA Langley Research Center (LaRC). The focus of the project is to achieve more effective use of NASA supercomputers by making the systems more easily available to the researchers, and by providing quicker turn-around for batch jobs, a larger range of available resources for computation, and a better distribution of the computational workload across multiple supercomputers.

---

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

But what exactly is a “Metacenter”? There are several differing interpretations. The definition that best illustrates this project is that of the National Science Foundation (NSF): “a metacenter is a computing facility whose computational capability is greater than the sum of the component systems.”

## 2.0 Why a Metacenter?

In July 1994 two IBM POWERparallel SP2 supercomputers were acquired by NASA under the HPCCPT-1 Cooperative Research Agreement (CRA) between NASA and a consortium led by IBM. Table 1 shows the configuration of the two systems.

**TABLE 1. SP2 Configurations**

| ARC SP2 (“babbage”) | LaRC SP2 (“poseidon”) |
|---|---|
| 160 IBM RS600 processors (66.7 MHz) | 48 IBM RS600 processors (66.7 MHz) |
| Minimum 128 MB memory per node | Minimum 128 MB memory per node |
| Six 512 MB memory nodes | Four 512 MB memory nodes |
| One GB temporary disk space per node | 0.5 GB temporary disk space per node |

In the Spring of 1995, the parallel systems staff at these sites began discussing the differences in the utilization of the two SP2 systems, babbage (ARC) and poseidon (LaRC). While babbage was three times the size of poseidon, it was achieving twenty times the utilization. Consequently, jobs on babbage had a slow turn-around time (up to 32 hours) in the queue.

Upon investigation, the staff found that poseidon’s lower utilization was due in part to the smaller size of the system and to a smaller user base. Much of the work in the CRA was intended to run on the larger SP2, resulting in an imbalance of users on the two systems.

When we began looking for solutions to this problem, the Metacenter idea was suggested. What if users from either system could submit jobs and have them transparently run on the most appropriate system? This would provide many benefits to the two SP2 user communities, including quicker turn-around for batch jobs, a larger range of available resources for computation, and a better balanced utilization of compute resources.

## 3.0 Creating the Metacenter: Administrative Coordination

Many obstacles had to be overcome to implement the Metacenter. In order to provide transparent movement of jobs between the two systems, the environments on both systems had to be the same. Table 2 lists the key software solutions utilized in the Metacenter.

**TABLE 2. Software Used in the Metacenter**

| Need/Requirement | Software Package | Email Address or URL for Additional Information |
|---|---|---|
| User Account Management | LAMS | accounts@nas.nasa.gov |
| Integrated Accounting | ACCT++ | acctgrp@nas.nasa.gov |
| Single Queuing System | PBS | http://science.nas.nasa.gov/Software/PBS |
| Metacenter Job Scheduler | PeerSched | jjones@nas.nasa.gov |
| Job Submission and Tracking | xPBS | http://parallel/Parallel/PBS/xpbs.html |
| System Monitoring | CTMS | http://eeyore.nas.nasa.gov/ctms.html |

### 3.1 Selecting a Batch Queuing System

The biggest difference in the environments of the two systems was the job management/queueing software in use. IBM's Loadleveler product was managing jobs on poseidon, but ARC had replaced Loadleveler in January 95 with the NAS-developed Portable Batch System (PBS) on babbage. PBS had been selected for babbage when LoadLeveler's job scheduling capability was determined to be inadequate for this size system. (For a current comparison of capabilities, see [Jon97].) Installing PBS had a dramatic effect on babbage, resulting in more than twice the utilization (see [Tra95]). However, at that time Loadleveler provided interactive access to the SP2 nodes, a requirement on poseidon, while PBS provided only batch access. Once support for interactive access was added to PBS, it was installed on poseidon as well. (Additional information about PBS is available at http://science.nas.nasa.gov/Software/PBS ).

### 3.2 Synchronizing System Software

Next we turned our attention to system software. Executable code compiled and linked on one system had to be able to run on the other. Libraries, compilers, operating systems, and parallel software all had to be the same. Over the rest of 1995 we worked to synchronize the software configuration on both systems.

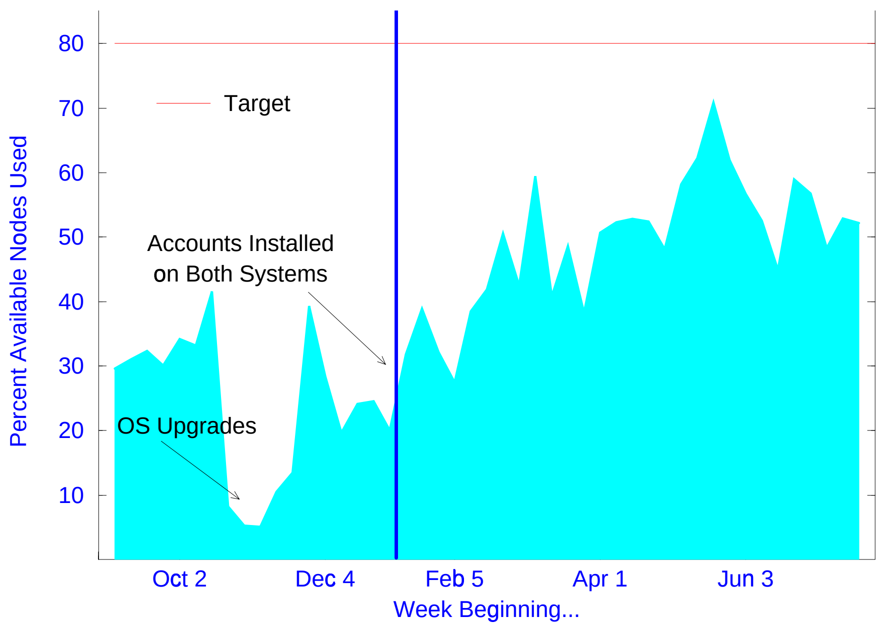
Also during this period, we had to synchronize the support of the two systems. All discussions of software and hardware changes necessitate coordination of the other site. This coordination is accomplished through a short weekly teleconference where we discuss systems changes, propose coordinated upgrades, exchange SP2 experience and knowledge, and track our progress toward the Metacenter.

The most critical of these changes was the upgrade to the next version of IBM's operating system (AIX 4.1.3) and parallel environment (PSSP 2.1.3) across all nodes of both systems. The administrators from both sites worked together, first in cooperation with Stanford University to upgrade Stanford's 16-node SP2 system. Much was learned from this small system that saved days of down time on the larger systems. Next was the upgrade of poseidon, where more experience was gained before tackling the larger system at ARC. (At that time, babbage was by far the largest system to attempt this upgrade.) The collaboration on the upgrades reduced the downtime of both sites, and provided important information to IBM on software bugs and stability problems in their installation tools. Many of these problems were corrected before other large system sites attempted the same upgrade A key benefit to users during these upgrades was continued access to an SP2 system. We set up a "routing queue" within the batch system between the two SP2's which allowed users to submit jobs directly to the other system.

### 3.3 Username and Account Management

The Metacenter team met at SuperComputing '95 to discuss the next steps in detail. In January 1996 we began efforts to ensure all users had accounts on both systems by default. Here we ran into a variety of problems. Ideally, we would have common usernames across both systems. But when we went to add accounts we found several dozen username conflicts. Realizing that getting users to voluntarily change their login names would be difficult, we decided a different approach to the problem was necessary. In setting up new accounts, we installed the new user with his/her username from the other system if no conflict existed.

**FIGURE 1: Metacenter SP2 Utilization, Late-95 thru 96**




However, if a conflict did exist, we selected a new unique username for the second system. To permit users to submit jobs to either system, without *having* to specify which username to run under, we implemented *username mapping* in both PBS and the job scheduler. This capability determined which username the job will run under, based on who submitted a job, and from where. Figure 1 illustrates how the utilization increased by simply increasing the size of the user-base.We also reduced the paperwork complexity for new accounts by combining the new user Account Request Form from both sites into a single document that could be used for either system. The NAS site-wide Login Account Management System (LAMS) was installed on the LaRC SP2 to assist with installation and management of user accounts. Procedures were put in place to inform both sites when a new account was installed.

Once the software environments were synchronized and most of the user accounts were installed on both machines, we opened the Metacenter for user testing. Users who wanted to take advantage of the second system had only to request their password for that system. The intent was to make both systems available while we worked on the next hurdle of the project: automatic load-balancing between the two systems.

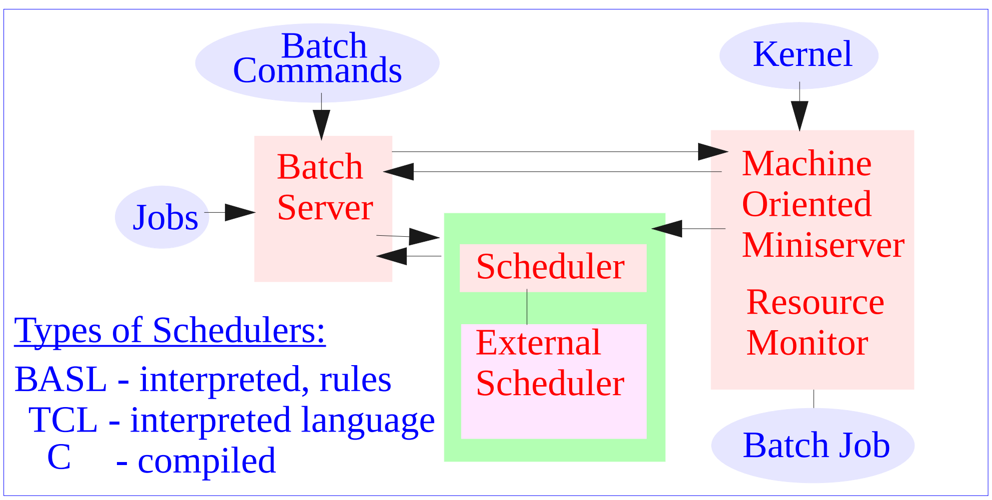
## 4.0 Creating the Metacenter: Functionality

With the administration support layer in place, we next focused on the functional areas of the project: job scheduling, file staging, job accounting, and support for locating and tracking jobs.

### 4.1 The PBS Job Scheduler

The first functional area we tackled was the job scheduler, which is external to the rest of PBS, as shown in Figure 2. The designers of PBS recognize that the

**FIGURE 2: External PBS Scheduler**




job scheduler is the most site-specific part of a batch queueing system, since it is the scheduler that implements the policy of each specific machine or site. Thus, PBS provides an “external scheduler,” one that can be modified as needed. PBS provides three interfaces to the scheduler: BASL (a scheduler scripting language), TCL (a general-purpose interpreted scripting language), and a C language application programming interface (API).

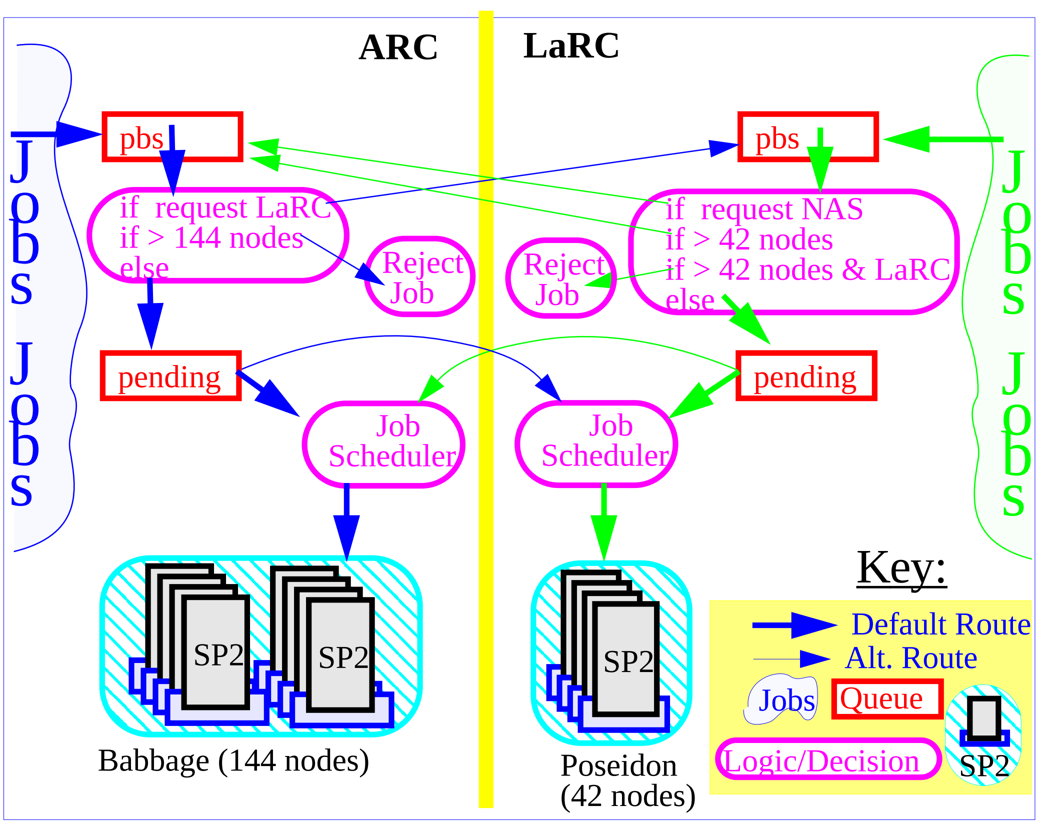
NAS implemented its first PBS scheduler in TCL, mainly because of the quick prototyping capability it afforded. When PBS was installed on the LaRC SP2, we decided to start with the TCL-based job scheduler there as well.

While TCL proved sufficient for prototyping, it was inadequate for the larger task ahead: creating a Metacenter scheduler. The TCL-based scheduler was rewritten in C and installed on both SP2s. The design of the scheduler called for a "configuration file" to be read by the scheduler upon start-up. This enables us to change scheduling parameters without having to recompile the program. The configuration file also allows us to have a single scheduler source code tree for all Metacenter systems, since the system-specific policies are defined in the configuration file.

### 4.2 Metacenter "Peer-Aware" Job Scheduler

Next we added support for "peer-scheduling". Under normal operational load, the Metacenter systems act as independent systems. However, when the utilization on one system drops below a pre-defined threshold, that system attempts to request jobs from its "peer systems". Figure 3 illustrates the separate but "peer-

**FIGURE 3: Metacenter Queues and Schedulers**



aware” design that we have implemented, showing our PBS queues and the ability of the job schedulers to retrieve work from either queue.

Of course, the scheduler will only request jobs with resource requirements it can fulfill. In addition, users were given the ability to request that their job run on a specific system. We have since decided that future phases of the Metacenter will not offer this option. Some users abused this option, always requesting a specific system when there was no compelling reason to do so. This had the result of reducing the overall efficiency of the Metacenter since the scheduler was limited in the amount of load-balancing it could perform.

### 4.3 Data Availability

To make the Metacenter functionality truly transparent to the user, a global shared filesystem between both systems is needed. We had originally anticipated DCE and DFS being available for general use, but this has been delayed. Once DCE and DFS are functioning and stable, we will consider integrating them into the Metacenter. In the meantime, other options are under investigation.

Until we do have a global shared filesystem, users can use the PBS-provided “file staging” capability to specify which files to stage onto (and off of) the host where PBS will run their job. In the second phase Metacenter, we will be improving the ease of use of PBS staging directives, since one of the most frequent reasons for job failure has been typos in the staging directives. Another problem that needs to be addressed is that some users refuse to use file-staging. One common “loophole” in the policy is exemplified by users keeping copies of their entire datasets on both system, and then specifying zero-length files be staged in with their jobs. We believe that making file-staging more robust and easier to use should help with this problem.

Another area that we worked on to improve data availability was creating consistent filesystem-naming conventions. Both systems had different names for each of the three home filesystems. We decided to hide these differences from users by adopting a “/u/<username>” naming structure for all home directories. This allowed the user to use the same path name on both systems to get to their home directory, regardless of the actual underlying filesystem name. We also changed the name of the scratch and Parallel I/O filesystems to be the same on both systems. The primary problem we encountered with these changes were the result of users who insisted on hardcoding specific filesystem names in their batch jobs or applications. Such hardcoded pathnames worked fine until we made a change to the underlying filesystem. Such changes would have been transparent had the users used the “/u/<username>” convention.

### 4.4 Job Tracking

PBS provides a tool which greatly simplifies using the Metacenter: a graphical user interface (GUI) to PBS called “xPBS”. From a single window, a user can query and monitor the status of jobs on all PBS systems where the user has an account. From here, the user can submit both batch and interactive jobs, specify files to stage in and stage out, list job dependencies, and even track jobs as they move between queues and servers.

### 4.5 Username Conflicts Revisited

Next we revisited the username issue. When we originally installed users on both systems, we resolved the username conflicts by giving some users different usernames on the two systems. Even though the username mapping was working, it was decided that we could simplify use of the Metacenter by requiring common usernames, UIDs, and GIDs on both systems. The users with conflicting usernames cooperated willingly for the good of the project. We used the LAMS software to change the user information on each system. (For details on LAMS and other software used in the Metacenter, see Table 2 on page 3.)

### 4.6 Job Accounting

In order to provide integrated batch job and system accounting, we installed the NAS accounting system ACCT++ on both systems. This consolidated all Metacenter accounting, making data for all component systems available through a single interface. From any Metacenter system, users are able to query their operational year allocation and usage for the entire Metacenter as well as individual system usage.

### 4.7 System Monitoring

An additional software tool installed on babbage is the Centralized Test Management System (CTMS). This is a client-server application which permits administrators to “subscribe” to receive notification of “events” that occur on specific systems or groups of systems. We use this tool primarily to monitor file-systems (reporting if threshold and maximum percentage utilization limits are reached) and critical system processes (e.g. NFS daemons, PBS daemons, IBM Job Manager daemons, switch daemons). Local tests also check the status of specific system components and report any problem via CTMS.

Following the installation and integration of the above described software and the peer-aware job scheduler, we began staff-testing the full system. We enabled the peer-scheduler for full user-testing in mid-August 1996. The Metacenter scheduler was used by default starting October 1, 1996, the beginning of the FY97 operational year. During the year, we measured the success of the Metacenter project against a set of metrics, as shown in Table 3. All the Metacenter metrics are available online: *http://parallel.nas.nasa.gov/Parallel/Metrics*

**Table 3: Metacenter Metrics**

| Goal | Metric | Measures... |
|---|---|---|
| Explore Low Utilization | Batch Jobs | How many batch jobs are run on the Metacenter systems. |
| Decrease Turnaround for Small Jobs | Job Queue Time | How long jobs wait in a queue before running, measuring how well the scheduler balances the workload. |
| Evaluate Effectiveness of Peer-scheduler | Job Migration | How many jobs are migrated from one SP2 to the other, allowing these jobs to run sooner. |
| Balance Utilization | System Utilization | How busy the scheduler keeps the system, given the available workload. |

## 5.0 Phase 1 Conclusions

Now that we have completed our first year of running the dynamically load-balancing Metacenter, we look forward to applying the lessons learned, experiences gained and technology developed to the next phase of the Metacenter Project. Although the Metacenter is still in development, the steps we have taken toward its implementation have resulted in substantial benefits to the researchers using the systems.

To date, the NASA Metacenter is the only successful extended attempt at dynamically distributing a real-user production workload across geographical distances using computational resources in different political domains. Accomplishments achieved in the past year include:

- Balancing demands on over-used and under-used systems;
- Providing faster job turnaround;
- Decreasing time-to-solution;
- Providing researchers with a wider range of available resources;
- Running larger jobs more often;
- Automatically migrating jobs, with ability for users to direct or limit the migration.

The Metacenter efforts, however, do not end here. We plan to continue to add capabilities and systems, illustrating the benefits and stability of our approach. Currently planned activities include:

**TABLE 4. Phase 2 Metacenter Timeline**

| Milestone | Timeline |
|---|---|
| Transfer technology to DoD sites (ASC and WES Major Shared Resource Centers) | On-going |
| Support for Global Shared Filesystem | Vendor Dependent |
| Involve additional sites (e.g NASA Lewis Research Center) | Fall 1997 |
| Transfer technology onto production (i.e. Cray) systems in support of the NASA Aeronautics Consolidated Supercomputing Facility | Fall 1997 |
| Transfer technology onto Phase 2 Metacenter (i.e. new testbed architecture) and continue development | Winter 1997 |
| Explore issues of a heterogeneous Metacenter | Spring 1998 |
| Scheduler Support for Synchronous Job Start | Summer 1998 |
| Scheduler Support for Jobs Which Span Multiple Systems | Summer 1998 |
| Scheduler Support for Dynamic Resource Allocation | Fall 1998 |

We are now in the process of reviewing the experiences of the past year and beginning to design the Phase 2 Metacenter. We anticipate making design modifications based on lessons learned and expected configuration changes. Specifically, we will be switching to a new hardware architecture, growing the Metacenter by one site (from 2 to 3) this fall, and planning for additional sites within the coming year.

## 6.0 Online Information and Current Status

Current information on usage, capability and project status of the NASA Metacenter is maintained online at:

*http://parallel.nas.nasa.gov/Parallel/Metacenter*

Plans and discussions for the Phase 2 Metacenter will be made available from this web-page. There is also a mailing list for discussion of the NASA Metacenter. Contact the author for additional information.

## 7.0 Acknowledgments

The Metacenter efforts have been possible through the continued efforts of many individuals. Policy support was provided by Geoff Tennille and Leigh Ann Tanner. Individuals involved with the design and/or technical implementation of the Metacenter include:

John Bond, Cristy Brickell, Nick Cardo, Archie de Guzman, Steve Heistand, Bill Hicks, Ed Hook, Mary Hultquist, James Patton Jones, Bob Lukens, Dave McNab, Chuck Niggley, Alan Powers, Mark Smith, Dave Tweten, Judith Utley, and Sarita Wood.

## 8.0 References

[Tra95] "A Dynamic-Backfilling Algorithm for Efficient Space-Sharing Scheduling on an IBM SP2", Bernard Traversat, NAS, NASA Ames Research Center, October 1995

[Jon97] "Second Evaluation of Job Queuing/Scheduling Software: Phase 1 Report," James Patton Jones and Cristy Brickell, NAS Technical Report NAS-97-013, NASA Ames Research Center, June 1997.

| | |
|---|---|
| | # NAS TECHNICAL REPORT |
| | Title: Implementation of the NASA Metacenter: Phase 1 Report |
| | Author(s): James Patton Jones |
| Two reviewers must sign. | Reviewers:<br>"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."<br>Signed: ______________________________<br>Name: ______________________________<br>Signed: ______________________________<br>Name: ______________________________ |
| After approval, assign NAS Report number. | Branch Chief:<br>Approved: ______________________________ |
| Date: | NAS ReportNumber: |