Purple Computational Environment

With Mappings to ACE Requirements for the General Availability User Environment Capabilities

Version 2.0

Blaise Barney, Jean Shuler (ed.)

Prepared by Lawrence Livermore National Laboratory
7000 East Avenue, Livermore, CA 94550-9234

Approved for public release, unlimited dissemination
UCRL-TR-223888

December 5, 2006

Lawrence Livermore National Laboratory

Disclaimer

This work was performed under the auspices of the U.S. Department of Energy by University of California, Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Purple Computational Environment
Version 2.0

Blaise Barney
Jean Shuler (ed.)
Lawrence Livermore National Laboratory
7000 East Avenue
Livermore, CA 94550

Abstract

Purple is an Advanced Simulation and Computing (ASC) funded massively parallel supercomputer located at Lawrence Livermore National Laboratory (LLNL). The Purple Computational Environment documents the capabilities and the environment provided for the FY06 LLNL Level 1 General Availability Milestone. This document describes specific capabilities, tools, and procedures to support both local and remote users. The model is focused on the needs of the ASC user working in the secure computing environments at Los Alamos National Laboratory, Lawrence Livermore National Laboratory, and Sandia National Laboratories, but also documents needs of the LLNL and Alliance users working in the unclassified environment.

Additionally, the Purple Computational Environment maps the provided capabilities to the Tri-lab ASC Computing Environment (ACE) Version 8.0 requirements. The ACE requirements reflect the high performance computing requirements for the General Availability user environment capabilities of the ASC community. Appendix A lists these requirements and includes a description of ACE requirements met and those requirements that are not met for each section of this document. The Purple Computing Environment, along with the ACE mappings, has been issued and reviewed throughout the Tri-lab community.

Acknowledgements

Thanks are extended to the following individuals who contributed to the content and review of this document:

Blaise Barney
Brian Carnes
Doug East
Dave Fox
Sheila Faulkner
Scott Futral
Mark Gary
Terry Girill
Stella Hadjimarkos
Pam Hamilton
Inez Heinz
Moe Jette
Terry Jones
Don Lipari
Michel McCoy
Chris Morrone
Terri Quinn
Randal Rheinheimer
Mark Seager
Jean Shuler
Joe Slavec
Tom Spelce
Becky Springmeyer
Judy Sturtevant
Py Watson
Dave Wiltzius
Mary Zosel

Change Table

Version	Date	Comments
1.0	4/10/06	First working draft distributed among the Purple Computational Environment team
1.1	6/01/06	Second working draft with comments incorporated from the Purple Computational Environment Team.
1.2	8/11/06	Third working draft – updates
1.3	10/31/06	Add Posix info (Jeff Fier) and Viz Queue info
1.4	11/22/06	Fourth working draft - updates and additions
2.0	12/5/06	Final version

Table of Contents

Figures and Tables

Acronyms and Abbreviations

Also see http://www.llnl.gov/computing/hpc/documentation/acronyms.html for additional acronyms

AIX	Advanced Interactive eXecutive (IBM's version of the UNIX operating system)
ACE	ASC Computing Environment
ASC	Advanced Simulation and Computing
CCC	Capability Computing Campaign
CEC	Capability Executive Committee
CPAC	Capability Planning Advisory Committee
CPI	Capability Performance Indicator
DAT	Dedicated Application Time
DSW	Directed Stockpile Work
EPR	(Tri-lab) Expedited Priority Run
GA	General Availability – (When the machine is ready for production computing and any of the class of users the machine is targeted to serve may request and be granted accounts.
GPFS	Global Parallel File System
HPC	High Performance Computing
HPSS	High Performance Storage System
LA	Limited Availability. A machine is available for use but not for general availability (GA). A limited number of user accounts are added.
LC	Livermore Computing
LCRM	Livermore Computing Resource Management system
OCF	Open Computing Facility
PCE	Purple Computational Environment
PUM	Purple Usage Model
SARAPE	Synchronized Account Request Automated Process
SCF	Secure Computing Facility
SLURM	Simple Linux Utility for Resource Management
SWL	Synthetic Work Load
WAN	Wide Area Network

Purple Computational Environment
Version 2.0

Figure 1: ASC Purple Photographs

Introduction

This document provides an accurate description of the capabilities available to support the Tri-lab General Availability Purple User Environment Milestone. As stated in the ASC 2006 Program Plan:

> **350: Develop a 100 teraOPS platform environment supporting tri-lab DSW and Campaign simulation requirements.**

This milestone is the direct result of work that started seven years ago with the planning for a 100 Teraflop Tri-lab platform, code named ASC Purple, to be located at LLNL. The achievement of this milestone was reached on November 6, 2006 when Purple was placed in General Availability (GA) operation for Stockpile Stewardship Program simulations.

Purple is designed to be used to solve the most demanding stockpile stewardship problems; that is, the large-scale application problems at the edge of our understanding of weapon physics. This fully functional 100 TeraOPS system must be able to serve a diverse scientific and engineering workload. It must also have a robust code development and production environment both of which facilitate the workload requirements.

The ASC Purple system represents a substantial increase in the classified compute resources at LLNL for NNSA. The central computing environment is designed to accept the Purple system and to scale with the increase of compute resources to achieve required end-to-end services. Networking, archival storage, visualization servers, global file systems, and system software have all been enhanced to support Purple's size and architecture. IBM and LLNL are sharing responsibility for Purple's system software. LLNL is responsible for the scheduler, resource manager, and some code development tools. Through the Purple contract, IBM is responsible for the remainder of the system software including the operating system, parallel file system, and runtime environment.

LLNL, LANL, and SNL share responsibility for the Purple user environment. As the host for Purple, LLNL has the greatest responsibility. LLNL will provide customer support for Purple to the Tri-labs and as such has the lead for user documentation, negotiating the Purple computational environment, mapping of the ASC Computational Environment requirements to the Purple environment and demonstrating those requirements have been met. In addition LLNL demonstrates important capabilities of the computing environment including full functionality of visualization tools, file transport between Purple and remote site file systems and the build environment for principle ASC codes. LANL and SNL are responsible for delivering unique capabilities in support of their users, porting important applications and libraries, and demonstrating remote capabilities. The key capabilities tested by LANL and SNL include user authorization and authentication, data transfer, file system, data management, and visualization, as well as porting and running in production mode a few key applications on a substantial number of Purple nodes.

In reference to the milestone and this document, General Availability extends to the capabilities provided by the system, system availability, and the number of users. General availability is when a machine is ready for production computing, and any of the class of users the machine is targeted to serve may request and be granted accounts. Account allocations and job scheduling are based on a newly developed governance model for allocating resources and scheduling the stockpile stewardship workload on ASC Capability Systems. This "Capability Compute System Scheduling Governance Model" is included as Appendix B to this document. Most capabilities that users expect to be available are in place; a few capabilities may be limited in performance and functionality. Follow-on work continues on these capabilities as indicated in this document. In the GA Purple system, attempts will be made to schedule downtimes, though it may not always be possible to schedule these with adequate lead time for users to plan their work schedule with confidence. It will be necessary for users to coordinate their major calculations with the computer center to enhance chances of a successful outcome. Each laboratory is requested to put forth applications for the General Availability milestone, with the intent to restrict the number of users and applications for the milestone.

Users of Purple are expected to engage in up to eight activities, all of which are supported in the General Availability milestone, and described in detail throughout this document:

1) Getting started (learning about the system, gaining access etc.),
2) Setting up the work environment,
3) I/O and Data migration,
4) Application and system code development,
5) Problem setup,
6) Running the application to solve the problem,
7) Processing simulation output,
8) Tri-lab coordinated operational support.

Additionally, the Purple Computational Environment maps the provided capabilities to the Tri-lab ASC Computing Environment (ACE) Version 8.0 requirements. The ACE requirements reflect the high performance computing requirements for the General Availability user environment capabilities of the ASC community. Appendix A lists these requirements and includes a description of ACE requirements met and those requirements that are not met for each section of this document. The Purple Computing Environment, along with the ACE mappings, has been issued and reviewed throughout the Tri-lab community.

In summary, this document describes specific capabilities, tools, and procedures to support both local and remote users of ASC Purple according to ACE requirements. The ACE model is focused on the needs of the ASC user working in the secure computing environments at LANL, LLNL, and Sandia. It also considers the needs of the Tri-lab users and Alliance users working in the unclassified Purple environment.

1. Getting Started (Learning About the System, Gaining Access, etc.)

1.1. Platform Map

ASC Purple actually consists of two separate systems: the classified Purple system and the unclassified uP (unclassified Purple) system. Both of these systems share the same basic configuration shown below.

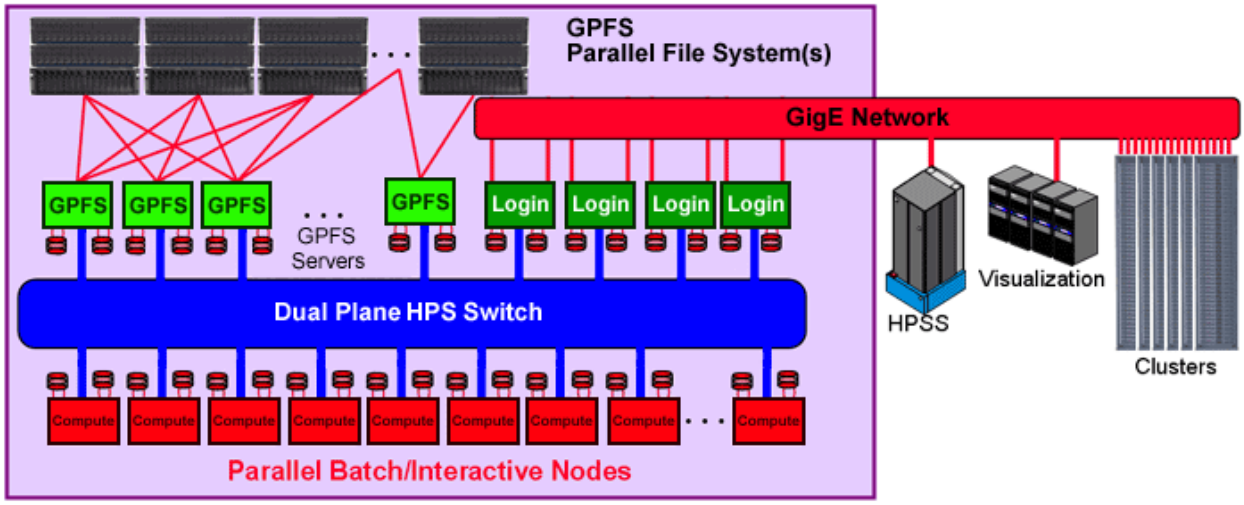


Figure 2: Purple Configuration Schematic

Both systems are comprised of the same basic components:

- IBM p5 575 nodes with eight POWER5 processors per node, operating at 1.9 GHz
- High Performance Switch (HPS) interconnect between nodes
- GPFS parallel file systems
- GigE connectivity to external systems including HPSS, visualization systems, outside networks and other clusters
- Frames to hold nodes and HPS hardware
- Hardware management consoles and associated hardware (not shown)

Both systems are also similar in that they configure available nodes into one of four possible uses:

- Compute nodes - the majority of the system, dedicated for batch use
- Interactive nodes - much smaller partition used for interactive use and/or visualization
- Login nodes - small number of nodes reserved for login, compiling, editing and other non-production work
- Server nodes - dedicated for file serving, primarily to the GPFS parallel file systems

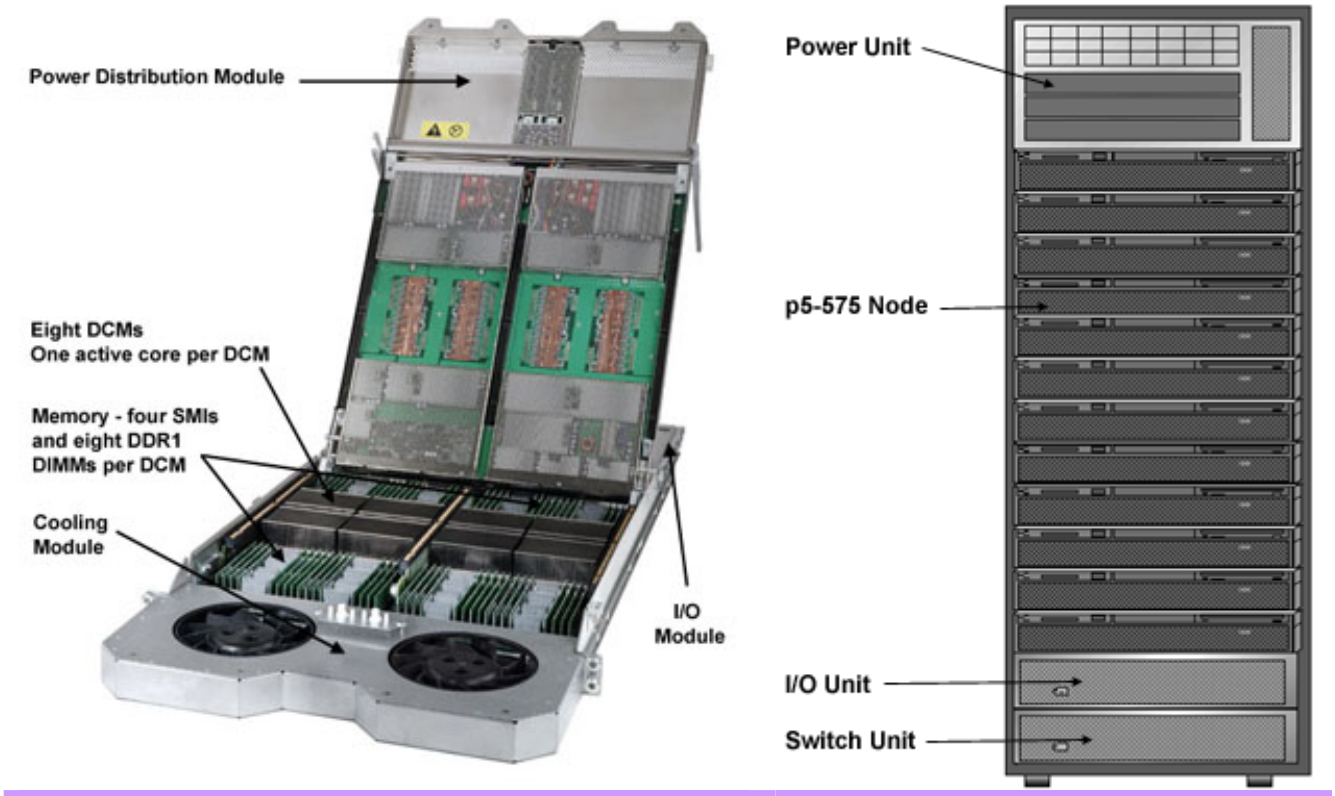
The primary differences between Purple and uP arise from their difference in size. Purple is approximately 14 times larger than uP, and is therefore comprised of more nodes, more disk,

more networking and a larger interconnect switch. The table below compares the two systems based upon these size differences.

Configuration Description	Purple	uP
Number of nodes	1532	108
Number of CPUs	12288	864
Batch compute nodes	1336	99
Interactive / visualization nodes	64	1
Login nodes	4	1
Server nodes	128	6
Theoretical peak performance (Tflop)	93.4	6.6
Memory per compute node (GB)	32	32
GPFS file systems (TB)	2000	140
Levels of HPS interconnect	3	2

Table 1: Purple and uP Configuration Comparisons

The heart of a Purple system is the p5 575 node (shown below). The p5 575 node is IBM's high-density, high-performance POWER5 node designed especially for scientific and technical HPC systems. Its special low form factor design allows up to 12 nodes (96 CPUs) to be housed in a frame for a smaller footprint and higher density than can be achieved with other types of POWER5 nodes. The p5 575 also incorporates simplicity and serviceability into its design. All four of its primary modules (power, I/O, cooling and CPU/memory) are field replaceable. Expansion for additional I/O and network (PCI-X) components is accomplished by connecting p5 575 nodes to an external I/O drawer that may also be mounted in the same frame.



p5 575 Node **p5 575 Frame**

Figure 3: p5 575 Node and p5 575 Frame With Primary Components Labeled

The p5 575 node is different from most other POWER5 nodes because it uses a Dual Chip Module (DCM) comprised of one L3 cache and two CPUs, of which one is inactive. This design allows for dedication of the L3 cache to a single CPU for better memory-bandwidth performance. Most other POWER5 nodes uses a Multi Chip Module (MCM) comprised of four L3 caches and eight CPUs, all of which are active. The logical layout of a DCM node is shown below.

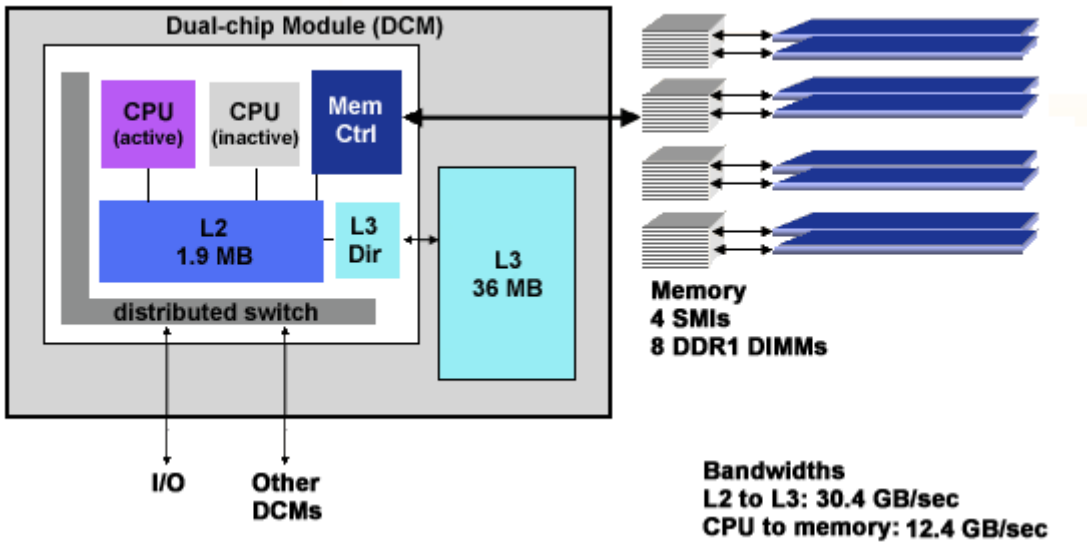


Figure 4: POWER5 Dual-chip Module With Primary Components Labeled

1.2. Learning About the System

For the purpose of using the ASC Purple systems, LC provides complete documentation covering virtually all areas of interest to users, including policy information, machine configurations, system status and usage, code development, software, tools, news, manuals, tutorials, and training. This information provides both high level overviews and detailed descriptions of how to conduct work in the Purple computational environment. Most of this documentation is web based, however many documents are also located in directories mounted on Purple systems, such as /usr/local/docs.

Policy information is clearly specified in the policy documents located on the http://www.llnl.gov/computing/hpc/accounts/#policies1 web page. Policies include:

- Access to LLNL Unclassified ASC Systems by ASC Alliance and DOE Sites
- Access to LLNL Centralized Unclassified Computer Systems by LLNL Employees and Collaborators
- Computer Security Plan and other requirements for foreign nationals
- LC purge policies and file quotas
- Livermore Computing Computer Security Briefing
- Livermore Computing Policies and Procedures
- Code of Conduct for Classified Computer Users at LLNL
- Access Protocols for ASC "Most Capable" Systems
- Capability Compute System Scheduling Governance Model

These policy documents may in turn describe additional policies, such as fraud and abuse, export control, good citizenship, resource allocation and job scheduling, software licensing, computer security, etc.

1.2.1. Web-based System Documentation

LC provides a single, well-established portal for all of its production computing platforms, located on the OCF and SCF web at http://www.llnl.gov/computing. From this starting point, users can select from the full range of topics associated with high performance computing at LC, including detailed user information for policies, getting started, systems status, hardware and software environment, code development, running jobs, data transfer, visualization and technical assistance contacts. Information on obtaining user accounts and accessing the system can also be obtained. Navigation is intuitive and allows users to progressively dive deeper for additional detail and related information. A good example of such navigation is found in the "OCF Machine Status" page, which starts with one line of machine status information for each platform. Each line of status information sinks to successively deeper detail including MOTD messages, important announcements, scheduled downtimes, system load information (updated every 5 minutes), job limits, purge policies and links to detailed machine configuration information. Search functionality is integrated into LC's web pages at three levels: LC portal-wide, ASC page-wide, and LLNL-wide. Information specific to a given platform and supporting platforms is provided. For ASC Purple this platform specific information is centrally located on the web at http://www.llnl.gov/asc/platforms/purple/ and also available in the Purple tutorial at http://www.llnl.gov/computing/tutorials/purple.

These web pages conform to the ASC web page standards insofar as such standards have been documented, and where applicable, to LLNL and LC web page standards. Web pages are maintained current and regularly updated and display the most recent revision date.

Frequently asked questions (FAQ's) to address and answer MPI issues are located at http://www.llnl.gov/computing/mpi/faq.html .

Users are encouraged to provide feedback by means of a "Customer Services Survey Form" located at http://www.llnl.gov/computing/hpc. LC also conducts occasional surveys and user interviews in order to better understand and serve its user's needs.

LC's "Good Citizen Policy" is incorporated into the "Access Protocols for ASC Most Capable Systems" document located at http://www.llnl.gov/asci/platforms/white/home/most_capable_protocols.html. It describes the guiding principles and procedures for computing on a Most Capable system:

- Facilitate large jobs
- Provide fair access
- Keep policies simple
- Minimize priority access
- Adapt to unique system characteristics.

1.2.2. On-line System Information

On-line information is maintained on both the OCF and SCF machines in the form of documentation and user manuals located in several directories, such as /usr/local/docs and /usr/global/docs. A current set of man pages are available by typing man 'utility' and the path to the man pages is automatically set when the user is given an account. Most locally developed software and utilities also have man pages. Application project specific information, which may be restricted to group privileges, is located in /usr/gapps or user controlled application related directories. Current news articles are available upon login, and an archive of news articles is kept in /var/news. Information for a particular software product or tool is generally kept in the installation directory for that product/tool, such as /usr/local/tools or /usr/global/tools. All systems, including ASC Purple have man pages installed. All systems also provide a login banner and MOTD.

Finally, if the user invokes a web browser while logged in, the entire LC web portal, either OCF or SCF, is available. Of special interest is the real time and static machine information that is available to the user community on the OCF. Information includes current configurations, planned activities, important announcements, software upgrades, dedicated time, scheduled preventative maintenance, queued and running jobs, and the general state of the machine, i.e. number of cpus running or free.

1.2.3. Written System Documentation

ASC Purple users have a wealth of printable system documentation available covering topics related to all LC platforms, and topics specific to Purple. Documentation common to all platforms can be found in the LC web pages at http://www.llnl.gov/computing/hpc/documentation. The wide range of topics includes everything from manuals on using LC's LCRM batch system, IBM documentation and file transfer tools to math packages. Additional printable documentation includes technical bulletins and viewgraphs from LC Users Meetings. Much of this documentation is already in PDF format, and also located online under /usr/local/docs on each machine. Tutorials on a number of parallel programming topics are also available in the LC web pages at http://www.llnl.gov/computing/hpc/training/index.html#training_materials. Tutorials exist in HTML format, which can easily be printed directly or converted to PDF format for printing.

Purple specific documentation includes usage documents located online, such /usr/local/docs/up.basics. Additionally, a tutorial specifically for Purple users is located in the LC web pages at http://www.llnl.gov/computing/tutorials/purple. The Purple tutorial in particular, is a recommended starting point for new users, and provides a "quick start" for becoming familiar with the system.

1.2.4. Training

Throughout the year, LC offers on-site workshops focusing on parallel programming, parallel tools, and the use of its ASC IBM and Linux cluster systems. Introductory level workshops are intended for new users, with the goals of improving LC user productivity and minimizing the

obstacles typically encountered by new users of such complex systems. Introductory level workshops typically include both lectures and hands-on exercises using the actual machines.

Other workshops are targeted towards more experienced users and can cover a range of topics related to new technologies, performance/programming tools, vendor product training, and other topics as requested by LC users, researchers, and staff. These workshops may or may not include "hands-on" exercise time. Invited trainers from other institutions and vendor companies are common for these topic specific and advanced workshops.

Training/workshop materials are located in the LC web pages at http://www.llnl.gov/computing/hpc/training/index.html#training_materials. Most of these materials are available openly on the web, though some vendor sensitive topics require password authentication. Trainer contact information is typically available within these materials.

LC local workshops are held in the Computation Training Center (CTC). The CTC is usually also involved in the workshop organization, registration and advertising processes. Additional advertising takes place through LC email lists, flyers, technical bulletins, the web portal, login messages, news items, and LLNL-wide publications such as NewsLine and Computation's Bits & Bytes.

Tri-lab, cross-platform (Q, Red Storm and Purple) workshops continue to be offered through collaboration between LC, LANL and Sandia. These workshops can be delivered at any of these three DOE labs, the ASC Alliance sites, or other locations upon request. They typically include information specific to more than one ASC architecture and are taught by HPC trainers from the three labs and sometimes other invited trainers. Like the LC local workshops, these workshops typically include both lectures and hands-on exercises using the actual ASC machines.

1.2.5. Consulting

Users may also learn about any LC system informally by contacting the LC Hotline consulting staff with specific questions. Section 8, which discusses user support, describes this and related activities in more detail.

1.3. Gaining Access to the System

1.3.1. Account and Password Management (Authorization)

A Tri-lab team was formed in the fall of 2005 to establish a new governance model for allocating and scheduling the stockpile stewardship workload on the ASC Capability Systems according to programmatic priority. The "Capability Compute System Scheduling Governance Model" document produced by this team describes the model fully (attached as Appendix B). LLNL's Purple machine is the first Tri-lab system to use this model. The model governs the allocation of capability computing resources for weapons laboratory deliverables that cannot be reasonably attempted on other resources and that merit priority on this class of resource. The process

outlined in this document describes how capability work can be evaluated and allocated resources, while also preserving a high effective utilization of the systems in order to provide the broadest possible benefit to the Program.

This model intends to make effective use of the machine, as far as possible within the context of its two major objectives, both by minimizing idle cycles and by enhancing the probability of productive and useful capability calculations. The two major objectives of this model are:

- Ensure that the capability system resources are allocated on a priority driven basis according to the program mission
- Utilize ASC Capability Systems for performing the large capability jobs for which they were intended

Another secondary objective is to keep the prioritization and allocation processes as simple and effective as possible, in order to help ensure success of the objectives. Considerations include how to organize the workload for submitting proposals and allocating resources, definitions for capability mode computing, how to prioritize the proposals, implementation of the model, and reporting requirements. This governance model provides clear guidance on the authorization approval criteria for getting an account on the Purple machine.

The method for requesting an account on Purple with instructions, criteria, policies, and procedures for can be found online at http://www.llnl.gov/computing/hpc/accounts/. Requesting an account can be done online by accessing the web site at https://www.llnl.gov/lcforms/ and filling out the form LC "Create User Account Form".

Non-LLNL Tri-Lab requests for Purple accounts are made through SARAPE (http://sarape.sandia.gov). SARAPE is a web-based application that allows users within restricted domains (Tri-Labs, NWC, and ASC University Alliance Partners) to request remote accounts for selected resources at each of the Tri-Labs. The LC Hotline account specialists provide a Tri-lab service to process these Purple account requests, with user transparent links to site-local authorization mechanisms.

1.3.2. Gaining Access to the Machine (Authentication)

As with other LC production systems, access to Purple systems differs between the OCF and SCF, and whether access is coming from within LLNL or from outside LLNL. Access procedures are summarized below. A complete discussion with examples can be found on the LC web page located at: http://www.llnl.gov/computing/hpc/access/remote_access.html.

1.3.2.1. Unclassified System Access

Authorized unclassified users can access uP from inside LLNL's internal restricted network or from outside LLNL over the Internet; the process for each is different. From within the LLNL restricted network, users are required to use an SSH client (version 2 compatible) to connect to uP. After connecting, a two-factor password authentication is required using a static PIN and a

random 6-digit One-time Password (OTP) generated by an RSA SecurID token. Users may increase the security of the authentication process if they desire by configuring SSH to prompt for an additional passphrase. Users may also minimize the login process by configuring SSH to permit password-less login.

If access is coming from outside LLNL, other procedures are followed, depending upon the situation. If the user is located at LANL or Sandia, they may first login to their local machine and then issue the kinit command to obtain Kerberos credentials. Because LANL and Sandia credentials are accepted by uP, they may then use SSH to login without the need for the two-factor OTP authentication. Without local Kerberos credentials, LANL and Sandia users will need to use the OTP password authentication.

Other access originating from outside LLNL over the Internet, such as from another institution, home, hotel, etc., requires using one of LLNL's Internet Access Services. Several different services are offered, including Virtual Private Network (VPN, VPN-C), Open Terminal Server (OTS), Integrated Service Digital Network (ISDN), Web Proxy Service (WPS, WPS-C) and IP Port Allow (IPA). IPA will be discontinued when SSL VPN is available and working on dual boot Linux boxes. After obtaining an account for one of these services, users connect to the service and authenticate using the two-factor OTP password procedure. After a successful authentication to the service, SSH is then used to connect to uP.

Internet Access Services are fully described on the LLNL Open LabNet web page located at: https://access.llnl.gov.

1.3.2.2. Classified System Access

Access to the classified Purple system may originate from within the LLNL classified network, or from outside LLNL over SecureNet. From within LLNL, SSH (version 2 compatible) must be used to connect to Purple. Authentication may be accomplished by means of the two-factor OTP method, or a static DCE password. Access from within the LANL or Sandia classified network also requires SSH. If the LANL or Sandia user logs into a local machine and then authenticates locally with the kinit command, they may then SSH to Purple without the need to enter a password. Other DOE sites may access Purple over SecureNet using SSH and either OTP or DCE authentication.

1.3.2.3. Other Types of Authentication

Users are automatically authenticated when running batch jobs and cron jobs.

1.3.3. System Availability and Scheduling Information

LC provides system availability and scheduling information for all of its production systems by several methods. On the SCF, users can access the LC home page (www.llnl.gov/computing)

and select the "SCF Machine Status" link. Note: we are in the process of determining whether the machine status page requires OTP authentication. Since this is on the classified network, we may allow open access. Users are able to view dynamically updated (every 5 minutes) status information for any SCF machine, including:

- Whether the machine is up or down
- Number of users
- CPU load
- Number of days up
- Last update stamp
- Number of nodes used/free/other
- Job queues
- Where each job is running (on which nodes)
- MOTD
- Announcements
- Job/queue resources and limits
- Purge policy
- Links to machine specific configuration information

The LC home page also provides an "Important Notices and News" page that posts timely information regarding any number of topics including machine maintenance schedules.

Machine status information is available for LC's classified system but users may need to authenticate to access specific user and code name information. Machine status information that has been stripped of sensitive information is available on the unauthenticated web page.

In addition to web based information, LC provides a unique email status list for every machine. Users can subscribe to any list, and will receive notification of important events for that machine. Email status lists are delivered on the OCF, but cover both OCF and SCF machines.

1.3.4. Requests for Priority Jobs

Allocation and scheduling of jobs is handled differently on classified Purple and unclassified Purple. Classified Purple jobs are allocated and scheduled according the Capability Computing Campaign (CCC) policy. The CCC policy is integrated into the Capability Compute System Scheduling Governance Model described previously and included in its entirety as Appendix B. Please consult this appendix for a complete discussion.

CCC proposals may be submitted from both the classified and unclassified LC website (http://www.llnl.gov/computing). This form, along with associated information is hosted on both the SCF as well as the OCF because some users may need to submit classified information on their form to justify their request.

Unclassified Purple provides for expedited priority runs according to LC's Dedicated Application Time (DAT) procedure. DATs are requested through LC's online form

(https://www.llnl.gov/lcforms/ASC_dat_form.html), and typically occur on weekends. DATs are frequently requested by the ASC Alliances.

2. Setting Up the Work Environment

2.1. File System Standards and Documentation

LC employs a consistent file system directory structure and naming conventions across all of its production platforms, including Purple and uP. To the extent that ACE standards for path naming conventions exist and are documented, the standards are met in this directory structure. Examples of this are listed below. Note that the character "?" indicates a wildcard substitution, depending on the situation. Note also that this list is not all inclusive.

File System / Directory	Purpose
/usr/local/docs	User documentation
/g/g?/username	Home directory
/g/g?/username/.snapshot	Home directory online backup
/nfs/tmp	NFS mounted global scratch space
/usr/tmp, /var/tmp	Locally mounted scratch space
/p/gscratch?	Parallel GPFS file system
/p/lscratch?	Parallel Lustre file system
/p/gscratch?/username	User directory in GPFS parallel file system
/p/lscratch?/username	User directory in Lustre parallel file system
/p/gscratch?/purgelogs; /p/lscratch?/purgelogs	Purge log directory
/usr/gapps	User project directories
/usr/local/tools	Local versions of tools
/usr/global/tools	Global versions of tools
/storage	FTP host for HPSS storage
/fis	FTP host for File Interchange System

Table 2: Example Livermore Computing File System Naming Conventions

Documentation for LC file systems, including backup, quota and purge policies, may be found in a number of locations starting from the LC Home page at http://www.llnl.gov/computing:

- EZFILES and Common Home Reference manuals under Documentation
- File Systems / File Management under Computing Resources
- Introduction to Livermore Computing Resources tutorial under Training

Online manuals and files located under /usr/local/docs also serve as documentation.

2.2. Setting up User Groups, Environment Variables, Modules, etc.

The Purple user environment is similar to other LC production systems. User accounts are setup with a complete set of default, standardized startup scripts (dot files) which cover all LC hosts. The master scripts (.profile, .cshrc, .login, etc.) serve to initialize the user environment across all LC machines. Supported shells include csh, tcsh, sh, ksh and bash. Machine specific scripts (.cshrc.white, .login.linux, .profile.compaq, etc.) are customized for a particular platform. These

scripts combine to setup the user environment with necessary environment variables, paths, aliases, standard host type environment variables, etc. Users can modify any of these files, and also, obtain fresh copies at any time from the standard locations of /gadmin/etc/<*arch*>/skel (for platform specific files) and /gadmin/etc/skel directory for the master files. Only a single copy of each startup file is needed, since user home directories are global and portable across all LC machines. Initialization files for applications and tools are also provided as needed. Currently LC does not employ the use of modules for these or other purposes.

Documentation for these files and their usage is available starting from the LC Home page at http://www.llnl.gov/computing:

- EZFILES and Common Home Reference manuals under Documentation
- File Systems / File Management under Computing Resources
- Introduction to Livermore Computing Resources tutorial under Training

User groups are managed by the LC hotline account specialists. Group creation originates when a request to create a new group is received by the hotline from an authorized LC coordinator. Once a valid group is established, users may then be added to this group. Assignment of the default user group occurs when the user account is first setup and entered into the LC Account Management System (LCAMS). In order to promote security, the user default group is the same as the userid and may not be changed. Other changes to user group information are managed through a process that originates with a request form having the necessary authorization signatures. The “Change, Update, or Delete Group” form available from the LC Home Page (under “Accounts, Allocations, Policies, Forms”) is usually used for this. Approved requests are then entered and tracked via LCAMS. Users and group owners are not permitted to manage information in the LCAMS systems; only authorized LC staff may do so. Access to export controlled file systems is accomplished by means of an "xcontrol" group, which contains only those users authorized for access to export controlled codes.

Within LC, there is single UID space, with plans underway to implement a single GID space as well. Within LLNL, there is interest in establishing a lab-wide UID/GID space, though the timeframe for implementing such is not clear at this time. Tri-lab-wide UID/GID space does not currently exist. Hierarchical groups are not supported by LC, and there are no plans to support them in the future. The limit on the number of groups a user may belong to is a function of several factors. On Purple platforms, group membership limits are not a concern for most file systems, however a few commonly used file systems do have a 16 group limit at this time.

3. I/O and Data Migration

ASC efforts are directed at developing and providing tools and services that enable Distance Computing. A goal of LLNL, working closely with the previous DisCom program element, is to provide adequate tools and high-performance throughput for both local and remote users. High performance data movement over the Wide Area Network (WAN) requires common solutions based on the DisCom network architecture. Throughput issues between sites can be resolved only through coordination and cooperation among the sites.

The DisCom WAN was re-competed and awarded to Qwest in early 2006, for implementation by mid-CY2006. The new WAN has increased bandwidth to 10 Gbps from the former 2.5Gbps, and deploys redundant bandwidth between the Tri-labs. Redundant bandwidth incorporates a "north" route (through Denver) in addition to the current "south" route (see map below), and aids in mitigating some single points of failure in the current topology. The new WAN uses an IP over Ethernet protocol instead of the former IP over ATM protocol.

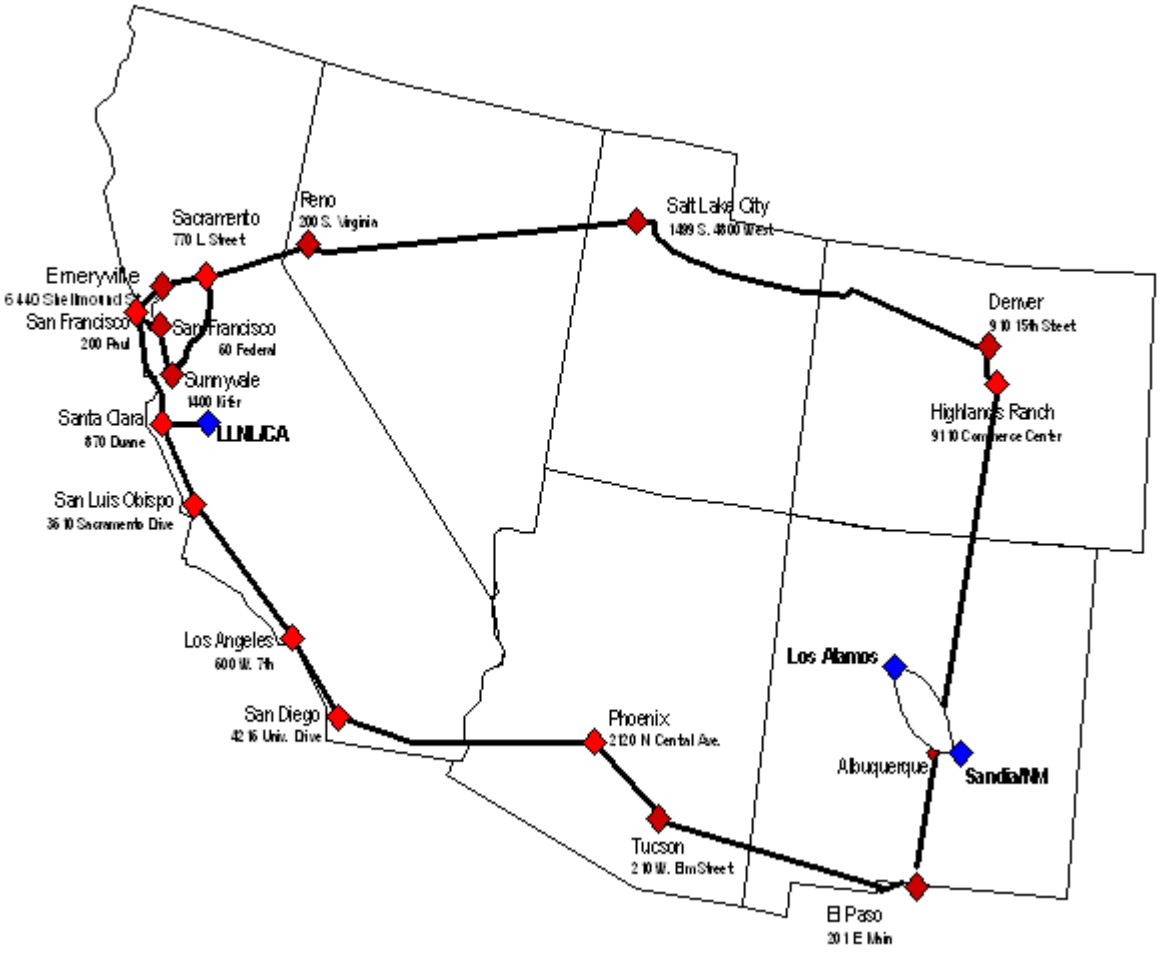


Figure 5: DisCom WAN 2006

3.1. Tools for Transferring Files

On all LC production machines, both OCF and SCF, a parallel FTP client (PFTP) is now the default and is automatically used instead of standard FTP when the ftp command is invoked. Additional infrastructure support for high speed bulk data transfers includes jumbo frame gigabit Ethernet links on most LC production machines, HPSS storage, and visualization servers. PFTP is used to transfer data between two different machines and between a machine and HPSS storage, making inter-LC data transfers optimal.

The ASC Tri-lab community also supports PFTP for high performance parallel data transfer over the classified network. PFTP servers exist at all three labs, and scripts are provided to transfer files between selected classified ASC Tri-lab machines. Interactive authentication is not permitted (e.g., entering a user name and password). Instead, PFTP automatically performs the authentication if a valid Kerberos credential is available. These scripts are installed in /usr/local/bin and include over a dozen commands designed to facilitate data transfer between specific Tri-lab systems and/or HPSS storage. The basic script is pftp, with other scripts being provided for convenience, named so as to describe a specific exchange. For example, pftp2lanl connects to LANL's HPSS, pftp2redstorm connects to Sandia's redstorm machine, etc.

Unclassified FTP between the Tri-lab sites is largely unsupported due to firewalls and other network restrictions. Secure Copy (SCP) can be used to copy files between hosts on a network. Anonymous FTP can also be used.

In addition to parallel FTP, LC also supports several other file transfer tools, primarily designed for use within LC. Hopper is a powerful, interactive, cross-platform tool that allows users to transfer and manipulate files and directories by means of a graphical user interface. Users can connect to and manage resources using most of the major file transfer protocols, including FTP, SFTP, SSH, NFT, and HTAR. On the classified network, Hopper is able to use parallel FTP as the transfer protocol. The Hopper web page is located at: www.llnl.gov/hopper. NFT (Network File Transfer) is LC's utility for persistent file transfer with job tracking. This is a command line utility that assumes transfers with storage and has a specific syntax. Documentation is available via its man page and the NFT tutorial located at http://www.llnl.gov/computing/tutorials/nft/NFTclass.pdf. HTAR is a highly optimized tool for creation of archive files directly into HPSS, without having to go through the intermediate step of first creating the archive file on local disk storage, and then copying the archive file to HPSS via some other process such as ftp. The program uses multiple threads and a sophisticated buffering scheme in order to package member files into in-memory buffers, while making use of the high-speed network striping capabilities of HPSS. The HTAR man page provides usage information. SCP (secure copy) is fully supported on all production machines also.

Shared local file systems that are Tri-lab accessible over the WAN have been provided in the past through DFS. However, since DFS is being phased out, NFSv4 may be deployed after FY06 to provide functionality similar to DFS across the Tri-lab environment.

3.2. Staging Data to the Machine (source code, input decks, data, etc.)

Users have several options for data placement/staging. Source code, input decks and other data are often placed in one's home directory. User home directories are large (16 GB per user), safe from purging, backed up regularly and additionally, provide an online backup directory (.snapshot) which allows file recovery without the need for contacting a system administrator or the hotline. Other available data locations include large, globally shared, NFS mounted scratch directories (/nfs/tmp0, /nfs/tmp1, etc.); smaller, SCSI mounted local directories (/var/tmp) and large, parallel file systems, local to each system and shared by other users. These locations are not recommended for long term storage, or for valuable data that is not replicated and backed up elsewhere, since they are all subject to purging, are not backed up, and are meant as temporary storage by design. Additionally, application developers can use the /usr/gapps file system for their projects. This file system permits group and world sharing based upon Unix permissions, and provides a safe (backed up), permanent location for project source code, libraries, input decks, documentation and any other type of data related to the project.

3.3. Archival Storage Policies

HPSS is the de facto standard for archival storage within the HPC community. It provides a scalable parallel storage system for massively parallel computers as well as traditional supercomputers and workstation clusters. Livermore Computing has been actively involved in the collaborative project to develop the High Performance Storage System (HPSS) for a number of years now, with the ASC program's extreme archival requirements being the primary driver for the HPSS development effort.

LC employs HPSS to meet the high end storage system and data management requirements of its users. All LC production systems, both OCF and SCF, are connected via gigabit Ethernet to an HPSS system (see diagram below).

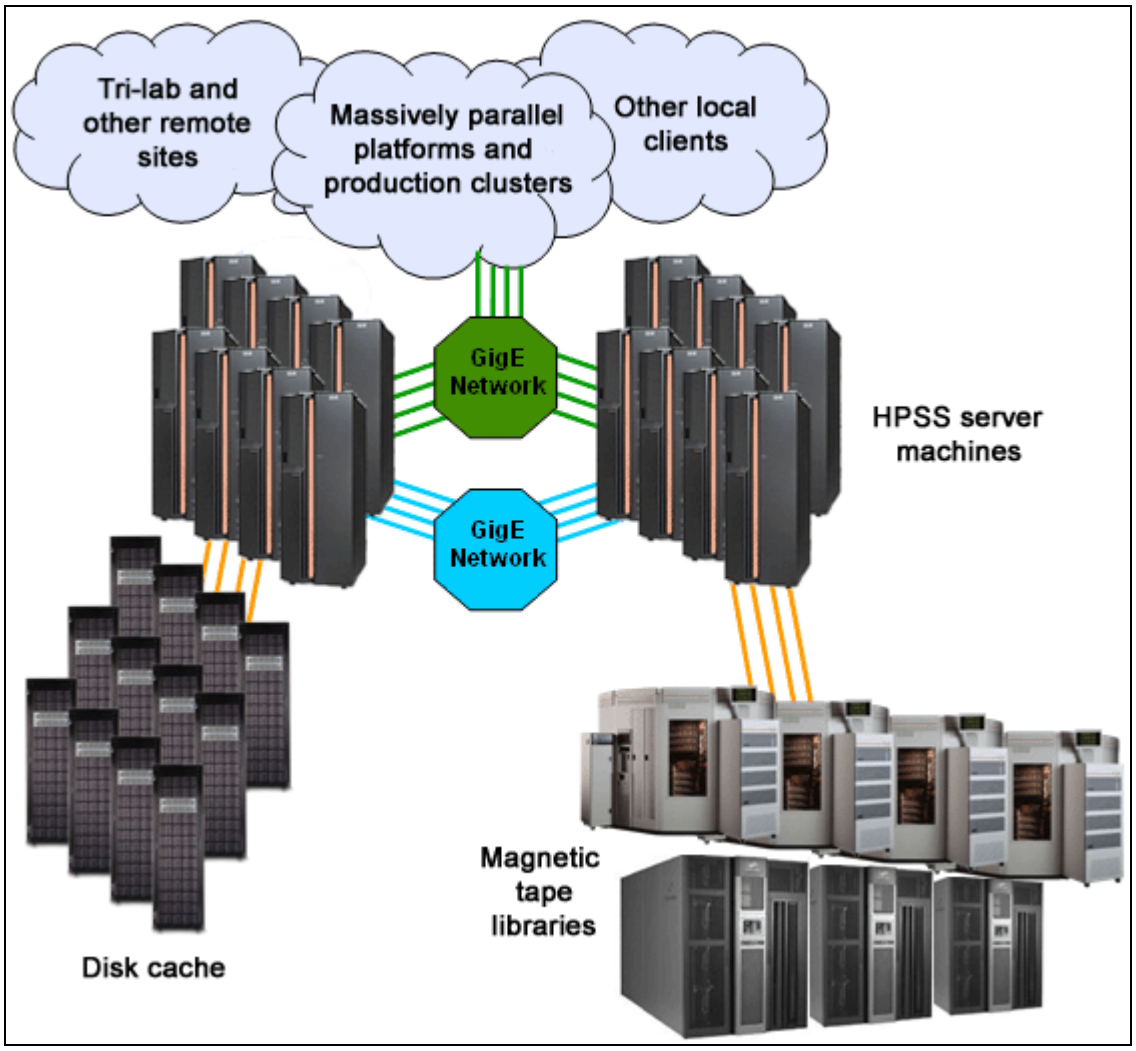


Figure 6: Livermore Computing HPSS Configuration Schematic

The OCF and SCF configurations are similar with respect to the types of equipment being used, though the number of devices may differ. Both capacity and performance are continually increasing to keep pace with ever increasing user needs and usage. For example, aggregate throughput has increased approx. 285-fold between 1999 and 2005, going from 9 MB/s to 2,573 MB/s. Single file transfer performance is currently 300 MB/s and the archive's capacity currently stands at 34 petabytes (combined OCF/SCF). OCF and SCF HPSS systems also employ the same HPSS software, and therefore provide an identical user interface.

From within LC, access to HPSS has been made as convenient as possible. Users simply need to use the "ftp storage" command and they will be automatically logged into the HPSS system. The usual FTP commands can then be used to transfer data to/from HPSS. Data transfers default to the parallel FTP client for maximum performance. Other methods for HPSS access include the Hopper, NFT and HTAR tools described previously in section 3.1. These alternate methods provide several advantages over the basic ftp method of access. Hopper can connect to storage by several different protocols, including NFT, which offers persistent file transfer with tracking. Hopper also provides a friendly graphical user interface that allows users to manage files and file

transfers with more flexibility. Directories can be searched by file name, access permissions changed, and "bulk" operations (delete, move, etc) can be performed on a group of selected files.

Access to HPSS from Sandia or LANL over the classified network has been facilitated with Tri-lab common PFTP scripts, also described in section 3.1.

HPSS security is based upon standard Unix security mechanisms (owner, group, world - rwx).

3.4. Addressing I/O: Effective Use of the File Systems, Serial and Parallel I/O

As discussed in section 2.1, Purple systems share the same file system architecture and usage as other LC production system. The commonly accessed user file systems are each dedicated for a specific purpose as shown in the section 2.1 table. The characteristics and intended usage of these file systems are extensively documented within the LC web pages so that users may easily comply with conventions and benefit from optimum usage.

Most user file systems are NFS mounted and suitable only for serial I/O. Parallel I/O is specifically intended for the parallel file systems attached to a given compute platform. For Purple systems, GPFS file systems are designated for all application parallel I/O. GPFS is a mature IBM product, and has been used on all LC ASC IBM systems prior to Purple also. Its primary features include:

- Scalable high-performance parallel file system for AIX and Linux clusters
- Capable of supporting multi-terabytes of storage and over 1000 disks within a single file system
- Shared-disk file system can provide every cluster node with concurrent read/write access to a single file
- Provides high-performance I/O by "striping" blocks of data from individual files across multiple disks (on multiple storage devices) and reading/writing these blocks in parallel. In addition, it can read or write large blocks of data in a single I/O operation, thereby minimizing overhead.
- High reliability/availability through redundant pathing and automatic recovery from node and disk failures
- A familiar user interface that is identical to any other Unix file system, with all of the usual file commands (ls, df, cp, rm, etc.) behaving as expected.

GPFS is well documented by IBM. Currently, a good place to access IBM's GPFS documentation is from the main GPFS page located at: http://www-03.ibm.com/servers/eserver/clusters/software/gpfs.html. Searching for GPFS on http://ibm.com will also disclose additional related documentation. Usage information and recommendations are able to be found by searching the LC web pages.

At LC, a GPFS file system is local to a single platform and is not shared between different platforms. The latest version of GPFS supports cross mounting between different clusters for shared GPFS file systems and there are now plans in place to implement such at LC. File system

mounting is accomplished over the switch so that compute nodes have the highest possible bandwidth with server nodes, which are connected by optical fiber to the actual storage disks. Purple systems employ two types of disks in their GPFS file systems. Commodity SATA disks (configured into RAID arrays) are used for serving data to applications and FC2 disks are used for metadata operations. To ensure maximum availability, reliability, and performance, the I/O subsystem is architected to have redundant paths to every RAID array. To that end, the GPFS block servers are arranged in groups of four so that if one fails, the other three can take over and serve up the SATA blocks from the RAID arrays. Additional details about Purple's GPFS configuration can be found on the Purple home page at: http://www.llnl.gov/asci/platforms/purple/details.html.

Parallel I/O libraries supported on Purple systems include IBM's thread-safe MPI implementation, HDF5 and PNetCDF. IBM's MPI library includes all of MPI-1 and all of MPI-2 except dynamic tasks. Parallel I/O is fully supported by the MPI-IO interface, and is optimized for use with IBM GPFS parallel file system. HDF5, like previous versions of the Hierarchical Data Format software from NCSA, is specifically designed for the efficient storage of large scientific datasets. This most recent version incorporates improvements and support for larger files, parallel I/O and threads. More information can be found at http://hdf.ncsa.uiuc.edu/HDF5/. PNetCDF is a parallel implementation of Unidata's NetCDF (Network Common Data Form) from Argonne National Lab, used for the efficient and portable storage of scientific data. More information can be found at http://www-unix.mcs.anl.gov/parallel-netcdf/ and http://www.unidata.ucar.edu/software/netcdf/.

4. Application and System Code Development

4.1. Gaining Access to a Machine for Code Development

The code development environment at LC, which encompasses Purple systems, offers resources and support for both application and system code development. Resources are comprised of hardware, networks, storage, and software. Purple developers have access to Purple production machines for development, debugging and making smaller production runs (all considered “routine” work) in preparation for capability computing (considered to be “priority” work). Other, non-production machines are also available to facilitate development work and testing. All users have access to the usual file systems (discussed in section 2.1) and mass storage for their development work.

Applications and projects that require their own project-managed storage are provided with directories in the /usr/gapps file system. These ‘project’ directories are created using normal Unix user and group permissions that will allow designated users to manage their own subdirectories. Application teams can use their project directories to install and update their codes.

In addition to the user support discussed in section 8, code developers are supported by LC’s Development Environment Group (DEG). DEG is responsible for installation, maintenance and in-depth support of a variety of software, including compilers, debuggers and memory tools, profilers, trace analyzers, performance analysis tools and other utilities. DEG also collaborates with vendors and third-party software developers to develop better tools, onsite training and a better environment for code developers. Documentation for DEG supported software can be found at http://www.llnl.gov/computing/hpc/code/software_tools.html. In particular, the web page for compilers currently installed on LC Platforms is located at http://www.llnl.gov/asci/platforms/bluepac/CompsAvails.html on the OCF.

4.2. Peculiarities of the System

From a user's perspective, ASC Purple systems are very similar to the previous ASC Blue-Pacific and ASC White systems. The operating system remains IBM's AIX, the primary compilers remain IBM's XL C/C++ and XL Fortran (though GNU is also available), and the parallel software environment remains IBM's Parallel Environment (PE). AIX is an open and standards-based operating system designed to conform to The Open Group's Single UNIX Specification Version 3. The IBM XL compilers are also standards-based. At this time, LC does not track or document any deviations that AIX or the XL compilers may posses from POSIX, ANSI/ISO or IEEE standards. However, interested individuals may search IBM's web site and product documentation for information regarding compliance with various standards. For example, IBM's XL C/C++ and Fortran compiler programming guides address IEEE Floating-point compliance, and also describe conformance of the compilers to the various language standards and what extensions they include. Another example is the IBM web page is currently located at: http://www-03.ibm.com/servers/aix/standards, which addresses standards and AIX. IBM's PE software is not subject to official standards, but instead, follows "de facto" HPC

industry standards, such as MPI-1 and MPI-2. Accordingly, IBM supports the full MPI standard with the exception of Dynamic Tasks in MPI-2.

Otherwise, LC's Purple systems possess only a few peculiar characteristics, and these are peculiar mostly because they are simply different from previous ASC IBM systems at LC. Each is discussed below.

4.2.1. SLURM

Purple users will probably find SLURM to be the most visible difference between Purple systems and previous ASC IBM systems. SLURM (Simple Linux Utility for Resource Management) is LC's locally developed scheduler. SLURM was originally designed for LC's Linux systems, but has now been ported to AIX for use on ASC Purple. Most of the SLURM implementation on Purple is transparent to users because they primarily interact with LCRM; LC's locally developed cross-platform meta-scheduler. LCRM then "talks" to the native SLURM scheduler. More information on SLURM is available at: http://www.llnl.gov/linux/slurm/slurm.html. It's usage on Purple is documented in the "Using ASC Purple" tutorial located at: http://www.llnl.gov/computing/tutorials/purple.

4.2.2. Large Pages

Previous ASC IBM systems at LC used standard AIX pages for memory operations. A standard AIX page is 4 KB. Purple systems are configured with IBM's large page feature, which allows pages to be 16 MB in size. For most applications, significant memory-cpu bandwidth improvements are realized with large pages. The caveat for users is that they must explicitly enable their application for large pages or else risk poor performance. This can be done through a simple, single line command, documented visibly by LC in its online information (such as /usr/local/docs/purple.basics) and on the web (such as in the previously mentioned "Using ASC Purple" tutorial).

4.2.3. Dual Core Compute Chips With Only One Active Core

As discussed in section 1.1, Purple p5-575 nodes are atypical from most POWER5 nodes produced by IBM because they are comprised of a Dual-Chip Module containing two CPUs, of which only one is active. This design is purposed towards scientific and engineering applications, with the intent of increasing memory-CPU bandwidth by dedicating the L3 cache to a single CPU instead of sharing it between two CPUs. To the user, this design peculiarity is completely transparent because p5 575 nodes are documented as having only 8 CPUs (instead of the physical 16 CPUs), and are therefore regarded as being 8 CPU nodes.

4.2.4. Simultaneous Multi-Threading (SMT)

SMT is a combination of POWER5 hardware and AIX software that creates and manages two independent instruction streams (threads) on the same physical CPU. SMT makes one processor appear as two processors. The primary intent of SMT is to improve performance by overlapping instructions so that idle execution units are used. Purple systems have been configured to enable SMT. Taking advantage of its possible performance improvements is transparent to the user, as the instruction streams are scheduled automatically by the AIX operating system.

4.2.5. POE Co-Scheduler

Under normal execution, every process will be interrupted periodically by the operating system in order to allow system daemons and other processes to use the CPU. On multi-CPU nodes, CPU interruptions are not synchronized. Furthermore, CPU interruptions across the nodes of a system are not synchronized. For MPI programs, the non-synchronized CPU interruptions can significantly affect performance, particularly for collective operations: some tasks will be executing while other tasks will be waiting for a CPU being used by system processes. IBM has enabled POE to elevate user task priority and to force system tasks into a common time slice. This is accomplished by the POE co-scheduler daemon. LC has configured this feature so that users automatically and transparently benefit from the co-scheduler's activities.

4.2.6. Remote Direct Memory Access (RDMA)

RDMA is a communications protocol that provides transmission of data from the memory of one computer to the memory of another without involving the CPU, cache or context switches. Since there is no CPU involvement, data transfer can occur in parallel with other system operations. Overlapping computation with communication is one of the major benefits of RDMA. RDMA is implemented in hardware on the network adapter (NIC) with data transmission occurring over the High Performance Switch. Purple systems are configured to permit RDMA, and LC automatically and transparently sets the necessary parameters so that user applications use this protocol during execution. RDMA is not unique to IBM or Purple, but is a well known and common type of protocol used elsewhere in the computing and networking industries.

4.3. Configuration Control

LC users and support staff have the standard Unix environment configuration control utilities available on all production systems, including Purple platforms. These include SCCS, RCS and in particular, the popular and widely used CVS (Concurrent Version Control System). Documentation for these configuration control systems is readily available on the web. Additionally, LC provides the Subversion revision control system. Subversion is built upon CVS and intends to improve upon it. As with the other revision control systems, Subversion documentation is readily available on the web, such as at http://subversion.tigris.org. All of these configuration control systems provide the capability for code developers and support staff to maintain their software without the need for system administrators or any special administrative privileges.

Configuration control for software installed by LC staff also includes regression testing. When new software is installed, such as a new Fortran compiler or MPI library, support staff will test the new software against a suite of codes designated for that purpose.

For major hardware/software installations, LC runs a series of regressions tests that include user applications and benchmarking codes. These are run as a part of the synthetic workload (SWL) suite for the acceptance of the machine. A review team determines whether the results from the regression tests are acceptable for user access and general availability (GA)

Finally, there is an annual security revalidation process for security significant software testing.

4.4. Parallel Programming Models and Run-Time Systems

MPI is supported on Purple through IBM's standard AIX MPI library. This library implements all of MPI-1 and all of MPI-2 with the exception of Chapter 5, Process Creation and Management. IBM's MPI library is thread-safe and able to be used in conjunction with POSIX threads and OpenMP (hybrid parallel programs). MPI libraries which are not thread-safe, such as MPICH and IBM's older signal-based MPI library are not supported on Purple systems. Because of this, users really have only one choice for MPI on Purple. Furthermore, it is not possible for users to select from different versions of IBM's MPI library, as it is part of IBM's Parallel Environment (PE) software suite. Each time PE is upgraded, all previous version software is replaced.

Building MPI applications is typically performed by using IBM's standard compiler scripts, such as mpxlc, mpxlf, mpCC, etc. These scripts are user readable and able to be copied and modified if desired, though very few users do this in practice. The MPI compiler scripts perform the necessary command line parsing, inclusion of required libraries and include files, and then call the native IBM C/C++ or Fortran compilers. Applications can be either 32-bit or 64-bit, but not mixed. Applications built with IBM's MPI can be run either interactively or in batch, with the primary difference being that LC's LCRM batch system has its own method for specifying certain parameters that differs from standard PE usage. Running jobs behave as described in IBM's PE documentation, with PE being responsible for managing the MPI tasks, such as job startup, signal propagation, message passing and job termination.

IBM's AIX provides a POSIX compliant threads package for C/C++ programs. Unlike most other vendors, IBM also provides a Fortran pthreads implementation, even though the POSIX standard for Fortran has never been defined.

Documentation for using MPI, OpenMP and POSIX threads is provided by through LC's web portal and other sources, such as IBM's PE manuals. LC provided documentation includes:

- LC Tutorials: http://www.llnl.gov/computing/tutorials
- MPI Home Page: http://www.llnl.gov/computing/mpi/

Training for MPI, OpenMP and POSIX threads programming is provided on a regular basis. Advanced MPI training is also available on an as-needed basis.

Support for user developed and user maintained parallel libraries and tools is available through LC's /usr/gapps file system, previously discussed in several sections, such as 2.1 and 4.

4.5. Third Party Libraries and Utilities

Third party supported libraries and utilities are managed and supported by the LC applications and systems staff. Information about updates and versions are posted on the web, documented in news items, and sent out by email to specific mail lists. Installers of such libraries and utilities are required to notify the LC hotline by sending email to lc-events@llnl.gov with the details of the installation so the users can be informed by the various methods. Announcements of the updates are also posted on the OCF and SCF web at https://lc.llnl.gov/dfs/www/computing/status/announce.html. New versions of libraries and software are installed on the machine and notices are sent out to users that these new versions are available. At a preannounced time, these tested versions are moved into the standard location and the replaced version is moved to the designated "old" library or utility path. Many commonly used libraries are located or symbolically linked in the standard /usr/lib and /usr/local/lib/ directories. Other libraries, particularly those related to applications or code development teams, will be found in other directories.

The LC hotline is the POC for third party products. They will be the first line of support and can answer most questions. In the event of further escalation of the problem, they will contact the appropriate staff member. Official problem reporting mechanisms are in place to report vendor library and software issues.

4.5.1. Math Libraries (including solvers)

LC provides a 3-tier structure for support of math libraries on all of its production systems. Primary support is provided for the standard math libraries installed by LC on a given platform, and includes installation, maintenance, documentation and consulting. Secondary support is provided for math libraries downloadable from LC's LINMath web site. LINMath (Livermore Interactive Numerical Mathematics software access utility) delivers advice on and descriptions of mathematical library subroutines, as well as the subroutine source files themselves. The LINMath web site uses the standard GAMS (Guide to Available Mathematical Software) hierarchical system for organizing math routines to structure its menu tree. The LINMath web site is located at http://www-r.llnl.gov/icc/linmath. Support for these libraries includes providing source and/or object code and hotline help. Third tier support is provided for all other types of libraries that code development groups or individual users may desire to install, build, maintain and use. This level of support is limited to providing file space and group access control, such as in the previously described /usr/gapps file system. Note that math libraries and solvers developed by various code development teams as LLNL, such as the Center for Applied Scientific Computing (CASC) group, are not included in the discussions here. For more information about

these, users can visit the CASC web pages located at http://www.llnl.gov/CASC/projects.shtml and http://www.llnl.gov/CASC/download/download_home.html.

The table below describes the primary third-party math libraries available on Purple systems. A starting point for documentation on LC's math libraries can be found in the "Mathematical Software Overview" manual located at http://www.llnl.gov/LCdocs/math. Searching the LC home page and IBM ESSL, PESSL and MASS documentation will provide additional sources of information.

Library	Description	Level
libm	Standard C math library from IBM	1
BLAS	Optimized BLAS from IBM	1
ESSL	IBM's Engineering Scientific Subroutine Library. Provides variety of optimized complex mathematical functions, for many different scientific and engineering applications such as Basic Linear Algebra Subroutines (BLAS), linear algebraic equation solvers (includes a few LAPACK routines), subset of LAPACK eigensystem analysis, Fourier Transforms. ESSL provides single and double precision versions for most routines.	1
PESSL	IBM's Parallel Engineering Scientific Subroutine Library. A parallel version of the ESSL library. It is used in conjunction with the serial version of ESSL. It contains number of Parallel BLAS (PBLAS) and ScaLAPACK routines.	1
MASS	Math Acceleration SubSystem. High performance versions of most math intrinsic functions.	1
MATHEMATICA	Provides symbolic computation and complex analysis, as well as 2D and 3D graphics, and programming.	1
FFTW	"Fastest Fourier Transform in the West". A portable, free, high performance DFT library which implements arbitrary radix, serial and multiple dimensional transforms in serial or parallel (MPI and POSIX threads models). One unique aspect of FFTW is its optional use of self-optimizing strategies, whereby subsequent calls become faster by building on previous timings.	1
PMATH	Portable version of the former Cray MATHLIB library with a few extra routines added, for a total of 68 routines. Includes routines for random number generation, maxima/minima, table lookup, statistics, linear algebra, differential equation solvers, root solvers, interpolation and more.	2
PETSc	Portable, Extensible Toolkit for Scientific Computation. A suite of data structures and routines that provide the building blocks for the implementation of large-scale application codes on parallel (and serial) computers.	2
ParMetis	Parallel Graph Partitioning and Sparse Matrix Ordering. An MPI-based parallel library that implements a variety of algorithms for partitioning unstructured graphs, meshes, and for computing fill-reducing orderings of sparse matrices.	2
LAPACK	LAPACK is a comprehensive, well-respected and freely distributed library of software for linear algebra. Based on Level 2 and 3 BLAS, which have been optimized for high performance by many computer manufacturers. Provides routines for solving systems of simultaneous linear equations, least-squares solutions of linear systems of equations, eigenvalue problems, and singular value problems.	2

Table 3: Math Libraries Available on Purple

4.5.2. Networking and Other Libraries

In addition to the math libraries discussed above, LC supports a variety of other libraries common to the C/C++/Fortran HPC environment. These libraries comprise a wide range of functionality. For example, Purple systems include IBM supplied AIX system libraries used for networking and communications, security, system management, X11, Java, national language support, OpenGL, diagnostics, system performance and tuning, user management, etc. Third party libraries include programming (MPI, Pthreads, Perl, Tcl/Tk, Python, shell scripting, etc.) and a range of tools used for profiling, memory analysis, debugging, performance analysis and tuning, and visualization. Additionally, application libraries developed locally by various LLNL groups are available on Purple systems.

4.6. Compilation

Purple compilers, as with the compilers on all LC platforms, are installed, maintained and supported by LC's Development Environment Group (DEG). A complete list of all platform compilers by version, including Purple, is maintained at http://www.llnl.gov/asci/platforms/bluepac/CompsAvails.html. DEG also maintains and supports most other development tools, as documented on the "Supported Software and Computing Tools" web page at http://www.llnl.gov/computing/hpc/code/software_tools.html.

Purple users have a choice of IBM or GNU compilers for C, C++, and Fortran. According to IBM product information, its compilers are "standards-based" and "fully compliant" with standards for C/C++ and Fortran (77, 90 and 95). The IBM Fortran compiler supports standard Fortran77, Fortran90, Fortran95, along with IBM extensions. The IBM C/C++ compiler supports standard C and C++, along with IBM extensions. Both IBM compilers fully support OpenMP. GNU compilers (gcc, g++, g77) are installed and can be used with Pthreads, but the current versions do not support OpenMP. OpenMP support will be available when versions 4.2 are installed in the future.

Recommendations for compiler options are available in the IBM compiler documentation, online files in /usr/local/doc, man pages, and in several LC tutorials, including "Using ASC Purple". Additionally, LC issues compiler recommendations on an as-needed basis through its machine status email lists and technical bulletins.

The usual common scripting languages are all supported on Purple, including perl, python, Tcl/Tk, sh, csh, tcsh, ksh, and expect. Java is supported through IBM's JDK toolkit. Dynamic shared libraries are also fully supported on Purple machines.

4.7. Debugging and Correctness Testing

TotalView, the standard ASC parallel debugger, is available on Purple platforms as with all other LC production platforms. As part of an ongoing collaborative effort with Etnus (TotalView

vendor), LC is striving to enhance and improve TotalView's functionality, usability, and scalability. Of particular note is TotalView's new memory debugging functionality which includes memory leak detection, graphical heap browsing, real-time heap status, pointer debugging, and real-time memory usage. Also of note are improvements made to TotalView's improved scalability on IBM systems, making debugging of large jobs more efficient and practical. Other parallel debuggers offered on Purple systems include IBM's pdbx debugger and Allinea Software's DDT debugger (on a limited / trial basis). Serial codes may also use the standard gdb and dbx debuggers.

Debugging production sized parallel (or serial) jobs on LC platforms is possible because LC permits users to login to batch nodes as long as their job is running on those nodes. Accessing individual nodes, as might be the case when a running batch job needs to be debugged, can be done by specifying the node's actual name. For example:
`ssh purple048.llnl.gov`. Once logged into such a node, it is possible to attach to a running job for debugging purposes, or to start a job with the debugger. Therefore, no dedicated or special resources are required for debugging large jobs on Purple systems. Using TotalView to debug large jobs is possible however, in practice may prove difficult and unwieldy for jobs numbering beyond a thousand processes. The actual upper limit on the number of parallel tasks is determined by IBM's Parallel Environment software. Currently, the maximum number of supported tasks is 8192, which is approximately 75% of the entire batch compute pool of nodes. However, jobs larger than the official IBM 8192 number are routinely run. Improving TotalView's practical scalability for Purple sized systems is an ongoing effort.

For purely interactive debugging of very small, short jobs, the unclassified uP system provides a minimal interactive pool of nodes, but the classified Purple system does not. Instead, classified Purple users may debug their jobs in the batch pool, use any free nodes in the special interactive/batch visualization pool, or use a non-production POWER5 system.

At the current time, LC provides minimal tools for static code analysis (lint) and testing code coverage (gcov).

4.8. Performance Measurement, Analysis and Tuning

As mentioned previously, LC's Development Environment Group maintains and supports most of LC's development tools, as documented on the “Supported Software and Computing Tools” web page at http://www.llnl.gov/computing/hpc/code/software_tools.html. For Purple, these tools include performance measurement, profiling, analysis and tuning tools such as TAU, Vampir/GuideView (VGV), PE Benchmarker, Paraver, IBM's HPC Toolkit, mpiP, mpi_trace, gprof, HPM Toolkit, Xprofiler, PAPI, PMAPI and MPX. Documentation for these tools is linked from the web page. These tools differ greatly in complexity and functionality, ranging from the simple command line driven, flat-text output gprof profiling utility, to the complex, full-featured, cross-platform TAU, VGV and Paraver performance analysis packages. The more complex packages incorporate tracing, profiling and graphical analysis to provide most of the functionality sought after in a complete performance analysis package. These tools can be used for parallel jobs, including MPI, threads and hybrid, profiling at the routine, loop and source line

level, gathering hardware counter statistics (such as cache usage, floating point usage, etc.), and graphically analyzing various performance parameters. LC works with the vendors of these tools and other parties (including Sandia and LANL) to continually enhance and improve their functionality, usability and scalability on large capacity platforms such as Purple.

IBM's Parallel Environment software includes a number of environment variables intended to provide users with options for performance tuning. LC has conducted extensive testing with these environment variables and has documented recommendations for users of Purple systems. These recommendations can be found in /usr/local/docs, LC tutorials, and relevant LC web pages.

4.9. Best Practices

LC is in the process of developing a “best practices” document for Purple. This document will be located on the policies web page http://www.llnl.gov/computing/hpc/accounts/index.html#policies1 as well as in the /usr/local/docs directory. In the interim, users may benefit from various recommendations that appear throughout LC's web pages and in documents located in the /usr/local/docs directory.

5. Problem Setup

Problem setup includes generating meshes, domain decomposition, materials databases and constitutive models, model/assembly management, visualization of meshes, and archiving of solid models. The Purple General Availability user environment provides all of the system resources and network connectivity needed by application developers for mesh generation, domain decomposition, material databases, and application databases. Because problem setup is usually highly specific to an application, LC does not install and support specific packages for this purpose. However, it does provide support for such by code development teams by providing file systems, license server management, compilers and other software tools, networks and machine compute resources.

5.1. Domain Decomposition

Prior to running a simulation on a parallel computer, the simulation must first be decomposed into tasks assigned to different processors. Efficient use of the machine requires that each processor have approximately the same amount of work to accomplish and that the quantity of interprocessor communication remains small. Finding the optimal decomposition can be a difficult, application specific task. LLNL code development teams make use of various software packages, such as Metis and ParMetis, or their own algorithms, to perform domain decomposition within their application. The Purple computational environment provides all of the necessary network, compute, and support software (compilers, libraries, etc.) resources for domain decomposition activities. Because of the very homogenous nature of Purple systems which have compute nodes configured identically for all practical purposes, and because of the static assignment of nodes to parallel jobs, mapping decompositions onto an underlying hardware resource is not supported. However, developers are free to implement dynamic load balancing from within their application, and in fact, this practice is quite common.

6. Running the Application to Solve the Problem

As with the application development environment, the ASC Purple application run-time environment is consistent with other LC production systems. Users will find the expected file systems, environment variables, shells, dot files, compilers, tools, utilities, software and documentation, all discussed previously in this document. Similarly, Purple systems employ the Livermore Computing Resource Management (LCRM) system to manage batch jobs, and as with all previous ASC IBM systems at LC, use IBM's AIX operating system and Parallel Environment for running jobs.

Interactive work on uP is currently limited to 2 nodes, and on Purple, it is restricted to a pool of 64 nodes reserved for ASC visualization work. Therefore, most jobs are run in the LCRM batch system. LCRM is an ongoing project at LC, originating in 1991. Its primary purpose is to allocate computer resources, according to resource delivery goals, for LC's production computer systems. It is the batch system that all LC users use to submit, monitor, and interact with their production computing jobs. LCRM provides all of the necessary batch system tools and utilities for job accounting, submission, monitoring, interaction, queries and termination. LCRM is thoroughly documented via web pages, user guide and reference manuals, and an LCRM tutorial, all found via the LC home page (http://www.llnl.gov/computing).

Classified ASC Purple is designated as a capability resource. Allocation and scheduling of the stockpile stewardship workload on Purple is defined by the "Capability Compute System Scheduling Governance Model" document included as Appendix B.

Noteworthy is that in 3Q06, the Tri-labs agreed upon and accepted a common batch scheduling system, called Moab, from Cluster Resources, Inc. This scheduler will fulfill the goal of a common resource manager across the Tri-Lab. The contract grants ASC use of Moab software, which provides workload management, system accounting, capacity planning, automated failure recovery, virtualization and a host of other capabilities in cluster, grid, and utility computing environments. In addition, the contract also includes collaborative research and development, consulting, 24/7 support and other professional services.

The Moab solution adds significant manageability and optimization to HPC resources, while providing deployment methods that effectively minimize the risk and cost of adoption. Unique Moab capabilities allow it to be transparently deployed with little or no impact on the end-user; these capabilities include system workload, resource, and policy simulation, batch language translation, capacity planning diagnostics, non-intrusive test facilities, and infrastructure stress testing. Moab includes a policy-based workload management and scheduling tool, as well as a graphical cluster administration interface and a web-based end-user job submission and management portal.

At the time of this document, Moab is in the first stages of implementation at LLNL. Current plans are to gradually migrate Moab to all of LC's production systems, including both Purple systems, during 2007-2008.

6.1. Submitting the Job (Local and Remote)

Local job submission for all LC production systems is currently accomplished thru LCRM. The process begins with a user constructed job script that contains LCRM keywords and parameters, shell script commands and optionally, comments. The user job script is then "submitted" to the LCRM system by a simple LCRM command called psub, usually entered on the shell command prompt after logging into the system where the job should run. Psub commands may also be issued "automatically" from within shell scripts, cron jobs or from within another LCRM job script, and can even be issued from an LC machine that is different than the one where the job should run.

Parameters associated with the job, such as which machine to use, how many nodes/processors to use, how long to run, where to send output, etc. are specified by the user in the job command script (or as arguments to the psub command) and enforced by LCRM. In routine usage, LCRM will not allocate more nodes or a greater run-time than permitted by the defined queues. Queues are established by LC management in conjunction with LLNL program managers and users. Queues limits are subject to change and are well documented within the LC home page. They may also be displayed with a simple command after logging in. For example "news job.lim.purple" will display the queue information for Purple.

Classified Purple has two queues, "pbatch" for production and "viz" for visualization and debugging. Unclassified Purple also has two queues used for the same purposes, named "pbatch" and "pdebug".

Job scheduling on classified Purple follows the Capability Compute System Scheduling Governance Model (Appendix B). Site-local resource management tools will provide a means of implementing and enforcing allocations according to the priorities established by this model. Since CCCs will progress at differing rates (due to idea gestation, extended analysis of results, bug searches, etc.), multiple CCCs will be approved to access the machine, effectively alternating use of the machine, in order to assure continued progress for all CCCs and to maximize utilization of the computer.

For unclassified Purple, it is based upon a "Fair Share with Half-Life Decay" algorithm within LCRM. All authorized users are allocated a specific number of "shares" within a "bank". Shares and banks work together to control the overall percentage of the machine resources a user may fairly use in relation to all other users on the system. If a user uses more than their "fair share" of the system resources, their effective priority is lowered, resulting in less service. After a period of time, the reduced service level "decays" until the user's priority returns to normal. Similarly, using fewer resources than allocated favors an increase in the user's priority, resulting in "better" service. Scheduling is dynamic, and may change at any time before a job actually starts to run, depending upon the priorities of any other jobs that may be submitted before execution begins. For this reason, it is difficult, if not impossible, to predict exactly when a normal user job will run. Users are provided with a limited amount of control in determining when a job should run, however. They can specify that a job should not run until another specific job has completed (dependency), or specify a date/time before which the job should not run.

Job turnaround time is a function of several interrelated factors, including job priority, resources requested (number of nodes, length of time), and system workload. Furthermore, because Purple is a capability platform, large jobs are favored over small jobs. In practice however, small, short jobs are often able to be "backfilled" by LCRM into suitable node/time slots for improved turnaround.

As an additional mechanism to maximize utilization of the computer, there will be a *separate* process to gain access to the computer, called Standby: a fair share bank will be given to each laboratory to be managed by that laboratory. Site allocations for the standby resources will be determined by the Capability Executive Committee. Jobs will be submitted using these banks and are free to run unless they are preempted by a CCC calculation. Some opportunity will be given to a Standby job to checkpoint itself; however, the intent is to minimize the frictional effect and frustrations encountered by the CCC team seeking immediate access to capability cycles. Utilization by Standby jobs will not be counted as part of any approved CCC.

6.2. Monitoring Job Status

Users have several ways to monitor their job's status. LCRM provides the pstat command. By default, pstat displays one line of basic information per job. More information can be displayed by using one of pstat's various options, such as the -f option which provides a full page display with over 30 fields of information for a selected job. Other, non-LCRM commands, may also be used to monitor a job, including spjstat, squeue and sinfo. Like pstat, these commands are issued interactively during a login session on the machine of interest. Finally, users can monitor job execution from the LC home page by selecting "OCF Machine Status". The information on these web pages is derived from the pstat command, but also includes an text-based graphical representation of exactly which nodes are being used by each job running.

As with other LC production systems, users can login to batch nodes if, and only if, they have a job running there. This provides an environment that is very similar to interactive computing. In particular, users can initiate a debugging session by starting a debugger such as TotalView and then attaching to a running job. Users can also start other tools and interact with their job for performance analysis, memory use, or other purposes. Additionally, LC provides the run / proxy utilities, which allow a user to remotely connect and interact with a running job.

By default, LCRM handles stderr and stdout by writing each to a unique file, the name of which is typically a concatenation of the job script name and the LCRM assigned job id number. However, users can select their own file names and also have both stdout and stderr go to the same output file. LCRM output files are created while the job runs, and are readable immediately by the user. LCRM's handling of stdin is necessarily different than the way it handles stdout and stderr. Because a batch job is not linked to the user's terminal, stdin is generally "redirected", in the standard Unix fashion, to the executable from within the job command script. For example, a file could be created containing input that would normally come from stdin. Then, within the job script, this input file could be redirected to the executable: myjob < inputfile.

6.3. Stopping the Job

LCRM permits users to terminate queued (non-running) and running jobs with the simple prm command. For a queued job, the effect of the prm command is to remove the job from the waiting queue; for a running job, prm will terminate the job first and then remove it from the running queue. Users may also put a non-running queued job on hold for an indefinite period by using the phold command. By default, the prm acts immediately, however by using the appropriate prm option flags, users can specify the date/time and grace time for a job that should be terminated/removed.

In some cases, users may desire that their job be signaled when it is about to be terminated so they can perform checkpointing and other graceful shutdown and cleanup activities first. LCRM provides a library interface that includes routines that may be used to facilitate such. Users can imbed the relevant library routines in their application to register for specific signals and terminating events. LCRM will then send the requested signals for the requested events to the user program, which is then responsible for capturing the signal (via a user created signal handling routine) and taking the desired upon actions prior to termination. The LCRM library is documented in LC manuals, web pages and man pages.

6.4. Interactive Use

The unclassified Purple system provides a small number (currently 2) of nodes that are dedicated for interactive work. These interactive nodes are provided for quick turnaround time to allow users the ability to run short jobs to accomplish code testing, input validation and verification, interactive debugging sessions, etc. Classified Purple is intended as a capability platform, and provides a limited number (currently 64) of interactive nodes intended only for ASC visualization work. Routine interactive use of the classified Purple system is not permitted except on these reserved nodes. Users can use other "non-production" POWER5 classified systems for interactive work, or run in standby mode on the reserved Purple interactive nodes. Standby mode allows any user to run on the reserved visualization nodes as though they were batch nodes with the caveat that a legitimate visualization job will cause their job to be terminated if there aren't sufficient nodes for the visualization job. In addition to permitting fewer nodes, interactive queue limits are shorter time-wise than batch queue limits.

6.5. Adapting the Job for Expected System Reliability

The HPC Reliability, Availability, and Serviceability (RAS) Tri-Lab working group is a forum for exploring potential areas of collaboration in machine reliability. A main topic of interest for this collaboration is to determine frequency of restart dumps to ensure the successful completion of the jobs while minimizing the amount of time spent on writing the restart dumps. This will be done by analyzing the mean time between failure (MTBF), system reliability, and other hardware and software factors and recommending optimal practices to the users. This effort is currently underway.

For scheduled maintenance and planned system downtime, LC users are notified in a variety of ways as discussed earlier in section 1.3.3. This allows them to plan how they might submit and monitor jobs to avoid unnecessary impacts by planned system unavailability. Unplanned and emergency downtimes are similarly announced with as much advance notice as possible. Nevertheless, experienced users who are concerned about losing results and/or progress for longer running jobs routinely checkpoint their applications for later restart in the event of unexpected service outages. Building checkpoint/restart capability into an application is the responsibility of the developer, and differs because of this.

In addition to checkpointing, LC encourages users to backup essential data, as most file systems, including the large parallel file systems, are not backed up. Unexpected network, disk hardware or other problems have been known to result in permanent loss of user data, something which is easy to prevent by making sure it is backed up to the archival system or copied to a file system that is routinely backed up (such as one's home directory, if possible).

By default, should a batch job be terminated unexpectedly due to a system outage, LC's policy is to automatically restart the job when the system returns to service. Users can elect to override this behavior by simply specifying "no restart" in their job script. In such cases, it will be up to the user to resubmit the job and wait in the queue again before it runs.

Machine reliability history is tracked by LC management; however reliability statistics are not routinely published for the LC user community.

6.6. System Reliability

See the discussion above in section 6.5. Additionally, system administrators are provided with tools, both from vendors and locally developed, to help maintain a robust and reliable computing system. 24x7 operations also help ensure prompt trouble response and system reliability.

7. Processing Simulation Output

Support for visualization and data analysis on the classified Purple system is implemented similar to way these activities are supported on the classified ASC White system. A subset of the entire system's compute nodes will be devoted to a special pool of visualization nodes. In the case of classified Purple, a total of 64 compute nodes will comprise this pool. These nodes are identical to other compute nodes, being IBM p5 575 POWER5 nodes configured with 32 GB of memory, connected to other nodes in the system through the High Performance Switch interconnect, configured with access to the large GPFS parallel I/O file systems and to HPSS archival storage over GigE.

The much smaller unclassified uP system does not provide a special pool of visualization nodes. Instead, the Sphere Linux Cluster may be used for post-processing visualization work. The Sphere cluster is comprised of 84 dual-processor, 2.8 GHz Intel Xeon nodes, configured with 2 GB of memory per node, a high-speed Quadrics interconnect, large Lustre parallel I/O file systems and GigE access to HPSS storage.

For the remainder of this section 7, visualization discussions will be limited to the classified Purple system.

7.1. Prepare Data for Analysis

Because of Purple's large GPFS parallel I/O file systems, simulation data can be written directly from compute nodes and made available to the visualization nodes without the need for intermediate file transfers. In this way, visualization, data analysis, data manipulation and post-processing tools may work the simulation data "in place" as it is produced in real time, and without the need for additional storage or compute systems. Tools for data preparation includes those supported by LC's Information Management and Graphics Group (IMGG) discussed below, and those developed or installed by users themselves.

7.2. Analyze the Results

7.2.1. Visualization

IMGG provides a full range of visualization related support services to LC users. Installation, development, documentation and consulting for visualization software and tools are included as part of IMGG's services. The approximately two dozen visualization tools, graphics libraries and utilities that IMGG supports are documented within their web pages, with the main web page being located at http://www.llnl.gov/icc/sdd/img/graphics_sw.shtml on both the OCF and SCF. Currently, Purple systems support the following visualization/graphics/analysis software:

AVS/Express	blockbuster	cgplot	DX-OpenDX
EnSight	FFMpeg	gnuplot	Grace

IDL	ImageMagick	MeshTV	NCAR
OpenGL	Open Inventor	POV-Ray	SDSC tools
sm tools	Tecplot	TeraScale Browser	VisIt
VRweb (OCF)	xanim	xmgr	xmovie
xv	ParaView *(implementation in progress at this time)*		

Of special interest to Tri-lab researchers, LC currently supports the full featured, parallel EnSight and VisIt visualization tools, and is working with Sandia to implement ParaView on Purple. Additionally, IMMG is involved with the development and support of scientific data management tools used for creating, storing, retrieving, and mining large-scale data sets within an integrated metadata-based environment. As mentioned above, visualization and data analysis tools can work with simulation data "in place", since the compute nodes and visualization nodes share common access to large, GPFS parallel file systems where there is no requirement for data migration.

The Purple "viz" pool is a special interactive 64-node pool, self-policed and dedicated for ASC visualization users, both interactive or batch. Usage and job limits are controlled and tracked through the usual LC batch system/scheduler mechanisms. However, if the nodes are not being used for ASC visualization work, other users are permitted to run jobs on them interactively or in "standby" mode for batch. Standby mode permits non-ASC visualization jobs to run only if no other visualization jobs are running, with the caveat that they will be terminated should an ASC visualization job need to run.

7.3. Hand Off the Results to Another User

Users can transfer files to other users by several methods. One of the easiest ways to do this is to use the "give" command, which is present on all LC production systems. To facilitate ease of use, a file given on any OCF machine can be listed and acquired by the "take" command from any other OCF machine, provided that the users involved have accounts on the respective machines. The same also holds true for SCF machines. Other means of transferring files between users include anonymous ftp, email, and group-shared directories such as those found in /user/gapps.

7.4. Archive the Results of the Simulation

See section 3.3.

8. Tri-lab Coordinated Operational Support

Operational support for ASC Purple accommodates the needs of both local and remote users. This requires integrated hotline support, timely dissemination of operational information (e.g., scheduled and unscheduled machine or WAN downtime), training and documentation that meets the needs of local and remote users, as well as the availability of common Tri-Lab web pages with consistent format and information. This goal can only be accomplished by continued integration and coordination efforts among the Labs. Efforts to achieve this include, but are not limited to, the face-to-face Tri-Lab Coordinated User Support Meetings (three yearly – one hosted by each lab), the Expedited Priority Runs (EPR) video conference meeting (bi-weekly), and the following weekly teleconferences: (1) Tri-Lab Web Group, (2) Tri-Lab/NWC configuration changes, Tri-Lab HPC support (metrics), NWC Group (helpdesk to helpdesk), (3) Tri-Lab Status and Monitoring, and (4) Tri-Lab User Training.

8.1. User Support

User support for Purple systems is largely handled by the LC Hotline, as with all other LC systems. Users can access support by any of four ways: the web, email, phone and walk-in. The LC web pages provide the necessary contact information and instructions on how to obtain support for all of these methods, such as email addresses, phone numbers, Hotline hours and location, etc. The LC Hotline provides local and remote support for its local platforms. It also serves as the "front line" or clearinghouse for local support of other, remote Tri-lab platforms when called upon for such.

Web support includes all of the resources described in section 1.2.1. It also includes the ability to enter and monitor a "trouble ticket" 24x7, by means of the LLNL-wide, web-based Remedy system, which then routes relevant tickets to the LC Hotline. Information about the trouble ticket system can be found at http://gethelp.llnl.gov. The trouble ticket system is only available on the OCF, however it is used to both report and track SCF problems also.

Assistance through email may be requested via lc-hotline@llnl.gov on the OCF and lc-hotline@pop.llnl.gov on the SCF. Phone assistance can be accessed at 925-422-4531. This hotline is be fully staffed from 7:30 AM to 4:45 PM Pacific Time. Q-cleared users can visit the LC Hotline during these hours on the first floor of building 453. All requests are responded to by a team of LC technical consultants or account specialists. Additional services, such as intersite account requests, training, sample codes, code porting, application tutorials, can be found at the LC Home Page 24 hours a day.

For after-hours and weekend support issues, LC operators, system administrators and network staff provide basic on-call user support services.

8.2. Trouble shooting (in-depth consulting)

The LC Hotline provides the majority of user support for all LC platforms. However, difficult and complex problems which require expert-level troubleshooting arise on a regular basis. Also, individual users may require in-depth, one-on-one assistance from time to time. For these type of support issues, the LC Hotline has available a number of other LC staff who are experts in various areas, such as compilers, tools, I/O, visualization, debuggers, hardware issues, networks, batch systems, etc. These same experts are also typically involved in supporting early systems and milepost work.

In most cases, the relevant expert is assigned responsibility for the "trouble ticket". Typically, the expert then works directly with the user and/or the problem until it can be resolved or a workaround solution found. If necessary, the expert will work with vendors or other relevant third parties in an attempt to facilitate their ownership and resolution of the problem. Because most in-depth consulting originates from a trouble ticket, it can be tracked by the usual means through the Hotline's Remedy trouble ticket system, which is used for both OCF and SCF problem reporting.

Appendix A

ACE Version 8 Mappings for the Purple Computational Environment

This appendix maps each ACE requirement to its associated section in this document. Requirements in regular black colored text are fulfilled in the Purple General Availability user environment. Requirements (or portions of requirements) that appear as red text are not fulfilled. Note that some requirements have been modified from the original ACE version 8 document for the purposes of spelling and grammar corrections, more logical placement in a different section, or removed because they are no longer deemed relevant. These modifications are noted where applicable.

1. Getting Started (Learning About the System, Gaining Access, etc.)

1.1 Platform Map

Requirements 1 and 2 moved to section 1.2. Requirement 3 moved to section 4.3

1.2 Learning About the System

1. Shall provide machine policy information. This will include but not be limited to limitations on interactive use. (moved from section 1.1)
2. Shall clearly specify platform security policies. These shall include but not be limited to export control policies. (moved from section 1.1)

1.2.1 Web-based System Documentation

1. Shall provide web-based system information.
2. The web-based information system shall include open, secure, non-password protected pages that are available to the Tri-labs.
3. The web-based information system shall include links to platform-specific web pages.
4. Shall provide current information about the system that is both open and secure.
5. Shall provide web pages that are searchable.
6. Shall provide a single, well-known point of entry for system information. This point of entry shall provide links to policy information.
7. Shall provide system information in a way to allow increasingly more detail as topic areas are explored.
8. Shall provide information about the main compute platform. It shall also provide information about supporting platforms this shall include but not be limited to information about development, visualization and open computing.
9. The ACE system information shall cover utilities for moving data between machines and within a platform group.
10. System information shall be kept current and shall include a last modification date and email address for the personnel responsible for the page content.
11. Shall provide a user feedback system.

12. Shall provide links to application project web pages that provide information about what applications run on what systems. (This requirement may only apply to secure environments given security policies.)
13. Shall maintain list of FAQs.
14. Shall provide access to current system configuration information.
15. Shall publish a "good neighbor" policy to advertise the policy for submitting jobs to the queues.

1.2.2 On-line System Information

1. Shall provide a current set of man pages for each platform to include locally developed software.

1.2.3 Written System Documentation

1. Shall provide a current set of (easily printable) documentation for each platform, especially compilers, libraries, MPI-IO and MPI that are available locally to users.
2. Shall provide a "quickstart" guide for users of new platforms.

1.2.4 Training

1. Shall provide vendor and locally developed training information.
2. Shall provide training materials that are electronically accessible.
3. Shall provide trainer contact information.
4. Shall provide periodic tailored training for remote users at their sites. Targeted to specific groups like end-users and developers.

1.2.5 Consulting

1.3 Gaining Access to the System

1.3.1 Account and Password Management (Authorization)

1. Shall provide the ability to request accounts on-line. This shall include instructions and criteria for getting accounts.
2. Shall provide information about Tri-lab access policies and procedures. This shall cover all ASC computing resources with links to platform specific web pages.
3. Shall provide site-specific access policies and procedures that specify differences from Tri-lab standards. The information system shall also provide links to Tri-lab and platform specific web pages.
4. Shall provide guidance to users concerning which platform they should consider applying for an account based on the machine maturity and their intended usage.
5. Shall provide clear guidance on the authorization approval criteria.
6. Shall provide a Tri-lab service to process account requests, with user transparent links to site-local authorization mechanisms.

1.3.2 Gaining Access to the Machine (Authentication)

1. Shall minimize the number of authentications users are required to make while meeting security requirements.
2. Shall support cross-laboratory honoring of credentials.

3. Automated authentication support for batch jobs (e.g. cron), for nightly builds, regression testing, etc. (including documented solutions)
4. Provide the ability for multiple users to have access control of a common running application.

1.3.3 System Availability and Scheduling Information

1. Shall provide real-time information via the web about system availability and system loading.
2. Shall provide an estimated uptime when the system is unavailable.
3. Shall provide the capability for users to determine the size of a job that can be run.
4. Shall be able to determine the state of the current queue and job limits.
5. Shall support use of lightweight scripts access to real-time availability information. (e.g. Perl, CSH)
6. Shall provide users with current maintenance and upgrade schedules.
7. Shall provide accurate information about resources available to the user, to include disk and node resources (memory, CPU, etc.). It is expected that the system will have removed any transient resources allocated the previous user.
8. Shall limit access to secure job status information.
9. A utility that returns the set of queues (authorized for user) that meet resource requirements (nodes, memory, etc.)
10. Every user should be able to get an interactive login (at least one) (with modules) for compiles or whatever, but there may have to be a limit on the number they can have.
11. Shall offer a single web site to provide information about platform status and availability to the Tri-lab environment. This site shall also provide recommendations about which platforms to avoid due to other commitments.

1.3.4 Requests for Priority Jobs

2. Setting Up the Work Environment

2.1 File System Standards and Documentation

1. Shall use standardized path names for the location of applications and project areas. (e.g. /projects/<project-name>/) (Note: To the extent standards are documented)
2. Shall use configurations for home directories, scratch areas and archival storage that are available and consistent across platforms.
3. Shall provide current documentation on configurations of home directories, scratch space and parallel file systems.
4. File names should implement similar policies. (e.g. scratch, netscratch, vizscratch.)
5. Shall provide appropriate information for all file systems. This shall include but not be limited to back-up policy, quota and purge policy.
6. Shall provide Tri-lab access to common disk space to enable a remote build capability.
7. At least one file system shall be provided backup for source code.

2.2 Setting up User Groups, Environment Variables, Modules, etc.

1. Shall provide a documented and maintained skeleton template cshrc file and others that provides the basics for setting up the user environment.

2. Shall allow users to change the login shell without requiring sys admin intervention. (Note: with caveats.)
3. Shall provide home directories that are portable across multiple architectures
4. Include a standard architecture-specific environment initialization capability. (e.g. cshrc.aix, .cshrc.tru64, etc.)
5. Shall have a standard module capability on all platforms to manage software dependencies.
6. Shall standardize the system environment variables across platforms (to the degree possible) to enable portable user scripts.
7. Shall provide paths to common software that are standardized across platforms. (Note: To the degree that paths are standardized and documented)
8. Provide other initialization scripts (termcap, emacs, vim)
9. Shall provide documented processes that allow tracking additions and deletions to groups.
10. Shall provide a service for group owners to set up and manage groups.
11. Shall provide a user level mechanism to change the default group.
12. Shall provide the capability to set up groups for export control purposes.
13. Shall have a single gid space across the complex to enable imported tar files to map to the correct group.

3. I/O and Data Migration

3.1 Tools for Transferring Files

1. Shall provide the infrastructure to support high speed remote bulk data transfer that can be executed unattended and is resilient.
2. Shall provide tools for high performance and scalable data transfer that can be initiated from either side. (currently pftp)
3. Shall support all combinations of high-speed (parallel) data transfer from local parallel file system/archive to remote parallel file system/archive on both the classified and unclassified networks. This shall include local to local, local to remote, remote to local and remote to remote.
4. Shall provide local files systems that are accessible over the WAN.

3.2 Staging Data to the Machine (source code, input decks, data, etc.)

1. Shall provide system environment components that facilitate development of common procedures for setting up similar types of problems on the platforms. (e.g. scripts, common file movement facilities, etc.)
2. Shall provide a location where libraries and tools associated with an application can be maintained.

3.3 Archival Storage Policies

1. The archive implementation shall provide high reliability. Provide disaster recovery mechanisms for user identified files to attain even higher reliability.
2. The archive implementation shall provide high availability. Provide failover for archive write.

3. Provide archive data management and organizational services (to include audit capabilities), consistent with security policies (e.g. GREP, FIND, LS, Head, Tail, Bulk file management)
4. The archive implementation shall provide data security. To include access control lists, portable across HPSS platforms/sites implementations.
5. The archive implementation shall provide the capacity sufficient to track user needs.
6. The archive implementations shall provide common functionalities.
7. The archive performance (bandwidth delivered to a single file archive) shall track user needs for archival storage as determined by problem size and expectations for time to archive. (e.g. a constant time to archive that tracks problem size).
8. Shall provide a common (across systems and Tri-lab) user interface to the archive.
9. Shall provide utilities to consolidate many small files into a single file. (currently htar).
10. Shall provide a persistent archiving capability with seamless transition across archival systems.

3.4 Addressing I/O: Effective Use of the File Systems, Serial and Parallel I/O

1. Need an API to determine whether a file system is local or global, and serial vs. parallel
2. Shall provide guidance for code developers to make effective use of serial and parallel file systems, including the main file systems primarily to include but not limited to large parallel file system-topology.
3. Shall provide platform specific single global parallel file system visible from all nodes with homogeneous node IO performance to the file system.
4. Shall provide a single parallel file system that spans platforms.
5. Shall provide links to information about the system parallel file system.
6. Shall provide optimized and standard compliant parallel I/O libraries: MPI IO, HDF5, PNetCDF, and UDM
7. Shall allow for the number of concurrent open files (including system files) equal to approximately ten times the number of processes.
8. Shall provide IO libraries that are thread safe with scalable performance.

4. Application and System Code Development

4.1 Gaining Access to a Machine for Code Development

1. Provide resources configured to facilitate code development (system and application.)

4.2 Peculiarities of the System

1. Shall document deviations from POSIX and ANSI/ISO standards.
2. Shall document deviations from compliance with language and run-time standards. (e.g. MPI, I/O, Cray pointers, etc.)
3. Shall publish deviations from IEEE arithmetic.
4. Single location (per platform) (e.g. a web site.) with pointers to commonly known or encountered deviations from industry HPC standards – could be implemented via an updated FAQ as items are identified, via release notes, or a bulletin board

4.3 Configuration Control

1. ASC platform status and configuration information shall be maintained and accessible (both push and pull model) to users. This information includes the current, historical, and planned (1) operating system and system libraries, (2) compiler and embedded, (3) MPI library, (4) other "standard and customary" libraries, and (5) system status e.g., maintenance down time, dedicated mode, etc. Attributes shall identify the entity including name, location, version, and patches; shall include statement regarding backward compatibility with previous versions; shall include the dates planned for delivery, actually validated, planned for withdrawal, and actually withdrawn.
2. Shall provide access to source repository (e.g. SourceForge, Razor, CVS, etc.)
3. Shall provide the capability for developers and support personnel to maintain products without requiring sys admin privileges.
4. Verify the environment by running a set of application based regression tests to exercise changes to the operating system, compilers, libraries, etc. (moved from section 1.1)

4.4 Parallel Programming Models and Run-Time Systems

1. Shall provide a standard compliant MPI library optimized for both on-box and across-box messaging.
2. Shall provide POSIX compliant thread package.
3. Shall provide support for hybrid OpenMP/MPI parallel programs.
4. Shall provide documentation with guidelines for running OpenMP and MPI programs (especially for thread-safe code.) (e.g. how to tune buffer sizes, etc.)
5. Shall make it clear to users which MPI library is loaded, especially when wrappered compiler commands are used.
6. Shall provide resources for the installation and support of locally developed libraries that can be centrally accessed. (e.g. LA-MPI, UPS, HYPRE).
7. Need a mechanism to force a specific version of MPI to be linked into the code
8. Need a standard interrupt propagation mechanism to tasks associated with a parallel application.
9. Consistent integration with the batch system and access to resource availability in order to build the MPI run command.
10. Provide mechanisms in MPI for enhanced reliability – e.g. survive NIC failures.

4.5 Third Party Libraries and Utilities

1. Shall manage changes to center supported third party libraries via a change control procedure.
2. Shall provide information about different versions of libraries that are on the system and guidance to the user as to which one to use including but not limited to MPI.
3. Shall provide system documentation that includes a POC for each center supported third party product on the system.

4.5.1 Math Libraries (including solvers)

1. Shall provide support for standard sparse matrix operations.
2. Shall provide support for parallel sparse linear solvers.
3. Shall provide optimized BLAS and LAPACK.
4. Shall provide system documentation that points to what solvers are installed and the support POC.

4.5.2 Networking and Other Libraries

1. Shall support libraries that are callable from C++. (Note: If important please supply which libraries.)
2. Shall support secure interprocess communications such as sockets, HTTPS, and/or ssh.
3. Shall support the deployment of client/server socket-based parallel tools (including visualization) on platforms.

4.6 Compilation

1. The ACE system information shall provide current documentation for compilers, libraries and development tools.
2. Shall provide user recommendations for recommended compiler options.
3. Shall provide the latest 64-bit versions of the gnu utilities. (e.g. make, g++, gawk, gdb, gawk, etc.).
4. Shall provide standard compliant compilers. C, C++, and FORTRAN9x.
5. Shall provide parallel make and compilation in build.
6. Shall provide information about different compiler versions resident on the system and guidance to the user as to which one to use.
7. Shall provide compiler support for OpenMP. C & OpenMP, C++ & OpenMP, and Fortran9X & OpenMP.
8. Shall provide compiler support for F77 codes.
9. Shall provide JAVA support. (editorial note – this is an attempt to clarify a previous requirement): JAVA – via standard JDK toolkit.
10. Shall provide common scripting languages. (e.g. perl, python, and Tcl/Tk, sh, csh,tcsh,ksh, expect)
11. Shall provide support for dynamic/shared libraries.

4.7 Debugging and Correctness Testing

1. Shall provide the ASC standard parallel debugger. (currently TotalView)
2. Shall provide a usable debugging capability for jobs that span across 1/3 of the system.
3. Shall provide common tools for parallel aware memory checking including user callable routines to track memory use and availability. (e.g. Valgrind type functionality)
4. Shall provide machine resources to be used for code debugging – (this requires dedicated interactive use of the resource - and make it clear to users which resources are to be used for this purpose).
5. Shall provide tools for static analysis of source. (e.g. flint)
6. Shall provide tools that measure testing code coverage for all supported standard languages.

4.8 Performance Measurement, Analysis and Tuning

1. Shall provide a message tracing tool that is usable for jobs that span across 1/3 of the system. (e.g. Vampir).
2. Shall provide a message profiling interface (e.g. mpiP).
3. Shall provide loop level application profiling tools.
4. Shall provide tools to analyze cache performance.

5. Shall provide user level access to hardware performance counters, preferably via the PAPI API.
6. Shall make it clear which tools can be used together. (e.g. compiler and memory tool, or debugger and mpi).
7. Shall provide tools to access thread performance.
8. Shall provide tools to characterize the IO performance of a code.
9. Shall document usage of environment variables and other settings that optimize application performance (NEW)

4.9 Best Practices

1. Shall publish "best practices" for performance.

5. Problem Setup

1. Shall provide resources and connectivity to users for the generation of meshes/netlists/schematics, material databases, and necessary data storage for application data bases.

5.1 Domain Decomposition

1. Shall provide real-time information to support smart domain decomposition to allow near optimal mapping onto the machine, taking into account non-homogeneous system resources.

6. Running the Application to Solve the Problem

6.1 Submitting the Job (Local and Remote)

1. Shall provide users with a common queue syntax and semantics across the tri-lab. (i.e., "large queue" means the same across the tri-lab) (Note: To the extent that a common queue syntax and semantics for the tri-lab exist and are documented.)
2. Shall provide a common user interface to access different underlying resource management system resource information. (Note: To the extent that a common user interface for the tri-lab exists and is documented. To achieve tri-lab common interface – may require change to your scripts.)
3. Shall provide the user the ability to specify the approximate requirements of their job. (e.g. the ability to specify the resources that are needed for the job.)
4. Shall provide the user with a sorted list of resources available to handle their job. (e.g. How many processors are available, In which queue, Estimate of time to start-up, Estimate of run time)
5. Shall make the remote job submittal process the same across the complex. (Note: To the extent that a common job submittal process exists and is documented.)
6. Shall provide resource management mechanisms that support accurate and reasonable time limit and node allocation requirements.
7. Shall support dependent jobs.
8. Shall provide automatic job submission e.g. for regression testing. (e.g. cron)
9. Shall provide job schedulers that place processes so as to optimize use of the platform topology or specialized resources. (e.g. for compilation, I/O, visualization, etc)
10. Need quick turnaround for small regression tests.

6.2 Monitoring Job Status

1. Shall allow users to examine the contents of files being used by an executing job. (with STDIN and STDOUT caveats)
2. Shall be able to attach to a running job to control, steer and monitor it. (e.g. run proxy.) (Note: Users will be able to attach to a running job to steer and monitor with TotalView.)
3. Shall be able to monitor memory and cpu use for load balancing purposes.

6.3 Stopping the Job

1. Shall provide a well-defined signal interface to be sent to all user processes prior to the system going down. The signal shall be sent out with sufficient notice to allow graceful shutdown and cleanup.
2. Shall make it possible for the user to inform the resource management system of the appropriate time delay between signal issue and process termination for their application.
3. Shall allow a user to initiate a kill of a job causing an immediate termination, to include an optional flushing of IO buffers - this capability needs to work in the presence of resource managers. (e.g. the signal propagates through the various layers of daemons to the user processes).

6.4 Interactive Use

1. Shall provide some portion of the system to be available to support timely interactive use.
2. Shall provide dedicated resource allocation for interactive use. (e.g. for computational steering or debugging)

6.5 Adapting the Job for Expected System Reliability

1. Shall provide a history of machine reliability to assist users in determining frequency of restart dumps.
2. Shall provide users with information about planned system down time.
3. Shall provide procedures for “automatic” job restart after system interruption.

6.6 System Reliability

1. Shall provide tools and systems to provide a reliable environment for users. (e.g. lock out unreliable nodes or nodes with other problems)
2. Shall provide the capability to transparently migrate user processes if a node goes down.

7. Processing Simulation Output

7.1 Prepare Data for Analysis

1. Shall provide specialized resources (i.e software and platforms) for the manipulation (e.g. combining) of visualization data - may require more memory than compute nodes, etc.

7.2 Analyze the Results

7.2.1 Visualization

1. Shall document what viz tools are available.
2. Shall establish policies for utilization of compute platform visualization nodes.

3. Shall provide the ability to visualize the data in place without the need to move data to another system.
4. Shall provide computing (hardware) resources to be used for data extraction and visualization as part of an interactive visualization process that does not require the movement of data.
5. Shall provide a full featured scalable parallel visualization tool. This requirement currently met by Ensight and VisIt.
6. Shall provide data analysis capabilities. IDL is the preferred tool due to a link to experimentalists' data.
7. Shall provide visualization from the desktop.

7.3 Hand Off the Results to Another User

1. Shall provide a command to transfer a file to another user. Must be a standard interface and a standard location. (e.g. "give" command)

7.4 Archive the Results of the Simulation

See section 3.3.

8.0 Tri-lab Coordinated Operational Support

8.1 User Support

1. Shall publish and maintain phone numbers, email addresses, and hours of support hotline and help desk.
2. Shall provide contact information for reporting problems outside working hours.
3. Shall provide adequate local support staff that is available during tri-lab working hours to answer user questions either in person or via the telephone (single access number/site.)
4. Shall have the capability for open and secure problem reporting and tracking for each system.
5. In the ACE, the local user support team shall act as a clearinghouse for local user support related to both local and remote platform usage.

8.2 Trouble shooting (in-depth consulting)

1. Shall provide experts to be available to resolve complex system problems in areas such as IO, compilers, visualization, parallel programming, performance, debugging, platform specific issues, etc.
2. Shall provide enhanced system support for early machine problems and milepost calculations.
3. Shall have the capability for open and secure problem reporting and tracking for each system.

Modifications to ACE requirements

Section 4.3
Previous:
Shall provide the ASC standard compliant MPI library (currently 1.2) optimized for both on-box and across-box messaging.
Now:
Shall provide a standard compliant MPI library optimized for both on-box and across-box messaging.
Rationale:
The MPICH library is not thread safe and will not run on Purple. IBM's MPI library is thread safe and complies with the MPI "standard" with the exception of dynamic tasks.

Section 4.5
Previous:
Shall provide compiler support for F77 codes including support for Cray pointers.
Now:
Shall provide compiler support for F77 codes.
Rationale:
Cray pointers are vendor specific and obsolete.

Section 4.8
New requirement added:
Shall document usage of environment variables and other settings that optimize application performance

Appendix B

Capability Compute System Scheduling Governance Model Advanced Simulation and Computing Program

April 20, 2006

Jim Ang, SNL, Brian Carnes, LLNL, Sudip Dosanjh, SNL,
Rick Martineau, LANL, Ken Perry, HQ DSW, Jim Rathkopf, LLNL,
Manuel Vigil, LANL, Cheryl Wampler, LANL/HQ rep

INTRODUCTION

In the fall of 2005, the ASC Program appointed a team to formulate a governance model for allocating resources and scheduling the stockpile stewardship workload on ASC Capability Systems (see Charter, Appendix A). This document describes a model that is devised to allocate capability-computing resources for weapons laboratory deliverables that merit priority on this class of resource and that cannot be reasonably attempted on other resources. The process outlined describes how capability work can be evaluated and approved for resource allocations, while also preserving high effective utilization of the systems. This approach will provide the broadest possible benefit to the Program.

The objectives of this initiative are:

- To ensure that the capability system resources are allocated on a priority-driven basis according to the Program requirements;

- To utilize ASC Capability Systems for the large capability jobs for which they were designed and procured

Within the constraints of meeting the two primary objectives, this model maximizes effective use of the machine both by minimizing idle cycles and by enhancing the probability of productive and useful capability calculations.

An important, but secondary, objective is to simplify the prioritization and allocation processes in order to assure that these do not impede successful attainment of the primary objectives.

This paper authorizes the creation of relevant review bodies, describes the character of work packages, establishes a framework for requesting and reviewing proposals as well as procedures for prioritizing proposals, allocating resources, and collecting relevant data to measure progress both from capability providers and users.

WORKLOAD ORGANIZATION

A capability class system is similar in value and uniqueness to a large experimental facility. For this reason, the process described here to request and review proposals to utilize ASC capability systems is similar to that of experimental facilities, while taking into account that these systems uniquely support the Stockpile Stewardship mission. Major programmatic computing efforts will be organized as computing work packages and will be reviewed and prioritized for relevance, importance and technical rationale. Each proposed work package, called a *Capability Computing Campaign* (CCC), consists of at least one major calculation needing a significant proportion of an ASC capability system, together with related supporting jobs of smaller sizes. The portfolio of CCCs should provide a reasonable balance between the objective to ensure that the resources are successfully applied to the highest programmatic needs and the objective to use ASC capability systems effectively for capability jobs that cannot be run on any other computing resource.

The CCC concept respects the tried-by-time strategy employed for running major computational efforts, in that several smaller calculations, often building up in size, are run in support of one or more large calculations. This approach provides the necessary verification of the calculation methodology to maximize the understanding and value gained from the full-size calculation. In addition, a major calculation is often followed by or accompanied by smaller supporting calculations (coarser mesh, assumed symmetries, physics approximations, etc.) to provide additional insight and explore sensitivities. For example, on a peak 100 TF class system, a reasonable CCC proposal might request several months of access and include a 40 TF main calculation supported by a number of smaller calculations. The proposed CCC would demonstrate that all smaller jobs are essential for and in support of the capability run(s). The allocation process (described below), in turn, will ensure that the technical rationale includes an explanation of why these smaller jobs are impractical to run on capacity resources.

Under the proposed model, there are three regimes of job size relevant for inclusion in a proposed CCC:

- *Category 1* (C1): Capability jobs that use 75% or more of the available nodes on a system. This class of job fully taxes the capability of the machine, and represents the most computationally demanding work run in the complex. Some C1 work will also include scaling studies in preparation for long running calculations on existing and future capability systems. Due to its size, a single job in this category will effectively require dedicated use of the machine, and will require careful scheduling in order to allow other CCCs to advance.
- *Category 2* (C2): Capability jobs that use between 30% and 75% of the available nodes on a system. This class of job will typically consist of large production weapons calculations and performance studies, which depending on their exact sizes, would generally permit two C2 jobs to run on the system simultaneously. This is desirable in order to permit multiple CCCs to advance simultaneously.
- *Category 3* (C3): Jobs that use less than 30% of the available nodes that are part of a CCC. These smaller jobs are essential for carrying out a CCC and can be run only on the capability system. A CCC may not consist of C3 jobs only but must include at least one C1 or C2 job.

This approach will be successful only if sufficient alternative resources are available to carry out the capacity workload of the complex, characterized by tens-of-thousands smaller calculations. If this is not the case, then an alternative utilization scenario must be devised for the capability platforms, as, inevitably, these will be preempted frequently for high-priority capacity calculations.

ALLOCATION

Allocation of capability resources will be achieved using a two-step process. First, a Capability Planning Advisory Committee (CPAC) will review the proposed CCCs and make a recommendation to the Capability Executive Committee (CEC). Second, the CEC will review this recommendation and make the final allocations.

The CPAC will consist of nine representatives, three from each of the three Laboratories appointed by the respective ASC Executive. CPAC members must be capable of representing the National Stockpile Stewardship program priorities and understand the DSW workload. Each Laboratory should include at least one computer-knowledgeable representative among its members. The CPAC will meet on a biannual basis to evaluate proposals and will be available to confer monthly if necessary for course corrections.

The CPAC will define the details of the proposal process, including the format of the request for Proposals and the Response to request for Proposals. It is important that it assure a “light-weight” proposal process for mid-year resubmissions of a continuing project.

After evaluating proposals, the CPAC will prioritize these based on the priorities of the Weapons Program, including relevance to Level 1 and Level 2 milestones, relevance to stockpile (DSW) deliveries, and importance in the progression towards predictability (e.g. removing the “knobs” as elucidated in the ASC Roadmap, in particular when such efforts are closely coordinated with the Campaigns). Since multiple capability platforms may be available to the complex at any given time, the CCCs will be prioritized based upon the aggregate capability available. In addition, the CPAC will be responsible for reviewing the technical rationale to ensure that the CCC is sound, ready to make effective use of the machine and is free of extraneous work. It is important that the review process ensure that the associated codes are validated and run efficiently. An important component of this technical review will be an examination of the proposed C3 class jobs to assess their potential to run on available capacity systems. The CPAC will present the recommended list of prioritized CCCs, directions to the host site(s) for executing these priorities, and the associated machine allocations, to the CEC.

The CEC will consist of the ASC Executive (or designee) from each Laboratory and representatives from the NNSA ASC office. It is also recommended that the CEC include at least one representative from federal DSW management. The CEC will review the prioritized list and allocations sent forward by the CPAC, and will make adjustments as necessary. The CEC will approve the final list of CCCs with associated machine allocations, as well as the list of CCCs approved for execution on targeted platforms for the review period and deliver this information to the sites. Note that in the event that the reviewed and approved CCCs exceed available

resources, the prioritized list will contain CCCs that cannot initially be accommodated immediately. These CCCs will be viewed as waiting for resources that may become available in the event that previously approved CCCs are cancelled or complete ahead of schedule.

Outside of the normal review period, the CEC is empowered at any time to advise the CPAC of CEC-directed changes to CCC approvals, priorities, and allocations to accommodate unanticipated and critically important programmatic work (preempting the normal CCC approval process). The CPAC will forward notification of these changes to the Tri-Lab EPR.

The tri-lab technical Expedited Priority Run (EPR) body will continue to meet weekly to manage emergency situations, report up to the CPAC, and address user issues in the tri-lab community.

It is the responsibility of each site hosting a capability system to implement and enforce CEC directions and allocations according to the approved prioritization. Although the host site will not be responsible for managing the work portfolio itself, it must be aware of the priorities established and coordinate efforts with the submitting site to successfully execute the CCCs in accordance with CEC direction. It is the responsibility of the submitting site to ensure execution of the prioritized CCCs. If the submitting site is not able to use its full allocation within the period of approved access, either due to the failure or early completion of one or more CCC within its portfolio, the hosting site will enable CCC(s) waiting for resources according to CPAC directions, with rapid notification to the CPAC. Complications that arise from this adjustment will be forwarded to the CPAC as well. If these complications result in the need to modify priorities or directions, the CPAC will seek approval for these modifications from the CEC. A CCC will be removed from the approved list when the allocation is exhausted, or the time period for access has passed, or the project team reports that the project is completed, whichever occurs first. If the project needs extra computing time to finish, a request must be made to the CPAC or to the relevant ASC Executive (see 5% provision below) for approval. At subsequent CPAC and CEC meetings, the Laboratories shall report CCC status and utilization.

In the event of urgent but unanticipated, high priority need for capability resources, the CEC has the authority to preempt the resources. In addition, the CEC may invoke a process to “renormalize” the allocations as necessary.

An allocation of 15% of the compute cycles per annum will be explicitly reserved for use at the discretion of the ASC Execs to cover urgent but unanticipated needs not explicitly met through active CCCs. This will be achieved by allocating 5% to each Laboratory. This allocation is intended for capability computing; however it is possible that pressing programmatic work may not meet the stringent Category 1 or 2 criteria, so no conditions will be explicitly imposed to enforce this outcome. Nonetheless, it is expected that Executives will be ready to explain and justify their interventions and choice of platforms at CEC meetings.

In addition, it must be acknowledged that the host site will retain use of the computer as required for system maintenance, upgrades, and software development for upgrades and enhancements of the system.

IMPLEMENTATION

Site-local resource management tools will provide a means of implementing and enforcing allocations according to the priorities established by the CEC. Since CCCs will progress at differing rates (due to idea gestation, extended analysis of results, bug searches, etc.), multiple CCCs will be approved to access the machine, effectively alternating use of the machine, in order to assure continued progress for all CCCs and to maximize utilization of the computer.

As an additional mechanism to maximize utilization of the computer, there will be a *separate* process to gain access to the computer, called Standby: a fair share bank will be given to each laboratory to be managed by that laboratory. Site allocations for the standby resources will be determined by the CEC. Jobs will be submitted using these banks and are free to run unless they are preempted by a CCC calculation. Some opportunity will be given to a Standby job to checkpoint itself; however, the intent is to minimize the frictional effect and frustrations encountered by the CCC team seeking immediate access to capability cycles. Utilization by Standby jobs will not be counted as part of any approved CCC. An example illustrating the envisioned utilization of the machine when in full capability mode is to be found in Figure 1, showing approved CCCs in various Categories and the Standby load.

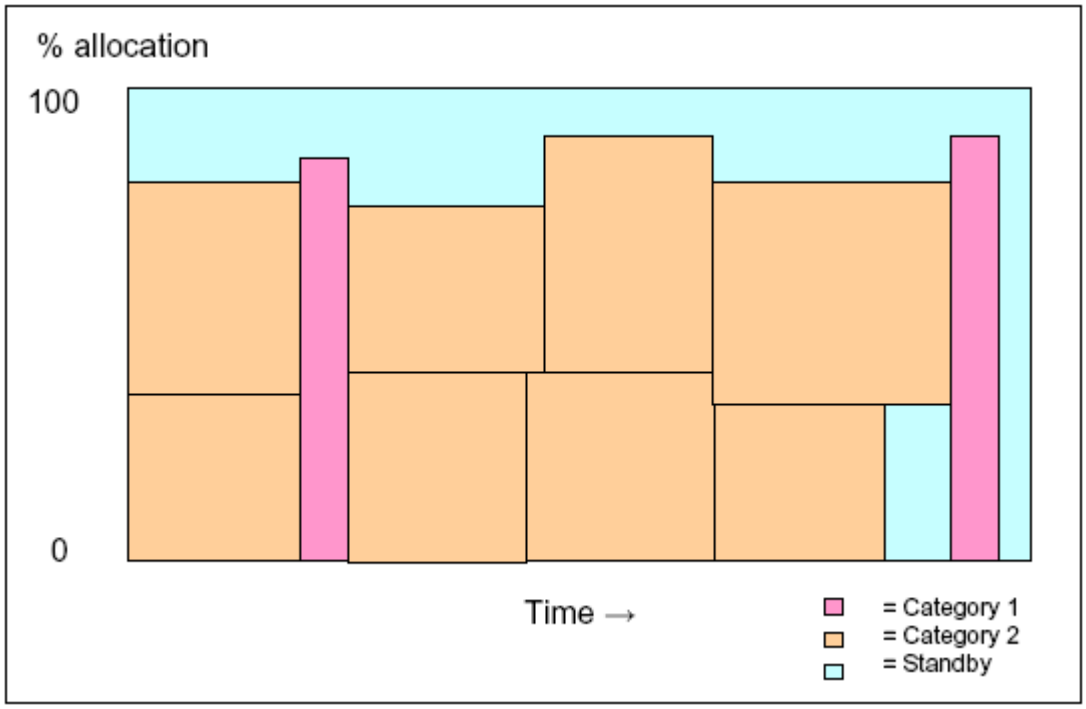


Figure 1: An illustration of the usage of an ASC Capability System versus time

REPORTING AND ACCOUNTABILITY

An annual ASC Capability Systems Tri-lab Review will be held to review the results of this governance model, providing an opportunity to adjust the process if necessary.
Host labs will prepare CCC utilization reports on a semi-annual basis, to coordinate with the semi-annual CPAC and CEC proposal review schedule. A single tri-lab report should be generated and distributed to both the CPAC and CEC and should include utilization broken down by CCC and by C1 through C3 categories for each capability class machine in the complex. It may be desirable to include additional information on job size or runtime breakdown.
A common reporting mechanism is needed for validating the CCC requests against their historical usage. This will improve accuracy in the resource requests and will ensure accountability in the use of the resources. It is recommended that this effort be coordinated with the Workload Characterization Project, with the aim that a tool be available for tri-lab common reporting.

UCRL-WEB-222594