OSCAR Cluster User's Guide
Software Version 1.2-v4.1
Documentation Version 1.2-v4.1

http://oscar.sourceforge.net/

oscar-users@lists.sourceforge.net

The Open Cluster Group
http://www.openclustergroup.org/

December 1, 2005

Contents

1 Introduction

OSCAR version 1.2-v4.1 is a snapshot of the best known methods for building, programming, and using clusters. It consists of a fully integrated and easy to install software bundle designed for high performance cluster computing. Everything needed to install, build, maintain, and use a modest sized Linux cluster is included in the suite, making it unnecessary to download or even install any individual software packages on your cluster.

OSCAR is the first project by the Open Cluster Group. For more information on the group and its projects, visit its website http://www.OpenClusterGroup.org/.

More information about OSCAR can be found at the OSCAR home page:

http://oscar.sourceforge.net/

Section 2 contains the user installation for OSCAR clusters. Cluster users will find general instructions on how to use the software installed on the OSCAR cluster. Additionally, the licenses and copyrights of each of the individual software packages contained in OSCAR are included in Section 3.

2 Individual Packages

The following sections provide general user-level information on the packages installed on OSCAR clusters.

2.1 Bamboo

Note, this material will appear in Section **??** *of the OSCAR User Guide, if it is directly included with the release (part of main tarball download).*

Bamboo is an implementation of the Queue/Job Manager component of the Scalable Systems Software suite. It provides command line tools, qsub, qstat, qsig, qdel, to allow users to submit, query, and signal jobs. It also provides the main Queue Manager server component which tracks, stores, and manages job execution.

The package typically includes:

- Bamboo-server - Main server component
- Bamboo-clients - Command line clients
- Bamboo-node-clients - Includes support for interactive jobs
- Bamboo-lib - Library used by the other components

Additional documentation including man pages is included with the components and on the Scalable Systems Software web site. http://www.scidac.org/ScalableSystems/

2.2 Berkeley Linux Checkpoint/Restart (BLCR) User's Guide

2.2.1 About Berkeley Linux Checkpoint/Restart

Checkpoint/Restart allows you to save a process to a file and later restart the process from that file. There are three main uses for this:

1. **Scheduling**: Checkpointing a program allows a program to be safely stopped at any point in its execution, so that some other program can run in its place. The original program can then be run again later.
2. **Process Migration**: if a compute node appears to be likely to crash, or there is some other reason for shutting it down (routine maintenance, hardware upgrade, etc.), checkpoint/restart allows any processes running on it to be moved to a different node (or saved until the original node is available again).
3. **Failure recovery**: a long-running program can be checkpointed intermittently, so that if it crashes due to hardware, system software, or some other non-deterministic cause, it can be restarted from a point midway in its execution, rather than run again from the beginning.

Berkeley Linux Checkpoint/Restart (BLCR) provides checkpoint/restart on Linux systems. BLCR can be used either with a single process on a single computer, or on parallel jobs (such as MPI applications) which may be running across multiple machines on a cluster of Linux nodes.

> *Note: checkpointing parallel jobs requires a library which has integrated BLCR support. At present, the only MPI implementation which supports checkpoint/restart with BLCR is the LAM/MPI library.*

2.2.2 Checkpoint/restarting within a BLCR-aware batch control system

One way to use BLCR is with a batch scheduler system (a.k.a. "job controller", "queue manager", etc.) that knows how to use the BLCR tools to checkpoint and restart the jobs under its control. You can simply tell such a system to "suspend" or "checkpoint" a job, and then later to "resume" or "restart" it.

Unfortunately BLCR has not yet been integrated with many batch systems. Currently the only system that supports BLCR with MPI jobs is the SciDAC Scalable Systems Software (SSS) Suite. If you are running on a system that uses the SSS Suite (this is the case with some versions of the OSCAR clustering toolkit), then refer to these instructions for using checkpoint/restart.

Support for serial jobs is available through SGE. See this report for more information.

The rest of this document assumes that your batch scheduler does **not** have built-in support for BLCR. In this case you will manually run the BLCR commands needed to checkpoint/restart your jobs.

Note: this does not mean that you cannot checkpoint/restart your applications if you use a batch system without built-in support for BLCR. It simply means that you have to do your checkpoints/restarts manually. To the batch system, a job that is checkpointed and terminated manually simply looks like a job that has "completed". A restart of an application looks like a "new" job.

2.2.3 Checkpointing Jobs with the BLCR command-line tools

Make sure BLCR is installed and loaded This guide assumes that BLCR has already been successfully built, installed, and configured on your system (presumably by you or your system administrator). One easy way to test this is to use the '`lsmod`' command to see if the BLCR kernel module is loaded on the node(s) that your program will run on:

```
% /sbin/lsmod
Module                  Size  Used by    Not tainted
blcr                   46936   0
vmadump_blcr           16544   0 [blcr]
iptable_filter          2412   0 (autoclean) (unused)
ip_tables              15864   1 [iptable_filter]
```

If you don't see '`blcr`' and '`vmadump_blcr`' in the output of '`lsmod`', than BLCR is **not** yet available on your system. Consult the BLCR Administrators Guide for instructions on building and installing BLCR.

Make sure your environment is set up correctly You must ensure that the BLCR commands, libraries and manual pages can be found in your shell.

Try running

```
% cr_checkpoint --help
```

If '`cr_checkpoint`' cannot be found, you need to modify your '`PATH`' to include the directory where '`cr_checkpoint`' lives. You will probably also want to modify your '`LD_LIBRARY_PATH`' variable to contain the directory where '`libcr.so`' lives, and add the BLCR man directory to your'`MANPATH`'.

Setting up your environment with 'modules': If your system uses the Environment Modules system to manage software packages, you may be able to get all of your needed environment settings simply by entering something like

```
% module add blcr
```

However, there is no requirement that 'blcr' is the name of the module you'll need–your administrator may have given it a different name ('checkpoint', etc.). Or s/he may have neglected to add BLCR to the set of packages managed by modules, in which case you'll need to use the 'manual' technique below.

Manually setting up your environment: To manually set up your environment for BLCR, the first thing you need to know is where it has been installed. By default, BLCR installs into the '`/usr/local`' directory tree, but your system administrator may have put it elsewhere by passing '`--prefix=`*PREFIX*' when BLCR was built (where *PREFIX* can be any arbitrary directory). See your system documents, or try commands such as '`locate cr_checkpoint`' or '`find`'.

Once you have determined where BLCR is installed, enter the following commands (depending on which type of shell you are using), replacing `PREFIX` with the value specified for the `--prefix` option used when configuring BLCR.

To configure a bourne-type shell (such as '`bash`' or '`ksh`'):

```
$ PATH=$PATH: PREFIX/bin
$ MANPATH=$MANPATH: PREFIX/man
$ LD_LIBRARY_PATH=$LD_LIBRARY_PATH: PREFIX/lib
$ export PATH MANPATH LD_LIBRARY_PATH
```

To configure a csh-type shell (such as '`csh`' or '`tcsh`'):

```
% setenv PATH ${PATH}: PREFIX/bin
% setenv MANPATH ${MANPATH}: PREFIX/man
% setenv LD_LIBRARY_PATH ${LD_LIBRARY_PATH}: PREFIX/lib
```

The above examples to set the PATH, MANPATH and LD_LIBRARY_PATH variables in your current session or window only. It is strongly recommended that you make these settings permanent, to make these settings affect future sessions or windows. To do this, you must add the example commands to your shell's start up files. For a single-user of BLCR, you should add the appropriate set of commands to the shell startup files in your home directory (`.bashrc` for bash, `.profile` for other bourne-type shells, or `.cshrc` for csh-type shells). For a system-wide installation, add the bourne shell commands to `/etc/bashrc` and `/etc/profile` and the csh commands to `/etc/cshrc`.

2.2.4 Checkpointing/restarting a single process application

Types of applications supported Checkpoint/restart supports:

- Single threaded applications
- Multithreaded applications using Linuxthreads or NPTL (i.e. we support both the old and new Linux pthreads implementations)
- Applications using signals

However, certain applications are not supported:

- Applications which have TCP/UDP sockets open at checkpoint time. You may register a checkpoint handler to shut down your sockets at checkpoint time, then re-open them at restart time (this is how MPI libraries typically work with BLCR). But the core BLCR checkpointer does not support saving/restoring TCP/UPD sockets.

Making an application checkpointable To be checkpointed successfully with BLCR, an application must contain some library code that BLCR provides. There are several ways of ensuring this:

1. Start your executable via the with the '`cr_run`' command:

   `% cr_run` *your_executable [arguments]*

   '`cr_run`' loads the BLCR library into your application at startup time. You do not need to modify an application to have it work with '`cr_run`'.

2. Link your application with BLCR's '`libcr`'. For instance, you could make a simple 'hello world' C program checkpointable via

   `% gcc -o hello hello.c -L` *PREFIX*`/lib -lcr`

   where *PREFIX* is the root of your BLCR install. Your application will now look for the BLCR library whenever it starts up, but note that this does not mean it will automatically be found: you will need to set your '`LD_LIBRARY_PATH`' environment variable to '*PREFIX*`/lib`' if libcr is not installed into a standard system library directory.

3. Link your application with a library which uses BLCR. For instance, if your MPI library has been made BLCR-aware, it will cause libcr to be loaded, and so simply linking with the MPI library is enough to make your application checkpointable.

4. Force the '`libcr.so`' dynamic library to do loaded at startup by adding it's full pathname to the `LD_PRELOAD` environment variable. In most cases, the pthread library will also be required. We do **not** recommend setting this in your environment in general–certain programs may interact poorly with the BLCR library logic. Instead, use a command like

   `% env LD_PRELOAD=` *PREFIX*`/lib/libcr.so.0:libpthread.so.0` *your_executable [arguments]*

   This is essentially how '`cr_run`' works.

If you do not start your program with '`cr_run`', it will simply die with an error if you try to checkpoint it. More specifically, it will receive a real-time signal (the exact one depends on your kernel and C library versions), which will cause your program to die by default, unless you handle the signal explicitly.

Checkpointing the process To checkpoint a process, simply run

`% cr_checkpoint` *PID*

where *PID* is the application's process ID.

By default, '`cr_checkpoint`' saves a checkpoint, and then lets your application continue running. This is useful for backing up a process in case it fails later, for instance.

If you wish to stop the process after it has been checkpointed, pass the '`--term`' flag:

```
% cr_checkpoint --term PID
```

This causes a `SIGTERM` signal to be received by the process at the end of the checkpoint. If you have a reason to send a different signal to your process at the end of the checkpoint, you can pass any arbitrary signal number instead via the '`--signal`' flag.

Files that contain checkpoints are called *context files*. By default, they are named '`context.`*PID*', where *PID* is the process ID that was checkpointed, and are stored in the current working directory that '`cr_checkpoint`' was run in. You may specify the name and location of the context file via the '`-f`' option.

There are a number of other options that '`cr_checkpoint`' provides. See the man page (or '`cr_checkpoint --help`') for details.

Restarting the process To restart from a context file, certain conditions must be met:

- The PID of the process in the context file must NOT be in use.
- The original executable must still exist, and its contents must be unchanged.
- All shared libraries used by the executable must exist, and their contents must be unchanged.

You may restart a program on a different machine than the one it was checkpointed on if all of these conditions are met (they often are on cluster systems, especially if you are using a shared network filesystem).

You can restart a process by using '`cr_restart`' on its context file:

```
% cr_restart context.15005
```

The original process will be restored, and resume running in the exact state it was in at checkpoint time. Note that this includes restoring its process ID, so you cannot restart a program unless the original copy of it has exited (otherwise '`cr_restart`' will fail with a message that the PID is already in use).

You may restart a process from a particular context file as many times as you wish. The context file is not automatically removed at any point–delete it if/when it is no longer useful to you.

2.2.5 Checkpointing/restarting an MPI application

Currently there is only one MPI library that has been modified to work with BLCR: the LAM/MPI library. This means that if you wish to checkpoint/restart programs on a BLCR-enabled system, you **must** use LAM/MPI. Also, you must have configured LAM correctly to use BLCR with it (i.e. use the `crtcp` or `gm` RPI). You should also *NOT* configure LAM to debug mode, i.e. do not pass `--with-debug` to LAM's configure script. See the the LAM/MPI documentation for details.

To start a checkpointable LAM/MPI application, simply run it with the regular LAM '`mpirun`' launcher:

```
% mpirun C hello_mpi
```

Note: you may need to start up the LAM environment first by running '`lamboot`' before starting your application.

To checkpoint the entire MPI application (across all nodes and processes), simply run

```
% cr_checkpoint 12305
```

Where '12305' is the process ID of the '`mpirun`' command. Do **not** pass the pid of your MPI executable: when '`mpirun`' is checkpointed, it automatically takes care of transitively checkpointing all of the processes involved in the MPI job.

To restart your MPI job, simply run '`cr_restart`' on the '`mpirun`' process's context file:

```
% cr_restart context.12305
```

All processes in the MPI job will be restarted as they were at checkpoint time.

2.2.6 Troubleshooting FAQ

My application dies with "Real-time signal 31" (or 32, etc.) when I try to checkpoint it Your application has not loaded the required BLCR library it needs to be checkpointable, and so it dies when a checkpoint signal arrives (BLCR may use a different real-time signal than 31, depending on your kernel and/or C library).

See the section on Making an application checkpointable for the various ways to fix this.

I get the error: ioctl(/proc/checkpoint/ctrl, CR_OP_RSTRT_REQ): Device or resource busy This is because a resource needed into order to restart the process is already in use. The most common problem is that another process already exists with the same pid (process ID)–the operating system will not allow you to create two programs with the same pid. Very frequently this is because a user is trying to 'restart' a process from a checkpoint, when the original process they took the checkpoint of is still running!

If you are unlucky enough that some other, unrelated process has grabbed the PID of your application, you must figure out some way to get rid of that process. If you own the process, you can of course simply kill it (or checkpoint it!). Otherwise, consider becoming root, or consulting your system administrator. BLCR will not kill another process for you (this 'feature' would raise certain security issues).

2.2.7 For more information

For more information on Checkpoint/Restart for Linux, visit the project home page: http://ftg.lbl.gov/checkpoint

For more information on LAM/MPI, see the LAM/MPI Documentation.

2.3 Using C3

2.3.1 Overview

The Cluster Command Control (C3) tools are a suite of cluster tools developed at Oak Ridge National Laboratory that are useful for both administration and application support. The suite includes tools for cluster-wide command execution, file distribution and gathering, process termination, remote shutdown and restart, and system image updates.

A short description of each tool follows:

- `cexec`: general utility that enables the execution of any standard command on all cluster nodes
- `cget`: retrieves files or directories from all cluster nodes
- `ckill`: terminates a user specified process on all cluster nodes
- `cpush`: distribute files or directories to all cluster nodes
- `cpushimage`: update the system image on all cluster nodes using an image captured by the SystemImager tool
- `crm`: remove files or directories from all cluster nodes
- `cshutdown`: shutdown or restart all cluster nodes

- `cnum`: returns a node range number based on node name
- `cname`: returns node names based on node ranges
- `clist`: returns all clusters and their type in a configuration file

The default method of execution for the tools is to run the command on all cluster nodes concurrently. However, a serial version of `cexec` is also provided that may be useful for deterministic execution and debugging. To invoke the serial version of `cexec`, type `cexecs` instead of `cexec`. For more information on how to use each tool, see the man page for the specific tool.

2.3.2 Basic File Format

Specific instances of C3 commands identify their compute nodes with the help of *cluster configuration files*: files that name a set of accessible clusters and describe the set of machines in each accessible cluster. `/etc/c3.conf`, the default cluster configuration file, should consist of a list of *cluster descriptor blocks*: syntactic objects that name and describe a single cluster that is accessible to that system's users. The following is an example of a default configuration file that contains exactly one cluster descriptor block: a block that describes a cluster of 64 nodes:

```
cluster cartman {
    cartman-head:node0  #head node
    node[1-64]          #compute nodes
}
```

The cluster tag denotes a new cluster descriptor block. The next word is the name of the cluster (in this example, "`cartman`"). The first line in the configuration is the head node. The first name is the external interface followed by a colon and then the internal interface (for example, an outside user can login to the cluster by `ssh`'ing to "`cartman-head.mydomain.com`"). If only one name is specified, then it is assumed to be both external and internal. Starting on the next line is the node definitions. Nodes can be either ranges or single machines. The above example uses ranges – `node1` through `node64`.

In the case of a node being offline, two tags are used: `exclude` and `dead`. `exclude` sets nodes offline that are declared in a range and `dead` indicates a single node declaration is dead. The below example declares 32 nodes in a range with several offline and then 4 more nodes listed singularly with 2 offline.

```
cluster kenny {
    node0            #head node
    dead placeholder #change command line to 1 indexing
    node[1-32]       #first set of nodes
    exclude 30       #offline nodes in the range
```

```
    exclude [5-10]
    node100          #single node definition
    dead node101     #offline node
    dead node102
    node103
}
```

One other thing to note is the use of a place holder node. When specifying ranges on the command line a nodes position in the configuration file is relevant. Ranges on the command line are 0 indexed.

For example, in the `cartman` cluster example (first example), `node1` occupies position 0 which may not be very intuitive to a user. Putting a node offline in front of the real compute nodes changes the indexing of the C3 command line ranges. In the `kenny` cluster example (second example) node1 occupies position 1.

For a more detailed example, see the `c3.conf` man page.

2.3.3 Specifying ranges

Ranges can be specified in two ways, one as a range, and the other as a single node. Ranges are specified by the following format: `m-n`, where m is a positive integer (including zero) and n is a number larger than m. Single positions are just the position number.

If discontinuous ranges are needed, each range must be separated by a "`,`". The range "0-5, 9, 11" would execute on positions 0, 1, 2, 3, 4, 5, 9, and 11 (nodes marked as `offline` will not participate in execution).

There are two tools used to help manage keeping track of which nodes are at which position: `cname(1)` and `cnum(1)`. `cnum` assumes that you know node names and want to know their position. `cname` takes a range argument and returns the node names at those positions (if no range is specified it assumes that you want all the nodes in the cluster). See their man pages for details of use.

NOTE: ranges begin at zero!

2.3.4 Machine Definitions

Machine definitions are what C3 uses to specify clusters and ranges on the command line. There are four cases a machine definition can take. First is that one is not specified at all. C3 will execute on all the nodes on the *default cluster* in this case (the `default cluster` is the first cluster in the configuration file). An example would be as follows:

```
$ cexec ls -l
```

the second case is a range on the default cluster. This is in a form of <:range>. An example cexec would be as follows:

```
$ cexec :1-4,6 ls -l
```

This would execute ls on nodes 1, 2, 3, 4, and 6 of the default cluster. The third method is specifying a specific cluster. This takes the form of <cluster_name:>. An example cexec would be as follows:

```
$ cexec cartman: ls -l
```

This would execute ls on every node in cluster cartman. The fourth (and final) way of specifying a machine definition would be a range on a specific cluster. This takes the form of <cluster_name:range>. An example cexec would be as follows:

```
$ cexec cartman:2-4,10 ls -l
```

This would execute ls on nodes 2, 3, 4, and 10 on cluster cartman. These four methods can be mixed on a single command line. for example

```
$ cexec :0-4 stan: kyle:1-5 ls -l
```

is a valid command. it would execute ls on nodes 0, 1, 2, 3, and 4 of the default cluster, all of stan and nodes 1, 2, 3, 4, and 5 of kyle (the stan and kyle cluster configuration blocks are not shown here). In this way one could easily do things such as add a user to several clusters or read /var/log/messages for an event across many clusters. See the c3-range man page for more detail.

2.3.5 Other Considerations

In most cases, C3 has tried to mimic the standard Linux command it is based on. This is to make using the cluster as transparent as possible. One of the large differences is such as using the interactive options. Because one would not want to be asked yes or no multiple times for each node, C3 will only ask *once* if the interactive option is specified. This is very important to remember if running commands such as "crm --all -R /tmp/foo" (recursively delete /tmp/foo on every node in every cluster you have access too).

Multiple cluster uses do not necessarily need to be restricted by listing specific nodes; nodes can also be grouped based on role, essentially forming a meta-cluster. For example, if one wishes to separate out PBS server nodes for specific tasks, it is possible to create a cluster called pbs-servers and only execute a given command on that cluster. It is also useful to separate nodes out based on things such as hardware (e.g., fast-ether/gig-ether), software (e.g., some nodes may have a specific compiler), or role (e.g., pbs-servers).

2.4 Using LAM/MPI

LAM (Local Area Multicomputer) is an MPI programming environment and development system for heterogeneous computers on a network. With LAM/MPI, a dedicated cluster or an existing network computing infrastructure can act as a single parallel computer. LAM/MPI is considered to be "cluster friendly," in that it offers daemon-based process startup/control as well as fast client-to-client message passing protocols. LAM/MPI can use TCP/IP and/or shared memory for message passing.

LAM features a full implementation of MPI-1, and much of MPI-2. Compliant applications are source code portable between LAM/MPI and any other implementation of MPI. In addition to providing a high-quality implementation of the MPI standard, LAM/MPI offers extensive monitoring capabilities to support debugging. Monitoring happens on two levels. First, LAM/MPI has the hooks to allow a snapshot of process and message status to be taken at any time during an application run. This snapshot includes all aspects of synchronization plus datatype maps/signatures, communicator group membership, and message contents (see the XMPI application on the main LAM web site). On the second level, the MPI library is instrumented to produce a cummulative record of communication, which can be visualized either at runtime or post-mortem.

2.4.1 Notes about OSCAR's LAM/MPI Setup

The OSCAR environment is able to have multiple MPI implementations installed simultaneously – see Section 2.11 (page 23) for a description of the `switcher` program.

LAM/MPI is configured on OSCAR to use the Secure Shell (`ssh`) to initially start processes on remote nodes. Normally, using `ssh` requires each user to set up cryptographic keys before being able to execute commands on remote nodes with no password. The OSCAR setup process has already taken care of this step for you. Hence, the LAM command `lamboot` should work with no additional setup from the user.

2.4.2 Setting up `switcher` to use LAM/MPI

In order to use LAM/MPI successfully, you must first ensure that `switcher` is set to use LAM/MPI. First, execute the following command:

```
$ switcher mpi --show
```

If the result contains a line beginning with "`default`" followed by a string containing "`lam`" (e.g., "`lam-6.5.9`"), then you can skip the rest of this section. Otherwise, execute the following command:

```
$ switcher mpi --list
```

This shows all the MPI implementations that are available. Choose one that contains the name "`lam`" (e.g., "`lam-6.5.9`") and run the following command:

```
$ switcher mpi = lam-6.5.9
```

This will set all *future* shells to use LAM/MPI. In order to guarantee that all of your login environments contain the proper setup to use LAM/MPI, it is probably safest to logout and log back in again. Doing so will guarantee that all of your interactive shells will have the LAM commands and man pages will be found (i.e., your `$PATH` and `$MANPATH` environment variables set properly for LAM/MPI).

Hence, you will be able to execute commands such as "`mpirun`" and "`man lamboot`" without any additional setup.

2.4.3 General Usage

The general scheme of using LAM/MPI is as follows:

1. Use the `lamboot` command to start up the LAM run-time environment (RTE) across a specified set of nodes. The `lamboot` command takes a single argument: the filename of a hostfile containing a list of nodes to run on. For example:

   ```
   $ lamboot my_hostfile
   ```

2. Repeat the following steps as many times as necessary:

   (a) Use the MPI "wrapper" compilers (`mpicc` for C programs, `mpiCC` for C++ programs, and `mpif77` for fortran programs) to compile your application. These wrapper compilers add in all the necessary compiler flags and then invoke the underlying "real" compiler. For example:

   ```
   $ mpicc myprogram.c -o myprogram
   ```

   (b) Use the `mpirun` command to launch your parallel application in the LAM RTE. For example:

   ```
   $ mpirun C myprogram
   ```

   The `mpirun` command has many options and arguments – see the man page and/or "`mpirun -h`" for more information.

   (c) If your parallel program fails ungracefully, use the `lamclean` command to "clean" the LAM RTE and guarantee to remove all instances of the running program.

3. Use the `lamhalt` command to shut down the LAM RTE. The `lamhalt` command takes no arguments.

Note that the wrapper compilers are all set to use the corresponding GNU compilers (`gcc`, `g++`, and `gf77`, respectively). Attempting to use other compilers may run into difficulties because their linking styles may be different than what the LAM libraries expect (particularly for C++ and Fortran compilers).

2.4.4 More Information

The LAM/MPI web site (`http://www.lam-mpi.org/`) contains much, much more information about LAM/MPI, including:

- A large Frequently Asked Questions (FAQ) list
- Usage tutorials and examples
- Access to the LAM user's mailing list, including subscription instructions and web archives of the list

Make today a LAM/MPI day!

2.5 maui

Note, this material will appear in Section **??** *of the OSCAR User Guide, if it is directly included with the release (part of main tarball download).*

Maui is an implentation of the scheduling component of the Scalable Systems Software suite. It provides command line tools, showq, showbf, showres, etc, to allow users to monitor the status of the system and their jobs.

The package typically includes:

- Maui-oscar - Scheduler and command line clients

Additional documentation including man pages is included with the components and on the Scalable Systems Software web site. http://www.scidac.org/ScalableSystems/

2.6 The OSCAR Password Installer and User Management (OPIUM)

The OPIUM package includes facilities which synchronize the cluster's accounts and configures `ssh` for users. The user account synchronization may only be run by root, and is automatically triggered at regular intervals. OPIUM configures `ssh` such that every user can traverse the cluster securely without entering a password, once logged on to the head node. This is done using `ssh` user keys, in the `.ssh` folder in your home directory. It is not recommended that you make changes here unless you know what you are doing. If you change your password, make sure to change it on the OSCAR head node, because changes propagate to the cluster nodes from there.

2.7 Packet Filtering with pfilter

When the **pfilter** packet filtering system is turned on, the default OSCAR settings allow any network communication between the machines in the cluster, and allow ssh and http access to the cluster main machine from the outside.

Communication between cluster machines and the outside network are limited to outgoing connections only. Incoming network connections to cluster machines are blocked.

To allow outside network connections to ports on the cluster machines, special rules will have to be added to the **pfilter** configuration. See your cluster administrator for help on this.

2.8 Parallel Virtual Machine (PVM)

PVM (Parallel Virtual Machine) is a software package that permits a heterogeneous collection of Unix and/or Windows computers hooked together by a network to be used as a single large parallel computer. Thus large computational problems can be solved more cost effectively by using the aggregate power and memory of many computers. The software is very portable. The source, which is available free thru netlib, has been compiled on everything from laptops to CRAYs.

PVM enables users to exploit their existing computer hardware to solve much larger problems at minimal additional cost. Hundreds of sites around the world are using PVM to solve important scientific, industrial, and medical problems in addition to PVM's use as an educational tool to teach parallel programming. With tens of thousands of users, PVM has become the de facto standard for distributed computing world-wide.

2.8.1 Using PVM

The default OSCAR installation tests PVM via a Torque/PBS job (see also: Section **??** on page **??**). However, some users may choose to use PVM outside of this context so a few words on usage may be helpful [1].

The default location for user executables is `$HOME/pvm3/bin/$PVM_ARCH`. On an IA-32 Linux machine, this is typically of the form: `/home/sgrundy/pvm3/bin/LINUX` (replace "`LINUX`" with "`LINUX64`" on IA-64). This is where binaries should be placed so that PVM can locate them when attempting to spawn tasks. This is detailed in the `pvm_intro(1PVM)` manual page when discussing the environment variables `PVM_PATH` and `PVM_WD`.

The "hello world" example shipped with PVM demonstrates how one can compile and run a simple application outside of Torque/PBS. The following screen log highlights this for a standard user *sgrundy* (Solomon Grundy).

[1] Note, the examples in this section assume a shared filesystem, as is used with OSCAR.

```
  # Crate default directory for PVM binaries (one time operation)
sgrundy: $ mkdir -p $HOME/pvm3/bin/$PVM_ARCH

  # Copy examples to local 'hello' directory
sgrundy: $ cp $PVM_ROOT/examples/hello*  $HOME/hello-example
sgrundy: $ cd $HOME/hello-example

  # Compile a hello world, using necessary include (-I) and library
  # (-L) search path info as well as the PVM3 lib.
sgrundy: $ gcc -I$PVM_ROOT/include  hello.c -L$PVM_ROOT/lib/$PVM_ARCH  \
> -lpvm3  -o hello
sgrundy: $ gcc -I$PVM_ROOT/include  hello_other.c -L$PVM_ROOT/lib/$PVM_ARCH \
> -lpvm3  -o hello_other

  # Move the companion that will be spawned to the default
  # PVM searchable directory
sgrundy: mv hello_other $HOME/pvm3/bin/$PVM_ARCH
```

At this point you can start PVM, add hosts to the virtual machine and run the application:

```
  # Start PVM and add one host "oscarnode1".
sgrundy: $ pvm
pvm> add oscarnode1
add oscarnode1
1 successful
                  HOST    DTID
            oscarnode1   80000
pvm> quit
quit

Console: exit handler called
pvmd still running.
sgrundy: $

  # Run master portion of hello world which contacts the companion.
sgrundy: $ ./hello
i'm t40005
from t80002: hello, world from oscarnode1.localdomain

  # Return to the PVM console and terminate ('halt') the virtual machine.
sgrundy: $
sgrundy: $ pvm
pvmd already running
pvm> halt
halt
Terminated
sgrundy: $
```

An alternate method is to use options in the hostfile supplied to `pvm` when started from the command-line. The advantage to the hostfile options is that you don't have to place your binaries in the default location or

edit any ".dot" files. You can compile and run the "hello world" example in this fashion by using a simple hostfile as shown here.

The example below uses the same "hello world" program that was previously compiled, but using a hostfile with the appropriate options to override the default execution and working directory. Remember that the "`hello`" program exists in the `/home/sgrundy/hello-example` directory:

```
sgrumpy: $ cat myhostfile
*   ep=/home/sgrundy/hello-example  wd=/home/sgrundy/hello-example
oscarnode1
```

The options used here are:

- `*` – any node can connect
- `ep` – execution path, here set to local directory
- `wd` – working directory, here set to local directory
- ***nodes*** – a list of nodes, one per line

Now, we can startup `pvm` using this `myhostfile` and run the `hello` application once again.

```
 # Now, we just pass this as an argument to PVM upon startup.
sgrundy: $ pvm myhostfile
pvm> quit
quit

Console: exit handler called
pvmd still running.

 # The rest is the same as the previous example
sgrundy: $ ./hello
i'm t40005
from t80002: hello, world from oscarnode1.localdomain

sgrundy: $ pvm
pvmd already running
pvm> halt
halt
Terminated
sgrundy: $
```

2.8.2 Other details

The OSCAR installation of PVM makes use of the `env-switcher` package (also see Section 2.11, page 23). This is where the system-wide `$PVM_ROOT, $PVM_ARCH` and `$PVM_RSH` environment variable defaults are set. Traditionally, this material was included in each user's ".dot" files to ensure availability with non-interactive shells (e.g. `rsh`/`ssh`). Through the `env-ewitcher` package, a user can avoid any ".dot" file adjustments by using the hostfile directive or default paths for binaries as outlined in the Usage Section 2.8.1.

For additional information see also:

- PVM web site: `http://www.csm.ornl.gov/pvm/`
- Manual Pages: pvm(1), pvm_intro(1), pvmd3(1)
- Release docs: `$PVM_ROOT/doc/*`

2.9 An overview of SIS

The System Installation Suite, or SIS, is a tool for installing Linux systems over a network. It is used in OSCAR to install the client nodes. SIS also provides the database from which OSCAR obtains its cluster configuration information.

The main concept to understand about SIS is that it is an *image based* install tool. An image is basically a copy of all the files that get installed on a client. This image is stored on the server and can be accessed for customizations or updates. You can even `chroot` into the image and perform builds.

Once this image is built, clients are defined and associated with the image. When one of these clients boots using a SIS auto-install environment, either on floppy, CD, or through a network boot, the corresponding image is pulled over the network using `rsync`. Once the image is installed, it is customized with the hardware and networking information for that specific client and it is then rebooted. When booting the client will come up off the local disk and be ready to join the OSCAR cluster.

2.9.1 Building a SIS image

Normally, an OSCAR image is built using the **`<Build OSCAR Client Image>`** button on the OSCAR wizard. This button brings up a panel that is actually directly from the SIS GUI `tksis`. Once the information is filled in, the SIS command `mksiimage` is invoked to actually build the image.

In addition to building an image, you can use `tksis` or `mksiimage` to delete images as well. Images can take a fair amount of disk space, so if you end up with images that you aren't using, you can delete them to recover some space.

2.9.2 Managing SIS clients

Much like the OSCAR image creation, the **<Define OSCAR Clients>** button actually invokes a `tksis` panel. There are a couple of SIS commands that are used to manage the client definitions. `mksirange` is used to define a group of clients. More importantly, `mksimachine` can be used to update client definitions. If, for example, you needed to change the MAC address after replacing one of your clients, you could use `mksimachine`.

2.9.3 Maintaining your client software

There are many different ways to maintain the software installed on the client nodes. Since SIS is image based, it allows you to also use an image based maintenance scheme. Basically, you apply updates and patches to your images and then resync the clients to their respective images. Since `rsync` is used, only the actual data that has changed will be sent over the network to the client. The SIS command `updateclient` can be run on any client to initiate this update[2].

2.9.4 Additional information

To obtain more detailed information about SIS, please refer to the many man pages that are shipped with SIS. Some of the more popular pages are:

- `tksis`
- `mksiimage`
- `mksidisk`
- `mksirange`
- `mksimachine`
- `systemconfigurator`
- `updateclient`

You can also access the mailing lists and other docs through the sisuite home page, http://sisuite.org/.

[2] In OSCAR, when using `updateclient` to maintain a cluster, the last step of the wizard must be re-run to keep the system configurations in sync, i.e., re-run "Complete Cluster Setup".

2.10 SSSLib Overview

These are the build and communication components from ANL for the Scalable System Software (SSS) project. For more details see the internal docs or the electronic notebooks at http://www.scidac.org/ScalableSystems.

2.11 An overview of `switcher`

Experience has shown that requiring untrained users to manually edit their "dot" files (e.g., `$HOME/.bashrc`, `$HOME/.login`, `$HOME/.logout`, etc.) can result in damaged user environments. Side effects of damaged user environments include:

- Lost and/or corrupted work
- Severe frustration / dented furniture
- Spending large amounts of time debugging "dot" files, both by the user and the system administrator

The OSCAR `switcher` package is an attempt to provide a simple mechanism to allow users to manipulate their environment. The `switcher` package provides a convenient command-line interface to manipulate the inclusion of packages in a user's environment. Users are not required to manually edit their "dot" files, nor are they required to know what the inclusion of a given package in the environment entails.[3] For example, if a user specifies that they want LAM/MPI in their environment, `switcher` will automatically add the appropriate entries to the `$PATH` and `$MANPATH` environment variables.

Finally, the OSCAR `switcher` package provides a two-level set of defaults: a system-level default and a user-level default. User-level defaults (if provided) override corresponding system-level defaults. This allows a system administrator to (for example) specify which MPI implementation users should have in their environment by setting the system-level default. Specific users, however, may decide that they want a different implementation in their environment and set their personal user-level default.

Note, however, that *`switcher` does not change the environment of the shell from which it was invoked.* This is a critical fact to remember when administrating your personal environment or the cluster. While this may seem inconvenient at first, `switcher` was specifically designed this way for two reasons:

1. If a user inadvertantly damages their environment using `switcher`, there is still [potentially] a shell with an undamaged environment (i.e., the one that invoked `switcher`) that can be used to fix the problem.

[3] Note, however, that it was a requirement for the OSCAR `switcher` package that advanced users should not be precluded – in any way – from either not using `switcher`, or otherwise satisfying their own advanced requirements without interference from `switcher`.

2. The `switcher` package uses the `modules` package for most of the actual environment manipulation (see http://modules.sourceforge.net/). The `modules` package can be used directly by users (or scripts) who wish to manipulate their current environment.

The OSCAR `switcher` package contains two sub-packages: `modules` and `env-switcher`. The `modules` package can be used by itself (usually for advanced users). The `env-switcher` package provides a persistent `modules`-based environment.

2.11.1 The `modules` package

The `modules` package (see http://modules.sourceforge.net/) provides an elegant solution for individual packages to install (and uninstall) themselves from the current environment. Each OSCAR package can provide a modulefile that will set (or unset) relevant environment variables, create (or destroy) shell aliases, etc.

An OSCAR-ized `modules` RPM is installed during the OSCAR installation process. Installation of this RPM has the following notable effects:

- Every user shell will be setup for modules – notably, the commands "`module`" and "`man module`" will work as expected.
- Guarantee the execution of all modulefiles in a specific directory for every shell invocation (including corner cases such as non-interactive remote shell invocation by `rsh`/`ssh`).

Most users will not use any `modules` commands directly – they will only use the `env-switcher` package. However, the `modules` package can be used directly by advanced users (and scripts).

2.11.2 The `env-switcher` package

The `env-switcher` package provides a persistent `modulues`-based environment. That is, `env-switcher` ensures to load a consistent set of modules for each shell invocation (including corner cases such as non-interactive remote shells via `rsh`/`ssh`). `env-switcher` is what allows users to manipulate their environment by using a simple command line interface – not by editing "dot" files.

It is important to note that *using the `switcher` command alters the environment of all* **future** *shells / user environments. `switcher` does not change the environment of the shell from which it was invoked.* This may seem seem inconvenient at first, but was done deliberately. See the rationale provided at the beginning of this section for the reasons why. If you're really sure that you know what you're doing, tou can use the "`switcher-reload`" command after changing your switcher settings via the `switcher` command. This will change your current shell/environment to reflect your most recent switcher settings.

`env-switcher` manipulates four different kinds of entities: tags, attributes, and values.

- *Tags* are used to group similar software packages. In OSCAR, for example, "mpi" is a commonly used tag.
- *Names* are strings that indicate individual software package names in a tag.
- Each tag can have zero or more *attributes*.
- An attribute, if defined, must have a single *value*. An attribute specifies something about a given tag by having an assigned value.

There are a few built-in attributes with special meanings (any other attribute will be ignored by `env-switcher`, and can therefore be used to cache arbitrary values). "default" is probably the most-commonly used attribute – its value specifies which package will be loaded (as such, its value is always a name). For example, setting the "default" attribute on the "mpi" tag to a given value will control which MPI implementation is loaded into the environment.

`env-switcher` operates at two different levels: system-level and user-level. The system-level tags, attributes, and values are stored in a central location. User-level tags, attributes, and values are stored in each user's `$HOME` directory.

When `env-switcher` looks up entity that it manipulates (for example, to determine the value of the "default" attribute on the "mpi" tag), it attempts to resolves the value in a specific sequence:

1. Look for a "default" attribute value on the "mpi" tag in the user-level defaults
2. Look for a "default" attribute value on the "global" tag in the user-level defaults
3. Look for a "default" attribute value on the "mpi" tag in the system-level defaults
4. Look for a "default" attribute value on the "global" tag in the system-level defaults

In this way, a four-tiered set of defaults can be effected: specific user-level, general user-level, specific system-level, and general system-level.

The most common `env-switcher` commands that users will invoke are:

1. `switcher --list`

   List all available tags.
2. `switcher <tag> --list`

   List all defined attributes for the tag `<tag>`.
3. `switcher <tag> = <value> [--system]`

   A shortcut nomenclature to set the "default" attribute on `<tag>` equal to the value `<value>`. If the `--system` parameter is used, the change will affect the system-level defaults; otherwise, the user's personal user-level defaults are changed.

4. `switcher <tag> --show`

   Show the all attribute / value pairs for the tag `<tag>`. The values shown will be for attributes that have a resolvable value (using the resolution sequence described above). Hence, this output may vary from user to user for a given `<tag>` depending on the values of user-level defaults.

5. `switcher <tag> --rm-attr <attr> [--system]`

   Remove the attribute `<attr>` from a given tag. If the `--system` parameter is used, the change will affect the system level defaults; otherwise, the user's personal user-level defaults are used.

Section 2.11.3 shows an example scenario using the `switcher` command detailing how to change which MPI implementation is used, both at the system-level and user-level.

See the man page for `switcher(1)` and the output of `switcher --help` for more details on the `switcher` command.

2.11.3 Which MPI do you want to use?

OSCAR has a generalized mechanism to both set a system-level default MPI implementation, and also to allow users to override the system-level default with their own choice of MPI implementation.

This allows multiple MPI implementations to be installed on an OSCAR cluster (e.g., LAM/MPI and MPICH), yet still provide unambiguous MPI implementation selection for each user such that "`mpicc foo.c -o foo`" will give deterministic results.

2.11.4 Setting the system-level default

The system-level default MPI implementation can be set in two different (yet equivalent) ways:

1. During the OSCAR installation, the GUI will prompt asking which MPI should be the system-level default. This will set the default for all users on the system who do not provide their own individual MPI settings.
2. As `root`, execute the command:

   ```
   # switcher mpi --list
   ```

   This will list all the MPI implementations available. To set the system-level default, execute the command:

   ```
   # switcher mpi = name --system
   ```

where “name” is one of the names from the output of the `--list` command.

NOTE: System-level defaults for `switcher` are currently propogated to the nodes on a periodic basis. If you set the system-level MPI default, you will either need to wait until the next automatic “push” of configuration information, or manually execute the `/opt/sync_files/bin/sync_files` command to push the changes to the compute nodes.

NOTE: Using the `switcher` command to change the default MPI implementation will modify the `PATH` and `MANPATH` for all *future* shell invocations – it does *not* change the environment of the shell in which it was invoked. For example:

```
# which mpicc
/opt/lam-1.2.3/bin/mpicc
# switcher mpi = mpich-4.5.6 --system
# which mpicc
/opt/lam-1.2.3/bin/mpicc
# bash
# which mpicc
/opt/mpich-4.5.6/bin/mpicc
```

If you wish to have your current shell reflect the status of your switcher settings, you must run the “`switcher-reload`” command. For example:

```
# which mpicc
/opt/lam-1.2.3/bin/mpicc
# switcher mpi = mpich-4.5.6 --system
# which mpicc
/opt/lam-1.2.3/bin/mpicc
# switcher-reload
# which mpicc
/opt/mpich-4.5.6/bin/mpicc
```

Note that this is *only* necessary if you want to change your current environment. All new shells (including scripts) will automatically get the new switcher settings.

2.11.5 Setting the user-level default

Setting a user-level default is essentially the same as setting the system-level default, except without the `--system` argument. This will set the user-level default instead of the system-level default:

```
$ switcher mpi = lam-1.2.3
```

Using the special name `none` will indicate that no module should be loaded for the `mpi` tag. It is most often used by users to specify that they do not want a particular software package loaded.

```
$ switcher mpi = none
```

Removing a user default (and therefore reverting to the system-level default) is done by removing the `default` attribute:

```
$ switcher mpi --show
user:default=mpich-1.2.4
system:exists=true
$ switcher mpi --rm-attr default
$ switcher mpi --show
system:default=lam-6.5.6
system:exists=true
```

2.11.6 Use `switcher` with care!

`switcher` immediately affects the environment of all future shell invocations (including the environment of scripts). To get a full list of options available, read the `switcher(1)` man page, and/or run `switcher --help`.

2.12 Warehouse

warehouse is a tool for obtaining and collecting real-time data from multiple points in a computer system. *warehouse* is specifically implemented to keep information on the state of all the resources in a computer cluster. Warehouse is unaware of the significance or the formatting of the information, so new information can be collected without modifying the warehouse infrastructure.

warehouse serves in the Scalable Systems Software distribution as the *System Monitor* component. The System Monitor keeps track of the state of all the computational resources in a cluster; the static information about each node (speed of CPU, amount of memory, ...) and also live performance infomation (CPU utilization, swap used, ...). In the current state of the SSS software distribution, this information is used by the scheduler to find out what nodes are available for jobs.

warehouse consists of two components, which are packaged in separate rpms. The daemon that runs on each compute node be monitored is `warehouse_monitor`, packaged in `warehouse_node.rpm`. It harvests

local information via the `hostname` command and the `/proc` file system, keeps that information updated, and opens a port for other warehouse processes to request that information. The performance is harvested from an external

The other component is `warehouse_System_Monitor`, packaged in `warehouse_SysMon*.rpm`. This is the actual software entity that provides the SSS System Monitor interface to the rest of the SSS software. This component contacts all of the `warehouse_monitor`s in the cluster, and requests a steady stream of updates from them. This aggregated information is then available for request by other components (currently just the scheduler).

Known Bugs and Testing This information applies to version 0.7.4 of *warehouse*, cut mid-October 2004.

The good news: all known bugs in *warehouse* resulting from connection failures have been resolved. That is, if a network connection cannot be made to a node, then that node is not monitored, but it does not obstruct other nodes from being monitored.

The bad news: connection to nodes is still done serially; that is, an attempt to connect to each node is made before any of the nodes are monitored. There is a re-vamp in the works to fix this and several other problems, but it did not make it into testing before the release. The updates will be versions 0.8 and later, and probably will be available before the end of calendar year 2004. The serial connection method means that if you have a large number of nodes configured that do not, in fact, have a warehouse node rpm running on them, then it will take a while to time out on those nodes.

That having been said, if the configuration file is correct, and the network is well behaved, then none of these things should be a problem.

3 Package Licenses and Copyrights

Since OSCAR includes many packages, the licenses and copyrights for each of them is included here for reference.

3.1 Bamboo

The Bamboo package contains the following copyrights and licenses.

```
Copyright (c) 2003, Iowa State University of Science and Technology
 All rights reserved.

This computer software was prepared by Iowa State University of
Science and Technology and B. Bode: (515) 294-9192;
```

FAX: (515) 294-4491; e-mail: brett@scl.ameslab.gov, hereinafter referred to as the Contractor, under Contract No. W-7405-ENG-82 with the Department of Energy (DOE).

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Ames Laboratory, Iowa State University of Science and Technology nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

3.2 Berkeley Lab Checkpoint/Restart (BLCR) for Linux

Berkeley Lab Checkpoint/Restart (BLCR) for Linux is Copyright (c) 2004, The Regents of the University of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from the U.S. Dept. of Energy). All rights reserved.

Portions may be copyrighted by others, as may be noted in specific copyright notices within specific files.

The library (libcr) and the associated header (*.h) files are covered by the following notice and LGPL license: Berkeley Lab Checkpoint/Restart (BLCR) for Linux is Copyright (c) 2004, The Regents of the University

of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from the U.S. Dept. of Energy). All rights reserved.

Portions may be copyrighted by others, as may be noted in specific copyright notices within specific files.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Library General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Library General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

The remaining files (including kernel modules and executables) are covered by the following notice and GPL license: Berkeley Lab Checkpoint/Restart (BLCR) for Linux is Copyright (c) 2004, The Regents of the University of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from the U.S. Dept. of Energy). All rights reserved.

Portions may be copyrighted by others, as may be noted in specific copyright notices within specific files.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA

3.3 C3

The C3 OSCAR package contains the following copyrights and licenses.

The C3 package contains:

```
      C3 version 4.0:   Cluster Command & Control Suite
       Oak Ridge National Laboratory, Oak Ridge, TN,
  Authors: M.Brim, R.Flanery, G.A.Geist, B.Luethke, S.L.Scott
```

(C) 2001 All Rights Reserved

NOTICE

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted provided that the above copyright notice appear in all copies and that both the copyright notice and this permission notice appear in supporting documentation.

Neither the Oak Ridge National Laboratory nor the Authors make any representations about the suitability of this software for any purpose. This software is provided "as is" without express or implied warranty.

The C3 tools were funded by the U.S. Department of Energy.

3.4 Disable Services

The `disable-services` package contains the following copyrights and licenses.

Indiana University has the exclusive rights to license this product under the following license.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1) All redistributions of source code must retain the above copyright notice, the list of authors in the original source code, this list of conditions and the disclaimer listed in this license;
2) All redistributions in binary form must reproduce the above copyright notice, this list of conditions and the disclaimer listed in this license in the documentation and/or other materials provided with the distribution;
3) Any documentation included with all redistributions must include the following acknowledgement:

"This product includes software developed at the Pervasive Technology Labs at Indiana University. For technical information contact Andrew Lumsdaine at the Pervasive Technology Labs at Indiana University. For administrative and license questions contact the Advanced Research and Technology Institute at 1100 Waterway Blvd. Indianapolis, Indiana 46202, phone 317-274-5905, fax 317-274-5902."

Alternatively, this acknowledgement may appear in the software itself, and wherever such third-party acknowledgments normally appear.

4) The name "disable-services" or shall not be used to endorse or promote products derived from this software without prior written permission from Indiana University. For written permission, please contact Indiana University Advanced Research & Technology Institute.
5) Products derived from this software may not be called "disable-services", nor may "disable-services" appear in their name, without prior written permission of Indiana University Advanced Research & Technology Institute.

Indiana University provides no reassurances that the source code provided does not infringe the patent or any other intellectual property rights of any other entity. Indiana University disclaims any liability to any recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise.

LICENSEE UNDERSTANDS THAT SOFTWARE IS PROVIDED "AS IS" FOR WHICH NO WARRANTIES AS TO CAPABILITIES OR ACCURACY ARE MADE. INDIANA UNIVERSITY GIVES NO WARRANTIES AND MAKES NO REPRESENTATION THAT SOFTWARE IS FREE OF INFRINGEMENT OF THIRD PARTY PATENT, COPYRIGHT, OR OTHER PROPRIETARY RIGHTS. INDIANA UNIVERSITY MAKES NO WARRANTIES THAT SOFTWARE IS FREE FROM "BUGS", "VIRUSES", "TROJAN HORSES", "TRAP DOORS", "WORMS", OR OTHER HARMFUL CODE. LICENSEE ASSUMES THE ENTIRE RISK AS TO THE PERFORMANCE OF SOFTWARE AND/OR ASSOCIATED MATERIALS, AND TO THE PERFORMANCE AND VALIDITY OF INFORMATION GENERATED USING SOFTWARE.

Indiana University has the exclusive rights to license this product under this license.

3.5 LAM/MPI

The LAM/MPI OSCAR package contains the following copyrights and licenses.

Software License for LAM/MPI

Indiana University has the exclusive rights to license this product under the following license.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1) All redistributions of source code must retain the above copyright notice, the list of authors in the original source code, this list of conditions and the disclaimer listed in this license;
2) All redistributions in binary form must reproduce the above copyright notice, this list of conditions and the disclaimer listed in this license in the documentation and/or other materials provided with the distribution;
3) Any documentation included with all redistributions must include the following acknowledgement:

"This product includes software developed at the Ohio Supercomputer Center at The Ohio State University, the University of Notre Dame and the Pervasive Technology Labs at Indiana University with original ideas contributed from Cornell University. For technical information contact Andrew Lumsdaine at the Pervasive Technology Labs at Indiana University. For administrative and license questions contact the Advanced Research and Technology Institute at 1100 Waterway Blvd. Indianapolis, Indiana 46202, phone 317-274-5905, fax 317-274-5902."

Alternatively, this acknowledgement may appear in the software itself, and wherever such third-party acknowledgments normally appear.

4) The name "LAM" or "LAM/MPI" shall not be used to endorse or promote products derived from this software without prior written permission from Indiana University. For written permission, please contact Indiana University Advanced Research & Technology Institute.
5) Products derived from this software may not be called "LAM" or "LAM/MPI", nor may "LAM" or "LAM/MPI" appear in their name, without prior written permission of Indiana University Advanced Research &

Technology Institute.

Indiana University provides no reassurances that the source code provided does not infringe the patent or any other intellectual property rights of any other entity. Indiana University disclaims any liability to any recipient for claims brought by any other entity based on infringement of intellectual property rights or otherwise.

LICENSEE UNDERSTANDS THAT SOFTWARE IS PROVIDED "AS IS" FOR WHICH NO WARRANTIES AS TO CAPABILITIES OR ACCURACY ARE MADE. INDIANA UNIVERSITY GIVES NO WARRANTIES AND MAKES NO REPRESENTATION THAT SOFTWARE IS FREE OF INFRINGEMENT OF THIRD PARTY PATENT, COPYRIGHT, OR OTHER PROPRIETARY RIGHTS. INDIANA UNIVERSITY MAKES NO WARRANTIES THAT SOFTWARE IS FREE FROM "BUGS", "VIRUSES", "TROJAN HORSES", "TRAP DOORS", "WORMS", OR OTHER HARMFUL CODE. LICENSEE ASSUMES THE ENTIRE RISK AS TO THE PERFORMANCE OF SOFTWARE AND/OR ASSOCIATED MATERIALS, AND TO THE PERFORMANCE AND VALIDITY OF INFORMATION GENERATED USING SOFTWARE.

Indiana University has the exclusive rights to license this product under this license.

3.6 Maui

The maui package contains the following copyrights and licenses.

Moab Scheduling System - End User Open Source License

This software is based on the Moab Scheduling System which was created by Cluster Resources, Inc.

Copyright (C) 1999-2004 Cluster Resources, Inc., all rights reserved.

Moab Scheduling System is a trademark of Cluster Resources, Inc.

This SOFTWARE is bound by an ’End User Open Source’ LICENSE from Cluster Resources Inc. The conditions of the ’End User Open Source’ LICENSE include, but are not limited to the conditions described below.

THE SOFTWARE IS PROVIDED AS IS, AND CLUSTER RESOURCES, INC. (CRI) AND ALL CONTRIBUTING PARTIES DISCLAIM ALL WARRANTIES RELATING TO THE SOFTWARE, WHETHER EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. NEITHER

CRI NOR ANYONE INVOLVED IN THE CREATION, PRODUCTION, OR DELIVERY OF THE SOFTWARE SHALL BE LIABLE FOR ANY INDIRECT, CONSEQUENTIAL, OR INCIDENTAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE SOFTWARE EVEN IF CRI HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES OR CLAIMS. IN NO EVENT SHALL CRI'S LIABILITY FOR ANY DAMAGES EXCEED THE CONSIDERATION PAID FOR THE LICENSE TO USE THE SOFTWARE, REGARDLESS OF THE FORM OF CLAIM. THE PERSON OR ENTITY USING THE SOFTWARE BEARS ALL RISK AS TO THE QUALITY AND PERFORMANCE OF THE SOFTWARE.

By installing or using this SOFTWARE you are accepting a non-exclusive 'End User Open Source' LICENSE from Cluster Resources Inc. and are bound to abide by the following conditions:

1. Inclusion of Notice and Disclaimer

All copies of the SOFTWARE, whether or not for redistribution and whether or not in source code or in binary form must include a conspicuous and appropriate publication of the above copyright notice and disclaimer.

2. Usage

Source and/or binary forms of this SOFTWARE may be used by any 'End User' organization pursuant to the conditions of this and other associated LICENSES at no charge and for an unlimited period of time. An 'End User' organization is defined as an organization that is using this SOFTWARE on their own systems and is not commercially redistributing, modifying, supporting, or providing other services specific to this SOFTWARE to other organizations for profit.

3. Modifications

SOFTWARE may be freely modified by the 'End User' as necessary to meet the needs of the 'End User' LICENSEE'S system. 'End User' may solicit the services of Cluster Resources Inc. or 'Authorized Distribution and Services Partners' of Cluster Resources Inc. that have received express prior written authorization to redistribute, modify or provide services for SOFTWARE. Available services include but are not limited to technical support, training, consultation or optimization services. 'End User' may not solicit or receive this SOFTWARE or services associated to the use, customization, training, development, or support on this SOFTWARE from any organization that is not an 'Authorized Distribution and Services Partner' of Cluster Resources Inc. Any organization that desires to become an 'Authorized Distribution and Services Partner' of Cluster Resources, Inc. may contact us at support@clusterresources.com. 'End User' organizations that desire services from Cluster Resources Inc., or an 'Authorized

Distribution and Services Partner' may contact us using the same email listed above.

4. Distribution

'End User' organizations that are academic and government agencies may redistribute this SOFTWARE subject to the condition that the distribution contains conspicuous publication of the acknowledgement statement found within the LICENSE agreement distributed with this SOFTWARE.

Organizations that are not academic and government agencies including commercial and other for-profit organizations may not redistribute this code or derivations of this code in any form whatsoever, including parts of SOFTWARE incorporated into other software programs without express written permission from Cluster Resources, Inc.

Redistribution of the SOFTWARE in any form whatsoever, including parts of the code that are incorporated into other software programs, must include a conspicuous and appropriate publication of the following acknowledgement:

'This product was developed by Cluster Resources, Inc. Moab Scheduling System is a trademark of Cluster Resources, Inc.'

Any redistribution or modification of the SOFTWARE must, when installed, display the above language, the copyright notice, and the warranty disclaimer.

Each time the SOFTWARE (or any work based on the SOFTWARE) is redistributed, the recipient must automatically receive this LICENSE, copyright notice, and the warranty disclaimer as described in this license agreement, which govern the ability to copy, distribute or modify the SOFTWARE subject to these terms and conditions, and have the choice of accepting or declining the LICENSE.

As the LICENSEE, you shall automatically provide the recipient with a copy of this LICENSE. Further restrictions are not to be imposed on recipients of the SOFTWARE by the LICENSEE beyond those expressly described herein.

5. Use of Modifications

LICENSEES with a redistribution agreement that wish to distribute their modifications (including government and academic institutions) must first send a copy of the modifications along with a brief explanation of why the modification was made and the resulting performance or functionality of the

modifications to Cluster Resources, Inc. at support@clusterresources.com. Failure to send a copy of distributed modifications renders the LICENSE invalid, as well as any LICENSES granted to third parties subsequent to the incorporation of the modifications into SOFTWARE. Any such modification of the SOFTWARE must, when installed, display the LICENSE, the copyright notice, and the warranty disclaimer as described in the LICENSE agreement/s distributed with this SOFTWARE. Those without a LICENSE to redistribute may send modifications to Cluster Resources for evaluation and possible incorporation into SOFTWARE.

Copyright owners of modifications to SOFTWARE hereby grant Cluster Resources, Inc. a non-exclusive, royalty-free, worldwide, irrevocable right and LICENSE to install, use, distribute, sublicense, and prepare derivative works of said modifications. Only organizations receiving an express prior written exclusion to this condition are exempted from providing these non-exclusive rights to Cluster Resources, Inc.

6. Communications about and Endorsement of SOFTWARE and Products/Software Derived from the SOFTWARE

The name 'Moab Scheduling System', 'Moab Scheduler', or any of its variants must not otherwise be used to endorse or to promote products derived from the SOFTWARE without prior written permission from CRI.

Products derived from or incorporating the SOFTWARE in whole or in part shall not contain as part of the product's name any form of the terms 'Cluster Resources, Inc.', 'CRI', 'Moab', 'Moab Scheduling System', 'Moab Scheduler', or 'Supercluster Development Group' unless prior written permission has been received from Cluster Resources, Inc.

All advertising materials for products that use or incorporate features of the SOFTWARE must display the following acknowledgement: 'This product includes software developed by Cluster Resources, Inc. for use in the Moab Scheduling System.'

7. Acceptance of this LICENSE

It is not required that you accept this LICENSE; however, if you do not accept the terms of this LICENSE, you are prohibited by law from installing, using, modifying or distributing the SOFTWARE or any of its derivative works. Therefore, by installing, using, modifying or distributing the SOFTWARE (or any of its derivative works), you have agreed to this LICENSE and have accepted all its terms and conditions.

If any portion of this LICENSE is held invalid or unenforceable under any

particular circumstance, the balance of the LICENSE will continue to apply.

3.7 MPICH

The MPICH OSCAR package contains the following copyrights and licenses.

COPYRIGHT

The following is a notice of limited availability of the code, and disclaimer which must be included in the prologue of the code and in all source listings of the code.

Copyright Notice
 + 1993 University of Chicago
 + 1993 Mississippi State University

Permission is hereby granted to use, reproduce, prepare derivative works, and to redistribute to others. This software was authored by:

Argonne National Laboratory Group

W. Gropp: (630) 252-4318; FAX: (630) 252-5986; e-mail: gropp@mcs.anl.gov
E. Lusk: (630) 252-7852; FAX: (630) 252-5986; e-mail: lusk@mcs.anl.gov
Mathematics and Computer Science Division
Argonne National Laboratory, Argonne IL 60439

Mississippi State Group
N. Doss: (601) 325-2565; FAX: (601) 325-7692; e-mail: doss@erc.msstate.edu
A. Skjellum:(601) 325-8435; FAX: (601) 325-8997; e-mail: tony@erc.msstate.edu
Mississippi State University, Computer Science Department &
 NSF Engineering Research Center for Computational Field Simulation
P.O. Box 6176, Mississippi State MS 39762

GOVERNMENT LICENSE

Portions of this material resulted from work developed under a U.S. Government Contract and are subject to the following license: the Government is granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable worldwide license in this computer

```
software to reproduce, prepare derivative works, and perform publicly
and display publicly.

                                DISCLAIMER

This computer code material was prepared, in part, as an account of
work sponsored by an agency of the United States Government.  Neither
the United States, nor the University of Chicago, nor Mississippi
State University, nor any of their employees, makes any warranty
express or implied, or assumes any legal liability or responsibility
for the accuracy, completeness, or usefulness of any information,
apparatus, product, or process disclosed, or represents that its use
would not infringe privately owned rights.
```

3.8 pfilter

The pfilter OSCAR package contains the following copyrights and licenses.

```
% Copyright 2002 Neil Gorsuch
%
% pfilter is free software; you can redistribute it and/or modify
% it under the terms of the GNU General Public License as published by
% the Free Software Foundation; either version 2 of the License, or
% (at your option) any later version.
%
% pfilter is distributed in the hope that it will be useful,
% but WITHOUT ANY WARRANTY; without even the implied warranty of
% MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
% GNU General Public License for more details.
%
% You should have received a copy of the GNU General Public License
% along with this program; if not, write to the Free Software
% Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
```

3.9 PVM

The PVM OSCAR package contains the following copyrights and licenses.

```
        PVM version 3.4:  Parallel Virtual Machine System
              University of Tennessee, Knoxville TN.
```

```
          Oak Ridge National Laboratory, Oak Ridge TN.
                  Emory University, Atlanta GA.
     Authors:  J. J. Dongarra, G. E. Fagg, G. A. Geist,
                J. A. Kohl, R. J. Manchek, P. Mucci,
        P. M. Papadopoulos, S. L. Scott, and V. S. Sunderam
                  (C) 1997 All Rights Reserved

                               NOTICE

 Permission to use, copy, modify, and distribute this software and its
 documentation for any purpose and without fee is hereby granted
 provided that the above copyright notice appear in all copies and
 that both the copyright notice and this permission notice appear in
 supporting documentation.

 Neither the Institutions (Emory University, Oak Ridge National
 Laboratory, and University of Tennessee) nor the Authors make any
 representations about the suitability of this software for any
 purpose.  This software is provided ''as is'' without express or
 implied warranty.

 PVM version 3 was funded in part by the U.S. Department of Energy,
 the National Science Foundation and the State of Tennessee.
```

3.10 SIS

The SIS OSCAR package contains the following copyrights and licenses.

The SystemImager package contains:

```
#  Copyright (C) 1999-2001 Brian Elliott Finley
#  <brian.finley@baldguysoftware.com>
```

The System Configurator and SystemInstaller packages contain:

```
#   Copyright (c) 2001 International Business Machines

# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2 of the License, or
# (at your option) any later version.
```

```
# This program is distributed in the hope that it will be useful,
# but WITHOUT ANY WARRANTY; without even the implied warranty of
# MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
# General Public License for more details.

# You should have received a copy of the GNU General Public License
# along with this program; if not, write to the Free Software
# Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307
# USA
```

The AppConfig package contains:

```
Copyright (C) 1998 Canon Research Centre Europe Ltd.  All Rights
Reserved.

This module is free software; you can redistribute it and/or modify it
under the same terms as Perl itself.
```

The Perl-Tk package contains:

```
# Copyright (c) 1992-1994 The Regents of the University of California.
# Copyright (c) 1994 Sun Microsystems, Inc.
# Copyright (c) 1995-1999 Nick Ing-Simmons. All rights reserved.
# This program is free software; you can redistribute it and/or

# modify it under the same terms as Perl itself, subject
# to additional disclaimer in Tk/license.terms due to partial
# derivation from Tk8.0 sources.
```

The Syslinux package contains:

```
SYSLINUX is Copyright 1994-2001 H. Peter Anvin, and is free software;
you can redistribute it and/or modify it under the terms of the GNU
General Public License as published by the Free Software Foundation,
Inc., 675 Mass Ave, Cambridge MA 02139, USA; either version 2 of the
License, or (at your option) any later version.
```

The tftp-hp package contains:

```
Copyright (c) 1983, 1993
The Regents of the University of California.  All rights reserved.
```

3.11 SSSLib

The SSSLib OSCAR package contains the following copyrights and licenses.

SSS-MCS Public License

Copyright (c) University of Chicago 1999.

1. The "Software", below, refers to the MCS Systems Administration Toolkit (in either source-code, or binary form) and a "work based on the Software" means a work based on either the Software, on part of the Software, or on any derivative work of the Software under copyright law: that is, a work containing all or a portion of the Software either verbatim or with modifications. Each licensee is addressed as "you" or "Licensee."

2. The University of Chicago as Operator of Argonne National Laboratory is the copyright holder in the Software. The copyright holder reserves all rights except those expressly granted to the Licensee herein and U.S. Government license rights.

3. A copy or copies of the Software may be given to others, if you meet the following conditions:

   a) Copies in source code must include the copyright notice and this license.

   b) Copies in binary form must include the copyright notice and this license in the documentation and/or other materials provided with the copy.

4. All advertising materials, journal articles and documentation mentioning features derived from or use of the Software must display the following acknowledgment:

   "This product includes software developed by the MCS Systems Group at Argonne National Laboratory (http://www.mcs.anl.gov/systems/)."

   In the event that the product being advertised includes an intact distribution of the Software (with copyright and license included) then this clause is waived.

5. You are encouraged to package modifications to the Software separately,

as patches to the Software.

6. If you modify a copy or copies of the Software or any portion of it, thus forming a work based on the Software, and give a copy or copies of such work to others, either in source code or binary form, you must meet the following conditions:

   a) The Software must carry prominent notices stating that you changed specified portions of the Software.

   b) The Software must display the following acknowledgment: "This product includes software developed by and/or derived from work of Argonne National Laboratory to which the U.S. Government retains certain rights."

7. LICENSEE AGREES THAT THE EXPORT OF GOODS AND/OR TECHNICAL DATA FROM THE UNITED STATES MAY REQUIRE SOME FORM OF EXPORT CONTROL LICENSE FROM THE U.S. GOVERNMENT AND THAT FAILURE TO OBTAIN SUCH EXPORT CONTROL LICENSE MAY RESULT IN CRIMINAL LIABILITY UNDER U.S. LAWS.

8. Portions of the Software resulted from work developed under a U.S. Government contract and are subject to the following license: the Government is granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable worldwide license in this computer software to reproduce, prepare derivative works, and perform publicly and display publicly.

9. The Software was prepared, in part, as an account of work sponsored by an agency of the U.S. Government. Neither the United States, nor the University of Chicago, nor any of their employees, makes any warranty express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.

10. IN NO EVENT WILL THE UNITED STATES, OR THE UNIVERSITY OF CHICAGO BE LIABLE FOR ANY DAMAGES, INCLUDING DIRECT, INCIDENTAL, SPECIAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM EXERCISE OF THIS LICENSE AGREEMENT OR THE USE OF THE SOFTWARE.

END OF LICENSE

3.12 Switcher

The Switcher OSCAR package contains the following copyrights and licenses.

The env-switcher package contains:

```
Software License for Env-switcher

Copyright (c) 2002 The Trustees of Indiana University.
                   All rights reserved.

Indiana University has the exclusive rights to license this product
under the following license.

Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

1) All redistributions of source code must retain the above
   copyright notice, the list of authors in the original source code,
   this list of conditions and the disclaimer listed in this license;
2) All redistributions in binary form must reproduce the above
   copyright notice, this list of conditions and the disclaimer listed
   in this license in the documentation and/or other materials
   provided with the distribution;
3) Any documentation included with all redistributions must include
   the following acknowledgement:

    "This product includes software developed at the Pervasive
    Technology Labs at Indiana University.  For technical information
    contact Andrew Lumsdaine at the Pervasive Technology Labs at
    Indiana University.  For administrative and license questions
    contact the Advanced Research and Technology Institute at 1100
    Waterway Blvd. Indianapolis, Indiana 46202, phone 317-274-5905,
    fax 317-274-5902."

Alternatively, this acknowledgement may appear in the software itself,
and wherever such third-party acknowledgments normally appear.

4) The name "Env-switcher" or shall not be used to endorse or promote
   products derived from this software without prior written
   permission from Indiana University.  For written permission, please
   contact Indiana University Advanced Research & Technology
   Institute.
```

```
5) Products derived from this software may not be called
   "Env-switcher", nor may "Env-switcher" appear in their name,
   without prior written permission of Indiana University Advanced
   Research & Technology Institute.

Indiana University provides no reassurances that the source code
provided does not infringe the patent or any other intellectual
property rights of any other entity.  Indiana University disclaims any
liability to any recipient for claims brought by any other entity
based on infringement of intellectual property rights or otherwise.

LICENSEE UNDERSTANDS THAT SOFTWARE IS PROVIDED "AS IS" FOR WHICH NO
WARRANTIES AS TO CAPABILITIES OR ACCURACY ARE MADE. INDIANA UNIVERSITY
GIVES NO WARRANTIES AND MAKES NO REPRESENTATION THAT SOFTWARE IS FREE
OF INFRINGEMENT OF THIRD PARTY PATENT, COPYRIGHT, OR OTHER PROPRIETARY
RIGHTS.  INDIANA UNIVERSITY MAKES NO WARRANTIES THAT SOFTWARE IS FREE
FROM "BUGS", "VIRUSES", "TROJAN HORSES", "TRAP DOORS", "WORMS", OR
OTHER HARMFUL CODE.  LICENSEE ASSUMES THE ENTIRE RISK AS TO THE
PERFORMANCE OF SOFTWARE AND/OR ASSOCIATED MATERIALS, AND TO THE
PERFORMANCE AND VALIDITY OF INFORMATION GENERATED USING SOFTWARE.

Indiana University has the exclusive rights to license this product
under this license.
```

The modules package contains:

```
                    GNU GENERAL PUBLIC LICENSE
                       Version 2, June 1991

 Copyright (C) 1989, 1991 Free Software Foundation, Inc.
     59 Temple Place, Suite 330, Boston, MA  02111-1307  USA
 Everyone is permitted to copy and distribute verbatim copies
 of this license document, but changing it is not allowed.

...
```

3.13 Warehouse

warehouse is licensed for use by Educational and Research institutions only. Such institutions are allowed to modify the software for their own use, however, distributing modified copies is *expressly forbidden*.

An open-source type license is pending, but as of the time of this writing (March 18, 2004) this has not been approved by the University of Illinois, the holder of the copyright on this software.

3.14 Xerces

The Xerces package contains the following copyright and licenses.

```
/*
 * The Apache Software License, Version 1.1
 *
 *
 * Copyright (c) 1999-2001 The Apache Software Foundation.  All rights
 * reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in
 *    the documentation and/or other materials provided with the
 *    distribution.
 *
 * 3. The end-user documentation included with the redistribution,
 *    if any, must include the following acknowledgment:
 *       "This product includes software developed by the
 *        Apache Software Foundation (http://www.apache.org/)."
 *    Alternately, this acknowledgment may appear in the software itself,
 *    if and wherever such third-party acknowledgments normally appear.
 *
 * 4. The names "Xerces" and "Apache Software Foundation" must
 *    not be used to endorse or promote products derived from this
 *    software without prior written permission. For written
 *    permission, please contact apache@apache.org.
 *
 * 5. Products derived from this software may not be called "Apache",
 *    nor may "Apache" appear in their name, without prior written
 *    permission of the Apache Software Foundation.
 *
 * THIS SOFTWARE IS PROVIDED ‘‘AS IS’’ AND ANY EXPRESSED OR IMPLIED
 * WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES
 * OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
 * DISCLAIMED.  IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR
 * ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
* LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF
* USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND
* ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
* OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT
* OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
* ====================================================================
*
* This software consists of voluntary contributions made by many
* individuals on behalf of the Apache Software Foundation and was
* originally based on software copyright (c) 1999, International
* Business Machines, Inc., http://www.ibm.com.  For more
* information on the Apache Software Foundation, please see
* <http://www.apache.org/>.
*/
```