M3C – An Architecture for Monitoring and Managing Multiple Clusters

Al Geist, Stephen Scott, and Jens Schwidder
Oak Ridge National Laboratory
Oak Ridge, TN 37830-6367

Contact:
Al Geist
Oak Ridge National Laboratory
P.O.Box 2008
Oak Ridge, TN 37831-6367

gst@ornl.gov
phone: (865) 574-3153
fax: (865) 574-0680

Abstract

While PC clusters now provide the aggregate power of supercomputers of just a few years ago, they lack the integrated monitoring and administrative tools that are needed to make them easy to use. Simple tasks like performance monitoring or installing software can be quite difficult when the "computer" is rows of PC towers. This paper describes a Java-based tool suite called M3C (Monitoring and Managing Multiple Clusters) being developed at Oak Ridge National Laboratory to monitor and administer PC clusters. M3C allows a single system administrator to monitor not just one but several clusters at the same time, which reduces the overhead cost of managing these clusters. The integrated tool suite includes tools for cluster users as well as system administrators. M3C provides a simple, intuitive, web based front-end to what is now an *ad hoc* set of scripts and programs to monitor PC clusters. This paper describes the overall design of M3C and provides details about each of the six tools presently integrated into the suite.

Keywords: PC clusters, Java, monitoring, administration

M3C – An Architecture for Monitoring and Managing Multiple Clusters

Al Geist, Stephen Scott, and Jens Schwidder*
Oak Ridge National Laboratory
Oak Ridge, TN 37831-6367

Abstract

While PC clusters now provide the aggregate power of supercomputers of just a few years ago, they lack the integrated monitoring and administrative tools that are needed to make them easy to use. Simple tasks like performance monitoring or installing software can be quite difficult when the "computer" is rows of PC towers. This paper describes a Java-based tool suite called M3C (Monitoring and Managing Multiple Clusters) being developed at Oak Ridge National Laboratory to monitor and administer PC clusters. M3C allows a single system administrator to monitor not just one but several clusters at the same time, which reduces the overhead cost of managing these clusters. The integrated tool suite includes tools for cluster users as well as system administrators. M3C provides a simple, intuitive, web based front-end to what is now an *ad hoc* set of scripts and programs to monitor PC clusters. This paper describes the overall design of M3C and provides details about each of the six tools presently integrated into the suite.

Keywords: PC clusters, Java, monitoring, administration

Introduction.

PC clusters are quickly becoming a popular scientific computing resource. This trend is being driven by the power of Pentium and Alpha CPUs, the low cost of COTS hardware, and the emergence of gigabit per second interconnection technologies. Together these allow institutions to hook together tens or even hundreds of computers at about one fifth the cost of a comparable commercial multiprocessor. What is often overlooked in calculating this low price is the cost of administration and management of the system software across a rack of PCs. For small clusters with fewer than 32 computers it is possible to install software by treating each computer separately or NFS mount all to one primary server; for clusters larger than this, administration complexity requires a more automated means to perform operations across the whole cluster. There is presently a lack of standard tools for managing clusters. This paper describes M3C, a Java-based tool suite being developed at Oak Ridge National Laboratory to fill this gap in cluster management software.

There are several groups around the world building ad hoc scripts for administering their clusters [6, 10]. Sandia National Laboratory has the largest PC cluster in the world and has been studying the scalability of administration tools [3, 9] for several years. Some tools such as schedulers have been ported from other systems to PC clusters; examples include Portable Batch Scheduler (PBS) [5], LSF [7], and Condor [1]. Cluster integrators such as Alta and VA Linux Systems are developing scripts and management tools [11]

* This research sponsored by the Department of Energy MICS office.

but are not considering how an organization with clusters from different vendors manages these clusters.

M3C is being developed to provide an integrated front-end to existing/future scripts and schedulers. It is not a replacement for software like PBS. Rather, M3C collects the information from remote users that tools like PBS then use and conversely collects monitoring information from the cluster that it displays to the users.

The next section describes the architecture of M3C and how it assists in the monitoring and administration of multiple clusters. This is followed by a description of how M3C interfaces with different backend tools. The rest of the paper presents the purpose, operation, and view of the six tools integrated into the M3C tool suite.

Multiple Clusters

Although the cost and rapid performance improvements in PCs have led to organizations having several clusters (of different generations), no tools presently allow the monitoring and management of multiple clusters. M3C tool suite is being designed to address this need. Initial M3C uses have been to administer both single and multiple clusters within Oak Ridge National Laboratory, which has ten PC clusters installed across various divisions. M3C leaves the control, policies, and cluster descriptions at the cluster sites. The cluster owners always retain control over access to their resources and what is run on them. M3C is a distributed program with pieces installed at the cluster sites and one or more graphical user interfaces running out on the Web. (See Figure 1) The respective cluster owners control the pieces installed at the cluster sites. Together all the distributed processes form the M3C tool suite. The cluster owners are free to choose their favorite backend software for carrying out the requests made by M3C users. By separating M3C from the backend software, it can be used to manage several clusters even when they each use different packages for monitoring, scheduling, and installing software.

M3C is a stand-alone tool suite for managing multiple clusters. It does not assume or require an existing infrastructure. Nevertheless, M3C can be used within a GRID [4] environment and use GRID schedulers and information servers if that is the policy of the cluster owner.

Figure 1 shows a diagram of the M3C architecture with two users. The tool suite is composed of three pieces: a Java applet-based GUI, a CGI script to read and write description files, and a proxy that allows the applet and the CGI scripts to work across multiple cluster domains. The first user is connected to a single CGI script and sees two clusters in her GUI. In this case the cluster owner has configured the CGI script to include multiple clusters within a single administrative domain. The second user is connected to three administrative domains and can monitor four clusters in M3C. In the latter case a proxy coordinates access to several clusters from a single GUI. The proxy appears to each of the CGI scripts as the GUI and appears to the GUI as the CGI script. The second user could be a system administrator in charge of maintaining clusters across several divisions, or he could be a scientist who has been given accounts on all four clusters because his research is in collaboration with each of the cluster owners. In the

latter case, the scientist could use M3C to submit jobs to several clusters simultaneously, or he could reserve a block of time across several clusters in order to make a large run.

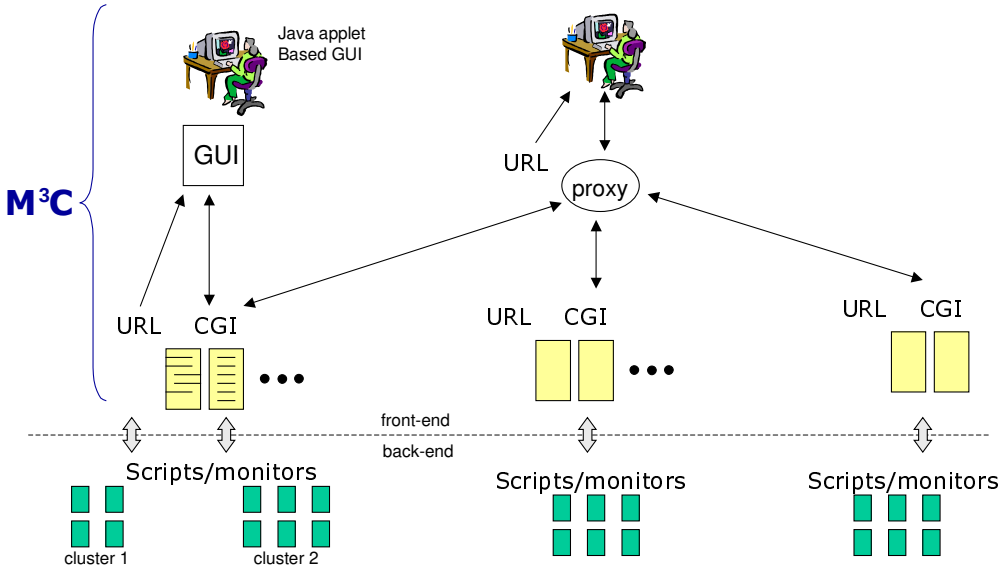

Figure 1. Diagram of M3C architecture

The CGI script in M3C performs four simple functions. It reads cluster description files created either manually or automatically. These files can contain any number of keywords/values that describe the properties of the cluster's nodes (OS, CPU, memory, NIC cards, etc.) The cluster owner, who is responsible for creating the cluster description file, decides how much information is provided to M3C users. The second CGI function is to read and update node reservation files with user requests and partition information. The third function is to read the cluster monitor files generated dynamically by monitors supplied by the cluster vendor or owner. The fourth function is to receive and process requests from M3C users for job submission and software installation. These requests are appended to particular middleware files for handling by backend processes.

There is no communication or synchronization among the CGI scripts in different domains. Each works independently, servicing requests from Java applets. The advantage of this design is that one cluster is unaffected by the behavior or availability (faults) of other clusters. So while M3C can interact and coordinate across multiple clusters, it does not allow the faults in one cluster to negatively impact any of the other clusters.

The proxy is responsible for the coordination of M3C actions across multiple clusters. For example, if user 2 asks to reserve nodes in all four clusters, then the proxy sends out three asynchronous requests to each of the CGI scripts. If, immediately before, user 1

reserved all the nodes in cluster 1, the proxy gets back some failure and some success replies, which it has to resolve before replying to user 2's GUI. Similar problems can occur if a cluster drops offline or one of the asynchronous requests times out. The default behavior of M3C is to report the problem to the user and suggest refreshing the displays.

Interface to back-end tools

The M3C middle layer is composed of CGI scripts that read monitoring data collected by back-end tools and delivers the information to the web based GUI. The scripts also take requests from multiple remote uses and place them in files for back-end tools such as Beboot from Rembo Technolgy [8] to read and operate on. The file format is kept purposely flexible and human readable to allow manual configuration and manipulation by the cluster owner. Each M3C tool in the suite reads and/or writes to its own file, preventing inadvertent interference between different tools and providing a clean method to add new tools to the suite. The backend tools can be running anywhere as long as they can read/write a cluster's M3C files. It is expected that the backend tools will modify the files when the requested operations are done, for example, deleting a job from the job submission file when it is completed.

The file-based interface to the back-end tools places policy control directly in the hands of the cluster owner. M3C is designed to support multiple user priority levels so that operations requested by the system administrator are handled before normal user requests. For example, the owner can take control of the cluster if a sponsor needs a demonstration on the spur of the moment. Initially there are five levels of user priorities designated: owner/administrator, important local user, local user, important external user, external user. These designations are just suggestions. The key concept is that cluster users, or more specifically M3C users, can be divided into five different priority levels by the cluster owners. Policies for each priority level are set by the owner and enforced by the back-end tools.

Integrated Tool Suite

M3C presently contains six tools, but is extensible to allow more tools to be added. All six tools are designed for remote cluster administration. Four of the tools also can be used by normal cluster users to simplify the monitoring and running of their jobs. The M3C suite includes: a monitor view that allows multiple users to see the real-time load and other metrics on specified nodes; a reservation tool that allows multiple users to easily see existing reservations on clusters and to reserve a set of nodes for any of the available times; a job submission tool that allows users to specify the number of nodes and other job requirements and submit this to a cluster's batch queue system; a software install tool that allows a user to install his data or programs on sets of nodes and allows system administrators to easily install upgrades or patches across multiple clusters; an administrative tool that allows the system administrator to reboot or shutdown selected nodes; and finally a partition tool for dividing a large cluster into sets of interactive and batch nodes.

Integrated across all these tools is the ability to display detailed information about nodes

and the ability to search for nodes both within and across clusters that have specified characteristics, for example, system load, or a particular version of the OS, or a certain software library.

Cluster monitor

A tool of wide utility to cluster administrators as well as cluster users is the M3C cluster monitor tool. As presently implemented the monitor view displays the load, temperature, and list of the top tasks running on the selected clusters. These three properties were chosen in order to illustrate/investigate how the monitor tool could display data graphically (load), by value (temperature), and by a text string (tasks). See Figure 2.

The M3C user can specify the update frequency for the monitor view. By default the view is only updated when the "refresh" button is pressed. This provides the minimum intrusion on the web system and the cluster. A pull down menu allows the user to set an update frequency in a range between 1 second and 1 minute.

Backend scripts and daemons running on the clusters collect the information displayed in the M3C monitor tool. The cluster owner can install any monitor scripts he wants. M3C does not care whether homemade scripts or commercial software are used to gather the information. The M3C monitor tool simply reads the information placed in a file by the scripts. The update frequency set in the M3C tool just defines how frequently the user views are updated; it is independent of the frequency that the cluster updates the monitor file. One reason they are independent is because multiple M3C tools can be running at the same time, each with different update frequencies set. Another reason they are independent is to prevent a M3C user from adversely impacting other users' cluster performance by setting a high update frequency.

Load, temperature, and running processes are presently displayed but other node properties can easily be added to the M3C monitor view. For example, performance metrics like number of FLOPS, cache misses, and memory hot spots are now available from on-chip registers. Portable tools such as PAPI [12] are being built to read this information. Integration of PAPI information with M3C is under investigation. If no monitors are running, then the monitor view shows the fields as blank.

The *find* feature in M3C can be used to search for monitor properties as well as static properties. For example, the user may wish to find all nodes running tasks in his "hung" application. The find and select feature turns out to be important for scalability. Although the number of nodes in a cluster is often much larger than can be displayed on a screen, the find feature allows the user or administrator to quickly locate all the nodes with given properties.

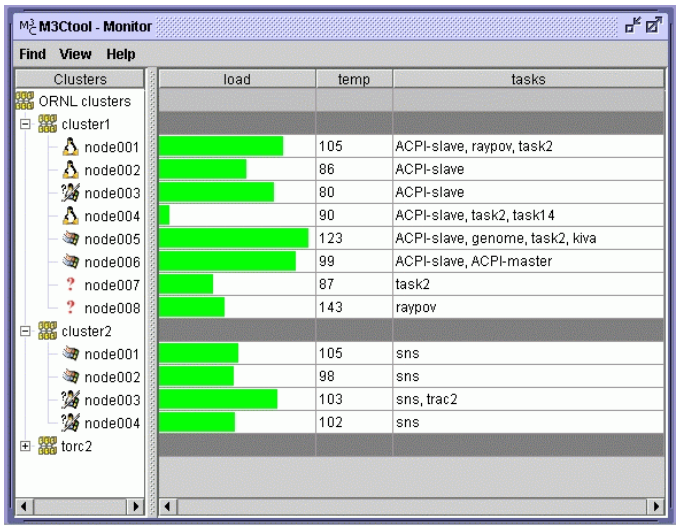

Figure 2. Monitoring node status on multiple clusters

Another scalability consideration built into the M3C monitor view is error detection. Because of the limited number of nodes that can be displayed on the screen, there is concern that the system administrator may not catch a fault or potential problem among the many nodes and clusters that the M3C tool can monitor. To make it easy to pinpoint the location of errors, the node name in the tree view turns red if the M3C tool detects a problem within that node. If any node in a cluster is red then the cluster name in the tree view turns red, and so on propagating up the tree. This allows the administrator to quickly follow the red branches down the tree to the problem node, and then double click on this node to display all known information about this node. In the M3C "node properties" window the detected problem is highlighted in red.

Alarms can be set in the cluster configuration file for any of the node properties that are being monitored. To illustrate this functionality, the present M3C tool has a maximum temperature alarm for nodes. (Nodes that are running hot are more likely to fail.) If the monitor-reported temperature value exceeds that node's maximum temperature parameter, then M3C raises an alarm by highlighting the cluster name and node name in red. This alarm is visible in all the tools in the suite so that it is not necessary to be in the monitor view to see the warning.

Multi-user reservation

Presently if a user wants to run a simulation on a dedicated block of cluster nodes, he contacts the system administrator who coordinates the request with other users' requests for dedicated time. The reservation tool in M3C provides a way for multiple users to see what times and which nodes are available on the cluster and allows the user to make reservations for any times that are not already booked. Figure 3 shows a screenshot of the reservation tool in use by a user named Al. All reservations made by others appear in yellow, while all reservations made by the user are delineated in green. The tool allows nodes to be blocked out across multiple clusters if the user needs more nodes of a certain

type than exist in any one cluster. System administrators can use this tool to block out and coordinate times for backups or system maintenance on several clusters.

In order to help the user find which and how many nodes have particular properties in the clusters managed by M3C, the tool suite includes a “Find” function that works with all six tools. In the case of reservations, the user’s application may need nodes with a certain amount of memory or perhaps with a particular software package installed. System administrators may need to apply a patch to all nodes with a certain version of the OS. The Find function highlights (selects) all the nodes that match all the user specified requirements. M3C has no knowledge about the cluster nodes beyond what the cluster owner is willing to share either statically in the description file or dynamically from monitors.

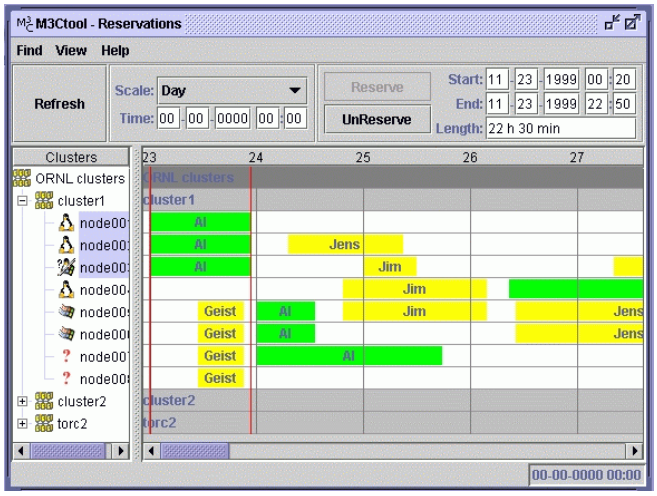

Figure 3. Interface for making cluster reservations

To use the reservation tool, a set of nodes from the “tree” view is selected. Then the time block when these nodes are desired is specified by left and right clicking in the time view. When the reserve button is pressed, the request is sent to the CGI script(s), which checks that another user has not reserved the same nodes in the last few moments. If the nodes are still available, then the reservation tool colors the reservation green and places the user’s name in the block. If the nodes are not available, an error is returned and the user can press the “refresh” button and see the latest reservation data and adjust the request accordingly.

The reservation view also shows which nodes are set aside into batch partitions (or other administrator-defined partitions). On large systems, batch partitions are often set to grow larger on nights and weekends and to shrink during workdays when the interactive partition is heavily used. This cyclical availability is visible in the reservation view.

The reservation tool allows the user to change the time scale in the window from minutes to days, weeks, or months and allows the user to specify the starting time of the display window. By default the starting time is when M3C is executed. Combining these two features it is possible to reserve a 30-minute block of time two months into the future.

If simulation requirements change, the M3C tool allows the user who made the reservation to "unreserve" the selected nodes, which then go back into the interactive pool for other users.

Job submission

A common way to share resources on a large cluster is to run a batch scheduler. M3C includes a tool to collect information for parallel batch schedulers such as PBS and Condor. The submit job tool is shown in Figure 3. The user supplies the job name or path to the executable, and any requirements for the job. For example, does it have to run on a particular cluster? How many nodes does it require? Are there any special node requirements? In Figure 4 the climate simulation job PCCM3 is allowed to run on any cluster as long as the nodes are running Linux. Other requirements could be added.

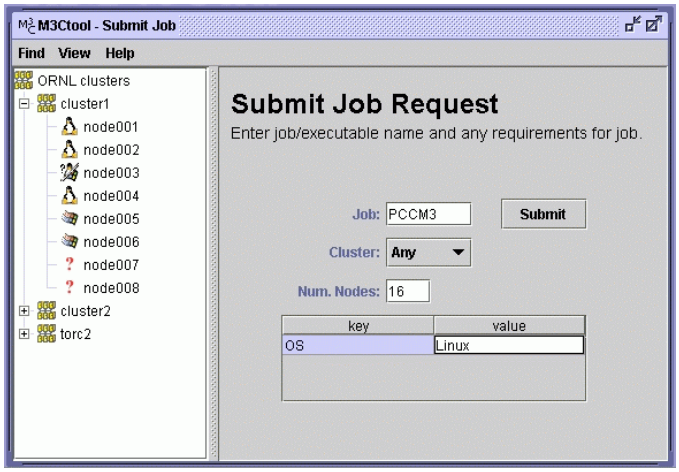

Figure 4. M3C interface to batch schedulers

The tree view is not used in the submit job tool except as a means to find out particular properties that the user may want to specify in the job requirements. (Double clicking on a node in the tree view brings up a small information box listing all known properties of that node.) Which nodes that the job runs on is left to the discretion of the batch queue system – selecting nodes in the tree view is ignored.

Install Software

M3C provides a tool for installing software on a set of selected nodes. This is useful to the system administrator who needs to install the latest security patch to the Linux nodes in all the clusters for which he is responsible. This tool is also useful to the scientist who wants to preload data files or his executable on a set of cluster nodes before starting a simulation. An example of the install tool in use is shown in Figure 5.

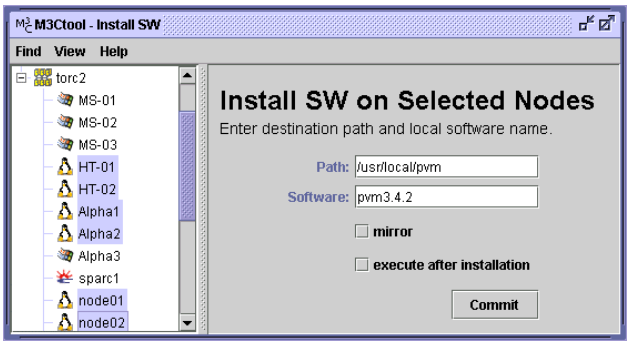

Figure 5. Installing a software package on all Linux nodes

The M3C user selects the nodes either manually or using the *find* feature, then fills out the destination path where the software is to be installed on each of the nodes and the name of the local file or directory to be installed. One of the options is to execute the software after it is installed. This could be useful to the system administrator who is installing monitors across the clusters. It could also be useful to users wanting to load and run an application.

One way that cluster administration has been simplified is to build one node with the desired OS version and software libraries and then automatically mirror this node's disk on all the other nodes in the cluster. The M3C install tool provides an interface to the backend scripts that install software by mirroring.

Reboot

A host of "system administrator only" tasks are needed to manage clusters. If maintenance needs to be performed then all the affected nodes need to be shut down. After new software is installed, systems often need rebooting. If an application hangs a set of nodes these may need to be rebooted in order for the next user to run his application. M3C provides an interface to the clusters for performing tasks that fall in the administrator-only category. Presently there are four functions built into the M3C interface: reboot, shutdown, add user, and delete user. The use of this tool is similar to the other M3C tools. The administrator selects a set of nodes, and then presses the function he wants applied. In the case of "add user" a dialog box opens to fill out required information. A screen shot of the M3C reboot tool is shown in Figure 6.

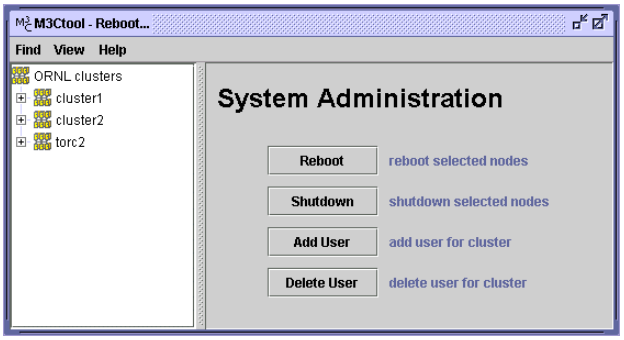

Figure 6. M3C tool for performing system administration functions

Partition a single cluster

Small clusters are often used as a single dedicated computational resource, but large clusters with hundreds of nodes are more likely to be partitioned into groups of nodes dedicated to batch jobs, groups of nodes for interactive use, and in special cases, groups of nodes dedicated to a single user for a large run. In the next few years we will see more and more of these large clusters [2, 9]. M3C has a tool to assist in setting up partitions within large clusters. These partitions then appear in the M3C reservations time view so that users can easily see what parts of the cluster are available for interactive use. Figure 7 shows a screen shot of the tool.

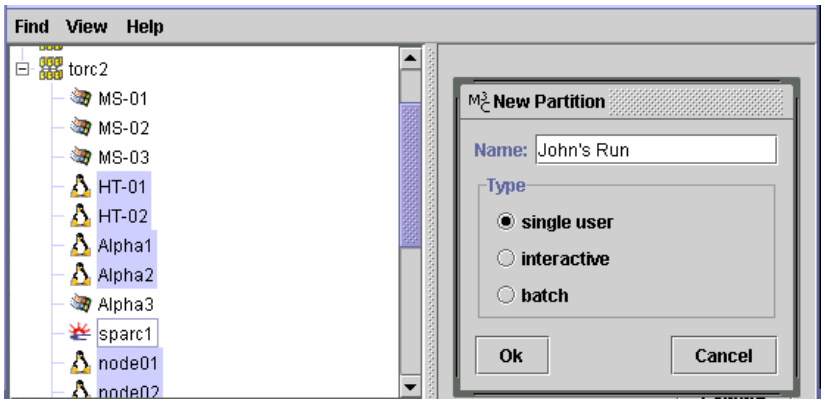

Figure 7. Cluster Partitioning tool being used to set up a large run

Partitions often vary between day and night shifts in order to provide larger interactive access during the workdays. Thus partitions are a function of both nodes and time just like reservations. In the M3C tool suite, partitions are treated as a special class of reservation. The partitions created with the M3C partition tool appear in the reservations view highlighted in a special color to distinguish them from normal reservations.

Conclusion

The Java based tool suite called M3C is being developed at Oak Ridge National Laboratory to monitor and manage multiple PC clusters. M3C architecture allows a single system administrator to manage several clusters at the same time, and it allows users the potential to utilize and monitor nodes across several clusters. M3C is designed so that information about a cluster and policies for its use are left under the control of the cluster owner. The operations across clusters are asynchronous so that policies or problems in one cluster do not affect the other clusters. M3C is designed to work with a wide range of backend scripts, schedulers, and monitors so that cluster owners can use the software of their choice.

Six tools are presently incorporated into the suite: a monitor view that allows multiple users to see the real-time load and processes on specified nodes and to be able to set alarms for particular conditions; a reservation tool that allows multiple users to easily see existing reservations on clusters and to reserve a set of nodes for any of the available times; a job submission tool that allows users to specify the number of nodes and other job requirements and submit this to the cluster's batch queue system; a software install tool that allows a user to install his data or programs on sets of nodes and allows system

administrators to easily install upgrades or patches across multiple clusters; an administration tool that allows a system administrator to reboot or shutdown selected nodes; and finally a partition tool for partitioning a large cluster into sets of interactive and batch nodes.

M3C is being developed as an open source project and the software is available from Oak Ridge National Laboratory. See web site www.csm.ornl.gov/torc for the latest release.

References

1. Basney, J., and Livny, M., Deploying a High Throughput Computing Cluster, to appear in *High Performance Cluster Computing*. Vol. 1, Prentice Hall, Inc., 1999. http://www.cs.wisc.edu/condor/

2. Bennett, Forrest H III, Koza, John R., Shipman, James, and Stiffelman, Oscar. Building a parallel computer system for $18,000 that performs a half peta-flop per day. In Banzhaf, Wolfgang, Daida, Jason, Eiben, A. E., Garzon, Max H., Honavar, Vasant, Jakiela, Mark, and Smith, Robert E. (Eds). *Proc. Genetic and Evolutionary Computation Conference,* Orlando, Florida, Morgan Kaufmann, 1999. http://www.genetic-programming.com/machine1000.html

3. Evensky, D., et al., Lilith: A Software Framework for the Rapid Development of Scalable Tools for Distributed Computing, *Proc. 7th IEEE International Symposium on High Performance Distributed Computing*, Chicago, IL, IEEE Computer Society, 1998. http://dancer.ca.sandia.gov/Lilith/

4. Foster, I., and Kesselman, C. (Eds), *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers, 1998.

5. NASA Portable Batch System, http://parallel.nas.nasa.gov/Parallel/PBS

6. Pant, A., Management Tools for NT Supercluster, *Proc. Beowulf and Beyond Conference*, Oak Ridge, TN, 1999. http://www.csm.ornl.gov/JPC4/

7. Platform Computing, LSF Suite http://www.platform.com/platform/platform.nsf/webpage/LSFOver

8. Rembo Technology SàRL, Remote reboot technology http://www.beoboot.com/

9. Riesen, R., Brightwell, R., Fisk, L. A., Hudson, T., Otto, J., and Maccabe, A. B., Cplant, *Proc. Second Extreme Linux Workshop*, Monterey, California, June 1999. http://www.cs.sandia.gov/cplant/

10. Sterling, T., Salmon, J., Becker, D., Savarese, D., *How to Build a Beowulf: A Guide to the Implementation and Application of PC Clusters*, MIT Press, 1999

11. VA Linux Systems, VACM Cluster Management. http://www.valinux.com/projects/vacm/

12. Mucci, P., Browne, S., Ho, G., Deane, C., PerfAPI—Performance Data Standard and API. http://icl.cs.utk.edu/projects/papi/