BUILDING AND MANAGING THE *e*MINERALS CLUSTERS:

A CASE STUDY IN GRID-ENABLED CLUSTER OPERATION

This article describes the configuration, monitoring, management, and use of compute clusters in an environment in which users access resources via grid middleware tools rather than direct logins. The work described here shows that setting up and managing clusters within grid environments is feasible without large investments in support or user training.

*e*Minerals[1] is a testbed project supported by the UK eScience program. The project's aim is to create a multi-institution collaborative infrastructure that uses grid technologies to support a team of molecular-simulation scientists working together on environmental science problems. Examples of work carried out within the *e*Minerals project include simulations of weathering processes, pollutant molecules and metal ions on mineral surfaces, and the development of nuclear-waste-encapsulation materials. The project members' applications show great diversity, with a significant degree of heterogeneity in compute and data-management requirements. The simulation methodologies include static energy relaxation for computation of properties and energies, lattice dynamics for thermodynamic properties, molecular dynamics simulations for studies of dynamic processes at elevated temperatures, and Monte Carlo simulations for studies of ordering processes.

As part of its work, the *e*Minerals project has set up a mid-sized grid infrastructure, called the *e*Minerals mini-grid.[2] The work has been guided by several key user requirements, including the need to provide

- support for high-throughput workflows (demand for high-performance computing is better met using national supercomputer facilities);
- support for heterogeneous applications (unlike other major grid projects, which tend to focus on small sets of applications and data types);
- ease of user access; and
- proper support for data management (thus enabling output files to be returned to desktops without significant overhead).

Another important requirement is that systems management should be as easy as possible, and certainly shouldn't require dedicated staff time.

This article describes our experiences in setting up three Linux clusters specifically for operation within a mid-sized grid environment. We discuss issues such as installation, configuration, data management, maintenance and management, job monitoring, and data backup. The data-management issue is one of the more novel features of our clusters. We use the Storage Resource Broker (SRB) as our primary data-management tool,[3,4] hav-

1521-9615/05/$20.00 © 2005 IEEE
Copublished by the IEEE CS and the AIP

RICHARD P. BRUIN AND MARTIN T. DOVE
Cambridge University
MARK CALLEJA
Cambridge eScience Centre
MATTHEW G. TUCKER
ISIS Neutron Scattering Facility

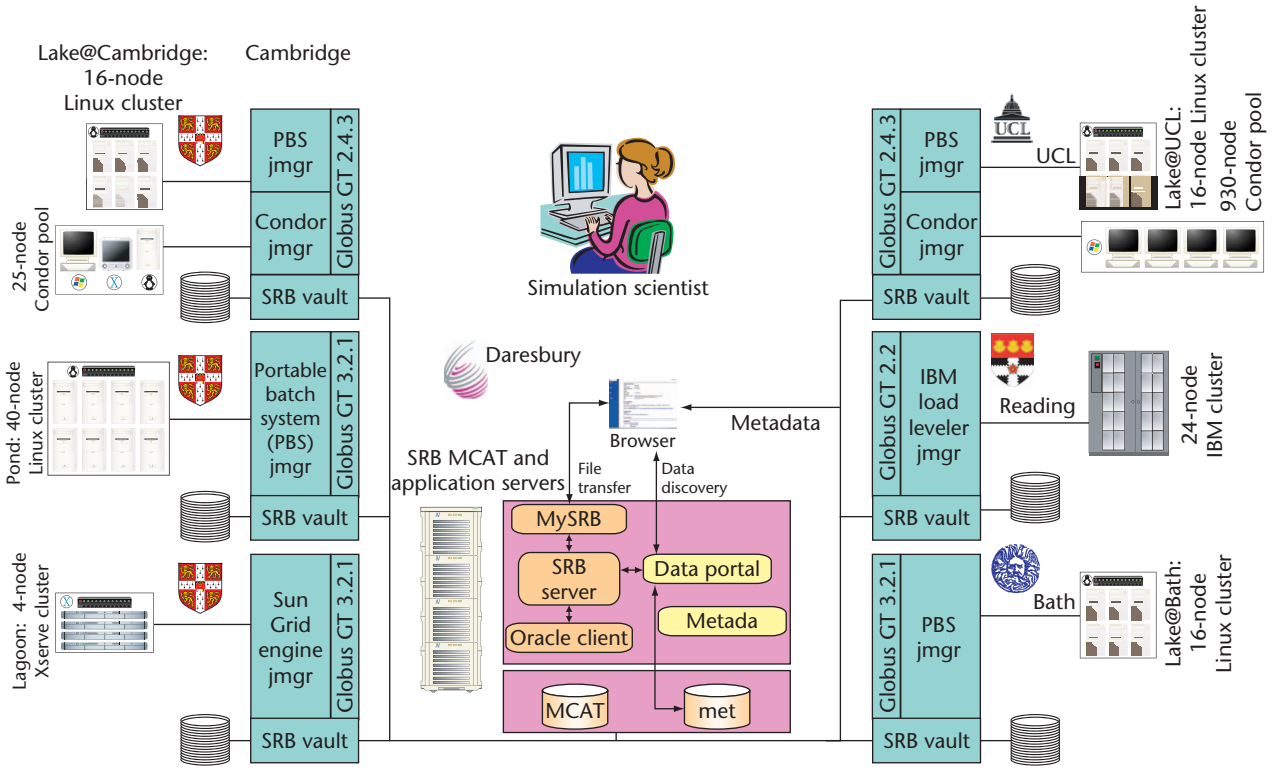

Figure 1. The *e*Minerals mini-grid. The compute infrastructure consists of three PC/Linux clusters (labeled Lake@Cambridge, Lake@UCL, and Lake@Bath), a similar cluster (Pond) adapted in light of our experience with the Lake clusters, a small high-memory Apple Xserve cluster (Lagoon), an IBM pSeries parallel computer, and desktop grids (pools) built with the Condor system in Cambridge and University College London. Users access the compute resources through the Globus and Condor/Condor-G client tools, with security handled through the use of X.509 digital certificates. Users are prohibited from accessing any resources via traditional login procedures such as `ssh`; instead, the Storage Resource Broker (SRB) provides a single access point.

ing built the integration of the clusters with the SRB into the design specifications; integration of the compute and data components is critical to the operation of the clusters within our grid environment.

Cluster Structure and Operation

Figure 1 illustrates the current structure of the *e*Minerals mini-grid and all the other compute resources (including desktop grids built using the Condor system[5]) available to project members. The three clusters are located in Cambridge (Lake@Cambridge), University College London (Lake@UCL), and Bath (Lake@Bath). Routine day-to-day maintenance is performed remotely from Cambridge, with extra assistance provided by project members local to each cluster.

Basic Installation

Each of the three clusters is assembled from standard commodity components. They consist of 16 compute nodes, each with a single Intel Pentium IV processor, 2 Gbytes of RAM, and a 120-Gbyte system hard drive. The nodes are connected via Gigabit Ethernet and controlled by a separate master node of nearly identical specifications. The master node also contains a RAID array to provide 720 Gbytes of data storage, which we can expand if needed. This storage is used only for users' files, not for operating system files.

Few of the codes that users run on the *e*Minerals clusters require constant internode communication—often, they need just bookkeeping communications at the start and end of runs. Because the main requirement is speed and bandwidth for bulk data transfer rather than ultra-low latency, Gigabit Ethernet provides appropriate internode network communications. We minimized the slight latency penalty compared to 100-Mbit Ethernet through careful optimization of the Linux kernel TCP/IP configuration. We achieved an average latency of approximately 100 μs with 64-byte packets, which we deemed to be acceptable. The key design spec-

```
if (defined($description->mpi_type()))
{
  if ($description->mpi_type eq "lam-intel")
  {
    $mpihome = "/opt/lam-intel";
    $ENV{LAMHOME}=$mpihome;
    $ENV{LAMRSH}="ssh -x";
  }
  ... other cases...
}
```

Figure 2. Modified section of the Portable Batch System file (pbs.pm) to enable use of the LAM message-passing interface (MPI). This modification is placed near the beginning of the submit subroutine to let users invoke their preferred compiler/MPI implementations.

ification is that the clusters are optimized for high-throughput computations rather than to be run as mini high-performance machines; *e*Minerals users have access to other resources for their high-performance requirements.

The clusters currently run Mandrake Linux 9.2 (www.mandrakesoft.com) and Oscar 2.3.1 (http://oscar.sourceforge.net). Oscar provides all the necessary cluster-installation tools as well as tools for all basic operations such as job scheduling and monitoring via the Portable Batch System (PBS; www.openpbs.org) and Maui job scheduler (www.clusterresources.com/products/maui).

Most of the simulation scientists who use the *e*Minerals mini-grid use codes written in Fortran. We provided the clusters with several Fortran compilers and message-passing interface (MPI) libraries (both MPICH and LAM implementations) because our users have found that code performance is affected by compiler choice and MPI implementation. We also installed various math libraries, including ScaLapac and BLACS, each of which we installed with standard configurations.

Middleware

Running and managing computations within a grid environment is quite unlike running jobs on stand-alone clusters. Oscar is designed primarily for use in stand-alone clusters, which users access through ssh or some other direct-connection mechanism. However, because our testbed project's focus is on experimentation with running production-level science projects within a grid environment, we allow access to mini-grid resources only through middleware tools rather than direct login. The one exception to this rule is that we allow code developers direct access to the Cambridge cluster via gsissh, rather than ssh, to compile and test their applications.

We use the Globus toolkit[6] (mostly employing the v2 toolkit's functionality, which has been propagated into subsequent versions) to provide access to the clusters. Globus uses X.509 certificate-driven authentication without requiring users to directly log in to submit jobs from their desktop computers using the Globus client tools. Because the middleware tools can be difficult for users to install, we've also installed Globus on a set of designated submit machines, to which users can log in to access the mini-grid. The use of the SRB ensures that users aren't restricted to these machines to access their data. We've created user accounts on each cluster, but the users don't know (and are unable to change) the passwords associated with their accounts on the cluster machines; indeed, they don't need to know them because Globus provides certificate authentication. Once jobs are submitted to a cluster, Globus passes them on to the cluster's internal scheduling system. In the *e*Minerals clusters, PBS provides this scheduling for internal job submission, and Maui handles job scheduling until required resources are available.

We've found that the direct use of Globus tools to access the *e*Minerals mini-grid places too heavy a burden on the simulation scientists. Although willing to work with scripts for job submission, and capable of editing scripts based on standard "recipes," our users have said they don't like editing scripts that contain Globus commands. Yet, experience shows that *e*Minerals users are relatively willing to work with scripts for submission to the *e*Minerals Condor pools; we've therefore written a Perl program called `my_condor_submit` to let users write small job-submission scripts using Condor-G, the Condor interface to Globus.[7]

We've found that the use of multiple MPI libraries leads to difficulties within the Globus implementation. As a result, we had to extend the standard Globus installation to let users specify which MPI installation to use at runtime. By default, the Globus PBS interface uses an MPI implementation, as specified by an environment variable on the cluster in use. The out-of-the-box implementation of the PBS job manager (usually located at `$GLOBUS_HOME/lib/perl/Globus/GRAM/JobManager/pbs.pm`) lacks the technology to perform this choice. On installation, it picks up only one PBS instance, so we extended the Perl implementation of the interface between the Globus and PBS job specifications to handle an extra value in the Globus job's Resource Specification

```
if($description->jobtype() eq "mpi")
{
  if ($mpihome =~ "mpich")
  {
    $pbs_job_script->print("$mpirun -np ". $description->count(). "");
    if($cluster)
    {
      $pbs_job_script->print(" -machinefile \$PBS_NODEFILE ");
    }
    $pbs_job_script->print($description->executable()
      . " $args < "
      . $description->stdin(). "\n");
  }
  elsif ($mpihome =~ "lam")
  {
    $pbs_job_script->print("$mpihome/bin/lamboot \$PBSNODEFILE\n");
    $pbs_job_script->print("$mpirun C ". $description->executable().
      " $args < ". $description->stdin(). "\n");
    $pbs_job_script->print("$mpihome/bin/lamhalt\n");
  }
}
... rest of code...
```

Figure 3. Modified section of `pbs.pm` file. This file allows support for selection between multiple installed message-passing interface (MPI) implementations. The first line defines the point of modification.

Language (RSL), specifying which MPI implementation to use. Globus then passes this extra RSL tag to the PBS scheduler (Maui), which then uses the specified MPI implementation.

To add the required logic to the Globus toolkit, we added a new attribute to the file `$GLOBUS_HOME/share/globus_gram_job_manager/pbs.rvf` to define a new variable that the Globus RSL could understand. To do so, we added the following lines to the file:

```
ValidWhen: GLOBUS_GRAM_JOB_SUBMIT
Attribute: mpi_type
Description:
"Choose the MPI flavour to be used."
ValidWhen: GLOBUS_GRAM_JOB_SUBMIT
```

We implemented the necessary extensions to the `pbs.pm` file by adding a simple switch near the beginning of the `submit` subroutine (Figure 2).

We also changed the working directory to the one requested by the user, and then called `mpirun` with the correct syntax for MPI jobs, depending on whether the job was a LAM or MPICH job (LAM jobs need `lamboot` to be called prior to `mpirun`). This involved adding the lines of code shown in Figure 3 to the `pbs.pm` file at the position defined by the first line.

These simple extensions let users invoke their preferred compiler/MPI implementations, assuming that the system administrator has prebuilt and installed the relevant combination on the cluster.

Data Management

Given that users can't directly log in to the minigrid clusters, their actions with output files are restricted. If users know the names and locations of the files being produced, they can use the `gridftp` tool provided with the Globus toolkit, but users sometimes don't know these exact details. To meet most users' needs, we employed a distributed data-management infrastructure based on the SRB.[3,4] (Note that our objective is to run a distributed data file system, which is distinct from a distributed database system.)

The SRB provides a single logical file structure and a single access point for users, even though data are distributed over several locations. Every file's geographical location is reduced to a file attribute. The SRB has a central metadata catalogue (MCAT) server that maintains information about

```
Universe = globus
Globusscheduler = <mini-grid resource>/jobmanager-pbs
Executable = <name of executable binary or script>
Notification = NEVER
#
# Next line is example RSL for a single-processor PBS job
# Modifications are required for other job managers
#
GlobusRSL = (arguments=none)(job_type=single)(stdin=<filename>)
#
# Next lines are modifications for access to the SRB
#
Sdir = <some directory in the SRB>
Sget = <list of input file names, or * for wildcard>
Sput = <list of output file names, or * for wildcard>
#
Output = <standard output file name>
transferoutput = False
Log = <name of log file>
Error = <name of standard error file>
Queue
```

Figure 4. Template input file for `my_condor_submit`. The extensions provided support SRB access as part of the job life cycle. Users simply change the values in "`<...>`" to relate to their own job requirements.

all files within the SRB. In the *e*Minerals project, the MCAT server is located at the Daresbury Laboratory (see Figure 1) and is mirrored to the Rutherford Appleton Laboratory to provide additional redundancy. In addition to the central MCAT server, we've set up one storage vault on each of the three clusters (the 720-Gbyte RAID arrays mentioned earlier). Moreover, we've installed SRB client tools on each cluster to provide full SRB access for job workflows. This serves a different purpose from the Globus Replica Location Service, which is much more oriented toward replicating data to facilitate data access (see http://www-unix.globus.org/toolkit/docs/3.2/rls).

A user job's compute/data workflow follows a logical sequence:

1. The user places all input files into the SRB.
2. The compute job downloads the relevant files from the SRB to the *e*Minerals mini-grid clusters.
3. The compute job executes, generating several output files.
4. The job's workflow places the output files into the SRB.

This workflow is implemented in the `my_condor_submit` tool, discussed later, which uses Condor's DAGman workflow tool.

One advantage to this approach is that the job life-cycle process generates an archive of the entire process, which is maintained on the SRB. This is particularly useful for collaborative work. Users have three ways to access the SRB:

- the InQ GUI tool for Microsoft Windows (www.npaci.edu/dice/srb/inQ/inQ.html),
- the MySRB Web interface (www.sdsc.edu/srb/mySRB/mySRB.html),[4] or
- the Scommands Unix command-line tools (www.sdsc.edu/srb/scommands).

The latter are invoked within the `my_condor_submit` tool.

The *e*Minerals project is not alone in using an SRB within a grid environment, but our application differs in several respects from others—most notably because the *e*Minerals mini-grid is somewhat smaller than other grids.[9,11] Our experience illustrates the feasibility of setting up a mid-sized integrated compute and data grid infrastructure for small-scale collaborations, accounting for the modest system-management resources that such a collaboration can reasonably provide.

User Access

Given that users must employ both the Globus and SRB tools to manage job submission and data handling in lieu of login access to the *e*Minerals clusters, the main challenge in making this infrastructure usable concerns usability issues, particularly because working with the Globus tools isn't straightforward. To that end, we developed the `my_condor_submit` tool so that users can run jobs within the *e*Minerals mini-grid by modifying a small script like the one in Figure 4.

The user provides the parameters in the angle brackets, `<...>`. The assumptions here are that the application executable is passed from the desktop machine to the cluster, and that all input/output files are held within one directory, called a *collection*, on the SRB. We have variations of the script that allow access to multiple collections on the SRB, which is particularly useful when an application executable is already held in a directory on the SRB rather than transferred from the desktop. In such cases, a copy of the application executable is unlikely to be held with the data files.

System Monitoring

To automatically monitor the large number of compute resources that comprise the *e*Minerals mini-grid, we developed a single tool with two distinct sections. The first is the actual monitoring system, which periodically queries all compute resources for their queue and disk-usage statistics by submitting Globus jobs to the clusters and other machines within the mini-grid (as shown in Figure 1). These jobs run queue-querying and disk-usage commands and return the output to the querying machine, which then parses that output and stores all useful information within a database. This process runs every 30 minutes, so that users have recent, if not completely current, information.

The second section of the tool displays the data collected from the first, enabling *e*Minerals project members to quickly check the status of any machine on the mini-grid and ensure that they can target their jobs to unused resources. On accessing this tool, the user automatically launches a database query; the display system then presents an image similar to that shown in Figure 5 in the Web browser (www.eminerals.org/gridStatus).

System Maintenance and Management

Although the *e*Minerals clusters are reliable and don't require much outside intervention, compute nodes occasionally stop responding to connection requests from their master nodes. In response,

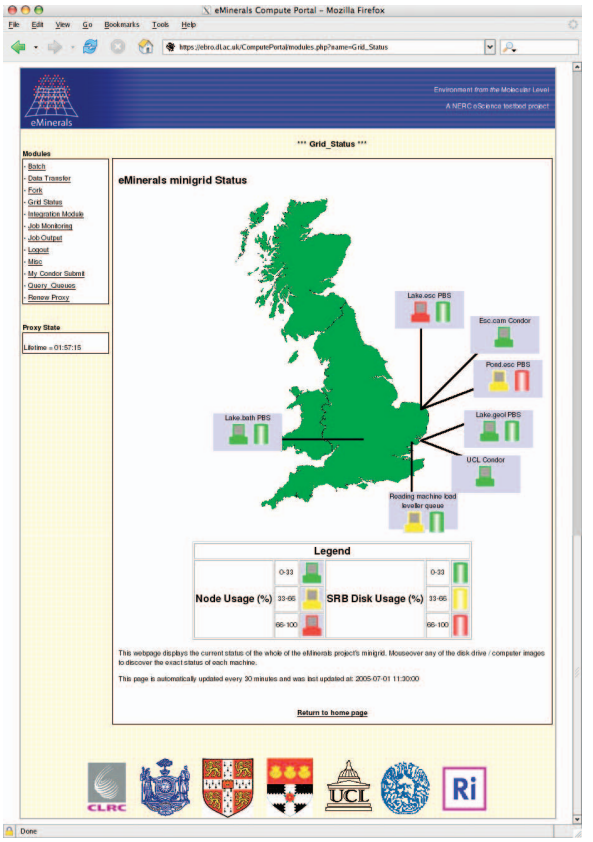

Figure 5. Status of the *e*Minerals mini-grid as shown in the system-monitoring tool. The blue boxes represent compute resources on the mini-grid. The computer image's color gives a simple representation of how heavily the machine is being used. The display of storage cylinders represents the existence of SRB storage vaults; again, the color shows the vault's approximate degree of utilization.

project members at the cluster's location reboot the node. If this fails to restore the node's communications, an administrator or local project member acting under remote supervision from an administrator performs a clean network reinstall from system images stored on the master node. The install images are constantly maintained, so a reinstall results in a fully functioning installation; no further steps are required to configure the node.

In most cases, these steps will reset the misbehaving node to a state in which it can again accept jobs from users and continue as normal. In cases of hardware failure, a remote administrator can perform troubleshooting remotely with assistance from local project members; only rarely will the local members need more than average system experience.

We've also provided a redundant system disk in the master node so that even project members without computing training can swap the system over to

the backup configuration in a matter of minutes. This returns the system to an operational state, allowing other project members to connect to the system and perform troubleshooting tasks from remote locations, independent of the local project members. To ensure that the redundant system disk would remain in a stable state and avoid propagating any system corruption, we didn't implement the use of a redundant system disk using a RAID system.

*e*Minerals project members have written the backup system employed on the clusters specifically for the project, making unique use of otherwise unutilized hard-drive space on the compute nodes. Each night, the backup system makes an archive of every file that each user has changed since the previous day. Once a week, the system archives every file in each cluster's home space and compresses them into one or more files for each user, according to the amount of data to be backed up.

Each compressed archive is stored on the relatively unused hard drive of one of the cluster nodes. Each week, the node to which the backup is stored alternates in a round-robin fashion. In this way, we can keep a complete backup of all users' files for up to 16 weeks before we start to overwrite previous backups (depending on the total size of each week's backup). The backup system then copies all the archives containing files that users have changed during the current week over to the compute node on which the previous week's complete backup is stored. The node to which any further backups are to be made is then incremented, such that the next compute node will store the next week's backup files at the time the backup is performed. Thus, we can combine and manipulate the files backed up to any node to produce any of the files stored on the next node in the backup sequence.

Our current backup system has two potential drawbacks. One is increased network communications because we use the same channel for internode communications and for moving backed-up data. However, this has not been a problem with our clusters. Our approach also provides no protection from physical risks such as fire or theft, but we believe these risks to be sufficiently low in our situation that we have chosen not to address it in our clusters. In some cases, additional security is provided by placing the master node in a separate room.

The *e*Minerals clusters are now in use in production mode for calculations on iron sulphides associated with acid mine leaching,[12] studies of radiation damage in nuclear encapsulation materials, surface-pollutant interactions, crystal growth, and cation ordering.[13] Our project shows that it is practical for consortia to set up and manage clusters within integrated compute and data grid environments without large investments in support or user training. Running three clusters over three sites hasn't required dedicated staff effort, and it has been possible to use untrained team members to help perform occasional troubleshooting tasks.

As an alternative to the command-line interface for access to *e*Minerals resources, we have recently developed a Web-based portal interface.[14] It gives members access to all mini-grid resources for their projects, including job submission, monitoring, and data retrieval. Other members of the *e*Minerals project are also developing a Web services-based interface to Condor,[15] which will give similar access to mini-grid resources.

Both will replicate the functionality contained within `my_condor_submit`, but they will bring two main advantages: avoiding the need for users to install client tools on their desktops, and reducing the problems experienced with communicating through firewalls between project sites. Although we can generally address firewall problems through coordination with the relevant administrators, other problems tend to arise because firewall policies are outside our control and can change without notice, sometimes rendering resources inaccessible until the implications of policy changes can be reversed. Although the new portal and Web services tools still require certain firewall ports to be open, the requirements are much less severe and therefore more tolerable.

We're also planning to upgrade the monitoring tool to automatically notify system administrators via email when any compute nodes stop responding to queries. The new version will also provide more queue statistics, including actual queue status command outputs to enable users to retrieve more information regarding the status of job queues.

Acknowledgments

We are grateful to the Natural Environment Research Council (UK) for funding the e*Minerals project, and to the Centre for Ecology and Hydrology (Monks Wood, Cambridgeshire, UK) for additional financial support for Richard Bruin.*

References

1. M.T. Dove and N.H. de Leeuw, "Grid Computing and Molecular Simulations: The Vision of the *e*Minerals Project," *Molecular Simulations*, vol. 31, no. 5, 2005, pp. 297–301.
2. M. Calleja et al., "Collaborative Grid Infrastructure for Molecular Simulations: The *e*Minerals Mini-Grid as a Prototype Integrated Compute and Data Grid," *Molecular Simulations*, vol. 31, no. 5, 2005, pp. 303–313.

3. D. Thain, T. Tannenbaum, and M. Livny, "Condor and the Grid," *Grid Computing: Making The Global Infrastructure a Reality*, F. Berman, A.J.G. Hey, and G. Fox G, eds., John Wiley & Sons, 2003, pp. 299–335.
4. I. Foster and C. Kesselman, "Globus: A Metacomputing Infrastructure Toolkit," *Int'l J. Supercomputer Applications*, vol. 11, no. 2, 1997, pp. 115–128.
5. J. Frey et al., "Condor-G: A Computation Management Agent for Multi-Institutional Grids," *J. Cluster Computing*, vol. 5, no. 3, 2002, pp. 237–246.
6. R.W. Moore and C. Baru, "Virtualization Services for Data Grids," *Grid Computing: Making the Global Infrastructure a Reality*, F. Berman, A.J.G. Hey, and G. Fox, eds., John Wiley & Sons, 2003, pp. 409–435.
7. A. Rajasekar, M. Wan, and R. Moore, "MySRB & SRB Components of a Data Grid," *Proc. 11th IEEE Int'l Symp. High-Performance Distributed Computing*, IEEE CS Press, 2002, pp. 301–310.
8. K. Nozaki et al., "The First Grid for the Oral and Maxillofacial Region and Its Application for Speech Analysis," *Methods of Information in Medicine*, vol. 44, no. 2, 2005, pp. 253–256.
9. W. Kroeger et al., "BaBar Data Distribution Using the Storage Resource Broker," *IEEE Trans. Nuclear Science*, vol. 51, no. 4, 2004, pp. 1462–1464.
10. M.E. Martone et al., "A Cell-Centered Database for Electron Tomographic Data," *J. Structural Biology*, vol. 138, nos. 1–2, 2002, pp. 145–155.
11. P. Andrews et al., "Scaling a Global File System to the Greatest Possible Extent, Performance, Capacity, and Number of Users," *Proc. 22nd IEEE/13th NASA Goddard Conf. Mass Storage Systems and Technologies*, IEEE CS Press, 2005, pp. 109–117.
12. S. Wells et al., "Ab Initio Simulations of Magnetic Iron Sulphides," *Molecular Simulations*, vol. 31, no. 5, 2005, pp. 379–384.
13. M. Alfredsson et al., "*e*Minerals: Science Outcomes enabled by new Grid Tools," *Proc. UK eScience All Hands Meeting*, EPSRC, 2005, pp. 788–795; www.allhands.org.uk/2005/proceedings/.
14. R. Tyer et al., "Portal Framework for Computation within the *e*Minerals Project," *Proc. UK eScience All Hands Meeting*, EPSRC, 2004, pp. 660–665; www.allhands.org.uk/2004/proceedings/.
15. C. Chapman et al., "Condor Birdbath: Web Service Interfaces to Condor," *Proc. UK eScience All Hands Meeting*, EPSRC, 2005, pp. 737–744; www.allhands.org.uk/2005/proceedings/.

Richard Bruin is a postgraduate student in the Department of Earth Sciences at the University of Cambridge. His technical interests include XML technologies, Semantic Web, intelligent/adaptive job submission, and grid technologies. He has a BSc in computer software theory from the University of Bath. Contact him at rbru03@esc.cam.ac.uk or www.eminerals.org/people/richard_bruin.html.

Martin T. Dove is Professor of Computational Mineral Physics in the Department of Earth Sciences at the University of Cambridge and director of the UK's National Institute for Environmental eScience. His recent research is in applying a range of computational simulation methods to explore the behavior of materials at a molecular level. He has a PhD in condensed matter physics from the University of Birmingham. Contact him at martin@esc.cam.ac.uk or www.esc.cam.ac.uk/astaff/dove.

Mark Calleja works in the Cambridge eScience Centre, but was a postdoctoral worker in the Department of Earth Sciences at the University of Cambridge when this work was undertaken. His technical interests include making grids usable, especially with respect to integrating grid middleware, to make them transparent and intuitive for the user. He has a PhD in physics from the University of Kent. Contact him at M.Calleja@damtp.cam.ac.uk or www.escience.cam.ac.uk/~mcal00.

Matt G. Tucker is currently an instrument scientist at the ISIS Neutron Scattering Facility in Oxfordshire, UK. He held a postdoctoral position in the Department of Earth Sciences at the University of Cambridge when this work was undertaken. He has a PhD in condensed matter physics from the University of Kent. Contact him at matt.tucker@physics.org.