

# Job submission to grid computing environments

RP Bruin, TOH White, AM Walker, KF Austen, MT Dove

*Department of Earth Sciences, University of Cambridge, Downing Street, Cambridge CB2 3EQ*

RP Tyer, PA Couch, IT Todorov

*CCLRC, Daresbury Laboratory, Warrington, Cheshire WA4 4AD*

MO Blanchard

*Royal Institution, 21 Albemarle Street, London W1S 4BS*

## Abstract

The problem of enabling scientist users to submit jobs to grid computing environments will eventually limit the usability of grids. The *eMinerals* project has tackled this problem by developing the “my\_Condor\_submit” (MCS) tool, which provides a simple shell-script interface to Globus, a flexible interaction with the Storage Resource Broker, metascheduling with load balancing within a grid environment, and automatic metadata harvesting. This paper provides an overview of MCS together with some use cases. We also describe the use of MCS within parameter-sweep studies.

## Introduction

For grid computing infrastructures to be exploited, it is essential that the tools built to provide access have usability designed into them from the outset. In our experience, it is unrealistic to ask most scientists to work with raw Globus job-submission commands – in the end they are likely to end up compromising by merely using `gsissh` to log into grid resources and submit jobs using more familiar batch queue commands. However, we have found that asking them to work with Condor job submission scripts is quite feasible [1]. In this paper we describe work we have done to develop Condor-like grid job submission tools that encompass integration with the San Diego Storage Resource Broker (SRB) [2] and new metadata capture tools [3].

The context for this work is the *eMinerals* minigrid structure [4,5]. This is a heterogeneous compute grid integrated with a data grid based on the SRB, as illustrated in Figure 1. Since the *eMinerals* project is an *escience* testbed project, we enforce a policy that access is controlled by the use of Globus job submission scripts (with authentication handled by Globus Grid Security Infrastructure (GSI) and X.509 digital certificates); we do not enable access via `gsissh` except for certain mission-critical exceptions (mostly for code developers and system administrators).

In previous papers we have described the *eMinerals* minigrid structure and access tools in some detail [4,5]. Our main job submission tool is “my\_Condor\_submit” (MCS). This uses Condor-G, a Condor wrapping of Globus job-submission tools. As noted above, we quickly learned within the *eMinerals* project that users can work quite easily with Condor job submission scripts, and working with MCS only requires users to prepare a short Condor-like job submission script [1].

The first versions of MCS [4,5] essentially carried out one small set of tasks, namely to retrieve data from the SRB, run a program on the *eMinerals* minigrid, and then upload generated files to the SRB. Subsequent to this, we have developed a new version of MCS with some innovations that will be described in this paper. These include metascheduling, generalisation to other grid infrastructures (including campus grids and the National Grid Service), and automatic metadata capture.

MCS is primarily designed for users to submit one job at a time to a grid infrastructure from a computer on which the Globus toolkit has been installed. An issue that requires us to go beyond this approach is the need to wrap MCS within an infrastructure that enables the scientist to submit many jobs at a single instance as part of a combinatorial or ensemble study. We have developed scripts enabling automatic generation of input files for parameter

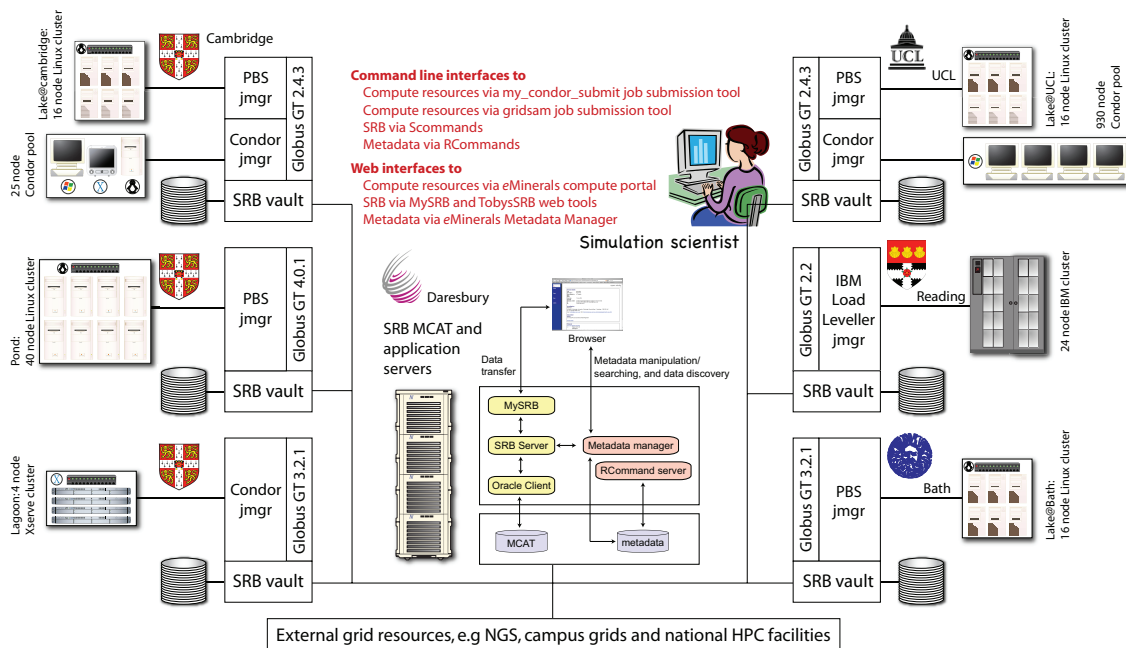


Figure 1. The eMinerals minigrid, showing the range of computational resources and the integrated data management infrastructure based on the use of the Storage Resource Broker.

sweeps which use MCS for their submission to address this need.

This paper describes how the eMinerals project handles grid computing job submission in a way that hides the underlying middleware from the users, and makes grid computing genuinely usable for scientists.

## The eMinerals minigrid 2006

The current status of the eMinerals minigrid is illustrated in Figure 1. At the heart are two sets of components: a heterogeneous collection of compute resources, including clusters and Condor pools, and the data architecture based on the SRB with a central metadata catalogue installed onto a central application server, with distributed data vaults. Subsequent to our previous descriptions of the minigrid [4,5] are the following developments:

- ▶ Links to campus grids, e.g. CamGrid [6]. CamGrid is a heterogeneous Condor pool made up of several distributed and individually-maintained Condor pools which are integrated into a single grid through the use of Condor flocking. These resources are linked to the eMinerals minigrid by installing a Globus gatekeeper with a Condor jobmanager to allow job submission in the same manner as for other resources;
- ▶ Links to other grid resources such as the National Grid Service (NGS) and NW-grid [7] with seamless access, provided they have

Globus gatekeepers and the possibility to install SRB and metadata tools;

- ▶ Creation of a metadata database and tools [3]. These enable automated metadata capture and storage so that simulation data can be archived in a manner that allows for much simpler retrieval at a later date. It also greatly facilitates collaborative working within a distributed virtual organisation such as eMinerals [8].

## my\_Conдор\_submit 2006

MCS consists of a fairly complicated perl program and a central database, and is used with a simple input file with slight extensions over a standard Condor submission script. The primary purpose of MCS is to submit jobs to a grid infrastructure with data archiving and management handled by the SRB. The use of the SRB in our context serves two purposes. First, it is a convenient way to enable transfer of data between the user and a grid infrastructure, bypassing some of the problems associated with retrieval of multiple files whose names may not be known beforehand using Condor and gridftp methods for example. Second, the use of the SRB enables users to archive all files associated with a study in a way that facilitates good data management and enables collaborative access.

We have recently completely rewritten MCS. The original version [4,5] was written as a quick development to jumpstart access to the

eMinerals minigrid for the project scientists. It turned out that MCS was more successful than anticipated, mainly because it matched users' requirements and way of working. Because the original version was written quickly, it was increasingly difficult to upgrade and integrate new developments. Thus a complete rewrite proved to be essential to make future developments easy to implement.

The current version of MCS (version 1.2 at July 2006) has the following features that extend versions previously described:

1. access to multiple collections (directories) on the SRB, since typically executables, data files and standard input files (such as pseudopotential files for quantum mechanical calculations) will be stored in different collections;
2. generalised to support job submission to external grid infrastructures, including campus grids and the NGS;
3. metascheduling with load balancing across all minigrid and external resources;
4. new metadata and XML tools to automatically obtain metadata from the output files and job submission parameters.

In the rest of this paper we will describe some of these in more detail.

## MCS and metadata

We have been systematically enabling the simulation codes used within the eMinerals project to write output and configuration files in XML. Specifically, we use the Chemical Markup Language (CML) [9]. One author (TOHW) has developed a Fortran library, 'FoX', to facilitate writing CML files [9]. Another of the authors (PAC) has developed a library, called 'AgentX', to facilitate general reading of XML files into Fortran using logical mappings rather than a concrete document structure called AgentX [10].

Our output XML files broadly have the data separated into three components; *metadata*, *parameter* and *property*. The *metadata* component consists of general information about the job, such as program name, version number etc. The *parameter* components are mainly reflections of the input parameters that control, and subsequently identify, the particular simulation. Examples range from the interesting parameters such as temperature and pressure to the more mundane but nevertheless important control parameters such as various cut-offs. The *property* components are the output data from the simulation. These lists are vast, and include step-by-step data as well as final averages or

final values. Typically for metadata collection we need to retrieve some data from each of these three components. It is worth remarking with regard to the property metadata that we blur the distinction between data and metadata. This is illustrated by a simple example. In a study of the binding energy of a family of molecules (such as the dioxin family, where members of the family differ in the number and location of chlorine or hydrogen atoms), we store the data on final energy as metadata. This allows us to use this energy for our metadata search tools; an example would be to search through a study of all dioxin molecules for the molecule with the lowest binding energy.

In MCS, metadata is extracted from the XML data documents using the AgentX library. AgentX implements a simple API that can be used by other applications to find data represented in a document according to their context. The context is specified through a series of queries based on terms specified in an ontology. These terms relate to classes of entities of interest (for example 'Atom', 'Crystal' and 'Molecule') and their properties (for example 'zCoordinate'). This approach requires a set of mappings relating these terms to fragment identifiers. It is these identifiers that are evaluated to locate parts of documents representing data with a well defined context. In this way, AgentX hides the details of a particular data format from its users, so that the complexities of dealing with XML and the necessity of understanding the precise details of a particular data model are removed. The user is not required to understand the details of the ontology or mappings, but must have an understanding of the terms used.

Once data have been located using AgentX, MCS associates each data item with a term (such as 'FinalEnergy') which is used to provide the context of the data. The data and term association are then stored in the project metadata database, making use of the RCommands [3].

We provide an example of extracting the final energy from a quantum mechanical energy relaxation using the SIESTA code [11]. In this work we use final energy as a metadata item because it is a useful quantity for data organisation and for search tools. The call to AgentX in MCS follows an XPath-like syntax:

```
AgentX = finalEnergy,  
chlorobenzene.XML:/Module  
[last]/PropertyList[title =  
'Final Energy']/Property  
[dictRef = 'siesta:Etot']
```

This directive specifies that MCS is to extract the metadata as a value from the file called 'chlorobenzene.XML', and it will associate this value with the string called 'finalEnergy'. The AgentX call looks for this value within the last module container, which by convention holds properties representing the final state of the system, then within a propertyList called 'Final Energy', and finally within a property value defined by the dictionary reference 'siesta:Etot'.

In addition to metadata harvested from output XML documents, we also collect metadata related to the user's submission environment. Examples include the date of submission, name of the machine submitted from, and the user's username on the submission machine. Metadata can also be collected about the job's execution environment, including the name of the machine on which the simulation was run, the completion date of the run, and the user's username on that machine. Users are also able to store arbitrary strings of metadata using a simple MCS command. All these types of metadata provide useful adjuncts to the scientific metadata harvested from the simulation output XML files.

## MCS and metascheduling

One problem with the earlier versions of MCS was that the user had to specify which compute resource any simulation should be submitted to. This resulted in many jobs being submitted to a few busy resources within the eMinerals minigrid whilst other resources were left idle. Because users are not allowed to log in to resources, it was not possible for them to check job queues; in any case, such an approach is not a scalable solution to resource monitoring.

An intermediate solution to this problem was the creation of a status monitoring web page<sup>1</sup>, which graphically shows the status of all available minigrid resources. However, this solution still requires user involvement to look at the page and decide where to run. Moreover, the web page caches the data for up to thirty minutes meaning that it would still be possible for users to submit to resources that have suddenly become busy since the last update.

To enable better load balancing across resources, querying of resource utilisation must be built into the submission mechanism, and must use up to date rather than cached data.

This type of metascheduling, based on a round-robin algorithm, has now been built into MCS. To use this metascheduling facility, the user specifies the type of machine they wish to use, using the keywords 'performance' to submit to clusters, or 'throughput' to submit to Condor pools. MCS then retrieves a list of machines from a central database (which is mirrored for reliability), and queries the machines in turn checking for available processors. The database contains a list of available resources ranked in the order in which they were last submitted to (the most recently used machine appears last in the list) and other machine specific information such as the path to the required queue command and the machine's architecture, etc.

Querying of the resource state is performed by sending a Globus fork job to the resource that executes the relevant local queue command (obtained from the central database). Querying the status of a PBS cluster will result in the use of the 'pbsnodes' command with suitable output parsing. Querying one of the available Condor pools will use a command written specifically for this purpose, wrapping the standard 'Condor\_status -totals'. The need for a purpose-written command is that this request will poll flocked Condor pools within a campus grid, and the standard queue queries do not return the required information in this case.

If all the resources are busy, MCS will inform the user and queue the jobs using information in the database to ensure even balance in the various resource queues.

As part of the metascheduling MCS takes account of the fact that not all codes will run on all resources. Again, this information is extracted from the database. Executables for various platforms are held within the SRB, and MCS will automatically select the appropriate SRB file download.

## Example MCS files

Figures 2 and 3 show two examples of MCS input files. For those familiar with Condor, the Condor-G wrapping roots of MCS can be seen in the structure of the file. Figure 2 is the simplest example. The first three lines give information about the executable (name and location within the SRB) and the standard GlobusRSL command. The three lines with names beginning with S provide the interaction with the SRB. The Sdir line passes the name

---

<sup>1</sup> <http://www.eminerals.org/gridstatus>

```

# Specify the name of the executable to run
Executable      = gulp

# Specify where the executable should get stdin from and put stdout to
GlobusRSL = (stdin=andalusite.dat)(stdout=andalusite.out)

# Specify an SRB collection to get the relevant executable from
pathToExe      = /home/codes.eminerals/gulp/

# Specify a metadata dataset to create all metadata within
RDatasetId     = 55

# Specify a directory to get files from, put files to and relate to
# metadata created below
Sdir           = /home/user01.eminerals/gulpminerals/
Sget           = *
Sput           = *

# Creates and names a metadata data object
Rdesc         = "Gulp output from andalusite at ambient conditions"
# Specify metadata to get from files with Agent-x - get environment
# and default metadata only
AgentXDefault= andalusite.XML
GetEnvMetadata = True

```

*Figure 2: Example of a simple MCS script file.*

of the SRB collection containing the files, and the “Sput \*” and “Sget \*” lines instruct MCS to download and upload all files. The lines beginning with R concern the interaction with the metadata database through the RCommands. The identification number of the relevant metadata dataset into which data objects are to be stored is passed by the RDatasetID parameter. The Rdesc command creates a data object with the specified name. Its associated URL within the SRB will be automatically created by MCS.

Figure 3 shows a more complex example, including the components of the simpler script of Figure 2. This script contains near the top parameters for the metascheduling task, including a list of specified resources to be used (preferredmachineList) and the type of job (jobType). The script in Figure 3 involves creation of a metadata dataset. It also contains commands to use AgentX to obtain metadata from the XML file. In this case, the study concerns an investigation of how the energy of a molecule held over a mineral surface varies with its z coordinate and the repeat distance in the z direction (latticeVectorC).

## Parameter sweep code

Many of the problems tackled within the eMinerals project are combinatorial in nature. Thus a typical study may involve running many

similar simulations concurrently (up to several hundreds), with each simulation corresponding to a different parameter value. Parameter sweep studies of this sort are well suited to the resources available in the eMinerals minigrid.

We have developed a set of script commands to make the task of setting up and running many concurrent jobs within a single study relatively easy. The user supplies a template of the simulation input file and information regarding the parameter values. The script then creates a set of collections in the SRB, one for each set of parameter values, containing the completed input files, and a commensurate set of directories on the user’s local computer containing the generated MCS input files. The actual process of submitting the jobs is performed by running another script command, which then walks through each of the locally stored MCS input files and submits them all completely independently from each other.

Now having the tools for setting up, running, and storing the resultant data files for large numbers of jobs, the scientist then faces the problem of extracting the key information from the resultant deluge of data. Although the required information will differ from one study to another, there are certain commonalities in parameter sweep studies. The task is made easier by our use of XML to represent data. We have written a number of analysis tools for gathering data from many XML files, using AgentX, and generating XHTML files with

```

# Specify the executable to run
Executable      = siesta
# Instruct Condor to not tell us the outcome from the job by email
Notification    = NEVER

# Specify which file to use for stdin and stdout
GlobusRSL = (stdin=chlorobenzene.dat)(stdout=chlorobenzene.out)

# Force overwriting when uploading/downloading files
SForce        = true

# Specify an SRB collection to get the relevant executable from
pathToExe     = /home/codes.eminerals/siesta/
# Specify a list of machines that we are happy to submit to
preferredMachineList = lake.bath.ac.uk lake.esc.cam.ac.uk
lake.geol.ucl.ac.uk pond.esc.cam.ac.uk
# Specify the type of machine to be submitted to;
# "throughput: for a Condor pool and "performance" for a cluster
jobType       = performance
# Specify how many processors to use on the remote machine
numOfProcs   = 1

# Specify a metadata study to create a dataset within
RStudyId     = 1010
# Create and name a metadata dataset to contain data objects
RDatasetName = "chlorobenzene on clay surface"

# Specify an SRB collection to do some transfers to/from
Sdir         = /home/user01.eminerals/clay_surface/
# Specify that we want to get every file from within this collection
Sget        = *

# Specify another SRB collection to do some transfers to/from
Sdir        = /home/user01.eminerals/chlorobenzene
# Specify that we want to put all local files into the specified collection
Sput        = *

# Create and names a metadata data object
Rdesc       = "chlorobenzene molecule on clay surface: first test"
# Specify metadata to get with Agent-x (Tied to the previous Sdir line)
# Get environment metadata
GetEnvMetadata = true
# Get default metadata from the specified file
AgentXDefault = pcbprimfixed.XML
# Get z coordinate information and store as zCoordinate in the metadata
# database
AgentX      = zCoordinate, pcbprimfixed.XML:/molecule[1]/atom[last]/
zCoordinate
# Get lattice vector information and store in the metadata database
AgentX      = latticeVectorA, pcbprimfixed.XML:/Module/LatticeVector[1]
AgentX      = latticeVectorB, pcbprimfixed.XML:/Module/LatticeVector[2]
AgentX      = latticeVectorC, pcbprimfixed.XML:/Module/LatticeVector[3]
# Get the final energy from the file and store in the metadata database
AgentX      = finalEnergy, pcbprimfixed.XML:/Module[last]/PropertyList
[title='Final Energy']/Property[dictRef='siesta:Etot']
# Store an arbitrary string of metadata
MetadataString = arbString1, "First test of molecule height & z separation"

# Leave the code's stderr on the remote machine, to be uploaded to the SRB
# at job end
Transfer_Error = false

# End the file (taken from the Condor input file)
queue

```

*Figure 3: Example of a complex MCS input script*

embedded SVG plots of interesting data [8,12].

To summarise, a full workflow can be constructed, starting with the parameter sweep information, creating multiple sets of input files, submitting and executing all the jobs, returning the data, and finally retrieving particular data of interest, and generating a single graph showing results from all jobs.

## Case examples

In this section we describe some of the *e*Minerals science cases for which MCS has proved invaluable. Some, but not all, of these examples use the parameter sweep tools as well. All use the SRB, metascheduling and metadata tools within MCS. The key point from each example is that the process of job submission, including parameter sweep studies, is made a lot easier for the scientists.

### Amorphous silica under pressure

Our work on modelling amorphous silica is described in detail elsewhere [13]. This study is easily broken down into many simulations, each with a different pressure but otherwise identical simulation parameters. The user provides a template input file with all of the necessary input parameters, and uses the parameter sweep tools to configure the scripts to vary pressure between the lower and upper values and in the desired number of steps (for example, values might vary from  $-5$  GPa to  $+5$  GPa in steps of  $0.01$  GPa, producing 100 simulations.). Once these values had been provided to the system then the user must simply run the command to set up the jobs and the data within the SRB, and then to submit the simulations to the available resources. MCS handles the complete job life cycle, including resource scheduling.

On completion of the jobs, the relevant task was to extract from all the resultant output XML files the values of pressure and volume. We also can monitor the performance and quality of the simulations by extracting XML data step by step.

### Quantum mechanical calculations of molecular conformations

MCS was used extensively in the parameterisation of jobs to study the molecular conformations of polychlorinatedbiphenyl (PCB) molecules [14]. Preliminary calculations were required in order to determine the minimum space required to enclose a PCB molecule in vacuum by sampling over different repeat distances. The initial sweep involved a

suite of 121 SIESTA calculations, but it subsequently transpired that more than one set of calculations was required. MCS facilitated an efficient use of the *e*Minerals compute resources, and led to the speedy retrieval of results. MCS also harvested metadata from the XML files, which was searchable by collaborators on the project, facilitating data sharing and results manipulation.

### Pollutant arsenic ions within iron pyrite

In addition to the gains offered by MCS to combinatorial studies, the usefulness of MCS is not restricted to that kind of study; it can be generalised to all studies involving a large number of independent calculations. We highlight here an example treated in more detail elsewhere [15], namely the environmental problem of the incorporation in of arsenic in  $\text{FeS}_2$  pyrite. This has been investigated by considering four incorporation mechanisms in two redox conditions through relatively simple chemical reactions. This leads to eight reactions with four to six components each. Therefore many independent calculations were performed for testing and obtaining the total energy of all reaction components using quantum mechanics codes. For this example, several submission procedures were needed, with at least one for each reaction component.

## Summary and discussion

The problem of accessing grid resources without making either unreasonable demands of the scientist user or forcing users to compromise what they can achieve in their use of grid computing has been solved by the MCS tool described in this paper. MCS has undergone a complete reworking recently, and is now capable of metascheduling and load balancing, flexible interaction with the SRB, automatic metadata harvesting, and interaction with any grid infrastructure with a Globus gatekeeper (including Condor pools as well as clusters).

The only disadvantage faced by users of MCS is the need to have the Globus client tools and Condor installed on their desktop computers. With recent releases of Globus, this is less of a problem for users of Linux and Mac OS X computers, but remains a problem for users of the Windows operating system. A coupled problem may arise from firewalls with policies that restrict the opening of the necessary ports. For users who face either of these problems, our solution is to provide a group of submit machines from which users can submit their jobs using MCS. However, this is

less than ideal, so the next challenge we face is to bring MCS closer to the desktop. We are working on two solutions. One is the wrapping of MCS into a web portal, and the other is to wrap MCS into a web service.

Although MCS has been designed within the context of the eMinerals project, efforts have been made to enable more general usability. Thus, for example, it enables access to other grid resources, such as the NGS and NW-Grid [7], provided that they allow access via a Globus gatekeeper. The key point about MCS is that it provides integration with the SRB, and hence the SRB SCommand client tools need to be installed. To make use of the metadata tools (not a requirement), the RCommands and AgentX tools also need to be installed. These tools can be installed within user filesystems rather than necessarily being in public areas.

More details on MCS, including manual, program download, database details and parameters sweep tools, are available from reference 16.

## Acknowledgements

We are grateful for funding from NERC (grant reference numbers NER/T/S/2001/00855, NE/C515698/1 and NE/C515704/1).

## References

1. MT Dove, TO White, RP Bruin, MG Tucker, M Calleja, E Artacho, P Murray-Rust, RP Tyer, I Todorov, RJ Allan, K Kleese van Dam, W Smith, C Chapman, W Emmerich, A Marmier, SC Parker, GJ Lewis, SM Hasan, A Thandavan, V Alexandrov, M Blanchard, K Wright, CRA Catlow, Z Du, NH de Leeuw, M Alfredsson, GD Price, J Brodholt. eScience usability: the eMinerals experience. *Proceedings of All Hands 2005* (ISBN 1-904425-53-4), pp 30–37
2. RW Moore and C Baru. Virtualization services for data grids. in *Grid Computing: Making The Global Infrastructure a Reality*, (ed Berman F, Hey AJG and Fox G, John Wiley), Chapter 16 (2003)
3. RP Tyer, PA Couch, K Kleese van Dam, IT Todorov, RP Bruin, TOH White, AM Walker, KF Austen, MT Dove, MO Blanchard. Automatic metadata capture and grid computing. *Proceedings of All Hands 2006*
4. M Calleja, L Blanshard, R Bruin, C Chapman, A Thandavan, R Tyer, P Wilson, V Alexandrov, RJ Allen, J Brodholt, MT Dove, W Emmerich, K Kleese van Dam. Grid tool integration within the eMinerals project. *Proceedings of the UK e-Science All Hands Meeting 2004*, (ISBN 1-904425-21-6), pp 812–817
5. M Calleja, R Bruin, MG Tucker, MT Dove, R Tyer, L Blanshard, K Kleese van Dam, RJ Allan, C Chapman, W Emmerich, P Wilson, J Brodholt, A.Thandavan, VN Alexandrov. Collaborative grid infrastructure for molecular simulations: The eMinerals minigrid as a prototype integrated compute and data grid. *Mol. Simul.* **31**, 303–313, 2005
6. M Calleja, B Beckles, M Keegan, MA Hayes, A Parker, MT Dove. CamGrid: Experiences in constructing a university-wide, Condor-based grid at the University of Cambridge. *Proceedings of the UK e-Science All Hands Meeting 2004*, (ISBN 1-904425-21-6), pp 173–178
7. <http://www.nw-grid.ac.uk/>
8. MT Dove, E Artacho, TO White, RP Bruin, MG Tucker, P Murray-Rust, RJ Allan, K Kleese van Dam, W Smith, RP Tyer, I Todorov, W Emmerich, C Chapman, SC Parker, A Marmier, V Alexandrov, GJ Lewis, SM Hasan, A Thandavan, K Wright, CRA Catlow, M Blanchard, NH de Leeuw, Z Du, GD Price, J Brodholt, M Alfredsson. The eMinerals project: developing the concept of the virtual organisation to support collaborative work on molecular-scale environmental simulations. *Proceedings of All Hands 2005* (ISBN 1-904425-53-4), pp 1058–1065
9. TOH White, P Murray-Rust, PA Couch, RP Tyer, RP Bruin, MT Dove, IT Todorov, SC Parker. Development and Use of CML in the eMinerals project. *Proceedings of All Hands 2006*
10. PA Couch, P Sherwood, S Sufi, IT Todorov, RJ Allan, PJ Knowles, RP Bruin, MT Dove, P Murray-Rust. Towards Data Integration for Computational Chemistry. *Proceedings of All Hands 2005* (ISBN 1-904425-53-4), pp 426–432
11. JM Soler, E Artacho, JD Gale, A Garcia, J Junquera, P Ordejón, D Sánchez-Portal. The SIESTA method for *ab initio* order-*N* materials simulation. *J. Phys.: Cond. Matter* **14**, 2745–2779, 2002
12. TOH White, RP Tyer, RP Bruin, MT Dove, KF Austen. A lightweight, scriptable, web-based frontend to the SRB. *Proceedings of All Hands 2006*
13. MT Dove, LA Sullivan, AM Walker, K Trachenko, RP Bruin, TOH White, P Murray-Rust, RP Tyer, PA Couch, IT Todorov, W Smith, K Kleese van Dam. Anatomy of a grid-enabled molecular simulation study: the compressibility of amorphous silica. *Proceedings of All Hands 2006*
14. KF Austen, TOH White, RP Bruin, MT Dove, E Artacho, RP Tyer. Using eScience to calibrate our tools: parameterisation of quantum mechanical calculations with grid technologies. *Proceedings of All Hands 2006*
15. Z Du, VN Alexandrov, M Alfredsson, KF Austen, ND Bennett, MO Blanchard, JP Brodholt, RP Bruin, CRA Catlow, C Chapman, DJ Cooke, TG Cooper, MT Dove, W Emmerich, SM Hasan, S Kerisit, NH de Leeuw, GJ Lewis, A Marmier, SC Parker, GD Price, W Smith, IT Todorov, R. Tyer, AM Walker, TOH White, K Wright. A virtual research organization enabled by eMinerals minigrid: An integrated study of the transport and immobilisation of arsenic species in the environment. *Proceedings of All Hands 2006*
16. <http://www.eminerals.org/tools/mcs.html>