ENACTS: Joint Scientific/Technological Activities and Studies - Demonstrating a European Metacentre: Feasibility Report

Chris Johnson, Jean-Christophe Desplat,
Edinburgh Parallel Computing Centre (EPCC);
Jacko Koster, Jan-Frode Myklebust,
Parallab, BCCS, University of Bergen;
Geoff Bradley,
Trinity College Dublin (TCD)

November 2, 2004

Contents

List of Figures

List of Tables

1 The ENACTS project

1.1 Remits

ENACTS is a Co-operation Network in the ‘Improving Human Potential Access to Research Infrastructures’ Programme.

This Infrastructure Co-operation Network brings together High Performance Computing (HPC) Large Scale Facilities (LSF) funded by the DGXII’s IHP programme and key user groups. The aim is to evaluate future trends in the way that computational science will be performed and the pan-European implications. As part of the Network’s remit, it runs a Round Table to monitor and advise the operation of the four IHP LSFs in this area, EPCC (UK), CESCA-CEPBA (Spain), CINECA (Italy), and BCPL-Parallab (Norway).

This co-operation network follows on from the successful Framework IV Concerted Action (DIRECT: ERBFMECT970094) [DIR] and brings together many of the key players from around Europe who offer a rich diversity of High Performance Computing (HPC) systems and services. In ENACTS, our strategy involves close co-operation at a pan-European level - to review service provision and distil best-practice, to monitor users' changing requirements for value-added services, and to track technological advances. Back in 1999, when the ENACTS programme was set up, the key developments in HPC were in the area of Grid computing and driven by large US programmes. In Europe we needed to evaluate the status and likely impacts of these technologies in order to move us towards our goal of European Grid computing, a 'virtual infrastructure' - where each researcher, regardless of nationality or geographical location, has access to the best resources and can conduct collaborative research with top quality scientific and technological support. One of the main goals of the ENACTS project was to perform these evaluations and assessment in the European environment.

ENACTS provides participants with a co-operative structure within which to review the impact of Grid computing technologies, enabling them to formulate a strategy for increasing the quantity and quality of access provided.

1.2 Scope and Membership

The scope of our network is computational science: the HPC infrastructures which enable it and the researchers, primarily in the physical sciences, which use it.

Table 1: ENACTS participants by role and skills

Centre	Role	Skills/Interests
EPCC	IHP-LSF	Particle physics, materials science
ICCC Ltd	User	Optimisation techniques, control engineering
UNI-C	LSF	Statistical computing, bioinformatics, multimedia
CSC	User	Meteorology, chemistry, physics, bio-sciences
ENS-L	Society	Computational condensed matter physics, chemistry
FORTH	User	Computer science, computational physics, chemistry
TCD	User	Particle physics, pharmaceuticals
CINECA	IHP-LSF	Meteorology, VR
CSCISM	User	Molecular sciences
UiB	IHP-LSF	Computational physics, geophysics, bio-informatics
PSNC	User	Computer science, networking
UPC	IHP-LSF	Meteorology, computer science
NSC	User	Meteorology, CFD, engineering
ETH-Zurich	LSF	Computer science, physics

Three of the participants (EPCC, CINECA and CESCA-CEPBA) are LSFs providing Researchers' Access in HPC under the HCM and TMR programmes. All were successful in bidding to Framework Programme V (FP V) for IHP funding to continue their programmes. In this, they have been joined by the Parallab and the associated Bergen Computational Physics Laboratory (BCPL) and all four LSFs are full partners in this network proposal and plan to co-operative more closely in the Transnational Access programme.

Between them, these LSFs have already provided access to over 500 European researchers in a very wide range of disciplines and are thus well placed to understand the needs of academic and industrial researchers. The other 10 ENACTS members are drawn from a range of European organisations with the aim of including representation from interested user groups and also by centres in economically less favoured regions. Their input will ensure that the Network's strategy is guided by users' needs and relevant to smaller start-up centres and to larger more established facilities.

A list of participants together with their role and skills is given in 1 whilst their geographical distribution is illustrated in Fig. (1).

1.3 Workplan

The principal objective was to enable the formation of a pan-European HPC metacentre. Achieving this goal required both capital investment and a careful study of the software and support implications for users and HPC centres. The latter was the core objective of this study. Bids for the former (*eg* RTD proposals) may also follow from this study.

The project was organised in two phases. A set of six studies of key enabling technologies was undertaken during the first phase:

- Grid service requirements (EPCC, PSNC);
- the roadmap for HPC (NSC, CSCISM);
- Grid enabling technologies (ETH-Zurich, Forth);
- data management and assimilation (CINECA, TCD);
- distance learning and support (UNI-C, ICCC Ltd.); and
- software efficiency and reusabilty (UPC, UiB).

Following on from the results of the Phase I projects, the Network has undertaken a demonstrator of the usefulness and problems of Grid computing and its Europe-wide implementation. Prior to the practical test, the Network undertook a user needs survey and assessment of how centres' current operating procedures would have to change with the advent of Grid Computing. As before, these studies have involved only a small number of participants, but with the results disseminated to all:

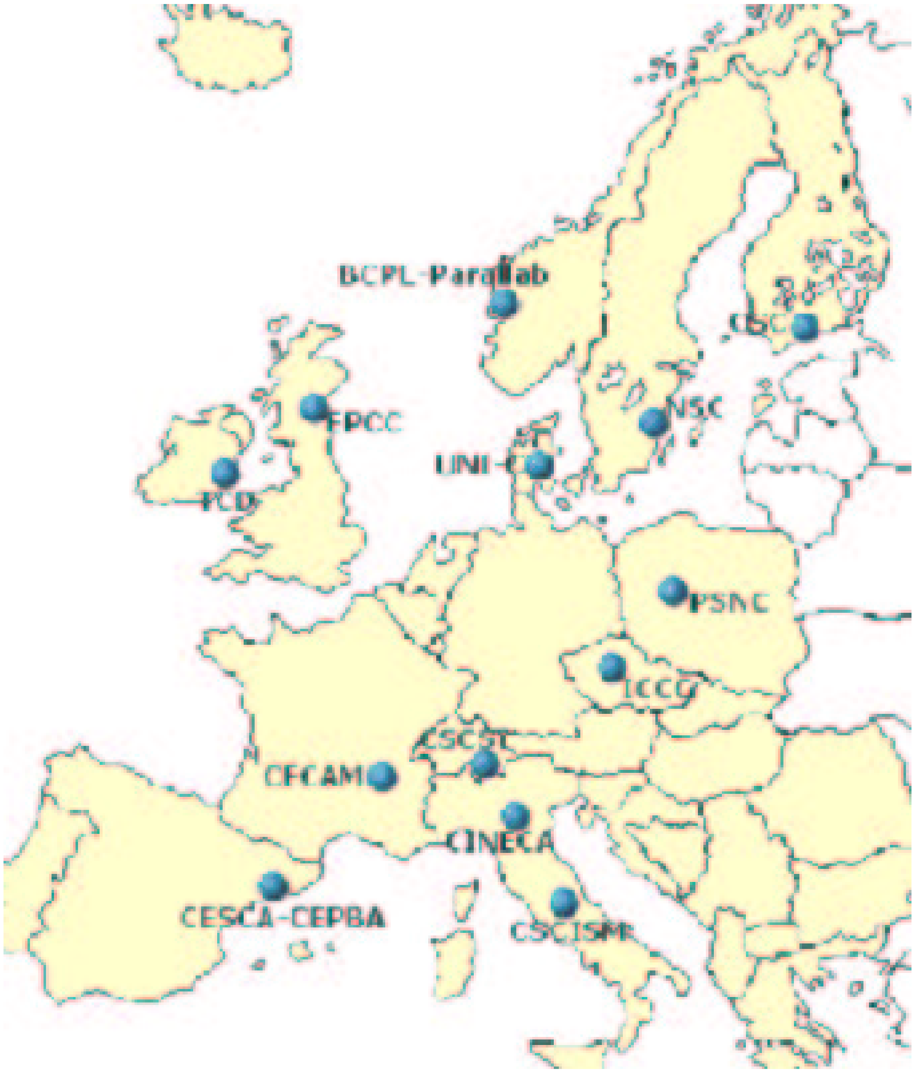


Figure 1: The ENACTS co-operation network

- European metacentre demonstrator: to determine what changes were required to be implemented by HPC centres to align them with a Grid-centric computing environment. This included a demonstrator project to determine the practicality of running production codes in a computational Grid and migrating the data appropriately (EPCC, UiB, TCD);

- user survey: to assess needs and expectations of user groups across Europe (CINECA and CSC);

- dissemination: initiation of a general dissemination activity targeted on creating awareness of the results of the sectoral reports and their implications amongst user groups and HPC centres (Forth, UNI-C, ENS-L). The Network Co-ordinator and the participants will continue their own dissemination activities during Year 2004.

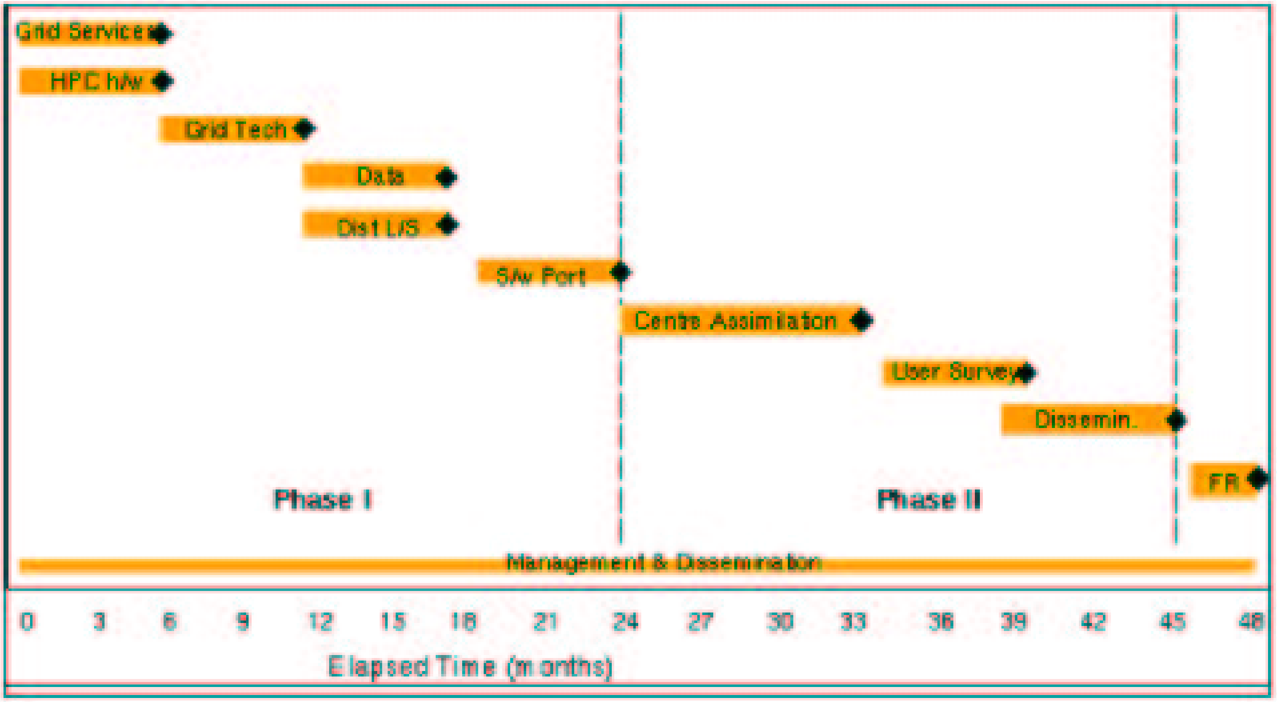


Figure 2: Gantt Chart for the Work of the ENACTS Network

1.4 The partners

1.4.1 EPCC, UK

Edinburgh Parallel Computing Centre (EPCC) was established as a focus for the University of Edinburgh's work in high performance computing. The Centre's task is to accelerate the effective exploitation of high performance parallel computing systems throughout academia, industry and commerce. It houses an exceptional range of computers. EPCC's goal is achieved through a range of activities spanning postgraduate and advanced training programmes, service provision, industrial affiliation and contract work. EPCC offers software design and development for distributed and high-performance systems and offers training courses and materials covering all aspects of HPC.

Alongside their on-going technology transfer work with business, EPCC is also heavily involved in developing software for the Grid. EPCC is a core part of the UK's e-Science programme and a member of the Globus Alliance. EPCC's particular interests lie in building on Grid software standards with platform-neutral distributed technologies like XML and Java in a high performance context to provide middleware solutions with potential commercial application. EPCC is also working in close collaboration with the UK National E-Science Centre (NeSC) to promote Grid technology within the UK scientific community [NES].

See http://www.epcc.ed.ac.uk for further information.

1.4.2 TCD, Ireland

Trinity Centre for High Performance Computing, Trinity College Dublin, Ireland.

Trinity College has a long and distinguished record of pioneering inter-disciplinary HPC research in Ireland. HPC activity spans departments as diverse as Economics and Micro-biology, and has an active community working in the area of Bio-Informatics, as well as the more traditional users such as the physical sciences and engineering departments. The Trinity Centre promotes world-class research and encourages collaboration between research groups and industry throughout Ireland. Research interests include automotive and pharmaceutical manufacturing, textiles, medical imaging, acoustics, virtual environments, fluids, aerodynamics, films, data mining, traffic management, financial modelling, fisheries and agriculture.

The Centres mission is as follows:

- To encourage, stimulate, and enable internationally competitive research in all academic disciplines through the use of advanced computing techniques.
- To train thousands of graduates in the techniques and applications of advanced computing across all academic disciplines.
- To bring the power of advanced computing to industry and business with a special emphasis placed on small and medium sized enterprises functioning in knowledge intensive niches.
- To become a recognised and profitable element of the infrastructure of the Information Technology industry located throughout the island of Ireland.
- To develop, nurture, and spin-off several commercial ventures supplying advanced computing solutions to business and industry.

TCHPC has been very active in EU and National funded research programmes. In particular the Irish Higher Education Authority, PRTLI Euro 9 million project which will fund the upgrading of compute and deliver an immersive visualisation facility to Trinity later this year.

The University of Dublin, Trinity College was founded in 1592. Our six faculties are Arts (Humanities); Arts (Letters); Business, Economic and Social Studies; Engineering and Systems Sciences; Health Sciences; and Science. The city centre campus occupies some 47 acres (including the Trinity College Enterprise Centre). There is in excess of $200,000$m^2 of buildings, including beautiful historic architecture and state-of-the-art modern facilities.

The Trinity College Dublin Strategic Plan aims to consolidate the College's position as a research-led university at the forefront of education and research both nationally and internationally. The strategy is designed to take account of the new global environment, where the College finds itself competing with universities throughout the world.

1.4.3 Parallab, Norway

Parallab is the High Performance Computing Laboratory of the University of Bergen. Parallab is organized as a unit in the Bergen Center for Computational Science (BCCS), which is a section of UNIFOB, the University of Bergen's company for externally funded research. The Laboratory operates closely with various departments of the Faculty for Mathematics and Natural Sciences and in particular the Department of Informatics.

Parallab's staff pursues cross-disciplinary research and carries out projects with industry and other branches of science. The staff works with applications as well as basic research tailored to parallel and distributed processing, with a special emphasis on knowledge transfer to bridge the gap between computer science and computational sciences as well as the gaps between academic research and industrial needs.

Parallab has a track record in parallel and distributed computation that dates back to 1985. Parallab participated in various Framework III, IV, and V projects, including FRONTIER, SISCI, PARASOL, EUROGRID (and ENACTS). In the past six years, Parallab has developed considerable interest and knowledge in grid-enabling technologies. Parallab is partner in a number of national and Nordic grid initiatives and partner in the new Framework VI projects EGEE and HPC-EUROPA.

Parallab also operates the supercomputer facilities of the University of Bergen and provides support to the users of these facilities. Parallab is one of the four nodes in the Norwegian HPC infrastructure that is funded in part by the Research Council of Norway and in part by in-kind contributions from the participating partners. As such, Parallab has considerable interest in helping the user community of the Norwegian facilities in developing portable and efficient software.

1.5 The authors

1.5.1 Dr. Jean-Christophe Desplat

JC joined EPCC as an applications scientist for the TRACS programme in December 1995. Since then, he has developed some expertise in Scientific Visualisation that he is now teaching as part of EPCC's MSc in High Performance Computing. As part of his long running involvement with the TRACS programme, JC provided in-depth technical support to over 80 European visiting researchers using a wide range of simulation techniques on HPC platforms as diverse as Crays J90/T3D/T3E, Hitachi SR2201, Beowulf clusters, Sun Fire 6800 and 15k, IBM p690 cluster, etc.), hence bringing a valuable insight on users' requirements and working practices.

When time permits, he also participates in the research activities of the University of Edinburgh's Condensed Matter group for whom he co-developed the parallel generalised Lattice-Boltzmann code "Ludwig". His other hobby within EPCC also includes proposal writing, including EPCC's contributions to the EC-funded DEISA and HPC-Europa projects.

His current duties include management responsibilities for a number of European activities such as the HPC-Europa project (Transnational Access co-ordinator), the ENACTS network, and EPCC's involvement in TT@MED, the accompanying measure for the EC Medical cluster EUTIST-M. Beside these activities, JC is also managing EPCC's involvement in the EPSRC-funded e-Science pilot project the Reality Grid [Gri].

Contact: j-c.desplat@epcc.ed.ac.uk.

1.5.2 Dr. Chris Johnson

Chris Johnson is an Applications Consultant at the Edinburgh Parallel Computing Centre (EPCC), UK. He works mainly on the EC-funded HPC-Europa programme [HE] where he has responsibilities dealing with the technical needs of visitors to the Transnational Access visitor programme and also the management of associated resources. He is also doing technical work for the JRA1 "Performance Optimisation" activity and is managing the NA3 "Data Management" activity.

Dr. Johnson's background is in mathematical physics and computing and more specifically QCD (Quantum Chromodynamics). His current interests are in parallel programming in general and he teaches on several of the MSc parallel programming courses at EPCC.

Contact c.johnson@epcc.ed.ac.uk.

1.5.3 Dr. Geoff Bradley

Geoff Bradley is a research fellow and technical projects officer at the Trinity Centre for High Performance Computing (TCHPC), Trinity College, Dublin. His duties at TCHPC include technical co-ordination of the EC funded projects ENACTS and HPC-Europa as well as their grid facilities.

Dr. Bradley's current research focus is the development of novel algorithms for Molecular Dynamics simulations.

Contact: Geoff.Bradley@tchpc.tcd.ie

1.5.4 Dr. Jacko Koster

Jacko Koster is senior scientist at the Bergen Center for Computational Science (BCCS). His main activities include project coordination and research. Koster has a background in Computer Science and Applied Mathematics.

Koster is site coordinator for the University of Bergen in the national HPC programme in Norway and is currently also coordinator for three development projects within that programme. His current research interests include sparse matrix computations tailored to parallel/distributed processing, and in particular scalable domain decomposition algorithms for solving large systems of linear equations that arise in academic and industrial finite-element problems.

Contact: jak@parallab.no.

1.5.5 Mr. Jan-Frode Myklebust

Jan-Frode Myklebust is the main system engineer at the Bergen Center for Computational Science (BCCS). Myklebust is in charge for the daily operation and support of the HPC facilities of the University of Bergen. These facilities currently include an IBM p690 Regatta installation and an IBM 1300 Linux cluster.

Myklebust has participated in and provided support for various distributed computing, grid computing, and other internet-related computing and storage projects. As a result, he has acquired considerable expertise in middlewares like Globus and UNICORE. Moreover, he also developed a Java package that underpinned a large scale international collaborative distributed computing effort for finding K-optimal lattice rules.

Contact: janfrode@parallab.no.

2 The ENACTS Demonstrator Activity

2.1 Introduction

Here we describe the ENACTS Demonstrator activity itself, starting with the objective stated in the Technical Annex [ENA]. We then go on to explain how the objectives evolved as new trends within the user community were followed. We then give a description of the deliverables expected. We also describe the way in which this “Demonstrator” activity fits into the ENACTS project as a whole.

2.2 Objective

Objective [Demonstrating a European Metacentre]: *To draw together the results from all of the Phase I technology studies and evaluate their practical consequences for operating a pan-European metacentre and constructing a best-practice model for collaborative working amongst facilities.*

Since the time at which the objective was written, trends and technology have moved on and a fast and unpredictable pace. In particular, large scale scientific communities have necessarily become more concerned with handling the large amounts of data they produce, rather than simply concerning themselves simply with getting the most out of compute cycles on large machines. For this reason, we concentrated on the data aspects of the “pan-European metacentre”. The intention being to demonstrate that “The Grid” is able to solve many of the problems of data-sharing across what are becoming known as “virtual organisations”, something which will become increasingly important over the coming years (see [EU]).

The Demonstrator activity began on 30th June 2003 following on from a kick-off meeting in Dublin earlier in 2003 and involved three European Partners, previously described:

Centre	Role	Skills & Interests
EPCC	leading activity	Particle Physics & Globus
Parallab	participating	Physics & Globus
TCD	participating & providing users	Physics & Globus

2.3 Deliverables

In contrast to the earlier ENACTS activities, the main deliverables of the Demonstrator activity do not consist of reports, but of the actual demonstrator itself.

This document describes the work done during the ENACTS Demonstrator activity. It is intended primarily as a “Feasibility” report for those interested in setting up a pan-European metacentre based on our findings in setting up such a centre.

2.4 Results from Earlier Reports

As one can see from the objective above, this Demonstrator project draws on the results of earlier technology studies: Data Management in HPC [ENA03], Grid Service Requirements [ENA02b] and Grid Enabling Technologies [ENA02a].

In [ENA03] we find a survey of current trends in data management. This document goes into some detail about the way in which data requirements have changed over the last few years along with the corresponding improvements in file systems and data management techniques. A number of tools are identified, including “replication management” tools implemented using Globus [Glo] and the LDAP database [Ope]. There is some discussion of metadata and XML in particular. All of these tools are fundamental to a system known as “QCDgrid” which we have employed during this activity. There is also a discussion of tools such as XDR which create portable data. We have employed XDR in this activity to enable inter-operability between systems we had available.

In [ENA02b] data management is considered on several different types of the Grid middleware including Globus and in [ENA02a] experience of using and installing Globus and other Grid middleware is described.

2.5 QCDgrid

QCDgrid was originally written for the QCD (Quantum Chromodynamics) community within the UK (known as UKQCD). The software was written to be as general as possible and not specific to QCD.

QCDgrid is a layer of software written on top of Globus which makes use of many basic functions of the Globus toolkit (the security infrastructure and basic Grid operations such as data transfer) along with some more advanced features such as the replica catalogue. QCDgrid was originally set up to manage the data belonging to the QCD community within the UK and was comprised of 6 geographically dispersed sites with a total of around five terabytes of data. However, this figure is expected to grow dramatically as the collaboration's purpose built HPC system, QCDOC, comes on line later in 2004 [QCD].

The system has a central control thread running on one of the storage elements which constantly scans the Grid, making sure that all of the storage elements are working and that all the files are stored in at least two suitable locations. Hence, when a new file is added to any storage node, it is rapidly replicated across the Grid onto two or more geographically separate sites. Similarly, when a storage element is lost from the system unexpectedly, the data Grid software will automatically replicate the files that were held there on to the other storage nodes. In this way, the QCDgrid has the ability to cope with the loss of an entire site without losing any data.

If the node on which the control thread is running is lost, then control simply moves to another node (the "secondary node"). An administrator is informed of all problems, (*eg* loss of nodes failed checksums) via email.

As well as storing data, the system also stores the associated metadata. The metadata is stored both on the Grid itself, and also in a metadata catalogue. The whole system can be controlled via a GUI or the command line (see Fig. (3)).

The way in which these elements are linked together is all explained in the rest of this report.

3 The QCDgrid System

3.1 Overview of QCDgrid

So far we have talked about "QCDgrid" without going into detail. "QCDgrid" itself actually involves a number of elements:

- The QCDgrid software itself which controls the data storage, replication, etc.

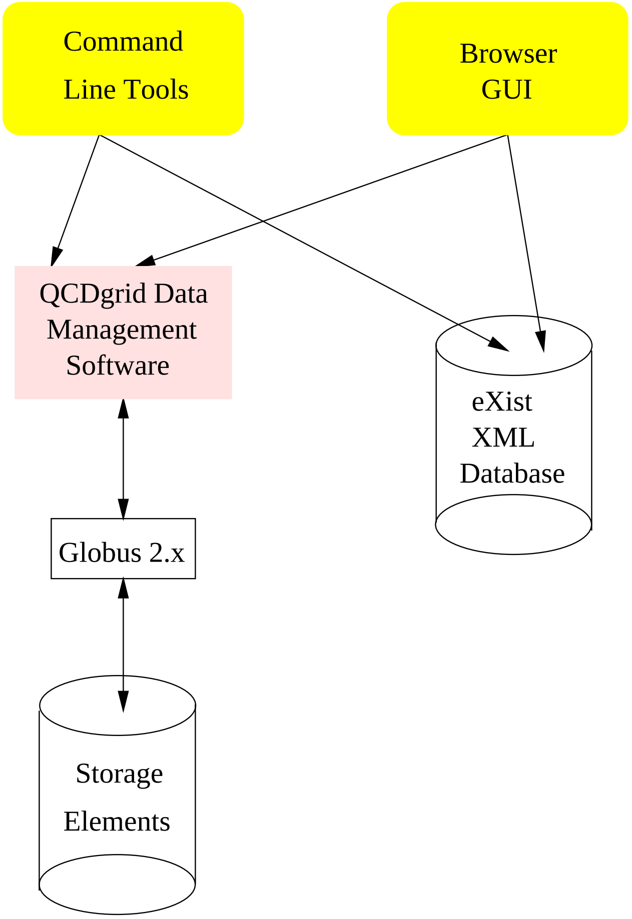


Figure 3: An Overview of the QCDgrid System

- The machines, underlying Software, etc. E.g. Linux PCs, Unix workstations, disk arrays, etc. all running Globus.
- The XML (Metadata) Catalogue which handles the metadata storage and searching.
- The QCDgrid GUI which controls all of the above.

3.2 Prerequisite knowledge and installations

Before we go on to describe the above elements of QCDgrid, it is worthwhile taking time to explain what prerequisite knowledge is needed to install and run QCDgrid, and what software is required to be installed on your system(s) before installing QCDgrid.

3.2.1 Hardware and Software

QCDgrid was built to run in any Unix environment and was developed on Linux. This report emphasises our efforts to port the software to a different Unix environment (namely Solaris) - this is described later. However, it is sufficient to say at this stage that what is required to run QCDgrid is a Unix environment with all the usual C (preferably GCC) and Java compilers. The machine must be set up in such a way that access can be allowed for a number of users to a central storage element which may just consist of an area of disk space. Globus 2.x must also be installed on all of the machines and basic Globus jobs must be allowed between all of the sites. In addition, the Globus "Grid Services" package must be installed on one (or more) of the nodes which will act as the "control node".

3.2.2 Knowledge

Once QCDgrid is set up, in order to use QCDgrid only a basic grasp of Globus and associated certificates should be required. For example, the ability to run a "hello world" Globus job between two machines is almost all you will need. The possible difficulties and frustrations in getting to this stage are highlighted later! See [ENA02a] for experiences in setting up and running Globus.

In order to make full use of the system, an understanding of XML metadata is necessary.

Installing the QCDgrid system yourself should be straightforward once Globus is up and running on all of the systems (a highly non-trivial task). To install the software on top of Globus a simple knowledge of compiling C and Java programs is required along with a good understanding of file Unix file and group permissions. There can also be issues involving the use of shared accounts as the control-thread itself will need an account to run in.

If Globus is not already installed on your system then this will have to be done. Later in this report we detail the issues involved in setting up the Globus software.

3.3 The QCDgrid Software

The QCDgrid software is written in the C programming language and interacts with Globus via system calls to Globus and the Globus Replica catalogue. The same software is installed on each participating system in the same manner, after accounting for local variations in compiling, etc. There are a number of configuration (.conf) files which control the behaviour of QCDgrid, allowing the system to be customised for each particular Grid. These files also determine the role of the node, for example whether or not the node is the "control-node" (see later).

The QCDgrid software is available in CVS from http://forge.nesc.ac.uk/projects/qcdgrid.

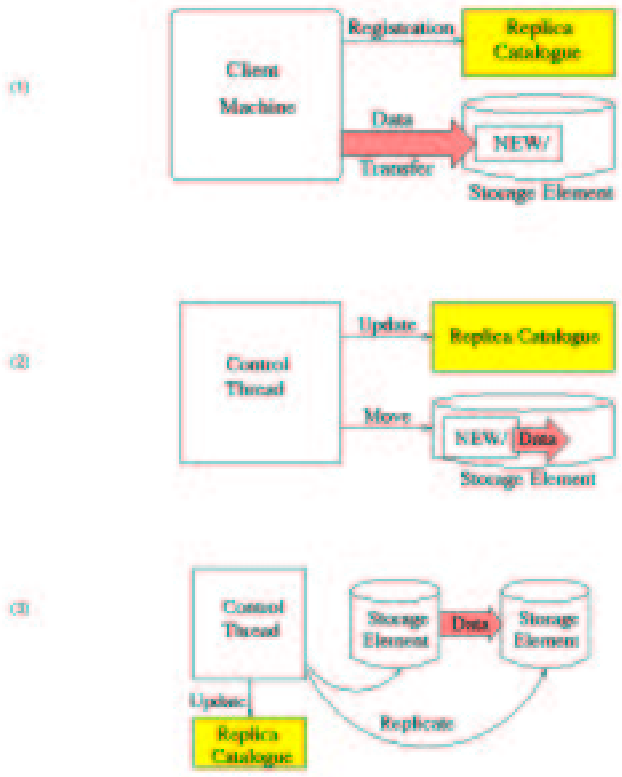


Figure 4: The QCDgrid replication procedure

3.3.1 The Replication Procedure

Once the system is set up a user can submit a piece of data (*eg* a binary file) to the Grid. This is done simply by issuing a *put-file-on-qcdgrid* command on the local machine, assuming that the Grid proxy has been started (*ie* a *grid-proxy-init* command has been run successfully).

The software then chooses a suitable storage element and copies the file to the “New” directory on that element. On its next scan, the control thread finds the new file and moves it to its actual home as well as registering it with the replica catalogue. On the following scan, the control thread (which is permanently running in the background) will notice that there is only one copy of the file and will thus make a copy at a suitable site, again registering this replica with the replica catalogue. This replication procedure is shown in Fig. (4).

In order to retrieve a file, the user simply issues a *get-file-from-qcdgrid* command. The nearest copy is then retrieved by the system. It does this by querying the replica catalogue which identifies a copy of the relevant file and copies it over via gridFTP.

There are a number of other commands available. For example, one can tag a file using the command *i-like-this-file*. This ensures that, provided space is available, a copy of the file in question is available at the local file store. The command *qcdgrid-list* allows one to see all of the files on the Grid. There are many other commands available for the administrator to add and remove nodes, etc.

Unfortunately, there is not yet any concept of file ownership and all files on the Grid and accessible to all members of the project and files can only be deleted by the administrator. However, this issue will be addressed in later versions of QCDgrid.

3.4 The underlying gridware

QCDgrid was originally built on top of the Globus Toolkit 2.0 and we discuss the porting of the software to Globus 2.4. It also makes use of the "replica catalogue" which is available in the "grid-services" package - an optional extra to the basic Globus software.

3.5 The Metadata Catalogue

3.5.1 Metadata

Essential to the Grid we were intending to set up, was the concept of metadata. Whilst the QCDgrid software controls the replication and retrieval of any kind of file with great success, (*eg* binary data file, XML metadata file, visualisation file, ASCII text, tar file etc.), this simply amounts to fairly sophisticated data management. What about the information that the data represents?

As any scientist or engineer knows, there are a number of ways of losing data, but as disk management, disaster-recovery and back-up strategies become more sophisticated and reliable, retaining data is not really the key difficulty. Losing *information*, on the other hand, is still very easy! Consider a binary data file which, although perfectly preserved, is not well described or is described in such a way that its description cannot be queried/searched. In the long term such a file may well become useless. Even if the contents of the file in question are well understood by those who created it, how can well could it be understood by others within a scientific collaboration, or from an entirely different collaboration? What happens when the creator of the file has moved on and no longer there to explain the details or origin of the data?

One solution is to standardise the way in which such files are described. Such a description, in effect data, describing data, is known as *metadata*. Conventionally the information which metadata represents has, if recorded at all, been stored in a rather *ad-hoc* way, perhaps within the file, or implied somehow by the directory structure or filename, or even simply implied by the size of the file. Often this information is stored separately within README files or even noted down in a lab book somewhere.

A strict definition of metadata is not easy and the line between data and metadata is not always obvious. As a general rule metadata should contain enough information to describe the properties of a piece of data. This usually implies the metadata should describe all of the input parameters and conditions, *ie* the machine on which the data was produced and the date on which it was produced. The actual “output”, *ie* the data, should be left as binary, although it maybe appropriate to store one or two output parameters as metadata if it aids the description of the data itself.

3.5.2 XML

So how do we actually store the metadata? The most common way of writing down metadata is using XML (Extensible Markup Language) [XML]. This format allows metadata to be stored in an arborescent fashion with a tag structure much like HTML and is also written is simple plain text form. For example, here is an extract from one of our own XML files:

```
<gauge_configuration>
...
 <implementation>
   <machine>
     <name>epcc-cray-t3e</name>
     <cpu_node_type>alphaesque</cpu_node_type>
     <number_of_nodes>64</number_of_nodes>
     <node_memory>64 Mbyte</node_memory>
   </machine>
   <code_version>MILC-IKS-vX.x</code_version>
   <date>2002-06-02</date>
   <precision_of_measurement/>
 </implementation>
...
</gauge_configuration>
```

In order to test the “correctness” of an XML file, it is necessary to compare it against a schema. A copy of the schema corresponding to the above file can be found on the UKQCD website.[1]

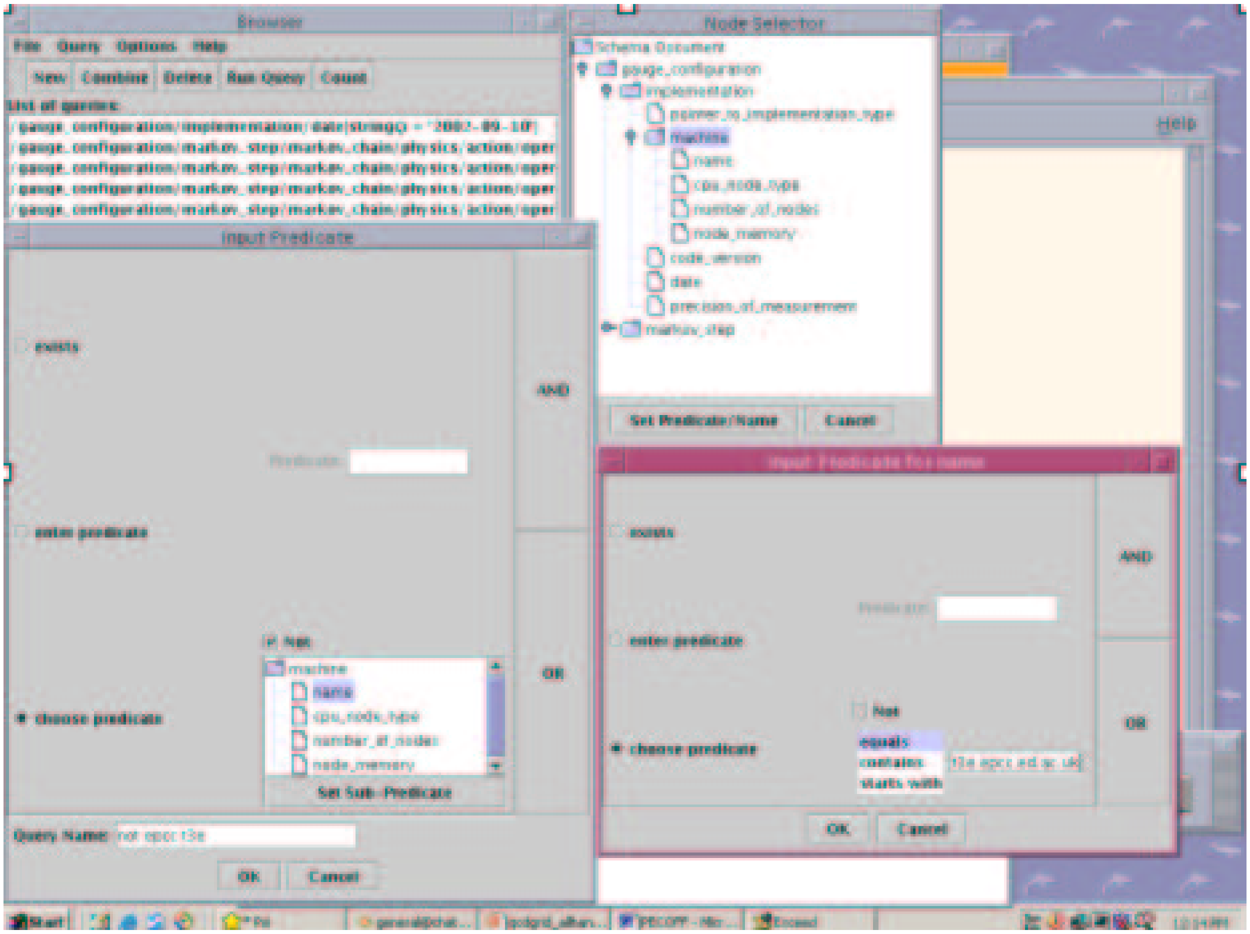
More information on the use of XML to promote the reuse of data and increase the possibilities of data portability can be found in [ENA04].

3.6 The QCDgrid GUI

So far we have described the main QCDgrid software as a “command-line” tool. However, the GUI described above allows both QCDgrid data-management software and the XML database to be controlled from a browser interface. Fig. (5) shows this browser

[1] http://www.ph.ed.ac.uk/ukqcd/community/the_grid/xml_schema/xml_schema.html

in use. The idea is that users simply submit the XML file describing their data and the actual data file itself using the Browser.



/gauge_configuration/implementation/date[string() = '2002-09-10']
/gauge_configuration/markov_step/markov_chain/physics/action/oper
/gauge_configuration/markov_step/markov_chain/physics/action/oper
/gauge_configuration/markov_step/markov_chain/physics/action/oper
/gauge_configuration/markov_step/markov_chain/physics/action/oper

Query Name: not epcc t3e

t3e.epcc.ed.ac.uk

Figure 5: A screen-shot of the QCDgrid GUI

3.7 Submitting Data and Metadata

Using the browser one can both add data and XML metadata to the datagrid and add XML data to the XML catalogue at the same time. To do this, simply ensure that the “pointer_to_value” tag in the XML document corresponds to the filename of the data file, *eg*

```
<gauge_configuration>
...
  <markov_step>
  ...
    <value>
      <pointer_to_value>datafile.dat</pointer_to_value>
    </value>
  </markov_step>
</gauge_configuration>
```

This will then submit the datafile as “datafile.dat” and the XML file as “datafile.dat.xml”.

This operation is the equivalent of doing a “put-file-on-qcdgrid” for the two files with logical Grid filenames “datafile.dat” and “datafile.dat.xml” and then doing a “put” within eXist of the file “datafile.dat.xml”.

3.8 Searching the metadata catalogue

The easiest way to search through the XML is to use the browser. For example, to search for a element (*eg* date) with a given value you first have to set up the query. To do this click on “Search Metadata Catalogue”. Click “New”. Select the element you wish to search on. Then click “Set Predicate/Name”. Then click “choose predicate”, “equals” and type in the value you wish to search for (*eg* 2002-06-02). Finally choose a “Query Name” and clock “OK”.

To run the query, go back to the Browser, select the query and click “Run Query”. It is best to use the “QCDgridResultHandler” which you can select via “Options”, “Set result handler”.

Queries can also be set up using the XPath query language [XPa]. Such queries can be entered directly into the search browser.

4 Deployment

4.1 Our Deployment

As previously described, QCDgrid was originally deployed on sites around the UK which were homogeneous both in terms of the architectures used and the operating policies and Grid environments present. Our task within the ENACTS project was to break away from these constraints and attempt to deploy QCDgrid across sites using the “typical” facilities already present. We wanted to set up a “pan-European” Grid and this implied setting up a Grid across the three participating partners in the activity (See Fig. (7)). To summarise, this meant that instead of deploying QCDgrid over a set of Linux machines all running GT (Globus Toolkit) 2.0 and all using the same Certificate Authority, we were provided with the following facilities:

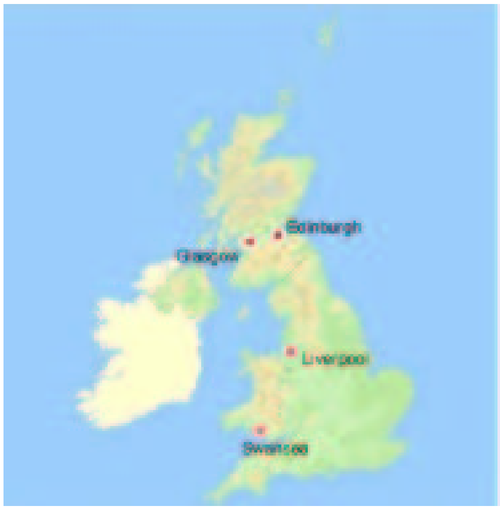


Figure 6: The deployment of QCD-grid by the UKQCD community.

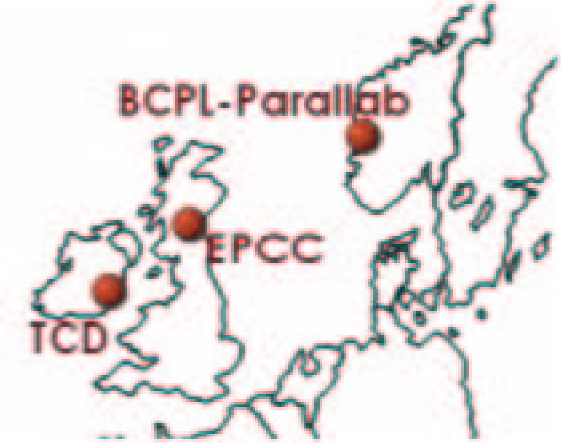


Figure 7: The Deployment of QCD-grid for the ENACTS Demonstrator activity.

Centre	Location	CA	Architecture OS	Gridware
EPCC	Edinburgh, UK	UK e-Science	Sunfire E15K Solaris 2.9	GT 2.4
Parallab	Bergen, Norway	NorGrid	Linux Cluster Redhat 7.1	GT 2.4 & Replica catalogue
TCD	Dublin, Ireland	Grid-Ireland	Linux Cluster Redhat 7.3	GT 2.4 & Replica catalogue

As can be seen although the version of GT used is the same in all cases, it is not the same version as was deployed by UKQCD. This was the first time the code had been ported to a heterogeneous environment and also the first time it had been ported to different certificate authority issuers.

In more detail, our architecture consisted of:

- A 52-processor, Sunfire E15K running Solaris 2.9. This machine is housed at EPCC, Edinburgh and is called **Lomond**. The batch system used to control parallel jobs is SGE (Sun Grid Engine). This machine runs Globus 2.4.
- A Linux cluster consisting of 32 dual-cpu Pentium III nodes, each with 2 Gbytes of memory running over a fast Ethernet connection. The operating system for this machine was Linux (RedHat 7.1). The batch system used for controlling job management was PBSPro. This machine is known as **Fire**. This machine runs Globus 2.4.
- A Linux cluster consisting of 32 dual PIII 1GHz nodes, each with 1GB of RAM. The interconnect consists of Myrinet for parallel inter-communication and fast ethernet for other TCP/IP applications. The operating system for the cluster is

RedHat 7.3 with OpenPBS and the MAUI scheduler controlling jobs management. The machine is funded as part of the IITAC project [REF], with the machine mchammer, running Globus 2.4, acting as the Grid gatekeeper.

A 32 node Linux cluster running Linux (RedHat 7.3). Myrinet interconnect.

This machine is known as **Mchammer**.

4.2 Deployment problems

Most of the problems faced when deploying QCDgrid in this fashion were actually related to the Globus Replica catalogue Globus Toolkit itself and this will be commented on first. The issues relating to the pan-European deployment of this software along using differing CAs and heterogeneous environments were actually of less significance, but are also worth commenting on.

The intention here is to describe the kinds of problems that are faced when setting up a pan-European Grid, based on what we encountered.

4.2.1 Globus and the Globus Replica Catalogue

There were really two separate issues facing us when dealing with Globus. Firstly there was the issue of setting up Globus to work between the sites in question, and secondly the issue of getting QCDgrid to work with the particular version of GT we were using.

At each of the three sites, installations of Globus had already been attempted and at two of the sites Globus was already in use. Our first task was to perform the most basic operation of running simple Globus jobs between the sites, with each of the sites able to act as either client or server to any of the other sites. In our case this was simply a two-way all-to-all task involving three sites, but it was still highly non-trivial. The reasons for this was due to both the fact that multiple CAs were involved (see later) and also to firewall issues which are discussed later.

It was surprising to find how many problems we were faced with when simply moving from GT 2.0 to GT 2.4. Some functionality previously found in libraries had disappeared and it turned out that the replica catalogue provided in GT 2.4 was not self-consistent. The problems associated with missing libraries were overcome with some simple re-writing of the QCDgrid code (the missing libraries were simply error handling routines). To solve the problems associated with the replica catalogue required us to completely rewrite the replica schema.

The installing of Globus on Solaris is not straightforward and for this reason the "information services" bundle was not installed. The implications of this were that the Solaris node could not act as the “control thread” and could not be used to submit data from. However, it was still able to act as a storage node.

4.3 Firewalls and Certificates

This section covers some of the issues observed by staff at TCHPC while setting up a Grid test-bed using globus, specifically with

- Firewalls and their configuration for globus services
- Globus Certificates and Certificate Authorities

4.3.1 Firewalls

A firewall is a device used to filter incoming/outgoing packets on a network. Most companies, Universities, Government agencies etc. deploy a firewall or firewalls to protect their internal network from the Internet. Figure 4.3.1 shows the firewall setup at TCHPC - other universities and companies would have similar setups.

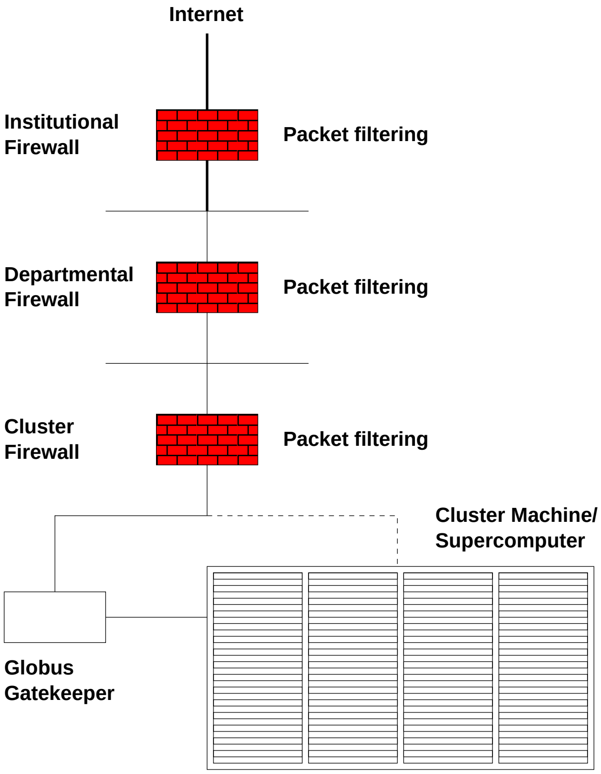


Figure 8: The firewall setup at TCHPC.

A firewall may be software based, *eg* netfilter/ipatables [net] running on a Linux/Unix machine, or it may be hardware based, *eg* a Cisco PIX or a Nokia Checkpoint.

A firewalls impact on setting up a Grid test-bed There may be several firewalls between two globus gatekeepers or between a client and server machine. It is possible that these devices will cause problems with the Grid setup by blocking some or all of the incoming and outgoing network traffic between machines. Often, the responsibility of configuring and monitoring firewalls is the responsibility of network administrators who may be wary of opening up additional ports for external services due to security concerns.

Grid Services Typical Grid services running on a Grid server node are

- Globus gatekeeper, port 2119
- Globus Resource Information Service (GRIS), port 2135
- Gsiftp, port 2811

Network traffic from remote client and server nodes must have access to gatekeeper nodes on these ports. Stateful connections from local clients and local gatekeeper machines must be allowed to remote Grid server nodes. Normally, incoming connections are restricted to a high port range [xxxxx-xxxxx] on the server machine. Typically 20 ports per user are required. These port ranges must be configured during the service setup of globus and access must be allowed through any intermediate firewalls.

It is also useful to allow secure shell (ssh) connections through the firewall to allow client users to log into the server machines.

Additional information on firewall requirements is available from the paper by Welsh, [Wel].

Security Risks When installing any new application, due consideration must be given to potential security vulnerabilities imposed by the application - especially on computers connected to the internet. A vulnerability in an application may lead to elevated privileges or unauthorized access. In the case of Grid computing, a compromise on one site could lead to compromise at other sites within the Grid.

In this testbed, the Globus Packaging Technology 2.2.10, [Glo] and the Globus toolkit, 2.2/2.4 were installed on RedHat 7.X and 8.0 machines, [Red] and on a SunFire machine. RedHat 7.X, 8.0 and 9.0 are now at end of life, *ie* they are no longer supported by RedHat. These RedHat operating systems are outdates and the installed versions of globus have been superseded, but the effort required to upgrade a cluster to a later version of Linux and to a newer version of Globus is significant. One option is to use the updates (patches) for the Globus Toolkit 2.4 which are available at http://www-unix.globus.org/toolkit/advisories.html?version=2.4 and the updates for RedHat 7.X, 8.0 and 9.0 which are available from the Fedora Legacy project [Fed].

4.3.2 Setting up globus

There are two main options available when setting up and configuring globus:

- Accessing the Grid as a client, *ie* offering no services. The globus toolkit 2.4 and globus client software must be installed on the client machine.

- Running Grid gatekeeper, *ie* offering Grid services. The globus toolkit 2.4, globus server and sdk bundles are required. Additional packages are required for additional services.

Certificates

The Grid uses the public key infrastructure (pki) for authentication of users, resources and services. Each resource has a key pair, a public and private key. A trusted Certificate Authority (CA) must digitally sign a key pair to confirm identity. A key pair confirms authentication of identity; it does not allow access to resources. Resource authorisation is provided by the owner of a resource.

Certificates are issued by a national Certificate Authority (*eg* see http://marianne.in2p3.fr/datagrid/ca/ca-table-ca.html for more details). Certificates use X509 format, a certificate display and signing utility to create host and user Certificates.

User Certificates If a user is connecting to the Grid as a client, they must have the globus toolkit and client software installed on their computer and they require a user certificate. A user certificate, usercert.pem (public key) and userkey.pem (private key) are issued by a CA and stored locally (only) in a users .globus directory. Information on the user certificate is extracted using the *grid-cert-info -subject* command.

Example: Obtaining user certificates - the process in the Republic of Ireland. Grid Ireland [GI] are the Certificate Authority for academic institutions in the Republic of Ireland. A summary of the steps required to obtain a user certificate are shown below:

- Access the CA via a secure web browser connection on a trusted machine. A trusted certificate from the CA is downloaded into the browser.
- On a secure browser (https) apply for a certificate and provide a password for the certificate.
- Read the terms and conditions of the CA and present photographic identification to CA
- Once issued, the certificate is downloaded in Netscapes pcks12 format (.p12) format and export to a file
- Openssl is used to extract the usercert.pem and userkey.pem files from the .p12 file to .pem format

Several passwords are required throughout this process - further details of this process are available at the Grid-Ireland web site.

This can be a slow process (several weeks) and the user must use Netscape 4.7/4.8. The process for obtaining certificates in other countries in Europe is similar. The user can now verify their identity on the Grid by using the *grid-proxy-init* command.

Host Certificates To provide Grid services, the host site must have a verified host certificate signed by a CA. The obtain a host certificate, the host site must generate a host_request.pem file and send it to their national CA for signing. Signed host certificates (hostcert.pem and hostkey.pem) are installed in the /etc/grid-security directory.

Example: Obtaining a host certificate - the process in the Republic of Ireland. A summary of the steps required to obtain a host certificate are as follows:

- Access the CA via a secure web browser connection on a trusted machine. A trusted certificate from the CA is downloaded into the browser.
- A host certificate request (produced with the

  ```
  ‘grid-cert-request -service host -host ‘hostname‘’
  ```

  command) is uploaded into the browser
- Read the terms and conditions of the CA and supply photographic identification to the CA.
- Download the supplied pki files which are install in the /etc/grid-security directory.

Further information is available on the Grid-Ireland website.

This can be a slow process (several weeks).

ISO 17799 ISO 17799 [refb] is an internationally recognised IT security standard consisting of two parts

- a code of practice for information security management.
- a specification for an information security management system.

In March 2004, Trinity College Dublin started following the IT security management guidelines as set out in the ISO 17799 documents. As a result, restrictions must be placed on third parties authorising access to managed systems within TCD. At present, it is unclear what implications this will have for Grid computing and certificates issued from Certificate Authorities.

Accessing Grid resources For a client user to gain access to a Grid service providers' resources

- the user must have a valid user certificate from their CA
- the provider must ‘trust’ that CA, *ie* the provider must have a copy of
  - the CA’s certificate, <hash>.0
  - the CA’s signing policy, <hash>.signing_policy
  - the CA’s certificate revocation list, <hash>.r0

  stored in the /etc/grid-security/certificates directory.
- the users Grid certificate information (the output of the *grid-cert-info -subject* command) must be mapped to a local username and stored in the providers grid-map file in the /etc/grid-security directory.

4.4 Firewalls and Certificates - Summary/Conclusions

Before setting up globus, it is advisable to sort out access to/from server or client machines. There may be security implications in opening up additional ports that need to be addressed.

4.5 Porting to Solaris

This was the first time Solaris had been used as an operating systems for one of the nodes. At EPCC, Solaris 2.9 is employed on the Sunfire E15K. This presented two significant challenges. Firstly, we had cope with installation difficulties with Globus on Solaris. Secondly, the code itself had to be ported to Solaris. Whilst we did not install Globus as part of this activity, the previous installation available to us was not complete due to the difficulty of such an installation. More specifically, the "Information Services" bundle had not been installed which contained some of the *pthreaded* versions of libraries. In an attempt to compensate for these missing libraries, we tried to use pre-compiled libraries from another, similar machine. However, this failed to work. As a consequence of this problem, the Solaris node of our system was never fully working.

4.5.1 Compiling the code on Solaris

We had available the GNU GCC compiler, with which the QCDgrid software had originally been compiled. This meant that the basic compilation did not require attention other than the usual small changes to the Makefile for the locations of libraries, etc.. However, the move from GT2.0 to GT2.4 did require some changing of the code as a number of features no longer exist in GT2.4. These missing features turned out to be simply error handing routines and were not important.

Some aspects of the QCDgrid were specific to the operating system and compiler they were originally written for

- The GCC compiler - We did not attempt to port the code to a different compiler.
- **The *df* command** It is surprisingly difficult to determine the amount of space left of a file system in a non-system specific manner. QCDgrid makes use of the basic Unix *df* command. Unfortunately the output from this command varies from system to system and this took sometime to getting working over our Grid of heterogeneous architecture and operating systems.

4.5.2 Other issues

XML database

There were a few minor problems in installing the XML database but these were soon overcome. These problems were mainly due to minor changes in the distribution of the eXist software.

The Java GUI was very easy to install and only required a few very minor changes to the code to allow it to run successfully. The Java browser does run very slowly if running over an X connection from one machine to another on our Grid, but this is not usually necessary as the browser can be installed on local machines.

4.5.3 General Issues relating to the wider scale use of QCDgrid

- Some of the “Timeout” parameters had to be changed due to the fact that the nodes were more geographically separated in our deployment than in the previous (UKQCD) deployment. Essentially this was just due to a higher “round trip time” (RTT) for packets sent across Europe using the Globus software.
- The code was adapted so that it could handle different batch systems and job managers over Globus. This simply required two extra parameters to the `.conf` files:

  ```
  extrarsl=(project=enacts)

  extrajsscontact=/job-manager-fork
  ```
- The software performs MD5 checksums to check whether the data has arrived successfully after being transfered. We found this often failed even when the files had arrived safely and intact. We are presently investigating the cause for this problem. It is possible to reduce the frequency of performing the checksum which has the effect of reducing the frequency of the error, but the checksum utility is useful for ensuring data integrity which is lost when the checksum feature is disabled.
- The software made heavy use of log files which document the behaviour of the control node. However the amount of information can easily be reduced.

4.6 Success of the Deployment

We now have QCDgrid running over the three centres with all three centres acting as storage nodes. Unfortunately, it is not possible to submit data from the Solaris node due to the missing “grid services” bundle.

5 Scientific Scenario

5.1 Choosing a code/community

In order to make our demonstrator as realistic as possible, we wanted to make use of a genuine scientific study. We were fortunate to get help from members of the QCD community. We investigated the use of three different QCD codes:

- GHMC - Generalised Hybrid Monte Carlo code from the QCD collaboration. We originally planned to use the GHMC code for our study. However we decide to use code that had already been made fully portable.
- QDP++ - QCD Data Parallel code written in C++. The QDP++ code would have been a very useful code to use given that it has built in XML capabilities. However, this code is still in development and we did not have enough time to deal with any issues the may have arisen because of this.
- The MILC code. This code is very portable and runs easily on most systems, including the systems we were going to use. This is the code we chose to use.

5.2 Aims & Achievements

The aims and achievements of the data Grid project are reviewed below.

5.2.1 Scientific aim: Modelling Quark Confinement and Breaking the QCD flux tube.

The scientific aim of this demonstration project was to use the MILC code (with some modifications) to model Quantum Chromodynamics (QCD). The particular area of interest is the study of confinement - no free quarks or gluons have been observed in nature even though they are thought to be the fundamental field of QCD.

As two quarks are pulled apart, the force between them stays constant (like an elastic string), even at large separations, R. The force stays constant because a tube of flux forms between the two quarks, Figure 9. This indicates that the energy of the system keeps growing. This is different to the theory of electromagnetism, where the force between two charges falls off as $\frac{1}{R^2}$ as charged particles are pulled apart.

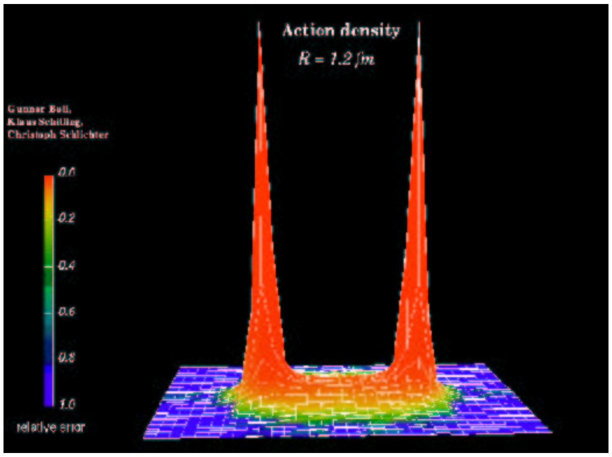


Figure 9: A quark pair separated by a flux tube, courtesy of Gunnar S. Bali [Bal].

Theory

In QCD, the flux tube can break when two quarks are created by quantum fluctuations of the vacuum - this is known as 'string breaking'. In the Yang-Mills theory (which has no quarks), the flux tube can keep on stretching.

SU(3) Yang-Mills theory	Gluons only
Quantum Chromodynamics	Quarks and gluons

The aim of this projects is to observe evidence of 'string breaking' in a Monte Carlo simulation of QCD on a lattice. This calculation requires supercomputing resources to generate an ensemble of QCD vacua with a configuration occurring with the right probability.

Calculation details

This project uses the publicly available MILC software suite to generate these vacua, then measures the Wilson loops on the ensemble. By studying a qualitative difference between the two theories as the Wilson loops get bigger or smaller, it is possible to predict if 'string breaking' occurs. These simulations produce a large number of binary data files. These data files are then used to seed further simulations on nodes within the Grid.

6 Portability

6.1 Introduction

The use of binary formats for storing large scientific datasets is usually very efficient (in both space and time). The representation of data in binary format is very close to the representation of data in computer memory, and often it is a direct copy. Transferring data between computer memory and file is fast since there is no need for data conversion (overhead) that slows down this transfer and the disk storage requirements are about the same as the memory requirements. In addition, binary formats are often preferred in situations where data sharing is not required or tightly controlled, especially if application performance and (disk) storage requirements are an issue.

Increasingly more often, scientists have a need to share raw data and derived data with their colleagues. However, sharing binary is often non-trivial, because there does not exist a single standard for a binary data format (or representation). The format of a binary data file is often recorded within one or two software programs created specifically to generate or process data and the only way to know the structure and contents of a binary data file is by examining the relevant I/O statements in these programs.

Fortunately, the IEEE floating-point standard has simplified a number of issues with respect to the representation of binary data and numeric precision. However, it is still necessary to know a few critical attributes before binary values can be interpreted in the right way. These attributes typically include the byte ordering (big-endian versus low-endian) and the blocksize (that describes how an application pads data values to maintain a uniform block size over a set of data segments). These attributes stem from the underlying physical representation of data in computer memory and are platform or environment (*eg* , compiler) dependent.

In addition, effective data sharing must provide more than just basic cross-platform read and write capabilities. Data sharing services must also support operations on a (binary) dataset that are common for many scientific applications. These include for example [BCE$^+$03]:

- the selection of a (relevant) subset of the data,
- combining data from multiple sources (*eg* , merging multiple datasets),
- the restructuring and reorganization of data,
- building index and catalogue structures for efficient access to data, and
- efficient transport of data among networked platforms.

The use of XML to represent binary data is tempting. An XML representation of a binary file however has a number of drawbacks. First, the XML representation would be significantly (2-4 times) larger than the (pure) binary representation. In addition, everything in XML is represented as a tree. This makes certain data structures that are common in scientific applications (*eg* , multi-dimensional arrays) difficult to represent or manipulate for extracting subsets of data.

The value of XML lies in representing the *metadata* associated with the (binary) data. As mentioned earlier, metadata typically describes things as how the data was created (parameters, algorithms used, etc.), when, where, and by whom. The metadata can also contain a standard canonical description of the structure and representation of the data itself.

6.2 XDR

The eXternal Data Representation Standard, XDR, protocol RPC 1832 [Sri95], is a standard for the description and encoding of data. It is a language written to concisely describe complex data formats and is compatible with the C programming language, [KR88]. XDR is used for transferring data between different computer architectures *ie* the data is architecture independent. Bit ordering is assumed to be consistent across all architectures and data is transmitted in four byte chunks. XDR is used in remote procedure calls (RPC).

6.3 Data Portability

Traditional HPC codes use binary format for I/O due to the significantly faster access speeds over ASCII format. However, binary data is architecture dependent, *ie* binary data produced on one architecture may not be readable on another architecture. The main reasons for this are

- little-endian (least significant bit first) / big-endian (most significant bit first) issues
- size of chars, ints, floats, doubles etc. may vary on machines even if there are no endian issues

The *Grid* is composed of heterogeneous architectures - so applications which produce binary data must be *grid-enabled* to produce architecture independent binary format. There are a number of options available for doing this, one option is to use XDR.

6.4 Why use XDR? A HPC users perspective

In the HPC arena, users generally require optimal I/O performance from their application code, so most applications use binary I/O format. XDR produces architecture independent binary I/O with an acceptable amount of overhead.

Users like portability of data if it allows them to access additional heterogeneous resources - XDR provides binary portability of data files on all machines. XDR files can be treated as if they were standard binary files

6.5 XDR - Implementation in *C*

Initial file access is handled in the normal way in a C program. An XDR stream is then created and used throughout the application for all I/O calls.

```
xdrstdio_create(XDR *xdrs, FILE *fp, enum xdr_op op)
{Initialises an XDR stream starting at *xdrs}
{writes to/reads from the standard I/O stream file}
```

{xdr_op op determines the direction of the stream}
{XDR data is written to/read from the XDR stream}

6.5.1 Primitive XDR data types

Using XDR, it is simple to produce machine independent primitive data types, *ie* chars, ints, floats, doubles etc. These are produced as follows

‘C’ representation	XDR
char	xdr_char(xdrs, i)
int	xdr_int(xdrs, i)
float	xdr_float(xdrs, i)
double	xdr_double(xdrs, i)

ie xdr_int(XDR *xdrs, int *i) translates between a C integer and its external representation where int *i is a pointer to an integer.

XDR also includes a number of very useful features such as transport of opaque data - an uninterrupted sequence of n arbitrary bytes of data. This can be used for converting complex structures directly into a file xdr_vector which allows translation of fixed length arrays to their corresponding external representations

More complicated high level data representations, based on enum, union, struct, etc. are also available.

6.6 Converting the MILC code to use XDR

One of the aims of this project was to take a piece of HPC code and assess the difficulties in modifying the code to produce machine independent XDR I/O. The application chosen was MILC - the MIMD Lattice Computation (MILC) Collaboration [MIL]. The necessary changes to MILC were implemented to Grid-enable the code. [2]

The core I/O parts of MILC are as follows:

- Read in an ASCII control file.
- Read in a binary lattice file in serial or parallel.
- Produces binary I/O in serial or parallel.

As a test scenario, serial binary I/O was modified to produce serial XDR I/O. To do this, it was necessary to modify the input and output routines and define new data types for reading in/out XDR format. This involved rewriting approximately twenty functions, with about 1000 lines of additional ‘C’ code.

[2] It should be noted that the MILC code was already capable of producing machine independent binary files.

It was reasonably straightforward to add XDR format to the MILC code. Therefore it is reasonable to assume that an applications developer would readily be able to modify their own code to incorporate XDR I/O.

Across the heterogeneous architectures in this project, there were minor compilation issues. It was necessary to use the xdr and rpc headers on the Linux test machines (mchammer & fire). Additionally, the nsl library was required on the SunFire machine (lomond)

6.7 Conclusions:

The main conclusion from this section is that it was reasonably straight forward, but time consuming, to implement XDR I/O in MILC. Only minor changes were required to the way I/O was handled so it should be possible to alter legacy HPC codes to use XDR. This modified code would then be more portable.

6.8 BinX

As mentioned earlier, XML can provide a very useful mechanism for representing metadata, but it is often inappropriate for representing large scientific datasets themselves.

The Binary XML description language (BinX, [BCE$^+$03]) provides the ability to describe the physical representation and the overall structure of arbitrary binary data files. BinX defines an annotation language for representing metadata for binary data streams, and specifies a data model to describe the data types and structures contained in binary data files. As an annotation language it has the power to describe (a broad range of) primitive data types, such as character, byte, integer, floating-point, and void. The BinX language also supports an extensible set of of data structures. The basic data structures include multi-dimensional arrays, ordered sequences of data fields, and unions (that allow dynamic specification of data types when reading a binary data file).

Another important data reference model in BinX is the ’defineType’ mechanism, which enables users to define their own data structures as macros that can be reused (referenced) repeatedly. These user-defined data structures are constructed from the basic data types and basic data structures that are defined within BinX or from other user-defined data structures. This mechanism is important as the user can assign semantically meaningful names to types and individual data elements.

A BinX document describes the structure and format of a binary data file and can also contains a URL which points to the actual binary data file. Overall, a BinX document can be used to describe three levels of features in a binary file: the underlying physical representation (*eg* , bit/byte ordering), the primitive and user-defined data types that are present in the binary file, and the structure of the binary data itself.

An example of a BinX document is given below [BCE+03]. The root tag is `<binx>`. User-defined type definitions are contained within the `<definitions>` tag. In this example, the complex number type `<complexType>` is declared that contains two floats called real and imaginary, respectively. The file description is contained within the `<dataset>` tag. In this example, the dataset is located in the (local) file testFile.bin. The file contains a float (StdDeviation), an integer (IterCount), followed by a fixed-sized two-dimensional array. Each entry in this array is of the newly defined complex type.

```
<binx byteOrder="bigEndian"
      blockSize="32">
  <definitions>
    <defineType typeName="complexType">
      <struct>
        <float-32 varName="real"/>
        <float-32 varName="imaginary"/>
      </struct>
    </defineType>
  </definitions>
  <dataset src="testFile.bin">
    <float-32 varName="StdDeviation"/>
    <integer-32 varName="IterCount"/>
    <arrayFixed>
      <useType typeName="complexType"/>
      <dim indexTo="99" name="x">
        <dim indexTo="4" name="y"/>
      </dim>
    </arrayFixed>
  </dataset>
</binx>
```

A standalone BinX document requires minimal additional storage space and network bandwidth to maintain. This representation is suitable for applications that manipulate the entire binary file. To manipulate subsets of the binary file, it can be advantageous to build a *DataBinX* representation of the binary file that merges the values from the binary file with the elements from the BinX document. Such document can then be used to perform queries (based on XML annotations) over a binary file. However, as the DataBinX document is typically much larger compared to the original binary file, it is not meant to be a long-term persistent representation scheme.

In certain cases, it can be useful to include the BinX description of the binary file in the binary file itself. If the BinX description of the binary file is stored in a different file, there is a danger that the correspondence between the two files may get lost and then the binary data would be rendered effectively useless. BinX also allows for this inclusion.

The XDR standard differs from BinX in that XDR defines aspects of the data encoding. BinX is intended to describe any (most) encodings rather than specify the features of the encoding that should be used. Another difference with XDR is that BinX is XML-based. Anything that can be (structurally) represented in XDR can be represented in BinX.

The BinX software library provides a core C++ API to read and write binary files based on the BinX description. The library implements a broad range of generic utilities that solve common data transformation problems between binary data formats. The high-level API can be used to develop domain-specific applications that require data extraction and data conversion of arbitrary binary data files. In many situations, a scientist or application will only have to deal with the (compact) XML description of the binary data and not with the (large) binary data itself.

The BinX library provides platform independence. Once data is described by a valid BinX document, it should be possible to read and write it (using the BinX library) on any platform regardless of the platform's native configuration.

Finally, BinX is being developed by the Edikt project [EDI] at the National e-Science Centre at the University of Edinburgh. The edikt project aims to construct novel software tools that underpin the linking, management and interpretation of the vast amounts of data available on global networks.

6.9 Making Our Data Portable

We would like to have employed both XDR and BinX over the course of the project, but unfortunately a working version of BinX for Solaris was only made available towards the end of the project which did not give us time to implement BinX.

6.10 Related Work

Recently, the Data Format Definition Language (DFDL, [DFD]) working group in the Global Grid Forum (GGF) grew out of BinX. DFDL is an XML-based language to delineate the structure and semantic content of all type files. BinX is a subset of DFDL, restricted to binary files and supporting limited semantics. Related work to BinX are the Binary Format Description language (BFD, [BFD]), the Hierarchical Data Format (HDF, HDF5, [DAA, refa]), the External Data Representation (XDR, [Sri95]), and the Resource Description Framework [LS99].

The Data access and Integration (DAIS) Working Group in GGF has specified protocols and interfaces for a Grid DATA Service (GDS). The DAIS specification is broad and flexible, allowing the construction of GDSs that provide access to relational databases, XML databases, and file systems composed of directories, text files and binary data files. BinX can be packaged as a GDS to access binary data files.

For other references to (portable) data management tools, we refer to the Sectoral Reports on *Data Management in HPC* and *Software Reusability and Efficiency* that were released earlier in the ENACTS network [ENA03, ENA04].

7 User Feedback

7.1 Successes

This demonstration project was successful in a number of areas:

- A data Grid has been successfully deployed on three clusters/supercomputers with QCDgrid running across the Grid.
- An XML Catalogue is operating with an XML Schema to describe the MILC metadata, [May], which is stored in an eXist database.
- The MILC code has been altered to produce machine independent XDR output.
- The MILC code has been altered to readin/writeout XML datafiles.
- The test users have certificates and accessed the data Grid.
- All components of the Grid have been demonstrated to the users.

7.2 Problem areas

The users had conceptual problems with the Grid and understanding the purpose of the Metacentre. Their limited knowledge of Grid technologies automatically led them to believe that they were gaining access to a computational Grid as opposed to a data Grid.

7.3 User feedback

The users were two QCD scientists in the School of Maths, Trinity College Dublin.

Q. How familiar were you with Grid technologies prior to this project ?

A. Only aware of a user working on this type of project at Edinburgh University.

Q. Would you be more/less likely to get involved in a Grid project now ?

A. It would be nice to use such resources, but I think it would need a huge number of nodes before it would be useful as other groups would also need access to the machine. We would rather run on local machines with guaranteed resources.

Q. What functionalities were missing from the Grid test-bed ?

A. Ideally, the Grid would have one central node that allocates jobs depending on the load of each cluster/supercomputer. Effectively the Grid should appear as one large cluster. Additional web based documentation of resources etc. required (Note: this was provided).

Q. Would access to this Grid improve your productivity/efficiency and how would it alter your work practices ?

A. Having loads of CPUs is helpful, but it would only really be usable if I didn't have to go searching for idle machines. A well organised file server would be useful

Q. Would you engage in collaborative activities more readily via the Grid ? Would you share datafiles, results, etc. ?

A. Yes !

7.4 Additional user feedback

The following additional user feedback was provided

- Having machine independent data is useful.
- It is useful to have the binary data accessible on all sites.
- The metadata for each jobs is very useful because if you're sharing data with other users you can easily find out the parameters they used to generate their results.
- Automatic upload of XML & binary output after run has completed would be useful. The MILC code was modified to do this.

- It would be desirable to have the entire jobs submission process available through a web interface.
- A global queueing system, global 'qsub' would be very useful.
- The large XML Schema used for MILC has many blank fields for this application which is confusing and messy.
- The whole Grid system seems to be a bit delicate !
- Where's the computational Grid ?
- In the longer term it would be better to take software written by the TCD QCD group and Grid enable it.
- The whole Certificate issue was very frustrating !

8 Feasibility

Our ENACTS "Demonstrator" activity has shown that it is feasible to set up a datagrid across geographically dispersed sites using available technology provided that considerable effort is first put in to set up Globus and it's associated packages. Given more time we could have increased the number of users to our system which would have provided us with more feedback.

One could also ask the question *"How would such a Grid scale up to use by more sites and users?* There does not appear to be an obvious limitation of the number of sites or number of users within QCDgrid itself although in the future it would be useful have more control over the ownership of files stored in the Grid. For example, it is not presently possible for users to delete files once they are on the Grid - this has to be done by the administrators.

The “demonstrator” itself was intended to take present technology and evaluate it, and we believe this has been successful. The activity has also motivated those involved in the QCDgrid project to generalise the software for use beyond their initial UK Grid.

Readers are encouraged to try the software for themselves [Per].

9 Dissemination

- A “Workshop” was held in Prague on March 11th-13th 2004. The second day of this workshop was devoted entirely to the Demonstrator activity. The presentations were very well received.
- A paper [CJD] was given at the UK All-Hands meeting being held in Nottingham in September 2004. This paper has been submitted jointly between ENACTS and QCDgrid.

10 Acknowledgements

We would like to thank James Perry and Lorna Smith and acknowledge the QCDgrid project as a whole. James’ help with setting up QCDgrid was invaluable to the ENACTS Demonstrator. We would also like to thank Craig McNeile and Bálint Joó of the UKQCD collaboration for several useful discussions about their present techniques. We would also particularly like to thank Mike Peardon and Jimmy Juge for acting as “users” of our Grid.

Appendix: Usage of QCDgrid

This short appendix is intended to demonstrate the use of QCDgrid.

References

[Bal] G. S. Bali. Website. `http://www.physics.gla.ac.uk/~bali/pics.html`.

[BCE$^+$03] R. Baxter, R. Carroll, D. J. Ecklund, B. Gibbins, D. Virdee, and Q. Wen. Binx - a tool for retrieving, searching and transforming structured binary files. All Hands Meeting, 2003. `http://www.edikt.org/binx/`.

[BFD] BFD. Binary format description language. Website. `http://collaboratory.emsl.pnl.gov/sam/bfd/`.

[CJD] L. Smith C. Johnson, J. Perry and J-C. Desplat. Experiences in setting up a pan-european datagrid using qcdgrid technology as part of the enacts demonstrator project. ISBN 1-904425-21-6. Website. http://www.allhands.org.uk/proceedings/.

[DAA] Goddard DAAC. Hierarchical data format (hdf). Website. http://daac.gsfc.nasa.gov/REFERENCE_DOCS/HDF/gdaac_hdf.html.

[DFD] DFDL. Data format description language. Website. http://forge.gridforum.org/projects/dfdl-wg/.

[DIR] DIRECT. Development of an inter-disciplinary round table for emerging computer technologies (direct). Website. http://www.epcc.ed.ac.uk/DIRECT.

[EDI] EDIKT. E-science data, information and knowledge transformation. Website. http://www.edikt.org/.

[ENA] ENACTS. The enacts technical annex. Website. http://www.enacts.org.

[ENA02a] ENACTS. Grid enabling technologies. Sectoral Report. Website, 2002. http://www.enacts.org.

[ENA02b] ENACTS. Grid service requirements. Sectoral Report. Website, 2002. http://www.enacts.org.

[ENA03] ENACTS. Data management in hpc. a joint scientific and technological study undertaken by cineca and tcd for the enacts network. Sectoral Report. Website, 2003. http://www.enacts.org.

[ENA04] ENACTS. Software reusability and efficiency. a scientific and technological study undertaken by parallab for the enacts network. Sectoral ReportWebsite, 2004. http://www.enacts.org.

[EU] EU. Community research and development information service. Website. http://www.cordis.lu.

[Fed] Fedora. Website. http://www.fedoralegacy.org.

[GI] Grid-Ireland. Website. http://www.grid-ireland.org.

[Glo] Globus. Website. http://www.globus.org/.

[Gri] Reality Grid. Reality grid. Website. http://www.realitygrid.org.

[HE] HPC-Europa. Hpc-europa. Website. http://www.hpc-europa.org/.

[KR88] B. W. Kerningham and D. M. Ritche. The c programming language, 2nd edition. Prentice Hall Software Series, 1988. ISBN 0-13-110370-9.

[LS99] O. Lassila and R. R. Swick. Resource description framework (rdf) model and syntax specification. W3C Recommendation. Website, 1999. http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

[May] C. Maynard. Website. http://www.ph.ed.ac.uk/ukqcd/community/the_grid/xml_schema.

[MIL] MILC. Mimd lattice computation (milc) collaboration. Website. http://physics.indiana.edu/$\sim$sg/milc.html.

[NES] NESc. Uk national e-science centre (nesc). Website. http://www.nesc.ac.uk.

[net] netfilter. Website. http://www.netfilter.org.

[Ope] Openldap. Website. http://www.openldap.org/.

[Per] J. Perry. Qcdgrid. Website. http://forge.nesc.ac.uk/projects/qcdgrid.

[QCD] QCDOC. Website. http://www.ph.ed.ac.uk/ukqcd/community/qcdoc/.

[Red] RedHat. Website. http://www.redhat.com.

[refa] Hdf5. Website. http://hdf.ncsa.uiuc.edu/HDF5/.

[refb] Information technology - code of practice for information security management. BS ISO/IEC 17799:2000, BS 7799-1:2000, ISBN 0 580 36958 7. http://.

[Sri95] R. Srinivasan. Rfc1832, xdr: External data representation standard. Network Working Group, Sun Microsystems. Website, 1995. http://www.faqs.org/rfcs/rfc1832.html.

[Wel] Von Welsh. Globus toolkit firewall requirements. Website. http://.

[XML] XML. Website. http://www.w3.org/XML/.

[XPa] XPath. Website. http://www.w3.org/TR/xpath.