CODAS

UKAEA
Restoring our Environment

JDN/H(06)039
Issue 1: 21/09/2006

UNCONTROLLED WHEN PRINTED

Running Grid Jobs from JET

Author: D.W.Robson

Abstract

The LHC Computing Grid is a large computing resource with 30,000 processors and 40 Petabytes of disk storage. Users at JET can run fusion analysis and simulation jobs on this grid. This document describes how users can register with the LCG, and how they can make use of its resources.

R.O.: D.W.Robson	Reviewers: K.Purahoo	Approver: E.M.Jones

Distribution
CODAS, NetworkServTeam@jetnet.jet.uk

Table of Contents

1 CONVENTIONS

When examples of operating commands are shown in this document, they are enclosed in a box. The command typed in by the user is shown in bold, and the output from the command is shown in italics. For example, running the uname command to show the name of the operating system would be represented by:

uname
Linux

The UNIX prompt is not shown.

2 INTRODUCTION

The LHC Computing Grid (LCG) is a large computing resource. At the time of writing it consists of approximately 200 sites with 30,000 processors and 40 Petabytes of disk storage. This is expected to rise to over 100,000 processors by the end of 2007. Whilst the LCG was initially designed for particle physics, it is now available for a number of other scientific disciplines including fusion.

This document explains how a fusion scientist can register with the LCG and use its resources.

3 REGISTERING FOR THE GRID

All users of the LCG must register. This process includes obtaining a grid certificate, becoming a member of a Virtual Organization (VO) and obtaining an account on a User Interface (UI) computer. The procedures for completing these steps are described below:

3.1 Reading the Grid Acceptable Use Policy

By registering as a Grid user, you are deemed to accept the conditions of use described in this document. The document is available here[1].

3.2 Obtaining a Grid Certificate

To use the LCG, you must possess a personal digital certificate from a Certification Authority (CA) recognised by the LCG. The CA is responsible for creating certificates and confirming the recipient is the correct person to own the certificate. Large CAs may delegate the confirmation to a local Registration Authority (RA). The procedure for obtaining a certificate includes meeting CA staff or RA staff in person and presenting photo ID documents such as passports or driving licenses. You should therefore choose a CA (or CA) that is convenient for you. You can find the nearest to you here[2].

If you are at JET, you should choose the UK e-Science CA. You can apply for your certificate at their web site[3]. You should also select the “Culham IT” RA. You will be prompted for a passphrase for your certificate that you must remember. You should then visit one of the Culham IT RA operators. They are David Robson (x4569) and Mark Rainford (x5027), and they can be

found in room J2B/1. They can authorize your certificate request after inspecting your site pass. Once authorized, you should be able to return to the UK e-Science web site and import your certificate into your web browser. You will probably be prompted for a password with which your browser will protect your certificates. Again, you must remember this password.

3.3 Joining a Virtual Organization

All users of the LCG must belong to a Virtual Organization (VO). This is a group of users with a common goal. The VO appropriate to you is **fusion**. You can join the fusion VO by following the instructions at the fusion VO web site[4].

3.4 Getting an account on a Grid "User Interface" Computer

In order to access the LCG, you must be able to log on to a "User Interface" computer (UI). This is the computer from which you will prepare and submit batch jobs to the LCG. It should have the latest version of the glite[5] software installed and must have an internet connection, and a firewall configured to accept all connections required by the glite software[6].

A UI called grid.jet.uk, has been installed at JET and is available to everyone who has an account on the other JET computers and is a member of the fusion VO. Anyone who qualifies for such an account can request one by emailing jac-adm@jet.efda.org. You will then be able to log on to grid.jet.uk from the JET Analysis Cluster[7] by issuing the command

ssh grid.jet.uk

grid.jet.uk is dedicated for the preparation, submission and monitoring of LCG jobs and should not be used for any other purpose.

3.5 Installing Your Certificate on the "User Interface" Computer.

After you have acquired a valid Grid Certificate, you must transfer it to the UI, and you must convert it into a format suitable for the grid. Firstly, you must export your certificate to a file. How this is done varies slightly from web browser to web browser.

If you are using Internet Explorer, you can go through the following menu options: TOOLS ➔ Internet Options ➔ CONTENT ➔ CERTIFICATES ➔ EXPORT

If you are using mozilla, you can go through the following menu options: EDIT ➔ PREFERENCES ➔ PRIVACY & SECURITY ➔ CERTIFICATES ➔ MANAGE CERTIFICATES ➔ BACKUP

If you are using firefox, you can go throught the following menu options: EDIT ➔ PREFERENCES ➔ ADVANCED ➔ VIEW CERTIFICATES ➔ BACKUP

You will be prompted for the name of a file to contain the exported certificate. You will need the password that your browser uses to protect your certificate. Now you will need to transfer this file

to the UI via the JAC. If you have used Internet Explorer, you can transfer the certificate from your PC to JAC by copying it to the Q drive. To transfer the file from the JAC cluster to the UI, type

scp myfile grid.jet.uk:

Now that you have transferred your certificate to the grid UI, you must log on to the UI and change the format of the certificate to one suitable for using the grid. You can log on to the grid from the JAC Cluster by running the following command

ssh grid.jet.uk

You are now on the grid UI. This is a linux computer. You should be able to list your recently transferred certificate with the **ls** command. You can now convert it to a format suitable for grid use by running the following two commands.

```
openssl pkcs12 -in myfile -clcerts -nokeys -out ~/.globus/usercert.pem
openssl pkcs12 -in myfile -nocerts -out ~/.globus/userkey.pem
```

You should then protect the newly created files by running:

```
chmod 644 ~/.globus/usercert.pem
chmod 600 ~/.globus/userkey.pem
```

And then you must delete all copies of the file exported from your web browser. You now have successfully set up your environment for Grid Computing. You will not need to repeat any of those previous steps until your certificate expires (typically after a year).

4 RUNNING JOBS ON THE GRID

This section describes how you can submit and manage jobs on the LCG.

4.1 Creating a Proxy Certificate

Before you can submit jobs or manage data on the grid, you must create a proxy certificate. This is a short-lived certificate that will be created from your longer term certificate that you acquired by following the steps in the previous section. The proxy certificate is not protected by a password, so is limited to a lifetime of typically 12 hours. You can create your proxy certificate by running the **voms-proxy-init** command. You will need to enter the passphrase you set up when you first applied for your certificate.

voms-proxy-init –voms fusion
Cannot find file or dir: /home/fbloggs/.glite/vomses
Your identity: /C=UK/O=eScience/OU=Culham/L=IT/CN=Fred Bloggs
Enter GRID pass phrase:
Creating proxy ..... Done
Your proxy is valid until Tue Sep 5 04:17:41 2006

Your proxy certificate has now been created. You can now execute commands to manage job and data on the grid. When you have completed your work, you should minimise the possibility of exposure of your proxy certificate by deleting it with the following command.

voms-proxy-destroy

Typically your proxy certificate will expire after 12 hours, and you will have to create another one. You can find out how long your certificate has to expire by running the **voms-proxy-info** command, see below.

voms-proxy-info
subject : /C=UK/O=eScience/OU=Culham/L=IT/CN=Fred Bloggs/CN=proxy
issuer : /C=UK/O=eScience/OU=Culham/L=IT/CN=Fred Bloggs
identity : /C=UK/O=eScience/OU=Culham/L=IT/CN=Fred Bloggs
type : proxy
strength : 512 bits
path : /tmp/x509up_u301
timeleft : 5:28:21

4.2 Defining a Job

Your next step is to define a job using a language called Job Description Language (JDL). This language allows you to describe a job by writing a number of lines in a file. A semicolon terminates each record. A simple example is shown below. It describes a job which will execute the program /bin/echo with an argument of “hello”. The standard output stream will be directed to a file called std.out, and the standard error stream will be directed to a file called std.err. These two file will make up what is called the “Output SandBox”. This contains the output files that can be retrieved from the remote computer and transferred to the UI computer from which you submitted the job.

This is a very simple example of JDL. In practice, you will need something a little more sophisticated. For example, this simple JDL example limits you to running an executable that has already been installed on the remote computer. Examples on more advanced JDL can be found in Advanced Job Management section of this document.

```
Executable            = "/bin/echo";
Arguments             = "hello";
StdOutput             = "std.out";
StdError              = "std.err";
OutputSandbox         = {"std.out","std.err"};
VirtualOrganisation   = "fusion";
```

You should name your file containing your JDL to something like hello.jdl. Before submitting the job, it is a good idea to test it by seeing which Computing Elements (CEs) on the grid can run your job. You can do this by running the following command

edg-job-list-match hello.jdl

Selected Virtual Organisation name (from JDL): fusion
Connecting to host lcgrb01.gridpp.rl.ac.uk, port 7772

****

COMPUTING ELEMENT IDs LIST
The following CE(s) matching your job requirements have been found:

CEId
ce.epcc.ed.ac.uk:2119/jobmanager-lcgpbs-fusion
ce2.egee.cesga.es:2119/jobmanager-lcgpbs-fusion
grid002.jet.efda.org:2119/jobmanager-lcgpbs-fusion
heplnx201.pp.rl.ac.uk:2119/jobmanager-lcgpbs-short
lcg02.ciemat.es:2119/jobmanager-lcgpbs-fusion
lcg06.sinp.msu.ru:2119/jobmanager-lcgpbs-fusion
lcgce01.gridpp.rl.ac.uk:2119/jobmanager-lcgpbs-S
lcgce01.gridpp.rl.ac.uk:2119/jobmanager-lcgpbs-fusionL
lcgce01.jinr.ru:2119/jobmanager-lcgpbs-fusion
cluster.pnpi.nw.ru:2119/jobmanager-pbs-fusion
node07.datagrid.cea.fr:2119/jobmanager-lcgpbs-fusion
ce-egee.bifi.unizar.es:2119/jobmanager-lcgpbs-fusion
heplnx201.pp.rl.ac.uk:2119/jobmanager-lcgpbs-fusion
ce1.egee.fr.cgg.com:2119/jobmanager-pbs-fusion
****

The output from the above command shows that there are 14 sites around the world with resources that match the requirements of your job. If there are no sites listed, there is probably something wrong with your JDL.

4.3 Submitting a Job

Now that you have prepared your JDL, you can now submit it to the grid. This can be done with the following command.

```
edg-job-submit hello.jdl

Selected Virtual Organisation name (from JDL): fusion
Connecting to host lcgrb01.gridpp.rl.ac.uk, port 7772
Logging to host lcgrb01.gridpp.rl.ac.uk, port 9002

**********************************************************************************
                        JOB SUBMIT OUTCOME
 The job has been successfully submitted to the Network Server.
 Use edg-job-status command to check job current status. Your job identifier (edg_jobId) is:

 - https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw

**********************************************************************************
```

Note that a job ID has been allocated to your job. In the above case, it is https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw. This is NOT a web address. You will need this job ID to manage your job through its lifetime.

4.4 Monitoring a Job

After your job has been submitted, you will need to monitor it so you can determine when it has completed. You can do this with the **edg-job-status** command and the job ID returned by the **edg-job-submit** command. An example of running this command is shown below.

```
edg-job-status https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw
**********************************************************************
BOOKKEEPING INFORMATION:

Status info for the Job : https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw
Current Status:     Running
Status Reason:      Job successfully submitted to Globus
Destination:        lcg06.sinp.msu.ru:2119/jobmanager-lcgpbs-fusion
reached on:         Mon Sep  4 15:19:17 2006
```

The “Current Status” field will change during the lifetime of the job. You may see that the job is initially scheduled to run at one site, only to see it being rejected and then scheduled at another. This is all part of the normal scheduling process. Eventually you will see the job move into the “Running” state, and then finally into the “Done” state. You can then retrieve the output files. See below.

4.5 Cancelling a Job

If you need to cancel a running or queuing job, you can do so with the **edg-job-cancel** command.

edg-job-cancel https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw

Are you sure you want to remove specified job(s)? [y/n]n **:y**

========== *edg-job-cancel Success* ==========
The cancellation request has been successfully submitted for the following job(s):

- https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw

==========

4.6 Retrieving Job Output

When your job has reached the “Done” state, you can retrieve all the output files that you specified in your JDL as part of the Output Sandbox. The files can be retrieved with the **edg-job-get-output** command that transfers the sandbox files to a new directory on the local User Interface computer. You can then post process these files as you wish.

edg-job-get-output https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw

Retrieving files from host: rb-egee.bifi.unizar.es (for https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw)

****

JOB GET OUTPUT OUTCOME

Output sandbox files for the job:
- https://lcgrb01.gridpp.rl.ac.uk:9000/SXaWvrT8rm_-WG0h_KWpRw
have been successfully retrieved and stored in the directory:
/tmp/jobOutput/fbloggs_ SXaWvrT8rm_-WG0h_KWpRw

****

ls /tmp/jobOutput/fbloggs_ SXaWvrT8rm_-WG0h_KWpRw
std.err std.out
cat /tmp/jobOutput/fbloggs_ SXaWvrT8rm_-WG0h_KWpRw /std.out
hello

5 MANAGING DATA ON THE GRID

5.1 Introduction

The Grid also provides a means of storing your data, without having to know its geographical location. Files can be stored on a remote computer called a Storage Element (SE). However, you don't have to remember the location of your files. The locations are stored in a catalogue. All you have to do is to access the files by its Logical File Name (LFN).

Before you run any commands on a UI, you must have a valid proxy certificate (see above), and you must set the environment variable to specify the location of the machine holding the catalogue. There is one catalogue per VO. As you will be part of the fusion VO, you need to set this variable by running the following command.

export LFC_HOST=rb-egee.bifi.unizar.es

If you are using JET's UI (grid.jet.uk), this has already been done for you. However, if you plan to execute data management commands from grid jobs, you will have to set this variable in your script. An example can be found in the Advanced Job Management section of this document.

5.2 Creating directories

First, you should create a "home" directory on the grid to store your data. You should choose a name of the format /grid/fusion/<username>. You should replace <username> by your username on the UI computer, or some other string that will identify you uniquely. In the examples below, a username of fbloggs is used.

You can now create the "home" directory by running

```
lfc-mkdir /grid/fusion/fbloggs
```

/grid/fusion/fbloggs is the LFN of the newly created directory. You can then use similar commands to create subdirectories under your "home".

5.3 Copying files to the grid and cataloguing them

Probably the data management command you will use the most is **lcg-cr**. This will copy a file from the local disk to the grid and catalogue it. For example, to copy a local file called myLocalFile.txt in your current directory to /grid/fusion/fbloggs/myRemoteFile.txt on the grid, you can run

```
lcg-cr --vo fusion -l /grid/fusion/fbloggs/myRemoteFile.txt   file:`pwd`/myLocalFile.txt
```

The `pwd` used above is a little trick used to expand the name of the file in your directory to a full pathname. Note that pwd is surrounded by backticks, not single quotes. On UK PC keyboards, the backticks can be found on the key below the escape key. On a UI machine, you will know your current directory, but this trick is useful if the command is executed in a grid job, as you will not be able to predict the current directory in advance.

5.4 Listing files on the grid

You can now confirm that your file is catalogued on the grid by listing the contents of the /grid/fusion/fbloggs directory.

```
lfc-ls  /grid/fusion/fbloggs
myRemoteFile.txt
```

5.5 Retrieving a file from the grid

Files catalogued on the grid can be retrieved to local disk with the **lcg-cp** command. The example below copies the

```
lcg-cp  --vo fusion lfn:/grid/fusion/fbloggs/ myRemoteFile.txt    file:`pwd`/myNewFile.txt
```

5.6 Deleting a file on the grid

You can also a delete file on the grid with **the lcg-de**l command. The example below deletes the lfn /grid/fusion/fbloggs/ myRemoteFile.txt grid file.

```
Lcg-del –a --vo fusion lfn:/grid/fusion/fbloggs/myRemoteFile.txt
```

5.7 Deleting a directory on the grid

Finally, if you have deleted all the files in a directory on the grid, it can be removed with the **lfc-rm** command.

```
lfc-del –a --vo fusion lfn:/grid/fusion/fbloggs/
```

6 BUILDING PROGRAMS FOR THE GRID

Not all programs are suitable for running on the grid. Firstly, the grid is primarily a batch environment, and is not suitable for displaying graphs or similar output. A common method of working is to submit "number-crunching" jobs to the grid to produce output data, and then to post-process that data in a graphical form on your own computer. Secondly, it is not worthwhile, running a short program (i.e. one that will complete in a couple of minutes), because grid jobs can take a couple of minutes before they start executing. Thirdly, your program may be incompatible with the software running on the remote computers. These machines will be running the Redhat Enterprise Linux 3.0 (RHEL 3.0) operating system, or a binary equivalent one such as Scientific Linux 3.0. This is a different version of Linux than is running on the JAC cluster. It will probably be a "bare" install, and will not have the additional software, e.g. matlab and IDL, that has been installed additionally on JAC. However, grid.jet.uk also runs Scientific Linux 3.0, and any program that runs on grid.jet.uk will run on remote grid computers.

The following table will highlight the issues of porting applications, from JAC to the grid.

Type of JAC application	Issues with running on the grid
C/C++	Will run without any problems, but if linked with shared libraries not part of the standard Linux distribution, the libraries will have to be transferred as part of the job. See Section "Running binary executables that have been built on JAC").
FORTRAN (built with the g77 compiler)	Will run without any problems, but if linked with shared libraries not part of the standard Linux distribution, the libraries will have to be transferred as part of the job. See Section "Running binary executables that have been built on JAC"). Alternatively, your executable could be linked statically.
FORTRAN (built with the Portland or Fujitsu compilers)	Will run without any problems, but the relevant FORTRAN runtime libraries, along with shared libraries not part of the standard Linux distribution, will have to be transferred as part of the job. See Section "Running binary executables that have been built on JAC"). Alternatively, your executable could be linked statically.
Shell scripts	Bash, csh and ksh scripts will run without any problems.
Perl	Perl will run without any problems. If extra modules are used, they will have to be transferred with the job, and the PERL5LIB environment variable set accordingly.
Python	Python will run without any problems. If extra modules are used, they will have to be transferred with the job, and the PYTHONPATH environment variable set accordingly.
MATLAB	Remote grid nodes will not have matlab installed, or have a matlab license. However, it should be possible to execute a matlab program that has been compiled with mcc. This has not been tested.
IDL	Remote grid nodes will not have IDL installed, or have a IDL license. It may be possible to execute a program compiled with IDL 6.2 or higher. Again this has not been tested.

7 ADVANCED JOB MANAGEMENT

JDL was introduced in the earlier section "Defining a Job" This section describes some more advanced JDL for practical cases.

7.1 Submitting Input Files with your job

The JDL in the section "Defining a Job" assumed that the program to be executed was already to be installed on the remote computers. In most cases, this will not be true. The job described below

will transfer the program myprog, in the current directory, via an input sandbox, to the remote computer for execution.

```
Executable          = "myprog";
Arguments           = "hello";
StdOutput           = "std.out";
StdError            = "std.err";
InputSandbox        = ”myprog";
OutputSandbox       = {"std.out","std.err"};
VirtualOrganisation = "fusion";
```

Note that the sandboxes are only designed for transmitting small files of a few kilobytes in length. Larger files should be catalogued to the grid via the **lcg-cp** command.

7.2 Running binary executables that have been built on JAC

You may wish to run an executable on the grid that has been compiled and linked on the JET Analysis Cluster (JAC). This will not run without some configuration, as the binary executables will probably contain references to shared libraries that exist on JAC, but not on the Grid. You can determine which libraries an executable uses by running the **ldd** program. For example,

```
ldd myprog
        linux-gate.so.1 =>  (0xffffe000)
        libc.so.6 => /lib/tls/libc.so.6 (0x0082f000)
        libpgc.so => /usr/local/depot/PGI/pgi-6.0/linux86/6.0/lib/libpgc.so (0xb7fcd000)
        libm.so.6 => /lib/tls/libm.so.6 (0x0094c000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x00812000)
```

The output from the **ldd** command shows that the myprog program uses five shared libraries. The linux-gate entry refers to a virtual shared library exposed by the kernel and can be ignored. The three libraries located under the /lib directory are standard linux libraries and will exist on the remote computer. The library in /usr/local/ is the runtime library for the Portland FORTRAN compiler and will not exist on the remote computers. This is clearly seen by transferring the program to grid.jet.uk and running the **ldd** command again on grid.jet.uk.

```
ldd myprog
        libc.so.6 => /lib/tls/libc.so.6 (0x00254000)
        libpgc.so => not found
        libm.so.6 => /lib/tls/libm.so.6 (0x005e0000)
        /lib/ld-linux.so.2 => /lib/ld-linux.so.2 (0x001b4000)
```

It is clearly seen that the libpgc.so shared library cannot be found and therefore this program will not run. In order to solve this common problem, the JAC administrators have catalogued a number of commonly used libraries on the grid. Your JDL should transfer your executable program and a

script, via the input sandbox, to the remote computer, and then execute the script. The script can copy the libraries from the catalogued entry, install them, and then run the executable.

An example JDL and script are shown below:

First the JDL …

```
// This JDL will run the myprog.sh script, which in turn will run the myprog program
Executable = "myprog.sh";
StdOutput = "std.out";
StdError = "std.err";
InputSandbox = {"myprog.sh","myprog”};
OutputSandbox = {"std.out","std.err"};
VirtualOrganisation="fusion";
```

… and now the script …

```
#!/bin/sh
################################
# This script (myprog,sh) will run the      #
# myprog program                            #
################################

# Set some environment variables
# Not all sites have VO_FUSION_DEFAULT_SE set
# The location of the fusion LFC needs to be set
# The LD_LIBRARY_PATH needs to include the JAC environment,
# that will be downloaded from an SE

if [ -z "$VO_FUSION_DEFAULT_SE" ]
then
    export VO_FUSION_DEFAULT_SE=grid001.jet.efda.org
fi
export LFC_HOST=rb-egee.bifi.unizar.es

# Download the JAC runtime environment

lcg-cp --vo fusion lfn:/grid/fusion/jet/jacenv.tgz file:`pwd`/jacenv.tgz


# Unpack it, this will create a lib directory containing the shared libraries

tar xzf jacenv.tgz

# Set the LD_LIBRARY_PATH variable to include the new lib directory,
# so the system knows where to find the shared libraries.

export LD_LIBRARY_PATH=`pwd`/lib:$LD_LIBRARY_PATH

# Set execute permissions on the executable.  Execute permission is only set
# by default for the file defined by the executable entry of the JDL

chmod 755 ./myprog

# Run the executable

./myprog
```

7.3 Running Long Jobs

There are two problems you might experience when running long jobs. The first problem arises because your program may exceed a CPU limit set by the remote computer. Typically, a CPU limit of 2 days may be set, but some sites may vary. If you find you are hitting a CPU limit, you can request that your job is scheduled towards a site with higher CPU limits. You can do this with a record in your JDL. For example, if you need a CPU time of at least 5000 minutes, add a record like the following.

Requirements = (other.GlueCEPolicyMaxCPUTime>5000);

The second problem arises because the proxy certificate that is transported with your job, will typically expire after 12 hours. You will find that your job will then be aborted. A typical error message from the **edg-job-status** command is shown below.

edg-job-status https://rb-egee.bifi.unizar.es:9000/SHi0KPb8X_vOAigKa4dSAg
****
BOOKKEEPING INFORMATION:

Status info for the Job : https://rb-egee.bifi.unizar.es:9000/SHi0KPb8X_vOAigKa4dSAg
Current Status: Aborted
Status Reason: Job proxy is expired.
Destination: lcg02.ciemat.es:2119/jobmanager-lcgpbs-fusion
reached on: Fri Sep 8 01:43:11 2006
****

The solution is to store a longer-living proxy certificate on a trusted computer called a Proxy Server (PS). When your job is submitted, it is initially passed to a Resource Broker (RB), a machine responsible for scheduling it to a suitable Computing Element (CE). If the RB is aware that the user for this job has saved a longer-living proxy certificate on a PS, it will regularly renew the certificate before it expires.

You can store your proxy certificate on a PS by running the myproxy-init command

myproxy-init -d -n
Your identity: /C=UK/O=eScience/OU=Culham/L=IT/CN=Fred Bloggs
Enter GRID pass phrase for this identity:
Creating proxy ..... Done
Proxy Verify OK
Your proxy is valid until: Wed Sep 27 11:00:25 2006
A proxy valid for 168 hours (7.0 days) for user /C=UK/O=eScience/OU=Culham/L=IT/CN=Fred Bloggs now exists on bdii-egee.bifi.unizar.es.

The output from the command shows that a proxy certificate that will last for 168 hours (a week) has been stored on a PS called bdii-egee.bifi.unizar.es. You can request a longer lasting certificate by supplying the additional parameters –c <nnn> , where <nnn> is the desired lifetime in hours.

Finally, you must inform the RB which PS you have used by adding an additional line in your JDL

```
MyProxyServer="bdii-egee.bifi.unizar.es"
```

7.4 Managing Large Numbers of Jobs

One of the biggest advantages of using the grid is that you can submit large numbers of jobs. You can manage large number of jobs by using appropriate arguments to the edg family of commands. The following commands will submit three jobs, and store their job identifiers in the file jobs.list. For brevity, the output of the commands is not shown for the next three examples.

```
edg-job-submit -o jobs.list hello.jdl
edg-job-submit -o jobs.list hello.jdl
edg-job-submit -o jobs.list hello.jdl
```

Now that the job identifiers are stored in the file, you can list the status of all the jobs with a single command

```
edg-job-status -i jobs.list
```

and, when the jobs are complete, you can download the contents of their sandboxes with a single command.

```
edg-job-get-output -i jobs.list
```

8 GLOSSARY

The following is a glossary of acronyms used in this document.

CA	Certification Authority
CE	Computing Element
LHC	Large Hadron Collider
LCG	LHC Computing Grid
JAC	JET Analysis Cluster
JDL	Job Description Language
LFN	Logical File Name
LHC	Large Hadron Collider
LCG	LHC Computing Grid
PS	Proxy Server
RA	Registration Authority
RB	Resource Broker
RHEL	Redhat Enterprise Linux
SE	Storage Element
UI	User Interface
VO	Virtual Organization

9 FURTHER INFORMATION

Further information can be found at the following locations.

The EFDA-JET Grid Cluster web site	http://w3grid.jet.efda.org (On JET intranet only)
Getting Started with EGEE (A brief User Guide for Fusion VO members)	http://grid.bifi.unizar.es/egee/fusion-vo/docs/GettingStartedWithEGEEv0_3.pdf
Job Description Language Attributes Section For the gLite Middleware (Release 1)	https://edms.cern.ch/document/555796/1

10 REFERENCES

1 https://edms.cern.ch/file/428036/3/Grid_AUP.pdf

2 http://www.eugridpma.org/members/worldmap/

3 http://www.grid-support.ac.uk/ca

4 http://grid.bifi.unizar.es/egee/fusion-vo/

5 http://glite.web.cern.ch/glite/

6 http://jra1mw.cvs.cern.ch:8180/cgi-bin/jra1mw.cgi/*checkout*/org.glite.site-info.ports/doc/middleware-ports.txt?rev=HEAD&content-type=text/plain

7 http://w3.jet.efda.org/CODAS/Document_Library/JDN/H00-12.pdf