Institute of Nuclear Physics
Grid deployment group
C. Filippidis C. Karafasoulis, C. Markou,
D. Loukas, K. Zachariadou

Grid Site User Guide

DRAFT 0

Outline

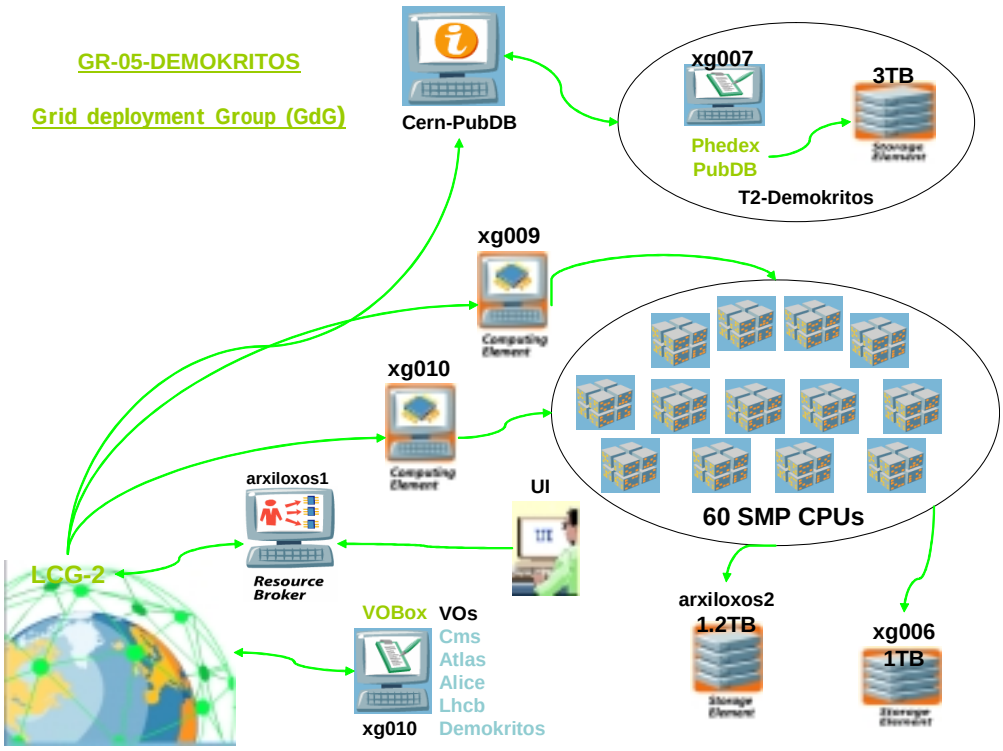
A. GR-05-DEMOKRITOS Grid Site

Domain:
inp.demokritos.gr

Monitoring:
http://goc.grid.sinica.edu.tw/gstat/GR-05-DEMOKRITOS

Accounting:
http://goc.grid-support.ac.uk/gridsite/accounting/tree/roc_view.php?ExecutingSite=GR-05-DEMOKRITOS

Web Site:
http://grid.demokritos.gr



In the framework of the analysis of CMS data, as well as the production of simulated events to be used in current and future analyses, the Institute of Nuclear Physics is developing a PC-Linux computing farm with high capacity and fast response. At the present time, the farm is still at an early stage of development and under continuous upgrade and expansion. At the time of writing this guide, the available infrastructure consists of 2 Computing Elements, two fileservers (2.5TB disk space overall) and 34 working nodes (22 PentiumIV-120Gb HD, 6 PentiumIII and 6 Xeon-80Gb double processor) configured so that they form a Grid Site , which is part of the LCG Grid.

The GR-05-Demokritos Grid Site is a Phedex (Physics Experimental Data Export) node controlling and maintaining 3TB of data for CMS accessible from CERNs PubDB.

B. User guide

In what follows we have tried to record the series of steps which a user has to take in order to use the available infrastructure in order to perform various computational tasks, both in the context of a general user, as well as in the context of CMS data production and analysis. **I**

Important Notice: This document is under construction and may be incomplete. Please provide feedback to Katerina Zachariadou (zacharia@inp.demokritos.gr) on missing, inaccurate, or unclear information.

1. Get a digital certificate

1. Login to the INP Grid cluster *User Interface (UI*) *(ui.inp.demokritos.gr*).
2. Create your personal key and certificate request:

```
ui> grid-cert-request –int
```

(You will be asked to create a password for your grid proxy)
Then you will be asked to insert the following parameters:

```
Country name →  GR
Level 0 Organization→ HellasGrid
Level 0 Unit  -→inp.demokritos.gr
Name → your name
```

A directory will be created named `.globus`. Inside the `.globus` directory you will find the files:

`usercert.pem` (an empty file)
`usercert_request.pem`
`usercert.pem`

3. Transfer the file `usercert_request.pem` to your computer and send it by e-mail to `hellasgrid-ca@physics.auth.gr`
4. Ask Christos Markou to send a fax to HellasGrid-CA verifying that you are a member of the IPF Demokritos.
5. HellasGrid-CA will send you by e-mail your personal certificate (username@inp.demokritos.gr.pem
6. Tranfer the file `username@inp.demokritos.gr.pem` to the UI (ui.inp.demokritos.gr) and locate it in your `.globus` directory.
7. Go to `.globus` directory and rename it :

```
ui> mv username@inp.demokritos.gr.pem usercert.pem
```

8. In order to import your private key and certificate <u>in your browser</u> you must create a `pkcs12` bundle. This can be achieved by issuing the command:

```
ui@username/.globus> openssl pkcs12-export –in
~/.globus/usercert.pem –inkey ~/.globus/userkey.pem –name
“My Certificate” –out mycertificate.p12
```

After issuing the above command, you will be asked to enter the pem pass phrase. This is the pass phrase you entered during the initial process of creating the certificate

request. Next you will have to enter an export password for the pkcs12 bundle and you will have to use it during the import procedure.

9. Transfer the pkcs12 bundle to your computer

10. Use Internet Explorer (recommended) and follow the instructions:

Step 1 (Importing HellasGrid CA Certificate to the browser)

11. Open `http://pki.physics.auth.gr/hellasgrid-ca/CA/` in your browser
12. Click on (`Click here to import HellasGrid CA certificate directly to your browser`).
13. A Window will pop up. Click on the button `open`.
14. Select the option `Install Certificate`. Click on `Next` two times and the `Finish`.
15. Answer **YES** to the question "`Do you want to ADD the following certificate to the Root Store?`"

You have successfully imported the HellasGrid CA Certificate into Internet Explorer

Step 2 (Import your pkcs12 bundle to the browser)

16. Open an Internet Explorer Window
17. Go to `File->Open`; Browse to the pkcs12 bundle you have transferred to the computer and open the file.
18. A new window will popup. Press `Next` to the following two screens.
19. I n the 3rd screen you will be asked to enter your password. This is the export password you entered previously.
20. Enable "`Enable strong private key protection. You will be prompted every time the private key is used by an application, if you enable this option.`"

DO NOT enable "`Mark this key as exportable. This will allow you to backup or transport your keys at a later time.`"

21. Select `Next` in the following 2 screens and then `Finish`.
22. Select `OK` to the following window.

Congratulations you have imported your certificate and private key to the browser.

Go to Microsoft Outlook

23. Go to `Tools`→`Options`→ →`Security` and press `Setup Secure e-mail.` Select `OK`. Go back to `Security` and select the Options
    a) `add digital signatures to outgoing messages`

b) `send clear text signed message`

Select `Apply` and `OK`

You are ready to send a digitally signed e-mail to hellasgrid-ca@grid.auth.gr stating that you accept your certificate and that you adhere to theHellasGrid Certification Policy:

To whom it may concern,

With this email I state that

1. **I, ########, accept my x509v3 digital certificate with**
   DN: /C=GR/O=HellasGrid/OU=inp.demokritos.gr/CN=#######
   Serial Number: 0x011C
   signed by /C=GR/O=HellasGrid/CN=HellasGrid CA

2. **I adhere the HellasGrid CA policy and usage rules found at:**
   http://pki.physics.auth.gr/hellasgrid-ca/CPS/HellasGrid-CPS.pdf
   (O.I.D. 1.3.6.1.4.1.13089.2.1.10.1.4)

2. Registering with LCG-2

Before a user can use the LCG-2 service, registration of some personal data with the LCG registration server (hosted at CERN) plus some additional steps are required. For detailed information please visit the following URL:

http://lcg-registrar.cern.ch/

To actually register oneself to the LCG-2 service, it is necessary to use a WWW browser with the user certificate installed for the request to be properly authenticated.

2.1 Virtual Organizations

A second requirement for the user is to belong to a *Virtual Organization (VO)*. A VO is an entity, which corresponds typically to a particular organization or group of people in the real world. The membership of a VO grants specific privileges to the user. For example, a user belonging to the *ATLAS* VO will be able to read the ATLAS files or to exploit resources reserved to the ATLAS collaboration.

Entering the VO of an experiment usually requires being a member of the collaboration; the user must comply with the rules of the VO relevant to him/her to gain membership. Of course, it is also possible to be expelled from a VO when the user fails to comply with these rules.

It is not possible to access the LCG-2 Grid without being member of any VO. Every user is required to select his/her VO when registering with LCG-2 and the supplied information is forwarded to the VO administration and resource prividers for validation before the registration process is completed. This forwarding is accomplished by the registration interface back-end automaticaly. It generates an email to the VO manager of the selected VO requesting addition of the user to the VO.

Currently, it is only possible to belong to one VO at a time. This is fine for most users. In the rare case that you need to belong to more than VO, then you should contact the LCG registration service (whose URL was given before).

A complete list of the VOs accepted by LCG-2 is available at the URL:
http://lcg-registrar.cern.ch/virtual_organization.html

3. Submit a job to the INP Grid cluster

3.1 Job without input /output data streams

Login to the INP Grid cluster *User Interface (UI*) *(ui.inp.demokritos.gr*).

Before interacting with the Grid make sure that the files inside your `.globus` directory have the following permissions:

```
-rw-r--r-- 1 democms production 2242 Oct 26 13:31 mycertificate.p12
-rw-r--r-- 1 democms  production 4861 Oct 26 13:28 usercert.pem
-rw-r--r-- 1 democms  production 1339 Oct 20 11:42 usercert_request.pem
-r-------- 1 democms  production  963 Oct 20 11:42 userkey.pem
```

You will need a proxy certificate. This proxy is your credential which authenticates you in every secure interaction on Grid and has a temporal life. In order to create a proxy:

```
[ui]/home/your_username>grid-proxy-init
```

If the command is successful, you will receive as output:

```
Your identity:
/C=GR/O=HellasGrid/OU=inp.demokritos.gr/CN=Katerina
Zachariadou
Enter GRID pass phrase for this identity: (you must
introduce your password)
Creating proxy
.......................................... Done
Your proxy is valid until: Thu Nov 10 02:06:53 2005
```

To submit a job, you must first create a **.jdl** file stating the executable image and the output files as in the following minimal **test jdl**:

```
Executable = "tjob.sh";
##Arguments     = "inputfile";
StdOutput = "tjob.out";
StdError = "tjob.err";
InputSandbox = {"./tjob.sh"};
###InputSandbox  = {"myexecutable","inputfile"};
OutputSandbox = {"tjob.out","tjob.err"};
Requirements = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms"
```

If your job has an input file named inputfile

Find a template job (`tjob.sh`) and a jdl file (`tjob.jdl`) in the directory `/tests/General` and copy them to your working directory. (Find also a minimal jdl file (`tfortran.jdl`) for submission of a simple (`test_fortran.exe`) fortran executable.

Submission is performed with the command:

```
[ui]/home/your_username> edg-job-submit --vo cms -o
job_output    tjob.jdl
```

The edg_jobId has been saved in job_output file

If job submission is successful, the middleware displays a message of the following type:

```
Selected Virtual Organisation name (from --vo option): cms
Connecting to host arxiloxos1.inp.demokritos.gr, port 7772
Logging to host arxiloxos1.inp.demokritos.gr, port 9002

=== edg-job-submit Success ================================
 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://arxiloxos1.inp.demokritos.gr:9000/Tr7Muof36saOHK67ynr
6Tw

 The edg_jobId has been saved in the following file:
 /home/democms/tests/job_output
============================================================
```

Monitor execution and recover output as follows:

```
[ui]/home/your_username> edg-job-status  JID
```

For the above example:

```
[ui]/home/your_username>edg-job-status  \
https://arxiloxos1.inp.demokritos.gr:9000/Tr7Muof36saOHK6
7ynr6Tw
```

```
BOOKKEEPING INFORMATION:

Status info for the Job :
https://arxiloxos1.inp.demokritos.gr:9000/Tr7Muof36s
aOHK67ynr6Tw
Current Status:     Done (Success)
Exit code:          0
Status Reason:    Job terminated successfully
Destination:
xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms
reached on:         Fri Nov  4 12:50:38 2005
**************************************************
```

Wait for **done** status flag. When the job status is Done then one can retrieve the results:

```
[ui]/home/your_username> edg-job-get-
```

eg.

```
[ui] /home/democms/tests > edg-job-get-output
https://arxiloxos1.inp.demokritos.gr:9000/Tr7Muof36saOHK6
7ynr6Tw

Retrieving files from host: arxiloxos1.inp.demokritos.gr
( for
https://arxiloxos1.inp.demokritos.gr:9000/Tr7Muof36saOHK6
7ynr6Tw )

**************************************************
               JOB GET OUTPUT OUTCOME
 Output sandbox files for the job:
 -
https://arxiloxos1.inp.demokritos.gr:9000/Tr7Muof36saOHK6
7ynr6Tw
 have been successfully retrieved and stored in the
directory:
 /tmp/jobOutput/democms_Tr7Muof36saOHK67ynr6Tw
```

meaning, that the system has decided to put the output files into the directory **/tmp/jobOutput/democms_Tr7Muof36saOHK67ynr6Tw**

A final step could be to copy the files to your home directory:

```
[ui]/home/your_username>cp /tmp/jobOutput/
democms_Tr7Muof36saOHK67ynr6Tw   $HOME/Output_Directory
```

If you want to delete the job:

```
[ui]/home/your_username > edg-job-cancel JID
```

3.2 Job with input /output data streams

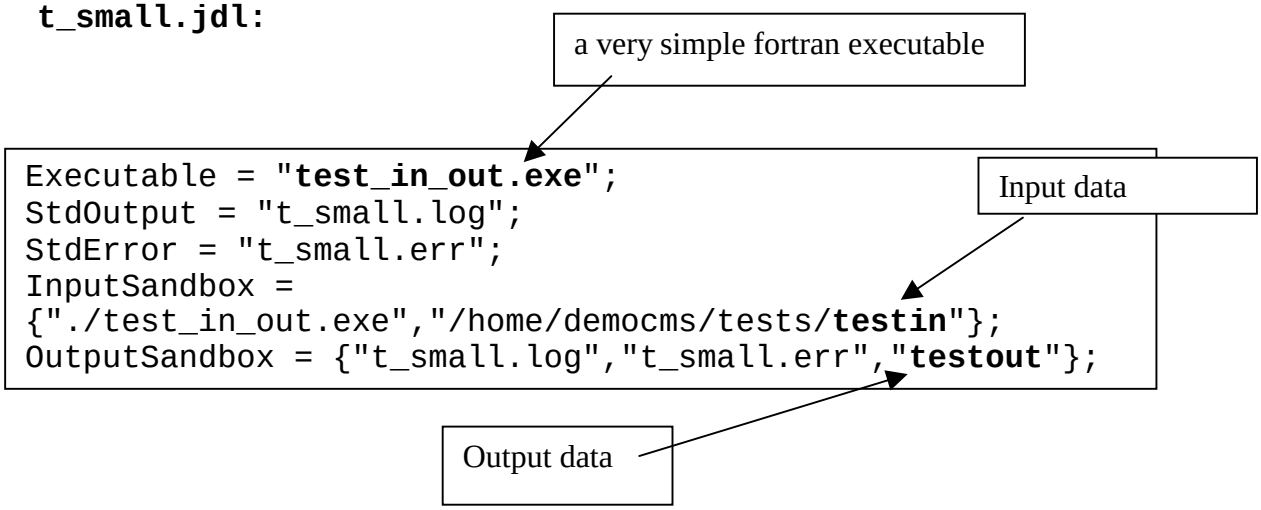
Important notice: The input and output sandboxes are intended for relatively small files (few megabytes) like scripts, standard input, and standard output streams. If you are using large input files or generating large output files, you should instead directly read from or write to a storage element. Abuse of the input and output sandboxes can fill the storage on the Resource Broker and cause the broker to crash.

3.1.a Small input/output data files

Use Input/Output Sandboxes in the .jdl file. Find a template jdl file (`t_small.jdl`) in the directory `/templates/General`. The above jdl executes a simple fortran executable (`test_in_out.exe`) that reads data from a file (`testin`) and writes data to an output file (`testout`)

1. Copy the files `t_small.jdl, testin, test_in_out.f, test_in_out.exe` to your working directory.

t_small.jdl:



2. Create a proxy if you haven't already created

```
[ui]> grid-proxy-init
```

3. Submit the job:

```
ui> edg-job-submit  -vo cms  -o tsmall_output t_small.jdl
```

4. Wait for done status flag:

Find JID in the `tsmall_output` file

```
ui> edg-job-status JID
```

5. Retrieve the output file (`testout`):

```
ui> edg-job-get-output JID
```

6. Copy the output (`testout`) to your working dir

3.1.b Large input/output data files

The Replica Catalog can be considered to be temporary storage that is available at each site on the grid store node (the Storage Element) of the site's grid gateway. All large data sets should be input and output to/from grid jobs via the Replica Catalog, rather than via the InputSandbox or OutputSandbox. Once registered with the Replica Catalog, files may be referred to via grid logical filenames (LFNs) that the user associates with the file at the time of registration.

Suppose that the input data file `testin` (see above) is a large data set. To register the file with the Replica Catalog use the command:

```
ui>lcg-cr --vo <VO name> file:<local_path>\ <file
name> -l <logical file name> -d <storage element>
```

eg:

```
ui>lcg-cr --vo cms
 file:<local_path>\testin -l testin \
-d arxiloxos2.inp.demokritos.gr
```

The above command returns the grid unique identifier (guid) of the grid file.

For example something like that: `guid:767da655-9d60-444d-bacf-3bec6c1d35ca`

The output file is located in the Storage Element (arxiloxos2@inp.demokritos.gr) `/storage/cms/generated/<current_date>.`

Files are accessed using the LFN name, therefore users are not involved in the directory location of files at any single SE.

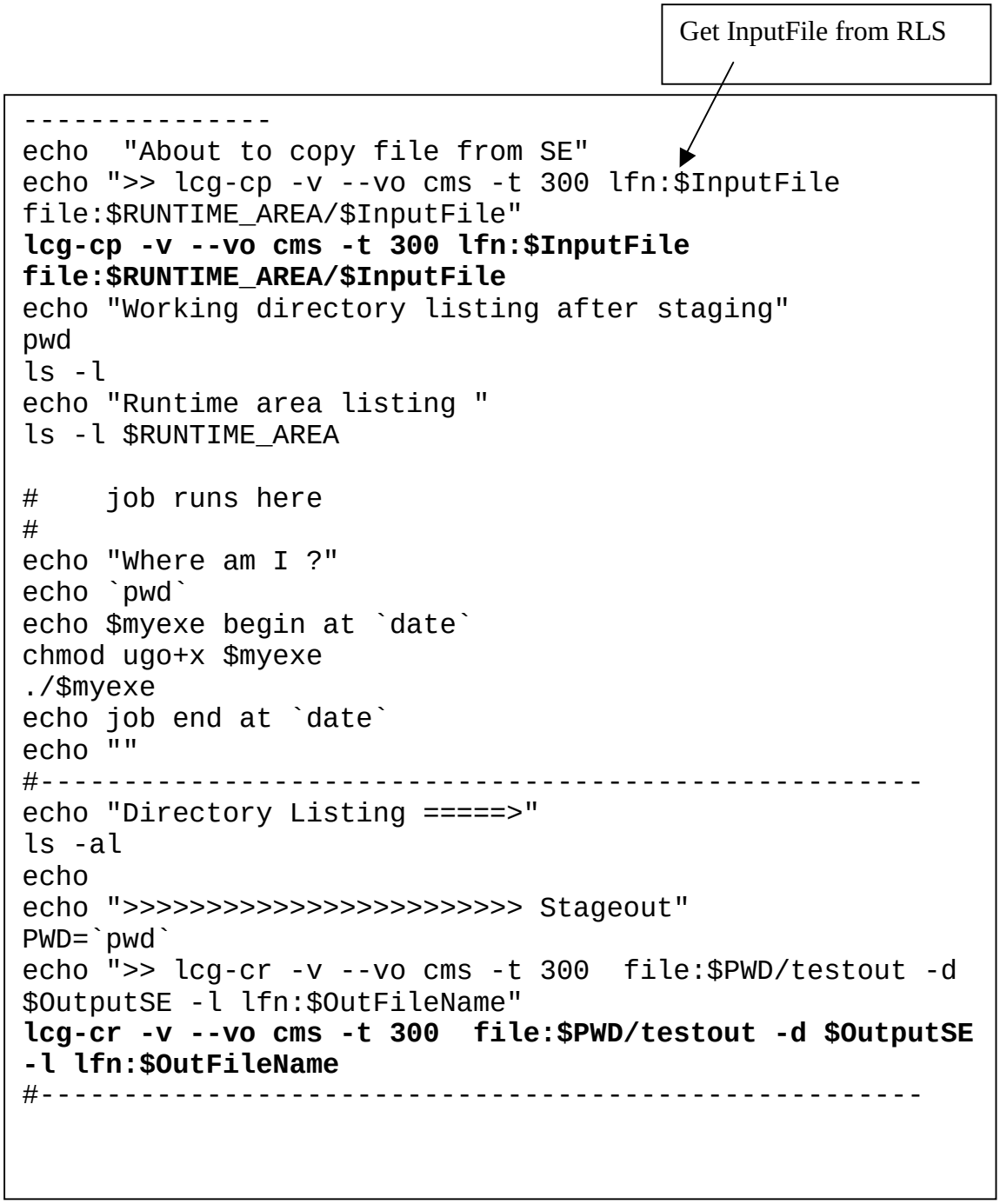
To input data from the Replica Catalog to the job and vice versa, use the following commands in a script (`eg. t_large2.sh`):

```
lcg-cp -v --vo cms -t 300 lfn:$InputFile
file:$RUNTIME AREA/$InputFile
```

```
lcg-cr -v --vo cms -t 300  file:$PWD/OutFileName -
d $OutputSE -l lfn:$OutFileName
```

t_large2.sh script:

```
#!/usr/local/bin/zsh
echo  "Job started at `date`"
w
finger
ls -l
export InputFile=testin
#
export OutputSE=arxiloxos2.inp.demokritos.gr
export OutputSE_DIR=/storage/cms
export SEPrefixDir=Katerina
export VO=cms
export RUNTIME_AREA=`pwd`
#
export myexe=test_in_out.exe
export  OutFileName=testout18
#*********************************************
ls -al
```




```
---------------
echo  "About to copy file from SE"
echo ">> lcg-cp -v --vo cms -t 300 lfn:$InputFile
file:$RUNTIME_AREA/$InputFile"
lcg-cp -v --vo cms -t 300 lfn:$InputFile
file:$RUNTIME_AREA/$InputFile
echo "Working directory listing after staging"
pwd
ls -l
echo "Runtime area listing "
ls -l $RUNTIME_AREA

#    job runs here
#
echo "Where am I ?"
echo `pwd`
echo $myexe begin at `date`
chmod ugo+x $myexe
./$myexe
echo job end at `date`
echo ""
#-------------------------------------------------------
echo "Directory Listing =====>"
ls -al
echo
echo ">>>>>>>>>>>>>>>>>>>>>>>> Stageout"
PWD=`pwd`
echo ">> lcg-cr -v --vo cms -t 300  file:$PWD/testout -d
$OutputSE -l lfn:$OutFileName"
lcg-cr -v --vo cms -t 300  file:$PWD/testout -d $OutputSE
-l lfn:$OutFileName
#-------------------------------------------------------
```

Put the name of the script (eg. `t_large2.sh`) and of the executable (eg. `test_in_out.exe`) in the .jdl file (eg. `t_large2.jdl`)

t_large2.jdl:

```
UserReq = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms";
#####################################################
Executable = "t_large2.sh";
InputSandbox =
{"/home/democms/tests/t_large2.sh","/home/democms/tests/te
st_in_out.exe"};
OutputSandbox = {"t_large2.out",
                 "t_large2.err"};
StdOutput = "t_large2.out";
StdError =  "t_large2.err";
Requirements = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms"
```

To run the above example copy the appropriate files (`test_in_out.exe`, `t_large2.sh`, `t_large2.jdl`) in your working directory, modify the paths and submit `the t_large2.jdl` to the grid:

```
[ui]> edg-job-submit --vo cms -o tlarge2 t_large2.jdl
```

You can retrieve the output file (eg with LFN: output17) from the RLS using the command:

```
[ui]>  lcg-cp --vo cms  lfn:testout17
file:$PWD/<local_filename>
```

Important notice: If you need to run the same job more than once be sure to register the output file with a different LFN or to delete from the RLS the LFN of the output before re-submit the job.

Useful commands:

To delete a registered file from the Replica Catalog:

```
ui>lcg-del --vo <VO name>  lfn:<logical file name> \
```

To find the guid of a registered file :

```
[ui]>lcg-lg –vo cms lfn:<lfn name>
```

4. Submit a CMKIN job to the INP Grid cluster (under construction)

Version CMKIN_4_4_0 is installed.

1. go to your working directory and create a directory eg: > mkdir cmkin
2. get your personal copy of the cmkin examples from the repository `/templates/CMKIN/CMKIN_4_4_0/examples`
3. Find in the subdirectory `/make_ntpl_jobs` the compilation script `kine_make_ntpl.com`. Modify it by adding in the beginning of the file the following lines:

```
set SCRATCH=<working_dir>/cmkin/examples/make_ntpl_jobs
set VERSIO=CMKIN_4_4_0
```

4. The compilation script is used as follows:

```
[ui]>kine_make_ntpl.com <generator>
```

where the first parameter can have one of the following values: *pythia*, *herwig*, *isajet*, *simple*, *single*, *double*, *simplemulti*, *cosmic*, *comphep*, *alpgen*, *madgraph*, *phase*, *toprex* or *stage*n. The optional second parameter *lhapdf* is given when the user wants to choose 'Les Houches Accord' parton density functions. Presently *lhapdf* can be used with *pythia*, *toprex* and *stage*n.
eg if you want to use the pythia generator:

```
[ui]>kine_make_ntpl.com pythia
```

eg, running the above command you will get the executable `kine_make_ntpl_pyt6227.exe`.

5. Add the following line in the beginning of the file `kine_make_ntpl.run`

```
set
SCRATCH=<your_working_dir>/cmkin/examples/make_ntpl_jobs
```

6. Replace the line:

```
set
DATACARDS=/cms1/Releases/CMKIN/CMKIN_4_4_0/examples/……
.
```

with the line:

```
set
DATACARDS=/storage/YOUR_USERNAME/cmkin/examples/make_ntpl_jobs/datacards/  \$DATACARDFILE
```

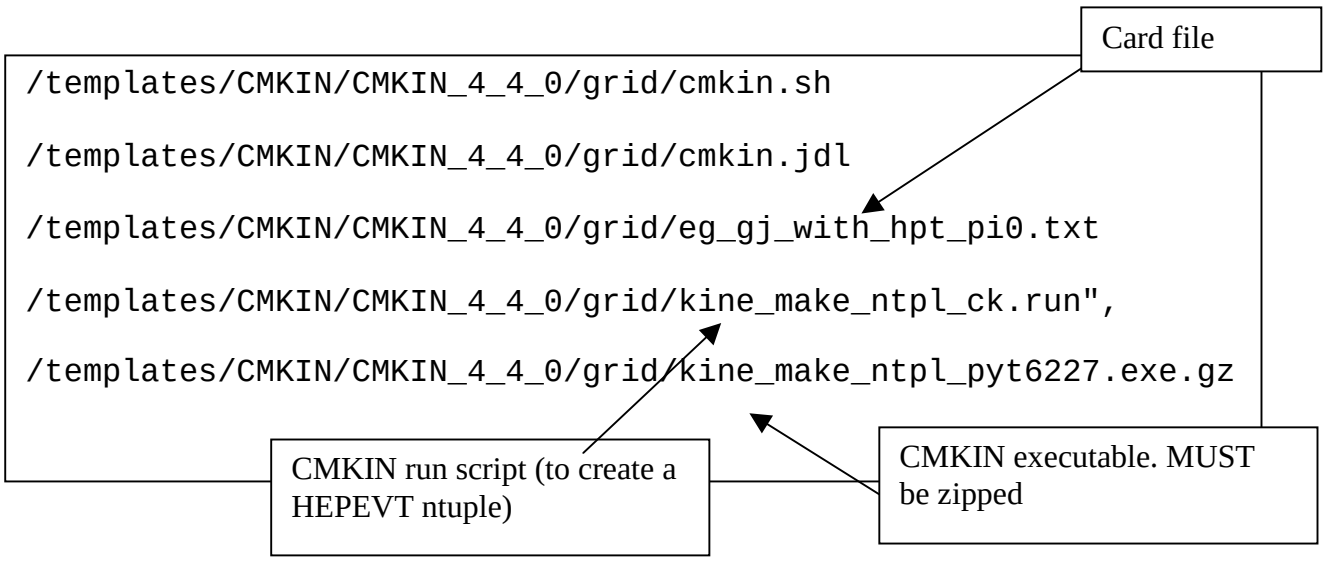
7. In case you want to run the cmkin executable interactively :

```
[ui]> kine_make_ntpl.run <generator>
```

eg:

```
[ui]> kine_make_ntpl.run pythia
```

In order to submit the cmkin executable to the grid, create a cmkin.jdl and a cmkin.sh file (find template files in the directory: /templates/CMKIN/CMKIN_4_4_0/grid) :



cmkin.sh:

```
#!/bin/csh
setenv  OutputNtuple  eg_gj_with_hpt_pi0_pt20-30.ntpl
setenv OutLFN   test_cmkin102.ntpl
#
setenv OutputSE arxiloxos2.inp.demokritos.gr
setenv  OutputSE_DIR /storage/cms
setenv VO cms
setenv SEPrefixDir Katerina
set PWD=`pwd`
echo $PWD
#*************************************************
if ( $VO_CMS_SW_DIR == ' ' ) then
   echo VO_CMS_SW_DIR not set
   echo Setting it manually to /opt/exp_software/cms
   setenv VO_CMS_SW_DIR /opt/exp_software/cms
else
   echo VO_CMS_SW_DIR is already set
   echo VO_CMS_SW_DIR=$VO_CMS_SW_DIR
endif

if ( -f $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.csh ) then
   echo Try to set the CMS Environmet by running the following script
   ls -l  $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.csh
  source $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.csh
   else
   echo CMS Environment could NOT be set ..... Exiting !!!!
  exit 10
endif
#-----------------------------------------------------------------------
gunzip kine_make_ntpl_pyt6227.exe.gz
chmod ugo+x kine_make_ntpl_pyt6227.exe
chmod ugo+x kine_make_ntpl_ck.run
 $PWD/kine_make_ntpl_ck.run pythia
ls -al
pwd
set PWD=`pwd`
echo  lcg-cr -v --vo $VO -t 300  file:$PWD/$OutputNtuple -d $OutputSE
-l lfn:/grid/cms/$OutLFN
lcg-cr -v --vo $VO -t 300  file:$PWD/$OutputNtuple -d $OutputSE -l
lfn:/grid/cms/$OutLFN

lcg-cp --vo cms lfn:/grid/cms/$OutLFN
file:/home/democms/tests/CMKIN/$OutLFN
```

cmkin.jdl:

```
#####################################################
# User defined part of the jdl: /data/test/test_cmkin.jdl
#####################################################
UserReq = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms";
#####################################################
Executable = "cmkin.sh";
InputSandbox = {"/home/democms/tests/CMKIN/cmkin.sh",

"/home/democms/tests/CMKIN/eg_gj_with_hpt_pi0.txt",
               "/home/democms/tests/CMKIN/kine_make_ntpl_ck.run",

"/home/democms/tests/CMKIN/kine_make_ntpl_pyt6227.exe.gz"};
OutputSandbox = {"cmkin_job.out",
                 "cmkin_job.err"};
StdOutput = "cmkin_job.out";
StdError = "cmkin_job.err";
RetryCount = 3;
CPUTime = other.GlueCEPolicyMaxCPUTime>1440;
Memory = other.GlueHostMainMemoryRAMSize>500;

Requirements = Member("VO-cms-
ORCA_8_7_3",other.GlueHostApplicationSoftwareRunTimeEnvironment)
&& CPUTime
&& Memory && UserReq;
```

CMKIN datacard file

To copy the CMKIN output to your local area use the command:

```
[ui] > lcg-cp --vo cms lfn:/grid/cms/$OutfileName   file:
/your_path/<file_to_create>
```

For the above example:

```
[ui] > lcg-cp  --vo cms lfn:/grid/cms/ test_cmkin102.ntpl
file: /your path/<file to create>
```

5. Submit a OSCAR job to the INP Grid cluster

You can find template files for submitting OSCAR jobs to the INP grid cluster in the directory:

`[ui]/templates/OSCAR`

1. Copy the template files into your working directory

```
[ui] >/templates/OSCAR > ll
total 24
-rw-r--r--   1 root    root   899 Dec 20 14:19 oscar_new2.jdl
-rwxr-xr-x   1 root    root     3731 Dec 20 14:16 oscar_new2.sh
-rw-r--r--   1 root    root    3941 Dec 20 14:16 oscarrc
```

2. Modify.jdl `oscar_job2.jdl` (optional):

```
Executable="oscar_new2.sh";
StdOutput= "oscar_new2.out";
StdError= "oscar_new2.err";
#
InputSandbox = {"oscar_new2.sh", "oscar_new2.jdl",
"oscarrc"};
OutputSandbox= { "oscar_new2.out", "oscar_new2.err"};
#
RetryCount = 0;

Requirements = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-
lcgpbs-cms"
```

2. **oscar_new2.sh**: (The script sets up the environment, runs OSCAR and saves the output to the SE:) Set the name of your input ntuple, and the LFN for the OSCAR output file (tar file)

```
#! /bin/csh -f
echo "--> Running grid job wrapper"
# ------------------------------------------------------
# modify the following lines:

setenv INPUT_NTPL kat_su05_TZt_bjj_1.ntpl
setenv OSCAR_OUTPUT_LFN myRECO639.output.tar
#
echo "----------> Input variables"
echo "--> LFN for Input Ntuple:  $INPUT_NTPL"
echo "--> LFN for OSCAR output tar file:  $OSCAR_OUTPUT_LFN"
#------------------------------------------------------------
# -- The Basics
echo "--> Environment"
date
hostname
cat /proc/cpuinfo
uname -a
echo $VO_CMS_SW_DIR
ls -l $VO_CMS_SW_DIR
source $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.csh
pwd
echo "--> End of env testing"

# ------------------------------------------------
# -- Setup environment
mkdir myRECO
cp oscarrc myRECO
ls -l

echo "--> locate lcg-cp"
locate lcg-cp
echo "--> retrieve CMKIN ntuple from SE"
lcg-cp --vo cms lfn:/grid/cms/$INPUT_NTPL
file:$PWD/myRECO/$INPUT_NTPL
ls -l
# -----
# -- Setup OSCAR
echo "--> Setup OSCAR"
date
scram list
scram project OSCAR OSCAR_3_6_5
setenv OSCARDIR   OSCAR_3_6_5

cd $OSCARDIR
eval `scram runtime -csh`
```

```
# -- Run OSCAR

cd ../myRECO
setenv OSCARSTEER ./oscarrc

echo "--> printenv "
printenv
echo "==> which  oscar: "
which oscar
echo "--> Run OSCAR"
date
oscar -c $OSCARSTEER >& $OSCARSTEER.log
echo "--> Done with OSCAR"
date
cd ../
tar cvf myRECO.output.tar myRECO
# ------------------------------------------------------------
echo "--> Saving output to SE: "
echo "    lcg-cr --vo cms -d arxiloxos2.inp.demokritos.gr -
llfn:/grid/cms/$OSCAR_OUTPUT_LFN  file:$PWD/myRECO.output.tar"
setenv GUID `lcg-cr --vo cms -d arxiloxos2.inp.demokritos.gr -l
lfn:/grid/cms/$OSCAR_OUTPUT_LFN  file:$PWD/myRECO.output.tar`
echo "--> GUID:  $GUID"
date
echo "--> end of grid job wrapper"
```

4. In **`oscarrc`** modify the following lines

```
# -- Modify the following lines
NumberOfEventsToBeProcessed  = 2
FilePath                     = ./
EventNtplReader:NtplFileName =
./kat_su05_TZt_bjj_1.ntpl
OutputFileCatalogURL         = file:./pfc-myRECO.xml
OutputDataSet                =
/System/SimHits365/myRECO
#
VCalShowerLibrary:FileName = vcal5x5.rz
VCalShowerLibrary:FilePath =
.:${CMS_PATH}/cmsim/cmdb/vcal
# ------------------------------------------
```

3. Copy and register your input ntuples to the Storage Element (arxiloxos2@inp.demokritos.gr) and register the file in the Replica Catalog :

```
lcg-cr  --vo cms   file:/my_workdir/myntuple.ntp -d
arxiloxos2@inp.demokritos.gr -l
lfn:/grid/cms/Alogical_File_Name
```

For the above example:

```
lcg-cr  --vo cms
file:/my_workdir/kat_su05_TZt_bjj_1.ntpl  -d
arxiloxos2@inp.demokritos.gr  -l
lfn:/grid/cms/kat_su05_TZt_bjj_1.ntpl
```

4. Submit the OSCAR job:

```
[ui] >  edg-job-submit   --vo cms –o oscar_output
oscar_new2.jdl
```

To check the output use the command:

```
[ui] > lcg-lr  --vo cms –l  lfn:/grid/cms/$OutfileName
```

Where $OutfileName is defined in oscar_new2.sh

To get the output file from the Replica Catalog use the command:

```
[ui] > lcg-cp –vo cms lfn:/grid/cms/$OutfileName   file:
/your_path/<file_to_create>
```

6 Submit a ORCA job to the INP Grid cluster

Find template files for submitting ORCA jobs to the INP grid cluster in the directory:

`[ui]/templates/DIGI`

1. Copy the template files into your working directory

```
[ui] /templates/DIGI > ll
-rw-r--r--   1 root   root     318 Mar  7 16:19 orca_new2.jdl
-rwxr--r--   1 root   root     1754 Mar  7 16:19 orca_new2.sh
-rw-r--r--   1 root    root    4068 Mar  7 16:20 orcarc
```

2. Modify the **`orca_new2.jdl`**:

```
#
Executable="orca_new2.sh";
StdOutput= "orca_new2.out";
StdError= "orca_new2.err";
#
InputSandbox = {"orca_new2.sh", "orca_new2.jdl", "orcarc"};
OutputSandbox= { "orca_new2.out", "orca_new2.err"};
#
RetryCount = 0;

Requirements = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms"
```

3. Modify the **orca_new2.sh**:

```
#!/usr/local/bin/zsh
#-------------modify the following lines
export OSCAR_XML=pfc-myRECO.xml
export OSCAR_OUTPUT_LFN=myRECO654.output.tar
#
export ORCA_OUTPUT_LFN=myRECO-1231.output.tar
echo "----------> Input variables"
echo "--> LFN for OSCAR output XML:  $OSCAR_XML"
echo "--> LFN for OSCAR output tar file:
$OSCAR_OUTPUT_LFN"
#
echo "--> LFN for ORCA output tar file:
$ORCA_OUTPUT_LFN"
#-----------------------------------------------------
# -- The Basics
echo "--> Environment"
date
hostname
cat /proc/cpuinfo
uname -a
echo $VO_CMS_SW_DIR
source $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.sh
pwd
echo "--> End of env testing"
# -- Setup environment
echo "--> retrieve oscar output from SE"
lcg-cp --vo cms lfn:/grid/cms/$OSCAR_OUTPUT_LFN
file:$PWD/$OSCAR_OUTPUT_LFN
pwd
ls –l
tar xvf $OSCAR_OUTPUT_LFN
cp orcarc  myRECO
# --------------- Setup ORCA
echo "--> Setup ORCA"
date
scram project ORCA ORCA_8_7_3
export ORCADIR=ORCA_8_7_3
cd ${ORCADIR}
eval `scram runtime -sh`
#-----------------------------------------
#modify the OSCAR xml paths
cd ../myRECO
echo "1st FClistPFN"
FClistPFN -u file:$OSCAR_XML -q "DataType='META'"
echo " ****"
echo "2nd  FClistPFN"
FClistPFN -u file:$OSCAR_XML -q "DataType='EVD' and
dataset='myRECO'"
echo "***"
```

```
echo "Starting first FCrename"
for PFN in `FClistPFN -u file:$OSCAR_XML -q "DataType='META'"`
; do
  echo $PFN
 FCrenamePFN  -u file:$OSCAR_XML -p $PFN -n ./`basename $PFN`
   if [ $? != 0 ] ; then
      echo "Problems renaming PFNs."
      exit 1021
   fi
done
echo "Starting second FCrename"
#
for PFN in `FClistPFN -u file:$OSCAR_XML -q "DataType='EVD' and
dataset='myRECO'"`  ; do
  echo $PFN
FCrenamePFN  -u file:$OSCAR_XML -p $PFN -n ./`basename $PFN`
   if [ $? != 0 ] ; then
      echo "Problems renaming PFNs."
      exit 1022
   fi
done
# end of OSCAR xml paths modification
"# -------------------- Run ORCA digi----------------------
export ORCASTEER=./orcarc
echo "==> which  writeAllDigis: "
which writeAllDigis
echo "--> Run ORCA digi"
date
writeAllDigis -c $ORCASTEER >& $ORCASTEER.log
echo "--> Done with ORCA digi"
date
# -- Wrap up output
pwd
cd ../
tar cvf myRECO.output.tar myRECO
# -----------------------------------------------------
echo "--> Saving output to SE: "
echo "    lcg-cr --vo cms -d arxiloxos2.inp.demokritos.gr -l
lfn:/grid/cms/$ORCA_OUTPUT_LFN file:$PWD/myRECO.output.tar"
export GUID=`lcg-cr --vo cms -d arxiloxos2.inp.demokritos.gr -l
lfn:/grid/cms/$ORCA_OUTPUT_LFN  file:$PWD/myRECO.output.tar`
echo "--> GUID:  $GUID"
date
echo "--> end of grid job wrapper"
```

4 modify the following lines in **orcarc** (if necessary):

```
# -- Modify the following lines
MaxEvents            = 2
FilePath             = ./
InputFileCatalogURL  = file:./pfc-myRECO.xml
InputCollections     = /System/SimHits365/myRECO
OutputFileCatalogURL = file:./pfc-myRECO.xml
OutputDataSet        = /System/Digis873/myDIGI
```

4. submit the ORCA job to the grid:

```
[ui] > edg-job-submit --vo cms –o orca_output
orca_new2.jdl
```

5. retrieve the output (tar file):

```
lcg-cp –vo cms lfn:/grid/cms/$ORCA_OUTPUT_LFN \
 file: /your_path/<file_to_create>.tar
```

7 Submit a DST job to the INP Grid cluster

Find template files for submitting DST jobs to the INP grid cluster in the directory:

`[ui]/templates/DST`

1. Copy the template files into your working directory

```
[ui] /templates/DST > ll
-total 16
-rw-r--r--    1 root     root         265 Mar 20 16:05 dst.jdl
-rw-r--r--    1 root     root         5142 Mar 20 16:06 dstrc
-rwxr--r--    1 root     root         2643 Mar 20 16:05 dst.sh
```

2. Modify the **`dst.jdl`**:

```
[ui] /templates/DST > more dst.jdl
#
Executable="dst.sh";
StdOutput= "dst.out";
StdError= "dst.err";
#
InputSandbox = {"dst.sh", "dst.jdl", "dstrc"};
OutputSandbox= { "dst.out", "dst.err"};
#
RetryCount = 0;

Requirements = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms"
```

3. Modify the **dst.sh**:

```
[ui] /templates/DST > more dst.sh
#!/usr/local/bin/zsh
#-----------modify the following lines
export ORCA_XML=pfc-myRECO.xml
export ORCA_OUTPUT_LFN=myRECO-1231.output.tar
#
export DST_OUTPUT_LFN=myRECO-1111.output.tar
#
echo "----------> Input variables"
echo "--> LFN for ORCA output XML:  $ORCA_XML"
echo "--> LFN for ORCA output tar file:  $ORCA_OUTPUT_LFN"
#
echo "--> LFN for DST output tar file:  $DST_OUTPUT_LFN"
# ------------------------------------------------
# -- The Basics
echo "--> Environment"
date
hostname
cat /proc/cpuinfo
uname -a
echo $VO_CMS_SW_DIR
source $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.sh
pwd
echo "--> End of env testing"
# ------------------------------------------------
# -- Setup environment
echo "--> retrieve DIGI output from SE"
lcg-cp --vo cms lfn:/grid/cms/$ORCA_OUTPUT_LFN
file:$PWD/$ORCA_OUTPUT_LFN
tar xvf $ORCA_OUTPUT_LFN
cp dstrc   myRECO
ls -l
#------------- -- Setup ORCA----------------------
echo "--> Setup ORCA"
date
scram project ORCA ORCA_8_7_3
export ORCADIR=ORCA_8_7_3
cd ${ORCADIR}
eval `scram runtime -sh`
#-------------------------------------------------
# modify the ORCA xml paths
#-------------------------------------------------------------
cd ../myRECO
ls
echo "  do  FClistPFN"
FClistPFN -u file:$ORCA_XML -q "owner='Digis873'"
echo " ****"
```

```
echo "Starting FCrename"

for PFN in `FClistPFN -u file:$ORCA_XML -q
"owner='Digis873'"`  ; \
do
  echo $PFN
 FCrenamePFN  -u file:$ORCA_XML -p $PFN -n ./`basename
$PFN`
   if [ $? != 0 ] ; then
      echo "Problems renaming PFNs."
      exit 1021
   fi
done
#
# end of xml paths modification
#
# ----------- Run ORCA dst--------------
export ORCASTEER=./dstrc
echo "==> which  writeAllDigis: "
which writeDST
echo "--> Run ORCA dst"
date
writeDST -c $ORCASTEER >& $ORCASTEER.log
echo "--> Done with ORCA dst"
date
# -- Wrap up output
cd ../
tar cvf myRECO.output.tar myRECO
# ------------------------------------------------
echo "--> Saving output to SE: "
echo "    lcg-cr --vo cms -d
arxiloxos2.inp.demokritos.gr -l
lfn:/grid/cms/$DST_OUTPUT_LFN
file:$PWD/myRECO.output.tar"
export GUID=`lcg-cr --vo cms -d
arxiloxos2.inp.demokritos.gr -l
lfn:/grid/cms/$DST_OUTPUT_LFN
file:$PWD/myRECO.output.tar`
echo "--> GUID:  $GUID"
date
echo "--> end of grid job wrapper"
```

4. Modify the following lines in **`dstrc`**

```
# -- Modify the following lines
MaxEvents            = 2
FilePath             = ./
InputFileCatalogURL  = @{
                         file:./pfc-myRECO.xml
                       }@
InputCollections     = /System/Digis873/myDIGI
OutputFileCatalogURL = file:./pfc-myRECO.xml
OutputDataSet        = /System/DST873/myDST
# ------------------------------------------------
```

5. Submit the DST job to the grid

```
[ui] >  edg-job-submit   --vo cms –o dst_output   dst.jdl
```

6. retrieve the output (tar file):

```
[ui] > lcg-cp –vo cms lfn:/grid/cms/$DST_OUTPUT_LFN \
 file: /your_path/<file_to_create>.tar
```

8. Submit an OSCAR+DIGI job to the INP Grid cluster

In case you want to run OSCAR and then DIGI jobs you can use the template files found in the directory:

`[ui]/templates/OSCAR_DIGI`

1. Copy the template files into your working directory

```
-rw-r--r--    1 root     root      4068 Feb 27 14:24 orcarc
-rw-r--r--    1 root     root     4428 Feb 27 14:24 oscarrc
-rw-r--r--    1 root     root     284 Feb 27 14:26 reco.jdl
-rwxr--r--    1 root     root     2322 Feb 27 14:26 reco.sh
```

2. Modify the **`oscarrc:`**

```
# -- Modify the following lines
NumberOfEventsToBeProcessed  = 5
FilePath                     = ./
EventNtplReader:NtplFileName = ./kat_su05_TZt_bjj_1.ntpl
OutputFileCatalogURL         = file:./pfc-myRECO.xml
OutputDataSet                = /System/SimHits365/myRECO
#
VCalShowerLibrary:FileName = vcal5x5.rz
VCalShowerLibrary:FilePath = .:${CMS_PATH}/cmsim/cmdb/vcal
# ------------------------------------------------
```

3. Modify the **`orcarc:`**

```
# -- Modify the following lines
MaxEvents            = 5
FilePath             = ./
InputFileCatalogURL  = file:./pfc-myRECO.xml
InputCollections     = /System/SimHits365/myRECO
OutputFileCatalogURL = file:./pfc-myRECO.xml
OutputDataSet        = /System/Digis873/myRECO
# ------------------------------------------------
```

4. Copy and register your input ntuple to the Storage Element (arxiloxos2@inp.demokritos.gr) and in the Replica Catalog :

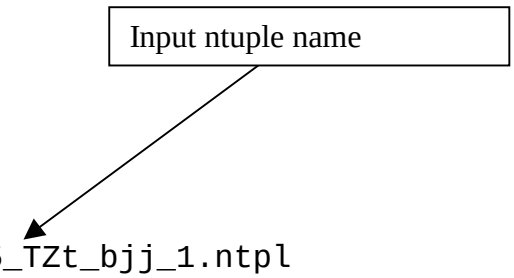
```
lcg-cr  --vo cms   file:/my_workdir/myntuple.ntp -d
arxiloxos2@inp.demokritos.gr -l
lfn:/grid/cms/Alogical_File_Name
```

For the above example:

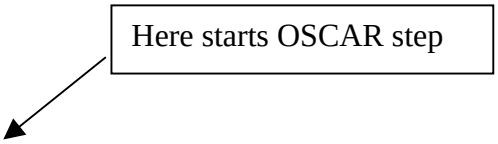
```
lcg-cr  --vo cms
file:/my_workdir/kat_su05_TZt_bjj_1.ntpl  -d
arxiloxos2@inp.demokritos.gr  -l
lfn:/grid/cms/kat_su05_TZt_bjj_1.ntpl
```

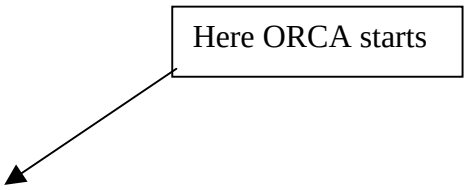
5. Modify the **`reco.sh`** for your input ntuple and your output files:

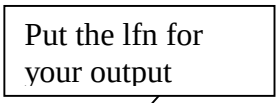
```
[ui] /templates/OSCAR_DIGI > more reco.sh
#! /bin/csh -f


echo "--> Running grid job wrapper"

# ------------------------------------------------------------
# -- The Basics
echo "--> Environment"
date
hostname
cat /proc/cpuinfo
uname -a
echo $VO_CMS_SW_DIR
ls -l $VO_CMS_SW_DIR
#####source $VO_CMS_SW_DIR/cmsset_default.csh
source $VO_CMS_SW_DIR/cmsset_slc3_ia32_gcc323.csh
pwd

echo "--> End of env testing"
# ------------------------------------------------------------
# -- Setup environment
mkdir myRECO
cp oscarrc myRECO
cp orcarc  myRECO
ls -l

echo "--> locate lcg-cp"
locate lcg-cp
echo "--> retrieve ntuple from SE"
lcg-cp --vo cms lfn:/grid/cms/kat_su05_TZt_bjj_1.ntpl
file:$PWD/myRECO/kat_su05_TZt_bjj_1.ntpl
```



```
ls -l
# -- Setup OSCAR
echo "--> Setup OSCAR"
date
scram list
scram project OSCAR OSCAR_3_6_5
setenv OSCARDIR   OSCAR_3_6_5

cd $OSCARDIR
eval `scram runtime -csh`
# -- Run OSCAR
##not needed? source src/Workspace/writeTrigger.csh
cd ../myRECO
setenv OSCARSTEER ./oscarrc
echo "--> printenv "
printenv
echo "==> which  oscar: "
which oscar
echo "--> Run OSCAR"
date
oscar -c $OSCARSTEER >& $OSCARSTEER.log
echo "--> Done with OSCAR"
date
# -- Move back in relative mode
cd ..
# -- Setup ORCA
echo "--> Setup ORCA"
date
scram project ORCA ORCA_8_7_3
setenv ORCADIR    ORCA_8_7_3
cd ${ORCADIR}
eval `scram runtime -csh`
# -- Run ORCA digi
cd ../myRECO
setenv ORCASTEER  ./orcarc
echo "--> printenv "
printenv
echo "==> which  writeAllDigis: "
which writeAllDigis
echo "--> Run ORCA digi"
date
writeAllDigis -c $ORCASTEER >& $ORCASTEER.log
echo "--> Done with ORCA digi"
# -- Wrap up output
cd ../
tar cvf myRECO.output.tar myRECO
echo "--> Saving output to SE: "
echo "    lcg-cr --vo cms -d arxiloxos2.inp.demokritos.gr -l
lfn:/grid/cms/myRECO.output.tar file:$PWD/myRECO.output.tar"
setenv GUID `lcg-cr --vo cms -d arxiloxos2.inp.demokritos.gr -
l lfn:/grid/cms/myRECO.output.tar file:$PWD/myRECO.output.tar`
echo "--> GUID:  $GUID"
date
```







6. Modify (if you want) the **reco.jdl:**

```
[ui] /templates/OSCAR_DIGI > more reco.jdl
#
Executable="reco.sh";
StdOutput= "reco.out";
StdError= "reco.err";
#
InputSandbox = {"reco.sh", "reco.jdl", "oscarrc",
"orcarc"};
OutputSandbox= { "reco.out", "reco.err"};
#
RetryCount = 0;

Requirements = other.GlueCEUniqueID ==
"xg009.inp.demokritos.gr:2119/jobmanager-lcgpbs-cms"
```

7. Submit the job to the grid:

```
[ui]>  edg-job-submit  --vo cms –o reco_output  reco.jdl
```

8. Retrieve the output :

```
[ui] > lcg-cp –vo cms lfn:/grid/cms/
myRECO.output.tar   file: /your_path/<file_to_create>
```

9. Submit a OSCAR+DIGI+DST job to the INP Grid cluster

In case you want to run OSCAR and then DIGI and DST job you can use the template files found in the directory:

```
ui>/templates/OSCAR_DIGI_DST
```

1. Copy the template files into your working directory

```
-rwxr--r--    1 root     root  2535 Feb 27 15:45 reco_dst.sh
-rw-r--r--    1 root     root   320 Feb 27 15:45 reco_dst.jdl
-rw-r--r--    1 root    root    4068 Feb 27 15:45 orcarc
-rw-r--r--    1 root     roo    4428 Feb 27 15:46 oscarrc
-rw-r--r--    1 root     root         4211 Feb 27 15:46 dstrc
```

2. Follow the steps of the previous section. In this case you should modify additionally the **`dstrc`** file:

```
# -- Modify the following lines
MaxEvents             = 3
FilePath              = ./
InputFileCatalogURL   = @{
                          file:./pfc-myRECO.xml
                        }@
InputCollections      = /System/Digis873/myRECO
OutputFileCatalogURL = file:./pfc-myRECO.xml
OutputDataSet         = /System/DST873/myRECO
```

10. Useful Links

1. *LCG2 User Guide (Workload Management, Data Management, Information System, GLUE)* https://edms.cern.ch/file/454439//LCG-2-Userguide.pdf
2. *LCG2 User Guide Manual Series* https://egee-na4.ct.inf.it/documentation/LCG-2-Userguide.pdf
3. *JDL syntax* http:/server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0142-0_2.pdf
4. *ClassAds syntax* https://www/cs/wisc.edu/condor/classad
5. *JDL/ClassAds HOWTO (a* very *useful reference document)* http:/server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0102-0_2.-Document.pdf
6. *lcg-utils (Replica Catalog commands)* http://www.gridpp.ac.uk/deployment/users/datamanagement/howtolcg.html
7. *GFAL library (used by lcg-utils)* http://grid-deployment.web.cern.ch/grid-deployement/gis/GFAL/gfal.3.html