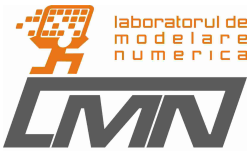
Universitatea "Politehnica" din Bucureşti

Facultatea de Inginerie Electrica –CIEAC-LMN

Marie Curie Doctoral School in EE and CS

Spl. Independenței 313, RO-060042, Bucureşti ROMANIA
http://www.lmn.pub.ro/ Tel/Fax: (40 21) 311 8004, (40 21) 316 95 71,
e-mail: lmn@lmn.pub.ro



REPORT TITLE: **Working with Unicore 6.x**

AUTHOR NAME: **Tauseef Jamal**

CONTENTS

1. What is UNICORE in General:

1.1 UNICORE

UNICORE (Uniform Interface to Computing Resources) offers a ready-to-run Grid system including client and server software. UNICORE makes distributed computing and data resources available in a seamless and secure way through intranets and internet.

UNICORE Quickstart package contains a complete UNICORE server installation, including demo certificates and an easy to use graphical installer. I can use the installation as a basis for easily creating your customized installations.

In order to run the UNICORE 6 server or client components, all we need Sun's Java Runtime Environment (JRE) 1.5 or higher. Since all components are platform independent, they can run under Linux, MAC and windows likewise.

UNICORE is a Grid Middleware, it submits jobs to already installed resource management systems/batch systems. It means that UNICORE can run without a resource management system/batch system, jobs are only forked then. But in production one normally wants a resource management system/batch system.

Each server component (Gateway, Registry, UNICORE/X and XUUDB) needs one X.509 private/public key pair signed by a CA (Certification Authority) as identity. Each user needs a signed X.509 private/public key pair which has to be loaded in the keystore of the client. Additionally, we need the certificates of the CAs. For testing purposes, we can use the same private/public key pair for the user and all the server components.

All network connections between the UNICORE components use client-authenticated SSL (Secure Sockets Layer), i.e. both sides of the connection check that they trust the other side. "Trust" means that the CA is checked.
If you use a single CA for all your certificates, the configuration is rather simple: Each server component needs to know the CA certificate, and additionally the CA has to be loaded in the client's keystore. If you use multiple CA's, consider how the UNICORE components work together: The client communicates with the Gateway, so the Gateway has to know the user's CA and the client has to know the CA of the Gateway. The Gateway also communicates with the Registry and UNICORE/X, so the Registry and UNICORE/X should know the Gateway's CA and vice versa. Additionally, the UNICORE/X communicates with the XUUDB, so both components need to know each other's CA certificates.

When adding a new user to theX UUDB, we need his signed certificate (public key). If we run the XUUDB in DN mode, the distinguished name (DN) of the user's public key will suffice.

we can generate a certificate request within the GPE client:
Click Settings -> Keystore Editor
to open up the keystore editor and select
Actions -> Generate Certification Request
The client will create a new private key, which is automatically stored in the keystore editor, and a certification request, which you are asked to save to disk. Send the certification request to a CA (Certification Authority).
You will get a signed certificate (public key) and a CA certificate in return. Store them on disk and click
Action -> Import
to import them into the keystore.

1.2 Grid Layout

Only the Gateway needs to be accessible from outside the firewall, using one port. The other components need not be accessible outside the firewall. It is possible to bypass the gateway for filetransfers, which will boost performance by a factor of 2, but will need a second open port to allow clients direct connections to the UNICORE/X server.

We can install UNICORE 6 on a single machine. However, it is recommended to use a dedicated machine for the gateway.

The best recommend way is three machines. One hosts the Gateway, one the UNICORE/X server and XUUDB. A third one runs the TSI, usually this will be the login node of our cluster.

If we use static initialisation, enter all VSites in the connections.properties file of the gateway
The XUUDB is accessed using web service calls, and is configured in the UNICORE server's main **configuration file** .If we run a batch system like Torque on our nodes. The VSite acts as "front end", so you can access your little cluster through UNICORE.

It's important to understand that UNICORE does not replace a local batch system. The Gateway and UNICORE/X should run as normal user. The TSI MUST run as root since it does setuid to the actual user, executes commands, calls the batch system, writes files, etc.

1.3 Server Components

Since Windows comes without hardly any applications that you can use for job submit, install for example Perl or Python. Then change the IDB (by default unicorex/conf/simpleidb) to recognise that application, here is an example for Python 2.4:

```
<idb:IDBApplication>
  <idb:ApplicationName>Python Script</idb:ApplicationName>
  <idb:ApplicationVersion>2.4.3</idb:ApplicationVersion>
  <jsdl:POSIXApplication xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix">
    <jsdl:Executable>C:\Python24\python.exe</jsdl:Executable>
    <jsdl:Argument>-d$DEBUG?</jsdl:Argument>
    <jsdl:Argument>-v$VERBOSE?</jsdl:Argument>
    <jsdl:Argument>$ARGUMENTS?</jsdl:Argument>
    <jsdl:Argument>$SOURCE?</jsdl:Argument>
  </jsdl:POSIXApplication>
</idb:IDBApplication>
```

It is necessary on windows that we start the server components as administrator.

Since Windows comes without hardly any applications that you can use for job submit, install for example Perl or Python. Then change the IDB (by default unicorex/conf/simpleidb) to recognise that application, here is an example for Python 2.4:

```
<idb:IDBApplication>
  <idb:ApplicationName>Python Script</idb:ApplicationName>
  <idb:ApplicationVersion>2.4.3</idb:ApplicationVersion>
  <jsdl:POSIXApplication xmlns:jsdl="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix">
    <jsdl:Executable>C:\Python24\python.exe</jsdl:Executable>
    <jsdl:Argument>-d$DEBUG?</jsdl:Argument>
    <jsdl:Argument>-v$VERBOSE?</jsdl:Argument>
    <jsdl:Argument>$ARGUMENTS?</jsdl:Argument>
    <jsdl:Argument>$SOURCE?</jsdl:Argument>
  </jsdl:POSIXApplication>
</idb:IDBApplication>
```

It is necessary on windows that you start the server components as administrator.

1.4 Clients Components

UNICORE shows the following job states of the client:

- STAGINGIN - the server is staging in data from remote sites into the job directory
- READY - job is ready to be started
- QUEUED - job is waiting in the batch queue
- RUNNING - job is running
- STAGINGOUT - execution has finished, and the server is staging out data to remote sites
- SUCCESSFUL - all finished, no errors occured
- FAILED - errors occured in the execution and/or data staging phases
- UNDEFINED - this state formally exists, but is not seen on clients

If we want to install UCC on windows, ucc doesn't like if it's installed in a path that does contain any spaces.
The config-directory must be created here: C:\Documents and Settings\<username>\.ucc. The explorer may refuse to create a directory that starts with a dot, in that case you can use mkdir from the command line.

5 UNICORE for testing purpose:

If you want to use UNICORE for testing purposes, there are three simple ways:

- Install the UNICORE server quickstart bundle plus the clients on your local machine. You need Sun Java 1.5. Although the UNICORE server components run under Windows, a Linux-like operation system is preferred since this document relays on applications like Bash.

- Use the UNICORE testgrid and install precon_gured clients, You need Sun Java 1.5. You can download a pre-con_gured client package from www.unicore.eu/testgrid.

- Use the UNICORE LiveCD, You need a i386-compatible computer that boots from CD. You can download the iso-image from www.unicore.eu/download/unicore6.

2. Server Components

2.1 Installation

Download the server bundle from www.unicore.eu/download and execute the installer by double-clicking on the jar-_le or entering in the command line:

java -jar unicore - quickstart -6.0.0. jar

Check the installation directory and ports used. You should not have any services already using these ports, however, on most machines the default ports should be free. If you are sure about the installation directory and ports, finish the installation.

2.2 Additional configuration

If the user submits a job with a specific application (e.g. a PovRay job or Bash script job), the server has to know where it can _nd the executable of this application. The paths to the application executables are configured in the IDB (Incarnation Data Base). Open the IDB of your newly installed server components, UNICORE_Quickstart/unicorex/conf/simpleidb, in a text editor and check the paths of the PovRay and Bash application. For now, you can ignore the other applications given in the IDB.

2.3 Start server components

Now we need to start the server components:
cd UNICORE_Quickstart
./ start .sh

3 Clients components Installation

UNICORE offers a set of different clients, each of which serving a different purpose. Currently the following clients are available:

- Grid Programming Environment (GPE) Application Client: prepare, submit and monitor single task jobs

- GPE Expert Client: prepare, submit and monitor multi-task jobs, edit workows
- GPE Filemanager: easy _le transfer and managment of _les on remote sites
- UNICORE Commandline Client (ucc): submit and monitor tasks described by Groovy scripts
- Eclipse-based client: currently under development, will replace the GPE Expert Client eventually

3.1 GPE Application Client

The GPE Application Client is a simple-to-use client for submitting single tasks. For each application you want to use you need a special plugin, a so called GridBean, which provides an interface for passing user input to the application.

3.2 Installation

This section describes how to configure the GPE Application Client at first usage. Download the GPE clients bundle from www.unicore.eu/download and execute the installer by double-clicking on the jar-_le or entering in command line:

java -jar gpe4unicore - client - package -1.4.3 - installer . jar

Check the installation directory and enter the UNICORE gateway hostname (localhost if you chose the default), the gateway port (default 8080), and the site name (default DEMO-SITE), then finish the installation.

Now start the Application Client:

cd GPE4Unicore Client / bin
./ application - client .sh

Under Windows the executable is called application-client.bat.
At first start, the client will create an empty keystore for you and ask you for a new password. The keystore will hold all the certificates you need to access the grid. Now you have to load your certificates into the keystore:

Select Settings->Keystore Editor from the menu.

The user certificate which authenticates you to the UNICORE server can be loaded via Actions->Import Keystore.... Select the demo certificate from the server bundle,

UNICORE_Quickstart / certs / demouser .p12

The password is the user.
Now load the CA certificate with which the client can check if it is talking to a trusted UNICORE server:

Select Actions->Import Certificate...
and open the demo CA certificate from the server bundle,

UNICORE_Quickstart / certs /fakeca - cert . pem

Save the keystore (File->Save) and exit the keystore editor (File->Exit). The client needs additional information about where to find the UNICORE server.

Select Settings->Preferences from the menu and select the tab Registries.

The client installer should have already set up an entry with the URL

https :// localhost :8080/ DEMO - SITE / services / Registry

If not, add a new entry now. Leave the preferences window and, if you changed
the registries, perform

File->Refresh Registries from the menu.

3.3 PovRay GridBean

You should see a registry entry in the upper left panel of the Application Client. In figure 1 you can see an example with a registry called localhost.
Right-click on the registry and select Refresh from the popup-menu to load the UNICORE
sites known to that registry.

For each site, you will maybe see an additional workflow entry, which is of no use in the Application client. Click on the site itself (not the workflow entry, in figure 1 it is called DEMO-SITE) and select Connect from the popup-menu.

Now you can create your first job:
rendering an image with PovRay.
Load the appropriate GridBean by selecting File->Load GridBean from the menu and Opening

GPE4Unicore Client / gridbeans / PovrayGridBean . jar

The second tab now should change its name to PovRay.

Select the PovRay tab, you will see the interface for configuring the PovRay job. In the PovRay tab, select the Source File tab to specify the input file (see _gure 2). Load a PovRay source file via File->Open... in the soucre file menu.
You will find an example file in the client installation,

GPE4Unicore Client / tutorial / files / Example . pov

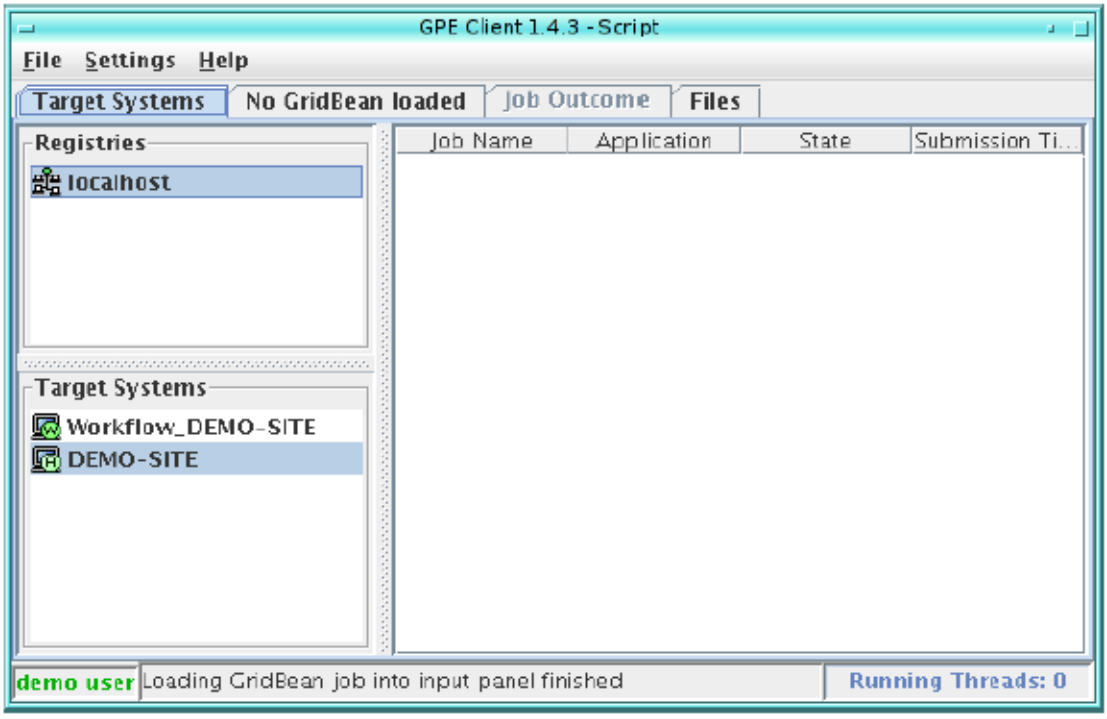


Figure 1: Application Client with loaded registries and sites

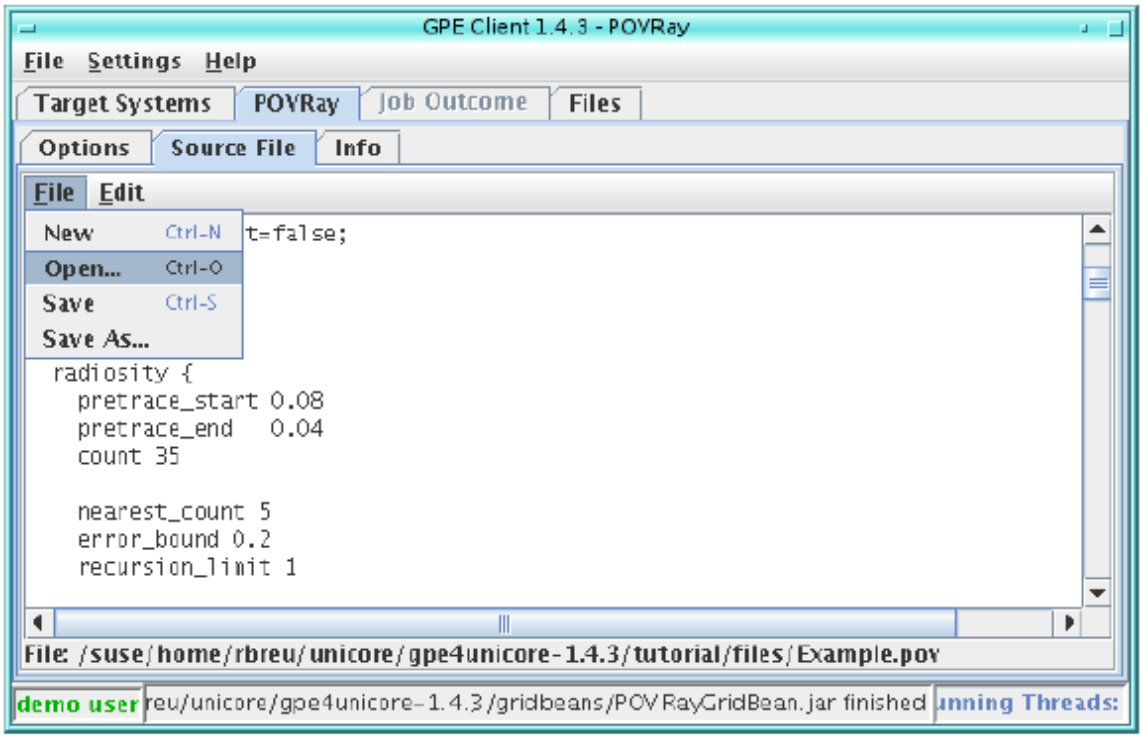


Figure 2: PovRay GridBean: Load a source file

Go back to the Target Systems tab and make sure that one target site is selected. Now you can submit the job by selecting File->Submit from the menu.
Confirm the maximal lifetime of the job. In the right panel, an entry for the submitted job should appear. You can right-click on the entry and choose Refresh from the popup-menu to manually update the status of the job.
When the job has run successful, right-click on it and select Fetch Outcome.
You should see that the Job Outcome tab now becomes active. Select the outcome tab, and in the outcome tab select the image tab to view the rendered image.

3.4 Script GridBean

Submitting a script with the script GridBean is quite similar to the PovRay job. Load the script GridBean via File->Load GridBean in the menu, then select

GPE4Unicore Client / gridbeans / ScriptGridBean . jar

Select the Script tab and in the Script tab the Script input tab. In the pull-down menu at the top of the tab you can choose the script type, for this example select Bash. If you don't see any entries in the pull-down menu, go back to the Target Systems tab and select a target site. The client will then read the possible script types from the server.
Now write a short script in the input panel in the middle and save it
(File->Save as...), e.g:

```
echo " Hello world "
date
```

Submit the job and fetch the outcome as in previous section.

3.5 Script GridBean & Output Files

A UNICORE jobs runs in a temporary directory which is deleted after the job has finished. Any file that is to survive the job runtime has to be moved to the job's outcome directory. The script GridBean does that automatically for standard output and standard error, and the PovRay GridBean additionally moves the rendered image to the outcome directory.
If you want to keep any further files, you have to name them explicitly.
Write a short script as in previous section which produces an output file, e.g.:

```
date > output .txt
```

Don't forget to save your script!
In the Script tab, select the Output Files tab at the bottom and click the Add...-button. Enter an arbitrary name in the Identi_er _eld and output.txt in the Filename field, see figure 3.
Submit the job and fetch the outcome as in previous section. In the Outcome tab, you can now select the Output Files tab, where your specified output file is listed. Select it to save it on your local disk by using the Save as...-button on the right.

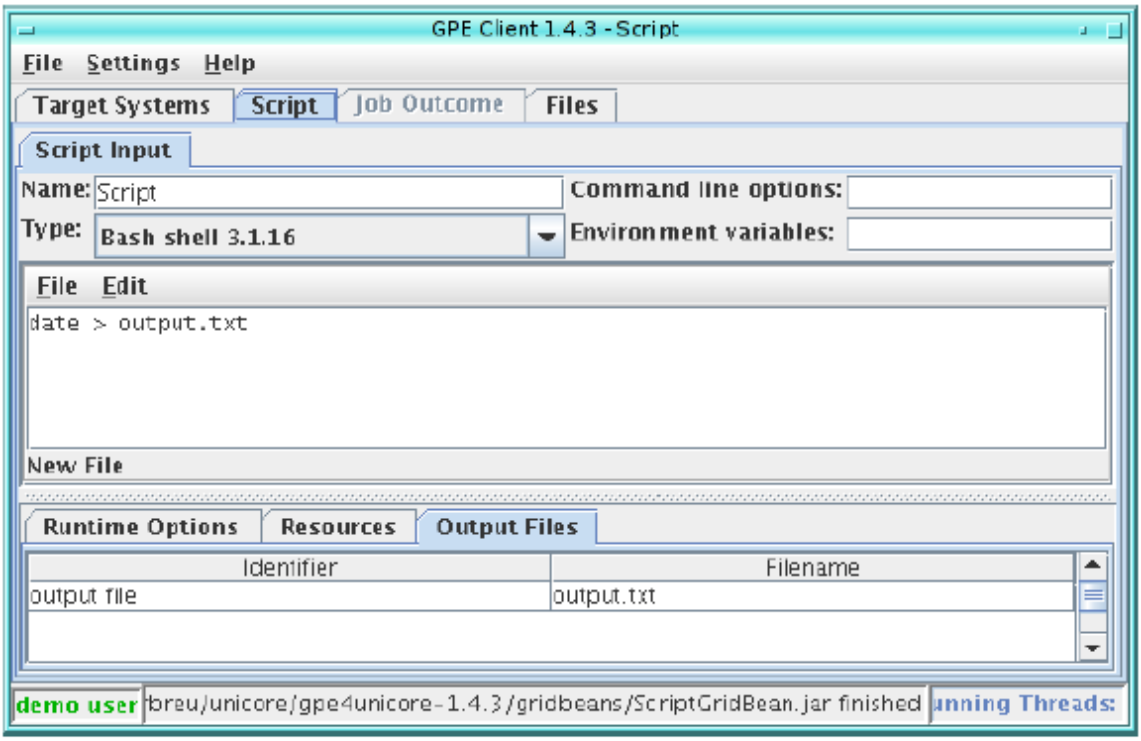


Figure 3: Script GridBean with output file specified

3.6 Script GridBean & Input Files

What about the other way round, if you need files moved to the job's working directory before execution?
Create two text files with nearly the same content and save them as text1.txt and text2.txt.

Select the Files tab and edit the entry InputFileset. Click Add File,
choose Local File and enter the path to text1.txt.
Repeat the procedure for text2.txt, see figure 4.
Now write a short script as in previous section that uses those input files, e.g.
a script which compares the files:
diff text1 .txt text2 . txt
Save the script, submit the job and fetch the outcome as in section 3.2.

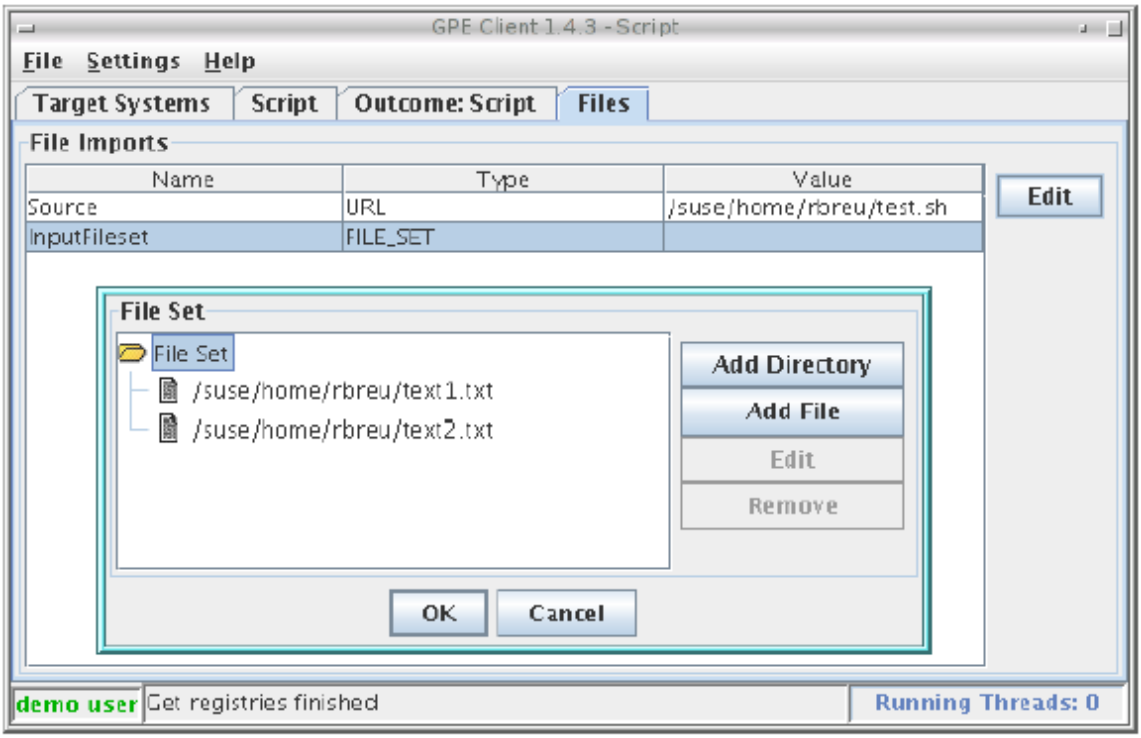


Figure 4: Script GridBean: Specify input files