



NGS

National Grid Service

101010001000000100100

101010001000000100100

?

@

1010100010000001

The NGS Applications Repository / Grid Portal

David Meredith

NGS + Grid Technology Group, e-Science Centre,
Daresbury Laboratory, UK

d.j.meredith@dl.ac.uk

NGS Portal

The NGS portal can be used to access and interact with the HPC and Data resources available on the NGS Grid via SSO (Certificates + myproxy):

- [Browse for different applications](#) available on a Grid this includes your own personal applications and pre-configured applications (e.g. the NGS is currently publishing applications within the NGS portal to be made easily available for its users).
- [Create descriptions of applications](#) and their associated resources using JSDL.
- [Submit/monitor](#) compute jobs/applications.
- [Access and move data](#) around the Compute and Data Grid.

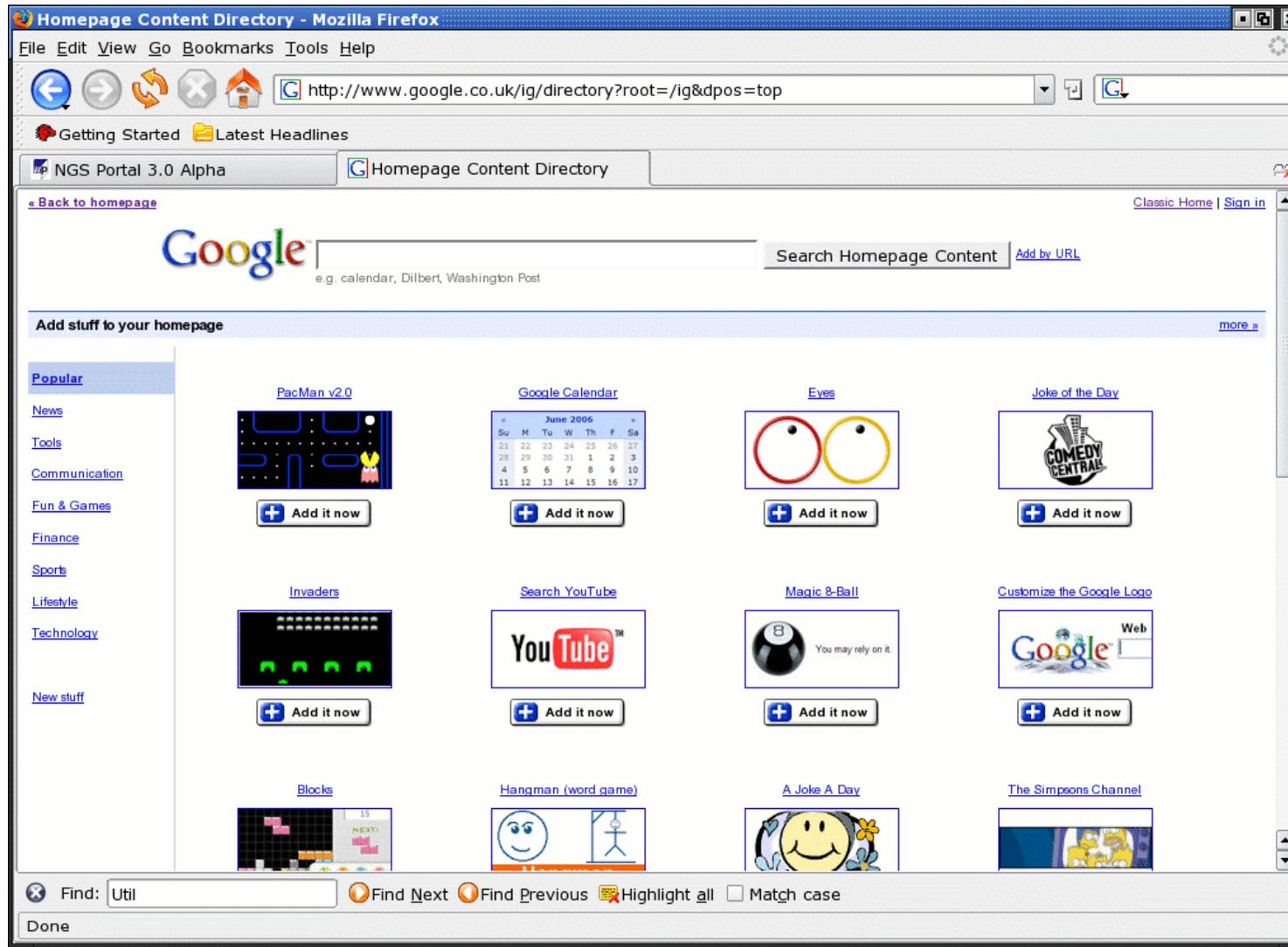
Portal and Portlets

- The NGS Grid portal extends a JSR-168 compliant portal container which hosts a selection of ‘portlets.’
- Portlets are online-accessible applications that are hosted and managed within the portal container.
- The list of portlets that are deployed to the portal make up the portal’s overall functionality (users may be interested only in a selection of portlets)
- Main Benefit: Portlets facilitate the sharing and re-use of applications (168-compliant 3rd party portlets can be used within a portal as required).
- NGS and SciTech e-Science Centre are developing a collection of portlets designed for the Computational and Data Grid.
- The NGS portal is a current implementation of these Grid portlets.

A Famous Portal + Portlets



Select Portlets of Interest (customisation)



NGS portal – <https://portal.ngs.ac.uk>

(Google for 'ngs portal')

The screenshot shows the NGS Job Portal in a Mozilla Firefox browser window. The address bar displays <https://portal.ngs.ac.uk/JobProfiles.jsf>. The page header includes the NGS logo (National Grid Service) and the Science & Technology Facilities Council logo. The main navigation menu contains links for Start, Authenticate/Credentials, Applications Repository, Upload/Download, Job Categories, Browse Host, Admin, and Info/ToDo. Below the navigation, there is a section for 'Active Job / Loaded Application' with a text input field containing 'JobProfile name' and buttons for Save, SaveAsNew, and New. A secondary navigation bar includes links for Detail, Candidate Hosts, Description, Args, Env, File Systems, StageData, Files/Links, JSDL, and Submit. A 'Logout' button is visible, along with the user information 'User: ANONYMOUS_USER' and a link to 'Authenticate Now'. The main content area is titled 'Applications Repository' and contains instructions: 'Load an application from the table below to change the Active Job / Loaded Application and its settings. (Executable permissions may be required to run selected jobs - contact NGS helpdesk for access)'. Below this, there is a 'List Applications' section with search filters: 'In Category: (Create/Edit) All', 'With Status: all', and two buttons: 'Find NGS Applications' and 'Find Personal Applications'. At the bottom, a table header is visible with columns: Details, Application Name, Version, Job Name/Detail, Modified, and Load. The browser status bar shows 'Done' and the URL 'portal.ngs.ac.uk'.

NGS portal main functionality

1. A **searchable database** of personal and shared JSDL job profile documents (Job Submission Description Language, i.e. ‘job recipes/templates’).
 - JSDL can be **browsed for, selected, loaded and saved** in order to run applications on the Grid (loaded either ‘out-of-the-box’ or, more usually loaded, modified/tweaked as required and saved).
 - JSDL can be **searched for** by category of interest in the portal (e.g bioinformatics, chemistry, tutorials/examples).
 - JSDL documents can be **pre-configured and published** by the portlet administrator(s) to be made available to all other users.
2. A **JSDL GUI editor** for graphically authoring, validating, sharing, uploading jobs described in JSDL.
3. An **extensible Grid job submission** and monitoring application for heterogonous Grids (currently, only Globus but more Grid middleware providers are being added, e.g. JDL/gLite).

JSDL – Job Submission Description Language

Ali Anjomshoaa, Fred Brisard, Michel Drescher, Donal K. Fellows, William Lee, An Ly, Steve McGough, Darren Pulsipher, Andreas Savva, Chris Smith

- **JSDL 1.0 is an OGF recommendation**
- **JSDL 1.0 is published as GFD-R-P.56 –**
 - Description of JSDL elements and XML Schema,
 - Available at: <http://www.ggf.org/gf/docs/?final>

**A language for
describing the
requirements of
computational jobs
for submission to
Grids and other
systems.**

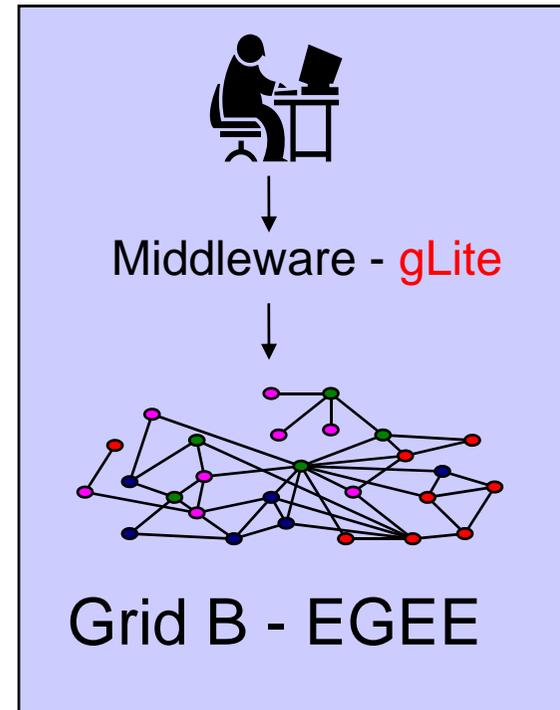
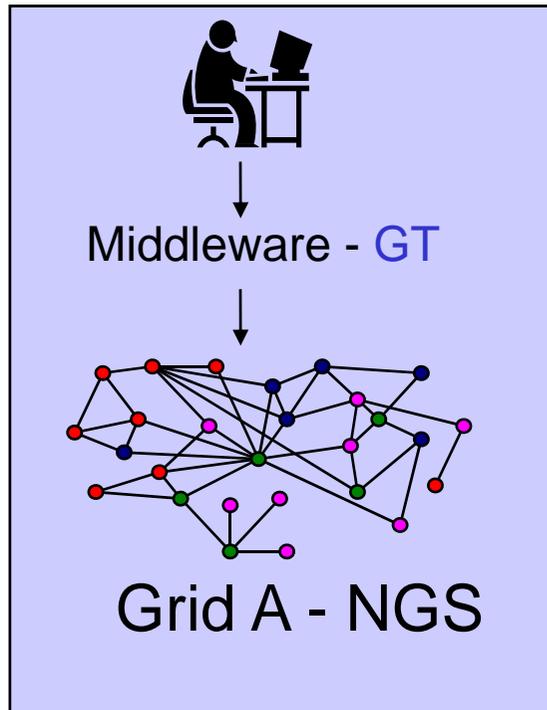
```

<jSDL:Application>
  <jSDL:ApplicationName>gnuplot</jSDL:ApplicationName>
  <jSDL-posix:POSIXApplication>
    <jSDL-posix:Executable>
      /usr/local/bin/gnuplot
    </jSDL-posix:Executable>
    <jSDL-posix:Argument>control.txt</jSDL-posix:Argument>
    <jSDL-posix:Argument>DavesControlFile.txt</jSDL-posix:Argument>
    <jSDL-posix:Input>input.dat</jSDL-posix:Input>
    <jSDL-posix:Output>output1.png</jSDL-posix:Output>
  </jSDL-posix:POSIXApplication>
</jSDL:Application>
<jSDL:Resources>
....
  
```

Example fragment

Why JSDL

Grids are currently heterogeneous:



- Different middleware implementations adopt different methods for the description of applications and their associated resources, and for their subsequent execution to a Grid.
- A Number of different data storage resources are also relevant for the management and transfer of data, e.g. GsiFTP, SRB, SRM, WebDav and (S)FTP.

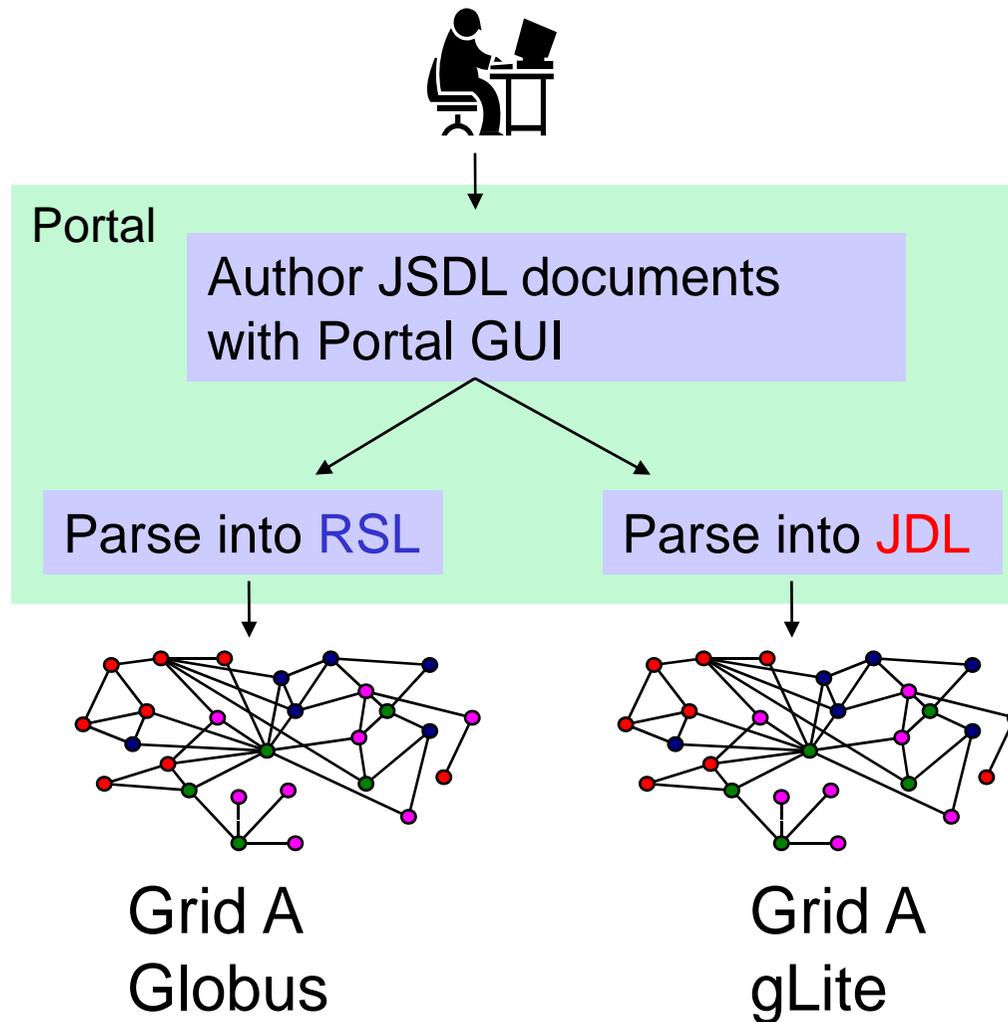
Grid A Globus RSL (Resource Specification Language)

```
&(executable=$(GLOBUSRUN_GASS_URL)/home/ngs0153/cpi) (arguments= 30 fileA)
(jobType=mpi) (environment = (NGSMODULES mpich-gm/1.2.5..10-
intel8.1:intel/fce/9.1.032) (TMP /tmp)) (count = 4) (hostCount = 8) (minMemory = 512)
(maxWallTime = 3) (directory=/home/ngs0153) (stdin=/home/ngs0153/cpi.in)
(stdout=/home/ngs0153/cpi.out) (stderr=/home/ngs0153/cpi.err)
```

Grid B gLite JDL (Job Description Language)

```
Type = "Job";
JobType = "Normal";
RetryCount = 3;
Executable = "/home/ngs0153/cpi";
Arguments = "30 fileA";
VirtualOrganisation = "myGridVOproject";
StdInput = "cpi.in";
StdOutput = "cpi.out";
StdError = "cpi.err";
InputSandbox = { "gsiftp://grid-data.rl.ac.uk:2811/home/ngs0153/cpi", "gsiftp://grid-
data2.dl.ac.uk:2811/myhome/fileA" };
InputSandboxDestFileName = { "cpi", "fileA" };
OutputSandbox = { "cpi.out" };
OutputSandboxDestURI = { "gsiftp://mygridhome.dl.ac.uk:2811/myhome" };
DeleteOnTermination = { "fileA" };
Environment = { "NGSMODULES=mpich-gm/1.2.5..10-intel8.1:intel/fce/9.1.032", "TMP=/tmp" };
Requirements = ( other.GlueCEInfoLRMSType == "PBS" ) && ( member( GlueCEInfoHostName,
{"grid-data.rl.ac.uk:2119" , "mygrid-resource.dl.ac.uk:2119" } ) ) && ( GlueHostProcessorModel ==
"Intel" );
Rank = -other.GlueCEStateEstimatedResponseTime;
```

JSDL and the NGS Grid portal



- One interface definition for job submission
- One job description language
- JSDL is a proposed standard job submission description language (middleware agnostic)

JSDL v 1.0

1. XML Schema language for describing compute jobs in a **platform independent** language (XML).
2. Is **agnostic of middleware** - no dependencies on Globus, WSRF, gLite (portal that is **generic and not tied to any particular set of Grid technologies**).
3. OGF (once GGF) **recommendation**.
4. JSDL documents can be **validated** against the JSDL and JSDL POSIX XSD Schema documents to ensure correctness.
5. A JSDL XML *document* describes the requirements of a compute job and its associated resources, e.g. description of required files / data
 - *What to do, not how to do it*
 - It is an **extensible** template language (elements defined in another XML schema with a different namespace can be added to a JSDL document). This facilitates customisation and nesting of extra information.
 - Is **loosely typed** – does not impose tight controls on the values of elements using advanced XML schema restriction techniques such as `xsd:patterns`
6. **Not included** in JSDL version 1.0
Scheduling, Workflow, Security ...

JSDL Document Structure Overview

- <JobDefinition>
- <JobDescription>
- <JobIdentification ... />?
- <Application ... />?
- <Resources... />?
- <DataStaging ... />*
- </JobDescription>
- </JobDefinition>

- Note:
 - None [1..1]
 - ? [0..1]
 - * [0..n]
 - + [1..n]

***** Slide content provided by S.McGough, London eScience Centre, Imperial College, London *****

```

<?xml version="1.0" encoding="UTF-8"?>
<jSDL:JobDefinition xmlns:jSDL="http://schemas.ggf.org/2005/11/jSDL"
  xmlns:jSDL-posix="http://schemas.ggf.org/jSDL/2005/11/jSDL-posix">
  <jSDL:JobDescription>
    <jSDL:JobIdentification>
      <jSDL:JobName>My gnuplot </jSDL:JobName>
      <jSDL:Description>my description here </jSDL:Description>
    </jSDL:JobIdentification>
    <jSDL:Application>
      <jSDL:ApplicationName>gnuplot</jSDL:ApplicationName>
      <jSDL-posix:POSIXApplication>
        <jSDL-posix:Executable>
          /usr/local/bin/gnuplot
        </jSDL-posix:Executable>
        <jSDL-posix:Argument>control.txt</jSDL-posix:Argument>
        <jSDL-posix:Argument>DavesControlFile.txt</jSDL-posix:Argument>
        <jSDL-posix:Input>input.dat</jSDL-posix:Input>
        <jSDL-posix:Output>output1.png</jSDL-posix:Output>
      </jSDL-posix:POSIXApplication>
    </jSDL:Application>
    <jSDL:Resources>
      <jSDL:IndividualPhysicalMemory>
        <jSDL:LowerBoundedRange>2097152.0</jSDL:LowerBoundedRange>
      </jSDL:IndividualPhysicalMemory>
      <jSDL:TotalCPUCount>
        <jSDL:Exact>1.0</jSDL:Exact>
      </jSDL:TotalCPUCount>
    </jSDL:Resources> ....
  </jSDL:JobDefinition>

```

Example JSDL 1.0 Document

JSDL – Job Submission Description Language XSD Schema

```

<!--=====-->
<xsd:complexType name="Environment_Type">
  <xsd:simpleContent>
    <xsd:extension base="xsd:string">
      <xsd:attribute name="name" type="xsd:NCName" use="required"/>
      <xsd:attribute name="filesystemName" type="xsd:NCName" use="optional"/>
      <xsd:anyAttribute namespace="##other" processContents="lax"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
<!--=====-->
<xsd:complexType name="Argument_Type">
  <xsd:simpleContent>
    <xsd:extension base="xsd:normalizedString">
      <xsd:attribute name="filesystemName" type="xsd:NCName" use="optional"/>
      <xsd:anyAttribute namespace="##other" processContents="lax"/>
    </xsd:extension>
  </xsd:simpleContent>
</xsd:complexType>
  
```

JSDL Repository / Database

Portal is 'open', are free to browse public JSDL documents without log-in and free to use portal as open JSDL editor.

Login to browse personal applications and submit jobs.

List jobs, read job descriptions and load a job to initialise the 'Active Job.'

Changes to the parameters in the GUI will update the generated JSDL automatically.

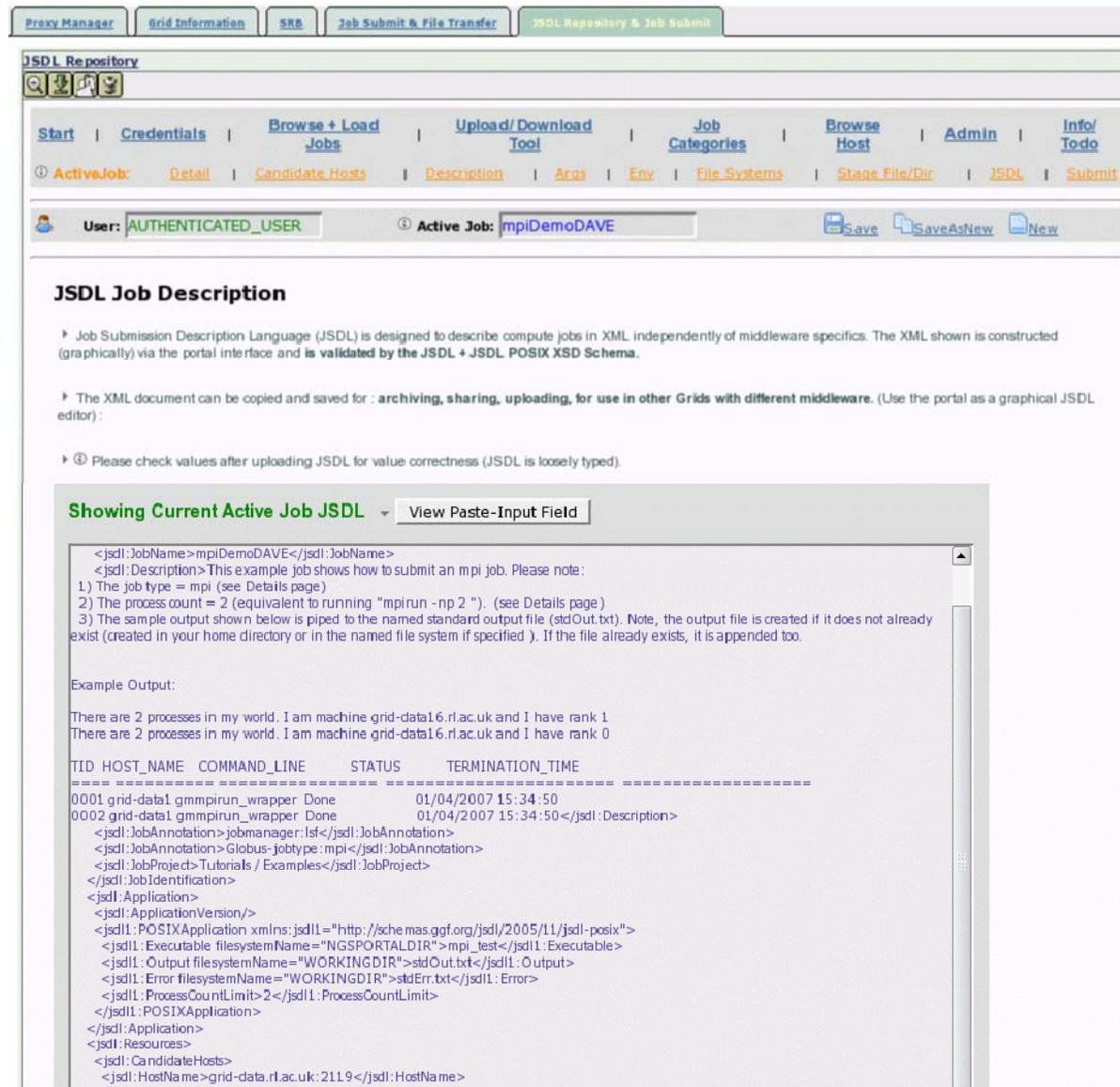
The screenshot shows the JSDL Repository web interface. At the top, there are navigation tabs: Proxy Manager, Grid Information, SRB, Job Submit & File Transfer, and JSDL Repository & Job Submit. Below the tabs, there are several icons and a set of navigation links: Start, Credentials, Browse + Load Jobs, Upload/Download Tool, Job Categories, Browse Host, Admin, and Info/ Todo. Underneath, there are more specific links for ActiveJob, Detail, Candidate Hosts, Description, Args, Env, File Systems, Stage File/Dir, JSDL, and Submit. A user information bar shows 'User: AUTHENTICATED_USER' and 'Active Job: JobProfile name', with Save, SaveAsNew, and New buttons. The main content area is titled 'Browse and Load the 'Active Job'' and contains two instructions: 1) Select a job category (e.g. Tutorials/Examples) and list the jobs in that category (NGS or personal jobs); 2) Select and load the ActiveJob from the list (click on the load link). Below the instructions is a search form with three fields: 'In Job Category: (Create/Edit JobCategories)' with a dropdown menu set to 'RAL Bioinformatics'; 'With Status:' with a dropdown menu set to 'all'; and 'Search / List Job Profiles:' with two buttons, 'NGS' and 'Personal'. Below the search form, it says 'Results Found: (9)'. At the bottom, there is a table with 9 rows of job listings. Each row has columns for Description, Name, Exe, Modified, Status (Check), and Load. The Load column contains a checkbox and a 'load' link. A 'Reset' button is located at the bottom right of the table.

Description	Name	Exe	Modified	Status (Check)	Load
View	SIESTA (paralell example)	/apps/siesta-mpi/siesta-2	Jan 17, 2007	<input type="checkbox"/>	load
View	FASTA (serial example)	/usr/local/applications/bio	Jan 17, 2007	<input type="checkbox"/>	load
View	NAMD (paralell example)	/usr/local/applications/bio	Jan 17, 2007	<input type="checkbox"/>	load
View	GROMACS (grompp exam	/usr/local/applications/bio	Jan 17, 2007	<input type="checkbox"/>	load
View	GROMACS (MD example)	/usr/local/applications/bio	Jan 17, 2007	<input type="checkbox"/>	load
View	mpiBLAST (blastn exampl	/usr/local/applications/bio	Jan 18, 2007	<input type="checkbox"/>	load
View	EMBOSS (seqret example)	/usr/local/applications/bio	Jan 18, 2007	<input type="checkbox"/>	load
View	EMBOSS (wossname exa	/usr/local/applications/bio	Jan 18, 2007	<input type="checkbox"/>	load
View	BLAST (serial application)	/usr/local/applications/bio	Jan 19, 2007	<input type="checkbox"/>	load

Active Job's JSDL

The Active Job JSDL is automatically created, updated and validated by the portal by changing parameters in the portal GUI.

The portlet acts as a JSDL GUI editor



JSDL Job Description

- Job Submission Description Language (JSDL) is designed to describe compute jobs in XML independently of middleware specifics. The XML shown is constructed (graphically) via the portal interface and is validated by the JSDL + JSDL POSIX XSD Schema.
- The XML document can be copied and saved for : archiving, sharing, uploading, for use in other Grids with different middleware. (Use the portal as a graphical JSDL editor) :
- Please check values after uploading JSDL for value correctness (JSDL is loosely typed).

Showing Current Active Job JSDL

```

<jSDL:JobName>mpiDemoDAVE</jSDL:JobName>
<jSDL:Description>This example job shows how to submit an mpi job. Please note:
1) The job type = mpi (see Details page)
2) The process count = 2 (equivalent to running "mpirun -np 2 "). (see Details page)
3) The sample output shown below is piped to the named standard output file (stdOut.txt). Note, the output file is created if it does not already exist (created in your home directory or in the named file system if specified ). If the file already exists, it is appended too.

Example Output:
There are 2 processes in my world. I am machine grid-data16.rl.ac.uk and I have rank 1
There are 2 processes in my world. I am machine grid-data16.rl.ac.uk and I have rank 0

TID  HOST_NAME  COMMAND_LINE  STATUS  TERMINATION_TIME
-----
0001  grid-data1  gmpirun_wrapper  Done    01/04/2007 15:34:50
0002  grid-data1  gmpirun_wrapper  Done    01/04/2007 15:34:50</jSDL:Description>
<jSDL:JobAnnotation>jobmanager:lsf</jSDL:JobAnnotation>
<jSDL:JobAnnotation>Globus-jobtype:mpi</jSDL:JobAnnotation>
<jSDL:JobProject>Tutorials / Examples</jSDL:JobProject>
</jSDL:JobIdentification>
<jSDL:Application>
<jSDL:ApplicationVersion>
<jSDL:POSIXApplication xmlns:jSDL="http://schemas.grid.org/jSDL/2005/11/jSDL-posix">
<jSDL:Executable filesystemName="NGSPORTALDIR">mpi_test</jSDL:Executable>
<jSDL:Output filesystemName="WORKINGDIR">stdOut.txt</jSDL:Output>
<jSDL:Error filesystemName="WORKINGDIR">stdErr.txt</jSDL:Error>
<jSDL:ProcessCountLimit>2</jSDL:ProcessCountLimit>
</jSDL:POSIXApplication>
</jSDL:Application>
<jSDL:Resources>
<jSDL:CandidateHosts>
<jSDL:HostName>grid-data.rl.ac.uk:2119</jSDL:HostName>
</jSDL:CandidateHosts>

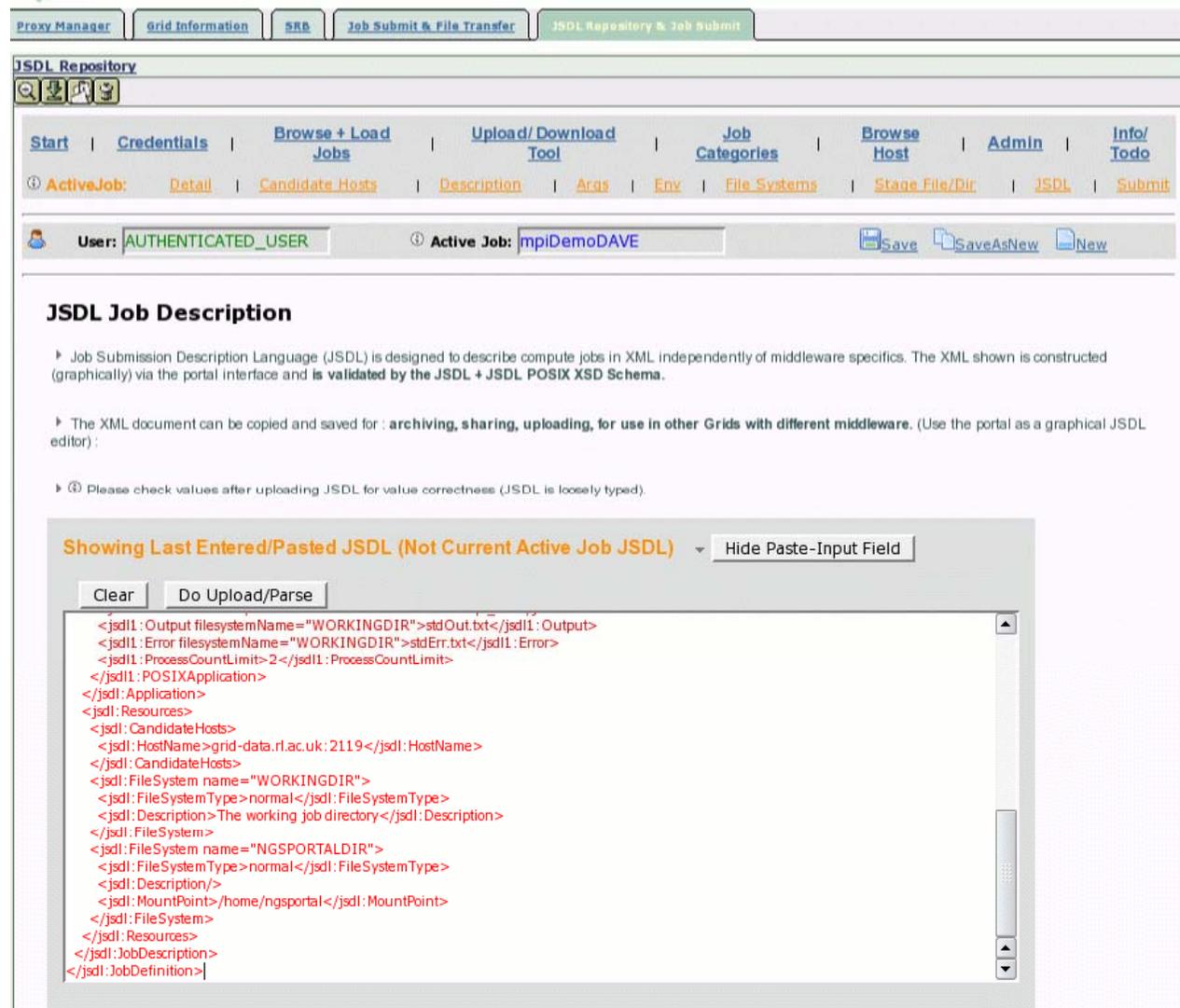
```

Upload / Share JSDL

JSDL documents can be uploaded to the portal in order to initialise the Active Job.

Validation errors and messages are displayed in the interface when uploading JSDL.

The portlet allows the sharing of job profiles and JSDL between users and user communities.



JSDL Repository

Start | Credentials | Browse + Load Jobs | Upload/Download Tool | Job Categories | Browse Host | Admin | Info/ToDo

Active Job: Retail | Candidate Hosts | Description | Args | Env | File Systems | Stage File/Dir | JSDL | Submit

User: AUTHENTICATED_USER | Active Job: mpiDemoDAVE

JSDL Job Description

- Job Submission Description Language (JSDL) is designed to describe compute jobs in XML independently of middleware specifics. The XML shown is constructed (graphically) via the portal interface and is validated by the JSDL + JSDL POSIX XSD Schema.
- The XML document can be copied and saved for : archiving, sharing, uploading, for use in other Grids with different middleware. (Use the portal as a graphical JSDL editor) :
- Please check values after uploading JSDL for value correctness (JSDL is loosely typed).

Showing Last Entered/Pasted JSDL (Not Current Active Job JSDL)

```
<jSDL:Output filesystemName="WORKINGDIR">stdOut.txt</jSDL:Output>
<jSDL:Error filesystemName="WORKINGDIR">stdErr.txt</jSDL:Error>
<jSDL:ProcessCountLimit>2</jSDL:ProcessCountLimit>
</jSDL:POSIXApplication>
</jSDL:Application>
<jSDL:Resources>
<jSDL:CandidateHosts>
<jSDL:HostName>grid-data.rl.ac.uk:2119</jSDL:HostName>
</jSDL:CandidateHosts>
<jSDL:FileSystem name="WORKINGDIR">
<jSDL:FileSystemType>normal</jSDL:FileSystemType>
<jSDL:Description>The working job directory</jSDL:Description>
</jSDL:FileSystem>
<jSDL:FileSystem name="NGSPORTALDIR">
<jSDL:FileSystemType>normal</jSDL:FileSystemType>
<jSDL:Description/>
<jSDL:MountPoint>/home/ngsportal</jSDL:MountPoint>
</jSDL:FileSystem>
</jSDL:Resources>
</jSDL:JobDescription>
</jSDL:JobDefinition>
```

Active Job Detail

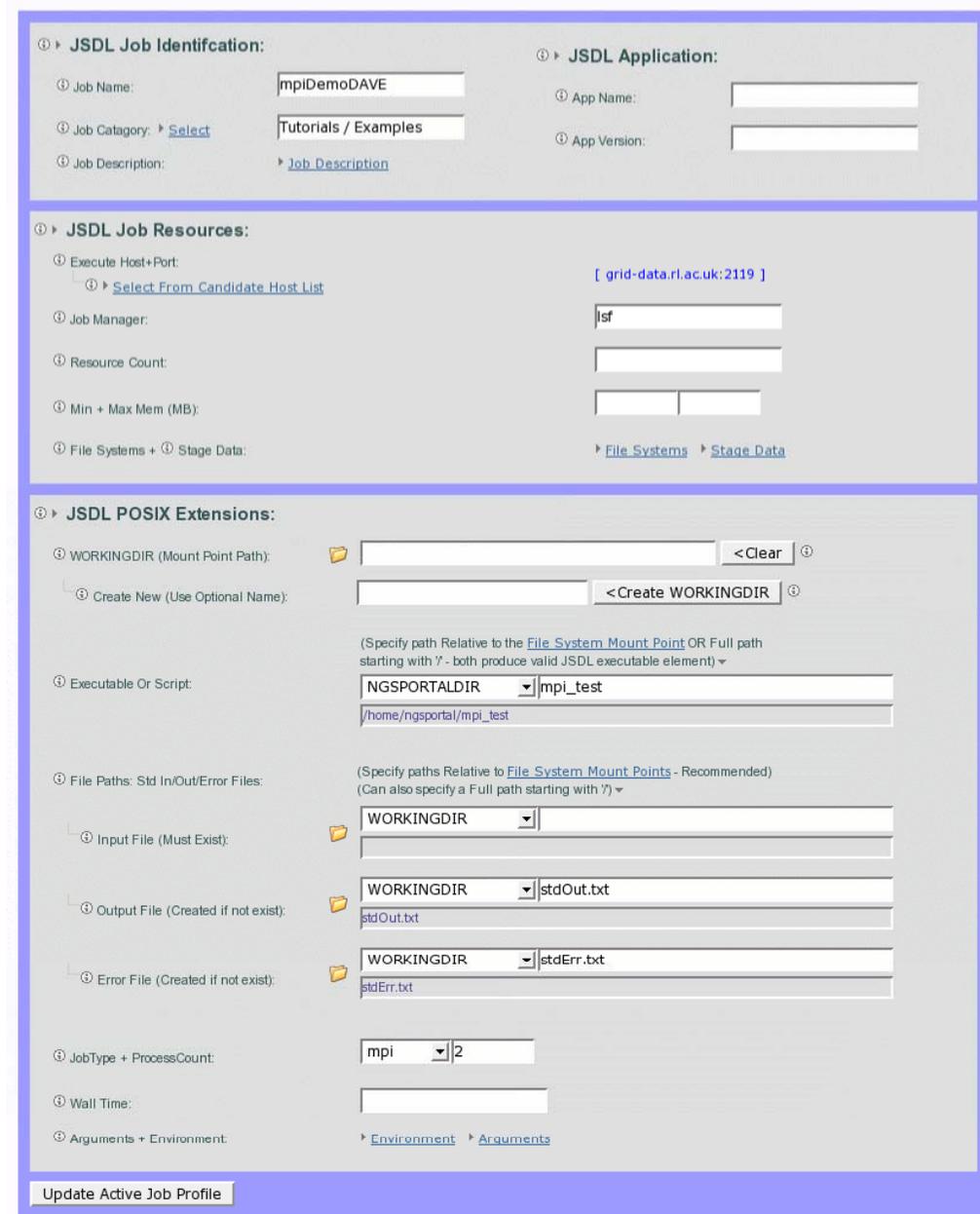
Input fields are filled out for pre-configured applications.

Input fields are taken from the JSDL and JSDL-POSIX extension schemas.

POSIXApplication is a normative JSDL extension. It defines standard POSIX elements

- stdin, stdout, stderr
- Working directory
- Command line arguments
- Environment variables

- **<POSIXApplication>**
- **<Executable ... />**
- **<Argument ... />***
- **<Input ... />?**
- **<Output ... />?**
- **<Error ... />?**
- **<WorkingDirectory ... />?**
- **<Environment ... />***
- ...
- **</POSIXApplication>**



The screenshot displays a web interface for configuring a job profile. It is divided into three main sections:

- JSDL Job Identification:** Fields for Job Name (mpiDemoDAVE), Job Category (Tutorials / Examples), Job Description (Job Description), JSDL Application (App Name and App Version).
- JSDL Job Resources:** Fields for Execute Host+Port (grid-data.rl.ac.uk:2119), Job Manager (lft), Resource Count, Min + Max Mem (MB), and File Systems + Stage Data (File Systems, Stage Data).
- JSDL POSIX Extensions:** Fields for WORKINGDIR (Mount Point Path), Create New (Use Optional Name), Executable Or Script (NGSPORTALDIR, mpi_test, /home/ngsportal/mpi_test), File Paths: Std In/Out/Error Files (WORKINGDIR, stdOut.txt, stdErr.txt), JobType + ProcessCount (mpi, 2), Wall Time, and Arguments + Environment (Environment, Arguments).

An "Update Active Job Profile" button is located at the bottom of the form.

Environment Variables / Arguments

Environment Variables

Setup environment variables required to run the job e.g. "NGSMODULES" ([how to](#))

Edit	Name	Value	
▶ Edit	TMP	/tmp	☐
▶ Edit	NGSMODULES	envVarValue1	☐

[Close Detail](#)

Add Environment Variable:

③ Name:

③ Value:

Paste and parse command line arguments (space and/or line separated values)

```
<jSDL1:POSIXApplication>
  <jSDL1:Argument>
    /usr/local/applications/bioinformatics....
  </jSDL1:Argument>
  <jSDL1:Argument>
    seqr.out
  </jSDL1:Argument>
```

Add required env vars, e.g. 'NGSMODULES' – used to configure application environment

```
<jSDL1:POSIXApplication>
  <jSDL1:Environment name="TMP">
    /tmp
  </jSDL1:Environment>
  <jSDL1:Environment name="NGSMODULES">
    envVarValue1
  </jSDL1:Environment>.....
```

Command Line Arguments

▶ Enter or paste arguments below

▶ Use Space and/or Line separated values

▶ Use double quotes if space is required in a single argument, e.g. "a and b"

Configured Arguments
/usr/local/applications/bioinformatics/EBI/data/rel_tpa_inv_01_r89.dat
seqr.out
-osformat
fasta

Named File Systems

Named file systems used to declare mount points that are required on the consuming system.

File system names are referenced throughout the portlet (and JSDL doc) for substituting mount points where required.

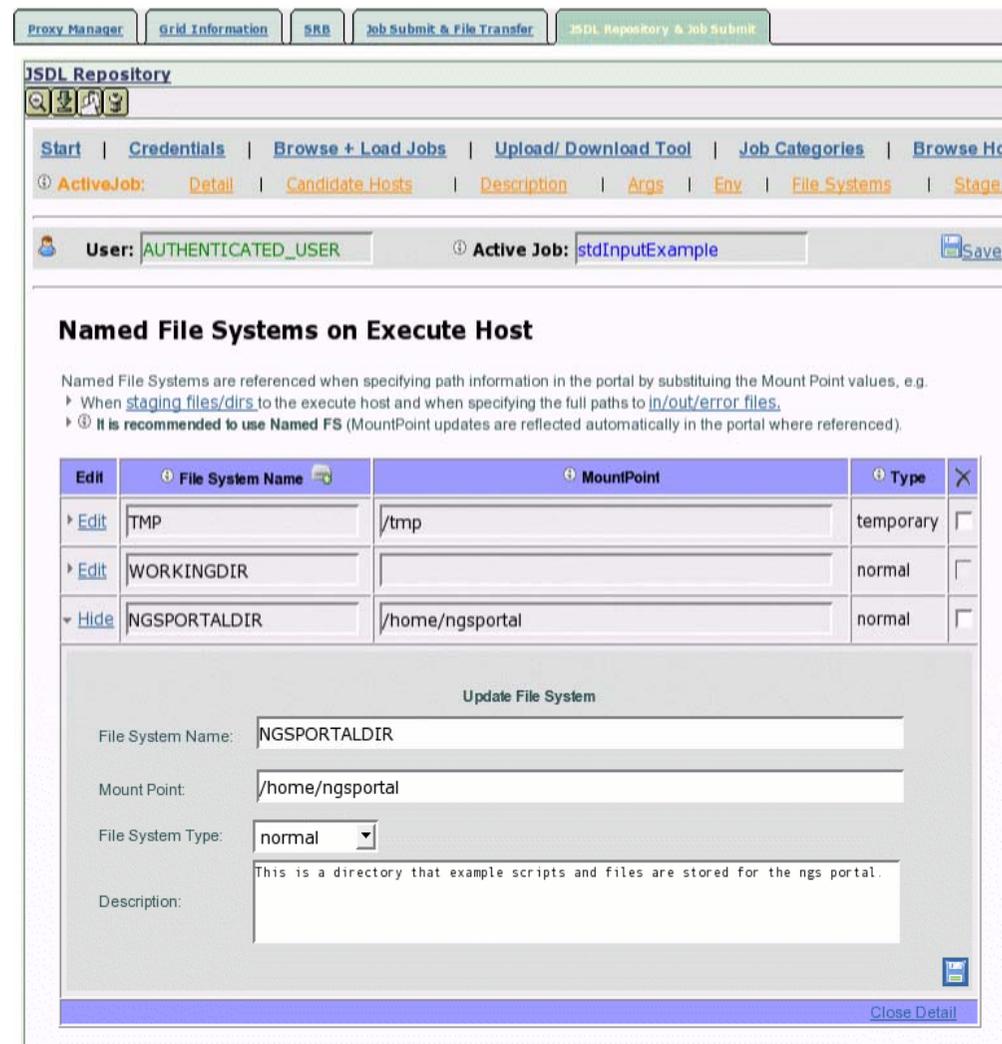
Changes to a FS mount point will be updated automatically throughout the portal/JSDL.

Used when specifying path info e.g., locations to files/dirs, stage data locations etc.

```
<jSDL:FileSystem name="WORKINGDIR">
  <jSDL:MountPoint>
    /home/ngs123
  </jSDL:MountPoint>
</jSDL:FileSystem>
```

- File System can be referenced in other parts of document, e.g.

```
<jSDLposix:Output filesystemName="WORKINGDIR"> cpi.out</jSDL1:Output>
```



The screenshot shows the 'JSDL Repository' web interface. At the top, there are navigation tabs: 'Proxy Manager', 'Grid Information', 'SRB', 'Job Submit & File Transfer', and 'JSDL Repository & Job Submit'. Below the tabs, there are several menu items: 'Start', 'Credentials', 'Browse + Load Jobs', 'Upload/ Download Tool', 'Job Categories', and 'Browse Ho'. A sub-menu for 'ActiveJob' includes 'Detail', 'Candidate Hosts', 'Description', 'Args', 'Env', 'File Systems', and 'Stage'. The user is identified as 'AUTHENTICATED_USER' and the active job is 'stdInputExample'. The main content area is titled 'Named File Systems on Execute Host' and contains explanatory text and a table of file systems.

Named File Systems are referenced when specifying path information in the portal by substituting the Mount Point values, e.g.

- ▶ When [staging files/dirs](#) to the execute host and when specifying the full paths to [in/out/error files](#).
- ▶ It is recommended to use Named FS (MountPoint updates are reflected automatically in the portal where referenced).

Edit	File System Name	MountPoint	Type	
▶ Edit	TMP	/tmp	temporary	<input type="checkbox"/>
▶ Edit	WORKINGDIR		normal	<input type="checkbox"/>
▼ Hide	NGSPORTALDIR	/home/ngsportal	normal	<input type="checkbox"/>

Update File System

File System Name:

Mount Point:

File System Type:

Description:

Stage Data

Compile a list of data from across the Grid that should be copied to the consuming system prior to job execution (source URI). Also after job completes (target URI).

JSDL does not mandate the protocol / URI format.

Data is staged relative to named file systems on the consuming system.

The source URI can be either specified manually or, more normally, browsed for in the 'Browse Host' page.

The screenshot shows the 'JSDL Repository' web interface. At the top, there are navigation tabs: Proxy Manager, Grid Information, SRB, Job Submit & File Transfer, and JSDL Repository & Job Submit. Below this is a search bar and a menu with options like Start, Credentials, Browse + Load Jobs, Upload/Download Tool, Job Categories, and Browse Host. The 'Active Job' is 'stdInputExample'. The user is 'AUTHENTICATED_USER'. The main section is titled 'Stage Data To The Execute Host' and contains instructions on building a list of files/dirs to be copied. Below the instructions are two links: 'Browse Host For Stage Data (Recommended)' and 'Manually Add Stage Data (Show Data Entry Panel)'. A table lists the staged data:

Edit	File System Name (Add)	File/Dir Name	Creation	Source URI (Browse Host)
▶ Edit	WORKINGDIR	dave.pl	overwrite	gsiftp://grid-data.rl.ac.uk:2
▶ Edit	WORKINGDIR	dave.txt	overwrite	gsiftp://grid-data.rl.ac.uk:2
▼ Hide	WORKINGDIR	dummy.scala	overwrite	gsiftp://grid-data.rl.ac.uk:2

Below the table is a form for adding new data. It includes fields for 'Data Source' (Source URI: gsiftp://grid-data.rl.ac.uk:2811/home/ngs0153/dummy.scala), 'Destination on Execute Host' (File System: WORKINGDIR, Name: dummy.scala), 'Description' (Please provide a valid description for this file/dir), and 'Creation Option' (overwrite). At the bottom, there are buttons for 'View Stage Log', 'Validate SourceURIs', 'Validate MountPoints', 'Do Staging', 'Delete Selected', and 'Close Detail'.

<jSDL:DataStaging>

<jSDL:FileName>dave.txt</jSDL:FileName>

<jSDL:FilesystemName>WORKINGDIR</jSDL:FilesystemName>

<jSDL:CreationFlag>overwrite</jSDL:CreationFlag>

<jSDL>DeleteOnTermination>true</jSDL>DeleteOnTermination>

<jSDL:Source>

<jSDL:URI>gsiftp://grid-data2.dl.ac.uk:2811/home/ngs0153/dave.txt</jSDL:URI>

</jSDL:Source>

</jSDL:DataStaging>

Candidate Hosts

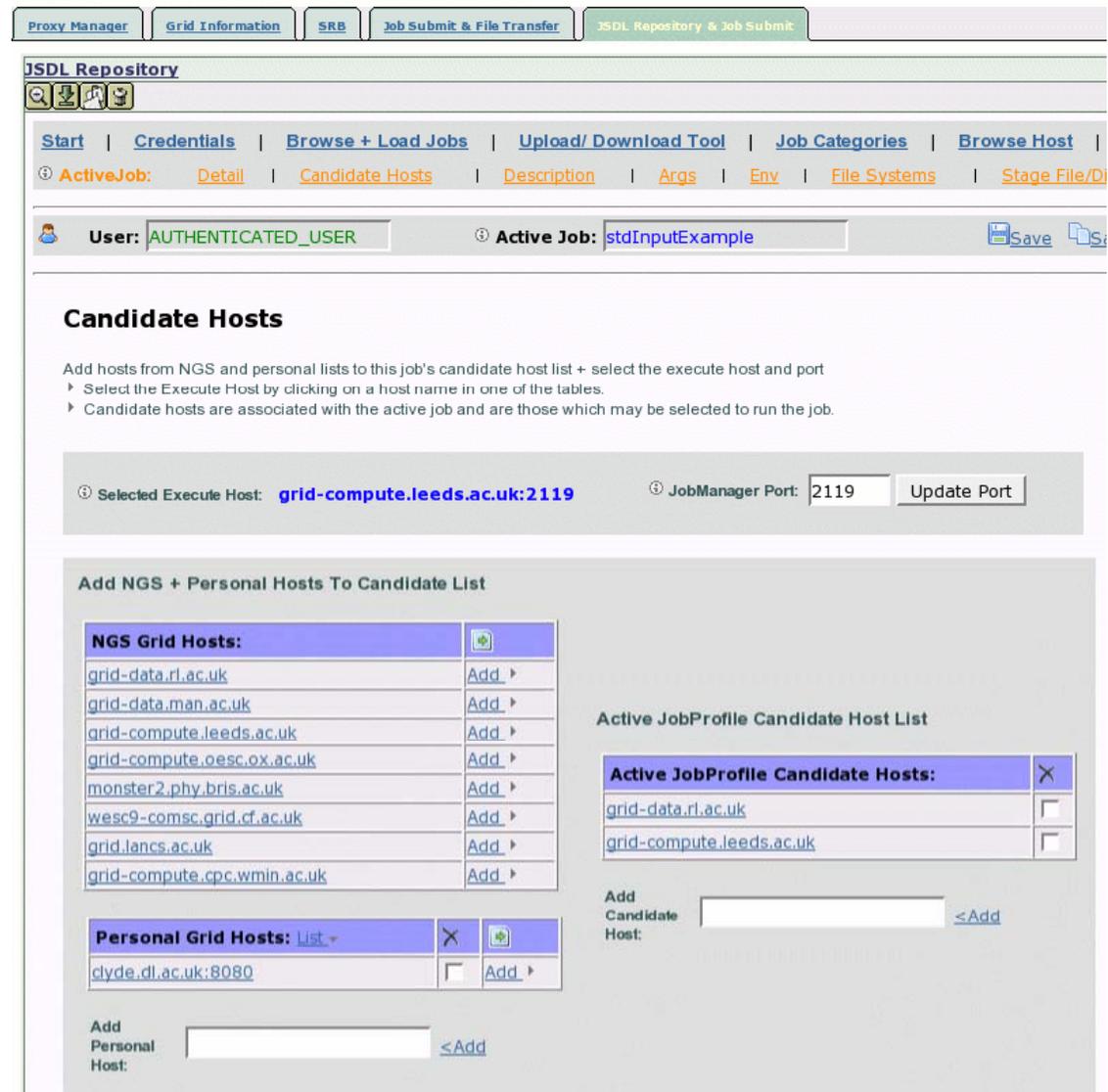
Candidate hosts are consuming systems (execute hosts) that can be to run the Active Job.

The candidate host list can be compiled from a personal host list and from a default host list (available to all users).

In future, a RB matchmaking will be used to auto-compile the candidate host list.

```

<jSDL:CandidateHosts>
  <jSDL:HostName>
    grid-data.rl.ac.uk:2119
  </jSDL:HostName>
  <jSDL:HostName>
    grid-compute.leeds.ac.uk:2119
  </jSDL:HostName>
</jSDL:CandidateHosts>
  
```



The screenshot shows the 'JSDL Repository' web interface. At the top, there are navigation tabs: Proxy Manager, Grid Information, SRB, Job Submit & File Transfer, and JSDL Repository & Job Submit. Below these are various menu options like Start, Credentials, Browse + Load Jobs, Upload/ Download Tool, Job Categories, and Browse Host. The user is logged in as 'AUTHENTICATED_USER' and the active job is 'stdInputExample'. The main section is titled 'Candidate Hosts' and contains instructions on how to add hosts. Below the instructions, there is a section for 'Selected Execute Host' with a dropdown menu showing 'grid-compute.leeds.ac.uk:2119' and a 'JobManager Port' field set to '2119'. There are also sections for 'Add NGS + Personal Hosts To Candidate List' and 'Active JobProfile Candidate Host List'.

Candidate Hosts

Add hosts from NGS and personal lists to this job's candidate host list + select the execute host and port

- ▶ Select the Execute Host by clicking on a host name in one of the tables.
- ▶ Candidate hosts are associated with the active job and are those which may be selected to run the job.

Selected Execute Host: **grid-compute.leeds.ac.uk:2119** JobManager Port: 2119 Update Port

Add NGS + Personal Hosts To Candidate List

NGS Grid Hosts:	
grid-data.rl.ac.uk	Add ▶
grid-data.man.ac.uk	Add ▶
grid-compute.leeds.ac.uk	Add ▶
grid-compute.oesc.ox.ac.uk	Add ▶
monster2.phy.bris.ac.uk	Add ▶
wesc9-comsc.grid.cf.ac.uk	Add ▶
grid.lancs.ac.uk	Add ▶
grid-compute.cpc.wmin.ac.uk	Add ▶

Personal Grid Hosts:	
clyde.dl.ac.uk:8080	<input type="checkbox"/> Add ▶

Add Candidate Host: <Add

Add Personal Host: <Add

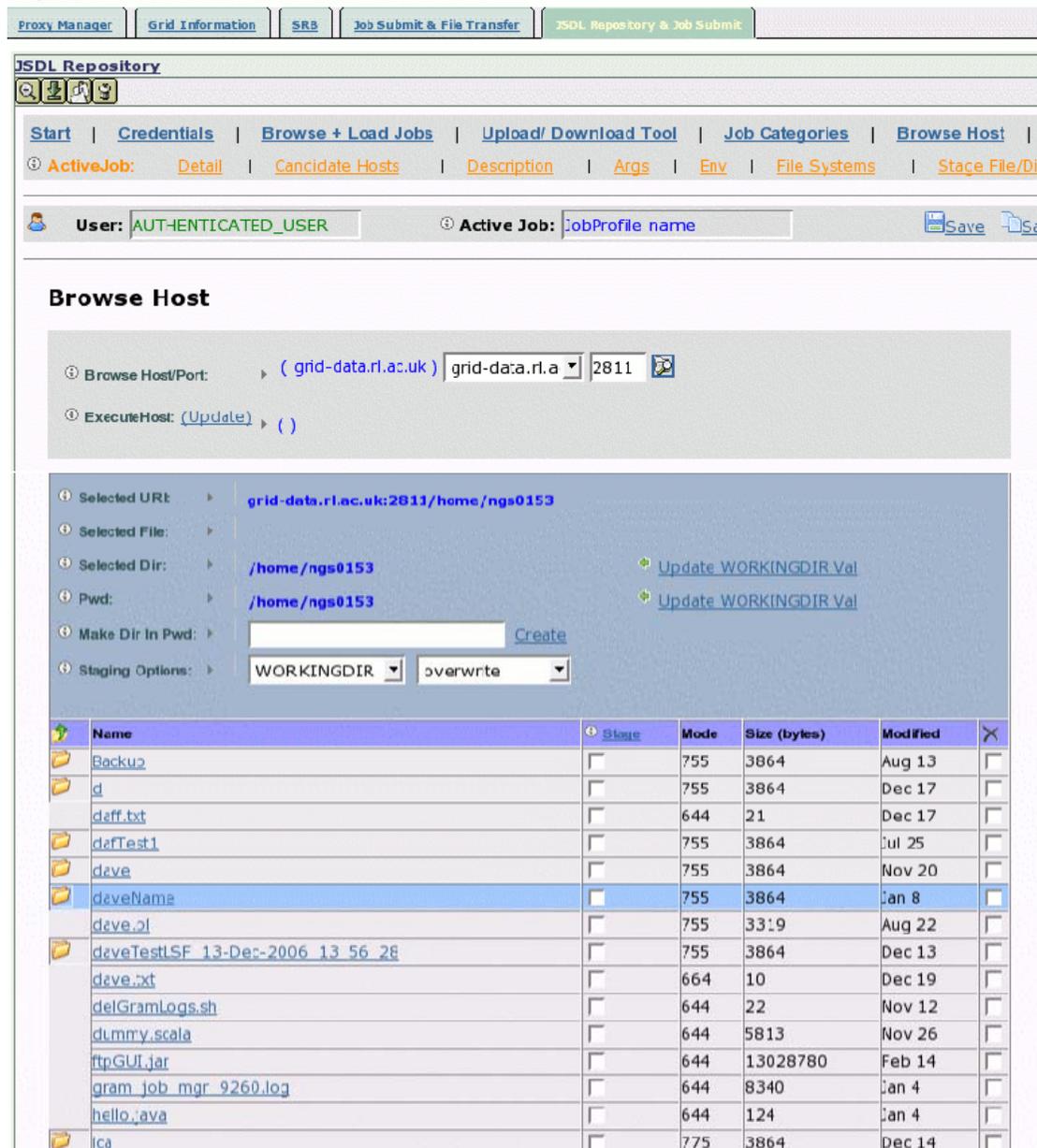
Active JobProfile Candidate Host List

Active JobProfile Candidate Hosts:	
grid-data.rl.ac.uk	<input type="checkbox"/>
grid-compute.leeds.ac.uk	<input type="checkbox"/>

Browse Hosts

Browse remote Grid hosts for data required for staging.

Select files and directories that should be copied to the consuming system via GsiFtp (more protocols to be supported inc srb, webdav).

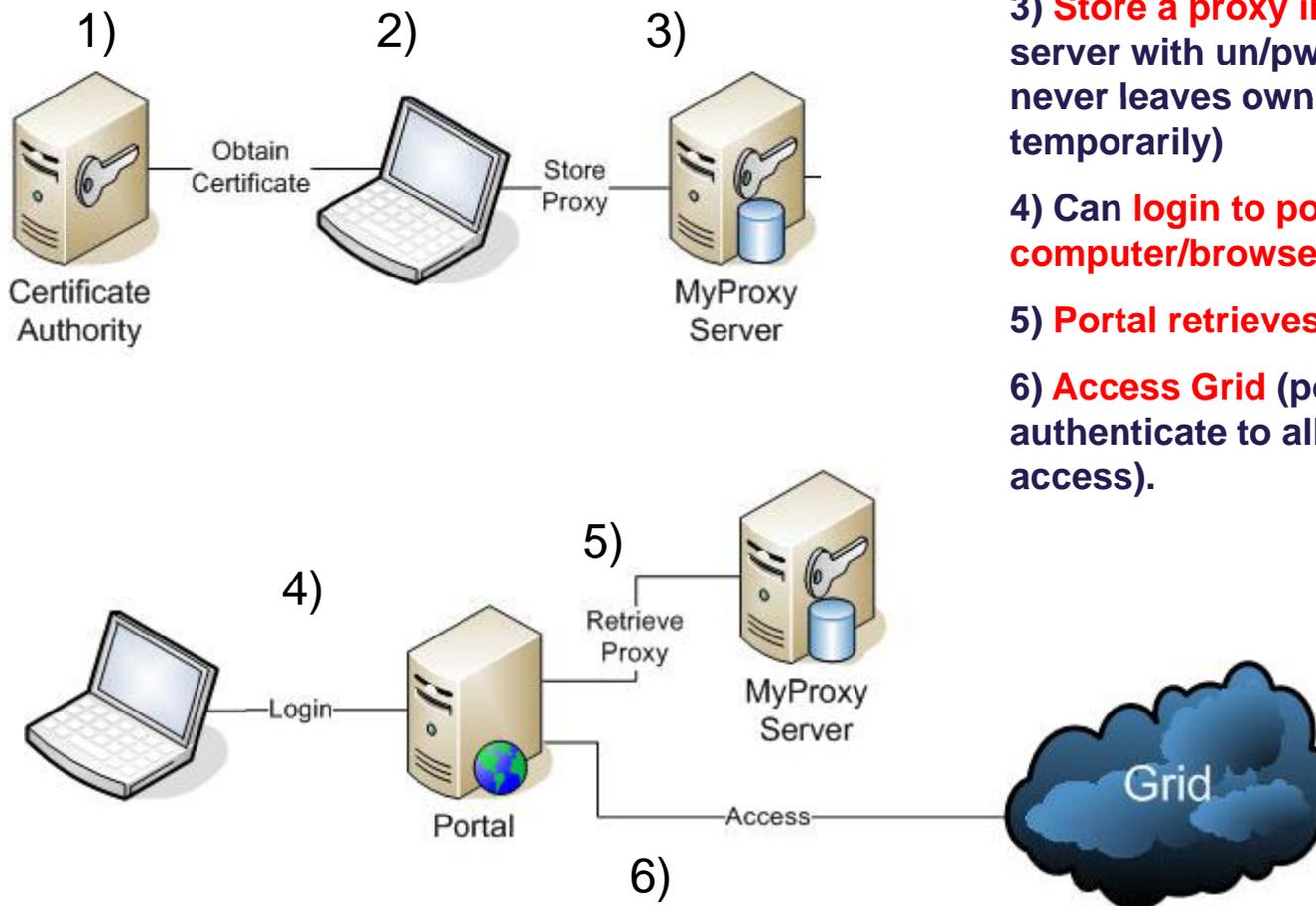


The screenshot shows the 'Browse Host' section of the JSDL Repository. At the top, there are navigation tabs: Proxy Manager, Grid Information, SRB, Job Submit & File Transfer, and JSDL Repository & Job Submit. Below these are various links: Start, Credentials, Browse + Load Jobs, Upload/ Download Tool, Job Categories, and Browse Host. A user information bar shows 'User: AUTHENTICATED_USER' and 'Active Job: JobProfile name'. The main content area is titled 'Browse Host' and contains several form fields: 'Browse Host/Port' (set to grid-data.rl.ac.uk:2811), 'ExecuteHost' (set to ()), 'Selected URT' (set to grid-data.rl.ac.uk:2811/home/ngs0153), 'Selected File', 'Selected Dir' (set to /home/ngs0153), 'Pwd' (set to /home/ngs0153), 'Make Dir In Pwd' (with a 'Create' button), and 'Staging Options' (set to WORKINGDIR and overwrite). Below these fields is a table listing files and directories in the remote host's directory.

Name	State	Mode	Size (bytes)	Modified
Backup	<input type="checkbox"/>	755	3864	Aug 13
d	<input type="checkbox"/>	755	3864	Dec 17
daff.txt	<input type="checkbox"/>	644	21	Dec 17
daffTest1	<input type="checkbox"/>	755	3864	Jul 25
dave	<input type="checkbox"/>	755	3864	Nov 20
daveName	<input checked="" type="checkbox"/>	755	3864	Jan 8
dave.pl	<input type="checkbox"/>	755	3319	Aug 22
daveTestLSF_13-Dec-2006_13_56_28	<input type="checkbox"/>	755	3864	Dec 13
dave.txt	<input type="checkbox"/>	664	10	Dec 19
delGramLogs.sh	<input type="checkbox"/>	644	22	Nov 12
dummy.scala	<input type="checkbox"/>	644	5813	Nov 26
ftpGUI.jar	<input type="checkbox"/>	644	13028780	Feb 14
gram_job_mgr_9260.log	<input type="checkbox"/>	644	8340	Jan 4
hello.java	<input type="checkbox"/>	644	124	Jan 4
lca	<input type="checkbox"/>	775	3864	Dec 14

NGS portal – <https://portal.ngs.ac.uk>

MyProxy Login Procedure



1) **Obtain a certificate** from CA - .p12 / .pfx certificate bundle (one time only)

2) **Create a public/private key pair** - usercert.pem / userkey.pem (one time only)

3) **Store a proxy in UK eScience MyProxy** server with un/pw – delegation, private key never leaves own computer (done temporarily)

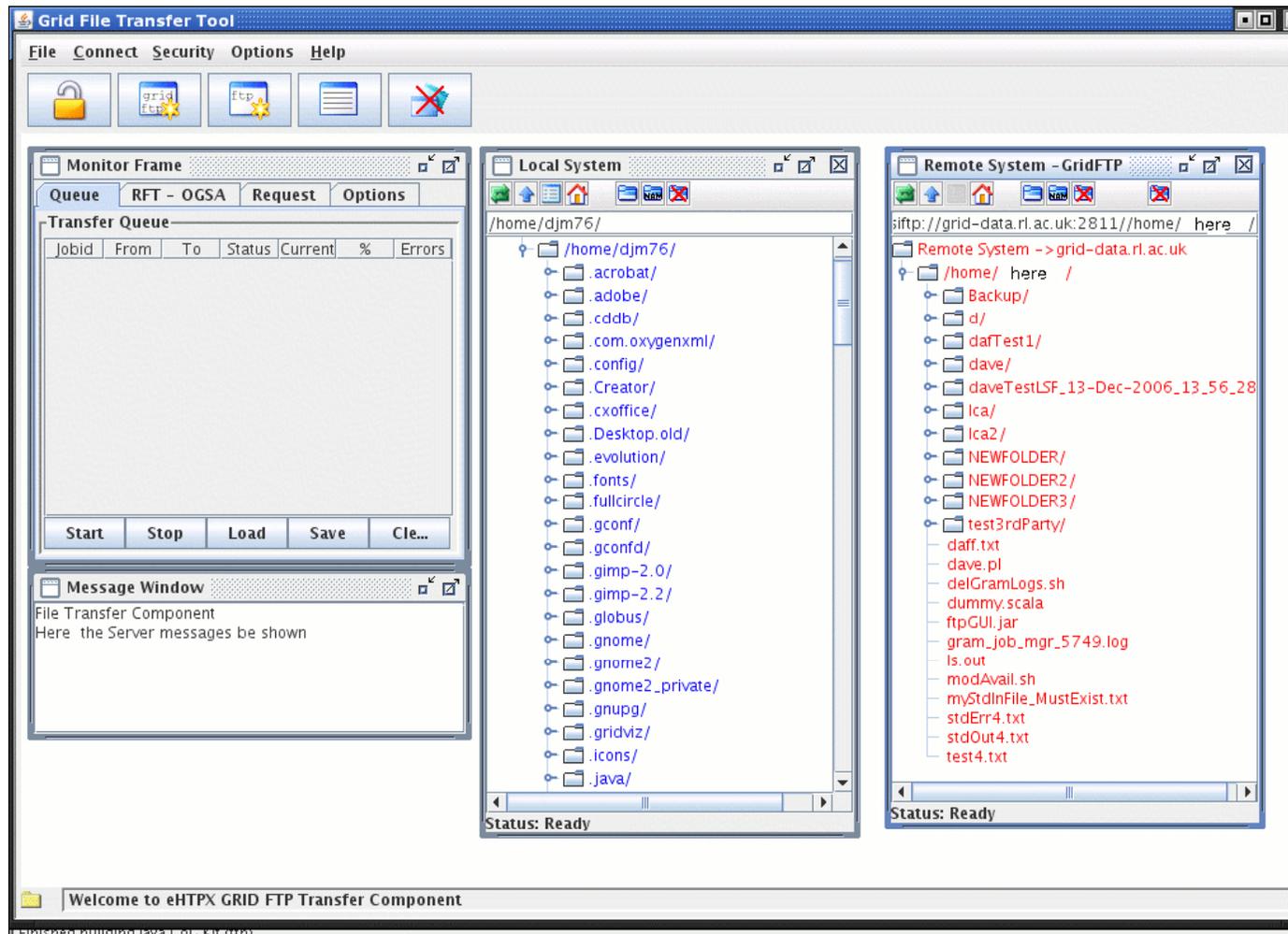
4) **Can login to portal from any computer/browser** using same un/pw.

5) **Portal retrieves proxy** using same un/pw.

6) **Access Grid** (portal uses proxy to authenticate to all Grid resources giving SSO access).

GridFtp Upload / Download / File Transfer Tool

Transfer files to/from your desktop and a remote Grid ftp server via 'drag-n-drop' – akin to Windows explorer for the Grid (in serious need of update – currently using depreciated libs).



Summary

1. Open / free to use JSDL GUI editor, browse the list of applications hosted to date. **Please contact NGS to request more hosted applications** (currently an ongoing task, more will appear soon).
2. The application is designed to be **generic and not tied to any particular set of Grid technologies** (facilitated using JSDL). Middleware dependencies added at **job-submit/monitor time when the specifics of the middleware have to be accommodated** (e.g. parsing the JSDL into RSL, adding mw-specific parameters, e.g. RSL JobType).
3. Application can be **deployed as a JSR-168 portlet or as a standalone Web application**. Portal can therefore be hosted in project specific portals.
4. Currently, application only supports Globus, but GridSAM and gLite JDL (NGS RB) are scheduled for addition (more on GridSAM next few days).

TODO / Future

- **Extend portal to accommodate more/new middleware**, e.g. NGS gLite resource broker and JSDL conversion to JDL.
- **Parametric jobs** (depends partly on parametric schema extensions for JSDL and subsequent middleware support).
- Extend application to support **staging from more Data Grid + Web protocols (SRB)**. This will involve browsing / interacting with different data storage resources within the interface (e.g. browse SRB). Performing staging across different protocols adds some complexity.
- Growing list of improvements / suggestions to refine interface
- **Release the application** for use in other projects / Grids (scheduled for OMII open source release).
- Grid will probably move to a more SOA style architecture – more use of temporary ‘sandboxes’ for job execution rather than provision of permanent user accounts on compute resources (staging become more important).

*** A portal is only as good as the underlying deployed infrastructure.... portal development often involves debugging the underlying consuming systems and middleware ***

Software Stack

1. **JSF (Java Server Faces)** for server application interface development: GUI interface components + MVC controller (Http session and request scope data). Vanilla JSF is JSR-168 compliant so can host applications as Web app or portlet.
2. **Spring v2.0 for managing objects in an n-tier server** application (recommended in a lightweight non J2EE application server, e.g. Tomcat/Jetty), e.g. managing singleton + prototype object graphs, declarative transaction demarcation, data source management, propagation of persistence context across long running transactions.
3. **C3p0** db connection pooling.
4. **JPA (Java persistence API) for ORM (object relational mapping)**. Hibernate 3.2 for domain model (could use Kodo, Toplink, apache openJPA)
5. **CogKit** for Globus API from Globus Alliance.
6. **Xml-to-object data binding framework** – Apache XMLBeans (or JAX-B).