



Enabling Grids for E-science

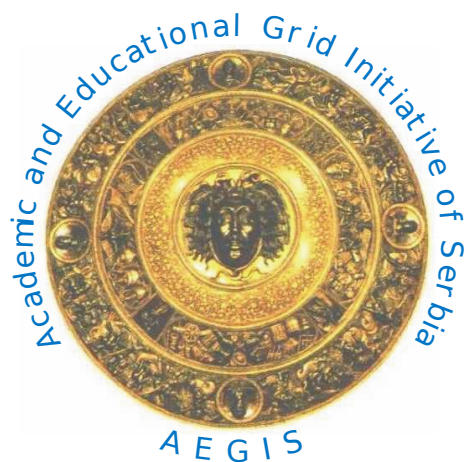
How to use computing resources at Grid

Nikola Grkic

ngrkic@ipb.ac.rs

Scientific Computing Laboratory

Institute of Physics Belgrade, Serbia



SEE-GRID-SCI
SEE-GRID infrastructure for regional eScience



Information Society



Nov. 03, 2009

www.eu-egee.org

- **JDL**

The Job Description Language (JDL) is a high-level language used to describe jobs and aggregates of jobs with arbitrary dependency relations

- **Simple example**

```
[
Type = "Job";
Executable = "/bin/hostname";
Arguments = "";
StdOutput = "std.out";
StdError = "std.err";
OutputSandbox = {"std.out", "std.err"};
Requirments = "";
]
```

• Additional attributes

```

InputSandbox = {"test.sh", "fileA", "fileB", ...}
InputSandbox = {
  "gsiftp://lxb0707.cern.ch/cms/doe/data/fileA", "fileB"};
VirtualOrganisation = "cms";
RetryCount = 0;
MyProxyServer = "myproxy.ipb.bg.ac.rs"
  
```

• Requirements

```

Requirements = RegExp("ce64.ipb.bg.ac.rs*", other.GlueCEUniqueID);
Requirements = Member("VO-cms-CMSSW_2_0_0",
  other.GlueHostApplicationSoftwareRunTimeEnvironment);
Requirements = (other.GlueHostArchitecturePlatformType ==
  "x86_64");
  
```

- **Single Job Submission**

```
glite-wms-job-list-match -a <jdl file>  
glite-wms-job-delegate-proxy -d <delegID>  
glite-wms-job-submit -a <jdl file>  
glite-wms-job-status <jobID>  
glite-wms-job-cancel <jobID>  
glite-wms-job-output <jobID>  
glite-wms-job-logging-info <jobID>
```

- **Simple job example**

```
Executable = "test.sh";  
Arguments = "fileA fileB";  
StdOutput = "std.out";  
StdError = "std.err";  
InputSandbox = {"test.sh", "fileA", "fileB"};  
OutputSandbox = {"std.out", "std.err"};
```

http://wiki.ipb.ac.rs/index.php/Submitting_jobs

- **Job collection**

Job1.jdl

```
Executable = "/bin/hostname";  
StdOutput = "std.out";  
StdError = "std.err";  
OutputSandbox = {"std.out", "std.err"};
```

Job2.jdl

```
Executable = "/bin/date";  
StdOutput = "std2.out";  
StdError = "std2.err";  
OutputSandbox = {"std2.out", "std2.err"};
```

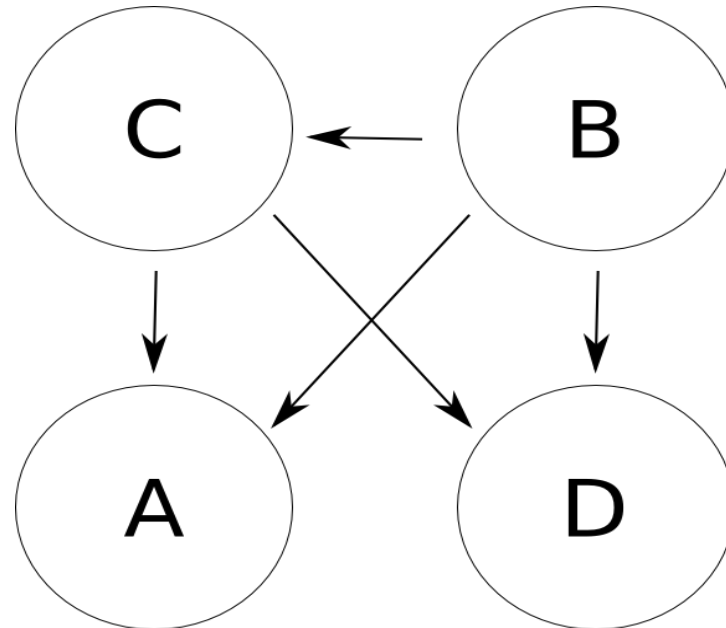
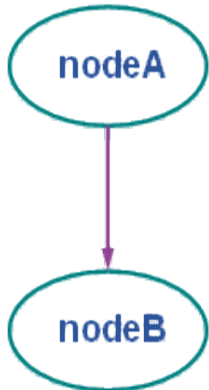
- **Parametric Job**

```
[
  JobType = "Parametric";
  Executable = "/bin/sh";
  Arguments = "message_PARAM_.sh";
  InputSandbox = "message_PARAM_.sh";
  Parameters= 6;
  ParameterStep = 2;
  ParameterStart = 0;
  StdOutput = "myoutput_PARAM_.txt";
  StdError = "myerror_PARAM_.txt";
  OutputSandbox = {"myoutput_PARAM_.txt",
  "myerror_PARAM.txt"};
  ShallowRetryCount = 1;]
]
```

- **DAG (directed acyclic graph) Jobs**

Dependencies =

$\{\{nodeA,nodeB\},\{nodeA,nodeC\},\{\{nodeB,nodeC\},nodeD\}$



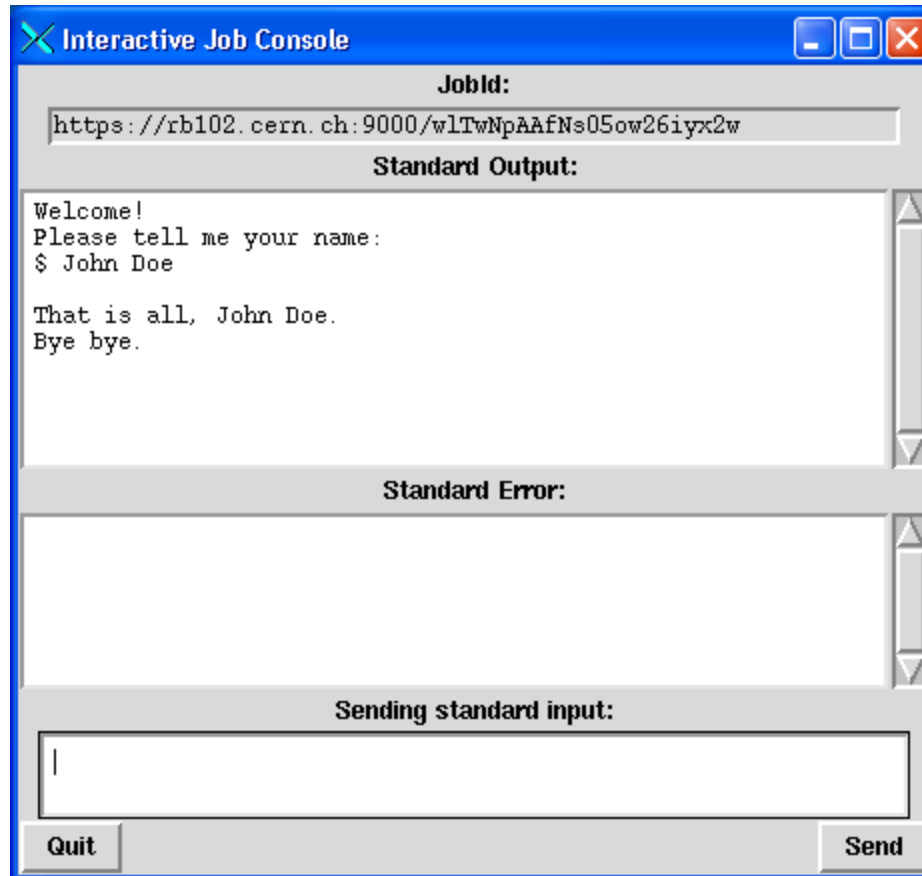
• Interactive Job

```

JobType = "Interactive" ;
Executable = "interactive.sh" ;
InputSandbox = {"interactive.sh"} ;
  
```

```

#!/bin/sh
echo "Welcome!"
echo "Please tell me your name: "
read name
echo "That is all, $name."
echo "Bye bye."
read name2
echo "Go away, $name2."
exit 0
  
```



- **MPI job**

```

JobType = "Normal";
CPUNumber = 3;
Executable = "mpi-start-wrapper.sh";
Arguments = "pi MPICH";
StdOutput = "mpi-start.out";
StdError = "mpi-start.err";
InputSandbox = {"mpi-start-wrapper.sh",
"mpi-hooks.sh", "pi.c"};
OutputSandbox = {"mpi-start.err", "mpi-start.out"};
Requirements =
    Member("MPI-START",
other.GlueHostApplicationSoftwareRunTimeEnvironment)
    && Member("OPENMPI",
other.GlueHostApplicationSoftwareRunTimeEnvironment);
    
```

Previously described gLite MPI framework can be tested by the following simple example that calculates the area under the function $4/(1+x^2)$ in the interval $[0,1]$. The following figure illustrates the problem:

