

MPI jobs with gLite

Giuseppe La Rocca

INFN Catania - Italy

NA4 Generic Applications Meeting

09-11.January.2006

www.eu-egee.org

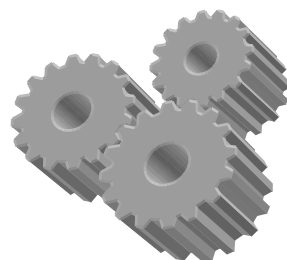


- Overview

- Requirements & Settings



- Options



- How to create a MPI job.

- MPI job in




middleware.

- MPI job in



middleware.



- Execution of parallel jobs is an essential issue for modern conceptions of informatics and application.
- 
- Most used library for parallel jobs support is **(Message Passing Interface) MPI**
 - At the state of the art, parallel jobs can run inside single Computing Elements (CE) **only**;
 - several projects are involved into studies concerning the possibility of executing parallel jobs on Worker Nodes (WNs) belonging to different CEs.



- In order to guarantee that MPI job can run, the following requirements **MUST BE** satisfied:
 - the **MPICH** software must be installed and placed in the PATH environment variable, on all the WNs of the CE.
 - Some MPI's applications required a shared filesystem among the WNs to run.
 - The variable **VO_<name_of_VO>_SW_DIR** will contain the name of a *directory* in case of **SHARED** filesystem.
 - The variable **VO_<name_of_VO>_SW_DIR** will contain "." if there is **NO SHARED** filesystem.





- The *Executable* that is specified in the JDL must not be the MPI application directly, but a *wrapper* script that invokes the MPI applications by calling `mpirun` command.

How to create a MPI Job



- For the user's point of view, jobs to be run as MPI are specified setting the JDL `JobType` attribute to **MPICH** and specifying the `NodeNumber` attribute as well.

E.g.:

```
JobType = "MPICH";
```

```
NodeNumber = 4;
```

number
needed for
application.



This attribute define
the required
of CPUs
the

- When these two attributes are included in a JDL the User Interface (UI) *automatically* add the following expression

```
(other.GlueCEInfoTotalCPUs >= NodeNumber) &&  
Member ("MPICH", other.GlueHostApplicationSoftwareRunTimeEnvironment)
```

to the JDL requirements expression in order to find out the best resource where the job can be executed.

MPI jobs with LCG middleware



- Unfortunately LCG project was not synchronized with the latter requirement *avoiding to share disk space* with nodes inside the same CE.

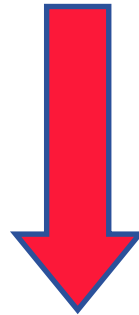


- This drove us to spend our time in providing a *ad-hoc* solution in order to find an efficient workaround to this problem.



- The solution adopted bypasses the problem by putting some intelligence inside the script passed in `Inputsandbox`.

- In detail each job has to *mirror*, via scp, its files on all nodes dedicated to it.



ssh hostbased authentication MUST BE well configured between all the WNs.

```
[  
  Type = "Job";  
  JobType = "MPICH";  
  Executable = "MPItest.sh";  
  NodeNumber = 5;  
  Arguments = "cpi 5";  
  StdOutput = "test.out"; StdError = "test.err";  
  InputSandbox = {"MPItest.sh","cpi"};  
  OutputSandbox =  
  {"test.err","test.out","executable.out"};  
  Requirements = other.GlueCEInfoLRMSType == "PBS" ||  
  other.GlueCEInfoLRMSType == "LSF";  
]
```

The number of threads specified with NodeNumber attribute agrees with the second Argument. It will be used during the invoking of mpirun command.

```

for i in `cat $HOST_NODEFILE` ; do
  echo "Mirroring via SSH to $i"
  # creates the working directories on all the nodes allocated for parallel
  # execution.
  ssh $i mkdir -p `pwd`
  # copies the needed files on all the nodes allocated for parallel
  # execution.
  /usr/bin/scp -rp .* $i:`pwd`
  # checks that all files are present on all the nodes allocated for parallel
  # execution.
  ssh $i ls `pwd`
done

# execute the parallel job with mpirun.
echo "Executing $EXE"
chmod 755 $EXE
mpirun -np $CPU_NEEDED -machinefile $HOST_NODEFILE
`pwd`/$EXE > executable.out
  
```

The Environment variable **\$HOST_NODEFILE** contains the list of WNs allocated for the parallel execution.

MPI jobs with gLite middleware



- Unlike the LCG middleware, gLite WMS is able to support both configurations (*shared* and *not shared*) automatically for both LSF and Torque.
- With gLite-1.4 **job wrapper** will take care to mirror the working directory in all nodes dedicated to the mpi job if the home are not shared.



[

JobType = "MPICH"; **NodeNumber = 6;** **Executable = "cpi";****StdOutput = "cpi.out";****StdError = "cpi.err";****InputSandbox = {"cpi"};****OutputSandbox = {"cpi.err","cpi.out"};****Requirements =****(Member("GLITE-1.4",****other.GlueHostApplicationSoftwareRunTimeEnvironment) && (other.GlueCEInfoTotalCPUs >= 10));**

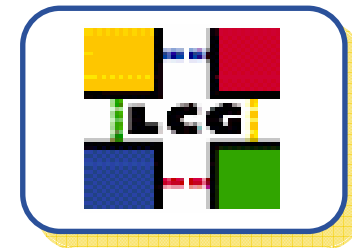
]

- The command sequence to submit this example is this one:

```
voms-proxy-init --voms gilda
```

```
edg-job-submit mpi.jdl
```

```
edg-job-status <JobID>
```



```
glite-job-submit mpi-glite.jdl
```

```
glite-job-status <JobID>
```

pi is approximately **3.1415926544231239**, Error is
0.0000000008333307

wall clock time = 10.008261

Process 0 of 6 on grid036.ct.infn.it

Process 1 of 6 on grid036.ct.infn.it

Process 2 of 6 on grid033.ct.infn.it

Process 3 of 6 on grid033.ct.infn.it

Process 5 of 6 on grid034.ct.infn.it

Process 4 of 6 on grid034.ct.infn.it

The limitation of the actual middleware, according to the RB schedule the MPI jobs only on a single CE reducing dramatically the number of WNs available for the parallel computation, can be overcome using the **CrossGrid testbed** based on LCG middleware.

To get more information you can contact **Jesus Marco** (Tuesday, 10 January 2006 arco@ifca.unican.es) or **Harald Kornmayer** (Tuesday, 10 January 2006 arald.kornmayer@iwr.fzk.de).

- **LCG-2 User Guide Manuals Series**
 - **Tuesday, 10 January 2006**
<https://edms.cern.ch/file/454439/LCG-2-UserGuide.pdf>
- **Tuesday, 10 January 2006**
<http://oscinfo.osc.edu/training/>
- **Tuesday, 10 January 2006**
<http://www.netlib.org/mpi/index.html>
- **Tuesday, 10 January 2006**
<http://www-unix.mcs.anl.gov/mpi/learning.html>

