LCG-2 from a user perspective

Leif Nixon
National Supercomputer Centre
nixon@nsc.liu.se

13 maj 2004

Getting stuff done

1. Get authenticated
2. Get authorized
3. Prepare input data
4. Create job description
5. Submit job
6. Retrieve output data

Architectural points

Jobs go through a central resource broker. No direct communication between client and computing element.

Only small amounts of data can be submitted together with the job. Any larger files *must* be placed in grid storage.

Authentication

Straight Globus style PKI.

Support for MyProxy; credential store or “wallet”

Authorization

- Register at the `lcg-registrar` site.
- Become member of a VO.

Job descriptions

Jobs are described using JDL (Job Description Language):

- **Condor ClassAds** Provides for flexible resource matching.
- **The Glue schema** Describes resources. Perhaps not undisputed.

No RSL!

Job descriptions – example

```
Executable = "runpov.sh";
StdError = "std.err";
Arguments = "101 150";
InputSandbox = {"runpov.sh", "model.pov"};
OutputSandbox = {"std.err"};
InputData = {"lfn:scene/bg.png", "lfn:scene/map.png"};
DataAccessProtocol = {"rfio", "gsiftp"};
OutputData = {
        [
         OutputFile="scene.png";
         LogicalFileName="lfn:scene/scene.png"
         StorageElement="lxshare0291.cern.ch"
        ]
};
Requirements = other.GlueCEInfoTotalCPUs > 1 &&
  Member("POVRAY",
         other.GlueHostApplicationSoftwareRunTimeEnvironment);
```

Data management

Replica Management System (RMS) by EDG.

All files are identified by a GUID:

`guid:38ed3f60-c402-11d7-a6b0-f53ee5a37e1d`

They have a number of replicas:

`sfn://tbed0101.cern.ch/flatfiles/SE00/dteam/generated/2004-02-26/file3596e86f-c402-11d7-a6b0-f53ee5a37e1d`

They can have a number of logical filenames (LFNs):

`lfn:importantResults/Test1240.dat`

Any form may be used to refer to a file.

Data management

Registering a file:

```
$ edg-rm --vo dteam cr file:///home/antonio/file1 -l lfn:my_alias1
guid:6ac491ea-684c-11d8-8f12-9c97cebf582a
```

Downloading a file:

```
$ edg-rm --vo dteam cp lfn:my_alias2 file:/home/antonio/file2
```

Job handling

Submitting a job:

```
$ edg-job-submit <jdl_file>
========================== 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 ic
 (edg_jobId) is:
 - https://lxshare0234.cern.ch:9000/rIBubkFFKhnSQ6CjiLUY8Q
====================================================================
```

Job handling

Checking status:

```
$ edg-job-status <jobId>

And an example of a possible output is

*************************************************************
BOOKKEEPING INFORMATION:

Printing status info for the Job:
https://lxshare0234.cern.ch:9000/X-ehTxfdlXxSoIdVLS0L0w

Current Status:     Ready
Status Reason:      unavailable
Destination:        lxshare0277.cern.ch:2119/jobmanager-pbs-infinite
reached on:         Fri Aug  1 12:21:35 2003
*************************************************************
```

Job handling

Downloading output:

```
$ edg-job-get-output https://lxshare0234.cern.ch:9000/snPegp1YMJcnS2

Retrieving files from host lxshare0234.cern.ch

*****************************************************************
                 JOB GET OUTPUT OUTCOME

Output sandbox files for the job:
- https://lxshare0234.cern.ch:9000/snPegp1YMJcnS22yF5pFlg
have been successfully retrieved and stored in the directory:
/tmp/jobOutput/snPegp1YMJcnS22yF5pFlg

*****************************************************************
```

Interactive jobs

A job can be of `JobType = "Interactive";`. It will be scheduled in the ordinary way, and will connect back to the client upon execution, providing access to the job's I/O streams, either as named pipes, or through a GUI terminal application.