Overview of WP1 Workload Management System in EDG 2.x

Massimo Sgaravatto
INFN Padova - DataGrid WP1

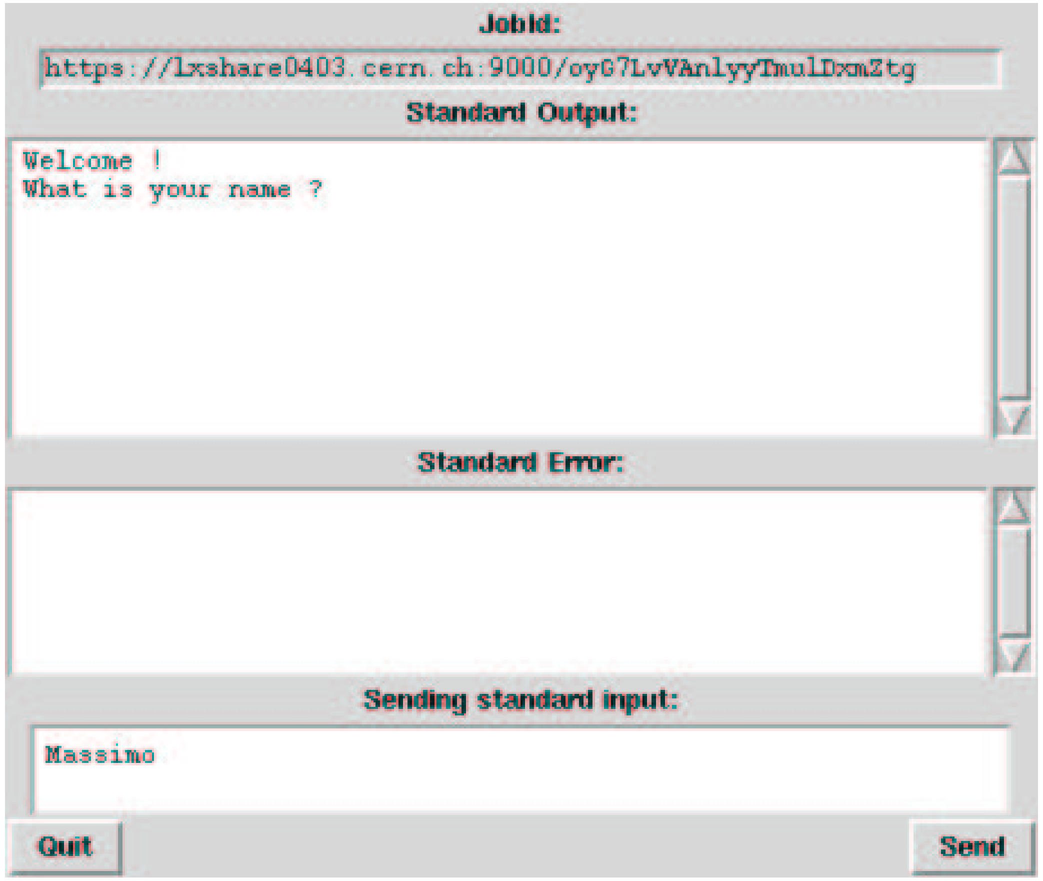
massimo.sgaravatto@pd.infn.it

WMS release 2

- Addressed reliability and scalability problems seen with EDG rel. 1.x
- Many many improvements (in design and implementation) and bug fixes
- New functionalities introduced
  - User APIs
  - GUI
  - Support for interactive jobs
  - Job checkpointing
  - Support for parallel jobs
  - Gangmatching
  - Support for automatic output data upload and registration
  - …

Interactive jobs

- Specified setting JobType = "Interactive" in JDL
- When an interactive job is executed, a window for the stdin, stdout, stderr streams is opened
  - Possibility to send the stdin to the job
  - Possibility the have the stderr and stdout of the job when it is running
- Possibility to start a window for the standard streams for a previously submitted interactive job with command edg-job-attach



```
https://lxshare0403.cern.ch:9000/oyG7LvVAnlyyTmulDxmZtg
```

```
Welcome !
What is your name ?
```

```
Massimo
```

Job checkpointing

- Checkpointing: saving from time to time job state
  - Useful to prevent data loss, due to unexpected failures
  - Approach: provide users with a "trivial" logical job checkpointing service
  - User can save from time to time the state of the job (defined by the application)
  - A job can be restarted from an intermediate (i.e. "previously" saved) job state
- Different than "classical checkpointing (i.e. saving all the information related to a process: process's data and stack segments, open files, etc.)
  - Very difficult to apply (e.g. problems to save the state of open network connections)
  - Not necessary for many applications
- To submit a checkpointable job
  - Code must be instrumented (see next slides)
  - JobType=Checkpointable to be specified in JDL

Job checkpointing example

```
int main ()
{
  …
  for (int i=event; i < EVMAX; i++)
       {  < process event i>;}
             ...
exit(0); }
```

Example of
Application
(e.g. HEP MonteCarlo
simulation)

Job checkpointing example

```
#include "checkpointing.h"

int main ()
{ JobState state(JobState::job);
   event = state.getIntValue("first_event");
   PFN_of_file_on_SE = state.getStringValue("filename");
   ….
   var_n = state.getBoolValue("var_n");
   < copy file_on_SE locally>;
  …
  for (int i=event; i < EVMAX; i++)
      {  < process event i>;
          ...
          state.saveValue("first_event", i+1);
         < save intermediate file on a SE>;
          state.saveValue("filename", PFN of file_on_SE);
           ...
           state.saveValue("var_n", value_n);
          state.saveState(); }
  …
  exit(0); }
```

User code
must be easily
instrumented in order
to exploit the
checkpointing
framework …

Job checkpointing example

```
#include "checkpointing.h"

int main ()
{ JobState state(JobState::job);
    event = state.getIntValue("first_event");
    PFN_of_file_on_SE = state.getStringValue("filename");
    ….
    var_n = state.getBoolValue("var_n");
    < copy file_on_SE locally>;
  …
  for (int i=event; i < EVMAX; i++)
       {  < process event i>;
            ...
            state.saveValue("first_event", i+1);
           < save intermediate file on a SE>;
            state.saveValue("filename", PFN of file_on_SE);
             ...
             state.saveValue("var_n", value_n);
            state.saveState(); }
  …
  exit(0); }
```

- User defines what is a state
- Defined as <var, value> pairs
- Must be “enough” to restart a computation from a previously saved state

Job checkpointing example

```
#include "checkpointing.h"

int main ()
{ JobState state(JobState::job);
    event = state.getIntValue("first_event");
    PFN_of_file_on_SE = state.getStringValue("filename");
    ….
    var_n = state.getBoolValue("var_n");
    < copy file_on_SE locally>;
  …
  for (int i=event; i < EVMAX; i++)
      {  < process event i>;
          ...
          state.saveValue("first_event", i+1);
         < save intermediate file on a SE>;
          state.saveValue("filename", PFN of file_on_SE);
           ...
           state.saveValue("var_n", value_n);
          state.saveState(); }
  …
  exit(0); }
```

User can save from time to time the state of the job

Job checkpointing example

```
#include "checkpointing.h"

int main ()
{ JobState state(JobState::job);
    event = state.getIntValue("first_event");
    PFN_of_file_on_SE = state.getStringValue("filename");
    ….
    var_n = state.getBoolValue("var_n");
    < copy file_on_SE locally>;
  …
  for (int i=event; i < EVMAX; i++)
       {  < process event i>;
            ...
            state.saveValue("first_event", i+1);
           < save intermediate file on a SE>;
            state.saveValue("filename", PFN of file_on_SE);
             ...
             state.saveValue("var_n", value_n);
            state.saveState(); }
  …
  exit(0); }
```

Retrieval of the last saved state
The job can restart from that point

Job checkpointing scenarios

- Scenario 1
  - Job submitted to a CE
  - When job runs it saves from time to time its state
  - Job failure, due to a Grid problem (e.g. CE problem)
  - Job resubmitted by the WMS possibly to a different CE
  - Job restarts its computation from the last saved state
    - → No need to restart from the beginning
    - → The computation done till that moment is not lost
- Scenario 2
  - Job failure, but not detected by the Grid middleware
  - User can retrieve a saved state for the job (typically the last one)
    - *edg-job-get-chkpt –o <state><edg-jobid>*
  - User resubmits the job, specifying that the job must start from a specific (the retrieved one) initial state
    - *edg-job-submit –chkpt <state> <JDL file>*

Submission of parallel jobs

- Possibility to submit MPI jobs
- MPICH implementation supported
- Only parallel jobs inside a single CE can be submitted
- Submission of parallel jobs very similar to normal jobs
  - Just needed to specify in the JDL:
    - JobType= “MPICH”
    - NodeNumber = n;
      - The number (*n*) of requested CPUs
- Matchmaking
  - CE chosen by RB has to have MPICH sw installed, and at least *n* total CPUs
  - If there are two or more CEs satisfying all the requirements, the one with the highest number of free CPUs is chosen

Gangmatching

- Allow to take into account both CE and SE information in the matchmaking
- For example to require a job to run on a CE close to a SE with at least 200 MB of available space:

```
Requirements = anyMatch(other.storage.CloseSEs,
  target.GlueSAStateAvailableSpace > 200);
```

Output data registration

```
OutputData = {
           [
             OutputFile = "filename1";
             LogicalFileName = "lfn:mylfn1";
             StorageElement = "testbed007.cnaf.infn.it"
           ],
           [
             OutputFile = "filename2"
           ],
           [
            OutputFile = "filename3";
            LogicalFileName = "lfn:mylfn2"
           ],
           [
            OutputFile = "filename4";
            StorageElement = "testbed007.cnaf.infn.it"
           ]
}
```

Both LFN and target SE specified

Nor LFN nor target SE specified

Only LFN specified

Only target SE specified

WP1 releases

WP1 RPMs	Release date
2.0.15	28 Jul 2003

EDG 2.0.x
LCG-1

WP1 releases

Fix for proxy renewal with short proxies

Fix for verbosity in log files

Fix for EDG_LOCATION setting (needed for interactions with WP2 Services)

Man pages

…

WP1 RPMs	Release date
2.0.15-1	28 Jul 2003
2.0.16-1	8 Aug 2003

EDG 2.0.x
LCG-1

WP1 releases

Fix for LM DB corruption causing LM and JC failures (#1807)

Fix for ‘done’ job status declared too early (#1808)

Fix for automatic output data upload and registration

…

WP1 RPMs	Release date
2.0.15-1	28 Jul 2003
2.0.16-1	8 Aug 2003
2.0.17-1	25 Aug 2003

EDG 2.0.x
LCG-1

WP1 releases

WP1 RPMs	Release date
2.0.15-1	28 Jul 2003
2.0.16-1	8 Aug 2003
2.0.17-1	25 Aug 2003
2.0.17-2	9 Sep 2003

EDG 2.0.x
LCG-1

Minor fix for automatic output data upload and registration

WP1 releases

Fix for gangmatching

Fix for log rotation

Fix for proxy renewal db corruption

Much more useful error messages (no more "an helper failed")

Fix for CEId in JDL causing WM crash

Fix for matchmaking problems when multiple closeSEs for a given CE

Fix for edg-Brokerinfo getDataAccessProtocol

…

WP1 RPMs	Release date
2.0.15-1	28 Jul 2003
2.0.16-1	8 Aug 2003
2.0.17-1	25 Aug 2003
2.0.17-2	9 Sep 2003
2.0.18-1	17 Sep 2003

EDG 2.0.x

Deployed in EDG dev testbed

Under deployment by LCG

WP1 releases

WP1 RPMs	Release date
2.0.15-1	28 Jul 2003
2.0.16-1	8 Aug 2003
2.0.17-1	25 Aug 2003
2.0.17-2	9 Sep 2003
2.0.18-1	17 Sep 2003
2.1.0	Very soon !

VOMS support

Support for ACLs in L&B (to set who can query the status of a given job)

Fix for matchmaking when attributes belonging to Subcluster objectclass used as rank

…

EDG 2.0.x

Deployed in EDG dev testbed

Under deployment by LCG

EDG 2.1.x

VOMS support

- VO taken from VOMS user proxy
- Matchmaking performed wrt VO
  - → Not necessary to publish anymore in the information service the list of authorized users (only list of authorized VOs needed)
- WMS works also with non-VOMS proxies
  - If not possible to get VO from proxy, VO taken:
    - from JDL (*VirtualOrganisation* attribute), if specified
    - from *--vo VO* option, if specified
    - from UI conf file
  - Matchmaking: if no matches wrt VO, matchmaking tried wrt user DN

Other issues raised by applications

- "Cryptic" error messages
  - Much more useful error messages with WP1 RPMs >= 2.0.18
    - No more "cannot plan (an helper failed)"
    - Examples of new error messages when jobs get aborted:
      - *Cannot plan: BrokerHelper: Problems querying the information service bbq.mi.infn.it*
      - *Cannot plan: BrokerHelper: The user is not authorized on any resource currently registered in bbq.mi.infn.it*
      - *Cannot plan: BrokerHelper: no compatible resources*
      - *Cannot plan: BrokerHelper: All compatible resources are unavailable (problems during rank evaluation)*
    - … and these are not failures in the WMS software …
  - Going to do the same for *edg-job-list-match*

Other issues raised by applications

- BrokerInfo responsibility
  - Writing and reading (edg-brokerinfo and brokerinfo APIs) now completely up to WP1
    - In the past brokerinfo reading was a WP1-WP2 joint effort
  - Now 1 open issue: "replacement for old getSelectedFile"
- "RB queries all suitable CEs"
  - This means that only CEs which match all the requirements are queried
  - Needed to see if the CE is still available
  - Needed to get fresher info than the one published in GOUT/bdII

Other issues raised by applications

- "Matchmaking: with multiple files will match on 1 even if the others are not available"
  - Nothing new: same algorithm used in release 1
  - Algorithm: choose the CE which meet all the requirements, and where most of the files are close to it
  - The application is then supposed to copy locally/access remotely the non-closed files
- Outbound connectivity for the WNs
  - We don't think it is a real problem …
  - … in any case we are going to evaluate a different mechanism to transfer sandbox files between WN and RB, for which outbound connectivity is not required for WNs anymore
  - Outbound connectivity still required for interactive jobs
  - What about other services (e.g. to copy files from/to a remote SE) ??

Other issues raised by applications

- "Cannot read JobWrapper output, both from Condor and from Maradona" problem
  - Background
    - The stdout of the script where the user job is wrapped around is transferred via Globus GASS from the CE node to the RB machine in order to check if the job was successfully executed
    - Many reasons for the "Cannot read …" problem found and addressed in release 1, and fixes are also in release 2
      - Not problems in WP1 software, but mostly addressed by WP1
    - In order to reduce the rate of failures, a modification was introduced in the WMS software, in order to transfer the standard output of job wrapper also via gridftp, and not only via GASS: Maradona
  - This problems now appears only from time to time and only on some sites
    - Local configuration problems ??
  - Warning: the problem appears if OutputData used in the JDL with WP1 RPMs < 2.0.17 (the bug has then been fixed)

Conclusions

- Thanks very much for your feedback
  - Your inputs have been always valuable and contributed to improve the EDG WMS
- Hope you are enough happy with what we provided and with our level of support
- Waiting for feedback on some new functionalities that seem not to have been tried by the applications yet
  - Many of them were not in the initial WP1 agenda, but they were addressed as requested by applications
    - E.g. support for interactive jobs, strongly asked by WP8
    - E.g. support for MPI jobs, requested by WP9 and WP10
- WP1 next future (up to the end of the project)
  - Support and bug fixes
  - Refinements and new functionalities (e.g. DAGMan)
    - Detailed plans to be discussed at the WP1 meeting here in Heidelberg