Job Submission

J.J.Blaising – CNRS/IN2P3

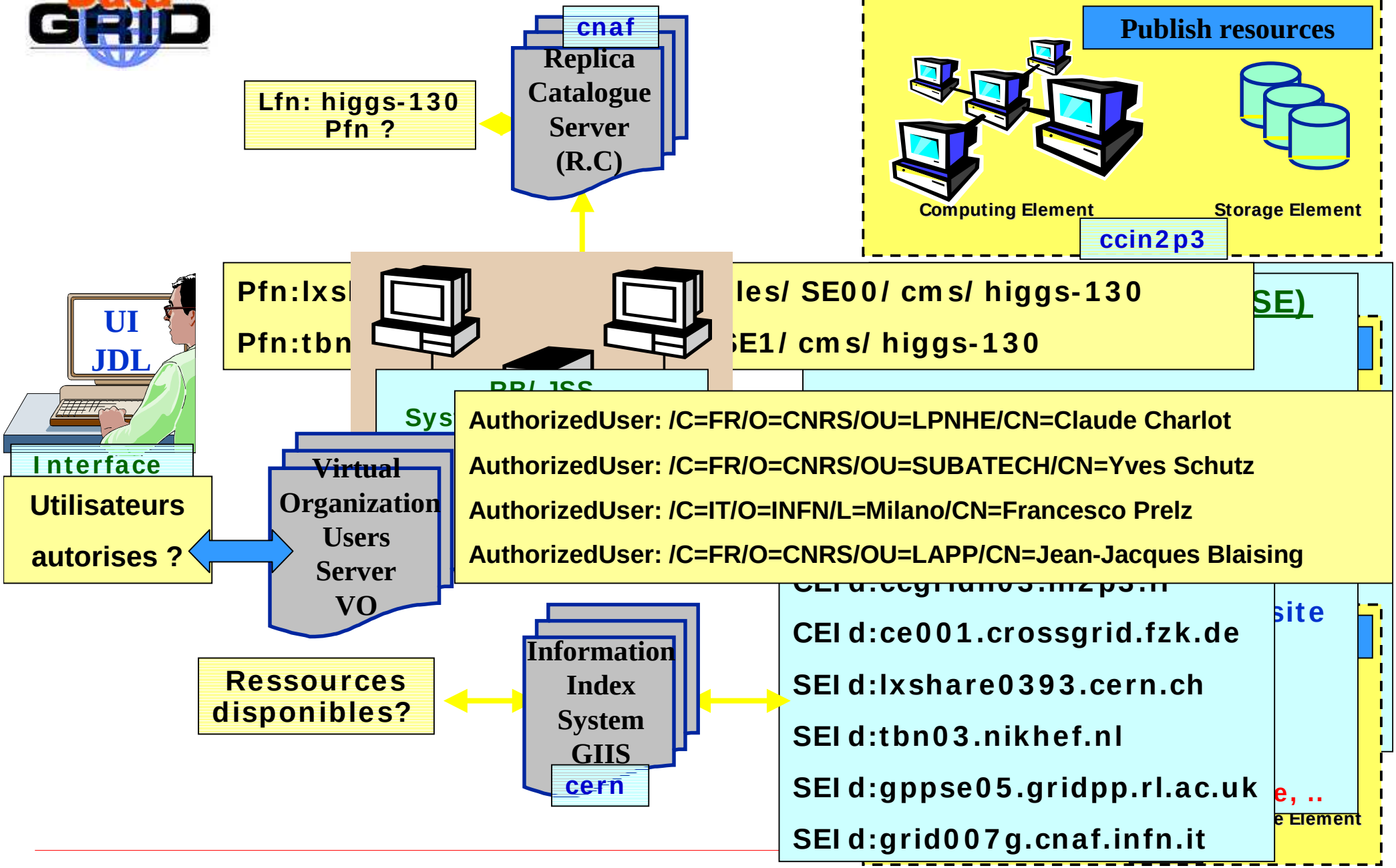
The European DataGrid Project Team

http://www.eu-datagrid.org

Contents

- **Grid Services description**
- **EDG Job management**
  - The EDG Workload Management System (WMS)
  - Job Preparation
  - Job Description Language (JDL)
  - WMS match making
  - Grid registration
  - Job Submission & Monitoring commands
  - Job Execution BrokerInfo file
- **A simple program example: the job lifecycle**
- **Job examples (Hello World, Tuto)**

Description des Services



The EDG WMS

- The user interacts with Grid via a **Workload Management System**
- It performs the **job management in a Grid environment**
- **It allow Grid users to:**
  - submit their jobs
  - execute them
  - get information about their status
  - retrieve their output
- **The WMS tries to optimize the job execution according to the user's request and the available resources**

WMS Components

- WMS is currently composed of the following parts:
  1. **User Interface** (UI): access point for the user to the GRID
  2. **Resource Broker** (RB): the broker of GRID resources, performing the match-making
  3. **Information Index** (II): a specialized Globus GIIS (LDAP server) used by the Resource Broker as a filter to the information service (IS) to select resources
  4. **Job Submission System** (JSS): provides submission functionality
  5. **Logging and Bookkeeping** services (LB): store Job Info available for users to query

Job Preparation:

In order to allow the WMS to manage a job, the system uses

- **The User requirements:**

  Program name

  Input data name, data access protocole (optional)

  Computing system (OS version, MaxCpuTime,)

  Experiment software (Run time environment, ie ATLAS software version x.y installed)

- **The computing and storage resources informations**

  Dynamic information (cpu, disk)

Job Description Language (JDL) 1/5

- Based upon Condor's *CLASSified ADvertisement language (ClassAd)*
- ClassAd is a fully extensible language
- ClassAd is constructed with the classad construction operator []

  It is a sequence of attributes separated by semi-colons. An attribute is a pair (key, value), where value can be a Boolean, an Integer, a list of strings, …

  <attribute> = <value>;

  ie , Executable = "tuto.csh";

  Requirements = Member(other.RunTimeEnvironment,ATLAS-3.2.1);

The JDL allows to define a set of attribute, the WMS takes into account when making its scheduling decision and the job wrapper.

Job Description Language (JDL) 2/5

- The supported attributes are grouped in three categories:
  - *Job (Attributes)*

    provided by the user while he/she edits job description file, split up into:

    Mandatory

    Mandatory with default value

    Define the job, inserted by the UI into the wrapper before submitting the job
  - *Computing Resource (Attributes)*

    used to build expressions of Requirements and Rank attributes by the user

    Taken into account by the RB for carrying out the matchmaking algorithm

    have to be prefixed with "other."
  - *Data and Storage resources (Attributes)*

    input data to process, SE where to store output data, protocols spoken by application when accessing the SEs.

Job Description Language (JDL) 3/5: job attributes

- Mandatory for every single JDL file:
  1. *Executable* (ie, **Executable = "tuto.csh";**)
- Mandatory for JDL file dealing with Data Management:
  2. *InputData* (Defines the LFN or PFN, **InputData = "LF:rwd_123456.dat";** of data used as input by the job, they are published in the VO Replica Catalog and stored in the Se.
  2. *ReplicaCatalog* (contains the VO Replica Catalog Identifier)
  3. *DataAccessProtocol*(contains the protocolor the list of protocols which the application is able to speak with for accessing *InputData* on a given SE)

  If *InputData* contains at least one PFN and no LFNs, only *DataAccessProtocol* is mandatory.

  If *InputData* contains at least one LFN, both *ReplicaCatalog* and *DataAccessProtocol* are mandatory.

Job Description Language (JDL) 4/5: resource attributes

- Mandatory attributes with default value for every single JDL file:

  1. Rank (contains a ClassAd Floating Point expression)

  The default value is -other.EstimatedTraversalTime.

  2. Requirements (contains a ClassAd Boolean expression)

  The default value is other.Active.

  The default value of these attributes are in the edg user interface configuration file, /opt/edg/etc/UI-ConfigENV.cfg .

  \ is the line continuation character

  Special characters or quoted strings are allowed in the Arguments attribute as long as they are between \' and \'

  Arguments = "\'+%d-%m-%Y\'"; (ie format of date)

Job Description Language (JDL) 5/5: other attributes 5/5

- Others:
  - OutputSE (contains the Resource Identifier of the SE)

    RB uses it to choose a CE that is compatible with the job and is close to SE.

    OutputSE="lxshare0393.cern.ch";
  - InputSandbox (list of files on the UI localdisk needed by the job for running

    InputSandbox = {"home/blaising/grid/tuto/tuto.csh"};

    The files are staged from the UI to the remote CE.
  - OutputSandbox (list of files, generated by the job, which have to be retrieved)

    StdError = "tuto.err";

    StdOutput = "tuto.out";

    OutputSandbox = {"tuto.out", "tuto.err", "BrokerInfo.lis"};

Example JDL File

```
Executable = “tuto.csh”;

InputData = “LF:rwd_123456.dat”;

ReplicaCatalog = “ldap://grid-vo.nikhef.nl:10389/ \

 lc=EDGtutorial WP1 RepCat, rc=EDGtutorial dc=eu-datagrid, dc=org”;

DataAccessProtocol = “gridftp”;

StdError  = “tuto.err”;

StdOutput = “tuto.out”;

OutputSandbox = {“tuto.err”, “tuto.out”};

InputSandbox = {“home/blaising/grid/tuto/tuto.csh”};

Rank = “other.FreeCPUs >= 1”;

Requirements = other.Architecture==“INTEL” && \
               other.OpSys==“RH 6.2” && other.MaxCPUTime >18000;
```

WMS Match Making 1/4

- The RB is the core component of WMS.
- It has to choose the CE where the job will be executed
- It interacts with Data Management service (file catalogue) and Information Service

  They supply the RB with the information required for the resolution of the matches
- The CE chosen by RB matches the job requirements (e.g. runtime environment, data access and resources requirements)

WMS Match Making 2/4

- The RB deals with three possible submission scenarios.

1. Scenario : Direct Job Submission (not useful for tuto)

- Job is scheduled on a given CE (specified in the dg-job-submit command via –r option)

```
dg-job-submit –o JobList.lis \
–r lxshare0227.cern.ch:2119/jobmanager-pbs-short \
tuto.jdl
```

- RB doesn't perform any matchmaking algorithm and no BrokerInfo file is produced

WMS Match Making 3/4

2. Scenario : Job Submission without data-access Requirements

- CE is not specified in the JDL
- RB starts the matchmaking algorithm, which consists of two phases:
  - Requirements check (RB contacts the IS to find a set of the suitable CEs)
  - Rank computation (RB acquires information from these suitable CEs)

If more than one CE satisfies the job requirements, the CE with the best rank is chosen by the RB

If the user doesn't specify any rank value, by default the RB considers resources with the lowest estimated traversaltime

If allCEs have the same rank value, the RB chooses the first CE in the list

WMS Match Making 4/4

3. Scenario : Job Submission with data-access Requirements

- CE is not specified in the JDL
- RB interacts with Data Management service to select the CE taking into account also the SEs where both input data sets are physically stored and output data sets should be staged on completion of job execution
- RB strategy consists of submitting jobs close to data
- The main two phases of the match making algorithm remain unchanged:
  - Requirements check
  - Rank computation
- What changes with respect to the second scenario?

  Now, the RB executes the two phases for each class of CEs that satisfy the data-access requirements (ie. which are close to data)

Proxy Creation, Renewal

- **Create a proxy**
  - grid-proxy-init -hours <hours> (default 12 hours)
- **WMS support automatic proxy renewal as long as the user credentials are handled by a proxy server.**
  - Register this proxy with the MyProxy server using (Safe)

    myproxy-init –s <server> [-t <cred> -c <proxy>]

    server is the server address (e.g. lxshare0375.cern.ch)

    cred is the number of hours the proxy should be valid on the server

    proxy is the number of hours renewed proxies should be valid

Avoid job failure because it exceeds the validity of the initial proxy, security

The Proxy is automatic renewed by WMS without user intervention for all the Job life

Job management UI Commands

- **dg-job-list-match job.jdl**

  lists resources matching a job description (quick to check the JDL and the RB status)

- **dg-job-submit**

  submits a job

- **dg-job-cancel**

  cancels a job

- **dg-job-status**

  displays the status of the job (submitted, waiting, ready, scheduled, running, …)

- **dg-job-get-output**

  returns the job-output to the user

- **dg-job-get-logging-info**

  displays logging information about submitted jobs

- **dg-job-xxxxx CR -> Function Usage**

Example of Command Options

- **dg-job-submit -r** *<res_id>* **-n** *<user e-mail address>* **-c** *<config file>* **-o** *<output file>* ***<job.jdl>***

  -r the job is submitted by the RB directly to the computing element identified by *<res_id>* no BrokerInfo file is created (don't use this option for the tuto).

  -n an e-mail message containing basic information regarding the job (status and identification) is sent to the specified *<e-mail address>* when the job enters one of the following status: Ready, Running, Done or Aborted

  -c the configuration file *<config file>* is pointed by the user instead of the standard configuration file

  -o the generated dg_jobId is written in the *<output file>*

- **dg-job-status -i** *<input file>* (or dg_jobId) -o <job-stat file>

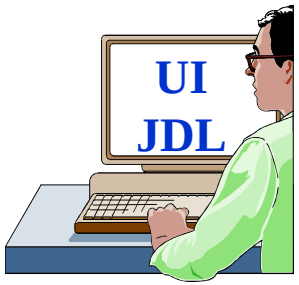
  -i display the job status information about dg_jobId contained in the *<input file>*

.BrokerInfo File

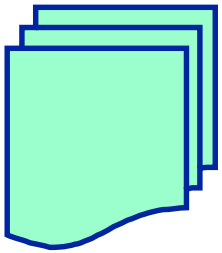
- **File created by the WMS in the temp directory of the WN where the job executes.** Provides informations useful for data management using the edg-brokerinfo api's (ie, edg-brokerinfo getCloseSEs).

```
[
 SEs ={};
 CE = "lxshare0227.cern.ch:2119/jobmanager-pbs-medium";
 SEProtocols ={};
 SEPorts = {};
 CloseSEs ={   "lxshare0393.cern.ch"};
 InputPFNs ={};
 SEMountPoint ={ "/flatfiles/SE00"};
 LFNs ={};
  DataAccessProtocol ={};
]
```

A Job Submission Example



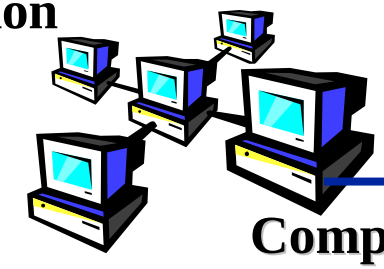
Replica Catalogue (RC)



Information Service (IS)

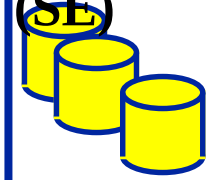
Resource Broker (RB)

Job Submission Service (JSS)



Compute Element CE)

Storage Element (SE)

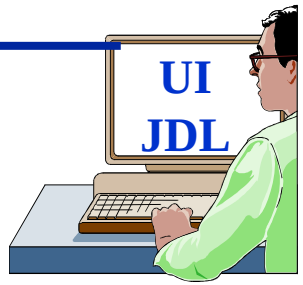


Logging & Book-keeping (LB)

A Job Submission Example

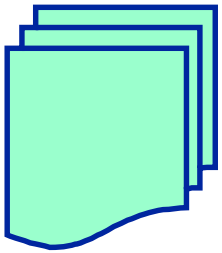
Job Status

submitted



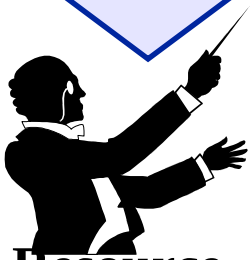
Input Sandbox & JDL

Replica Catalogue (RC



Information Service (IS)

Job Submit Event

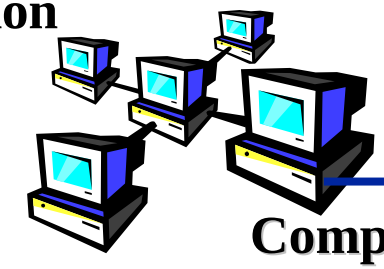


Resource Broker (RB)

Logging & Book-keeping (LB)

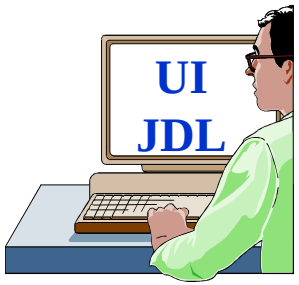
Job Submission Service (JSS)

Storage Element (SE)

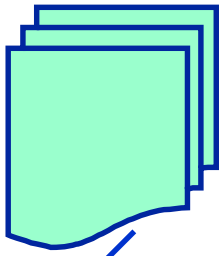


Compute Element (CE)

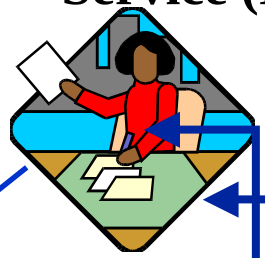
A Job Submission Example



Replica Catalogue (RC)

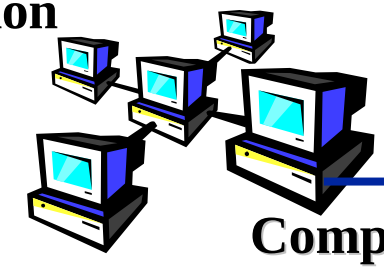


Information Service (IS)

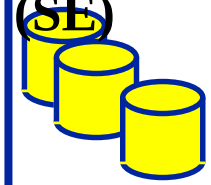


Resource Broker (RB)

Job Submission Service (JSS)



Compute Element (CE)

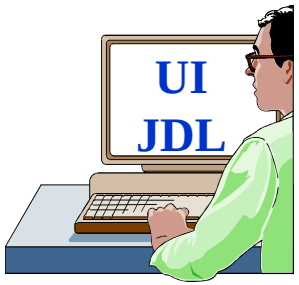


Storage Element (SE)

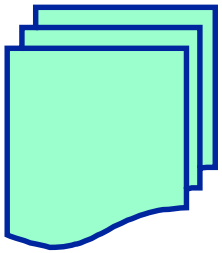
Logging & Book-keeping (LB)

Job Status

A Job Submission Example

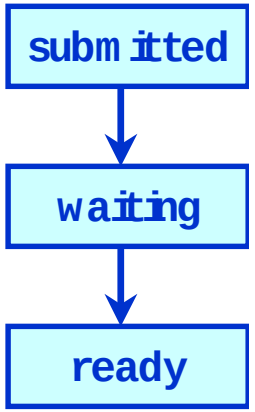


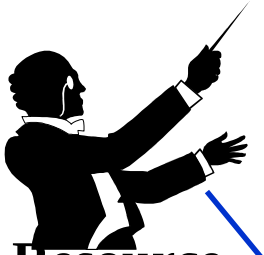
Replica Catalogue (RC)



Information Service (IS)

Job Status

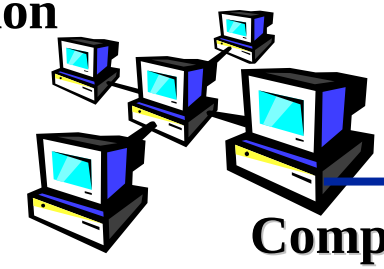




Resource Broker (RB)

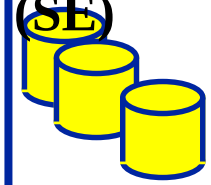
Logging & Book-keeping (LB)

Job Submission Service (JSS)

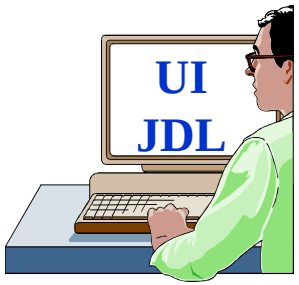


Compute Element (CE)

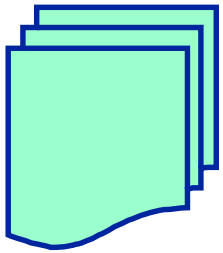
Storage Element (SE)



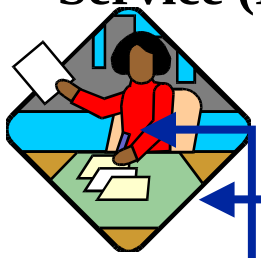
A Job Submission Example



Replica Catalogue (RC)



Information Service (IS)

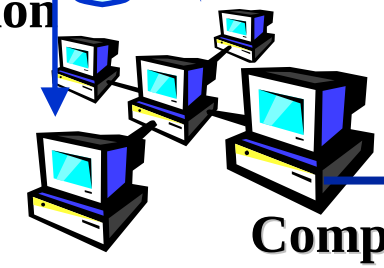


Resource Broker (RB)

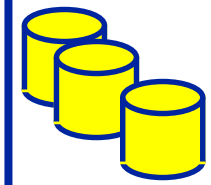
Logging & Book-keeping (LB)

Job Submission Service (JSS)

BrokerInfo



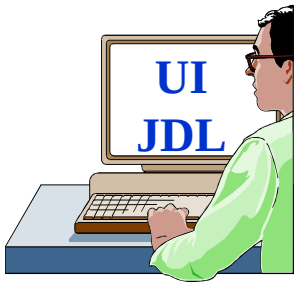
Compute Element (CE)



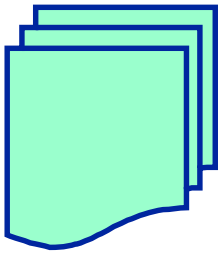
Storage Element (SE)

Job Status

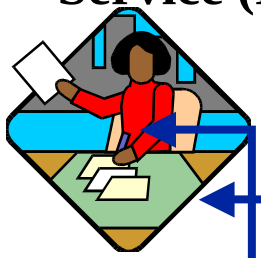
A Job Submission Example



Replica Catalogue (RC)



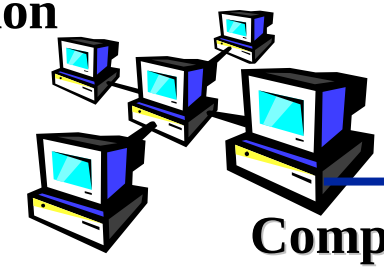
Information Service (IS)



Resource Broker (RB)

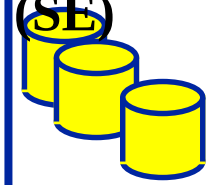
Input Sandbox

Job Submission Service (JSS)



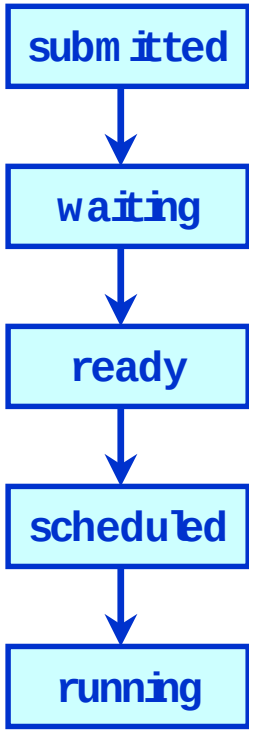
Compute Element (CE)

Storage Element (SE)

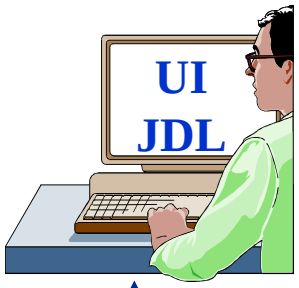


Logging & Book-keeping (LB)

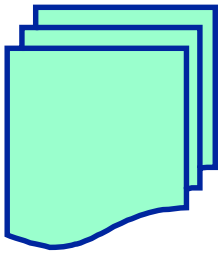
Job Status



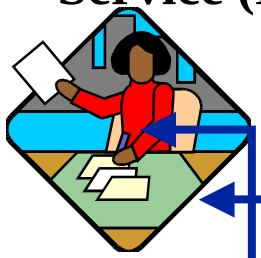
A Job Submission Example



Replica Catalogue (RC)



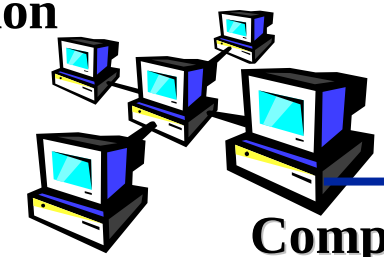
Information Service (IS)



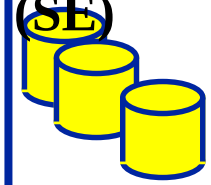
Resource Broker (RB)

Job Submission Service (JSS)

Logging & Book-keeping (LB)



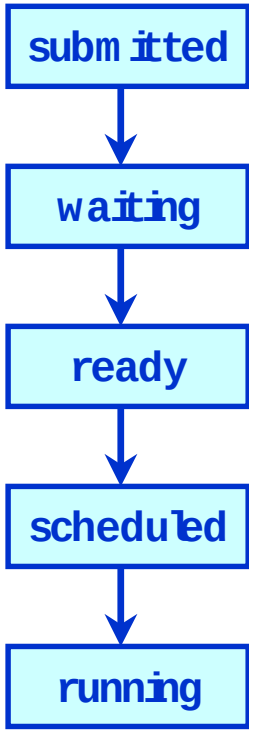
Compute Element (CE)



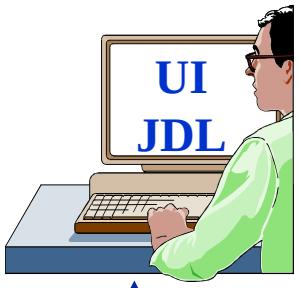
Storage Element (SE)

Job Status

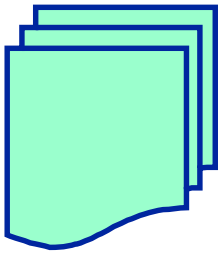
Job Status



A Job Submission Example



Replica Catalogue

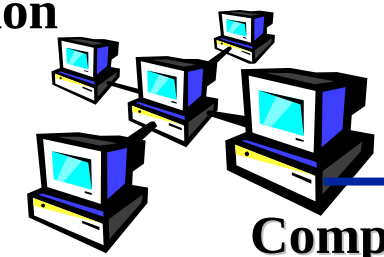


Information Service

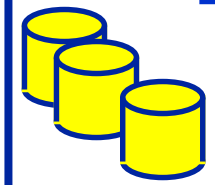
Resource Broker

Job Submission Service

Logging & Book-keeping

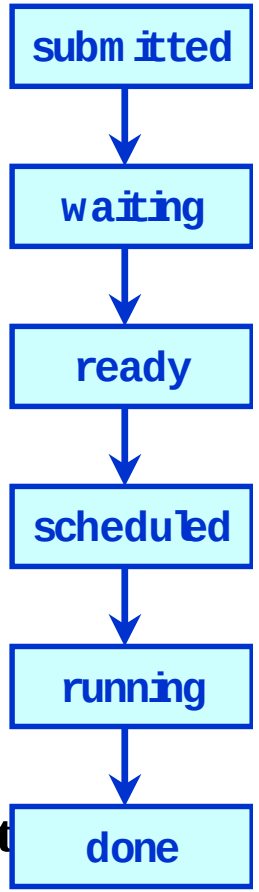


Compute Element

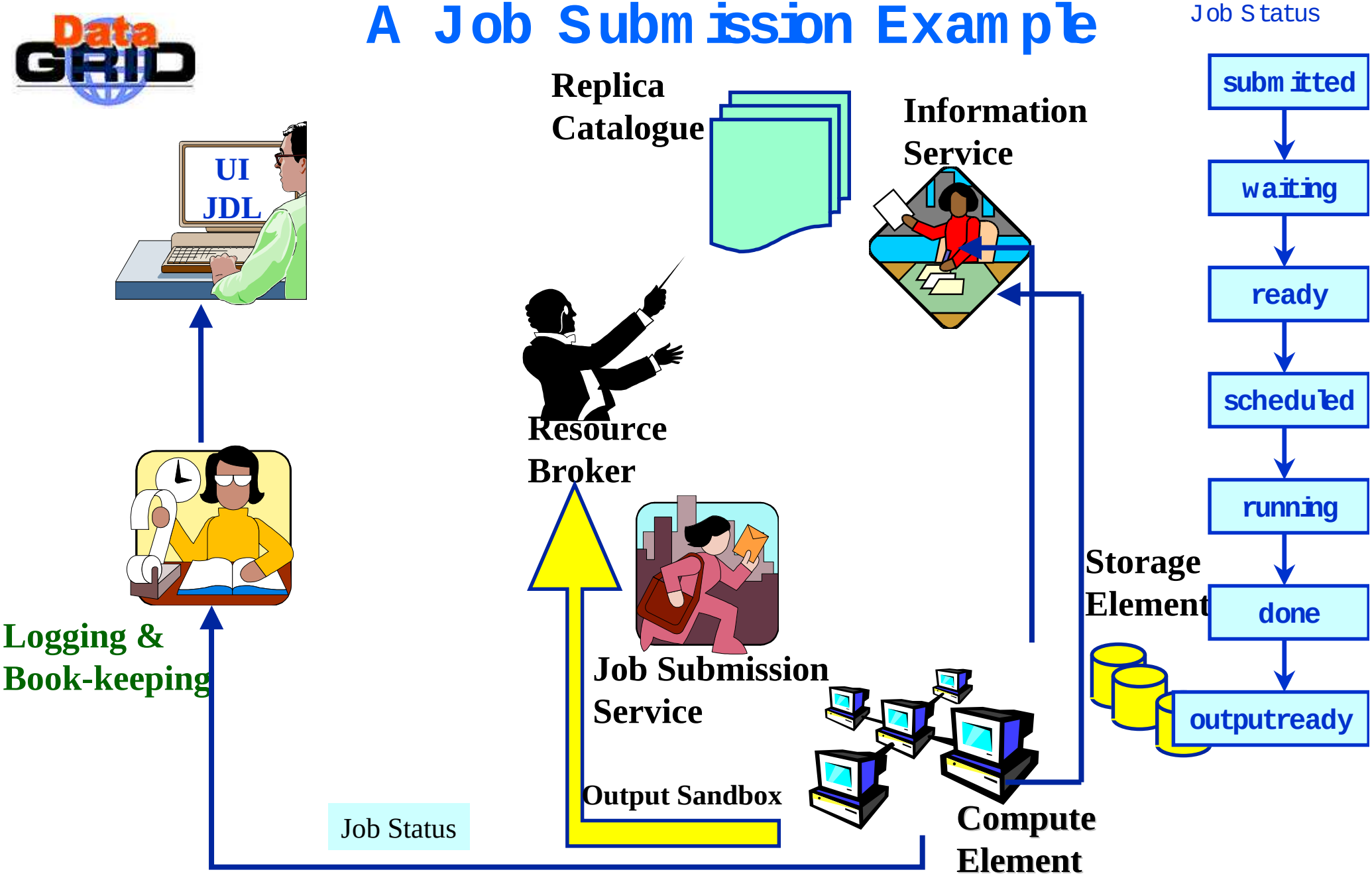


Storage Element

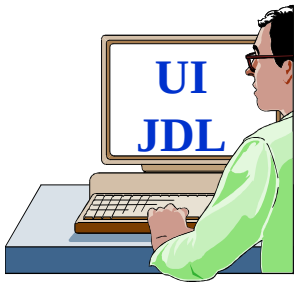
Job Status



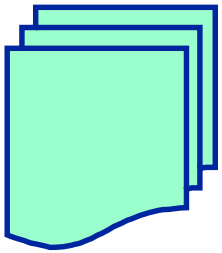
A Job Submission Example



A Job Submission Example



Replica Catalogue (RC)



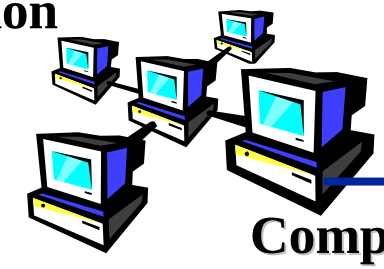
Information Service (IS)

Output Sandbox

Resource Broker (RB)

Logging & Book-keeping (LB)

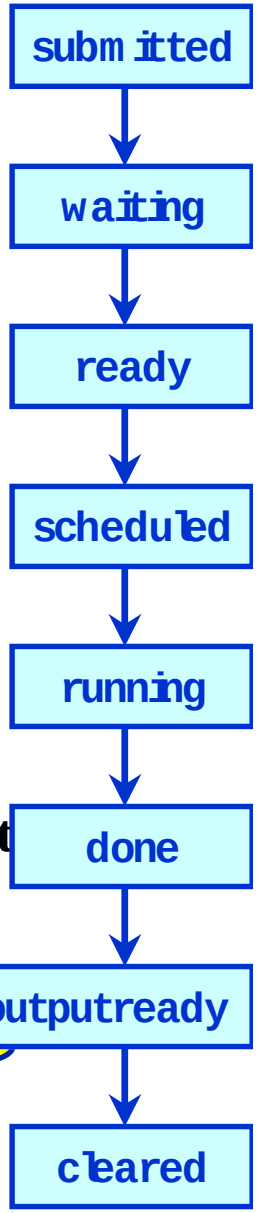
Job Submission Service (JS)



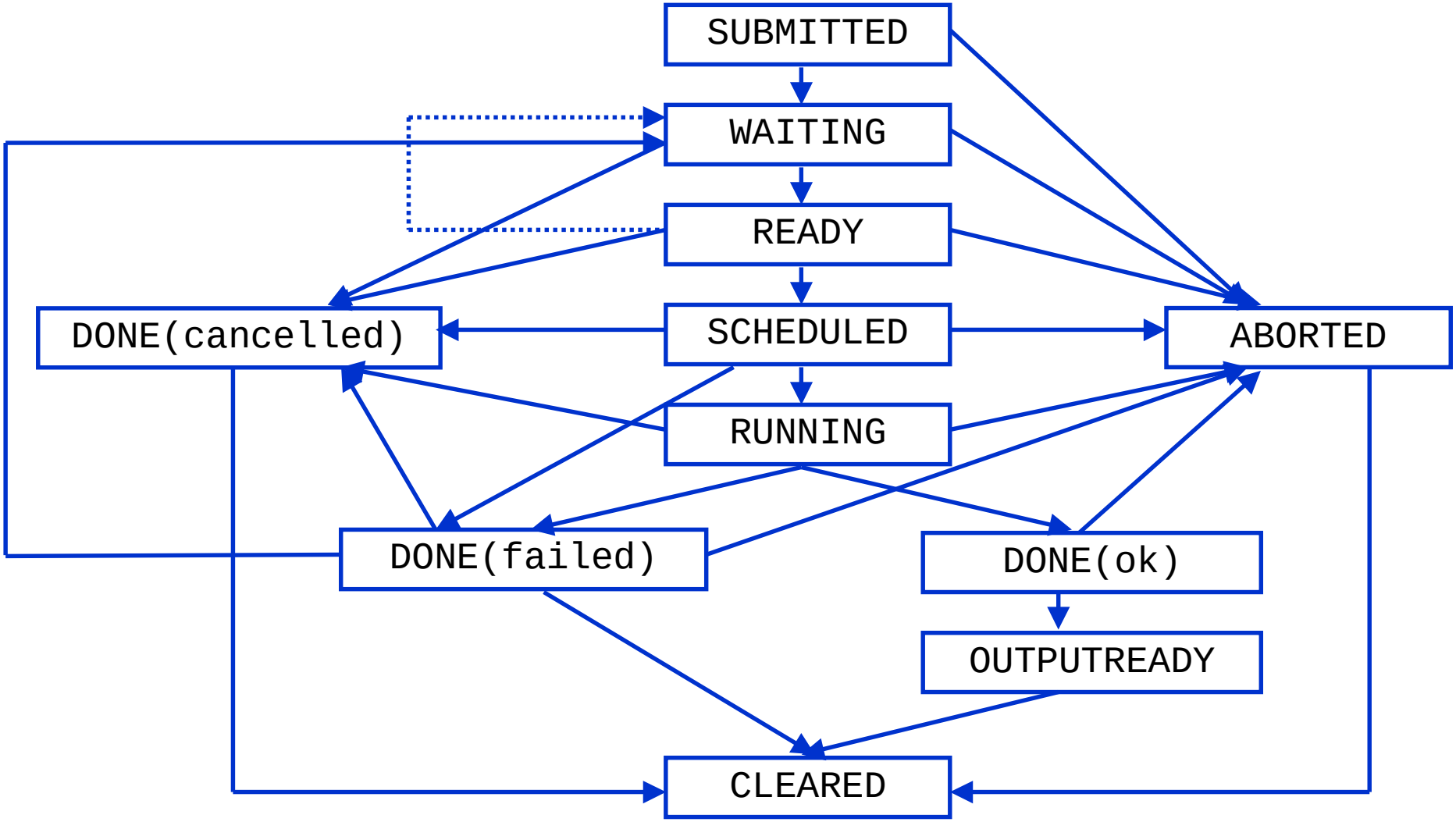
Compute Element (CE)

Storage Element (SE)

Job Status



Possible Job States



Job Submission Hello World(1)

- User logs in on the UI
- User issues a *grid-proxy-init* and enters his certificate's password, getting a valid Globus proxy
- User sets up his or her JDL file
- Example of Hello World JDL file :

```
Executable = "/bin/echo";
Arguments = "Hello World";
StdOutput = "Messagge.txt";
StdError = "stderr.log";
OutputSandbox = {"Message.txt","stderr.log"};
```

Job Submission HelloWorld(2)

- *dg-job-submit HelloWorld.jdl*

  submits job and gets back from the system a unique Job Identifier (JobId)

- *dg-job-submit -o JobIdList HelloWorld.jdl*

  same but write the JobId in the file JobidList

- d*g-job-status JobId*

- d*g-job-status -i JobIdList -o JobStatusFile*

  gets information about the current status of the Job

- When the "OutputReady" status is reached, the user can issue a *dg-job-get-output JobId*

  write all the ouput files in a temporary directory and return its name

  dg-job-get-output -iJobIdList -dir Dirname

  write all the ouput files in the directory Dirname

Job Submission tuto (1)

- **tuto JDL file :**

```
# File tuto.jdl
Executable    = "tuto.csh";
Arguments     = "BrokerInfo.lis";
StdOutput     = "tuto.out";
StdError      = "tuto.err";
InputSandbox  = {"/home/blaising/grid/tuto/tuto.csh"};
OutputSandbox = {"tuto.out","BrokerInfo.lis",\
"tuto.err"};
Requirements  = other.CEID == "lxshare0227.cern.ch:2119/jobmanager-pbs-short";
```

Job Submission tuto (2)

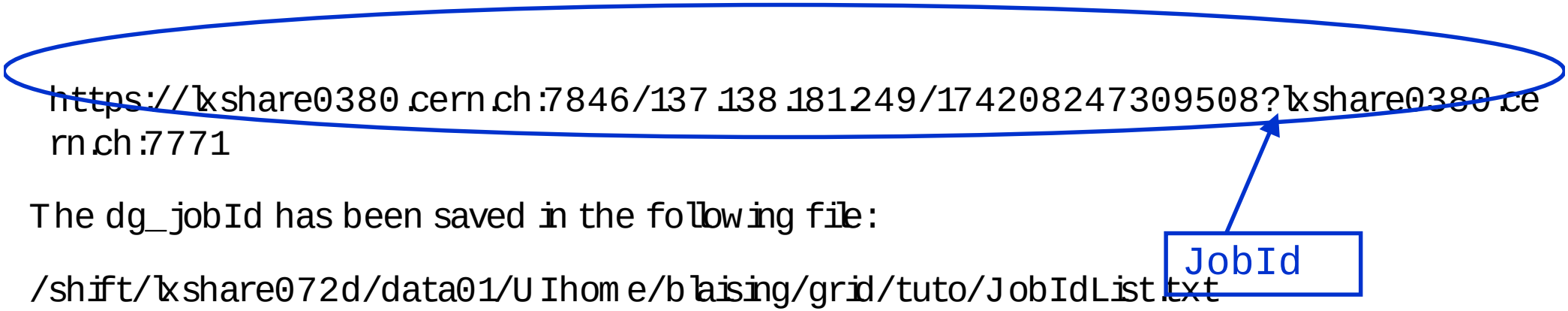
- **tuto.csh file :**

```
# !/bin/csh -f
echo "*+ EXECUTE $0 on:$HOST"
set lfname=$1
if( -f BrokerInfo ) then
  cat BrokerInfo > BrokerInfo.lis
  set CloseSE=`edg-brokerinfo getCloseSEs`
  set MountP=`edg-brokerinfo getSEMountPoint $CloseSE`
  echo "*+ CloseSE:$CloseSE, MountP:$MountP"
  echo "*+ rfdir $CloseSE":"$MountP"
  rfdir $CloseSE":"$MountP
endif
```

Job Submission tuto (3)

dg-job-submit -o JobIdList.txt tuto.jdl

```
Connecting to host lxshare0380.cern.ch, port 7771

Logging to host lxshare0380.cern.ch, port 15830

========================================== dg-job-submit Success ==================

 The job has been successfully submitted to the Resource Broker.

 Use dg-job-status command to check job current status. Your job identifier (dg_jobId)
  is:

  https://lxshare0380.cern.ch:7846/137.138.181.249/17420824730950 8?lxshare0380.ce
  rn.ch:7771

 The dg_jobId has been saved in the following file:

 /shift/lxshare072d/data01/UIhome/blaising/grid/tuto/JobIdList.txt
```



Job Submission tuto (4)

dg-job-status -i JobIdList.txt -o Jobstatus.txt

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

BOOKKEEPING INFORMATION:

Printing status info for the Job : https://lxshare0380.cern.ch:7846/137.138.181.249/1754572520557 81?
lxshare0380.cern.ch:7771

Some bookkeeping information has not reached the LB server yet.

Missing information should come from GlobusJobmanager

dg_JobId
  =https://lxshare0380.cern.ch:7846/137.138.181.249/1754572520557 81?lxshare0380.cern.ch:7771

Status            =    OutputReady

Last Update Time (UTC) =    Tue Jan 28 17:55:50 2003

Job Destination       =    lxshare0227.cern.ch:2119/jobmanager-pbs-short

Status Reason         =    terminated

Job Owner             =    /C=FR/O=CNRS/OU=LAPP/CN=Jean-Jacques Blaising/Email=blaising@lapp.in2p3.fr

Status Enter Time (UTC)=    Tue Jan 28 17:55:50 2003
```

Job Submission tuto (5)

testbed010.cern.ch 285: dg-job-get-output -i JobIdList.txt -dir output

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

                    JOB GET OUTPUT OUTCOME

 Output sandbox files for the job:

https://lxshare0380.cern.ch:7846/137.138.181.249/175457252055781?lxshare0380.cern.
  ch:7771

 have been successfully retrieved and stored in the directory:
  /shift/lxshare072d/data01/UIhome/blaising/grid/tuto/output/175457252055781

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

Job Submission tuto (6)

- tuto.out file :

```
*+ EXECUTE ./tuto.csh on: adc0037.cern.ch
*+ CloseSE: lxshare0393.cern.ch, MountP: /flatfiles/SE00
*+ rfdir lxshare0393.cern.ch:/flatfiles/SE00
drwxrwsr-x  2 osynge  root      4096 Dec 09 14:37 .
drwxrwsr-x  6 osynge  alice     4096 Jan 28 18:11 alice
drwxrwsr-x 12 osynge  atlas    20480 Jan 28 18:11 atlas
drwxrwsr-x 22 osynge  cms       4096 Jan 28 18:11 cms
drwxrwsr-x 11 osynge  lhcb      4096 Jan 28 18:11 lhcb
drwxrwsr-x 11 osynge  biome     4096 Jan 28 18:11 biome
drwxrwsr-x  7 osynge  eo       49152 Jan 28 18:11 eo
drwxrwsr-x  6 osynge  wpsix     4096 Jan 28 18:11 wpsix
drwxrwsr-x  3 osynge  iteam     4096 Jan 28 18:11 iteam
drwxrwsr-x  2 osynge  tutor     4096 Jan 28 18:11 tuto
```

Further Information

- The EDG User's Guide

  http://marianne.in2p3.fr

- WMS and JDL

  http://www.infn.it/workload-grid

- ClassAd

  https://www.cs.wisc.edu/condor/classad

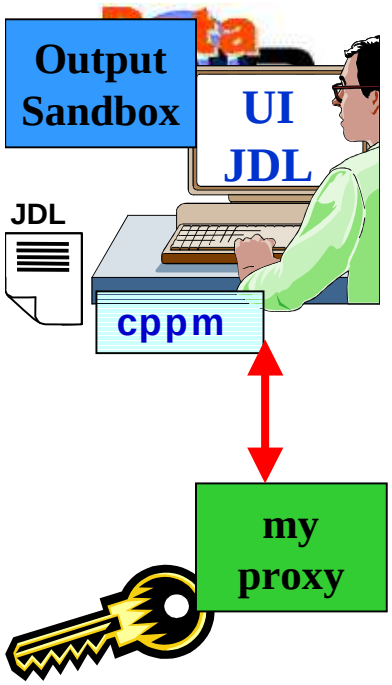
Common Error Messages 1/2

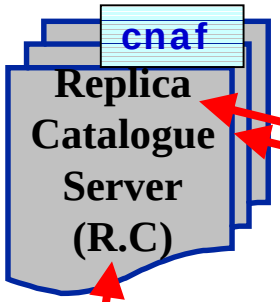
- The UI commands accept some arguments in input. If the user makes a mistake via command line, the following messages can appear:
  - Argument jobid/.. must be specified at the end of the command (both the jobId and JDL file name must be put at the end of the command line)
  - Argument is missing for the "–output" option (the user forgot to add the parameter, required by the argument)
  - Argument "-all" cannot be specified with argument "–input" (some arguments are OR-exclusive)
  - CEId format is: <fullhostname>:<port number>/jobmanager-<service>. The provided CEID: "http://lx01.absolute.com:10854/jobmanager" has a wrong format. (the user has mis-spelled the CE identifier after –resource)
- During the calling of the RB API, the following can happen:
  - Resource Broker "grid013g.cnaf.infn.it:7771" not available (can't open a connection with the RB specified in the UI configuration file)
  - Unable to get LB address from RB "grid013g.cnaf.infn.it" (the function get_lb_contact returned an error)

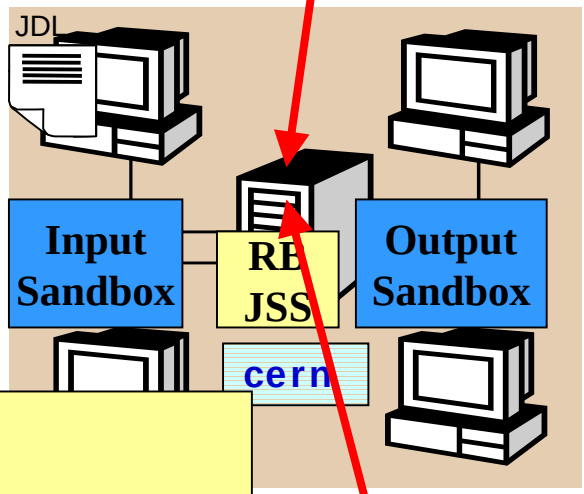
Common Error Message 2/2

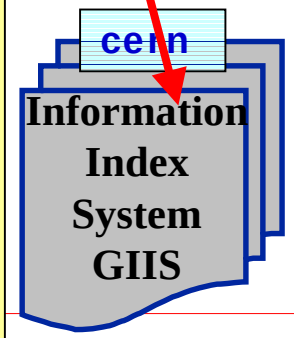
- While the UI commands are checking the JDL file, the following errors may occur:
  - Mandatory Attribute default error in the configuration file "/opt/edg/etc/UI_ConfigENV.cfg" (there aren't any default values)
  - Mandatory Attribute missing in JDL file "Executable" (Executable is one of the mandatory attributes)
  - Multiple "InputSandbox" attribute found in JDL file (InputSandbox attribute is repeated twice)
  - Wrong function call for list attribute *.Function usage is: "Member/IsMember(List,Value)" (e.g. in the requirements attribute the function Member/IsMember is used with a wrong syntax)

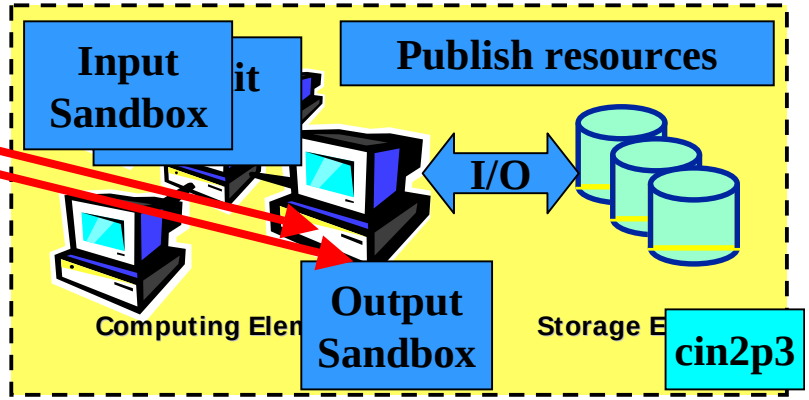
Exemple de fonctionnement

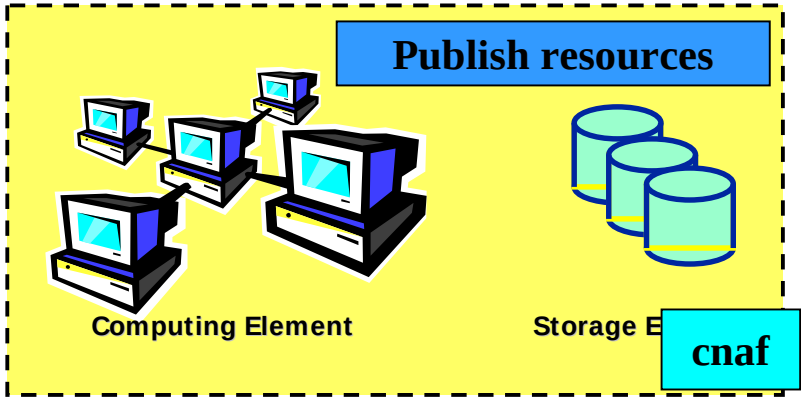


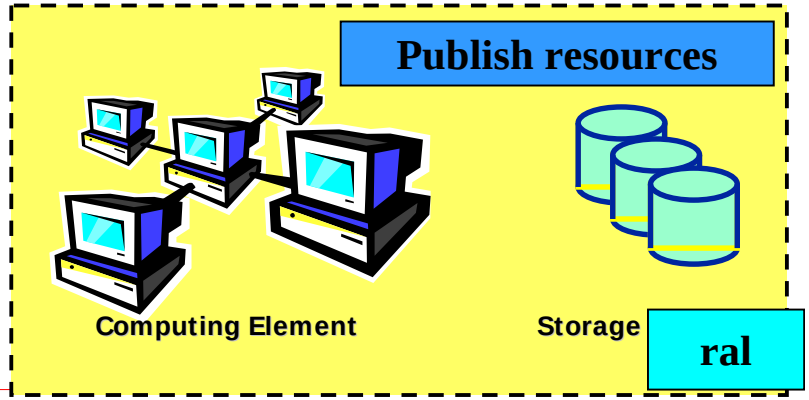













```
# File job.jdl
Executable    = "job.csh";
Arguments     = "123";
StdOutput     = "job.out";
StdError      = "job.err";
InputSandbox  = {"job.csh"};
OutputSandbox  = {"job.out",job.log,job.hist"};
InputData     = "LF:higgs-130:;
ReplicaCatalog = "ldap://atrc.cnaf.infn.it";
```