



Enabling Grids for E-scienceE

gLite/EGEE in Practice

Alex Villazon (DPS, Innsbruck)

Markus Baumgartner (GUP, Linz)

ISPDC 2007

5-8 July 2007

Hagenberg, Austria

With material from
www.eu-egee.org



- Introduction to the gLite grid middleware (part 1)
- **Live tutorial using gLite on the GILDA testbed (part 2)**
 - Grid security
 - Job submission
 - Data Management
 - Grid Information System

- GILDA: **Grid Infn Laboratory for Dissemination Activities**
- GILDA is a testbed for **learning** purposes only
- It is **NOT the EGEE production** infrastructure
- No guarantee for successful job execution



- **GILDA testbed**
 - Series of services
 - gLite as middleware
 - Grid demonstrator (GENIUS web portal)
 - GILDA certification authority (CA)
 - GILDA Virtual Organization (VO)
 - Grid Tutor during tutorials (full version of GENIUS web portal)
 - Monitoring system (based on GridICE)
 - Mailinglist

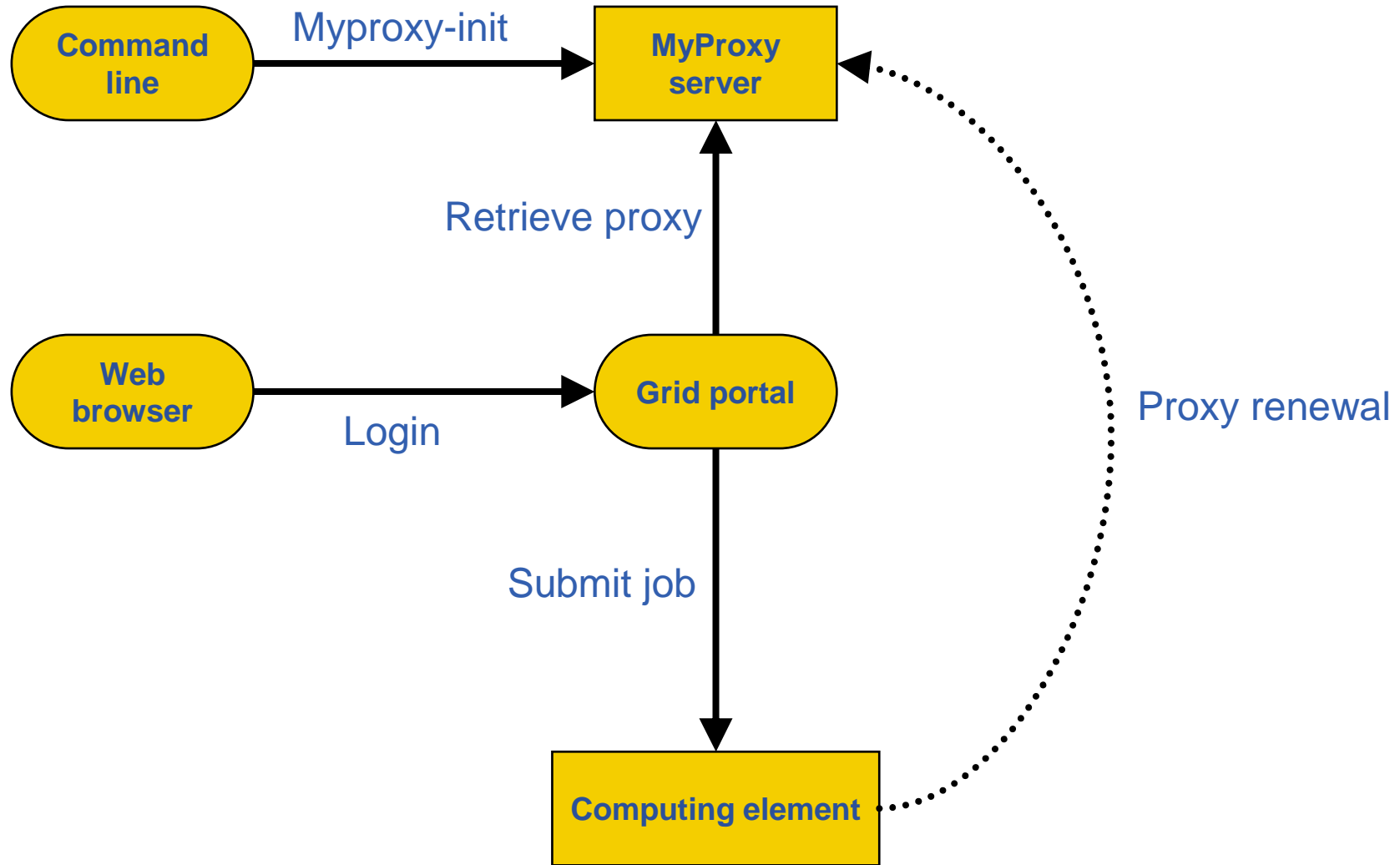
Grid services

This is a table of the general Grid Services available on GILDA.

SERVICE	HOST
Resource Broker (RB)	glite-rb.ct.infn.it
Backup Resource Broker (RB)	glite-rb2.ct.infn.it
WM proxy Resource Broker (RB)	glite-rb3.ct.infn.it
Information Index (BDII)	grid004.ct.infn.it
Backup Information Index (BDII)	glite-rb3.ct.infn.it
VOMS server	voms.ct.infn.it:8443/voms/gilda
DGAS Price Authority	grid-demo1.ct.infn.it:56568
DGAS HLR	grid-demo1.ct.infn.it:56567
GridICE Monitoring System	alifarm7.ct.infn.it:50080
LCG File Catalog (LFC)	lfc-gilda.ct.infn.it
gLite File Transfer Service	fts.ct.infn.it
AMGA Metadata Catalog	amga.ct.infn.it:8822
R-GMA server	https://rgmasv.ct.infn.it:8443/R-GMA/
MyProxy Server	grid001.ct.infn.it
Backup MyProxy Server	grid014.ct.infn.it

- Each grid user needs valid credentials to use the grid
 - gLite uses X.509 certificates as credentials
 - Certificates are issued by a Certification Authority (CA)
 - X.509 credentials consist of
 - The certificate (public key) stored as `~/.globus/usercert.pem`
 - The private key stored as `~/.globus/userkey.pem`
 - The private key is protected by a passphrase
- To improve security, proxy certificates are used
 - The private key is NOT protected by a passphrase
 - A proxy certificate has a limited lifetime (default: 12 hours)
 - Proxy certificates are stored together with their private key

- MyProxy credential repository
 - MyProxy servers store proxy certificates
 - The user first uploads the certificate to the server
 - A proxy can then be retrieved by the user when needed
- Used for
 - Automatic proxy renewal for long-running jobs
 - Grid portals (e.g. GILDA)



- In the following exercise you will:
 - Create a proxy certificate
 - Upload the proxy to the MyProxy server

Exercise 1: Grid security (0)

- Login to your local workstation
 - **User name:** **ispdc**
 - **Password:** **..ispdc**
- These slides can be found in your home directory – please open them so you can do the exercises at your own speed

Exercise 1: Grid security (1)

- Initialize your grid proxy certificate
 - **Login to glite-tutor.ct.infn.it using ssh (putty)**
 - > ssh hagenbergXX@glite-tutor.ct.infn.it
 - Substitute XX by your account number (01 to 25)
 - Use the password "GridHAGXX"
 - **Create a proxy certificate**
 - > voms-proxy-init –voms gilda
 - Provide the passphrase "**HAGENGERG**" to decrypt the user key
 - **Verify the proxy certificate**
 - > voms-proxy-info -all
 - **After the practicals, delete your temporary proxy**
 - > voms-proxy-destroy

If successful the output will be "Your proxy is valid until..."

This will show you subject, issuer, etc. of your local proxy certificate

Exercise 1: Grid security (2)

- Upload your proxy certificate to the MyProxy server:

- **Initialize and upload your MyProxy certificate**

> myproxy-init -s grid001.ct.infn.it -o gilda -c 120

If successful the output will be "A proxy valid for .. hours..."

- Provide the passphrase "**HAGENERG**" to decrypt the user key
- Choose a passphrase for the proxy certificate on the server

- **Verify the MyProxy certificate on the server**

> myproxy-info -s grid001.ct.infn.it

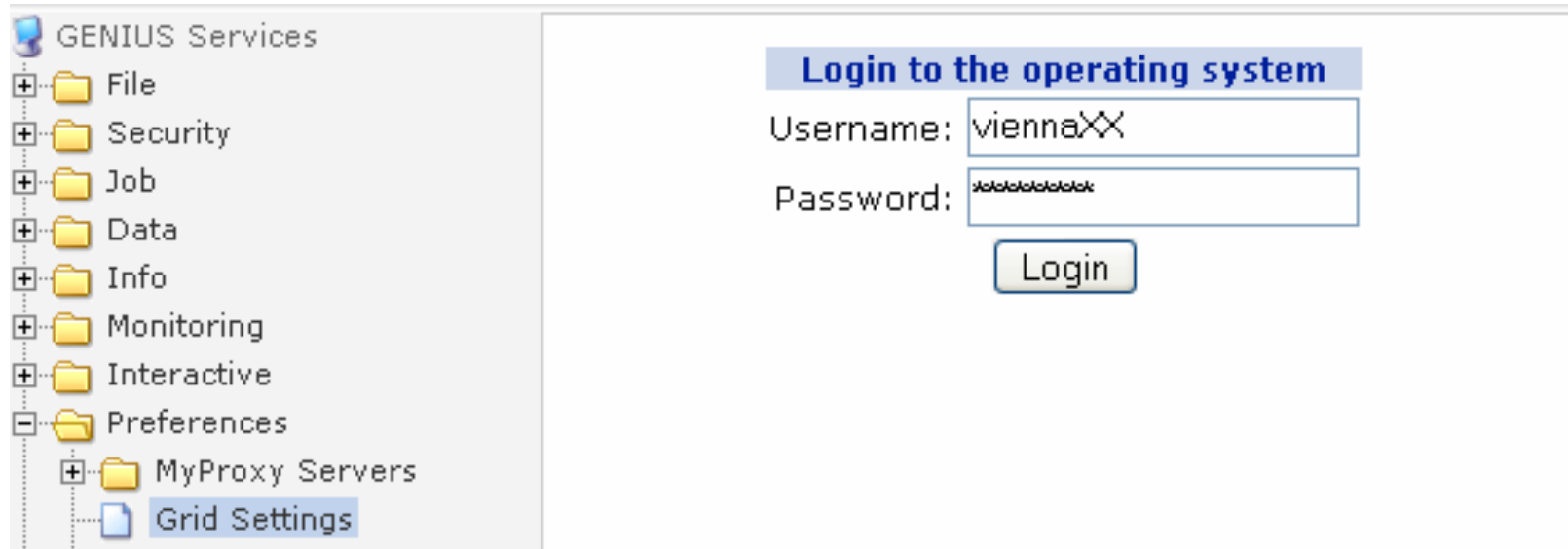
Shows you details about your proxy on the MyProxy server

- After the practicals, remove your proxy certificate from the server

> myproxy-destroy -s grid001.ct.infn.it

- **Please do exercise 1**
- **Time: 5 minutes**
- **After this exercise you should have**
 - a valid grid proxy with VO "gilda"
 - On glite-tutor.ct.infn.it (ssh)
 - a valid MyProxy certificate
 - On grid001.ct.infn.it

- <https://glite-tutor.ct.infn.it>
- Choose 'Preferences - Grid Settings' and log in to the portal



- **Username:** hagenberg01 ... hagenberg30
- **Password:** GridHAG01 ... GridHAG30
- **Slides:** http://www.gup.uni-linz.ac.at/~mbaumgar/grid/glite_hagenberg.ppt
http://www.gup.uni-linz.ac.at/~mbaumgar/grid/glite_hagenberg.pdf

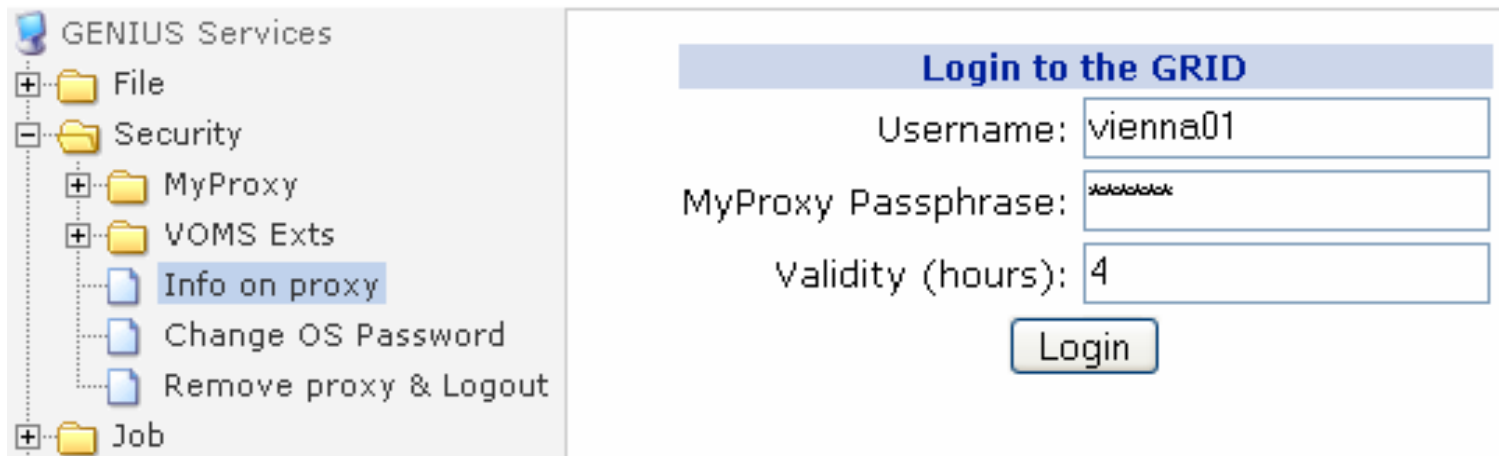
Grid Settings

With this service you can select the Resource Broker and the Replica Location Server you want to use.

It is possible also to set the MyProxy Server on which you stored your temporary proxy to get from GRID Authentication.

Resource Broker (RB):	<input type="text" value="gilda - glite-rb.ct.infn.it (currently selected)"/>
Catalog Server Host:	<input type="text" value="gilda - lfc-gilda.ct.infn.it (currently selected)"/>
MyProxy Server:	<input type="text" value="grid001.ct.infn.it (currently selected)"/>
AMGA Server:	<input type="text" value="gilda - amga.ct.infn.it (currently selected)"/>
DGAS Price Authority Server:	<input type="text" value="gilda - grid-demo1.ct.infn.it (currently selected)"/>
DGAS Home Location Register Server:	<input type="text" value="gilda - grid-demo1.ct.infn.it (currently selected)"/>

- **Resource Broker** knows all resources available and assigns jobs to them
- **Catalog** defines the Replica Catalog used by default
- **MyProxy Server** defines the server storing the myproxy certificate
- **DGAS Servers** provide accounting services for the grid



- **Login to the GRID**
 - Specify username and your MyProxy passphrase
- **Get information on the currently used proxy certificates**
 - Info on the local proxy certificate
 - Info on the local VOMS proxy certificate
 - Info on the certificate stored on the MyProxy server

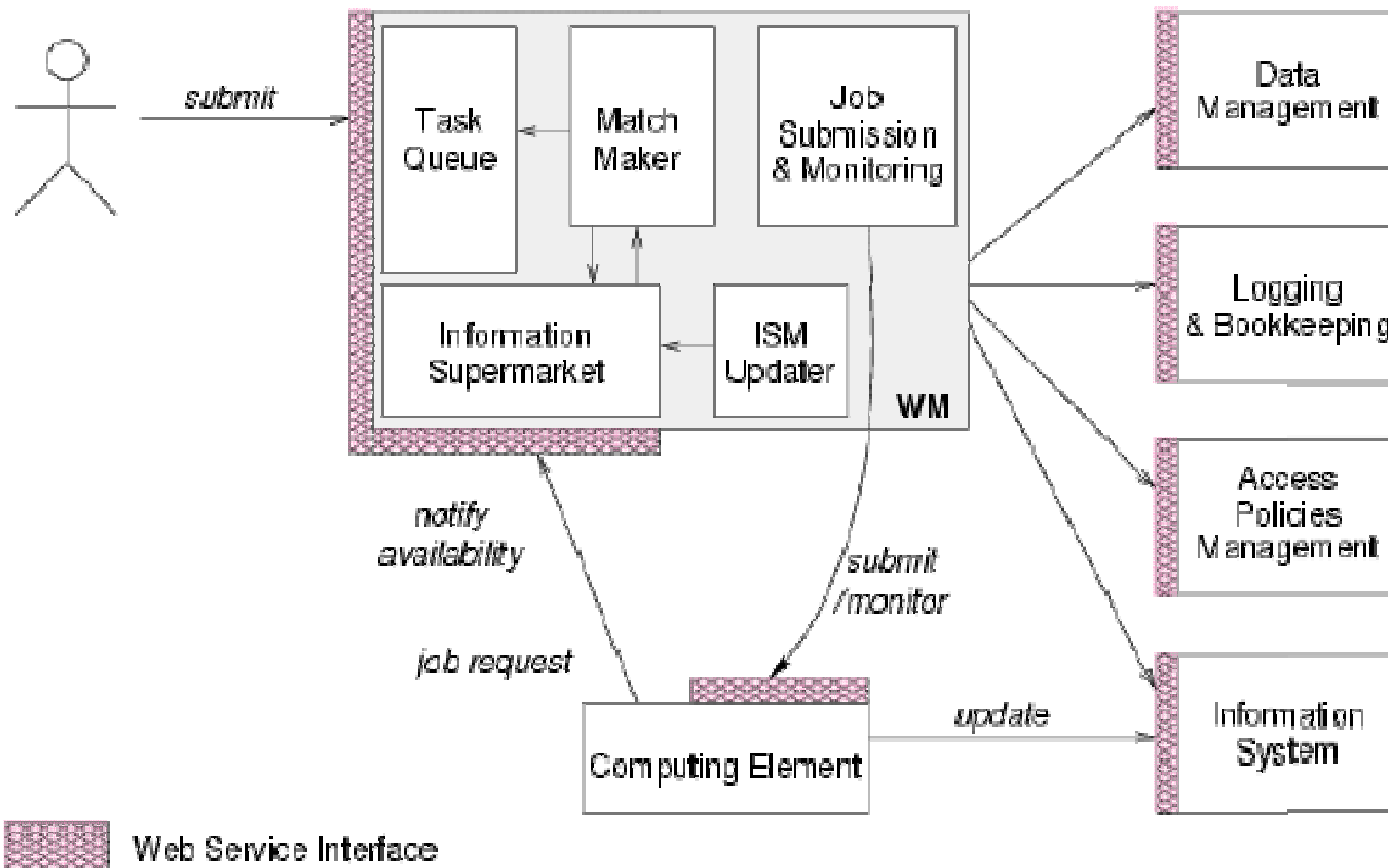
Exercise 2: Security Services

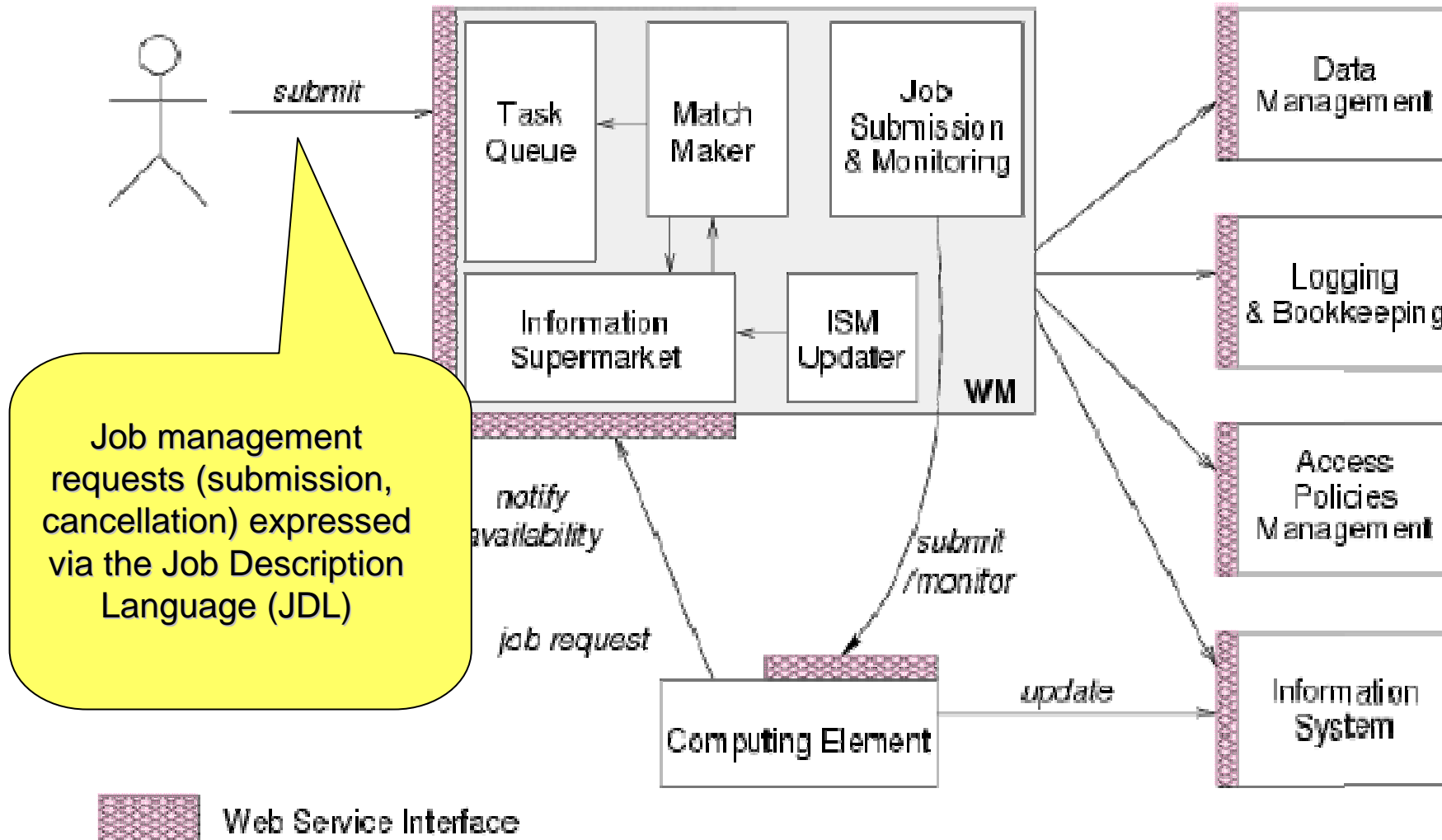
UI <code>glite-tutor.ct.infn.it</code>	GENIUS WebUI
<ul style="list-style-type: none"> > <code>voms-proxy-info</code> > <code>myproxy-info -s grid001.ct.infn.it</code> 	<ul style="list-style-type: none"> Click on "Info on VOMS proxy" Click on "Info on MyProxy"
<ul style="list-style-type: none"> > <code>myproxy-destroy -s grid001.ct.infn.it</code> > <code>myproxy-info -s grid001.ct.infn.it</code> 	<ul style="list-style-type: none"> Click on "Info on MyProxy" (should give an error)
<ul style="list-style-type: none"> > <code>voms-proxy-destroy</code> > <code>voms-proxy-info</code> 	<ul style="list-style-type: none"> Click on "Info on proxy" (should give an error)
<ul style="list-style-type: none"> > <code>myproxy-init -s grid001.ct.infn.it -o gilda</code> > <code>myproxy-info -s grid001.ct.infn.it</code> 	<ul style="list-style-type: none"> Click on "Info on MyProxy"
<ul style="list-style-type: none"> > <code>myproxy-get-delegation -s grid001.ct.infn.it</code> 	
<ul style="list-style-type: none"> > <code>voms-proxy-info</code> 	<ul style="list-style-type: none"> Click on "Info on proxy"

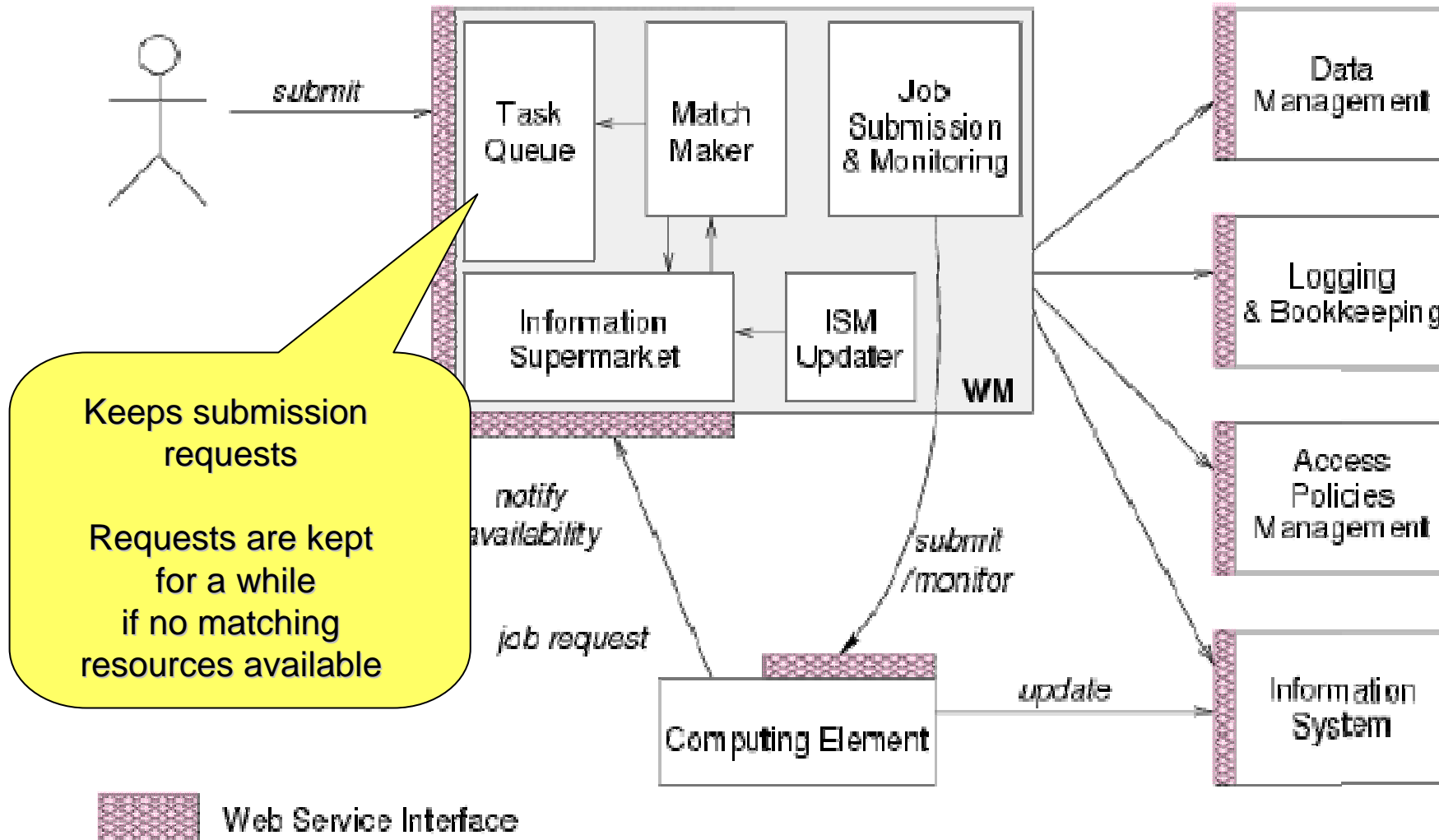
- Please do exercise 2
- Time: 5 minutes
- Please make sure that your grid settings are as shown below after this exercise.

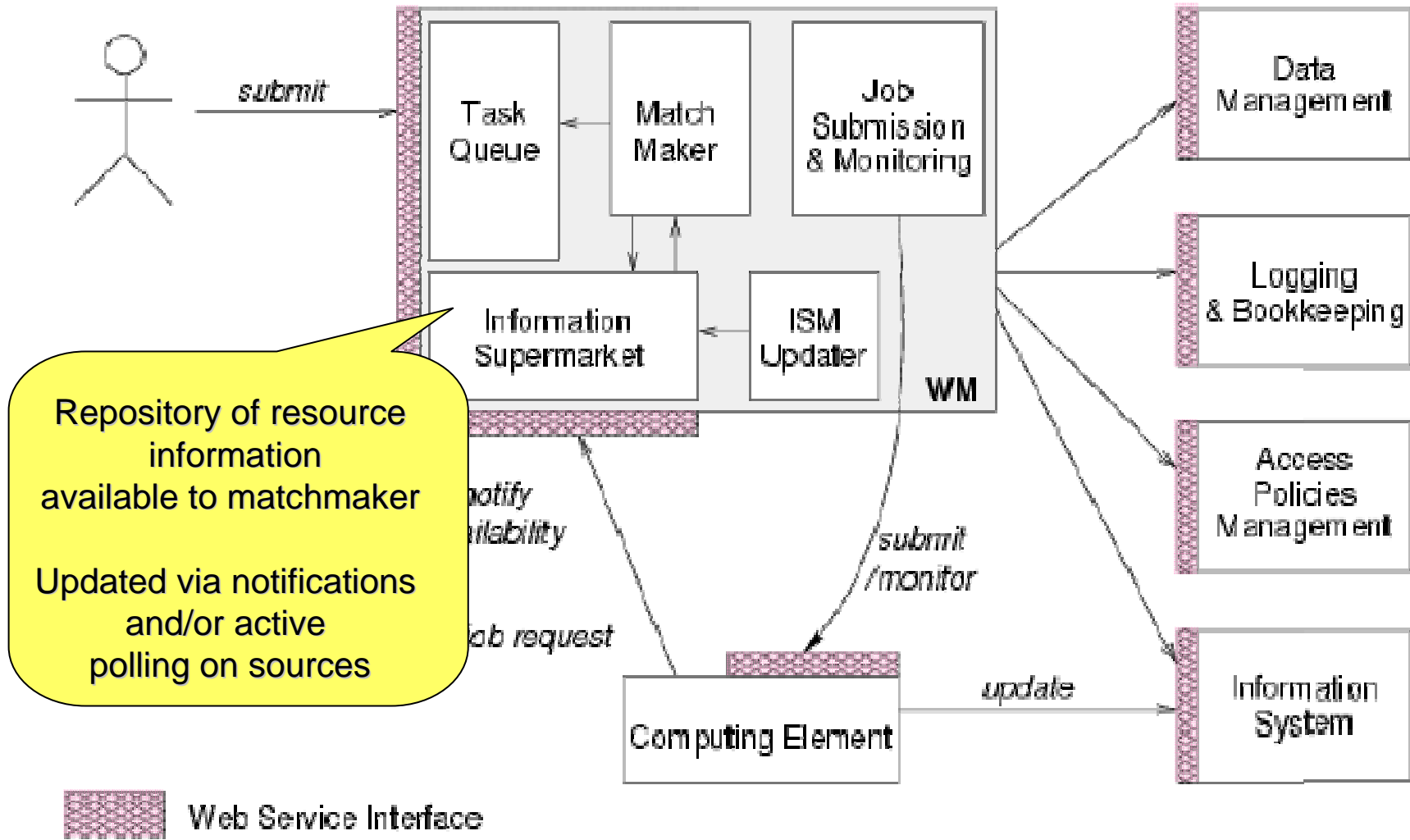
A screenshot of the GENIUS web interface. The top navigation bar includes logos for INFN (Istituto Nazionale di Fisica Nucleare), eGEE (Enabling Grids for E-science), and the 'genius' logo. Below these logos, the text reads 'Grid Enabled web eNvironment for genius site Independent User job Submission'. A blue bar at the bottom of the navigation area contains the text 'Resource Broker: gilda Virtual Organization: gilda LFC Host: lfc-gilda.ct.infn.it', which is highlighted with a red rectangular box. To the right of this bar are links for 'Your Data' and 'Logout'. On the left side of the interface, there is a 'Welcome dombim01' message and a 'GENIUS Services' link.

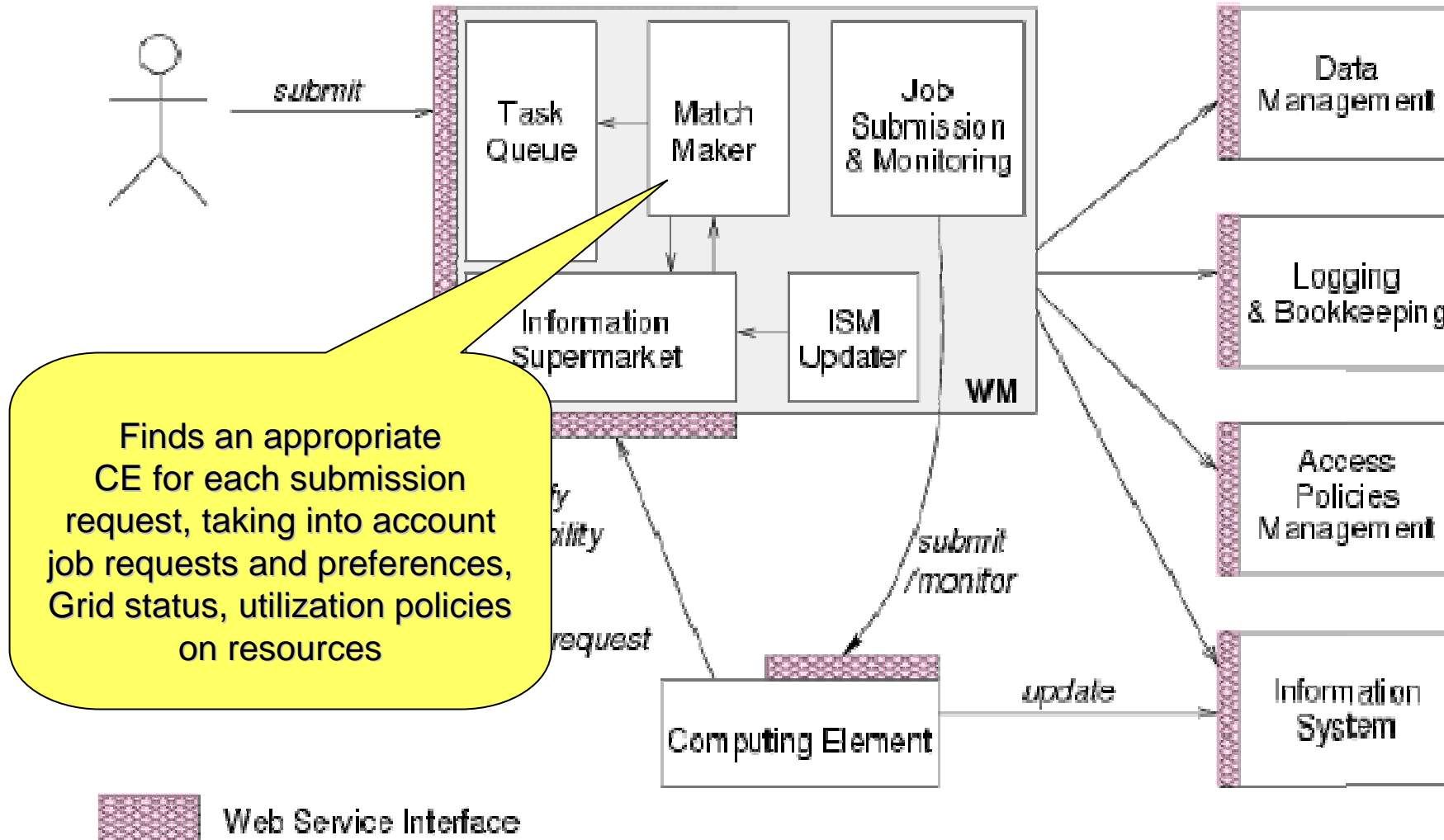
- **Workload Management System (WMS) comprises a set of Grid middleware components responsible for distribution and management of tasks across Grid resources**
 - applications are conveniently, efficiently and effectively executed.
- **Comparable services from other grid projects are, among others, the EDG WMS, Condor and the Eurogrid-Unicore resource broker.**

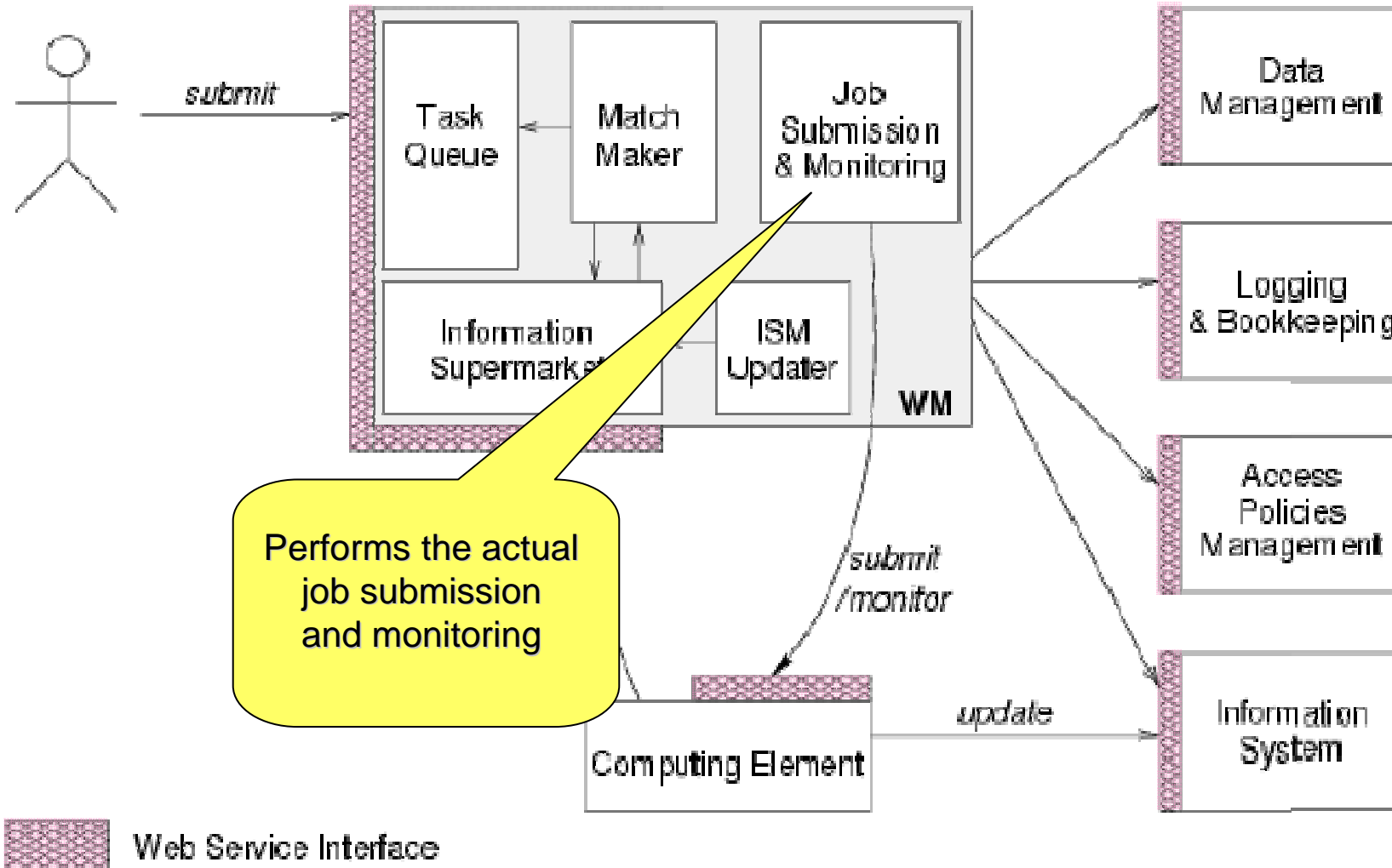






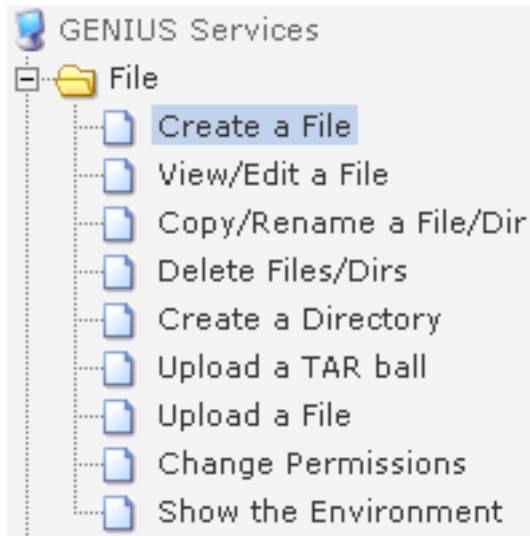






- In the following exercise you will:
 - Create a program to be run in the grid (shell script)
 - Create a job description using the JDL
 - Submit the job to the grid
 - Inspect the output of the job

Exercise 3: Job submission (1)



- Create a file named 'myhelloworld.sh'. This will be the job to be run in the grid.

```
#!/bin/sh
```

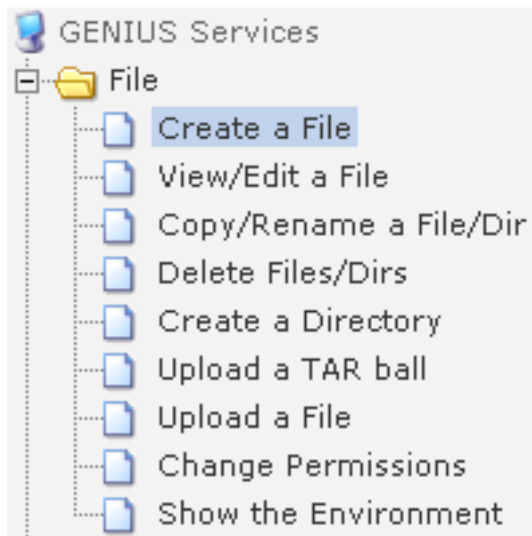
```
MYNAME="Your Name"
```

```
WORKER_NODE=$(hostname)
```

```
echo "Hello ${MYNAME}"
```

```
echo "Greetings from ${WORKER_NODE}!"
```

Exercise 3: Job submission (2)



- Create a JDL file named 'myhelloworld.jdl'

[

Executable="myhelloworld.sh";

StdOutput="std.out";

StdError="std.err";

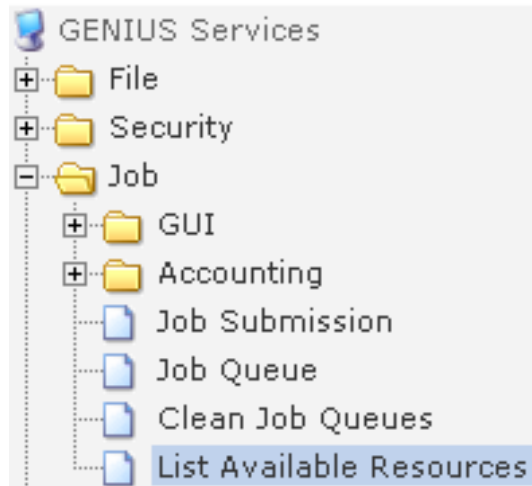
VirtualOrganisation="gilda";

InputSandbox={"myhelloworld.sh"};

OutputSandbox={"std.out", "std.err"};

RetryCount=10;

]



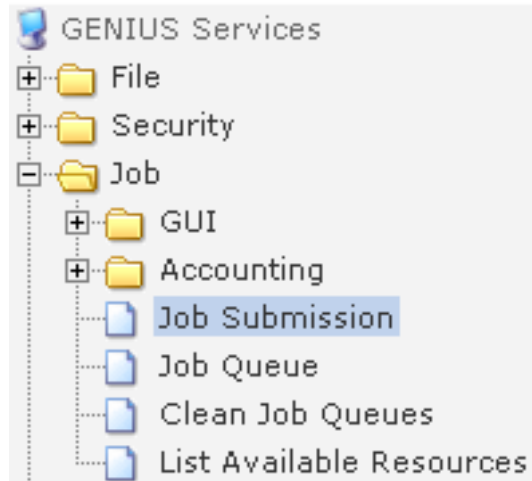
Finding available resources

- From the Main menu choose "Job Services"
 - Choose "List Available Resources"
 - Specify the previously created JDL file and query for matching resources

- Using the command line interface:

- Login to your account on `glite-tutor.ct.infn.it` using `ssh`.
- In the home directory you can find the previously created JDL file
- Issue the following command:

```
> glite-job-list-match <JDL-file>
```



Submitting the job

- From the "Job" menu choose "Job Submission"
 - Specify the JDL file
 - Proceed by pressing the "Next" button
 - Submit the job by pressing the "Submit Job" button

- Submit the job using the command line interface by

> glite-job-submit <JDL-file>

The glite-job-submit command returns the **job identifier** which is going to be used in subsequent slides.

Example: https://glite-rb2.ct.infn.it:9000/ZpmKtN2NtPJAny4G10_YmA

Job Queue						
#	Globus JobID	Last Update Time	Status	Destination	Exit Code	Name
1	RZr456M2AH9nMDemLloS7A	2006 Mar 14 23:46	Ready <input type="button" value="v"/>	egee008.cnaf.infn.it:2119/blah-pbs-short		hostname.jdl
▶Details						

Checking the job status

- Query the job status by pressing "Job Queue"
 - The "Status" changes from "Ready" to "Scheduled" to "Running", and eventually to "Done",
 - It is possible to cancel the job while it is not done.
- These functions are equal to the
 - > glite-job-status <JobID>
 - > glite-job-cancel <JobID>
 commands on the command line.

Job Queue						
#	Globus JobID	Last Update Time	Status	Destination	Exit Code	Name
1	RZr456M2AH9nMDemLloS7A	2006 Mar 14 23:46	Done	egee008.cnaf.infn.it:2119/blah-pbs-short	0	hostname.jdl
			<input type="button" value="Details"/>			
			<input type="button" value="GET OUTPUT"/>			

Retrieving the job output

- Fetch the output by pressing "Get Output"
- You can inspect the files by clicking on them
- On the command line you can do this with
 - > `glite-job-output <JobID>`
- The output is stored in the `~/JobOutput/` directory. The exact location is given to you as the command returns.
- Go there and inspect the files.

- **Please do exercise 3**
- **Time: 15 minutes**

Submitting your own program

- In the following exercise you will:
 - Write and compile a C++ program to be run in the grid
 - Create a JDL file specifying the requirements for that job
 - Submit the job to be run in the grid
 - Inspect the output of the job

- Create a file "myprogram.cpp" with the following content:

```
#include <iostream>
int main() {
    std::cout << "Hello World!" << std::endl;
}
```

- Compile and run

```
> g++ -o myprogram myprogram.cpp && ./myprogram
```

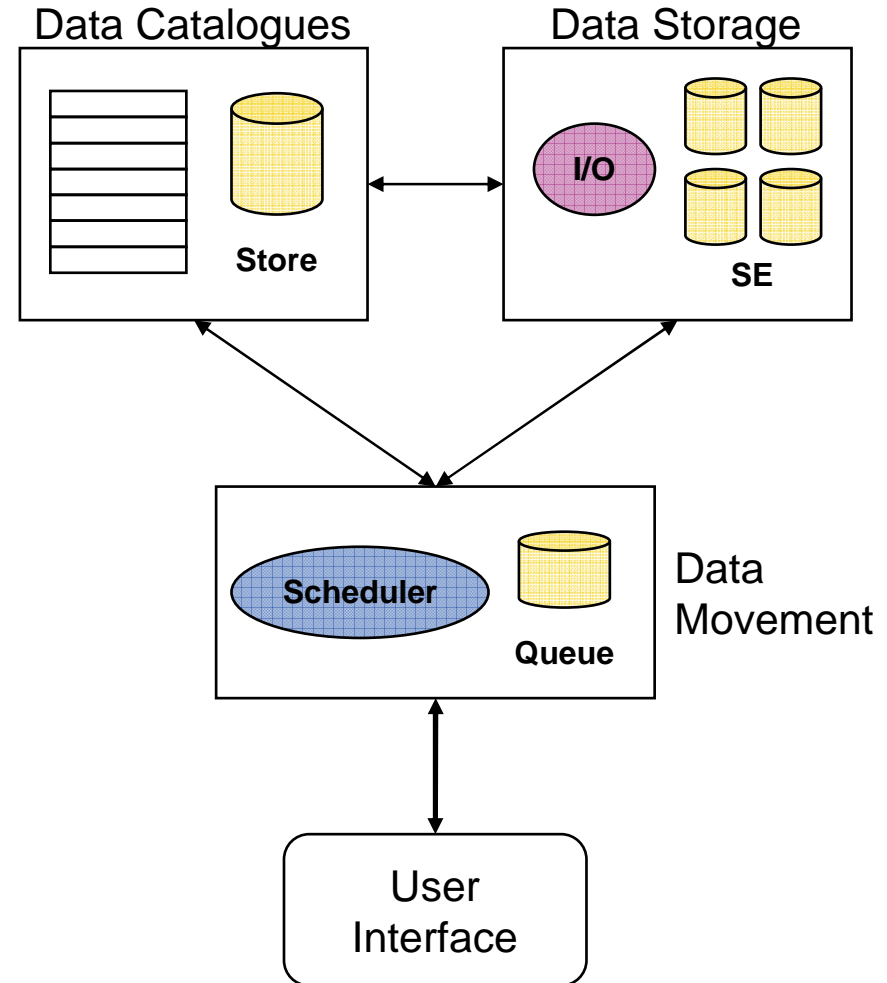
- Create the JDL file ,myprogram.jdl' with the following content:

```
[  
    Executable="myprogram";  
    StdOutput="std.out";  
    StdError="std.err";  
    InputSandbox={"myprogram"};  
    OutputSandbox={"std.out","std.err"};  
    RetryCount=10;  
]
```

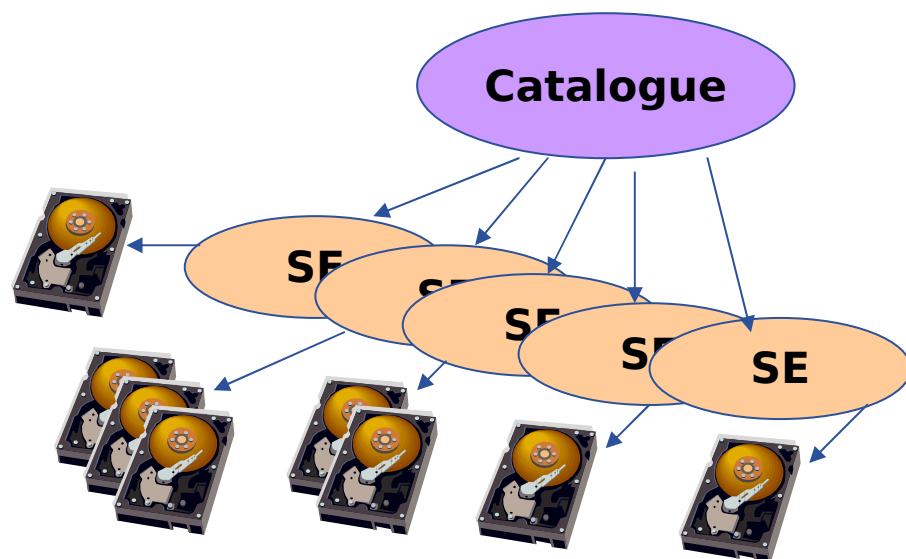
- Submit it to the Grid
- Check the status
- Fetch and inspect the output

- **Please do exercise 4**
- **Time: 10 minutes**

- **Storage Element**
 - Storage Resource Manager
 - POSIX-I/O
 - Access protocols
- **Catalogues**
 - File Catalogue
 - Replica Catalogue
 - File Authorization Service
 - Metadata Catalogue
- **File Transfer**
 - Data Scheduler
(not implemented yet)
 - File Transfer Service
 - File Placement Services
- **User Interface**



- Each file has a unique identifier
- Files/directories are organized on a Catalogue
 - Similar to a filesystem (Logical File Name)
- There is one Catalogue per VO
- The data can be stored on several Storage Elements (SE)
- The Catalogue hides the actual location



Logical File Name

LFN : /grid/gilda/hagenberg/file.txt

Storage Resource Manager

srm://trigrig-ce01.unime.it/dpm/unime.it/home/gilda/generated/
2006-09-20/filef026441a-5834-431f-b28d-06cb7e4c784f

Physical Filename

/home/gilda/generated/2006-09-20/filef026441a-5834-431f-
b28d-
06cb7e4c784f

- **File Catalog**

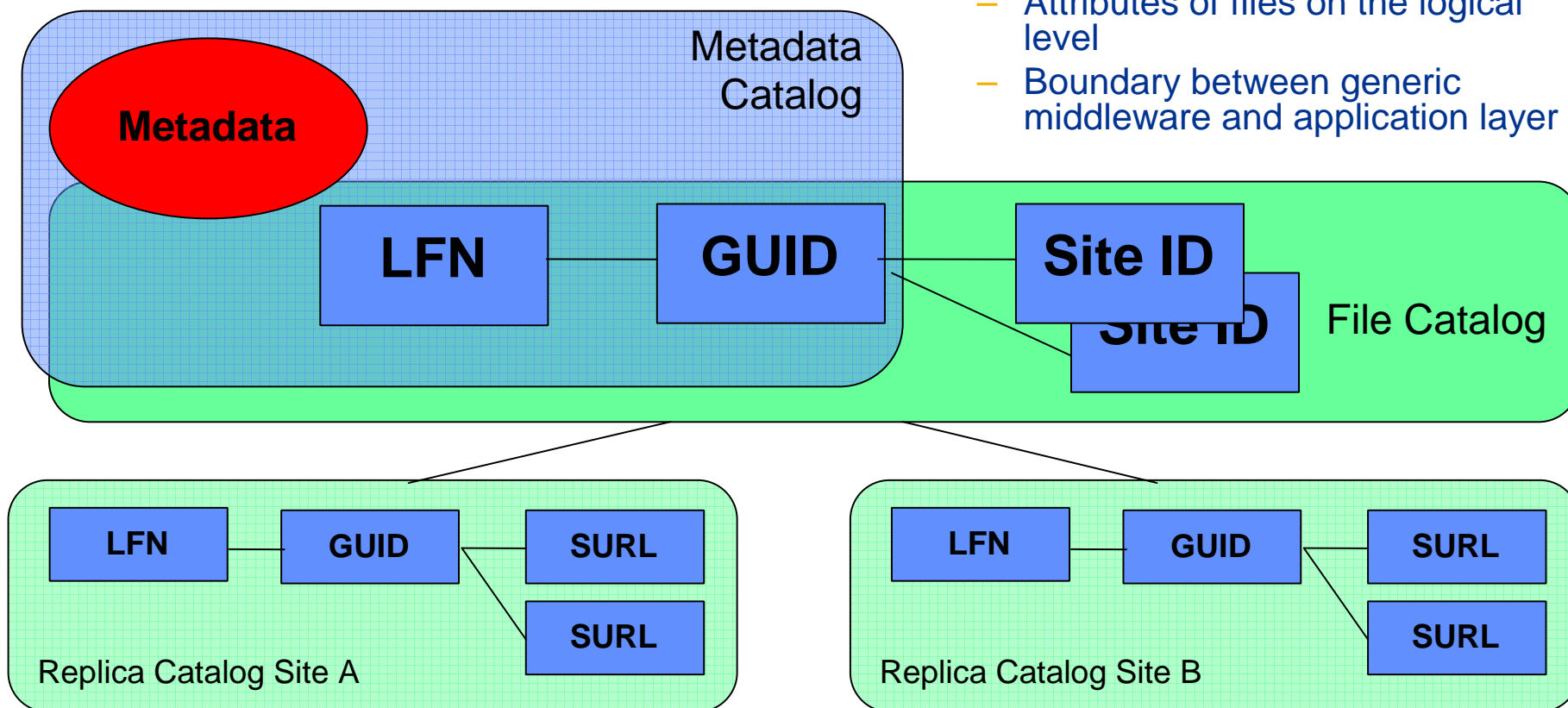
- Filesystem-like view on logical file names
- Keeps track of sites where data is stored
- Conflict resolution

- **Replica Catalog**

- Keeps information at a site

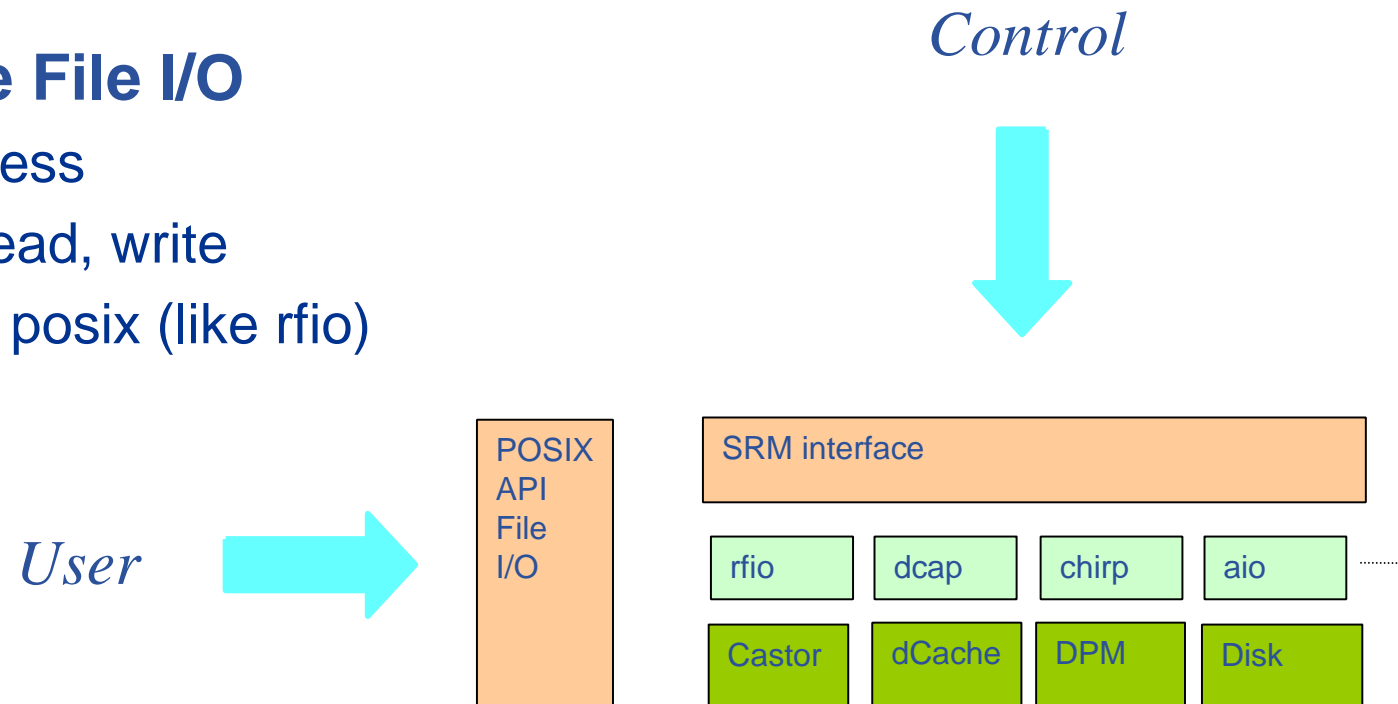
- **(Meta Data Catalog)**

- Attributes of files on the logical level
- Boundary between generic middleware and application layer



- **SRM interface**
 - Management and control
 - SRM (with possible evolution)

- **Posix-like File I/O**
 - File Access
 - Open, read, write
 - Not real posix (like rfio)



- In the following exercise you will:
 - Browse the LFC catalogue
 - Create and remove directories in the catalogue
 - Publish files in the catalogue

- **Browse the file catalog using**
 - > `lfc-ls -l /grid/gilda`
- **Create a new directory in the catalog**
 - > `lfc-mkdir /grid/gilda/hagenbergXX`
- **Explore the properties of the newly created directory**
 - > `lfc-ls -ld /grid/gilda/hagenbergXX`
 - > `lfc-getacl /grid/gilda/hagenbergXX`
- **Some more commands: `lfc-chown`, `lfc-ln`, `lfc-rename`, `lfc-rm`**

- **Store a local file on the storage element and assign it a LFN (Logical File Name) in the catalogue**
 - > `lcg-cr --vo gilda -l /grid/gilda/hagenbergXX/hostname.jdl \`
`file://$HOME/hostname.jdl`
- **Browse the file catalog again**
 - > `lfc-ls -l /grid/gilda/hagenbergXX`
- **Retrieve the file from the storage element**
 - > `lcg-cp --vo gilda lfn:/grid/gilda/hagenbergXX/hostname.jdl \`
`file://$HOME/hostname.jdl.bak`
- **Remove it from the storage element and the catalogue**
 - > `lcg-del -a --vo gilda lfn:/grid/gilda/hagenbergXX/hostname.jdl`
- **Remove the directory from the catalog**
 - > `lfc-rm -r /grid/gilda/hagenbergXX`
- **Browse the file catalog**
 - > `lfc-ls -l /grid/gilda/hagenbergXX`

- **Please do exercise 5**
- **Time: 10 minutes**

- **Please remove your directory from the catalogue after this exercise**

- **The JDL is used to define**
 - Job characteristics
 - Job requirements
 - Data requirements

- **Complete JDL reference**
 - <http://glite.web.cern.ch/glite/documentation/>

- **The supported attributes are grouped into two categories:**
 - Job Attributes
 - Define the job itself
 - Resources
 - Taken into account by the Workload Manager for carrying out the matchmaking algorithm (to choose the “best” resource where to submit the job)
 - **Computing Resource**
 - *Used to build expressions of Requirements and/or Rank attributes by the user*
 - *Have to be prefixed with “other.”*
 - **Data and Storage resources**
 - *Input data to process, Storage Element (SE) where to store output data, protocols spoken by application when accessing SEs*

- **JobType (mandatory)**
 - *Normal* (simple, sequential job), *DAG*, *Interactive*, *MPICH*, *Checkpointable*
- **Executable (mandatory)**
 - The command name
- **Arguments (optional)**
 - Job command line arguments
- **StdInput, StdOutput, StdError (optional)**
 - Standard input/output/error of the job
- **Environment (optional)**
 - List of environment settings

- **InputSandbox (optional)**
 - List of files on the UI's local disk needed by the job for running
 - The listed files will be staged to the remote resource automatically
- **OutputSandbox (optional)**
 - List of files, generated by the job, which have to be retrieved
- **VirtualOrganisation (mandatory)**
 - The virtual organisation the user submitting the job is working for
 - Can be omitted if preconfigured on the UI or in the VOMS proxy

- Requirements (**optional**)
 - Job **requirements on computing resources**
 - Specified using attributes of resources published in the Information Service
- Rank (**optional**)
 - **Expresses preference** (how to rank resources that have already met the Requirements expression)
 - Specified using attributes of resources published in the Information Service
 - If not specified, default value defined in the UI configuration file is considered
 - Default: *other.GlueCEStateEstimatedResponseTime* (the lowest estimated traversal time)
 - Default: *other.GlueCEStateFreeCPUs* (the highest number of free CPUs) for parallel jobs (see later)

- **InputData (optional)**
 - Refers to data used as input by the job: these data are published in the Replica Catalog and stored in the Storage Elements
 - LFNs and/or GUIDs
- **DataAccessProtocol (mandatory if InputData has been specified)**
 - The protocol or the list of protocols which the application is able to speak with for accessing *InputData* on a given Storage Element
- **OutputSE (optional)**
 - The Uniform Resource Identifier of the output Storage Element
 - RB uses it to choose a Computing Element that is compatible with the job and is close to the Storage Element

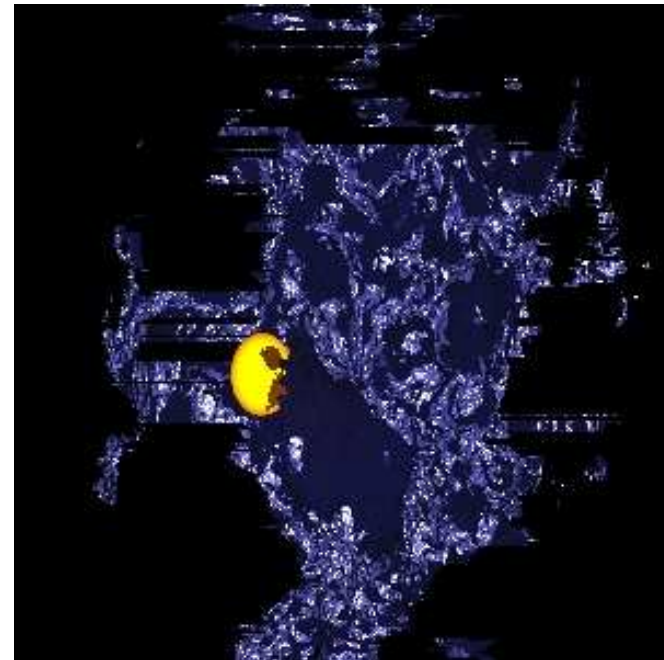
- **Requirements allow you to influence the matchmaking process**
 - Used to specify software requirements (e.g. a specific software has to be installed on the worker node in order to execute the job)
 - A resource must fulfill the constraints specified in order to be allowed to execute the job (e.g. a specified number of processors must be available; a job requires some minimum amount of RAM being available, etc.)
 - Example:

```
Requirements=other.GlueHostOperatingSystem=="LINUX" &&  
other.GlueCEStateFreeCPUs>=4;
```



Allows you to influence the matchmaking process!

- In the following exercise you will:
 - Create a povray raytracing scene to be calculated in the grid
 - Create a JDL file specifying the requirements for that job
 - Submit the job to be run in the grid
 - Inspect the output of the job



- **Objective:** Execute a POVRay rendering job
- **Requirements:** POVRay needs to be installed on the worker node

Create the POVRay scene description file "nuggets.pov":

```
// Thanks to „Dazza“
// http://astronomy.swin.edu.au/~pbourke/povray/scc3/final/
global_settings{radiosity{}}

#declare f=function{pigment{granite}};

union {
  isosurface {
    function{x*x+y*y+z*z-f(x,y,z).red}
    pigment{color rgb<1,1,2>/2}
    finish{specular 1}
  }
  sphere{<-.13,-.05,-.7> .05
  pigment{rgb<4,2,0>}
}
translate z*1.5
}
light_source{0,1}
```

Exercise 6: Job Requirements(2)

- Create a script named "startNuggets.sh" which calls "povray" and renders the scene on the compute element

```
#!/bin/sh
```

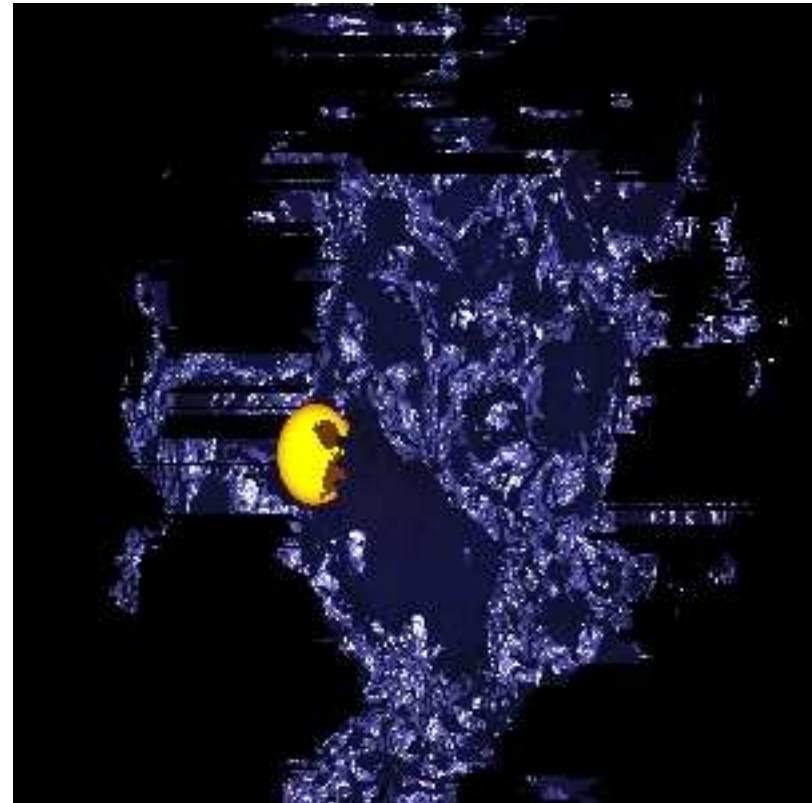
```
povray +Inuggets.pov +Onuggets.tga +FT +W300 +H300 +V -D +X
```

- Create the JDL file nuggets.jdl for rendering the scene:

```
[  
  Executable="startNuggets.sh";  
  StdOutput="nuggets.out";  
  StdError="nuggets.err";  
  InputSandbox={"startNuggets.sh", "nuggets.pov" };  
  OutputSandbox={"nuggets.out", "nuggets.err", "nuggets.tga"};  
  Requirements= Member("POVRAY-3.5",  
    other.GlueHostApplicationSoftwareRunTimeEnvironment);  
]
```

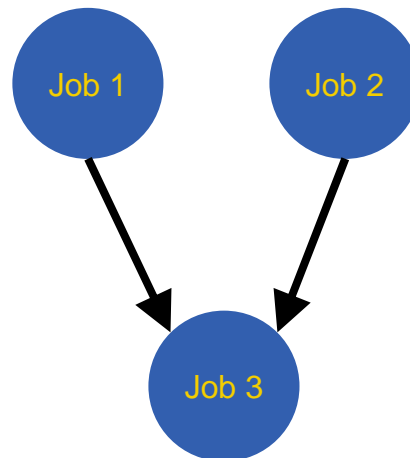
Exercise 6: Job Requirements(4)

- Submit the job
- Fetch the output as soon as it is available
- View the resulting image with your preferred image viewer



- **Please do exercise 6**
- **Time: 15 minutes**

- A DAG job is a **set of jobs** where input, output or execution of one or more jobs can **depend on other jobs**
- Dependencies are represented through Directed Acyclic Graphs where the nodes are jobs and the edges are the dependencies
- Sub-jobs are scheduled only when the corresponding DAG node is ready

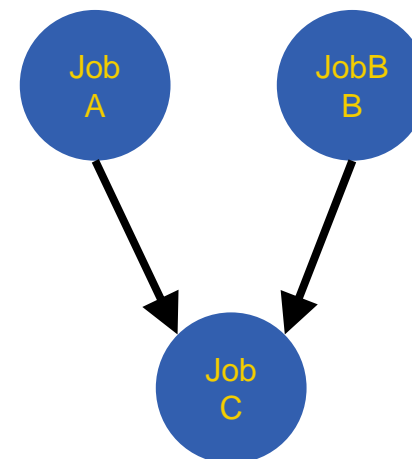


- A DAG job is described using either:
 - a single JDL file containing the entire JDL for each sub-job
 - a master JDL file for the DAG that references one JDL file for each job in the DAG.

The DAG JDL file contains a section for each sub-job.

Also note the dependencies section:

```
dependencies = {
  {{nodeA, nodeB}, nodeC}
}
```

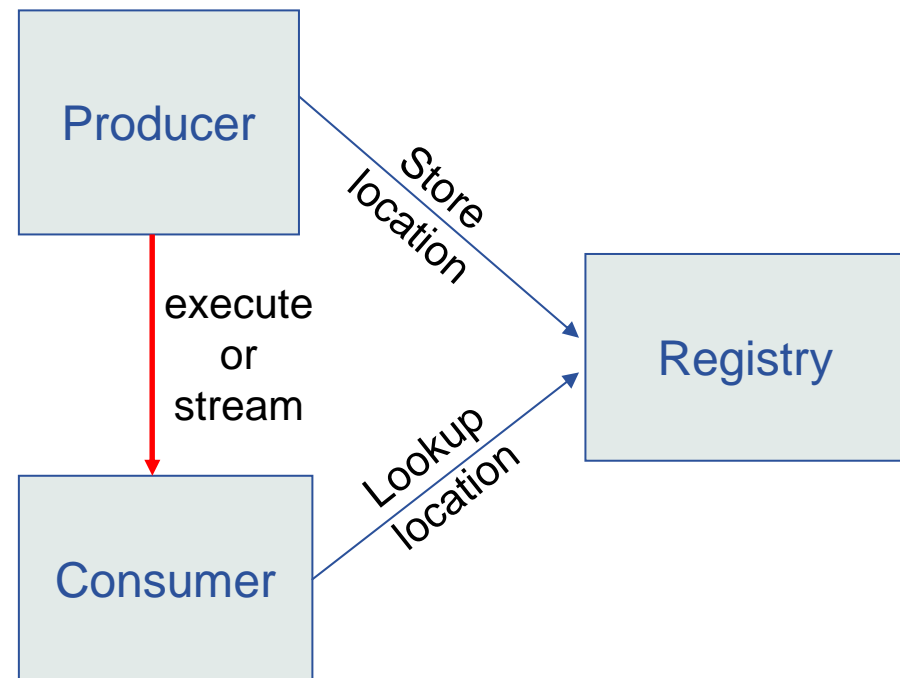


```
# JDL for DAG
[
  nodes = {
    nodeA = [
      file = "nodeA.jdl" ;
    ],
    nodeB = [
      file = "nodeB.jdl" ;
    ],
    nodeC = [
      file = "nodeC.jdl" ;
    ]
  ];

dependencies = {
  {{nodeA, nodeB}, nodeC}
}
]
```

- **R-GMA is based on the Grid Monitoring Architecture (GMA) from the Grid Global Forum (GGF)**
- **Simple Consumer-Producer model that models the information infrastructure of a Grid as**
 - consumers (that request information)
 - producers (that provide information) and,
 - a central registry
- **Mediates the communication between producers and consumers**
- **Provide replication of data**
- **Offers a global view of the information as if each Virtual Organization had one large relational database**
- **Producers contact the registry to announce their intention to publish data**
- **R-GMA adds a standard query language (a subset of SQL) to the GMA model,**

- The **Producer** stores its information in the **Registry**.
- The **Consumer** can query the Registry to find out what type of information is available and locate the best producers that provide such information (*mediation*).
- Once a Consumer has this information it can contact the Producer to get all the data.
- The Registry contains, for each table, a list of producers who have offered to publish rows for it.



- In the following exercise you will:
 - Browse the R-GMA information system

- Metainformation about the state of the Grid is provided through R-GMA (Relational - Grid Monitoring Architecture)
- In order to use the R-GMA commands you will need a working proxy.
- To start working with R-GMA just run the command `rgma` at the prompt
- To get help issue the command `help` at the prompt

[glite-tutor] /home > rgma

```
Welcome to the R-GMA virtual database for Virtual Organisations.
=====
```

```
Your local R-GMA server is:
```

```
https://rgmasrv.ct.infn.it:8443/R-GMA
```

```
You are connected to the following R-GMA Registry services:
```

```
https://rgmasrv.ct.infn.it:8443/R-GMA/RegistryServlet
```

```
You are connected to the following R-GMA Schema service:
```

```
https://rgmasrv.ct.infn.it:8443/R-GMA/SchemaServlet
```

```
Type "help" for a list of commands.
```

```
rgma>
```


- Displays help for a specific command
rgma> help <command>
- Exit the R-GMA command line
rgma> exit or quit
- To show a list of all table names:
rgma> show tables;
- To show information about a table MyTable :
rgma> describe MyTable;
- To show a table of properties for the current session:
rgma> show properties;
- To show a list of all R-GMA producers that produce the table MyTable :
rgma> show producers of MyTable;

- Show all tables that are present in the Schema:
rgma> show tables;
- Choose one table where you want insert your information and see its attribute:
rgma> describe <NameTable>
- Some example queries:
rgma> select UniqueId,Status from GlueCE;
rgma> select UniqueId,Name from GlueService;
rgma> select Name from Site where Latitude > 0;
rgma> select Endpoint, Type from Site,Service where Site.Name = Service.Site_Name and Latitude > 0;

- R-GMA servers provide a web interface
- URL of the GILDA R-GMA server
 - <https://rgmasrv.ct.infn.it:8443/R-GMA>
- Access control is implemented using X.509 certificates
- To access an R-GMA server, the grid certificate must be imported into the web browser (usually, the browser can only handle PKCS12 certificates, so conversion is necessary)
- In GILDA, the PKCS12 certificate is automatically generated and stored in `$HOME/.globus`
- The certificate of the GILDA CA must be imported into the browser as well

R-GMA Browser Home Page - Mozilla

File Edit View Go Bookmarks Tools Window Help

Back Forward Reload Stop <https://rgmasrv.ct.infn.it:8443/R-GMA/> Search Print

Home Bookmarks mozilla.org mozillaZine mozdev.org

R-GMA Browser

Home
Predefined:
[GlueCE](#)
[GlueSE](#)
[GlueSubCluster](#)
[GlueService](#)
[GlueServiceStatus](#)
[GlueSite](#)
[RGMALogs](#)
Table Sets


Query: `SELECT * FROM GlueCE`
Properties: Latest

UniqueID	Name	GlueClusterUniqueID	TotalCPUs	LRMSType	LRMSVersion	GRAMVersion	HostName	GatekeeperPort	RunningJobs
glite01.twcic.net:2119/jobmanager-lcgpbs-long	long	glite01.twcic.net	0	pbs	2.1.6	NULL	glite01.twcic.net	2119	0
glite01.twcic.net:2119/jobmanager-lcgpbs-short	short	glite01.twcic.net	0	pbs	2.1.6	NULL	glite01.twcic.net	2119	0
glite01.twcic.net:2119/jobmanager-lcgpbs-infinite	infinite	glite01.twcic.net	0	pbs	2.1.6	NULL	glite01.twcic.net	2119	0

Query aborted before receiving all tuples; the result may be incomplete. Try using a longer timeout value.

Number of rows: 3

Wait for seconds



Done

- **Please do exercise 8**
- **Time: 10 minutes**

- <http://www.glite.org/>
- <https://gilda.ct.infn.it/>
- <http://www.eu-egee.org/>
- <http://www.egee.nesc.ac.uk/>
- <http://www.cern.ch/>