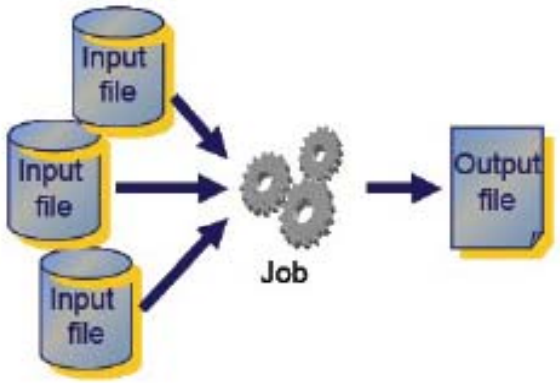
Laboratory: Hands-on using EGEE Grid and gLite middleware

Athanasia Asiki
aassiki@cslab.ece.ntua.gr
Computing Systems Laboratory,
National Technical University of Athens

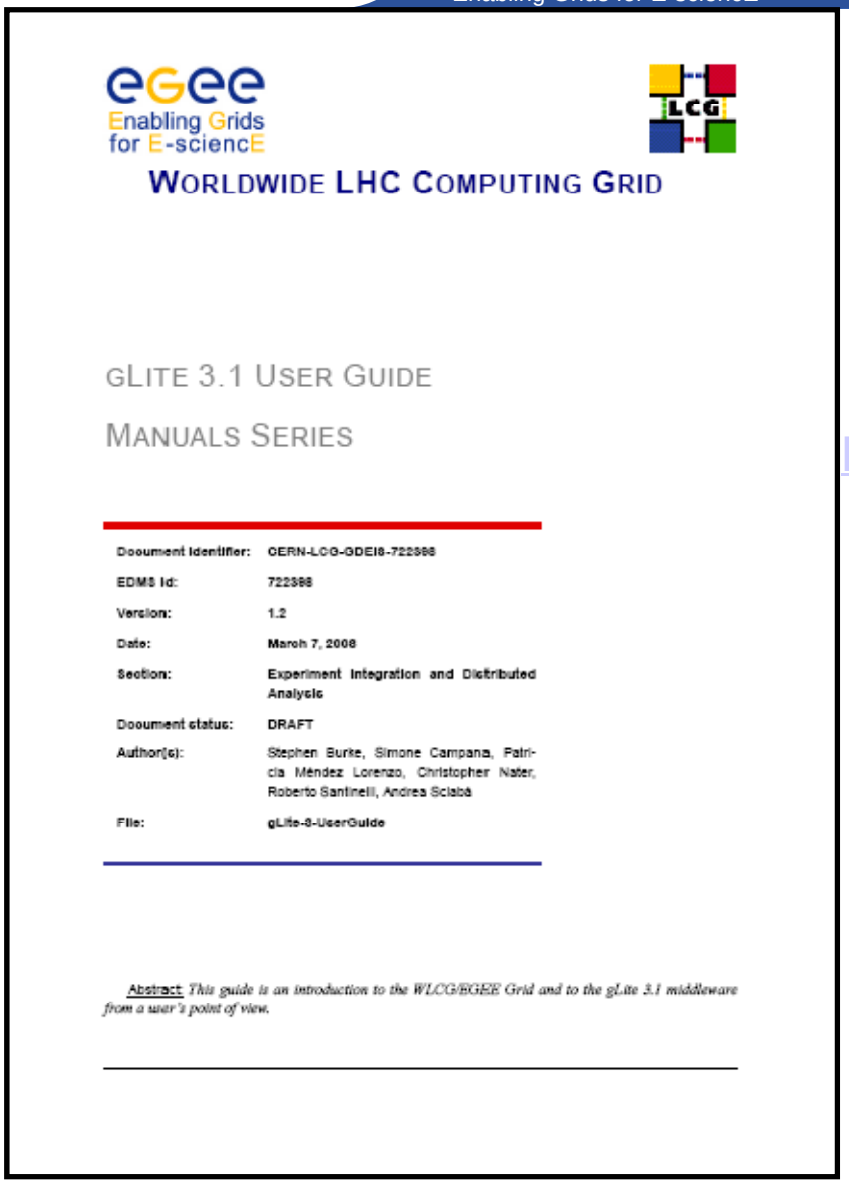
Let's enter the Grid…

Application's structure



- **The execution of a typical Grid application follows this scenario:**
  - The user submits its application's job to the "Grid"
  - The job is being executed
  - The job's execution may include the processing of one or more **Input Files** stored in a Storage node
  - The job may produce one or more **Output Files**
  - The **Output Files** can be stored somewhere in the Grid system (perhaps in the Storage Element or in the User Interface)
  - The User can access the **Output Files** using the corresponding Grid mechanisms

Documentation



Enabling Grids for E-sciencE

WORLDWIDE LHC COMPUTING GRID

gLITE 3.1 USER GUIDE

MANUALS SERIES

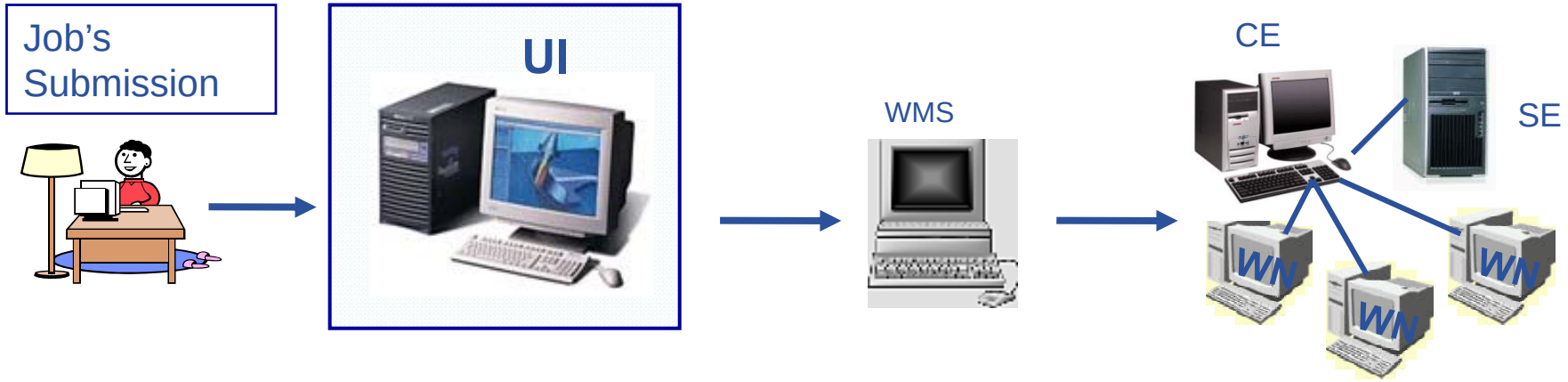
Document Identifier:	CERN-LCG-GDEIS-722398
EDMS Id:	722398
Version:	1.2
Date:	March 7, 2008
Section:	Experiment Integration and Distributed Analysis
Document status:	DRAFT
Author(s):	Stephen Burke, Simone Campana, Patricia Méndez Lorenzo, Christopher Nater, Roberto Santinelli, Andrea Sciabà
File:	gLite-3-UserGuide

Abstract: *This guide is an introduction to the WLCG/EGEE Grid and to the gLite 3.1 middleware from a user's point of view.*

http://glite.web.cern.ch/glite/documentation/

User Interface (1)

- **Allows users to access Grid functionalities**
- **A machine where users have a personal account and where the user certificate is installed**
- **Gateway to Grid Services**



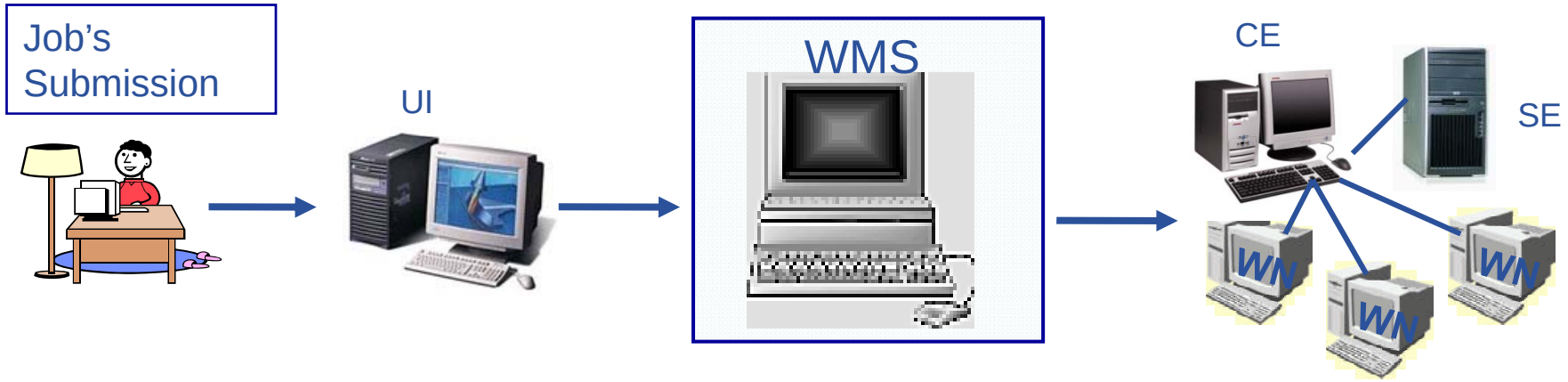
User Interface (2)

- *It provides a Command Line Interface to perform some basic Grid operations such as:*

- List all the resources suitable to execute a given job
- Submit jobs for execution
- Show the status of submitted jobs
- Cancel one or more jobs
- Retrieve the logging and bookkeeping information of jobs
- Retrieve the output of finished jobs
- Copy, replicate and delete files from Grid

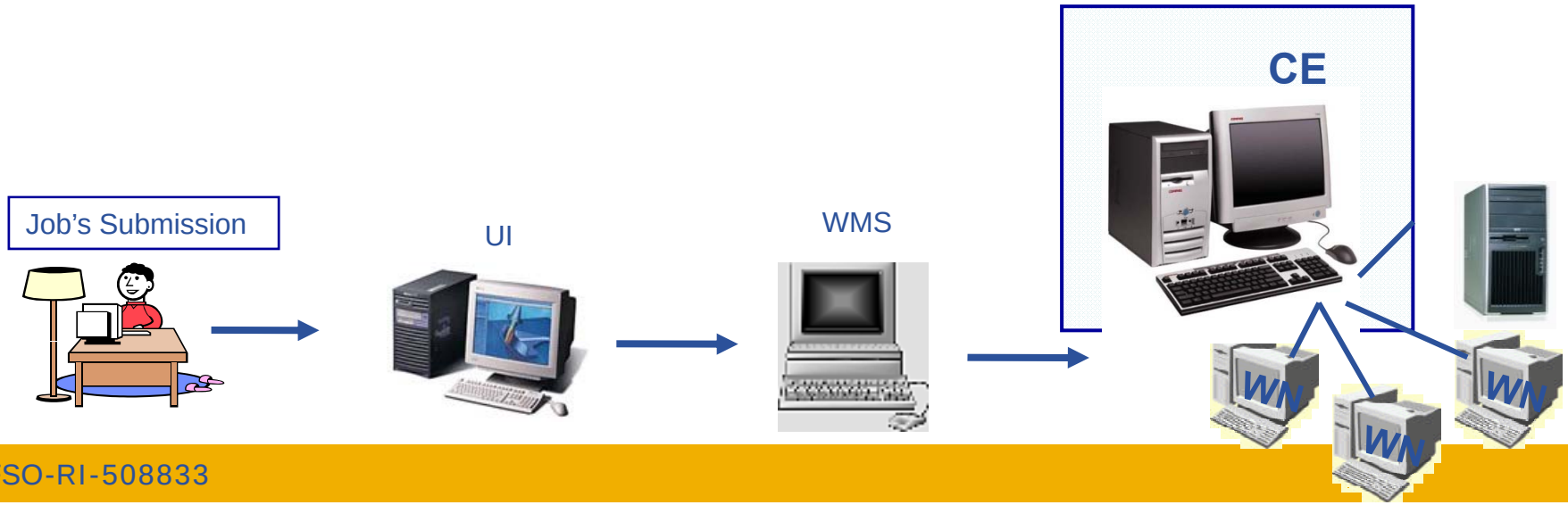
Workload Management System

- **The resource broker is responsible for the acceptance of submitted jobs and for sending those jobs to the appropriate Computing Element**
- **Retrieves information from Information Catalogues so as to find the proper available resources depending on the job requirements**



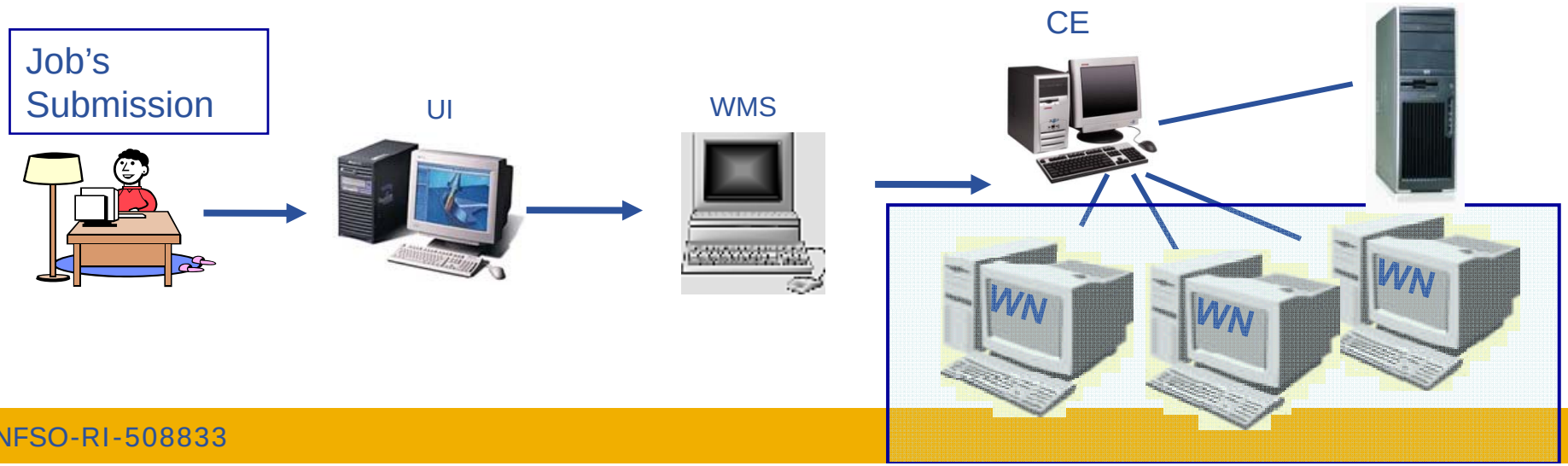
Computing Element

- **"Grid interface"**
- **It is built on a farm of a computing nodes called Worker Nodes (WNs)**
- **Executes the basic queues functions**
- **In the Computing Element, a process is being executed that accepts jobs and dispatch them for execution to the Worker nodes (WNs)**
- **The state of an executing job is being watched by the Computing Element**



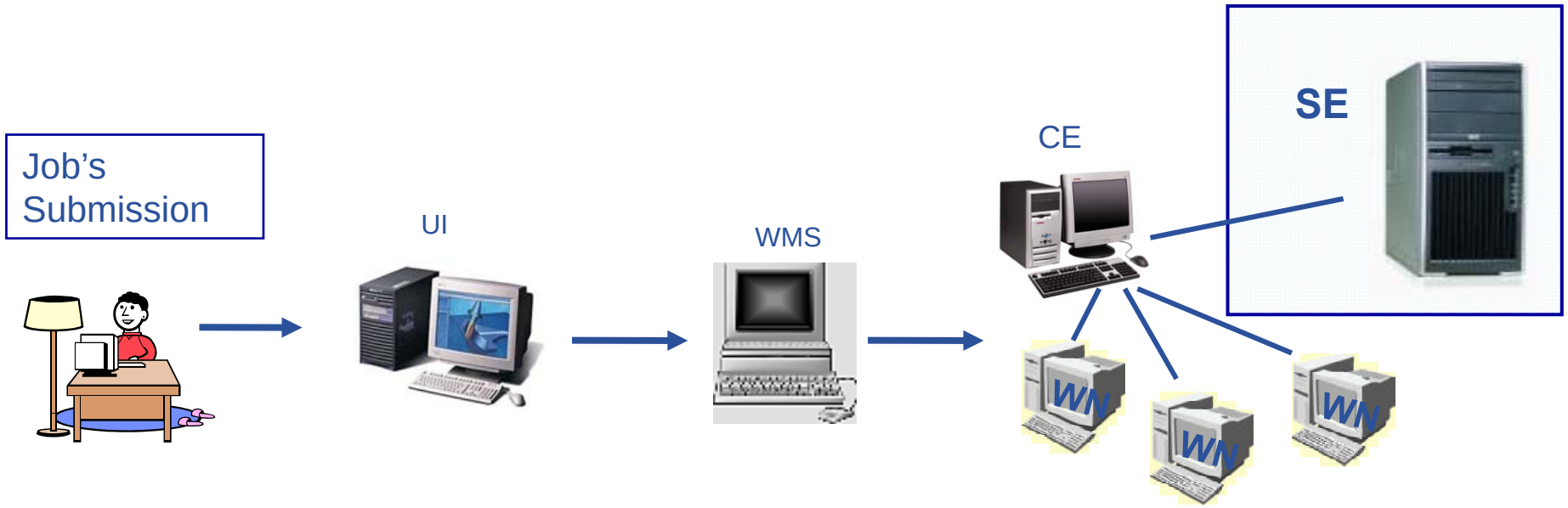
Worker Node

- The submitted jobs are being executed in the Worker nodes
- Need only outbound connectivity
- Only basic services of middleware are required to be provided by the Worker nodes such as
  - Application libraries
  - Application Programming Interfaces (API)
  - Commands for performing actions on Grid resources and Grid data

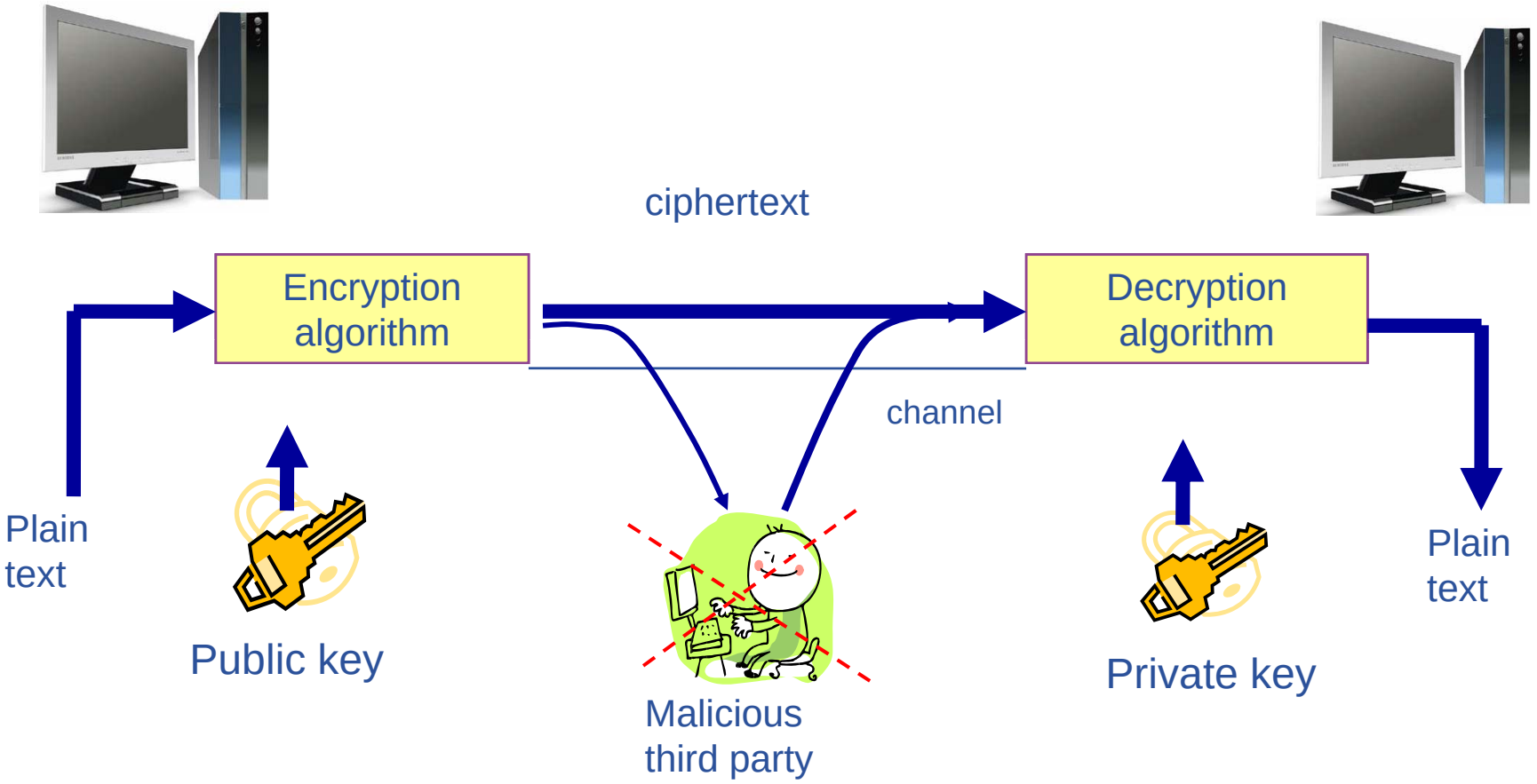


Storage Element

- **It provides uniform access to storage resources**
  (it may control simple disk servers, large disk arrays or Mass Storage Systems (MSS)
- **Each site may provide one or more SEs**



Cryptography components

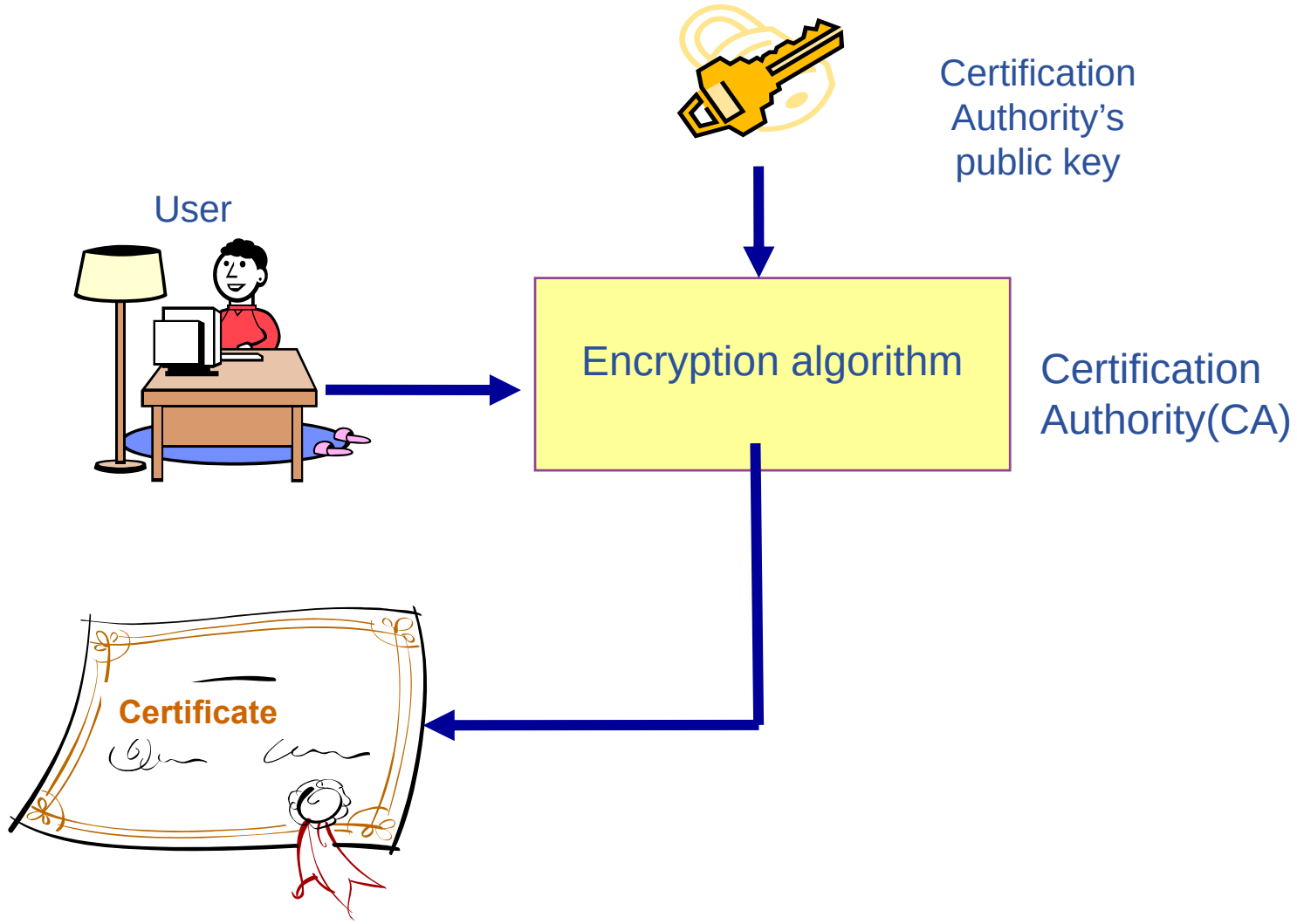


Digital certificate X.509

- **Each entity (user, resource) must obtain a certificate**
- **The certificate includes information, such as the expiration date, the Certification Authority that signed it, the owner's public key and a DN**
- **The DN defines uniquely the owner and has the following fields:**

C = Owner's country
O = Owner's organization
OU = Owner's group
CN = Owner's name

Certificate authority



Proxy certificates

- A new temporal certificate created taking into account the issued certificate by the corresponding CA
  ⇨ a new key pair is created to be used during the period that the proxy is valid

- The new private key is not secured by a password

- The use of a proxy is recommended because:

  ✓ the *proxy* has a short lifetime
  ✓ uses a different private key from the issued certificate

VOMS

- **Virtual Organisation Membership Service (VOMS)**
  - A system which allows a proxy to have extensions containing information:
    - About the VO
    - The groups the user belongs to in the VO
    - Any roles the user is entitled to have

- ***Group*: subset of the VO containing members who share some responsibilities or privileges in the project**
  - Hierarchically organised
  - A user can be a member of any number of groups
  - VOMS proxy contains the list of all groups the user belongs to
  - Group ⇨ privileges the user **ALWAYS** has

- ***Role*: Attribute which typically allows a user to acquire special privileges to perform specific tasks**
  - Role ⇨ privileges the user needs to have only from time to time

Information System (IS)

- **It provides information about the Grid resources and their status**
  ⇨ This information is essential for the operation of the whole Grid
- **Location of availiable Computing Elements to run jobs**
- **Finding of SEs that holding replicas of Grid files and the catalogs keeping the information on these files**
- **The information is stored in databases**
- **The published information is used for**
  - ✓ monitoring purposes ⇨ for analyzing usage and performance of the Grid, detecting fault situations and any other interesting events
  - ✓ accounting purposes ⇨ for creating statistics of the applications run by the users in the resources

MDS

- **Globus Moinitoring and Discovery service**
- **Resource Discovery and publishing the resource status**
- **OpenLDAP** which is an open source implementation of the ***Lightweight Directory Access Protocol (LDAP)***, a specialised database optimised for reading, browsing and searching information
- **Hierarchical architecture:**
  - In every resource runs a ***Grid Resource Information Server (GRIS)*** providing relevant information about the resource
  - At each site runs a ***Site Grid Information Server (GIIS)*** that collects information from the local GRISes and republishes it. The GIIS uses a ***Berkeley Database Information Index (BDII)*** to store data
  - A BDII is used to read from a group of sites, depicting a view of the overall Grid resources (on top of the hierarchy)

Getting started

- **Obtaining a certificate**
- **Registering with LCG / EGEE**
- **Choosing a VO**
- **Accounts for the training events:**
  - ssh ui01.isabella.grnet.gr (Putty)
    - login as: **egee04 – egee50**

Installing the certificate

- ✓ **[egee01@ui01 egee01]$ mkdir .globus**
  - Create directory .globus under the user home directory

- ✓ **[eg**
  - **em**
  - em

- ✓ **[ege**
  - **em**

- ✓ **[eg**
  - **n**

- **Prepare certificates for the training event only:**

- **[egee01@ui01 egee01]$ ./preparecerts.sh**

- **[egee01@ui01 egee01]$ ls -l ~/.globus**

```
total 12
-r--r--r--   1 egee01   training     5535 Sep 14 16:55 usercert.pem
-r--------   1 egee01   training      963 Sep 14 16:55 userkey.pem
```

- ✓ **[egee01@ui01 egee01]$ chmod 400 ~/.globus/userkey.pem**
  - The key must be readable only by the user

Certificate Info

- **Retrieving information of the user certificate**

✓ **[egee01@ui01 egee01]$ grid-cert-info**

```
Certificate:
Data:
    Version: 3 (0x2)
    Serial Number: 1788 (0x6fc)
    Signature Algorithm: sha1WithRSAEncryption
    Issuer: C=GR, O=HellasGrid Demos, OU=Certification Authoritites, CN=HellasGrid
  Demo CA 2006
    Validity
        Not Before: Sep 14 11:25:01 2007 GMT
        Not After :  Sep 29 11:25:01 2007 GMT
    Subject: C=GR, O=HellasGrid Demos, OU=People, L=Thessaloniki, CN=User 1788
    Subject Public Key Info:
        Public Key Algorithm: rsaEncryption
        RSA Public Key: (1024 bit)
            Modulus (1024 bit):
                00:c6:2e:31:bb:14:12:27:c3:a7:74:1a:48:3a:59:
```

Proxy certificate

- **Creating a proxy**

✓ [egee01@ui01 egee01]$ **voms-proxy-init --voms=hgdemo**

Your identity: /C=GR/O=HellasGrid Demos/OU=People/L=Thessaloniki/CN=User 1978

Enter GRID pass phrase:

Creating temporary proxy ..... Done

Contacting voms.grid.auth.gr:15030

[/C=GR/O=HellasGrid/OU=auth.gr/CN=voms.grid.auth.gr] "hgdemo" Done

Creating proxy ..... Done

Your proxy is valid until Mon Sep 17 00:48:09 2007

- **Destroying a proxy**

✓ [egee01@ui01 egee01]$ **voms-proxy-destroy**

Job Description Language

- A high-level language based on the ***Classified Advertisement (ClassAd) language***
- JDL describes jobs and aggregates of jobs with arbitrary dependency relations
- JDL specifies the desired job characteristics and constraints, which are taken into account by the WMS to select the best resource to execute the job
- A JDL file consists of lines having the format:

  attribute = expression;

  - Expressions can span several lines, but only the last one must be terminated by a semicolon
  - Literals are enclosed in double quotes
  - " in strings must be escaped with a backslash ("\"Hallo")
  - The character " ' " cannot be used in the JDL
  - Comments of each line begin with # or //
  - Multi-line comments must be enclosed between "/*" and "*/"
  - **No blank characters or tabs should follow the semicolon at the end of a line**

Attributes (1)

Executable	✓ The value of this attribute is the executable filename or the command to be run by the job ✓ If the command is already present on the WN, it must be expressed as a absolute path
StdOutput	✓ The name of the files containing the standard output
StdError	✓ The name of the files containing the standard error
StdInput	✓ The names of the files used as Input files
OutputSandbox	✓ The files to be transferred back to the UI after the job is finished
Environment	✓ Modifies the shell environment of the job
Virtual Organisation	✓ Explicitly specify the VO of the user
Requirements	✓ Expresses constraints on the resources where the job should run ✓ Its value is a Boolean expression that must evaluate to true for a job to run on that specific CE (example: Requirements = other.GlueCEInfoLRMSType == "PBS" && other.GlueCEInfoTotalCPUs > 1;)

Attributes (2)

RetryCount MaxRetryCount	✓ Times that the WMS automatically resubmitts jobs which failed for some reason (deep resubmission ⇨ when the job failed after started running in a WN)
ShallowRetryCount MaxShalowRetryCount	✓ Times that the WMS automatically resubmitts jobs which failed for some reason (shallow resubmission – gLite)
MyProxyServer	✓ The Proxy server to be used for certificate renewal
Rank	✓ The CE with the highest rank is selected by the WMS to execute a job ✓ by default *Rank = other.GlueCEStateEstimatedResponseTime* (but other.GlueCEStateFreeCPUs other.GlueCEStateWaitingJobs)

A simple job

✓ [egee01@ui01 egee01]$ less testJob1.sh

```
#!/bin/bash
echo "****** Running...  date ****** "
date
echo "****** Running... hostname ******"
hostname
echo "****** Running... pwd ****** "
pwd
echo "****** Running... ls ****** "
ls -l
echo "****** Running... uptime ****** "
uptime
echo "****** Learn your process ****** "
ps aux | grep home
echo "****** Running... ls ****** "
ls -l
echo "****** Printing Input files ****** "
echo "First file:"
cat $1 > >merge.out
echo "Second file:"
cat $2 >> merge.out
```

A simple JDL file

- **[egee01@ui01 egee01]$ less testJob1.jdl**

```
Executable = "testJob.sh";
Arguments = "fileA fileB";
StdOutput = "std.out";
StdError = "std.err";
InputSandbox = {“./testJob.sh", “./fileA", “./fileB"};
OutputSandbox = {"std.out", "std.err",  merge.out"};
```

Job execution

WMS

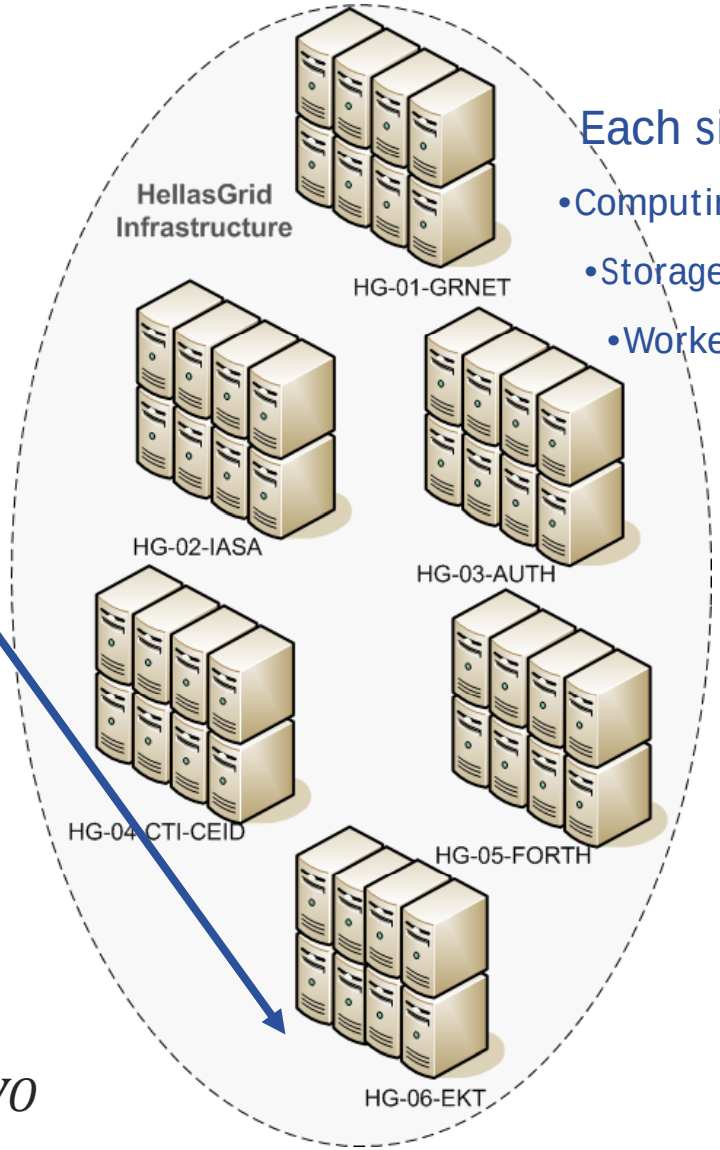
•Choose an available site for the job execution

The user connects to User Interface using a SSH-client

ui01.isabella.grnet.gr

• User's personal account and its certificate

•Command Line Interface for job submission



Each site has:

•Computing Element

•Storage Element

•Worker Nodes

*Job flow for **hgdemo** VO*

Job matching to CEs

- **Listing computing elements that match a job description**
  - ✓ **[egee01@ui01 egee01]$ glite-wms-job-list-match -a testJob1.jdl**

```
Connecting to the service https://wms01.egee-see.org:7443/glite_wms_wmproxy_server

================================================================================

                    COMPUTING ELEMENT IDs LIST
 The following CE(s) matching your job requirements have been found:

         *CEId*
 - ce01.afroditi.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
 - ce01.ariagni.hellasgrid.gr:2119/jobmanager-lcgpbs-hgdemo
 - ce01.athena.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
 - ce01.isabella.grnet.gr:2119/jobmanager-pbs-hgdemo
 - ce01.kallisto.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
 - ce01.marie.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
 - ce02.athena.hellasgrid.gr:2119/blah-pbs-hgdemo
 - ce02.marie.hellasgrid.gr:2119/jobmanager-pbs-hgdemo
 - glite-ce01.marie.hellasgrid.gr:2119/blah-pbs-hgdemo
 - node001.grid.auth.gr:2119/jobmanager-pbs-hgdemo

================================================================================
```

Job submission

- **Single Job submission**
  - ✓ **[egee01@ui01 egee01]$ glite-wms-job-submit -o jobId -a testJob1.jdl**

- **Connecting to the service https://wms01.egee-see.org:7443/glite_wms_wmproxy_server**

========== glite-wms-job-submit Success ==========

The job has been successfully submitted to the WMProxy
Your job identifier is:

https://wms01.egee-see.org:9000/6INrYSPP4XfkgTYHuqHuww

The job identifier has been saved in the following file:
/home/training/egee01/jobId

==========

✓ **glite-wms-job-submit –o jobId –r ce01.isabella.grnet.gr:2119/jobmanager-pbs-hgdemo -a testJob.jdl**
  - -r : sends the job directly to the specified CE

Job status' retrieval

- **Retrieving the status of a job**
  - ✓ [egee01@ui01 egee01]$ **glite-wms-job-status -i jobId**

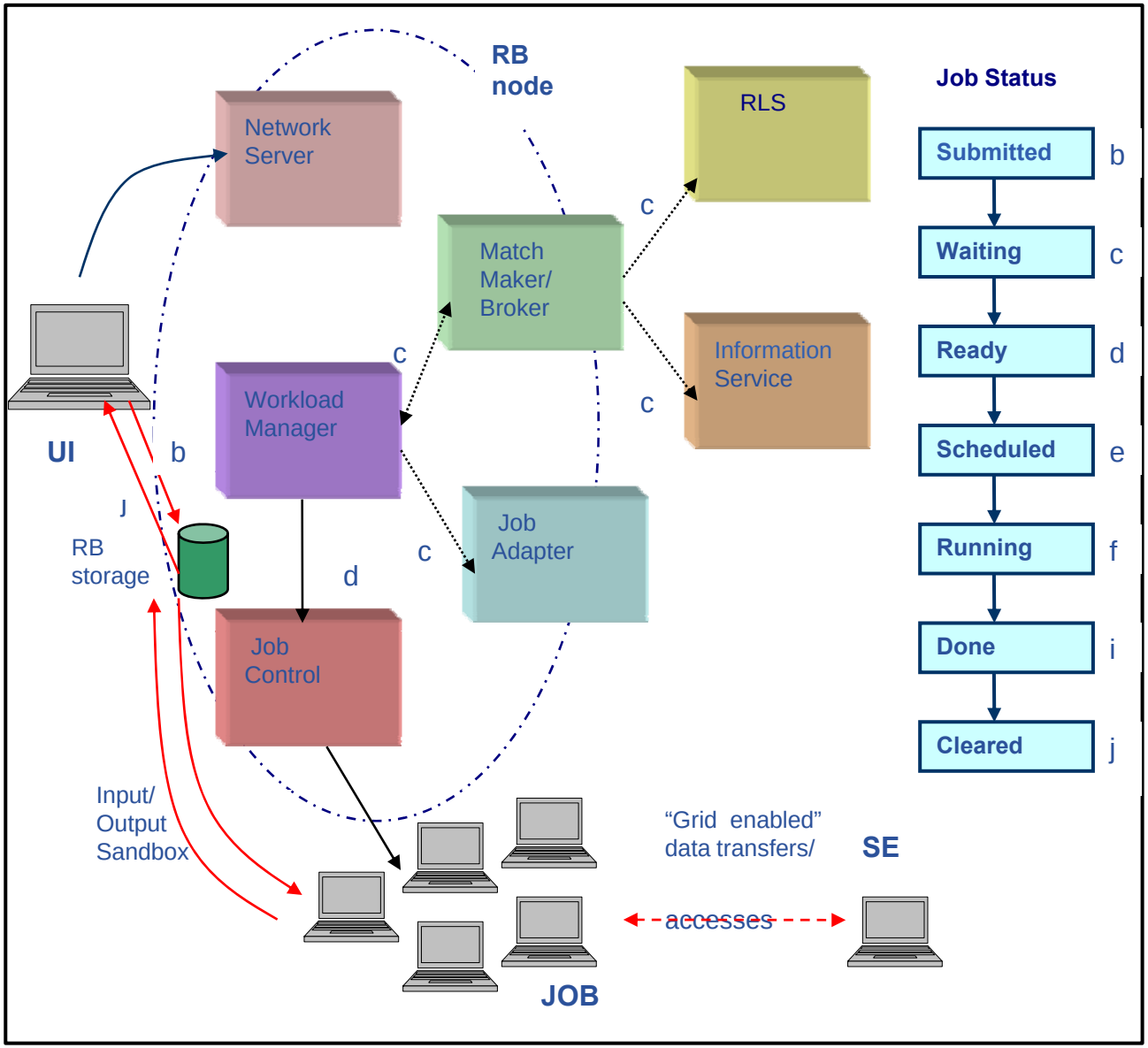
```
**********************************************************************
BOOKKEEPING INFORMATION:

Status info for the Job : https://wms01.egee-see.org:9000/u-
Slc372Ny_DQ04reimrHw
Current Status:     Scheduled
Status Reason:      Job successfully submitted to Globus
Destination:        ce01.ariagni.hellasgrid.gr:2119/jobmanager-
lcgpbs-hgdemo
Submitted:          Tue Sep 18 03:16:47 2007 EEST
**********************************************************************
```

  :

  - ✓ [egee01@ui01 egee01]$ **watch "glite-job-status -i jobId"**

(To exit ctrl + C)

Job flow (1)


RB node
Network Server
RLS
Match Maker/ Broker
Information Service
Workload Manager
Job Adapter
Job Control
RB storage
UI
b
c
c
c
c
d
J
Input/ Output Sandbox
JOB
"Grid enabled" data transfers/
SE
accesses
Job Status
Submitted
b
Waiting
c
Ready
d
Scheduled
e
Running
f
Done
i
Cleared
j

Job flow (2)

- **Job submission**
  - The user logs in the UI and submits the job to a Resource broker.
  - If one or more files need to be copied from the UI to the WN, this is specified in the job description and the files are initially copied to the RB. This set of files is called the **Input Sandbox**
  - **Job status ⇨ SUBMITTED**
- **Finding the proper CE**
  - The **WMS** interrogates the **Information Supermarket (ISM)** *(*an internal cache of information read from the BDII) , to determine the status of computational and storage resources.
  - The WMS interogates the File Catalogue to find the location of any required input files
  - **Job status ⇨ WAITING**
- **Job submission from the RB to the selected CE**
  - The RB prepares a wrapper script that will be passed together with other parameters, to the selected CE.
  - **Job status ⇨ READY**
- **Job arrival to the CE**
  - **The CE receives the request and sends the job for execution to the local LRMS.**
  - **Job status ⇨SCHEDULED**

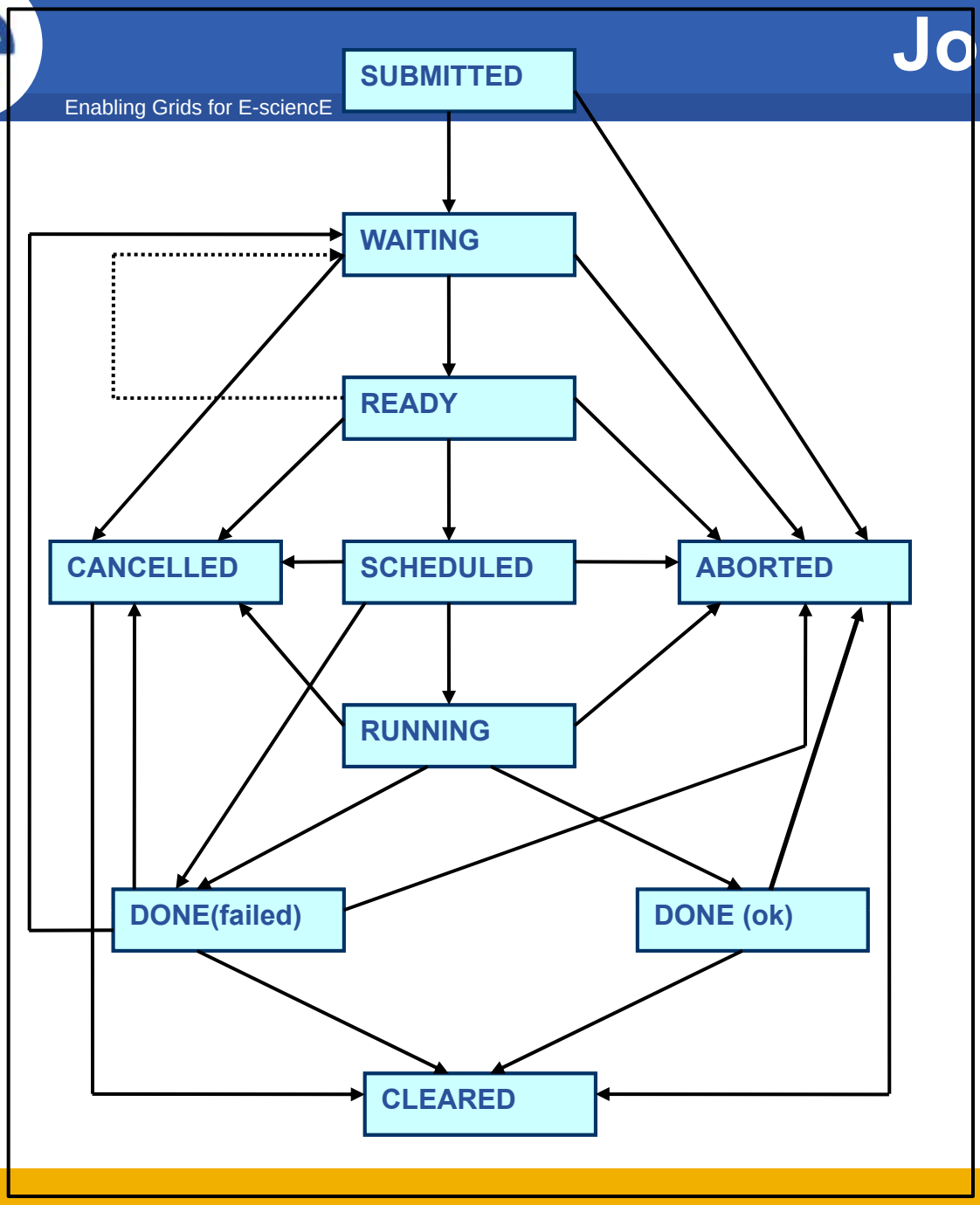
Job flow (3)

- **Job submission to the Worker node**
  - The LRMS handles the execution of jobs on the local Worker Nodes.
  - The Input Sandbox files are copied from the RB to an available WN where the job is executed.
  - While the job runs, Grid files can be directly accessed from an SE using either the RFIO or gsidcap protocol
  - Any new produced output files which can be uploaded to the Grid and made available for other Grid users to use. This can be achieved using the Data Management tools described later. Uploading a file to the Grid means copying it to a Storage Element and registering it in a file catalogue.
  - **Job status ⇨ RUNNING**

- **Job finished**
  - If the job ends without errors, the small output files specified by the user in the ***Output Sandbox*** are transferred back to the RB node.
  - **Job status ⇨ DONE**

- **Output retrieval**
  - The user can retrieve the output files to the UI
  - **Job status ⇨ Cleared**

Job status


SUBMITTED
WAITING
READY
CANCELLED
SCHEDULED
ABORTED
RUNNING
DONE(failed)
DONE (ok)
CLEARED

Job cancelling

- ## Cancelling a job

  ✓ **[egee01@ui01 egee01]$** **glite-wms-job-cancel -i jobId**

  Are you sure you want to remove specified job(s) [y/n]y : y

  Connecting to the service
  https://195.251.53.233:7443/glite_wms_wmproxy_server

  ================ glite-wms-job-cancel Success ==============

  The cancellation request has been successfully submitted for the following job(s):

  - https://wms01.egee-see.org:9000/E_Ykk3oGFkXTQcDi_XZs5w

  ==

Output retrieval

- If the job's status is DONE, then its output can be copied to the UI with the commands:
  - ✓ [egee01@ui01 egee01]$ **glite-wms-job-output -i jobId**

```
Connecting to the service
https://195.251.53.233:7443/glite_wms_wmproxy_server

==========================================================

                JOB GET OUTPUT OUTCOME

Output sandbox files for the job:
https://wms01.egee-see.org:9000/6INrYSPP4XfkgTYHuqHuww
have been successfully retrieved and stored in the directory:
/tmp/glite/glite-ui/egee01_6INrYSPP4XfkgTYHuqHuww

==========================================================
```

Q&A

Thank you !

References

- **Commandline Tutorial**
  - *http://wiki.egee-see.org/index.php/Programming_from_the_Command_Line*