The EGEE middlewares and the GILDA t-Infrastructure

Roberto Barbera
University of Catania and INFN
ISSGC05
Vico Equense, 20.07.2005

www.eu-egee.org

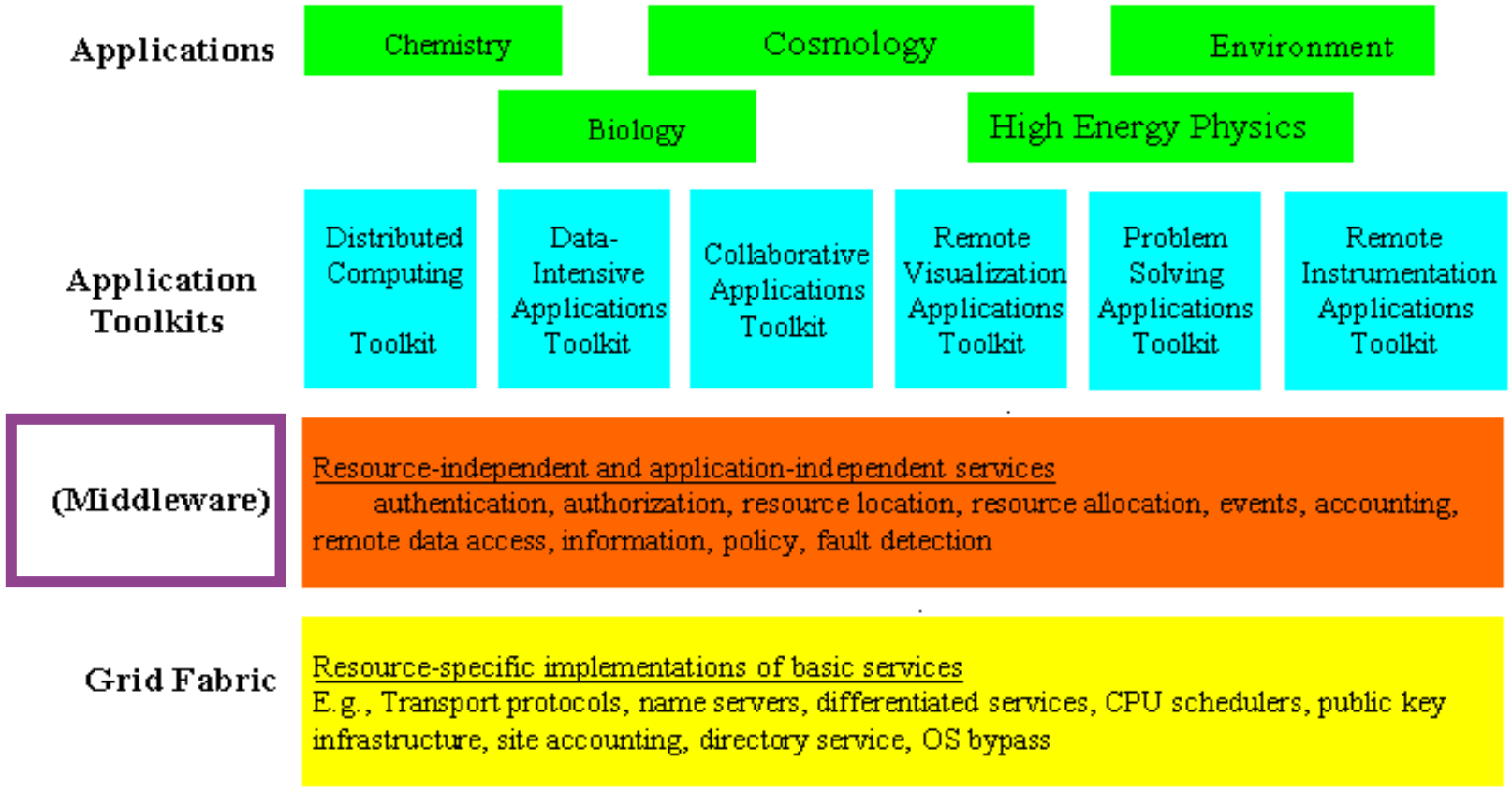
Information Society

Outline

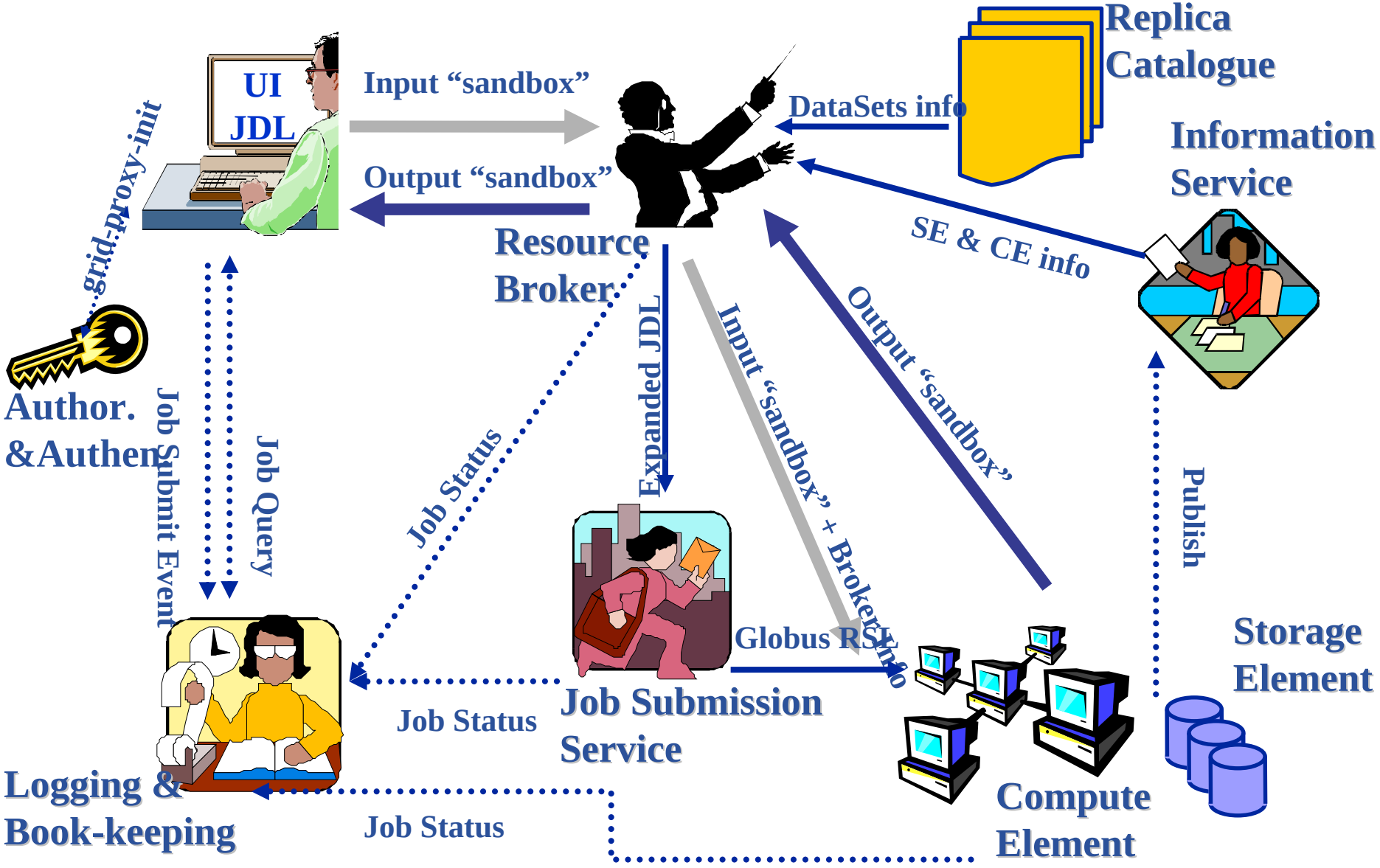
- **Generalities**
- **Security System**
  - GSI
  - VOMS
  - MyProxy
- **Information System**
  - lcg-infosites
  - R-GMA
- **Workload Management System**
- **Data Management System**
  - LFC
  - FiReMan
- **The GILDA t-Infrastructure**
  - services
  - tools
  - applications
  - tutorial lay-out
- **Summary and conclusions**

Generalities

The Grid from a Services View



A typical job workflow



EGEE middlewares face to face

LCG (the present)

- **Security**
  - GSI
- **Job Management**
  - Condor + Globus
  - CE, WN
  - Logging & Bookkeeping
- **Data Management**
  - LCG services
- **Information & Monitoring**
  - BDII (evolution of MDS)
- **Grid Access**
  - CLI + API

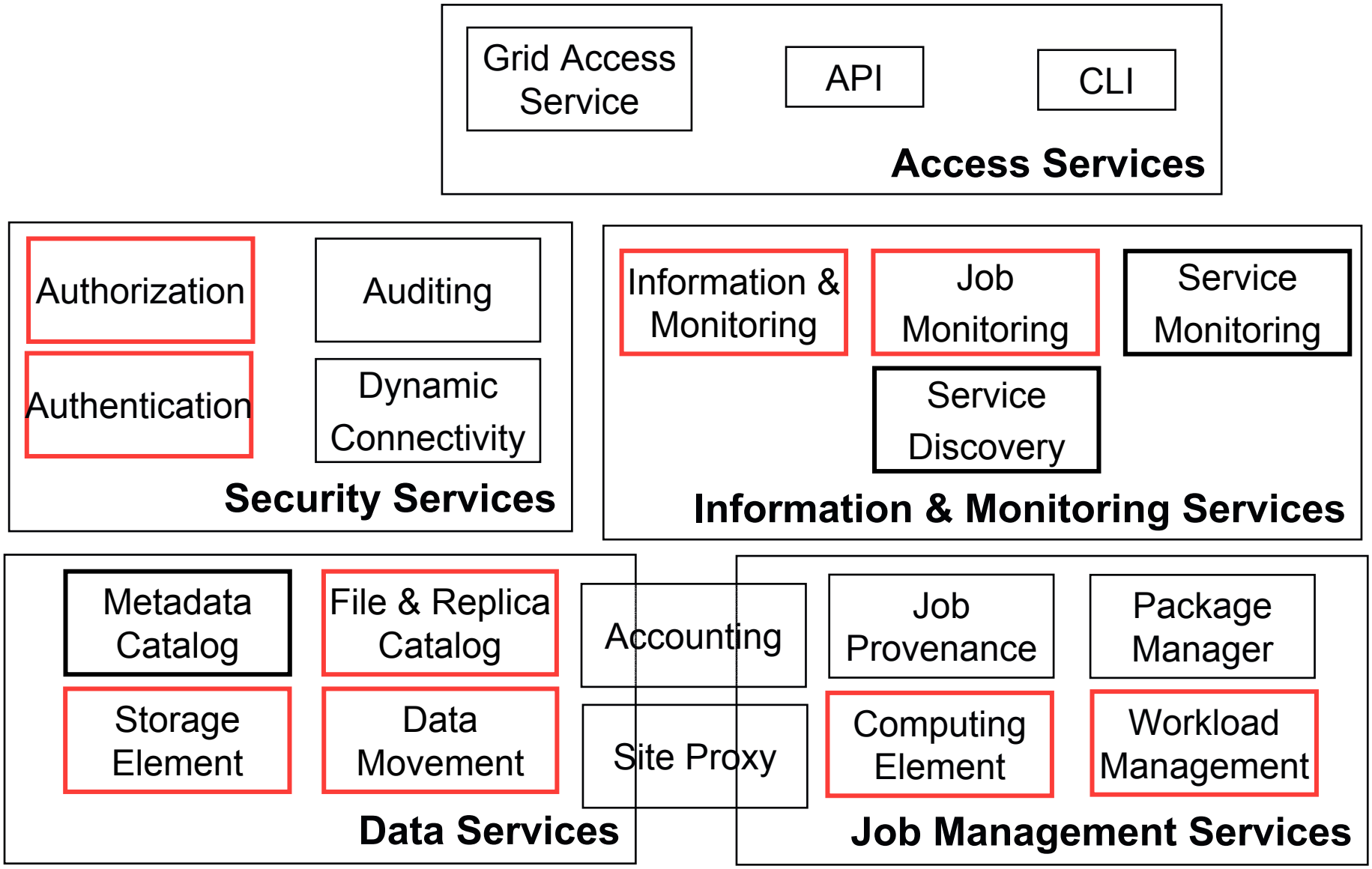
gLite (the future)

- **Security**
  - GSI and VOMS
- **Job Management**
  - Condor + Globus + blahp
  - CE, WN
  - Logging & Bookkeeping
  - Job Provenance
  - Package management
- **Data Management**
  - LFC
  - gLite-I/O + FiReMan
- **Information & Monitoring**
  - BDII
  - R-GMA + Service Discovery
- **Grid Access**
  - CLI + API + Web Services

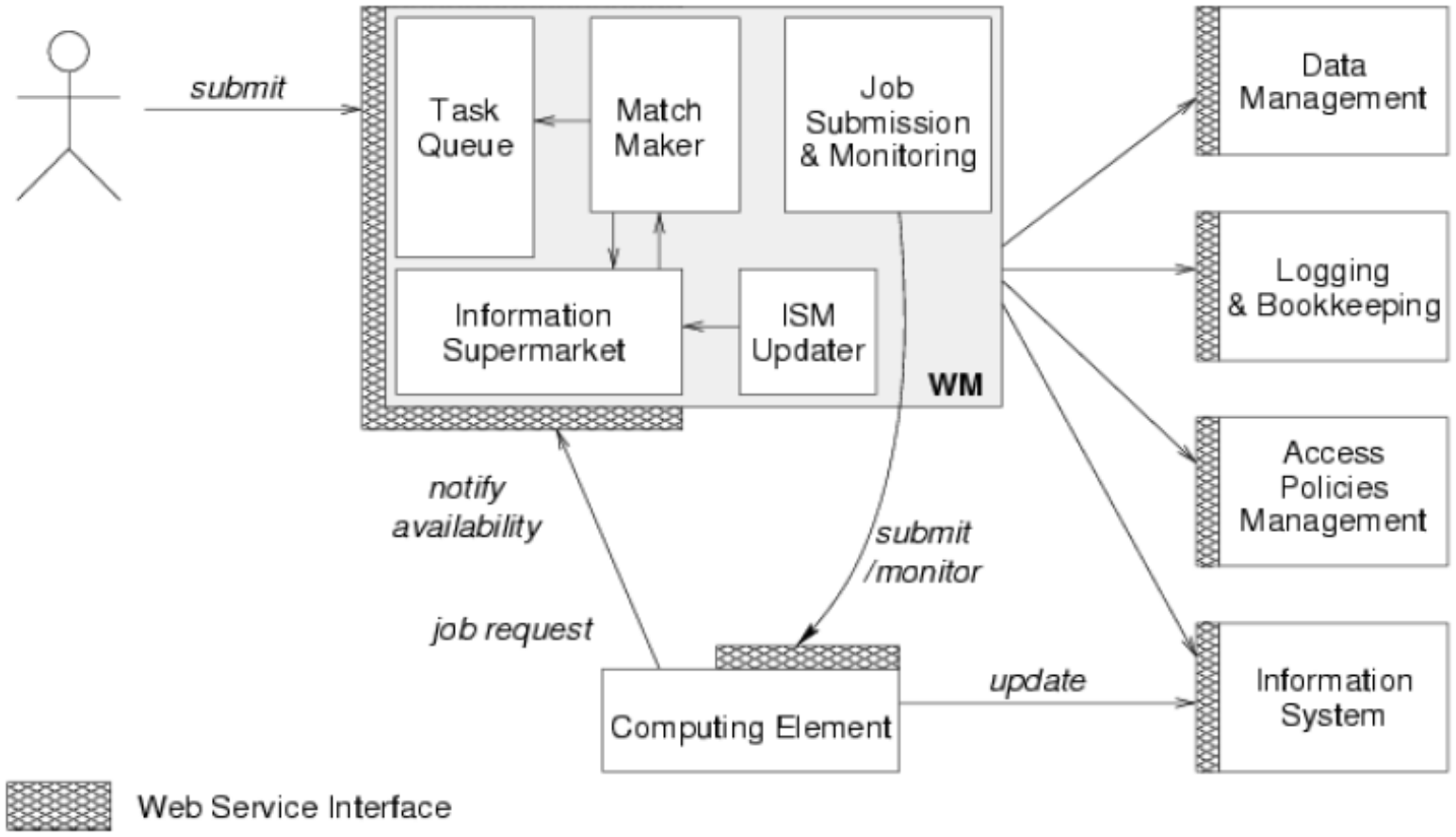
Overview of EGEE Middleware

- The gLite Grid services follow a *Service Oriented Architecture*
  - **facilitate interoperability among Grid services**
  - **allow easier compliance with upcoming standards**
- Architecture is not bound to specific implementations
  - **services are expected to work together**
  - **services can be deployed and used independently**
- The gLite service decomposition has been largely influenced by the work performed in the LCG project

gLite components overview



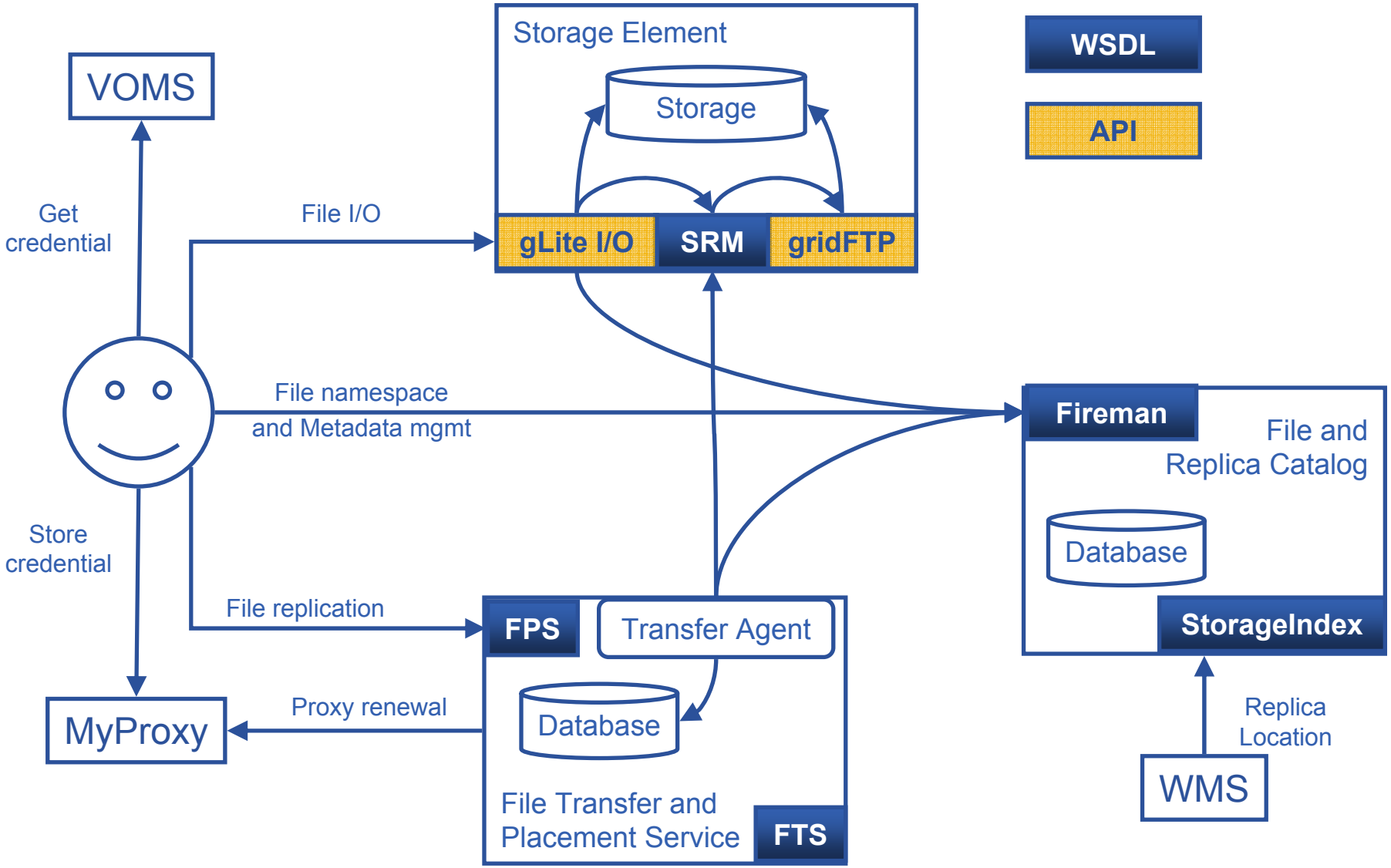
Job Management Services



Data Management Services

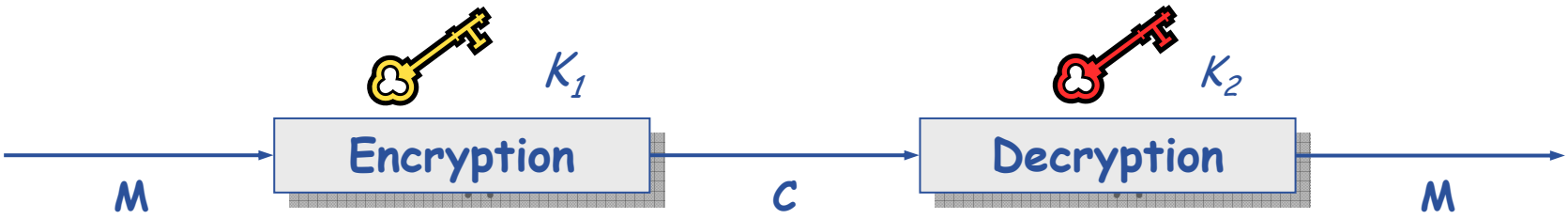
- **Storage Element**
  - Storage Resource Manager
  - POSIX-I/O
  - Access protocols gsiftp, https, rfio, file, …
- **Catalogs**
  - File Catalog
  - Replica Catalog
  - File Authorization Service
  - Metadata Catalog
- **File Transfer**
  - File Transfer Service
  - File Placement Service

Data Management Interactions



Security System
(GSI, VOMS, and MyProxy)

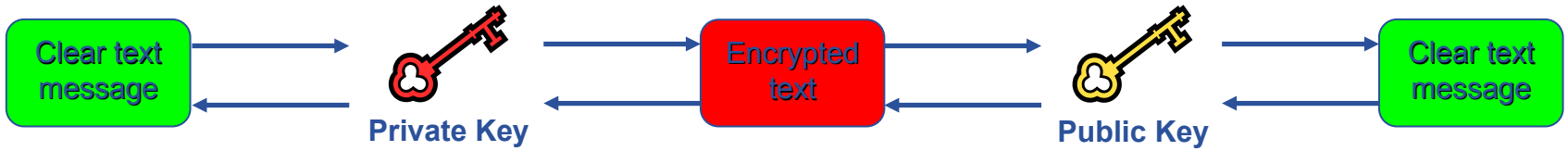
Cryptography



- **Mathematical algorithm that provides important building blocks for the implementation of a security infrastructure**
- **Symbology**
  - Plaintext: *M*
  - Cyphertext: *C*
  - Encryption with key K_1 : $E_{K_1}(M) = C$
  - Decryption with key K_2 : $D_{K_2}(C) = M$
- **Algorithms**
  - **Symmetric**: $K_1 = K_2$
  - **Asymmetric**: $K_1 \neq K_2$

Public Key Infrastructure

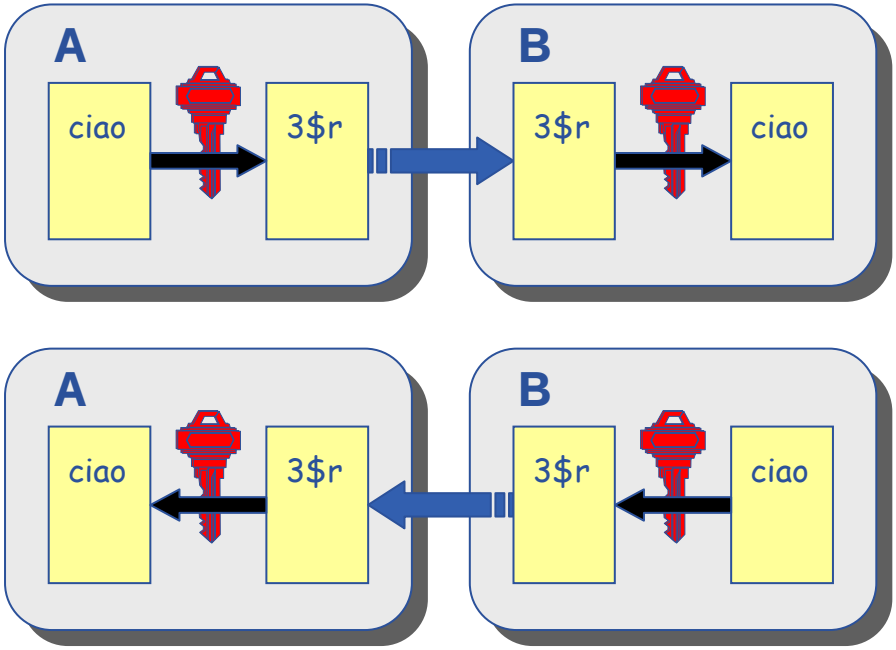
- **Provides authentication, integrity, confidentiality, non-repudiation**
- **Asymmetric encryption**



- **Digital signatures**
  - A hash derived from the message and encrypted with the signer's private key
  - Signature checked decrypting with the signer's public key
- **Allows key exchange in an insecure medium using a trust mode**
  - Keys trusted only if signed by a trusted third party (Certification Authority)
  - A CA certifies that a key belongs to a given principal
- **Certificate**
  - Public key + principal information + CA signature
  - X.509 format most used
- **PKI used by SSL, PGP, WS security, S/MIME, etc.**

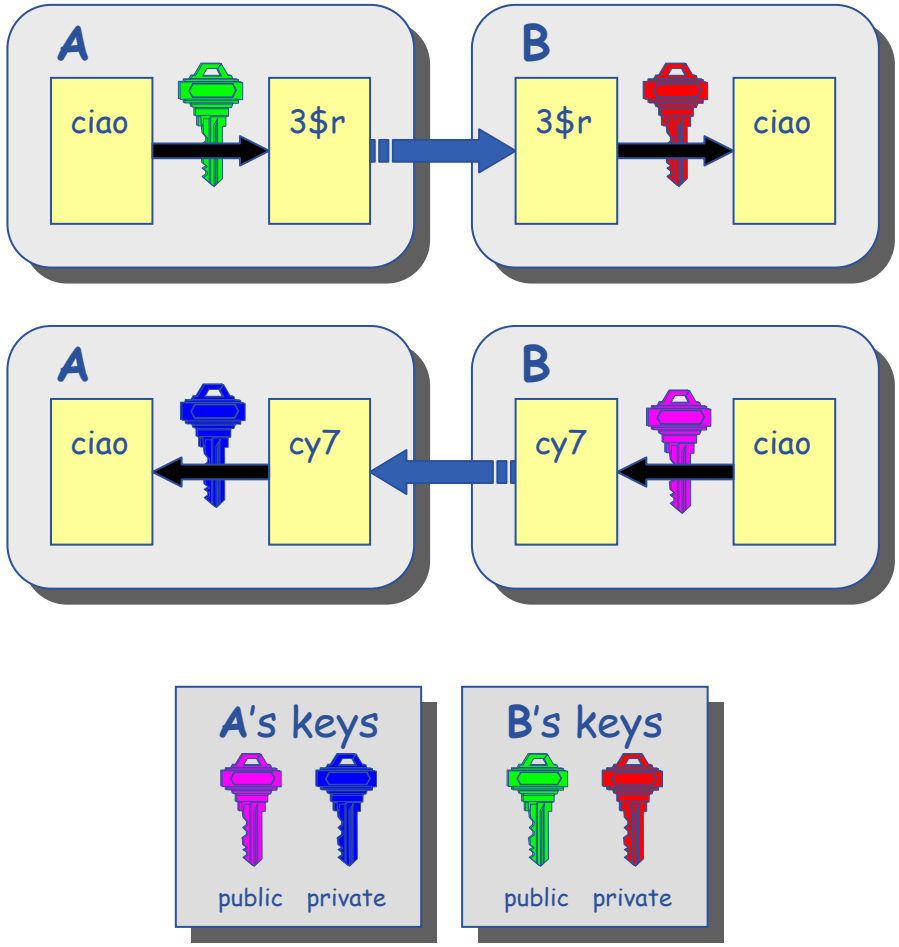
Symmetric Algorithms

- **The** same **key is used for encryption and decryption**
- **Advantages:**
  - Fast
- **Disadvantages:**
  - how to distribute the keys?
  - the number of keys is $O(n^2)$
- **Examples:**
  - **DES**
  - **3DES**
  - **Rijndael** (AES)
  - **Blowfish**
  - **Kerberos**



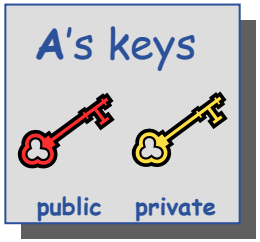
Public Key Algorithms

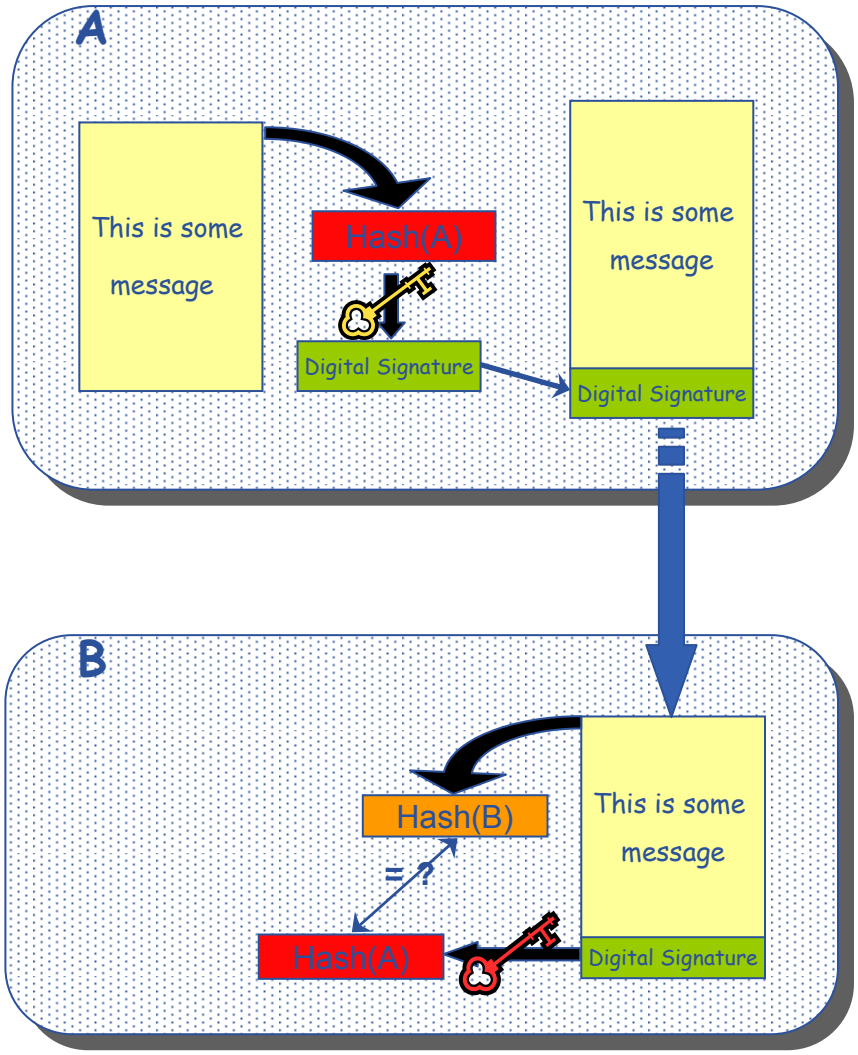
- **Every user has two keys: one *private* and one *public*:**
  - it is *impossible* to derive the private key from the public one;
  - a message encrypted by one key can be decripted **only** by the other one.
- **No exchange of secrets is necessary**
  - the sender cyphers using the *public* key of the receiver;
  - the receiver decripts using his *private* key;
  - the number of keys is O(n).
- **Examples:**
  - **Diffie-Helmann** (1977)
  - **RSA** (1978)



Digital Signature

- A calculates the *hash* of the message
- A encrypts the hash using his *private* key: the encrypted hash is the *digital signature*.
- A sends the signed message to B.
- B calculates the hash of the message and *verifies* it with the one received by A and decyphered with A's *public* key.
- If the two hashes are equal, the message wasn't modified and A cannot repudiate it.

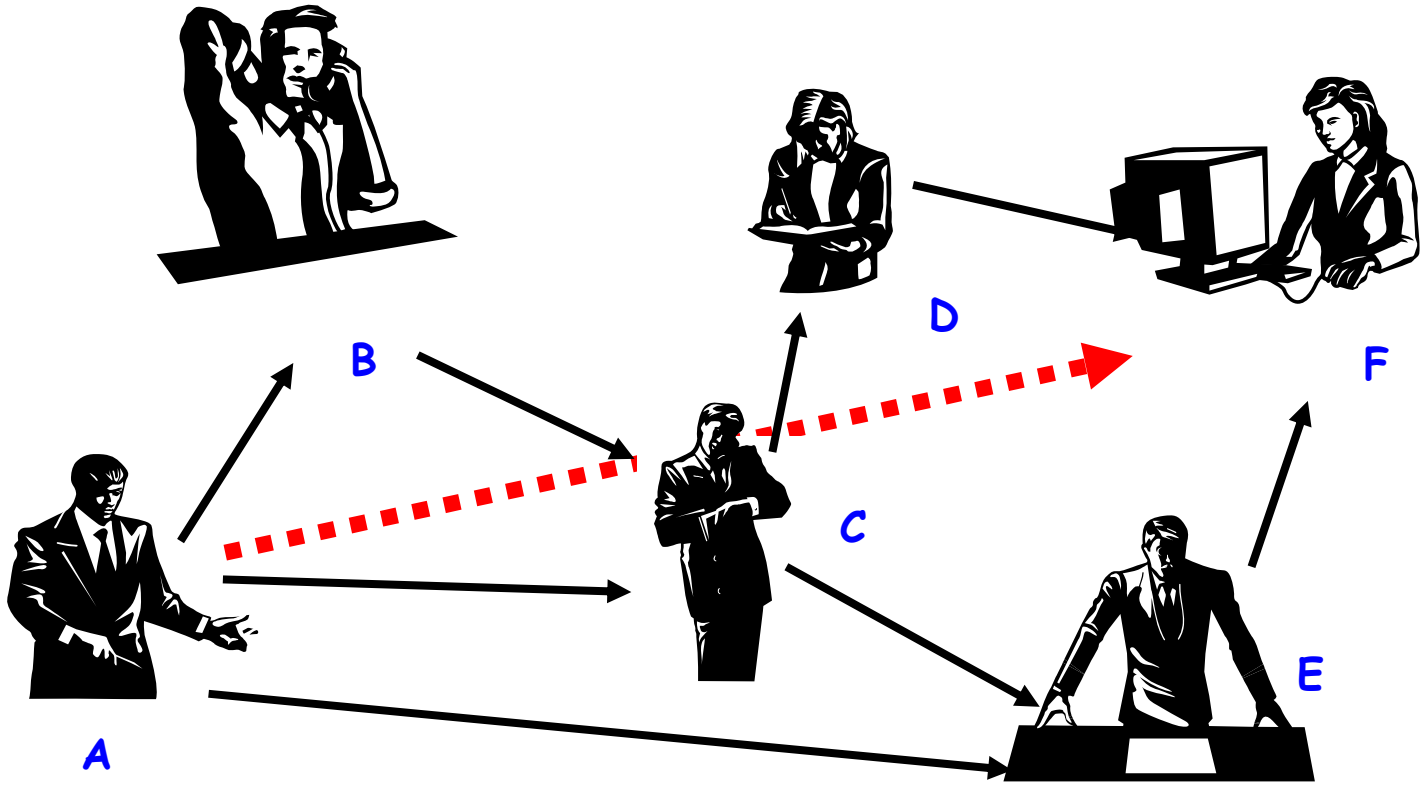




Digital Certificates

- **A's digital signature is safe if:**
  1. A's private key is not compromised
  2. B knows A's public key
- **How can B be sure that A's public key is really A's public key and not someone else's?**
  - A *third party* guarantees the correspondence between public key and owner's identity, by signing a document which contains the owner's identity and his public key (**Digital Certificate**)
  - Both A and B must trust this third party
- **Two models:**
  - PGP: "web of trust";
  - X.509: hierarchical organization.

PGP “web of trust”



- F knows D and E, who knows A and C, who knows A and B.
- F is reasonably sure that the key from A is really from A.

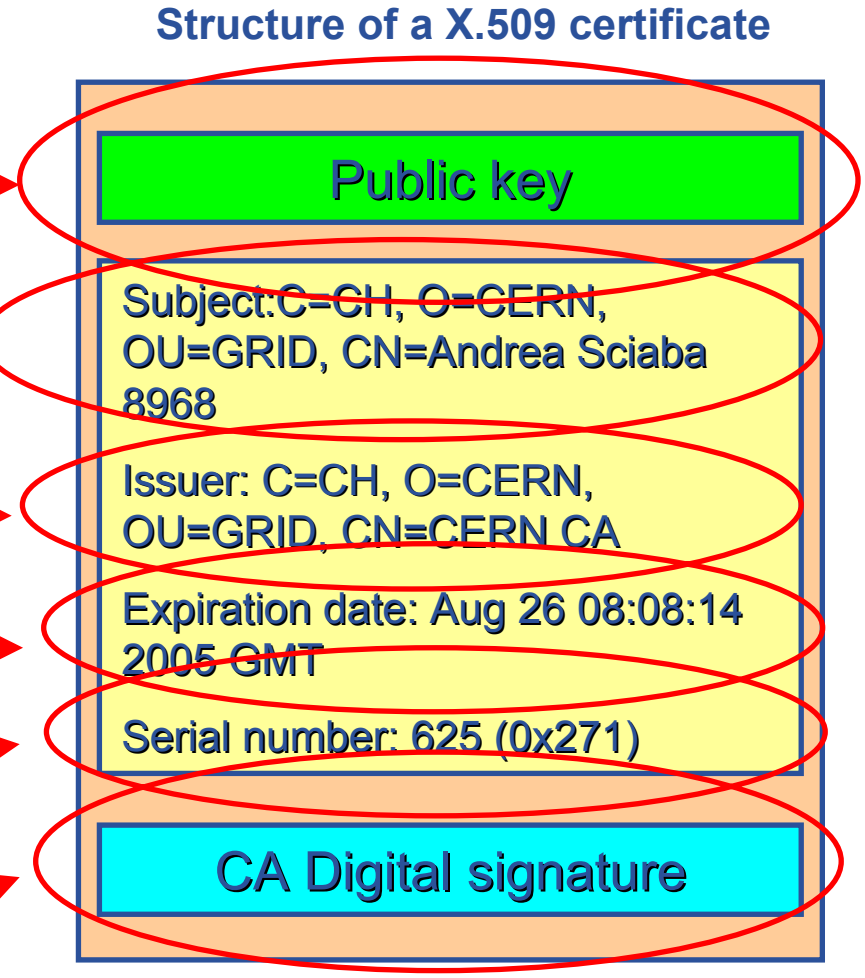
X.509

The "third party" is called *Certification Authority* (CA).

- **Issue certificates for users, programs and machines**
- **Check the identity and the personal data of the requestor**
  - Registration Authorities (RAs) do the actual validation
- **CA's periodically publish a list of compromised certificates**
  - **Certificate Revocation Lists** (CRL)
    - They contain all the revoked certificates yet to expire
  - **Online Certificate Status Protocol** (OCSP).
- **CA certificates are self-signed**

X.509 Certificates

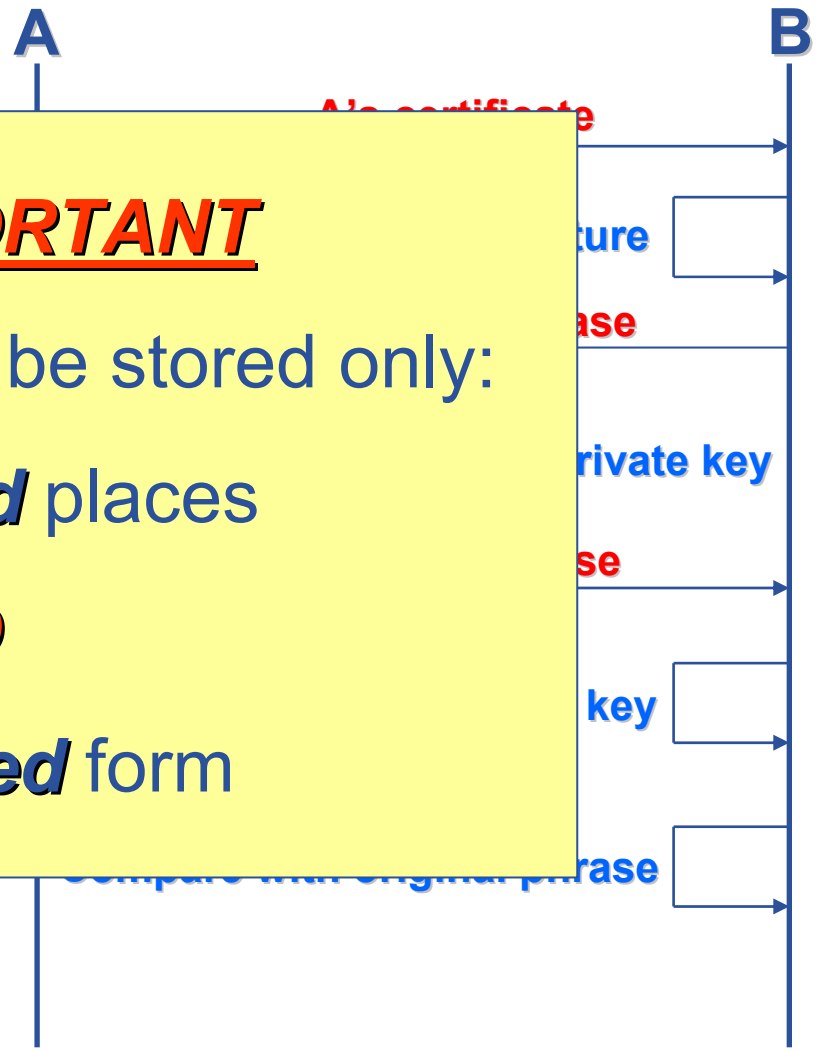
- **An X.509 Certificate contains:**
  - owner's public key;
  - identity of the owner;
  - info on the CA;
  - time of validity;
  - Serial number;
  - digital signature of the CA



The Grid Security Infrastructure (GSI)

Based on X.509 PKI:

- **every us**
  certifica
- **certifica**
  the loca
- **every Gr**
  authentic
  1. A se
  2. B ve
  3. B se
  4. A en
     priva
  5. A se
  6. B us
     challenge.
  7. B compares the decrypted string with the original challenge
  8. If they match, B verified A's identity and A can not repudiate it.



VERY IMPORTANT

Private keys must be stored only:

in ***protected*** places

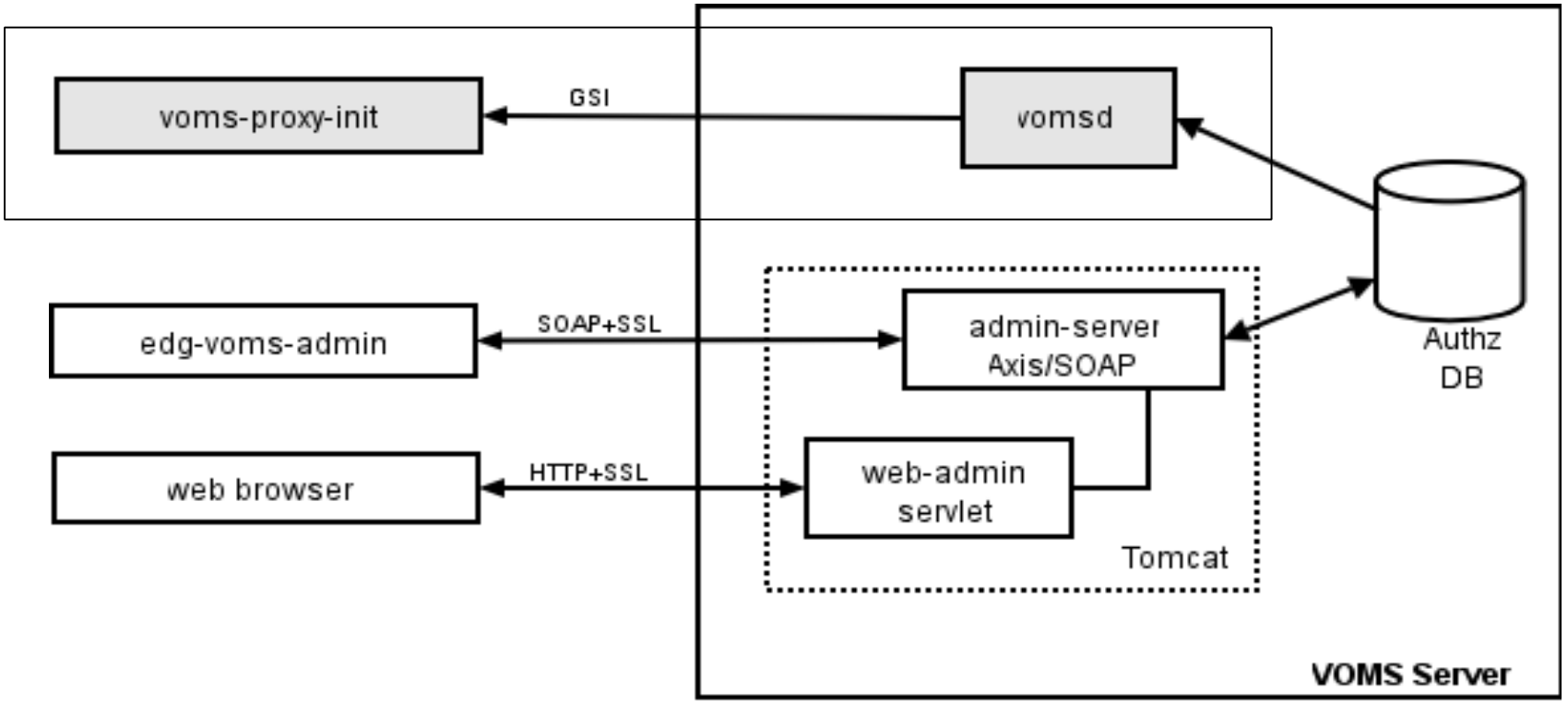
AND

in ***encrypted*** form

VOMS "at a glance"

- Virtual Organization Membership Service (VOMS) is a service that keeps track of the members of a VO and grants users authorization to access the resource at VO level, providing support for group membership, roles (e.g. administrator, sofware manager, student) and capabilities.
- Support for it is integrated in most of the grid services.
- Provide a secure system for VO to organize the user in groups and/or roles and to disseminate this information
- User should be able to decide which information wants to publish
- Compatibility with Globus Toolkit
- Each VO has its own server(s) containing groups membership, roles and capabilities information for each member
- User contacts the server requesting his authorization info
- The server sends the authorization info to the client
- The client includes it in a proxy certificate

VOMS - components



Authz DB is a RDBMS (both MySQL and Oracle are currently supported).

References

- VOMS
  - Available at http://infnforge.cnaf.infn.it/voms/
  - Alfieri, Cecchini, Ciaschini, Spataro, dell'Agnello, Fronher, Lorentey, From gridmap-file to VOMS: managing Authorization in a Grid environment
  - Vincenzo Ciaschini, A VOMS Attribute Certificate Profile for Authorization
- GSI
  - Available at www.globus.org
  - A Security Architecture for Computational Grids. I. Foster, C. Kesselman, G. Tsudik, S. Tuecke. *Proc. 5th ACM Conference on Computer and Communications Security Conference*, pp. 83-92, 1998.
  - A National-Scale Authentication Infrastructure. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer, V. Welch. *IEEE Computer*, 33(12):60-66, 2000.
- RFC
  - S.Farrell, R.Housley, An internet Attribute Certificate Profile for Authorization, RFC 3281

MyProxy

Consists of a server and a set of client tools that can be used to delegate and retrieve credentials to and from a server.

MyProxy Client commands:

- *myproxy-init*
- *myproxy-info* `// myproxy-info -s <host name> -d`
- *myproxy-destroy*
- *myproxy-get-delegation*

```
// myproxy-get-delegation -s <host name> -d
   –t <hours> -o <output file> -a <user proxy>
```

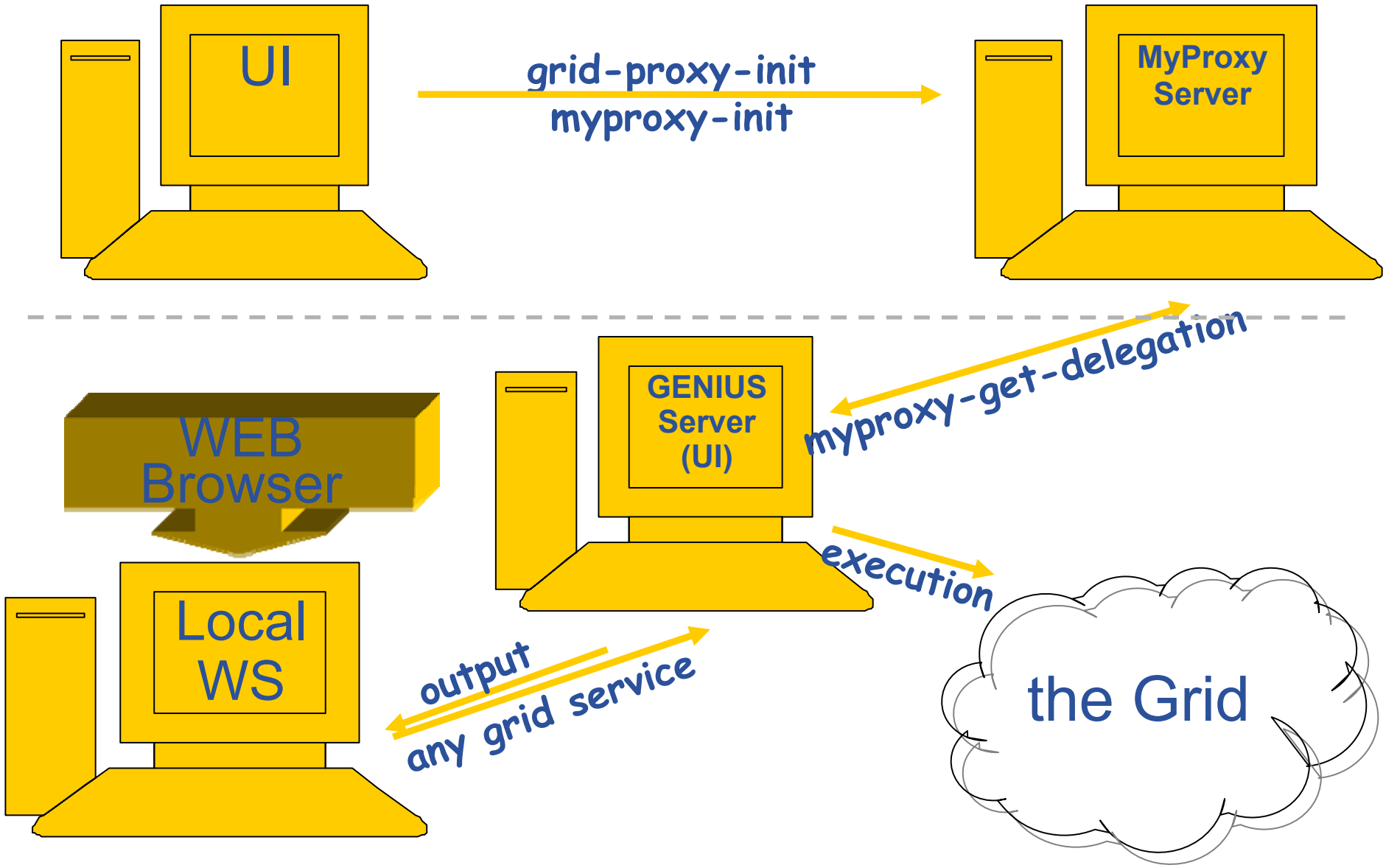
- *myproxy-change-pass-phrase*

The ***myproxy-init*** command allows you to create and send a delegated proxy to a MyProxy server for later retrieval; in order to launch it you have to assure you're able to execute the grid-proxy-init or vomsproxy-init command.

```
myproxy-init -s <host name> -t <hours> -d –n
```

The myproxy-init command stores a user proxy in the repository specified by <host name> (the –s option). Default lifetime of proxies retrieved from the repository will be set to <hours> (see -t) and no password authorization is permitted when fetching the proxy from the repository (the -n option). The proxy is stored under the same user-name as is your subject in your certificate (-d).

Grid authentication with MyProxy



Information System (lcg-infosites and R-GMA)

lcg-infosites
(the present)

Uses of the IS in EGEE/LCG

If you are a user

Retrieve information of Grid resources and status

Get the information of your jobs status

If you are a middleware developer

Workload Management System:
Matching job requirements and Grid resources

Monitoring Services:
Retrieving information of Grid Resources status and availability

If you are site manager or service

You “generate” the information for example relative to your site or to a given service

Elements behind the IS

```
*********************************************************************************
These are the data for alice: (in terms of CPUs)
*********************************************************************************
#CPU  Free   Total Jobs    Running   Waiting    Computing Element
---------------------------------------------------------------------------------
52     51       0            0         0   ce.prd.hp.com:2119/jobmanager-lcgpbs-long
16     14       3            2         1   lcg06.sinp.msu.ru:2119/jobmanager-lcgpbs-long
[……]
The total values are:
----------------------
10347  5565     2717       924       1793
```

She will use some EGEE/LCG tools and after few moments…

- Something has managed this information: (General IS architecture)
- Something has provided it: (Providers, Servers)
- It is following a certain "schema": (GLUE Schema)
- And she has accessed it following a protocol: (Access Protocol: LDAP)

The Information System Elements

MDS: Monitoring and Discovery Service

- Adopted from Globus
- It is the general architecture of EGEE/LCG to manage Grid information

General steps:

1st. At each site providers report static and dynamic service status to servers

2nd. A central system queries these servers and stores the retrieved information in a database

3rd. This information will be accessed through a given access protocol

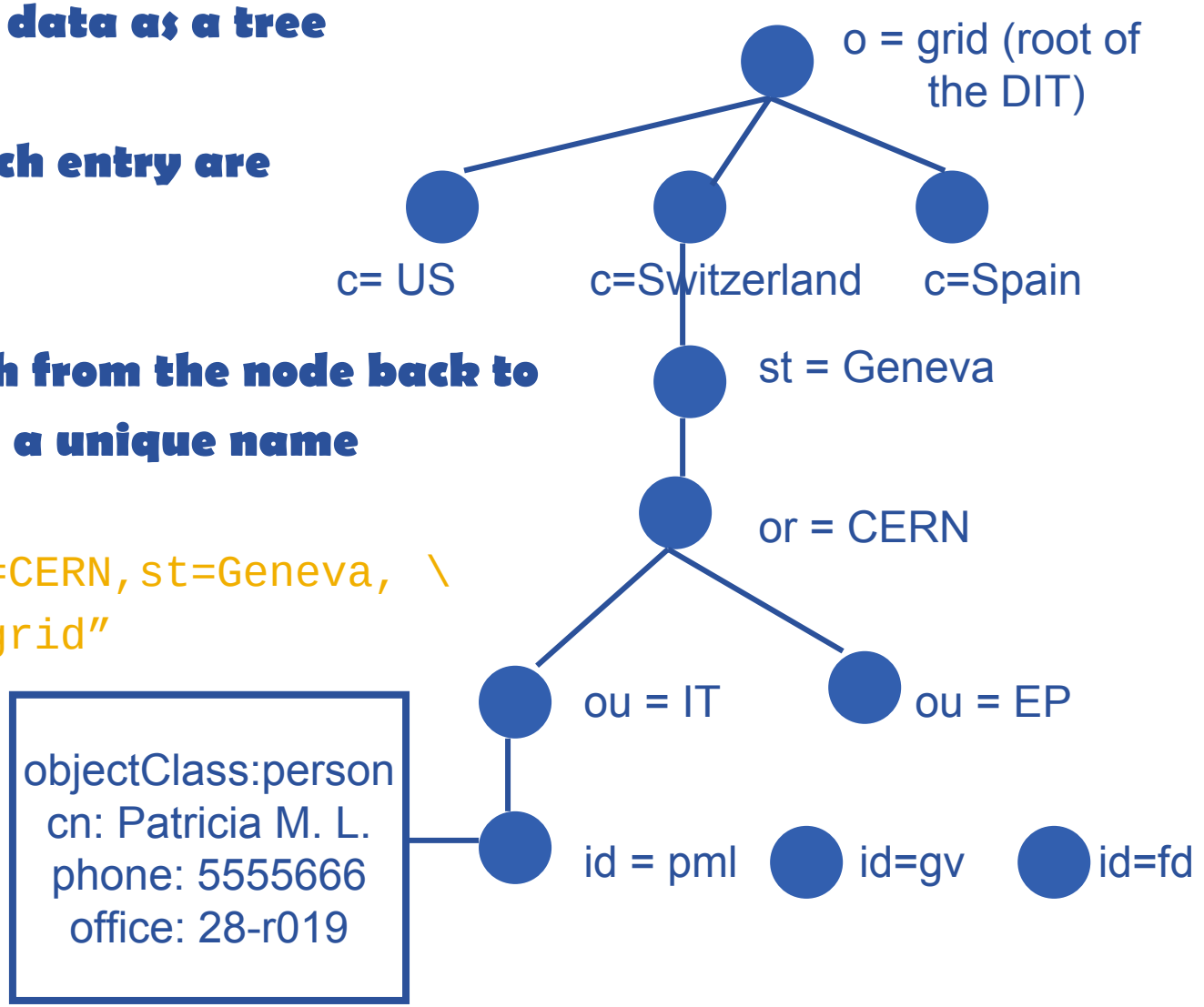
4th. The central system provides the information in a given schema

BDII (a MDS evolution) is the current EGEE/LCG Information System and it is based on LDAP

The LDAP Protocol

- **LDAP structures data as a tree**
- **The values of each entry are uniquely named**
- **Following a path from the node back to the root of the DIT, a unique name is built (the DN):**

```
"id=pml,ou=IT,or=CERN,st=Geneva, \
c=Switzerland,o=grid"
```



Implementation of IS in LCG-2

♠ lcg-infosites

- Already deployed in LCG-2 in the last release
- It is intended to be the most complete information retriever for the user:
  - ✓ Once he arrives at the Grid (on UIs)
  - ✓ To be used by the user applications (on WNs)
- Several versions of this script have been included in the software packages of ATLAS and the monitoring services of Alice (MonAlisa)
- You do not need a proxy

This will be tested during the hands-on session

lcg-infosites

```
> lcg-infosites --vo <your_vo> feature –-is <your_bdii>
```

- **It's mandatory to include the vo and the feature**
- **The –is option means the BDII you want to query. If not supplied, the BDII defined into the LCG_GFAL_INFOSYS will be interrogated**

Features and descriptions:

closeSE	Names of the CEs where the user's VO is allowed to run together with their corresponding closest SEs
ce	Number of CPUs, running and waiting jobs and names of the CEs
se	SEs names together with the available and used space
lrc (rmc)	Name of the lrc (rmc) for the user's VO
all	It groups all the features just described
help	Description of the script

lcg-infosites

```
> lcg-infosites --vo alice se --is lxb2006.cern.ch
```

```
************************************************
These are the data for alice: (in terms of SE)
************************************************
Avail Space (Kb)      Used Space (Kb)            SEs
-----------------------------------------------------------------
33948480              2024792                    se.prd.hp.com
506234244             62466684                   teras.sara.nl
1576747008            3439903232                 gridkap02.fzk.de
1000000000000         500000000000               castorgrid.cern.ch
304813432             133280412                  gw38.hep.ph.ic.ac.uk
651617160             205343480                  mu2.matrix.sara.nl
1000000000000         1000000000                 lcgads01.gridpp.rl.ac.uk
415789676             242584960                  cclcgseli01.in2p3.fr
264925500             271929024                  se-a.ccc.ucl.ac.uk
668247380             5573396                    seitep.itep.ru
766258312             681359036                  t2-se-02.lnl.infn.it
660325800             1162928716                 tbn17.nikhef.nl
1000000000000         1000000000000              castorftp.cnaf.infn.it
14031532              58352476                   lcgse01.gridpp.rl.ac.uk
1113085032            1034242456                 zeus03.cyf-kr.edu.pl
                     [… … … … …]
```

R-GMA

(the future)

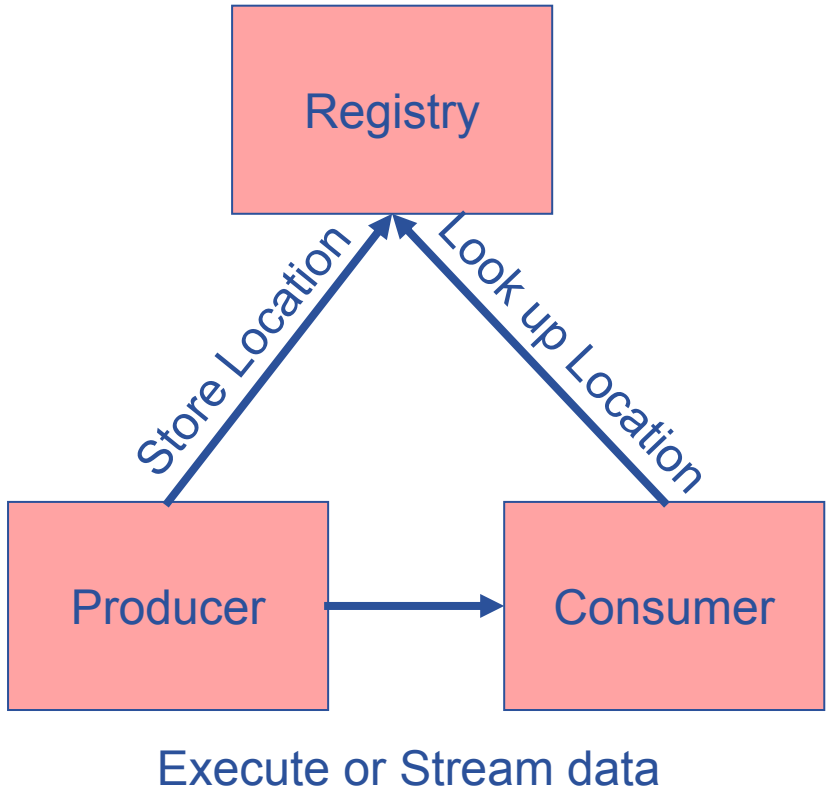
Introduction to R-GMA

- **Relational Grid Monitoring Architecture (R-GMA)**
  - Developed as part of the EuropeanDataGrid Project (EDG)
  - Now as part of the EGEE project.
  - Based the Grid Monitoring Architecture (GMA) from the Global Grid Forum (GGF).

- **Uses a relational data model.**
  - Data is viewed as a table.
  - Data structure defined by the columns.
  - Each entry is a row (tuple).
  - Queried using Structured Query Language (SQL).

GMA Architecture and Relational Model

- The Producer stores its location (URL) in the Registry.
- The Consumer looks up producer URLs in the Registry.
- The Consumer contacts the Producer to get all the data.
- Or the Consumer can listen to the Producer for new data.

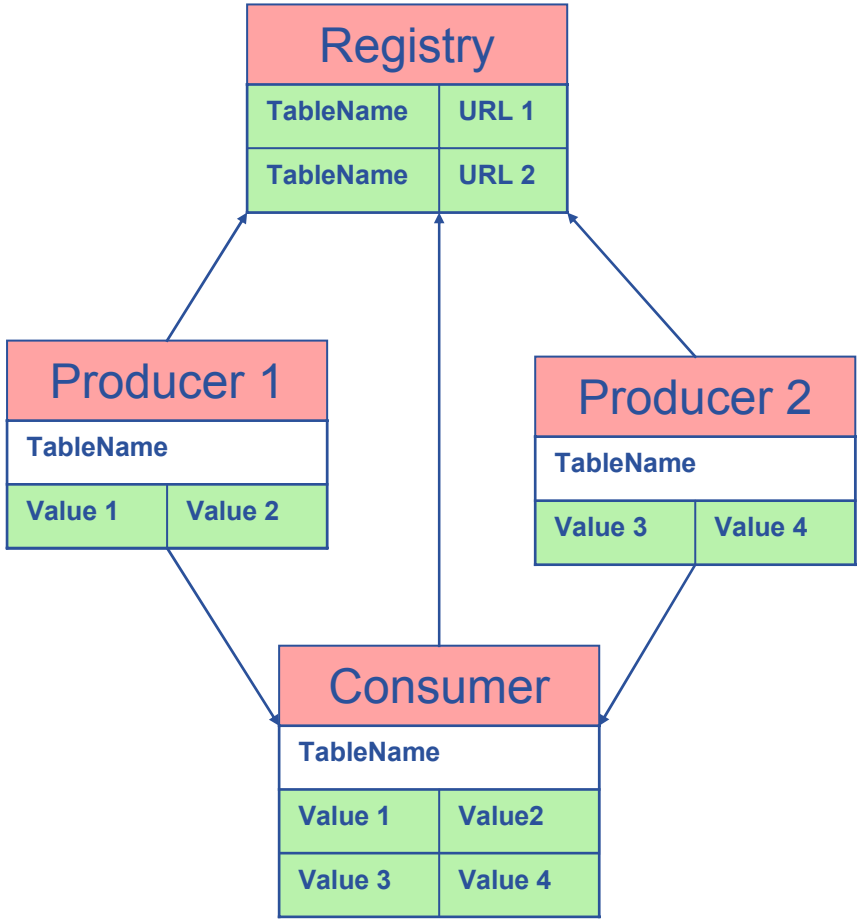


name	ID	birth	Group
Tom	4	1977-08-20	HR

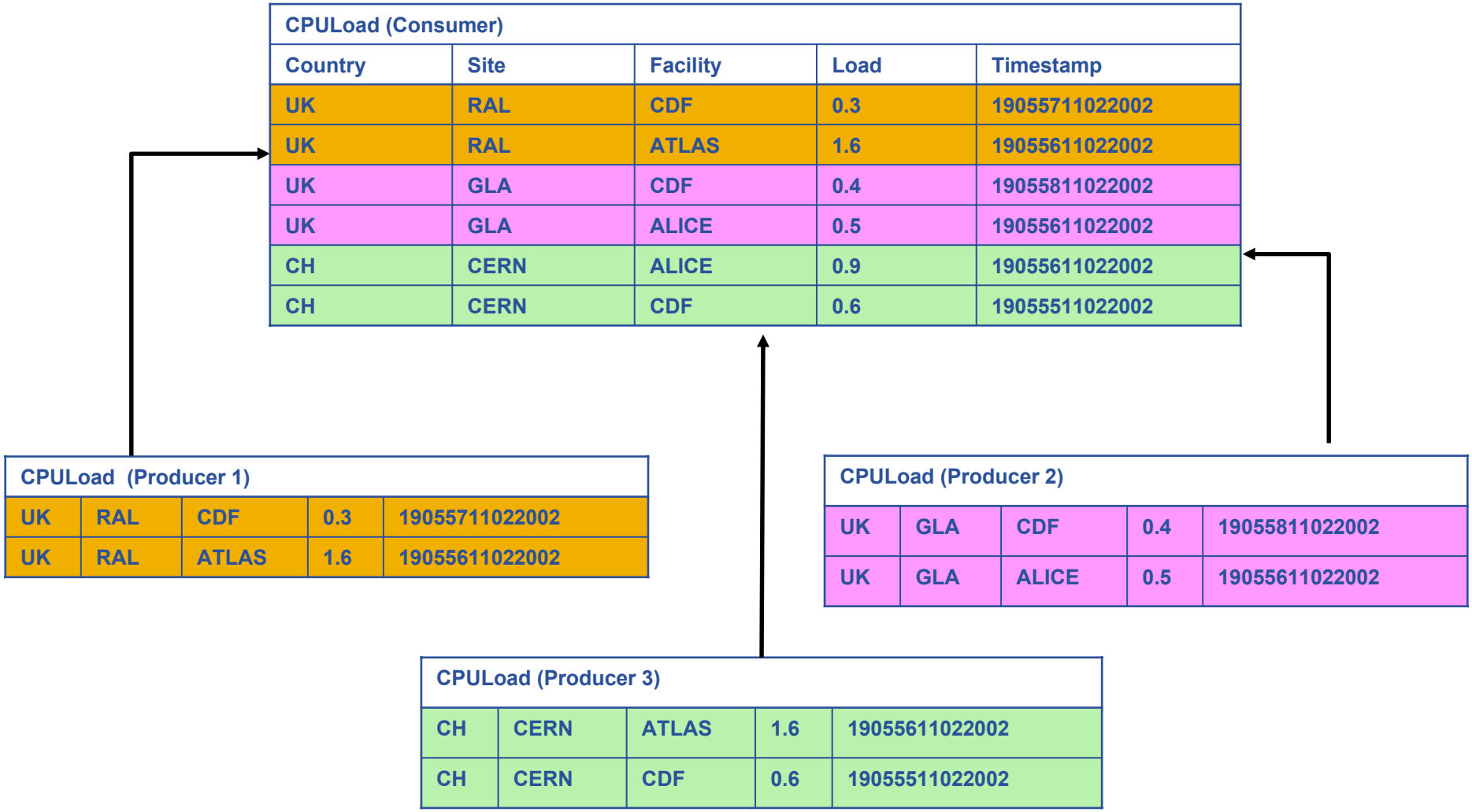
SELECT * FROM people WHERE group='HR'

Multiple Producers

- The Consumer will get all the URLs that could satisfy the query.
- The Consumer will connect to all the Producers.
- Producers that can satisfy the query will send the tuples to the Consumer.
- The Consumer will merge these tuples to form one result set.



Select * from CPULoad



CPULoad (Consumer)

Country	Site	Facility	Load	Timestamp
UK	RAL	CDF	0.3	19055711022002
UK	RAL	ATLAS	1.6	19055611022002
UK	GLA	CDF	0.4	19055811022002
UK	GLA	ALICE	0.5	19055611022002
CH	CERN	ALICE	0.9	19055611022002
CH	CERN	CDF	0.6	19055511022002

CPULoad (Producer 1)

UK	RAL	CDF	0.3	19055711022002
UK	RAL	ATLAS	1.6	19055611022002

CPULoad (Producer 2)

UK	GLA	CDF	0.4	19055811022002
UK	GLA	ALICE	0.5	19055611022002

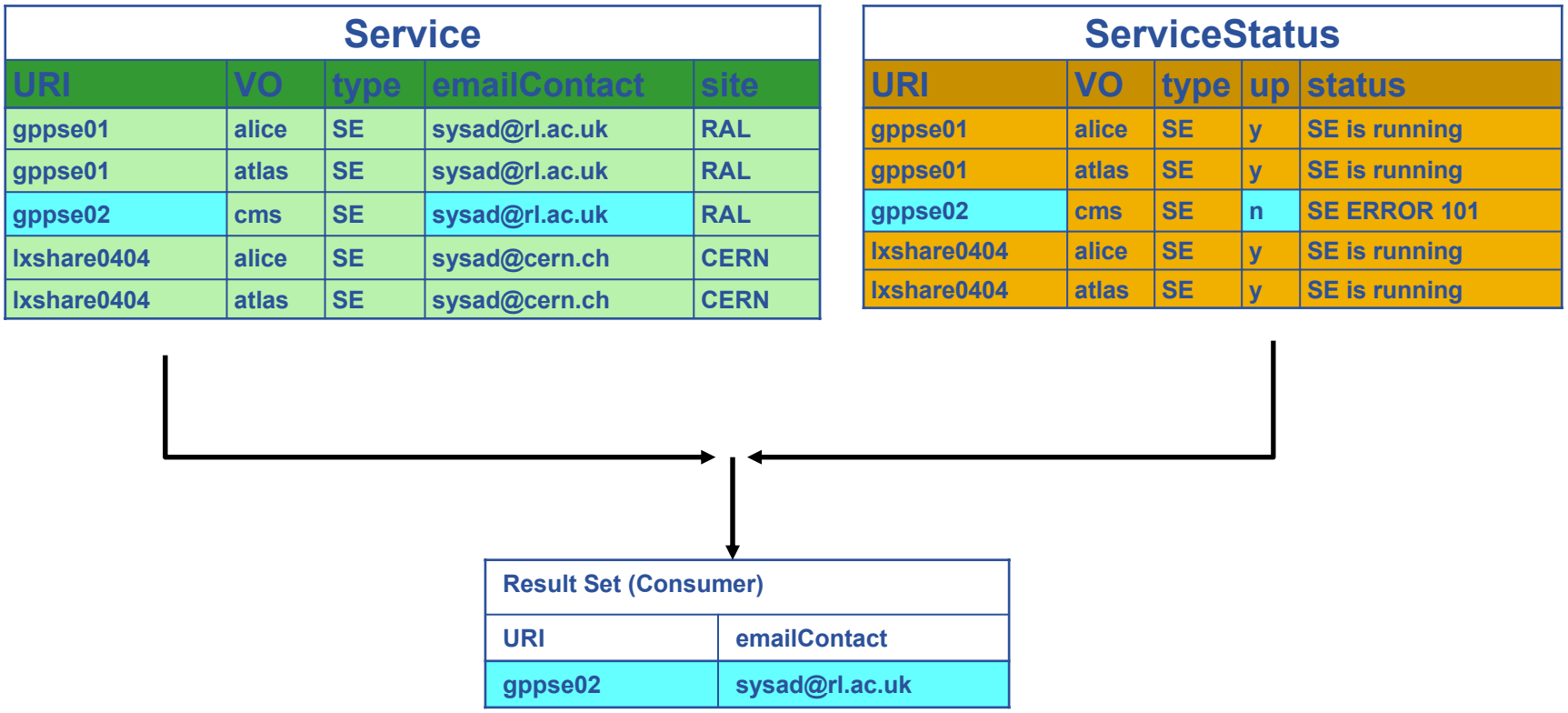
CPULoad (Producer 3)

CH	CERN	ATLAS	1.6	19055611022002
CH	CERN	CDF	0.6	19055511022002

The Mediator

- **The Mediator is the intelligence of R-GMA**
  - Not a single component, but distributed.
  - Enables queries to be accurately and efficiently returned.
- **The table name is stored next to the URL in the Registry.**
  - For simple queries, only the URLs that can answer query are passed to the Consumer.
  - If the query has a predicate, only the URLs that could satisfy the query will be passed to the Consumer.
- **The Mediator will also try to do joins.**
  - For complex queries the query must use a Producer with a database backend (secondary producer).
  - Merges and produces the resulting result set.
- **The Consumers URL and query is also stored in the Registry.**
  - Enables the Registry to notify listening Consumers about new Producers.

Joins



Service

URI	VO	type	emailContact	site
gppse01	alice	SE	sysad@rl.ac.uk	RAL
gppse01	atlas	SE	sysad@rl.ac.uk	RAL
gppse02	cms	SE	sysad@rl.ac.uk	RAL
lxshare0404	alice	SE	sysad@cern.ch	CERN
lxshare0404	atlas	SE	sysad@cern.ch	CERN

ServiceStatus

URI	VO	type	up	status
gppse01	alice	SE	y	SE is running
gppse01	atlas	SE	y	SE is running
gppse02	cms	SE	n	SE ERROR 101
lxshare0404	alice	SE	y	SE is running
lxshare0404	atlas	SE	y	SE is running

Result Set (Consumer)

URI	emailContact
gppse02	sysad@rl.ac.uk

SELECT Service.URI Service.emailContact FROM Service S, ServiceStatus SS
WHERE (S.URI= SS.URI and SS.up='n')

Others…

- **Security is available in R-GMA**
  - Uses https instead of http.
  - Authentication via Grid Certificates.
  - Authorization will be coming soon.
  - … But not currently used in LCG!

- **Soft registration:**
  - For producer and consumer servlets
    - They will close after the termination interval
    - The client needs periodically to show a sign of life
  - For entries in the registry
    - Producers must contact periodically (automatically done by R-GMA)

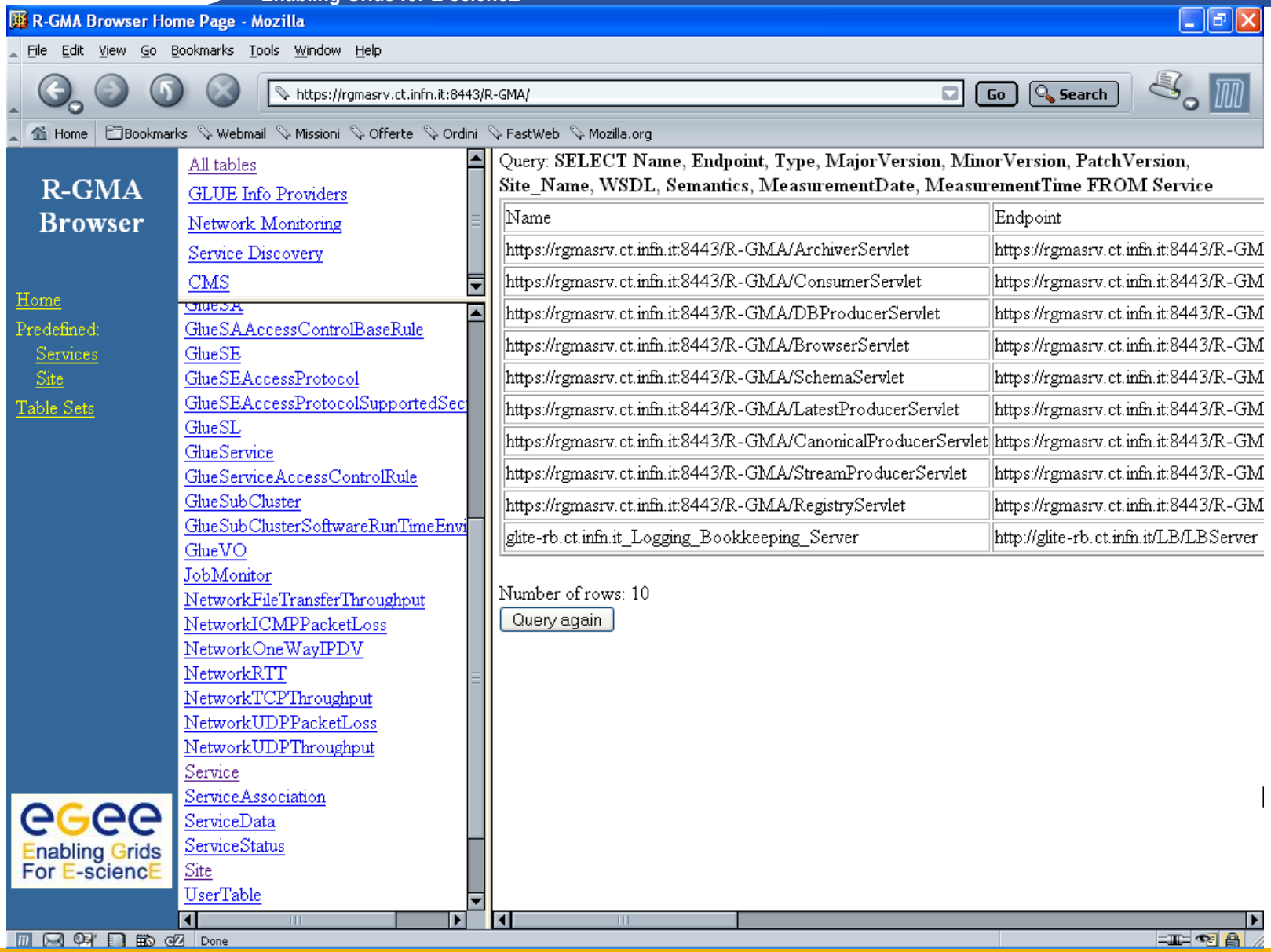
The R-GMA Browser

- **The easiest way to try out R-GMA.**
  - It is installed on the machine running the Registry and Schema:

    https://rgmasrv.ct.infn.it:8443/R-GMA
  - You can also install it along with the Producer and Consumer Servlets.

- **Using the Browser you can do the following.**
  - Browse the tables in the schema.
  - Look at the table definitions.
  - See all the available producers for a table.
  - Query a table.
  - Query only selected producers.

The R-GMA Browser (II)



R-GMA APIs

- **APIs exist in Java, C, C++, Python.**
  - For clients (servlets contacted behind the scenes)
- **They include methods for…**
  - Creating consumers
  - Creating primary and secondary producers
  - Setting type of queries, type of produces, retention periods, time outs…
  - Retrieving tuples, inserting data
  - …
- **You can create your own Producer or Consumer.**

Links

- **R-GMA overview page.**
  - http://www.r-gma.org/
- **R-GMA in EGEE**
  - http://hepunx.rl.ac.uk/egee/jra1-uk/
- **R-GMA Documenation**
  - http://hepunx.rl.ac.uk/egee/jra1-uk/LCG/doc/

Workload Management System

Overview of gLite WMS

- Job Management Services
  - main services related to job management/execution are
    - computing element
      - *job management (job submission, job control, etc.), but it must also provide*
      - *provision of information about its characteristics and status*
    - workload management
      - *core component discussed in details*
    - accounting
      - *special case as it will eventually take into account*
        - computing, storage and network resources
    - job provenance
      - *keep track of the definition of submitted jobs, execution conditions and environment, and important points of the job life cycle for a long period*
        - debugging, post-mortem analysis, comparison of job execution
    - package manager
      - *automates the process of installing, upgrading, configuring, and removing software packages from a shared area on a grid site.*
        - extension of a traditional package management system to a Grid

Workload Management System

- 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.

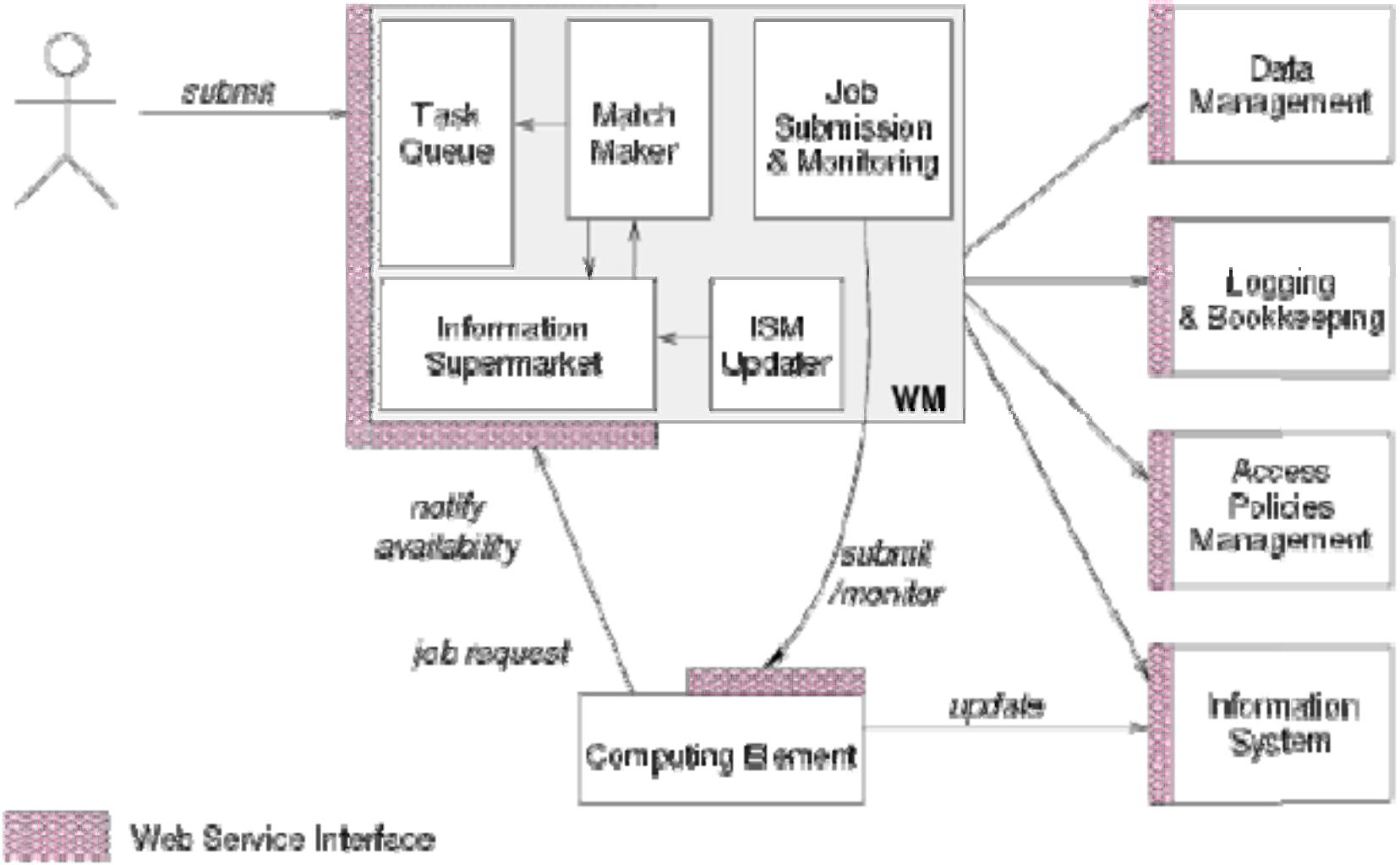
Workload Management System

- Purpose of Workload Manager (WM) is accept and satisfy requests for job management coming from its clients
  - **meaning of the submission request is to pass the responsibility of the job to the WM.**
    - **WM will pass the job to an appropriate CE for execution**
      - ***taking into account requirements and the preferences expressed in the job description***
- The decision of which resource should be used is the outcome of a *matchmaking* process between submission requests and available resources
  - **availability of resources for a particular task depends**
    - **on the state of the resources**
    - **on the utilisation policies**
      - ***assigned for the VO the user belogs***

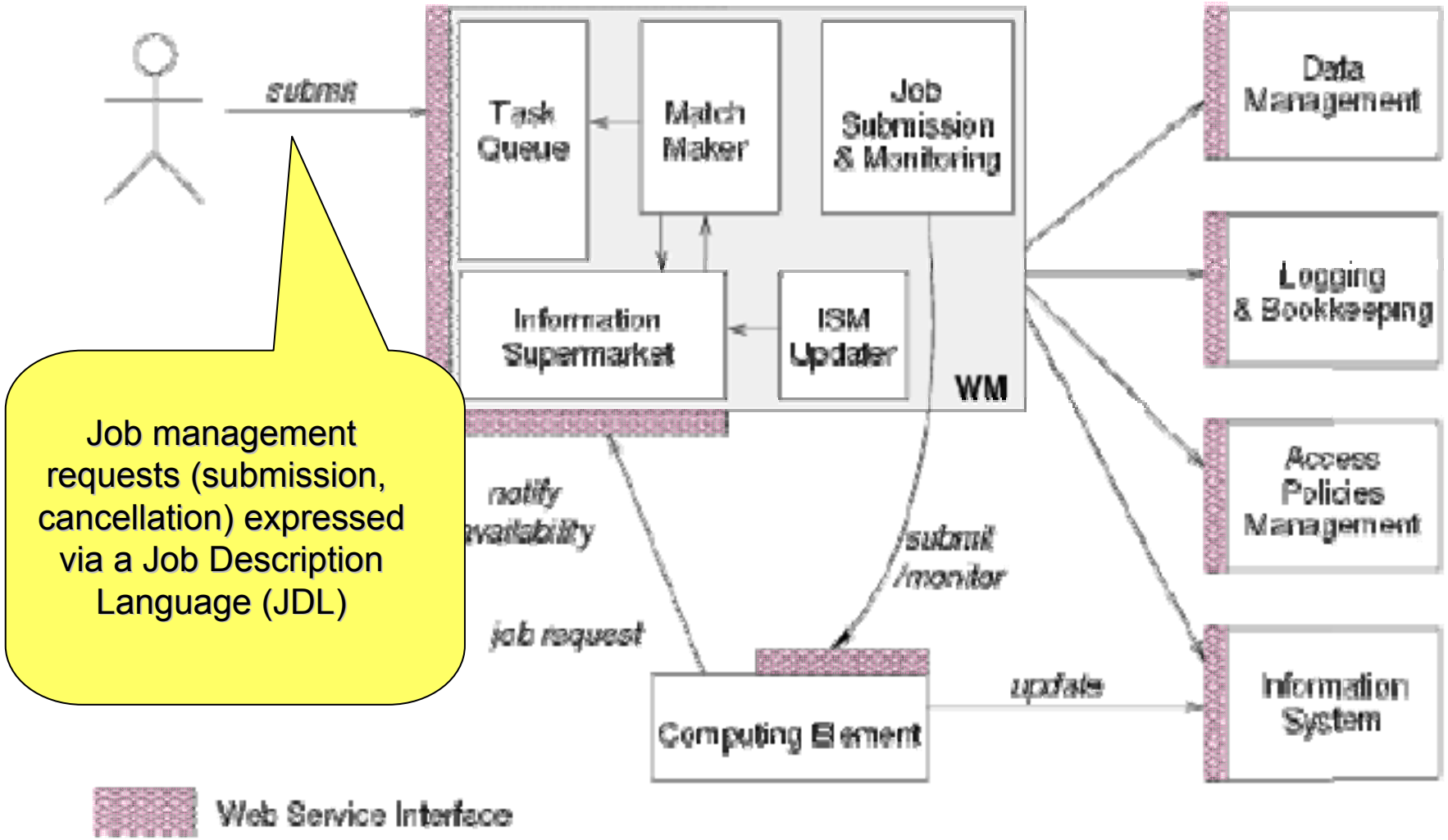
WMS's Scheduling Policies

- WM can adopt
  - **eager scheduling ("push" model)**
    - **a job is bound to a resource as soon as possible and, once the decision has been taken, the job is passed to the selected resource for execution**
  - **lazy scheduling ("pull" model)**
    - **foresees that the job is held by the WM until a resource becomes available, at which point that resource is matched against the submitted jobs**
      - ***the job that fits best is passed to the resource for immediate execution.***
- Varying degrees of eagerness (or laziness) are applicable
  - **match-making level**
    - **eager scheduling**
      - ***implies matching a job against multiple resources***
    - **lazy scheduling**
      - ***implies matching a resource against multiple jobs***

WMS's Architecture

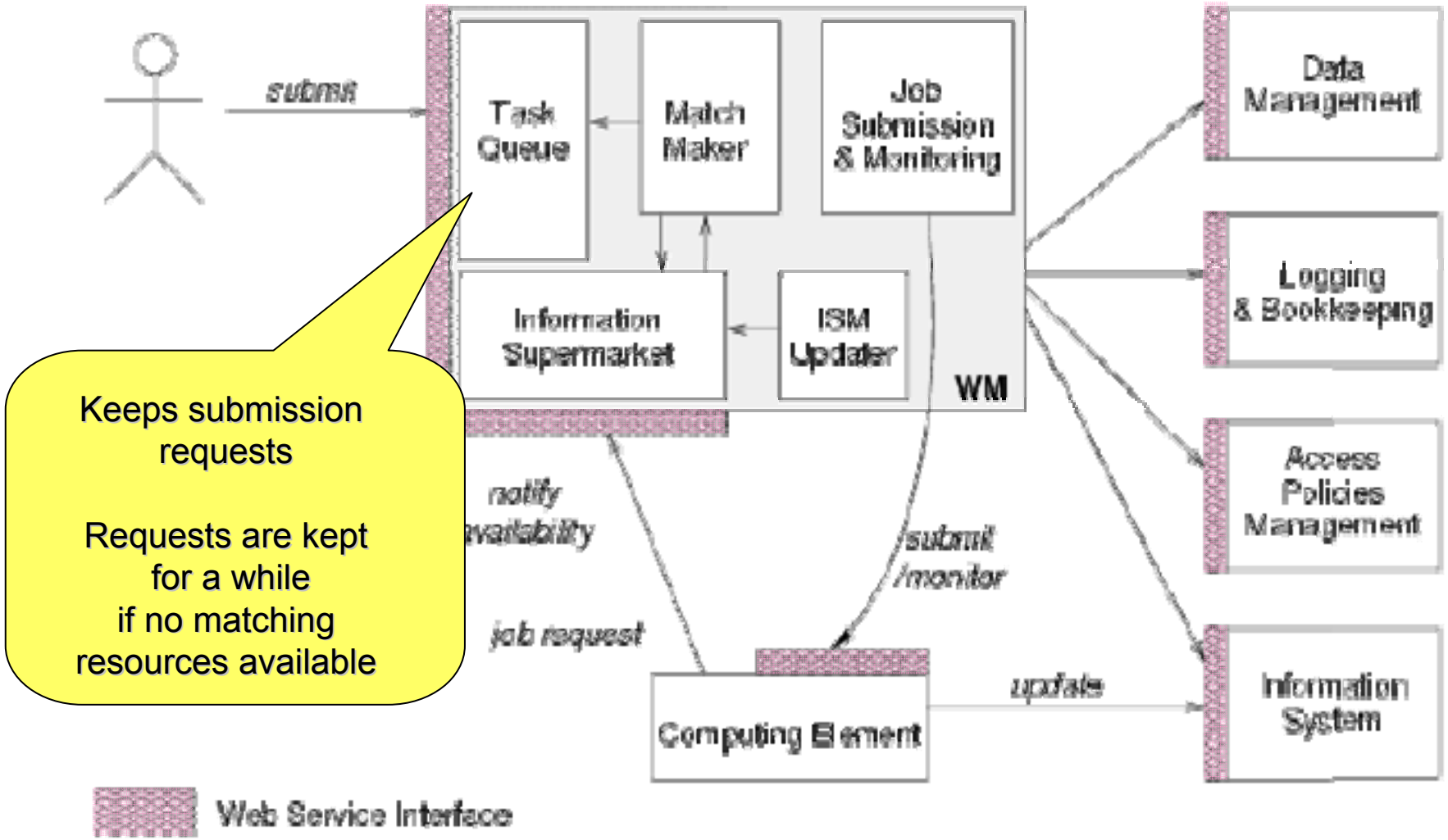


WMS's Architecture



Job management requests (submission, cancellation) expressed via a Job Description Language (JDL)

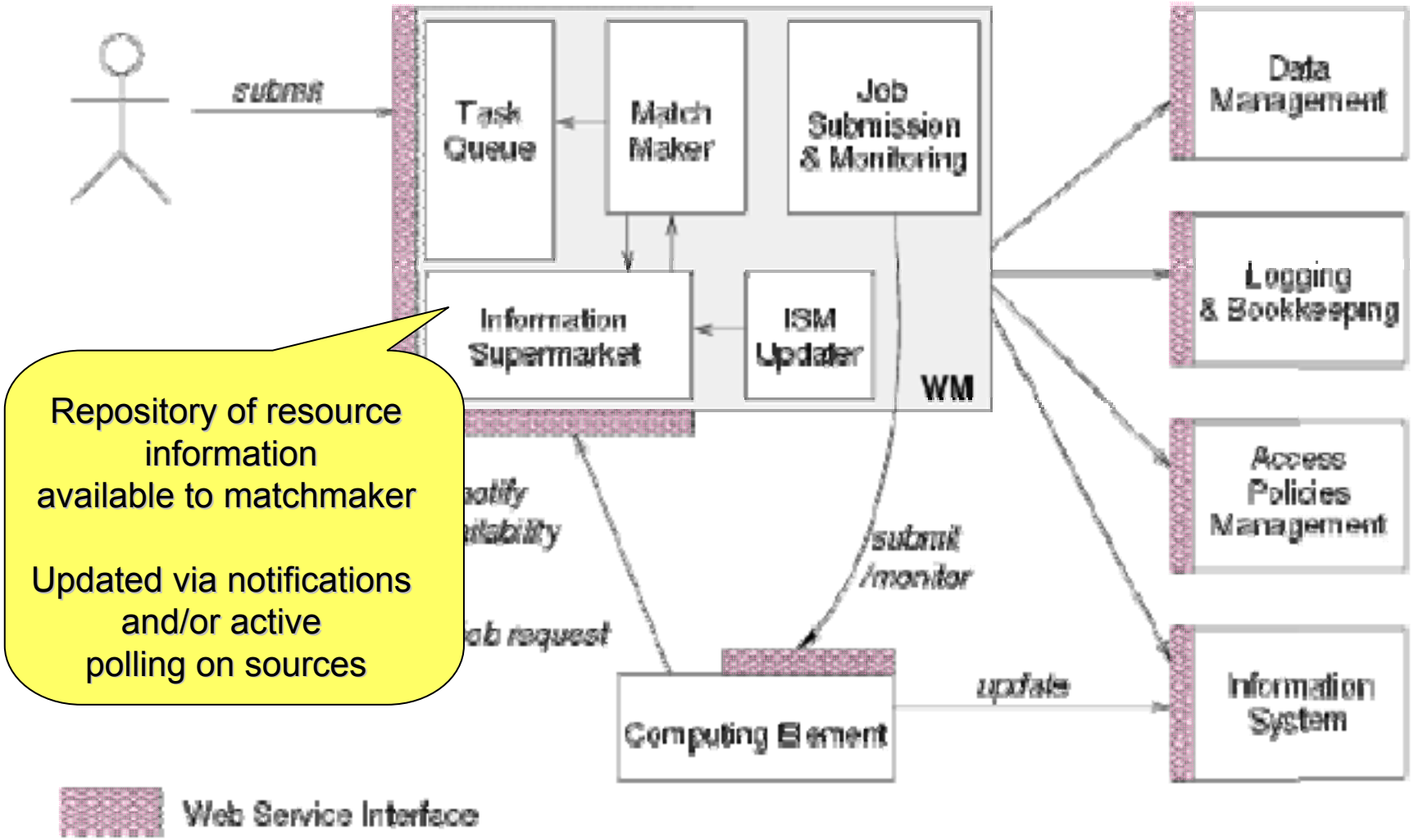
WMS's Architecture



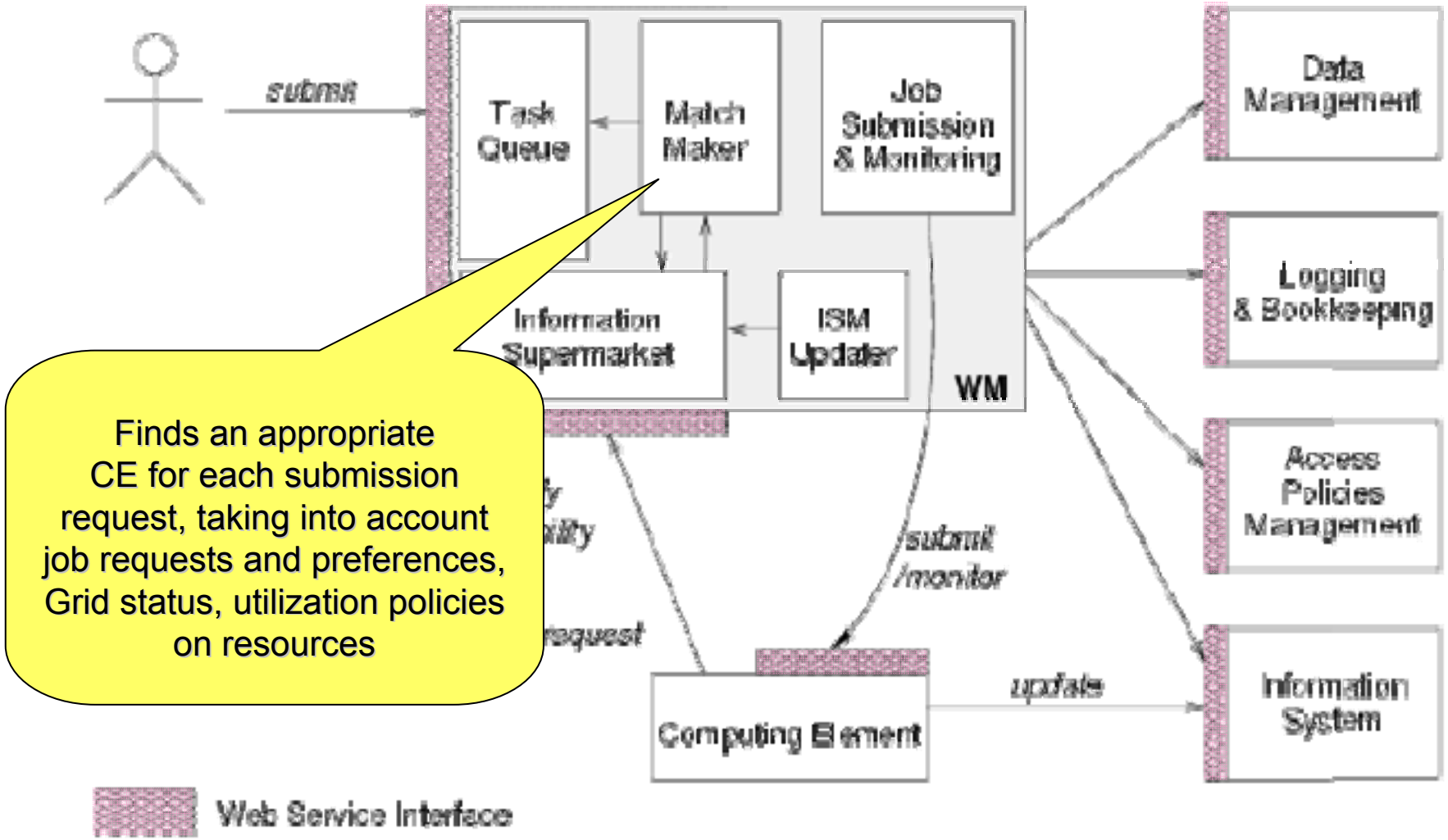
Keeps submission requests

Requests are kept for a while if no matching resources available

WMS's Architecture

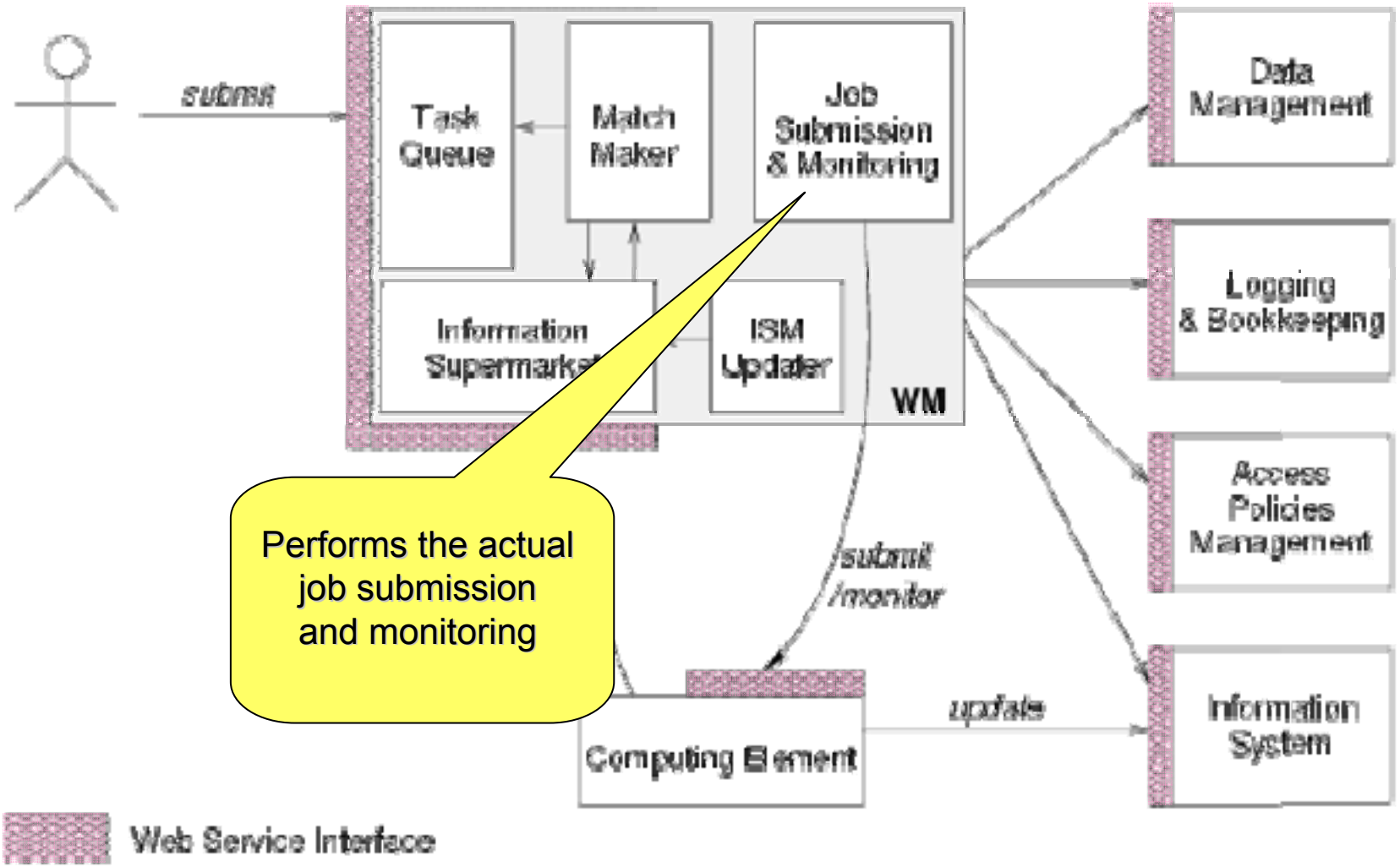


WMS's Architecture



Finds an appropriate CE for each submission request, taking into account job requests and preferences, Grid status, utilization policies on resources

WMS's Architecture



The Information Supermarket

- ISM represents one of the most notable improvements in the WM as inherited from the EU DataGrid (EDG) project
  - **decoupling between the collection of information concerning resources and its use**
    - **allows flexible application of different policies**
- The ISM basically consists of a repository of resource information that is available in read only mode to the matchmaking engine
  - **the update is the result of**
    - **the arrival of notifications**
    - **active polling of resources**
    - **some arbitrary combination of both**
  - **can be configured so that certain notifications can trigger the matchmaking engine**
    - **improve the modularity of the software**
    - **support the implementation of lazy scheduling policies**

The Task Queue

- The Task Queue represents the second most notable improvement in the WM internal design
  - **possibility to keep a submission request for a while if no resources are immediately available that match the job requirements**
    - **technique used by the AliEn and Condor systems**
- Non-matching requests
  - **will be retried either periodically**
    - **eager scheduling approach**
  - **or as soon as notifications of available resources appear in the ISM**
    - **lazy scheduling approach**

Job Logging & Bookkeeping

- L&B tracks jobs in terms of *events*
  - **important points of job life**
    - **submission, finding a matching CE, starting execution etc**
      - ***gathered from various WMS components***
- The events are passed to a physically close component of the L&B infrastructure
  - ***locallogger***
    - **avoid network problems**
      - ***stores them in a local disk file and takes over the responsibility to deliver them further***
- The destination of an event is one of *bookkeeping servers*
  - **assigned statically to a job upon its submission**
    - **processes the incoming events to give a higher level view on the job states**
      - **Submitted, Running, Done**
    - **various recorded attributes**
      - ***JDL, destination CE name, job exit code***
- Retrieval of both job states and raw events is available via legacy (EDG) and WS querying interfaces
  - **user may also register for receiving notifications on particular job state changes**

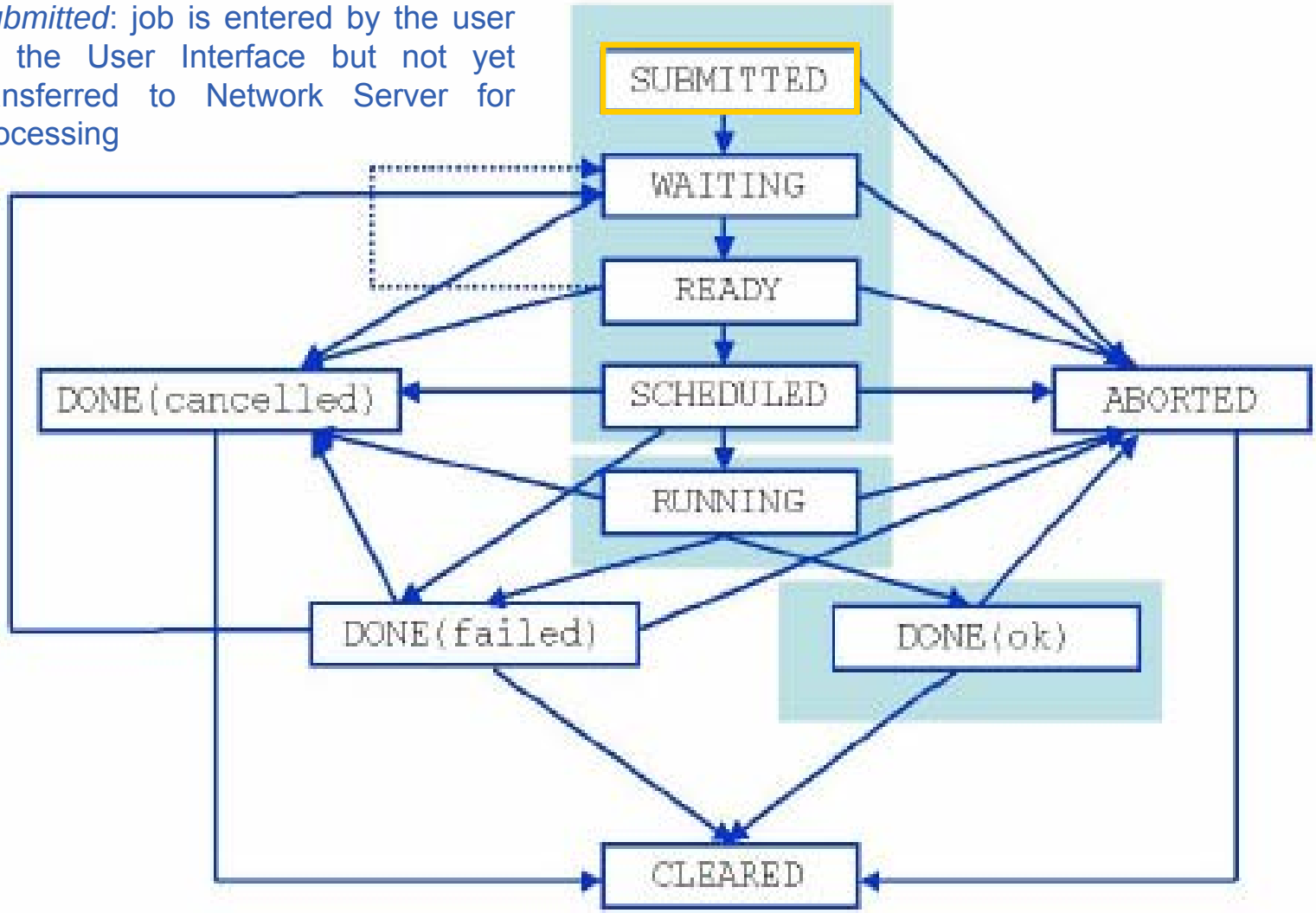
Job Submission Services

WMS components handling the job during its lifetime and performing the submission

- **Job Adapter**
  - **is responsible for**
    - **making the final touches to the JDL expression for a job, before it is passed to CondorC for the actual submission**
    - **creating the job wrapper script that creates the appropriate execution environment in the CE worker node**
      - ***transfer of the input and of the output sandboxes***
- **CondorC**
  - **responsible for**
    - **performing the actual job management operations**
      - ***job submission, job removal***
- **DAGMan**
  - **meta-scheduler**
    - **purpose is to navigate the graph**
    - **determine which nodes are free of dependencies**
    - **follow the execution of the corresponding jobs.**
  - **instance is spawned by CondorC for each handled DAG**
- **Log Monitor**
  - **is responsible for**
    - **watching the CondorC log file**
    - **intercepting interesting events concerning active jobs**
      - ***events affecting the job state machine***
    - **triggering appropriate actions.**

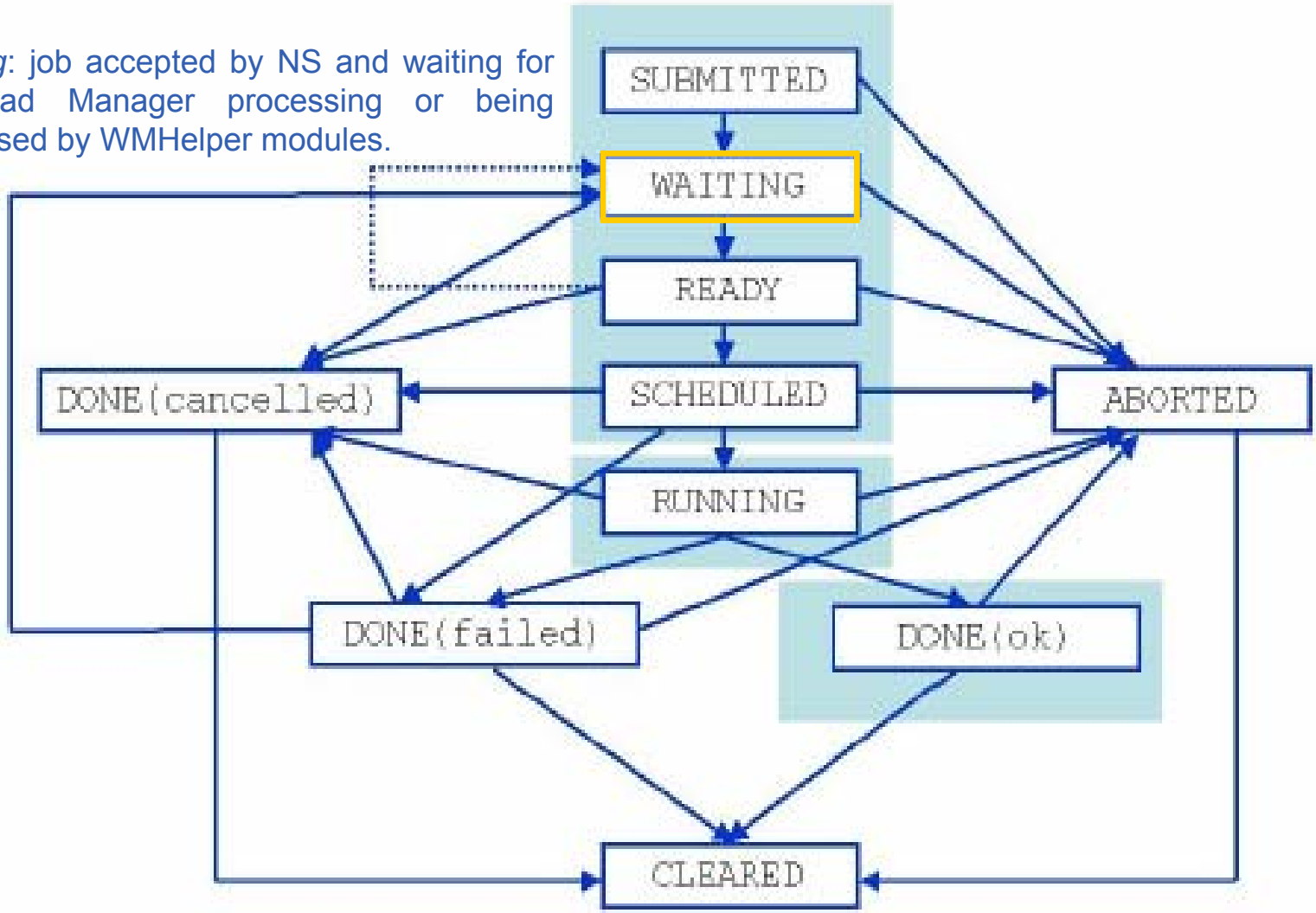
Jobs State Machine (1/9)

Submitted: job is entered by the user to the User Interface but not yet transferred to Network Server for processing

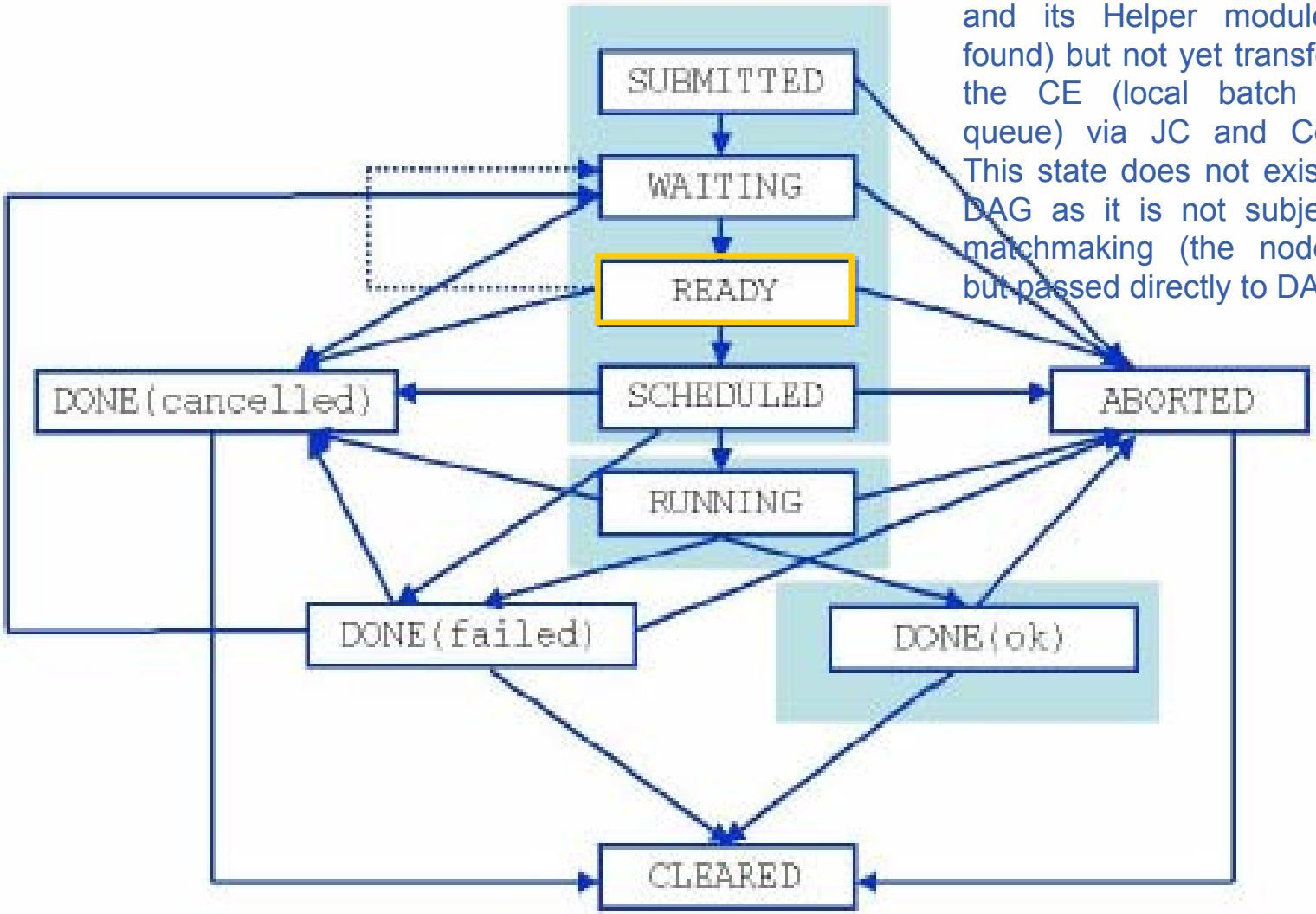


Jobs State Machine (2/9)

Waiting: job accepted by NS and waiting for Workload Manager processing or being processed by WMHelper modules.

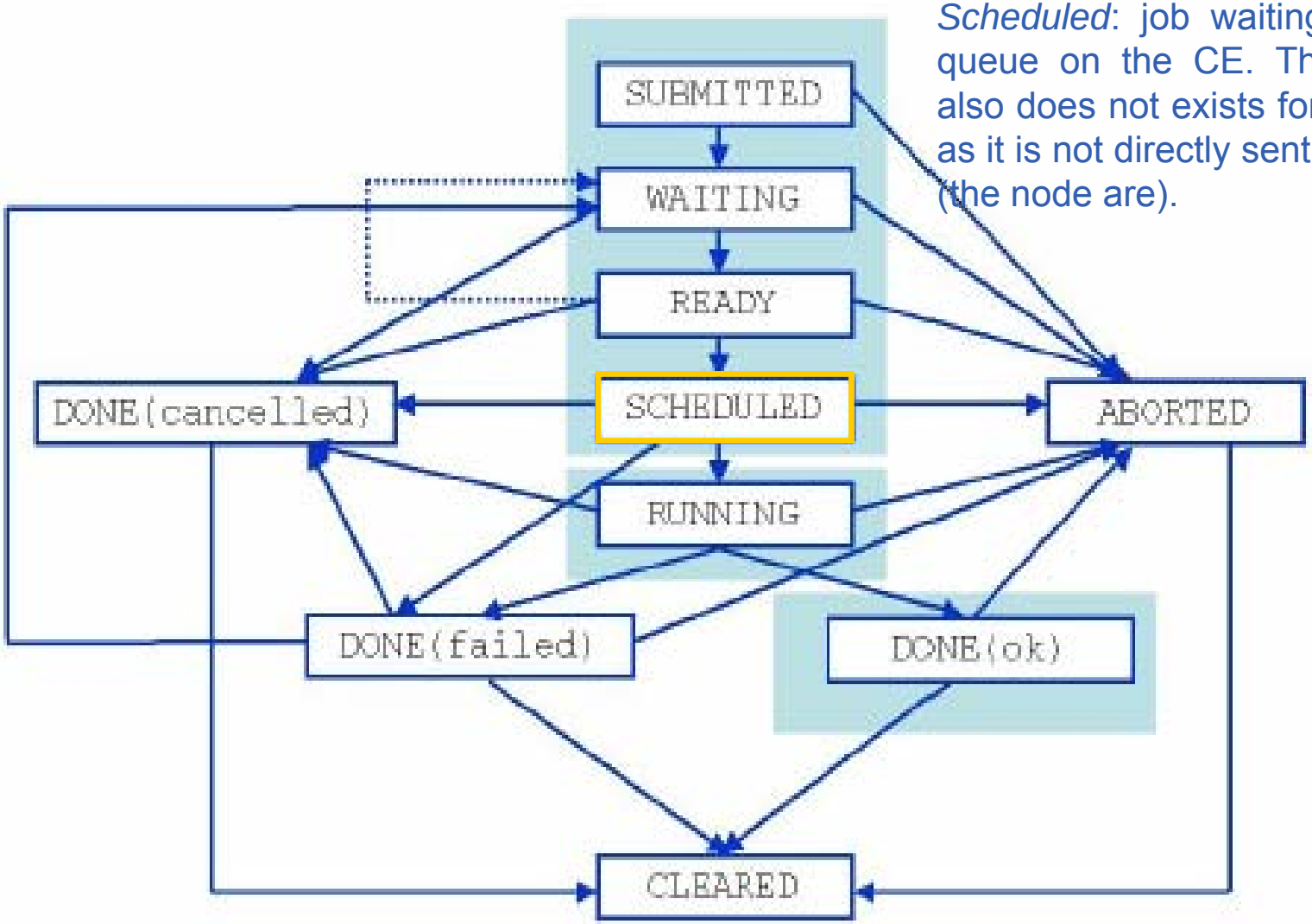


Jobs State Machine (3/9)



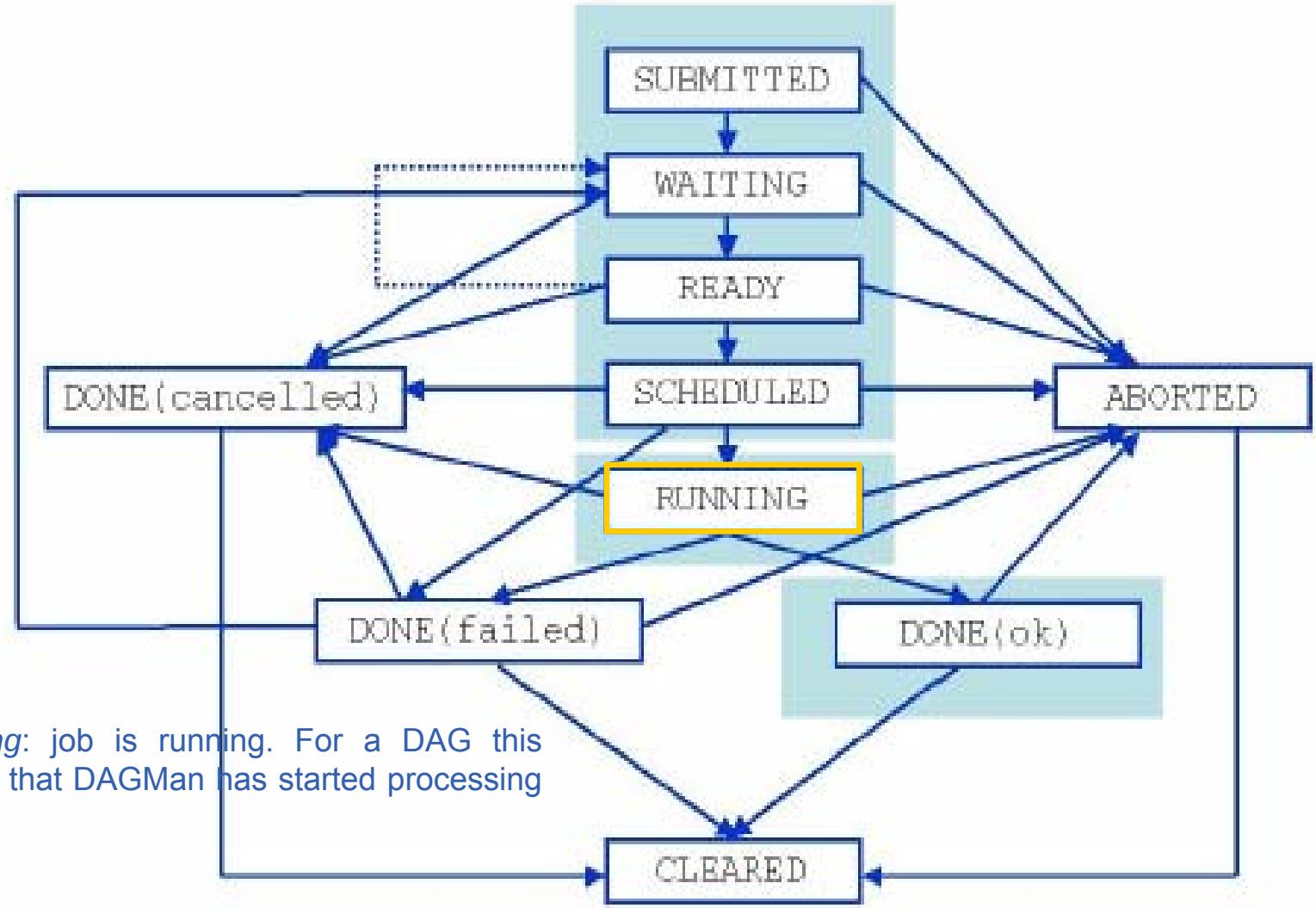
Ready: job processed by WM and its Helper modules (CE found) but not yet transferred to the CE (local batch system queue) via JC and CondorC. This state does not exists for a DAG as it is not subjected to matchmaking (the nodes are) but passed directly to DAGMan.

Jobs State Machine (4/9)



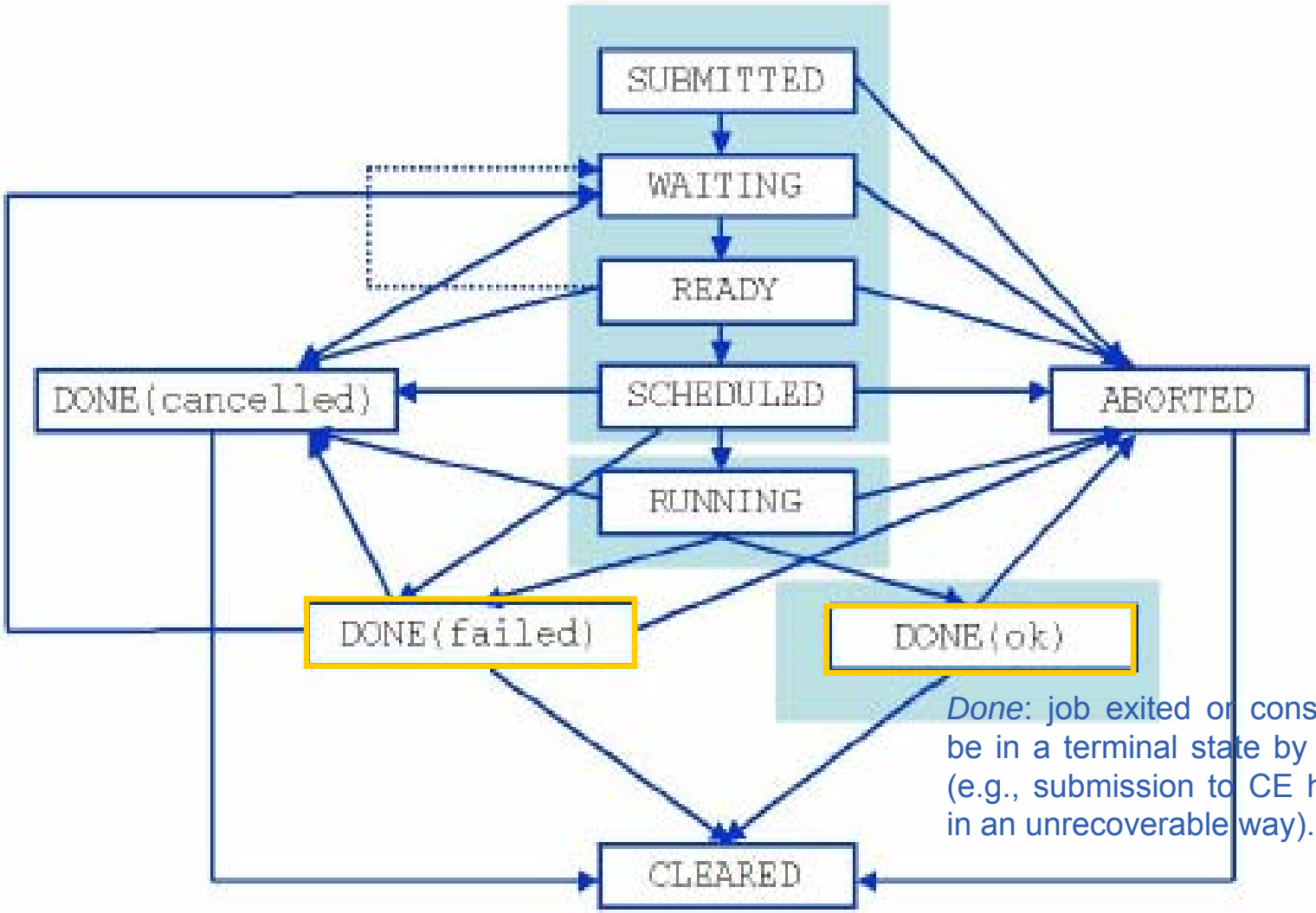
Scheduled: job waiting in the queue on the CE. This state also does not exists for a DAG as it is not directly sent to a CE (the node are).

Jobs State Machine (5/9)



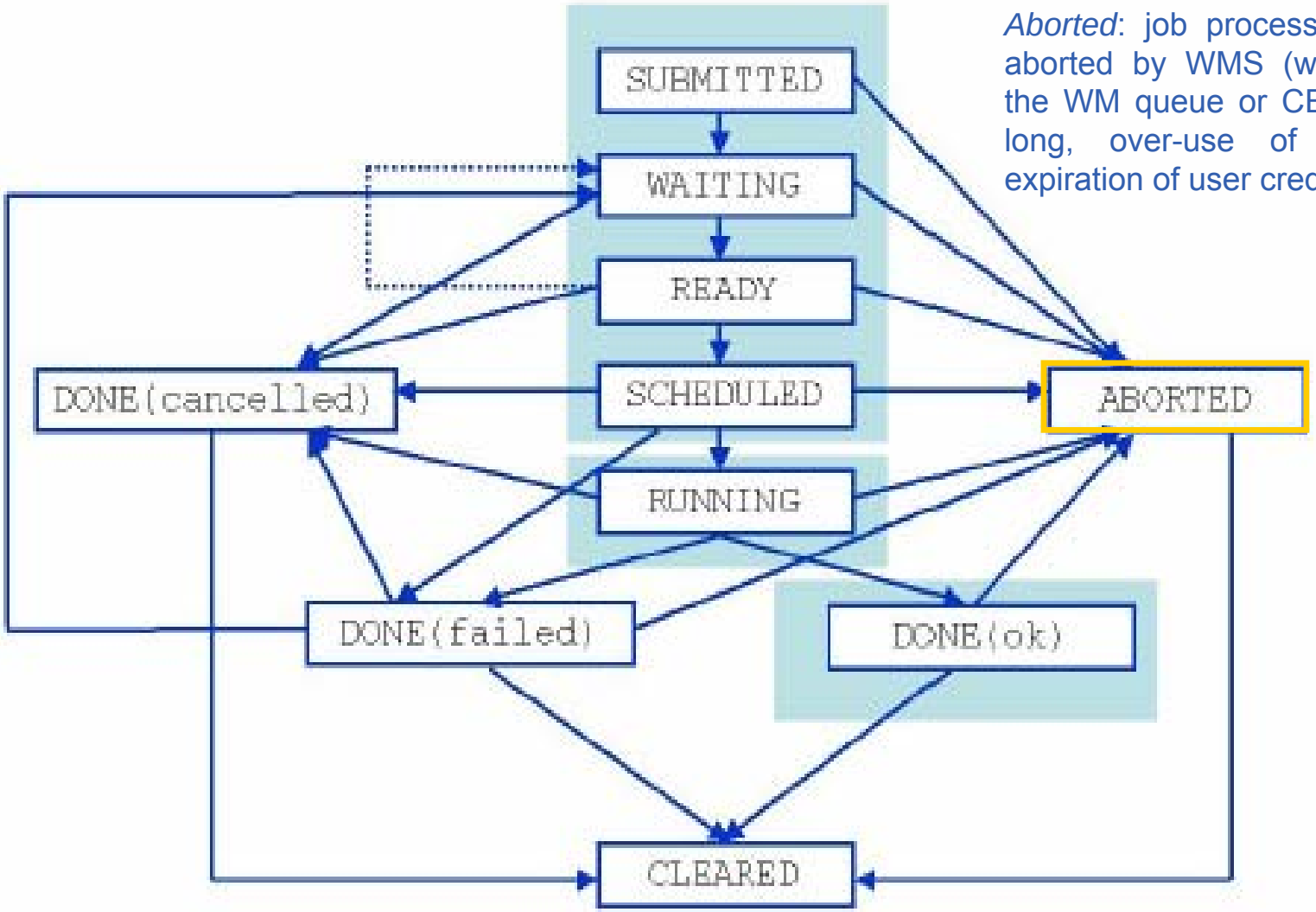
Running: job is running. For a DAG this means that DAGMan has started processing it.

Jobs State Machine (6/9)



Done: job exited or considered to be in a terminal state by CondorC (e.g., submission to CE has failed in an unrecoverable way).

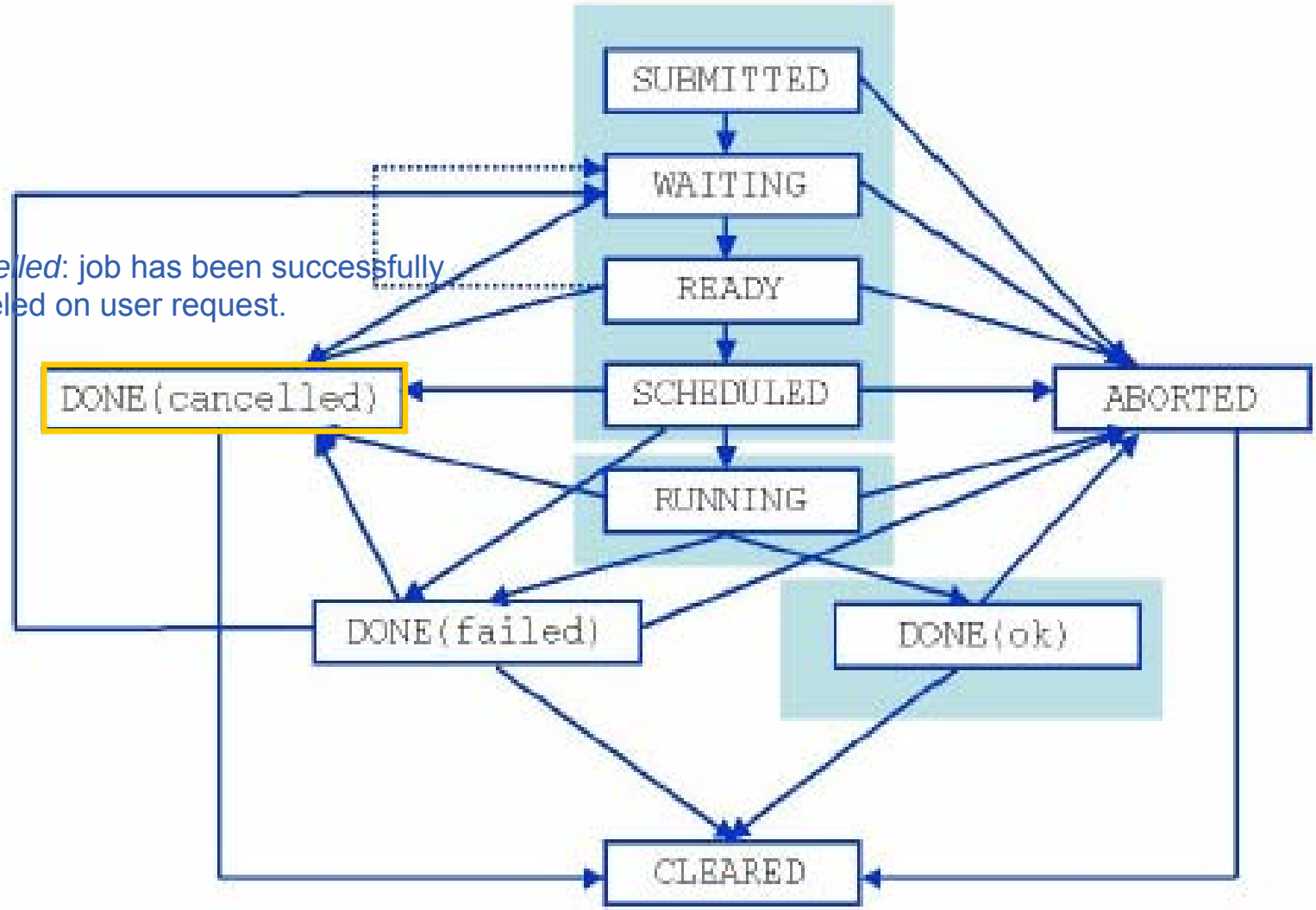
Jobs State Machine (7/9)



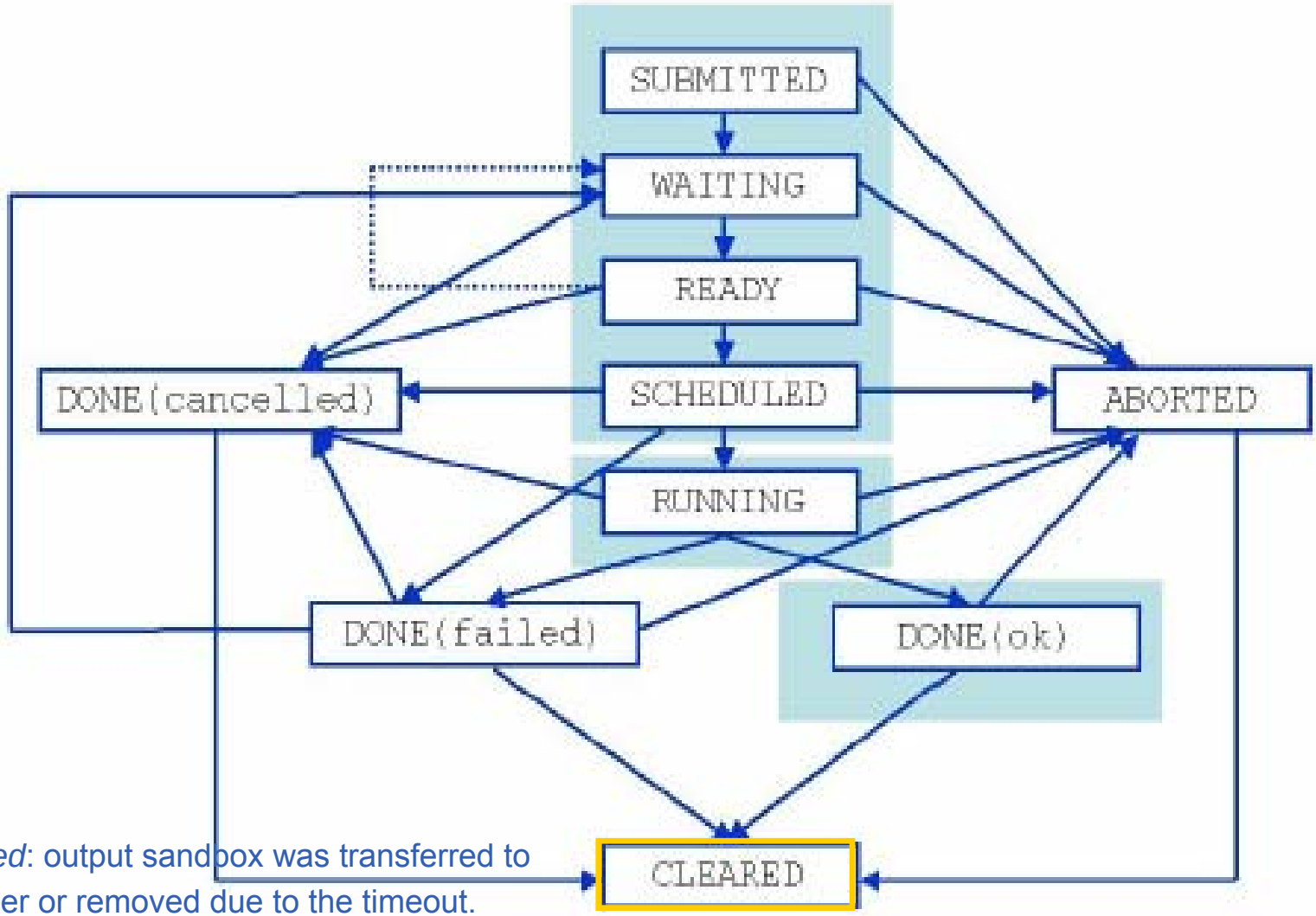
Aborted: job processing was aborted by WMS (waiting in the WM queue or CE for too long, over-use of quotas, expiration of user credentials).

Jobs State Machine (8/9)

Cancelled: job has been successfully canceled on user request.



Jobs State Machine (9/9)



Cleared: output sandbox was transferred to the user or removed due to the timeout.

Grid Accounting

A generic Grid accounting process accumulates info on Grid Usage by users/groups (VOs) and involves many subsequent phases as:

- **Metering:** Collection of usage metrics on computational resources.
- **Accounting:** Storage of such metrics for further analysis.
- **Usage Analysis:** Production of reports from the available records.
- **Pricing:** Assign and manage prices for computational resources.
- **Billing:** Assign a cost to user operations and charge them.
- **To be used:** To track resource usage | To discover abuses (and help avoiding them).

- **Allows implementation of submission policies based on resource usage**
  - Exchange market among Grid users and Grid resource owners, which should result in market equilibrium → Load balancing on the Grid

During the metering phase the user payload on a resource needs to be correctly measured, and unambiguously assigned to the Grid User that directly or indirectly requested it to the Grid → Load Dedicated Sensors for Grid Resources

These pieces of information, when organized, form the Usage Record **for the user process → *Grid Unique Identifier* (for User, Resource, Job) plus the metrics of the resource consumption.**

A distributed architecture **is essential, as well as reliable and** fault tolerant communication **mechanisms.**

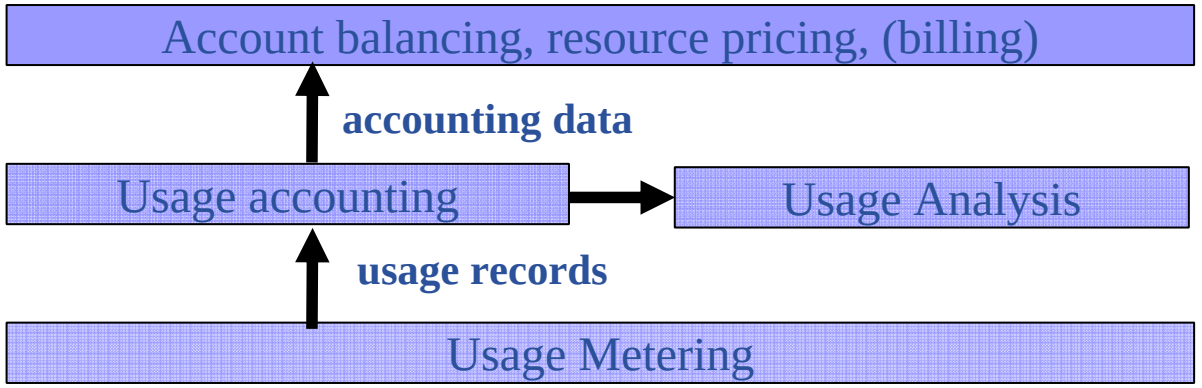
Different types of users are interested in different views of the usage records.

DGAS

The *Data Grid Accounting System* was originally developed within the EU Datagrid Project and is now being maintained and re-engineered within the EU EGEE Project.

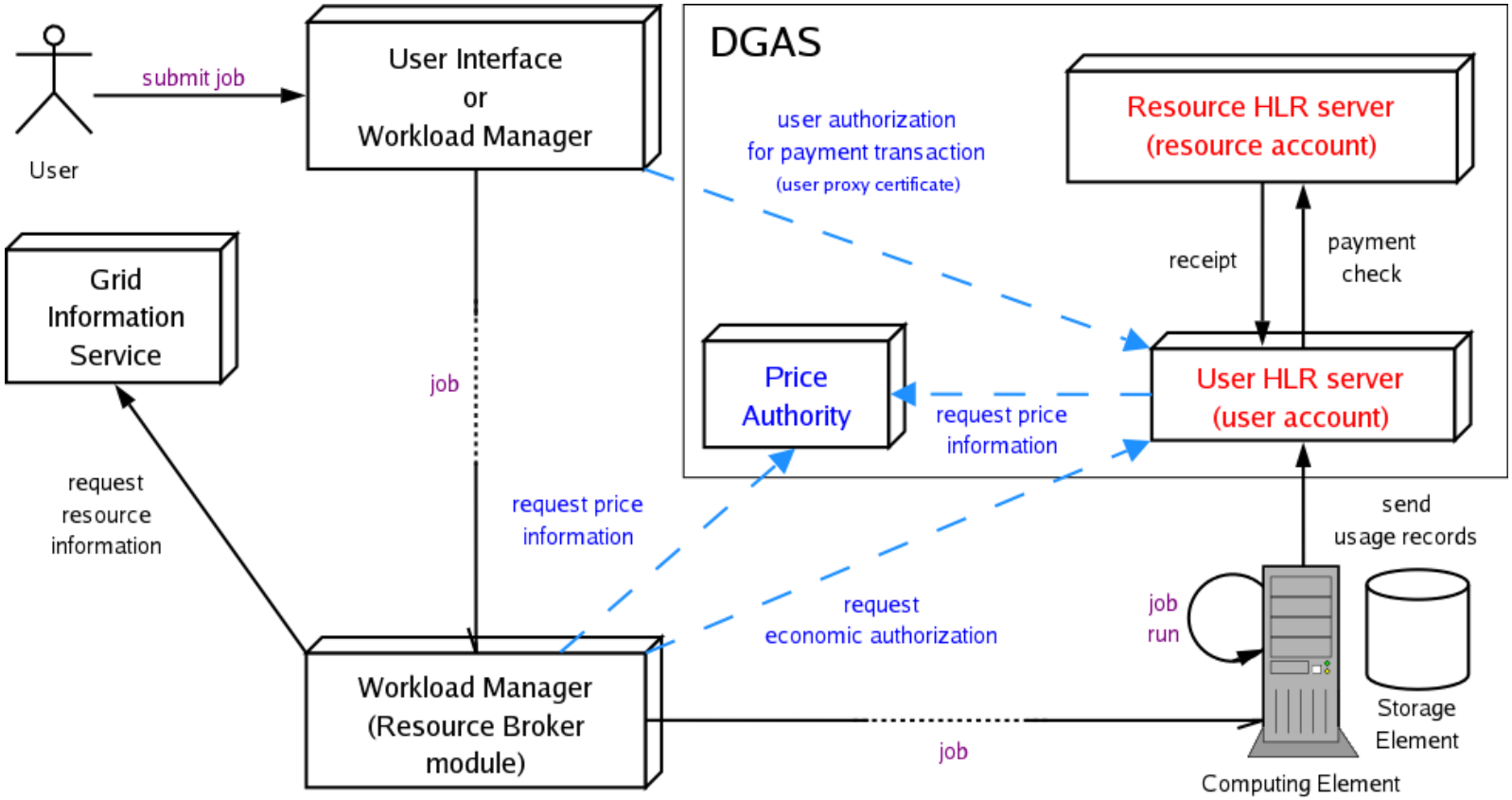
The Purpose of *DGAS* is to implement *Resource Usage Metering*, *Accounting* and *Account Balancing* (through *resource pricing*) in a fully distributed Grid environment. It is conceived to be distributed, secure and extensible**.**

The system is designed in order for Usage Metering, Accounting and Account Balancing (through resource pricing) to be indipendent layers.



DGAS Accounting Architecture

A simplified view of DGAS within the WMS context.



References

- **gLite WMS's User Guide**
  - https://edms.cern.ch/document/572489/1
- **EGEE Middleware Architecture DJRA1.1**
  - https://edms.cern.ch/document/476451/
- **Practical approaches to Grid workload management in the EGEE project – CHEP 2004**
  - https://edms.cern.ch/document/503558
- **Grid accounting in EGEE, current practices – Terena Network Conference 2005**
  - http://www.terena.nl/conferences/tnc2005/programme/presentations/show.php?pres_id=107

Data Management System

Introduction

- **User and programs produce and require data**

- **Data may be stored in Grid datasets (files)**
  - Located in Storage Elements (SEs)
  - Several replicas of one file in different sites
  - Accessible by Grid users and applications from “everywhere”
  - Locatable by the WMS (data requirements in JDL)

- **Also…**
  - Resource Broker can send (small amounts of) data to/from jobs: Input and Output Sandbox
  - Data may be copied from/to local filesystems (WNs, UIs) to the Grid

Data Management Tasks

- **File Management**
  - Storage
  - Access
  - Placement
  - Cataloguing
  - Security
- **Metadata Management**
  - Secure database access
  - Schema management
  - File-based metadata
  - Generic metadata

Data management: general concepts

- **What does "Data Management" mean ?**
  - Users and applications produce and require data
  - Data may be stored in Grid files
  - Granularity is at the "file" level (no data "structures")
  - Users and applications need to handle files on the Grid
- **Files are stored in appropriate permanent resources called "Storage Elements" (SE)**
  - Present almost at every site together with computing resources
  - We will treat a storage element as a "black box" where we can store data
    - Appropriate data management utilities/services hide internal structure of SE
    - Appropriate data management utilities/services hide details on transfer protocols

Data Management Services

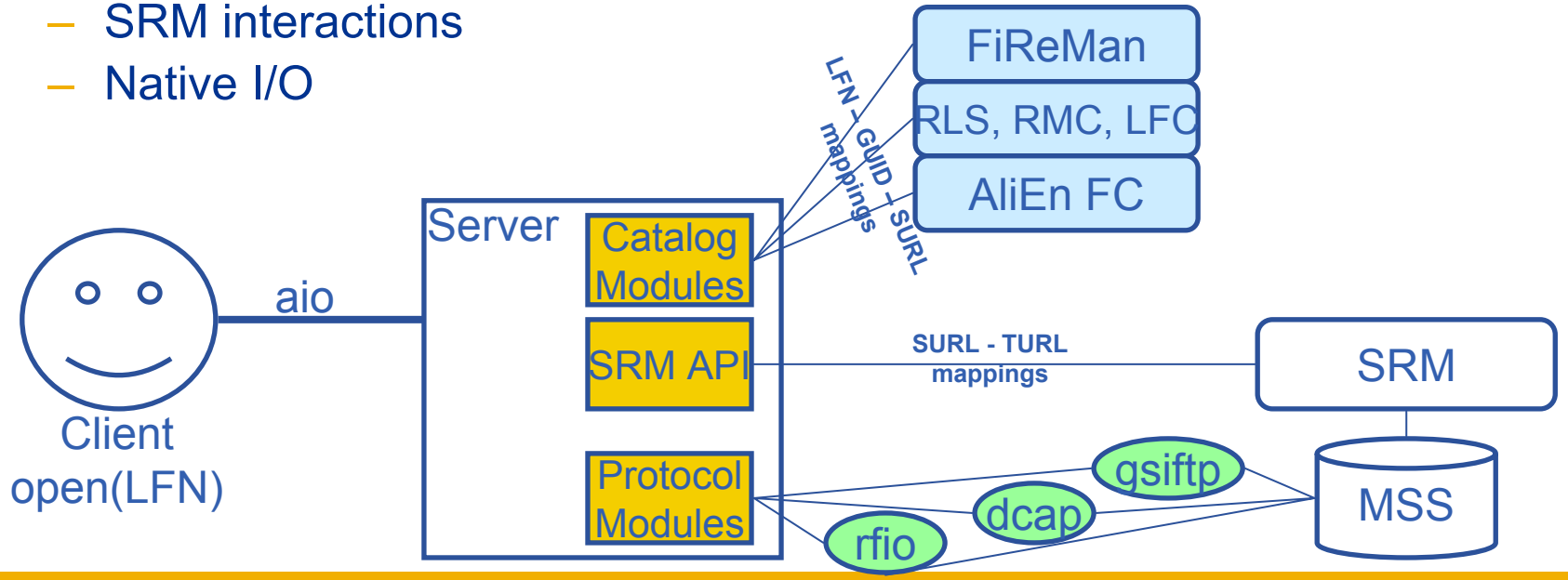
- **Storage Element**
  - Storage Resource Manager
  - POSIX-I/O
  - Access protocols gsiftp, https, rfio, file, …
- **Catalogs**
  - File Catalog
  - Replica Catalog
  - File Authorization Service
  - Metadata Catalog
- **File Transfer**
  - File Transfer Service
  - File Placement Service

Product Overview

- File Storage
  - **Storage Elements** with **SRM** (Storage Resource Manager) interface
  - Posix I/O interface through **glite-io**
  - Supports transfer protocols (bbftp, https, ftp, **gsiftp**, **rfio**, **dcap**, …)
- Catalogs
  - **File and Replica Catalog**
  - **File Authorization Service**
  - **Metadata Catalog**
  - Distribution of catalogs, conflicts resolution (**messaging**)
- Transfer
  - Top-level **Data Scheduler** as global entry point (there may be many).
  - Site **File Placement Service** managing transfers and catalog interactions
  - Site **File Transfer Service** managing incoming transfers (the network resource)

File Access Overview

- **Client only sees a simple API library and a Command Line Interface**
  - GUID or LFN can be used, i.e. open("/grid/myFile")
- **GSI Delegation to gLite I/O Server**
- **Server performs all operations on User's behalf**
  - Resolve LFN/GUID into SURL and TURL
- **Operations are pluggable**
  - Catalog interactions
  - SRM interactions
  - Native I/O



MSS and SRM

- gLite IO server relies against a Mass Storage System implementing SRM interface
- gLite IO server comunicates with MSS through SRM
- SRM is not provided by gLite !
- Tested MSS are, till now, CASTOR and dCache
- Full support to functionalities depending also from MSS
- Installing and configuring MSS is apart from gLite issues
- How to and guides to do so

http://egee-na4.ct.infn.it/wiki/out_pages/dCache-SRM.html
http://storage.esc.rl.ac.uk/documentation/html/D-Cache-Howto

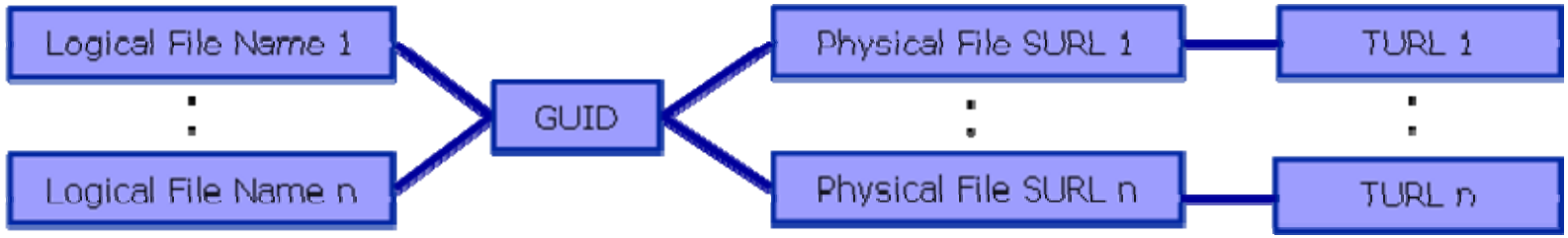
Data transfer and replication

- **Data movements capability (should be…) provided by**
  - Data scheduler (DS) (top-level)
  - File Placements Services (FPS) (local)
  - Transfer Agent (FTA) (local)
  - File Transfer Library (low lewel, called by applications)
- **DS keeps track of data movement request submitted by clients**
- **FPS pools DS fetching transfers with local site as destination, updating catalog**
- **FTA mantains state of transfers and manages FTA**
- **Data scheduler has not been released with gLite 1.1**
  - So actually no replica can be performed with gLite DMS

LCG File Catalog (LFC)

Name conventions

- **Logical File Name (LFN)**
  - An alias created by a user to refer to some item of data, e.g. "lfn:cms/20030203/run2/track1"
- **Globally Unique Identifier (GUID)**
  - A non-human-readable unique identifier for an item of data, e.g. "guid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6"
- **Site URL (SURL) (or Physical File Name (PFN) or Site FN)**
  - The location of an actual piece of data on a storage system, e.g.
    "srm://pcrd24.cern.ch/flatfiles/cms/output10_1" (SRM)
    "sfn://lxshare0209.cern.ch/data/alice/ntuples.dat" (Classic SE)
- **Transport URL (TURL)**
  - Temporary locator of a replica + access protocol: understood by a SE, e.g.
    "rfio://lxshare0209.cern.ch//data/alice/ntuples.dat"



File Catalogs in LCG

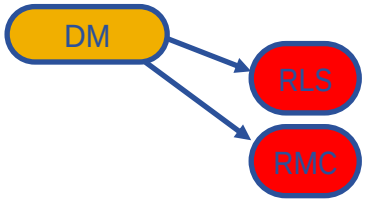
- **File catalogs in LCG:**
  - They keep track of the location of copies (replicas) of Grid files
  - The DM tools and APIs and the WMS interact with them
- **EDG's Replica Location Service (RLS, "old!")**
  - Catalogs in use in LCG-2
  - Replica Metadata Catalog (**RMC**) + Local Replica Catalog (**LRC**)
  - Some performance problems detected during Data Challenges
- **New LCG File Catalog (LFC, "current!")**
  - In production in next LCG release; deployment in January 2005
  - Coexistence with RLS; migration tools provided:

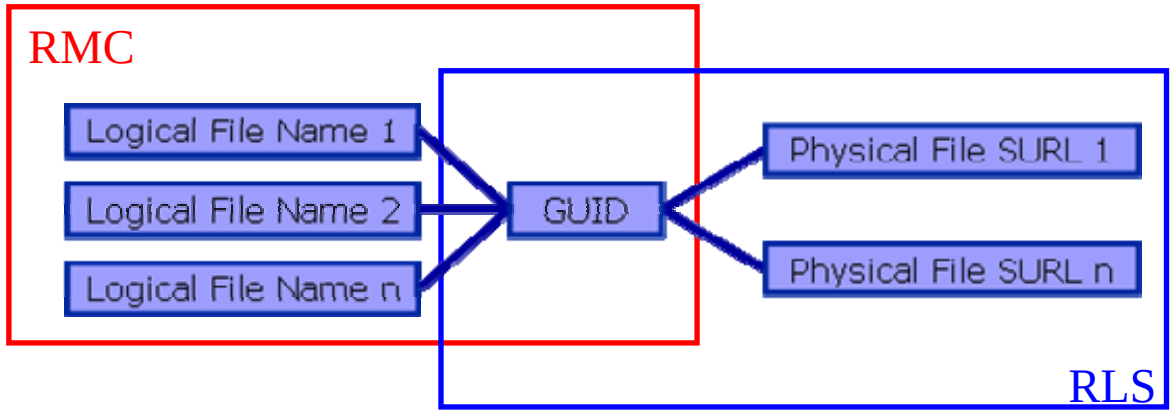
http://goc.grid.sinica.edu.tw/gocwiki/How_to_migrate_the_RLS_entries_into_the_LCG_File_Catalog_%28LFC%29

  - Accessible by defining: $LCG_CATALOG_TYPE=lfc and $LFC_HOST
  - Better performance and scalability
  - Provides new features: security, hierarchical namespace, transactions...

The RLS (the past)

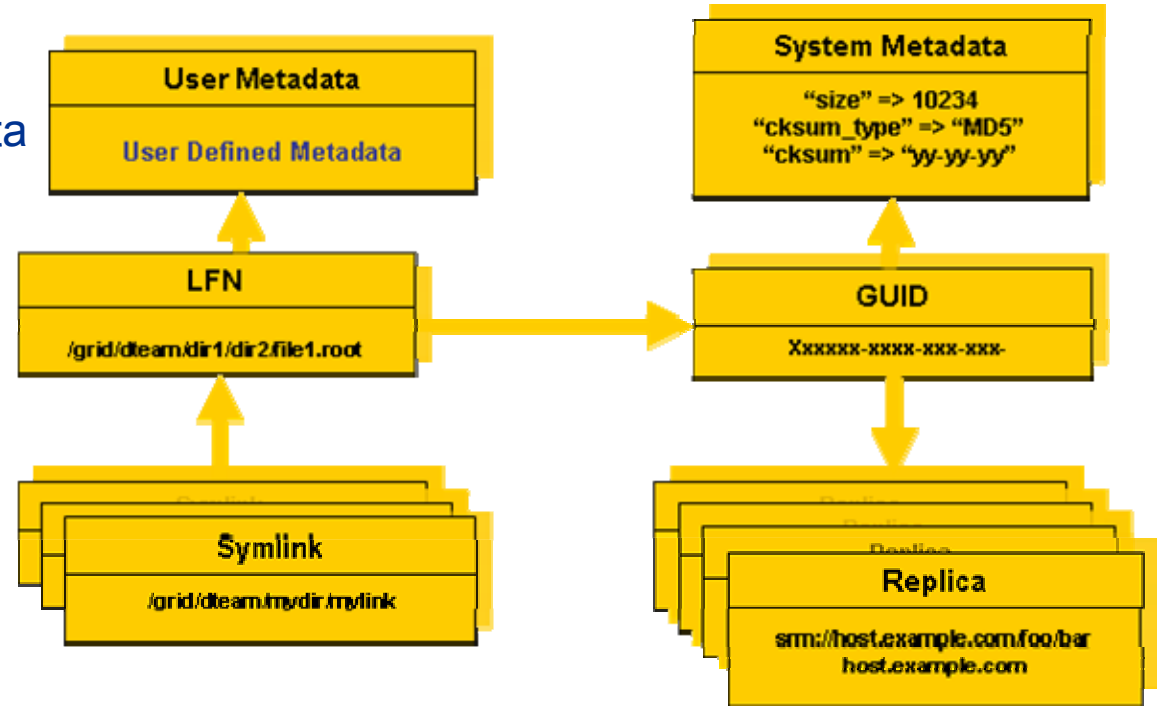
- **RMC:**
  - Stores LFN-GUID mappings
  - Accessible by edg-rmc CLI + API
- **RLS:**
  - Stores GUID-SURL mappings
  - Accessible by edg-lrc CLI + API
- **Main weaknesses:**
  - Insecure (anyone can delete catalog entries)
  - Bad performance (java clients…)





The LFC (the present)

- **One single catalog**
- **LFN acts as main key in the database. It has:**
  - Symbolic links to it (additional LFNs)
  - Unique Identifier (GUID)
  - System metadata
  - Information on replicas
  - One field of user metadata



The LFC (II)

- **Fixes EDG catalogs performance and scalability problems**
  - Cursors for large queries
  - Timeouts and retries from the client
- **Provides more features than the EDG Catalogs**
  - User exposed transaction API (+ auto rollback on failure)
  - Hierarchical namespace and namespace operations (for LFNs)
  - Integrated GSI Authentication + Authorization
  - Access Control Lists (Unix Permissions and POSIX ACLs)
  - Checksums
- **New features will be added soon (requests welcome!)**
  - Integration with VOMS, FiReMan
  - POOL Integration is in progress
  - Sessions
  - Bulk operations

LFC Interfaces

- **LFC client commands**
  - Provide administrative functionality
  - Unix-like
  - LFNs seen as a Unix filesystem (/grid/<VO>/ …)
- **LFC C API**
  - Alternative way to administer the catalog
  - Python wrapper provided
- **Integration with GFAL and lcg_util APIs complete**
  - ➔ lcg-utils access the catalog in a transparent way
- **Integration with the WMS completed**
  - The RB can locate Grid files: allows for data based match-making
  - Using the Data Location Interface
  - Not yet tested in production

Data Management CLIs & APIs

- **lcg_utils: lcg-* commands + lcg_* API calls**
  - Provide (all) the functionality needed by the LCG user
  - Transparent interaction with file catalogs and storage interfaces when needed
  - Abstraction from technology of specific implementations
- **Grid File Access Library (GFAL): API**
  - Adds file I/O and explicit catalog interaction functionality
  - Still provides the abstraction and transparency of lcg_utils
- **edg-gridftp tools: CLI**
  - Complete the lcg_utils with low level GridFTP operations
  - Functionality available as API in GFAL
  - May be generalized as lcg-* commands

lcg-utils commands

Replica Management

lcg-cp	Copies a grid file to a local destination
lcg-cr	Copies a file to a SE and registers the file in the catalog
lcg-del	Delete one file
lcg-rep	Replication between SEs and registration of the replica
lcg-gt	Gets the TURL for a given SURL and transfer protocol
lcg-sd	Sets file status to “Done” for a given SURL in a SRM request

File Catalog Interaction

lcg-aa	Add an alias in LFC for a given GUID
lcg-ra	Remove an alias in LFC for a given GUID
lcg-rf	Registers in LFC a file placed in a SE
lcg-uf	Unregisters in LFC a file placed in a SE
lcg-la	Lists the alias for a given SURL, GUID or LFN
lcg-lg	Get the GUID for a given LFN or SURL
lcg-lr	Lists the replicas for a given GUID, SURL or LFN

LFC C API

Low level methods (many POSIX-like):

lfc_access
lfc_aborttrans
lfc_addreplica
lfc_apiinit
lfc_chclass
lfc_chdir
lfc_chmod
lfc_chown
lfc_closedir
lfc_creat
lfc_delcomment
lfc_delete
lfc_deleteclass
lfc_delreplica
lfc_endtrans
lfc_enterclass
lfc_errmsg
lfc_getacl
lfc_getcomment
lfc_getcwd
lfc_getpath
lfc_lchown
lfc_listclass
lfc_listlinks
lfc_listreplica
lfc_lstat
lfc_mkdir
lfc_modifyclass
lfc_opendir
lfc_queryclass
lfc_readdir
lfc_readlink
lfc_rename
lfc_rewind
lfc_rmdir
lfc_selectsrvr
lfc_setacl
lfc_setatime
lfc_setcomment
lfc_seterrbuf
lfc_setfsize
lfc_starttrans
lfc_stat
lfc_symlink
lfc_umask
lfc_undelete
lfc_unlink
lfc_utime
send2lfc

LFC commands

Summary of the LFC Catalog commands

lfc-chmod	Change access mode of the LFC file/directory
lfc-chown	Change owner and group of the LFC file-directory
lfc-delcomment	Delete the comment associated with the file/directory
lfc-getacl	Get file/directory access control lists
lfc-ln	Make a symbolic link to a file/directory
lfc-ls	List file/directory entries in a directory
lfc-mkdir	Create a directory
lfc-rename	Rename a file/directory
lfc-rm	Remove a file/directory
lfc-setacl	Set file/directory access control lists
lfc-setcomment	Add/replace a comment

LFC other commands

Managing ownership and permissions:

- lfc-chmod
- lfc-chown

Managing ACLs:

- lfc-getacl
- lfc-setacl

Remember that per user mapping can change in every session.

The default is for LFNs and directories to be VO-wide readable.

Consistent user mapping will be added soon.

Renaming:

- lfc-rename

Removing:

- lfc-rm

An LFN can only be removed if it has no SURLs associated.

LFNs should be removed by lcg-del, rather than lfc-rm.

Bibliography

- Information on the file catalogs
  - LFC, gfal, lcg-utils:

    "Evolution of LCG-2 Data Management (J-P Baud, J. Casey)"

    http://indico.cern.ch/contributionDisplay.py?contribId=278&sessionId=7&confId=0

  - LFC installation, administration, migration from RLS:
    - Wiki entries indicated through the presentation:
      - *http://goc.grid.sinica.edu.tw/gocwiki/How_to_set_up_an_LFC_service*
      - *http://goc.grid.sinica.edu.tw/gocwiki/How_to_migrate_the_RLS_entries_into_the_LCG_File_Catalog_%28LFC%29*

  - LFC contacts:
    - Jean-Philippe.Baud@cern.ch
    - Sophie.Lemaitre@cern.ch

File and Replica Management catalog (FiReMan)

(the future)

Cataloguing Requirements

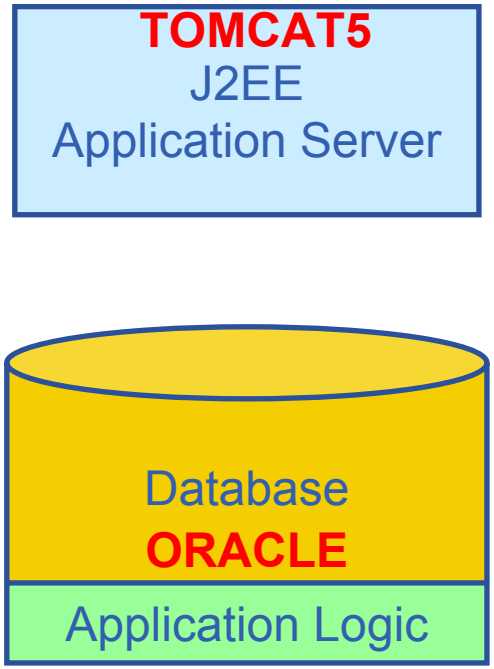
- **Catalogs built based on requirements from HEP experiments and the Biomedical EGEE community**
- **Started design from AliEn File Catalog**
  - Logical namespace management
  - Virtual Filesystem view (DataSets via directory hierarchy)
  - Support Metadata attached to files
  - Bulk Operations
  - Strong security: basic unix permissions and fine-grained ACLs (i.e. not just directory but file-granularity)
  - Support flexible deployment models
    - Single central catalog model
    - Site local catalogs connected to a single central catalog model
    - Site local catalogs without single central catalog model
  - Scalable to many clients and to a large number of entries; address performance issues seen with EDG RLS

Service Implementation

- 2 independent implementations exist

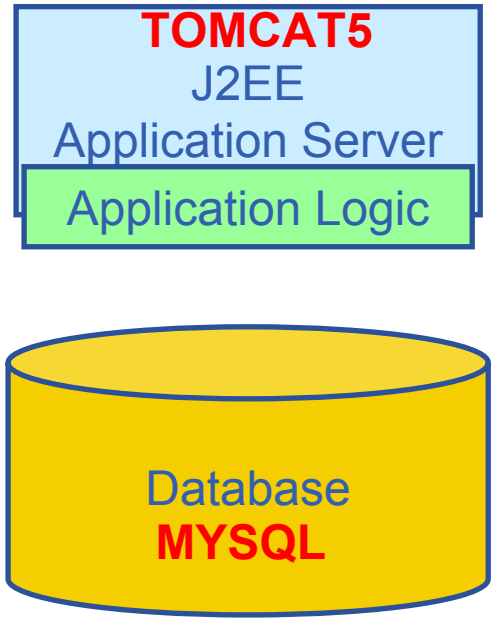
Oracle Implementation

- Catalog Logic lives inside Oracle as Stored Procedures
- Tomcat parses credential only, passes operations through to DB



MySQL Implementation

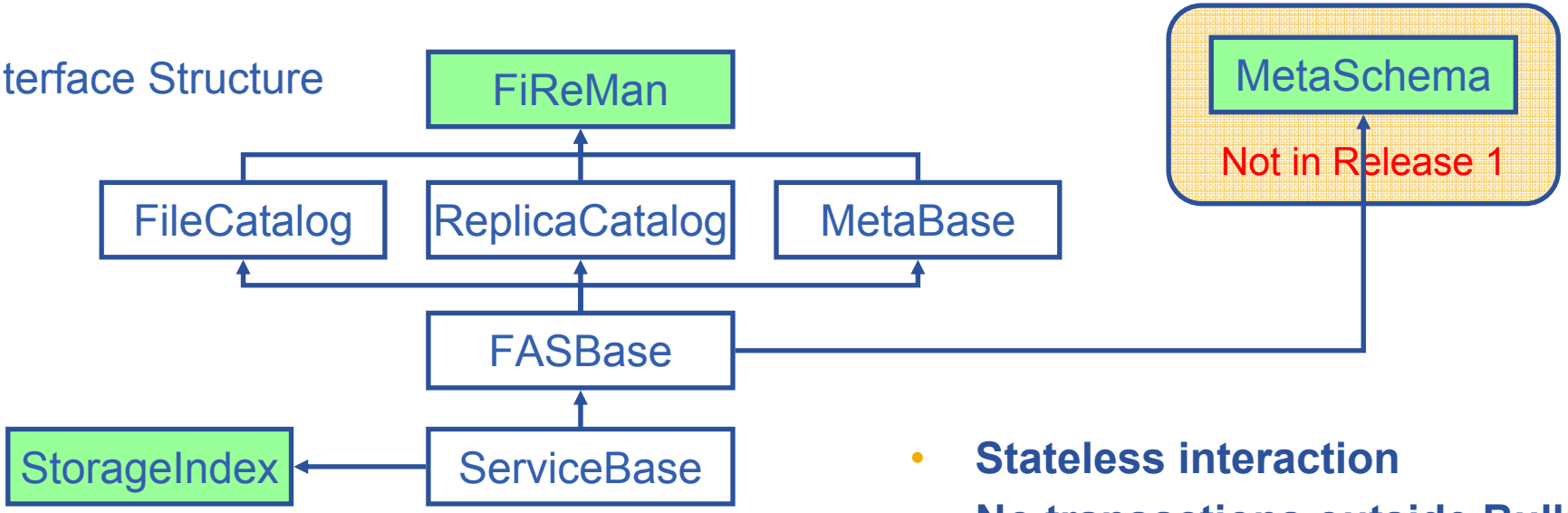
- Simple Table Structure using InnoDB tables
- Credential parsing and all of the logic is in Tomcat



Fireman Catalog Interface

- Logical File Namespace management — FileCatalog
- Replica locations — ReplicaCatalog
- File-based metadata — MetaBase
- Metadata Management — MetaSchema
- Authentication and Authorization information (ACLs) — FASBase
- Service Metadata — ServiceBase
- WMS interaction and global file location — ServiceIndex

Interface Structure



- Stateless interaction
- No transactions outside Bulk

Features

- **Web-services interface**: Guarantees client support on many platforms and many languages.
- **Standardization effort ongoing**. It is being managed through the EGEE PTF. Are provided:
  - **Linux Command Line tools**
  - **C/C++ API**
  - **Java API**
  - **Perl modules**
  - **JavaScript (for web clients)**
  - **gLite integrated bash (glitesh) – prototype**
- **Security:** Fine-grained ACL support with minimal performance penalty.
  - DNs own the files
  - VOMS group support
  - Basic Unix security (ugo rwx)
  - Additional ACLs for setPermission, list, remove, setMetadata, getMetadata

gLite Catalog Releases

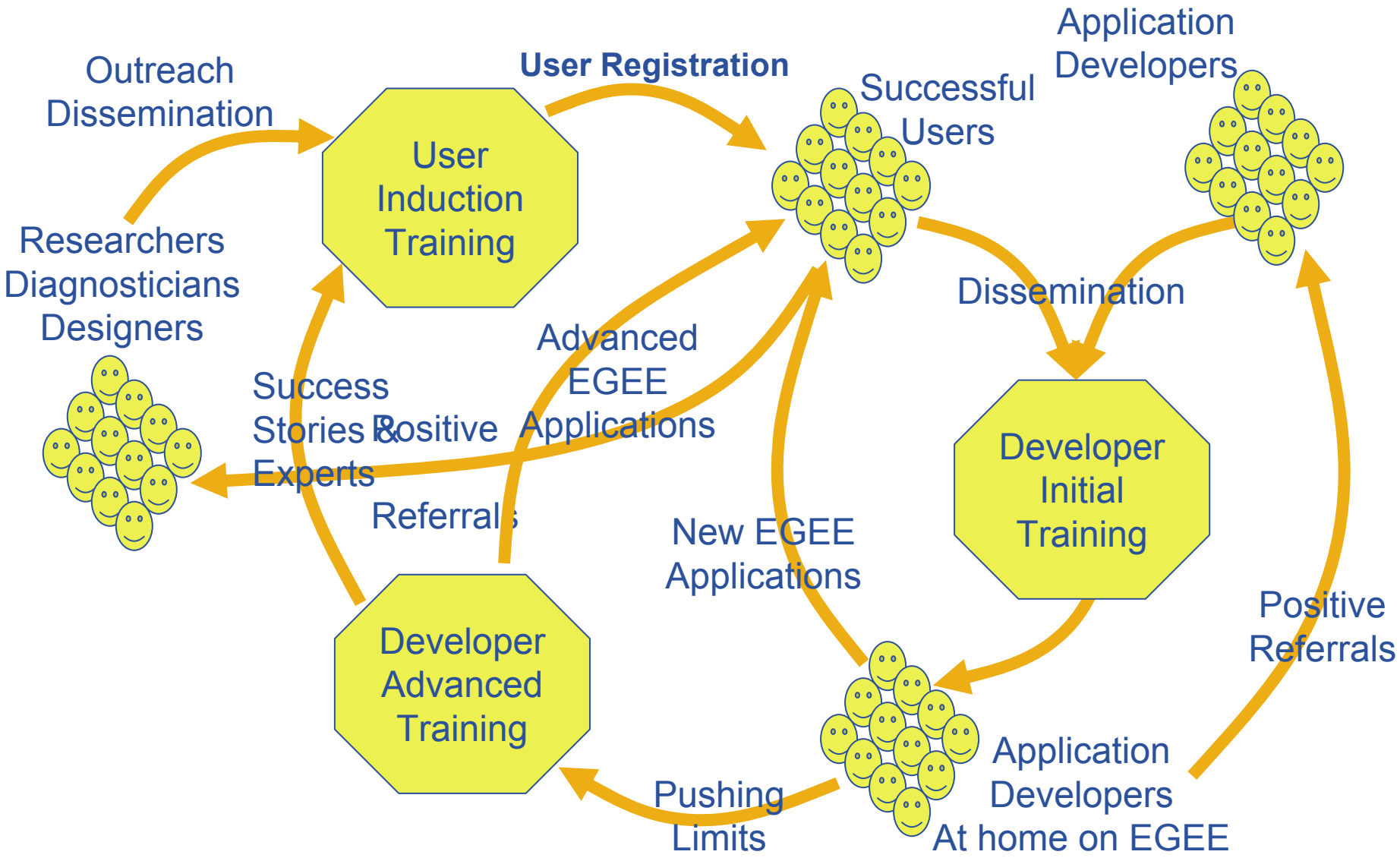
- **FiReMan Catalog**
  - Release 1: Single Central deployment model only
  - Release 2: Distributed catalog according to design using Java Messaging Services to propagate updates between catalog instances
- **Storage Index**
  - Already in Release 1
  - Main interaction point with Workload Management
- **Metadata Catalog**
  - Release 1: Base Implemented by FiReMan
  - Also a standalone service, single central instance
  - Release 2: distribution using a messaging infrastructure

For More Information

- **JRA1 Data Management homepage**
  http://cern.ch/egee-jra1-dm
- **gLite FiReMan user guide**
  - Overview
    https://edms.cern.ch/file/570643/1/EGEE-TECH-570643-v1.0.pdf
  - Command Line tools
    https://edms.cern.ch/file/570780/1/EGEE-TECH-570780-v1.0.pdf
  - C/C++ API
    https://edms.cern.ch/file/570780/1/EGEE-TECH-570780-C-CPP-API-v1.0.pdf
  - Java API
    https://edms.cern.ch/file/570780/1/EGEE-TECH-570780-JAVA-API-v1.0.pdf
- **gLite Release 1**
  - http://glite.web.cern.ch/glite/packages/R1.0/R20050331
  - http://glite.web.cern.ch/glite/documentation

The GILDA t-Infrastructure

EGEE Virtuous Cycle

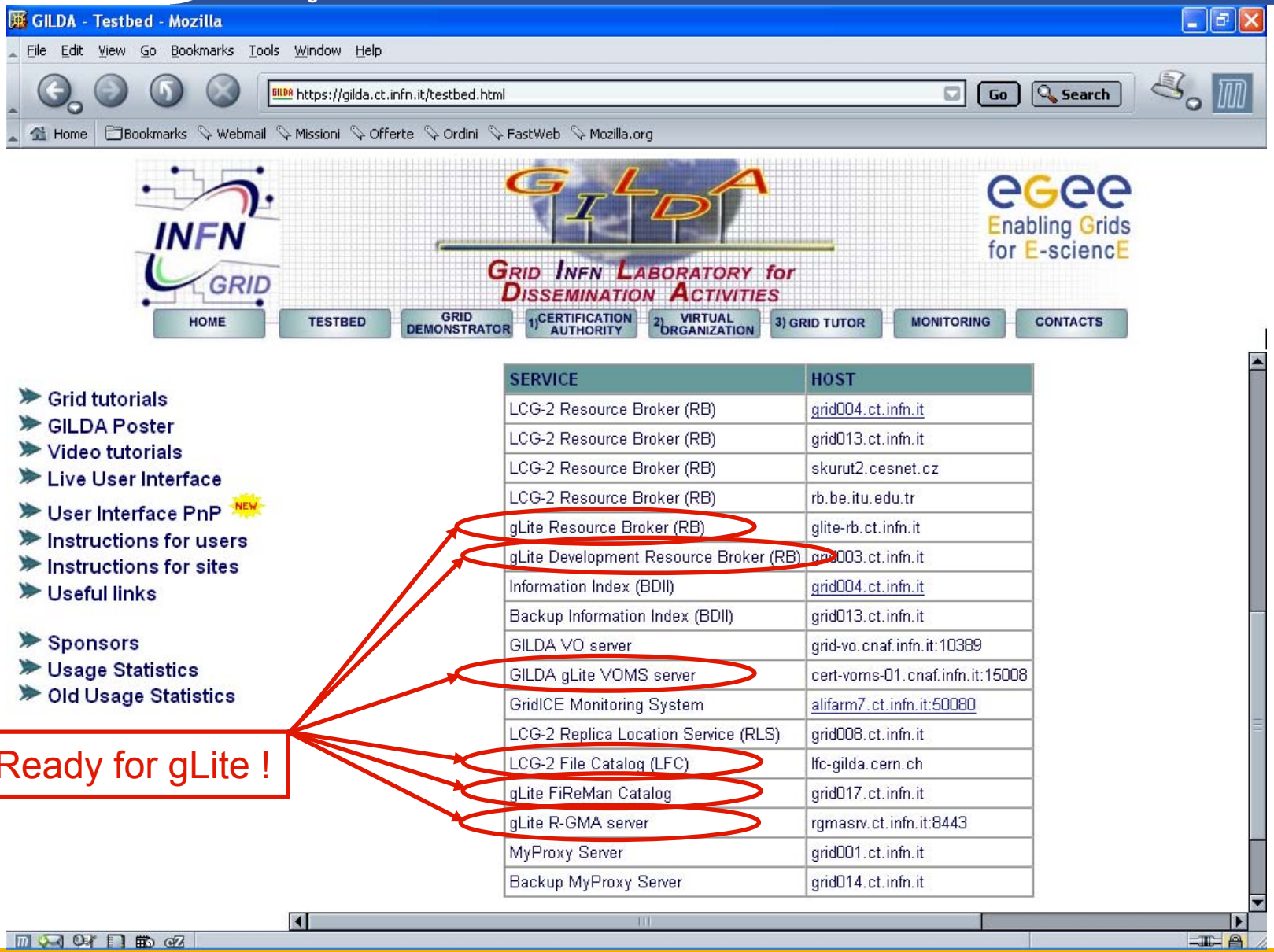


The GILDA project (https://gilda.ct.infn.it)

The GILDA Test-bed (https://gilda.ct.infn.it/testbed.html)

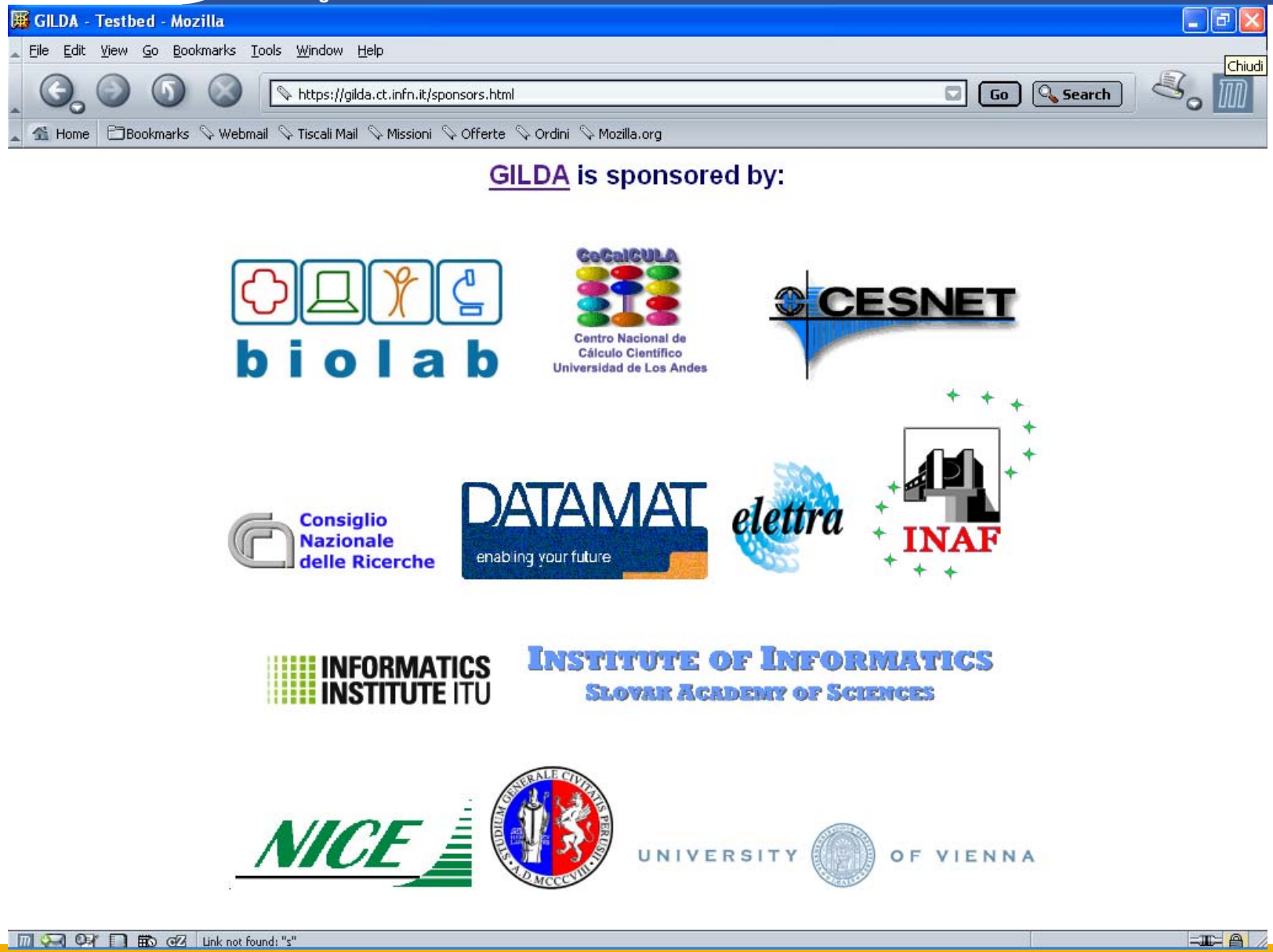
15 sites in 3 continents !

The GILDA Services (https://gilda.ct.infn.it/testbed.html)

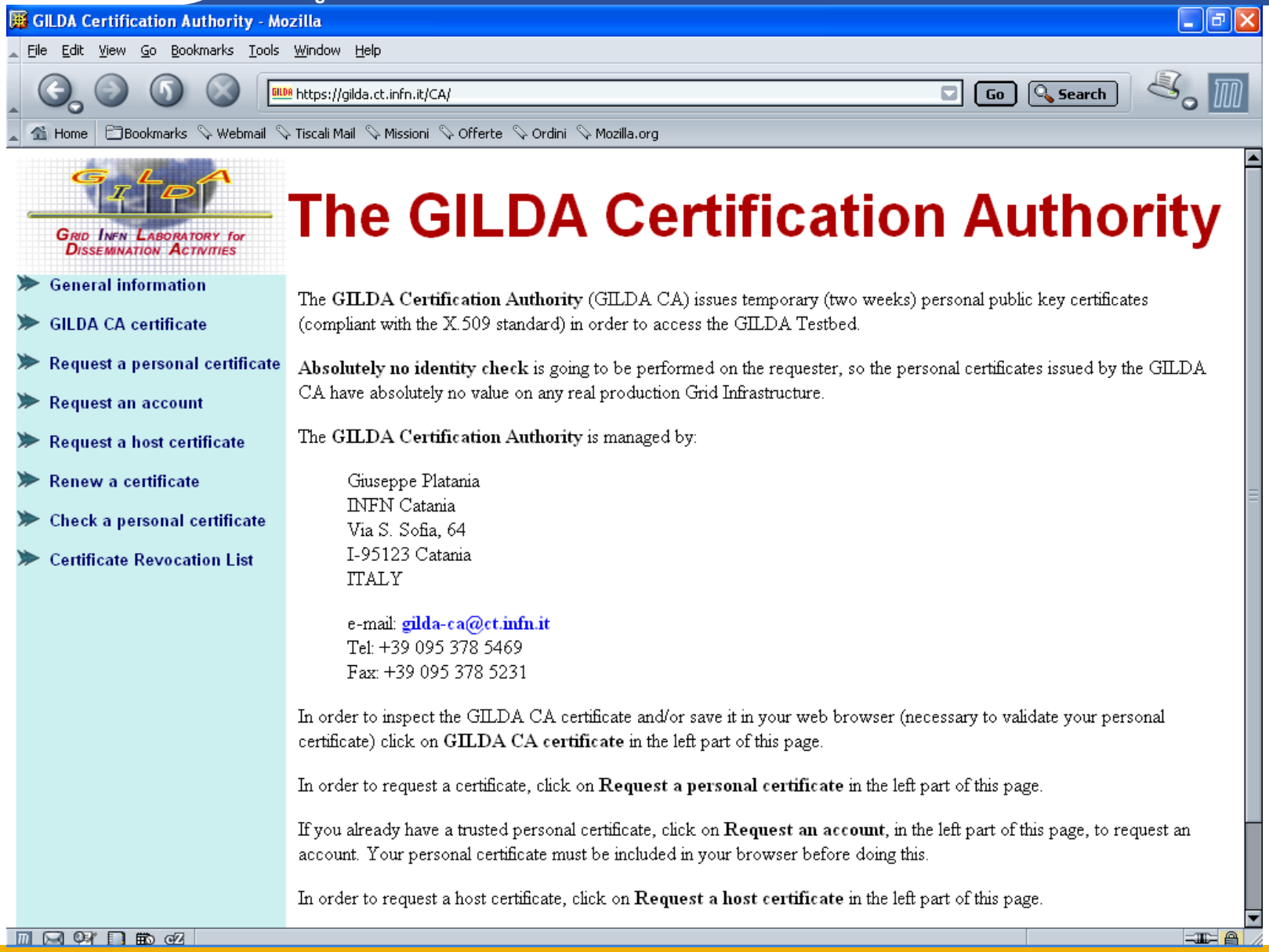


SERVICE	HOST
LCG-2 Resource Broker (RB)	grid004.ct.infn.it
LCG-2 Resource Broker (RB)	grid013.ct.infn.it
LCG-2 Resource Broker (RB)	skurut2.cesnet.cz
LCG-2 Resource Broker (RB)	rb.be.itu.edu.tr
gLite Resource Broker (RB)	glite-rb.ct.infn.it
gLite Development Resource Broker (RB)	grid003.ct.infn.it
Information Index (BDII)	grid004.ct.infn.it
Backup Information Index (BDII)	grid013.ct.infn.it
GILDA VO server	grid-vo.cnaf.infn.it:10389
GILDA gLite VOMS server	cert-voms-01.cnaf.infn.it:15008
GridICE Monitoring System	alifarm7.ct.infn.it:50080
LCG-2 Replica Location Service (RLS)	grid008.ct.infn.it
LCG-2 File Catalog (LFC)	lfc-gilda.cern.ch
gLite FiReMan Catalog	grid017.ct.infn.it
gLite R-GMA server	rgmasrv.ct.infn.it:8443
MyProxy Server	grid001.ct.infn.it
Backup MyProxy Server	grid014.ct.infn.it

The GILDA Sponsors (https://gilda.ct.infn.it/sponsors.html)

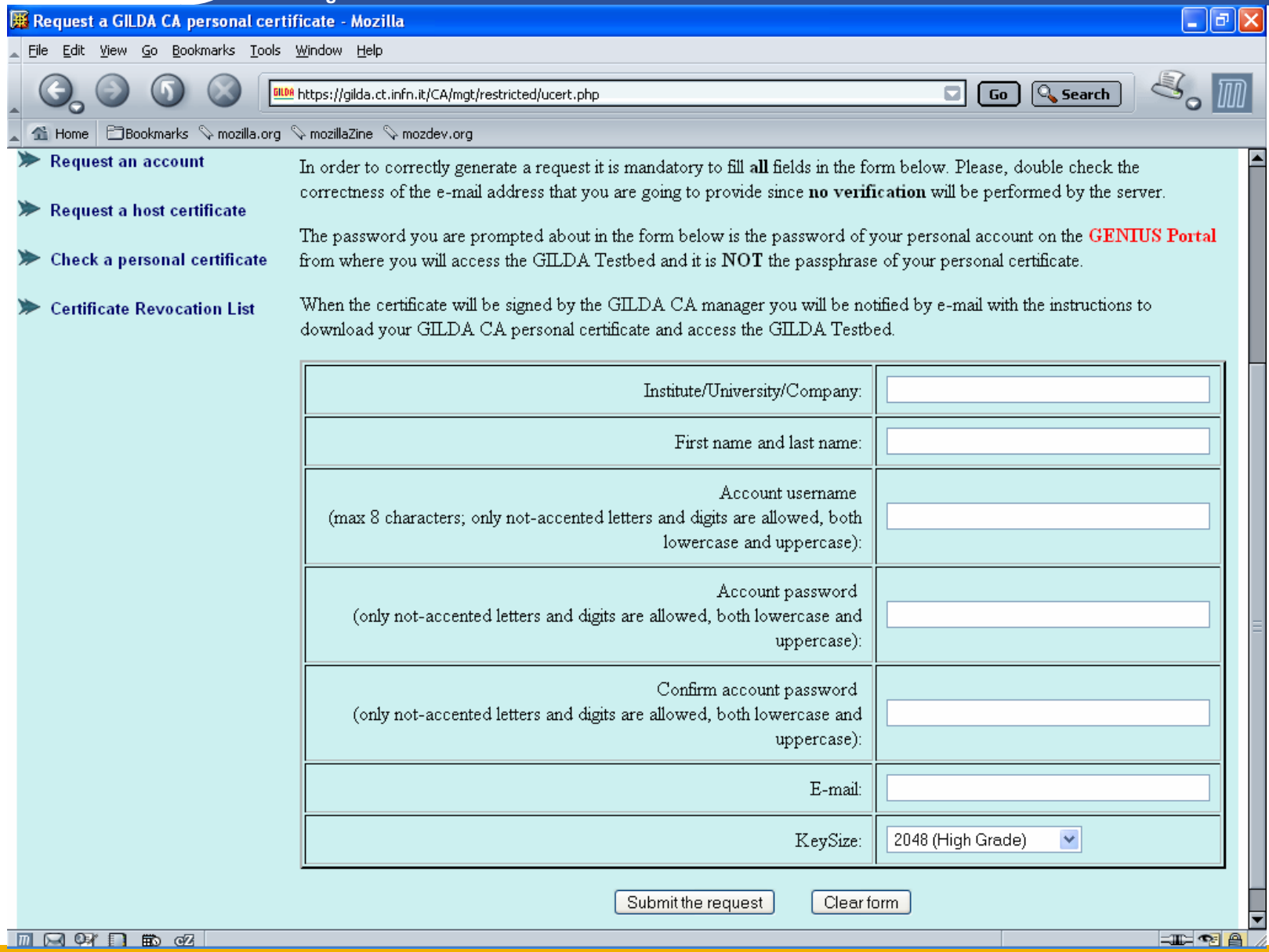


The GILDA Certification Authority (https://gilda.ct.infn.it/CA)



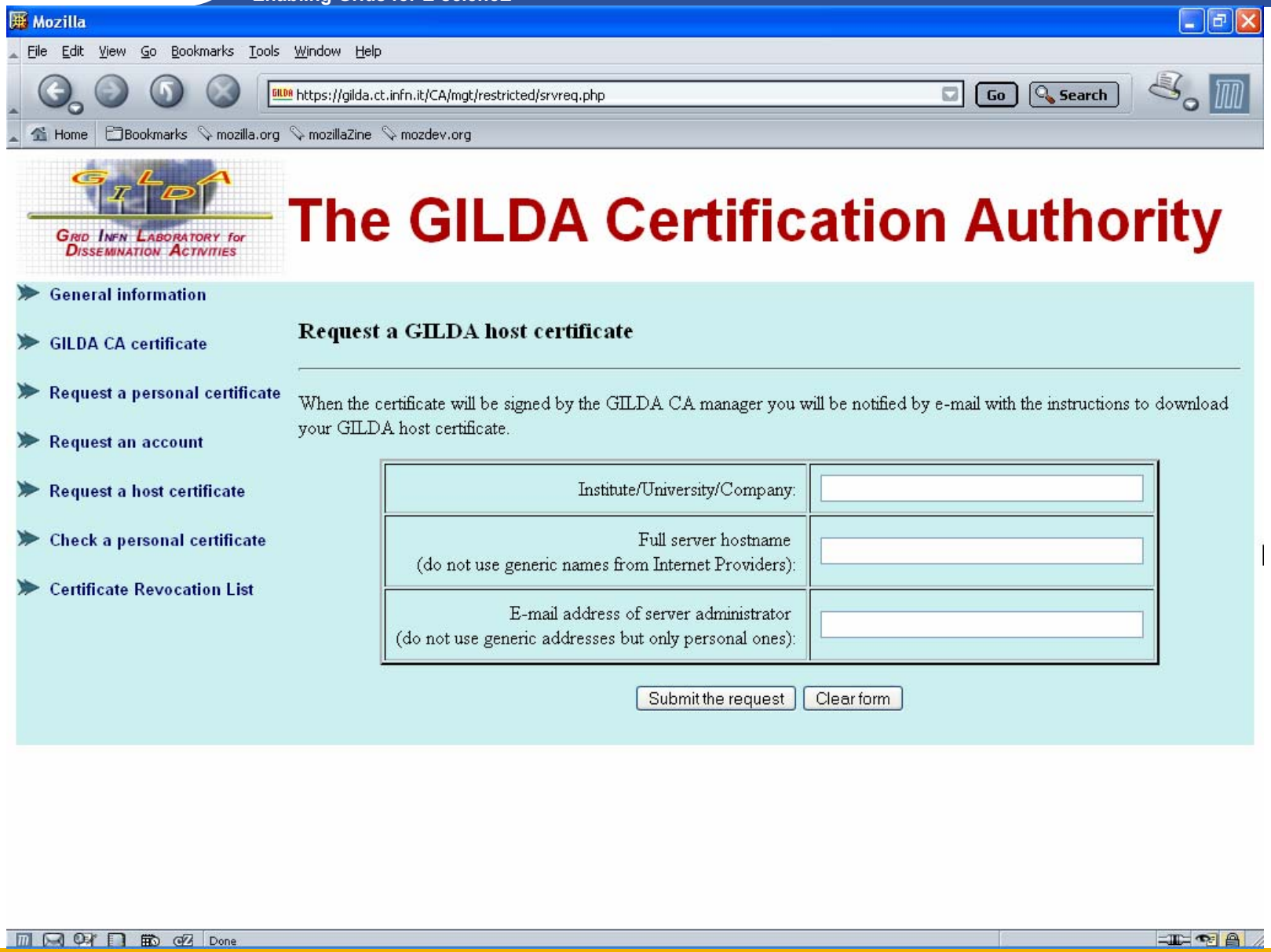
The GILDA Certification Authority

Enabling Grids for E-sciencE

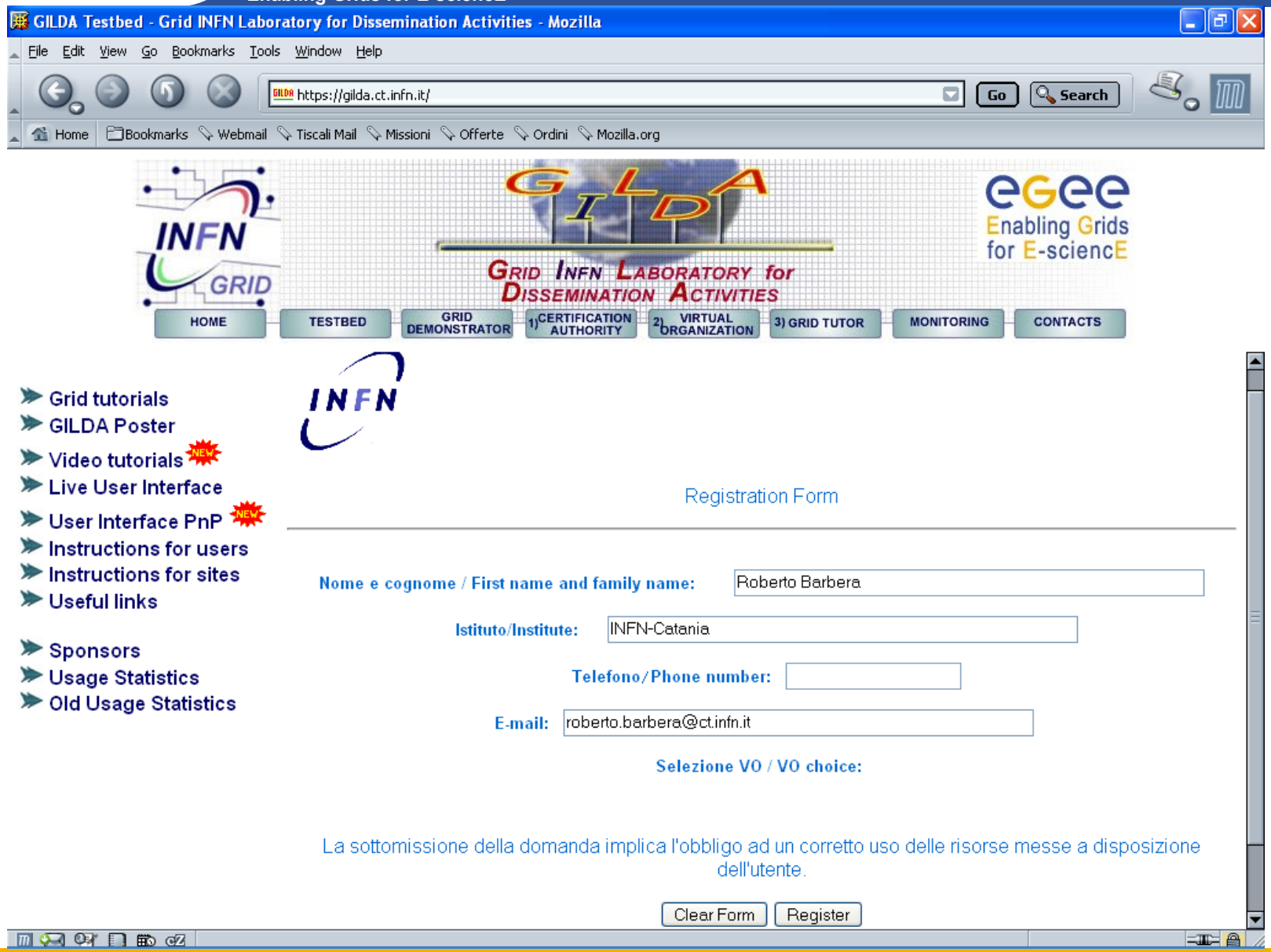


The GILDA Certification Authority (4/4)

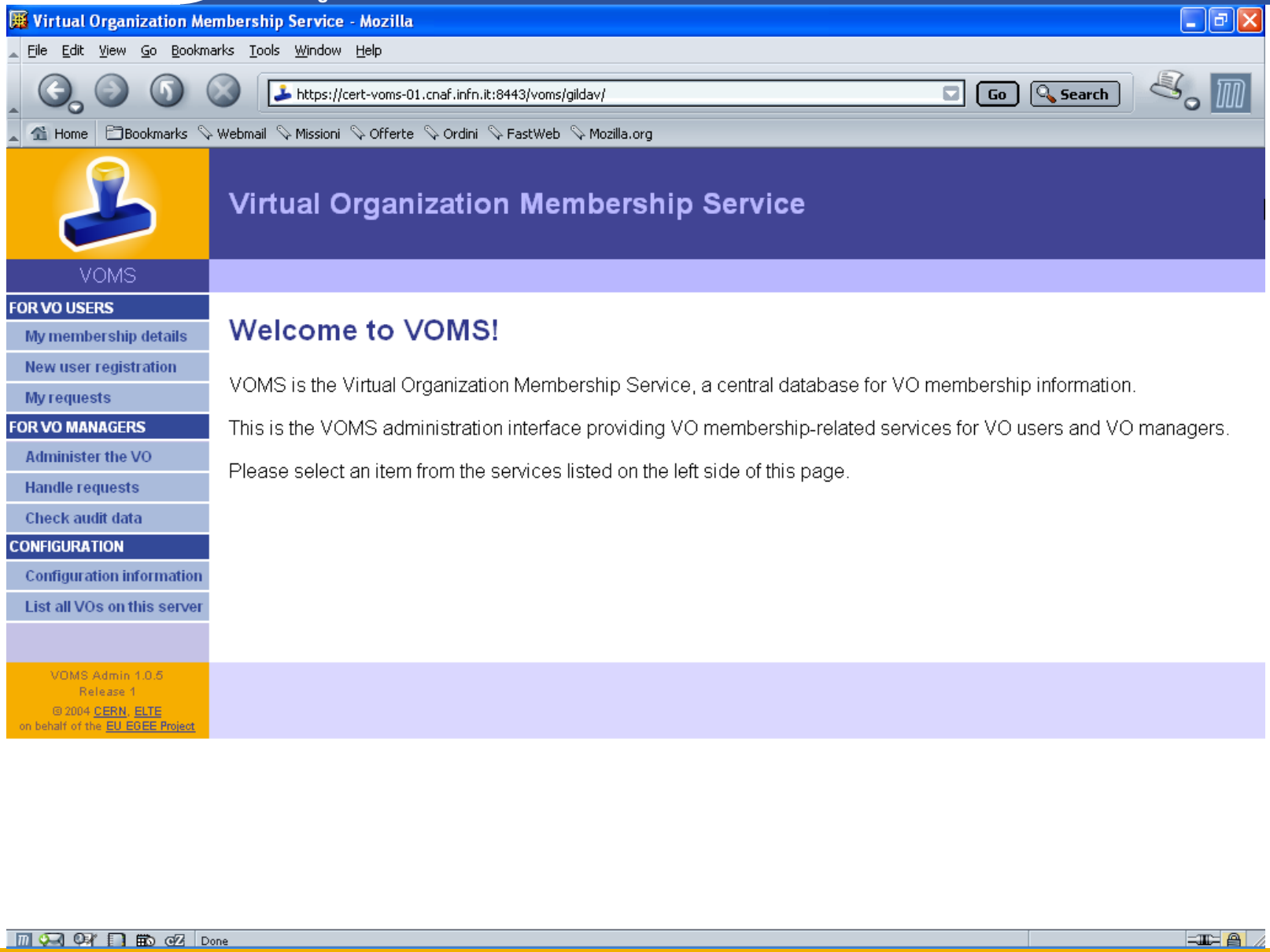
Enabling Grids for E-sciencE



The GILDA Virtual Organization

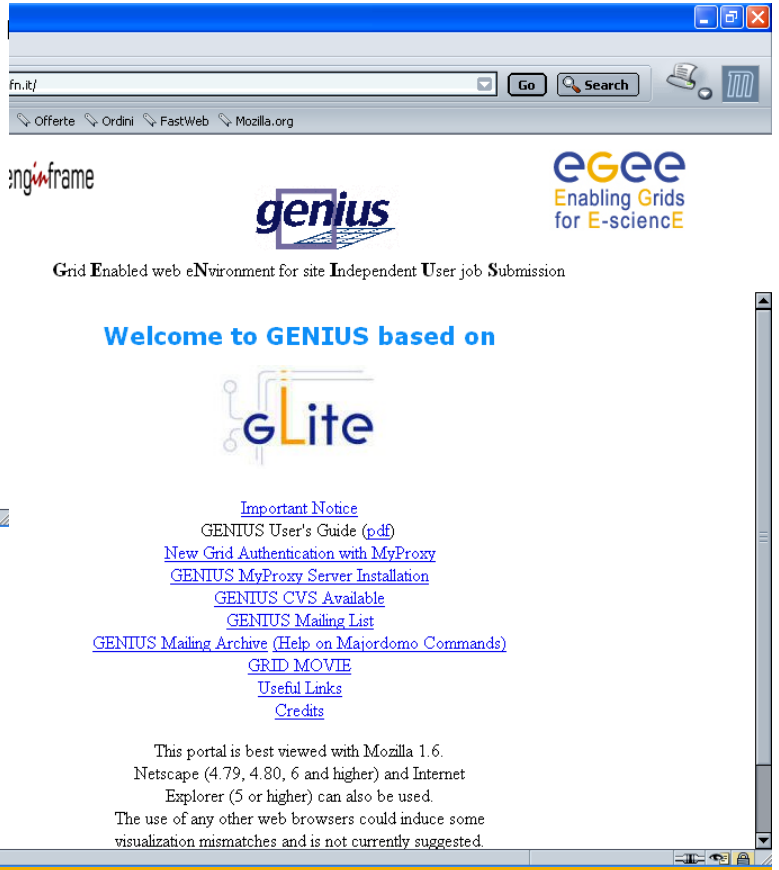


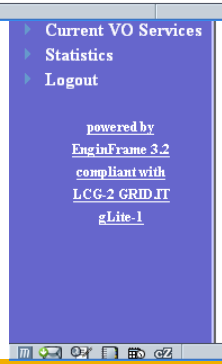
The GILDA VOMS



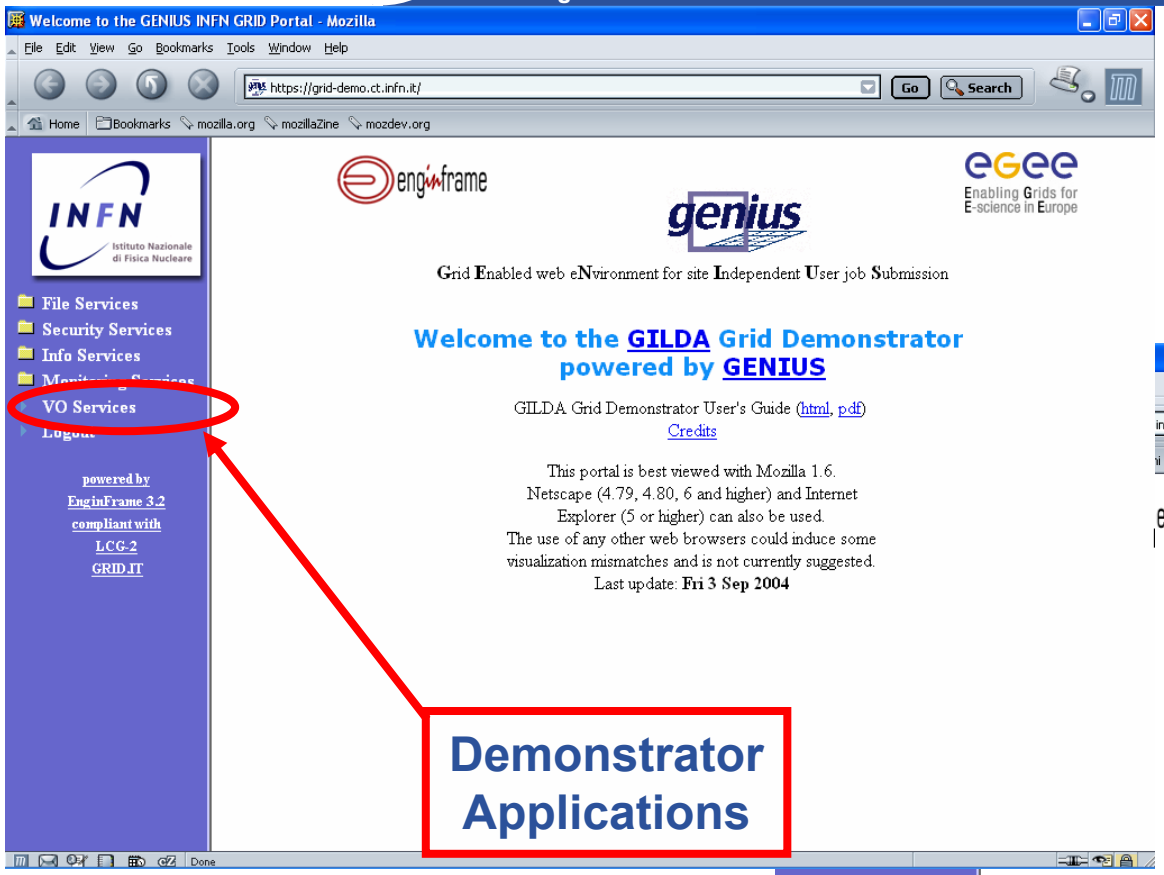
The GILDA Monitoring System (http://alifarm7.ct.infn.it:50080/gridice)

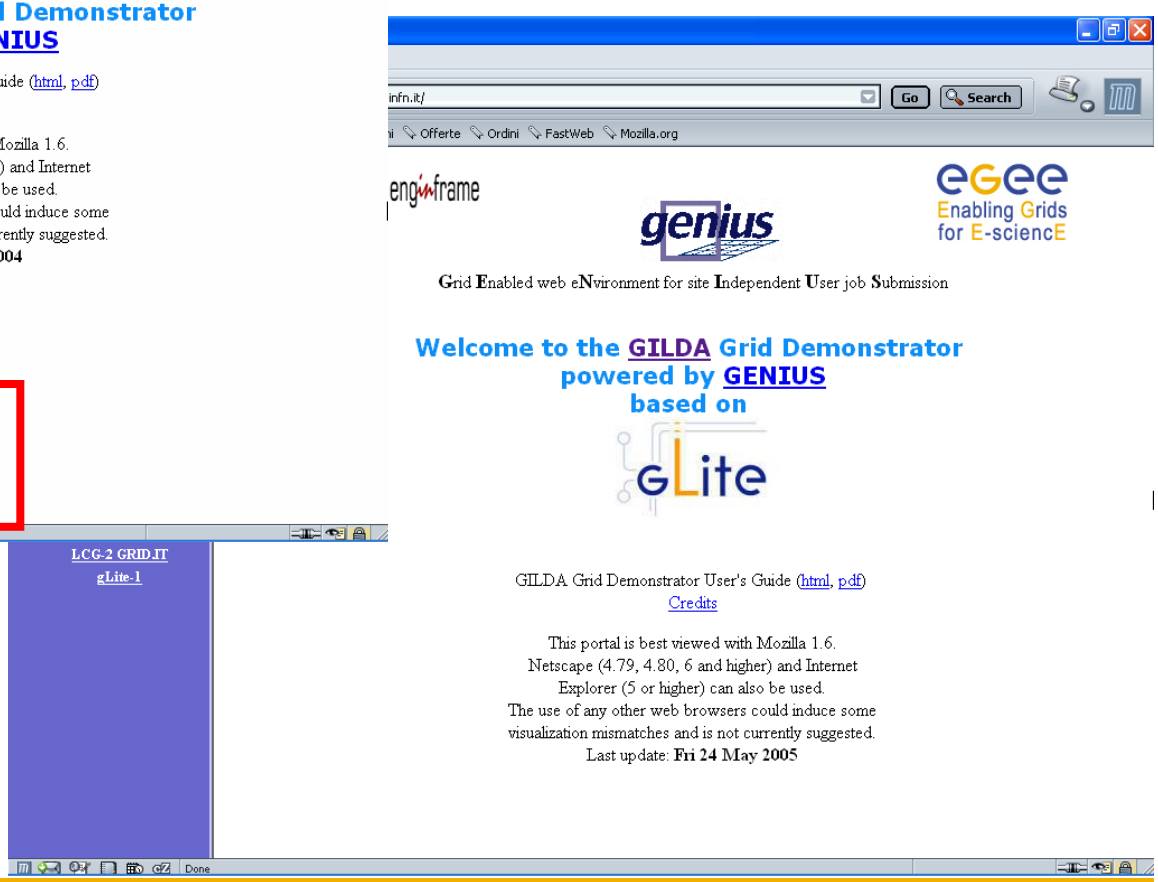
The Grid Tutor
(https://grid-tutor.ct.infn.it, https://glite-tutor.ct.infn.it)



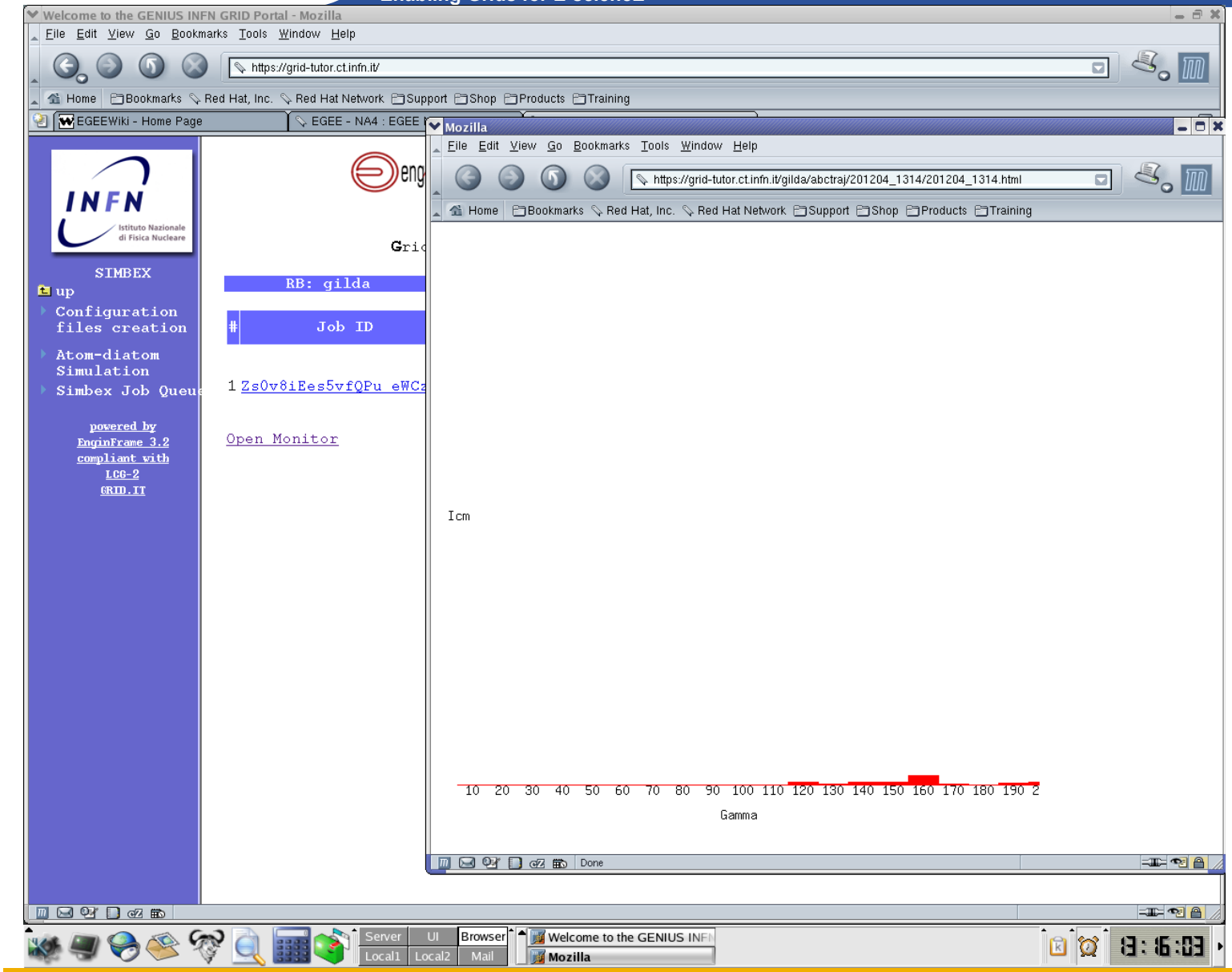


The Grid Demonstrator (1/2)
(https://grid-demo.ct.infn.it, https://glite-demo.ct.infn.it)





GEMS example

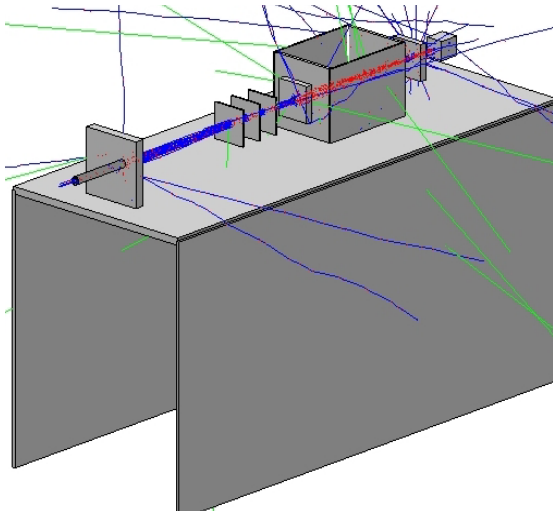


Interactive MPI jobs !

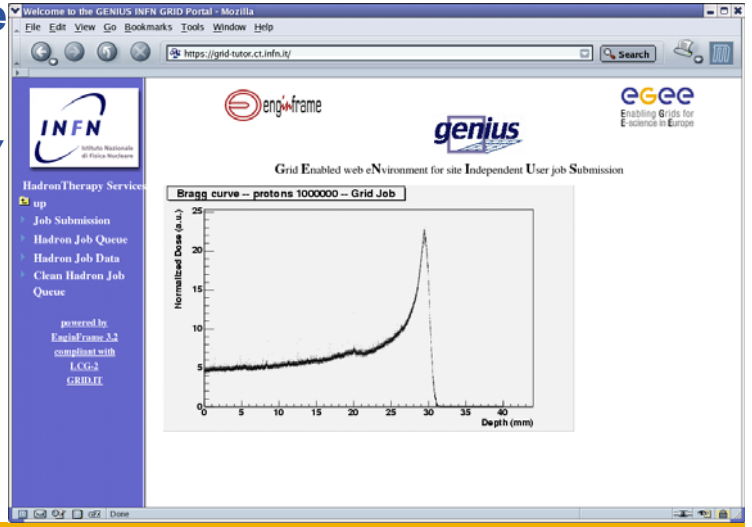
hadronTherapy example

CATANA beam line in reality

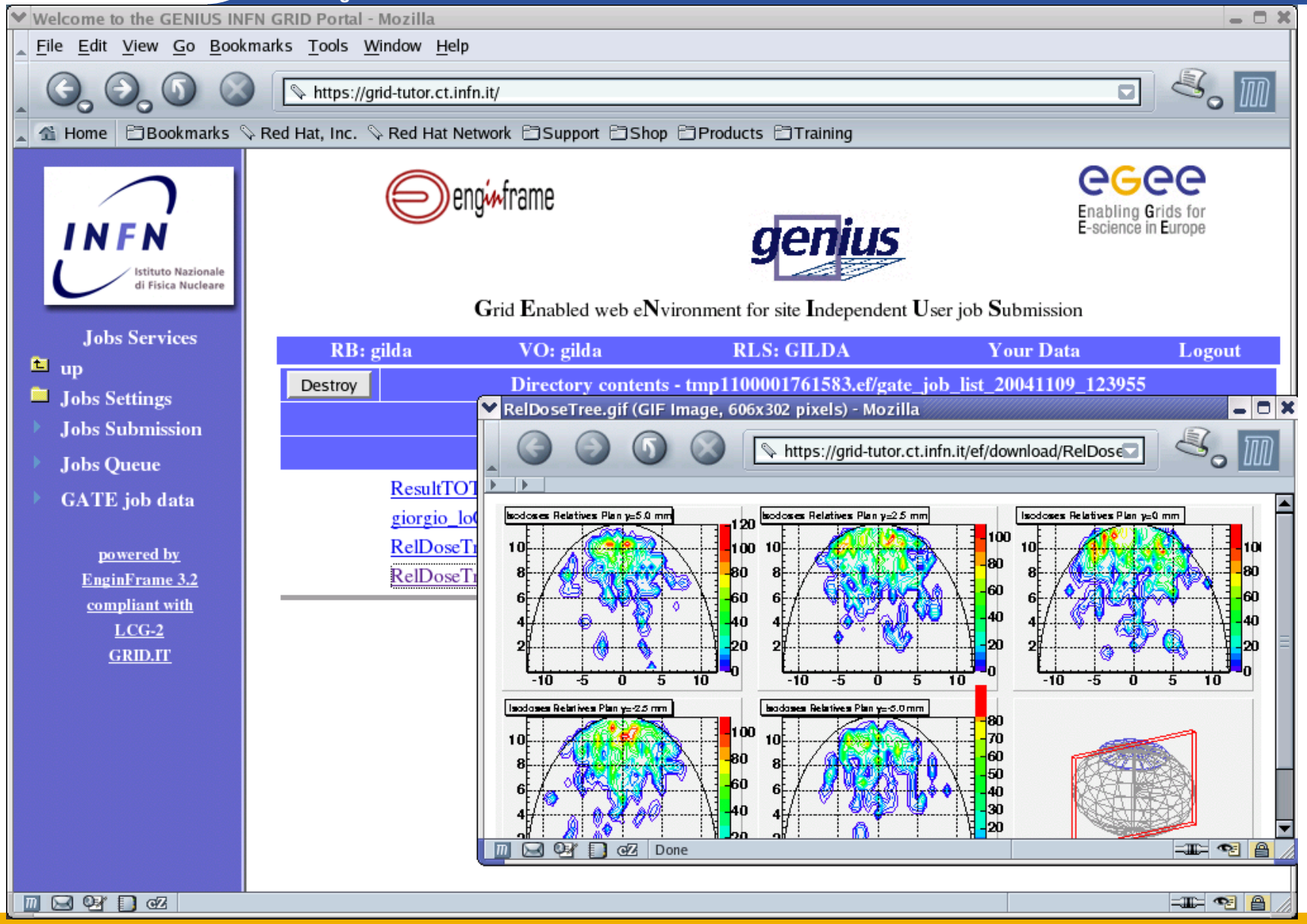
hadronTherapy in GENIUS



CATANA beam line simulated by *hadronTherapy*

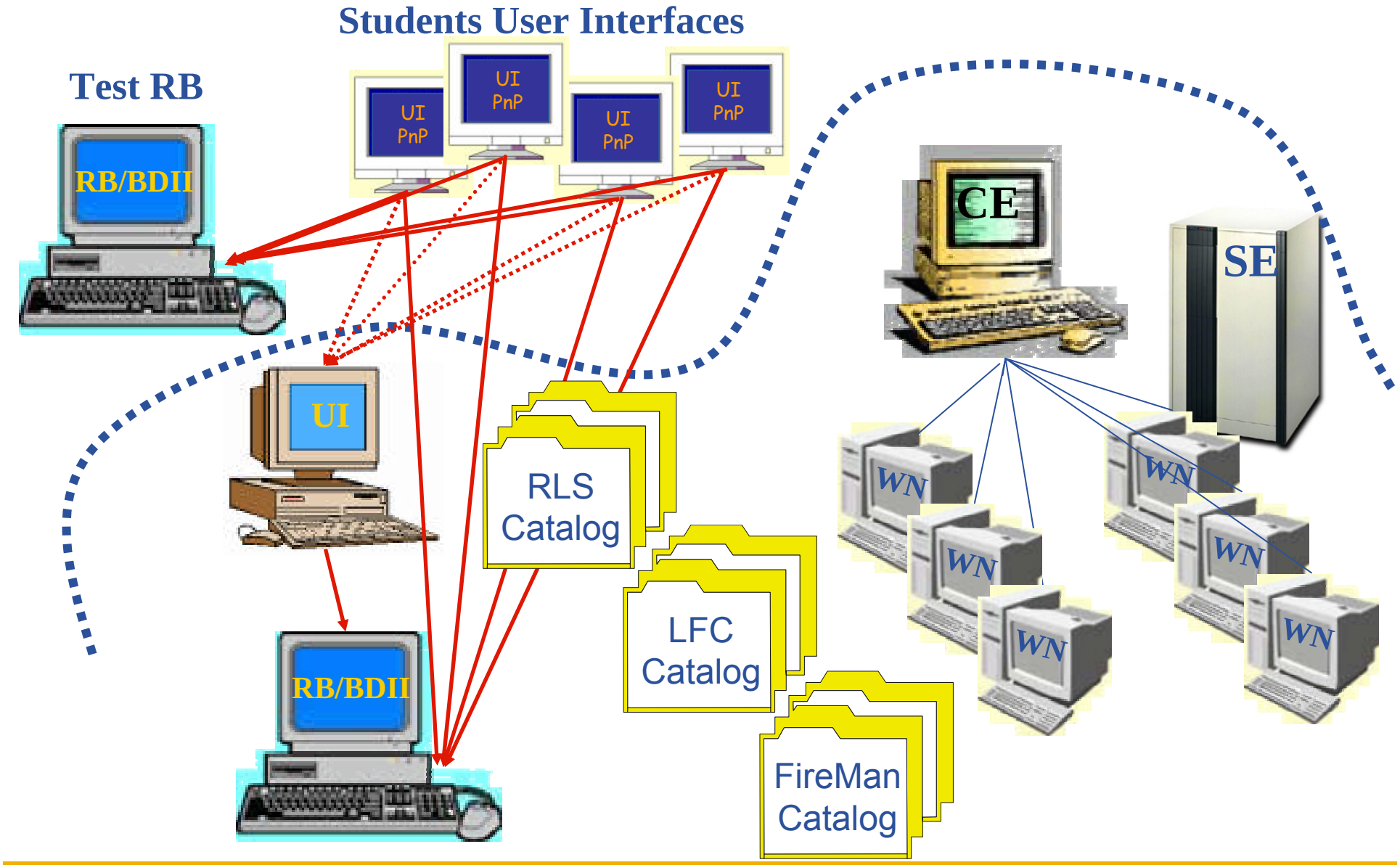


GATE example

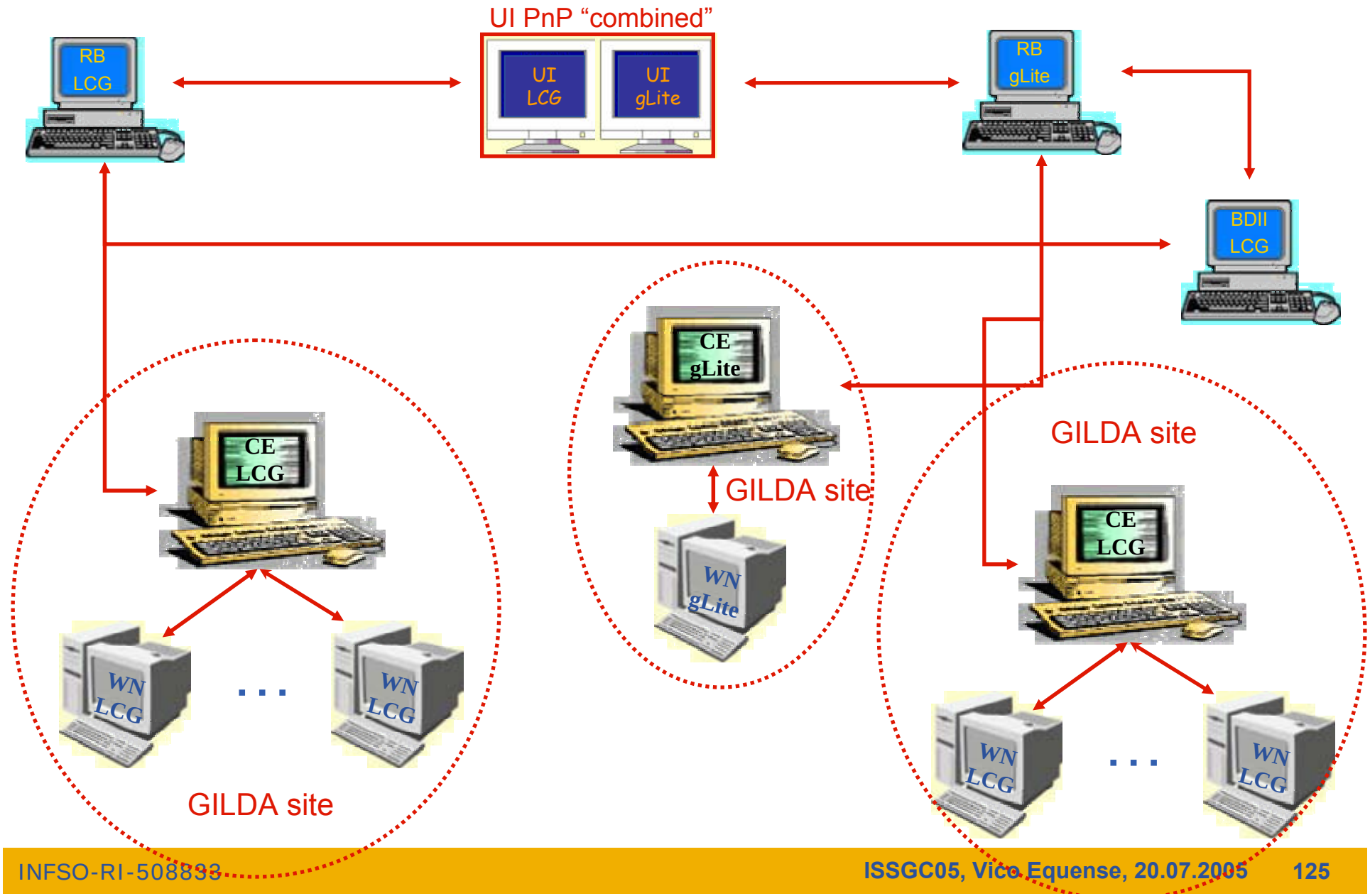


The GILDA User Interface Plug&Play combined (https://gilda.ct.infn.it/UIPnPcomb/)

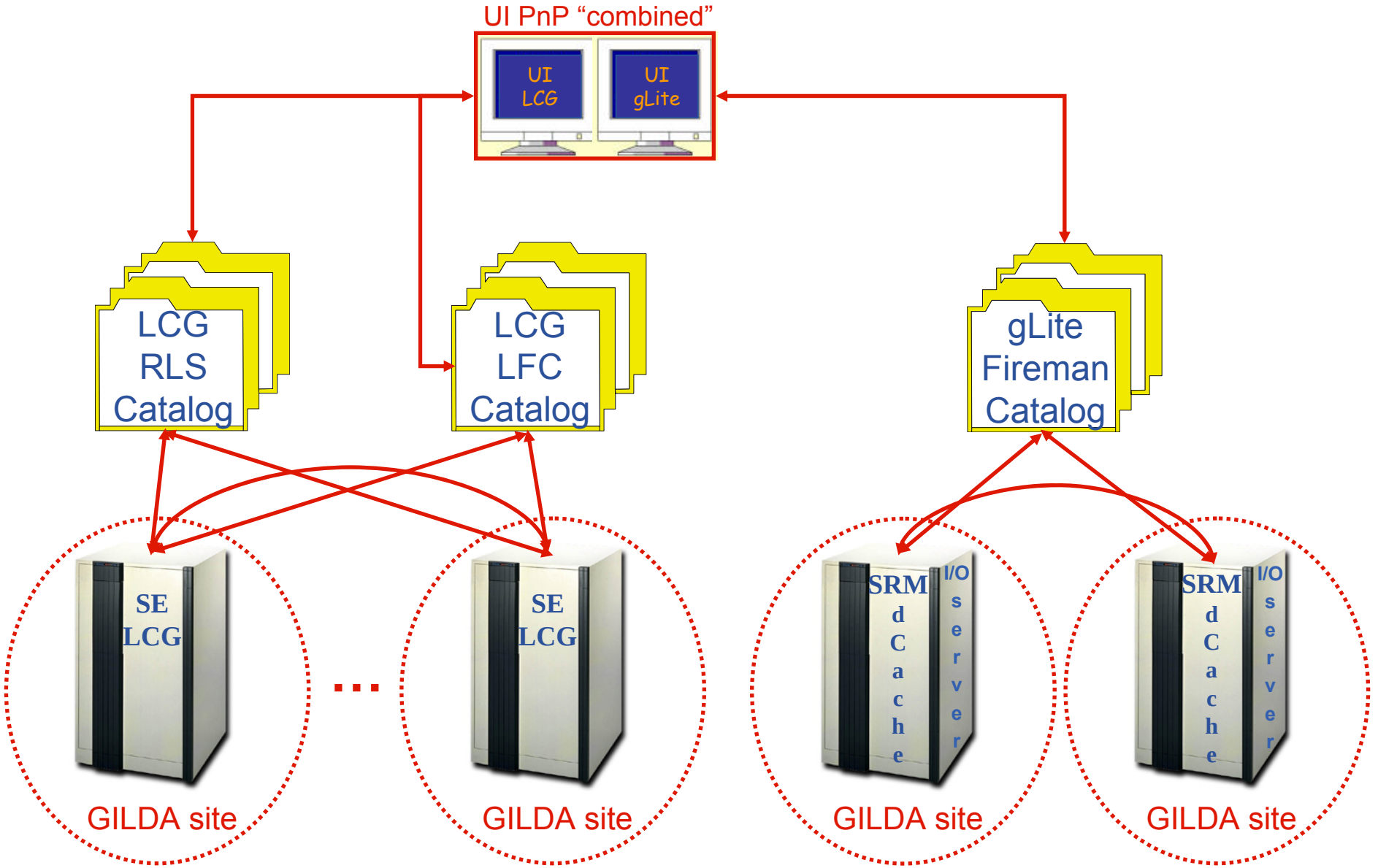
Tutorial layout and acronyms



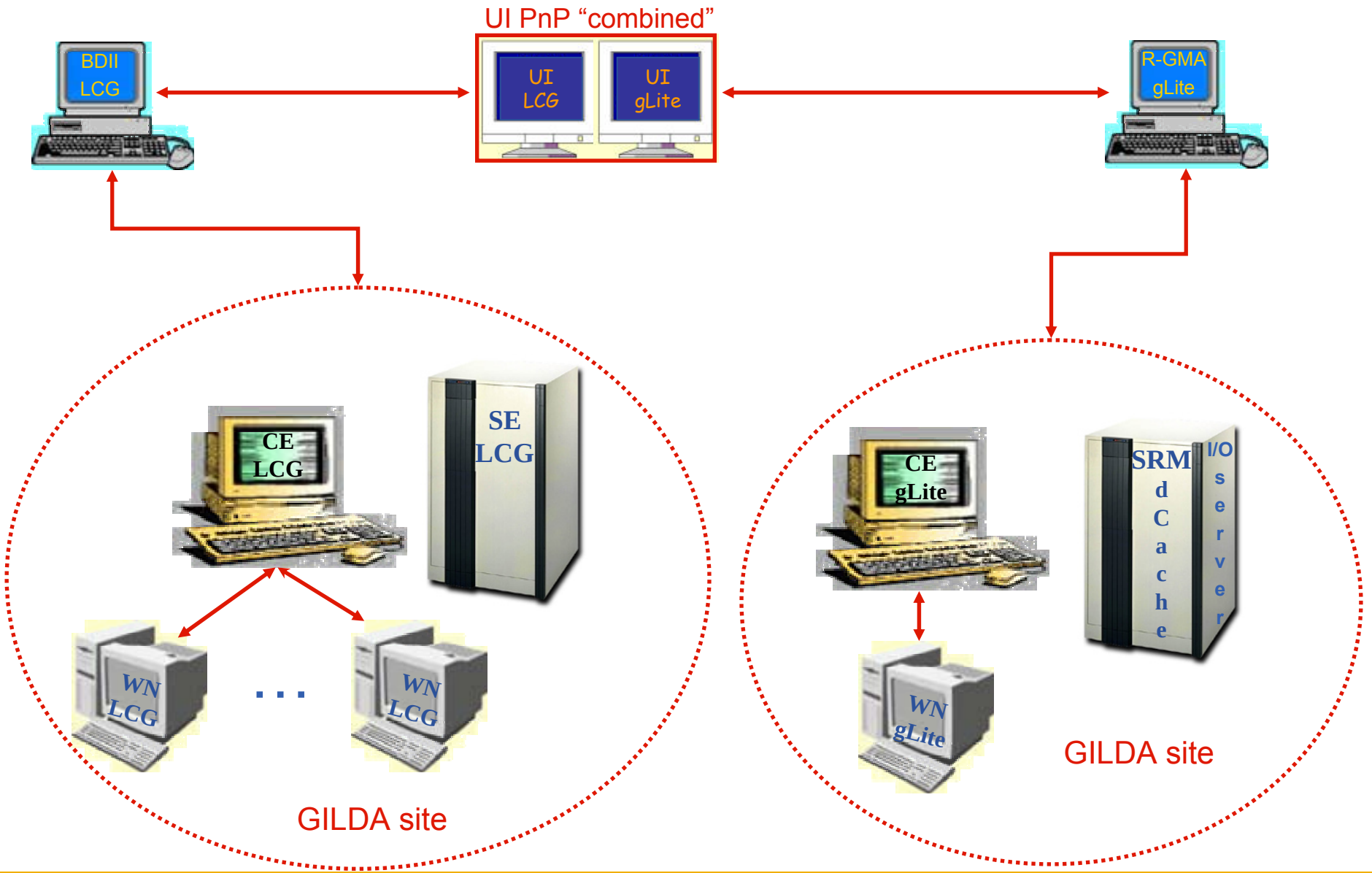
WMS layout in GILDA



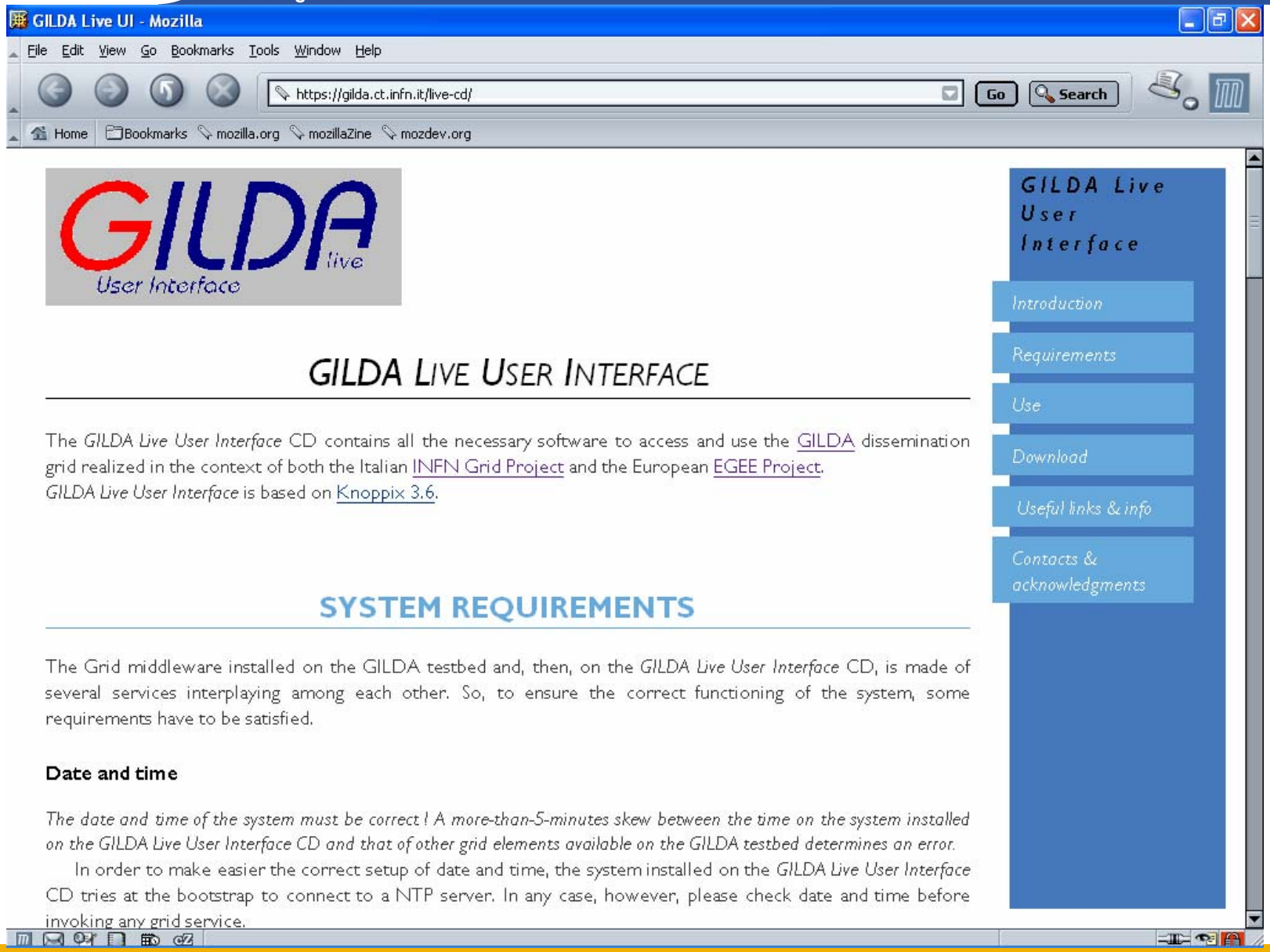
DMS layout in GILDA



IS layout in GILDA



The GILDA Live User Interface (1/2) (https://gilda.ct.infn.it/live-cd/)



The GILDA Tutorials/Demonstrations (1/2) (https://gilda.ct.infn.it/tutorials.html)

2004

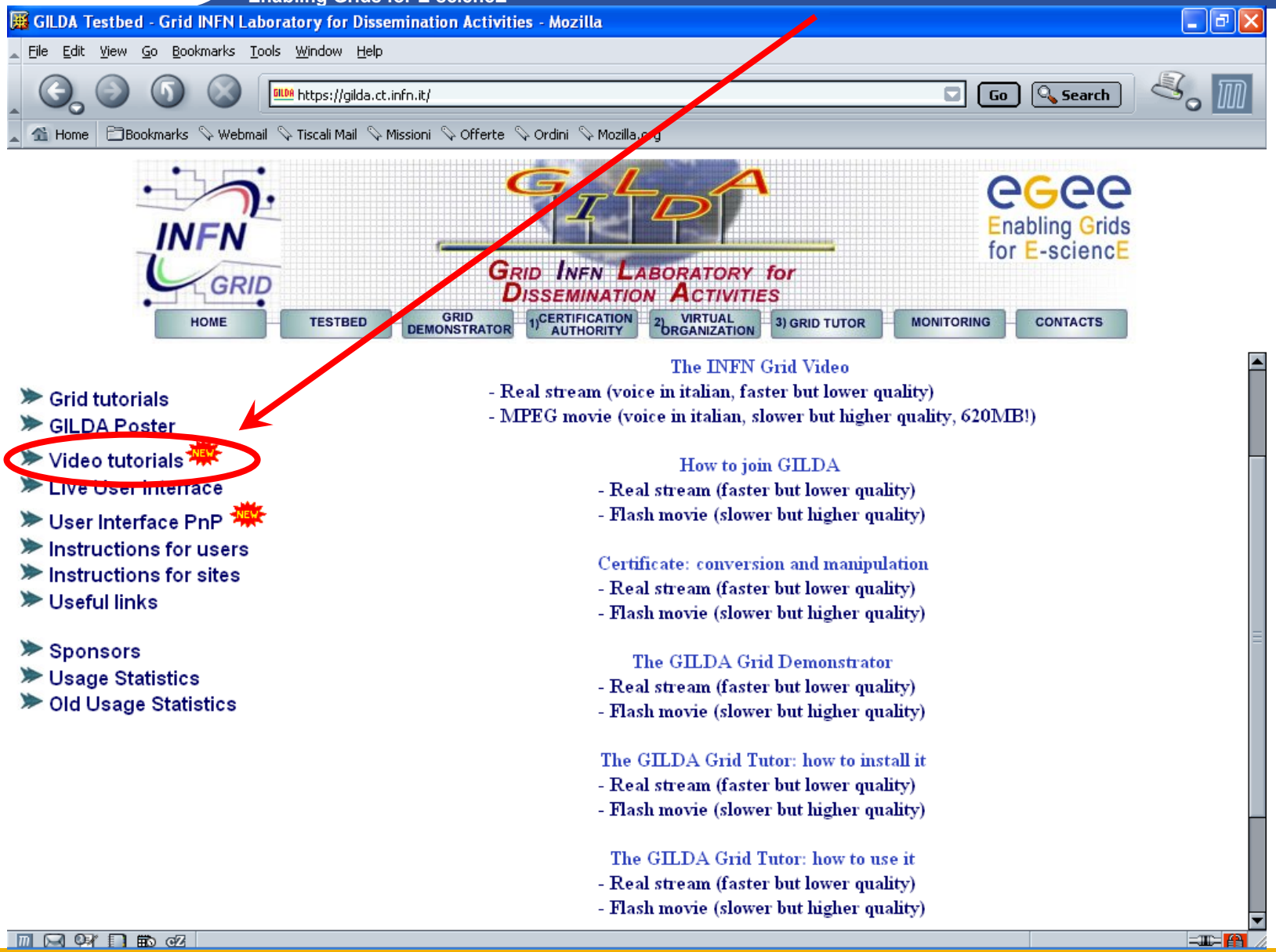
Edinburgh, 7 April 2004, slides, pictures
Tunis, 22-23 April 2004, pictures
Edinburgh, 26-28 April 2004, slides, pictures
CERN, 17-19 May 2004, pictures
Catania, 24-25 May 2004, home page, pictures
Dubna, 29 June - 2 July 2004, agenda
Edinburgh, 6 July 2004, home page
Catania, 14-16 July 2004, home page, pictures
Vico Equense, 19 July 2004, slides, pictures
Vico Equense, 6-10 September 2004, home page
Catania, 4-8 October 2004, home page, agenda
Vilnius, 5-6 October 2004, agenda
London, 6 October 2004
Madrid, 6-7 October 2004, agenda
Heidelberg, 11-14 October 2004
CERN, 16 October 2004
Prague, 26 october 2004, home page
Warsaw, 4-6 November 2004, home page, agenda
Lyon, 9-10 November 2004, agenda
The Hague, 15-17 November 2004, pictures
Merida, 15-20 November 2004, home page, agenda, slides, pictures
Tunis, 20 November 2004
Rio de Janeiro, 22-23 November 2004, home page, agenda, pictures
The Hague, 24 November 2004, agenda
CERN, 29-30 November 2004, agenda
Kosice, 30 November - 1 December 2004, agenda
Tunis, 6-7 December 2004
Bochum, 7-10 December 2004, home page, agenda
Edinburgh, 8 December 2004, home page
Istanbul, 9-10 December 2004, agenda, slides, pictures
Shanghai, 9-10 December 2004, agenda
Aurillac, 13-14 December 2004
Prague, 16 December 2004, home page, pictures
Tel Aviv, 22-23 December 2004, agenda, pictures

2005

CERN, 13 January 2005, agenda
Torino, 18-19 January 2005, home page, agenda
CERN, 20 January 2005, agenda
CERN, 2-4 February 2005, agenda
Roma, 3 February 2005, home page, agenda, pictures
Sydney, 3-4 February 2005, home page
CERN, 9-11 February 2005, agenda
Amsterdam, 14-16 February 2005, home page
Trento, 23-25 February 2005, home page, agenda
Amsterdam, 28 February - 1 March 2005, home page
Julich, 9 March 2005,
Clermont-Ferrand, 9-31 March 2005, agenda
Vienna, March-August 2005
Hamburg, 23-24 March 2005, home page, agenda
Ula-Merida, 31 March-1 April 2005, agenda
Zilina, 4 April 2005, home page and agenda
Edinburgh, 9-13 May 2005, home page and agenda
Catania, 13-15 June 2005, home page, agenda
Valencia, 14-16 June 2005, home page, agenda

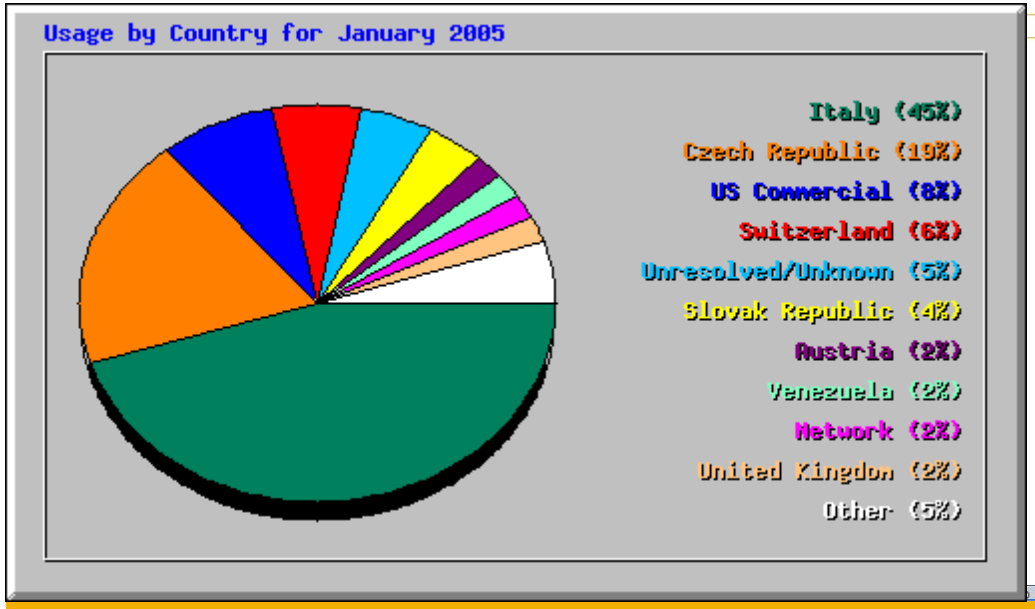
The GILDA Tutorials/Demonstrations (2/2) (https://gilda.ct.infn.it/tutorials.html)

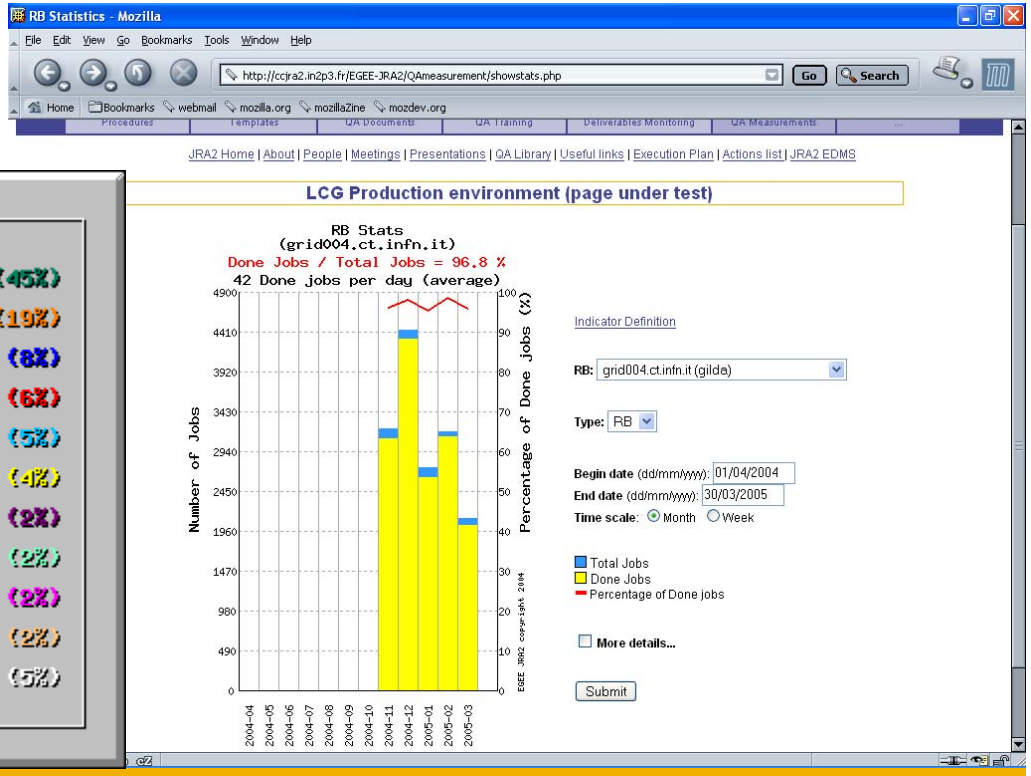
The GILDA Video Tutorials (https://gilda.ct.infn.it/video.html)



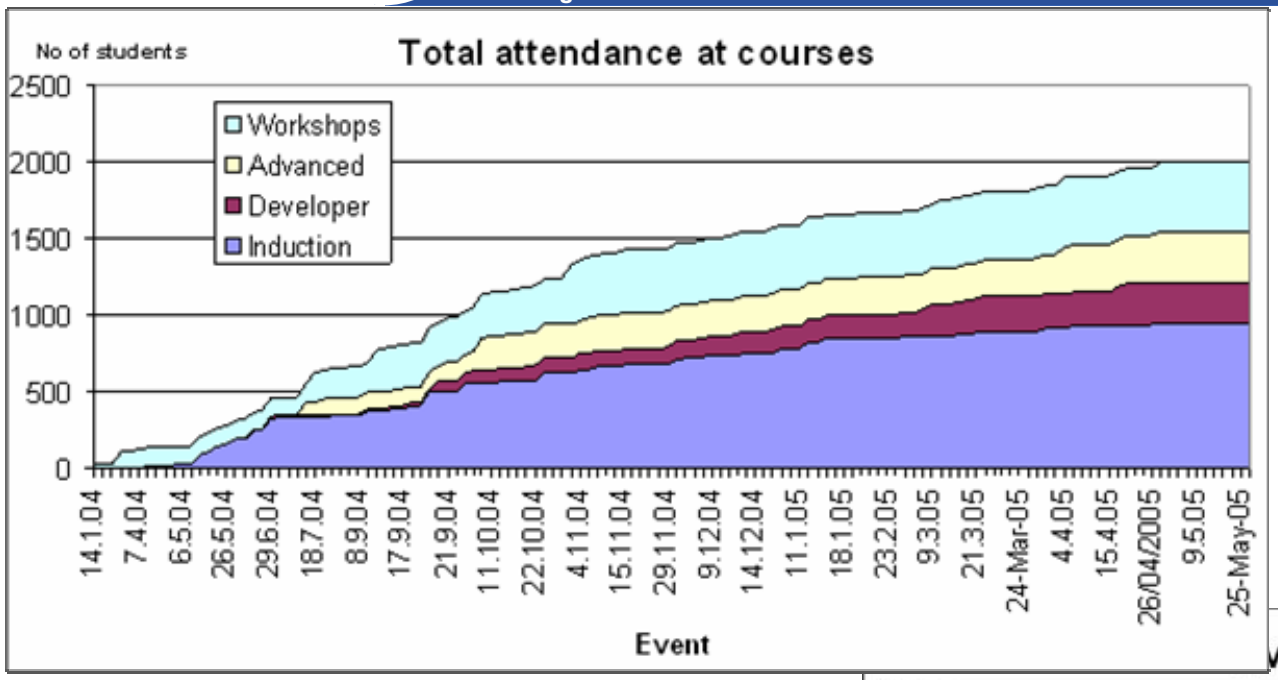
GILDA summary numbers

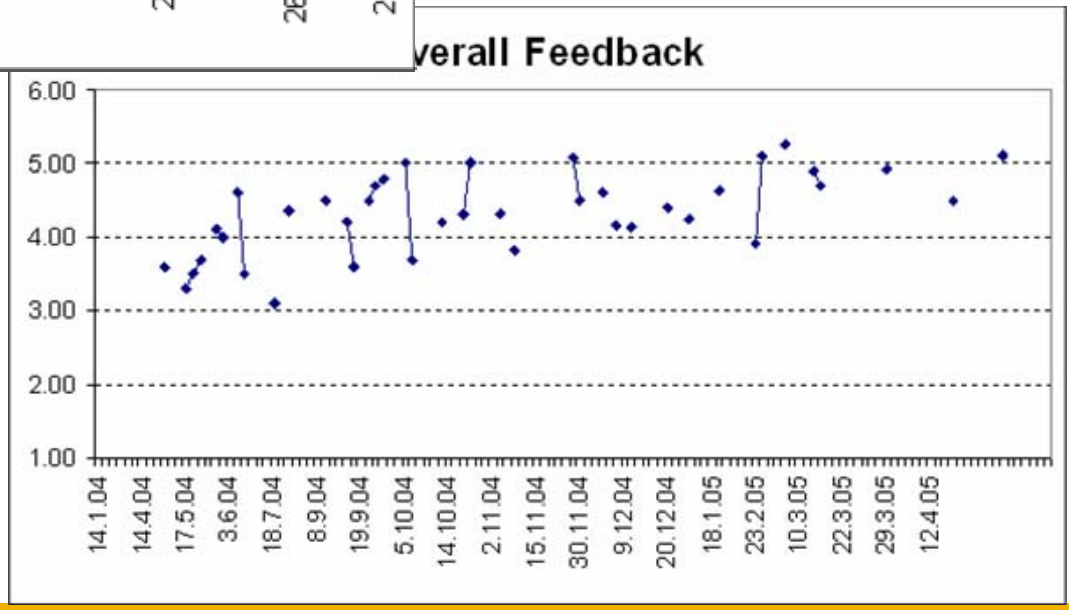
- 15 sites in 3 continents
- > 1600 certificates issued, 15% renewed at least once
- > 45 tutorials and demos performed in 15 months
- > 40 jobs/day on the average
- Job success rate above 80%
- > 600,000 hits (35,000 visits) on (of) the web site from 10's of different countries
- > 400 GB of videos and UI's downloaded from the web site





Training achievements





EGEE-NA4 Applications and GILDA

- **7 Virtual Organizations supported:**
  - Biomedicine (Biomed)
  - Earth Science Academy (ESR)
  - Earth Science Industry (CGG)
  - Astroparticle Physics (MAGIC)
  - Computational Chemistry (GEMS)
  - Grid Search Engines (GRACE)
  - Astrophysics (PLANCK)
- **Development of complete interfaces with GENIUS for 3 Biomed Applications: GATE, hadronTherapy, and Friction/Arlecore**
- **Development of complete interfaces with GENIUS for 4 Generic Applications: EGEODE (CGG), MAGIC, GEMS, and CODESA-3D (ESR) (successfull demos of EGEODE and GEMS at EGEE review)**
- **Development of complete interfaces with GENIUS for 16 demonstrative applications available on the GILDA Grid Demonstrator (https://grid-demo.ct.infn.it)**
- **Development of complete interface with CLI for NEMO**

Summary and conclusions

- **The EGEE middleware:**
  - Is exiting prototyping phase and entering real production phase (LHC first real data are only 2 years away from now!)
  - Implements a full and complete stack of grid services that can be used all together or separately at user's discretion
  - Closely follow the standardization process going in GGF and other for a
- **GILDA is a real virtual laboratory for dissemination of grid computing:**
  - It is a "de facto" standard t-Infrastructure adopted both by EGEE and some forthcoming EU-FP6 projects (EELA, EUCHINAGRID, EUMEDGRID, **ICEAGE**)
  - It is a complete suite of grid elements (test-bed, CA, VO, monitoring system, web portal, live user interface, user interface plug&play) and applications fully dedicated to dissemination purposes and pre-porting of new applications to EGEE Infrastructure
  - GILDA runs latest production (stable) version of both the LCG grid middleware but it is also early adopting gLite in order to make the transition to the new middleware smoother and easier