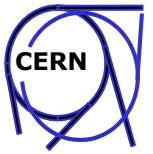


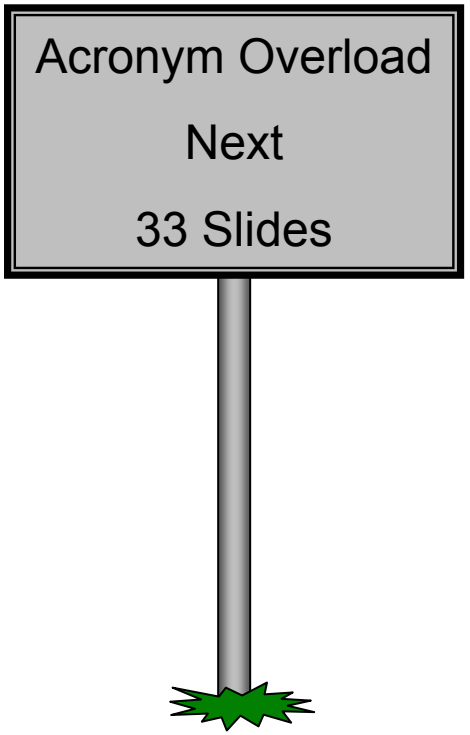
Introduction to Grid Computing

Markus Schulz
IT/GD

4 August 2004

Outline

- What are Grids (the vision thing)
  - What are the fundamental problems
  - Why for LHC computing?
- Bricks for Grids
  - Services that address the problems to build a grid
- How to use a GRID ?
- Where are we?
- What's next?



What is a GRID

- A genuine new concept in distributed computing
  - Could bring radical changes in the way people do computing
  - Named after the electrical power grid due to similarities
- A hype that many are willing to spend $$s on
  - many researchers/companies work on "grids"
    - More than 100 projects
    - Only few large scale deployments aimed at production
  - Very confusing (hundreds of projects named Grid something)
- Names to be googled: Ian Foster and Karl Kesselman
  - Or have a look at http://globus.org
    - not the only grid toolkit, but one of the first and most widely used
  - EGGE, LCG

Power GRID

- Power on demand
  - User is not aware of producers
- Simple Interface
  - Few types of sockets
- Standardized protocols
  - Voltage, Frequency
- Resilience
  - Re-routing
  - Redundancy
- Can't be stored, has to be consumed as produced
  - Use it or loose it
  - Pricing model
  - Advance reservation

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

QuickTime™ and a TIFF (Uncompressed) decompressor are needed to see this picture.

What is a GRID

- Basic concept is simple (I.Foster has a checklist, here a practical approach)
  - I.Foster: "coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations. "
  - Or:"On-demand, ubiquitous access to computing, data, and services"
  - From the user's perspective:
    - I want to be able to use computing resources as I need
    - I am today acting on behalf of organisation A, tomorrow B
    - A and B can be created today and deleted tomorrow
    - I don't care who owns resources, or where they are
    - I don't want to adapt my programs to different sites
    - I am willing to pay (could be done by the organization)
  - The owners of computing resources (cycles, storage, bandwidth)
    - My resources can be used by any authorized person (not for free)
    - Authorization is not tied to my administrative organization
  - Notice: The resources are **NOT** under **centralized control**, nor are the users
  - A Grid makes the coordination and negotiation between the two possible
- Most of the challenges that come with grids arise from this basic concept

Grids are not a magic bullets

Often cited in this context:

"When the network is as fast as the computer's internal links, the machine disintegrates across the net into a set of special purpose appliances"
(George Gilder)

- Imagine your CPU in CERN, your disks in Taipei, your memory at BNL
- For everything that requires low latency you are out of luck
  - But this is true for every wide area distributed system (Physics)

```
[lxplus094] ~ > ping adc0018.cern.ch
PING adc0018.cern.ch (137.138.225.48) from 137.138.4.103 : 56(84) bytes of data.
--- adc0018.cern.ch ping statistics ---
11 packets transmitted, 11 received, 0% loss, time 10099ms
rtt min/avg/max/mdev = 0.204/0.405/1.332/0.332 ms
[lxplus094] ~ > ping lcg00105.grid.sinica.edu.tw
PING lcg00105.grid.sinica.edu.tw (140.109.98.135) from 137.138.4.103 : 56(84) bytes of data.
--- lcg00105.grid.sinica.edu.tw ping statistics ---
10 packets transmitted, 10 received, 0% loss, time 9086ms
rtt min/avg/max/mdev = 301.246/301.342/301.837/0.714 ms
```

O,4ms

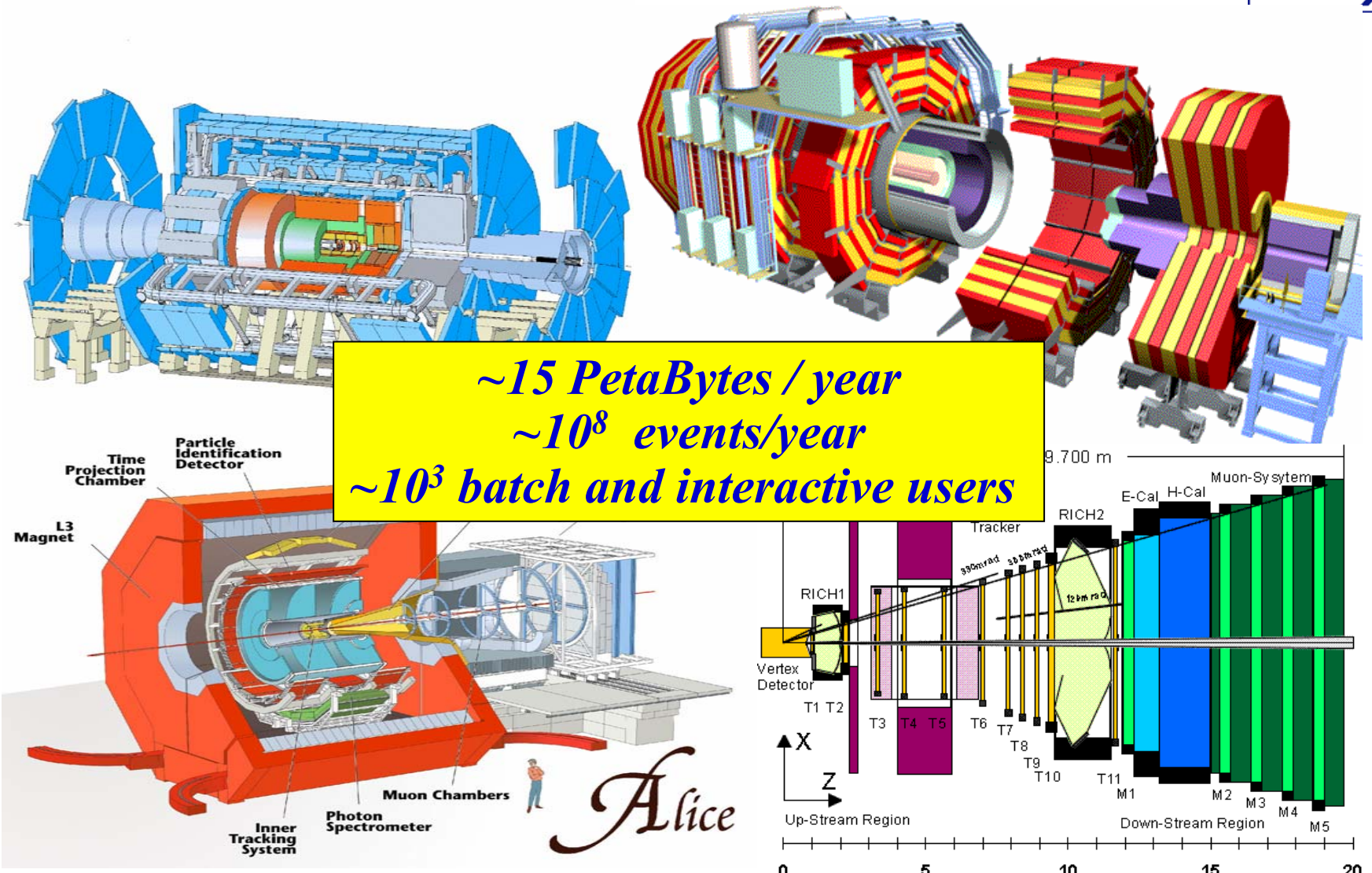
301ms

Problems

- With no central control and distributed heterogeneous resources
  - How do I find resources with given caracteristiques ?
  - How can my programs access resources (like files)?
  - How can I trust resources?
  - How can I be trusted by the resources?
  - How can I know how to use the resources?
  - How can I cope with resurces failing?
  - How can I charge for using my resources?
  - How can I handle security incidents if there is no central something?
  - How to share resources between local users and grid users?
- Distributed computing in the past avoided these problems
  - Keeping everything under central control
  - Giving up on accounting
  - Extreme split of problem into micro tasks, "statistical" approach to quality (seti@home etc)

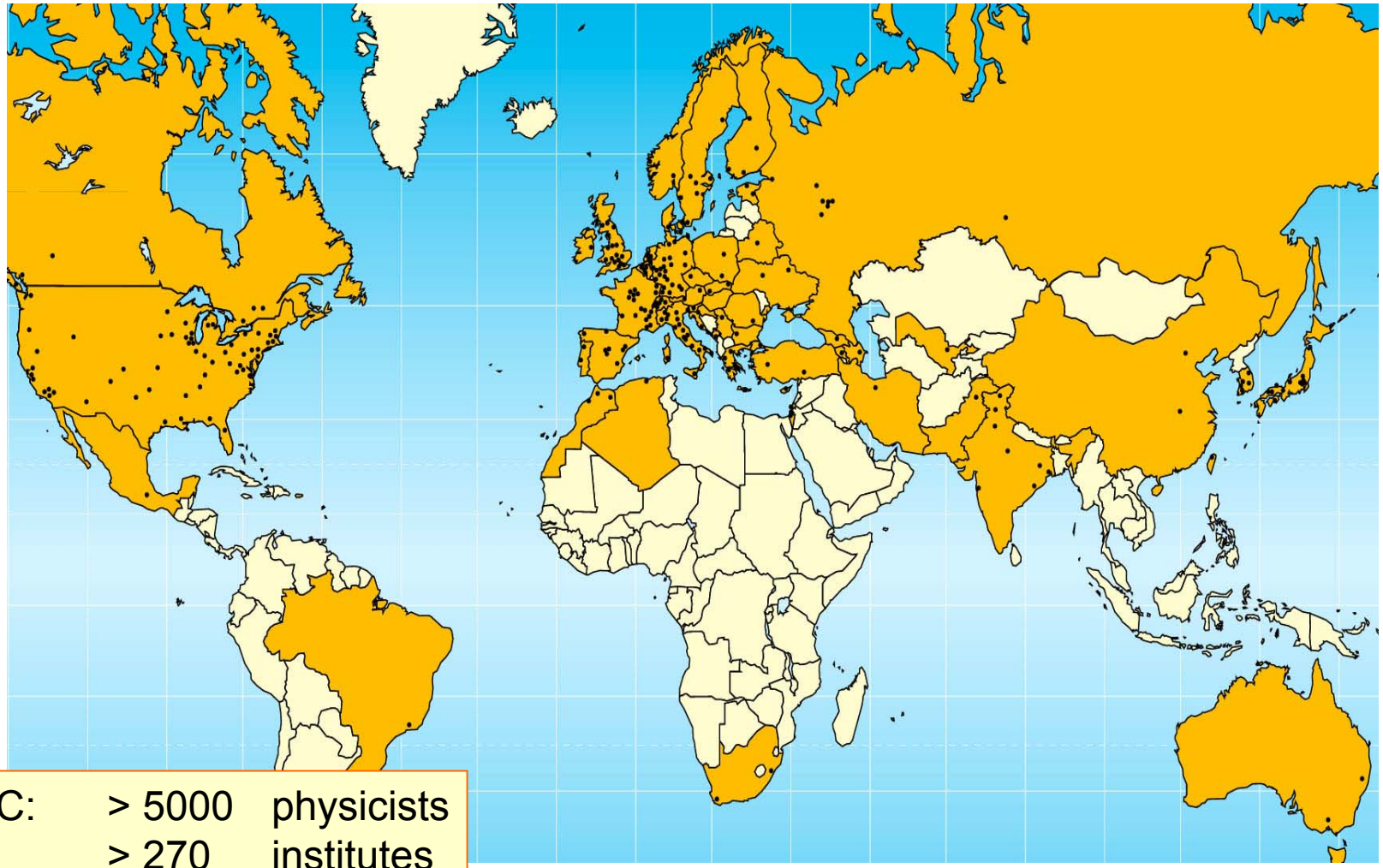
Grid computing is to give efficient, workable answers to these questions

Why Grids for LHC NOW?



Federico.carminati , EU review presentation

User Community



LHC:	> 5000	physicists
	> 270	institutes
	> 60	countries

Why for LHC Computing?

- Motivated:
  - O(100k) boxes needed + gigantic mass storage
  - Many reasons why we can't get them in one place
    - funding, politics, technical..
  - Need to ramp up computing soon (MC production)
- What helps:
  - Problem domain quite well understood
    - Have already experience with distributed MC production and reconstruction
    - Embarrassing parallel (a huge batch system would do)
  - Community established
    - trust can be build more easily
  - Need only part of the problems solved
    - Communities not too dynamic (an experiment will stay for 15years)
    - Well structured jobs

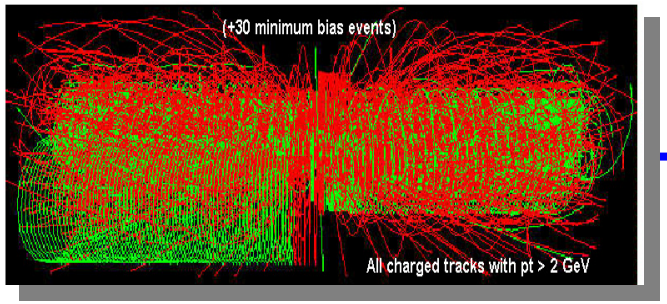
HEP Jobs

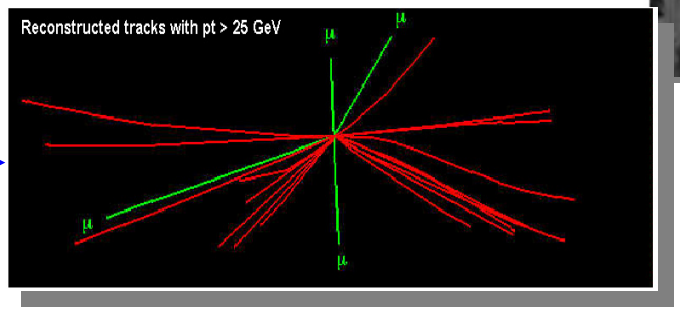
- **Reconstruction**: transform signals from the detector to physical properties
  - energy, charge, tracks, momentum, particle id.
  - this task is computational intensive and has modest I/O requirements

All operate on an event by event basis

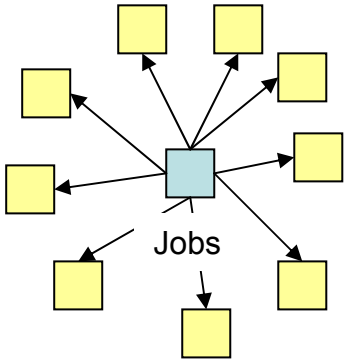
embarrassingly parallel

  - structured activity, but larger number of parallel activities
- **Analysis**: complex algorithms, search for similar structures to extract physics
  - very I/O intensive, large number of files involved
  - access to data cannot be effectively coordinated
  - iterative, parallel activities of hundreds of physicists

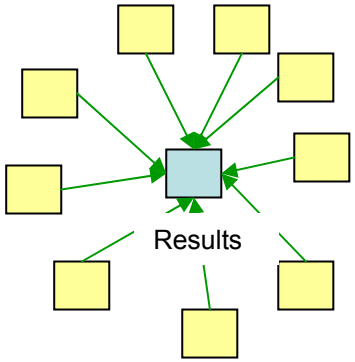




High Energy Physics (past)



- Experience with distributed computing since almost 20years
  - Solution then: The invention of the "**production manager**"
    - Accounts created on all sites that offer resources
    - Agreed quotas on the sites (negotiation in the Collaboration Board)
    - Accounts very often shared by a set of very experienced managers
    - Knowledge about resources represented in set of scripts and file catalogues
    - Minimalist's approach to programming environment:
      - F77, IO-Libs, CERN libs, almost no external SW
    - Data often shipped offline (tapes, etc.)
  - Worked remarkably well for HEP
    - thanks to generations of Phd students and freaks working long hours
  - But:
    - Negotiation process problematic
    - Adding smaller resources often not worth it
    - Changes to a site had to be communicated (by e-mail, confusion)
    - Sometimes the trust in the quality of the sites was not 100%
      - Reprocessing at CERN before publishing was quite common in LEP days
    - Mostly used from a central point, limited number of sites, all well known

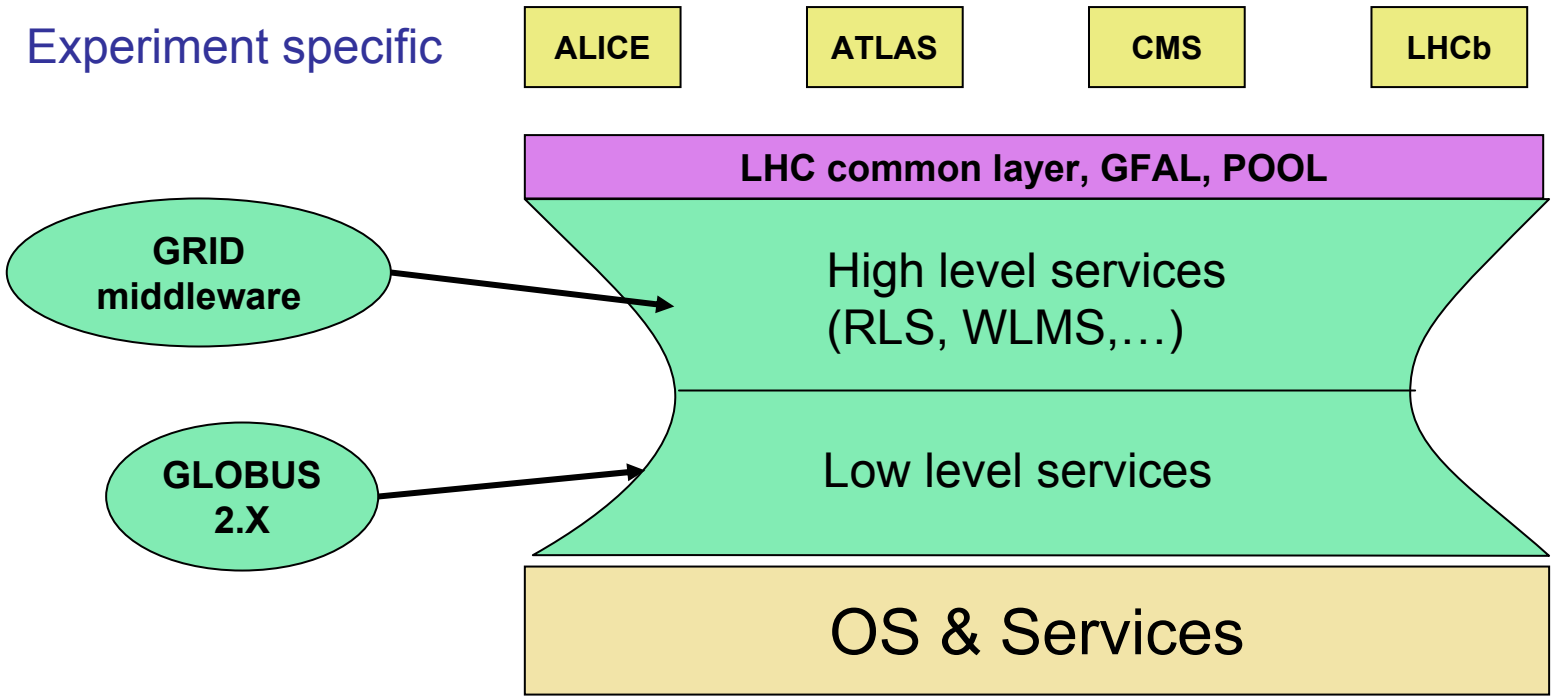


Bricks for Grids: Problems

- With no central control and distributed heterogeneous resources
  - How do I find resources with given caracteristiques ?
  - How can my programs access resources (like files)?
  - How can I trust resources?
  - How can I be trusted by the resources?
  - How can I know how to use the resources?
  - How can I cope with resurces failing?
  - How can I charge for using my resources?
  - How can I handle security incidents if there is no central something?
  - How to share resources between local users and grid users?
- Needed:
  - Trust between users and sites (without central control)
  - Manage diversity (Programs, OS, MSS, Batch Systems, ..)
  - Access to information about resources, their availability and state
  - System to keep track where user data is and to bring jobs and data together
  - Resilience against failure
- Described solutions are close to LCG2 (in operation)
- Note: Architecture for EGEE will be in some aspects different

Diversity

- "Hourglass" model
  - standardized protocols (not application specific ones)
  - abstraction layer between GRID visible services, OS and local services



Trust

- **GSI**: Globus Security Infrastructure
  - Provides authentication and authorization
  - Based on **PKI** X509 (Public Key Infrastructure)
- Authentication
  - Tells only that you are you (ID Card)
- Authorization
  - Determines what you can do (Credit Card (has reference to ID))
- At the core a set of Certification Authorities (**CA**) (few)
  - Issue X509 certificates for **users** and **service** authentication (valid 1year)
    - Extensions can carry authorisation/restriction information
  - Revoces certificates if key is compromised
  - Publish formal document about how they operate the service
    - Certificate Policy and Certification Practice statement (**CP/ CPS**)
    - Mechanisms to issue and revoke certificates
  - Site decides on which CAs are accepted based on CP/CPS (and org.)
    - Maintains a list of root certs, updates CRLs

What is a passport?

X509

- X509 user certificate (for details see RFC X509, (open)SSL, GSI)
  - User certificate + private key create a proxy certificate
    - has limited life time (hours), travels with jobs (delegated credential)
    - mechanism for long term jobs (MyProxy Server)

```
Certificate:
   Data:
      Version: 3 (0x2)
      Serial Number: 37 (0x25)
      Signature Algorithm: md5WithRSAEncryption
      Issuer: C=CH, O=CERN, OU=cern.ch, CN=CERN CA
      Validity
         Not Before: Sep 16 16:02:01 2002 GMT
         Not After : Nov 30 12:00:00 2003 GMT
      Subject: O=Grid, O=CERN, OU=cern.ch, CN=Markus
Schulz
      Subject Public Key Info:
         Public Key Algorithm: rsaEncryption
         RSA Public Key: (1024 bit)
            Modulus (1024 bit):
               00:9d:e5:3b:e7:ce:31:a6:b6:1b:c0:f3:ed:ce:14:
               2e:86:ab:66:5c:f2:2e:9b:41:e9:9a:7b:1b:b2:9a:
               73:2f:3f:09:63:f5:bc:b7:07:9c:87:5d:a4:0b:fb:
 ======cut=
```

Part 2 contains extensions

```
Exponent: 65537 (0x10001)
      X509v3 extensions:
         Netscape Base Url:
            http://home.cern.ch/globus/ca
         Netscape Cert Type:
            SSL Client, S/MIME
         Netscape Comment:
            For DataGrid use only
         Netscape Revocation Url:

http://home.cern.ch/globus/ca/bc870044
.r0
         Netscape CA Policy Url:
=========
```

User has to keep key save!!
User has to renew cert.!!

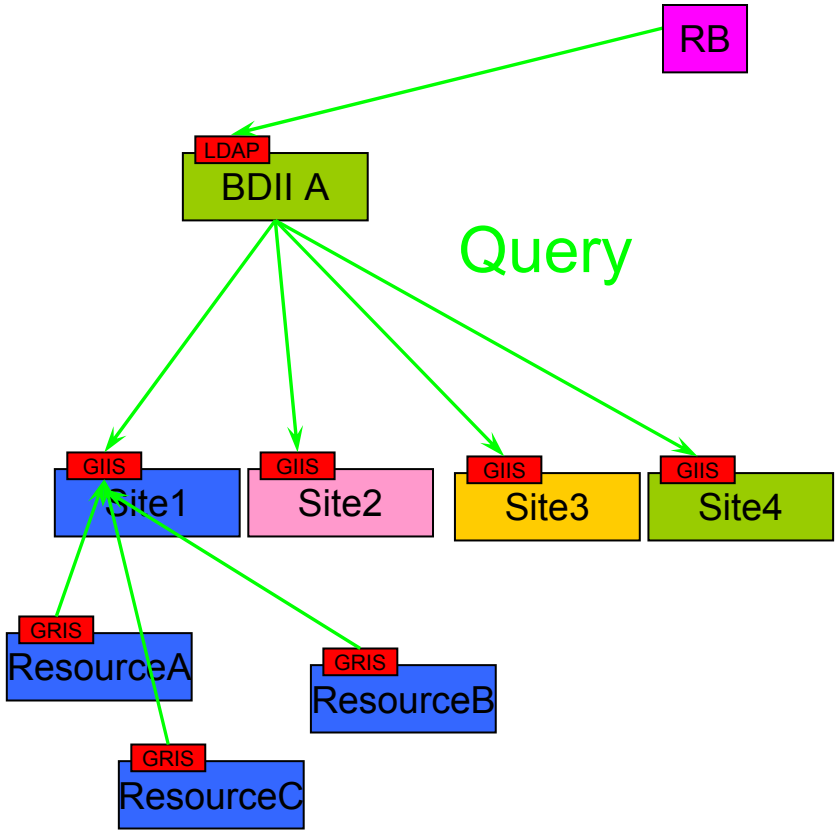
Authorization

- Virtual Organizations (VO)
  - In LCG there are two groups of VOs
    - The experiments (Atlas, Alice, CMS, LHCb, dTeam)
    - LCG1 with everyone who signed the Usage Rules (http://lcg-registrar.cern.ch)
  - Technical this is a LDAP server
    - publishes the subjects of members
  - By registering with the LCG1 VO and an experiment VO the user gets authorized to access resources that the VO is entitled to use
  - In the real world the org. behind a VO is responsible for the resources used
- Site providing resources
  - Decides on VOs that are supported (needs negotiation offline)
  - Can block individuals from using resources
  - The grid mapfile mechanism maps Subjects according to VO to pool of local accounts.
    - O=Grid,O=CERN,OU=cern.ch,CN=Markus Schulz .dteam
    - Jobs will run under dteam0002 account on this site
- Soon major change: VOMS will allow to express roles inside VO
  - Production manager, simple user, etc.

Information System

- Monitoring and Discovery Service (**MDS**)
  - Information about resources, status, access for users, etc.
  - There are static and dynamic components
    - Name of site (static), Free CPUs (dynamic), Used Storage (dynamic)….
  - Access through LDAP
  - Schema used defined by **GLUE** (Grid Laboratory for a Uniform Environment)
    - Every user can query the MDS system via ldapsearch or a ldap browser
- Hierarchical System with improvements by LCG
  - Resources publish their static and dynamic information via the **GRIS**
    - Grid Resource Information Servers
  - GRISs register on each site with the site's **GIIS**
    - Grid Index Information Server
  - On top of the tree sits the **BDII** (Berkeley DB Information Index)
    - Queries the GIISes, fault tolerant
    - Acts as a kind of cache for the information
- If you move up the tree the more stale the information gets (2minutes)
  - For submitting jobs, the BDIIs are queried

LCG2 Information System

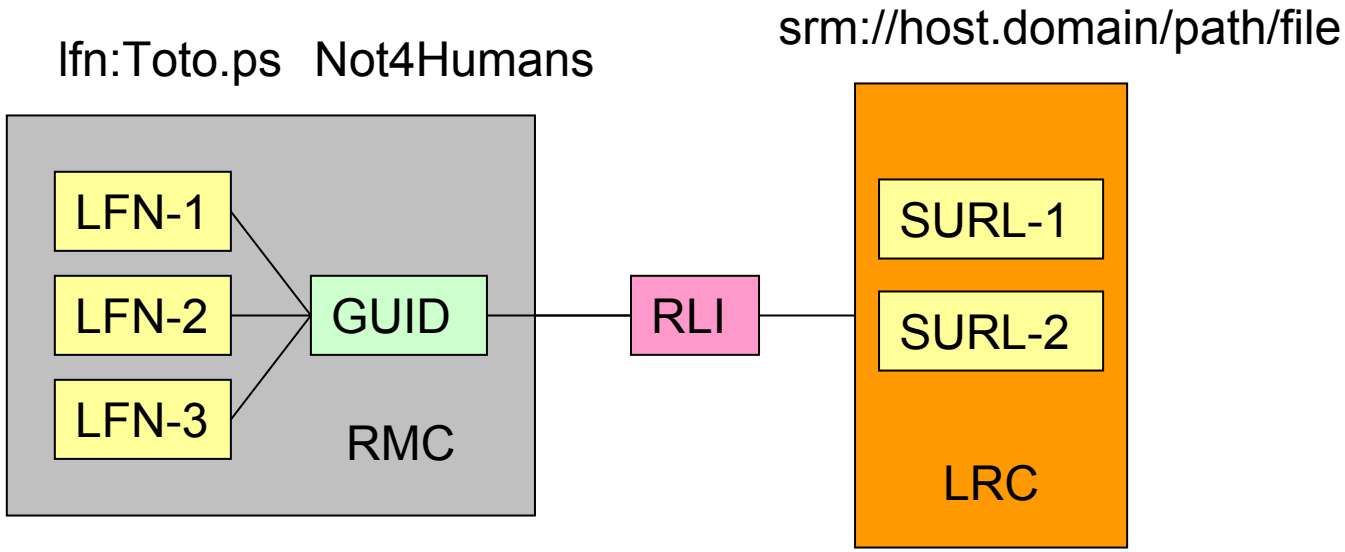


Moving Data, Finding Data

- GridFTP (very basic tool)
  - Version of parallel ftp with GSI security (gsiftp) used for transport
- Interface to storage system via **SRM** (Storage Resource Manager)
  - Handles things like migrating data, to from MSS, file pinning, etc.
  - Abstract interface to storage subsystems http://sdm.lbl.gov/indexproj.php?ProjectID=SRM
- EDG-**RLS** (Replica Location Service) Keeps track of where files are
  - Composed of two catalogues
    - Replica Metadata Catalog RMC and Local Replica Catalog (LRC)
    - Provides mappings between logical file names and locations of the data (SURL)
    - Design distributed, currently one instance/VO at CERN
  - http://hep-proj-grid-tutorials.web.cern.ch/hep-proj-grid-tutorials/doc/edg-rls-userguide.pdf
- EDG-**RM**, (Replica Manager) moving files, creating replications
  - Uses basic tools and RLS
  - http://hep-proj-grid-tutorials.web.cern.ch/hep-proj-grid-tutorials/doc/edg-replica-manager-userguide.pdf
- Transparent access to files by user via **GFAL** (GRID File Access Lib)
  - http://lcg.web.cern.ch/LCG/peb/GTA/LCG_GTA_ES.htm

RLS

1. User assigns **L**ogical **F**ile **N**ame and aliases to file (toto.ps)
2. RMC stores relation between LFNs and GUID (global unique ID)
3. RLI (Replica Location Index) knows about LRCs
4. LRCs know mapping between GUIDs and name needed by SRM
5. SRM does then knows how to handle the storage system



Work Load Management System

- The services that brings resources and the jobs together
  - Runs on a node called **RB** (Resource Broker)
  - Keeps track of the status of jobs (**LBS** Logging and Bookkeeping Service)
  - Talks to the globus gate keepers and resource managers on the remote sites (LRMS) (CE)
  - Matches jobs with sites where data and resources are available
  - Re-submission if jobs fail
- Uses almost all services: IS, RLS, GSI, ..
  - Walking trough a job might be instructive (see next slide)
- The user describes the job and its requirements using JDL (Job Description Lang.)

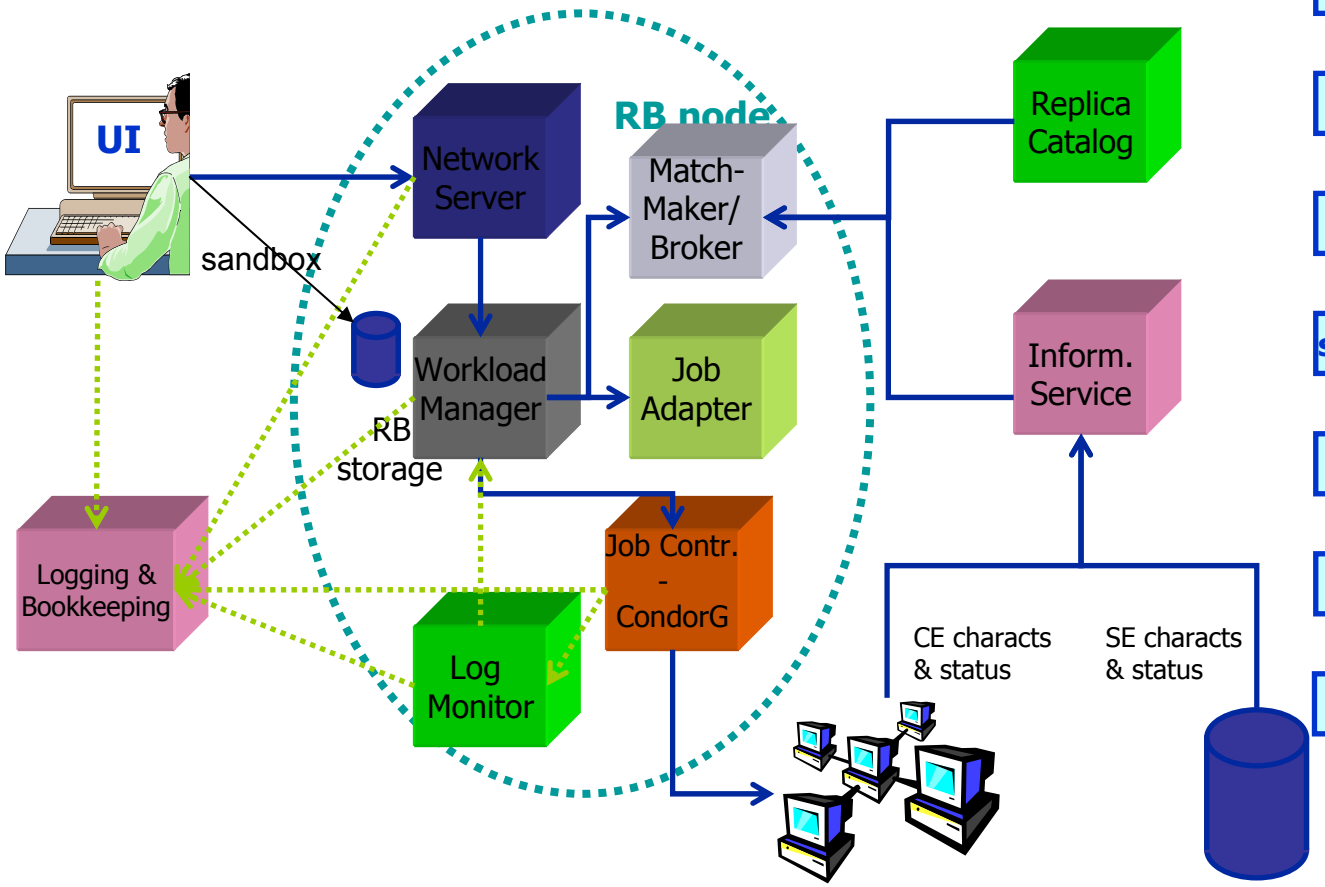
RB

```
[
JobType="Normal";
Executable = "gridTest";
StdError = "stderr.log";
StdOutput = "stdout.log";
InputSandbox = {"home/joda/test/gridTest"};
OutputSandbox = {"stderr.log", "stdout.log"};
InputData = {"lfn:green", "guid:red"};
DataAccessProtocol = "gridftp";
Requirements = other.GlueHostOperatingSystemNameOpSys == "LINUX"
            && other.GlueCEStateFreeCPUs>=4;
Rank = other.GlueCEPolicyMaxCPUTime;
]
```

http://www.infn.it/workload-grid Docs for WLMS

Work Load Management System

Input Sandbox is what you take with you to the node
Output Sandbox is what you get back

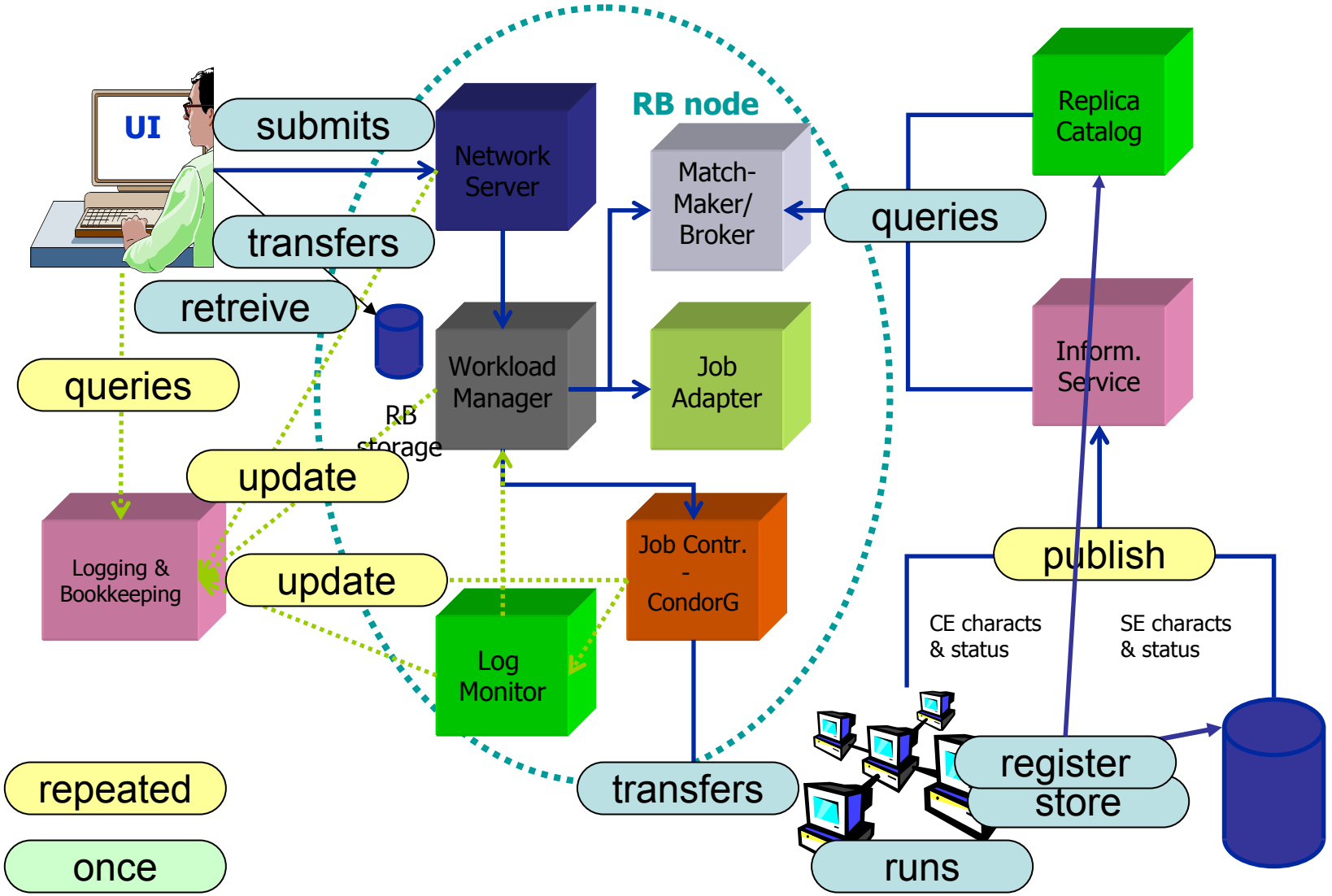


Job Status

- submitted — Arrived on RB
- waiting — Matching
- ready — Job Adapter
- scheduled — On CE
- running — Processed
- done — Output back
- cleared — User done

Failed jobs are resubmitted

Work Load Management System



Grouping the Bricks

UI

User Interface
Client libs. & commands
Used by User

RB

WLMS
Services
GridFTP

Proxy

MyProxy server

BDII

Top of the IIS

Larger Sites

CE

Gatekeeper,
LRMS
Gateway to Ces
Site GIIS, GRIS

WNs

LRM client
Client libs.
Client commands

SE

GridFTP
GRIS
SRM
MSS

Every Site

RMC

RLS

LRC

VO

1/VO

CA

1/Region

Where are we now

- Initial set of tools to build a production grid are there
  - Many details missing, core functionality there
  - Stability not perfect, but improving
  - Operation, Accounting, Audit, need a lot of work
  - Some of the services not redundant (One RLS/VO)
- Grid computing dominated by de facto standards (== no standards)
  - Need standards to swap components
  - Interoperability issues
- First production experience with Experiments
  - Lots of feedback
- Need to simplify usage
  - Move closer to the vision
- Need to be less dependent on system config.
  - EDG/Globus have complex dependencies
- Integrated significant number of sites (> 70, >6000CPUs)
  - tested scalability issues (and addressed several)
  - build a community
  - learned to "grid enable" local resources (computing and MSS)
  - have seen 100ks jobs on the grid/week

How complicate is it to use LCG?

- A few simple steps:
- Get a certificate
- Sign the "Usage Rules"
- Register with a VO
- Initialize /Register the proxy
- Write the JDL (copy modify)
- Submit the job
- Check the status
- Retrieve the output
- Move data around, check the information system etc.
- Next slides show some frequently used commands

The Basics

- **Get the LCG-2 Users Guide**
- http://grid-deployment.web.cern.ch/grid-deployment/cgi-bin/index.cgi?var=eis/homepage
- **Get a certificate**
  - Go to the CA that is responsible for you and request a user certificate
    - List of CAs can be found here
      - http://lcg-registrar.cern.ch/pki_certificates.html
  - Follow instructions on how to load the certificate into an web-browser
  - Do this.
  - Register with LCG and a VO of your choice: http://lcg-registrar.cern.ch/
  - In case your cert is not in PEM format change it to it by using openssl
    - Ask your CA how to do this
  - Find a user interface machine (adc0014 at CERN)
  - GOC page http://goc.grid-support.ac.uk/gridsite/gocmain/

Get ready

- Generate a proxy (valid for 12h)
  - $ grid-proxy-init (will ask for your pass phrase)
  - $ grid-proxy-info (to see details, like how many hours until t.o.d.)
  - $ grid-proxy-destroy
- For long jobs register long term credential with proxy server
  - $ myproxy-init -s lxn1788.cern.ch -d -n Creates proxy with one week duration

Job Submission

- Basic command: **edg-job-submit --vo <VO> test.jdl**
  - Many, many options, see WLMS manual for details
  - Try -help option (very useful -o to get job id in a file)
  - Tiny JDL file

```
executable = "testJob.sh";
StdOutput = "testJob.out";
StdError = "testJob.err";
InputSandbox = {"./testJob.sh"};
OutputSandbox = {"testJob.out","testJob.err"};
```

```
Connecting to host lxshare0380.cern.ch, port 7772
Logging to host lxshare0380.cern.ch, port 9002

================================ edg-job-submit Success ====================================
 The job has been successfully submitted to the Network Server.
 Use edg-job-status command to check job current status. Your job identifier (edg_jobId) is:

 - https://lxshare0380.cern.ch:9000/1GmdXNfZeD1o0B9bjFC3Lw

 The edg_jobId has been saved in the following file:
 /afs/cern.ch/user/m/markusw/TEST/DEMO/OUT
============================================================================================
```

http://www.infn.it/workload-grid Docs for WLMS

Where to Run?

- Before submitting a job you might want to see where you can run
  - edg-job-list-match --vo <VO> <jdl>
- Switching RBs
  - Use the --config-vo < vo conf file>
  - and --config <conf file>
  - (see User Guide)
  - Find out which RBs you could use

```
Connecting to host lxshare0380.cern.ch, port 7772

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

                *CEId*
adc0015.cern.ch:2119/jobmanager-lcgpbs-infinite
adc0015.cern.ch:2119/jobmanager-lcgpbs-long
adc0015.cern.ch:2119/jobmanager-lcgpbs-short
adc0018.cern.ch:2119/jobmanager-pbs-infinite
adc0018.cern.ch:2119/jobmanager-pbs-long
adc0018.cern.ch:2119/jobmanager-pbs-short
dgce0.icepp.s.u-tokyo.ac.jp:2119/jobmanager-lcgpbs-infinite
dgce0.icepp.s.u-tokyo.ac.jp:2119/jobmanager-lcgpbs-long
dgce0.icepp.s.u-tokyo.ac.jp:2119/jobmanager-lcgpbs-short
grid-w1.ifae.es:2119/jobmanager-lcgpbs-infinite
grid-w1.ifae.es:2119/jobmanager-lcgpbs-long
grid-w1.ifae.es:2119/jobmanager-lcgpbs-short
hik-lcg-ce.fzk.de:2119/jobmanager-lcgpbs-infinite
hik-lcg-ce.fzk.de:2119/jobmanager-lcgpbs-long
hik-lcg-ce.fzk.de:2119/jobmanager-lcgpbs-short
hotdog46.fnal.gov:2119/jobmanager-pbs-infinite
hotdog46.fnal.gov:2119/jobmanager-pbs-long
hotdog46.fnal.gov:2119/jobmanager-pbs-short
lcg00105.grid.sinica.edu.tw:2119/jobmanager-lcgpbs-infinite
lcg00105.grid.sinica.edu.tw:2119/jobmanager-lcgpbs-long
lcg00105.grid.sinica.edu.tw:2119/jobmanager-lcgpbs-short
lcgce01.gridpp.rl.ac.uk:2119/jobmanager-lcgpbs-infinite
lcgce01.gridpp.rl.ac.uk:2119/jobmanager-lcgpbs-long
lcgce01.gridpp.rl.ac.uk:2119/jobmanager-lcgpbs-short
lhc01.sinp.msu.ru:2119/jobmanager-lcgpbs-infinite
lhc01.sinp.msu.ru:2119/jobmanager-lcgpbs-long
lhc01.sinp.msu.ru:2119/jobmanager-lcgpbs-short
wn-02-29-a.cr.cnaf.infn.it:2119/jobmanager-lcgpbs-infinite
wn-02-29-a.cr.cnaf.infn.it:2119/jobmanager-lcgpbs-long
wn-02-29-a.cr.cnaf.infn.it:2119/jobmanager-lcgpbs-short
zeus02.cyf-kr.edu.pl:2119/jobmanager-lcgpbs-infinite
zeus02.cyf-kr.edu.pl:2119/jobmanager-lcgpbs-long
zeus02.cyf-kr.edu.pl:2119/jobmanager-lcgpbs-short
**********************************************************************
```

And then?

- Check the status:
  - edg-job-status -v <0|1|2> -o <file with id>
  - Many options, play with it, do a -help --noint for working with scripts
- In case of problems:
  - edg-job-get-logging-info (shows a lot of information) controlled by -v option
- Get output sandbox:
  - edg-job-get-output, options do work on collections of jobs
    - Output in /tmp/jobOutput/1GmdXNfZeD1o0B9bjFC3Lw
- Remove the job
  - edg-job-cancel
    - Getting the output cancels the job, canceling a canceled job is an error

Information System

- Query the BDII (use an ldap browser, or ldapsearch command)
  - Sample: BDII at CERN lxn1189.cern.ch
  - Have a look at the man pages and explore the BDII, CE and SE

BDII

```
ldapsearch -LLL -x -H ldap://lxn1189.cern.ch:2170 -b "mds-vo-name=local,o=grid" "(objectClass=glueCE)" dn
```

CE

```
ldapsearch -LLL -x -H ldap://lxn1184.cern.ch:2135 -b "mds-vo-name=local,o=grid"
```

SE

```
ldapsearch -LLL -x -H ldap://lxn1183.cern.ch:2135 -b "mds-vo-name=local,o=grid"
```

More comfortable:
http://goc.grid.sinica.edu.tw/gstat/

Data

- The edg-replica-manager and lcg tools allow to: edg-rm
  - move files around UI->SE WN->SE,
  - Register files in the RLS
  - Replicate them between SEs
  - Locate replicas
  - Delete replicas
  - get information about storage and it's access
  - Many options -help + documentation
- Move a file from UI to SE
  - Where? edg-rm --vo=dteam printInfo
  - edg-rm --vo=dteam copyAndRegisterFile file:`pwd`/load -d srm://adc0021.cern.ch/flatfiles/SE00/dteam/markus/t1 -l lfn:markus1
    - guid:dc9760d7-f36a-11d7-864b-925f9e8966fe is returned
    - Hostname is sufficient for -d (without the RM decides where to go)

Next Generation (EGEE)

- First generation of grid toolkits suffered from lack of standardization
  - Build an open systems
  - OGSA (Open Grid Service Architecture)
  - Use Web services for "RPC"
  - Standard components from the Web world
    - SOAP (*Simple Object Access Protocol*) to convey messages (XML payloads)
    - WSDL (*Web Service Description Language*) to describe interface
  - Rigorous standards -> different implementations can coexists (competition)
  - Start with wrapping existing tools
- Can be "hosted" in different environments
  - Standalone container, TomCat, IBM Websphere
  - .NET
- **Big leap forward**
  - If the big players do not dilute the standards as they did with the WWW