The European Data Grid

Getting Real Science Applications on a Production Grid

David Groep, NIKHEF
davidg@nikhef.nl

http://www.dutchgrid.nl/

http://www.eu-egee.org/
http://www.edg.org/
http://cern.ch/lcg

Outline

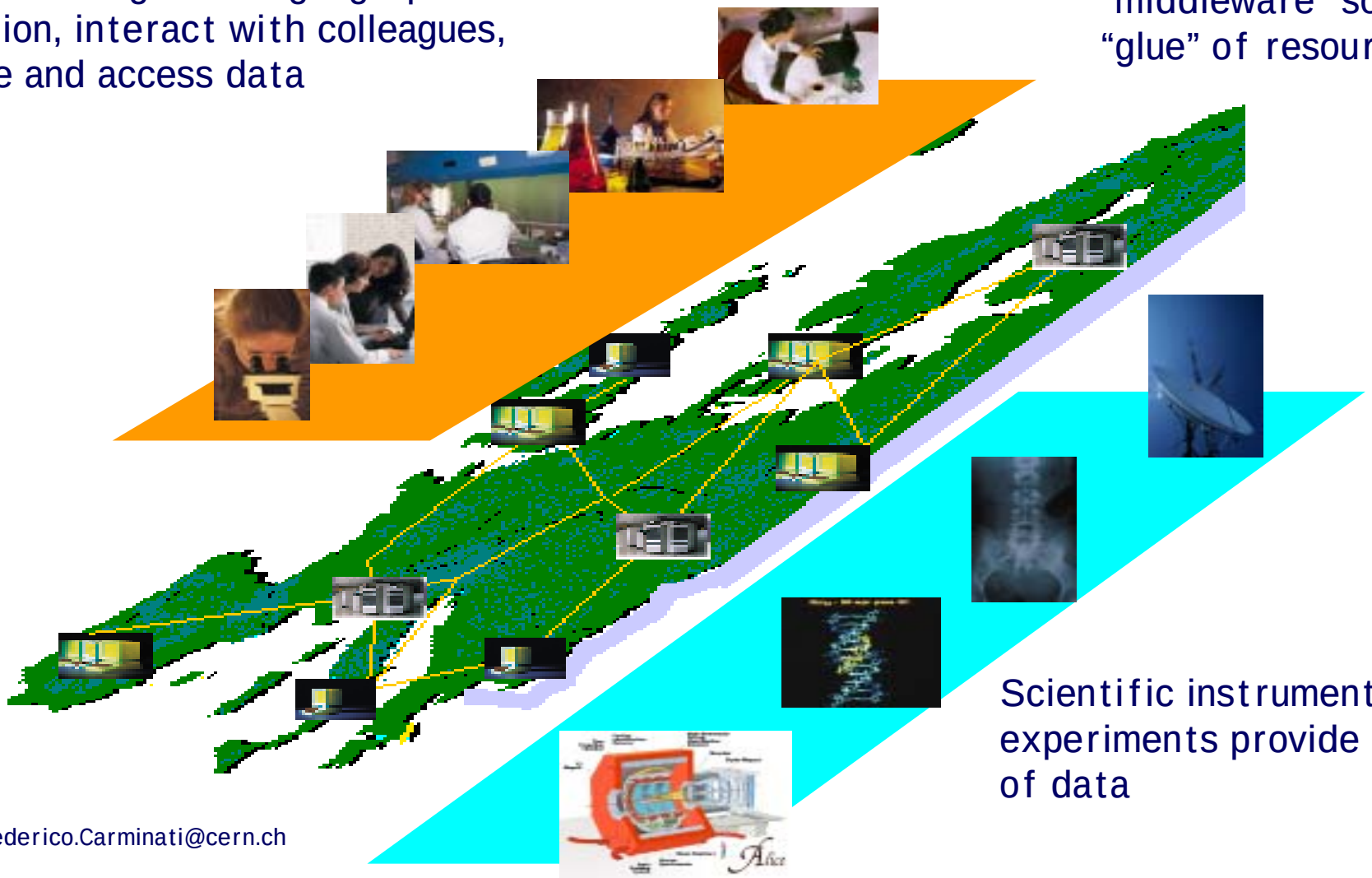
- Why a Data Grid for Science?
- Applications: Earth Observation, Bio*, Physics
- Data, data, and more data
- Services in a Data Grid
- Building a production Grid for Science in Europe

- Does the Grid really work – a physics use case
- Lessons learned

Grid – a vision

Researchers perform their activities regardless geographical location, interact with colleagues, share and access data

The GRID: networked data processing centres and ”middleware” software as th “glue” of resources.



Scientific instruments and experiments provide huge amounts of data

Federico.Carminati@cern.ch

The Growth of Computational Grids

- Linking super computers to solve specific 'grand-challenges'
  - aircraft design simulations
  - climate modelling
  - ASCI: weapons simulations
- Scale
  - 1—10 supercomputers
  - 10—100 users
  - grid access next to "regular" access via telnet, ssh, ftp, …
- Extremely well connected sites
- Typical lifetime of a Grid: days to weeks – i.e., only during demos
- I-WAY, GUSTO; middleware: Globus 1.x

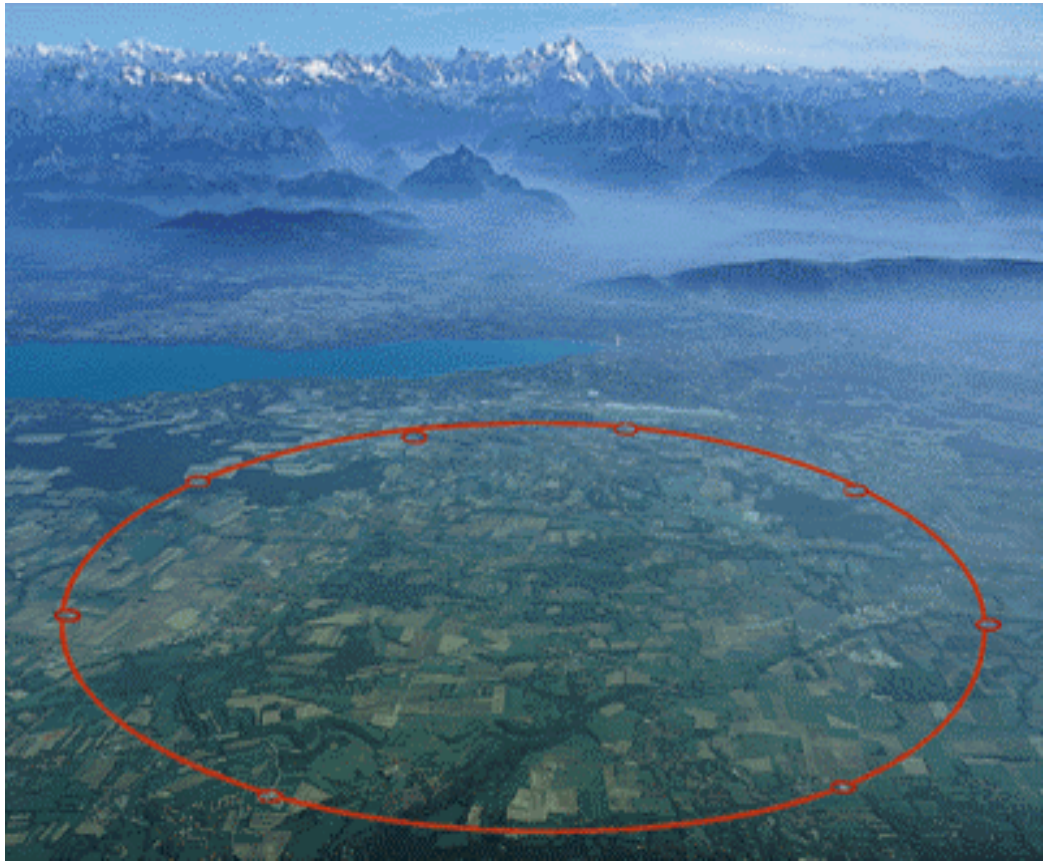
The Application data crisis

- Scientific experiments start to generate *lots* of data
  - high-resolution imaging: ~ **1 GByte** per measurement (day)
  - Bio-informatics queries: **500 GByte** per database
  - Satellite world imagery: ~ **5 TByte**/year
  - Current particle physics: **1 PByte** per year
  - LHC physics (2007): **10-30 PByte** per year

- Scientists are highly distributed in international collaborations
  - either the data is in one place and the people distributed
  - or the data circles the earth and the people are concentrated

Example: the Large Hadron Collider

- Why does matter have mass?
- Why is there any matter left in the universe anyway?



CERN
European Particle Physics Lab

LHC
Large Hadron Collider

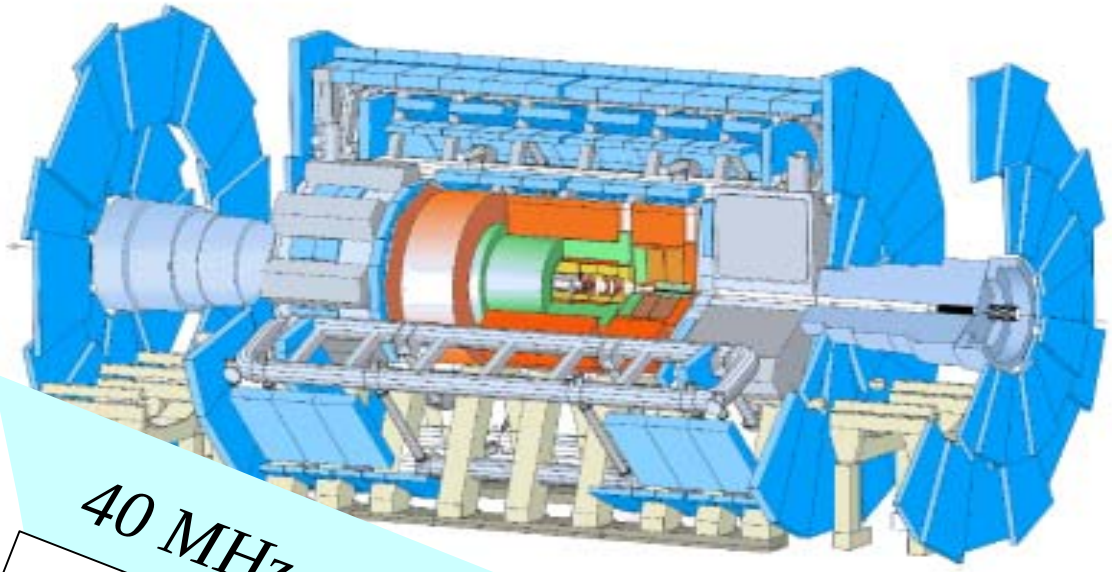
- 27 km circumference, 4 experiments
- first beam in 2007: 10 PB/year
- data 'challenges' :

2004	10%
2005	20%
2006	50%

Data Collection at the LHC

Physics @ CERN

- 10-15 Petabyte per year
- 150 countries
- > 10000 Users
- lifetime ~ 20 years



40 MHz (40 TB/sec)

level 1 - special hardware

75 KHz (75 GB/sec)

level 2 - embedded

5 KHz (5 GB/sec)

level 3 - PCs

100 Hz

(100 MB/sec)

data recording & offline analysis

http://www.cern.ch/

Data Handling and Computation for Physics Analysis

ector

event filter (selection & reconstruction)

reconstruction

raw data

event reprocessing

event summary data

proces data

batch physics analysis

analysis

analysis objects
(extracted by physics topic)

event simulation

simulation

interactive physics

The Grid, a solution for DISc?

Grid was the logical next step in the end of the 1990:

- Harnassing desktop power became commonplace
  - 1988: Condor, later: SETI@Home, Entropia, Distributed.NET
- Peer-to-peer data access protocols emerged
  - 1999: Napster, later: Gnutella, KaZaa, BitTorrent
- Network access became extremely fast
  - 1997: wide area bandwidth starts to double every 9 months!
- 1997: **Globus** starts developing basic middleware
  - 1996: middleware by Legion, 2000: Unicore
- Massive take-up of the Grid vision in 1999
  - lead in Europe by the EU DataGrid
  - others include: NASA-IPG, CrossGrid, GridLab, PPDG, Alliance, …

Application constraints

- Running of legacy code
  - 100 Mlines of code in HEP, in Fortran-IV to C++ (and Java)
  - need smooth transition from current systems
  - complex dependencies on system and external libraries
- Seemingly simple distribution model
  - events are independent
  - only a few `interesting' events
  - trivially parallel problem
- Complex distribution model
  - large data sets
  - highly distributed user, resource community

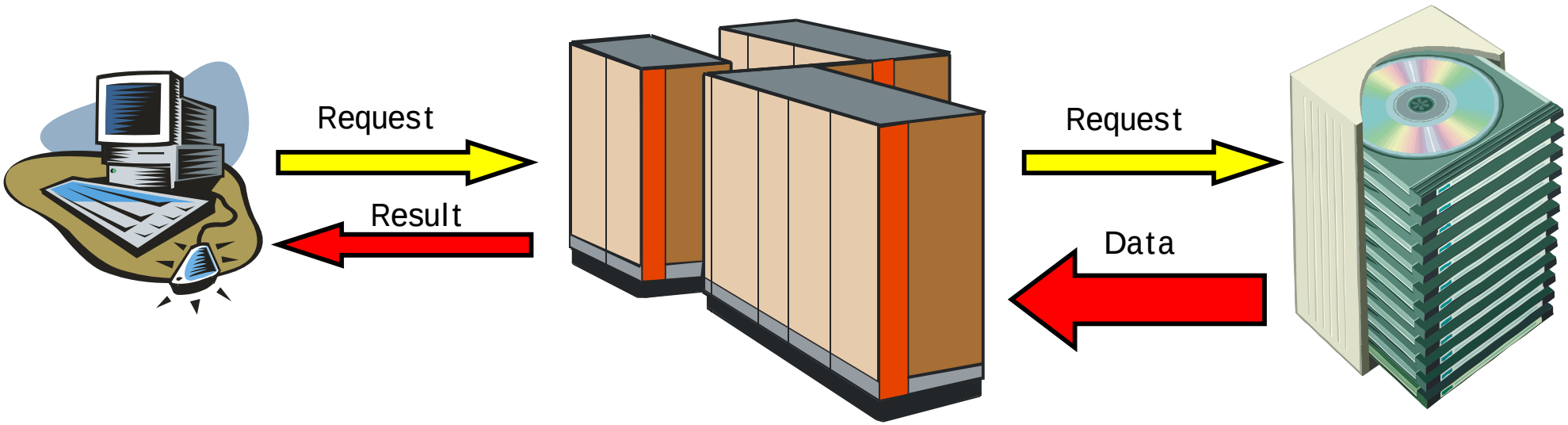
A Working Grid: the EU DataGrid

Objective:

build the next generation computing infrastructure providing intensive computation and analysis of shared large-scale databases, from hundreds of TeraBytes to PetaBytes, across widely distributed scientific communities

- official start in 2001
- 21 partners
- in the Netherlands: NIKHEF, SARA, KNMI
- Pilot applications:
  earth observation, bio-medicine, high-energy physics
- aim for production and stability

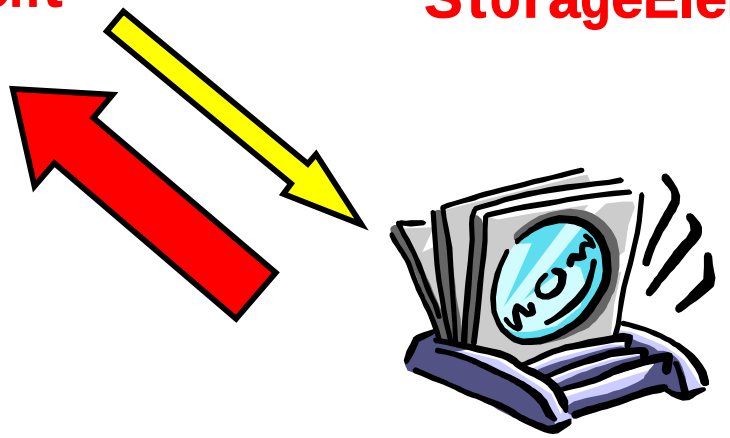
A ‘tiered’ view of the Data Grid



Client
‘User Interface’

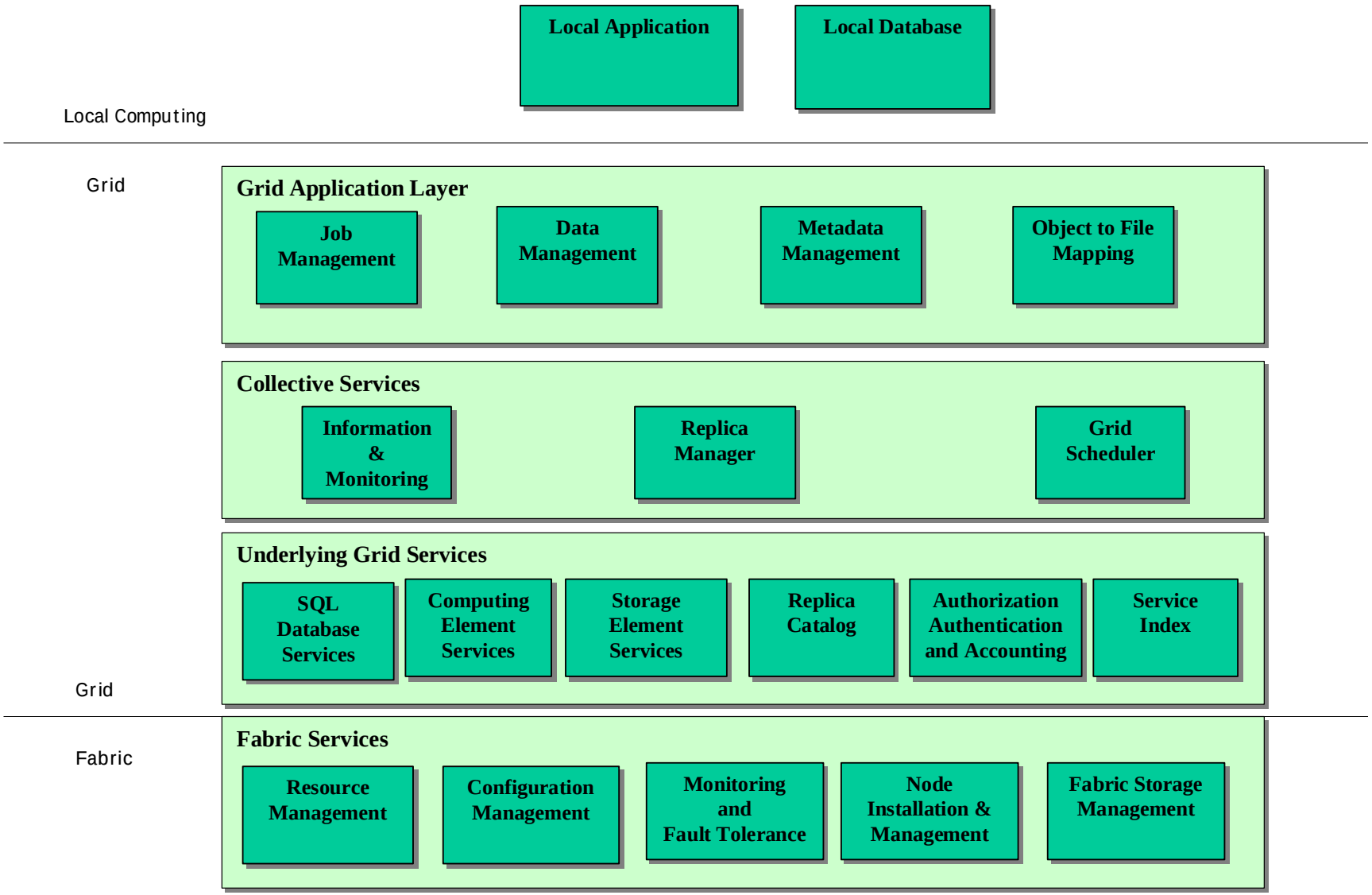
Execution Resources
‘ComputeElement’

Data Server
‘StorageElement’



Database server

A DataGrid 'Architecture'



Fabric services

- Full (Fool?) proof installation of grid middleware
  - each grid component has ~50 parameters to set
  - there are ~50 components
  - there are *at least* 2500 ways to mess up a single site
  - x 100 sites
  - 2500**100 = 10^339 ways to misconfigure the Grid … and only 1 correct way!
- automated installation and configuration of grid service nodes
- versioned configuration data: centrally checked, local derivates
- Installs everything from OS, middleware, etc.
  - no user intervention
  - installs a system from scratch in 10 minutes.
  - scales to >1000 systems per site.

- Fabric monitoring and correlation of error conditions

Security (=AAA) services

- 1st generation grids were only user based; weak identity vetting
- scientific collaboration involves
  - putting people in groups (all people looking for J/ψ's)
  - assigning roles to people ('disk space administrator')
  - handing out specific capabilities (a 100 Gbyte quota for this job)
- Sites do not need know about groups and roles
- Sites should not (but may!) need to know about users

Building a VO enabled grid:

- Site administrators can enable VO's as a whole
- traceability must be maintained (if only for legal reasons)

Virtual Organisations on EDG, LCG

A *single* grid for *multiple* virtual organisations

- HEP experiments: ATLAS, ALICE, CMS, LHCb
- software developers: ITeam
- GOME Earth Observation
- BioMedical Applications Group

Resources shared between LCG 'at large' and locally

- VL-E Certification testers in a separate VO 'P4'
- use LCG-1 resources at NIKHEF and SARA, but nowhere else

- One identity, many VOs: coordinated Grid AuthN PKI in Europe.

AuthN: Many VOs - a common identity

- EU Grid Policy Management Authority (EUGridPMA)
- A single PKI for Grid authentication, based on 20 member CAs
- Hand-on group to define minimum requirements
  - each member drafts a detailed CP/CPS
  - identity vetting: in person, via passports or other "reasonable" method
  - physical security: off-line system, HSM FIPS-140 level 3
  - no overlapping name spaces
  - no external auditing, but detailed peer-review
- Links in with **gridpma.org** and the International Grid Federation
- Each individual gets a single identity certificate –
  to be used for all Virtual Organisations

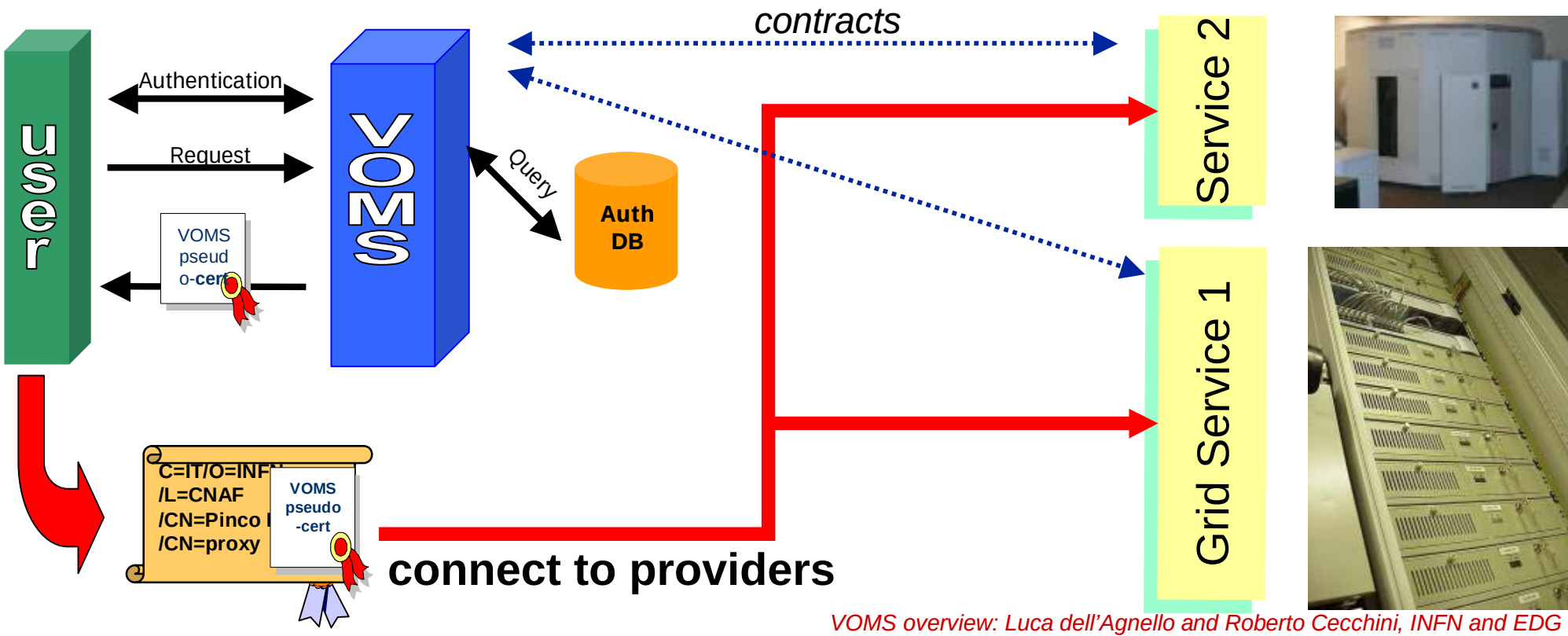
AuthZ: GSI and VOMS

Crucial in Grid computing: it gives **Single Sign-On**

GSI uses a Public Key Infrastructure with ***proxy-ing*** and ***delegation***

multiple VOs per user, groups and role support in **VOMS**

VO Membership Service



Basic DataGrid building blocks

Computing Element Service

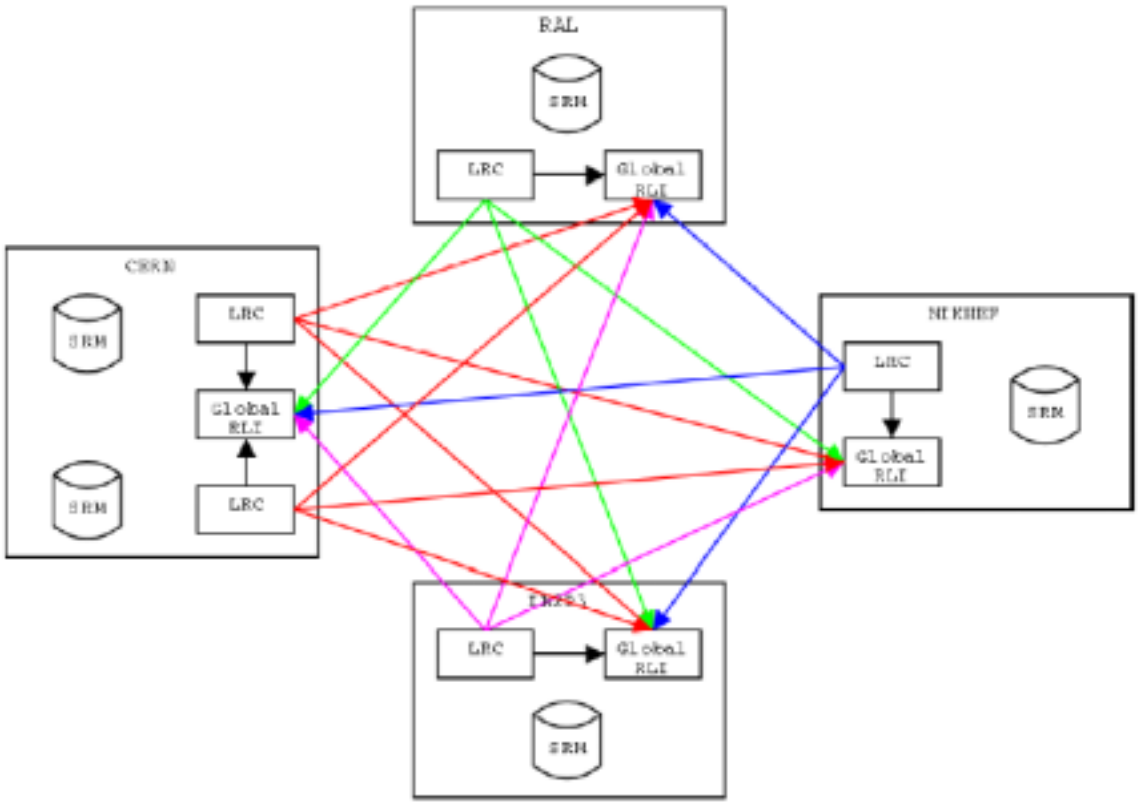
- accept authorised job requests
- acquire credentials (uid, AFS token, Kerberos principals; NIS or LDAP)
- run the job with these credentials on a cluster or MPP system
- provide job management interface (on top of PBS, LSF, Condor)

Storage Element Service

- more than just GridFTP!
- pre-staging, optimize tape access patterns, pinning
- cache management (esp. for replica clean-out: CASTOR, dCache, DMF)

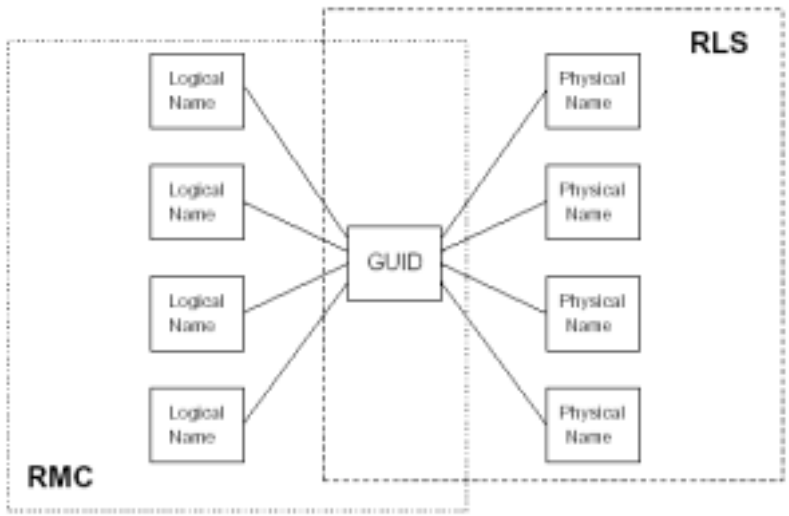
Replica Location Service

- Search on file attributes and experiment-specific meta-data (**RMC**)
- Find replicas on (close) Storage Elements (**LRC**)
- Distributed indexing of LRCs (the **RLI**)

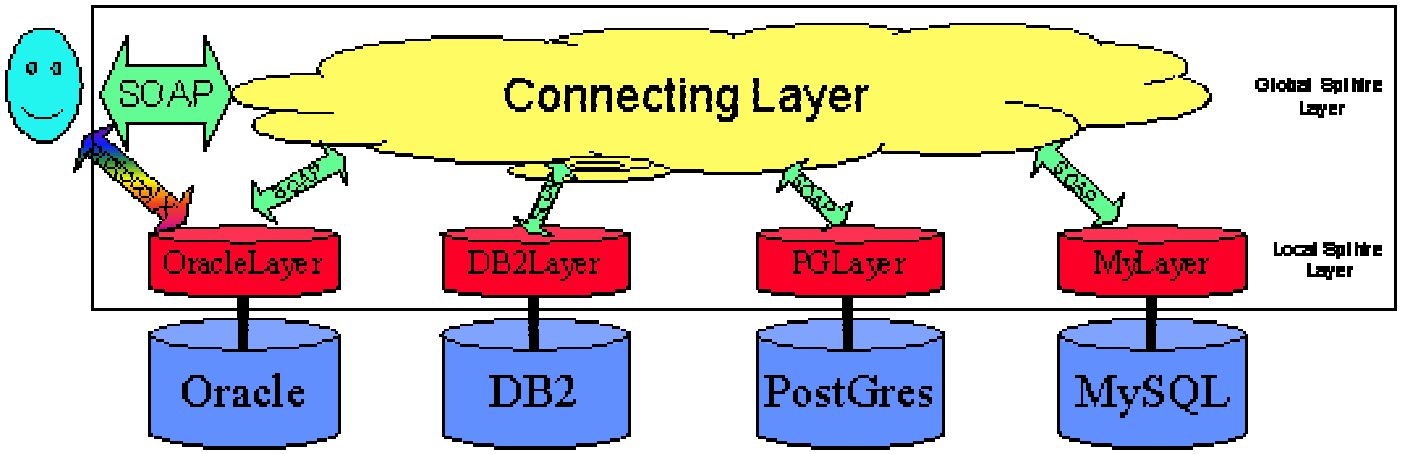


ATLAS Replica Loc. Service

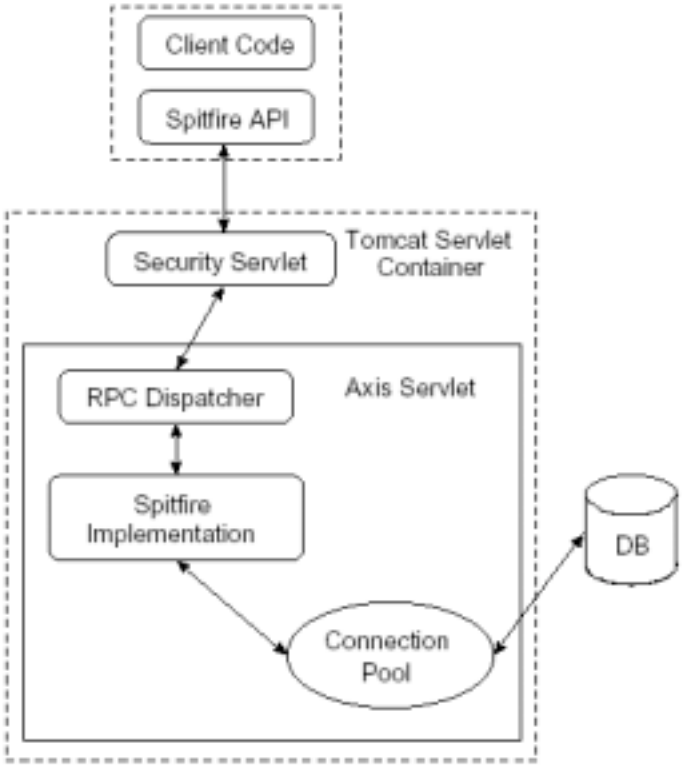
```
higgs1.dat > GUID
GUID>      sara:atlas/data/higgs1.dat
           cern:lhc/atlas/higgses/1.d
higgs2.dat, ...
...        cern:lhc/atlas/higgses/2.d
```



Spitfire: Database access & security



- common access layer for MySQL, Oracle, DB/2, …
- includes GSI, VOMS-based authorisation (per cell granularity)
- connection caching (for accesses with same set of VOMS attributes)

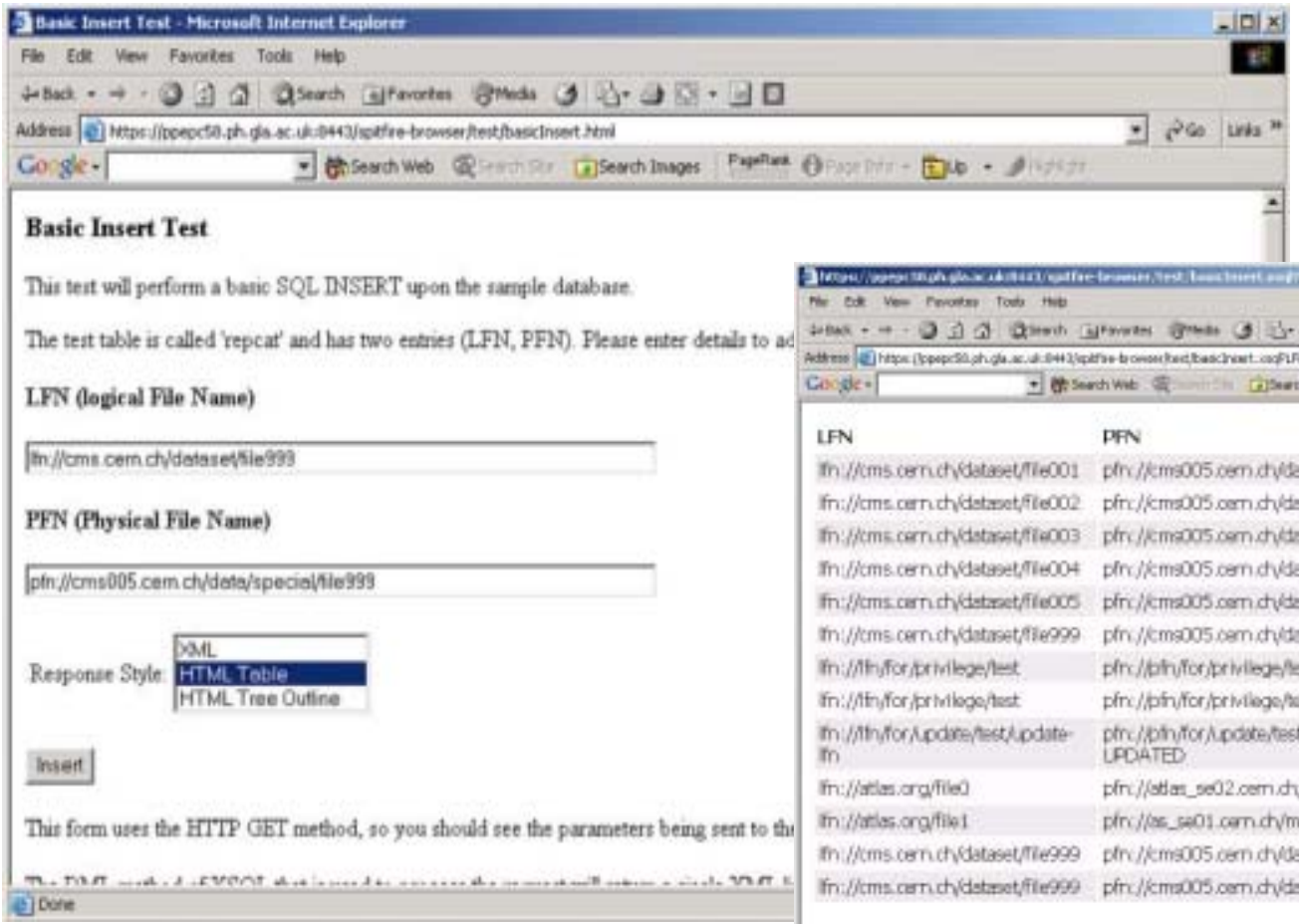


Spitfire: Access to Data Bases

- find datasets based on content queries
- e.g. GOME satellite data within a geographic region

Access via

- Browser
- Web Service
- Commands



Screenshots: Gavin McCance, Glasgow University and EDG

More advanced blocks

- Accounting
  - we keep usage data (from the job manager)
  - rely on existing local systems (for PBS, LSF, etc)
  - no exchange of information within the grid yet

scientists do not like accounting,
need no billing,
and certainly no payment!

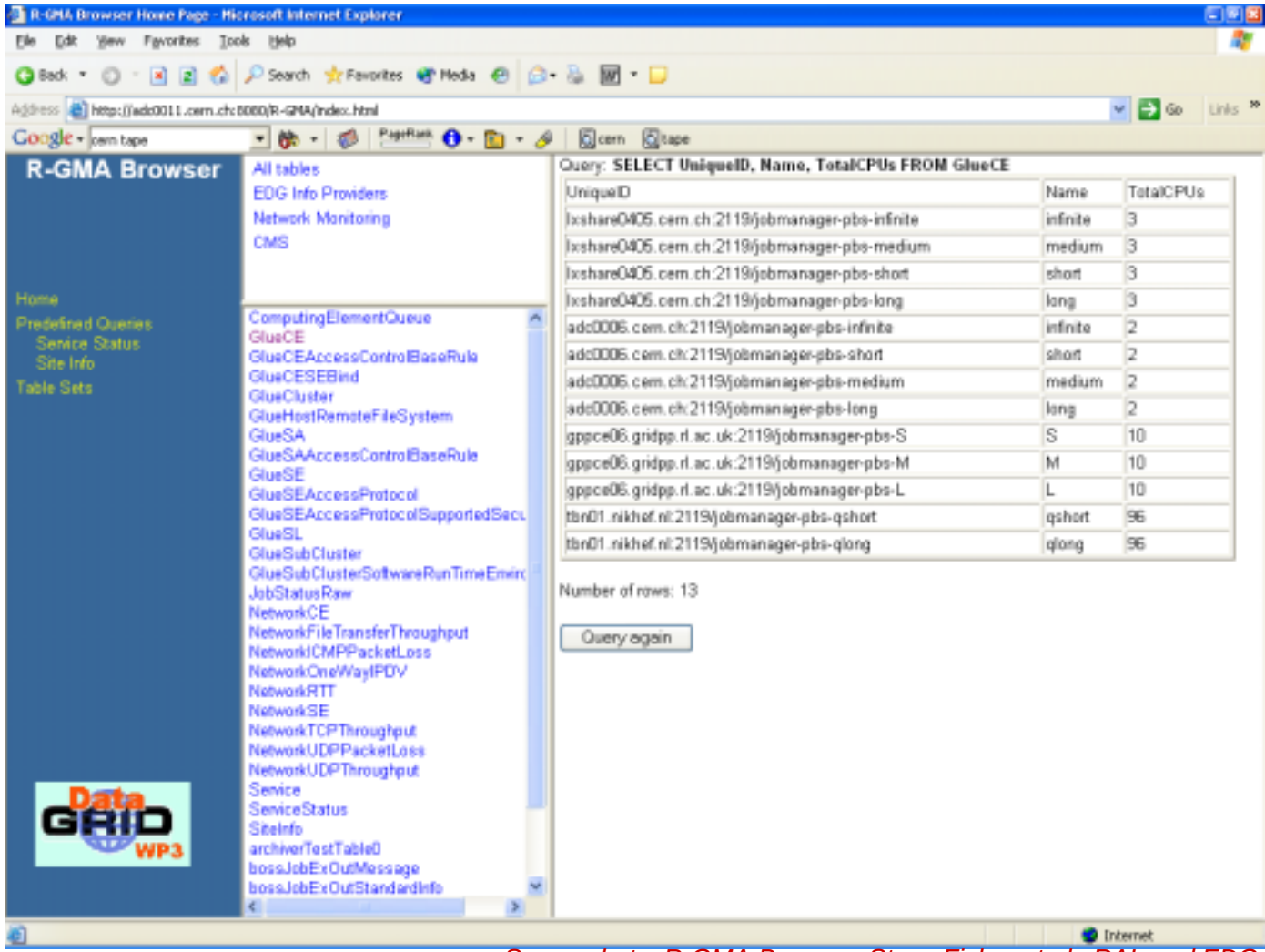
Collective services

- Information and monitoring
  - Finding resources with certain characteristics (RunTimeEnvironmentTag)
  - Finding correlated resources ('close' SEs to a CE, NetworkCost function)
- Grid Scheduler
  - Resource Broker:
    Environment requirements
    Quantitative requirements (#CPUs, WallTime)
    Dataset requirements (LFNs needed, output store needed)
    } **JDL**
  - Workload Management System
    Sandboxing input and output files
    Resilience
    mobile and asynchronous use
- Replica Manager
  - Reliable file transfer
  - migrate data to close(r) storage elements
  - give the best location to get a file from

Grid information: R-GMA

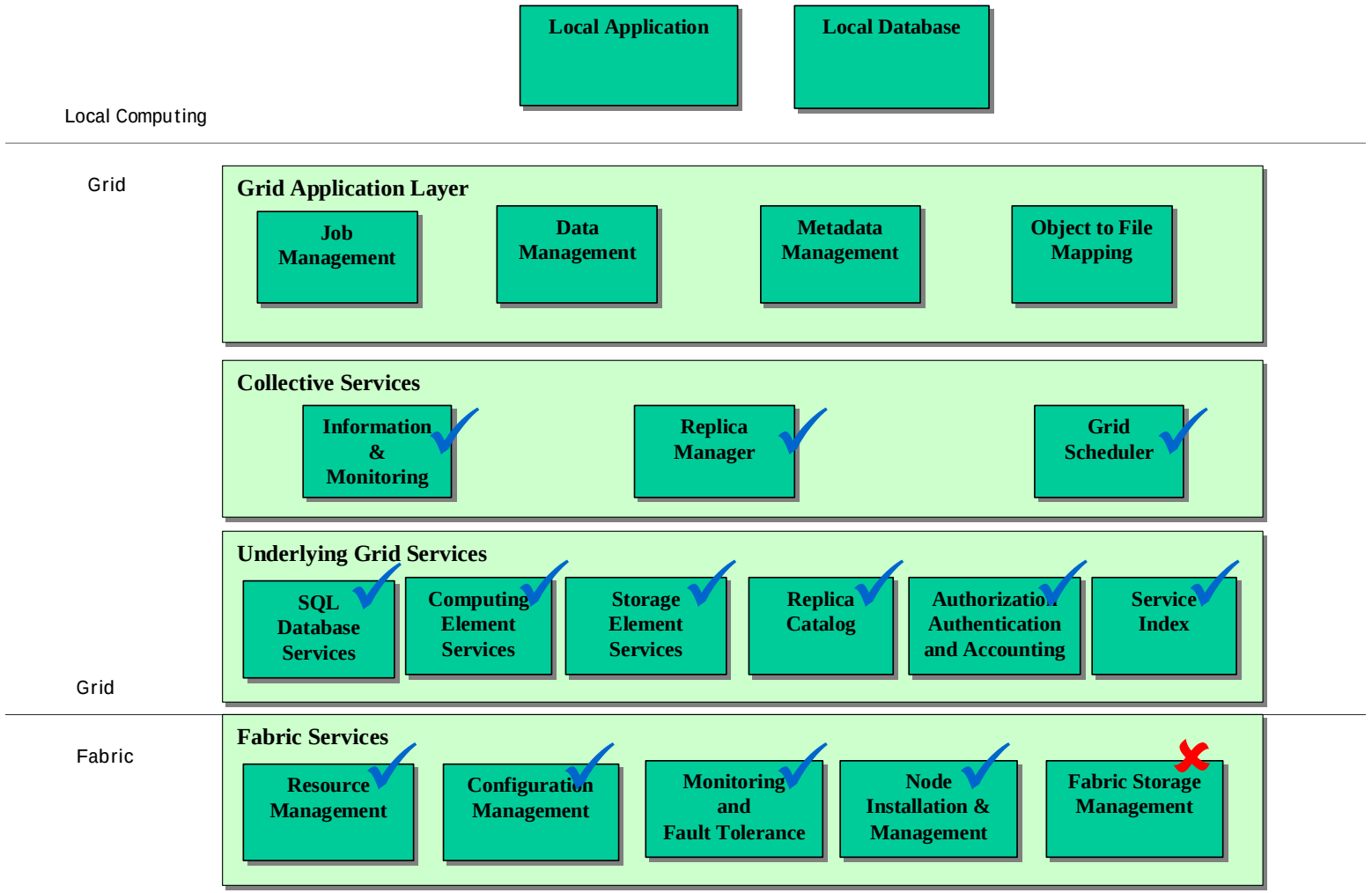
Relational Grid Monitoring Architecture

- a Global Grid Forum standard
- Implemented by a relational model
- used by grid brokers
- application monitoring



Screenshots: R-GMA Browser, Steve Ficher et al., RAL and EDG

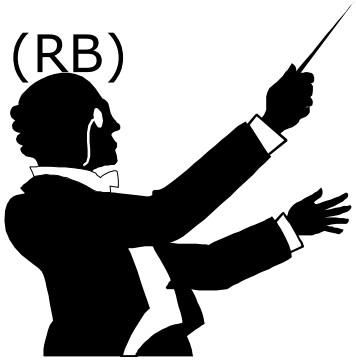
Components in the Architecture



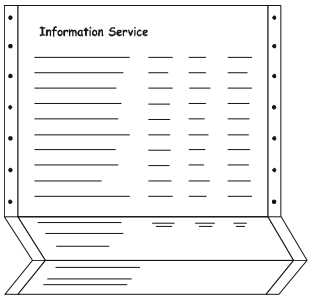
Fitting things together: GridElements

1. User Interface (UI)

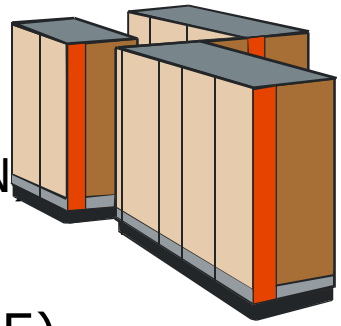
2. Resource Broker (RB)



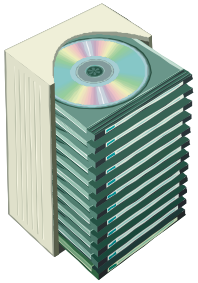
3. Information Service (IS)



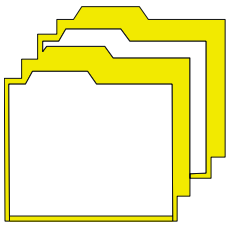
4. Computing Element (CE)
   - Gatekeeper
     (Front-end Node)
   - Worker Nodes (WN



5. Storage Element (SE)



6. Replica Catalog (RC)



An Actual Job Submission

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

  -r the job is submitted by the RB directly to the computing element identified by *<res_id>*

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

    DONE or ABORTED

    READY

    RUNNING

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

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

    Useful for other commands, e.g.:

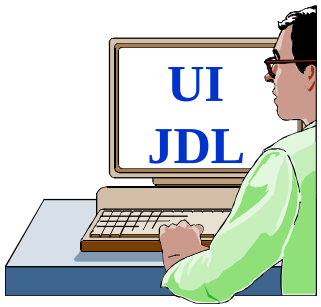
      edg-job-status -i *<input file>* (or dg_jobId)

        -i the status information about dg_jobId contained in the *<input file>* are displayed

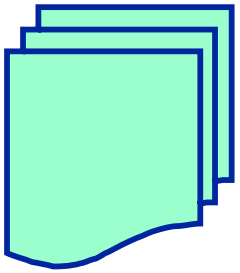
Job Description Language

```
Executable    = “dzero-reco-p14.1-runjob”;
StdError      = “stderr.log”;
StdOutput     = “stdout.log”;
InputSandbox  = {“/home/joda/test/gridTest”};
OutputSandbox= {“stderr.log”, “stdout.log”};
InputData     = “LF:AtlRun064558-18-jun-2003.root”;
ReplicaCatalog= “ldap://sunlab2g.cnaf.infn.it:2010/ \
                lc=test, rc=WP2 INFN Test, dc=infn, dc=it”;
Requirements  = other.Architecture==“INTEL” && \
                other.OpSys==“RedHat-7.3” && other.FreeCpus >=4;
Rank          = “other.EstimatedTraversalTime”;
DataAccessProtocol = “gridftp”;
```

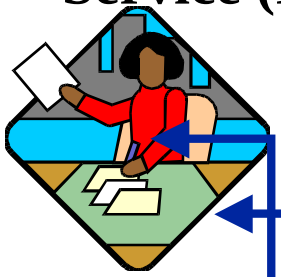
Job Submission Scenario



Replica Catalogue (RC)

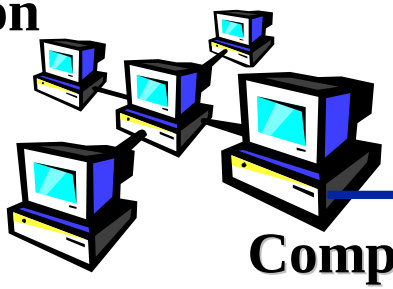


Information Service (IS)



Resource Broker (RB)

Job Submission Service (JSS)

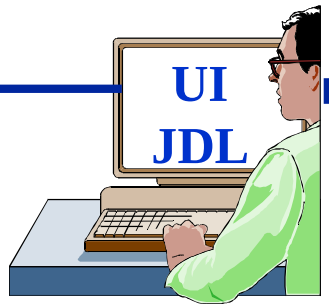


Compute Element CE)

Storage Element (SE)

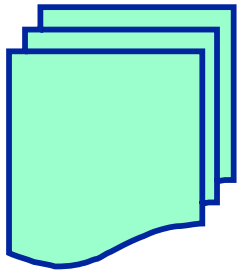
gging & ookkeeping B)

A Job Submission Example

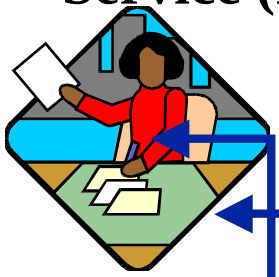


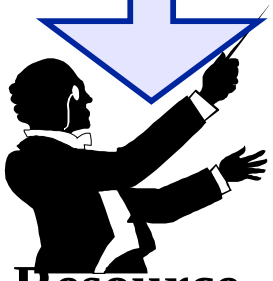
Input Sandbox

Replica Catalogue (RC)



Information Service (IS)



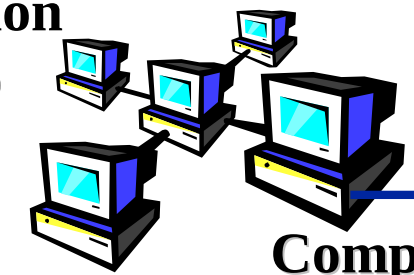


Resource Broker (RB)

Job Submit Event

ogging & ookkeeping B)

Job Submission Service (JSS)



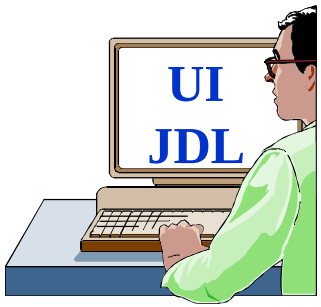
Compute Element (CE)

Storage Element (SE)

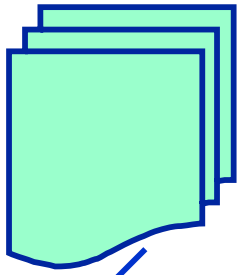
Job Status

submitt

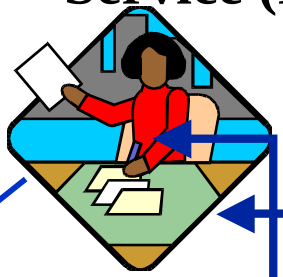
A Job Submission Example



Replica Catalogue (RC)

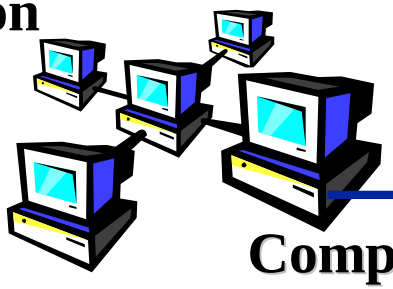


Information Service (IS)

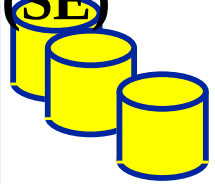


Resource Broker (RB)

Job Submission Service (JSS)



Compute Element (CE)



Storage Element (SE)

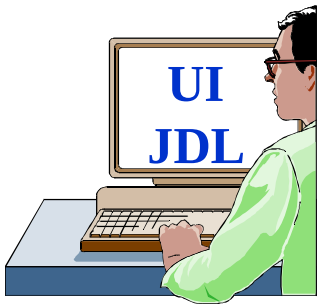
ogging & ookkeeping B)

Job Status

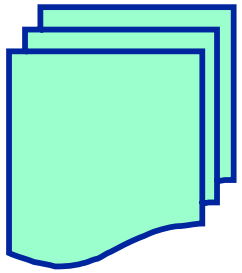
submitt

waitin

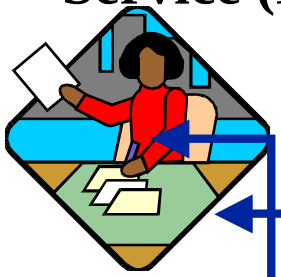
A Job Submission Example



Replica Catalogue (RC)



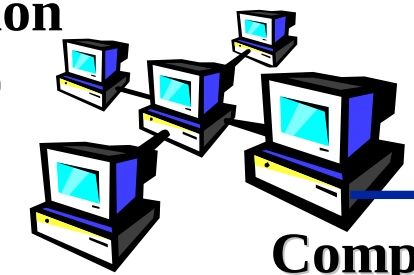
Information Service (IS)



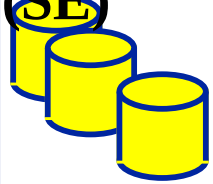
Resource Broker (RB)

ogging & ookkeeping B)

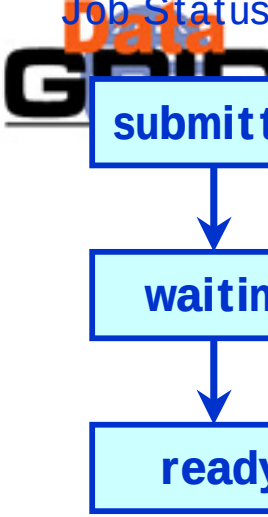
Job Submission Service (JSS)



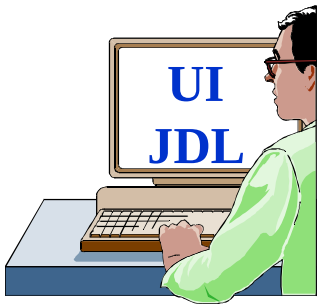
Compute Element (CE)



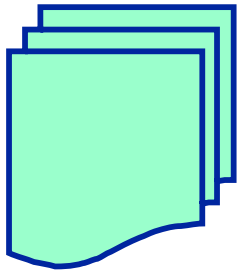
Storage Element (SE)



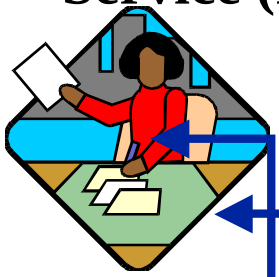
A Job Submission Example



Replica Catalogue (RC)



Information Service (IS)



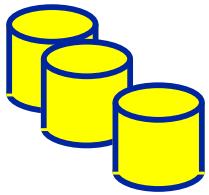
Resource Broker (RB)

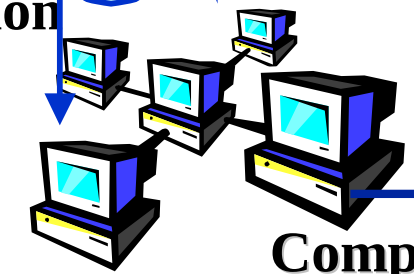
ogging &
ookkeeping
B)

Job Submission Service (JSS)

BrokerInfo

Storage Element (SE)





Compute Element (CE)

Job Status

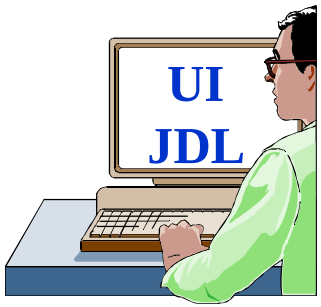
submitt

waitin

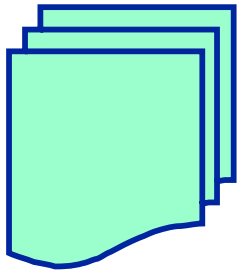
ready

schedul

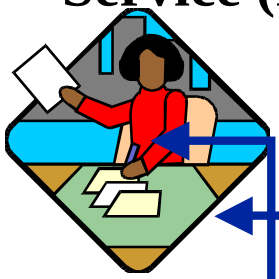
A Job Submission Example



Replica Catalogue (RC)



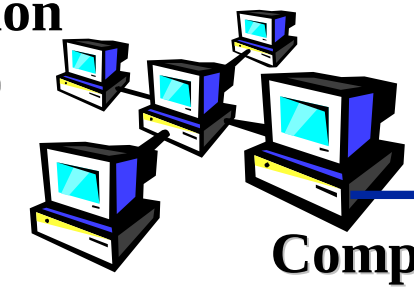
Information Service (IS)



Resource Broker (RB)

Input Sandbox

Job Submission Service (JSS)



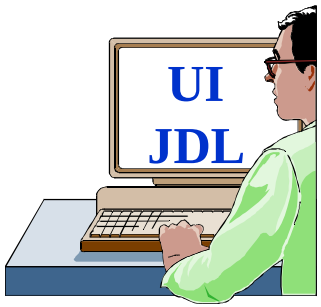
Compute Element (CE)

Storage Element (SE)

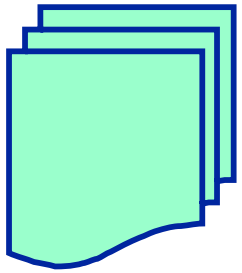
ogging & ookkeeping B)

Job Status

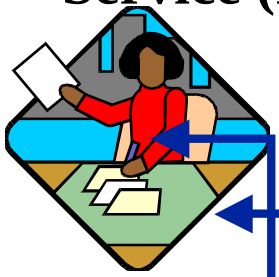
A Job Submission Example



Replica Catalogue (RC)



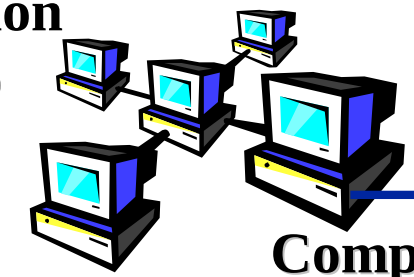
Information Service (IS)



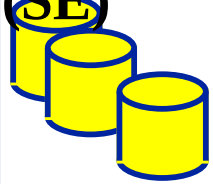
Resource Broker (RB)

Job Submission Service (JSS)

ogging & ookkeeping B)



Compute Element (CE)



Storage Element (SE)

Job Status

Job Status

submitt

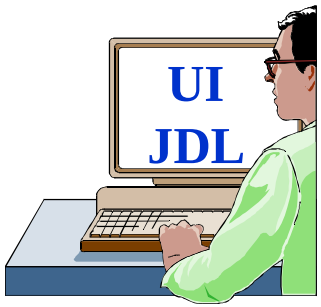
waitin

ready

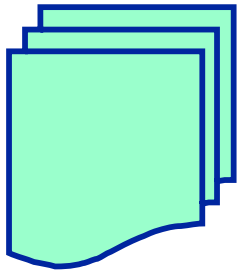
schedul

runnin

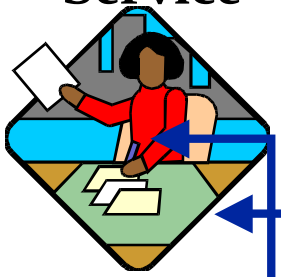
A Job Submission Example



Replica Catalogue



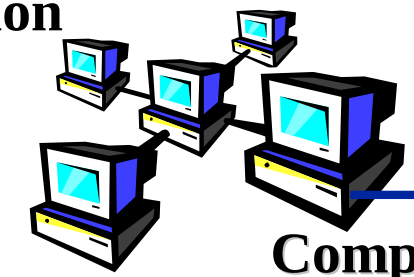
Information Service



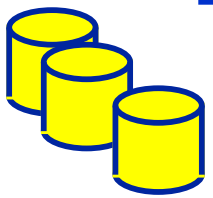
Resource Broker

gging &
ookkeeping

Job Submission Service

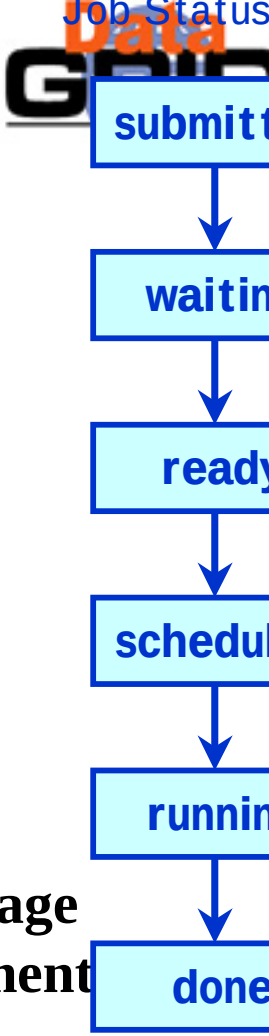


Compute Element

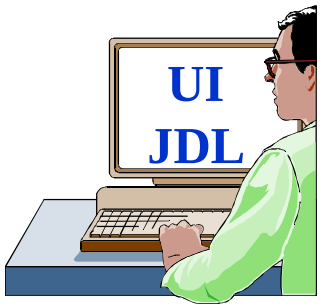


Storage Element

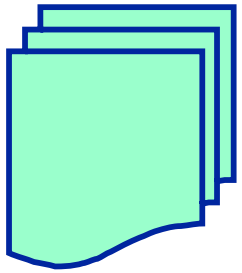
Job Status



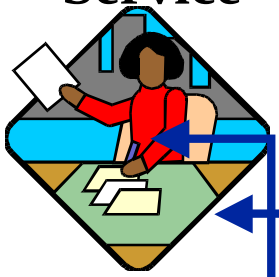
A Job Submission Example



Replica Catalogue



Information Service

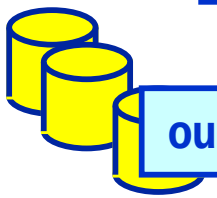


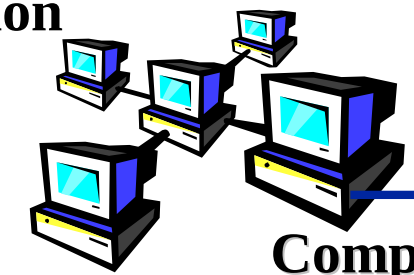
Resource Broker

ogging &
ookkeeping

Job Submission Service

Storage Element





Output Sandbox

Job Status

Compute Element

Job Status

submitt

waitin

ready

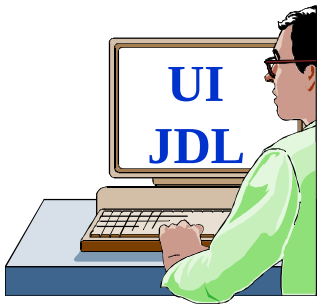
schedul

runnin

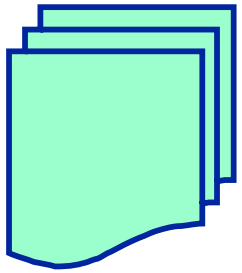
done

outputrea

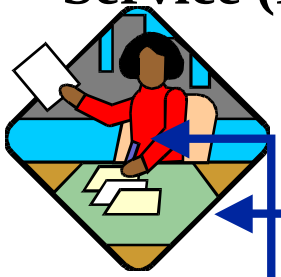
A Job Submission Example



Replica Catalogue (RC)



Information Service (IS)

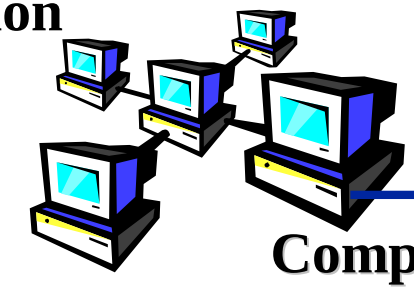


Output Sandbox

Resource Broker (RB)

ogging &
ookkeeping
B)

Job Submission Service (JS)



Compute Element (CE)

Storage Element (SE)

Job Status

submitt

waitin

ready

schedul

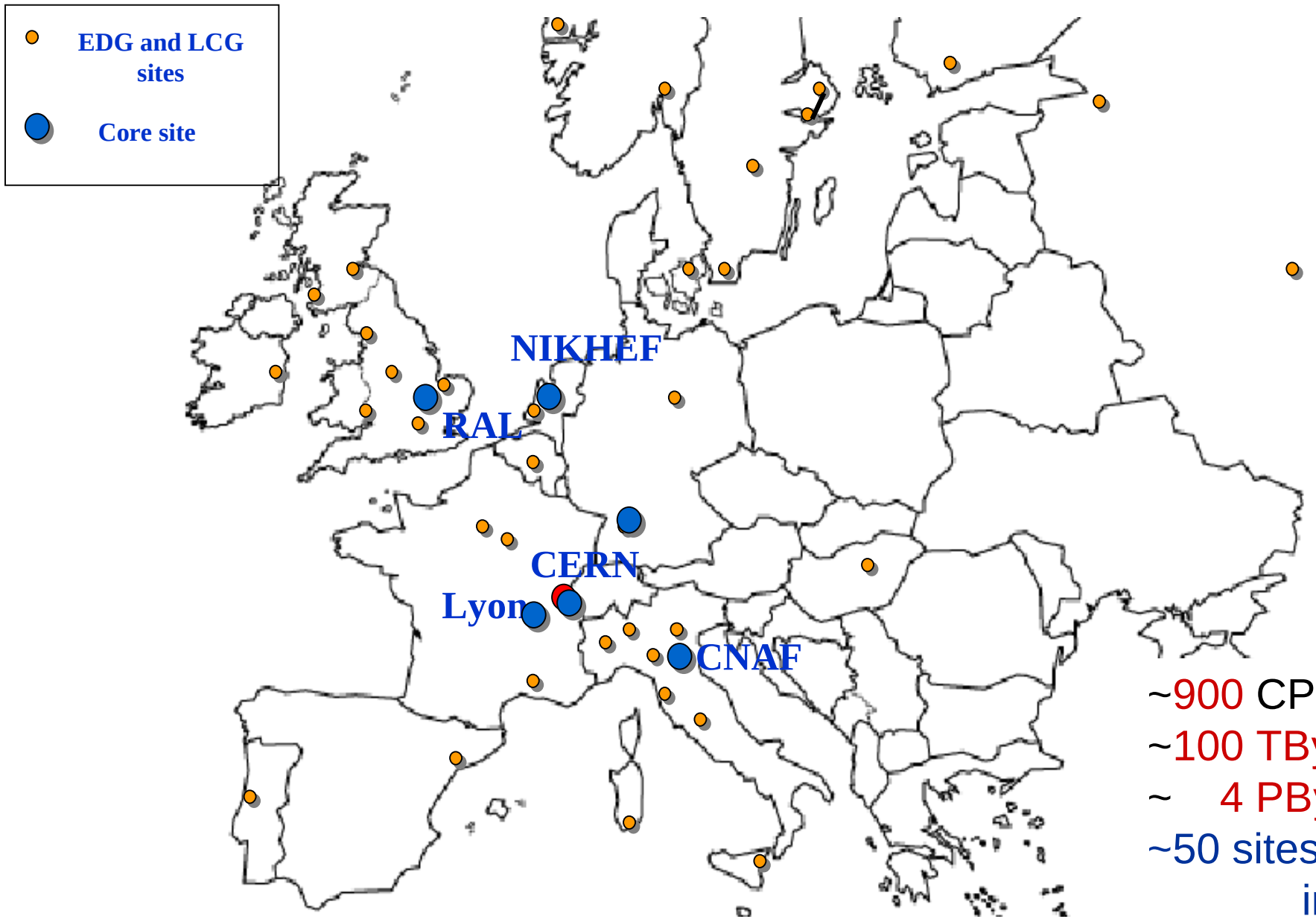
runnin

done

outputrea

cleare

Current EDG and LCG Facilities

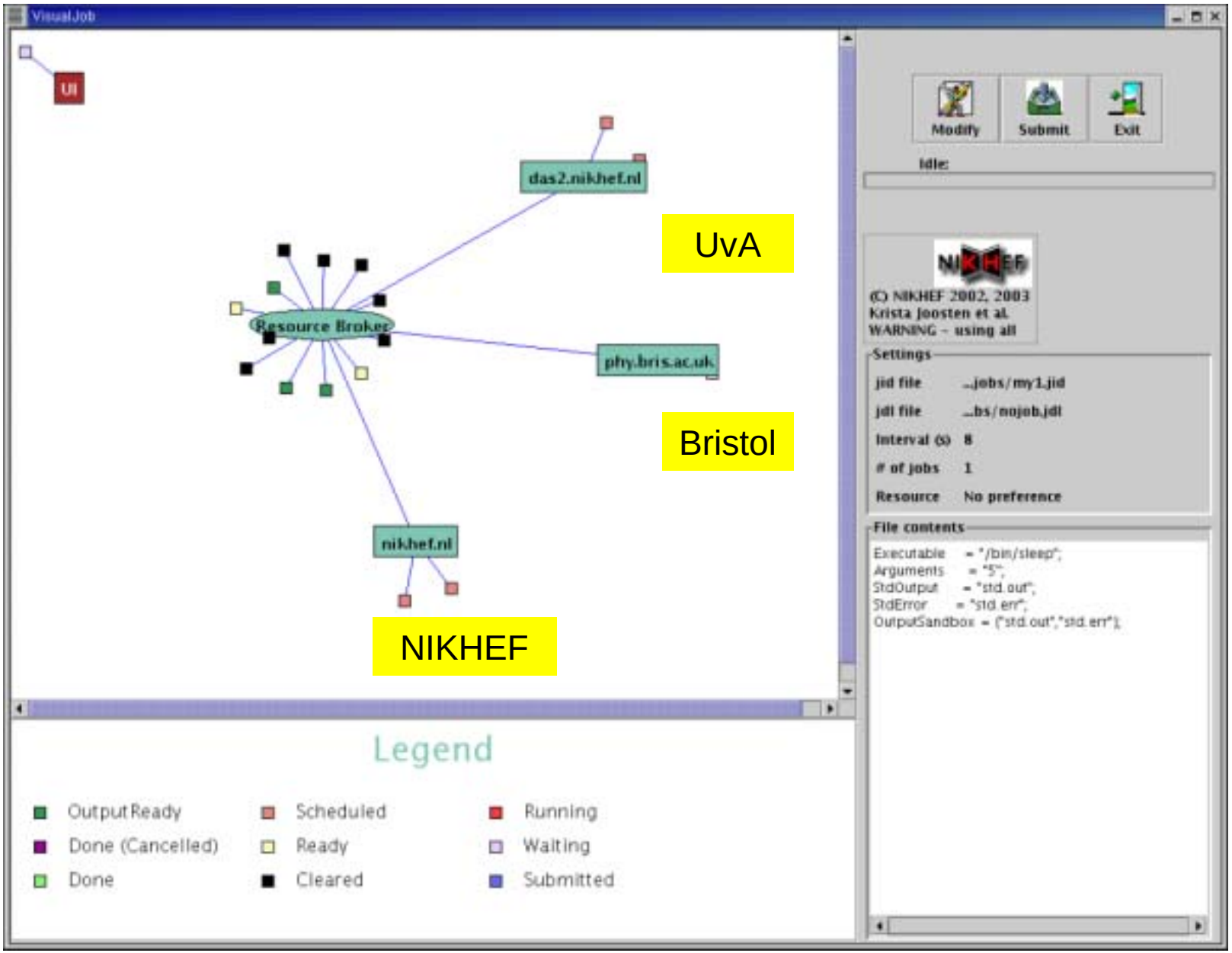


Building it: LCG Production Facility

- ~50 resource provider centres (some go up, some go down)
- Many 'small' ones and a few large ones:

```
…                                                                934 Total
GlueCEUniqueID=lhc01.sinp.msu.ru                      GlueCEInfoTotalCPUs: 2
GlueCEUniqueID=compute-0-10.cscs.ch                   GlueCEInfoTotalCPUs: 4
GlueCEUniqueID=dgce0.icepp.s.u-tokyo.ac.jp            GlueCEInfoTotalCPUs: 4
GlueCEUniqueID=farm012.hep.phy.cam.ac.uk              GlueCEInfoTotalCPUs: 5
GlueCEUniqueID=golias25.farm.particle.cz              GlueCEInfoTotalCPUs: 6
GlueCEUniqueID=lcgce01.gridpp.rl.ac.uk                GlueCEInfoTotalCPUs: 6
GlueCEUniqueID=lcg00105.grid.sinica.edu.tw            GlueCEInfoTotalCPUs: 8
GlueCEUniqueID=lcgce01.triumf.ca                      GlueCEInfoTotalCPUs: 8
GlueCEUniqueID=hik-lcg-ce.fzk.de                      GlueCEInfoTotalCPUs: 14
GlueCEUniqueID=t2-ce-01.roma1.infn.it                 GlueCEInfoTotalCPUs: 22
GlueCEUniqueID=grid109.kfki.hu                        GlueCEInfoTotalCPUs: 26
GlueCEUniqueID=t2-ce-01.to.infn.it                    GlueCEInfoTotalCPUs: 28
GlueCEUniqueID=adc0015.cern.ch                        GlueCEInfoTotalCPUs: 34
GlueCEUniqueID=t2-ce-01.mi.infn.it                    GlueCEInfoTotalCPUs: 40
GlueCEUniqueID=zeus02.cyf-kr.edu.pl                   GlueCEInfoTotalCPUs: 56
GlueCEUniqueID=t2-ce-01.lnl.infn.it                   GlueCEInfoTotalCPUs: 124
GlueCEUniqueID=wn-02-29-a.cr.cnaf.infn.it             GlueCEInfoTotalCPUs: 136
GlueCEUniqueID=grid-w1.ifae.es                        GlueCEInfoTotalCPUs: 150
GlueCEUniqueID=tbn20.nikhef.nl                        GlueCEInfoTotalCPUs: 238
```

Using the DataGrid for Real



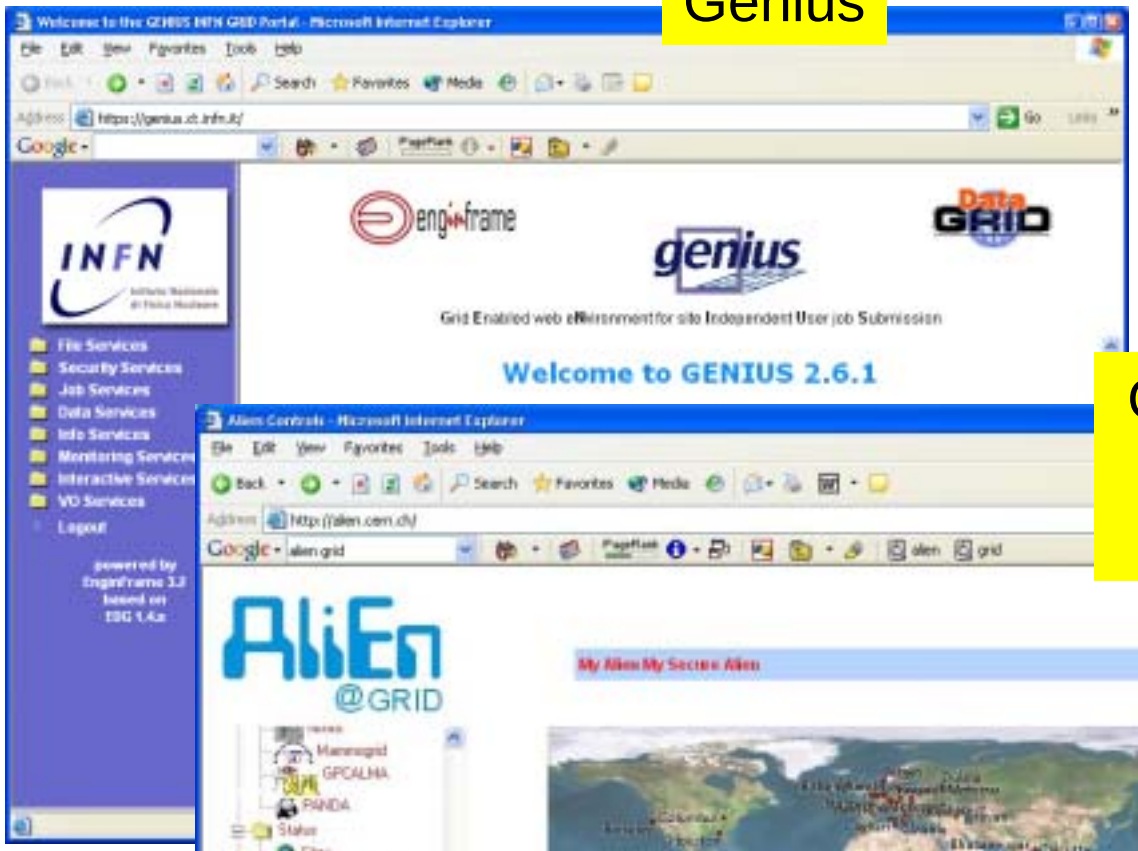
Screenshots: Krista Joosten and David Groep, NIK

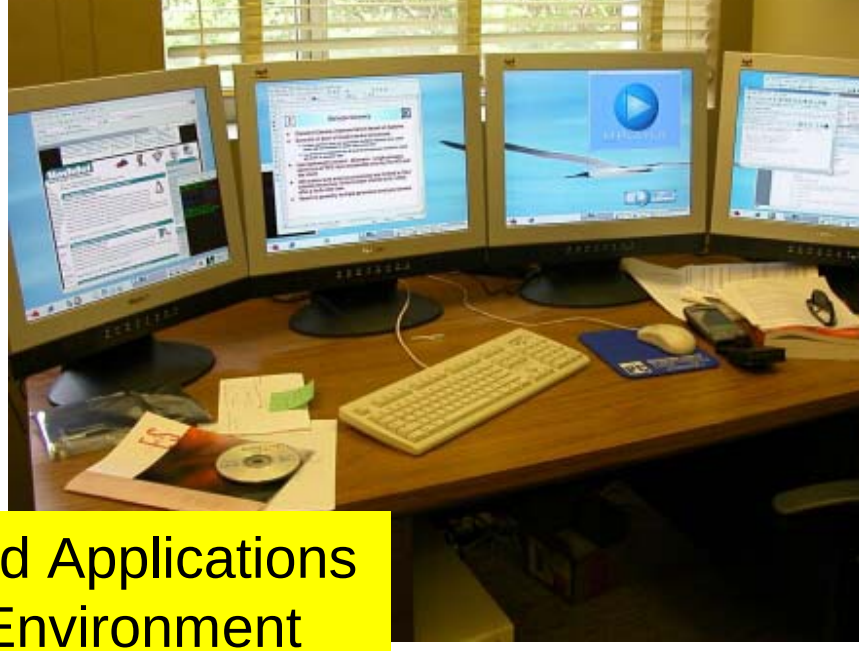
The Next Steps

- A Common Application Layer
  - more collective services
  - vo-specific interfaces
  - a “Grid API”
  - HEPCAL, end-to-end solutions for scientific problems
- Integration in VO-specific applications
  - GridBLAST front-end
  - CMS GAE
  - GANGA

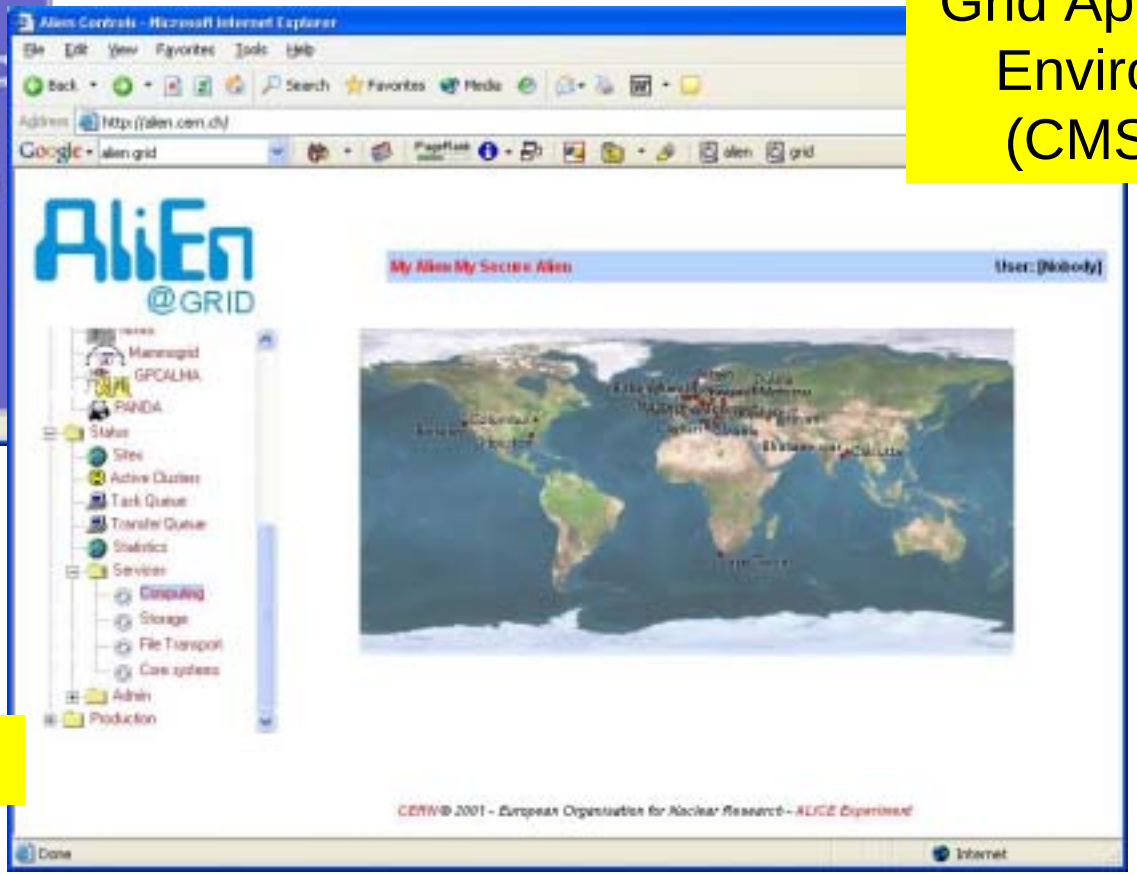
Portals

Genius

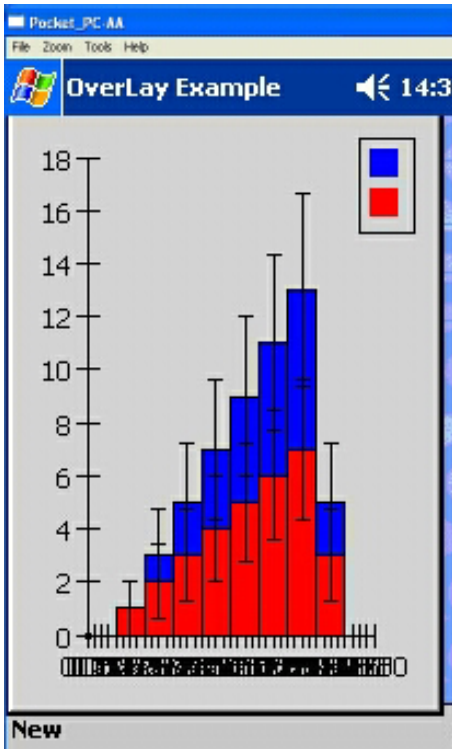




Grid Applications Environment (CMS GAE)



AliEn



Challenges ahead of us

Getting a real project to run on EDG and LCG

NIKHEF:
Jeff Templon
Kors Bos
Willem van Leeuwen
David Groep

Uni Wuppertal:
Torsten Harenberg

Imperial College:
Rod Walker

and many, many others

D0: Looking for Top and Higgs

The D0 reprocessing

- Need to reconstruct ~1000 M events from raw data
- approx. 1 PByte total, stored at FNAL
- uses the existing SAM meta-data system of the D0 experiment (Sequential Access to Meta-data)

- with new software versions coming out every month
- much legacy code (2 Gbyte of binaries in shared objects + exec)
- 2 Gbyte input file per job (~2k events/file)
- 2 Gbyte output file per job

- Can we run this on the 1000 processors in EDG and LCG?

SAM Grid: a solution?

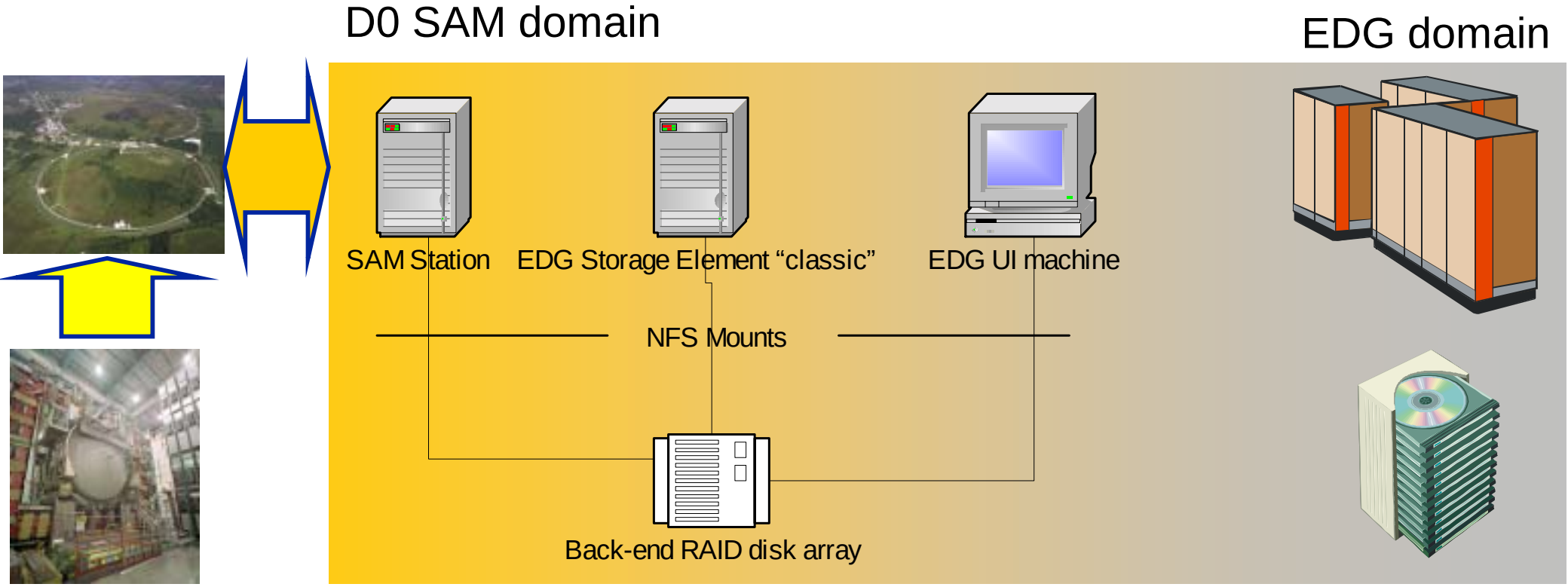
- Home-grown solution from the D0 experiment
- core assumptions:
  - all machines are for the D0 experiment
  - so they should only run D0
  - so we can decide what to install on it
  - or add just one extra box to your fabric for D0 (yeah)
- resource centres want users but they should comply with requirements
  - tracability of individual entities
  - responsibilities must be clear (site, VO, CA, end-user)
  - should not break existing stuff
- the total infrastructure should be scalable
  - not one extra box per supported VO

Getting it to run on EDG

- Getting the input data files available inside EDG Data Management
- adaptation of D0 rereco software and environment
- distribute the software to the worker nodes
- monitor job progress, success rate, and provenance
- declare reprocessed results into SAM

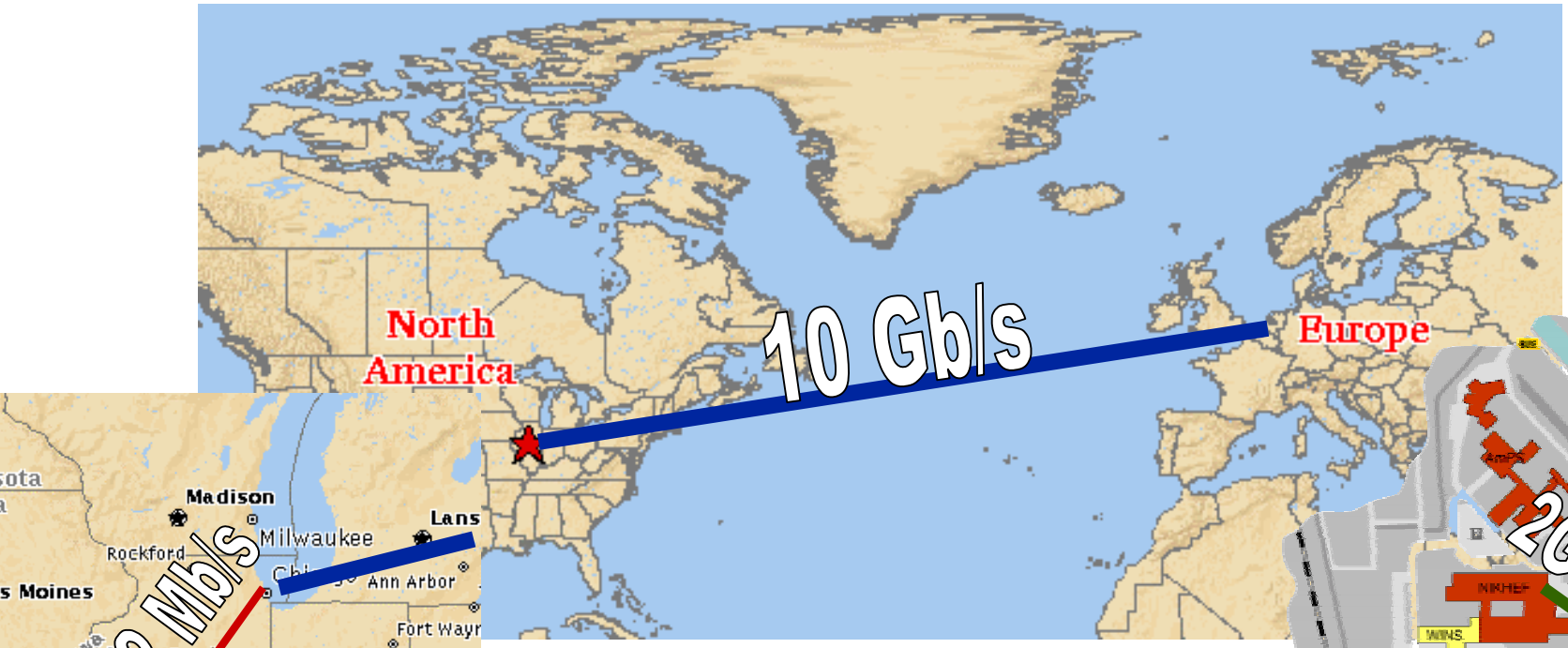
Getting the data into the system

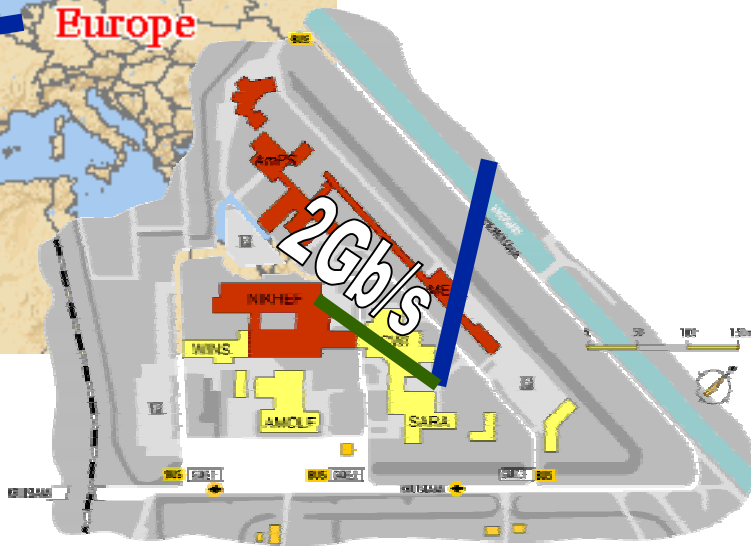
- D0 has an existing meta-data system (SAM)
- also used for job submission, and provenance information, and …
- this system is not elegant, but it works



Data rate limitations

- All data is on a *single robot* at FNAL, with only 20 tape drives
- event-picking (selective processing) kills performance
- *single SE* to mediate SAM to EDG is a bottleneck





The Software

- much legacy code, sensitive to even small changes
- many external packages (specific Python, glibc, KAI)
- even CPU arch differences are visible in the output
- physics results (and nobel prizes) depend on code reliability

➔ Build a self-contained package, with all stuff included

Obstacles

- long pathnames generated by GASS-cache hit a **`character*100`** in old code
- if job fails, 2 GByte of junk is left on the worker node in **`/tmp`** …
  … need a transient **`$TMPDI`**
- software changes too rapidly to be pre-installed
- still spurious interactions between D0 sandbox and system (glibc)

Installing the software

- The re-reco software is much too large for the input sandbox

Keeping in mind that "software is just like data"

1. tar the distribution up in a single file
2. register the proper version in the RLS
3. replicate it to the sites you expect to visit
4. use '`edg-rm getBestFile d0rc-p15.1.tar.gz`'
   to get it to the worker node

works even for new sites, just slower

But I really need *X* v3.2.2pl16-r8!

- if you insist on a particular environment, you will not get many resources!
- if you ship everything with the job, you will kill of the WMS
- some software is common to many jobs: pre-install out-of-band
- for your one-off programmes: ship it using the replica manager

- In the long run: more resource virtualisation
  - User Mode Linux
  - VMware-like sandboxing
  - Condor Bypass & Parrot
- or just rewrite the 100Mlines of legacy code in Java ☺

Runing the job: the Info System

- **Information System**: only way to find out what is there!
- In an RB push-model the InfoSys will get many queries:
  - a call to the IS to find the Replica Locations
  - one grid-wide resource query to get list of Ces
  - ask 'likely' CEs about their current state (ETT, free CPUs)
- Queries scale linearly with the # sites, and # jobs
  - 100 users submitting 50 job daily to 50 sites: 3 queries/second!
  - But: they all submit them around 10 AM

- **Globus GRIS/GIIS** – soft-registration & caching, but too slow
- **LCG BDII**: fast caching front-ends (DB3) counter amnesia in GIISes

Resource Brokering and Management

- There is no such thing as a consistent state of the grid
- The grid is too large for an RB to grasp (brokering takes 45 sec/job)

"What works with 4 sites and 0.5 users will surely break"

- **New directions**
  - the 'pull' model: sites ask for workload from a (per-VO?) work task queue
  - proximity model: akin to the super-node network in KaZaa

- The WMS keeps to much data per job that it overloads
- Clueless users are possibly worse than malicious users:
  - Users will send 5GByte of data in their input sandbox
  - Users will not realize what and how much they are submitting

RB-level Monitoring

- The Logging and Bookkeeping service to see where your job is:

  `edg-job-status`

  `edg-job-get-logging-info`

- More detailed monitoring: *instrument the job*
- (distributed) monitoring database to get this info out
  - when was the job started
  - how much CPU time has been consumed
  - which worker node was used (also for debugging)
  - where in the re-reco process did it crash
- R-GMA – *Relational Grid Monitoring Architecture*
  - Flexible Archiver: consumer+producer+MySQL database
  - Works across disconnected networks using a site *MON*-box

What is happening to the job?

Jobs may, and will, fail because…

- the UI cannot contact any RB
- job transfer from the UI fails
- RB cannot find matching resources
- the InfoSys or RB is overloaded (disk full, response too slow)
- remote side cannot get the sandbox in (firewalls, out-of-disk space)
- a worker node becomes a black hole
- input file or software cannot be retrieved from the remote SE (firewalls)
- job executes correctly, output file is there, but cannot be registered
- Monitoring system is down again
- ……
- …

Why is it so difficult to debug

- Peeking at the worker node to find out about your job is a thing of the past!
- the D0 rereco challenge showed the #failure modes for a job!
- some typical constraints
  - worker nodes are on private IP addresses
  - or have no inbound connectivity – active GridFTP sessions will fail
  - may have no outbound connectivity as well
  - e-mail as a monitoring agent does not work (sic) …
- Distributed grid monitoring systems (GMA, R-GMA, a Spitfire DB)
  - this system then has failure modes as well!

Stability, stability, stability

- With many jobs in the system, stability is crucial!
- every job has a producer, the Archiver a consumer for each one, etc., etc., etc. – **and all can fail!**

A Universal Truth:

- a logical service (or server) must not correspond to a process!
- **Example**
  the job manager interface to manage individual jobs
  - If there are 1000 jobs running, and 2000 queued, that's 3000 JMs
  - if you implement it as a process, that means a system load of 3000!
- Processes (all of them) should be stateless
  (also: producers in the IS, GridFTP servers, …)
- state must be in databases or other permanent storage

My job is done!

- Each job generates
  - 2 Gbyte of reconstructed data
  - a log file
  - two python script to update the meta-data catalogue

- *tar* all these together and put them on a SE

  helps to keep the job state consistent

- clean up after the job completes (job takes 5 Gbyte of temp space)

Dealing with output

- A new re-reco run is planned for June ’04
- Store reconstructed DSTs at SARA
- Resource Information schema limitations: cannot handle SE-only

- Copied from a disk cache at NIKHEF to SARA via edg-rm
- registered in SAM with external location **`gsiftp://teras.sara.nl/`**
- write duty-cycle problems:
  - drives can stream 30 MByte/s
  - average data rate from D0 is below this limit
  - but we wrote 2 weeks of production in 1 day at 100 Mbyte/s…

It Worked!

```
tbn08:bomb> ./testsub.py
INSERT INTO d0jsu4 ( jobID,submit_time,ui_node,submitter ) VALUES
('https://boszwijn.nikhef.nl:9000/wRg7tJLmPdX4lePDJ3_0IQ',1074864908,'tbn08.nikhef
.nl','/O=dutchgrid/O=users/O=nikhef/CN=Jeff Templon (dzero)' )
INSERT INTO d0jsu4 ( jobID,submit_time,ui_node,submitter ) VALUES
('https://boszwijn.nikhef.nl:9000/B630yPseZrOwbyIC3hKcuw',1074864920,'tbn08.nikhef
.nl','/O=dutchgrid/O=users/O=nikhef/CN=Jeff Templon (dzero)' )
INSERT INTO d0jsu4 ( jobID,submit_time,ui_node,submitter ) VALUES …
```

```
tbn08:~> edg-rgma –c "history
select d0jen4.cpu_time,d0jen4.wall_time,d0jen4.success_code,d0jen4.out_lfn
FROM d0jsu4,d0jst4,d0jen4 WHERE d0jsu4.submit_time>1071787660 AND
d0jst4.jobid=d0jsu4.jobid AND d0jen4.jihash=d0jst4.jihash“


+----------+-----------+------------------+---------------------------+
| cpu_time | wall_time | success_code     |out_lfn                    |
+----------+-----------+------------------+---------------------------+
| 51291    | 57428     | Job completed OK |lfn:reco_all_…55625.tar.gz |
+----------+-----------+------------------+---------------------------+
1 Rows in set
```

Fermi News

February 2004 issue

DZero

BREAKS NEW GROUND in global computing efforts

First steps toward Grid application with 'real data'

ON THE WEB:

Reprocessing of DZero Run II data:
www-d0.fnal.gov/computing/reprocessing

by Kurt Riesselmann

Wuppertal's landmark, the elevated train line

"In the past, particle physics collaborations have used remote computing sites to carry out Monte Carlo simulations. We are now one of the first experiments to process real data at remote sites. The effort has opened up many new computing resources. The evaluation of our experience will provide valuable input to the Grid development."
– Daniel Wicke, University of Wuppertal, Germany

Tower Bridge, London

Street scene in Lyon

"We've participated in large-scale Monte Carlo production in the past, but data reprocessing involves large volumes of data to be transferred in both directions on a scale that was simply unthinkable a few years ago. It will open new possibilities that we are only beginning to explore."
– Patrice Lebrun (right), with Tibor Kurca, CCIN2P3, Lyon, France

Amsterdam, famous for its canals

"The re-processing was a major milestone for DZero. For us it is also important that we have been able to show that we can really use the LHC Computing Grid for DZero processing. We saw jobs submitted from Wuppertal being executed on our CPUs, and we executed jobs in Karlsruhe, at Rutherford Appleton Laboratory and a few more places."
– Kors Bos (front row, second from left) and the Scientific Computing team at NIKHEF, Amsterdam, Netherlands

Conclusions

- Grids can be used for real scientific work
- We know the core components, and have working prototypes
- **Stability is the key to success**
  - no state in processes
  - no single point of information collection
  - resilient to misconfiguration
- Many interesting research topics remain, e.g.
  - push vs pull for RBs, or a super-node model, or proximity
  - resource virtualisation
  - how to make your job feel "at home" – *or* equip it like a paratrooper!
  - advance reservation
  - load balancing
  - representation of resources (ETT estimators, scalable information systems)
  - authorisation (per-user, per-VO, both in a push or pull model)
  - …

What more is there to see and do?

The current Grids are only the beginning:

- **more functionality, better resilience, strong reliability**
- **joining the Grid will be as simple as joining a file-sharing network**
- **more research, more deployment, more fun!**

The EU DataGrid project	*www.edg.org*
The LHC Computing Grid	*www.cern.ch/lcg*
European Grid Infrastructures	*www.eu-egee.org*, *www.deisa.org*
DutchGrid Platform	*www.dutchgrid.nl*
Grid Forum Nederland	*gridforum.nl*
For other grid projects, see	*www.gridstart.org* *www.enterthegrid.com*