# E-science Applications on the Grid

*Fotis Georgatos <fotis@cern.ch>*
*ETHZ / CSCS*

*June 29th- July 4th, 2009*
*SZTAKI Summer Course 2009, Budapest, Hungary*

www.eu-egee.org

Information Society and Media

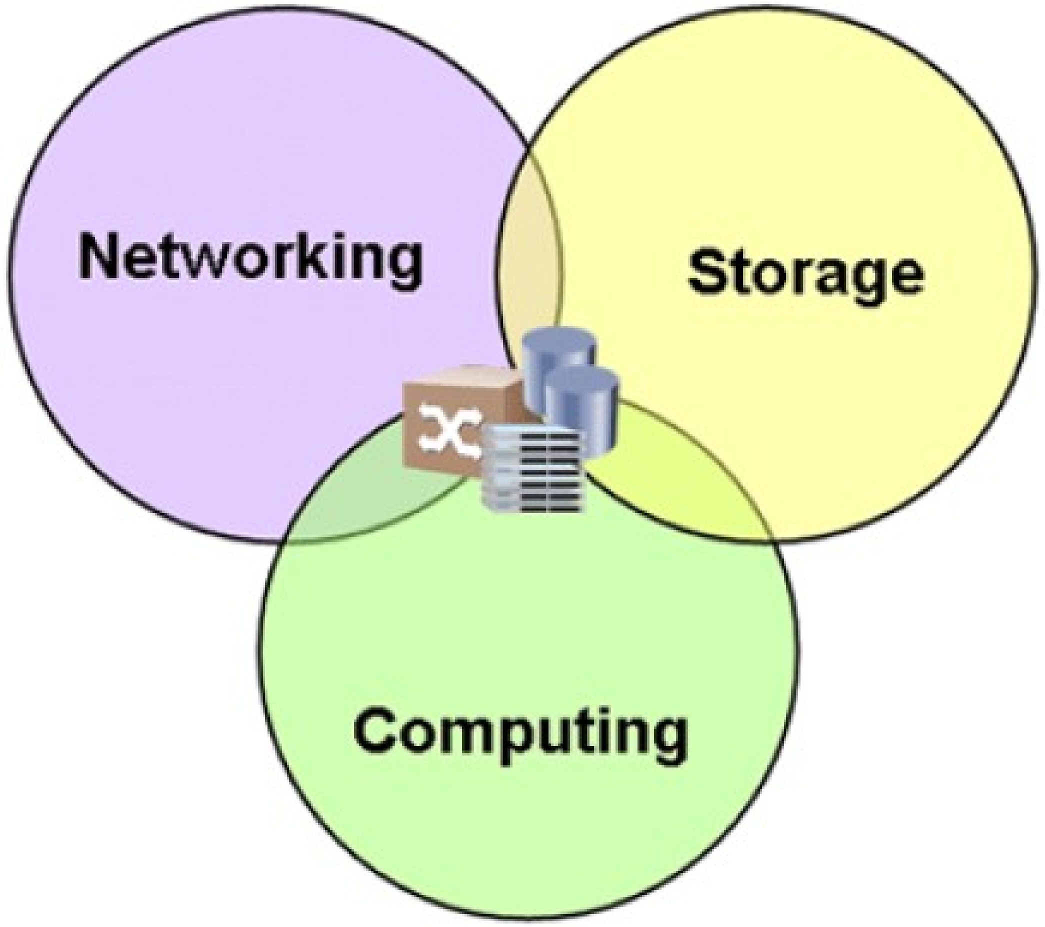
## Acknowledgments

- **Portion of slides (derived from those) prepared by:**
  - Mike Mineter, NESC
  - Charles Loomis, LAL-Orsay
  - Roberto Barbera and his GILDA team
    University of Catania and INFN
  - Colleagues from CSCS, ETHZ or GRNET
  - EGEE-II NA2/NA3/NA4 Activity Members

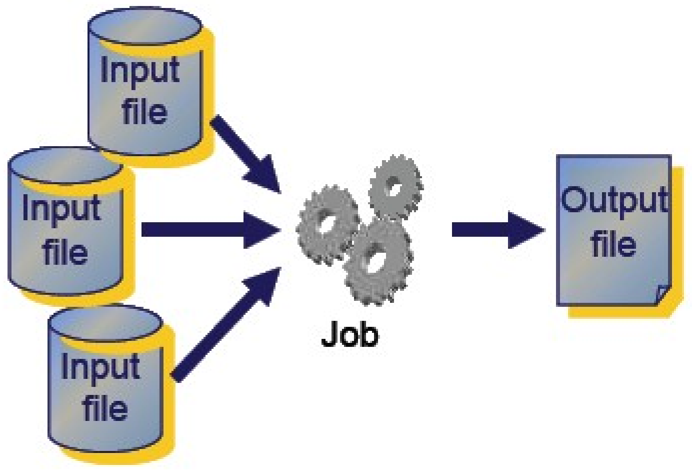
## Contents

- **Basics; the necessary concepts**
- **Workflows and parametric jobs**
- **Portals and job submission tools**
- **Credential management**
- **AMGA & Metadata management**
  - A demonstration of a UNOSAT application
- **Trade-offs and special topics**
  - Performance Optimization
  - Security vs Performance
  - Information System consistency
  - Checkpointing
- **Other resources that could be useful**

## Basics: what grid provides



# Basics: grid job structure



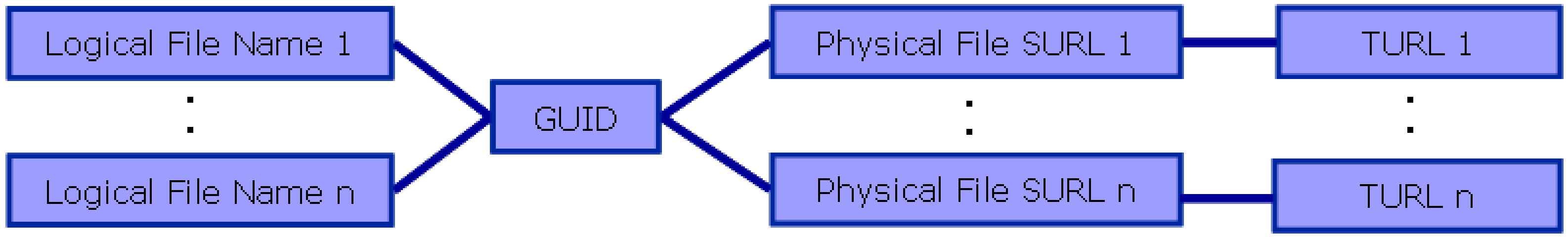
The user asks for the execution of an application at a remote system. It is typically described with a “.jdl”. The application handles input data (that perhaps exist on the Grid already) and produces some output data, which are also stored on the Grid, quite likely on the Storage Element or the UI itself.

## Basics: grid architecture

- **UI: User Interface**
  - This is the system through which a User submits her jobs
  - It might be any Linux system, including a portable notebook.
- **WMS: Workload Management System (or RB)**
  - This is the node where User’s jobs arrive, before being sent to a CE
- **CE: Computing Element**
  - This is the front-end node that provides access to a set of WNs
- **WN: Worker Node**
  - These nodes provide the computational services to the Grid
- **SE: Storage Element**
  - These nodes provide access to disk and tape subsystems
- **LFC: LHC File Catalog**
  - Technically it is the file index, the “parent” of all Storage Elements.
- **BDII: Berkeley Database Information Index (IS)**
  - It provides the information «Where/who/what is the Grid»

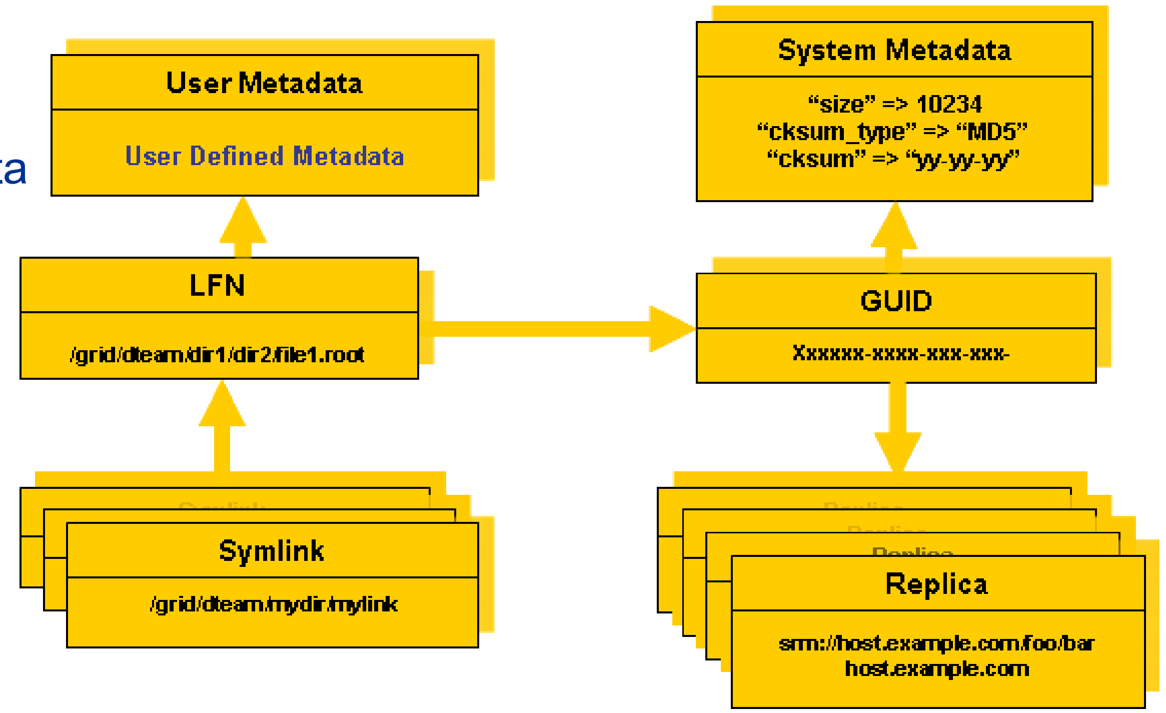
## Basics: Storage architecture

- 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.
    "gsiftp://lxshare0209.cern.ch//data/alice/ntuples.dat"



# Basics: LFC, Logical File Catalog

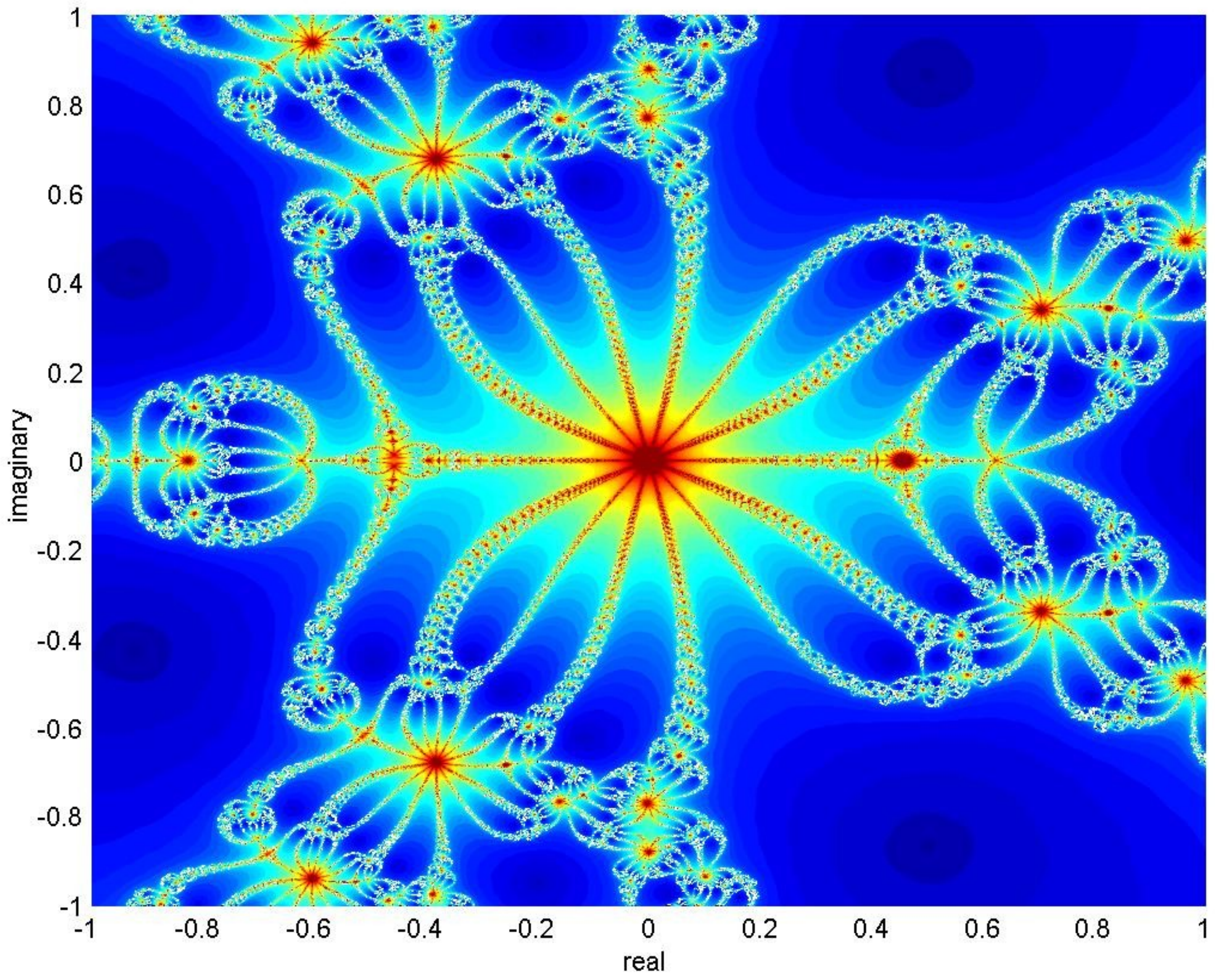
- **Manages the identification, sharing and replication of data in the gLite Grid.**
- **LFN acts as main key in the database. It has:**
  - Symbolic links to it (additional LFNs)
  - Unique Identifier (GUID)
  - System metadata
  - Information on replicas
  - One field of user metadata



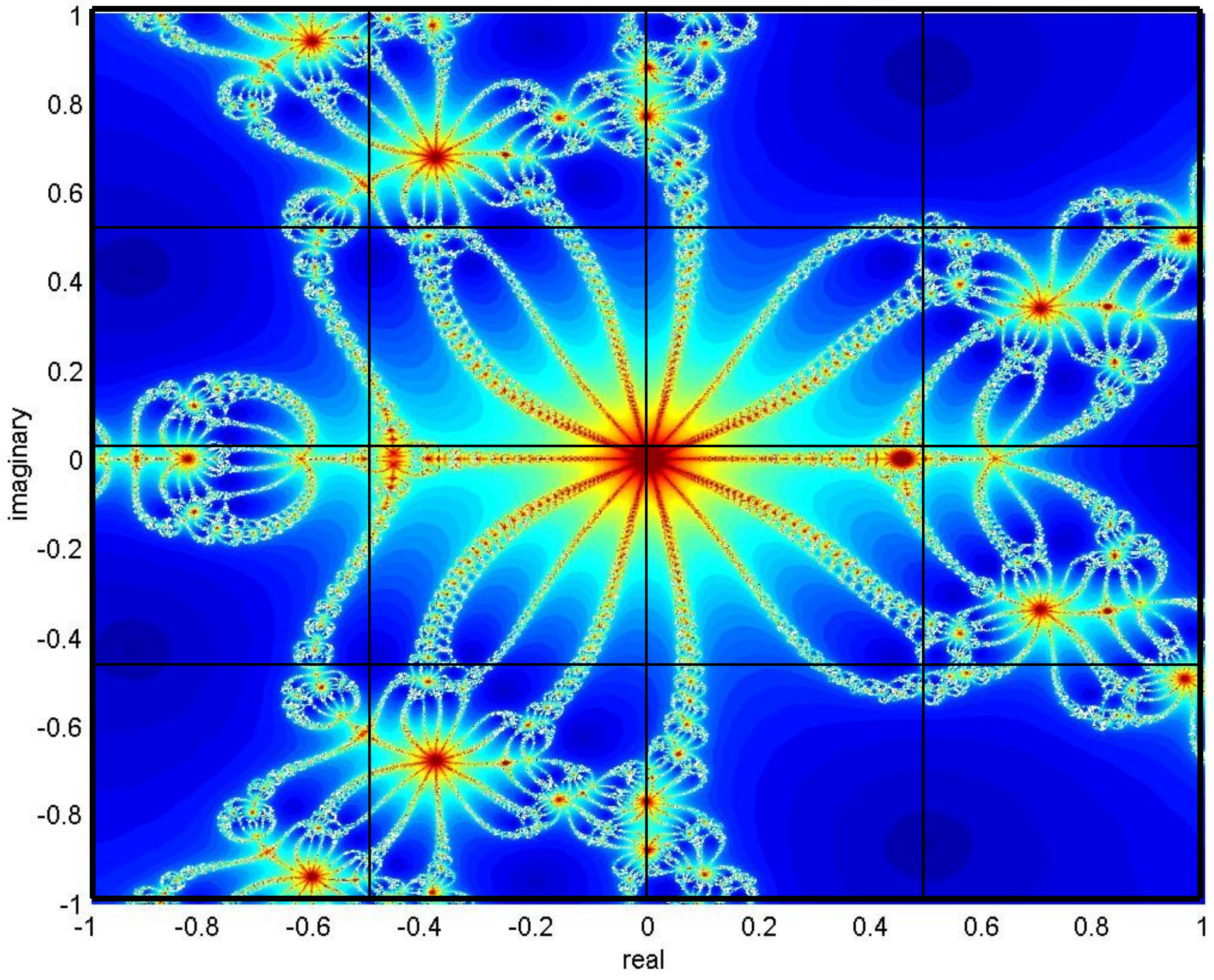
## Grid: Applications to run on

- **Simulations (CPU intensive, need lots of CPUs)**
- **Bulk data processing (lots of storage, lots of CPU)**
- **Parallel jobs (typically a few MPI supporting clusters)**
  - MPI fully supported in many (huge) clusters
  - Interconnects are typically Gigabit, but some sites have Myrinet
  - Constellation-like calculations should be possible in the future
  - Many Grid sites are designed with Gigabit provision e2e.
- **We'd like to see: workflows of ~medigrid applications**
  - Weather models, Fire behaviour, Flooding, Landslides etc
  - Anything with direct or indirect impact on level-of-living
- **We'd like to see: Scavenged resources & Applications**
  - We can do that, Desktop Grids prove it is technically possible
  - Needs lots of experimentation on resource & job matchmaking
  - Needs lots of evolution in the Information System (Glue Schema)

# Workflows and parametric jobs



## Workflows and parametric jobs

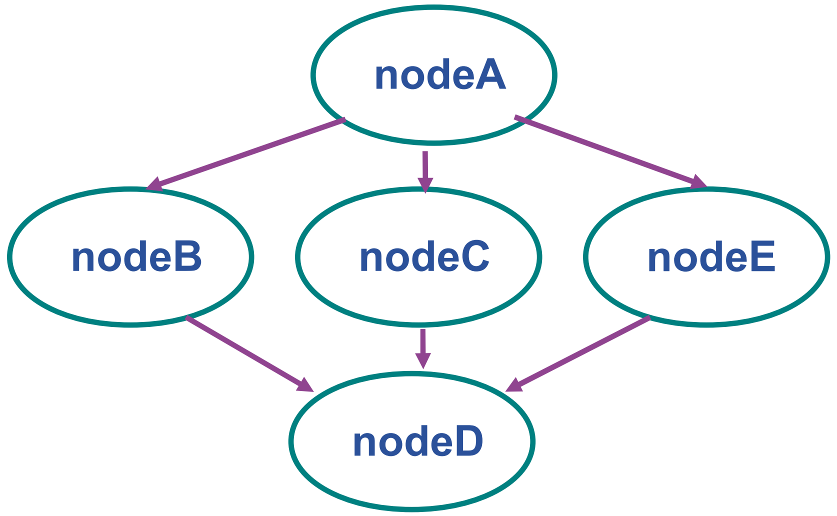


## Workflows and parametric jobs

- Workflow is a set of related jobs, whose execution results are interdepended, either because they have to finish concurrently or, because the output of one job is input for another -in a tree or chain setup- etc
- They are also known as DAGs, standing for Directed Acyclic Graphs, which is indicating their structure
- A flat workflow is the common case of embarrassingly parallel tasks, which are the most trivial grid usage
- Flat workflows can often be described as parametric jobs which means they are defined in domain (1..N).
- Enumerable sets from mathematical set theory would help you understand if your application is parametric; A surprisingly high number of applications are such!
-

# Workflows and parametric jobs

- **Direct Acyclic Graph (DAG) is a set of jobs where the input, output, or execution of one or more jobs depends on one or more other jobs**
- **A Collection is a group of jobs with no dependencies**
  - basically a collection of JDL's

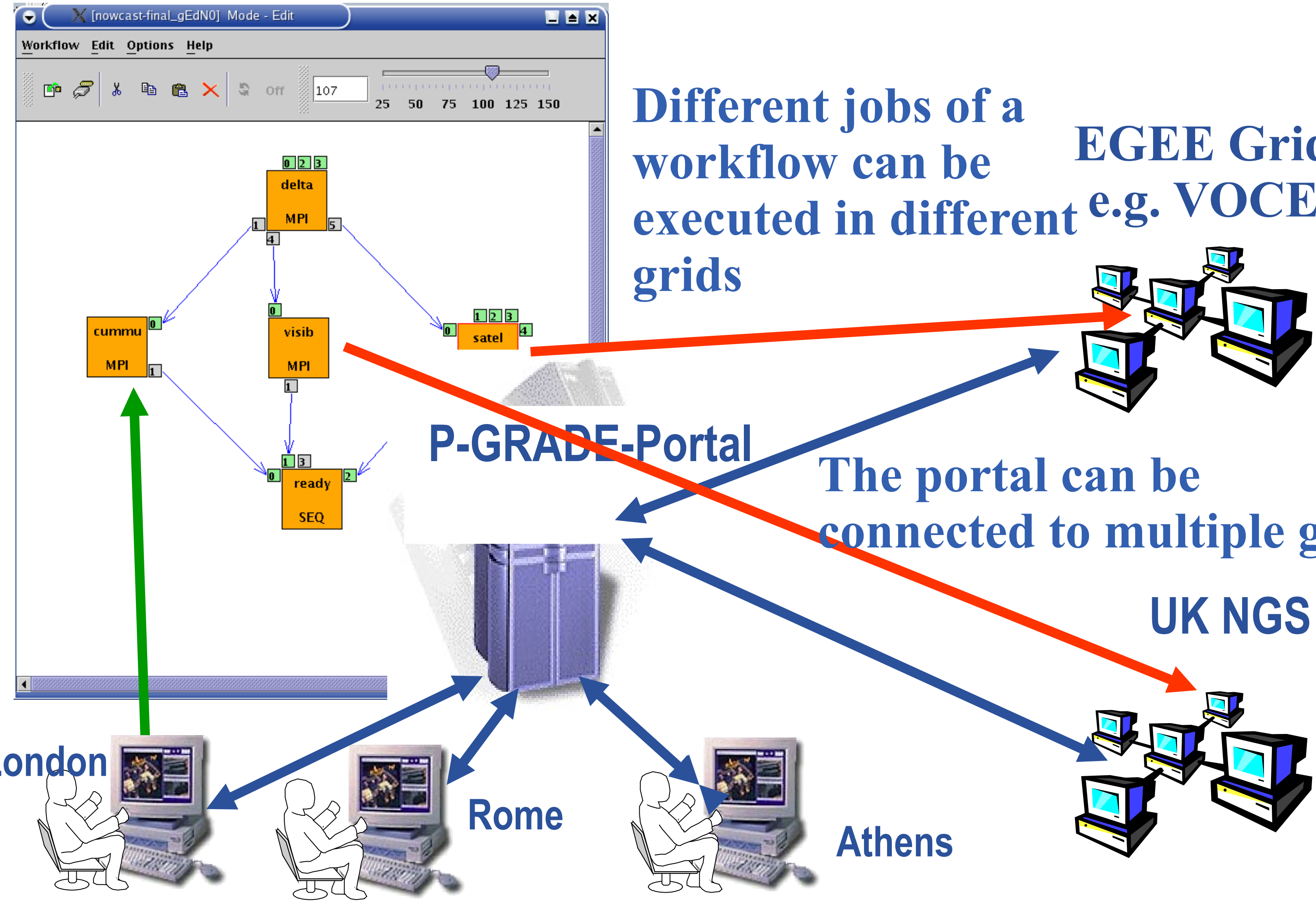


- **A Parametric job is a job having one or more attributes in the JDL that vary their values according to parameters**
- **Using compound jobs it is possible to have one shot submission of a (possibly very large, up to thousands) group of jobs**
  - Submission time reduction
    - Single call to WMProxy server
    - Single Authentication and Authorization process
    - Sharing of files between jobs
  - Availability of both a single Job Id to manage the group as a whole and an Id for each single job in the group

## Portals and job submission tools

- gLite WMS includes a special facility for execution of DAGs and parametric jobs, which make the life of a developer much easier and turn the grid into a powerful automatic tool working on the background.
- You will either have to execute your workflows with your own custom scripts under Unix or, use one of:
  - P-Grade portal, which can be used as a workflow design and management environment; it is being setup in Patras University
  - Genius portal from INFN/Catania provides similar functionality as above, but appears somewhat more closed project
  - Ganga is a job submission framework widely accepted within the LHC VOs. Panda is being added to it for ease-of-use.
  - Taverna is a workflow design environment, language AND enactment engine from BIOMEDICAL community people
  - ...there are more, we find a new one every month!

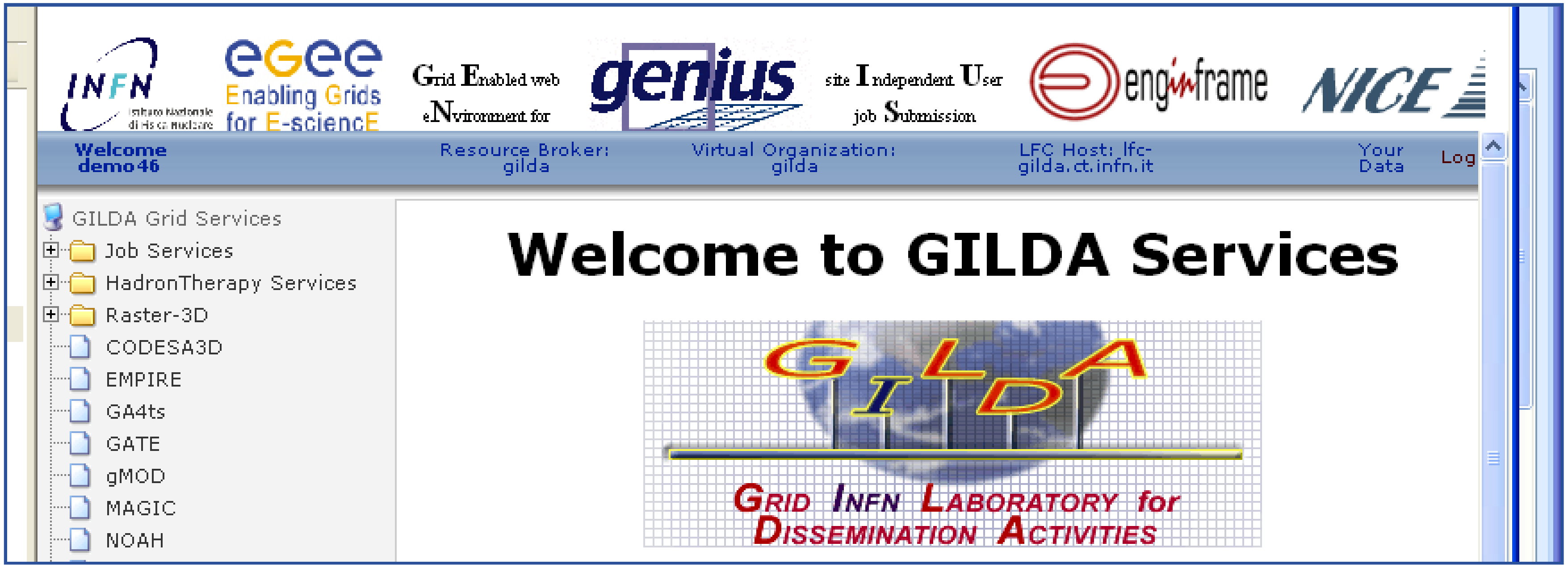
## P-Grade portal



Different jobs of a workflow can be executed in different grids

The portal can be connected to multiple grids

# GENIUS



Welcome demo46 | Resource Broker: gilda | Virtual Organization: gilda | LFC Host: lfc-gilda.ct.infn.it | Your Data | Log

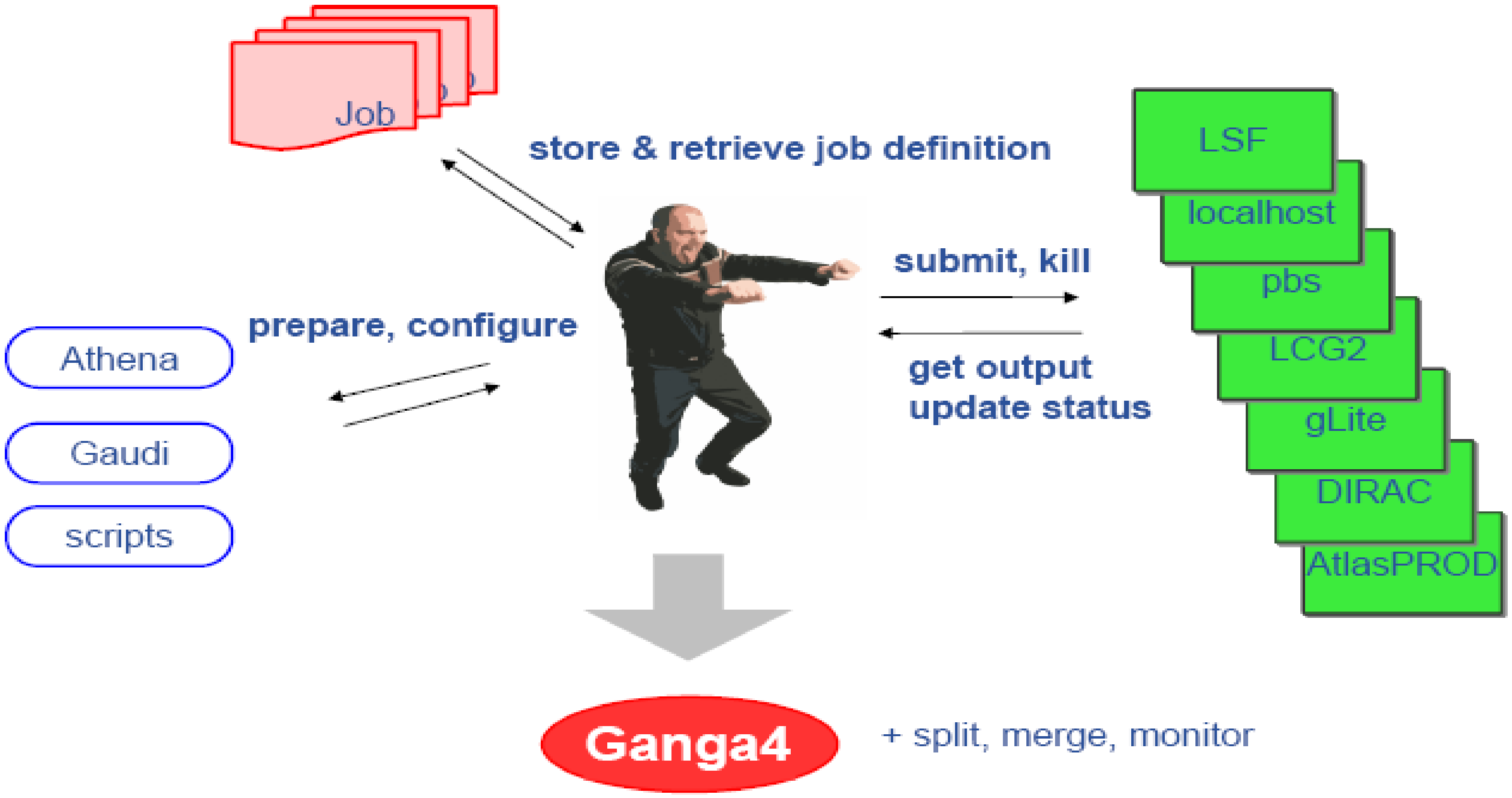
GILDA Grid Services
- Job Services
- HadronTherapy Services
- Raster-3D
- CODESA3D
- EMPIRE
- GA4ts
- GATE
- gMOD
- MAGIC
- NOAH

Welcome to GILDA Services

INFN
Istituto Nazionale di Fisica Nucleare
egee
Enabling Grids for E-sciencE
Grid Enabled web eNvironment for site Independent User job Submission
genius
enginframe
NICE
GILDA
Grid INFN Laboratory for Dissemination Activities

- **For many application communities**
  - Interface can be tailored for specific requirements; mostly a “portal”
- **For demonstration purposes**
  - https://glite-demo.ct.infn.it/
    - Available for anyone to use
  - https://glite-tutor.ct.infn.it/
    - Fuller functionality for users who have stored long-lived proxy in MyProxy server

# Ganga



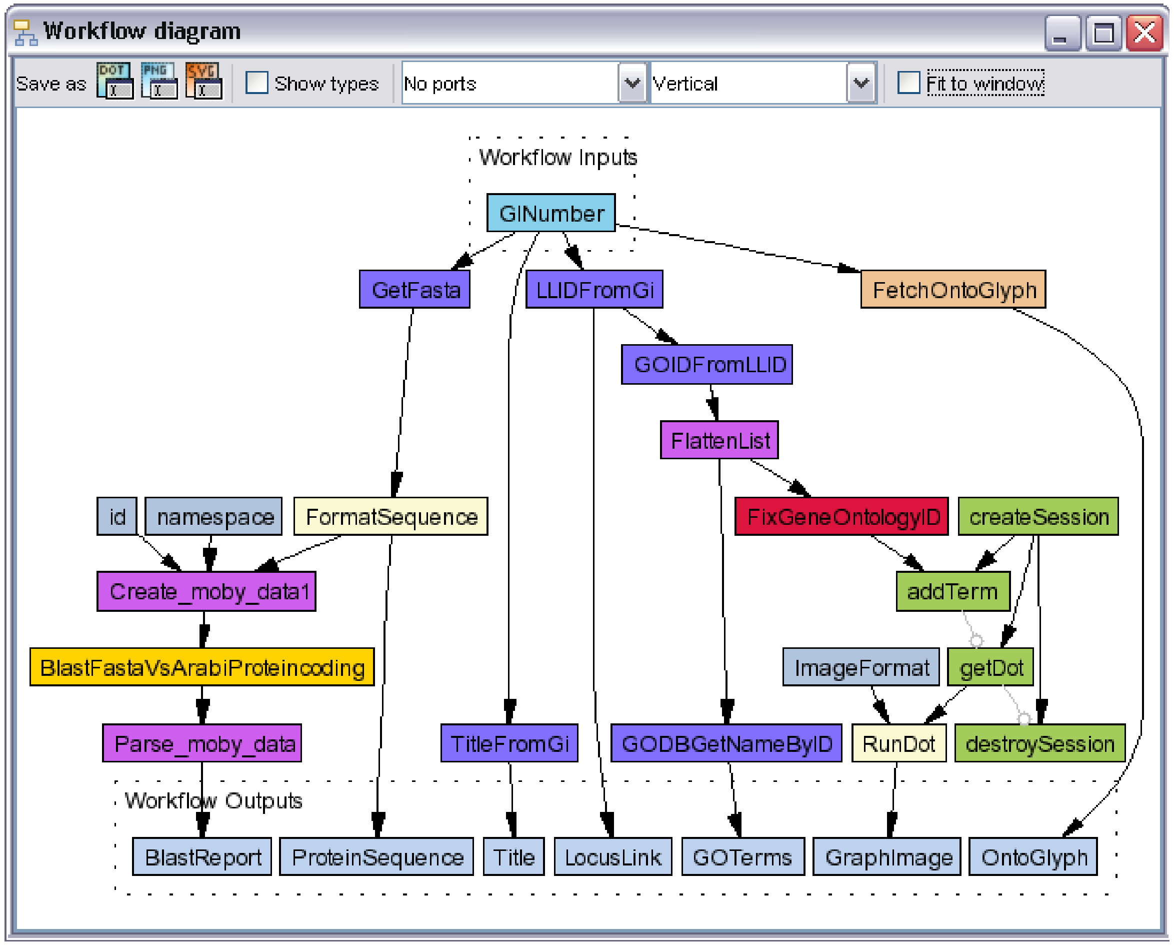
- **Ganga is a lightweight user tool**

**ganga**.web.cern.ch/

- **But also: Ganga is a developer framework**

## Taverna

- Taverna in MyGrid http://www.mygrid.org.uk/
- “allows the e-Scientist to describe and enact their experimental processes in a structured, repeatable and verifiable way”
- GUI
- Workflow language
- Enactment engine



## Credential management

- Credential is a token you use for authentication; typically on a grid this is a proxy X.509 certificate

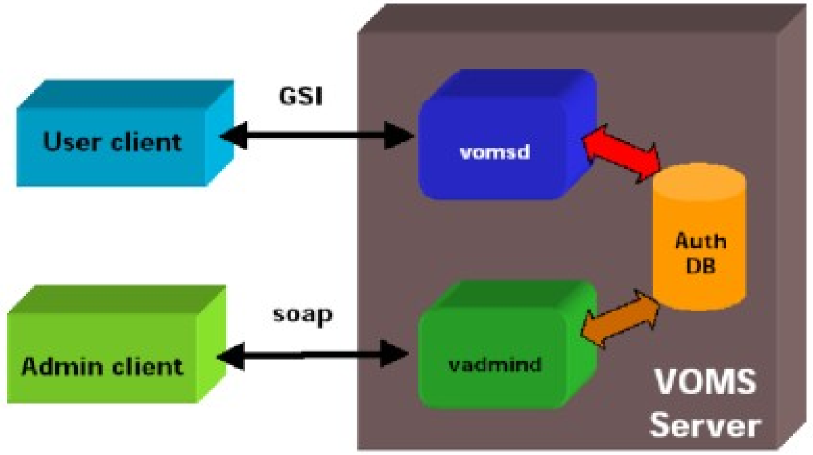
- Every VO provides a VOMS service where your certificate should be listed, once you are a member

- MyProxy is a service of online credential repository
- MyProxy is necessary for credential renewal which is particularly useful for long-running jobs
- It is a much safer storage for long-term certificates

Ref: http://grid.ncsa.uiuc.edu/myproxy/ (++)

## Virtual Organization Membership Service

- Multiple VOs
- Multiple roles in VO
  - compatible X509 extensions
  - signed by VOMS server
- Web admin interface
- Supports MyProxy



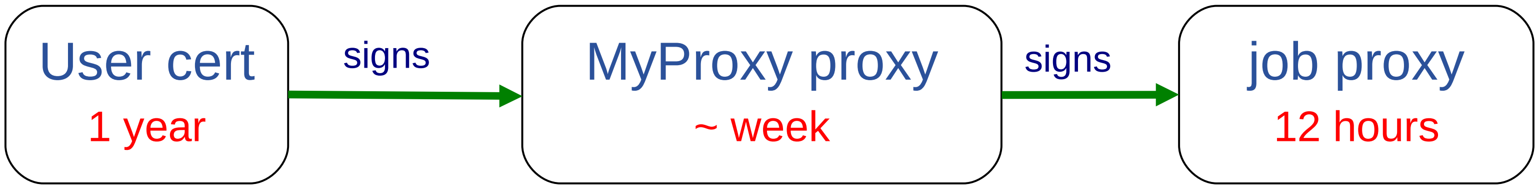
- Resource providers grant access to VOs or roles
- Sites map VO members/roles to local auth mechanism (unix user accounts)
  - allows for local policy

Layer of abstraction: individual members irrelevant

# MyProxy

MyProxy...

- ...allows longer lived jobs / increases security
  - WMS renews proxy
  - users should not produce long lived proxies :-)
- ...allows for secure user mobility
  - user does not need to copy globus-keys around
- ...stores medium-lived proxy (days ~ weeks)



## AMGA & Metadata management

- **AMGA is a Metadata service for gLite-based grids**
- **Metadata is data about data**
- **On the Grid: information about files**
  - Describe files
  - Locate files based on their contents, categories, keywords, etc
- **But also we need simplified DB access on the Grid**
  - Many Grid applications need structured data
  - Many applications require only simple schemas
    - Can be modelled as metadata
  - Main advantage: better integration with the Grid environment
    - Metadata Service is a Grid component
    - **Grid security**
    - Hide DB heterogeneity

## AMGA & Metadata management

- **AMGA – ARDA Metadata Grid Application**
  - ARDA: A Realisation of Distributed Analysis for LHC
- **Now part of gLite middleware**
  - Official Metadata Service for EGEE
  - Also available as standalone component
- **Expanding user community**
  - HEP, Biomed, UNOSAT…
  - Nearly anyone who needs an RDBMS solution

## AMGA & Metadata management

- AMGA provides access to metadata for files stored on the Grid, as well as a simplified general access to relational data stored in database systems
- AMGA is a service in front of a relational database back-end. Currently Oracle, PostgreSQL, MySQL and SQLite are supported
- AMGA has two different front-ends: a web service front-end implemented using gSOAP and a text-protocol (TCP, SSL) based front-end, which is able to stream data back to the client

# AMGA & Metadata management

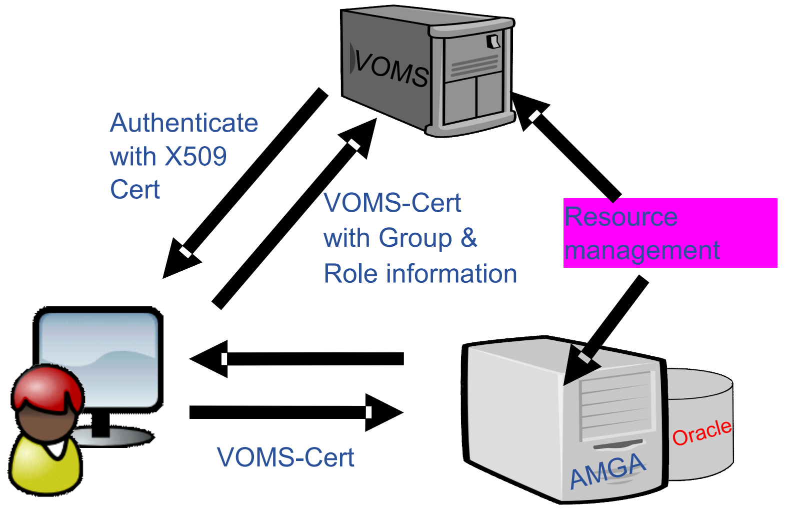
- **Some Concepts**
  - Metadata - List of attributes associated with entries
  - Attribute – key/value pair with type information
    - Type – The type (int, float, string,…)
    - Name/Key – The name of the attribute
    - Value - Value of an entry's attribute
  - Schema – A set of attributes
  - Collection – A set of entries associated with a schema
  - Think of schemas as tables, attributes as columns, entries as rows

## AMGA & Metadata management

- **LHCb-bookkeeping examples**
  - Migrated bookkeeping metadata to ARDA prototype
    - 20M entries, 15 GB
    - Large amount of static metadata
  - Feedback valuable in improving interface and fixing bugs
  - AMGA showing good scalability
- **Ganga is using it, too**
  - Job management system
    - Developed jointly by ATLAS and LHCb VOs
  - Uses AMGA for storing information about job status
    - Small amount of highly dynamic metadata

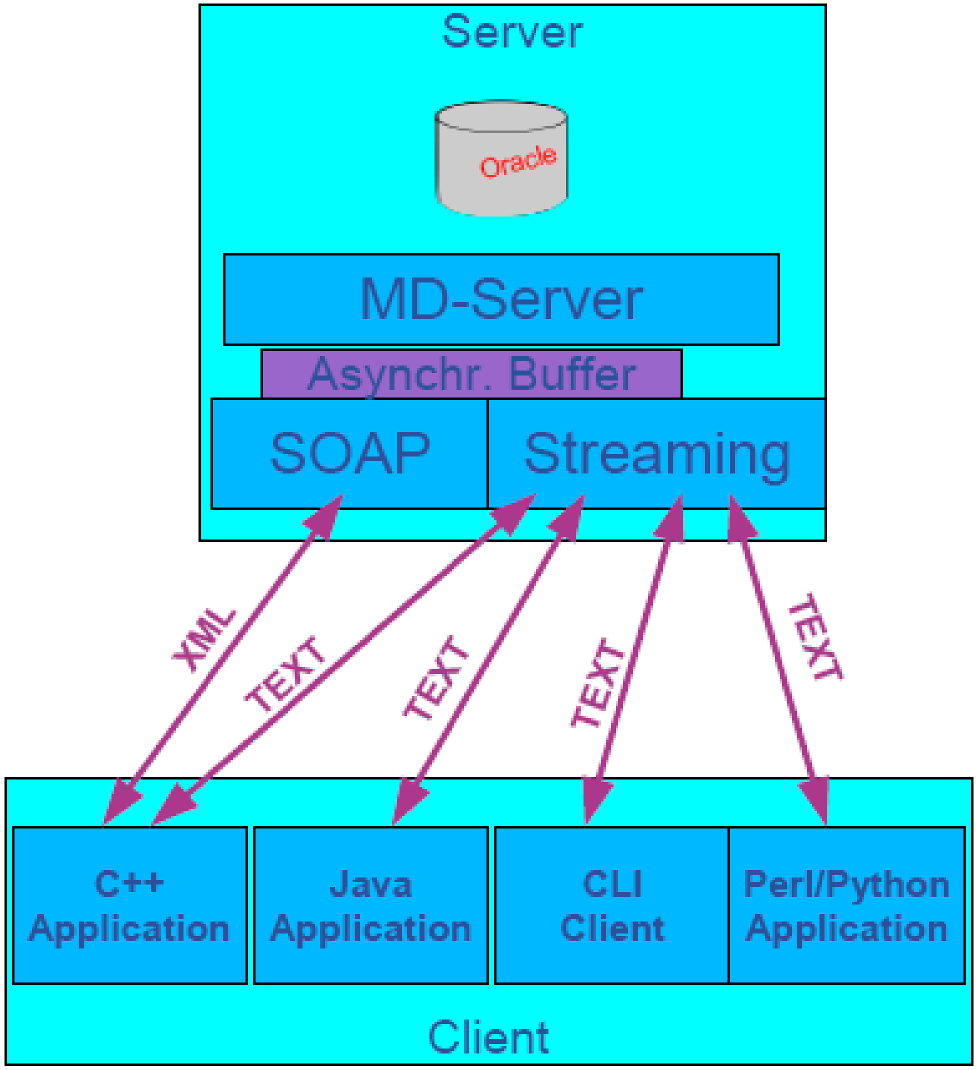
## AMGA Security

- Unix style permissions
- ACLs – Per-collection or per-entry.
- Secure connections – SSL
- Client Authentication based on
  - Username/password
  - General X509 certificates
  - Grid-proxy certificates
- Access control via a Virtual Organization Management System (VOMS):


VOMS
Authenticate with X509 Cert
VOMS-Cert with Group & Role information
Resource management
VOMS-Cert
AMGA
Oracle

# AMGA Implementation

- **AMGA Implementation:**
  - SOAP and Text frontends
  - Streamed Bulk Operations
  - Supports single calls, sessions & connections
  - SSL security with grid certs (negociated by client)
  - Own User & Group management + VOMS
  - PostgreSQL, Oracle, MySQL, SQLite backends
  - Works alongside LFC
  - C++, Java, Perl, Python clients



## AMGA Features

- **Dynamic Schemas**
  - Schemas can be modified at runtime by client
    - Create, delete schemas
    - Add, remove attributes
- **Metadata organised as an hierarchy**
  - Schemas can contain sub-schemas
  - Analogy to file system:
    - Schema ⇔ Directory; Entry ⇔ File
- **Flexible Queries**
  - SQL-like query language
  - Joins between schemas

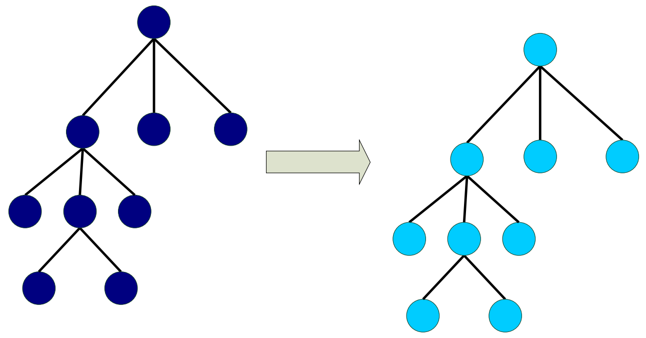
## Metadata Replication

- **Currently working on replication/federation mechanisms for AMGA**
- **Motivation**
  - Scalability – Support hundreds/thousands of concurrent users
  - Geographical distribution – Hide network latency
  - Reliability – No single point of failure
  - DB Independent replication – Heterogeneous DB systems
  - Disconnected computing – Off-line access (laptops)
- **Architecture**
  - Asynchronous replication
  - Master-slave – Writes only allowed on the master
  - Replication at the application level
    - Replicate Metadata commands, not SQL → DB independence
  - Partial replication – supports replication of only sub-trees of the metadata hierarchy

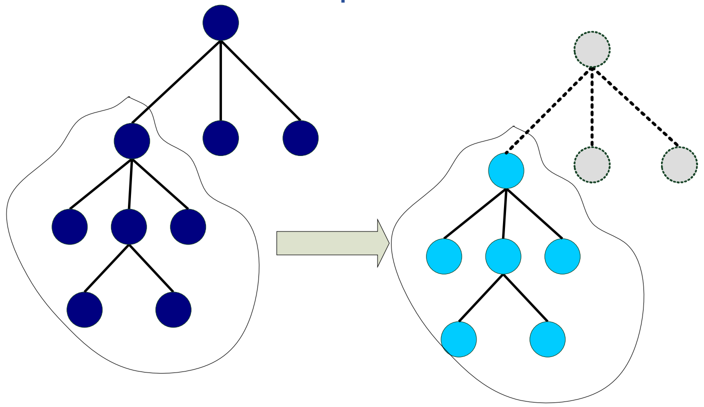
# Metadata Replication

## Some use cases

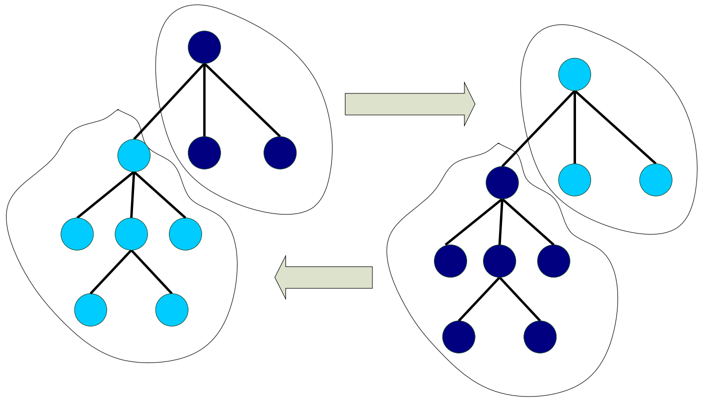
Full replication



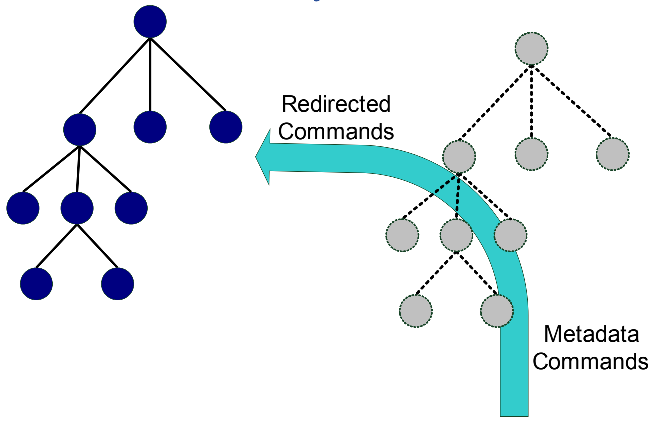
Partial replication



Federation



Proxy

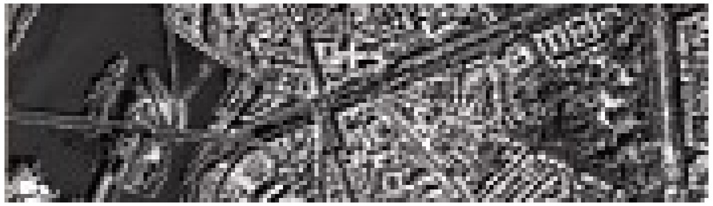


## AMGA: UNOSAT Presentation

### UNOSAT is a United Nations Initiative

- Objectives
  - Provide the humanitarian community with access to satellite imagery and Geographic Information System services
    - Reduce disasters and plan sustainable development
  - Ensure cost-effective and timely products
- Core Services
  - Humanitarian Mapping
  - Image Processing

VEGETATION – 1 Km


IKONOS – 1m

## One step further: GRID

- Potential Bottlenecks:
  - UNOSAT beginning to suffer from limited capacity and processing power
  - Multiple satellites being launched
  - Larger and larger storage capacity needed
- In summer 2005 CERNprovided a whole structure for UNOSAT:
  - UNOSAT Virtual Organization (VO)
  - 3.5TB in CASTOR
  - Computing Elements, Resource Brokers
  - Collaboration with ARDA group
  - AFS area of 5GB

We have provided the whole GRID infrastructure At CERN

- We have run some UNOSAT tests (images compression) inside the GRID environment (quite successful)
- The framework developed for in principle for Geant4 (See Alberto Ribon's presentation) has been adapted for UNOSAT needs

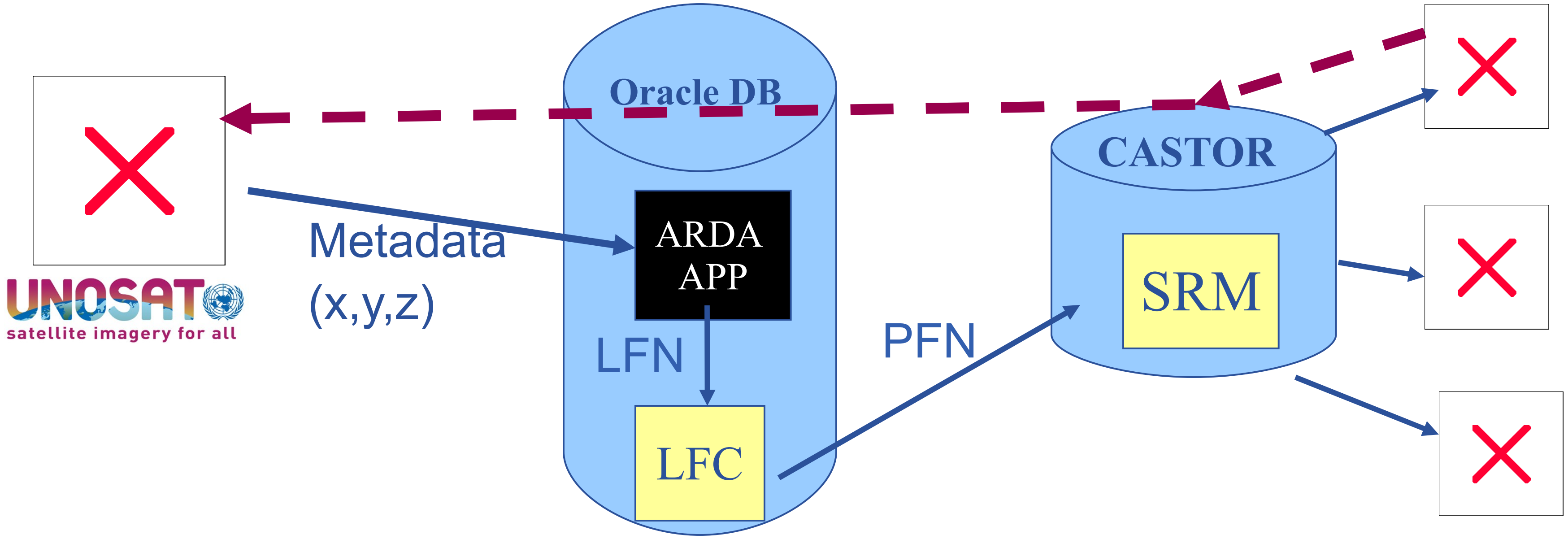
## UNOSAT Production inside GRID

- UNOSAT provided a set of images for testing
- Associated to each image a metadata file was included
  File name, directory path, geographical coordinates
- Steps:
  STORAGE LEVEL
  - Copy and registration of the images in Castor@CERN
    - Use of the LFC Catalog
  - Parse the metadata files to extract the different metadata
  - Use of the AMGA tool to parse metadata to location of the files

  COMPUTING LEVEL
  - Use of compression tools to compress images inside LCG resources
  - Use of the general submission tool adapted to UNOSAT needs

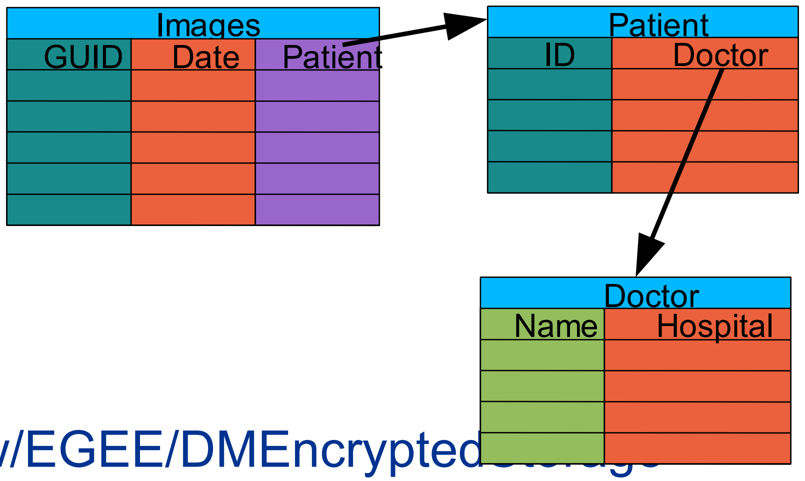
## A GRID Metadata Catalogue

- LFC Catalogue
  - Mapping of LFN to PFN
- UNOSAT requires
  - User will give as input data certain coordinates
  - As output, want the PFN for downloading
- The ARDA Group assists us setting up the AMGA tool for UNOSAT



## AMGA: Biomed

- **Medical Data Manager – MDM**
  - Store and access medical images and associated metadata on the Grid
  - Built on top of gLite 1.5 data management system
  - Demonstrated at past EGEE conference (October 05, Pisa)
- **Strong security requirements**
  - Patient data is sensitive
  - Data must be encrypted
  - Metadata access must be restricted to authorized users
- **AMGA used as metadata server**
  - Demonstrates authentication and encrypted access
  - Used as a simplified DB



- **More details at**
  - https://uimon.cern.ch/twiki/bin/view/EGEE/DMEncryptedStorage

## Further information

- **on AMGA and gLibrary:**
  - http://indico.eu-eela.org/conferenceTimeTable.py?confId=37
  - (go to day 3 for the AMGA tutorial )

- **AMGA Web Site**

  http://project-arda-dev.web.cern.ch/project-arda-dev/metadata/

## Applications to run

- **Simulations (CPU intensive, >1000 of CPU*hours)**
  - CPU-intensive, they have to be profiled & optimized
    - >75000 provided by EGEE/LCG grid infrastructure,
    - Many more if you can tap Desktop Grids in your region
  - HOW: a range of x86 execution environments
    - all are 32bit i386 compatible: IA32, IA64, Athlons, Opterons etc.
    - But, they can vary considerably in performance
- **Bulk data processing (Storage, >1TB+CPU)**
  - Involve the management of huge datasets
  - Best performance occurs if data is distributed/replicated
  - They might involve workflow and/or meta-data techniques
  - Plus, very similar computing needs as simulations
  - HOW: Create replicas and initiate a distributed workflow
    - There is plenty of Storage; many PBs on EGEE/LCG
    - Create *at least* three replicas of each dataset "slice"

## Applications to run

- **Responsive Applications**
  - Interactive applications have been attempted in the past
  - During CrossGrid there has been success, if Grid was unloaded
  - Short responce time, requires low-latency, QoS
  - In the future we may be able to pause-reanimate jobs on demand
  - Necessary feature to support Urgent Computing (EQ, floods, fires)
  - Middleware has to evolve further, in order to support them well
  - HOW: Not simple, but not impossible. It requires VO coordination.
- **Workflows**
  - It can involve:
    - a GUI to design it
    - a language to represent it
    - Enactment engine
  - Typically used to tackle any sufficiently complex problem
  - HOW: Look for tools like Ganga, Taverna, P-Grade, Genius etc.

## Applications to run

- **Parallel jobs**
  - Interdepended, communicating tasks
  - They have to use an interconnect (hardware) and MPI (software)
  - Very critical to ensure proper MPI libraries are there
  - Applications might have to be recompiled
  - It should be possible (in the future) to do constellation calculations
  - HOWTO: Request support from NA4 activity, ASAP.
- **Legacy applications**
  - Commercial or closed source binaries
  - User must ensure that there is at least one compatible license model, that can be applied on the grid (or request for it!)
  - It is a serious issue that they can't be reengineered for the grid
  - FUSE/ELFI, flexlm somewhere on a central server
  - HOWTO: Try to deploy it (Cross your fingers that it will run OK...)

## Topics: Deploying your application

- **Maybe you are lucky, and the application is there.**
- **Ask the NA4 about it. They are doing this job for you!**
  - If it is Free software+RPMs (like gcc, octave, anything in Quantian DVD) should be easily accepted for evaluation.
- **or, you could find the source code, compile it, and send it upon job submission**
- **If it is a big executable, you can use DM to replicate it.**
- **You could also compile it at run-time on the WN.**
  - This is quite appropriate for eg. MPI programs on Opterons.
- **Since much of the grid has Intel processors, ~75% of the infrastructure, it is best to use the Intel compiler, which might imply having your own Interface. It is a big overhead, but you might even see 100% speedup (*2).**

**Final advice: If you don't know what to do, do what you know, what is simple and works for you.**

## Topics: How consistent is the IS?

- Do clusters have Worker Nodes with...
  - Same OS? Same CPU? Same clockrate? Same RAM? ...
- There is some heterogeneity in the resources provided by a given cluster (eg. slightly different clockrates).
- It is imposed by the combination of hardware diversity, the capabilities of the IS, ie. GLUE schema v1.x, and above all, the exact middleware implementation!
- GLUE schema v2 is a future proposal, which will allow the description of "subclusters", so this issue should be overcome; should be decided within 2007.
- For the time being, ask users to put their minimum job requirements in a .jdl, and if they see an error, to report immediately to Operations (SA1)'s help-desk tools.

## Topics: Why do checkpointing?

- **Check-pointing is the technique of storing the state of your job, typically in a set of files, so that you can later "reanimate" it. It is very critical for “long jobs”.**
- **If jobs take more than 24 hrs, it is imperative for users to consider check-pointing for a number of reasons:**
  – User: software engineering reasons, ie. debugging a checkpointable job is simpler to debug on longer-term issues, since it is possible to restart it on eg. the 6th day of execution
  – User: overcome the limit of grid queues (infinite jobs!) and be able to pause and reanimate the job arbitrarily
  – User: by limiting the execution time of your jobs, you can now tap much more resources (queues), which will decrease the total time
  – Systems administration: cluster downtimes can happen for either scheduled reasons (m/w upgrade, hardware installation or reconfiguration etc) or various unscheduled reasons (power/network/airconditioning outages, security issues)

## Topics: How to do checkpointing

- **There are currently two well-known techniques to implement checkpointing (c/p):**
  - Update some central service eg. some database like AMGA, about the current state of the job (easy, if state==integer)
  - Save the state of the job in a set of files, make a tarball (.tar.gz) & register it on a nearby SE with a predefined LFN!
- **Middleware is supposed to be able to assist, but in gLite v3.0 this is currently unsupported function (?)**
- **In effect, workflow and job management tools (ganga?) do state-maintenance at the large-scale of a gridified application, can this be integrated with checkpointing?**
- **Maybe future tools that are able to do workflow management, will be able to implement this internally.**
- **A known trade-off: high-throughput vs low latency c/p**

## Some more random tips

- Medical applications might want to set-up special encryption or anonymization techniques for their data, eg. Hydra, and perhaps enter negotiations with Sites for Confidentiality Agreements (this won't be easy)
- Security and flexibility are well known trade-offs
- Beware of IEEE 754 arithmetic; compilers and archs
- Beware of architecture-specific optimizations, eg. AMD
- Beware of cache-related performance tuning (BLAS!!!)
- Beware of current and future network evolution (IPv6!)
- Be aware of highly relevant research literature (LSSC...)
- FTS/FPS are at your disposal for Data Management...

## Reference Material

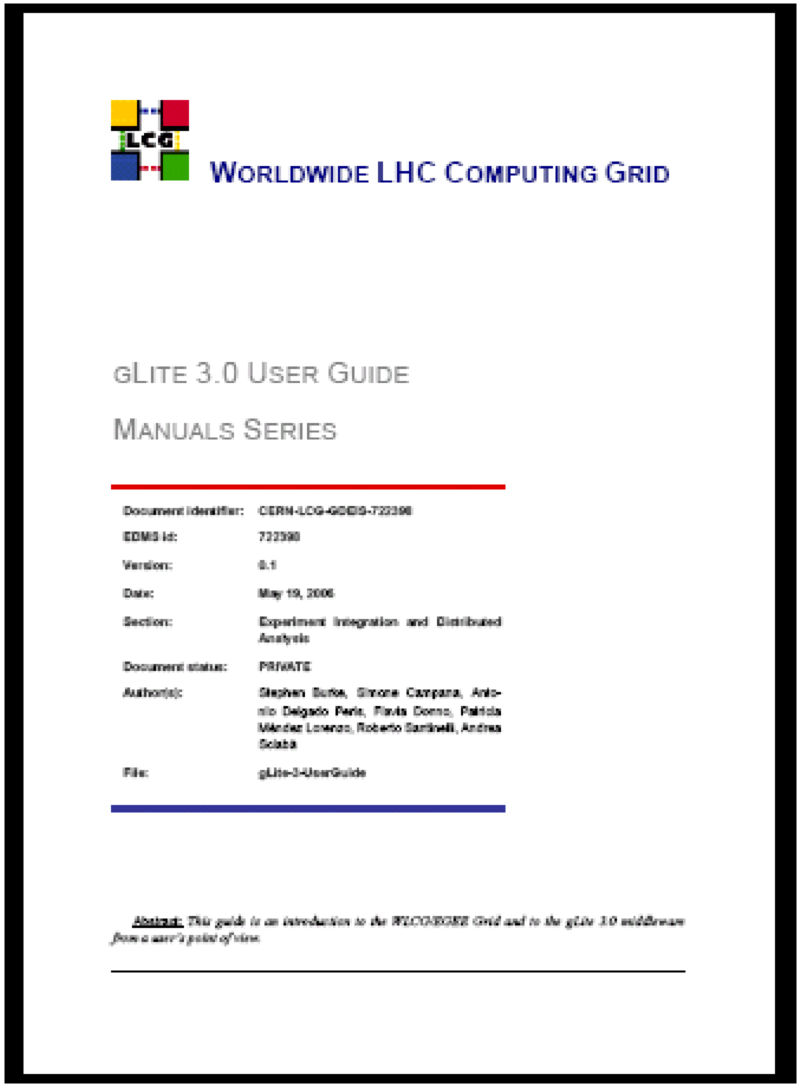
- **There are some important documents to read, in order to learn shell and language (python) scripting:**
  - Unix **Shell Scripting** Tutorial, Ashley J.S Mills, 2002
  - **Advanced Bash-Scripting** Guide; An in-depth exploration of the art of shell scripting, Mendel Cooper, 2004
  - An **Introduction to Python**, David M. Beazley, 2000
  - **Advanced Python** Programming, David M. Beazley, 2000

## GLITE 3.0 USER GUIDE

- **Glite instruction manual, with multiple middleware commands, 166 pages .pdf:**
  - Getting started
  - Information Service
  - Workload Management & .jdl
  - Data Management
  - User tools
- **RTFM!**

**(=Read The Fine Manual!)**

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



WORLDWIDE LHC COMPUTING GRID

GLITE 3.0 USER GUIDE

MANUALS SERIES

| | |
|---|---|
| Document identifier: | CERN-LCG-GDEIS-722398 |
| EDMS id: | 722398 |
| Version: | 0.1 |
| Date: | May 19, 2006 |
| Section: | Experiment Integration and Distributed Analysis |
| Document status: | PRIVATE |
| Author(s): | Stephen Burke, Simone Campana, Antonio Delgado Peris, Flavia Donno, Patricia Méndez Lorenzo, Roberto Santinelli, Andrea Sciabà |
| File: | gLite-3-UserGuide |

***Abstract:*** *This guide is an introduction to the WLCG/EGEE Grid and to the gLite 3.0 middleware from a user's point of view.*

# Who can benefit from the grid?

## Medical/Healthcare
- Imaging
- Diagnonis & Treatment
- Drug design (d2ol, for SARS, anthrax, embola etc)

## Bioinformatics
- Study of the human and other genomes (genome@home)
- Protein folding (folding@home, predictor@home)

## Geological and climate applications
- Weather Forecasting
- Climate Simulation (climate@home)
- Ocean current analysis
- Oil and Gas Exploration
- Seismic Signal Analysis

## Pharmaceutical, Chemical, Biotechnology
- atmospheric chemistry
- systems biology
- materials science
  - material interaction simulations
  - catalysis investigations)
- molecular modeling
- nanotechnology

## Mathematics and Basic Research
- prime numbers (gimps/mprimes effort)
- The (partial) verification of Riemann's Hypothesis

## Business decision support
- Financial analysis
- Portfolio optimization
- Risk management applications
- Route Optimization
  - Transportation
  - LAN and WAN Networking
- Supply Chain and Demand Chain Optimization
- Search and Retrieval (huge databases, data mining)

## Electrical, Mechanical and Civil Engineering
- Energy production and distribution strategy optimization
- Engineering and digital design
- CAD / CAM
- Construction verification against earthquakes
  - eg. finite elements method
- Aerodynamic simulation (wind tunnel simulation)
- Digital Rendering (raytracing, digital video synthesis)

## Physics & Astrophysics
- High Energy Physics simulations and signal analysis
- N-body problem simulation
- space probe signal analysis (einstein@home)
- radio telescope signal analysis (seti@home)

## Computer Science
- Cryptography (distributed.net)
- Search Engines (grud, a distributed Internet crawler)

**Many! http://distributedcomputing.info/distrib-2003/distrib-projects.html**

## The End

**Thank You**

**fotis@cern.ch**