Grid Middleware

Andreas Unterkircher
gLite Release Manager
CERN

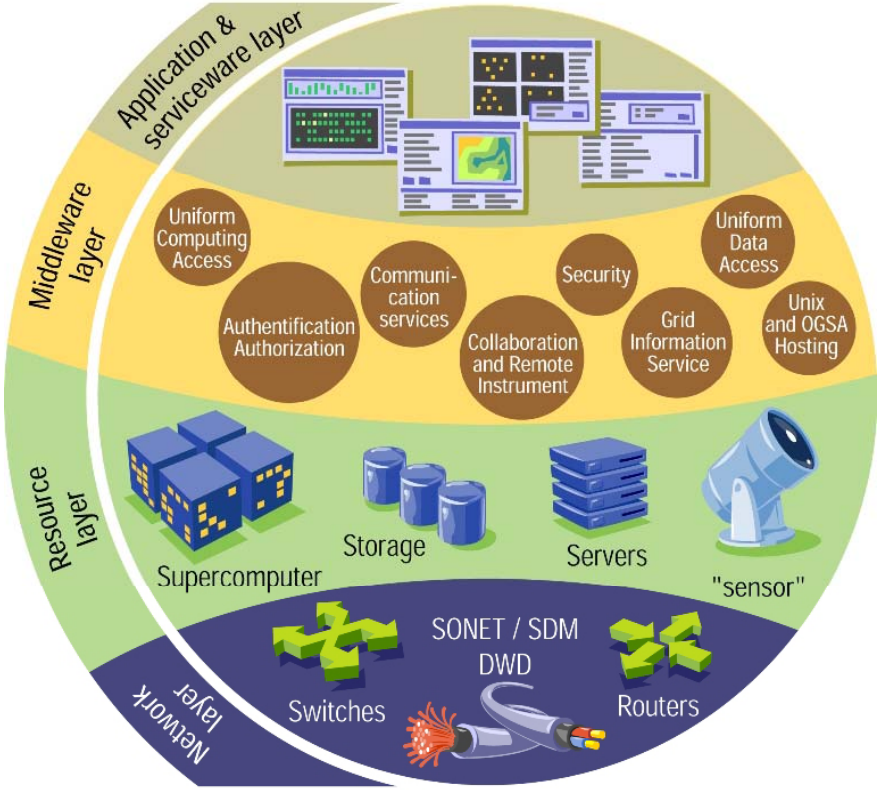
Openlab Summer Student Lecture 2009

www.eu-egee.org

e-infrastructure

Content

- Grid Computing
- Grid Usage
- The gLite middleware
  - http://glite.org/
  - http://www.eu-egee.org/
- Future development



Why does CERN need the Grid?

- Let's ask Les Robertson (Head of WLCG until end of 2008):
- "Computing wasn't included in the original costs of the LHC. […] We clearly required computing, but the original idea was that it could be handled by other people".

Around 2000, these "other people" had not stepped forward.

"There was no funding at CERN or elsewhere. A single organization could never find the money to do it. We realized the system would have to be distributed.

(Interview with Les Robertson, iSGTW Feb. 08)

How to use distributed resources?

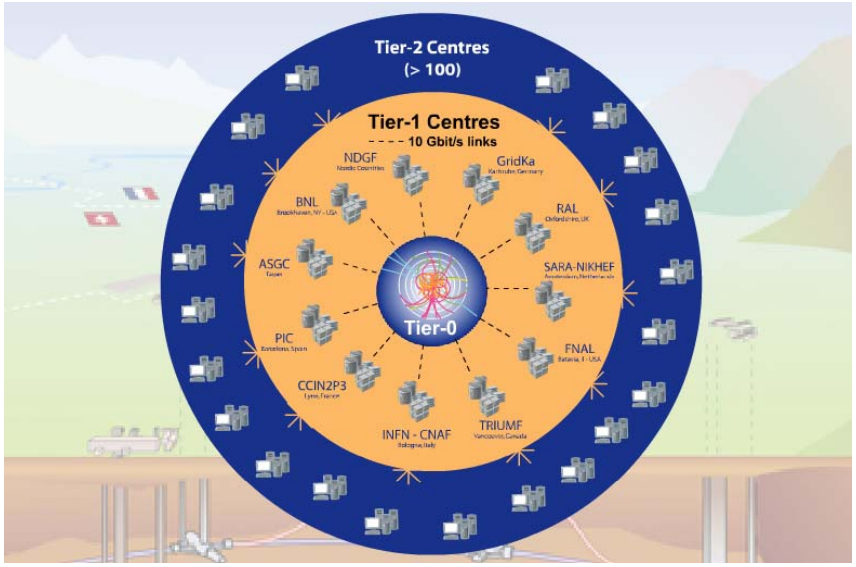
- **Different computing centers provide resources, but**
  - A single user has to ask for access individually at every site
  - Usage rights, disk quotas, compute shares etc. are different at every site
  - Different batch systems (different CLI/API)
  - How do you get your data to every site?

At the end of the 90ies the HEP community initiated the MONARC project to do distributed computing for LHC.

At the same time “Grid Computing” emerged.

Grid Computing

- **All about collaboration and sharing resources**
  - Distributed groups can pool their computing resources
  - Large and small computing centers can contribute
  - Users everywhere can get equal access to data and computing power
  - Exploit funding wherever it is provided



Grid Computing early this century

- **Many Grid infrastructures (using different middleware) have been built, e.g.:**
  - Open Science Grid (USA)
  - Naregi (Japan)
  - DEISA (European Supercomputer Grid)
  - Nordugrid (North European Grid)
  - TeraGrid (USA)
  - EGEE

    World's largest grid infrastructure (June 09 numbers)
    - 17 000 users
    - 139 000 LCPUs
    - 25Pb disk, 39Pb tape
    - 12 million jobs/month
    - 268 sites
    - 48 countries
    - 162 VOs

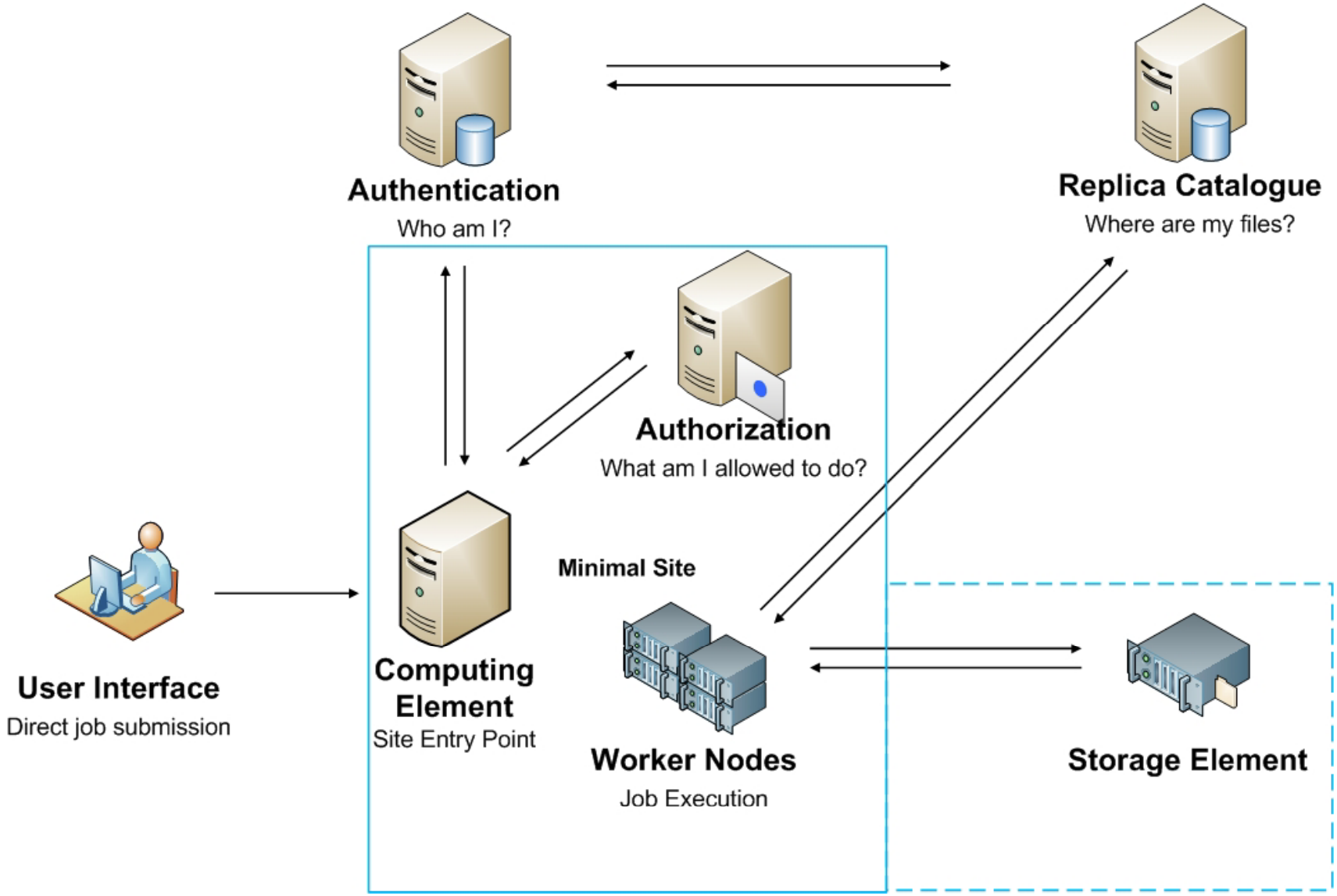
Who uses the Grid?

- **Virtual Organizations (VO)**
  - Researchers form VOs to collaborate, share resources and access common data.
  - Typically a long-lived group of researchers with a common purpose.
    - Each LHC experiment is managed by a single VO (large VOs ~ 2000 members)
    - Single VO can contain several subgroups that collaborate on specific topics but actually have little interaction with one another (e.g. biomedicine)
  - EGEE supports >150 VOs

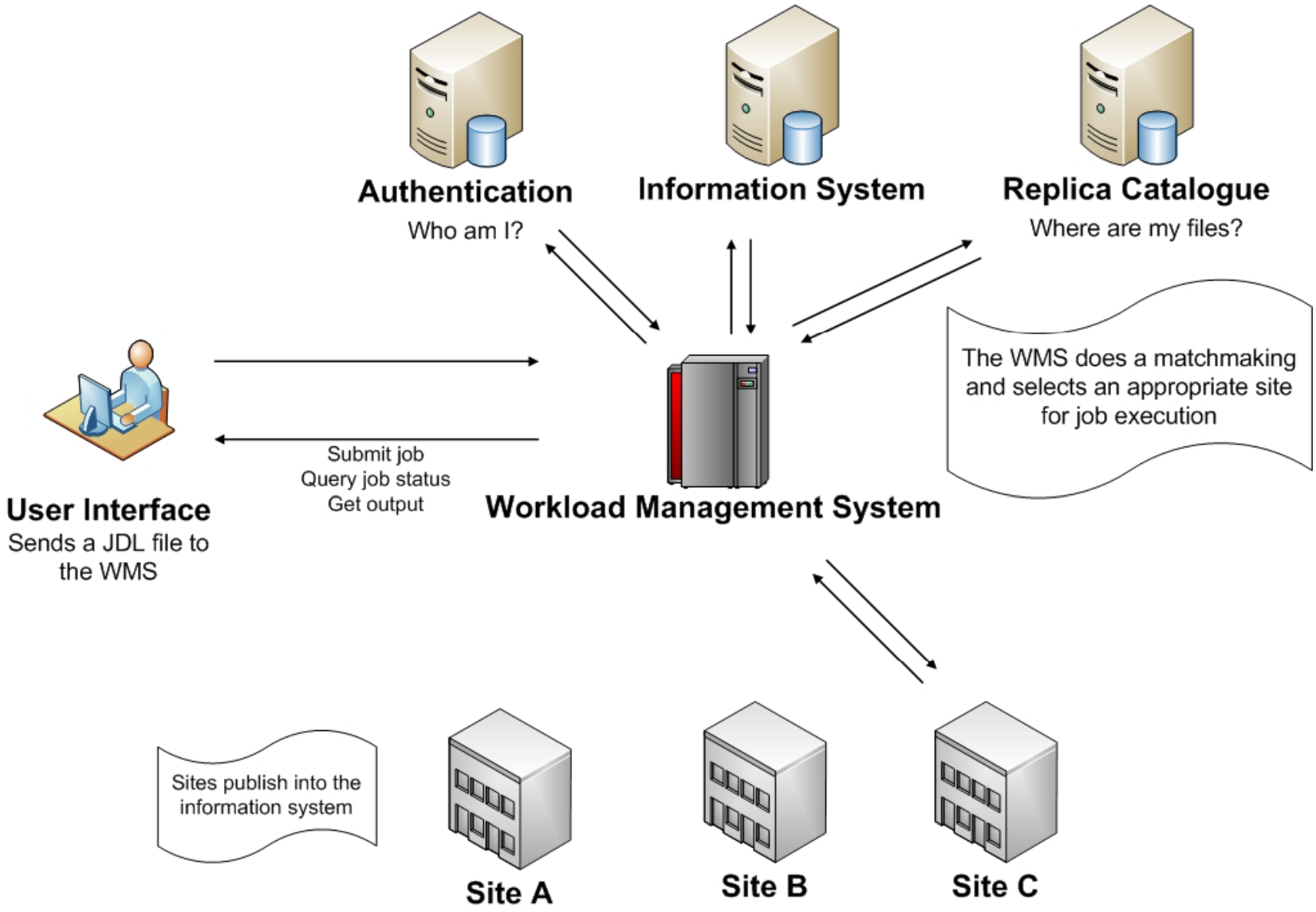
How to use the Grid?

- **Grids are designed to handle large sets of limited duration jobs that produce or use huge amount of data**
- **How to submit a job to the Grid?**
  - Direct job submission
  - Job submission via the Workload Management System (WMS)
  - Pilot job submission

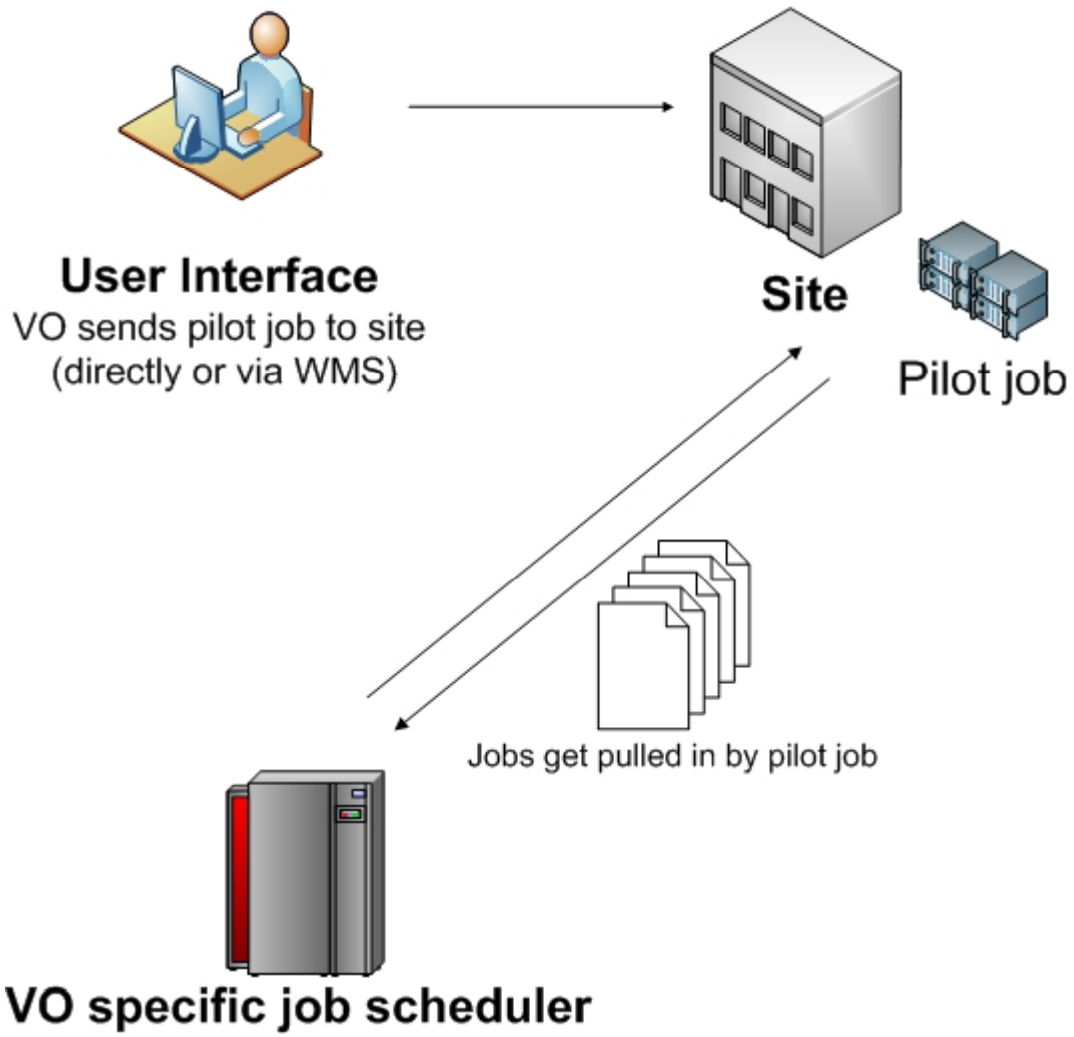
Direct job submission



Job submission via WMS

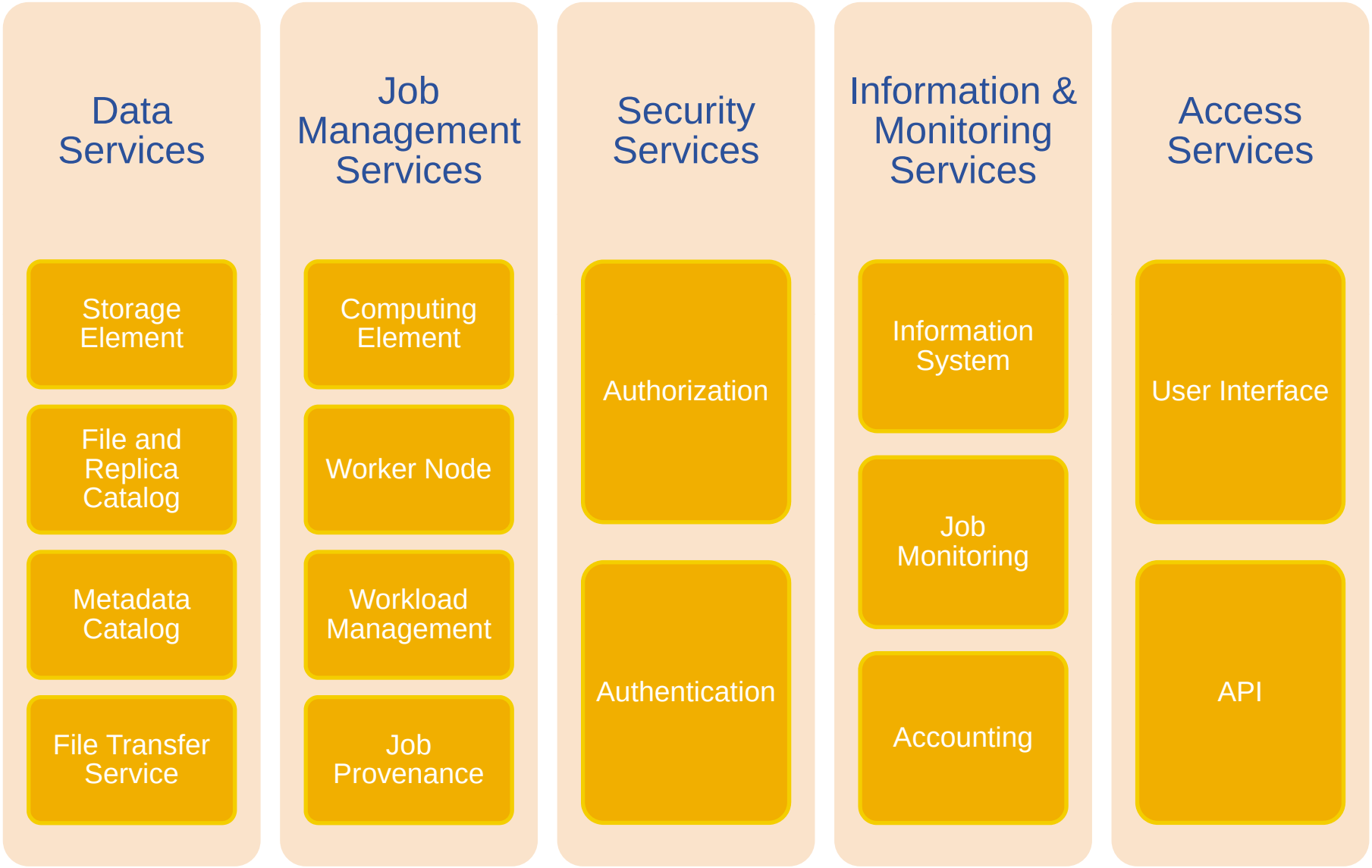


Pilot job submission



- WN reserved by pilot job is a first class resource (no need to wait for batch system or WMS)
- Just in time scheduling, VO policies implemented at the central VO queue
- Needs mechanism for identity switch for accounting (pilot job submitted by different user than pulled in jobs)

gLite middleware

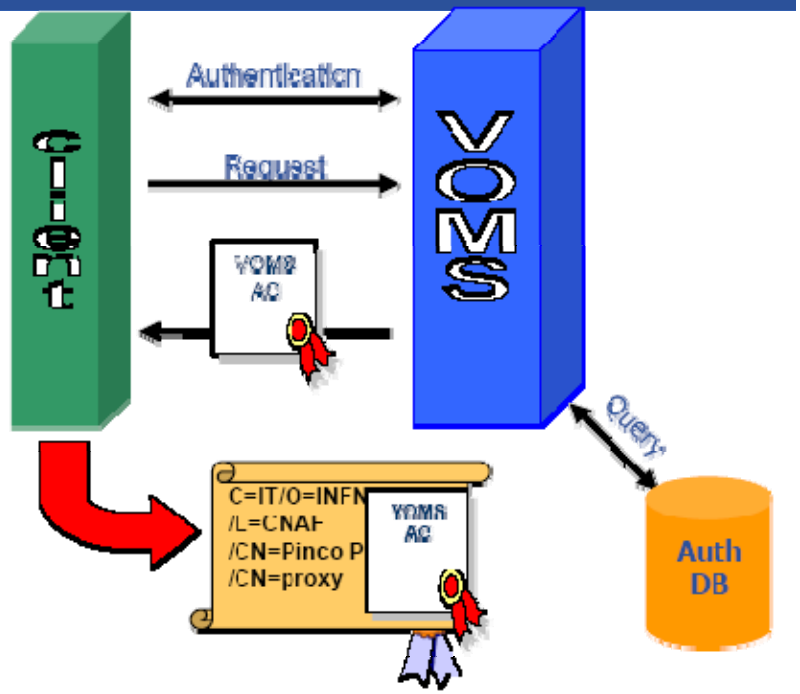


Authentication

- **Authentication is based on X.509 PKI infrastructure**
  - Certificate Authorities (CA) issue (long lived) certificates identifying individuals (much like a passport)
    - Commonly used in web browsers to authenticate to sites
  - Trust between CAs and sites is established (offline)
  - In order to reduce vulnerability, on the Grid user identification is done by using (short lived) proxies of their certificates
- **Short-Lived Credential Services (SLCS)**
  - issue short lived certificates or proxies to its local users
    - e.g. from Kerberos or from Shibboleth credentials (new in EGEE II)
- **Proxies can**
  - Be delegated to a service such that it can act on the user's behalf
  - Be stored in an external proxy store (MyProxy)
  - Be renewed (in case they are about to expire)
  - Include additional attributes

Authentication

- **Virtual Organization Membership Service**
- **Central repository for user authorization information, providing support organizing users into a general group hierarchy, keeping track of their roles etc.**
- **VOMS service issues Attribute Certificates that are attached to certificate proxies**
  - Provide users with additional capabilities defined by the Virtual Organization
  - Base for the Authorization process



Authorization

- **Based on their VOMS proxies users are mapped to a local account on the site**
- **Site Access Control (SAC) components**
  - Local Centre Authorization Service (LCAS)
    - Makes yes/no authorization decisions
  - Local Credential Mapping Service (LCMAPS)
    - Translates VOMS proxies to UNIX accounts
  - Site Central Authorization Service (SCAS)
    - Central administration point, contacted by clients (e.g. LCMAPS)
  - gLExec
    - Executes a binary with different uid (sudo); uses LCMAPS to get the uid. Used for pilot jobs
- **ARGUS**
  - New authorization service - more flexible, better support to specify policies, improved fault tolerance

Computing Element

- **Entry point to the site**
- **Takes grid job and gives it to the local batch system for execution**
- **lcg-CE: to be phased out**
- **CREAM (Compute Resource Execution And Management)**
  - Basic job operation (cancel, suspend etc.)
  - Proxy renewal & delegation
  - CLI

Clients: Worker Node & User Interface

- **Worker Node**
  - A batch system node where the actual computation takes place
  - Has clients for data management, authorization service and information system
- **User Interface**
  - Contains all the clients needed to interact with the Grid: Job submission, data management, authorization, information system
  - To be installed on the user's PC

Storage Element

- **Disk Pool Manager (DPM)**
  - Manages storage on disk servers
  - Nearly 200 instances in use
  - Can manage up to 360TB
- **dCache**
  - Distributed with gLite but not developed within EGEE
  - Can also handle tape storage
- **CASTOR**
  - Handles tape storage
  - Not distributed with gLite
- **STORM**
  - Disk based
  - Not distributed with gLite

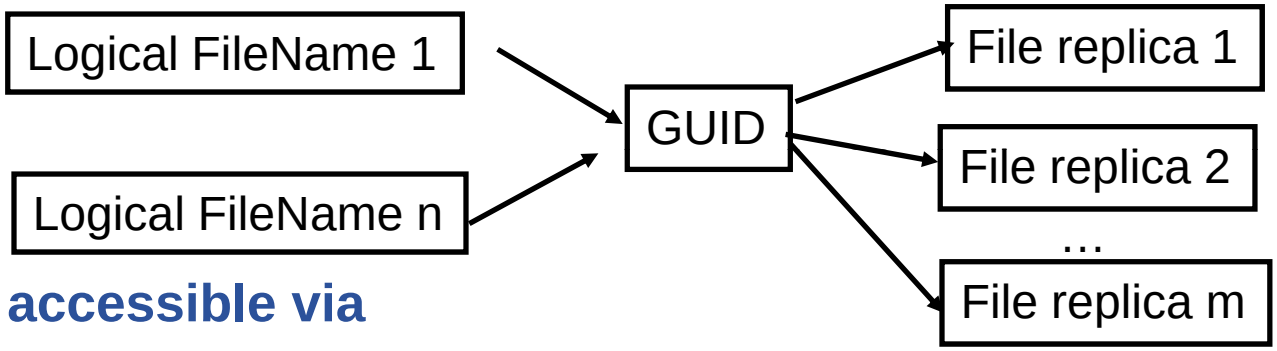
These SEs have their own File Access Protocols but also implement the SRM interface

Storage Resource Manager

- **Storage Resource Manager (SRM)**
  - Middleware component to provide dynamic space allocation and file management on shared storage components.
  - SRM v2.2 interface specified. Targets the specific needs in Grid Computing.
    - Storage Classes
    - Manage and reserve storage space
    - Filesystem-like operations
    - Volatile vs. permanent data
    - Transparent, automatic or forced migration to tertiary storage
    - Mechanisms for locating data
    - Namespaces
- **Implemented by all SEs (DPM, CASTOR, dCache, STORM)**

LCG "File" Catalog

- **The LFC stores mappings between**
  - Users' file names
  - File locations on the Grid



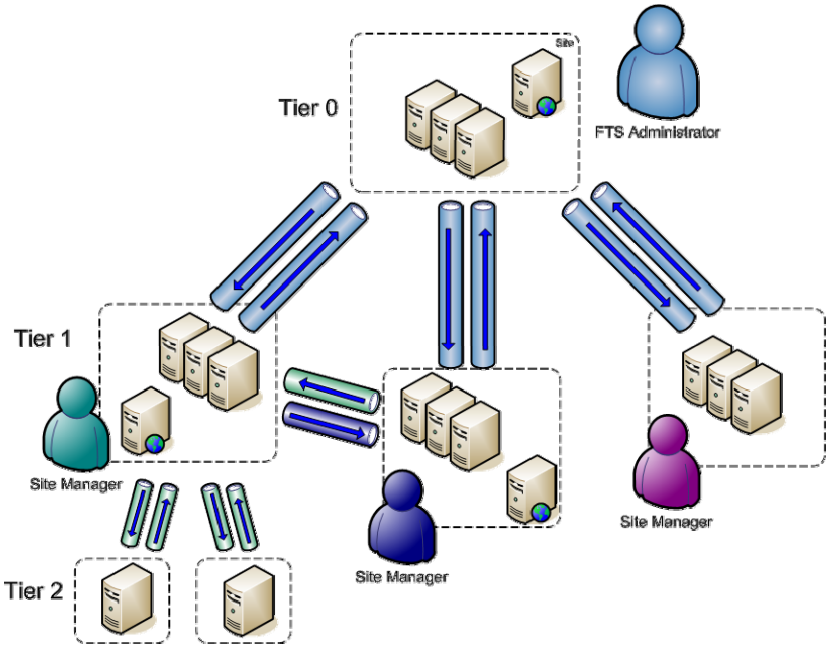
- **The LFC is accessible via**
  - CLI, C API, Python interface, Perl interface
    - Supports sessions and bulk operations
  - Data Location Interface (DLI)
    - Web Service used for match making:
      - *given a GUID, returns physical file location*
- **46 instances are in use**

DM clients

- **Hide the complexity from the user**
  - Interact with information system, file catalogues and SRM storage elements
- **GFAL**
  - POSIX-like I/O functions (open(), read() etc.)
  - SRM abstraction layer
  - C, Python APIs and CLI
- **lcg_util**
  - Cover most common use cases (file creation, registration, replication, deletion etc.)
  - C, Python APIs and CLI

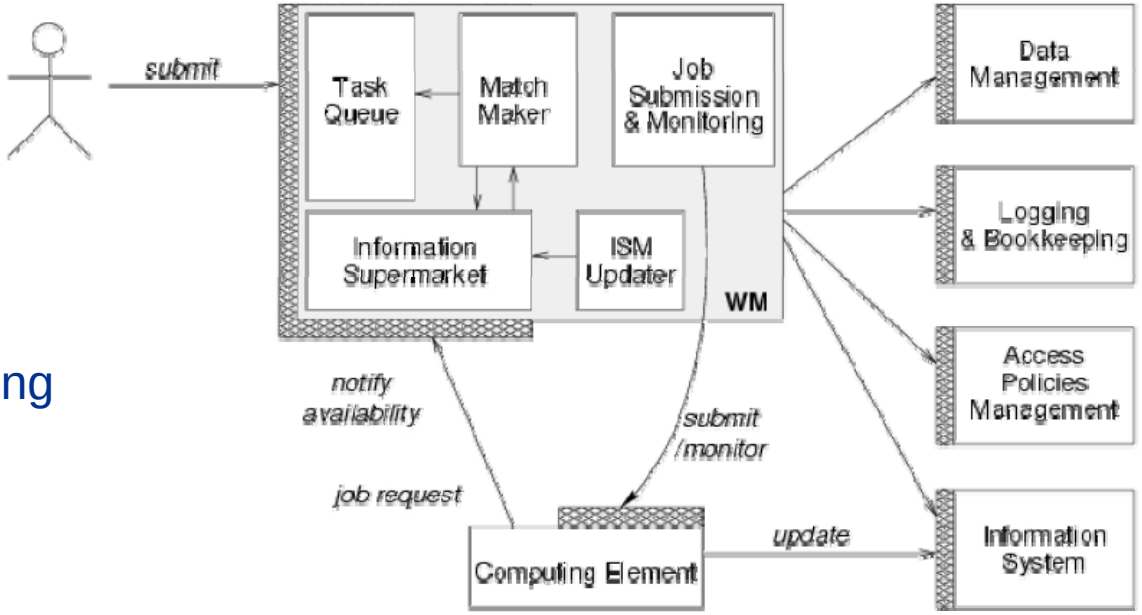
File Transfer Service (FTS)

- **Reliable File Transfer Service**
  - Bulk data transfer between SRM compliant SEs (batch system for file transfers)
  - Multi-VO service to balance network/SE utilization
  - Prevents overloading network/SE resources
  - Service monitoring and statistics



Workload Management System

- **WMS: Resource brokering, workflow management, I/O data management**
  - ➔ Web Service interface: WMProxy
  - – Task Queue: keep non matched jobs
  - – Information SuperMarket: optimized cache of information system
  - – Match Maker: assigns jobs to resources according to user requirements (possibly including data location)
  - – Job submission & monitoring
    - ➔ Condor-G
    - ➔ *ICE (to CREAM)*
  - – External interactions:
    - Information System
    - Data Catalogs
    - Logging&Bookkeeping
    - *Policy Management systems*



Job Description Language (JDL)

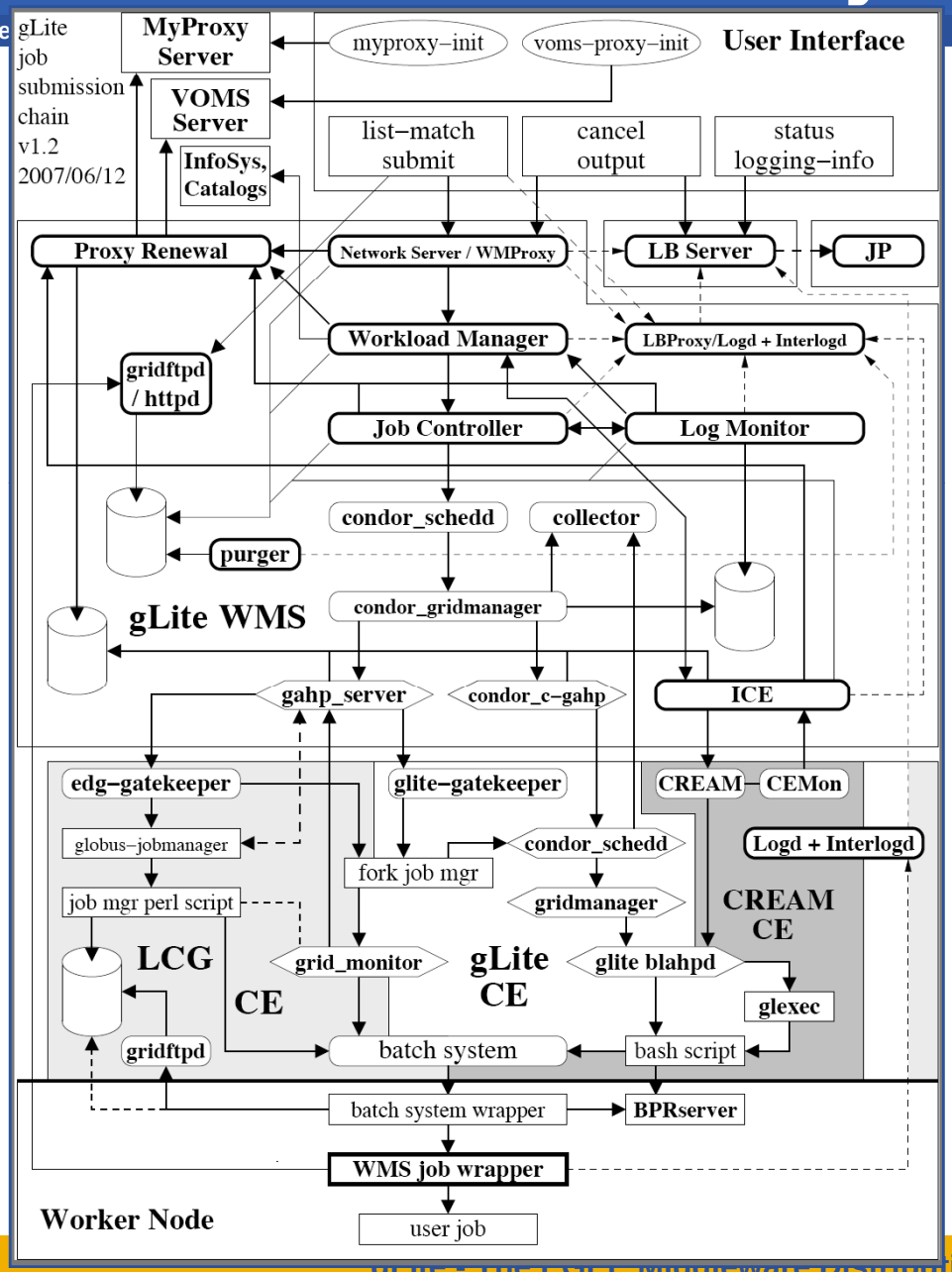
- Describes (aggregates of) jobs and their characteristics and constraints
- Based on the Condor ClassAds
- Accepted by WMS and CREAM

Example:

```
Executable = “test.sh”;
Arguments = “fileA”;
StdOutput = “std.out”;
StdError = “std.err”;
InputSandbox = {“test.sh”, ”fileA”};
OutputSandbox = {“std.out”, “std.err”};
Requirements = Member(“VO-dteam-SW-v2-
01”,other.GlueHostApplicationSoftwareRunTimeEnvironment);
```

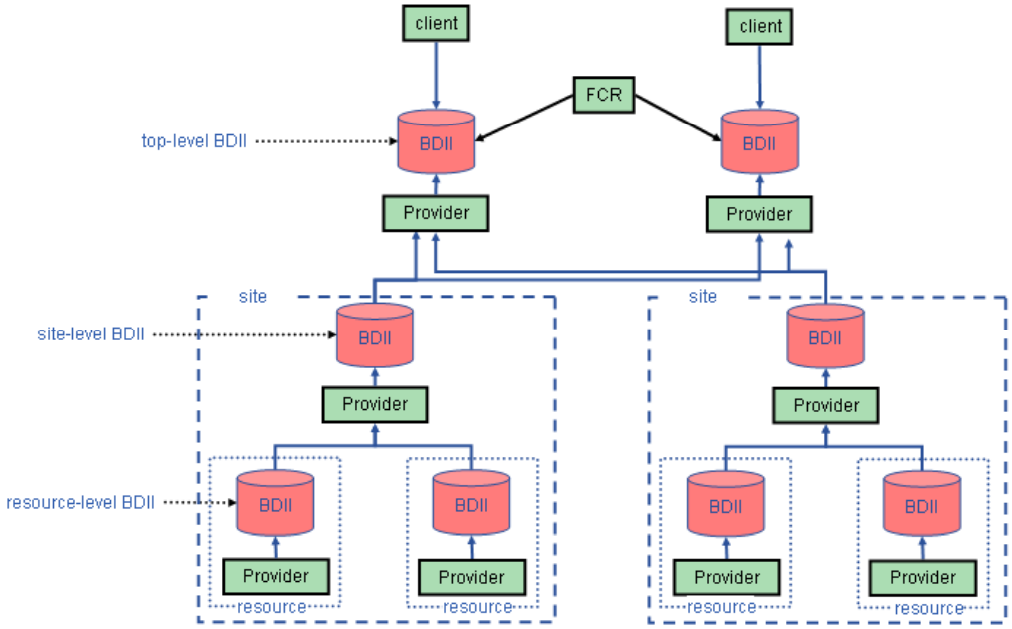
What it really does

- **Thanks to Maarten we know:**
  - Simplified view



Information System

- **Berkeley Database Information Index (BDII)**
  - Uses a standard OpenLDAP server as supplied by the OS
    - With the Berkeley database backend
  - Updated by a Python process
    - Configuration file containing LDAP URLs for the sites
    - Use ldapsearch command used as it is stable



Information System

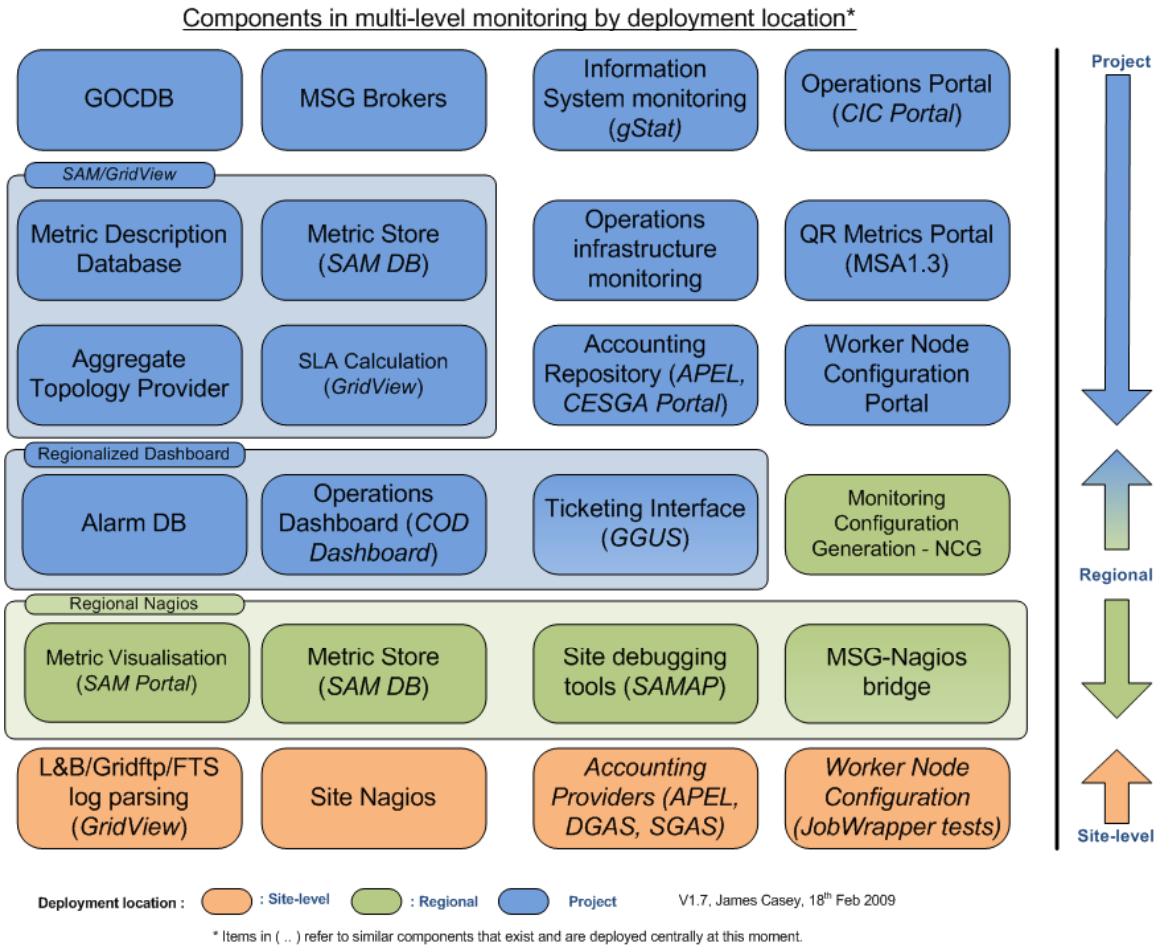
- **Generic Information Provider (GIP)**
  - Framework for information providers (plugins)
  - Populate BDIIs with information and also read information from other BDII (resource, site, top level hierarchy)
  - Get information and format it as LDIF
- **Freedom of Choice for Resources (FCR)**
  - FCR portal has list of services available to a VO
  - VO manager can black list sites
  - Information is propagated to VO specific top level BDII
- **GLUE Schema**
  - Schema that describes grid resources (computing, data, storage, services)
  - Collaborative effort between different grid projects, now within OGF

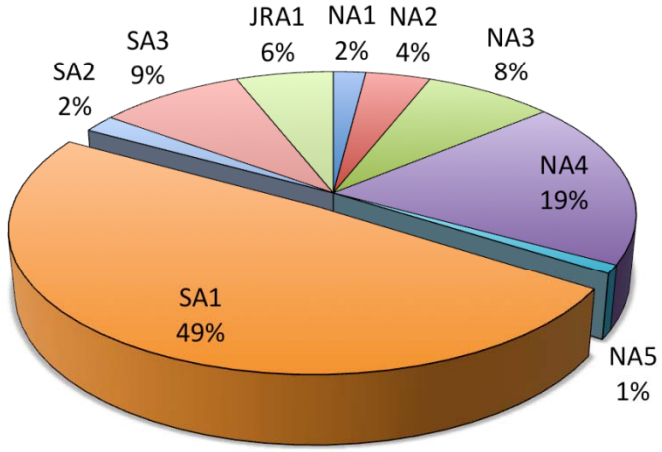
Other services

- **Logging and Bookkeeping (LB)**
  - Tracks jobs in terms of events (submission, starting execution etc.) gathered from the WMS and CE.
  - Events are stored and can be queried.
- **HYDRA**
  - Encrypted storage on SEs
  - Encryption keys are split an stored in >2 keystores
- **AMGA**
  - Metadata catalogue
  - Metadata Usually lives in relational databases
    - Why not accessing DBs directly on the Grid? Possible but
      - *Authentication (VOMS)*
      - *Connection pooling*
      - *Data replication*

EGEE Operations Tools

- **Not part of gLite but indispensible for running the grid**





gLite development process

- **Middleware developed by geographically distributed teams (mostly at research institutes and Universities).**
- **A team focuses on a particular service.**
- **Teams are quite independent**
  - Coding conventions
  - Documentation
  - Naming conventions
  - ...
- **Currently ~ 20 FTEs (we are in EGEE III, manpower was more than double in the previous phase).**
- **More than 80 people in 12 different institutions involved.**

gLite development process

- **The project has a technical director but no single architect.**
- **Decisions are being taken in a collaborative, consensus based process.**
  - Bi-weekly phone conference to discuss short term priorities (bug fixing etc.).
  - Monthly Technical Management Board to discuss more strategic issues (new features etc.). Takes also input from the user community.
  - 2-3 all hands meetings per year.

gLite characteristics

- Distributed under an open source license.
- Main platform is Scientifc Linux (recompiled RH EL).
- Many 3rd party dependencies (tomcat, log4*,gSOAP , ldap etc.). Pulled in from public repositories

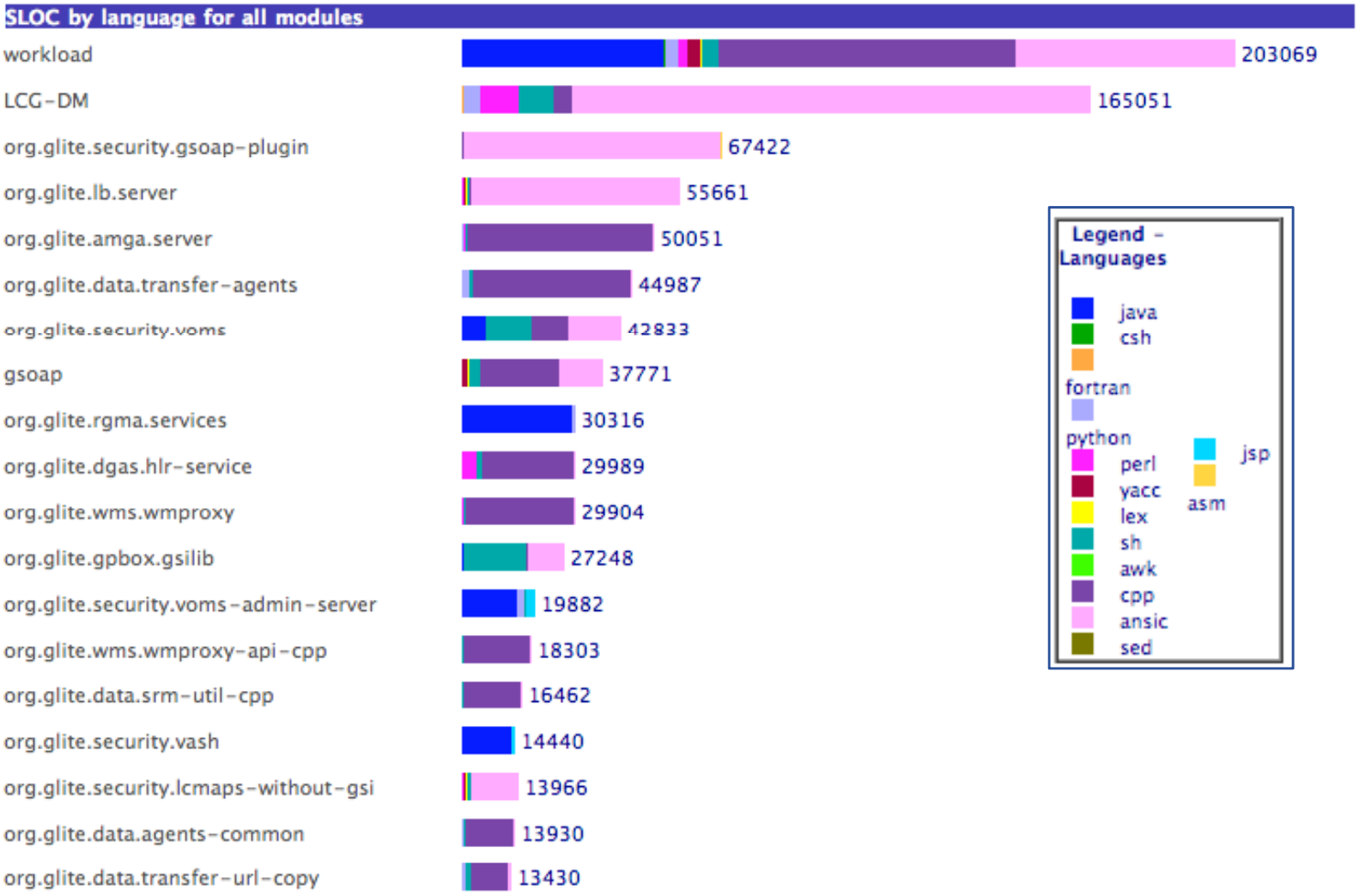
Total Physical Source Lines of Code (SLOC)

SLOC = 1622714

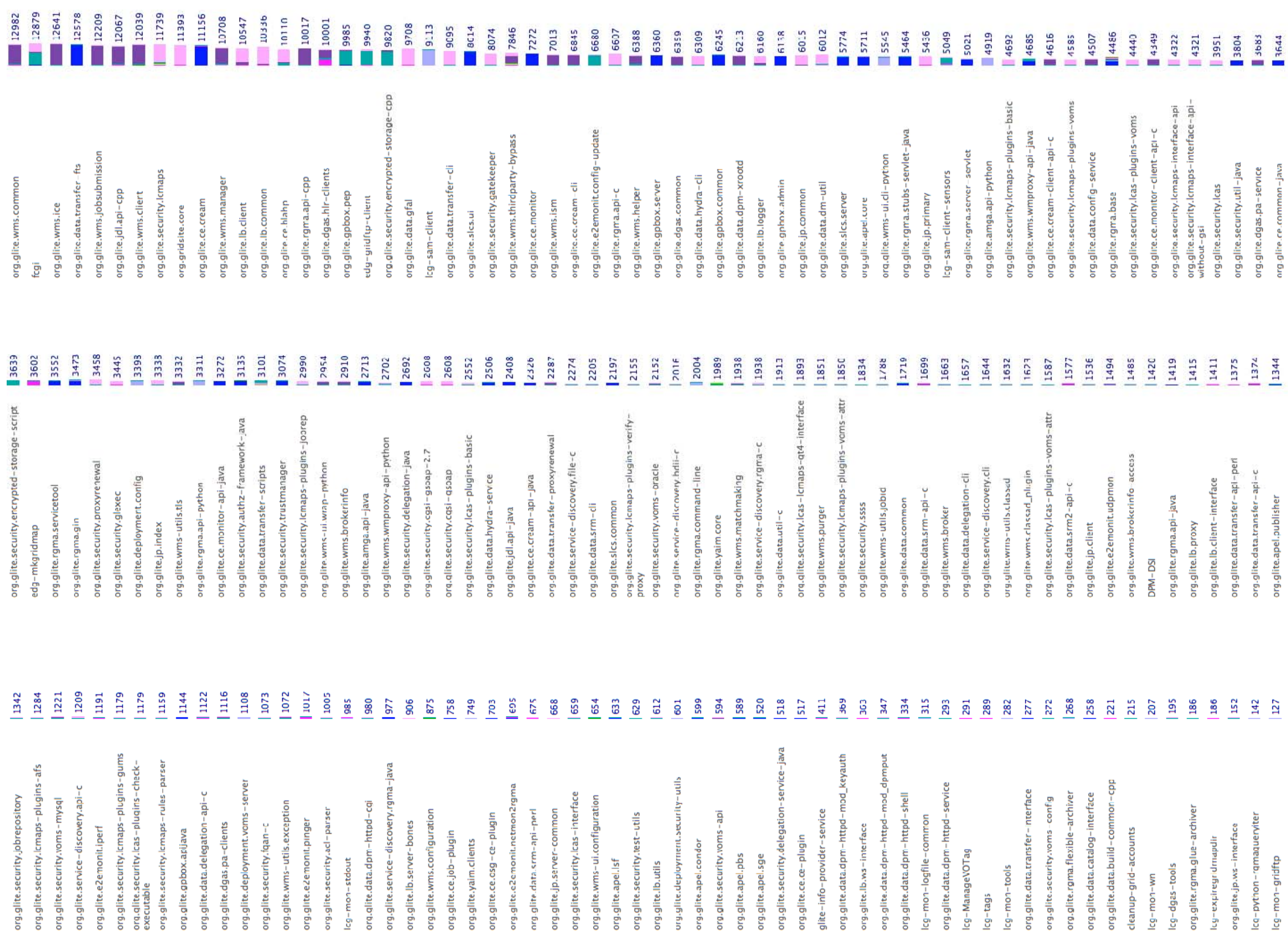
Total SLOC grouped by language (dominant language first)

Language	Total SLOC
ansic	578598 (35%)
cpp	491801 (30%)
java	251382 (15%)
sh	191798 (11%)
python	54510 (3%)
perl	39258 (2%)
yacc	7445 (0%)
jsp	4444 (0%)
lex	2274 (0%)
csh	701 (0%)
awk	307 (0%)
fortran	124 (0%)
sed	68 (0%)
asm	4 (0%)

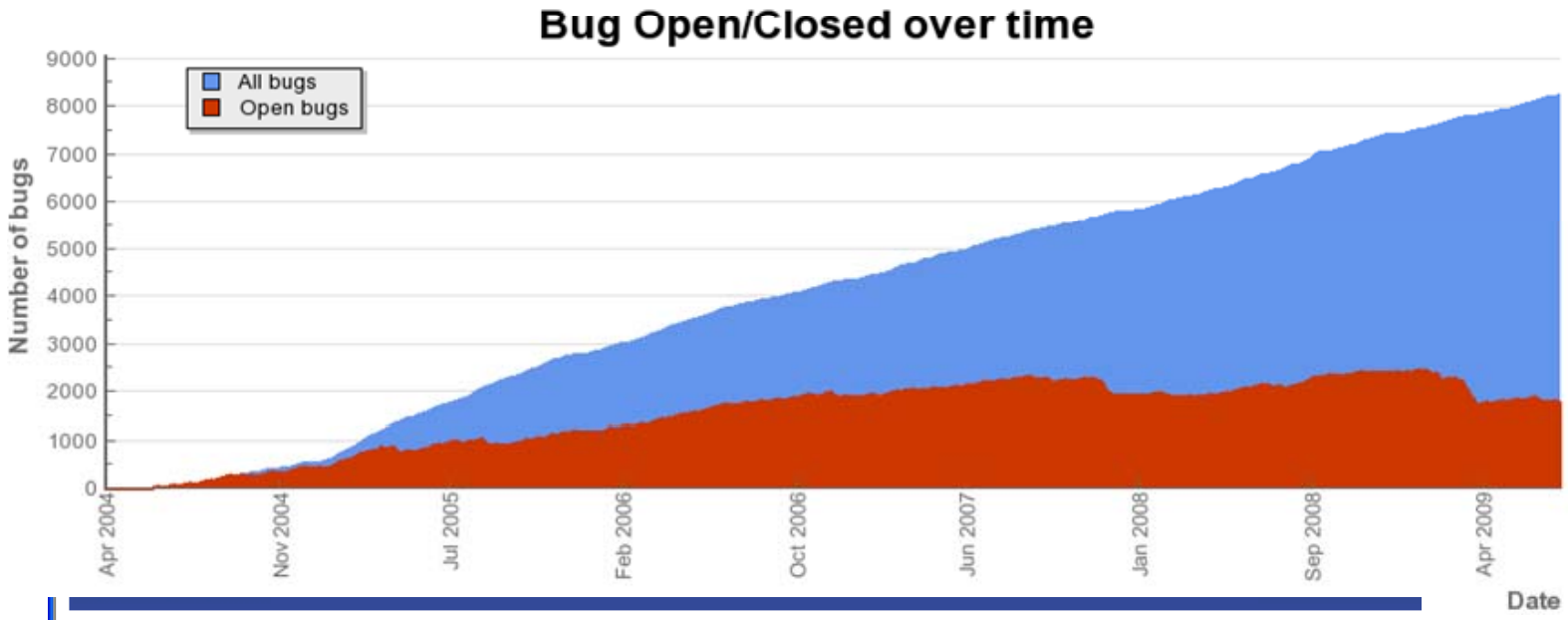
gLite code details

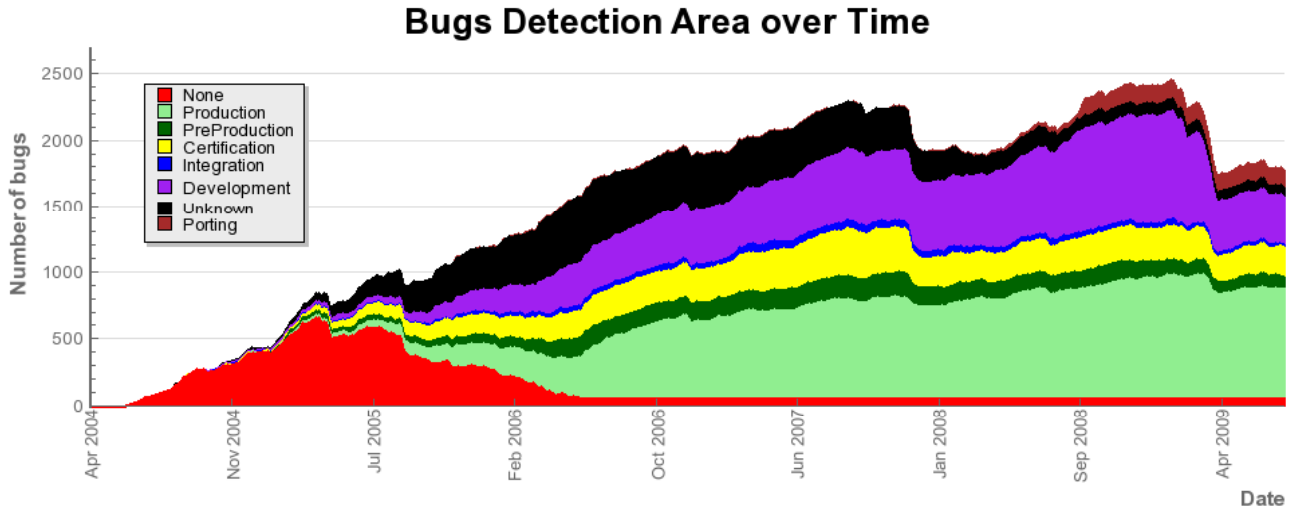


gLite code details



gLite bug statistics





gLite release process

- **Several releases per month**
- **Release model: pull model (rpm repository, tarballs to download).**
  - Sites pick up updates if they like to.
  - Multiple versions of services are in production.
  - Retirement of old versions is a lengthy process.
- **Release model: phased. We release updates to individual node types when needed, no big bang.**

The future

- **EGEE project will terminate in April 2010**
- **European Grid Initiative (EGI)**
  - To establish a sustainable grid infrastructure in Europe
  - http://web.eu-egi.eu/
  - Handle middleware maintenance, integration, testing and release within EGI
  - To produce the Unified Middleware Distribution (UMD)
  - Convergence of ARC, gLite and Unicore
  - http://knowledge.eu-egi.eu/knowledge/index.php/UMD