Experiences with the GLUE information schema in the LCG/EGEE production Grid

Stephen Burke, Sergio Andreozzi and Laurence Field

CHEP07, Victoria, Canada
Overview

• Why we need a schema
• The GLUE project
• How GLUE is used in the LCG/EGEE Grid
• Experience, successes and problems
• Outlook

• This talk focuses on the use of the GLUE schema in the LCG/EGEE Grid – other Grids are available!
Why do we need a schema?

• A Grid consists of many sites with a wide variety of resources

• Users, applications and middleware need to know what resources are available and what their properties are
  – What Resource Brokers are available to CMS?
  – Find a Computing Element running SL4 with > 2 Gb memory
  – Find a Storage Element with 20 Tb of free space

• Grid management and operations staff need an overview of the state of the Grid
  – How many jobs are running in the UK?
  – How much disk space has ATLAS used?

• The schema allows the resource properties to be published and queried in a uniform way

• The information is transported via an information system, but the schema is logically independent of it
Desirable schema properties

- **Not too big** – don’t describe every detail, only the things which are really needed
- **Not too small** – need to capture all the important features
- **Flexible** – must be able to cope with a very wide range of resource configurations
- **Precise** – the semantics should be clear and unambiguous
- **Simple** – easy to understand what the attributes mean
- **Calculable** – it should be possible to determine the values of attributes in a short time (typically < 1 second)
- **Extensible** – it must be possible to evolve the schema without breaking existing software
Why standardise the schema?

- Many different Grids, interoperability is a major activity
  - For WLCG this means EGEE, OSG and NDGF
- Must publish the same information, with the same names and semantics
  - At least for a core set of attributes
  - Even trivial differences, e.g. units or case of text, can cause problems
  - Can sometimes write translators between different schemas, but this is complex and error-prone
- Different Grids can learn from each other’s experience
  - And avoid duplicating work
- But need to focus on what is really needed
  - Standardisation activities can be slow
  - Design by committee not always optimal
- The European DataGrid project (predecessor of EGEE) initially had its own schema (2001)
- The GLUE (Grid Laboratory for a Uniform Environment) project was a collaboration between EDG, EU DataTAG, iVDGL (predecessor of OSG) and Globus to promote interoperability
  - The GLUE schema 1.0 was defined in September 2002 after several months of discussion
  - Version 1.1 was released with some minor improvements in April 2003, and deployed by EDG and then LCG and EGEE
  - Version 1.2 was agreed in February 2005, finalised during 2005 and deployed (fairly gradually) by LCG/EGEE in 2006
  - Version 1.3 was agreed in October 2006 and is starting to be deployed now
Evolution process

• Evolution has been fairly slow – two upgrades in four years
  – Collect problems and ideas, discuss by email/phone – several months
  – One face-to-face meeting to agree changes – intensive but productive
  – Write documentation, update schema implementations and deploy them – also a few months
  – Update information providers – timescale varies, can be 1-2 years
  – Update clients – timescale varies, can be infinite!

• Backward compatibility maintained through the whole process – significant constraint
  – Some sites take a very long time to upgrade
  – Many legacy objects/attributes
Evolution issues

• **The schema touches everything (middleware, users, sysadmins, …) so lots of dependencies**
  – But most people are not schema experts
  – Small group involved directly in defining the schema, most with other jobs

• **GLUE must meet the needs of many different Grids**
  – Current contributors include EGEE, OMII-Europe, KnowArc, TERAGRID, NAREGI, UNICORE, NGS, OSG, APACGrid, …

• **Different implementation technologies**
  – Currently LDAP, relational (R-GMA), classad, XML
  – Places constraints on structure
    ▪ No tables or primary keys in LDAP
    ▪ No multivalued attributes in a relational schema
    ▪ classads are flat lists
The schema is defined in an abstract UML format
- Objects with attributes and relations
- Attributes have types, can be single- or multi-valued

**GlueSite** – describes a Grid site
- Location, contacts, affiliation, …

**GlueService** – describes attributes of a generic service
- Type, endpoint, status, ACL, …

**GlueCE** – a complex set of objects and attributes describing a computing resource
- Queue, policy, cluster, …

**GlueSE** – a complex set of objects and attributes describing a storage resource
- Storage area, control and access protocols, policies, …

Also many subsidiary objects
• The Resource Broker allows job submission to be directed according to requirements on the GLUE schema attributes, and a ranking expression defines the order of preference
  – JDL (Job Definition Language) uses classads, which need to be mapped to the schema

• The data management clients query the information system for the attributes of Storage Elements

• Other tools present information directly to the user

• Monitoring tools collect summary information for the whole Grid

• The storage schema is currently used for a prototype storage accounting system
  – Although this is not in general a target use for the schema
• **Glue 1.2 introduced the GlueService concept to publish generic service information**
  – gLite has a Service Discovery API to query it
  – Can publish some service-specific information (key/value pairs)

• **Slow takeup**
  – Mainly data management so far

• **GlueService is not explicitly linked to GlueCE/SE**
  – Clients may make ad-hoc assumptions to link them, e.g. matching hostnames

• **No generic information provider**
  – Just static configuration
  – Some services have custom publishers
  – New publisher now being certified
Mapping the schema

• Real systems are very varied and complex
  – The schema is uniform and simple
• Not always obvious how to relate the two
  – Usually have to make assumptions and simplifications
  – May be wrong, or generate mismatches between sites
• CE: now have a framework with plugins for each batch system
  – Uniform assumptions, common code where possible
• SE: main target is SRM, but still under development
• First priority is to err on the safe side
  – Beware of black holes!
• If an attribute isn’t used people may not be careful to get it right, so then it can’t be used!
• Need schema validation tools to check for sanity
Publishing the information

- Need a precise definition of attributes, even where it seems trivial
  - Long discussion about OS names (not defined in schema)
- Sysadmins ideally should not need to define things by hand
  - Typos, misunderstanding of semantics
- Are attributes optional?
  - Technically yes in most cases, but not always obvious what it means
  - Usually no special value to mean N/A
- Units must be clear
  - Gb vs Gib etc
- Information providers sometimes slow, can load the system
  - Introduce caching, but then the information is older
- Some things are hard to calculate
  - EstimatedResponseTime
  - Used/Free space for storage
- Mistakes/hacks can be hard to fix
  - Incorrect assumptions get embedded in client code
CE issues

- **Basic structure is CE – Cluster – SubCluster**
  - CE is a batch queue, cluster is the hardware
  - SubClusters are homogeneous groups of nodes
  - Original schema design had many detailed host-level attributes – largely unused

- **Resource Broker can’t choose SubClusters, so the concept isn’t usable**
  - In practice we have one heterogeneous subcluster per cluster

- **Queue concept for CE too simple**
  - Some batch systems have no queues
  - Usually have fairshares within a queue
  - Glue 1.2 introduced the VOView to represent a share

- **Some ambiguities in mapping to real systems**
  - CPUs vs job slots
  - Memory per node, or per job?

- **No information about disk space for scratch files**
SE issues

- Original schema was defined for “classic SE” – simple disk server + gridftp
  - Plus other access protocols, e.g. rfio, file
- Now using Storage Resource Manager (SRM)
  - In transition from SRM v1 to v2
- GLUE schema v 1.3 has several enhancements for SRM
  - But not much experience yet – just starting to deploy it
- Lots of debate about free/used/available/reserved space
  - Schema defines lots of attributes, we will see what can be published
Interaction with users

• Users need to understand the meaning and limitations of the schema attributes
• Production users now often have complex Requirement/Rank expressions built up from years of experience
  – Ordinary users may be less sophisticated
• Users often ignore attributes which are “usually” not relevant
  – Max queued jobs, OS type, CPU/wallclock time limits, …
• Need frameworks to shield users from the details
• The GLUE schema has developed over 6 years of practical use by EDG/LCG/EGEE
  – And other Grids
• It has proved to be sufficient to allow many users to submit large numbers of jobs, manage data and monitor the Grid
  – No show stoppers
• However, many rough edges
  – But now we know where most of the problems lie
• GLUE is now an OGF working group
  – Aiming for a major redesign – GLUE 2.0
  – Includes experience from many more Grids
  – See poster session
  – … watch this space!