GLUE 2.0: Rollout strategy and current status

Stephen Burke, RAL

EGEE09, Barcelona

Overview

- **Why do we need a new schema?**
- **The GLUE 2.0 schema**
  - What does it look like?
  - What are the major changes from GLUE 1.x?
  - What advantages does it have?
- **Implementation and rollout in EGEE**
  - BDII
  - Information providers
  - Clients
- **Summary**

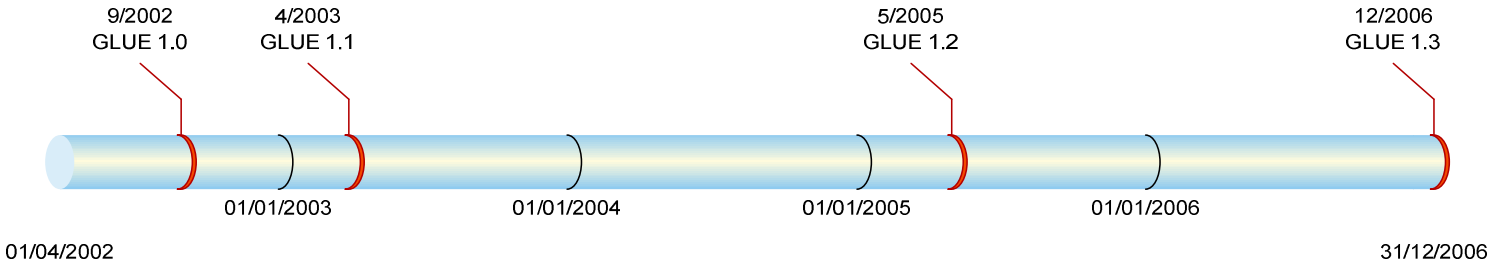
Why do we need a schema?

- **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 Workload Managers are available to CMS?
  - Find a Computing Service running SL5 with > 3 Gb memory
  - Find a Storage Service with 20 TB of free space
- **Grid and VO 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?
  - What is the total installed CPU power available to LCG?
- **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**

GLUE history

- **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 in 2003/4
  - Version 1.2 was agreed in February 2005, finalised in May 2005 and deployed (fairly gradually) by EGEE in 2006
  - Version 1.3 was agreed in October 2006, finalised in December 2006 and deployed from 2007 onward (adoption is still in progress!)



Problems with GLUE 1.x

- **The schema has worked, but we have many accumulated issues**
- **Initial schema definitions were based on limited experience**
  - Only for CE and SE
    - No SRM for storage in 2002, just “classic SE”
  - Embedded assumptions which turned out to be too restrictive
    - Not easily extendable
- **Definitions not always clear, documentation somewhat limited**
  - Case sensitivity, optional attributes, units, special values (“undefined”)
  - Ambiguities (CPUs/job slots)
  - Not future-proof (cores)
  - Too specific (only two CPU benchmarks, SpecInt 2k and SpecFloat 2k)
  - Many things effectively defined only by EGEE practice
- **We always required changes to be backward-compatible to make upgrading easier**
  - 1.x schema had limited scope for additions, so changes often “shoe-horned” into the available structure
  - 1.2 schema introduced a generic GlueService object, but it had no connection to the existing CE and SE objects

Moving into the Open Grid Forum

- **Always intended to defer conceptual and structural changes to a major revision called GLUE 2.0**
  - Complete redesign, no backward compatibility
    - But without losing what we already had in GLUE 1
  - Incorporating lessons from many years of experience
- **First discussion on GLUE 2.0 at the October 2006 meeting**
- **Decision made to define GLUE 2.0 within the OGF**
  - Many (~ 14) Grid projects participating
  - End up with a genuine standard
  - The OGF process didn't create too much overhead
- **Positive Outcomes**
  - GLUE widely accepted within OGF
    - Interacts with other OGF standards (BES, SAGA, JSDL, …)
  - Increased participation from, and hence acceptance by, other projects
    - Broad range of viewpoints challenged implicit assumptions
  - Raised visibility/commitment within EGEE
    - Recent mentions from the EGEE project leader and technical director!

Steps to GLUE 2

- **Schema migration is a complex process:**

1) **Define the abstract schema**
2) **Define the LDAP rendering**
3) **Implement the schema in the BDII and rollout**
   - You are here!
4) **Write and deploy information providers**
5) **Update client tools to understand GLUE 2**
6) **(Retire GLUE 1)**

- **The schema interacts with everything, so the rollout must be a gradual process without breaking anything**

GLUE 2.0 timeline

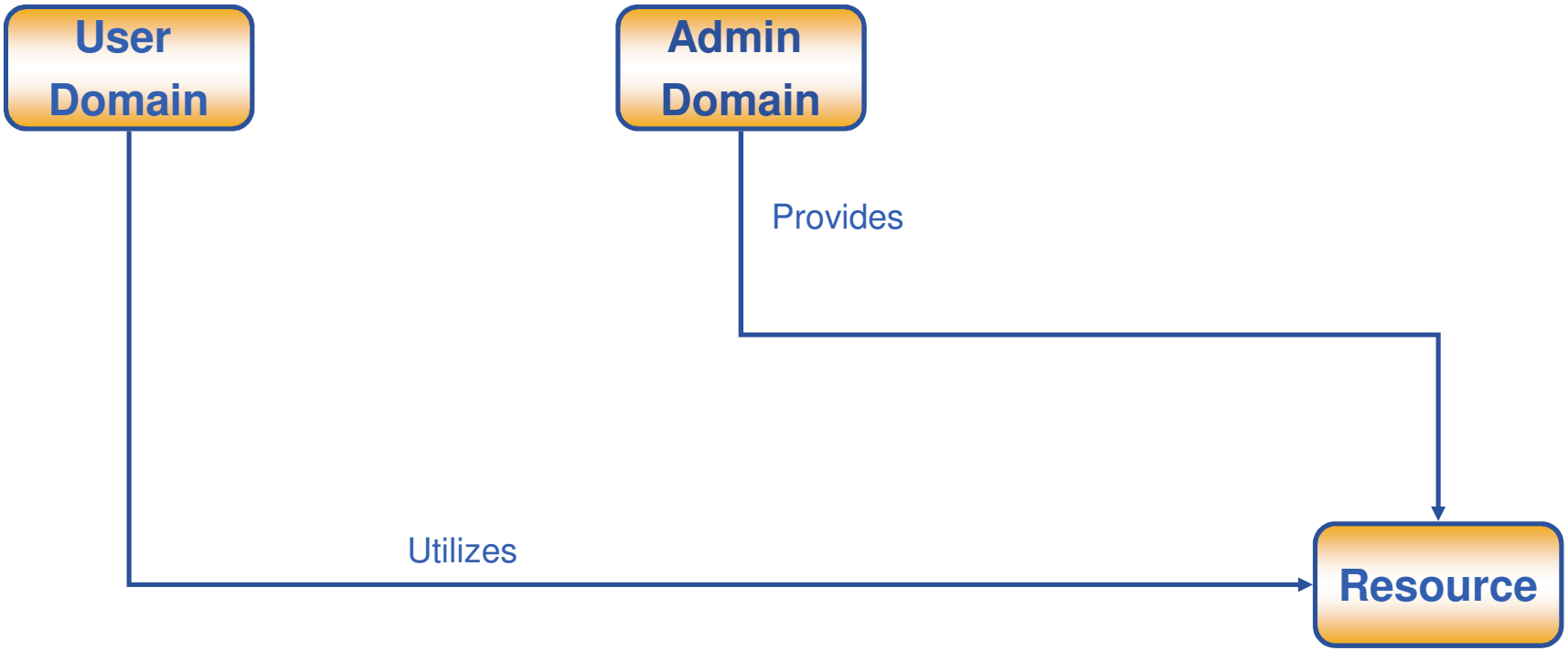
- October 2006: Decision taken to move into the OGF
- January 2007 (OGF 19): First working group meeting
- June 2008 (OGF 23): Draft specification opened to public comment
- August 2008: Public comment period ended
- November 2008: Started addressing comments
- January 2009: Final specification ready
- March 2009 (OGF 25): GLUE 2.0 becomes an official OGF standard
  - http://www.ogf.org/documents/GFD.147.pdf

- 1st April 2009: Started work on GLUE 2.1 ☺
  - Some concepts known to be somewhat fuzzy
  - No doubt we will find problems with 2.0 once we start implementing!
    - A few things found already, but nothing major

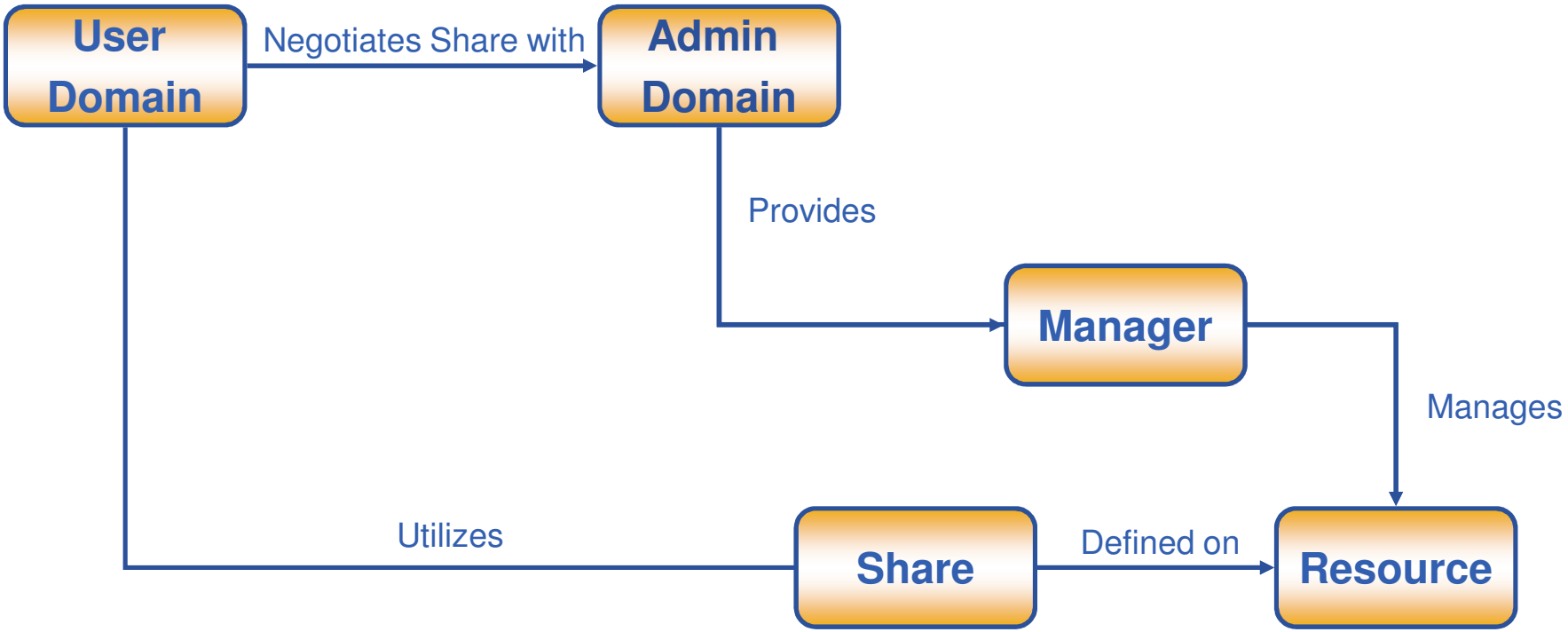
Some statistics

- **~ 60 phone conferences**
  - 1.5-2 hours each, so ~ 5 days talking
  - ~5 people participating, so ~ 4 months FTE invested in total
    - This does not include the time invested by editor (OMII-Europe)
- **~ 1000 emails in the GLUE mailing list**
- **~50 draft versions of the specification document**
  - 2 years from first meeting to final version
    - Document updated nearly every week (public comment period excepted)
  - 76 pages, 27609 words
- **~ 18 sessions in 7 OGF events**
- **~ 40 issues gathered from public comments**
  - + ~60 more issues left to be discussed after public comments
- **Schema defines 246 attributes in 35 objects**
  - Not counting relations or inherited attributes
- **Three different renderings**
  - LDIF, XML, Relational
  - + RDF? + CIM??

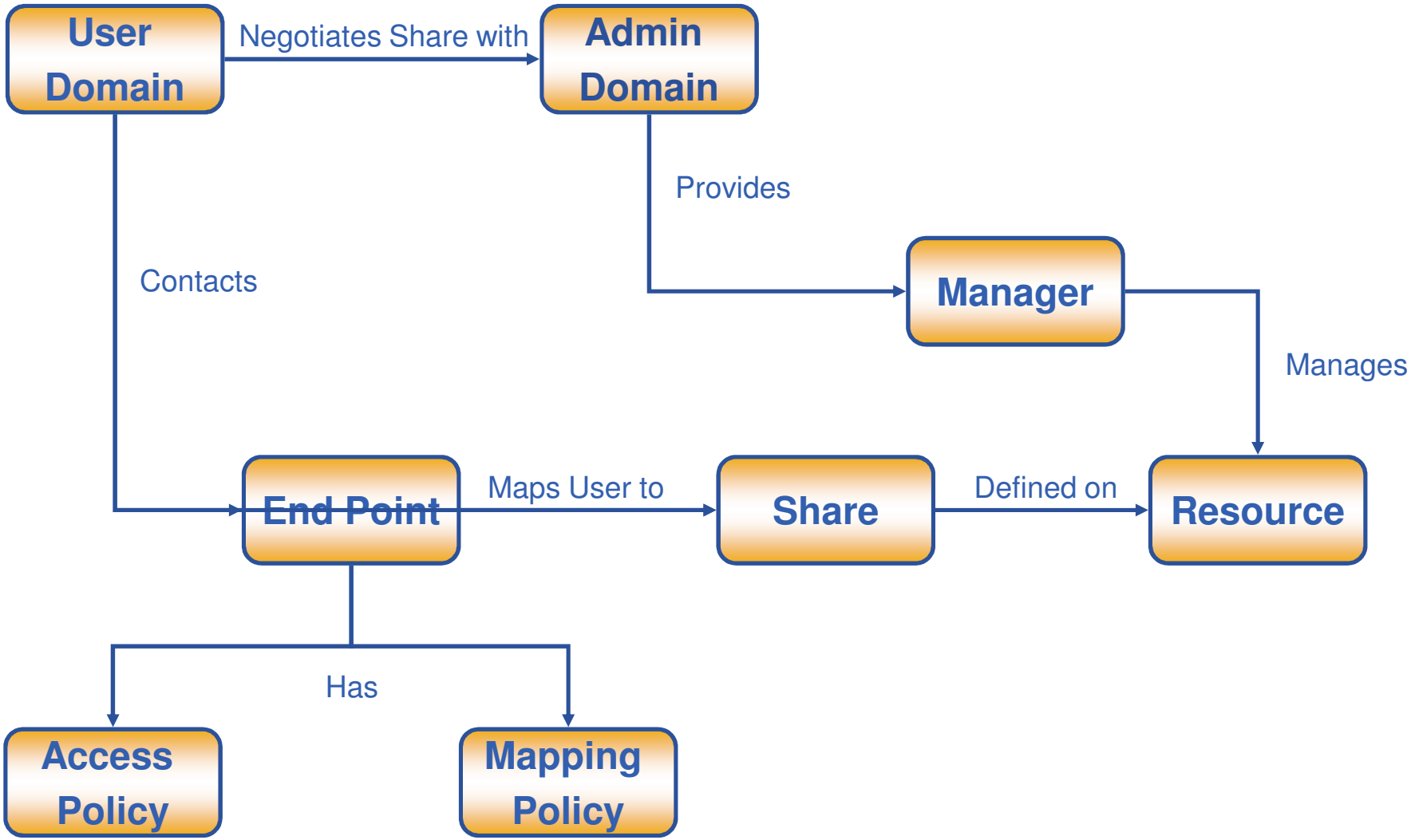
GLUE 2.0 Key Concepts



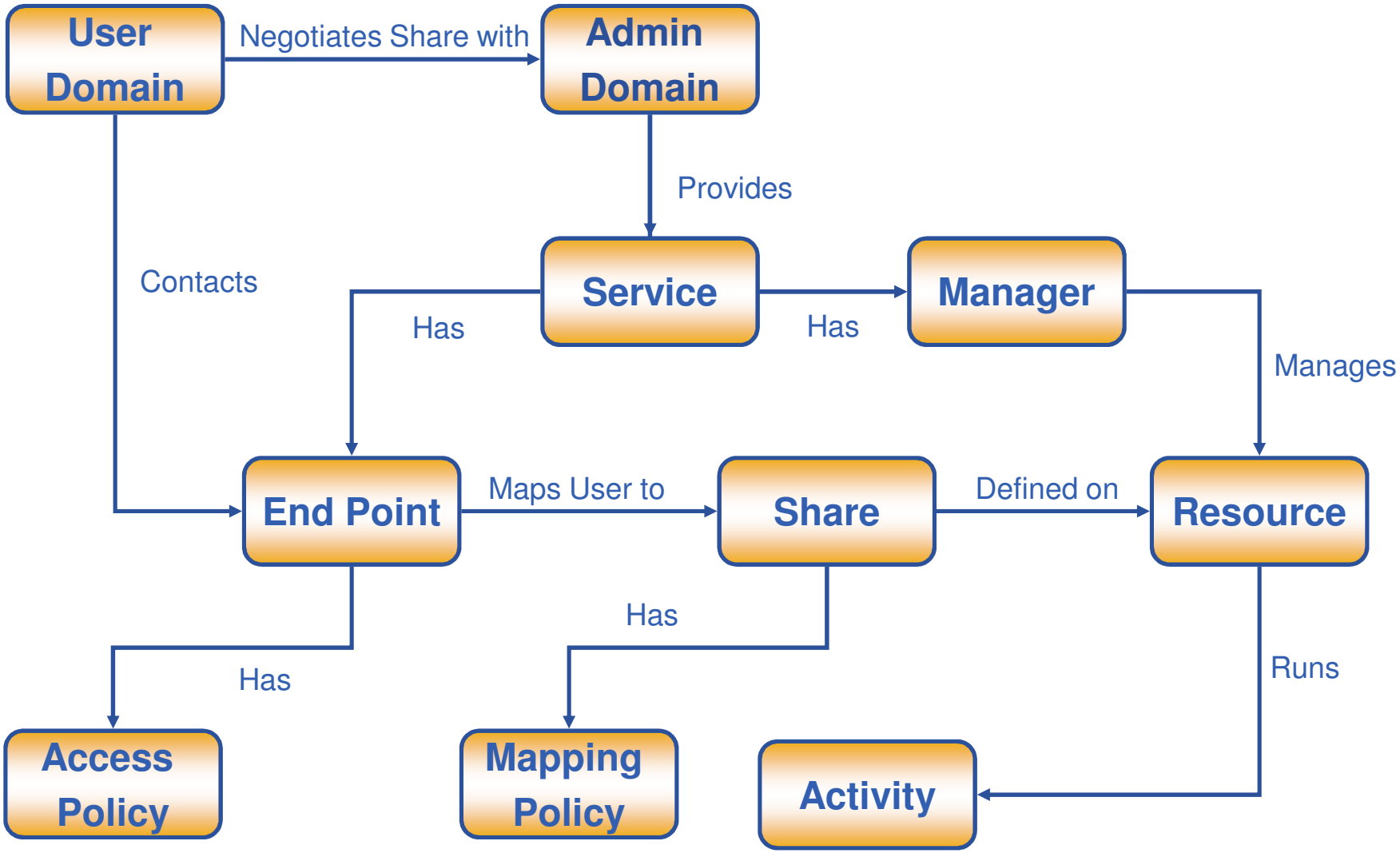
GLUE 2.0 Key Concepts



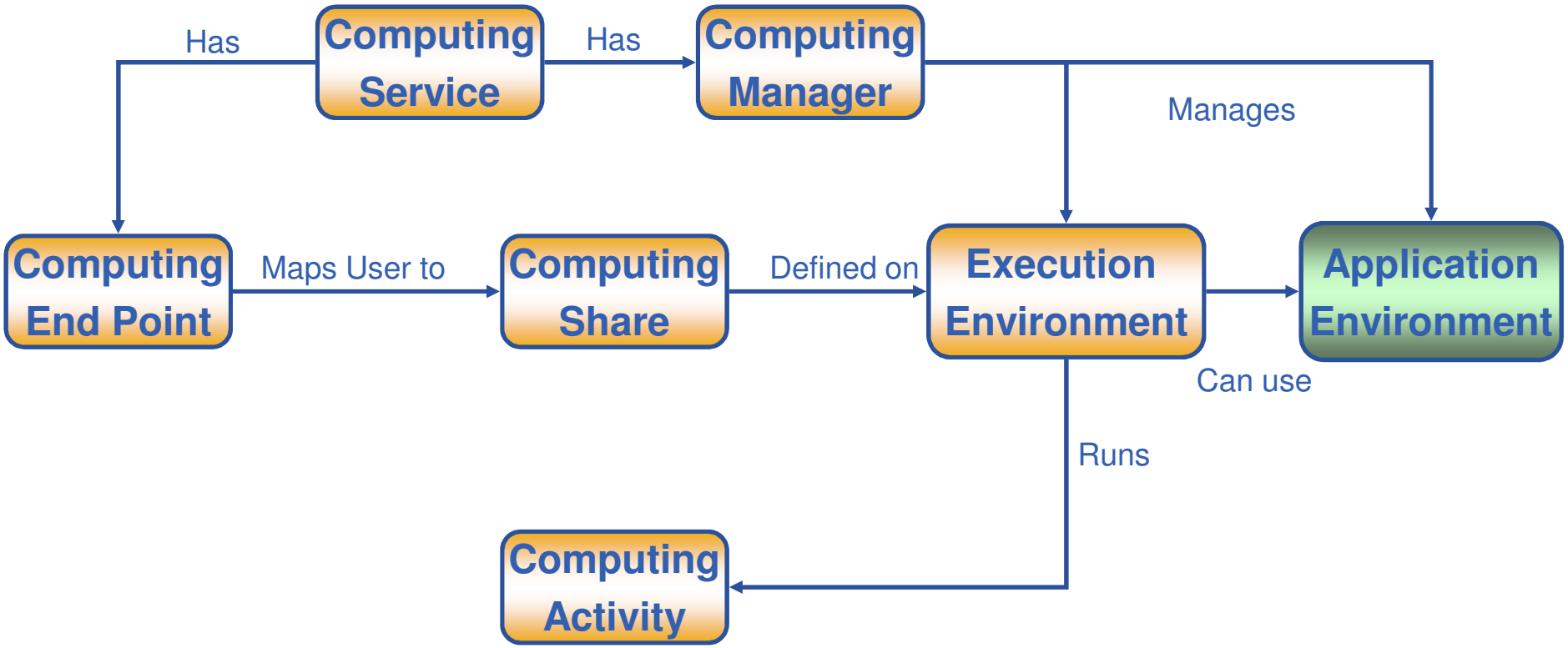
GLUE 2.0 Key Concepts



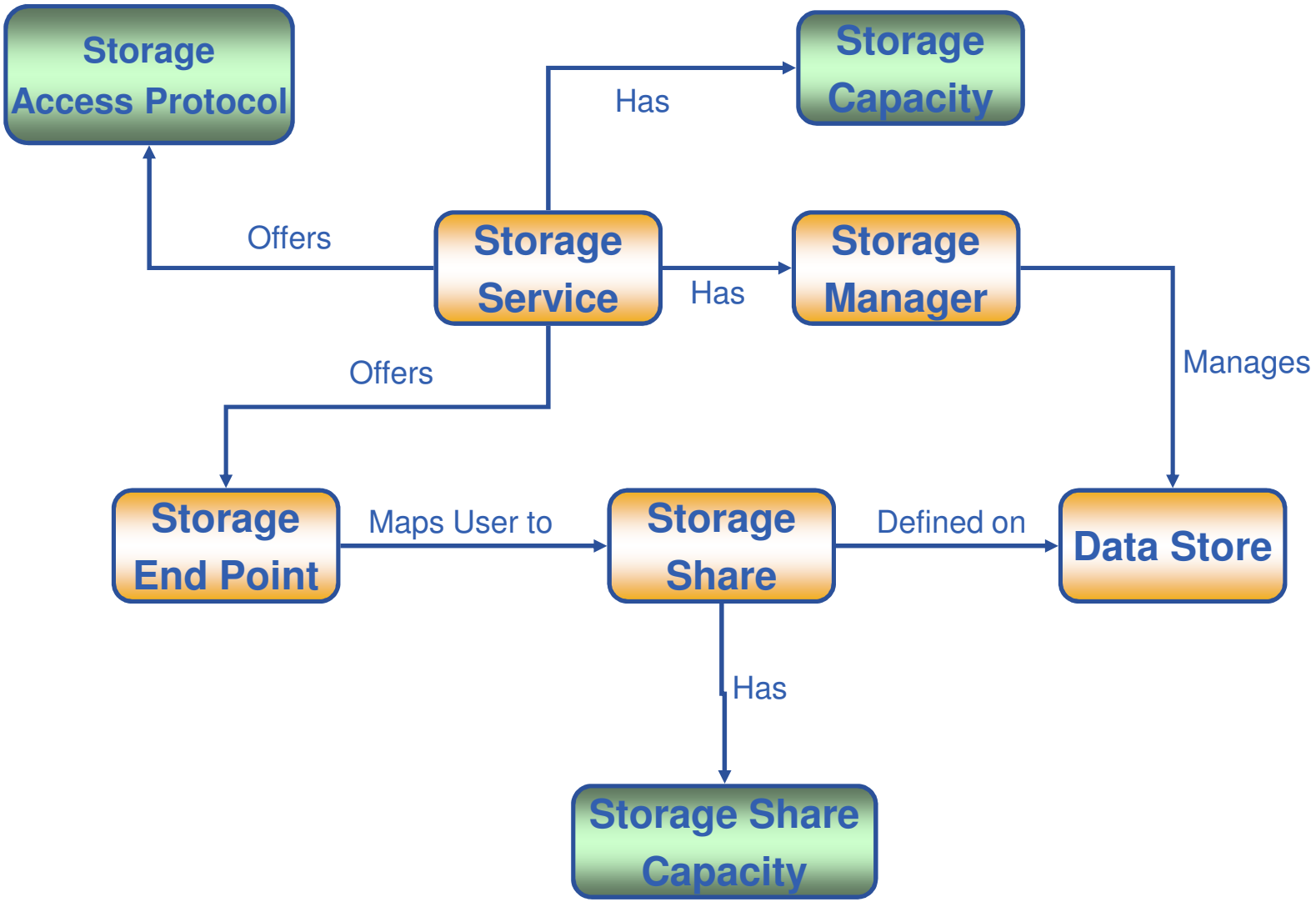
GLUE 2.0 key concepts



GLUE 2.0 computing schema



GLUE 2.0 storage schema



Changes in terminology

- **GLUE 2 looks a bit different to GLUE 1, but most of the concepts are there under different names**
  - Site -> AdminDomain
  - (VO) -> UserDomain
  - Element -> Service, Service -> Endpoint
  - AccessControlBaseRule -> AccessPolicy, MappingPolicy
  - CE, VOView -> ComputingManager, ComputingShare
    - Remove duplication/double counting
  - Cluster/SubCluster -> ExecutionEnvironment
  - (Job) -> Activity
  - SA/VOInfo -> StorageShare
- **Existing attributes should all map to something**
  - Unless they were unused
  - All existing use cases should be met
- **Some new things introduced for use cases from other Grids**
  - EGEE won't publish everything
  - We will have a profile document to specify how we use the schema

Major changes from GLUE 1

- **Generic concept of a Service as a coherent grouping of Endpoints, Managers and Resources**
  - ComputingService and StorageService are specialisations, sharing a common structure as far as possible
  - Generic concepts for Manager (software) and Resource (Hardware)
- **All objects are extensible**
  - Multivalued string “OtherInfo” and/or Key-Value pairs
- **All objects have a globally unique ID**
- **Many objects allow many-to-many relations**
  - More flexible, but more complex
- **Some concepts made more generic/flexible by making them separate objects rather than attributes**
  - Location, Contact, Policy, Benchmark, Capacity
- **More complete/rigorous definitions**
  - Many more enumerated types – but not fully defined yet
  - Placeholder values, case sensitivity, optional vs mandatory
  - People will no doubt still find ambiguities!

Why is this good for EGEE?

- **General structure for any service**
  - CE, SE, WMS, VOMS, MyProxy, LFC, FTS, …
  - Generic service discovery tool
- **Much more expandable**
  - All objects can be extended
  - We always find new cases we didn't anticipate
  - Schema upgrades can take a long time
- **Fixes many long-standing problems**
  - No showstoppers, but many annoying "features"
  - StorageService designed for SRM!
  - ComputingService has a better separation of Grid endpoint, LRMS and queue/fairshare
- **Interoperability and standardisation**
  - Brave new world: EGEE -> EGI
  - May get external implementations

LDAP rendering

- **Needs some basic decisions about how to map the abstract schema to LDAP**
  - Defined in ~ 6 phone meetings in May/June
- **Generally follow GLUE 1 practice, but some changes**
  - Case sensitivity!
  - Some attributes are mandatory
- **Object class naming**
  - Prefix schema class name with “GLUE2”
    - `GLUE2ComputingShare`
- **objectclasses**
  - One objectclass per schema class
  - Derived classes inherit parent objectclasses

```
objectclass: GLUE2Entity
objectclass: GLUE2Share
objectclass: GLUE2ComputingShare
```

LDAP rendering (cont)

- **Attribute naming**
  - Follows the object class naming
    - `GLUE2ComputingShareRunningJobs`
  - Exception for unique ID
    - `GLUE2ShareID` not `GLUE2EntityID`
- **Foreign keys**
  - Only needed at one end of a relation, all relations have a key
  - Generally point logically “up” (child points to parent)
  - Named specifically for the relation they represent
    - `GLUE2ComputingShareComputingServiceForeignKey`
- **DN construction (LDAP tree)**
  - Root -> AdminDomain -> Service -> component -> subcomponent
    - `GLUE2MappingPolicyID=xxx, GLUE2ShareID=xxy, GLUE2ServiceID=xxz, GLUE2AdminDomain=zyx, GLUE2AdminDomain=xyz,o=glue`
  - Dummy grouping object to insert `GLUE2GroupID` anywhere in the DN
    - No semantics, just makes the tree easier to follow, e.g. in an LDAP browser
    - c.f. `mds-vo-name`
  - Should navigate using foreign keys, not DNs

BDII

- **Merged schema, GLUE 1.3 + GLUE 2**
- **Single LDAP server (port 2170)**
- **Separate root DN**
  - `o=glue` vs `o=grid`
- **Resource BDII:** `GLUE2GroupID=resource, o=glue`
- **Site BDII:** `GLUE2AdminDomain=<site-name>, o=glue`
- **Top BDII:** `GLUE2AdminDomain=<site-name>, GLUE2AdminDomain=<grid-name>, (GLUE2GroupID=top?,) o=glue`

- **GLUE2-enabled BDII has been certified and is in PPS**
  - Released to production on Tuesday
  - Nothing published yet!

Information providers

- **Alpha-test provider exists**
  - Drop-in replacement for the existing generic GlueService provider for GLUE 1.3
- **Add support for new attributes, multiple Endpoints per Service**
  - Should be quick – once I get back from Barcelona!
- **Roll out service by service (or all at once?)**
  - Nothing to break, so may be good to get something out quickly
  - Start with the BDII itself?
  - Start certification ~ end of October?
  - Hope to have something in production before the end of the year
- **Computing and Storage providers will be more complex**
  - Use generic Service provider with plugins?
    - Different batch systems, different SRM implementations
  - Need to re-use GLUE 1.3 code where possible
  - Aim to have a first version by the end of EGEE?

Clients

- **All clients need to become GLUE2-aware**
  - Must be backward-compatible
  - Can happen gradually

- **WMS: JDL**
- **Storage: lcg-utils/GFAL/FTS**
- **Service discovery: lcg-info(sites), glite-sd-query**
- **Monitoring: gstat**
- **Accounting: resource accounting, not APEL**
- **Users**

- **Timescale???**

Service discovery

- **Working towards publication for all services**
  - WMS, LCG-CE, CREAM, LB, SRM, FTS, LFC, BDII, MyProxy, Hydra, AMGA, VOMS, R-GMA, VOBOX, Nagios, …
  - In place or in progress
  - Using GLUE 1.3 GlueService/GlueServiceData
    - Somewhat limited in scope
- **Some generic service discovery tools**
  - lcg-info, glite-sd-query
- **GLUE 2.0 allows publication of any service in a generic way, with as much structure as needed**
  - Will need to get some experience
  - Start by mapping existing Service to GLUE 2 Service + Endpoint
  - Add more objects if it seems useful
- **Should be possible to have a more powerful generic query tool**
  - Including Computing and Storage Services
  - Work going on (at RAL) for the OGF SAGA working group

SRM usage

- **SRM v2.2 introduced fairly recently with several new features**
  - "Space tokens"
- **GLUE 1.3 was defined with SRM in mind, but little practical experience**
  - Usage in LCG is still evolving
- **So far we tried to fit publication into the GLUE 1.3 model**
  - Somewhat clumsy but possible
  - Generic SACapability attribute to carry extra information about space tokens
- **Various SRM issues still not totally clear**
  - Logical vs physical view of spaces
  - Treatment of dynamic reservations, unreserved space, staging/cache
- **LCG model doesn't exactly match SRM model**
  - Underlying hardware (tape/disk) vs functional description (custodial/replica) and latency (online/nearline)
  - Scratch space, shared spaces
- **For GLUE 2.0 we tried to cover the use cases we know about**
  - Schema has more flexibility to add new things
    - Which also means more complexity
  - We may still find problems as we implement – GLUE 2.1!

Resource accounting

- **Accounting wasn't an intended use case for GLUE 1 or GLUE 2**
  - Schema has no history, just a current snapshot
  - Fine-grained publication would give too much data volume
  - No encryption or signing of data in the BDII - world-readable
- **For storage LCG currently has no other solution**
  - Can do accounting at the level of space tokens
  - Regular copy to an independent database
- **LCG management want extra information for both computing and storage**
  - Distinguish installed capacity (static) from available capacity (dynamic)
  - VO shares of resources
  - Dealing with multicore CPUs, new benchmark (HEP-SPEC06)
  - Scaling of CPU/wall times in batch system to a reference benchmark
- **New document defines the use of GLUE 1.3 attributes for these cases**
  - Possible but somewhat clumsy
  - Not rolled out yet – watch this space!
- **For GLUE 2 some of these things are explicitly supported**
  - Multiple benchmarks, time scaling, installed vs available capacity, VO Shares
  - Storage accounting will be largely unchanged
  - May have scope to publish things which can't be done in GLUE 1.3
    - ComputingActivity can be a source of usage records for CPU accounting

Summary

- **The GLUE schema has developed over 8 years of practical use**
- **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, but many rough edges and known problems
- **The right time for a major new version**
  - Incorporates all our experience, and input from many other Grids
  - OGF backing makes this a worldwide Grid standard
    - Should help with interoperability and buyin
- **GLUE 2.0 should cover all current use cases for EGEE**
  - And allow things we can't do at the moment
  - And be much more flexible for the cases we still haven't anticipated
- **Starting to roll it out ~now, but the transition process will take several years**

References

- **OGF GLUE working group home page**
  - http://forge.ogf.org/sf/projects/glue-wg
- **GLUE 2.0 specification**
  - http://www.ogf.org/documents/GFD.147.pdf
- **GLUE 1.3 specification**
  - http://glueschema.forge.cnaf.infn.it/Spec/V13
- **“Usage of Glue Schema v1.3 for WLCG Installed Capacity information”**
  - https://twiki.cern.ch/twiki//pub/LCG/WLCGCommonComputingReadinessChallenges/WLCG_GlueSchemaUsage-1.8.pdf