Modeling Storage Resources with GLUE information providers

Jeff Templon, Pierre Girard, Stephen Burke, Peter Kunszt, Ron Trompert

February 3, 2006

1 Introduction

look into using capability attributes to model stuff ...

This document is motivated by problems observed in trying to accomplish two use cases on grid-enabled storage resources. While trying to figure this out, it became clear that

- the current software stack is using only a small fraction of the GLUE information published,
- most of the resources are publishing only a fraction of the GLUE attributes
- the information that is published is often at variance with the GLUE specification, and
- it is not at all clear how one is supposed to model a real SRM-type SE using the GLUE schema.

A good example of this last point is the GLUE `StorageArea` class, which seems to mix virtual concepts like quotas per VO and SURL paths, with physical concepts such as capabilities to pin files onto a disk cache area.

We hope that this document will not only result in a proposal on how to accomplish the use cases, but also in a description of how one should actually model a real storage element using the GLUE schema, and inform the GLUE development process for the next round.

1.1 How to find disk SEs

LHCb wanted to know how to find the space at each site that was 'pure disk', meaning that if they wrote a file there, it would stay on disk and never need to be staged out from tape. Evidently this was not easy via grid methods since the suggested solution was to define two SEs,

- `se-disk.nikhef.nl` and
- `se-mss.nikhef.nl`,

effectively using the SE hostname as storage element metadata.

1.2 How to ensure pinning (in cache area) of files

Several groups have expressed interest in having files, stored in MSS-based systems, being 'pinned' on fast cache areas for long times. It's not clear at all how to do this without contacting a center and specifically requesting the staff to do this.

2 Assumptions

The goal of GLUE seems to be, at the highest level, a description of a virtualized storage resource, independent of the underlying hardware. We try to align our proposal with this goal as long as it remains practical to do so. This also agrees with the goal of the SRM interface, which is supposed to be largely virtual (independent of the underyling storage architecture).

3 Definitions

Reports are that the SRM team put a lot of thought into terminology, so we propose to use that if it really exists. Not all authors have seen the 'definitive SRM nomenclature document' yet but we'll give its existence the benefit of the doubt here.

Please try to use, in these discussions, the terms below only in the sense described below. Using them in other ways is a quite effective technique for creating counterproductive discussions.

SURL Storage URL. We use this here to mean the URL itself, as well as the file it refers to. The existence of a SURL in a given SE just means that it is possible to access that file from that SE (interaction with other SEs is not needed, neither by the user nor the SE in question). However it says nothing about whether the file is permanently on disk, on both back-end slow archival storage and on disk cache space, or only present on the back-end archival store.

TURL Transfer URL. We use this here to mean the URL itself, as well as the file to which it refers. A TURL is a file on an SE that can be accessed immediately. This probably means that it is either located on a disk-based SE (all files are permanently active) or else located on the cache area of a MSS system.

n.b. can we please find some definitive words on what a TURL means in the SRM world? There appear to be some systems in the world that will give you a TURL immediately but this does not imply that the file does not need to be staged in (i.e. DMF would work this way).

online this means that the file is currently located on fast storage, hence one could receive a TURL for it and access it immediately without waiting for it to be staged in from archival medium.

online staging area We've seen 'cache' and 'buffer' used for this term as well, online staging area is SRM terminology. We *think* It refers to the area where the SE creates TURLs when it stages them in from archival medium. SURLs which have corresponding TURLs in the online staging area are online in the sense defined above.

nearline storage the archival medium of a MSS. The implication is that this is slow storage, ie it might take time to get it out into the online staging area and return a TURL.

4 Disk-Based vs. MSS-Based SEs

There is an easy way to distinguish between the two: the Architecture property of the GLUE `StorageElement` class. Allowed values listed in the GLUE 1.2 specification are 'disk, tape, multidisk, other'. 'multidisk' is not being used on LCG at the moment; it refers to disk-based systems that are operating in a raid-like redundant configuration.

It is not clear that looking for Architecture type 'disk' or 'multidisk' is what an experiment really wants to do; what they really want in this case is to put the data somewhere so that it will be quickly accessible for a long time to come. There may be disk systems for which this is not true, *e.g.* a DPM-based disk SE with a single network interface and a limited number of allowed incoming transfer connections; you might have to wait before you are given a TURL.

We can't change the schema in the short term (less than six months time scale) so it seems best to use an attribute that was reserved for grid-project-specific usage: `SEAccessProtocolCapability`. We propose something like

```
GlueSEAccessProtocolCapability: online
```

meaning for this SE, all files are online.

This will solve the LHCb problem, but is non-optimal in the long term since its use implies that *all* files of this SE are online. Current usage among the LCG SRMs indicates that it is much more logical to declare a Storage Area online than an entire SE, so future schema versions should aim to support this, through e.g. a new field under SAState like

```
GlueSAStateOnline: TRUE
```

Another issue is that it is possible to have HSM type and disk-type storage on a single server machine. In order to present this clearly given the current schema, the best way appears to be to have the two machines use a different GlueSEUniqueID. This reemphasizes the point that this attribute should not be the same as the hostname!! The Architecture attribute would define which one is disk and which one tape, and the contact details are in the endpoints for the control and access protocols.

For the long term, it may be better to move the `Architecture` attribute to the `StorageArea` class, as at least one of the current HSMs allows one to have both disk-based and tape-backed areas under a single server instance.

5 What to do about online staging areas

Both SRM and GLUE SE assume they are trying to come up with a virtualized description of a storage resource. Given this goal and the current organization of the GLUE SE schema, it seems we have to conclude that the GLUE `StorageArea` refers to, as it is now (January 2006), a space for SURLs; it is the only part of the SE schema that refers to VOs at all.

A related question, discussed at length during EDG but never resolved, is whether a SURL should uniquely identify a file. If SURL paths look like

```
srm://dm.lbl.gov/my.file
```

then we might have six of these critters, each assigned to a different VO. Things get tricky if we ever need to have VOs access each others' files. Not important for HEP most likely, but probably important for many other groups. On the other hand,

```
srm://dm.lbl.gov/atlas/my.file
```

would work since the VO name is part of the path. Current LCG usage is inconsistent. Here is a list of the various types of (`GlueSAPath`) values currently declared for ATLAS:

```
GlueSAPath: atlas:/atlas
GlueSAPath: atlas:/castor/grid.sinica.edu.tw/sc/atlas
GlueSAPath: atlas:/pnfs/gridpp.rl.ac.uk/data/atlas
GlueSAPath: atlas:/pnfs/gridpp.rl.ac.uk/tape/atlas
GlueSAPath: atlas:/pnfs/usatlas.bnl.gov/data
GlueSAPath: /castor/ific.uv.es/grid/atlas
GlueSAPath: /dpm/scotgrid.ac.uk/home/atlas
GlueSAPath: /grid/atlas
GlueSAPath: /grid/fzk.de/mounts/nfs/data/lcg1/SE00/atlas
GlueSAPath: /hpss/in2p3.fr/grid/atlas
GlueSAPath: /pnfs/grid.sara.nl/data/atlas
GlueSAPath: atlas
GlueSAPath: atlas:/dpm/itep.ru/home/atlas
GlueSAPath: /castor/pic.es/grid/atlas
GlueSAPath: atlas:data2/atlas/
GlueSAPath: /storage/atlas
```

Note the inconsistent usage of a) specifying the VO name in the path part, b) leading '/' characters on the path, c) specification of the site name in the path, and d) including the "voname: " prefix.

We suggest the following: the GlueSAPath is a component (how about "path") in how a program should try to access the file. We expect to be able to use it like this:

```
srm://se-hostname/GlueSAPath/users_file
```

for a SRM based SE, and

```
gsiftp://GlueSEUniqueID/GlueSAPath/users_file
```

for a gridftp-based "classic" SE. The path should contain no `vo:` part nor protocol information.

5.1 General Concerns

In general an SE with back-end nearline storage will have various available quotas for VO/groups on the nearline store(s) (GlueSAs) as well as various available quotas and policies for online staging areas. As far as we can tell there are more questions than answers here, since the schema apparently didn't anticipate any information about online staging areas.

- Common usage: for SRMs in production now, is there a one-to-one mapping between SAs and online staging areas? Is there such a thing as a 'staging space' (chunk of online storage dedicated to a single VO, like a physical disk partition) or is it more like a GlueCE that supports several VOs??

  Answers so far: there are some sites for which there is a one-to-one mapping SA to OSA, and others for which there isn't.

- Do we not even try to model online staging areas? Some sites indicate that they thought it made no sense to do so. Keep this possibility in mind; we assume below that we are going to try and do this modeling.

- Do we limit the online staging area concept to where it makes sense (only in the case of systems with nearline back-ends) or do we insist on uniformity and have 'fake online staging areas' for disk-based SEs? It seems more logical to have disk SEs with no staging object defined.

- How do we model an OSA? Do we define another StorageArea with a 'onlineStagingArea' or 'cache' type? Or do we define a Glue `CacheArea` or `OnlineStagingArea` subclass of the SE schema?

- What do 'pin durations' mean for disk-based SEs? And what do they mean for the other type of SE? Does it mean how long the archival storage is guaranteed? Or does it mean for a surf stored in this SA, how long you are allowed to pin it as a turf in the corresponding CA?

- for the `GlueSAType` parameter, the specification says "guarantee on the lifetime for the storage area". Does this really mean this? If NIKHEF has a StorageArea `cmsflammable`, declared with `GlueSAType: volatile` then is NIKHEF free to delete the entire StorageArea at any time? Or does it refer to files stored under this area? Probably what a user wants to know is "under which area can I put files if I want them to have a certain lifetime property". Not "which StorageArea might completely disappear at any time".

6 Guide to Publishing Information about an SE

In this section we will try to indicate, for the GLUE 1.2 schema, what should be published about an SE, what it all means, and how the middleware should use it.

6.1 GlueSE

This is the top level of the Storage Element GLUE schema.

UniqueID This should be a unique name for the storage element. It is possible that a single machine may host more than one storage element service, so software should not rely on this UniqueID to be the same as the hostname. The `lcg-utils` apparently do rely on this to be the hostname. Note that the schema definition explicitly states that the UniqueID and LocalID attributes "MUST not be interpreted as having any meaning other than as an indentifier. In particular there is no relationship between an ID and a network endpoint."

Name This is some name for the storage element. Current LCG usage shows that the names all have the form

```
GOCDB-SITENAME:archinfo
```

where `archinfo` is one of `classic`, `disk`, or `srm_v1`. If there is any software using this structure (apparently `lcg-utils` does), it should cease doing so as this information is published elsewhere in the schema. We recommend that this SEName should be a free string, unless we find that there is no other way in the schema to represent some piece of needed information.

Architecture current allowed values are

disk the storage medium is one or more physical disks.

mutidisk the storage medium is based on disks in some redundant configuration such as mirroring or RAID-5.

tape the storage medium is tape.

other a medium not listed above.

People wanting to make sure that files do not go to tape could do `GlueSEArchitecture != tape` in their software.

SizeTotal The total amount of storage possible, in **gigabytes**, on this storage element.

SizeFree The total amount of storage, in **gigabytes**, still available for use.

InformationServiceURL the endpoint of the information service publisher for this storage element. A valid example for an info service based on MDS/LDAP/BDII looks like

```
ldap://tbn18.nikhef.nl:2135/mds-vo-name=local,o=grid
```

Note that this attribute follows a different convention than the rest, as it is not prefixed by "SE" in the LDAP translation: it is

```
GlueInformationServiceURL
```

The Glue schema specification does not make this clear, and needs to be rectified to do so.

6.2 GlueSA

Type valid answers here are `permanent`, `volatile`, and `durable`. Recommendation: for a disk SE publish `durable` and for a tape SE publish `permanent`, unless a site has good reason to do otherwise. The meaning of these answers are

permanent The site administrators are not allowed to remove data in this storage area; only the owner of the data may remove them (or inform the site that they are free to remove them). The implication is 'custodial storage' and probably means that the data are on a HSM system.

volatile These data may be removed at the site's discretion at any time, as long as any pins defined on the data have expired.

durable These site administrator may remove these data (after any pins defined have expired) but negotiation with the owner of the data is required first.

In the experiment use cases we've studied so far, 'tape' is synonymous with 'permanent' and 'disk' is synonymous with 'durable'. Probably when real work starts we will have scratch disk SEs with type 'volatile'.

6.2.1 GlueSAPolicy

Quota quota

7 Various

The stuff below has yet to find a place.

Assumptions:

pinning only refers to files in CACHE space, not the files "put" into the SRM. from notes from

http://sdm.lbl.gov/srm-wg/meeting0509/SRM-wg-050914.html

7.1 Definitions of GlueSAType parameter values

from http://www.nesc.ac.uk/events/GGF10-DA/programme/papers/SRMInterfaceSpecification.pdf

Note: "items" below can be files or spaces; not exactly clear about spaces, since the first doc above (sdm.lbl.gov link) has conflicting statements about whether spaces have types or not; clearly GLUE thinks that they DO have types since there is an SAType parameter.

permanent can only be removed by user/VO owning the item

volatile can be removed when its lifetime expires

durable can in principle be removed when lifetime expires, but one must take some action ie either notifying the owners that removal is imminent, or first making an archive copy before removal.

If the above three types can refer to files and spaces, one needs to make clear what the implications are; it is also unclear whether the 'files' on cache space of an SRM have the same types (or types at all) as do the corresponding files on the back end system.