GWD-R, GWD-I or GWD-C
GLUE-WG

Sergio Andreozzi, INFN
Felix Ehm, CERN
Laurence Field, CERN
Balázs Kónya, Lund University
January 17, 2008

GLUE Specification v. 2.0 (draft 19)

Status of This Document

This document provides information to the Grid community regarding the specification of the GLUE information model. Distribution is unlimited. This document is a draft.

Copyright Notice

Trademark

Abstract

Comment [SA1]: To be written

Contents

1. Introduction

In this document, we present a conceptual information model for Grid entities described in natural language enriched with a graphical representation using UML Class Diagrams. As a conceptual model, this is meant to be implementation-independent. Mapping to concrete data models such as XML Schema, LDAP, relational and RDF are provided in the Appendix. From the semantic viewpoint, the concrete data model should represent the same concepts and relationships of the conceptual information model; nevertheless it can contain simplifications specific to the target data model in order to improve query performance or other aspects.

Comment [SA2]: To be added

This information model is based on the experience of several modeling approaches being used in current production Grid infrastructures (e.g., GLUE Schema 1.x [glue-1.x], NorduGrid schema [ng-schema], Naregi model [naregi-schema]). The proposed initial collection of entities is motivated also by the use cases document [glue-usecases].

Comment [SA3]: To be extended

The Information Model and its renderings have to be consider case-sensitive

Comment [SA4]: To be moved in a better place

2. Notational Conventions

Only include this section if applicable.

The key words ‘MUST,” “MUST NOT,” “REQUIRED,” “SHALL,” “SHALL NOT,” “SHOULD,” “SHOULD NOT,” “RECOMMENDED,” “MAY,” and “OPTIONAL” are to be interpreted as described in RFC 2119 [BRADNER1]

3. Conceptual Model of the Main Entities

The GLUE information model proposes a number of main entities and relationships represented in Figure 1.

Comment [SA5]: extend intro

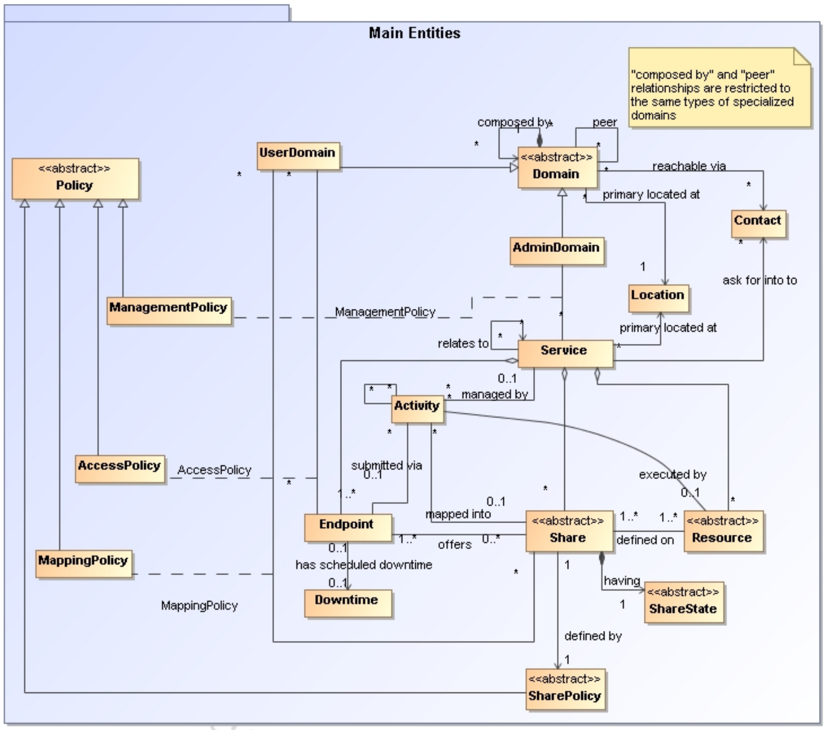


Figure 1 GLUE main entities and their relationships

3.1 Location

Entity	Inherits from			Description
Location				A geographical position
Property	**Type**	**Mult.**	**Unit**	**Description**
LocalID	String	1		An opaque local identifier
Name	String	1		A human-readable name
Address	String	0..1		Street address
Place	String	0..1		Name of town/city
Country	String	0..1		Country name
PostCode	String	0..1		Postal code
Latitude	Real32	0..1	Degree	The position of a place north or south of the equator measured from -90° to +90° with positive values going north and negative values going south
Longitude	Real32	0..1	Degree	The position of a place east or west of Greenwich, England measured from -180° to +180° with positive values going

				east and negative values going west

The location entity is meant to be used for describing reference geographical positions of domains and services. They aim is to provide a simple way to express geographical information and is not intended to be used in complex geographical information systems. The accuracy of latitude and longitude should be defined in an interoperability profile.

3.2 Contact

Entity	Inherits from			Description
Contact				Information enabling to establish a communication with a person or group of persons part of a domain
Property	**Type**	**Mult.**	**Unit**	**Description**
LocalID	String	1		An opaque local identifier
URL	URL	1		URL embedding the contact information. The syntax of URI depends on the communication channel
Type	ContactType_t	1		Type of contact
OtherInfo	String	*		Placeholder to publish info that does not fit in any other attribute. Free-form string, comma-separated tags, (name, value) pair are example of syntax

Comment [SA6]: What about if an email address is used for usersupport and security? (multiple types or decoupling ID from contact info?)

This entity can be used to represent contact information for user support, security, sysadmin. The various types of contact are identified by the Type attribute. In case of time-depend contact information, the instances of this entity should represent only the active contact information.

For telephone and fax: http://www.ietf.org/rfc/rfc2806.txt
For email: http://www.ietf.org/rfc/rfc2368.txt
For irc: http://www.w3.org/Addressing/draft-mirashi-url-irc-01.txt

http://www.ietf.org/rfc/rfc2806.txt

3.3 Domain

Entity	Inherits from			Description
Domain				A collection of actors that can be assigned with roles and privileges to entities via policies. A domain may have relationships to other domains.
Property	**Type**	**Mult.**	**Unit**	**Description**
ID [key]	URI	1		A global unique ID
Name	String	0..1		Human-readable name
Description	String	0..1		A description of the domain
WWW	URL	*		The URL identifying a web page with more information about the domain
OtherInfo	String	*		Placeholder to publish info that does not fit in any other attribute. Free-form string, comma-separated tags, (name, value) pair are example of syntax

Comment [SA7]: Add recommendation from Stephen Burke mentioned document

This is an abstract entity not meant to be instantiated.

3.3.1 AdminDomain

Entity	Inherits from			Description
AdminDomain	Domain			A collection of actors that can be assigned with administrative roles and privileges to services via policies. An AdminDomain manages services that can be geographically distributed, nevertheless a primary location should be identified.
Inherited Property	**Type**	**Mult.**	**Unit**	**Description**
ID [key]	*URI*	*1*		*A global unique ID*
Name	*String*	*0..1*		*Human-readable name*
Description	*String*	*0..1*		*A description of the domain*
WWW	*URL*	*		*The URL identifying a web page with more information about the domain*
OtherInfo	*String*	*		*Placeholder to publish info that does not fit in any other attribute. Free-form string, comma-separated tags, (name, value) pair are example of syntax*
Property	**Type**	**Mult.**	**Unit**	**Description**
Distributed	Boolean	0..1		True if the services managed by the admindomain are considered geographically distributed by the administrators themselves
Owner	String	*		Owner of the managed resources

Comment [SA8]: Add recommendation from Stephen Burke mentioned document

3.3.2 UserDomain

Entity	Inherits from			Description
UserDomain	Domain			A collection of actors that can be assigned with user roles and privileges to services or shares via policies
Inherited Property	**Type**	**Mult.**	**Unit**	**Description**
ID [key]	*URI*	*1*		*A global unique ID*
Name	*String*	*0..1*		*Human-readable name*
Description	*String*	*0..1*		*A description of the domain*
WWW	*URL*	*		*The URL identifying a web page with more information about the domain*
OtherInfo	*String*	*		*Placeholder to publish info that does not fit in any other attribute. Free-form string, comma-separated tags, (name, value) pair are example of syntax*
Property	**Type**	**Mult.**	**Unit**	**Description**
Level	Int32	0..1		The number of hops to reach the root for hierarchically organized domains described by the “composed by” association (0 is for the root)
ManagerEndpoint	URI	*		The Endpoint ID managing the users part of the domain and the related attributes such as groups or roles

Comment [SA9]: Add recommendation from Stephen Burke mentioned document

In the GLUE Information Model, the Virtual Organization can be realized by using the concept of UserDomain. If the VO has an internal structure, this can be represented by using different domains related to each other. A Virtual Organization (VO) comprises a set of individuals and/or institutions having direct access to computers, software, data, and other resources for collaborative problem-solving or other purposes. Resources utilized by a VO are expected to be accessible via network endpoints and constrained by defining utilization targets called shares. The VO can exhibit the internal structure in terms of groups of individuals, each of them being a UserDomain. UserDomains can be hierarchically structured. This structure can be represented via the “composed by” association. A userDomain can be also related to other other userDomains via a “peer” relationship.

Comment [SA10]: We do no have use cases for instantiating the peer relationship; if we won’t have, then we should remove it

As regards the ManagerEndpoint, a commonly used implementation is the VOMS.

Comment [SA11]: Add reference

3.4 Policy

Entity	Inherits from			Description
Policy				Statements, rules or assertions that specify the correct or expected behavior of an entity
Property	Type	Mult.	Unit	Description

This is an abstract entity not meant to be instantiated.

3.4.1 ManagementPolicy

Entity	Inherits from			Description
ManagementPolicy	Policy			Statements, rules or assertions that assign management capabilities to actors as regards a manageable entity
Property	Type	Mult.	Unit	Description

Comment [SA12]: Specify that this is added to have a consistent conceptual model; example implementation in LDAP/XML is parent-child relationship between AdminDomain and Service

The existence of relationship among an AdminDomain and a Service implies that an AdminDomain can manage a Service. Currently, there is no use cases for having attributes in this entity.

3.4.2 AccessPolicy

Entity	Inherits from			Description
AccessPolicy	Policy			Statements, rules or assertions that provides coarse-granularity information about the access by actors to an entity
Property	Type	Mult.	Unit	Description
Scheme	PolicyScheme_t	1		Scheme adopted to define the policy rules
Rule	String	*		A policy rule
TrustedCA	DN_t	*		Distinguished name of the trusted

Comment [SA13]: Add more clarification about why it is coarse-granular

Comment [SA14]: Add basic policy scheme with VO, VOMS FQAN, (ALLOW)/DENY

Comment [SA15]: Evaluate if trustedCA goes together with access control information

				Certification Authority

This entity can be used to express which UserDomains can access a certain service endpoint. The granularity of these policies should be coarse-grained and suitable for pre-selection of services. The actual decision on the service side is performed by an authorization component that can contain a finer-grained set of policy rules that in some case can contradict the published coarse-grained policy rules. Examples of actors involved in this entity are userDomains representing VOs or groups.

3.4.3 MappingPolicy

Entity	Inherits from			Description
MappingPolicy	Policy			Statements, rules or assertions that provides coarse-granularity information about the mapping of activities to shares based on user membership
Property	**Type**	**Mult.**	**Unit**	**Description**
Scheme	PolicyScheme_t	1		Scheme adopted to define the policy rules
Rule	String	*		A policy rule

Comment [SA16]: do we need this? Is it a special case of access policy or a different category?

Comment [SA17]: Add more clarification about why it is coarse-granular

This entity can be used to express which UserDomains can consume a certain share of resources.

3.5 Service

Entity	Inherits from			Description
Service				An abstracted, logical view of actual software components that participate in the creation of an entity providing one or more functionalities useful in a Grid environment. A service exposes one or more endpoints having well-defined interfaces, zero or more shares and zero or more resources. The service is autonomous and denotes a weak aggregation among endpoints, the exposed resources, and the defined shares. The service enables to identify the whole set of entities providing the functionality with a persistent name.
Property	**Type**	**Mult.**	**Unit**	**Description**
ID [key]	URI	1		A global unique ID
Name	String	0..1		Human-readable name
Capability	ServiceCapability_t	*		The capability provided by this service according to the OGSA architecture
Type	ServiceType_t	1		The type of service according to a middleware classification
QualityLevel	QualityLevel_t	1		Maturity of the service in terms of quality of the software components
StatusPage	URL	*		Web page providing additional information like monitoring aspects

Complexity	String	0..1		Human-readable summary description of the complexity in terms of the number of endpoint types, shares and resources. The syntax should be: endpointType=X, share=Y, resource=Z.
OtherInfo	String	*		Placeholder to publish info that does not fit in any other attribute. Free-form string, comma-separated tags, (name, value) pair are example of syntax

The simplest Service is composed by one endpoint, no share and no resource (e.g. a metadata catalog service). In the context of a Service, the same resource part of it can be exposed via multiple endpoints based on defined shares. For instance, in the area of storage systems, SRMv1 and SRMv2.2 interfaces can expose the same resource via different endpoints offering different interface version; in the area of computing systems, the CREAM and GRAM endpoints can expose the same batch system. Endpoints, shares and resources can belong to only one service.

Comment [SA18]: To be verified by real-world use cases

3.6 Endpoint

Entity	Inherits from			Description
Endpoint				A network location having a well-defined interface and exposing the service functionalities
Property	Type	Mult.	Unit	
ID [key]	URI	1		A global unique ID
Name	String	0..1		Human-readable name
URL	URL	1		Network location of the endpoint to contact the related service
Capability	EndpointCapability_t	*		The capability exposed by this interface
Type	EndpointType_t	1		The type of endpoint according to a middleware classification
QualityLevel	QualityLevel_t	1		Maturity of the service in terms of quality of the software components
SpecificationName	String	0..1		Name of the interface specification
SpecificationVersion	String	0..1		Version of the interface
Implementor	String	0..1		Main organization implementing this software component
ImplementationName	String	0..1		Name of the implementation
ImplementationVersion	String	0..1		Version of the implementation (e.g.,

Comment [SA19]: To be verified if we keep both here and in service or only in one part

Comment [SA20]: Suggestion to use URI for identifying categories; Donal will provide examples

Comment [SA21]: What is the relationship between values for this attribute and values for the service.qualityLevel?

				major version.minor version.pathcversion)
HealthState	EndpointHealthState_t	1		A state representing the health of the endpoint
HealthStateInfo	String	0..1		Textual explanation of the state endpoint
ServingState	ServingState_t	1		The serving state (production, draining, queueing, closed)
WSDL	URL	*		URL of the WSDL document describing the offered interface (applies to Web Services endpoint)
SupportedProfile	URI	*		URI identifying a supported profile
Semantics	URL	*		URL of a document providing a human-readable description of the semantics of the endpoint functionalities
StartTime	DateTime	0..1		The timestamp for the start time of the endpoint
IssuerCA	DN_t	0..1		Distinguished name of Certification Authority issuing the certificate for the endpoint
Association End		**Mult.**	**Description**	
Association to UserDomain via Access Policy				

Comment [SA22]: Verify if a single value is enough

Comment [SA23]: to be extended, should capture what is currently called AccessControlBaseRule in GLUE 1.x

For Grid services requiring a richer set of properties for the endpoint, specific models can be derived by specializing from the Endpoint entity and adding new properties or relationships. The current proposal contains the ComputingEndpoint specialization (see Section)

Comment [SA24]: add section reference

3.7 Downtime

Entity	Inherits from			Description
Downtime				A description of a scheduled downtime event
Property	**Type**	**Mult.**	**Unit**	
DowntimeAnnounce	DateTime	0..1		The timestamp for the announcement of the next scheduled downtime
DowntimeStart	DateTime	1		The starting timestamp of the next scheduled downtime
DowntimeEnd	DateTime	0..1		The ending timestamp of the next scheduled downtime
DowntimeInfo	String	0..1		Description of the next scheduled downtime

3.8 Share

Entity	Inherits from	Description
Share		A utilization target for a set of resources offered via related endpoints defined by

Comment [SA25]: shares can be related to each other for instance via hierarchy

				policies and characterized by status information
Property	Type	Mult.	Unit	Description
LocalID [key]	String	1		An opaque local identifier
Name	String	0..1		Human-readable name

This is an abstract entity not meant to be instantiated.

3.9 ShareState

Entity	Inherits from			Description
ShareState				State information for a share
Property	Type	Mult.	Unit	Description

This is an abstract entity not meant to be instantiated.

3.10 SharePolicy

Entity	Inherits from			Description
SharePolicy				Statements, rules or assertions that specify the correct or expected behavior of a share
Property	Type	Mult.	Unit	Description

This is an abstract entity not meant to be instantiated.

3.11 Resource

Entity	Inherits from			Description
Resource				An entity useful in a Grid environment part of a logical service, reachable via one or more endpoints and having one or more shares defined on it. A resource usually represents aggregated information
Property	Type	Mult.	Unit	Description
ID [key]	URI	1		A global unique ID
Name	String	0..1		Human-readable name

This is an abstract entity not meant to be instantiated. For Grid resources requiring a richer set of properties, specific models can be defined by specializing from the Resource entity and adding new properties or relationships. The current proposal contains the Computing Resource specialization (see Section).

Comment [SA26]: add section reference

3.12 Activity

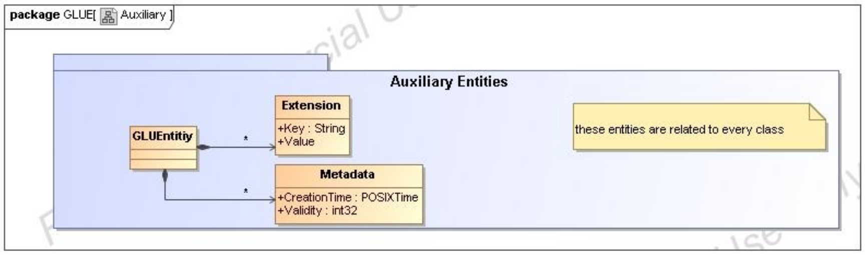
Entity	Inherits from	Description
Activity		An activity is a unit of work managed by a service and submitted via an endpoint; an activity can have relationships to other

				activities being managed by different services, therefore it shares a common context.
Property	Type	Mult.	Unit	Description
ID [key]	URI	1		A global unique ID
Type	ActivityType_t	1		The type of this activity

Grid jobs are example of activities for a Computing Service. An interesting type of relationship for jobs derives from its propagation through several services. For instance, a broker service submits a Grid job to a selected execution service, upon completion the execution service submits a logging record to an accounting service. Each of these services will have associated an instance of a Grid job related to the lifecycle of the job within the service. All instances refer to the same conceptual job submitted by the user.

4. Auxiliar Entities

The auxiliary entities currently provides extensibility mechanisms and metadata applicable to all GLUE entities. Widely used extensions will be considered for addition in future GLUE information model revision as primary properties.


package GLUE[Auxiliary]
Auxiliary Entities
GLUEntitiy
Extension
+Key : String
+Value
Metadata
+CreationTime : POSIXTime
+Validity : int32
*
*
these entities are related to every class

4.1 Extension

Entity	Inherits from			Description
Extension				A key,value pair providing extra information not captured in the current model
Property	Type	Mult.	Unit	Description
Key	String	1		A local ID, typically an attribute name that could be added in future info model revisions
Value	String	*		A value for the attribute

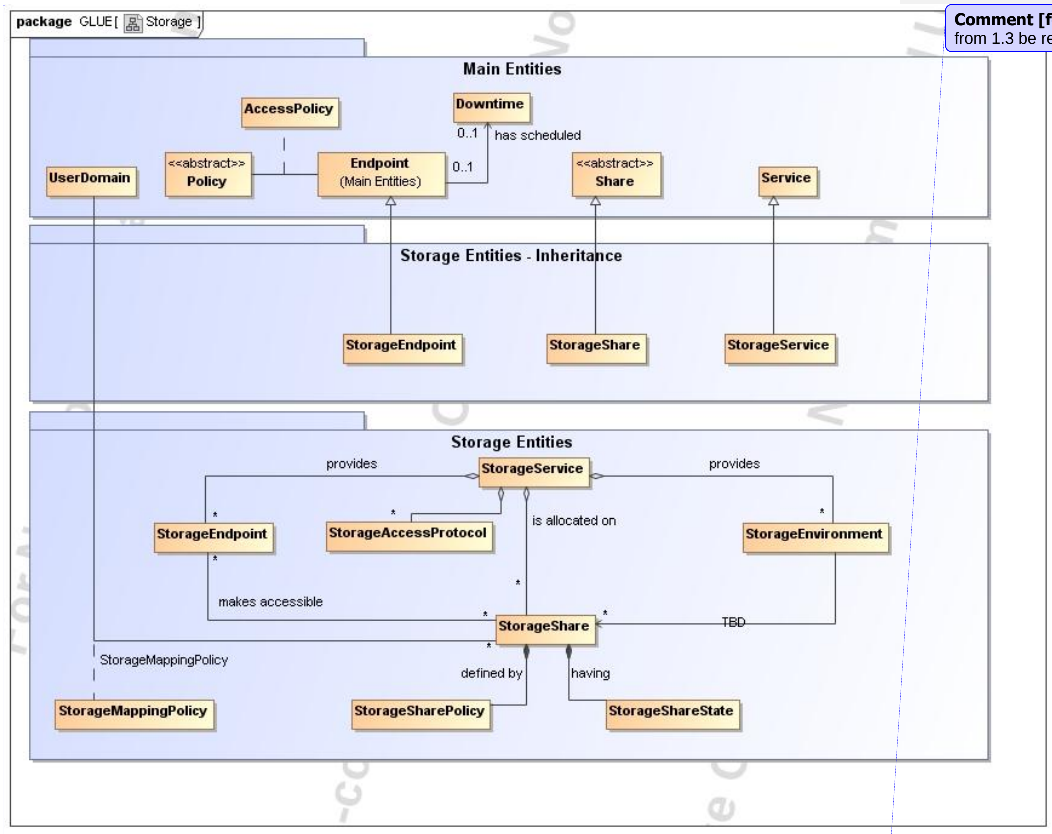
4.2 Metadata

Entity	Inherits from			Description
Metadata				
Property	Type	Mult.	Unit	Description
CreationTime	DateTime	1		Timestamp when the entity instance was generated
Validity	Int32	1	s	The time period for how long the generated

				information is considered to be relevant by the information provider

5. Conceptual Model of the Storage Service

Like the Computing Service, the conceptual model of the Storage Service is based upon the main entities and uses specializations for those entities. Further on, storage related concepts such as StorageShareState, StorageSharePolicy, StorageAccessPolicy, StorageEnvironment and StorageAccessProtocol are introduced.



Comment [fe27]: How should the CESEBind from 1.3 be represented?

Figure 2 Entities and relationships for the Storage Element model

5.1.1 StorageService

Entity	Inherits from			Description
StorageService	Service			An abstracted, logical view of actual software components that participate in the creation of a storage capacity in a Grid environment. A storage service exposes one or more endpoints having well-defined interfaces and one or more storage shares. The service is autonomous and denotes a weak aggregation among endpoints and the defined storage shares. The service enables to identify the whole set of entities providing the storage functionality with a persistent name.
Inherited Property	Type	Mult	Unit	Description
ID [key]	*URI*	*1*		*A global unique ID*
Name	*String*	*0..1*		*Human-readable name*
Capability	*ServiceCapability_t*	***		*The capability provided by this service according to the OGSA architecture*
Type	*ServiceType_t*	*1*		*The type of service according to a middleware classification*
QualityLevel	*QualityLevel_t*	*1*		*Maturity of the service in terms of quality of the software components*
StatusPage	*URL*	***		*Web page providing additional information like monitoring aspects*
Complexity	*String*	*0..1*		*Human-readable summary description of the complexity in terms of the number of endpoint types, shares and resources. The syntax should be: endpointType=X, share=Y, resource=Z.*
OtherInfo	*String*	***		*Placeholder to publish info that does not fit in any other attribute. Free-form string, comma-separated tags, (name, value) pair are example of syntax*
Property	Type	Mult.	Unit	Description
Implementation Name	String	1		The name of the running software
Implementation Version	String	1		The version of the running software
Information Service URL	URI	0..1		A resource location where to retrieve local information
Architecture	String	1		The Architecture this storage management software is running on.
TotalOnlineSize	Int64	0..1	GB	Total storage capacity of the Service
TotalNearlineSize	Int64	0..1	GB	Total storage capacity of the Service on secondary storage
UsedOnlineSize	Int64	0..1	GB	Used storage capacity
UsedNearlineSize	Int64	0..1	GB	Used storage capacity on secondary storage

Comment [fe28]: Do we need to account separate disk/tape usage?

Comment [fe29]: Attributes to be discussed

The storage service is formed by storage endpoints offering interfaces to the service and storage shares which represent allocated storage capacity on the service which can be utilized for storage activities. The access to the endpoint and shares is controlled by a mapping policy instance.
A storage service is instantiated when it offers at least one endpoint. It may have zero or more shares. A storage service without a storage share does not offer any storage capabilities.

The Total/Used-size numbers are capacity status metrics of the storage service. If no secondary storage (like a tape backend) is available on the service the ‘Nearline’ attribute is meaningless and should not be specified.

5.2 StorageEndpoint

Entity	Inherits from			Description
StorageEndpoint	Endpoint, Downtime			Endpoint for accessing and controlling storage activities.
Inherited Property	Type	Mult	Unit	Description
ID [key]	URI	1		A global unique ID
Name	String	0..1		Human-readable name
URL	URL	1		Network location of the endpoint to contact the related service
Capability	EndpointCapability_t	*		The capability exposed by this interface
Type	EndpointType_t	1		The type of endpoint according to a middleware classification
QualityLevel	QualityLevel_t	1		Maturity of the service in terms of quality of the software components
SpecificationName	String	0..1		Name of the interface specification
SpecificationVersion	String	0..1		Version of the interface
Implementor	String	0..1		Main organization implementing this software component
ImplementationName	String	0..1		Name of the implementation
ImplementationVersion	String	0..1		Version of the implementation (e.g., major version.minor version.pathcversion)
HealthState	EndpointHealthState_t	1		A state representing the health of the endpoint
HealthStateInfo	String	0..1		Textual explanation of the state endpoint
ServingState	ServingState_t	1		The serving state (production, draining, queueing, closed)
WSDL	URL	*		URL of the WSDL document describing the offered interface (applies to Web Services endpoint)
SupportedProfile	URI	*		URI identifying a supported profile
Semantics	URL	*		URL of a document providing a human-readable description of the semantics of the endpoint functionalities
StartTime	DateTime	0..1		The timestamp for the start time of the endpoint
IssuerCA	DN_t	0..1		Distinguished name of Certification Authority issuing the certificate for the endpoint
DowntimeAnnounce	DateTime	0..1		The timestamp for the announcement of the next scheduled downtime
DowntimeStart	DateTime	1		The starting timestamp of the next scheduled downtime
DowntimeEnd	DateTime	0..1		The ending timestamp of the next scheduled downtime
DowntimeInfo	String	0..1		Description of the next scheduled downtime
Property	Type	Mult.	Unit	Description
Capability	String	*		Other information regarding this Endpoint

Comment [SA30]: To be verified if we keep both here and in service or only in one part

Comment [SA31]: Suggestion to use URI for identifying categories; Donal will provide examples

Comment [SA32]: What is the relationship between values for this attribute and values for the service.qualityLevel?

Comment [SA33]: Verify if a single value is enough

A StorageEndpoint exposes one interface of how a storage service can be contacted. It gives information about the control protocol and its status as well as possible downtimes.

A storage endpoint is linked to storage shares and thereby knows which shares it gives access to.
The Capability field can be used to specify other restrictions such as WAN read-only/LAN read-write.

5.3 StorageAccessProtocol

Entity	Inherits from			Description
StorageAccessProtocol				Describes the access protocols of a Service.
Property	Type	Mult.	Unit	Description
LocalID	String	1		An opaque local identifier
Type	String	1		The name of the protocol
Version	String	1		The version of the protocol
Streams	Int64	1		The number of parallel streams this protocol supports

5.4 StorageShare

Entity	Inherits from			Description
StorageShare	Share			A utilization target for a set of storage resources defined by policies and characterized by status information
Inherited Property	*Type*	*Mult*	*Unit*	*Description*
LocalID [key]	String	1		An opaque local identifier
Name	String	0..1		Human-readable name
Property	Type	Mult.	Unit	Description
Path	String	0..1		
ExpirationMode	expirationMode_t	0..1		Never, Warn, Release
Tag	String	*		A user defined tag for additional information

Comment [fe34]: Attributes to be discussed

A storage share represents allocated, (to a user domain) dedicated logical storage space within a storage service and can be accessed through the service's endpoint(s).
The access of UserDomains to StorageShares is described by the StorageMappingPolicy.

5.5 StorageEnvironment

Entity	Inherits from			Description
StorageEnvironment				Description of the storage environment of the StorageShare.
Property	Type	Mult.	Unit	Description
ID [key]	String	1		A global unique ID
Type	share_t	0..1		Volatile, Durable, Permanent
AccessLatency	accessLatency_t	0..1		Online, Nearline, Offline
RetentionPolicy	retentionPolicy_t	0..1		Custodial, Output, Replica

Comment [fe35]: Attributes to be discussed

5.6 StorageSharePolicy

Entity	Inherits from			Description
StoragesSharePolicy	SharePolicy			Set of policies that define storage related attributes for share
Inherited Property	*Type*	*Mult*	*Unit*	*Description*

Property	Type	Mult.	Unit	Description
MinFileSize	Int64	0..1	GB	The minimum file size which is allowed in this Share
MaxFileSize	Int64	0..1	GB	The maximum file size which is allowed in this Share
MaxNumFiles	Int64	0..1		The maximum number of files which is allowed in this Share
MaxPinDuration	Int64	0..1	min	Time how long a file stays o the share, before its

				deleted.

Comment [fe36]: Attributes to be discussed

This entity describes the characteristics of a Share in a more detailed way. It is left to the implementations of a StorageService how to handle data in the Share if those attributes are **not** specified.
Example: StorageService A keeps by default the files in Share for 2 days pinned but StorageService B would keep them only for 12 hours.

The 'MinFileSize' attribute allows a StorageService to handle a Share more efficiently.
Example: A Share is defined to be stored on tape permanently. Since it is more efficient in terms of reading time to have 'big' files on the tape it is permitted to store files with a minimum size (e.g. 500MByte).
The MaxFileSize attribute describes the maximum files size this Share can store. This is the case when the underlying storage resource (e.g. file system(s)) can only handle files up to a certain size.

5.7 StorageShareState

Entity	Inherits from			Description
StorageShareState				Describes the State of a StorageShare.
Property	**Type**	**Mult.**	**Unit**	**Description**
free space		1	GByte	The free space left on this Share
used space		1	GByte	The used space of thid Share
total space		0..1	GByte	The total size of this Share
reserved space		0..1	GByte	The reserved size on this share
Status	ShareStatus_T	1		Up / Down / Maintenance

Comment [fe37]: Attributes to be discussed

5.8 StorageMappingPolicy

Entity	Inherits from			Description
StorageMappingPolicy	MappingPolicy			Statements, rules or assertions that specify which instantiation of a Domain may use the associated StorageShare..
Inherited Property				***Description***
Scheme	*PolicyScheme_t*			*1*
Rule	*String*			*
Property	**Type**	**Mult.**	**Unit**	**Description**
LocalID	URI	1		A local identifier for this Policy
Name	String	1		An descriptive name for this Policy
Path	String	1		Path used by VO for writing in an associated Share
Tag	String	1		A user defined tag for this policy

Glue 1.3 : VOInfo
The StorageMappingPolicy describes the relationship of a Userdomain and StorageShare it may access. It keeps further information of how the Userdomain may utilize the StorageShare.

5.9 StorageShare – ComputingShare Relationship

6. Relationship to OGF Reference Model

In this section, we describe the integration of the GLUE information model with the OGF Reference Model.

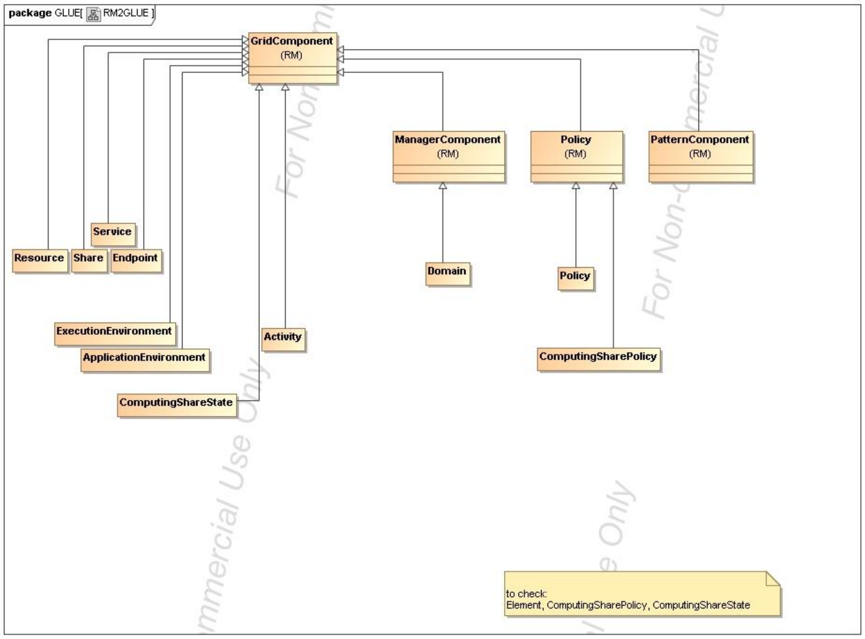


Figure 3 GLUE and Reference Model integration (draft)

7. Template

Comment [SA38]: Describe template

Entity	Inherits from			Description
Property	Type	Mult.	Unit	Description

8. Security Considerations

Please refer to RFC 3552 [RESCORLA] for guidance on writing a security considerations section. This section is required in all documents, and should not just say “there are no security considerations.” Quoting from the RFC:

> “Most people speak of security as if it were a single monolithic property of a protocol or system, however, upon reflection, one realizes that it is clearly not true. Rather, security is a series of related but somewhat independent properties. Not all of these properties are required for every application.
>
> We can loosely divide security goals into those related to protecting communications (COMMUNICATION SECURITY, also known as COMSEC) and those relating to protecting systems (ADMINISTRATIVE SECURITY or SYSTEM SECURITY). Since communications are carried out by systems and access to systems is through communications channels, these goals obviously interlock, but they can also be independently provided.”

9. Author Information

Contact information for authors.

The actual Authors (or Editors) listed on the title page are those committed to taking permanent stewardship for this document – receiving communication in the future and otherwise being responsive to its content. The GFSG recommends at most three Author/Editors be listed on the title page, unless there are compelling reasons to list more.

10. Contributors & Acknowledgements

We gratefully acknowledge the contributions made to this document (in no particular order) by

11. Glossary

Recommended but not required.

12. Intellectual Property Statement

The OGF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the OGF Secretariat.

The OGF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this recommendation. Please address the information to the OGF Executive Director.

13. Disclaimer

This document and the information contained herein is provided on an “As Is” basis and the OGF disclaims all warranties, express or implied, including but not limited to any warranty that the use of the information herein will not infringe any rights or any implied warranties of merchantability or fitness for a particular purpose.

14. Full Copyright Notice

Copyright (C) Open Grid Forum (applicable years). All Rights Reserved.

Comment [HK39]: I don't think it is just "current year." For example, a document wad started to create from 2003, it should be "2003, 2004, 2005" or "2003-2005."

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the OGF or other organizations, except as needed for the purpose of developing Grid Recommendations in which case the procedures for copyrights defined in the OGF Document process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the OGF or its successors or assignees.

15. References

Note that only permanent documents should be cited as references. Other items, such as Web pages or working groups, should be cited inline (i.e., see the Open Grid Forum, http://www.ogf.org). References should conform to a standard such as used by IEEE/ACM, MLA, Chicago or similar. Include an author, year, title, publisher, place of publication. For online materials, also add a URL. It is acceptable to separate out “normative references,” as IETF documents typically do. Some sample citations:

[glue-wg] The Glue Working Group of OGF, *https://forge.gridforum.org/sf/projects/glue-wg*
[glue-usecases] Glue 2.0 Use Cases (early draft), *https://forge.gridforum.org/sf/go/doc14621*
[glue-1.x] The Glue Schema 1.3, *https://forge.gridforum.org/sf/go/doc14185*
[ng-schema] The NorduGrid/ARC Information System, NORDUGRID-TECH
4, *https://forge.gridforum.org/sf/go/doc14273*
[naregi-schema] NAREGI information and data model, *https://forge.gridforum.org/sf/go/doc14300*
[ogf-ts] Technical Strategy for the Open Grid Forum 2007-2010. GFD-I.113.
http://www.ogf.org/documents/GFD.113.pdf
[omii-jra2-djra2.1] Sergio Andreozzi, Antonia Ghiselli, Chunming Hu, Jinlei Jiang, Balazs Konya, Morris Riedel, Davy Virdee, Li Zha. D:JRA2.0 Report on Grid Activities relevant to the identification of new services http://omii-europe.org/OMII-Europe/News/DJRA20.pdf

16. Appendix A: Data Types

16.1 ContactType_t

Open enumeration: security, sysadmin, usersupport, general

16.2 PolicyScheme_t

16.3 DateTime

Extended ISO 8061 format: [-]CCYY-MM-DDThh:mm:ss[Z|(+|-)hh:mm]
This data type maps the XSD dateTime simple type.
We restrict this syntax to GMT timezone: yyyy '-' mm '-' dd 'T' hh ':' mm ':' ss Z

Comment [SA40]: Ad examples or more description

16.4 ServiceCapability_t

List of values initially drafted from [omii-jra2-djra2.1]. To be refined by examples

Security.Authentication	Capacity of providing authentication mechanisms for Grid users machine and services
Security.CredentialStorage	Capacity of providing an online credential repository that allows users to securely obtain credentials when and where needed
Security.Delegation	capacity for a user to give a service the authority to undertake specific activities or decisions on its behalf
Security.Authorization	capacity of handling authorization aspects, making authorization decisions about the subject and the requested mode of access based upon combining information from a number of distinct sources

Security.IdentyMapping	capacity of mapping Grid-level credentials to local level credentials (e.g., mapping a user X.509 certificate into a UNIX account).
Security.AttributeAuthority	capacity of associating a user with a set of attributes in a trusted manner to a relying party, by way of digitally signed assertions
Security.Accounting	capacity of systematically recording, reporting, and analyzing the usage of resources
Data.Transfer	capacity of moving a file from one network location to another. It refers to the actual transfer (e.g., as performed by protocols like FTP, GridFTP, or HTTP)
Data.Management.Transfer	capacity of managing a transfer of files from the start to the completion
Data.Management.Replica	capacity of managing the creation of file replicas upon request
Data.Management.Storage	capacity of managing a storage resource, from simple systems like disk-servers to complex hierarchical systems
Data.Naming.Resolver	capacity of resolving one name to another (for example, search the associated abstract name to a certain human-oriented name)
Data.Naming.Scheme	capacity of attaching names to data resources. (To evaluate if it should moved to the main category infrastructure instead of data). In OGSA, a three-level naming scheme is defined: (1) human-oriented name, (2) abstract name and (3) address
Data.Access.Relational	capacity of providing access to a relational data source

Data.Access.XML	capacity of providing access to an XML data source
Data.Access.FlatFiles	capacity of providing access to a flat file
Information.Model	capacity of modelling resources based on a community accepted definition
Information.Discovery	capacity of locating unknown resources or services, possibly satisfying a set of requirements
Information.Logging	capacity of recording data, often chronologically
Information.Monitoring	capacity of periodically observing measurements, transform them and make available to users or other applications
Information.Provenance	capacity of providing long-term storage of information related to Grid activity and to let this information be accessed by users or other applications.
ExecMan.BES	capacity of executing a job or set of jobs.
ExecMan.JobDescription	capacity of letting users be able to describe a job submission request based on a machine-processable language
ExecMan.JobManager	capacity of managing the execution of a job or set of jobs from start to finish
ExecMan.ExecutionAndPlanning	capacity of building schedules for jobs, that is, the capability of defining mappings between services and resources, possibly with time constraints
ExecMan.CandidateSetGenerator	capacity of determining the set of resources on which a nit of workcan execute
ExecMan.Reservation	capacity of managing reservation of resources for future usage

Open enumeration:

16.5 ServiceType_t

Every item should start with org.MIDDLEWARENAME.

Open enumeration: org.glite.wms, org.glite.lb

16.6 QualityLevel_t

Closed enumeration: production, pre-production, testing, development

16.7 EndpointCapability_t

The initial set of values is drafted from [omii-jra2-djra2.1]. To be refined by examples.

Security.Authentication
Security.CredentialStorage
Security.Delegation
Security.Authorization
Security.AttributeAuthority
Security.IdentyMapping
Security.Accounting
Data.Transfer
Data.Management.Transfer
Data.Management.Replica
Data.Management.Storage
Data.Naming.Resolver
Data.Naming.Scheme
Data.Access.Relational
Data.Access.XML
Data.Access.FlatFiles
Information.Model
Information.Discovery
Information.Logging
Information.Monitoring
Information.Provenance
ExecMan.BES
ExecMan.JobDescription
ExecMan.JobManager
ExecMan.ExecutionAndPlanning
ExecMan.CandidateSetGenerator
ExecMan.Reservation

Open enumeration:

16.8 EndpointState_t

Closed enumeration: OK, Warning, Critical, Unknown, Other

16.9 DN_t

17. License_t

Closed enumeration: opensource, commercial, unknown

18. Appendix B: XML Rendering

In the final section, this page will contain the XML Schema rendering of GLUE 2.0. Meanwhile, the draft schema can be located at the following page:

http://forge.ogf.org/sf/wiki/do/viewPage/projects.glue-wg/wiki/GLUE2XMLSchema

19. Appendix C: LDAP Rendering

In the final section, this page will contain the LDAP rendering of GLUE 2.0 (both schema and Directory Information Tree description). Meanwhile, the draft schema can be located at the following page:

http://forge.ogf.org/sf/wiki/do/viewPage/projects.glue-wg/wiki/GLUE2LDAP

20. Appendix D: Relational Rendering

In the final section, this page will contain the Relational Schema rendering of GLUE 2.0. Meanwhile, the draft schema can be located at the following page:

http://forge.ogf.org/sf/wiki/do/viewPage/projects.glue-wg/wiki/GLUE2Relational