Table of Contents

GLUE 2.0 LDAP DIT & Schema finalization

Introduction

This page contains a list of issues with GLUE2 LDAP DIT that we would like to solve within EMI. The GLUE2 LDAP rendering document, still in a DRAFT state, gives some hints on how to do it. EMI will take decisions on some open topics and define a DIT and schema changes that suits all the LDAP consumers within the EMI middlewares.

This twiki page shows a list of changes organized in three ways:

- NEEDED

  Mandatory changes
- NEEDED-OPEN

  Changes that SHOULD be made, but there is problems to investigate further
- DESIRED

  Changes that MAY be made, but are not that important. Some might contain problems to investigate.

NEW **ChangeLog of the latest schema file**

Summary of changes

The following decisions were taken at the All Hands Meeting 2012 in Budapest after a discussion with ARC and gLite information system experts:

Savannah bug	Change number	Summary	Accepted/Rejected	Action	Status	Testing
BUG:97717	1 and 5	Rename GLUE2GroupID to GLUE2GroupName and `resource` to `services`	Accepted	Schema file change is needed v2.0rc4 BDII configuration change is needed: Define relay in slapd.conf file.	Schema change done BDII done	Tested in BDII and test results sent for verification
	3	Change classes from auxiliary to structural	Accepted	Schema file change is needed v2.0rc4	Done	Tested in BDII and seems OK
	4	Change of OIDs	Accepted	Schema file change is needed v2.0rc4	Done	Tested in BDII and seems OK
BUG:98046	6	String types should be DirectoryString (UTF-8) and not ASCII	Accepted	Schema file change is needed v2.0rc4	Done	To be tested in BDII
-	7	DomainID must be URI	Rejected	ARC may decide to use URIs in any case	-	-
-	2a		Rejected	-	-	-

		Change DNs for specialised objects				
	2b	Additional ID attributes for specialised objects	Rejected	-	-	-

DIT changes

The DIT structure is produced by information providers of each middleware.

1. Changes of attribute values depending on resource, site, top

NEEDED

We would like to have a more clean way to distinguish different level of information. The experience with BDII and ARC identified a three-level structure in which the information can be aggregated in an LDAP tree, and we think is generic enough to represent not just GRID services but any kind of distributed computing environment.

These are the identified levels (taken from a recent review of the OGF rendering draft [LINK]:)

- 1.1 **Local** -level DIT ("resource" insertion point):
  - Right beneath the o=glue root it MUST contain a GLUE2Group entry with **GLUE2GroupID=local**. This entry SHOULD accommodate all the local services, with their complete subtrees
  - Right beneath the o=glue root it MAY contain **GLUE2AdminDomain** or **GLUE2UserDomain** entries.
  - The following groupings were introduced and MUST be placed into the tree as shown on the figure: **GLUE2GroupID=ComputingActivities**, **GLUE2GroupID=ExecutionEnvironments**, **GLUE2GroupID=ApplicationEnvironments**
  - Sample local tree
  - Detailed local tree (incl. objectclasses and DN)
  - TODO:(ARC: **done**)(gLite:mark when done)
    - rename **GLUE2GroupID=resource,o=glue** to **GLUE2GroupID=local,o=glue** on resource-level bdii
    - ARC CE level will include a "GLUE2AdminDomainID=,o=glue" entry, and a **GLUE2GroupID=local,o=glue** containing all CE resources/services

- 1.2 **Domain** -level DIT (affects GLUE2GroupID=resource,GLUE2DomainID=<domain>,o=glue insertion points):
  - Right beneath the o=glue root it MUST contain **only one GLUE2AdminDomain** and MAY have a GLUE2UserDomain
  - The AdminDomain entry MUST have a GLUE2Group entry with **GLUE2GroupID=services** node that SHOULD contain all the service trees from the local level belonging to that domain.
  - Right beneath the o=glue root it MAY contain a GLUE2Group entry with **GLUE2GroupID=local**. This entry MAY be used to describe the LDAP service itself.
  - Sample domain tree
  - Detailed domain tree (incl. objectclasses and DN)
  - TODO:(gLite:mark when done)
    - new insertion point for all resources aggregated by the site-bdii: "GLUE2GroupID=services,GLUE2DomainID=<domain>,o=glue"
    - rename "GLUE2GroupID=resource,GLUE2DomainID=<domain>,o=glue" to "GLUE2GroupID=services,GLUE2AdminDomainID=<domain>,o=glue" on the site-level bdii.

- 1.3 **Global** -level DIT (affects GLUE2GroupID=resource,GLUE2DomainID=<domain>,GLUE2GroupID=grid,o=glue insertion points)::
    - Right beneath the o=glue root it MUST contain a GLUE2Group entry with **GLUE2GroupID=grid**. This entry SHOULD accommodate all the local AdminDomains and UserDomains with their complete subtrees
    - Right beneath the o=glue root it MAY contain a GLUE2Group entry with **GLUE2GroupID=local**. This entry MAY be used to describe the LDAP service itself.
    - Sample global tree
    - Detailed global tree (incl. objectclasses and DN)
    - TODO:(gLite:mark when done)
        - new insertion point for all resources aggregated by the site-bdii: "GLUE2GroupID=services,GLUE2DomainID=<domain>,GLUE2GroupID=grid,o=glue"
        - rename "GLUE2GroupID=services,GLUE2DomainID=<domain>,GLUE2GroupID=grid,o=glue" to "GLUE2GroupID=services,GLUE2DomainID=<domain>,GLUE2GroupID=grid,o=glue" on the top-level bdii for all sites. Beware that due to 1.1, an ARC tree needs its GLUE2GroupID=local insertion point to be rewritten as GLUE2GroupID=services, and a nested under a node with the domain. A detailed picture will be provided about this.

2b. Attributes that are part of the DN

NEEDED-OPEN

The rule to chose such attributes was set as "the first object that inherits from Entity" However this rule leads to strange choices. For example, ComputingActivityID and BenchmarkID become part of the DN, while ComputingServiceID and ComputingEndpointID are NOT. We think it would be simpler and straightforward to simply have the object name as an ID instead of "the first non-abstract object in the inheritance chain"

For example for ComputingService currently ServiceID is the DN, would be nice to have GLUE2ComputingServiceID.

Currently there is no such GLUE2ComputingServiceID attribute, this means that it has to be created in the schema. See task #2a.

PROBLEMS: It might then be necessary to provide both ServiceID and ComputingServiceID in the ComputingService record, which is not favourable. ComputingServiceID MUST replace ServiceID. Can this be done?

7. format and the GLUE2 model (not yet discussed with balazs -- FloridoPaganelli - 16-Oct-2012)

GFD147 (the GLUE2 conceptual model) says that IDs are URIs. In the current LDAP rendering IDs became Strings, for no reason. The GLUE2 model also says (page 5) that IDs MUST NOT be intepreted by the user or the system as having any meaning other than an identifier.

Therefore I would like to use the AdminDomainName and UserDomainName as relevant fields for aggregation, that should actually contain the Domain Name or whatever identifies the Domain in a human world. We can discuss if this field must be part of the DN or not; in this case it might be, so we are consistent with the current status.

for the ID we can use a fictious uri such as urn:ad:

Example:

```
# urn:ad:EMIDomain, glue
dn: GLUE2DomainName=EMIDomain,o=glue
GLUE2EntityCreationTime: 2012-10-16T11:57:08Z
GLUE2EntityValidity: 60
GLUE2EntityName: EMIDomain
objectClass: GLUE2Domain
objectClass: GLUE2AdminDomain
GLUE2DomainID: urn:ad:EMIDomain
```

Schema changes

2a. ID attributes for each specialized object in the rendering (DONE) that do not break backward compatibility

NEEDED-OPEN

DONE : Download GLUE20-v2.0rc5.schema here Warning: breaks backward compatibility!

NEW **DONE - UPDATED** : Download GLUE20-v2.0rc6.schema here changes in schema that do NOT break backward compatibility.

Some important objects like ComputingService or ComputingEndpoint have no specific ComputingServiceID or ComputingEndpointID attribute.

It would be nice to have such attributes to integrate them as part of the DN (See task #2b)

The latest modification of this schema, GLUE20-v2.0rc6, introduces the new attributes but they are MAY, this means, they're optional in a LDIF tree.

3. Change some AUXILIARY objects to STRUCTURAL. (DONE)

NEEDED

DONE : Download GLUE20-v2.0rc1.schema here

In the current schema, objects like ComputingService are AUXILIARY. However, RFC4512 states that

```
"An object class defined for use in the structural
specification of the DIT is termed a structural object class."
```

ComputingService is actually used to define a structure in the DIT, since for example a ComputingService object will have other objects underneath that Service will never have, like ComputingEndpoint. Therefore we propose the following changes:

- ComputingService, StorageService must be declared as

```
SUP GLUE2Service
STRUCTURAL
```

- ComputingEndpoint, StorageEndpoint

```
SUP GLUE2Endpoint
STRUCTURAL
```

7. DomainID format and the GLUE2 model (not yet discussed with balazs --FloridoPaganelli - 16-Oct-2012)

- ComputingShare, StorageShare

```
SUP GLUE2Share
STRUCTURAL
```

- ComputingManager

```
SUP GLUE2Manager
STRUCTURAL
```

- GLUE2ExecutionEnvironment

```
SUP GLUE2Resource
STRUCTURAL
```

Since AdminDomains and UserDomains can have inheritance relationships, I think they should be STRUCTURAL as well.

- GLUE2AdminDomain

```
SUP GLUE2Domain
STRUCTURAL
```

- GLUE2UserDomain

```
SUP GLUE2Domain
STRUCTURAL
```

These changes have been implemented in
https://twiki.cern.ch/twiki/pub/EMI/Glue2LdapStructure/GLUE20-v2.0rc1.schema

4. OID changes (DONE)

NEEDED

DONE : Download GLUE20-v2.0rc2.schema here

A document describing the new assigned OIDs has been uploaded:
https://twiki.cern.ch/twiki/pub/EMI/Glue2LdapStructure/ReassigningGLUE2OIDs.pdf

The OIDs must be changed to allow extension of attributes. GDF147 says:

```
Since it is recommended that each attribute type should be
linked to an object, we can clearly identify attributes as parts
of an object OID subtree.
In the case of inherited objects, we can also identify them
as the parent's object OID subtree.
```

But the above is not applied consistently, it used only for policy and domain.

```
The suggested order is that attribute types should appear first in the OID
tree and object children should appear later in a concrete Object OID subtree.
```

The proposed order is not extensible when it comes to adding new attributes. For example:

objectID = 1.3.6.1.4.1.6757.100.1.1.5.6 assigned to GLUE2Service

objectID = **1.3.6.1.4.1.6757.100.1.1.5.6.1** assigned to GLUE2ComputingService

But in the current schema

attrubutetype **1.3.6.1.4.1.6757.100.1.1.5.6.1** assigned to GLUE2ServiceID

attributetype 1.3.6.1.4.1.6757.100.1.1.5.6.1.1 assigned to ComputingServiceID

Note that GLUE2ComputingService OID == GLUE2ServiceID attributetype!

Moreover if one wants to add an attribute to Service he will have to use existing OIDs assigned to other objects.

We think the current rule for OIDs is too complicated and does not help extensibility.

Proposal:

Flatten the OID numbering to a simple approach lithe this one, without taking into account inheritance:

- Every objectID has an incremental number:
  - reserve the 1.3.6.1.4.1.6757.100.1.1.5.1 for Entity OID.
  - reserve the 1.3.6.1.4.1.6757.100.1.1.5.1.Y for Entity attributetypes, where Y increments with each attributetype.
  - reserve the 1.3.6.1.4.1.6757.100.1.1.5.X for all other GLUE2 OIDs, where X increments with each new entity.
  - reserve the 1.3.6.1.4.1.6757.100.1.1.5.X.Y for entity X attributetypes, where Y increments for each attributetype.

Example:

objectID = 1.3.6.1.4.1.6757.100.1.1.5.6 assigned to GLUE2Service

attributetype = 1.3.6.1.4.1.6757.100.1.1.5.6.1 assigned to GLUE2ServiceID

attributetype = 1.3.6.1.4.1.6757.100.1.1.5.6.2 assigned to GLUE2ServiceName

objectID = 1.3.6.1.4.1.6757.100.1.1.5.7 assigned to GLUE2ComputingService

attributetype = 1.3.6.1.4.1.6757.100.1.1.5.7.1 assigned to GLUE2ComputingServiceID

attributetype = 1.3.6.1.4.1.6757.100.1.1.5.7.2 assigned to GLUE2ComputingServiceName

and so on

These changes have been implemented in
https://twiki.cern.ch/twiki/pub/EMI/Glue2LdapStructure/GLUE20-v2.0rc2.schema

Reference document: https://twiki.cern.ch/twiki/pub/EMI/Glue2LdapStructure/ReassigningGLUE2OIDs.pdf

5. renaming attributes. (DONE)

DESIRED

DONE : Download GLUE20-v2.0rc4.schema here

GLUE2GroupID is NOT an ID. It does not even follow the GLUE2 recommendations for IDs (to be URIs).

Therefore would be better to change the name to GLUE2GroupName or GLUE2GroupLabel. This can be discussed.

The solution presented below allows to seamlessly change the object in a backward compatible way. OpenLDAP will automatically rename all occurrences of the attributes 'GLUE2GroupID' or 'GLUE2GroupLabel' into **GLUE2GroupName**.

```
attributetype ( 1.3.6.1.4.1.6757.100.1.1.8.1.1
        NAME ( 'GLUE2GroupName' 'GLUE2GroupID' 'GLUE2GroupLabel' )
        DESC 'Local unique NAME'
        EQUALITY caseExactIA5Match
        SUBSTR caseExactIA5SubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
        SINGLE-VALUE
 )

objectclass ( 1.3.6.1.4.1.6757.100.1.1.8.1
        NAME 'GLUE2Group'
        DESC 'Group object'
        STRUCTURAL
        MUST GLUE2GroupName
 )
```

6. String types for attributes (DONE) Found 1 issue

NEEDED-OPEN

DONE Download schema v2.0rc3 Found 1 issue

In the LDAP to implementation document it is said that LDAP strings are of type **DirectoryString** (SYNTAX 1.3.6.1.4.1.1466.115.121.1.15). However, this has not been reflected in the schema shipped within EMI, that has **IA5 String** (SYNTAX 1.3.6.1.4.1.1466.115.121.1.26) instead.

PROBLEM: It seems that this have been changed during the creation of the schema for technical reasons. Why is not possible to have DirectoryString? what were the discovered problems?

- Found 1 issue: DirectoryString values cannot be empty. If an ldapadd is performed on an object with an attribute that is empty, slapd refuses to publish the entire object. Reasons I found for this are here:
  http://www.openldap.org/lists/openldap-software/200109/msg00366.html
  Sample error in slapd logs:

  ```
  send_ldap_result: err=21 matched="" text="glue2policyrule: value #0 invalid per syntax"
  ```

SOLUTION: I think we should keep using DirectoryString, but give recommentations. Thefollowing two recommendations for implementors of infoproviders apply:

1. Do NOT publish empty attributes. This is sane; why would one publish an attribute without values??

2. Take extreme care when creating attributes that for the dn. dn should not contain non-utf8 characters. If this happens, infoproviders should be able to handle the strings so to be compliant. Information about this topic can be found in RFC4514. Good news is that the dn is UTF8 friendly, so there's just few characters that are not allowed (like spaces)

This topic: EMI > Glue2LdapStructure
Topic revision: r22 - 2012-11-06 - MariaALANDESPRADILLO