PROCEEDINGS OF SCIENCE

The ARC Information System: overview of a GLUE2 compliant production system

Florido PAGANELLI*
Lund University
E-mail: florido.paganelli@hep.lu.se

Balázs Kónya
Lund University
E-mail: balazs.konya@hep.lu.se

Oxana Smirnova
Lund University
E-mail: oxana.smirnova@hep.lu.se

Mattias Ellert
Uppsala University
E-mail: mattias.ellert@fysast.uu.se

Anders Wäänänen
Copenhagen University
E-mail: waananen@nbi.dk

Information system is the backbone of any kind of a distributed Grid system. NorduGrid ARC was one of the first middlewares that came with a reliable and comprehensive information system architecture. This solution, initially inspired by an LDAP-based approach of Globus, has been used as a distributed dynamic database for Grid resource discovery and monitoring in research infrastructures for many years. This paper gives an overview of the system architecture, the underlying technology and features. Although the original ARC information system came with a custom information schema, the standard GLUE2 model is being broadly endorsed by middleware developers now. ARC contributed to the GLUE2 development, and is among the first to implement it, as described in this paper. Ongoing new developments targeting further convergence and harmonization of otherwise different Grid information systems is also discussed.

EGI Community Forum 2012 / EMI Second Technical Conference
26-30 March, 2012
Munich, Germany

*Speaker.

PoS(EGICF12-EMITC2)074

1. Overview

The paper is organised as follows: Section 2 presents an overview of the existing solutions, Section 3 underlines ARC Information System key concepts and technologies, Section 4 gives a technology overview, Section 5 gives a deeper technical insight, Section 6 shows some of the upcoming integration challenges, Section 7 contains experiences on early GLUE2 adoption, and Section 8 gives a summary.

2. Existing Solutions and Motivation

An analysis conducted mainly by Laurence Field within the EMI project [1] identified similarities and differences between the information systems of gLite [15], ARC [3] and UNICORE [16]. Figure 1 highlights the three different information flow models. The information always flows from the **local or resource** level to the **index or registry or cache** level, but the transfer methods differ.

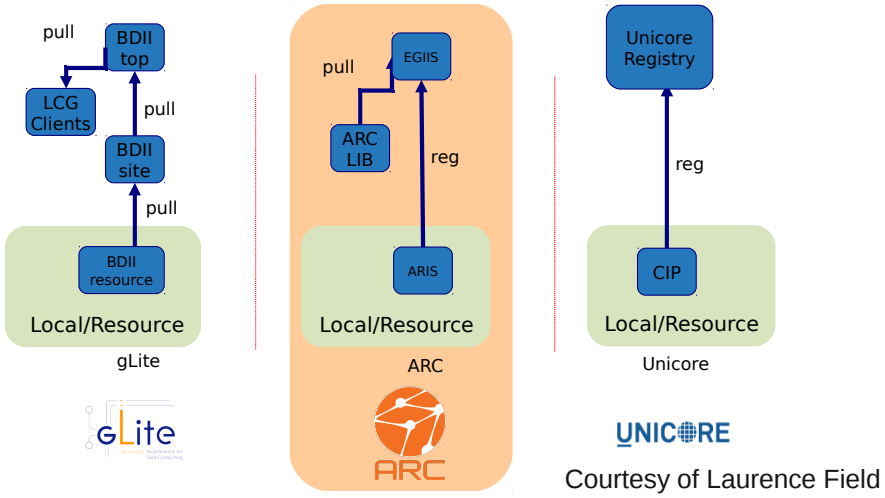

Figure 1: The gLite information system (left) features a pull model, where the information is pulled by upper levels from the level below. Unicore and ARC (center and right) feature a push model where the local level registers to a registry.

ARC [3] features a lightweight push model, where the local information system called **ARIS**, ARC Resource Information System, sits locally on a Computing Element or resource and registers by only sending a small amount of information to an index.

Client and other tools can then use the ARC library **arclib** to query the custom NorduGrid index **EGIIS** [4], Enhanced Grid Information Indexing Service. ARC distributes a collection of client utilities that are able to perform such queries.

Table 1 shows a comparison of the different features for each middleware and the main EMI federated index prototype (EMIR) [2].

Stack	Service Level	Registry	Information Model	Data Model	Global Cache	Transport Model	Federated
gLite	BDII (resource)	GOC DB (Not EMI)	GLUE 1.3	LDIF	BDII (Top)	Pull	Kind of
ARC	ARIS	EGIIS	NorduGrid Schema	LDIF	No	Pull	Not really
Unicore	CIP	Unicore Registry	GLUE 2.0	XML	No	Pull	Not really
EMI	ERIS	EMIR	GLUE 2.0	LDIF	BDII	Pull	Yes

Courtesy of Laurence Field

Table 1: Features of the information systems within the EMI project. Here, `Transport Model` refers to the way information consumer or a client is supposed to get information from the information system as a whole.

2.1 Motivation for a more versatile ARIS

Grid information systems are currently in a process of transition, converging to a common information system that will make use of the GLUE2 model [13]. To ease the introduction of such a model and preserve backwards compatibility with existing models, ARIS has been enhanced to be able to interact with information system components from other middlewares. In this way, ARC implements federation of resources across different domains and different technologies, which has always been one of the main drives for Grid computing.

A key concept in ARC is that clients perform resource discovery, matchmaking and monitoring. ARC own clients always relied on the information system to choose where a job can be submitted, and to get Computing Elements usage information; with the approach of offering multiple information system interfaces on the Computing Element (CE), clients will be able to discover and monitor resources of a cluster with ARC CE in several ways, which has the following benefits:

Reliability: if one of the information system interfaces is down, it is possible to try another; moreover, clients not supporting an interface can find one that fulfill their needs;

Fine-grained resource discovery: different models present different kind of information and features.

Ease of coding new clients: ARIS uses well-known standardised interfaces.

3. ARC Information System Key Concepts

ARC information system is based on four key concepts:

Freshness of information. Information must be **as up to date as possible**. Therefore, most of the ARC information must be consumed on the **local or resource level**, which is where the information originates.

Lightweight index. The index level must contain the **minimum information needed** to reach the local levels of interest. ARC index can be viewed as a collection of URLs pointing at resource level information systems.

No caching. As information must be fresh, ARC index **does not cache resource level information** in the index level. Instead of index level caching, information at the local level is periodically refreshed, and only the minimum information on how to contact the local level is sent to the index.

Multiple interfaces. ARIS is capable of **speaking all the possible information system "dialects"**. This is achieved by supporting three information models (NorduGrid/ARC, Glue1.2/1.3 and GLUE2), two GLUE2 renderings (LDIF and XML) and multiple server protocols (LDAP, WSRF, extended BES and EMI-ES).

As a consequence of these principles, clients or consumers willing to retrieve information must always end up contacting a local information system.

The action of collecting information about services a client wants to use is called **resource discovery**. A basic resource discovery algorithm is as follows:

1. Client contacts index level and retrieves a list of local information system URLs;
2. For each URL, the client contacts the local information system accessible by that URL directly;

After a successful resource discovery, using all the information gathered, the client decides which service to contact, depending on its needs.

4. Technologies overview

ARC Information System is a collection of custom crafted tools and several off-the-shelf products available on the Open Source market.

The whole information system consists of the following:

A customized LDAP setup that implements both local and index level. It consists of the custom *NorduGrid schema* [4] and a custom LDAP-based index server called *EGIIS* [4].
It also consists of the Open Source LDAP server *OpenLDAP* [5] publishing LDAP information, and of a collection of Open Source Berkeley DB update scripts (*bdii-update*). Registration to an EGIIS index is achieved via custom scripts.

A Web Services interface based on the custom Web Service container HED (Hosting Environment Daemon or *arched*) [6] to serve local information and perform RESTful registration.

A collection of Perl scripts – the *infoproviders* [7], capable of collecting information at the local level and produce representations in the form of XML or LDIF documents.

The two interfaces (LDAP and Web Service based) can live together but it is not mandatory to have both enabled. ARIS is fully functional even if just one of them is running.

4.1 ARC information system "speaking" only the ARC "dialect"

ARC information system architecture is shown in Figure 2.

The lower level represents the local or resource level, where ARIS operates. In a complete ARC environment, various instances of EGIIS are organized as an hierarchy of index-level services.

ARIS registers to a known EGIIS by sending small amount of information, performing simple standard LDAP operations.

Then, a client requesting information from any EGIIS must perform a custom LDAP query that will return a collection of EGIIS or ARIS endpoints/URLs.

The client will then contact the local endpoints directly using standard LDAP protocol queries to gather further information about local resources, or perform other custom queries to other EGIIS servers.

Width of arrows in Figure 2 is meaningful: amount of information exchanged between ARIS and EGIIS and between clients and EGIIS is quite small, while exchange between clients and ARIS carries most of the available information, hence the thicker arrow.

4.2 ARC information system "speaking" other "dialects"

One of the purposes of ARIS is to easily connect and share information with multiple indexing and caching systems. Figure 3 shows how it interacts with existing caching technologies (BDII-top [8]) and the upcoming EMI Registry EMIR [2].

The orange band shows the ARC ecosystem, consisting only of ARC components. The blue band shows those software products that don't belong to ARC, such as gLite BDII-top and EMI's EMIR.

In the case of gLite, a pull model is used to gather information from an ARIS instance. Basically, BDII-top will have a list of URLs to contact the various LDAP servers for different services. By querying directly the ARIS local information system, it will pull all the data needed for caching, using standard LDAP searches. The thick arrow here shows that a large amount of information is transferred between ARIS and BDII-top. Note that the arrow between a client and a BDII-top is thick as well, because BDII-top is a cache and contains a lot of information that can be directly fetched.

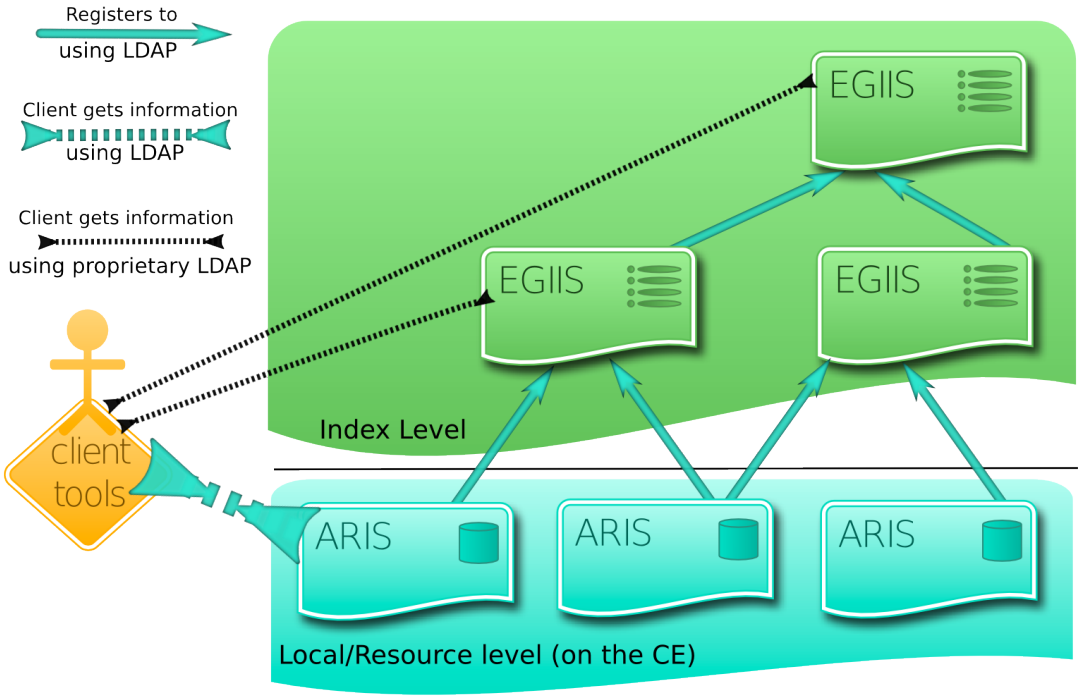

Figure 2: ARC information system "speaking" only the ARC "dialect".

In the case of EMIR, the registry is meant to contain minimal static GLUE2 endpoint information. The communication protocol towards EMIR is a RESTful Web Service interface. ARIS is capable of contacting EMIR and sending a small amount of information regarding its GLUE2 services and endpoints. Note that, as in the case of ARC index EGIIS, the amount of information the client gets from EMIR is the minimal information necessary to access services on a Computing Element where ARIS is running.

5. Technologies: a detailed look

In this section we will describe in more detail ARIS components, supported schema and renderings, and the algorithm with which information is generated and served.

The process of communicating to different index levels is explained in Section 5.3.

5.1 Components

In what follows, we describe ARIS internals shown in Figure 4. *arched* is ARC's own Open Source Web Service container. Written in C++, it is extensible and allows running Web Services with different purposes. Job submission interfaces are served through *arched*, and they usually allow information system requests. A-REX, the ARC computing element and job management service, exposes three interfaces: WSRF, XBES (eXtended BES) and EMI-ES (EMI Execution Service). While WSRF is basically used to obtain an XML information document, XBES and EMI-ES are well defined interfaces that feature information system retrieval functionality. All this information is generated from an XML document.

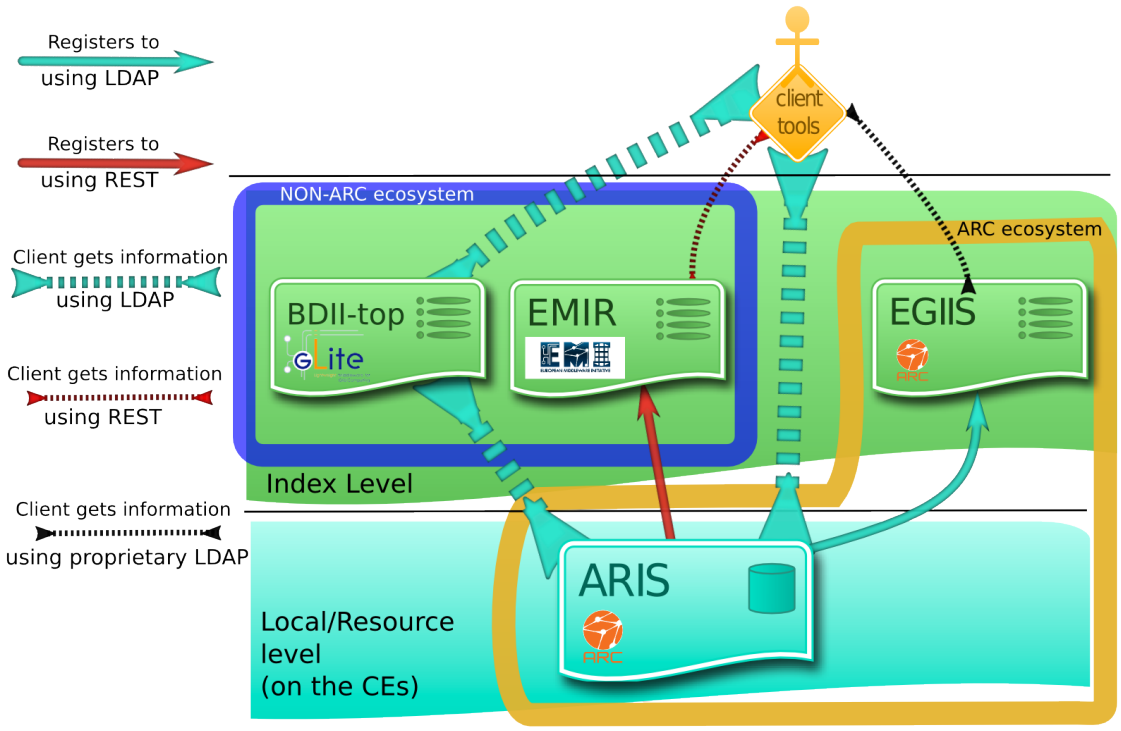

Figure 3: ARIS "speaking" other "dialects"

LDAP information is served via the Open Source LDAP server OpenLDAP and its daemon *slapd*. The *slapd* daemon manages and serves information contained in Berkeley databases. LDAP information is described in terms of LDIF trees or LDIF documents [9]. Such information is updated by LDAP protocol [10] operations.

Another Open Source external component called *bdii-update*, a Python script, takes care of gathering LDIF files representing database data, and merges them in an efficient way in order to perform add, delete and update LDAP operations against the *slapd* server.

The *infoproviders* are set of Perl scripts, developed by the ARC team, that can generate XML and LDIF documents needed by the former two subsystems. The information contained in these documents is shaped according to three main schemas: NorduGrid [4], Glue 1.2/1.3 [12] and GLUE2 [13]. In particular, GLUE2 is rendered in both XML and LDIF, while NorduGrid and Glue 1.2/1.3 only come as LDIF. All these can be published together. A fully enabled ARC CE can, at the same time, publish all the combinations of schemas and protocols listed in Table 2.

The two startup scripts (*a-rex* and *grid-infosys*) configure all the subsystems and the start them. It is important to note that even if ARC uses off-the-shelf components such as OpenLDAP and *bdii-update*, these are configured in one single central configuration file (*arc.conf* [11]) in such a way that configuration and startup of these services when working together with ARC is independent from their canonical system setup. This means that an organisation can seamlessly run an LDAP server or a BDII system together with an ARC CE without changing a single line of their distribution-based configuration. ARC will configure and run special instances of those for itself.

Schema	**Protocol**
NorduGrid	LDAP
Glue 1.3	LDAP
GLUE2	LDAP
GLUE2	WSRF
GLUE2-based	XBES
GLUE2-based	EMI-ES

Table 2: The ARIS schemas and protocols they can be queried with. GLUE2-based means that the protocol might change the GLUE2 data structure according to the protocol needs.

5.2 High-Level Algorithm

The high-level algorithm with which the components interact is as follows:

1. Concurrently,
   (a) *a-rex* startup script configures and starts the *arched* daemon.
   (b) *grid-infosys* startup script configures and starts the OpenLDAP *slapd* daemon and the *bdii-update* script, then starts the whole LDAP subsystem.
2. *arched* periodically executes the *infoproviders*.
3. the *infoproviders* generate an XML document and a set of scripts able to generate LDIF trees, compliant to different schemas (see Table 2)
4. Concurrently,
   (a) *arched* takes the XML document and wraps it to serve GLUE2 information over all the supported Web Service based protocols (WSRF, XBES and EMI-ES)
   (b) the *bdii-update* scripts use the LDIF generator scripts to update information served by *slapd*.

Figure 4 shows the components involved in the process.

5.3 Connection to the Index level

As mentioned in Section 4.2, ARIS can talk to different index level information systems.

This is achieved by the different subsystems enforcing the technologies the index level supports.

Figure 5 shows which interface is responsible for communication to different index technologies, the reader is referred to it while reading the description below.

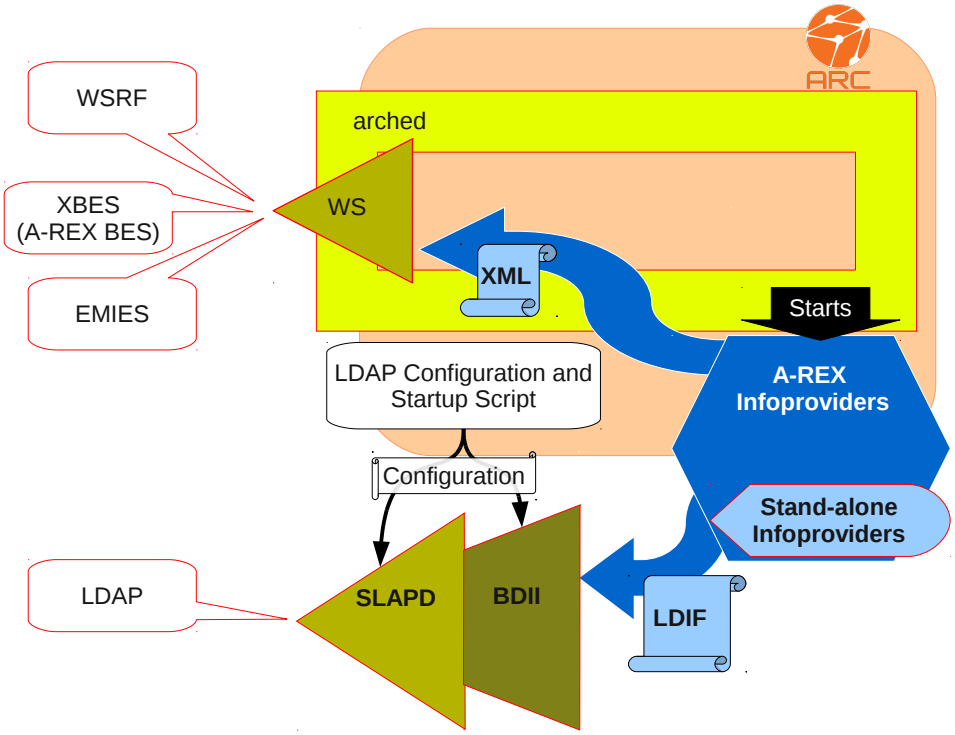

Figure 4: ARIS components. The orange container surrounds the ARC components. The *grid-infosys* startup script provides configuration files for *slapd* and BDII. BDII represents the Berkeley DB update scripts. Stand-alone infoproviders can be either components external to ARC or legacy infoproviders that existed prior to ARC v.1.0.0.

5.3.1 Registering to an EGIIS

ARC index EGIIS is a custom LDAP server using the Globus-MDS based registration schema. Registration of a cluster to EGIIS is achieved by sending simple *ldapadd* requests to an index where the cluster is authorised.

Registration is actually performed by a set of utilities that run from time to time. The time interval can be defined by the system administrator. These scripts are independent from *arched*, and are configured and executed by the *grid-infosys* startup script.

EGIIS was designed to be a lightweight index, so the registration record contains minimal amount of information needed to reach the local resource level.

5.3.2 Caching information on a BDII-top

The BDII-top pull model expects an LDAP server serving information according to the Glue 1.3 or GLUE2 schema. ARIS features an LDAP server capable of providing such information.

For this kind of caching to work, the BDII-top must know the ARC LDAP server's endpoint URL. The BDII-top will then perform simple LDAP queries against ARIS's *slapd* daemon to get the chunks of LDAP trees needed.

Details of this mechanism will be described in Section 6.

In the case of the BDII-top, a huge amount of information is pulled from the local level to the index level.

5.3.3 Registering to an EMIR service

EMIR [2] is a novel Endpoint Registry architecture being developed within the EMI project [1]. Its relevant features considered in this paper are a RESTful interface for communication and the use of GLUE2 dialect to describe services and endpoint URLs records. The communication follows a push model, in which a registrant willing to be in the index pushes information via a RESTful interface.

By leveraging the *arched* versatile Web Service container, an ARC CE is able to send registration records by reshaping the XML output of inforproviders and creating the expected RESTful message.

As in EGIIS, very small amount of information is expected to be sent to the registry, namely, services and their endpoint URLs.

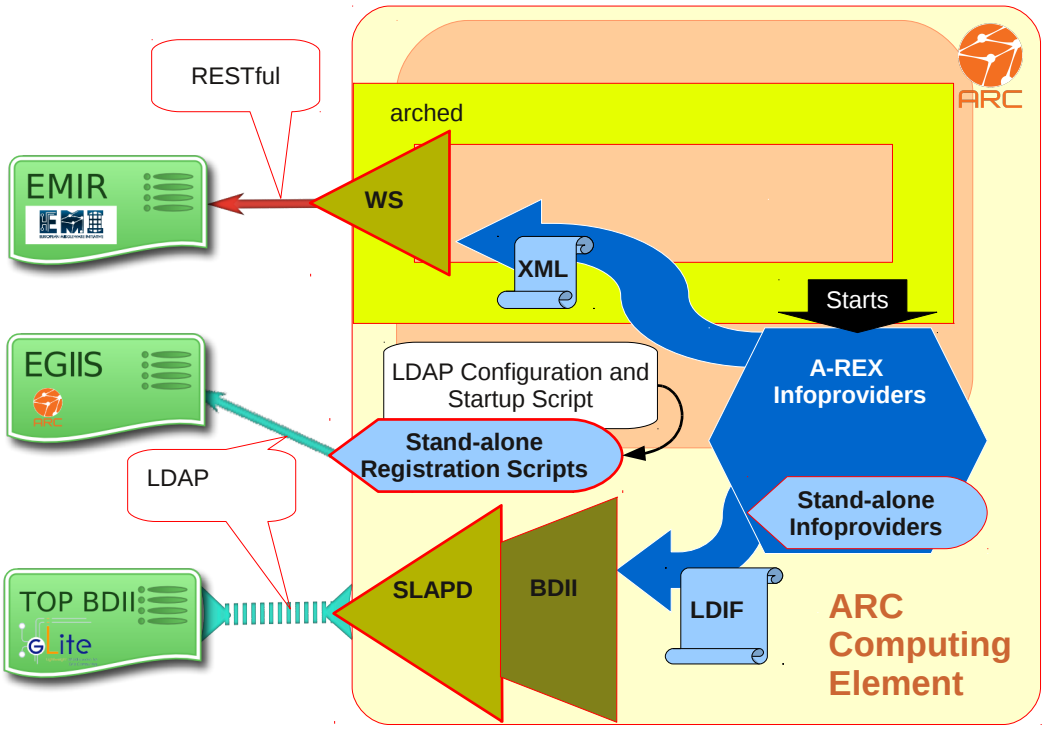

Figure 5: ARIS and other index level technologies. The size of arrows in the figures is important: information towards EGIIS and EMIR is minimal and unidirectional, while information between BDII-top and ARIS is huge and uses LDAP queries.

5.4 The information trees

In this section we give a brief overview, in pictures, of how the ARC information trees for different renderings (XML or LDIF) of the GLUE2 schema look like. At the time of writing, there

are no official trees defined, therefore these figures represent only a proposal that is subject to change in the future.

Figure 6 shows how does the LDAP root look like, Figure 7 shows how the GLUE2 schema is rendered by our proposed LDIF, and Figure 8 shows how the GLUE2 schema is rendered by our proposed XML document and its main differences from the LDIF rendering.

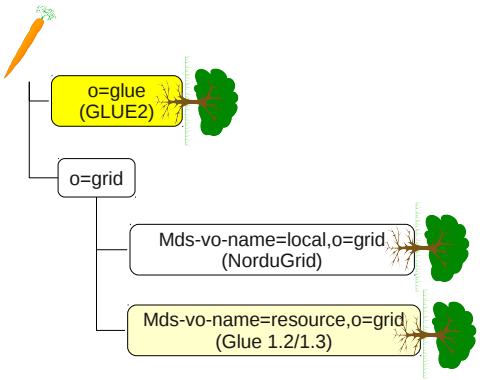

Figure 6: The LDIF/LDAP tree root. It includes all schemas. *o=glue* is the root of the GLUE2 tree, *mds-vo-name=local,o=grid* is the root of the NorduGrid schema, *mds-vo-local=resource,o=grid* is the root of the Glue 1.2/1.3 schema. They can all be published simultaneously.

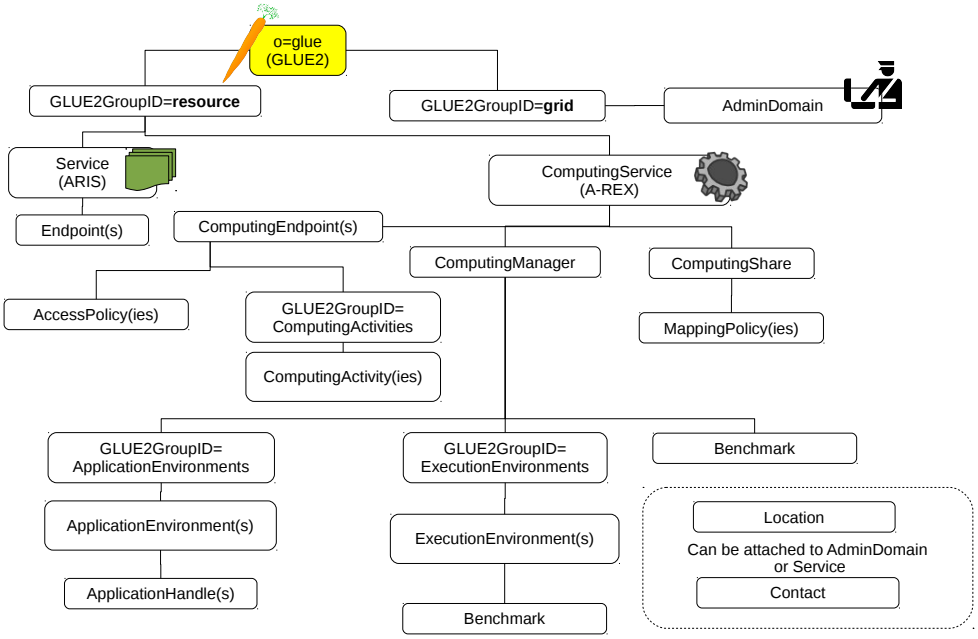

Figure 7: The ARC GLUE2 LDAP tree. The proposed structure has been divided into two branches. The choice is to separate administrative-bureaucratic information (the *GLUE2GroupID=grid* branch) containing the *AdminDomain* and *UserDomain* GLUE2 objects from technical-resource branch (*GLUE2GroupID=resource*) presenting all the needed information to perform job submission and management choices. Please note that in ARC, job information is also published, in the form of GLUE2 *ComputingActivities*. Note also that the tree structure is subject to change depending on future OGF GLUE2 Working Group agreements.

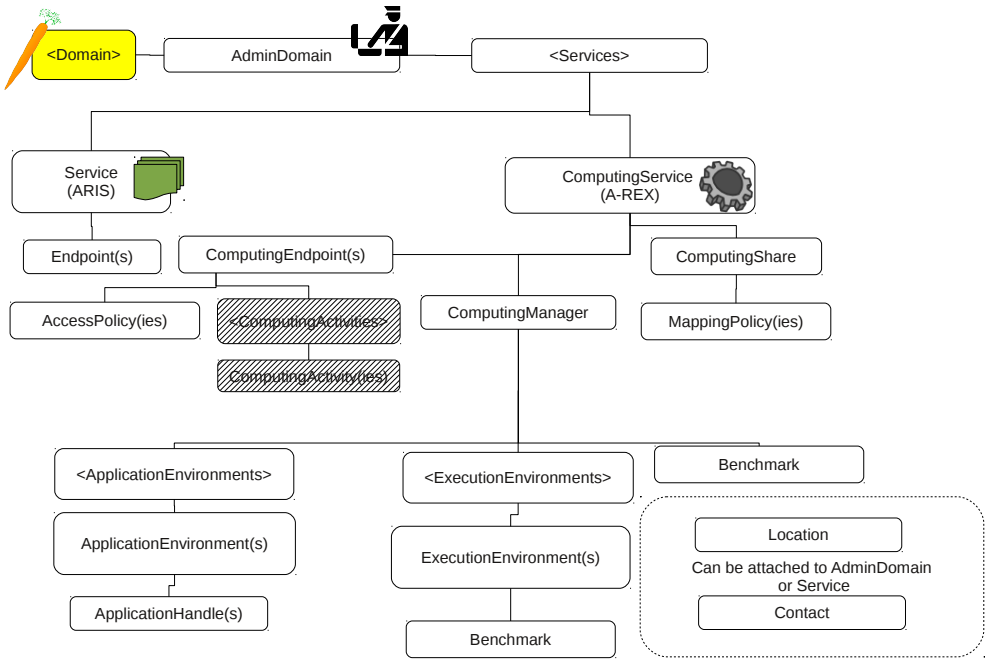

Figure 8: The ARC GLUE2 XML tree. The proposed structure has no distinction between administrative and resource objects, plus a <*Domain*> object serves as the *o=glue* root in LDAP. A <*Services*> object is introduced to behave like *GroupID=resource* in LDAP. Note that *ComputingActivities* are greyed out because they are not published in the XML document. Instead, such information is calculated and stored elsewhere to be served on demand, by means of the Web Service based protocols that *arched* supports. Note also that the tree structure is subject to change depending on future OGF GLUE2 Working Group agreements.

6. Integration with BDII-top

The purpose of integrating with BDII-top demands a clear understanding and a common agreement of how an LDAP tree looks like. The draft rendering document [14] that has been proposed and used as a guide for GLUE2 does not fix this structure and leaves the developers free to choose it.

However, the information gathered from BDII-top by querying ARIS via LDAP is merged using slightly similar *bdii-update* scripts that ARC uses. For this reason, an agreement is needed on how to merge LDAP trees. This topic is currently under discussion.

Figure 9 shows a BDII-top tree and a proposal on how to merge an ARC local information tree with the BDII-top index information tree.

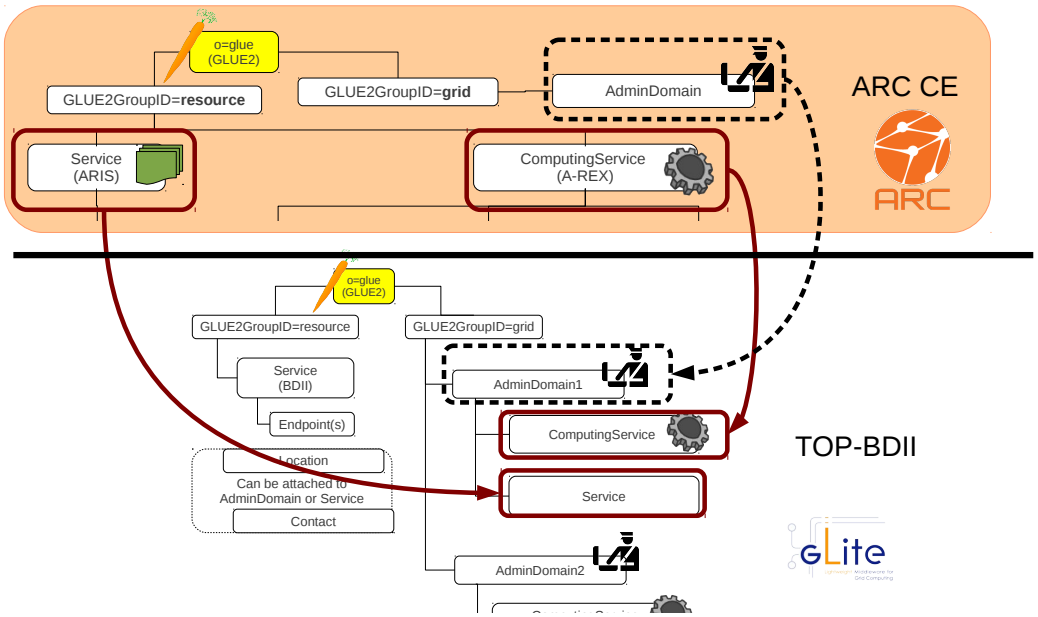

Figure 9: ARIS and BDII-top. The proposed merge of the ARC GLUE2 LDAP and BDII-top LDAP trees. The BDII-top preserves all *AdminDomains* under the *GLUE2GroupID=grid* branch. Thus, ARC *GLUE2GroupID=grid* branch should be merged under the same branch in the BDII-top (dashed) and all the objects under ARC *GLUE2GroupID=resource* tree should be connected with the corresponding *AdminDomain* branch in the BDII-top. Note that the proposal is subject to change depending on future OGF GLUE2 Working Group agreements.

7. Notes on early GLUE2 adoption

The ARC developers team was one of the first to come with an XML and later LDAP implementation of GLUE2 as part of the ARC information system. The team is also participating in the ongoing discussions about the still not finalized renderings.

In this section we would like to highlight some of the challenges encountered while implementing and structuring the information with respect to the GLUE2 model and its recommended realizations.

At the time of writing, no official rendering recommendation from OGF is out.

At the time of development, a reference realization draft for XML [17] and a draft XSD schema [18] were produced by the GLUE2 OGF group so that also the XML rendering could be generated and validated . In the last 5 years, experimental ARC Web Services ran using the GLUE2 information generated by the XML infoprovider back-end, but unfortunately there were no real consumers of that information.

A reference realization draft for LDAP existed [14], but did not contain a detailed description on the DIT structure, leaving the implementers free to shape subtrees the way they want. The ARC team believes a well structured DIT to be a key issue in LDAP rendering. This approach is based on previous experience in the publication of dynamic information with the NorduGrid schema that showed that structured information makes it easier to craft queries, and it is also easy to read for human beings. Such structured information is used by ARC clients (job submission and monitoring) to perform efficient queries. Hence ARC team created its own GLUE2 DIT, following the constraints detailed in the realization draft [14].

In order to simplify the generation and interpretation of the GLUE2 data, we decided to have an LDAP DIT structure as close as possible to the one suggested by the GLUE2 XSD schema, and introduced groupings in the DIT similar to XML grouping elements.

An example where this is useful is when a cluster holds thousands of jobs, and an information consumer wants to know only those jobs that have been submitted via a specific *ComputingEndpoint*.

By leveraging the ARC DIT, one can limit the size of the query by specifying the following base (`-b` option in the *ldapsearch* command):

```
ldapsearch -x -h myhost -p 2135
       -b 'GLUE2GroupID=ComputingActivities,GLUE2EndpointID=myEndpoint,
       GLUE2ComputingServiceID=myService,GLUE2GroupID=resource,o=glue'
```

which is shown to be 3 times faster than any filtered search. Moreover, the grouping makes it easier to get the base *dn*: for *ComputingActivities* on each *ComputingEndpoint*. However, performance figures are out of the scope of this paper, so the topic will not be discussed further.

The EMI2 release pushed the OGF group to review the LDAP and XML schemas, and all the EMI partners to update their renderings to synchronise to the latest version.

However, the main challenges were not in the implementation of schema-compliant LDIF and XML documents, but in the generation and validation of attribute values.

The first challenge was to find a good recipe for universally unique and persistent IDs. Universal uniqueness was easy to achieve, although not completely trivial. Persistency, that is, the time an ID should exist unchanged, is yet to be understood, and it depends mostly on operational needs.

The second issue was with open enumerations. The listed open enumerations contained in the GLUE2 model [13] are clearly insufficient to perform useful discovery. For example, the `Capability_t` *information.discovery* does not tell if a Service is an index or a local information system. Therefore, new ones had to be created and an effort is ongoing to have a consistent shared database of these strings, especially for enumerations like `Capability_t` that are crucial for understanding what a service can do. Novel services like EMI-ES triggered new `ServiceType_t` and `InterfaceName_t` values that should be available for all the information consumers to use. In this case, little recommendation is given in the GLUE2 documents on how these names should be constructed. An implicit suggestion is to use some namespace schema, but nothing has been declared as mandatory. Therefore, the current scenario is that many products have free strings as names, with no domain information attached. This is also due to the fact that some products are orphaned by their initial developers.

The third issue is about the concept of *UserDomain* that is still not well understood in terms of implementation, not only in ARC but as a general issue: how does it interact with the current authentication and authorisation models, how are these records filled, how can clients and servers benefit from it. Therefore, its implementation has been put on hold for the moment.

A fourth issue arises at the infrastructure level. As said, GLUE2 benefits will come from the two listed systems: EMIR and BDII-top. However, both systems have different ways of aggregating records from a local source, and it is yet unclear which GLUE2 data is needed to successfully achieve service discovery both on the local and index levels. For this reason, work needs to be done to identify what are the GLUE2 bits of information that are crucial to create a real shared multi-purpose information system. The ARC team identified the following minimal GLUE2 attributes to be important for service discovery: *Service ID*, *Service Type*, *Service Capability*, *Endpoint ID*, E*ndpoint InterfaceName*, *Endpoint Capability* and *Endpoint URL*, the latter being the effective entry point to any service. The reasons behind these choices are out of the scope of this paper, so no further detail will be given.

8. Summary

In this paper we presented the state of the art of ARC Information System, especially regarding the coming GLUE2 compliance.

Four major highlights of our system are as follows:

- ARC enhanced information system has a layered structure with clean separation of functionalities:
  - Local/resource level (ARIS);
  - Index level (EGIIS and third-party services: BDII-top and EMIR);
- ARIS is a powerful local information service:
  - Speaks multiple dialects (NorduGrid, Glue1, GLUE2);
  - Creates LDIF and XML renderings – ARC is the only middleware that renders both;
  - Talks many protocols using LDAP and Web Services: NorduGrid LDAP, LDAP, WSRF, XBES, EMI-ES and REST;
- Information consumers:
  - Are easy to write as they would be based on well known standard technologies;
  - Can get very accurate and fresh information;
- Is aiming at a full GLUE2 integration with BDII-top and EMIR

9. Acknowledgements

The authors would like to thank Laurence Field for data and plots about the status of current existing information systems and Christian Ulrik Søttrup for his help reviewing the paper.

This work is partially funded by the European Commission as part of the EMI Project under the Grant Agreement number INFSO-RI-261611.

Last revision of this document: 20th July, 2012.

References

[1] **European Middleware Initiative project**, http://www.eu-emi.eu

[2] **EMI Registry Requirements Specification and Architecture V.0.2**, *Memon, A.S. (FZJ) ; Field, L. (CERN) ; Marton, I. (NIIF) ; Szigeti, G. (NIIF)*, http://cdsweb.cern.ch/record/1359909/

[3] **Advanced Resource Connector middleware for lightweight computational Grids**, *M. Ellert, M. Grønager, A. Konstantinov, B. Kónya, J. Lindemann, I. Livenson, J.L. Nielsen, M. Niinimäki, O. Smirnova, A. Wäänänen*, http://www.sciencedirect.com/science/article/pii/S0167739X06001178

[4] **The NorduGrid/ARC Information System**, *B. Kónya*, The NorduGrid Collaboration, NORDUGRID-TECH-4, http://www.nordugrid.org/documents/arc_infosys.pdf

[5] **OpenLDAP, an Open Source implementation of the Lightweight Directory Access Protocol**, http://www.openLDAP.org/

[6] **The Hosting Environment of the Advanced Resource Connector middleware**, *A. Konstantinov and J. Jönemo*, The NorduGrid Collaboration, NORDUGRID-TECH-19, http://www.nordugrid.org/documents/ARCHED_article.pdf

[7] **ARC Information System developer's handbook**, not yet released.

[8] **Berkeley Database Information Index V5**, https://twiki.cern.ch/twiki/bin/view/EGEE/BDII

[9] **RFC2849 - The LDAP Data Interchange Format (LDIF) - Technical Specification** *IETF, G. Good* http://www.ietf.org/rfc/rfc2849.txt

[10] **Lightweight Directory Access Protocol (LDAP): The Protocol** *IETF, J. Sermersheim, Ed.* http://tools.ietf.org/html/rfc4511

[11] **ARC Computing Element System Administrator Guide**, *F. Paganelli, Zs. Nagy, O. Smirnova and others*, The NorduGrid Collaboration, NORDUGRID-MANUAL-20, http://www.nordugrid.org/documents/arc-ce-sysadm-guide.pdf

[12] **GLUE Schema Version 1.2, GLUE Schema Version 1.3**, http://glueschema.forge.cnaf.infn.it/Spec/V12 http://glueschema.forge.cnaf.infn.it/Spec/V13

[13] **GLUE Specification v2.0**, *S. Andreozzi and others*, GFD-R-P.147, http:/www.ogf.org/documents/GFD.147.pdf

[14] **DRAFT: GLUE v.2.0 - Reference Implementation of an LDAP schema**, *Sergio Andreozzi and others*, http://forge.ogf.org/sf/go/doc15518?nav=1

[15] **gLite - Lightweight Middleware for Grid Computing**, http://glite.cern.ch/

[16] **UNICORE (Uniform Interface to Computing Resources)**, http://www.unicore.eu/

[17] **DRAFT: GLUE 2.0 - Reference Realizations to Concrete Data Models**, *Sergio Andreozzi and others*, https://forge.ogf.org/sf/docman/do/downloadDocument/projects.glue-wg/docman.root.drafts.archive/doc15221

[18] **DRAFT: OGF GLUE 2.0 - XML Schema mapping**, *Sergio Andreozzi and others*, https://github.com/OGF-GLUE/XSD/blob/master/schema/GLUE2.xsd