PROCEEDINGS OF SCIENCE

EMIR: An EMI Service Registry for Federated Grid Infrastructures

Shiraz Memon[*†]
Jülich Supercomputing Center
E-mail: a.memon@fz-juelich.de

Morris Riedel
Jülich Supercomputing Center
E-mail: m.riedel@fz-juelich.de

Laurence Field
CERN
E-mail: laurence.field@cern.ch

Gabor Szigeti
NIIF
E-mail: szigeti@niif.hu

Ivan Marton
NIIF
E-mail: martoni@niif.hu

The European Middleware Initiative(EMI) is a European project that represents a collaboration of four middlewares namely ARC, dCache, gLite, and UNICORE.All these middleware services should be easily deployable in a Grid Infrastructure. However the immediate challenge is the discovery of those services in a particular infrastructure that is typically done via so-called registries. This is a major requirement for operational systems, and the middleware itself. Existing registries such as ARC Information Index or UNICORE registry are designed to index middleware specific services. Given the centralized nature, the scope of these registries can become limited when considering a federated infrastructure that relies on service of different technology providers. Distributed Grid infrastructures such as EGI are be federated, therefore a unified registry should reflect this requirement. In this paper, a common registry EMIR is proposed, which attempts to overcome the challenges of federation, robustness, and performance implications of ever expanding Grids.

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

[*]Speaker.
[†]Author

PoS(EGICF12-EMITC2)073

1. Introduction

EMI[1] aims at providing a unified software bundle to Grid Infrastructures. The bundle contains all of the software and services which a Grid would need to exploit the infrastructure resources. Despite of being unified, the services publish and register their information to the platform or middleware specific service registries; such as ARC job submission service is typically registered to the ARC information system before being discovered by an ARC submission client, likewise for the UNICORE and gLite. Having multiplicity of such registries implies proprietary protocols and interfaces which may force the operators to setup all the registries before allowing clients/users to use their infrastructure or reduce the scope of offering to limited number of middleware services. As a consequence, 'information islands' are created that hinder the information exchange across middleware and often infrastructure boundaries. Considering the problems, EMI developed a unified, but federated service registry "EMIR", which offers a common interface to publish and discover all the services within the scope of EMI. The interface of EMIR is a Web service based on Representational State Transfer (REST)[2] using Java Script Object Notation (JSON)[3] for the message exchange, whereby HTTP URIs and methods are exposed to interact with the service indexes. In addition to the common interface, it offers federation and global tier P2P based replication for robustness and scalability to match the current trends of Grid infrastructures that expand in service scope and size. The information model of the registry adopts a subset of the GLUE 2.0[4] entities: a Abstract Service and Endpoint Model. For which a new JSON rendering has been derived to create service endpoint descriptions. From a security perspective, a fine grained and standardised authentication and access control mechanisms such as X.509 and XACML[5] have been provided to cope with the emerging security requirements and compatibility with the EMI driven infrastructures.

The EMIR can be seen as instrumental in unifying the service discovery of all the EMI services in modern large scale computing infrastructures (e.g. EGI[6], XSEDE[7], NorduGrid[8]). Having REST interface enables the services (and their providers) to intuitively advertise the reference/endpoint information. The adoption of the OGF's GLUE 2.0 standard facilitates the process of publishing middleware specific services to the EMIR. A generic but highly configurable EMIR client has been implemented for middleware agnostic service publishing, thus offering a way for any remote service to be indexed and discoverable. Enabling growing federation of services in current DCIs is a trend today, therefore right from EMIR's beginning federation was one of the intrinsic parts of the design and therefore strongly supported. Furthermore, the global tier replication at the global registries enables the service discovery on global level but in decentralized and robust manner. The feature set offered by EMIR benefits not only the operations personnel but increases efficiency of the overall functioning of a Grid Infrastructure.

This paper focuses on related service registries in section 2, which have been providing service discovery in homogeneous environment where middleware specific services can be cataloged. In section 3, we introduce GLUE 2.0 entities in the EMIR information model and formalizing the JSON for message exchange and service endpoint description. The federated architecture in the section 4 describes high level architecture and its core components. Finally in the section 5, we conclude with the summary and future developments.

2. Related Work

While designing the EMIR service registry, number of requirements were gathered, notably: the decentralization through inherent distribution of registries, support for federations, scalability, adoption of standardized GLUE 2.0 information model, and (most importantly) the integration with (all of) the EMI services. While investigating during the design phase number of service registries have been evaluated to identify whether it fulfills the above requirements. The short description of the comparison analysis is given under the following sections.

UNICORE 6 Global Registry[9] is an implementation of the WS-Service Group specification[10] and uses WS-Addressing[11] endpoints to define UNICORE Services. It is a centralized shared registry which requires all the UNICORE sites (hosting the services) to publish their endpoint information. Each record manifesting the information has corresponding time-to-live (TTL) attribute, thus obligates the registrant to update the endpoint information before it gets expired. Therefore the basic functionalities such as: store, maintain, and providing the services' information is equivalent to the EMIR. Given the centralized nature the registry is highly susceptible to bottle neck in a Grid infrastructure. Due to its close coupling with the UNICORE, discovery of the non-UNICORE based services is not seamless.

The Information System Indexing Service (ISIS)[12] is an ARC's information system, like EMIR's global registry, based on a P2P model. The replication model is eventual consistent, in which all the records in a peer (an ISIS server instance) are expected to propagate to its neighbors in a sufficient period of time, this eventually make all the peers consistent (or identical). The core interface of the ISIS is SOAP based Web services, while supporting XPath 1.0 as a query language to retrieve the services' information. Despite of being Web services interface, the clients are highly customised to support the ARC specific services, hence eliminating the discoverability of other EMI based services.

Grid Operations Centre Database[13] is a central service and a site registry which give access to the Grid information, specifically about the Regions, Countries, Resources, and Users. It uses the Psuedo Object Database Model[14] - the relational meta-model - to store and maintain the given information. Along with the Web front-end, the application or client developers can also use read-only RESTful API to view the indexed information. There is a significant difference in architecture if we strive to compare GOCDB with EMIR, most notably the client/server architecture of GOCDB which requires central administration of the server component, whereas in EMIR autonomous distributed and replicated registry nodes can be commissioned. The distributed nature of registries in EMIR would bring robustness and scalability to the Grid infrastructures.

3. Adoption of GLUE 2.0 Information Model

3.1 Overview

One of the fundamental requirements of EMIR is to have flexible, standardized, interoperable, and expressive model to represent service endpoints in a Grid infrastructure. The flexibility here manifests not only expressing the existing, but evolving services which can be commissioned during lifetime of the infrastructure. In an attempt to make EMIR services standardised and interoperable, OGF's GLUE 2.0[4] entities have been adopted. Since it is a widely accepted standard

in Grid community and its adoption in a number of production Grid infrastructures (e.g. XSEDE, NorduGrid, and EGI) would have great impact while integration EMIR with Grid services and clients. However, reducing the effort of complying with a new (or proprietary) specification for every service registry.

3.2 Main Entities and Realization

It is important to note that only a subset of the GLUE 2.0 entities have been endorsed to advertise the Grid service endpoints. Therefore each service endpoint record in the EMIR embodies attributes from the abstract service classes: *Service, Endpoint, Location*, and *Contact*. Due to schema- free (or NOSQL) nature of database back-end, extra (non-GLUE 2.0) attributes can also be indexed to enrich the description of corresponding service endpoints.

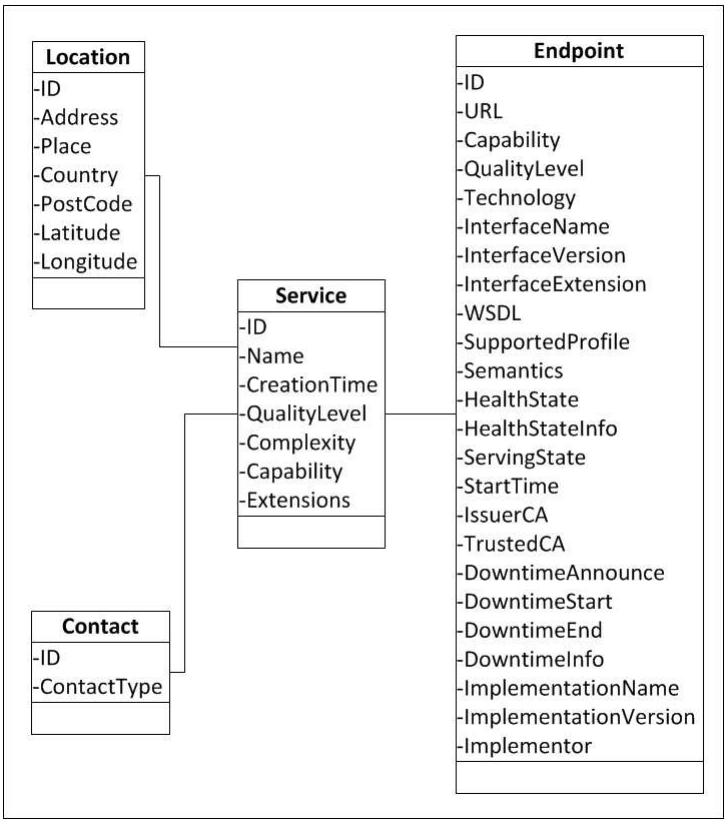

Figure 1: GLUE 2.0 Service Entities for the EMIR

As mentioned earlier and shown in the figure1, the key entity in the information model is an *Endpoint*, it abstracts hardware or software entity's access point, showing what capabilities have been offered from a given service's endpoint. It can be utilize by a discovery clients (monitoring systems, batch job submitters, or other middlewares). The description also contains the abstract *Service*, *Location*, and *Contact* to specify the service type , geographical location, and necessary contact information associated with an endpoint.

EMIR information model and message exchange relies upon JSON[3], which vouches for rich yet simplified means to define service endpoint. Therefore each service endpoint record is a JSON

document which contains all the attributes in a flat manner. Moreover, the projection of XML documents compliant to the GLUE 2.0 specification has also been supported and can be retrieved while querying with appropriate HTTP content type.

4. Federated Architecture

One of the underpinning requirements of the Grids was to support the federated service discovery. The federation here implies an aggregated view of deployed services in a Grid infrastructure. EMIR supports such a federation by aggregating geographically disparate EMIR servers in a unified, manageable, and in robust manner. The EMIR architecture is bipartite, hierarchical and P2P. In hierarchical, each EMIR node/server also called Domain Service Registry (DSR) advertises the contained service endpoints to its pre-configured parent EMIR node/server. In this way all the aggregated services get propagated to the top level EMIR node - the Global Service Registry. Such organization of the EMIR nodes is succinctly depicted in figure 2. The P2P network is formed between top level nodes. The records stored at each get forwarded to every other top level node in the network.

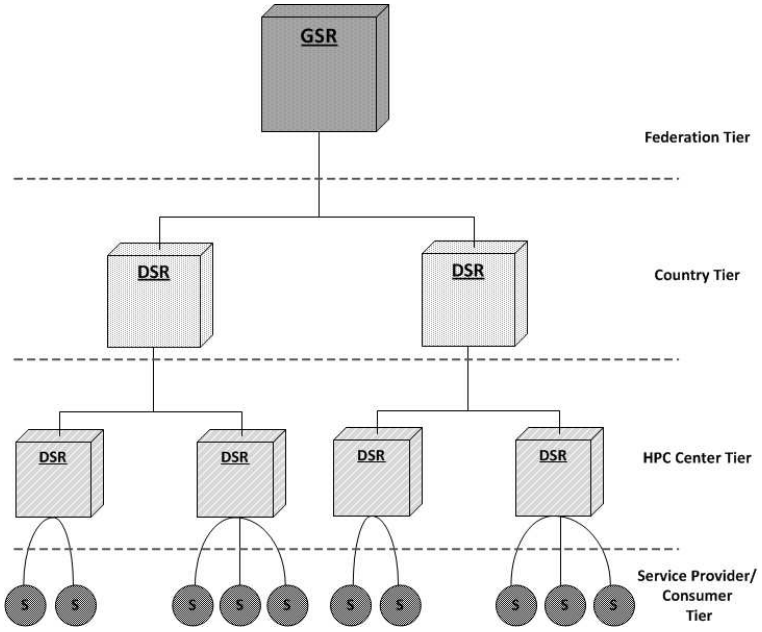

Figure 2: EMIR Network in a Federation

4.1 Domain Service Registry (DSR)

It is a core server component in a hierarchical/tree network of EMIR, offers a number of functionalities to perform service discovery in a federation. The remote interface to the DSR is a REST API - a remote Web services interface to manage registrations and execute sophisticated and URI based queries. As shown in 2, the DSR at the leaf level has a single parent to which it synchronises all its content. Akin to Pub-Sub messaging model, the synchronisation takes place between parent and child DSR is push based. Therefore enabling a child DSR to subsequently propagate the

updates to its immediate parent, similarly for all the higher level DSRs until the original endpoint record reaches the hierarchy's root GSR. In case of failures, such as temporary unavailability of any of higher level DSR in the hierarchy, the immediate child DSR recognizes it and creates an in-memory soft-state database, which contains track changes to the service endpoint record collections. If at some point of time the failed DSR re-joins, the content of the already created soft-state database will get transferred to make the newly rejoined DSR consistent. Additionally, for the guaranteed consistency, explicit propagation of all the content from a child DSR to its parent is scheduled to once a day.

4.2 Global Service Registry (GSR)

A GSR is a DSR server behave as a top/root depending on the configuration, aggregates all the records contained within underlying DSRs in a hierarchy or federated infrastructure. The service endpoint records stored in a GSR database are replicated among other GSRs using the structured Peer-to-Peer (P2P)[15] and eventual consistent replication model[16]. The notion of given model is to bring all the GSRs in a consistent state by certain period of time. Minor delays in the consistency can be ignored due to the trade-off between availability and consistency, moreover the nature of content in a EMIR database is static and less critical. Since the service endpoint records are replicated among all the GSRs, the records stored in off-line GSRs' database can be discovered from the available GSRs, to enable robustness and fault tolerance.

4.2.1 P2P Connection Bootstrapping

Another challenging aspect pertinent to the P2P network was bootstrapping and connection of GSR peer nodes. For that we have introduced a globally locatable list, containing address of all the GSRs which intend to join the network. Since each GSR discovers every other GSR peer address (from the global list), the service endpoint records residing at a GSR gets propagated to every other GSR peer, in a fixed size chunks. The use of chunks has significantly decreased the underlying network congestion and eased the handling of failures while synchronisation.

4.2.2 Routing in the Network

The GSRs in EMIR's P2P network are synchronised while sending the notifications of *Create*, *Update*, and *Delete* operations to its peer. This yields redundancy and fault tolerant message routing between any two GSR nodes.

The most important configuration parameter of a GSR is the *sparsity*. It is a non-negative integer which determines the number of neighbors as a function of the actual number of member GSR peer nodes in the network. Whereas the number of neighbors can change due to churn in P2P network. Providing a greater value will shape the graph sparse (see figure 3) and provided when the number of expected GSRs to join are not many.

However, to achieve maximum robustness and high level of consistency of GSRs in the P2P network, the smaller sparsity value (as small as 1) should be selected, this would result in more dense graph, as shown in (see figure 4).

Every peer (or node) in the network has always exactly s-based logarithmic **N** neighbors where **N** is the number of GSR nodes registered into the network, **s** is the level of sparsity. There can be

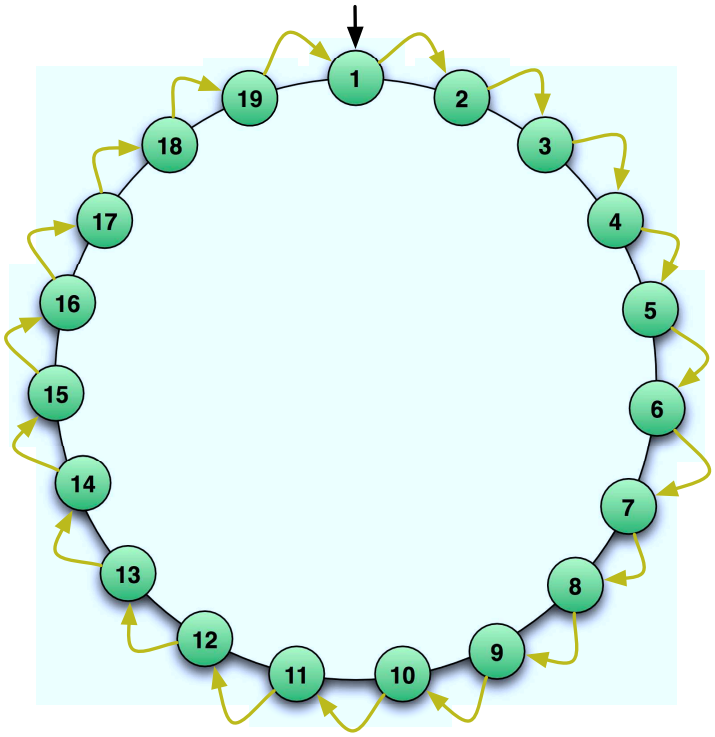

Figure 3: Sparse graph due to higher sparsity value

exactly *n-way* redundancy achieved in the system so at most *n - 1* node can disappear without any serious communication impair.

Although the methodology provides a secure and quite fast solution for the message delivery, but has a high communication cost as there are **n*N** messages required (in general) for every *Create*, *Update* or *Delete* message on the top level GSR network. If the greater sparsity is configured for the GSR, then the number of expected peers decreases, in the extreme case to only one peer neighbor. In this case there are only **N** (the theoretically minimum) messages traversing in the network at the expense of a slower and more non-robust data propagation.

As a data-driven routing is being used in the peer-to-peer network, the peers examine the received Service Endpoint Record, store, and then forward only those messages that are newer than the already stored version belonging to the same service identifier. If the information in the message is out-of-date it will be simply dropped. By using this method it is not necessary to store the formerly seen nodes in the messages, but the routing decision Ů that is who to send the message to Ů is based on local information. Since the routing is based on timestamps, it is very important to keep the peers' clock in synchronous with each other or with an outer reference, such as Network Time Protocol (NTP). Another advantage of the adopted P2P approach is the capability of handling the case of swapped Register/Update and Delete messages. It is possible for a Delete message to fore run a previously generated Register or Update message. This causes just a temporary imprecise entry in the database however the fault can be quickly fixed while not propagating it to further peers. The states such as "Valid", "Expired", or "Removed" are maintained and assigned to each endpoint record, whereby the actions are performed on records according to the state assigned by

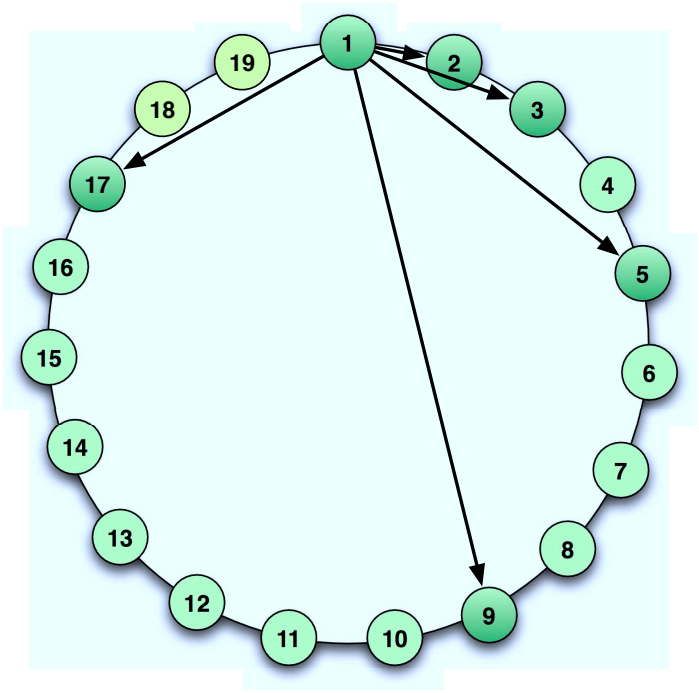

Figure 4: Dense graph due to lower sparsity value

GSR. The only exception is the delete message which sets records' state to "Removed" without performing real delete operation, hence the records are kept for some arbitrary period of time to avoid accidental deletion.

5. Conclusion and Future Work

While considering the amount of multifarious services offered by EMI, we can conclude that a common and unified service registry is indispensable. However EMI's EMIR enabling the federations and the flexible information model, is vital for the scientific communities using the Grid infrastructures. The benefits of being federating is to communicate the service provider's offered services to the potential consumers (discovery clients). While the flexible information model enhances the expressiveness and interoperability of already published and the forthcoming Grid services.

The future implementation work is focused on: (1) performing distributed deployment to measure the message exchange latency while synchronization and replication of the services among EMIR nodes, (2) robust handling of failures while synchronizing and replication of the distributed EMIR nodes, (3) automated (de)commissioning of services and nodes, (5) using EMI's common authentication library for standardised authentication, and finally (4) integrating with all the EMI offered services.

6. Acknowledgments

This work is a part of JRA1 activity within the EMI Project, supported by European Commis-

sion under the grant number EMI INFSO-RI-261611. We also thank Balazs Konya, and Lorenzo Dini for their strong support and guidance in realizing the EMIR.

References

[1] EMI: European Middle Initiative, http://www.eu-emi.eu

[2] Roy Fielding, Richard Taylor, *Principled design of the modern Web architecture, ACM Trans. Internet Techn.* **02**, [ISSN 1533-5399]

[3] JSON, http://www.json.org

[4] Sergio Andreozzi, Stephen Burke, Felix Ehm, Laurence Field, Gerson Galang, Balazs Konya, Maarten Litmaath, Paul Millar, JP Navarro, *GLUE Specification 2.0, OGF* **09**, [GFD-R-P.147]

[5] Tim Moses, *eXtensible Access Control Markup Language Version 2.0, OASIS* **05**, http://docs.oasis-open.org/xacml/2.0

[6] EGI: European Grid Initiative, http://www.egi.eu

[7] XSEDE, http://www.xsede.org

[8] NorduGrid, http://www.nordugrid.org

[9] UNICORE: Distributed Computing and Data Resources, http://www.unicore.eu, accessed on 05.04.2012

[10] Tom Maguire, David Snelling, Tim Banks, *Web Services Service Group Specification v1.2, OASIS* **06**, [wsrf-ws-service-group-1.2-spec-os]

[11] Don Box et al, *Web Services Addressing, W3C* **05**, http://www.w3.org/Submission/2004/SUBM-ws-addressing-20040810/

[12] *ARC Peer-to-Peer Information System*, **12 [**NORDUGRID-TECH-21**]**

[13] Gilles Mathieu, Dr. Andrew Richards, Dr. John Gordon, Cristina Del, Cano Novales, Peter Colclough, Matthew Viljoen, *GOCDB, a topology repository for a worldwide grid infrastructure, CHEP* 09 [062021**]**

[14] Gilles Mathieu, Peter Colclough, *A Pseudo Object Database Model and Its Applications on a Highly Complex Distributed Architecture, DBKDA* 09

[15] Ralf Steinmetz, Klaus Wehrle, *Peer-to-Peer Systems and Applications, LNCS* 05, [ISBN 3-540-29192-X, Volume 3485**]**

[16] Werner Vogels, *Eventually Consistent, ACM Queue* 08 [Volume 6, No. 6**]**