GRID-DL – Semantic GRID Information Service

Olexandr Pospishniy[1], Sergii Stirenko[1]

[1] National Technical University of Ukraine "Kyiv Polytechnic Institute", Kiev, Ukraine
pospishniy@kpi.in.ua, stirenko@ugrid.org

Abstract. The effectiveness of modern complex Grid systems strongly depends on the availability, accuracy and relevance of information on all connected resources, their characteristics and state. An access to this information plays a very important role in any Grid system, providing necessary information for other Grid components and users. We set a goal for "intellectualization" of key Grid systems to promote it to a larger audience of users that sometimes have difficulties adjusting to way Grid is operated. We believe that application of semantic technologies opens up many new possibilities and prospects for further improvement of the basic elements of Grid systems, promoting the emergence of new models of user interaction with them. In this work we present Grid-DL - a prototype semantic Grid information service that relies on ontologies in order to build up a knowledge base of Grid resources and process user queries to it. We share our experience designing an idea of "pluggable" ontologies and sufficient core system taxonomy, while facing severe performance challenges implementing our system.

Keywords: Grid, information service, semantics, ontology, OWL

1 Introduction

Grid computing proved to be effective and powerful instrument for modern data-intensive science and engineering. The idea was simple, yet very powerful – to integrate geographically dispersed computing resources from multiple administrative domains and provide shared access to them. A set of software libraries, called Grid middleware, was developed to provide an extendable platform for creating virtual organizations that would pool and share their resources in order to achieve some common goals.

One of the distinct characteristics of grid system is resource heterogeneity. Every Grid site is unique with respect to their hardware and software composition. Also each resource, apart from being shared within a Grid environment, could be used and managed by its immediate owner. Thus effective management and use of such complex heterogeneous systems as Grids is entirely dependent on the availability, accuracy and relevance of information on all available resources, their characteristics, condition and usage policy. An access to this information should be as clear as possible for a wide range of users and at the same time sufficiently flexible and adaptive for a wide range of tasks.

Traditional Grid information services tend to force users to comply with its semantics. Users describe requirements of software they want to run in terms of allowed attributes. This quite often becomes a source of erroneous assignments of tasks to Grid resources, reducing overall system throughput

In order to address this issue we hypothesize that semantic technologies, developing under the vision of the Semantic Web, can be effectively applied to Grid systems.

2 Grid resource ontology

Grid resource ontology is a keystone in our vision of semantic grid information services. It gives us a foundation to build upon, as we introduce more complex and specific ontologies on top of it.

The ontology we developed[1] is based on a specially designed scheme for referencing Grid entities - Grid Laboratory Uniform Environment, GLUE [1]. This scheme describes most of the Grid components and their characteristics, and is used in modern information services such as MDS [2] and BDII [3].

Terminological component of our ontology contains 65 classes, 33 object properties and 106 data properties. Ontology corresponds to the *SHIF(D)* expressiveness of description logic, which relates it to OWL-Lite dialect.

There are 3 classes on the upper level of hierarchy: GridEntity, DomainConcept and Enumeration. First class serves as the superclass of all core Grid entities, the second class defines the supporting domain concepts, and the latter is used for the enumerated concepts.

Ontology defines the following basic elements of the Grid system (Fig. 1):

i. CoreEntity: Service and Site
ii. ComputingResource: Cluster, SubCluster and ComputingElement
iii. StorageResource: StorageElement and StorageArea

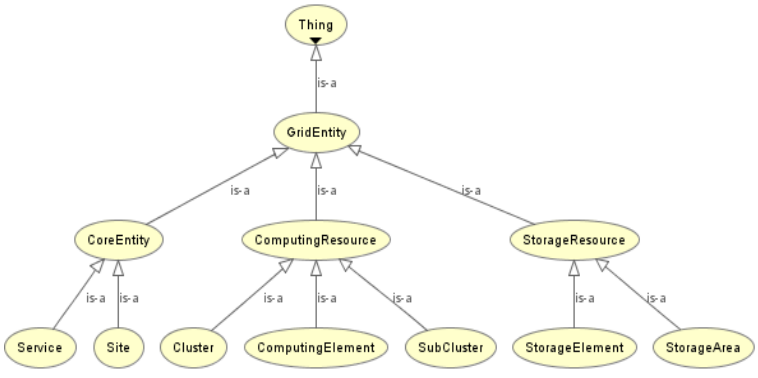

Fig. 1. Hierarchy of base Grid concepts

[1] http://grid-ontology.googlecode.com/files/GLUE.owl

Fig. 2 shows the class hierarchy of *DomainConcept* and *Enumeration* classes, which are used to describe the basic elements the Grid system.

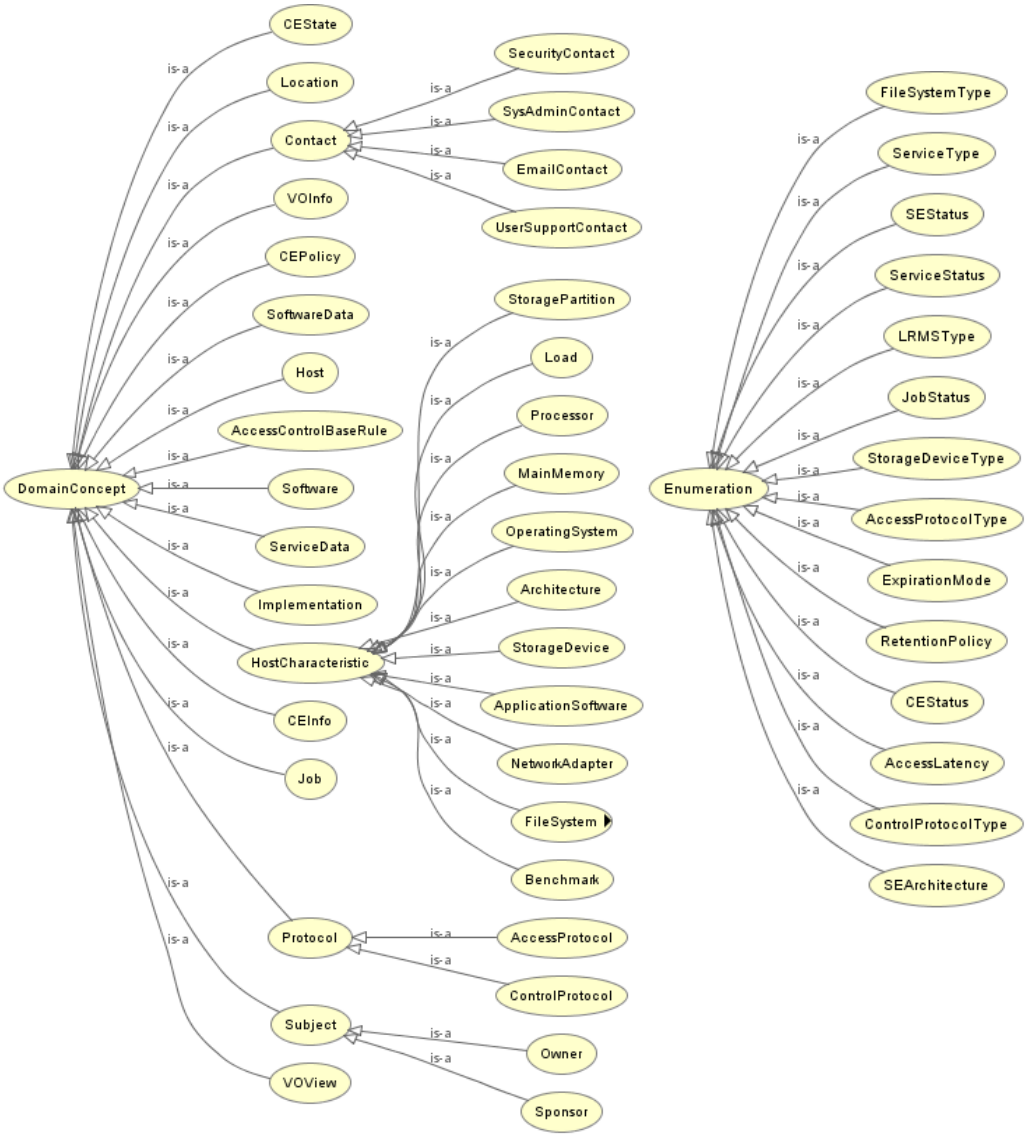

Fig. 2. Hierarchy of the supporting domain and enumeration concepts

Ontology was developed using the ontology editor and knowledge-building tool Protégé [4]. Protégé editor does not perform any knowledge processing, i.e. does not

contain a reasoner. For these purposes, an external third-party OWL reasoners must be connected through the OWLAPI interface [5].

However, to be of any use to us, ontology needs to be filled with a set of assertions about individuals that represent physical Grid resources (ABox).

For the purpose of generating an ABox we have developed a program[2] to import data from the LHC Computing Grid (WLCG), the most ambitious Grid system to date, which serves to carry out the experiments on the Large Hadron Collider.

Application is not only limited to the LHC Grid and can be used to import data from any other Grid system that has BDII- or MDS-based information service.

3 Semantic information service architecture

To test out and refine our ideas we have built a prototype of Semantic Information Service that we call *Grid-DL*. Grid-DL is an autonomous Web-application that contains a set of Web-services and a simple Web-interface. Project is implemented using Java 7 platform and requires Apache Ant tool to be compiled and packaged. The resulting web-application in the form of war-file is ready for deployment in any J2EE-compatible application server, such as Apache Tomcat. Figure 3 outlines overall Grid-DL system architecture with all major components.

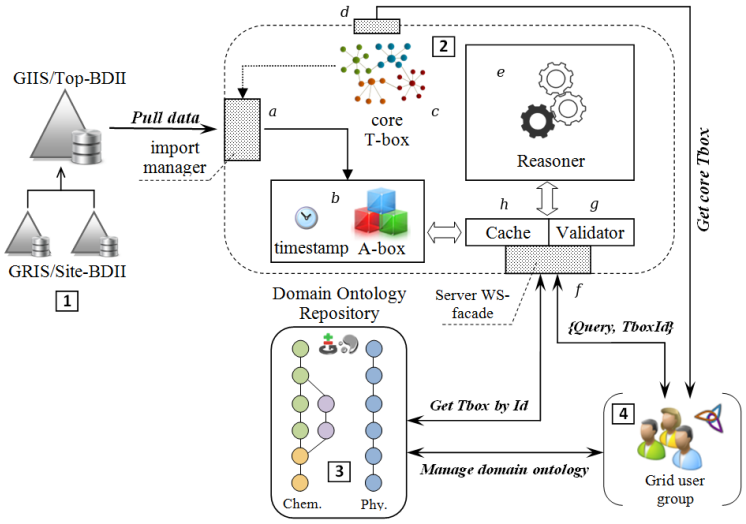

Fig. 3. Grid-DL system architecture

3.1 Backbone Grid information service

At this stage of approbation of our ideas we decided not to concern ourselves with the developing of some new resource monitoring framework, but rather adapt to the

[2] Source code available http://code.google.com/p/grid-ontology/source/checkout

traditional Grid information systems, widely uses in production. Thus a special module in Grid-DL, called import manager retrieves all required information about online Grid resources from the top-level information server. We consider BDII, GIIS and EGIIS from gLite, Globus and ARC middleware respectively, as such top-level information providers.

For universality we developed a mechanism of adapters to connect Grid-DL to arbitrary compatible Grid information service. Thanks to ontologies, all data obtained from external information source will be given a generalized invariant representation.

3.2 Semantic Information Service

Based on the philosophy of the Grid systems, it is useful to distinguish between two separate operational levels: a common Grid-wide space and an isolated virtual organization where users do their tasks. We exploit this division by using two ontologies when working with information service: core system ontology and user ontology.

Core ontology described in the previous section (TBox) is relatively broad, overarching and static in its nature. Its purpose is to create a solid foundation for storing all available data on resources acquired from a Grid information service and provide material for user ontologies to be built upon.

Virtual organizations, on the other hand, are usually formed for solving some specific tasks within some domain, usually bringing together researchers from same or relative fields of science. That is why we think that it is plausible to extend core system ontology with additional domain-specific knowledge that will capture the specificity of these virtual organizations. We hope that multiple users that work in the same field of study will collaborate and come up with an extension to core ontology that will contain new constructs that would be helpful for them. Some possible extensions could contain descriptions of various algorithms and methods, tools, terminologies and any arbitrary assertions common to researches within this virtual organization.

Domain ontologies will be created and managed by the virtual organizations themselves thus such ontologies will be relatively specific and dynamic.

In Grid-DL (Fig.3.) information about all Grid resources is coming through an import module (a) and based on the terminology presented in the core T-box (c) forms a time stamped assertion box (b) that contains all the information on Grid resources. To retrieve the core T-box a user must use provided web-service (d). TBox ontology is read-only and identified by version number. ABox is stored with a time stamp in order to manage relevance of the retrieved results.

For reasoning Grid-DL could use any OWL-reasoner (e) that supports OWLAPI interface. Our test environment uses Pellet [6] for this purpose.

All interaction with semantic information service is carried out through a web-service façade (f). We implemented this component using JAX-WS library from J2EE platform in order to provide interoperability with any client that supports a standardized web-service technology stack.

All requests coming to Grid-DL are validated (g) in order to find semantic errors and logical inconsistencies in search queries. Additionally we cache (h) user requests to increase performance.

3.3 Domain Ontology Repository

Domain ontology repository is available as a common platform for collaborative ontology development and refinement that will be used with semantic information service. This component could be viewed as a standalone server with installed revision control system (Mercurial in our case). This way, users can participate in the joint development of domain ontologies, or use any available ontology that will suit their needs. Semantic information service will be constantly referring to this repository while processing user quires.

3.4 Users and clients of Grid-DL

Since all interaction with semantic information service is carried out via web-services, any application that supports standard web-service technology stack (URI, XML, SOAP, WSDL) could be a client of Grid-DL. The description of provided services could be accessed through URL: "server:port/Grid-DL/ServiceFacade?wsdl".

To administrate Grid-DL and monitor the state of all incoming requests we developed a simply web-interface available through URL: "server:port/Grid-DL/qtasks".

4 Interaction with semantic information service

The query to Grid-DL must be an OWL-class expression that would represent the instances of desired resources. Upon query submission, Grid-DL returns a unique request Id that will be used to retrieve results.

In its simplest form, when we do not use or take into account the domain ontology repository, users should be familiar with the content of core system TBox in order to send a query request using OWL Manchester syntax [7].

When using domain ontology repository, a client must specify in its request what ontologies should be used during classification. The path to ontologies is specified relative to the root of the domain ontology repository. For example: "/general/operatingSystems.owl". Upon incoming request, Grid-DL will synchronize with domain ontology repository and acquire all the necessary files that will be used by OWL-reasoner during a query execution.

An interaction with domain ontology repository is carried out according to usual procedures of interaction with distributed control revision system.

Let us consider a small example. A user that has an access to Grid system and is a member of some virtual organization wants to conduct a molecular dynamics computation. He starts by browsing what domain ontology is used within his organization and sees following declarations:

```
MolDynSubCluster ≡ GROMACS_Cluster or LAMMPS_Cluster
MolDynCE ≡ ComputingElement and partOf some (Cluster and
contains some  MolecularDynamicsSubCluster)
```

```
GROMACS_App ≡ ApplicationSoftware and hasRunTimeEnvironment
some string [pattern "GROMACS"]
GROMACS_Host ≡ Host and describedBy some GROMACS_App
GROMACS_Cluster ≡ MPI_SubCluster and X86_64_SubCluster and
(SubCluster and describedBy some GROMACS_Host)
```

This ontology, among other thing, defines molecular dynamics software packages and Grid resources capable of running them. The definition of `MPI_SubCluster` and `X86_64_SubCluster` is drawn from more general ontology, which is used in all virtual organizations. In particular there will be a definition of a MPI-enabled cluster and x86-64 platform:

```
OPENMPI ≡ ApplicationSoftware and hasRunTimeEnvironment
  value "OPENMPI"
MPICH ≡ ApplicationSoftware and hasRunTimeEnvironment
  value "MPICH"

MPI_Library ≡ MPICH or OPENMPI
MPI_Host ≡ Host and describedBy some MPI_Library
MPI_SubCluster ≡ SubCluster and describedBy some MPI_Host
MPI_Cluster ≡ Cluster and contains some MPI_SubCluster

X86_64_Arch ≡ Architecture and hasPlatformType value "x86_64"
X86_64_Host ≡ Host and describedBy some X86_64_Arch
X86_64_SubCluster≡ SubCluster and describedBy some X86_64_Host
X86_64_Cluster ≡ Cluster and contains some X86_64_SubCluster
```

At this stage our user adds his personal assertions, such as an available computing element and finally defines a computing element he is looking for (`CEForMyWork`):

```
Availible_CE ≡ ComputingElement and hasState some
  (CEState and hasRunningJobs value 0 and hasWaitingJobs
  value 0 and hasFreeJobSlots some integer[>0])

MyVoACL ≡ AccessControlBaseRule and hasPrefix value "VO"
 and hasSCN value "myVO"

CEForMyWork ≡ MolDynCE and Availible_CE and
  hasAccessControlBaseRule some MyVoACL
```

Finally user submits `CEForMyWork` query to Grid-DL, specifying ontologies he just used and retrieves all available computing elements on the Grid that could carry out his task. This way user stays almost isolated from the complexity of the Grid.

5 Future work

When working with the LCH Grid, we acquire a knowledge base with over 900,000 axioms for more than 21,000 named individuals, with data property assertions being dominant.

All modern OWL reasoners have significant difficulties classifying ontology of such size and structure. In fact it takes more than few hours to complete. That is the reason we are currently moving away from the tableaux reasoners because of severe performance penalties that come with it. We are also in the process of switching our core ontology to the OWL EL profile for the same reasons, sacrificing some expressivity for polynomial complexity.

Work is being done to switch to ELK [8] reasoner, which has proved to be one of the most well optimized reasoners for EL profile. Currently we are working on a sufficient datatype support[3] for ELK beyond EL profile in order to carry out our task.

6 Conclusion

Application of semantic technology opens up many possibilities and prospects for further improvement of the basic elements of Grid systems, promoting the emergence of new models of user interaction with them. We set a goal for "intellectualization" of key Grid systems to promote it to a larger audience of users that sometimes have difficulties adjusting to way Grid is operated.

A source code of presented prototype[4] is freely available for application and improvement.

7 References

1. *Andreozzi S., Burke S., Donno F. et. al.*: GLUE Schema Specification (version 1.3) – http://glueschema.forge.cnaf.infn.it/Spec/V13
2. *Czajkowski K., Fitzgerald S., Foster I., Kesselman C.:* Grid information services for distributed resource sharing. Proc. of the 10-th IEEE International Symposium on High Performance Distributed Computing. – IEEE Press. – 2001. – P. 181-195.
3. Berkeley Database Information Index V5 Documentation – https://twiki.cern.ch /twiki/bin/view/EGEE/BDII/
4. The Protégé Ontology Editor and Knowledge Acquisition System – http://protege.stanford.edu/
5. OWLAPI Project homepage, http://owlapi.sourceforge.net/
6. *Sirin E., Parsia B., Grau B. et al.:* Pellet: A practical OWL-DL reasoner. Web Semantics: science, services and agents on the World Wide Web. – 2007. – Vol. 5, N 2. – P. 51-53.
7. *Horridge M., Drummond N., Goodwin J.:* The Manchester OWL syntax. Second International Workshop OWL: Experiences and Directions (OWLED 2006). – 2006. – Vol. 216.
8. *Yevgeny Kazakov, Markus Krötzsch, František Simančík*. Concurrent Classification of EL Ontologies. In Aroyo et al. (eds.): Proceedings of the 10th International Semantic Web Conference (ISWC-11). LNCS 7032, Springer 2011.

[3] https://elk-reasoner.googlecode.com/svn/branches/elk-parent-datatypes/

[4] https://github.com/pospishniy/Grid-DL