The Knowledge of the Grid: A Grid Ontology

Michael Parkin[1], Sven van den Burghe[2],
Oscar Corcho[1], Dave Snelling[2], John Brooke[1]

[1] School of Computer Science and Manchester Computing,
The University of Manchester, M13 9PL, United Kingdom.
[2] Distributed Services Research Group,
Fujitsu Laboratories of Europe Ltd., Middlesex, UB4 8FE, United Kingdom.

Abstract

This paper presents a knowledge architecture and set of ontologies that can be used as the foundation to facilitate the matching of abstract resource requests to services and resources, to determine the functional equivalence of Grid middlewares and deployments and to allow the description of 'hybrid' compound Grids composed of individual heterogeneous Grids. This is necessary as in all these cases what is required is mediation between different views or descriptions of Grids, which requires a formal reference vocabulary. We present a framework and ontologies for achieving this.

1 Introduction

The Grid computing vision is to enable coordinated resource sharing through the creation of application-independent middleware and protocols. In Japan and Europe there are, mainly, two Grid systems deployed: Globus [1], predominantly deployed in cluster based computing projects, and Unicore [2], typically used to allow access to heterogeneous high performance computing (HPC) architectures.

The initial impetus for the development of a Grid Resource Ontology was the need to develop a resource broker that could use either the MDS structure adopted by Globus or the information in Unicore's Incarnation DataBase (IDB). On closer examination the critical translations that needed to be established were between the Grid Uniform Laboratory Environment [3] used to provide a uniform resource description for Grid projects based on Globus and the syntax used by Unicore for the entries in the IDB [4]. The solution to this problem was ultimately to develop along with other projects standard languages for common Grid patterns of actions such as job submission [5].

However in the course of this work a deeper issue arose, namely how we assert that two different Grid systems are providing the same functionality. This led us to examine not just the syntax but also the semantics of both Globus and Unicore. In the terms developed by the proposers of the Semantic Grid we were examining knowledge of the Grid in contrast to the majority of work on Semantic Web, which is directed towards knowledge in documents and services on the Web or Grid [6]. We aim to capture the semantics implicit in the architectural assumptions of differing Grid systems and to formalize them in a

standard ontological language. We believe that if sufficient community consensus can be captured in such a formalism, it will be possible to develop reasoning tools which can be used to check if a pattern of actions can be performed on any given Grid system. We can thus check if interoperability is possible rather than just assuming it to be desirable and then finding for each interoperability venture ad-hoc methods of working across Grids.

We believe that this task is an urgent one for the Grid community, since there now exist several other middleware systems aiming to capture entire Grid functionality (gLite, OMII-UK, CROWN, NAREGI, NordicGrid, DGrid). Thus pair-wise translation of corresponding terms leads to a combinatorial explosion. We also claim that careful modeling and formalization of Grid architectures provides important guidance for the development of Grid and Web Service standards. Since such modeling involves community consensus we describe also how we intend to enable community participation in the further development of the modular ontology we have developed as a starting point.

2 Methodology for Modeling

2.1 Requester-Provider Duality

This work is motivated by considering how a Grid operates, since observation of Grids indicates there are two parties interacting when an piece of work is requested from/on the Grid; the 'client' application that forms, describes and requests the work and the resource providers that are willing to share their resources and accept and execute these pieces of work[1]. This is a fundamental, abstract duality that underlies the idea of coordinated resource sharing that allows us to capture many of the complex interactions in a Grid ecosystem.

However, an impediment to these two parties interoperating is the fact that they may not, and need not, see the Grid in the same way. For example a scientist's meteorological simulation may specify the resolution of the computational mesh, the time for which the simulation is to be run, and the different physics routines to be used. This has implications for resource consumption but is not directly expressible in CPU numbers and speed and memory size, which is how grid nodes are described. Furthermore a client and resource provider may use different languages to describe 'their' interpretation of a Grid request.

There exists, therefore, a requirement for mediating or reconciling these two views of the Grid by producing a common vocabulary that defines, relates and allows us to share interpretations of the Grid. Thus, to enable this mediation we can build ontologies, defined as "formal, explicit specifications of a shared conceptualization" [7], to allow the formal definition of terms and relationships that will be used by Grid middleware and applications in order that the metadata that we generate is explicitly described and represented and can be shared more easily among different implementations and deployments.

[1] This does not preclude other third parties, e.g. resource brokers, since the requester-provider model works recursively. For example, a broker is a provider to a client and a requester to Grid resource management systems.

2.2 Grid Ontology Requirements

Thus, our ontology must fulfill two requirements in order to satisfy the client/resource provider view of the Grid we have presented: firstly, it must allow us to express resource requirements in an abstract, resource and middleware independent form. Secondly, the ontology must express, again in an abstract manner, both the actions requested and the resources that enable these actions. In this paper we refer to the process of translating between these two forms as *incarnation*, a term that was originally used by Unicore, literally meaning "the act of causing to exist"[2]. For example, an abstract directory representation in Unicore (a Java object) is incarnated by the Network Job Supervisor into a concrete directory on the resource itself. The abstract notion of the directory is generated by the need to perform the action of creating a container for files, the concrete representation as a directory pathname is the concrete realization on a given system.

Having an ontology that fulfills these requirements allows us to perform many tasks. We can use the ontology to:

- Group actions and the dependencies between then in an abstract form that can be translated into differing workflow languages.
- Tie an abstract resource to an abstract action that can be translated into a JSDL or native Resource Manager syntax job request.
- Create interoperable Grids by demonstrating that patterns of actions or resources described in different middleware syntax are equivalent, this avoids the need for pair-wise mapping of middleware-specific terms.

3 Relationship to Other Work

Creating a common, structured set of terms and vocabularies to Grid describe applications and middleware is not new. The GLUE schema, Unicore Abstract Job Objects (AJOs), the Common Information Model [8] and the OGSA Glossary [9] are good examples of existing vocabularies. However, these examples lack some of the characteristics of ontologies we have described above. They either: lack formality in their definitions, which makes their usability and reusability more difficult; do not express a shared point of view of Grid applications and middleware; are not explicit; or they do not cover all the concepts and components a Grid may have. A brief description of these efforts is now given.

- **GLUE Schema**. The Grid Laboratory Uniform Environment (GLUE) Schema was developed to provide "an abstract model ... for Grid resources and mapping to concrete schemas that can be used in Grid Information Services" [3]. GLUE was developed for use with the Globus Toolkit's Monitoring and Discovery Service (MDS) so that Globus users could determine what resources were and what they did according to the reference schema. Thus, the GLUE Schema fulfills one half of the requirements described in Section 2.2 – that of describing the capabilities of the resource provider.

[2]The Oxford English Dictionary.

It does not, however, fulfill the second half of our identified requirements, i.e. describing abstract requests for resources.

- **Unicore AJOs**. The Unicore AJOs are a set of open-source Java classes that provide an abstract language with implicit semantics to describe a piece of work to be performed on a computational resource. A job can contain a single AJO or groups of AJOs that may have dependencies on each other; i.e. AJOs can be grouped into directed acyclic graphs describing complex interdependent computational Grid tasks, or workflows. AJOs provide an elegant model to describe abstract resource requests. However, the semantics of the AJOs are not explicit, they do not describe the capabilities of the Grid resources, nor do they describe aspects of a Grid such as it's security model, infrastructure and processes, etc.
- **The DMTF Common Information Model (CIM)**. CIM, a standard and schema defined by the Distributed Management Task Force (DTMF), provides a comprehensive, formal model of "management information systems, networks, applications and services" [8] and the relationships between them. However, it does not capture certain aspects of the Grid such as the description of abstract jobs and processes such as incarnation.
- **The OGSA Glossary**. The OGSA Glossary has been produced by the Global Grid Forum (GGF) to "provide information to the Grid community regarding the concepts and terms used by the Open Grid Services Architecture" [9]. However, as comprehensive and accurate as the glossary is, it is not suitable as a basis for an ontology of the Grid as it lacks formality, i.e. the descriptions are in natural language and have no structure.
- **Unique Ontologies Developed for Specific Applications**. Recently, ontologies have developed for Grid middleware for the purposes of exposing metadata of Grid services and resources. This has been done for several purposes, including service and resource discovery [10], enabling data integration [11, 12] and controlling authorization to resources [13]. All these developments embrace the vision of the Semantic Grid – an extension of the Grid where rich resource metadata is exposed, then shared. The layering of an explicit semantic infrastructure over Grid infrastructure has the potential to increase interoperability and provide greater flexibility when composing services. In the context of the applications that have been previously developed, different ontologies were produced with many of them the including the same terms but redefined in different ways due to the lack of a 'reference' ontology to describe them. Thus, it is the aim of this work to produce that reference ontology to 'ground' Grid concepts.

4 An Ontology of the Grid

4.1 Knowledge Architecture

The structure, or knowledge architecture, of the modular ontologies is shown in Figure 1, where 'import' denotes that an ontology imports the concepts from another ontology. Thus, the Foundational Ontology defines the high-level, com-

mon, general-purpose Grid concepts that can be re-used in the description of any Grid middleware or application, protocols, services, resources and Virtual Organizations. This ontology is described in detail in Section 4.2. Within the Base Concepts layer we have separated out ontologies containing OGSA [14] and S-OGSA [15] concepts because although OGSA and S-OGSA are middleware independent architectures for Grids, not all middleware is based on these architectures. Thus, the Foundational Ontology contains agreed concepts that are present in all Grid middleware (including some OGSA concepts) whilst the OGSA and S-OGSA ontologies contain terms that are only relevant to them.

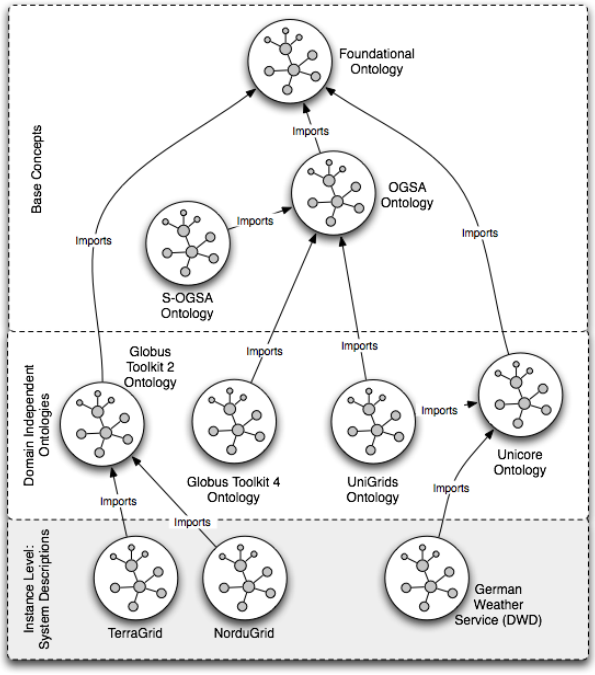

Fig. 1: The Grid Knowledge Architecture

The second tier of ontologies in our hierarchy provides domain-independent but middleware-specific ontologies that describe instances of implementations of Grid middleware. As examples, Figure 1 shows how Globus Toolkit 2, Globus Toolkit 4, Unicore and UniGridS [18] middleware fits into this hierarchy, though we could have also included middleware such as Condor, for example. As we have described, the ontologies in this layer import and extend the Base Concepts; UniGrids and Globus Toolkit 4 both import the OGSA ontology since these implementations are based on OGSA. Note that ontologies can also have dependencies between each other; the UniGrids ontology imports concepts from Unicore as it has concepts based on that middleware. The final layer in our ontology hierarchy contains instances of actual Grid deployments and is described

in Section 4.4.

4.2 Foundational Ontology

The Foundational Ontology contains 104 classes and 154 object properties. Due to reasons of space it is not possible to list and describe them here, therefore a selection of classes (denoted by `typewriter` font) and their properties have been chosen to illustrate important concepts.

4.2.1 User & Security Concepts

Concepts relating to the use of a Grid are tightly-coupled with security concepts, therefore a selection of user and security concepts are presented together.

- A `User`[3] (who may have zero or more `Identities`) interacts with the Grid.
- An `Identity` is an attribute that "allows entities to be distinguished from all others" [16] and is incarnated to a physical `Xlogin` on a system.
- An `Identity` is established by at least one `Credential`.
- A `Credential`, issued by a `Credential Issuer`, establishes an `Identity` by being authenticated by an `Authentication Point`.
- An authenticated `Credential` is authorized to perform a task on a Grid by an `Authorisation Point`.

4.2.2 Action & Activity Concepts

Fundamental to the Grid is the ability to describe pieces of work in an manner not specific to a Grid implementation, thus allowing portability of jobs and the evaluation and execution of jobs based on their explicit semantics. This section describes some of these concepts and their properties.

- Each `Action`, the superclass of all the types of abstract actions supported, has a `Action State` and, eventually, an `Outcome`.
- An `ActionGroup` is a type of `Action` which can contain zero, one or more `Actions` that may have `Dependencies` on other `Actions`. As it is a subclass of `Action` it has both an `Action State` and `Outcome`.
- A `Task`, a type of `Action`, is "a definable unit of work" [14].

A piece of work for the Grid expressed in these abstract terms it is then incarnated at an `Incarnation Point` to an `Activity`, which is runnable on a `Resource Collection`. A running `Activity` is a `Job`, which has a `Job State` and may have an `Input` and `Job Working Space`.

4.3 Domain-Independent Ontologies

The domain-independent ontologies describe middleware-specific concepts. Examples from the Unicore and Globus Toolkit 2 middlewares are shown here.

- A `Globus GateKeeper` is a type of `Access Point` that supports the `Globus GSI` (Grid Security Infrastructure) protocol.

[3]Note a User is not the same as a Client, a piece of software that interacts with the Grid.

- `Globus MDS` is an `Information Service` that supports Globus's `GRRP` and `GRIP` (Grid Resource Registration/Inquiry Protocols).
- A `Unicore Gateway` is an `Access Point` to a Unicore `USite` (a type of `Administrative Domain`) that supports the `UPL` (Unicore Protocol Layer).

4.3.1 Inferred Types & Semantic Equivalence of Concepts

Examples of how, using the properties of a concept, we can deduce or *infer* what other types of concept it is and show the semantic equivalence of middleware-specific concepts are now given. In the Foundational Ontology we have defined that a `Service` is anything that supports a `Protocol`. Thus, as the `Unicore Gateway` supports the `UPL` (a type of `Protocol`) reasoning over the ontology tells us that a `Unicore Gateway` is also a `Service`. Further, the `Unicore Gateway` can also be deduced as an `Authentication Point` as it also authenticates `X.509 Certificates` (a type of `Credential`). `Globus Gatekeepers` also authenticate `Credentials`, therefore they too are `Authentication Points` and the `Unicore Gateway` and `Globus Gatekeeper` can be said to be semantically equivalent. Thus we can use outcomes of reasoning over the ontology to map terms between heterogeneous Grids and promote interoperability between them.

4.4 Instance-Level Ontologies

The final layer in the Grid knowledge architecture presented in this paper is intended to contain concrete *instances* of Grids and hold details of Grid systems and deployments described using the Grid ontology. The UK's National Grid Service, US TeraGrid and Earth Science Grids, NorduGrid, etc. are all examples of production Grids that would be included in this layer of the hierarchy.

5 Conclusions & Future Work

This work has briefly presented a knowledge architecture and reference ontologies for Grid computing that not only capture concepts such as a Grid's infrastructure but also how it is used by Grid consumers (through abstract descriptions of Grid actions) together with common Grid processes. By taking this novel perspective of the Grid we have provided the foundations for describing Grid resource/service usage and interoperability based on the semantic descriptions of Grids and Grid users' abstract actions that can be incarnated into the representation appropriate for that Grid.

Our future work will concentrate on promoting the uptake and use of these ontologies within EU IST projects such as OntoGrid. Barcelona Supercomputing Center are using the ontology with their GRID Superscalar programming framework to develop a prototype system to provide advanced scheduling techniques based on the semantic properties of client job submitted to resource providers.

The Foundational, OGSA, S-OGSA, Globus and Unicore ontologies we have introduced in the paper are available from `http://www.unigrids.org/ontology.html` for public inspection, use and comment.

6 Acknowledgments

Michael Parkin's work was supported under the RealityGrid DTA programme funded by EPSRC. Oscar Corcho's work is supported by the EU FP6 OntoGrid project (STREP 511513) funded under the Grid-based Systems for solving complex problems, and by the Marie Curie fellowship RSSGRID (FP6-2002-Mobility-5-006668). John Brooke, Dave Snelling and Sven van der Berge's work was supported by the EU FP6 UniGrids project (STREP 004279).

References

1. Globus: A Metacomputing Infrastructure Toolkit. I. Foster, C. Kesselman. International J. Supercomputer Applications, 11(2): pp. 115-128. Summer 1997.
2. UNICORE: A Grid Computing Environment. D. Erwin, D. Snelling. Proceedings of 7th International Euro-Par Conference: pp. 825-834. Aug. 2001.
3. The GLUE Schema, Version 1.2. S. Andreozzi *et al.* Dec. 2005.
4. Semantic matching of Grid resource in Grid Computing. J.M. Brooke, D. Fellows, K. Garwood and C. Goble. Second European AcrossGrids Conference. Revised Papers Editors: Marios D. Dikaiakos, LNCS 3165, pp. 240-249. Jan. 2004.
5. Job Submission Description Language (JSDL) Specification, Version 1.0. A. Anjomshoaa *et al.* GGF Document GFD-R.056. Nov. 2005.
6. Enhancing Services and Applications with Knowledge and Semantics. C. Goble, *et al.* Ch. 23 in The Grid 2: Blueprint for a New Computing Infrastructure. 2004.
7. Knowledge Engineering: Principles and Methods. R. Studer, V. R. Benjamins, D. Fensel. IEEE Transactions on Data and Knowledge Engineering 25(1): pp. 161-197. March 1998.
8. DMTF Common Information Model (CIM), Version 2.1.3. Sept. 2006.
9. Open Grid Services Architecture Glossary of Terms. J. Treadwell (ed.). GGF Document GFD-I.044. Jan. 2005.
10. A Suite of DAML+OIL Ontologies to Describe Bioinformatics Web Services and Data. C. Wroe *et al.* International Journal of Cooperative Information Systems 12(2): pp. 197-224. March 2003.
11. The Earth System Grid Ontology. `http://marinemetadata.org/vocabularies/refs/mapping/document.2005-05-27.6566880301/` Nov. 2006.
12. An Ontology-Driven Framework for Data Transformation in Scientific Workflows. S. Bowers, B. Ludaescher. International Workshop on Data Integration in the Life Sciences: pp. 1-16. March 2004.
13. An Authorisation Scenario for S-OGSA. P. Alper *et al.* Proceedings of Posters and Demos, 3rd European Semantic Web Conference: pp. 7-8. June 2006.
14. The Physiology of the Grid: an Open Grid Services Architecture for Distributed Systems Integration. I. Foster, *et al.* OGSI Working Group, GGF. June 2002.
15. An Overview of S-OGSA: A Reference Semantic Grid Architecture. O. Corcho *et al.* Journal of Web Semantics 4(2): pp. 102-115. June 2006.
16. RFC 2828 - Internet Security Glossary. R. Shirey. Network Working Group, The Internet Society. May 2000.
17. eXtensible Access Control Markup Language (XACML) Version 2. OASIS Standard. T. Moses (ed). Feb. 2005.
18. UNiform Interface to GRID Services. `http://www.unigrids.org`