Semantic Description and Matching of Grid Services Capabilities

Ye Zhang and William Song
Department of Computer Science
University of Durham
ye.zhang@durham.ac.uk, w.w.song@dur.ac.uk

Abstract

In the Grid environment, shared resources and users typically span different organizations. Thus the demand on semantic description and an effective and efficient search support for Grid services is becoming increasingly significant. Currently we use Globus Metacomputing Directory Service (MDS) to register Grid services. However, MDS is limited in semantic expressiveness for matching. MDS does not fully support semantic descriptions of Grid services and only supports keyword based search, so it is quite problematic to reflect the features of services. Therefore, we propose an approach, Semantic Grid Node (SGN) framework, to build up a semantic description and representation for Grid services. We also develop a semantic based search and reasoning engine for the Grid services registration and discovery.

1 Introduction

The UK e-science programme aims to develop support for large-scale science through distributed global collaborations. A significant focus of the e-science programme is the development of a communication and computational infrastructure to underpin the work of scientists termed the Grid. The Grid promotes the rapid formation of virtual co-laboratories allowing scientists to work together and share resources irrespective of the location of the scientists or the resources they are using.

There are a number of grid applications being developed in e-Science. To achieve the flexible assembly of Grid service requires information about the functionality, availability and interfaces of the various services. Currently the Grid services use metadata to describe the nodes and implement the nodes' discovery. Metadata is key to achieving the Grid Service vision. However, simple metadata and simple queries give a small but not insignificant improvement in information integration.

Traditional service matching is done based on symmetric, attribute-based matching. Globus MDS [5] and UDDI are two such examples; MDS has been widely used in the Grid community for node discovery while UDDI has been used in the web community for business service discovery. Both MDS and UDDI support simple query languages. In these systems, the values of attributes advertised by nodes are compared with those required by jobs. For the comparison to be meaningful and effective, the node providers and consumers have to agree upon attribute names and values. The exact matching and coordination between providers and consumers make such system inflexible and difficult to extend to new characteristics or concepts. Moreover, in a heterogeneous multi-institutional environment such as the Grid, it is difficult to enforce the syntax and semantics of node descriptions. Therefore, MDS and UDDI do not offer expressive description facilities, nor provide sophisticated matchmaking capabilities.

We propose to use the Semantic Web technologies to describe the Grid nodes and to discover the most suitable nodes that meet the users' requirements in the Grid. We developed a Semantic Grid Node (SGN) framework and a new semantic makeup language OWL-SGN based on SGN framework. We propose a flexible and extensible approach for Grid node selection using an ontology-based matchmaker.

2 Semantic Grid Node Framework

In the Grid environment users and software agents should be able to discover, invoke, compose, and monitor Grid nodes offering particular services and having particular

properties. An important goal for Semantic Grid, then, is to establish a framework within which these descriptions are made and shared. Web sites should be able to employ a set of basic classes and properties for declaring and describing nodes. Therefore, we develop a semantic description framework for Grid nodes, called Semantic Grid Node Framework (SGN Framework), see Fig. 1.
Our structuring of the ontology of nodes are motivated by the need to provide several essential types of knowledge about a node, each characterized by the question it answers:

- What do the node require of the user(s), or other agents, and provide for them? The answer to this question is given in the ``Content.'' Thus, the class NODE presents a NODECONTENT.
- How does it work and how it can be controlled? The answer to this question is given in the ``Operation.'' Thus, the class NODE is describedBy a NODEOPERATION.
- What user is fit for the node? The answer to this question is given in the "User". Thus, the class NODE provide a USER.
- What does the resource provide? The answer to this question is given in the "Resource". Thus, the class NODE is supportedBy a NODERESOURCE.
- What relationship between Node and Node? The answer to this question is given in the "Relationship". Thus, the class NODE has a RELATIONSHIP.

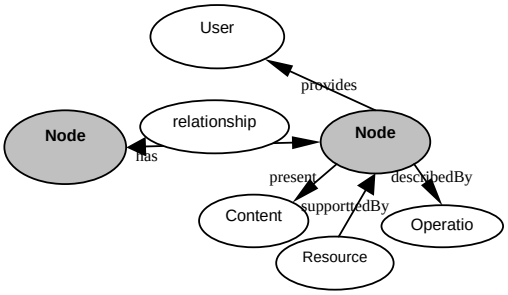

Fig. 1 Semantic Grid Node Framework

The class ***NODE*** provides an organizational point of reference for declaring Grid nodes; one instance of ***NODE*** will exist for each distinct published node. The properties presents, describedBy, provides, has and supporttedBy are properties of ***NODE***. The classes ***NODECONTENT, NODEOPERATION, USER, RELATHIONSHIP*** and ***NODEGRESOURCE*** are the respective ranges of those properties. Each instance of ***NODE*** will present a descendant class of ***NODECONTENT***, be describedBy a descendant class of ***NODEOPERATION***, has a descendant class of ***RELATIONSHIP***, provide a descendant class of ***USER***, and be supporttedBy a descendant class of ***NODERESOURCE***. The details of contents, operations, relationships, users and resources may vary widely from one type of node to another--that is, from one descendant class of ***NODE*** to another. But each of these five classes provides an essential type of information about the node.
Generally speaking, the ***NODECONTENT***, ***NODERESOURCE, RELATIONSHIP*** and ***USER*** provide the information needed for an agent to discover a node. Taken together, the ***NODEOPERATION***, ***NODERESOURCE*** and ***RELATIONSHIP*** objects associated with a service provide enough information for an agent to make use of a service. In this paper, we focus on node discover, hence, we only introduce ***NODECONTENT***, ***NODERESOURCE, RELATIONSHIP*** and ***USER*** in details.

NODECONTENT is a set of functions that the node is to possess. It provides the descriptive information for a service-seeking agent to determine whether the functions provided by the node meet its needs. There are some human-readable properties including *nodeName*, *nodeDescription* and *contactInformation*. Other properties of node content include *nodeFunctionality*, *nodeCategory* and so on.

NODEREOURCE defines a family of specifications for accessing resources using Grid services and service resource has own state. Resource is one of the main distinguishing characteristics of the Grid services application domain (e.g., data, instruments, computers, humans, etc.) It includes the *ResourceProperties*, *ResourceLifetime*, *BaseFaults*, and *ServiceGroup* specifications

RELATIONSHIP describes an association between node and node, based on attributes of those two entities. A relationship maintains a set of identifying attributes and ontology information from the related entities. Most relationships simply relate the objects of one entity to those of another by comparing attribute values and ontology information between them. Node can generate some attributes and ontology information from the other nodes when they have some relationships. It includes *SomeDistincFrom AllDistinctFrom*, *Equivalent* and so on.

USER describes a software agent that wishes to interact with a node in order to request that a task be performed on behalf of its owner that the person or organization that wishes to use a node. It includes *userName, userDescription,*

userCategory, *userLocation* and *userQuality*.

3 OWL-SGN

Efforts toward the creation of the Semantic Web are gaining momentum. Soon it will be possible to access Web resources by content rather than just by keywords. A significant force in this movement is the development of a new generation of Web markup languages such as OWL and its predecessor DAML+OIL. These languages enable the creation of ontologies for any domain and the instantiation of these ontologies in the description of specific Grid nodes.

Grid nodes do not merely provide static information but allow one to effect some action or change in the world. The Semantic Web should enable users to locate, select, employ, compose, and monitor Grid nodes automatically. A software agent needs a computer-interpretable description of the node, and the means by which it is accessed. An important goal for us, then, is to establish a framework in which these descriptions are made and shared and represent it using current Semantic Web languages. Grid node should be able to employ a set of basic classes and properties for declaring and describing nodes, and the ontology structuring mechanisms of OWL provide the appropriate framework within which to do this.

Our contribution is to use OWL syntax [2] to represent our SGN framework. Actually, SGN framework can be represented in many different semantic makeup languages such as RDF, DAML+OIL and OWL, and we choose the most advanced one – OWL. Probably the other upper ontology languages will appear in the future, and then we should change to deal with it.

4 Semantic Grid Node Matching

Users' discovery of a service node that satisfies requirements means that a published service matches a user request. The concept "match" means that the description of a published node (or advertisement) is sufficiently similar to that of the user's requested node. Of course, the problem of this concept lies in that how similar is "sufficiently similar". One extreme is that, an advertisement and a request are "sufficiently similar" if they are describing exactly the same node. This definition is too restrictive, because it is not realistic that the advertisers and the requesters have reached an agreement on how a node is represented; let along that they usually have quite distinct objectives and perspectives. This too restrictive criterion on matching is therefore deemed to have little opportunity to find "matches" between the service providers' advertisements and the users' requests of nodes.

Consequently, we need to allow matching engines to perform flexible matches. For example, the match engines suggest the different degrees of similarities between the advertisements and the requests. The node requesters should also be able to decide the degree of flexibility. In the following, we just briefly define three types of matches:

Full match (request.input <= advertisement.output) – This is the situation where the node advertised contains more than or equal to the node requested. Full match is the best situation because the result returned may probably give more than what the requester expects.

Part match (request.input > advertisement.output) – This is the situation where the provided node cannot completely fulfill the request. In other words, the set of the advertised functions is a subset of the set of the requested functions.

Outer match (request.input and advertisement.output are disjoint): No subsumption relation can be found between advertisement and request is identified.

A match between an advertisement and a request consists of all the matches of their children nodes, parent nodes and the nodes themselves. Our match algorithm is mainly using the LARKS matching algorithm [1] based on OWL-SGN.

For every element in OWL-SGN, the algorithm delivers a large set of match candidates. Hence, the first results may still be too large for many matching tasks. The volume of the candidate nodes presented to a human being may be also overwhelming, even if the candidate nodes are ordered in ranking. We provide three filters for this large amount of matching results: text filter, domain filter and I/O filter, which can help to narrow down the candidate nodes.

Text Filter: Pre-checking process to filter out those human-readable node explanation parts, for example, comments, text descriptions, etc.

Domain Filter: checking whether the advertised node and the requested node are in the same branch in an ontology hierarchy.

I/O Filter: I/O Filter uses three degrees of match

methods: exact (i.e. requst.input=advertisement.output), full and part. The requester can determine which degree of match methods is needed. This is an enabling solution that opens the door to the semantic engines that will ultimately provide the semantic functionality to Grid services

The users can select any combination of these filters at the search time.

5 Implementation and Evaluation

Our prototype matchmaker currently supports semantic node matchmaking. Currently, the matchmaker reads available nodes and requests (described by vocabularies in the resource and request ontology, respectively) from an OWL-SGN file. Ontology instances are manually created (which saves instances in the OWL-SGN format).

The user can then activate the matchmaker by submitting a query asking for nodes that satisfy the request specification. The query is then processed by database system using matchmaking algorithm, in combination with background knowledge and ontologies, to find the best match for the request.

Our design and development of the matchmaker follows the DFACE (Define-Focus-Analyze-Create-Evaluate) methodology from Stanford University.

The responses of the users in terms of requirements for Grid services include three aspects. The most important requirement from both of system integrators and users is the reduction of the cost of development and administration. The second requirement is interoperability with the current service systems, i.e., to maintain the standards already in use. The third one is a track record and security in general. In term of Grid services search engine itself, the most important thing is easy to use and fast search speed, rather than advanced functions. Those requirements have made us decide to make our system to be compatible with the current standards and technologies. Therefore, we expect this Grid node matchmaker contributes to the track record.

6 Conclusion

We briefly propose a semantic description framework for Grid service based on Semantic Web technology OWL-SGN, and design a semantic based search engines for the Grid nodes. This is an enabling solution that opens the door to the semantic engines that will ultimately provide the semantic functionality to Grid services in the UK e-science programme.

Next step we will focus on implementation of the Semantic Grid Node matchmaker to manage the interactions among Grid services. We are constructing an OWL-SGN virtual machine to handle these interactions among Grid services.

References

1. Katia Sycara, Seth Widoff, Matthias Klusch and Jianguo Lu, **"LARKS: Dynamic Matchmaking Among Heterogeneous Software Agents in Cyberspace."** Autonomous Agents and Multi-Agent Systems, 5, 173–203, 2002.
2. The OWL Services Coalition: **OWL-S: Semantic Markup for Web Services** this white paper accompanied OWL-S version 1.0, http://www.daml.org/services/owl-s/1.0
3. M. Paolucci, T. Kawamura, T. R. Payne, K. Sycara: **Semantic Matching of Web Services Capabilities**, Proceedings of First International Semantic Web Conference (ISWC 2002), IEEE, pp. 333-347, 2002
4. T. Kawamura, J. D. Blasio, T. Hasegawa, M. Paolucci, K. Sycara: **Preliminary Report of Public Experiment of Semantic Service Matchmaker with UDDI Business Registry**, First International Conference on Service Oriented Computing (ICSOC 2003)
5. K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman. **Grid information services for distributed resource sharing**. In the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10). IEEE Press, August 2001.