

S-MDS: A Semantic Information Service for Advanced Resource Discovery and Monitoring in WS-Resource Framework

Said Mirza Pahlevi and Isao Kojima

National Institute of Advanced Industrial Science and Technology (AIST),
Grid Technology Research Center, Tsukuba, Ibaraki 305-8568, Japan
{mirza, kojima}@ni.aist.go.jp

Abstract

This paper proposes an information service framework to support automatic service discovery and monitoring in WS-Resource Framework (WSRF). The service framework uses ontologies to describe service state and provides an efficient mechanism to aggregate and maintain the ontology instances. The mechanism utilizes the constructs and message exchanges defined by the WSRF specifications so that it is applicable in any WSRF implementation. To demonstrate the practicality of the framework we have developed a prototype system in a small testbed cluster using the de facto standard, Globus Toolkit 4.0. We also present performance evaluation results showing the usability of the framework.

1 Introduction

The WS-Resource Framework (WSRF) defines a family of specifications for accessing stateful resources using Web services. The specifications define what is termed the *WS-Resource approach* to modeling and managing state in a Web services context. A *WS-Resource* is defined as an entity composed of a *stateful resource*, such as a database or file, and a *Web service* with which it interacts.

While WSRF defines the means for querying, monitoring, and aggregating WS-Resources, it does not provide a mechanism to use the constructs to facilitate automatic service/WS-Resource discovery and monitoring. Automatic Web service discovery, invocation, composition, and interoperation constitute an important issue in the semantic Web service community. Needless to say, such automatic tasks are also important in WSRF because a Web service is one component of a WS-Resource.

Providing such mechanism in WSRF context should meet the following requirements. First, since a WS-Resource includes a stateful component that is described by resource property values, the mechanism should expose the values to allow good resource selection and discovery. Second, since the resource property values may change frequently over time, the mechanism should provide up-to-date resource property values. Third, since WS-Resources are (mainly) transient, the mechanism must consider the WS-Resource lifetime so that it does not provide invalid information. Fourth, to ease sharing of data across applications/domains and implementation of the automatic tasks, the mechanism should present the data including semantic information.

The following use case illustrates the need for semantic information to perform automatic service discovery and selection in the Globus Toolkit (GT) 4.0 environment [1]. Suppose there is a group of bioinformatics databases exposed by OGSA-DAI Data Services [2] as relational databases and Grid Resource Allocation and Management (GRAM) services [3] to submit jobs on grid computing resources. To discover a new drug, suppose a pharmacologist needs to join data from several available databases and perform some simulation task using appropriate grid computing resources, e.g., hosts with UNIX-compatible OS and IA-32 CPU architecture, by providing the (joined) data as one of the job inputs.

The first problem occurs in the table-joining process. The Data Services provide logical schemas of the managed databases via their resource properties. However, since the schemas do not contain semantic information, it is difficult to perform automatic resource discovery and selection to join tables. For example, it would be difficult to automatically find tables that have a column containing a specific domain value, such as a *protein compound*. The second problem occurs in selecting appropriate computing resources, since no concepts of

the UNIX-compatible OS and IA-32 CPU architecture are available in the resource properties of the GRAM service.

This paper proposes a framework for creating, aggregating, and maintaining the *semantic metadata* of WS-Resources. Semantic metadata are information described using an ontology that allows semantic and syntactic interoperability. To harvest the well-developed semantic web technologies, the semantic metadata are described using the OWL-S ontology [4]. The framework is based on the WSRF architecture and utilizes constructs and message exchanges defined by the WSRF specifications. Therefore, any WSRF implementation software can make use of the framework. We expect that intelligent agents/software will be able to use the aggregated semantic metadata to perform advanced resource discovery and monitoring.

The rest of the paper is organized as follows. Section 2 reviews related work. Section 3 describes the proposed framework, S-MDS. Section 4 describes the implementation of the framework with GT4.0 software. Section 5 gives the performance evaluation results. Section 6 concludes the paper.

2 Related Work

GT4.0 provides a service, called the Index Service [5], which is used for grid resource monitoring and discovery. This service collects data from various sources and publishes them as ServiceGroup entries. This service provides structured (XML) data, but without semantic constraints. Therefore, it is quite difficult to perform automatic resource discovery and selection based on such data, as shown in the example in the previous section.

Ontology-based Matchmaker (OMM) [18] is an ontology-based resource selector for solving resource matching in the grid. Resources and requests are described by (different) ontologies and matched using matching rules. OMM is designed for the matchmaking purpose, rather than as a service registry. Moreover, it does not consider any WSRF or OWL-S constructs.

Our previous work [16] also maps the service state into an ontology. The work was done in the context of GT3.2 [6]. This paper extends the previous work so that it can work in the broader WSRF context. It also replaces the proprietary subscription/notification mechanism with a mechanism that uses the standard WS-Notification family of specifications. Furthermore, it extends the previous work by utilizing the existing OWL-S ontology including the supported technologies and tools.

Several approaches have already been suggested for

adding semantics to the Web service standards for improved service discovery. METEOR-S [15] is a framework for semi-automatically marking up Web service descriptions with ontologies. UDDI-M^T [14] deals with Web service directories by allowing users to specify metadata for the information stored in the directories. UDDIe [17] also adds metadata to the directories data, but with a service-leasing mechanism. These systems do not deal with dynamic service property values as in our framework.

3 S-MDS

3.1 Outline

One component of a WS-Resource is a Web service. This framework uses the ontology defined by the OWL-S [4] to describe a WS-Resource. OWL-S is an ontology of service concepts for describing the properties and capabilities of Web services in a machine-interpretable form. Using OWL-S ontology to describe a WS-Resource has the following advantages. First, the existing OWL-S tools, such as matchmakers [7], can use the semantic metadata provided by the framework, thereby accelerating the realization of the automatic tasks in the WSRF environment. Second, the framework can utilize tools built to support OWL-S, such as OWL-S editors [8]. This greatly reduces the effort required to construct and maintain the semantic metadata.

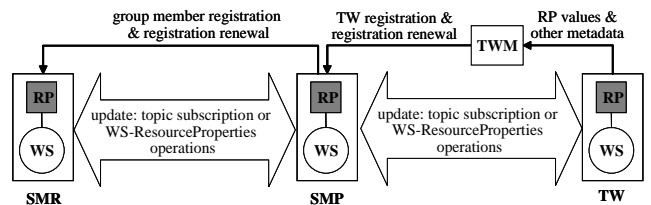


Figure 1. Main components

Figure 1 shows the main components of the proposed framework. Each target WS-Resource (TW) is associated with a TW manager (TWM) and a WS-Resource, called the Semantic Metadata Provider (SMP). A TWM accepts metadata from a TW (including its resource properties) as input and creates *semantic metadata* as output. The output is created by mapping the (given) metadata to a predefined ontology. A TWM registers a TW by passing the semantic metadata of the TW to an SMP and then maintains the registration by periodically updating the registration lifetime.

The role of an SMP is to represent one or more TWs and manage their semantic metadata. For each (registered) TW, SMP registers it to a WS-Resource, called the Semantic Metadata Repository (SMR), by passing the TW’s semantic metadata as the registration parameter. As with the TWM, SMP maintains the registration by periodically renewing the registration lifetime. The function of an SMR is to aggregate and maintain semantic metadata from SMPs. Both SMP and SMR maintain the freshness and validity of semantic metadata using the constructs and message exchanges defined by the WSRF and WS-Notification family of specifications.

3.2 Semantic Metadata Management

On accepting a registration request from a TWM, an SMP creates an *Entry* resource property to store the semantic metadata¹. The SMP then extracts resource property names (of a TW) from the semantic metadata and monitors the property values. The monitoring process is done by periodically pulling the property values (from the TW) using a WS-Resource-Properties operation if the TW does not implement the Notification-Producer interface²; otherwise, it involves subscribing to topics defined for the resource properties. On receiving the updated property values, the SMP updates the corresponding semantic metadata in the Entry resource property.

Next, the SMP registers the Entry to an SMR and passes the semantic metadata as the registration parameters. The SMP periodically renews the registration as long as the semantic metadata are still valid.

It is important to note that the semantic metadata are kept *fresh* by monitoring updates on the resource property values of the TW and that the semantic metadata are kept *valid* by implementing the WS-ResourceLifetime interfaces of the WSRF specifications³.

The procedure used by an SMR to manage semantic metadata is similar to that of the SMP. The difference is that instead of monitoring the resource property of a TW an SMR monitors the Entry resource property of SMPs. It is done by using the subscription/notification mechanism.

¹An SMP (also an SMR) implements interfaces defined by the ServiceGroup specifications. The specifications define the Entry resource property to store the member information.

²This interface is used to create and send a notification based on a defined topic.

³A TWM may delete an Entry by either sending a delete request or ceasing to adjust its lifetime.

4 Prototype System Implementation

We have implemented the proposed framework using the Aggregator Framework provided by GT4.0 [1]. In the implementation, each WS-Resource is classified into a specific domain according to its functions and an ontology, called the domain-specific ontology (DSO), is defined for each domain. The current prototype uses two ontologies: the *GLUE ontology* (Figure 2) for computing WS-Resources (e.g., GRAM services), which is based on the GLUE schema [9], and the *Database ontology* for database WS-Resources (e.g., OGSA-DAI services), which is based on the CIM-based Grid Schema [10]⁴. Each main class in the DSO has an (optional) object property *hasSemantics* that further describes the instances of the class. For example, the range of *hasSemantics* of class *glue:Processor* could be an instance of an ontology that describes the processor architecture and class.

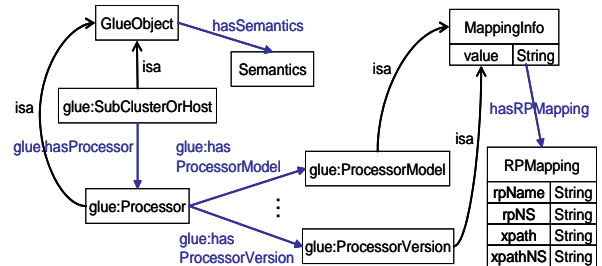


Figure 2. GLUE ontology

To update semantic metadata efficiently, each resource property value of a TW is associated with an instance of a specific class in the DSO, called the *MappingInfo* (see Figure 2). This class has a data property *value* to store the associated resource property value and an object property *hasRpMapping* to store the information on how to extract the value from the resource property. The range of *hasRpMapping* is an instance of class *RPMapping*, which has the following data properties: *rpName*, which stores the name of the resource property, *rpNS*, which stores the namespace of the resource property, *xpath*, which stores an XPath expression that specifies how to extract the value from the resource property, and *xpathNS*, which stores the namespace mapping used in the XPath expression. For example, resource property value `<glue:Processor glue:Model="Pentium IV"/>` is associated with an instance of class *glue:ProcessorModel* (which is a subclass of the class *MappingInfo*) with *value*= "Pentium IV" and *xpath* points the value of the *glue:Model* attribute.

⁴The class Profile of the OWL-S ontology is extended to include the DSOs.

Figure 3 shows the architecture of a cluster where we deployed the prototype system. It consists of 12 nodes and uses Torque [11] as a local scheduler. GT4.0 is installed in the main node and SMP and SMR services are deployed in the globus container. We also installed Ganglia [12] on the cluster so that the globus GRAM service provides detailed information about the nodes in the cluster.

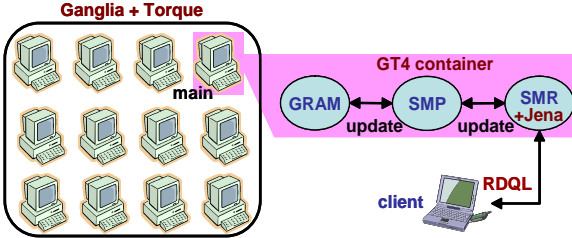


Figure 3. A testbed cluster

To enrich the GLUE ontology with useful information for resource discovery and monitoring we further define two ontologies: CPU and AIST ontology. The former describes the CPU architecture of instances of the class *Processor* while the latter describes the organizational unit of instances of the class *Host* of the DSO.

A TWM reads the resource property of the GRAM service, converts it into semantic metadata (i.e., ontology instances of the three ontologies). A monitoring client sends an RDQL query every 30 seconds to the SMR to monitor the CPU load of nodes belonging to GTRC unit that have SUN CPU architecture. The SMR processes the RDQL query by deriving additional assertions from the semantic metadata using the Jena Rule Reasoner [13] and extracting nodes that satisfy the given condition. Figure 4 shows the change of the CPU load of one of the matched nodes.

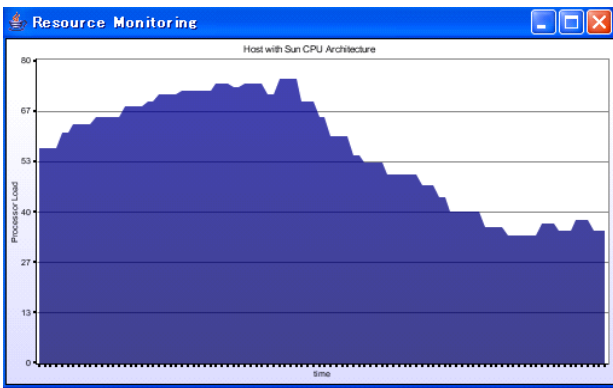


Figure 4. Monitoring result

5 Performance Evaluation

This section gives the performance analysis of the proposed framework. In particular, it describes the *TW update propagation time (TW time)* for various number of TWs and resource property update time intervals. The TW time is defined as the time required for propagating an update from a TW to the SMR. We also measure the *SMP update propagation time (SMP time)* which is the part of the SMR time. It is defined as the time required for propagating an update from an SMP to the SMR.

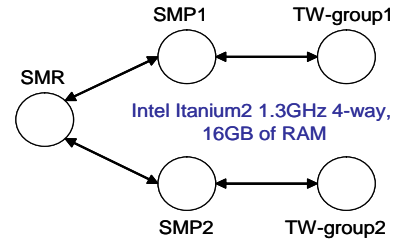


Figure 5. Evaluation testbed architecture

Figure 5 shows the architecture of the evaluation testbed. It consists of five identically configured nodes connected by the Myrinet networks: Intel Itanium2 1.3GHz 4-way with 16GB of RAM. One node is dedicated for SMR, two nodes for SMP and the rest two nodes for TWs. We divide TWs into two groups where each group is associated with an SMP (e.g., TW-group1 with SMP1). To balance the workload among the SMP and TW-group nodes, we evenly distribute TWs between the two TW-group nodes. For example, when we ran experiment with 100 TWs, we ran 50 TWs on each TW-group node.

A TW instance is created every one second. The calculation is started after all TW instances have been started up and stopped after each instance has performed 100 updates. The size of the semantic metadata of a TW is about 64KB. Thus, it is the sizes of update messages sent from the SMPs to the SMR.

Figure 6 shows the average of the TW time for various number of TWs. Each line shows the result when the resource property update time interval of the TWs is set to 0.5, 1 and 2 minutes. As expected, as the number of TWs increases, the TW time also increases. The TW time becomes larger as the update frequency increases (i.e., the update time interval decreases). We observe that, for 100 TWs or less and 150 TWs with 1 minute update time interval or more the average TW time is quite small (i.e., less than 4 seconds). However, for 150 TWs with a high update frequency (i.e., twice

every minute) the average TW time increases largely (about 50 seconds).

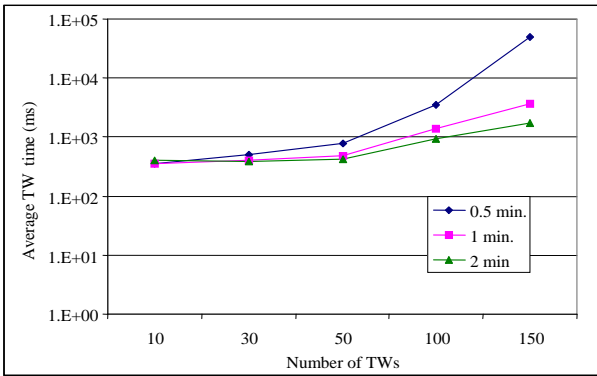


Figure 6. Average TW time for various number of TWs

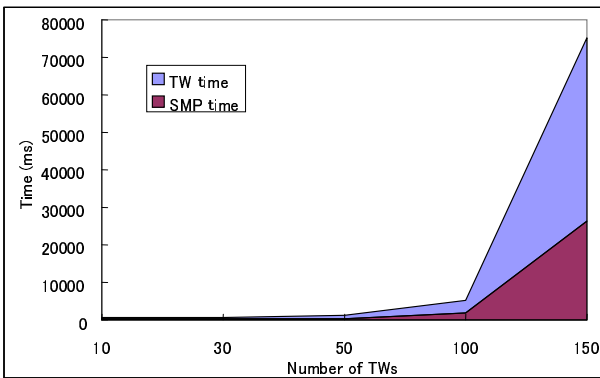


Figure 7. SMP time proportion (update time interval=0.5 min.)

Figure 7 shows the proportion of the SMP time in the TW time for update time interval 0.5 minute⁵. It takes about 50% of the time. This indicates that when a TW is also an SMP (i.e, its resource property is a semantic metadata) the TW time can be significantly decreased.

6 Conclusion

We have proposed a general information service framework to support automatic service discovery and monitoring in WSRF. We have also built a prototype system in a small grid cluster to see the practicability of the framework. From the evaluation results, the

⁵The results for update time interval 1 and 2 minutes are similar so we omit them.

framework can efficiently handle up to 100 TWs even though the resource property update frequency is high. For 150 TWs with a high update frequency, the average TW time greatly increases. This suggests us to provide a distributed architecture of SMRs to support a great number of TWs.

References

- [1] <http://www.globus.org/toolkit/>.
- [2] <http://www.globus.org/toolkit/docs/4.0/techpreview/ogsadai/>.
- [3] <http://www.globus.org/toolkit/docs/4.0/execution/>.
- [4] <http://www.daml.org/services/owl-s/1.1/>.
- [5] <http://www.globus.org/toolkit/docs/4.0/info/index/>.
- [6] <http://www.globus.org/toolkit/downloads/3.2.1/>.
- [7] <http://owlsm.projects.semwebcentral.org/>.
- [8] <http://owlseditor.semwebcentral.org/>.
- [9] <http://www.cnaf.infn.it/~sergio/datatag/glue/>.
- [10] <http://forge.gridforum.org/projects/cgs-wg>.
- [11] <http://www.clusterresources.com/>.
- [12] <http://ganglia.sourceforge.net/>.
- [13] <http://jena.sourceforge.net/>.
- [14] S. Miles et al. Personalised grid service discovery. In *Proc. of 19th Annual UK Performance Engineering Workshop*, pages 131–140, 2003.
- [15] A. Patil et al. METEOR-S web service annotation framework. In *Proc. of the World Wide Web Conference*, pages 553–562, 2004.
- [16] M. P. Said et al. Ontology-based grid index service for advanced resource discovery and monitoring. In *Advanced in Grid Computing - EGC 2005*, volume 3470 of *LNCS*, pages 144–153, 2005.
- [17] A. ShaikhAli et al. UDDle: An extended registry for web services. In *Proc. of SAINT'03 Workshop*, 2003.
- [18] H. Tangmunarunkit et al. Ontology-based resource matching in the grid—the grid meets the semantic web. In *Proc. of SemPG03*, volume 2870, pages 706–721, 2003.