Grid Metadata Catalog Service-Based OGC Web Registry Service

Peisheng Zhao, Aijun Chen, Yang Liu, Liping Di, Wenli Yang, Peichuan Li

Laboratory for Advanced Information Technology and Standards

School of Computational Science, George Mason University

Fairfax, Virginia 22030, USA

1-301-552-3829

{pzhao, chen6, yliu5, ldi, wyang1, pli2}@gmu.edu

ABSTRACT

Grid is a promising e-Science infrastructure that promotes and facilitates the sharing and collaboration in the use of distributed heterogeneous resources through Virtual Organization (VO). A critical factor to the overall utility of Grid is a scalable, flexible and robust registry mechanism. Although it provides some mechanisms to store and access metadata for publishing and discovering resources, such as MCS (Metadata Catalog Service), the Grid registry is inadequate for dealing with domain-specific resources. To enhance the earth science Grid systems, this paper presents a geospatial registry approach in which the OGC (Open GIS Consortium) WRS (Web Registry Service), a *de facto* standard that supports the publishing of and run-time access to geospatial resources, as a wrapper is used to extend the capabilities of the conventional Grid MCS to the processing of geospatial queries against multiple heterogeneous spatial data sources and services. The approach presented not only focuses on the specifics of descriptive information about spatial data, services, and relevant information objects, but also emphasizes using ontology to infer the semantic relationships between vocabularies for integrating different information models. The implementation of presented approach used in NASA Grid Data Service environment is also illustrated in this paper.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information filtering, Retrieval model, Search process

General Terms

Management, Design, Standardization, Languages.

Keywords

Grid, OGC, Catalog, ontology, OWL, semantic, information model.

GIS'04, Nov 12–13, 2004, Washington D. C., USA.

1. INTRODUCTION

Over the past several years, Grid technology has become an increasingly prominent basic information infrastructure that provides a series of services to support the sharing and coordinated use of diverse resources in the dynamic, distributed "virtual organizations"[9]. With its protocols and services concerned with communication and authentication, resource registry and negotiating access to multiple resources, there is now an opportunity to provide a large-scale collaboration e-science environment that spans many different projects, institutions and countries. The Globus Toolkit [8][9], an open-source set of services and libraries that implement key Grid protocols, has been widely adopted as the Grid technology solution for scientific and technical computing. In the field of earth sciences, the Committee on Earth Observation Satellites (CEOS) started a CEOS-Grid testbed [2] in 2002 for supporting diverse users worldwide in easily accessing earth observation geospatial information and aiding data providers in improving their operation efficiency. And the Earth System Grid (ESG) [7] is being developed to provide a seamless and powerful environment that enables the next generation of global climate research. With the development of Open Grid Services Architecture (OGSA) [10], NASA is developing and deploying the Information Power Grid (IPG) [12] with Earth Sciences Web Services Applications and Grid Data Services to provide an interoperable, flexible, and scalable sharing environment for the Earth Science modeling and analysis community.

Earth science is a data-intensive scientific domain in which the applications always produce and analyze a large volume of distributed heterogeneous geospatial information. The Grid system for earth science relies heavily on a metadata service to support publishing and discovery of geospatial and non-geospatial resources, such as service offers, interface definitions, dataset descriptions, application schemas, style descriptors and taxonomies. Metadata in catalogs represents resource characteristics that can be presented and queried for understanding and further processing by both humans and software. The Grid Metadata Catalog Service (MCS) [19] as a Grid middleware provides the ability to store and access metadata and to allow users to register and retrieve data items based on the attributes of data rather than data names in Grid. The MCS provides the mechanism to extend its schema for a specific scientific domain and implements policies regarding the consistency guarantees, authentication, authorization, and auditing capabilities. It also supports a logical name space that is independent of physical name space and allows the specification of a logical collection hierarchy and the notion of containers to aggregate small files. However, the MCS is concerned with only those schemes and interfaces that are at the general implementation level. There is no explicit scheme at the logic level for cataloging registered objects and indicating their relationships in the MCS. Consequently, it is very difficult to represent and interpret the complex date models and relationships in the MCS. For example, the MCS can not explicitly specify the "operate on" association between a dataset and its service, service binding and specification link. For interoperability, modeling a geometry object to describe geographic position may use GML (Geographic Markup Language), OGC (Open GIS Consortium) WKT (Well-Known Text) or some binary encoding. The information scheme of the MCS is insufficient to describe such objects and relevant standards and information models. Moreover, the MCS interfaces are proprietary and don't support complex conceptual transactions and queries for geospatial resources in the Grid Web service environment. Therefore, the MCS can not satisfy the demands of the earth science Grid system.

Currently OGC WRS (Web Registry Service) [16] as the part of the OGC Open Web Services initiative is becoming the *de facto* standard that supports the discovery of and binding to registered geospatial information resources within an information community. Its Registry Information Model (OGCRIM) provides a formal structure representing geospatial metadata resources and their interrelationships and a conceptual schema constraining the kinds of objects stored in the registry and how these registry objects and the relevant descriptive information are organized and interpreted. The WRS also defines a set of standard public interfaces for Web-based discovery and exploitation of geo-processing functions. Note that the WRS is open because it does not specify any implementation scheme. Therefore, user can extend it at the application level and implement it using any language on any platform and any system.

The analysis above illustrates that the WRS and the MCS are complementary approaches: the former defines the metadata representation at the conceptual level and the latter provides an implementation scheme for storage and access of metadata. Obviously, it is significant to integrate the WRS with the MCS to provide a basic interoperable catalog for publishing and discovering distribute heterogeneous geospatial information in the earth science Grid systems. In this paper, we propose a flexible and extensible approach that implements the WRS information model and interfaces with the MCS. This approach not only focuses on the specifics of descriptive information about spatial data, services, and relevant information objects, but also addresses using Ontology Web Language (OWL) to build semantic model for inferring semantic equivalence between terms in different information models.

The remainder of this paper is organized as follows. In section 2, we provide background information about the WRS and the MCS. In section 3, we present our approach for integrating different information model and computational model between the MCS and the WRS. In section 4, we describe our implementation of Grid MCS-based OGC WRS. In section 5, we summarize some related work. And finally, in section 6, we conclude our work and present some future research directions.

2. BACKGROUND

2.1 OGC WRS

A registry service plays a 'directory' role in the open, distributed systems: providers advertise the availability of their resources using metadata in a registry, and users can then query the metadata in the registry to discover interesting resources and determine how to interactively run-time access them. OGC WRS defines a Web-based common mechanism to classify, register, describe, search and access metadata about geospatial information.

2.1.1 Information Model

The OGC registry information model (OGCRIM) is based on the ebXML registry information model (ebRIM) [14]. This information model specifies formally how domain objects are organized, constrained and interpreted based on conceptual structure. A high-level view of the model with most of the classes appears in Figure 1.

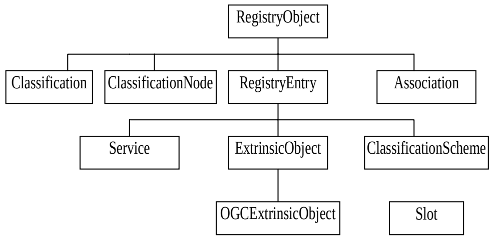

Figure 1. Part of OGCRIM Class Hierarchy.

The "*RegistryObject*" class at the top level is an abstract base class and provides minimal metadata for registry objects, such as name, object type, identifier and so on. The "*Slot*" instances provide a dynamic way to add arbitrary attributes to a registry object. The "*ClassificationScheme*" class defines a tree structure made up of "*ClassificationNode*"s to describe a structured way for classifying or categorizing "*RegistryObject*"s. An "*ExtrinsicObject*" provides required metadata about the content being submitted to the registry, thus allowing any type of object to be catalogued. And the "*OGCExtrinsicObject*" class adds the "*contentURL*" attribute in order to refer to the content stored in remote repositories outside of the registry. The "*Association*" class uses an "*associationType*" attribute to identify the relationship between a source "*RegistryObject*" and a target "*RegistryObject*". Figure 2 shows how a service is tightly-coupled with a dataset by specifying the value "*operatesOn*" for the "*associationType*" attribute.

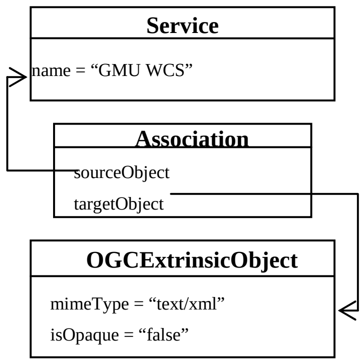

Figure 2. Association of "Service" and "Dataset".

2.1.2 Public Interfaces

The WRS defines two of its own Web-based interfaces, "*WRSQuery*" and "*WRSTransaction*" as figure 3 describes, to constraint on 'find', 'bind' and 'publish' registry objects at the geospatial conceptual level. Not only do the WRS interfaces provide the basic set of operations, such as add, delete, modify and query resource offers and type descriptions, but also provide a

```
<portType name="WRSQueryPortType">
  <operation name="getCapabilities">…… </operation>
  <operation name="describeType">……   </operation>
  <operation name="getRecord">
    <input message="wrs:GetRecordRequest" />
      <output message="wrs:GetRecordResponse" />
  </operation>
  <operation name="getResourceByID">…… </operation>
</portType>
<portType name="WRSTransactionPortType">
  <operation name="transaction">…… </operation>
  <operation name="lockRecord">……</operation>
  <operation name="registerResource">…… </operation>
</portType>
```

Figure 3. WSDL [20] definitions for "WRSQuery" and "WRSTransction".

number of specific capabilities, such as modify classification scheme and change registry object classification. The WRS adopts the OGC filter [17] syntax for expressing spatial query constraints in XML. This XML-encoded filter is a system neutral representation of a query predicate that can be easily validated, parsed and then transformed into whatever the target language is. For example, it could be transformed into a WHERE clause for a SQL SELECT statement to fetch data stored in a relational database, or an XPath or XPointer expression for fetching data from XML documents.

2.2 Grid MCS

The MCS provides a mechanism for publishing, discovering and accessing metadata that describes the creation, transformation, meaning and quality of data files or data items in Grid. It maintains the mappings between logical name attributes of data items and other descriptive metadata attributes and allows users to query based on data attributes rather than data names.

Currently, MCS contains exclusive logical files information, where the term "*logical file name*" (LFN) denotes a unique logical identifier for data content, while assuming that physical file metadata, which depends on the actual location of the file and the characteristics of a given storage system, is stored in Globus Replica Location Service (RLS) or elsewhere. The MCS provides management of logical collections of files for supporting authorization on groups of files, and supports logical views that consist of zero or more logical files, collections and/or other logical views for allowing more flexibility for users to group files according to their interests. In addition, the MCS provides authentication and authorization on the data items based on the Grid Security Infrastructure to enable authorized users to manipulate the MCS mappings and attributes. The MCS also manages auditing metadata to record actions performed by the metadata service, annotation metadata to describe logical data items and transformation history metadata to record the creation and transformations of a logical file. To maintain consistency among data items, the MCS provides a *"Master_Copy"* attribute to identify the physical location of a master copy of a file. In the new version of MCS, the simple spatial attributes including *"Point"*, *"LineString"* and *"Polygon"* and spatial relations including *"Mbrwithin"*, *"Mbrdisjoint"*, *"Mbrequal"*, *"Mbrintersects"* and *"Mbroverlaps"* are introduced to support spatial location queries against a minimum bounding rectangle. However, this support is still too general and simple to meet all of the requirements of geospatial community. For the purpose of supporting domain-specific metadata schemas, the MCS schema is extensible for accommodating new user-defined attributes. Users may create their own attributes by associating an attribute identifier from table 1 with a particular object and an attribute value from table 2.

The MCS assumes a file-based data model and provides a general low level of data scheme and interfaces for implementation coupled with replica service. Therefore, the MCS should be extended to support more complex data types and relationships for services, users and other resources and to provide an application level of common interfaces for the domain-specific applications in the service-oriented Grid environments.

Table 1. MCS_ATTRIBUTE schema.

Field Name	Type	Description
Id	Integer	Identifier for the attribute
Name	Varchar(50)	Name of the attribute
Attribute_typ	Varchar(20)	String/Integer/Float/Date/Time
Object_typ	Integer	0 -- logical file, 1 -- logical collection, 2 -- logical view

Table 2. MCS_**_ATTRIBUTE schema**

(**: string/integer/float/date/time).**

Field Name	Type	Description
Att_id	Integer	Attribute id
Obj_id	Integer	Object id to which the attribute belong to
Att_value	****	Value of the attribute

The MCS and the WRS are different: the WRS provides a geospatial conceptual level of information scheme and interfaces for registry and the MCS provides a general implementation level of metadata information scheme and interfaces. However, they provide complementary approaches which can enhance the current earth science Grid systems.

3. INTEGRATING the WRS with the MCS

From the descriptions presented in section 2, it is clear that there are differences between the WRS information model and the MCS data scheme. In this section we explain how to resolve these differences to integrate the WRS with the MCS using ontology, namely rendering the OGCRIM class hierarchy through the MCS data scheme by the OWL semantic model.

3.1 Semantic Model

The initial implementation of MCS assumes a file-based data model in which the most manipulated unit is a logical file, while OGCRIM assumes that any object can be registered. In order to support new objects, the meaning of "file" in the MCS should be extended to that of "*RegistryObject*" in ORCRIM, by which "file" means not only data file, but also service, classification, association and so on. And the table "*MCS_LOGICAL_FILE*" can store the metadata for all kinds of objects instead of just for data files. Thus "*RegistryObject*" is equal to "*MCS_LOGICAL_FILE*" semantically:

```
<owl:Class rdf:ID="MCS_LOGICAL_FILE">
  <owl:equivalentClass rdf:resource="RegistryObject"/>
</owl:Class>
```

Table 3. Mapping OGCRIM "RegistryObject" elements to MCS elements.

OGC RIM Element Path	MCS Element Path
RegistryObject/Name /LocalizedString/@value	MCS_LOGICAL_FILE /Logical_name
RegistryObject/@objectType	MCS_LOGICAL_FILE /Data_type
RegistryObject/@id	MCS_LOGICAL_FILE /Data_id
RegistryObject /AccessControlPolicy /Permission/@methodName	MCS_PERMISSIONS /Permissions
RegistryObject /AccessControlPolicy /Permission/Privilege /PrivilegeAttribute/@name	MCS_DATA_PERMISSIONS/Subject

Table 3 shows that the basic attributes of "*RegistryObject*" from OGCRIM have the elements corresponding to "*MCS_LOGICAL_FILE*" from MCS. The "*Name*" and "*ObjectType*" of "*RegistryObject*" can be mapped directly into the "*Logical_name*" and "*Data_type*" of "*MCS_LOGICAL_FILE*" directly:

```
<owl:DatatypeProperty rdf:ID="Logical_name">
  <rdfs:domain rdf:resource="#MCS_LOGICAL_FILE"/>
  <owl:equivalentProperty rdf:resource="#value"/>
 </owl:DatatypeProperty>
<owl:DatatypeProperty rdf:ID="Data_type">
  <rdfs:domain rdf:resource="#MCS_LOGICAL_FILE"/>
  <owl:equivalentProperty rdf:resource="#ObjectType"/>
 </owl:DatatypeProperty>
```

Since the "*AccessControlPolicy*" class, as one of the attributes of "*RegistryObject*", aggregates the user's access privilege and "*MCS_LOGICAL_ COLLECTION*" describes authorization on groups of files, the "*PrivilegeAttribute*" of "*AccessControlPolicy*" can not be mapped into the elements of "*RegistryObject*" directly. Hence the "*MCS_PERMISSIONS*" and the "*MCS_DATA_PERMISSIONS*" from the MCS are combined to accommodate the "*Role*", "*Identity*" and "*Group*" of "*PrivilegeAttribute*":

```
<owl:Class rdf:ID="Permission">
  <owl:unionOf rdf:parseType="Collection">
   <owl:Class rdf:about="#MCS_PERMISSIONS "/>
   <owl:Class rdf:about="MCS_DATA_PERMISSIONS"/>
  </owl:unionOf>
</owl:Class>
```

"*MCS_LOGICAL_FILE/Data_id*" could not be used to store "*RegistryObject/@id*" directly although they have same meaning, because the former is an integer and the latter is a UUID string. Note that MCS is easily and efficiently extensible to support new attributes. Table 4 shows the corresponding elements of "*RegistryObject/@id*" in MCS. Thus, "*MCS_Attributes*" and

Table 4. Mapping OGCRIM "RegistryObject@id" to MCS elements.

OGC RIM Element Path	MCS Element Path
RegistryObject/@id	MCS_Attributes/Name=["ID"]
	MCS_Attributes/AttributeType=["string"]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_String_Attributes/Att_value = [RegistryObject/@id]

"*MCS_String_ Attributes*" from MCS are linked to present "RegistryObject@id":

```
<owl:DatatypeProperty rdf:ID="id">
   <rdfs:domain rdf:resource="#RegistryObject"/>
</owl:DatatypeProperty>
 <owl:Class rdf:ID="MCS_ID">
  <owl:unionOf rdf:parseType="Collection">
  <owl:intersectionOf rdf:parseType="Collection">
   <owl:Class rdf:about="#MCS_ATTRIBUTTES "/>
      <owl:Restriction>
       <owl:onProperty rdf:resource="#Name" />
       <owl:hasValue rdf:resource="id" />
      </owl:Restriction>
      <owl:Restriction>
       <owl:onProperty rdf:resource="#AttributeType" />
       <owl:hasValue rdf:resource="string" />
      </owl:Restriction>
   </owl:Class>
  </owl:intersectionOf>
  <owl:intersectionOf>
   <owl:Class rdf:about="MCS_STRING_ATTRIBUTE"/>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#Obj_id" />
        <owl:allValuesFrom  rdf:resource="#Data_id" />
      </owl:Restriction>
      <owl:Restriction>
        <owl:onProperty rdf:resource="#Att_value" />
        <owl:allValuesFrom  rdf:resource="#id" />
      </owl:Restriction>
   </owl:Class>
  </intersectionOf>
 </unionOf>
</class>
```

Actually the link between "*MCS_Attributes*" and "*MCS_****_Attributes*" in table 5 performs as an OGCRIM "Slot" that provides a dynamic way to add user-defined attributes to a registry object (**** means here the basic data type, such as string, integer). A new class “*MCS_Attribute*” is used for representing such a kind of link.

Note that the relationship between “*Slot*” and “*MCS_Attribute*” is “*subClassOf*”, not “equivlantClass” since “*Slot*” must be “*MCS_Attribute*” and “*MCS_Attribute*” is not necessarily “*Slot*”:

```
<owl:Class rdf:ID="MCS_Attribute">
  <rdfs:subClassOf rdf:resource="#Slot" />
   …
 </owl:Class>
```

Table 5. Mapping OGCRIM “Slot” elements to MCS elements.

OGC RIM Element Path	MCS Element Path
Slot/ @name	MCS_Attributes/Name
Slot/@slotType	MCS_Attributes/AttributeType
Slot/RegistryObject/@id	MCS_String_Attributes/Att_value
Slot/ValueList@values	MCS_****_Attributes/Att_value

Moreover, this link can also be used to describe the subclasses of OGCRIM “*RegistryObject*”, that is, every attribute specific to the subclass is treated as a user-defined attribute in MCS and associated with its owner by subclass identifier. Table 6

Table 6. Mapping OGCRIM “Association” elements to MCS elements.

OGC RIM Element Path	MCS Element Path
Association /@associationType	MCS_Attributes/Name=[”associationType”]
	MCS_Attributes/AttributeType=[”string”]
	MCS_String_Attributes/ Obj_id = [Association@id]
	MCS_String_Attributes/Att_value = [Association/@associationType]
Association /sourceObject/@id	MCS_Attributes/Name=[”sourceObject”]
	MCS_Attributes/AttributeType=[”String”]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_String_Attributes/Att_value = [Association/sourceObject/@id]
Association/ targetObject/@id	MCS_Attributes/Name=[”targetObject”]
	MCS_Attributes/AttributeType=[”String”]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_String_Attributes/Att_value = [Association/targetObject/@id]

shows the relevant elements in MCS for the attributes of “*Association*”, a subclass of “*RegistryObject*”. Following is a semantic description about one of the attributes of “Association”, “*associationType*”:

```
<owl:Class rdf:ID="associationType">
 <rdfs:subClassOf rdf:resource="#MCS_Attribute" />
   <owl:Restriction>
     <owl:onProperty rdf:resource="#name" />
     <owl:hasValue rdf:resource="associationType" />
   </owl:Restriction>
  <owl:Restriction>
     <owl:onProperty rdf:resource="#AttributeType" />
     <owl:hasValue rdf:resource="string" />
   </owl:Restriction>
  <owl:Restriction>
     <owl:onProperty rdf:resource="#Obj_id" />
     <owl:allValuesFrom rdf:resource="&Data_id"/>
   </owl:Restriction>
  <owl:Restriction>
     <owl:onProperty rdf:resource="#Att_value" />
     <owl:allValuesFrom rdf:resource="&Assoication;id"/>
   </owl:Restriction>
 </rdfs:subClassOf>
</owl:Class>
```

Table 7. Mapping OGCRIM “Service” and “Classification” elements to MCS elements.

OGC RIM Element Path	MCS Element Path
Service /ServiceBinding /@id	MCS_Attributes/Name=[”ServiceBinding”]
	MCS_Attributes/AttributeType=[”String”]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_Integer_Attributes/Att_value = [Service/ServiceBinding/@id]
Service /ServiceBinding /SpecificationLink /@id	MCS_Attributes/Name=[”SpecificationLink”]
	MCS_Attributes/AttributeType=[”String”]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_String_Attributes/Att_value = [Service/ServiceBinding/SpecificationLink@id]
Classification /ClassifiedObject /@id	MCS_Attributes/Name=[”ClassifiedObject”]
	MCS_Attributes/AttributeType=[”String”]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_String_Attributes/Att_value = [Classification/ClassifiedObject/@id]
Classification /ClassificationNode /@id	MCS_Attributes/Name= [”ClassificationNode”]
	MCS_Attributes/AttributeType=[”String”]
	MCS_String_Attributes/ Obj_id = [MCS_LOGICAL_FILE/Data_id]
	MCS_Sring_Attributes/Att_value = [Classification/ClassificationNode/@id]

Likewise, consider the subclass “*Service*” which not only records its own metadata, but also associate itself with “Classification” for type query and “*ServiceBinding*” for run-time accessing. Table 7 indicates how a “Service” and its associations are related to the elements in the MCS. A “*ServiceBinding*” may have a

"*SpecificaionLink*" instance to describe how to access the service using a technical specification in form of a WSDL document or another. And the "*ClassificationNode*" within its classification scheme are applied to the "*ClassifiedObject*" in the "*Classification*". A part of semantic description of the attributes of "*Service*" and "*Classification*" is as follows:

```
<owl:Class rdf:ID="serviceBinding">
  <rdfs:subClassOf rdf:resource="#MCS_Attribute" />
    <owl:Restriction>
       <owl:onProperty rdf:resource="#Att_value" />
     <owl:allValuesFrom rdf:resource="&ServiceBinding;id"/>
     </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="specificationLink">
  <rdfs:subClassOf rdf:resource="#MCS_Attribute" />
    <owl:Restriction>
       <owl:onProperty rdf:resource="#Att_value" />
     <owl:allValuesFrom rdf:resource="&SpecificationLink;id"/>
     </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="classifiedObject">
  <rdfs:subClassOf rdf:resource="#MCS_Attribute" />
    <owl:Restriction>
       <owl:onProperty rdf:resource="#Att_value" />
     <owl:allValuesFrom rdf:resource="&Service;id"/>
     </owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
<owl:Class rdf:ID="classificationNode">
  <rdfs:subClassOf rdf:resource="#MCS_Attribute" />
    <owl:Restriction>
       <owl:onProperty rdf:resource="#Att_value" />
     <owl:allValuesFrom rdf:resource="&ClassificationNode;id"/>
</owl:Restriction>
</rdfs:subClassOf>
  </owl:Class>
```

```
<Service id="urn:uuid:39a8afb6-0773-425d-9ced-c9e2a6c0e17f" objectType="Service">
<Name><rim:LocalizedString rim:lang="en" value="WCS provider" /></Name>
<ServiceBinding id="urn:uuid:4fea22c7-2c34-41dc-a74b-f215aaac32e4" objectType="ServiceBinding"
accessURI="http://laits.gmu.edu/cgi-bin/NWGISS/WCS">
 <SpecificationLink id="urn:uuid:0618d7cc-ec33-4a98-8763-912c0fa42f6f" objectType="SpecificationLink">
<UsageDescription> <rim:LocalizedString rim:lang="en" value="Query operations bound to HTTP GET and/or
HTTP POST and/or SOAP." />
</UsageDescription> ......
```

Id	Name	Attribute_type	…
101	Service	String	
102	ServiceBinding	String	
103	accessURI	String	
104	SpecificationLink	String	
105	UsageDescription	String	

MCS_ATTRIBUTES

Data_id	Logical_name	Data_type	…
10001	WCS provider	Service	
10002		ServiceBinding	
10003		SpecificationLink	

MCS_LOGICAL_FILE

Att_id	Obj_id	Att_Value
101	10001	39a8afb6-0773-425d-9ced-c9e2a6c0e17f
102	10002	4fea22c7-2c34-41dc-a74b-f215aaac32e4
104	10003	0618d7cc-ec33-4a98-8763-912c0fa42f6f
102	10001	4fea22c7-2c34-41dc-a74b-f215aaac32e4
103	10002	http://laits.gmu.edu/cgi-bin/NWGISS/WCS
104	10001	0618d7cc-ec33-4a98-8763-912c0fa42f6f
105	10003	Query operations bound to HTTP GET and/or HTTP POST and/or SOAP

MCS_STRING_ATTRIBUTES

Figure 4. Storing OGCRIM "Service" instance into MCS tables.

3.2 Semantic Inference

Once obtaining the semantic model as describe above, an inference engine is used to derive the mapping facts which are entailed from some base OWL and instance data together with other optional ontology information and the axioms and rules associated with the inference engine.

The figure 4 illustrates how an instance of the "Service" class appears in the WRS and its mapping in the MCS. From the semantic model, we can derive the facts as follows:

(gcor:Service owl:sameAs gcor:MCS_LOGICAL_FILE)

(gcor:ServiceBinding owl:sameAs gcor:MCS_LOGICAL_FILE)

(SpecificationLink owl:sameAs gcor:MCS_LOGICAL_FILE)

……

(gcor:MCS_ATTRIBUTES gcor:Name gcor:Service)

(gcor:MCS_ATTRIBUTES gcor:Name gcor:ServiceBinding)

(gcor:MCS_ATTRIBUTES gcor:Name gcor:SpecificationLink)

(gcor:MCS_STRING_ATTRIBUTES gcor:Att_value gcor: 39a8afb6-0773-425d-9ced-c9e2a6c0e17f*)*

(gcor:MCS_STRING_ATTRIBUTES gcor:Att_value gcor: 4fea22c7-2c34-41dc-a74b-f215aaac32e4*)*

(gcor:MCS_STRING_ATTRIBUTES gcor:Att_value gcor: 0618d7cc-ec33-4a98-8763-912c0fa42f6f*)*

……

Note that these facts include inferences based on subclass inheritance (being a *"gcor:Service"* implies it is an *"gcor:RegistryObject"* which is equivalent to "gcor:*MCS_LOGICAL_FILE*"; likewise with "gcor:*ServiceBinding*" and "gcor:*SpecificationLink*") and the property restriction ("*gcor:Att_value*" derives from the specific "*AllValuesFrom*" restriction).

4. IMPLEMENTATION

Figure 5 presents the architecture of MCS-based WRS as it is currently implemented in our testbed system. Figure 6 shows the Web interface for *"getRecord"* operation in this prototype (http://llinux.laits.gmu.edu:8080/WRS/index.html). The current implementation uses Globus Toolkit 2.2 as the basic infrastructure, an Apache Web server front and a MySQL database backend. All of the applications are written in Java.

"WRS Agent" is a core component that plays the role similar to a wrapper between the WRS and the MCS. Its *"OGCRIM Object Generator"* is invoked by WRS message to generate necessary OGCRIM class instances. And then these instances are mapped by its *"WRS/MCS Mapping Engine"* into MCS data scheme based on the semantic model described in section 3. The previous WRS message thus is translated into a MCS command and a RLS command. Conversely, the *"WRS/MCS" Mapping Engine* receives a response from MCS and RLS to map it into OGCRIM scheme and the *"OGCRIM Object Generator"* uses this scheme to generate relevant objects. So the MCS/RLS response becomes the WRS response.

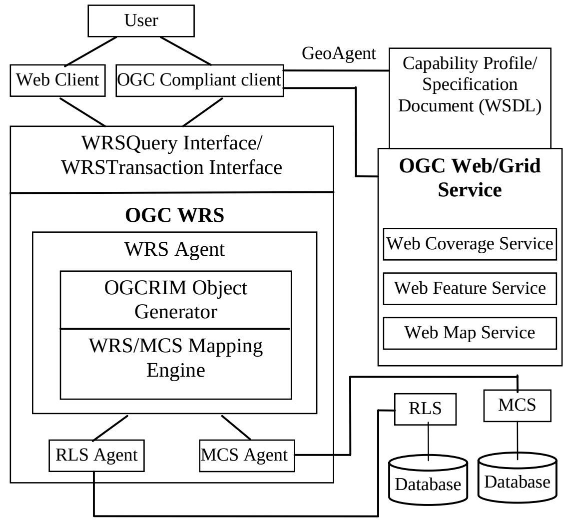

Figure 5. Architecture of the implementation.

Every OGC Web service has its own capability profile containing the metadata about the service and the service provider and describing which operations the service provides and the data layers it serves. The *"Geo-Agent"* component is responsible for checking the available capability profiles and using them to register the resources they represent through the *"WRSTransaction Interface"*. It also supports to check and parse other specification documents, such as WSDL, for publishing.

Figure 6. Web Interface for *"getRecord"* Operation.

Following is a simple scenario for discovery and access of service and data, which uses WRS, MCS and RLS.

1) The *"Client"* queries the *"OGC WRS"* using a WRS message through *"WRSQuery Interface"*.
2) The *"OGC WRS"* translate the WRS query into MCS query using *"WRS Agent"*.

3) The MCS query is sent by "*MCS Agent"* to the "*MCS Server"* for querying.
4) The "*MCS Server"* responds with a list of logical name attributes for query items with matching attributes.
5) The "OGC WRS" queries "RLS Server" using "RLS Agent".
6) The "*RLS Server"* returns a list of physical name attributes for the query items identified by the logical name.
7) The "*OGC WRS"* translate the results into a WRS message.

5. RELATED WORK

The MCS initially is file-data oriented, that is, it records the file metadata for data discovering. It is based on MCAT Metadata Catalog [13] of Storage Resource Broker (SRB) [1] from San Diego Supercomputing Center. Both of them support logical name space, logical collection and container, and GSI authentication, but the MCAT stores both logical and physical metadata and can be used as a stand-alone component, while the MCS is one of component in a layered, composed Grid architecture. In [19], it gives out some MCS application experiences on ESG. Replica Metadata Catalog developed by European DataGrid's Reptor project [11] has similar design and function as the MCS.

The WRS owes a great deal to the ebXML model. Currently, prominent models within the Web services realm include the ebXML and the UDDI [15] model. The API (Application Program Interface) associated with both models support multiple query patterns: browse and drill-down, or filtered queries against specified registry objects. The UDDI model focuses more on business entities and associated service descriptions. An extended UDDI registry, which allows to record user-defined attributes about service, is described in [18]. The ebRIM, which draws on the ISO 11179 set of standards to provide comprehensive facilities for managing metadata, is more general and extensible. How to exploit the class hierarchies in ebXML registries at the semantic level for efficient service discovery and composition is reported in [5]. In [6], there is a description of how OWL ontologies stored and accessed through the ebXML classification hierarchy are used by software agents for automatic service discovery and composition. The WRS extends the capabilities of the ebXML service to address the relationship between service and data explicitly.

In [4], how to integrate Grid technology with OGC Web services for NASA EOS data is described. In this paper, we combine the merits of the MCS and the WRS to build a MCS-based WRS for publishing and discovering OGC Web service and data in Grid through standard Web interfaces.

6. CONCLUSION

To be able to exploit services and relevant data in Grid, catalog middleware should contain not only metadata information about individual item but also relationship information to indicate the association between registry objects. Therefore a well-defined registry information model plays a very important role in Grid catalog middleware. In this paper, we have presented how a MCS is augmented with the OGCRIM to support cataloguing geospatial service and data in Grid and described the design and implementation of a MCS-based WRS. What distinguishes our work is that it makes the MCS geo-enabled and object-oriented by mapping the OGCRIM into the MCS data scheme and implementing the WRS interfaces in the MCS. And For the purpose of supporting flexible semantic matching between different information models, we are investigating how the WRS and the MCS can be enhanced by using ontologies.

In the next step we will implement a semantic OGCRIM based on DAML-S [3] ontologies and geospatial domain ontologies. Here DAML-S ontologies modulate the structure of registry object and indicate the relationships between registry objects, and domain ontologies play the role as meta-ontologies about DAML-S ontologies for indicating the relationships between the terms used in DAML-S ontologies. By using ontologies, the matching process can perform inference on the subsumption hierarchy to get the recognition of semantic matches regardless of syntactic differences. However, how to build domain ontologies remains a significant problem for us because of the complexity of geospatial information. Currently, there are three sets of metadata standard about geospatial information, respectively from ISO19115, FGDC and ECS. They are being widely used to describe geospatial information. Therefore, the terms in these standards are the best candidates to be used to build domain ontologies.

7. ACKNOWLEDGMENTS

This research project is supported by grants from NASA Earth Science Technology Office (ESTO), NASA Earth Science Data and Information System (ESDIS) project, and Open GIS Consortium (OGC).

8. REFERENCES

[1] Baru, C., Moore, R., Rajasekar, A., and Wan, M., The SDSC Storage Resource Broker", presented at *Proc. CASCON'98 Conference*, 1998.

[2] CEOS-Grid Task Team, V1.0 CEOS-Grid Interim Task Team Project Plan, Sep. 20, 2002

[3] DAML Services Coalition. DAML-S: Web Service Description for the Semantic Web. *In The First International Semantic Web Conference (ISWC)*, June 9-12, 2002, Sarinia, Italy.

[4] Di, L., Chen, A., Yang, W., Zhao, P., The Integration of Grid Technology with OGC Web Services (OWS) in NWGISS for *NASA EOS Data. HPDC12 & GGF8* on June 24 - 27, 2003, Seattle, USA.

[5] Dogac, A., Kabak, Y., Laleci, G., A Semantic-Based Web Service Composition Facility for ebXML Registries. *The 9th International conference of Concurrent Enterprising*, Espoo, Finland, June 2003

[6] Dogac, A., Kabak, Y., Laleci, G., Enriching ebXML Registries with OWL Ontologies for Efficient Service Discovery. *The 14th International Workshop on Research Issues on Data Engineering.* Boston, USA , March 28-29, 2004

[7] ESG, "The Earth Systems Grid". http://www.earthsystemgrid.org

[8] Foster, I., Kesselman, C. Globus: A Toolkit-Based Grid Architecture. In Foster, I. and Kesselman, C. eds. The Grid:

Blueprint for a New Computing Infrastructure, Morgan Kaufmann, 1999, 259-278

[9] Foster, I., Kesselman, C. and Tuecke, S. The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, 15(3). 200-222. 2001. www.globus.org/researcch/papers/anatomy.pdf

[10] Foster, I., Kesselman, C., Nick, J. and Tuecke, S. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. http://www.globus.org/ogsa/, 2202a

[11] Guy, L., Kunszt, P., Laure, E., Stockinger, H., Stockinger, K., Replica Management in Data Grids. Global Grid Forum 5.

[12] IPG, "NASA Power Information Grid". http://www.ipg.nasa.gov

[13] MCAT, MCAT- A Mtea Information Catalog (Version 1.1).

[14] OASIS/ebXML Registry Technical Committee. OASIS/ebXML Registry Information Model v2.0 Approved Committee Specification. http://www.oasis-open.org/committees/regrep/documents/2.0/specs/ebRIM.pdf

[15] OASIS UDDI Specifications TC – Committee. UDDI Version 3.0.1 UDDI Spec Technical Committee Specification. http://uddi.org/pubs/uddi-v3.0.1-20031014.pdf

[16] Open GIS Consortium Inc. OWS1 Registry Service. 2002-07-26

[17] Open GIS Consortium Inc. Filter Encoding Implementation Specification Document. 2001-12-19

[18] Shaikhali, A., Rana, O., Al-Ali, R., Walker, D., "UDDIe: An Extended Registry for Web Services". *Workhshop on Service Oriented Computing: Models, Architectures and Applications* at SAINT Conference, Florida, January 2003

[19] Singh, G., Bharathi, S., Chervenak, A., Deelman, E., Kesselman, C., Manohar, M., Patil, S. and Pearlman. L. A Metadata Catalog Service for Data Intensive Applications. *SC'03*, November 15-21, 2003, Phoenix, Arizona, USA

[20] Web Service Description Language(WSDL). http://www.w3.org/TR/wsdl