Building a distributed digital library for natural disasters metadata with grid services and RDF

Wei Xing and Marios D. Dikaiakos
Department of Computer Science, University of Cyprus, Nicosia, Cyprus
Hua Yang
Xian Institute of Post and Telecommunication, Nicosia, Cyprus, and
Angelos Sphyris and George Eftichidis
Algosystems SA, Kalithea, Greece

Abstract

Purpose – This paper aims to describe the main challenges of identifying and accessing useful information and knowledge about natural hazards and disasters results. The paper presents a grid-based digital library system designed to address the challenges.

Design/methodology/approach – The need to organize and publish metadata about European research results in the field of natural disasters has been met with the help of two innovative technologies: the Open Grid Service Architecture (OGSA) and the Resource Description Framework (RDF). OGSA provides a common platform for sharing distributed metadata securely. RDF facilitates the creation and exchange of metadata.

Findings – Using grid technology allows the RDF metadata of European research results in the field of natural disasters to be shared securely and effectively in a heterogeneous network environment.

Originality/value – A metadata approach is proposed for the extraction of the metadata, and their distribution to third parties in batch, and their sharing with other applications can be a quickly process. Furthermore, a method is set out to describe metadata in a common and open format, which can become a widely accepted standard; the existence of a common standard enables the metadata storage in different platforms while supporting the capability of distributed queries across different metadata databases, the integration of metadata extracted from different sources, etc. It can be used for the general-purpose search engines.

Keywords Digital libraries, Data analysis, Natural disasters

Paper type Research paper

Introduction

Research in natural hazards focuses on unraveling and understanding processes, comprehensive risk assessment, forecasting and risk management and mitigation. Advances have been made in seismic research, forest fires, landslides, floods, volcanic hazards, avalanches and technological hazards, particularly with the development of improved models and technologies for hazard forecasting, risk assessment and mitigation. Research projects focusing on natural hazards and disasters produce results in the form of explicit or tacit knowledge represented by reports, project deliverables, data-sets derived from field work, interesting training and dissemination material, etc. These artifacts are usually published and described on web sites

Library Management
Vol. 26 No. 4/5, 2005
pp. 230-245
© Emerald Group Publishing Limited
0143-5124
DOI 10.1108/01435120510596080

maintained by project partners during the duration of the respective projects. Following project completion, however, project teams dissolve and web-site maintenance and support gradually fade out. Hence, general-purpose search engines are used to search for past-project results. Nevertheless, search-engine query results provide large numbers of unrelated links. Furthermore, hyperlinks pointing to potentially useful material do not come with references or additional links to adequate information describing the "value" of identified resources. Consequently, identifying and accessing useful information and knowledge becomes very difficult. Effectively, valuable knowledge is lost and it is practically impossible to find and take advantage of it.

To address this problem, the Directorate General for Research of the European Union undertook the initiative to establish the European Mediterranean Disaster Information Network (EU-MEDIN, 2003). EU-MEDIN's goal is to foster coordinated and increased access to data and expert know-how before, during, and after a disaster strikes. The availability of reliable and timely information could contribute to our knowledge for reducing impacts of hazards and risks and bring about improved disaster preparedness in Europe in the near future. As the first step for the deployment of EU-MEDIN, the EU commissioned Algosystems SA with the development of a thematic web portal to support the storage and retrieval of metadata pertaining to results of research in natural disasters. Project-related metadata can be inserted via a Web interface to a back-end database (EU-MEDIN, 2003). Interested researchers can use the EU-MEDIN portal to query the database and search for project-artifacts.

This approach, however, encodes and maintains the metadata in the platform-specific format of the particular database system chosen for the development of the EU-MEDIN portal. Therefore, the extraction of the metadata, their distribution to third parties in batch, and their sharing with other applications can be a lengthy process. Furthermore, there is a need to describe metadata in a common and open format, which can become a widely accepted standard; the existence of a common standard enables the metadata storage in different platforms while supporting the capability of distributed queries across different metadata databases, the integration of metadata extracted from different sources, etc.

In this paper, we present gDisDL, a grid-based digital library system designed to address some of the problems mentioned above. Our approach comprises:

- A schema for describing project-related metadata in a platform-independent form, using the Resource Description Framework (RDF). RDF is a general framework for describing metadata of Internet resources and for processing this metadata; it is a standard of World Wide Web Consortium (W3C). RDF supports the interoperability between applications that exchange machine-understandable information on the web.
- A digital library system enabling the collection and storage of RDF-encoded metadata in distributed repositories, and the retrieval thereof from remote sites. This library is implemented as a Grid-service architecture comprised of a set of grid services, which allow the storage, management, and query of RDF metadata in a secure and distributed manner. To develop the library we use the Globus Toolkit 3 (Sotomayor, 2003) for programming grid services and the Jena toolkit (JENA, 2003) for handling RDF data.

- A set of graphical-user interfaces developed in Java to enable authorized end-users to create RDF metadata for natural-disaster research artifacts and to conduct keyword-based searches in RDF repositories.

The remainder of this paper is organized as follows. In the second section, we give a short overview of the technologies that we adopted to design and build gDisDL: in particular, the RDF, the Jena RDF toolkit, and the Open Grid Service Architecture (OGSA). In the third section, we present an overview of the EU-MEDIN RDF schema. The design challenges and architecture of the gDisDL system are presented in the fourth section. In the fifth section, we give an overview of some interesting implementation issues. The final section concludes our paper.

Background

RDF

RDF is a language used mainly for representing information about resources on the world wide web (Manola and Miller, 2003). In particular, it is intended for representing metadata about documents or other entities (e.g, web resources, software, publications, reports, image files etc), such as title, author, modification date, copyright, and licensing information. Although originally intended to be used on web resources, RDF is capable of representing information about things that can be identified on the Web, even when they cannot be directly retrieved from the web (Manola and Miller, 2003). This capability makes RDF an appropriate metadata schema language for describing information related to the various results and outcomes of natural-disaster research. RDF is designed for situations in which information needs to be processed by applications, rather than only being displayed to people. RDF provides a common framework for expressing information and can thus be exchanged between applications without loss of meaning. Since it is a common framework, application designers can leverage the availability of common RDF parsers and processing tools. The ability to exchange information between different applications means that the information can be made available to applications other than those for which it was originally created. The way, RDF is able to describe various resources (not restricting the user to the description of web resources) and represent different kinds of information into one unique data model, makes it a good candidate for our purpose, since the data that the users will be dealing with, can be of any format or type.

Jena

Jena is a Java toolkit for building Semantic Web applications (JENA, 2003). The Jena Java RDF API is developed by HP Labs for creating and manipulating RDF metadata. It comprises:

- "Another RDF Parser" (ARP), a streaming parser suitable for validating the syntax of large RDF documents. ARP complies with the latest recommendations of the RDF Core WG, as well as with standards and recommendations, such as XML (Bray *et al.*, 2004), DAML (Connolly *et al.*, 2001), and URI (Berners-Lee *et al.*, 1998).
- A persistence subsystem, which provides persistence for RDF metadata through the use of a back-end database engine. It supports RDQL queries.

- The RDF query language (RDQL), which is an implementation of an SQL-like query language for RDF. Jena generates RDQL queries dynamically and executes RDQL queries over RDF data stored in the relational persistence store.
- A reasoning subsystem, which includes a generic rule-based inference engine, together with configured rule sets for RDFS and for the OWL/Lite (Patel-Schneider *et al.*, 2004).
- An ontology API, which supports OWL, DAML + OIL and RDFS.

Open Grid Service Architecture

The grid supports the sharing and coordinated use of diverse resources in dynamic, distributed "virtual organizations" (VOs) (Foster *et al.*, 2001). The Open Grid Services Architecture (OGSA) is a service-oriented grid computing architecture, which is an extensible set of grid services that may be aggregated in various ways to meet the needs of VOs (Foster *et al.*, 2002). OGSA defines uniform grid service semantics and standard mechanisms for creating, naming, and discovering grid services. Web service technologies, such as XML, SOAP, WSDL, UDDI, etc., are adopted to build up the grid services infrastructure. A grid service is a web service that provides a set of well-defined interfaces and follows specific conventions (Foster *et al.*, 2001). The interface and behaviors of all grid services are described by the Grid Web Service Description Language (GWSDL) (Sotomayor, 2003). Furthermore, the Open Grid Service Infrastructure (OGSI) gives a formal and technical specification of the Grid service, and the Globus Toolkit 3 (GT3) offers a programming environment for implementing Grid services (Tuecke, 2002).

Metadata elicitation

As mentioned earlier, the goal of the gDisDL system is to support the storage and retrieval of metadata (i.e., structured data about data) that pertain to a variety of results derived from research in natural disasters, such as earthquakes, floods, forest fires, industrial hazards, landslides, and volcano eruptions. To this end, we need a metadata schema that would provide a useful, adequate representation of project "resources" The term "resource" here is used to collectively refer to results of projects, as well as projects themselves. Furthermore, we need to define this schema in a common and open format, which will:

- promote the standardization of natural disaster metadata, while at the same time allowing future extensions;
- enable the storage of metadata in different platforms according to this common, standard schema; and
- support the interoperability of different metadata repositories; in particular the specification of queries and the execution thereof upon different metadata databases.

To develop such as schema, the Coordinator and Steering-group members of the EU-MEDIN project (EU-MEDIN, 2003) undertook the effort of reviewing several projects in natural-disaster research with the aim of describing each type of resource by means of an EU-MEDIN metadata schema. One of the first steps in that effort was the identification of the entire set of possible types of results that could arise during the course of a project, and for which it would be interesting to create respective metadata records in a database. Currently, the following types of resource are included in the EU-MEDIN metadata collection: project, report/deliverable, journal paper, other scientific paper, student dissertation, press article, media presentation, book, event, hardware, software, web site, spatial digital dataset, experimental dataset (laboratory), experimental dataset (field), and unclassified activity.

An important element of this schema is the set of subjects of a resource. The subjects of a resource provide a classification for the resource, which is understandable quickly and easily, in very much the same way as the words drama/historical would indicate the nature of a film or a TV program in a printed listing. For each and every resource in the database, a value is stored for its set of subjects. The subjects of a resource are in effect, two-dimensional, that is, they are composed of two values. The first of these is termed "primary subject" whereas the second is termed "secondary subject". In this setting, the primary subject indicates a type of risk with which the resource is associated (e.g. volcanoes, earthquakes), whereas the secondary subject indicates an aspect of such a risk. The possible values for the primary and secondary subjects were derived through a consensus process involving the EU-MEDIN project Steering Group and officers at the Directorate General of Research of the European Commission. The primary subjects of a resource are drawn from the following set of values: "Forest Fires", "Desertification", "Droughts", "Floods", "Storms", "Avalanches", "Landslides", "Earthquakes", "Seismic Engineering", "Volcanoes", "Industrial Hazards", "Other Risk". The secondary subjects of a resource are drawn from the set: "Hazard Assessment", "Forecasting and Monitoring", "Modeling and GIS", "Earth Observation/RS", "Vulnerability Assessment", "Risk Analysis and Mapping", "Damage Assessment/Restoration", "Management/Mitigation", "Crisis Management/Intervention", and "Other Aspect". The primary and secondary subjects value for a resource may be any series of ordered pairs of values drawn from the above two sets. If a user selects a pair containing "Other Risk" or "Other Aspect," then a mechanism should be provided for them to specify just what this "other" selection signifies.

We adopted the RDF Schema (Brickley and Guha, 2000) to describe the identified set of resources classes, properties, and values for the EU-MEDIN resources. Following the classification of distinct resources identified in the EU-MEDIN project, we introduced an RDF Schema with 16 classes ("EC project", "Report/deliverable", "Journal paper", etc), and defined their properties and the relationships between class instances accordingly. The resulting schema is named EU-MEDIN RDF schema and represents our proposed metadata standard for natural-disaster research resources. This schema was based on two already existing schemas that have been established through other initiatives: the Dublin Core Metadata Element Set (DCMES) (Dublin Core, 2003) and the Federal Geographic Data Commission Content Standard for Digital Geospatial Metadata (FGDC-CSDGM). The EU-MEDIN schema relies more on DCMES than it does on FGDC-CSDGM. Figure 1 shows the class hierarchy of the EU-MEDIN RDF schema.

Below, we give an example extracted from the EU-MEDIN RDF schema. This excerpt of our schema includes two classes, Press article, and EC Project, and three properties, author, belongTo and name. Using these classes and properties, we can

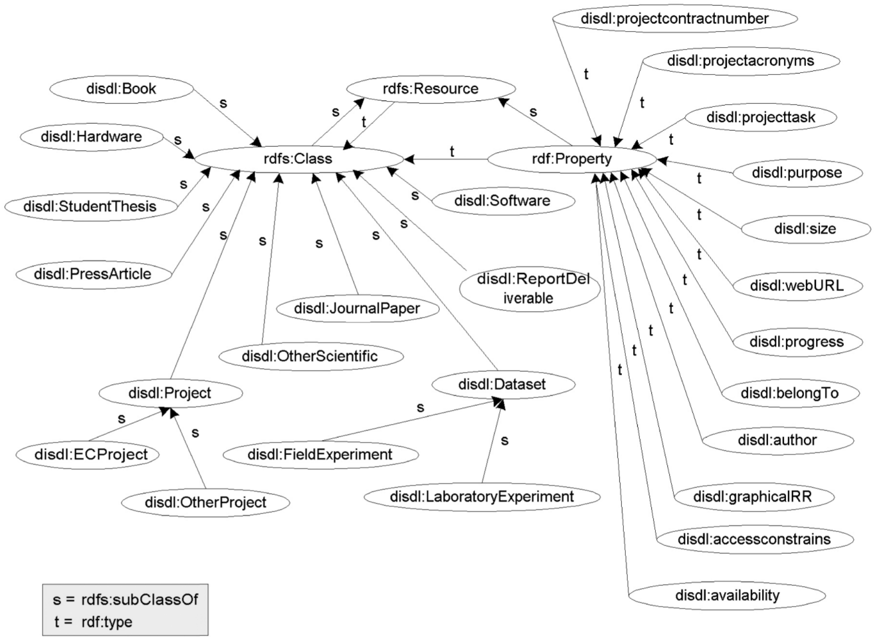


Figure 1.
Class hierarchy for the EU-MEDIN RDF schema

describe the following piece of knowledge in RDF: "John Smith wrote a Paper, which belongs to FloodSim project. The paper's title is 'Flood Simulation on the Grid'". Figure 2 shows part of the schema definition and the RDF description of the example, presented as an RDF graph. The RDF data can also be represented and stored physically as a set of RDF triples in N3 (Berners-Lee, 2000):

```
@prefix:<http://www.eu-medin.com/publication/floodG# >
@prefix dlib:<http://www.eu-medin.com/2003/02/schema# >
@prefix rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns# >
```

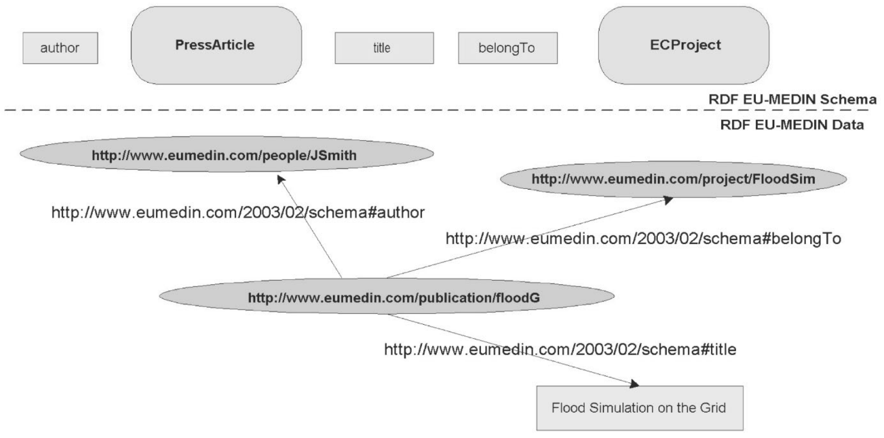

author
PressArticle
title
belongTo
ECProject
RDF EU-MEDIN Schema
RDF EU-MEDIN Data
http://www.eumedin.com/people/JSmith
http://www.eumedin.com/project/FloodSim
http://www.eumedin.com/2003/02/schema#author
http://www.eumedin.com/2003/02/schema#belongTo
http://www.eumedin.com/publication/floodG
http://www.eumedin.com/2003/02/schema#title
Flood Simulation on the Grid

Figure 2.
Example of an EU-MEDIN RDF schema

```
:floodG rdf:type dlib:publication;
:floodG dlib:author "JSmith";
:floodG dlib:belongTo
:<http://www.eumedin.com/project/FloodSim> ;

:floodG dlib:title "Flood Simulation on the Grid".
```

gDisDL system design

gDisDL system is a grid service-oriented system designed to collect, process, store, and query the RDF metadata encoded according to the RDF EU-MEDIN schema. As shown in Figure 3, the gDisDL system is comprised of a number of geographically distributed gDisDL nodes and a UDDI registry. Each node consists of a Data Aggregator and a Searcher. The Data Aggregator collects, validates, and encodes metadata in RDF; the Searcher is designed for querying RDF metadata. The functions of the components of the gDisDL system will be provided through two gDisDL grid services: a Data Aggregator grid service and a Searcher grid service. The UDDI server is used for publishing and discovering information about the gDisDL grid services (UDDI version 3.02). The Grid Authentication service is a credential service of the Grid Security Infrastructure (GSI) to protect gDisDL grid services invoking by unauthorized clients (Foster *et al.*, 1998).

Finally, the Editor and the Searcher GUI are grid service clients for the Data Aggregator and the Searcher grid services, enabling users to interact easily with gDisDL through a graphical-user interface.

Design goals

The main design issue of the gDisDL system is that the distributed RDF metadata should be shared efficiently and securely in an *ad-hoc*, dynamic basis. To address this

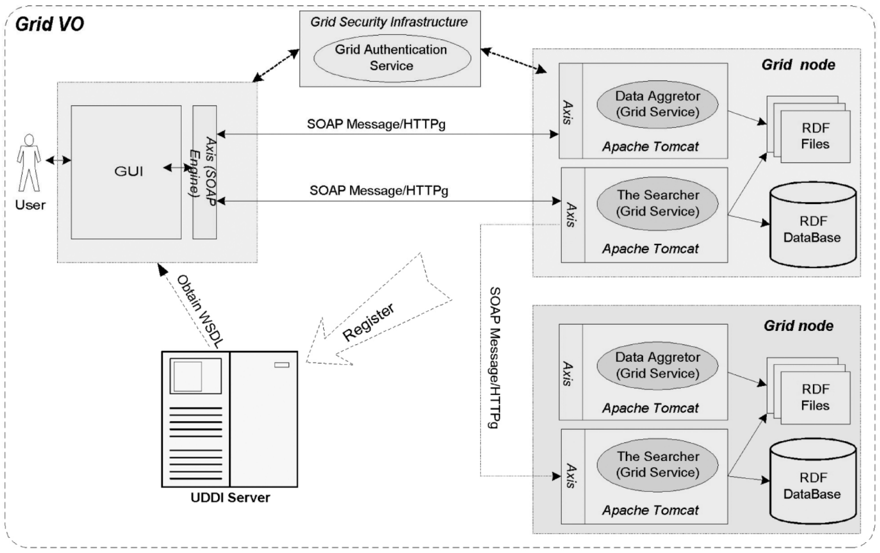


Figure 3.
The architecture of the gDisDL system

challenge, we organize the distributed gDisDL nodes and their RDF metadata using the scalable Virtual Organization (VO) mechanism of the grid (Foster *et al.*, 2001). To this end, we have designed gDisDL to comply with the Open Grid Service Infrastructure (OGSI) specifications (Foster *et al.*, 2002).

One design challenge is how to encode and store metadata in RDF. Currently, most RDF-based systems ask users to feed RDF metadata directly (Alexaki *et al.*, 2001). Therefore, users have to encode resource metadata into RDF syntax manually, a process which is inconvenient and difficult. To cope with this problem, we have designed and implemented the DataAggregator, a component which generates the RDF syntax automatically and subsequently stores the RDF-encoded metadata in RDF storage.

Another challenge is the storage and query of RDF metadata. Typically, an RDF database can be used to store RDF metadata, and an RDF query language can be used to express queries and execute them on the database. In the EU-MEDIN use-case scenario, however, most projects do not contribute large bodies of metadata; also, metadata updates are not very frequent. Therefore, a database system would be an overly expensive solution for our system requirements. Moreover, we would have to choose one among several existing RDF databases and query languages (Karvounarakis *et al.*, 2002), and integrate it with gDisDL. Thus, the system would depend heavily on the chosen database system. Currently, the default behavior of gDisDL is to store RDF metadata in plain files. In order to make gDisDL more open and extensible, however, we have designed the Searcher component as a "query translator" that processes user requests and generates queries according to the back-end storage used in each gDisDL site. Thus, the back-end storage system remains transparent to the user and any RDF database system can be deployed with gDisDL.

Another important design consideration is security. The security of the gDisDL is mainly concerned with two aspects: one is accessing distributed RDF metadata securely; another is that the shared RDF metadata should be protected from unauthorized access. In other words, the RDF metadata sharing of the gDisDL system should be governed by a set of rules and policies, such as what is shared, who is allowed to share, and the conditions under which sharing occurs. To address the security issues, we adopt the Secure Socket Layer (SSL) with the Grid Security Infrastructure (GSI) credentials in the gDisDL system (Freier, 1996). GSI credentials are based on X.509 certificates, and the GSI-enabled HTTP protocol (HTTPg) is used as the transport layer protocol to establish an encrypted connection between gDisDL clients and services (Housley *et al.*, 2002).

gDisDL components

The Data Aggregator. The Data Aggregator is a grid service that encodes information about EU-MEDIN resources into RDF. The RDF encoding work is done by creating a Jena RDF model in built-in memory and inserting the information from the Editor as a set of triples in Subject, Predicate, Object format using the Jena RDF framework (JENA, 2003). A Jena RDF model is an RDF graph in which RDF triples are represented as node-arc-node subgraphs. In a subgraph, the nodes are corresponding to the subject and the object of the triple; whereas the directed arc corresponds to the predicate.

In particular, the Data Aggregator service:

- Gets the information describing an EU-MEDIN resource from the Editor. The Editor invokes the Data Aggregator service and sends it the resource information using SOAP messages via HTTPg POST command.
- Validates the provided information with respect to the EU-MEDIN RDF schema, taking into account resource classes, class properties, restrictions, and data types.
- If the information is deemed valid, the Data Aggregator will generate a unique URI to identify the resource. The URI contains two parts: one is the location of the gDisDL system that the user uses (e.g. the domain name); the other is the time when the RDF metadata was generated.
- If the data is not valid, the Data Aggregator returns a service fault message to the Editor, and ends the process.
- The Data Aggregator transforms the validated data together with the created URI into an RDF Jena model, which is a collection of triples, each consisting of a subject, a predicate and an object (Klyne and Carroll, 2002). The RDF metadata of the resource is thus created.
- The RDF metadata is encoded in RDF/XML format and saved in an RDF file.

By default, in a gDisDL node, RDF metadata for the same kind of EU-MEDIN resources (i.e. resources belonging to the same EU-MEDIN class) are stored into the same RDF file. For example, all RDF metadata describing journal papers are kept in one file, all RDF metadata describing data sets are kept in another file, and so on. Thus, when we look for metadata about journal papers, we can search into local or remote RDF files dedicated to journal-paper metadata.

The Searcher. The Searcher is a grid service responsible for searching the distributed metadata and answering queries about the RDF resources. A Searcher grid service allows a client to query the RDF information held in the Jena models. The query of the Searcher service is RDF triple-oriented. In other words, an RDF triple pattern (e.g. {subject(?), predicate(?), object(?)}) will be generated according to the user's request; and it is used to match the RDF triples in the Jena model. For example, a simple user request "Find reports from the CrossGrid project" specifies that the resource is about a Report whose predicate is projectacronyms and its object is "CrossGrid". Therefore, the query triple pattern can be represented as {?x, disdl:projectacronyms, "CrossGrid"} of the Report RDF resource. The Searcher executes this query by matching all the RDF triples of the report resource in the RDF Jena model against the triple pattern, and retrieves a set of matches, which have project property ("p" = disdl:projectacronyms), and value "CrossGrid" ("o" = CrossGrid).

As shown in Figure 4, the Searcher may need to search the RDF metadata not only stored locally (i.e. the RDF metadata is located in the gDisDL node that the Searcher service belongs to), but also stored remotely (the RDF metadata is stored in another gDisDL node) in order to answer a query. On purpose of simplicity, the Searcher will start with searching the RDF metadata that is stored "locally". After receiving a request from a client, the Searcher checks the data of the request according to the RDF Schema, translates it into an RDF triple pattern. Next step, the Searcher will locate the RDF file that may contain the desired RDF metadata, and upload the RDF metadata into the Jena model as a set of RDF triples from the RDF file. After that, the Searcher

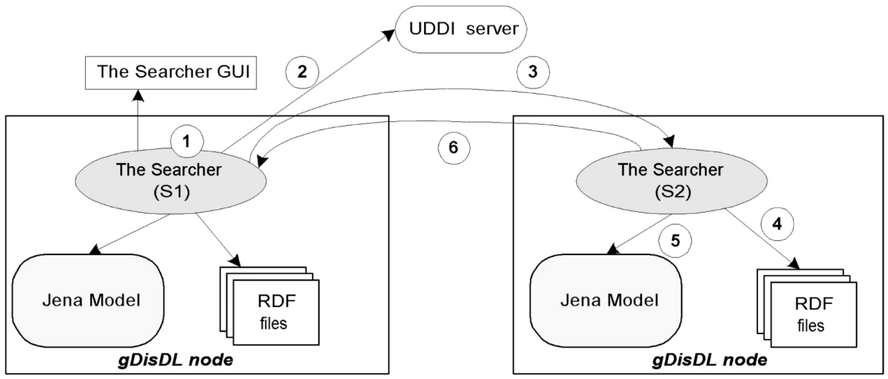


Figure 4. Diagram describing the searching process

can explore all the RDF triples in the model (e.g. the name or URI of the resources and their properties), compare them according to the RDF triple pattern, and retrieve the matched RDF metadata. Finally, the client will get the matched RDF metadata from the Searcher in a SOAP message responding its query in an HTTPg GET option. In the case that the back-end storage is an RDF database system, the Searcher will "translate" the user's request (i.e. the resource type, the property, values of the properties) into a proper RDF query language format, and then query the RDF database.

If the RDF metadata is not stored locally, the Searcher will need to query the RDF metadata in a "remote" distributed gDisDL node. In this case, the Searcher service will be a client to invoke another Searcher grid service in order to search the desired RDF metadata that is stored there (see Figure 4). In other words, if the desired RDF metadata can not be found in the "local" gDisDL node, the Searcher will invoke another Searcher service.

The process of remote search can be described as a series of steps (see also Figure 4):

(1) The Searcher (S1) does not get the requested RDF metadata from the "local" node. The S1 grid service will act as a "client" to invoke another Searcher grid service in order to find the desired RDF metadata.
(2) The "client" (i.e. the S1) first queries the UDDI server to find a properSearcher grid service based on the published GWSDL and SDE information.
(3) S1 invokes another Searcher service (S2) and sends it the query using SOAP messages via HTTPg POST method.
(4) S2 locates an RDF file that may contain the desired RDF metadata, and then uploads the RDF metadata as a set of RDF triples into the Jena model from the RDF file.
(5) S2 searches the RDF metadata in this model.
(6) The RDF metadata will be sent back to S1 in SOAP using HTTPg POST method.

There are many Searcher services available in the distributed gDisDL nodes whilst the "client" (e.g. S1) needs to locate another Searcher service. In order to facilitate locating a "remote" Searcher service, the Searcher service provides information about the RDF metadata stored in the gDisDL node, such as the types of the gDisDL RDF resources, the different predicate index information of the RDF metadata stored in the gDisDL node (e.g., author index information, project index information). The information should be well structured and attached to a Searcher service. We adopt Service Data Elements (SDE) mechanism of the grid service for this purpose (Tuecke, 2002). The SDE of a grid service is a structured collection of information that is attached to the grid service. The information contained in the SDE will be published in the UDDI server and can be queried by the clients of the service. Currently the provided information is about the types of the gDisDL RDF resources. The structure and the content of the SDE that is attached to the Searcher grid service is shown below:

```
1 <rdf:RDF
2 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-
ns#"
3 xmlns:dc="http://purl.org/dc/elements/1.1/"
4 xmlns:dlib="http://www.eu-medin.org/2003/02/Schema#"
>
5 <rdf:Description rdf:about="http://www.eu-
medin/gDisDL/gDStore1">
6 <dlib:availability> plain file </dlib:availability>
7 <dlib:availability> RDF database </dlib:availability>
8 <dc:type> plain RDF file </dc:type>
9 <dc:coverage>
10 <rdf:Bag>
11 <rdf:li> Project </rdf:li>
12 <rdf:li> JournalPaper </rdf:li>
13 <rdf:li> DataSet </rdf:li>
14 <rdf:li> Software </rdf:li>
15 <rdf:li> Hardware </rdf:li>
16 </rdf:Bag>
17 </dc:coverage>
18 <dc:type> RDF database </dc:type>
19 <dc:language> RDQL </dc:language>
20 </rdf:Description>
21 </rdf:RDF>
```

As you can see, lines 5 and 6 specify that the back-end storage is available both in plain RDF files and in the RDF database; lines 10 to 16 describe that in the RDF metadata there are about five kinds of RDF resources, namely Project, JournalPaper, DataSet, Software, and Hardware. Line 19 specifies that the query language of the RDF database is RDQL.

The UDDI Server. The UDDI server of the gDisDL system is a provider-specific server that is used to publish the information about gDisDL grid services. It is also a registry of the gDisDL system where all the gDisDL grid service providers should register their gDisDL Grid service by submitting the description of the gDisDL grid services, access policy, security policy, and service data elements (SDE). The information is then published via the internet and a client in turn can use the published information to discover and invoke a desired gDisDL grid service.

Two key pieces of data are maintained in an entry of the UDDI repository: the GWSDL file represented as a tModel, which represents the grid-service metadata in UDDI, and the binding information of the service implementation (UDDI version 3.0.2).

A client can find a desired gDisDL grid service by the tModel, and invoke the desired service by the binding information.

gDisDL GUIs. Two graphical interfaces are designed to facilitate the end user: the Editor and the GUI Client of the Searcher. The Editor is a GUI client of the Data Aggregator grid service where a user can input information and data about EU-MEDIN resources. Similar to the EU-MEDIN portal, it also provides some forms which can be used to collect information about the EU-MEDIN resources. The user will manually input the information using the provided forms, and then the Editor will collect them in XML. The Editor invokes a Data Aggregator grid service, passing the collected information using SOAP message. By default, we set the "local" Data Aggregator service to the Editor. In OGSI, a client locates a grid service using its Grid Service Handle (GSH). The GSH of the "local" Data Aggregator service will thus be assigned to the Editor by default GSH (Tuecke, 2002).

The Searcher GUI Client of the Searcher grid service is needed for users to input the parameters of metadata queries and get results (see Figure 5). The GUI allows users to specify the resource type, the property, values of the properties, etc. The GUI Client also decodes the RDF query results into human-readable form, and displays it on the result window (see Figure 5). The "local" Searcher service is also set as a default service for the Searcher GUI.

Implementation

In this section, we provide some details about our gDisDL prototype implementation.

Figure 6 shows the layered architecture of the gDisDL grid services. The grid gDisDL is implemented within the Open Grid Services Infrastructure (OGSI). Globus Toolkit 3 and Jena are the main development tools used in our implementation. GT3 is a software toolkit that can be used to program grid-based applications. It is implemented in Java following the OGSI specification. GT3 provides several services, programs, utilities, etc. Jena is a Java API that can be used to create and manipulate RDF graphs. It is comprised of object classes to represent graphs, resources, properties, and literals; a graph is called a model and it is represented by the model interface. We use Jena RDF toolkit for creating, manipulating and querying RDF metadata.

To implement a grid service, the first and most important task is to specify the interface of the grid service in Grid Web Service Description Language (GWSDL) (Sotomayor, 2003). Once the interface is correctly defined in GWSDL, implementing the service using Java and other tools is straightforward. Thus we will focus on describing how the interface is defined and what kind of operations can be invoked.

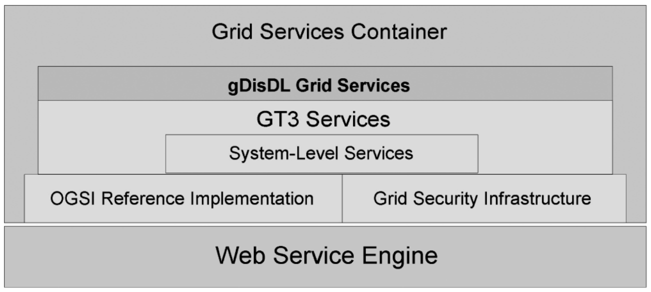
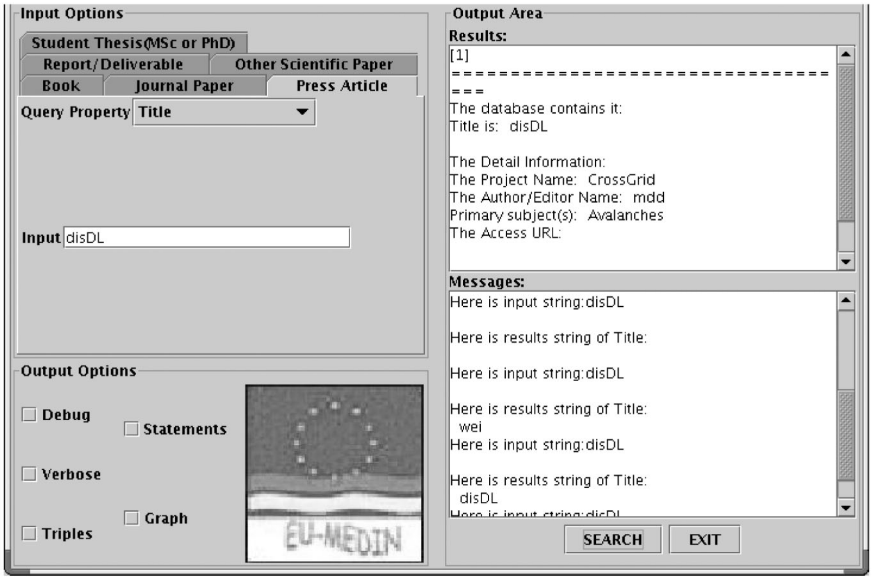


Figure 5.
GUI of gDisDL searcher



Query Property: Title

Input: disDL

Results:
```
[1]
===================================
The database contains it:
Title is:  disDL

The Detail Information:
The Project Name:  CrossGrid
The Author/Editor Name:  mdd
Primary subject(s):  Avalanches
The Access URL:
```

Messages:
```
Here is input string:disDL

Here is results string of Title:

Here is input string:disDL

Here is results string of Title:
  wei
Here is input string:disDL

Here is results string of Title:
  disDL
```

Figure 6.
Layered view of the gDisDL grid services

Data Aggregator grid services and interface
The Data Aggregator service processes the collected information and data from the Editor client, encodes it into RDF format, and saves it as RDF files. The interface of the DataAggregator grid service is defined in GWSDL, as shown below:

```
<gwsdl:portType name="DataAggregatorPortType" extends="ogsi:GridService">
<operation name="retriveInfo">
<input message="tns:GetInputMessage"/>
<output message="tns:GetOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 </operation>
<operation name="getRDF">
<input message="tns:GetRDFInputMessage"/>
<output message="tns:GetRDFOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 </operation>
<operation name="validate">
<input message="tns:ValInputMessage"/>
<output message="tns:ValOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 </operation>
```

Operation/PortType. retrieveInfo() is used to get values of the RDF triples from a client; getRDF() creates an RDF graph and assigns the values of the triples; validate() checks and validates the input data according to the syntax of EU-MEDIN RDF metadata; saveRDF() saves the RDF metadata into a file in RDF/XML syntax.

Searcher grid services and interface

The Searcher grid service is used for receiving and answering user queries about EU-MEDIN resources. When searching for RDF metadata, the Searcher can either use the RDF metadata document match() method to search the RDF metadata in an RDF file, or, alternatively, the search can be conducted upon an RDF database, using a database-specific plug-in. Currently we have implemented only the first case. The interface is defined as shown:

```
<gwsdl:portType name="SearcherPortType" extends="ogsi:GridService">
<operation name="preprocess">
<input message="tns:PreInputMessage"/>
<output message="tns:PreOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 < /operation>
<operation name="searchList">
<input message="tns:ListInputMessage"/>
<output message="tns:ListOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 </operation>
<operation name="match">
<input message="tns:MatchInputMessage"/>
<output message="tns:MatchOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 </operation >
<operation name="getStatements">
<input message="tns:GSInputMessage"/>
<output message="tns:GSOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
 </operation>
<operation name="insertStatements">
<input message="tns:ISInputMessage"/>
<output message="tns:ISOutputMessage"/>
<fault name="Fault" message="ogsi:FaultMessage"/>
```

Operation/PortType. The preprocess() operation is used for pre-processing user requests. The searchList() gets the remote RDF metadata information from the UDDI server. The match() operation is used to match RDF triples. The getRDFStatements() is used to fetch the desired RDF metadata. The insertStatements() operation allows for the insertion of RDF triples into the RDF Jena model.

Conclusions and future work

In this paper, we presented an RDF-based grid service approach for organizing and capitalizing European research results in the field of natural disasters. Our approach allows the RDF metadata of European research results in the field of natural disasters to be shared securely and effectively in a heterogeneous network environment, using grid technology. We have described the design and the prototype implementation of gDisDL, an RDF-based, grid-enabled system. gDisDL system is a platform independent system which provides good interoperability with other systems. It can store, manage, and query RDF metadata in a secure and distributed manner.

In the future, we plan to extend gDisDL with RDF-database plug-ins for supporting more efficient storage of RDF metadata, and to extend the searching mechanisms of the Searcher in order to integrate RDF databases into our system. Furthermore, we are investigating the development of a semantic UDDI that would improve the utilization of the gDisDL grid services. Our approach can be easily generalized to cope with metadata of different kinds and evolve as a generic search engine for RDF-encoded metadata posted on the grid.

References

Alexaki, S., Christophides, V., Karvounarakis, G., Plexousakis, P. and Tolle, K. (2001), "The ICS-FORTH RDFSuite: managing voluminous RDF description bases", in Staab, S. (Ed.), *Proceedings of the Second International Workshop on the Semantic Web (SemWeb '01)*, pp. 1-13.

Berners-Lee, T. (2000), *Primer: Getting into RDF and Semantic Web using N3*, The World Wide Web Consortium.

Berners-Lee, T., Fielding, R. and Masinter, L. (1998), "Uniform Resource Identifiers (URI):Generic Syntax", The Internet Engineering Task Force.

Bray, T., Paoli, J., Sperberg-McQueen, C.M., Maler, E. and Yergeau, F. (2004), *Extensible Markup Language (XML) 1.0*, 3rd ed., World Wide Web Consortium.

Brickley, D. and Guha, R.V. (2000), *Resource Description Framework (RDF) Schema Specification 1.0*, World Wide Web Consortium.

Connolly, D., van Harmelen, F., Horrrocks, I., McGuiness, D.L. and Patel-Schneider, P.F. (2001), *DAML + OIL (March 2001) Reference Description*, World Wide Web Consortium.

Dublin Core (2003), "Dublin Core metadata element set, version 1.1: reference description", available at: http://dublincore.org/documents/2003/06/02/dces

EU-MEDIN (2003), EU-MEDIN portal, available at: www.eu-medin.org

Foster, I., Kesselman, C., Tsudik, G. and Tuecke, S. (1998), "A security architecture for computational grids", paper presented at the 5th ACM Conference on Computer and Communications Security Conference.

Foster, I., Nick, J., Kesselman, C. and Tuecke, S. (2002), "The physiology of the grid: an open grid services architecture for distributed systems integration", paper presented at the Open Grid Service Infrastructure WG, Global Grid Forum.

Foster, I., Tuecke, S. and Kesselman, C. (2001), "The anatomy of the grid: enabling scalable virtual organizations", *Supercomputer Applications*, Vol. 15 No. 3.

Freier, A.O. and Paul, C. (1996), "The SSL Protocol Version 3.0", Transport Layer Security Working Group.

Housley, R., Polk, W., Ford, W., Solo, D. and "Internet, X. (2002) Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile".

JENA (2003), *Jena – a Semantic Web Framework for Java*, Hewlett-Packard, available at: http://jena.sourceforge.net.

Karvounarakis, G., Alexaki, S., Christophides, V., Plexousakis, D. and Scholl, M. (2002), "RQL: a declarative query language for RDF", paper presented at the 11th International World Wide Web Conference (WWW'02), Honolulu, HI, May 7-11, Vol. 02.

Klyne, G. and Carroll, J. (2002), *Resource Description Framework (RDF): Concepts and Abstract Data Model*, technical report, The World Wide Web Consortium.

Manola, F. and Miller, E. (2003), "RDF primer". W3C Working Draft, available at: www.w3.org/TR/rdf-primer/

Patel-Schneider, P.F., Hayes, P. and Horrock, I. (2004), *OWL Web Ontology Language Semantics and Abstract Syntax*, World Wide Web Consortium.

Sotomayor, B. (2003), *The Globus Toolkit 3 Programmer's Tutorial*, technical report, The Globus Alliance.

Tuecke, S. *et al.* (2002), "Open Grid Service Infrastructure (OGSI) version 1.0", Open Grid Service Infrastructure WG, Global Grid Forum.