

# A Grid-enabled Architecture for Geospatial Data Sharing

Yanfeng Shu, Jack Fan Zhang, Xiaofang Zhou  
School of ITEE  
The University of Queensland  
{yshu, jfz, zxf}@itee.uq.edu.au

## Abstract

*This paper explores combining OGC and Grid technologies for geospatial data sharing. OGC and Grid technologies have different focuses: OGC technologies focus more on interface specifications and encodings for the access to geospatial data sources, while Grid technologies focus more on the collaboration and sharing of resources. By combining both, we hope to leverage each other's strengths. In the paper, we first introduce our Grid-enabled architecture which is based on both OGC and Grid technologies, then we discuss how Grid-enabled OGC web services can be integrated with non-Grid-enabled ones, and finally, we report our initial implementation of a case study.*

## 1 Introduction

Geospatial data, in general, refers to a class of data that has a geographic or spatial nature, e.g., the information that identifies the geographic location and characteristics of natural or constructed features and boundaries on the earth. Today, the value of related geospatial data in various application domains has been widely recognized. For example, by analyzing the sightings data together with other geospatial data, we can model species distribution and make useful predictions. As geospatial data is stored in a variety of systems and formats, one important issue for geospatial applications is interoperability, i.e., seamless integration and sharing of geospatial data from distributed heterogeneous data sources.

To address this interoperability issue, several initiatives have been undertaken, among which is the Open Geospatial Consortium (OGC) [14]. OGC is a non-profit, international, voluntary consensus standards organization that is leading the development of standards for geospatial and location based services. Currently, the OGC has developed a number of web services specifications that enable the interoperability of geospatial data sources in a distributed environment, such as Web Coverage Services (WCS) [5], Web Fea-

ture Services (WFS) [17], Web Map Services (WMS) [3], and Catalogue Services (CSW) [12].

While the OGC and its web services specifications represent domain-specific efforts towards geospatial information sharing, Grid technologies and infrastructures represent domain-independent efforts which aim at a much broader “resource sharing and problem solving in dynamic, multi-institutional virtual organizations” [8]. The Open Grid Services Architecture (OGSA) [6] is designed to facilitate the interoperability among different Grid deployments, which aligns Grid technologies with Web Services technologies, and introduces a service-oriented paradigm into the Grid. The Open Grid Services Infrastructure (OGSI) [15] is the first formal and technical specification of OGSA, which has several implementations such as Globus Toolkit 3.0 (GT3) [10], and is now evolving towards the Web Services Resource Framework (WSRF) [19] to embrace new Web Services standards.

Compared to OGC technologies, Grid technologies are largely “*geospatial-unaware*”, that is, they take little account of the characteristics of geospatial data. However, they have been very successful in the collaboration and sharing of resources, which the current OGC technologies somewhat lack in. By combining both technologies, we hope to leverage each other's strengths for geospatial data sharing: on the one hand, Grid technologies become *geospatial-aware*, and on the other hand, the OGC web services are *Grid-enabled*, both of which facilitate the distributed management of huge amount of geospatial data sets in a controlled and secure manner.

This paper makes an initial effort to the above attempt. In the paper, we first propose a Grid-enabled architecture for geospatial data sharing, which is based on both OGC and Grid technologies. Based on the proposed architecture, each OGC web service is made Grid-enabled, and exposed as a Grid service that is fully integrated with other Grid services through OGSA-DAI (OGSA Data Access and Integration) [13] and OGSA-DQP (OGSA Distributed Query Processor) [2]. Then we discuss how Grid-enabled OGC web services can be integrated with non-Grid-enabled ones.

And finally, we report our initial implementation of a case study.

The rest of the paper is organized as follows: Section 2 presents some background information; Section 3 introduces a Grid-enabled architecture for geospatial data sharing, and describes at detail how the basic services are interacted in a specific scenario under the proposed architecture; Section 4 discusses the integration of Grid-enabled OGC web services with non-Grid-enabled ones; Section 5 reports our current implementation status; Section 6 reviews the related work; and Section 7 concludes the paper and points out the future work.

## 2 Background

### 2.1 OGC Web Services

Among the OGC Web services, WFS, WMS, and WCS specify the interfaces for the access to geospatial data sources: WFS provides access to map features (vector data) which are encoded in GML (Geographic Markup Language); WMS produces maps which are encoded as pictures in raster formats like GIF, JPEG, and SVG; and WCS extends WMS and allows access to coverage data (values or properties of geographic locations). All these services define the implementation and use of several basic operations. For example, WFS defines 5 operations, *getCapabilities*, *describeFeatureType*, *getFeature*, *transaction*, and *lockFeature*. To access a WFS, a client usually first sends a *getCapabilities* request to the WFS server to learn which feature types it can service and what operations are supported on each feature type; then the client sends a *describeFeatureType* request to get the structure information of the interested feature type; and finally a *getFeature* request is sent to get the feature data.

Different from WFS, WMS, and WCS, CSW is a catalogue service, which specifies the interfaces through which other services can be published and discovered.

### 2.2 Grid Services

OGSA adopts a common representation for all resources (e.g., computational and storage resources, programs, databases): each resource in OGSA is represented as a *Grid service*, i.e., a Web service that provides a set of well-defined interfaces and follows specific conventions [7].

OGSI further specifies the basic interfaces (or portTypes) to be implemented by Grid services, such as *GridService* (GS), *Factory*, *ServiceGroup*, and so on. Most of these interfaces are optional, except *GridService*, which is a mandatory interface, and must be implemented by all Grid services. Depending on what interfaces are implemented, Grid services with different functionalities may result.

Grid services can be instantiated dynamically. Each instantiation of a Grid service generates a Grid service instance which is identified by the Grid Service Handle (GSH) and the Grid Service Reference (GSR). The difference between GSH and GSR is that, GSH is invariant, while GSR is stateful and can change over the life time of a service instance. To create a service instance, a Grid service, called ‘factory’, is invoked, which implements the *Factory* interface (services and factories are located by another Grid service, called ‘registry’, which implements *ServiceGroup* (SGR) portType).

### 2.3 OGSA-DAI/DQP

The main objective of OGSA-DAI/DQP is to provide a uniform service interface for data access and integration over the Grids. Both OGSA-DAI and OGSA-DQP build upon OGSA.

OGSA-DAI extends Grid services with new services and portTypes for individual data access, such as *Grid Data Service* (GDS), *Grid Data Transport* (GDT), *Grid Data Service Factory* (GDSF), and *DAI Service Group Registry* (DAISGR). *Grid Data Service* is the primary OGSA-DAI service, which supports data access through the GDS portType (via the *perform* operation) and data delivery through GDT portType. GDS instances are created by invoking GDSF which can be located by DAISGR.

OGSA-DQP extends OGSA-DAI with two new services (and their corresponding factories) for distributed query processing over multiple data sources: a *Grid Distributed Query Query Service* (GDQS) which compiles, optimises, partitions and schedules distributed query execution plans over multiple execution nodes in the Grids, and a *Grid Query Evaluation Service* (GQES) which is in charge of a partition of the query execution plan assigned by a GDQS. In GDQS, a *Grid Distributed Query* (GDQ) portType is added for importing source schemas. OGSA-DQP itself does not do any schema integration.

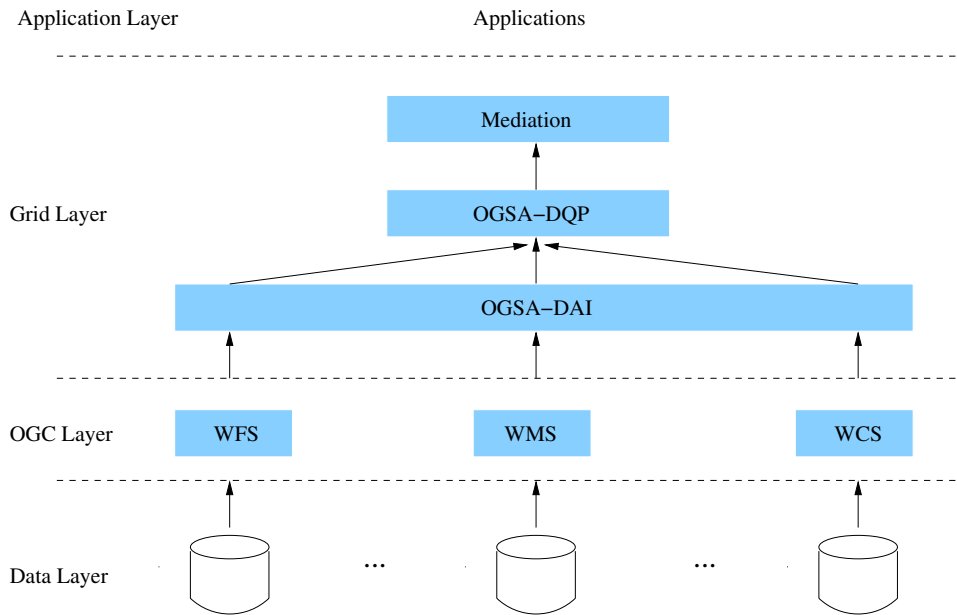
## 3 “Grid-enabling” OGC Web Services

In this section, we first introduce our proposed architecture which integrates OGC and Grid technologies for geospatial data sharing by Grid-enabling OGC Web Services; then we describe at detail how services are interacted in a typical application scenario under such an architecture.

### 3.1 The Grid-enabled Architecture

Figure 1 shows the architecture, which is abstracted into four layers:

- *Data Layer*: this layer consists of a set of autonomous, distributed geospatial data sources, where data may be



stored in ESRI shapefile, PostGIS, ArcSDE, Oracle and DB2 with spatial options, and the like.

- **OGC Layer:** this layer hides the heterogeneities among different geospatial data sources, and exposes to the upper layer as *OGC-compliant* web services such as WFS, WMS, and WCS. By OGC-compliant, we mean that, the internal implementations of these services are compatible with OGC specifications.
- **Grid Layer:** this layer builds upon OGSA-DAI and OGSA-DQP. At this layer, we make OGC web services (i.e., WFS, WMS, and WCS) Grid-enabled by implementing or extending mandatory portTypes (e.g., GDS portType) of OGSA-DAI. Accordingly, OGC web services are exposed as Grid services, thus integrating into the Grids.

We base on OGSA-DQP for distributed query optimization and execution. As OGSA-DQP touches little on schema mediation among heterogeneous data sources, we introduce a Mediation module which includes two new services, Schema Mapping Service (SMS) and Mediator Service (MS). SMS is used for building mapping relationships among schemas of different data sources, while MS is used for query reformulation based on mappings imported from SMS. Both services extend OGSA-DAI and OGSA-DQP with new portTypes. SMS implements GDS and GDT portTypes of OGSA-DAI, GDQ portType of OGSA-DQP (for importing schemas of data sources), and

adds a new portType, BM, for building mappings among different sources. MS implements GDS and GDT portTypes of OGSA-DAI, and adds two new portTypes, IM for importing mappings from SMS, and QR for query reformulation.

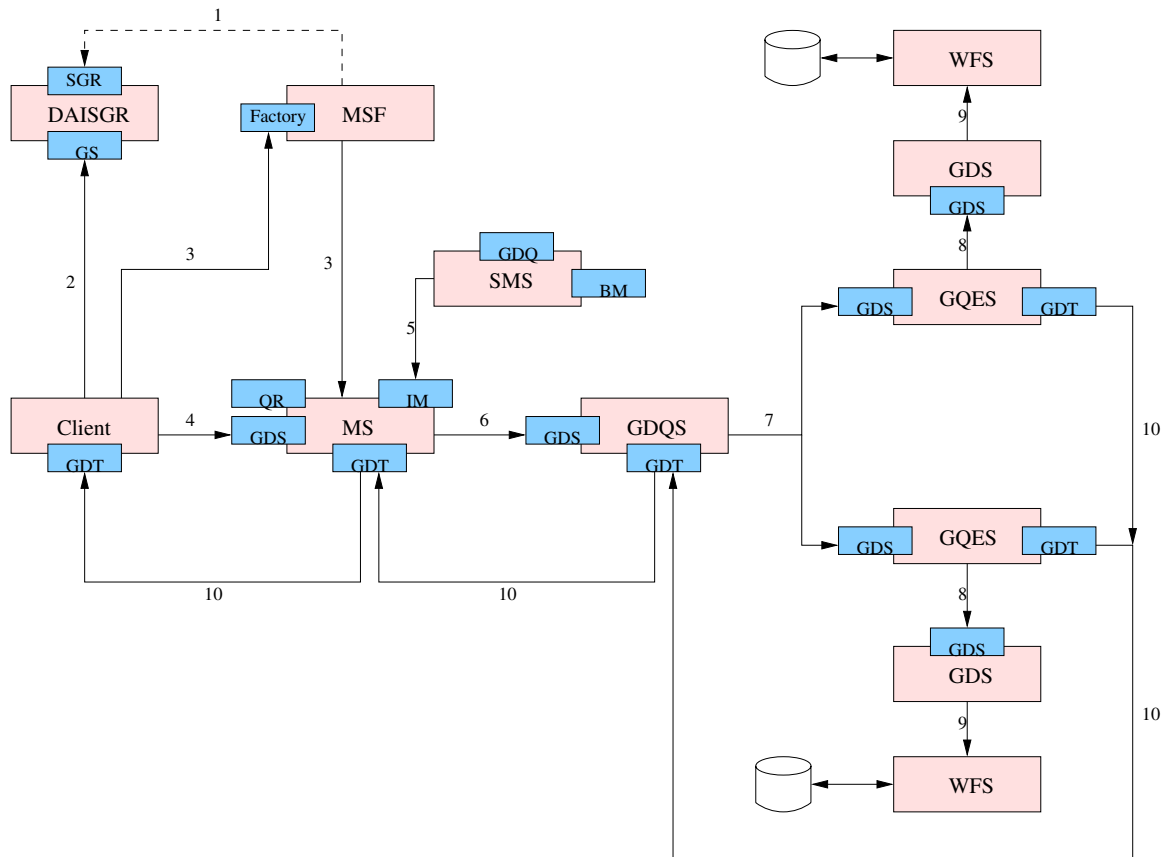
- **Application Layer:** this level performs some data intensive operations, e.g., data analysis spanning over several data sources including geospatial ones.

### 3.2 Service Interaction

This part describes how services are interacted under the proposed architecture. Figure 2 shows service interactions in a typical scenario. In the figure, dashed lines denote interactions that take place during the system setup, and solid lines denote interactions that take place during query processing.

Suppose two geospatial data sources are accessed and shared, the steps of service interactions are as follows:

1. Each Factory instance, e.g., MSF, GDQSF, GDQF, registers itself with the DAISGR registry during the system setup via a *registerService* operation. Note that the registry service in our architecture is extended to support geospatial metadata.
2. A client service finds a factory instance that supports the required retrieval by sending a *findServiceData* request to the DAISGR.

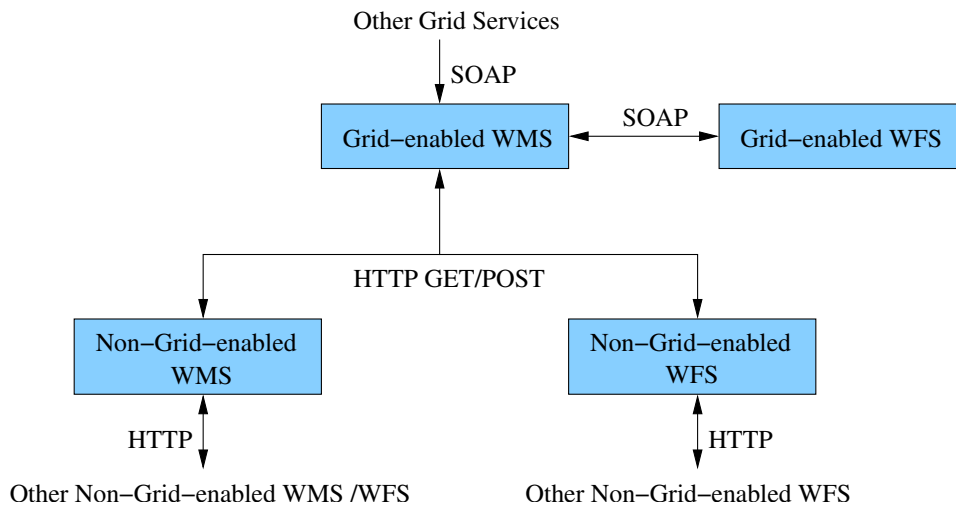


3. The client service uses the factory instance obtained from the last step to create the corresponding service instance. For example, the Mediator Service factory instance is used to create MS instance. Note that step 2 and step 3 are invoked whenever a new service instance is needed.
4. A user query is submit via the *perform* operation in the GDS portType.
5. The MS imports semantic mappings <sup>1</sup> from a Schema Mapping Service instance (obtained through service interactions in step 2 and 3), and then reformulates the user query via the Query Reformulation portType.
6. The reformulated query is passed to the GDQS instance via a *perform* call.
7. The GDQS compiles the query into a distributed query plan, each partition of which is scheduled in one or more execution nodes (in our scenario, we suppose two partitions are generated, with each scheduled in one

<sup>1</sup>We suppose mappings are built during the system setup.

- execution node). For each execution node, a GQES instance is created, and a partition of the query plan is handed over to the GQES instance.
8. Each GQES instance starts the evaluation, and interacts with GDS instances.
9. Each GDS instance obtains data via operations defined by OGC web services specifications. In case of WFS, operations such as *GetCapabilities*, *DescribeFeatureType*, and *GetFeature*, are invoked.
10. Finally, results are propagated back to the client via the GDT portType.

All Grid services are described by the Grid Web Service Description Language (GWSDL) - an extended WSDL file to support Grid services, and they use UDDI registries for service discovery and the SOAP for inter-service communication.



#### 4 Integrating with non-Grid-enabled OGC Web Services

This section discusses how Grid-enabled OGC web services can be integrated with non-Grid-enabled ones.

The OGC web services can be cascaded in some way. For example, a WFS can use another WFS as its feature source; and a WMS can act as a WMS client and combine contents from several WFSs. The benefit from such cascading is that, the contents and capabilities of several services can be aggregated, thus facilitating the access. Thus, to integrate with non-Grid-enabled OGC web services, we can make the cascaded OGC web services Grid-enabled, and use these Grid-enabled OGC web services as gateways to non-Grid-enabled ones.

As Grid-enabled services are invoked through the SOAP over HTTP, we need to parse requests to these services, extract all the parameters, and then repackage them in a new request and send over HTTP (i.e., HTTP GET and POST, as defined in OGC specifications) to non-Grid-enabled services.

Figure 3 illustrates one example integration process. In the figure, a Grid-enabled WMS acts as the client to a Grid-enabled WFS, and meanwhile, acts as the client to a non-Grid-enabled WMS and a non-Grid-enabled WFS. When both services are Grid-enabled, they communicate via SOAP, otherwise, HTTP is used instead.

#### 5 Implementation Status

We begin our implementation with an environmental case study, which is centered around an existing research

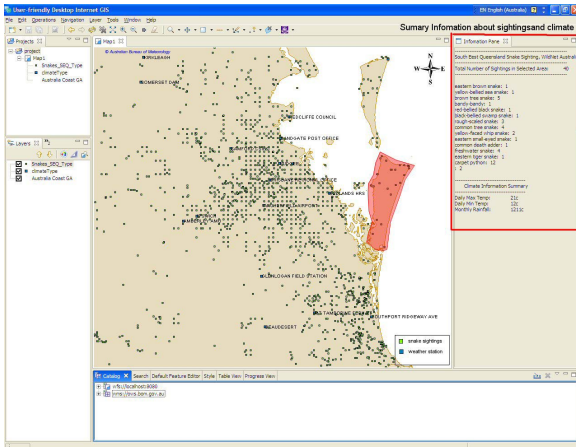
and development program in the Queensland Environmental Protection Agency (EPA), WildNet. In the following, we first have a look at WildNet, then describe our current implementation status.

The WildNet database contains 3.5 million records of wildlife sightings and listings of around 20,000 species such as plants, mammals, birds, reptiles, amphibians, freshwater fish, marine cartilaginous fish and butterflies in Queensland. Species are classified by a taxonomy including multiple levels, kingdom names, class names, family names, scientific names, and common names. One fundamental function required by WildNet is sightings visualization, i.e., sightings can be visualized on the map for a selected area such as protected area, forestry area, or local government area, or a defined area bounded by minimum and maximum latitudes and longitudes. Beyond this, we can model the species distribution, and identify the abnormalities or outliers through the analysis of the sightings data together with the environmental data such as vegetation data or climate profiles.

Based on the proposed architecture, our implementation consists of two steps:

- OGC compliance. All geospatial data sources (e.g., sightings data, climate data, vegetation data, ) are exposed as OGC-compliant web services.
- Grid enabling. All data sources are exposed as Grid services. In this step, we make OGC-compliant web services Grid-enabled.

Currently, we have finished the first step. We chose GeoServer [9] for the server-side WFS server development and uDig [16] for the client-side interface development. Our implementation was developed using Java Servlet technology under Eclipse + Tomcat: basic WFS requests (e.g.,



getCapability, getFeatureDescription, getFeature) are supported through HTTP GET /POST, and results which are in XML or XML-compliant GML formats are returned.

As we expect that the Grid registry service can be extended for geospatial data sources, we only implemented a simple catalogue service which can meet our needs but may not follow the OGC specifications.

Figure 4 gives a screen snapshot which shows snake sightings together with climate profiles in southeast Queensland. A uDig plugin has been developed to create a polygon selection (rather than a simpler boundary box), through which we can specify a specific area we are interested in (e.g., North Stradbroke Island as shaded in the figure), and get the data of interested features (e.g., snake sightings and climate profiles) for further analysis (e.g. statistics summarization).

## 6 Related Work

Research on combining OGC and Grid technologies is mainly done by LAITS group in George Mason University [4, 20, 18]: [4] describes a preliminary implementation that applies Grid technology to the Earth observation environment through the integration of the Globus Toolkit with the NASA Web GIS Software Suite (NWGISS); both [20] and [18] focus on the design and implementation of a Grid-enabled geospatial catalogue service, with one main difference in the GT version used for the serve implementation (in [20], the implementation is based on GT 2.2, while in [18], the implementation is based on GT 3.2). Our work is different from theirs is in that, we introduce a Grid-enable architecture for geospatial data sharing, and focus more on geospatial data integration and sharing based

on OGSA-DAI and OGSA-DQP. Moreover, we consider the integration of Grid-enabled OGC web services with non-Grid-enabled ones.

Other orthogonal work includes [1] and [11], both of which consider Web Service-compatible OGC web services. In [1], an alternative model based on UDDI is designed for publishing and discovering geospatial services, in order to have the advantage being interoperable with other web services; In [11], an implementation method of Web Feature Service based on Web Services is discussed.

## 7 Conclusion and Future Work

In this paper, we explored the integration of OGC and Grid technologies for geospatial data sharing. Though OGC technologies have done well in interface specifications and encodings for the access to geospatial data source, they consider little about the coordination and sharing of resources. By combing OGC and Grid technologies, we hope to facilitate the distributed management of geospatial data sets in a controlled and secure manner. To achieve so, we introduced a Grid-enabled architecture based on both technologies. Meanwhile, we discussed how Grid-enabled OGC web services can be integrated with non-Grid-enabled ones.

The future work includes the full implementation of the case study based on OGSA-DAI/DQP. Meanwhile, we plan to put more efforts on large-scale data sharing. Modern data sharing is typically characterized by the large volumes of data involved, and the heterogeneity of sources accessed. With such a setting, we need to address such issues like interoperability, extensibility, and scalability. Grid technologies only address the interoperability issue in a limited scope, with more focus on the system heterogeneity than data heterogeneity, and they consider little about the extensibility and scalability issues: resources (services) are managed either in a centralized or hierarchical manner, which cannot cope and scale well with large numbers of dynamic data sources. Traditional data integration technologies address data heterogeneity, however, they depend on a centralized model for schema mediation, which is not extensible for the ad hoc sharing of large numbers of autonomous data sources.

To make up for the insufficiencies of both Grid and data integration technologies, we are considering to employ P2P technologies and integrate them into both in our architecture (P2P technologies share the same final goal as Grid technologies, i.e., resource sharing, however, they focus more on decentralization and scalability). For example, to combine P2P technologies with Grid technologies, we can organize resources (or services) in Grids in a P2P manner for scalable service discovery and deployment: each service (peer) is connected to a set of other services (neighbors); given a request, a service first checks whether itself

is the required service; if not, it will forward the request to its neighbors, and so on, until the request is satisfied. By doing so, we avoid centralized management of resources. Data integration can also be done in a P2P manner. For example, rather than defining a global mediated schema, we can build semantic mappings directly between schemas of different sources. Each data source corresponds to a peer, which maintains a few mappings with other peers (neighbors). Given a query, a peer transforms the query based on the mappings maintained locally, and then forwards the reformulated query to semantically related neighbors.

**Acknowledgements** This material is based upon the project supported by the Australian Research Council (ARC) under grant No. SR0567393. We would like to thank the Queensland Environmental Protection Agency (EPA) for providing the sightings data, and Dr. David Pullar for the environmental data and useful discussions.

## References

- [1] M. S. Aktas, G. Aydin, G. C. Fox, H. Gadgil, M. Pierce, and A. Sayar. Information Services for Grid/Web Service Oriented Architecture (SOA) Based Geospatial Applications. In *Technical Report*. <http://grids.ucs.indiana.edu/pliupages/publications/>, 2005.
- [2] M. Alpdemir, A. Mukherjee, N. Paton, P. Watson, A. Fernandes, A. Gounaris, and J. Smith. OGSA-DQP: A Service-Based Distributed Query Processor for the Grid. In *Proceedings UK e-Science All Hands Meeting*, 2003.
- [3] J. de La Beaujardiere. OpenGIS Web Map Service (WMS) Implementation Specification, version 1.3.0. 2006.
- [4] L. Di, A. Chen, W. Yang, and P. Zhao. The Integration of Grid Technology with OGC Web Services (OWS) in NWGISS for NASA EOS Data. In *GGF8, HPDC12*, 2003.
- [5] J. Evans. OpenGIS Web Coverage Service (WCS) Implementation Specification, version 1.0. 2006.
- [6] I. Foster, C. Kesselman, J. Nick, and S. Tuecke. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. In *Technical Report, Globus Project*. <http://www.globus.org/research/papers/ogsa.pdf>, 2002.
- [7] I. Foster, C. Kesselman, J. M. Nick, and S. Tuecke. Grid Services for Distributed System Integration. In *IEEE Computer Society Press, Volume 35, Issue 6*, 2002.
- [8] I. Foster, C. Kesselman, and S. Tuecke. The anatomy of the Grid: enabling scalable virtual organizations. In *International Journal of Supercomputer Applications*, 15(3):200-222, 2001.
- [9] GeoServer. <http://www.refractive.net/geoserver/>.
- [10] Globus Toolkit 3. <http://www.globus.org/toolkit>.
- [11] W. Jia, Y. Chen, J. Gong, and A. Li. Web Service Based Web Feature Service. In *Proceedings of the 20th ISPRS Congress*, 2004.
- [12] D. Nebert and A. Whiteside. OGC Catalogue Services Specifications, version 2.0.0. 2005.
- [13] OGSA-DAI. <http://www.ogsa-dai.org.uk>.
- [14] Open Geospatial Consortium. <http://www.opengeospatial.org>.
- [15] S. Tuecke, K. Czajkowski, I. Foster, J. Frey, S. Graham, C. Kesselman, T. Maquire, T. Sandholm, D. Snelling, and P. Vanderbilt. Open Grid Services Infrastructure (OGSI) Version 1.0. In *IGlobal Grid Forum Draft Recommendation*, 2003.
- [16] uDig. <http://udig.refractive.net/confluence/display/UDIG/Home>.
- [17] P. Vretanos. OpenGIS Web Feature Service (WFS) Implementation Specification, version 1.1. 2005.
- [18] Y. Wei, L. Di, B. Zhao, G. Liao, A. Chen, Y. Bai, and Y. Liu. The design and implementation of a Grid-enabled catalogue service. In *IEEE International Geoscience And Remote Sensing Symposium (IGARSS)*, 2005.
- [19] WSRF. <http://www.globus.org/wsrp>.
- [20] P. Zhao, A. Chen, Y. Liu, L. Di, W. Yang, and P. Lio. Grid Metadata Catalog Service-Based OGC Web Registry Service. In *ACM International Symposium on Advances in Geographic Information System (ACM-GIS)*, 2004.