International Journal of Digital Earth
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/tjde20

A capability matching and ontology reasoning method for high precision OGC web service discovery

Nengcheng Chen [a], Zeqiang Chen [a b], Chuli Hu [a] & Liping Di [b]
[a] State Key Lab for Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, 430079, China
[b] Center for Spatial Information Science and Systems, George Mason University, Fairfax, VA, 22032, USA

Available online: 24 May 2011

To cite this article: Nengcheng Chen, Zeqiang Chen, Chuli Hu & Liping Di (2011): A capability matching and ontology reasoning method for high precision OGC web service discovery, International Journal of Digital Earth, 4:6, 449-470

To link to this article: http://dx.doi.org/10.1080/17538947.2011.553688

International Journal of Digital Earth,
Vol. 4, No. 6, November 2011, 449–470

Taylor & Francis
Taylor & Francis Group

A capability matching and ontology reasoning method for high precision OGC web service discovery

Nengcheng Chen[a]*, Zeqiang Chen[a,b], Chuli Hu[a] and Liping Di[b]

[a]*State Key Lab for Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China;* [b]*Center for Spatial Information Science and Systems, George Mason University, Fairfax, VA 22032, USA*

(*Received 8 June 2010; final version received 7 January 2011*)

Finding the right spatially aware web service in a heterogeneous distributed environment using criteria such as service type, version, time, space, and scale has become a challenge in the integration of geospatial information services. A new method for retrieving Open Geospatial Consortium (OGC) Web Service (OWS) that deals with this challenge using page crawling, link detection, service capability matching, and ontology reasoning, is described in this paper. Its major components are distributed OWS, the OWS search engine, the OWS ontology generator, the ontology-based OWS catalog service, and the ontology-based multi-protocol OWS client. Experimental results show that the execution time of this proposed method equals only 0.26 of that of Nutch's method. In addition, the precision is much higher. Moreover, this proposed method can carry out complex OWS reasoning-based queries. It is being used successfully for the Antarctica multi-protocol OWS portal of the Geo-Information Web Service Portal of the Polar.

Keywords: geospatial information service; link detection; capability matching; OWL-S; ontology reasoning

1. Introduction

The Open Geospatial Consortium (OGC), a major international standards organization, has developed a series of geospatial data web services, such as the Web Map Service (WMS) (Beaujardiere 2006). The Web Feature Service (WFS) (Vretanos 2005) and the Web Coverage Service (WCS) (Whiteside and Evans 2008) provide interoperability, allowing users to work through standard interfaces with the geospatial data, information, and services on the Web.

At present, Geographical Information System (GIS) software makers have extended the existing Web GIS software, such as ArcIMS9.0 of ESRI, MapXtreme 6.5 of MapInfo, and MapGuide of Autodesk (OGC 2010) to support the web services of OGC. Also some software makers and organizations have developed new systems following those standards; for example, LAITS (laboratory of George Mason University in the USA) supports the Earth Observation Data Service by implementing the WCS, Coordinate Transformation Service, and the Image Classification Service (Zhao *et al*. 2005). The OGC standards also play an important

*Corresponding author. Email: cnc_dhy@hotmail.com

ISSN 1753-8947 print/ISSN 1753-8955 online

http://dx.doi.org/10.1080/17538947.2011.553688
http://www.tandfonline.com

role in the construction of regional and national Spatial Data Infrastructure (SDI); for instance, European Spatial Data Infrastructure (ESDI) (Bernard *et al*. 2004) and German's GDINRW (Bernard 2002). Also, they provide interoperable services that support Digital Earth and solve challenging global issues (Yang *et al*. 2008).

There is much OGC service information on the Web. One can obtain 1,370,000, 337,000, and 1,010,000 related page links through Baidu (2010) using 'WMS,' 'WFS,' and 'WCS' as keywords, 15,500,000, 6,700,000, and 30,900,000 related page links through Yahoo (2010), and 4,800,000, 1,790,000, and 9,270,000 related page links through Google (2010). Those links include the URLs of WMS, WFS, and WCS, papers and news about WMS, WFS, and WCS. Because there are so many results, some problems must be faced:

(1) The URLs are too numerous to show which are the effective (real OGC web service) services about the OGC Web Service (OWS).
(2) Whether the OWS query precision can be improved to what is needed. As is hoped when using OWS keywords (such as 'WMS,' 'WFS,' and 'WCS') to search, the results are either OWS links or a greater proportion of OWS links than using keywords method.
(3) Whether a syntax-based query can be developed into a semantics-based query.

Although a number of spatial search engines are sensitive to geospatial scope or names, effective links to spatial information services are not available. So, challenging problems are how to retrieve OWS services as opposed to others and provide semantic querying.

In order to solve the above problems, this paper mainly focuses on three aspects: (1) enriching the syntactic method of OWS discovery and improving its precision. (2) Building a bridge between a syntax-based search engine and a semantic search engine. A semantic search engine has at least two properties. One is that the crawling content is a semantic document or file; the other is the crawling content is text content, and extracts information to create a semantic index/document/file. This paper focuses the latter for that semantic document/file is not common on the Web now especially for OWS, but OWS XML document or potential document is common. (3) Creating a semantic index/document/file from keyword-based steps, and providing semantic queries of OWS with parameters in which users are always interested, for example service type, service name, and spatial range. This paper improves our previous paper (Chen *et al*. 2007) by proposing a method for retrieving OWS based on page crawling, link detection, service capability matching, and ontology reasoning. The paper contains the following sections: Section 2 introduces previous related work, including the current status of geographic ontology, geospatial information about the semantic web, and spatial search technology. Section 3 expounds the method of OWS discovery, including system architecture, components, and the realization of those components. Section 4 presents some experiments to show the precision and efficiency of the OWS search engine and reasoning-based query. Section 5 shows how this method is used in the Antarctic multi-protocol OWS portal. Finally, the conclusions of this paper are presented and future work discussed.

2. Related work

SDI portals like Geospatial One-Stop (GOS) and Infrastructure for Spatial Information in Europe (INSPIRE) have done some work to discover OWS with a syntactic method.

GOS is one of 24 E-Government initiatives sponsored by the Federal Office of Management and Budget (OMB) to enhance government efficiency and to improve citizen services (GOS 2010). INSPIRE establishes an infrastructure for spatial information in Europe to support Community environmental policies, and policies or activities which may have an impact on the environment (INSPIRE 2010). Those two portals are supporting OWS to enhance geo-data and resource sharing and interoperation. Using portals, geospatial information can be accessed easily and conveniently; moreover, the time and effort to find geospatial data are reduced. However, despite the advantages of these search portals, they rely on metadata and do not support formal semantics.

Ontology in the field of information technology is usually considered 'an explicit specification of a conceptualization' (Gruber 1993). Ontology-enhanced information retrieval has been developed. There are three well-known projects, Ageni (Vega *et al*. 1998), Ontobroker (Ontobroker 2010), and SKC (SKC 2010) imposing ontology onto information retrieval. Ageni aims to help users find the needed ontology existing in the World Wide Web (WWW), mainly applying a reference ontology that is built on the basis of ontology existing in the WWW, saving all kinds of ontological metadata. Ontobroker is oriented to the resources in the WWW, with the purpose of retrieving the web pages which contain the content the user needs. SKC is an ongoing project, which aims to solve the heterogeneous semantics problems in information systems, to achieve interoperability between self-governing heterogeneous systems. But those projects rarely consider geo-ontology about geographical data, metadata, and services. Geo-ontology projects, such as NASA's Earth and Environmental Terms Semantics (SWEET) (Raskin 2009), NSF's GIS metadata ontology (Islam *et al*. 2003), and the semantic web-based geography knowledge discovery of NGA (Di *et al*. 2006, Zhao and Di 2006), have been tried in the USA and Europe to describe the semantics of the data sets and scientific concepts. However, metadata ontology for geospatial information services is still evolving, and the ontology definition for the OGC services is still deficient.

Meanwhile, a semantic approach is also used to find geospatial data and services. Wiegand and García (2007) propose a task-based and semantic web approach to find geospatial data. The purpose of the project is to improve data discovery and facilitate automatic retrieval of data sources. The system formalizes the relationships between types of tasks, including emergency responses, and types of data sources needed for those tasks. Domain knowledge, including criteria describing data sources, is recorded in an ontology language. However, this paper pays more attention to ontology, which is created based on metadata that should be already known. Discovering suitable geo-processing services is now a major challenge. Current (keyword-based) approaches to service discovery are inherently restricted by the ambiguities of natural language, which can lead to low precision and/or recall. To alleviate these problems, Lutz (2007) proposed using an ontology-based approach based on two ideas for geographic information service discovery. *Ontologies describing geospatial operations* are used to create descriptions of requirements and

service capabilities; matches between these descriptions are identified based on *function subtyping*. But this paper focuses on geo-processing services, and discovery service relies on the inputs of the data source. Zhang *et al*. (2010) proposed a framework for a geospatial semantic web-based spatial decision support system (SDSS) for Digital Earth. In this framework, heterogeneous ontology integration, ontology-based catalog service, and web service composition were introduced. The proposed interoperable SDSS enables decision-makers to reuse and integrate geospatial data and geo-processing resources from heterogeneous sources across the Internet. This paper, however, mainly discusses ontology and the geo-process of ontology itself in its framework.

The Web Ontology Language for Services (OWL-S; OWL-S 2004) describes the web services and makes them intelligent. Zaharia *et al*. (2009) has studied the implementation of geospatial web services that meet the semantics requirement. At present, some use OWL-S to describe the semantics of spatial information services. For example, Chen *et al*. (2006) have given a framework for spatial information web services based on semantics, Yue *et al*. (2008) have studied the automatic conversion from a description of an OGC standards-compliant geospatial web services chain of OWL-S to the BPEL description, and Jing *et al*. (2005) have studied the OWL-S services model and the upper-level ontology and the framework of the geospatial semantic services. Our work refers to those OWL-S works.

Many semantics-enhanced or semantic search engines have been developed. Dong *et al*. (2008) classify semantic search technologies into six main categories. In those six categories, they detail several semantic search engines (Chiang *et al*. 2001, Guha and McCool 2003, Lee and Tsai 2003, Liu *et al*. 2003, Bhagwat and Polyzotis 2005) and hybrid semantic search engines (Rocha *et al*. 2004, Han and Chen 2006, Kandogan *et al*. 2006), but all those search engines rarely consider geographical scope, geographical services, and OWL-S. Up to now, geographical scope and geographical names database matching have been used mainly in spatial data searching, and are sensitive only to the contents of the connecting HTML pages (Bai and Yang 2004), not to the XML pages about the spatial information service and its ability.

Chen *et al*. (2007) have done some work on high precision WMS discovery and the proposed architecture of high precision WMS retrieval. They proposed a method for retrieving WMS using an extended search engine and service capabilities match. It is a try on OWS discovery and there is still some work to do, as extends from WMS to WMS, WFS, and WCS. Our paper is improved over that of Chen *et al*. (2007): it extends retrieval of OWS from WMS to WMS, WFS, and WCS; extends WMS ontology to OWS ontology; and adds a reason-based semantic query.

3. Methodology

3.1. System architecture

This section discusses design strategies for an OWS information service. There is a set of criteria for compliance with our design. (1) Features based on distributed service-oriented architecture, communicating the components by interfaces and protocols and deploying flexibly. (2) Ability to handle different OWS service versions. (3) Machine and platform independence, to allow for worldwide use on the internet.

As Figure 1 shows, the architecture of the proposed high precision OGC web information service discovery and retrieval system has six core components: *Distributed OWS, OWS Search Engine, OWS Ontology Generator, Ontology-based OWS Catalogue Service, Reasoning-based Query Engine,* and *Multi-protocol OWS Client.*

The *Distributed OWS* is the distributed OGC geospatial information service on the internet (such as WMS, WFS, and WCS). It is the service source of the search engine for retrieval.

The *OWS Search Engine* is the core component in the architecture. It discovers the OWS service by crawling page links, querying with OWS keywords, detecting OWS page links, getting and combining effective OWS page links, and storing the results in a descriptive file.

The *OWS Ontology Generator* is responsible for generating OWL-S instances from OWS capabilities and type information. OWL-S instances are stored in files or a database. Before querying, they are registered in the ontology-based OWS catalog service. It transforms OWS capabilities information into OWL-S instances using Extensible Stylesheet Language Transformations (XSLT).

The *ontology-based OWS catalog service* is responsible for registering, managing, and querying OWS services. The ontology instances from OWS are auto-registered into a catalog service using the ebRIM catalog implementation specification and semantic web technology (Yue *et al*. 2006).

The Reasoning-based Query Engine uses a reasoner to query OWS with the parameters sent by the user. This query is executed by the reasoner.

Multi-protocol OWS client is the interface layer for users. Its functions include dealing with user requests, querying OWS services from CSW, getting an OWS

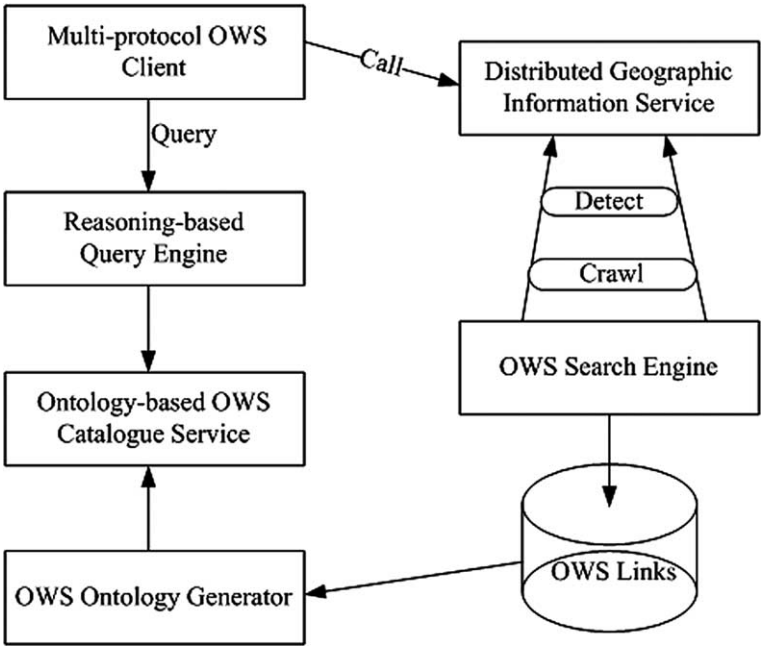

Figure 1. Architecture of high precision OWS discovery and retrieval.

service or services, and visualizing OWS services (for example, showing the image result of the getMap operation of WMS, the feature result of the getFeature operation of WFS, or the coverage result of the getCoverage operation of WCS).

3.2. Components and their realization

3.2.1. OWS search engine

The OWS search engine was developed from a keyword-based search engine. Capability matching and link detection for discovering OWS services were added.

3.2.1.1. OWS capabilities. The XML response documents are OWS capabilities when a 'GetCapabilities' operation is sent. The root elements are either 'WMT_MS_Capabilities' from version 1.0.0 to 1.1.1 or 'WMS_Capabilities' (version 1.3.0) of WMS, 'WFS_Capabilities'(from version 1.0.0 to 1.1.0) of WFS, and 'WCS_Capabilities' (version 1.0.0) or 'Capabilities' (version 1.1.0). All those root elements have 'version' and 'updateSequence' attributes. There are 'Layer' elements in WMS_capabilities, 'Feature' elements in WFS_capabilities, and 'Coverage' elements in WCS_capabilities. All OWS includes the mandatory 'Name,' 'Title,' 'Abstract,' keywords, and bounding box information. The above attributes and elements are adopted by the following OWS capability detection.

3.2.1.2. Method. Figure 2 shows the six procedures of OWS-extended discovery. The procedures are as follows:

(1) Crawl: use a popular open source search engine (such as Nutch) to track every crawled page and its related link, and generate the URL database (A) from the specified URL links.
(2) Query: given the web content, the keywords 'WMS,' 'Web Map Service,' 'WFS,' 'Web Feature Service,' 'WCS,' and 'Web Coverage Service' are used to query the indexed web content, and then the potential OWS URL database can be found (B).
(3) Parse: the potential OWS URL database has the links whose content has the OWS keywords. The content is parsed using an html document parser, and

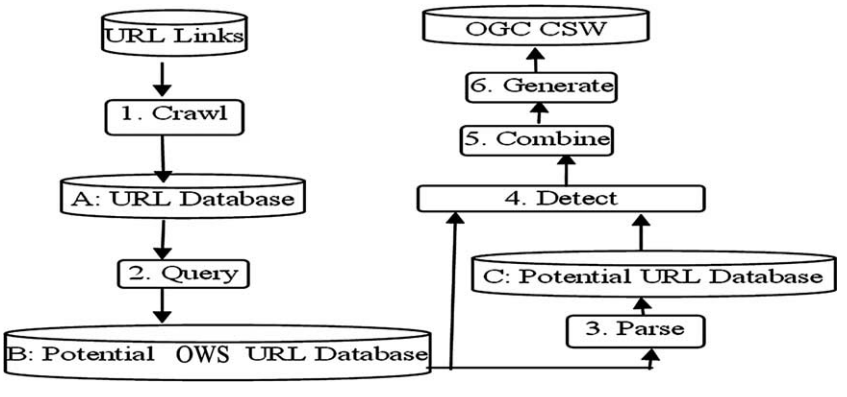

Figure 2. The flow of OWS discovery.

some OWS relevant links can be found and stored in 'Potential URL Database' (C).

(4) Detect: A get is sent or a 'GetCapabilities' request posted to the OWS links and the response obtained. The URL and metadata of OWS can be obtained if the response contains the capabilities.
(5) Combine: the OWS URLs are compared with each other, and a uniform OWS URL database is generated.
(6) Generate: each OWS service is registered as a service ontology record in OGC CSW; the record is generated through the capability document of OWS.

3.2.2. OWS ontology generator

3.2.2.1. Auto-build OWL-S of OWS. OWL-S instances of OWS are built by the OWS capabilities document and the relationship of OWS and OWL-S. The method is as shown in Figure 3.

The OWS URL is the link address of the OWS service. It can acquire OWS capabilities by sending a 'GetCapabilities' request. Analyzing the OWL-S class, the OWS capabilities schema, and the relationship between OWL-S and OWS gives OWL-S.xsl, which is the style sheet to transform an OWS capabilities document into OWL-S instances of OWS. The relationship of OWS and OWL-S is the mapping between them, as in Section 3.2.2.2.

3.2.2.2. Mapping from OWS to OWL-S.. Ontology Web Language for Services (OWL-S) is an OWL-based web service ontology that supplies web service providers with a core markup language for describing the properties and capabilities of their web services in unambiguous, computer-interpretable form. OWL-S markup of web services will facilitate the automation of web service tasks, among them, automated web service discovery, execution, composition, and interoperation. There are three main parts in OWL-S: the service profile for advertising and discovering services, the process model for a detailed description of a service's operation, and grounding, for details on how to interoperate with a service, via messages (OWL-S 2004). OWL-S is web service ontology, which emphasizes the operation of web services, while OWS is open OWSs specifications in which each OWS web service has its own operations to

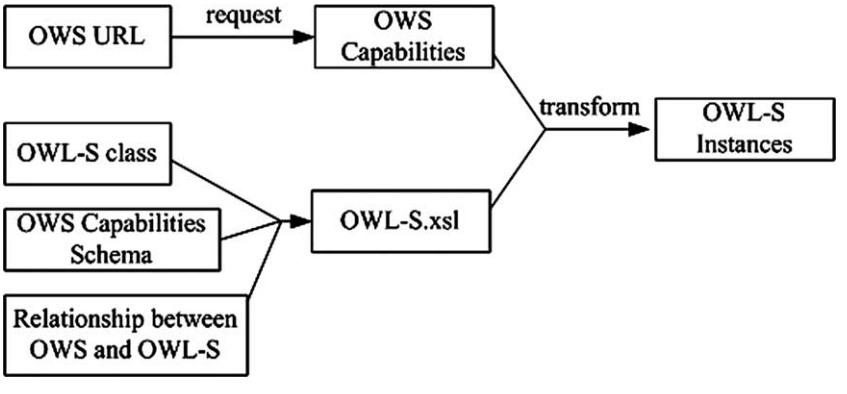

Figure 3. The flow of OWS OWL-S instances construction.

describe its OWL-S by describing its operation. Table 1 shows the mapping from OWS to OWL-S. WMS, WFS, and WCS have the 'GetCapabilities' operation. All the response documents of the 'GetCapabilities' operations have a similar structure, so they can describe their OWL-S by describing their 'GetCapabilities' operation. The main content of the response to a 'GetCapabilities' request is the information about 'serviceIdentification,' 'serviceProvider,' 'MetadataOperation,' 'Layer,' 'Feature,' and 'Coverage.' Some elements of 'serviceIdentification' are 'title,' 'name,' 'keywords,' and 'abstract,' and some elements of 'serviceProvider' are 'title,' 'phone,' 'fax,' 'email,' 'address,' and 'webURL.' 'MetadataOperation' concerns the description of all operations of a web service. 'Layer' is the layer information for WMS, 'Feature' is the feature information for WFS, and 'Coverage' is the coverage information for WCS. 'Layer,' 'Feature,' and 'Coverage' all have the information about 'title,' 'name,' 'keyword,' 'abstract,' and 'BBox.' 'BBox' is a class that shows spatial range and records the lower and upper corner points of a rectangular space; it also has coordinate system information. When OWL-S describes these elements of OWS it is said to be the OWL-S instance of OWS. The request parameters and return results of the 'GetCapabilities' operation map to the 'input' and 'output' of 'Process' of OWL-S. Because the request in any 'GetCapabilities' operation can be composed of three parameters, 'request,' 'version,' and 'service'; the 'input' of 'Process' maps to these three parameters. The 'output' of 'Process' maps to 'GetCapabilitiesOutput.' The URL of the web service maps to the 'serviceName' of 'Profile.' The Service types of WMS, WFS, and WCS map to the 'categoryName' of 'Profile.' The outputs 'Layer,' 'Feature,' and 'Coverage' map to three new separate classes of Layer, Feature, and Coverage. Those three classes are instances of 'sParameter' of 'Profile.' Class Layer, Feature, and Coverage have data attributes 'title,' 'name,' 'keyword,' and 'abstract,' and they associate with the BBox class through the 'hasBBox' object properties. In all the ways detailed above, OWS maps to an instance of OWL-S.

3.2.3. Reasoning-based Query Engine

A reasoning-based Query Engine is responsible for querying an OWS service through a rule-based reasoner. The new system uses Jena to query OWL-S instances of OWS. Jena, which is a Java framework for building semantic web applications, is open source, developed by the HP Labs Semantic Web Program. It provides a programmatic environment for RDF, RDFS, OWL, and SPARQL and includes a rule-based inference engine. There are many query classes in Jena, such as 'OntClass,' 'subclass,' 'listInstances,' and 'superClass.' Also, there are many properties such as 'OntProperty,' 'ObjectProperty,' 'DatatypeProperty,' 'subProperty,' 'superProperty,' 'domain,' and 'range.' Those programs can get the direct result of class, instance, and property but not indirect class, instance, and property, so to solve this problem we use the rule-based reasoner of Jena.

3.2.3.1. Model of reasoning-based query. The aim of query OWS is to acquire the URL of the OWS. There are three aspects to building a query sentence: service type (e.g. WMS, WFS, and WCS), spatial range, and name. The model of reasoning-based OWL-S query of OWS is as shown in Figure 4.

Table 1. Mapping OWS to OWL-S.

OWS element/parameters/content		OWL-S element/instance	
request, version, service (OWS GetCapabilities request parameters)		Process/input	
GetCapabilitiesOutput (OWS GetCapabilities response)		Process/output	
OWS service URL		Profile/serviceName	
'WMS,' 'WFS,' 'WCS' (service types)		Profile/categoryName	
WMS	Layer/title	Layer (a class of the instance of Profile/sParameter)	Layer/title
	Layer/name		Layer/name
	Layer/keyword		Layer/keyword
	WMS/Layer/abstract		Layer/abstract
WFS	FeatureTypeList/FeatureType/title	Feature (a class of the instance of Profile/sParameter)	Feature/title
	FeatureTypeList/FeatureType/name		Feature/name
	FeatureTypeList/FeatureType/keywords		Feature/keyword
	FeatureTypeList/FeatureType/abstract		Feature/abstract
WCS	Contents/CoverageSummary/title	Coverage (a class of the instance of Profile/sParameter)	Coverage/title
	Contents/CoverageSummary/title		Coverage/name
	Contents/CoverageSummary/keywords		Coverage/keyword
	Contents/CoverageSummary/abstract		Coverage/abstract
WMS	Layer/BoundingBox@SRS	BBox (Layer, Feature, and Coverage associate with BBox class with hasBBox object properties)	Srs
	Layer/BoundingBox@minx		lowerCornerX
	Layer/BoundingBox@maxx		upperCornerX
	Layer/BoundingBox@miny		lowerCornerY
	Layer/BoundingBox@maxy		upperCornerY
WFS	'WGS84'	BBox (Layer, Feature, and Coverage associate with BBox class with hasBBox object properties)	Srs
	FeatureTypeList/FeatureType/WGS84BoundingBox/LowerCorner		lowerCornerX
	FeatureTypeList/FeatureType/WGS84BoundingBox/UpperCorner		upperCornerX
	FeatureTypeList/FeatureType/WGS84BoundingBox/LowerCorner		lowerCornerY

Table 1 (*Continued*)

OWS element/parameters/content		OWL-S element/instance	
	FeatureTypeList/FeatureType/ WGS84BoundingBox/ UpperCorner		upperCornerY
WCS	'WGS84'	BBox (Layer, Feature, and Coverage associate with BBox class with hasBBox object properties)	Srs
	Contents/ CoverageSummary/ WGS84BoundingBox/ LowerCorner		lowerCornerX
	Contents/ CoverageSummary/ WGS84BoundingBox/ UpperCorner		upperCornerX
	Contents/ CoverageSummary/ WGS84BoundingBox/ LowerCorner		lowerCornerY
	Contents/ CoverageSummary/ WGS84BoundingBox/ UpperCorner		upperCornerY

The classes are 'Profile,' 'ServiceName,' 'ServiceParameter,' 'Layer,' 'Feature,' 'Coverage,' and 'BBox.' All those classes are related by the ObjectProperties, which are shown in Figure 4; for example, 'Profile' has an ObjectProperty 'serviceParameter' to connect to the 'ServiceParameter' class. A query with service type is equivalent to querying the content of 'serviceName' with the content of 'categoryName'; a query with spatial range is equivalent to querying the content of 'serviceName' with the content 'BBox'; a query with name is equivalent to querying the content of 'serviceName' with the content 'title'; a compound query is equivalent to querying the content of 'serviceName' with the content 'categoryName,' 'BBox,' and 'title.'

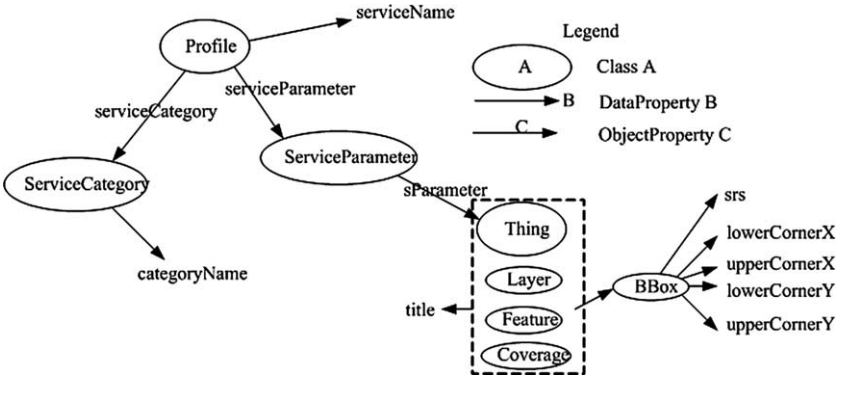

Figure 4. Model of reasoning-based OWS OWL-S query.

3.2.3.2. Method of reasoning-based query. According to reasoning query rule of Jena, the steps to query OWS, are as in Figure 5.

'Write Rules' writes the rules for a reasoning query; 'Create Model' creates 'Model'; 'Read data' reads the data that will be queried; 'Bind reasoner' binds rules to a kind of reasoner; 'Get InfModel' obtains the information model of the reasoning query; 'Get Result' obtains the results of the reasoning query. Figure 6 is an example program using service type to query results with the steps of the reasoning query.

In this program, lines 4–6 are 'Write Rules,' lines 7 and 8 are 'Create Model,' line 9 is 'Read data,' line 10 is 'Bind reasoner,' line 11 is 'Get InfModel,' and the rest is 'Get Result.' The key step in Figure 5 is 'Write Rules.' Some important reasoning rules are supported by Jena reasoners:

Rule a: relationship between 'Profile' and 'categoryName'

'[r1: (?a
http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceCategory?b) (?b
http://www.daml.org/services/owl-s/1.1/Profile.owl#categoryName?c)->(?a
http://swe.whu.edu.cn/ows.owl#t1?c)]';

Rule b: relationship between 'Profile' and 'Thing'

'[r1: (?a http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceParameter?c)(?c http://
www.daml.org/services/owl-s/1.1/Profile.owl#sParameter?d)->(?d http://swe.whu.edu.cn/
ows.owl#t1?b)]';

Rule c: relationship from 'Layer,' 'feature,' and 'Coverage' to 'BBox'

'[r1:(?a

http://swe.whu.edu.cn/ows.owl#hasBBox?b)->(?a
http://swe.whu.edu.cn/ows.owl#t1?b)]';

Rule d: relationship from 'Layer,' 'feature,' and 'Coverage' to 'title'

'[r1:(?a

http://swe.whu.edu.cn/ows.owl#title?b)->(?a
http://swe.whu.edu.cn/ows.owl#t1?b)]';

Rule e: relationship between 'categoryName' and 'Thing'

'[r1: (?a

http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceParameter?b)(?b
http://www.daml.org/services/owl-s/1.1/Profile.owl#sParameter?c) (?a
http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceCategory?d) (?d
http://www.daml.org/services/owl-s/1.1/Profile.owl#categoryName?e->(?c
http://swe.whu.edu.cn/ows.owl#t1?e)]';

A query with service type, spatial range, title or their compounds uses the rules above as in Figure 7.

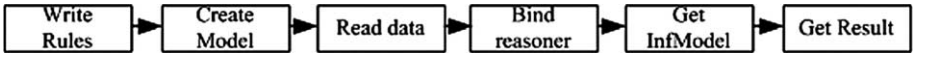

Figure 5. Steps of reasoning query.

```
public static ArrayList categoryName2serviceName(String categoryName)
    {
        ArrayList serviceNameArrayList = new ArrayList();
        String rules = "[r1:(?a http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceCategory ?b)
        (?b http://www.daml.org/services/owl-s/1.1/Profile.owl#categoryName ?c)->
        (?a http://swe.whu.edu.cn/ows.owl#t1 ?c)]";
        OntModel m = ModelFactory.createOntologyModel(OntModelSpec.OWL_MEM, null);
        addAltPaths(m);
        m.read("http://swe.whu.edu.cn/OwsProfile.owl");
        com.hp.hpl.jena.reasoner.Reasoner reasoner = new GenericRuleReasoner(Rule.parseRules(rules));
        InfModel infmodel = ModelFactory.createInfModel(reasoner, m);
        com.hp.hpl.jena.rdf.model.Property b = infmodel.getProperty("http://swe.whu.edu.cn/ows.owl#t1");
        for(StmtIterator itr = infmodel.listStatements(null, b, null); itr.hasNext();)
        {
            Statement stm = (Statement)itr.next();
            Resource t1 = stm.getSubject();
            RDFNode t3 = stm.getObject();
            if(t3.toString().equals(categoryName))
            {
                Individual ind = m.getIndividual(t1.toString());
                com.hp.hpl.jena.rdf.model.Property pro = m.getProperty("
                http://www.daml.org/services/owl-s/1.1/Profile.owl#serviceName");
                RDFNode val = ind.getPropertyValue(pro);
                if(val.isLiteral())
                    serviceNameArrayList.add(val.toString());
            }
        }

        return serviceNameArrayList;
    }
```

Figure 6. An example of reasoning query.

(1) Query with service type: first get the instances of 'Profile' and 'categoryName' with Rule a. Given the service type, the instance of 'Profile' can be obtained and then the content of the data property 'serviceName' of 'Profile,' which is also the URL of the web service.
(2) Query with title: first, get the instances of 'title' and 'Profile' with Rule c and Rule b. Given the title, obtain the instance of 'Profile' and then the URL of web service.
(3) Query with spatial range: first, get the instances of 'BBox' and 'Profile' with Rule d and Rule b. Given BBox (the spatial range), obtain the instance of 'Profile' and thus the URL of the web service.
(4) Query with service type and title: first get the instances of 'categoryName' and 'Thing' with Rule e and then obtain the title instances as specified by the

Query parameter	rule	result
service type	Rule a	OWS link
spatial range	Rule d , Rule b	
title	Rule c , Rule b	
service type , spatial range	Rule e , Rule c , Rule a	
service type , title	Rule e , Rule a	
spatial range , title	Rule c , Rule b	
service type , spatial range , title	Rule e , Rule c , Rule a	

Figure 7. Different query parameters use different rules.

instance of 'Thing.' Secondly, acquire the instance of 'Profile' with Rule a, and thus the URL of the web service.

(5) Query with service type and spatial range: first, obtain the instances of 'categoryName' and 'Thing' with Rule e and then get the BBox instances as specified by 'Thing.' Secondly, obtain the instance of 'Profile' with Rule b and then the URL of the web service.

(6) Query with spatial range and title: first, get the instances of 'BBox' and 'Thing' with Rule c and then obtain the title instances as specified by the data property of the instance of 'Thing.' Secondly, get the instance of 'Profile' with Rule b and then the URL of the web service.

(7) Query with service type, spatial range, and title: first, obtain the instances of 'categoryName' and 'Thing' with Rule e and then get the title instances as specified by the data property of the instance of 'Thing.' Second, obtain the instance of 'title' with the instances of 'Thing' and Rule c, and then get the instance of 'Profile' with Rule a and thus the URL of the web service.

4. Experiments

To verify the proposed crawl, detect, and query techniques for links, some experiments were carried out. All experimental data is from the Scientific Committee on Antarctic Research (SCAR)-SITE (http://www.scar.org/). All experiments were on a personal computer with two 2.66 GHz processors and 4.0 GMB of memory, running Microsoft Windows 7. The results are in Table 2. The real number of OWS links in SCAR-SITE is 5.

The results of Table 2 are the search results for OWS retrieval by the OWS Search Engine. Each part compares the search results obtained by the OWS Search Engine when the 'detect' operation is switched on with those for "detect" off. When the 'detect' operation is off, there is only the 'crawl' operation which performs only a textually based search. However, when the 'detect' operation is on, the OWS Search Engine will perform the crawl, query, parse and combine operations mentioned in Section 3.2.1.2. In this experiment, the OWS Search Engine uses Nutch (an open keyword-based search engine) for 'crawl' operation and a detect module for 'detect' operation. In Table 2, 'depth' is the crawl depth of the web site; 'topN' is the top (first obtained) results that will be selected to store for each depth; 'T(s)' is the time required for the crawl or detect operation (s, short for seconds, is the units), and

Table 2. Compares the results of the crawl and detect operations on SCAR-SITE.

		Crawl							Detect								
			WMS		WFS		WCS		WMS			WFS			WCS		
Depth	topN	T(s)	H	E	H	E	H	E	T(S)	H	E	T(S)	H	E	T(S)	H	E
2	100	143	1	0	1	0	1	0	174	60	0	143	1	0	143	1	0
4	100	3418	32	0	32	0	13	0	3788	326	1	3659	232	1	3418	13	0
6	100	6245	84	0	60	0	19	0	6883	641	5	6590	338	5	6396	166	0
8	100	5467	102	0	76	0	24	0	6196	660	5	5716	356	5	5594	209	0
10	100	5334	102	0	76	0	24	0	6172	660	5	5601	356	5	5459	209	0

detect depends on crawl; 'H' is the number of OWS keywords contained for the crawl operation and the number that will be detected for the detect operation parsed from the number of crawl operations; it is also called hits; 'E' is the number of effective OWS links.

4.1. Precision analysis

Precision (P) in this paper is defined as the ratio of the effective number of OWS links (E) queried by a method and the real OWS links (R) the website contains ($P = E/R$). First the precision as a function of depth is studied. Figure 8 shows the results.

Those for WMS and WFS are the same, and both can be shown in this figure. The more depth a detect operation has, the higher precision it has. However, when the depth reaches a certain number, 6 in the example, the precision tends to stabilize and finally tends to 1 (100%). Meanwhile, the precision of crawl is very low and seems insensitive to the crawl depth. Capability match-based detection allows retrieval of links which describe OWS not only in the Crawl database but also in the corresponding page document content like 'http://.' Since these links are similar to OWS requests, the retrieved documents are the OWS relevant to the query. When the 'Detect' option is off, it appears that all retrieved links involve the OWS link. Unfortunately, many of these links are not actually OWS links. Detect precision is not 100% at low crawl depths, such as 2 in this experiment, because when the crawl depth is very low, some of the OWS links in the web pages below the crawl depth will not be detected. Figure 9 shows the precision of crawl and detection operations considering only the real OWS links in those crawled web pages.

Figure 8 and 9 show that a capability match-based detection method has a higher precision.

4.2. Analysis of time required

Figure 10 shows that more CPU time is required to execute retrieval using 'Detect' than using 'Crawl.'

This difference is due mainly to the time required for the matching capability in 'Detect.' From the total mean response time, 'Detect' is about 4642.6 seconds for WMS and 4341.8 for WFS, and 'Crawl' is about 4121.4 seconds for both WMS and WFS; the 'Detect' time is 1.13 times that of 'Crawl' for WMS and 1.05 times for WFS. From the mean response time per hit shown in Figure 11, 'Detect' is

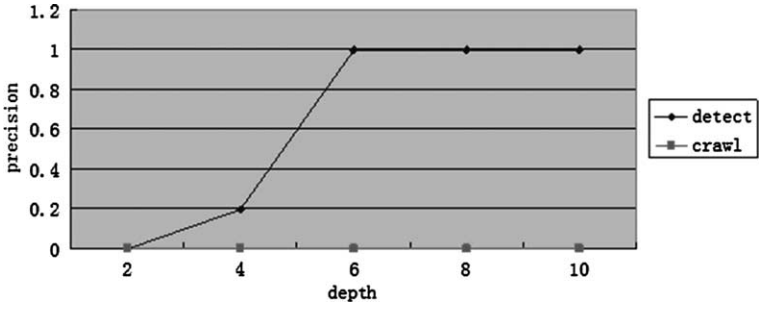

Figure 8. The WMS discovery precision of Crawl and Detect.

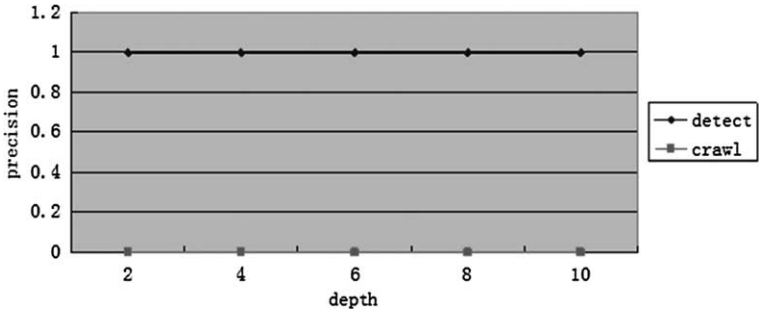

Figure 9. The WFS discovery precision of Crawl and Detect.

about 8.8 seconds and 'Crawl' is about 86.01 seconds for WMS, so the cost time of 'Detect' is 0.10 times of that of the 'Crawl' for WMS; Meanwhile, 'Detect' is about 42.0 seconds and 'Crawl' is about 99.2 seconds for WFS, so 'Detect' is only 0.42 times of that of the 'Crawl' for WFS. The average is as low as 0.26 $((0.10 + 0.42)/2 = 0.26)$ times.

We then studied the time required for crawl or detect with different depths, i.e. crawl composed of different numbers of depths. Figure 10 shows the result. The response time increases with the crawl depth: the more depth a crawl has, the more CPU time is required for detection. The increase displays a linear tendency and then tends to stabilize. This is mainly due to the hits; the more potential the 'WMS' link has, the more CPU time is required for detection.

The time required for detect as a function of number of hits was studied. The result is shown in Figure 10. Response time increases with the number of hits; the more hits a crawl has, the more CPU time is required for detect, and the increase displays a linear tendency.

4.3. The analysis of reasoning-based query

Table 3 compares results of crawl, detect, and reasoning-based queries. In Table 3, the item names have the same meaning as the ones in Table 2. The numbers in 'H' of crawl and detect operation are the results obtained using OWS keywords 'road' to query. The reasoning-based query queries the Ontologies built on the effective OWS links obtained by the detect operation. 'H' in this item is the OWS link numbers, 'E' is the number of OWS links containing the 'road' keyword.

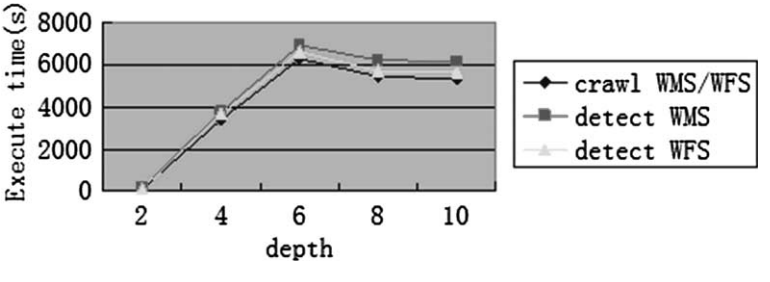

Figure 10. The total response time of Crawl and Detect.

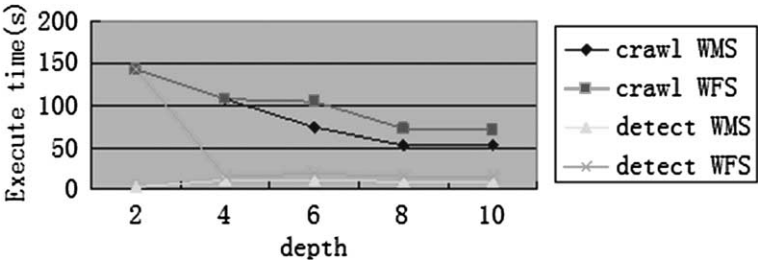

Figure 11. The mean response time per hit of Crawl and Detect.

4.3.1. Precision comparison

Table 3 shows the following:

(1) Even though some web pages contain query keywords, it is very hard to find effective OWS links events using keyword-based queries in a keyword-based search engine. In this experiment, 2–7 web links contain the keywords 'road,' 'WFS,' and 'web map service' for crawl depths from 2 to 10. At the same crawl depth, 1–5 web links contain the keywords 'road,' 'WMS ', and 'web feature service.' Unfortunately, there is no effective OWS link in those web links. This may be because the keyword-based query is only text sensitive. When the OWS link text description is 'invisible' in the web page, it can't find the OWS link.

(2) A reasoning-based query in a reasoning-based search engine easily obtains results with higher precision. Capability match-based detection has higher precision in obtaining OWS links. In fact, after detection, the resulting reasoning-based search engine data source links are real OWS links. A reasoning-based query in this search engine is needed only to find the results and not whether the links are OWS links. As in the table, in a reasoning-based query item, the 'H' links are OWS links, needed only to judge those links by a criterion. From the table, the precision of crawl and detection is zero, but of reasoning-based query is 100%.

Table 3. Results of crawl, detect, and reasoning query operations on SCAR-SITE.

		Crawl				Detect				Reasoning-based query			
		Wms		wfs		wms		Wfs		wms		Wfs	
Depth	topN	H	E	H	E	H	E	H	E	H	E	H	E
2	100	0	0	0	0	0	0	0	0	0	0	0	0
4	100	2	0	1	0	2	0	1	0	1	1	1	1
6	100	7	0	3	0	7	0	3	0	5	5	5	5
8	100	10	0	5	0	10	0	5	0	5	5	5	5
10	100	10	0	5	0	10	0	5	0	5	5	5	5

The conclusion of this discussion is that a reasoning-based search engine has higher precision then a keyword-based search engine.

4.3.2. Feasibility comparison

Feasibility in this paper means whether it is possible to query OWS through a search engine with multi-condition query methods. A simple query method uses spatial range as the query parameter.

The ontology reasoning-based search engine proposed in this paper shows better feasibility than a keyword-based search engine. This is because the latter index deals with terms (the keywords of this paper); a query leads to a search by looking for similar terms'. So a keyword-based method is restricted mainly to a term query. But the former is very different. It indexes the terms, but also represents the concepts of these indexed terms and their relationship. When querying, the search engine computes the semantic relation between concepts, not only the terms' similarity. Section 3.2.3.2 shows that a reasoning-based search engine can provide service type, title, spatial range, and their combination. Using the spatial range (–60,–57,–61,–63), which is the location of King George Island in Antarctica, to query two search engines separately, the keyword-based search engine can't find any result, but the reasoning-based search engine can find results.

All the experiments show that capability match-based detection has higher OWS retrieval precision, and a semantic search engine based on this detection method has a higher precision than a keyword-based search engine.

5. A practical study

This method has advantages for OWS discovery and retrieval in the internet. It has been used in practice for a project called the Geo-Information Web Service Portal of Polar.

5.1. Study area

Antarctica plays a key role in the study of numerous scientific questions, many of which are related to global climate change. In most of this research, the spatial component is crucial. Location is fundamental to field data collection and management and is a key to supporting advanced data mining in and across extensive spatially enabled databases. The Antarctic Spatial Data Infrastructure (AntSDI), sponsored by the SCAR Standing Committee on Antarctic Geographic Information (SC-AGI), is responsible for Antarctic spatial data maintenance and sharing through the application of OGC specifications. However, AntSDI faces the challenge of coordinating access to diverse information resources and services created by many organizations and initiatives. As a result, users face difficulties in locating suitable information and services from these sources. The Chinese Antarctic Center of Surveying and Mapping has developed a project called Geo-Information Web Service Portal of Polar to deal with the problems outlined above. Its aim is to integrate all the data and data services about Antarctica. It uses the common interface implemented to support multiple types of geospatial web services (WMS, WFS, WCS) from different implementers supporting the OGC specifications.

Through the portal, international polar spatial information services can be published, registered, found, invoked, and integrated. Users can use the portal to find and access distributed polar spatial data and data services through any compatible registered service. Such services include, among others, services from the Chinese polar spatial database using WMS and WFS, the Canadian cyber atlas of Antarctica using WMS, the German King George Island spatial data using WMS and WFS, UK's ADDI data using WFS, the Australian Antarctic Data Centre holdings using WMS, and the USA's USGS polar database using WMS.

5.2. Results

There are two important aspects to the implementation of the Geo-Information Web Service Portal. One is the portal website page and the other is the data services registered in the data center for this portal as the service server. The portal website page is developed by Java Server Page (JSP) as shown in Figure 12.

The service server is the key to this portal. According to the architecture of Figure 1, several steps are required to implement this service server. The SCAR website http://www.scar.org/ is used to show those steps.

(1) Collect all the web resources about Antarctic data services. For example, get the resource URL http://www.scar.org/, and configure the search engine.
(2) Crawl and detect the effective URLs about Antarctic data services. First, using the OWS search engine, search the website http://www.scar.org/ (depth

Figure 12. Demo of Polar geo-information web service portal.

is 20, and topN is 100). Then, using 'WMS,' 'WFS,' and 'WCS' as separate keywords to query, there are, respectively, 30, 30, and three results. There are 661 potential URL links about WMS, 538 potential URL links about WFS, and 57 potential URL links about WCS. Detect all those URL links and obtain five effective URL links:

http://www.add.scar.org:8080/geoserver/wms

http://www.kgis.scar.org/cgi-bin/kgis_wms

http://www.kgis.scar.org:7070/geoserver/wms

http://www.add.scar.org:8080/geoserver/wfs

http://www.kgis.scar.org:7070/geoserver/wfs

(3) Convert all the effective data services to ontology instances and then register them at the data center. Use the five effective URL links in Step 2 and OWL-S.xsl to auto-build OWL-S instances as shown in the flow of Figure 3. There are three files about the classes of OWL-S (Service.owl, Process.owl, and Profile.owl) and each effective OWS URL can build instances into three.owl files mapping to these three OWL-S classes. Instances of Profile contain the contents of 'serviceName,' 'textDescription,' 'contactInformation,' 'input,' 'output,' 'serviceParameter,' 'serviceCategory.' Instances of Service contain the contents of 'profile,' 'process,' 'WsdlGrounding.' Instances of Process contain the contents of 'name,' 'input,' and 'output.'

After carrying out these three steps, the portal can use these effective data services.

6. Conclusions and future work

This paper proposes a new methodology for finding the correct spatially aware web service for retrieving an OWS information service in a heterogeneous distributed environment in geospatial web-based applications, based on link detecting and capability matching. This methodology is better than traditional methods. It is a more flexible approach and has higher retrieval precision, lower retrieval cost, and semantics based query. Some of the significant advantages are as follows.

It is a flexible method and architecture. First, the proposed method is compatible with different versions of different web services. For example, it is compatible with versions 1.0.0–1.3.0 of WMS, and with WFS and WCS. Secondly, it is compatible with different OWS because all the OWS web services are abstracted from the same OWS abstract implementation specification and they have the same contents and operation methods; for example, they all have the 'serviceIdentification,' 'service-Provider,' and 'MetadataOperation' information. Thirdly, it is a flexible deployment method. The search engine adopts service-oriented architecture to package the crawl, detect, and register procedures into the service.

It builds a bridge between a syntax-based search engine and a semantic search engine. It uses a syntax-based search engine to obtain effective OWS, and then builds OWL-S instances of OWS. This search engine can provide a semantic query

interface. Then, the syntax operation of OWS becomes a semantic operation. It is of benefit to integrate web data, information, and services into a semantic web resource.

It is higher precision and lower cost. First, of all the URLs about OWS obtained using a search engine, some are effective and others are not. Capabilities detection finds all the ineffective URLs, so the precision becomes higher. This gives a higher precision from the syntax viewpoint. From the experiment, the precision of the 'detection' strategy is much better than that of the traditional 'crawl.' Second, the effective URLs about OWS may not be the URLs that users want, but by building OWL-S instances, all the web services can be precisely obtained with service type, spatial range, and title reasoning-based queries. This allows higher precision from the semantic viewpoint. Third, the time required for higher precision and better search range is lower. Experiments show that the execution time for the 'detection' strategy only costs 0.26 times that of the traditional 'crawl.'

Future work will be to optimize the OWS Search Engine in the system architecture, and adding an Ontology search module to search Web Ontology resources to integrate syntax-based and semantic-based search.

Acknowledgements

This work has been supported in part by the National Basic Research Program of China (973 Program) under Grant 2011CB707101, by the National Natural Science Foundation of China under Grant 41023001, 41021061, and by the ShenZhen R&D Foundation under Grant CXB200903090023A. We also sincerely thank our colleague, Dr. Barry Schlesinger, for proof reading the manuscript. The authors would like to thank the editors and anonymous reviewers for their valuable comments and insightful ideas.

Notes on contributors

Nengcheng Chen received the B.Sc. degree in Geodesy from Wuhan Technical University of Surveying and Mapping in 1997, the M.S. degree in Geographical Information System from the Wuhan University in 2000, and the Ph.D. degree in Photogrammetry and Remote Sensing from the Wuhan University in 2003. He was a post-doctoral research associate in Center for Spatial Information Science and Systems, George Mason University, Greenbelt, MD from 2006 to 2008. Currently, he is a Professor of geographic information science of the State Key Lab for Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan, Hubei, China. His research interests include Smart Planet, Sensor Web, Semantic Web, Digital Antarctica, Smart City, and Web GIS.

Zeqiang Chen received the B.Sc. degree in Geography from Huazhong Nomal University in 2005, the M.S degree in Geographical Information System from Wuhan University in 2008. He is a Ph.D. candidate in LIESMARS at Wuhan University. He is also a research assistant in CSISS at George Mason University. His current research interests include Semantic Web and Sensor Web.

Chuli Hu received the M.S degree in Geographical Information System from Wuhan University in 2010. He is a Ph.D. candidate in LIESMARS at Wuhan University. His current research interests include Smart Planet and Sensor Web.

Liping Di received the B.Sc. degree in remote sensing from Zhejiang University in 1982, the M.S. degree in remote sensing/ computer applications from the Chinese Academy of Science in 1985, and the Ph.D. degree in geography from University of Nebraska–Lincoln in 1991. He is a Professor of geographic information science and the director of the Center for Spatial

Information Science and Systems of George Mason University. His research interests include remote sensing, geographic information science and standards, spatial data infrastructure, global climate and environment changes, and advanced Earth observation technology.

References

Bai, Y. and Yang, C., 2004. Research on spatial information search engine. *Journal of China University of Mining and Technology*, 33 (1), 90–94.

Baidu, 2010. *Baidu search engine website.* Baidu. Avaliable from: http://www.baidu.com/ [Accessed 7 June 2010].

Beaujardiere, J., 2006. *OpenGIS® Web Map Service implementation specification, version 1.3.0, OGC®06-042*. Wayland, MA: Open Geospatial Consortium.

Bernard, L., 2002. Experiences from an implementation Testbed to set up a National SDI. *In*: M. Ruiz, M. Gould and J. Ramon, eds., *5th AGILE conference on geographic information science 2002*, Palma de Mallorca, 315–321.

Bernard, L., *et al*., 2004. Towards the implementation of the European spatial data infrastructure-getting the process right. *In*: *7th AGILE conference on Geographic Information Science*, 29 April–1May 2004, Heraklion, Greece. Parallel Session 1.1- 'Spatial Data Infrastructure I'.

Bhagwat, D. and Polyzotis, N., 2005. Searching a file system using inferred semantic links. *In: Proceedings of the sixteenth ACM conference on hypertext and hypermedia*, 6–9 September 2005, Salzburg, Austria. DOI:1145/1083356.1083372.

Chen, J., Yang, S., and Li, C., 2006. Implementation of spatial information web services based on ontology. *Journal of Zhejiang University (Engineering Science)*, 40 (3), 376–380.

Chen, N., Gong, J., and Chen, Z., 2007. A high precision OGC web map service retrieval based on capability aware spatial search engine. *In*: *Proceedings of the 2nd international conference on advances in computation and intelligence*, 21–23 September 2007, Wuhan, China. Lecture Notes in Computer Science, 468312007, 558–567.

Chiang, R., Chua, C., and Storey, V., 2001. A smart web query method for semantic retrieval of web data. *Data & Knowledge Engineering*, 38 (1), 63–84.

Di, L., *et al*., 2006. Ontology-driven automatic geospatial-processing modeling based on web-service chaining. *In*: *Proceedings of the sixth annual NASA earth science technology conference*, 27–29 June 2006, College Park, MD (CD-ROM).

Dong, H., Hussain, F., and Chang, E., 2008. A survey in semantic search technologies. *In*: *Proceedings of the 2nd IEEE international conference on digital ecosystems and technologies*, 26–29 February 2008, Phitsanulok, Thailand, IEEE-DEST 2008, 403–408.

Google, 2010. *Google search engine website.* Google. Available from: http://www.google.com/ [Accessed 7 June 2010].

GOS, 2010. *GOS portal website.* Available from: http://gos2.geodata.gov/wps/portal/gos [Accessed 14 Dec. 2010].

Gruber, T., 1993. A translation approach to portable ontologies. *Knowledge Acquisition*, 5 (2), 199–220.

Guha, R. and McCool, R., 2003. TAP: a semantic web platform. *Computer Networks*, 42 (2), 557–577.

Han, L. and Chen, G., 2006. The HWS hybrid web search. *Information and Software Technology*, 48 (1), 687–695.

INSPIRE, 2010. *Website of infrastructure for spatial information in Europe project.* Available from: http://inspire.jrc.ec.europa.eu/ [Accessed 14 Dec. 2010].

Islam, A., *et al*., 2003. *Ontology for geographic information – metadata.* Available from: http://loki.cae.drexel.edu/~wbs/ontology/iso-19115.htm [Accessed 7 June 2010].

Jing, D., Bi, S., and Wu, F., 2005. Geospatial information services on the basis of agent and OWL-S. *In*: *Proceedings of the IEEE international geoscience and remote sensing symposium*, 25–29 July 2005, Seoul, South Korea, IEEE International.

Kandogan, E., *et al*., 2006. Avatar semantic search: a database approach to information retrieval. *In*: *Proceedings of SIGMOD'06 Chicago*, 26–28 June 2006, Chicago, IL.

Lee, W. and Tsai, T., 2003. An interactive agent-based system for concept-based web search. *Expert Systems with Applications*, 24 (4), 365–373.

Liu, D., Shen, M., and Liao, C., 2003. Designing a composite e-service platform with recommendation function. *Computer Standards & Interfaces*, 25 (2), 103–117.

Lutz, M., 2007. Ontology-based descriptions for semantic discovery and composition of geoprocessing services. *Geoinformatica*, 11 (1), 1–36.

OGC, 2010. *OGC compliant products website.* Available from: http://www.opengeospatial.org/resource/products/compliant/ [Accessed 9 September 2010].

Ontobroker, 2010. *Ontobroker website. Ontobroker.* Available from: http://ontobroker.semanticweb.org/ [Accessed 7 June 2010].

OWL-S, 2004. Semantic markup for web service. OWL-S. Available from: http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/ [Accessed 7 June 2010].

Raskin, R., 2009. *Enabling semantic interoperability for earth science data.* Available from: http://sweet.jpl.nasa.gov/EnablingFinal.doc [Accessed 7 June 2010].

Rocha, C., Schwabe, D., and Aragao, M., 2004. A hybrid approach for searching in the semantic web. *In*: *Proceedings of WWW'04 New York*, 17–22 May 2004, New York, NY.

SKC, 2010. *Scalable knowledge composition website.* SKC. Available from: http://infolab.stanford.edu/SKC/ [Accessed 7 June 2010].

Vega, J.C.A., *et al*., 1998. (Onto)agent: An ontology based WWW Broker to Select Ontologies. *In*: *Proceedings of ECAI'98 workshop on app1ication of ontologies and problem-solving methods*, 24–25 August 1998, Brighton, England.

Vretanos, P., 2005. *OpenGIS web feature service implementation specification, version 1.1.0, OGC®04-094.* Wayland, MA: Open Geospatial Consortium, 131.

Whiteside, A. and Evans, J., eds., 2008. *Web coverage service implementation standard, version 1.1.2, OGC®07-067r5.* Wayland, MA: Open Geospatial Consortium, 133.

Wiegand, N. and García, C., 2007. A task-based ontology approach to automate geospatial data retrieval. *Transaction in GIS*, 11 (3), 355–376.

Yahoo, 2010. *Yahoo search engine website.* Yahoo. Available from: http://www.yahoo.com/ [Accessed 7 June 2010].

Yang, C., *et al*., 2008. Distributed geospatial information processing: sharing distributed geospatial resources to support Digital Earth. *International Journal of Digital Earth*, 1 (3), 259–278.

Yue, P., *et al*., 2006. Semantic augmentations for geospatial catalogue service. *In*: *Proceedings of 2006 IEEE international geoscience and remote sensing symposium (IGARSS06)*, 31 July–4 August, 2006, Denver, CO, IEEE.

Yue, P., Gong, J., and Di, L., 2008. Automatic transformation from semantic description to syntactic specification for geo-processing service chains. *In*: *Web and wireless geographical information systems – 8th international symposium*, 11–12 December 2008, Shanghai, China. Lecture Notes in Computer Science, 5373/2008, 50–62.

Zaharia, R., *et al*., 2009. Semantic execution meets geospatial web services: a pilot application. *Transactions in GIS*, 12 (Suppl. 1), 59–73.

Zhang, C., Zhao, T., and Li, W., 2010. The framework of a geospatial semantic web-based spatial decision support system for Digital Earth. *International Journal of Digital Earth*, 3 (2), 111–134.

Zhao, P., *et al*., 2005. Geospatial web service client. *In*: *Proceedings of ASPRS 2005 annual conference*, 7–11 March 2005, Baltimore, MD.

Zhao, P. and Di, L., 2006. Semantic web service based geospatial knowledge discovery. *In: Proceedings of 2006 IEEE international geoscience and remote sensing symposium*, 31 July–4 August, 2006, Denver, CO.