National Spatial Data Infrastructure
2009 Cooperative Agreement Program

Utilizing GOS Map and Data Services for Cross-agency Earth Science and Geospatial Cyberinfrastructure Communities

Interim Project Report

Submitted to:
Mr. Doug Nebert, Program Manager, 2009 NSDI CAP Cat.2, and
Ms. Brigitta Urban-Mathieux, NSDI CAP Coordinator
MS 590, USGS National Center
Reston, VA 20192

Submitted by:
The Joint Center of Intelligent Spatial Computing
George Mason University
4400 University Drive, MS 4C6, Fairfax, Virginia 22030
http://www.cisc.gmu.edu/

Interim Report Summery Information
Date: Nov.23, 2009
Agreement Number: G09AC00103
Project title:
2009 CAP-Category 2: Utilizing GOS Map and Data Services for Cross-agency Earth Science and Geospatial Cyberinfrastructure Communities
Interim or Final report:
Interim Report

Organizations:
Joint Center for Intelligent Spatial Computing (CISC),
George Mason University (GMU),
4400 University Drive, MS 4C6, Fairfax, Virginia 22030
http://www.cisc.gmu.edu

American Association of Geographers &
Federation of Earth Science Information Partners

Project PI: Chaowei Yang, (703)-993-4742, cyang3@gmu.edu

Interim Report

Executive Summary

The project is to develop open-source portlets and a standalone portal client to utilize the GOS map and data services to support Earth and geography science communities. The project is a partnership of George Mason University's joint Center for Intelligent Spatial Computing (CISC), the Federation of Earth Science Information Partnership (ESIP), and the Cyberinfrastructure Specialty Group (CISG) within the American Association of Geographers (AAG).

As of today, the project has finished most of the technical development, which has resulted in a unified discovery and display portal, "Uniportal", enabling searching, query, visualization, and integration of GOS map resources. This portal can be further customized to be embedded as a portlet in other portals or to be used as a standalone client for accessing GOS map resources.

In the upcoming months, we will finalize the software documents, polish the source code for open access, and deploy the service for partner communities.

Project Narrative

Activity I: Develop modules based on expertise and requirements

A prototype Uniportal has been developed and deployed based on the expertise and requirements of partners. The portal is featured by the following modules:

1. Discovery module

The GOS catalog service is an implementation of the OGC Catalogue Services 2.0 specification (CSW ebRIM Profile), edited by Nebert and Whiteside (2005). To use this service, CSW clients may use the following URL: http://gos2.geodata.gov/Portal/csw202/discovery as the service interface URL. In this project we have integrated search and display access to the GOS CSW service.

The GOS CSW service supports three mandatory CSW requests including GetCapabilities, DescribeRecord, GetRecords and also support one optional request GetRecordById. Both KVP (Key-Value Pair) and POST style requests are supported by the GOS CSW service.

1.1 GetCapabilities

GetCapabilities allows CSW clients to retrieve service metadata from a server. The response to a GetCapabilities request is an XML document containing service metadata about the server.

KVP style request:

http://gos2.geodata.gov/Portal/csw202/discovery?REQUEST=GetCapabilities&SERVICE=CSW

&version=2.0.2

POST style request:

```
<csw:GetCapabilities xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" service="CSW"
xmlns="http://www.opengis.net/ows">
</csw:GetCapabilities>
```

1.2 DescribeRecord

DescribeRecord allows a client to discover elements of the information model supported by the target catalogue service. The operation allows some or all of the information model to be described.

KVP style request:

http://gos2.geodata.gov/Portal/csw202/discovery?REQUEST=DescribeRecord&service=CSW&version=2.0.0

POST style request:

```
<csw:DescribeRecord service="CSW" version="2.0.2"
   xmlns:csw="http://www.opengis.net/cat/csw/2.0.2">
</csw:DescribeRecord>
```

1.3 GetRecords

GetRecords operation works as the primary mean of resource discovery in the HTTP protocol binding. It does a search and a piggybacked present. A GetRecords request requires a constraint language and since the GOS CSW service doesn't support CQL (Common Query Language), only XML Filter with POST style requests are supported.

KVP style request:

Not supportted

POST style request:

```
<csw:GetRecords xmlns:csw="http://www.opengis.net/cat/csw/2.0.2" version="2.0.2"
  service="CSW" resultType="results" startPosition="1" maxRecords="10">
  <csw:Query typeNames="csw:Record" xmlns:ogc="http://www.opengis.net/ogc"
  xmlns:gml="http://www.opengis.net/gml">
  <csw:ElementSetName>full</csw:ElementSetName>
  <csw:Constraint version="1.1.0">
  <ogc:Filter>
  <ogc:PropertyIsGreaterThan>
```

<ogc:PropertyName>dct:modified</ogc:PropertyName>
<ogc:Literal>2000-01-01</ogc:Literal>
</ogc:PropertyIsGreaterThan>
</ogc:Filter>
</csw:Constraint>
<ogc:SortBy>
<ogc:SortProperty>
<ogc:PropertyName>dc:title</ogc:PropertyName>
<ogc:SortOrder>ASC</ogc:SortOrder>
</ogc:SortProperty>
</ogc:SortBy>
</csw:Query>
</csw:GetRecords>

1.4 GetRecordById

GetRecordById retrieves the default representation of catalogue records using their identifier. It returns the detailed information for some specific objects (Components and/or Services) of interest.

KVP style request:

http://gos2.geodata.gov/Portal/csw202/discovery?REQUEST=GetRecordById&service=CSW&version=2.0.2&Id=B6A0EC8C-826D-11D8-BADF-080020ECC953

POST style request:

```
<csw:GetRecordById xmlns:csw="http://www.opengis.net/cat/csw/2.0.2"
    version="2.0.2" service="CSW">
      <csw:Id>B6A0EC8C-826D-11D8-BADF-080020ECC953</csw:Id>
</csw:GetRecordById>
```

The module is capable of integrating one or more CSW (both CSW 2.0.1 and 2.0.2) services and GOS CSW service is integrated. Search options include keyword search in title, subject and abstract or a certain extend.

1.1 CSW Adapter

In this module, we have developed a CSW Adapter (Fig. 1) to mashup the differences among various CSW services. Its communication sequence includes: 1) the CSW Client collects user's input (such as CSW server URL, searching keywords, time span and geographic region) and submits a search request to CSW Adapter; 2) QueryString Analyzer extracts the parameters and sends a GetCapabilities request to CSW servers; 3) The Capabilities file Parser interprets the query filters and response schemas supported by this CSW server; 4)

Common Query Language (CQL) Generator then uses this information and query parameters to build the CQL that satisfies this CSW server and sends the request; 5) Response Metadata Interpreter receives the response and selects a proper metadata interpreter to interpret and format the results.

For example, FGDC interpreter will be selected if a CSW server returns FGDC-standard metadata. CQL Generator is used to integrate the different query filters supported by different CSWs based on service capabilities information; Response Metadata Interpreter is used to handle different response formats due to the different application profiles. All the metadata interpreters can plug-and-play to make the CSW Adapter flexible and scalable. By using the CSW Adapter, the CSW client could provide an intuitive and interactive GUI to collect user's input and display the formatted results. The differences of different CSW services are encapsulated and made transparent to others by the CSW adapter.

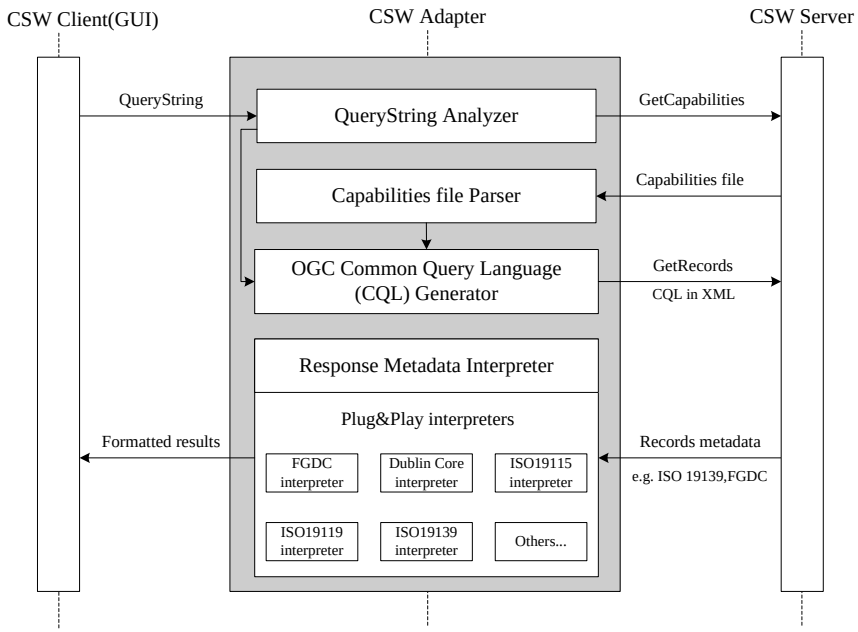

Figure 1. CSW Adapter Architecture and its communication sequence with the CSW client and server.

1.2 Ajax-based synchronous multi-CSW search.

CSW Adapter simplifies the integration of multiple CSW services into a single component. However, when searching from multiple services, the traditional approach is to send the request, receive and process the response sequentially. Therefore, the time spent increases linearly with the increase of server numbers. To address this issue, we adopted the multi-thread principle and utilized the Ajax techniques and designed an AJAX-based synchronous multi-catalogue search architecture (Fig. 2). In this architecture, Ajax-based

request/response controller is responsible for 1) sending requests to the distributed CSW servers simultaneously and 2) displaying results dynamically once getting response from any server.

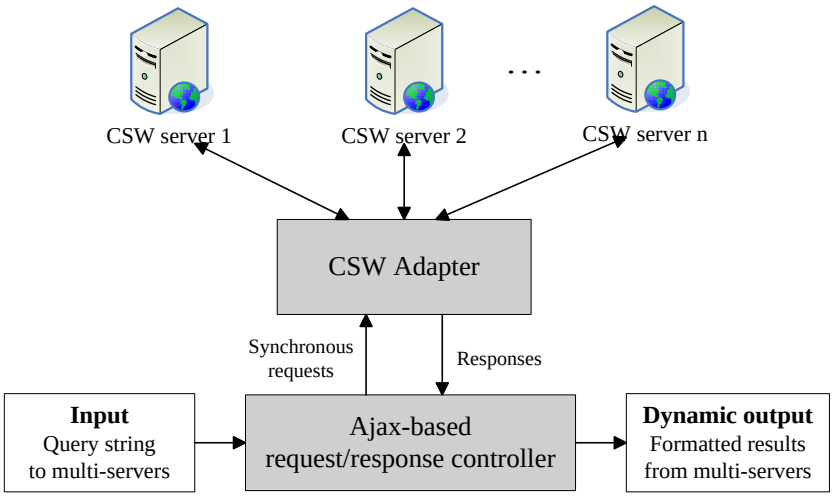

Figure 2. Architecture of AJAX-based synchronous multi-catalogue search

1.3 Integrating Semantic Search

To further enhance the intelligence, performance and interoperability of this discovery module, we utilized Semantic Search by utilizing Semantic Web for Earth Environmental Terminology (SWEET) and Web Ontology Service (WOS).

2. WMS integration module

2.1 Service Capability Clearing House

In order to provide a seamless integration framework for WMS services, we have designed and developed a Service Capability Clearing House (SCCH) (Fig. 3) to accelerate the WMS integration. The major goal of SCCH is to harvest, pre-process the WMS capabilities files and to cache the processed information in a database. SCCH works as a centralized service broker to cache service metadata on a centralized server. Therefore, the cached information can not only be accessed by the user invoking the operation, but also by everyone connecting to SCCH. WMSs are harvested by SCCH from CSW search results, user input and service crawler: a) Capabilities file parser is responsible for extracting the layer-based metadata and store them in the service layer repository; b) capabilities updater is running in the background to update the service metadata by reprocessing the capabilities information on a pre-defined frequency (e.g. daily) to keep them consistent with providers. Layer-based performance evaluator is responsible for testing and recording the response time for each layer. APIs are provided for other components to retrieve information from the SCCH.

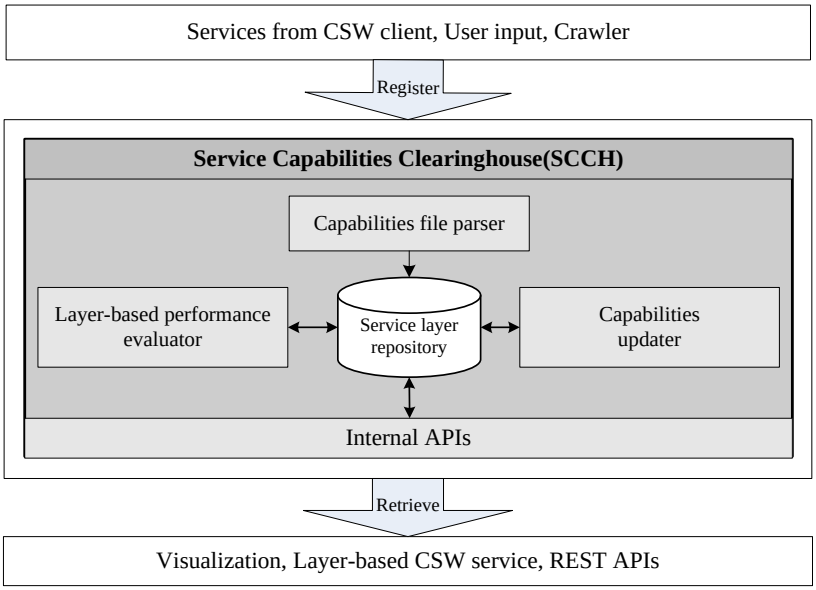

Figure 3. Architecture and workflow of SCCH

When a service metadata is requested, SCCH will search the requested records in its database and respond to the users immediately if service is already registered to reduce the waiting time. If the requested service is not in SCCH, SCCH will automatically harvest this service and return the requested records, in which case it needs a little more time for downloading and interpreting capability file on-the-fly. SCCH automatically collects services from users' requests, which makes it a self-growing system.

2.2. Layer-based search module

The actual service unit of WMS is a layer, which means every service contains one or, in most cases more than one layers. When users accesses a service, what they need are actually the layers provided by the service. SCCH acts as a layer repository and has detailed meta-information for each layer; this meta-information is adequate to build the query string to fetch real data from the server.

Based upon such a scenario, we have designed and developed a layer-based search API to provide functions that can search the layer information (such as name, title, keywords, reference system, geographic extent etc) directly from SCCH. This layer-based search APIs not only enhances the search efficiency but also makes SCCH interoperable. Other APIs can be utilized to access processed information in SCCH such as each layer's performance information.

3. Functionality demonstrations

3.1 Figure 4 shows the discovery module GUI with Multi-CSW and Semantic Search support

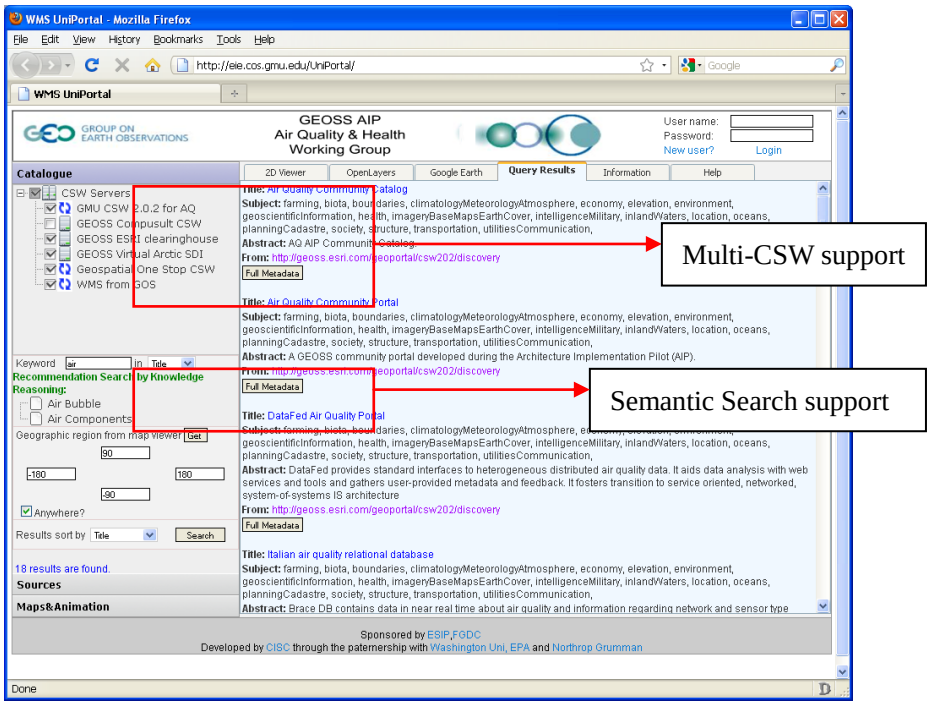

Figure 4. Discovery module GUI with Multi-CSW and Semantic Search support

3.2 WMS servers and layers are managed in a tree structure. Quality information of these servers and layers are shown as icons. On clicking any of these layers or servers, the major pane displays its metadata as shown in Fig.5. By checking any layers presented, 2D map view will display the map layer as show in Fig 6.

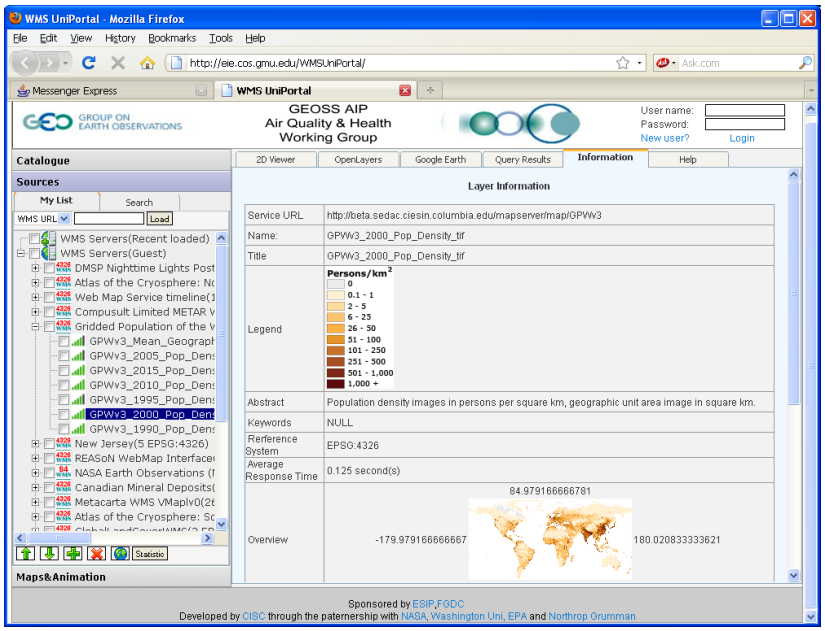

Figure 5. Metadata view

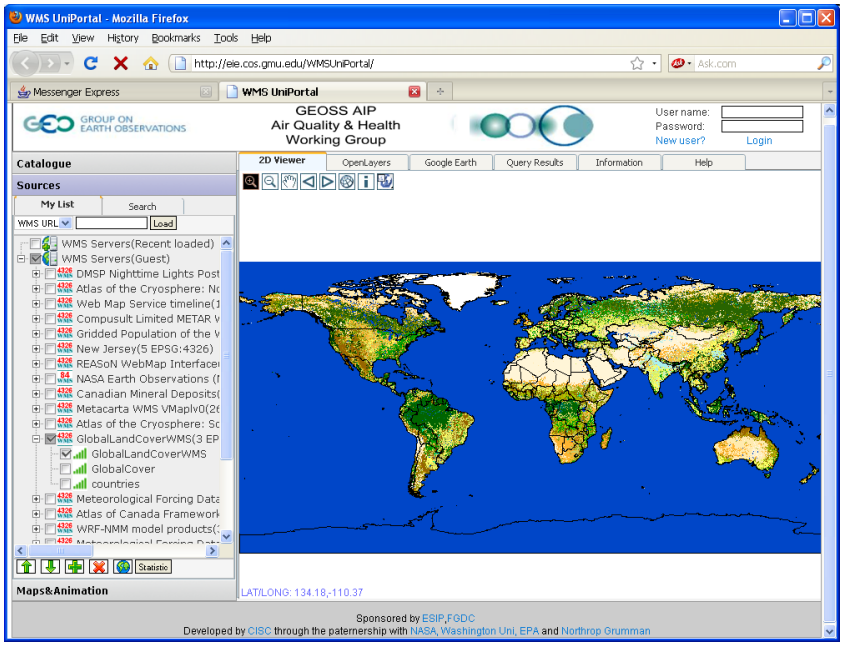

Figure 6. 2D Map view

3.3 Through check on, check off, move up, move down, these map layers can be arbitrary combined to thematic maps as shown in Fig.7.

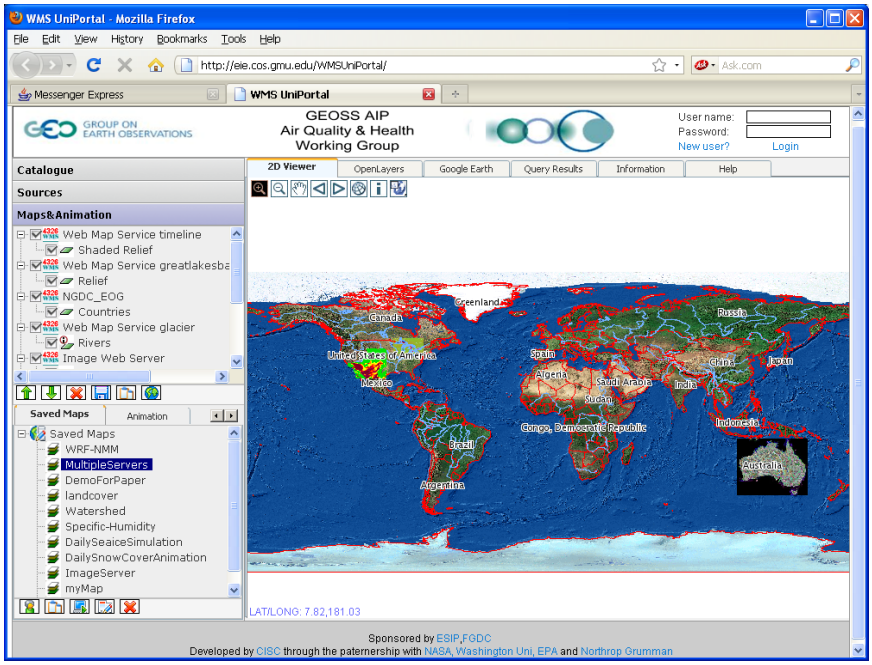

Figure 7 Arbitrary combined thematic map

3.4 New WMS servers can be added to the map layer tree by simply adding URL or ATOM feed (see Fig. 8).

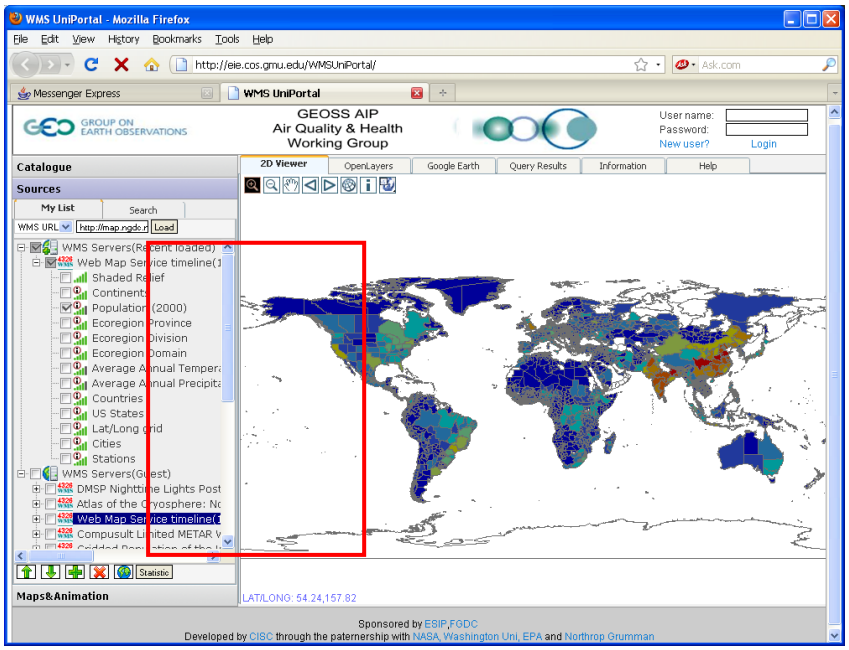

Figure 8. Add new WMS servers

3.5 Keyword search in these layers are provided and clients can choose these layers with better quality as shown in Fig.9.

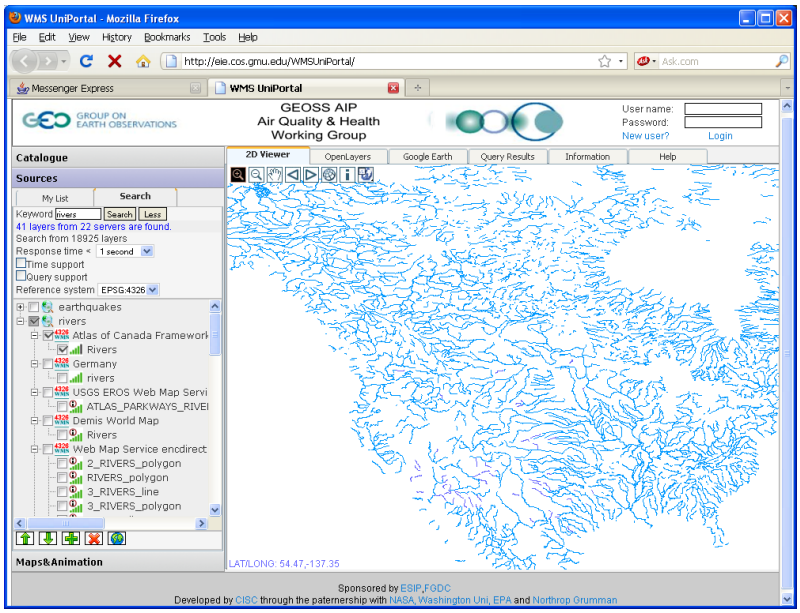

Figure 9. Search thematic layers

Next Steps:

Future work will focus on documentation and make the module open-source and the following tasks:

4. A composed map can be saved as a Map Context File and can be opened anywhere.

5. The module will provide map/data access through three viewers: 2D viewer, OpenLayers Viewer, Google Earth
6. The 2D viewer and Google Earth viewer will support map animation based on WMS-Time
7. Integrate to support different applications, such as Arctic SDI portal, Prototype portlet/client to integrate GOS metadata and quality monitoring results,

References

1. de La Beaujardiere, J. (Ed.), 2004, *Web Map Service Implementation Specifications, Version 1.3. OGC Document Number: 04-024*, Open Geospatial Consortium, U.S., 85pp.
2. Nebert, D., and A. Whiteside, 2005. Catalog Services, Version 2, OGC Implementation Specification, URL: http://portal.opengis.org/files/?artifact_id_5929, OGC
3. Nebert, D., 2004, *Developing Spatial Data Infrastructures: The SDI Cookbook*, Version 2.0, Global Spatial Data Infrastructure, D. Nebert (Eds), 171pp.
4. Yang C., Wong D., Yang R., Kafatos M., and Li Q., 2005, Performance Improving Techniques in WebGIS , *International Journal of Geographical Information Science*, 19, 319-342.
5. Yang C., Cao Y., Evans J., 2007. WMS Performance and Client design principles, *The journal of GeoInformation Science and Remote Sensing*, 44, 320-333.

Publications

1. Yang C., Wu H., Huang Q., Li Z., Li J., 2009. Spatial Computing for Supporting Physical Science, *PNAS*. (submitted)
2. Wu H., Li Z., Zhang H., Yang C., Shen S., 2009, Monitoring and Evaluating Web Map Service Resources for Optimizing Map Composition over the Internet to Support Decision Making, *Computer and Geoscience*. (Submitted)
3. Li Z., Yang C., Wu H., Li W., Miao L., 2009, An optimized framework for seamlessly integrating OGC web services to support geospatial sciences, *IJGIS*. (submitted)