WEB-BASED GEOSPATIAL RESOURCE SHARING THROUGH GEOPW

Xi Zhai, Liangcun Jiang, Peng Yue

State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing (LIESMARS),
Wuhan University, 129 Luoyu Road, Wuhan, 430079, China
Email: pyue@whu.edu.cn

Commission VI

KEY WORDS: Geospatial Web Service, Geoprocessing Web, Workflow, Service Chaining, Online Collaboration

ABSTRACT:

As Web-related technologies have matured in recent years, an increasing amount of geospatial resources (e.g. geospatial services, workflows, and geospatial data) are available in the distributed Web environment. Consequently, effective and efficient sharing and management of geospatial resources on the Web are necessary for better utilizing these resources for education and scientific research. This matches the vision of Geoprocessing Web, which emphasizes the sharing and access of geoprocessing utilities from the perspectives of communication, collaboration, and participation. Previous work on GeoPW has provided a large number of geoprocessing services over the Web. In this paper, GeoPW goes further to offer a Web platform for sharing geospatial resources. The paper presents the design, implementation, and functions of the platform, which offers a user-friendly environment for publication, discovery, and communication of geospatial data, services, and workflows.

1. INTRODUCTION

With the advancement of Web technologies and geospatial information service, geospatial resources including geospatial data and services are increasingly rich and accessible in the distributed Web environment. The wide spread of Web 2.0 technology also increases the dissemination of geographical information on the Web (Gong et al., 2009). Consequently, effective and efficient sharing and management of geospatial resources on the Web are necessary for better utilizing these resources for education and scientific research. This matches the vision of Geoprocessing Web (Zhao et al., 2012), which emphasizes the share and access of geoprocessing utilities from the perspectives of communication, collaboration, and participation. Over the last few years, geospatial resources created enormous potential for education and scientific research. They are distributed in the Web environment, each accessible through standard interfaces and protocols. When geospatial users are interested in using these geospatial resources, the limited accessibility of acquiring data and analysis workflows hampers their activities. It is necessary to offer a Web platform to significantly promoting the wide sharing of, discovery of, and access to distributed geospatial resources. Although geospatial infrastructures (e.g. Spatial Data Infrastructure), provide benefits to resources sharing and management, the mechanisms to store and transfer resources in these infrastructures are technologically complex. Current work focuses on the registration and discovery of specific geospatial data and services using standards-compliant API, with less effort on the development of Web-based platform for communication, collaboration, and participation using these data and services. For example, existing work on the discovery of sensors and observations focuses on the comprehensive design of a registry using CSW-ebRIM to support advanced functions such as metadata harvesting and registration in a Sensor Web environment (Zhai et al., 2012). Our previous work on GeoPW (Yue et al., 2010) also provided a large number of geoprocessing services over the Web. Yet we are still lacking of tools or platforms to facilitate the sharing of these resources.

In this paper, GeoPW goes further to offer a Web platform for sharing geospatial resources. The paper presents the design and implementation of the platform, which offers a user-friendly environment for publishing, discovering, and communicating on geospatial data, services, and workflows. It adopts RESTful (REST, 2013) style Web service for sharing geospatial resources. It is developed based on Ruby on Rails (Craig et al., 2006) Web Framework and follows the Model View Controller (MVC) (MVC, 2014) architecture. The MVC is a software pattern for implementing user interfaces. It divides a given software application into three interconnected parts, including model, view and controller, to implement a kind of dynamic programming and make the follow-up program modification and extension simply. The programming language used for the development is Ruby (Flanagan and Matsumoto, 2008) due to its light-weight and human readable features. The platform offers a user-friendly environment for publication, discovery, and communication of geospatial data, services, and workflows. The remainder of this paper is organized as follows: Section 2 presents the architecture of this platform; Section 3 describes implementation and functions; Conclusion is provided in Section 4.

2. ARCHITECTURE

The architecture of the platform is shown in Figure 1. It is based on the Ruby on Rails Web application framework and follows the Model View Controller pattern. OpenID (OpenID 2014) provides a platform for user-centric identity management, allowing user choose digital addresses, identity providers, and services providers. End users can visit the Web site through external OpenID services or the internal username/password mechanism. Or they can also adopt the method of anonymous access. In the *view* layer, it provide an HTML based Web interface together with other kinds of interfaces, such as RSS interface, OAI-ORE interface, and the RESTful XML interface. The RESTful interface allows applications to be easily connected and accessed. The *controller* layer is responsible for forwarding the request and processing requests. In the *model*

layer, the models follow the architectural design of Rails. These data models are managed via case basis library, which is supported through a set of entities, including research objects (e.g. packs, services and groups) and basic metadata entities (citations, tags and policies). Furthermore developers can add data models according to the needs of applications. For example, a W3C (World Wide Web Consortium) service can be registered in this platform as a service entity using the modelling mechanism. Currently, there are nearly thirty entities used in this platform. Examples of existing entities include services, workflows, blogs, announcements, bookmarks, users, citations, groups, jobs, tags, contributions, messages, reviews, policies comments, friendships, and packs.

In term of the wide sharing of, discovery of, and access to distributed geospatial resources (e.g. data, services and workflows), the data server, Web server, workflow server, mail server, database server, and search server are all separate systems to which the main application connects. Geospatial data, geoprocessing functions, and sensor Web resources are encapsulated as services with standard interfaces and protocols to allow Web-based sharing and automatic access. The platform can collect geospatial resources from the corresponding server (e.g. data server, Web server, and workflow server). The OGC (Open Geospatial Consortium) service interfaces provide the possibility to harvest resources, including data and services. The database, which is an important component of the Ruby on Rails system, is hosted on the database server in the form of MySQL. This search server runs the Solr search server, which is a Java implementation running as a Java servlet in Tomcat (De Roure et al., 2008). When a new user is registered in this platform, the mail server will play a pivotal role in sending confirmation letters for security.

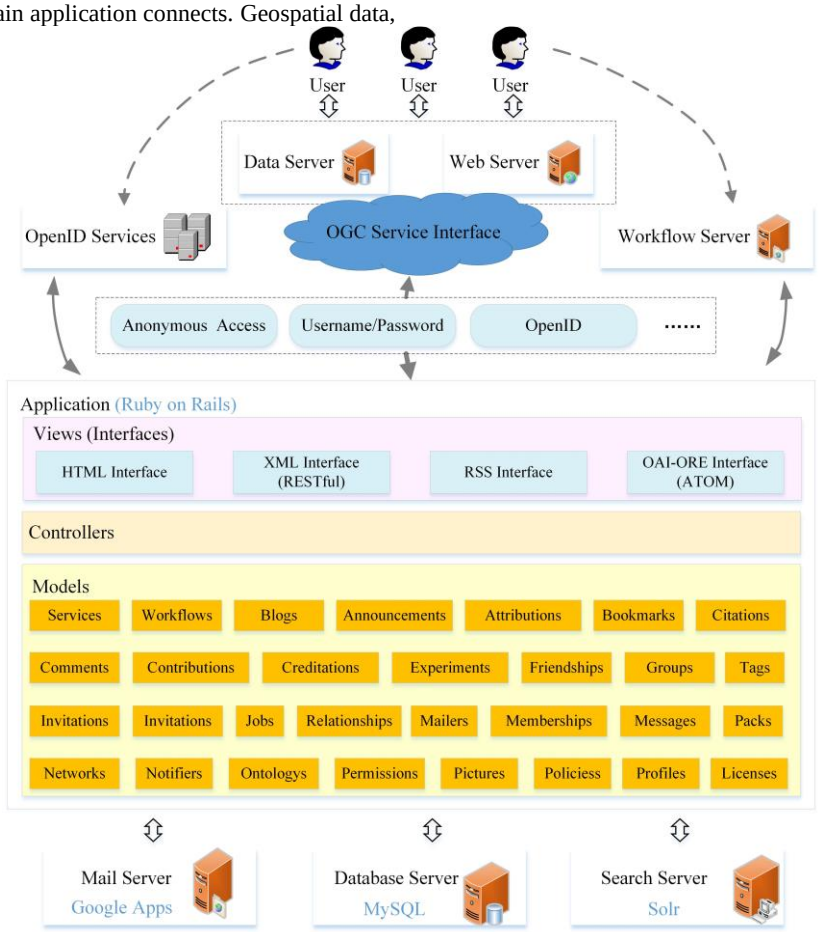


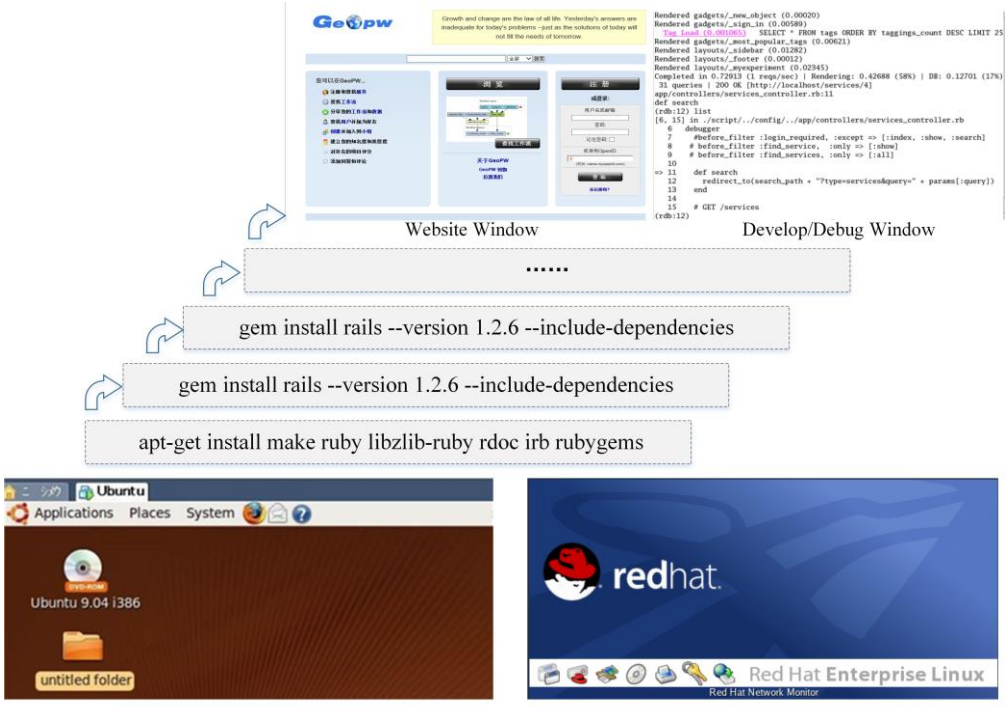
Figure 1. The architecture of the platform

3. IMPLEMENTATION AND FUNCTION

A. Implementation

The Web platform is designed as a Browser/Server (B/S) website. The implementation follows the Model-View-Control pattern. Geospatial resources objects are stored as models. The common controller makes sure that views and models keep consistent. The platform running environment needs to be set up primarily. The Linux operating systems such as Ubuntu or RedHat can be used to develop and run the platform. Ruby on Rails system, as the major component, are deployed, together with the JDK and Tomcat servlet. All metadata are stored in a local MySQL relational database. It is necessary to operate database migration and load balancing, in order to docking data and guarantee the stability of concurrent access. All operations can be executed by the command line of Linux (Figure 2). Next, the platform framework is built on the Linux system. The models for various geospatial resources, including sensor observation service , are generated by programmer and inherits from the model basis library. The development of programs follows the MVC pattern. The final step is to load balancing

processing and test concurrent access performance of the platform.



Ubuntu Linux system RedHat Linux system

Figure 2. The process of building the platform running environment

B. **Function**

1) Registration of Services

The platform can register different types of geospatial services, including Web services following W3C standards, and geoprocessing services, and sensor Web services following the OGC specifications. For example, if a user want to register a wps (Web Processing Service), one URL of the service description (e.g. http://geopw.whu.edu.cn:8080/wps/RSWebProcessingService?service=wps&request=getcapabilities) is necessary to harvest the metadata of this service. The metadata of services, including name, version, and abstract, can be extracted and checked against the records in the database. If the service is new, it can harvest the capabilities document in the regular way by decomposing the document into fragments according to the information model for WPS services. For example, the platform has published more than one hundred services from different WPS providers. These services are open and accessible to the public, and support integrated geoprocessing on the Web. Figure 3 shows that uses can check metadata information of a specific service, such as names, functions, and providers, in an intuitive Web form. By clicking each WPS process, users can also see the process information for each geoprocessing function.

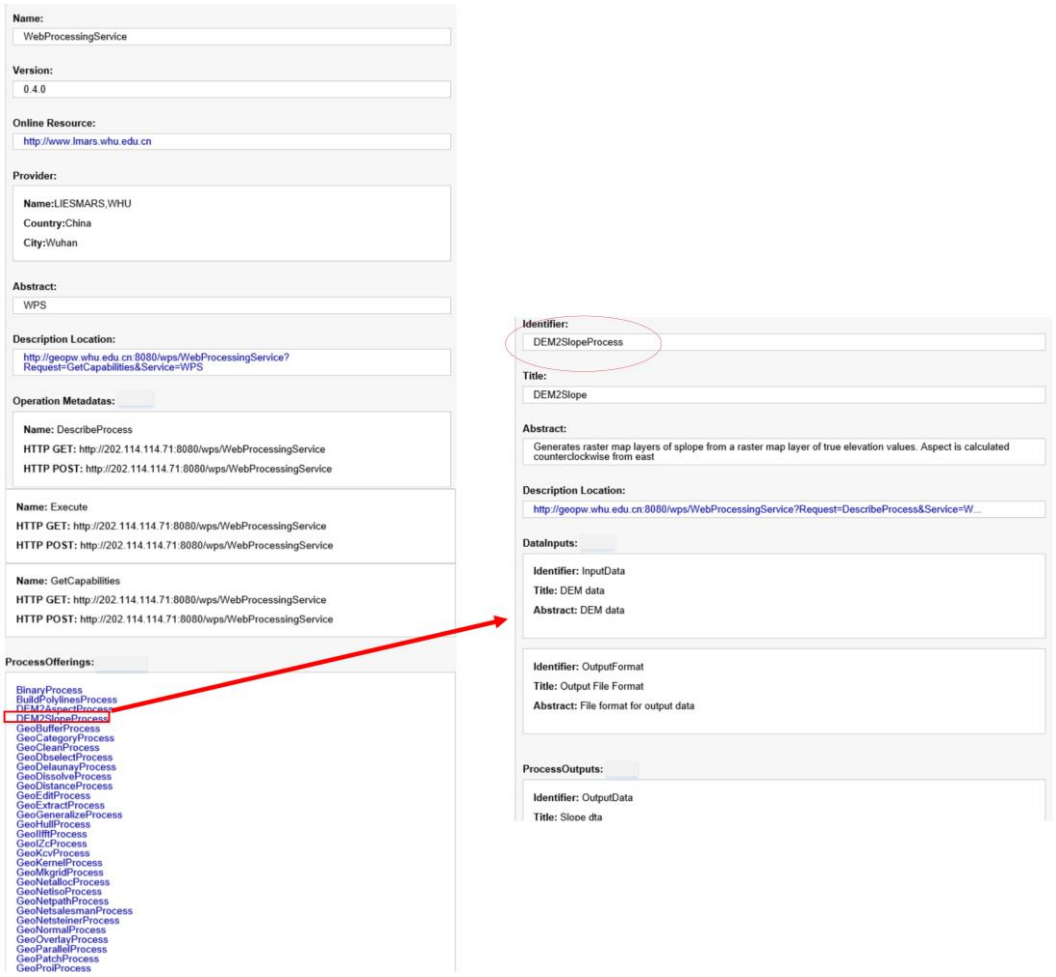


Figure 3. WPS Service Registration

2) Registration of Workflows

Once different services are chained as workflows, the workflows can also be published in the platform as a kind of geospatial resources. Users can download any workflow package in the platform, and open it using an open-source model builder, i.e. GeoJModelBuilder. It is a light-weight, open-source model builder, having its own workflow description and execution engine and is available at SourceForge.net (http://sourceforge.net/projects/geopw/). For instance, in the flood analysis case, one workflow named "water extraction workflow" (Figure 4), is provided by an author using the GeoJModelBuilder. Once the workflow is uploaded in the platform, it can be widely open accessible. If a researcher want to carry out the similar geospatial analysis, she/he need not to build the similar workflow again. Thus it is easy and convenient for students and researchers to reuse exist contributions and share research results to others to help solve complex geoprocessing tasks. The platform allows users to edit and update the other information such as ratings, tags, comments, and privilege.

Figure 4. Workflow Registration

3) Registration of Geospatial Data

In addition to services and workflows, geospatial data can be also published and discovered by this platform. If datasets are files, they can be uploaded directly into the platform. If they are accessible through OGC-compliant data services, such as Web Feature Services and Web Coverage Services, they can be published following the same way as geoprocessing services.

4) Discovery Function

All data, services, and workflows can be searched in the platform in an intuitive way like a Web search engine. The discovery function in the platform is implemented by developing a search engine middleware based on the solr search server, which delegates searches as SQL queries in the background database. The search engine middleware also adopts caching strategies, which can store existing query results in the cache to improve efficiency. There are two methods for resource search, keyword and filter queries. The former one is to use keywords to carry out search from all records stored in

MySQL. This is a free-text based search. The later one is the discovery of geospatial resources using filter parameters, including classification, type, user, and license. Users freely choose least one or more conditions (e.g. name of a specific user) listed below each parameter to conduct query. It gives full consideration to the requirements of the user operation. And the query results will only indicate what types of query parameters selected by user. For example, if a term as the user type is adopted as filter parameter, the results will only refer to resources whose user/author is the term. Other records (e.g. using the term as the classification) will be ignored (Figure 5).

Figure 5. Filter Query and Result

5) Other Functions

Users can register and form groups in the platform to facilitate the communication and collaboration. There will be different groups according to interests of users, so it is convenient for information exchange and geospatial resources sharing. For example, if a student has the background of remote sensing, he can set up a group and invite other people in this domain or interested in remote sensing to join in them. It will help them exchange and advance the knowledge. Registered users can upload, search, collect, manage resources in this platform. If there are some news, it supports to send announcement to a notice user. A set of Web 2.0 functions, such as customer tagging and third-party evaluation on geospatial resources, are supported in this platform. It is useful to improve popularity and credibility of the geospatial resources based on the community review and feedback functions in the platform.

4. CONCLUSIONS

The results of this research provide a Web platform to support better utilizing geospatial resources for education and scientific research. The paper provides a comprehensive design for registration and discovery of geospatial resources, including geospatial services, workflows, and geospatial data. The platform is implemented followed MVC framework and it is with characteristics of Web 2.0 style. The work makes it much easier and faster for researchers to access and use geospatial resources in a distribute environment.

ACKNOWLEDGEMENTS

This research is supported by National Basic Research Program of China (2011CB707105), National Natural Science Foundation of China (41271397), and Program for New Century Excellent Talents in University (NCET-13-0435).

REFERENCES

Craig, M., Nguyen, T., Luu, B., Manarang, E. M., Williams, C., 2006. Ruby on Rails.

De Roure, D., Goble, C., Bhagat, J., Cruickshank, D., Goderis, A., Michaelides, D., Newman, D, 2008. myExperiment: defining the social virtual research environment, In: IEEE Fourth International Conference on eScience (eScience 2008), IEEE publication, 8 pp, 2008.

Flanagan, D., and Matsumoto, Y., 2008. The ruby programming language, " O'Reilly Media, Inc.".

Gong, J., Wu, H., Gao, W., Yue, P., Zhu, X., 2009. Geospatial Service Web. In Li, D., Shan, J., Gong, J. (eds) Geospatial Technology for Earth Observation Data. Springer New York, Dordrechet Heidelberg London, pp.355–379.

MVC, 2014. http://zh.wikipedia.org/wiki/MVC, (Accessed 24 March, 2014).

OpenID, 2014. http://zh.wikipedia.org/wiki/OpenID, (Accessed 10 February, 2014).

REST, 2013. http://zh.wikipedia.org/wiki/REST, (Accessed 25 March, 2014).

Yue, P., Gong, J., Di, L., Yuan, J., Sun, L., Sun, Z., and Wang, Q., 2010. GeoPW: Laying Blocks for the Geospatial Processing Web. Transactions in GIS, 14 (6), pp.755–772.

Zhai, X., Zhu, X., Lu, X., Yuan, J., Li, M., Yue, P, 2012. Metadata Harvesting and Registration in a Geospatial Sensor Web Registry. Transactions in GIS, 16(6), pp.763-780.

Zhao, P., Foerster, T., and Yue, P., 2012. The geoprocessing web. Computers & Geosciences, 47 (10), pp.3-12.