Computers & Geosciences 47 (2012) 143–150

Contents lists available at SciVerse ScienceDirect

Computers & Geosciences

journal homepage: www.elsevier.com/locate/cageo

GWASS: GRASS web application software system based on the GeoBrain web service

Fang Qiu [a,*], Feng Ni [a], Bryan Chastain [a], Haiting Huang [a], Peisheng Zhao [b], Weiguo Han [b], Liping Di [b]

[a] Geospatial Information Sciences, University of Texas at Dallas, Richardson, TX, USA
[b] Center for Spatial Information Science and Systems, George Mason University, Fairfax, VA, USA

ARTICLE INFO

Article history:
Received 29 March 2011
Received in revised form
20 January 2012
Accepted 31 January 2012
Available online 4 February 2012

Keywords:
GIS
GRASS
SOA
Web service

ABSTRACT

GRASS is a well-known geographic information system developed more than 30 years ago. As one of the earliest GIS systems, GRASS has currently survived mainly as free, open-source desktop GIS software, with users primarily limited to the research community or among programmers who use it to create customized functions. To allow average GIS end users to continue taking advantage of this widely-used software, we developed a GRASS Web Application Software System (GWASS), a distributed, web-based, multi-tiered Geospatial Information System (GIS) built on top of the GeoBrain web service, a project sponsored by NASA using the latest service oriented architecture (SOA). This SOA enabled system offers an effective and practical alternative to current commercial desktop GIS solutions. With GWASS, all geospatial processing and analyses are conducted by the server, so users are not required to install any software at the client side, which reduces the cost of access for users. The only resource needed to use GWASS is an access to the Internet, and anyone who knows how to use a web browser can operate the system. The SOA framework is revitalizing the GRASS as a new means to bring powerful geospatial analysis and resources to more users with concurrent access.

Published by Elsevier Ltd.

1. Introduction

Geographic Resources Analysis Support System, commonly referred to as GRASS, is a geographic information system used for geospatial data management, image processing, analysis and modeling, visualization, and graphics production with many different types of data. Developed originally by the U.S. Army Construction Engineering Research Laboratories in 1982, GRASS has since evolved into popular GIS software with a wide range of application domains (GRASS, 2011). With the development of many commercial GIS software, such as ArcGIS, MapInfo, IDRISI, etc., GRASS has gradually given away its share in the GIS software market and survived mainly as free/open-source software, with users primarily limited to the researchers and programmers in universities, academic institutions and governmental agencies, for creating customized functions not available in commercial software (Mitasova et al., 1995). This is because until recently, GRASS was mainly a Unix-based command line system, which posed a steep learning curve for modern GIS users, who are accustomed to graphical user interface (GUI) based systems.

Various efforts have been made to keep GRASS alive as a functional GIS system for modern day users. Since 1995, Baylor University and other members of the GRASS Research Group and Development team have taken over the development and support of GRASS. In 1999, GRASS software began to be released under the GNU General Public License (GPL). GRASS is now one of the eight initial software projects of the Open Source Geospatial Foundation.

The first effort of the new GRASS development team was the porting of the software to other Unix compliant operating systems, such as GNU/Linux, Solaris, SGI IRIX, HP UX, Mac OS X, IBM AIX, BSD-Unix variants, FreeBSD, and CRAY Unicos (GRASS, 2011). The process was relatively easy and required minimum modification of the source code due to their similar architecture. The transport of GRASS into other non-Unix compliant platforms such as Microsoft Windows was much more strenuous, involving years of efforts to refit millions of lines of code into the new architecture. For many years, users had to install Cygwin, a Linux user interface emulator in order to run GRASS from within Windows (GRASS, 2011). However, users' experience indicates that GRASS running on the Cygwin platform was less efficient and stable than GRASS on the native UNIX/Linux platform. To alleviate these issues, a new native Windows system was developed, called winGRASS, with its first beta release in 2006 (GRASS, 2011). This new environment employs the wxPython UI to provide a more familiar interface to users who are accustomed to contemporary commercial GIS software. The current version of winGRASS is considered to be in "mature beta" status. While almost all of the main functions perform well as they do on Unix-based GRASS,

* Corresponding author. Tel.: +1 9728834134; fax: +1 9728836573.
E-mail address: ffqiu@utdallas.edu (F. Qiu).

0098-3004/$ - see front matter Published by Elsevier Ltd.
doi:10.1016/j.cageo.2012.01.023

there are minor issues such as legacy module support not working at this time (GRASS, 2011). One major drawback of GRASS remains; the interface and functionality is not the same from one operating system to another. To allow an identical user experience from one platform to another, regardless of operating system or hardware, a uniform web-based GUI for GRASS could be a solution.

In this paper, we propose to utilize the emerging Service Oriented Architecture (SOA) technology to revitalize GRASS. Unlike the previous efforts, the reuse of this software is at the process level, instead of the source code level. Therefore, no copying code, incorporating shared libraries, inheriting objects, or modifying functions to fit the software in the new operating system is required. Consequently, redevelopment costs are greatly reduced, while the efficiency and stability of the system are not comprised. Other benefits include easy to use web-based GUI, concurrent access to the system, zero cost for software licensing, upgrading, and maintenance to the end users, and the additional free access to multi-terabytes of geospatial data.

2. Service oriented architecture

Service-oriented architecture (SOA) is an approach to systems development and integration through a loose coupling of functionality in the form of interoperable services, so that existing and new applications developed in diverse operating systems, programming languages and architectures can be dynamically combined and reused to build software solutions (Pallos, 2001). The fundamental elements of a service-oriented architecture are services, autonomous, stateless software units that perform actions. These services communicate with each other by passing data (instead of parameters) from one service to another, or by coordinating an activity between two or more services. The implementation of SOA over the Internet is manifested by a special kind of services – web services, which use standard Internet protocols to communicate with each other to solve complicated tasks (Di et al., 2006), which enables the interoperable use of heterogeneous data and processing within a single network environment (Curbera et al., 2002).

There are three major players in SOA, including service providers who publish and provide specific services over the Internet, service requesters (users) who find and invoke request for certain services to construct a software solution, and service brokers who allow services providers to register their services in a catalog and offer search mechanism to help requestors locate appropriate services. Web Services Definition Language (WSDL), is often employed to supply descriptions of the web services' application programming interface (API). Communication between service providers and requesters is typically enabled by simple object access protocol (SOAP) messages and calls. The Universal Description, Discovery, and Integration (UDDI) standard defines a protocol for directory services that enables web service requesters to locate candidate services and discover their details (Curbera et al., 2002). These protocols and standards that the service providers, requesters, and brokers utilize to perform SOA's basic operations are all based on eXtensible Markup Language (XML). XML is now established lingua franca for information and data encoding for platform independent communication, which allows for the creation of web services from a conglomeration of components built on different machines, running different operating systems, developed in different programming languages, and based on diverse architecture (Pallos, 2001).

The GIS community also becomes interested in utilizing SOA technology to disseminate geospatial data and geoprocessing capabilities online. For example, Environmental Systems Research Institute Inc. (ESRI, 2011), currently the largest GIS software vendor has embraced the SOA technology by providing ArcWeb Services to its user community, allowing access to an extensive collection of map content and analysis capabilities, as well as the ability to integrate customer data in users' web sites or standalone applications (ESRI, 2011). The Open Geospatial Consortium (OGC), a non-profit, international standards organization with contributors from both the private industry sector and academia, has also taken the lead in developing web service standards for publishing geospatial data and operations. The open standards that OGC defines for geospatial data and services, such as Web Coverage Service (WCS), Web Feature Service (WFS), Web Map Service (WMS), and Catalog Service for Web (CSW) are being increasingly adopted by the GIS developers (Zhao et al., 2005).

Another standard developed by OGC is the Web Processing Service (WPS), which was officially adopted in 2007 (OGC, 2007). This standard specifies a service interface for publishing and performing geospatial processes over the web. Since its adoption, several WPS implementations that adhere to the OGC standard have been developed, many of which utilize GRASS functions. One such service was developed by the 52° North Initiative, which uses WPS as a wrapper for GRASS functionality and makes it accessible programmatically via the JAVA language (Brauner and Schaeffer, 2008; Müller et al., 2010). Another popular OGC WPS implementation is pyWPS, which also has native GRASS support, and can be programmed in either the JAVA or Python languages (PyWPS, 2011). Bergenheim et al. (2009) employed an different approach to WPS, which uses PHP to enable users to access GRASS geoprocessing capabilities on several open source desktop GIS platforms, with the unique benefit of not requiring any programming. Instead, their method only requires writing XML code to access different geoprocessing tools. However, this service is currently limited only to vector processing.

GeoBrain is an open, interoperable, distributed, standard-compliant, multi-tier web-based geospatial information service based on SOA technology. It was built as part of a broader endeavor supported by NASA in providing a data-rich on-line learning and research environment that enables students, faculty members, and researchers from institutes all over the world to easily access, analyze, and model with the huge amount of NASA Earth Observing System (EOS) data just as if they possessed those vast resources locally at their desktops (Di, 2004b). One of the development strategies adopted by GeoBrain is to facilitate web-based interoperability by making existing geospatial processing packages web-service enabled. GRASS, along with some GDAL (Geospatial Data Abstraction Library) functions, was chosen by the project as the major software package to support web service development due to its free and open source nature.

Instead of launching a full overhaul of the functions from scratch, the creators of GeoBrain used the concepts of geo-objects and geo-trees to create distributed geospatial web services under SOA (Di, 2004b). Geo-objects consist of data, metadata, and a set of methods to operate on said data. The decomposition of user geo-objects into a tree representing the process workflow (including input geo-objects) is a geo-tree. The geo-tree is therefore a virtual geo-object, representing the specific types of products that it can create. From a web services point of view, a geo-tree represents a complex service chain, where tasks are combined in a dependent series to achieve larger tasks, such as geoprocessing requests (Di, 2004b). Thus, GeoBrain is able to represent complicated GRASS commands as a series of geo-objects, which are stored as a geo-tree and wrapped with a SOAP interface, making them easily accessible SOA compliant geospatial processing web services.

By representing GRASS geoprocessing tasks as SOA web services, subsequent use of the web services only requires

knowing their names and interfaces, which shields the algorithmic details and the infrastructure complexities from the outside world. By incorporating the ideas of open standards and SOA technology, the GeoBrain web service also provides an excellent environment to build new geoprocessing web services for sharing geoscience algorithms via the web (Li et al., 2010). GeoBrain has the added benefit of providing easy access to large quantities of spatial data over the web. Through GeoBrain, users have access to over 12 terabytes of data from such sources as Landsat, MODIS, ASTER, GOES, Sounder, etc. (GeoBrain, 2011). While other web processing services exist for GRASS, GeoBrain is unique in its inherent data access abilities.

However, the design of SOA is mainly intended for programmatic access, not terminal access. Consequently, the rich web service provided by the GeoBrain and other geoprocessing web services built on top of GRASS cannot be directly utilized by end users and their benefits are only available to the sophisticated GIS programmers and analysts (Di, 2004a). In order to capitalize the GeoBrain web service and put its specific functionality into practice, either thin client web-based applications or thick desktop-based applications should be built to consume its web services. Since web-based applications can reach more users via Internet access, with cross-platform availability, and require no installation, configuration, or maintenance that often haunt many desktop-based systems, we decided to develop a GRASS Web Application Software System (GWASS) on top of the GeoBrain web service as a new means of bringing the powerful geospatial analysis capabilities of the GIS software to the contemporary end users around the world.

3. GRASS web application software system

GWASS was developed to serve as a bridge between GIS end users and the rich GRASS functionalities. The architecture of the GWASS system is shown in Fig. 1. In essence, GWASS functions as a web implementation of the GRASS GIS system. To achieve this objective, many state-of-art IT techniques have been utilized. The GeoBrain web service is the core of the GWASS system, which provides access to the data management and geoprocessing functionalities of GRASS through their web services. On the server side, cluster computing environment has been utilized to improve performance of the GeoBrain web service (Di, 2004b). We built the GWASS as a web-based thin client application, which serves to fill the gap between the GeoBrain web service and end users. In order to improve the user experience, GWASS also incorporated AJAX technology to support web browsers like Internet Explorer, FireFox, and Google Chrome, which provided end users with much easier access to its functionality and data. Open source Apache Tomcat and AXIS have also been employed to construct the system architecture. Apache Tomcat is a web application server, while Apache AXIS plug-in is an implementation of the SOAP, which functions as the web services server.

The request from the end user is sent from the browser to GWASS, which then translates the custom request as XML messages or calls to pass to the GeoBrain server through SOAP. Each of GeoBrain's services has a WDSL description, which allows for the user to call the corresponding GRASS geoprocessing web service via URL to process the data. GWASS receives the results from GeoBrain via XML, and delivers the output to the end user when GRASS finishes the job. All operations occur on the server side and are recorded in a log file for system administration and error debugging.

The system has the following major module components: the *Multiple User Management* that allows multiple authorized users to login to the system with user name and password; the *Data Management* that enables users to upload custom data to the system, add links to distributed online data sources, delete intermediate data, download output data, and display image data; and the *GRASS based Geoprocessing* that provides users with the capabilities to perform various geoprocessing tasks.

3.1. Multiple user management

Unlike the original desktop GRASS software that only supports one concurrent end user at a time, GWASS as a web based application system needs to allow simultaneous access to the system by multiple users from different computer terminals to perform geospatial operations at the same time. This was the first challenge we faced in developing an online system through converting a single-user desktop program to a multiple-user web-based application. We tackled this challenging problem by utilizing the UUID (Universally Unique Identifier) technique. A new user will be asked to provide a unique login name and password. At the same time, a request is sent to the GWASS server to construct a corresponding Database identified by the UUID, which is empty workspace. The UUID number will be stored in the GWASS User Database with its login name and password

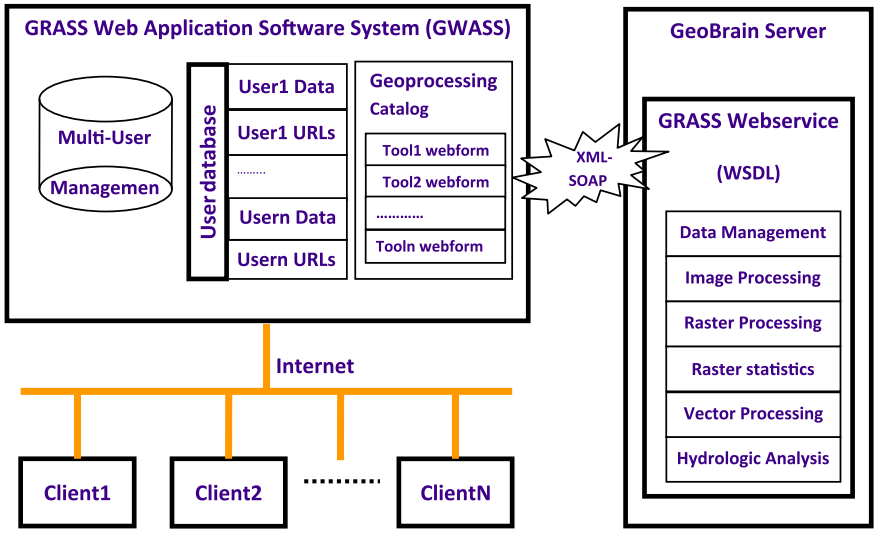

Fig. 1. The architecture of the GWASS system.

(Fig. 1). This process initializes the user workspace and is only necessary the first time a user logs on. All subsequent logons of returning users will not trigger the database construction process any more.

This user management strategy greatly reduces system overhead by avoiding the construction of separate databases for each user session. Additionally, once users put data onto the database, the data will persist across sessions, which allows users to continue geoprocessing tasks based on the results of previous operations. The construction of individual databases for each user bestows the system with the critical capability to support concurrent access by multiple users without worrying about access confliction.

3.2. User Data Management

The User Data Management module allows users to manage the data to be analyzed by the GWASS system. The data can either be uploaded to the GWASS server or be directly accessed by a URL (Fig. 2).

For users who wish to use custom data located in their local computers, it needs to be uploaded into GWASS server first before it can be processed by the GWASS system. Users can upload single data files or zipped files containing multiple data files into GWASS system. Then GWASS incorporates this data into the server by referencing the data in its user database. The users' custom data and output data stored in specific image formats (e.g., tiff, png) can easily be viewed with a web browser. Users have the option of viewing the data as a re-sampled image with lower resolution, which reduces the loading time. Users may also choose to view the data at its full image resolution. Users can delete intermediate data, download output data, and rename the data using the User Data Management tool.

In addition to uploading custom data to the GWASS server, end users may exploit various sources of geospatial data online. For example, GeoBrain itself provides multiple-sources of large, heterogeneous geospatial data for free through the GeoBrain's Geospatial Data Services Client via WFS, WMS, WCS and CSW protocols. GeoBrain is just one of many websites that make their geospatial data available for free download or with minimal charge. The GWASS User Data Management module provides the capability for users to manage multiple sources of data, including the user's own data, distributed at different locations. Users can add the URL of a data source into a user's URL database without actually importing the data into the GWASS system. A URL can be selected from the list of URLs that stored in the user database and used as an input parameter to GWASS geoprocessing tools. If a user chooses to keep the actual file of a URL data source, the data can also be optionally imported into the user database.

3.3. GRASS based geoprocessing

GRASS functions are organized into five major categories of web service in GeoBrain: Image Processing, Raster Processing, Raster Statistics, Vector Processing and Hydrological Analysis. For convenience, various functions in GWASS are also similarly grouped into each of the five categories within a tree-view catalog (Fig. 3). The catalog is composed of multilevel folders that are expandable to reach sub-folders and/or functions (Fig. 3). A geoprocessing operation is triggered by clicking on a leaf item of the catalog tree. To end users, the GWASS tree view catalog is similar to the ArcToolBox interface of the widely-used desktop ArcGIS system.

For each geoprocessing tool in the GWASS system, users can select either a data file or a URL from the respective list in the user's database as an input data parameter to supply to a GeoBrain service. After users invoke the GWASS system to perform a geoprocessing task, a request is sent to the GeoBrain system along with the associated URLs of input data. To respond to the request, the GeoBrain server first physically pulls the data over the internet and transforms it into GRASS format using GDAL services if necessary. If the URL points to data residing in the GeoBrain server, no data pulling will occur. After GeoBrain completes the geoprocessing tasks and sends a response to GWASS, GWASS forwards this response to the client and informs the user with the resultant data previewed on the web form. The users have the option to save the data as either a temporary file or a physical file in the GWASS user database for further analysis.

4. Results

GWASS is a Web-based, graphical user interface (GUI) driven GIS software, which completely eliminates the need for end users to manually enter any command as they do in the desktop GRASS system. Each of the geoprocessing tools is implemented as a web form composed of a number of web controls, such as input box, button, list and combo box. It is very user friendly in that anyone comfortable with an Internet browser should be capable of operating any and all of the available GWASS tools. A prototype of the GWASS

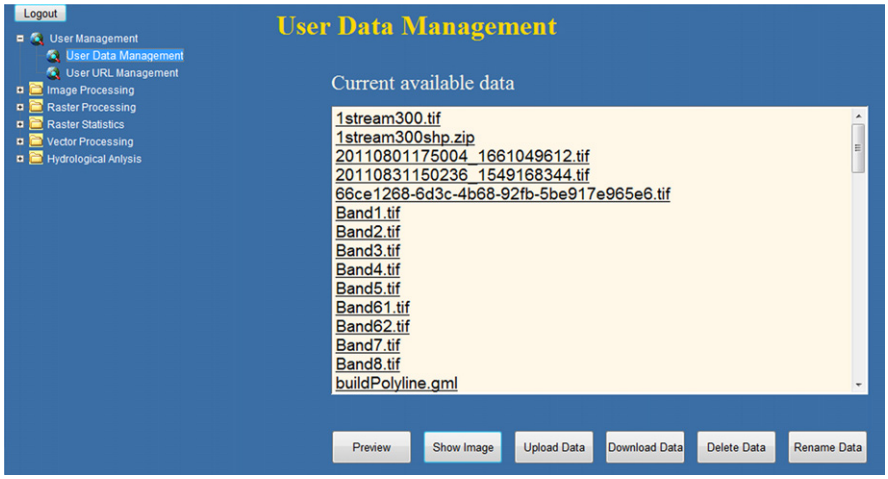

Fig. 2. GWASS Web-based user interface. This is the interface for the user data management with image/raster and vector layers in the data list.

system is available at http://wastetoenergy.utdallas.edu/gwass. Readers who would like to experiment with the GWASS system should contact the authors for a username and password.

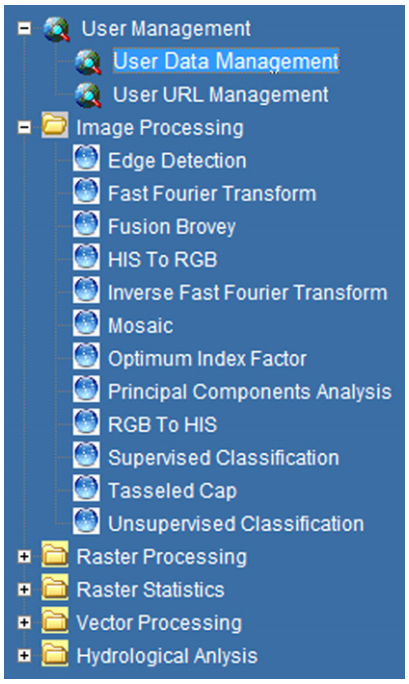

Fig. 3. GWASS Geoprocessing Tree View Catalog.

In order to demonstrate the ease of use and the flexibility of the GWASS system, two examples are presented below, one involving image classification with a custom data file and another entailing hydrological modeling with a distributed online data source.

After a user logs into the GWASS system, s/he can upload his/her custom data or add distributed online data sources via URL into the user database. As previously mentioned, for raster data with specific image formats, users can take a quick view of a data file in re-sampled, low resolution images. To investigate the quality of the data in detail, users can visualize the data in full resolution, utilizing operations like zoom-in, zoom-out, and pan to examine the data in more detail (Fig. 4).

When the user decides on the data to be analyzed, s/he needs to expand the geoprocessing tree catalog to locate the tool to conduct the analysis. For example, a user wanting to perform an "Unsupervised Classification" can find the tool under the Image Processing folder. By clicking the tool, the web form for the Unsupervised Classification tool appears (Fig. 5). The user then completes the web form by specifying various parameters. For instance, s/he can select 6 bands of a TM dataset from the GWASS raster data list and add them to the input data list, which actually displays their URL in the GWASS server. Then s/he can specify other parameters if they are different from default values, such as setting the number of classes to 5, maximum number of iterations to 30, percent of convergence to 98%, and output filename to "unsuper_tm." Upon clicking the Submit button, a request for unsupervised classification web service is sent to the GeoBrain server along with all the parameters. Upon receiving the request, the GeoBrain server will then call the corresponding geoprocessing web service to perform the operation.

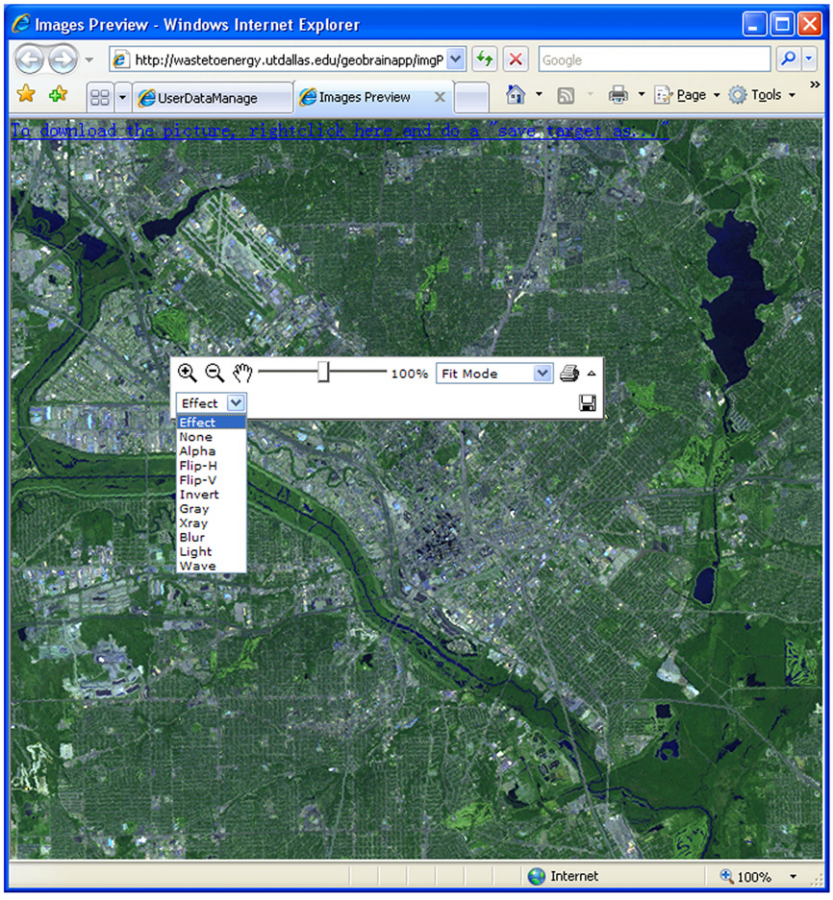

Fig. 4. Visualize downtown Dallas, Texas data in full image resolution. Using zoom-in, zoom-out, zoom to specific extent and pan, users can view every part of the image.

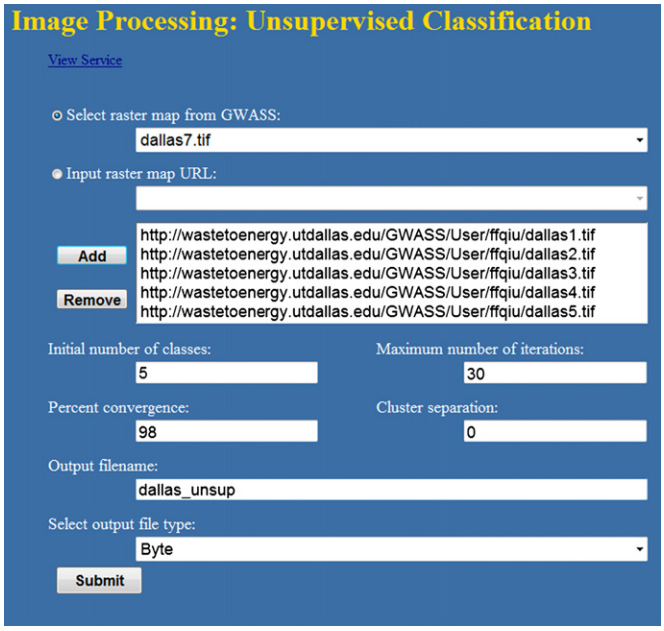

Fig. 5. GWASS web form for Unsupervised Classification.

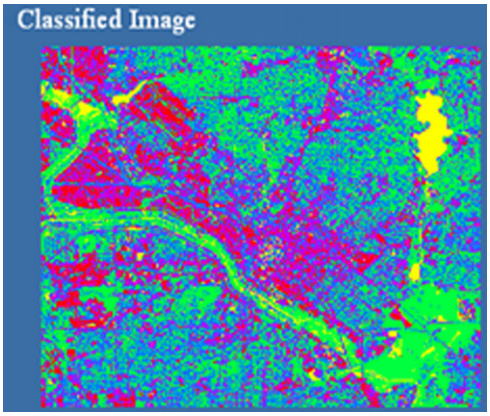

Fig. 6. Preview of output from an Unsupervised Classification of the Downtown Dallas, TX image.

When the operation is completed, the client will be informed about the output by a preview image on the web form. The user has the option to download it to the user's local disk drive if so desired Fig. 6.

While GWASS is able to employ the geoprocessing tools to process user uploaded data, its built in URL input capability also provides the user with choice to directly analyze data from distributed online sources, without being limited to the data on his/her own computer. Additionally, not only does GeoBrain have a large data repository, it also provides an online visualization system to search the data within the GeoBrain database through a system called GeOnAs. GeOnAs allows for interactive querying, viewing and selecting of various geospatial data sources, including Landsat, MODIS, and ASTER, based on user specified location and date. By doing so, there is no need to download the online data to a local desktop and then upload them to GWASS for analysis. After selecting an appropriate dataset in GeOnAs, a user can then simply check the data properties to obtain its underlying URL (Fig. 7). For instance, to perform a Stream Extraction analysis, a researcher could locate the tool under the Hydrological Analysis folder and then select a URL of an online DEM dataset (from GeOnAs in this case) as an input parameter (Fig. 8). The user then sets other parameters for the stream extraction geoprocessing tool, including threshold of flow accumulation (e.g., 300 pixels), output file format (e.g., GeoTIFF, ESRI Shapefile), output filename (e.g., stream300), etc. After all of the required parameters are set, the user presses the Submit button. The request is then sent to the GeoBrain server to launch the Stream Extraction web service, which internally combines a chain of multiple web services, such as flow direction, flow accumulation, flow accumulation thresholding and raster-to-vector conversion (if the output file format is ESRI shapefile). The resulting stream network in the selected output image format can be previewed in the GWASS system (Fig. 9(a)), or can optionally be downloaded in vector format (Fig. 9(b)). In addition to GeoBrain data, GWASS can also accept input data from other online GIS data repositories, such as ESRI, USGS EarthExplorer, NASA, NOAA, etc., as long as the input data is directly accessible through a specific URL.

To assess the performance of the GWASS system, we have tested the Drainage Basin tool under Hydrological Analysis with DEMs of various file sizes residing in different locations (Table 1). For a DEM located in the GeoBrain server with a size of 6340 kB, it took 28.12 s to finish the process, while for the same DEM located on a different website, it took 30.03 s to complete the task, indicating that approximately 2 s were used for pulling/uploading the data. For a DEM with a size of 15,558 kB, again with one local to the GeoBrain server and the other remote, it took 56.24 s and 66.17 s to complete the process, respectively. For a DEM with a size of 106,687 kB, it took 215.31 s and 275.22 s to process the data, respectively. Percentages of time used for uploading the DEMs to the GeoBrain server are 6%, 15%, and 22%, respectively, suggesting the overhead for processing data distributed in other locations increases much less drastically than the size of the data files themselves.

As a web application, GWASS allows simutaneous access to the same geoprocessing tools by many users. GeoBrain is deployed in a cluster computing environment with multiple servers to support concurrent request of a web service. To evaluate this capability, we tested the Drainage Basin tool in a classroom with a total of 25 students each individually processing a 6,340 kB DEM located in the GeoBrain server. The minimum, maximum, mean and standard deviation of the processing time are recorded for 10, 20, and 25 concurrent users (Table 2). When there were 10 concurrent users, the average responding time was 38.48 s (with a range from 17.00 s to 59.61 s). With the number of concurrent users doubled to 20, the average processing time increased 63% to 62.27 s (ranging from 21.16 to 94.12 s). With all 25 concurrrent users, the average processing time increased correspondingly to 89.87 s (with a range from 27.17 to 134.16). Considering one user needs 28.12 s to complete the process, and the maximum response time for 25 concurrent users was only 4.77 times larger (134.16 s), the capability of the GWASS system in handling simutaneous access to the same tool is quite impressive, primarily owing to the underlying cluster computing environment.

5. Conclusions and future work

GWASS is a web-based application that brings live the functions of the well-known GRASS GIS system to GIS end users based on the latest service oriented architecture. GWASS is made possible through the GeoBrain web service, which comprises an open, distributed, standard compliant, multi-tier, web-service system that provides a data-rich and on-line learning and research environment for Earth System Science (ESS) education and research. The purpose of GWASS is to provide a GUI based GIS system over the Internet. Any GIS end users with an

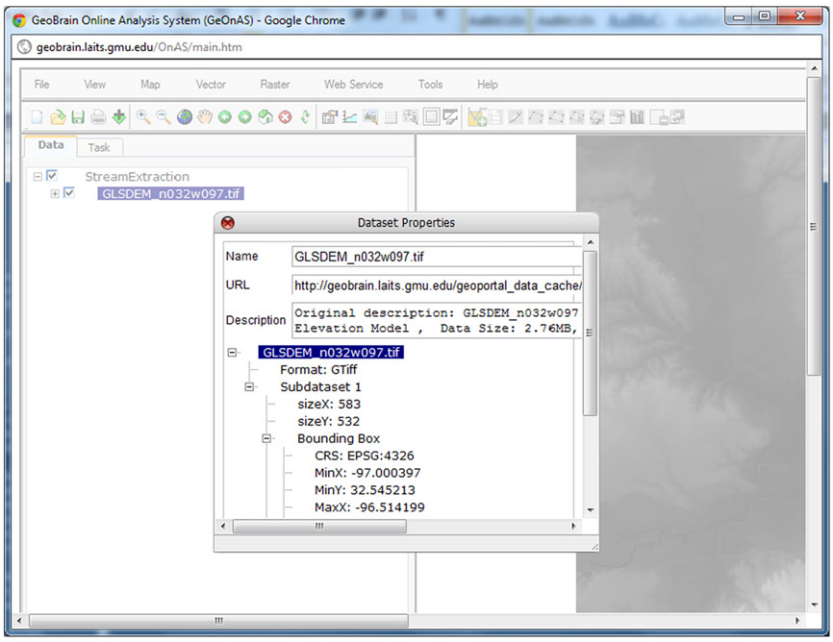

Fig. 7. Selecting a data and obtaining its URL from GeOnAs.

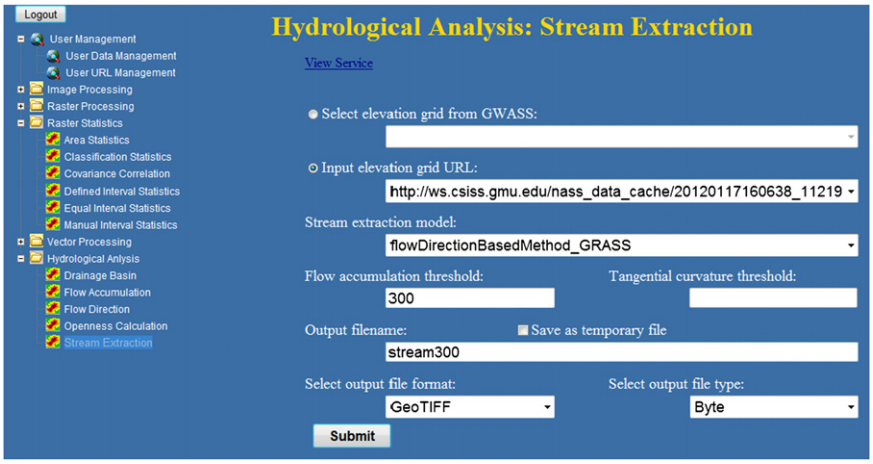

Fig. 8. Using a distributed online source as an input parameter for the Stream Extraction geoprocessing tool in GWASS.

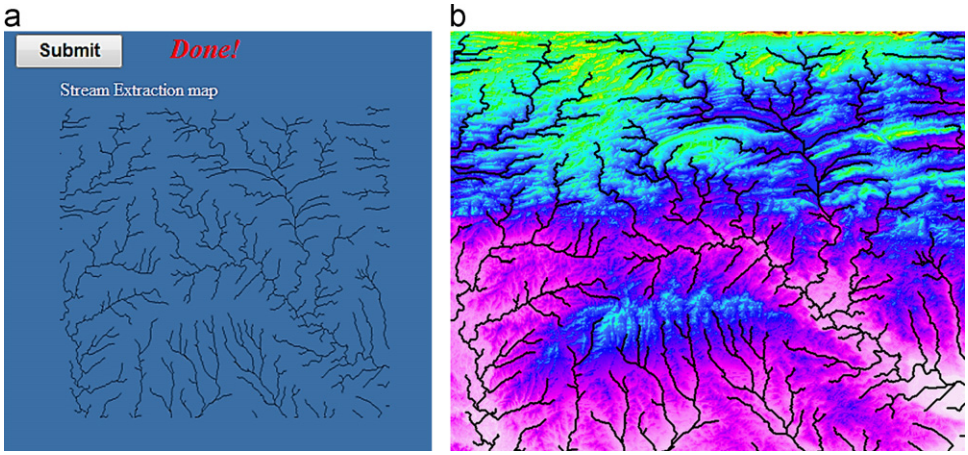

Fig. 9. Results from the Stream Extraction Tool in GWASS.

Table 1
Performance of the drainage basin tool in GWASS in completing the task with different file sizes and data sources.

	Size of DEM		
Data sources	6340 kB	15,558 kB	106,687 kB
GeoBrain server	28.12 s	56.24 s	215.31 s
Other server	30.03 s	66.17 s	275.22 s
% of uploading time	6%	15%	22%

Table 2
The concurrent processing time of GWASS with multiple users simultaneously accessing the same geoprocessing tool.

# concurrent user	Minimum	Maximum	Mean	Std. deviation
10 users	17.00	59.61	38.48	14.23
20 users	21.16	94.12	62.27	23.34
25 users	27.17	134.16	89.87	31.24

Internet connection can use the system to process large volumes of image, raster, grid3D, and vector GIS data without installing any desktop GIS/RS programs on their computers. Currently, the intended users of GWASS are faculty members and students in community colleges and teaching universities who cannot afford to run and maintain expensive commercial desktop GIS/RS software systems. It is also an ideal proof-of-concept tool for research universities that are starting exploration of geospatial data to solve their real world problems.

Although far from being mature, GWASS demonstrates that SOA-based GIS offers an effective and practical alternative to current commercial desktop GIS solutions. It shares development philosophies with the open-source software development community and builds a paradigm based on web service to access the functions of the popular open-source GRASS GIS package. The SOA framework is revitalizing the GRASS software system as a new means to bring powerful geospatial analysis and resources to more people around the world.

Future improvements to GWASS are anticipated in the following aspects. An issue in implementing the GWASS system is the transportation of large volumes of data over the Internet. File size is usually not a problem for desktop-based applications, which can easily process data files over hundreds of mega-bytes with the support of the native operating system and hardware configuration. However, for web-service enabled GIS systems allowing multiple user access, the problem of data transportation may be a concern. Users must allow additional time to upload/download data and to view images if their size is significantly large during a busy hour. For large data that do not reside in the GeoBrain server, but need to be processed many times with different parameters, the data should be uploaded into the GeoBrain server first to avoid repetitive downloads each time the data is processed. For data already residing in the GeoBrain server, the processing time is the same as that of the local data, unless the output data needs to be downloaded to the user side.

GWASS is constructed on top of open-source software and open standards. At present, the data being processed are accessed from URL links, which implies that security of the system can be a problem, which should be addressed in order for the system to be widely adopted.

Currently, GWASS provides two viewing capabilities to visualize either the re-sampled, lower resolution or the original full resolution images. These capabilities are very useful for the web-based system, but they are still not as convient as desktop-based applications because vector and grid3D data must be converted to raster images before they can be rendered on-line.

The GeOnAS previously discussed (Di et al., 2007) is a web-based desktop-application-simulation system, facilitated with improved geospatial data visualization on the Internet. GeOnAs provides robust display and visualization capabilities for online data, and has direct access to some limited geoprocessing capabilities. However, it currently does not allow users to upload and analyze their own data, but only existing data in the GeoBrain database. Conversely, GWASS has many geoprocessing functions, and allows for analysis of uploaded custom data, although it is somewhat limited in data visualization. As GeoBrain based subprojects that are developed independantly in parallel under different programming environments, GWASS and GeOnAS should be further integrated to better support users' needs in both data visualiztion and analysis. Through a loosely-coupled integration, GWASS and GeOnAs now complement each other by allowing users to reference datasets and access functions in the other system from external links. However, a more tightly coupled integration is desired in order to bring the advantages of the two systems within a unified environment and to form a more powerful single system in the future.

References

Bergenheim, W., Sarjakoski, L.T., Sarjakoski, T., 2009. A Web Processing Service for GRASS GIS to Provide on-line Generalisation. In: 12th AGILE International Conference on Geographic Information Science, Hannover, Germany, pp. 10.

Brauner, J., Schaeffer, B., 2008. Integration of GRASS functionality in web based SDI service chains. FOSS4G 2008, OSGeo, Cape Town, South Africa, 420–429.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S., 2002. Unraveling the Web Services Web: An Introduction to SOAP, WSDL, and UDDI. IEEE Internet Computing 6, 86–93.

Di, L., 2004a. Distributed Geospatial Information Services-Architectures, Standards, and Research Issues. In: Proceedings of the 20th ISPRS Congress, Istanbul, Turkey, pp. 187–193.

Di, L., 2004b. GeoBrain: A Web Services based Geospatial Knowledge Building System. In: Proceedings of NASA Earth Science Technology Conference 2004, Palo Alto, CA, pp. 1–8.

Di, L., Zhao, P., Han, W., Wei, Y., Li, X., 2007. Web Service-based Online Analysis System (GeOnAS). In: Proceedings of NASA Earth Science Technology Conference 2007, College Park, MD, pp. 1–7.

Di, L., Zhao, P., Yang, W., Yue, P., 2006. Ontology-driven Automatic Geospatial-Processing Modeling based on Web-service Chaining. In: Proceedings of the Sixth Annual NASA Earth Science Technology Conference, College Park, MD, pp. 1–7.

ESRI, 2011. Environmental Systems Research Institute, 〈http://www.esri.com〉.

GeoBrain, 2011. NASA EOS Higher-Education Alliance (NEHEA) – GeoBrain. George Mason University, 〈http://geobrain.laits.gmu.edu/〉.

GRASS, 2011. Geographic Resources Analysis Support System, 〈http://grass.itc.it/〉.

Li, X., Di, L., Han, W., Zhao, P., Dadi, U., 2010. Sharing geoscience algorithms in a Web service-oriented environment (GRASS GIS example). Computers & Geosciences 36, 1060–1068.

Mitasova, H., Mitas, L., Brown, W.M., Gerdes, D.P., Kosinovsky, I., Baker, T., 1995. Modelling spatially and temporally distributed phenomena: new methods and tools for GRASS GIS. International Journal of Geographical Information Systems 9, 433–446.

Müller, M., Bernard, L., Brauner, J., 2010. Moving Code in Spatial Data Infrastructures – Web Service Based Deployment of Geoprocessing Algorithms. Transactions in GIS 14, 101–118.

OGC, 2007. OpenGIS Web Processing Service. Open Geospatial Consortium, Wayland, Version 1.0.0. Document number OGC 05-007r7.

Pallos, M.S., 2001. Service-Oriented Architecture: A Primer. eAI Journal 9, 32–35.

PyWPS, 2011. The Python web processing service, 〈http://pywps.wald.intevation.org/〉.

Zhao, P., Deng, D., Di, L., 2005. Geospatial Web Service Client. In: ASPRS 2005 Annual Conference, Baltimore, MD.