ELSEVIER

Computers & Geosciences 32 (2006) 1403–1410

COMPUTERS & GEOSCIENCES

www.elsevier.com/locate/cageo

A generalized web service model for geophysical data processing and modeling

Igor Morozov*, Brian Reilkoff, Glenn Chubak

Department of Geological Sciences, University of Saskatchewan, 114 Science Place, Saskatoon, Sask., Canada S7N 5E2

Received 29 July 2005; received in revised form 23 October 2005; accepted 21 December 2005

Abstract

A web service model for geophysical data manipulation, analysis and modeling based on a generalized data processing system was implemented. The service is not limited to any specific data type or operation and allows the user to combine ~190 tools of the existing package, and new codes easily includable. It allows remote execution of complex processing flows completely designed and controlled by remote clients who are presented with mirror images of the server processing environment. Clients are also able to upload their processing flows to the server, thereby building a knowledge base of processing expertise shared by the community. Flows in this knowledge base are currently represented by a hierarchy of automatically generated interactive web forms. These flows can be accessed and the resulting data retrieved by either using a web browser or through API calls from within the clients' applications. The server administrator is thus relieved of the need for development of any content-specific data access mechanisms. The underlying processing system is fully developed and includes a graphical user interface, parallel processing capabilities, on-line documentation, on-line software distribution service and automatic code updates. Currently, the service is installed on the University of Saskatchewan seismology web server (http://seisweb.usask.ca/SIA/ps.php) and maintains a library of processing examples (http://seisweb.usask.ca/temp/examples) including a number of useful web tools (such as UTM coordinate transformations, calculation of travel times of seismic waves in a global Earth model and generation of color palettes). Important potential applications of this web service model for building intelligent data queries, processing and modeling of global seismological data are also discussed.

Keywords: Web service; Geophysics; Seismology; Processing

1. Introduction

The idea of distributed and integrated computational infrastructure for geoscience has been broadly discussed in the recent years, with spectacular progress made in the GIS community (e.g., Bachula, 2000; McKee, 2003). The need for such an infrastructure is dictated by explosive growth of geospatial and geoscience-related data sets distributed across the Internet and by the abundance of code needing interfacing, benchmarking and maintenance. Among the most notable efforts for such data and software integration is the computational infrastructure for geodynamics (CIG) initiative

*Corresponding author. Tel.: +1 306 966 2761; fax: +1 306 966 8593.
E-mail address: Igor.morozov@usask.ca (I. Morozov).

0098-3004/$ - see front matter © 2006 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cageo.2005.12.010

(http://www.geodynamics.org/) funded by the Geoinformatics program of the US National Science Foundation. Among its principal goals, CIG focuses at formulation of a software framework to interlink multiple codes and data crated and maintained by the community. Within a narrower scope of geophysics, we have also been persistently working towards software integration, starting from a multicomponent seismic processing system (Morozov and Smithson, 1997) which was later generalized into a geophysical software framework that effectively implements the same goal (Chubak and Morozov, in press). Here, we describe a novel feature of this framework that allows introduction of web services as a part of remote or distributed geophysical data analysis.

Web services (such as numerous search engines, web forms and online stores) are pervasive in today's world-wide web. With the help of a web service, a client with a web browser becomes able to execute programs on the server side, thereby taking advantage of its database or computational resources. The communication is performed through the standard HTTP protocol and is (relatively) safe, as the server performs only a limited number of predefined operations.

In geoscience, and especially in seismology, the use of web services is still mostly limited to rudimentary HTML form postings (e.g., see the Incorporated Research Institutions for Seismology (IRIS) data request portal at http://www.iris.edu/data/req_methods.htm). However, taking a somewhat broader look at web services as distributed data manipulation, in becomes clear that this model can hardly be sufficient. Compared to the traditional web services (e.g., online ticket bookings), geophysical processing and modeling requests are complicated by the variety of uncertain properties expected from the data, and by the infinite variety of operations the client might wish to apply to them. These differences suggest a more sophisticated and general model of a web service that would allow complex custom server-side processing controlled by the user. The benefits of such a general approach would be quite significant. For example, by preprocessing earthquake data (extracting event windows of interest, loading header values and performing custom quality control or plotting) on the data center server, the user might often be able to reduce the size of the request by as much as 1000 times. A seismic data request shipped on several DLT tapes today could be easily delivered via a web browser. Finally, a mechanism for complex custom web requests could facilitate on-demand modeling using large shared computational resources that are not available to the individual users.

2. Specialized vs. universal processing services

By the degree of granularity of distributing their data and operations, web services can be broadly subdivided into two end-member categories. In the first, most common approach, servers perform only relatively simple operations that are presented to the clients through APIs or web forms. Servers are, thus, specialized to narrow specific tasks, and clients are principally responsible for combining the data manipulation steps. IRIS data handling interface (based on CORBA methodology; http://www.iris.edu/DHI) is an example of such an approach in observational seismology. One problem with such an approach is that as the needs of the users grow, new codes and APIs need to be constantly added to the servers. With large data sets, volumes of data transmitted between the tools may also become significant in this service model. As we are looking for a "processing" service offering hundreds of operations and potentially combining dozens of them at any time, such a finely granulated service model appears to be far from optimal.

The only case where such a small specialized service has no alternative is in operating low-power instrumentation with severely limited resources. However, in this paper, we are interested in more user-oriented "data analysis" services that could involve distributed, multi-stage data processing relying on least average computational server resources.

By contrast to the model above, the service presented in this paper is inspired by the way modern seismic processing is typically performed. In this model, only one type of server is ever needed, but such that it is capable of performing complex, fully user-defined, multi-stage processing. Specialization of the servers and restrictions on their functionalities are imposed by the available resources and security considerations but not by the design of their codes. In this model, client programs (if used) represent "mirror images" of the server processing, and users formulate and execute processing sequences as if they were working directly on the servers. Client programs are identical to those of the server and are also capable of carrying out all of the basic processing.

Several significant advantages are associated with this architecture: (1) all services have the same user interface and functionality and are thus easier to learn, use and manage, (2) programming the services is performed by a data processor at application level and does not involve computer programming, (3) network traffic is minimized by performing as much processing on the servers and/or clients as possible, (4) new services can be added by combining data manipulation tools without adding new computer codes, (5) only one type of code needs to be installed on both servers and clients, and (6) as all servers offer only one functionality, no special service registration is required. Also, clearly, any tasks performed by the specialized servers above can be carried out by such a "universal" server, and the tasks can be programmed by the clients without modifications to the server code.

The only downside of this approach is the complexity of the code, potentially requiring more attention to its installation and maintenance. However, the experience with building seismic processing systems (commercial, such as ProMAX, Focus and Omega, and academic such as Seismic Un*x, SIOSEIS and SIA) shows how such code can be consistently developed and maintained. Modern software development packages (GNU compilers, utilities and Qt used in this project) are mature, powerful and well-supported on many platforms, and thus maintaining the code does not present any difficulties (although our recent development has been carried out under Linux, earlier versions of the system also operated on Sun, SGI and IBM machines; Morozov and Smithson, 1997). At present, the package described below includes a suite of development utilities and a even code maintenance web service (http://seisweb.usask.ca/SIA/cs.php) making its installation a simple three-step procedure. Distribution packages can be customized for each installation and, therefore, different web servers could have different tool configurations depending on their needs. The service also supports daily automatic code updates and bug reporting, in a way similar to most modern desktop software.

In summary, with its providing unlimited scalability, efficiency, minimizing network traffic and codes kept constantly up to date, the concept of a general-purpose processing service appears to be the clear winner over the small specialized server model, at least for the broader "processing service" applications. Below, we describe an implementation of such a service in our processing system (Morozov and Smithson, 1997).

3. General geophysical data/modeling web service

Our implementation takes advantage of what apparently is the most general, flexible and, practically, content-agnostic geophysical data processing system available to date, called SIA (Morozov and Smithson, 1997; http://seisweb.usask.ca/SIA). The system supports batch and graphical user interface (GUI) operation, includes capabilities for parallel processing, large shared libraries and ~190 precompiled and dynamically linked plug-in tools. Although started as a replacement to DISCO reflection processing system, its unique back-propagation processing strategy and abstract data model (Morozov and Smithson, 1997) led us to extending SIA to the analysis of wide-angle and earthquake records, travel-time, velocity models, graphics and adding interfaces to popular academic applications (such as the generic mapping tools, GMT; Wessel and Smith, 1995). The package has been extensively used for over 10 yr in many projects ranging from 2D and 3D reflection seismic and ground penetrating radar to wide-angle and teleseismic data analysis. Parallel 1D and 3D finite-difference modeling, and 2D and 3D potential-field data analysis were also recently included in the system.

By contrast to other processing systems built around some specific data file format (such as Seismic Un*x), SIA design is independent of the data type, and focuses on supporting serial, step-by-step processing using highly integrated tools and development libraries. Significant efforts have been applied to streamlining the process of code development, and to providing online documentation (http://seisweb.usask.ca/SIA/index). The system is implemented in C++, with some legacy C and Fortran codes. The use of object-oriented, compiled and optimized code base with common-to-all tools address space, ensures high stability and computational efficiency. Further descriptions of the SIA processing framework and code design are beyond the scope of this paper, and for a recent overview we refer the reader to Chubak and Morozov (in press).

The new web service utilizes the ability of SIA to build complex processing sequences from a library of specialized tools using textual flow descriptions. Such batch-mode execution is also typical, although not very commonly used in other seismic processing

systems (such as ProMAX, Focus, Omega and Seismic Un*x). In the scripts, the user describes the desired processing sequence, which could consist of data input/output, various types of filtering, database operations, constructing geophysical models, plotting, etc. The scripts can be created by either using any text editor (Morozov and Smithson, 1997) or a specialized flow editor in the client's GUI (Chubak and Morozov, in press). To implement web service operation, we simply implemented sending and receiving such processing flow descriptions encoded in HTTP messages.

The processing web service is contacted via a PHP program (http://seisweb.usask.ca/SIA/ps.php) that executes SIA batch processes configured with full or restricted data processing capabilities. If some job flow scripts are already present on the server, they can be invoked by using HTTP "GET" requests. For example, such requests can be issued from a web browser (such as http://seisweb.usask.ca/SIA/ps.php?job = examples%2Fs13%2Fe1.job&user = CG&exec&par = Hello%20CG%20reader for our "Hello world" example in http://seisweb.usask.ca/temp/examples). However, if real processing is desired, sending long job scripts from a web browser may become cumbersome and error-prone. An identical copy of SIA at the client's end performs such submissions. Another important advantage of using a full-featured system as a client is also its ability to perform much of the processing locally, with approaching the server only when special resources are required.

To initiate a service request from an SIA client, the user executes a processing flow containing one or several instances of a tool named "web service client" in the GUI (Fig. 1a). Each of these tools accesses another flow, also built by the user (Fig. 1b), and submits it to the corresponding web server for execution. In addition to the flow, the client can also transfer command-line parameters and several data files (currently, only ASCII) to the server. The server flow (Fig. 1b) is not limited in its functionality and can perform, e.g., seismic modeling, processing, database operations, or create PostScript or GMT images. Contacting the server is implemented via a series of background HTTP "GET" requests (could be changed to "POST" in later revisions) and, therefore, the service can operate as an application programming interface (API). Finally, one flow can simultaneously request different (or the same) processing from multiple servers.

Our present implementation designed a prototype for an intelligent, semi-interactive data mining from an online data repository, such as the IRIS data management center (DMC). In this application, the purpose of the remote job is typically to extract from the server database a set of data records answering some user's selection criteria, create plots and tables, maybe produce some synthetics and return all these items to the user. Such data delivery is implemented by creating an HTML index page into which the various

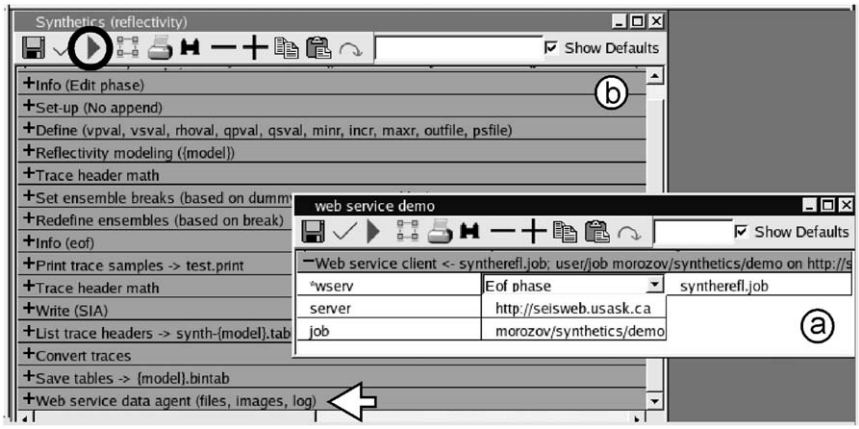


Fig. 1. Fragment of client's SIA GUI window showing (a) client and (b) server processing flows performing a web service. When flow (a) is executed, "web service client" tool sends flow (b) to remote server. In this example, server process performs 1D waveform modeling using *reflectivity* method (Fuchs and Müller, 1971). Note the complex modeling sequence, with symbolic substitutions, several data transformations, output and plotting performed. Results (model, output seismic data files, images and processing log) are posted on returned web page by "web service data agent" tool, marked with an arrow in flow (b). Note that if flow (b) requires no special system environment (computing power, databases), the client is also able to execute it on its local network (by pressing circled "Play" button).

tools can print and post links to their data files (Fig. 2, bottom). The resulting data and image files are posted on this page by a "data agent" tool included in the server flow (marked with an arrow in Fig. 1b), and the tools can also print information directly into the index file (Fig. 2, middle and bottom). The data agent can also create a compressed archive file of all the outputs for easier data transfer.

If the job takes a long time to finish, a self-refreshing page is returned by the server asking the user to wait. Some of the tools (such as file input or some modeling) have built-in progress indicators, and thus they can advise the user of the estimated waiting times. After the remote job is completed, the server returns an HTML document containing links to this and other files of interest to the user (Fig. 2, top and middle). The user can then assess the results and manually download the resulting data files as necessary.

Currently, our service executes two principal types of requests: (1) run a data processing flow supplied by the client and (2) run a flow whose description is already present in the server's database. Note that virtually any type of server action falls into one of these two categories, and so no other requests are needed. In the first of these cases, the request includes the entire server flow (Fig. 1b) encoded in a web-friendly XML format. The second request type is used to invoke some standard server operations, such as retrieval of common parameters shared by the server-side flows (see http://seisweb.usask.ca/SIA/ps.php?info, which is implemented through a pre-configured processing flow info. job residing on the server). Such requests are also used to execute the service from a web browser, by using the processing scripts and web tools already placed on the server (Fig. 3; http://seisweb.usask.ca/temp/examples). Note that, as explained below, any client can contribute flows to this library by using requests of the first type above.

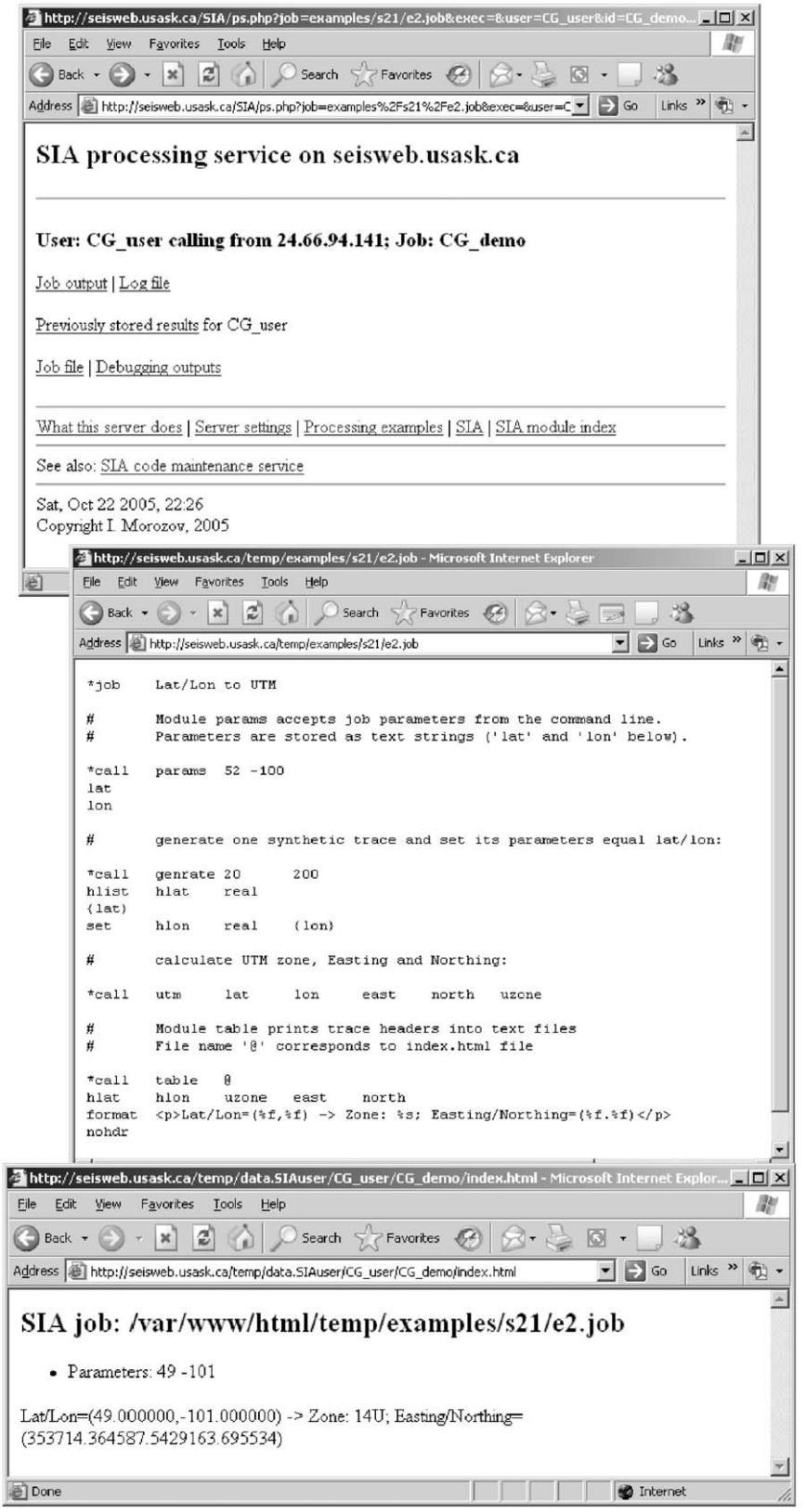
To identify itself, the client passes to the server its desired user and job names (fields 'user' and 'job' in Figs. 1a and 3) along with the processing flow string. These names are used to create a unique directory for the data and the output HTML pages. All other operations, such as user directories cleanup, logging, retrieval of server info. and messages to the server administrator, are performed by the various tools included in the remote flow.

In order to customize the user's processing environment (such as data directories, file names, or local database call formats) a method generally similar to HTML server side includes (SSI) is employed. Symbolic names for the various items in the environment can be defined by its administrator and made available to all server flows (http://seisweb.usask.ca/SIA/ps.php?info). These definitions can also be uploaded by the clients during the job posting process below.

Posting processing jobs is an important example of using custom remote flows for server-side operations. A user who has built a specific processing sequence (such as a special type of data retrieval or a standard simulation) may want to share this sequence with others. This is achieved by simply placing a posting tool "expert" into the server flow (Fig. 1b). When received and executed by the server, such a flow creates a cross-linked entry in a hierarchy of web forms and posts itself in this hierarchy (Fig. 3; also see http://seisweb.usask.ca/temp/examples). By opening these forms in a web browser, flows can be viewed and copied to the client (e.g., by dragging and dropping text into the GUI; Fig. 3, inset). At present, the primary use of this library is collection of processing examples and SIA usage hints by the Universities of Saskatchewan and Wyoming students. At the same time, some of these examples represent useful complete flows that can be parameterized and executed on the server (such as seismic synthetics in Fig. 3 or UTM coordinate calculators in *Coordinate Transformations* section). A simple keyword search utility (Fig. 3) extracts subsets of this flow library based on user's keywords. Note that this keyword search is also implemented via invocation of a server-side flow, and it can also be unclouded in client's remote processing flows (Fig. 1a).

4. Conclusions

The above-generalized processing web service design could offer dramatic improvements in the ways users could utilize shared community server's data or computational capabilities. Because processing flows can contain any combination of tools, virtually any type of data processing or modeling can be performed in the ways described above. Using such processing services, the user community would be able to build the knowledge base of data handling or modeling expertise without any


http://seisweb.usask.ca/SIA/ps.php?job=examples/s21/e2.job&exec=&user=CG_user&id=CG_demo...
File Edit View Favorites Tools Help
Back Search Favorites
Address http://seisweb.usask.ca/SIA/ps.php?job=examples%2Fs21%2Fe2.job&exec=&user=C
Go Links
SIA processing service on seisweb.usask.ca
User: CG_user calling from 24.66.94.141; Job: CG_demo
Job output | Log file
Previously stored results for CG_user
Job file | Debugging outputs
What this server does | Server settings | Processing examples | SIA | SIA module index
See also: SIA code maintenance service
Sat, Oct 22 2005, 22:26
Copyright I. Morozov, 2005
http://seisweb.usask.ca/temp/examples/s21/e2.job - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://seisweb.usask.ca/temp/examples/s21/e2.job
Go Links
*job Lat/Lon to UTM
Module params accepts job parameters from the command line.
Parameters are stored as text strings ('lat' and 'lon' below).
*call params 52 -100
lat
lon
generate one synthetic trace and set its parameters equal lat/lon:
*call genrate 20 200
hlist hlat real
(lat)
set hlon real (lon)
calculate UTM zone, Easting and Northing:
*call utm lat lon east north uzone
Module table prints trace headers into text files
File name '@' corresponds to index.html file
*call table @
hlat hlon uzone east north
format Lat/Lon=(%f,%f) -> Zone: %s; Easting/Northing=(%f.%f)
nohdr
http://seisweb.usask.ca/temp/data.SIAuser/CG_user/CG_demo/index.html - Microsoft Internet Explor...
File Edit View Favorites Tools Help
Back Search Favorites
Address http://seisweb.usask.ca/temp/data.SIAuser/CG_user/CG_demo/index.html
Go Links
SIA job: /var/www/html/temp/examples/s21/e2.job
Parameters: 49 -101
Lat/Lon=(49.000000,-101.000000) -> Zone: 14U; Easting/Northing=
(353714.364587.5429163.695534)
Done
Internet

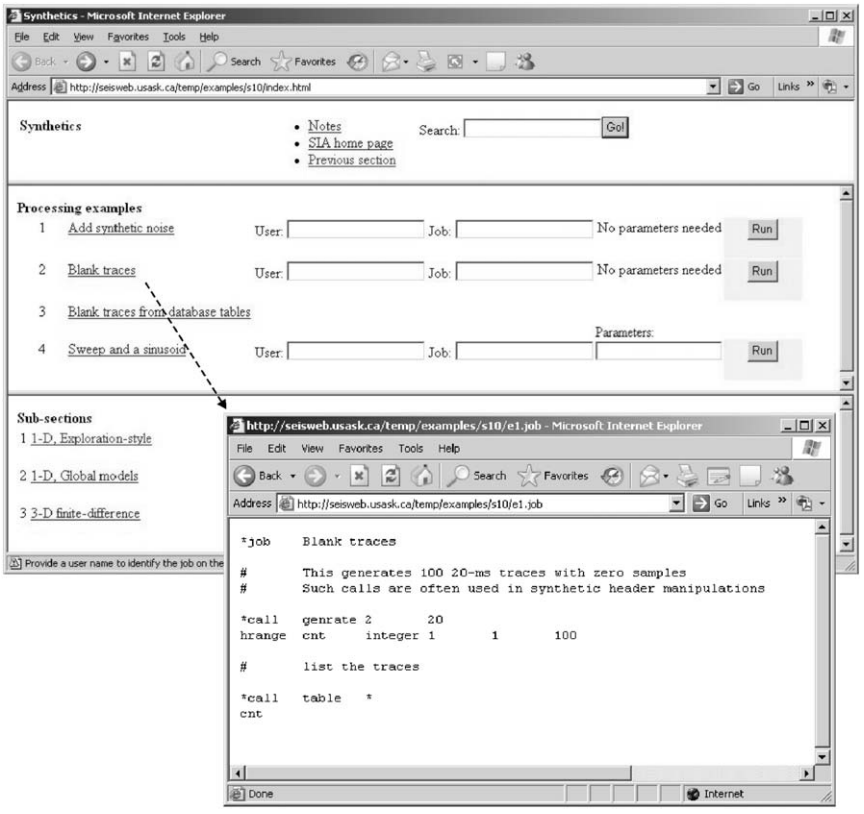


Fig. 3. Sample page (section "Synthetics") in current library of processing examples (http://seisweb.usask.ca/temp/examples). Such pages are generated by tool 'expert' included in processing flows executed on server. Larger window—in upper frame, comments to section (link "notes"), a link to system on-line documentation and a link to parent section are provided. The search tool is implemented by executing a predefined processing flow, search.job, on server, with its command-line parameters (key words) supplied by user. In middle frame, several simple synthetic jobs are shown. Using links, job scripts can be displayed (inset shows text of "blank traces" example). Flows that can be executed on server are provided with parameter input forms and "run" buttons. Bottom frame links this page to its sub-sections designed in same way. Inset—sample processing flow. Note that commentaries explaining processing sequence are provided. Such scripts can be dragged and dropped into client's GUI or a text editor, edited and executed (Chubak and Morozov, in press).

involvement from the server administrator. For example, currently, IRIS DMC staff is heavily involved in the development of seismic query and data delivery products, such as BREAQ_FAST (http://www.iris.edu/SeismiQuery/breq_fast.htm), BUD (http://www.iris.washington.edu/bud_stuff/bud/bud_start.pl) or WILBER (http://www.iris.edu/cgi-bin/wilberII_page1.pl). Although constantly improving, these tools still offer only very basic data selection capabilities and are not deemed perfect by the seismologists. This situation will probably remain the same as long as only a few developers continue striving to meet the needs of a large and heterogeneous research community. By contrast, in the new model proposed here, only a limited database access interface would need to be developed for SIA on IRIS platform and, thereupon, the clients would be able to build and deploy similar web tools entirely from their ends. Elegant interactive web forms similar to those mentioned above could also be created and deployed on the server by the processing jobs. This approach would

Fig. 2. Typical format of service results (single-point UTM co-ordinate conversion example). *Top window*—HTML document returned by server. It provides links to job script, output (index) and log files, user's directory, as well as links to service info and system help. *Middle*—current job script. Note that last tool (table) prints the results directly into output file, index.html, represented by symbol '@'. *Bottom*—job output file (index.html) containing output string. Other tools such as post, print and table, could make additional printouts and add links to output files.

foster a true community involvement, help quickly create numerous request mechanisms needed for various groups of users and relieve the data center administration of any need to carry out expensive development of user-end data access or processing.

Our present implementation is still principally intended to introduce the concept of general geophysical web-service data processing and to illustrate its key components. Note that unless the server can offer some special computational power or specialized databases, using this web service gives no advantage to an SIA user, as the client itself is also fully capable of executing the jobs (Fig. 1b). Our server platform is currently an old Pentium III PC not designed for serious seismic data storage or processing. However, this study already shows that both the concept and implementation of a general geophysical processing web service are fully functional. Its first application to developing a shared library of interactive web tools and examples represents a new and useful tool for teaching and research in geophysics. In the near future, we hope to collaborate with IRIS on applying this approach on a larger scale, to real on-demand global earthquake data mining, processing and modeling.

Acknowledgment

Constructive comments and suggestions by an anonymous reviewer have greatly helped in improving the original manuscript.

References

Bachula, G. R., 2000. Spatial data infrastructure: on the road to twenty-first century economic success. Open GIS Consortium White Paper, Open GIS Consortium, Inc. http://www.opengeospatial.org/press/?page=papers.

Chubak, G., and Morozov, I. B. Integrated software framework for processing of geophysical data. Computers & Geosciences, in press, doi:10.1016/j.cageo.2005.10.005.

Fuchs, K., Müller, G., 1971. Computation of synthetic seismograms with the reflectivity method and comparison with observations. Journal of the Royal Astronomical Society 23, 417–433.

McKee, L., 2003. The spatial web. Open GIS Consortium White Paper, Open GIS Consortium, Inc. http://www.opengeospatial.org/press/?page=papers.

Morozov, I.B., Smithson, S.B., 1997. A new system for multicomponent seismic processing. Computers & Geosciences 23 (6), 689–696.

Wessel, P., Smith, W.H.F., 1995. New version of the generic mapping tools released. EOS Transactions of the American Geophysical Union 76, 329.