Abstract

Where shortcuts as WMS, WFS or WCS are well known in our geoinformation vocabulary, WPS is still rather new. OGC Web Processing Service (WPS) is approaching from version 0.4.0 to version 1.0.0. A WPS can be configured to offer any sort of GIS functionality to clients across a network, including access to pre-programmed calculations and/or computation models that operate on spatially referenced data. A WPS may offer calculations as simple as subtracting one set of spatially referenced numbers from another (e.g., determining the difference in influenza cases between two different seasons), or as complicated as a global climate change model. This processes can be calculated on remote servers and so, the local stations can be common desktop computers.

With new versions of web 2.0 based map applications (e.g. Google Maps, etc.) end user is able to browse high interactive maps in web pages. Although it is possible to build custom map application with help of proprietary API, such as Google Maps API, Seznam.cz API or similar, it can often lead to legal issues. Adding custom data can be also not so straightforward. Furthermore support for OGC services is not extensive in such kind of applications.

In this article OpenSource tools, which are enabling building custom maps “widgets” at any web site, are introduces. The map can have look & feel, such e.g. Google Maps has, but without license issues.

Keywords: Web GIS, FOSSGIS, WPS, PyWPS, Web Services, OpenLayers, Postgis, JavaScript, PostgreSQL, TileCache, ExtJS, Web 2.0

Introduction

The Open Geospatial Consortium, Inc.® (OGC) is a non-profit, international, voluntary consensus standards organization that is leading the development of standards for geospatial and location based services. OGC specifications are technical documents that detail interfaces or encodings. Software developers use these documents to build support for the interfaces or encodings into their products and services. One of this document OGC Web Processing Service (WPS). It is relatively new standard (compared to other, more used standards, like OGC Web Mapping Service (WMS) or Web Feature Service (WFS)). The document number 05-007r4 describes version 0.4.0 of the WPS standard. Request for comments to this standard was published February 2006.
Nowadays Web GIS applications do require more than just displaying the map in web browser. The map has to be more and more interactive, with special controls and features. Users do want to add their own vector features to the map, like icons, paths, polygons with special attributes, and share their maps with other people.

We can say, that with Google Maps API [1], the way, how maps are embed into common web page, has changed completely. Smooth drag&drop panning, smooth map zooming, custom icons and the way, how the map navigation is done, this all features introduced revolution in Web GIS applications.

Several portals do follow this new map layout, however, there was no independent library, which would be usable for building custom applications without license issues. Also the way how to add custom layers was not so straight forward, how some developers would like. With OpenLayers, TileCache and other projects this became possible.

**OGC Web Processing Service**

The standard describes the way, how geospatial operations (referred as “processes”) are distributed across networks. Now days, there are only two implementation of the standard: 52north WPS and PyWPS. Also client applications, able to communicate with the server via WPS standard are only few. Interesting is, that also big projects and companies did ignore this standard till now.

The communications consists of three types of request-response pairs. Request can be in Key-Value-Pairs encoding (KVP) or it can be send to the server as XML document. Server response is always formated as XML document. Similar to other OGC specifications, the requests are GetCapabilities, DescribeProcess and Execute.

**GetCapabilities**

As response to the GetCapabilities client request, Capabilities XML document is returned. It consists of two main parts: ServiceIdentification and ProcessOfferings. The structure of ServiceIdentification part is mainly documented in OpenGIS® Web Service Common Implementation Specification. Server provider as well as constrains, fees and other additional data (if any) are listed here. In the ProcessOfferings part, list of on server available processes is appended. Client, which is able to parse the Capabilities document correctly, has list of processes offered by the server. After the document is parsed, it can request more detailed process description of selected processes.

**DescribeProcess**

At DescribeProcess request, ProcessDescription XML document will be returned. It contains detailed process characteristics, as it's title, identifier and abstract with more detailed informations. Also necessary inputs and their types are listed here. Results of geospatial calculations will be stored in Outputs section. Client, which is able to parse XML ProcessDescription document, gets general overview, which inputs it has to request from the user in order to be able handle them to the server and let the process to be executed. It can also prepare it self at the server response, because process outputs, their formats and types are described in the second part of the document as well.
**Execute**

At Execute request, client sends the input data and/or values to the server, waiting for the server response. After the final response with process outputs is obtained, client can display the data to the user.

In- and Output data can be of three different kinds:

- **LiteralData** – character strings, integer numbers as well as double numbers
- **ComplexValue and ComplexValueReference** – those are raster, vector or other (large) data files – maps in various formats.
- **BoundingBox** – two pairs of coordinates.

ComplexValue differs from ComplexValueReference in the fact, that the data are part of the request/response, where ComplexValueReference does handover only URL reference to the location, where the data can be downloaded.

**WPS Data Management models**

Different approaches can be taken, to pass input data to the server. They can be combined together, as well as only one approach can be selected.

For Input data:

- Data are stored on the server. So it looks at least, from the client point of view. No (spatial) input data are required to be send with the request to the server. The data can be stored at the server directly, as well as at remote servers and WPS server can request them via SQL, WCS or other standards. Client does not know anything about this process.

Example of the design can be seen in any Shortest path calculation: All necessary input data (road network) is usually stored at the server, clients overhands only start and stop coordinates.

- Data are shipped as part of the request documented. In this case, request is usually send as XML document. Vector data, can be stored in several XML formats, namely GML, SVG, GeoRSS and others. This XML documents can be included into XML request as its part. For raster data, CDATA section can be used, in which any text or binary data can be stored.

For example, buffer process requires input vector object, so it can make buffer around it.

- Only references are part of the request document, server will have to download them from remote server. This enables us, to make the client as simple as possible: No data do have to be stored at client side. Client can only point to one service to another (WPS to WCS or WFS in this case).

For example, image classification process can get URL to Web Covarage Service, where it gets necessary input raster maps.

This last example demonstrates in our opinion the power of web services and their possible usage. Web services are bind together to provide working solution.

Above mentioned list implies also for data outputs. Output XML file can contain directly vector or raster maps, embed as included XML or as content of the CDATA object. It is also possible, to send only references to output data, which can be later downloaded. Finally, server can return raw data directly, as response to the WPS Execute request.
**Asynchronous vs. Synchronous process execution**

Some geospatial calculations can take longer time, in terms of hours, days or even weeks. This can easily lead to Server Timeout error and also, it is hardly possible to get more information about the calculation progress. Therefore “status” input parameter can be used. If set to “true”, then (if supported) the server will send Execute response back immediately after it obtains client's request with a message, that the request was accepted together with an URL, where the client can regularly check calculation progress and will find out what is happening as well as guessed percentage of remaining time.

**Issues of OGC WPS 0.4.0 standard**

Christopher Michael and Daniel P. Ames described in their paper six proposals to WPS 0.4.0 standard, which are trying to fix some missing features of the format. Namely

- Adding section, which informs client application, how the request input data from the user. Currently, only data types are supported (String, Complex data). But e.g. number format (integer or double format), raster or vector data selection is missing.
- New PromptMethod element is proposed, which would include defined type of user input, such as “browseforraster”, “browseforvector” or “getboundingbox”.
- Adding section of data available at the server, so the client could make data selection by itself.
- Returning raw data as Execute response should be avoided – XML document should be always returned back.
- It is not possible to cancel running process. Methods for calculation break should be added as well.
- Each process should have single URL – currently, all processes do have common URL. This enhancement would make the WPS standard inconsistent, if compared to other OGC standards.
- More highly structured exception system should be implemented. Current system is only primitive and offers Exception codes only for limited number of cases (ServerBusy, FileSizeExceeded, InvalidParameterValue, NoApplicableCode).

Also other improvements, such as missing internationalization support, could be implemented in new standard versions. Some of them were included in to OGC WPS 1.0.0, which was published in the summer 2007.

**OGC WPS 1.0.0**

New version of the standard [3] introduces new features, which are enlarging all possible ways of WPS usage, as well as making life of programmers easier, namely:

- SOAP and WSDL usage. SOAP (Service Oriented Architecture Protocol) is a protocol for exchanging XML-based messages over computer networks, normally using HTTP/HTTPS. SOAP forms the foundation layer of the Web services stack, providing a basic messaging framework upon which abstract layers can be built. The Web Services Description Language (WSDL) is an XML-based language that provides a model for describing Web services. The WSDL defines services as collections of network
endpoints, or ports. WSDL specification provides an XML format for documents for this purpose. It describes the public interface to the web service.

**WPS implementations**

Before we start to talk about programs implementing WPS standard, we have to mention web-based geoprocessing systems and approaches similar to WPS, that have been implemented by various entities. Most notably, the Environmental Systems Research Institute (ESRI) product ArcInfo 8.3 (ESRI, 2003). It contains a feature called the Geoprocessing Server, which ran on large-scale UNIX servers to perform geoprocessing on behalf of ESRI client software which submitted jobs for processing. The ESRI Geoprocessing Server protocol is proprietary and closed such that only ESRI software is able to make use of the remote processing capabilities. Interestingly, this feature was removed from the following version (ArcGIS 9.0). A similar but subtly different feature was introduced in ArcGIS Server 9.2. However, unlike WPS, the ESRI implementation is not compatible with non-ESRI products and a closed, proprietary communications protocol preventing it from being adopted at large or studied in a non-ESRI environment.
Currently (autumn 2007), only several implementations of OGC WPS are known and all from them are implementing the 0.4.0 version of the standard. However it is known, that 52north WPS server development team is actively working on 1.0.0 WPS standard implementation.

Currently known WPS server projects:

- **Degeree framework** is Java environment, which implements most of OGC standards. It is used mostly in German state administration.

- **WPSint** a JAVA plug-in for Spring (full-stack Java/JEE application framework). Although it implements 0.4.0 version of the standard, it can be used via WSDL/SOAP interface.

- **PyWPS** states for Python Web Processing Service. It is written with direct support for GRASS GIS and other GIS command-line tools.

- **52 North WPS** is written in Java, as plug-in for Java Tomcat serverlet container. Part of 52 North WPS are also plugin-client applications, for uDig and Open-Jump GIS geodata viewers.
- **GeoConnections WPS** is implementation of one of the older versions of the standard (0.2.0). The project seems to be no longer supported.

**PyWPS** is project, which is developed since 2006, and tries to implement OGC WPS standard in it's 0.4.0 version. It is written in Python programming language. The main goal of PyWPS is, that it has been written from the beginning, with directo support for GRASS GIS. So, PyWPS can be understand, as kind of translation library, which translates requests complain to WPS standard, overhands them to GRASS GIS [9] or other command line tool (such as GDAL/OGR, PROJ.4 or R statistical package), monitors the calculation progress and informs the user and after the calculation is completed, it returns back it's result.

PyWPS released under terms of GNU/GPL licence. Currently, version 2.0.0 is available. It is actively maintained by Help Service - Remote Sensing company as one of our projects. All applications described in this paper are using PyWPS as back-end. Currently, we are continuing with PyWPS development and would like to introduce implementation of WPS 1.0.0 standard by Summer 2008.

**OpenLayers**

OpenLayers enables putting a dynamic map in any web page simple way. It can display map tiles and markers loaded from any source. MetaCarta developed the initial version of OpenLayers and gave it to the public to further the use of geographic information of all kinds. OpenLayers is completely free, Open Source JavaScript, released under the BSD License. From developer point of view, OpenLayers is a pure JavaScript library for displaying map data in most modern web browsers, with no server-side dependencies. OpenLayers implements an object-oriented JavaScript API for building rich web-based geographic applications, similar to the Google Maps and MSN Virtual Earth APIs. Furthermore, OpenLayers implements industry-standard methods for geographic data access, such as the OpenGIS Consortium's Web Mapping Service (WMS) and Web Feature Service (WFS) protocols and many others.
OpenLayers data types and features

OpenLayers is able to display various types of raster and vector data formats. Naturally, it supports OGC WMS specification, as well as common Image formats (in PNG, GIF or JPEG format). There is also support for multiple proprietary formats, like Google Maps, Yahoo maps and others. OpenLayers do use so called tiling of raster data.

Numbers of vector (and text) data formats are supported as well. There is possibility for rendering vector features in GML, OGC WFS, GeoRSS, KML formats. Creation of regular shapes (boxes, circles, ...) is supported as well. Points can be displayed as special point features with image icon or like vector point features.

Numbers of controls are available to support map interactivity and customization. Among others, zoom bar, overview map, layer switcher, various toolbars and mouse action handlers can be used. Thanks to advanced event model, it is no big problem to program custom map control.

OpenLayers data rendering

For rendering of raster data, so called image tiling is used. In each layer, the (raster) map is split into smaller images, ordered in to a grid. While panning, only new tiles are loaded and not whole map image.

The layers itself are ordered into stack – the layers are lying on top of each other. When one layer turned on/off, others are not affected.

For rendering of vector features, SVG (Scalable vector graphics - Firefox, Opera) and VML (Vector macro language - MS Internet Explorer) are used. This enables vector data editing directly in the web browser.

While the client rendering is due to tiling technology relatively fast and smooth, on the server side, the server load is n-times higher, where n is number of generated tiles per map view. One of possible solution is to generate map tiles in advance and display done tiles in the map directly. To generate pre-cached tiles is possible with e.g.
**TileCache**

TileCache is an implementation of a WMS-C compliant server made available under the BSD license by MetaCarta. It provides a Python-based WMS/TMS server with caching mechanisms and rendering backends. Local disk-based cache of any WMS server can be created, and use the result in any WMS-C supporting client can be used. The server load decrease significantly, because no map server process has to be started – data are already prepared in image form. One of the disadvantages of such solution is, that the data must be pre-generated in fixed zoom steps. Also reprojection of such pre-processed data is not later possible.

**WPS Demo**

From summer 2006, WPS Demo application can be used, which was developed by Help Service Remote Sensing company. The demo is continuously updated, so it can demonstrate all WPS possibilities. Today's version uses PyWPS 2.0.0 as back-end and OpenLayers map client as front-end for displaying the data. The application is entirely written using JavaScript language. All processes are using GRASS GIS as background geoprocessing tool and GDAL/OGR and PROJ.4 libraries, for data transformation.

Currently available processes are:
- **Shortest path** calculation, which uses v.net.path module of GRASS GIS. The output vector file is exported as GML vector format which is embed into Execute Response.
Unsupervised image classification is another process. The user has to zoom to specified region and then she can process the classification by setting desired number of classes. One of the input parameters is WCS URL pointing to Landsat imagery file. When the file is downloaded and imported, unsupervised classification with help of i.cluster and i.class is performed. Resulting raster file is converted to PNG file and returned back to the client together with reference to original GeoTIFF file. PNG file can be displayed directly as one of the layers in OpenLayers Map object.

Visibility analysis works similar to previous process. The user have to define coordinates of the viewport (per mouse click) and after a while, resulting PNG file from r.los is displayed.

Flow analysis calculates raster map of flowlines, using r.flow module. User have to zoom to desired region. The flow analysis is done on the digital elevation model, stored on the server. Resulting GeoTIFF or PNG files can be then displayed by the client.

Prefarm

Prefarm is a project for providing optimal fertilization calculations. It is able to produce raster maps and tabular outputs, which can be used in the farmers decision process while finding the optimal fertilization variant. Output raster maps are later used in tractor computer together with GPS unit, so the fertilizer is distributed over the fields optimally.

Current version of Prefarm has two components:

1. Analytical part, which is done as plug-in for ESRI ArcView 4.0 program. The operator has to perform all necessary steps in order to get resulting maps and tabular data according to farmers needs. If some input parameter is changed by the farmer, the operator has to perform all calculations once again.

2. Visual part, which is done as web project using UMN MapServer for displaying maps, together with PHP and JavaScript. The farmer can check in the web browser calculation results and use the data in further decision process.

It is very complicated, or impossible to setup the analytical part of the system for more then one user. It is only hard imaginable to open the access to the application via Internet. Always there have to be person, which reacts at farmers needs and which do have to make the calculations manually.

In our currently developed version of Prefarm, we chose another approach, which is has also two main parts with one interlayer:

1. Analytical part is done as stand-alone scripts, using GRASS GIS tools. The scripts are wrapped in a file written in Python programming language. The scripts are doing special tasks, like creating raster maps for each nutrient and filling the PostgreSQL database with tabular data.

2. Visual part, which is done as web project using UMN MapServer for displaying maps, together with PHP and JavaScript. The farmer can check in the web browser calculation results and use the data in further decision process as well as recalculate the analysis with different input values (maximal or minimal amount of fertilizer, different fertilizer composition, ...).
3. PyWPS as interlayer, which translates requests from the web application, runs server scripts, displays calculation progress and returns the calculation results back to the web browser.

This new solution, based on OGC standards, is multi-user system. Each user (farmer) can run her real-time analysis, modify input data directly, version control and management and other benefits. Operator is needed only for special cases, like user and fields management. As there is only one installation of the application, upgrades and bug fixes are easy to maintain. Also from development point of view, as the application is modular (visualization part and calculation part, the calculation part has separated modules for each nutrient), we can work on it's parts without fear of breaking the whole system.
Touristic portal of region Posázaví

The portal of region Posázaví needed to display interactive map within web pages, which would be used for displaying points of interest (POI) data, as well as linear features (bicycle tours and other) on top of common topographical map as well as aerial images. Naturally, OpenLayers have been chosen, with several improvements, which do adjust the application behavior in the desired way.

Most of the point data are represented as OpenLayers.Layer.Text feature. It is possible to click on any feature in the map and get it's attributes. The map can be found at the address http://tourist.posazavi.com/cz/mapy.asp

OpenLayers was also used for displaying result of bicycle tours search tool. User can define starting and ending point of the tour, as well as in the middle point of interest. User can then adjust one of used points with mouse or with help of menu and in a few moments, map is recalculated.

Together with optimal path, near-by lying points of interest (0,5 km buffer) are displayed, as well as tour height profile. The tour calculation together with POIs list is done in PostgreSQL / PostGIS and UMN Mapserver as WPS process. Also path profile, calculated along the route, by sampling digital elevation model every 200 meters is returned.
Geo-game of Liberec region

Geo-game was designed as application for common web-site visitor, who wants to check his/her knowledge of this north bohemian region. The web page is split into two parts: in the left part, some phenomenon photographs, such as natural monument (cave, rock, lake, ...) or human imprint (church, castle, ...) are shown and in the right part, interactive map of Liberec region is displayed and user has to place marker on the position, he/she thinks, the phenomenon is located. Then the right location is displayed together with user set location and geographical distance is between set and real position is calculated. The users have to place up to ten marks and at the end, mean value of distances between reality and users guess. User can then store his/her name into table of best results.

The application was again built up with help of OpenLayers, PostgreSQL/PostgGIS database. Displayed data are combination of standard OGC WMS and precached tiles with help of TileCache. Markers are represented as pin images in the layer of type OpenLayers.Layer.Markers.
Geophotogallery

The geo-photo-gallery has been built for the Liberec region. It uses the same underlying database as geo-game. User may select some stored photogallery and then browse the photos different ways (selecting in the list or in the map view. Objects details are available anytime too.

The goal is to have symmetric pilot application (not map centric and not data centric) but with equilibrium of all components which are relative independent and may be combined according to user requirements.

The map is provided with help of OpenLayers, with custom layer switcher. All the graphical user interface was designed with help of ExtJS library, another handy tool for development of Web 2.0 types of applications.
Conclusion

Though WPS is relatively new standard, it is mature enough to be used in production applications. There are both, server and client implementations, which are actively developed. From usage of (OGC) standards do benefit both: users and code writers. Users (clients) can share data, informations and tools regardless who is providing the server solution. Code writers can make their applications easy interoperable and modularized.

In this paper, we introduced the standard and some of it's possible enhancements. Some of the standard implementations for both, client and server side, were described, with special attention to PyWPS.

Possible usage was demostrated on examples with PyWPS with both - proprietary (WPS Demo, Prefarm) and open clients (OpenLayers plugin).

Concrete benefits of WPS usage was demonstrated on Prefarm application, were current (old) solution together with new (developed) was compared.

Now days, applications are returning to the server-client concept. The WPS standard was the missing piece in the OGC Web Service standards, which makes it possible, to write thin client applications, which are using for analytical and data storage remote servers.

OpenLayers is growing to be one of the most interesting tools for embedding interactive map into custom web page. Even though some developers do complain, about it's performance (compared to e.g. Google maps), the situation is changing rapidly. Also the functionality of OpenLayers goes far behind any proprietary API.

Together with other tools (TileCache, ExtJS), we get completely OpenSource base for building Web 2.0 geoapplications.
Acknowledgements

The paper is prepared on the base of outputs from project: SPRAVA DAT - the solution was achieved with financial support from state resources provided by the Academy of Sciences of the Czech Republic for support of project of the program “Information Society” with registration number T2060300407 and name “Data Management”.

References


[3] Service for Use in a Client-Side GIS


