APPLIED FREE AND OPEN SOURCE SOFTWARE WEB MAPPING CLIENTS IN EDUCATION AND APPLICATION SIDES

Truong Xuan Quang, Maria Antonia Brovelli, Luana Valentini

DIIAR, Politecnico di Milano – Como Campus (Como), Italy.
xuan.truong@mail.polimi.it, maria.brovelli@polimi.it, luana.valentini@mail.polimi.it

ABSTRACT

The paper presents our approach in creating Web mapping clients using different Free and Open Source Software (FOSS) frameworks and the comparison among them. We used MapServer and HTML, MapServer with Chameleon, MapServer with p.mapper, MapServer and GeoServer both with Openlayers and MapFish framework. Thus, different frameworks implemented with the same data set are provided as useful examples for users who want to do the same cases of application; advantages and disadvantages of each framework to help users and developers in selecting the more suitable for their Web mapping are briefly presented. The examples are appropriated for education and training but also useful for small and medium size companies or organizations which want to start working on these topics.

1. INTRODUCTION

Free and Open Source Software (FOSS) is playing a very important role in development of Web mapping client, with the advantage of granting the right of users to use, study, change, and improve its design through the availability of its source code. With the increasing availability of spatial data and Web applications, FOSS for Web mapping have been rapidly increasing in quantity and quality.

We present a general comparison between some common adopted solutions, emphasizing advantages and disadvantages of each method. Our work represents a good example for education and training and it is also useful for small and medium size company to refer for their work.

Our research aims to apply the following technologies to same dataset: HTML and MapServer, Chameleon and MapServer, p.mapper and MapServer, MapSever and GeoServer both with Openlayers, GeoExt and MapFish.

2. SAMPLE DATA USED

To have a direct comparison among the different frameworks, the same sample data have been used in the WebGIS. In particular, we have decided to use the most common raster and vector formats (as listed below) provided by the Lombardia Region. In addition we have added some layers not stored in local but provided by geoservices through the Web, such as Web Map Services (WMS) and Web Feature Services (WFS).

The used data include:

- Vector data: ESRI shapefiles (municipalities, railways, lakes, railways stations) and MapInfo data (Alluvional cones).
- Raster data: Regional raster map (TIFF and TFW).
- Services include data source for railways and rivers provided as WFS by The General Direction of Soil Protection of the Ministry of the Environment (ME), Land and Sea.

Digital Terrain Model (DTM) provided as WMS by Lombardy Region, Orthophoto provided as WMS by (ME).

The described data belong to two different reference frames, the Gauss-Boaga Roma40 (meters, EPSG code: 3003) and the UTM WGS84 (meters, EPSG code: 32632, zone 32N). The aim of using data belonging to different reference systems is to show that the geographic servers can handle them in the same project, re-projecting the layers in the reference system chosen for the displayed map (in this case UTM WGS84).

3 . THE SERVER-SIDE

Among the different FOSS server software we have decided to focus on MapServer and GeoServer, since these are the most common geographic servers used and supported by a large developers community. The main features are presented in next sections.

3.1 MapServer

MapServer is an Open Source project whose purpose is to display dynamic spatial maps over the Internet. The main features are listed below:

- support for display and querying of hundreds of raster, vector and database formats;
- run on different operating systems (Windows, Linux, Mac, etc.);
- support of script languages and development environments (PHP, Python, Perl, Ruby, Java, .NET);
- high quality map rendering;
- on-the-fly projections;
- fully customizable application output.

Additionally, MapServer is a CGI program that sits inactive on the Web server. When a request is sent to MapServer, it uses information passed in the request URL and the mapfile to create an image of the requested map. MapServer supports numerous OGC specifications as WMS, WFS, WCS (*Web Coverage Service*) and SOS (*Sensor Observation Service*).

3.2 GeoServer

GeoServer is an Open Source software server written in Java that allows users to share and edit geospatial data. Designed for interoperability, it publishes data from any major spatial data source using open standards. GeoServer is the reference implementation of OGC Web Feature Service and Web Coverage Service standard, as well as a high performance certified compliant Web Map Service. Nowadays the performances of MapServer and GeoServer are comparable, as it has been seen in the 2010 FOSS4G benchmark (Aime, 2010).

4. THE CLIENT-SIDE

4.1 HTML and MapServer

A simple MapServer application consists of geographical data, HTML pages and a mapfile, that is a structured text file for MapServer configuration. It defines the map extent, the projection to be used, the data source, the layers that you want to display and how they have to be drown. HTML pages are the interface between the user and MapServer and they

normally sit in Web root. In its simplest form, MapServer can be called to place a static image on an html page. To make the map interactive, the image is placed in an html form on a page. Since CGI programs are 'stateless', every request they get is new. For this reason, every time the application sends a request to MapServer, it needs to pass context information (what layers are on, where you are on the map, etc.) in hidden form or URL variables.
The simplest MapServer CGI configuration may include two html pages:

- the initialization file uses a form with hidden variables to send an initial query to http server and MapServer;
- the template file, that controls how the maps and legends output by MapServer will appear in the browser. The template also determines how the user can interact with the MapServer application (browse, zoom, pan, and query).

4.2 Chameleon with MapServer

Chameleon is defined as a distributed highly configurable environment for developing Web mapping applications. It is built on MapServer as the core mapping engine and works with all MapServer supported data formats through a regular mapfile. An advantage of Chameleon is the possibility to quickly setup new applications from a common pool of widgets that can be placed in an HTML template. These widgets provide a fixed piece of functionality, but the representation of the widget is usually highly configurable. Chameleon technology was developed as the Configurable Web Mapping Client Component (CWC2). In addition to navigation functionalities (like pan, zoom in/out, zoom to full extent and refresh the map), Chameleon allows the user to implement the queries easily. Nowadays this project is no more developed, the latest version is of 2007.

4.3 p.mapper and MapServer

The p.mapper framework can be a good solution to setup a MapServer application providing a full set of functionalities. Based on MapServer, its use requires some knowledge of the MapServer environment and in particular, users have to be familiar with mapfile creation and configuration. It is important to underline that this framework can support both WMS and WFS. This framework makes use of PHP, XML, JavaScript, HTML and CSS.

4.4 OpenLayers

OpenLayers is a map viewing library written in JavaScript library with no server-side dependencies, so it can be used both with MapServer and GeoServer. OpenLayers also implements a JavaScript API to incorporate maps from a variety of sources into a Web application. It can support WMS and WFS and other different data sources. OpenLayers is Free and Open Source and it is a project of the Open Source Geospatial Foundation.

4.5 Mapfish

An OpenLayers - based client can be enriched with the use of MapFish. MapFish is a flexible and complete framework for building rich Web-mapping applications. MapFish is based on Pylons Python Web framework, extending it with geospatial-specific functionality. It also provides a complete JavaScript toolbox, a JavaScript testing environment and tools for compressing JavaScript code. The JavaScript toolbox is composed of the ExtJS, OpenLayers, GeoExt JavaScript toolkits and specific components for interacting with MapFish Web services.

5. IMPLEMENTED SYSTEMS

Data flow of the implemented clients is shown in the diagram below.

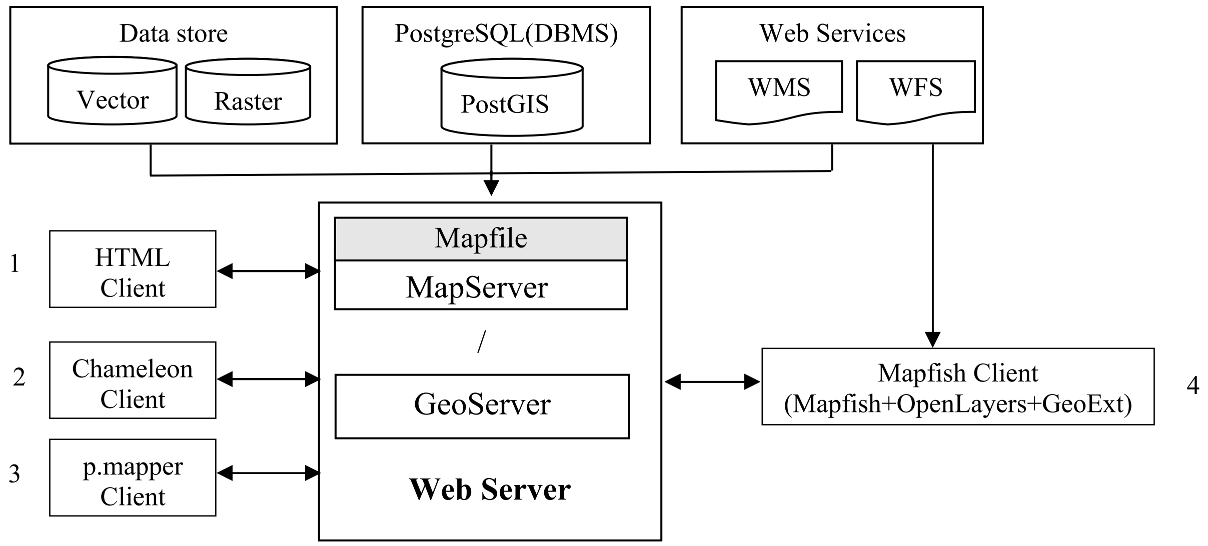

Figure 1: Data flow and system components including the four technologies used

In case of (1), (2) and (3) the mapfile is used to define data to be used, map rendering, query parameters, etc. In case (4) the creation of the mapfile is compulsory only if MapServer is used at the server-side, otherwise in case of GeoServer it is not necessary. In OpenLayers WMS and WFS can be directly imported in the map without passing through a geographic server and, using MapFish, users can also connect to PostGIS DBMS directly.

A comparison of the main features of the four technologies is implemented is shown in the following table.

Table 1: Web map client comparison

Technology	HTML (CGI Application)	Chameleon	p.mapper	Mapfish &OpenLayers
Server used	MapServer	MapServer	MapServer	MapServer/GeoServer
Source code language	PHP, JavaScript	PHP, JavaScript	PHP, JavaScript	Python, PHP, JavaScript, Java
Search and query	Mapfile and HTML configuration	Mapfile and HTML configuration	XML, mapfile and JavaScript configuration	Mapfish service (PostGIS) function search (MapServer)
Map views functionality	Need to refresh to update the map	Need to refresh to update the map	Auto - refresh	Auto - refresh
Mailing list	No	Yes	Yes	Yes
Double click	No	No	Yes	Yes
Zoom bar	No	No	Yes	Yes
Functionality	Zoom, pan, query	Zoom, pan, query	Full set of functionalities	Full set of functionalities
Service OGC	WMS, WFS	WMS, WFS	WMS, WFS	WMS, WFS, SOS
PostGIS	Yes (using MapServer)	Yes (using MapServer)	Yes (using MapServer)	Yes (using MapServer/Geoserver or Mapfish Server directly)
Last Version		Sept, 2007	Oct, 2010	Sept, 2010

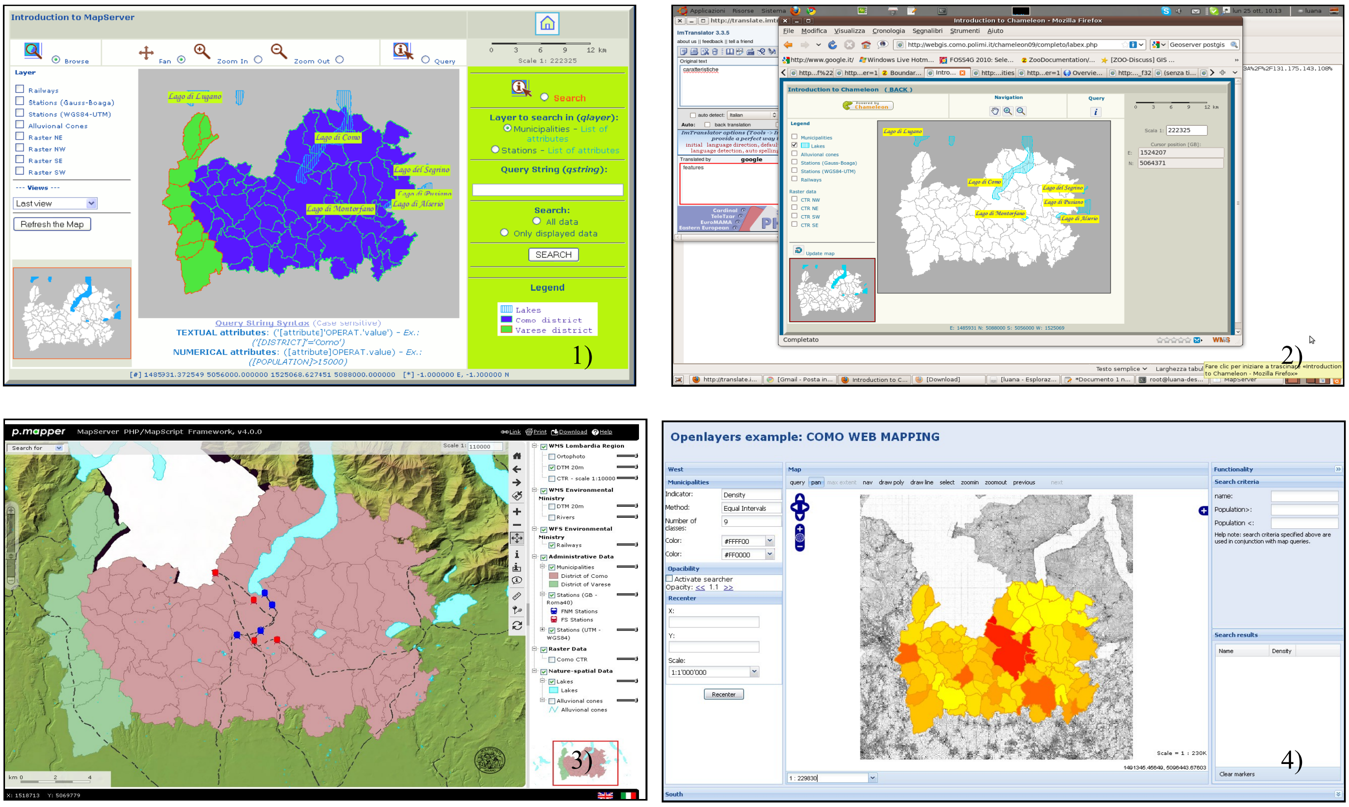

Figure 1: HTML with MapServer (1), Chameleon (2), p.mapper (3) and Openlayers with Mapfish (4) frameworks

As it can be seen from the table, the first two solutions are less interactive and they may be less intuitive for the end-users. In fact, p.mapper and Mapfish provide an auto-refresh of the map when the user applies some changes, for example turning on/off a layer, while with Chameleon or simply HTML the user have to press the 'Refresh' button. Regarding the available functionalities, p.mapper easily provides a huge set of functions simply enabling the plug-ins in the configuration file. On the other hand, Mapfish is less easy to implement but it is more suitable to develop ad-hoc functions, especially using the GeoExt libraries.

Those last two frameworks are the most interesting ones, even because they are still developed (while Chameleon last version has been done in 2007). From a didactic point of view, p.mapper may be a good solution in terms of functionalities/effort to build up the WebGIS, while for companies or professionals a solution based on OpenLayers may be the most appropriate to develop additional functionalities and to provide a less 'standard' configuration. For instance, it is possible to create different map windows inside the same HTML page, create different panels and forms for users data entry, etc.

6. CONCLUSIONS

We implemented and developed our WebGIS using different technologies but the same kind of data to have a direct comparison of the different frameworks. Some changes in the dataset have been done to show the possibility of using Web services, such as WMS or WFS. For a didactic purpose we have inserted in the same WebGIS data in different reference systems, to show the possibility of re-projection provided by the geographic servers. Our implementations have already been used for educational purpose, teaching students how to develop them step by step. Currently we are preparing all the material, both source code and

Web mapping products (http://webgis.como.polimi.it/webgis_demo.php?lang=en), to be published on line with the developed examples, so that they can be a good reference for education and training field.

For the future, we are going to focus on p.mapper and OpenLayers with or without Mapfish support, testing also the possibility of represent huge vector and raster data. Our interest will be focused also on the possibility of providing advanced geo-services and functionalities. Furthermore, nowadays it is possible to visualize not only 2D maps, but also 3D web viewers, such as Google Earth or Street View. A future improvement will be the possibility to visualize in one side of the screen 2D maps and on the other side a 3D GIS visualizer, in order to improve user's perception of the interested area. This solution will be developed using OpenLayers with the support of GeoExt libraries.

References

Carrillo G. (2010), *Web mapping client comparison,* Soluciones Geoiformasticas Libres, http://geotux.tuxfamily.org/index.php?option=com_myblog&show=comparaci%F3n-de-clientes-web-para-sig-v.4.html&Itemid=59&lang=en

Maria A. Brovelli, (2010), *Lecture notes on InternetGIS and GIS course*, Politecnico di Milano – Polo Regionale di Como.

Raghavan V., Phisan S., Shinji M., Tatsuya N. (2004), *Implementation of Web Map Server Test bed and Development of Training Material for Advancing FOSS4G Solutions,* Proceedings of the FOSS/GRASS Users Conference-Bangkok, Thailand.

Aime A. (2010), *WMS Benchmarking*, Presentation adapted and updated for FOSS4G Barcelona 2010, Spain.