The use of OpenLayers for the demonstration and web visualization of environmental applications

Martin Dresen, Divyang M. Trivedi

geo SYS, Nansenstr. 17, 12047 Berlin, Germany - martin.dresen@geosysnet.de

Abstract

How far can we go the easy way?

This is one of the main questions that have been asked at the beginning of a recently started project to support different environmental NGO activities and projects in Ethiopia. The project partners should gain access to geo-data and simple GI functions like digitizing or visualizing feature data via a webmapping solution. Unlike many other comparable projects the favored software or tool should be as simple as possible. According to this the requirements of the project are on condition that no software has to be installed and no complex server technology will be used within the project. The user should not need specific knowledge as regards the use of GI or webmapping functionalities and of course, the tool should be composed of open source tools and functions. Additionally, an internet connection should not be necessarily.

In order to comply with the strict requirements a multi-level structure and realization was favored. The following contribution will describe and examine the easiest part of that structure which includes the exact realization of the projects' requirements by using and enhancing OpenLayers. The use of OpenLayers for the demonstration and visualization of environmental data will be described by using examples from forestry applications and ecosystem management in Ethiopia.

Introduction

Environmental projects very often include the participation of different parties and scientific fields. In the case of development cooperation projects like the discussed one from Ethiopia, complex and international organizational structures often impede the management and processing of the project.

In addition to that, the collaboration of participated NGOs, organizations and institutions within such projects is constricted frequently because of the spatial separation between the different project locations as, for example the main office, project office or field office.

This was exactly the existing situation in Ethiopia, where the creation of a biosphere reserve within a remote and hard accessible area had caused organizational and technical problems. The local personnel were assigned to collect land use boundaries and land use changes with GPS data loggers. The generated track files and waypoints and a description of the tracks were subsequently sent to the main office for processing and preparation of the data. Incomplete descriptions, tracking errors, other erroneous data or misused data loggers are many sources of errors that have resulted in the necessity of new data entry. Without going into detail there is an apparent need for action. It was

figured out quickly that a webmapping solution might help in solving the structural problems. The local personnel might upload their collected data, adjust or process it if necessary. Nevertheless the development of such complex client-server architecture was not possible because of lacking funds. Therefore a simple, free and easy to use tool has been required to fulfill the main requirements and tasks. The development or implementation of a 'real' webmapping solution has been postponed to a later time.
OpenLayers was found to be a tool that supports many of the required functions for this reason it was selected for the first and easiest implementation scheme. Later on, the OpenLayers functions should be extended with server-based functionality. At this point the first and simple implementation will be presented primarily. The detailed requirements and the realization using OpenLayers will be discussed in the next sections.

Functional requirements for the project

Before discussing the possibilities of using OpenLayers, the main requirements for the project will be given, as defined for the first implementation level.

The main requirements are:

- Preliminary planning of GPS data collection with the aid of topographic maps, satellite images and/or different web services
- (Up)loading of GPS tracks and waypoints into the system, visualization and overlaying with maps and/or other geo-data for checking and updating purposes
- Digitizing and saving data as KML files (region boundaries, test areas, roads and paths, forest clearing areas)
- Visualizing land cover changes by overlaying time-based layers or services
- Forwarding or sending of project files and collected data must be possible. Everybody should be able to open and visualize a whole project without installing any kind of additional software.
- No installations of software or plug-ins are allowed.
- No expert knowledge (client-server, webmapping, programming or even HTML or JavaScript) should be required for using the tool.

These restrictions – especially not to install any kind of software – require the development of a pure client-based application by using standard HTML and JavaScript elements exclusively, an application, that should be executable in all web browsers.

Using and gradually adapting OpenLayers for environmental applications

OpenLayers, initially developed by MetaCarta, consists of a single JavaScript library that allows the client-based displaying of different standardized formats like WMS, WFS or GML, which can be integrated as layers without any server side dependencies. Furthermore, OpenLayers supports basic navigation features (zooming, zooming to full extent, panning), client-based tiling and the integration of markers for example.

Although OpenLayers offers many opportunities for visualizing environmental information and can be easily integrated in existing web pages it still requires basic knowledge in HTML and JavaScript. Furthermore, just a few of the required functions are implemented in the OpenLayers API.
Due to these limitations a gradually process of adapting OpenLayers was implemented, which is still in progress.
Initially a form-based HTML page was created to facilitate the use of OpenLayers by dynamically creating applications based on the selected tools and services. In this way even environmental professionals with no or little knowledge about webmapping or HTML and JavaScript can use the tool without getting in contact with any kind of code.

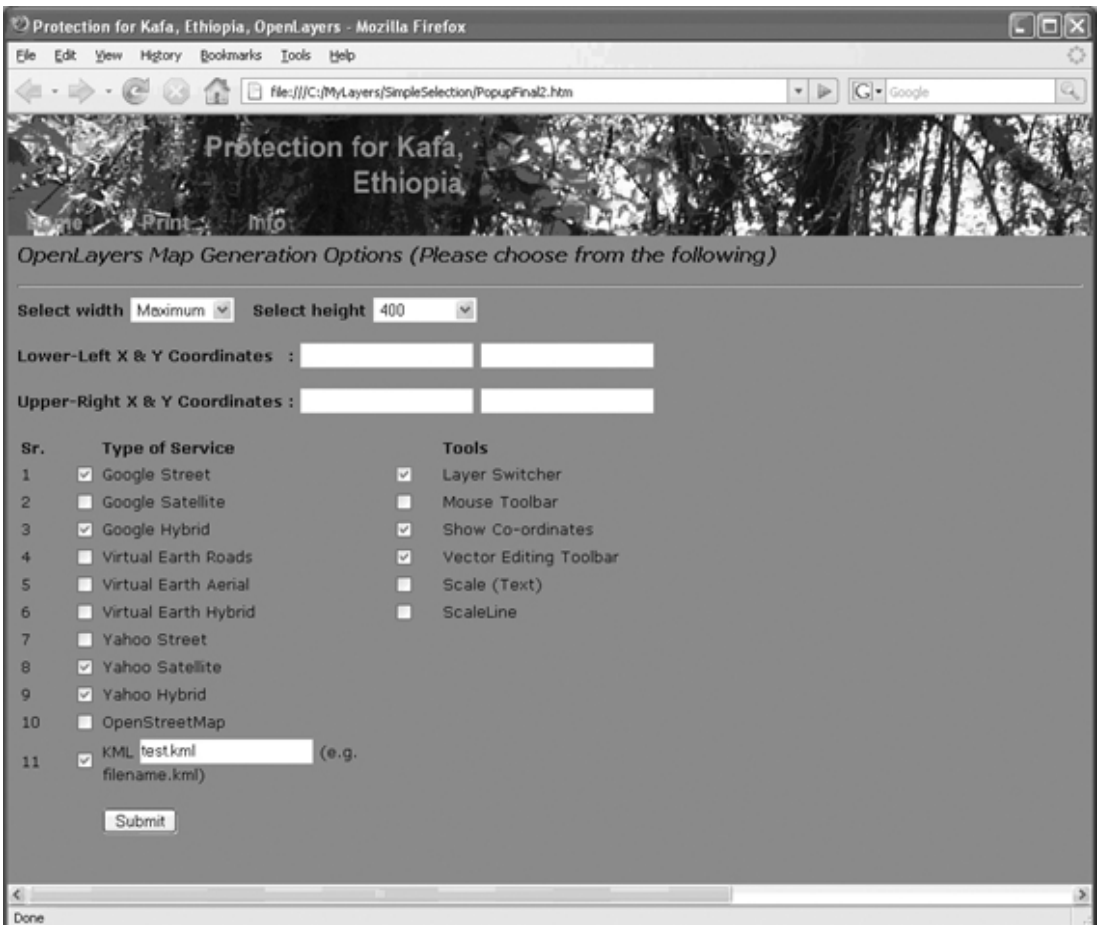

Figure 1. Screenshot of HTML form for the automatic creation of OpenLayers applications

A second step has included the evaluation of existing functions in the OpenLayers API and its adaptation and extensibility. It was quite obvious that a pure client-based application would be insufficient for the required functionalities without installing any other software or library.
Although the ‘lightweight’ OpenLayers API has the advantage of being incorporated easily into current web pages and can be edited with a simple text editor, there are still some programming skills required for the adaptation and further development. Without any server side technologies there is no possibility to use server controls or to integrate own geo-data except for XML-based data. Besides that there are some cross platform issues with JavaScript.

For this reason a multi-level development was aspired that might include server-based functionalities. That would also have the advantage of providing additional functions. Figure 2 gives a review about different standard GI tools and functions that might be implemented and used with OpenLayers. Some of the functions can only be implemented by combining OpenLayers with (server-based) functions from other webmapping solutions. In this case GeoServer and UMN MapServer functions have been used. The difficulty level shows the realization complexity of integrating the tools and functions into OpenLayers.

No.	Function	Difficulty level	OpenLayers 2.6	Geoserver 1.6.4	UMN Mapserver (MS4W 5.0.2)
Basic GIS functions					
1	Zoom in	•	default		x
2	Pan	•	default		x
3	Zoom to full extent	•	default		x
4	Zoom to last extent	••••	customized		x
5	Overview map	••	default		x
6	Attributes of layer	••••	customized		x
7	Scalebar	•	default		x
8	Zoom to extent	•••	customized		x
9	TOC / layer switcher	•	default		x
10	Show legend	•••	customized		x
11	Show coordinates	•	default		x
Extended GIS functions					
12	Query	••••		x	x
13	Refresh view	••			x
14	Zoom to scale	•••	customized		x
15	Layer opacity	••	customized		x
16	Change layer symbology (SLD)	•••		x	x
17	Loading status image	••••	customized		x
18	Time stamp	••••	customized		x
19	Calculate length / area	••••	customized		x
20	Add / remove markers (maptips, hyperlinks)	••••	customized		x
21	Bookmarks	••••	customized		x
22	Save / export map or layout	•••			x
23	Units converter for measure	••••	customized		x
Integration of geo-data					
24	Load vector data (gml, kml, xml)	••	default	x	x
25	Load vector data (shape)	•••		x	x
26	Load raster data (tif, png etc.)	•••		x	x
27	Load service (WMS, WFS)	•••	default	x	x
28	Data from DB	•••		x	x
29	Data to DB	••••			x
30	Load different coordinate systems	••	default	x	x
Digitizing / Editing features					
31	Create new features (graphics)	••	default		x
32	Edit WFS features	••••			x
33	Edit attributes	••••			x
34	Save features (file)	•••	customized	x	x
35	Save features (DB)	••••			x

Score	Description
•	Can be Integrated easily
••	Requires basic HTML / Javascript knowledge
•••	Requires extended HTML / Javascript knowledge
••••	OL API for creating user-defined functions and controls etc.

Figure 2. Integrating functions and tools into OpenLayers and extended capabilities

The so far existing or realized functions with OpenLayers are as follows:

- Load default web service (Google Maps, OpenStreetMap, etc.)
- Layer Switcher (turning on / off different layers
- Navigation tools (zoom in, zoom to extent, zoom to last extent, zoom to full extent, pan)
- Measure feature with included units converter (line, area)
- Load KML files
- Create vector graphics
- Save and load a region of interest (ROI), bookmark
- Change opacity of layers, transparency
- Show attributes of a layer / web map service
- Show coordinates and a scale bar
- Animation of time series / change detection (here: forest change)

The so far realized functions with OpenLayers and the integration of GeoServer and/or UMN MapServer are as follows:

- Edit Shapefiles
- Query features
- Load data from PostgreSQL/PostGIS database
- Load own geo-data as WMS or WFS services from GeoServer
- Print data / map
- Show legend

Many of the required functions could already be implemented. A very important task could be realized by giving the local personnel the possibilities to plan their data collection by GPS in a much better way than before. Test or collection areas may be planned and digitized by the adjustment of existing data or loaded web services. The digitized data can be saved and passed to other project members. Afterwards, generated track files or waypoints can be visualized and corrected or checked directly if necessary, before they will be sent to the main office. This process simplifies and improves the whole project management and the quality of the processed data. Whether loading of own geo-data (except for KML files) was not necessary, a simple OpenLayers HTML file with embedded JavaScript functions is sufficient for most required functions. If the application is compiled with the mentioned form, it will be created automatically with one click in a stunning way. Even local personnel without any knowledge about webmapping are able to use the tool. The hence resulting possibilities of forwarding all contained files easily should also be mentioned here. Altogether the process flows of the project could be advanced considerably. This was really astonishing as no trainings had to be given to the local personnel. The tool is so easy that everybody could understand its use and possibilities.

```
function createExtList(){

    var mylist = document.getElementById("extList");
    cntEntry = chkCookie();
    var removeEntryNo;
    for(k = 0; k < cntEntry; k++){
        mylist.options[k]=new Option("Extent" + (k + 1) );
        removeEntryNo = k;
    }
    mylist.remove(removeEntryNo);

    return false;

}

function chkCookie(){
 var idl= "Extent";

 for (j = 1; j <= 100; j++){
    var newid = idl + j;
    var returnval = getCookie(newid)

    if (returnval != ""){
    }
    else{
        return j;
    }

 }

}
```

Figure 3. Code example of the implemented “save extent” function within OpenLayers

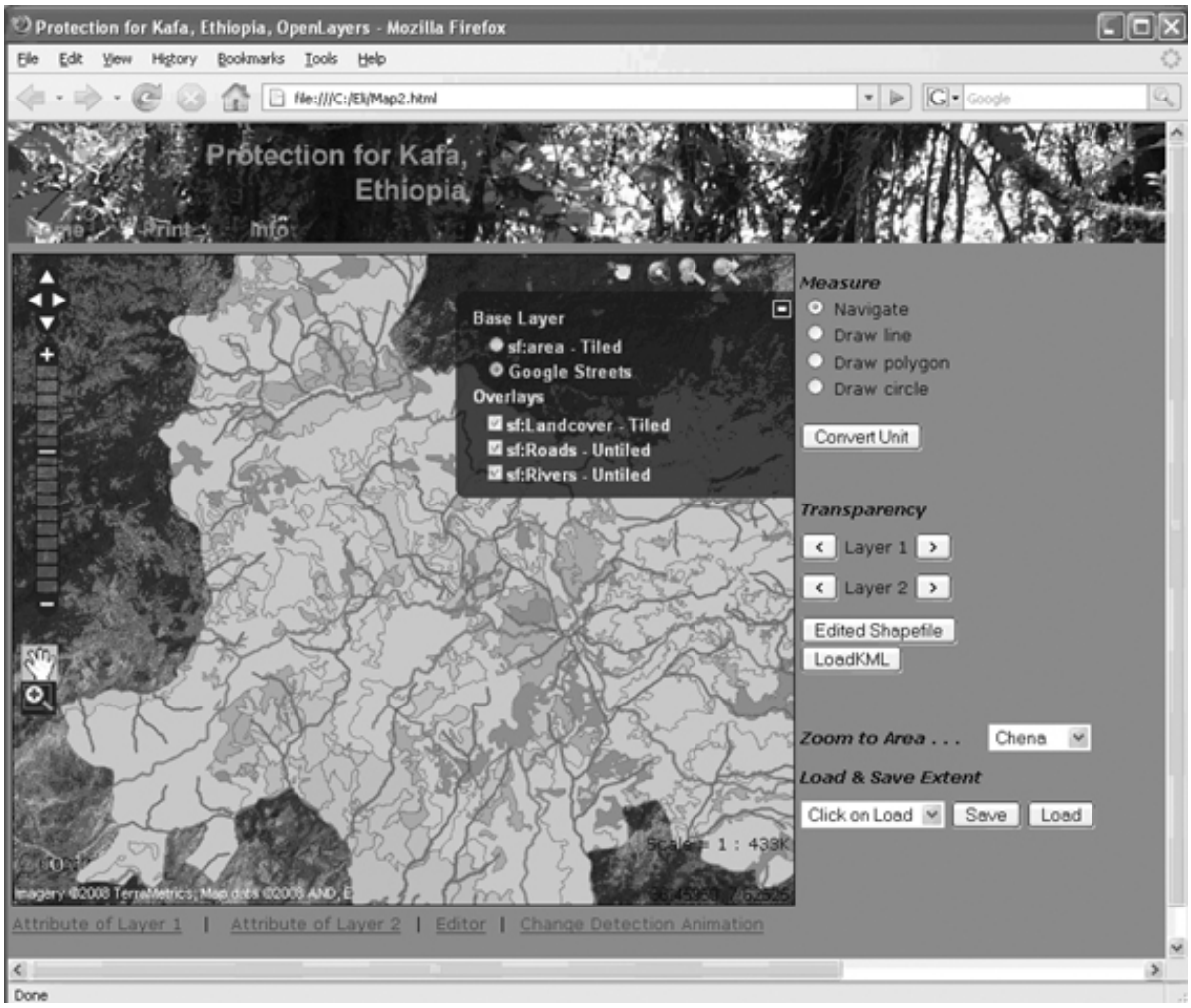

Figure 4. Screenshot of the demo application

Outlook

Although the already realized tasks enable an enhanced use and processing of geo-data there are manifold possibilities for improvements. The available data will be centralized to a geo-data server, which includes the installation of a map server technology for the provision of web services. This will continually improve the geo-data workflows even though the simplicity of the OpenLayers will partially get lost.

Nevertheless, a fully integration and combination of OpenLayers and server-based technologies will be aspired. Besides that, the construction of a GIS training center is planned for the project area (especially GPS training, Webmapping training). This clearly points out that the project is on the right track.

With regard to future developments the initially question should be changed. It is not the thrilling easy way anymore, but the project is still under way.

References

OpenLayers Class Diagram, viewed 12 August 2008, <http://www.smartmapbrowsing.org /html/images/uml/ClassDiagram_OL2.4RC5.pdf>

OpenLayers Documentation, viewed 12 August 2008, <http://trac.openlayers.org/wiki/ Documentation>

OpenLayers Homepage, viewed 12 August 2008, <http://www.openlayers.org>

OpenLayers JavaScript Mapping Library, viewed 12 August 2008, <http://dev.openlayers.org /releases/OpenLayers-2.6/doc/apidocs/files/OpenLayers-js.html>

OpenLayers User Guide, viewed 12 August 2008, <http://trac.openlayers.org/wiki/UserGuide>

Ramsey, Paul. 2007, 'The State of Open Source GIS', viewed 12 August 2008, <http://www.refractions.net/expertise/whitepapers/opensourcesurvey/survey-open-source-2007-12.pdf>

Schuetze, Emanuel. 2007, 'Stand der Technik und Potenziale von Smart Map Browsing im Webbrowser am Beispiel der Freien WebMapping-Anwendung OpenLayers', (Master thesis), University of Bremen.