REVIEW OF CONSOLIDATED WEB MAPPING TECNOLOGIES FOR THE MANAGEMENT AND VISUALIZATION OF MULTIPLE GEOSPATIAL DATA LAYERS

Alberdi I., Laka M., Olaizola I.

Vicomtech-IK4, Mikeletegi Pasealekua 57, 20009 Donostia, Spain
{ialberdi; mlaka; iolaizola}@vicomtech.org

Abstract
In recent years, a significant advance in technology for the development of web-based applications and services is taking place. As a consequence, most desktop applications are evolving towards service based solutions that can be accessed remotely. The traditional GIS desktop applications are also evolving into web-based services partially thanks to the capabilities that web mapping technologies offer. This paper reviews the up-to-date web mapping technologies and research providing features, performance, data-format compatibility and interoperability capabilities of them. Among all the existing web-mapping technologies, the proprietary Google Maps and arcGis Web Mapping as well as the open source solutions OpenLayers and Leaflet must be highlighted. In order to feature testing of highlighted open source technologies, a use case scenario for the visualization of weather prediction layers was implemented and tested.

Keywords: *Web mapping, GIS, Meteorology*

INTRODUCTION

Since 1995 began operating the global positioning system (GPS) [1] with full operational capability, representation of navigation maps changed radically. This allowed many companies to develop new business models around the representation of maps using different computer systems. Moreover, the large development that is taking the world of internet and the ability to stay connected from any device at any time has enabled web systems to take the center stage among users. Web technologies that more evolved and achieved an outstanding success are HTML5, AJAX, JavaScript and CSS3 web languages. These type of technologies allow the users to access and manage the content they want from anywhere on any device. This advance gave new opportunities to develop more accessible, faster, multi-device systems based on displaying geographically referenced information through map technologies. More precisely, in the field of Meteorology web applications for representing both weather forecasts and real time weather information increased notably, since web mapping systems were developed. Recently, the development of GoogleEarth/Maps owner system and open source Leaflet and OpenLayers have given a new boost to the applications based on web-mapping technologies.

The OGC [1] promotes the use of standards for Web mapping services, which have helped to establish a common framework to access, display and download spatial data on the Internet (Web Map Service, Web Feature Service, Web Coverage Service), discover them (Catalog Service for the Web), present them by means of styles (Style Layer Descriptor), filter them (Filter encoding), store them, transport them (Geography Markup Language and Keyhole Markup Language) and process them (Web Processing Service). The Web Map Service (WMS) and Web Feature Service (WFS) are the mainly used services for Web mapping purposes; therefore, support for one of them is a must for the web-mapping clients selected for the present study.

One of the main objectives of this paper is to give some guidelines about various relevant existing network technologies [2] [3] [9] [10] for displaying weather information. The weather data will be provided through images generated by WMS service that can be in PNG, GIF or JPEG format. WMS produced maps can be invoked by any corporate software platform or a trained GIS viewer for displaying this type of services. The WMS standard defines three operations: GetCapabilities, GetMap and GetFeatureInfo. But in this work we centered in the operation GetMap because it returns a map image with geographic and dimensional parameters are defined.

The work presented in this paper is structured as follows. Firstly proprietary and open source web-mapping platforms that have the most widespread use among internet users are studied. After this exhaustive state of the art, the scenario where the features and performance of the highlighted open source technologies is explained at section three. At section four, the results obtained at the analysis scenario are exposed, and finally, the conclusions drawn are presented in section five.

FEATURE COMPARISON OF WEB MAPPING TECHNOLOGIES

In this review of web-mapping technologies, the study has focused on the capabilities of web-mapping technologies to add visual representation of geographically distributed data. The studied technologies have been grouped according to their ownership, dividing into proprietary and open source technologies.

Proprietary Systems

Most proprietary web mapping platforms that exist have very similar functionalities in their public versions. One of the reasons is that the interest of the users is centered on finding places and routes. All platforms offer the option to display the maps as a cartographic layer or with satellite images or as a combination of them. This kind of options provides to the users a more immersive and guidance experience in place searching.

A noteworthy evolution of these systems is the streetview systems that place the users in the requested streets giving a 360 degree view of the localization. This feature permits the immersion of the users in the interested area. Some of the mapping systems also provide layers reporting traffic conditions. It is still not spread to the whole world, but looks a promising service for users that is offered in combination with other technologies such as GPSs localization system.

What really set these services among them are information searches. Most proprietary services have more users belong to the major search engines. It allows a transversality between search engines and maps to provide to the user a more complete and useful information, which ultimately means a better use experience and a greater number of users.

Google Maps

The first version of Google Maps [4] [11] [12] appeared in February of 2005 and it was developed by Google. Nowadays the API is in the 3rd version and it is known as Google Maps JavaScript API V3. The Google Maps APIs let a web page developer embed freely Google Maps on web pages or mobile applications. It is the most popular proprietary online mapping service and it is used in all over the world in more than 800,000 sites. Google also provides various APIs that allow the users to query routes, estimate the duration and distances, elevations, locations, geocoding images or watching StreetView images. Using Google Developers Service, the developers agree to be bound by Google Terms of Services [13]. All users that want to include Google Maps in their websites have to load the Google Maps API using an API key. If the usage of Google Maps API exceeds the usage limit, the user will need to pay an additional quota. It permits a maximum of 2500 request per 24 hours period in its free version.

Google Maps API permits the developers to add elements like markers, icons, polylines, polygons and layers. It allows the inclusion of different kind of layers such as Dynamic Maps Engine Layers, Tile layers or KmlLayer. KmlLayer represents KML and GeoRSS elements, while TileLayers represents WMS. Google Maps API also permits loading information stored in geoJSON files. In addition to this, Google itself also provide information layers such as the traffic, weather, cycling paths, public transports and panoramio images that can be load in the maps.

Nokia HERE Maps

HERE maps platform [14] was launched in 2007 by Nokia under the name of Ovi Maps. It includes an API for JavaScript which includes a set of interfaces that permit web applications developers build rich and interactive maps. The API is divided in 4 main components Maps, places, directions and traffic. HERE maps also provide an enterprise API for JavaScript that lets the user add business-to-business(B2B) specific features and functionalities of a collection of RESTful web services like points of interest, searches, geocoding or routing. API also includes W3C positioning that takes advantages of the W3C geolocation API which is supported by many browsers. Nokia defines a Terms Of Use [15] were there are available three licensing option for the developers interested in the use of HERE Maps APIs: base, which permits a free monthly and basic access, evaluation and commercial. HERE API users need two random strings, id and app_code, to have secure access to HERE API services, in the case of evaluation licenses it expire in 90 days.

In the HERE Maps API for JavaScript developer beyond include marker, image, custom overlays, grid overlays or geo shapes based on coordinates to the map like polygons, polylines, circles or rectangles it also can add custom overlays from tile servers like WMSs. Also it permits the integration of KML files and data sets compliant with versions 2.1 and 2.2 of the KML standard in web applications.

Bing Maps

Developed in June of 2009 by Microsoft, Bing Maps platform [16] provides two APIs to develop Bing Maps applications. If the aim of the developer is to create an application for windows 8 it has defined the .Net API. In addition, it also provides an API for Silverlight/WPF platform that permits the location and local search features into their applications. The Bing Maps platform also provides a JavaScript API that is based on the Bing Maps AJAX V7 control which is supported by standard browsers and mobile devices. Bing Map also implements REST and Spatial Data Services what permits the user to include geocode and reverse-geocode location data, elevation information for a set of locations, polylines or areas on the Earth. It also permits the inclusion of map tile URLs and imagery provider information that is hosted by Bing™ Maps, directions and route information for driving, walking or using transit and information about traffic incidents and issues in a specified area. The use of the APIs is governed by the Microsoft® Bing™ Maps Platform APIs' Terms Of Use [17]. The user needs to create a developer account on the Bing Maps Account Center. It allows the users to create a Bing Maps Key to use in map applications.

For including elements in the maps, Bing Maps API defines Entities. Entities are classes that define element types like Infobox, Polygon, Polyline, Pushpin, TileLayer, or EntityCollection. For the integration of TileLayers in the maps, Bing Maps API permits the definition of TileSources in an easy way and furthermore it permits the inclusion of WMS layers using just a line of code.

arcGis Web Mapping

ArcGis Web Mapping [18] was developed in May of 2000 by ESRI. It was known as ArcIMS 3.0 and was ESRI's first public release. Among the analyzed systems it is the most focused on mapping and also is the system that provides most mapping visualizations capabilities. In order to include applications in Flex and Silverlight/WPF platforms ArGis Web Mapping has defined well documented APIs that can be found in their web page. In addition, it also offers an API to include maps in web environments developed in JavaScript that leverages latest HTML5 and CSS3 standards, increasing the performance and functionality of the maps and permitting their visualization in any kind of devices. ArcGis defines a Terms Of Use [19] that explains the permission and prohibitions on the use of ArcGIS Online.

ArcGis Web Mapping has different classes defined for the inclusion of layers. Some of most employed ones are ArcGISDynamicMapServiceLayer, ArcGISTiledMapServiceLayer and ArcGISImageServiceLayer classes. Those classes permit users to work with a dynamic map service resource, with a cached map service resource and with an image map service resource respectively. The cached service accesses tiles from the memory instead of rendering images dynamically on the fly. For base layers there is a class called OpenStreetMapLayer which allows users to use basemaps from OpenStreetMap easily. Regarding the display of layers, ArGis API provides GeoRSSLayer, KMLLayer, streamLayer, MapImageryLayer and WebTileLayer classes that are used for visualization of GeoRSS, KML, stream of data using HTML5 websockets, georeferencied images and non-ArcGis server map tiles. With the purpose of displaying layers based in OGC standards the API also provides classes like WMSLayer and WMTSLayer that permits the visualization of Web Map Services layers and Web Map Tile Service layer.

Open Source Systems

The open source systems have a great success among developers of web mapping services. Being open systems, not only does allow users to create plugins for developed systems, but also to adapt the code to the needs and projects of each developer. For the development of the different platforms, users create communities themselves and develop applications in a collaborative manner. Thank to those communities users help each other with the questions and concerns that users and contributors can have during the adaptation of the code to their own needs. It gives an added value to the documentation and service that many of other proprietary systems cannot offer.

OpenLayers

OpenLayers [5] [6] [20] was developed in June of 2005 by MetaCarta, but since 2007 till now it is an open source project of Geospatial Foundation (OSGeo) [21]. It is a library based only in JavaScript that permits users to visualize maps without any type of dependencies by the side of the server. OpenLayers API is developed for and by the Open Source software community and its last stable release is 2.13.1, but actually it is working on development of its 3.0 version focusing on performance improvements, WebGL features and integration of the new Cesium library. OpenLayers has extensive documentation with which users can learn easily and fast to integrate maps into websites. The OpenLayers code is published under 2-clause BSD license.

Being one of the oldest open source mapping system and belonging to such important foundation as OSGeo makes OpenLayers the JavaScript library able to display a huge variety of data types. It supports GeoRSS, Keyhole Markup Language (KML), Geography Markup Language (GML), GeoJSON and map data from any source using OGC-standards as Web Map Service (WMS) or Web Feature Service (WFS).

Leaflet

Leaflet [7] [22] was developed in May of 2011 by Vladimir Agafonkin accompanied by a team of dedicated contributors. It is a modern open-source JavaScript library for mobile-friendly interactive maps that is in his version number 0.7. Leaflet was designed thinking in simplicity, performance and usability but including all the features that web mapping user can need. It is designed so efficiently that the weight of JavasScript code is about 33KB. It has no large documentation, but the source code implemented is so simple and easy understanding that does not need more for a good comprehension. Leaflet API was designed for being especially efficient in mobile platforms to achieve this it takes advantage of HTML5 and CSS3 languages. Third-party patches are absolutely essentials for extending the functionality and are included by different plugins developed by supporters of the community. The leaflet code is published under the very permissive 2-clause BSD License.

Leaflet supports Web Map Service (WMS) layers, GeoJSON layers, Vector layers and Tile layers natively. Many other types of layers are supported via plugins. Plugins also permit adding geo web services, such as ArcGIS Server, Arc2Earth, GeoIQ, CartoDB and GIS Cloud; free tile providers like OSM, OpenCycleMap, MapQuest, Stamen and ESRI; and formats such as GPX, KML and CSV. Moreover, in combination with svg tag-s allow the users to create an interactive application.

Cesium

Cesium [23] was developed in 2012 by Analytical Graphics, Inc. (AGI) and supported by an active open source community. This development was created because of AGIs customers needs for a cross-platform virtual globe for dynamic-data visualization. Cesium is a JavaScript library for creating and viewing 3D globes and 2D maps based in WebGL in a web browser. It supports a 3D globe, 2D map, and 2.5D Columbus View with the same API and can be added functionalities by the different plugins provided by the contributors of the community. Cesium is open source code under the Apache 2.0 license, which means, it is free for commercial and non-commercial use.

Cesium supports imagery layers using TMS, WMS, Bing, OpenStreetMaps, ESRI standards and standard images in any format supported by web browsers. Also shows vector data from CZML, GeoJSON, TopoJSON, KML, ESRI Shapefiles, and WebGL Globe JSON. It draws elements like polylines, polygons, circles, icons, labels, and custom objects. Cesium uses a material system to change their appearance to adapt to the user needs. Apart from the mentioned technologies Cesium core can be extended, can add functionalities and avoid large third-party dependencies for non-essential features by plugins.

UMN MapServer

MapServer [24] was developed in the mid-1990's at the University of Minnesota, but now is managed and administered by OSGeo and is maintained and improved by developers around the world. The aim of this platform is to visualize, consult and analyze geographical information through the network using Internet Map Server (IMS) technology. Beyond browsing GIS data, MapServer allows to create "geographic image maps". This open source geographic data rendering engine is written in C and actually is in its version number 6.4.1. The possibility of using as a map server for third party applications, using MapScript API or following OGC specifications, has permitted the publication of geospacial data. MapScript is a powerful scripting environment that support popular scripting and development environments like PHP, Python, Perl, Ruby, Java, and .NET. MapServer is published under MIT license.

MapServer platform supports numerous OGC standards such as WMS for client and server, non-transactional WFS for client and server, WMC, WCS, Filter Encoding, SLD, GML, SOS, and OM. It also allows a multitude of raster and vector data formats like TIFF or GeoTIFF, EPPL7, and many others via GDAL, ESRI shapfiles, PostGIS, ESRI ArcSDE, Oracle Spatial, MySQL and many others via OGR. In combination with Proj.4 library, MapServer provide to the users the possibility of making on-the-fly map projections.

ANALYSIS SCENARIO

An advanced web-mapping use-case scenario was defined and implemented in order to develop a real testing of the studied features. The use case scenario was defined for providing weather forecasts as layers above a web mapping service. The parameters of weather forecasts are stored in NetCDF files and therefore it was necessary a server like ncWMS that can interpret this kind of files and can answer tilelayer requests of weather parameters. The designed architecture also included MapCache server. It is a library part of MapServer which implements tile caching to speed up access to WMS layers from ncWMS server and deliver to the client side. The Figure 1 shows, the architecture of the implemented use case that consist on a two-sided architecture. In the server side there are MapCache and ncWMS servers. In the client side there is a client developed in HTML5, JavaScript and CSS3.

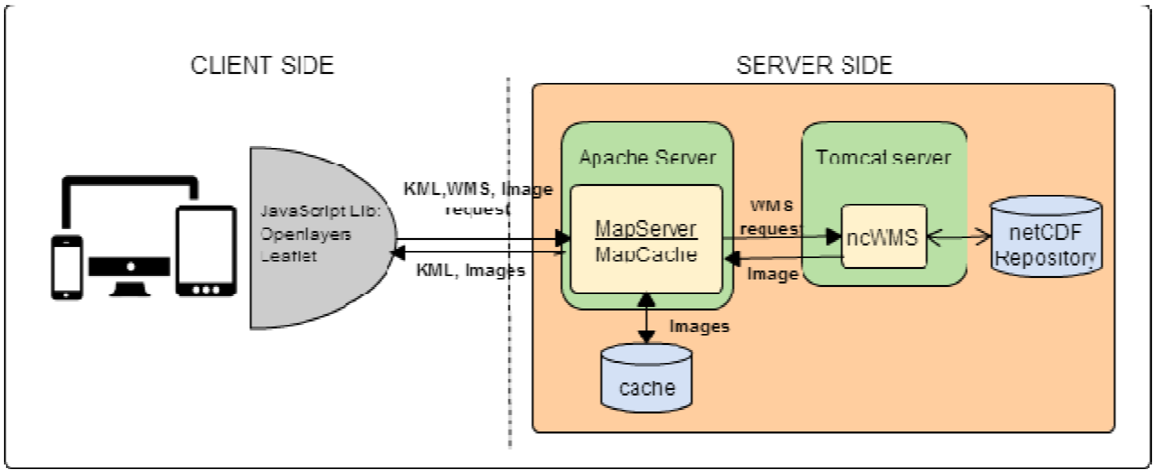

Figure 1. The architecture of the use case scenario.

At client side, the implemented use case application offers the same features for both implemented technologies OpenLayers and Leaflet. Thus, the user needs to select one of them but the system will work exactly the same way. After selecting the web-mapping technology, the user selects the days and the weather forecast parameters that is interested in visualizing. The platform offers two maps in where the parameters can be displayed in order to help in the comparison of the multiple forecasted parameters.

The system provides the possibility to request three different types of visualizations for each data. The first visualization permits to display a combination of temperature, pressure and wind layers. User also can visualize a blending of 7 layers such as pressure, temperature, wind, snow, rain, wind chill and humidity. Finally, the client application provides to the user the possibility to only display the interested parameters depending on the values limits selected by the user.

PERFORMANCE COMPARISON

The analysis scenario was used in order to compare the performance of the widely used open source platform OpenLayers and Leaflet. For OpenLayers the released version 2.13.1 of the API was selected and for Leaflet the release version 0.7.

The hardware of the client where the performance was measured consisted on a processor Intel® Core™2 Quad CPU Q8300 @ 2.50GHz × 4 of 64-bit with operating system Debian 4.7.3-7. The computer memory was 3.9 GiB and it included a GeForce 9400 GT/PCIe/SSE2 graphic card. On the other side, the hardware used at server side was a Intel® Core™2 Quad CPU Q6600@ 2.4GHz x 4 of 32-bit with operating system Ubuntu 12.04 LTS and a memory of 3.9GiB.

Description of the Testing

The main objective of the performance testing was to analyze the management of scientific layers by the selected web mapping technologies. With this aim, two stress tests were designed. The first one consisted on studying the performance when loading the same layer repeatedly. The second one consisted on loading seven different layers. With the second testing the influence of the typology of the layer was also studied. The results shown in next section are the average time of 10 repetition of each test.

Same Layer Loading

To test the preload of a layer, the same layer was loaded repeatedly. Thus it permitted us to evaluate the effect that an initial load of layers and subsequent request of same layers has regarding time. That is the reason for making interactions of 1, 2 , 3, 4 , 5, 6, 7 , 8, 9 , 10, 20 , 30, 40 , 50 , 60, 70 , 80, 90 , 100, 200 , 300 , 400 and 500 layers.

Through this experiment has also tried to compare the maximum load bearing layers each of the tools that have been used.

Different Layers Loading

In order to check if the time required for each platform in loading the same layer is equal to loading a different layer, we performed a very similar test to the previous. The difference is that in this case we were limited by the number of layers we had. In the architecture we presented in the section before, there were only defined 7 different types of weather layers so we had to adapt the experiment to the number of layers available. Therefore we made iterations with 1, 2, 3, 4, 5, 6 and 7 layers and no more.

Results

This section shows the results obtained in the tests for comparing the performance of OpenLayers and Leaflet. As we divided the experiment into two phases, firstly the results are presented separately and afterwards the overall results obtained are presented.

Same Layer Loading Results

As can be seen in Figure 2, in terms of total loading times of layers, Leaflet showed to be notably faster. The tendency shows that the total loading time of OpenLayers increases in comparison with total loading time of Leaflet. It means that Leaflet performance is better than OpenLayers performance loading the same layer repeatedly.

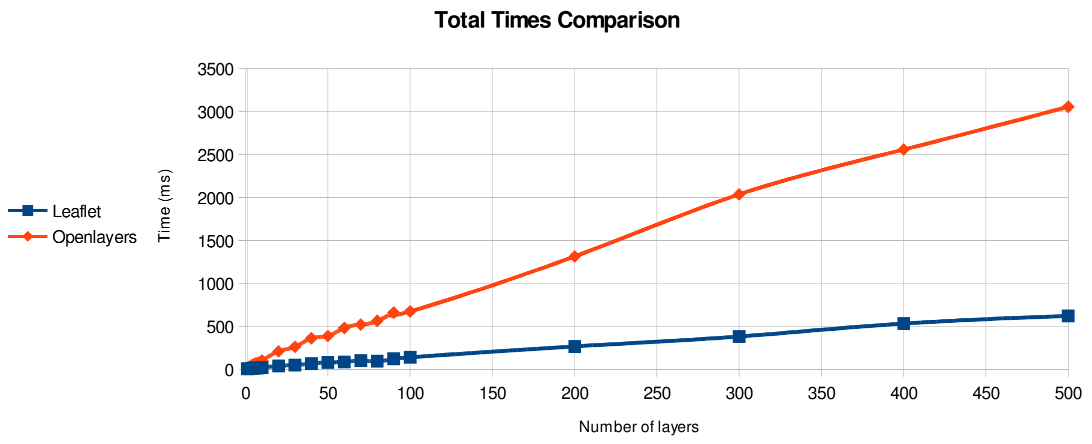

Figure 2. Average time comparison loading same layer repeatedly.

In Figure 3 the loading performance of the technologies can be analyzed deeply. It shows the time needed to load the different layers of both libraries. The graph represents the times that each technology needs to load the first layer, the times that needs for the remaining layers and the average of time that needs each technology to load a layer. When loading a layer repeated times, the maximum loading happens in both libraries when the first layer is loaded. It happens because when the user asks the system for the file it requests once for the information and in the next requests the system has all layers cached in. After loading the first layer, loading times decrease progressively from 17.52ms to 6.08ms in OpenLayers, and from 5.96ms to 1.23ms in Leaflet. It means that if a system preload all layers at the beginning of the application and then requests are reused, it would save considerable time.

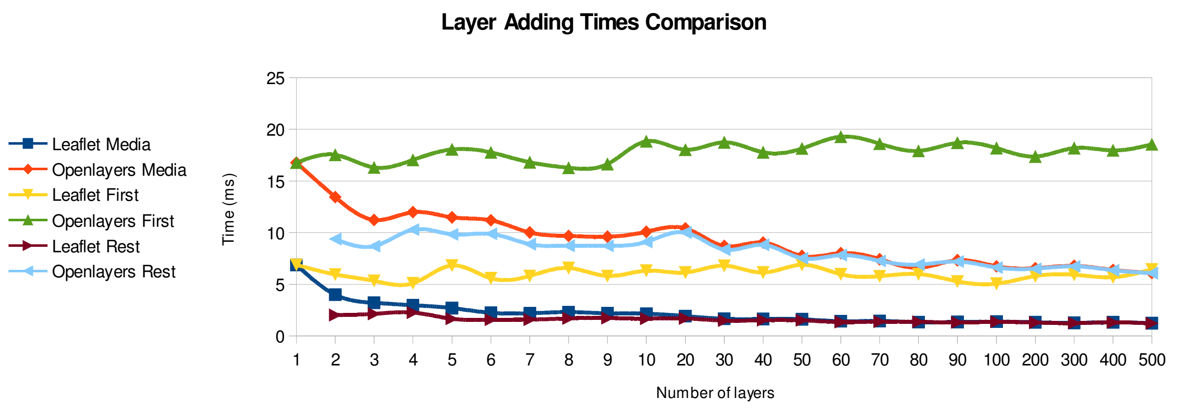

Figure 3. Average time comparison loading same layer repeatedly.

Together with the study of the performance of loading a layer multiple times, the performance limit in regard of layer cuantity was also studied. For OpenLayers execution started to slow down notably and visual elements of the web application giving problems when 300 layers were load. The same performance was obtained with Leaflet after loading 400 layers.

Different Layers Loading Results

As in the previous case, the time required for Leaflet is shorter than for OpenLayers when loading independent layers and it increases when the number of loaded layers increases. Loading a new different layer it takes an average time of 17ms for OpenLayers and 5.5ms for Leaflet.

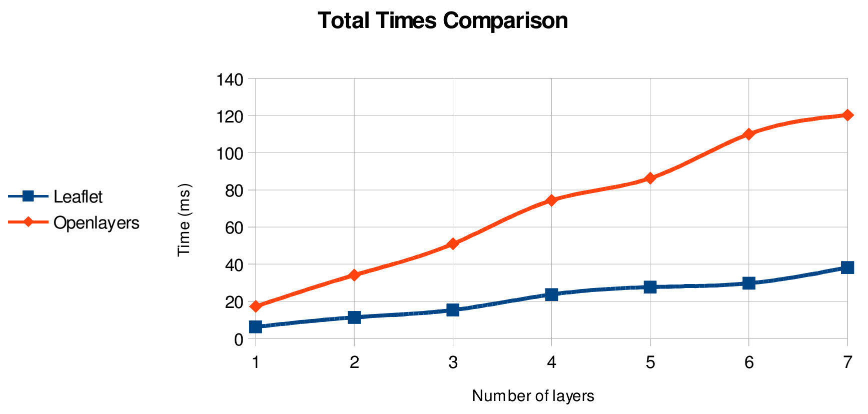

Figure 4. Average times comparison loading different layers.

This test highlighted that both technologies require a stable time for loading new layers regardless the typology of the layer. It also can notice that the performance in both the first layer and the average of the other layers in Leaflet is noticeably better than in OpenLayers. The trends observed in Figure 5 support the conclusion that there are no notable differences between the charging time of the first layer and the other layers, since the values in all cases are very similar. So if a developer has to work with different layers without the repetition of those layers, the loading will be balanced and similar to the data obtained for all of them in both technologies.

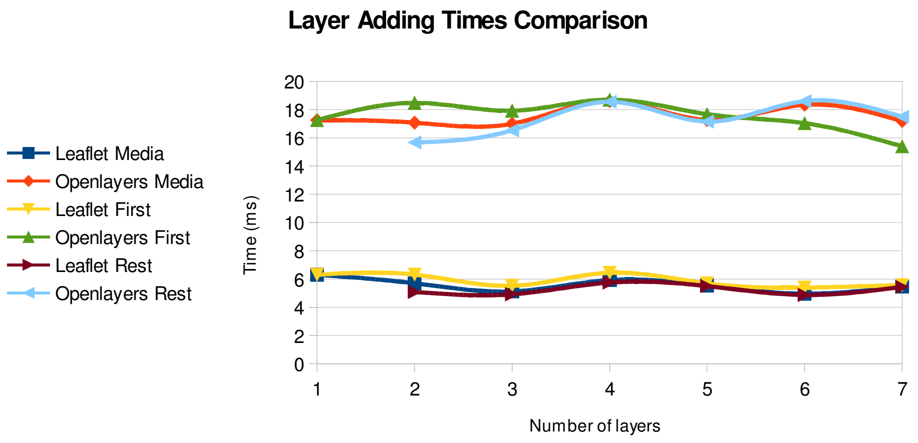

Figure 5. Average times comparison loading different layers.

Overall Results

Analyzing the data obtained in the cases of loading different layers and in the case of loading the same layers repeatedly, it reflects higher performance for Leaflet library.

Regarding the total time obtained, it can be clearly seen that the performance of Leaflet is better than OpenLayers performance. This is because Leaflet was designed to run on web and mobile devices and the resources available on mobile devices are more limited than in personal computers. In the insertion of different layers can be seen that the trend in the first 7 layers is the same as in the insertion of the same layer repeatedly, but to a lesser degree. It happens because in Leaflet the time in the case of the same layer is considerably reduced from the request of the first layer to the second layer, while in the second case the charging time is the same for all layers independently to the number of the layers.

In the total time analysis, from the beginning it can be seen that the execution time when the platforms add different layers or add the same several times the performance is better in Leaflet than in OpenLayers. In OpenLayers it does not matter if a layer already exists or a new one is added because the time taken is the same in each case.

If user desires preloading all the layers at the start the performance of Leaflet is also better. The average time it takes to load the first layer on Leaflet is less than what it takes with OpenLayers. Besides if the already loaded layer is reloaded, time is greatly reduced, even a quarter of what it takes to load the first.

CONCLUSIONS

Geolocation as weather or many other areas need web mapping technologies. For this work there are plenty of open source and proprietary technologies available. Each user has the freedom to choose the technologies that best suits them depending on the needs and available technologies it admit. Through this study we have been able to see the most relevant technologies that exist today and that without great programming knowledge anyone can enter and display on his website through different layers with different types of technologies relevant information.

In this work, actual web-mapping technologies were studied. There is a tendency for geolocating any type of information through web-mapping technologies via internet that is consolidating. In that sense, it was of high importance to study the performance of the available technology that would help to select the appropriate technology depending on the use. Although both proprietary and open technologies were studied, performance testing was applied to most common open source technologies such as OpenLayers and Leaflet.

All of the technologies analyzed integrate greater or lesser extent the standards defined by OGC. It supposes an advantage and permits the migration from one technology to the other without a major complication. It also provides universality to the data published by the network.

Specifically in the technologies tested to see its behavior in an environment of meteorological information management, it has been seen that both technologies loading times are less than 18ms in all cases. It shows that with this kind of technologies can be develop agile applications with real time answers to users. However, note that Leaflet presents loading times much better than OpenLayers reaching to 7ms as maximum. It makes Leaflet technology much more suitable for mobile environments where processing capability of devices is limited.

AKNOWLEDGMENTS

The study presented in this work is a result of the collaboration between Vicomtech-IK4, Dominion, Universidad de Leon and Near Technologies supported by the Ministry of Industry, Energy and Tourism of Spain (IPT-310000-2010-022). The authors greatly acknowledge all the members of the project.

REFERENCES

[1] Nelson, Robert A. 1999. The Global Positioning System. Applied Technology Insitute. Retrieved May 28, 2007.

[2] Carrillo, G. Web mapping client comparison v.6. http://geotux.tuxfamily.org/index.php/en/geo-blogs/item/291-comparacion-clientes-web-v6, 2013.

[3] RAMSEY, Paul. *The State of Open Source GIS,* http://www.refractions.net/expertise/whitepapers/opensourcesurvey/survey-open-source-2007-12.pdf, 2007.

[4] Svennerberg, G. Beginning Google Maps API 3. 2010.

[5] Santiago Perez, A. OpenLayers Cookbook. Packt Publishing, 2012.

[6] Hazzard, E. Openlayers 2.10 Beginner's Guide. Packt Publishing, 2011.

[7] Kosev, G., Mitreski, M. HTML5 Data and Services Cookbook. Packt Publishing, 2013.

URLS

[8] OGC: http://www.opengeospatial.org

[9] Comparison of web map services: http://en.wikipedia.org/wiki/Comparison_of_web_map_services

[10] Ohloh Code's comparison between OpenLayers, Leaflet and Cesium:
http://www.ohloh.net/p/compare?project_0=OpenLayers&project_1=Leaflet&project_2=Cesium+WebGL+Virtual+Globe+and+Map

[11] Google Maps API V3: https://developers.google.com/maps/documentation/javascript/reference?hl=en

[12] Google Earth API: http://developers.google.com/earth/?hl=en

[13] Google Terms of Services: http://www.google.com/intl/es/policies/terms/

[14] Nokia HERE Maps API: https://developer.here.com

[15] Nokia HERE Maps API Term of Use: http://here.com/terms/?lang=en-GB

[16] BING Mapas API: https://www.microsoft.com/maps/choose-your-bing-maps-API.aspx

[17] BING Term of Use: http://www.microsoft.com/maps/product/terms.html

[18] ArcGis API for Javascript: https://developers.arcgis.com/javascript/

[19] ArcGis Term of Use: https://developers.arcgis.com/en/terms/

[20] Openlayes: http://openlayers.org

[21] OSGeo: http://www.osgeo.org

[22] Leaflet: http://leafletjs.com

[23] Cesium: http://cesiumjs.org

[24] MapServer: http://www.mapserver.org

BIOGRAPHY

Ion Alberdi Bartolomé is a research assistant at Vicomtech-Ik4 GraphicsMedia.net in the area of Digital TV & Multimedia Services. He graduated in Informatics Engineering in 2011 at the University of the Basque Country (UPV-EHU). Nowadays he is studding a Master Degree in Advanced Web Systems. His research interest includes GIS, geo-visualizations, web technologies, advanced data visualizations and multimedia services.

Maider Laka Iñurrategi received her diploma degree in Telecommunication Engineering from the University of Navarra (2005). She is a staff research member of the Vicomtech-IK4 since 2005 in the field of Digital Television, Augmented Reality and GIS. She collaborated with the University of the Basque Country in the Computer Architecture and Technology as a teacher (2008). She received her Master Degree in Geophysics and Meteorology from the Universidad de Granada (2013). Her research interests include GIS, geo-visualizations, virtual reality and environmental information systems.

Dr. Igor G. Olaizola received a six years degree in Electronic and Control Engineering from the University of Navarra, Spain (2001) and a PhD in Infomatics from the Faculty of Informatics of San Sebastián, in the University of Basque Country (2014). He developed his master thesis at Fraunhofer Institut für Integrierte Schaltungen (IIS), Erlangen Germany 2001 where worked for a year as research assistant on several projects related to MPEG standard audio decoding. He is member of Vicomtech technological centre since 2002. Nowadays he is the head of the Digital Interactive TV and Multimedia Services department. In 2006 he also participated as a technology consultant in Vilau (www.vilau.net) company for one year. His PhD focused on automatic multimodal indexing and management of multimedia content.