Backend and Frontend Strategies for Deployment of WebGIS Services

Alexey Noskov and Alexander Zipf

Heidelberg University, Im Neuenheimer Feld 348, Heidelberg, Germany;

ABSTRACT

Nowadays, improving of accessibility of cloud computing services leads to increasing amount of WebGIS applications. First, internet maps were managed as static files. Then, interaction was implemented by Common Gateway Interface and server-side programming languages. Currently, WebGIS are built on top of advanced Web 2.0 solutions.

Geo-Spatial Data Repository (GSDR) is a web service developing for quality assessment of open geo-spatial data. GSDR is deployed in a computing cloud. A non-blocking web server allows handling multiple concurrent intensive requests. Requests can implement geoprocessing tasks required by users. Tasks are processed in-parallel using multiple CPUs. Utilization of Open Source GIS libraries enables to implement various geo-spatial algorithms. A central database allows multiple concurrent connections.

One of the most important challenges for modern WebGIS applications is providing responsive design suitable for different devices, such as desktop computers, laptops, tablets and smart phones. GSDRs frontend provides a generic responsive web design solutions, which may be applied for other map-based applications. The design approach was tested on various web maps implementing multiple visualization techniques including regular feature visualization by various shapes, colors and sizes, as well as, heatmap and tile-based visualization.

The found solutions were modularized into a set of relatively independent projects providing the source code and instructions. These projects are available through a number of public version control repositories. One can easily evaluate and utilize the described backend and frontend strategies for any kind of WebGIS applications.

Keywords: Online GIS, Parallel Geoprocessing, Cloud Computing, Responsive Web-Map Design, Web GIS, Non-Blocking Web Services

1. INTRODUCTION

Rich map-centered web applications, also known as WebGIS, play an important role today. Millions of people use such applications for routing, address search, book rooms in hotels or just to look around. Thousands of applications are provided by private companies, government institutions and even by independent enthusiasts. Many types of WebGIS are delivered. Generic WebGIS services, like Google and Bing Maps, allows the user to find objects, calculate the shortest way, observe areas in 3D etc. With high-resolution imagery users can even recognize their cars.

On the other hand, specialized WebGIS provide narrowed and focused on a concrete scope services. For instance, such applications can provide advanced tools for evaluation of an ecological situation. Advanced tools, like buffering or digitizing, are often delivered by specialized services.

In this heterogeneous and dynamic environment, both users and developers want to have some "rules of the game". For users, frontend strategies plays an important role. The fronend needs to be familiar and comfortable to a user, even is she visits an application for the first time. As users, developers also need such kind of common rules. Frontend development and support requires significant resources. Classic desktop applications are able to exist independently from developers after the release; web applications require 24/7 monitoring and support. Web in general and WebGIS in particular are developed in an extremely dynamic environment. It is hard to

Further author information: (Send correspondence to Alexey Noskov)
Alexey Noskov: E-mail: a@n-kov.com, Telephone:+49 6221 54-5506, a.n-kov.com

imagine if a popular WebGIS service will survive for five years without multiple significant revisions and intensive every-day support.

In this work, we try to recognize general rules for WebGIS deployment actual for these days. We develop and support a WebGIS service, Geo-Spatial Data Repository (GSDR),[1] for quality assessment and improvement of geo-spatial data utilized, harvested, processed and delivered in a frame of a WeGovNow[2] project. GSDR automatically assess geo-spatial data involved in the project and represents results of quality assessment in a form of multiple interactive maps. Three types of data are assessed: OpenStreetMap in pilot sites (a base map for all WeGovNow components), Public Sector Information provided by municipalities in a form of geo-spatial data layers, and logged users spatial contributions. Additionally, GSDR provides facilities for quick real-time quality assessment and improvement tools (e.g., auto-snapping, map-objects-picking, spell-checking etc.) applicable for small peaces of data, normally, provided by components of WeGovNow.

In Section 2, a brief history and most important aspects behind WebGIS are provided. Backend and frontend strategies are described in Sections 3 and 4 correspondingly. In Section 5, a processes of the fronend and backend integration is discussed. Conclusions are provided in Section 6.

2. BRIEF HISTORY OF WEBGIS

WebGIS are developed for more then twenty years ago. The first application (PARC Map Viewer) was presented in 1994.[3] Putz wrote the following sentences. "The Interactive Map Viewer is a World-Wide Web application developed at the Xerox Palo Alto Research Center. This application combines the ability of HTML documents to include graphical images with the ability of HTTP servers to create new documents in response to user input. The HTML format and the HTTP protocol are used to provide a custom user interface for browsing and viewing geographic map data." With this sentences the age of WebGIS has been started.

One can mention that Puts did not use the term "WebGIS". In a document "Thoen, Bill,1995, Web GIS: toy or tool? http://www.gisnet.com/gis/webgis.html" the term was applied, probably, for the first time. To the best of our knowledge, web page is lost. In this work we spell the term without space. Alternatively, various terms can be used by different authors: "Web-based GIS", "Web Map", "Web-Map Service" etc. According to Google Adwords, average monthly searches for the mentioned terms are as follows: 10K-100K for "WebGIS" and for "Web GIS" (but bigger for the former), 1K-10K for "Web Map" and "Web-Map Service", 100-1K for "Web-based GIS". Some popular search words, like "GIS Online", have a higher search rank (1M-10M), but cannot be used to express the considered topic precisely. Therefore, "WebGIS" is used in this article.

As mentioned above, the WebGIS age was started in 1994. History of the topic was assessed by us using Google Scholar with a search text: *"web gis" or "webgis" -site:com.cn* applied separately for each year from 1994 to now. Results are presented in Fig. 1. The term was not used in 1994 and 1996. As mentioned earlier, Putz[3] used a long definition. For the first time among the indexed publications "WebGIS" was mentioned by Roumani-Denn[4] as a reference to a web document. However, only from 1998 the term has started to spread world wide. Surprisingly, after being briefly mentioned in a number of American articles, the term was used in publications in Chinese and Korean language. Chen et al.[5] proposed "to develop WebGIS and draft of the standard and normal for geo-spatial data and their exchange". Then, in 1999, the term was appeared in European publications including publications in English, Italian[6] and German.[7]

Real burst of the publications has begun from 2000 (18 publications in this year). For the first time, the term was bunched with SQL,[8] CAD[9] and VRML. Seemingly, the burst was significantly leveraged by releasing UMN Mapserver[10] and ArcIMS[11] in 2000 and Geoserver in 2001.[12] And then, movement was supported by Nasa World Wind in 2003.[12]

According to Fig. 1, from 2004 the number of publications was increased dramatically and achieved sort of stagnation from 2010 remaining still popular. In 2004, OpenStreetMap and Yahoo Maps projects have emerged. Google Maps has started in 2005.[13] These events lead us to the current world of WebGIS. Tiled web maps[13] changed the rules of the game. They made web maps extremely popular. It was achieved by simplicity of the tiled web map technology and ability to easily manage thousands and millions of users' connections. OpenStreeMap formed an novel environment where spatial data are free and available to everyone even for a commercial use. The last was proved by a successful start of MapBox in 2010.[13]

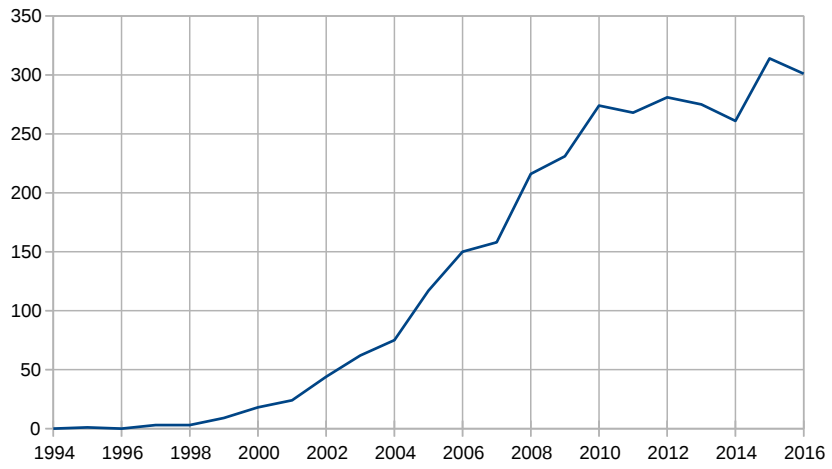

Figure 1. Number of WebGIS publications indexed by Google Scholar (1994-2016)

3. GIS BACKEND SOLUTIONS

3.1 Web Hosting

First of all, a developer deploying a web site encounters with a problem of hosting. Multiple tutorials propose users do deploy web solutions on *localhost*. Then many users try to access their *localhost* from internet; in most cases, it is difficult to implement.[14] In few cases, users have dedicated IPv4 address. This allows self home hosting. For internet companies resided in IT hubs Internet Service Providers usually propose dedicated IPv4 address for monthly payment. In this case, companies can set up their own servers. It requires significant resources including staff, room, electricity, hardware support etc.

Tyler[14] propose several solutions to resolve the problem the lack of IPv4 problem. Using of abundant IPv6 addresses was not considered due to most of users have only IPv4 network connection. Others solution relied on tunneling and port forwarding. NAT solutions and the lack of IPv4 lead to such problems. IPv6 can be a possible solution in case of wide spreading in the future. Alternatively, decentralized networks, like I2P and Tor, can be utilized prospectively.

Normally, self home/office hosting is not considered to non high traffic web sites.[15] Reasonable options for medium and small web sites are as follows: Shared, Virtual Private Server(VPS), Dedicated and Cloud hosting. Shared hosting provides a low-cost solutions; it is impossible to obtain the root access to a server. Thus, only preinstalled and preconfigured software can be used with a limited room to manipulations. Normally, PHP-MySQL solutions are provided. For WebGIS, non-standard software (e.g., Geoserver, PostGIS, GDAL/OGR) is required,[12] therefore, shared hosting is not usually considered.

All remained solutions provide the root assess and, as a result, allow full server control. VPS hosting provides a virtual machine started on a physical server in parallel with other virtual machines. Dedicated hosting allows use of a physical machine without sharing resources. Both VPS and dedicated solutions relied on a physical server;[15]it harms to "uptime" of services. This problem is not critical for cloud solutions.

Cloud hosting relies on a bunch of distributed physical servers.[15] Hence, a fault on one machine does not lead to an interruption of web services, as a result, cloud systems are more sustainable. What is more, today, low-cost cloud solutions are available. This is especially relevant to ARM-based architectures. Scaleway Rackspace, Amazon AWS, Linode, Joyent Cloud, GoGrid, OpenStack, DigitalOcean, Parallels, Proxmox and LXC are popular cloud hosting providers.[16]

For GSDR we have chosen the Scaleway cloud hosting located in Amsterdam (Netherlands) with the following characteristics: 4 Dedicated ARM cores, 2GB Memory, 50GB SSD Disk, 1 Flexible Public, 200Mbit/s Unmetered bandwidth, 1Gbit/s Internal Bandwidth. The ARM architecture in 4 time cheaper than x86 64bit one and, at the same time, quite effective. Because of lower popularity of ARM machines some precompiled binary software

packages are not available; they have to be build from the source code. We did not observe problems related to the work of the service, but the current number of users is relatively low, thus, only after one more year it will be possible to evaluate quality of the service under the real load. The current price of the service is lower than a standard price for VPS products. It is important because currently GSDR is under active development and, hence, does not require richer resources. In the future, under the real load, the sever characteristics can be upgraded without reinstallation of the software.

3.2 Review of Server-Side Technologies

WebGIS services can be resided on various operational systems, including Windows family and Unix-like distributions (e.g., Debian GNU/Linux).[10] For deploying and shipping source code of GIS project distributed on various operational systems, container tools, like Docker,[17,18] can be effectively utilized. Moreover, some containers allows users to pack all required dependencies and produce a single executable file that can be started in various operational systems without modifications. For instance, Starpack is designed for suck tasks to deliver Tcl/TK projects.[19,20] In order to effectively manage and track the source code of WebGIS project, various version control systems are used. Among them Fossil is distinguished by simplicity, compactness and rich functionality.[21] It works through standard http/https ports and provides the following facilities: native web frontend, wiki, user authorization and bug tracking system, GCI interface support, Tcl-like language for extensions. Moreover, Fossil stores everything into a single SQLite file which can be easily moved into another location or even sent by email.

The source code of WebGIS is run by various web servers, like Microsoft ISS, NGINX, Apache, Tomcat.[22] By default web servers manage static files (usually, html, css, js). However, they can be configured to generate dynamic web pages using CGI, FastCGI, SCGI, WSGI (for Python), PSGI (for Perl), Rack (for Ruby) programming interfaces.[23] Apart from that, popular web servers offer extensions allowing effectively utilize various programming languages. For instance, Apache Web Server has mod_php for PHP programming language and Apache Rivet for Tcl.[20] The latter is designed for MPM (Multi-Processing Module) prefork which implements a non-threaded pre-forked web server allowing thread safe parallel processing of HTTP requests. Another example is NaviServer (originally designed for Tcl programming language) providing nsphp module for PHP support. Alternatively, some programming environments provide encapsulated solutions which do not require external web servers (e.g., Node.JS, Java) or implement small independent web servers delivered with a programming language distribution (e.g., SimpleHTTPServer of Python, TclHttpd of Tcl).

3.3 Implementation

From review of technologies 3.2, we have decided to use a bunch of Debian-Apache-Tcl. It is described in detail further. In general, the bunch enables us to rich a holistic deeply integrated programming environment. A similar environment has been deployed on desktop machines. That allows us to develop the web service in a very effective distributed secure manner.

Debian GNU/Linux 8 "jessie" has been selected as operational system. It was the latest stable release when the hosting was obtained one year ago. The distribution will be supported by community by April/May 2020. After terminating the further support, the system will be upgraded to a latest one. Today, Debian "jessie" is described as "old stable". Currently, Debian is one of the most popular GNU/Linux distribution, it is supported by most hosting providers (excepting the shared hosting which is relied mainly on RHEL and CentOS GNU/Linux distributions). Thus, it is easy to switch a hosting provider; even migration to another GNU/Linux or BSD system is a quite trivial procedure, because most underlying software is standard. Moreover, we use Debian on desktop computers. It is very continent to use a same system on the server and the desktop. Most of required software packages Debian provides as a part of a default system, others has been installed from a Debian's repository (using the classic command *sudo apt-get install PackageName*) or compiled from the source code.

Apache Web Server handles HTTP requests. Several instances of GSDR will be managed by Apache as virtual hosts. Currently, two instances are managed: a frozen third prototype and an under-development version (towards a first pilot version, scheduled release is May 2018). A domain name has been parked and SSL certificates has been configured for all instances of GSDR, thus, the web service is accessed through the secure encrypted HTTPS port. The Apache Rivet module is responsible for generating the dynamic content. It is designed

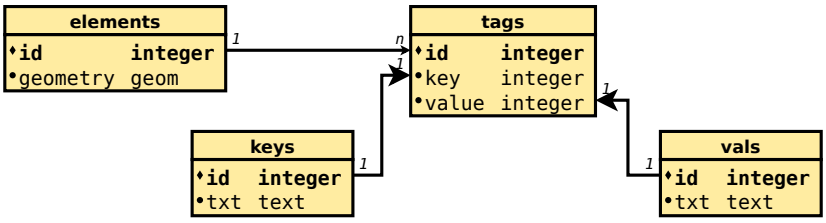

Figure 2. An universal model of geo-spatial data.

to extend Apache Web Server by the Tcl programming language. Rivet enables Apache to handle multiple concurrent requests and utilize all CPU cores in parallel effectively.

Tcl is a main programming language we use for GSDR's server-side functionality. First of all, web content is generated and requests are handled by Tcl. Apart from that, this programming language is utilized for the following tasks. A number of Cron scheduled jobs are set as executable **.tcl* files. A number of external libraries and software intensively utilized by GSDR has been written in Tcl. *Expect* (an extension of Tcl) will be applied for automation of remote tasks. Moreover, we use extended and configured tclsh (Tcl's programming shell) instead of Bash shell. Additionally, SQL code is extended by Tcl and a Tcl-like language is used as an extension language for a version control system of GSDR's code.

For data management, SQLite database is used. This is an extremely light weight and simple solution in comparison to MySQL/MariaDB or PostgreSQL. At the same time, it is very functional and extendable. SpatiaLite is an spatial extension which offers almost similar to PostGIS functionality. One of the biggest advantages offered by SQLite/Spatialite is ability to store data in a single file. It is extremely easy to manage these files (e.g., make backups, use various versions of database, etc.). It is not recommended to use SQLite for high-traffic web sites, moreover, SQLite cannot effectively handle multiple writing operations in parallel. In order to adequately manage concurrent read operations, SQLite has been built from the source code with enabled Multi-Thread mode. At the same time, all preforked Apache instances were configured to be implemented in a separate environment with independent connections to the SQLite database. This enables us to adequately process hundreds of concurrent request. Enabled Apache caching functionality allows GSDR to easily balance high load.

Originally, SQLite has been designed for Tcl by Richard Hipp. Hence, it is convenient to use it from the Tcl environment. Furthermore, SQLite can be easily extended by many programming languages. In contrast to PostgreSQL (i.e., PL/Tcl,PL/Perl,PL/Python), SQLite can be easily extended by a programming language function which does not need any adoption and can be used from normal packages and extensions of a programming language. Using this, we extended GSDR's SpatiaLite database by a number of spatial functions provided by Tiles Common Framework(Tiles.CF).[24] That allows the service to support advanced web tile operations. Moreover, a number of tools provided by Integrated GIS Tool Kit (IGIS.TK)[25] are utilized to generate and process GSDR's SpatiaLite databases. The both mentioned open source projects were originally developed as a part of GSDR and, then, extracted from it to allow using solutions in other projects by everyone.

GSDR and IGIS.TK utilize a novel spatial data model introduced in Fig. 2. According to the figure, a main table of the database is *element* which represents a spatial feature comprised of an *id* number and *geometry* object. The *tags* table establishes connections between elements, keys and values. *keys* and *vals* tables stores unique text objects preventing duplicated strings. It reduces the size of database significantly. *elements* has one-to-many relationship with *tags*, while *tags* has one-to-one with *keys* and *vals* tables. In other words, *elements* can contain many *tags*, while the *tags* table forms objects consisting one key and one value. One can notice, that our data model inherits in some sense GeoJSON data model which allows storing various number of objects' characteristics. In contrast to this, classical geo-spatial data model (e.g., ESRI's shape file or MapInfo's TAB files) utilizes a "table data model" when every object has an equal number of properties (with null-values for undefined properties).

GNUMakeTcl. In order to effectively deploy and support several instances of GSDR and manage its dependencies, a collection of general-purpose open source Tcl libraries and tools[26] was developed. *Make* is

a GNU Make file for installation and removal of a Tcl package (extension). *nkov.tcl* comprises a number of functions for debugging and profiling the code, reading large text files, generating random object (e.g., color), converting the number base, generating JSON data, intersecting and selecting geometries by bounding box (light implementation). These functions are utilized by GSDR, IGIS.TK and Tiles.CF, thus, in order to prevent duplication of code, they are provided separately from the mentioned products. *GNUMakeTcl* is a main tool of the collection. It extends standard GNU Make package available for most operational systems distributions. GNU Make controls the generation of executable and other non-source files of a program from the program's source files. We have extended the tool by Tcl and use it to automate GSDR and its dependencies deployment and remove duplicating pieces of the source code.

Currently, *GNUMakeTcl* is used for deployment of GSDR instances and as a static files generator (for Tcl, HTML, JavaScript and CSS files). Each file processed by *GNUMakeTcl* passes of two iterations of a preprocessing, main process and postprocessing. The preprocessing reads common header files in a directory tree, the main process evaluates a building file, the postprocessing reads footer files in a directory tree. In each step, Tcl commands encapsulated in header/footer and building files are evaluated with regards to evaluated Tcl code on previous stages. It enables us to use in footers and headers variables set in a building file. In other words, if a date of latest modification was set in a building HTML file, this information can be displayed by a common header of footer. In any step, we can execute any command. For instance, during footer execution (preprocessing), a footer can contain some Tcl code to create a destination directory, a building Tcl file can copy to a created directory some dependencies or even download something (e.g., a PNG file) from the internet, and, finally, during a footers execution, some files can be made executable (*chmod +x* in Unix systems) or removed. Thus, *GNUMakeTcl* is a flexible universal tool for the project deployment. In the future, it will be used with TclKit (a Docker-like older solution provided by Tcl) for shipping products.

The *GNUMakeTcl* deployment system is deeply cooperated with version control systems. By default, it uses the RSC which is a simple system based on plain text files. *GNUMakeTcl* checks out a latest version of code from a RCS file. RCS manages General-Purpose Tools and Libraries.[26] GSDR, IGIS.TK and Tiles.CF are managed by Fossil SCM.[21] It provides more advanced facilities, like wiki, users authorization, ticketing etc. At the same time, Fossil is a very simple system. It uses a single SQLite file for data repository and requires just the HTTP access. Using Fossil, we have developed a distributed decentralized system of source code repositories. The repositories can be synchronized in various directions.

In this section we have provided a detailed description of the server-side architecture. One can conclude that the architecture is holistic and deeply integrated. At the same time, several components are hosted separately. Tcl is an universal programming language widely applied in various ways (i.e., Apache Rivet, extensions and packages, Fossil, tclsh). We have manage to use only this language for all server side tasks. The SQLite database management system is utilized as data storage for the WebGIS service and version control system.

4. WEBGIS FRONTEND

4.1 Review of Client-Side Technologies

The evolution of the WebGIS's frontend is strongly related to the evolution of web browsers and web standards. In 1990, Tim Berners-Lee invented HTTP, HTML, the first web browsed and the first web server. The first web page developed by Berners-Lee is still available online.[27] Further history of Web is well illustrated with the aid of a "Browsers and Technologies" diagram provided by an *evolutionoftheweb* web site.[28] According to the diagram, Web passes the following important benchmarks. In 1992 a Mosaic browser relied on already existing HTTP and HTML1 was released. New browsers, NetScape, Opera and Internet Explorer, started their way in 1994-1995 years. They were already relied on HTML2, Cookies and SSL technologies. In 1996-1999, several popular client side solutions were introduced XML, HTML3.2, HTML4, CSS2 and AJAX.

A period from 2001 to 2007 is a stage toward HTML5. During this time, SVG, Canvas and XMLHTTPRequest2 solutions has been introduced. Safari and Firefox have started their way in 2002 and 2004 correspondingly. From 2008, Web has switched to the age of HTML5. This is strongly related to a first release of a Chrome web browser which has implemented HTML5. Since then, several important technologies has been introduced: CSS3, Geolocation, WebGL, WebRTC, Touch Events, Full Screen API etc. The modern WebGIS frontend is strongly relied on HTML5 facilities.

4.2 Implementation

Our WebGIS frontend is developed in the responsive manner. It means that various devices (i.e., desktop computers, laptop, tablets, smartphones) can correctly render offered web pages. Also, using facilities of CSS3 and JavaScript, same client-side web code represents various views of web pages. In small screens menus and panels are minimized, while in big screens they are maximized. Thus, responsive web applications are automatically modified for various web browsers and screens.

Regarding the web page layout, we use classical for WebGIS design with the left-side panel. The panel can be easily resided on right for Right-To-Left writing languages (e.g., Hebrew). The global menu is resided on the top and footer is resided on the bottom of a web page. A map is a central part of our frontend; it is stretched and occupies available empty space of a web page. In Fig. 3, the GSDR's fronend is demonstrated for big screens (left) and small smartphone screens (right).

The left view in accessible by the following URL:

`https://wgn-pt3.gsdr.gq/qview.rvt?curlay=3&attr=DAt&area=SanDona&layer=OSM-ONM.PSI&cx=1398840.073&cy=5721212.278&zoom=19&rotate=0`

This big screen layout consists of several component. A menu panel is maximized; all menu elements are visible (*PSI,OSM,PSI OnToMap-OSM,Docs*). On the left home button with a title are resided. *About* button in on the right. The left side panel is scrollable. If a content does not fit to the panel a user can scroll down to see a hidden part. In the figure, the content fits to the panel, hence, scroll bars are invisible. The side panel contains several control elements; users can select various parameters and, then, click an *Apply* button to apply changes. A *Show data layer* button enables the user to portray various vector data layers. A *hide* button hides the side panel to increase area of a map element.

The map element is a main component of the layout. The OpenLayers 4.6 library is used. OpenLayers is a WebGL-based library. Thus, it allows full-screen maximization and rotation. Moreover, WebGL enables to manipulate with raster web tiled map. We convert "on-the-fly" standard OpenStreetMap color tiles to grayscale raster tiles. Additionally, various tools for raster manipulations can be applied (e.g., changing contrast and brightness, inverting colors etc.). WebGL functionality is very convenient for an advanced map analysis. The presented URL encodes parameters defined by a user (*Area, Layer,Color by, Data Layer*) and a current configuration of a map (coordinates of a central points,a zoom level, a map rotation value). The frontend implements of a JavaScript class. The class tracks all changes and modifies URL. This is convenient, because it is not required to manage Cookies and users may easily save a map in bookmarks or share URL.

On the right side of Fig. 3, a mobile view is presented; the web page is accessibly by the following URL:

`https://wgn-pt3.gsdr.gq/qview.rvt?curlay=3&attr=DAt&area=SanDona&layer=OSM-ONM.PSI&cx=1398842.761&cy=5721230.193&zoom=19&rotate=0.62604`

We use an event listener for automatic minimizing (maximizing) of the side panel interactively if a width of a web browser decreases lower 700 pixels. The side panel is minimized into a vertical bar. Users can touch (click) this bar to maximize it. A button resided in the top-right corner of the map widget is responsible for either reset rotation or full-screen mode. The map canvas can be rotated using *Shift-LeftMouse* on a desktop computer or by two fingers on a touch screen devices. Combining this with full-screen mode yields significant map-usability benefits for users, especially in touch screens. Leaflet, a very popular alternative to OpenLayers, does not provide such facilities.

The presented frontend strategy provides a flexible solution for various web-map applications. It was tested on various devices including desktop and laptop computers, tablets and smart phones. The source code of our solutions can be evaluated and adopted by everyone for WebGIS projects, thanks to openness of web technologies.

5. ASSEMBLY

The defined backend and frontend solutions has been described earlier. Assembly of the solutions forms the GSDR WebGIS service. Currently, the third prototype of the service is available. It provides series of interactive

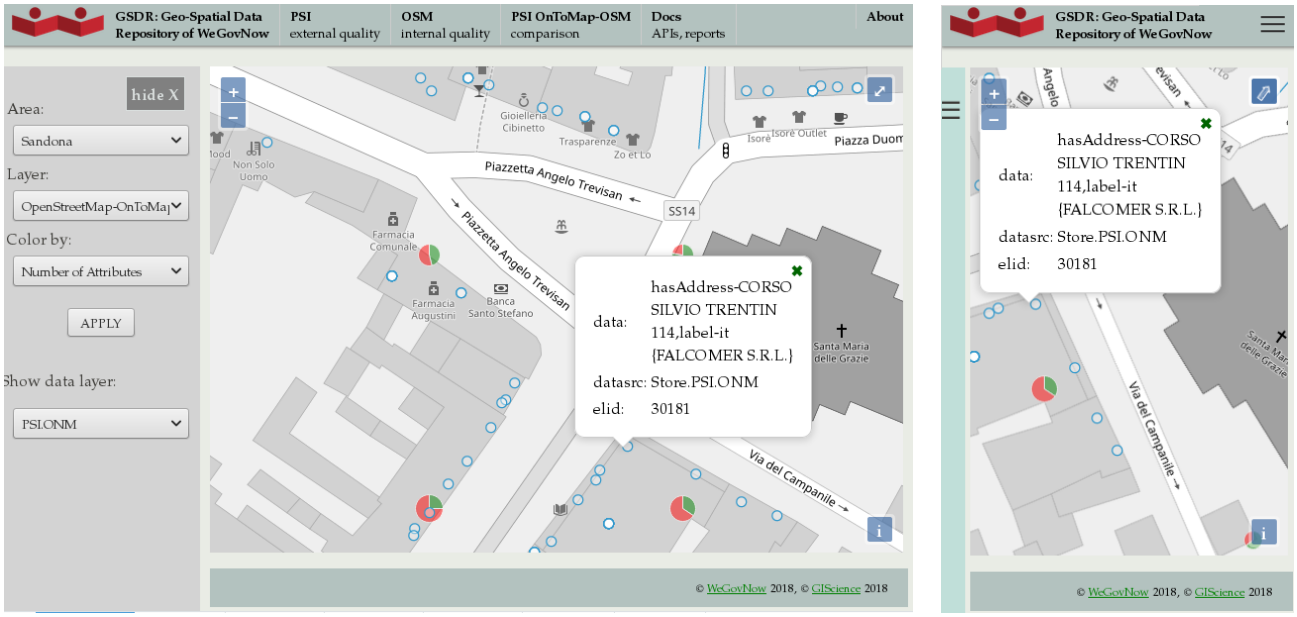

Figure 3. WebGIS's responsive layout: desktop(left) and mobile(right).

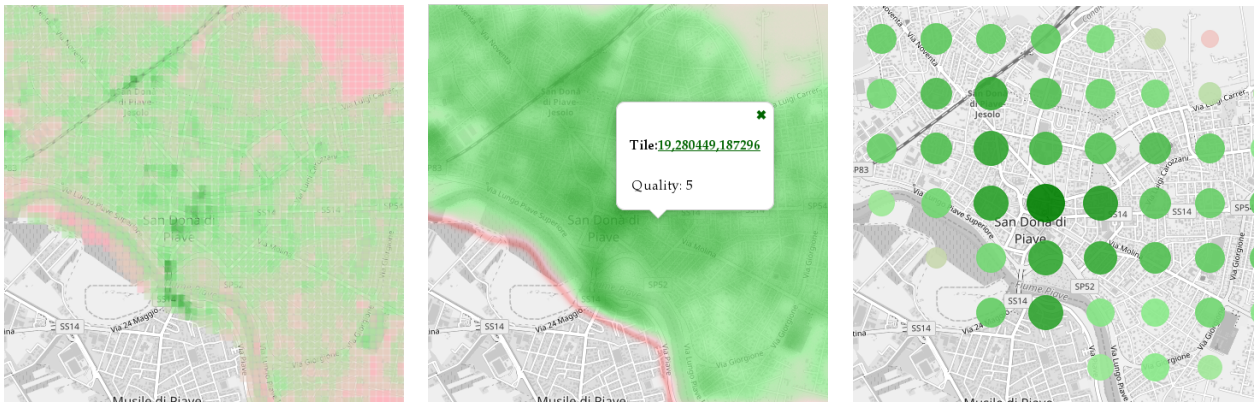

Figure 4. Map views from left to right: features , heatmap and tiles

web maps and a number of tools for improvement of users input. Further, more facilities will be delivered by the service.

GSDR's maps represent spatial data inventory results involved into the WeGovNow platform. The inventory was implemented by tiles. Various aspects were assessed (e.g., number of points, length of lines, area of polygons, number of attributes). Results provide an insight into the considered geo-spatial data. A mock aggregated data quality values were calculated. In Fig. 4, examples of maps are presented. Currently, maps are available in feature, heatmap and tile views. The feature view represents row data using a square icon colored according to a value. The heatmap view represents blurred features; it gives a smooth representation of spatial data. The tile view can be used for multiple zoom levels, because features portray aggregated by tiles data according to a zoom level, thanks to the IGIS.TK library. This view type allows users to represent values by both size of feature and color. Currently, in the tile view features are aggregated according to the rule $zl + 2$, i.e., current zoom level plus 2. This rule can be changed easily.

Besides, GSDR provides a number of tools for quick user input evaluation to be utilized by WeGovNow

components. Now, the following tools are offered: text autocompleting, spell checking, picking vector features on a map, client and server side snapping functionality. For text input improvement the text stored in *keys* and *vals* tables of the spatial database (see Fig. 2). Picking objects on a map allows users to simplify input, because with this tool users do not need to digitize new objects on map; they just need to use an existing object. Snapping functionality simplifies digitizing. OpenStreetMap data and Public Sector Information Open Data are utilized by the described service.

6. CONCLUSIONS

In this work, we propose solutions for deploying backend and frontend components of WebGIS services. A consequent workflow has been presented. The implementation has been conducted for the Geo-Spatial Data Repository (GSDR) designed as a web service of WeGovNow for data quality assessment and improvement.

A holistic and integrated Tcl-centered backend has been proposed. The backend provides an agile developer-friendly environment. Using the found solutions, three independent open source project has been derived from GSDR's backend code. That allows everyone to use the code in other projects. Also, we have demonstrated the flexibility of SQLite-based database solutions. Extendability and simplicity of SQLite make it a good candidate to be intensively applied in medium and small size projects. It has been discovered that compilation of SQLite/Spatialite in multi-threading mode enables to effectively handle multiple requests. The SQLite's inability to manage multiple writing operations (because of file-oriented nature of SQLite) is obstacle to apply it in big-size projects with intensive users contributions.

The proposed WebGIS frontend solutions are developed in a responsive manner. Hence, applications based on such solution can be used on various devices including big-screen desktop computers and small-screen mobile devices. Moreover, intensive utilization of WebGL based map API enables to take advantages of such facilities like full-screen maps, map canvas rotation, manipulation with colors of raster tiles of a background map. Navigation information and a current web page configuration are reflected by URL. That simplifies sharing of a web page.

Currently, GSDR covers only one pilot site. Further, GSDR will cover all pilot sited involved into the WeGovNow platform. What is more, the mock data quality indicator will be replaced by real indicators. A list a data components represented by GSDR's interactive maps will be extended. Tools for quick improvement of users input will be utilized by WeGovNow components. GSDR's findings will be shared through the introduced open source projects.

ACKNOWLEDGMENTS

This work has been funded by the European Union's Horizon 2020 research and innovation programme under grant agreement n. 693514 ("WeGovNow"). The article reflects only the authors' view and the European Commission is not responsible for any use that may be made of the information it contains.

REFERENCES

[1] Noskov, A., "Geo-Spatial Data Repository for Grand Quality, WeGovNow Data Quality Assessment." `https://wgn.gsdr.gq/` (2018). [Online; accessed 1-March-2018].

[2] Boella, G., Francis, L., Grassi, E., Kistner, A., Nitsche, A., Noskov, A., Sanasi, L., Savoca, A., Schifanella, C., and Tsampoulatidis, I., "Wegovnow: a map based platform to engage the local civic society," *Proceedings of the 8th International Workshop on Location and the Web - LocWeb'18* (2018).

[3] Putz, S., "Interactive information services using world-wide web hypertext," *Computer Networks and ISDN Systems* **27**, 273280 (Nov 1994).

[4] Roumani-Denn, V., "Data for gis: the essential ingredient," *Geoscience Information Society* , 31 (1995).

[5] Jicheng, C., Li, T., and Zhao, Y., "Information super highway and geographical infromation system," *Journal of Image and Graphics: Series A* **3**(6), 485–489 (1998).

[6] Budoni, A. and De Bonis, L., "Processo di pianificazione del piano di parco e governance dello sviluppo locale," in [*Federalismo, integrazione europea e crescita regionale, XXX Conferenza AISRe*], (2009).

[7] Kurzwernhart, M., "Neue gis-technologien–was bringen sie in der praxis," *Österreichische Zeitschrift für Vermessung & Geoinformation* (1), 13–21 (1999).

[8] Wang, S., Loh, C., and Huang, T., "Developing a virtual disaster information management and analysis system in internet for chi-chi earthquake," in [*Proc., Int. Conf. on Engineering Education*], (2000).

[9] Kim, H.-S., Im, J.-H., Kim, J.-D., and Shin, S.-E., "Development of gis application using web-based cad," *Journal of the Korean Association of Geographic Information Studies* **3**(3), 69–76 (2000).

[10] Mitchell, T., [*Web mapping illustrated - using open source GIS toolkits.*], O'Reilly (01 2005).

[11] Fu, P. and Sun, J., [*Web GIS: Principles and Applications*], Esri Press (2010).

[12] Davis, S., [*GIS for Web Developers: Adding where to Your Web Applications*], Pragmatic Bookshelf Series, Pragmatic Bookshelf (2007).

[13] Muehlenhaus, I., [*Web Cartography: Map Design for Interactive and Mobile Devices*], Taylor & Francis (2013).

[14] Belimpasakis, P., Saaranen, A., and Walsh, R., "Home dns: Experiences with seamless remote access to home services," *2007 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks* (Jun 2007).

[15] Bervoets, W., [*Fast, Scalable And Secure Web Hosting For Web Developers: Learn to set up your server and website*], leanpub.com (2016).

[16] Komarek, A., Pavlik, J., Mercl, L., and Sobeslav, V., "Vnf orchestration and modeling with etsi mano compliant frameworks," *Internet of Things, Smart Spaces, and Next Generation Networks and Systems* , 121131 (2017).

[17] Merkel, D., "Docker: Lightweight linux containers for consistent development and deployment," *Linux J.* **2014** (Mar. 2014).

[18] Swetnam, T. L., Pelletier, J. D., Rasmussen, C., Callahan, N. R., Merchant, N., Lyons, E., Rynge, M., Liu, Y., Nandigam, V., and Crosby, C., "Scaling gis analysis tasks from the desktop to the cloud utilizing contemporary distributed computing and data management approaches: A case study of project-based learning and cyberinfrastructure concepts," in [*Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale*], *XSEDE16*, 21:1–21:6, ACM, New York, NY, USA (2016).

[19] Flynt, C., [*Tcl/Tk: A Developer's Guide*], The Morgan Kaufmann Series in Software Engineering and Programming, Elsevier Science (2012).

[20] Roseman, M., "Using starkits for easy deployment of server applications," *http://markroseman.com/pubs/deployserverapps.pdf* (2013).

[21] Confer, W., "Distributed project control management: Using fossil to get beyond source control," *J. Comput. Sci. Coll.* **28**, 67–68 (June 2013).

[22] Peterson, M. P., [*Online Maps with APIs and WebServices*], Springer Berlin Heidelberg (2012).

[23] Kapranoff, A., [*Nginx Troubleshooting*], Community experience distilled, Packt Publishing (2016).

[24] Noskov, A., "Tiles Common Framefork: a library for web tiled maps." `http://tiles.cf` or `http://tiles.n-kov.com` (2018). [Online; accessed 1-March-2018].

[25] Noskov, A., "Integrated Geographic Information System Tool Kit: a collection of high-level tools for spatial data conversion, fusion, generalization and quality managment.." `http://igis.tk` or `http://igis.n-kov.com` (2018). [Online; accessed 1-March-2018].

[26] Noskov, A., "Collection of General-Purpose Tools and Libraries." `http://n-kov.com/LibsAndTools/` (2018). [Online; accessed 1-March-2018].

[27] Berners-Lee, T., "World Wide Web." `http://info.cern.ch/hypertext/WWW/TheProject.html` (1990). [Online; accessed 1-March-2018].

[28] "The Evolution of the Web." `http://www.evolutionoftheweb.com` (2018). [Online; accessed 1-March-2018].