MAPMINT: THE WEBGIS IN THE CLOUD

Nicolas Bozon[1] , Gérald Fenoy[2] and Naoki Ueda[3]

[1]Media Center, Osaka City University
3-3-138 Sugimoto, Sumiyoshi-ku, Osaka 558-8585, Japan
Email: nicolas.bozon@gmail.com
[2]GeoLabs sarl, Futur Building 1
1280 Av. des Platanes, 34970, Lattes, France
Email: gerald.fenoy@geolabs.fr
[2]Cartoworks Inc., Osaka Station Square
1-11-4 Umeda, Kita-ku, Osaka 530-0001, Japan
Email: naoki.ueda@cartoworks.com

ABSTRACT

MapMint is a comprehensive task manager for implementing and managing spatial data infrastructures (SDI). It is a complete and robust authoring platform to organize, edit, process and publish spatial data to the Internet. MapMint is based on extensive use of the Open Geospatial Consortium (OGC) specifications, and allows an orchestration of Web Feature Services (WFS(-T)), Web Mapping Services (WMS), Web Map Tiling Services (WMT-S) and Web Processing Services (WPS) through a modular and intuitive Web application.MapMint core is based on ZOO-Project and MapServer open source software. Every functions are run as WPS requests, calling both general services (for configuration, data and project management, user permissions etc.) and geospatial data management services based on MapServer (access to features, .map file writing/editing, symbology and classification). Spatial data is published as WMS and/or WFS by MapServer and can also be cached according to the WMTS principle.Further, geospatial data abstraction and geoprocessing services are powered using the GDAL/OGR library as WPS (for format and projection management, vector and raster operations, spatial analysis). MapMint is available through a modern administration interface built upon jQuery and OpenLayers Javascript libraries. It proposes a complete and user-friendly back-office to control and deploy standardized Web maps using data from local or remote datastores, process it, create new layers, and finally organize Web mapping projects to be published online. MapMint architecture and main features will be introduced in this paper, and its four inter-connected modules will be described. Several real world example will also be presented.

1. INTRODUCTION

Progress of geographic information systems (GIS) and the more systematic use of the Open Geospatial Consortium Webservices (OWS) has led to a variety of available technologies and methods to store and spread GIS data over the Internet. Standardization of spatial data and metadata have become crucial in the context of collaborative Web GIS development, but also due to specific directives or policies regarding data use and sharing, such as the INSPIRE directive for the European context. The quiet recent but very fast development of new Web GIS techniques (partly due to new opportunities offered by Web 2.0 and Cloud Computing), is leading to a growing public and governmental awareness on the necessity of using standards for Web-based spatial data infrastructures (SDI).

Numerous tools are available today to strore and spread spatial data over the Internet, through the Web Map Service (WMS), Web Map Tiles Service (WMTS) and Web Feature Service (WFS). Many open source or proprietary GIS solutions now supports such standards and these methods have become very popular. By contrast, only a few solutions are available for processing such data through Web Processing Service.

The MapMint solution is developped in such a context, where both geographic data processing, rendering and display needs to be standardized and available as fast and efficient Web Services. This service-oriented aproach allows the use of complete and rich WebGIS solutions in the Cloud, and the user can thus access and use advanced geospatial algorithms using his own data from an Internet browser only.

The MapMint architecture will first be presented in this paper, in order to demonstrate the advantage and efficiency of the WPS standard for building such a Internet platform. The implementation and orchestration of WPS, WMS, WMTS and WFS is first explained, using and demonstrating the capabilities of the selected software. The resulting platform and cartographic client applications are then presented, with details on their dynamic creation, as well as on the functionnalities proposed to the end-user. Some real world examples are finally presented and illustrated, using and publishing open data with the MapMint software.

2. MAPMINT ARCHITECTURE

2.1 Using Open Source geospatial software together

Several well-known and stable open source software were chosen for our research and development. The Open Source Geospatial Foundation (OSGeo) provides numerous quality software and acts as an umbrella foundation to help them grow and spread, for both academic and industry GIS applications. Among this software stack, some are written and C language and appared to be generic enough for building the targeted platform, namely ZOO-Project, GDAL/OGR and MapServer for the server-side, and OpenLayers for the client-side.

Once these software were selected, the decision to make them interact according to the available standards from the Open Geospatial Consortium (OGC) was taken. WPS naturally appeared as a standard of choice for gathering the selected libraries and software, as most of them could be run as standardized Web Services. The resulting data would then be published as WMS and WFS, using the existing functionalities available in MapServer and OpenLayers.

ZOO-Project was thus chosen as the core component of MapMint, and WPS is thus the corner-stone of the proposed system. ZOO-Project is a server-side C Kernel which makes it possible to create, manage and chain WPS 1.0.0 compliant Web Services, by loading dynamic libraries and handling them on-demand (Fenoy and al[1], 2012). Thus, it can easily connect to geospatial libraries and scientific models, but also with the common cartographic engines and spatial databases. ZOO Kernel (ZOO-Project core engine) is written in C language, and Web Services can be programmed in C, Python, Java, Fortran, PHP and JavaScript. This multi-language support is convenient to develop new Web Services but also to use existing code to create. Open Source GIS libraries or specific code (spatial based or not) can so be ported server-side with minor modifications.

These capabilities were thus extensively used, in order to build Web Services able to identify, store, process and publish GIS data. The GDAL/OGR algorithms were first ported as WPS, allowing the support of numerous vector and raster GIS formats into MapMint. It was also very useful to identify and store information about data sources such as formats, projections, type (point, lines, polygons, matrix), size and other parameters.

Once the system was able to manage common GIS data sources, some other functionalities were developed in order to enable them as WMS, WMTS or WFS automatically. This was achieved using a collection of WPS services able to automate the the web maps configuration, using a tight coupling of ZOO-Project and MapServer.

2.2 Everything is a process

MapMint is thus built on top of several open source software, which can run together through the use of a collection of WPS services, mainly written in Python and JavaScript (Fenoy and al[2], 2012). The latter are organized into several directories, according to the main capabilities of the system, as listed bellow:

- List and identify GIS data (using the *ogrinfo* and *gdalinfo* algorithms as WPS)
- Convert and (re)project GIS data (using the *ogr2ogr* and *gdal_translate* algorithms)
- Add and style layers to prepare webmapping projects (writing MapServer mapfiles)
- Edit and configure data layers (using WFS-T and writing MapServer mapfiles)
- Save configurations into a project (writing MapServer mapfile and store them)
- Configure the client map (saving HTML, CSS and OpenLayers parameters)
- Publish the final application (merging a mapfile and a preconfigured HTML template)

All these actions are performed through WPS requests, executed by ZOO-Project, and allowing to execute both geospatial and non geospatial operation in a standardized way. Geospatial output are thus automatically passed to MapServer which automatically publish them trough WMS and WFS. The MapServer MapCache support is also automatically configured to create pyramids of tiles to publish data as WMTS when needed.

Figure 1. Simplified scheme of the MapMint architecture

The systematic use of WPS requests thus allows to interact with other software and to configure and drive the publication of GIS data in a simple way. MapMint automates every step of the web maps creation, and allows the GIS user to configure and publish data without any coding and using the Internet browser only.

2.2 Several steps to store, configure and publish web-mapping applications

MapMint is made of four interconnected Web modules which are briefly presented in this section. Their use allows to publish web maps from raw GIS data, through the several steps listed below.

The Dashboard is the first MapMint module which is opened by default once connected. It provides a configuration interface for WPS, WMS and WFS servers, simple server statistics, as well as quick links to the maps that were previously created using the Manager module.

The Distiller module provides an intuitive interface for GIS data management, allowing to import, list and organize GIS data into MapMint. It allows the creation of Data Stores where data can be stored. It can also connect to PostGIS databases, allowing to use tables as any other OGR data source. Once data sources are listed successfully, they are then already published as WMS/WFS, thanks to a mapfile which is written for every created DataStore, or for every active spatial database connection.

The Manager is the third module of the MapMint application. It provides a web-mapping interface to edit, style and process data layers which are created from the data sources added to the Distiller module. It is composed of a data layers panel on the left side and a map viewer panel on the right side, as shown in the figire bellow. Layers can be added, styled, and organized manually (colors, symbols, layers order, layers properties...). The map (i.e the project's mapfile) can be saved and updated at anytime.

The Publisher module finally provides an other Web interface for configuring the final web map that will be published. Using the project previously saved in the Manager, it allows more configuration options. The HTML layout and CSS theme can thus be selected, and various options can be added, such as map title, copyright, author and keywords (metatags).
The data layers can be set as vector (WFS), raster(WMS) or tiles(WMTS), according to the user needs and the size of the datasource. Several third-party base layers such as OpenStreetMap can also be added to the map. The map tools can finally be selected, in order to activate navigation, selection, query and spatial operations functionalities in the final map.

Once the previous steps are executed and every needed parameters are selected, the final web-mapping application is published once again using several WPS request, and the map is available online.

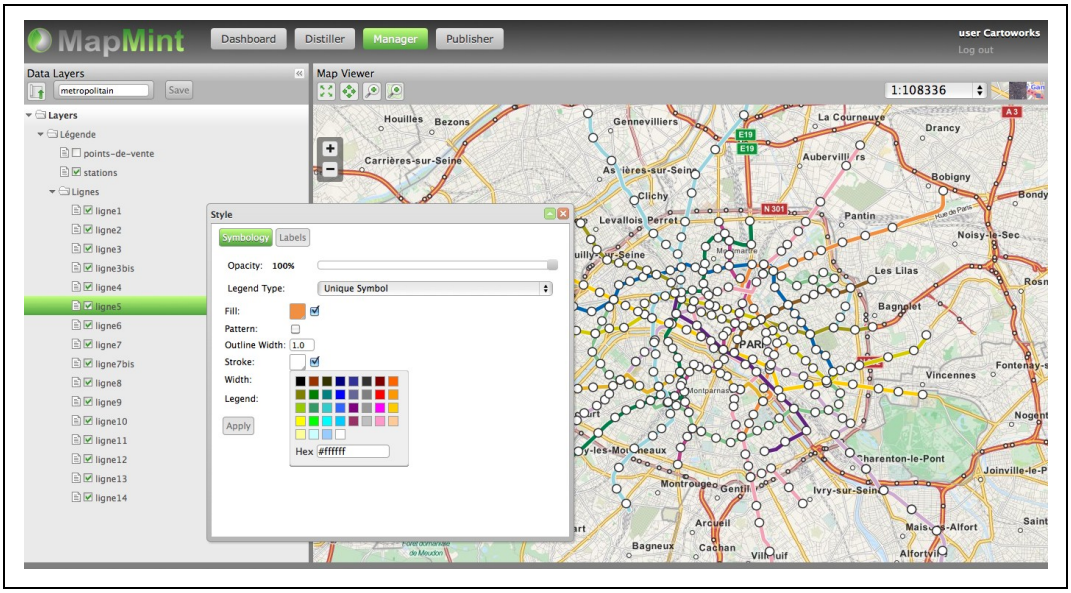

Figure 2. The MapMint Manager module interface

3. EXAMPLE WEBGIS CLIENT APPLICATIONS

The MapMint administration interface let the user publish web-mapping applications using the different modules. These web maps are built on top of the OpenLayers and Jquery Javascript libraries, according to the parameters provided by the user. The project map is rendered using the OpenLayers abilities to display GIS data both vector, WMS and WMTS layers. Both the layers styles and properties and the general map settings defined previously are taken into account.

The two client maps presented below were generated using MapMint, using OpenStreetMap data. The latter was published with the Distiller, added to the Manager and styled and finally made available online with the Publisher module.

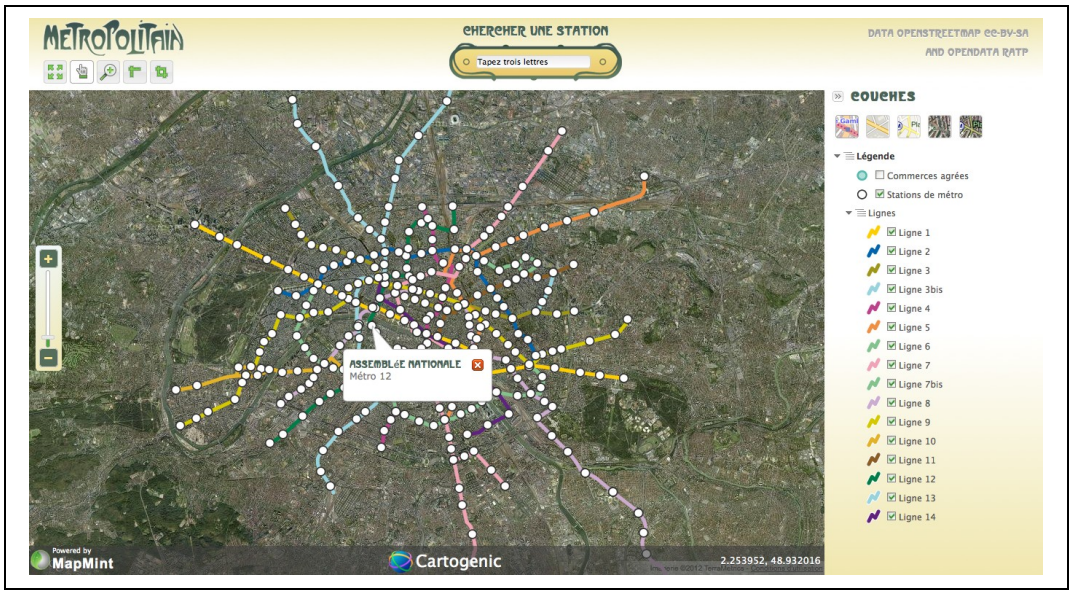

Figure 3. Example of a resulting web map application (1)

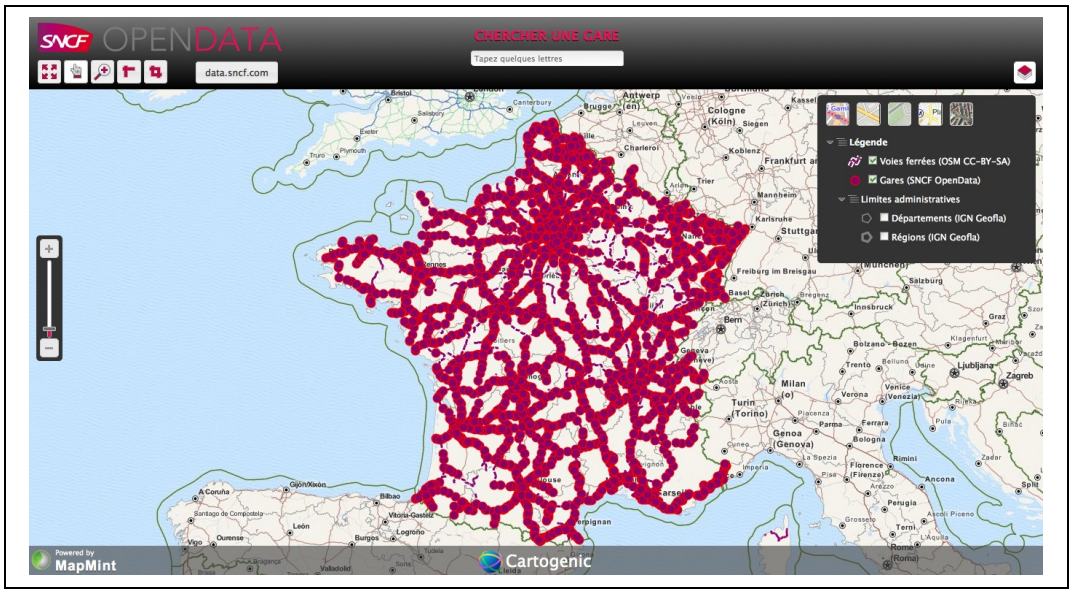

Figure 4. Example of a resulting web map application (2)

4. CONCLUSION

A service-oriented approach for building a geospatial platform in the cloud was presented in this paper. Based on the major OGC like WPS, WMS, WMTS and WFS, the MapMint software gathers and tightly couples several open source software into a single Web application. It allows the GIS user to publish data online and to configure modern web-mapping applications, using a complete web administration interface.

A set of geospatial functionalities were made available through the development of a collection of WPS services. The latter provides basic tasks for managing and publishing GIS data, but also more complex operations such as spatial operations and queries or data filtering and export. Current research aims to implement external environmental models into MapMint, for digital terrain models processing, water-shade and flow modeling.

6. REFERENCES

Fenoy G., Bozon N. and Raghavan V., 2012. *ZOO-Project, the Open WPS platform*. Applied Geomatics, Springer, Vol.4

Fenoy G., Bozon N. and Ueda N., 2012. *A commercial open source solution for managing geospatial content ans services* . Geoinformatics conference 2012 proceedings.