Towards Open Big Geospatial Data for geodata.gov.gr

Angelos Tzotsos, Michail Alexakis, Spiros Athanasiou, Yiannis Kouvaras

Institute for the Management of Information Systems / "Athena" Research and Innovation Center

Abstract

Open data provided by the public sector constitute a significant opportunity for growth. PublicaMundi (http://publicamundi.eu) is an EU FP7-ICT project aiming to make open geospatial data easier to discover, reuse, and share by fully supporting their complete publishing lifecycle in open data catalogues. To achieve this, PublicaMundi extends and integrates leading open source software for open data publishing and geospatial data management. During the first year of the project, the main focus of the development was on extending the CKAN project with new geospatial capabilities through integration with leading OSGeo software, such as pycsw, rasdaman, ZOO-Project, GeoServer and GDAL. CKAN was extended to support various established metadata models (including ISO-19115 and INSPIRE schemata), through integration with OWSLib and pycsw. Additionally, CKAN has been successfully integrated with the rasdaman raster data engine, adding big data capabilities to the PublicaMundi software stack, providing the ability to serve hundreds of TBs of Earth Observation data through the WCS OGC API or through PublicaMundi's Data API. A first integrated prototype is already available on labs.geodata.gov.gr, providing beta access to data publishers and developers for Greek open geospatial data.

Keywords

PublicaMundi, CKAN, pycsw, Open Data Catalogues, OGC

1 Introduction

Open data provided by the public sector constitute a significant opportunity for growth. Geospatial data account for an estimated 80% of public sector information and are the most significant category of open data due to their high production, procurement, and update costs, as well as their relevance in multiple domains. Despite their importance, they are increasingly difficult to reuse, especially in a cross-boundary multilingual context. The vast majority of open data catalogues in the EU have limited support for geospatial information with insufficient capabilities in publishing methodologies and tools, limited technical foundations to support value added services, and simplistic non-scalable support for geospatial data visualization. PublicaMundi (http://publicamundi.eu) is an EU FP7-ICT project aiming to make open geospatial data easier to discover, reuse, and share by fully supporting
their complete publishing lifecycle in open data catalogues. To achieve this, PublicaMundi extends and integrates leading open source software for open data publishing and geospatial data management. In particular, PublicaMundi extends CKAN, the leading open data catalogue, into treating geospatial data as “first-class citizens” and providing automatic OCG and INSPIRE access to geospatial data.

A first integrated prototype is already available on labs.geodata.gov.gr, providing beta access to data publishers and developers for Greek open geospatial data. In the next few months, PublicaMundi is being deployed to geodata.gov.gr to serve as the main open geospatial data catalogue of the Greek government.

This paper is structured as follows: in Section 2, the system architecture is briefly described. The integration environment of the PublicaMundi project is presented in Section 3, while geospatial extensions of CKAN and open data publishing workflow are described in Section 4. In Section 5 the support for big geospatial data is presented. In Section 6, the processing engine is shown and Section 7 describes future work.

2 System Architecture

This chapter describes the System Architecture of PublicaMundi, which consists of several application levels. PublicaMundi project was designed based on seven application levels, from the high level client applications to low level data storage components. The system follows a multi-tier, multi-level and multi-user approach aiming to provide tools for all stages of the open data lifecycle.

The goal of this multi-layer architecture was to address specific deployment needs (e.g. different map servers, deploy other CKAN extensions if needed, deploy on top of a database cluster, or change a specific component of the system without affecting the rest of the components). At the same time, another goal was to make the system architecture extensible for future improvements. The system was envisioned to be built on many loosely coupled layers in order to be able to scale and be deployed in a modular way, according to the end-user needs.

More specifically, the application levels are:

- **Data Storage.** This layer includes all the storage units, database clusters and cloud storage infrastructures that are used or developed in the PublicaMundi system. Currently, this layer of the system is based on PostgreSQL/PostGIS, rasdaman (Baumann, 2014) and cloud storage provided by Synnefo cloud stack (Koukis et al 2013).

- **Data Processing.** This layer includes all the spatial software that is responsible for transforming and processing spatial data (vector, raster and their metadata) before the core CKAN application publishes them on the web. Currently, this layer of the system is based on GDAL, PROJ, OWSLib and ZOO WPS.

- **CKAN (core).** The core application is based on the widely used open data catalogue CKAN. CKAN is based on the Pylons Web Framework, which follows the Model, View, Controller approach (MVC). This approach
has the advantage of loose coupling of resources and software components, which leads to great re-usability of the source code and better management of the stack. This layer of the system is based on CKAN application in stable version 2.2.

- Application Modules. This layer consists of CKAN extensions that were developed in order to spatially extend CKAN but also to provide new functionality such as multilinguality. Currently, this layer of the system is based on pycsw, ckanext-schematic, ckanext-publicamundi, ckanext-harvest and ckanext-spatial. The PublicaMundi extension also includes modules to enhance scalability of the application (caching, proxy and analytics).
- Web Services. This layer consists of several geospatial servers that are integrated with CKAN providing geospatial data support for the web. These servers are compatible with OGC and come from the OSGeo stack. Currently, this layer of the system is based on GeoServer, rasdaman, ZOO WPS and pycsw. Support for MapServer and MapProxy is being implemented.
- Developer APIs. This layer consists of all the development tools and APIs that are provided to developers in order to: create maps, re-use data, process data and publish/search/harvest metadata. On top of CKAN API and several OGC APIs, PublicaMundi provides its users with two new APIs: the Mapping API and the Data API. The Mapping API is based on integration of OpenLayers and Leaflet. The Data API is based on PostGIS and ZOO-WPS.
- Client Applications. This layer consists of external applications that use PublicaMundi’s APIs in order to interact with open data. Currently, this layer of the system is based on QGIS, CKAN harvesters and other CSW clients.

Figure 1 provides a detailed overview of the software stack used to implement the system.

3 Integration environment

For the purposes of the system development and the first deployment to labs.geodata.gov.gr, an integration environment was established. On this environment, beta testing of the software takes place and the development team gathers feedback from the early users/testers of the system. For this environment, a set of high-end servers was acquired and the software components were deployed. Also, new versions of the software components are periodically integrated from the project’s git repository and are available for evaluation. Labs.geodata.gov.gr was initialized with real-world geospatial datasets from geodata.gov.gr (geospatial datasets plus metadata) to enable user to experiment with all implemented data publishing/reuse services.

The integration environment of PublicaMundi is deployed on top of the Synnefo cloud stack (Koukis et al 2013), within a number of virtual machines. Synnefo is a complete open source cloud stack written in Python that provides Compute, Network, Image, Volume and Storage services, similar to the ones offered by Amazon Web Services (AWS).
Synnefo manages multiple Ganeti clusters at the backend for handling of low-level VM operations and uses Archipelago to unify cloud storage. To boost 3rd-party compatibility, Synnefo exposes the OpenStack APIs to users. Synnefo keeps a clear separation between the traditional cluster management layer and the cloud layer. This unique design approach leads to a completely layered architecture. The layered approach for both the Compute and the Storage service boosts production readiness, maintainability and upgradability (Koukis et al 2013). The modular design allows for linear scalability, gradual extensibility and ease of operations. In Figure 2, an overview of the Synnefo architecture is presented, showing the major components on each layer.

For the deployment of PublicaMundi, the Synnefo stack was installed on the available server infrastructure, and a Web User Interface (UI) was made available to the administrators in order to maintain and manage the cloud resources for the integration environment. In Figure 3, the administration page of the Synnefo UI is shown. Through that interface, the user can have an overview of the virtual machines available, can restart, shut down, start up, delete, and create a virtual machine. For the deployment of labs.geodata.gov.gr several virtual machines were created, using the Debian 7 GNU/Linux operating system. This strategic decision was based on the maturity and stability of this operating system, and the fact that it is used in production servers on geodata.gov.gr since 2010.
Figure 2: A detailed description of the Synnefo Architecture.

Figure 3: The main interface web page of Synnefo for administration of the cloud resources.

The software components of the system were deployed into 8 virtual clusters, with the provision of spinning up more virtual machines into each cluster if
necessary. In Figure 4, the catalogue page of the integrated environment is presented.

![Figure 4: The catalogue front page in the integrated environment.](image)

### 4 Geospatial extensions to CKAN catalogue

As described previously, CKAN is a Pylons/Python application, providing a mechanism to add functionality through extensions. This way, the core application is improved but at the same time the source code is easier to be maintained through mainstream code revisions (releases).

A main objective of the PublicaMundi project was to provide reusable and extensible software, while spatially enabling CKAN. Throughout the development process there was a focus on maintaining compatibility with upstream CKAN project, so that the new extensions would easily be applicable not only to geodata.gov.gr open data portal but also on other CKAN deployments, without breaking existing functionality.

During the development of the CKAN extensions significant progress was made in the metadata components, with pycsw (OGC CSW 2.0.2 reference implementation) being tightly integrated into CKAN. Initially, pycsw was loosely integrated under the CKAN spatial extension and used in the US data.gov project. The integration of pycsw into CKAN brought harvesting support of metadata from OGC services as well as support for INSPIRE and OGC OpenSearch Geo/Time specifications.

Moreover, CKAN was extended to support various established metadata models (including ISO-19115, ISO-19139 and INSPIRE), through OWSLib and pycsw. A new metadata extension was implemented (ckanext-schematic), providing the
capability for developing new metadata schema support in CKAN, through Pythonic Zope interfaces. This new extension also provides a way to easily create metadata editor user interfaces and extend the current CKAN publishing workflow. For geodata.gov.gr, the publishing workflow has been heavily extended, adding capabilities of automatic ingestion of geospatial vector data to a spatial database (PostGIS) and automatic publication of OGC services from within the CKAN administrative user interface (through GeoServer). Furthermore, CKAN was extended to support raster data as well (through rasdaman).

Data publishers are guided step-by-step into creating metadata in their schema of choice (Figures 5,6) or importing existing metadata. The metadata provided can then be transformed on-the-fly in any supported schema. Additionally, if an existing data catalogue or Spatial Data Infrastructure is available, the data publisher only needs to provide a simple entry point, and all available metadata are automatically harvested.

The new open data publishing workflow significantly lowers the entry barrier for data publishers, while also accommodating the data publishing needs of organizations with existing data catalogues and SDIs, maintaining full compatibility with INSPIRE and OGC web services.

For the management of data resources, a new administrator dashboard was implemented, which adds capabilities of creating new derivative resources (e.g. create a WMS resource from a GML file) (Figure 7).

![Figure 5: The publishing workflow to create a new INSPIRE dataset.](image-url)
Figure 6: The PublicaMundi metadata editor (INSPIRE schema).

Figure 7: The PublicaMundi administration dashboard.
CKAN was extended to natively support geospatial vector data management, by integrating PostGIS, the leading open source geospatial database. Data publishers can upload geospatial data in any format and coordinate reference system. The system automatically stores the dataset and can provide it in another data format (on-demand) or through OGC compatible services. As such, data publishers can provide any data they have at hand, without additional effort into transforming their data in specific-purpose formats. Furthermore, as soon as the data is published, they are automatically available for querying and visualization with no extra effort through a Mapping API and a Data API (Figures 8 and 9).

As part of the CKAN spatial extension, the developed Mapping API was integrated in CKAN, thus providing data preview for geospatial data (Figure 8). Also, in order to provide similar functionality as previous geodata.gov.gr, a full Javascript map client application was developed in CKAN with data search and map composition capabilities. (Figure 10).

**Figure 8: Geospatial data previewer based on PublicaMundi Mapping API.**
Figure 9: Demonstration of PublicaMundi’s Data API.

Figure 10: PublicaMundi’s Mapping application.
5 Big Data support
CKAN has been successfully integrated with the rasdaman raster data engine (through a CKAN extension named raster-storer), adding big data capabilities to the PublicaMundi's software stack, providing the ability to serve hundreds of TBs of Earth Observation data through the WCS OGC API or through PublicaMundi's Data API.

The rasdaman system is a scalable, multi-dimensional array database engine and analytics server. Rasdaman is a domain-neutral Array Database System: it extends standard relational database systems with the ability to store and retrieve multi-dimensional raster data (modeled as arrays) of unlimited size through an SQL-style query language. Array intensive services can be set up using rasdaman, such as those found in data centers offering remote sensing, sensor series, image, simulation, and statistics data. Domains taking advantage of an Array Database can be Earth, Space, and Life sciences. The query language offered over arrays enables flexibility, speed, and scalability.

On top of the Array Database System, a geospatially enabled layer is provided that adds spatiotemporal semantics over array data. Within the geo layer fall the petascope components and the related geo-import tools (currently known as rasgeo and wms-import). Geo-enablement of rasdaman provides access to geo raster standards-based services, including OGC WCS and WCPS, the OGC raster query language, along with WCS-T, and WPS. A key advantage of rasdaman is its internal management of data based on n-D tiling which allows data to be stored in the most convenient way, depending on the desired performance profile along properties, such as fast retrieval along some dimensions and fast data ingestion rates.

The PublicaMundi project is based on CKAN as a publishing platform that streamlines the process of publishing, sharing, finding and using data. Several CKAN extensions were developed to allow for ingestion of georeferenced data, be it vector or raster. For raster and earth observation data in particular, a raster storer CKAN extension was developed, in order for CKAN to be able to automatically ingest raster data and expose them through a variety of OGC services. In this context, a WCS-T standard was also implemented to allow simple ingestion of raster data into rasdaman.

The raster storer plugin receives datasets (or links to datasets) from the user, containing data in various formats (GeoTiff, JPEG2000, zip archives etc) and, using the GDAL library, reads or deduces all the necessary metadata directly from the file and creates a GML file that references the original file as the source of the raw data. This provides an important degree of flexibility. Rasdaman is loosely coupled with CKAN, allowing the use of this ingestion method in other projects that will not use CKAN as the front-end. Data is never copied or moved around, the GML file only references the original file, allowing rasdaman to read its contents and ingest it in its internal format.

6 Processing engine
An important part of the PublicaMundi system, is the ability to re-use the catalogue's open data and create new datasets with added value. To this purpose, a geospatial processing engine was implemented as part of the
system. The open source WPS implementation ZOO-Project was used at different levels of the PublicaMundi project, especially to address the goal of providing a reliable and scalable processing engine to PublicaMundi architecture, as well as a collection of generic and user-centric WPS services. According to the OGC, the WPS standard provides a set of rules for standardizing inputs and outputs (requests and responses) of geospatial processing services. The standard defines how a client can request the execution of a process, and how the output from the process is handled. It defines an interface that facilitates the publishing of geospatial processes and clients' discovery of and binding to those processes. WPS thus allows to process geospatial data over the Internet and ZOO-Project implements it through an open architecture and a modular source code mostly written in C, Python and JavaScript.

In the presented framework, a WPS extension of the CKAN software was implemented in order to provide a bridge between the Data Catalog and the Processing Engine. The so-called ckanext-wps module brings WPS capabilities to the CKAN powered open data catalogues by introducing the possibility of adding an existing WPS server as a new CKAN resource from the CKAN administration panel. The support for WPS GetCapabilities, DescribeProcess and Execute requests support was implemented using the Python language and the CKAN logic in order to create a suitable extension. The Processing Engine can thus be directly exploited from CKAN and this allows to directly process open data and create, store, and reference new information directly from the Catalog. Every WPS service returned by a GetCapabilities request can be configured through appropriate CKAN forms and executed using any suitable geospatial resource available from the raster and vector data stores as input data. The output data finally generates a new CKAN resource and add it to the targeted data store. Further improvements are still under development and their release is already planned for the next phase of the project.

6 Future work

PublicaMundi project is still under heavy development. Future work includes multilingual support for both data and metadata storers, as well as utilities to enable publishers to crowdsouce their translations of metadata and data. At the same time, work is being done to implement the upcoming OGC CSW 3.0 and WPS 2.0 standards, as well as the DCAT Application Profile developed by the European Commission. Furthermore, work is being done for data interlinking and integration of Data and Mapping API with WPS and WCPS.

References