GeoNode: an open source framework to build spatial data infrastructures

Paolo Corti[1], Francesco Bartoli[2], Alessio Fabiani[3], Cristiano Giovando[4], Athanasios Tom Kralidis[1], and Angelos Tzotsos[1]

[1]Open Source Geospatial Foundation, Beaverton, OR, USA
[2]Geobeyond SRL, Rome RM, Italy
[3]GeoSolutions SAS., Massarosa LU, Italy
[4]World Bank, Washington DC, USA

Corresponding author:
First Author[1]

Email address: pcorti@gmail.com

ABSTRACT

GeoNode is an open source framework designed to build geospatial content management systems (GeoCMS) and spatial data infrastructure (SDI) nodes. Its development was initiated by the Global Facility for Disaster Reduction and Recovery (GFDRR) in 2009 and adopted by a large number of organizations in the following years. Using an open source stack based on mature and robust frameworks and software like Django, OpenLayers, PostGIS, GeoServer and pycsw, an organization can build on top of GeoNode its SDI or geospatial open data portal. GeoNode provides a large number of user friendly capabilities, broad interoperability using Open Geospatial Consortium (OGC) standards, and a powerful authentication/authorization mechanism. Supported by a vast, diverse and global open source community, GeoNode is an official project of the Open Source Geospatial Foundation (OSGeo).

INTRODUCTION

GeoNode [1] is a web framework, based on an open source software technology stack, which enables organizations to build and deploy geospatial content management systems (GeoCMS), public spatial data infrastructures (SDI) (Infrastructures, 2004; Kralidis, 2009) and open geospatial data catalogues. The Global Facility for Disaster Reduction and Recovery (GFDRR) [2] started GeoNode development in 2009 and since then the project was joined by a large number of organizations, universities and research centers [3].

GeoNode is a web application which allows users to share geospatial datasets and combine them in thematic web maps. All of the datasets are accessible from clients using Open Geospatial Consortium (OGC) [4] standards such as Web Map Service (WMS) [5], Web Feature Service (WFS) [6], Web Coverage Service (WCS) [7], Catalog Service for the Web (CSW) [8].

CAPABILITIES

GeoNode lets users share spatial datasets in a variety of vector and raster formats. By default, users can upload shapefiles (vector) and GeoTIFFs (raster) from the user interface, but it is possible to extend the application to use any GDAL [9] supported format.

[1]http://geonode.org/
[2]https://www.gfdrr.org
[3]https://opendri.org/resource/opendri-geonode-a-case-study-for-institutional-investments-in-open-source/
[4]https://www.opengeospatial.org/
[5]https://www.opengeospatial.org/standards/wms
[6]https://www.opengeospatial.org/standards/wfs
[7]https://www.opengeospatial.org/standards/wcs
[8]https://www.opengeospatial.org/standards/cat
[9]https://www.gdal.org/

Once a geospatial dataset or layer is uploaded and registered in the GeoNode catalogue, the framework provides users of the system with a number of possibilities:

- The layer owner, or another user with appropriate permissions, can edit layer metadata. These metadata are automatically exposed by an OGC compliant CSW catalogue to provide search/discovery capability
- Users with appropriate permissions can add the layer to a map and combine it with other layers to create thematic maps which can be accessed by the general public
- Users with appropriate permissions can edit styles for a specific layer and its feature geometries (when the layer is of vector type)
- The owner of a layer can assign permissions for the given layer to a number of users or groups. There are different permissions which let the dataset owner to granularly set specific permissions: who can view the data and its metadata, who can download it, who can change its styles and its features, who can change its metadata
- Each geospatial dataset can support different OGC standards: WMS, WMS-C, WFS, WFS-T, WCS, CSW, as well as mass market search standards OAI-PMH [10], SRU [11], and OpenSearch [12]. This way an external application, for example a desktop tool like QGIS [13], can use the layer in a map canvas, and possibly change the feature geometries

ARCHITECTURE

GeoNode is a web application developed in Django [14], a popular Python web framework, which orchestrates a spatial data server (based on GeoServer [15], the default option, or QGIS Server), a user interface, a spatial cache data server (based on GeoWebCache [16]), a spatial database (based on PostgreSQL [17] and PostGIS [18]), a catalogue (based on pycsw [19], the default option, or GeoNetwork OpenSource [20] or deegree [21]). Optionally it is possible to add in the stack a search engine (such as Solr [22] or Elasticsearch [23]) to implement a very fast search API and Celery[24]/RabbitMQ [25], a tasks queue where it is possible to run asynchronously time consuming tasks, such as thumbnails generation, remote map services harvesting and other operations which would slow down the ideal fast client/server HTTP request/response interaction.

Geospatial datasets are uploaded from the user's browser or registered by an administrator in the spatial data server using a GeoNode administrative command. Vector datasets are stored in the spatial database, while raster datasets are stored in the file system. Map tiles are returned to the client using standards like WMS-C or WMS. Vector datasets can be returned and edited from the client using standards like WFS and WFS-T. The catalogue enables metadata for being consumed by CSW clients, supporting a publish/find/bind design pattern of information search/retrieval.

Components

In the following paragraphs a more detailed description of each component is provided: web application, user interface, spatial data server, spatial cache data server, spatial database, catalogue, search engine and tasks queue (Figure 1).

[10]https://www.openarchives.org/pmh/
[11]http://www.loc.gov/standards/sru/recordUpdate/
[12]http://www.opensearch.org
[13]https://www.qgis.org
[14]https://www.djangoproject.com/
[15]http://geoserver.org/
[16]https://docs.geoserver.org/latest/en/user/geowebcache/index.html
[17]https://www.postgresql.org/
[18]https://postgis.net/
[19]https://pycsw.org/
[20]https://geonetwork-opensource.org/
[21]https://www.deegree.org/
[22]http://lucene.apache.org/solr/
[23]https://www.elastic.co/
[24]http://www.celeryproject.org/
[25]https://www.rabbitmq.com/

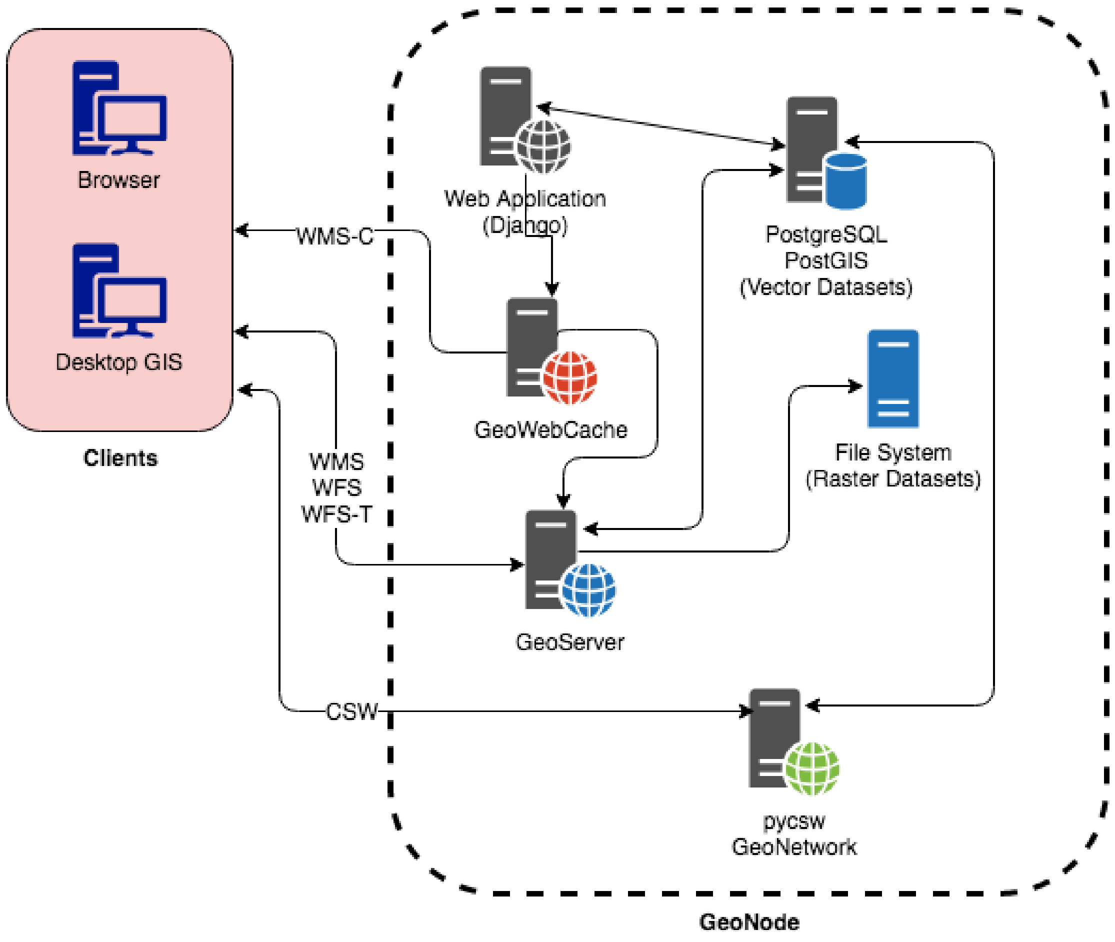

Figure 1. GeoNode Architecture.

Web Application

The GeoNode web application is the central component of the framework and it is the orchestrator of the client interaction with the other components of the system. It is developed in Django, a Python web framework, and it uses a relational database, most typically based on PostgreSQL or SQLite [26] - but it can use any RDBMS supported by Django. Figure 2 shows a simplified version of the entity relationship model of the GeoNode database.

The core information model in GeoNode is the **ResourceBase** which provides a baseline set of core properties such as a universal unique identifier (uuid), a bounding box, a title, other metadata properties, publishing workflow statuses. Relevant resources (layers, maps, documents) inherit from this model in order to manage common properties in a consistent fashion.

GeoNode core entities are layers and maps (in a few instances there can be also documents but for the sake of simplicity we will leave them out in this paper). It is possible to assign one **category** to each layer or map, a set of geographic **regions** (countries, continents) and a number of **tags**.

A **Layer** represents a spatial dataset in the system - which can be of vector or raster type or from a remote internet service. Each layer can be associated to different **styles**, and - in the case of vector datasets, can be composed of different layer **attributes**.

A **Map** represents a GeoNode web map and can be composed of different map layers. Each layer can be internal to GeoNode, or a base map from some different organizations or providers like OpenStreetMap

[26]https://www.sqlite.org

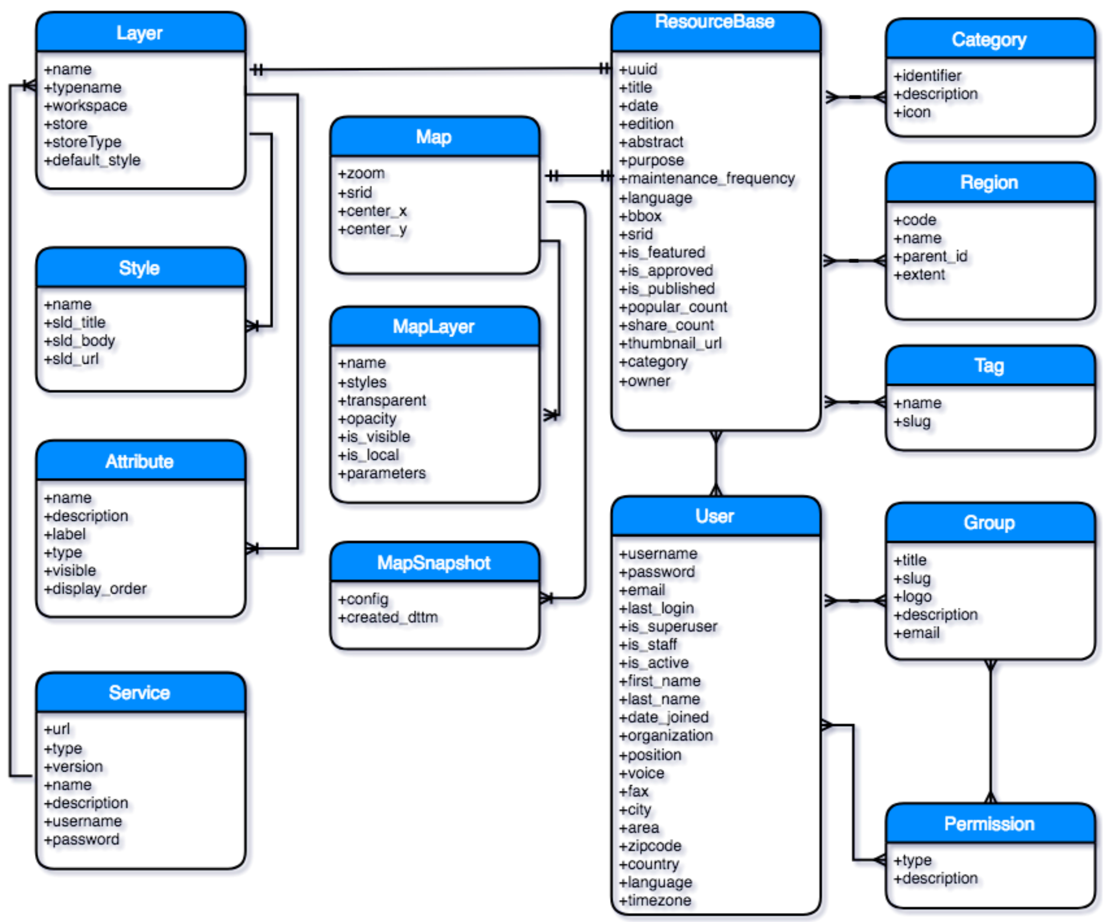

Figure 2. GeoNode database entity relationship model.

[27], Google Maps [28], Microsoft Bing Maps [29], Esri ArcGIS server [30] or MapBox [31]. It is also possible to register in GeoNode external map services, using OGC standards or Esri protocols [32], and have remote layers. For each layer in a given map GeoNode tracks in its database the name of the layer and properties like the visibility, opacity and styles. GeoNode also tracks different versions of a specific map, therefore the user will be able to return to a previously saved version (snapshot) of the same map.

The web application can be run using a web server such as nginx [33] or Apache httpd [34] and a Python engine like uWSGI [35] or Gunicorn [36].

User Interface

GeoNode uses Django templates and a number of JavaScript libraries and frameworks in order to provide the user interface to the clients. Most notably, GeoNode uses AngularJS [37] to let the users to access the Application Programming Interface (API) - which is based on Django Tastypie [38]. The

[27] https://www.openstreetmap.org
[28] https://www.google.com/maps
[29] https://www.bing.com/maps
[30] https://www.arcgis.com
[31] https://www.mapbox.com/
[32] https://developers.arcgis.com/documentation/core-concepts/rest-api/
[33] https://www.nginx.com/
[34] https://httpd.apache.org
[35] https://uwsgi-docs.readthedocs.io
[36] https://gunicorn.org/
[37] https://angularjs.org/
[38] https://django-tastypie.readthedocs.io

jQuery [39] framework is also used widely in the context of the application, for example to provide words autocompletion in the search text boxes. The user interface includes a mapping client, which can be based on OpenLayers [40] or Leaflet [41]. At present, a number of mapping clients are available:

- GeoExplorer [42], based on the OpenLayers, GeoExt [43] and ExtJS [44] Javascript frameworks, which is the default and possibly the most common choice for GeoNode deployments
- MapLoom [45], which is built on top of Angular and OpenLayers. An interesting feature of MapLoom is its GeoGig [46] integration, which enables geospatial feature versioning
- MapStore [47], based on the React Javascript framework. It is not tied to a specific Javascript mapping library, but it can indifferently use OpenLayers, Leaflet or CesiumJS [48]
- WorldMap [49], developed by Harvard University Center of Geographic Analysis, which is a forked version of GeoExplorer, which adds to it features such a attribute based classifications style editor, layer categories in the table of content, a note editor and a gazetteer

Spatial Data Server

GeoNode includes a spatial data server which is used by the client running in the browser, or by an external GIS application, to render map tiles or stream/edit vector geometries using OGC standards such as WMS and WFS/WFS-T. By default GeoNode uses GeoServer as its internal spatial data server, and we will refer to this configuration for the following of the present article. It is possible, though, to use QGIS Server as an alternative to GeoServer.

GeoServer is a powerful Java web application, which can be run using a servlet engine such as Tomcat [50] or Jetty [51], which is developed on top of the GeoTools [52] Java geospatial library.

Spatial Cache Data Server

GeoNode uses GeoWebCache as a tiling server: its aim is to eliminate redundant request processing and save large amounts of processing time. Typically the GeoNode mapping client request tiles from the WMS-C GeoServer endpoint: if a given tile has already been generated and is not expired then it is returned from GeoWebCache to the mapping client. Otherwise it is generated by a WMS request to GeoServer.

Spatial Database

GeoNode supports both raster and vector geospatial datasets and store them respectively in the file system and in a spatial database, based on PostgreSQL/PostGIS. It is also possible for simple installations not to use the spatial database and store the vector layer on the file system as shapefiles. The spatial database provide a powerful multi user storage for the vector datasets, which let the database to be used also by different applications. For example it is possible to directly connect to the database with a GIS client to create more advanced cartographic representations, to edit data, to perform GIS analysis on the data and so on.

PostGIS adds the benefit to have a powerful SQL spatial syntax which let external users to run powerful and fast geospatial operations.

Catalogue

A catalogue is required in a SDI to provide an interoperability mechanism to discover data and metadata (Nogueras-Iso et al., 2005). The OGC CSW standard (Consortium, 2016) has been designed for this purpose: GeoNode implements it using pycsw, which comes bundled with the GeoNode installation, or

[39] https://django-tastypie.readthedocs.io
[40] https://openlayers.org/
[41] https://leafletjs.com/
[42] https://connect.boundlessgeo.com/docs/suite/4.6/geoexplorer/
[43] http://geoext.org/
[44] https://www.sencha.com/products/extjs/
[45] https://github.com/ROGUE-JCTD/MapLoom
[46] http://geogig.org/
[47] https://mapstore.geo-solutions.it
[48] https://cesiumjs.org/
[49] https://github.com/cga-harvard/geoexplorer-worldmap-client
[50] http://tomcat.apache.org/
[51] https://www.eclipse.org/jetty/
[52] http://docs.geotools.org/

alternatively GeoNetwork OpenSource or deegree. pycsw, the default GeoNode catalogue engine, uses the Django relational database tables to provide data discoverability to external clients using CSW requests such as GetCapabilities, DescribeRecord, GetRecordById and GetRecords. A typical example of its use is via the QGIS MetaSearch plugin [53], which enables the user to search GeoNode datasets from within QGIS. User discovered layers can be added to the map using the WMS/WMTS, WCS or WFS GeoServer endpoints.

Search Engine

Optionally GeoNode can use a search engine such as Solr or Elasticsearch, both based on Apache Lucene [54], to implement extremely fast and scalable search queries and enable searching capabilities such as keyword, temporal and space facets, text stemming and synonyms detection, scored results and many others.

GeoNode uses Haystack [55], a Django library which abstracts the search engine backend. By using Haystack and its flexible configuration parameters, some of the database models are indexed in the search engine. If the GeoNode models are changing often it is possible to reindex them on a regular basis by creating periodic tasks in the task queue.

Task Queue

GeoNode can optionally use a task queue, based on Celery and RabbitMQ, to run time demanding operations which are better handled asynchronously from the regular HTTP request/response cycle. Typical operations which can be sent to the task queue for asynchronous processing are layer registration in GeoServer, layer thumbnails generation, email notifications, remote services harvesting. It is also possible to use the task queue to schedule periodic tasks, for example to reindex the search engine for an instance of a layer or a map or to do cleanup operations on spatial datasets.

Security and authorizations

The GeoNode authorization mechanism provides a way to authorize users or group of users to access to the resources (layers or maps). For both layers and maps it is possible, using the GeoNode user interface, to enable specific users or group of users to view, download, change the metadata of a specific resource. For vector layers it is possible to enable users or groups to edit the features geometry and to change the style.

GeoNode implements security and authorizations at two levels: at the Django web application level is used the django-guardian [56] library and at the GeoServer level is used the GeoFence plugin [57]. django-guardian serialize on the Django relational database the possible authorization types for a user or group of users for a given resource. GeoFence, which is a module embedded in GeoServer, uses a relational database (typically H2 [58] or PostGIS) to store the permissions for a given user or group of users to a specific layer. OGC endpoints like WMS, WMS-C, WFS, and others will be enabled by GeoFence in GeoServer for each specific user or group of users.

Containers

GeoNode is developed with scalability in mind, and for this reasons the recommended installation method to test it out is through Docker [59] containers and the docker-compose [60] tool.

At the time of writing docker-compose uses separate services, one for each component of the GeoNode stack to deploy it into the following containers (Figure 3):

- nginx, on which the web server runs
- django, on which the GeoNode Django web process runs
- postgresql, on which PostgreSQL and its spatial extension, PostGIS are running
- geoserver, on which GeoServer runs

[53]https://docs.qgis.org/2.8/en/docs/user$_{m}$*anual/plugins/plugins*$_{m}$*etasearch.html*
[54]http://lucene.apache.org/
[55]https://django-haystack.readthedocs.io
[56]https://django-guardian.readthedocs.io
[57]https://docs.geoserver.org/latest/en/user/community/geofence-server/index.html
[58]http://www.h2database.com/html/main.html
[59]https://www.docker.com/
[60]https://docs.docker.com/compose/

- celery, on which Celery runs
- rabbitmq, on which RabbitMQ runs
- elasticsearch, optional, where the search engine runs

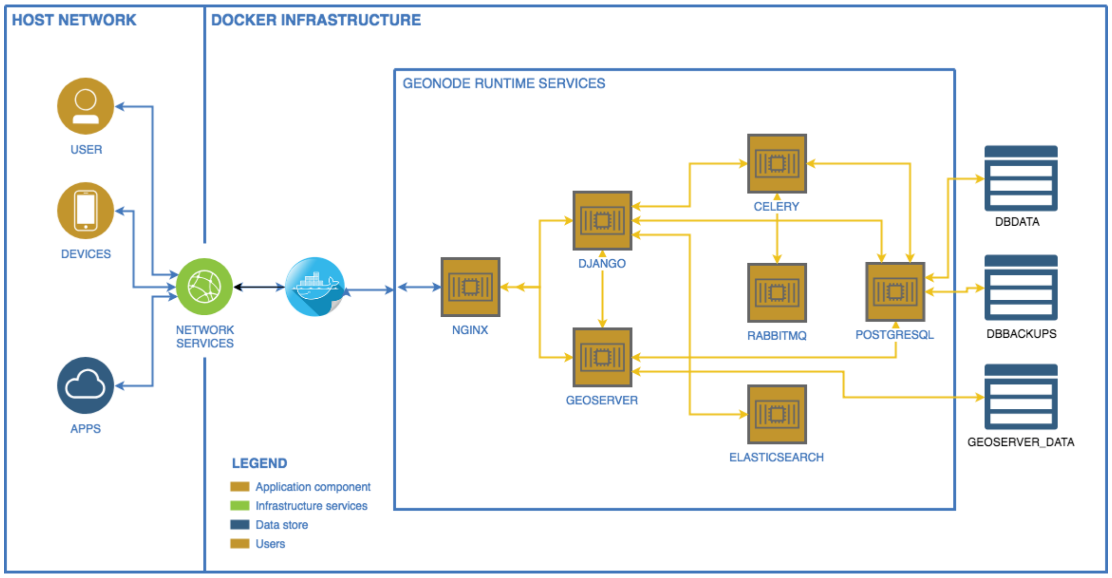

Figure 3. GeoNode containers architecture.

This containers architecture is also used by the GeoNode Continuous Integration (CI) infrastructure for each build of the system, which happens any time a pull request is created in GitHub.

As per deployment operations, the GeoNode community is discussing a possible production ready deployment procedure based on Docker and Kubernetes [61]. Kubernetes is emerging as the de facto container orchestration system and would make it possible to run GeoNode on a containerized architecture managed by a cloud service (IBM Kubernetes Service, Google Kubernetes Engine, Azure Kubernetes Service, etc) or rolled in house.

OPEN SOURCE COMMUNITY

GeoNode is an open source project, started in 2009 from World Bank and Boundless (at that time named OpenGeo) with pilot programs in El Salvador and Guatemala. GeoNode was initially focused on sharing data created by the Central America Probabilistic Risk Assessment (CAPRA) in order to assess and mitigate the risk due to adverse Natural Disasters . Since then more and more organizations, universities, research centers, institutes, private companies and numerous developers have joined the project. In the earlier adopters there are the United States Agency for International Development (USAID), the United Nations World Food Programme (WFP), the Harvard University, the GEM Foundation and ITHACA. At the time of writing GeoNode is used in hundreds of installations worldwide to power SDIs, web GIS and open data portals.

In almost 10 years there have been code contribution from more than 170 software developers. The two GeoNode public mailing lists, geonode-developers and geonode-users, experience significant daily traffic and a Project Steering Committee (PSC) composed by five persons has been constituted to provide high level guidance and coordination. GeoNode was incubated as an OSGeo project in 2016. Every year the GeoNode developers organize a GeoNode summit and code sprint aimed to finalize software releases. These events have been organized worldwide: in Italy, US, United Kingdom, Germany and many other countries. Code sprints were hosted at almost each yearly OSGeo FOSS4G international conference.

GeoNode is released with a GPL v3.0 license and it is entirely built using other open source libraries, software and framework. Here is a list of most notable libraries and projects being part of the GeoNode ecosystem:

[61] https://kubernetes.io/

- User Interface: AngularJS, jQuery, ExtJS, GeoExt, OpenLayers, Leaflet, MapStore
- Web Application: Django, Celery, Pillow, python-memcached, psycopg, PyYAML, Beautiful Soup, django-pagination, django-taggit, django-guardian, django-polymorphic, diango-tastypie, django-invitations, django-haystack, django-oauth-toolkit, python-oathlib, Requests, pyproj, OWSLib, Shapely, gsconfig, geopy, uWSGI, gunicorn
- OGC Services: GeoServer, QGIS Server, GeoWebCache, pycsw, GeoNetwork
- Databases: SQLite, PostgreSQL, PostGIS
- Other: Elasticsearch, Solr, RabbitMQ, Docker, docker-compose

SUCCESS STORIES

In this section we provide an incomplete list of successful implementations of GeoNode.

The Center for Geographic Analysis of Harvard University in 2009 started using GeoNode to develop and build its public SDI: Harvard WorldMap (Guan et al., 2012). WorldMap [62] is built to assist academic research and teaching as well as the general public and supports discovery, analysis, visualization, communication and archiving of multi-disciplinary, multi-source and multi-format data, organized spatially and temporally. Since its inception more than 20,000 thousand users contributed geospatial datasets and published web maps in WorldMap. Currently on the system there are almost 30,000 different geospatial layers and more than 5,000 maps. Notable maps and datasets includes the metro area of Boston (Figure 4)., East Asia, Africa, Vermont and the city of Paris. These web maps are used in university classes as well as by individual researchers.

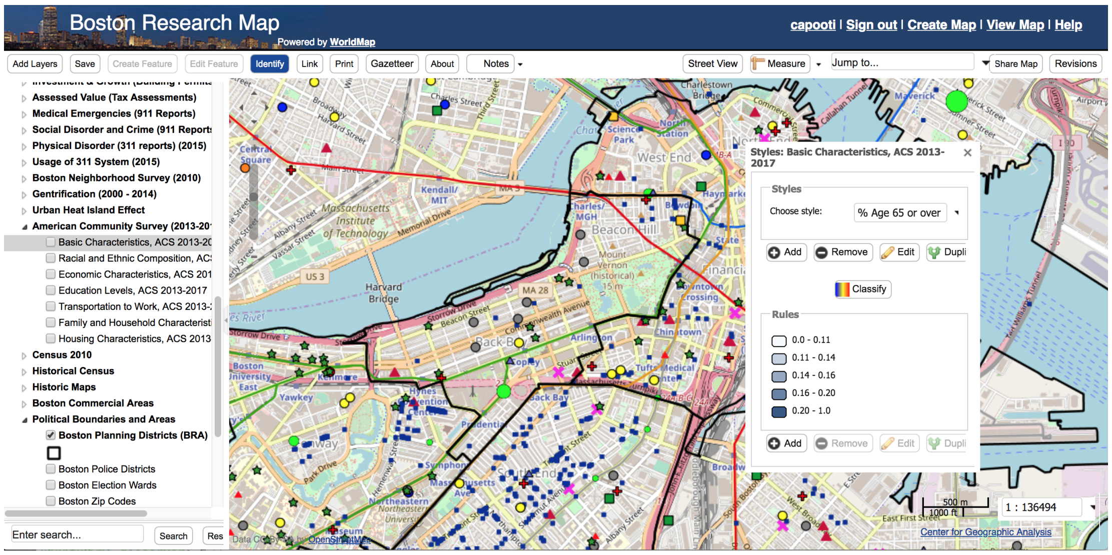
Boston Research Map
Powered by WorldMap
capooti | Sign out | Create Map | View Map | Help
Add Layers
Save
Create Feature
Edit Feature
Identify
Link
Print
Gazetteer
About
Notes
Street View
Measure
Jump to...
Share Map
Revisions
Assessed Value (Tax Assessments)
Medical Emergencies (911 Reports)
Social Disorder and Crime (911 Reports)
Physical Disorder (311 reports) (2015)
Usage of 311 System (2015)
Boston Neighborhood Survey (2010)
Gentrification (2000 - 2014)
Urban Heat Island Effect
American Community Survey (2013-2017)
Basic Characteristics, ACS 2013-2017
Racial and Ethnic Composition, ACS
Economic Characteristics, ACS 2013
Education Levels, ACS 2013-2017
Transportation to Work, ACS 2013-
Family and Household Characteristics
Housing Characteristics, ACS 2013
Census 2010
Historical Census
Historic Maps
Boston Commercial Areas
Political Boundaries and Areas
Boston Planning Districts (BRA)
Boston Police Districts
Boston Election Wards
Boston Zip Codes
Enter search...
Search
Styles: Basic Characteristics, ACS 2013-2017
Styles
Choose style:
% Age 65 or over
Add
Remove
Edit
Dupli
Classify
Rules
0.0 - 0.11
0.11 - 0.14
0.14 - 0.16
0.16 - 0.20
0.20 - 1.0
Harvard Bridge
Beacon Hill
Back Bay
Massachusetts Institute of Technology
1 : 136494
Center for Geographic Analysis

Figure 4. Boston Research Map in Harvard WorldMap.

The United Nations World Food Programme (WFP) has started building its public SDI using GeoNode [63] in 2012. The WFP catalogue contains more than 1,000 public and private layers, contributed by more than 300 users which created more than 100 maps. The platform has been adopted as the institutional public web mapping platform, and it contains datasets related to environment, food security, natural disasters, logistics.

The MapStory foundation uses GeoNode as the core engine for its MapStory.org platform [64], which aims to be a free atlas of change that everyone can edit. Users can import data and create story layers and map stories which aim to explain how and why geographic change occurs in the world.

The Global Facility for Disaster Reduction and Recovery (GFDRR) uses GeoNode for its OpenDRI Initiative which aims to use open data to the challenges of reducing vulnerability and building resilience to natural hazards and the impact of climate changes across the globe. Within this context a number

[62]https://worldmap.harvard.edu/
[63]https://geonode.wfp.org/
[64]https://mapstory.org/

of GeoNode instances have been deployed: InnovationLab GeoNode [65] maintains a curation of hazard datasets which are used by other GFDRR applications. MASDAP [66] is a geoportal used by the Government of Malawi to disseminate open data coming from various ministries to build resilience to disasters caused by the changing climate.

The Global Earthquake Model Foundation (GEM) [67] uses GeoNode for building the OpenQuake Platform, its open geospatial data framework. The platform contains a number of layers and models which can be run to analyze the earthquake risk for a specific region. The output of the analysis can be shared by the users by publishing in OpenQuake new layers and maps.

The European Commission DG-JRC use GeoNode in several projects such as the Risk Data Hub [68] and Biopama [69]. The Risk Data Hub improves the access and sharing of curated European-wide risk data, tools and methodologies for fostering Disaster Risk Management (DRM) related actions. The BIOPAMA project aims to build a solid information base for decision making on protected areas in the Africa, Caribbean, Pacific (ACP) region.

CONCLUSIONS AND FUTURE WORK

The GeoNode community is discussing how to address the future efforts in order to get to a new architecture, which will be implemented by GeoNode 3.0. Here is a list of a few concepts which should be implemented:

- GeoNode should support multiple data providers: classic OGC implementations (GeoServer, QGIS Server), and implementations more targeted to big data (GeoMesa [70], Solr, ElasticSearch)
- GeoNode via pycsw should provide a pycsw CSW endpoint which is directly tied to search engine technology (Elasticsearch or Solr). This will provide both broad interoperability as well as performance benefits of search engines (Corti et al., 2018a,b)
- GeoNode should support multiple map caching frameworks: so beside of GeoWebCache, it should be possible to use other cache providers like MapProxy [71] or Tegola [72]
- The main GeoNode application will still be based on Django and will be web UI framework agnostic, acting exclusively as a web based RESTful API, using an OpenAPI v3 specification and automatically documenting the web API with a Swagger UI [73]. A GeoNode developer will be able to include this API in its web portal keeping the GeoNode infrastructure separate from the user interface and using its favourite JavaScript framework and mapping client.
- Capacity to build and manage data workflows, for example processes which produce layers out of analysis and processing
- Multi-tenancy support and scaling the infrastructure with Kubernetes clusters

ACKNOWLEDGMENTS

The authors thank all the contributors to the GeoNode platform source code, particularly: Giovanni Allegri, Luca Battistella, Sebastian Benthall, Daniel Berry, Matt Bertrand, Olivier Dalang, Simone Dalmasso, Vivien Deparday, Tyler Garner, Jeffrey Johnson, Dimitris Karakostis, Benjamin Lewis, Matteo Nastasi, Ariel Núñez, Paolo Pasquali, Toni Schönbuchner, Francesco Stompanato, Tim Sutton, David Winslow,

REFERENCES

Consortium, O. G. (2016). Catalogue Service. http://www.opengeospatial.org/standards/cat/. [Online; accessed 12-March-2018].

Corti, P., Kralidis, A. T., and Lewis, B. (2018a). Enhancing discovery in spatial data infrastructures using a search engine. *PeerJ Computer Science*, 4:e152.

[65] https://www.geonode-gfdrrlab.org/
[66] http://www.masdap.mw/
[67] https://www.globalquakemodel.org/
[68] https://drmkc.jrc.ec.europa.eu/risk-data-hub
[69] http://geonode-rris.biopama.org/
[70] https://www.geomesa.org/
[71] https://mapproxy.org/
[72] https://tegola.io/
[73] https://swagger.io/

Corti, P., Lewis, B., and Kralidis, A. T. (2018b). Hypermap registry: an open source, standards-based geospatial registry and search platform. *Open Geospatial Data, Software and Standards*, 3(1):8.

Guan, W. W., Bol, P. K., Lewis, B. G., Bertrand, M., Berman, M. L., and Blossom, J. C. (2012). Worldmap–a geospatial framework for collaborative research. *Annals of GIS*, 18(2):121–134.

Infrastructures, D. S. D. (2004). the sdi cookbook. *GSDI/Nebert*.

Kralidis, A. T. (2009). Geospatial web services: The evolution of geospatial data infrastructure. In *The Geospatial Web*, pages 223–228. Springer.

Nogueras-Iso, J., Zarazaga-Soria, F. J., and Muro-Medrano, P. R. (2005). Geographic information metadata for spatial data infrastructures. *Resources, Interoperability and Information Retrieval*.