SERVING GEOSPATIAL DATA USING MODERN AND LEGACY STANDARDS: A CASE STUDY FROM THE URBAN HEALTH DOMAIN

J. Simoes[1,*], A. Cerciello[1]

[1] Byteroad, Lisbon, Portugal - (jo, ant)@byteroad.net

Commission IV, WG IV/4

KEY WORDS: Geospatial Data, Standards, Interoperability, API, Spatial Data Infrastructure, Urban Health, pygeoapi, Metadata

ABSTRACT:

The eMOTIONAL Cities project sets out to understand how the natural and built environment can shape the feelings and emotions of those who experience it. It does so with a cross-disciplinary approach which includes urban planners, doctors, psychologists, neuroscientists and engineers. At the core of this research project, lies a Spatial Data Infrastructure (SDI) which assembles datasets that characterise the emotional landscape and built environment, in different Cities across Europe and the US. The SDI is a key tool, not only to make the research data available within the project consortium, but also to allow cross-fertilisation with other ongoing projects and later on, to reach a wider public audience. For more than twenty years SDIs have adopted the OGC W*s service interfaces, which are based on SOAP, the Simple Object Access Protocol. In recent years a new "family" of APIs has emerged within OGC, which is more aligned with modern web practices. In this project, we set out to leverage the advantages of this new approach, and compiled a stack to implement an SDI based on OGC APIs. However, we realised that we still need to support the legacy standards, either because an OGC API replacement is not mature enough, or there are no implementations available. This has led us to compile another stack based on the legacy standards. In this paper we describe our architecture, along with the challenges that we had to address. Both stacks are based on OSGeo Software, and they are available on GitHub.

1. INTRODUCTION

Urban planning and design play an essential role in amplifying or diminishing built environmental threats to health promotion and disease prevention (Keedwell, 2017, Hackman et al., 2019). However, there is still a lack of good evidence and objective measures on how environmental aspects impact individual behaviour. The eMOTIONAL Cities project (eMOTIONAL Cities Consortium, 2022) sets out to understand how the natural and built environment can shape the feelings and emotions of those who experience it. It does so with a cross-disciplinary approach that includes urban planning, health, psychology, neuroscience, and environmental research. Data dissemination is one of the main focuses of the project. According to FAIR Principles, scientific research data must be findable, accessible, interoperable, and reusable. In a scientific project, good data dissemination must involve a good level of machine-actionability (Go FAIR, 2022).

At the core of this research project lies a Spatial Data Infrastructure (SDI), which assembles disparate datasets that characterise the emotional landscape and built environment in different Cities across Europe and the US. The SDI is a key tool to make the research data available within the project consortium and allow cross-fertilisation with other ongoing projects from the Urban Health Cluster and, later on, to reach a broader public audience.

1.1 Spatial Data Infrastructures

The notion of SDIs emerged more than 20 years ago and has been constantly evolving, in response to both technological and organisational developments. According to (Kotsev et al., 2020) the term SDI encompasses multiple facets, ranging from the legal and political setting, to the organisational aspects, and technological enablers that make the sharing and use of geographic data possible. In this paper we will focus on the latter, which does not mean - by any means - that other aspects are of less importance.

Traditionally, SDIs adopt the OGC W*s service interfaces (e.g.: WMS, WFS, WCS), which are based on SOAP, the Simple Object Access Protocol. However, in recent times, we have seen the rise of new architectural approaches, which are taken from a web-native perspective and, at the same time, prioritise a machine-actionability approach and data-centrism (Simoes and Cerciello, 2022a). Modern web-based APIs have numerous advantages, which speak for their efficiency and simplicity. They provide a simple approach to data processing and management functionalities, offer different encodings of the payload (e.g.: JSON, HTML, JSON-LD), can easily be integrated into different tools, and can facilitate the discovery of data through mainstream search engines such as Google and Bing (Kotsev et al., 2020). These APIs often follow a RESTful architecture, which simplifies their usage, while minimising the bandwidth usage. Moreover, the OpenAPI specification (OpenAPI Initiative, 2011) allows documentation of APIs in a vendor-independent, portable and open manner, which provides an interactive testing client within the API documentation.

OGC has embraced this new approach in its new family of standards called OGC APIs (OGC, 2020). Although still under active development, it already produced several approved standards: the 'OGC API - Features' (OGC, 2022b), the 'OGC API - EDR' (OGC, 2022a), and the 'OGC API - Processes' (OGC, 2022c) which provide standardised APIs for ensuring modern access to spatial data and processes using those data. There are many similarities in the process of designing and implementing open source and open standards. OSGeo encourages the

* Corresponding author

https://doi.org/10.5194/isprs-archives-XLVIII-4-W1-2022-419-2022 | © Author(s) 2022. CC BY 4.0 License.

use of open standards, like those from OGC and there is even a Memorandum of Understanding between the two organisations, initially signed in 2008 and updated recently, in 2022 (OSGeo, 2022, OSGeo and OGC, 2022). In practice, many long-standing OSGeo projects implement OGC standards and they often contribute to the standards development (e.g.: GDAL, Geoserver, QGIS, OpenLayers, Leaflet). However, in the majority of cases, they still implement the legacy W*s standards, rather than the new OGC APIs.

2. THE EMOTIONAL CITIES SDI

The eMOTIONAL Cities project collects mostly vector datasets, from the neuroscience and urban planning domain (see Figure 1). These datasets typically associate a location and a timestamp to other numerical or textual attributes.

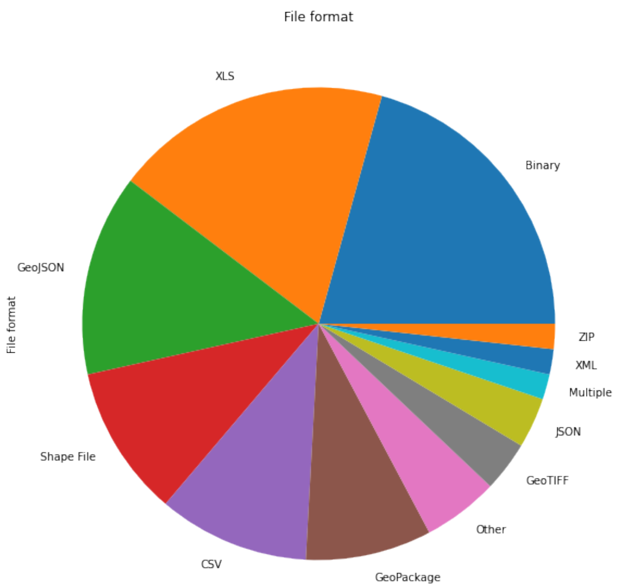

Figure 1. eMOTIONAL Cities data formats allocation

When surveyed about the purpose of these data, the majority of the researchers in the project mentioned interactive mapping and analysis as their main use cases (Simoes and Cerciello, 2022a). With this background in mind, we have selected a set of OGC standards that are best suited to accommodate the project assets in an SDI. In Table 1, we list the selected standards considering the legacy OGC W*s, and their translation to OGC APIs. At the very least, we need to be able to provide access to vector attributes and geometry and provide an interface for web mapping. All datasets should be discovered and accessed through their metadata.

OGC W*s	OGC API	Description
WFS	Features	Provides access to feature data.
WMTS	Tiles	Provides access to pre-rendered tiles of geospatial information.
CSW	Records	Provides discovery and access to metadata about geospatial resources.

Table 1. Comparison of OGC API over OGC Web Service standards

2.1 Server-side Architecture

Once we selected the standards for the SDI, we set out to select our software stack from existing server-side implementations of the OGC API standards, which are listed under the relevant OGC API GitHub (OGC, 2022d, OGC, 2022h, OGC, 2022g). If available, we would favour choosing OSGeo software, as we would like to benefit from all the advantages of using Free and Open-Source Software (FOSS), including the ability to contribute changes back, if deemed relevant. In the interest of simplicity, we would also favour software projects that implement more than one of the OGC APIs we selected, and if possible all of them. Currently, there is only one software project that addresses these two requirements: pygeoapi.

pygeoapi is a Python server implementation of the OGC API suite of standards, which “emerged as part of the next generation OGC API efforts in 2018”. It implements a number of OGC APIs, including OGC API - Tiles and OGC API - Features, and it is certified OGC Compliant and an OGC Reference Implementation for OGC API - Features - Part 1: Core 1.0. Rather than being a facade for existing W*s implementations, pygeoapi implements all the OGC APIs from scratch, leveraging all the benefits of modern web practices. For instance, it makes use of HTTP verbs (e.g.: *GET/PUT/POST/DELETE*) and codes (e.g.: 200, 201, 400, etc) and relies on content negotiation to retrieve the relevant media types. It also adopts JSON (JavaScript Object Notation) encodings, which is very popular among web developers and a first-class citizen in RESTfull (REST- REpresentational State Transfer) web services (Kralidis, T., 2019).

According to the pygeoapi documentation (pygeoapi Team, 2022), several data providers are supported for publishing vector data as OGC API - features (see Figure 2). ElasticSearch, along with SensorThingsAPI, is the data provider which provides the full range of functionality, including support for DateTime, which is an important aspect of neuroscience data.

Provider	property filters/display	resulttype	bbox	datetime	sortby
CSV	✅/✅	results/hits	❌	❌	❌
Elasticsearch	✅/✅	results/hits	✅	✅	✅
GeoJSON	✅/✅	results/hits	❌	❌	❌
MongoDB	✅/❌	results	✅	✅	✅
OGR	✅/❌	results/hits	✅	❌	❌
PostgreSQL	✅/✅	results/hits	✅	❌	✅
SQLiteGPKG	✅/❌	results/hits	✅	❌	❌
SensorThingsAPI	✅/✅	results/hits	✅	✅	✅

Figure 2. pygeoapi data providers and functionalities

ElasticSearch is a search engine, dual-licensed under the source-available Server Side Public Licence and the Elastic licence (Elastic, 2022a). It is part of the ELK stack, which includes Logstash, a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to ElasticSearch and Kibana, a tool to visualise ElasticSearch data with maps, charts and graphs (Elastic, 2022b). By adopting ElasticSearch as a vector data provider for pygeoapi, we can also leverage the complete ELK stack, to ingest and visualise our data assets.

The architecture of the eMOTIONAL Cities SDI is described in Figure 3 and Figure 10 of the APPENDIX. Data and metadata are ingested from GeoJSON files in a data lake, and published as OGC API - Features, OGC API - Records and OGC API - Tiles.

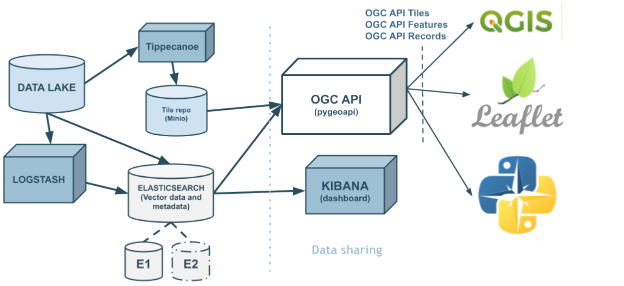

Figure 3. eMOTIONAL Cities SDI architecture

Tippecanoe (Mapbox team, 2022), is used to generate vector tiles, which are then served in a Minio bucket (MinIO, 2022). MinIO is a High-Performance Object Storage released under a GPL3.0 License. Its API is compatible with the Amazon S3 cloud storage service, which makes it a very good alternative to store large datasets in an unstructured manner, for fast access.

In order to ease the deployment of the SDI and to increase the reproducibility of this research, the software stack was virtualized into a set of docker containers, which are orchestrated using docker-compose (Docker Inc., 2022). The virtualized architecture, described in the diagram below, is published on GitHub, under a GPL License (Simoes and Cerciello, 2022b). This stack was deployed at this endpoint, where you can browse through some preliminary datasets from the project: https://emotional.byteroad.net/

2.2 Clients

The availability of client implementations of the selected standards, is a critical aspect of the usefulness of an SDI. While we don't have software to consume data published under the OGC API standards, we cannot expect a serious uptake from the users. Our mission in this project does not end with publishing data using the selected standards. We want to make sure that the researchers are able to integrate these data in their workflows, either by using their usual tools, or by using new tools which we should be able to recommend and provide training for. As in the case of server-side implementations, we referred to the GitHub pages of the standards, in order to identify existing client-side implementations of the selected OGC API standards.

OGC API - Features already features eight client implementations, including desktop applications, libraries and JavaScript APIs (OGC, 2022d). It is supported in well-known OSGeo projects such as GDAL or QGIS (see Figure 4). In addition, as pygeoapi publishes data in GeoJSON, it is compatible with all clients which are capable of consuming GeoJSON.

The support to OGC API - Tiles was merged into the main branch of OpenLayers, in September 2021 (https://github.com/openlayers/openlayers/pull/10963). On this repository (Simoes, 2022), we prototyped a web map consuming vector tiles from OGC API - Tiles, using OpenLayers (see Figure 5).

Although in the eMOTIONAL Cities SDI we are focused on vector tiles, it is worth mentioning that during the March 2022

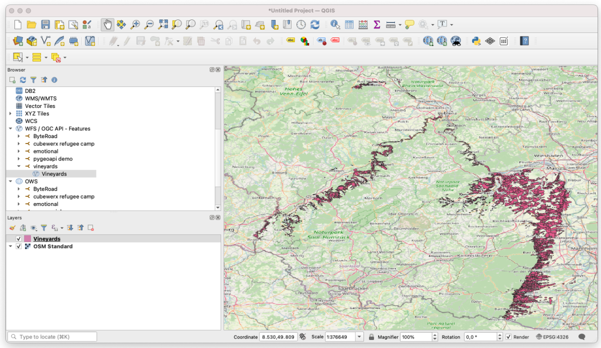

Figure 4. OGC API - Features support in QGIS

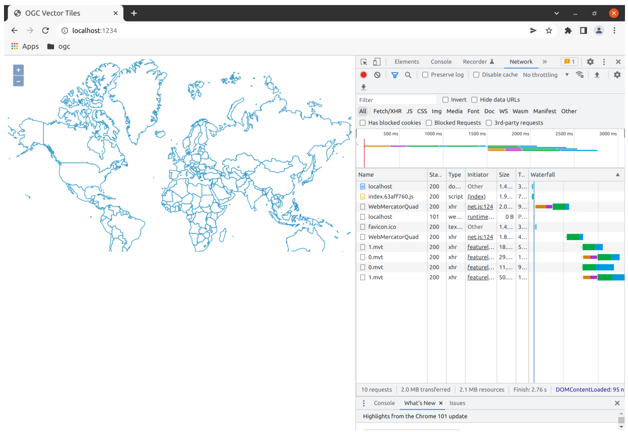

Figure 5. A basic implementation using OGC API - Tiles with OpenLayers

Joint OGC-OSGeo-ASF code sprint (OSGeo, 2021), a plugin for Leaflet which adds support for map tiles was prototyped (https://github.com/openlayers/openlayers/pull/10963).

We provide some considerations about clients for OGC API Records, in section 2.4.

2.3 The legacy stack

During the steps of collecting the requirements for the SDI of eMOTIONAL Cities, no participant has explicitly expressed the need to publish data following a specific standard. Neuroscience research does not adopt many standards, although that need is acknowledged, also for GIS data (Poline et al., 2022). For this reason, we have tried to create solutions that rely on emerging technological trends and thus are more likely to stay around in the future. On the other hand, researchers from the urban planning domain are often used to work with the legacy W*s, and as we described in section 2.2, there are not that many client implementations available, which means that in some cases their preferred tool may not support an OGC API, yet. As a compromise, we decided to provide a backward compatibility system. This has led us to deploy a legacy stack, composed of mature free and open-source solutions widely used in the GIS domain, along with our OGC API stack.

The legacy stack is built around the free and open-source GeoServer, which implements industry-standard OGC protocols such as Web Feature Service (WFS), Web Map Service (WMS), and Web Coverage Service (WCS). Additional formats and publication options are available as extensions including Web Processing Service (WPS), and Web Map Tile Service (WMTS) (GeoSolutions, 2022a). The architecture of this solution is described in Figure 6 and in Figure 11 of the APPENDIX.

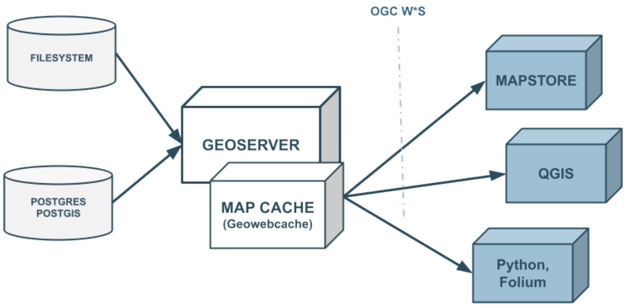

Figure 6. eMOTIONAL Cities Legacy stack architecture

GeoServer can be configured to collect data from different sources, like filesystems and DBs (spatial or not). As part of the project, GeoServer will exclusively share data, which will be gathered in the Data Lake, the same used in the OGC API stack, and ingested in GeoServer's stores by suitably configured data pipelines. A diagram with the server side architecture of this stack is shown below. Data from GeoServer can be easly consumed by a wide range of GIS clients, including the MapStore, an highly modular Open Source WebGIS framework, included in the EC's SDI (GeoSolutions, 2022b).

2.4 Metadata

In a project built around FAIR principles, metadata leads a primary role. Metadata allows the description of the datasets exposed by the other components of the SDI, making the data searchable and accessible. Using metadata to describe resources enables its understanding by humans as well as machines. The literature is rich on the subject, and it is not an exaggeration to consider an SDI complete only when it is provided with a catalogue of its data (AIMS FAO, 2019). The reference standard for catalogues of GIS records is the OGC CSW. In CSW, metadata records are XML documents served over HTTP. CSW is a profile of the OGC Catalog Service specification (OGC, 2016). OGC API - Records (from now OARec) specification has been developed with the same objective as CSW, but built with the characteristics of an OGC API. Unlike CSW, which exposes records as XML documents, OARec can expose records in different formats, although the JSON format remains an essential requirement (OGC, 2022e). According to (OGC, 2022f) there are a number of ways that records can be deployed as a "collection of records" or a catalogue.

- Crawlable catalogue: To implement a catalogue where records are stored as individual files in one or more web-accessible locations and can be crawled by simply following hypermedia controls
- Searchable catalogue: To implement a catalogue where records are stored in some backend database and are accessed through an access/search API
- Local resources catalogue: To implement a local resource catalogue at an OGC API resources endpoint

Deciding which record catalogue to use for EC was not an easy choice, because metadata is the single entry point of an SDI, and thus it does not make sense to have multiple deployments, using legacy and modern standards. Therefore, it was decided to use the OGC API option, since it makes no sense to present an entry point that does have the characteristics and benefits that we favour adopting in the other solutions. The exploratory nature of the project enables us to adopt this relatively new standard. However, we do acknowledged that the fact that the standard is still a draft may bring challenges in other contexts, such as government agencies. Given the fundamental role of metadata in an SDI, the fact that the standard is not yet stable enough to be covered under national and international directives, such as INSPIRE, may be an issue for many use cases. For those cases, OGC CSW still remains a safer choice (Chester, S., 2021). Nonetheless, the promising uptake of the standard, which even in an early stage of development, already features server-side implementations must be acknowledged. Free and Open Source solutions, such as pygeoapi, allow users to have an implementation of OARec aligned with advances in standards, which is perfectly usable in a real SDI (OGC, 2022g).

On the client-side, the OGC API - Records searchable catalogue is supported natively in QGIS, through the Metasearch core plugin (Kralidis, T. and Brury, A. and Dubinin, M. and Tzotsos, A. and Duivenvoorde, R., 2022). Figure 7 shows a screenshot of this plugin, which allows users to browse through the records of OGC API - Records catalogues, view the metadata and pull the associated data. This tool does not provide however, an interface for the authoring of metadata.

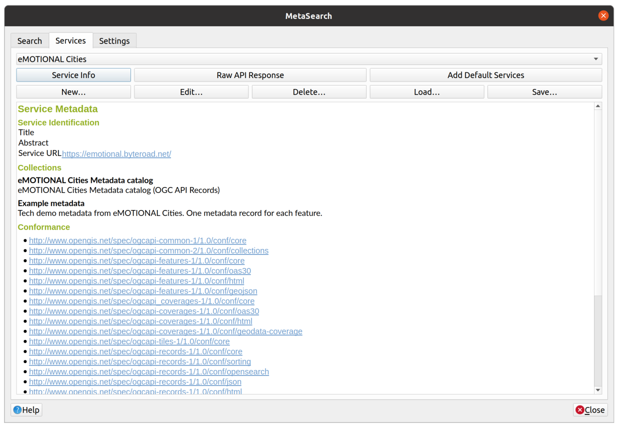

Figure 7. Metasearch

We also did not find any web application, which enabled browsing OGC API - records on the web, apart from the compatible STAC Vue browser, which is restricted to crawlable catalogues (Radiant Earth Foundation, 2022). A screenshot of this application is shown on Figure 8.

This has led us to develop a metadata browser, A-gis-full-of-records (Cerciello and Simoes, 2022), using React Framework (Meta Open Source, 2022) and Leaflet (Agafonkin, V., 2022). The tool, shown on Figure 9, is still under development, but it's a good demonstration of the developer friendliness of OGC API.

3. DISCUSSION AND CONCLUSIONS

The OGC APIs promise exciting features for modern SDIs. Although (at least some of them are) still at an early stage, there is an active development taking place in code sprints and

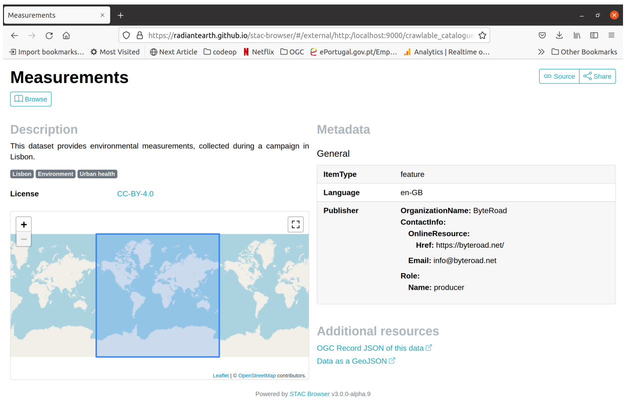

Figure 8. A crawlable catalogue in Stac Browser

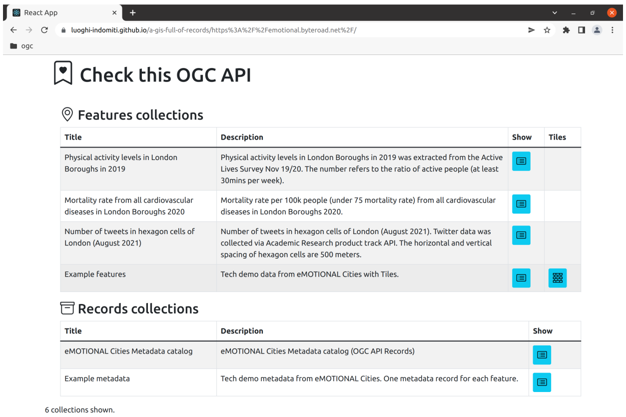

Figure 9. The metadata browser prototype implemented for eMOTIONAL Cities project

GitHub repositories, inclusive of the developer community and welcoming feedback. Similarly to what happens with OSGeo projects, hopefully, this trend will lead to high-quality specs with increased ownership from the developer community. We were positively surprised to see that it is already possible to implement a server-side architecture for an SDI, entirely based on OGC APIs and OSGeo software. pygeoapi proved to be a sound solution for implementing multiple OGC API standards. It is very easy to install, especially using the official docker image (GeoPython, 2022), and is very flexible through its modular architecture, which relies on plugins for multiple data providers. Although the project is under active development, since it is free and open-source, we were able to contribute to its development.

However, in terms of clients, there are still few implementations, and widely used products by the GIS community (e.g. Leaflet, MapStore, Kibana) still lack native support to the OGC APIs, along with their support to legacy W* services.

This has led us to deploy a stack for the legacy W*s to enable the GIS researchers from the eMOTIONAL Cities project to access the project data assets using their usual tools. We believe this type of hybrid solution will work for many use cases, such as public services or research projects, where there are GIS experts, trained in legacy W*s services but with little knowledge about the OGC APIs. However, it should be mentioned that even in those cases, it will be easier to onboard a web developer with no knowledge of GIS to use the OGC APIs than to use the legacy stack.

Although we have described many advantages of the OGC APIs, we believe there will be no massive uptake until enough client implementations are available and users are trained to use them. We are committed to these two fronts and have already delivered training in using clients during the OGC code sprints (Simoes, J., 2021, Cerciello, A., 2021) and started our client developments. We hope these efforts, within the scope of the four year eMOTIONAL Cities project, will contribute towards the development of the OGC APIs and, not least significantly, towards their visibility within the GIS community and the broader developer community.

ACKNOWLEDGEMENTS

This research is developed under the eMOTIONAL Cities' Project, which received funding from European Union's Horizon 2020 research and innovation programme, under the grant agreement No 945307. The eMOTIONAL Cities Project is a consortium of 12 partners co-coordinated by IGOT and FMUL, taking place between 2021 and 2025. More information at https://emotionalCities-h2020.eu/.

REFERENCES

Agafonkin, V., 2022. Leaflet. leafletjs.com (01 May 2022).

AIMS FAO, 2019. FAIR Principles and Digital Objects: The role of METADATA. aims.fao.org/news/fair-principles-digital-objects-role-metadata (01 May 2022).

Cerciello, A., 2021. Bootstrap an OGC API client with React. github.com/Luoghi-indomiti/bootstrap-ogc-api-react (01 May 2022).

Cerciello, A., Simoes, J., 2022. Luoghi-indomiti/a-gis-full-of-records. doi.org/10.5281/zenodo.6623262.

Chester, S., 2021. INSPIRE and OGC APIs - Part 1: Modernising INSPIRE. ogc.org/blog/4603 (01 May 2022).

Docker Inc., 2022. Docker Compose. docs.docker.com/compose (01 May 2022).

Elastic, 2022a. Elastic Licence. elastic.co/licensing/elastic-license (01 May 2022).

Elastic, 2022b. ELK Stack. elastic.co/what-is/elk-stack (01 May 2022).

eMOTIONAL Cities Consortium, 2022. eMOTIONAL Cities - Mapping the cities through the senses of those who make them. emotionalcities-h2020.eu.

GeoPython, 2022. pygeoapi docker image. hub.docker.com/r/geopython/pygeoapi (01 May 2022).

GeoSolutions, 2022a. GeoServer. geoserver.org (01 May 2022).

GeoSolutions, 2022b. MapStore. mapstore.readthedocs.io/en/latest/ (01 May 2022).

Go FAIR, 2022. FAIR Principles. www.go-fair.org/fair-principles (01 May 2022).

Hackman, D. A., Robert, S. A., Grübel, J., Weibel, R. P., Anagnostou, E., Hölscher, C., Schinazi, V. R., 2019. Neighborhood environments influence emotion and physiological reactivity. *Sci Rep*, 9, 9498. doi.org/10.1038/s41598-019-45876-8.

Keedwell, P., 2017. *Headspace: The Psychology of City Living*. Aurum Press, London.

Kotsev, A., Minghini, M., Tomas, R., Cetl, V., Lutz, M., 2020. From Spatial Data Infrastructures to Data Spaces - A Technological Perspective on the Evolution of European SDIs. *ISPRS International Journal of Geo-Information*, 9(3), no. 176. doi.org/10.3390/ijgi9030176.

Kralidis, T., 2019. pygeoapi. pygeoapi.io (01 May 2022).

Kralidis, T. and Brury, A. and Dubinin, M. and Tzotsos, A. and Duivenvoorde, R., 2022. Metasearch. github.com/qgis/QGIS/tree/master/python/plugins/MetaSearch (01 May 2022).

Mapbox team, 2022. Tippecanoe. Github. github.com/mapbox/tippecanoek (01 May 2022).

Meta Open Source, 2022. React. reactjs.org (01 May 2022).

MinIO, 2022. MinIO. min.io (01 May 2022).

OGC, 2016. Catalogue Service. ogc.org/standards/cat (01 May 2022).

OGC, 2020. OGC APIs - Building Blocks for Location. ogcapi.ogc.org (01 May 2022).

OGC, 2022a. OGC API - Environmental Data Retrieval. Github. github.com/opengeospatial/ogcapi-environmental-data-retrieval (01 May 2022).

OGC, 2022b. OGC API - Features. Github. github.com/opengeospatial/ogcapi-features (01 May 2022).

OGC, 2022c. OGC API - Processes. Github. github.com/opengeospatial/ogcapi-processes (01 May 2022).

OGC, 2022d. OGC API Features Implementations. github.com/opengeospatial/ogcapi-features/tree/master/implementations#servers (01 May 2022).

OGC, 2022e. OGC API Records - Encodings. docs.ogc.org/DRAFTS/20-004.html#clause-encodings (01 May 2022).

OGC, 2022f. OGC API Records - Overview. docs.ogc.org/DRAFTS/20-004.html#overview (01 May 2022).

OGC, 2022g. OGC API Records Implementations. github.com/opengeospatial/ogcapi-records/blob/master/implementations.md (01 May 2022).

OGC, 2022h. OGC API Tiles Implementations. github.com/opengeospatial/ogcapi-tiles/blob/master/implementations.adoc (01 May 2022).

OpenAPI Initiative, 2011. OpenAPI Specification. spec.openapis.org/oas/latest.html (01 May 2022).

OSGeo, 2021. Joint OGC OSGeo ASF Code Sprint 2021. osgeo.org/events/joint-ogc-osgeo-asf-code-sprint-2021 (01 May 2022).

OSGeo, 2022. OGC Memorandum of Understanding. OSGeo. osgeo.org/resources/ogc-memorandum-understanding/ (31 January 2022).

OSGeo and OGC, 2022. MEMORANDUM OF UNDERSTANDING. osgeo.org/wp-content/uploads/MOU_OGC_OSGeo_2022_signed.pdf (31 January 2022).

Poline, J., Kennedy, D., Sommer, F., Ascoli, G., Van Essen, D., 2022. Is Neuroscience FAIR? A Call for Collaborative Standardisation of Neuroscience Data. *Neuroinform*, 2022. doi.org/10.1007/s12021-021-09557-0.

pygeoapi Team, 2022. pygeoapi Data Providers. docs.pygeoapi.io/en/stable/data-publishing/index.html#providers-overview (01 May 2022).

Radiant Earth Foundation, 2022. Stack Browser. github.com/radiantearth/stac-browser (01 May 2022).

Simoes, J., 2021. How-to access OGC API Features without writing one line of code. github.com/opengeospatial/developer-events/wiki/2022-Joint-OGC-(01 May 2022).

Simoes, J., 2022. emotional-cities/ol-oat. doi.org/10.5281/zenodo.6620894.

Simoes, J., Cerciello, A., 2022a. Deliverable 3.1 - data management plan i. emotionalcities-h2020.eu/wp-content/uploads/2022/02/eMC_2021.08_D3.1_Data_Management_Plan.pdf (01 May 2022).

Simoes, J., Cerciello, A., 2022b. emotional-cities/openapi-sdi. doi.org/10.5281/zenodo.6591180.

APPENDIX

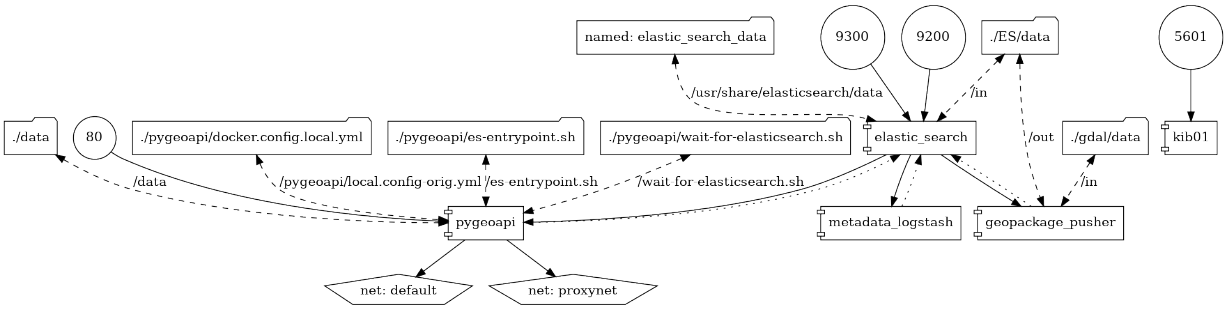

Figure 10. eMOTIONAL Cities SDI docker-compose structure

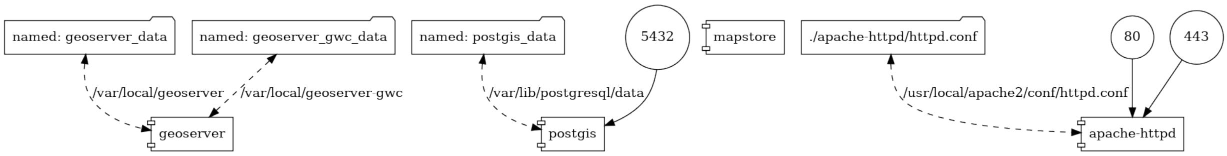

Figure 11. eMOTIONAL Cities Legacy stack docker-compose structure