The SPARQL Unicorn Ontology documentation: Exposing RDF geodata using static GeoAPIs

Timo Homburg[1], Florian Thiery[2]

[1] Research Squirrel Engineers Network & Hochschule Mainz University of Applied Sciences
[2] Research Squirrel Engineers Network & Leibniz-Zentrum für Archäologie (LEIZA)

DOI: 10.5281 / zenodo.10570985

Abstract: We introduce the ontology documentation feature of the SPARQLing Unicorn QGIS plugin, allowing the conversion of RDF data dumps to static HTML deployments with static versions of well-known APIs such as OGC API Features, thereby deploying linked open data as linked open usable data. Contents of RDF data are deployed interoperability and accessible for many research communities, e.g. directly in QGIS. This paper discusses the tool's motivation, the conversion process with test datasets, limitations, standardization issues, and future developments of this approach in general.

Keywords: SPARQLing Unicorn, RDF, Static GeoAPIs

1 Introduction

In recent years, many academic disciplines, but also companies in the industry, rely on knowledge graphs to represent, among others, geospatial data encoded in the Resource Description Framework (RDF) [1]. RDF is preferred in many research projects because it connects different research data using standardized and interoperable vocabularies. RDF data can be hosted as single files or in specialized databases called triple stores [2], which use a REST API [3], typically a SPARQL [4] endpoint, to provide access to said geospatial data. It is then up to individual applications to display this geospatial information in web frontends, e.g. using Leaflet or other means to visualize its contents.

Triple stores are a good solution for institutions with the financial and technical means to host their linked databases. However, researchers without such means will commonly only host RDF data dumps in repositories such as Zenodo[1] or Github[2].

These data dumps contain useful data but are often not easily accessible to a wider audience beyond the linked open data (LOD) community, as the discovery, modification, and processing of these data dumps usually require knowledge of programming and LOD software libraries or tools. These tools however provide access to RDF triples and not necessarily data in formats end users expect.

Data dumps fulfil the FAIR criteria of data publishing [5] but do fall short of producing Linked Open Usable (LOUD) data [6], which is easily consumable by common applications per research community that could be targeted by the publication of the dataset.

[1] https://zenodo.org/
[2] https://github.com/

This publication wants to explore and find the limits of how (geo-)spatial data encoded in RDF and limited to static data publications can be exposed via static APIs to end users and applications in such ways that they can consume data directly from e.g. a Github Page static webspace.

To that end, we present the SPARQL Unicorn Ontology documentation feature, which allows the generating of enhanced HTML deployments from RDF data, which may be hosted on static web spaces and expose static (Geo-)APIs for use in applications such as QGIS [7]. This feature is based on the SPARQL Unicorn Principles [8] and the SPARQLing Unicorn QGIS Plugin [9,10].

2 Foundations

RDF knowledge graphs are directed graphs in which each node and edge are identified by a unique resource identifier (URI) [11]. The graph is specified as a set of statements called triples, comprised of a subject (a node), a predicate (an edge), and an object (a node). Special nodes in the graph describe the data schema (TBox), and other nodes contain data (ABox). Data schemas and the data itself are both represented as linked open data, even though often published in separate LOD files. Geospatial data in knowledge graphs may be encoded in various vocabularies, compatible with Best Practices to publish spatial data on the web [12]. An example of geometry representation in RDF is the OGC-defined GeoSPARQL vocabulary [13,14] which can encode geospatial vector data as features, geometries, and collections of these.

The documentation of RDF data in HTML is not a new concept. Commonly, applications such as Pubby[3] or the HTML frontend of Wikidata [15] use HTML templates filled by results of SPARQL queries that query a triple store in real-time. For static data dumps, this is not an option because a SPARQL endpoint and database are not available. Hence, data dumps can only be downloaded and processed by a client application.

A static API is a collection of (usually JSON) files hosted on a webspace. The collection of files and their contents are organized in such a way that they mimic the behaviour of commonly used APIs for the audience they target. Unlike a dynamic API, the static API cannot perform dynamic tasks, i.e. searches or calculations, unless the results of these calculations have been precalculated and published as result documents along with the static API. A common usage of static APIs is that they are published by static site generators such as Jekyll to provide machine-readable access to items served on the static homepage in HTML. We choose a similar approach for the ontology documentation plugin.

Geospatial data APIs, usually defined by the Open Geospatial Consortium (OGC), have been used to expose geospatial data for many years. Web Feature Services (WFS) [16] were used to share vector data, Web Map Services (WMS) were used to share map image tiles and Web Coverages Services (WCS) were used to share raster and coverage data. For a few years, the OGC has begun to update these API specifications as OGC API standards, with OGC API Features replacing WFS to a much greater extent. Consequently, mimicking OGC API Features as a static API would be a suitable way to provide geodata to the geospatial user community.

[3] https://github.com/cygri/pubby

3 Ontology Documentation Workflow

The SPARQL Unicorn Ontology documentation script receives an RDF dump as input and processes it as follows. It detects namespaces, which expose data in the knowledge graph. One of these data namespaces, if not specified as an input parameter, is chosen as the target namespace for the generation of the documentation. Each subject in the target namespace is converted to an HTML+RDFa [17] page, which contains all predicates and objects linked to it in the knowledge graph. In addition, equivalent representations in TTL [18] and JSON-LD [19] are generated by default. With the generation of additional index pages and navigation, the user can navigate the contents of the knowledge graph in a static HTML deployment. Figure 1 describes the main steps in the documentation workflow and the next sections will describe the main outcomes of this way of documentation.

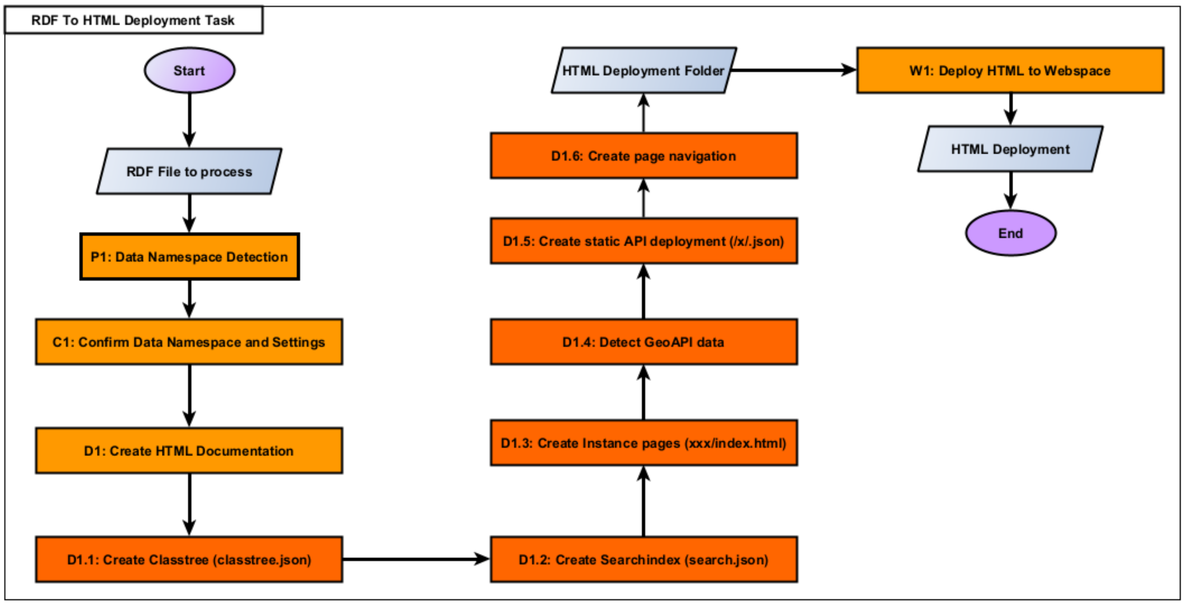

RDF To HTML Deployment Task
Start
RDF File to process
P1: Data Namespace Detection
C1: Confirm Data Namespace and Settings
D1: Create HTML Documentation
D1.1: Create Classtree (classtree.json)
D1.2: Create Searchindex (search.json)
D1.3: Create Instance pages (xxx/index.html)
D1.4: Detect GeoAPI data
D1.5: Create static API deployment (/x/.json)
D1.6: Create page navigation
HTML Deployment Folder
W1: Deploy HTML to Webspace
HTML Deployment
End

Figure 1: Ontology documentation workflow broken down in its main steps; the deployment to a webspace is optional. Timo Homburg, CC BY 4.0.

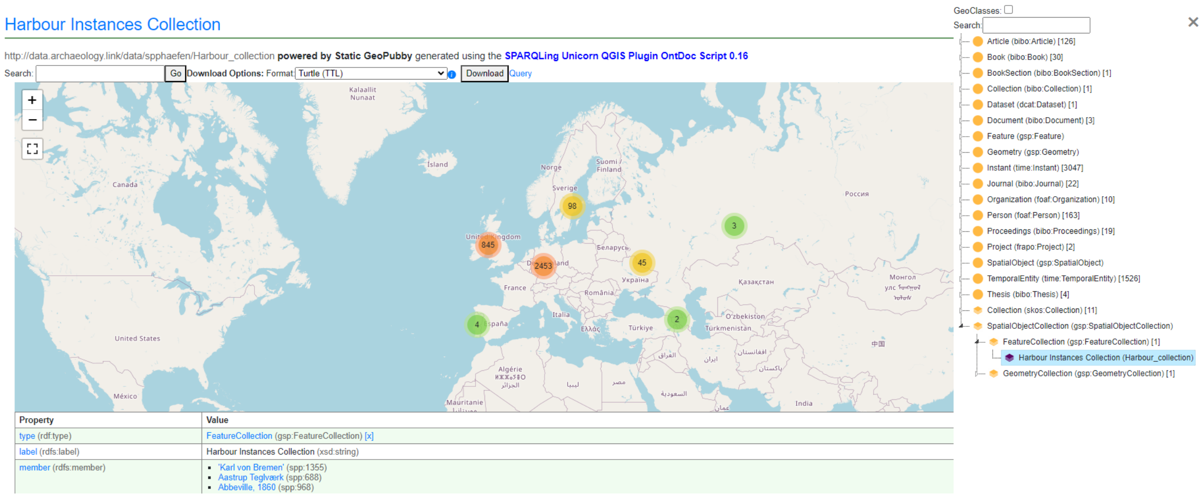
3.1 Detection of Vocabularies

According to the linked open data principles [20], RDF graphs can be expected to reuse already established vocabularies common in the linked open data community. The SPARQL Unicorn Ontology documentation script [21] is able to detect these vocabularies in the knowledge graph. The reasoning behind this is that each vocabulary that is used for modeling specific kinds of data, which could possibly be exported to or exposed as their own community-specific formats. For example, GeoSPARQL-formatted data in a knowledge graph could be represented as GeoJSON [22], bibliography related information in the knowledge graph modeled using the BIBO ontology [23] could be exposed as BibTeX files [24], etc.

Being able to detect domain-specific contents of well-formatted RDF graphs hence becomes the key for providing linked open usable data for a variety of research communities. This is in our opinion an often-overlooked piece when working and providing Linked Open Data.

3.2 Generation of HTML Views

Since the ontology documentation script may detect vocabularies from the given knowledge graph, HTML Views can be optimized to display data contained in these knowledge graph parts using different views. For instance, geodata may be visualized using a Leaflet HTML view [25], as previously shown in Figure 2, image data could be shown as an image grid or bibliographical data could be shared in the BibTeX format in HTML ready to be copied as shown in Figure 3.



Harbour Instances Collection

http://data.archaeology.link/data/spphaefen/Harbour_collection **powered by Static GeoPubby** generated using the **SPARQLing Unicorn QGIS Plugin OntDoc Script 0.16**

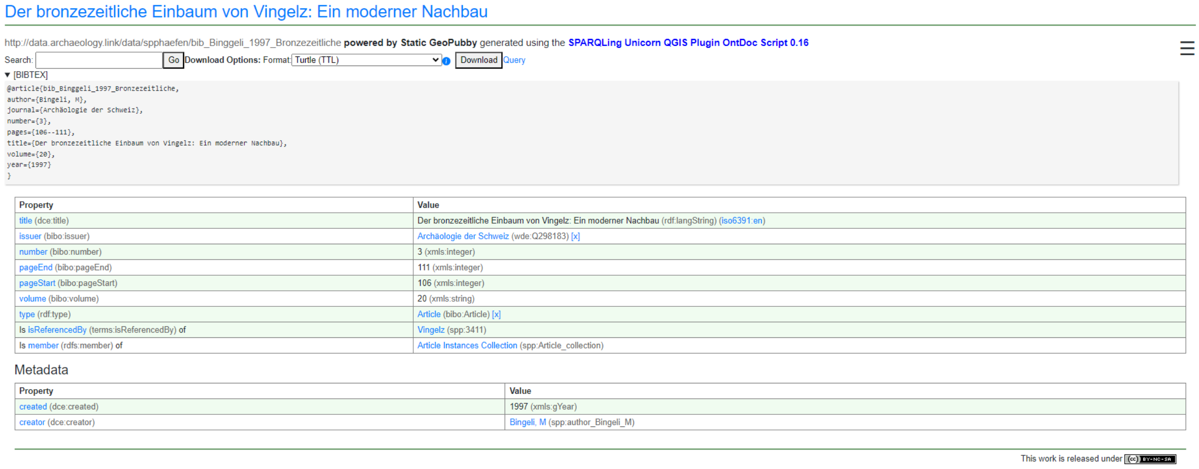
Search: Go **Download Options:** Format: Turtle (TTL) Download Query

Property	Value
type (rdf:type)	FeatureCollection (gsp:FeatureCollection) [x]
label (rdfs:label)	Harbour Instances Collection (xsd:string)
member (rdfs:member)	▪ 'Karl von Bremen' (spp:1355) ▪ Aastrup Teglværk (spp:688) ▪ Abbeville, 1860 (spp:968)

GeoClasses: □
Search:

- Article (bibo:Article) [126]
- Book (bibo:Book) [30]
- BookSection (bibo:BookSection) [1]
- Collection (bibo:Collection) [1]
- Dataset (dcat:Dataset) [1]
- Document (bibo:Document) [3]
- Feature (gsp:Feature)
- Geometry (gsp:Geometry)
- Instant (time:Instant) [3047]
- Journal (bibo:Journal) [22]
- Organization (foaf:Organization) [10]
- Person (foaf:Person) [163]
- Proceedings (bibo:Proceedings) [19]
- Project (frapo:Project) [2]
- SpatialObject (gsp:SpatialObject)
- TemporalEntity (time:TemporalEntity) [1526]
- Thesis (bibo:Thesis) [4]
- Collection (skos:Collection) [11]
- SpatialObjectCollection (gsp:SpatialObjectCollection)
  - FeatureCollection (gsp:FeatureCollection) [1]
    - Harbour Instances Collection (Harbour_collection)
  - GeometryCollection (gsp:GeometryCollection) [1]

Fig. 2. SPP Harbours Dataset: Visualisation of a FeatureCollection of harbours and a classtree navigation to navigate concepts, geodata and linked adjacent data encoded in RDF[4].



Der bronzezeitliche Einbaum von Vingelz: Ein moderner Nachbau

http://data.archaeology.link/data/spphaefen/bib_Binggeli_1997_Bronzezeitliche **powered by Static GeoPubby** generated using the **SPARQLing Unicorn QGIS Plugin OntDoc Script 0.16**

Search: Go **Download Options:** Format: Turtle (TTL) Download Query

▼ [BIBTEX]

```
@article{bib_Binggeli_1997_Bronzezeitliche,
author={Bingeli, M},
journal={Archäologie der Schweiz},
number={3},
pages={106--111},
title={Der bronzezeitliche Einbaum von Vingelz: Ein moderner Nachbau},
volume={20},
year={1997}
}
```

Property	Value
title (dce:title)	Der bronzezeitliche Einbaum von Vingelz: Ein moderner Nachbau (rdf:langString) (iso6391:en)
issuer (bibo:issuer)	Archäologie der Schweiz (wde:Q298183) [x]
number (bibo:number)	3 (xmls:integer)
pageEnd (bibo:pageEnd)	111 (xmls:integer)
pageStart (bibo:pageStart)	106 (xmls:integer)
volume (bibo:volume)	20 (xmls:string)
type (rdf:type)	Article (bibo:Article) [x]
Is isReferencedBy (terms:isReferencedBy) of	Vingelz (spp:3411)
Is member (rdfs:member) of	Article Instances Collection (spp:Article_collection)

Metadata

Property	Value
created (dce:created)	1997 (xmls:gYear)
creator (dce:creator)	Bingeli, M (spp:author_Bingeli_M)

This work is released under (cc) BY-NC-SA

[4] cf. https://archaeolink.github.io/SPP1630Harbours-RDF/Harbour_collection/index.html

Fig. 3. Visualisation of a bibliographic reference modeled using the BIBO ontology[5]. Bibliographic references may be visualised and can confirm other statements, e.g. of geodata in the knowledge graph. Here, the location of a Harbour is confirmed.

[5] cf. https://archaeolink.github.io/SPP1630Harbours-RDF/bib_Binggeli_1997_Bronzezeitliche/index.html

Different other views are a possibility, even though some depend on different data exports to be created. Currently, the documentation script, apart from RDF and geodata, is able to generate graph formats such as GraphML [26] , general purpose formats such as CSV [27] or TSV. It is likely that these formats will be expanded in the future when more semantic web vocabularies will be supported.

4 Static API Support

The SPARQL Unicorn Ontology documentation script allows the generation of a static OGC API Features API [28] with the following capabilities:

- Listing all FeatureCollections with metadata gained from the knowledge graph
- Querying all features of a collection

This allows for supported clients to download all available FeatureCollections, but not to filter said FeatureCollections by creating a server-side query. Since FeatureCollections are a new addition to GeoSPARQL and since most of Semantic Web Data are usually typed, but not necessarily grouped, FeatureCollections may also be inferred by the SPARQLUnicorn documentation script. These collections do not change graph data but provide new views on its contents.

Two other examples for static APIs that can be generated are the static IIIF API [29] , which allows for the download of images, similar to Canopy IIIF[6], but not in every resolution or modification (cut outs etc) and the CKAN API[7], which allows the download of selected graph parts, but does not provide full search functionality.

4.1 Static API Frontends in HTML and JavaScript

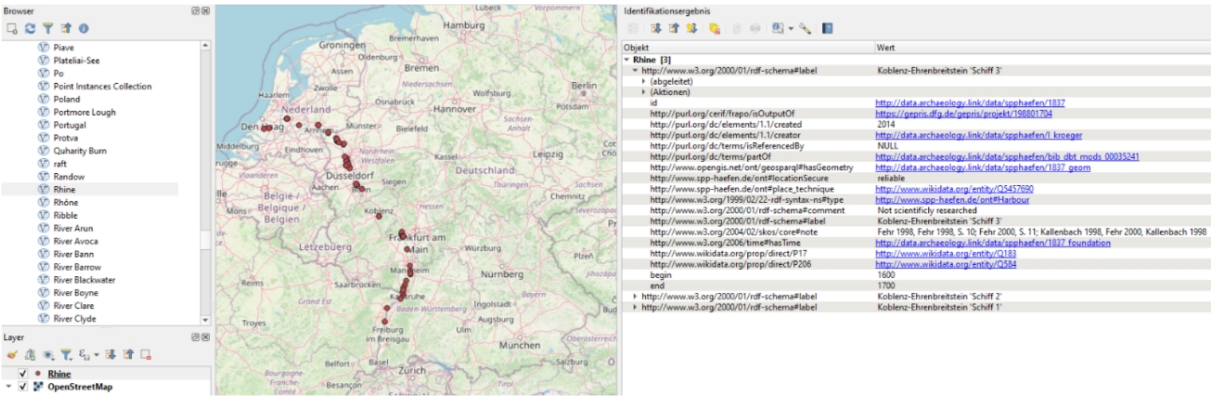
To properly expose and inform users of the existence of a static API in their HTML deployment, static APIs should also follow the principles of good API documentations. A majority of APIs are documented by creating an OpenAPI [30] specification document both in JSON and in HTML, which we also employ for OGC API Features. Other APIs such as the IIIF image API provide clients in JavaScript to access API data which make use of the generated static API deployments. We think that an API specification document, API access in the browser and using specified clients are essential for the provision of usable data for the respective user bases.

4.2 Static API Support in QGIS

Current implementations of QGIS provide support for OGC API Features. We tested that QGIS is prepared to load the metadata of the OGC API Features service first and subsequently tries to load a data sample (by default) 10 features from the OGC API Features API. The static OGC API Features API is not equipped to answer a query of the first 10 features and will instead return the whole Feature Collection. In the current implementation this in not problematic in QGIS. It will load and display the respective feature collections as can be seen in Figure 4. The configuration of such a static OGC API Features endpoint is shown in Figure 5.

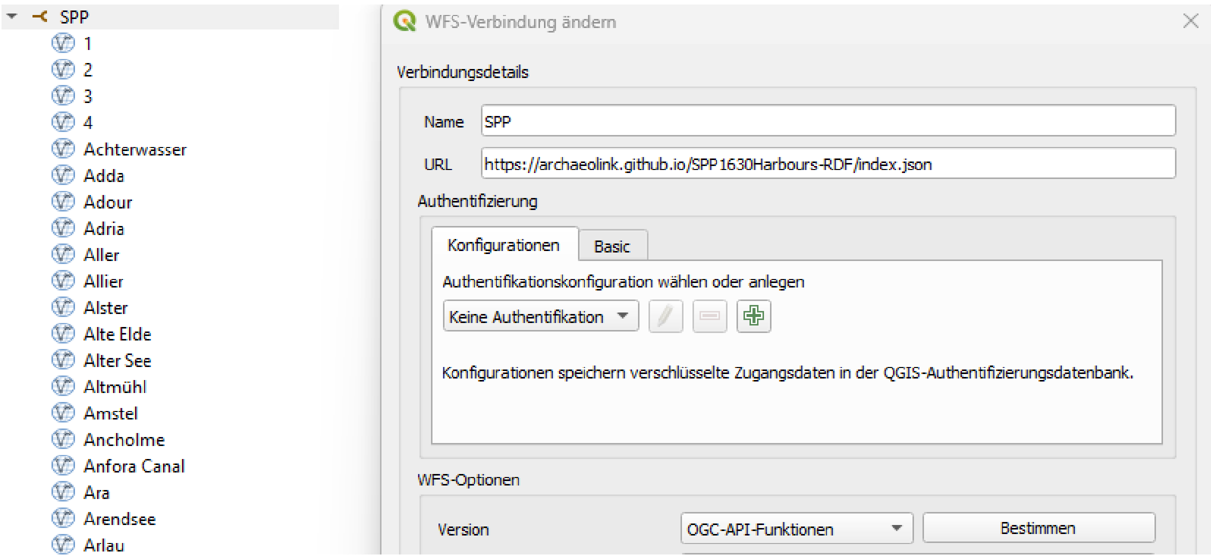
[6] https://canopy-iiif.github.io/docs/

[7] https://docs.ckan.org/en/2.9/api/



Objekt	Wert
Rhine [3]	
http://www.w3.org/2000/01/rdf-schema#label	Koblenz-Ehrenbreitstein 'Schiff 3'
(abgeleitet)	
(Aktionen)	
id	http://data.archaeology.link/data/spphaefen/1837
http://purl.org/cerif/frapo/isOutputOf	https://gepris.dfg.de/gepris/projekt/198801704
http://purl.org/dc/elements/1.1/created	2014
http://purl.org/dc/elements/1.1/creator	http://data.archaeology.link/data/spphaefen/l_kroeger
http://purl.org/dc/terms/isReferencedBy	NULL
http://purl.org/dc/terms/partOf	http://data.archaeology.link/data/spphaefen/bib_dbt_mods_00035241
http://www.opengis.net/ont/geosparql#hasGeometry	http://data.archaeology.link/data/spphaefen/1837_geom
http://www.spp-haefen.de/ont#locationSecure	reliable
http://www.spp-haefen.de/ont#place_technique	http://www.wikidata.org/entity/Q5457690
http://www.w3.org/1999/02/22-rdf-syntax-ns#type	http://www.spp-haefen.de/ont#Harbour
http://www.w3.org/2000/01/rdf-schema#comment	Not scientificly researched
http://www.w3.org/2000/01/rdf-schema#label	Koblenz-Ehrenbreitstein 'Schiff 3'
http://www.w3.org/2004/02/skos/core#note	Fehr 1998, Fehr 1998, S. 10; Fehr 2000, S. 11; Kallenbach 1998, Fehr 2000, Kallenbach 1998
http://www.w3.org/2006/time#hasTime	http://data.archaeology.link/data/spphaefen/1837_foundation
http://www.wikidata.org/prop/direct/P17	http://www.wikidata.org/entity/Q183
http://www.wikidata.org/prop/direct/P206	http://www.wikidata.org/entity/Q584
begin	1600
end	1700
http://www.w3.org/2000/01/rdf-schema#label	Koblenz-Ehrenbreitstein 'Schiff 2'
http://www.w3.org/2000/01/rdf-schema#label	Koblenz-Ehrenbreitstein 'Schiff 1'

Fig. 4. Usage of the static OGC API Features API in QGIS. Harbours next to rivers and lakes generated as FeatureCollections in the knowledge graph and accessible as FeatureCollections within QGIS.



Verbindungsdetails

Name: SPP

URL: https://archaeolink.github.io/SPP1630Harbours-RDF/index.json

Authentifizierung

Authentifikationskonfiguration wählen oder anlegen

Keine Authentifikation

Konfigurationen speichern verschlüsselte Zugangsdaten in der QGIS-Authentifizierungsdatenbank.

WFS-Optionen

Version: OGC-API-Funktionen

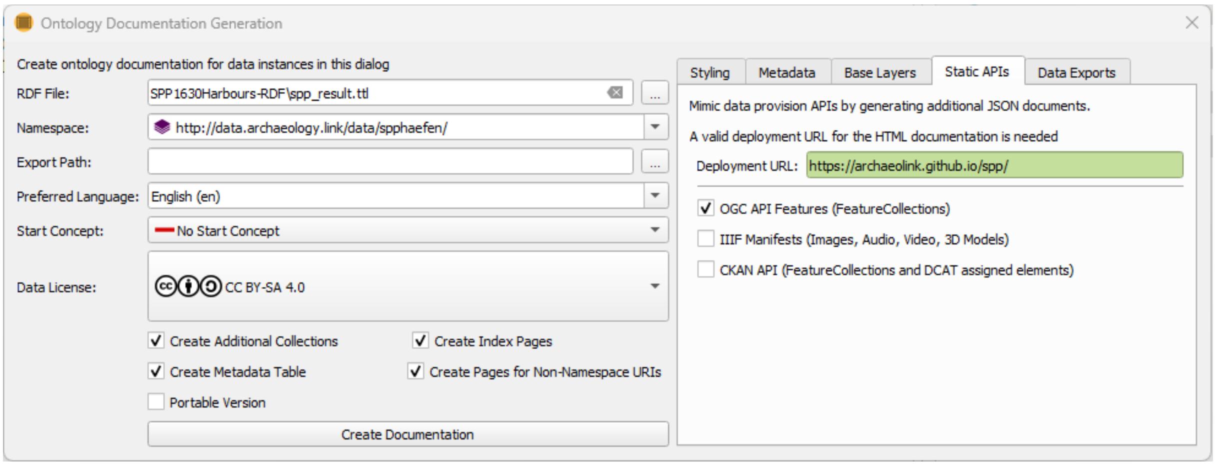
Fig. 5. Configuration of the static API in QGIS as an OGC API Features endpoint. A simple address to a Github page with an index.json document representing the OGC API Features landing page suffices for configuration.

4.3 Script Application

The SPARQL Unicorn ontology documentation script can be used in three different ways, all of which are available with the publication of this paper.

The first way of using the script is probably the easiest to be accessible: By installing the SPARQLing Unicorn QGIS Plugin[8] within QGIS (cf. Figure 5). The ontology documentation dialog will perform data namespace detection, allow the configuration of HTML templates and allows to choose the different static APIs which should be generated along with the data publication.

[8] https://plugins.qgis.org/plugins/sparqlunicorn/



Ontology Documentation Generation

Create ontology documentation for data instances in this dialog

RDF File: SPP1630Harbours-RDF\spp_result.ttl

Namespace: http://data.archaeology.link/data/spphaefen/

Export Path:

Preferred Language: English (en)

Start Concept: No Start Concept

Data License: CC BY-SA 4.0

☑ Create Additional Collections ☑ Create Index Pages

☑ Create Metadata Table ☑ Create Pages for Non-Namespace URIs

□ Portable Version

Mimic data provision APIs by generating additional JSON documents.

A valid deployment URL for the HTML documentation is needed

Deployment URL: https://archaeolink.github.io/spp/

☑ OGC API Features (FeatureCollections)

□ IIIF Manifests (Images, Audio, Video, 3D Models)

□ CKAN API (FeatureCollections and DCAT assigned elements)

Fig. 6. SPARQL Unicorn Ontology Documentation GUI in the SPARQLing Unicorn QGIS plugin. The SPP dataset has been loaded and the correct data namespace has been detected. Users can now customize their deployment, e.g. by choosing static APIs to be supported.

The second way to use the script is using the newly published python package sparqlunicorn-ontdoc[9] which can be installed directly using the PIP package manager for python packages. The package provides a command line interface which allows to use the script e.g. on your local computer.

The last way to use the script is as a Github Action[10] or as a Gitlab reusable workflow[11]. We expect many users to share their RDF data in repositories on Github or Gitlab and to regularly update their data deployments. Using a Github Action, users can trigger the ontology generation script including a deployment of the generated documentation on e.g. Github pages. This allows generated static GeoAPIs to be instantly used in applications such as QGIS without the need to set up an own webspace. The usage examples we show in the next section are set up in exactly this way and can be accessed on Github as showcases.

5 Usage examples available on Github

We provide a few usage examples which are results generated by the SPARQL Unicorn Ontology documentation script. Each usage example highlights specific features of the script (maybe important for the scientific question from a non-technical point of view) and will be discussed in the following.

- The *SPP Dataset*[12] and *Ancient Ports Dataset*[13]: Datasets of European Harbors with an inception year and references in the scientific literature as an example of a static OGC API Features API

[9] https://github.com/sparqlunicorn/sparqlunicornGoesGIS-ontdoc

[10] https://github.com/sparqlunicorn/sparqlunicornGoesGIS_testdata

[11] https://gitlab.com/sparqlunicorn/sparqlunicornGoesGIS_testdata

[12] https://github.com/archaeolink/SPP1630Harbours-RDF

[13] https://github.com/archaeolink/AncientPorts_RDF

- The ARS-LOD Dataset[14] and NAVISone Dataset[15] : Image datasets highlighting the generation of a static IIIF API exposed in JSON and directly viewable using the JavaScript Viewer Mirador[16]
- The *Campanian Ignimbrite Dataset*[17], the *Croton Dataset*[18], the *Linked Open Ogham Dataset*[19], and the *BB-5KBC Dataset*[20]: archaeological / geoscience datasets that have especially geospatial data which is important for (further) research. In this particular case, the SPARQL Unicorn documentation tool allows early-career-researchers with less experience in LOD technology to easily publish and visualize (FAIRify) their data, e.g. PhD-candidates Fiona Schenk with Campanian Ignimbrite findspots [31–34], Sophie C. Schmidt with ceramics data [35] (slides 25 ff.) or Stefanie Baars with data from coinage of Croton [36].

All repositories document their various API capabilities in their README files. Besides the accessibility of the contents via the Github page, the deployments can be shown and downloaded by accessing the repositories gh-pages branch.

6 Limitations

There are certain limitations using the ontology documentation script and subsequent deployments users need to be aware about. At the time of writing this publication, neither Github Pages nor Gitlab Pages[21] support content negotiation[22] [37] for their supported webspaces. Content negotiation is used and is recommended by several linked open data standards to request specific serializations of website contents sharing the same URL. The desired serialization is communicated in the Accept header of the HTTP request. Webspaces supporting content negotiation would allow for a single URL to act as an API endpoint for e.g. OGC API Features, a linked open data dump and a HTML serialization at the same time, whereas currently, the serialisation needs to be specified by adding index.EXTENSION to the URL as shown also in Figure 3.

Another limitation comes with the limited capabilities of the static geospatial data APIs. Because the generation of every possible outcome of an API call as static files would not be an advisable undertaking, suitable defaults which are commonly queries in software need to be determined and implemented. In contrast to the first limitation mentioned, this limitation is hard to overcome statically, but specifications of geodata standards could issue static-compatible API descriptions in the future to better supports use cases such as the ones described in this publication.

[14] F. Thiery et al. (RGZM + i3mainz): https://github.com/RGZM/ars-lod
[15] F. Thiery & A.W. Mees (LEIZA): https://github.com/archaeolink-lod/navisone
[16] Example in: https://archaeolink-lod.github.io/navisone/iiif/
[17] Fiona Schenk & Florian Thiery: https://github.com/Research-Squirrel-Engineers/campanian-ignimbrite-geo
[18] Stefanie Baars & Florian Thiery: https://github.com/Research-Squirrel-Engineers/croton-geo
[19] Florian Thiery: http://view.ogham.link/
[20] Sophie C. Schmidt & Florian Thiery: https://github.com/Research-Squirrel-Engineers/bb-5kbc
[21] https://gitlab.com/gitlab-org/gitlab/-/issues/26085
[22] https://datatracker.ietf.org/doc/html/rfc7231#section-5.3

7 Discussion

In the beginning of this paper, we set out to produce linked open usable data with a particular-lar emphasis of making geodata in RDF graphs which can only be hosted statically easier accessible for the geospatial data community without the need to use special preprocessing software or even the SPARQLing Unicorn Plugin itself. In that endeavour we were able to create linked open data sets from previously published static data dumps, which are due to their HTML content easier to explore by a wider audience, do not lose the semantic enrichment and interoperability provided by the knowledge graph, while at the same time being accessible by clients using standardized APIs to access domain-specific data. Once the data is encoded in RDF and in standardised vocabularies, the provision of aforementioned features is due to the documentation script not a difficult task anymore. This prompts the question whether this kind of data provision of an enriched knowledge graph combined with navigation, static APIs and community-specific data exports could become a recommended way to share RDF data results in the future.

8 Conclusions

We presented the SPARQL Unicorn ontology documentation script, which allows the generation of HTML and further data deployments enriched with static APIs to provide access to knowledge graph data for various research communities. The script is available as a plugin, standalone script and a GitHub Action and has been demonstrated to work under different data projects. This allows research and hobby projects with heterogeneous data to expose different kinds of data using the LOUD principles so that different communities may consume the data they need in the way they are used to using static APIs.

9 Future Work

Future extensions of the SPARQL Unicorn ontology documentation script will include more incorporations of standardized linked open data vocabularies, resulting in more serializations and views. In addition, HTML pages that aggregate data in knowledge graphs and the already existing HTML deployments per URI will be examined. For example, a dashboard view highlighting graph data, especially formatted graph data, could make research data more accessible. For geospatial data, we would like to investigate the incorporation of raster images and coverages, which are only beginning to be standardised in semantic web vocabularies at the time of writing. Improved integration and visualization of spatiotemporal data should also be a priority moving forward. Further future work to be incorporated are further common standard vocabularies in relation to geospatial data and the downlifting of RDF graph data to specific geospatial data schemas such as GML or INSPIRE.

Author’s Contact

Timo Homburg M.Sc.
Hochschule Mainz University of Applied Sciences
Lucy-Hillebrandt-Straße 2, 55128 Mainz
Timo.homburg@hs-mainz.de
Research Squirrel Engineers Network
Mainz, Germany
http://squirrel.link

Florian Thiery M.Sc.
Leibniz-Zentrum für Archäologie (LEIZA)
Ludwig-Lindenschmit-Forum 1, 55116 Mainz
florian.thiery@leiza.de
Research Squirrel Engineers Network
Mainz, Germany
http://squirrel.link

Bibliography

[1] Lanthaler, M.; Wood, D.; Cyganiak, R. RDF 1.1 Concepts and Abstract Syntax | W3C Recommendation Available online: http://www.w3.org/TR/rdf11-concepts/.

[2] Curé, O.; Blin, G. *RDF Database Systems: Triples Storage and SPARQL Query Processing*; Morgan Kaufmann, 2014.

[3] Biehl, M. *RESTful Api Design*; API-University Press, 2016; Vol. 3.

[4] Seaborne, A.; Harris, S. SPARQL 1.1 Query Language | W3C Recommendation Available online: http://www.w3.org/TR/sparql11-query/.

[5] Wilkinson, M.D.; Dumontier, M.; Aalbersberg, Ij.J.; Appleton, G.; Axton, M.; Baak, A.; Blomberg, N.; Boiten, J.-W.; da Silva Santos, L.B.; Bourne, P.E.; et al. The FAIR Guiding Principles for Scientific Data Management and Stewardship. *Scientific data* **2016**, *3*, 1–9.

[6] Sanderson, R. LOUD: Linked Open Usable Data Available online: https://linked. art/loud/.

[7] QGis, D.T. Quantum GIS Geographic Information System. *Open source geospatial Foundation project* **2011**, *45*.

[8] Thiery, F.; Schmidt, S.C.; Homburg, T.; Trognitz, M. The SPARQL Unicorn: An Introduction. *Squirrel Papers* **2020**, *2(1)*, #1, doi:10.5281/zenodo.3742185.

[9] Thiery, F.; Homburg, T. SPARQLing Unicorn QGIS Plugin (all versions). *Squirrel Papers* **2023**, *1(1)*, #5, doi:10.5281/zenodo.3786814.

[10] Thiery, F.; Homburg, T. QGIS - A SPARQLing Unicorn? Eine Einführung in Linked Open Geodata zur Integration von RDF in QGIS Plugins. *Squirrel Papers* **2020**, *2(1)*, #2, doi:10.5281/zenodo.3719127.

[11] Berners-Lee, T.; Fielding, R.T.; Masinter, L.M. Uniform Resource Identifier (URI): Generic Syntax Available online: https://www.rfc-editor.org/info/rfc3986.

[12] Barnaghi, P.; Tandy, J.; Homburg, T.; Brink, L. van den *Spatial Data on the Web Best Practices*; W3C, 2023.

[13] Nicholas J. Car; Timo Homburg; Matthew Perry; John Herring; Frans Knibbe; Simon J.D. Cox; Joseph Abhayaratna; Mathias Bonduel *OGC GeoSPARQL - A Geographic Query Language for RDF Data*; Open Geospatial Consortium, 2022.

[14] Car, N.J.; Homburg, T. GeoSPARQL 1.1: Motivations, Details and Applications of the Decadal Update to the Most Important Geospatial LOD Standard. *ISPRS International Journal of Geo-Information* **2022**, *11*, 117.

[15] Vrandečić, D.; Krötzsch, M. Wikidata: A Free Collaborative Knowledgebase. *Communications of the ACM* **2014**, *57*, 78–85.

[16] Vretanos, P.; others Web Feature Service Implementation Specification, Verson 1.1. 0. **2005**.

[17] Adida, B.; Birbeck, M.; McCarron, S.; Pemberton, S. RDFa in XHTML: Syntax and Processing. *Recommendation, W3C* **2008**, *7*, 14.

[18] Prud'hommeaux, E.; Carothers, G. RDF 1.1 Turtle | W3C Recommendation Available online: http://www.w3.org/TR/turtle/.

[19] Kellogg, G.; Champin, P.-A.; Longley, D. JSON-LD 1.1 | W3C Recommendation Available online: https://www.w3.org/TR/json-ld11/.

[20] Hyland, B.; Villazón-Terrazas, B.; Atemezing, G.A. *Best Practices for Publishing Linked Data*; W3C, 2014;

[21] Homburg, T.; Thiery, F. SPARQL Unicorn Ontology Documentation. *Squirrel Papers* **2023**, *5(2)*, #5, doi:10.5281/zenodo.8190763.

[22] Butler, H.; Daly, M.; Doyle, A.; Gillies, S.; Schaub, T.; Hagen, S. RFC 7946: The GeoJSON Format Available online: https://www.rfc-editor.org/info/rfc7946.

[23] D'Arcus, B.; Giasson, F. Bibliographic Ontology Specification. Specification Document, 4 November 2009. *Retrieved August* **2009**, *10*, 2011.

[24] Patashnik, O. BIBTEX 101. *TUGboat* **1984**, *15*, 269–273.

[25] Crickard III, P. *Leaflet. Js Essentials*; Packt Publishing Ltd, 2014;

[26] Brandes, U.; Eiglsperger, M.; Lerner, J.; Pich, C. Graph Markup Language (GraphML) 2013.

[27] Shafranovich, Y. Common Format and MIME Type for Comma-Separated Values (CSV) Files 2005.

[28] Hobona, G.; Simmons, S.; Masó-Pau, J.; Jacovella-St-Louis, J. OGC API Standards for the Next Generation of Web Mapping. *Abstracts of the ICA* **2023**, *6*, 91.

[29] Snydman, S.; Sanderson, R.; Cramer, T. The International Image Interoperability Framework (IIIF): A Community & Technology Approach for Web-Based Images. In Proceedings of the Archiving conference; Society for Imaging Science and Technology, 2015; Vol. 2015, pp. 16–21.

[30] Schwichtenberg, S.; Gerth, C.; Engels, G. From Open API to Semantic Specifications and Code Adapters. In Proceedings of the 2017 IEEE International Conference on Web Services (ICWS); IEEE, 2017; pp. 484–491.

[31] Thiery, F.; Schenk, F. Campanian Ignimbrite Geo Locations. *Squirrel Papers* **2023**, *5(2)*, #2, doi:10.5281/zenodo.10361309.

[32] Thiery, F.; Schenk, F. Modelling of Uncertainty in Geo Sciences Sites. *Squirrel Papers* **2023**, *5(1)*, #4, doi:10.5281/zenodo.10255259.

[33] Thiery, F.; Schenk, F. CI Site 52: Urluia (Romania). *Squirrel Papers* **2023**, *Research Squirrel Engineers, via @campanian-ignimbrite-geo*, http://fuzzy-sl.squirrel.link/data/cisite_52.

[34] Thiery, F.; Schenk, F. CI Site Instances Collection. *Squirrel Papers* **2023**, *Research Squirrel Engineers, via @campanian-ignimbrite-geo*, https://research-squirrel-engineers.github.io/campanian-ignimbrite-geo/Site_collection/index.html.

[35] Thiery, F.; Mees, A. Sharing (Linked) Open Data with Domain-Specific Data-Driven Community Hubs on the Example of the German National Research Data Infrastructure (NFDI) Consortium NFDI4Objects and the Data Hub Archaeology.Link. *Squirrel Papers* **2023**, *5(5)*, #4, doi:10.5281/zenodo.10260778.

[36] Thiery, F.; Schenk, F.; Baars, S. Dealing with Doubts: Site Georeferencing in Archaeology and in the Geosciences. *Squirrel Papers* **2023**, *5(1)*, #6, doi:10.5281/zenodo.10291889.

[37] Fielding, R.T.; Reschke, J. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content 2014.