Electronic Communications of the EASST
Volume 85 Year 2025

**deRSE25 - Selected Contributions of the 5th Conference for Research Software Engineering in Germany**

*Edited by: René Caspart, Florian Goth, Oliver Karras, Jan Linxweiler, Florian Thiery, Joachim Wuttke*

# Linked Open Usable Data Dumps: Maximizing reusability of linked open data dumps for a multitude of research audiences

Timo Homburg, Florian Thiery

**DOI:** 10.14279/eceasst.v85.2630

Electronic Communications of the EASST (https://eceasst.org).
Published by **Berlin Universities Publishing**
(https://www.berlin-universities-publishing.de/)

# Linked Open Usable Data Dumps: Maximizing reusability of linked open data dumps for a multitude of research audiences

Timo Homburg[1], Florian Thiery[2]

i3mainz – Institute for Spatial Information and Surveying Technology, Mainz University of Applied Sciences & Research Squirrel Engineers Network[1], Leibniz-Zentrum für Archäologie (LEIZA) & Research Squirrel Engineers Network[2]

Abstract: This publication introduces Linked Open Usable Data Dumps (LOUD Dumps), a way of deploying linked open data dumps with additional metadata, a consistent semantically enriched HTML deployment, and additional derived data exports provided using static APIs following the LOUD principles. We describe the need for this form of publication as a research data deployment, describe the exact contents of a LOUD Dump, and show an implementation using the SPARQL Unicorn Ontology Documentation Script that can produce the data dump we suggest. Finally, we provide example use cases, set the work in the context of a research data management workflow, and conclude the work with a discussion on further developments and the feasibility of implementation.

Keywords: Linked Open Data, Linked Open Usable Data, Linked Open Data Dump, Static API, Data Export, Research Data

## 1 Introduction

Publishing sustainable research data and providing appropriate access for various research communities challenges many players: researchers, research software engineers (RSEs) [BCG+12], standardization organizations, and data repositories. While data repositories being set up by various national initiatives, e.g., the German National Research Data Infrastructure (NFDI) [TBS+23, TMW+23, TM24] usually enforce a set of standardized metadata requirements and may help with the long-term storage of research data and their identification using appropriate long-term available identifiers, the possibility of hosting data alone does not guarantee that data is being reused and accepted by different research communities. The latter should be the goal of every published research dataset. Furthermore, data included in a research project is often not of a single type. Usually, many different data sets in different data formats are generated and separately published during the course of a research project.

One way to not sacrifice the coherency of published research datasets is to meaningfully describe them using a knowledge graph [FŞA+20]. Such a knowledge graph, ideally reusing a variety of controlled and standardized vocabularies defined in the Resource Description Framework (RDF) [Pan09], helps to connect research data in a formalized manner [TSB24, TT23]. However, to access knowledge graphs, one needs infrastructures, i.e., a graph database, a solution such as triple pattern fragments [VVH+16], or simply host the linked open data dump in a file repository.

The latter solution seemingly provides the least usability since users need to be proficient in Linked Open Data (LOD) technologies [HKR09] and process the data dump in a useful way for the task they set out to solve. In addition, a LOD dump by itself is not easily discoverable by introspection by other researchers without taking the initiative and tools to check its contents. While metadata describing the dump may exist, metadata is not, in all cases, enough for a researcher to decide whether a dataset is immediately useful. Metadata may help to get a researcher interested in the dataset. Still, the researcher needs to be able to access the dataset according to their abilities to determine its usefulness. The easier this process is for the researcher, the more likely the dataset will be reused.

At the end of a research project, providing a linked open data dump is the most likely and usually available option since a research project-dependent service can only be guaranteed long-term in rare cases without the help of a provided research service infrastructure. Hence, maximizing the usability of linked open data dumps for many research communities becomes a desirable prospect.

This publication, therefore, sets out to identify how linked open data dumps can be made usable and accessible for a variety of research communities, which may potentially be targeted by the data included in the linked open data dump at hand. The authors would like to define LOD dumps that provide added value not only for researchers dealing with linked open data but also for domain-specific researchers who are used to their own customized data formats and setups. The definition of a more broadly applicable LOD dump, a Linked Open Usable Dump (LOUD Dump), following the LOUD principles[New18], could enhance the reuse of linked open data and research data in particular. It could be a guideline to follow by ontology engineers and research software engineers when publishing linked open data without infrastructures in place.

To achieve this goal, we first review related work on linked open data dumps and linked open data publishing in Section 2. We then identify the needs of a set of research communities targeted for this publication and the typical requirements of said research communities in Section 3. Once we have identified the needs, we provide solutions to meet these needs by defining a linked open usable data dump (LOUD dump) in Section 4. To create said LOUD dumps, we present the SPARQLUnicorn Ontology Documentation Script[1], a proof of concept software that will help to reproduce the LOUD dumps we introduce in this publication. Next, Section 7.1 explains how LOUD Dumps can be incorporated into a research data publishing workflow. We present use cases to illustrate the usefulness of LOUD data dumps in Section 6. Finally, the paper closes by discussing the current limitations of this method of publication and future perspectives in further improving the constitution of LOUD dumps.

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

## 2 Related Work and Linked Open Data Publishing

Linked Open Data Dumps have existed as a form of publishing since the definition of linked open data [Hau11]. Different serializations of RDF [Bec04, CP14, FMG+13], with different advantages, allow for streaming RDF data, more expressive versions, more human-readable syntax, and even binary versions of RDF. We define a linked open data dump as follows:

Definition 1 A linked open data dump is the publication of linked open data encoded in RDF in one of its serializations as a file on the internet.

As a form of publication, linked open data dumps provide the means of a file with the necessary content to be further processed by people proficient in linked open data processing. Apart from the modeled data contents, the linked open data dump should follow a common set of rules to be considered good research data:

- Follow Best Practices of Linked Data Publishing as set out by W3C [HAV14]
- Follow the FAIR principles [ABB+16] of data publishing and vocabulary publication [GP20]
- Be published under an open license [KB15]
- Follow common practices of ontology design [GP09, BHR19] and best practices of implementing FAIR ontologies on the web [C+20]
- Include metadata about the publisher and contents of the research data they include
- Include a citation suggestion, e.g., as CFF [Dru19] for a Github repository or as metadata in the data dump, e.g. using the BIBO ontology [DSI12]
- Document their contents using an appropriate vocabulary such as VOID [AZCH11]

Compared to other forms of linked open data provision, e.g., SPARQL [C+13] endpoints or triple pattern fragments [VVH+16], linked open data dumps provide the easiest accessible form of data publishing and the form of publishing with the least maintenance requirements for the data publisher. Since LOD Dumps are files, they can be hosted on any webspace or an accessible hard drive. Still, they should ideally be addressable under the URI [BFM05] they are hosted at to allow them to be dereferenceable. This publishing method provides the least barriers to publishing research data as RDF for researchers since long-term hosting of files is usually provided for little or no money by services such as Zenodo or Github [PS15].

Beyond the accessibility of linked open data dumps, their contents are expected to follow the Best Practices for Publishing Linked Open Data [VAH14]. In particular, URIs are expected to be dereferenceable, at least in an RDF serialization. Ideally, more serializations, e.g., HTML serialization, are recommended for better human accessibility. When these practices are not followed, and RDF data is only presented using a SPARQL endpoint, the navigation of its contents may only be achieved by querying the SPARQL endpoint for information. URIs that are not published as linked open data but only remain present in the respective triple store database will fail to be resolvable [VSN+18].

Beyond these definitions of how linked open data can be hosted and linked open data dumps should be described, researchers have identified shortcomings with linked open data (dump) publishing, which they would like to address [PKF+20]. These shortcomings concern data quality, size, scalability, findability, and curation.

However, none of these approaches discuss forms of deployment of linked open data that are easily accessible by a general audience without prior knowledge of linked open data technologies—a significant number of people who might need to quickly browse and possibly assess the contents of a linked open data dump to determine its eligibility for further reuse. We anticipate changing this by proposing, implementing, and discussing a new form of deployment, the Linked Open Usable Data Dump (LOUD Dump).

In the following, we discuss the needs of researchers of this group for selected research communities. We then show how these needs can be addressed.

## 3 Requirements of research data communities

In this section, we try to explore the needs of research communities concerning data dumps, beginning from our own experiences as researchers in computer science, GIS, archaeology, and digital humanities towards a more formalized way considering the Linked Open Usable LOUD principles.

### 3.1 Typical needs of research communities

Research communities have different needs when using and reusing research data in their particular setup. However, in our daily research environment, we encounter the following needs and questions of researchers who are confronted with the question of whether to invest the time in investigating a data set:

1. Needs concerning the content of research data: Which data is accessible that I might reuse for my research?

2. Provision of data formats accepted by a research community: In which data formats is the data accessible for my research?

3. Correct and sufficient metadata descriptions of the data contents: Is the data sufficiently described so that I understand its context?

4. Access to data using APIs, common in the research community: How can I access the research data for my context and browse its contents?

These needs are research community-specific, i.e., each research community and user base will have differing requirements on data provision that people of the respective communities expect. Ideally, a linked open data dump could be extended to serve the interests of all research communities where it provides data, provided this can be determined from the RDF dump's contents alone. To abstract from these community-specific needs and provide a general solution for publishing a LOUD dump, we investigate the general requirements of linked open data publishing that may be valid for all research communities along the lines of the LOUD principles.

## 3.2 Investigating LOUD principles for data

Published linked open data, in the view of many research communities, should also follow the LOUD principles [New18], which advocate for the following important data accessibility aspects:

LOUD1 The correct abstraction layer to view the data for their target audience

LOUD2 Few barriers to entry for developers

LOUD3 Comprehensibility by introspection - Data discoverability without barriers

LOUD4 Documentation with working examples

LOUD5 Consistent patterns in data and access to different related data sets

The LOUD principles serve as the guideline for the definition of LOUD Dumps, which we advocate for in this publication. The question to be asked is:

> How should a linked open data dump be created to conform to the LOUD principles?

## 3.3 Case Study: Archaeology and GISScience

The authors, in particular, have experience and have surveyed people in archaeology and GISScience research communities. These communities identify with the needs we set out in Section 3.1, are usually no experts in Linked Open Data technologies, and have specific use cases in mind when working with and investigating data dumps. On top of that, we can illustrate typical specific kinds of data, which these two communities are usually working with as an example: For GISScience studies:

- Geodata in data formats such as GeoJSON, GML, KML, or Well-Known Text, as suggested by standardization organizations such as the OGC (GeoSPARQL)
- Visualizations of Geodata to interpret its spatial patterns

For Archaeology research:

- Data about archaeological artifacts (CIDOC-CRM)
- Linguistic Linked Open Data to model textual contents on artifacts (Ontolex-Lemon)

- Geodata and 3D data documenting excavation sites and scanned artifacts (GeoSPARQL)

We will consider these research community-specific needs when creating elements of the LOUD Dump in the next sections and revisit these needs in the application cases in a later section.

## 3.4 List of general requirements

Having investigated typical needs for research communities, the LOUD principles, and a case study in Archaeology and GISScience, we now create a list of general requirements we would like to fulfill when creating LOUD Dumps. We determine the criteria in this list from the perspective of a researcher interested in the contents of the LOUD dump and describe concrete implementation steps that need to be taken for the representation of a static data dump to become compatible with the LOUD principles. In essence, we translate the LOUD requirements to LOUD Dump requirements that we discuss in the following:

RQ1 The correct abstraction layer for the target audience [$LOUD1$]

- RQ1.1 Create one or many common data views per ontology vocabulary that fulfills the most common needs of the respective research community
- RQ1.2 In addition to RDF and HTML, provide data in common formats that fit the communities' expectations
- RQ1.3 Anticipate common data views from RDF vocabularies that the target audience needs for understanding the dataset's contents
- RQ1.4 Provide overviews on common data patterns
- RQ1.5 Use content negotiation to provide suitably many versions of data instances

RQ2 Few barriers to entry for developers [$LOUD2$]

- RQ2.1 Follow common static ways of data provision
- RQ2.2 Provide static APIs for data access
- RQ2.3 Document static APIs with API documentation
- RQ2.4 Document LOD and data exports with inferred metadata from the knowledge graph

RQ3 Comprehensibility by introspection [$LOUD3$]

- RQ3.1 Easy navigation and discoverability of the Linked Open Data Dump using at least one user interface
- RQ3.2 Augment the dataset with additional data useful for navigation
- RQ3.3 Provide metadata in such a way that navigation by external applications is simplified

RQ3.4 Provide options to issue queries against the data dump in the main HTML view

RQ4 Documentation with working examples [*LOUD*4]

RQ4.1 Highlight documented usage examples in the LOD dump

RQ4.2 Infer further examples for specific LOD dump components

RQ4.3 Provide web applications for example use cases

RQ5 Consistent patterns in data and access to different related data sets [*LOUD*5]

RQ5.1 Highlight and/or resolve links to other datasets and resources in HTML navigation

RQ5.2 Visualize consistent patterns in examples

Now that we have laid out the requirements that a LOUD dump needs to fulfill, we will discuss the components that a LOUD dump should have to fulfill these requirements.

## 4 Linked Open Usable Data Dumps

The last section categorized and exemplified the needs of selected research communities under the guidance of the LOUD principles. This section introduces LOUD dumps that accommodate the aforementioned needs. A traditional linked open data dump usually contains solely one RDF file containing the knowledge graph data. Commonly, the knowledge graph links to the research data published elsewhere on the web and describes it in its context. The RDF graph also contains metadata describing the published research data using standardized controlled vocabularies. In the following, elements that make these traditional linked open data dumps usable are introduced.

### 4.1 Automatic augmentation of RDF contents 3.4[RQ3.2]

Data published in an RDF data dump is not necessarily easy for users to navigate, even when using SPARQL queries, especially when using an RDF data dump. There are several reasons for this:

- RDF data instances are usually assigned an RDF or OWL [AH09] class, but often from a different namespace, whose definition may not be part of the RDF data dump itself
- RDF data instances are not necessarily grouped intuitively from the users' perspective. They can, therefore, not be easily consumed by users and data crawlers.
- The class hierarchy of RDF classes might be insufficiently modeled in the data dump itself, but might be present in referenced vocabularies

To make RDF data more accessible in this way, the following RDF instances are to be created automatically in LOUD dumps:

- Collection instances grouping data instances with the same class or the same property combinations
- Additional metadata triples to the knowledge graph that better or more consistently describe research data
- Vocabulary-specific triples to supplement existing data or to correct mistakes in data [PGS14]
- Resolution of linked vocabularies and integration of classtree-relevant linked open data triples, e.g., rdfs:subClassOf relations

It is important to mention that the triples added to the linked open data dump do not change its contents. They are mere triples that easily group existing data or integrate information into the LOUD dump already defined in a linked vocabulary. The increase in triples could generally be linear to the number of data instances in the data dump. That is, the impact of the newly generated triples is significant, but does not necessarily have to be included in the version of the data that is to be downloaded. A LOUD data dump version and a traditional LOD dump version could be linked to it.

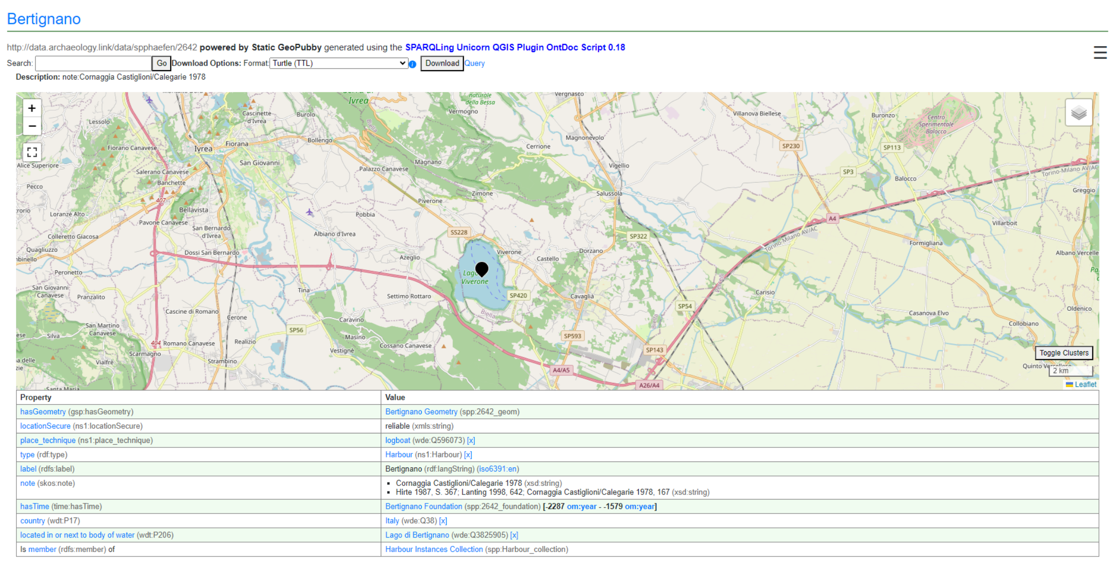
#### 4.1.1 Application example

The following example illustrates the usefulness of the automatic augmentation approach: Suppose a knowledge graph exposes a set of Harbour instances of rdf:type hr:Harbour. Furthermore, assume that the definition of class hr:Harbour is not part of the LOD dump. Finally, assume that one HTML rendering of each instance included in the LOD dump is created to allow dereference. Then, a user will not find an HTML page showing the complete collection of all hr:Harbour included in the dataset to navigate to them. Here, the augmentation step creates an RDF collection, e.g., a skos:Collection or a geo:FeatureCollection, encompassing all instances of the type hr:Habour. This step does not alter the graph's contents but gives additional means of navigating the graph in HTML and RDF. In that sense, the augmentation of special RDF triples like collections aids in lowering entry barriers for developers and users. It provides the correct abstraction layer for users intending to explore the linked open data dump, as envisioned in LOUD principle 3.

### 4.2 HTML Documentation [RQ1.1]

The HTML documentation of a linked open data dump follows the precedents set by Linked Open Data browsers such as LODLive [CMA12] or Pubby[2]. These linked open data browsers generate one HTML page per data instance, which is to be documented. Usually, these data instances are described by one or many data namespaces present in the linked open data dump. A typical data instance description page (e.g., Figure 1[3])

[2] https://github.com/cygri/pubby
[3] https://archaeolink.github.io/SPP1630Harbours-RDF/2642/index.html



Bertignano

http://data.archaeology.link/data/spphaefen/2642 **powered by Static GeoPubby** generated using the **SPARQLing Unicorn QGIS Plugin OntDoc Script 0.18**

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

**Description:** note:Cornaggia Castiglioni/Calegarie 1978

| Property | Value |
|---|---|
| hasGeometry (gsp:hasGeometry) | Bertignano Geometry (spp:2642_geom) |
| locationSecure (ns1:locationSecure) | reliable (xmls:string) |
| place_technique (ns1:place_technique) | logboat (wde:Q596073) [x] |
| type (rdf:type) | Harbour (ns1:Harbour) [x] |
| label (rdfs:label) | Bertignano (rdf:langString) (iso6391:en) |
| note (skos:note) | ▪ Cornaggia Castiglioni/Calegarie 1978 (xsd:string)<br>▪ Hirte 1987, S. 367; Lanting 1998, 642; Cornaggia Castiglioni/Calegarie 1978, 167 (xsd:string) |
| hasTime (time:hasTime) | Bertignano Foundation (spp:2642_foundation) **[-2287 om:year - -1579 om:year]** |
| country (wdt:P17) | Italy (wde:Q38) [x] |
| located in or next to body of water (wdt:P206) | Lago di Bertignano (wde:Q3825905) [x] |
| Is member (rdfs:member) of | Harbour Instances Collection (spp:Harbour_collection) |

Figure 1: A linked data page describing the location of the Harbour Bertignano from the SPP dataset. The harbor's location and inception time are sourced from a connected data instance.

consists of a table with the predicate and objects connected to the data instance and highlighting specific triple patterns in a separate page area. For example, an image link is resolved and displayed as an image. A LOUD Dump HTML instance page extends this idea with the following features:

- Representation of the data instance in Microformats (e.g., RDFa [AHBM15] or JSON-LD [KLC20]) [RQ3.1]
- Integration of connected data instances which constitute an essential part of the data instance to be described (e.g., show linked images or linked data instances with images) [RQ3.1]
- Exposure of each data instance through SEO optimization techniques [SSGK19] and citation hints embedded in the HTML template [RQ3.1]
- Allow for content negotiation [FR14] to deploy different serializations of the same research data to serve different client applications [RQ1.2]
- Suggest interesting dataset samples by generating overview pages for parts of the research data dump [RQ1.1]
- Provide visibility about dataset metrics and dataset contents in a machine-readable way (VOID descriptions [AZCH11]) and as an HTML rendering for interested researchers [RQ3.3]

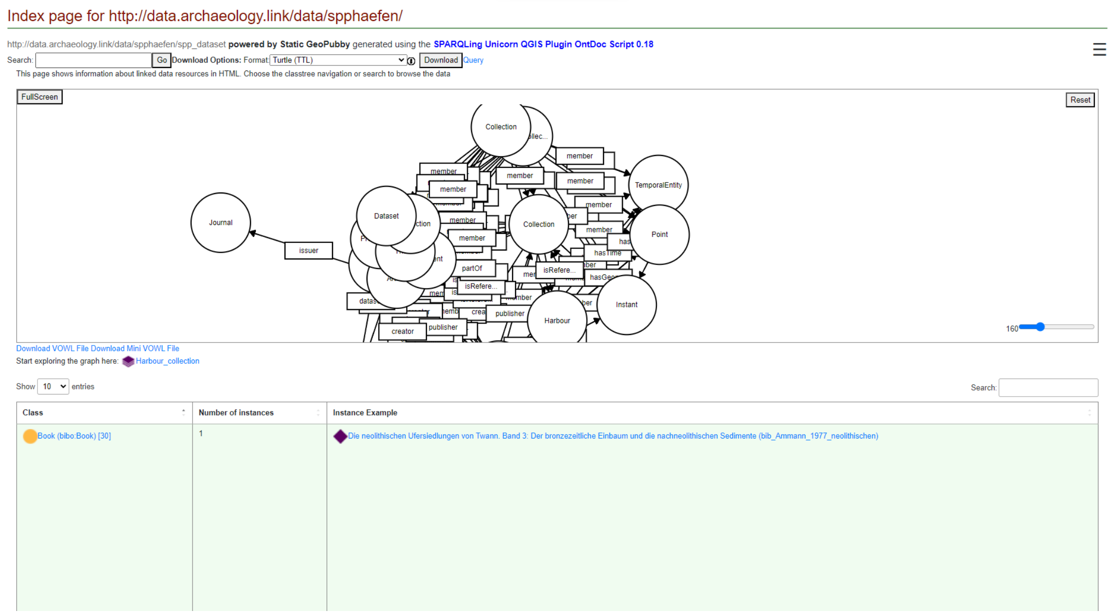
- Distinguish metadata triples connected to an instance from actual data instances and make these distinctions visible (Metadata statements are identified by metadata vocabularies, e.g., DublinCore) [RQ3.3]

Every so-generated HTML page should also be available in at least one RDF serialization so that the linked data instance may be resolved by web crawlers and by web browsers using content negotiation [SCF+09].

The HTML documentation thereby fulfills the following aspects described by the LOUD principles: - Comprehensibility by introspection: Every RDF instance is accessible using machine-readable means (e.g., web crawlers or RDF parsers) - Documentation with working examples: Visualization of instances of data as HTML for users for better accessibility and discoverability of the dataset

### 4.2.1 Index HTML pages [RQ1.4]

In addition to HTML pages, which document single data instances, overview pages document parts of the folder structure, which represent the HTML Deployment. For instance, suppose the data namespace http://www.example.com/data/ is to be rendered. Then, overview pages for the subfolders http://www.example.com/data/first and http://www.example.com/data/first/part1 are generated if these folders contain more than one entity.


Index page for http://data.archaeology.link/data/spphaefen/

http://data.archaeology.link/data/spphaefen/spp_dataset **powered by Static GeoPubby** generated using the **SPARQLing Unicorn QGIS Plugin OntDoc Script 0.18**

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

This page shows information about linked data resources in HTML. Choose the classtree navigation or search to browse the data

Download VOWL File Download Mini VOWL File

Start exploring the graph here: Harbour_collection

Show 10 entries Search:

| Class | Number of instances | Instance Example |
|---|---|---|
| ● Book (bibo:Book) [30] | 1 | ◆ Die neolithischen Ufersiedlungen von Twann. Band 3: Der bronzezeitliche Einbaum und die nachneolithischen Sedimente (bib_Ammann_1977_neolithischen) |

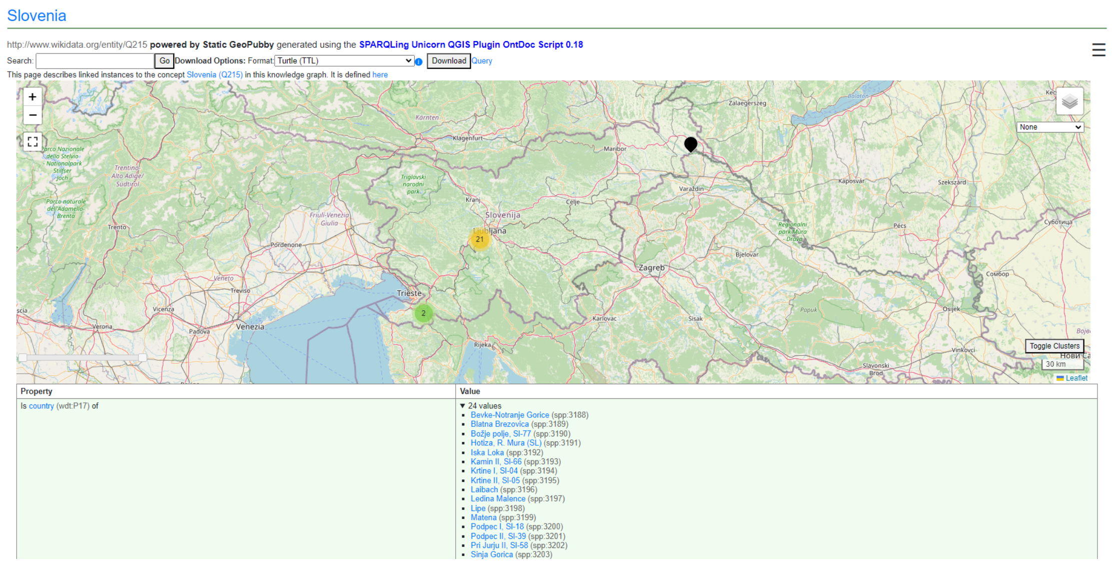
Figure 2: Index pages representing the contents of a folder. The page contains a table of occurring classes and instances, a small rendered graph, and, optionally, a highlighted concept.

Figure 2 shows an example of an index page that allows users to get an impression of the contents of a subgraph. The subgraph represented by the index page is available as an RDF file at the same URL and could, similar to the single page, be resolved using content negotiation or simply used as a smaller data dump to load by applications processing RDF data on the client-side. Index Pages thereby serve LOUD principle 3, comprehensibility by introspection, and by lowering barriers of entry to the dataset.

### 4.2.2 NonNS Pages [RQ1.3]

Non-namespace pages (NonNS pages) are HTML pages that represent RDF instances that are not included in the chosen data namespace of the RDF data dump but that may be of interest to a user to be shown in context in HTML.

For example, suppose an RDF dump represents a harbor location dataset. Every harbor instance in RDF contains a property relation ”country”, which points to a URI describing the country where the harbor is located. The URIs describing countries are in a different namespace (e.g., Wikidata) and are not rendered as HTML pages. Users may resolve these URIs to their definition pages, e.g., on Wikidata. Still, they might rather expect a page summarizing the occurrences of the harbors in the respective country value in the data dump deployment.


Slovenia

http://www.wikidata.org/entity/Q215 **powered by Static GeoPubby** generated using the **SPARQLing Unicorn QGIS Plugin OntDoc Script 0.18**

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

This page describes linked instances to the concept Slovenia (Q215) in this knowledge graph. It is defined here

| Property | Value |
|---|---|
| Is country (wdt:P17) of | ▼ 24 values<br>▪ Bevke-Notranje Gorice (spp:3188)<br>▪ Blatna Brezovica (spp:3189)<br>▪ Božje polje, SI-77 (spp:3190)<br>▪ Hotiza, R. Mura (SL) (spp:3191)<br>▪ Iska Loka (spp:3192)<br>▪ Kamin II, SI-66 (spp:3193)<br>▪ Krtine I, SI-04 (spp:3194)<br>▪ Krtine II, SI-05 (spp:3195)<br>▪ Laibach (spp:3196)<br>▪ Ledina Malence (spp:3197)<br>▪ Lipe (spp:3198)<br>▪ Matena (spp:3199)<br>▪ Podpec I, SI-18 (spp:3200)<br>▪ Podpec II, SI-39 (spp:3201)<br>▪ Pri Jurju II, SI-58 (spp:3202)<br>▪ Sinja Gorica (spp:3203) |

Figure 3: NonNS page highlighting all data instances in the SPP Harbour dataset which have a relation wdt:P17 (country) wd:Q215 (Slovenia).

Figure 3 shows a page such as this, which can be linked from every data instance page and may also be deployed in additional data formats for easy accessibility (e.g., here as a GeoJSON instance).

Users are empowered to discover groupings of RDF instances that are implicitly stated in the data dump but expressed in other non-rendered vocabularies. Since these NonNS pages may be rendered using the same rendering process as normal HTML pages, they may also expose an RDF version of the contents. The problem here is the missing URI for this particular page. An RDF version of these contents will have to render the contents described by a blank node or a generated unique RDF URI that is guaranteed not to conflict with further data in the data dump. Again, this aspect of the LOUD dumps serves the principle of comprehensibility by introspection. Creating overview pages that summarize relations of data included in the LOD dump to external vocabularies gives added perspectives on the dataset's purpose and structure.

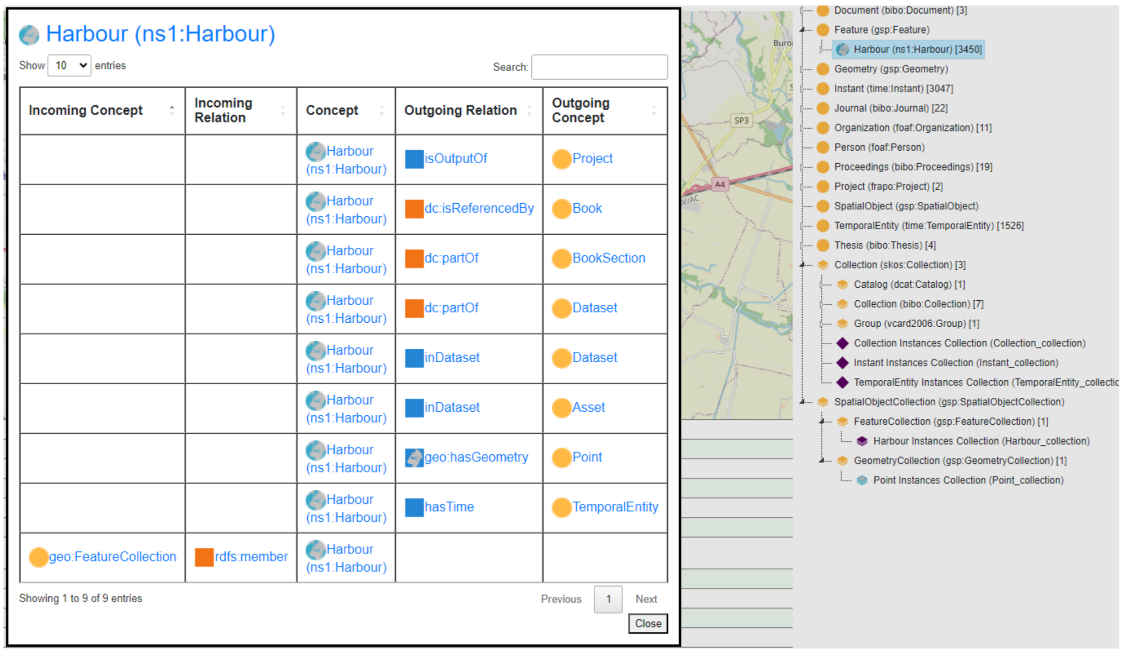
## 4.3 Navigation in RDF contents: Classtrees [RQ3.1]

To discover the contents of RDF data dumps, a way of navigation within the RDF data dump that is accessible to a variety of users has to be generated. A common way to do this is to generate one of many possible classtrees by inferring a class hierarchy from the class structure present in the knowledge graph and its connected vocabularies. Commonly, this is done by querying rdf:type and rdfs:subClassOf relations or its equivalent relations, if other vocabularies, e.g. the Wikidata Vocabulary, are used. A postprocessing algorithm then takes these results, removes loops that may be formed by the given rdfs:subClassOf relations, and finally returns one possible classtree which may be saved and rendered. While the VOID vocabulary does provide for the representation of class partitions, i.e., to express how many instances of a class are present, it does not provide opportunities to model an annotated classtree relation with metadata. The Classtree vocabulary[4] was created to capture, among others, the following types of classtree items and their relations.

- The kind of class that is described (e.g., A class describing geometries)
- The number of instances of the respective class
- The kind of relation that is expressed (e.g., collection relation)
- The kind of collections present

Using the classtree vocabulary, classtrees can, therefore, be formalized and, in terms of the HTML rendering, also visualized as an essential aid to navigate the Linked Open Data Dump as an interested researcher. The next essential information for the researcher is how the given class is connected to other instances in the knowledge graph. This information is provided using a class relation dialog, which can be seen in Figure 4. It allows the user to see the relations to and the relations from the given class. In addition, it is important to know the data schema delivered with a given class. Figure 5 shows an example data schema for a class in the knowledge graph. This view enables the user to make an informed decision about the usefulness of this dataset without downloading the data beforehand. Therefore, the navigation of a linked open data dump can be realized using the classtree as a navigation and two dialogs to identify the context of the respective tree elements and their most common properties.

[4] https://purl.org/vocab/classtree


Harbour (ns1:Harbour)

Show 10 entries Search:

| Incoming Concept | Incoming Relation | Concept | Outgoing Relation | Outgoing Concept |
|---|---|---|---|---|
| | | Harbour (ns1:Harbour) | isOutputOf | Project |
| | | Harbour (ns1:Harbour) | dc:isReferencedBy | Book |
| | | Harbour (ns1:Harbour) | dc:partOf | BookSection |
| | | Harbour (ns1:Harbour) | dc:partOf | Dataset |
| | | Harbour (ns1:Harbour) | inDataset | Dataset |
| | | Harbour (ns1:Harbour) | inDataset | Asset |
| | | Harbour (ns1:Harbour) | geo:hasGeometry | Point |
| | | Harbour (ns1:Harbour) | hasTime | TemporalEntity |
| geo:FeatureCollection | rdfs:member | Harbour (ns1:Harbour) | | |

Showing 1 to 9 of 9 entries Previous 1 Next

Close

- Document (bibo:Document) [3]
- Feature (gsp:Feature)
  - Harbour (ns1:Harbour) [3450]
- Geometry (gsp:Geometry)
- Instant (time:Instant) [3047]
- Journal (bibo:Journal) [22]
- Organization (foaf:Organization) [11]
- Person (foaf:Person)
- Proceedings (bibo:Proceedings) [19]
- Project (frapo:Project) [2]
- SpatialObject (gsp:SpatialObject)
- TemporalEntity (time:TemporalEntity) [1526]
- Thesis (bibo:Thesis) [4]
- Collection (skos:Collection) [3]
  - Catalog (dcat:Catalog) [1]
  - Collection (bibo:Collection) [7]
  - Group (vcard2006:Group) [1]
  - Collection Instances Collection (Collection_collection)
  - Instant Instances Collection (Instant_collection)
  - TemporalEntity Instances Collection (TemporalEntity_collectio
- SpatialObjectCollection (gsp:SpatialObjectCollection)
  - FeatureCollection (gsp:FeatureCollection) [1]
    - Harbour Instances Collection (Harbour_collection)
  - GeometryCollection (gsp:GeometryCollection) [1]
    - Point Instances Collection (Point_collection)

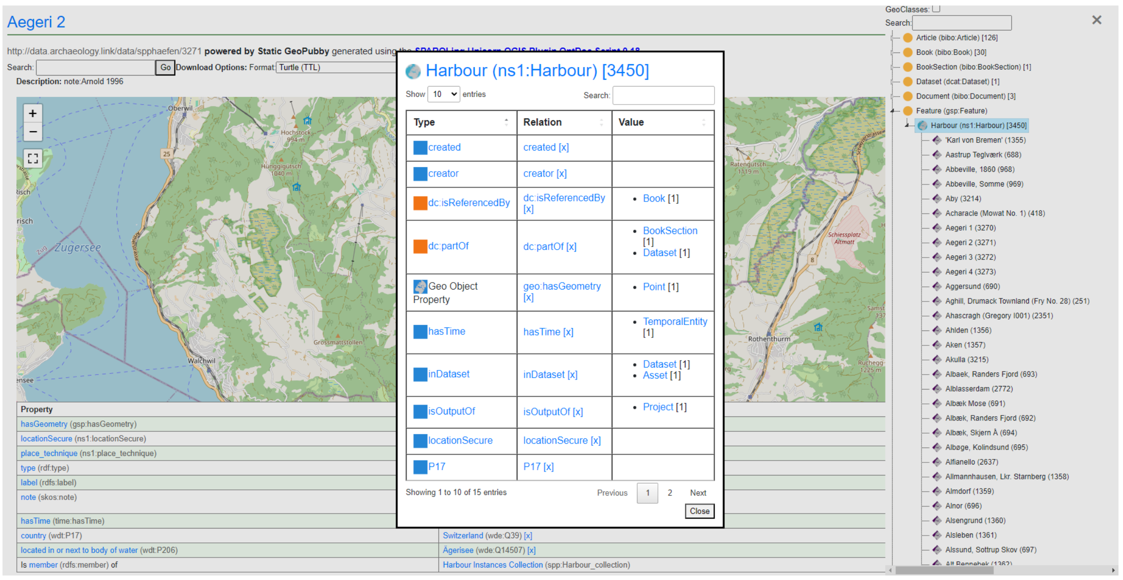
Figure 4: Relations of instances of the class Harbour in the SPP dataset. The instances are connected to research projects, scientific literature, time instances, and geometries, and are parts of dataset definitions. Also, users may learn that there are collections of Harbours in the knowledge graph, which may be useful to observe.

## 4.4 Linked Open Data Access [RQ2.2]

Client applications must access linked open data stored in LOD dumps. These applications can load the whole LOD dump, subsequently parse its contents, and finally allow the user to query for the data they are requesting. This clearly allows other applications, such as linked open data databases, to periodically update data based on the contents of the linked open data dump. In the case that these infrastructures are not available to the user, another solution can be to query the linked open data dump using e.g., JavaScript libraries developed as a result of discussions of the RDFJS community group works[5]. This way is shown in Figure 6 using the example of the SPP dataset[6].
For LOUD dumps, this means ensuring that data is accessible to a user unfamiliar with linked open data tools, e.g., on a homepage, and making it clear to a human and a machine how a linked open data dump can be accessed. For the latter, the VOID vocabulary [AZCH11] provides a way to describe linked open data dumps with metadata so that algorithms can classify and assess them for suitability for their specific use cases. As shown by [GS11], VOID descriptions may even serve as parts in a SPARQL Endpoint Federation scenario.

[5] https://www.w3.org/community/rdfjs/

[6] https://archaeolink.github.io/SPP1630Harbours-RDF/sparql.html?endpoint=https://archaeolink.github.io/SPP1630Harbours-RDF/



Aegeri 2

http://data.archaeology.link/data/spphaefen/3271 **powered by Static GeoPubby** generated using

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

**Description:** note:Arnold 1996

| Property | |
|---|---|
| hasGeometry (gsp:hasGeometry) | |
| locationSecure (ns1:locationSecure) | |
| place_technique (ns1:place_technique) | |
| type (rdf:type) | |
| label (rdfs:label) | |
| note (skos:note) | |
| hasTime (time:hasTime) | |
| country (wdt:P17) | Switzerland (wde:Q39) [x] |
| located in or next to body of water (wdt:P206) | Ägerisee (wde:Q14507) [x] |
| Is member (rdfs:member) of | Harbour Instances Collection (spp:Harbour_collection) |

Harbour (ns1:Harbour) [3450]

Show 10 entries Search:

| Type | Relation | Value |
|---|---|---|
| 🟦 created | created [x] | |
| 🟦 creator | creator [x] | |
| 🟧 dc:isReferencedBy | dc:isReferencedBy [x] | • Book [1] |
| 🟧 dc:partOf | dc:partOf [x] | • BookSection [1]<br>• Dataset [1] |
| Geo Object Property | geo:hasGeometry [x] | • Point [1] |
| 🟦 hasTime | hasTime [x] | • TemporalEntity [1] |
| 🟦 inDataset | inDataset [x] | • Dataset [1]<br>• Asset [1] |
| 🟦 isOutputOf | isOutputOf [x] | • Project [1] |
| 🟦 locationSecure | locationSecure [x] | |
| 🟦 P17 | P17 [x] | |

Showing 1 to 10 of 15 entries Previous 1 2 Next

Close

GeoClasses:

Search:

- Article (bibo:Article) [126]
- Book (bibo:Book) [30]
- BookSection (bibo:BookSection) [1]
- Dataset (dcat:Dataset) [1]
- Document (bibo:Document) [3]
- Feature (gsp:Feature)
  - Harbour (ns1:Harbour) [3450]
    - 'Karl von Bremen' (1355)
    - Aastrup Teglværk (688)
    - Abbeville, 1860 (968)
    - Abbeville, Somme (969)
    - Aby (3214)
    - Acharacle (Mowat No. 1) (418)
    - Aegeri 1 (3270)
    - Aegeri 2 (3271)
    - Aegeri 3 (3272)
    - Aegeri 4 (3273)
    - Aggersund (690)
    - Aghill, Drumack Townland (Fry No. 28) (251)
    - Ahascragh (Gregory I001) (2351)
    - Ahlden (1356)
    - Aken (1357)
    - Akulla (3215)
    - Albaek, Randers Fjord (693)
    - Alblasserdam (2772)
    - Albæk Mose (691)
    - Albæk, Randers Fjord (692)
    - Albæk, Skjern Å (694)
    - Albøge, Kolindsund (695)
    - Alfianello (2637)
    - Allmannhausen, Lkr. Starnberg (1358)
    - Almdorf (1359)
    - Alnor (696)
    - Alsengrund (1360)
    - Alsleben (1361)
    - Alssund, Sottrup Skov (697)
    - Alt Rennebek (1362)

Figure 5: Dataschema of the class Harbour in the SPP dataset. The window lists all properties associated with the class and its assigned values for reference.

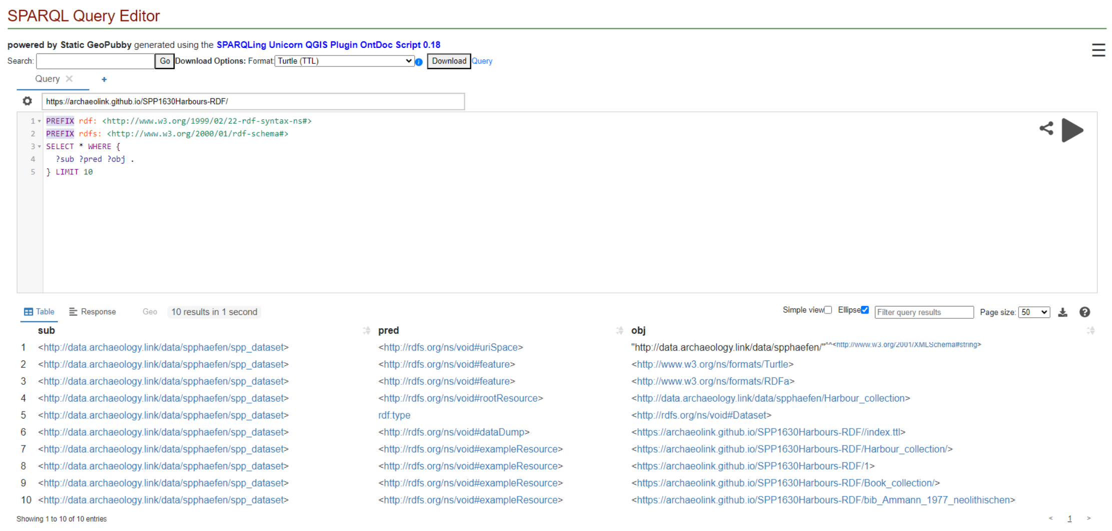
For a user, VOID dumps, e.g., the one visualized in Figure 7, can provide certain useful information.

- Statistics about the contents of the data
- Statistics about the vocabularies referenced in the data dump
- Indications about the contents of the dataset

For the latter, dataset contents may be generated by analyzing vocabularies and mapping them to well-known concepts associated with the respective vocabulary. For example, if the GeoSPARQL vocabulary is detected, a describing concept could be ”geospatial data”.

## 4.5 Data exports [RQ1.2]

Parts of the knowledge graph are more suitable for consumption by software written by specific research communities if they are served in specific data formats. For example, geodata in a knowledge graph which is encoded using the GeoSPARQL vocabulary [CH22] is best served to a community of geospatial researchers in one of the OGC-recommended geodata formats such as Geography Markup Language (GML) [Por07], Keyhole Markup Language (KML) [NL14] or as GeoJSON. Therefore, a LOUD dump should provide data exports beyond RDF that may be useful to other research communities. To identify the possibilities to export such data, the graph needs to be analyzed for triple patterns, which:



SPARQL Query Editor

powered by Static GeoPubby generated using the SPARQLing Unicorn QGIS Plugin OntDoc Script 0.18

Search: Download Options: Format: Turtle (TTL)

https://archaeolink.github.io/SPP1630Harbours-RDF/

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT * WHERE {
  ?sub ?pred ?obj .
} LIMIT 10
```

10 results in 1 second

|  | sub | pred | obj |
|---|---|---|---|
| 1 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#uriSpace> | "http://data.archaeology.link/data/spphaefen/"^^<http://www.w3.org/2001/XMLSchema#string> |
| 2 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#feature> | <http://www.w3.org/ns/formats/Turtle> |
| 3 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#feature> | <http://www.w3.org/ns/formats/RDFa> |
| 4 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#rootResource> | <http://data.archaeology.link/data/spphaefen/Harbour_collection> |
| 5 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | rdf:type | <http://rdfs.org/ns/void#Dataset> |
| 6 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#dataDump> | <https://archaeolink.github.io/SPP1630Harbours-RDF//index.ttl> |
| 7 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#exampleResource> | <https://archaeolink.github.io/SPP1630Harbours-RDF/Harbour_collection/> |
| 8 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#exampleResource> | <https://archaeolink.github.io/SPP1630Harbours-RDF/1> |
| 9 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#exampleResource> | <https://archaeolink.github.io/SPP1630Harbours-RDF/Book_collection/> |
| 10 | <http://data.archaeology.link/data/spphaefen/spp_dataset> | <http://rdfs.org/ns/void#exampleResource> | <https://archaeolink.github.io/SPP1630Harbours-RDF/bib_Ammann_1977_neolithischen> |

Showing 1 to 10 of 10 entries

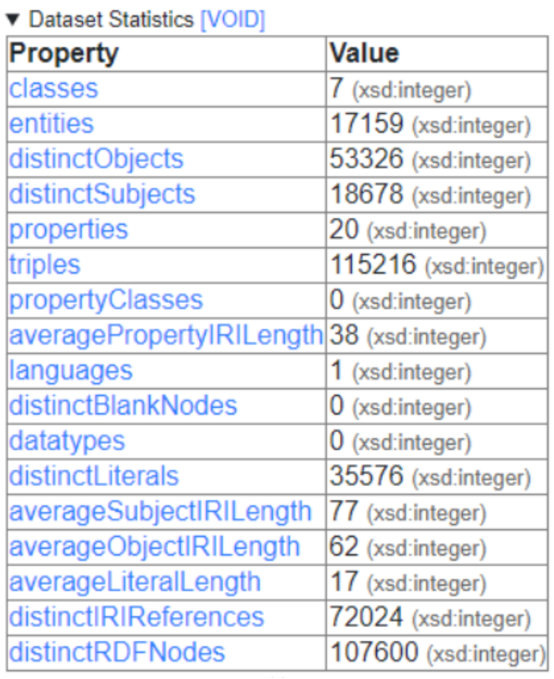
Figure 6: Special HTML page providing access to the RDF dump using a SPARQL query window. The data dump is loaded, using JavaScript, into the web browser.

- Describe a certain knowledge domain
- Are described in a standardized, well-known vocabulary
- Are linked to items in the data namespace that the linked open data dump describes

In this way, the linked open data dump does not serve as an RDF dump only, but also provides added value for software and researchers. Due to its interconnectedness, certain assumptions must be made to export specific data from a knowledge graph. Correct partial knowledge graphs suitable for exports in different export formats must be identified. For example, only the data immediately connected to the to-be-exported entity could be considered.

## 4.6 Static APIs for LOUD dumps [RQ2.2]

Another step in providing the correct abstraction layer to view data for specific target audiences can be to serve files as downloadable artifacts and provide access to them using APIs, which are commonly used by communities interacting with said data. Using common APIs helps discover research data and provides a convenient way for field practitioners to access it. For example, Geospatial vector data is commonly expected to be served by APIs following the specification of OGC API Features [HSMJ23]. While a LOUD Dump cannot provide an API in the sense of a web service provided by a server, since it is only a data dump, it is possible to analyze said common APIs for compatibility to create so-called static APIs. Static APIs are collections of documents (often JSON) organized to mimic the results of a dynamic API, which generates results on the fly when a request reaches the server.



▼ Dataset Statistics [VOID]

| Property | Value |
|---|---|
| classes | 7 (xsd:integer) |
| entities | 17159 (xsd:integer) |
| distinctObjects | 53326 (xsd:integer) |
| distinctSubjects | 18678 (xsd:integer) |
| properties | 20 (xsd:integer) |
| triples | 115216 (xsd:integer) |
| propertyClasses | 0 (xsd:integer) |
| averagePropertyIRILength | 38 (xsd:integer) |
| languages | 1 (xsd:integer) |
| distinctBlankNodes | 0 (xsd:integer) |
| datatypes | 0 (xsd:integer) |
| distinctLiterals | 35576 (xsd:integer) |
| averageSubjectIRILength | 77 (xsd:integer) |
| averageObjectIRILength | 62 (xsd:integer) |
| averageLiteralLength | 17 (xsd:integer) |
| distinctIRIReferences | 72024 (xsd:integer) |
| distinctRDFNodes | 107600 (xsd:integer) |

Figure 7: Example of VOID statistics visualized for one dataset.

#### 4.6.1 Static API Documentation [RQ2.3]

When a static API is generated from an existing API, its functionality is often reduced, but it still provides the core functionality the API seeks to provide. All APIs exposed by the LOUD dump should be described using an OpenAPI description [PR22]. The advantage of OpenAPI is that it is an established standard and can be documented in static HTML and using tools such as Swagger. In this sense, static APIs of LOUD dumps again become accessible in machine-readable and human-accessible ways. By documenting the static API, the LOUD dump removes barriers to data access, which may stem from problems in understanding the API's functionality for developers.

#### 4.6.2 Static API Applications in JavaScript [RQ4.3]

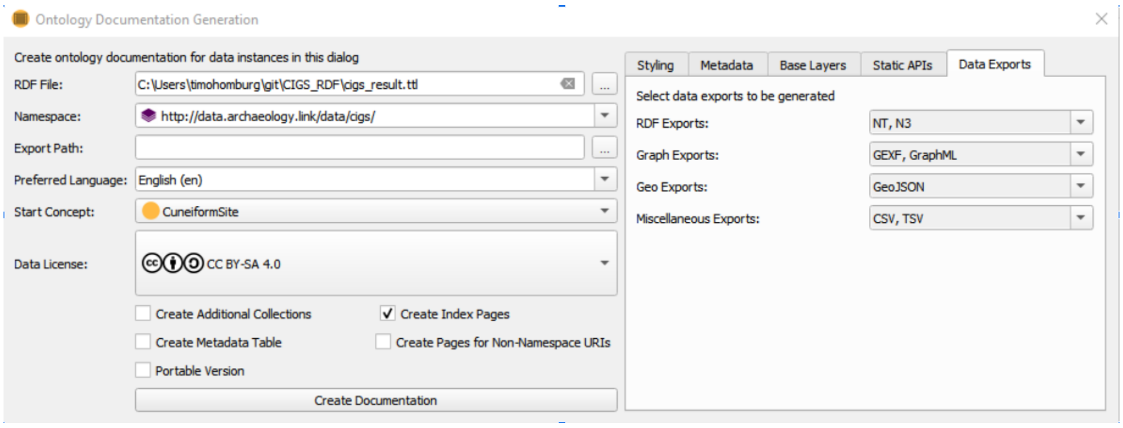
Web applications may also access static APIs. This provides the option of adding standard applications to display the contents of static APIs in the HTML deployment part of the LOUD dump. For instance, the contents of (static) IIIF APIs may be visualized using IIIF viewers such as Tify [7] or OpenSeaDragon[8]. This allows users of different communities to immediately access a view of the data they are most familiar with in the web browser.

[7] https://github.com/tify-iiif-viewer/tify
[8] https://openseadragon.github.io/

This concludes the presentation of the central elements included in a LOUD dump. The next section will discuss a proof of concept implementation creating these kinds of dumps: The SPARQLUnicorn Ontology Documentation Script in Version 0.17[HT24].

## 5 SPARQLUnicorn Ontology Documentation Script

The SPARQLUnicorn Ontology Documentation Script[9] allows for the generation of aforementioned LOUD dumps. It is available as a Python package, a Github Action and Gitlab CI Workflow, and a standalone Python script and as part of the SPARQLing Unicorn QGIS Plugin [TH20, THS+21] (cf. Figure 8) In any of the aforementioned forms, the



Create ontology documentation for data instances in this dialog

| | |
|---|---|
| RDF File: | C:\Users\timohomburg\git\CIGS_RDF\cigs_result.ttl |
| Namespace: | http://data.archaeology.link/data/cigs/ |
| Export Path: | |
| Preferred Language: | English (en) |
| Start Concept: | CuneiformSite |
| Data License: | CC BY-SA 4.0 |

☐ Create Additional Collections ✓ Create Index Pages
☐ Create Metadata Table ☐ Create Pages for Non-Namespace URIs
☐ Portable Version

Select data exports to be generated

| | |
|---|---|
| RDF Exports: | NT, N3 |
| Graph Exports: | GEXF, GraphML |
| Geo Exports: | GeoJSON |
| Miscellaneous Exports: | CSV, TSV |

Figure 8: SPARQLUnicorn Ontology Documentation Script as part of the SPARQLing Unicorn QGIS plugin. A documentation generation requires at least one data URI as an input parameter.

script receives at least the location of the RDF data dump to document and its data namespace URI as input parameters and generates the data dump in the way that is described throughout this section.

### 5.1 Data Dump Generation

To generate the final knowledge graph, the data dump undergoes a multiple-stage analysis shown in . At first, the data dump is augmented with further information, as described in Section 4.1. Next, in a first stage, the following statistics about the data dump are created:

- The namespace(s) that are used to model the data in question
- The vocabulary used in the linked open data dump
- The classes and properties used in the linked open data dump

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

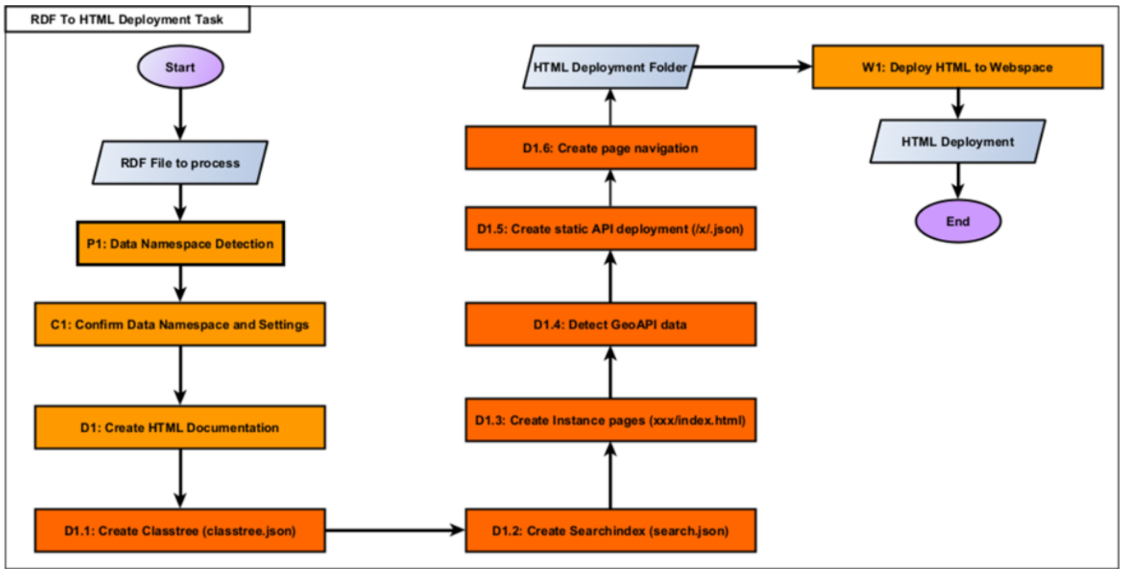

RDF To HTML Deployment Task
Start
RDF File to process
P1: Data Namespace Detection
C1: Confirm Data Namespace and Settings
D1: Create HTML Documentation
D1.1: Create Classtree (classtree.json)
D1.2: Create Searchindex (search.json)
D1.3: Create Instance pages (xxx/index.html)
D1.4: Detect GeoAPI data
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 9: Example data dump generation process for a geodata RDF dump generated using the SPARQLing Unicorn Ontology Documentation script

- The typical usages of vocabularies as RDF data concepts

The latter information is used to infer one of many possible classtrees according to the classtree vocabulary[10]. This classtree may be used for navigation in the HTML rendering, making the dataset more discoverable for humans interested in its contents.

The second stage generates the classtree from the SPARQL query displayed in Listing 1.

Listing 1: SPARQL query querying the class hierarchy of a linked open data dump. Vocabularies may vary if no standard vocabularies like owl, rdf and rdfs are use to describe class relations.

```
SELECT DISTINCT ?subject ?label ?supertype WHERE {
   { ?individual rdf:type ?subject . }
   UNION { ?subject rdf:type owl:Class  . }
   UNION { ?subject rdf:type rdfs:Class . }
   OPTIONAL { ?subject rdfs:subClassOf ?supertype . }
   OPTIONAL { ?subject rdfs:label ?label. filter(langMatches(lang(?label),"en")) }
   OPTIONAL { ?subject rdfs:label ?label }.
}
```

The query returns a set of triples indicating classes and their subclass relations. In post-processing, these results are created as a tree structure, then exported as a JSON-LD file in the classtree vocabulary.

The third stage generates the individual HTML deployment pages as described in Section 4.2 and index and NonNS pages. In this process, information regarding the nature of the individual data instances is collected to determine, e.g., whether instances of a class contain geometries and may be treated as such in further processing steps.

[10] http://purl.org/vocab/classtree/

Data exports for individual data instances that may be resolved using, e.g., content negotiation, are also generated in this step.

The fourth stage generates data exports and static APIs after the HTML page deployments have finished. These data exports and HTML deployments concern many data instances and are therefore generated in this stage. This stage also includes generating possible JavaScript applications that directly use the generated static APIs.

The fifth and last stage enhances the given classtree with the information gained by processing the individual data instances to generate the HTML deployments. Information about the class schema, its type according to the classtree library, and its connections to other classes is generated and saved as JSON files.

At the end of the previous stages, the program produced a selection of folders with HTML, data exports, and API data. This HTML deployment can be viewed on the local computer in the standalone script. In the case of the GitHub Action, this deployment is directly deployed to the respective GitHub page of the repository, allowing immediate access to the deployment on the web.

## 6 Application Cases

We present the following application cases, illustrating the publication of linked open data dumps and highlighting specific important aspects of these datasets.

### 6.1 SPP Dataset

The SPP dataset is a database of the locations of ancient harbors in Europe. It describes the harbor's inception, location, and the types of boats that were used there during its existence. Every harbor is linked to at least one publication from which the knowledge graph information is sourced and its dependent elements, such as authors (persons), journals, and publishers. The knowledge graph was modeled using the following vocabularies:

- The GeoSPARQL vocabulary [CH22] to describe the geospatial features of the harbour locations
- The OWL time ontology [PH06] to model the inception and abandonment of the harbours
- The BIBO ontology[11] to describe publications
- The Ontology for Units of Measurements [RAT13] to model units of time
- The FOAF vocabulary [BM15] to describe persons acting as authors of the publications linked to the dataset
- Metadata vocabularies such as VOID [AZCH11] and DublinCore [WKLW98] to model different aspects of the publication date

[11] https://www.dublincore.org/specifications/bibo/

- A customized vocabulary to model attributes in the dataset not covered by any other reusable vocabulary

Given the structure of this dataset, the SPARQLUnicorn ontology documentation script will provide a standard HTML deployment as described in Section 5 and additional data exports for the geodata included in the Harbour dataset.

### 6.1.1 RDF augmentation

The SPP dataset consists of one main class, the Harbour class, which describes Harbour instances. For this class, the Ontology documentation script creates a geo:FeatureCollection, since it detects that the class is associated with geometries modelled in the GeoSPARQL vocabulary. Further collections are created for the publication types and persons (authors) found in the knowledge graph. An SPP Harbour is connected to a waterway, a country, and a ship type. Since all types are represented in Wikidata and not in the local data dump, NonNS pages are generated for all of them.

### 6.1.2 Data Exports

The documentation script creates data exports according to input parameters and specific graph patterns in the knowledge graph. In this application case, we assume that the user suggested a GeoJSON export of the harbors in question and a CSV export of the harbor's metadata. The documentation script will also create vCard files for the individual people represented as authors in the knowledge graph, since they are defined as authors of the dataset. Finally, the bibliographic information describing the publications, which provides the basis for the bibliographic dataset, is saved in BibTeX [Grä07] files per publication instance, each and per publication collection type.

### 6.1.3 APIs

The SPP data dump exposes three different APIs stemming from its source data.

- A static OGC API Features [HSMJ23] implementation[12] (cf. Figure 10) to include geospatial data in GeoJSON [BDD+16] into GIS applications such as QGIS [KDPD16] easily

- A static CardDAV API [Dab11] allows the retrieval of the person's information, which is stored as the authors of the publications in the knowledge graph

- A static CKAN API [Win13], (cf. Figure 11) which allows the download GeoJSON files, vCard files [Per11], and, e.g., CSV files [Sha05], which can be generated as data exports from parts of the knowledge graph

[12] https://archaeolink.github.io/SPP1630Harbours-RDF/index.json

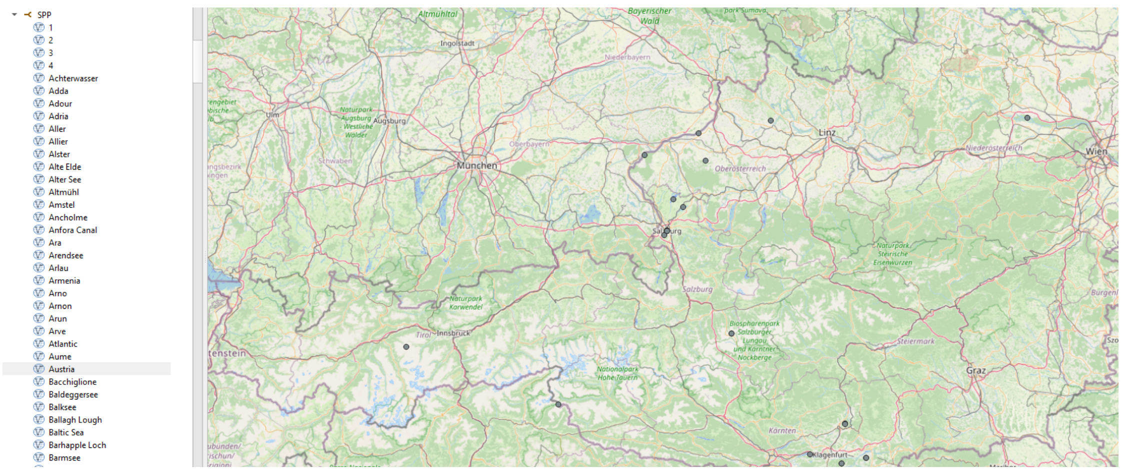


Figure 10: Static OGC API Features API used in QGIS to highlight the Harbours of Austria as sourced from the annotations in the knowledge graph.

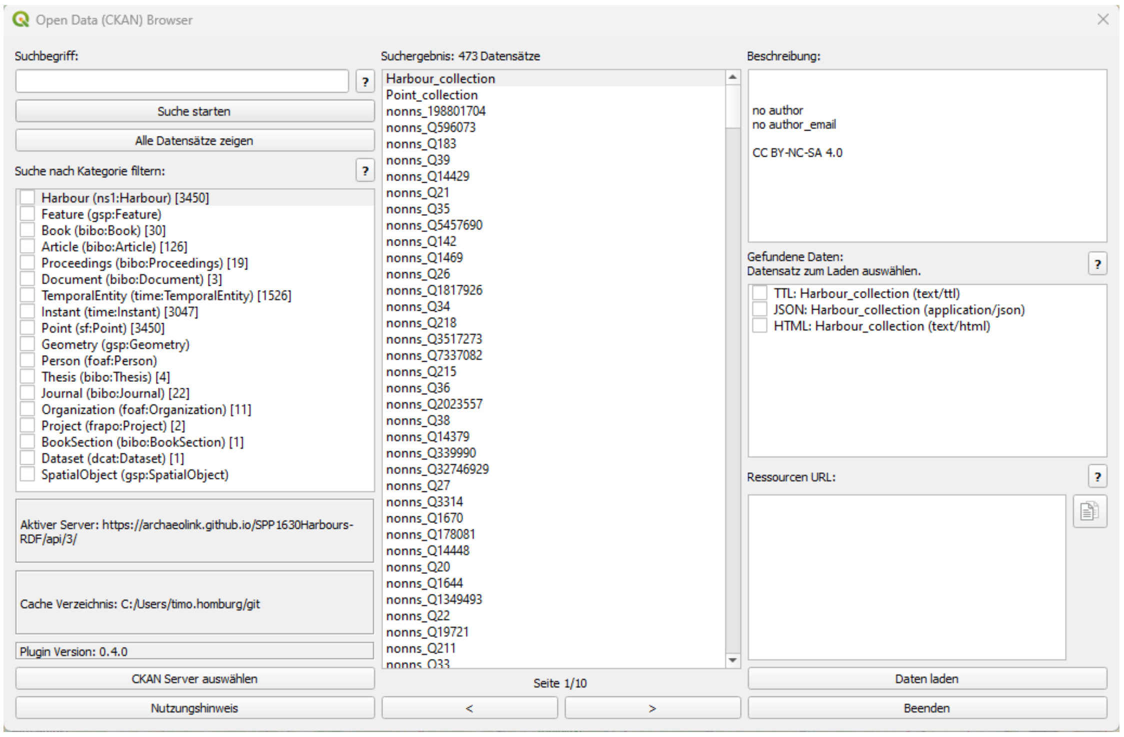
## 6.2 ARS-LOD Dataset

The ARS-LOD dataset[13] represents annotated 3D scans of African Red Slip Ware (ARS) created in the ARS3D (African Red Slip Ware digital) project [TVR+23, TRDB21]. Annotations are represented twofold: As Well-Known Text coordinates in a 3D space and as cut-out images in JPG format [Wal92]. The knowledge graph further classifies the given image cutouts by the symbols represented in the image cutouts and the kind of pottery represented in the 3D scans.

### 6.2.1 APIs

Since the ARS-LOD dataset mainly includes links to media data, it makes sense to expose said media using an IIIF API [SSC15]. Hence, a static IIIF API is exposed, which allows the retrieval of the images hosted at their respective locations. A static API, in comparison to a traditional IIIF API, may expose data as a download but may not serve operations such as rotation, clipping, or resizing. However, if web annotations of images were included in the knowledge graph, it would be possible to clip the annotation images when creating the linked open data dump and include them in the IIIF API. Also, it is possible to highlight annotations included in the IIIF manifest in image viewers. For the purposes of a linked open usable dump, the functionality to download and view image data is sufficient, though. For a user exploring that dataset, the rotation, resizing, and clipping functionality could also be provided on the client side, possibly using JavaScript directly in the web browser. Other IIIF-capable clients might also provide client-side support for image manipulation as shown in Figure 12[14] . In the case of the ARS-LOD dataset, the following parts are generated:

[13] https://situx.github.io/ars-lod/
[14] https://situx.github.io/ars-lod/iiif/


Open Data (CKAN) Browser

Suchbegriff:

Suchergebnis: 473 Datensätze

Beschreibung:

Suche starten

Alle Datensätze zeigen

Suche nach Kategorie filtern:

- Harbour (ns1:Harbour) [3450]
- Feature (gsp:Feature)
- Book (bibo:Book) [30]
- Article (bibo:Article) [126]
- Proceedings (bibo:Proceedings) [19]
- Document (bibo:Document) [3]
- TemporalEntity (time:TemporalEntity) [1526]
- Instant (time:Instant) [3047]
- Point (sf:Point) [3450]
- Geometry (gsp:Geometry)
- Person (foaf:Person)
- Thesis (bibo:Thesis) [4]
- Journal (bibo:Journal) [22]
- Organization (foaf:Organization) [11]
- Project (frapo:Project) [2]
- BookSection (bibo:BookSection) [1]
- Dataset (dcat:Dataset) [1]
- SpatialObject (gsp:SpatialObject)

Aktiver Server: https://archaeolink.github.io/SPP1630Harbours-RDF/api/3/

Cache Verzeichnis: C:/Users/timo.homburg/git

Plugin Version: 0.4.0

CKAN Server auswählen

Nutzungshinweis

Harbour_collection
Point_collection
nonns_198801704
nonns_Q596073
nonns_Q183
nonns_Q39
nonns_Q14429
nonns_Q21
nonns_Q35
nonns_Q5457690
nonns_Q142
nonns_Q1469
nonns_Q26
nonns_Q1817926
nonns_Q34
nonns_Q218
nonns_Q3517273
nonns_Q7337082
nonns_Q215
nonns_Q36
nonns_Q2023557
nonns_Q38
nonns_Q14379
nonns_Q339990
nonns_Q32746929
nonns_Q27
nonns_Q3314
nonns_Q1670
nonns_Q178081
nonns_Q14448
nonns_Q20
nonns_Q1644
nonns_Q1349493
nonns_Q22
nonns_Q19721
nonns_Q211
nonns_Q33

Seite 1/10

< >

no author
no author_email

CC BY-NC-SA 4.0

Gefundene Daten:
Datensatz zum Laden auswählen.

- TTL: Harbour_collection (text/ttl)
- JSON: Harbour_collection (application/json)
- HTML: Harbour_collection (text/html)

Ressourcen URL:

Daten laden

Beenden

Figure 11: Static CKAN API using a CKAN API Explorer Plugin in QGIS. CKAN can provide data exports for users to download in a unified interface.

- A static IIIF API that exposes the images included in the knowledge graph
- JavaScript IIIF viewer Mirador[15]

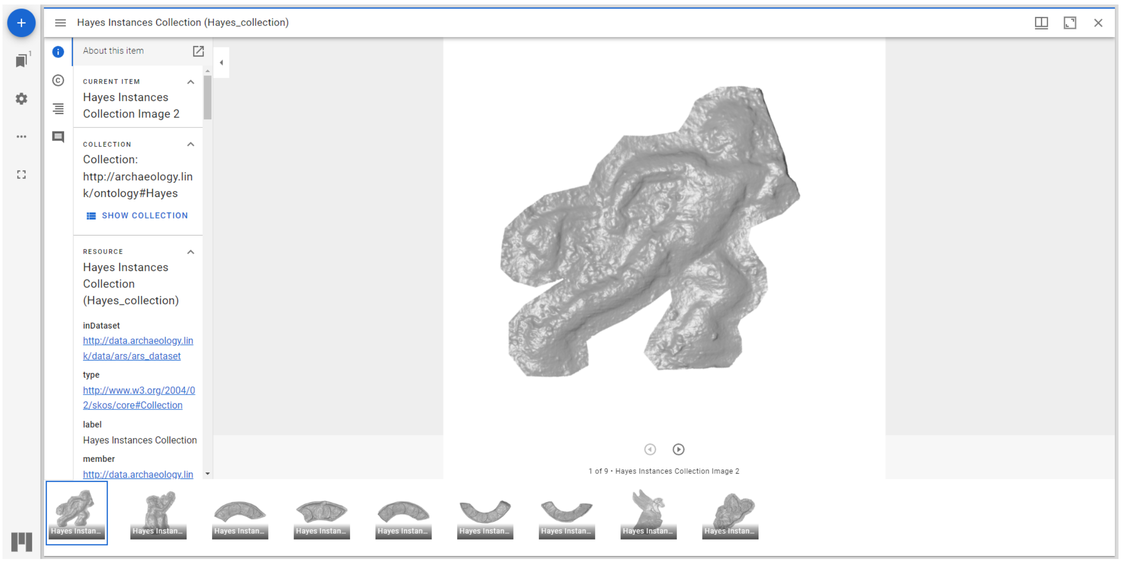
## 7 Discussion

After introducing LOUD dumps and some application cases illustrating possible usages, this section discusses the possible benefits of this type of publication.

### 7.1 LOUD Dumps in a research data publishing workflow

At the end of a research project, publishing the research data and explaining how the research data connect to each other is usually a requirement by funding agencies and a desirable research project goal. While the kind of data to be published and its metadata are usually already part of a research project data management plan, exactly how data should be published to maximize its exposure is an aspect that is not necessarily defined in greater detail during this necessary planning step. Depending on the research project at hand, oftentimes, these considerations are not seen as essential, and the wish is for

[15] https://projectmirador.org/


Hayes Instances Collection (Hayes_collection)

About this item

CURRENT ITEM
Hayes Instances Collection Image 2

COLLECTION
Collection: http://archaeology.link/ontology#Hayes

SHOW COLLECTION

RESOURCE
Hayes Instances Collection (Hayes_collection)

inDataset
http://data.archaeology.link/data/ars/ars_dataset

type
http://www.w3.org/2004/02/skos/core#Collection

label
Hayes Instances Collection

member
http://data.archaeology.lin

1 of 9 • Hayes Instances Collection Image 2

Figure 12: Mirador JavaScript Viewer highlighting contents exposed by the static IIIF API of the ARS-LOD dataset

a standard tool that can publish given research data in a way that is optimal or at least sufficient to fulfill the requirements of the research funding agency. However, even the research data management plan, if published, could be modeled in RDF and set in relation to the research data generated in the research project. In this way, LOUD dumps can be part of the research data publishing in the following ways:

1. Planning research data before a research project starts, and documenting this plan also in RDF
2. Documenting the changes to the research data management plan in RDF
3. Set results of the research data in relation to the initial research data management plan in RDF
4. Use the knowledge graph of the research data results as a LOUD Dump to better communicate the results achieved in the research project

In the end, the research project in data becomes a self-contained unit described as a knowledge graph, which can be visualized as a LOUD data dump.

### 7.2 Discoverability of research data

Research data needs to be discoverable by humans and machines to increase the chances of reusing data. This discoverability can be seen when the dataset is indexed and listed in certain catalog registries. The VOID description allows for such indexing. Once research data has been discovered, the researcher has to evaluate it. The HTML serialization enables a first glance at the data contents, their interconnectedness, and the various methods of access provided to assess the linked open data dump. Through the possibility of navigation through the dataset using classtrees and collection instances, chances of a better understanding of the structures and components of the dataset, especially for non-IT experts, are increased.

### 7.3 Sustainable publishing of research data

Sustainable publishing for research data means publishing research data in a way that guarantees its reuse over a long period of time. While the Linked Open Data Dump approach cannot guarantee that media linked in the knowledge graph are hosted permanently for a long(er) period of time, the provision of an HTML deployment that can be hosted on a webspace can be sustainable with just two primary considerations:

- Availability of the webspace - currently not a problem with the availability of Gitlab and Github instances
- Aging of the HTML, JavaScript, and CSS used in the HTML deployment to the extent that web browsers do not support the displaying of essential components anymore

While the first aspect is an issue of data provision, it is likely to assume that webspace is still easy to come by in the foreseeable future. The second issue is a concern but could be mitigated by continuously upgrading the documentation script and its HTML templates. Then, keeping the HTML deployment up-to-date is only a matter of recreating it dynamically and in a timely manner.

### 7.4 LOUD Dumps as research results

Publishing research data as a knowledge graph can provide coherence of different research data across different media. The knowledge graph can be the basis of static deployments of research data results, out of which the ontology documentation shown in this paper can be one aspect. Therefore, the authors think that, independent of publishing research data, users would like to explore which content can be found in a research project. This information, traditionally only found as project homepages or possibly incompletely described in research publications, can, therefore, be formalized and serve as the documentation of research projects and point to their results published in long-term repositories.

### 7.5 Limitations

LOUD Dumps face certain kinds of limitations, which should be expressed in the following. The first limitation cannot be overcome since it is inherent to all statically hosted content. SPARQL queries with dynamic results and API calls involving server-side processing cannot be fulfilled in a statically hosted dump. Secondly, the usefulness of the LOUD Dump depends on the software's ability to generate the dump. In other words, the software needs to implement at least one profile for visualization per standardized vocabulary. While the current implementation, the SPARQLing Unicorn Ontology Documentation plugin, includes support for many standard vocabularies, it is not exhaustive. Hence, users need to be prepared that their favorite view on data, which depends on a certain vocabulary, might need additional programming effort to be supported. This also means that a customized vocabulary can, without additional effort, only be visualized in a very generic manner. In addition. LOUD Dumps cannot replace a web application that uses the LOD dump for specific research project-related purposes. While data visualization can provide views for many different data types, these visualizations may not be appropriate for all kinds of use cases a user has in mind.

## 8 Conclusions

This work has explored how linked open data dumps can be made more accessible and reusable for various research communities. To do that, we extended the LOUD principles with requirements and concrete implementation specifics that must be fulfilled to achieve this goal. The result, LOUD Dumps, include the rendering of RDF resources as HTML pages, the augmentation of the RDF dump with the generation of new collection classes, the generation of a machine- and human-readable classtree navigation and the provision of parts of the knowledge graph as data exports using commonly used (static) APIs that research communities expect. We applied the linked open data dumps in two different use cases in geospatial data and archaeology domains, highlighting static versions of geodata and image data deployment using static versions of the IIIF and OGC API Features APIs. Finally, we discussed LOUD dumps' advantages for publishing research data and their current limitations. While an establishment of LOUD data dumps is just a proposal for the time being, we hope that this concept or an extension of it can bridge the gap between the linked open data community and research communities interested in the synergies that linked open data provides in the ways they are used to work in.

### 8.1 Future Work

With the concept of LOUD dumps in mind, obvious further extensions would be to examine more research communities, the ways they interact with research data, and which data formats are standard. Only the investigation of research data practices can provide further inspiration for generating valuable views on data. This also includes investigating user feedback from these communities on currently existing generic views. Also, future work should discover further vocabularies and APIs commonly used to describe data and work on either discovering or defining new static APIs from existing non-static APIs. Only by doing so can we provide standard access to research data supported by already-established software.

Another aspect is the definition, improvement, standardization, publication, application, and subsequent recognition of more semantic web vocabularies in research communities. Once these processes have been finalized, scripts may also detect them, create LOUD dumps, and properly expose them. Finally, one aspect to be worked on is integrating and registering several linked open data dumps to cross-connect research data on a LOUD dump level, make these results better visible in the HTML deployments, and allow a cross-data querying on the client side in JS and other languages. It also stands to reason whether LOUD dumps can be published similarly to SOLID pods, i.e., be the cornerstone for the publication of research data in general.

## Bibliography

[ABB$^+$16] M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas et al. The FAIR Guiding Principles for scientific data management and stewardship. Scientific data 3(1), 2016.
doi:10.1038/sdata.2016.18

[AH09] G. Antoniou, F. v. Harmelen. Web Ontology Language: OWL. Pp. 91–110, 2009.
doi:10.1007/978-3-540-92673-3_4
https://doi.org/10.1007/978-3-540-92673-3_4

[AHBM15] B. Adida, I. Herman, M. Birbeck, S. McCarron. RDFa Core 1.1 - Third Edition. Mar. 2015.
https://www.w3.org/TR/2015/REC-rdfa-core-20150317/

[AZCH11] K. Alexander, J. Zhao, R. Cyganiak, M. Hausenblas. Describing Linked Datasets with the VoID Vocabulary. W3C note, W3C, Mar. 2011.
https://www.w3.org/TR/2011/NOTE-void-20110303/

[BCG$^+$12] R. Baxter, N. Chue Hong, D. Gorissen, J. Hetherington, I. Todorov. The Research Software Engineer. Sept. 2012. Digital Research 2012 ; Conference date: 10-09-2012 Through 12-09-2012.
https://www.research.ed.ac.uk/files/65195747/DR2012_12_1_.pdf

[BDD$^+$16] H. Butler, M. Daly, A. Doyle, S. Gillies, T. Schaub, S. Hagen. The GeoJSON Format. Technical report 7946, Aug. 2016.
doi:10.17487/RFC7946
https://www.rfc-editor.org/info/rfc7946

[Bec04] D. Beckett. RDF/XML Syntax Specification (Revised). Feb. 2004.
https://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/

[BFM05] T. Berners-Lee, R. T. Fielding, L. M. Masinter. Uniform Resource Identifier (URI): Generic Syntax. Technical report 3986, Jan. 2005.
doi:10.17487/RFC3986
https://www.rfc-editor.org/info/rfc3986

[BHR19] M. Bravo, L. F. Hoyos Reyes, J. A. Reyes Ortiz. Methodology for ontology design and construction. Contaduría y administración 64(4), 2019.
doi:10.22201/fca.24488410e.2020.2368

[BM15] D. Brickley, L. Miller. FOAF Vocabulary Specification 0.99. 2014. Namespace Document, 2015.
http://xmlns.com/foaf/spec/

[C+13] W. W. W. Consortium et al. SPARQL 1.1 Overview. Mar. 2013.
https://www.w3.org/TR/2013/REC-sparql11-overview-20130321/

[C+20] G. Cota et al. Best practices for implementing fair vocabularies and ontologies on the web. Applications and practices in ontology design, extraction, and reasoning 49:39, 2020.

[CH22] N. J. Car, T. Homburg. GeoSPARQL 1.1: Motivations, Details and Applications of the Decadal Update to the Most Important Geospatial LOD Standard. ISPRS International Journal of Geo-Information 11(2), 2022.
doi:10.3390/ijgi11020117
https://www.mdpi.com/2220-9964/11/2/117

[CMA12] D. V. Camarda, S. Mazzini, A. Antonuccio. LodLive, exploring the web of data. In Proceedings of the 8th International Conference on Semantic Systems. I-SEMANTICS ’12, p. 197–200. Association for Computing Machinery, New York, NY, USA, 2012.
doi:10.1145/2362499.2362532
https://doi.org/10.1145/2362499.2362532

[CP14] G. Carothers, E. Prud’hommeaux. RDF 1.1 Turtle. W3C recommendation, W3C, Feb. 2014.
https://www.w3.org/TR/2014/REC-turtle-20140225/

[Dab11] C. Daboo. CardDAV: vCard Extensions to Web Distributed Authoring and Versioning (WebDAV). RFC 6352, Aug. 2011.
doi:10.17487/RFC6352
https://www.rfc-editor.org/info/rfc6352

[Dru19] S. Druskat. The Citation File Format (CFF): Why, what, how? 2019.
https://citation-file-format.github.io

[DSI12] B. Dimić Surla, M. Segedinac, D. Ivanović. A BIBO ontology extension for evaluation of scientific research results. In Proceedings of the Fifth Balkan Conference in Informatics. Pp. 275–278. 2012.
doi:10.1145/2371316.237137

[FMG+13] J. D. Fernández, M. A. Martínez-Prieto, C. Gutiérrez, A. Polleres, M. Arias. Binary RDF representation for publication and exchange (HDT). Journal of

Web Semantics 19:22–41, 2013.
doi:10.1016/j.websem.2013.01.002
https://www.sciencedirect.com/science/article/pii/S1570826813000036

[FR14] R. T. Fielding, J. Reschke. Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. RFC 7231, June 2014.
doi:10.17487/RFC7231
https://www.rfc-editor.org/info/rfc7231

[FŞA+20] D. Fensel, U. Şimşek, K. Angele, E. Huaman, E. Kärle, O. Panasiuk, I. Toma, J. Umbrich, A. Wahler, D. Fensel et al. Introduction: what is a knowledge graph? Knowledge graphs: Methodology, tools and selected use cases, pp. 1–10, 2020.
doi:10.1007/978-3-030-37439-6_1

[GP09] A. Gangemi, V. Presutti. Ontology Design Patterns. In Staab and Studer (eds.), Handbook on Ontologies. Pp. 221–243. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
doi:10.1007/978-3-540-92673-3_10
https://doi.org/10.1007/978-3-540-92673-3_10

[GP20] D. Garijo, M. Poveda-Villalón. Best practices for implementing fair vocabularies and ontologies on the web. In Applications and practices in ontology design, extraction, and reasoning. Pp. 39–54. IOS Press, 2020.
doi:10.48550/arXiv.2003.13084

[Grä07] G. Grätzer. Bibtex. New York, NY, 2007.
doi:10.1007/978-0-387-68852-7_16
https://doi.org/10.1007/978-0-387-68852-7_16

[GS11] O. Görlitz, S. Staab. SPLENDID: SPARQL endpoint federation exploiting VOID descriptions. P. 13–24, 2011.
doi:10.5555/2887352.2887354

[Hau11] M. Hausenblas. Utilising linked open data in applications. In Proceedings of the International Conference on Web Intelligence, Mining and Semantics. WIMS '11. Association for Computing Machinery, New York, NY, USA, 2011.
doi:10.1145/1988688.1988697
https://doi.org/10.1145/1988688.1988697

[HAV14] B. Hyland, G. A. Atemezing, B. Villazón-Terrazas. Best Practices for Publishing Linked Data. Jan. 2014.
https://www.w3.org/TR/2014/NOTE-ld-bp-20140109/

[HKR09] P. Hitzler, M. Krotzsch, S. Rudolph. Foundations of semantic web technologies. Chapman and Hall/CRC, 2009.
https://www.semantic-web-book.org

[HSMJ23] G. Hobona, S. Simmons, J. Masó-Pau, J. Jacovella-St-Louis. OGC API Standards for the Next Generation of Web Mapping. Abstracts of the ICA 6:91, 2023.
doi:10.5194/ica-abs-6-91-2023
https://ica-abs.copernicus.org/articles/6/91/2023/

[HT24] T. Homburg, F. Thiery. sparqlunicorn/sparqlunicornGoesGIS-ontdoc: Version 0.17. Mar. 2024.
doi:10.5281/zenodo.10780476
https://doi.org/10.5281/zenodo.10780476

[KB15] M. Khayyat, F. Bannister. Open data licensing: More than meets the eye. Information Polity 20(4):231–252, 2015.
doi:10.3233/IP-150357
https://journals.sagepub.com/doi/abs/10.3233/IP-150357

[KDPD16] G. Kurt Menke, G. Dr. Richard Smith Jr., L. Pirelli, G. Dr. John Van Hoesen. Mastering QGIS. Community experience distilled. Packt Publishing, 2016.
https://books.google.de/books?id=jYdcDgAAQBAJ

[KLC20] G. Kellogg, D. Longley, P.-A. Champin. JSON-LD 1.1. July 2020.
https://www.w3.org/TR/2020/REC-json-ld11-20200716/

[New18] D. Newbury. LOUD: Linked Open Usable Data and linked. art. In 2018 CIDOC Conference. Pp. 1–11. 2018.

[NL14] D. Nolan, D. T. Lang. Keyhole Markup Language. Pp. 581–618, 2014.
doi:10.1007/978-1-4614-7900-0_17
https://doi.org/10.1007/978-1-4614-7900-0_17

[Pan09] J. Z. Pan. Resource Description Framework. In Staab and Studer (eds.), Handbook on Ontologies. Pp. 71–90. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.
doi:10.1007/978-3-540-92673-3_3
https://doi.org/10.1007/978-3-540-92673-3_3

[Per11] S. Perreault. vCard Format Specification. RFC 6350, Aug. 2011.
doi:10.17487/RFC6350
https://www.rfc-editor.org/info/rfc6350

[PGS14] M. Poveda-Villalón, A. Gómez-Pérez, M. C. Suárez-Figueroa. Oops!(ontology pitfall scanner!): An on-line tool for ontology evaluation. International Journal on Semantic Web and Information Systems (IJSWIS) 10(2):7–34, 2014.

[PH06] F. Pan, J. R. Hobbs. Time Ontology in OWL. W3C working draft, W3C 1(1):1, 2006.

[PKF+20] A. Polleres, M. R. Kamdar, J. D. Fernández, T. Tudorache, M. A. Musen. A more decentralized vision for Linked Data. Semantic Web 11(1):101–113, 2020.
doi:10.3233/SW-190380
https://journals.sagepub.com/doi/abs/10.3233/SW-190380

[Por07] C. Portele. Opengis® geography markup language (GML) encoding standard. Version 3.2. 1. 2007.
https://www.ogc.org/de/standards/gml/

[PR22] J. S. Ponelat, L. L. Rosenstock. Designing APIs with Swagger and OpenAPI. Simon and Schuster, 2022.
https://www.manning.com/books/designing-apis-with-swagger-and-openapi

[PS15] M. Potter, T. Smith. Making code citable with Zenodo and GitHub. Software Sustainibility Institute, 2015.
https://www.software.ac.uk/blog/making-code-citable-zenodo-and-github

[RAT13] H. Rijgersberg, M. van Assem, J. Top. Ontology of units of measure and related concepts. Semantic Web 4(1):3–13, 2013.
doi:10.3233/SW-2012-0069
https://journals.sagepub.com/doi/abs/10.3233/SW-2012-0069

[SCF+09] X. Sanchez-Loro, J. Casademont, J. L. Ferrer, V. Beltran, M. Catalan, J. Paradells. Dynamic content negotiation in web environments. In Context-Aware Computing and Self-Managing Systems. Pp. 153–200. Chapman and Hall/CRC, 2009.
doi:10.1201/9781420077728-12

[Sha05] Y. Shafranovich. Common Format and MIME Type for Comma-Separated Values (CSV) Files. RFC 4180, Oct. 2005.
doi:10.17487/RFC4180
https://www.rfc-editor.org/info/rfc4180

[SSC15] S. Snydman, R. Sanderson, T. Cramer. The International Image Interoperability Framework (IIIF): A community & technology approach for web-based images. In Archiving conference. Volume 12, pp. 16–21. 2015.

[SSGK19] D. Sharma, R. Shukla, A. K. Giri, S. Kumar. A Brief Review on Search Engine Optimization. In 2019 9th International Conference on Cloud Computing, Data Science & Engineering (Confluence). Pp. 687–692. 2019.
doi:10.1109/CONFLUENCE.2019.8776976

[TBS+23] T. Tietz, O. Bruns, L. Söhn, J. Tolksdorf, E. Posthumus, J. J. Steller, H. Fliegl, E. Norouzi, J. Waitelonis, T. Schrade et al. From Floppy Disks to 5-Star LOD: FAIR Research Infrastructure for NFDI4Culture. In 3rd Workshop on Metadata and Research (objects) Management for Linked Open Science

(DaMaLOS), co-located with ESWC. 2023.
doi:10.5445/IR/1000159591

[TH20] F. Thiery, T. Homburg. QGIS - A SPARQLing Unicorn? Eine Einführung in Linked Open Geodata zur Integration von RDF in QGIS Plugins. Tagungsband FOSSGIS-Konferenz 2020 2020:68–71, Mar. 2020.
doi:10.5281/zenodo.3719127

[THS+21] F. Thiery, T. Homburg, S. C. Schmidt, J. Voß, M. Trognitz. SPARQLing Geodesy for Cultural Heritage – New Opportunities for Publishing and Analysing Volunteered Linked (geo-)data. FIG Peer Review Journal FIG e-Working Week 2021 – Virtually in the Netherlands 21-25 June 2021, June 2021.
doi:10.5281/zenodo.5639381

[TM24] F. Thiery, A. W. Mees. Sharing Linked Open Data with domain-specific data-driven community hubs – archaeology.link in NFDI4Objects. Archeologia e Calcolatori 35(2):63–74, 2024.
doi:10.19282/ac.35.2.2024.08
https://www.archcalc.cnr.it/journal/articles/1328

[TMW+23] F. Thiery, A. W. Mees, B. Weisser, F. F. Schäfer, S. Baars, S. Nolte, H. Senst, P. Von Rummel. Object-Related Research Data Workflows Within NFDI4Objects and Beyond. In Sure-Vetter and Goble (eds.), 1st Conference on Research Data Infrastructure (CoRDI) - Connecting Communities. Volume 1, pp. CoRDI2023–46. TIB Open Publishing, Hannover, Sept. 2023.
doi:10.52825/cordi.v1i.326
https://www.tib-op.org/ojs/index.php/CoRDI/article/view/326

[TRDB21] F. Thiery, L. Rokohl, A. Dingler, S. Beck. African Red Slip Ware Digital (ARS3D) - How to create a Feature? Squirrel Papers 3(1):#4, Nov. 2021.
doi:10.5281/zenodo.5642976

[TSB24] F. Thiery, F. Schenk, S. Baars. Dealing with doubts: site georeferencing in archaeology and in the geosciences. Archeologia e Calcolatori 35(2):97–106, 2024.
doi:10.19282/ac.35.2.2024.11
https://www.archcalc.cnr.it/journal/articles/1331

[TT23] F. Thiery, P. Thiery. Linked Open Ogham. How to publish and interlink various Ogham Data? Archeologia e Calcolatori 34(1):105–114, 2023.
doi:10.19282/ac.34.1.2023.12
https://doi.org/10.19282/ac.34.1.2023.12

[TVR+23] F. Thiery, J. Veller, L. Raddatz, L. Rokohl, F. Boochs, A. W. Mees. A Semi-Automatic Semantic-Model-Based Comparison Workflow for Archaeological Features on Roman Ceramics. ISPRS International Journal of Geo-

Information 12(4):167, Apr. 2023.
doi:10.3390/ijgi12040167
https://www.mdpi.com/2220-9964/12/4/167

[VAH14] B. Villazón-Terrazas, G. A. Atemezing, B. Hyland. Best Practices for Publishing Linked Data. W3C note, W3C, Jan. 2014.
https://www.w3.org/TR/2014/NOTE-ld-bp-20140109/

[VSN+18] A. Valdestilhas, T. Soru, M. Nentwig, E. Marx, M. Saleem, A. N. Ngomo. Where is My URI? In The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings. Pp. 671–681. 2018.
doi:10.1007/978-3-319-93417-4_43
https://doi.org/10.1007/978-3-319-93417-4_43

[VVH+16] R. Verborgh, M. Vander Sande, O. Hartig, J. Van Herwegen, L. De Vocht, B. De Meester, G. Haesendonck, P. Colpaert. Triple Pattern Fragments: A low-cost knowledge graph interface for the Web. Journal of Web Semantics 37-38:184–206, 2016.
doi:https://doi.org/10.1016/j.websem.2016.03.003
https://www.sciencedirect.com/science/article/pii/S1570826816000214

[Wal92] G. Wallace. The JPEG still picture compression standard. IEEE Transactions on Consumer Electronics 38(1):xviii–xxxiv, 1992.
doi:10.1109/30.125072

[Win13] J. Winn. Open data and the academy: an evaluation of CKAN for research data management. 5 2013.
https://repository.lincoln.ac.uk/articles/conference_contribution/Open_data_and_the_academy_an_evaluation_of_CKAN_for_research_data_management/25187387

[WKLW98] M. Wolf, J. A. Kunze, C. Lagoze, D. S. Weibel. Dublin Core Metadata for Resource Discovery. Technical report 2413, Sept. 1998.
doi:10.17487/RFC2413
https://www.rfc-editor.org/info/rfc2413