Universität Leipzig
Fakultät für Mathematik und Informatik
Institut für Informatik
Abteilung für Betriebliche Informationssysteme

Master Thesis

DataID
Semantically Rich Metadata for Complex Datasets

Leipzig, 6th January 2017	Markus Freudenberg *freudenberg@informatik.uni-leipzig.de*
Betreuender Hochschullehrer:	Dr.-Ing. Sebastian Hellmann and Dimitris Kontokostas Fakultät für Mathematik und Informatik Betriebliche Informationssysteme

Abstract

The rapid increase of data produced in a data-centric economy emphasises the need for rich metadata descriptions of datasets, covering many domains and scenarios. I will examine the current landscape of dataset metadata, evaluating their provisions for machine-readable resources, provenance, licensing, access and their ability extend easily while being able to interoperate with other vocabularies. The multilayer ontology of `DataID` will be introduced and described in detail to contrast its abilities in providing semantically rich metadata for complex datasets with other vocabularies. The satisfaction of complex demands on dataset metadata by `DataID` will be demonstrated with the `DBpedia` use case. Resulting in an evident increase in reuse, comprehension, linkability, discoverability, trust, access and processability of the published datasets. Particular attention was placed on the preservation of extensibility, to fit any given use case, while maintaining the interoperability with other metadata formats.

Acknowledgements

I would like to thank my supervisors Sebastian Hellmann and Dimitris Kontokostas for their unwavering support, patience and for all favours exchanged. My colleagues, who came to my aid on multiple occasions. Ricarda, who had to endure many stares into empty space. And my parents, for never doubting me while having my back.

Contents

List of Figures

Namespaces

Prefix	Namespace
dataid	`http://dataid.dbpedia.org/ns/core#`
dataid-acp	`http://dataid.dbpedia.org/ns/acp#`
dataid-ld	`http://dataid.dbpedia.org/ns/ld#`
dataid-md	`http://dataid.dbpedia.org/ns/md#`
dataid-mt	`http://dataid.dbpedia.org/ns/mt#`
datacite	`http://purl.org/spar/datacite/`
dcat	`http://www.w3.org/ns/dcat#`
dc	`http://purl.org/dc/elements/1.1/`
dct	`http://purl.org/dc/terms/`
dlo	`http://aligned-project.eu/ontologies/dlo#`
foaf	`http://xmlns.com/foaf/0.1/`
lvont	`http://lexvo.org/ontology#`
odrl	`http://www.w3.org/ns/odrl/2/`
org	`http://www.w3.org/ns/org#`
owl	`http://www.w3.org/2002/07/owl#`
prov	`http://www.w3.org/ns/prov#`
rdf	`http://www.w3.org/1999/02/22-rdf-syntax-ns#`
rdfs	`http://www.w3.org/2000/01/rdf-schema#`
r3d	`http://www.re3data.org/schema/3-0#`
sd	`http://www.w3.org/ns/sparql-service-description#`
skos	`http://www.w3.org/2004/02/skos/core#`
spdx	`http://spdx.org/rdf/terms/#`
time	`http://www.w3.org/2006/time#`
void	`http://rdfs.org/ns/void#`
xsd	`http://www.w3.org/2001/XMLSchema#`

Table 0.1: Namespaces and their prefixes used throughout this work

1 Introduction

1.1 Motivation

In 2006, Clive Humby coined the phrase "the new oil" for (digital) data[1], heralding the ever-expanding realm of what is now summarised as Big Data. Attributed with the same transformative and wealth-producing abilities, once connected to crude oil bursting out of the earth, data has become a cornerstone of economic and societal visions. In fact, the amount of data generated around the world has increased dramatically over the last years, begging the question whether those visions have already come to pass.

The steep increase in data produced can be ascribed to multiple factors. To name just a few:

- The growth in content and reach of the World Wide Web[2].
- The digitalising of former analogue data[3],[4]
- The realisation of what is called the Internet of Things (IoT)[5].
- The shift of classic fields of research and industry to computer-aided processes and digital resource management (e.g. digital humanities[6], industry 4.0[7]).
- Huge data collections about protein sequences or human disease taxonomies are established in the life sciences[8].
- Research areas like Natural Language Processing or Machine Learning are generating and refining data[9].

[1] https://www.theguardian.com/technology/2013/aug/23/tech-giants-data

[2] http://www.internetworldstats.com/emarketing.htm

[3] https://www.loc.gov/programs/national-recording-preservation-board/

[4] https://archive.org

[5] http://siliconangle.com/blog/2015/10/28/page/3\#post-254300

[6] http://www.dh.uni-leipzig.de/wo/

[7] http://www.europarl.europa.eu/RegData/etudes/STUD/2016/570007/IPOL_STU(2016)570007_EN.pdf

[8] https://www.ncbi.nlm.nih.gov/genbank/statistics/

[9] http://deeplearning.net

- In addition, open data initiatives like the Open Knowledge Foundation[10] are following the call for 'Raw data, Now!'[11] of Tim Berners-Lee, demanding open data from governments and organisations.

But with Big Data comes a big challenge. The increasing deluge of data is submerging data producers and possible consumers in a wave of unfiltered, unstructured and apparently unmanageable information. As a new discipline, data engineering is dealing with the fallout of this trend, namely with issues of how to extract, aggregate, store, refine, combine and distribute data of different sources in ways which give equal consideration to the four V's of Big Data: Volume, Velocity, Variety and Veracity[12].

Datasets are the building blocks of these endeavours. They are the combination of multiple data points bundled together by at least one dimension of distinction (such as source, topic or temporal information). When working with these chunks of data, extra data about data (or metadata) is needed. Dataset metadata enables users to discover, understand and (automatically) process the data it holds, as well as providing provenance on how a dataset came into existence. This data is often created, maintained and stored in diverse data repositories featuring disparate data models that are often unable to provide the necessary information to automatically process the datasets described. In addition, many use cases for dataset metadata call for more specific information than provided by most available metadata vocabularies. Extending existing metadata models to fit these scenarios is a cumbersome process resulting often in non-reusable solutions.

One vocabulary for dataset metadata is breaking this trend. Since its introduction in 2013, the Data Catalog Vocabulary (`DCAT`) [49], has been widely adopted as a foundation for dataset metadata in research, government and industry[13]. The very general approach adopted by the authors of `DCAT` allows for portraying any given (digital) object with this ontology. Extending `DCAT` is very easy and mappings to other metadata formats are not difficult to achieve.

Conversely, the abstract approach of `DCAT` is often too generic where specificity is needed, resulting in:

- Insufficient provenance information
- Missing relations between Datasets
- Relations to agents are too cursory

[10] https://okfn.org

[11] http://www.wired.co.uk/news/archive/2012-11/09/raw-data

[12] http://www.ibmbigdatahub.com/infographic/four-vs-big-data

[13] https://joinup.ec.europa.eu/sites/default/files/isa_field_path/2016-05-13_dcat-ap_intro_v0.05.pdf

- Technical description of resources on the Web (e.g. API endpoints) is lacking, restricting the accessibility of the data
- General lack of specificity, inviting non-machine-readable expressions of resources

Similar findings were concluded at the W3C/VRE4EIC workshop 'Smart Descriptions & Smarter Vocabularies' (SDSVoc) in 2016 [3].

As a result of lacking specificity, current representations of datasets with DCAT are often not contributing to the main benefits of publishing data on the Web: *"Reuse, Comprehension, Linkability, Discoverability, Trust, Access, Interoperability and Processability"* [48]. This, in turn, amplifies broader problems with published datasets, especially in the open data community. This is reflected by the Open Data Strategy[14], defining the following six barriers for "open public data" [15], proposed by the European Commission in 2011:

1. a lack of information that certain data actually exists and is available,
2. a lack of clarity of which public authority holds the data,
3. a lack of clarity about the terms of re-use,
4. data made available in formats that are difficult or expensive to use,
5. complicated licensing procedures or prohibitive fees,
6. exclusive re-use agreements with one commercial actor or re-use restricted to a government-owned company.

Many issues with DCAT itself or their manifestation in reality can be solved by existing ontologies, even when restricted only to well established and often recommended ontologies. For example, the Provenance Ontology [52], deals with questions on how to record provenance information on a very granular level. While the Open Digital Rights Language [53] provides machine readable descriptions of licenses and other policies. The existence of problems, like those listed above, despite these offered solutions, speaks to a larger problem of missing organisational structures for a 'landscaping of vocabularies' (offering recommendations on combining, revising and usage of ontologies). A study of 91 commonly used vocabularies concluded [24]:

> "Our validation detected a total of 6 typos, 14 missing or unavailable ontologies, 73 language level errors, 310 instances of ontology namespace violations and 2 class cycles which we believe to be errors."

[14] http://europa.eu/rapid/press-release_IP-11-1524_en.htm?locale=en

[15] http://europa.eu/rapid/press-release_MEMO-11-891_en.htm

These errors accumulate when strong interdependencies exist between vocabularies, adding logical and practical problems and aggravating unification issues of ontologies.

1.2 Objectives

This thesis will present the metadata model of `DataID`, a multi-layered metadata ecosystem, which, in its core, describes complex datasets and their different manifestations, their relations to other datasets and agents (such as persons or organisations) endowed with rights and responsibilities.

Improving the portrayal of Provenance, Licensing and Access, while maintaining the easy Extensibility and Interoperability of `DCAT`, are the linchpin objectives in my effort to present a comprehensive, extensible and interoperable metadata vocabulary. Multiple well-established ontologies (such as `PROV-O`, `VoID` and `FOAF`) are reused for maximum compatibility to establish a uniform and accepted way to describe and deliver dataset metadata for arbitrary datasets and to put existing standards into practice.

The `DataID Ecosystem` is a suite of ontologies comprised of `DataID core` and multiple extension ontologies, clustered around `DataID core`. It is the result of a modularisation process, which was necessary to preserve Extensibility and Interoperability of the `DCAT` vocabulary, on which all ontologies are based.

I want to present my solution for most of the current problems with dataset metadata in general and `DCAT` in particular, following these objectives:

Objective 1. Provide sufficient support for extensive and machine-readable representations for Provenance, Licensing and data Access.

Objective 2. Extend `DCAT` with well-established ontologies to resolve the pressing issues with dataset metadata if possible.

Objective 3. Show, that by modularising into a landscape of ontologies, `DataID` preserves the general character of `DCAT`, supporting Extensibility and Interoperability.

Objective 4. Prove that the resulting ecosystem is capable of serving for complex demands on dataset metadata (proving Extensibility).

Objective 5. Demonstrate the Interoperability with other metadata formats.

Objective 6. Evaluate the universal applicability of `DataID` for datasets against common demands on data publications.

In addition, `DataID` shall support the FAIR Data principles [76] (cf. Section 2.1.5) as well as the best practices defined by the Data on the Web Best Practices working group [48] (cf. Section 9.1.1) of the W3C (restricted to those practices where metadata is of concern).

`DataID` was developed under the H2020 project ALIGNED[16] (GA-644055), following its main goals:

- to be part of a unified software and data engineering process;
- describing the complete data lifecycle and domain model;
- with an emphasis on quality, productivity and agility.

In the context of ALIGNED, `DataID` is part of a shared model of software and data engineering to enable unified governance and coordination between aligned co-evolving software and data lifecycles:

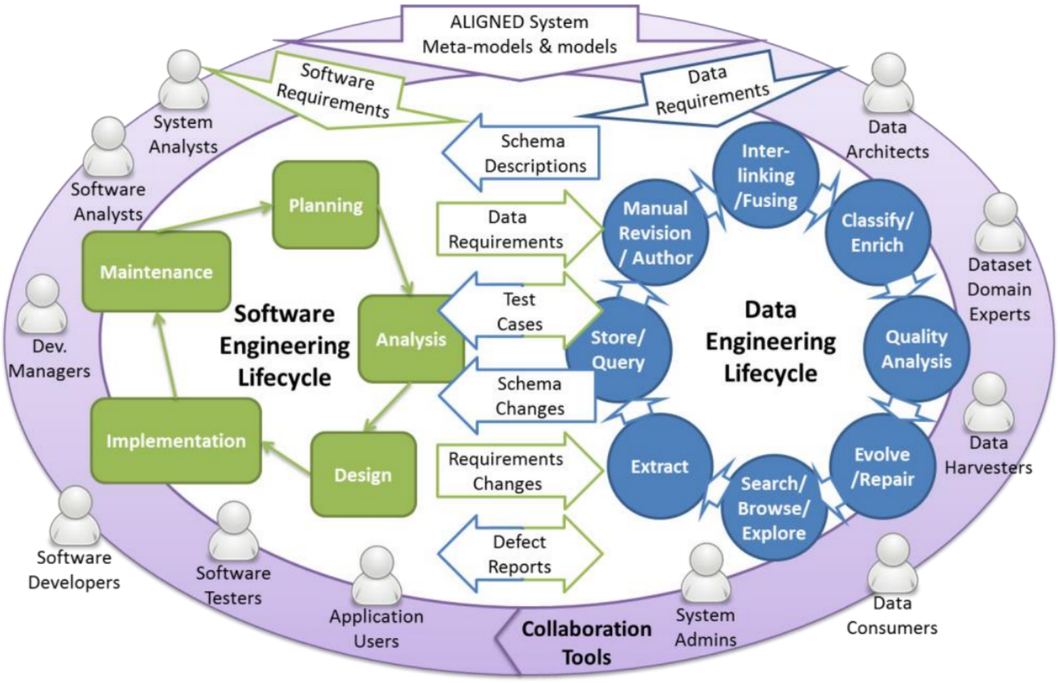

Figure 1.1: ALIGNED Software and Data Engineering Processes

add a last sentence?

[16] `http://aligned-project.eu`

1.3 Structure

This work is a comprehensive introduction to `DataID`, with a particular focus on the `DataID core` ontology, at the heart of the `DataID Ecosystem`. It is largely based on four publications:

1. Martin Brümmer, Ciro Baron, Ivan Ermilov, Markus Freudenberg, Dimitris Kontokostas and Sebastian Hellmann "DataID: Towards Semantically Rich Metadata for Complex Datasets". In: Proceedings of the 10th International Conference on Semantic Systems. [14] - An introduction to the first version of `DataID`.
2. Monika Solanki, Bojan Bozic, Markus Freudenberg, Dimitris Kontokostas, Christian Dirschl and Rob Brennan "Enabling Combined Software and Data Engineering at Web-Scale: The ALIGNED Suite of Ontologies". In: The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan [69] - An overview of the landscape of ontologies developed by the ALIGNED project.
3. Markus Freudenberg, Martin Brümmer, Jessika Rücknagel, Robert Ulrich, Thomas Eckart, Dimitris Kontokostas and Sebastian Hellmann "The Metadata Ecosystem of DataID". In: Metadata and Semantics Research: 10th International Conference, MTSR 2016, Göttingen, Germany [28] - An introduction to the `DataID Ecosystem`.
4. DataID core Ontology: A W3C member submission of the University of Leipzig, under review by the W3C at the time of writing, authored by Martin Brümmer and me (a preliminary version[17]).

After a look at related work (Chapter 3) on the subject of dataset metadata, I will present the `DataID Ecosystem` in Chapter 4, to introduce the guiding principles of this work. Chapter 5 describes the `DataID core` ontology in detail, containing a running example of a `DBpedia` language edition (cf. Section 3.2.9). Chapter 6 provides a best practice about publishing data on the Web with `DataID`, followed by an application of those practices to an application on how to solve complex metadata challenges with the `DataID Ecosystem`, by looking at Data Management Plans (Chapter 7). Chapter 8 provides mappings between `DataID` and multiple `CMD` profiles of the Component MetaData Infrastructure (`CMDI`). `DataID` will be evaluated in Chapter 9. A discussion on its development and future work follows in Chapter 10.

[17] `http://vmdbpedia.informatik.uni-leipzig.de/temporary/html/dataid-submission-pre.html`

2 Foundations

2.1 Data

Data is an almost intangible term. It is highly ambiguous and touches many fields of interest, stretching from philosophy to digital signal processing. Even in the context of Information Science, ***Data*** has multiple possible definitions. Here are some of them:

> "Data is a symbol set that is quantified and/or qualified." (Prof. Aldo de Albuquerque Barreto, Brazilian Institute for Information in Science and Technology, Brazil [79])

> "Data are sensory stimuli that we perceive through our senses." (Prof. Shifra Baruchson–Arbib, Bar Ilan University, Israel [79])

> "By data, we mean known facts that can be recorded and that have implicit meaning." (Prof. Shamkant Navathe, College of Computing at the Georgia Institute of Technology, USA [22])

> "Etymologically, data [...] is the plural of datum, a noun formed from the past participle of the Latin verb dare–to give. Originally, data were things that were given (accepted as "true"). A data element, d, is the smallest thing which can be recognised as a discrete element of that class of things named by a specific attribute, for a given unit of measure with a given precision of measurement." (Prof. Charles H. Davis, Indiana University, USA [79])

> "Data are the basic individual items of numeric or other information, garnered through observation; but in themselves, without context, they are devoid of information." (Dr. Quentin L. Burrell, Isle of Man International Business School, Isle of Man [79])

Information and ***Data*** seem to be closely linked and are often used interchangeably, yet they are not the same thing:

> "Datum is every thing or every unit that could increase the human knowledge or could allow to enlarge our field of scientific, theoreti-

> cal or practical knowledge, and that can be recorded, on whichever support, or orally handed. Data can arouse information and knowledge in our mind." (Prof. Maria Teresa Biagetti, University of Rome 1, Italy; based on C. S. Peirce, 1931, 1958 [79])

I will take a broader look at the term ***Data***, delineating it from the concepts of Information and Knowledge.

2.1.1 Data, Information, Knowledge

Data is the result of the application of syntactical rules against a set (sequence, string, etc.) of signs or signals, out of which a message between a sender and recipient is constructed [10]. Additional schematics might apply, adding structural data. ***Information*** can be gleaned from ***Data*** if one can find meaning in it. The result of this semantic expansion (or interpretation) of ***Data*** can be weighted by its novelty value or exceptionalness in the context of existing ***Information***, using Shannon's entropy [67]. ***Data*** becomes tokens of perceptions, or more commonly used in Information Science: instances of concepts, capable of changing the understanding of a context for the recipient of the original message. Linking ***Information*** in a context to determine their interrelations and inferring additional ***Information*** under the presumption of an intellectual goal, are the processes (*Pragmatics*) turning ***Information*** into ***Knowledge***.

The pyramidal structure depicted (Figure 2.1) is often crowned by an additional field called "Wisdom" [64]. Wisdom could be described as a form of evaluated ***Knowledge*** or understanding, as the pinnacle of human endeavour or enlightenment. But in the era of fake news, pathological mistrust and truthiness[18], this last step does not seem to follow naturally.

Many authors describe this hierarchy or derivations of it. The following common aspects are presented throughout this literature [64]:

- the key elements are ***Data***, ***Information*** and ***Knowledge***,
- these key elements are virtually always arranged in the same order, some models offer additional stages, such as wisdom, or enlightenment,
- the higher elements in the hierarchy can be explained in terms of the lower elements by identifying an appropriate transformation process,
- the implicit challenge is to understand and explain how ***Data*** is transformed into ***Information*** and information is transformed into ***Knowledge***.

[18] `https://en.wikipedia.org/wiki/Truthiness`

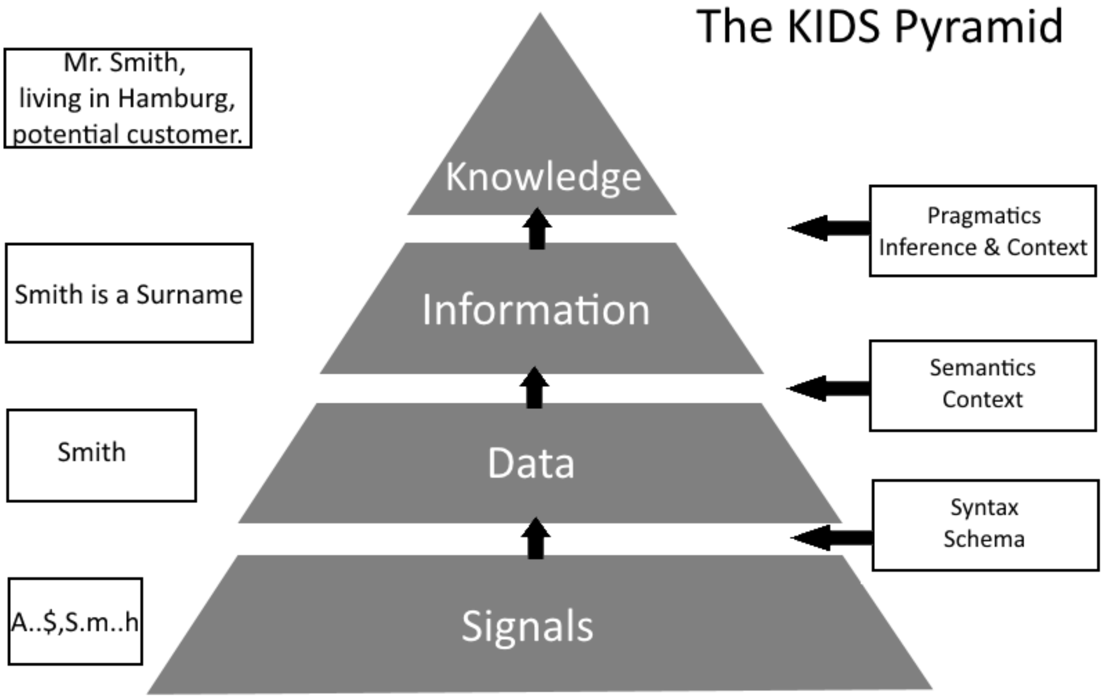

Figure 2.1: The KIDS Pyramid [10]

2.1.2 Digital Data

Digital Data is represented using the binary number system of ones (1) and zeros (0). Typically, these are combined into eight of their kind - named Byte. Bytes are used to identify characters of a given alphabet, which, in turn, provide the building stones for any operation, program or data point (datum).

In general, the term ***Digital Data*** is used to describes a collection of bytes, representing a digital mapping of an analogue counterpart (e.g. sound waves), or characters of an alphabet understood by humans or machines.

Digital Data is often categorised in structured and unstructured data. Unstructured data does not follow any predefined model and has to be interpreted by the recipient (reader) by its own merit (usually free text). Structured data is strictly adherent to a given data model, facilitating the interpretation of data by machines and humans alike (e.g. Schema Definition Language - XSD [72]).

2.1.3 Dataset

A ***Dataset***[19] is a bundle of data points, which have at least one common dimension of distinction. For example, a music album of an artist can be viewed as a ***Dataset***, where a single song represents a unit of data. Multiple songs are collected in an album with the common feature (among others) - the artist. Most commonly a ***Dataset*** corresponds to a collection of structured (digital) data in a single location (e.g. a database table, XML document etc.).

Datasets can manifest in different formats (e.g. in different file types). Therefore, the distinction between ***Dataset*** as a container for collecting data points of similar content or structure, and its final manifestation on a file system (database, service endpoint, etc.) is advisable.

2.1.4 Metadata

The National Information Standards Organization[20] (NISO), a United States non-profit standards organisation, published a paper in 2004, defining ***Metadata*** in a widely adopted manner:

> "Metadata is structured information that describes, explains, locates, or otherwise makes it easier to retrieve, use, or manage an information resource. Metadata is often called data about data or information about information." [59]

Metadata does not contribute additional content to the original message, but it can ease its transmission, processing and understanding. The meaning of the term ***Metadata***, and for what kind of data it applies, is different, depending on context, disciplines and communities. For example, library catalogue information about a certain book might be understood as ***Metadata*** in regard to the book itself, while in the context of a Library Management System this type of ***Metadata*** is considered data.

Specialised types of ***Metadata*** can be broadly separated into three types [59]:

- ***Descriptive Metadata*** describes a resource for purposes such as discovery and identification. It can include elements such as title, abstract, author, and keywords.
- ***Structural Metadata*** indicates how compound objects are put together, for example, how pages are ordered to form chapters.

[19] or 'data set', though this spelling seems to be replaced more and more

[20] `http://www.niso.org/home/`

- ***Administrative Metadata*** provides information to help manage a resource, such as when and how it was created, file type and other technical information, and who can access it. Subsets of administrative ***Metadata*** include:
  - ***Rights Management Metadata***, which deals with intellectual property rights and licenses.
  - ***Preservation Metadata***, which contains information needed to archive and preserve a resource.

A ***Metadata*** record conforms to a given schema since the use of unstructured data to qualify a different data resource is an exercise in futility. Various ***Metadata*** schemata or ontologies are available, often describing similar types of data. The most commonly used ***Metadata*** vocabulary is Dublin Core[21] (DC) by the Dublin Core Metadata Initiative (DCMI):

> "The original objective of the Dublin Core was to define a set of elements that could be used by authors to describe their own Web resources. [...] the goal was to define a few elements and some simple rules that could be applied by noncatalogers [sic]." [59]

The following example illustrates the use of Dublin Core attributes (e.g. `dc:description`) to describe a publication released as a PDF file:

```
dc:title="Metadata Demystified"
dc:creator="Brand, Amy"
dc:creator="Daly, Frank"
dc:creator="Meyers, Barbara"
dc:subject="metadata"
dc:description="Presents an overview of metadata conventions."
dc:publisher="NISO Press"
dc:publisher="The Sheridan Press"
dc:date="2003-07-01"
dc:type="Text"
dc:format="application/pdf"
dc:identifier="http://www.niso.org/standards/resources/metadata.pdf"
dc:language="en"
```

Listing 2.1: Dublin Core example

All Dublin Core attributes are optional, repeatable (non-functional) and present without order. Since its introduction in 1995, the initial list of 15 attributes has been revised and extended, forming the so-called DCMI Metadata Terms[22] (DCT). The very general character of this vocabulary provides a useful foundation for more complex schemata reusing DC (e.g. the Data Catalog Vocabulary).

[21] `http://dublincore.org/documents/dces/`

[22] `http://dublincore.org/documents/dcmi-terms/`

Metadata can describe any resource in any state or aggregation (single resource, collections, a part of a resource), at any abstraction level of a domain model. The International Federation of Library Associations and Institutions[23] (IFLA) defined the "Functional Requirements for Bibliographic Records" [37], a conceptual model for retrieval and access in online library catalogues and bibliographic databases: **Item** is an example of a **Manifestation**, which embodies an **Expression**, realising a **Work** of an author.

"For example, a *Metadata* record could describe a report, a particular edition of the report, or a specific copy of that edition of the report." [59]

Metadata can be embedded in the described digital object, alongside the object (e.g. in the same directory) or stored separately. HTML documents often keep *Metadata* in the HTML header or completely emerged in the document (cf. RDFa [50]). While a close coupling between data and *Metadata* is useful when updating them together, a separate approach often simplifies the management of large numbers of records.

The advantages of reliable *Metadata* for digital objects are manifold. These are the more noteworthy attributions of *Metadata*:

Resource Discovery: *Metadata* is the foremost source of information which aids agents (persons, software, etc.) to discover wanted data.

Organising Electronic Resources: Entities (e.g. books in a library) are organised in catalogues with the help of their digital *Metadata* records.

Interoperability: The interoperability of two digital resources can be determined by comparing their *Metadata* entries, on a syntactical as well as on a semantic level.

Digital Identification: Digital identifiers (such as URIs Section 2.2) are stored with *Metadata* records.

Provenance: An extensive record on provenance is a key for trustworthiness, detailing facts like source data, responsible agents or origin activities.

Quality: The quality of a data source can be enhanced significantly by rich *Metadata* records.

Reusability: Increasing discoverability, interoperability, provenance and quality are instrumental requirements for increasing reusability.

Preservation: "Metadata is key to ensuring that resources will survive and continue to be accessible into the future." [59]

[23] `http://www.ifla.org`

In the chain of processes transforming Signals, Data, Information into Knowledge (Section 2.1.1), ***Metadata*** can help with all transformation steps:

- To provide information about schemata to which a set of structured data adheres or the syntax description needed to understand the signs transmitted.
- To advance the interpretation of data by providing context information (e.g. geographical or temporal).
- To point out related (web-) resources to broaden the context of information (e.g. links to similar ***Datasets*** or related website).

Based on the observations of [59], as well as the concepts common to all domains of ***Metadata*** (discussed in [21]), ***Good Metadata*** follows these principles:

Simplicity "Key to the simplicity of [a language] is both limited vocabulary and simple structure." [44] In the context of ***Metadata*** vocabularies: A few general metadata terms which are applicable to any domain or use case and are understood by anyone.

Modularity "It allows designers of metadata schemas to create new assemblies based [...]." [21] on established ***Metadata*** schemas and benefit from observed best practice.

Reusability "Reuse is the extent to which [an object] can operate effectively for a variety of users in a variety of [...] contexts over time to achieve the same or a different objective from that envisaged by its supplier." [60]

Extensibility A set of core elements can be extended with further elements to describe the specific data of particular relevance to a community.

Interoperability "[...] is the ability of multiple systems with different hardware and software platforms, data structures, and interfaces to exchange data with minimal loss of content and functionality." [59]

In general, the argument can be made that simplicity and modularity are prerequisites for reusability, extensibility and interoperability.

2.1.5 The FAIR Data Principles

In 2014, a workshop in Leiden, Netherlands, was held, named "Jointly Designing a Data Fairport". A wide group of academics and representatives of companies and other organisations concluded the workshop by drafting a concise set of principles to overcome common obstacles, impeding data discovery and reuse of (scientific) data for humans and machines alike.

> "[...] humans increasingly rely on computational agents to undertake discovery and integration tasks on their behalf. This necessitates machines to be capable of autonomously and appropriately acting when faced with the wide range of types, formats, and access-mechanisms/protocols [...]. It also necessitates that the machines keep an exquisite record of provenance such that the data they are collecting can be accurately and adequately cited." [76]

The **FAIR** Guiding Principles as drafted by the attendees of this workshop [76]:

F To be Findable:

1 (meta)data are assigned a globally unique and persistent identifier

2 data are described with rich metadata (defined by R1 below)

3 metadata clearly and explicitly include the identifier of the data it describes

4 (meta)data are registered or indexed in a searchable resource

A To be Accessible:

1 (meta)data are retrievable by their identifier using a standardized communications protocol

   i. the protocol is open, free, and universally implementable

   ii. the protocol allows for an authentication and authorization procedure, where necessary

2 metadata are accessible, even when the data are no longer available

I To be Interoperable:

1 (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.

2 (meta)data use vocabularies that follow FAIR principles

3 (meta)data include qualified references to other (meta)data

R To be Reusable:

1 (meta)data are richly described with a plurality of relevant attributes

2 (meta)data are released with a clear and accessible data usage license

3 (meta)data are associated with detailed provenance

4 (meta)data meet domain-relevant community standards

The basic notion reflected by these principles is, that a minimal set of community-agreed guiding principles would help to access, appropriately integrate, re-use and cite the vast amount of data (generated continuously around the world). Which is especially important for machines, for their lack of an "intuitive sense of ′semantics′" [76].

Additionally, a scale for reflecting the ′FAIRness′ of a repository or its contained data objects was formulated. The different ′levels for FAIRness′ are [26]:

1. Each Data Object has a PID[24] and intrinsic FAIR metadata (in essence ′static′)
2. Each Data Object has ′user defined′ (and updated) metadata to give rich provenance in FAIR format of the data, what happened to it, what it has been used for, can be used for etc., which could also be seen as rich FAIR annotations.
3. The Data Elements themselves in the Data Objects are ′technically′ also FAIR, but not fully Open Access and not Reusable without restrictions (for instance Patient data or Proprietary data).
4. The metadata as well as the data elements themselves are fully FAIR and completely public, under well defined license. (Non-licensed data considered ′public′ by their owner will still be excluded from integration projects by for instance Pharmaceutical companies).

[24] persistent identifier

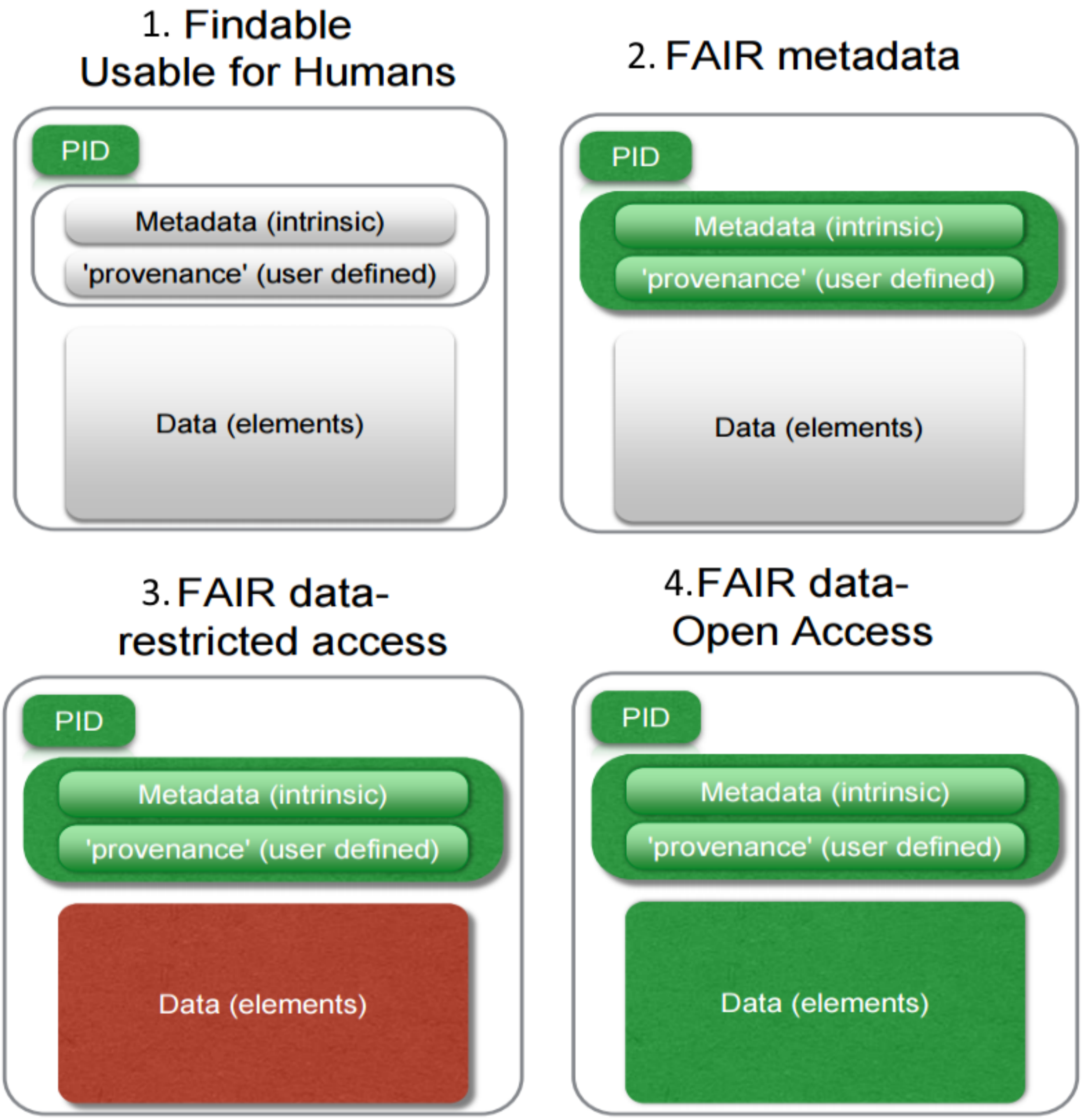

Figure 2.2: Data Objects with different 'levels of FAIRness' [26]

2.2 Semantic Web

I presented the common understanding of how data can herald information and knowledge in Section 2.1.1. I refrained from specifying which step or state might be restricted to humans or machines. None of those restrictions would probably be correct. While I don't want to elaborate on the question of: "Can machines have Knowledge?", I do state that machines can glean information from data. To interpret a message and derive meaning is not limited to the human mind. All that is needed is context and an understanding of the concepts and relations constituting a domain (semantics).

In 2001, Tim Berners-Lee, Hendler, and Lassila laid out their expectation of how the World Wide Web will eventually extend to become a Semantic Web [8]. The simple extension of web resources with structured, well-defined data (or metadata) would give meaning to resources, previously only decipherable by humans. To identify these data resources uniquely by Universal Resource Identifiers (URI) and provide links to other resources, would be the first steps in the direction towards a "web of data" [8] that can be processed directly by machines.

> "The Semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation." [8]

Tim Berners-Lee is the director of the World Wide Web Consortium[25] (W3C), which is responsible for the development of open standards for the World Wide Web, and by extension for the Semantic Web. At its core, the Semantic Web is defined by a collection of standards, which are *Recommendations* by the W3C. "Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data." [26].

2.2.1 Resource Description Framework (RDF)

This foundational technology of the Semantic Web and recommendation of the W3C [45] is used to describe any resource, using URIs as identifiers. Resources are defined by a set of characteristics which are expressed as attributes or relations to other resources. Statements in the form of "subject, predicate, object" (named "triples") are used to convey such characteristics. The resource described is uniquely identified by the URI of the subject. The object corresponds to

[25] `https://w3.org`

[26] `https://w3.org/standards/semanticweb/`

the content or reference of the statement. Literals (strings) are used to serialise content (or values), URIs of resources provide the target of a reference. The predicate is the semantic link between the subject and the object and defines the meaning of this statement. This simple linguistic construct makes RDF data understandable for humans and machines alike:

```
<http://dbpedia.org> -> publisher -> "DBpedia Association"
```

This triple describes the resource http://dbpedia.org (identified by the URI of the subject). Its object is the literal "DBpedia Association" connected to the subject with the predicate: publisher. Without the predicate, no meaning could be ascribed to the datum "DBpedia Association", as the type of relation between subject and object would be unknown. Thus, the predicate is what lends meaning to a statement.

The same information about the publisher of a website could be expressed differently in RDF data model. Since "DBpedia Association" represents an organisation, it could be introduced and described as an instance of a concept named "Organisation". This instance is capable of providing multiple attributes:

```
<http://dbpedia.org> -> publisher -> <http://dbpedia.org/DBpediaAssociation>
<http://dbpedia.org/DBpediaAssociation> -> type -> "Organisation"
<http://dbpedia.org/DBpediaAssociation> -> name -> "DBpedia Association"
<http://dbpedia.org/DBpediaAssociation> -> headquarter -> "Leipzig"
```

It is obvious that the instantiation of the objects "Organisation" and "Leipzig" would bear the same benefits. By extending this list of statements, describing additional resources and their characteristics, a directed graph is constructed, where nodes are represented by a resource and vertices are relations between resources. A URI can identify a resource without ambiguity (when carefully constructed based on HTTP domain names). This allows for linking of resources without regard to resource locations (datasets, service endpoints, etc.), data restrictions (e.g. licenses) or accessibility, creating, what is called, "Linked Data" (Section 2.2.3).

While labels like "publisher" have meaning for humans, they are ambiguous and could be misinterpreted. Especially machines cannot resolve this ambiguity and could not infer meaning from such statements. To address this issue, predicates are also identified by URIs that can be looked up for further information. The example below displays a valid RDF syntax called N-Triples.

```
<http://dbpedia.org> <http://purl.org/dc/terms/publisher> <http://dbpedia.org/DBpediaAssociation>.
<http://dbpedia.org/DBpediaAssociation> <http://purl.org/dc/terms/name> "DBpedia Association".
```

Predicates are called properties in the RDF data model. Sets of properties can be defined and documented by institutions (such as the DCMI - cf. Section 2.1.4), which , in turn, can be used by others, increasing interoperability and reusability.

These sets of properties together with associated concepts and annotations are called ontologies or vocabularies.

2.2.2 Web Ontology Language (OWL)

The Web Ontology Language is a W3C standard [66] based on the RDF data model, which could be summarised as an ontology for defining ontologies. According to Gruber,

> "An ontology is an explicit specification of a conceptualization." [31]

I delineate an ontology as a set of concepts, properties and logical axioms, with which to model a domain of knowledge or discourse. This conceptualization of a given domain or idea allows for a schema-bound representation of data with RDF as well as its automatic interpretation by machines.

Concepts are classes under which all objects of a domain can be classified, dividing up the domain in abstract objects. Properties provide meaning for the links to instances of concepts or literal objects (values), which, in turn, lends meaning to the concepts themselves. Additionally, subclass relations and restrictions on properties (e.g. domain and range definitions) help to specify more complex relations of a domain. This semantic layer between the domain knowledge on one side and its representation in RDF on the other is free of restrictions imposed by underlying technologies, distinguishing it from other data models (e.g. database schemata). A wide range of ontologies is available for any domain. Upper ontologies are general use vocabularies (such as Dublin Core Section 2.1.4) which can be reused together with other ontologies. High reusability is a desirable feature of any ontology. OWL is based (with some restrictions) on the RDF Schema (RDFS [32]), which provides basic elements for the description of ontologies (allowing for class hierarchies and basic relations). Multiple profiles are available [25], introducing different logical regimes to OWL, such as Description Logic (OWL-DL) or based on Rule Languages (OWL2-RL). Profiles allow for reasoning over RDF data, adhering to ontologies under such regimes.

Description Logic is a fragment of first-order predicate logic [6] and a formalism for representing knowledge under an *Open World Assumption*. OWL is heavily influenced by Description Logic (projects like DAML+OIL [35]) to achieve a beneficial trade-off between language expressiveness and computational complexity of reasoning.

2.2.3 Linked Data

Linked Data is the idea of how data is freed from islands of unconnected data, so it can be reused all over the World Wide Web referencing simply its URIs. Links between data objects from different sources are not bound to restrictions, authorisation procedures, licensing or any technical obstacles, they simply state that: 'There is a data object (published in a well-defined manner - e.g. RDF) which is related and it is identifiable on the Web with this URI.' Machines and humans can follow up these links and explore the 'Web of Data', expanding the contextual information.

> "Technically, Linked Data refers to data published on the Web in such a way that it is machine-readable, its meaning is explicitly defined, it is linked to other external data sets, and can, in turn, be linked to from external data sets." [9]

Additionally, a set of best practices were contrived by Tim Berners-Lee to identify the necessary steps needed to publish and link structured data on the Web, since "a surprising amount of data isn't linked in 2006, because of problems with one or more of the steps" [7].

1. Use URIs as names for things.
2. Use HTTP URIs so that people can look up those names.
3. When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL).
4. Include links to other URIs, so that they can discover more things.

Linked Data extends the demands of RDF as a data model by specifying HTTP as the access layer and requiring the openness of the resource to be regarded of high quality. The result is a 'Web of Data', a machine-readable, semantic network of structured data, opposed to the HTML-based web of documents (from humans, for humans).

3 Related Work

3.1 Dataset vocabularies

This section is dedicated to dataset metadata vocabularies and application profiles, to compare them and list their (dis-) advantages.

Based on the attributes of 'Good Metadata' (cf. Section 2.1.4), the FAIR Data Principles (Section 2.1.5) and my own list of principle goals of dataset metadata (Section 1.2), I contrived the following list of aspects, against which I want to evaluate each vocabulary.

A **The vocabulary encourages the use of richly described and machine-readable resources.** Concepts are defined as exhaustive as necessary to describe all relevant aspects, avoiding free text properties in general. For example, replacing a literal with a well-structured instance of `foaf:Agent`.

B **The vocabulary assigns globally unique URIs to metadata resources.** Demanding URIs as identifiers, independent of the chosen data representation (even for non-RDF or XML metadata).

C **The vocabulary can describe data access and access restrictions, consumable for humans and machines alike.** Sufficient effort has been made to describe the technical aspects of data access and possible restrictions to it (e.g. API authorisation methods), considering all possible formats of a dataset (files, APIs, etc.).

D **The vocabulary can portray provenance information extensively.** Dataset provenance can be described extensively, including other datasets (e.g. sources), activities (e.g. data generation activities) and agents (e.g. publisher) as well as inter-relational properties between these concepts.

E **The vocabulary provides for detailed descriptions of rights and licenses.** Machine readable licenses are of utmost importance.

F **The vocabulary provides properties to cite identifiers of the data described.** The possibility to reference the data described directly (by identifier) in the metadata is available.

G **The vocabulary provides for qualified references between resources.** Relations between instances of dataset metadata can be qualified by roles (specifying the type of relations), time and other restrictions.

H **The vocabulary is easy to extend, to fit any given use case.** The vocabulary can extend easily. No unnecessary restrictions impeding Extensibility, like restrictive cardinalities, are in place.

I **The vocabulary is unambiguous and easy to map to other metadata vocabularies.** The vocabulary is general enough to be able to interoperate with other metadata formats. Properties are defined clearly without semantic overlap to others (so users know which properties to use).

J **The vocabulary offers additional properties to aid dissemination and discovery.** Properties are in place to provide keywords, genres, taxonomies and general statements explaining the intention, use or gain of a dataset.

I will assign one of the following ratings to every item: **(2)** The requirement is supported in full. **(1)** The requirement is partially met. **(0)** The vocabulary does not support this requirement. While this list is helpful for evaluation and comparison purposes, the quality of a dataset vocabulary is also dependent on the intended domain of use and other factors, which I will mention as well. A summary of these findings is available as part of the evaluation (cf. Section 9.2.2).

3.1.1 The Data Catalog Vocabulary (DCAT)

In [1] the authors introduce a standardised interchange format for machine-readable representations of government data catalogues. The Data Catalog Vocabulary (DCAT) is a W3C recommendation [49] and serves as a foundation for many available dataset vocabularies and application profiles. Vocabulary terms for DCAT are inferred from the survey on seven data catalogues from Europe, US, New Zealand and Australia.

> "By using DCAT to describe datasets in data catalogs, publishers increase discoverability and enable applications easily to consume metadata from multiple catalogs. It further enables decentralized publishing of catalogs and facilitates federated dataset search across sites." [49]

DCAT defines three levels of abstraction, based on the following distinctions: A dataset describes a "collection of data, published or curated by a single agent, and available for access or download in one or more formats" [49], and represents the commonalities and varieties of the data held within (the 'idea' or intellectual content of that dataset). A Dataset is part of a data catalogue, representing multiple datasets (e.g. of an organisation). Datasets manifest themselves (are available) in different forms (such as files, service endpoints, feeds, etc.), expressed with the class `dcat:Distribution`. A dataset might be available for download at two different locations on the Web and available to be queried through an API endpoint. This scenario is describable by using three different `dcat:Distribution` instances.

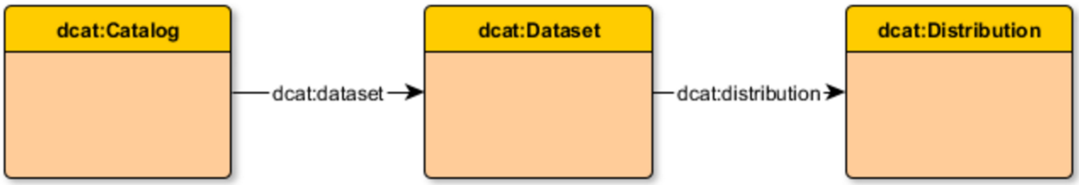

Figure 3.1: The basic idea of DCAT

This basic idea of differentiating between catalogue, dataset and distribution, has prevailed throughout the metadata domain for digital resources (not only datasets), and has become a quasi-requirement for metadata representations of web resources[27]. In fact, the general approach the authors of DCAT took, makes it possible to describe any digital object. This is a highly desirable feature, supporting Extensibility and Interoperability of the vocabulary.

A downside to this approach is the possible unspecificity of resources, especially regarding machine-readability, where uncertainty about formats is problematic. The full list of possible issues with exclusive DCAT solutions:

Insufficient provenance information:

- DCAT expresses provenance in a limited way using a few basic properties such as `dct:source` or `dct:creator`, which can not be further qualified.
- No possibility to specify activities involved in the creation of datasets.
- There is no support or incentive to describe source datasets, related publications or conversion activities of transformations responsible for the dataset. This lack is crucial, especially in a scientific context, as it omits the processes necessary to replicate a specific dataset.
- Insufficient portrayal of context information (e.g. licenses, geography, etc.)

Missing relations between Datasets:

- in general: Referencing related datasets is only possible on a very generic scale (e.g. `dct:relation`).

[27] https://www.w3.org/TR/dwbp/#context

- hierarchical: No inherent portrayal of dataset hierarchies is possible.
- evolutionary: No versioning pointers between dataset representations.

Relations to agents are too cursory:

- A very restricted number of properties pointing out agents, without any further qualification (e.g. `dct:publisher`, `dcat:contactPoint`), other related entities, like software, projects, funding, etc., are neglected altogether.
- No agent role concept to define new relations.

Technical description of distributions is lacking, restricting the accessibility of the data:

- Only superficial attributes for describing the technical characteristics of a distribution are available (e.g. `dcat:downloadURL`).
- No access information are available, such as access restrictions or service description, needed to describe service endpoints.
- No specificity when describing serialisation or media type of a distribution (e.g. file format).

The general lack of specificity, inviting non-machine-readable expressions of resources:

- Insufficient specificity of property ranges (e.g.: `dct:license`, `dct:temporal`, `dct:spatial`, `dct:language`, `dcat:mediaType`), thereby neglecting exactness of relevant metadata resources, such as licenses.
- A lack of referential and functional integrity due to missing role-based qualifications of properties (such as `dct:maintainer`) [39].

While this seems to be an extensive list of shortcomings, most of the points listed above are due to the general approach of `DCAT`. The list merely indicates that portraying dataset metadata with `DCAT` alone, might not be sufficient for most domains and use cases.

In turn, extending `DCAT` as upper ontology provides a sufficient basis for any metadata descriptions, without regard to a domain or use case. Adopting not only its basic ideas but the aspects of easy Extensibility and Interoperability, should prove beneficial. Addressing the issues with Provenance, Licensing and Access as well as domain-specific demands for metadata is the central task I set out to complete.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of `DCAT`	1	2	1	0	0	1	0	2	2	1	10

3.1.2 Vocabulary of Interlinked Datasets (VoID)

The Vocabulary of Interlinked Datasets (`VoID`) [2] is widely accepted and used within the Semantic Web community, for instance in projects such as OpenLink Virtuoso[28] , LODStats[29] , World Bank[30] and others. `VoID` can be used to express general metadata, statistical metadata, structural metadata and links between Linked Data datasets. Tools to create `VoID` metadata are described in [11] where authors also present techniques of reduction to create descriptions for Web-scale datasets. In the same paper, the importance of `VoID` is well established but there is still a lack of important metadata which is not described, for example for, Licensing and Provenance. For simple datasets, `VoID` performs well, which is supported by the vocabularies wide acceptance amongst many applications in research and industry. However, in a case of complex datasets `VoID` is not expressive enough. In particular, the access metadata property `void:dataDump`, which points to the data files of the particular dataset, is questionable. This property should link directly to dump files as described in W3C Interest Group Note: Describing Linked Datasets with the `VoID` Vocabulary[31]. Thus, additional semantic information about the data files and the structure of the dataset can not be expressed using `VoID`. This lack of a distribution concept is depriving the vocabulary of an important level of abstraction. Nevertheless, this ontology is useful especially for Linked Data datasets, offering many useful statistical properties (such as `void:triples`) and the very handy concept of `void:LinkSet` describing the particular relation between datasets, where one holds links to instances of the other.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of `VoID`	1	2	0	0	0	0	1	1	2	1	8

3.1.3 Comprehensive Knowledge Archive Network (CKAN)

Metadata models vary, and most of them do not offer enough granularity to describe complex datasets sufficiently in a semantically rich way. For example, `CKAN`[32] (Comprehensive Knowledge Archive Network), a data management system used widely in (open) data portals (such as datahub.io[33]) to provide web representations for datasets, operates on a JSON-based schema[34] developed by

[28] `http://virtuoso.openlinksw.com/`
[29] `http://stats.lod2.eu/`
[30] `http://worldbank.270a.info/`
[31] `http://www.w3.org/TR/void/`
[32] `http://ckan.org/`
[33] `http://datahub.io/`
[34] `https://github.com/KSP-CKAN/CKAN/blob/master/CKAN.schema`

the Open Knowledge Foundation[35]. CKAN allows simple access to a whole range of functions related to the management of datasets (such as search and faceting of data-sources) accessible via a REST interface. Its metadata schema has some similarities with DCAT: Datasets are collected under organisation objects, which are used as s primitive stand in for catalogues. Datasets have ’Resources’ which assume a similar role as a distribution in the DCAT vocabulary.

Alas, there is no clear definition of the resource-object within CKAN documentation, nor are there any noteworthy restrictions. This has led to a medley of different use cases for this concept, where it assumes the role of a dcat:Distribution in one dataset (containing all data of the dataset) and providing different slices of the data in another example [57]. Furthermore, the extensive use of key-value pairs for additional data led to a sheer host of (semi-) structured data, with only marginal agreement on key names between different Data Portals [57] adding to the general unclarity of this metadata format, complicating the mapping to other vocabularies.

This data model is semantically poor and inadequate for most applications consuming data automatically. I strongly discourage any organisation from adopting the CKAN format for their dataset metadata. Since it is used so frequently in data portals, I feel obliged to point out that there are mapping tools for most vocabularies to CKAN (as I provided for DataID in [14]).

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of CKAN	0	2	0	0	0	0	0	1	1	0	4

3.1.4 Metashare

The META-SHARE ontology[51] is the offspring of a prior, XSD-based [72] "metadata schema that allows aspects of [language resources] accounting for their whole lifecycle from their production to their usage to be described"[51]. META-SHARE differentiates between language resources (basically datasets with a language related purpose - text, audio etc.), technologies (e.g., tools, services) used for their processing and additional entities like reference documents, agents, projects or licenses. This allows for the portrayal of provenance in the domain of Natural Language Processing. Also, it offers an exemplary way of describing licenses and terms of reuse[36]. However, META-SHARE is highly specialised for language resources, thus lacking generality and extensibility for other use cases. While not implementing the DCAT vocabulary, META-SHARE does provide an almost complete mapping to DCAT. Mappings to other ontologies might

[35] https://okfn.org

[36] http://www.cosasbuenas.es/static/ms-rights/

prove difficult, due to the large size of the vocabulary and the often employed (and ample) controlled vocabularies. The related META-SHARE XSD schema has been implemented in the META-SHARE web portal[37], providing many NLP related datasets for download.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of META-SHARE	2	2	1	1	2	0	0	0	1	1	10

3.1.5 Asset Description Metadata Schema (ADMS)

The Asset Description Metadata Schema[38] (ADMS) is a profile of DCAT, which is specialised to describe "Semantic Assets". Assets (as subclass of dcat:Dataset) are highly reusable metadata (e.g. code lists, XML schemata, taxonomies, vocabularies, etc.) expressing the intellectual content of the data, which is represented (in most cases) in relatively small files.

ADMS adopts the DCAT structure and provides a well-defined way of versioning between entities. Its specialised nature makes it unsuited for a broader approach of portraying datasets (as intended by the authors), but it can still contribute useful properties to DCAT based vocabularies (e.g. Section 3.1.6). Since ADMS does not impose any restrictions, it can be extended to DCAT without any consequences for DCAT based metadata documents. The evaluation below therefore does not differ from DCAT.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of ADMS	1	2	1	0	0	1	0	2	2	1	10

3.1.6 DCAT Application Profile for data portals in Europe (DCAT-AP)

The DCAT Application Profile for data portals in Europe[39] (DCAT-AP) is a specification based on DCAT (extended with ADMS properties) for describing public sector datasets in Europe. It was developed by a working group under the auspices of the European Commission. Its basic use case is "to enable cross-data portal search for datasets and to make public sector data better searchable across

[37] http://www.meta-share.org

[38] https://www.w3.org/TR/vocab-adms/

[39] https://joinup.ec.europa.eu/asset/dcat_application_profile/home

borders and sectors" [30]. This can be achieved by the exchange of descriptions of datasets among data portals.

Traits of the resulting profile[40] (version 1.1) released in October 2015 [78]:

- It proposes mandatory, recommended or optional classes and properties to be used for a particular application;
- It identifies requirements to control vocabularies for this application;
- It gathers other elements to be considered as priorities or requirements for an application such as conformance statement, agent roles or cardinalities.

DCAT-AP has been endorsed by the Standards Committee of ISA2[41] in January of 2016[42] for the use in data portals. Further, it has been implemented by over 15 open data portals in the European Union, including the European Data Portal[43].

In general, while some recommendations are in place (e.g. using ODRL license documents - Section 3.2.2), DCAT-AP can not propose concrete improvements in extending DCAT to advance Provenance, Licensing or Access. As remarked in section 7 of its specification[44], the representation of different agent roles is lacking in the current version of DCAT-AP. In my opinion, the second solution proposed within, using PROV-O (Provenance Ontology - see Section 3.2.1), is the most comprehensive way of resolving this issue. Due to some cardinality restrictions (e.g. those on dcat:accessURL) and its specialisation for data portals, extending DCAT-AP to serve more elaborate purposes, can pose challenges.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of DCAT-AP	1	2	1	0	0	1	0	1	2	1	9

3.1.7 The HCLS Community Profile

The W3C interest group Semantic Web for Health Care and Life Sciences (HCLS) represents many stakeholders of the Life Sciences, seeking to "develop, advocate for, and support the use of Semantic Web technologies across health care, life sciences, clinical research and translational medicine" [16].

Their community profile (an ongoing effort by this W3C interest group[45]), extends DCAT with versioning and detailed summary statistics, through a three-

[40] https://joinup.ec.europa.eu/catalogue/distribution/dcat-ap-version-11

[41] https://ec.europa.eu/isa/isa2/index_en.htm

[42] https://joinup.ec.europa.eu/community/semic/news/dcat-ap-v11-endorsed-isa-committee

[43] https://www.europeandataportal.eu/

[44] https://joinup.ec.europa.eu/catalogue/distribution/dcat-ap-version-11

[45] https://www.w3.org/blog/hcls/

component model. This model introduces an additional abstraction level between dataset and distribution, the so-called 'Version Level Description', which contains version specific properties (e.g. `dct:isVersionOf`). The profile is structured in multiple modules, dealing with different levels of specificity [19]:

Core Metadata captures generic metadata about the dataset, e.g., its title, description, and publisher.

Identifiers describe the patterns used for identifiers within the dataset and the URI namespaces for RDF datasets.

Provenance and Change describe the version of the dataset and its relationship with other versions of the same dataset and related datasets, e.g., an external dataset that is used as a source of information.

Availability/Distributions provides details of the distribution files, including their formats, in which the dataset is made available for reuse.

Statistics used to summarise the content of the dataset.

The HCLS profile reuses 18 vocabularies with 61 properties [19], covering many goals of my evaluation. The chosen approach of this profile is sound and achieves qualitatively good metadata, with an emphasis on FAIR Data principles.

One problem is the large number of reused vocabularies with overlapping purposes, which cause difficulties when mapping to other vocabularies. Its cardinality restrictions can pose problems when extending the profile to use cases, especially outside the health care domain. Although efforts were made to cover provenance in general, problems like qualifying otherwise static relations to agents or datasets can not be solved by the incorporated vocabularies. The approach to portraying licenses does not improve `DCAT`.

A specific problem is the use of the property `dcat:accessURL`, which, according to the profile, could be used on the dataset abstraction level ('Summary Level Description'). This clearly violates the specification of `DCAT`. In general the border between `dcat:Dataset` and `dcat:Distribution` has to be defined more carefully when adding an additional layer in between.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of HCLS Profile	2	2	2	1	0	1	0	2	1	1	12

3.1.8 CERIF

The metadata format CERIF[46] (Common European Research Information Format) is a metadata development, which began in the early 1990s. The shortcomings of its first approach were addressed by CERIF2000, which became an EU recommendation for its member states for research data. Since 2002 CERIF is further developed by the Current Research Information Systems organisation[47].

CERIF provides generalised base concepts and relations between them, based on an entity-relationship approach [40]. Relations (or 'linking entities') are qualified with roles, temporal and spatial statements, supplemented with provenance and versioning information. In contrast to Dublin Core based metadata standards (i.e. DCAT), CERIF was developed with a particular focus on referential and functional integrity of resources, avoiding ambiguity in interpretation [39]. CERIF provides much more than just metadata for datasets, it addresses metadata needs for the whole science community, describing projects, funding, facilities, organisations and other.

This paper lines out the main characteristics of CERIF metadata [39]:,

- it separates base entities from relationships between them and thus represents the more flexible fully-connected graph rather than a hierarchy;
- it has generalised base entities with instances specialised by role (e.g. <person> rather than <author>), in the linking entities;
- it handles multilinguality by design and temporal information (so representing versions) to the appropriate attribute treated as an entity (example <title> linked to <publication>);
- temporal information is in the link entities not the base entities (e.g. employment between two dates is in the linking relation between <person> and <organisation> and not an attribute of either of the base entities);
- the temporal information in linking entities provides provenance and versioning recording (e.g. versions of datasets and – in the associated role attribute – the method of update or change);
- CERIF separates the semantics into a special 'layer' which is referenced from CERIF instances. The semantic layer includes permissible values for roles in any linking entity and controlled values of attributes in base entities (e.g. ISO country codes). Thus semantic terms are stored once and referenced many times (preserving integrity).

[46] http://www.eurocris.org/cerif/main-features-cerif

[47] http://www.eurocris.org

CERIF offers a comprehensive approach to solving a host of metadata demands in a sophisticated manner. The chosen abstraction levels (layers) are appropriate, and the adherence to an entity-relationship approach is arguably a working solution for qualifying relations. The main problem with CERIF is the complexity of its ontology[48] [49] together with the unique approach to metadata (unlike the DCAT based understanding of metadata). The imposes hurdles when studying, mapping and extending this ontology. Furthermore, the ontology proves to be not specific enough when dealing with information on Access and Licensing.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of CERIF	2	2	1	1	0	2	2	2	2	1	15

3.1.9 Component MetaData Infrastructure (CMDI)

The Component MetaData Infrastructure is a component-based framework for the creation and utilisation of metadata schemata[13]. It allows the distributed development of metadata components (defined as sets of related elements) and their combination to profiles in any level of detail, forming the basis for the creation of resource-specific XML Schemata and around one million publicly available metadata files. CMDI is a flexible metadata framework, which can be applied to resources from any scientific field of interest. It is especially relevant in the context of the European research infrastructure CLARIN[34] where it is used to describe resources with a focus on the humanities and social sciences.

The very flexible and open approach of the CMDI which allows for its wide applicability, may lead in parts to problems regarding consistency and Interoperability. Despite being rich in descriptive metadata, some CMD profiles lack consistent information of the kind stated in Section 7.1. This includes the explicit specification of involved persons, descriptions of authoritative structures as well as technical details and actual download locations.

The highly modularised nature of this vocabulary makes it hard to attach a single evaluation to it. However, considering the onesided distribution of relevant CMD profiles to actual metadata instances (cf. Section 8.2), we can base this evaluation on the most probable profiles.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of CMDI	1	2	0	0	1	2	0	2	0	1	9

[48] http://eurocris.org/ontologies/cerif/1.3/

[49] http://eurocris.org/ontologies/semcerif/1.3/

3.1.10 DataID version 1.0.0

The common shortcomings of dataset vocabularies revealed in this section were also afflicting the previous version (1.0.0) of the `DataID` ontology [14]. Rooted in the Linked Data world, it neglected important information or provided properties (e.g. `dataid:graphName`) which are orphans outside this domain.

While it already imported the important Provenance Ontology (`PROV-O`-Section 3.2.1), to cover the general issues with Provenance, it was lacking regarding the specificity of Access and Licensing. The narrow definition of datasets (i.e. restricted to Linked Data datasets) was inadequate for use cases outside this domain and so inhibited Extensibility.

Requirement	A	B	C	D	E	F	G	H	I	J	Sum
Evaluation of DataID 1.0.0	1	2	1	1	2	1	2	1	2	1	14

3.2 Secondary Literature

This section proffers a collection of associated literature, not directly touching on the subject of dataset metadata. Many subjects, such as the representation of licenses and data quality, are relevant for providing metadata of datasets.

3.2.1 The Provenance Ontology (PROV-O)

> "Provenance is defined as a record that describes the people, institutions, entities, and activities involved in producing, influencing, or delivering a piece of data or a thing. In particular, the provenance of information is crucial in deciding whether the information is to be trusted, how it should be integrated with other diverse information sources, and how to give credit to its originators when reusing it. In an open and inclusive environment such as the Web, where users find information that is often contradictory or questionable, provenance can help those users to make trust judgements." [56]

The Provenance Ontology[52] (`PROV-O`) is a widely adopted W3C recommended standard and serves as a lightweight way to express the provenance and interactions between activities, agents and entities (e.g. datasets).

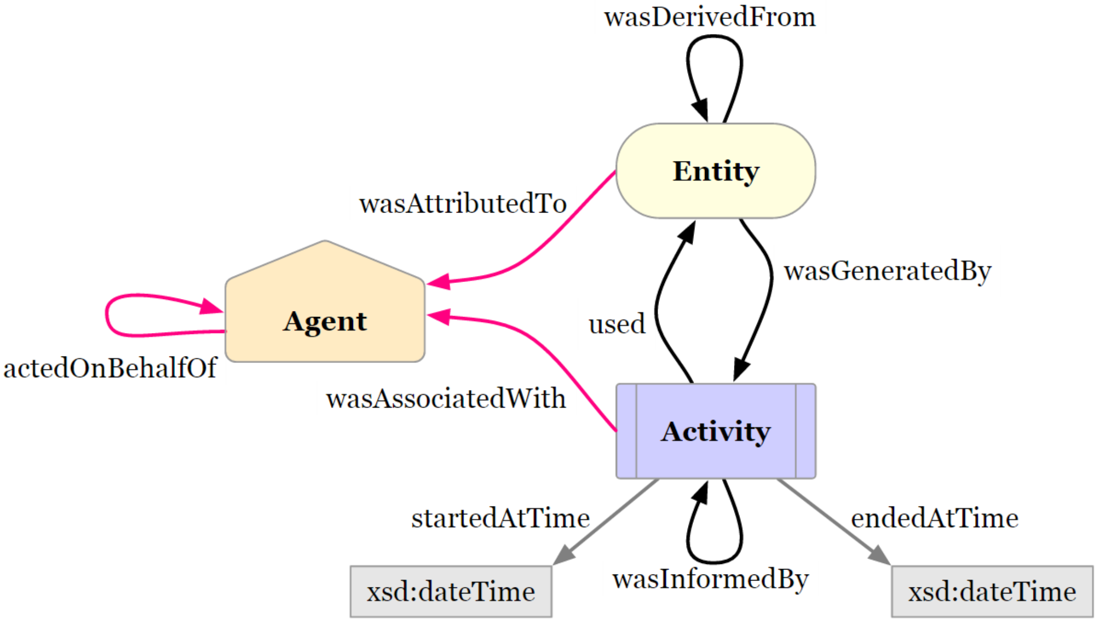

Figure 3.2: Linchpin of the Provenance Ontology: Entities, Agents, Activities [52]

In the context of datasets, provenance information on the activities which helped to create a dataset, which agents were involved in these processes and who to contact about it as well as source datasets or other entities involved are of interest, especially when trying to determine the trustworthiness of data. This ontology is "[...] the foundation to implement provenance applications in different domains that can represent, exchange, and integrate provenance information generated in different systems and under different contexts." [52]

Each of the relations depicted (Figure 3.2) have suitable qualification classes (e.g. `prov:Association`), so that a property (`prov:wasAssociatedWith`) can be inferred from the qualified property path (`prov:qualifiedAssociation/prov:agent`). Qualification classes provide qualification via properties such as `prov:role`.

In this example (from the official specification of `PROV-O` [52]), an association provides an additional description about the `:illustrationActivity` that an agent named 'Derek' influenced:

```
@prefix prov:                   <http://www.w3.org/ns/prov#> .
@prefix :                       <http://example.org#> .

:illustrationActivity
   a prov:Activity;                  ## this illustration activity was
   prov:wasAssociatedWith :derek;    ##   associated with Derek in some way.
.

:derek a prov:Agent .

:illustrationActivity
   prov:qualifiedAssociation [       ## Qualify how the :illustrationActivity
      a prov:Association;            ##   was associated with the Agent Derek.
      prov:agent   :derek

      prov:hadRole :illustrationist; ## Qualification: The role that Derek served
      prov:hadPlan :tutorial_blog;   ## Qualification: The plan (or instructions)
                                     ##   that Derek followed when creating
   ].                                ##   the graphical chart.

:tutorial_blog   a prov:Plan, prov:Entity .
:illustrationist a prov:Role .
```

`PROV-O` will have a central role in the creation of DataID. Further reading on the key requirements, guiding principles, and design decisions which influenced the PROV Family of Documents[50] is advised (cf. [55]).

[50] https://www.w3.org/TR/2012/WD-prov-overview-20121211/

3.2.2 Open Digital Rights Language (ODRL)

The Open Digital Rights Language (ODRL)[51] is an initiative of the W3C community group with the same name[52], aiming to develop an open standard for policy expressions. The ODRL Version 2.1 core model defines licensing policies regarding their permissions granted, duties and constraints associated with these permissions as well as involved legal parties. Thus, an ODRL description allows specifying, in a machine-readable way, if data can be edited, integrated or redistributed.

3.2.3 Lexvo.org

Lexvo.org[53] is a service publishing language related information as Linked Data on the Web. The data published is conform to the Lexvo Ontology, providing unique identifiers for human languages in the context of geography, language families, words and word senses, scripts and characters [54]. All in all, the Lexvo dataset consists of over 8000 languages with a broad spectrum of language-related information that is extensively used by many data publishers and communities.

3.2.4 Friend of a Friend vocabulary (FOAF)

The 'Friend of a Friend' ontology (FOAF), provides a way to create machine-readable resources, portraying agents (such as people, companies, organisations) together with their interests and relationships. FOAF can describe three kind of networks: "social networks of human collaboration, friendship and association; representational networks that describe a simplified view of a cartoon universe in factual terms, and information networks that use Web-based linking to share independently published descriptions of this inter-connected world." [12] Over time, FOAF has become a fundamental ontology of the Semantic Web, used mostly to describe basic properties of agents in the context of the World Wide Web (e.g. name, e-mail addresses, etc.). This vocabulary for basic structured information on agents is a community driven effort, available in version 0.99.

[51] https://www.w3.org/ns/odrl/2/ODRL21

[52] https://www.w3.org/community/odrl/

[53] http://www.lexvo.org

3.2.5 DataCite Ontology

DataCite is a "global non-profit organisation that provides persistent identifiers (DOIs) for research data [with the goal] to help the research community locate, identify, and cite research data with confidence"[54].

DataCite published an XML schema for describing and citing research data[55], which is elaborate and at times novel approach for providing dataset metadata. Due to its rigid XML structure with many cardinality restrictions, it does not feature in my collection of dataset metadata in this chapter.

Of more interest, is the DataCite ontology which was published as an OWL ontology[56] and has a particular focus on representing identifiers (Figure 3.3) and is part of the SPAR Ontology Suite[57]. The `datacite:Identifier` class is divided into ResourceIdentifier and AgentIdentifier. In addition, an IdentifierScheme defines the format of the literal which represents the identifier. As opposed to an approach where data-types express the pertaining scheme of a identifier literal, this ontology allows for adding new schemes without altering the vocabulary itself and adding additional qualifications to the scheme entity [62]. Furthermore, the DataCite ontology contains a multitude of predefined `datacite:IdentifierScheme` instances, ready to be used.

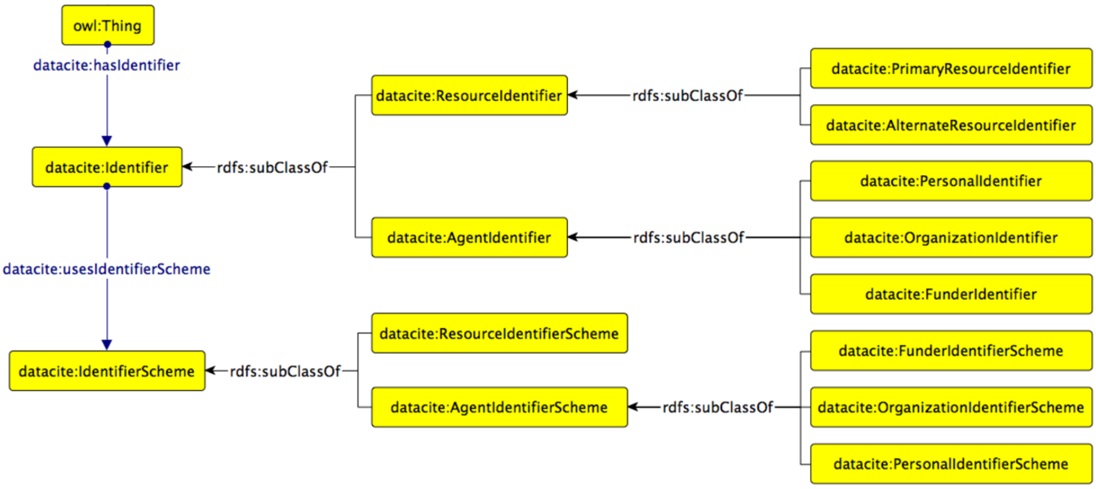
owl:Thing
datacite:hasIdentifier
datacite:Identifier
rdfs:subClassOf
datacite:usesIdentifierScheme
datacite:IdentifierScheme
rdfs:subClassOf
datacite:ResourceIdentifier
rdfs:subClassOf
datacite:AgentIdentifier
rdfs:subClassOf
datacite:ResourceIdentifierScheme
datacite:AgentIdentifierScheme
rdfs:subClassOf
datacite:PrimaryResourceIdentifier
datacite:AlternateResourceIdentifier
datacite:PersonalIdentifier
datacite:OrganizationIdentifier
datacite:FunderIdentifier
datacite:FunderIdentifierScheme
datacite:OrganizationIdentifierScheme
datacite:PersonalIdentifierScheme

Figure 3.3: DataCite Identifier and IdentifierScheme [62]

[54] `https://www.datacite.org/mission.html`

[55] `http://schema.datacite.org/meta/kernel-4.0/`

[56] `http://www.sparontologies.net/ontologies/datacite/source.html`

[57] `http://www.sparontologies.net`

3.2.6 re3data.org

The re3data[58] registry currently lists over 1.600 research repositories, making it the largest and most comprehensive registry of data repositories available on the web. By providing a detailed metadata description of repositories, the registry helps researchers, funding bodies, publishers and research organisations to find an appropriate data repository for different purposes[61]. Initiated by multiple German research organisations, funded by the German Research Foundation[59] from 2012 until 2015, re3data is now a service of DataCite[60]. In 2014 re3data merged with the DataBib registry for research data repositories into one service[61]. The re3data project was initiated by the Library and Information Services section (LIS) of the German Research Centre for Geosciences (GFZ), the library of the Karlsruhe Institute of Technology (KIT) and the Berlin School of Library and Information Science (BSLIS) at Humboldt-Universität zu Berlin.

One central goal of re3data is to enhance the visibility of existing research data repositories and to enable all those who are interested in finding a repository to assess a respective information service. This is achieved by an extensive and quality approved metadata description of the listed research data repositories. The basis for this description is the "Metadata Schema for the Description of Research Data Repositories", having 42 properties in the current version 3.0 [65].

3.2.7 Data Lifecycle Ontology (DLO)

The Data Lifecycle Ontology[62] provides a set of conceptual entities, agents, activities, and roles to represent the general data engineering process. Furthermore, it is the basis for deriving specific domain ontologies which represent life-cycles of data engineering projects such as `DBpedia`. With the incorporation of DLO into the `DataID Ecosystem`, *Activities & Plans* facilitates a basis for the information exchange needed of combined software and (big) data engineering [69].

3.2.8 The Organization Ontology (ORG)

The Organization Ontology[63] "[...] describes a core ontology for organizational structures, aimed at supporting linked data publishing of organizational infor-

[58] http://www.re3data.org/

[59] http://www.dfg.de/

[60] https://www.datacite.org/

[61] http://www.re3data.org/tag/databib/

[62] https://w3id.org/dlo

[63] https://www.w3.org/TR/vocab-org/

mation across a number of domains. It is designed to allow domain-specific extensions to add classification of organizations and roles, as well as extensions to support neighbouring information such as organizational activities." [63] ORG is a W3C recommendation and provides the backdrop of organisational structures to any given domain.

3.2.9 DBpedia

DBpedia is a community effort to extract structured information from Wikipedia and to make this information available on the Web[64], where it is used as a common point of reference for interlinking and enriching most of the structured data on the Web today, establishing it as the center of the so called 'LOD cloud[65]',

The main focus of the DBpedia extraction lies in mapping of info boxes, templates and other easily identified structured data found in Wikipedia articles to properties of the DBpedia ontology, reflecting the relations and classifications found within Wikipedia [46]. Together with the mappings to the Wikipedia XML templates, this ontology is curated by a community of interested people around the world to update schematic changes in time for the bi-yearly DBpedia releases.

Each release consists of around one hundred datasets for each of its 130 (2016) language editions (reflecting most of the languages of Wikipedia). Datasets are often created around particular properties (such as rdf:type) or other distinguishing features. To reflect this large corpus of structured data, the need for specific demands on its metadata is self-evident.

3.3 Extensibility and Interoperability

This section is dedicated to a discussion on Extensibility and Interoperability. Both concepts are of significant import to this thesis and are therefore discussed here in detail. In the context of metadata schemata, both tend to influence the respective other concept eminently.

[64] http://dbpedia.org

[65] http://lod-cloud.net

3.3.1 Extensibility

An extensible ontology is a well-designed ontology. Thomas Gruber elevated "Extendibility" into his list of design criteria for ontologies:

> "An ontology should be designed to anticipate the uses of the shared vocabulary. It should offer a conceptual foundation for a range of anticipated tasks, and the representation should be crafted so that one can extend and specialize the ontology monotonically. In other words, one should be able to define new terms for special uses based on the existing vocabulary, in a way that does not require the revision of the existing definitions." [31]

Gómez-Pérez defines this term slightly different:

> "The expandability of an ontology [...] is a measure of the cost and effort required, in order to extend the ontology by new definitions." [29]

The trait of Extensibility is of high importance to metadata vocabularies. To successfully depict a domain with a metadata vocabulary, a suitable extension of the provided axioms is often necessary, to reflect the peculiarities of a given scenario. Extending an ontology has multiple advantages in contrast to creating a vocabulary from scratch, which is laid out in this paper [41].

3.3.2 Interoperability

> "It is becoming generally accepted in the information community that interoperability is one of the most important principles in metadata implementation. [...] From the very beginning of a metadata project, the principles that enable user-centered and interoperable services should be foremost in design and implementation." [17]

Many attempts have been made to define the concept of Interoperability:

> "[...] the ability of multiple systems with different hardware and software platforms, data structures, and interfaces to exchange data with minimal loss of content and functionality." [59]

> "[...] the compatibility of two or more systems such that they can exchange information and data and can use the exchanged information and data without any special manipulation." [71]

Haselhofer & Klas offer a definition for metadata Interoperability based on their survey of techniques for achieving metadata interoperability:

> "Metadata interoperability is a qualitative property of metadata information objects that enables systems and applications to work with or use these objects across system boundaries." [33]

The notion of Interoperability can be further delineated: Tolk [73] presented a revised version of the 'Levels of Conceptual Interoperability Model' comprised of:

1. no interoperability
2. technical interoperability (communication infrastructure established)
3. syntactic interoperability (common structure to exchange information)
4. semantic interoperability (common information model)
5. pragmatic interoperability (context awareness)
6. dynamic interoperability (ability to comprehend state changes)
7. conceptual interoperability (fully specified, but implementation independent model)

OWL-based metadata is capable of establishing pragmatic Interoperability between two systems (i.e. transferring the intent of the author to the consumer by portraying the same contextual details available to the author). This is a theoretical feature and in practice not easy to achieve. However, providing well-defined and detailed (and machine-readable) resources elevates metadata vocabularies over the mere ability to provide semantic Interoperability.

Different approaches exist to achieve Interoperability between two metadata formats. Chan [17] describes seven approaches: Uniform standard, Application profiling, Derivation, Crosswalk/mapping, Switching schema, Lingua franca and a Metadata framework, which are not mutually exclusive. Haselhofer & Klas delineate four main techniques: Model Agreement, Meta-Model Agreement, Model Reconciliation and Metadata Mapping [33]. Where Metadata Mapping subsumes both schema and instance level mapping of the Model Reconciliation approach, under the assumption that all metadata information is expressed in the same schema definition language (making the reconciliation at a language level redundant).

3.3.3 Extensibility vs. Interoperability

When regarding the definitions of Extensibility and Interoperability, taking into account the intricate requirements of many use cases (such as in Chapter 7), Extensibility and Interoperability seem contradictory when leaving the more general levels of a domain description. A vocabulary capable of interacting with other metadata vocabularies might be too general to fit certain scenarios of use. Restrictive extensions to a vocabulary might encroach on its ability to translate into other useful metadata formats.

This discrepancy was already described by Gruber, who noted an obvious contradiction between Extensibility on the one hand and ontological commitment[66] on the other:

> "Another apparent contradiction is between extendibility and ontological commitment. An ontology that anticipates a range of tasks need not include vocabulary sufficient to express all the knowledge relevant to those tasks [...]. An extensible ontology may specify a very general theory, but include the representational machinery to define the required specializations." [31]

Interoperability is very similar to the concept of ontological commitment, as in both cases we can anticipate a set of tasks/properties which are relevant in the other/target domain. The notion of contradiction between Extensibility and Interoperability is also corroborated by Frystyk Nielsen:

> "Unfortunately evolvability[67] and interoperability don't work towards the same goal - in fact, they can be considered to be two forces that work against each other." [58]

To find the right blend of both concepts depends on the use case and the target vocabulary (or system) with which to interoperate. The Interoperability between two vocabularies of the same domain tends to increase when the Extensibility of at least one is heightened. A general or upper ontology can more easily interoperate with another ontology.

[66] ontological commitment is the measure of how specialized an ontology is compared to the weakest theory - only those terms that are essential to the communication of knowledge consistent with that theory.

[67] I do not differentiate between `evolvability` and Extensibility (as done in this source) in the context of this thesis. The discrepancies with Interoperability are true for both concepts. Letting features 'die out' over time does not impact, in my understanding, the aspect of Extensibility.

Disambiguation

There are multiple interpretations of the word/acronym `DataID` depending on the context. It can refer to a `DataID` metadata document, the serialisation of a `DataID` RDF graph. Such a graph is the result of the appliance of `DataID` ontologies to one or more datasets, resulting in a collection of RDF statements based on these ontologies. Or it is used to name an instance of the concept `dataid:DataId`, meaning the entry into a `dcat:Catalog`, the most abstract entity in every `DataID` document. Furthermore, there is the notion of a `DataID Ecosystem`, which describes the environment consistent of multiple `DataID` ontologies and additional extensions. I will be explicit and use the terms `DataID` document, `DataID` graph or `DataID` resource (or instance, entity) as well as `DataID Ecosystem` in the remainder of this thesis.

The following naming regime for keywords is adopted:

Keywords such as ***MediaType*** (and their plural form) refer to instances of concepts with the same name, or named individuals in the `DataID` ontologies. This is only true for concepts of ontologies from the `DataID Ecosystem`. When referring to a ***Dataset***, an instance of `dataid:Dataset` is addressed and not, for example, the concept with the same name in the `DCAT` vocabulary. There is one exception: Entity refers to an instance of the concept `prov:Entity` (of the `PROV-O` ontology). It is generally used in the context of this document to summarise all instances of concepts in the `DataID core` ontology which are subclasses of `prov:Entity`: `dataid:DataId`, `dataid:Dataset` and `dataid:Distribution`.

This measure provides specificity about the subject at hand, without having to define each occurrence anew.

4 The DataID Ecosystem

4.1 Problem Statement

The inadequacies of current metadata vocabularies are manifold and diverse (Section 3.1). As already introduced (Section 1.1), there are some issues which protrude from the rest, due to their ubiquitousness in use cases or their import on aspects like interoperability, trustworthiness and governance of data.

This list of important aspects of metadata reflects these issues and explains them in detail. In the course of this theses I will take particular care to resolve these issues thoroughly.

(A1) Provenance: a crucial aspect of data, required to assess correctness and completeness of data conversion, as well as the basis for the trustworthiness of the data source (no trust without provenance).

(A2) Licensing: machine-readable licensing information provides the possibility to automatically publish, distribute and consume only data that explicitly allows these actions.

(A3) Access: publishing and maintaining this kind of metadata together with the data itself serves as documentation benefiting the potential user of the data as well as the creator by making it discoverable and crawlable.

(A4) Extensibility: extending a given core metadata model in an easy and reusable way, while leaving the original model uncompromised expands its application possibilities fitting many different use cases.

(A5) Interoperability: the interoperability with other metadata models is a hallmark of a widely usable and reusable metadata vocabulary. It is a prerequisite for uniform access to digital resources on the Web.

Based on the issues explored in Section 3.1 and the important aspects of dataset metadata above, I conclude, not only is there a gap between existing dataset metadata vocabularies and requirements thereof, but it seems unlikely that one will be able to solve all these diverse problems with just one, monolithic ontology.

4.2 The multi-layer ontology of DataID

> "Metadata modularity is a key organizing principle for environments characterized by vastly diverse sources of content, styles of content management, and approaches to resource description. It allows designers of metadata schemas to create new assemblies based on established metadata schemas and benefit from observed best practice, rather than reinventing elements anew." [21]

While trying to solve the different aspects I discussed in the previous section, and tending to the needs of different usage scenarios, the DataID ontology grew in size and complexity (extending `DataID` Version 1.0.0 - cf. Section 3.1.10). In order to keep the `DataID` ontology reasonable in size and complexity as well as not to jeopardise Extensibility and Interoperability (cf. Section 3.3.3), I modularised `DataID` in a core ontology and multiple extensions. The onion-like layer model (Figure 4.1) illustrates the import dependencies between the ontologies:

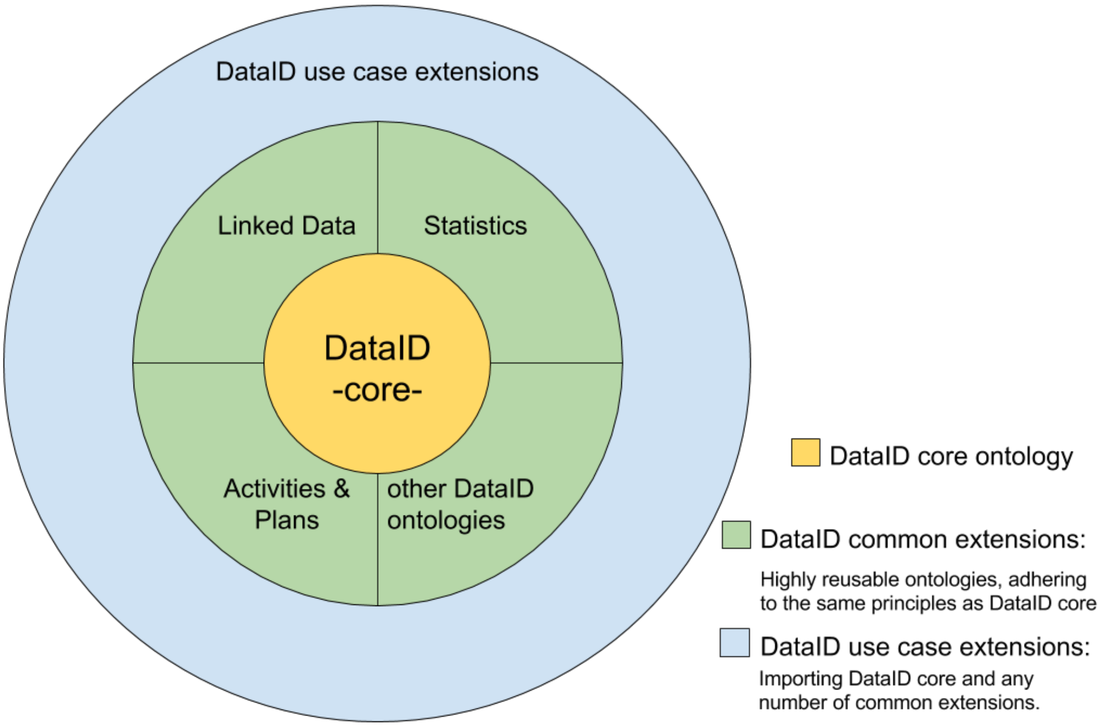
DataID use case extensions
Linked Data
Statistics
DataID
-core-
Activities &
Plans
other DataID
ontologies
DataID core ontology
DataID common extensions:
Highly reusable ontologies, adhering to the same principles as DataID core
DataID use case extensions:
Importing DataID core and any number of common extensions.

Figure 4.1: The Metadata Ecosystem of DataID

The scaling approach used to modularize the original `DataID` ontology adopts principles of the modular programming technique, separating concepts and properties of a large ontology into independent, interchangeable modules, specialised to fit common use cases, dependent only on DataID core. Thus, any vocabulary in this sphere must import DataID core, with the exception of DataID core itself. The mid-layer (or common extensions) of this model is comprised of highly reusable ontologies, extending DataID core to cover additional aspects of dataset metadata. None of these are mandatory imports for any ontology, yet, in many cases, some or all of them will be useful contributions. While I will not (and can not) impose any restrictions as to which ontologies not to import, ideally, ontologies of this layer should only import DataID core, to minimise discrepancies between mid-layer ontologies of different authors with overlapping purposes. The outermost layer of this sphere represents all vocabularies importing DataID core and any number if mid-layer ontologies and adding additional semantics to portray domain or use case-specific demands for metadata.

Ontologies under the DataID multilayer concept usually do not offer cardinality restrictions, making them easy to extend and adhere to OWL profiles (cf. also Section 5.1). An application profile, featuring restrictions like cardinalities for the DataID service stack, which is beeing developed at the moment (cf. Chapter 10), was declared using SHACL [68].

4.3 DataID Extensions of the Common Layer

The DataID Ecosystem is a suite of ontologies comprised of DataID core and its extensions, which were created to satisfy different use cases in a reusable manner. DataID core specifies the basic description of a dataset and serves as a foundation for all extensions to DataID. While the next chapter provides a detailed introduction to DataID core (cf. Chapter 5), this section offers a short overview of the already established landscape of extensions in the common layer of the DataID Ecosystem.

4.3.1 Linked Data

The Linked Data extension (prefix: dataid-ld) extends DataID core with an interface for the SPARQL Service Description vocabulary [77], which not only allows specifying SPARQL endpoints but offers properties useful for Linked Data datasets as well (such as sd:defaultGraph)[68]. A specialised sub-class of dataid:Dataset and sd:Dataset was introduced to better represent this type of data (dataid-ld:LinkedDataDataset). While the VoID vocabulary is already im-

[68] https://github.com/dbpedia/DataId-Ontology/tree/master/ld

ported in DataID core (cf. Section 5.1), many VoID and properties only become useful in this context (e.g. void:triples). In addition to the void:Linkset class, describing the relation between datasets, where one holds links to instances of the other, the statistical properties from VoID about entities in a Linked Data dataset are of interest (e.g. void:distinctObjects).

4.3.2 Activities & Plans

Activities & Plans extension (prefix: dataid-acp) provides provenance information of activities which generated, changed or used datasets[69]. The goal is to record all activities needed to replicate a dataset as described by a DataID graph. Instrumental to this effort is the further qualification of prov:wasAttributedTo property, pointing out agents responsible for activities which transform datasets. Plans can describe which steps (activities, precautionary measures) are put in place to reach a certain goal (such as a final dataset, data preservation tasks, etc). This extension relies heavily on the PROV-O ontology and incorporates the Data Lifecycle Ontology (cf. Section 3.2.7) of the ALIGNED project [69]. A more detailed summary of this extension (incl. Figure 7.1) is available in Section 7.2.

4.3.3 Other DataID ontologies

More extension ontologies, of similar general character as the ontologies of this layer, are already under way but are not officially released at the moment of writing.

Statistics (dataid-md) will provide the necessary measures to publish metadata about multi-dimensional data, such as statistics about datasets, based on the Data Cube Vocabulary[18].

Data Quality (dataid-dq) based on the Data Quality Vocabulary [38], this extension is only concerned with quality measurements of the datasets published, closing a evident gap in the current DataID metadata landscape.

MediaType Vocabulary (dataid-mt) is currently a namespace used to collect instances of dataid:MediaType to reuse them more easily. An full extension on the matter of more intricate depiction of media types (cf. Section 5.2.4) is also planned.

[69] https://github.com/dbpedia/DataId-Ontology/tree/DataManagementPlanExtension/acp

4.4 The landscaping approach of the DataID Ecosystem

Multiple requirements are planned to be enforced for the adoption of new ontologies into the common (or mid) layer of the DataID Ecosystem. They might contain (while not being restricted to):

- Authors must provide information about the reason for the new extension (and why the expected result is not achievable with existing extensions).
- Authors must document the Interoperability with other extensions of this layer (and where problems (such as semantic overlap) are to be expected).
- Authors must inform about conformity with OWL profiles.

The semantic regulations of interrelations between `DataID` ontologies are not yet finalised and are subject of further discussion. The general idea is to provide a controlled environment, in which the interplay of different extensions and `DataID core` is regulated, or at least sufficiently documented. Thus, providing a structured approach to combining multiple ontologies for a particular purpose, without having to be aware of every individual axiom. Possible side effects of combining two ontologies would be dealt with, or brought to the user's attention, by the authors of the involved extensions. This concept is my proposal of providing much-needed landscaping between ontologies, to confront such problems as detected in [24].

A useful tool to help integrate multiple ontologies is the Dacura Data Curation System[70]. It provides the Dacura Quality Service, "[...] which is designed to consume OWL and RDF linked data schemata and identify potential problems in their specifications. This reasoner uses a much less permissive interpretation than that of standard OWL to find issues which are likely to stem from specification errors, even in cases where they produce valid OWL models." [23] Such an approach is imperative to prevent logical and practical problems between different ontologies (cf. Section 1.1).

Furthermore, extending this ecosystem of dataset metadata with domain-specific OWL ontologies adds further opportunities for applications clustered around datasets. I will demonstrate the methodology of working with these ontologies to satisfy the metadata requirements of complex use cases in a best practice (Section 6.2) as well as an application of these practices in Chapter 7.

[70] `http://dacura.cs.tcd.ie`

5 DataID core Ontology

5.1 Overview

This section will provide an overview of the DataID core ontology and some background on basic design decisions made. Most of the vocabularies reused by this ontology have already been presented in Chapter 3 and need no further introduction.

DataID core developed out of a previous version of DataID (cf. Section 3.1.10) by following the aspiration to modularise DataID into a core-ontology, surrounded by its dependents. Many design decisions of DataID core were already introduced in its former version, which was designed to make DCAT, combined with the VoID and PROV-O vocabularies, fit the requirements of the DBpedia use case for a hierarchy of Linked Data datasets (cf. Section 3.2.9).

> "The DBpedia dataset, with its different versions and languages, multiple SPARQL endpoints and thousands of dump files with various content serves as one example of the complexity metadata models need to be able to express. We argue that the DCAT vocabulary as well as the established VoID vocabulary only provide a basic interoperability layer to discover data. In their current state, they still have to be expanded to fully describe datasets as complex as DBpedia [...]." [14]

DataID core is founded on two pillars: the DCAT and PROV-O ontologies. To incorporate DCAT as the basis of DataID, to further extend DCAT with PROV-O, introducing extensive provenance records in the process, and adding properties specific to the DBpedia use case was the original premise of this endeavour. Also, the VoID vocabulary was adopted to cover Linked Data specific semantics and provide more general properties, such as void:subset for establishing dataset hierarchies. The wide application of these vocabularies in the context of the Semantic Web was the rationale behind these decisions, furthering the goal of Interoperability.

DataID core is centred around the dataset and distribution concepts which were imported from DCAT. I introduced the class dataid:DataId (cf. Section 5.2.1),

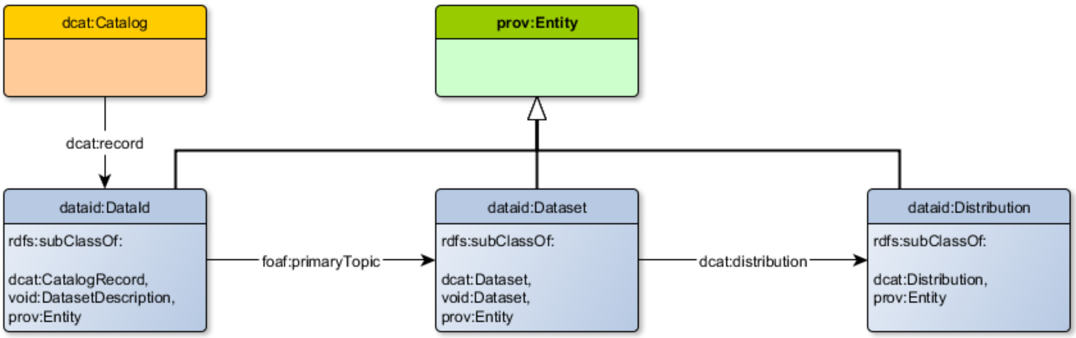

Figure 5.1: Foundations: Combining DCAT, VoID and PROV-O

merging both dcat:CatalogRecord and void:DatasetDescription into one, providing an additional level of abstraction between the conscepts of dataset and catalogue. Instances of this class can be compared to root elements in the XML documents, since all subsequent instances, describing a ***Dataset***, are hierarchically below a ***DataId*** instance. Though, this comparison is of course misleading, since a graph has no orientation. Nevertheless, most of DataID documents[71] will have structural similarities to XML documents.

All dataset related classes of DataID core (dataid:DataId, dataid:Dataset and dataid:Distribution) are sub-classes of prov:Entity, which is the interface needed to harness the possibilities of the Provenance Ontology (PROV-O). In the context of this description of DataID core, the word 'Entity' is used to refer to instances of prov:Entity (ergo: instances of these three classes).

With PROV-O, describing the circumstance of provenance for any domain is possible (Section 3.2.1). Central to this, is the concept of qualification classes, providing qualifications for more general properties (e.g. for prov:wasAttributedTo). Describing the interrelations between ***Entities*** (such as a particular ***Dataset*** or just a single ***Distribution*** of it) and ***Agents*** (e.g. a person or an organisation) are salient requirements in most environments for datasets and their metadata. Thus, DataID core has singled out these relations to further qualify them and provide much needed referential integrity.

explain

For example, the property dataid:associatedAgent (which is a sub-property of prov:wasAttributedTo) is a universal relation between an ***Entity*** and an ***Agent***. It can (and should) be qualified by an instance of dataid:Authorization, a sub-class of prov:Attribution. An ***Authorization*** adds qualifications and restrictions to the original property, such as an agent role (defining the role the ***Agent*** has in regard to the ***Entities*** involved - see Section 5.2.6). DataID core

[71] a DataID graph serialised as a metadata document

allows for assigning Actions to an ***Authorization***, which specify what an ***Agent*** can do and for which tasks an ***Agent*** is responsible.

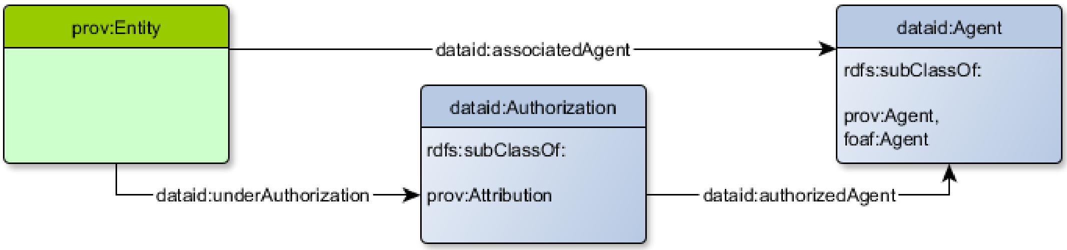

Figure 5.2: Foundations: Using PROV-O to qualify properties

In its current version 2.0.0, DataID core offers a general approach for describing dataset metadata, incorporating important ontologies to extend DCAT with Provenance, qualifying relations and their intended range classes, hierarchical dataset structures, management of rights and responsibilities of ***Agents*** and exhaustive descriptions for data Access.

In general, DataID core forbore the use of cardinality restrictions[72], making it easier to adhere to the OWL 2 RL (cf. Section 2.2.2) profile and maintaining the benefits of easy Extensibility and Interoperability prevalent with DCAT. In turn, DataID core restricts some of the very general property ranges of Dublin Core and DCAT properties (such as dct:license or dcat:mediaType), to reduce impreciseness and increase machine-readability.

DataID core is making use of the following namespace:

```
http://dataid.dbpedia.org/ns/core#
```

At this URI, an extended description of each class and property, introduced by this ontology, can be found as well as the complete ontology specification in Turtle[73] or OWL[74] serialisation. The current version is available in Appendix I.

For a better understanding of the imported concepts and properties, which are reused by DataID core, I would advise to read up on DCAT [49] and PROV-O [52] (since these are used most commonly), to gain a more complete picture of the underlying structure and rationale of these ontologies. While I gave a concise introduction to both in Section 3.1.1 and Section 3.2.1 respectively, this work is not the place to repeat this in more detail.

[72] this is the purpose of an application profile, not of an ontology
[73] http://dataid.dbpedia.org/ns/core.ttl
[74] http://dataid.dbpedia.org/ns/core.owl

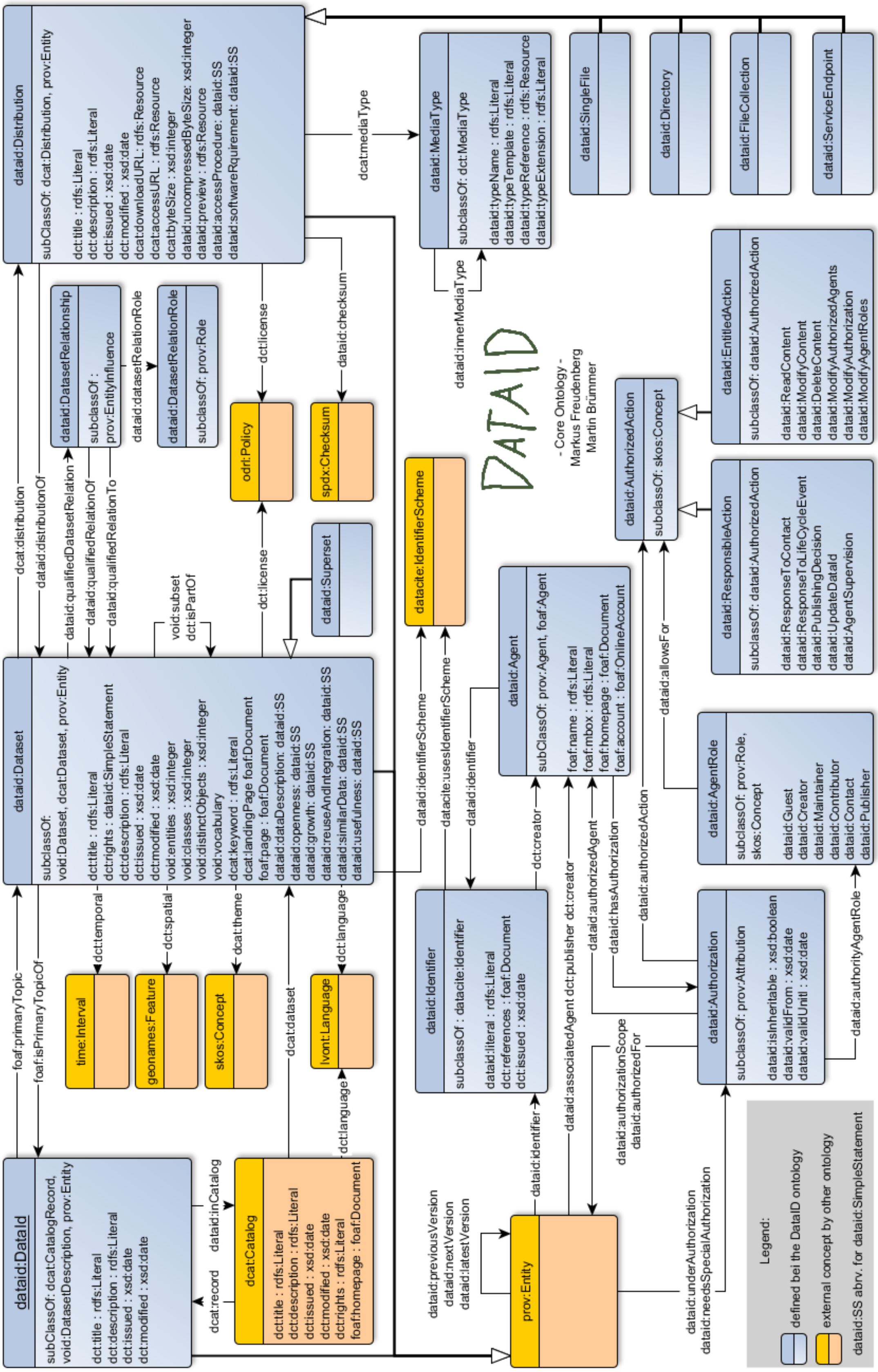
dataid:DataId
subClassOf: dcat:CatalogRecord, void:DatasetDescription, prov:Entity
dct:title : rdfs:Literal
dct:description : rdfs:Literal
dct:issued : xsd:date
dct:modified : xsd:date
dcat:record
dataid:inCatalog
dcat:Catalog
dct:title : rdfs:Literal
dct:description : rdfs:Literal
dct:issued : xsd:date
dct:modified : xsd:date
dct:rights : rdfs:Literal
foaf:homepage : foaf:Document
foaf:primaryTopic
foaf:isPrimaryTopicOf
dct:language
dcat:dataset
time:Interval
geonames:Feature
skos:Concept
lvont:Language
dct:temporal
dct:spatial
dcat:theme
dct:language
dataid:Dataset
subclassOf: void:Dataset, dcat:Dataset, prov:Entity
dct:title : rdfs:Literal
dct:rights : dataid:SimpleStatement
dct:description : rdfs:Literal
dct:issued : xsd:date
dct:modified : xsd:date
void:entities : xsd:integer
void:classes : xsd:integer
void:distinctObjects : xsd:integer
void:vocabulary
dcat:keyword : rdfs:Literal
dcat:landingPage foaf:Document
foaf:page : foaf:Document
dataid:dataDescription: dataid:SS
dataid:openness: dataid:SS
dataid:growth: dataid:SS
dataid:reuseAndIntegration: dataid:SS
dataid:similarData: dataid:SS
dataid:usefulness: dataid:SS
dcat:distribution
dataid:distributionOf
dataid:qualifiedDatasetRelation
dataid:qualifiedRelationOf
dataid:qualifiedRelationTo
void:subset
dct:isPartOf
dct:license
dataid:DatasetRelationship
subclassOf : prov:EntityInfluence
dataid:datasetRelationRole
dataid:DatasetRelationRole
subclassOf: prov:Role
odrl:Policy
spdx:Checksum
dataid:Superset
dct:license
dataid:checksum
dataid:Distribution
subClassOf: dcat:Distribution, prov:Entity
dct:title : rdfs:Literal
dct:description : rdfs:Literal
dct:issued : xsd:date
dct:modified : xsd:date
dcat:downloadURL : rdfs:Resource
dcat:accessURL : rdfs:Resource
dcat:byteSize : xsd:integer
dataid:uncompressedByteSize: xsd:integer
dataid:preview : rdfs:Resource
dataid:accessProcedure: dataid:SS
dataid:softwareRquirement: dataid:SS
dcat:mediaType
dataid:MediaType
subClassOf: dct:MediaType
dataid:typeName : rdfs:Literal
dataid:typeTemplate : rdfs:Literal
dataid:typeReference : rdfs:Resource
dataid:typeExtension : rdfs:Literal
dataid:innerMediaType
dataid:SingleFile
dataid:Directory
dataid:FileCollection
dataid:ServiceEndpoint
DATAID
- Core Ontology -
Markus Freudenberg
Martin Brümmer
datacite:IdentifierScheme
dataid:identifierScheme
datacite:usesIdentifierScheme
dataid:identifier
dataid:Identifier
subclassOf : datacite:Identifier
dataid:literal : rdfs:Literal
dct:references : foaf:Document
dct:issued : xsd:date
dataid:previousVersion
dataid:nextVersion
dataid:latestVersion
dataid:identifier
prov:Entity
dct:creator
dataid:associatedAgent dct:publisher dct:creator
dataid:authorizedAgent
dataid:hasAuthorization
dataid:authorizedAction
dataid:authorizationScope
dataid:authorizedFor
dataid:Agent
subClassOf: prov:Agent, foaf:Agent
foaf:name : rdfs:Literal
foaf:mbox : rdfs:Literal
foaf:homepage : foaf:Document
foaf:account : foaf:OnlineAccount
dataid:AuthorizedAction
subclassOf: skos:Concept
dataid:allowsFor
dataid:ResponsibleAction
subClassOf: dataid:AuthorizedAction
dataid:ResponseToContact
dataid:ResponseToLifeCycleEvent
dataid:PublishingDecision
dataid:UpdateDataId
dataid:AgentSupervision
dataid:EntitledAction
subClassOf: dataid:AuthorizedAction
dataid:ReadContent
dataid:ModifyContent
dataid:DeleteContent
dataid:ModifyAuthorizedAgents
dataid:ModifyAuthorization
dataid:ModifyAgentRoles
dataid:AgentRole
subClassOf: prov:Role, skos:Concept
dataid:Guest
dataid:Creator
dataid:Maintainer
dataid:Contributor
dataid:Contact
dataid:Publisher
dataid:Authorization
subClassOf: prov:Attribution
dataid:isInheritable : xsd:boolean
dataid:validFrom : xsd:date
dataid:validUntil : xsd:date
dataid:authorityAgentRole
dataid:underAuthorization
dataid:needsSpecialAuthorization
Legend:
defined bei the DataID ontology
external concept by other ontology
dataid:SS abrv. for dataid:SimpleStatement

Figure 5.3: DataID core

5.2 Classes

This section is partitioned by concepts to introduce the DataID core ontology. Each class is presented with an item of written comment about the general reasoning behind its existence and its intended use. For illustration purposes, a running example was woven into the descriptions of concepts and properties. This example is a reduced version of an original DataID document of the Arabic DBpedia (release: 2015-10[75]). Under the main ***Dataset***, only two ***Sub-Datasets*** are shown (as opposed to over 50 in the real world example from which this is drawn). Each Dataset has two distributions for two different RDF serialisations (Turtle ′.ttl′ and Quad-Turtle ′.tql′). One is also represented by the official DBpedia SPARQL endpoint. This example was chosen to cover many aspects of DataID core and to provide an easy use case which could arise in a similar fashion outside the DBpedia domain. The full example is available in *Turtle* serialisation[76] in Appendix II. The basic structure of its DataID document is outlined below (Figure 5.14).

For the purpose of this running example the following base URI was used to shorten all subsequent URIs:

```
@base <http://downloads.dbpedia.org/2015-10/core-i18n/ar/2015-10_>
```

The example often omits the more common properties of Dublin Core (cf. Section 2.1.4) and RDFS (cf. Section 2.2.2) such as dct:title, dct:description, dct:modified, dct:issued and rdfs:label to make this example more easy to read.

In addition, this running example does not only use DataID core, but it also introduces a first DataID extension from the common layer of the DataID Ecosystem (Section 4.2) for Linked Data datasets. The Linked Data extension[77] (prefix: dataid-ld) is used to describe dataset attributes, specific to Linked Data.

The remainder of this section features each subsection with a summary, stating the purpose of each concept, a list of properties commonly used to describe instances of this concept, a schematic depiction of the concept, followed by at least one example instance of this class (taken from the running example of DBpedia). The list of properties features a rating for each property to advise whether an application profile based on DataID core should declare a property to be: mandatory **(M)**, recommended **(R)** or optional **(O)** (again: this is only a recommendation, DataID core does not provide cardinality restrictions).

[75] http://vmdbpedia.informatik.uni-leipzig.de/dataid-w3c-submission/html/2015-10_dataid_ar.ttl

[76] https://www.w3.org/TeamSubmission/turtle/

[77] https://github.com/dbpedia/DataId-Ontology/tree/master/ld

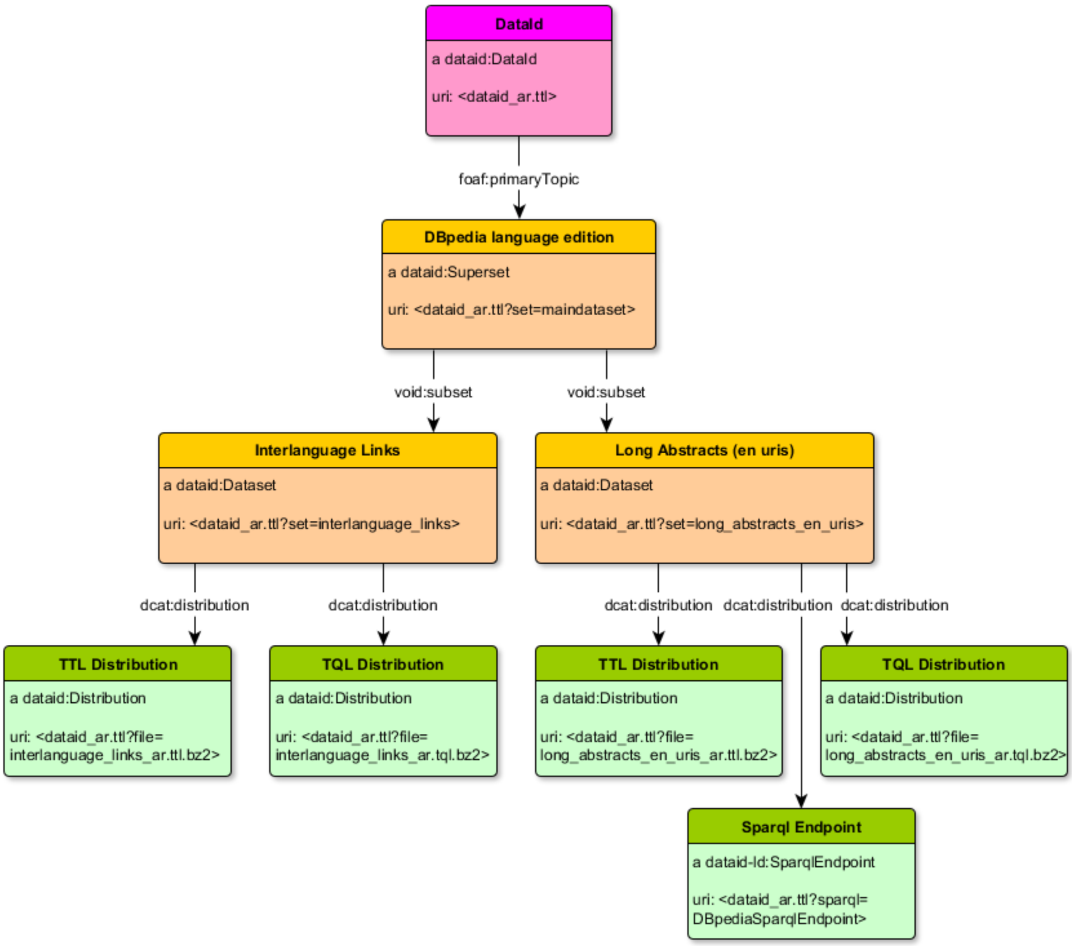

Figure 5.4: A shallow dataset hierarchy of the Arabic DBpedia language edition.

5.2.1 DataId

The class `dataid:DataId` inherits from `dcat:CatalogRecord` and `void:DatasetDescription`, which does not represent a dataset, but metadata about a dataset's entry in a catalogue. Additionally, in the context of `DataID core`, it represents metadata about a `DataID` document (or the graph serialised in it), such as version pointers, modification dates and relations to contextual ***Entities*** (agents, catalogues, repositories). This `DataID` resource is the most abstract ***Entity*** in any `DataID` graph.

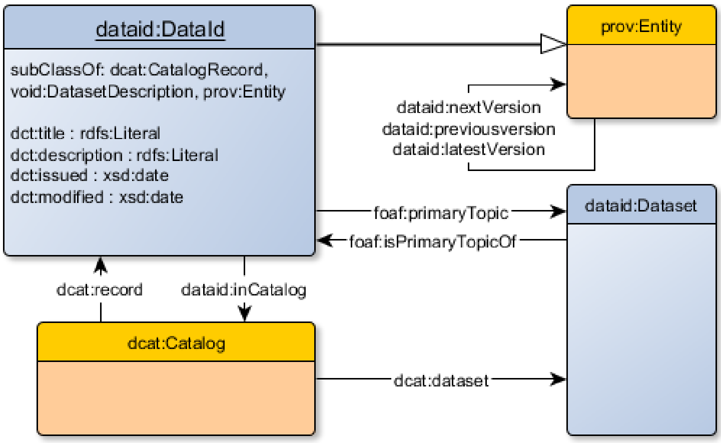

Figure 5.5: The DataId concept

Properties specifically used with `dataid:DataId`:

`dataid:inCatalog`: the inverse of `dcat:record` references the `DCAT` catalogue a `DataID` is entered in. **(R)**

`foaf:primaryTopic`: this functional property is used to point out the ***Dataset*** resource this `DataID` represents. **(M)**

Additionally, the following list contains properties which can be used with any sub-class of `prov:Entity`, presented only for this concept out of convenience:

`dataid:associatedAgent`: points out all ***Agents*** which have some (unspecified) influence (or authority) over this ***Entity***. **(O)**

`dataid:underAuthorization`: refers to an instance of `dataid:Authorization` which qualifies the generic `dataid:associatedAgent` relation (cf. Section 5.2.6). **(R)**

`dct:publisher`: While the `PROV-O` based way to declare associated ***Agents*** and their roles in regard to an ***Entity*** is in place, using established properties to point out ***Agents*** (such as `dct:publisher` of `dcat:contactPoint`) is not redundant and should be kept as best practice. **(R)**

`dataid:identifier:` similar to ***Agents***, ***Entities*** often have unique identifiers defined for them in a different context (in addition to the URI used in the context of a `DataID` document). This property points out additional identifiers (cf. Section 5.2.9). **(O)**

`dataid:nextVersion:` is used to identify instances of the same type (in this case `dataid:DataId`), which are next in the chain of different versions of this resource. **(O)**

`dataid:previousVersion:` points out the previous version of this instance. **(R)**

`dataid:latestVersion:` points out the latest version of this instance. **(R)**

The example below illustrates the use of this concept. It provides a pertaining dataset catalogue (via `dataid:inCatalog`), uses version pointers to define its position in a branch of a versioning system, points out ***Agents*** and ***Authorizations***, and the ***Dataset*** this `DataID` is about (`foaf:primaryTopic`).

```
<dataid_ar.ttl>
        a                       dataid:DataId ;
        dataid:associatedAgent  <http://wiki.dbpedia.org/dbpedia-association> ;
        dataid:inCatalog        <http://downloads.dbpedia.org/2015-10/2015-10_dataid_catalog.ttl>
             ;
        dataid:latestVersion    <http://downloads.dbpedia.org/2016-04/core-i18n/ar/2016-04
             _dataid_ar.ttl> ;
        dataid:nextVersion      <http://downloads.dbpedia.org/2016-04/core-i18n/ar/2016-04
             _dataid_ar.ttl> ;
        dataid:previousVersion  <http://downloads.dbpedia.org/2015-04/core-i18n/ar/2015-04
             _dataid_ar.ttl> ;
        dataid:underAuthorization  <dataid_ar.ttl?auth=creatorAuthorization> ;
        dct:hasVersion          <dataid_ar.ttl?version=1.0.0> ;
        dct:publisher           <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title               "DataID metadata for the Arabic DBpedia"@en ;
        foaf:primaryTopic       <dataid_ar.ttl?set=maindataset> .
```

Listing 5.1: Instance of a DataId

5.2.2 Dataset

This is the central concept of the `DataID core` ontology and offers a multitude of useful properties to describe a dataset comprehensively.

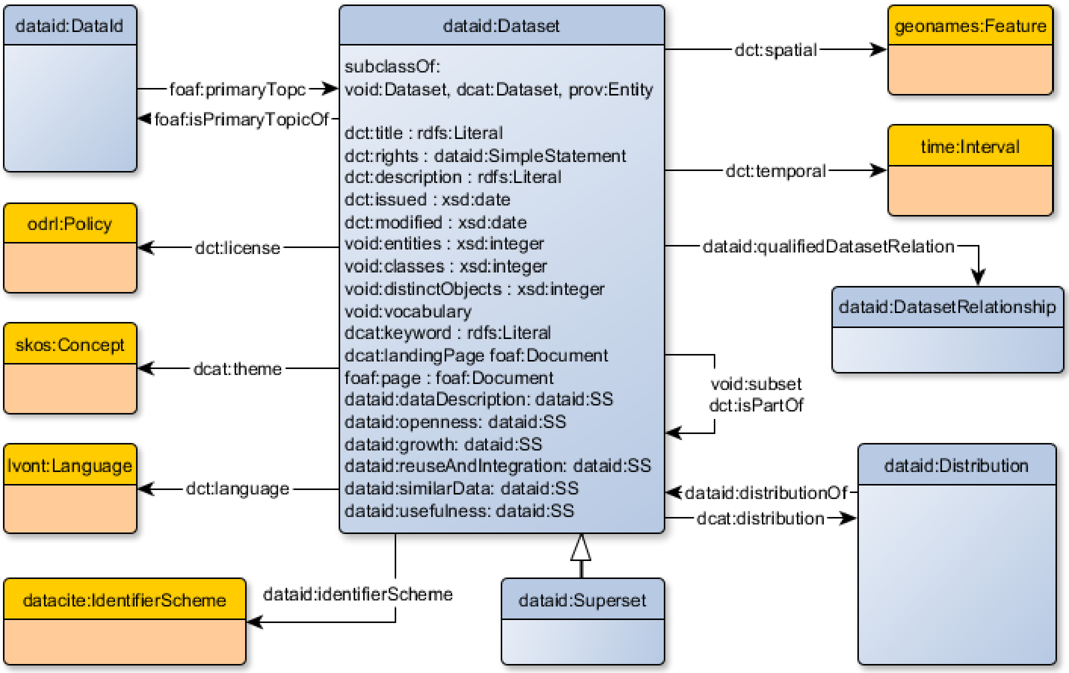
dataid:DataId
foaf:primaryTopc
foaf:isPrimaryTopicOf
dataid:Dataset
subclassOf:
void:Dataset, dcat:Dataset, prov:Entity
dct:title : rdfs:Literal
dct:rights : dataid:SimpleStatement
dct:description : rdfs:Literal
dct:issued : xsd:date
dct:modified : xsd:date
void:entities : xsd:integer
void:classes : xsd:integer
void:distinctObjects : xsd:integer
void:vocabulary
dcat:keyword : rdfs:Literal
dcat:landingPage foaf:Document
foaf:page : foaf:Document
dataid:dataDescription: dataid:SS
dataid:openness: dataid:SS
dataid:growth: dataid:SS
dataid:reuseAndIntegration: dataid:SS
dataid:similarData: dataid:SS
dataid:usefulness: dataid:SS
dct:spatial
geonames:Feature
dct:temporal
time:Interval
dataid:qualifiedDatasetRelation
dataid:DatasetRelationship
void:subset
dct:isPartOf
dataid:distributionOf
dcat:distribution
dataid:Distribution
odrl:Policy
dct:license
skos:Concept
dcat:theme
lvont:Language
dct:language
datacite:IdentifierScheme
dataid:identifierScheme
dataid:Superset

Figure 5.6: The Dataset concept

The dataset concept of both the `DCAT` and `VoID` vocabularies were merged into `dataid:Dataset`, providing useful properties about the content of a ***Dataset*** from both ontologies. In particular, the property `void:subset` allows for the creation of dataset hierarchies, while `dcat:distribution` points out the ***Distributions*** of a ***Dataset***. The `dataid:Superset` as a subclass of `dataid:Dataset` shall be used to represent multiple ***Sub-Dataset***, representing dataset collections or hierarchical dataset structures in general. Opposed to a conventional ***Dataset***, a ***Superset*** is prohibited from possessing ***Distributions*** (referred to with `dcat:distribution`). It is strongly recommended that each ***Dataset*** shall either have at least one ***Distribution*** or one ***Sub-Dataset***.

In the running example, the main ***Dataset*** (an instance of `dataid:Superset`), is used as a hierarchical top entity, representing all ***Sub-Datasets*** clustered around a common topic. In the case of `DBpedia`, all ***Datasets*** were arranged under a ***Superset*** representing a `DBpedia` language edition.

Properties commonly used with instances of this class are:

dcat:distribution provides the ***Distributions*** of a ***Dataset*** and should be used distinct from void:subset. **(M for non Supersets)**

void:subset is the property used to reference ***Sub-Datasets*** which are part of a ***Superset***. Its use is limited to dataid:Superset. **(M if Superset)**

dct:license this property is restricted in the context of DataID to instances of the class odrl:Policy (cf. Section 3.2.2)of the ODRL ontology, providing machine-readable licensing information. **(M)**

dct:language: this property from the Dublin Core vocabulary is used to point out the predominant language used within a ***Dataset***. Since a rdfs:range statement is not provided in its original definition, even simple literals could be used with this predicate. DataID core restricts the range of this property to instances of lvont:Language from the Lexvo ontology (cf. Section 3.2.3), describing human languages in a machine readable way. **(R)**

void:vocabulary in the VoID vocabulary, this property is used to point out the associated ontology used to create a ***Dataset***. DataID core broadens its use so that any schema document (e.g. XSD or database schemata) can be pointed out, depending on the type of data. **(R)**

foaf:page is commonly used to point out web pages which have the function of a manual or documentation for the resource at hand. **(R)**

dct:rights this property is used to provide a statement or resource about issues like copyrights or legal notes. To avoid unqualifiable literal statements, this property is restricted to be used with dataid:SimpleStatement (cf. **??**). **(O)**

foaf:isPrimaryTopicOf is the inverse property of foaf:primaryTopic (cf. Section 5.2.1). **(O)**

dataid:relatedDataset: generic pointer to related ***Datasets*** (sub-property of dct:relation), qualifiable by dataid:qualifiedDatasetRelation. Refer to Section 5.2.8 for more. **(O)**

dataid:qualifiedDatasetRelation provides an instance of concept dataid:DatasetRelationship which is a qualification class for the generic property dataid:relatedDataset (cf. Section 5.2.8). **(O)**

dataid:identifierScheme: provides a resource, describing the identifiers used within the ***Dataset*** to uniquely identify a record (data point, datum). Refer to Section 5.2.9 to learn more about ***Identifiers***. **(O)**

dcat:keyword: a simple keyword or tag associated with this ***Dataset*** **(O)**

`dcat:landingpage`: references a web page of general character (such as the homepage of an organisation) **(O)**

Multiple properties for textual statements on different aspects of a ***Dataset*** were created. All of which provide ***Publishers***, ***Maintainers*** and other ***Agents*** a way to convey statements of general character on such subjects as described below. This information will be useful in many scenarios related to dissemination tasks, for example, those described by the Horizon 2020[78] data management plan guidelines[79] (see also Chapter 7).

All of these properties listed below have the common `rdfs:range` of `dataid:SimpleStatement`, which allows to either provide a literal or reference a web page containing the textual information about the resource (cf. Section 5.2.10).

`dataid:dataDescription`: provides a detailed textual description of the data represented by this ***Dataset***. **(O)**

`dataid:growth`: an indication of what size the approximated end volume of the ***Dataset*** is. **(O)**

`dataid:usefulness`: is used to state to whom the ***Dataset*** could be useful, and whether it underpins a scientific publication. **(O)**

`dataid:similarData`: a statement on the existence (or absence) of similar data (see also `dataid:relatedDataset`). **(O)**

`dataid:reuseAndIntegration`: information on the possibilities for integration and reuse of the ***Dataset***. **(O)**

`dataid:openness`: General description of how data will be shared. For example embargo periods (if any), outlines of technical mechanisms for dissemination or a definition of whether access will be widely open or restricted to specific groups. In case the ***Dataset*** cannot be shared, the reasons for this should be mentioned (e.g. ethical, rules of personal data, intellectual property, commercial, privacy-related, security-related). **(O)**

[78] `https://ec.europa.eu/programmes/horizon2020/`

[79] `https://ec.europa.eu/research/participants/data/ref/h2020/grants_manual/hi/oa_pilot/h2020-hi-oa-data-mgt_en.pdf`

The running example is structured in a shallow hierarchy with an instance of `dataid:Superset` representing all ***Datasets*** of an Arabic `DBpedia` language edition:

```
<dataid_ar.ttl?set=maindataset>
        a                           dataid:Superset ;
        dataid:associatedAgent      <http://wiki.dbpedia.org/dbpedia-association>  ;
        dataid:growth               <dataid_ar.ttl?stmt=growth> ;
        dataid:openness             <dataid_ar.ttl?stmt=openness> ;
        dataid:reuseAndIntegration  <dataid_ar.ttl?stmt=reuseAndIntegration> ;
        dataid:similarData          <dataid_ar.ttl?stmt=similarData> ;
        dataid:usefulness           <dataid_ar.ttl?stmt=usefulness> ;
        dct:hasVersion               <dataid_ar.ttl?version=1.0.0> ;
        dct:language                 <http://lexvo.org/id/iso639-3/ara> ;
        dct:license                  <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:publisher                <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:rights                   <dataid_ar.ttl?rights=dbpedia-rights> ;
        void:subset                 <dataid_ar.ttl?set=long_abstracts_en_uris>,
        <dataid_ar.ttl?set=interlanguage_links> ;
        void:vocabulary             <http://downloads.dbpedia.org/2015-04/dbpedia_2015-10.owl> ;
        dcat:keyword                "maindataset"@en , "DBpedia"@en ;
        dcat:landingPage            <http://dbpedia.org/> ;
        foaf:isPrimaryTopicOf       <dataid_ar.ttl> ;
        foaf:page                   <http://wiki.dbpedia.org/Downloads2015-10> .
```

Listing 5.2: Instance of a Superset

This ***Dataset*** has no ***Distributions***, the data it represents is referred to via its ***Sub-Datasets***. As an example for one of its ***Sub-Datasets***, the following listing exemplifies this difference (use of `dcat:distribution` instead of `void:subset`):

```
<dataid_ar.ttl?set=long_abstracts_en_uris>
        a                       dataid:Dataset, dataid-ld:LinkedDataDataset ;
        dataid:associatedAgent  <http://wiki.dbpedia.org/dbpedia-association> ;
        dataid:qualifiedDatasetRelation   <dataid_ar.ttl?relation=source&target=pages_articles> ;
        dataid:relatedDataset   <dataid_ar.ttl?set=pages_articles> ;
        dct:hasVersion           <dataid_ar.ttl?version=1.0.0> ;
        dct:isPartOf             <dataid_ar.ttl?set=maindataset> ;
        dct:language             <http://lexvo.org/id/iso639-3/ara> ;
        dct:license              <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:title                "long abstracts en uris"@en ;
        void:rootResource       <dataid_ar.ttl?set=maindataset> ;
        void:triples            232801 ;
        void:sparqlEndpoint      <http://dbpedia.org/sparql> ;
        dcat:distribution       <dataid_ar.ttl?sparql=DBpediaSparqlEndpoint> ,
            <dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2> ,
            <dataid_ar.ttl?file=long_abstracts_en_uris_ar.tql.bz2> ;
        dcat:keyword            "long_abstracts_en_uris"@en , "DBpedia"@en ;
        dcat:landingPage        <http://dbpedia.org/> ;
        sd:defaultGraph         <http://ar.dbpedia.org> ;
        foaf:page               <http://wiki.dbpedia.org/Downloads2015-10> .
```

Listing 5.3: Instance of a Dataset

5.2.3 Distribution

The class `dataid:Distribution` provides the technical description of the data, the ’manifestation’ of a ***Dataset***. In addition, it serves as documentation of how to access the data described (e.g. `dcat:accessURL`), and which conditions apply (e.g. `dataid:accessProcedure`). Every ***Distribution*** of a ***Dataset*** must contain all data of the ***Dataset*** in the format and location described. It may contain additional data exceeding the defined ***Dataset***, for example when describing a service endpoint. Two ***Distributions*** of the same ***Dataset***, therefore, must either contain the same data (for example in two different serialisations), or one ***Distribution*** must completely subsume the other. The ***Distribution*** concept, introduced by the `DCAT` vocabulary, is crucial to be able to automatically retrieve and use the data described in a `DataID` document, simplifying, for example, data analysis. Additional sub-classes, to further distinguish how the data is available on the Web, were introduced:

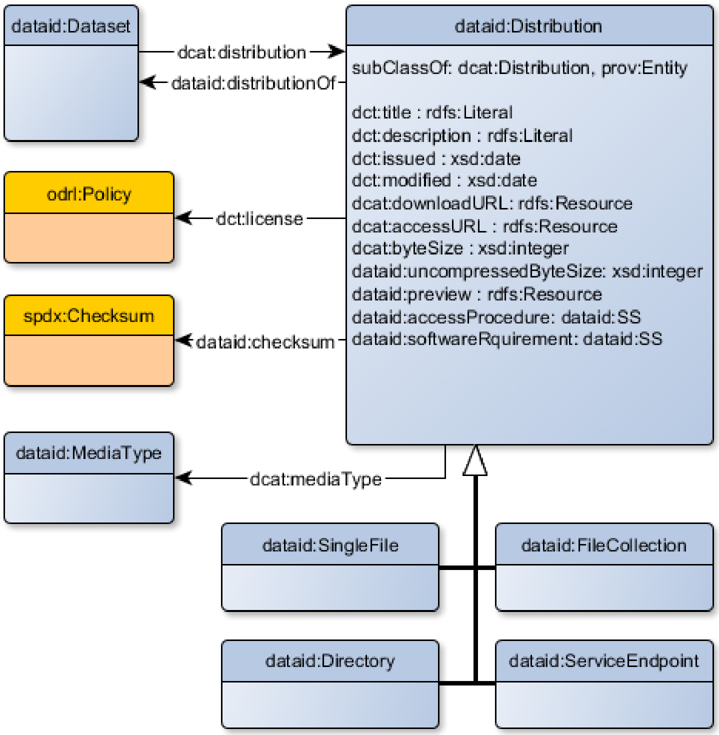
dataid:Dataset
dcat:distribution
dataid:distributionOf
dataid:Distribution
subClassOf: dcat:Distribution, prov:Entity
dct:title : rdfs:Literal
dct:description : rdfs:Literal
dct:issued : xsd:date
dct:modified : xsd:date
dcat:downloadURL: rdfs:Resource
dcat:accessURL : rdfs:Resource
dcat:byteSize : xsd:integer
dataid:uncompressedByteSize: xsd:integer
dataid:preview : rdfs:Resource
dataid:accessProcedure: dataid:SS
dataid:softwareRquirement: dataid:SS
odrl:Policy
dct:license
spdx:Checksum
dataid:checksum
dataid:MediaType
dcat:mediaType
dataid:SingleFile
dataid:FileCollection
dataid:Directory
dataid:ServiceEndpoint

Figure 5.7: The Distribution concept

`dataid:SingleFile`: all data of a ***Dataset*** is available in a single file.

`dataid:Directory`: the data of a ***Dataset*** is represented by the files in a single Directory on a file system.

`dataid:FileCollection`: an arbitrary collection of (different files, not restricted to one file system), which in their accumulation represent the data of a *Dataset*.

`dataid:ServiceEndpoint`: is a superclass for all service/api endpoints which could provide datasets. For example, REST APIs for databases[80] or SPARQL endpoints.

Except for `dataid:SingleFile`, all of these subclasses may need additional semantics to describe them in a useful manner. This is not an objective of `DataID core`, further extensions of this ontology will address these issues.

Properties used with `dataid:Distribution`:

`dcat:mediaType`: is restricted to a range of `dataid:MediaType` (cf. Section 5.2.4). This property is absolutely necessary to digest the content of a ***Distribution*** automatically. **(M)**

`dcat:downloadURL`: this property supplies a URL under which the described ***Distribution*** can be downloaded directly (without any access procedure or intermediate steps) and completely. **(M if no `dcat:accessURL`)**

`dcat:accessURL`: when additional steps are necessary to achieve access to the data of this ***Distribution*** (such as authorisations, querying or selecting data from a repository), `dcat:accessURL` is used to provide the initial URL. If the process necessary to retrieve the data is not evident by the content of the web page referred, publishers should make use of property. `dataid:accessProcedure`. **(M if no `dcat:downloadURL`)**

`dataid:accessProcedure`: is used to convey a description of necessary steps, needed to retrieve the data of this ***Distribution*** from the `dcat:accessURL`. This property has the range`dataid:SimpleStatement` and should not be used in connection with `dcat:downloadURL`. **(R if no `dcat:downloadURL`)**

`dataid:checksum`: the range of this property is restricted to `spdx:Checksum` [47], providing `spdx:checksumValue` and `spdx:algorithm` properties for an exact definition of a checksum. Checksums can be used to validate the correctness of downloaded files, directories and API endpoint responses. **(R)**

`dcat:byteSize`: The exact size of a ***Distribution*** in bytes. **(R)**

`dataid:softwareRequirement`: Some data formats/serialisations are only useful with a particular software product. This statement offers the possibility to name such circumstances. **(O)**

[80] `https://docs.oracle.com/cloud/latest/mysql-cloud/CSMCS/index.html`

dataid:uncompressedByteSize: Often, files and media streams are compressed to reduce the number of bytes to be transferred. This optional property provides the means to specify the size of the ***Distribution*** in its uncompressed extension. **(O)**

dataid:preview: While the exact format and serialisation of the ***Distribution*** is defined by `dataid:MediaType`, it is often beneficial for publishers and consumers to have a look at the actual format of the (uncompressed) data. This property points out a web resource providing such a preview. **(O)**

dataid:distributionOf: the inverse property of `dcat:distribution`, pointing back to the ***Dataset*** instance this ***Distribution*** belongs to. **(O)**

dct:license: see Section 5.2.2 for a detailed description. Often, the license used for the ***Distribution*** is the same as for the pertaining ***Dataset***. For those cases where this is not true, the use of this property in the context of a ***Distribution*** is advised. **(O)**

The first example is an instance of `dataid:SingleFile`, describing a single RDF file (which contains the whole ***Dataset***) in Turtle syntax, compressed with the bzip2 compression[81]. It can be downloaded directly (without any intermediate steps), hence the property `dcat:downloadURL` is used to point out the resource on the Web. Since it is a compressed file, the byte size in its compressed and uncompressed state is provided. An instance of `spdx:Checksum` was included, providing the checksum value for this ***Distribution***.

```
<dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2>
        a                         dataid:SingleFile ;
        dct:license               <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:publisher             <http://wiki.dbpedia.org/dbpedia-association> ;
        dataid:associatedAgent    <http://wiki.dbpedia.org/dbpedia-association> ;
        dataid:checksum           <dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2&checksum=md5>
             ;
        dataid:isDistributionOf   <dataid_ar.ttl?set=long_abstracts_en_uris> ;
        dataid:preview            <http://downloads.dbpedia.org/preview.php?file=2015-10_sl_core-
             i18n_sl_ar_sl_long_abstracts_en_uris_ar.ttl.bz2> ;.
        dataid:uncompressedByteSize       186573907 ;
        dcat:byteSize                     33428372 ;
        dcat:downloadURL          <long_abstracts_en_uris_ar.ttl.bz2> ;
        dcat:mediaType            dataid:MediaType_turtle_x-bzip2

<dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2&checksum=md5>
        a                    spdx:Checksum ;
        spdx:algorithm       spdx:checksumAlgorithm_md5 ;
        spdx:checksumValue   "2503179cd96452d33becd1e974d6a163"^^xsd:hexBinary .
```

Listing 5.4: Example of a SingleFile Distribution

[81] http://www.bzip.org

The second example is an instance of dataid-ld:SparqlEndpoint, a sub-class of dataid:ServiceEndpoint and sd:Service which was introduced with the DataID extension for Linked Data (cf. Section 4.3). Additional properties from the SPARQL 1.1 Service Description Language are used to describe the endpoint further. As opposed to the previous example, this SPARQL endpoint provides multiple ***Datasets*** at once in the context of the original DataID from DBpedia.

```
<dataid_ar.ttl?sparql=DBpediaSparqlEndpoint>
        a                         dataid-ld:SparqlEndpoint ;
        dataid:associatedAgent    <http://support.openlinksw.com/> ;
        dataid:accessProcedure    <dataid_ar.ttl?stmt=sparqlaccproc> ;
        dct:hasVersion            <dataid_ar.ttl?version=1.0> ;
        dct:license               <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dcat:accessURL            <http://dbpedia.org/sparql> ;
        dcat:mediaType            <http://dataid.dbpedia.org/ns/mt#MediaType_sparql-results+xml> ;
        sd:endpoint               <http://dbpedia.org/sparql> ;
        sd:supportedLanguage      sd:SPARQL11Query ;
        sd:resultFormat           <http://www.w3.org/ns/formats/RDF_XML>,
            <http://www.w3.org/ns/formats/Turtle> .
```

Listing 5.5: Example Distribution of a SPARQL endpoint

5.2.4 MediaType

DCAT does not offer an intrinsic way of specifying the exact format of the content described by a ***Distribution***. While the property dcat:mediaType does exist, its expected range dct:MediaTypeOrExtend is an empty concept, not extended by DCAT. Therefore, the dataid:MediaType was introduced to qualify this crucial piece of information in a better way. The following properties are of interest:

dataid:typeTemplate: the IANA[82] media type[83] - also named mime type [27]. This property is of utmost importance for the automatic processing of content. Through its registration with the IANA, each type is unambiguously defined and the format of data content can be interpreted in automated processes (e.g. 'text/turtle'). **(M)**

dataid:typeName: name of the described format or serialisation. **(R)**

dataid:typeExtension: a common file extension for the type described (e.g. '.ttl'). **(O)**

[82] http://www.iana.org

[83] http://www.iana.org/assignments/media-types/media-types.xhtml

`dataid:typeReference`: in some instances, a reference to a useful resource about a type is advisable, to further aid the apprehension of consumer agents (person or software). **(O)**

`dataid:innerMediaType`: with this property, descriptions of nested formats become possible (such as a compressed XML file - '.xml.bz2'), useful in pipeline processing. **(O)**

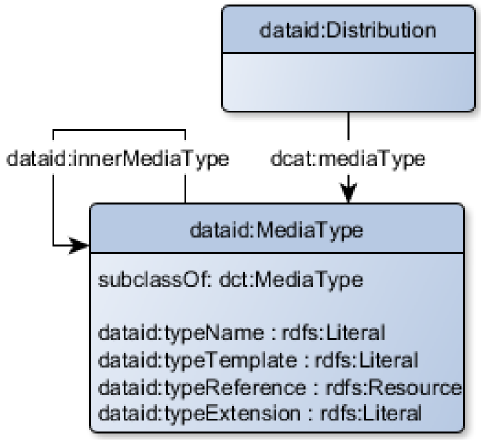
dataid:Distribution
dataid:innerMediaType
dcat:mediaType
dataid:MediaType
subclassOf: dct:MediaType
dataid:typeName : rdfs:Literal
dataid:typeTemplate : rdfs:Literal
dataid:typeReference : rdfs:Resource
dataid:typeExtension : rdfs:Literal

Figure 5.8: The MediaType concept.

The following extract exemplifies the use of these properties: [84]

```
<http://dataid.dbpedia.org/ns/mt#MediaType_turtle_x-bzip2>
        a                      dataid:MediaType ;
        dataid:innerMediaType  <http://dataid.dbpedia.org/ns/mt#MediaType_turtle> ;
        dataid:typeName        "Turtle Bzip2" ;
        dataid:typeExtension   ".bz2" ;
        dataid:typeTemplate    "application/x-bzip2"

<http://dataid.dbpedia.org/ns/mt#MediaType_turtle>
        a                     dataid:MediaType ;
        dataid:typeName       "Turtle" ;
        dataid:typeExtension  ".ttl" ;
        dataid:typeTemplate   "application/x-turtle" ;
        dataid:typeTemplate   "text/turtle" .
```

Listing 5.6: Example of a complex MediaType

[84] note: the namespace `http://dataid.dbpedia.org/ns/mt#` for common MediaTypes is used on a preliminary basis)

5.2.5 Agent

An *Agent* is something or someone that bears some form of responsibility for an *Entity* or activities which create, transform or manage *Entities* in some way. *Agents* are real or legal persons, groups of persons, programs, organisations, etc. The class `dataid:Agent` subsumes both agent concepts of the `PROV-O` and `FOAF` ontologies, to further incorporate `PROV-O` into the context of `DCAT` (which uses the class `foaf:Agent` to portray this concept).

The attributes of the `FOAF` vocabulary[85] are used to describe aspects such as name and e-mail address of a person. In addition, the following properties were introduced in `DataID core`:

`dataid:hasAuthorization`: the inverse property of `dataid:authorizedAgent`, pointing out an *Authorization* which grants an *Agent* some kind of authority over *Entities*. This is explained in detail in the example on *Authorizations* in the next Section 5.2.6. **(R)**

`dataid:identifier`: often, *Agents* already have unique identifiers specified in a different context on the Web (in addition to the URI used in the context of a `DataID` document, such as URI, ORCID[86] or researcherId[87]). This property points out additional identifiers (cf. Section 5.2.9). **(O)**

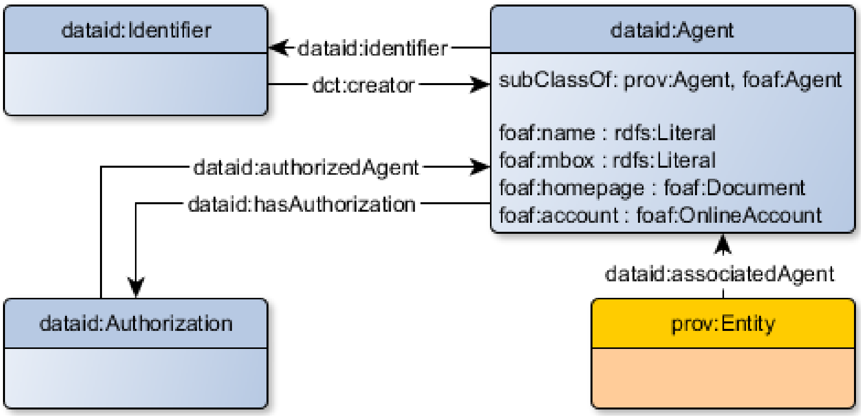

Figure 5.9: The Agent concept

[85] `http://xmlns.com/foaf/spec/`
[86] `http://orcid.org`
[87] `http://www.researcherid.com`

This example of an *Agent* portrays the `DBpedia` Association[88]:

```
<http://wiki.dbpedia.org/dbpedia-association>
        a                          dataid:Agent ;
        dataid:hasAuthorization    <dataid_ar.ttl?auth=creatorAuthorization> ;
        foaf:homepage              <http://dbpedia.org> ;
        foaf:mbox                  "dbpedia@infai.org" ;
        foaf:name                  "DBpedia Association" .
```

Listing 5.7: Example of an Agent

5.2.6 Authorization

One objective of `DataID core` is the detailed expression of the relations between *Agents* and *Entities*. To qualify these relations (summarised under the property `dataid:associatedAgent`) *AgentRoles* have to be assigned to the involved *Agents* (such as *Maintainer*, *Publisher*, etc.). This is achieved by the class `dataid:Authorization`, which is a sub-class of `prov:Attribution`, a qualification of the property `prov:wasAttributedTo`. It basically states, which *AgentRoles* (pointed out with `dataid:authorityAgentRole`) an *Agent* (via `dataid:authorizedAgent`) has, regarding a certain collection of *Entities* (`dataid:authorizedFor`). This mediator is further qualified by an optional period for which it is valid and access restrictions by the *Entities* themselves, allowing only specific *Authorizations* to exert influence over them (cf. `dataid:needsSpecialAuthorization`).

Section 5.3 contains a detailed example on *Authorizations*, to deepen the understanding of this concept as well as to provide a suitable use case.

`dataid:authorizedAgent`: each *Authorization* shall have at least one associated *Agent*, which is pointed out via this property (sub-property of `prov:agent`). **(M)**

`dataid:authorityAgentRole`: provides the *AgentRole* the *Agent*(s) of this *Authorization* are assigned with (sub-property of `prov:hadRole`). **(M)**

`dataid:authorizedFor`: points out those *Entities* for which this *Authorization* is valid (sub-property of `dataid:authorizationScope`). **(M)**

`dataid:authorizedAction`: an *AgentRole* entails the right (or responsibility) for *Agents* to execute a predefined collection *AuthorizedAction*(s). This property can be inferred by a chain of properties (OWL property chain axiom): `dataid:authorityAgentRole` / `dataid:allowsFor`. **(O)**

[88] `http://wiki.dbpedia.org/dbpedia-association`

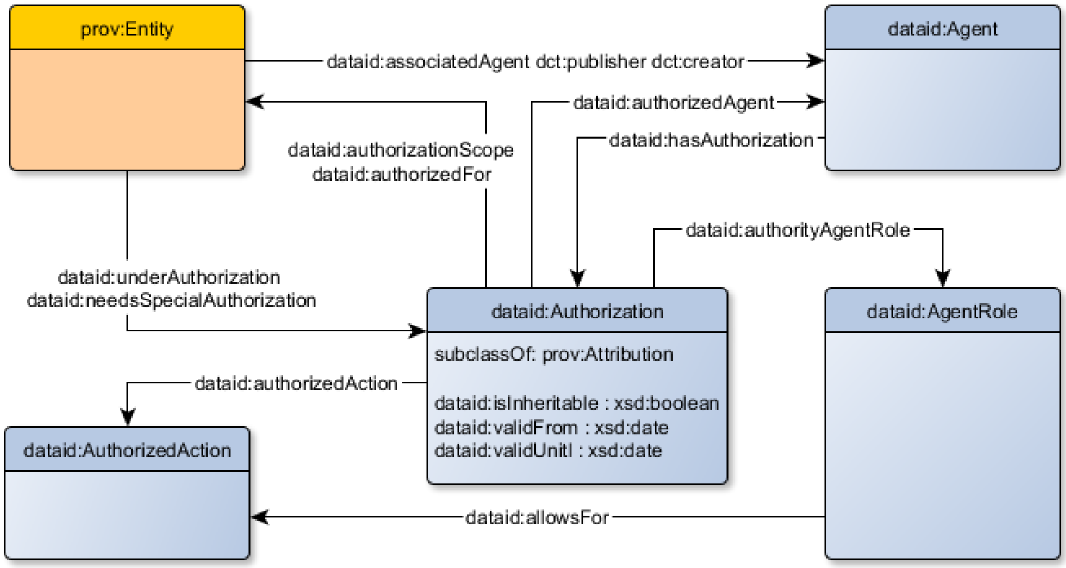
prov:Entity
dataid:Agent
dataid:associatedAgent dct:publisher dct:creator
dataid:authorizedAgent
dataid:hasAuthorization
dataid:authorizationScope
dataid:authorizedFor
dataid:authorityAgentRole
dataid:underAuthorization
dataid:needsSpecialAuthorization
dataid:Authorization
subclassOf: prov:Attribution
dataid:isInheritable : xsd:boolean
dataid:validFrom : xsd:date
dataid:validUnitl : xsd:date
dataid:AgentRole
dataid:authorizedAction
dataid:AuthorizedAction
dataid:allowsFor

Figure 5.10: The Authorization concept

`dataid:isInheritable:` indicates whether an *Authorization* is transferable when changing versions of a `DataID`. Thus, keeping *Agent*, *AgentRole* and *Actions* in place for the updated versions of all involved *Entities*. **(O)**

`dataid:validFrom:` defining the temporal beginning (inclusive) of an *Authorization* (the time from which on out the axioms of an *Authorization* are valid). **(O)**

`dataid:validUntil:` defining the temporal ending (exclusive) of an *Authorization* (the time from which on out the axioms of an *Authorization* are no longer valid). **(O)**

The property `dataid:authorizationScope` is an abstract super-property of `dataid:authorizedFor`, pointing to all referred and inferred *Entities* which are under a certain *Authorization*. Triples with this predicate are inferred by its sub-properties `dataid:authorizedFor` and the virtual properties `dataid:authorizationChain1` to `dataid:authorizationChain9`.

Furthermore, property `dataid:underAuthorization` is the inverse of `dataid:authorizationScope` and a sub-property of `prov:wasAttributedTo`. It points out an *Authorization* which qualifies the relation `dataid:associatedAgent` of an *Entity*. Its sub-property `dataid:needsSpecialAuthorization` was introduced to restrict the reach of *Authorizations*, to the exclusion of those *Authorizations*, not referenced via this property (or simply: if an *Entity* has a `dataid:needsSpecialAuthorization` instance, all *Authorizations* without this referral have no influence over this ***Entity***, disregarding its specification). An exact explanation of ***Authorizations*** and all involved properties is accompanying the extended example on ***Authorizations*** at the end of this chapter (cf. Section 5.3).

The following snippet provides a simple example of two ***AgentRoles*** being assigned to two ***Agents*** for a ***DataId*** instance (and thereby for every ***Entity*** involved in this `DataID`- see Section 5.3 for more).

```
<http://wiki.dbpedia.org/dbpedia-association/persons/Freudenberg>
        a                        dataid:Agent ;
        dataid:hasAuthorization  <dataid_ar.ttl?auth=maintainerAuthorization> ;
        foaf:mbox                "freudenberg@informatik.uni-leipzig.de" ;
        foaf:name                "Markus Freudenberg"                    ;
        dataid:identifier        <http://www.researcherid.com/rid/L-2180-2016> .

<dataid_ar.ttl?auth=maintainerAuthorization>
        a                          dataid:Authorization ;
        dataid:authorityAgentRole  dataid:Maintainer ;
        dataid:authorizedAgent     <http://wiki.dbpedia.org/dbpedia-association/persons/Freudenberg>
             ;
        dataid:authorizedFor       <dataid_ar.ttl> .

<http://wiki.dbpedia.org/dbpedia-association>
        a                        dataid:Agent ;
        dataid:hasAuthorization  <dataid_ar.ttl?auth=creatorAuthorization> ;
        foaf:homepage            <http://dbpedia.org> ;
        foaf:mbox                "dbpedia@infai.org" ;
        foaf:name                "DBpedia Association" .

<dataid_ar.ttl?auth=creatorAuthorization>
        a                          dataid:Authorization ;
        dataid:authorityAgentRole  dataid:Creator ;
        dataid:authorizedAgent     <http://wiki.dbpedia.org/dbpedia-association> ;
        dataid:authorizedFor       <dataid_ar.ttl> .
```

Listing 5.8: Example of two Agents and their pertaining Authorizations

5.2.7 AuthorizedAction & AgentRole

The ***AgentRole*** assigned to an ***Agent*** in the context of an `dataid:Authorization` is defined only by the property `dataid:allowsFor`, pointing out the ***AuthorizedActions*** it entails. A `dataid:AuthorizedAction` shall either be a `dataid:EntitledAction`, representing all ***AuthorizedActions*** an ***Agent*** could take, or the ***AuthorizedActions*** an ***Agent*** has to take (`dataid:ResponsibleAction`).

AuthorizedActions and ***AgentRoles*** defined in this ontology are only examples of possible implementations, reflecting a common environment of a File or Do-

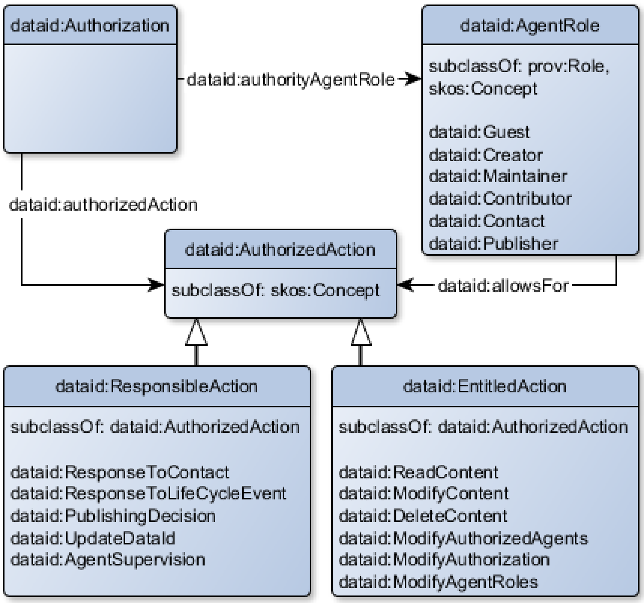

Figure 5.11: The AuthorizedAction and AgentRole concepts

cument Management System. They can be replaced to fit the use case at hand. Implementing them as a `skos:ConceptScheme`[89] offers additional semantics, for example in determining which ***AgentRole*** can override ***AuthorizedActions*** initiated by ***Agents*** with other ***AgentRoles*** for the same ***Entity***. The following descriptions will shortly introduce each individual of these classes, predefined in the `DataID core` ontology.

AgentRoles:

`dataid:Creator:` Creator of the resource. An ***AgentRole*** that is credited with the main part in the initial creation of the resource.

`dataid:Contact:` An ***Agent*** that can be contacted for general requests about the resource.

`dataid:Contributor:` Contributor to the resource. An ***Agent*** that was involved in creating or maintaining the resource but did not have the main part in this activity.

`dataid:Guest:` A visitor or anonymous ***Agent*** which only has the right to read public documents.

`dataid:Maintainer:` Maintainer of the ***Dataset***. An ***Agent*** that ensures the technical correctness, accessibility and up-to-dateness of a ***Dataset***.

[89] `https://www.w3.org/TR/skos-reference/#ConceptScheme`

dataid:Publisher: An ***Agent*** that makes the ***Dataset*** accessible online on a server or repository without necessarily being involved in its creation and decides on all dissemination related tasks as well (e.g. data portal entries).

ResponsibleActions:

dataid:ResponseToContact: The responsibility to respond to contact attempts by external ***Agents***. A contact point for the ***Entity***.

dataid:ResponseToLifeCycleEvent: The responsibility to manage changes and react to any event related to the lifecycle of a ***Dataset*** (e.g. issue tracker entries, unavailable download URL, etc.).

dataid:PublishingDecision: The final decision if an ***Entity*** (or a version of it) should be published.

dataid:UpdateDataId: The responsibility to update dataset metadata.

dataid:AgentSupervision: The responsibility to supervise other ***Agents***.

EntitledActions:

dataid:ReadDataId: read the `DataID` dataset metadata.

dataid:ReadContent: read the content of an ***Entity***.

dataid:ModifyContent: modify the content of an ***Entity***.

dataid:DeleteContent: delete some content of an ***Entity***.

dataid:ModifyAuthorization: modify an ***Authorization***.

dataid:ModifyAgentRoles: modify ***AgentRoles*** and ***AuthorizedActions***.

This example has been lent from the `DataID core` ontology document itself. Here the ***AgentRole*** 'Contact' is defined in the context of a `skos:ConceptScheme`.

```
dataid:Contact
        a owl:NamedIndividual, dataid:AgentRole;
        rdfs:isDefinedBy <http://dataid.dbpedia.org/ns/core#> ;
        dataid:allowsFor dataid:ReadDataId;
        dataid:allowsFor dataid:ModifyContent;
        dataid:allowsFor dataid:ReadContent;
        dataid:allowsFor dataid:ResponseToContact;
        rdfs:comment "An agent that can be contacted for general requests about the resource."@en ;
        skos:prefLabel "contact"@en ;
        skos:inScheme dataid:AgentRoleScheme ;
        skos:broader dataid:Publisher, dataid:Maintainer .
```

Listing 5.9: Example of an AgentRole definition

5.2.8 DatasetRelationship

A *DatasetRelationship* is a qualification of the generic property `dataid:relatedDataset` (which is a sub-property of `dct:relation`). The `dataid:DatasetRelationship` is a subclass of `prov:EntityInfluence` and is defined by three properties:

`dataid:datasetRelationRole`: specifying the role (or type) of this relationship, defining the exact role the 'target' *Dataset* takes regarding the 'origin' *Dataset* of this relationship. **(M)**

`dataid:qualifiedRelationOf`: the inverse property of `dataid:qualifiedDatasetRelation` is pointing out the origin *Dataset* of this qualification. **(M)**

`dataid:qualifiedRelationTo`: the target *Dataset* of this qualification. **(M)**

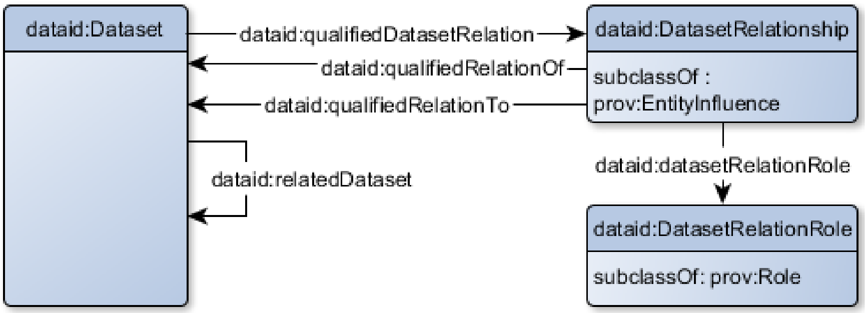
dataid:Dataset
dataid:qualifiedDatasetRelation
dataid:qualifiedRelationOf
dataid:qualifiedRelationTo
dataid:relatedDataset
dataid:DatasetRelationship
subclassOf :
prov:EntityInfluence
dataid:datasetRelationRole
dataid:DatasetRelationRole
subclassOf: prov:Role

Figure 5.12: The DatasetRelationship and DatasetRelationRole concept

The class `dataid:DatasetRelationRole` is not further qualified in the context of `DataID core`, which could be done in an extension to `DataID core`, similar to `dataid:AgentRole`. Some instances of this class are already provided:

`dataid:GenericRelation`: specifies a `dataid:DatasetRelationship` between two *Datasets* which have a relation to an unknown quality (such a relation is equivalent to direct `dct:relation` property).

`dataid:DerivateRole`: specifies a `dataid:DatasetRelationship` where one *Dataset* points out a second *Dataset*, which is a derivate of the first.

`dataid:SourceRole`: specifies a `dataid:DatasetRelationship` where the origin *Dataset* is created by transforming/collecting data from the target *Dataset*.

`dataid:CopyRole`: specifies a `dataid:DatasetRelationship` where the origin *Dataset* is an exact copy of the target *Dataset* (e.g. when republished under a different domain).

dataid:SimilarityRole: specifies a dataid:DatasetRelationship where the origin ***Dataset*** has a significant similarity to the target ***Dataset*** (without any assertion as to a dimension of similarity).

In the example, the Wikipedia source Dataset (named 'pages_articles') of a given DBpedia dataset is referred to with the help of this concept:

```
<dataid_ar.ttl?relation=source&target=pages_articles>
        a                          dataid:DatasetRelationship ;
        dataid:datasetRelationRole  dataid:SourceRole ;
        dataid:qualifiedRelationOf  <dataid_ar.ttl?set=long_abstracts_en_uris> ;
        dataid:qualifiedRelationTo  <dataid_ar.ttl?set=pages_articles> .
```

Listing 5.10: Example of a DatasetRelationship

5.2.9 Identifier

The class dataid:Identifier uniquely identifies any resource (incl. ***Entities*** and ***Agents***), given an identifier as a literal and a corresponding datacite:IdentifierScheme (e.g. ORCID, ResearcherID etc.). Typically an organisation is responsible for issuing and managing Identifiers described with this concept, which can be referred to with dct:creator. DataID core adopted this approach from Datacite ontology (cf. Section 3.2.5) to provide a schematic way of adding additional, existing identifiers to ***Entities*** and ***Agents*** referenced in a DataID document.

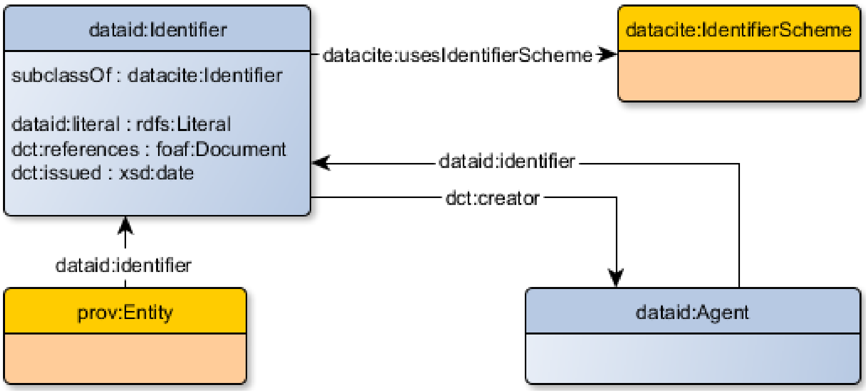
dataid:Identifier
subclassOf : datacite:Identifier
dataid:literal : rdfs:Literal
dct:references : foaf:Document
dct:issued : xsd:date
datacite:usesIdentifierScheme
datacite:IdentifierScheme
dataid:identifier
dct:creator
dataid:identifier
prov:Entity
dataid:Agent

Figure 5.13: The Identifier concept

`dataid:literal`: the identifier as literal (e.g. a URI as literal). **(M)**

`datacite:usesIdentifierScheme`: an IdentifierScheme defines (among other attributes) a pattern against which the literal of an identifier is validated. Thereby the validity of an identifier is tested. **(M)**

`dct:references`: often identifier agencies have web presentations for their identifiers (e.g. ORCID[90]). Such a website can be referenced with this property. **(O)**

`dct:creator`: can be used to identify the identifier agency, responsible for an identifier and pertaining scheme. **(O)**

```
<http://www.researcherid.com/rid/L-2180-2016>
        a                           dataid:Identifier ;
        dataid:literal              "L-2180-2016" ;
        dct:issued                  "2016-08-01"^^xsd:date ;
        dct:references              <http://www.researcherid.com/rid/L-2180-2016> ;
        datacite:usesIdentifierScheme  datacite:researcherid .
```

Listing 5.11: Example of an Identifier

5.2.10 SimpleStatement

The concept `dataid:SimpleStatement` is intended as a tool for conveying a statement, definition or point of view about a certain topic. Using either a simple literal (`dataid:literal`) to provide a quotation or by a referencing a web resource providing or representing the statement in any medium (picture, text, video, etc.). This class is a sub-class of `prov:Entity` and implements also the following Dublin Core classes: `dct:ProvenanceStatement`, `dct:RightsStatement`, `dct:Standard` . With this measure, it is possible to attach provenance information onto instances of this concept and to use `dataid:SimpleStatement` as the range of `dct:rights` and its sub-properties, `dct:provenance`, `dct:conformsTo` and others.

This reification approach with an intermediate resource was chosen to cover as many scenarios as possible including many edge cases which do not have to be modelled explicitly. To provide a minimum of structure for textual statements, as well as providing a resource onto which provenance information could be attached.

[90] `http://orcid.org/0000-0002-1825-0097`

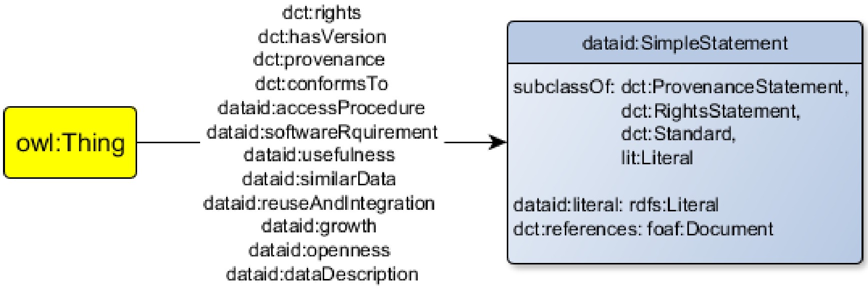

Figure 5.14: The SimpleStatement concept

`dataid:literal`: a textual statement 'from humans for humans'. **(R)**

`dct:references`: the alternative reference to a web resource containing the statement comprehensible by humans. **(O)**

Two instances from the running example demonstrating the different usage scenarios of this concept. The first is the official rights statement of the `DBpedia` Association, while the second is the access procedure for the `DBpedia` SPARQL endpoint, where `dct:reference` is pointing out the SPARQL 1.1 specification (as a web page specifying the means of querying the endpoint).

```
<dataid_ar.ttl?rights=dbpedia-rights>
        a                 dataid:SimpleStatement ;
        dataid:literal  """DBpedia is derived from Wikipedia and is distributed under the same
             licensing terms as Wikipedia itself. As Wikipedia has moved to dual-licensing, we also
             dual-license DBpedia starting with release 3.4. Data comprising DBpedia release 3.4 and
             subsequent releases is licensed under the terms of the Creative Commons Attribution-
             ShareAlike 3.0 license and the GNU Free Documentation License. Data comprising DBpedia
             releases up to and including release 3.3 is licensed only under the terms of the GNU Free
              Documentation License."""@en .

<dataid_ar.ttl?stmt=sparqlaccproc>
        a                 dataid:SimpleStatement ;
        dct:references    <https://www.w3.org/TR/sparql11-overview/> ;
        dataid:literal  "An endpoint for sparql queries: provide valid queries."@en .
```

Listing 5.12: Examples of SimpleStatements

5.3 Complex Example on Authorizations

I decided to provide a more prolific example on the subject of ***Authorizations*** since this concept is of more complex nature. In particular, the impact of `dataid:authorizationScope` with its sub-properties is difficult to understand at first sight.

The property `dataid:authorizationScope` has the role of an abstract super property, pointing out all referred and inferred ***Entities*** under a given ***Authorization*** and is usually not instantiated in a `DataID` graph. It can be inferred directly by the existence of property `dataid:authorizedFor`, which is used to reference ***Entities*** to which an ***Authorization*** applies and its rules and restrictions are tailored for.

The following axioms for the transitive property `dataid:authorizationScope` would be desirable to extend the influence of an ***Authorization*** along any property path combined of `foaf:primaryTopic`, `void:subset` and `dcat:distribution`, initiated by an instance of `dataid:authorizedFor`.

$$
\begin{aligned}
foaf : primaryTopic &\sqsubseteq dataid : authorizationScope\\
void : subset &\sqsubseteq dataid : authorizationScope\\
dcat : distribution &\sqsubseteq dataid : authorizationScope\\
dataid : authorizedFor &\sqsubseteq dataid : authorizationScope
\end{aligned}
$$

To not hijack foreign ontologies [24] (i.e. `DCAT`, `VoID` or `FOAF`), a series of properties (`dataid:authorisationChain1` - `dataid:authorisationChain9`) were introduced to simulate this behaviour with the help of the OWL property chain axiom[91].

Properties `dataid:authorisationChain1` to `dataid:authorisationChain6`:

$$
\begin{aligned}
foaf : primaryTopic \circ dcat : distribution &\sqsubseteq dataid : authorizationScope\\
foaf : primaryTopic \circ void : subset &\sqsubseteq dataid : authorizationScope\\
void : subset \circ dcat : distribution &\sqsubseteq dataid : authorizationScope\\
void : subset \circ void : subset &\sqsubseteq dataid : authorizationScope\\
void : subset \circ void : subset \circ dcat : distribution &\sqsubseteq dataid : authorizationScope\\
foaf : primaryTopic \circ void : subset \circ dcat : distribution &\sqsubseteq dataid : authorizationScope
\end{aligned}
$$

With these properties, all subsequent ***Entities*** connected to the origin ***Entity*** (the ***Entity*** referenced from an ***Authorization*** with `dataid:authorizedFor`), are under the influence of this ***Authorization***, connected with it via the transitive property `dataid:authorizationScope`. One exception remains: those ***Entities***

[91] https://www.w3.org/TR/owl-primer/#Property_Chains

second in line after the origin ***Entity*** are skipped in this progression. Properties `dataid:authorisationChain7` to `dataid:authorisationChain9` are solving this issue:

$$dataid : authorizedFor \circ foaf : primaryTopic \sqsubseteq dataid : authorizationScope$$
$$dataid : authorizedFor \circ void : subset \sqsubseteq dataid : authorizationScope$$
$$dataid : authorizedFor \circ dcat : distribution \sqsubseteq dataid : authorizationScope$$

For example: if a knowledge base (KB) holds some statements, such as:

```
ex:someAuthorization    dataid:authorizedFor         ex:DataId .
ex:DataId               foaf:primaryTopic            ex:RootDataset .
ex:RootDataset          void:subset                  ex:DatasetC .
```

we can infer the following statements:

```
ex:someAuthorization    dataid:authorizationScope    ex:DataId .
ex:someAuthorization    dataid:authorizationScope    ex:RootDataset .
ex:someAuthorization    dataid:authorizationScope    ex:DatasetC .
```

by inferring these statements first:

```
ex:DataId               dataid:authorizationChain2   ex:DatasetC .
ex:someAuthorization    dataid:authorizationChain7   ex:RootDataset .
```

With these virtual properties the inference of `dataid:authorizationScope` along the depicted property paths becomes feasible:

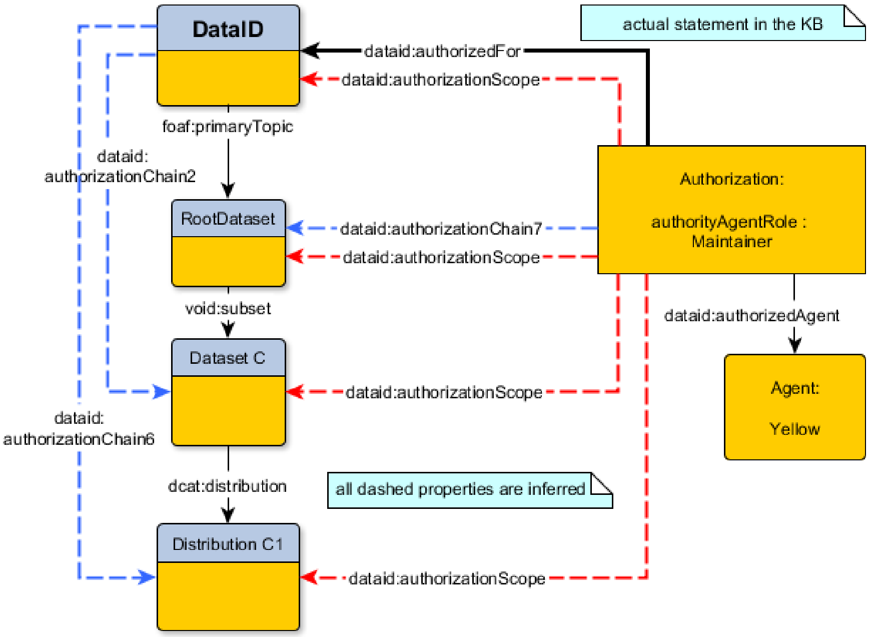

Figure 5.15: Inferences of `dataid:authorizationScope`

Here the ***Authorization*** for ***Agent*** Yellow is not only valid for the ***DataId*** entity, referred to via `dataid:authorizedFor`. By inferring additional statements of this kind, the scope of this ***Authorization*** is extended to every ***Dataset*** and ***Distribution*** connected via `foaf:primaryTopic`, `dcat:distribution` and `void:subset`. By this means, extending the influence (or scope) of an ***Authorization*** over multiple ***Entities***, without having to point out all of them with `dataid:authorizedFor`, is realised. External properties involved did not have to be distorted, by adding sub-property axioms or an inclusion of rule-based axioms (such as SWRL[92]), avoiding unforeseeable complications in use case extensions, importing `DataID core`.

The automatic extension of an ***Authorization*** has also its drawbacks. By introducing multiple ***Authorizations*** in the context of a `DataID` document, providing the same ***AgentRole*** for an ***Entity***, the author can encounter unintended behaviours. In this example the previous context is enriched by introducing an additional ***Agent*** Blue with ***AgentRole Maintainer***:

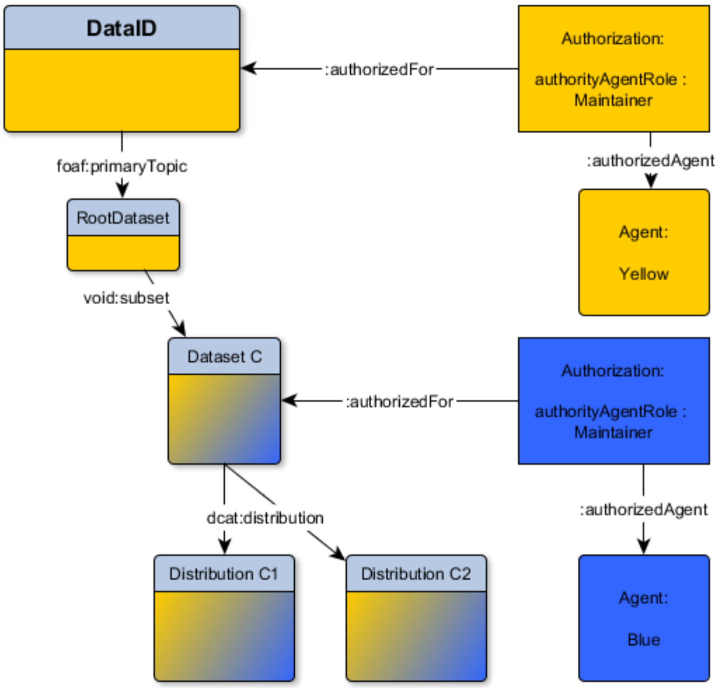

Figure 5.16: Two Agents sharing responsibilities of a Maintainer

Dataset C (and all its ***Distributions***) has two ***Maintainers***, both equally permitted to wield ***AuthorizedActions*** as defined by the definition of `dataid:Maintainer`. This behaviour may or may not be intended by the author.

[92] `https://www.w3.org/Submission/SWRL/`

To provide the means for restricting ***Entities*** to specific ***Authorizations***, the property `dataid:needsSpecialAuthorization` was introduced. This sub-property of `dataid:underAuthorization` (the inverse of `dataid:authorizationScope`) allows to point out those ***Authorizations*** with sufficient importance to exert their authority over an ***Entity***, to the exclusion of other ***Authorizations*** referenced via `dataid:authorizationScope`.

The following example again expands the already known scenario, by introducing a third ***Authorization*** for ***Agent*** Green:

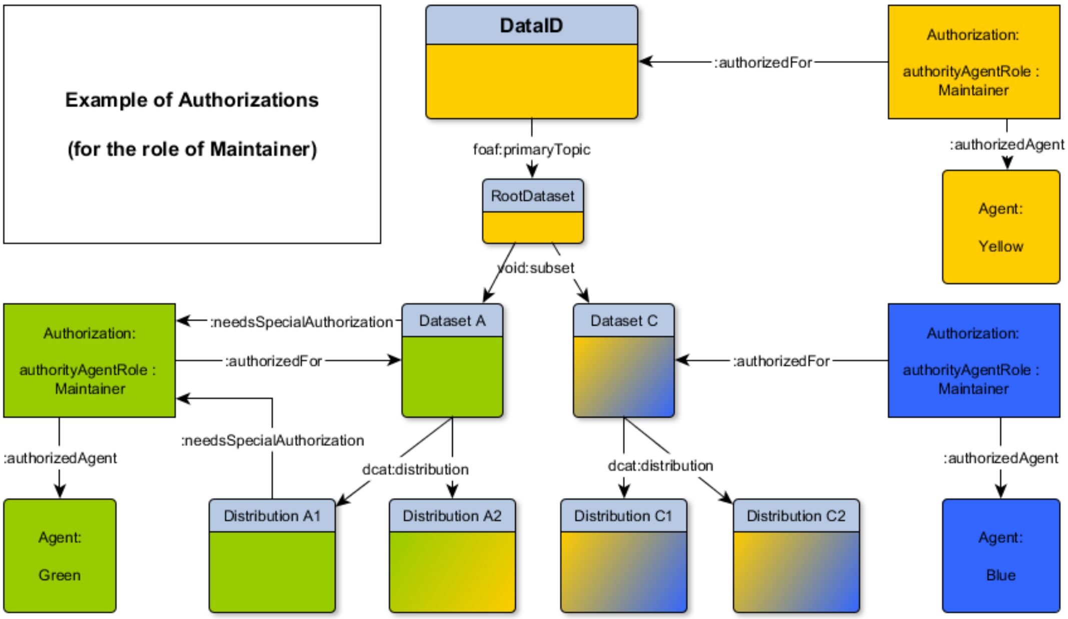

Figure 5.17: Restricting the influence of Authorizations

While *Dataset A* and *Distributions A1* and *A2* are under the ***Authorizations*** of ***Agent*** Yellow and ***Agent*** Green, only *Distribution A2* will be maintained by both ***Agents***. *Dataset A* and *Distribution A1* require specifically the ***Authorization*** of ***Agent*** Green for the purpose of providing the ***AgentRole*** of ***Maintainer***.

This mechanism is useful when introducing different levels of privacy into the domain, for example, a Document Management System (DMS). Two groups of users are specified: The first group (yellow group) should only be able to read the content of a given collection of documents, while the second group (blue group) is also allowed to modify these documents. Therefore, defining two new ***AgentRoles*** is advisable. ***AgentRole*** 'Reader' can only read the content of ***Entities*** available to it, while the 'Editor' also allows for modifying the content. These ***AgentRoles*** are linked to via `dataid:authorityAgentRole` from the respective ***Authorizations*** of the two groups (`dataid:authorizedAgent` points out the members of a group).

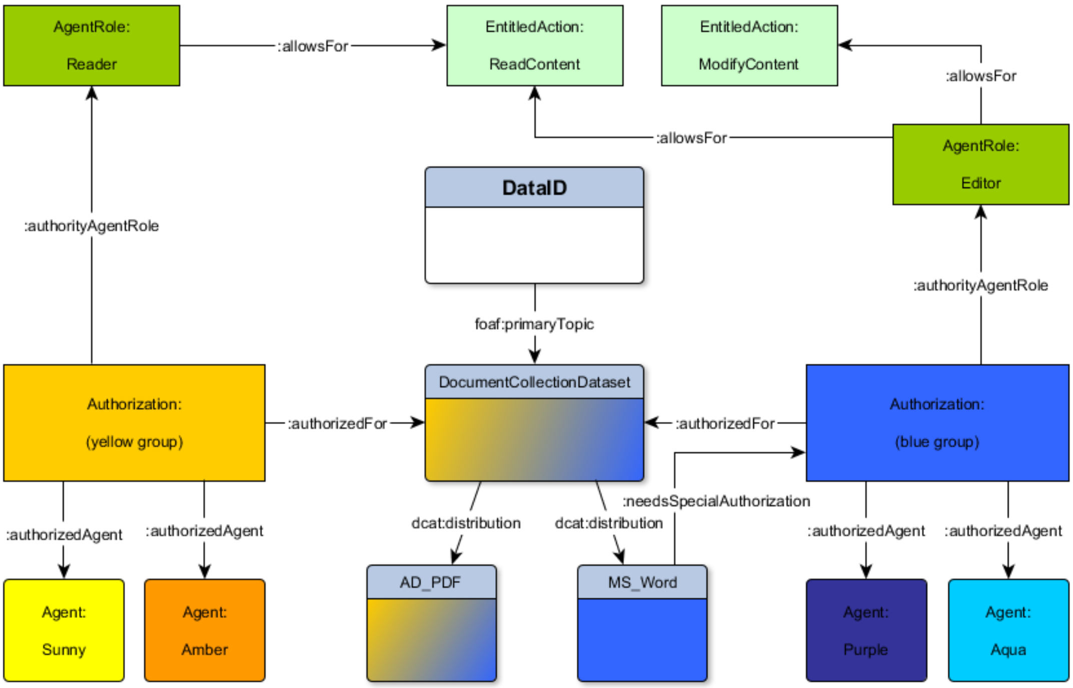

Figure 5.18: Example: Document Management System

Both ***Authorizations*** are authorised for the same document collection (***Dataset***) and its ***Distributions*** as PDF and MS_Word versions of the same content in the DMS. Since the MS_Word version of the documents is used for editing the content, while its PDF counterpart is the publishing version, it is sensible to allow only the Editors (blue group) access to the MS_Word ***Distribution*** by using `dataid:needsSpecialAuthorization`.

6 Publishing Datasets with DataID

6.1 Best Practices

Best practices on any kind of methodology or problem have become a ubiquitous presence in the current landscape of the World Wide Web. Be it a Ted Talk on how to live to be 100[93] or how to get accurate results when using machine learning techniques[94].

Astonishingly, when it comes to publishing data in a widely accepted manner, only a hand full of comprehensive best practices exist. Most of these best practices, workflows or checklists are further constricted to a certain field of research, methodology or data type:

Methodological Guidelines for Publishing Government Linked Data "[...] a preliminary set of methodological guidelines for generating, publishing and exploiting Linked Government Data" [74]

Best Practices for Publishing Linked Data "[...] a series of best practices designed to facilitate development and delivery of open government data as Linked Open Data." [36]

Key components of data publishing "From an assessment of the current data-publishing landscape, we highlight important gaps and challenges to consider, especially when dealing with more complex workflows and their integration into wider community frameworks." [5]

There are two not quiet distinct categories of best practices on data publishing: workflows and checklists.

Workflows introduce a specific order of activities, the flow of artefacts between them as well as agents and their roles in these processes. Workflows are often summarised in so called Data Lifecycles (or Data Engineering Lifecycle).

[93] https://www.ted.com/talks/dan_buettner_how_to_live_to_be_100

[94] http://machinelearningmastery.com/machine-learning-checklist/

The LOD2 Lifeycle of Linked Data [4] - used by the ALIGNED project (cf. Figure 1.1) - is a portrayal of such a workflow in a domain-specific environment.

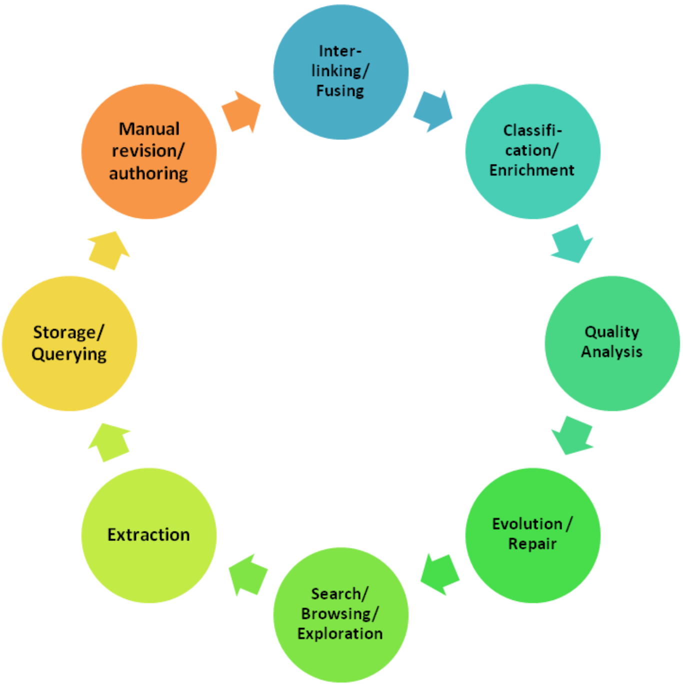
Inter-linking/ Fusing
Classifi-cation/ Enrichment
Quality Analysis
Evolution / Repair
Search/ Browsing/ Exploration
Extraction
Storage/ Querying
Manual revision/ authoring

Figure 6.1: The LOD2 Lifecycle of Linked Data [4]

Many depictions of Data Engineering Lifecycles have domain-independent similarities. Villazón-Terrazas et al. presented a more generic version of such a cycle, which I am going to adopt in the context of this work.

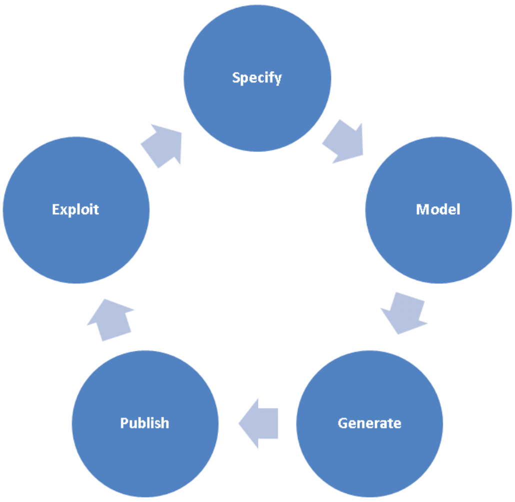

Figure 6.2: The Government Linked Data Lifecycle [74]

A concise summary, tracing the bubble graph above:

- Specify
  - analysis of data sources (what data is useful) or previous version of the dataset
  - Select/Design an identifier scheme used to identify all instances
  - decide on data type and serialisation of you data
- Model
  - search/create/update a suitable schema/vocabulary on which to base the datasets
- Generate
  - transform the data sources
  - clean (and validate) the result
- Publish
  - improve discoverability (choose suitable data portal, announce release at different venues)
  - publish the created datasets
  - publish the pertaining metadata
- Exploit
  - different consumer side activities (browsing, integrating, analysing etc.)
  - collect feedback from consumers, improve data

Checklists are collections of general advises, how-tos and precautions on publishing data without a particular order of items.

The most comprehensive collection of best practices about publishing data has emerged with the establishment of the 'Data on the Web' W3C working group[95] and their best practices document [48] (released as recommendation candidate by the time of writing).

This set of recommendations touches upon most of the crucial issues when publishing data (with the noted exception of dissemination tasks). I endorse these

[95] `https://www.w3.org/2013/dwbp/wiki/Main_Page`

35 best practices to their full extent and recommend to follow all of the suggestions made. Datasets conform to these suggestions, would reap all benefits towards which these practices were created: *Reuse, Comprehention, Linkability, Discoverabilty, Trust, Access, Interoperability and Processability* [48].

A different selection of best practices was already discussed in Section 2.1.5. The FAIR principles [76] are ensuring the quality of data concerning machine-readability, discovery and reuse.

In this chapter, I want to build on these foundations, by expanding on metadata composition and deployment with `DataID` (Section 6.2). In addition, I want to contribute to the discussion on publishing data. A checklist for publishing Linked Data datasets is presented in Section 6.3, based on the experiences accumulated from publishing official `DBpedia` releases.

6.2 Composing and Publishing DataID based Metadata

The data engineering lifecycle depicted in Figure 6.2 does not only apply to the data being generated and published. It is also a valid depiction of a lifecycle for metadata which is co-evolving parallel to the data it is portraying.

In the case of `DataID` metadata, the tasks necessary to eventually publish a metadata document are outlined in this best practice. I will apply the five stages of the data engineering lifecycle (cf. Figure 6.2) to the process of creating `DataID` metadata.

Specify use case requirements for metadata Based on the analysis of data sources, external requirements and advise collected in the run up to the effort of creating data, a set of requirements for dataset metadata has to be outlined as well. This is a highly use-case-specific process. Some fundamental question which might guide this process are listed below:

- Who is the intended audience of the data (what kind of data and metadata is needed)?
- What type of data will be published (Linked Data, tables, etc.)?
- How was data published by my organisation/partners until now?
- How will the data be consumed and who will consume it (machines vs. humans)?

- What are the legal terms under which the data is published?
- What kind of provenance information needs to be conveyed?
- Does the dataset have different versions?
- What standards or vocabularies were used by the source datasets?

Model a DataID ontology and application profile A crucial step when implementing a DataID based metadata solution is the decision on which DataID extension ontologies to use.

Deciding on which combination of DataID ontologies to use for a Dataset description is a domain and problem dependent process. For example, a DataID based ontology for Linked (Open) Data datasets dealing with multi-dimensional data may look schematically like this: (importing DataID core and the extensions for Linked Data and Statistics, as well as some additional properties only used in this use case.)

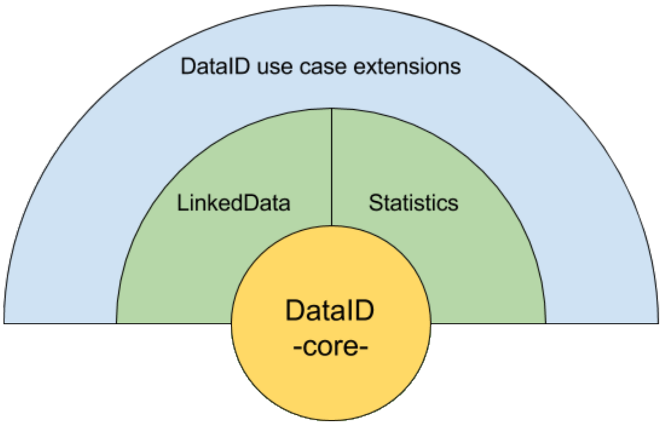

Figure 6.3: Example of combining multiple DataID ontologies

It may be necessary to add additional properties combined in a use case extension to satisfy all requirements. This process will be the focus of the Data Management Plan use case presented in Chapter 7. In general, the usual recommendations for ontology engineering apply, when creating a use case extension for DataID.

- Follow an established methodology when creating an ontology
- Reuse well-established ontologies where possible (such as W3C recommended ontologies)

- Create new classes based on more general concepts to harness additional functionality and increase interoperability (e.g. `prov:Entity`)
- Make use of established tooling chains for modelling ontologies (such as Protégé[96])
- Provide sufficient documentation to ease understanding and reuse

In addition, I recommend the following:

- Keep the import portfolio as slim as possible - only import extensions or other ontologies which are really needed for the use case at hand to avoid semantic overlap.
- Separate ontology from application profile - don't use the extension ontology for specifying use case specific restrictions, enter those into a second document. I recommend using the Shapes Constraint Language (SHACL) [68] from the RDF Data Shapes W3C Working Group[97] which is available as a working draft at the time of writing. This measure will aid the effort for Interoperability and Extensibility.
- Specify mappings to other metadata formats at this stage to detect missing requirements. Reasons for this step might be already existing metadata files, or other metadata formats are needed for specific tasks such as a `CKAN` profile (Section 3.1.3) for publishing datasets on a data portal.

Generate DataID documents Creation of metadata documents is usually a task executed after the creation and before the publication of datasets. Nevertheless, there are multiple reasons not to wait until this stage to gather the necessary data:

- Describing source datasets is a task best done at a time before the extraction process, after source datasets have been studied.
- The dataset creation process might take a long time, depending on the use case. Metadata for temporary result datasets can be useful for further extraction/transformation steps or to keep track of activities and agents involved (which is useful provenance metadata).
- Quality measures and procedures can be accomplished with the toolchain dependent on a dataset and its metadata. Running such tasks in between extraction/transformation steps of data (and not only after the generation process is finished) is necessary to detect quality gaps early on.

[96] `http://protege.stanford.edu`

[97] `https://www.w3.org/2014/data-shapes/wiki/Main_Page`

When publishing a series of similar or versioned datasets, the automatic creation of DataID documents becomes unavoidable. Integrating the creation of metadata as part of an automated workflow environment or ETL Framework (e.g. Unified Views [42]) would be an obvious means of solving this issue.

As for the data generated, DataID metadata needs validation to make sure of its correctness. While a syntactical validation of an RDF document is a necessary step, it is not sufficient. Tools for a semantic validation of an RDF graphs, against the ontology specified in the previous step, are available (e.g. RDFUnit [43]).

Publish DataIDs alongside the datasets Since dataset metadata is so indispensable for most consumers, the publication of DataID metadata is released parallel to the publishing of datasets. DataID was originally created under the assumption of metadata co-published alongside the datasets (DataID files are in the same folder or at the root of a dataset file hierarchy). This is no longer a requirement since DataID core is expressive enough to find the associated dataset on the Web. However, storing metadata in proximity to its described dataset is good practice and eases the discoverability of both.

Additionally, I propose the following best practice for publishing datasets on a given server:

Similarly to a `robot.txt` file[98], I propose a file to be put in the top-level directory of a server, named `datacatalogs.txt`. Its content is a collection of relative paths, referencing files on the server, containing either an instance of `dcat:Catalog`, pointing out datasets in turn, or instances of `dcat:CatalogRecord` (e.g. DataIDs). The format of these secondary files should be Turtle, which, in my opinion, is the RDF serialisation featuring the best compromise between readability and file size.

In example, next to the `http://dbpedia.org/robots.txt` that explicitly excludes certain directories from being crawled, the `http://dbpedia.org/datacalaog.txt` explicitly states where metadata descriptions of DBpedia datasets can be found. This best practice allows publishers to reference descriptions of all hosted datasets in one place and also enables users to discover and access these datasets easily. It facilitates easy aggregation of an institution's or project's datasets and massively cuts down on time spent on navigating and searching for relevant datasets on diverse websites.

[98] `http://www.robotstxt.org`

Exploitation of DataIDs This stage summarises all activities of dataset consumers which are exploiting the benefits of the pertaining metadata as well. But exploitation is not limited to consumers alone. Data publishers, or pertaining organisations benefit from reliable and machine-readable metadata as well.

For example, the tables of the official download page for `DBpedia` releases are all based on the `DataID` documents accompanying the `DBpedia` datasets[99]. Many other uses for `DataID` documents are conceivable.

6.3 Checklist for Publishing RDF Data

The considerable experience members of the `DBpedia` Association[100] have gathered in regard to creation and dissemination of RDF data is presented in this section, summarising all important steps necessary for releasing such data. While this checklist is focused on Linked Data datasets, many items present could be generalised to fit broader use cases for any type of data. In addition, particular consideration has been given to tasks related to or dependent on dataset metadata (i.e. `DataID`). I will highlight those items.

Documentation When publishing Linked Open Data, a comprehensive documentation of data sources, ontology, versioning and other important context needs to be presented in an easy and accessible manner.

- Decide early which form your documentation will take (online documentation, text files, etc.), depending on size or impact of a dataset.
- When preparing a new version of an existing dataset, list the coming changes to the dataset, as well as a release date.
- Documentation is an iterative and never ending process. Therefore, improve documentation as early as possible to cover every aspect from the outset.
- **`DataID`**: Make sure to collect all references to documentation pages and append them to the pertaining dataset descriptions (e.g. with property `foaf:page`)
- Facilitate means to gather feedback from dataset consumers (mailing list, issue tracker, etc.).

[99] http://wiki.dbpedia.org/downloads-2016-04

[100] http://wiki.dbpedia.org/dbpedia-association

- Record your plans on maintenance and versioning. A well maintained and regularly updated dataset will attract more consumers.
- Ask for support from your data consumers.

Sources Basing the generation of data on specific data sources is key for a coherent dataset release. A comprehensive documentation of the original sources should be one of the first steps in any data release.

- **`DataID`**: Metadata about the source data is highly desirable. Supplying consumers with additional information about the size of a probe, location, conditions, temporal information, dataset creator or publisher will further their understanding and acceptance of the data. It is useful to create a depiction of the source datasets with `DataID core` as well.
- Name the exact version your release is based on. Do not divert from the selected version in the middle of your extraction.
- Provide links to source files or host them yourself. Access to original data is vital for reproducibility. **`DataID`**: use the concept of `dataid:DatasetRelationship` together with the role `dataid:SourceRole` to reflect this in the metadata.
- Refer to external vocabularies & ontologies utilized by your source data. Point to information about the schema of the source data.
- Discuss quirks and unexpected issues you discovered about the source data.

Ontology & Mappings Deciding on an ontology for the chosen domain is deciding on conceptual entities and relationships that will represent your data source. The appropriate level of abstraction is as vital as creating the mapping between both datasets. Note that all types of data extraction processes use mappings. Some mappings are obvious such as R2RML [70], while others not so obvious (i.e. hidden in software code).

- If no existing ontology can represent your data, engineer your own. Try using as many concepts of existing vocabularies as possible. **`DataID`**: use property `void:vocabulary` to refer your ontology.
- Document any schema or mapping components.
- Before starting an extraction enrich and align your schema and mappings.
- During an extraction use a static snapshot of used ontologies and mappings. These versions are the schematic foundation of your upcoming release and are not to be changed in the extraction and publishing process.

- Use a version control system to maintain your ontology and mappings. Especially large ontologies should be maintained by using a version control system like Git [101].

Software Document exactly what kind of software, in what version is used for every step of the pre-processing, extraction and post-processing cycles.

- When using custom software, use a version control system.
- Provide sufficient information about the environment you are working with.
- Provide additional information about deployment steps and configuration if necessary for the extraction process.
- Create software snapshots to enable reproducibility of an extraction process.
- **`DataID`**: define your software in the context of `DataID` as an agent who is responsible for the execution of certain transformation step. If a special software product is needed to process the data produced, state this with `dataid:softwareRequirement`.

Extraction & Dataset Generation While the generation process of an RDF dataset is a publisher-dependent process, there are some significant points to consider.

- Prefer generating resulting datasets in different syntax formats.
- One triple per line is a preferred approach to enable easier procession of the RDF files with non-RDF tools (e.g. command-line directives).
- Group RDF datasets by category or context. Break up large files, grouping triples by property. This way, subsets of properties, which are of no interest, can be left out of future tasks.
- Use a consistent and precise naming strategy when creating files. File names should reflect exactly those triples stored inside (e.g. naming files by property and extraction step).
- **`DataID`**: Store provenance information alongside the files, preferably with your dataset metadata. Additional information about the origin or extraction steps leading to the triple will be useful in following tasks.
- **`DataID`**: Define the exact media type of your resulting files with `dataid:MediaType`.

[101] `http://github.com`

Validation Validating extraction results to confirm their syntactic and semantic correctness is a necessary step, which will, in turn, confirm the correctness of ontology, mappings and the selected extraction process. While validation should also be part of the extraction process itself (e.g. enforcing datatype conformance), most validation is done after the completion of an extraction step. An unsuccessful validation will trigger the cycle of 1) finding the fault, 2) fixing the problem and 3) rerunning the last extraction step(s).

- Try to integrate validation directly in the extraction & generation process to reduce post-validation steps.
- Dataset size is an obvious indication of a release status. Finding obvious errors by comparing the size of a file with the size of previous versions.
- A manual inspection of triples with sampling can identify errors early in the release process.
- Statistical metrics is a good means to provide an overview about quality. Information about both the source and the result data can provide a clear picture on how accurate an extraction performed. If a stark difference of equal types can not be explained by ontology or mappings, a bug in the extraction process is the probable answer.
- Create your own validation tools and use external, general purpose RDF validation tools.
- Use a staging SPARQL endpoint and/or linked data interface for browsing the data.
- Tools, such as RDFUnit [43], perform ontology conformance tests and should be used for a thorough validation of your data.
- **`DataID`**: Any validation result of your data is valid metadata, which can be stored in your `DataIDs`. (e.g. with Data Quality Vocabulary cf. Section 4.3).

Enrichment Dataset enrichment with information from other data sources is a useful step to increase the precision and/or coverage of a knowledge base. However, Enrichment is a step that is closely coupled to a dataset and is therefore no specific advice in this regard.

- RDFS or OWL inferencing using the main or external schemata is definitely a means to add missing knowledge
- Dataset-dependant heuristics can be employed
- Different types of enrichment can be performed before and after the *linking* step but in most case must be done prior to validation.

Linking As a basic principle of Linked Open Data [7], , linking is one of the most important steps in the process of creating RDF datasets. This can be achieved by multiple approaches:

- Performing a manual linking over time. (tends to be a very limited solution)
- Copying existing link sets from other datasets to previous versions of the extracted dataset (which will be outdated).
- Detecting identifiers pertaining to entities of the source or result datasets in different datasets. Generating owl:sameAs triples would complete this method.
- By using the transitive trait of owl:sameAs.
- Provide interlinks between same resources in a dataset (e.g. in different language editions)
- **`DataID`**: Provide instances of concept `dataid-ld:Linkset` to point out the involved datasets (between which links were found) and additional information such as link predicate and number of links found.

Publishing Publishing a dataset is a complex task, which can be done in many ways. Providing raw RDF data alone is not enough to disseminate Linked Open Data.

- Update user documentation (e.g. online data catalogues - i.e. datahub.io).
- Announce the release at different venues.
- Upload your dump files to an accessible location with consistent serialization formats.
- If possible, make the data available through a SPARQL Endpoint and/or a Linked Data interface.
- **`DataID`**: Publish/update metadata documents, to provide a comprehensive dataset description. Providing a metadata such as `DataID` will increase the visibility of your data and automatically create link sets to other datasets.
- **`DataID`**: Store the created `DataID` documents a) alongside your created datasets, as well as in data portals specialised for `DCAT` metadata or convert into other metadata formats if needed.

7 Application: Data Management Plans (DMP)

Over the last years, Data Management Plans (DMP) have become a requirement for project proposals within most major research funding institutions. It states what types of data and metadata are employed, which limitations apply, where responsibilities lie and how the data is stored, both during research project and after the project is completed.

The use case described here will introduce an extension to the DataID core ontology to extensively describe a Data Management Plan for digital data in a universal way, laying the foundation for tools helping researchers and funders with the drafting and implementing of DMPs. Based on multiple requirements, raised from different DMP guidelines, I will showcase the creation of a DataID extension. I incorporated the re3data ontology to describe repositories and institutions, exemplifying the use of external ontologies. This approach will demonstrate the application of the best practices introduced in Section 6.2 for the stages 'Specify' and 'Model' of the lifecycle for dataset metadata. Furthermore, this use case exemplifies the ability of the DataID Ecosystem to solve complex demands on dataset metadata based on its trait of Extensibility.

7.1 Specifying requirements of a DMP

The following requirements were distilled from an extensive list of DMP guidelines of different research funding bodies, covering most of the non-functional demands raised pertaining to digital datasets.

1. Describe how data will be shared (incl. repositories and access procedures).
2. Describe the procedures put in place for the long-term preservation of the data.
3. Describe the types of data and metadata, as well as identifiers used.

4. Provisioning of copyright and license information, including other possible limitations to the reusability of the data.
5. Outline the rights and obligations of all parties as to their roles and responsibilities in the management and retention of research data.
6. Provision for changes in the hierarchy of involved agents and responsibilities (e.g. a Primary Investigator (PI) leaving the project).
7. Include provenance information on how datasets were used, collected or generated in the course of the project. Reference standards and methods applied.
8. Include statements on the usefulness of data for the wider public needs or possible exploitations for the likely purposes of certain parties.
9. Provide assistance for dissemination purposes of (open) data, making it easy to discover it on the web.
10. Is the metadata interoperable allowing data exchange between different metadata formats, researchers and organisations?
11. Project costs associated with implementing the DMP during and after the project. Justify the prognosticated costs.
12. Support the data management life cycle for all data produced.

Guidelines and checklists on Data Management Plans of different research (related) organisations were surveyed to generate this list of requirements. The most influential guidelines to this process are listed below. References like **(R1)** refer to the requirements listed above and connects guidelines or checklists to a requirement respectively. A complete list of involved organisations is available on the Web[102].

1. **Horizon 2020 (H2020)**: Horizon 2020 is a framework programme of the European Commission funding research, technological development, and innovation[103]. (DMP guidelines[104])
2. **National Science Foundation (NSF)**: The National Science Foundation[105] is a United States government agency that supports fundamental research and education in all the non-medical fields of science and engineering. (DMP guidelines[106])

[102] http://wiki.dbpedia.org/use-cases/data-management-plan-extension-dataid\#Organisation

[103] https://ec.europa.eu/programmes/horizon2020/

[104] https://ec.europa.eu/research/participants/data/ref/h2020/grants_manual/hi/oa_pilot/h2020-hi-oa-data-mgt_en.pdf

[105] http://www.nsf.gov/

[106] http://nsf.gov/eng/general/ENG_DMP_Policy.pdf

3. **Economic and Social Research Council (ESRC)**: The Economic and Social Research Council[107] is one of the seven Research Councils in the United Kingdom and provides funding and support for research and training work in social and economic issues. (DMP guidelines[108])
4. **Deutsche Forschungsgemeinschaft (DFG)**: The DFG is the largest independent research funding organisation in Germany. It promotes the advancement of science and the humanities by funding research projects, research centres and networks. (DMP guidelines[109])
5. **Inter-university Consortium for Political and Social Research (ICPSR)**: ICPSR[110] advances and expands social and behavioural research, acting as a global leader in data stewardship and providing rich data resources and responsive educational opportunities. (DMP guidelines[111])
6. **UK Data Archive**: The UK Data Archive[112] is the curator of the largest collection of digital data in the social sciences and humanities in the United Kingdom. (DMP checklist[113])

	R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12
H2020	✓	✓	✓	✓		✓			✓	✓	✓	✓
NSF	✓	✓	✓	✓	✓	✓		✓	✓		✓	
ESRC		✓		✓	✓		✓					
DFG	✓	✓		✓							✓	
ICPSR	✓	✓	✓	✓	✓				✓	✓	✓	✓
UKDA		✓	✓	✓	✓		✓				✓	

Table 7.1: DMP requirements of research (related) institutions (selection)

To implement these demands in an ontology, the following implications are already evident:

1. making further use of PROV-O is necessary to deal with the extensive demands for provenance,
2. a clear specification of involved agents and their responsibilities is needed,
3. an extensive description of repositories retaining the described data is inescapable.

[107] http://www.esrc.ac.uk/

[108] http://www.esrc.ac.uk/funding/guidance-for-grant-holders/research-data-policy/

[109] http://www.dfg.de/download/pdf/foerderung/antragstellung/forschungsdaten/richtlinien_forschungsdaten.pdf

[110] https://www.icpsr.umich.edu/

[111] http://www.icpsr.umich.edu/files/datamanagement/DataManagementPlans-All.pdf

[112] http://www.data-archive.ac.uk/

[113] http://www.data-archive.ac.uk/create-manage/planning-for-sharing/data-management-checklist

My goal is to provide aid for researchers in drafting a DMP and implementing it with all requirements in mind: during the proposal phase, while the project is ongoing and the long-term implementation of the DMP.

7.2 Modelling the DataID approach

The final DMP use case extension, created to resolve the requirements listed in the last section, will extend DataID core together with the existing extension *Activities & Plans* and the repository ontology of *re3data.org* (cf. Section 3.2.6). All three ontologies will be outlined in this section in a concise manner. For an exhaustive description, please refer to the online documentation referenced below.

Activities & Plans To utilise the full breath of provenance offered by PROV-O and adding further semantics to satisfy all provenance related requirements is the purpose of this extension, providing the means to record all steps which are necessary to recreate a certain dataset, given the same data sources. Besides PROV-O and DataID core, it imports the following ontologies:

DLO The Data Lifecycle Ontology (cf. Section 3.2.7) is used to define stages of the data management lifecycle. By combining both ontologies of the data engineering side of the ALIGNED project (i.e. DLO and DataID core) [69], I established the means to portray datasets (or any data artefact) in the context of the data management life-cycle.

ORG The Organization Ontology (cf. Section 3.2.8) extends DataID core with important context information about the organisational background of involved Agents, which is an important detail when describing the interplay of ***Agents*** and ***Activities***.

To integrate DLO in the existing landscape of DataID and PROV-O concepts (Figure 7.1 - specification:[114]), I extended the list of sub-class axioms for dataid:Dataset, to inherit dlo:DataEntity[115]. DLO is used to classify different types of activities (dlo:LifeCycleProcess) and to adopt its basic semantics for an exchange of data artefacts between activities. Similar to dataid:associatedAgent and dataid:Authorization (cf. Section 5.2.6),

[114] https://github.com/dbpedia/DataId-Ontology/blob/07a6a1795637973a921a37f086d0e4f57968b1d5/acp/activitiesPlansExt.ttl

[115] Note: this is no ontology hijacking, since one reason for creating the DataID Ecosystem was to provide a controlled environment for redefining concepts or properties in the extension (cf. Section 4.4).

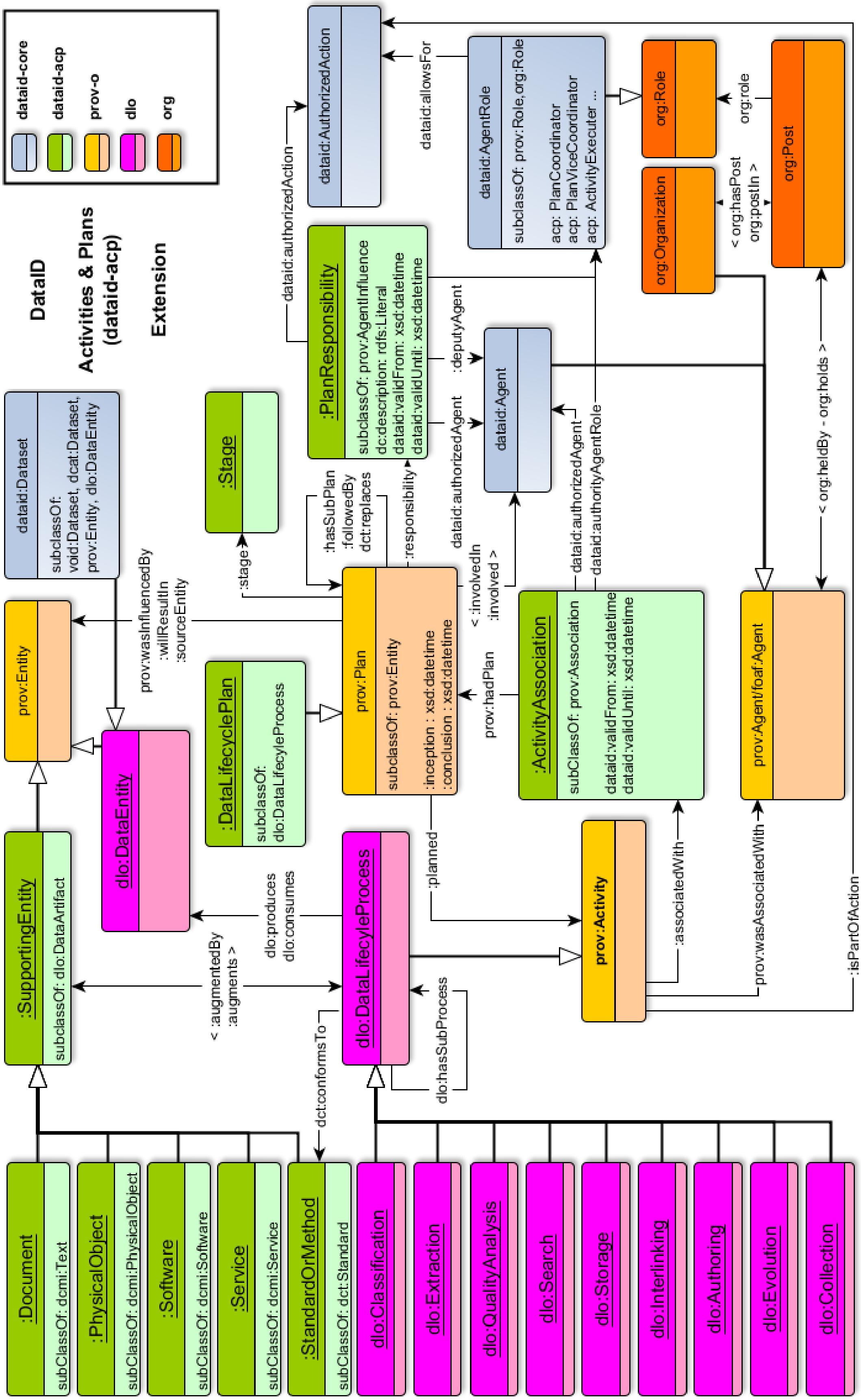

Figure 7.1: Activities & Plans extension

the property prov:wasAssociatedWith is further qualified by the class dataid-acp:ActivityAssociation, reusing relevant properties for this kind of mediator (such as dataid:authorizedAgent). This class can not only qualify a generic association with an ***Agent***, ***AgentRole*** and a time interval but provides the means of referencing a prov:Plan, which was responsible for the scheduling of an ***Activity*** in the first place.

Plans offer the necessary properties to (pre-) define a sequence of ***Activities***, governed by ***Agents*** (with responsibilities), which are generating ***Entities***. An instance of prov:Plan is constituted of multiple sub-plans, which have a defined order. This order specifies when an ***Activity*** is executed. While dataid-acp:ActivityAssociation (and its associated ***AgentRole***) specifies responsibilities for a specific ***Activity***, dataid-acp:PlanResponsibility defines responsibilities of ***Agents*** regarding the ***Plan*** itself (such as defining the order of ***Activities*** etc.).

The re3data.org ontology To cover the requirements for preservation (e.g. **(R1)**), a comprehensive description of repositories is necessary. The re3data schema [65] (cf. Section 3.2.6) does provide a thorough description of repositories and the unique opportunity to incorporate an existing, up-to-date collection of research repositories in future DataID-based applications. To accomplish the integration into the DMP ontology extension, I transformed the current XML-based schema into an OWL-ontology, using established vocabularies like PROV-O and ORG. The schema, as well as the data provided by re3data, will be available as Linked Data (e.g. via re3data ReSTful-API), thus making it discoverable and more easily accessible for services and applications, reaching a larger circle of users. This effort was support generously by the re3data.org project.

Alongside the repository concept, a rudimentary description of institutions which are hosting or funding a repository is needed to ensure long-term sustainability and availability of a repository. The derived re3data ontology (Figure 7.2) supplements r3d:Repository and r3d:Institution with fitting PROV-O sub-classes (prov:Entity, prov:Organization), making them subject to provenance descriptions. The ORG ontology is used to extend the institution class further, providing organisational descriptions.

In the ontology version of the re3data schema, I tried to combine multiple similarly structured XML elements into a single concept, where possible. For example, access regulations to the repository and the research data must be clarified, as well as the terms of use. The re3data ontology unifies all license and policy objects under the class r3d:Regulation, using the property dct:license to point out odrl:Policy descriptions of licenses, as

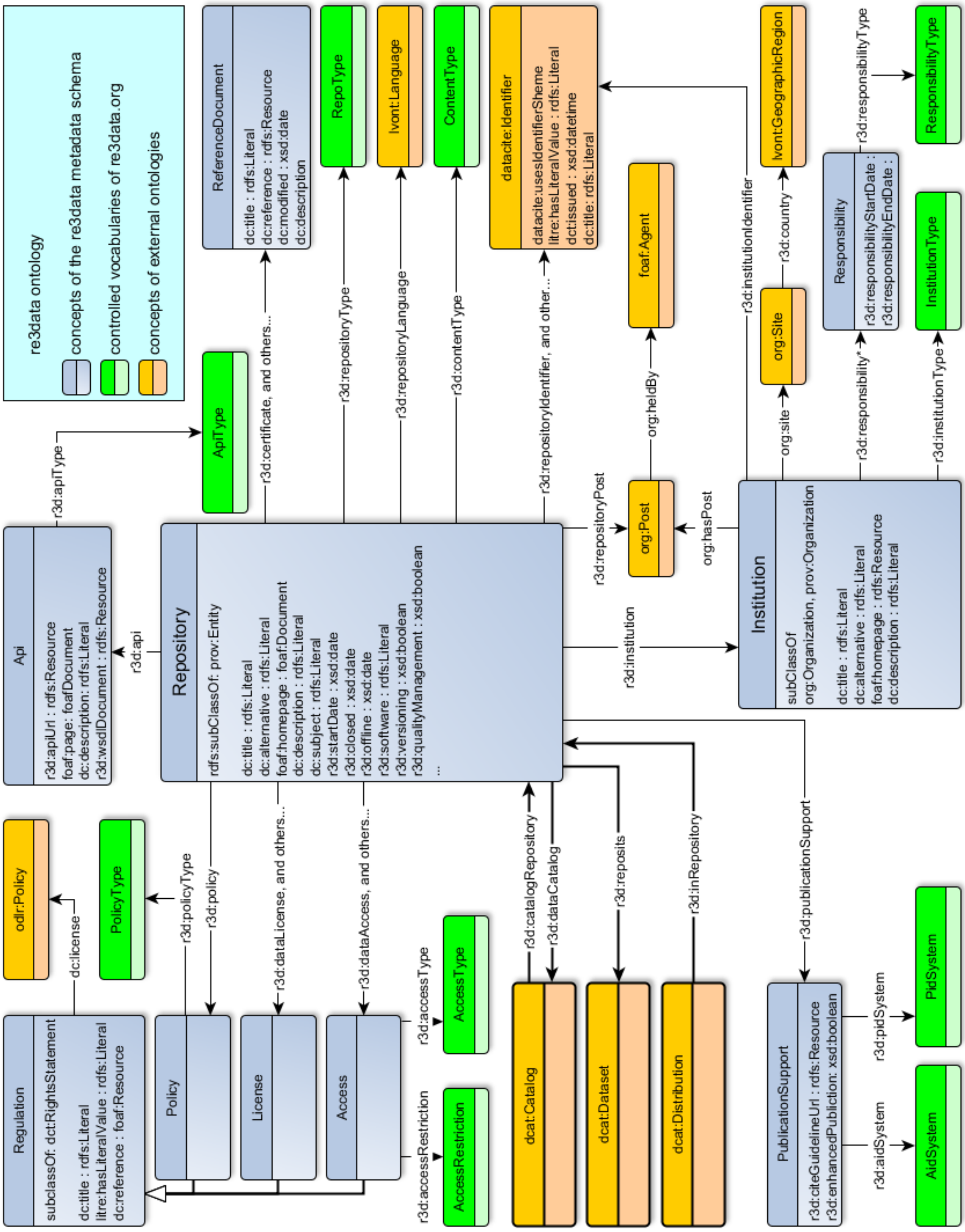

Figure 7.2: re3data.org ontology

Note: This is a reduced depiction of the ontology omitting some properties and all instances of controlled vocabularies (green boxes). A complete depiction is available here:
`https://github.com/re3data/ontology/blob/master/visuals/r3dOntology.png`
The current version can be accessed here:
`https://github.com/re3data/ontology/blob/master/r3dOntology.ttl`

used in the DataID core ontology. The ranges of multiple properties (i.e. r3d:certificate, r3d:metadataStandard and r3d:syndication) were bundled to form r3d:ReferenceDocument (a sub-class of foaf:Document).

As with DataID core, I tried to replace commonly used concepts with existing classes of well-established vocabularies. Property r3d:repositoryLanguage is pointing out instances of class lvont:Language (of the Lexvo ontology - cf. Section 3.2.3) and properties calling for identifier like structures are referencing instances of datacite:Identifier (a natural fit, due to the fact that re3data.org is now under the care of DataCite).

By linking to dcat:Catalog via r3d:dataCatalog and dcat:Dataset with r3d:reposits, we introduced the necessary means to relate descriptions of data stored in a repository. By providing this interface with the DCAT vocabulary, DataID can be used for the description of data in the re3data context.

The DMP use case Extension On top of these foundational components, we added an additional semantic layer, solving the requirements listed in Section 7.1, creating the DMP use case extension (specification: [116]). Extensive use of the PROV-O ontology and the concepts and properties introduced by the *Activities & Plans* extension is key to DMP, providing the means for describing sources and origin activities of datasets **(R7)**.

In the same vein, using the dataid:Authorization concept, augmented with a DMP specific set of dataid:AgentRole and dataid:AuthorizedAction, adds necessary provenance and satisfies requirement **(R5)** and **(R6)**.

A description of repositories involved in a DMP is provided by the concept r3d:Repository, including exact documentation of APIs and access procedures **(R1)**. More detailed information on the type of data or additional software necessary to access the data was introduced with dataid:Distribution.

As in DataID core, information about licenses and other limitations are provided via dct:license and dct:rights **(R4)**, or the complementary properties of the re3data ontology concerning access and other policies. Helpful information on usefulness, reusability and other subjects for possible users of the portrayed datasets are added to the dataid:Dataset concept: dataid:usefulness, dataid:reuseAndIntegration, dataid:exploitation etc. **(R8)**.

Requirement **(R3)** is intrinsic to DataID and needs no further representation, while **(R10)** is exemplified by the next chapter (Chapter 8).

[116] https://github.com/dbpedia/DataId-Ontology/blob/07a6a1795637973a921a37f086d0e4f57968b1d5/dmp/dataManagementPlanExt.ttl

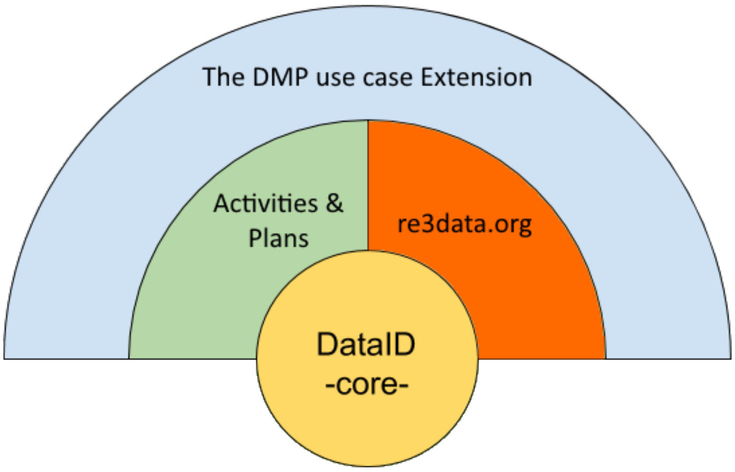

Figure 7.3: DMP use case

Note: The re3data ontology is of cause not part of the common layer of the DataID Ecosystem and does not import DataID core. The placement of ontologies was chosen for convenience.

The heart of the DMP extension (Figure 7.4) are two subclasses of prov:Plan: The dmp:DataManagementPlan provides the most general level of textual statements about the DMP itself or the planned dissemination process **(R9)**, as well as the necessary references to pertaining projects. While dmp:PreservationPlan entities can describe different approaches for the preservation of different datasets **(R2)** or provide temporal scaling (e.g. regarding embargo periods). Besides textual statements about general goals and provisions for security and backup, using the dataid-acp:planned property to point out specific tasks, put in place to preserve data long term, is one of the more notable provenance information.

The concept dmp:BudgetItem is an optional tool to list costs of activities, responsibilities (consequently costs of agents) and any entity involved in a plan like dmp:PreservationPlan. Together with dmp:approxCost and dmp:justification it satisfies requirement **(R11)**.

Several functional requirements raised by the guidelines of research funding bodies (which are not included in the requirements of this section) will be covered by the DataID service (cf. Chapter 10). It will provide a versioning system for DataIDs (based on properties like dataid:nextVersion), enabling features like tracking changes to a DataID over time. Thereby, the full Data Management Lifecycle of datasets is supported **(R12)**. Not only by the formal description with dataset metadata but by the tooling being created around it.

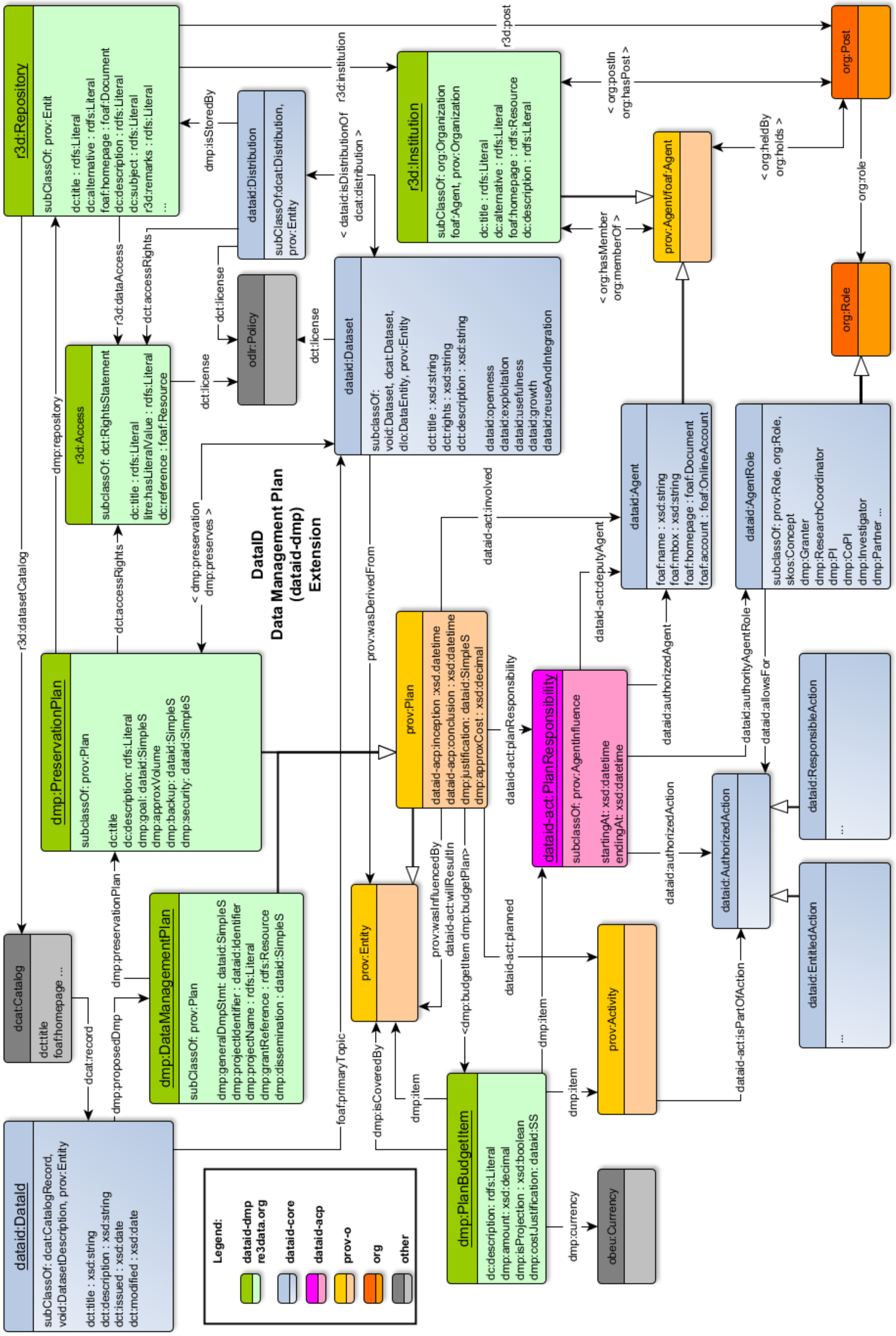

Figure 7.4: DMP use case extension

7.3 Summary

I created 3 classes and 17 properties for this use case extension, which, together with the concepts and properties introduced by the `re3data` ontology and the *Activities & Plans* extension, can describe Data Management Plans as demanded by the requirements of Section 7.1.

This approach of extending and specialising the use of `DataID core`, while reusing (partial) solutions to more specific problems, demonstrates the capability of the `DataID Ecosystem` to adapt to any use case for dataset metadata, without impacting the kernel nature of `DataID core`. Thereby, proving its Extensibility.

This section demonstrated the first two stages of the engineering lifecycle for `DataID` metadata parallel to its counterparts in the Data Management Lifecycle (cf. Section 6.2). The stages of *Generate, Publish and Exploit* are outside of the scope of this document.

8 The Interoperability of DataID

This section is focused on the Interoperability of DataID core with other dataset metadata vocabularies. As discussed in Section 3.3.3, Interoperability is more easy to establish, when interoperating with an upper (general) ontology. In the case of DataID, Interoperability is achieved on a semantic level (the fourth level of conceptual interoperability - cf. Section 3.3.2) without consideration for contextual restrictions on properties. Vocabularies imported into DataID core are conceivably easy to interoperate with, since an agreement on the metadata format exists (i.e. Dublin Core, DCAT, PROV-O, VoID). Other vocabularies need a metadata mapping to achieve full Interoperability.

8.1 DCAT Application Profile

I will demonstrate how easy a metadata mapping can be achieved between DataID core and another DCAT-based vocabulary. DCAT-AP is an application profile (cf. Section 3.1.6) of DCAT which also imports the ADMS (cf. Section 3.1.5), Dublin Core and other vocabularies [30]. Properties of the DCAT or Dublin Core vocabularies only need minimal attention to achieve a mapping. In fact, only those properties which are restricted in range (such as dct:license, dcat:mediaType or dct:language) are of concern. Others need no mapping at all. Since DCAT-AP uses the same basic concept structure and the same levels of abstraction, no special consideration of redistributing properties over different concepts with different meanings has to be undertaken. When converting DataID metadata to DCAT-AP, information about ***Authorizations*** and all pertaining concepts are lost. Some associated agent relations might be converted to standard properties such as dct:publisher or dct:maintainer.

When converting into a DataID, properties of the ADMS vocabulary, reused by DCAT-AP, will need either a mapping or are otherwise dealt with. The easiest way to resolve most of the issues in this particular instance, would be to extend DataID core with the ADMS vocabulary if the surrounding circumstances would allow for this change.

This is the list of all properties of the DCAT-AP vocabulary, which would need a mapping (or other actions) to interoperate safely with DataID core:

property	reason or possible solution
adms:status	missing equivalent
adms:identifier	property mapping: dataid:identifier
adms:sample	property mapping: void:exampleResource
adms:versionNotes	missing equivalent
dct:license	range restriction: odrl:Policy
dct:rights	range restriction: dataid:SimpleStatement
dct:accessRights	range restriction: dataid:SimpleStatement
dct:conformsTo	range restriction: dataid:SimpleStatement
dct:provenance	range restriction: dataid:SimpleStatement
dct:language	range restriction: lvont:Language
dct:format	property mapping: dcat:mediaType
dct:identifier	property mapping: dataid:literal
skos:notation	property mapping: dataid:literal

Table 8.1: Properties of the DCAT-AP vocabulary in need of a mapping to DataID core

Possible resolutions:

missing equivalent: DataID core has no suitable property to reflect this information. To resolve this mapping, either an extension to DataID core is constructed, an unused property (e.g. from DCAT) is re-purposed (which I strongly discourage) or it is neglected all together.

property mapping: An existing property is available in DataID core to convey this kind of information. In where the ranges of both properties do not match, a transformation of the object instance/value has to be performed (see next item).

range restriction: DataID core imposes a range restriction on this property. As a result, the values used with DCAT-AP often need to be transformed into an instance of the concept it is restricted to (e.g. for dct:language: literal "en" transformed to <http://lexvo.org/id/iso639-3/eng>).

I will not produce an actual mapping here. It is obvious that the extend of this effort can not be immense. One advantage of basing DataID on DCAT is the evident ease in with many metadata formats can be interoperated with.

8.2 Component MetaData Infrastructure (CMDI)

This section exemplifies how diverse metadata formats like CMDI can easily be transformed into a DataID, by providing a metadata mapping between them.

The variety of supported resource types by this component-based framework (cf. Section 3.1.9), may lead to a substantial effort in aligning existing instances to their DataID counterparts. Earlier work on the conversion of CMD profiles into RDF/RDFS[20] reflects the complete bandwidth of CMDI-based metadata, but also some idiosyncrasies that may constrain its usage in other contexts. It is expected that a transformation of relevant data to a uniform, DataID-based vocabulary will enhance visibility and exploitation of CMDI resources in new communities. Despite the fact that currently more than 80 CMD profiles are actually in use, the amount of metadata instances created on their basis, is far from being equally distributed. I created explicit mappings for CMD profiles, accountable for 56% of all publicly available metadata files, matching the appropriate DataID classes and applied them on all respective instance files via XSPARQL[117]. An overview of created mappings can be found on Github[118].

CMD profile	CMD instances (in % of all)	Supported properties of dataid:Dataset	Supported dataid:AgentRoles
OLAC-DcmiTerms	156.210 (17,4%)	13	3
Song	155.403 (17,3%)	9	1
imdi-session	100.423 (11,2%)	9	2
teiHeader	87.533 (9,7%)	10	2

Table 8.2: Most popular CMD profiles and their completeness regarding DataID classes

The creation and further adaptation of these mappings showed that the support of data considered essential in DataID differs between all profiles. The summary table 8.2 demonstrates this effect for primary properties of dataid:Dataset and the support of different ***AgentRoles*** specified in dataid:Agent. Apparently there is a varying degree of conformance of both approaches, indicating possible shortcomings in specific CMD profiles. An example for such a potential deficit is the fine-grained modelling of involved persons or organisations via DataID's Agent concept that is only partially supported in most profiles. Some CMD profiles can not be covered solely with DataID core. In these cases, an specific extension (or the reuse of existing DataID extensions) is necessary to convey the complete metadata document with DataID.

[117] https://www.w3.org/Submission/xsparql-language-specification/

[118] https://github.com/dbpedia/Cmdi-DataID-mappings

9 Evaluation

9.1 Publishing DBpedia datasets with DataID metadata

In this first approach to evaluate DataID, I want to have a closer look on the already presented example of DBpedia datasets (cf. Section 5.2) in affiliation with DataID metadata. Therefore, this section will not only evaluate metadata specific aspects of this example, but tries to measure the overall effect of better metadata on datasets. I will do so by evaluating against the best practices presented by the W3C 'Data on the Web' working group and the FAIR Principles (cf. Section 2.1.5). In both cases I will gauge each practice by its state of fulfilment for the DBpedia use case. I will assign one of the following ratings to every practice: **(2)** The requirement is supported in full. **(1)** The requirement is partially met. **(0)** DBpedia does not support this requirement. A short statement will explain the decision. While this is an evaluation of the DBpedia datasets against these practices, many of them are dependent on the DataID documents, published alongside the datasets, for completion. To point out these instances, a second look at these particular practices is taken in Section 9.2.1.

maybe rephrase

9.1.1 Data on the Web Best Practices

This collection presented by the 'Data on the Web' (DWBP) W3C working group[119] and their best practices document [48] (released as recommendation candidate by the time of writing) is the most comprehensive selection of best practices on data publishing available (cf. Section 6.1). Fortunately, the 'Data on the Web' working group decided early on to include DBpedia as a candidate in their implementation report. The following evaluation of DBpedia with DataID was done in collaboration with the W3C working group. A complete table of this evaluation is available on the Web[120].

[119] https://www.w3.org/2013/dwbp/wiki/Main_Page

[120] http://dataid.dbpedia.org/ns/dwbp.html

1. **Provide metadata** *Provide metadata for both human users and computer applications.* - DataID is dataset metadata. (2)

2. **Provide descriptive metadata** *Provide metadata that describes the overall features of datasets and distributions.* - This is the basic idea of DataID. (2)

3. **Provide structural metadata** *Provide metadata that describes the schema and internal structure of a distribution.* - DataID can portray this with void:vocabulary and the information available through dataid:MediaType. (2)

4. **Provide data license information** *Provide a link to or copy of the license agreement that controls use of the data.* - This is achieved with instances of odrl:Policy. (2)

5. **Provide data provenance information** *Provide complete information about the origins of the data and any changes you have made.* - This is a central concept of DataID core. (2)

6. **Provide data quality information** *Provide information about data quality and fitness for particular purposes.* - Data quality can not be represented with DataID core. Yet, a data quality extension is already planned (cf. Chapter 10). (0)

7. **Provide a version indicator** *Assign and indicate a version number or date for each dataset.* - Property dct:hasVersion is provided for each dataset. (2)

8. **Provide version history** *Provide a complete version history that explains the changes made in each version.* - indirectly provided through the diff of RDF statements between DataID documents for different versions. (1)

9. **Use persistent URIs as identifiers of datasets** *Identify each dataset by a carefully chosen, persistent URI.* - Each dataset has a unique URI which is persistent and identifies the DBpedia language edition as well as the DBpedia version. (2)

10. **Use persistent URIs as identifiers within datasets** *Reuse other people's URIs as identifiers within datasets where possible.* - DBpedia URIs are persistent. Also, many URIs of equivalent instances in other datasets are referenced (linked) with owl:sameAs. (2)

11. **Assign URIs to dataset versions and series** *Assign URIs to individual versions of datasets as well as to the overall series.* - URIs of datasets and DataIDs are divided into a) the type of dataset (without language and version indicator - as a general identifier), b) for language specific datasets and c) for language and DBpedia release version specific dataset. (2)

12. **Use machine-readable standardized data formats** *Make data available in a machine-readable, standardized data format that is well suited to its intended or potential use.* - Datasets are available in RDF. **(2)**

13. **Use locale-neutral data representations** *Use locale-neutral data structures and values, or, where that is not possible, provide metadata about the locale used by data values.* - This is not achieved for every possible datatype in `DBpedia`. However, commonly used types (such as dates) are locale-neutral. **(1)**

14. **Provide data in multiple formats** *Make data available in multiple formats when more than one format suits its intended or potential use.* - `DBpedia` provides data in two RDF serialisations as well as a table representation of selected datasets. **(2)**

15. **Reuse vocabularies, preferably standardized ones** *Use terms from shared vocabularies, preferably standardized ones, to encode data and metadata.* - The `DBpedia` ontology reuses multiple vocabularies (e.g. `DCterms`, `OWL`, `RDFS`, etc.). `DataID` imports well-established metadata formats (such as `DCAT` and `PROV-O`). **(2)**

16. **Choose the right formalization level** *Opt for a level of formal semantics that fits both data and the most likely applications.* - This is a difficult task for `DBpedia`, since it is a community-based effort. In general, the `DBpedia` ontology is kept as shallow (or abstract) as possible. **(1)**

17. **Provide bulk download** *Enable consumers to retrieve the full dataset with a single request.* - `DBpedia` offers its datasets as bulk downloads. **(2)**

18. **Provide Subsets for Large Datasets** *If your dataset is large, enable users and applications to readily work with useful subsets of your data.* - `DBpedia` provides not the whole data releases as one file but divided into languages and subdatasets, which are available as bulk downloads. **(2)**

19. **Use content negotiation for serving data available in multiple formats** *Use content negotiation in addition to file extensions for serving data available in multiple formats.* - As far the official `DBpedia` SPARQL endpoint is concerned (which does not offer every dataset of a release), this is true. **(1)**

20. **Provide real-time access** *When data is produced in real time, make it available on the Web in real time or near real-time.* - The official DBpedia releases are snapshots of data at a certain point in time. However, there `DBpedia` Live[121] which offers real time access to the current `Wikipedia` data. **(1)**

21. **Provide data up to date** *Make data available in an up-to-date manner, and make the update frequency explicit.* - See practice 20. **(1)**

[121] `http://live.dbpedia.org`

22. **Provide an explanation for data that is not available** *For data that is not available, provide an explanation about how the data can be accessed and who can access it.* - DBpedia's primary data provided are static dump files, which should always be accessible, for every release. The data not represented in the public endpoint is not accounted for its absence there. **(0)**

23. **Make data available through an API** *Offer an API to serve data if you have the resources to do so.* - Some of the data (mostly from the English language edition) is available via the official SPARQL endpoint of DBpedia. **(1)**

24. **Use Web Standards as the foundation of APIs** *When designing APIs, use an architectural style that is founded on the technologies of the Web itself.* - Provided for by the official SPARQL endpoint. **(2)**

25. **Provide complete documentation for your API** *Provide complete information on the Web about your API. Update documentation as you add features or make changes.* - The official DBpedia endpoint conforms to SPARQL 1.1. its documentation is available from its provider, Open Link[122]. **(2)**

26. **Avoid Breaking Changes to Your API** *Avoid changes to your API that break client code, and communicate any changes in your API to your developers when evolution happens.* - This is outside of the scope of DBpedia, but since the endpoint adheres to SPARQL 1.1 changes are rare and are adopted over time by Open Link. **(1)**

27. **Preserve identifiers** *When removing data from the Web, preserve the identifier and provide information about the archived resource.* - DBpedia follows Wikipedia when it comes to deleted wiki pages, providing dbo:redirect, pointing out the resource Wikipedia is redirecting to. The identifier itself is preserved. **(2)**

28. **Assess dataset coverage** *Assess the coverage of a dataset prior to its preservation.* - This is a difficult topic for DBpedia since its reach and growth is unpredictable and driven by a community. **(0)**

29. **Gather feedback from data consumers** *Provide a readily discoverable means for consumers to offer feedback.* - At the moment Feedback is collected via multiple mailing lists. **(2)**

30. **Make feedback available** *Make consumer feedback about datasets and distributions publicly available.* - All current and future means of feedback will be readily available for anyone. **(2)**

[122] https://virtuoso.openlinksw.com/dataspace/doc/dav/wiki/Main/VOSSparqlProtocol

31. **Enrich data by generating new data** *Enrich your data by generating new data when doing so will enhance its value.* - New datasets are being created, for example, based on NLP algorithms. **(2)**

32. **Provide Complementary Presentations** *Enrich data by presenting it in complementary, immediately informative ways, such as visualizations, tables, Web applications, or summaries.* - This is a task for the `DBpedia` community. `DBpedia` does provide releases as table representations. **(1)**

33. **Provide Feedback to the Original Publisher** *Let the original publisher know when you are reusing their data. If you find an error or have suggestions or compliments, let them know.* - `Wikipedia` as the original publisher of most of the data has a cumbersome (wiki-based) interface for relaying feedback. `DBpedia` does no extend the feedback loop back to `Wikipedia` yet. **(0)**

34. **Follow Licensing Terms** *Find and follow the licensing requirements from the original publisher of the dataset.* - `DBpedia` does follow the licensing terms of `Wikipedia`. **(2)**

35. **Cite the Original Publication** *Acknowledge the source of your data in metadata. If you provide a user interface, include the citation visibly in the interface.* - `DBpedia` points out the source in the dataset metadata (the original XML dump files). **(2)**

In summary, `DBpedia` does support 31 of the 35 best practices at least partially, and 22 to their full extent. This is also evident from the official implementation report of the 'Data on the Web' working group[123]. In this report, 59 datasets, data portals or vocabularies from different domains were evaluated against all 35 practices, "[...] in order to demonstrate that each of the best practices has been recommended or adopted in at least two environments [...]" [15]. Figure 9.1 shows the evidence gathered for each of the best practices by the working group. Figure 9.2 compares those of the 59 candidates, which implement at least seven best practices. A description of every candidate and general information on methodology and implementation is available in the report[124].

[123] `http://w3c.github.io/dwbp/dwbp-implementation-report.html`

[124] `https://www.w3.org/2013/dwbp/wiki/BP_Implementation_Report`

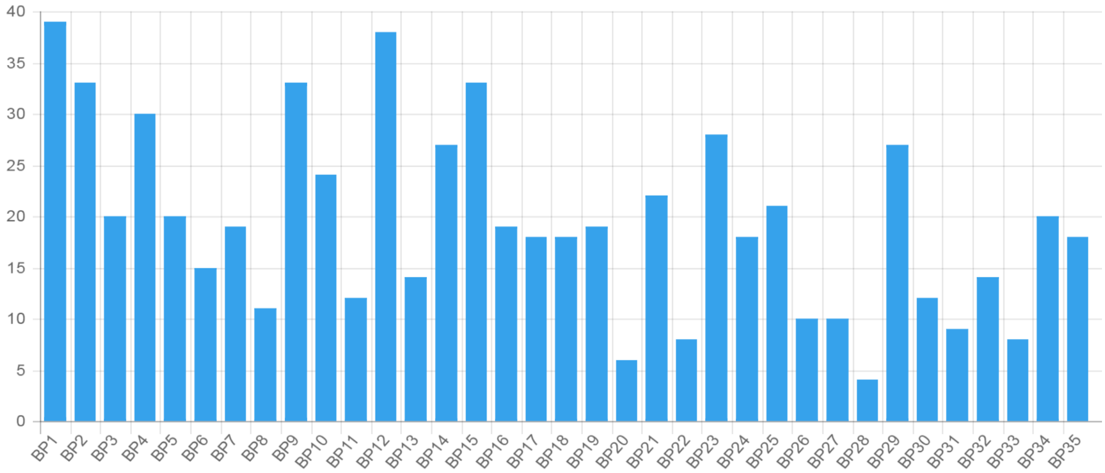

Figure 9.1: DWBP Evidence: Number of candidates implementing a given best practice. [15]

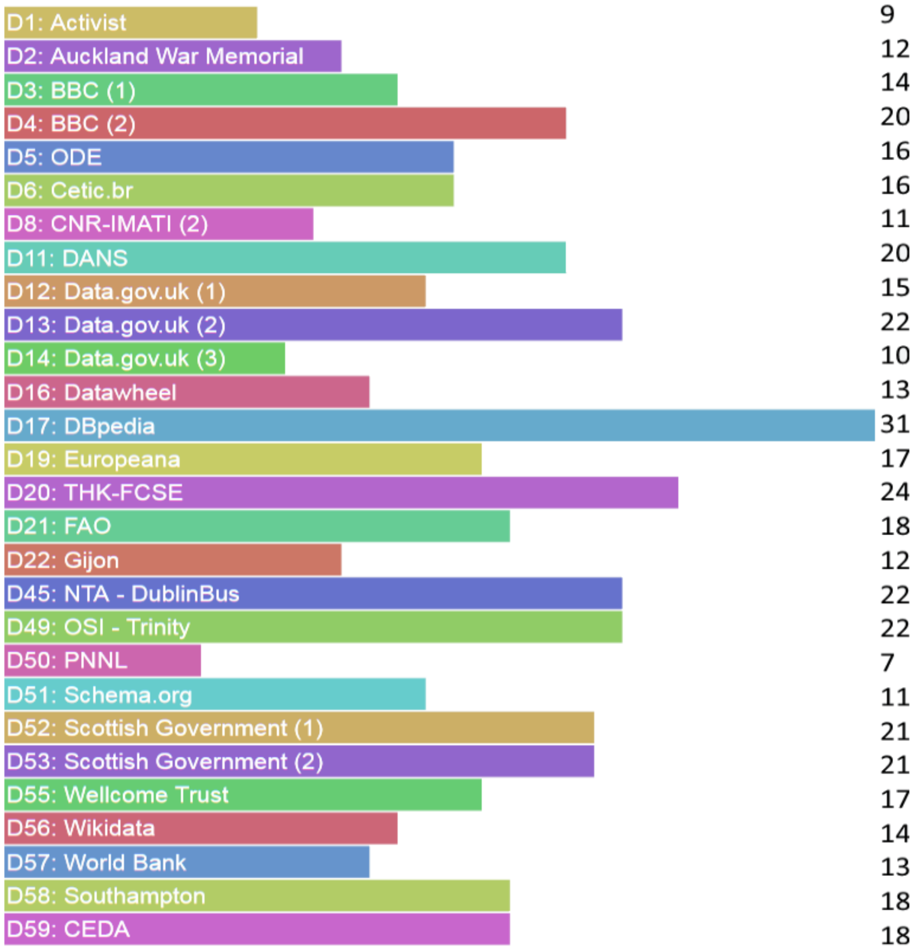

Figure 9.2: DWBP Implementation Report Summary: Candidates with number of practices implemented.

9.1.2 The FAIR Data Principles

As in the previous section, I want to evaluate `DBpedia` with its `DataID` metadata against this collection of principles. In this case, I want to find out if `DBpedia` offers not only Linked Open Data but if this data is also **FAIR** (cf. Section 2.1.5).

F To be Findable:

1 (meta)data are assigned a globally unique and persistent identifier

`DataID` & `DBpedia` provide unique identifiers **(2)**

2 data are described with rich metadata (defined by R1 below)

`DBpedia` provides rich metadata with `DataID` **(2)**

3 metadata clearly and explicitly include the identifier of the data it describes

`DataID` includes the identifiers of datasets **(2)**

4 (meta)data are registered or indexed in a searchable resource

`DBpedia` provides the official `DBpedia` SPARQL endpoint **(1)**

A To be Accessible:

1 (meta)data are retrievable by their identifier using a standardized communications protocol

i. the protocol is open, free, and universally implementable

ii. the protocol allows for an authentication and authorization procedure, where necessary

the official `DBpedia` SPARQL endpoint supports these requirements **(1)**

2 metadata are accessible, even when the data are no longer available

`DataID` documents will remain available **(2)**

I To be Interoperable:

1 (meta)data use a formal, accessible, shared, and broadly applicable language for knowledge representation.

(Meta)data is RDF based on well-documented ontologies in both cases. **(2)**

2 (meta)data use vocabularies that follow FAIR principles

The `DBpedia` ontology and `DataID core` allow for FAIR principles. **(2)**

3 (meta)data include qualified references to other (meta)data

This is especially true for both DBpedia data and DataIDs. **(2)**

R To be Reusable:

1 (meta)data are richly described with a plurality of relevant attributes

True for data and metadata. **(2)**

2 (meta)data are released with a clear and accessible data usage license

True for data and metadata. **(2)**

3 (meta)data are associated with detailed provenance

DBpedia provides provenance on triple level (in the query of the graph URI), while DataIDs of DBpedia provide origin relations and qualified agent relations (both can be improved) **(1)**

4 (meta)data meet domain-relevant community standards

True for data and metadata. **(2)**

The DBpedia datasets released together with pertaining DataIDs prove to be in an excellent FAIR condition, ensuring findability, accessibility, Interoperability and reusability. The underlying prerequisite of these principles (i.e. machine-readability) is also targeted by the general approach of DataID. According to the FORCE11[125] web page about the FAIR principles, DBpedia releases can be classified as 'FAIR data with open access': "The metadata as well as the data elements themselves are fully FAIR and completely public, under well-defined license."[126] This is equivalent to the fourth (and last) level in their ranking of 'increasingly FAIR digital objects' (cf. Section 2.1.5).

DataID seems to be a very capable metadata format to assist in the effort of releasing FAIR data.

[125] https://www.force11.org

[126] https://www.force11.org/fairprinciples

9.2 Evaluating DataID as dataset metadata

This section offers an evaluation of DataID (and in particular DataID core) against multiple possible gauges. This approach will ensure an evaluation of DataID from the perspective of different communities with their requirements and circumstances.

As in the previous section, requirements can be met in full **(2)**, partially **(1)** or not at all **(0)**. These ratings describe the ability of a vocabulary to fulfil a requirement and not an actual use case.

9.2.1 Comparison of DataID core and DCAT

In this section, I want to extrapolate a general adequacy of DataID metadata, from results of the last section, regarding the best practices reviewed.

Table 9.1 singles out any best practice of the last section for which dataset metadata is a requirement, or it has a direct influence on the expected results. The compliance of these practices by DataID core will be compared to the compliance of metadata recorded based solely on the DCAT vocabulary.

Note: I have taken the liberty to curtail the language of some listed practices. Due to the lack of space, I also shortened DataID core to DataID in the heading of the table. Also, while my decisions for each rating for core can be found in the previous section, in the case of DCAT this is not the case. Section 3.1.1 discusses the shortcomings of DCAT in length, a reiteration in this chapter is not necessary.

I conclude that DataID core offers significant improvements, compared to DCAT, regarding Provenance, Licensing, Access, machine-readability, reusability and Interoperability. Further, a general increase in the richness of describable features is evident, aiding, for example, to describe citations, structural metadata, versioning and coverage. I grant that multiple aspects still need further improvement (e.g. data quality description or specific properties for deprecated data etc.). DataID core is a general approach for dataset metadata by design. Its ability to extend easily (cf. Chapter 7) and the explicit context in which to do so (the DataID Ecosystem), provides all necessary means to cover those requirements in a simply achieved extension.

Ref.	Best Practice	DataID	DCAT
	Data on the Web Best Practices		
1	Provide metadata	(2)	(2)
2	Provide descriptive metadata	(2)	(2)
3	Provide structural metadata	(2)	(1)
4	Provide data license information	(2)	(1)
5	Provide data provenance information	(2)	(0)
6	Provide data quality information	(0)	(0)
7	Provide a version indicator	(2)	(2)
8	Provide a version history	(1)	(1)
9	Use persistent URIs as identifiers of datasets	(2)	(2)
12	Use machine-readable standardized data formats	(2)	(1)
13	Use locale-neutral data representations	(1)	(1)
15	Reuse vocabularies, preferably standardized ones	(2)	(2)
22	Provide an explanation for data that is not available	(1)	(1)
28	Assess dataset coverage	(2)	(1)
35	Cite the Original Publication	(2)	(1)
	FAIR Data Principles		
F1	globally unique and persistent identifier	(2)	(2)
F3	explicitly includes the identifier of data described	(2)	(1)
I1	uses formal, accessible, shared, applicable language	(2)	(2)
I2	uses vocabularies that follow FAIR principles	(2)	(2)
I3	includes qualified references to other metadata	(2)	(0)
R1	is described with a plurality of relevant attributes	(2)	(1)
R3	is associated with detailed provenance	(2)	(0)
R4	meets domain-relevant community standards	(2)	(2)

Table 9.1: Comparison of `DataID core` and `DCAT` against metadata relevant best practices of DWBP and FAIR

9.2.2 Comparing DataID to a host of vocabularies

I want to go a pace further and compare `DataID core` to every vocabulary discussed in Section 3.1, as well as some of their combinations. This will provide a clearer picture of what has been accomplished with this work.

While the means of evaluation in the previous section, based on the principles discussed before, is a useful tool for comparing vocabularies, it is not sufficient. Important aspects of dataset metadata are not reflected in the best practices (such as reflection on Extensibility and Interoperability). Therefore, I will use the principle goals of dataset metadata, which I introduced in Section 3.1, to evaluate `DataID core` against other metadata vocabularies.

A **The vocabulary encourages the use of richly described and machine-readable resources.** This is one of the underlying objectives of `DataID`; to provide machine-readable resources where possible (cf. Section 1.2). **(2)**

B **The vocabulary assigns globally unique URIs to metadata resources.** That is a given for every RDF-based vocabulary. **(2)**

C **The vocabulary can describe data access and access restrictions, consumable for humans and machines alike.** `DataID core` provides detailed descriptions of media types `dataid:MediaType` and other related aspects (e.g. access procedures). Additional effort is needed to describe, for example, API endpoints. Extensions for such a purpose is subject to future work. **(1)**

D **The vocabulary can portray provenance information extensively.** `PROV-O` is imported into `DataID core` and additional qualifications were created (such as `dataid:Authorization`. **(2)**

E **The vocabulary provides for detailed descriptions of rights and licenses.** Licenses and other policies can be described in detail with `odrl:Policy`. **(2)**

F **The vocabulary provides properties to cite identifiers of the data described.** This is achievable with axioms of the `PROV-O` ontology. The *Activities & Plans* extension of `DataID` offers the class `dataid-acp:SupportingEntity` which (with its sub-classes) can further delimit such entities. **(1)**

G **The vocabulary provides for qualified references between resources.** In addition to the natural qualifying mechanisms of `PROV-O`, `DataID core` provides for further qualifications for related Agents and Datasets. **(2)**

H **The vocabulary is easy to extend, to fit any given use case.** This feature was demonstrated with Chapter 7 of this thesis. **(2)**

I **The vocabulary is unambiguous and easy to map to other metadata vocabularies.** DataID core does not impede the Extensibility of DCAT and is therefore easy to map to other vocabularies (cf. Chapter 8). **(2)**

J **The vocabulary offers additional properties to aid dissemination and discovery.** Multiple properties have been introduced for Datasets to further describe its intended purpose and other useful information, applicable to dissemination tasks. **(2)**

This table includes all evaluations of that kind from Section 3.1, compared to DataID core. I also included some amalgamation between DCAT and other ontologies to demonstrate how metadata based on such an approach would fare. This rather crude method of gauging the quality of a vocabulary is of cause plagued with inadequacies, but it reflects the broader aspects of the DataID approach compared to other dataset metadata in an assessable manner.

Vocabulary	A	B	C	D	E	F	G	H	I	J	Sum
DCAT	1	2	1	0	0	1	0	2	2	1	10
VoID	1	2	0	0	0	0	1	1	2	1	8
CKAN	0	2	0	0	0	0	0	1	1	0	4
META-SHARE	2	2	1	1	2	0	0	0	1	1	10
ADMS	1	2	1	0	0	1	0	2	2	1	10
DCAT-AP	1	2	1	0	0	1	0	1	2	1	9
HCLS Profile	2	2	2	1	0	1	0	2	1	1	12
CERIF	2	2	1	1	0	2	2	2	2	1	15
CMDI	1	2	0	0	1	2	0	2	0	1	9
DataID 1.0.0	1	2	1	1	2	1	2	1	2	1	14
DCAT & PROV-O	1	2	1	1	0	1	2	2	2	1	13
DCAT & VoID	1	2	1	0	0	1	1	2	2	1	11
DCAT & PROV-O & VoID	1	2	1	1	0	1	2	2	2	1	13
DataID core	2	2	1	2	2	1	2	2	2	2	18

Table 9.2: Comparison of dataset vocabularies introduced in Section 3.1 to DataID core

As the table indicates, DataID has made some strides to cover aspects of dataset metadata, which are inadequately represented or neglected altogether by other metadata formats. Especially aspects such as the representation of Licensing, Provenance, machine-readability, Access, discoverability, Extensibility and Interoperability were accomplished.

9.2.3 Implementation of Objectives

Improving the portrayal of Provenance, Licensing and Access, while maintaining the easy Extensibility and Interoperability of `DCAT`*, are the linchpin objectives in my effort to present a comprehensive, extensible and interoperable metadata vocabulary* (cf. Section 1.2).

With the evidence already gathered in this section, I want to assess the fulfilment of those goals set for `DataID` in Section 1.2. I will list all possible proof for accomplishing each objective as a foundation for a final judgement of success.

Objective 1. Provide sufficient support for extensive and machine-readable representations for Provenance, Licensing and data Access.

Beginning with the very first version of `DataID`, Provenance was one of the topmost items on my list to improve dataset metadata, as an omnipresent requirement in most use cases with large or complex datasets. With the conception of the Provenance Ontology (`PROV-O`) and its characteristic way of representation and interchange of provenance information on the Web, the ideal companion ontology for `DCAT` was found. By creating the means to qualify relations, (re-) introducing the effective approach of inserting relationship objects as mediators between subject and object, relationships can express roles, temporal or spatial restrictions and experience state changes. This measure ensures the referential and functional integrity of the involved Entities.

talk about referential integrity

In addition to the basic possibility to qualify relations between Entities, Agents and Activities, `DataID core` specialised in particular on relations between (dataset) Entities and Agents. With `dataid:Authorization` a qualification object for such relations was introduced, providing a record about what an Agent is allowed to do with a particular collection of Entities. The `dataid:DatasetRelationship` mediator between different datasets allows for a well-defined possibility to declare any relation, two datasets might enter into. Currently, the `DataID Ecosystem` provides the extension *Activities & Plans*, which further specifies ways to record Provenance, focusing on Activities and Plans. Its applicability was demonstrated in Chapter 7.

The better portrayal of Licensing information was accomplished by the adoption of the `ODRL` ontology to represent any policy or license agreement. `ODRL` descriptions provide a flexible and interoperable mechanisms to support the transparent use of digital content in publishing, distribution, and consumption of digital media in a machine-readable way. Restricting the range of `dct:license` to instances of `odrl:Policy`, ensuring the use of machine-readable licenses, offers a way to automatically decide if a dataset is admissible in the context of a particular transformation activity.

Various efforts have been made to support a sustainable mode of Access to the digital objects described in a DataID document. The initial idea of separating Datasets as a concept and their possible manifestations in Distributions was a significant improvement of the DCAT vocabulary to previous representations. Alas, DCAT is too generic, which leaves too much room for interpretation and ambiguity.

To alleviate this lack of specificity, I implemented the dataid:MediaType class, to define the type of a digital object in a machine-readable way. Multiple subclasses of dataid:Distribution offers an interface for further specification (e.g. for API endpoints). Additional properties (such as dataid:checksum aid the acquisition of data in an automatic processing environment.

Objective 2. Extend DCAT with well-established ontologies to resolve the pressing issues with dataset metadata if possible

Throughout the DataID Ecosystem, I tried to reuse existing ontologies which are either well-establish in the Semantic Web community (preferably W3C recommended), or innovative and of sufficient maturity to contribute its semantics to DataID. This table shows the result of this effort. Each reused ontology from the DataID Ecosystem is recorded here, together with its origin and a statement about its maturity. Those ontologies used in DataID core are marked with an asterisk.

Ontology	Origin	Maturity
Dublin Core*	DCMI Initiative	deemed a Good Ontology (W3C)
DCAT *	W3C Working Group	W3C recommended
PROV-O *	W3C Working Group	W3C recommended
VoID *	W3C Interest Group	W3C Interest Group specification
FOAF *	The FOAF project	deemed a Good Ontology (W3C)
DataCite*	DataCite	Part of the SPAR ontology suite
DLO	ALIGNED project	parallel developed with DataID
LVONT *	Lexvo.org	widely used ontology throughout the NLP community
ODRL *	W3C Community Group	final specification
ORG	W3C Working Group	W3C recommended
SD	W3C Working Group	W3C recommended
SPDX *	Linux Foundation	specification

Table 9.3: All external vocabularies reused by ontologies of the DataID Ecosystem

DataID core only imports recommended or otherwise supported ontologies by the W3C. Ontologies, such as LVONT, ODRL and SPDX are used to restrict some property ranges but are not imported, as to keep a separation between DataID and developments of the origin environment of these ontologies.

datacite?

Objective 3. Show, that by modularising into a landscape of ontologies, DataID preserves the general character of DCAT, supporting Extensibility and Interoperability.

As discussed in Section 4.2, to keep DataID as extensible and interoperable as DCAT, I decided to modularize DataID into multiple extension ontologies around a single core.

The modular approach paired with the effort to keep DataID core as general (simplistic) as possible, without compromising on the goals lined out, has proven to be an expedient solution. As a side effect, DataID core adheres to profile OWL2 RL[127] which adds additional possibilities to use cases where reasoning over metadata is of concern and simplifies the extending of this ontology.

put this somewhere not in a footnote

Modularity and simplicity are general aspects of good metadata (cf. Section 2.1.4) and can be seen as the foundation for subsequent traits such as Extensibility and Interoperability. Both of these characteristics were demonstrated in Chapter 7 and Chapter 8 respectively. With these measures in place, DataID core extends DCAT in a way, which does not impede its wide applicability in data portals and other applications. Existing DCAT solutions can therefore easily be transformed into DataID based metadata.

Objective 4. Prove that the resulting ecosystem is capable of serving for complex demands on dataset metadata (proving Extensibility).

I exemplified the implementation of specialised dataset metadata vocabularies extending DataID core in Chapter 7. Based on workflow for publishing DataID metadata parallel to its datasets in Chapter 6. I have shown that by extending DataID core with existing addendums and even external ontologies, one could satisfy complex metadata requirements like those of Data Management Plans while keeping the ability to inter-operate with other metadata vocabularies (like CMDI- cf. Chapter 8) in turn.

Objective 5. Demonstrate the Interoperability with other metadata formats.

[127] with the exception of the 5 axioms of the PROV-O ontology pointed out in the PROV-O specification: https://www.w3.org/TR/prov-o/#owl-profile

Exchanging information between systems using different metadata schemas is an important aspect in any such scenario. I have shown in Chapter 8 how DataID is supporting the Interoperability of its metadata.

- By basing DataID on DCAT, a well established and often used vocabulary for dataset metadata offering the easiest form of Interoperability: agreement on a schema.
- By modularising DataID, easing mapping efforts by reducing ambiguity.
- By allowing for extensions which could either incorporate the other vocabulary altogether or accommodate any missing expressiveness with a special extension.

Objective 6. Evaluate the universal applicability of DataID for datasets against common demands on data publications.

While objectives 1. to 5. represented the principle goals for DataID, there are other important demands on dataset metadata which DataID can serve. Most of these demands are already mentioned in Section 9.2.1. I will single out the more salient aspects.

- **Machine-readability** - DataID tries to be as specific as possible. For example, replacing simple literals with dataid:SimpleStatement or other concepts suitable for the property at hand.
- **Versioning** - DataID introduced simple (but effective) version pointers between dataset Entities.
- **Identifiers** - by adopting the identifier concept of the DataCite ontology, DataID core provides a comprehensive means to specify (alternative) identifiers, for any digital object.
- **Dissemination** - the Dataset concept of DataID core provides multiple textual statements for publishers to record intent, purpose or possible usefulness of datasets, all of which are helpful for dissemination tasks and general discoverability.
- **Rights & Responsibilities** - DataID offers a prominent feature to not only define roles an Agent might have regarding an Entity but also what kind of responsibilities and rights this entails. Exact definitions of authorisations in the context of datasets become feasible.

In summary, DataID does not only fulfil the objectives set but extends a wide range of useful additions to the core model established by DCAT.

9.2.4 Conformity to the ALIGNED principles

In the context of the ALIGNED project, `DataID` is part of a shared model of software and data engineering to enable unified governance and coordination between aligned co-evolving software and data lifecycles. I want to confirm that the overall approach of `DataID` does comply with the three principles of ALIGNED, which were set up to guide all relevant tasks:

To be part of a unified software and data engineering process; With the incorporation of the Data Lifecycle Ontology of the ALIGNED project (cf. Section 7.2) into the *Activities & Plans* extension of `DataID`, I combined both ontologies of the Data Management side of the ALIGNED Suite of Ontologies [69], specifically designed to model the information exchange needs of combined software and data engineering.

describing the complete data lifecycle and domain model; Both ontologies can jointly describe a domain data model in the context of a progressing data lifecycle and provide the necessary artefacts to aid the software lifecycle advancing in parallel.

with an emphasis on quality, productivity and agility. Rich metadata of datasets will increase the quality of any dependent process and its results. The ability to adapt easily to a given use case and its high level of Interoperability provides sufficient flexibility for agile engineering processes. The focus on machine-readability and detailed descriptions of resources is an excellent basis for a high degree of automation in data engineering workflows.

The `DataID` project started out solely as a tool to publish `DBpedia` datasets with more descriptive metadata. With the start of the ALIGNED project, which uses `DBpedia` as one of its use cases, `DataID` has been included in the ALIGNED suite of ontologies that provide semantic models of design intents, domain-specific datasets, software engineering processes, quality heuristics and error handling mechanisms [69]. The suite contributes immensely towards enabling Interoperability and alleviating some of the complexities involved.

Combining data and software engineering processes to increase quality, productivity and agility, is a challenge being faced by several organisations aiming to exploit the benefits of Big Data. Ontologies and vocabularies developed in accordance with competency questions, objective criteria and ontology engineering principles can provide useful support to data scientists and software engineers undertaking the challenge.

10 Conclusion and Future Work

This work presented a comprehensive introduction to DataID, providing structured dataset metadata for any domain, type of data or dataset structure. In particular, DataID core was portrayed in great detail as the kernel element of the DataID Ecosystem.

Based on a study of existing metadata vocabularies, an extensive list of inadequacies and further demands on dataset metadata was compiled, guiding the course of this work. As a result, the monolithic approach of a first version of the DataID ontology was modularised into multiple extension ontologies around a single core. DataID core extends the DCAT vocabulary to provide greater specificity on many important issues (such as Access, Licensing and Provenance), without encroaching on the central traits of DCAT, easy Extensibility and Interoperability with other existing dataset metadata formats. Furthermore, this thesis demonstrated the abilities of the DataID Ecosystem to accommodate for complex metadata requirements (i.e. Data Management Plans), by extending DataID core and even incorporating external ontologies (i.e. re3data ontology), while keeping the ability to interoperate with other metadata vocabularies (such as CMDI) in turn. Its ability to model complex authorisations with minimal effort for any dataset environment offers additional possibilities for comprehensive metadata demands.

DataID metadata has proven to be a very promising partner for dataset publishers, such as DBpedia, aiding not only publication tasks, but providing the comprehensive metadata needed to align with the best practices of FAIR and the Data on the Web W3C working group. Moreover, DataID has shown that it is capable of serving as a pivotal member in the ALIGNED Suite of Ontologies to cover all demands for dataset metadata in the context of the co-evolving lifecycles of software and data management.

In total, DataID core consists of 19 classes, 43 object properties, eight datatype properties and 29 named individuals. It is consistent with the OWL2 RL profile (with minor adaptations, see **??**). The DataID Ecosystem provides a controlled todo environment, in which the interplay of different extensions and DataID core is regulated and documented, making it easy to combine multiple extensions with DataID core to solve a given usage scenario. Currently, DataID core can be extended by three existing extensions from the common layer of the DataID Ecosystem, with more already planned.

Additional work has to be done in defining the proper inter-semantics of the DataID Ecosystem. In that, appropriate regulations and guidelines have to be in place before incorporating new DataID extensions into the common layer of ontologies. A toolchain for validating the interoperability of multiple extensions in one ontology would go a long way in helping users to compile their use case extensions. I have suggested the Dacura Data Curation System[128] as a possible solution.

Currently, a DataID service stack and website is being implemented, to simplify and automate the creation, validation and dissemination of DataIDs, supporting humans in creating DataIDs manually, as well as automation tasks with a service endpoint. Multiple DataID related tasks will be usable for humans and machines:

- support for the creation of dataset metadata, guiding the user through all necessary steps in creating a DataID
- providing help when selecting the right combination of DataID extensions to solve a particular use case
- validation of newly created DataIDs or any other DataID document with a website or directly through a service endpoint
- automatic publication of DataID documents to dataset portals like Datahub[129] or Linghub[130]

Integrating DataID fully into the processes and tools defined by the ALIGNED project is another outstanding task.

[128] http://dacura.cs.tcd.ie

[129] https://datahub.io

[130] http://linghub.lider-project.eu

Glossary

Vocabulary On the Semantic Web, vocabularies define the concepts and relationships (also referred to as "terms") used to describe and represent an area of concern. Vocabularies are used to classify the terms that can be used in a particular application, characterize possible relationships, and define possible constraints on using those terms. In practice, vocabularies can be very complex (with several thousands of terms) or very simple (describing one or two concepts only)[75].

Ontology There is no clear division between what is referred to as "vocabularies" (cf. Vocabulary) and "ontologies". The trend is to use the word "ontology" for more complex, and possibly quite formal collection of terms, whereas "vocabulary" is used when such strict formalism is not necessarily used or only in a very loose sense. Vocabularies are the basic building blocks for inference techniques on the Semantic Web[75].

Application Profile a set of metadata elements defined for a particular application or other limited purposes, often based on a broader schema (like an ontology). Such a profile is either constructed under a closed world assumption or provides many restrictions to create a view

go on

AP-NISO Profiles are subsets of a scheme that are implemented by a particular interest group. Profiles can constrain the number of elements that will be used, refine element definitions to describe the specific types of resources more accurately, and specify values that an element can take.

Extension An extension is the addition of elements to an already developed scheme to support the description of an information resource of a particular type or subject or to meet the needs of a particular interest group. Extensions increase the number of elements.

Entity–relationship model describes inter-related things of interest in a specific domain of knowledge. An ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.

References

[1] Fadi Maali et al. "Enabling Interoperability of Government Data Catalogues." In: *EGOV*. Ed. by Maria Wimmer et al. LNCS. Springer, 2010.

[2] Keith Alexander and Michael Hausenblas. "Describing linked datasets - on the design and usage of void, the ?vocabulary of interlinked datasets". In: *In Linked Data on the Web Workshop (LDOW 09), in conjunction with 18th International World Wide Web Conference (WWW 09*. 2009.

[3] Phil Archer and Keith Jeffery. *Smart Descriptions & Smarter Vocabularies Report*. W3C Workshop report. URL: `https://www.w3.org/2016/11/sdsvoc/report.html`.

[4] Sören Auer et al. "Managing the Life-Cycle of Linked Data with the LOD2 Stack." In: *International Semantic Web Conference (2)*. Vol. 7650. Lecture Notes in Computer Science. Springer, 2012, pp. 1–16. URL: `http://dblp.uni-trier.de/db/conf/semweb/iswc2012-2.html#AuerBDEHILMMNSTW12`.

[5] Claire C Austin et al. *Key components of data publishing: Using current best practices to develop a reference model for data publishing*. Dec. 2015. DOI: `10.5281/zenodo.34542`. URL: `https://doi.org/10.5281/zenodo.34542`.

[6] Jon Barwise. "An Introduction to First-Order Logic". In: *HANDBOOK OF MATHEMATICAL LOGIC*. Elsevier BV, 1977, pp. 5–46. DOI: `10.1016/s0049-237x(08)71097-8`. URL: `http://dx.doi.org/10.1016/S0049-237X(08)71097-8`.

[7] Tim Berners-Lee. "Linked Data - Design Issues". In: *W3C* 09/20 (2006).

[8] Tim Berners-Lee, James Hendler, and Ora Lassila. "The Semantic Web A new form of Web content that is meaningful to computers will unleash a revolution of new possibilities". In: (2001). URL: `http://www.sciam.com/article.cfm?id=the-semantic-web%5C&%5C#38;print=true`.

[9] C. Bizer, T. Heath, and T. Berners-Lee. "Linked data - the story so far". In: *Int. J. Semantic Web Inf. Syst.* 5.3 (2009), 1?22.

[10] F. Bodendorf. *Daten- und Wissensmanagement*. Springer-Lehrbuch. Springer, 2003. ISBN: 9783540001027. URL: `https://books.google.de/books?id=f1wjPAAACAAJ`.

[11] Christoph Böhm, Johannes Lorey, and Felix Naumann. "Creating voiD descriptions for Web-scale data". In: *J. Web Sem.* 9.3 (2011), pp. 339–345. DOI: 10.1016/j.websem.2011.06.001. URL: http://dx.doi.org/10.1016/j.websem.2011.06.001.

[12] Dan Brickley and Libby Miller. *FOAF Vocabulary Specification 0.99*. Specification. http://xmlns.com/foaf/spec/20140114.html. Aug. 2014. URL: http://xmlns.com/foaf/spec/20140114.html.

[13] Daan Broeder et al. "A Data Category Registry- and Component-based Metadata Framework". In: *Proceedings of LREC 2010*. European Language Resources Association, 2010. ISBN: 2-9517408-6-7.

[14] Martin Brümmer et al. "DataID: Towards Semantically Rich Metadata for Complex Datasets". In: *Proceedings of the 10th International Conference on Semantic Systems*. SEM '14. Leipzig, Germany: ACM, 2014, pp. 84–91.

[15] Newton Calegari, Caroline Burle, and Bernadette Farias Loscio. *DWBP Implementation Report*. W3C Implementation Report. W3C, Dec. 2016. URL: https://www.w3.org/2013/dwbp/wiki/BP_Implementation_Report.

[16] SEMANTIC WEB HEALTH CARE and LIFE SCIENCES (HCLS) INTEREST GROUP. *HCLS mission statement*. 2016. URL: https://www.w3.org/blog/hcls/ (visited on 12/15/2016).

[17] Lois Mai Chan. "Metadata Interoperability: A Study of Methodology". In: *Proceedings of The 3rd China-United States Library Conference. 22-25 March. Shanghai, China*. 2005.

[18] Richard Cyganiak et al. *The RDF Data Cube Vocabulary*. W3C Recommendation. URL: https://www.w3.org/TR/vocab-data-cube/.

[19] Michel Dumontier et al. "The health care and life sciences community profile for dataset descriptions". In: *PeerJ* (2016). DOI: 10.7717/peerj.2331. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4991880/.

[20] Matej Durco and Menzo Windhouwer. "From CLARIN Component Metadata to Linked Open Data". In: *LDL 2014, LREC Workshop*. 2014.

[21] Erik Duval et al. "Metadata principles and practicalities". In: *D-lib Magazine* 8.4 (Apr. 2002), pp. 1–16. URL: https://lirias.kuleuven.be/handle/123456789/133275.

[22] Ramez Elmasri and Shamkant B. Navathe. *Fundamentals of Database Systems (5th Edition)*. Addison Wesley, Mar. 2006. ISBN: 0321369572. URL: http://www.amazon.ca/exec/obidos/redirect?tag=citeulike04-20%7B%5C&%7Dpath=ASIN/0321369572.

[23] Kevin Feeney. "The Dacura Data Curation System". In: *Computational History and Data-Driven Humanities - Second IFIP 12.7 International Workshop, CHDDH 2016, Dublin, Ireland, May 25, 2016, Revised Selected Papers*. 2016, pp. 15–20. DOI: `10.1007/978-3-319-46224-0_2`. URL: `http://dx.doi.org/10.1007/978-3-319-46224-0_2`.

[24] Kevin Feeney, Gavin Mendel-Gleason, and Rob Brennan. "Linked data schemata: fixing unsound foundations". In: *Semantic Web Journal-Special Issue on Quality Management of Semantic Web Assets* (2016).

[25] Achille Fokoue et al. *OWL 2 Web Ontology Language Profiles (Second Edition)*. W3C Recommendation. http://www.w3.org/TR/2012/REC-owl2-profiles-20121211/. W3C, Dec. 2012. URL: `https://www.w3.org/TR/2012/REC-owl2-profiles-20121211/;print=true`.

[26] FORCE11. *Guiding Principles for Findable, Accessible, Interoperable and Reusable Data Publishing version b1.0*. 2016. URL: `https://www.force11.org/fairprinciples` (visited on 12/15/2016).

[27] Ned Freed and Nathaniel Borenstein. *RFC2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types*. 1996. URL: `http://tools.ietf.org/html/rfc2046`.

[28] Markus Freudenberg et al. "The Metadata Ecosystem of DataID". In: *Metadata and Semantics Research: 10th International Conference, MTSR 2016, Göttingen, Germany, November 22-25, 2016, Proceedings*. Ed. by Emmanouel Garoufallou et al. Cham: Springer International Publishing, 2016, pp. 317–332. ISBN: 978-3-319-49157-8. DOI: `10.1007/978-3-319-49157-8_28`. URL: `http://dx.doi.org/10.1007/978-3-319-49157-8_28`.

[29] Asunción Gómez-Pérez, Mariano Fernández-López, and Óscar Corcho. *Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web*. Advanced Information and Knowledge Processing. Springer, 2004. ISBN: 978-1-85233-551-9. DOI: `10.1007/b97353`. URL: `http://dx.doi.org/10.1007/b97353`.

[30] DCAT Application profile working group. *DCAT Application Profile for data portals in Europe Version 1.1*. 2016. URL: `https://joinup.ec.europa.eu/asset/dcat_application_profile/asset_release/dcat-ap-v11` (visited on 12/15/2016).

[31] Thomas R. Gruber. "Toward Principles for the Design of Ontologies Used for Knowledge Sharing". In: *Int. J. Hum.-Comput. Stud.* 43.5-6 (Dec. 1995), pp. 907–928. ISSN: 1071-5819. DOI: `10.1006/ijhc.1995.1081`. URL: `http://dx.doi.org/10.1006/ijhc.1995.1081`.

[32] Ramanathan Guha and Dan Brickley. *RDF Schema 1.1.* W3C Recommendation. http://www.w3.org/TR/2014/REC-rdf-schema-20140225/. W3C, Feb. 2014. URL: http://www.w3.org/TR/2014/REC-rdf-schema-20140225/;print=true.

[33] Bernhard Haslhofer and Wolfgang Klas. "A Survey of Techniques for Achieving Metadata Interoperability". In: *ACM Comput. Surv.* 42.2 (Mar. 2010), 7:1–7:37. ISSN: 0360-0300. DOI: 10.1145/1667062.1667064. URL: http://doi.acm.org/10.1145/1667062.1667064.

[34] Erhard Hinrichs and Steven Krauwer. "The CLARIN Research Infrastructure: Resources and Tools for e-Humanities Scholars". In: *Proceedings of LREC 2014*. European Language Resources Association (ELRA), 2014.

[35] Ian Horrocks et al. "OWL: a Description Logic Based Ontology Language for the Semantic Web". In: *The Description Logic Handbook: Theory, Implementation, and Applications (2nd Edition)*. Ed. by Franz Baader et al. Cambridge University Press, 2007. Chap. 14. URL: download/2003/HPMW07.pdf.

[36] Bernadette Hyland, Ghislain Auguste Atemezing, and Boris Villazón-Terrazas. *Best Practices for Publishing Linked Data*. W3C Note. http://www.w3.org/TR/2014/NOTE-ld-bp-20140109/. W3C, Jan. 2014.

[37] IFLA. *Functional Requirements for Bibliographic Records: Final Report*. K. G. Saur, 1998. URL: http://www.amazon.com/Functional-Requirements-Bibliographic-Records-Publications/dp/359811382X.

[38] Antoine Isaac and Riccardo Albertoni. *Data on the Web Best Practices: Data Quality Vocabulary*. W3C Note. https://www.w3.org/TR/2016/NOTE-vocab-dqv-20161215/. W3C, Dec. 2016.

[39] Keith G Jeffery and Anne Asserson. "Position Paper: Why CERIF?" In: 2016. URL: %5Curl%7Bhttps://www.w3.org/2016/11/sdsvoc/SDSVoc16_paper_15%7D.

[40] Keith Jeffery and Anne Asserson. "CERIF-CRIS for the European e-Infrastructure". In: *Data Science Journal* 9 (2010), CRIS1–CRIS6. DOI: 10.2481/dsj.CRIS1. URL: http://dx.doi.org/10.2481/dsj.CRIS1.

[41] C. Maria Keet. "The Use of Foundational Ontologies in Ontology Development: An Empirical Assessment". In: *Proceedings of the 8th Extended Semantic Web Conference on The Semantic Web: Research and Applications - Volume Part I*. ESWC'11. Heraklion, Crete, Greece: Springer-Verlag, 2011, pp. 321–335. ISBN: 978-3-642-21033-4. URL: http://dl.acm.org/citation.cfm?id=2008892.2008920.

[42] Tomás Knap et al. “UnifiedViews: An ETL Framework for Sustainable RDF Data Processing.” In: *ESWC (Satellite Events)*. Vol. 8798. Lecture Notes in Computer Science. Springer, 2014, pp. 379–383. ISBN: 978-3-319-11954-0. URL: http://dblp.uni-trier.de/db/conf/esws/eswc2014s.html#KnapKMSTV14.

[43] Dimitris Kontokostas et al. “Databugger: A Test-driven Framework for Debugging the Web of Data”. In: *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion*. WWW Companion ’14. Seoul, Korea: International World Wide Web Conferences Steering Committee, 2014, pp. 115–118. ISBN: 978-1-4503-2745-9. DOI: 10.1145/2567948.2577017. URL: http://jens-lehmann.org/files/2014/www_demo_databugger.pdf.

[44] Carl Lagoze. *Accommodating Simplicity and Complexity in Metadata: Lessons from the Dublin Core Experience*. Technical Report. 2000. URL: https://ecommons.cornell.edu/bitstream/handle/1813/5792/2000-1801.pdf.

[45] Markus Lanthaler, David Wood, and Richard Cyganiak. *RDF 1.1 Concepts and Abstract Syntax*. W3C Recommendation. http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/. W3C, Feb. 2014. URL: http://www.w3.org/TR/2014/REC-rdf11-concepts-20140225/;print=true.

[46] Jens Lehmann et al. “DBpedia - A Large-scale, Multilingual Knowledge Base Extracted from Wikipedia”. In: *Semantic Web Journal* 6.2 (2015), pp. 167–195. URL: http://jens-lehmann.org/files/2014/swj_dbpedia.pdf.

[47] SPDX Workgroup a Linux Foundation Project. *SPDX Specifications*. 2016. URL: https://spdx.org/specifications (visited on 12/15/2016).

[48] Bernadette Farias Loscio, Caroline Burle, and Newton Calegari. *Data on the Web Best Practices*. W3C Proposed Recommendation. W3C, Dec. 2016. URL: https://www.w3.org/TR/2016/PR-dwbp-20161215/.

[49] Fadi Maali, DERI, and NUI Galway. *Data Catalog Vocabulary (DCAT)*. W3C Recommendation. URL: https://www.w3.org/TR/vocab-dcat/.

[50] Shane McCarron et al. *RDFa Core 1.1 - Third Edition*. W3C Recommendation. http://www.w3.org/TR/2015/REC-rdfa-core-20150317/. W3C, Mar. 2015.

[51] John P. McCrae et al. “One ontology to bind them all: The META-SHARE OWL ontology for the interoperability of linguistic datasets on the Web”. In: *Proc. of 12th Extended Semantic Web Conference (ESWC 2015) Satellite Events, Portorož, Slovenia*. Vol. 9341. June 2015, pp. 271–282.

[52] Deborah McGuinness, Timothy Lebo, and Satya Sahoo. *The PROV Ontology*. W3C Recommendation. URL: http://www.w3.org/TR/prov-o/.

[53] Mo McRoberts and Victor Rodriguez Doncel. *ODRL Version 2.1 Ontology*. W3C Community Group Specification. http://www.w3.org/ns/odrl/2/ODRL21. W3C, Mar. 2015. URL: https://www.w3.org/ns/odrl/2/.

[54] Gerard de Melo. "Lexvo.org: Language-Related Information for the Linguistic Linked Data Cloud". In: *Semantic Web* 6.4 (Aug. 2015), pp. 393–400.

[55] Luc Moreau et al. "The Rationale of PROV". In: *Web Semant.* 35.P4 (Dec. 2015), pp. 235–257. ISSN: 1570-8268. DOI: 10.1016/j.websem.2015.04.001. URL: http://dx.doi.org/10.1016/j.websem.2015.04.001.

[56] Luc Moreau and Paolo Missier. *PROV-DM: The PROV Data Model*. W3C Recommendation. http://www.w3.org/TR/2013/REC-prov-dm-20130430/. W3C, Apr. 2013.

[57] Sebastian Neumaier, Jürgen Umbrich, and Axel Polleres. "Challenges of mapping current CKAN metadata to DCAT". In: *W3C Workshop on Data and Services Integration*. Amsterdam, the Netherlands, Nov. 2016. URL: https://www.w3.org/2016/11/sdsvoc/SDSVoc16_paper_16.

[58] Henrik Frystyk Nielsen. *Interoperability and Evolvability*. https://www.w3.org/Protocols/Design/Interevol.html.

[59] NISO. *Understanding metadata*. ISBN 1-880124-62-9. National Information Standards Organization. 2004. URL: http://www.niso.org/standards/resources/UnderstandingMetadata.pdf.

[60] Kevin Palmer and Pete Richardson. "Learning Object Reusability - Motivation, Production and Use". In: *11th International Conference of the Association for Learning Technology (ALT). University of Exeter, Devon, England, 14-16 September 2004.* 2004.

[61] Heinz Pampel et al. "Making Research Data Repositories Visible: The re3data.org Registry". In: *PLoS ONE* 8(11).e78080 (2013).

[62] Silvio Peroni et al. "DataCite2RDF: Mapping DataCite Metadata Schema 3.1 Terms to RDF". In: (Feb. 2016). DOI: 10.6084/m9.figshare.2075356.v1. URL: https://figshare.com/articles/DataCite2RDF_Mapping_DataCite_Metadata_Schema_3_1_Terms_to_RDF/2075356.

[63] Dave Reynolds. *The Organization Ontology*. W3C Recommendation. http://www.w3.org/TR/2014/REC-vocab-org-20140116/. W3C, Jan. 2014.

[64] Jennifer Rowley. "The Wisdom Hierarchy: Representations of the DIKW Hierarchy". In: *J. Inf. Sci.* 33.2 (Apr. 2007), pp. 163–180. ISSN: 0165-5515. DOI: 10.1177/0165551506070706. URL: http://dx.doi.org/10.1177/0165551506070706.

[65] Jessika Rücknagel et al. "Metadata Schema for the Description of Research Data Repositories". In: GFZ Germans Research Center for Geosciences.

[66] Sebastian Rudolph et al. *OWL 2 Web Ontology Language Primer (Second Edition)*. W3C Recommendation. http://www.w3.org/TR/2012/REC-owl2-primer-20121211/. W3C, Dec. 2012. URL: https://www.w3.org/TR/owl2-primer/;print=true.

[67] Claude Shannon. "A Mathematical Theory of Communication". In: *Bell System Technical Journal* 27 (July 1948), pp. 379–423, 623–656. URL: http://cm.bell-labs.com/cm/ms/what/shannonday/shannon1948.pdf.

[68] *Shapes Constraint Language (SHACL)*. W3C Working Draft. https://www.w3.org/TR/2016/WD-shacl-20160814/. W3C, Aug. 2016.

[69] Monika Solanki et al. "Enabling Combined Software and Data Engineering at Web-Scale: The ALIGNED Suite of Ontologies". In: *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part II*. 2016, pp. 195–203. DOI: 10.1007/978-3-319-46547-0_21. URL: http://dx.doi.org/10.1007/978-3-319-46547-0_21.

[70] Seema Sundara, Richard Cyganiak, and Souripriya Das. *R2RML: RDB to RDF Mapping Language*. W3C Recommendation. http://www.w3.org/TR/2012/REC-r2rml-20120927/. W3C, Sept. 2012.

[71] Arlene G. Taylor. *The organization of information*. English. Libraries Unlimited Englewood, Colo, 1999, xx, 280 p. ISBN: 1563084988 1563084937.

[72] Henry Thompson et al. *W3C XML Schema Definition Language 1.1 (XSD) Part 1: Structures*. W3C Recommendation. http://www.w3.org/TR/2012/REC-xmlschema11-1-20120405/. W3C, Apr. 2012.

[73] Andreas Tolk. "What Comes After the Semantic Web - PADS Implications for the Dynamic Web". In: *Proceedings of the 20th Workshop on Principles of Advanced and Distributed Simulation*. PADS '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 55–. ISBN: 0-7695-2587-3. DOI: 10.1109/PADS.2006.39. URL: http://dx.doi.org/10.1109/PADS.2006.39.

[74] Boris Villazón-Terrazas et al. "Methodological Guidelines for Publishing Government Linked Data Linking Government Data". In: *Linking Government Data*. Ed. by David Wood. New York, NY: Springer New York, 2011. Chap. 2, pp. 27–49. ISBN: 978-1-4614-1766-8. DOI: `10.1007/978-1-4614-1767-5\_2`. URL: `http://dx.doi.org/10.1007/978-1-4614-1767-5%5C_2`.

[75] W3C. *VOCABULARIES*. `https://www.w3.org/standards/semanticweb/ontology`. [Online; accessed 01-December-2016]. 2015.

[76] Mark D. Wilkinson et al. "The FAIR Guiding Principles for scientific data management and stewardship". In: *Scientific Data* 3 (Mar. 2016), pp. 160018+. ISSN: 2052-4463. DOI: `10.1038/sdata.2016.18`. URL: `http://dx.doi.org/10.1038/sdata.2016.18`.

[77] Gregory Williams. *SPARQL 1.1 Service Description*. W3C Recommendation. http://www.w3.org/TR/2013/REC-sparql11-service-description-20130321/. W3C, Mar. 2013.

[78] Brecht Wyns et al. "DCAT Application Profile for Data Portals in Europe". In: 2016. URL: `%5Curl%7Bhttps://www.w3.org/2016/11/sdsvoc/SDSVoc16_paper_30%7D`.

[79] Chaim Zins. "Conceptual Approaches for Defining Data, Information, and Knowledge: Research Articles". In: *J. Am. Soc. Inf. Sci. Technol.* 58.4 (Feb. 2007), pp. 479–493. ISSN: 1532-2882. DOI: `10.1002/asi.v58:4`. URL: `http://dx.doi.org/10.1002/asi.v58:4`.

Appendix I

DataID core specification This work will not include the whole specification of `DataID core`. Instead I provide the revision number at the time of writing of the ontology specification file (dataid.ttl):

bbdacd50dd6389bc435a576a03753c7101a6ac02[131]

The latest version of this specification is always available under the namespace URI of `DataID`:

`http://dataid.dbpedia.org/ns/core.ttl`

[131] `https://github.com/dbpedia/DataId-Ontology/commit/bbdacd50dd6389bc435a576a03753c7101a6ac02`

Appendix II

```
@prefix dataid: <http://dataid.dbpedia.org/ns/core#> .
@prefix dataid-ld: <http://dataid.dbpedia.org/ns/ld#> .
@prefix dataid-mt: <http://dataid.dbpedia.org/ns/mt#> .
@prefix dcat:  <http://www.w3.org/ns/dcat#> .
@prefix datacite: <http://purl.org/spar/datacite/> .
@prefix void:  <http://rdfs.org/ns/void#> .
@prefix spdx:  <http://spdx.org/rdf/terms#> .
@prefix owl:   <http://www.w3.org/2002/07/owl#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix prov:  <http://www.w3.org/ns/prov#> .
@prefix foaf:  <http://xmlns.com/foaf/0.1/> .
@prefix dct:    <http://purl.org/dc/terms/> .
@prefix sd:    <http://www.w3.org/ns/sparql-service-description#> .
@base <http://downloads.dbpedia.org/2015-10/core-i18n/ar/2015-10_> .

<dataid_ar.ttl>
        a                          dataid:DataId ;
        dataid:associatedAgent     <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.
             dbpedia.org/dbpedia-association/persons/Freudenberg> ;
        dataid:inCatalog           <http://downloads.dbpedia.org/2015-10/2015-10_dataid_catalog.ttl>
             ;
        dataid:latestVersion       <http://downloads.dbpedia.org/2016-04/core-i18n/ar/2016-04
             _dataid_ar.ttl> ;
        dataid:nextVersion         <http://downloads.dbpedia.org/2016-04/core-i18n/ar/2016-04
             _dataid_ar.ttl> ;
        dataid:previousVersion     <http://downloads.dbpedia.org/2015-04/core-i18n/ar/2015-04
             _dataid_ar.ttl> ;
        dataid:underAuthorization  <dataid_ar.ttl?auth=maintainerAuthorization> , <dataid_ar.ttl?auth
             =creatorAuthorization> ;
        dct:hasVersion              <dataid_ar.ttl?version=1.0.0> ;
        dct:issued                  "2016-08-02"^^xsd:date ;
        dct:modified                "2016-10-13"^^xsd:date ;
        dct:publisher               <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                   "DataID metadata for the Arabic DBpedia"@en ;
        foaf:primaryTopic          <dataid_ar.ttl?set=maindataset> .

#### Agents & Authorizations ####

<http://wiki.dbpedia.org/dbpedia-association/persons/Freudenberg>
        a                        dataid:Agent ;
        dataid:hasAuthorization  <dataid_ar.ttl?auth=maintainerAuthorization> ;
        dataid:identifier        <http://www.researcherid.com/rid/L-2180-2016> ;
        foaf:mbox                "freudenberg@informatik.uni-leipzig.de" ;
        foaf:name                "Markus Freudenberg" .

<dataid_ar.ttl?auth=maintainerAuthorization>
        a                          dataid:Authorization ;
        dataid:authorityAgentRole  dataid:Maintainer ;
        dataid:authorizedAgent     <http://wiki.dbpedia.org/dbpedia-association/persons/Freudenberg>
             ;
        dataid:authorizedFor       <dataid_ar.ttl> ;
        dataid:isInheritable       true .
```

```
<http://www.researcherid.com/rid/L-2180-2016>
        a                              dataid:Identifier ;
        dataid:literal                  "L-2180-2016" ;
        dct:issued                      "2016-08-01"^^xsd:date ;
        dct:references                  <http://www.researcherid.com/rid/L-2180-2016> ;
        datacite:usesIdentifierScheme  datacite:researcherid .

<http://wiki.dbpedia.org/dbpedia-association>
        a                        dataid:Agent ;
        dataid:hasAuthorization  <dataid_ar.ttl?auth=creatorAuthorization> ;
        foaf:homepage            <http://dbpedia.org> ;
        foaf:mbox                "dbpedia@infai.org" ;
        foaf:name                "DBpedia Association" .

<dataid_ar.ttl?auth=creatorAuthorization>
        a                          dataid:Authorization ;
        dataid:authorityAgentRole  dataid:Creator ;
        dataid:authorizedAgent     <http://wiki.dbpedia.org/dbpedia-association> ;
        dataid:authorizedFor       <dataid_ar.ttl> ;
        dataid:isInheritable       true .

<https://wikimediafoundation.org>
        a                        dataid:Agent ;
        dataid:hasAuthorization  <http://dbpedia.org/dataset/pages_articles?lang=ar&dbpv=2016-04&file
             =pages_articles_ar.xml.bz2&auth=publisherAuthorization> , <http://dbpedia.org/dataset/
             pages_articles?lang=ar&dbpv=2016-04&auth=publisherAuthorization> ;
        foaf:mbox                "info@wikimedia.org" ;
        foaf:name                "Wikimedia Foundation, Inc." .

<dataid_ar.ttl?auth=publisherAuthorization>
        a                          dataid:Authorization ;
        dataid:authorityAgentRole  dataid:Creator, dataid:Publisher ;
        dataid:authorizedAgent     <https://wikimediafoundation.org> ;
        dataid:authorizedFor       <dataid_ar.ttl?set=pages_articles> ;
        dataid:isInheritable       true .

########## Main Dataset ##########

<dataid_ar.ttl?set=maindataset>
        a                       dataid:Superset ;
        dataid:associatedAgent  <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.dbpedia.
             org/dbpedia-association/persons/Freudenberg>  ;
        dataid:growth               <dataid_ar.ttl?stmt=growth> ;
        dataid:openness             <dataid_ar.ttl?stmt=openness> ;
        dataid:reuseAndIntegration  <dataid_ar.ttl?stmt=reuseAndIntegration> ;
        dataid:similarData          <dataid_ar.ttl?stmt=similarData> ;
        dataid:usefulness           <dataid_ar.ttl?stmt=usefulness> ;
        dct:description          """DBpedia is a crowd-sourced community effort to extract structured
              information from Wikipedia and make this information available on the Web. DBpedia
             allows you to ask sophisticated queries against Wikipedia, and to link the different data
              sets on the Web to Wikipedia data. We hope that this work will make it easier for the
             huge amount of information in Wikipedia to be used in some new interesting ways.
             Furthermore, it might inspire new mechanisms for navigating, linking, and improving the
             encyclopedia itself."""@en ;
        dct:hasVersion           <dataid_ar.ttl?version=1.0.0> ;
        dct:issued               "2016-07-02"^^xsd:date ;
        dct:language             <http://lexvo.org/id/iso639-3/ara> ;
        dct:license              <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified             "2016-08-01"^^xsd:date ;
        dct:publisher            <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:rights               <dataid_ar.ttl?rights=dbpedia-rights> ;
        dct:title                "DBpedia root dataset for Arabic, version 2015-10"@en ;
        void:subset             <dataid_ar.ttl?set=long_abstracts_en_uris>, <dataid_ar.ttl?set=
             interlanguage_links> ;
        void:vocabulary         <http://downloads.dbpedia.org/2015-04/dbpedia_2015-10.owl> ;
        dcat:keyword            "maindataset"@en , "DBpedia"@en ;
        dcat:landingPage        <http://dbpedia.org/> ;
        foaf:isPrimaryTopicOf   <dataid_ar.ttl> ;
        foaf:page               <http://wiki.dbpedia.org/Downloads2015-10> .

############ Datasets #############

<dataid_ar.ttl?set=interlanguage_links>
        a                       dataid:Dataset, dataid-ld:LinkedDataDataset ;
        rdfs:label              "interlanguage links"@en ;
        dataid:associatedAgent  <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.dbpedia.
             org/dbpedia-association/persons/Freudenberg> ;
        dct:description          "Dataset linking a DBpedia resource to the same resource in other
             languages and in Wikidata."@en ;
        dct:hasVersion           <dataid_ar.ttl?version=1.0> ;
        dct:isPartOf             <dataid_ar.ttl?set=maindataset> ;
        dct:issued               "2016-07-02"^^xsd:date ;
        dct:language             <http://lexvo.org/id/iso639-3/ara> ;
        dct:license              <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified             "2016-08-02"^^xsd:date ;
        dct:publisher            <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                "interlanguage links"@en ;
        void:rootResource       <dataid_ar.ttl?set=maindataset> ;
        void:triples            7480764 ;
        dcat:distribution       <dataid_ar.ttl?file=interlanguage_links_ar.ttl.bz2> , <dataid_ar.ttl?
             file=interlanguage_links_ar.tql.bz2> ;
        dcat:keyword            "DBpedia"@en , "interlanguage_links"@en ;
        dcat:landingPage        <http://dbpedia.org/> ;
        sd:defaultGraph         <http://ar.dbpedia.org> ;
        foaf:page               <http://wiki.dbpedia.org/Downloads2015-10> .

<dataid_ar.ttl?set=long_abstracts_en_uris>
        a                       dataid:Dataset, dataid-ld:LinkedDataDataset ;
        rdfs:label              "long abstracts en uris"@en ;
        dataid:associatedAgent  <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.dbpedia.
             org/dbpedia-association/persons/Freudenberg> ;
        dataid:qualifiedDatasetRelation   <dataid_ar.ttl?relation=source&target=pages_articles> ;
        dataid:relatedDataset   <dataid_ar.ttl?set=pages_articles> ;
        dct:description          "Full abstracts of Wikipedia articles, usually the first section.
             Normalized resources matching English DBpedia."@en ;
        dct:hasVersion           <dataid_ar.ttl?version=1.0.0> ;
        dct:isPartOf             <dataid_ar.ttl?set=maindataset> ;
        dct:issued               "2016-07-02"^^xsd:date ;
        dct:language             <http://lexvo.org/id/iso639-3/ara> ;
        dct:license              <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified             "2016-08-02"^^xsd:date ;
        dct:publisher            <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                "long abstracts en uris"@en ;
        void:rootResource       <dataid_ar.ttl?set=maindataset> ;
        void:triples            232801 ;
        void:sparqlEndpoint      <http://dbpedia.org/sparql> ;
        dcat:distribution       <dataid_ar.ttl?sparql=DBpediaSparqlEndpoint> , <dataid_ar.ttl?file=
             long_abstracts_en_uris_ar.ttl.bz2> , <dataid_ar.ttl?file=long_abstracts_en_uris_ar.tql.
             bz2> ;
        dcat:keyword            "long_abstracts_en_uris"@en , "DBpedia"@en ;
        dcat:landingPage        <http://dbpedia.org/> ;
        sd:defaultGraph         <http://ar.dbpedia.org> ;
        foaf:page               <http://wiki.dbpedia.org/Downloads2015-10> .

<dataid_ar.ttl?set=pages_articles>
```

```
        a                           dataid:Dataset ;
        rdfs:label                  "Wikipedia XML source dump file"@en ;
        dataid:associatedAgent      <https://wikimediafoundation.org> ;
        dataid:needsSpecialAuthorization  <dataid_ar.ttl?auth=publisherAuthorization> ;
        dct:description              "The Wikipedia dump file, which is the source for all other
             extracted datasets."@en ;
        dct:hasVersion               "20160305" ;
        dct:issued                   "2016-03-05"^^xsd:date ;
        dct:language                 <http://lexvo.org/id/iso639-3/ara> ;
        dct:license                  <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:publisher                <https://wikimediafoundation.org> ;
        dct:title                    "Wikipedia XML source dump file"@en ;
        dcat:distribution           <http://dbpedia.org/dataset/pages_articles?lang=ar&dbpv=2016-04&
             file=pages_articles_ar.xml.bz2> ;
        dcat:keyword                "Wikipedia"@en , "XML dump file"@en ;
        dcat:landingPage            <https://meta.wikimedia.org/wiki/Data_dumps> .

########## Distributions ###########

<dataid_ar.ttl?file=interlanguage_links_ar.ttl.bz2>
        a                             dataid:SingleFile ;
        rdfs:label                    "interlanguage_links_ar.ttl.bz2" ;
        dataid:associatedAgent        <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.
             dbpedia.org/dbpedia-association/persons/Freudenberg> ;
        dataid:checksum               <dataid_ar.ttl?file=interlanguage_links_ar.ttl.bz2&checksum=md5>
              ;
        dataid:isDistributionOf       <dataid_ar.ttl?set=interlanguage_links> ;
        dataid:preview                <http://downloads.dbpedia.org/preview.php?file=2015-10_sl_core-
             i18n_sl_ar_sl_interlanguage_links_ar.ttl.bz2> ;
        dataid:uncompressedByteSize  1184687767 ;
        dct:description                "Dataset linking a DBpedia resource to the same resource in
             other languages and in Wikidata."@en ;
        dct:hasVersion                 <dataid_ar.ttl?version=1.0> ;
        dct:issued                    "2016-07-02"^^xsd:date ;
        dct:license                    <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified                   "2016-08-02"^^xsd:date ;
        dct:publisher                  <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                      "interlanguage links"@en ;
        dcat:byteSize                 62761863 ;
        dcat:downloadURL              <http://downloads.dbpedia.org/2015-10/core-i18n/core-i18n/ar/
             interlanguage_links_ar.ttl.bz2> ;
        dcat:mediaType                dataid-mt:MediaType_turtle_x-bzip2 .

<dataid_ar.ttl?file=interlanguage_links_ar.tql.bz2>
        a                             dataid:SingleFile ;
        rdfs:label                    "interlanguage_links_ar.tql.bz2" ;
        dataid:associatedAgent        <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.
             dbpedia.org/dbpedia-association/persons/Freudenberg> ;
        dataid:checksum               <dataid_ar.ttl?file=interlanguage_links_ar.tql.bz2&checksum=md5>
              ;
        dataid:isDistributionOf       <dataid_ar.ttl?set=interlanguage_links> ;
        dataid:preview                <http://downloads.dbpedia.org/preview.php?file=2015-10_sl_core-
             i18n_sl_ar_sl_interlanguage_links_ar.tql.bz2> ;
        dataid:uncompressedByteSize  1598056873 ;
        dct:description                "Dataset linking a DBpedia resource to the same resource in
             other languages and in Wikidata."@en ;
        dct:hasVersion                 <dataid_ar.ttl?version=1.0> ;
        dct:issued                    "2016-07-02"^^xsd:date ;
        dct:license                    <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified                   "2016-08-02"^^xsd:date ;
        dct:publisher                  <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                      "interlanguage links"@en ;
        dcat:byteSize                 71946917 ;
        dcat:downloadURL             <http://downloads.dbpedia.org/2015-10/core-i18n/core-i18n/ar/
             interlanguage_links_ar.tql.bz2> ;
        dcat:mediaType               dataid-mt:MediaType_n-quads_x-bzip2 .

<dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2>
        a                            dataid:SingleFile ;
        rdfs:label                   "long_abstracts_en_uris_ar.ttl.bz2" ;
        dataid:associatedAgent       <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.
             dbpedia.org/dbpedia-association/persons/Freudenberg> ;
        dataid:checksum              <dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2&checksum=
             md5> ;
        dataid:isDistributionOf      <dataid_ar.ttl?set=long_abstracts_en_uris> ;
        dataid:preview               <http://downloads.dbpedia.org/preview.php?file=2015-10_sl_core-
             i18n_sl_ar_sl_long_abstracts_en_uris_ar.ttl.bz2> ;
        dataid:uncompressedByteSize  186573907 ;
        dct:description               "Full abstracts of Wikipedia articles, usually the first
             section. Normalized resources matching English DBpedia."@en ;
        dct:hasVersion                <dataid_ar.ttl?version=1.0> ;
        dct:issued                   "2016-07-02"^^xsd:date ;
        dct:license                   <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified                  "2016-08-02"^^xsd:date ;
        dct:publisher                 <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                     "long abstracts en uris"@en ;
        dcat:byteSize                33428372 ;
        dcat:downloadURL             <http://downloads.dbpedia.org/2015-10/core-i18n/core-i18n/ar/
             long_abstracts_en_uris_ar.ttl.bz2> ;
        dcat:mediaType               dataid-mt:MediaType_turtle_x-bzip2 .

<dataid_ar.ttl?file=long_abstracts_en_uris_ar.tql.bz2>
        a                            dataid:SingleFile ;
        rdfs:label                   "long_abstracts_en_uris_ar.tql.bz2" ;
        dataid:associatedAgent       <http://wiki.dbpedia.org/dbpedia-association> , <http://wiki.
             dbpedia.org/dbpedia-association/persons/Freudenberg> ;
        dataid:checksum              <dataid_ar.ttl?file=long_abstracts_en_uris_ar.tql.bz2&checksum=
             md5> ;
        dataid:isDistributionOf      <dataid_ar.ttl?set=long_abstracts_en_uris> ;
        dataid:preview               <http://downloads.dbpedia.org/preview.php?file=2015-10_sl_core-
             i18n_sl_ar_sl_long_abstracts_en_uris_ar.tql.bz2> ;
        dataid:uncompressedByteSize  204174726 ;
        dct:description               "Full abstracts of Wikipedia articles, usually the first
             section. Normalized resources matching English DBpedia."@en ;
        dct:hasVersion                <dataid_ar.ttl?version=1.0> ;
        dct:issued                   "2016-07-02"^^xsd:date ;
        dct:license                   <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified                  "2016-08-02"^^xsd:date ;
        dct:publisher                 <http://wiki.dbpedia.org/dbpedia-association> ;
        dct:title                     "long abstracts en uris"@en ;
        dcat:byteSize                36026709 ;
        dcat:downloadURL             <http://downloads.dbpedia.org/2015-10/core-i18n/core-i18n/ar/
             long_abstracts_en_uris_ar.tql.bz2> ;
        dcat:mediaType               dataid-mt:MediaType_n-quads_x-bzip2 .

<dataid_ar.ttl?sparql=DBpediaSparqlEndpoint>
        a                        dataid-ld:SparqlEndpoint ;
        rdfs:label               "The official DBpedia sparql endpoint"@en ;
        dataid:accessProcedure   <dataid_ar.ttl?stmt=sparqlaccproc> ;
        dataid:associatedAgent   <http://support.openlinksw.com/> ;
        dataid:isDistributionOf  <dataid_ar.ttl?set=long_abstracts_en_uris> ;
        dct:description           "The official sparql endpoint of DBpedia, hosted graciously by
             OpenLink Software (http://virtuoso.openlinksw.com/), containing all datasets of the /core
              directory."@en ;
        dct:hasVersion            <dataid_ar.ttl?version=1.0> ;
        dct:issued                "2016-07-02"^^xsd:date ;
        dct:license               <http://purl.oclc.org/NET/rdflicense/cc-by-sa3.0> ;
        dct:modified              "2016-08-02"^^xsd:date ;
        dct:title                 "The official DBpedia sparql endpoint"@en ;
        sd:endpoint                                <http://dbpedia.org/sparql> ;
        sd:supportedLanguage     sd:SPARQL11Query ;
        sd:resultFormat                  <http://www.w3.org/ns/formats/RDF_XML>, <http://www.w3.org/
             ns/formats/Turtle> ;
        dcat:accessURL           <http://dbpedia.org/sparql> ;
        dcat:mediaType           <http://dataid.dbpedia.org/ns/mt#MediaType_sparql-results+xml> .

########### Relations ############

<dataid_ar.ttl?relation=source&target=pages_articles>
        a                           dataid:DatasetRelationship ;
        dataid:datasetRelationRole  dataid:SourceRole ;
        dataid:qualifiedRelationOf  <dataid_ar.ttl?set=long_abstracts_en_uris> ;
        dataid:qualifiedRelationTo  <dataid_ar.ttl?set=pages_articles> .

########### Checksums ############

<dataid_ar.ttl?file=interlanguage_links_ar.ttl.bz2&checksum=md5>
        a                   spdx:Checksum ;
        spdx:algorithm      spdx:checksumAlgorithm_md5 ;
        spdx:checksumValue  "b1a6885fba528b08c53b0ad800a94f7a"^^xsd:hexBinary .


<dataid_ar.ttl?file=interlanguage_links_ar.tql.bz2&checksum=md5>
        a                   spdx:Checksum ;
        spdx:algorithm      spdx:checksumAlgorithm_md5 ;
        spdx:checksumValue  "d34de153e77570f118b7425e5cf1ca0b"^^xsd:hexBinary .

<dataid_ar.ttl?file=long_abstracts_en_uris_ar.ttl.bz2&checksum=md5>
        a                   spdx:Checksum ;
        spdx:algorithm      spdx:checksumAlgorithm_md5 ;
        spdx:checksumValue  "2503179cd96452d33becd1e974d6a163"^^xsd:hexBinary .

<dataid_ar.ttl?file=long_abstracts_en_uris_ar.tql.bz2&checksum=md5>
        a                   spdx:Checksum ;
        spdx:algorithm      spdx:checksumAlgorithm_md5 ;
        spdx:checksumValue  "ffdf034c2477d81b5aaeced0312984d4"^^xsd:hexBinary .

########### Statements ###########

<dataid_ar.ttl?rights=dbpedia-rights>
        a                 dataid:SimpleStatement ;
        dataid:literal  """DBpedia is derived from Wikipedia and is distributed under the same
             licensing terms as Wikipedia itself. As Wikipedia has moved to dual-licensing, we also
             dual-license DBpedia starting with release 3.4. Data comprising DBpedia release 3.4 and
             subsequent releases is licensed under the terms of the Creative Commons Attribution-
             ShareAlike 3.0 license and the GNU Free Documentation License. Data comprising DBpedia
             releases up to and including release 3.3 is licensed only under the terms of the GNU Free
              Documentation License."""@en .

<dataid_ar.ttl?version=1.0.0>
        a                 dataid:SimpleStatement ;
        dataid:literal  "1.0.0" .

<dataid_ar.ttl?stmt=sparqlaccproc>
        a                 dataid:SimpleStatement ;
        dct:references    <https://www.w3.org/TR/sparql11-overview/> ;
        dataid:literal  "An endpoint for sparql queries: provide valid queries." .

<dataid_ar.ttl?stmt=openness>
```

```
        a                 dataid:SimpleStatement ;
        dataid:statement  "DBpedia is an open dataset, licensed under CC-BY-SA 3.0."@en .

<dataid_ar.ttl?stmt=growth>
        a                 dataid:SimpleStatement ;
        dataid:statement  "DBpedia is an ongoing open-source project. Goal of the project is the
             extraction of the Wikipedia, as complete as possible. Currently, 126 languages are being
             extracted. In the future, DBpedia will try to increase its importance as the center of
             the LOD cloud by adding further external datasets"@en .

<dataid_ar.ttl?stmt=similarData>
        a                 dataid:SimpleStatement ;
        dataid:statement  "Similar data can be found in datasets like Freebase (https://freebase.com)
             , Wikidata (https://www.wikidata.org), Yago (http://www.mpi-inf.mpg.de/departments/
             databases-and-information-systems/research/yago-naga/yago//) or OpenCyc (http://opencyc.
             org)."@en .

<dataid_ar.ttl?stmt=usefulness>
        a                 dataid:SimpleStatement ;
        dataid:statement  "DBpedia is a useful resource for interlinking general datasets with
             encyclopedic knowledge. Users profitting from DBpedia are open data developers, SMEs and
             researchers in data science and NLP"@en .

<dataid_ar.ttl?stmt=reuseAndIntegration>
        a                 dataid:SimpleStatement ;
        dataid:statement  "DBpedia data can be integrated into other datasets and reused for data
             enrichment or mashup purposes"@en .

########### MediaTypes ###########

<http://dataid.dbpedia.org/ns/mt#MediaType_sparql-results+xml>
        a                    dataid:MediaType ;
        dataid:typeTemplate  "application/sparql-results+xml" ;
        dct:conformsTo        <http://dataid.dbpedia.org/ns/core> .

dataid-mt:MediaType_turtle_x-bzip2
        a                      dataid:MediaType ;
        dataid:innerMediaType  dataid:MediaType_turtle ;
        dataid:typeExtension   ".bz2" ;
        dataid:typeTemplate    "application/x-bzip2" ;
        dct:conformsTo          <http://dataid.dbpedia.org/ns/core> .

dataid-mt:MediaType_n-quads_x-bzip2
        a                      dataid:MediaType ;
        dataid:innerMediaType  dataid:MediaType_n-quads ;
        dataid:typeExtension   ".bz2" ;
        dataid:typeTemplate    "application/x-bzip2" ;
        dct:conformsTo          <http://dataid.dbpedia.org/ns/core> .

dataid:MediaType_n-quads
        a                     dataid:MediaType ;
        dataid:typeExtension  ".nq", ".tql" ;
        dataid:typeTemplate   "application/n-quads" ;
        dct:conformsTo         <http://dataid.dbpedia.org/ns/core> .

dataid:MediaType_turtle
        a                     dataid:MediaType ;
        dataid:typeExtension  ".ttl" ;
        dataid:typeTemplate   "text/turtle" ;
        dct:conformsTo         <http://dataid.dbpedia.org/ns/core> .
```

Declaration

"Ich versichere, dass ich die vorliegende Arbeit selbständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel angefertigt habe, insbesondere sind wörtliche oder sinngemäße Zitate als solche gekennzeichnet. Mir ist bekannt, dass Zuwiderhandlung auch nachträglich zur Aberkennung des Abschlusses führen kann".

Ort Datum Unterschrift