Getting started with DCAT-AP

Everything you never wanted to know about data catalogs, but had to learn anyway

Bart Hanssens

2024-06-20

Table of Contents

DRAFT

```
This document is only a rough draft.
```

Preamble

About this book

This book intends to be a pratical introduction to the metadata specification "DCAT-AP", which is generally used (but not limited to) describing web services and downloadable files on - mostly European - data portals.

The intention is not to be an encyclopedic reference of all the nitty-gritty details, but to be offer a hands-on guide to developers and maintainers of data portals and data suppliers.

Some technical background is required. More specifically, a basic knowledge of XML is expected, since many examples will be provided in XML.

About the author

Bart Hanssens is the architect of the Belgian data portal data.gov.be. While the front-end of this portal may not be the most attractive site in existence, the back-end was designed to work with DCAT-AP since 2014, collecting and harmonizing metadata from various local, regional and federal sources.

Bart contributed to the DCAT-AP specification, and the open source linked data library Eclipse RDF4J.

Introduction to RDF

The following sections contains an quick introduction of RDF, the framework underpinning linked open data on the semantic web.

Linked data and the semantic web

The World Wide Web (WWW) revolutionized the way information is shared across the globe: from sharing cutting-edge research results - its original purpose - to clickbait about celebrities, it’s all on the web.

Problem is, while we humas are smart enough to figure out what a web page is all about, HTML is mostly gibberish to those poor machines automating things for us. Although browers can display images and text in a pleasing way, they really can’t differentiate between a hotel review and a sports article, based on HTML alone.

So, clever minds came up with the idea of describing things in a meaningful (`semantic`) way, and serving this information over the same HTTP-driven network as the human-centic web. Including, of course, linking various sources together…

Enter `Resource Description Framework`, or RDF in short.

Statements and triples

> Everything that comes in threes is perfect.
>
> — Aristotle, 4th century BC

RDF is all about making *statements*, which are very basic sentences to describe something, written down as a combination of exactly 3 parts (hence the name *triples*):

`<subject> <predicate> <object>`

Example 1. Example of statements

```
<John> <buys an> <apple>
<Jane> <is born in> <Paris>
```

Less is more…

So it is possible to express *everything*, albeit not necessarily in the most concise way. For example, it requires multiple statements to express a sentence like `John has been working for ACME Corp since September 2015`

```
<John> <is an employee in> <Contract>
<ACME Corp> <is an employer in> <Contract>
<Contract> <starts in> "9/2015"
```

This also shows that statements can easily be linked, or more formally, the `object` of one statement can be the `subject` of numerous other statements and vice versa.

Identifiers

Now, in order to be able to properly *link* things, we need to *identify* them first. Which in turn means that identifiers have to be created and maintained, preferably in a structured but decentralized way (so anyone can create identifiers).

Sounds familiar ? Links pointing to webpages (URLs) are exactly that.

URI and URLs

NOTE With the exception of the `example.com` domain for documentation purposes, it is considered bad practice to "create" URIs in someone elses domain.

NOTE `http://example.com` and `https://example.com` (note the `s`) are **not** the same identifier.

Dereferencable URIs

Dereferencable is a fancy way to say that a URI will actually return something meaningful when a browser or another tool accesses it.

In most cases, this is via a HTTP GET request. Using the good old HTTP `Accept` header, it is possible to

NOTE A URI does not *have* to be dereferencable in order to be useful, but it helps.

Literals

Not everything has to be an identifier, often a simple value or *literal* will do just fine: book titles, timestamps, house numbers… are just a few random examples.

Language tags and data types

Now, a very common use case for titles and descriptions is to have translations, or at least an indication of the language.

Following the Turns out there is a shortcut: literal values can take a language tag *or* a datatype (not both).

… and beyond that

In a *graph* or sometimes called a *context*

And more recently, RDF* (RDF-Star)

Modelling

Classes and properties

Classes are For instance, a `Document`, a `Person`…

Classes may be subclasses of other (parent) classes

Properties Properties may be subproperties of other (parent) properties.

Both class names and property names are case-sensitive. By convention, class name start with an uppercase and property names with a lowercase.

NOTE Properties are often not tightly coupled to classes, allowing them to be reused across completely different classes.

Vocabularies and ontologies

A *vocabulary* is a well-defined collection of classes and properties.

An *ontology* is a vocabulary on steroids: not only does it contain definitions, it also adds some logic constraints. For instance, an ontology may not allow that something is both a `Document` and an `Organization` at the same time.

Reusing vocabularies

Vocabularies can be mixed and matched.

In fact, it's even a best practice to reuse existing ones when developping new vocabularies: doing so reduces the learning curve for other parties, and increases interoperability between different data sources.

In order to reuse vocabularies, one should be able to *find* them first. A great compilation of freely available vocabularies is the Linked Open Vocabularies portal.

Another interesting source is SEMIC: it contains vocabularies specifically developed by / for administrations of the European Union, including DCAT-AP.

RDF Schema

RDF Schema, or RDFS, is

Describing classes

Describing properties

`Domain` and `Range`

Multiple domains are allowed.

Some properties are indeed very generic, e.g. a `name` property makes sense on a `Person` class, but can be used on `Organizations` and `Images` as well.

NOTE

Unlike object-oriented programming, a property doesn't really belong to a specific class.

Which also means it's not a good idea to use the class name as part of the property name, e.g. `MyClass_Property`

Range:

The class a range points to, does not have to be part of the same vocabulary: it is quite common to point to classes from well-known vocabularies.

OWL

Web Ontology Language (OWL) is much more complex

Yes, the abbreviation should have been `WOL`, but `OWL` sounds so much better…

Reasoning

```
If it looks like a duck, swims like a duck, and quacks like a duck, then it probably
is a duck.
```

Vocabularies and ontologies do not magically turn RDF data into vast pools of knowledge. It requires special tools, *reasoners*, to make assumptions and derive new facts from the RDFS / OWL

Validation

While reasoners can be used to detect some inconsistencies in data, they don't quite fit the bill as a general data quality tool.

Even worse, reasoners can derive new statements and may come to logical but surprising results, which is typically not the intended behavior when performing low-level quality checks.

For instance, if an ontology specifies that a person can only live in 1 place at the same time, and we throw the statements `Jane lives in Paris` and `Jane lives in London` into the mix, a reasoner may conclude that `Paris` and `London` are actually the same place…

SHACL

Validation is relative new

File formats

RDF data can be *serialized* to several file formats.

This may come in handy when using RDF data in non-RDF data flows, though in practice - due to the flexible - doing so may not exactly be a walk in the park.

Let's compare a few common file formats using the following set of statements

Example 2. Set of statements

```
<vCard> <is a> <Standard>
<vCard> <has label> "Ontology for vCard"@en
<vCard> <is published on> "22 May 2014"
```

N-Triples

N-Triples is a very simple text format: every line contains exactly one one unabbreviated statement. It can easily be streamed, and works quite nice with well-know Unix command-line tools like `grep`.

The downside is that N-Triples files are quite verbose, since the format does not allow the use of prefixes to abbreviate commonly used namespaces, nor does the format provide options to group or structure statements in a visually appealing way ("pretty-printing").

Example 3. N-Triples file

```
<http://www.w3.org/2006/vcard/ns#> <http://www.w3.org/1999/02/22-rdf-syntax-
ns#type> <http://purl.org/dc/terms/Standard> .
<http://www.w3.org/2006/vcard/ns#> <http://www.w3.org/1999/02/22-rdf-syntax-
ns#label> "Ontology for vCard"@en .
```

```
<http://www.w3.org/2006/vcard/ns#> <http://purl.org/dc/terms/issued> "2014-05-
14"^^<http://www.w3.org/2001/XMLSchema#date> .
```

Turtle

Turtle is a slightly more complicated format, but it is much more compact and easier to read. Namespace prefixes can be used, and some syntactic sugar is available to produce smaller and `prettier` files.

It is therefore often used for files that are likely to be viewed by subject experts, e.g. data models and thesauri.

The following example shows how the statements can (but don't have to) be nicely grouped together, how namespaces prefixes can be used as a shorthand,

Since the rdf:type, the `a` is a

Example 4. Turtle file

```
@prefix dct: <http://purl.org/dc/terms/> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

<http://www.w3.org/2006/vcard/ns#>:
  a dct:Standard ;
  rdf:label "Ontology for vCard"@en ;
  dct:issued "2014-05-14"^^xsd:date .
```

Modern RDF parsers also accept `PREFIX` instead of `@prefix`

RDF/XML

RDF/XML was one of the first serialization formats, which is not surprisingly since RDF was developed within the W3C consortium, which was also instrumental in the development of XML.

The format is quite generic and flexible, which also means that - even for small amounts of data - there are multiple ways to express the same data.

As with general XML files, indentation does not matter.

Example 5. RDF/XML file

```
<?xml version="1.0" encoding="utf-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dct="http://purl.org/dc/terms/">

  <dct:Standard rdf:about="http://www.w3.org/2006/vcard/ns#">
```

```
    <rdf:label xml:lang="en">Ontology for vCard</rdf:label>
    <dct:issued rdf:datatype="http://www.w3.org/2001/XMLSchema#date">2014-05-
14</dct:issued>
  </dct:Standard>

</rdf:RDF>
```

Other

Triples in XML is another XML format. It is hardly used anymore.

Notation 3

TriG

HDT

JSON-LD

JSON for Linked Data

Embedding in webpages

RDFa in HTML

RDF in Attributes, or RDFa, allows structured but non-RDF formats like HTML to embed RDF data in a non-disruptive way.

The benefit is that both the human-friendly HTML representation and the machine-friendly data are present in the same webpage, which should make it easier to maintain both views.

At one time there were high hopes for this format, but most web content management systems lack decent support for RDFa. A less complex variant, RDF-Lite, was introduced, but didn't gain much traction either. It probably didn't help that yet another (non-RDF) specification, Microdata, entered the market as well.

Nowadays the legacy of RDFa lives on in the more popular Open Graph protocol, developed and supported by Facebook to share info in a social media context. OGP was inspired by RDFa, but it is less complicated and thus easier to implement.

More information can be found in the RDFa Primer, and the RDFa portal.

JSON-LD in HTML

Search engines like Google benefit from structured data, and can use some

See https://developers.google.com/search/docs/appearance/structured-data/dataset

Storing RDF: Triple stores

RDF statements are often stored in specialized data stores, called *triple stores*.

Most of these triple stores offer import/export from multiple file formats, and create/read/update/delete operations via the SPARQL query and update language.

It is, however, not always necessary to use a triple store to generate RDF data. Sometimes a database and a template engine will do just fine.

Querying and updating with SPARQL

Those who are familiar with XML may recognize functions

Data Catalog (DCAT)

DCAT

Is a very simple, based on DCTERMS

```
title simplified model

Catalog - Dataset
Dataset - Distribution
```

DCAT 2

DCAT version 2 adds better support for (web)services

DCAT 3

DCAT version 3 focusses on documenting series of related datasets.

It is up to the publishere of the datasets to decide what "related" means: it could be a collection of statistics published throughout the years, for instance, or a set of road maps, ...

DCAT-AP

See also https://semiceu.github.io/DCAT-AP/releases/3.0.0/

Catalog

Describing the catalog

Dataset

Describing the dataset

`dct:title`

`dct:description`

`dcat:keyword` `dcat:theme`

Licenses and rights

Distribution

Accessing and downloading

`dcat:accessURL` `dcat:downloadURL`

Dataservice

Various specializations

BRegDCAT-AP

See also https://github.com/SEMICeu/BregDCAT-AP

CiteDCAT-AP

See also https://ec-jrc.github.io/datacite-to-dcat-ap/

DCAT-AP-JRC

See also https://ec-jrc.github.io/dcat-ap-jrc/

DCAT-AP HvD

Implementing Regulation 2023/138/EU of 21 December 2022 laying down a list of specific high-value datasets and the arrangements for their publication and re-use

See also https://semiceu.github.io/DCAT-AP/releases/2.2.0-hvd/

EPOS-DCAT-AP

See also https://epos-eu.github.io/EPOS-DCAT-AP/

GeoDCAT-AP

Directive 2007/2/EC of 14 March 2007 establishing an Infrastructure for Spatial Information in the European Community (INSPIRE)

https://semiceu.github.io/GeoDCAT-AP/releases/

GNSS-DCAT-AP

See also https://zenodo.org/records/10955559

HealthDCAT-AP

See also https://healthdcat-ap.github.io/

MLDCAT-AP

See also https://semiceu.github.io/MLDCAT-AP/releases/2.0.0/

MobilityDCAT-AP

Directive 2010/40/EU of 7 July 2010 on the framework for the deployment of Intelligent Transport Systems in the field of road transport and for interfaces with other modes of transport (ITS)

See also https://w3id.org/mobilitydcat-ap/releases/

StatDCAT-AP

See also https://github.com/SEMICeu/StatDCAT-AP

National variants

Besides the application profiles listed before, several countries have created their own variants, which may slightly differ in the number of required properties. Some of them may not be actively developed anymore.

Belgium

DCAT-AP-BE DCAT-AP-VL

Finland

https://www.avoindata.fi/en/dcat-ap

Germany: DCAT-AP.DE

https://www.dcat-ap.de/def/dcatde/2.0/spec/

Italy: DCAT-AP_IT

https://www.dati.gov.it/content/dcat-ap-it-v10-profilo-italiano-dcat-ap-0

The Netherlands: DCAT-AP-DONL

https://dataoverheid.github.io/dcat-ap-donl/

Norway: DCAT-AP-NO

https://data.norge.no/specification/dcat-ap-no

Poland: DCAT-AP-PL

https://dane.gov.pl/dcat-ap-pl/

Spain: DCAT-AP-Spain

Sweden: DCAT-AP-SE

https://docs.dataportal.se/dcat/en/

Switserland: DCAT-AP CH

https://www.dcat-ap.ch/

Vocabularies

The following section provides an introduction to vocabularies that are commonly used with, or referred to by, DCAT-AP.

Once again the aim is not to give a complete overview, but to provide some background information on the most important classes and properties within the context of data catalogs.

ADMS

Full name	Asset Description Metadata Schema
Namespace	http://www.w3.org/ns/adms#
Prefix	adms
See also	https://semiceu.github.io/ADMS/releases/2.00/
Classes	Identifier
Properties	identifier, sample

DCTERMS

Full name	Dublin Core Metadata Initiative Terms
Namespace	http://purl.org/dc/terms/
Prefix	dcterms (or soemtimes dc or dct)
See also	https://www.dublincore.org/specifications/dublin-core/dcmi-terms/
Classes	FileFormat, Frequency, LicenseDocument, LinguisticSystem, Location, MediaType, MediaTypeOrExtent, PeriodOfTime, ProvenanceStatement, RightsStatement, Standard
Properties	accessRights, accrualPeriodicity, available, conformsTo, contributor, created, creator, description, format, identifier, issued, language, license, modified, provenance, publisher, references, relation, rights, rightsHolder, source, spatial, subject, title, type

DCAT leans heavily on the popular and well-supported Dublin Core vocabulary.

The date properties `created`, `issued`, `modified`

The `title` and `description` properties are free text values to provide a meaningful title and description of a subject. It is not uncommon to provide titles and/or descriptions in multiple languages, with a tag to indicate the language. Even when there is only one title or description, it is a good idea to add a language tag anyway, in case the value needs to be machine-translated or combined with other datasets in a multilingual context.

```
`creator`, `contributor`, `rightsHolder`
```

`accessRights, `license`, `rights`, the latter two pointing to `LicenseDocument` and `RightsStatement` classes.

`conformsTo`, `Standard` class

A more compelling name for `accrualPeriodicity` would be probably be update frequency, since the range of the property is a `Frequency` class.

NOTE Most people will associate Dublin with the capital of Ireland, but in this case it refers to Dublin in Ohio, USA.

DQV

Full name	Data Quality Vocabulary
Namespace	http://www.w3.org/ns/dqv#.
Prefix	dqv
See also	https://www.w3.org/TR/vocab-dqv/

FOAF

Full name	Friend-of-a-Friend
Namespace	http://xmlns.com/foaf/0.1/
Prefix	foaf
See also	http://xmlns.com/foaf/spec/
Classes	Agent, Document, Organization, Person
Properties	familyName, givenName, homepage, name, page, primaryTopic

There is some overlap with other vocabularies like VCARD and Schema

A `Person` or an `Organization`, acting an an `Agent` `

GeoSPARQL

Full name	GeoSPARQL Ontology
Namespace	http://www.opengis.net/ont/geosparql#
Prefix	geo (or gsp)
See also	http://www.opengis.net/doc/IS/geosparql/1.1
Data types	wkt

LOCN

Full name	Location Core Vocabulary
Namespace	http://www.w3.org/ns/locn#
Prefix	locn
See also	https://www.w3.org/ns/legacy_locn
Classes	
Properties	

Physical location

It was developed under the ISA program A newer version is being developed under the SEMIC.eu umbrella as the Core Location Vocabulary

ORG

OWL

Full name	Web Ontology Language
Namespace	http://www.w3.org/2002/07/owl#
Prefix	owl
See also	https://www.w3.org/TR/owl2-rdf-based-semantics/
Classes	
Properties	sameAs, versionInfo

While OWL is used to describe ontologies, some OWL properties do pop up in datasets as well.

`owl:versionInfo' is sometimes used to add a version number or label to datasets.

`owl:sameAs` can be used to indicate that two different URIs are actually describing the exact same thing. This may have some unintended side-effects when a reasoner comes into play, because it implies that any statement about A is also a statement about B and vice versa, so use with care.

An alternative approach is to use the `skos:exactMatch` property, which merely indicates that different subjects match, without affecting reasoning.

PROV

Full name	Provenance Ontology
Namespace	http://www.w3.org/ns/prov#
Prefix	prov
See also	https://www.w3.org/TR/prov-o/
Properties	endDate, startDate

Schema

Full name	Schema.org
Namespace	https://schema.org/
Prefix	schema (or sdo)
See also	https://schema.org
Properties	endDate, startDate

Schema.org is a massive collection of useful classes and properties. Founded by search engins Google, Yahoo, (Microsoft) Bing and Yandex, it features an interesting mix of e-commerce, health and other topics .

DCAT originally used `schema:startDate` and `schema:endDate` to indicate the temporal coverage of a dataset, but DCAT version 2 added two very similar properties ``dcat:startDate`and `dcat:endDate``.

See https://github.com/w3c/dxwg/issues/85 for an in-depth discussion on why these properties were duplicated. Most readers only need to remember that the `dcat:`-versions are now the preferred way to document start and end date.

SKOS

Full name	Simple Knowledge Organization System
Namespace	http://www.w3.org/2004/02/skos/core#
Prefix	skos
See also	https://www.w3.org/TR/skos-reference/ and https://www.w3.org/TR/skos-primer/
Classes	Concept, ConceptScheme
Properties	altLabel, broader, hasTopConcept, inScheme, narrower, notation, prefLabel, sameAs, topConceptOf

It is very well suited to publish code lists and

A *term* (entry in a thesauri) `skos:Concept`

The `skos:broader` (and the inverse property `skos:narrower`) is used to create hierachical structures.

Labels

Every term should have a preferred label `skos:prefLabel` (possibly in multiple languages), and may have multiple alternative labels `skos:altLabel`

In addition - or instead of a - prefLabel

Mapping to other thesauri

It is also possible to compare terms in one thesaurus with terms belonging to another thesaurus, using the `skos:broadMatch`, `skos:narrowMatch`, `skos:closeMatch` and `skos:exactMatch` properties.

The EU Publications Office publishes various code lists and thesauri in SKOS, ranging from simple lists like the Authority tables to massive thesauri like EUROVOC

VCARD

Full name	vCard Ontology
Namespace	http://www.w3.org/2006/vcard/ns#
Prefix	vcard
See also	https://www.w3.org/TR/vcard-rdf/
Classes	Individual, Kind, Organization
Properties	fn, hasEmail

Is a bit…messy.

XSD

Full name	XML Schema Part 2: Datatypes
Namespace	http://www.w3.org/2001/XMLSchema#
Prefix	xsd
See also	http://www.w3.org/TR/xmlschema-2/
Data types	anyURI, date, dateTime, decimal, duration, integer

The `anyURI` can be used to indicate that a literal value must follow the format of a URI. For instance, the URL of a webpage or a mailto-link. This is much less used than one may expect, because e.g. a `dcat:landingPage` must be an RDF *resource* (which cannot take a data type), not a *literal*.

The `integer` data type is used to express positive or negative numeric values without decimal point. It does not impose upper- or lower limits, so be careful when mapping xsd:integers to `int` data types in programming languages like PHP, Java or c, or SQL databases like Postgresql or MySQL. You may need to use bigger data types like `bigint` instead.

The same goes to some extent for the `decimal` data type for numeric values with a decimal point. It has arbitrary precision, meaning that a `float` or even a `double` might not cut it to preserve all significant digits.

The `date` and `dateTime` data types are heavily based on ISO8601, but not exactly the same in some corner cases. Both data types can take a timezone.

The lesser known `duration` is used to document a period of time, measured in various units of time.

Appendix A: More on identifiers

URN, URI, IRI

Persistent URLs (PURL)

A persistent URL (or PURL in short) is nothing more (and nothing less) than an URL that does not change. For how long ? Basically forever… once a PURL has been created, it is supposed to remain available and unchanged until the dawn of time.

Skolem URI

Persons

It is sometimes useful to add metadata about individuals to datasets. Researchers, for instance, often want to be mentioned as the author of - or a contributor to - a dataset or scientific paper.

Unfortunately names are unlikely to be unique - just imagine how many `John Smith’s there are - so it’s not always possible to

orcid

Organizations

OpenCorporates collects information from

ror

File formats

DCAT-AP requires the use of the Publication Office’s File type autority table for `dct:format` URIs

However, the dcat properties `dcat:mediaType`, `dcat:compressFormat`, `dcat:packageFormat` should all be using URIS of registered IANA mimetypes.

Note that there isn’t always a registered IANA mimetype when there is an entry in the Publication Offices’s authority table, or vice versa.

In general, the Publication Office is quite flexible in adding new file formats. Everyone can make suggestions via the `Contribut`] button on the File type overview page.

IANA procedures are a bit more strict, especially when it comes down to registering vendor-specific formats, but they too offer a webform to submit suggestions.

Geographic areas

Geonames.org

Appendix B: Common mistakes in DCAT-AP

Not using the EU Publications Office controlled lists

Some portals claim to adhere to DCAT-AP, but are in fact producing DCAT.

Incorrect date or duration formats

Mixing contact point IDs and organization IDs

Mixing IDs and URLs

This can lead to undesired side-effects when quering / combining data.

Not setting explicit language tags

Appendix C: Tools

The Interop TestBed

RDF libraries

Java

Apache Jena and Eclipse RDF4J are two popular open source libraries to read and write RDF. Both include a triple store.

Jena provides various built-in reasoners, while RDF4J has an excellent SHACL validation engine.

Javascript / NodeJS

Comunica makes it easy to query

Python, PHP

The Redland RDF Libraries are a set of libraries and tools written in c, with bindings for Python and PHP

Triplestores

In addition to the triplestores provided by Jena, RDF4J and LibRDF

The following commercial data stores can be seemingly used with Eclipse RDF4J:

- Ontotext GraphDB
- Oracle Graph server
- StarDog

Another popular data store is OpenLink Virtuoso, an open source version is available.

Other tools

Prefix.cc makes it easy find the preferred namespace prefix for a given namespace, and vice versa.

EasyRDF is an online tool to quickly convert RDF snippets into another format. While it is useful for testing, it is not intended for converting large files.

==