Describing Linked Datasets

On the Design and Usage of voiD, the "Vocabulary Of Interlinked Datasets"

Keith Alexander[*]
Talis Ltd.

Richard Cyganiak[†]
DERI, National University of Ireland, Galway

Michael Hausenblas[‡]
DERI, National University of Ireland, Galway

Jun Zhao[§]
Department of Zoology, University of Oxford

ABSTRACT

In this paper we discuss the design and implementation of voiD, the "Vocabulary Of Interlinked Datasets", a vocabulary that allows to formally describe linked RDF datasets. We report on use cases for voiD, the current state of the specification and its potential applications in the context of linked datasets.

1. INTRODUCTION

With the growth of the number of linked datasets [12], automating certain tasks, such as discovery, selection and optimisation, becomes more and more important. Now, one might argue that URIs and RDF [17] are all one needs to explore the linked datasets; follow-your-nose[1], however bears some inherent problems. The possible links that can be followed from a starting URI raises both performance and trust issues. The main reason for these issues lies in the granularity level of the available descriptions. Additionally, the dynamics of the data-sources [13] also has an impact on the performance of, say, a crawl over a collection of datasets; and the reliability of secondary data resources [26].

In the early days of linked data (2006 and 2007) [5] the main focus of the community was on publishing data and finding good practices. Now, in the second phase, other issues such as usability, quality, performance, reliability of the infrastructure and the data in the linked data ecosystem are increasingly recognized to be important.

How can we overcome the limitations of follow-your-nose while retaining the self-descriptive momentum and being able to exploit available tools, methodologies, etc.? A simple yet effective approach is to decrease granularity. Rather than talking about single resources, we talk about something which up to now only existed in drawings, such as in the LOD cloud[2], which graphically represents the landscape of linked data on the Web using bubbles for datasets and arcs between bubbles for the links between these datasets. The voiD vocabulary, the "Vocabulary Of Interlinked Datasets", allows one to describe `datasets` (the bubbles) and `linksets` (the arcs between the bubbles), and in turn enables a number of tasks to be automated in a scalable manner.

The remainder of this paper is as follows: in Section 2 we discuss some use cases that provide the motivation for voiD. Then, in Section 3 we report on the design of the voiD core vocabulary and its usage along with other vocabularies such as Dublin Core [25]. We describe the publication and the consumption of voiD descriptions in Section 4. In Section 5 we discuss current and potential applications of voiD and report on related work in Section 6. We discuss future plans and conclude in Section 7.

2. USE CASES

In the following we will describe our motivation use cases for voiD. In the context of linked data, we basically differentiate between:

- on the one hand linked data **publisher** (a person or organisation exposing structured data as RDF on the Web and interlink it with other datasets), and
- linked data **consumer** on the other hand; these might be machines, for example using a semantic indexer or a query engine or, as well, humans, e.g., when using a Web of Data browser such as Tabulator [4].

It is worth noting that the following use cases are not necessarily restricted to the linked data domain.

2.1 Efficient Discovery of Datasets

2.1.1 Dataset Publisher

A dataset publisher might not be identical with the party who created the raw datasets, but one who publishes them onto the Web in a more accessible format. A dataset publisher wants to be able to publish metadata about the dataset such that:

- The dataset can be found and aggregated by search engine applications, or discovered in relevant searches;
- The metadata provides clear licensing information so that consumers can know how they can use the data

[*]keith.alexander@talis.com
[†]richard.cyganiak@cyganiak.de
[‡]michael.hausenblas@deri.org
[§]jun.zhao@zoo.ox.ac.uk

[1]`http://esw.w3.org/topic/FollowLinksForMoreInformation`
[2]`http://www4.wiwiss.fu-berlin.de/bizer/pub/lod-datasets_2008-09-18.html`

LDOW2009, April 20, 2009, Madrid, Spain.

and to whom they should attribute credits for creating/publishing the dataset;

- That consumers can obtain information about access interfaces, such as APIs and SPARQL endpoints.

It is in the best interest of a dataset publisher to provide potential users of the data with information that supports them in accessing and using the dataset.

2.1.2 Search Engine Provider

A search engine provider wants to discover detailed descriptions about datasets efficiently. A crawler has stumbled upon an individual RDF document on the Web. How will it discover metadata that applies to the entire dataset and cannot be repeated in every single document? The simple approach of just putting the voiD description online and linking it from somewhere on the site does not meet our needs, as it could take the crawler a long time to find the description. It is important that the voiD description is discoverable as soon as the crawler finds the first RDF document, so that the crawler can use voiD metadata to guide its processing of the data.

The Sindice search engine [23] already uses Semantic Sitemaps [7] to enable discovery and efficient processing of datasets. It seems natural to address the situations above by building on Semantic Sitemaps.

2.2 Expressing Research Data

A developer working together with biologists wants to help domain experts to find research data published by their peer colleagues. These are often produced for a particular experiment, for a particular study or publication, or hosted by a particular public database. Scientists might know whose datasets they would prefer to access because they often have a clear idea about their content or they trust more on that data provider. When looking for new datasets, they might search for datasets that provide relevant content (such as information about genes, proteins, or micro-array gene expression), that are produced in a right experiment environment, or that provide additional information that will complement their local experiment results.

To find the right dataset and to make this dataset accessible for biologists in a user-facing application, the developer often has to go through the following process:

- Locate a dataset that contains information relevant to biologists' research interests, such as information about a specific organism; or more specifically, genomic information about a particular organism;

- Find out how this dataset can be programmatically accessed, as an RDF dump, through SPARQL endpoint or any other protocol;

- Find out the licence associated with the dataset, making sure that data are accessible under open-access licence or certain attribution;

- Understand the content of the dataset in order to perform an alignment with other datasets. Information about URIs used in the dataset can help one with the data identity alignment, schema(s) used in the dataset for data schema alignment, and its links with other datasets for assisting data integration.

2.3 Effective Dataset Selection

A consumer may have discovered several datasets, for example as a result of an indexer query. The question then arises how to select appropriate datasets from this list of potential candidates. The consumer, either a human or a query federation engine, might wish to define "appropriateness" along the following criteria:

1. The *content* of the dataset, that is, what the dataset mainly is about. Based on some kind of categorisation scheme a selection could take place;

2. The *interlinking* to other datasets, that is, to which other datasets and how the dataset is interlinked;

3. *Vocabularies* used in the dataset.

The criteria listed above can be understood in terms of quality and quantity. For example, one might be interested only in datasets containing `foaf:interest` links to a certain other dataset. Or, where the number of links are of interest, one may specify that only datasets with more than one million links should be taken into account.

2.4 Query Optimisation

With many datasets now on the Web both connectable (through shared vocabulary terms) and connecting (by linking to resources in other datasets), it is naturally desirable to query across multiple datasets at once with SPARQL.

Optimisation of SPARQL queries can be achieved in a static way. A set of logical rules [22] can be applied to a query engine, to calculate all equivalent query plans for a given query and then choose the most optimised query plan to be executed. To optimise SPARQL queries dynamically, i.e., deciding the best execution approach during the execution phase [14], one can use the statistical information about datasets, such as how much information is provided about a particular entity or property. This information can be used by the query mediator to optimise query plans by, for example, modifying the order in which a query pattern is executed according to the estimated size of data results [8].

3. VOCABULARY DESIGN

The Vocabulary of Interlinked Datasets (voiD) [2] is a vocabulary and a set of instructions that enables the discovery and usage of linked datasets. The principle of the voiD effort is to use real requirements to guide the scope of the design, and to re-use existing vocabularies wherever possible instead of creating our own. Therefore, we have kept the creation of new classes and properties under the voiD namespace (`http://rdfs.org/ns/void#`) to the minimum.

3.1 Datasets

In the following, we will define and explain the basic concepts voiD operates with. A fundamental entity in voiD is a **dataset**.

Definition 1. *A* dataset *is a set of RDF triples that are published, maintained or aggregated by a single provider.*

We think of a dataset as a *meaningful* collection of triples, that deal with a certain topic, originate from a certain source or process, are hosted on a certain server, or are aggregated by a certain custodian. The term thus has a social dimension

that is not easy to capture in a formal definition. This differentiates datasets from *RDF graphs* [17], which are purely mathematical constructs. Any arbitrary set of RDF triples is an RDF graph, by definition, regardless of the triples' semantics. Also, typically a dataset is accessible on the Web, for example through resolvable HTTP URIs or through a SPARQL endpoint, and it contains sufficiently many triples that there is benefit in providing a concise summary.

The ultimate purpose of creating a `void:Dataset` instance is that this single resource represents the entire RDF dataset, and thus allows us to make statements about the entire dataset within the standard RDF model. The relationship between a `void:Dataset` instance and the concrete triples contained in the dataset is established only in an operational manner: A voiD description usually contains access information, such as the address of a SPARQL endpoint where the triples can be accessed.

We find that most datasets describe a well-defined set of resources. Hence, a dataset can also be seen as a set of descriptions for certain resources, which often share a common URI prefix (such as `http://dbpedia.org/resource/`).

HTTP URIs have "owners", due to their use of DNS domain names. *URI ownership* is defined as "a relation between a URI and a social entity, such as a person, organisation, or specification."[15] Information about a URI that is provided by the URI owner is called *authoritative information*. We use this notion to define **authoritative datasets**:

Definition 2. *A dataset is* authoritative *with respect to a certain URI namespace if it contains information about resources named by URIs in this namespace, and is published by the URI owner.*

A straightforward method of publishing authoritative datasets is by using resolvable HTTP URIs in the *linked data* style. The URI owner also configures the server that responds when the URI is resolved. Therefore, if resolving yields a description of the resource named by the URI, then the data is authoritative.

The notion of authoritative information supports the social convention that a URI owner gets to decide what a URI identifies. Providing authoritative information is how the URI owner communicates this decision to the world. Even if third parties disagree with that information, they can still agree that they are talking about the same thing, which would be much harder without the grounding provided by the existence of an authoritative source.

3.2 Linksets

Besides datasets, voiD also deals with interlinking between datasets. Interlinking in voiD is a first-class citizens, hence modelled as a class.

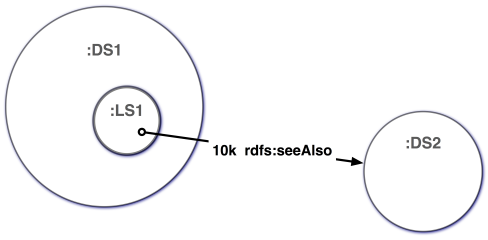


Figure 1: Interlinking modelling in voiD.

The conceptual model of voiD links is depicted in Fig. 1. Let us assume there are two datasets. One of them contains links to the other, that is, it contains RDF triples that connect resources from both datasets. We model this in voiD using two instances of `void:Dataset`, and another dataset `:LS1` which is a subset of one of the datasets, and declared to be of type `void:Linkset`. We define `void:Linkset` as:

Definition 3. *A linkset LS is a set of RDF triples where for all triples $t_i = \langle s_i, p_i, o_i \rangle \in LS$, the subject is in one dataset, i.e. all s_i are described in DS_{src}, and the object is in another dataset, i.e. all o_i are described in DS_{sink}.*

The natural expectation is that both DS_{src} and DS_{sink} are themselves described in voiD. We note that the triples t_i are often referred to as "interlinking triples".

3.2.1 Inline links vs. 3rd-party links

In voiD we are able to model two different situations: the **classic LOD**[3] case vs. the **3rd-party** case (Fig. 2). In the classic LOD case, the linkset is a subset of one of the two involved datasets, while in the 3rd-party case a third dataset is involved that actually contains the linkset.

Though the 3rd-party cases is not yet widely implemented in the context of linked data, this pattern of keeping links separate from interlinked datasets has been well argued in existing research such as found in the Hypertext community [6]. In LOD, there are already first applications (RKB explorer, see section 5.1), and it is very likely that such systems will evolve over time and grow considerably.

3.2.2 Interlinking Regarding Directionality

Independent of the former situation, voiD distinguishes between the **non-directed** vs. **directed** cases. In some cases one is interested in stating the direction of the interlinking (for example with `foaf:interest`), and in other situations the direction is of no interest (e.g., `owl:sameAs`), as shown in Fig. 3.

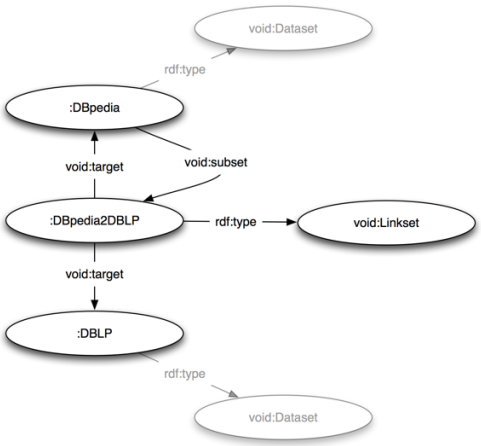
In order to express the interlinking as outlined above, voiD offers the following RDF properties:

- `void:subset` to state where the interlinking triples reside (read: a dataset `:DS` *has a subset* `:LS`);
- `void:target` to declare an interlinking target (for the non-directed case); in the directed case, one can use `void:subjectsTarget` and `void:objectsTarget` to determine the direction (both being sub-properties of `void:target`);
- `void:linkPredicate` to express the RDF property (type) of the interlinking in a linkset.

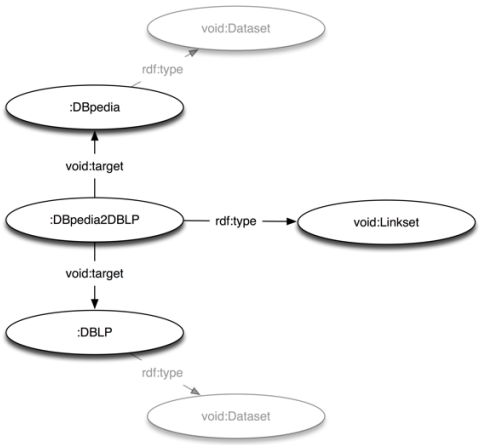
We note that it is expected that per RDF predicate a respective instance of `void:Linkset` could be created, depending on the needs of an application. Further, one may take into consideration that due to the modelling of `void:target` and its sub-properties, light-weight subsumption inferencing may be necessary to apply generic queries that will not distinguish between the directed and non-directed case.

In listing 1 a sample voiD description is depicted describing the interlinking from DBpedia to DBLP. It is an example

[3]LOD ... Linking Open Data datasets, see `http://esw.w3.org/topic/SweoIG/TaskForces/CommunityProjects/LinkingOpenData`



(a) Classic LOD case: Describing the `:DBpedia` dataset and its contained `:DBpedia2DBLP` linkset



(b) 3rd-party case: Describing the stand-alone `:DBpedia2DBLP` linkset

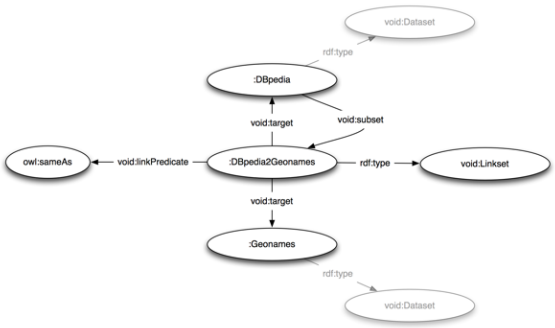
Figure 2: Interlinking regarding authoritative datasets.

for a directed case. The description defines nothing about who published this voiD description about DBpedia, which means that it could also be an example for a 3rd-party case. Further, the listing 2 shows a SPARQL query that is executed against listing 1 to search for a dataset that is about "computer science" and which is linked from DBpedia. The result yields the dataset `:DBLP`.

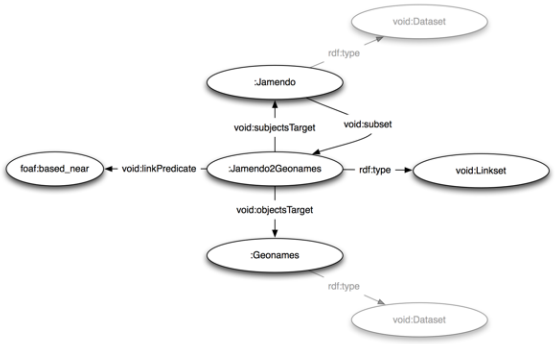
3.3 Reuse of Other Vocabularies

In the voiD guide [1] we describe the reuse of other vocabulary terms not directly defined in the core voiD vocabulary alongside with voiD. Some important properties from other vocabularies are listed in the following. We note that there are many other aspects one may want to choose to describe in a dataset. A complete description of recommended usage can be found from the voiD user guide [1].

- Properties from the `dcterms` namespace for general metadata, such as the publishing organization and publishing date of a dataset;



(a) Non-directed case.



(b) Directed case.

Figure 3: Interlinking regarding direction.

- `foaf:homepage` of the dataset's homepage should be used, to allow one to connect different descriptions of the same dataset provided in different places on the Web. The recommended process in voiD is IFP smushing on `foaf:homepage` property;

- `dcterms:subject` should be used to categorise a dataset. For the general case, we recommend the use of a DBpedia resource URI (`http://dbpedia.org/resource/XXX`) to categorise a dataset, where XXX stands for the thing which best describes the main topic of what the dataset is about. However, DBpedia might not contain concepts for describing some domain specific datasets. For example, there are no exact DBpedia resource URIs for describing that a dataset is about "in situ hybridisation image". We hence encourage publishers to describe such datasets using concepts widely adopted in their own communities, so that they can not only capture precisely the categorisation of their datasets but also ensure that these datasets could be connected with other relevant data from their domains;

- Statistical information represented using the "Statistical Core Vocabulary" (SCOVO)[4] [11].

3.4 Dataset Licensing

As stated in Section 2, it is crucial for a data publisher to associate appropriate licensing information with their published data, so that potential users of the dataset would

[4] `http://purl.org/NET/scovo`

```
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dc: <http://purl.org/dc/terms/> .
@prefix void: <http://rdfs.org/ns/void#> .
@prefix dbp: <http://dbpedia.org/resource/> .

:DBpedia a void:Dataset ;
   foaf:homepage <http://dbpedia.org/> ;
   void:subset :DBpedia2DBLP .

:DBLP a void:Dataset ;
   foaf:homepage <http://dblp.l3s.de/d2r/> ;
   dc:subject dbp:Computer_science ;
   dc:subject dbp:Journal ;
   dc:subject dbp:Proceedings .

:DBpedia2DBLP a void:Linkset ;
   void:subjectsTarget :DBpedia ;
   void:objectsTarget :DBLP ;
   void:linkPredicate owl:sameAs .
```

Listing 1: An exemplary voiD description.

```
SELECT DISTINCT ?dataset
WHERE {
 ?dataset a void:Dataset ;
          dcterms:subject dbp:Computer_science .
 ?linkset void:subjectsTarget :DBpedia ;
          void:objectsTarget ?dataset .
 }
```

Listing 2: An exemplary query on voiD.

know under which terms they can use it and what attribution they should apply. The `dcterms:license` property should be used to to point to the license under which a dataset has been published. Further, to allow automatic analysis of datasets, voiD also recommends a set of canonical identifiers for well-known licenses [1]. The example below states that the DBpedia dataset is published under the terms of the GNU Free Documentation License.

```
:DBpedia a void:Dataset ;
   dcterms:license
        <http://www.gnu.org/copyleft/fdl.html> .
```

Listing 3: An exemplary voiD description about data license.

Licensing of datasets is a complex issue. Datasets are collections of facts rather than creative works, and different laws apply. Scientists are most cautious about publishing their datasets onto the Web and they might request very specific or strict policies for sharing their data. Most licenses such as Creative Commons or the GPL are based on copyright and are designed to protect creative works, but not databases, and applying them to datasets might not have the desired legal result. Meanwhile, efforts such as Open Data Commons [19] and Science Commons [16] are developing dedicated licenses for data.

3.5 Statistics

Of special interest to distributed SPARQL agents will be the statistics about the triples available in the dataset, described with the `void:statItem` predicate. We adopt SCOVO for representing statistics. The main class in SCOVO is the `scovo:Item`, which records a single number or statistical value along with so called dimensions. We provide two types of information for describing statistics:

- Statistics concerning the whole dataset or linkset, such as overall triple count or fine-grained statistics, expressing the number of instances of a class or property by using different pre-defined dimensions, including `void:numberOfResources`, etc.;
- Attributing statistics to a source, recording where a statistical datum stems from.

Listing 4 demonstrates possible statistic information one can publish for their dataset. The current modelling of statistics

```
:DBpedia a void:Dataset ;
   void:statItem [
        rdf:value 20000;
        scovo:dimension void:numberOfResources ;
        scovo:dimension foaf:Person ;
        dcterms:source <http://wiki.dbpedia.org/>  ;
   ] .
```

Listing 4: Expressing statistics about a dataset in voiD.

in voiD is still experimental. We had to make choices between (i) a precise usage of scovo through a rather verbose expression and (ii) creations of shortcuts to express statistics needed for describing linked datasets:

- Scovo has an implicit assumption that all `scovo:Items` associated with the dataset they describe share the same dimensions. This does not fit well with our requirements for being able to mix items of different dimensions for a dataset. On the other hand, the correct Scovo modelling would lead to awkwardly complex and verbose notation for simple statistics.
- We encourage the use of classes and properties in places where scovo requires an instance of `scovo:Dimension`. This breaks the symmetry of the scovo model. scovo would require us to create a `scovo:Dimension` for each class or property. This would be quite verbose.

Because of the issues above, queries for statistics information using SPARQL can be awkward. It will often require a verbose check to make sure that an item has only certain dimensions and no others.

3.6 Additional Terms in voiD

RDF datasets use one or more RDF-Schema vocabularies or OWL ontologies, hence we provide the `void:vocabulary` to list vocabularies used in a dataset. To express technical features of a dataset, such as formats in which the data is available, one can use `void:feature`. Further, a SPARQL endpoint that provides access to a dataset via the SPARQL protocol can be announced using the `void:sparqlEndpoint` property. Listing 5 shows the usage of the terms described above. We note that a complete list of the terms is available from the voiD user guide [1].

```
:DBpedia void:sparqlEndpoint
     <http://dbpedia.org/sparql> ;
          void:feature [ dcterms:format
               "application/rdf+xml" ; ] ;
          void:vocabulary
               <http://xmlns.com/foaf/0.1/> .
```

Listing 5: Additional voiD terms usage.

4. PUBLICATION AND CONSUMPTION

We envision dataset publisher to offer a voiD description along with their dataset. A voiD description typically has two parts, (i) manually created part (categorisation, vocabulary, license, etc.), and (ii) automatically generated part, mainly regarding statistics.

In the following we will discuss the publication process of voiD descriptions and their discovery in order to consume them.

4.1 Publication

Publishing a voiD file means to physically deploy it on the Web in an RDF serialisation. We have detailed out the options in the voiD guide [1].

For dataset that are published as a collection RDF documents, as commonly seen in the linked data publishing style, one can use a `dcterms:isPartOf` triple in each document to link back to the URI identifying the voiD dataset, as shown in listing 6. Resolving the dataset URI will answer a voiD descriptions about the entire dataset, allowing agents to discover the voiD description when encountering an individual document from the collection. The intuition behind using the `dcterms:isPartOf` property is that the RDF document *contains an RDF graph* whose triples are *part of the dataset*.

```
<http://dbpedia.org/data/Berlin> dcterms:isPartOf
     :DBpedia .
```

Listing 6: Use backlinks publish voiD description of a dataset.

As discussed in [10], we can imagine that voiD descriptions are crawled and indexed by semantic search engines (such as Sindice [23] or Yahoo's search monkey [18]) in order to provide a central point of lookup.

4.2 Discovery via Sitemaps

A discovery mechanism for use by RDF-harvesting web crawlers (Fig. 4) has been defined as follows:

1. Given a domain name, the client gets the file `robots.txt` and searches for a line that starts with `Sitemap:`; the rest of that line is the URI of a sitemap;
2. The semantic sitemaps extension to the sitemap protocol defines a `<sc:dataset>` element that can have a `<sc:datasetURI>` child element. If present, the value of that element is a URI that identifies the dataset `datasetURI`;
3. The dataset URI `datasetURI` is dereferenced which yields the voiD description of the dataset.

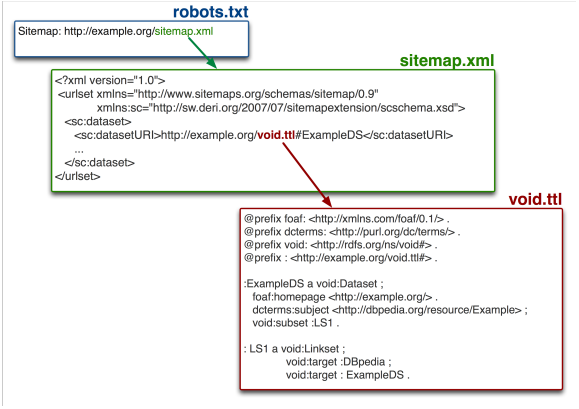


Figure 4: The voiD discovery-via-sitemaps process.

5. VOID IN THE WILD

After releasing the first edition of voiD in early 2009, we have seen a certain community uptake. People and organisations would start using in different areas and for different purposes, potentially far beyond what we have envisioned in the realm of our own use cases. We report on known usages of voiD in the following and point out potential application areas.

5.1 Existing Applications

5.1.1 Tools for Creating voiD Descriptions

To boot-strap the process of creating voiD descriptions, several tools are available: *ve*, the voiD editor (Fig. 5), liftSSM[5], an XSLT script able to boot-strap from a Semantic Sitemap, and, for creating the quantitative, statistical data, a new release of the NX parser[6], offering a voiD export for statistics.

5.1.2 "Linked Datasets Explorer" (LDE)

To let user browse and explore a collection of voiD descriptions, we have developed the LDE demonstrator. Fig. 6 shows the current state of LDE[7] which operates on a manually created, so called "seed" set of voiD descriptions.

5.1.3 RKB explorer

The RKB explorer has a voiD site[8] which enables querying and browsing for CRS datasets. Further, the interlinking of the RKB sites can be visualised using the underlying voiD descriptions (Fig. 7).

5.1.4 Query Federation

Only recently Clarck-Parsia announced their voiD support[9]:

> "There is a touch point with the linked data effort, which meant that the new voiD vocabulary

[5] `http://rdfs.org/ns/void-guide#sec_4_3_Publishing_tools`
[6] `http://sw.deri.org/2006/08/nxparser/release/nxparser-1.1.jar`
[7] `http://ld2sd.deri.org/lde/`
[8] `http://void.rkbexplorer.com/`
[9] `http://clarkparsia.com/weblog/2009/02/04/distributed-query-pellet-into-the-void/`

for describing datasets turns out to be very useful for describing the distributed data sources that we query over, including their interrelations."

Further, OpenLink plans to release its "Smart SPARQL Federation capabilities", based on voiD, soon.

5.1.5 Middleware

OpenLink's Sponger Middleware uses voiD for generating linked data from non-RDF data sources such as HTML pages. An example[10] from `http://linkeddata.uriburner.com/` with the voiD description deployed as XHTML+RDFa is shown in Fig. 8. Further, the statistics maintenance in their Virtuoso Quad Store is performed based on voiD.

5.2 Potential Applications

We envision voiD to be applied in many scenarios, some of which we have identified earlier in section 2. Only recently, for example, we have started to develop a dataset ranking algorithm based on voiD descriptions; this is subject to more research. One could further apply voiD to DARQ (Distributed ARQ) [22].

A totally different application domain is visualisation: for example, "The Map of Data"[11] in Sindice can be generated automatically thanks to voiD.

Ultimately, to be of use, one wants applications that benefit from voiD. Put in other words, this means that, given there are applications that consume voiD and offer some added value, the incentive for publishers to provide voiD descriptions is self-evident. One such application could be a sort of dynamic dataset selector which, configured with a specification of the dataset (topics, license, interlinking with certain other datasets) would at run-time of an application discover and select appropriate datasets according to the search specification.

6. RELATED WORK

To the best of our knowledge, no comparable approach to voiD exists. That is, in the context of the Web of Data, we are not aware of any specification that allows the description of datasets and their interlinking the way voiD does. However, we acknowledge previous work of Semantic Sitemaps [7] and build upon it.

In the scope of the Web of Documents, we note that at the time of writing a W3C Working Draft of POWDER (Protocol for Web Description Resources) [3] is available. POWDER aims at providing information about Web resources, such as scope, authoritative information, etc., without retrieving the resources themselves. POWDER comes in two flavours, (i) as human-legible XML, and (ii) in an RDF version. It also provides a GRDDL transformation to turn the former into the latter. The descriptions can be applied to groups of resources defined via listing of URIs, regular expressions, etc. Several publishing methods are suggested (via HTML `<link>` in the header, HTTP Link: header or using XHTML+RDFa). Especially in the Web of Trust, POWDER is expected to play a vital role, though implementation complexity might hinder wide-spread adoption.

Further, OASIS's XRDS (eXtensible Resource Descriptor Sequence) [24] is an XML format for metadata discovery about a resource. The discovery protocol for XRDS documents given a URI was defined in 2006 as part of Yadis, focusing on services such as OpenID and OAuth. In early 2008, XRDS-Simple was proposed[12], but is now obsolete.

Only very recently, the latest draft of "/host-meta" [21] was proposed. The core of this proposal is a single "well-known location", `/host-meta`, acting as a directory of the interesting metadata about a Web site. The format allows different types of site metadata to be referenced by an URI or included inline.

One could understand the "HTTP Link: header" [20] proposal related to voiD, as it also supports discovery, offering metadata about resources by resurrecting a (currently deprecated) feature of HTTP. This proposal is at the time of writing still under vivid discussion and not yet seen stable.

Regarding federated SPARQL queries, DARQ (Distributed ARQ) [22] proposes so called "service descriptions" that are able to specify capabilities of a SPARQL endpoints. The service descriptions enable the DARQ query engine to decompose a query into sub-queries, each of which can be answered by an individual service using query rewriting and cost-based query optimisation to speed-up query execution. Further, we note an attempt called "SPARQL Endpoint Description"[13] that aimed to allow the announcement of endpoint capabilities and contents, support discovery through service directories, and supply browsing and federation hints. Both proposals seem to be not further maintained and/or have not reached wide-spread adoption.

Finally, we note that the W3C Technical Architecture Group (TAG) started to contemplate about "Uniform Access to Metadata"[14], basically being a survey regarding the problem of specifying a uniform method for obtaining information pertaining to a resource without necessarily having to parse a representation of the resource.

7. OUTLOOK

We have released the voiD vocabulary and voiD user guide to linked data communities in January this year. In this release, we have used the use cases presented in section 2 to guide the design scope of the voiD vocabulary. Supports for describing the quality, provenance and versions of linked datasets are to be addressed in the next release of voiD. Also, the statistics modelling in the current voiD model is still experimental. We are communicating with user communities and the SCOVO team in order to propose a more stable modelling in the coming release[15]. Additionally we will liaison with initiatives such as the "Ontology Metadata Vocabulary" [9] sharing similar goals.

To test and evaluate the usefulness of voiD, we need tools that use voiD to support the discovery of datasets or the SPARQL query federation. Fortunately, semantic query engines like Sindice and SPARQL query processing systems (like OpenLink) are adopting voiD in their implementations. It is challenging to completely automate the creation of voiD descriptions. We need tools like the NX parser to take as

[10] `http://linkeddata.uriburner.com/about/html/http://twitter.com/mhausenblas#Dataset`
[11] `http://sindice.com/map`
[12] `http://www.hueniverse.com/hueniverse/2008/03/putting-xrds-si.html`
[13] `http://esw.w3.org/topic/SparqlEndpointDescription`
[14] `http://www.w3.org/2001/tag/doc/uniform-access-20090205.html`
[15] See `http://code.google.com/p/void-impl/issues/list?can=2&q=milestone:Release2.0forplannedissues`.

much as possible of the heavy lifting for non-technical data publishers as possible.

Acknowledgements

Our work has partly been supported by the European Commission under Grant No. 217031, FP7/ICT-2007.1.2, project "Domain Driven Design and Mashup Oriented Development based on Open Source Java Metaframework for Pragmatic, Reliable and Secure Web Development" (Romulus)[16], and the Joint Information Systems Committee [Project "FlyWeb"]. The authors would further like to thank (alphabetically): Orri Erling, Hugh Glaser, Olaf Hartig, Tom Heath, Andreas Langegger, Ian Millard, Marc-Alexandre Nolin, Yves Raimond, Yrjänä Rankka, Francois Scharffe, and Giovanni Tummarello.

8. REFERENCES

[1] K. Alexander, R. Cyganiak, M. Hausenblas, and J. Zhao. voiD guide—Using the Vocabulary of Interlinked Datasets. Community Draft, voiD working group, 2009. `http://rdfs.org/ns/void-guide/`.

[2] K. Alexander, R. Cyganiak, M. Hausenblas, and J. Zhao. voiD, the "Vocabulary of Interlinked Datasets". Community Draft, voiD working group, 2009. `http://rdfs.org/ns/void/`.

[3] P. Archer, K. Smith, and A. Perego. Protocol for Web Description Resources (POWDER): Description Resources. W3C Working Draft 14 November 2008, POWDER Working Group, 2008.

[4] T. Berners-Lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, and D. Sheets. Tabulator: Exploring and analyzing linked data on the Semantic Web. In *In Proceedings of the 3rd International Semantic Web User Interaction Workshop (SWUI06)*, Athens, Georgia, USA, 2006.

[5] C. Bizer, T. Heath, K. Idehen, and T. Berners-Lee. Linked Data on the Web (LDOW2008). In *Linked Data on the Web Workshop(WWW2008)*, 2008.

[6] L. A. Carr, D. C. DeRoure, W. Hall, and G. J. Hill. The Distributed Link Service: A tool for publishers, authors and readers). In *Proceedings of the 4th International World Wide Web Conference: The Web Revolution)*, pages 647–656, Boston, USA, 1995.

[7] R. Cyganiak, H. Stenzhorn, R. Delbru, S. Decker, and G. Tummarello. Semantic Sitemaps: Efficient and flexible access to datasets on the Semantic Web. In *Proceedings of the 5th European Semantic Web Conference*, volume 5021, pages 690–704, Tenerife, Spain, 2008.

[8] O. Hartig and R. Heese. The SPARQL query graph model for query optimization. In *Proceedings of the 4th European Semantic Web Conference 2007*, pages 564–578, Innsbruck, Austria, 2007.

[9] J. Hartmann, Y. Sure, P. Haase, R. Palma, and M. del Carmen Suárez-Figueroa. OMV – Ontology Metadata Vocabulary. In C. Welty, editor, *ISWC 2005 - In Ontology Patterns for the Semantic Web*, 2005.

[10] M. Hausenblas. Discovery and usage of linked datasets on the Web of data. In *Talis NodMag 4*, 2008.

[11] M. Hausenblas, W. Halb, Y. Raimond, L. Feigenbaum, and D. Ayers. SCOVO: Using statistics on the Web of data. In *6th European Semantic Web Conference (ESWC2009), Semantic Web in Use Track*, 2009.

[12] M. Hausenblas, W. Halb, Y. Raimond, and T. Heath. What is the size of the Semantic Web. In *Proceedings of I-Semantics 2008, Graz, Austria*, 2008.

[13] M. Hausenblas, W. Slany, and D. Ayers. A performance and scalability metric for virtual RDF graphs. In *3rd Workshop on Scripting for the Semantic Web (SFSW07)*, Innsbruck, Austria, 2007.

[14] HP Lab. TDB/Optimizer. `http://jena.hpl.hp.com/wiki/TDB/Optimizer`, 25 October, 2008. Accessed in March 2009.

[15] I. Jacobs and N. Walsh. Architecture of the World Wide Web, Volume One. W3C Recommendation 15 December 2004, W3C Technical Architecture Group (TAG), 2004.

[16] J. Klump, R. Bertelmann, J. Brase, M. Diepenbroek, H. Grobe, H. Höck, M. Lautenschlager, U. Schindler, I. Sens, and J. Wächter. Data publication in the open access initiative. *Data Science Journal*, 5:79–83, 2006.

[17] G. Klyne, J. J. Carroll, and B. McBride. RDF/XML Syntax Specification (Revised). W3C Recommendation, RDF Core Working Group, 2004.

[18] P. Mika. Microsearch: An interface for semantic search. In *Semantic Search, International Workshop located at the 5th European Semamntic Web Conference (ESWC 2008)*, volume 334 of *CEUR Workshop Proceedings*, pages 79–88. CEUR-WS.org, 2008.

[19] P. Miller, R. Styles, and T. Heath. Open data commons, a license for open data. In *Proceedings of the Workshop on Linked Data on the Web (WWW2008)*, 2008.

[20] M. Nottingham. Link relations and HTTP header linking. Internet-Draft, 1 December 2008, IETF Network Working Group, 2008.

[21] M. Nottingham and E. Hammer-Lahav. Host metadata for the Web. Internet-Draft, 10 February 2009, IETF Network Working Group, 2009.

[22] B. Quilitz and U. Leser. Querying distributed RDF data sources with SPARQL. In *Proceedings of the 5th European Semantic Web Conference 2008)*, pages 524–538. Springer, 2008.

[23] G. Tummarello, R. Delbru, and E. Oren. Sindice. com: Weaving the open linked data. *Proceedings of the 6th International Semantic Web Conference 2007 (ISWC2007)*, 4825:552–565, 2007.

[24] G. Wachob, D. Reed, L. Chasen, W. Tan, and S. Churchill. Extensible Resource Identifier (XRI) Resolution Version 2.0. Committee Draft 03 28 February 2008, OASIS eXtensible Resource Identifier (XRI) TC, 2008.

[25] S. Weibel, A. S. for Information Science, and Technology. The Dublin Core: A simple content description model for electronic resources. *Bulletin of the American Society for Information Science and Technology*, 24(1):9–11, 1997.

[26] J. Zhao, A. Miles, G. Klyne, and D. Shotton. Linked data and provenance in biological data webs. *Briefings in Bioinformatics*, 2008.

[16] `http://www.ict-romulus.eu/`

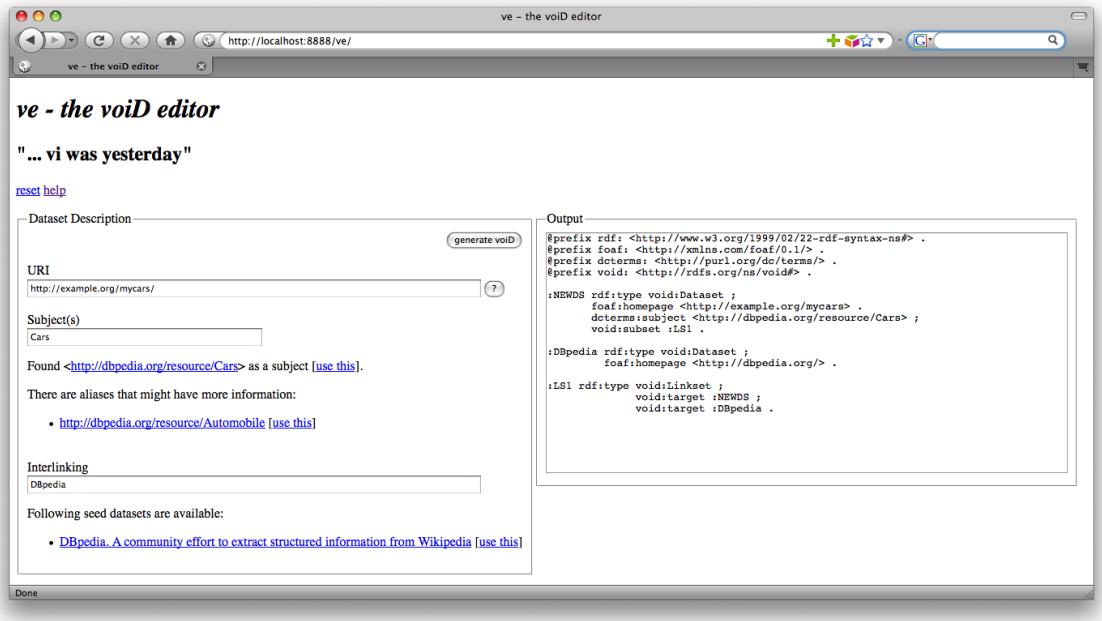
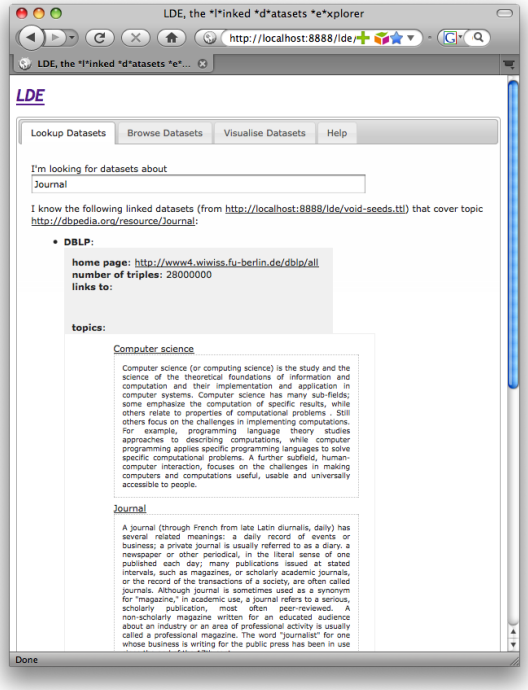
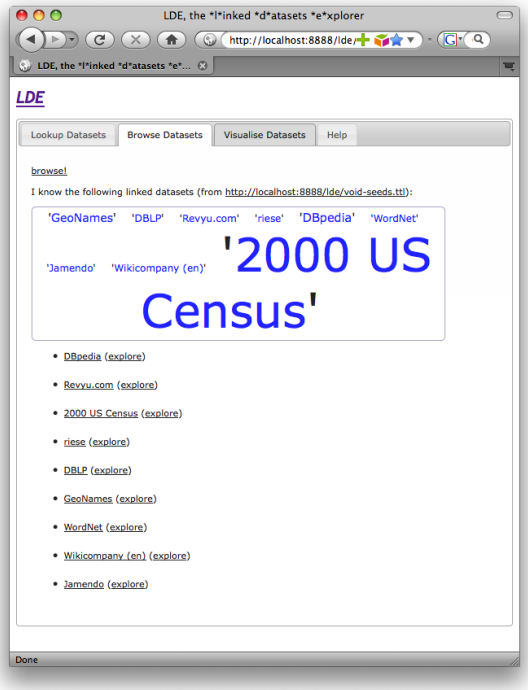


Figure 5: Manual creation of voiD descriptions with *ve*.



(a) Looking-up datasets.



(b) Browsing datasets.

Figure 6: Linked Datasets Explorer (LDE).

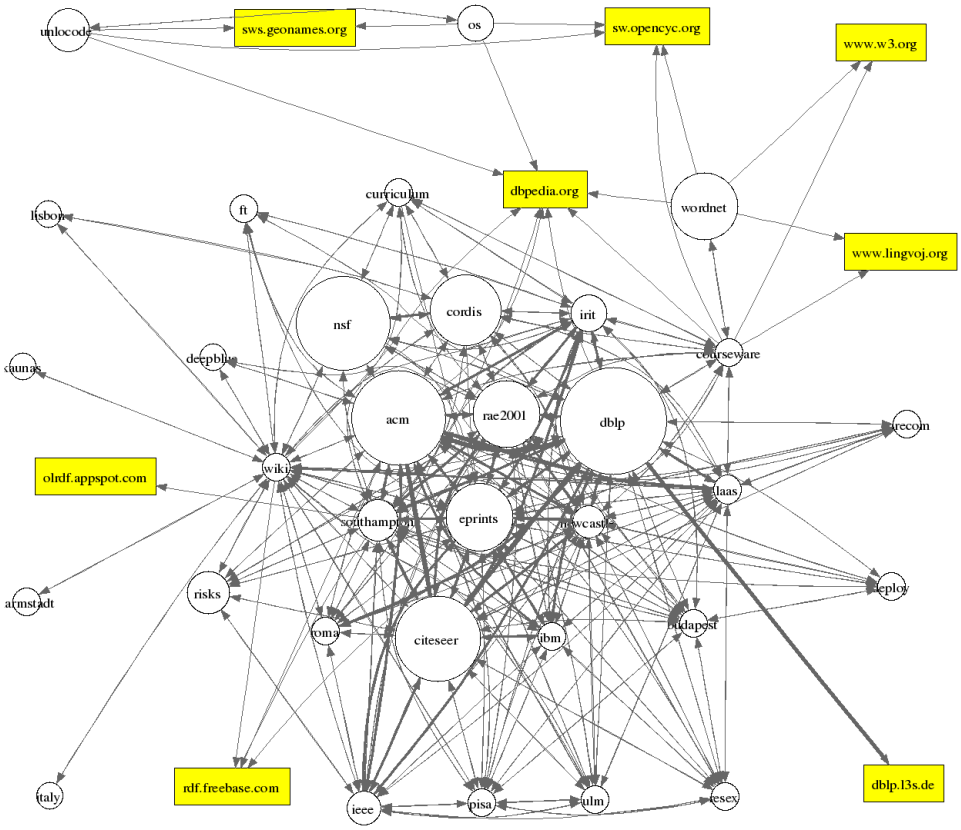


Figure 7: Visualisation of RKB interlinking.

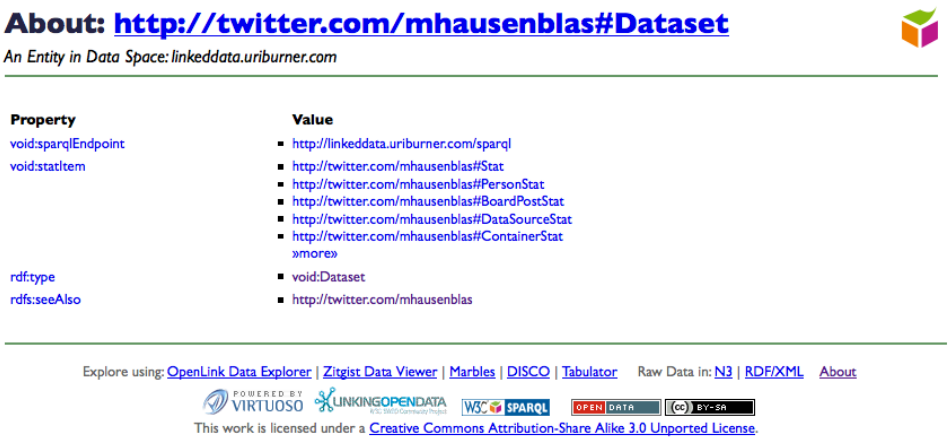


Figure 8: OpenLink's instant-voiD-generator for structured HTML pages.