Undefined 1 (2014) 1–5
IOS Press

From datasets to datanodes: An Ontology Pattern for networks of data artifacts

Enrico Daga [a], Mathieu d'Aquin [a], Aldo Gangemi [b] and Enrico Motta [a]
[a] *Knowledge Media Institute, The Open University, Walton Hall, Milton Keynes, Buckinghamshire MK7 6AA United Kingdom*
E-mail: {enrico.daga,mathieu.daquin,enrico.motta}@open.ac.uk
[b] *Istituto di Scienze e Tecnologie della Cognizione (ISTC), Consiglio Nazionale delle Ricerche, Via S. Martino della Battaglia 44, 00185 Roma (RM)*
Italy
E-mail: aldo.gangemi@cnr.it

Abstract. Data is at the center of current developments on the World Wide Web. In the Semantic Web, different kinds of data artifacts (datasets, catalogues, provenance metadata, etc.) are published, exchanged, described and queried every day. Data hubs are also emerging in the context of the web of Linked Data, as a way to manage this heterogeneity. There are a number of use cases related to data hub management that can only be addressed if we are able to specify the relations between the managed data artifacts in a way that support useful inferences. This includes the understanding of how the features of the data artifacts propagate. This may not be trivial if we consider complex relations, possibly including datasets in different repositories or data flows happening in independent processes and workflows.

We propose an abstract, foundational model which focuses on the graph of relationships between generic "data objects" (which we call *datanodes*). Following an ontology building methodology based on the analysis of Semantic Web applications, we devise a foundational Ontology Design Pattern by collapsing different types of data objects together, and by remodelling structured relations to simple binary relations. Our pattern represents *a datanode related with a datanode*, where the relation can be specified in six fundamental ways.

We extend this foundational model and propose a conceptual framework designed to express relations between data nodes, implemented as an extendable OWL ontology covering the possible relationships between such data nodes. We show how the Datanode approach can support the management of (possibly distributed and interlinked) catalogues of web data and reasoning over the relationships between items in data catalogues.

Keywords: Ontology Design Pattern, Data Hubs, Linked Data, Knowledge Engineering, Data Integration, Data Documentation

1. Introduction

Data are at the center of current developments on the Semantic Web, where different kinds of data are published, exchanged, described and queried every day. In particular, there is a growing effort towards publishing and using a number of repositories, catalogues and registries to support the discovery and reuse of Semantic Web data. Data hubs are emerging as a mean to to manage this heterogeneity [1,2,3,4,5,6,7]. Data artifacts play different roles in this environment. For example they can be data files, query results, ontologies, meta level descriptions like VoID files [8], provenance bundles [9] or catalogues. Schemas aimed at the description of the different roles of these data artifacts have been developed: VoID [8], DCAT [10] or Prov-O [11] to name just a few. There are a number of use cases related to data hub management that can only be addressed if we are able to specify the relations between the managed data objects in a way that useful inferences are supported. This includes the understanding of how the features of the data propagate to related items. This may not be trivial if we consider complex relations, possibly

including datasets in different repositories or data flows happening in independent processes and workflows.

In this article we describe an Ontology Design Pattern (ODP), which generalizes the relations between these "data objects", which we call *datanodes*. Starting from the analysis of Semantic Web applications, we realized that many interesting relations can be devised from data flows of applications that, along with the ones expressed by existing vocabularies, can support the analysis, understanding, development and maintenance of data hubs. These relations have been organized in a composite hierarchy under a foundational Content Ontology Design Pattern [12].

We analysed the description of the data flows of semantic web applications to observe the rationales behind these data flows. We observed that these systems have a strong focus on data nodes (datasets, ontologies, catalogues, etc.) that may emerge as input or output at different steps of the process and/or layers of an application's architecture. We decided not to represent them in terms of tasks and processes but to describe them as a graph connecting nodes of data, as a means to specifying the possible relations between them. As a result, a foundational pattern emerged, where complex structures are collapsed to a binary relation. The resulting representation is a direct graph where the nodes are the data and the links their relations. We named the single class *Datanode*. Our pattern represents *a Datanode related with a Datanode* in six fundamental ways, as shown in Figure 1.

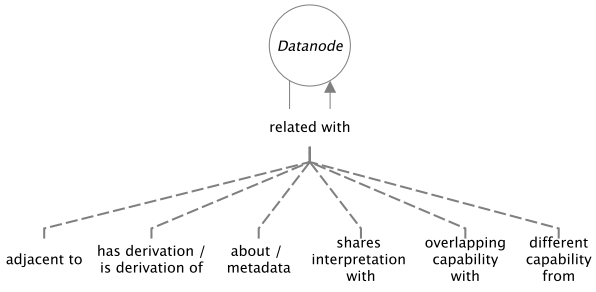

Fig. 1. The Datanode Ontology Design Pattern.

The article is structured as follow. The next section is dedicated to the genesis of the pattern. In Section 3 we discuss related vocabularies and modeling practices, in particular the relation of the Datanode Pattern with existing Semantic Web ontologies. In Section 4 we describe the Ontology Pattern in detail, and we organize the hierarchy of relations that specialize it in the Datanode Ontology. In this section we also report about alignments with Semantic Web ontologies as well as about basic reasoning properties of the ontology. In Section 5 we describe a scenario of Data Hub management, where we believe this approach is desirable to enhance existing solutions. We show how the Datanode approach can provide a method to reason on some common properties of complex dataflows. Finally, Section 6 will report on other possible applications of the pattern and discusses future work.

2. Genesis of the pattern

The research question that motivate our work is: Is it possible to devise generic conceptualisation to describe the heterogeneous data artifacts that compose the Semantic Web? Our methodology to build such a conceptualisation therefore starts from a survey of existing Semantic Web Applications. Semantic Web Applications can be considered as systems operating in the World Wide Web that follow principles or incorporate technologies of the W3C Semantic Web such as RDF and other standards [13]. We restrict this pragmatic definition by considering as Semantic Web Applications systems that, in addition have all the following characteristics:

- Make use of information from sources they do not control;
- Manipulate it to produce new knowledge;
- Process the resulting data for presentation in a new form.

Our methodology is summarised as follow:

1. **Selection**. Select applications described in recent contributions to Semantic Web conferences by using the definition given above.
2. **Acquisition**. Isolate the descriptions of each application from the related paper.
3. **Use case abstraction**. Generalise the description by translating it to a generic synopsis.
4. **Design**. Annotate the synopsis with names of classes and properties that appear in the text; generate a draft ontology and then refine it through discussions.
5. **Testing**. Model the synopsis using the ontology. Testing was done in parallel with the design phase, until all cases were fully covered.

In the following sections we report on the process that led to the Datanode Ontology Pattern. As illustration, we will follow the evolution of a snippet extracted

from one of the system analysed, namely DBRec [14]. DBRec is a music recommendation system built using DBPedia. The paper [14], among other aspects, describes the preparation of the data done to maximise the efficiency of the Linked Data Semantic Distance algorithm.

2.1. Selection & acquisition

We selected six applications described in recent contributions to Semantic Web conferences that fit well our definition of Semantic Web Application: Aemoo [15], DBRec [14], DiscOU [16], Spud [17], Yokohama Art Spot [18], EventMedia [19]. All these are applications that do not start from a dedicated dataset, but transform some existing sources to achieve a target task. We started by isolating the portions of text describing the system. When the system was including several different features and descriptions spread over many points in the document, we isolated all of the snippets and merged them in a unique picture at a later stage. This example is extracted from the discussion of a data preparation process in DBRec:

"On the other hand, we identified lots of redundancy and inconsistencies in our DBpedia subset. Especially, many links between resources are defined redundantly as http://dbpedia.org/ontology/xxx and at the same time as http://dbpedia.org/pro-perty/xxx. We then removed duplicates, leading to 1,675,711 triples, i.e. only 55.7% of the original dataset."

2.2. Use case abstraction

The aim of this activity was to try and extract generic use cases from specific ones. We went through each description and translated it into a list of atomic sentences, each describing a step of the process (or a specific feature). We obtained a synopsis from each text snippet. During this process we tried to abstract from the specific examples, replacing peculiar aspects with their generalisation, when possible. We obtained a set of synopses, covering several aspects of Semantic Web Applications, similar to the following:

1. having data with redundancies and inconsistencies
2. remove redundancies and remove inconsistencies
3. remove duplicates (same as redundancies?)
4. data is now 55.7 percent of before

As a result of this process we obtained a list of text snippets describing system features, each coupled with a synopsis. The whole set of synopses constituted our modelling requirements[1].

2.3. Ontology design

We traversed the collection of synopses and treated each sentence on its own. We annotated each sentence with a set of keywords. We then looked at the keywords and modified them in order to make them represent either a verb or a type of things. In the second case, we wrote it capitalised:

0) access, Data, Redundancy, Inconsistency
1) remove, Redundancy, Inconsistency
2) remove, Duplicate
3) Data, Now, Before

We traversed all synopses several times, trying to harmonise the concepts (e.g., merging synonyms) while maintaining the specificities emerging from each case.

We then collected all 159 keywords from our annotations, analysed them trying to reduce redundancy (for example normalising names to be singular). The result was a list of names of possible concepts and relations. We automatically generated a draft ontology considering all concepts to be istances of `owl:Class` and all relations to be object properties between (yet) unspecified things. This draft ontology included 132 classes and 168 properties, covering several different aspects of a Semantic Web Application. Here is a sample list: Operation, Collect, Optimize, Capability, newer-version-of, not-fits, before, Entity, Dataset, about, Summarize, component-of, Stand-in, concern-of, derive-from. This phase required to abstract and simplify this automatic, disorganised and heterogeneous ontology. This was based on three main observations:

1. Many of the types represented data sources or objects in various forms;
2. Many of the processes could be represented through the relationships between their input data and their output data.
3. We can organize the relationships hierarchically, and infer abstract notions from specific relations

With this foundation established, we then refactored all the elements obtained in the previous steps so that

[1] We do not explicitely formulated these requirements as Competency Questions (CQ) [12]. Nevertheless, they had the role of competency questions in our methodology, guiding the design and testing of the ontology.

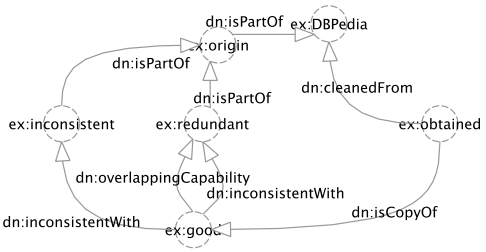

Fig. 2. Example test case, from DBRec [14]. The graph shows how the process that goes from the DBPedia dataset to an optimized and curated dataset can be formalised with Datanode. The datanode *ex:good* is the result of the whole process, it is described as a copy of the datanode obtained as a result of a data cleaning of DBPedia targeted to the task (right side of the picture). In other terms, only a subset of DBPedia has beed considered - *ex:origin*, the corrupted part *ex:inconsistent* has been left out (is not compatible with the result: *dn:inconsistentWith*), and that the result has overlapping capabilities with a *ex:redundant* part that has been also left out from the result (the centre of the picture). We will detail these and other relations in Section 4.

they are either abstracted into the class Datanode, or modelled as properties of the class Datanode. We formalized each relation and organized them in a hierarchy, testing the changes with the available use cases iteratevely. Figure 2 shows one of these test cases. We refined the property hierarchy specifying ontological constraints - symmetricity, transitivity, etc. - when appropriate. We grouped all relations under the top property `relatedWith` (we will discuss the pattern in depth in Section 4).

2.4. Testing

We iteratively modelled all synopsis with the hierarchy of relations at hand at any stage, and verified that the inferred model was fitting the intended meaning of the description. For testing the ontology we considered two more applications, namely Rexplore [20] and IBM Watson [21], to strenghten our evaluation. On each test iteration we descovered new implications derived from the organization of the properties as a composite hierarchy (in the spirit of the Extreme Design [12] methodology).

2.5. Result

The foundation of the Datanode Ontology is therefore devised as a simple abstraction including a unique class, Datanode, which represents “data” in a broad sense (data sources of any kind or format), and formalised relationships between these datanodes. The Datanode Ontology Design Pattern is the top layer of the property hierarchy, showing six fundamental aspects through which relate datanodes: metalevel, derivation, overlapping capability, different capability, shared interpretation and adjacency (see also Figure 1).

Before describing it in detail, it is useful to contextualize our proposal in the current state of the art.

3. Related vocabularies and modeling practices

The literature on the representation of data artifacts is too wide to be fully covered in the scope of this article. Technologies and standard for data representation and annotation have a long tradition in library cataloguing and interoperability (for example in the context of the Open Data Archive initiative[2]) and in large part they contributed to the current standards of the Semantic Web (the Dublin Core Metadata Initiative is the most prominent case[3]). Our work aims to contribute to the discussion about the improvement of the methods for managing Semantic Web data. For this reason we will not discuss data representation languages and formats outside the Semantic Web. We are aware that there are numerous formalisms that could be compared to our work in the fields of Software Engineering and Data Integration, just to name two. However, we consider this analysis as an interesting future work.

The are a number of Semantic Web ontologies that share the same domain as Datanode, and have contributed to the genesis of the pattern. The PROV data model [22] and ontology [11] are W3C recommentations designed to describe information *"about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness"*[4]. VoID is the "Vocabulary of Interlinked Datasets" [8]; it has been proposed to describe several aspects of datasets, from general metadata (inheriting Dublin Core terms) to access methods, internal organization (partition) and linking. The Data Catalog Vocabulary (DCAT) is a W3C recommendation for the description of data catalgues [10]. DCAT defines a dataset as a collection that is published and curated by an organization that provides access to it in one or more formats. Examples of data catalogues using DCAT are

[2] `http://www.openarchives.org/`
[3] `http://dublincore.org/`
[4] `http://www.w3.org/TR/2013/REC-prov-dm-20130430/`

data.gov.uk[5] and the Open Government Dataset Catalog [5]. The RDF Data Cube Vocabulary [23] is the W3C recommendation for the publication of multidimensional data tables, mainly in the domain of statistical data. Its scope is on providing a schema to publish data, not a method to describe datasets. However, it includes interesting concepts for Datanode, like a method to qualify the structure of portions of datasets. VOAF [24] and VANN[6] [25] have been designed to describe vocabularies in the Linked Data cloud [7]. The FOAF vocabulary[8] contains maybe the first "metadata to data" relation: `foaf:primaryTopic` (and `foaf:primaryTopicOf`) of the Semantic Web.

All these Semantic Web Ontologies have been taken into account while designing Datanode. Their focus is to provide a reusable schema for the description of certain classes of data objects, being data tables, vocabularies, datasets, catalogues or RDF documents. Datanode tries to design a general conceptualization of the relations beween data artifacts. In Section 4 we discuss how all of these ontologies align with Datanode.

The DOOR ontology was implemented to represent the way Semantic Web ontologies can be related each other in ontology repositories [26]. Datanode is meant to address a very similar problem, but considered in a wider scenario.

The Asset Description Metadata Schema (ADMS) [27] is an RDF vocabulary proposed to be a common representation and exchange format of e-Government repositories. The Data Documentation Initiative has developed a metadata specification for the social and behavioral sciences [28]. The intent is to describe data sets resulting from social science research (like survey data). An RDF version of the vocabulary has been recently developed[9] to foster the publishing into the Web of Linked Data. ADMS and DDI provide reusable datasets as well as reusable metadata models and standards, while our contribution has at his core the analysis and organization of the possible relations between data artifacts.

Research about workflow descriptions and provenance strongly relate with our work, especially if considering the methodology we used to build the ontology, starting from the analysis of data flows. For example, [29] define a set of use cases for provenance modeling from which they generalize a number of abstract requirements, including the value of making explicit the intent of the agents that make part to an activity, that semantically affects the result. The Datanode branch related to derivation of datasets can be also considered as an attempt to make explicit the possible semantics of data manipulations. In [30] the authors develop a vocabulary to annotate data with provenance information, including metadata about the process, involved software and provider. The goal is to support the selection of linked data sources based on trustworthiness. Datanode can be used as a complement to provenance, to represent and reason on what are the implications of these aspects in terms of the relationships between the data artifacts.

Workflow description is also part of the research on Semantic Web Services [31] and on Linked Stream Data processing [32]. Workflow-centric Research Objects [33] have been designed with the objective of make persistent (and reproducible) research experiments in the scientific discourse. Workflows are made of agents, tasks, processes and methods. In our work the goal is to devise possible relations between data objects from data flow descriptions, this can be a mean to extend the inference capabilities on workflow models (like provenance information, as we will show in Section 5).

Our generalization can be seen as a method to reduce a composite element of a workflow description (representing an event or an activity, for example) to a (set of) binary relation between the data items that have a role in it. In [34] the author proposes a method to collapse an N-Ary relation to a set of binary relations trying to preserve the same semantic value. However, we do not intend to preserve the information of the activity in full, we just collapse it to express the relation between the nodes. Our model can leave out a number of entities in favour of making explicit the relations between data artifacts. An example of a similar simplification is in [35], where authors reduce the expressiveness of a large dataset of CAD models in an ontology for the sake of supporting more effectively the task of reviewing in industrial design.

The approach of abstracting to optimize a given process can be comparable to the documentation of software architectures [36]. The literature in this field emphasises the multiplicity of structures existing in information systems and how abstracting helps the understanding and reasoning over complex models.

[5] `http://data.gov.uk`

[6] `http://purl.org/vocab/vann/`

[7] See the Linked Open Vocabularies activity: `http://lov.okfn.org/dataset/lov/`.

[8] `http://xmlns.com/foaf/spec/`.

[9] `http://rdf-vocabulary.ddialliance.org/discovery.html`.

Table 1

Competency questions mapped to abstract relations in Datanode. Each CQ should be read considering a given Datanode as focus.

Aspect	CQ: Given a Datanode...
Metalevels	Which are the datanodes that contain metadata about it? Which datanodes are described in it?
Derivation	Which datanodes has been produced by activities that used it as input? Is it the result of a manipulation, observation or processing of another datanode?
Adjacency	Which node shares the same container? Which nodes are part of the same catalogue? Which datanodes are part of the same datanode?
Capabilities (overlap)	Which node shares a feature with it?
Capabilities (different)	Which node has different features?
Shared interpretation	Which datanode can have a role on affecting the inferences that can be derived?

Datanode is a Content Ontology Pattern [12], its domain of application is data catalogue description and management. Our design methodology has been iterative, incremental and test-driven, and is strongly inspired by Extreme Design in ontology engineering [37].

4. The Datanode Ontology Pattern

The Datanode ODP is the result of a design issue. We found data to have several different facets in the systems' descriptions, examples include: sets of data, data sources, identifiers, metalevel descriptions. The Datanode class abstracts from the notion of dataset and it is independent from the notions of containment, individual or identifier.

A *datanode* is any data artifact. The class groups datasets, ontologies, schema elements such as classes and properties, as well as identifiers under the same umbrella. The nature of this concept is voluntarly underspecified. The ontology pattern defines a unique type - `Datanode` - and six top relations, starting from a single top property: `relatedWith`, having Datanode as domain and range, as shown in Figure 1. The Datanode terms are under the following namespace:

`http://purl.com/datanode/ns/`

The Datanode ODP can be specialized by extending the six "branches", thus expressing more specific relations. Table 1 lists a number of abstract competency questions that illustrate how the pattern can be specialized. In the following sections we describe the top relations and the specialized relations that compose the Datanode Ontology [10]. Tables will list the relations, their OWL type and the relations to which they are `rdfs:subPropertyOf`. Table 2 lists the symbols used to indicate OWL property types.

Table 2

Legend of OWL 2 Property types used in Tables 3-8

Symbol	OWL Property type
T	TransitiveProperty
S	SymmetricProperty
R	ReflexiveProperty
F	FunctionalProperty
I	InverseFunctionalProperty
X	IrreflexiveProperty

Table 3

Properties in the Metalevels branch

relation	sub property of
metadata	relatedWith
about	relatedWith
describedBy	metadata
describes	about
isAnnotationOf	about attachedTo
hasAnnotation	hasAttached metadata
hasStatistic	describedBy hasComputation
isStatisticOf	describes isComputationOf

4.1. Metalevels

This branch covers the relations between something and its metadata. The property `metadata` is used to designate a relation with information that applies to the datanode as a whole, possibly including it. This relation has for inverse `about`.

This kind of relations specialises as `describes` / `describedBy`, `hasAnnotation` / `isAnnotationOf` and `hasStatistic` / `isStatisticOf` (we will follow this convention to pair relations that are inverse of each other).

Table 3 shows the structure of this branch[11].

4.2. Derivations

This relation indicates that a datanode is the origin of another, in the sense that the second has been produced using the first as information source. It is possible that a source `hasInterpretation` / `isInterpretationOf` some other data, as a result of a mining process -

[10] Documentation is online at `http://purl.org/datanode/docs/`.

[11] The reader will note that many relations fall under more then a branch (for example, the relation `hasStatistic` is a specialization of `describedBy` as well as of `hasComputation`).

hasExtraction / isExtractionOf; or a logical process - hasInference / isInferenceOf. A relation processedFrom / processedInto is identified when a derivation is also the transformation of a data node into another one, being it cleaned, optimized, refactored, remodelled or the result of a computation - hasComputation / isComputationOf. It is possible to derive a new data node by making a copy - hasCopy / isCopyOf, when making snapshots or caching strategies (hasSnapshot / isSnapshotOf, hasCache / isCacheOf). We consider also the case when a derivation is done by cloning a part of a datanode with specific features - hasSelection / isSelectionOf.

Table 4 shows the full list of relations that are part of this branch.

4.3. Adjacencies

The top relation adjacentTo represents proximity between two datanodes in a data container (catalogue or dataset). Proximity may result from being parts of the same dataset - disjointPartWith. Another case of proximity is between an object and its attachment - hasAttached / attachedTo. The hasAnnotation / isAnnotationOf relation, mentioned above, specialises also hasAttached / attachedTo. Table 5 shows the implications in this branch.

4.4. Capabilities (overlapping and different)

Capability is intended as *"the power or ability to generate some outcome"*[12]. Similarly to consistency, capability is covered with two separate branches starting from overlappingCapabilityWith and differentCapabilityFrom respectively. Two data nodes may have similar potential. This may refer to any kind of feature, be it structural (e.g., they share schema elements), physical (e.g., they are both in XML) or related to the domain (they both talk about Music Artists) - to name but a few examples. This relation is intentionally left abstract since there might be many different ways to express capabilities. Extensions for specific use cases can of course be

[12]Definition from http://en.wiktionary.org/wiki/capability.

Table 4
Properties in the derivation branch

relation	sub property of
isDerivationOf	relatedWith
hasDerivation	relatedWith
isSummarizationOf	isDerivationOf
hasStandIn	hasDerivation overlappingCapability-With
processedInto	hasDerivation
combinedIn	hasDerivation
hasSummarization	hasDerivation
combinationFrom	isDerivationOf
hasCopy T	hasDerivation sameCapabilityAs
hasSelection	hasDerivation hasPart
isStandInOf	isDerivationOf overlappingCapability-With
isCopyOf T	isDerivationOf sameCapabilityAs
hasInterpretation	hasDerivation
isSelectionOf	isDerivationOf isPartOf
isInterpretationOf	isDerivationOf
processedFrom	isDerivationOf
refactoredFrom	processedFrom
hasCache	hasSnapshot hasStandIn
hasExample	hasSelection
isInferenceOf	isInterpretationOf
cleanedInto	overlappingCapabilityWith processed-Into
hasReification	processedInto
hasComputation	processedInto
isAnonymizedOf	isStandInOf processedFrom
cleanedFrom	overlappingCapabilityWith processed-From
remodelledFrom	processedFrom samePopulationAs
hasAnonymized	hasStandIn processedInto
hasSnapshot	hasCopy versionOf
isExtractionOf	isInterpretationOf
isSnapshotOf	isCopyOf versionOf
refactoredInto	processedInto
isCacheOf	isSnapshotOf isStandInOf
hasExtraction	hasInterpretation
hasInference	hasInterpretation
isComputationOf	processedFrom
remodelledTo	processedInto samePopulationAs
isExampleOf	isSelectionOf
optimizedInto	overlappingCapabilityWith processed-Into
optimizedFrom	overlappingCapabilityWith processed-From
isReificationOf	processedFrom
hasStatistic	describedBy hasComputation
isStatisticOf	describes isComputationOf

Table 5
Properties in the adjacency branch

relation	sub property of
adjacentTo S T	relatedWith
disjointPartWith S	adjacentTo
attachedTo	adjacentTo
hasAttached	adjacentTo
isAnnotationOf	about attachedTo
disjointPortionWith S	differentPopulationFrom disjoint-PartWith
disjointSectionWith S	differentVocabularyFrom disjoint-PartWith
hasAnnotation	hasAttached metadata

made as specialisations of this relation. The Datanode Ontology therefore only contains a few general cases: `overlappingVocabularyWith` and `overlappingPopulationWith`, both leading to `redundantWith`, `sameCapabilityAs` and `duplicate` - all describing a similar phenonmenon with different intentions. Under this scope we also positioned `optimizedFrom` / `optimizedInto` - to state the empowerment of an existing capability; and `cleanedFrom` / `cleanedInto` - to represent the result of a process aimed to make emerge a potential capability whilst maintaining another fundamental one.

In contrast, two datanodes may have different potential. For example, two data nodes use different vocabularies (`differentVocabularyFrom`, or `disjointSectionWith`, when parts of the same datanode) or have different population (`differentPopulationFrom`, or `disjointPortionWith`). When both vocabularies and populations are different it is possible that two datanodes have disjoint capabilities - `disjointCapabilityWith`.

Having an overlapping population can be the implication of a part-whole relation: `hasPart` / `isPartOf`. A distinction is done via `hasSection` / `isSectionOf` and `hasPortion` / `isPortionOf`. We call *section* the partition of a datanode by the means of a set of attributes (with no information about its population), while *portion* is a partition done by keeping a part of the population (with no information about the attributes). In the extreme case, hasSection can be further specialised into `hasIdentifiers` / `identifiersOf`. A portion can also be a sample - `hasSample` / `isSampleOf`. Deriving by selection is also a way of isolating a part of the source, thus `hasSelection` / `isSelectionOf` appears also on this branch, with the sub-property `hasExample` / `isExampleOf`.

Overlapping capability may be implied by being a *version*. This property implies a temporal relation between two data nodes that are meant to be the same at a different point in time. We find under this category relations such as `newerVersionOf` / `olderVersionOf` and `nextVersionOf` / `previousVersionOf`.

`versionOf` itself is symmetric and does not specify a direction. It is not transitive. While it can be argued that the identity of something tracked over time should not change, thus implying transitivity, we want to support the case when a datanode has more than one single following version (branching).

Table 6
Properties in the overlapping capabilities branch

relation		sub property of
overlappingCapabilityWith	S	relatedWith
cleanedInto		overlappingCapabilityWith processedInto
cleanedFrom		overlappingCapabilityWith processedFrom
hasStandIn		hasDerivation overlappingCapabilityWith
overlappingPopulationWith		overlappingCapabilityWith sharesInterpretationWith
overlappingVocabularyWith		overlappingCapabilityWith sharesInterpretationWith
isStandInOf		isDerivationOf overlappingCapabilityWith
versionOf	S	overlappingCapabilityWith
optimizedInto		overlappingCapabilityWith processedInto
optimizedFrom		overlappingCapabilityWith processedFrom
hasCache		hasSnapshot hasStandIn
isAnonymizedOf		isStandInOf processedFrom
hasAnonymized		hasStandIn processedInto
isPartOf	T	overlappingPopulationWith
hasSnapshot		hasCopy versionOf
samePopulationAs	T	overlappingPopulationWith
links		overlappingPopulationWith references
isSnapshotOf		isCopyOf versionOf
redundantWith	T	overlappingPopulationWith overlappingVocabularyWith
linkedBy		overlappingPopulationWith referencedBy
sameVocabularyAs	T	overlappingVocabularyWith
newerVersionOf	T	versionOf
isCacheOf		isSnapshotOf isStandInOf
olderVersionOf	T	versionOf
hasPart	T	overlappingPopulationWith
hasExample		hasSelection
remodelledFrom		processedFrom samePopulationAs
hasUpdatedVersion		hasUpdate previousVersionOf
identifiersOf		isSectionOf
nextVersionOf	F	newerVersionOf
previousVersionOf	I	olderVersionOf
hasIdentifiers		hasSection
isSampleOf		isPortionOf
hasPortion	T	hasPart
hasSection	T	hasPart
isUpdatedVersionOf		isUpdateOf nextVersionOf
hasCopy	T	hasDerivation sameCapabilityAs
hasSelection		hasDerivation hasPart
hasSample		hasPortion
sameCapabilityAs	T	samePopulationAs sameVocabularyAs
isPortionOf	T	isPartOf
remodelledTo		processedInto samePopulationAs
isCopyOf	T	isDerivationOf sameCapabilityAs
isSectionOf	T	isPartOf
isExampleOf		isSelectionOf
isSelectionOf		isDerivationOf isPartOf
duplicate	S T X	sameCapabilityAs

Table 6 shows the implications in the overlapping capabilities branch, while Table 7 the ones in the difference capabilities branch.

Table 7
Properties in the different capabilities branch

relation		sub property of
differentCapabilityFrom	S	relatedWith
differentPopulationFrom	S	differentCapabilityFrom
differentVocabularyFrom	S	differentCapabilityFrom
disjointPortionWith	S	differentPopulationFrom disjointPartWith
disjointSectionWith	S	differentVocabularyFrom disjointPartWith
disjointCapabilityWith	S	differentPopulationFrom differentVocabularyFrom

4.5. Shared interpretation

This branch is dedicated to the possibility that datasets might share the same interpretation. This is designed to capture the possibility that a datanode might contribute to infererences that can be made in another one. Two datanodes *might* be "understood" together, i.e. its knowledge can be compared, or the interpretation (inferences) of one may affect the interpretation (inferences) of another. The top relation `sharesInterpretationWith` is transitive and symmetric. Table 8 shows the implications in the shared interpretation branch. The explanations of the reasons why the immediate sub-relations imply a shared interpretation are summarized in Table 9, and discussed in the following part of this section.

Being (in)consistent. A shared interpretation obviously derives when we describe two datanodes to be *consistent* or *inconsistent* with each other. This aspect is covered by two properties: `consistentWith` and `inconsistentWith`. We intend (in)consistency in a broad sense: two datanodes are (in)consistent because they are (not) compatible in some fundamental respect. For example, `(in)consistentWith` may be used to indicate a data node that should (not) be used together with another. In some settings, it may be useful to specify that some datanodes are consistent with others, eventually to state that a datanode is `consistentWith` itself. It is also possible that a datanode is `inconsistentWith` itself, meaning that it is wrong, buggy or somehow corrupted.

Being an update. A data node may be related to another because it contributes to improve its validity or currency. `hasUpdate` (and its inverse `isUpdateOf`) has this role. This property can be specialized to indicate a datanode that is meant to substitute the original: `hasUpdatedVersion` / `isUpdatedVersionOf` (that are also under the `hasVersion` umbrella). While the updating (as activity) can be seen as a way of versioning, the two concepts are different when modelled as relations between datanodes. An

Table 8
Properties in the shared interpretation branch

relation		sub property of
sharesInterpretationWith	S T	relatedWith
isVocabularyOf		sharesInterpretationWith
inconsistentWith	S	sharesInterpretationWith
overlappingPopulationWith		overlappingCapabilityWith sharesInterpretationWith
consistentWith	S	sharesInterpretationWith
hasVocabulary		sharesInterpretationWith
overlappingVocabularyWith		overlappingCapabilityWith sharesInterpretationWith
references		sharesInterpretationWith
referencedBy		sharesInterpretationWith
hasUpdate		sharesInterpretationWith
isUpdateOf		sharesInterpretationWith
schemaUsedBy		referencedBy
descriptorsOf		isVocabularyOf
hasDatatypes		hasVocabulary
hasUpdatedVersion		hasUpdate previousVersionOf
isPartOf	T	overlappingPopulationWith
samePopulationAs	T	overlappingPopulationWith
links		overlappingPopulationWith references
hasTypes		hasVocabulary
isChangeOf	F	isUpdateOf
isUpdatedVersionOf		isUpdateOf nextVersionOf
redundantWith	T	overlappingPopulationWith overlappingVocabularyWith
typesOf		isVocabularyOf
datatypesOf		isVocabularyOf
usesSchema		references
linkedBy		overlappingPopulationWith referencedBy
sameVocabularyAs	T	overlappingVocabularyWith
hasChange	I	hasUpdate
isDependencyOf		referencedBy
hasDescriptors		hasVocabulary
hasPart	T	overlappingPopulationWith
hasDependency		references
hasCache		hasSnapshot hasStandIn
hasExample		hasSelection
isDeletionOf		isChangeOf
remodelledFrom		processedFrom samePopulationAs
relationsOf		descriptorsOf
hasDeletion		hasChange
identifiersOf		isSectionOf
hasSnapshot		hasCopy versionOf
hasRelations		hasDescriptors
hasIdentifiers		hasSection
isSampleOf		isPortionOf
attributesOf		descriptorsOf
hasPortion	T	hasPart
isSnapshotOf		isCopyOf versionOf
hasSection	T	hasPart
hasAttributes		hasDescriptors
hasCopy	T	hasDerivation sameCapabilityAs
hasSelection		hasDerivation hasPart
hasAddition		hasChange
hasSample		hasPortion
isCacheOf		isSnapshotOf isStandInOf
sameCapabilityAs	T	samePopulationAs sameVocabularyAs
isAdditionOf		isChangeOf
isPortionOf	T	isPartOf
remodelledTo		processedInto samePopulationAs
isCopyOf	T	isDerivationOf sameCapabilityAs
isSectionOf	T	isPartOf
isExampleOf		isSelectionOf
isSelectionOf		isDerivationOf isPartOf
duplicate	S T X	sameCapabilityAs

Table 9
Relations that imply a shared interpretation.

:isUpdateOf	being an update implies that two nodes can be compared to understand how a specific information evolved/changed
:hasUpdate	similarly, know that some data has an update implies that its validity or timeliness is compromised
:overlapping-PopulationWith	If two data nodes contains the same entities, certainly they might be read together
:overlapping-VocabularyWith	If two data nodes express the information with the same properties or types then they may be combined
:inconsistentWith and :consistentWith	Being (in)consistent is defined as being (in)compatible, thus imply sharing something at a fundamental level.
:references and :referencedBy	If a node references another one, then its interpretation could be affected by the other (and vice versa).
:hasVocabulary and :isVocabularyOf	The understanding of the terms used as properties and types surely affect the interpretation of the data.

update may not replace its target datanode. It is the case when the update is meant to only modify the data node: `hasChange` / `isChangeOf` and its sub-properties `hasAddition` / `isAdditionOf` and `hasDeletion` / `isDeletionOf`.

Being referenced. A datanode includes a mention to something which is another datanode. This branch covers linking (as intended in the context of linked data) - `links` / `linkedBy`; as well as dependencies between data objects (`hasDependency` / `isDependencyOf`) and between data and a schema definition, for example an ontology (`usesSchema` / `schemaUsedBy`).

Being a vocabulary. The range of hasVocabulary is a datanode which enumerates a set of terms that are all used by the subject data node as identifiers (inverse is `isVocabularyOf`), to name structural elements like `hasDescriptors` / `descriptorsOf`. These can be attributes, relations, or to link a datanode to its types - `hasTypes` / `typesOf` or datatypes with `hasDatatypes` / `datatypesOf`. This is different from the case of a relation with a schema (under "references"): a schema is defined independently from the actual data.

4.6. Alignments with Semantic Web ontologies

As also discusssed in Section 3, there exist a number of Semantic Web ontologies that represent aspects of data artifacts taylored to common use cases. Datanode is indeed related to DCAT [10], Prov-O [11], VoID [8] and VOAF [24] by aligning a number of Classes to the class `Datanode`, and then by placing the relations that occur between datanodes in the property hierarchy that extend the pattern. Figure 3 displays a number of classes that we specify to be specializations of Datanode, while

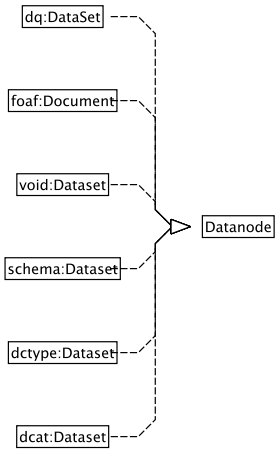

Fig. 3. The alignments to classes of some Semantic Web ontologies. All these classes are sub classes of Datanode.

Table 10
Properties of some Semantic Web ontologies aligned to Datanode relations using `rdfs:subPropertyOf`.

relation	**sub property of**
http://purl.org/dc/terms/references	references
http://purl.org/dc/terms/subject	about
http://purl.org/vocommons/voaf#extends	references
http://purl.org/vocommons/voaf#extends	isDerivationOf
http://purl.org/vocommons/voaf#generalizes	references
http://purl.org/vocommons/voaf#generalizes	isDerivationOf
http://purl.org/vocommons/voaf#metadataVoc	usesSchema
http://purl.org/vocommons/voaf#reliesOn	references
http://purl.org/vocommons/voaf#similar	overlappingCapabilityWith
http://purl.org/vocommons/voaf#specializes	references
http://purl.org/vocommons/voaf#specializes	isDerivationOf
http://purl.org/vocommons/voaf#usedBy	references
http://rdfs.org/ns/void#subset	hasPart
http://rdfs.org/ns/void#target	links
http://rdfs.org/ns/void#vocabulary	usesSchema
http://www.w3.org/ns/dcat#dataset	describes
http://www.w3.org/ns/dcat#record	hasPortion
http://www.w3.org/ns/prov#wasDerivedFrom	isDerivationOf
http://www.w3.org/ns/prov#wasRevisionOf	isUpdatedVersionOf
http://xmlns.com/foaf/0.1/primaryTopic	describes

Table 10 lists alignments between Datanode and the related relations.

The main focus of DCAT is to provide the basic tools to classify datasets in repositories using concepts, pointing to publishing web pages and distinguishing a dataset and its distributions (concrete instances). The property `dcat:record` links a catalogue to an catalogue record that has `foaf:primaryTopic` a given `dcat:Dataset`. The property `dcat:dataset` makes a direct link from the catalogue to each dataset that is described. The Prov-O ontology includes the concepts of derivation and revision, and both are positioned

Table 11
Reasoning properties of Datanode.

Profile	?
OWL 2	✓
OWL 2 DL	✓
OWL 2 RL	x
OWL 2 EL	x
OWL 2 QL	x

in the hierachy of Datanode. Section 5 will show how this connection can have interesting consequences in the inferred model. The VOAF vocabulary enumerate a number of ways datanodes that are schemas can be linked together[13], most of them imply derivation and reference to be in place.

4.7. OWL 2 Profile

Table 11 shows the compatibility of the whole Datanode Ontology with relation to OWL2 profiles. The Datanode Ontology is in the OWL 2 DL profile.

It is not in the OWL-RL profile because uses `owl:IrreflexiveProperty` to express the relation `duplicate`. This is the only axiom that prevents the ontology to fall into the OWL-RL profile. In cases when it is desirable, that axiom can be easily removed. The ontology is not in the OWL-QL profile because of its use of properties of type `owl:TransitiveProperty` (for example, `hasPart` is transitive) and of type `owl:InverseFunctional` (for example, `previousVersionOf`). For similar reasons, it is not in the OWL-EL profile: It includes `owl:inverseOf` (many relations have their inverse declared) and properties of type `owl:SymmetricProperty` (eg, `differentCapabilityFrom`) as well as `owl:FunctionalProperty` and `owl:InverseFunctionalProperty`[14].

[13]It is worth to remember that the meaning we give to Vocabulary in Datanode is being an enumeration of symbols all used in some data. We prefer to call the objects of the VOAF ontology "schemas", or ontologies.

[14]The compatibility with OWL 2 profiles has been tested practically with the service `http://mowl-power.cs.man.ac.uk:8080/validator/`.

5. Scenario

In this section we describe a protypical use case, focused on understanding change propagations in a data hub. A datahub contains a number of datasets owned and managed by different actors. Nevertheless, many datasets "depend" on others in their interpretation, thus need to be reviewed (for example updated) if some change in the features of their reference dataset occur.

The UK Higher Education Statistics Agency publishes the Unistats dataset[15]. Each September HESA publishes a new version of the dataset, with changes in the data schema, containing improvements and new data. HESA updates the Unistats data regularly. The format of the data is an XML serialization based on the Key Information Set (KIS) [38,39].

The LinkedUp Project is a FP7 Support Action "which pushes forward the exploitation and adoption of public, open data available on the Web, in particular by educational organisations and institutions"[16]. The project publishes a catalogue of datasets as Linked Data [3]. For example, the Open Data from the Italian National Research Council (CNR) [40] and from the PROD project[17] are also registered in the same catalogue, as well as many other datasets. Since 2012, the catalogue includes a Linked Data version of the Unistats database (Unistats LD). Unistats LD is a *remodelling* of the Unistats XML published by HESA, using a schema (KIS-LD) that is mostly based on the RDF Data Cube Vocabulary [23] and is itself a remodelling of the original schema (KIS). The PROD dataset, however, contains `owl:sameAs` links to the institutions described in the Unistats LD dataset. The manager of the LinkedUp catalogue regularly pings the HESA web site for newer version of the data, and the software used to translate the information is kept up to date once a new version of the schema is published. The LinkedUp catalogue add a new graph every year, once the information for a new academic year is published, and keep the old data as historical database.

The Open University (OU)[18], like all UK universities, provide its KIS record to HESA, so that it can be combined with the data from other universitities to produce the statistics published as the Unistats dataset. However, the university also wants to use the slice of the Unistats

[15]See `https://www.hesa.ac.uk/unistatsdata`.

[16]See `http://linkedup-project.eu/about/`.

[17]PROD is a directory and monitoring tool for JISC funded projects. See `http://prod.cetis.ac.uk/`.

[18]`http://www.open.ac.uk`

that refer to the OU - Unistats OU. For this reason the *portion* of the latest Unistats LD that is about the OU is selected and published in the Linked Open Data portal of the university[19] [41] aside other institutional datasets, including taught courses and qualifications, among others.

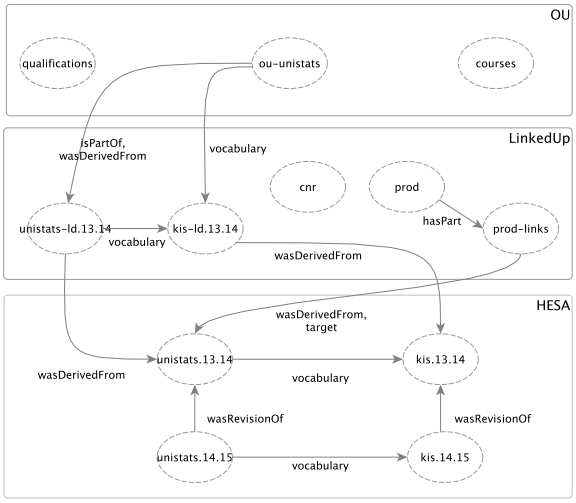

Fig. 4. Summary of the scenario. Relation are expressed using Prov-O and VoID. In this picture we omit the 2012/2013 version of the datasets, for clarity.

To summarize:

- the `unistats` dataset uses `kis` as XML schema
- the `unistats-ld` uses `kis-ld`, both are the RDF remodelling of the first
- `prod` links to `unistats-ld`
- `ou-unistats` is a part of `unistats-ld`
- subsequent versions of `unistats` and `unistats-ld` are archived
- there are other datasets in `linkedup-cat` and `ou-cat`, like `cnr`, `ou-courses` and `ou-qualifications`

The scenario is depicted in Figure 4. Thank to Semantic Web ontologies, we can express the information about the catalogues (see Figure 5), as well as metadata about the data flow. Figure 6 shows the scenario described with the W3C Prov-O ontology and VoID[20]. For clarity, we omit part of the provenance information, for example the type declarations and the statements that relate the datasets to their respective owner using the `prov:wasAttributedTo` property.

The scenario described above contains a number of different actors (HESA, LinkedUp, OU), actions that have dependencies (data updates) and data items that are transformed into others (`unistats`/`unistats-ld`), selected and moved (`ou-unistats`), and linked (`prod` / `unistats-ld`). The whole process can be traced using the W3C Prov model so the administrators can

[19] `http://data.open.ac.uk`

[20] It can be noted that we are describing the Unistats XML datasets as being part of the Linked Data. They are not, being in a non-RDF format. However, this does not change the meaning of the scenario for our demonstration.

```
:linkedup-cat
 dcat:dataset :prod ;
 dcat:dataset :cnr ;
 dcat:dataset :unistats-ld.12.13 ;
 dcat:dataset :unistats-ld.13.14 ;
 ...

:ou-cat
 dcat:dataset :ou-courses ;
 dcat:dataset :ou-qualifications ;
 dcat:dataset :ou-unistats ;
 ...
```

Fig. 5. A snippet from the LinkedUp and OU catalogues description.

```
:unistats.13.14
    prov:wasRevisionOf :unistats.12.13 .

:unistats-ld.12.13
    prov:wasDerivedFrom :unistats.12.13 .

:unistats-ld.13.14
    prov:wasDerivedFrom :unistats.13.14 ;
    prov:wasRevisionOf :unistats-ld.12.13 .

:prod dcterms:hasPart :prod-links .

:prod-links
    dcterms:isPartOf :prod ;
    prov:wasDerivedFrom :unistats-ld.13.14 ;
    void:target :unistats-ld.13.14 .

:ou-unistats
    dcterms:isPartOf :unistats-ld.13.14 ;
    prov:wasDerivedFrom :unistats-ld.13.14 .

:kis.13.14
    prov:wasRevisionOf :kis.12.13 .
:kis-ld.12.13
    prov:wasDerivedFrom :kis.12.13 .
:kis-ld.13.14
    prov:wasDerivedFrom :kis.13.14 .

:unistats.12.13 void:vocabulary :kis.12.13 .
:unistats.13.14 void:vocabulary :kis.13.14 .
:unistats-ld.12.13
    void:vocabulary :kis-ld.12.13 .
:unistats-ld.13.14
    void:vocabulary :kis-ld.13.14 .
```

Fig. 6. The scenario described with Prov-O and VoID.

```
:unistats.14.15
    prov:wasRevisionOf :unistats.13.14 .

:kis.14.15
    prov:wasRevisionOf :kis.13.14 .

:unistats.14.15 void:vocabulary :kis.14.15 .
```

Fig. 7. The information added to the dataflow description once the new Unistats data about 2014/2015 is published by HESA.

have understanding of how data is produced, published and consumed[21].

However, when the statistics of the new academic year (eg. 2014/2015) are published by HESA, there are a number of items in the catalogues that are affected in terms of timeliness, at least, and that require intervention by the related managers:

1. the LinkedUp catalogue administrator need to know which datasets in its catalogue needs to be updated (Unistats LD versions should be, while CNR should not);
2. the LinkedUp developer will have to review and update the KIS-LD schema to inherit the changes in the structure of the KIS XML schema;
3. and needs to adapt the extraction software to use the new schema and generate the new dataset;
4. old datasets are kept as historical reference, but pointers should be added to the latest version;
5. the OU needs to replace their dataset, it has to be a selection of the last Unistats LD 2014/15, and not of the 2013/14;
6. the PROD dataset needs to be updated. The portion containing the links need to be recomputed towards the new version because organizations or URIs might have changed.

Figure 7 lists the triples that reflect the change in the datahub, and that enrich the provenance database (an excerpt has been shown in Figure 6). The manager can query the provenance database to detect a number of possible items that are affected by the publishing of the new KIS dataset. Figure 8 shows a possible query to detect items that share some information with the new Unistats LD dataset. However, we can abstract the above points to a general Competency Question (CQ). Theoretically, any item that might *share an interpretation with* the new coming dataset is potentially

[21] While we could have detailed activities using Prov-O extensions, we simplified the implementation using only binary relations to show how, even with this semplification, querying the provenance graph can not be trivial.

```
SELECT distinct ?node WHERE {
  {
    :unistats.14.15  prov:wasRevisionOf ?node
  } UNION {
    ?node prov:wasDerivedFrom* ?something .
    :unistats.14.15  prov:wasRevisionOf* ?something
  } UNION {
    ?node dcterms:hasPart ?apa .
    ?apa prov:wasDerivedFrom ?ste .
    ?ste prov:wasDerivedFrom ?ast .
    :unistats.14.15 prov:wasRevisionOf ?ast
  } UNION {
    :unistats.14.15 void:vocabulary ?sc .
    ?sc prov:wasRevisionOf* ?node
  }
}
```

Fig. 8. SPARQL query to select affected items from the Provenance dataset.

affected by this event. The CQ would be: *Which are the datasets that might share the same interpretation model of the new coming dataset?* With Datanode we can add an abstraction layer on top of the provenance information. At the same time, we can specify many interesting relations that are not directly captured by existing information, for example the fact that `ou-unistats` is actually a selection of `unistats.13.14`, or that `prod` and `cnr` are adjacent, being member of the same catalogue (see Figure 9).

By specifying the relations between datanodes we can infer which nodes "share interpretation with" the new `unistats.14.15` node, that is the kind of abstraction needed by this use case. This relation will occur with datanodes such `:prod`, but not with the node `:cnr`. In fact, while both datasets are part of the same `dcat:Catalogue` (being `dcat:dataset` a sub property of `dn:describes`) and they are `dn:adjacentTo`, they do not share necessarly the same interpretation. Existing methods do not directly support the process and modelling described in this scenario, they can only be adapted with ad-hoc complex queries like the one shown in Figure 8. Figure 10 shows the result: Datanode nicely enhances the provenance information with an inferencing shortcut.

Data artifacts are often part of composite scenarios. With Datanode, many interesting relations between datasets can be represented and inferred, supporting the management, monitoring and the understanding of (linked) data catalogues.

6. Conclusions, perspectives and future work

Nowadays, data hubs includes a number of datasets with different scopes and relations between each other

```
:unistats.13.14
    dn:isUpdatedVersionOf :unistats.12.13 .

:prod dn:links :unistats-ld.13.14 .
:prod dn:hasPart :prod-links .
:prod-links dn:isSelectionOf :unistats-ld.13.14 .
:prod dn:adjacentTo :cnr .

:unistats-ld.12.13
    dn:remodelledFrom :unistats.12.13 .
:unistats-ld.13.14
    dn:remodelledFrom :unistats.13.14 .

:unistats-ld.13.14
    dn:isUpdatedVersionOf :unistats-ld.12.13 .

:ou-unistats dn:isCopyOf [
    dn:isSelectionOf :unistats-ld.13.14 ] .

:kis.13.14
    dn:nextVersionOf :kis.12.13 .
:kis-ld.12.13
    dn:remodelledFrom :kis.12.13 .
:kis-ld.13.14
    dn:remodelledFrom :kis.13.14 .

:unistats.12.13 dn:usesSchema :kis.12.13 .
:unistats.13.14 dn:usesSchema :kis.13.14 .
:unistats-ld.12.13 dn:usesSchema :kis-ld.12.13 .
:unistats-ld.13.14 dn:usesSchema :kis-ld.13.14 .

:unistats.14.15
    dn:isUpdatedVersionOf :unistats.13.14 .
:kis.14.15
    dn:nextVersionOf :kis.13.14 .
:unistats.14.15 dn:usesSchema :kis.14.15 .
```

Fig. 9. Extending the Provenance information with Datanode we can qualify the relations between data items further.

and with external data hubs, covering aspects like acquisition, persistence, versioning, delivering, processing, distributing, partitioning etc... . Many of these operations include the usage of multiple data sets and targets the creation of new ones from the sources. Data publishers have to deal with aspects such evolution. This becomes very difficult if data comes from complex processes. Keeping track of the propagation of changes in data sources, for example, can be hard when the number of indirect affected datasets increase, especially if they belong to different owners. Similarly, understanding the propagation of features of datasets from sources to derived datasets may be crucial if we want the catalogue users to benefit from the other datasets. Thus, reasoning on complex data flows (eg the propagation of the properties of data) becomes important and may be very complex, especially if relying on several metalevel descriptions like provenance, access control or terms and conditions.

Being able to harmonize portions of different metalevel descriptions in a unified inference process can be beneficial in many contexts, as the scenario recounted

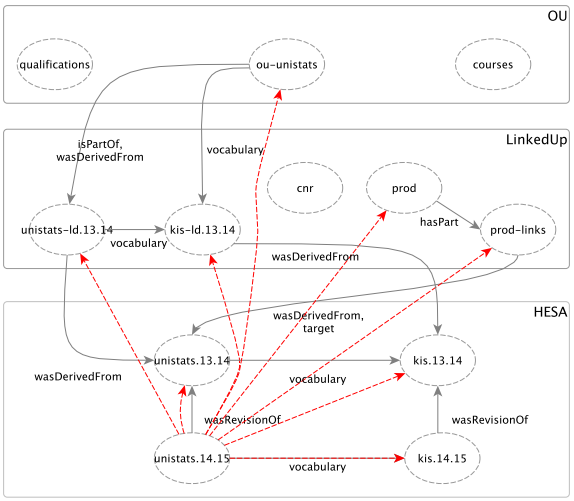

Fig. 10. Adding Datanode to provenance information: the dotted red arcs show the Datanode relation `sharesInterpretationWith`.

in Section 5 illustrated. However, there might be many other metadata schemas that could be aligned and translated to datanode descriptions, covering scenarios that include data mining processes (described with NIF [42], for example) or complex annotations (using EARMARK [43]). This applications still needs to be explored and evaluated. In addition, we concentrated on basic inferences that can be produced by using standard OWL constructs, like property types and relations. It is our intent to extend the inference model of Datanode, designing rules with OWL property chains or production rules.

The Datanode Ontology Pattern provide the abstraction necessary to unify a number of concern with relation to data management. Datanode is also a framework for the building of networks of data objects to support deep analysis of the structure of data hubs. Datanode does not offer an alternative to existing vocabularies, but aims to be a method to organize knowledge in an unified way to foster the analysis and understanding of data collections in the Semantic Web.

References

[1] Yoshitaka Minami, Hideaki Takeda, Fumihiro Kato, Ikki Ohmukai, Noriko Arai, Utsugi Jinbo, Motomi Ito, Satoshi Kobayashi, and Shoko Kawamoto. Towards a data hub for biodiversity with lod. In *Semantic Technology*, pages 356–361. Springer, 2013.

[2] Amir Nasr and Azadeh Keshtiarast. Datahub for aurin and ands project. *Spatial Data Access and Integration To Support Liveability*, page 75, 2013.

[3] Mathieu d'Aquin, Alessandro Adamou, and Stefan Dietze. Assessing the educational linked data landscape. In *Proceedings of the 5th Annual ACM Web Science Conference*, pages 43–46. ACM, 2013.

[4] AJ Ball, Kevin Ashley, Patrick McCann, Laura Molloy, and Veerle Van den Eynden. Show me the data: The pilot uk research data registry. In *9th International Digital Curation Conference*. University of Bath, 2014.

[5] John S Erickson, Eric Rozell, Yongmei Shi, Jin Zheng, Li Ding, and James A Hendler. Twc international open government dataset catalog. In *Proceedings of the 7th International Conference on Semantic Systems*, pages 227–229. ACM, 2011.

[6] Matias Frosterus, Eero Hyvönen, and Joonas Laitio. Datafinland—a semantic portal for open and linked datasets. In *The Semanic Web: Research and Applications*, pages 243–254. Springer, 2011.

[7] Michael Martin, Martin Kaltenböck, Helmut Nagy, and Sören Auer. The open government data stakeholder survey. In *OKCon*, 2011.

[8] 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*. Citeseer, 2009.

[9] Luc Moreau and Timothy Lebo. Linking across provenance bundles. 2013.

[10] John Erickson and Fadi Maali. Data catalog vocabulary (DCAT). W3C recommendation, W3C, January 2014. http://www.w3.org/TR/2014/REC-vocab-dcat-20140116/.

[11] Satya Sahoo, Timothy Lebo, and Deborah McGuinness. PROV-o: The PROV ontology. W3C recommendation, W3C, April 2013. http://www.w3.org/TR/2013/REC-prov-o-20130430/.

[12] Valentina Presutti, Enrico Daga, Aldo Gangemi, and Eva Blomqvist. extreme design with content ontology design patterns. In *Proc. Workshop on Ontology Patterns, Washington, DC, USA*, 2009.

[13] Eyal Oren. *Algorithms and components for application development on the semantic web*. PhD thesis, Citeseer, 2008.

[14] Alexandre Passant. Dbrec—music recommendations using dbpedia. In *The Semantic Web–ISWC 2010*, pages 209–224. Springer, 2010.

[15] Alberto Musetti, Andrea Giovanni Nuzzolese, Francesco Draicchio, Valentina Presutti, Eva Blomqvist, Aldo Gangemi, and Paolo Ciancarini. Aemoo: Exploratory search based on knowledge patterns over the semantic web. *Semantic Web Challenge*, 2012.

[16] Mathieu d'Aquin, Carlo Allocca, and Trevor Collins. Discou: A flexible discovery engine for open educational resources using semantic indexing and relationship summaries. In *International Semantic Web Conference (Posters & Demos)*, 2012.

[17] Spyros Kotoulas, Vanessa Lopez, Raymond Lloyd, Marco Luca Sbodio, Freddy Lecue, Martin Stephenson, Elizabeth Daly, Veli Bicer, Aris Gkoulalas-Divanis, Giusy Di Lorenzo, et al. Spud—semantic processing of urban data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2014.

[18] Fuyuko Matsumura, Iwao Kobayashi, Fumihiro Kato, Tetsuro Kamura, Ikki Ohmukai, and Hideaki Takeda. Producing and consuming linked open data on art with a local community. In *COLD*, 2012.

[19] Houda Khrouf and Raphaël Troncy. Eventmedia: A lod dataset of events illustrated with media. *Semantic Web journal, Special Issue on Linked Dataset descriptions*, pages 1570–0844, 2012.

[20] Francesco Osborne, Enrico Motta, and Paul Mulholland. Exploring scholarly data with rexplore. In *The Semantic Web–ISWC 2013*, pages 460–477. Springer, 2013.

[21] Wikipedia. Watson_(computer) — Wikipedia, the free encyclopedia, 2014. [Online; accessed 10-June-2014].

[22] Luc Moreau and Paolo Missier. PROV-dm: The PROV data model. W3C recommendation, W3C, April 2013. http://www.w3.org/TR/2013/REC-prov-dm-20130430/.

[23] Dave Reynolds and Richard Cyganiak. The RDF data cube vocabulary. W3C recommendation, W3C, January 2014. http://www.w3.org/TR/2014/REC-vocab-data-cube-20140116/.

[24] Pierre-Yves Vandenbussche and Bernard Vatant. Metadata recommendations for linked open data vocabularies. *Version*, 1:2011–12, 2011.

[25] I Davis. Vann: A vocabulary for annotating vocabulary descriptions. Technical report, Technical report, 2005.

[26] Carlo Allocca, Mathieu d'Aquin, and Enrico Motta. Door: Towards a formalization of ontology relations. 2009.

[27] Gofran Shukair, Nikolaos Loutas, Vassilios Peristeras, and Sebastian Sklarß. Towards semantically interoperable metadata repositories: The asset description metadata schema. *Computers in Industry*, 64(1):10–18, 2013.

[28] Mary Vardigan, Pascal Heus, and Wendy Thomas. Data documentation initiative: Toward a standard for the social sciences. *International Journal of Digital Curation*, 3(1):107–113, 2008.

[29] Sudha Ram and Jun Liu. A new perspective on semantics of data provenance. In *SWPM*. Citeseer, 2009.

[30] Olaf Hartig and Jun Zhao. Publishing and consuming provenance metadata on the web of linked data. In *Provenance and Annotation of Data and Processes*, pages 78–90. Springer, 2010.

[31] Carlos Pedrinaci, John Domingue, and Amit P Sheth. Semantic web services. In *Handbook of semantic web technologies*, pages 977–1035. Springer, 2011.

[32] Danh Le-Phuoc, Minh Dao-Tran, Josiane Xavier Parreira, and Manfred Hauswirth. A native and adaptive approach for unified processing of linked streams and linked data. In *The Semantic Web–ISWC 2011*, pages 370–388. Springer, 2011.

[33] Óscar Corcho, Daniel Garijo Verdejo, K Belhajjame, Jun Zhao, Paolo Missier, David Newman, Raúl Palma, Sean Bechhofer, Esteban Garc a Cuesta, José Manuel Gómez-Pérez, et al. Workflow-centric research objects: First class citizens in scholarly discourse. 2012.

[34] Niels Grewe. A generic reification strategy for n-ary relations in dl. In *OBML 2010 Workshop Proceedings*, 2010.

[35] Jorge Posada, Carlos Toro, Stefan Wundrak, and André Stork. Ontology supported semantic simplification of large data sets of industrial plant cad models for design review visualization. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 184–190. Springer, 2005.

[36] Paul Clements, David Garlan, Len Bass, Judith Stafford, Robert Nord, James Ivers, and Reed Little. *Documenting software architectures: views and beyond*. Pearson Education, 2002.

[37] Valentina Presutti, Eva Blomqvist, Enrico Daga, and Aldo Gangemi. Pattern-based ontology design. In *Ontology Engineering in a Networked World*, pages 35–64. Springer, 2012.

[38] Kay Renfrew, Helen Baird, Howard Green, Peter Davies, Amanda Hughes, Jean Mangan, and Kim Slack. Understanding the information needs of users of public information about higher education. 2010.

[39] Peter Davies. Can governments improve higher education through 'informing choice'? *British Journal of Educational Studies*, 60(3):261–276, 2012.

[40] Aldo Gangemi, Enrico Daga, Alberto Salvati, Gianluca Troiani, and Claudio Baldassarre. Linked open data for the italian pa: the cnr experience. *Informatica e Diritto*, 1(2), 2011.

[41] Fouad Zablith, Miriam Fernandez, and Matthew Rowe. The ou linked open data: production and consumption. In *The Semantic Web: ESWC 2011 Workshops*, pages 35–49. Springer, 2012.

[42] Giuseppe Rizzo, Raphaël Troncy, Sebastian Hellmann, and Martin Bruemmer. Nerd meets nif: Lifting nlp extraction results to the linked data cloud. *LDOW*, 937, 2012.

[43] Silvio Peroni and Fabio Vitali. Annotations with earmark for arbitrary, overlapping and out-of order markup. In *Proceedings of the 9th ACM symposium on Document engineering*, pages 171–180. ACM, 2009.