Towards Conceptual Structure Verification of Linked Data Vocabularies

Vojtěch Svátek[1], Miroslav Vacura[2], Martin Homola[3], Ján Kl̆uka[3]

[1] Dept. of Information and Knowledge Engineering, University of Economics, Prague,
W. Churchill Sq.4, 130 67 Prague 3, Czech Republic
svatek@vse.cz

[2] Department of Philosophy, University of Economics, Prague,
W. Churchill Sq.4, 130 67 Prague 3, Czech Republic
vacuram@vse.cz

[3] Department of Applied Informatics, Comenius University in Bratislava,
Mlynská dolina, 842 48 Bratislava, Slovakia
{homola,kluka}@fmph.uniba.sk

1 Introduction

Linked Data (LD) principles are increasingly exploited when publishing structured data in the WWW. Powerful mashups can for example be easily built over public SPARQL endpoints providing encyclopaedic resources such as DBpedia, government data resources (such as public spendings), or geographical resources.

One of the advantages of LD w.r.t. proprietary Web-enabled databases is their adherence to publicly available and widely shared vocabularies; notorious examples are FOAF (for personal profiles), Dublin Core (for bibliographic metadata), Music Ontology (MO; for recordings and other musical information), or GoodRelations (GR; for e-commerce data). One of the important prerequisites to smooth adoption of individual vocabularies, and, in particular, to matching multiple vocabularies (needed when integrating datasets that subscribe to different vocabularies) seems to be their conceptual consistency.[1]

Our proposal, outlined further in the paper, addresses consistency in terms of a so-called deep (conceptual) model that is 'hidden behind' the surface representation of a vocabulary. The crucial assumption is that the actual RDF/OWL representation (i.e., the *surface model*) often only roughly approximates the structure of the implicit conceptual model (i.e., the *deep model*) that faithfully captures the real-world state of affairs. To illustrate this problem, let us consider a 'simple fact' in RDF stating that a company's business is repairing, relying on the GR vocabulary (indicated by the 'gr' prefix):

ex:MyCompany gr:hasBusinessFunction gr:Repair .

While the subject and the object of this fact are, syntactically, both individuals in OWL DL terms, their deep conceptual types are strikingly different: while ex:MyCompany

[1] Not to be confused with logical consistency; unlike ontologies designed primarily for reasoning tasks, typical vocabularies (even if formally adhering to the OWL standard) contain too few complex axioms to become logically inconsistent.

M. Bieliková, M. Šimko (Eds.): WIKT 2012, pp. 89–92.

is truly an individual real-world object (ontologically said, a 'particular'), gr:Repair represents a quality (ontologically said, a 'universal') that can be assigned to many individuals. It is not hard to realize that the same conceptual relationship can be expressed (assuming an alternative GoodRelations vocabulary prefix, say, 'gr-alt') as

ex:MyCompany rdf:type gr-alt:RepairCompany .

The latter representation is much more faithful to the deep model, clearly indicating that an individual is assigned to a class (in other terms, declared to have some universal quality). While, in reality, we can rarely transform the structure of a widely used vocabulary (with many datasets referring to it) in the indicated way, we could at least label the entities of an existing vocabulary with the 'conceptual distinctions' of the deep model. For example, labelling the range of the gr:hasBusinessFunction property as a 'conceptual class' may help to avoid assigning to it (in some dataset) real-world individuals as instances. While in the example above the distinction seems obvious, it is not always straightforward to distinguish between a true individual and a conceptual class based on the name of the entity only (cf. examples in Sect. 3).

There are other works [5, 3, 2] aiming to cope with discrepancies and limitations of the surface LD models, most prominently OntoClean [4], which differs from our approach in its focus on annotating classes and repairing their taxonomic relationships.

In the rest of the paper, we first sketch the structure of the deep model. As a practical use case, we look at a fragment of the Music Ontology vocabulary, and show how the deep model can warn us of possible conceptual discrepancies when the vocabulary is merged with or mapped to another one. We also discuss our plans for practical use of the deep model approach for detecting problems in vocabularies and their mappings.

2 Deep Model for Linked Data Vocabularies

To avoid confusion with the surface models, we will prefix the primitives from the deep model with $\mathcal{D}$-, and partly even use distinct terms. The first two building units of deep models will be $\mathcal{D}$-objects and $\mathcal{D}$-classes.

$\mathcal{D}$-object refers to a real-world object, which can be tangible (such as people, animals, things, etc.) or intangible (various abstract entities such as topics, processes, etc.). It is analogous to the notion of *individual* in the surface model, and $\mathcal{D}$-objects also correspond to surface individuals in the majority of cases (e.g., the surface individual ex:MyCompany from the introduction maps to a $\mathcal{D}$-object in the deep model).

$\mathcal{D}$-class refers to a real-world class. Therefore it is usually a set of $\mathcal{D}$-individuals instantiating the same concept or sharing a common property. For simplicity, it also covers the notion of *quality* (e.g., red color), which is ontologically slightly different but plays the same role in the model structure.

The single key distinction between $\mathcal{D}$-objects and $\mathcal{D}$-classes is, respectively, that between particulars (which never have instances) and universals (which possibly may have instances). In the surface model, however, $\mathcal{D}$-classes may be reflected either as true RDFS/OWL classes, but sometimes also as surface individuals (e.g., gr:Repair as seen in the introduction).

Let us now have a look at relationships between entities. **$\mathcal{D}$-relationship** refers to a (particular) relationship between two entities (for instance, an individual is produced by some producer, or a person owns some individual). Its universal counterpart is **$\mathcal{D}$-relation**, which refers to a real world conceptual relation, a class of relationships.

Finally we enrich the model with data values and associated constructs. The particular called **$\mathcal{D}$-valuation** refers to an assignment of a data value to some entity (e.g., some person has height of 199 cm). Its universal counterpart is **$\mathcal{D}$-attribute**, referring to real world valuations of the same (usually quantitative) property.

Although in practice these relationships possibly involve more than two individuals, we focus on binary relationships, to keep the deep model aligned with RDF/OWL. Thus, $\mathcal{D}$-relationships ($\mathcal{D}$-valuations) are analogous to the surface notion of *object (data) property assertions*, and $\mathcal{D}$-relations ($\mathcal{D}$-attributes) are analogous to *object (data) properties*.

Note that the entities which take part in $\mathcal{D}$-relationships ($\mathcal{D}$-valuations) are not only $\mathcal{D}$-objects but also $\mathcal{D}$-classes as we show in practical examples in the next section.

3 Use Case: Deep Model and the Music Ontology Vocabulary

To demonstrate a practical use of the deep model, we now have a look on the Music Ontology vocabulary with the purpose of 'deep disambiguation' of selected constructs. Using this vocabulary we are able to express that the CBS 1992 CD release of Yo-Yo Ma's performance of J. S. Bach's 'Six Cello Suites' is an album:

```
ex:CBS1992Cd_YoYoMa_JSBach_SixCelloSuites
    mo:release_type   mo:album .
mo:album   rdf:type   mo:ReleaseType .
```

At the surface level, ex:CBS1992Cd_YoYoMa_JSBach_SixCelloSuites and mo:album are individuals and mo:ReleaseType is a class. Considering the deep model, however, we see that while the first surface individual is a $\mathcal{D}$-object, mo:album is in fact a $\mathcal{D}$-class (as is mo:ReleaseType). This is because mo:album can have instances, as documented by the example—ex:CBS1992Cd_YoYoMa_JSBach_SixCelloSuites is its instance. What we learn from the deep model is that the surface individual mo:album is of a specific kind (it is a $\mathcal{D}$-class) and it needs to be given special attention: e.g., it is of little sense to assign some data attributes to it as it does not stand for any real object.

There are other insights that can be learned here: mo:ReleaseType is in fact a specific type of $\mathcal{D}$-class—its instances are again classes (and should not be $\mathcal{D}$-objects). Moreover, the regular surface property mo:release_type represents a specific kind of $\mathcal{D}$-relation, namely *instantiation*, more typically represented by rdf:type. Note that a relationship between a $\mathcal{D}$-object and a $\mathcal{D}$-class is not always an instantiation; e.g.,

```
ex:YoYoMa   mo:primary_instrument   mo-mit:Cello .
```

clearly represents a regular $\mathcal{D}$-relationship, although the surface individual mo-mit:Cello represents a $\mathcal{D}$-class (many physical musical instruments are celli). These cases are not yet clearly recognized by our model, and are subject to our ongoing investigation.

4 Next Steps: Evaluation, Tool Support and Practical Application

Ongoing work concerns systematic mapping of LD vocabularies to the deep model structures, in order to evaluate the plausibility of the approach. We have already built a nearly-complete mapping for MO, GR, and FOAF, in the form of annotation of individual surface entities (in particular, classes, individuals, and property ranges) with 'meta-properties' corresponding to deep model primitives.

While this initial work was carried out without a dedicated tool support, we are, in parallel, developing an *annotator tool*—a Protégé plugin that would allow to create annotations (referring to a 'deep conceptual annotation' ontology) for each vocabulary, either generically or with respect to the use of the given vocabulary entity in a particular dataset, and to store them in a dedicated annotation space. On this basis we can extend the analysis to further vocabularies, on which we could test the degree of inter-annotator agreement (assuming shared annotator guidelines that are also under design).

Among the intended practical applications of this effort on the Semantic Web, we foresee assistance in *visual inspection* of vocabularies and datasets (summaries) via in-viewer transformation among different surface representations of the same deep structure (using tools such as PatOMat [6]). The annotations of entities with deep conceptual distinctions may also be used by reasoners, to detect some of the modeling discrepancies as sketched in Sect. 3. Finally, deep annotations would contribute to consistent *vocabulary mapping* [1].

Acknowledgements. We are grateful to Aldo Gangemi and Alan Rector for feedback to our initial ideas. This work was partially supported from the CSF grant no. P202/10/1825 (PatOMat—Automation of Ontology Pattern Detection and Exploitation), from the Slovak VEGA project no. 1/1333/12, and from the Czecho-Slovak bilateral mobility project no. SK-CZ-0208-11 (APVV agency) and no. 7AMB12SK020 by (AIP ČR agency).

References

1. Bizer, C., Schultz, A.: *The R2R Framework: Publishing and Discovering Mappings on the Web.* In: Proc. COLD'10. 1st International Workshop on Consuming Linked Data (COLD 2010), Shanghai, November 2010.
2. Gangemi, A.: *SuperDuper Schema: an OWL2+RIF DnS pattern.* In: Deep Knowledge Representation Challenge Workshop at K-CAP'11.
3. Glimm, B., Rudolph, S., Völker, J.: *Integrated metamodeling and diagnosis in OWL 2.* In: Proc. ISWC'10.
4. Guarino, N., Welty, C.: An Overview of OntoClean. In: Staab, S., Studer, R., eds.: *The Handbook on Ontologies*, Springer-Verlag, pp. 151–172.
5. Sunagawa E., Kozaki K., Kitamura Y., Mizoguchi R.: *Role organization model in Hozo.* In: Proc. EKAW'06.
6. Šváb-Zamazal, O., Svátek, V., Iannone, L.: *Pattern-Based Ontology Transformation Service Exploiting OPPL and OWL-API.* In: Proc. EKAW'10.