Ontology Engineering in the Era of Linked Data
by Oscar Corcho, María Poveda-Villalón and Asunción Gómez-Pérez

In computer science, the term ontology refers to a “formal, explicit specification of a shared conceptualization” [1]. Conceptualization refers to an abstract model that allows describing something relevant in the world, for which we normally use concepts, properties and constraints on their application (for instance, the Unified Modeling Language [UML] class diagrams that many software developers use, the entity-relationship models used to organize a database or any drawing that one makes in a whiteboard to start organizing an information model for system development). All those entities in the abstract model need to be described explicitly so that we cover as much as possible of the world phenomenon that we are trying to represent.

(For example, if we are talking about different types of persons or organizations, let’s include the different categories of persons and organizations that are involved in our model of the world, as well as the relationships and constraints that hold among them.) Being formal refers to the ontology being machine-readable – that is, available in some language such as the Resource Description Framework Schema (RDFS) or the Web Ontology Language (OWL) that can be easily processed. And finally, and most importantly, shared reflects the notion that an ontology captures consensual knowledge; that is, it is not private to some individual, but accepted by a group.

The discipline of ontology engineering [2] can be understood, in a broad manner, as the one that works on methods, tools and techniques to facilitate the development of ontologies. This discipline has been active for more than two decades and has witnessed an important evolution during its lifetime, much of it related to the development of the following:

- **Ontology languages**, for example, Ontolingua and Loom in the early 90s, moving into ephemeral languages such as OIL or DAML+OIL, and then the appearance of the World Wide Web Consortium (W3C)
recommendations, RDF Schema and OWL, in the early 2000s

- **Ontology engineering tools** such as Protégé, OntoEdit, KAON, NeOn Toolkit and TopBraid Composer
- **Ontology development methodologies** such as Methontology, On-To-Knowledge, Diligent, NeOn Methodology.

For a long time, the ontology engineering community was primarily involved in the construction of such languages, tools and methodologies, and less focused on developing core pieces of knowledge that may be potentially used and reused, with important exceptions such as a few upper-level and generic ontologies and some ontologies in the biology and medical domains. Ontology design patterns emerged at the end of the last decade as best practices for ontology modeling, and standardization bodies such as the W3C started to promote ontologies as recommendations in various domains such as provenance, organizations, contact details, data catalogues and sensor networks.

More recently, linked data has emerged as a publishing paradigm that allows exposing data on the web in a structured manner, following a set of clear principles that exploit the characteristics of the HTTP protocol and make extensive use of the W3C RDF specification. Even though linked data per se does not impose the restriction of using ontologies for structuring the data that is being exposed using such principles, it is a well-known practice to relate the data that is being generated and published with existing ontologies, which are also published following linked data principles.

As an example, let us imagine several linked datasets that provide information about Cervantes without defining, by means of ontologies, what the term *Cervantes* refers to. Hence Cervantes may be a tapas bar, an authority in a library, the name of a street, the name of a municipality and so forth. If we were using URIs (universal resource identifiers) to refer to all these meanings for Cervantes and were inadvertently linking them using the owl:sameAs property, a simple search for the properties applicable to Cervantes may output one or several telephone numbers, GPS coordinates, the length of the street, a birthdate, the number of inhabitants in a city or the famous book *Don Quixote de la Mancha*, among others. Adding a little semantics would help avoid this confusion by defining the tapas bar as a bar, the author as a Spanish writer, the street as part of the infrastructure of a city and the municipality as a territorial unit. Paraphrasing the slogan from the SHOE project, “A little semantics goes a long way” ([www.cs.rpi.edu/~hendler/LittleSemanticsWeb.html](http://www.cs.rpi.edu/~hendler/LittleSemanticsWeb.html)).

Linked data puts ontology engineering in a new context upon which we reflect in this article. We will focus on two main dimensions to explain this evolution: (1) the types of ontologies that are being created, published and used and (2) the types of ontology engineering skills that are being exploited.

### Types of Ontologies in the Era of Linked Data: From Lightweight to Frankenstein

Two main dimensions have been traditionally used for the classification of ontologies [4]: (1) lightweight vs. heavyweight and (2) application vs. domain vs. generic vs. upper-level.

The differentiation between lightweight and heavyweight ontologies is mostly based on the amount and characteristics of the axioms included in the ontology. Lightweight ontologies are those that are mostly defined with concept and property definitions, as well as simple concept taxonomies supporting simple taxonomic inferences. This level is also typical of UML class diagrams and entity-relationship models. Heavyweight ontologies include the previous constructs plus other types of restrictions and axioms that allow performing richer inferences with the underlying data. In traditional ontology engineering, the generation of heavyweight ontologies was viewed with favor, in general, because it demonstrates a deep understanding of the domain being modeled. The differentiation between lightweight and heavyweight ontologies is not completely crisp; that is, there are several degrees, as graphically displayed by Lassila and McGuiness [5].

From the point of view of the second dimension, ontologies can be categorized as upper-level, generic, domain or application-focused, depending on their degree of usability and reusability (adapted from [4]). Upper-level and generic ontologies are reusable across domains since they cover concepts that are applicable for many different domains (for example, units of measure, time and space). Domain ontologies are reusable inside a
repositories like LOV (http://lov.okfn.org/) and selecting any of the results obtained without checking whether they are actually compatible or whether the original semantics of the reused terms are preserved in the ontology being developed.

Ontology Engineering Skills in the Era of Linked Data

Our second dimension on the evolution of ontology engineering in the era of linked data is related to the skills required for those who embark on ontology engineering practices in this context. Following on the discussion raised during the EKAW2014 keynote on “Ontology Engineering by and for the Masses” (http://goo.gl/g0TPBe), we can talk about at least five different groups of ontology engineers in this linked data context:

1. The **upper-level ontology engineers** have deep knowledge about formal logic and philosophy. This community traditionally writes formal upper-level ontologies such as DOLCE, BFO, GFO or SUMO using first order logic, OWL or OBO.

2. **Heavyweight ontology engineers** may be domain experts (for example, graduates in biology or geography) or computer scientists with a good background in various forms of logic. The OWL-based domain and application ontologies developed by this profile may reuse axioms, properties and concepts from upper-level ontologies like the ones mentioned above.

3. **Lightweight ontology engineers** develop vocabularies to be used in the linked data context. These vocabularies are usually written in RDF Schema or in OWL profiles with little expressivity.

4. **SKOS (Simple Knowledge Organization System) concept scheme developers** are those who are interested in developing thesauri and other types of classifications. Many of them have a strong background in information and library sciences.

5. Finally, web developers contributing to Schema.org or using it to annotate websites from the cluster of **Schema.org vocabulary developers**. The technologies involved in these activities are mainly HTML, RDFa and JSON-LD.
ontologies. When looked at through serious ontology engineering lenses, these ontologies are breaking many of the well-established rules traditionally considered in ontology engineering.

On the other hand, linked data has brought with it the need for more ontology engineers – ontology engineering by the masses – that are able to generate vocabularies that can be used for creating linked data. This new breed of ontology engineers may not have received enough training in knowledge modeling, resulting in badly designed combinations of lightweight ontologies, SKOS concept schemes and Schema.org terms. Furthermore, they may be creating their ontologies with the idea of applying constraint checking on the underlying data as if they were modeling a database, which also leads to problems in the design of their ontologies. Providing support to this new blend of ontology engineers and linked data producers will be one of the main challenges that the ontology engineering field will have to address.

The table above provides a summary of such relationships among profiles, skills and objectives.

We can also graphically characterize these profiles according to three dimensions: usual ontology size, level of consensus achieved and the degree of formality of the developed ontologies as shown in Figure 1.

The Future: Ontology Engineering by and for the Masses

This article has reviewed in a superficial manner how ontology engineering has evolved with the emergence of linked data. On the one hand, linked data has brought with it the challenge of developing ontologies that are lightweight and that also use ontology terms coming from different and sometimes disconnected ontologies. These practices are followed in order to maximize the reusability and interoperability of the data that is being exposed as linked data, but they lead sometimes to the so-called *Frankenstein*
Resources Mentioned in the Article


