# Is An Upper Ontology Useful?

Peter Winstanley [1]

[1] *Semantic Arts, Scotland and Fort Collins, CO, USA.*

**Abstract**

Ontologies play a fundamental role in information integration using semantic technologies. Whilst most ontologies are specific to a single domain, other ontologies provide a level of integration for these domain-specific ontologies and should be the starting point of any domain ontology work. They are referred to as 'upper ontologies'. This presentation will consider the benefits of using an upper level ontology and illustrate how they might be deployed in everyday use.

**Keywords**

Ontology; information architecture; knowledge management



## 1. Introduction

The short answer (TL; DR; response) to the question is 'Yes!'. So let's dig into this question a bit deeper to find out why an upper ontology is useful.

First, I want to consider the following questions:

- What is an upper ontology?
- Is there only one, or are there potentially many?
- If there are many, why is that, and could we simplify the situation?

Terms like 'upper ontology' and 'top-level ontology' are prepositional, and the accepted convention is that 'upper' corresponds to 'more abstract', 'larger grained' etc. However, the key point is that these ontologies are not directed at any specific domain. This distinguishes them from 'domain ontologies', and it means that the competency questions that they have to be able to answer are very different to those of a domain ontology. Ontology is not a topic of interest solely to academic research – it is a practical engineering discipline practiced in a very wide range of industries by a small number of people. Sometimes these ontologists adopt an upper ontology built by others, and sometimes they are in organizations such as mine, Semantic Arts, who have been involved in ontology engineering for over 20 years and have through that time developed their own, called 'gist' [1]. This paper comes from a practitioner's perspective and 'useful' will mean helpful in facilitating the timely development and implementation of highly functional and logically correct ontologies. The principal functionality of the ontologies we build using "gist" as the upper ontology is to use them in enterprise knowledge graphs as the basis for information interoperability. But as we will see, that isn't the only function of an ontology. At Semantic Arts, we look to help clients exploit all the benefits of an ontology.

In the rest of the paper, I am not going to be exhaustive in describing and comparing upper ontologies. There are several reviews available [2,3]

## 2. Is there a 'Goldilocks' level of abstraction?

Can upper ontologies be too abstract, or not abstract enough? The answer, unfortunately, is "it depends". Clearly having an ontology composed of disjoint owl:Thing and owl:Nothing might be

ISKO UK conference, July 24–25, 2023, Glasgow, Scotland
EMAIL: peter.winstanley@semanticarts.com (A. 1);
ORCID: 0000-0002-3970-5150 (A. 1);

logically consistent, but it's not really that helpful. At the other end of the spectrum, a complex ontology with several hundred classes and properties, isn't that helpful either. Looking at the history of the 'gist' ontology over the years of its 12 versions there seems to have been a consistency in the numbers of classes – in the order of 140, and the number of axioms in total are in the 1500-1700 range. I think that these are on a very human scale. The classes are roughly the same number as the people in our year at high school, and the axiom count is in the same order as the number of people in the whole school. These numbers mean that most people should be able to get to know the ontology quite well. This, to my mind, is a relevant aspect of working out the 'Goldilocks' level of abstraction. With fewer classes and axioms, the ontology is going to naturally be more abstract. This means that for every domain ontology, high-level concepts are going to have to be (re-)created which will almost inevitably lead to some repetitive work across domain ontologies. Conversely, if there are many more classes and axioms, then there is likely to be too much detail for a top-level ontology or too much restriction/specialization to promote reuse. That, in turn, leads to the creation of still more classes and axioms to suit the immediate purpose.

## 3. What is an upper-level ontology useful for?

### 3.1. A starter palette of classes and properties

The first thing that I think an upper-level ontology is useful for, is to provide an initial palette of components for building domain ontologies. Without this upper-level, starting from owl:Thing, we would need to build out similar classes and properties for every domain ontology that we are developing. Using an upper-level ontology each time saves us time and effort. And it has the added value of having been vetted through use by other modelers.

### 3.2. Commonality between domain ontologies

Beyond saving time and effort, if we decide to use the same upper-level ontology each time we are building a domain ontology, then there are going to be the same common upper-level classes. This provides a link in a semantic way between domain ontologies. It will also link the domain ontologies in a logical way. There are obviously pros and cons for doing this. The pros are that we get information integration, albeit at a high level, for all data that is compiled using our domain ontologies. The potential con arises from the fact that the expressivity of the final ontology might not be suited to either the task at hand, or to the software (triplestores, reasoners, etc.) that we are using with our data.

I'm discussing OWL2 ontologies in this paper because they are the ones we use in practice at Semantic Arts, and this is a paper based on practical data engineering in real-world settings. OWL2 is based on a family of description logics. These translate into OWL2 profiles, and the profiles are optimisations of OWL2 for specific tasks. If there are logical relationships or constraints expressed in the upper-level ontology that are not consistent with the required profile of the domain ontology, then something has to give. Either we need a new upper-level ontology, or else we do some axiom pruning of the full representation of the domain ontology and its imported upper-level ontology.

The gist ontology has the expressivity SROIQ(D). It is consistent with the OWL2 DL. However, the triplestore RDFox works with ontologies of the OWL2 RL profile [4]. The pruning of axioms to bring an OWL2 DL ontology into line with the RL profile requires the removal of …. and this can be done using tooling such as the OWL toolkit [5].

### 3.3. Logical consistency of models

When working with ontologies, using the description logics as architectural logical principles, we can at any point in the development of a domain ontology assess the logical consistency of our developing model. This is an important step of ontology development, and, unfortunately, tools that don't include the possibility of using a reasoner make undertaking this check really difficult. The key point I want to make about the utility of an upper level ontology in this regard is that, if it is welldesigned, there will be a number of axioms that help partition the world of 'things' - it will 'cleave nature at its joints' using disjoint axiom ("this" is not "that"), and if this heavy lifting is done once, and at a high level of abstraction, then it saves us time by not only doing the job, but even just remembering to do the job. From a practical perspective, anything that we can do to save time and to build in effectiveness in the logical consequences of our ontology axioms is going to improve the performance and accuracy of reasoning over data using our ontologies. Using the same upper-level ontology over many domain ontologies ensures logical consistency across the datasets built using these domain ontologies.

## 3.4. A common vocabulary

As we all know, words matter. Using the upper-level ontology as a common vocabulary across an organisation provides a shared set of abstract terms that allow people to chunk their thought and to collaborate across workstreams and domains effectively. The ontology is not only to be used by computers – it is part of the vocabulary of the organization. Choice of an upper-level ontology is going to be critical here if we want to encourage the use of the class names to enter/reflect the corporate vocabulary. 'Continuant fiat boundary' might not be for everyone. Clearly here I'm not talking about using the most abstract ontologies. But ones such as gist, Dolce Ultralite, and Dolce Zero are good candidates for upper ontologies where the vocabulary matches that of everyday speech in the organization. Encouragement to use these ontology terms in the work environment can lead to improved specificity of communication and more rapid collaboration on information sharing. Another aspect of 'common vocabulary' is that explicit semantics expressed through ontologies and graph serialisations such as RDF provide a common vocabulary between people and computers – the code, the data, everything that the computer has to work with, is serializable to something that people can easily understand.

## 3.5. A tool for thought

Once we have a shared vocabulary of abstract concepts that is shared across an enterprise, it becomes a tool for thinking. Psychological research tells us about the 5 +/- 2 slots that we have in working memory, and that there is an auditory loop. Chunking our thinking using well-understood abstractions within the enterprise as a standard modus operandi is something that psychological research describes. It is not only the upper-level abstractions, but also their co-occurrences that help us relate domain knowledge [6]. As it is with people, so it is with the data of a knowledge graph. We can use the high-level abstractions to work up, down, and horizontally through the graph to get a more thorough understanding of the data contained in the graph.

## 3.6. Integrated data

Knowledge graph data from multiple domains converted or created using a domain ontology based on a common upper ontology becomes interoperable in that it can be queried from a single set of upper ontology terms. I have described this as being like arranging clothes in a closet using a hanger. The simple step of using that upper ontology as the 'hanger' makes viewing and arranging (sorting, querying, re-arranging, etc.) data from all datasets a joy rather than a pain. This tends to be the main selling point of using an upper ontology, but it is just one of many.

## 3.7. Partitioning data

The corollary of integrating data on the basis of abstract terms in an upper-level ontology means also that these integrated data can be partitioned into trees on the basis of a simple single selection step. This should speed up querying, but it may also be that it helps secure data. For example, in our data we may have clients, patients, suppliers and so on, but knowing that each of these is a rdfs:subClassOf e.g. gist:Person makes it easier than it might otherwise be to identify data assets that are controlled by GDPR or CCPA.

## 4. Conclusion

A reflection on practice supports the idea that an upper ontology is significantly more useful than simply a means to enable information integration. The development of practice will include exploiting further the additional utility that I have described.

## 5. Acknowledgements

Thanks to all colleagues in Semantic Arts, and especially to Dylan Abney, Dan Carey and Rebecca Younes for review and suggestions.



## 6. References

[1] Semantic Arts gist ontology, 2023. URL: https://semanticarts.com/gist

[2] V. Mascardi, V. Cordì, P. Rosso, A Comparison of Upper Ontologies. In: WOA 2007: Dagli Oggetti agli Agenti. 8th AI*IA/TABOO Joint Workshop "From Objects to Agents": Agents and Industry: Technological Applications of Software Agents, Genova, Italy. Seneca Edizioni, Torino, 2007, pp.55-64.

[3] Construction Innovation Hub: A survey of top level ontologies. URL: https://www.cdbb.cam.ac.uk/files/a_survey_of_top-level_ontologies_lowres.pdf

[4] RDFox, URL: https://docs.oxfordsemantic.tech/reasoning.html#owl-2-ontologies-vs-rdfox-rules

[5] OWL Toolkit, URL: https://github.com/noorbakerally/owl-toolkit

[6] J.R. Binder, In defense of abstract conceptual representations, Psychon Bull Rev 23, (2016) 1096–1108