4.0: Prelude to the Web Ontology Language

The language that we will use to serialize the ontology is the most widely used ontology language for computational purposes, being the Web Ontology Language OWL. OWL was standardized first in 2004 and a newer version was standardized in 2009, which has fuelled tool development and deployment of ontologies in ontology-driven information systems. OWL looks like yet another a language and notation to learn, but the ones that we will consider (the DL-based ones) have the same underlying principles. It does have a few engineering extras, which also has as consequence that there are several ways to serialize the ontology so as to cater for software developers’ preferences. Thus, theoretically, there is not really anything substantially new in this chapter, but there will be many more options and exercises to practically engage with the ontology languages, automated reasoning, and toy ontologies to play with on the computer. Depending on your interests, things start to get ‘messy’ (for a theoretician) or finally concrete (for an engineer).

OWL actually constitutes a family of languages consisting of OWL “species”, and we will focus on those species that are based on DLs, which are all fragments of the most expressive one, OWL 2 DL. To understand how these species came about in the way they are, with these features and not others, we will touch upon OWL in Section 4.1 first, which will shed light on questions such as: What goes into standardizing a language? Why precisely these ones came out of the efforts, and in this way? Why should an ontology engineer even consider the previous version and not only the latest?

Afterward, an overview of the DL-based OWL 2 languages is provided in 4.2:OWL 2. Yes, plural; that’s not a typo. As we shall see, there are good reasons for it both from a computational viewpoint for scalable implementations and to please the user-base. The computational aspect is summarized in Section 4.2:OWL 2 "Complexity Considerations for OWL". If you have completed a course on theory of computation, this will be easy to follow. If not, you would want to consult Appendix C, which provides an explanation why one cannot have it all, i.e., both a gazillion of language features and good performance of an ontology-driven information system. Experience has seen that that sort of trade-off can annoy a domain expert become disappointed with ontologies; Section 4.2:OWL 2 "Complexity Considerations for OWL" (and the background in Appendix C) will help you explain to domain experts it’s neither your fault nor the ontology’s fault. Finally,
OWL does not exist in isolation—if it were, then there would be no tools that can use OWL ontologies in information systems. Section 4.3:OWL in Context therefore sets it in context of the Semantic Web—heralded as a ‘next generation’ World Wide Web—and shows that, if one really wants the extra expressiveness, it can fit in another logic framework and system even up to second order logic with the Distributed Ontology, Model, and Specification Language (DOL) and its software infrastructure.