

Diagrammatic Ontology Engineering

Gem STAPLETON^{a,1}

^a*Visual Modelling Group
University of Brighton,
Brighton, UK*

Abstract. Ontology engineering involves defining axioms to capture required constraints when modelling a domain of interest. Ontologies arise in many areas, with potentially a diverse range of end users involved in their creation. This leads to the requirement for accessible approaches to ontology engineering, as some stakeholders need not be fluent or trained in symbolic notations such as OWL. This paper summarises concept diagrams and property diagrams which are designed to be an accessible alternative to OWL. The paper reports on two empirical studies, the first of which focuses on how to choose effective concept diagrams for expressing simple OWL axioms. The second study compares these effective diagrams to both OWL and DL, demonstrating that they can bring about significant improvements in task performance for novice users. These results support the incorporation of concept diagrams into ontology engineering tools, such as Protégé or WebProtege. This is an exciting prospect, allowing more stakeholders to fully engage with the ontology engineering process, leading to more efficiently produced and robust ontologies in the future.

Keywords. ontology engineering, visualization, concept diagrams

1. Introduction

Ontology engineers are often faced with the challenging problem of axiomatising complex systems using formal, logical notations such as OWL [1]. To support the use of OWL in ontology engineering, various tools have been developed with Protégé being a prominent example [2]. However, the symbolic-like nature of OWL can pose a barrier to entry for those who are not mathematically trained. This barrier is potentially problematic: often, producing a set of accurate axioms requires the input from a variety of people, reflecting the need for ontologies in diverse domains such as privacy engineering and biomedical sciences. In contrast to symbolic approaches, visual (diagrammatic) approaches are often seen as user friendly notations that can be accessible to a broad user base. A major goal is to make ontology engineering more broadly accessible by providing a fully formal, empirically supported, diagrammatic logic. This paper briefly summarises *concept diagrams* [7] and the newly designed *property diagrams*, along with early-stage empirical studies to demonstrate their efficacy [3,4]. It goes on to summarise some key challenges that must be addressed to make concept diagrams and property diagrams widely usable in practice.

¹Corresponding Author: Gem Stapleton, University of Brighton, Brighton, UK; E-mail: g.e.stapleton@brighton.ac.uk.

2. Concept Diagrams and Property Diagrams

Concept diagrams and property diagrams are formed from Euler diagrams augmented with additional syntax to give a highly expressive logic. Concept diagrams are geared towards making OWL 2.0 class expression axioms, although they are also capable of making assertions about properties such as their domain and range. Likewise, property diagrams are aimed at making OWL 2.0 object property expression axioms, primarily to describe property hierarchies and disjointness. For both concept diagrams and property diagrams there is not an exact equivalence in expressive power with the implied OWL 2.0 fragment: each diagrammatic notation is able to express information that is not expressible in the implied OWL fragment and vice versa.

A full formalization of concept diagrams can be found in [11]. A brief introduction to their core syntax and semantics is given here via examples in which the distinction between syntax and semantics is sometimes blurred. As with Euler diagrams, closed curves represent sets (called *concepts* in description logic and *classes* in OWL). *Properties* or *roles* (i.e. binary relations) are represented by arrows. Individuals are represented by dots or, more generally, trees.

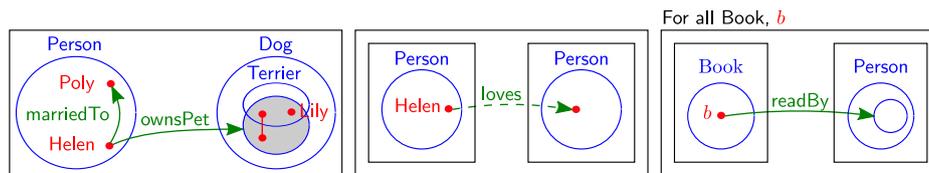


Figure 1. Concept diagrams.

Suppose that we have some information about the individual Helen that we wish to axiomatize:

1. Helen is a person.
2. Helen is married to the person Poly, and nothing else.
3. Helen owns exactly two pets, both of which are dogs.
4. One of Helen's pets is a terrier called Lily.

The left-hand diagram in figure 1 expresses this information, along with other things. We start by noting that there are three classes, Person, Dog and Terrier. These three classes are represented using the three labelled closed curves. Notice that the curves for Person and Dog do not overlap; this expresses that Person and Dog are disjoint classes. In addition, the enclosure of Terrier by Dog asserts subsumption in the obvious way. We also need to represent the individuals Helen, Poly and Lily. Each of them is represented by a labelled dot in the desired location; for example, Lily is located inside the curve labelled Terrier. What remains is to represent the information about properties. Focusing first on the property *married*, the arrow in the diagram connecting Helen to Poly asserts that Helen is married to Poly and only Poly. The other arrow sourced on Helen targets an unlabelled curve, analogous to an unnamed class. This curve represents the set of thing that are pets owned by Helen and, since it is drawn inside the Dog curve, these pets are all dogs. The curve includes Lily and one other, unnamed, individual represented by the two-node tree (so Helen owns two pets, including Lily). We do not know whether the unnamed individual is a terrier. This uncertainty is captured by use of two nodes, one

inside both the Dog and Terrier curves and the other inside the Dog curve but outside the Terrier curve. Shading is used to express that the only dogs owned by Helen are represented by the trees. In general, in a shaded region, all individuals must be represented by nodes or trees.

As well as using *solid* arrows to represent restrictions on properties, concept diagrams also use *dashed* arrows. These dashed arrows are used when we do not wish to express complete information about the image of a property under the domain restriction imposed by the arrow's source. For example, we may wish to express that Helen loves some person, without identifying the set of things that Helen loves (i.e. the image of the property loves when its domain is restricted to Helen). A concept diagram expressing this is in the middle diagram of figure 1. The arrow connects diagrammatic syntax placed in different boxes to ensure that we have not asserted that the person Helen loves is different from Helen.

The right-hand diagram of figure 1 expresses that every book is read by only people. The *quantification expression* written outside of the rectangles tells us that the diagram is making an assertion about all books. Lastly, we note that concept diagrams can also make assertions involving inverse relations, by annotating arrow labels using the symbol $\bar{}$, and negation by labelling a bounding box with 'Not'.

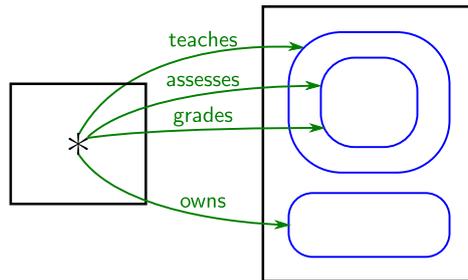


Figure 2. Expressing property subsumption, disjointness and equivalence.

The *property diagram* in figure 2 illustrates how to express property subsumption, disjointness and equivalence diagrammatically. A key difference, compared to earlier examples, is the use of * as an arrow source. This syntactic device acts as a universal quantifier over elements of the domain. For instance, this property diagram tells us that for each thing t , the set of things that t assesses is a subset of the set of things that t teaches. Therefore, the property assesses is subsumed by teaches. Two of the arrows, labelled assesses and grades, both target the same curve, thus asserting that assesses and grades are equivalent. Through the use of curves with disjoint interiors, the diagram also tells us that teaches and owns are disjoint properties. In addition, we can also see that assesses and grades are disjoint from owns.

Figure 3 extends figure 2, expressing that the property researches is subsumed by interests. By exploiting the additional rectangle, we have not made any assertion about the relationship between these two properties (researches and interests) and teaches, assesses, grades or owns. In addition, the inclusion of the class Topic indicates that the range of both researches and interests is Topic. Domain information can be expressed by property diagrams, using inverses.

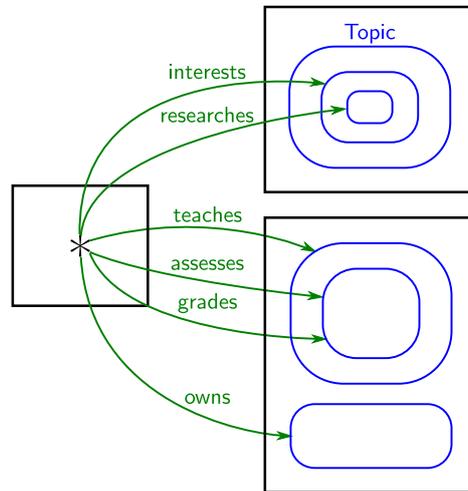


Figure 3. Using multiple boundary rectangles.

3. Choosing Effective Concept Diagrams for Common Axioms

Common ontology axioms include class subsumption and disjointness constraints (see figures 4 and 5), along with All Values From, Some Values From, Domain, and Range restrictions. As with any logic, concept diagrams offer a variety of ways to assert these kinds of relationships. To best support ontology engineers, it is important to understand the relative impact of different choices of axiomatization on task performance. To this end, the authors of [3] set out to identify features of concept diagrams that support better user task performance, measuring time and accuracy. They proposed three different kinds of diagrammatic *patterns* for defining axioms:

1. purely diagrammatic (called *unquantified* in [3]), in which no explicit use of logical operators (e.g. Not) or quantifiers (e.g. For all) is permitted ,
2. quantified diagrams using solid arrows, and
3. quantified diagrams using dashed arrows.

Examples of the purely diagrammatic versions of the patterns are shown in figures 4 to 9.

The three types of pattern were evaluated by conducting an empirical study, collecting performance data using multiple choice questions. The error rates were as follows: purely diagrammatic, 23.44%; quantified with solid arrow, 27.81%; quantified with dashed arrow, 79.62%. The mean times taken, in seconds, to provide a correct answer were: purely diagrammatic, 18.40; quantified with solid arrow, 20.89; quantified with dashed arrow, 29.05. The statistical analysis of their data indicated that avoiding explicit quantification, and representing the information purely diagrammatically, best supports task performance. Thus, this study guides ontology engineers who are using concept diagrams towards avoiding explicit quantification where possible.

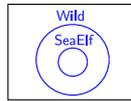


Figure 4. Subsumption.

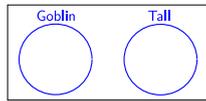


Figure 5. Disjointness.

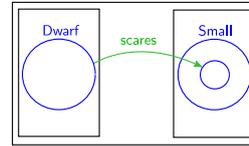


Figure 6. All values from.

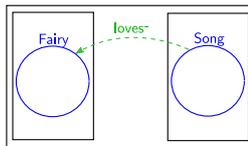


Figure 7. Some values from.

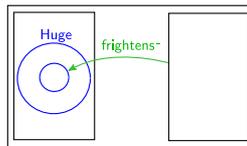


Figure 8. Domain.

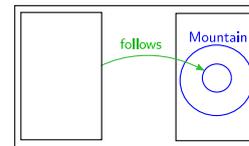


Figure 9. Range.

4. Comparing with OWL and DL: A Test of Efficacy

Having identified effective diagrammatic patterns for common ontology axioms, it was felt important to determine whether there really is an advantage in using diagrammatic patterns over standard notations in ontology engineering. An empirical evaluation compared the six diagrammatic patterns illustrated in figures 4 to 9 with equivalent axioms expressed in OWL (as displayed in the stylized form of the Protégé version 4.3 interface) and description logic [4]. Participants were asked to select the meaning of the diagram or statement from a choice of four options. Concept diagrams were found to support significantly better task performance than both OWL and DL. As an indication of the scale of benefit, we include here the error rates and mean times taken to provide a correct answer to questions. Regarding errors, participants exposed to concept diagrams returned an error rate of 7.59%, which increased to 27.03% when using OWL and 33.58% when using description logic. The mean times taken were 13.51 seconds for concept diagrams, 17.25 seconds for OWL and 22.99 seconds for description logic. These data suggest that, as well as providing a statistically significant benefit, the scale of this benefit is likely to be of real practical relevance.

This second study raises an obvious question: why are diagrams more effective? The theory of well-matchedness, introduced by Gurr, can help to explain this phenomenon: a notation is well-matched to its meaning if its semantic relationships are matched by, or mirrored by, its syntactic relationships [5]. In the case of concept diagrams, specifically the underlying Euler diagrams, the use of spatial relationships between closed curves is well-matched. For instance, to express subsumption, that is C_2 is a subset of (i.e. contained by) C_1 , the curve C_2 is contained by the curve C_1 . Here, as with disjointness, the semantics are clearly mirrored by the diagram used for the axiom in question. Generally speaking, diagrams are often well-matched, unlike symbolic and textual notations. This property of diagrams is one reason why it is often believed that they lead to improved task performance.

The concept diagrams for *subsumption*, *disjointness*, *all values from* and *range* axioms are particularly well-matched to meaning. In the study, participants performed well with them (the raw data can be found at <https://sites.google.com/site/eisamalharbi/understandingontologies>). The highest error rate for these four axiom types was 6.67% when using concept diagrams, with just one

error for each of *all values from* and *range*; by contrast, the *lowest* error rates for OWL and DL were 14.44% and 22.22%, respectively, both for *subsumption*.

However, it could be argued that the other two axiom types, *some values from* and *domain* do not have such well-matched diagrams. In particular, the cognitively more difficult inverse properties used in these axioms are not well-matched to meaning. The study yielded, for these two axiom types, the highest error rates for concept diagrams (21.11% and 8.88% respectively). In the case of *some values from*, though, it is notable that both OWL and DL have particularly high error rates too (40.00% and 38.88%), indicating that this kind of axiom is cognitively difficult. It is certainly possible, given these data, that inverse properties bring with them cognitive difficulty, but the main burden may lie in understanding *some values from*, insight supported by other research [6,9,8,10,13].

5. Conclusions and Future Challenges

Concept diagrams and property diagrams, designed for ontology engineering, have real potential as a formal alternative to OWL, supported by initial empirical evidence. However, much remains to be done in order to realise this potential and to convincingly demonstrate their utility to ontology engineers. Firstly, empirical studies are required that focus on interpreting more complex concept diagrams, such as that in figure 10, as well as property diagrams. Some work has already begun in this regard, where participants were asked to identify classes and properties that are necessarily empty (i.e. they are unsatisfiable); again, concept diagrams supported significantly better task performance than OWL [12]. Alongside this, we need to understand whether it is easier for ontology engineers to construct correct ontology axioms using concept diagrams rather than the standard approach of using OWL.

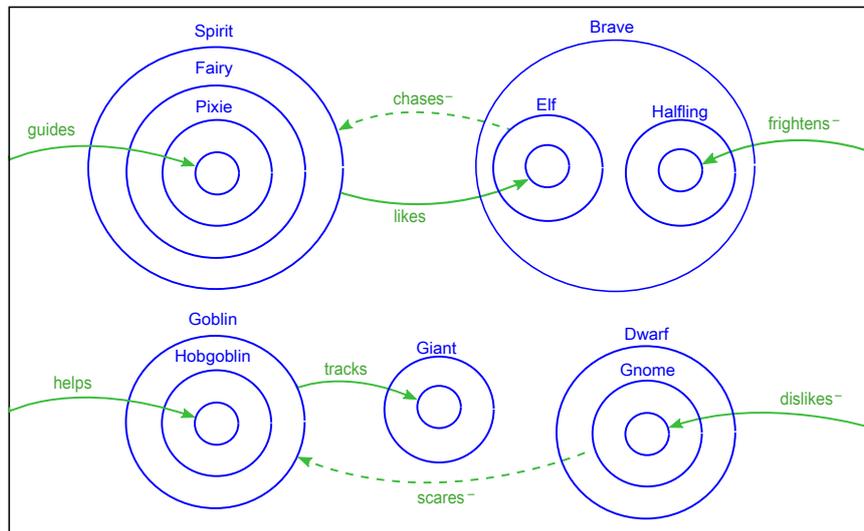


Figure 10. A more complex concept diagram.

Beyond empirical research, it is important to develop tools to support the use of these diagrams that are fully integrated with existing software. To this end, work has

begun on implementing a plug-in for WebProtege that allows users to draw concept and property diagrams, automatically converting them to OWL and displaying the OWL axioms in the standard WebProtege interface. It will be particularly difficult to implement translations from OWL to diagrams, however. A major challenge is define efficient translation algorithms that produce effective diagrams, rather than arbitrary diagrams, which are semantically equivalent to the given OWL specification.

Acknowledgements

This paper summarises research that has been undertaken by members of the Visual Modelling Group at the University of Brighton, in particular by the author in collaboration with John Howse, Eisa Alharbi, Jim Burton, Aidan Delaney, and Ali Hamie alongside Peter Chapman at Edinburgh Napier University. Michael Compton (formally at CSIRO) is also acknowledged for his contribution to the design of property diagrams and leading the implementation of the WebProtege plug-in mentioned in section 5.

References

- [1] OWL. <http://www.w3.org/TR/owl2-overview/>. accessed April 2014.
- [2] Protégé. <http://protege.stanford.edu/>. accessed April 2014.
- [3] E. Alharbi, J. Howse, G. Stapleton, and A. Hamie. Evaluating diagrammatic patterns for ontology engineering. In *Diagrams 2016 (accepted)*. Springer, 2016.
- [4] E. Alharbi, J. Howse, G. Stapleton, and A. Hamie. Helping people understand ontologie. In *in preparation*, 2016.
- [5] C. Gurr. Effective diagrammatic communication: Syntactic, semantic and pragmatic issues. *Journal of Visual Languages and Computing*, 10(4):317–342, 1999.
- [6] M. Horridge, N. Drummond, J. Goodwin, A. Rector, R. Stevens, and H. Wang. The Manchester OWL syntax. In *OWLed*, volume 216, 2006.
- [7] J. Howse, G. Stapleton, K. Taylor, and P. Chapman. Visualizing ontologies: A case study. In *International Semantic Web Conference*, pages 257–272. Springer, 2011.
- [8] A. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, and C. Wroe. Designing user interfaces to minimise common errors in ontology development: The CO-ODE and Hy-OntUse projects. In *Proceedings of the UK e-Science All Hands Meeting*, volume 2004, pages 493–499, 2004.
- [9] A. Rector, N. Drummond, M. Horridge, J. Rogers, H. Knublauch, R. Stevens, H. Wang, and C. Wroe. OWL pizzas: Practical experience of teaching OWL-DL: Common errors & common patterns. In *Engineering Knowledge in the Age of the Semantic Web*, pages 63–81. Springer, 2004.
- [10] R. Schwitter and M. Tilbrook. Controlled natural language meets the semantic web. In *Australasian Language Technology Workshop*, volume 2004, pages 55–62, 2004.
- [11] G. Stapleton, J. Howse, P. Chapman, A. Delaney, J. Burton, and I. Oliver. Formalizing concept diagrams. In *19th International Conference on Distributed Multimedia Systems*, pages 182–187, 2013.
- [12] T. Hou, P. Chapman, and A. Blake. Antipattern comprehension: An empirical evaluation. In *9th Conference on Formal Ontology in Information Systems*. IOS Press, 2016.
- [13] P. Warren, P. Mulholland, T. Collins, and E. Motta. The Usability of Description Logics. In *The Semantic Web: Trends and Challenges*, volume 8465 of *LNCS*, pages 550–564. Springer, 2014.