

# From ACE to OWL and from OWL to ACE

Kaarel Kaljurand

University of Zurich, University of Tartu  
kalju@ififi.unizh.ch

**Abstract.** We describe ongoing work on a bidirectional mapping between Attempto Controlled English (ACE) and OWL DL. ACE is a well-studied controlled language, with a parser that converts ACE texts into Discourse Representation Structures (DRS). We show how ACE can be translated into OWL DL (by using the DRS as interlingua) and how OWL DL can be verbalized in ACE. This mapping renders ACE an interesting companion to existing OWL front-ends.

## 1 Introduction

Existing OWL tools (Protégé<sup>1</sup>, SWOOP<sup>2</sup>, etc) are user-friendly graphical editors, but for complex class descriptions they require the user to possess a large knowledge of Description Logics (DL). E.g. [7] list the problems that users encounter when working with OWL DL and express the need for a “pedantic but explicit” paraphrase language.

To answer this need, we envision a text-based system that allows the users to express the ontologies in the most natural way — in natural language. Such a system would provide a natural syntax for logical constructions such as disjointness or transitivity, i.e. it would not use keywords but instead a syntactic structure to represent those complex concepts. It would also hide the sometimes artificial distinction between an ontology language and a rule language. The system would be tightly integrated with an OWL DL reasoner, but the output of the reasoner (if expressed in OWL DL as a modification of the ontology) would again be verbalized in natural language, so that all user interaction takes place in natural language and the central role in the system is carried by plain text.

As a basis of the natural language, we have chosen Attempto Controlled English (ACE), a subset of English that can be converted through its DRS representation into first-order logic representation and automatically reasoned about (see [1] for more information). The current version of ACE offers language constructs like countable and mass nouns; collective and distributive plurals; generalized quantifiers; indefinite pronouns; negation, conjunction and disjunction of noun phrases, verb phrases and sentences; and anaphoric references to noun phrases through proper names, definite noun phrases, pronouns, and variables. The intention behind ACE is to minimize the number of syntax and interpretation rules

---

<sup>1</sup> <http://protege.stanford.edu>

<sup>2</sup> <http://www.mindswap.org/2004/SWOOP/>

needed to predict the resulting DRS, or for the end-user, the reasoning results. At the same time, the expressivity of ACE must not suffer. The small number of ACE function words have a clear and predictable meaning and the remaining content words are classified only as verbs, nouns, adjectives and adverbs. Still, ACE has a relatively complex syntax compared to the OWL representation e.g. in the OWL Abstract Syntax specification ([6]), but as ACE is based on English, its grammar rules are intuitive (already known to English speakers) and experiments show that ACE can be learned in a few days.

Some existing results show the potential and the need for a natural language based interface to OWL. [3] paraphrase OWL class hierarchies but their target language is not a controlled language and cannot be edited and parsed back into a standard OWL representation. [8] propose writing ontologies in a controlled language, but do not provide a natural syntax for writing TBoxes.

Our work tries to overcome these shortcomings and addresses the following issues:

- Show that there is a mapping from a subset of ACE (which we call OWL ACE) into a syntactic subset of OWL DL (i.e. a subset which does not use all the syntactic constructs in OWL DL but is still capable of expressing everything that OWL DL can express).
- Show that the two involved subsets and the mapping from one to the other are easy to explain to the users. This means that the entailment and consistency results given by the OWL DL reasoners "make sense" on the ACE level.
- Show that there is a mapping from the syntactic subset of OWL DL into OWL ACE. This mapping (which can be called a verbalization) must, again, be easily explainable.
- Implement a converter from OWL DL to the chosen syntactic subset of OWL DL. By this, we will be able to handle all OWL DL ontologies on the web.
- If needed, extend ACE to provide a more natural syntax or more syntactic variety for expressing the OWL DL constructs.
- Extend the verbalization process to target a richer syntactic subset of OWL ACE.
- Extend all the aspects of this mapping in order to be compatible with future standards of OWL DL, e.g. OWL 1.1 ([5]) or extensions of it, e.g. SWRL ([4]).

So far, we have focused on the first 3 steps. In the following, we describe a mapping from OWL ACE to OWL DL (in RDF/XML syntax), the problems encountered, the OWL ACE subset and the verbalization of OWL DL.

## 2 From ACE to OWL

The following figure shows the DRS corresponding to the ACE text "Bill who is a man likes himself. Bill is William. Every businessman who owns at least 3 things is a self-made-man or employs a programmer who knows Bill." (Note

that the example is somewhat artificial to demonstrate concisely the features of OWL DL as expressed in ACE.)

```
[A, B, C, D, E, F]
object(A, atomic, named_entity, person, cardinality, count_unit, eq, 1)-1
named(A, Bill)-1
object(C, atomic, man, person, cardinality, count_unit, eq, 1)-1
predicate(E, state, be, A, C)-1
predicate(B, unspecified, like, A, A)-1
object(D, atomic, named_entity, person, cardinality, count_unit, eq, 1)-2
named(D, William)-2
predicate(F, state, be, A, D)-2
  [G, H, I]
  object(H, atomic, businessman, person, cardinality, count_unit, eq, 1)-3
  object(G, group, thing, object, cardinality, count_unit, geq, 3)-3
  predicate(I, unspecified, own, H, G)-3
  =>
  []
  [J, K]
  object(J, atomic, self_made_man, person, cardinality, count_unit, eq, 1)-3
  predicate(K, state, be, H, J)-3
  v
  [L, M, N]
  object(L, atomic, programmer, person, cardinality, count_unit, eq, 1)-3
  predicate(N, unspecified, employ, H, L)-3
  predicate(M, unspecified, know, L, A)-3
```

The DRS makes use of a small number of predicates, most importantly *object* derived from nouns and *predicate* derived from verbs. The predicates share information by means of discourse referents (denoted by capital letters) and are further grouped by embedded DRS-boxes, that represent implication (derived from ‘every’ or ‘if... then...’), negation (derived from various forms of English negation), and disjunction (derived from ‘or’). Conjunction — derived from relative clauses, explicit ‘and’, or the sentence end symbol — is represented by the co-occurrence in the same DRS-box.

The mapping to OWL DL does not modify the existing DRS construction algorithm but only the interpretation of the DRS. It considers everything in the toplevel DRS to denote individuals (typed to belong to a certain class), or to denote relations between individuals. Individuals are introduced by nouns, so that propernames map to individuals with type *owl:Thing* and common nouns to an anonymous individual with the type derived from the corresponding noun (e.g. class *Man*). Properties are derived from transitive verbs and transitive adjectives (‘fond of’, ‘taller than’). Special meaning is assigned to the copula ‘be’ which introduces an equality between individuals.

An embedded implication-box introduces a *subClassOf*-relation between classes — the head of the implication maps to the class description, the body to its superclass description. Transitive verbs and adjectives introduce a property restriction with *some ValuesFrom* a class denoted by the object of the verb or adjective,

and the copula introduces a class restriction. Co-occurrence of predicates maps to *intersectionOf*. Negation and disjunction boxes introduce *complementOf* and *unionOf*, respectively. Any embedding of them is allowed. The plural form of the word ‘thing’ allows to define cardinality restrictions. Thus our DRS has the following meaning (in Description Logics notation):

bill  $\in$   $\top$   
 m1  $\in$  Man  
 william  $\in$   $\top$   
 bill = m1  
 bill = william  
 likes(bill, bill)

(Businessman  $\sqcap$  owns  $> 3$ )  $\sqsubseteq$   
 (SelfMadeMan  $\sqcup$  ( $\exists$  employs (Programmer  $\sqcap$  ( $\exists$  knows {bill}))))

Note that an ACE construct like “A man who owns a dog likes an animal.” describes relationships between individuals and not classes, since the corresponding DRS does not have any embedded DRSs. In full English, this sentence is ambiguous by also having a reading which relates classes. In ACE, one would have to use ‘every’ instead of ‘a’ to get this reading.

OWL ACE allows properties to have superproperties. A superproperty (e.g. ‘likes’) for a given property (e.g. ‘loves’) can be defined as:

Everybody who loves somebody likes him/her.

Describing the transitivity of properties and inverse properties is quite “mathematical” in ACE, but there does not seem to be a better way in natural languages, unless one defines a keyword such as ‘transitive’ which then has to be explained to the average users. Consider e.g.

If a thing A is taller than a thing B and B is taller than a thing C then A is taller than C.

If a thing A is taller than a thing B then B is shorter than A. If a thing A is shorter than a thing B then B is taller than A.

Note that property definitions make use of indefinite pronouns (‘everybody’, ‘somebody’) or a noun ‘thing’, which all map to *owl:Thing*.

The current mapping does not target all the syntactic variety defined in the OWL DL specification, e.g. elements like *disjointWith* or *equivalentProperty* cannot be directly expressed in ACE, but their semantically equivalent constructs can be generated.

### 3 Problems

Now we look at some of the problems that we have encountered when implementing the mapping from ACE to OWL DL.

Complex class descriptions as arguments to *someValuesFrom* are difficult to map to OWL DL, since the DRS representation resembles more a rule language than a DL style property restriction. Also, *allValuesFrom* cannot be created in the most natural way because ACE lacks function words like ‘only’ or ‘nothing but’.

- \*Every carnivore eats only meat.
- \*Every carnivore eats nothing but meat.

To express this meaning, the user can use double negation or a rule-like construction.

- Every carnivore does not eat something that is not a meat.
- If a carnivore eats something then it is a meat.

The ACE negation does not generate an implication-box, but for class descriptions like “No man is a woman.” it would be desirable. Therefore, we first convert the negation-box into an implication-box (containing a negated *then-part*).

The fact that *inverseOf* is symmetrical is also difficult to implement because the ACE-way of expressing this creates two implication-boxes which have to be handled as one unit in the mapping.

Currently, there is no support for enumerations (*oneOf*). One possibility would be to extend ACE with NP disjunction.

- Every student is John or Mary or Bill.
- Everybody likes John or Mary or likes John or Bill.
- Everybody who is John or Bill is a man and is a student.

Also, at this point, ACE has no support for datatype properties. And finally, metalevel constructions such as URIs, imports, annotation properties, versioning, etc, which essentially make OWL DL a Semantic Web language cannot be cleanly expressed in ACE.

## 4 Explaining OWL ACE

Some restrictions to OWL ACE compared to full ACE are easy to explain: there is no support for intransitive and ditransitive verbs, prepositional phrases, adverbs, intransitive adjectives, and most forms of plurals.

In addition, there are constraints on the DRS structure which might be difficult to explain to the average user. Currently, an implication-box is only allowed to occur at a toplevel DRS. Disjunction, however, is not allowed to occur at the toplevel. Negation at the toplevel is handled by converting it first into an implication, or alternatively, as a negation of the equivalence of individuals. A further restriction requires the predicates in the implication-box to share one common discourse referent as the subject argument, unless the subject is directly or indirectly an object of a predicate that binds it to the common subject. This allows

us to exclude sentences like “If a man sees a dog then a cat sees a mouse.” but to include sentences like “If a man sees a dog that sees a cat then the man sees a mouse.” The first sentence does not seem to map nicely to OWL DL but instead to a more powerful rule language (such as SWRL). Also, no subject can occur as an object in an embedded if-then box (“Every man hates a dog that bites the man.”)

## 5 From OWL to ACE

The mapping in the opposite direction must handle all OWL DL constructs, some of which the ACE-to-OWL mapping does not produce. Also, it involves parsing RDF, which is the normative syntax for OWL DL. Another issue is raised by the naming conventions used for OWL classes and properties. E.g. OWL ontologies can contain class names like *SpicyPizza*, *MotherWith3Children* and property names like *accountName*, *brotherOf*, *isWrittenBy*. OWL ACE on the other hand would prefer classes to be named by singular nouns and properties by transitive verbs or adjectives.

So far, we have implemented a simple prototype in XSLT, which directly generates ACE. An alternative would target the DRS instead, and use an existing general mapping from the DRS to a canonical ACE form [2].

Currently, the ACE representation ends up being quite repetitive and unordered. For large ontologies this might become a problem and a more complex strategy is needed. Consider e.g. the following sentences.

Every professor who teaches a class is a teacher.

If there is a professor and he teaches a class then he is a teacher.

If there is a professor P and P teaches a class then P is a teacher.

Those sentences are equivalent, as far as ACE is concerned. Still, one could argue that some of those sentences are more readable than others, e.g. the *every*-construction with a relative clause is more readable than the *if-then* constructions with full clauses. On the other hand, relative clauses cannot express more complex structures (without causing ambiguity in the output), thus the more general *if-then* construction must be used. A flexible ACE generation system could use relative clauses in case they allow to correctly express all the references in the DRS and revert to using *if-then* sentences in case a more flexible reference system is needed.

Note also, that a variety of different verbalizations can be achieved by changing the input ontology with a reasoner which restructures the ontology and/or modifies it by adding/removing certain (possibly redundant) information. I.e. we could provide a relatively direct OWL-to-ACE mapping, but use a reasoner to customize the verbalization procedure for our needs.

## 6 Future work

The current mapping lacks support for datatype properties and enumerations. Also, *allValuesFrom* cannot be directly generated, but its semantics can be

achieved by using double negation. We will add support of those constructs along with support of proposed extensions to the current version of OWL DL, such as qualified cardinality and local reflexivity restrictions. Some of those changes require modification of the ACE syntax. ACE also needs support for namespaces, at least at the tokenizer level, to be called a Semantic Web language.

We will also study if the OWL ACE subset is easier or harder to teach to the users than full ACE.

## References

1. Attempto project. Attempto website, 2006. <http://www.ifi.unizh.ch/attempto>.
2. Norbert E. Fuchs, Kaarel Kaljurand, and Gerold Schneider. Deliverable I2-D5. Verbalising Formal Languages in Attempto Controlled English I. Technical report, REVERSE, 2005. <http://reverse.net/deliverables.html>.
3. Daniel Hewlett, Aditya Kalyanpur, Vladamir Kovlovski, and Chris Halaschek-Wiener. Effective Natural Language Paraphrasing of Ontologies on the Semantic Web. In *End User Semantic Web Interaction Workshop (ISWC 2005)*, 2005.
4. Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosz, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. W3C Member Submission 21 May 2004. Technical report, W3C, 2004. <http://www.w3.org/Submission/2004/SUBM-SWRL-20040521/>.
5. Peter F. Patel-Schneider. The OWL 1.1 Extension to the W3C OWL Web Ontology Language. Editor's Draft of 19 December 2005. Technical report, 2005. <http://www-db.research.bell-labs.com/user/pfps/owl/overview.html>.
6. Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks. OWL Web Ontology Language Semantics and Abstract Syntax. W3C Recommendation 10 February 2004. Technical report, W3C, 2004. <http://www.w3.org/TR/owl-semantics/>.
7. Alan L. Rector, Nick Drummond, Matthew Horridge, Jeremy Rogers, Holger Knublauch, Robert Stevens, Hai Wang, and Chris Wroe. OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns. In Enrico Motta, Nigel Shadbolt, Arthur Stutt, and Nicholas Gibbins, editors, *Engineering Knowledge in the Age of the Semantic Web, 14th International Conference, EKAW 2004*, volume 3257 of *Lecture Notes in Computer Science*, pages 63–81. Springer, October 5–8 2004.
8. Rolf Schwitter. Controlled Natural Language as Interface Language to the Semantic Web. In *2nd Indian International Conference on Artificial Intelligence (IICAI-05)*, Pune, India, December 20–22 2005.