RC23755 (W0510-142) October 21, 2005
Computer Science

IBM Research Report

A Reusable Ontology for Fluents in OWL

Christopher Welty
IBM Research Division
Thomas J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598

Richard Fikes, Selene Makarios
Stanford University

Research Division
Almaden - Austin - Beijing - Haifa - India - T. J. Watson - Tokyo - Zurich

A Reusable Ontology for Fluents in OWL

Christopher Welty, Richard Fikes, and Selene Makarios
welty@us.ibm.com, fikes@ksl.stanford.edu, makarios@ksl.stanford.edu

Abstract

A critical problem for practical KR is dealing with relationships that change over time. This problem is compounded by representation languages such as OWL that are biased towards binary relations, even when the relationships that vary with time are binary. We discuss several approaches to this problem in OWL, and focus on the advantages of a four-dimensionalist (perdurantist) solution, which allows us to use more of the expressive power of the language.

Content Areas: Ontologies, Semantic Web, Time, Fluents.

Introduction

A critical problem for practical KR is dealing with information that changes over time. This problem is compounded by representation languages that are biased towards binary relations. Such languages include OWL, RDF, description logics in general, and frame-based and object-oriented languages. Even when the relationships we wish to represent are binary, such as being a member of an organization or living at an address, the fact that they may change with time complicates matters enough that many ontologies implemented in these representation languages ignore the problem of time completely.

In this paper, we will review different approaches to representing relationships that change with time in OWL, and describe how our needs led us to adopt a four-dimensionalist approach to this problem. We will then discuss some practical issues involved with this choice, and how we interoperate with systems that represent the same information differently.

Tools of Formal Ontology

The field of ontology begins with the premise that things exist in the universe, and has the goal of describing what kinds of things there are in a way that – as much as possible – removes our own human perceptions and biases [Smith and Welty, 2001]. At the most basic level, the field is divided into two groups distinguished by their solution to the problem of diachronic identity: i.e., how do we logically account for the fact that the "same" entity appears to be "different" at different times?

The 3D view, sometimes called the endurantist view, maintains a basic distinction between *endurants* (physical objects such as people, chairs, etc.) and *occurrants* (events such as sitting on a chair or going to work). The difference between the two is that endurants are *wholly present* at all times during which they exist, while occurrants have *temporal parts* that exist during the times the entity exists. The problem of diachronic identity of endurants is addressed by establishing certain essential properties of entities through which they are identified – these properties are essential and thus must always hold. Other properties, such as having brown hair (or having hair at all), are understood to be changeable. This approach has the problem, however, that Leibniz's Law (i.e., X and Y are identical if and only if they share all and only the same properties) must be qualified with this distinction.

The 4D view, sometimes called the perdurantist view, maintains that all entities are perdurants. The idea is that, on a universal scale, even the lifetime of a planet is a simple event, and the prescription that physical objects are somehow different is a product of our own very fine-grained perspective on time and space. Thus, all entities have temporal parts and can be thought of intuitively as four dimensional "spacetime worms" whose temporal parts are slices of the worm [Sider, 2001]. The problem of diachronic identity becomes trivial since entities are four dimensional, and the notion of change is accounted for simply by giving different properties to different temporal parts of an entity so that Leibnitz's law always holds. This approach has the problem, however, that determining what is an entity is rather arbitrary; in fact any mere collection of matter over time can be an entity.

The 4D approach is clearly not something that immediately appeals to common sense. However, its proponents claim that special relativity

necessitates it as a correct view of reality. It was first introduced into computer science by Hayes in his seminal work on Naïve Physics [Hayes, 1985]. Its history in philosophy dates back over a hundred years, though Willard Quine [1950] is often associated with its origins, and David Lewis [1971] is probably the first to have formalized it in a logical theory.

In this paper we will not attempt to offer evidence in favor of or contrary to these two views as basic ontologies of the universe, as has been done elsewhere [Hayes, et al, 2002]. Rather, we wish to show that understanding these ontological choices and their tradeoffs gives us another tool to use when solving a practical problem, just as understanding the behavior of different sorting algorithms with respect to the distribution of data gives the software engineer choices when designing software.

Background

In our work, we are using OWL to represent an ontology of entities and relations that are discussed in general news articles and may be of interest to analysts in areas such as business, finance, and government intelligence. The data that instantiates the ontology is automatically extracted though text analysis.

Choosing OWL to represent our ontologies gave us the ability to easily build interoperable systems to produce, reason over, visualize, etc., this data. Further, the large quantity of data we are producing requires highly optimized reasoning components, which are available for several subsets of OWL (e.g. Racer [Haarslev and Möller, 2001] and Pellet [Parsia and Sivrin, 2004]). We were also able to use off-the-shelf editors (e.g. Protégé [Noy, et al, 2000]) and user-interfaces (e.g. TGViz [Alani, 2003]). Finally, the ability to view the resulting data as a simple graph has proven very important in demonstrating the technology because end users find such visualizations to be natural and easily understandable.

The ontologies we use are human created, and the vast amounts of information we reason with is generated automatically from news articles. Briefly, we use large-scale information extraction techniques with a specially designed integration component that maps the results of information extraction to instances in the knowledge-base. This extraction is fairly shallow, in particular we avoid overloading the knowledge-base with syntactic information and parse trees and map only the named entities and binary relations between them that are identified by text analysis components. For example, we would expect that text analysis of the following sentence:

> "IBM appointed a new CEO, Sam Palmisano, a graduate of The Johns Hopkins University."

would generate the following data (we employ here the OWL abstract syntax [Patel-Schneider, et al, 2004]):

```
Individual(IBM type(Company))
Individual(JohnsHopkinsUniversity type(University))
Individual(SamPalmisano type(Person)
    value(ceoOf IBM)
    value(alumnusOf JohnsHopkinsUniversity))
```

There are a number of difficult research problems in doing such text analysis and in mapping the results to OWL that we will not address in this paper. The principle point here is that producing these binary relationships (shown as the value and type statements) is a standard practice in this community, and we wished to make use of this existing work to populate a knowledge-base.

Representation Problems

The main shortcoming of the representation used in the example above is that it is *synchronic*, i.e. it refers to only one point in time. In many domains, and indeed in natural human understanding, these relationships are *diachronic*, i.e. they vary with time. For example, consider this sentence:

> "Sam Palmisano was named chief executive officer of the IBM Corporation effective March 1, 2002."

The sentence clearly indicates that Sam Palmisano was not always the CEO of IBM, and that the relationship *ceoOf*(sam,ibm) will be true for some period of time that began on March 1, 2002.

Moving from a synchronic to a diachronic representation, however, creates a significant representation problem if we are to use RDF and OWL, even if we limit ourselves to binary relations that vary with time. In general, this problem is one of representing *fluents*. According to McCarthy and Hayes [1969], a fluent is a function that maps from objects and situations to truth. We will simplify this notion slightly and consider fluents to be relations that *hold* within a certain time interval and not in others. The text analysis we use is based on TimeML [Ingria and Pustejovsky, 2002], and we map these results into the OWL-Time ontology [Hobbs and Pan, 2004] for time points and

intervals. This paper remains neutral to these representations, however, and any time ontology that provides an adequate notion of time intervals will do.

The most common way to represent a binary fluent in FOL is to simply add a time argument to the binary predicate, as in McCarthy&Hayes [1969], and described as a standard practice already at that time. Thus the synchronic *ceoOf*(Sam,IBM) becomes the diachronic *ceoOf*(Sam, IBM, t_1), where t_1 is a time interval that begins on March 1, 2002. This leaves us with a ternary predicate, which cannot be represented in OWL unless we move to reification, which is discussed below, or to one of the other approaches described by the W3C Semantic Web Best Practices working group [Noy and Rector, 2005].

In the original formulation of fluents discussed by McCarthy&Hayes, another choice presented is to add a meta-logical predicate to relate the relationship to a time interval. That choice has become the standard in most formalizations (e.g., situation calculus and Allen's Interval Temporal Logic). In our example, this would be *holds*(*ceoOf*(Sam, IBM), t_1). While this seems *prima facie* to be within the limits of OWL since it uses only binary predicates, OWL does not allow relationships themselves to participate in relationships, which this would require.

Another important factor in our work is the requirement that the representation of fluents be fully round-trip transformable with a representation in KIF that uses the *holds* predicate. Within this representation, we perform more advanced temporal reasoning with a special purpose reasoner plug-in for JTP [Fikes, et al, 2003].

Solution Approaches for OWL

The problem of representing fluents in a binary-limited representation is not new. Here we outline some of the choices we considered, and evaluate their usefulness. A more comprehensive survey of techniques for representing higher arity predicates in OWL can be found at [Noy and Rector, 2005].

Reification

Independent of representing change, the most common approach to dealing with predicates of higher arity than 2 in languages like OWL is to *reify* the relationships – that is, turn each tuple in the extension of a relation into an object that itself has binary relationships, typically called *roles,* identifying all the elements of the tuple. A class, which we will call the *reified relationship class*, is created for each relation, whose extension includes all the objects representing tuples in the extension of the corresponding relation.

For example, the *ceoOf*(Sam, IBM, t_1) relationship would be represented as:

```
Individual(IBM type(Company))
Individual(JohnsHopkinsUniversity type(University))
Individual(SamPalmisano type(Person))
Individual(Rel1 type(CeoOf)
      value(CeoOf:ceo SamPalmisano)
      value(CeoOf:company IBM)
      value(CeoOf:holds t1))
```

Note that the ":" character in this example is used simply for visual purposes, and while it has a syntactic meaning in OWL, it can be understood in these examples to simply make the name of each role of a relationship unique to the reified relationships of that type.

Reification has a number of known problems that are outlined in more detail in [Noy and Rector, 2005]. None of these are serious, but they are important to be aware of. In brief, they are:

- *Proliferation of objects.* For every tuple a new object is created, along with an additional binary relationship for every element of the tuple. For the type of relations we are interested in (binary relations that change with time), we end up creating an object and four relationships (including the type relationship) for every tuple. This is a mild concern as our application extracts many relationships per document, and we would like to scale the corpus to at least one million documents.
- *Redundant Objects.* Since relationship reifications are objects, it is possible to create multiple objects that reify the same relationship, where in other languages that permit n-ary predicates there would only be one tuple in the extension of the corresponding relation. This can be problematic when using reasoners (such as Racer [Haarslev and Möller, 2001]) that make a unique names assumption, as these multiple objects if present can be counted twice when satisfying counting axioms such as cardinality constraints.
- *Confusing ontology.* When classes and their instances are defined in an ontology, it is important to define what they refer to. The presence of reified relation classes and roles can create a confusion between events and fluents if one is not careful.
- *Limited use of OWL reasoning.* The use of OWL language constructs becomes severely limited with reified relationships.

The final point is of particular interest here, and deserves further discussion, as this will be the primary problem our solution will address.

In our ontology, and the circumstances are quite common, the vast majority of our fluents are binary relations that vary with time. The relationships themselves are, again, extracted automatically through text analysis. Some examples are *ownerOf, managerOf, ceoOf, employeeOf, memberOf, locatedAt, colocatedWith*, etc. Each of these relations holds between any two individuals for some period of time, and does not hold for other periods of time.

Resorting to reification to add the time argument to each binary relationship prevents us from employing many of the most practically useful OWL operators in describing the intended semantics of the relations, and thus drastically limits the utility of OWL reasoning.

For example, empirical studies have shown that simple reasoning about inverses can have the highest impact, in terms of relevance to the user, on improvements in recall [Welty, 1998]. For most binary relations, the inverse is potentially as important, even if it is not explicitly stated. For example, the inverse of the *ownerOf* relation is *hasOwner*. There is no *a priori* way to determine which direction may be of more use to an end user. Reification prevents us from being able to express in OWL the relation inverses, thus we are stuck with whatever direction is stated in text.

Reification also prevents usage of the OWL operators *transitive, symmetric, functional*, and *inverseFunctional* on the intended relation.

Local range restrictions on properties in the reified relation class must also be understood to be global restrictions on the intended relation. To express the equivalent of local range restrictions on the intended relation, one must use nested restrictions on the role inverses. For example, "A person is managed by a person" would be expressed as:

```
ObjectProperty(ManagerRel:manages
    inverseOf(ManagerRel:managedBy))
Class(Person partial
    Restriction(ManagerRel:managedBy
        allValuesFrom(
            restriction(ManagerRel:manager
                allValuesFrom(Person)))
```

ManagerRel:manages and *ManagerRel:manager* are the roles of the relation.

Finally, for representing binary fluents, cardinality and value restrictions on the role inverses end up restricting the intended relation for all time. In other words, we cannot express in OWL that a person can have at most one manager *at a time*. The only thing we can say is:

```
Class(Person partial
    Restriction(ManagerRel:managedBy
        maxCardinality (1)))
```

which says that every instance of *Person* can only ever have one manager. In most cases where such an axiom may be useful, e.g. the *hasMother* relation, it's not clear the relation should be a fluent at all. At the very least, in our ontologies we could not find a use for cardinality or value restrictions on the role inverses.

Temporal Description Logics

Given our decision to use OWL, whose semantics is borrowed from description logics and which has a well-defined description logic fragment, it may seem natural to pursue temporal description logics in order to represent fluents. A good survey of temporal description logics can be found at [Artale and Franconi, 2001].

Like all description logics, temporal description logics restrict the operators of the language, and make critical choices in the formalization, in order to keep all reasoning problems in the language decidable. Following in the style of this community, temporal description logics do not have variables in the syntax, thus making quantification implicit. Most importantly for our purposes, temporal description logics reduce to a modal logic, in which times (or situations) are removed from the syntax and instead are part of the implicit quantification with only one ordering relation.

The result is that we are unable to talk about what is true at any particular time in a temporal DL, we can only quantify over past and future times. Furthermore, we can not use the full Allen calculus, which is critical to our application.

The Four-Dimensional Approach

The basic idea of a 4D ontology is to consider entities to exist in time the way material objects exist in space – they *occupy* spacetime. These 4D entities (perdurants) have temporal parts that represent the entity during some time interval – 4D entities have arbitrarily many temporal parts. When two entities participate in some fluent, the fluent holds between temporal parts of those entities. Often perdurants are described as 4D worms whose temporal parts are slices of the

worm. We present below a high-level and reusable ontology in OWL for 4D fluents:

```
(Ontology 4dFluents
   (Class TimeSlice)
   (DisjointClasses TimeSlice TimeInterval)
   (Property fluentProperty  Symmetric
      (domain TimeSlice)
      (range TimeSlice))
   (Property tsTimeSliceOf Functional
      (domain TimeSlice)
      (range  complementOf(TimeInterval)))
   (Property tsTimeInterval Functional
      (domain TimeSlice)
      (range TimeInterval)))
```

As is often the case with high-level ontologies, there are more intended semantic constraints than can be expressed in OWL. To begin with, we intend that time slices be maximal with respect to the time interval of all fluents it participates in. That is, the time interval of a time slice is defined to be the duration of the fluent holding. Thus if a time slice participates in more than one fluent, they must hold for precisely the same interval. We also intend that two time slices of the same entity for the same interval are equal. In our system, it is the responsibility of the component that transforms the output of text analysis into the knowledge-base to enforce these semantics.

The class *TimeInterval* here is actually a stand-in for whatever class is provided in a time ontology. In our work we use a portion of the OWL-Time ontology without events, and we define:

```
(Ontology 4dFluentsOwlTime
   (Annotation owl:imports 4dFluents)
   (Annotation owl:imports OwlTime)
   (EquivalentClasses owlTime:Interval TimeInterval))
```

Given this ontology, we can define a simple ontology for our example:

```
(Ontology example1
   (Annotation owl:imports 4dFluentsOwlTime)
   (Class Person)
   (Class Company)
   (DisjointClasses Person Company)
   (Property ceoOf super(fluentProperty) inverseOf(hasCeo)
      (domain
         (restriction(tsTimeSliceOf (allValuesFrom Person))))
      (range
         (restriction(tsTimeSliceOf (allValuesFrom Company))))))
```

The *ceoOf* relation in our example holds between the temporal part of Sam Palmisano that begins on March 1, 2002, and the temporal part of IBM that begins at the same time:

```
Individual(SamPalmisano type(Person))
Individual(IBM type(Company))
Individual(IBM1 type(TimeSlice)
      value(timeSliceOf IBM)
      value(timeInterval t1))
Individual(SP1 type(TimeSlice)
      value(timeSliceOf SamPalmisano)
      value(timeInterval t1)
      value(ceoOf IBM1))
```

The first advantage of the 4D representation is that we can use the OWL inverse operator in the expected way, as shown in the example ontology. With this axiom, the data above would entail the *hasCeo* relation between IBM_1 and SP_1.

Transitivity and symmetry also have the expected meaning, allowing us to express many spatial axioms in the OWL ontology:

```
   (Property colocatedWith Symmetric)
   (Property locatedIn Transitive)
```

This allows us to use OWL reasoners rather than more computationally expensive ones to derive these relationships in the data.

Cardinality restrictions can also be expressed with this approach in a way that was useful for our domain. We had several intended relations whose semantics required temporally qualified cardinality, such as "a company has at-most one CEO at a time". This would be approximated as:

```
   Class(Company partial
      restriction(hasTimeSlice
         allValuesFrom(restriction(hasCeo
               maxCardinality(1)))))
```

This is only an approximation of the intended semantics as it does not prevent the unintended model in which a company has two overlapping time slices that have a *hasCeo* relation to different people, it only eliminates the cases in which a company has two values of the *ceoOf* relation during the same interval. Nevertheless, that is an improvement over any other approach to representing binary fluents in OWL, since it eliminates some unintended models. Note that this also depends somewhat on the ability to enforce the identity conditions on time slices discussed above, which cannot be represented in OWL.

The 4D approach is the worst among the approaches discussed here with respect to the proliferation of objects. A single fluent requires two extra objects (the time slices) and four triples. We are exploring heuristic techniques for pruning this information, depending on the reasoning task at hand, but in general the amount of data generated is a shortcoming.

Conclusion

Representing changing information is critical to practical Knowledge Representation and

Reasoning. OWL is intended to be a practical KR&R language, and its expressiveness is limited in favor of desirable computational properties. One limitation, the restriction to unary and binary predicates, creates an obstacle to representing relationships that change with time, even when the relationships themselves are binary. We have discussed these obstacles as well as the most common solution, reification of relationships, and shown that it further limits the use of what limited expressiveness OWL already has.

Our approach, which involves treating entities in the domain of discourse as four dimensional with temporal parts that participate in the relation, corresponds to an established ontological position in analytical metaphysics called perdurantism. The perdurantist approach offers several advantages, and most importantly increases the amount of OWL expressiveness that can be utilized to practically describe the semantics of a domain in which binary relationships change with time.

Acknowledgements

This work was supported in part by the Advanced Research and Development Activity (ARDA)'s Novel Intelligence for Massive Data (NIMD). The authors would also like to thank Pat Hayes for extensive discussions that helped clarify these ideas.

References

Alani, H. 2003. TGVizTab: An Ontology Visualization Extension for Protégé. In *K-Cap'03 Workshop on Visualization Information in Knowledge Engineering*, Sanibel Island.

Artale, Alessandro and Enrico Franconi. 2001. A Survey of Temporal Extensions of Description Logics. *Annals of Mathematics and Artificial Intelligence* (AMAI), Kluwer Academic Press, 30(1-4):171—210.

Fikes, Richard, Jessica Jenkins, and Gleb Frank. 2003. JTP: A System Architecture and Component Library for Hybrid Reasoning. *Proceedings of the Seventh World Multiconference on Systemics, Cybernetics, and Informatics*. Orlando, Florida.

Haarslev, Volker and Ralf Möller. 2001. High Performance Reasoning with Very Large Knowledge Bases: A Practical Case Study. *IJCAI-01 Proceedings.* Seattle:AAAI.

Hayes, P.J. 1985. The Second Naïve Physics Manifesto. In *Formal Theories of the Commonsense World.* Pp 1-36. Norwood, NJ: Ablex.

Hayes P.J., Fritz Lehmann and Chris Welty. 2002. *Endurantism and Perdurantism: An Ongoing Debate*. Edited by Adam Pease. http://ontology.teknowledge.com:8080/rsigma/dialog-3d-4d.html

Hobbs, Jerry R., and Feng Pan, 2004. An Ontology of Time for the Semantic Web', *ACM Transactions on Asian Language Information Processing*: **3**(1).

Ingria, Bob and James Pustejovsky. 2002. *TimeML: A Formal Specification Language for Events and Temporal Expressions*. http://www.cs.brandeis.edu/~jamesp/arda/time/timeMLdocs/TimeML12.htm.

Lewis, David. 1971, "Counterparts of Persons and their Bodies", *Journal of Philosophy*, 68: 203-211.

Parsia, B. and Sivrin, E. 2004. Pellet: an OWL-DL Reasoner. *ISWC 2004 Proceedings*.

Patel-Schneider, P, Ian Horrocks and Patrick Hayes. 2004. *OWL Web Ontology Language Semantics and Abstract Syntax.* W3C Recommendation. http://www.w3c.org/TR/owl-semantics/

Quine, W.V.O. 1950, "Identity, Ostension and Hypostasis", in *From a Logical Point of View*. Pp. 65-79. Cambridge: Harvard.

McCarthy, John and P.J. Hayes. 1969. Some Philosophical Problems from the Standpoint of Artificial Intelligence'. In D. Michie (ed), *Machine Intelligence:* **4**. New York: Elsevier.

Noy, N. F., W. Grosso, & M. A. Musen. 2000. Knowledge-Acquisition Interfaces for Domain Experts: An Empirical Evaluation of Protege-2000. *SEKE2000 Proceedings*. Chicago.

Sider, Theodore. 2001. *Four-Dimensionalism.* Oxford University Press. 2001

Smith, B. and C. Welty. 2001. Ontology: Towards a new synthesis. In *Formal Ontology in Information Systems*. Ongunquit:ACM Press.

Welty, C. 1998. Augmenting Abstract Syntax Trees for Program Understanding. *ASE-98 Proceedings.* Tahoe: IEEE CS Press.