# $CO_3L$: Compact $O_3F$ Language

Pedro Ramos
ADETTI / ISCTE
"We, the Body and the Mind" Group
Av. Forças Armadas, Lisboa, Portugal
+351 217 903 099
Pedro.Ramos@iscte.pt

Luis Botelho
ADETTI / ISCTE
"We, the Body and the Mind" Group
Av. Forças Armadas, Lisboa, Portugal
+351 217 903 099
Luis.Botelho@iscte.pt

## ABSTRACT

This paper presents CO3L, a compact, expressive and easy to use language for ontology representation. CO3L reflects the O3F model of ontology representation proposed elsewhere.

The proposed language enables the representation of basic ontological entities, their relationships, and arbitrary axioms of the domain. Ontological entities include classes, properties, methods, facets and types. Relationships include n-ary associations and inheritance. Axioms may be used to capture complex constraints and relations between entities, to define relational and functional methods, and to represent the effects of the execution of action methods in the world.

CO3L is based on the language of the first order logic, with explicit world states; the relational operator *State/1*, which is true of world state designators; the functional operator *Do/2*, which returns the world state designator resulting of the execution of an action in a given world state; and the modal operator *Holds/2* which is true of propositions satisfied in given world states.

## 1. INTRODUCTION

In open agent systems it is important to ensure interoperability at the several levels of the so-called communication stack. One such level pertains ontology technology and its uses. Often, agents need to dynamically use ontological information for the most diverse tasks, including service discovery [2] and message translation. Unfortunately there is no consensus regarding the requirements of ontology services for open agent systems. Some need just a simple way of describing the classes of the domain. Others need more than classes; they need to know the kinds of actions in a given domain. Yet others need detailed descriptions of the existing relations between input and output parameters of dynamic computations, or even of descriptions of the effects of performing domain actions.

Each of the above requirements has been handled to different degrees by the most common approaches to ontology representation and use. This has resulted in a heterogeneous landscape of ontology technologies that has not contributed to improve interoperability.

One way to go about this problem is by developing general, more abstract frameworks of which, the existing approaches can be seen as particular cases. This paper contributes to the advancement of one such a framework – O3F, the object-oriented ontology framework.

O3F [10], is a general framework for ontology representation that can be used to capture ontological information originally described in several formalisms including DAML+OIL [4], OWL [11], Ontolingua [5] and OKBC [3]. Additionally, O3F has several advantages in relation to these other approaches, namely the possibility of representing arbitrary axioms, which can only be found in Ontolingua and OKBC, and the possibility of declaratively representing action methods and their effects in the world, which is not offered by any other approach. Besides, [10] presents a significant advance in terms of the formal specification of several kinds of translation relations between ontologies, which were based on the concept of basic mapping.

Given the above, O3F is a natural candidate for ontology representation in web and agent based applications. However, the ontology representation language proposed in [10] uses long cumbersome expressions. This problem is a considerable handicap of the O3F framework because it will impair its effective use and dissemination in the agent and web communities.

The difficulty of using and understanding the proposed representation language derives from two main facts. First, it was directly shaped from the UML class diagram representing the O3F ontology representation model. Second, it straightforwardly adapts constructs used in the agent content language proposed in [1], which are not necessarily the best for an ontology representation language.

This paper presents a solution for the difficulties encountered in the ontology representation language proposed for the O3F representation framework, by attacking the two mentioned problems. The proposed solution preserves the full expressiveness of the language while drastically improving its readability. Furthermore, the representation power of the original O3F framework is also not affected.

Section 2 briefly describes the O3F original approach with special emphasis on the originally used representation language. Section 3 describes our approach to improve the readability and usability of the representation language. First, we show how it is possible to deviate from the original language constructs used in [10] without losing generality or expressiveness. Then, we present CO3L, the new ontology representation language for O3F. Section 4 analyses our approach in the scope of related work. Finally, section 5 presents conclusions and future work.

## 2. $O_3F$ FRAMEWORK

O3F [10] is a general object oriented framework for ontology representation. An ontology is composed of a set of basic entities, simple relationships between them and arbitrary domain-dependent axioms. Basic entities are classes, properties, facets, methods and types.

As in DAML+OIL, properties, methods and axioms have autonomous existence separated from the classes with which they might be related. The independence of properties from the classes with which they may be related enables O3F to capture ontologies originally represented in DAML+OIL and OWL.

The possibility to represent arbitrary axioms using first order logic enables O3F to capture ontologies originally represented in Ontolingua. Finally, the possibility to represent procedural methods is an advantage of O3F over their direct competitors.

Using the approach described in [10], for instance, it is possible to specify the class named Restaurant, the properties named *Name* and *Price*.

```
(instance (Class :name Restaurant) Class)
(instance (Property :name Name) Property)
(instance (Property :name Price) Property)
```

It is also possible to associate the attributes *RestauratName* and *MaxPrice* to the class *Restaurant*. Originally, in O3F, associations between classes were represented through instances of the special class *O3FRelation. O3FRelation* is an O3F class used in the language to represent all the O3F relations (e.g., Archetype_Attribute, Argument_Attribute).

The class *O3FRelation* has the attributes *name*, which is the name of a specific relation; *arguments*, which is the set of archetypes that are associated in a given relation; and *attributes*, which is the private attributes that belong to a relation. Each element in the set of arguments of a relation is an instance of the class RArgs (Relation Arguments), which is composed of the class of the archetype being associated (e.g., *Class* or *Property*), the key attribute of the archetype being associated (e.g., *name*) and the name of that archetype being associated (e.g., *Restaurant*). The following statement says that the property named *Name* is associated with the class *Restaurant*.

```
(instance
   (O3FRelation
    :name Archetype_Attribute
    :arguments (set
      (RArg
       :class Property
       :key name
       :object Name)
      (RArg
       :class Class
       :key name
       :object Restaurant))
    :attributes (set
      (Attribute
       :name Name
       :value RestaurantName)))
   O3FRelation)
```

An association between an archetype and a property defines an attribute of the archetype. The above association between the property *Name* and the Class *Restaurant* defines the attribute *RestaurantName* of the class *Restaurant*.

The example discussed in this section makes it clear that the specification of an ontology using the proposed language is a cumbersome process which results in a large set of long, difficult to read expressions. Since this process may have to be handwritten, it is better to make it easier and shorter. The next section shows our proposal to achieve this goal.

# 3. THE NEW $O_3F$ LANGUAGE

In this section we show how the O3F class architecture must be re-interpreted before it can be used as the basis for the design of the ontology representation language.

## 3.1 Representing $O_3F$ Basic Entities

Class, Property, Method, Type, Facet and other entities are classes of the O3F class diagram. Particular domain classes, properties, methods, types and facets are instances of those classes. Using the language proposed in [10], those instances are represented through the two-place relation *instance*. Here we propose to use one reserved word for specifying instances of each of these classes. The relations *Class*, *Property*, *Method*, *Type*, *Facet,* and *Axiom* will be used to specify the classes, the properties, the methods, the types the facets and the *Axioms* of the ontologies.

The statement (Ontology RestaurantOntology 'Agentcities Consortium') asserts the fact that RestaurantOntology is an ontology whose author is the Agentcities Consortium. The statement (Class Restaurant RestaurantOntology) means that Restaurant is a class of the RestaurantOntology). As another example, the statement (Property Price RestaurantOntology) means Price is the name of a property of the ontology RestaurantOntology. From now on the ontology will be omitted from the arguments list of the language statements, so that the explanations can stay focused.

## 3.2 $O_3F$ Class Diagram Re-Cast

The main problem with the interpretation of the O3F framework is the way associations are represented. Originally, all associations were represented using the special purpose class *O3FRelation* (see section 2). This policy, together with the way instances were specified using the two-place predicate *Instance/2* lead to lengthier, more complex and cumbersome expressions.

Here we propose to use one specific relation for each of the possible associations of the model. For example, in order to represent the attribute *RestaurantName* of the Class Restaurant it is not necessary to have a language we use the attribute statement (Attribute Restaurant RestaurantName), which contrasts with the long expression in section 2.

## 3.3 Arbitrary Axioms

Besides the relations specified in sections 3.1 and 3.2, it is also necessary to define a set of primitive operators used in the definition of arbitrary axioms.

Often the declaration of classes, properties, methods and relationships between them is not enough to capture the relations of the domain.

In order to represent arbitrary axioms in first order logic, it is necessary to have the usual logical connectives and quantifiers. This is not new when compared with other ontology representation languages such as KIF [6] in Ontolingua.

In order to define the effects of action methods (i.e., methods whose execution changes the state of the world), it is necessary to talk about states of the world, and to capture the execution of methods in the world along with the changes they produce.

We introduce the following operators:

*State/1* is a relational operator used to declare an identifier for a world state. State(<State Identifier>) means <State Identifier> is a state identifier.

*Holds/2* is a modal operator used to say that a given proposition holds in a given state of the world. Holds(<Proposition>, <State Identifier>) means <Proposition> is true in the state of the world identified by <State Identifier>. In the current approach, *Holds/2* is a modal operator instead of a relational operator as in [10], in order to avoid the reification of relations, logical operators and quantifiers to functions.

*Do/2* is a functional operator used to represent the state that results of the execution of a given method in a given state of the world. Methods are represented by terms called action identifiers. Do(<Method>, <State Identifier>) represents the sate of the world that results of the execution of the method <Method> in the state identified by <State Identifier>.

Given these operators it is possible to represent the effects of the execution of action methods and it is also possible to represent constraints over states of the world. The axioms written below represent the method BookFlight used to book a plane ticket in a travel agency. BookFlight takes the flight number (f), the date of departure (d), the flight class (c) and the identification of the client (x).

$\forall s, f, d, x\ \exists t\ State(s) \wedge Holds(\neg(t \in Ticket) \wedge x \in Client \wedge <c , f> \in Flight\_Classes, s) \Rightarrow$
  Holds( $t \in Ticket \wedge <x , t> \in Client\_Ticket \wedge$
  $<t , <c , f >> \in Flight\_Classes\_Ticket \wedge t.Date = d,$
  Do(Ticket.BookFlight(f, d, c, x), s))

There is a t for which if it is not a ticket in situation s, it becomes a ticket ($t \in Ticket$) in the situation resulting of booking a ticket.
Moreover, the booked ticket becomes associated with the client specified in the reservation ($<x , t> \in Client\_Ticket$); the booked ticket becomes associated with the pair of the class and flight specified in the reservation ($<t, <c , f >> \in$ Flight_Classes_Ticket); and the date of the ticket becomes the date specified in the reservation (t.Date = d).

The axioms of the ontology are expressions in which the logic and other general-purpose symbols such as quantifiers and reserved operators belong to the language, and the domain symbols such as classes (e.g., Ticket) and associations (e.g., Client_Ticket, Flight_Classes) must belong to the ontology.

Here, we have used the notation commonly used in textbooks about logic. This is not the actual syntax of CO3L, since CO3L is an abstract language (section3.4).

## 3.4 Compact $O_3F$ Language

This section presents examples of the abstract syntax of the O3F Representation Language. Basically, this is a first order logic, with modal operator *Holds/2*, and a set of reserved relational and functional operators. In [8] we present a simple ontology example using the UML notation. The example is a fragment of the Travel Agent scenario described in the OAS'03 Challenge Problem.

The provided abstract syntax defines an abstract language that may be instantiated through a variety of concrete syntaxes such as the S-Expression, the usual prefix first order logic syntax, and an XML syntax commonly used in web applications. The top-level symbol representing Compact O3F Language statements is OntologicalProposition, which can be an ontology declaration, an ontological entity declaration, an ontology relationship, an arbitrary axiom[1], or an instance definition.

| **Frame Name** | OntologicalProposition | Abstract |
|---|---|---|
| **Kind of** | CO3LExpression | |
| **Description** | Top level language statement | |

The frame OntologicalProposition is an abstract concept in the sense that it cannot be instantiated. OntologicalProposition is a CO3LExpression, thus the slot **kind-of** is CO3LExpression, which is the top most CO3L expression.

| **Frame Name** | EntityDeclaration | Abstract |
|---|---|---|
| **Kind of** | OntologicalProposition, AtomicProposition | |
| **Description** | Declares the existence of ontological entities such as classes, properties, methods, types | |

An entity declaration is also abstract. Only its sub-frames (e.g., ClassDeclaration, PropertyDeclaration, MethodDeclaration) can have concrete instances.

| **Frame Name** | OntologyRelationship | Abstract |
|---|---|---|
| **Kind of** | OntologicalProposition, AtomicProposition | |
| **Description** | Declares the existence of a relationship between two or more ontological entities (e.g., the relation between a property and its type) | |

Simple ontological relationships are represented through atomic propositions. That's why the frame OntologyRelationship is also a kind of AtomicProposition.

The full language description involves several other frames. However, space constraints preclude their presentation. The complete specification can be found in [7].

The presented abstract grammar enables the definition of diverse concrete syntaxes as appropriate for the application at hand. Our agents have been using S-Expression syntax, but others can also be defined, in particular XML syntax.

[1] An axiom is certainly a relationship between entities of the ontology. This distinction highlights the difference between the simple relations usually captured in ontology representation languages, and the complex relationships that can be expressed by axioms such as the one in this section.

## 4. RELATED WORK

In the same vein as RDF / RDF Schema [12] and DAML+OIL [4], CO3L is also an abstract language for which several concrete syntaxes may be defined including S-Expression and XML syntaxes.

As in DAML+OIL [4] and OWL [11], properties have autonomous existence outside the scope of classes. This allows defining hierarchic and other relations between properties. The same holds for methods, which may also exist outside the scope of classes.

Besides a predefined set of facets, CO3L allows the definition of new facets if desired. As far as we are aware of, only Ontolingua [5] and OKBC [3] have this feature.

Unlike other well-known ontology language, CO3L allows the declaration of methods, their arguments and return values (in case of functional methods). This feature has been proposed has a desired extension to DAML+OIL in [9] but was never officially integrated as part of DAML+OIL. Ontolingua allows the definition of functions and relations but can't associate them to classes.

As with Ontolingua [5] and OKBC [3], CO3L allows the definition of arbitrary axioms. However, unlike them, CO3L allows the definition of action methods (i.e., methods whose execution changes the state of the world) through state constraint and state change axioms. This is possible only because CO3L has a set of operators used to capture world state and world state changes.

## 5. CONCLUSIONS AND FUTURE WORK

This paper proposes a compact version of the ontology representation language of O3F. Since O3F is a reasonably comprehensive ontology representation framework capable of representing ontologies originally expressed in a variety of other frameworks such as DAML+OIL, OWL, and Ontolingua, the proposal of an expressive and yet simple to use language for O3F can be a useful contribution to advance the state of the art.

Some of the properties of the framework exceed the capabilities of common ontology representation approaches. In particular, the O3F framework allows to represent all commonly used ontological entities such as classes, properties and facets; it also represents some concepts that are not so common such as arbitrary axioms; and it represents other concepts that are not represented in other approaches such as action method definition.

Moreover, since the compact language proposed in this paper is much more usable and readable than the original proposal, it is at least likely that our work will contribute to facilitate the ontology specification process.

Finally, since we have provided an abstract grammar for the proposed language, it makes it possible to easily define diverse concrete syntaxes, which may be more adequate for the application at hand. For instance, the XML syntax of O3F will certainly be welcome in the web community.

The first future step is the development of a user-friendly editor for O3F to facilitate the definition and maintenance of ontologies using the O3F framework.

One would also profit from the existence of plug-in parsers and generators for other ontology representation languages such as DAML+OIL, OWL and Ontolingua. This way, it would be possible to import and export O3F ontologies from and to other formats.

Most ontology representation frameworks, such as description logic based approaches, have been proposed with the goal of enabling the development of decidable theorem provers. In general, it is necessary to trade off expressiveness for decidability. One of the future steps of our work will be to define subsets of the O3F representation language with different types of decidability.

## 6. ACKNOWLEDGMENTS

The research is supported by UNIDE/ISCTE.

## 7. REFERENCES

[1] Botelho, L.B.; Antunes, N.; Mohmed, E.; and Ramos, P. "Greeks and Trojans Together". In Proc. of the AAMAS 2002 Workshop "Ontologies in Agent Systems". 2002

[2] Botelho, L.M.; Mendes, H.; Figueiredo, P.; and Marinheiro, R. "Send Fredo off to do this, send Fredo off to do that". Submitted to the International Workshop on Cooperative Information Agents (CIA-2003). 2003

[3] Chaudhri, V.K.; Farquhar, A.; Fikes, R.; Karp, P.D.; and Rice, J.P. "Open Knowledge Base Connectivity 2.0.31". http://www.ai.sri.com/~okbc/spec.html. 1998

[4] DARPA Agent Markup Language. "Reference description of the DAML+OIL (March 2001) ontology markup language". 2001

[5] Farquhar, A.; Fikes, R.; and Rice, J. "Tools for Assembling Modular Ontologies in Ontolingua". In Proc. of the Fourteenth National Conference on Artificial Intelligence (AAAI'97). 1997

[6] Genesereth, M.R.; and Fikes, R., eds. Knowledge Interchange Format. *Draft proposed American National Standard* (dpANS). NCITS.T2/98-004 . 1998

[7] http://we-b-mind.org/o3f/language/abstract_spec_v1.htm

[8] http://we-b-mind.org/o3f/ontologies/travel_agency.htm

[9] Mota, L. "Extension to DAML+OIL: representation of methods". 3rd Agentcities.net Information Day. 2003

[10] Mota, L.; Botelho, L.M.; Mendes, H.; and Lopes, A. $O_3F$: an Object Oriented Ontology Framework. Proc. of the Second International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS2003), 2003, to appear

[11] W3C. OWL Web Ontology Language Overview. W3C Working Draft 31 March 2003. http://www.w3.org/TR/owl-features/. 2003

[12] W3C. Resource Description Framework (RDF). http://www.w3.org/RDF/