Title	WSML - a Language Framework for Semantic Web Services
Author(s)	Polleres, Axel; Fensel, Dieter
Publication Date	2005
Publication Information	Holger Lausen, Jos de Bruijn, Axel Polleres, Dieter Fensel "WSML - a Language Framework for Semantic Web Services", Proceedings of the W3C Workshop on Rule Languages for Interoperability, 2005.
Item record	http://hdl.handle.net/10379/459

WSML - A LANGUAGE FRAMEWORK FOR SEMANTIC WEB SERVICES

Position Paper for the W3C rules workshop, Washington DC, USA, April 2005

Authors:
Holger Lausen
Jos de Bruijn
Axel Polleres
Dieter Fensel

Digital Enterprise Research Institute (DERI)
National University of Ireland, Galway, Ireland
University of Innsbruck, Austria
Email: {holger.lausen, jos.debruijn, axel.polleres, dieter.fensel} 'at' deri.org

ABSTRACT

The Web Service Modeling Language (WSML) provides a framework of different language variants to describe semantic Web services. This paper presents the design rationale and relation with existing language recommendations. WSML is a frame based language with an intuitive human readable syntax and XML and RDF exchange syntaxes, as well as a mapping to OWL. It provides different variants, allowing for open and closed world modeling; it is a fully-fledged ontology and rule language with defined variants grounded in well known formalisms, namely Datalog, Description Logic and Frame Logic. Taking the key aspects of WSML as a starting point, we rationalize the design decisions which we consider relevant in designing a proper layering of ontology and rule languages for the Semantic Web and semantic Web services.

INTRODUCTION

With the Web Service Modeling Language WSML [de Bruijn et al., 2005] we provide a language framework for semantic Web services, based on the conceptual model of WSMO [Roman et al., 2005]. WSML provides means to describe semantic Web services [Fensel & Bussler, 2002] and its related aspects, i.e. ontologies, web Services, goals, and mediators. Those descriptions aim at automating Web service related tasks such as discovery, mediation and invocation. Reasoning based on formal descriptions can be used to achieve automation. In this paper we will mainly focus on the modelling of ontologies and logical expressions.

We believe that different application scenarios require different expressiveness of the underlying logical formalism. For example in the application area of Web service discovery [Keller et al., 2004] it has been shown that depending on the concrete scenario different levels of expressiveness are required. However interoperability and reuse of common domain ontologies is required across those different layers. WSML provides such a framework including a proper layering that enables the use of shared terminologies.

We believe that current Semantic Web language recommendations are not sufficient for the domain of semantic Web service modeling. Approaches such as OWL-S take different language recommendations (OWL, SWRL) and combine them without proper conceptual layering and lack of formal semantics. We take a different approach: with WSML we construct an overall framework for the specification of ontologies, Web services, goals, and mediators, with strict layering and a mapping to existing standards.

WEB SERVICE MODELING LANGUAGE

WSML makes a clear distinction between the modeling of the different conceptual elements (ontologies, Web services, goals, and mediators). Furthermore WSML separates between conceptual syntax and logical expression syntax, where the latter is used for describing additional constraints and axioms. The separation allows for an easy adoption by non-experts, since the conceptual syntax allows an intuitive understanding for many readers, whereas complex logical expressions require more familiarity and training with the language. In this context, we emphasize the normative "human readable" syntax, which allows better legibility. For the purpose of exchange on the Web XML and RDF serializations are specified.

Illustrating Example The example in Listing 1 illustrates some of the design principles of WSML. Identifiers are IRIs (the successor standard of URIs), delimited using `_"` and `"` in WSML. Namespaces are used to abbreviate IRIs. In WSML every non-prefixed identifier is preceded by the default namespace (in our example http://www.example.org/Family#) and prefixed identifier such as `dc#source` are expanded to full IRIs by replacing the prefix with the corresponding namespace.

The conceptual model allows for each element the specification of non-functional properties for additional related information. We recommend the use of the Dublin Core metadata set [Weibel et al., 1998]. In the example one concept and one axiom is declared, the concept describes several aspects of a human being (lines 7 through 10). Attributes can have features such as being the inverse of another attribute, but in contrast to OWL WSML defines such features local to the concept. In the example the age of a human has to be given as an integer value (`_integer` corresponds to the integer datatype of XML Schema). Attributes can have cardinality restrictions. For instance `hasAgeInYears` is specified as optional attribute with the maximum cardinality of one (i.e. it is functional). The example uses closed world constraints on the type of attributes (keyword `ofType`), such that every attribute value is required to be provably of the type specified. Note that this is different to OWL where the type of an attribute value is inferred based on the given range, which corresponds to the use of the `impliesType` keyword in WSML.

```
 1 wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"
 2
 3 namespace {_"http://www.example.org/Family#",
 4   dc _"http://purl.org/dc/elements/1.1#"}
 5
 6 ontology Family
 7   concept Human
 8     hasParent inverseOf(hasChild) ofType Human
 9     hasChild ofType Human
10     hasAgeInYears ofType (0 1) _integer
11
12   axiom DefinitionTeenager
13     nonFunctionalProperties
14       dc#source hasValue _"http://dictionary.reference.com/search?q=teenager"
15     endNonFunctionalProperties
16     definedBy
17       forall {?teen,?age} (
18         ?teen memberOf Teenager impliedBy
19           ?teen[hasAgeInYears hasValue ?age] memberOf Human and
20           ?age >= 13 and ?age =< 19).
```

Listing 1: Example for a Concept and Axiom Description in WSML

Axioms allow the refinement of an ontology with the use of logical expressions. The axiom in the example defines a Teenager as a human of an age between 13 and 19. The logical expression syntax is based on F-Logic [Kifer et al., 1995], but using different keywords, such as `memberOf` instead of `:` to express the concept-membership relation. Logical expressions are interweaved with the conceptual syntax using the `axiom` keyword and an optional identifier. In the example, we use additional non-functional properties to specify the source of the information contained in the axiom.

WSML Layering The WSML framework allows a number of variants with different expressiveness. All Variants and their interrelation are depicted in Figure 1. Whereby extension means that every valid specification of an extended variant is also a valid specification of the new variant. Furthermore, all consequences inferred from a "lower" WSML variant are also valid consequences of the extended variant. The layering is defined by syntactical restrictions on the language.

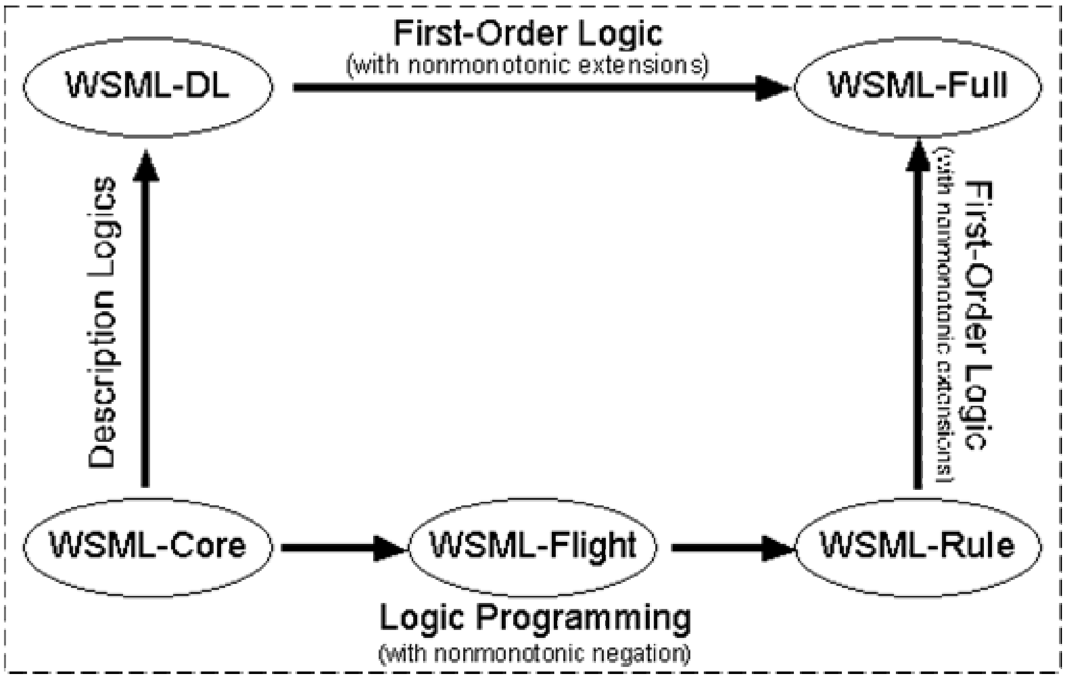

Figure 1: WSML Variants

WSML-Core corresponds with the intersection of Description Logic and Horn Logic [Grosof et al., 2003] (without function symbols and without equality), extended with datatype support in order to be useful in practical applications. WSML-Core is fully compliant with a subset of OWL.

WSML-DL extends WSML-Core to an expressive Description Logic, namely, *SHIQ*, thereby covering that part of OWL which is efficiently implementable.

WSML-Flight extends WSML-Core in the direction of Logic Programming. WSML-Flight has a rich set of modeling primitives for different aspects of attributes, such as value and integrity constraints. Furthermore, WSML-Flight incorporates a rule language, while still allowing efficient decidable reasoning. More precisely, WSML-Flight allows any Datalog rule, extended with inequality and (locally) stratified negation.

WSML-Rule extends WSML-Flight to a fully-fledged Logic Programming language, by allowing function symbols and unsafe rules.

WSML-Full unifies all WSML variants under a common First-Order umbrella with non-monotonic extensions which allow to capture non-monotonic negation of WSML-Rule.

WSML and the Web WSML makes extensive use of existing Web standards. For example, it uses IRIs to identify entities, as illustrated in Listing 1.

The Extensible Markup Language (XML) is a markup language for interchange of structured data. WSML adopts the use of namespaces from XML and the datatypes from XML Schema. Furthermore, WSML specifies an XML exchange syntax for interoperability on the Web.

The Resource Description Framework (RDF) defines a triple data structure for meta data description on the Web and RDF Schema describes how to use RDF to describe vocabularies. WSML reuses and extends the RDF Schema vocabulary to allow existing RDF(S)-based tools to understand parts of a WSML specification. WSML puts viewer restrictions on the kind of allowed triples then OWL DL does.

For logical expressions we reuse the WSML/XML serialization, since an encoding of complex constructs into triples results in many triples and complicated processing.

The Web Ontology Language (OWL) is not reused directly as basis for our language, given the goal of a strict semantic layering and computational beneficial properties. Instead, we provide mappings to those fragments of OWL that have desirable computational properties. Specifically, there exist bi-directional mappings for WSML-Core and WSML-DL to and from OWL.

SUMMARY OF POSITION

Given the design decisions taken and the relation to existing specifications, we summarize our rationale as follows:

One syntactic framework for a set of layered languages

We believe different semantic Web applications need languages of different expressiveness. Instead of retrofitting the layering to existing language proposals and recommendations we propose a coherent framework with strict layering and desirable computational properties. This is different than other approaches like SWRL, which layer on top of OWL DL. SWRL has syntactical restrictions such as allowing only binary relations, that are not motivated by computational properties.

Normative, human readable syntax

It has been argued that tools will hide language syntax from the user; however, as has been shown with SQL, an expressive but understandable syntax is one of the criteria for successful adoption of a language. Developers and users will have to deal with the concrete syntax and a human readable version (compared to XML or RDF) allows a faster uptake of the language.

Separation of conceptual model and logical expression

The conceptual syntax allows the modeling of ontologies on the level of concepts, relations and instances, while logical expression allow further refinements. This separation provides a clear way to include annotations and enhance the structure of a knowledge base, in the sense that the structure of an ontology can be easier recognized within the conceptual model then the encoding in logical expressions allows.

Semantics based on well known formalism

Our proposal unifies well known formalisms such as Datalog and Description Logics with a syntactical framework compliant with current Web standards, while maintaining the original well researched computational properties. Furthermore it allows the reuse of tools already developed for those formalisms.

Closed world and open world assumption

We believe it is beneficial to allow developers to choose between both assumptions for application within the semantic Web. Allowing the closed world assumption will allow users coming from the database area to adopt the technology more easily.

REFERENCES

[de Bruijn et al., 2005] J. de Bruijn, H. Lausen, R. Krummenacher, A. Polleres, L. Predoiu, M. Kifer, D. Fensel. *The Web Service Modeling Language WSML*. WSML Deliverable D16.1v0.2, 2005. Available from http://www.wsmo.org/TR/d16/d16.1/v0.2/

[Fensel & Bussler, 2002] D. Fensel and C. Bussler: *The Web Service Modeling Framework WSMF*, Electronic Commerce Research and Applications, 1(2), 2002.

[Grosof et al., 2003] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker. Description logic programs: Combining logic programs with description logic. In *Proc. of the Twelfth International World Wide Web Conference (WWW 2003)*, pages 48-57. ACM, 2003.

[Keller et al., 2004] U. Keller, R. Lara, A. Polleres (eds.): *WSMO Web Service Discovery*, WSMO deliverable D5.1 version 0.1. Available from http://www.wsmo.org/2004/d5/d5.1/v0.1/.

[Kifer et al., 1995] M. Kifer, G. Lausen, and J. Wu: Logical foundations of object-oriented and frame-based languages. *Journal of the ACM*, 42:741-843, July 1995.

[Roman et al., 2004] D. Roman, H. Lausen, and U. Keller (eds.): *Web Service Modeling Ontology - Standard (WSMO - Standard)*, WSMO deliverable D2 version 1.1. Available from http://www.wsmo.org/TR/d2/v1.1/.

[Weibel et al., 1998] S. Weibel, J. Kunze, C. Lagoze, and M. Wolf: *RFC 2413 - Dublin Core Metadata for Resource Discovery*, September 1998.

ACKNOWLEDGMENTS

The work is funded by the European Commission under the projects DIP, Knowledge Web, InfraWebs, SEKT and ASG; by Science Foundation Ireland under the DERI-Lion project; by the FIT-IT (Forschung, Innovation, Technologie - Informationstechnologie) under the projects RW^2 and TSC. The editors would like to thank to all the members of the WSML working (cf. http://www.wsmo.org/wsml/) group for their advice and input to this document

$Date: 2005/04/10 02:43:56 $