



Data, Information and Process Integration
with Semantic Web Services

DIP

Data, Information and Process Integration with Semantic Web Services

FP6 - 507483

Deliverable

D1.9 WSML Reasoner

Boris Motik
Gabor Nagypal
Stephan Grimm

20 June 2005



EXECUTIVE SUMMARY

Deliverable D1.9 is a prototype of a reasoning engine for the WSML language defined in D2.7. It allows for inferencing according to the semantics of WSML-Flight defined in deliverable D1.7. This preliminary version of the document serves as a fact sheet for the WSML Reasoner implementation and its underlying inference engine KAON2.

Document Information

IST Project Number	FP6 – 507483	Acronym	DIP
Full title	Data, Information, and Process Integration with Semantic Web Services		
Project URL	http://dip.semanticweb.org		
Document URL			
EU Project officer	Brian Macklin		

Deliverable	Number	1.2	Title	Prototype of the ontology based query parser and engine
Work package	Number	1	Title	Ontology reasoning and querying








Date of delivery	Contractual	M30	Actual	01-Jul-05
Status	version 0.1		final	<input type="checkbox"/>
Nature	Prototype	<input checked="" type="checkbox"/>	Report	<input type="checkbox"/>
	Dissemination	<input type="checkbox"/>		
Dissemination Level	Public	<input type="checkbox"/>	Consortium	<input checked="" type="checkbox"/>

Authors (Partner)	Stephan Grimm (FZI)			
Responsible Author	Stephan Grimm		Email	stephan.grimm@fzi.de
	Partner	FZI	Phone	+49 721 9654816



Abstract (for dissemination)	This deliverable is just a fact sheet for the WSML Reasoner and its underlying inference engine KAON2.
Keywords	WSML reasoning, basic inference services, query answering

Version Log			
Issue Date	Rev No.	Author	Change
20-06-05	001	S. Grimm	Deliverable written

Project Consortium Information

Partner	Acronym	Contact
National University of Ireland Galway	NUIG  <i>National University of Ireland, Galway</i> <i>Ollscoil na hÉireann, Gaillimh</i>	Prof. Dr. Christoph Bussler Digital Enterprise Research Institute (DERI) National University of Ireland, Galway Galway Ireland Email: chris.bussler@deri.org Tel: +353 91 512460
Fundacion De La Innovacion.Bankinter	Bankinter 	Monica Martinez Montes Fundacion de la Innovation. BankInter Paseo Castellana, 29 28046 Madrid, Spain Email: mmtnez@bankinter.es Tel: 916234238
Berlecon Research GmbH	Berlecon 	Dr. Thorsten Wichmann Berlecon Research GmbH Oranienburger Str. 32 10117 Berlin, Germany Email: tw@berlecon.de Tel: +49 30 2852960
British Telecommunications Plc.	BT 	Dr John Davies BT Exact (Orion Floor 5 pp12) Adastral Park Martlesham Ipswich IP5 3RE, United Kingdom Email: john.nj.davies@bt.com Tel: +44 1473 609583
Swiss Federal Institute of Technology, Lausanne	EPFL 	Prof. Karl Aberer Distributed Information Systems Laboratory École Polytechnique Fédérale de Lausanne Bât. PSE-A 1015 Lausanne, Switzerland Email : Karl.Aberer@epfl.ch Tel: +41 21 693 4679
Essex County Council	Essex 	Mary Rowllatt, Essex County Council PO Box 11, County Hall, Duke Street Chelmsford, Essex, CM1 1LX United Kingdom. Email: maryr@essexcc.gov.uk Tel: +44 (0)1245 436524
Forschungszentrum Informatik	FZI 	Andreas Abecker Forschungszentrum Informatik Haid-und-Neu Strasse 10-14 76131 Karlsruhe Germany Email: abecker@fzi.de Tel: +49 721 9654 0

<p>Institut für Informatik, Leopold-Franzens Universität Innsbruck</p>	<p>UIBK </p>	<p>Prof. Dieter Fensel Institute of computer science University of Innsbruck Technikerstr. 25 A-6020 Innsbruck, Austria Email: dieter.fensel@deri.org Tel: +43 512 5076485</p>
<p>ILOG SA</p>	<p>ILOG  Changing the rules of business</p>	<p>Christian de Sainte Marie 9 Rue de Verdun, 94253 Gentilly, France Email: cma@ilog.fr Tel: +33 1 49082981</p>
<p>inubit AG</p>	<p>Inubit  the integration experts</p>	<p>Torsten Schmale inubit AG Lützowstraße 105-106 D-10785 Berlin Germany Email: ts@inubit.com Tel: +49 30726112 0</p>
<p>Intelligent Software Components, S.A.</p>	<p>iSOCO </p>	<p>Dr. V. Richard Benjamins, Director R&D Intelligent Software Components, S.A. Pedro de Valdivia 10 28006 Madrid, Spain Email: rbenjamins@isoco.com Tel. +34 913 349 797</p>
<p>The Open University</p>	<p>OU </p>	<p>Dr. John Domingue Knowledge Media Institute The Open University, Walton Hall Milton Keynes, MK7 6AA United Kingdom Email: j.b.domingue@open.ac.uk Tel.: +44 1908 655014</p>
<p>SAP AG</p>	<p>SAP </p>	<p>Dr. Elmar Dörner SAP Research, CEC Karlsruhe SAP AG Vincenz-Priessnitz-Str. 1 76131 Karlsruhe, Germany Email: elmar.dorner@sap.com Tel: +49 721 6902 31</p>
<p>Sirma AI Ltd.</p>	<p>Sirma  Ontotext Knowledge and Language Engineering Lab of Sirma</p>	<p>Atanas Kiryakov, Ontotext Lab, - Sirma AI EAD Office Express IT Centre, 3rd Floor 135 Tzarigradsko Chausse Sofia 1784, Bulgaria Email: atanas.kiryakov@sirma.bg Tel.: +359 2 9768 303</p>
<p>Tiscali Österreich GmbH</p>	<p>Tiscali </p>	<p>Dieter Haacker Tiscali Österreich GmbH. Diefenbachgasse 35 A-1150 Vienna Austria Email: Dieter.Haacker@at.tiscali.com</p>

		Tel: +43 1 899 33 160
Unicorn Solution Ltd.	<p>Unicorn</p> 	Jeff Eisenberg Unicorn Solutions Ltd, Malcha Technology Park 1 Jerusalem 96951 Israel Email: Jeff.Eisenberg@unicorn.com Tel.: +972 2 6491111
Vrije Universiteit Brussel	<p>VUB</p> 	Carlo Wouters Starlab- VUB Vrije Universiteit Brussel Pleinlaan 2, G-10 1050 Brussel ,Belgium Email: carlo.wouters@vub.ac.be Tel.: +32 (0) 2 629 3719

1 INTRODUCTION

Deliverable D1.9 is a prototype of a reasoner for the WSML language. The underlying inference engine of the WSML reasoner is KAON2, a first version of which was delivered with D1.2. WSML comes in different variants based on the two different knowledge representation paradigms of concept description languages and logic programming. KAON2 is a hybrid reasoning systems that combines the two paradigms by allowing datalog-style rules to interact with structural description logics knowledge bases. It supports the *SHIQ(D)* description logic and disjunctive datalog programs.

The WSML reasoner is a wrapper around KAON2 that supports reasoning in WSML-Flight. It has a connection to the WSMO4J API in that it processes WSMO4J ontology objects for reasoning and querying with inference support. The conceptual elements of a WSML ontology are mapped to a KAON2 internal representation and reasoning is performed through the KAON2 querying interface.

2 SYSTEM REQUIREMENTS

The WSML reasoner prototype has been written in Java 1.5. To work with the system, please download the latest Java release from <http://java.sun.com/>. The system should work on any computer under any operating system capable of running Java 1.5.

3 INSTALLATION

The first version of the preliminary prototype is delivered packaged in a file, whose name is `WSML_Reasoner.zip`. To install the system, simply unzip the package into any directory. There are no particular requirements on which directory you should choose, and no further settings on the computer are necessary. The core software library comes as a jar-File containing the necessary Java classes.

To uninstall the system, simply remove the directory where you unzipped the package.

After you install the system, your installation directory should contain the following directories and files:

- `WSML4Kaon2-0.1.jar` – This is the executable file of the prototype. It contains all necessary class files.
- `apidoc` – This directory contains the Javadoc documentation of the library's API.
- `lib` – This directory contains external libraries which the WSML reasoner is dependent on. This includes a version of WSMO4J.

4 IMPLEMENTED FUNCTIONALITY

The WSML Reasoner is a software library that provides an API for reasoning with WSML ontologies through WSMO4J. This section describes the functional principle of the WSML Reasoner as a wrapper around KAON2 and its interface.

4.1 Functional Principle

Internally, the WSML Reasoner works such that WSML elements are mapped to KAON2 instances reified from concepts and instances in WSML. Appropriate KAON2 rules assure that these elements are interpreted properly according to the WSML semantics. In the current status of implementation the WSML Reasoner only covers the conceptual part of the WSML syntax whereas logical expressions in axioms are ignored because their handling is not supported by the WSMO4J version used. The part of the semantics for the conceptual WSML syntax elements, i.e. concepts, attributes and instances, is realised by the following internal meta-level rules that assure transitivity and strict inheritance of subsumption:

$$\begin{aligned} \text{subConceptOf}(c_1, c_3) &\leftarrow \text{subConceptOf}(c_1, c_2) \wedge \text{subConceptOf}(c_2, c_3) \\ \text{memberOf}(i, c_2) &\leftarrow \text{subConceptOf}(c_1, c_2) \wedge \text{memberOf}(i, c_1) \end{aligned}$$

The reason for representing the WSML elements as reified KAON2 instances is to be able to handle the Metamodelling characteristics of WSML-Flight, according to which vocabularies for concepts and instances are not necessarily separate.

4.2 Reasoner API

The WSML Reasoner offers the basic reasoning services of *concept subsumption* and *instance checking* for WSML-Flight ontologies. It operates on WSMO4J elements, this is, it receives a WSMO4J ontology object as input at initialization time while concepts and instances are identified by their WSML String identifiers being URIs. The Reasoner also provides some functionality related to instance retrieval and schema querying. However, in the current status of implementation the capabilities of the KAON2 query answering interface are not fully exploited, since some more conceptual work has to be done in integrating KAON2 into the WSMO4J and ORDİ API. This is covered by D1.8 Ontology Querying. A full support of KAON2 disjunctive queries requires the programmatic handling of logical expressions in the WSMO4J API, which is not included in its current version.

Figure 1 depicts the internal architecture of the WSML Reasoner in an UML class diagram. The class `WSMLReasoner` outlines the interface of reasoning services provided. It is initialized with a WSMO4J ontology object seen as the knowledge base with respect to which reasoning is performed – any change to this ontology after the initialization of the reasoner does not affect the reasoning process.

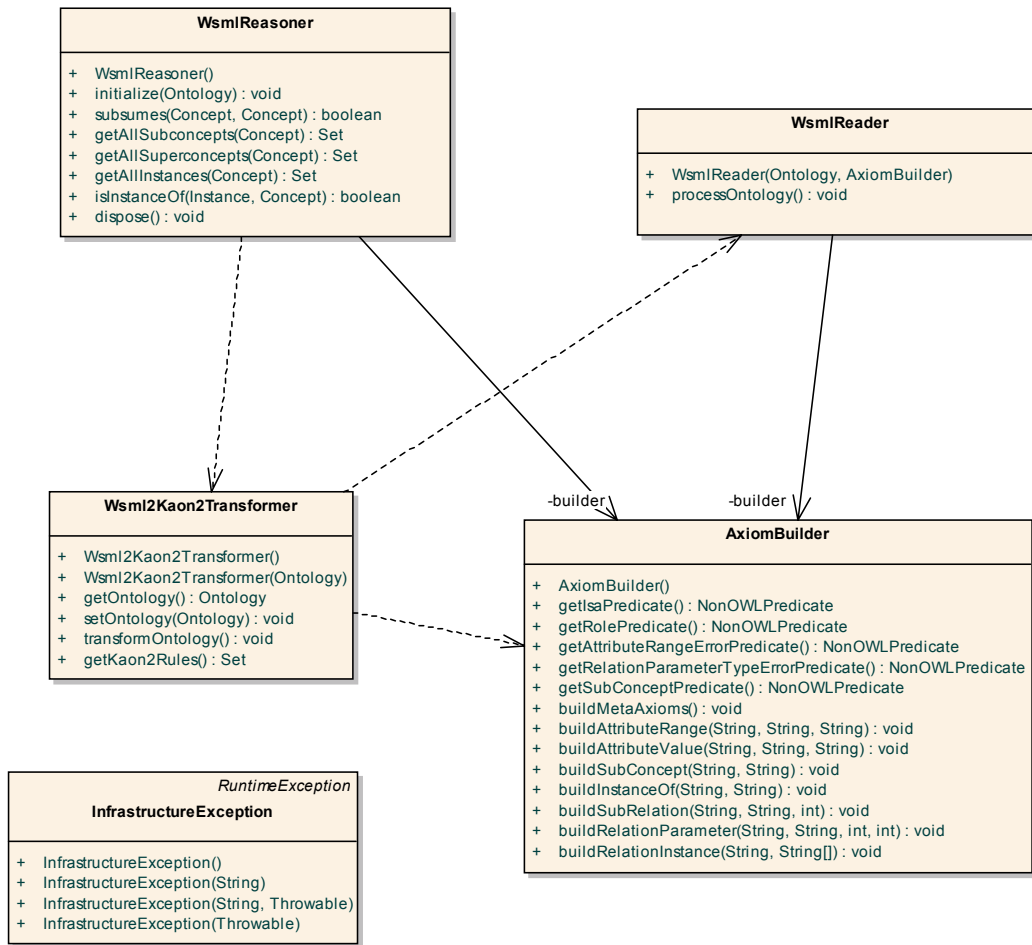


Figure 1 UML class diagram outlining the WSML Reasoner interface

The method

```
boolean subsumes(Concept c1, Concept c2)
```

checks whether the concept C_1 is a super concept of the concept C_2 , i.e. whether

$$Ontology \models \forall x C_2(x) \rightarrow C_1(x) \quad .$$

The method

```
boolean isInstanceOf(Instance i, Concept c)
```

checks whether I is an instance of the concept C , i.e. whether

$$Ontology \models C(I) \quad .$$

The method

```
Set<Instance> getAllInstances(Concept c)
```

retrieves the extension of the concept C , i.e. the following set of known individuals.

$$\{I: Ontology \models C(I) \quad \} \quad .$$

The methods

```
Set<Concept> getAllSubconcepts (Concept c),  
Set<Concept> getAllSuperconcepts (Concept c)
```

provide a similar retrieval service for schema querying.

5 OUTLOOK

In the future, D1.9 will be extended in the following ways:

- We shall integrate the upcoming releases of the WSMO4J API to support the full range of WSML-Flight language constructs, in particular logical expressions.
- We shall support the full WSML-Flight semantics for the above mentioned language constructs.
- We shall extend KAON2 with concrete domain (datatype) support and make it available for WSML reasoning.
- We shall provide a query answering interface through the WSML Reasoner, connected to WSMO4J, to make the KAON2 disjunctive query answering machinery available to the user.