



DIP D2.4: A Versioning Tool for Ontologies

Prototype Fact Sheet 30 June 2006

This version:

<http://www.omwg.org/tools/versioning/v1.0/FactSheet.html>

Latest version:

<http://www.omwg.org/tools/versioning/v1.0/FactSheet.html>

Previous version:

<http://www.omwg.org/tools/versioning/v0.2/FactSheet.html>

Authors:

Pieter De Leenheer, VUB, pieter.de.leenheer@vub.ac.be
 Jacek Kopecký, UIBK, jacek.kopecky@deri.org
 Erel Sharf, Unicorn, erel@il.ibm.com
 Aldo de Moor, VUB, ademoor@vub.ac.be

Reviewers:

Carlos Pedrinaci, OU
 Michael Stollberg, UIBK

This document is also available in a non-normative [Adobe PDF](#) version.

Copyright © 2006 by DIP. All Rights Reserved. DIP liability, trademark, document use, and software licensing rules apply.

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	http://www.omwg.org/tools/versioning/v1.0/FactSheet.html		
EU Project Officer	Kai Tullius		

Deliverable Number	2.4	Title	A Versioning Tool for Ontologies
Workpackage Number	2	Title	Ontology Management

Date of Delivery	contractual	M30	actual	30-June-2006
Status	version	1.0	final	
Nature	Prototype <input checked="" type="radio"/> Report <input type="radio"/> Dissemination <input type="radio"/> Ontology <input type="radio"/>			
Dissemination Level	Public <input checked="" type="radio"/> Consortium <input type="radio"/>			

Authors	Pieter De Leenheer (VUB STARLab), Jacek Kopecky (UIBK DERI), Erel Sharf (Unicorn), Aldo de Moor (VUB)		
Responsible Author	Pieter De Leenheer	Email	pieter.de.leenheer@vub.ac.be
Partner	STARLab, VUB	Phone	(+32) 629 35 43

Abstract (for dissemination)	<p>For describing Semantic Web services, so-called domain ontologies are elicited which formalize the knowledge necessary for capturing the meaning of the services and the exchanged data. In this setting, evolution is a persistent problem throughout the ontology life-cycle. In order to come to a shared, reusable and agreed meaning, the elicitation of these ontologies is conducted massively in a distributed fashion, and is governed by a combination of various ontology engineering processes. This results in a sequence of transformations and versions that expand, contract and revise the ontology; even after deployment. Furthermore, different engineers might want to contextualize the definition for their particular purpose, leading to a diverging range of subjective views (versions). Next to changing ontology versions, we consider the compatibility between different versions as important with a view to losing as little information as possible. Especially in the Semantic Web setting, it might be that not all resource providers commit to the most recent ontology version. Our work has two aspects: first, we adapt the existing ontology versioning research for the particular case of the Web Services Modeling Language (WSML) ontologies; second, we support at least partial automatic generation of transformations between ontology versions. Such transformations are necessary for interoperability between deployments of the different versions, a scenario that Semantic Web practically guarantees.</p>
Keywords	Ontology evolution, ontology versioning

Version Log			
issue date (dd-mm-yy)	revision no.	author	change
07-11-05	001	Pieter De Leenheer	V0.3 draft based on V0.2
28-12-05	002	Pieter De Leenheer	V0.3 final
09-06-06	003	Pieter De Leenheer	V1.0 final

Reviewer Information			
1	Carlos Pedrinaci	Email	C.Pedrinaci@open.ac.uk
	Partner OU	Phone	+44 (0) 1908 654773
2	Michael Stollberg	Email	michael.stollberg@deri.org
	Partner UIBK	Phone	+43 - 512 - 507 6479

Table of contents

1 Availability and Contacts

2 Purpose and Functionality

2.1 The Current Version

3 Requirements

4 Licensing

4.1 Versioning Library License Agreement

4.2 Licensing of Third Party Libraries

5 Installation and Usage

5.1 Installation of the versioning library

5.2 Usage examples

6 Future Plans

Appendix A. References

1 Availability and Contacts

Version: 1.0, 30 June 2006.

Download: <http://www.omwg.org/tools/versioning/v1.0/versioning.zip>

Accompanying theory document: Located in <http://www.omwg.org/tools/dip/factsheets/D2.4-theorydoc.pdf>

Source control: In CVS of the DOME SourceForge project at <http://dome.cvs.sourceforge.net/dome/versioning/>.

Contact person: Jacek Kopecký, jacek.kopecky@deri.org

2 Purpose and Functionality

The purpose of the versioning library is to enable user-driven creation of versions of WSMO ontologies. This library implements what is described in the accompanying theory document, located in <http://www.omwg.org/tools/dip/factsheets/D2.4-theorydoc.pdf>. It consists of a versioning library described in this document and the GUI tool integrated in deliverable D2.8.

The library extends WSMO API and WSMO4J with versioning functionality. It is split into two Java packages:

- `org.omwg.versioning` — extension of WSMO API with the methods necessary for versioning functionality
- `org.omwg.versioningimpl` — implementation of the extended versioned WSMO API that uses WSMO4J by default

The downloadable library package contains the following top-level files and directories:

- `FactSheet.html` — this file
- `dip.css` — CSS stylesheet for the fact sheet
- `images` — images for the fact sheet
- `versioning.jar` — a compiled Java library that can be put in the classpath of an application that needs to use it
- `src` — the directory containing the sources of this library
- `doc` — the directory containing the javadoc API documentation

A user guide can be found at <http://www.omwg.org/tools/dip/oms/>.

2.1 The Current Version

The current version has the following major capabilities:

Version identification and metadata — the API contains interfaces and classes for versioned WSMO API Identifiers, and for version metadata containing version comment, date of creation etc.

Ontology versioning API — allows the user of this API to start a new version of an ontology, to go back to the previous version, and to commit (finalize) a version. Further, the user of this API has full control over the version identifier of a committed version.

Version changelog functionality — during the creation of a new version, the significant changes are logged and when a version is committed, this change log is available to the application.

Partial version mapping — from the change log a partial mapping is generated for mediation between the old and the new version. This partial mapping is an input to a human designer who can complete it as appropriate.

Alternate evolution strategies — identified in the theoretical part of this deliverable, alternate evolution strategies are implemented as a user wizard in the GUI tool in deliverable D2.8. This wizard guides the user through resolving the impact of changes like concept removal, whose instances and subconcepts can be handled in different ways depending on the intent of the change.

3 Requirements

Nature: Java library

Interfaces (API, Web Services): a Java API.

Platform: JDK 1.4.

Supported standards: WSMO(?)

Required Libraries (OMWG, SDK Cluster, WSMO-related):

- The OMWG ontology mapping API from DOME is used to capture the partial version mapping generation.
- WSMO4J and WSMO API is an API and a reference implementation for building Semantic Web Services applications compliant with the Web Service Modeling Ontology (WSMO).

Required Libraries (others):

- none

4 Licensing

4.1 Versioning Library License Agreement

Copyright © 2005, DERI International.

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version. This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details. You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA.

4.2 Licensing of Third Party Libraries

Licensing of third party libraries and components required for the versioning library:

- wsmo4j - Copyright © Ontotext Lab, Sirma. It is an open-source library, available under LGPL.

5 Installation and Usage

5.1 Installation of the versioning library

The library file versioning.jar in the installation package needs to be in the classpath of the application that wishes to use it, together with WSMO4J and the mapping libraries.

5.2 Usage examples

The following is an excerpt from a simple demo program that uses the versioning library:

```
String FILE =
    "namespace { \"http://example.com/Workplace#\" }           \n" +
    "ontology _ \"http://example.com/Workplace\"              \n" +
    "  concept Person                                         \n" +
    "    name ofType _string                                  \n" +
    "    boss ofType Manager                                  \n" +
    "  concept Manager subConceptOf Person                    ";

// parse the file above
Map map = new HashMap();
map.put(Factory.LE_FACTORY, Factory.createLogicalExpressionFactory(new HashMap()));
map.put(Factory.PROVIDER_CLASS, VersionedParserImpl.class.getName());
Entity[] parsed = Factory.createParser(map).parse(new StringReader(FILE));
map.clear();
map.put(Factory.PROVIDER_CLASS, WSMOFactoryImpl.class.getName());
WsmoFactory factory = Factory.createWsmoFactory(map);
DataFactory df = Factory.createDataFactory(null);
assert parsed.length == 1;
assert parsed[0] instanceof Ontology;
assert Versioning.isVersioned((Ontology)parsed[0]);
assert !Versioning.isCommitted((Ontology)parsed[0]);

// create serializer
map.put(Factory.PROVIDER_CLASS, WSMLSerializerImpl.class.getName());
serializer = Factory.createSerializer(map);

// we know it is a versioned Ontology
Ontology ontology = (Ontology) parsed[0];
Versioning.commit(
    ontology,
    VersionIdentifier.NO_VERSION,
    new VersioningIRIPattern("http://example.com/Workplace{-v$ver}"),
    new HashMap());
printOntology("initial workplace ontology", ontology);

// get the first concept, happens to be Person
Concept c = (Concept) ontology.listConcepts().iterator().next();

// make some changes
c.createAttribute(factory.createIRI(
    "http://example.com/Workplace#birthday"));
printOntology("introduced a new attribute - birthday", ontology);

// drop changes
Versioning.dropChanges(ontology);
printOntology("changes dropped", ontology);

// make some changes and commit them
Attribute att = c.createAttribute(factory.createIRI(
    "http://example.com/Workplace#birthday"));
att.addType(df.createWsmldataType("http://www.wsmo.org/wsmldata-syntax#date"));
Versioning.commit(ontology, new HashMap());
printOntology("birthday reintroduced and committed", ontology);

// make some more changes
att = c.createAttribute(factory.createIRI(
    "http://example.com/Workplace#salary"));
att.addType(df.createWsmldataType("http://www.wsmo.org/wsmldata-syntax#float"));
printOntology("adding new attribute - salary", ontology);
```

6 Future Plans

The major driving forces for the future development of the library:

- support for the evolution of the related standards and tools (wsmo4j, WSMO, WSML);
- improvements and fixes related to the usage of the library.

Appendix A. References

[WSML0.2] J. de Bruijn, H. Lausen, R. Krummenacher, A. Polleres, L. Predoiu, M. Kifer, D Fensel: *The Web Service Modeling Language WSML*. Deliverable d16.1v0.2, WSML, 2005. <http://www.wsmo.org/TR/d16/d16.1/v0.2/>

[WSMO1.2] D. Roman, H. Lausen, U. Keller (eds); J. de Bruijn, Ch. Bussler, J. Domingue, D. Fensel, M. Hepp, M. Kifer, B. Konig-Ries, J. Kopecky, R. Lara, E. Oren, A. Polleres, J. Scicluna, M. Stollberg: *Web Service Modeling Ontology (WSMO)*. Deliverable d2v1.2, WSMO, 2005. <http://www.wsmo.org/TR/d2/v1.2/>