D1.2.10v2 Ontology Repositories and Content Evaluation

Denny Vrandečić and York Sure
Institut AIFB, Universität Karlsruhe (TH)
Raul Palma and Francisco Santana
Universidad Politecnica de Madrid (UPM)

Abstract.
EU-IST Network of Excellence (NoE) IST-2004-507482 KWEB Deliverable D1.2.10v2 (WP1.2)
In this deliverable we present results obtained from the work described in the first version of this deliverable (D1.2.10) about ontology repositories. In particular, we provide an overview of the sustainability of the Ontology Metadata Vocabulary (OMV), the proposal of a hierarchy for the classification of ontology domains and evaluation results for the distributed ontology metadata repository (Oyster). Furthermore, we discuss the flow of knowledge between Oyster and the ontology metadata portal Onthology that rely on the content evaluation metrics described at the end of the deliverable. Therefore, the second part of the deliverable discusses how to properly define metrics for ontologies, and introduce the notions of *ontology normalization* for measuring, and of *stable metrics*. This will help to properly define better ontology metrics in subsequent research in this area. The normalization has been implemented in a prototype as part of the KAON2 OWL Tools (source code available for download at OntoWare.org). Parts of this work have been already accepted for publication at the European Semantic Web Conference (ESWC2007).
Keyword list: ontology metadata, P2P, repository, reuse, ontology evaluation, ontology assessment, ontology measurements, measuring, metrics

Document Identifier	KWEB/2007/D1.2.10/v2.0
Project	KWEB EU-IST-2004-507482
Version	v2.0
Date	June 30, 2007
State	final
Distribution	public

Knowledge Web Consortium

This document is part of a research project funded by the IST Programme of the Commission of the European Communities as project number IST-2004-507482.

University of Innsbruck (UIBK) - Coordinator
Institute of Computer Science
Technikerstrasse 13
A-6020 Innsbruck
Austria
Fax: +43(0)5125079872, Phone: +43(0)5125076485/88
Contact person: Dieter Fensel
E-mail address: dieter.fensel@uibk.ac.at

Ècole Polythechnique Fédérale de Lausanne (EPFL)
Computer Science Department
Swiss Federal Institute of Technology
IN (Ecublens), CH-1015 Lausanne
Switzerland
Fax: +41 21 6935225, Phone: +41 21 6932738
Contact person: Boi Faltings
E-mail address: boi.faltings@epfl.ch

France Telecom (FT)
4 Rue du Clos Courtel
35512 Cesson Sévigné
France. PO Box 91226
Fax: +33 2 99124098, Phone: +33 2 99124223
Contact person : Alain Leger
E-mail address: alain.leger@rd.francetelecom.com

Freie Universität Berlin (FU Berlin)
Takustrasse 9
14195 Berlin
Germany
Fax: +49 30 83875220, Phone: +49 30 83875223
Contact person: Robert Tolksdorf
E-mail address: tolk@inf.fu-berlin.de

Free University of Bozen-Bolzano (FUB)
Piazza Domenicani 3
39100 Bolzano
Italy
Fax: +39 0471 315649, Phone: +39 0471 315642
Contact person: Enrico Franconi
E-mail address: franconi@inf.unibz.it

Institut National de Recherche en Informatique et en Automatique (INRIA)
ZIRST - 655 avenue de l'Europe -
Montbonnot Saint Martin
38334 Saint-Ismier
France
Fax: +33 4 7661 5207, Phone: +33 4 7661 5366
Contact person: Jérôme Euzenat
E-mail address: Jerome.Euzenat@inrialpes.fr

Centre for Research and Technology Hellas / Informatics and Telematics Institute (ITI-CERTH)
1st km Thermi - Panorama road
57001 Thermi-Thessaloniki
Greece. Po Box 361
Fax: +30-2310-464164, Phone: +30-2310-464160
Contact person: Michael G. Strintzis
E-mail address: strintzi@iti.gr

Learning Lab Lower Saxony (L3S)
Expo Plaza 1
30539 Hannover
Germany
Fax: +49-511-7629779, Phone: +49-511-76219711
Contact person: Wolfgang Nejdl
E-mail address: nejdl@learninglab.de

National University of Ireland Galway (NUIG)
National University of Ireland
Science and Technology Building
University Road
Galway
Ireland
Fax: +353 91 526388, Phone: +353 87 6826940
Contact person: Christoph Bussler
E-mail address: chris.bussler@deri.ie

The Open University (OU)
Knowledge Media Institute
The Open University
Milton Keynes, MK7 6AA
United Kingdom
Fax: +44 1908 653169, Phone: +44 1908 653506
Contact person: Enrico Motta
E-mail address: e.motta@open.ac.uk

Universidad Politécnica de Madrid (UPM)
Campus de Montegancedo sn
28660 Boadilla del Monte
Spain
Fax: +34-913524819, Phone: +34-913367439
Contact person: Asunción Gómez Pérez
E-mail address: asun@fi.upm.es

University of Karlsruhe (UKARL)
Institut für Angewandte Informatik und Formale
Beschreibungsverfahren - AIFB
Universität Karlsruhe
D-76128 Karlsruhe
Germany
Fax: +49 721 6086580, Phone: +49 721 6083923
Contact person: Rudi Studer
E-mail address: studer@aifb.uni-karlsruhe.de

University of Liverpool (UniLiv)
Chadwick Building, Peach Street
L697ZF Liverpool
United Kingdom
Fax: +44(151)7943715, Phone: +44(151)7943667
Contact person: Michael Wooldridge
E-mail address: M.J.Wooldridge@csc.liv.ac.uk

University of Manchester (UoM)
Room 2.32. Kilburn Building, Department of Computer
Science, University of Manchester, Oxford Road
Manchester, M13 9PL
United Kingdom
Fax: +44 161 2756204, Phone: +44 161 2756248
Contact person: Carole Goble
E-mail address: carole@cs.man.ac.uk

University of Sheffield (USFD)
Regent Court, 211 Portobello street
S14DP Sheffield
United Kingdom
Fax: +44 114 2221810, Phone: +44 114 2221891
Contact person: Hamish Cunningham
E-mail address: hamish@dcs.shef.ac.uk

University of Trento (UniTn)
Via Sommarive 14
38050 Trento
Italy
Fax: +39 0461 882093, Phone: +39 0461 881533
Contact person: Fausto Giunchiglia
E-mail address: fausto@dit.unitn.it

Vrije Universiteit Amsterdam (VUA)
De Boelelaan 1081a
1081HV. Amsterdam
The Netherlands
Fax: +31842214294, Phone: +31204447731
Contact person: Frank van Harmelen
E-mail address: Frank.van.Harmelen@cs.vu.nl

Vrije Universiteit Brussel (VUB)
Pleinlaan 2, Building G10
1050 Brussels
Belgium
Fax: +32 2 6293308, Phone: +32 2 6293308
Contact person: Robert Meersman
E-mail address: robert.meersman@vub.ac.be

Changes

Version	Date	Author	Changes
2.0	04.06.07	Raul Palma	Creation
2.1	25.06.07	York Sure	Content evaluation input
2.2	10.07.07	Raul Palma	Content update (merge content evaluation input and ontology repository input)
2.3	04.08.07	Raul Palma	Content update
2.4	08.08.07	Raul Palma	Content update

Executive Summary

Ontologies have seen quite an enormous development and application in many domains within the last years, especially in the context of the next web generation, the Semantic Web. Besides the work of countless researchers across the world, industry starts developing ontologies to support their daily operative business. Currently, most ontologies exist in pure form without any additional information, e.g. missing domain specific or application related information, such as provided by Dublin Core for text documents, denoting the difficulty for academia and industry to identify, find and apply – basically meaning to reuse – ontologies effectively and efficiently. Our contribution consists of a proposal for a metadata standard, so called Ontology Metadata Vocabulary (OMV) and two complementary reference implementations which show the benefit of such a standard in decentralized and centralized scenarios, i.e. the Oyster P2P system and the up-and-running metadata portal Onthology ("anthology of ontologies"), which implements the proposed OMV to support users in accessing and reusing of ontologies. Both applications were presented in the previous version of this deliverable (D1.2.10) and therefore we present here some key results derived from their evaluation. The sustainability of this work is driven by reusing and exploiting the resulting applications and ontologies in other EU projects (e.g. NeOn Project[1]). Furthermore, we describe how knowledge is exchanged between these two applications in order to provide a comprehensive infrastructure to support the sharing and reuse of ontologies. The flow of knowledge relies on the definition of quality measures for ontologies introduced on this deliverable.

You can only control what you can measure. Measuring ontologies is necessary to evaluate ontologies both during engineering and application. Metrics allow the fast and simple assessment of an ontology and also to track their subsequent evolution. In the last few years, a growing number of ontology metrics and measures have been suggested and defined. But many of them suffer from a recurring set of problems, most importantly they do not take the semantics of the ontology language properly into account. The work presented here is a principal approach to facilitate the creation of ontology metrics with the clear goal to go beyond structural metrics to proper semantic-aware ontology metrics. We have developed guidelines and a set of methodological tools based on the notions of "normalization" and "stable metrics" for creating ontology metrics. These guidelines allow the metric author to decide which properties metrics need to fulfil and to appropriately design the desired metric. A discussion of an exemplary metric (taken from literature)

[1]http://www.neon-project.org/

illustrates and motivates the issues and suggested solutions. The normalization has been prototypically implemented as part of the KAON2 OWL Tools (source code is available at OntoWare.org). Parts of this work have been already accepted for publication at the European Semantic Web Conference (ESWC2007).

Contents

Chapter 1

Introduction

Ontologies are commonly used for a shared means of communication between computers and between humans and computers. To reach this aim, ontologies should be represented, described, exchanged, shared and accessed based on open standards such as the W3C standardized web ontology language OWL. However, most ontologies today exist in a pure form without any additional information about authorship, domain of interest and other meta data about ontologies. Therefore, searching and identifying existing ontologies which are potentially reusable because they e.g. are applied in similar domains, used within similar applications or who have similar properties is a rather hard and tedious task.

We argue that metadata in the sense of machine processable information for the Web[1] helps to improve accessibility and reuse ontologies. Further, it can provide other useful resource information to support maintenance. Thus, we claim that metadata not only helps when applied (or, attached) to documents, but also to ontologies themselves.

As a consequence, ontologies which are annotated by metadata require an appropriate technology infrastructure as well. This includes tools and metadata repositories which comply to the ontology metadata standard and which provide the required functionalities to support reuse of ontologies. Such tools and repositories typically should support the engineering process, maintenance and distribution of ontologies.

Furthermore, a critical aspect to take into account when reusing ontologies is the quality of the ontology. In general, ontologies with a certain degree of quality become well known and accepted in the community and are commonly reused. Therefore, it will largely benefit the reuse of ontologies if we have a measure over the quality of the ontologies.

Did you ever dare to raise the issue of ontology quality assurance? How did you control the process of improvement? As in many other related fields, you can only control what you can measure [DeM82]. Measuring ontologies is necessary to evaluate ontologies

[1]`http://www.w3.org/Metadata/`

both during engineering and application and is a necessary precondition to perform quality assurance and control the process of improvement. Metrics allow the fast and simple assessment of an ontology and also to track their subsequent evolution. In the last years, many ontology metrics and measures have been suggested and some principal work has been done to study the nature of metrics and measures for ontologies in general. We are extending this work.

There is a recurring set of problems with existing ontology metrics and measures, whereby we focus on the W3C standardized ontology language OWL [SWM04]. We argue that most metrics are based on *structural notions* without taking into account the *semantics* which leads to incomparable measurement results. First, most ontology metrics are defined over the RDF graph that represents an OWL DL ontology and thus are basically graph metrics which take only structural notions into account. Second, only a very small number of metrics is taking the semantics of OWL DL into account (subsumption etc.). Third, almost no metric is taking the open world assumption into account. We believe that foundational work addressing these issues will substantially facilitate the definition of proper ontology metrics in the future.

In this work we will study these issues, describe how they can be avoided, and under what circumstances they have to be avoided, and under which they are acceptable. We will outline the foundations for a novel set of metrics and measures, and discuss the advantages and problems of the given solutions. Our approach is based on two notions, first "normalization" of an ontology, and second "stable metric".

Normalization consists of the five steps (i) name anonymous classes, (ii) name anonymous individuals, (iii) materialize the subsumption hierarchy and unify names, (iv) propagate instances to deepest possible class or property within the hierarchy, and (v) normalize property instances. We argue that such a normalization is useful as a kind of pre-processing in order to apply known structural metrics *in a semantics-aware way*. For instance, a known structural metric is the depth of of the class-hierarchy. However, the current measures of ontology depth depend on a number of structural parameters such as whether subsumption reasoning has been performed and whether the results have been materialized before measurement. Performing the normalization steps before measuring ensures that the value for the maximum depth of an ontology is comparable to the maximum depth of another ontology. The normalization has been implemented as part of the KAON2 OWL Tools[2].

Stable metrics are metrics that take the open world assumption properly into account, that means that they are stable with regards to possible additions of further axioms to the ontology. Stable metrics allow us to make statements about the behaviour of an ontology in the context of a dynamic and changing world wide web, where ontologies may frequently be merged together in order to answer questions over integrated knowledge. We give an exemplary extension of the depth metric towards a stable metric in order to demonstrate how a classic metric can be turned into a stable one.

[2]Source Code is available at `http://owltools.ontoware.org/`

In this work we assume the term to include both axioms and facts (as well as annotations and ontology properties, although those are not taken into regard for normalization), i.e. the TBox and the ABox. Here, ontology does not mean only the axioms (as it is assumed in many other works), but also a knowledge base, and any of them could be empty. Thus we follow the definition of ontology in the OWL standard [SWM04].

The deliverable is structured as follows. Chapter 2 presents the results from the work presented in the previous version of this deliverable (D1.2.10). In section 2.1 we discuss the sustainability of the ontology metadata vocabulary proposed as a standard model for describing ontologies. In section 2.2 we propose a topic hierarchy based on DMOZ for classifying the ontology domain. In section 2.3 we discuss how the knowledge is exchanged between the two applications presented in the previous version of this deliverable (D1.2.10) (i.e. Oyster and Onthology) and we highlight the necessity in this process of quality measures for ontologies (described later in this deliverable). Finally, in section 2.4 we present the evaluation results of Oyster in the form of usage statistics and a brief discussion about future plans. In Chapter 3 we introduce the content evaluation work. In section 3.1 we will examine existing metrics and measures, and thus survey related work. section 3.2 contrasts the underlying notions of semantic metrics with structural metrics, and discusses which scenario will require what kind of metric. Section 3.3 introduces the notion of "normalization" of an ontology which forms the heart of our approach. In section 3.4 we illustrate the practical application of normalization on examples. Section 3.5 addresses the issue of stable metrics with regards to the open world assumption. We conclude in Chapter 4, where we also discuss future work.

Chapter 2

Ontology repositories

2.1 Ontology Metadata Vocabulary

The ontology metadata vocabulary (OMV[1]) introduced in [HPP05] is the proposal for standard intended to capture reuse-relevant information about ontologies (i.e. ontology metadata) in a machine-understandable form (see complete description at `http://omv.ontoware.org/`)

The benefits of using OMV for describing ontologies were illustrated by the two complementary applications Oyster and ONTHOLOGY introduced in [HPP05]. Both applications rely on OMV to annotate ontologies in order to provide an efficient solution for accessing and reusing ontologies in decentralized and centralized scenarios (for more information see deliverable [HPP05]).

2.1.1 Sustainability of OMV

OMV has not only been successfully implemented by semantic web applications within the Knowledge Web project as it is described in [HPP05], but additionally, OMV also has been reused within the NeOn EU Project[2] as the standard to model the metadata for networked ontologies (see deliverable [HRW$^+$06]) hence it has been implemented by several NeOn applications (e.g. KaonWeb [3], KaonP2P [4], Oyster2 [5]).

As a consequence, OMV had to face new challenges that led to the development of a second version which includes a refinement of the core, the use of naming conventions

[1]The ontology is available for download in several ontology formats at `http://ontoware.org/projects/omv/`

[2]`http://www.neon-project.org/`

[3]`http://ontoware.org/projects/kaonweb`

[4]`http://ontoware.org/projects/kaonp2p`

[5]`http://ontoware.org/projects/oyster2`

and the definition of several extensions (the complete technical report can be downloaded from `http://omv.ontoware.org`). In a nutshell, the Conceptualisation class was taken out of the core into the new *Conceptualisation extension* in order to simplify the OMV core and to extend the information related to the conceptual model of the ontology, while the Implementation class was renamed to Ontology. Additionally, the following extensions were developed or are under development:

- *Peer extension* which provides the metadata required to describe peers including descriptive information about the peers themselves, their relationship with other peers, as well as information about the resources they provide.

- *Mapping extension* that captures the metadata about ontology mappings (e.g. the mapping method, the mapping creator, the mapped ontologies, etc.).

- *Multilinguality extension* is under development and it will capture the necessary information to describe ontologies with labels in more than one language.

2.2 Ontology Domain Classification

One of the goals of OMV is to provide all the necessary applicability information about ontologies, including the nature of the content of the ontology, the domain topic of the ontology and the natural language of the content of the ontology, i.e. English, German, etc. Thus, applicability information enables users to build complex semantic queries for searching ontologies based on the domain, type or language. Moreover, applicability information can be exploited by other interesting tasks. For instance, ontologies could be physically stored based on the type of the ontology in order to have related ontologies together. As an other example, in a distributed environment, the domain of the ontology could be used to enable intelligent query routing, so as queries including the ontology domain will be propagated only to nodes that have ontologies about that the domain (see section 2.4). However in order to rely on the domain of the ontology, we need to use an appropriate topic hierarchy to express the ontology domain. Typically, the domain can be expressed as classification against established topic hierarchies such as the general purpose topic hierarchy DMOZ[6] or the domain specific topic hierarchy ACM for the computer science domain. The idea is to recommend one classification scheme, but allow relating to others as well.

In the rest of this chapter we elaborate on the proposal of a recommended topic hierarchy for ontology domain classification. This hierarchy is based on the DMOZ topic hierarchy, but refined to model better the domain of ontologies.

[6]`http://dmoz.org`

2.2.1 DMOZ overview

ODP was founded as Gnuhoo by Rich Skrenta and Bob Truel in 1998 and the Gnuhoo directory went live on June 5, 1998. In October of 1998, it was acquired by Netscape Communications Corporation and became the Open Directory Project. Netscape released the ODP data under the Open Directory License. The Open Direct Project is also known as DMOZ, an acronym for Directory Mozilla. This is the name reflects its loose association with Netscape's Mozilla project, an open source browser initiative.

The Open Directory Project is the largest, most comprehensive human-edited directory of the Web. It is constructed and maintained by a vast, global community of volunteer editors. The Open Directory was founded in the spirit of the open source, and is the only major directory that is 100% free to anyone who agrees to comply with their free use license. There is not, nor will there ever be, a cost to submit a site to the directory, and/or to use the directory's data.

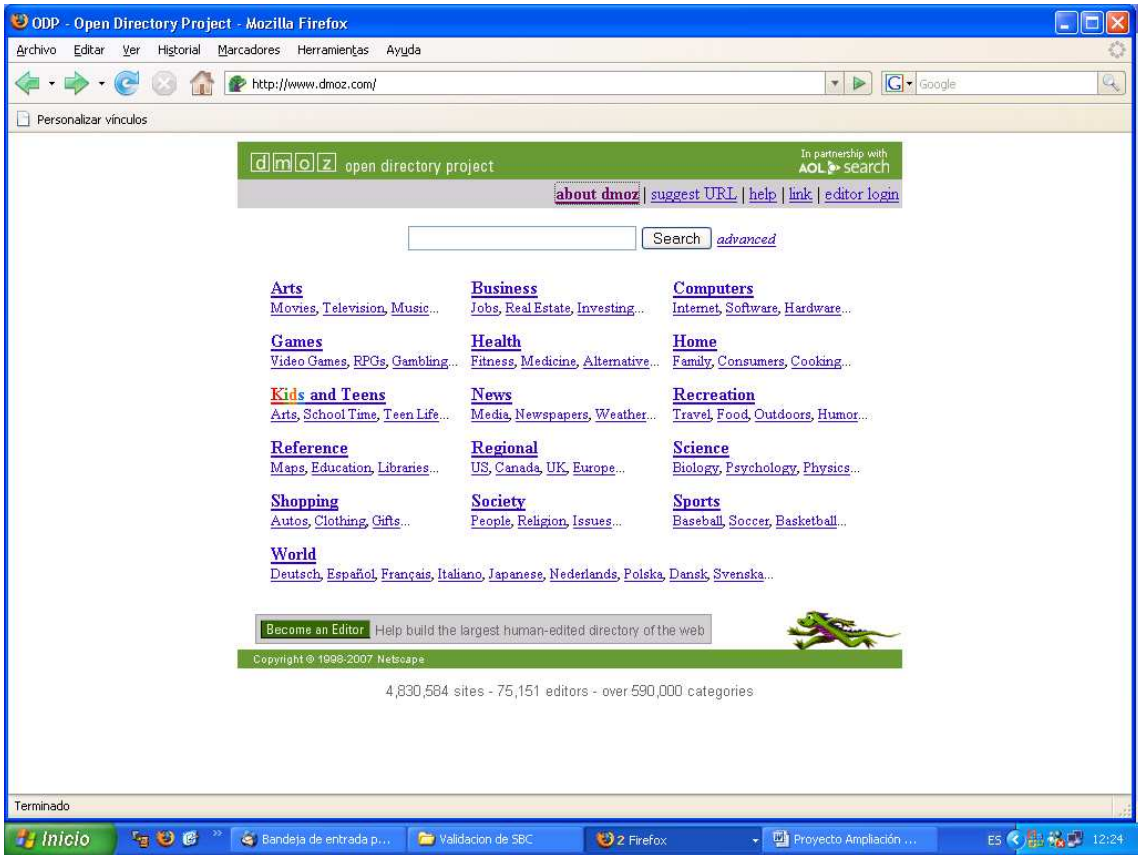
The ODP is a Web directory, not a search engine. It helps the web searching engine to get useful information for internet citizen around the net. The ODP is simply a data provider and is developed and managed by a constantly growing community of net-citizens who are experts in their areas of interest. Given this vast community of subject expertise and the global nature of the directory, there is always someone working on the directory: processing submissions, resolving dead links, culling out the bad and keeping only quality information, and discovering new topics to add.

2.2.2 DMOZ classification

DMOZ uses a hierarchical ontology scheme for organizing site listings (see figure 2.1). Listings on a similar topic are grouped into categories, which can then include smaller categories. This category was base on the popular category list existent at the web on the beginning of DMOZ and it the same used by other project directory on the Web.

As the DMOZ hierarchy is constructed and maintained by a community of volunteer editors, the list of items is very helpful to organize information (e.g. web sites, web information, etc.).

However, usually in order to find an appropriate category, users may have to navigate deep into DMOZ hierarchy which is divided into 0-9 levels. Similarly, it may happen that a category may exists in many different places on the hierarchy creating some confusion to the users. Furthermore, one of the main problems of DMOZ is the reliability of the categorization. As we pointed earlier, DMOZ structure is human-edited where most of the editors are volunteers for each specific area but they might not have knowledge about organization and/or classification. Hence, it may cause several structural problems of the directory.


ODP - Open Directory Project - Mozilla Firefox
Archivo Editar Ver Historial Marcadores Herramientas Ayuda
http://www.dmoz.com/
Google
Personalizar vínculos
dmoz open directory project
In partnership with AOL search
about dmoz | suggest URL | help | link | editor login
Search
advanced
Become an Editor
Terminado
Inicio
Bandeja de entrada p...
Validacion de SBC
2 Firefox
Proyecto Ampliación ...
ES
12:24

Arts
Movies, Television, Music...

Business
Jobs, Real Estate, Investing...

Computers
Internet, Software, Hardware...

Games
Video Games, RPGs, Gambling...

Health
Fitness, Medicine, Alternative...

Home
Family, Consumers, Cooking...

Kids and Teens
Arts, School Time, Teen Life...

News
Media, Newspapers, Weather...

Recreation
Travel, Food, Outdoors, Humor...

Reference
Maps, Education, Libraries...

Regional
US, Canada, UK, Europe...

Science
Biology, Psychology, Physics...

Shopping
Autos, Clothing, Gifts...

Society
People, Religion, Issues...

Sports
Baseball, Soccer, Basketball...

World
Deutsch, Español, Français, Italiano, Japanese, Nederlands, Polska, Dansk, Svenska...

Help build the largest human-edited directory of the web

Copyright © 1998-2007 Netscape

4,830,584 sites - 75,151 editors - over 590,000 categories

Figure 2.1: DMOZ topic hierarchy

2.2.3 DMOZ classification refinement

The initiative to modify/update the DMOZ classification started as the result of trying to classify ontologies domain using the DMOZ topic hierarchy.

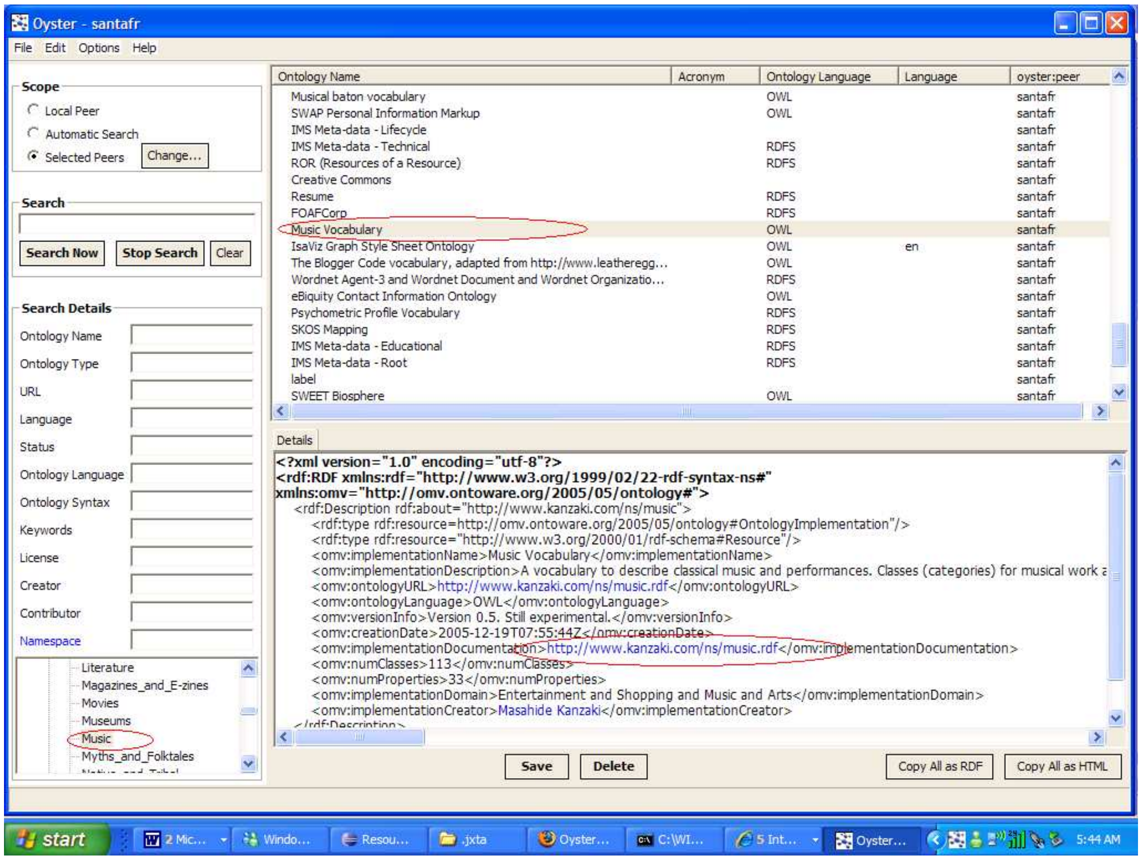
The analysis of DMOZ and the ontology domain showed many ambiguities when assigning a topic. Moreover, some of the ontology domains didn't match appropriately with any item in the hierarchy.

So, our proposed refinement includes changes at the top level (i.e. add and rename some categories at the top level) and also to relocate some subcategories currently under the second, third, forth or fifth level into an upper level to help users to find faster and efficiently an appropriate category for the ontology domain avoiding them to go deep down into the hierarchy browsing the different levels and subcategories.

Analysis process

For the evaluation of the DMOZ topic hierarchy to classify ontologies domain we used the Oyster P2P system (see section2.4). In the following we present the steps followed:

1. Start Oyster in local peer.
2. Search the available peers in the oyster network and retrieve all the available ontologies metadata provided by all of them. In particular, we analyze the ontology metadata available at the well known provider UPM-Main peer (see figure 2.2).
3. Select one ontology metadata from the result list.
4. The ontology metadata is identified and then the ontology itself is retrieved using the location URL.
5. Read and analyze information about the ontology and access (if available) the available web link of the ontology documentation to understand the content (see figure 2.3).
6. Select the appropriate category on the list and drag it into the appropriate row at the result list.
7. Save the ontology metadata on the Oyster local peer.
8. Repeat steps (2-6) for the rest of the ontology metadata available.
9. Export file from local peer with the ontology metadata classified in order to use the information at the UPM-Main Peer.
10. If we do not find a reliable category for the ontology metadata we can not classify it.

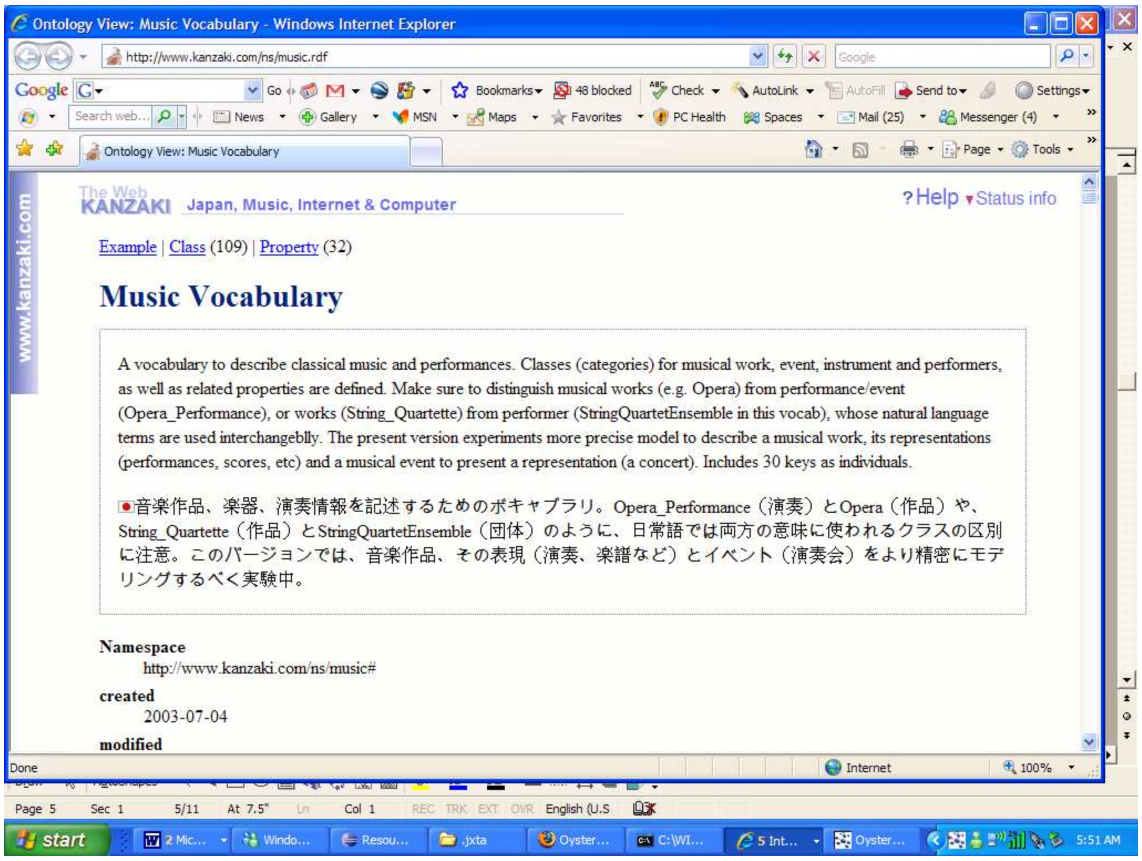


Scope: [✓ Selected Peers]

Ontology Name	Acronym	Ontology Language	Language	oyster:peer
Musical baton vocabulary		OWL		santafr
SWAP Personal Information Markup		OWL		santafr
IMS Meta-data - Lifecycle				santafr
IMS Meta-data - Technical		RDFS		santafr
ROR (Resources of a Resource)		RDFS		santafr
Creative Commons				santafr
Resume		RDFS		santafr
FOAFCorp		RDFS		santafr
Music Vocabulary		OWL		santafr
IsaViz Graph Style Sheet Ontology		OWL	en	santafr
The Blogger Code vocabulary, adapted from http://www.leatheregg...		OWL		santafr
Wordnet Agent-3 and Wordnet Document and Wordnet Organizatio...		RDFS		santafr
eBiquity Contact Information Ontology		OWL		santafr
Psychometric Profile Vocabulary		RDFS		santafr
SKOS Mapping		RDFS		santafr
IMS Meta-data - Educational		RDFS		santafr
IMS Meta-data - Root		RDFS		santafr
label				santafr
SWEET Biosphere		OWL		santafr

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:omv="http://omv.ontoware.org/2005/05/ontology#">
  <rdf:Description rdf:about="http://www.kanzaki.com/ns/music">
    <rdf:type rdf:resource=http://omv.ontoware.org/2005/05/ontology#OntologyImplementation"/>
    <rdf:type rdf:resource="http://www.w3.org/2000/01/rdf-schema#Resource"/>
    <omv:implementationName>Music Vocabulary</omv:implementationName>
    <omv:implementationDescription>A vocabulary to describe classical music and performances. Classes (categories) for musical work a
    <omv:ontologyURL>http://www.kanzaki.com/ns/music.rdf</omv:ontologyURL>
    <omv:ontologyLanguage>OWL</omv:ontologyLanguage>
    <omv:versionInfo>Version 0.5. Still experimental.</omv:versionInfo>
    <omv:creationDate>2005-12-19T07:55:44Z</omv:creationDate>
    <omv:implementationDocumentation>http://www.kanzaki.com/ns/music.rdf</omv:implementationDocumentation>
    <omv:numClasses>113</omv:numClasses>
    <omv:numProperties>33</omv:numProperties>
    <omv:implementationDomain>Entertainment and Shopping and Music and Arts</omv:implementationDomain>
    <omv:implementationCreator>Masahide Kanzaki</omv:implementationCreator>
  </rdf:Description>
```

Figure 2.2: Oyster search result


Ontology View: Music Vocabulary - Windows Internet Explorer
http://www.kanzaki.com/ns/music.rdf
The Web KANZAKI Japan, Music, Internet & Computer
? Help ▾Status info
www.kanzaki.com

Example | Class (109) | Property (32)

Music Vocabulary

A vocabulary to describe classical music and performances. Classes (categories) for musical work, event, instrument and performers, as well as related properties are defined. Make sure to distinguish musical works (e.g. Opera) from performance/event (Opera_Performance), or works (String_Quartette) from performer (StringQuartetEnsemble in this vocab), whose natural language terms are used interchangeblly. The present version experiments more precise model to describe a musical work, its representations (performances, scores, etc) and a musical event to present a representation (a concert). Includes 30 keys as individuals.

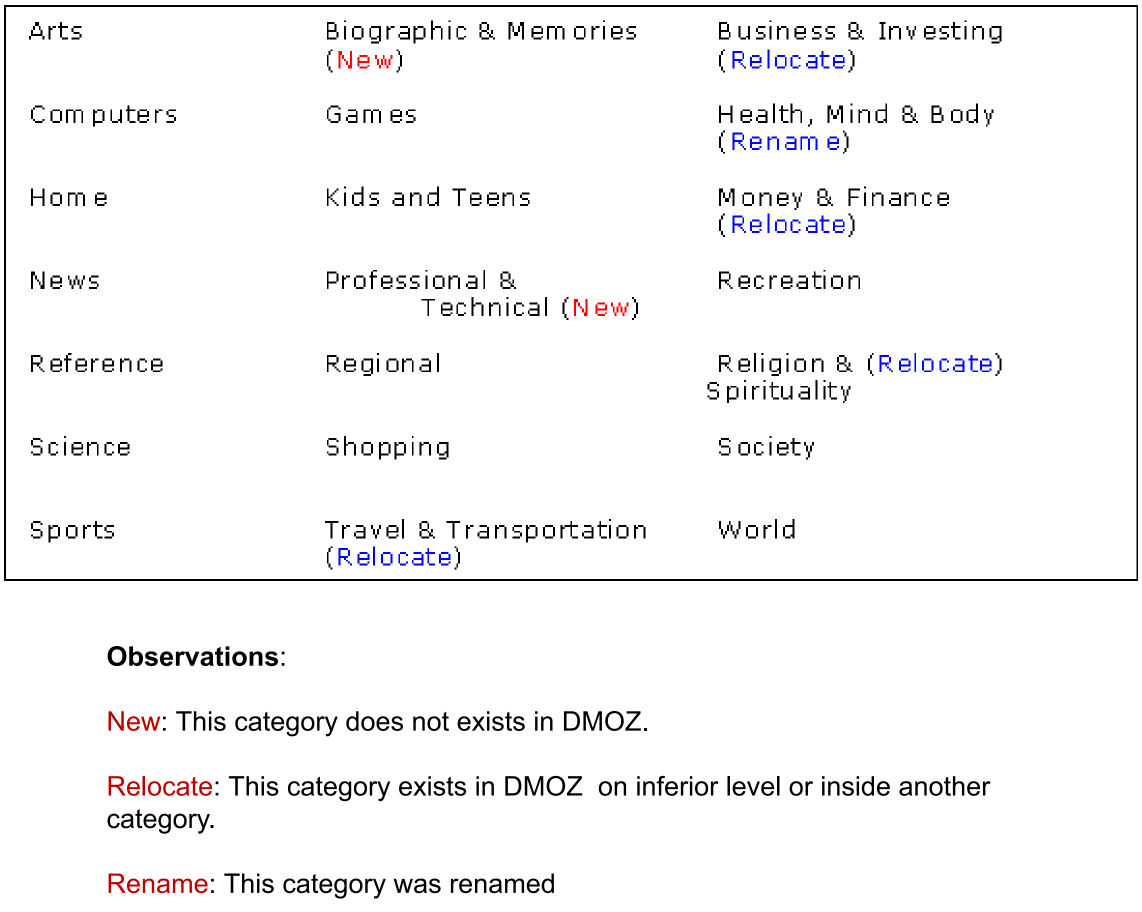
音楽作品、楽器、演奏情報を記述するためのボキャブラリ。Opera_Performance（演奏）とOpera（作品）や、String_Quartette（作品）とStringQuartetEnsemble（団体）のように、日常語では両方の意味に使われるクラスの区別に注意。このバージョンでは、音楽作品、その表現（演奏、楽譜など）とイベント（演奏会）をより精密にモデリングするべく実験中。

Namespace
http://www.kanzaki.com/ns/music#

created
2003-07-04

modified

Figure 2.3: Ontology documentation sample



Arts	Biographic & Memories (New)	Business & Investing (Relocate)
Computers	Games	Health, Mind & Body (Rename)
Home	Kids and Teens	Money & Finance (Relocate)
News	Professional & Technical (New)	Recreation
Reference	Regional	Religion & (Relocate) Spirituality
Science	Shopping	Society
Sports	Travel & Transportation (Relocate)	World

Observations:

New: This category does not exists in DMOZ.

Relocate: This category exists in DMOZ on inferior level or inside another category.

Rename: This category was renamed

Figure 2.4: Proposed Top Level

Proposed Hierarchy

Figure (2.4) shows the new appearance of the DMOZ front page after implementing the suggested changes. For the proposed hierarchy we only focus on the first two levels in order to allow a quick search of the target topic. Besides, in our experimental scenario, Oyster only shows the first two levels of the category list.

In the following we present the description of the individual proposed changes to the hierarchy including for each of them the current DMOZ location of the topic, the possible subcategories, the justification of the change and additional references.

Biographic & Memories

- *DMOZ location*: This category is already on DMOZ. They are mentioned separately as part of other items
- *Possible Subcategories*:
  - Artists
  - Educators
  - Philosophers
  - Writers
- *Justification*: Under this category we can classify all domains related to Writers and Literature. The analysis of the sample set of ontologies showed an important percentage of ontologies referring to these domains.
- *Reference*:
  - `http://www.amazon.com/`
  - `http://www.burkinaphonebook.com/en/`
  - `http://www.burkinaphonebook.com/en/classification/`

Business & Investing

- *DMOZ location*: The business category is already on DMOZ but the world category is a subcategory of business.
- *Possible Subcategories*:
  - All Investing items
  - Management
  - Tax Planning
  - Personal Investment
- *Justification*: Generally, when thinking about business people think of investing. Many of the ontologies analyzed included terms related to business and investing, therefore these ontologies would be better classified under this category.
- *Reference*:
  - `http://www.dmoz.com/`
  - `http://www.dmoz.com/Business/Investing/`

Community

- *DMOZ location*: It is already at the fourth category of Society and is part of Religion and spiritually and is inside of Sport and Health category.
- *Possible Subcategories*:
  - All community items
  - Places of Worship
  - Libraries
  - City Government
  - Autonomous Community
- *Justification*: We suggest to move to the second level of Society.
- *Reference*:
  - `http://www.burkinaphonebook.com/en/`
  - `http://www.burkinaphonebook.com/en/classification/`

Health, Mind & Body

- *DMOZ location*: Currently the Health category already exists, the classification mind and body are not present inside of this category but other possible subcategories.
- *Possible Subcategories*:
  - Mental treatment
  - Medical Center
  - Hospital
  - Body Health
  - Body treatment
- *Justification*: All items related to health should be grouped together to help users in the classification of ontologies related to this topic.
- *Reference*:
  - `http://dir.yahoo.com/Health/`
  - `http://www.dmoz.com/Health/`

Money & Finance

- *DMOZ location*: Is part of third and fourth level of Regional and second level in Business and Society.
- *Possible Subcategories*:
  - Banks
  - Accountants
  - Insurance
  - Credits
  - Taxes
  - Debits
- *Justification*: This topic helps users to classify ontologies about finance and money operations domain. We found many ontologies including terms about money and finance.
- *Reference*:
  - `http://www.dmoz.com/`
  - `http://www.amazondirectory.net/`

Professional & Technical

- *DMOZ location*: This category is not present in DMOZ. It is only mentioned on Shopping as fourth level and does not has any subcategory associated.
- *Possible Subcategories*:
  - Acupuncture
  - Architecture
  - Computer Technician
  - Dentistry
  - Designer
  - Electricity
  - Engineering
  - Finance
  - Law
  - Medicine

  - Pharmacy
  - Police Officer
  - Modelling
  - Mechanics
  - Nurse

- *Justification*: We found many ontologies describing job occupations, so it was easier when classifying those ontologies if the occupations where all grouped together.

- *Reference*:

  - `http://www.op.nysed.gov/proflist.htm`
  - `http://www.dmoz.com/`

Religion & Spirituality

- *DMOZ location*: This category is already on the second level of Society and the third level of Science but could be at the first level due to its relevance, so we only move it up.

- *Possible Subcategories*:

  - Christianity
  - Judaism
  - New Age
  - Pagan
  - Taoism
  - Islam
  - Confucianism

- *Justification*: Moving up this category in the hierarchy will help users classifying ontologies about different beliefs.

- *Reference*:

  - `http://www.dmoz.com/`
  - `http://dir.yahoo.com/Society_and_Culture/Religion_and_Spirituality/Faiths_and_Practices/`

Travel & Transportation

- *DMOZ location*: This category is already at the second level of Recreation; Transportation is on the third level. For travel activity you need a mean of transportation we suggest to modify Travel by Travel & Transportation.
- *Possible Subcategories*:
  - Travel Agents
  - Taxis
  - Hotels
  - Resort
  - Tourism
  - Buses
  - Airport
  - Sea port
  - All categories already in travel.
- *Justification*: Travel is present in many different places of the hierarchy, so the idea is to group all the different subcategories in only one place to avoid confusion. Besides we included transportation topic in this category because when you travel you need a mean of transportation (which is also usually modelled in the ontologies).
- *Reference*:
  - `http://www.dmoz.com/Recreation/Travel/Transportation/`
  - `http://www.dmoz.com/Recreation/Travel/`
  - `http://www.burkinaphonebook.com/en/Travel-and-Tourism/`

2.3 Integration and Knowledge exchange between Oyster and ONTHOLOGY

In [HPP05], we introduced the two complementary applications Oyster and ONTHOLOGY which implements the proposed OMV in order to support users in accessing and reusing ontologies in decentralized and centralized scenarios. The Oyster P2P system for sharing ontology metadata in an early stage of ontology engineering. Well-accepted or recommended ontologies within Oyster are shipped to the up-and-running metadata portal ONTHOLOGY (anthology of ontologies). We argue that both applications are covering a variety of different tasks. For users who want to store metadata individually similar to managing his personal favorite song list, a repository is required to which a user has full access and can perform any operation (e.g. create, edit or delete metadata) without any consequences to other users. For example, users from academia or industry might use a personal repository for a task dependant investigation or ontology engineers, might use it during their ontology development process to capture information about different ontology versions. We argue that a decentralised system is the technique of choice, since it allows the maximum of individuality while it still ensures exchange with other users. Centralised systems allow reflecting long-term community processes in which some ontologies become well accepted for a domain or community and others become less important. Such well accepted ontologies and in particular their metadata need to be stored in a central metadata portal which can be accessed easily by a large number of users. The benefit of connecting both systems lies mainly in the simple use of existing ontology metadata information within Oyster. So, while users are applying or even developing their own ontologies they can manage their own metadata along with other existing metadata in one application (in Oyster). If some metadata entries from Oyster have reached a certain confidence, an import into ONTHOLOGY can be performed easily. In combination, both systems ensure efficient and effective ontology metadata management for various use cases. The ontologies that will be imported from Oyster to ONTHOLOGY are selected based on recommendations and quality evaluation mechanisms (defined in chapter 3 of this deliverable). As exchange mechanism we rely on the Open Archive initiative[7] which has been successfully applied for a large number of digital libraries among the world. The developed OMV is used as exchange language.

In the next section we present the evaluation results and future plans of Oyster P2P system. According to our information, there are currently no plans to extend ONTHOLOGY, which is why our evaluation focuses on Oyster.

2.4 Oyster

As it was presented in the previous version of this deliverable, Oyster is a java-based Peer-to-Peer application that exploits semantic web techniques in order to provide an innovative and useful solution for exchanging and reusing ontologies. In order to achieve this purpose, Oyster provides facilities for managing, searching and sharing ontology metadata in a P2P network, thereby implementing the OMV proposal for the standard set of ontology metadata (for further information we refer the reader to `http://oyster.ontoware.org/`)

[7]http://www.openarchives.org/

2.4.1 Evaluation

Beta evaluation

Initially, Oyster system was tested by UPM with the collaboration of UKarl (approx 10 peers). As a result, we received the following feedback:

- Include the namespace of the ontology as a possible criteria for searching ontologies,
- Include a short usage guide of how to use templates.
- Use user friendly names in the user interface for the property names of the metadata,
- Include hints for the meaning of each property in the user interface,
- Include a link to a more detailed description of the meaning of the properties (e.g. OMV properties) in the help menu,
- Minor fixes in the interface,
- Include a Macintosh version of Oyster (on progress)

Production evaluation

The previous comments were addressed and a new version of Oyster was released and evaluated. In the rest of this section we present the process of evaluation for Oyster system regarding usage statistics.

In order to compile usage information, Oyster requests the user permission when it is installed for sending usage statistics (i.e. logs) to a central server. This information was collected during a period time of 6 months, in randomly chosen days of the months. We logged user behavior and actions in a time window of 15 minutes. During this time period the peers issued 333 queries overall and received 29076 distinct results in terms of RDF statements, which corresponds to about 3000 ontology metadata entries.

A total of 250 ontology metadata entries were shared, among 42 peers, with an average of 6 ontologies per peer. However, the distribution had a high variance: While only three peers shared 85% of the total content, a lot of peers provided only one or two entries or were free-riding.

The analysis of the type of query the users sent revealed that around 50% of the queries were using at least one of the two ontologies used in Oyster (i.e. OMV and the topic hierarchy). OMV properties (e.g. name, type) were used in more that the 27% of the queries and in over 23% of the queries the users asked for topics of the DMOZ topic hierarchy. Based on that analysis we could determine that the preferred properties for searching ontologies by the users are the domain, the name and the ontology language. The other 50% of the queries were general searches (i.e. get all the available ontologies).

Due to the limited size of the network and the lack of domain information on most of the ontology metadata entries, the evaluation of the expertise based peer selection couldn't be shown. However, in simulation experiments with larger peer networks with thousands of peers, it has been shown improvements in order of one magnitude in terms of recall of resources and relevant peers[HSvH04]. For evaluation of other Swapster based system we refer the reader to [HBE$^+$04].

The latest release of Oyster has been downloaded 184 times (140 windows + 44 linux) from ontology engineering platform OntoWare[8]. It is in the list of top downloaded projects of Ontoware (824 downloads, including all versions and releases).

Sustainability of Oyster

Oyster has been successfully accepted and used within the semantic web community. It has been demonstrated by the number of downloads and feedback received that users consider Oyster an interesting semantic web application. Additionally, Oyster was elected one of the five best semantic web applications in the Semantic Web Challenge 2005 organized in conjunction with the forth International Semantic Web Conference.

As part of the sustainability plans for Oyster, it is being reused within the NeOn EU Project[9] as one of the components of the NeOn basic infrastructure. Consequently, Oyster had to face many new challenges which led to the development of Oyster2[10]. The new version of Oyster includes several changes in the architecture (e.g. it uses an OWL infrastructure instead of RDF infrastructure). In addition of having a GUI to interact with Oyster, Oyster2 includes a well defined API that makes it easy to integrate it and use it within other applications. Currently we are working on the definition of a Web Service to allow an easy loosely coupled integration of Oyster2 with other applications.

We expect to have a large dissemination of Oyster(2) as it will be delivered as one of the components of the NeOn toolkit. Finally our future plans include providing support for additional functionalities/tasks (e.g. support for change propagation between ontologies and related metadata, support of networked ontologies, support of trust information, etc.)

[8]`http://ontoware.org/`

[9]`http://www.neon-project.org/`

[10]`http://ontoware.org/projects/oyster2/`

Chapter 3

Ontology evaluation

3.1 Current metrics and measures

In this deliverable we will concentrate on some foundational aspects that form the base for automatically acquirable measures. Therefore we will not define a number of metrics and measures, but rather take a step back and discuss conditions that measures have to adhere to in order to be regarded as semantically aware ontology metrics. This also helps to understand clearly what it means for a metric to remain on a a structural level.

Thus the scope of this work compares best to other metric frameworks, like the QOOD (quality oriented ontology description) framework [GCCL06b] and the O^2 and *oQual* models [GCCL06a]. The authors created semiotic models for ontology evaluation and validation, and thus describe how measures should be built in order to actually assess quality. They also describe the relation between the ontology description, the ontology graph, and the conceptualization that is expressed within the graph, and they define measures for the structural, functional, and usability dimension. In [GCCL05] they introduce further measures that can be applied within that framework. We will take one of the measures introduced in [GCCL05] as an example in Chapter 3.4, and show some shortcomings of the actual descriptions of such a measure (not of the framework as a whole!). We think that the work described here fits well into the QOOD framework by making the assumptions underlying such measures explicit.

A framework for metrics in the wider area of ontology engineering is provided by OntoMetric [LTGP04]. The authors name and sort a long list of metrics into several different areas, like tools, languages, methodologies, costs, and content. They define the relations between the different metrics, their attributes, and the quality attributes they capture. Within the OntoMetric framework, the work presented in this deliverable is based solely in the area of content metrics. It extends the discussions around content metrics, and elaborates properties of such metrics in more detail. Whereas OntoMetric regards all kind of metrics, we gear the results described here towards automatically measurable metrics.

OntoQA is a tool that implements a number of metrics [TAM$^+$05], and thus it allows for the automatic measurement of ontologies. They define metrics like richness, population, or cohesion. Whereas all these metrics are interesting, they fail to define if they are structurally or semantically defined – which is a common lapse. Most of the metrics in OntoQA actually can be applied both before and after normalization (as described in the following chapter). We suppose that comparing these two measures will yield further interesting results.

Often metrics are defined purely structural. An example is given by [AB06], where the authors describe metrics for ranking ontologies, like the class match or the density measure. Interestingly even the so called semantic similarity measure is not a semantic measure in the sense described here, since they apply all these measures on the graph that describes the ontology, not on the ontological model.

OntoClean [GW02], currently the most well-known ontology evaluation approach, is a philosophically inspired approach for the evaluation of formal properties of a taxonomy. Some tools offer support for the manual tagging with OntoClean properties (OntoEdit [SAS03] and WebODE [ACFLGP01]), a recent work deals with the automation of OntoClean [VVS05]. From a practical perspective OntoClean provides means to derive measurable mismatches of a taxonomy with respect to an ideal structure which takes into account the semantics of the “is-a” relationship. Such mismatches have a structural nature, e.g. one is able to derive that a certain concept should not be the subconcept of another concept. OntoClean provides an explanation of why mismatches occur which subsequently might help to improve the taxonomical structure. For many people the philosophical notions of OntoClean are subject of long discussions, however, strictly speaking, this is not part of the evaluation but of the ontology engineering because deciding the proper nature of a class forces the ontology to commit itself to a more specified meaning, which in turn allows for a more objective evaluation technique.

Measures applied to ontologies from the Semantic Web are usually still in a very simple state [Wan06] (unsurprising due to the overall bad quality of ontologies in the wild, and the high costs on resources for providing reasoning on a big number of ontologies). We think that the most prevalent hurdle towards applying more semantic measures on ontologies on the web is an actual lack of some foundational work towards defining such measures, and a subsequent lack of implementation. The work presented here is a step towards such an implementation, that will allow to measure the web in several new dimensions.

3.2 Ontological metrics

As shown in the previous chapter, current metrics often measure structural properties of the ontology. In the case of OWL DL, this often means that they measure the structure of the RDF graph that describes the ontology with well-known graph measures. Another approach is to measure the explicitly stated facts and axioms. Within this deliverable, we regard both approaches as structural. Structural metrics are often useful, and this work does not suggest to replace them. It rather offers a way to extend the possibilities available to the ontology engineer with truly ontological metrics.

We define ontological, or semantic, metrics to be those who do not measure the structure of the ontology, but rather the models that are described by that structure. In a naïve way, we could state that we base our metrics not on the explicit statements, but on every statement that is entailed by the ontology.

But measuring the entailments is much harder than measuring the structure, and we definitively need a reasoner to do that. We also need to make a difference between a statement X that is entailed by an ontology O to be true ($O \models X$), a statement that is not entailed by an ontology ($O \not\models X$), and a statement that is entailed not to be true ($O \models \neg X$). To properly regard this difference leads us to so called stable metrics that can deal with the open world assumption of OWL DL. We will return to them in Chapter 3.5.

Note that measuring the entailments is more an intuitive description of how to describe ontological metrics than the actual approach. In many cases – for example for a measure that simply counts the number of statements in an ontology – measuring all entailed statements instead of measuring all explicit statements often leads to an infinite number of statements. Just to give one example, the ontology $\exists R.\top \sqsubseteq C$ also entails the statements $\exists R.\exists R.\top \sqsubseteq C$, $\exists R.\exists R.\exists R.\top \sqsubseteq C$, and so on, an endless chain of existentials. But only terminating measures are of practical interest, and thus we need approaches that allow us to capture ontological metrics in a terminating way.

In order to gain the advantage of the simple and cheap measurement of structural features, we can transform the structure of the ontology. These transformation need to preserve the semantics of the ontology, that is, they need to describe the same models. But they also need to make certain semantic features of the ontology explicit in their structure – thus we can take structural measures of the transformed ontology and interpret them as ontological measures of the original ontology. We call this kind of transformations normalization. The following chapter describes five steps of normalization.

With these tools we will be enabled to define ontological metrics in a simpler and less error prone way than in current practice. We will show this on an exemplary metric in Chapter 3.4.

3.3 Normalization of an ontology

This Chapter describes several steps of normalization. Their goal is to explicate some features of the semantics of an ontology within its structure, so that the structural metrics actually capture the semantics they are supposed to capture.

The following normalization steps are defined here:

1. name all relevant classes, so no anonymous complex class descriptions are left
2. name anonymous individuals
3. materialize the subsumption hierarchy and normalize names
4. instantiate the deepest possible class or property
5. normalize property instances

Notice that if we speak of names, we mean, in the context of OWL DL, the URI of the class, property, or individual, not the human readable label.

3.3.1 First normalization

In the **first normalization** our aim is to get rid of anonymous complex class descriptions. After the first normalization, the TBox will contain two kind of axioms: class definitions of the form $A \equiv C$, where A is a class name and C a class description (or class name), and subsumption axioms of the form $A \sqsubseteq B$, where both A and B are class names. The ABox will consist of property instantiations of the form $R(i, j)$, and of facts of the form $A(i)$, with A being a class name.

The first normalization can be done as follows:

1. in all axioms of the form $C \sqsubseteq D$ where C (or D) is a complex class description, add a new axiom $A \equiv C$ ($B \equiv D$) with A (B) being a new class name. Replace the axiom $C \sqsubseteq D$ with $A \sqsubseteq D$ ($C \sqsubseteq B$, or even $A \sqsubseteq B$)
2. in all axioms of the form $C \equiv D$ where both C and D are complex class descriptions, replace that axiom with the two axioms $A \equiv C$ and $A \equiv D$, with A being a new class name
3. in all axioms of the form $C \equiv A$ where C is a complex class descriptions and A an atomic class name, replace that axiom with $A \equiv C$
4. in all axioms of the form $C(i)$ where C is a complex class description, replace that axiom with the axioms $A(i)$ and $A \equiv C$ with A being a new class name

None of these structural changes change the possible models, that means, that they are semantically equivalent. They do introduce new class names to the ontology, which may not be desirable in all cases (for example for presentation purposes, for counting the classes, and so on).

Note that it is possible to introduce named classes that are unsatisfiable. This does not mean that the ontology becomes unsatisfiable, but solely these newly introduced classes. Instead of introducing new names for unsatisfiable classes though, we could simply use the name $\perp$.

3.3.2 Second normalization

The **second normalization** gets rid of anonymous individuals. This means that every blank node that is of the (asserted or inferred) type individual needs to be replaced with an URI reference. Especially in FOAF [BM05] files this occurs regularly since, for some time, it was regarded as good practice *not* to define URIs for persons. Integration of data was not done via the URI, but with inverse functional properties. This practice is problematic, since the semantics of blank nodes in RDF are rather often not fully understood, and should thus be avoided. The second normalization as defined here captures the semantics most users wanted to express anyway.

It is possible that these newly introduced individual names give a further name to already existing (or other newly introduced) individuals. But since OWL DL does not adhere to a unique name assumptions, this is no problem. Furthermore, the next step of normalization will take care to resolve such synonyms.

3.3.3 Third normalization

The **third normalization** will materialize the subsumption hierarchy and normalize the names. The first step requires a reasoner.

1. for all pairs of simple class names (A, B) in the ontology, add the axiom $A \sqsubseteq B$ if the ontology entails that axiom (that is, materialize all subsumptions between simple named classes).

2. detect all cycles in the subsumption structure. For each set of classes $A_1 \ldots A_n$ that participate in a cycle, remove all subsumption axioms from the ontology where both classes are members of this set. In subsumption axioms where only one class is a member of this set, replace the class with B in the axioms. Add the axioms $B \equiv A_1 \ldots B \equiv A_n$ to the ontology. B is a new class name for each cycle. If B is unsatisfiable, take $\bot$ instead of B. If B is equal to $\top$, take $\top$.

3. regarding solely the subontology H_3 that consists of all subsumption axioms of an ontology O, remove all redundant ones (that is, remove all subsumption axioms that are redundant due to the transitivity of the subsumption relation alone).

The subsumption structure now forms a directed acyclic graph that represents the complete subsumption hierarchy of the original ontology. We define a set of normal classes of an ontology as follows: every class that participates in an subsumption axiom after the third normalization of an ontology is a normal class of that ontology.

Since we got rid of facts with complex class descriptions in the first normalization, we do not need a reasoner in order to take care of fact normalization. We still have to replace every class name that is not normal with its normal equivalent within the facts.

Note that instead of creating a new class name for each detected cycle, often it will make more sense to choose a name from the set of classes involved in that cycle, based on some criteria (like the class name belonging to a certain namespace, the popularity of the class name on the web, etc.). For many ontology metrics, this does not make any difference, so we disregard it for now, but we expect the normalizations to have beneficial effects in other scenarios as well, in which case some steps of the normalization need to be revisited in more detail. We will further discuss this in Chapter 4.

Since in OWL DL it is not possible to make complex property descriptions besides inverse properties, property subsumption, and transitivity, (extensions towards enabling more complex property descriptions are suggested in the OWL 1.1 proposal [Gra06]) no heavy reasoning is involved for property normalization in most cases. In case a property has more than one name, we choose one (or introduce a new name and state the equality). All normal property names have to be stated explicitly to be equivalent to all other property names they are equal to (that is, we materialize the equality relations between the normal property names and the non-normal ones). All occurrences of non-normal property names (besides within the axiom stating equality with the normal property name, and besides within annotation property instances) are replaced with the normal property name.

The same holds true for individuals. In case an individual has more than one name, we decide on or introduce a normal one and state explicitly equality to the normal name, and then replace all occurrences of the non-normal individual names with the normal one (besides within the axiom stating equality with the normal individual name, and besides within annotation property instances).

We disregard annotation property instances since they may be used to state annotations about the URI, and not about the actual concept, property, or individual. There could be annotations that describe when a certain URI was introduced, who created it, its deprecation state, or that point to a discussion related to the introduction of the URI. Some annotations on the other hand may be useful for the normal name as well – especially labels, or sometimes comments. Since annotations do not have impact on the DL semantics of the ontology anyway, they may be dropped for the purpose of measuring semantic metrics. Nevertheless, if the normalization is done for some other purpose, and it is planned to further use the normalized version of the ontology in some scenario, than the possible replacement of names within annotation property instances depends both on the scenario and the instantiated annotation property (for example, it may be useful to normalize the label when the ontology will be displayed on the user interface, but it may be bad to normalize versioning information that is captured within annotations).

3.3.4 Fourth normalization

The **fourth normalization** aims towards moving the instantiations to the deepest possible level, as this conveys the most information explicitly (and deriving instantiations of higher levels is very cheap because of the asserted explicitness of the hierarchy due to third normalization). This does not mean that every instance will belong to only one class, multiple instantiations will still be necessary in general.

Here is a possible (though not efficient) algorithm to perform the fourth normalization of an ontology O.

1. for each normal class C and each normal individual i in O, add $C(i)$ to O if it is entailed by the ontology.
2. for each normal object property instance $R(i, j)$ and each object property S so that $S \sqsubseteq R$ is an explicit axiom in O, add $S(i, j)$ if it is entailed by the ontology. Check this also for the property instances added this way (this step will terminate since the subsumption hierarchy is finite).
3. for each normal data property instance $T(i, d)$ and each data property U, proceed as in the previous step.
4. create a subontology H_4 out of O including only the facts (that is, the ABox), and the explicitly stated subsumption hierarchy of the classes and properties (after third normalization)
5. remove all facts from O that are redundant in H_4

We do not want to remove all redundant facts from the ontology at this step, since there may be some facts that are redundant due to an interplay of different other axioms in the TBox. For example, in the following ontology:

$Person(Adam).$
$likes(Adam, Eve).$
$Person \sqsubseteq \exists likes.\top$

the first statement is actually redundant, but would not be removed by the above algorithm (the third statement states that the domain of *likes* is *Person*). This is because we only remove axioms that are redundant within the subontology H_4, and the axiom stating the domain of *Person* would not be part of it. This is due to the fact that after first normalization, the ontology would look like this:

$Person(Adam).$
$likes(Adam, Eve).$
$Person \sqsubseteq A$
$A \equiv \exists likes.\top$

So H_4 would not include the last axiom, and thus the first axiom would not be redundant within H_4.

3.3.5 Fifth normalization

The **fifth normalization** finally normalizes the properties: we materialize property instances of symmetric and inverse properties, and we clean the transitivity relationship. This can be done similar to the creation of the subsumption hierarchy in the third normalization: after materializing all property instances, we remove all that are redundant in the subontology H_5, which contains only the property instances of all transitive properties, and the axioms stating the transitivity of these properties.

It is important to mention that normalization does not lead to a canonic normalized version. This means that there may be many different ontologies that result from the normalization of an ontology. Often normalizations do not result in canonical, unique results (think about conjunctive normal forms). The normalization as described here can be extended in order to result in canonic normalized forms, but the benefit of such an extension is not clear. Considering that common serializations, like the RDF/XML serialization of OWL ontologies [SWM04], lack a canonic translation anyway, and thus ontologies cannot be compared on a character by character base, for example as some version control systems like CVS or SVN would require.

Also, normalization is not an applicable solution for every metric. For example, if we want to know the number of atomic classes in an ontology, first normalizing it and then calculating the number actually will return the wrong result in the general case. The goal of normalization is to actually provide the metric designer some tools in order to simplify the description of his metric. In the following chapter we describe an example of how to apply the normalization for the description of a metric.

3.4 Examples of normalization

The metric we will regard in this example is the *depth of the ontology*. What we want to measure is intuitively described as the length of the subsumption hierarchy, or else the number of levels the class hierarchy has. In [GCCL05], this is the measure (M3), called *Maximal depth*, and the definition is given as follows:

$$m = N_{j \in P}$$

$$\forall i \exists j (N_{j \in P} \geq N_{i \in P})$$

where $N_{j \in P}$ is the set of all nodes in the path j from the set of all paths through the digraph g that represents the ontology, that is, the definition is the length of the longest succession of explicitly stated subsumption relations.

Let us regard the following ontology:

$C \equiv \geq 1R.\top$
$D \equiv \geq 2R.\top$
$E \equiv \geq 3R.\top$

By the definition of (M3), the depth of the ontology is 1 (since there are no explicitly stated subsumption axioms, every path has one node). But after normalization the ontology gets transformed to this:

$C \equiv \geq 1R.\top$
$D \equiv \geq 2R.\top$
$E \equiv \geq 3R.\top$
$D \sqsubseteq C$
$E \sqsubseteq D$

Now the very same metric, applied to the normalized ontology, actually captures the intuition of the depth of the ontology and returns 3.

As discussed earlier, this example also shows us that some metrics will not work with normalization. In [GCCL05], metric (M30) is the *axiom/class ratio*. On the original ontology it is 1, but raises to 5/3 in the normalized version. In case the original ontology is being distributed and shared, (M30) – if stated as metadata of the ontology, for example in some kind of ontology repository [HSH$^+$05] – should be 1, and not calculated on the normalized version.

Let us regard another example. In the following ontology

$D \sqsubseteq C$
$E \sqsubseteq D$
$D \sqsubseteq E$
$F \sqsubseteq E$

(M3) will be ∞ due to the subsumption cycle between D and E. The cycle can be resolved by rewriting the axioms in the following way:

$D \sqsubseteq C$
$D \equiv E$
$F \sqsubseteq E$

But due to the definition, (M3) would yield 2 here – there are two explicit subsumption paths, C, D and E, F, both having two nodes, and thus the longest path is 2. The structural measure again does not bring the expected result. After normalization, though, the ontology will look like this:

$A \sqsubseteq C$
$A \equiv D$

$A \equiv E$
$F \sqsubseteq A$

We have introduced a new class name A that replaces the members of the cycle, D, E. Now the depth of the ontology is 3, as we would have expected from the start, since the cycle is treated appropriately.

Existing structural metrics, as discussed in Chapter 3.1, often fail to capture what they are meant for. Normalization is a tool that is easy to apply and that can easily repair a number of such metrics. Even seemingly simple metrics, as demonstrated here with the ontology depth, are defined in a way that makes too many assumption with regards to the structure of the measured ontologies.

As we can see in this chapter, simple structural measures on the ontology do yield values, and often these values may be highly interesting. If we know that (M3) resolves to ∞, then this tells us that we have a cycle in the subsumption hierarchy. Also a high number of classes and complex axioms, but a low (M3) may indicate an expensive to reason about ontology, since the major part of the taxonomy seems to be implicitly stated (but such claims need to be evaluated appropriately). But both results do not capture what the measure was meant to express, that is, the depth of the class hierarchy.

But this leads us to the possibility of creating measures by combining structural metrics on the original ontology and on its normalized version, for example to calculate ratios like $M_3(O)/M_3(N(O))$ (with $M_3(O)$ returning measure (M3) as described above, and $N(O)$ being a function that returns the normalized version of the ontology O). This could describe the *explicitness of the subsumption hierarchy*. Further work needs to investigate and evaluate such measures, and to assess their usefulness for evaluating ontologies.

3.5 Stability of metrics

Often metrics intend to capture features of the ontology that are independent of the actual representation of the ontology. But as we have seen, structural transformations of the ontology description often lead to differences in the metrics even though the semantics remained untouched. Normalization offers a way to overcome these problems in many cases.

One aspect of metrics, that are not touched upon by normalization, is the issue of how stable the metrics are with regards to the open world assumption of OWL DL ontologies. In order to illustrate this issue let's take a look at a simple example. Imagine an ontology with the following three facts:

$author(paper, York).$
$author(paper, Denny).$
$author(paper, Zdenko).$

Now let us ask the simple question: how many authors does the $paper$ have? It seems that the answer should be 3. But now, if you knew that $Zdenko$ is just another name for $Denny$, and thus state $Zdenko \approx Denny$, then you suddenly would change your answer to 2, or even, becoming more careful, giving an answer like "I am not sure, it is either 1 or 2". So finally we can state that $York \not\approx Denny$ and thus arrive at the answer that the paper indeed has 2 authors (and even that is possibly wrong if we consider that we could add statements any time in an open world that add further authors to the paper – all we know *as of now* is that the paper has *at least* two authors).

When creating a metric, we have to ask ourselves the following, similar question: how does the metric behave when additions to the ontology happen? Since ontologies are meant to be smushed and integrated constantly and dynamically, can we predict how certain properties of the ontology will behave, that is, if $M(O_1)$ and $M(O_2)$ for a metric M and two ontologies O_1 and O_2 are known, what can we state about $M(O_1 \cup O_2)$? Or even, can we give a function f_M so that $f_M(M(O_1), M(O_2)) = M(O_1 \cup O_2)$ without having to calculate $M(O_1 \cup O_2)$ (which may be much more expensive)?

In the previous chapter we have discussed the simple example of ontology depth. Let us return to this example again. We define the function $M_3(O)$ that returns the measure (M3) as described in [GCCL05], and already described above. If we have an ontology O_1:

$D \sqsubseteq C$
$E \sqsubseteq D$

And a second ontology O_2:

$C \sqsubseteq D$
$E \sqsubseteq D$

In this case, $M_3(O_1) = 3, M_3(O_2) = 2$. We would expect $M_3(O_1 \cup O_2)$ to be 3, since M(3) is defined as the maximal depth, but since the union of both ontologies actually creates a cycle in the subsumption hierarchy, (M3) is ∞ – or, after normalization, just 2, and thus even smaller than the maximal depth before the union.

We can avoid such behaviour of the metrics by carefully taking the open world assumption into account when defining the metric. But this leads us to three possibilities for defining metrics,

1. to base the value on the ontology as it is,
2. to measure an upper bound, or
3. to measure a lower bound.

We need a more complicated example to fully demonstrate these metrics:

$C \equiv D \sqcup E$
$D \sqcap E \sqsubseteq \bot$
$F \sqsubseteq E$
$G \equiv \neg C$
$H \sqsubseteq C$
$F(i).$
$D(j).$
$G(k).$

This ontology says that D and E form a complete partition of C (the first two axioms), that E has the subclass F, that there are elements that are not in C, and it states the existence of three individuals, i, j and k, and the classes they belong to.

The normalized version of this ontology looks like this (shortened slightly for readability):

$C \equiv D \sqcup E$
$\bot \equiv D \sqcap E$
$D \sqsubseteq C$
$E \sqsubseteq C$
$F \sqsubseteq E$
$G \equiv \neg C$
$H \sqsubseteq C$
$F(i).$
$D(j).$
$G(k).$

(M3) of this ontology is 3 (C, E, F). But besides the actual depth, we can also calculate the minimal depth of this ontology, that is, no matter what axioms are added, what is the smallest number of levels the ontology will have (under the condition that the ontology remains satisfiable)?

In the given example, if we add the axiom $F \equiv E$, (M3) will decrease to 2. But on the other hand, no matter what axiom we further add, there is no way to let C collapse with D and E, therefore C is a proper superset of both (that is, it contains more individuals than D or E alone). And because C cannot become $\top$ (due to k being outside of C), the minimum depth of the ontology is 2.

The maximum depth of an ontology is usually ∞ (since we can always add axioms about an arbitrarily long class hierarchy). Only in the case of an ontology with a closed domain, that is, if we have an axiom like $\top \equiv \{a, b, c\}$, then the maximum depth is set (to $|\top| - 1$, since there may be a class C with one element, a class D with two elements that subsumes C, and then $\top$ with three elements, but since $\top$ is not counted, the longest path would be (C, D), and every further class in this path would become equivalent to an already existing class or be empty). But we expect such axioms to usually appear only in theoretical musings and hardly be of any practical relevance.

Therefore we need to define a maximum depth in a slightly different way in order to be of practical value. In the following, we will discuss two possible definitions.

Instead of allowing for arbitrary axioms that may be added, we only allow to add axioms of the form $A \sqsubseteq B$ with A and B being normal class names of the normalized ontology. In the above example, we may add the axiom $H \sqsubseteq F$ to the ontology in order to increase (M3) from 3 to 4. No longer subsumption path is possible, since all the other named classes would become unsatisfiable when added to an existing path. So this metric will provide with a maximum depth of the ontology, assuming no new class names are added.

Another possibility to constrain the axioms to be added, is to allow only for axioms that do not relate to the existing ontology, that is, the intersection of the signatures of the two ontologies is empty. The signature of an ontology is the set of all names used in the ontology (besides the names from the OWL, RDF, RDFS, and XSD namespaces). In this case, (M3) of the merged ontology is the maximal (M3) of the single ontologies, since no interaction between the axioms happen that may increase or reduce (M3). We can thus define $f_{M_3}(M_3(O_1), M_3(O_2)) = \mathsf{max}(M_3(O_1), M_3(O_2))$, which is much cheaper to calculate than $M_3(O_1 \cup O_2)$.

Stable metrics are metrics that take the open world assumption into account. Stable metrics will help us to evaluate ontologies for the wide wild web. Since we expect ontologies to be merged on the web dynamically, stable metrics allow us to state conditions that the ontology will fulfil in any situation. The depth of an ontology may be a too simple example to demonstrate the advantages of stable metrics, but imagine a dynamic, ontology-based graphical user interface. Having certain guarantees with regards to the future development of the properties of the ontology may help the designer of the user interface tremendously, even if it is such a seemingly trivial statement like “the depth of the ontology is never less than 3”.

There is no simple recipe to follow in order to turn a metric into a stable metric, but the question outlined at the beginning of this chapter, and then discussed throughout the rest – how does the ontology behave when axioms are added? – can be used as a guideline in achieving a stable metric.

We expect that the ready availability of metrics that take the open world assumption into account will lead to more robust ontologies. Since ontology engineers will have these numbers available at engineering and maintenance time, they will learn easier how to achieve their actual goals. For example, ontology engineers that want to create a class hierarchy that will not collapse to less levels can always check if the minimum depth as described above corresponds to the asserted depth. Tools could guide the ontology engineer towards achieving such goals. Ontology engineers get more aware of such problems, and at the same time get tools to measure, and thus potentially control them.

Chapter 4

Conclusion and Future Work

Reusing existing ontologies is a key issue for sharing knowledge on the Semantic Web. Our contribution aims at facilitating reuse of ontologies which are previously unknown for ontology developers by providing an Ontology Metadata Vocabulary (OMV) and two prototypical applications for decentralized (Oyster) and centralized (Onthology) sharing of ontology metadata based on OMV .

Important aspects like quality measures which are applied to identify *good ontologies* are of great importance for an efficient integration and usage of Oyster and Onthology.

We proposed a hiearchy for the classification of ontology domains that was the result of the analysis performed on experiments for classifying ontologies using Oyster P2P system. Also, we included usage statistics of Oyster to show the usability of the system as well as the benefits of using OMV for searching ontologies. Furthermore, we discussed the sustainability of our work as the result of reusing OMV and the applications proposed (e.g. oyster) in the context of other EU projects (e.g. NeOn).

We have discussed the properties of ontology metrics. Sometimes simple structural metrics are sufficient for the task at hand, and many structural metrics exist today. Our goal in this work was to raise the awareness for the difference between structural and ontological metrics, and to provide principle means for the simple definition of metrics that take the semantics of the ontology appropriately into account.

Ontology normalization was introduced as a preprocessing step in order to align structural measures with intended semantic measures. Further properties, like the stability of a metric towards ontology extension and merges, and the non-dichotomous nature of ontologies were discussed, and an approach towards encapsulating these problems was suggested by introducing stable metrics.

In addition to offering the theoretical tool of normalization, we have implemented it prototypically as an extension to the KAON2 OWL Tools[1]. The implementation allows to access both from the command line as well as from a Java API.

[1]Source Code is available at `http://owltools.ontoware.org/`

Next steps include the standardization of OMV on a wider scope, followed by a close cooperation with tool providers for ontology engineering environments and applications providers for e.g. OMV support in the envisioned NeOn toolkit platform for networked ontology engineering lyfecycle. The agreement and application of a standard on a global level will greatly facilitate the reuse of ontologies for all participating parties.

Regarding the content evaluation, the work presented here opens the path to future extensions which go well beyond the duration and borders of the project Knowledge Web. We plan to continue our research in many directions.

First, we plan to carry out experiments with real-life ontologies to show the usefulness of normalisation and stable metrics.

Making normalization available to further tools and metric suites will allow us to evaluate if there are further benefits to normalized ontologies. We assume such benefits with regards to query answering performance, usability, and ontology maintenance. Some properties of normalization suggest advantages in these and other areas, but we expect some parts of the normalization process to be adapted based on differing requirements by these other use cases.

Based on the foundational work provided in this deliverable, we plan to adapt, extend, and implement several metrics already known in literature. We hope that a thorough evaluation of these metrics will allow to correlate quality attributes to these metrics, and thus to finally lead to viable sets of metrics and measures for the whole ontology life cycle. We don't think that there is one single such set, but the ideas presented here make several design decision when creating metrics more explicit and point to common problems and pitfalls when creating metrics and measures in this field. This will help in deciding which metrics to choose for a given scenario.

We expect that future work will continue on this basis in order to create a bigger tool set for everybody dealing with ontologies to allow them to evaluate ontologies during every step of the ontology life cycle. This will lead to an overall higher quality of ontologies, and thus to a stronger foundation on which the Semantic Web is being built.

Bibliography

[AB06] H. Alani and C. Brewster. Metrics for ranking ontologies. In Vrandečić et al. [VdCSFGS06].

[ACFLGP01] J. C. Arpírez, O. Corcho, M. Fernández-López, and A. Gómez-Pérez. WebODE: a scalable workbench for ontological engineering. In *Proceedings of the First International Conference on Knowledge Capture (K-CAP) Oct. 21-23, 2001, Victoria, B.C., Canada*, 2001.

[BM05] Dan Brickley and Libby Miller. The friend of a friend (FOAF) vocabulary specification, July 2005. Namespace Document 27 July 2005 ('Pages about Things' Edition).

[DeM82] Tom DeMarco. *Controlling Software Projects: Management, Measurement & Estimation*. Yourdon Press, New York, 1982.

[GCCL05] A. Gangemi, C. Catenacci, M. Ciaramita, and J. Lehmann. Ontology evaluation and validation: an integrated formal model for the quality diagnostic task. Technical report, Laboratory of Applied Ontologies – CNR, Rome, Italy, 2005. available at http://www.loa-cnr.it/Publications.html.

[GCCL06a] Aldo Gangemi, Carola Catenaccia, Massimiliano Ciaramita, and Jos Lehmann. Modelling ontology evaluation and validation. In Y. Sure and J. Domingue, editors, *Proceedings of the 3rd European Semantic Web Conference (ESWC2006)*, number 4011 in LNCS, Budva, Montenegro, June 2006. Springer-Verlag.

[GCCL06b] Aldo Gangemi, Carola Catenaccia, Massimiliano Ciaramita, and Jos Lehmann. Qood grid: A metaontology-based framework for ontology evaluation and selection. In Vrandečić et al. [VdCSFGS06].

[Gra06] Bernardo Cuenca Grau(ed.). OWL 1.1 web ontology language, November 2006. Available at http://owl1_1.cs.manchester.ac.uk/.

[GW02] N. Guarino and C. Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, February 2002.

[HBE$^+$04] P. Haase, J. Broekstra, M. Ehrig, M. Menken, P. Mika, M. Plechawski, P. Pyszlak, B. Schnizler, R. Siebes, S. Staab, and C. Tempich. Bibster - a semantics-based bibliographic peer-to-peer system. In *Proceedings of the Third International Semantic Web Conference, Hiroshima, Japan, 2004*, NOV 2004.

[HPP05] J. Hartmann, R. Palma, and E. Paslaru. Ontology repositories. Knowledge Web Deliverable D1.2.10, UKARL/UPM/FUB, 2005.

[HRW$^+$06] Peter Haase, Sebastian Rudolph, Yimin Wang, Saartje Brockmans, Raul Palma, Jéróme Euzenat, and Mathieu d'Aquin. D1.1.1 networked ontology model. Technical Report D1.1.1, Universität Karlsruhe, NOV 2006.

[HSH$^+$05] Jens Hartmann, York Sure, Peter Haase, Raul Palma, and Mari del Carmen Suarez-Figueroa. OMV – ontology metadata vocabulary. In Chris Welty and Aldo Gangemi, editors, *ISWC 2005 - In Ontology Patterns for the Semantic Web*, Galway, Ireland, November 2005.

[HSvH04] P. Haase, R. Siebes, and F. van Harmelen. Peer selection in peer-to-peer networks with semantic topologies. In *Proceedings of the First International IFIP Conference on Semantics of a Networked World: ICSNW 2004, Paris, France, June 17-19, 2004.*, pages 108–125, 2004.

[LTGP04] A. Lozano-Tello and A. Gómez-Pérez. OntoMetric: A method to choose the appropriate ontology. *Journal of Database Management, Special Issue on Ontological analysis, Evaluation, and Engineering of Business Systems Analysis Methods*, 15(2), April-June 2004.

[SAS03] Y. Sure, J. Angele, and S. Staab. OntoEdit: Multifaceted inferencing for ontology engineering. *Journal on Data Semantics*, 1(1):128–152, NOV 2003. LNCS 2800.

[SWM04] M. K. Smith, C. Welty, and D. McGuinness. OWL Web Ontology Language Guide, 2004. W3C Recommendation 10 February 2004.

[TAM$^+$05] S. Tartir, I. B. Arpinar, M. Moore, A. P. Sheth, and B. Aleman-Meza. OntoQA: Metric-based ontology quality analysis. In *Proceedings of IEEE Workshop on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources*, 2005.

[VdCSFGS06] D. Vrandečić, M. del Carmen Surez-Figueroa, A. Gangemi, and Y. Sure, editors. *Proceedings of the 4th International Workshop on Evaluation of Ontologies for the Web (EON2006) at the 15th International World Wide Web Conference (WWW 2006)*, Edinburgh, Scotland, May 2006.

[VVS05] J. Völker, D. Vrandečić, and Y. Sure. Automatic evaluation of ontologies (AEON). In Y. Gil, E. Motta, V. R. Benjamins, and M. A. Musen, editors, *Proceedings of the 4th International Semantic Web Conference (ISWC2005)*, volume 3729 of *LNCS*, pages 716–731. Springer Verlag Berlin-Heidelberg, NOV 2005.

[Wan06] Taowei David Wang. Gauging ontologies and schemas by numbers. In Vrandečić et al. [VdCSFGS06].