Checking Class Labels against Naming Conventions: First experience with the OntoCheck Protégé plugin

Daniel Schober[1*], Vojtěch Svátek[2], Martin Boeker[1]

[1]Institute of Medical Biometry and Medical Informatics (IMBI), University Medical Center, 79104 Freiburg, Germany
[2]University of Economics, Prague, Nám. W. Churchilla 4, 130 67 Praha 3, Czech Republic

ABSTRACT

Background: Although ontology naming conventions have been proposed by policy makers, the lack of tool support for testing and enforcing naming practices has hindered widespread compliance. We have developed OntoCheck, a Protégé plugin, which allows testing labels in an ontology on naming inconsistencies.

Objective: We report on initial experience in applying the tool in different settings and show that OntoCheck contributes to quality assurance in a test-set of ontologies.

Methods: We apply OntoCheck in four different ontology engineering efforts and test a variety of different ontologies on prevalence of naming issues. For each, we analyze the percentages of class names and labels violating outlined conventions and correlate the check types to the set of OBO Foundry naming conventions.

Results: Application of OntoCheck revealed that heterogeneity in class labels is still a common feature, even in release versions of ontologies, and that many of these could be detected and rectified by tool support. Nearly half of the OBO Foundry naming conventions could be assisted by OntoCheck, the remaining fraction relying on more complicated parsing and availability of lexica. Besides requirements drawn from naming conventions themselves, mismatches in string-based ontology alignment algorithms are identified as sanity check on the impact of labelling consistency. Analysing the prevalence of false positive and negative ontology alignment mismatches could prove valuable in deriving new naming conventions and test their effects in cross ontology harmonization efforts.

Conclusion: Our results show that typographical and syntactical labelling heterogeneity can be improved by tool support. The application of OntoCheck supports the verification of naming conventions and will ultimately ease string based ontology alignment.

1 INTRODUCTION

Although term labeling guidelines have recently made it into the '*Ten Commandments of Ontological Engineering*' (Jansen & Schulz, 2011), and years after the introduction of ontology class naming conventions (NC) (Schober *et al*., 2009) by the OBO Foundry (Smith *et al*. 2009), typographic and lexical variance still persists to be a potential source for heterogeneity in and between ontologies. But consistent naming is not a mere aesthetic requirement, as it has been shown to

- increase introspection of the intended meaning at data annotation time,
- increase readability within ontology class hierarchies,
- foster communication in collaborations with external projects to ensure effective maintenance of modularity and orthogonality, and
- avoid errors and increase precision and recall in automatic ontology matching and alignment algorithms that rely on lexical/string-based similarity measures.

To promote the application and verification of naming conventions, we complemented the Protégé 4 Editor[1] with tool support, the OntoCheck plugin[2], extending its curation abilities to help cleaning up an ontology with regard to labeling inconsistencies. Besides metadata completeness checks, its main capabilities target the comparison of class names and labels against self-defined or stored typographical and lexical naming patterns. Detected violations can be corrected to foster consistency in entity naming within an artifact or between import-dependency structures.

Within this paper, we summarize first experiences in applying OntoCheck in a variety of practical use cases and different ontology engineering efforts. OntoChecks functionalities are compared to the requirements of the OBO Foundry set of naming conventions. We provide an outlook on future strategies to justify naming conventions and verify requirements for tool support.

Our main intention is to report initial findings, testing the tool on a variety of OWL ontologies and briefly reporting on the prevalence of labeling issues and naming convention violations found in the tested ontologies, as well as discuss potential future tool enhancements.

2 METHODS

2.1 Requirement Collection: Ontology Matching

In order to draw real-life examples of synonym variance across ontologies, we surveyed string-based alignment mismatches found in the Ontology Alignment Evaluation Initiative (OAEI)[3]. Of the 18 matching algorithms, we

* To whom correspondence should be addressed: schober@imbi.uni-freiburg.de

[1] The Protégé Ontology Editor and Knowledge Acquisition System: http://protege.stanford.edu/, last accessed 20.01.2012

[2] The OntoCheck Plugin: http://www.imbi.uni-freiburg.de/ontology/OntoCheck/, last accessed 20.01.2012

[3] Ontology Alignment Evaluation Initiative - OAEI-2011 Campaign: http://oaei.ontologymatching.org/2011/, last accessed 20.01.2012

choose the three best, namely AgreementMaker (Cruz *et al.*, 2009), LogMap (Jiminez-Ruiz & Cuenca, 2011) and CODI (Noessner & Niepert, 2010), and looked at the exploited labels and the labeling problems that lead to mapping mismatches (false positives) or undetected matches (false negatives). In addition the algorithm developers were asked via email to report on string mismatch examples. For each of these, we investigated if, and which naming conventions would have helped avoiding those mismatches and whether these could have been detected and curated via OntoCheck.

2.2 Checked Ontologies

Six ontologies were selected to be checked for labeling issues via OntoCheck. Each author tested two ontologies from different engineering efforts, namely the DebugIT project[4], BioTop[5], GoodOD[6], Aneurist[7] and PatOMat[8]. The projects cover a wide thematic scope, i.e. from the biomedical domain over the educational domain up to the business domain. Inclusion criteria for the ontologies were that they had more than forty classes, were freely accessible in OWL and covered a wide range of domains and modeling-background philosophies. We here briefly describe the ontologies and their attitude to the naming conventions.

Biotop[9]: This biomedical upper level ontology follows the OBO Foundry conventions, but uses semantic, instead of numeric IDs.

DCO (Schober *et al.*, 2010): This large ontology serves the semantic interoperability platform for the DebugIT project on antibiotics resistance prevention. It adheres to the OBO Foundry naming conventions amended with more detailed explicit naming conventions outlined in a design principle document. It uses semantic IDs instead of numeric ones.

NTDO[10]: This tropical disease and epidemiology ontology adheres to the OBO Foundry naming conventions, but uses semantic instead of numeric IDs.

GoodRelations (Hepp, 2008): A vocabulary for publishing product details and services on the web, suitable for search engines and mobile applications in the e-commerce context. GoodRelations is a small ontology (~40 classes), with sophisticated design, e.g. use of longer and shortcut relational paths impacting class naming.

Vehicle Sales Ontology[11]: A vocabulary with descriptors for cars, boats, bikes, and other vehicles, serving e-commerce as complement to the GoodRelations ontology when applied in the respective field.

Aneurist Ontology[12]: An ontology providing terminological services for an integrated IT infrastructure for the neurological research and clinical care of intracranial aneuryisms.

2.3 OntoCheck Application

The OntoCheck plugin[13] was applied to test and curate the selected ontologies within the Protégé 4.1 framework.

For each ontology, we created, stored and applied a different set of checks. These were either self-employed in alignment to the specific requirements of the particular artifact, or were taken from the respective design principle documentations. Absolute counts and the percentages of found classes violating the checks were measured. Found labeling inconsistencies were rectified directly or submitted to the respective curators for later amendment. The outcome of this analysis has been collated in Tab. 2-4. A more elaborated list and supplemental material can be found on our webpage[14].

2.4 Tested Naming Conventions

After checking the ontologies on conformance to their own respective naming practices, we investigated whether OntoCheck can help to enforce the 16 published naming conventions of the OBO Foundry (Schober *et al.*, 2009). This test was done by mapping the ontology-specific tests onto their respective equivalents within the Foundry. These served as a proxy to test each numbered convention, i.e. if OntoCheck could be used to detect violations of this convention type. We tracked the reason, where conventions could not be supported by the tool in its present state.

3 RESULTS

3.1 Mismatch Examples drawn from OAEI

Although the complete table of mismatch pairs drawn from the Ontology Alignment Evaluation Initiative (OAEI) can be found on our website, we here list a few examples, together with proposals for naming conventions expected to alleviate the mismatches (Tab.1). The naming conventions from the last column which can be tested with OntoCheck are described in Section 3.3.

Reference	Label Ontology A	Label Ontology B	Mismatch Reason	NC

[4] DebugIT: http://www.debugit.eu/, last accessed 20.01.2012

[5] BioTop A Top-Domain Ontology for the Life Sciences: http://www.imbi.uni-freiburg.de/ontology/biotop/, last accessed 20.01.2012

[6] The GoodOD Project, http://www.iph.uni-rostock.de/Good-Ontology-Design.902.0.html, last accessed 20.01.2012

[7] @neurist – Integrated Biomedical Informatics for the Manage-ment of Cerebral Aneurisms: http://www.imbi.uni-freiburg.de/aneurist/ontology/

[8] The PatOMat Project: http://patomat.vse.cz/index.html, last accessed 20.01.2012

[9] As above: BioTop http://www.imbi.uni-freiburg.de/ontology/biotop

[10] NTDO – Neglected Tropical Disease Ontology: http://www.cin.ufpe.br/~ntdo/, last accessed 20.01.2012

[11] Vehicle Sales Ontology: http://www.heppnetz.de/ontologies/vso/ns, last

[12] @neurist – Integrated Biomedical Informatics for the Manage-ment of Cerebral Aneurisms: http://www.imbi.uni-freiburg.de/aneurist/ontology/, last accessed 20.01.2012

[13] The OntoCheck Plugin: http://www.imbi.uni-freiburg.de/ontology/OntoCheck/, last accessed 20.01.2012

[14] As above

Bodenreider, 2005	MA:tendon	NCI:Tendon	One refers to bone, one to muscle tissue	1.2, 3.2
Bodenreider 2005	MA: cervical vertebra 1	NCI: C1 Vertebra	Unresolved acronym	3.4, 2.1
Svab, 2008	AssociatedChair	Chair	One is a person role, one is an object	1.2, 3.2
Cruz, 2009	MA: prostate gland smooth muscle	NCI:gallbladder smooth muscle tissue	Refer to different muscles	3.1
Cruz, 2009	FMA:Trapezoid	NCI:Trapezoid	First refers to bone, latter to tissue	1.2, 3.2
Jiminez-Ruiz, 2011	Review	Reviewer	First is paper type, second a person role	3.2

Table 1. Selected mismatches and suitable OBO Foundry naming conventions (NC) with potential for rectification and better alignment precision.

3.2 Evaluation on Ontology Checks

Overall 61 checks were carried out on 6 different ontologies (the result table is available on our website). 29 of 61 checks (47 %) were done on ontologies without imports, either because the ontologies were self-sufficient (Biotop, GoodRelations), or to avoid redundancy, i.e. on Biotop, which is normally imported into DCO. A test that needed to be carried out on the full import closure was the check on pre- and postfixes of certain classes, as the supernode needs to be selected for all, e.g. biotop:Role classes.

Most checks were carried out on rdf:ID and rdfs:label, but sometimes proprietary, or Dublin Core annotation properties were checked.

For only two checks, a specific entry node was selected in order to check for standard postfixes and keep the label explicit. These subtrees were biotop:ValueRegion and biotop:Role (Tab. 2) in our case, but could be expanded to test further subtrees for consistent postfix usage.

Tested Entity	**Entry Node**	**Check**	**Violations abs (%)**
<rdf:ID>	Thing	CamelCase	34 (8)
<rdfs:label>	Thing	SpaceDelimiter	7 (4)
<rdf:ID>	Role	RegExp,'Role' postfix	2 (3)
<rdf:ID>	ValueRegion	RegExp,'ValueRegion' postfix	167 (54)
<rdfs:label>	Thing	MinCard.=1	184 (12)
<rdf:ID>,<rdfs:label>	Thing	NameEqualsLabel	304 (21)
<ru-meta:synonym>	Thing	MinCard.>2	238 (40)
<ru-meta:shortLabel>	Thing	MaxCharCount <20	3 (.5)

Table 2. Extract of launched checks on DCO, illustrating OntoCheck's capabilities and showing the amount of detected violations.

3.3 Correlating OntoCheck with OBO Foundry Naming Conventions

Here we list preliminary findings in correlating OntoCheck's capabilities in verifying OBO Foundry naming conventions (see Tab. 3 and 4). Of the 16 conventions published in (Schober *et al.*, 2009), seven could be checked with our plugin, so nearly half of the Foundry conventions were supported by our tool. For each naming convention, we here list aspects served by the OntoCheck tool (original list numbering skipped where OntoCheck is not applicable):

1.1 Use explicit and concise names: Apply RegExp check for stopword detection, apply name length checks, i.e. labels shorter than three characters are an important source of mismatches in alignment algorithms (Burgun & Bodenreider, 2005).

1.2 Use context independent names: Apply RegExp check on explicit pre-, in-, or postfixes. E.g. all 'ValueRegion' subclasses should contain either the postfix 'ValueRegion' or 'Region', testing for the RegExp: .*ValueRegion|.*Region

1.3 Avoid taboo words: Apply RegExp check to warn on 'metalevel' postfixes like 'class', 'type', 'concept', and 'entity'.

2.2 Avoid conjunctions: Apply RegExp to warn on logical connectives like Boolean operators 'and', 'or'. E.g. Biotop had CarbohydrateMoleculeOrResidue and OligoOrPolymer.

2.4 Use positive names: Apply RegExp check for lexical indicators of negations, e.g. checking 'non', 'anti 'or 'dis'.

3.3 Use space as word separator: Apply word delimiter checks.

3.4 Expand abbreviations and acronyms: Apply RegExp check like '\.'. Also a CaseConventionTest on all upper case can detect acronyms.

4.1 Prefer lower case beginnings: Apply word case check, e.g. CamelCase for IDs and all lower case for labels.

The remaining OBO Foundry conventions, which the tool was not explicitly able to check for were: 1.4 Avoid encoding administrative metadata in names, 2.1 Use univocous names and avoid homonyms, 2.3 Prefer singular nominal form, 2.5 Avoid catch-all terms, 3.1 Recycle strings, 3.2 Use genus-differentia style names, 3.5 Expand special symbols to words, 4.2 Avoid character formatting.

The above would need a more thorough lexical analysis, requiring a lexicon, or synonym exploitation, which is not yet implemented in this version of OntoCheck. Checks on these conventions would also require the comparison of lexical parts between different classes.

Ontology	#Checks	NC Checks	%	NC (times)
DCO	11	9	81	1.2 (4x), 2.1, 2.2, 3.1, 4.1, 4.2
NTDO	2	1	50	1.3
Biotop	13	4	30.7	1.3 (2x), 4.1, 3.1
Aneurist	13	5	38.4	1.3 (2x), 3.1, 4.1
GoodRel	11	4	36.3	2.2 (2x), 2.4, 4.1
Vehicles	10	6	60	2.2 (3x), 4.1

Table 3. Overall checks done on test ontologies and the amount of checks that could be associated with a Foundry naming convention. Checks for metadata completeness, i.e. on label presence, were not counted as '1.2 use context independent names' here.

OntoCheck Function	#Checks	%	NC(times)
CaseConventionTest	5	8.1	4.1(5x)
CompareValuesBetweenCls	1	1.6	2.1
CompareValuesForSingleCls	5	8.1	3.1(3x)
WordDelimiterCheck	5	8.1	3.3(5x)
RegExp, infix	13	21.3	2.2(7x), 1.3(5x), 2.4
RegExp, postfix	2	3.2	1.2(2x)
RegExp, length	3	4.9	1.2(3x)

Table 4. Applied OntoCheck functions, mapped onto particular enforceable naming conventions.

4 DISCUSSION

Looking at the results of the checks shown in Tab 2-4, we can summarize that the plugin was useful in detecting labeling errors in practical application scenarios. Although a considerable amount of the OBO Foundry naming conventions could be tested with the help of OntoCheck, a significant fraction could not yet be supported as neither the ontological structure (subsumption hierarchy or relations), nor lexical background knowledge (e.g. synonym lexica) are used at the moment. In particular conventions 1.4, 2.1, 2.5 could be served by simple inclusion of predefined lists of terms to be checked for appearance in labels. Conventions number 2.1, 2.3, 3.1, 3.2 rely on deeper structural comparison of labels between classes, whereas 3.5 and 4.2 could be implemented by applying standard transliteration lists, mapping special characters onto expanded UTF codes.

Bad class naming has been identified as potential source for mismatches in lexical ontology alignment approaches (Euzenat *et al.,* 2004). The reason is that alignment platforms such as AgreementMaker (Cruz *et al.,* 2009) and PROMPT (Noy & Musen, 2001) use string distance metrics to discover semantic mappings between ontology classes (Shvaiko & Euzenat, 2008).

Within the Ontology Alignment Evaluation Initiative - OAEI-2011 (Euzenat *et al.,* 2011), only less than half of the tools generated acceptable results trying to match classes from the anatomy domain[15]. As lexical string mapping is the most relevant technique in these alignment approaches (Massmann *et al.,* 2011), applying naming conventions and enforcing them via OntoCheck should increase precision and recall of string based matching algorithms (see Tab. 1). It would be interesting to investigate which of the string based alignment methods compared in the OAEI effort would profit most from a particular convention, also with regard to increased matching time. To test the effect of enforced naming conventions on the ease of alignment, alignment coherence and velocity, an ontology should be compared for its alignment precision before vs. after OntoCheck application.

Ideally the OntoCheck plugin would make use of the LiLA framework for the linguistic analysis of entity labels in ontologies[16], which provides an interface to various natural language processing tools and resources. The LiLA API (Ritze *et al.,* 2010) is still in early development, but it would be interesting to use it to expand OntoChecks 'lexical awareness', as was demanded by the alignment community earlier (Jimeno-Yepes *et al.,* 2009). Leveraging on lexical background knowledge, such as parsers and part of speech tagging, would not only allow for a much greater percentage of OBO Foundry naming conventions to be checked (around 70 percent), but recommendations for better labels, as well as structural modifications could be issued. The conventions profiting most from LiLa integration would be 2.1, 2.3, 3.1, 3.2.

Work on lexically induced cross-product generation in the gene ontology project (Mungall *et al.*, 2004) as well as classic ontology inference from text (Buitelaar *et al*., 2004, Svab-Zamazal & Svatek, 2008) illustrated that compositional analysis of terms can contribute to directly infer structural patterns and make suggestions for the use of naming patterns. In (Stevens *et al.,* 2003), the authors show, how labels can be exploited to infer missing subsumptions, i.e. a 'heparin biosynthesis' is-a 'glycosaminoglycan biosynthesis', as 'heparin' is-a 'glycosaminoglycan'. Such inferences could only be drawn by more thorough lexical analysis given naming conventions are applied consistently. Then, by exploiting re-occurring strings among sibling classes, a 'morpheme-frequency analyzer' could infer, check or correlate subclass labels to a parent class affix form. E.g. if in a subtree variances like X-itis, X-inflammation and inflammation-of-X occur, a tool could issue suggestions for harmonization, i.e. by suggesting the morpheme with the highest usage frequency or the morpheme used in the common superclass.

[15] Results for OAEI 2011: http://oaei.ontologymatching.org/2011/results/anatomy/index.html

[16] LiLA (Linguistic Label Analysis) framework for the linguistic analysis of phrases that can occur as class or property labels in ontologies: http://code.google.com/p/lila-project/, last accessed 20.01.2012

Generally, a complete pre-release check specification with a report could be generated, e.g. checking a complete set of conventions from a policy provider like the OBO Foundry. At the moment this is hindered by the fact that in most cases naming conventions are not outlined formally. To this end, we have recently joined forces with the ontology design pattern community[17] in order to formalize traceable naming patterns. Formalizing e.g. the Foundry naming conventions, and making them available under OntologyDesignPatterns.org would then allow a user to select a complete set of conventions, e.g. complying with the Foundry or other suitable policy makers.

5 CONCLUSION

Our OntoCheck-facilitated analysis of class labels in a range of ontologies of different size and scope has led to the detection of typographically heterogeneous, unclear, unintuitive and misleading labels. It has been shown that a considerable amount of labels violating either a groups proprietary own (intra-ontology) labeling policies, or universal naming conventions outlined by policy makers could be detected and rectified with the new Protégé OntoCheck plugin. These results have led to the plan to carry out a more thorough analysis on labeling issues which will be based on requirements rooted particularly in ontology alignment needs.
We hope widespread usage of our plugin will contribute to making ontology class hierarchies look cleaner and render artifacts more informative and robust when subjected to ontology matching and alignment approaches that leverage on string similarities of class names. Ultimately, we hope this Protégé extension will ease lexical post-processing of annotated data and hence increase overall secondary data usage by humans and computers alike.

ACKNOWLEDGEMENTS

This work was supported by the Deutsche Forschungsgemeinschaft (DFG) grant JA 1904/2-1, SCHU 2515/1-1 GoodOD (Good Ontology Design). Vojtěch Svátek is supported by the CSF under P202/10/1825 (PatOMat). We thank Ilinca Tudose for the implementation of OntoCheck.

REFERENCES

Bodenreider, O, Hayamizu, TF, et al.: (2005). Of mice and men: Aligning mouse and human anatomies. *AMIA Annu Symp Proc*. pp. 61–65, http://www.ncbi.nlm.nih.gov/pmc/articles/PM

Buitelaar P, Olejnik D, Sintek M (2004). A protégé plug-in for ontology extraction from text based on linguistic analysis. *Proceedings of the 1st European Semantic Web Symposium* (ESWS).

Burgun A and Bodenreider O (2005). An ontology of chemicals helps identify dependence relations among Gene Ontology terms, In: First Symposium on Semantic Mining in Biomedicine, SMBM 2005, EBI, Hinxton, UK, *CEUR vol.* **148**

Cruz IF, Antonelli FP, and Stroe C (2009). AgreementMaker: Efficient Matching for Large Real-World Schemas and Ontologies. *PVLDB* **2**(2):1586–1589

Euzenat, J, et al. (2011). Ontology Alignment Evaluation Initiative: six years of experience. IN: *J Data Semantics XV, Lecture Notes in Computer Science*, vol. **6720**, *158-192*

Euzenat, J, et al. (2004). State of the art on ontology alignment. *Deliverable* 2.2.3

Hepp M: GoodRelations (2008). An Ontology for Describing Products and Services Offers on the Web. In: *EKAW '08, Proceedings of the 16th international conference on Knowledge Engineering: Practice and Patterns*. Springer, LNCS **5268**, 329-346.

Jansen L, Schulz S (2011). The Ten Commandments of Ontological Engineering, *Proceedings of the 3rd Workshop of Ontologies in Biomedicine and Life Sciences (OBML)*, Berlin, 06.-07.10. 2011

Jimenez-Ruiz, E and Cuenca Grau, B (2011). LogMap: Logic-based and Scalable Ontology Matching. In: L.A. (ed.) *The 10th International Semantic Web Conference (ISWC). LNCS*, vol.**7031**, pp. 273–288. Springer

Jimeno-Yepes A, Jimenez-Ruiz E, Berlanga R, Rebholz-Schuhmann D (2010). Reuse of terminological resources for efficient ontological engineering in Life Sciences. *BMC Bioinformatics*.**10**(Suppl 10):S4. doi: 10.1186/1471-2105-10-S10-S4.

Massmann S, Raunich S, Aumüller D, Arnold P, Rahm E (2011). Evolution of the COMA Match System, http://disi.unitn.it/~p2p/OM-2011/om2011_Tpaper5.pdf

Mungall CM et al. (2004). Obol: Integrating Language and Meaning in Bio-Ontologies. *Comparative and Functional Genomics*, **5**:509-520.

Noessner, J and Niepert, M (2010). CODI: Combinatorial Optimization for Data Integration–Results for OAEI 2010. In: *Proceedings of the 5th Workshop on Ontology Matching*. Ontology Matching, page 142

Noy, N and Musen, M (2001). Anchor-prompt: Using non-local context for semantic matching.In: *Proc. IJCAI 2001* workshop on ontology and information sharing, Seattle (WA US). 63–70

Ritze D, Völker J, Meilicke C, Svab-Zamazal O (2010). Linguistic Analysis for Complex Ontology Matching. *Proceedings of the ISWC workshop on Ontology Matching* (OM)

Schober D et. al. (2010). The DebugIT core ontology: semantic integration of antibiotics resistance patterns. *Stud Health Technol Inform*.;160 (Pt 2):1060-4.

Schober D, Smith B, Lewis S, *et al*. (2009). Survey-based naming conventions for use in OBO Foundry ontology development. *BMC Bioinformatics*, Vol.**10**, Issue 1, http://www.biomedcentral.com/content/pdf/1471-2105-10-125.pdf

Shvaiko, P and Euzenat, J (2008). Ten challenges for ontology matching. In: *On the Move to Meaningful Internet Systems (OTM Conferences)*

Smith B, Ashburner M, Rosse C et al. (2007). The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. *Nat Biotechnol.* **25**(11):1251–1255. doi: 10.1038/nbt1346.

Stevens R, Wroe C, Bechhofer S, Lord P, Rector A, Goble C (2003). Building Ontologies in DAML + OIL, *Comp Funct Genomics*. February; **4**(1): 133–141.

Svab-Zamazal O, Svatek V (2008). Analysing Ontological Structures through Name Pattern Tracking, *EKAW* 2008, http://www.springerlink.com/content/j808w138253683m4/fulltext.pdf

[17] Ontology Design Patterns: http://ontologydesignpatterns.org/wiki/Main_Page, last accessed 20.01.2012