A Catalogue of OWL Ontology AntiPatterns

Catherine Roussey
Université de Lyon CNRS
Université Lyon 1, LIRIS
UMR5205
Villeurbanne, France
catherine.roussey@liris.cnrs.fr

Oscar Corcho
Ontology Engineering Group.
Departamento de Inteligencia
Artificial. Universidad Politécnica de Madrid Spain
ocorcho@fi.upm.es

Luis Manuel Vilches Blázquez
Ontology Engineering Group. Departamento de Inteligencia Artificial. Universidad Politécnica de Madrid, Spain
lmvilches@ fi.upm.es

ABSTRACT

Debugging inconsistent OWL ontologies is a time-consuming task. Debugging services included in existing ontology engineering tools are still far from providing adequate support to ontology developers and domain experts for this task, due to their lack of efficiency or precision when explaining the main causes for inconsistencies. We present a catalogue of common antipatterns found in inconsistent ontologies that can be used in combination with these tools to make this task more effective.

Categories and Subject Descriptors
I.2.4 Knowledge Representation Formalisms and Methods – *representation languages*

General Terms
Design, Languages, Verification

Keywords
OWL, ontology, debugging, antipattern

1. INTRODUCTION

Several tools exist for debugging OWL ontologies ([1,2]). These tools aim at isolating inconsistency-leading axioms, finding the roots of inconsistencies, which are then propagated throughout the concept hierarchy. Although useful, they are still far from optimal in providing adequate explanations about the reasons for inconsistencies and in proposing alternatives to resolve them. Besides, in complex cases the generation of inconsistency explanations takes several hours, what makes these tools hard to use. As a result, we found out that domain experts usually change axioms from the original ontology in a somehow random manner, even changing the intended meaning of the real definitions instead of correcting errors in their formalisations.

We made an effort to understand common inconsistency-leading patterns used by domain experts when implementing OWL ontologies, based on existing ontology design patterns and knowledge patterns and anti-patterns.

2. PATTERNS AND ANTI-PATTERNS

In contrast to ontology design patterns, the work on anti-patterns is less detailed ([3, 4]). Four LAPs are presented in [3], focused on property domains and ranges. And [4] describes common difficulties for DL newcomers in understanding the logical meaning of expressions. However, none groups anti-patterns in a common classification.

2.1 A Classification of Ontology Design Anti-Patterns

We have identified a set of patterns that are commonly used by domain experts in their DL formalisations and OWL implementations, and that normally result in inconsistencies. We have categorized them into three groups:

- Logical Anti-Patterns (LAP). They represent errors that DL reasoners detect.
- Non-Logical (aka Cognitive) Anti-patterns (NLAP). They represent possible modelling errors that are not detected by reasoners (they are not logical but modelling errors, which may be due to a misunderstanding of the logical consequences of the used expression).
- Guidelines (G). They represent complex expressions used in definitions that are logically correct, but in which the ontology developer could have used other simpler alternatives for encoding the same knowledge.

2.2 Logical Antipatterns

AntiPattern AndIsOr (AIO)

$C1 \sqsubseteq \exists R.(C2 \sqcap C3)$, disj(C2,C3)[1]

This is a common modelling error that appears due to the fact that in common linguistic usage, "and" and "or" do not correspond consistently to logical conjunction and disjunction respectively [12].

AntiPattern OnlynessIsLoneliness (OIL)

$C1 \sqsubseteq \forall R.C2$, $C1 \sqsubseteq \forall R.C3$, disj(C2,C3)

The ontology developer has created a universal restriction to say that C1 can only be linked with R role to C2. Next, a new universal restriction is added saying that C1 can only be linked with R to C3, disjoint with C2. In general, this means that the ontologist forgot the previous axiom.

[1] This does not mean that the ontology developer has explicitly expressed that C2 and C3 are disjoint, but that these two concepts are determined as disjoint from each other by a reasoner. We use this notation as a shorthand for $C2 \sqcap C3 \sqsubseteq \bot$.

K-CAP'09, September 1–4, 2009, Redondo Beach, California, USA.
ACM 978-1-60558-658-8/09/09.

AntiPattern UniversalExistence (UE)

$C1 \sqsubseteq \forall R.C2$, $C1 \sqsubseteq \exists R.C3$, disj(C2,C3)

The ontology developer has added an existential restriction for a concept without remembering the existence of an inconsistency-leading universal restriction for that concept.

AntiPattern EquivalenceIsDifference (EID)

$C1 \equiv C2$, disj(C1,C2)

This inconsistency comes from the fact that the ontology developer wants to say that C1 is a subclass of C2 (that is, that C1 is a C2, but at the same time it is different from C2 since he has more information). This anti-pattern is only common for ontology developers with no previous training in OWL modelling, since after a short training session they would discover that they really want to express $C1 \sqsubseteq C2$.

2.3 Non Logical Anti-Patterns

AntiPattern SynonymeOfEquivalence (SOE)

$C1 \equiv C2$

The ontology developer wants to express that two classes C1 and C2 are identical. This is not very useful in a single ontology that does not import others. Indeed, what the ontology developer generally wants to represent is a terminological synonymy relation: the class C1 has two labels: C1 and C2. Usually one of the classes is not used anywhere else in the axioms defined in the ontology.

AntiPattern SumOfSome (SOS)

$C1 \sqsubseteq \exists R.C2$, $C1 \sqsubseteq \exists R.C3$, disj(C2,C3)

The ontologist has added a new existential restriction without remembering that he has already defined another existential restriction for the same concept and role. Although this could be ok in some cases (e.g., a child has at least one mother and at least one father), in many cases it represents a modelling error.

AntiPattern SomeMeansAtLeastOne (SMALO)

$C1 \sqsubseteq \exists R.C2$, $C1 \sqsubseteq (\geq 1\ R.T)$

The cardinality restriction is superfluous.

2.4 Guidelines

Guideline DisjointnessOfComplement (DOC)

$C1 \equiv$ not C2

The ontology developer wants to say that C1 and C2 cannot share instances. Even if the axiom is correct from a logical point of view, it is more appropriate to state that C1 and C2 are disjoint.

Guideline Domain&CardinalityConstraints (DCC)

$C1 \sqsubseteq \exists R.C2$, $C1 \sqsubseteq (=2R.T)$

Ontology developers with little background in formal logic find difficult to understand that "only" does not imply "some" [12]. This antipattern is a counterpart of that fact. Developers may forget that existential restrictions contain a cardinality constraint: $C1 \sqsubseteq \exists R.C2 \models C1 \sqsubseteq (\geq 1R.C2)$. Thus, when they combine existential and cardinality restrictions, they may be actually thinking about universal restrictions with those cardinality constraints.

Guideline GroupAxioms (GA)

$C1 \sqsubseteq \forall R.C2$, $C1 \sqsubseteq (\geq 2R.T)$ (just as an example)

In order to facilitate the comprehension of complex class definitions, we recommend grouping all the restrictions of a class that use the same role R in a single restriction.

Guideline MinIsZero (MIZ)

$C1 \sqsubseteq (\geq 0R.T)$

The ontology developer wants to say that C1 can be the domain of the R role. This restriction has no impact on the logical model being defined and can be removed.

CONCLUSIONS

In this paper, we collect a list of common anti-patterns that can be found in ontologies and that cause a large percentage of inconsistency problems. Besides, we list some anti-patterns that do not have an impact on the logical consequences of the ontology being developed, but are important to reduce the number of errors in the intended meaning of ontologies or to improve their understandability.

ACKNOWLEDGMENTS

This work was partially funded by the Spanish project GeoBuddies and by the COST Action C21, sponsored by the EC under the grant number STSM-C21-04241.

REFERENCES

[1]. Horridge M, Parsia B, Sattler U. "Laconic and Precise Justifications in OWL". In Proceedings of the *7th International Semantic Web Conference* (ISWC), Karlsruhe, Germany; LNCS 5318: 323-338. (2008).

[2]. Kalyanpur A, Parsia B, Sirin E, Cuenca-Grau B. "Repairing Unsatisfiable Classes in OWL Ontologies". In Proceedings of the *3rd European Semantic Web Conference* (ESWC), Budva, Montenegro; LNCS 4011: 170-184 (2006).

[3] Collection of antipatterns available at http://wiki.loa-cnr.it/index.php/LoaWiki:MixedDomains

[4] Rector AL, Drummond N, Horridge M, Rogers L, Knublauch H, Stevens R, Wang H, Wroe C. "OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns*"*. In Proceedings of the 14th *International Conference Knowledge Acquisition, Modeling and Management* (EKAW), Whittlebury Hall, UK. LNCS 3257: 63-81 (2004).