COST Technical Committee "Transport and Urban Development"

COST Action C21
"Urban Ontologies for an improved communication in urban civil engineering projects" - TOWNTOLOGY Project
http://www.towntology.net/

Technical report n°7
Short Term Scientific Mission Report – Formal Ontologies for Database Interoperability

Catherine ROUSSEY (Université de Lyon, CNRS, LIRIS UMR 5205, Université Claude Bernard Lyon 1)

Version: 1
Preparation date: April 2009

1 Introduction

The Short Term Scientific Mission for COST Action C21 "Urban Ontologies for an improved communication in UCE projects – TOWNTOLOGY" took place from 10th January to 10th April 2009 in Spain. The Host Institution was the lab Ontology Engineering Group in the Facultad de Informática of the Universidad Politécnica de Madrid, Campus de Montegancedo, Boadilla del Monte, Spain.

This scientific mission has been already useful. On the theoretical level, I have improved my knowledge about formal ontology, spatial and time reasoning and data integration. On the operational level, I have help to debug a formal ontology in hydrography using the Protégé OWL tool and more particularly the explanation workbench.

In the following report, the second section presents the aims of this mission. In the third section, I will summarize my notes about the courses I have followed on Description Logics and spatial and temporal reasoning. The section four presents a small state of the art about semantic data integration; Followed by a description of the work carried out for OWL ontology debugging and its results and perspectives. The last sections present the future collaboration between the two partners: OEG and LIRIS and the host institution report.

2 Aim of the STSM

Our proposal for this STSM was concerning the development of formal ontologies dealing with geographical objects. This kind of ontologies is used in data integration which is the evolution of database interoperability. Thus the goals of this mission are:

- To understand how spatial or temporal knowledge are used for reasoning purpose.
- To learn how to build a formal ontology with dedicated tool, like neon tool suite or Protégé OWL.
- To make a state of the art in ontology for data integration.
- To evaluate some of this technique for interoperability in spatial database.

During this STSM, I had study several courses of Oscar Corcho: one about Description Logics (Razonamiento con taxonomías) and another about Temporal and Spatial Reasoning (Razonamiento espacial y temporal). I had work on the development of a formal ontology in hydrographic field called HydrOntology. This ontology has been build using Protégé OWL tool [Protégé]. Before the beginning of my STSM a reasoner applied on this ontology will classify several concepts as incoherent. So one of my goals was to correct them using the new Protégé Explanation workbench. I had also started a small state of the art about how formal ontologies are used in database interoperability.

3 Formal Ontology and Spatial reasoning

I have followed the courses of Oscar Corcho about knowledge representation formalism and Description Logics. These courses are required to understand the courses about spatial and temporal reasoning. During the courses we have been trainee on Protégé OWL ontology editor tool.

3.1 Description Logics

Description logics (DLs) are a family of knowledge representation languages which can be used to represent the concept definitions of an application domain.

DL is an extension to frames and semantic networks with formal logic-based semantics: DL expression can be translated into first-order logic. DL has become a cornerstone of the Semantic Web for its use in the design of ontologies. The OWL-DL and OWL-Lite sub-languages of the W3C-endorsed Web Ontology Language (OWL) are based on DL.
DLs describe domain knowledge in terms of concepts (classes), roles (relationships or property) and individuals. A concept denotes the set of individuals that belongs to it, and a role denotes a relationship between concepts. DLs are composed of the TBox, or terminological box, containing the class definitions and the ABox, or assertional box, containing the description of individuals.

DLs are specific languages characterised by the constructors and axioms used to assert knowledge about classes, roles and individuals.
The most common constructors in class definitions are:

- intersection or conjunction of concepts $C1 \cap C2$,
- union or disjunction of concepts $C1 \cup C2$,
- negation or complement of concepts $\neg C1$,
- universal restriction or value restriction $\forall R.C$,
- existential restriction $\exists R.C$,
- maximum cardinality $\leq nR$
- minimum cardinality $\geq nR$

The most common axioms in class, role and individual definitions are:

- subclass $C1 \subseteq C2$
- equivalence $C1 \equiv C2$
- disjointness $C1 \cap C2 \subseteq \perp$
- subproperty $R1 \subseteq R2$
- membership $I \in C1$

DL reasoner proposes several inference mechanisms:

1. *instance checking* to check if a particular instance is a member of a given concept
2. *relation checking* to check if a relation/role/property hold between two instances
3. *subsumption* to check if a concept is a subset of another concept
4. *Concept consistency* to check if there is no contradiction among the concept definitions.

Description Logics are based on two hypotheses that are not share by Database theory. DLs do not have **the Unique Name Assumption (UNA):** Thus two concepts with different names may be allowed by some inference to be equivalent. DLs do not have the **Closed World Assumption (CWA)**, but rather this type of language have the **Open World Assumption (OWA)**. CWA means that if proposition P cannot be proved True, then P is assumed to be False (and vice versa). CWA is used in Database theory: If a proposition P belong to a database, then P is true, otherwise P is false (the proposition P does not belong to the database, thus P is false). OWA means that if a proposition P cannot be proved True, Then P is not assumed to be false (and vice versa).

3.2 Spatial and temporal reasoning

Humans make decisions about space and time trough both quantitative and qualitative assertions.
Quantitative assertions use numbers to represent knowledge: for example quantitative data are the coordinate used to represent the position of an object B at a time point. Thus quantitative reasoning is based on number computation. Computation can find the distance between two objects or the velocity of an object or the time point where an object A will hit an object B.

Qualitative assertions describe a particular situation based on a finite vocabulary in which numerical quantities are avoided in favour of symbolic qualitative values. For example using the vocabulary "closed to" or "touch", you can assert that an object A is closed to an object B or an object A touch an object B. Thus a situation is described by a set of qualitative formulae.
Qualitative reasoning can be:

- Satisfiability: find a situation where all the formulae are satisfied. Thus all the instances are associated to a qualitative value.
- Model finding: finding the minimal description of a situation where all the formulae are satisfied.
- Deduction: discover if a qualitative relation exists between two instances.

Notice that qualitative reasoning can not be combined with quantitative one. No tool exists that combine these two types of reasoning. Thus if you need to do some quantitative computation you should do it first with the appropriate tools and after translate your data in a qualitative reasoning tools.

- Protégé enables to store some quantitative data properties but it does not allow declaring some computation function on these data.
- SparQ is a collection of tools for Qualitative Spatial Reasoning in applications released as free software under GNU GPL.

4 Data integration

Data integration is the process of combining data residing at different sources and providing the user with a unified view of these data [Maurizio Lenzerini 2002]. Thus the user will have a single query interface to search among a set of heterogeneous data sources.
Two trends of Data integration exists: ETL trend and mediation one

4.1 ETL trend

The first trend proposes to build a common database thanks to heterogeneous one (see Figure 1). Each data source has to be process by ETL (Extraction, Transformation, Loading) process in order to update the common database. When a data source is updated all the ETL process has to be executed again. It is not easy to build the common database if you have no access to the whole database.

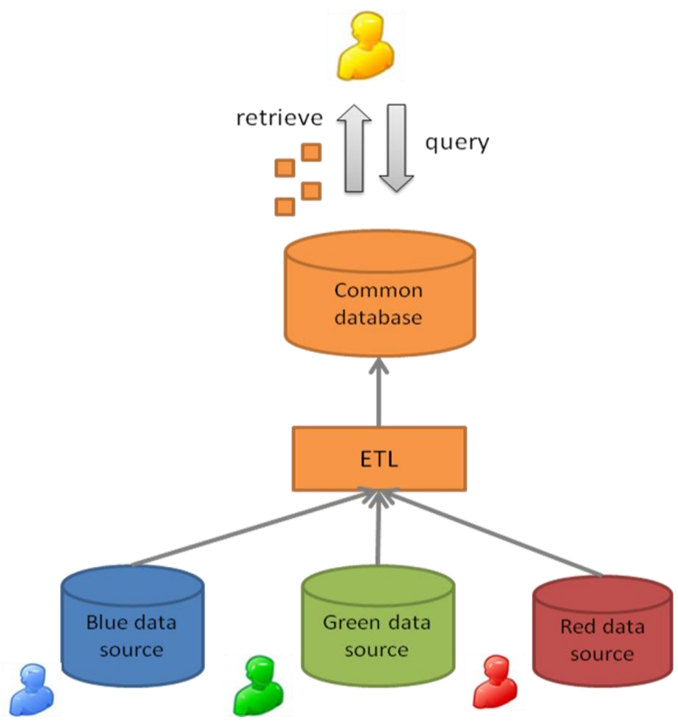

Figure 1: common database creation

4.2 Mediation trend

The second trend of data integration provides a uniform query interface over a mediated schema (see figure 2). The mediated schema is a virtual database. There exist three approaches of mediation data integration. All these approaches used several components:

- The global schema is the mediated schema which describes all the data accessible in all data sources.
- A local schema per data source, each of them describes the data stored in a single data source.
- A set of mappings between local schemas and global schema.
- A wrapper per data source, which is the system that provides a way to abstract the data from a data source and transform them in components of the global schema. Wrappers play the role of a translator between the local schema and the global one.
- The mediator is able to identify each different representations of the same data stored in a data source. Thus the mediator can query each local data source by using the associated wrapper and gather all the result. Mediator decides how to access each data sources and in which order, normally by making a query planning step.

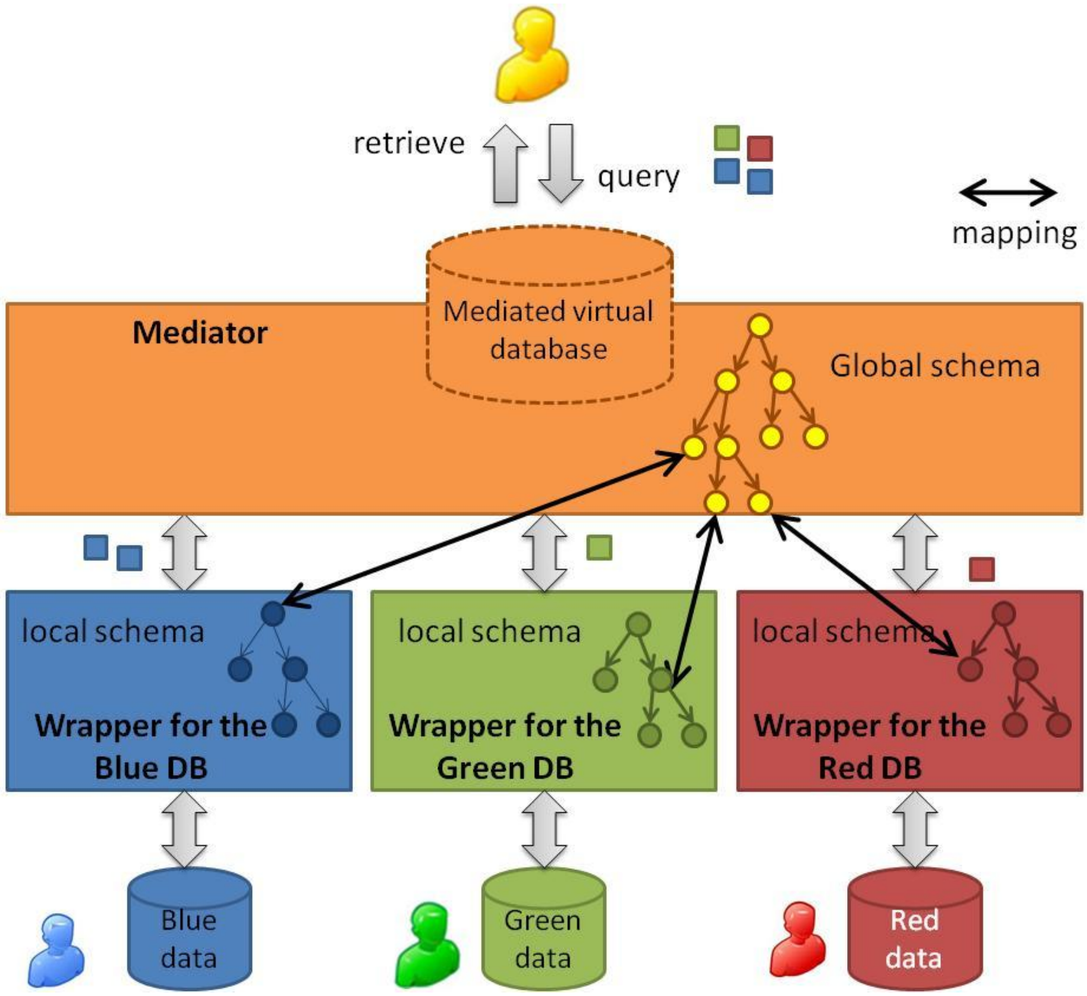

Figure 2: mediated schema virtual database.

GaV: Global as View

The mediated schema is designed to be a view over the sources. This approach called Global as View (GaV) — where "Global" refers to the global (mediated) schema — is often used due to the simplicity involved in answering queries issued over the global schema. A component of the global schema is associated to queries on local data sources. The mappings define a component of the global schema according to components of local ones. Thus a query on the global schema is transformed into specialized queries over the data sources. However, the obvious drawback is the need to modify the global schema and rewrite the view whenever a new source is to be integrated and/or an existing source changes its schema. As example of the GaV approach, we can cite:

- STIMMIS (Stanford-IBM Manager of Multiple Information Sources) project uses the Object Exchange Model (OEM) to describe the data source and express queries, [Chawathe & al, 1994].
- SIMS (Service and Information Management of decision System) project use the LOOM language, [Arens & al, 1996]. SIMS use a global ontology has a query model.

LaV: Local as View

The local data sources are considered to be a view over the (non existent) global (mediated) schema. This approach called Local as View (LaV) — where "Local" refers to the local data sources — is used due to the simplicity to manage update on the data sources without changing the global schema. In this approach the mappings define a component of a local schema according to components of global schema. However, the drawback is to reformulate a query in terms of global schema, using terms of local schemas. As example of the GaV approach we can cite:

- Information Manifold (IM) was the first system that is based on the LaV approach [Levy & al, 1996].
- Most parts of the semantic data integration system based on ontologies are classified as LaV approach. For a survey on this type of data integration system see [Wache & al, 2001]

Hybrid Approaches

These approaches want to combine the advantages and avoid the limitations of GaV and LaV approaches in order to propose a solution easy to maintain if data sources changed like in LaV approach and with scalable query formulation procedure like in GaV approach. We can cite:

- Global Local as View (GLaV) approach proposed in [Friedman & al, 1999] use a flexible language to combine predicates in the same first order logic sentence.
- The Both as View (BaV) approach proposed during the AutoMed project [McBrien and Poulovassilis, 2004].
- [Cali & al, 2002] propose a translation algorithm to turn LaV into GaV.

4.3 Related Projects

OBSERVER

OBSERVER system is an example of LaV approach using multiple ontologies. Each local data source is modelized by a preexisting ontology: For example, parts of Wordnet are used in [Mena & al, 1996]. Thus it requires creation of mappings between the multiple ontologies, called inter ontology mapping. In [Mena & al, 1996] Inter ontology mapping are based on synonymy relation between terms. Notice that even if a global schema does not exist in this system, at the end a common vocabulary is build, shared partially by local ontologies.

R20 and ODE Mapster

ODEMapster use the language R20 [Barrasa & al, 2004] to transform each database record to a concept instance of an ontology. They can produce very complex mapping: a mapping is a transformation procedure that could be applied on certain condition (table, record, properties, values, relation transformation). At the end the final user query the knowledge base associated to the ontology. This approach can be considered to apply ETL technique for ontology population.

5 Formal Ontology Debugging

During the STSM, I participate to the development of a formal ontology about hydrography called HydrOntology. I have to debug this OWL ontology with the help of domain experts in hydrography. Debugging of inconsistent OWL ontologies is normally a tedious and time-consuming task where a combination of ontology engineers and domain expert is often required to understand whether the changes to be performed in order to make the OWL ontology consistent are actually changing the intended meaning of the original knowledge model. This task is aided by existing ontology debugging systems, incorporated in existing reasoners and ontology engineering tools, which ameliorate this problem but in complex cases are still far from providing adequate support to ontology engineers, due to lack of efficiency or lack of precision in determining the main causes for inconsistencies.

During our debugging task, we found a set of anti-patterns commonly found in OWL ontologies, which can be useful in the task of ontology debugging in combination with those debugging tools.

5.1 HydrOntology

The Spanish National Geographic Institute (IGN-E) developed a common reference model by means of a domain ontology, called HydrOntology. IGN-E wants to build this ontology in order to facilitate the semantic harmonization of hydrographic information among data producers at different levels (national, regional and local).

The statistical data (metrics) and its different taxonomic relations appearing below provide an overview of the HydrOntology characteristics.

HydrOntology is saved in the OWL format; it has 150 classes, 34 object properties, 66 data properties and 256 axioms. Some examples of the four taxonomic relations defined in the Frame Ontology [Farquhar & al, 1997] and the OKBC Ontology [Chaudri & al, 1998], namely, Subclasses, Disjoint-Decomposition, Exhaustive-Decomposition and Partitions, have been implemented in the ontology. Further details are shown in [Vilches-Blázquez & al, 2007]. The ontology documentation is exhaustive, thus, definitions and their definition sources can be found in each concept (class). The ontology has an important amount of labels with alternative names (synonyms) as well as concept and synonym provenances.

A domain expert about geographical information was trained to build an ontology in Description Logic using Protégé tool (Protégé-OWL version 4). He built the ontology following METHONTOLOGY, a widely-used ontology building methodology. A detailed description of this methodology can be found in [Gómez-Pérez & al, 2003].

HydrOntology has been developed according to the ontology design principles proposed by [Gruber, 1995] and [Arpírez & al, 1998]. Some of its most important characteristics are that the concept names (classes) are sufficiently explanatory and rightly written. According to some naming conventions, each class is written with a capital letter at the beginning of each word, while object and data properties are written with lower case letters. At the end of the development process 102 concepts were classified as incoherent by the classifier.

When implementing this ontology in OWL several issues arose with respect to its consistency, given its complexity. In the first iteration of implementation, where the domain expert took the conceptualization following Methontology's intermediate representations and encoded it with Protégé 4, all the classes in the ontology were considered inconsistent. Then the process of refinement started, using the OWL ontology debugging facilities of Protégé. Indeed, the debugging systems used did not provide enough information about root unsatisfiable classes or adequate (e.g., understandable by domain experts) justifications of the reasons for their unsatisfiability. Thus, we made an effort to understand inconsistency-leading patterns used by domain experts when implementing OWL ontology. Moreover in several occasions during the debugging process the

generation of justifications for inconsistencies took several hours, what made these tools hard to use.
Ontology developer needs more recommendation for debugging than those provided by actual tools. We found out that in several occasions domain experts were just changing 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.
After several iterations, which resulted in a large number of changes to the original implementation, the final consistent ontology could be delivered.
In this paper we propose a detailed list of such anti-patterns, compiling all the relevant cases that we came across when helping ontology developers to debug their ontologies.

5.2 Antipatterns

We have identified a set of patterns that are commonly used by domain experts in their OWL implementations and that normally result in inconsistencies that may be easy or difficult to solve by them. This set of patterns is what we call anti-patterns, and we have categorized them in three groups:

- Logical Anti-Patterns (LAP). They represent errors that DL reasoners detect. These are the ones for which tool support is easier to provide and hence some support already exists.
- Non-Logical (aka Cognitive) Anti-patterns (NLAP). They represent possible modelling errors that are not detected by reasoners (they are not logical but model-ling errors, which may be due to a misunderstanding of the logical consequences of the used expression).
- Guidelines (G). They represent complex expressions used in an ontology component definition that are correct from a logical point of view, but in which the ontology developer could have used other simpler alternatives for encoding the same knowledge.

In the rest of this section we describe each of the anti-patterns identified in each group, providing their name and acronym, their template logical expressions and a brief explanation of why this anti-pattern can appear. As aforementioned, it is important to note that LAP are identified by existing ontology debugging tools, although the information that is provided back to the user explaining the reason for the inconsistency is not described according to such a pattern, what makes it difficult for ontology developers to find out where the inconsistencies are coming from. With respect to NLAP and G, they are not currently detected by these tools as such, although in some cases their combination may lead into inconsistencies that are detected (although not appropriately explained) by tools. We think that tool support for them could be a major step forward in this task.
Finally, all these anti-patterns should be seen as elementary units that cause ontology inconsistencies. That is, they can be combined into more complex ones.

5.3 Logical Anti-Patterns

AntiPattern AndIsOr (AIO)

$C1 \subseteq \exists R.C2 \cap C3)$, Disj(C2,C3)[1]

---

[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 $C2 \cap C3 \subseteq \bot$.

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 [10]. For example, I want a cake with milk and chocolate is ambiguous. Does the recipe of cake contain some chocolate plus some milk? (Cake_Recipe $\subseteq$ ($\exists$contain.Chocolate)$\cap$($\exists$contain.Milk). Does the recipe of cake contain chocolate-flavoured milk? (Cake_Recipe$\subseteq$ $\exists$contain.(Chocolate$\cap$Milk)). Does the recipe of cake contain some chocolate or some milk? (Cake_Recipe $\subseteq$ $\exists$contain.(Chocolate$\cup$Milk)). The domain expert makes a confusion between the linguistic "and" and the logical "or". Notice that the position of the logical "and" has an importance in the semantic of an axiom.

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$\subseteq$C2. This inconsistency can hide also a terminological synonymy relation between classes like in SOE.

AntiPattern OnlynessIsLoneliness (OIL)

C1$\subseteq\forall$R.C2, C1$\subseteq\forall$R.C3, Disj(C2,C3)
The ontology developer has created an universal restriction to say that C1 can only be linked with a 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 ontology developer forgot the previous axiom

AntiPattern OnlynessIsLonelinessWithInheritance (OILWI)

C1$\subseteq$ C2, C1$\subseteq\forall$R.C3, C2$\subseteq\forall$R.C4, Disj(C3,C4).
The ontology developer has added a universal restriction for class C1 without remembering that he had already defined another universal restriction with the same property in a parent class. This anti-pattern is a specialization of OIL.

AntiPattern OnlynessIsLonelinessWithPropertyInheritance (OILWPI)

R1$\subseteq$R2, C1$\subseteq\forall$R1.C2, C1$\subseteq\forall$R2.C3, Disj(C2,C3)
The ontology developer misunderstands the subproperty relation between roles, thinking that it is similar to a part-of relation. This anti-pattern is a specialization of OIL because C1$\subseteq\forall$R1.C2, R1$\subseteq$R2 $\models$ C1$\subseteq\forall$R2.C2

AntiPattern UniversalExistence (UE)

C1$\subseteq\forall$R.C2, C1$\subseteq\exists$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 UniversalExistenceWithInheritance1 (UEWI_1)

C1$\subseteq$C2, C1$\subseteq\exists$R.C3, C2$\subseteq\forall$R.C4, Disj(C3,C4)

The ontology developer has added an existential/universal restriction in a concept without remembering that there was already an inconsistency-leading universal/existential restriction in a parent class, respectively. This anti-pattern is a specialization of UE.

AntiPattern UniversalExistenceWithInheritance2 (UEWI_2)

$C1 \subseteq C2$, $C1 \subseteq \forall R.C3$, $C2 \subseteq \exists R.C4$, Disj(C3,C4)
Same reasons as UEWI_1.

AntiPattern UniversalExistenceWithPropertyInheritance (UEWPI)

$R1 \subseteq R2$, $C1 \subseteq \exists R1.C2$, $C1 \subseteq \forall R2.C3$, Disj(C2,C3)[2]
The ontology developer misunderstands the subproperty relation between roles, thinking that it is similar to a part-of relation. This anti-pattern is a specialization of UE because $C1 \subseteq \exists R1.C2$, $R1 \subseteq R2 \models C1 \subseteq \exists R2.C2$.

AntiPattern UniversalExistenceWithInverseProperty1 (UEWIP_1)

$C2 \subseteq \exists R^{-1}.C1$, $C1 \subseteq \forall R.C3$, Disj(C2,C3)
The ontology developer has added restrictions about C2 and C1 using a role and its inverse. This antipattern is a specialization of UE because: $C2 \subseteq \exists R^{-1}.C1 \models C1.1 \subseteq \exists R.C2$, $C1.1 \subseteq C1$

AntiPattern SumOfSomIsNeverEqualToOne (SOSINETO)

$C1 \subseteq \exists R.C2$, $C1 \subseteq \exists R.C3$, $C1 \subseteq \leq 1R.T$, Disj(C2,C3)
This anti-pattern can also be written like this
$C1 \subseteq \exists R.C2$, $C1 \subseteq \exists R.C3$, $C1 \subseteq =1R.T$, Disj(C2,C3)
The ontology developer has added a new existential restriction without remembering that he has already defined another existential and a cardinality restriction for the same concept and role. This pattern is not an elementary one because it contains the NLAP SOS and the G DCC (presented latter), none of these elementary antipattern cause inconsistency; nevertheless it is a good example that a combination of NLAP and G cause inconsistencies.

5.4 Non Logical Anti-Patterns

As aforementioned, these anti-patterns are not necessarily errors, but describe common templates that ontology developers use erroneously trying to represent a different piece of knowledge.

AntiPattern SynonymeOfEquivalence (SOE)

$C1 \equiv C2$
The ontology developer wants to express that two concepts C1 and C2 are identical. This is not useful at all in a single ontology. 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.

---

[2] Note that $R1 \subseteq R2$, $C1 \subseteq \forall R1.C2$, $C1 \subseteq \exists R2.C3$, Disj(C2,C3) is not an antipattern.

AntiPattern OnlynessIsLonelinessWithInverseProperty (OILWIP)

$C2 \sqsubseteq \forall R^{-1}.C1$, $C1 \sqsubseteq \forall R.C3$, Disj(C2,C3)

The ontology developer has created two universal restrictions using a role and its inverse. If this antipattern is associated to the axiom $C2 \sqsubseteq \geq 1 R^{-1}.C1$ then there is an incoherence, because: $C2 \sqsubseteq \forall R^{-1}.C1$, $C2 \sqsubseteq \geq 1 R^{-1}.C1 \models C2 \sqsubseteq \exists R^{-1}.C1 \models C1.1 \sqsubseteq C1$, $C1.1 \sqsubseteq \exists R.C2$. This combination of antipattern is a specialization of UE.

AntiPattern UniversalExistenceWithInverseProperty_2 (UEWIP_2)

$C2 \sqsubseteq \forall R^{-1}.C1$, $C1 \sqsubseteq \exists R.C3$, Disj(C2,C3)

The ontology developer has added restrictions about C2 and C1 using a role and its inverse. If this antipattern is associated to the axiom $C2 \sqsubseteq \geq 1 R^{-1}.C1$ then there is an incoherence, because: $C2 \sqsubseteq \forall R^{-1}.C1$, $C2 \sqsubseteq \geq 1 R^{-1}.C1 \models C2 \sqsubseteq \exists R^{-1}.C1 \models C1.1 \sqsubseteq C1$, $C1.1 \sqsubseteq \exists R.C2$. This combination of antipattern is a specialization of UE.

AntiPattern SumOfSom (SOS)

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

The ontology developer 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 SumOfSomWithInheritage (SOSWI)

$C1 \sqsubseteq C2$, $C1 \sqsubseteq \exists R.C3$, $C2 \sqsubseteq \exists R.C4$, Disj(C3,C4)

The ontology developer has added an existential restriction in a concept without remembering that he had already defined another existential restriction with the same role in a parent class. This Anti-Pattern is a specialization of SOS.

AntiPattern SumOfSomWithPropertyInheritance (SOSWPI)

$R1 \sqsubseteq R2$, $C1 \sqsubseteq \exists R1.C2$, $C1 \sqsubseteq \exists R2.C3$, Disj(C2,C3)

The ontology developer misunderstands the subproperty relation between roles, thinking that it is similar to a part-of relation. This Anti-Pattern is a specialization of SOS because $C1 \sqsubseteq \exists R1.C2$, $R1 \sqsubseteq R2 \models C1 \sqsubseteq \exists R2.C2$

AntiPattern SumOfSomWithInverseProperty (SOSWIP)

$C2 \sqsubseteq \exists R^{-1}.C1$, $C1 \sqsubseteq \exists R.C3$, Disj(C2,C3)

The ontology developer has created two existential restrictions using a role and its inverse. This anti-pattern specializes SOS because: $C2 \sqsubseteq \exists R^{-1}.C1 \models C1.1 \sqsubseteq C1$, $C1.1 \sqsubseteq \exists R.C2$.

AntiPattern SomeMeansAtLeastOne (SMALO)

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

The cardinality restriction is superfluous, because if there is an existential restriction that means that the cardinality restriction using the same role is at least equal to 1. The ontology developer had created the axiom $C1 \sqsubseteq \geq 1 R.T$ first, to say that C1 should be defined by the R role. Next, he specialized his definition and forgot to remove the first restriction.

5.5 Guidelines

As aforementioned, guidelines represent complex expressions used in an ontology component definition that are correct from a logical point of view, but in which the ontology developer could have used other simpler alternatives for encoding the same knowledge.

Guideline DisjointnessOfComplement (DOC)

$C1 \equiv$ not $C2$

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

Guideline Domain&CardinalityConstraints (DCC)

$C1 \subseteq \exists R.C2$, $C1 \subseteq (\geq 2R.T)$ (for example)

Ontology developers with little background in formal logic find difficult to understand that universal restriction does not imply existential one [10]. This antipattern is a counterpart of that fact. Developers may forget that existential restrictions contain a cardinality constraint: $C1 \subseteq \exists R.C2 \models C1 \subseteq (\geq 1R.C2)$. Thus, when they combine existential and cardinality restrictions, they may be actually thinking about universal restrictions with those cardinality constraints.

Guideline GroupAxioms (GA)

$C1 \subseteq \forall R.C2$, $C1 \subseteq (\geq 2R.T)$ (for example)

In order to facilitate the comprehension of complex class definition, we recommend grouping all the restrictions of a concept that use the same role R in a single restriction. The previous restriction becomes $C1 \subseteq (\forall R.C2) \cap (\geq 2R.T)$

Guideline MinIsZero (MIZ)

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

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.

5.6 Future Works

A previous versions of this set of antipatterns with examples from HydrOntology was already accepted for publication in the workshop "Construction d'ontologies : vers un guide des bonnes pratiques" Hammamet (Tunisie), 25 Mai 2009.

We are still working to propose some recommendations associated to each antipatterns, so that we can provide better explanations of the reasons why a specific class or set of classes of the ontology are inconsistent, and hence improve the efficiency of the ontology debugging process.

All these anti-patterns should be seen as elementary units that cause ontology inconsistencies. That is, they can be combined into more complex ones. However, providing a solution for the individual ones will be a good advance to the current state of the art, and our future work will be also devoted to finding the most common combinations and providing recommendations for them.

We have applied this list of anti-patterns to the development of an ontology in the hydrology domain (HydrOntology [Vilches-Blázquez & al, 2007]), resulting in an improvement in the efficiency of the debugging process that we have not

actually measured. However, our intuition suggests that the process has been much faster than what it would have been without the use of such anti-patterns, that is, with the use of debugging tools alone.
Our next steps towards providing effective tools to help domain experts in their ontology building tasks are making formal experiments with a set of inconsistent ontologies, built by domain experts that we have been collecting in the past year. The aim of these experiments would be to compare the time needed to complete the debugging process with and without the use of our anti-patterns, and the quality of the final models generated after debugging, in case that there are differences. Finally, another piece of work that we are planning to do in the future is to organize this list of anti-patterns into a set of debugging guidelines for the creation of a better-specified method for ontology debugging that can be more effective.

6 Results and Future Collaboration

Parts of the work carried out during the STSM are already accepted for publication.
Oscar Corcho, Catherine Roussey, Luis M Vilches Blazquez. "Catalogue of Anti-Patterns for formal Ontology debugging". In Proceedings of AFIA Workshop "Construction d'ontologies : vers un guide des bonnes pratiques" Hammamet (Tunisie), 25 Mai 2009.

After the end of the STSM, the collaboration between OEG and LIRIS will go on to complete the work on ontology debugging guidelines in order to provide a more complete method and to evaluate it. We plan to propose a contribution for K-CAP 2009 The Fifth International Conference on Knowledge Capture.

After this work on formal ontology debugging, each partners will have an example of consistent formal ontologies dealing with geographic data. Thus we are interested to investigate semantic data integration for geographic data bases using formal ontologies.

7 References

[Arens & al, 1996] Y. Arens, C. Hsu, C.A. Knoblock. "Query Processing in SIMS information mediator" in Advanced Planning Technology, editor, Austin Tate, AAAI Press, Menlo Park, CA, 1996. http://citeseer.ist.psu.edu/article/arens96query.html

[Arpírez & al, 1998] Arpírez JC, Gómez-Pérez A, Lozano A, Pinto HS. "(ONTO)2Agent: An ontology-based WWW broker to select ontologies". In Gómez-Pérez A, Benjamins RV (eds) ECAI'98 Workshop on Applications of Ontologies and Problem-Solving Methods. Brighton, United Kingdom, 1998. pp 16–24

[Barrasa & al, 2004] Barrasa J, Corcho O, Gómez-Pérez A. "R2O, an Extensible and Semantically Based Database-to-Ontology Mapping Language". Second Workshop on Semantic Web and Databases (SWDB2004). Toronto, Canada. August 2004.

[Chaudri & al, 1998] Chaudhri VK, Farquhar A, Fikes R, Karp PD, Rice JP. "Open Knowledge Base Connectivity 2.0.3". Technical Report KSL-98-06, Knowledge

Systems Laboratory, Stanford, CA, 1998. http://www.ai.sri.com/ okbc/okbc-2-0-3.pdf

[Chawathe & al, 1994] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. "The TSIMMIS Project: Integration of Heterogeneous Information Sources". In Proceedings of IPSJ Conference, pp. 7-18, Tokyo, Japan, October 1994.

[Cali & al, 2002] A. Cali, D. Calvanese, G. De Giacomo, M. Lenzerini. "On the expressive power of data integration systems". In Proceedings of 21st International Conference on Conceptual Modeling (ER2002), Tampere, Finland, October 2002. p 338–350.

[Farquhar & al, 1997] Farquhar A, Fikes R, Rice J. "The Ontolingua Server: A Tool for Collaborative Ontology Construction". International Journal of Human Computer Studies, 46 (6), 1997: 707–727

[Friedman & al, 1999] M. Friedman, A. Levy, and T. Millstein. "Navigational plans for data integration". In Proceedings of the 16th National Conference on Artificial Intelligence (AAAI'99), Orlando, Florida, 1999. p 67–73.

[Gómez-Pérez & al, 2003] Gómez-Pérez A, Fernández-López M, Corcho O. "Ontological Engineering". Springer-Verlag, London (United Kingdom), 2003

[Gruber, 1995] Gruber TR. "Toward principles for the design of ontologies used for knowledge sharing". International Journal of Human-Computer Studies, vol 43 n.5-6, 1995.

[Levy & al, 1996] A Y. Levy, A Rajaraman, J Ordille. "Querying Heterogeneous Information Sources Using Source Descriptions" In Proceedings of the Twenty-second International Conference on Very Large Data Bases (VLDB' 1996) Mumbai, India, september 1996. P 251-262

[McBrien and Poulovassilis, 2004] P.J. McBrien and A. Poulovassilis. "Defining Peer-to-Peer Data Integration using Both as View Rules". In Proceedings of DBISP2P 2003 Revised Paper. Springer Verlag LNCS, Volume 2944, 2004, ISBN 3-540-20968-9. P 91-107.

[Mena & al, 1996] E. Mena, V. Kashyap, A. Sheth, A. Illarramendi. "OBSERVER: An Approach for Query Processing in Global Information Systems based on Interoperation across Pre-existing Ontologies". In Proceedings of the First IFCIS International Conference on Cooperative Information Systems (CoopIS'96), Brussels, Belgium, June 19-21, 1996. p14-25.

[Maurizio and Lenzerini, 2002]. Maurizio Lenzerini (2002). "Data Integration: A Theoretical Perspective". PODS 2002: 233-246.]

[Protégé] Protégé-OWL version 4: free ontology editor developed by Stanford for more detail http://protege.stanford.edu/

[Vilches-Blázquez & al, 2007] Vilches-Blázquez LM, Bernabé-Poveda MA, Suárez-Figueroa MC, Gómez-Pérez A, Rodríguez-Pascual AF. "Towntology &

hydrOntology: Relationship between Urban and Hydrographic Features in the Geographic Information Domain”. In Ontologies for Urban Development. Studies in Computational Intelligence, vol. 61, Springer, 2007. pp 73–84

[Wache & al, 2001] H. Wache, T. Vögele, U. Visser, H. Stuckenschmidt, G. Schuster, H. Neumann, S. Hübner. "Ontology-Based Integration of Information A Survey of Existing Approaches". In Proceedings of the IJCAI-Workshop Ontologies and Information Sharing, A. Gomez-Perez, M. Gruninger, H. Stuckenschmidt and M. Uschold (eds.), Seattle, WA, September, 2001. p 108-117.