Preliminary Report on an Algebra of Lightweight Ontologies

(Extended Abstract)

Marco A. Casanova¹, Eveline R. Sacramento¹,², José A. F. de Macêdo², Ângela M. A. Pinheiro², Vânia M. P. Vidal², Karin K. Breitman¹, Antonio L. Furtado¹

¹Department of Informatics – PUC-Rio – Rio de Janeiro, RJ – Brazil
(casanova, esacramento, karin, furtado)@inf.puc-rio.br
²Department of Computing, Federal University of Ceará – Fortaleza, CE – Brazil
(jose.macedo, angela, vvidal)@lia.ufc.br
³Ceará State Foundation of Meteorology and Water Resources – Fortaleza, CE – Brazil
eveline@funceme.br

1 Motivation

We argue that certain familiar ontology design problems are profitably addressed by treating ontologies as theories and by defining a set of operations that create new ontologies, including their constraints, out of other ontologies. Such operations extend the idea of namespaces to take into account constraints.

Consider first the problem of designing an ontology to publish data on the Web. If the designer follows the Linked Data principles (Bernes-Lee, 2006; Bizer et al., 2007), he must select known ontologies, as much as possible, to organize the data so that applications “can dereference the URIs that identify vocabulary terms in order to find their definition”. We argue that the designer should go further and analyze the constraints of the ontologies from which he is drawing the terms to construct his vocabulary. Furthermore, he should publish the data in such a way that the original semantics of the terms is preserved. To facilitate ontology design from this perspective, we introduce three operations on ontologies, called projection, union and deprecation.

Consider now the problem of comparing the expressive power of two ontologies, O₁=(V₁,Σ₁) and O₂=(V₂,Σ₂). If the designer wants to know what they have in common, he should create a mapping between their vocabularies and detect which constraints hold in both ontologies, after the terms are appropriately mapped. The intersection operation answers this question. We argued elsewhere (Casanova et al., 2010) that intersection is also useful to address the design of mediated schemas that combine several export schemas in a way that the data exposed by the mediator is always consistent.

On the other hand, if the designer wants to know what holds in O₁=(V₁,Σ₁), but not in O₂=(V₂,Σ₂), he should again create a mapping between their vocabularies and detect which constraints hold in the theory of Σ₁, but not in the theory of Σ₂, after the terms are appropriately mapped. The difference operation answers this question.

2 Definition of the Ontology Operations

The definition of the operations depends only on the notion of theory, which we introduce in the context of Description Logic (DL) (Baader and Nutt, 2003).

Let Σ be a set of inclusions in a vocabulary V. The theory of Σ in V, denoted τ[Σ], is the set of all inclusions in V that are logical consequences of Σ. We say that two sets of inclusions, Γ and Θ, are equivalent, denoted Γ ≡ Θ, iff τ[Γ]=τ[Θ].
An ontology is a pair $O=(V,\Sigma)$ such that $V$ is a finite vocabulary, whose atomic concepts and atomic roles are called classes and properties of $O$, respectively, and $\Sigma$ is a set of inclusions in $V$, called the constraints of $O$. Two ontologies $O_1=(V_1,\Sigma_1)$ and $O_2=(V_2,\Sigma_2)$ are equivalent, denoted $O_1 \equiv O_2$, iff $\Sigma_1$ and $\Sigma_2$ are equivalent.

**Definition 1**: Let $O_i=(V_i,\Sigma_i)$ and $O_2=(V_2,\Sigma_2)$ be two ontologies, $W$ be a subset of $V_i$, and $\Psi$ be a set of constraints in $V_j$.

(i) The projection of $O_i=(V_i,\Sigma_i)$ over $W$, denoted $\pi(W)(O_i)$, returns the ontology $O_p=(V_p,\Sigma_p)$, where $V_p=W$ and $\Sigma_p$ is the set of constraints in $\tau(\Sigma_i)$ that use only classes and properties in $W$.

(ii) The deprecation of $\Psi$ from $O_i=(V_i,\Sigma_i)$, denoted $\delta(\Psi)(O_i)$, returns the ontology $O_D=(V_p,\Sigma_D)$, where $V_p=V_i$ and $\Sigma_D=\Sigma_i - \Psi$.

(iii) The union of $O_i=(V_i,\Sigma_i)$ and $O_2=(V_2,\Sigma_2)$, denoted $O_i \cup O_2$, returns the ontology $O_U=(V_U,\Sigma_U)$, where $V_U=V_i \cup V_2$ and $\Sigma_U=\Sigma_i \cup \Sigma_2$.

(iv) The intersection of $O_i=(V_i,\Sigma_i)$ and $O_2=(V_2,\Sigma_2)$, denoted $O_i \cap O_2$, returns the ontology $O_e=(V_e,\Sigma_e)$, where $V_e=V_i \cap V_2$ and $\Sigma_e=\tau(\Sigma_i) \cap \tau(\Sigma_2)$.

(v) The difference of $O_i=(V_i,\Sigma_i)$ and $O_2=(V_2,\Sigma_2)$, denoted $O_i - O_2$, returns the ontology $O_F=(V_F,\Sigma_F)$, where $V_F=V_i$ and $\Sigma_F=\tau(\Sigma_i) - \tau(\Sigma_2)$.

We observe that we may generalize union, intersection and difference by considering a renaming of one or both vocabularies of the ontologies involved and appropriately renaming the terms that occur in the constraints when comparing the theories.

3 Final remarks

Previous work by the authors (Casanova et al., 2011; Casanova et al., 2012a) introduced the notion of open fragment, which is captured by the projection operation, and some of the operations discussed in detail in this paper. We have also developed procedures to compute the operations when the inclusions are lightweight, based on the machinery developed in (Casanova et al., 2010, 2012b). A tool that implements the operations is described in (Pinheiro, 2013).

As future work, we intend to expand the implementation of the operations to cover a more expressive family of ontologies, using the results presented in (Casanova et al., 2012a). We also intend to integrate the OntologyManagement tool with the Protégé ontology editor to take advantage of all functionalities already available in Protégé, such as ontology modeling and visualization, inference and reasoning tasks.


**References**


