Interrogation et Vérification de documents OWL dans le modèle des Graphes Conceptuels

Thomas Raimbault*, Henri Briand**, Rémi Lehn**, Stéphane Loiseau *

*LERIA, Université d'Angers, 2 bd Lavoisier 49045 ANGERS Cedex 01
{thomas.raimbault, stephane.loiseau}@info.univ-angers.fr
**LINA, École Polytechnique de Nantes, rue C. Pauc BP 50609 44306 Nantes Cedex 3
{henri.briand, remi.lehn}@polytech.univ-nantes.fr

Résumé. OWL est un langage pour la description d'ontologies sur le Web. Cependant, en tant que langage, OWL ne fournit aucun moyen pour interpréter les ontologies qu'il décrit, et étant orienté machine, il reste difficilement compréhensible par l'humain. On propose une approche de visualisation, d'interrogation et de vérification de documents OWL, regroupées dans un unique environnement graphique : le modèle des graphes conceptuels.

1 Introduction

OWL (W3C, 2004) - Ontology Web Language - est un langage pour décrire des ontologies et les diffuser sur le Web. Il est important de noter que d'une part, OWL est un langage et qu'à ce titre il ne fournit aucun moyen pour interroger ou vérifier ses documents, et d'autre part étant orienté machine, il reste difficilement compréhensible par l'humain. Des outils ont donc été créés pour répondre à ces exigences. Cependant, ces outils traitent soit séparément l'un de ces besoins (HP, 2000; WonderWeb, 2002), soit les traitent de façon globale (Protégé, 2004; Haarslev et Müller, 2001) mais avec des interrogations prédéfinies et un ensemble figé de vérifications.

Dans cet article, notre approche est de regrouper dans un unique environnement, adaptable par l'utilisateur, à la fois la représentation de documents OWL, ainsi que des outils de raisonnement sur ces documents. Pour se faire, nous avons choisi comme base de travail le modèle des graphes conceptuels (GCs). Ce modèle, introduit par (Sowa, 1984), est un modèle formel et visuel de représentation des connaissances muni d'une sémantique logique. Nous utilisons dans cet article le modèle issu de (Mugnier et Chein, 1996) et étendu aux GCs emboîtés (Chein et Mugnier) avec règles (Salvat, 1998) et contraintes (Baget et Mugnier, 2002).

Notre travail fournit deux contributions fondamentales. La première est de coder les différentes notations qui décrivent - c'est-à-dire le métamodèle - un sous-langage OWL dans un support du modèle des GCs, noté support_{OWL} (Section 2). Ainsi, nous proposons une traduction générique - et donc automatisable - d'un document OWL en un GC, défini sur ce support_{OWL}, qui lui est sémantiquement équivalent et que nous appelons *GC-document OWL* (Section 3). La seconde contribution (Section 4) est une méthode, utilisant les opérateurs qu'offre le modèle des GCs, pour interroger un document OWL ou pour en vérifier la validité au travers de *spécifications orientées ontologie*.

2 Modélisation des notations OWL dans le support$_{OWL}$

2.1 Courte introduction à OWL

Une ontologie formalise les connaissances d'un domaine en définissant les concepts du domaine en termes de classes hiérarchiquement organisées et leurs propriétés (attributs ou slots). Les individus (instances de classes) et leurs relations composent la connaissance. Ces relations peuvent être pleinement définies au moyen de propriétés (restrictions) qui les contraignent.

La Figure 1 présente un document OWL sous sa forme en triplets [sujet, prédicat, objet] de graphe RDF, où un sommet source correspond à un sujet, un sommet destination à un objet et un arc représente un prédicat. Par exemple, le triplet [f#Marie, rdf:type, f:Femme] nous indique que 'Marie' est une instance de la classe 'Femme'. Où 'Femme' est bien une classe puisqu'il s'agit d'une sous-classe de 'Humain' qui est une classe. En effet [f:Femme, rdfs:subClassOf, f:Humain] et [f:Humain, rdf:type, owl:Class] l'indiquent.

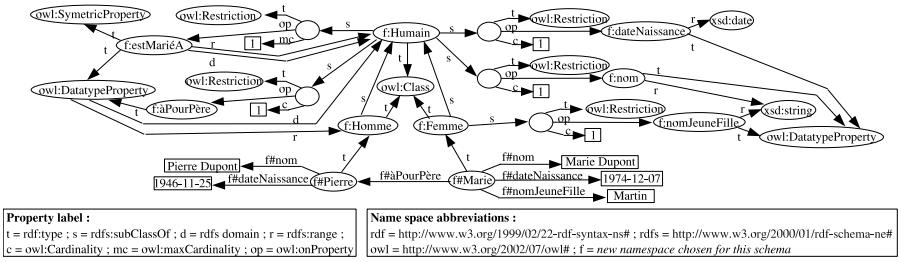

FIG. 1 – *Exemple de document OWL (représenté par son graphe RDF)*

2.2 Support$_{OWL}$

Nous définissons et organisons le vocabulaire qui décrit le langage OWL (W3C, 2004) - son métamodèle - dans un support de GCs, noté support$_{OWL}$. Cette organisation est basée un sous-langage OWL proche de OWL Lite et OWL DL, de part les similitudes et divergences entre les langages OWL et le modèle des GCs (Coupey et Faron, 1998). L'utilisation de ce support permet une traduction générique, et donc automatique, d'un document OWL en un GC sémantiquement équivalent. En effet, le support$_{OWL}$ n'est pas la définition de la connaissance contenue dans une ontologie particulière, mais bien la définition des notations qui font le langage OWL.

Définition 1 (Support$_{OWL}$). *Le support$_{OWL}$ est un* support *de graphes conceptuels, où les notations OWL sont définies et organisées dans l'ensemble des types concepts (Figure 2) et les ensembles des types relations (Figure 3).*

3 GC-document OWL

Nous indiquons maintenant quelques uns de nos mécanismes de traduction d'un document OWL en un GC sémantiquement équivalent, noté *GC-document OWL*. La Figure 4 par

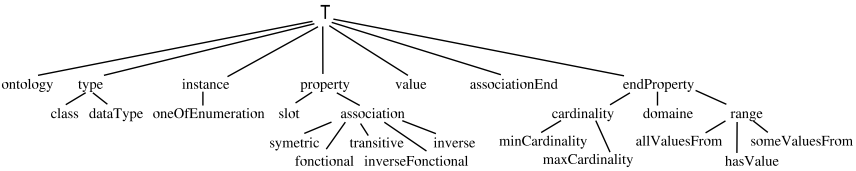

FIG. 2 – *Ensemble des types de concepts du support$_{OWL}$*

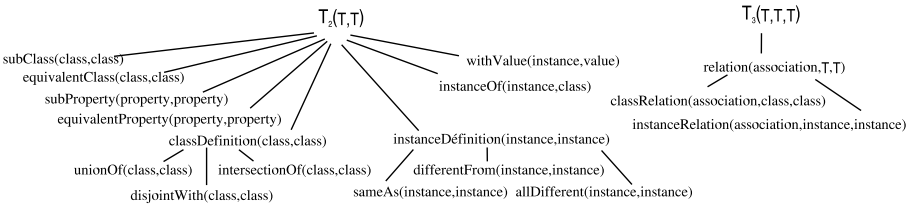

FIG. 3 – *Ensembles des types de relations du support$_{OWL}$*

exemple est la résultat de la traduction du document OWL présenté Figure 1 en son équivalent GC. Cette traduction présente l'avantage de regrouper dans un unique environnement : une représentation visuelle et compréhensible par l'humain d'un document OWL, tout en permettant l'interrogation et la vérification (Section 4) du document.

Définition 2 (GC-document OWL). *Un GC-document OWL est un graphe conceptuel emboîté qui est défini sur le support$_{OWL}$, conformément aux Propositions 1 à 5.*

3.1 Les classes et leurs relations

Proposition 1 (Classe et GC-document OWL). *Une classe OWL est représentée par un sommet concept de type* class*, dont le marqueur individuel est l'identifiant de la classe s'il existe, sinon de marqueur générique.*
Un slot d'une classe est représenté par un sommet concept individuel de type slot*, dont le marqueur individuel est l'identifiant de la propriété* owl:DatatypeProperty*. Ce sommet concept est emboîté au sein du sommet concept représentant la classe qu'il caractérise.*

Proposition 2 (Sous-classe et GC-document OWL). *Une relation de généralisation entre deux classes est modélisée par un sommet relation de type* subClass*. Le premier voisin de cette relation est la classe la plus générale, le second la classe la plus spécifique (la sous-classe).*

Proposition 3 (Association entre classes et GC-document OWL). *Une association entre deux classes est centralisée par un sommet concept individuel de type* association *ou un de ses sous-type, et dont le marqueur individuel est l'identifiant de la propriété* owl:ObjectProperty*. Les classes qui participent à l'association y sont liées par un sommet relation de type* classRelation*, dont le premier voisin est le sommet concept de type* association*, le deuxième le sommet concept représentant la classe, et le troisième un sommet concept générique de type*

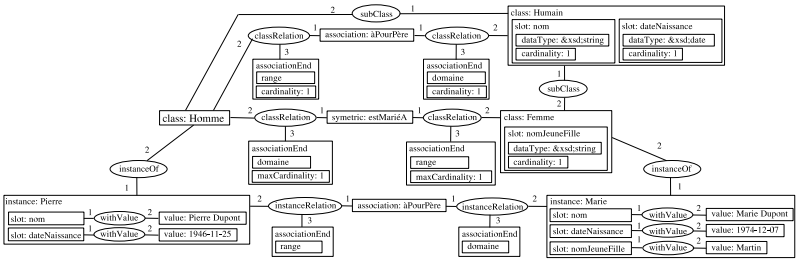

FIG. 4 – *Le* GC-document OWL *du document OWL en Figure 1*

associationEnd*. Au sein de ce dernier sont emboîtées les propriétés qui caractérisent comment la classe participe à l'association - comme la cardinalité -. Les sommets concepts qui représentent ces propriétés sont d'un type sous-type de* endProperty.
Pour de la définition d'une classe par rapport à d'autres classes, cette classe est liée aux autres par un sommet relation binaire dont le type est un sous-type de classDefinition.

3.2 Les individus et leurs relations

Proposition 4 (Instance d'une classe et GC-document OWL). *Un individu, ou instance d'une classe, est représenté par un sommet concept individuel de type* instance*, dont le marqueur individuel est le nom de l'individu. Chaque slot de la classe, dont l'individu est instance, est reconstruit et emboîté dans le sommet concept représentant l'individu. Puis à un slot est lié, par un sommet relation de type* withValue*, un sommet concept individuel de type* value *avec pour marqueur individuel la valeur du slot.*

Proposition 5 (Association entre individus et GC-document OWL). *Une association entre deux individus est représentée selon le même principe qu'une association entre classes : centralisée par un même sommet concept de type* association*. Les individus et le sommet concept de type* association *sont liés entre eux par un sommet relation de type* instanceRelation.
Pour la définition d'un individu par rapport à d'autres individus, cet individu est lié aux autres par un sommet relation dont le type est un sous-type de instanceDefinition.

4 Interroger et Vérifier un GC-document OWL

Nous proposons une méthode pour interroger et vérifier des GC-documents OWL, par l'utilisation des opérateurs du modèle des GCs. Comme GC-documents OWL, requêtes et contraintes sont définies sur un même niveau de représentation, les interrogations et les vérifications peuvent faire intervenir toute la connaissance présente dans l'environnement GC.

4.1 Interroger un GC-document OWL

En s'appuyant sur le support$_{OWL}$, il est possible d'interroger un document OWL. Une requête s'exprime simplement par un GC défini sur ce même support. Les réponses à une requête sont les sommets concepts du GC-document OWL pour lesquels les sommets concepts génériques - de marqueur générique * - de la requête coïncident.

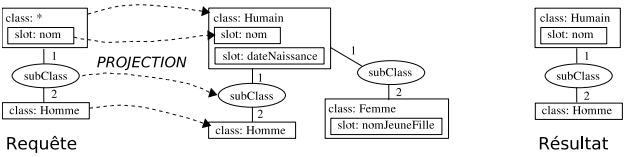

FIG. 5 – *Exemple de requête et son résultat*

Notons qu'avant d'interroger un GC-document OWL, il est nécessaire de faire ressortir explicitement les informations implicites contenues dans le document OWL. Par exemple, la notion de transitivité de la propriété *rdfs:subClassOf*. L'application de règles (Salvat, 1998; Baget et Mugnier, 2002), au sens graphe conceptuel, permet d'ajouter à notre GC-document OWL ce genre d'informations implicites (non traité ici).

Définition 3 (Résultat d'une requête). *Le(s) résultat(s) d'une requête Q, définie sur le support$_{OWL}$, sur un document OWL D est l'ensemble de projections de Q sur D.*

La partie gauche de la Figure 5 est un exemple de requête : "La classe *Homme* a-t-elle une classe mère qui possède un attribut nommé *nom* ?". En effet, la classe 'Homme' est modélisée par un sommet concept de type *class* avec un marqueur individuel *Homme*. Cette classe est liée à une classe parent qui n'est pas identifiée, représentée par le marqueur générique *, mais qui possède un slot 'nom' (sommet concept emboîté de type *slot*, ayant pour marqueur individuel *nom*). Cette requête, appliquée sur le GC-document OWL de la Figure 4, admet comme résultat, à droite de la Figure 5 : "oui, la classe *Homme* a une classe mère qui est *Humain*".

4.2 Vérifier un GC-document OWL

Pour vérifier la sémantique d'un GC-document OWL, nous utilisons des contraintes positives et négatives (Baget et Mugnier, 2002) du modèle des GCs. Ce sont des GCs emboîtés bicolores où la condition d'une contrainte est sur fond blanc et l'obligation (resp. l'interdiction) d'une contrainte positive (resp. négative) sur fond noir. Basées sur le support$_{OWL}$, nous définissons des contraintes, appelées *spécifications orientées ontologie*, de telles sortes qu'elles valident ou non la bonne définition sémantique d'une ontologie. Ces contraintes ne sont pas figées est peuvent être complétées ou adaptées graphiquement par l'utilisateur.

Définition 4 (Validité d'un GC-document OWL). *Un GC-document OWL est valide si et seulement si il satisfait toutes les* spécifications orientées ontologie.

Par exemple, la Figure 6 montre une contrainte négative C_1 et une contrainte positive C_2. C_1 exprime l'interdiction (sur fond noir) de cycle d'héritage par la relation transitive *subClass* : "une classe *A* ne peut être une sous-classe de *B*, qui est une sous-classe de *A*". La condition (sur fond blanc) de C_2 est une classe définie par une énumération d'individus puisqu'elle possède au moins un individu énuméré, qui est représenté par un sommet concept de type *instance* lié à la classe par un sommet relation de type *oneOfEnumeration*. S'ajoute à cette condition une autre instance de cette classe liée par un sommet relation de type *instanceOf*. L'obligation (sur fond noir) de C_2 est que cette dernière instance doit faire partie des individus énumérés de la classe, car une telle classe doit être constituée d'un ensemble exhaustif d'individus énumérés.

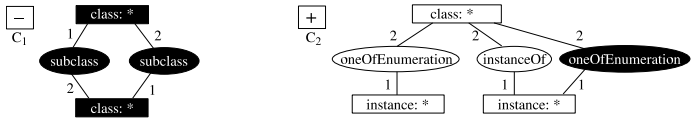

FIG. 6 – *Exemple de spécifications orientées ontologie*

5 Conclusion

Ce travail fournit une méthode de traduction automatique d'un sous-langage OWL en un graphe conceptuel sémantiquement équivalent. Le but de cette traduction est de disposer dans un unique environnement graphique : une visualisation des documents OWL d'une part, et d'autre part la possibilité de les interroger ou de les vérifier grâce aux opérateurs du modèles des graphes conceptuels.

Références

Baget, J. et M. Mugnier (2002). Extensions of Simple Conceptual Graphs : the Complexity of Rules and Constraints. *JAIR 16*(12), 425–465.

Chein, M. et M. Mugnier. Positive Nested Conceptual graphs. In *Proc. of ICCS'97*, pp. 95–109.

Coupey, P. et C. Faron (1998). Towards Correspondence between Conceptual Graphs and Description Logics. In *Proc. of ICCS'98*, Volume 1453 of *LNAI*, pp. 165–178. Springer.

Haarslev, V. et R. Müller (2001). Racer system description. In *Proc. of IJCAR'2001*, Volume 2083 of *LNAI*, pp. 701–705. Springer.

HP (2000). HP Labs, Jena A Semantic Web Framework for Java. http://jena.sourceforge.net/.

Mugnier, M. et M. Chein (1996). Représenter des connaissances et raisonner avec des graphes. *Revue d'intelligence artificielle 10*(1), 7–56.

Protégé (2004). Protégé OWL Plugin User's Guide. http://protege.stanford.edu/plugins/owl/.

Salvat, E. (1998). Theorem Proving Using Graph Operations in the Conceptual Graph Formalism. In *Proc. of ECAI'98*.

Sowa, J. (1984). *Conceptual Structures : Information Processing in Mind and Machine*. Addison Wesley.

W3C (2004). OWL Web Ontology Language Overview. http://www.w3.org/TR/owl-features/.

WonderWeb (2002). WonderWeb - Ontology Infrastructure for the Semantic Web. http://wonderweb.semanticweb.org.

Summary

OWL is a language to describe ontologies on the Web. Owing to the fact that OWL is a language, it does not provide any means to interpret ontologies which it defines, and because of its oriented machine aspect, it is not easily comprehensible by the human. We propose an approach to visualize, interrogate and check OWL documents in an unique graphical environment: the model of the conceptual graphs.