oogo : Ontologie des Outils utiles à la Gestion d'Ontologies

Sylvie Despres

oogo : Ontologie des Outils utiles à la Gestion d'Ontologies

Sylvie Despres

Université Paris 13, Sorbonne Paris Cité, LIMICS, (U1142), INSERM, Sorbonne Universités, UPMC Université Paris 6,
74 rue Marcel Cachin F-93017 Bobigny cedex, France,
sylvie.despres@univ-paris13.fr

Résumé : Cet article présente, oogo, une ontologie ayant pour objectif la description des outils utiles à la gestion d'ontologies. A l'origine, ce travail répondait à une demande qui était de présenter à des chercheurs et/ou des industriels des outils utilisables pour élaborer une ontologie de domaine et si nécessaire leur permettre de les sélectionner en fonction de leurs besoins. Une ontologie nous a semblé constituer la forme la plus adaptée pour délivrer cette information. Le modèle conceptuel la structurant est fondé sur le cycle classique de construction d'ontologies commun à toutes méthodologies. Les activités afférentes à la construction d'ontologies servent de guide à la présentation des fonctionnalités les plus utiles de ces outils. Les scénarios d'usage sont construits à partir des besoins des acteurs intervenant autour d'une ontologie. Un état de l'art relatif à chaque catégorie d'outils sous tend la construction du modèle conceptuel de oogo.

Mots-clés : oogo, ontologie, ingénierie des ontologies, outils pour l'ingénierie ontologique.

1 Introduction

Dans cet article, nous décrivons, oogo, une ontologie ayant pour objectif la description des outils utiles à la gestion d'ontologies. A l'origine, ce travail répondait à une demande des organisateurs de la journée PDIA2017[1] qui était de présenter à des chercheurs et/ou des industriels des outils utilisables pour élaborer une ontologie de domaine et si nécessaire leur permettre de les sélectionner en fonction de leurs besoins.

La première question qui s'est posée pour répondre à cette requête a été de trouver la forme la plus pertinente pour renseigner ces acteurs. Il existe en effet de nombreux articles décrivant les différents outils et de nombreuses comparaisons les concernant. Ces dernières sont difficiles à exploiter car elles sont statiques, dédiées à des catégories d'outils particulières et les critères de comparaison utilisés peuvent varier d'un article à l'autre. Dans ce contexte, l'utilisation d'une ontologie nous a semblé la forme la mieux adaptée pour donner accès à l'ensemble des caractéristiques des outils actuellement disponibles. Elle permet entre autres de répondre aux questions les plus fréquemment posées lors de la sélection des outils intervenant dans la gestion d'une telle ressource. Elle est destinée à être publiée pour être réutilisée et enrichie par la communauté d'ingénierie des ontologies. Les utilisateurs concernés par oogo sont des ingénieurs de la connaissance ou des chargés de mission en « information technology » en charge de la construction d'ontologies et s'interrogeant sur les fonctionalités des outils existants.

Le modèle conceptuel de oogo est fondé sur les activités intervenant dans le cycle classique de construction d'ontologies (scénario 1 de la méthodologie Neon) commun à toutes méthodologies de construction (Suárez-Figueroa et al., 2015). Les activités afférentes à la

[1] https://afia.asso.fr/wp-content/uploads/2018/01/cr-PDIA-2017-vf.pdf

construction d'ontologies servent de guide à la présentation des fonctionnalités les plus utilisées de ces outils (Suárez-Figueroa, Gomez Perez, 2008). Le périmètre de l'ontologie est établi en adoptant la méthodologie de (Uschold et Gruninger, 1996). Les scénarios d'usage identifiés sont issus de cas réels rencontrés lors de la construction d'ontologies. A partir de ces scénarios, une liste non exhaustive de questions de compétence, auxquelles oogo doit être en mesure de répondre, a été élaborée à partir des questions posées par les acteurs devant construire une ontologie. Un état de l'art relatif à chaque outil aide à la construction du modèle conceptuel de oogo.

Dans la suite de l'article, la section 2 décrit le modèle adopté pour le concept Outil à partir duquel les modèles des outils présentés dans la section 3 sont décrits. Puis nous concluons dans la section 4 sur l'intérêt et l'usage d'une telle ontologie.

2 Modèle adopté pour le concept Outil

Nous envisageons l'outil comme un moyen permettant d'obtenir un résultat correspondant à la mise en œuvre d'un service. Nous proposons un modèle pour le concept d'outil dédié à la construction de ressources du web sémantique que nous spécialiserons en fonction du type d'outil étudié.

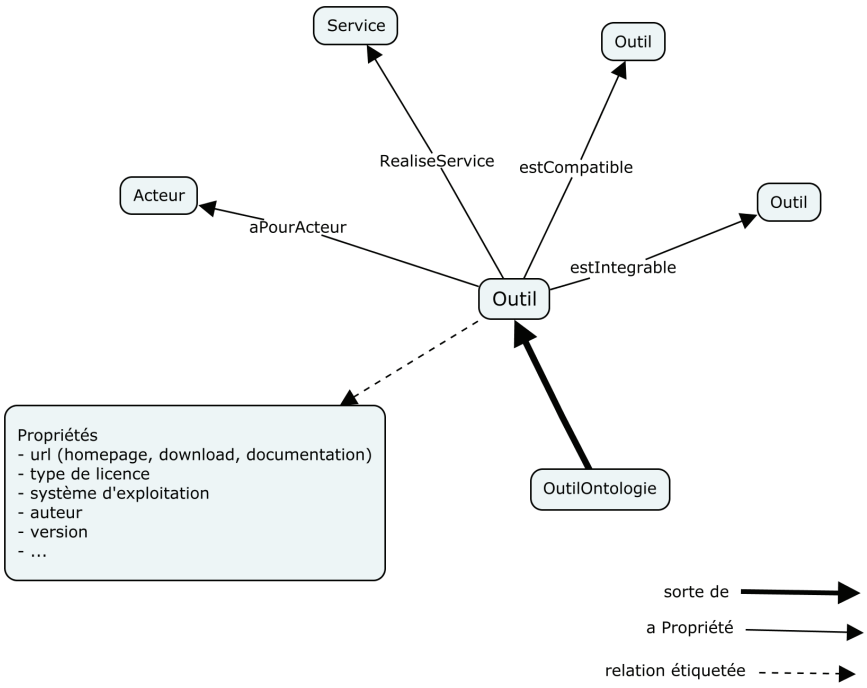


FIGURE 1 – Modèle du concept Outil

En utilisant plusieurs vocabulaires standards, un outil est identifié par une URI correspondant à sa homePage, une URI de documentation et une URI de téléchargement. Il peut être caractérisé par des mots-clés. Il a été conçu par un auteur ou une équipe d'auteurs et développé par une organisation. Il est exécutable sous un SE. Il est identifié par un type de licence. Il est mis à disposition selon différentes modalités. Il fournit un type de service. Il est interopérable avec d'autres outils. Il dispose d'un format de stockage pour les fichiers contenant les ressources. Il permet l'import et l'export de ressources. Il est capable de gérer les sauvegardes avec ou sans gestion de version.

Plusieurs vocabulaires sont réutilisés pour la construction du modèle concept Outil : dcterms, doap, dvia, educore, foaf, org.

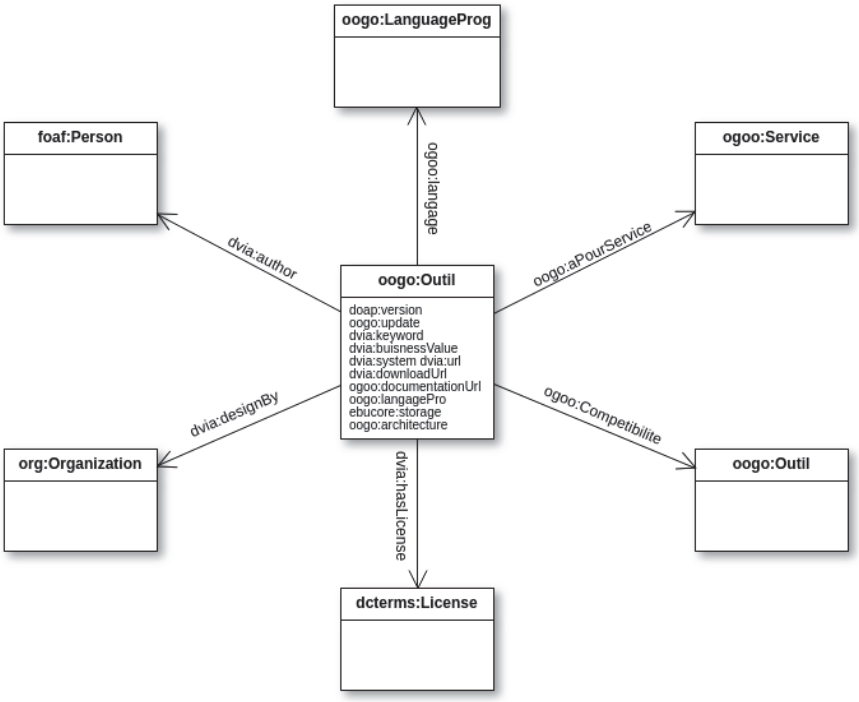


FIGURE 2 – Aperçu des vocabulaires réutilisés

3 Démarche de construction de l'ontologie oogo

Nous nous intéressons aux outils dédiés à la construction d'ontologie définie comme « une spécification formelle explicite d'une conceptualisation partagée » (Studer et al., 1998). Le langage de représentation considéré pour la formalisation de ces ontologies est OWL/OWL2 standard du W3C.

Le cycle classique de développement d'une ontologie est constitué des étapes de spécification, planification, conceptualisation, formalisation, implémentation (Suárez-Figueroa et al., cf. *supra*). Au cours de ce cycle, les activités mises en œuvre (Suárez-Figueroa, Gomez Perez, cf. *supra*) comportent les activités de gestion, orientées développement (pré-développement, développement, post-développement) et de support à la construction. Dans oogo, nous décrivons une partie des outils supportant les activités de support et de développement.

Le cadre méthodologique dans lequel nous nous situons est celui du projet NeON (http://www.neon-project.org/). Nous adoptons le cycle classique de construction d'une ontologie (spécification, planification, conceptualisation, formalisation, implémentation) correspondant au scénario 1 de la méthodologie Neon.

La spécification des besoins auxquels doit répondre l'ontologie a été décrite dans l'introduction. La sélection des outils présentés reposent sur des critères qui sont relatifs à la disponiblité pour une installation, leur maintenance, leur actualité et leur formalisation en OWL. Elle est non exhaustive et pourra être étendue au fil de l'utilisation de l'ontologie. Une

partie a été collectée *via* le site du w3c[2] et de AI[3] (3) et les articles synthétisant les différents outils existants. Ces outils ont servi à l'élaboration des modèles leur correspondant et sont décrits dans oogo.

Nous avons identifié, pour chacun d'entre eux, les scénarios d'usage issus de l'expérience acquise au cours des différents projets de recherche menés au LIMICS et les questions de compétences qui leur sont associées, c'est-à-dire les questions auxquelles notre ontologie doit être en mesure de répondre. La construction des modèles de connaissances qui leur sont associés, est décrite dans la suite de ce paragraphe.

3.1 Activités de support

Les activités de support que nous considérons sont la modélisation des connaissances, la réutilisation, l'évaluation et la validation. Les outils pouvant supporter ces activités sont les outils de recherche d'ontologies (moteurs de recherche et bibliothèques d'ontologies) et les outils de validation.

3.1.1 Outils servant à la modélisation

Des outils de modélisation graphique semi-formelle peuvent être utilisés dans la phase de conceptualisation. Ils permettent de construire le modèle conceptuel du domaine qui sera ensuite formalisé avec l'éditeur d'ontologie. Leur utilisation intervient au cours de la phase d'acquisition de connaissances qui est réalisée de manière incrémentale.

3.1.1.1 Les scénarios

Scénario 1 : L'utilisateur cherche à cerner le périmètre de l'ontologie à construire. Il doit choisir une représentation sous forme de carte cognitive.

Scénario 2 : L'utilisateur cherche à représenter graphiquement le modèle du domaine en partant d'un concept central. oogo doit décrire les outils de modélisation avec leurs propriétés.

3.1.1.2 Les questions de compétence

Question 1 : J'ai identifié un concept central et je cherche à décrire les entités connexes en indiquant les relations qui les lient. Quel outil me permet de le faire ?

Question 2 : Comment déterminer la granularité de la représentation ?

Question 3 : Est-il possible de traduire automatiquement les cartes obtenues au format OWL ?

3.1.1.3 Les outils de modélisation sélectionnés

Nous avons sélectionné les outils Yed Graph Editor[4], XMind[5] et CmapsTools[6]. Ces trois outils disposent de versions exploitables sousWindows, OsX et Linux. Nous ne considérons pas dans cet article les métamodèles de représentation de OWL dans le formalisme UML et le langage GOWL de modélisation graphique, polymorphique et typé pour la construction d'ontologie (Héon et *al*., 2016).

Yed Graph Editor est un éditeur de graphes de bas niveau, open source et multiplateforme, personnalisable par programme. Il est souple dans les options de présentation disponibles et possède plusieurs fonctionnalités de disposition automatique des éléments du graphe. Il permet d'exporter les graphes au format graphml, SVG, png, emf, eps, etc.

---

[2] https://www.w3.org/wiki/

[3] http://www.mkbergman.com/904/listing-of-185-ontology-building-tools/

[4] http://www.yworks.com/yed

[5] http://www.xmind.net/

[6] https://cmap.ihmc.us/

XMind est un outil de carte heuristique qui permet également la construction de carte conceptuelle. Il permet de créer, gérer et partager des cartes conceptuelles et de les exporter au formats txt, html, image, SVG. XMind propose une version open source et une version payante. Dans la seconde version, il est possible d'exporter au format csv. Cette fonctionnalité peut ensuite être exploitée par d'autres outils de création semi-automatisée de l'ontologie.

CmapTools est un outil qui permet de créer et partager facilement des cartes conceptuelles. Il est utilisable en ligne *via* un simple navigateur ou téléchargeable sous la forme d'un exécutable. Il dispose d'options de mise en page automatique et permet d'exporter au format image, PDF, etc. Il permet de fusionner des nœuds identiques, de créer des nœuds imbriqués de valider des liens, de réparer automatiquement les liens rompus et d'annoter les nœuds. Il permet également la comparaison des cartes qui est une fonctionnalité utile pour la construction collaborative.

3.1.1.4 Le modèle pour les outils de modélisation

Le modèle du concept Outil de modélisation prend en compte les fonctionnalités qui leur correspondent. Nous avons en particulier retenu le mode de mise en forme, les formats d'export, leur licence, les options de partage, le mode d'exploitation (en ligne ou téléchargeable). Les notions de cartes heuristique et conceptuelle sont décrites.

3.1.2 Outils de recherche

3.1.2.1 Les scénarios

Scénario 1 : L'utilisateur cherche à réutiliser des ressources ontologiques pour construire sa propre ressource. Il s'interroge sur les outils lui permettant de réaliser cette recherche et le type de recherche qu'il peut effectuer. oogo doit représenter les outils avec les types de recherche associés.

Scénario 2 : L'utilisateur s'interroge sur le type de recherche utilisé par l'outil pour retrouver une ressource. oogo doit décrire les différentes modalités de recherche associées aux outils.

Scénario 3 : L'utilisateur a réalisé une recherche qui lui renvoie une liste de ressources, il cherche une explication du classement des ressources retournées. oogo doit préciser les métriques utilisée pour le classement (ranking) des ressources.

3.1.2.2 Les questions de compétence

Question 1 : Quels outils de recherche existent pour m'aider à trouver une ontologie de domaine ? (scénario 1)

Question 2 : Existe-t-il des vocabulaires dont je pourrais réutiliser les termes pour désigner les concepts ou les relations de mon ontologie ? (scénario 1)

Question 4 : Quelles sont les modalités de recherche mises en œuvre pour trouver une ressource ? (scénario 2)

Question 5 : Comment s'explique la classification des ressources proposées par l'outil de recherche ? (scénario 3)

3.1.2.3 Les outils de recherche sélectionnés

Parmi les outils de recherche permettant de trouver des ressources ontologiques répondant à des besoins de réutilisation figurent les moteurs de recherche sémantique, les portails et les bibliothèques d'ontologies.

- Les moteurs de recherche Swoogle (Finin et al., 2005), Watson (D'Aquin et al., 2011) et Vocab.cc (Stadtmüller et al., 2013) sont dédiés à la recherche d'éléments d'ontologies dont les vocabulaires constituent un sous-ensemble. L'expression de la recherche se fait en langue naturelle avec des mots clés et les résultats produits sont classés. Les modalités de recherche

sont les suivantes : - par URI ; - par mots-clés correspondant à une classe, une relation, un individu, et au périmètre de l'ontologie (Label, Commentaire, etc.) ; - par requêtes sur des triplets[7] ((terme générique, relation, terme) (terme générique, relation, ?), (terme générique relation, ?)) (Sabou et *al*., 2008).

- Le portail Linked Open Vocabularies (LOV) (une version indépendante est hébergée à l'OKFN), un des modules de Datalift (Vandenbussche et Vatant, 2014), qui est destiné à cataloguer, trouver et choisir des ontologies. Chaque vocabulaire est maintenu et publié de manière indépendante tout en entretenant des liens de dépendance vis-à-vis des autres vocabulaires.

- Les bibliothèques d'ontologies (BioPortal, AgroPortal, OBOFoundry, ODP) dont la finalité est de permettre et faciliter l'interopérabilité, fournir des ontologies reconnues et testées. Elles sont caractérisées par leur domaine, leur contenu et leurs fonctionnalités.

- L'outil OntoFox[8] prend en charge la réutilisation des ontologies. Il permet aux utilisateurs de saisir des termes, d'extraire des propriétés sélectionnées, des annotations et certaines classes de termes connexes à partir d'ontologies sources et de sauvegarder les résultats à l'aide de la sérialisation RDF / XML du langage OWL. Ontofox dispose également d'un algorithme d'extraction de termes fondé sur SPARQL qui extrait des termes liés à un ensemble donné de termes de signature. En outre, Ontofox fournit une option pour extraire la hiérarchie enracinée à un terme d'ontologie spécifié.

3.1.2.4 Le modèle proposé pour les outils de recherche

Le modèle pour les moteurs et le portail LOV a été construit à partir du tableau comparatif établi par (Vandenbussche et al., 2017). Les caractéristiques retenues comportent la méthode d'accès à la ressource (automatique et/ou manuelle), le type de la ressource (Semantic Web Document, concepts, termes de vocabulaire, vocabulaires), la métrique de ranking, le filtrage du domaine, l'accès au service web, etc.).

Le modèle pour les bibliothèques a été établi à partir du tableau comparatif proposé par (Debashis Naskar et Biswanath Dutta, 2016). Les caractéristiques décrites sont le domaine, les fonctionnalités de navigation et de recherche (sujet, structure, langue), le type d'appariement supporté (entre classe ou ontologie), le processus de soumission, l'utilisation et l'accès aux ontologies, les formats de fichiers en entrée et sortie, les outils associés.

3.1.3 Outils de validation

3.1.3.1 Les scénarios

Scénario 1 : L'utilisateur cherche à valider l'ontologie qu'il a construite et s'interroge sur les outils lui permettant de réaliser cette tâche. L'ontologie doit décrire les outils permettant les différents types de validation.

Scénario 2 : Quels sont les critères garantissant la validité d'une ontologie ? oogo doit préciser les critères utilisés pour la validation et les outils leur correspondant.

3.1.3.2 Question de compétences

Question 1 : Existe-il des outils pour valider les ontologies ? (scénario 1)

Question 2 : Quel outil me permet de vérifier la structure de mon ontologie ? (scénario 1 et 2)

Question 2 : Quel outil me permet de vérifier le modèle de l'expertise représenté dans mon ontologie ? (scénario 2)

[7] http://scarlet.open.ac.uk/poweraqua/index.html

[8] http://ontofox.hegroup.org/

3.1.3.3 Les outils de validation sélectionnés

Nous avons utilisé l'analyse des outils de validation réalisée par (Richard, 2017) pour la sélection des outils de validation. La validation de la qualité d'une ontologie se définit selon deux axes principaux qui sont la validation de la structure et la validation de la sémantique. La validation de la structure correspond à une validation formelle de la logique du modèle. Elle peut être réalisée automatiquement, grâce au développement d'outils dédiés. La validation de la sémantique consiste à mesurer l'adéquation du modèle conceptuel avec la réalité qu'il modélise. Cette étape nécessite une collaboration entre les ontologues et les spécialistes du domaine modélisé dans l'ontologie.

Les outils de validation de structure

Les raisonneurs permettent de vérifier la consistance et la cohérence d'un modèle (cf. Les outils de raisonnement, *infra*).

OntoCheck (Schober et *al*., 2012) constitue un module d'extension à l'éditeur d'ontologies Protégé et vise à contrôler le respect des conventions de nommage, ainsi que l'exhaustivité des méta-données. OntoCheck vérifie les cardinalités, la complétude des méta-données, les labels, les conventions typographiques ainsi que les métriques de l'ontologie.

XD Analyzer a été développé dans le cadre du projet NeOn[9], pour faire un retour qualitatif à l'utilisateur en suivant la méthodologie XD. Cette dernière fournit une liste de bonnes pratiques (concernant entre autres, les labels, les commentaires, les concepts non utilisés) à respecter pour la construction d'ontologies. Le module d'extension RaDON (Ji et al., 2009) a été ajouté à Neon Toolkit afin de faciliter les tâches de vérification de la consistance. Il comporte deux plugins permettant de travailler sur une ontologie ou un réseau d'ontologies.

OntOLogy Pitfall Scanner! (OOPS!)[10] (Poveda-Villalón et *al*., 2012) est un outil indépendant de tout éditeur d'ontologies. Il s'utilise uniquement en ligne. Le but de OOPS ! est l'identification des anomalies ou des mauvaises pratiques dans une ontologie. L'application prend en entrée l'URI d'une ontologie ou le code source en RDF. L'ontologie est chargée *via* l'API Jena avant d'être analysée pour en extraire les erreurs potentielles. Le résultat est une page répertoriant les erreurs selon le piège (pitfall) identifié, avec une proposition de résolution de l'erreur. Les pitfalls peuvent concerner des éléments individuels, plusieurs éléments ou toute l'ontologie.

Les outils de validation sémantique

Les outils permettant de valider la sémantique d'une ontologie nécessitent le recours à des experts du domaine modélisé par l'ontologie. Des outils supportant le développement de construction collaborative d'ontologies contribue à la validation sémantique.

Le serveur Ontolingua propose un environnement collaboratif pour parcourir, créer, éditer, modifier ou utiliser les ontologies (Farquhar et *al*., 1997)

WebProtégé : a été développé en reprenant l'architecture de Protégé. Il permet également le développement collaboratif d'ontologies et est accessible *via* n'importe quel navigateur web (Tudorache et *al*., 2013).

3.1.3.4 Le modèle pour les outils de validation

Le modèle des outils de validation décrit les fonctionnalités concernant la validation collaborative d'ontologies et précise les types de validation possible.

3.2 Activités de développement

Parmi les activités intervenant dans la phase de développement, nous considérons la conceptualisation, la formalisation, la modularisation, l'alignement. Les outils supportant ces

[9] http://www.neon-project.org

[10] http://oops.linkeddata.es/catalogue.jsp

activités sont les outils d'édition, les raisonneurs, les outils d'alignement et les outils de visualisation.

3.2.1 Outils d'édition

Un éditeur d'ontologie est un outil permettant d'inspecter, de parcourir, de codifier, de modifier les ontologies et de soutenir de cette manière le développement de l'ontologie et la tâche de maintenance.

3.2.3.1 Les scénarios

Scénario 1 : L'ontologue est débutant et cherche un éditeur pour construire une ontologie. Il n'a pas forcément une idée précise de fonctionnalité de l'outil. oogo doit décrire les différentes fonctionnalités de la phase de développement de l'ontologie.

Scénario 2 : L'ontologue a une idée précise de la nature de l'ontologie qu'il souhaite construire et des tâches qu'elle permettra de réaliser (annotation, raisonnement).

3.2.3.2 Les questions de compétence

Question1 : Quels sont les éditeurs d'ontologie disponibles ? (scénario 1)
Question 2 : Quel est le meilleur éditeur d'ontologie pour un débutant ? (scénario 1)
Question 3 : Quelles sont les activités supportées par l'éditeur d'ontologie ? (scénario 2)

3.2.3.3 Les outils d'édition sélectionnés

Nous avons utilisé le wiki[11] du W3C pour répertorier les éditeurs les plus utilisés actuellement pour construire des ressources ontologiques au sens où nous les avons définis dans cet article. Par conséquent nous n'avons pas retenu les éditeurs de vocabulaire. Parmi la liste des éditeurs inventoriés nous avons retenu Protégé, NeOn Toolkit, SWOOP et TopBraid Composer.

Nous avons également exploité les comparaisons des différents éditeurs pour identifier les propriétés des outils d'édition et construire un modèle de ces outils (Abburu, 2012 ; Abburu & Babu, 2013 ; Alatrish, 2013 ; Kapoor & Sharma, 2010 ; Parveen et al., 2016 ; Slimani, 2015).

3.2.3.4 Le modèle pour les outils d'édition

Nous avons retenu les caractéristiques servant à ces comparaisons pour construire le modèle des outils d'édition. Elles concernent l'architecture de l'outil et sa capacité d'évolution, l'interopérabilité, le paradigme de représentation des connaissances et le support méthodologique, les services d'inférences, les services de visualisation, la gestion des versions et l'utilisabilité. Certaines de ces caractéristiques sont déjà décrites dans le modèle du concept Outil (cf. Figure 1).

L'étude de l'utilisabilité des éditeurs réalisée par (Garcia Barriocal et al., 2006) pour des ontologues débutants met en évidence des propriétés relatives à la facilité de prise en main des outils. Certaines propriétés nous semble quantifiables comme accessibilité du langage utilisé à une communauté d'utilisateurs non spécialistes, la visualisation hiérarchique des classes, la navigation d'une classe vers ses instances, le filtrage des classes à partir de critères fixés par l'utilisateur

3.2.4 Outils de raisonnement

Une ontologie exprimée en OWL peut être considérée comme un ensemble de formules FOL (First Order logic), appelées axiomes, et par conséquent comme une théorie logique. Le vocabulaire exprimé en OWL est bien défini et sans ambiguïté. Cependant, il y a souvent un décalage entre la représentation visée et la représentation réelle de la connaissance décrite

[11] https://www.w3.org/wiki/Ontology_editors

dans l'ontologie. Alors que la connaissance du domaine d'intérêt est généralement bien maîtrisée ou tout au moins comprise, il n'est pas trivial de prévoir et de comprendre les conséquences logiques d'un ajout, d'un retrait ou d'une modification d'axiome, notamment quand la terminologie est fortement interconnectée. Ainsi, un ontologue a besoin d'utiliser des systèmes de déduction automatisés, communément appelés raisonneurs, pour vérifier certaines formes de déduction à partir d'axiomes préétablis et de rendre explicites ceux qui sont déclarés implicitement. Les raisonneurs sont donc, à l'heure actuelle, des éléments clés pour travailler avec des ontologies (Alaya, 2016).

3.2.4.1 Les scénarios

Scénario 1 : L'utilisateur cherche l'outil support du raisonneur (Neon, Protégé, Swoop, etc.). L'ontologie doit donc permettre de représenter les différents supports et de leur associer les raisonneurs existants.

Scénario 2 : L'utilisateur cherche à connaître les tâches supportées par un raisonneur. L'ontologie doit décrire les tâches de raisonnement et les entités sur lesquels ils portent.

Scénario 3 : L'utilisateur cherche les capacités d'explication et de gestion des erreurs du raisonneur.

Scénario 4 : L'utilisateur cherche le raisonneur adapté au contenu de l'ontologie à exploiter ? Dépend de la formalisation adoptée pour l'ontologie qui dépend elle-même des tâches à réaliser (annotation, raisonnement). Au différents profils sont associés des raisonneurs

3.2.4.2 Les questions de compétence

Question 1 : Sous quel système d'exploitation est-il exécutable ? Quel est le type de licence ? (scénario lié au modèle outil)

Question 2 : Avec quel éditeur le raisonneur est-il compatible ? Est-il standalone ?

Question 3 : Le raisonneur peut-il exploiter des règles et quel est le formalisme supporté ?

Question 4 : Le raisonneur prend t-il en charge le raisonnement sur les individus ?

Question 5 : Le raisonneur est-il adapté à la nature de l'ontologie que j'ai construite ?

Question 6 : Quels types de justification (explications sur les inconsistances) peut fournir le raisonneur ?

Question 7 : Quel type de profil OWL est pris en charge par le raisonneur ?

3.2.4.3 Les outils de raisonnement

Nous avons utilisé les comparaisons réalisées par (Dentler et *al*., 2011), (Abburu, 2012), (Matenzoglu et *al*., 2016) et (Alaya, 2016) pour construire le modèle des outils de raisonnement.

(Dentler et *al*., 2011) dressent une liste de caractéristiques guidant l'utilisateur dans le choix d'un raisonneur en fonction des besoins d'une application. Elle est envisagée selon deux dimensions : les caractéristiques du raisonnement et l'utilisabilité pratique. La première dimension regroupe : la procédure ou l'algorithme utilisé par le raisonneur pour le raisonnement en logique de description (algorithme Tableau, Hyper-Tableau et Consequence-Based), la robustesse et la complétude (vérifie si toutes les inférences possibles sont réalisées) qui peut aider à une accélération du temps de raisonnement, l'expressivité et la complétude (lien avec les profils OWL), la classification incrémentale (synchronisation du raisonneur), le support de l'utilisation de règles (par exemple, SWRL), la justification (explication pour un concept inconsistant, disjonction de concept, implication logique, etc.), le support des tâches de raisonnement sur la Abox (raisonnement sur les individus, vérification de la consistance de la Abox, etc.). La seconde a trait à la disponibilité (plugin, standalone), au type de licence et d'autres caractéristiques comprenant le type de code, les plateformes compatibles, l'interface Jena native, etc.

Après avoir décrit l'architecture des raisonneurs raisonneurs Pellet, RACER, FaCT++, Snorocket SWRL-IQ, ELK, HermiT, CEL et TrOWL (Abburu, 2012) en dresse une comparaison selon les caractéristiques définies par (Dentler, 2011).

(Matenzoglu et *al*., 2016) réalise une comparaison de 35 raisonneurs dont la plupart ont été développé entre 2010 et 2015. Elle est fondée sur des critères d'utilisabilité (services de raisonnement supporté, les niveaux d'expressivité et la complétude du raisonneur) et les algorithmes de raisonnement implémentés.

3.2.4.4 Le modèle pour les outils de raisonnement

Le modèle des outils de raisonnement a été établi à partir du tableau comparatif présenté par (Alaya, 2016) qui est fondé sur l'article (Matenzoglu et al., 2016). Les raisonneurs décrits sont actuellement FaCT++ , Pellet , ELK , HermiT , Konclude , Euler/Eye et NoHR .

3.2.5 Outils de conversion de données en OWL

La finalité de ces outils est la conversion de données contenues dans des feuilles de calcul en OWL.

3.2.5.1 Les scénarios

Scénario 1 : L'utilisateur dispose de documents existant sur la connaissance à formaliser, le plus souvent sous la forme de listes ou de feuilles de calcul. Il souhaite pouvoir les greffer facilement sur le squelette de son ontologie en utilisant les relations qu'il a prédéfinies.

Scénario 2 : L'utilisateur souhaite importer des éléments de hiérarchie à partir d'ontologies existantes avec cependant la possibilité de les amender ou de les complèter.

Scénario 3 : L'utilisateur souhaite mettre à la disposition d'intervenants extérieurs des parties de son travail en cours à des fins de vérification ou de validation.

Scénario 4 : L'utilisateur souhaite introduire dans l'ontologie d'importantes quantités de faits ou de caractéristiques sans avoir la charge de les éditer tous manuellement.

3.2.5.2 Les questions de compétence

Question 1 : L'outil permet-il l'import de données exprimées dans des tableaux (feuilles de calcul) ?

Question 2 : L'outil permet-il l'opération inverse (export vers des feuilles de calcul) d'extraits sélectionnés de son ontologie ?

Question 3 : Ces fonctionnalités sont-elles disponibles pour une utilisation en mode « ligne de commande » pour supporter un certain degré d'automatisation des tâches répétitives lors de l'enrichissement de son ontologie ?

3.2.5.3 Les outils de conversion de données

Nous avons sélectionné l'outil Cellfie un plugin Protégé 5 intégrant MappingMaster (O'Connor M. J., 2010) et l'outil FOE (Despres & Guezennec, 2017). Populous et WebPopulous offrent des fonctionnalités similaires ainsi que l'option d'import Excel disponible dans Topbraid Composer .

3.2.5.4 Le modèle pour les outils de conversion de données

Pour construire le modèle des outils de conversion de données, nous avons le type et d'export et les modalités d'utilisation, le degré d'automatisation autorisé et les constructions qu'il est possible de créer.

3.2.6 Outils d'alignement

Pour construire le modèle des outils d'alignement nous avons commencé à tester les outils répertoriés sur le site de AI3[12], la librairie des outils d'alignement hébergée par l'INRIA[13] et le site du projet Ontobeep[14]. Nous inventorions également les outils de visualisation d'alignement tels que (Lanzenberger et *al*., 2011). Cette partie du travail est en cours et ne sera pas présenté dans l'article.

3.2.7 Outil de gestion d'évolution

Pour construire le modèle des outils d'évolution nous avons commencé à inventorier les outils existants. Nous travaillons à partir des travaux récents de (Lambrix et *al*., 2016) qui réalisent une synthèse de ces outils du point de vue de la visualisation.

Les outils que nous analysons sont CODEX (Hartung et al., 2012), REX (Christen et *al*., 2015), NeonToolkit (Palma et *al*., 2012).

3.2.8 Outils de visualisation

La visualisation d'ontologies est une tâche qui aide à la compréhension des modèles représentés. Elle est utile à l'ensemble des acteurs participant à sa construction. Or, il n'est pas simple de créer une visualisation des ontologies en raison de la complexité de ces ressources. Les entités à visualiser sont la hiérarchie des concepts, les relations entre les concepts, les attributs des concepts et les individus relevant des concepts représentés (Katifori, 2007). Nous avons identifié deux publications faisant références à des ontologies VO formalisant les connaissances utiles la visualisation d'ontologies (Shu, 2007) et (M. Voigt & J. Polowinski, 2011) qui décrivent les primitives utilisées dans le domaine de la visualisation. Elles ne sont pas directement exploitables mais nous avons réutilisé certaines des classes modélisées.

3.2.8.1 Les scénarios

Scénario 1 : L'utilisateur souhaite visualiser la hiérarchie des concepts.

Scénario 2 : L'utilisateur souhaite se centrer sur un concept afin de visualiser globalement l'ensemble des liens qu'il entretient avec le réseau de concepts.

Scénario 3 : L'utilisateur souhaite visualiser les explications fournies par le raisonner et disposer d'une représentation graphique.

3.2.8.2 Les questions de compétence

Question 1 : Quel outil permet de visualiser l'ontologie ?

Question 2 : Quelles sont les entités visualisable de l'ontologie ?

Question 3 : Quelles sont les actions possibles pour visualiser l'ontologie ?

3.2.8.3 Les outils de visualisation

Les outils décrits dans la version actuelle de oogo sont GraphViz, CropCircles, KC-Viz, Knoocks, LogDiffViz, OntoGraph, OntologyVisualizer, Jambalaya, OntoTGVizTab, VOWL, WebVOWL, Protupos.

3.2.8.4 Le modèle pour les outils de visualisation

Le modèle des outils de visualisation a été établi à partir de la comparaison et l'évaluation des visualisations d'ontologies présentées par (Balzer et al., 2015) et des publications relatives à certains outils (Kuhar & Podgorelec, 2012), (Lohmann et al., 2016).

[12] http://www.mkbergman.com/1769/50-ontology-mapping-and-alignment-tools/

[13] http://alignapi.gforge.inria.fr/impl.html

[14] http://www.ontobee.org/tutorial/ontobeep#features

Dans le cadre de la construction du service web de visualisation centré utilisateur Protupos, (Despres &Nobecourt, 2018) ont caractérisé les outils de visualisation actuellement utilisés :

- Graphviz est un plugin de Protégé qui s'exécute dans son propre onglet. Il sert principalement à la visualisation graphique d'ontologie grâce à l'utilisation d'un graphe de nœuds et d'arcs qu'il est possible d'étendre et de réduire dynamiquement via des actions sur des boutons. Sur le principe, Graphviz est très utile pour parcourir la hiérarchie et la manipuler (expansion, respectivement réduction, des propriétés relatives à un nœud). Il permet également la visualisation de la hiérarchie des classes inférées.
- *VOWL* est un plugin de Protégé qui s'exécute dans son propre onglet et qui ne prend pas en compte les actions effectuées par ailleurs. Par exemple, il n'est pas possible de se centrer sur un concept et de naviguer autour de ce dernier dans l'onglet VOWL. Sur le principe, VOWL est une spécification de notations graphiques pour les ontologies écrites en OWL http://purl.org/vowl/spec/. L'outil est clairement tourné vers l'ontologue.
- WebOWL (http://visualdataweb.de/webvowl/) est un service web implémentant VOWL indépendamment de Protégé. Les actions peuvent être effectuées grâce aux boutons et aux menus. Un inconvénient majeur de WebOWL est que comme pour tout service web, l'ontologie doit être téléchargée.
- CropCircles est un outil intégré à SWOOP qui permet de visualiser la hiérarchie des concepts. Un concept est représenté par un cercle, les fils de ce concept sont représentés à l'intérieur de ce cercle et la hiérarchie est visualisé selon un représentation en coupe. Il est possible de zoomer sur une couche particulière de la hiérarchie en cliquant dans le cercle et de naviguer dans la hiérarchie en cliquant à l'extérieur du cercle. La vision en coupe par niveau hiérarchique permet notamment de bien appréhender la densité d'un sous-arbre par rapport à un autre.

Le développement de Protupos a permis de définir des fonctionnalités que nous avons utilisées pour construire le modèle des outils de visualisation :

- visualisation interactive de la hiérarchie : utilisation d'une représentation circulaire zoomable par actions de l'utilisateur ;
- de la hiérarchie par coupe de niveau ;
- visualisation de chaîne de propriétés centrée sur un nœud avec la possibilité de développer dynamiquement la construction de la chaîne ;
- visualisation sous la forme de réseaux d'un type de propriété ;
- visualisation centrée sur un concept en utilisant une fonction « élastique (permettant de modifier le focus) » ;
- nuage de tags associés à un concept.

4. Présentation de oogo et évaluation

oogo a été éditée avec le logiciel Protégé (version 5.2.0)[15]. L'ontologie et la description de ses ressources et propriétés seront publiées selon les principes des données liées sur le web et le schéma sera identifié par l'URI HTTP https://www-limics.smbh.univ-paris13.fr/oogo#. Plusieurs vocabulaires sont utilisés pour la construction du modèle outil : dcterms, doap, dvia, foaf, org. Dans cette version de oogo, nous ne pouvons pas réutiliser dvia:url car le type anyUri n'est pas directement supporté par Protégé 5.

[15] https://protege.stanford.edu/

4.1 Métrique de oogo

Metrics

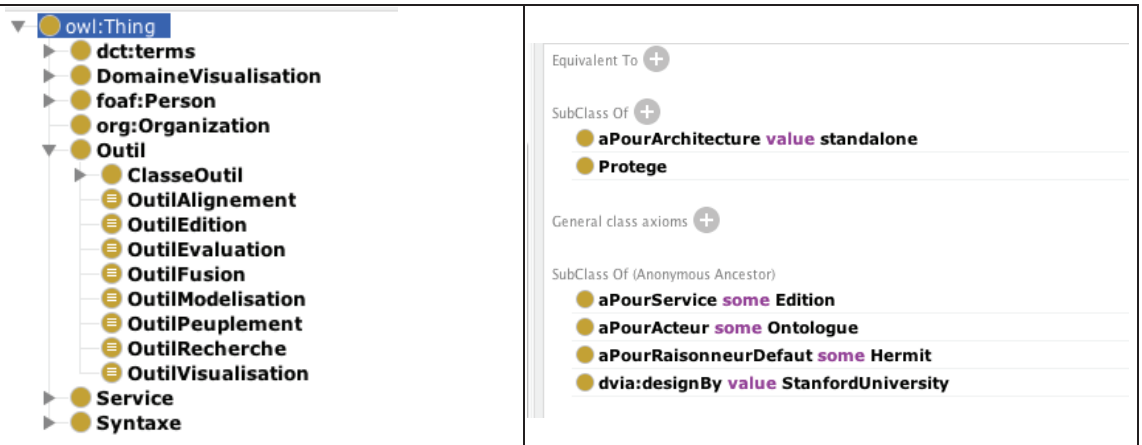
Axiom	756
Logical axiom count	368
Declaration axioms count	208
Class count	155
Object property count	17
Data property count	12
Individual count	23
Annotation Property count	6
DL expressivity	ALCHO(D)

FIGURE 3 – Métrique de l'ontologie oogo

La métrique de l'ontologie est présentée Figure 3. Elle comporte actuellement 155 classes, 17 object property et 12 data property. Elle est en cours d'évolution.

4.2 Aperçu des classes de oogo

Nous présentons une description du concept Outil dans oogo. Dans la classe ClasseOutil l'ensemble des outils étudiés sont décrits comme des sous-classes. Par exemple, la classe Protégé comporte les sous-classes Proétég3-x, Protégé4-x, Protégé5-x. Des individus relèvent de ces différentes sous-classes. Elle est également définie comme disposant d'un service d'édition. Les classes définies, comme OutilEdition de la figure 4(A), permettent d'utiliser le raisonneur pour caractériser les outils selon les services qui leur correspondent.



(A) (B)

Figure 4 – Vue globale de oogo (A) et Définition d'un outil (B)

4.3 Requêtes DL Query

Nous montrons par le biais de quelques exemples de requêtes DL Query que les questions de compétence trouvent une réponse.

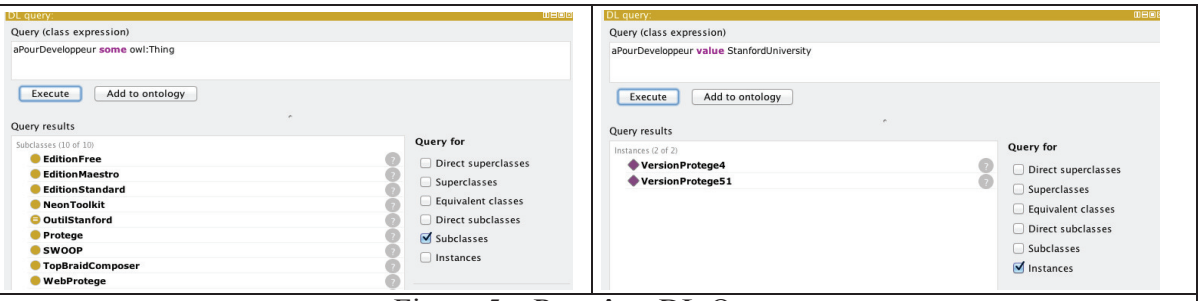


Figure 5 – Requêtes DL Query

Conclusion

Cet article présente un travail en cours qui répondait à une demande qui nous a permis de prendre conscience de la nécessité de centraliser l'expérience de la communauté de manière synthétique afin de répondre simplement et rapidement à des questions pratiques concernant la disponibilité des différents outils utilisables dans le cadre de la construction d'ontologies. L'ontologie ébauchée dans cet article est par nature destinée à évoluer régulièrement une fois qu'elle aura été mise en ligne dans sa version initiale. Actuellement, la langue utilsée pour le développement de oogo est le français néanmoins l'ensemble des entités sont décrites avec des labels et altlabel en anglais. Nous avons le projet de rendre sa construction collaborative lorsqu'elle sera publiée afin de s'assurer qu'elle ne devienne pas obsolète. De cette façon, les retours d'expérience pourront être valorisés et bénéficier à l'ensemble de la communauté.

Références

ABBURU, S. (2012). A Survey on Ontology Reasoners and Comparison. International Journal of Computer Applications, 57(17):33–39.

ABBURU, S., BABU, G. S. (2013). Survey on Ontology Construction Tools. International Journal of Scientific & Engineering Research, Volume 4, Issue 6, pp.1748-1752.

ALATRISH E. S. (2013). Comparison Some of Ontology Editors, in Management Information Systems, vol 8, n°2, pp. 018-024, 2013.

ALAYA N. (2016). Managing The Empirical Hardness of The Ontology Reasoning using The Predictive Modelling. Thèse de l'université de Tunis El Manar et de l'université Paris8.

BALZER L., DO M.T., MASELUK D. (2015). Comparison and Evaluations of Ontology Visualizations. Rapport de recherche, H.5.2, I.2.4.

CHRISTEN V., GROSS V., HARTUNG M. Region Evolution eXplorer a tool for discovering evolution trends in ontology regions. Journal of Biomedical Semantics, 6:Article 26, 2015.

D'AQUIN M., MOTTA E. (2011). Watson, more than a Semantic Web search engine. Semantic Web 2(1): 55-63.

DENTLER K., CORNET R., & ten TEIJE A., & de KEIZER N. (2011). Comparison of reasoners for large ontologies in the OWL 2 EL profile. Semant. web, 2(2):71–87.

DESPRES S., GUEZENNEC G. - Flat OWL Editor : un outil utilisant des feuilles de calcul pour découpler les tâches des acteurs impliqués dans la gestion d'une ontologie. Toth2017, (à paraître).

DESPRES S., NOBECOURT J.- Quelles fonctionnalités pour un outil de visualisation d'ontologie ? Atelier VIF Visualisation d'informations, Interaction, et Fouille de données EGC 2018 (http://gt-vif.polytech.univ- nantes.fr/egc-vif2018/atelier_VIF_EGC2018.pdf

FININ T., DING L., PA R., JOSHI A., KOLARI P. (2005). Java A., Peng Y. Swoogle: Searching for knowledge on the semantic web. In Anthony Cohn, editor, Proceedings of the 20th National Conference on Artificial Intelligence - Volume 4, AAAI'05, pages 1682–1683. AAAI Press.

GARCIA-BARRIOCANAL E., SICILIA M. A. & SANCHEZ-ALONSO S. (2006). Usability Evaluation of Ontology Editors. Knowledge Organization 32(1),1-9.

HARTUNG M., GROSS A., RAHM E. CODEX: Exploration of semantic changes between ontology versions. Bioinformatics, 26(6):895–896, 2012.

HEON M., NKAMBOU R., LANGHEIT C. (2016). Toward G-OWL: A graphical, polymorphic and typed syntax for building formal OWL2 ontologies. Proceedings of the 25th International Conference Companion on World Wide Web. International World Wide Web Conferences Steering Committee.

JAIN V., PRASAD S.V.A.V. (2016). Evaluation and Validation of ontology using Protégé Tool. In International Journal of Research in Engineering & Technology (IMPACT: IJRET) ISSN (E): 2321-8843; ISSN (P): 2347-4599 Vol. 4, Issue 5, May 2016, 1-12.

KAPOOR, B., SHARMA, S.A. (2010). Comparative Study Ontology Building Tools for Semantic Web Applications. International journal of Web & Semantic Technology (IJWesT), 1(3).

KATIFORI A., HALATSIS C. LEPOURAS G., VASSILAKIS C., GIANNOPOULOU E. (2007). Ontology Visualization Methods – A Survey. ACM Computing Surveys, Vol. 39, N°4, Article 10.

LAMBRIX P., DRAGISIC Z., IVANOVA V., ANSLOW C. Visualization for Ontology Evolution, VOILA 2016 Visualization and Interaction for Ontologies and Linked Data, 2016, pp.54-67.

LANZENBERGER M., SAMPSON J. (2011). Ontology Alignment Quality: A Framework and Tool for Validation. IJISMD 2(3): 1-23

LOHMANN S., LINK V., MARBACH E. & NEGRU S. (2016). Visualizing ontologies with VOWL.

7(4),21. Already accepted papers for the EKAW 2014 special issue, extended version.

MATENTZOGLU N., Leo J., Hudhra V., SATTLER U., & Parsia B.. (2015). A survey of current, stand-alone OWL reasoners. In Proceedings of the 4th International Workshop on OWL Reasoner Evaluation, pages 68–79.

NASKAR D., DUTTA B. (2016) - Ontology Libraries: A Study from an Ontofier and an Ontologist Perspectives. 19th International Symposium on Electronic Theses and Dissertations, Lille.

O'CONNOR M. J., HALASCHEK-WIENER C., MUSEN M. A. (2010). Mapping master: a flexible approach for mapping spreadsheets to OWL. In ISWC'10 Proceed-ings of the 9th international semantic web conference on The semantic web - Volume Part II, 194-208.

PALMA R., ZABLITH F., HAASE P., CORCHO O. Ontology evolution. In MC Suarez-Figuero, A Gomez-Prez, E Motta, and A Gangemi, editors, Ontology Engineering in a Networked World, pages 235–255. 2012.

PARVEEN, KUMAR SAHNI D.K., KHURANA D., NANDAL R. (2016). Ontology Development Tools and Languages: A Review. Vol. 5, Issue 6, June, 2016

POVEDA-VILLALON M., M.C., GOMEZ-PEREZ A, POVEDA-VILLALON M. (2014). OOPS! (OntOlogy Pitfall Scanner!): An On-line Tool for Ontology Evaluation. Int. J. Semantic Web Inf. Syst. 10(2): 7-34.

RICHARD M. (2017). Apports de la modélisation ontologique pour le partage des connaissances en psychiatrie. Thèse de l'université Pierre et Marie Curie - Paris VI.

SABOU M., D'AQUIN M., MOTTA E. (2008). SCARLET: SemantiC RelAtion DiscoveRy by Harvesting OnLinE OnTologies. ESWC 2008: 854-858.

SEONGWOOK Y., ANCHIT A., PREETHAM C., PAAVANY J., ASHISH M. and SHIKHA S. (2009). Survey about Ontology Development Tools for Ontology-based Knowledge Management, University of Southern California.

SLIMANI, T. (2015). Ontology Development: A Comparing Study on Tools, Languages and Formalisms. In Indian Journal of Science and Technology, Vol 8(24), doi:10.17485/ijst/2015/v8i34/54249.

STADTMÜLLER S., HARTH A., GROBELNIK M. (2013). Accessing information about linked data vocabularies with vocab.cc. In Juanzi Li, Guilin Qi, Dongyan Zhao, Wolfgang Nejdl, and Hai-Tao Zheng, editors, Semantic Web and Web Science, Springer Proceedings in Complexity, pages 391–396. Springer New York. doi:10.1007/978-1-4614-6880-6,34.

SHU G., AVIS N.J. (2008). Bringing semantic to vizualization services. Advances in Engineering Software 39. Pp.514-520.

STEIGMILLER A., LIEBIG T., & BIRTE. G. (2014).Konclude: System description. Web Semantics: Science, Services and Agents on the World Wide Web, 27(1).

STOJANOVIC L., MOTIK B. (2002) Ontology Evolution within Ontology Editors. In Conference on the Evaluation of Ontology-based Tools.

STUDER R., BENJAMIN V. R et FENSEL. (1998). D. Knowledge Engineering: Principles and methods. Data & Knowledge Engineering 25, pp. 161-197.

SUAREZ-FIGUEROA M.C., GOMEZ-PEREZ A., FERNANDEZ-LOPEZ M. (2015). The NeOn Methodology framework: A scenario-based methodology for ontology development. Applied Ontology 10(2): 107-145.

SUAREZ-FIGUEROA M.C., GOMEZ-PEREZ A. (2008). Towards a Glossary of Activities in the Ontology Engineering Field, LREC.

TSARKOV D. & HORROCKS I. (2006). Fact++ description logic reasoner: System description. In Proceedings of the Third International Joint Conference on Automated Reasoning, pages 292–297.

TUDORACHE T., NYULAS C., NOY, N. F. & MUSEN, M. A. (2013). Webprotégé : A collaborative ontology editor and knowledge acquisition tool for the web. Semantic web, 4(1) :89–99. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3691821/.

USCHOLD M., GRUNINGER M. (1996). Ontologies: Principles, methods and applications. The knowledge engineering review, vol. 11, no 2, p. 93–136.

VANDENBUSSCHE P.Y. & VATANT B. (2014). Linked Open Vocabularies. ERCIM news, 96:21–22.

VANDENBUSSCHE P.Y., ATEMEZING G., POVEDA-VILLALON M., VATANT B. (2017). Linked Open Vocabularies (LOV): A gateway to reusable semantic vocabularies on the Web. Semantic Web 8(3): 437-452.

VOIGT M. & POLOWINSKI J. (2011). Towards a Unifying Visualization Ontology. Technical Report – Technische Universität Dresden (3/2011).