TENET, un outil pour construire des ontologies OWL à partir de textes en langue naturelle

David Rouquet*, Aurélien Lamercerie**, Valérie Bellynck***
Christian Boitet***, Vincent Berment****, Guillaume de Malézieux***,****

*Tétras Libre
**UGA, LIG, GETALP, Grenoble INP
***UGA, LIG, GETALP
****CS Group

Résumé. Cet article présente la démonstration de TENET, un outil générique et Open Source pour construire des ontologies OWL à partir de textes écrits en langue naturelle. Nous illustrons les différentes étapes de la chaîne de traitement, notamment le passage par une représentation sémantique pivot des énoncés (sérialisée en RDF), ainsi que le paramétrage et la réalisation du processus d'extraction et de construction de l'ontologie cible. Enfin, nous montrons comment l'ontologie cible peut être exploitée, dans un contexte industriel, avec un exemple de vérification automatique d'exigences système.

1 Introduction

Cet article présente TENET [1], une plateforme générique pour construire des ontologies OWL à partir de textes écrits en langue naturelle (LN). Le code source est disponible sous licence CeCILL-B dans un dépôt Gitlab dédié [2]. Cette application est basée sur les standards du Web Sémantique du W3C [3] (RDF, OWL, SPARQL, SHACL). Elle implémente une *enconversion* des textes en UNL [4], une représentation sémantique pivot et un procédé d'extraction fondé sur le concept de transduction sémantique compositionnelle (Lamercerie (2021)).

Le développement de ce prototype s'inscrit dans le cadre du projet RAPID UNSEL [5], financé par la DGA [6] de 2019 à 2021. Le but du projet est de fournir des outils pour accompagner la spécification de systèmes complexes (par exemple, un système de communications sol-air pour un aéroport ou un système de freinage d'urgence). Précisément, nous développons des processus permettant de raisonner sur un ensemble de spécifications, afin de détecter des incomplétudes ou des incohérences. Les énoncés sont traités en suivant une approche multilingue et interactive qui s'appuie sur une représentation pivot interlingue.

1. TENET : Tool for Extraction using Net Extension by (semantic) Transduction
2. `https://gitlab.tetras-libre.fr/unl/tenet`
3. `https://www.w3.org/standards/semanticweb/`
4. `http://www.unlweb.net/wiki/images/a/ab/Spec33.pdf`
5. UNseL : Universal Networking system engineering Language
6. Direction Générale de l'Armement, `https://www.defense.gouv.fr/dga`

Dans cette optique, nous cherchons à construire automatiquement un système formel de type "ontologie métier" ou "ontologie de domaine", implémenté dans le langage logique OWL [7]. Nous visons une représentation du sens de chaque exigence, ainsi que des liens structurels et séquentiels entre exigences, de façon aussi précise, exacte et complète que possible.

Notre réponse prend la forme d'une chaîne de traitement utilisant une représentation pivot, UNL, dans le cadre du projet UNSEL. Dans un premier temps, un analyseur linguistique, développé avec l'environnement Ariane-H [8], permet de construire des graphes UNL correspondant aux énoncés traités. Le processus d'extraction, illustré par la figure1, s'applique ensuite. Les règles d'extraction, notées CTS, sont exprimées sous la forme de requêtes SPARQL. Partant d'une ontologie cadre donnant les grandes classes attendues, elles permettent de construire l'ontologie cible en suivant une démarche compositonnelle. L'étape de vérification permet ensuite de repérer des erreurs dans les énoncés et de produire des messages d'alerte. En complément, l'application WebVOWL [9], qui implémente une notation visuelle pour OWL (Lohmann et al. (2016)), donne une visualisation interactive des ontologies.

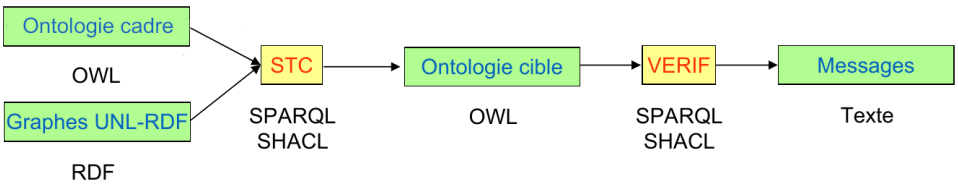

FIG. 1 – *Architecture du processus d'extraction.*

La suite détaille notre proposition. La section 2 décrit la structure pivot utilisée, puis le concept d'ontologie cadre est précisé dans la section 3. Le processus d'extraction est ensuite détaillée dans la section 4, et un cas d'usage est donné dans la section 5.

2 Structure pivot UNL-RDF

L'extracteur TENET requiert l'usage d'une structure sémantique intermédiaire entre les énoncés LN et le traitement principal. Le langage UNL [10], utilisé dans le cadre du projet UNSEL, permet d'utiliser des représentations précises et indépendantes des langues.

On représente le sens d'un énoncé (français par exemple) sous forme d'un (hyper)-graphe UNL, qui est une structure sémantique abstraite d'un énoncé anglais équivalent (voir figure 2). Les symboles de relations et d'attributs sémantiques proviennent de l'anglais, ainsi que les "lexèmes interlingues" (UW), de sorte que les graphes UNL sont compréhensibles et constructibles par tous les chercheurs et développeurs du monde.

7. W3C OWL Working Group
8. `https://linguarium.org`
9. http ://vowl.visualdataweb.org/webvowl.html
10. UNL Specification 3.3 (2004)

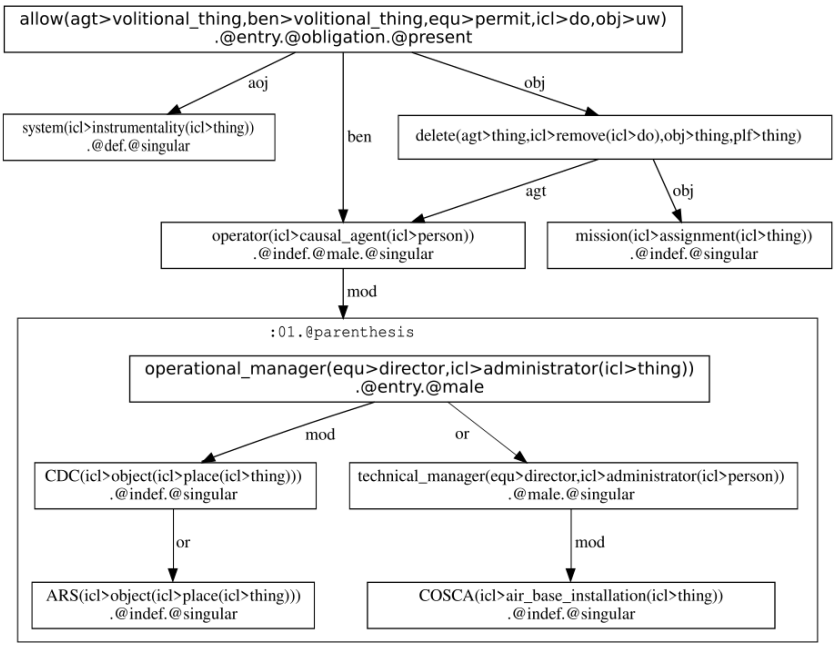

FIG. 2 – *Graphe UNL pour "Le système doit permettre à un opérateur (gestionnaire opérationnel d'un CDC ou ARS, ou gestionnaire technique COSCA) de supprimer une mission".*

Le lexique d'UNL est composé de lexèmes interlingues, appelés UW (Universal Words), et formés d'un mot-vedette suivi d'une liste éventuellement vide de restrictions. En général, le mot-vedette est un lemme anglais, simple ou composé, tandis que les restrictions associent une relation sémantique et un autre lexème sous la forme `relation>UW` ou `relation<UW`. Exemple : l'UW `land(icl>do, agt>person, obj>plane, plt>lake)` dénote "amerrir", ainsi que `land(icl>drive>do, agt>person, obj>aeroplane, plt>sea)`.

Un graphe UNL inclut des nœuds, simples ou généralisés, et des arcs. Un nœud simple porte un UW, issu du lexique, et une liste d'attributs sémantico-pragmatiques. Un nœud généralisé, nommé scope, est constitué d'un sous-graphe connexe par arcs dont les arcs portent le même numéro de scope. Les arcs sont orientés et portent une relation sémantique, comme agent (`agt`), objet (`obj`), bénéficiaire (`ben`), localisation (`plc`), durée (`dur`), destination (`plt`), possesseur (`pos`), etc.

La transformation des énoncés LN en graphes UNL est réalisée à l'aide d'un analyseur syntaxico-sémantique (enconvertisseur), développé avec l'environnement Ariane-H [11]. Une représentation structurale complète, dont la correction est garantie par construction et interaction, permet d'aboutir aux graphes UNL. Ces derniers sont ensuite sérialisés en RDF [12], ce qui permet de fonder entièrement la suite du traitement sur les standards du Web sémantique du W3C.

11. `https://linguarium.org`
12. Rouquet et al. (2020)

3 Ontologies cadres et graines d'extraction

La notion d'ontologie cadre permet de préciser l'interprétation attendue des énoncés traités. Précisément, une ontologie cadre, telle que montrée par la figure 3, définit les grandes classes d'objets du contexte métier visé, ainsi que des *graines d'extraction* permettant l'initialisation du traitement.

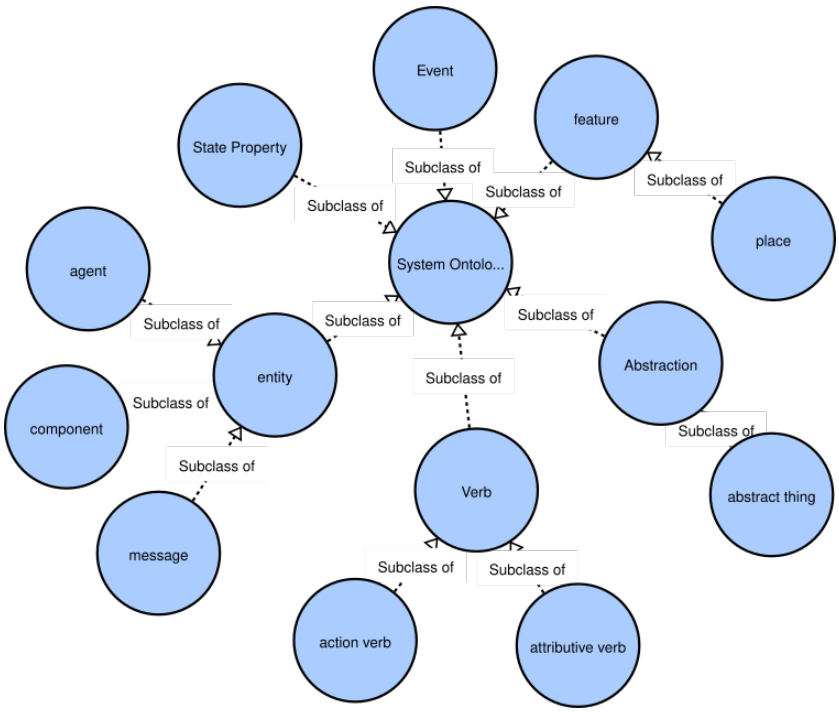

FIG. 3 – *Extrait d'une ontologie cadre définissant un point de vue "système". Elle inclut des entités (agents, composants ou messages), des caractéristiques (places), des événements, des propriétés d'état, des verbes (actions ou attributs) et des abstractions.*

Dans la première version, les graines sont spécifiées au niveau de chaque classe sous la forme d'un ensemble de restrictions d'UW, permettant d'identifier les concepts de base pour chaque classe. Il est également possible d'utiliser plusieurs ontologies pour décrire des facettes différentes des énoncés traités. Ainsi, un même UW, par exemple `operator(icl>human)`, peut signifier un composant, un agent ou une entité externe, en fonction de l'ontologie cadre utilisée.

4 Extraction de contenu avec des règles SPARQL

Le processus d'extraction implémente le concept de transduction sémantique (Lamercerie (2021)), qui consiste à appliquer une série de transformations sur un graphe sémantique (UNL). Une règle d'extraction produit un *filet* à partir d'une instance de sa partie gauche, qui est un schéma décrivant une partie du graphe initial et des filets préalablement construits. Le filet produit est ajouté à la structure de travail.

Ce processus prend un ensemble de graphes UNL-RDF en entrée, et une ontologie cadre en paramètre. La sortie est un ensemble de triplets RDF-OWL enrichissant et instanciant l'ontologie cadre. La figure 4 montre un extrait d'une ontologie construite automatiquement sur un corpus d'une quarantaine d'exigences, en s'appuyant sur l'ontologie cadre de la figure3.

FIG. 4 – *Extrait de l'ontologie construite sur un corpus d'exigences système.*

L'analyse de la structure pivot (UNL) est guidée à l'aide de schémas de transduction compositionnelle (notés *STC*). Implémentés sous la forme de requêtes SPARQL-construct, les STC sont intégrés dans un graphe de contraintes (*shapes*) respectant la spécification SHACL-

SPARQL [13]. Ils sont organisés en niveaux d'application, grâce au mécanisme `sh:order` [14]. Ces règles dépendent fortement de la structure des graphes sémantiques en entrée, c'est à dire du formalisme UNL et des spécificités du corpus (syntaxe, registre, style, parfois phraséologie). En revanche, elles sont génériques du point de vue du contenu métier des phrases, qui est seulement précisé au niveau de l'ontologie cadre.

En complément, un mode d'exécution à quatre niveaux est adopté pour le mise en œuvre du processus : (1) l'**extraction des éléments atomiques**, (2) la **formation d'éléments composites** par un procédé récursif, (3) l'**extraction des propriétés et relations** pour les éléments atomiques et composites, (4) la **construction de l'ontologie cible**. Le typage des filets permet d'ajuster le traitement pour en assurer l'efficacité et la terminaison. La complexité est linéaire dans nos observations expérimentales : précisément, nous avons un temps de calcul moyen de l'ordre d'un quart de seconde par exigence sur un ordinateur de bureau moderne. Toutes les requêtes (STC) utilisées sont accessibles dans le dépôt git (`https://gitlab.tetras-libre.fr/unl/tenet`).

5 Utilisation pour la vérification d'exigences système

L'analyse de l'ontologie obtenue permet la vérification d'un ensemble de points à contrôler, portant sur la sémantique des énoncés et la signification de l'ensemble des exigences d'un document de spécification ou d'une documentation structurée. Cette étape exploite également des règles SPARQL, ainsi que des raisonneurs logique généraux comme Pellet [15] ou FACT++ [16]. Elle génère des messages d'alerte et des suggestions, illustrés sur la figure 5.

Notre prototype permet ainsi de vérifier les points suivants :

— **Concepts sous-spécifiés** dans les exigences, par exemple pour la mention *gestionnaire opérationnel* dans une exigence, parle-t-on bien de toutes les sous-classes comme *gestionnaires opérationnels* d'un *CDC*, d'un *ARS*, etc. ?

— **Défaut terminologique**, par exemple si la seule sous-classe de *mission*, extraite dans toutes les exigences, est *mission COSCA*, c'est soit que l'on a deux termes pour identifier le même concept, soit que l'on a sous-qualifié *mission* dans certaines exigences.

— **Qualification homogène des éléments en relation**, par exemple si une exigence mentionne qu'un *gestionnaire COSCA peut supprimer une mission COSCA* et qu'une autre indique qu'un *opérateur d'un CDC peut créer une mission*, on pourra proposer de compléter *mission* par *CDC* dans la seconde exigence. Pour cela, on pourra utiliser la résolution d'équations analogiques de type `a : b :: c : x` (Lepage (2003)).

Ce premier jeu de vérifications est loin d'être exhaustif. Il peut être enrichi en ajoutant des règles SPARQL suivant le modèle utilisé. D'autres vérifications de consistance entre les exigences peuvent être réalisées à l'aide de raisonneurs logiques génériques. Le rapport Rouquet et al. (2020) détaille un exemple de ce type, portant sur deux exigences : (1) *"Une voie radio peut prendre deux états : écoute et veille."* et (2) *"L'opérateur place la voie radio dans l'état trafic."*

13. `https://www.w3.org/TR/shacl/#dfn-shacl-sparql`
14. `https://www.w3.org/TR/shacl/#order`
15. `https://github.com/stardog-union/pellet`
16. `http://owl.cs.manchester.ac.uk/tools/fact/`

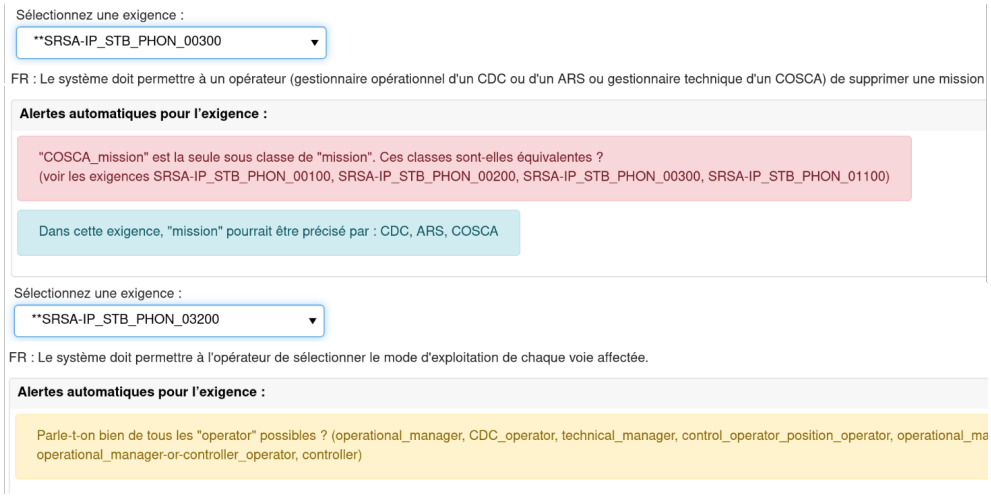

FIG. 5 – *Exemples d'alertes dérivées de l'ontologie construite sur un ensemble d'exigences.*

Une ontologie incohérente est produite avec ces deux exigences, car *trafic* ne fait pas partie de l'ensemble {*écoute*, *veille*} des états possibles pour une *voie radio*.

L'application du processus sur 40 exigences d'un corpus pilote a permis de produire près de 2000 triplets RDF-OWL complétant l'ontologie cadre. Ces triplets incluent 33 classes organisées en hiérarchie et reliées par des relations ensemblistes, dont 12 classes d'agents et de 21 composants, et 14 propriétés liant les classes précédentes. Les informations extraites ont permis de détecter 100% des anomalies repérées manuellement sur le corpus. De plus, l'examen du graphe OWL produit est directement instructif pour un humain et fournit des informations denses et pertinentes pour la compréhension du système. Par exemple, dans la figure 4, la notion de mission est bien explicitée comme une entité contenant des voies radio. Les différents types de mission sont extraits, ainsi que les types d'agents qui peuvent les créer, les supprimer ou les modifier.

6 Perspectives

Cet article expose l'application concrète d'une chaîne de traitement, partant d'exigences système exprimées en langue naturelle, pour aboutir à un système formel représenté dans le langage OWL. La démarche est cependant générique, et pourrait s'appliquer à d'autres problématiques, telles que la recherche d'informations dans un document, la mise en œuvre d'un système expert ou l'intégration automatique de procédures.

Nos expérimentations s'inscrivent dans le domaine de l'ingénierie des exigences. Elles ont permis de montrer qu'il est possible de construire des axiomes OWL significatifs qui supportent des raisonnements non triviaux. Il devient ainsi envisageable de gérer la proximité sémantique des termes, de vérifier la cohérence structurelle des entités décrites dans les exigences ou de mettre en évidence des incohérences sur les propriétés définies.

L'usage des graphes UNL permet de réduire la dépendance monolingue des logiciels envisagés, tandis que le processus de désambiguïsation interactive apporte une garantie de sens sur les représentations pivots exploitées lors de l'extraction. En effet, un des problèmes des approches "à l'état de l'art" est qu'on suppose que le parseur vers une représentation sémantique interlingue, comme UNL ou AMR, fournit automatiquement un résultat "correct", ou plutôt "fidèle", c'est à dire correspondant à ce qu'a voulu dire le rédacteur. Or, même avec une phase de désambiguïsation automatique s'appuyant sur une ontologie et des calculs de scores ou de préférences, les résultat sont en pratique très souvent incorrects. Les problèmes principaux sont causés par les ambiguïtés, qu'elles soient lexicales, d'attachement ou de dépendance sémantiques. L'étape de désambiguïsation interactive proposée apporte une réponse à ces problèmes, réponse renforcée par le choix d'UNL. Ce formalisme est moins connu, mais nettement plus expressif et précis qu'AMR, notamment pour la partie lexico-sémantique.

L'analyse par transduction sémantique se révèle être un procédé simple, traçable et adaptable permettant d'interpréter les énoncés et de construire les ontologies visées. Il n'y a pas de freins pour le passage à l'échelle autre que la disponibilité des graphes UNL et les contraintes techniques liées au logiciel support (maquette permettant de faire le lien avec un système opérationnel de gestion des exigences, comme DOORS d'IBM ou SBoCS de C-S).

Références

Lamercerie, A. (2021). *Principe de transduction sémantique pour l'application de théories d'interfaces sur des documents de spécification*. Thèse, Université Rennes 1 ; Rennes 1.

Lepage, Y. (2003). *De l'analogie rendant compte de la commutation en linguistique*. Habilitation à diriger des recherches, Université Joseph-Fourier - Grenoble I.

Lohmann, S., S. Negru, F. Haag, et T. Ertl (2016). Visualizing ontologies with VOWL. *Semantic Web 7*(4), 399–419.

Rouquet, D., V. Bellynck, V. Berment, et C. Boitet (2020). Natural language representation and content extraction using RDF, SHACL and the universal networking language (UNL).

UNL Specification 3.3 (2004). http ://www.unlweb.net/wiki/images/a/ab/Spec33.pdf.

W3C OWL Working Group. OWL-2 Overview (http ://www.w3.org/TR/owl2-overview/).

Summary

This paper presents the demonstration of TENET, a generic and open source tool for building OWL ontologies from texts written in natural language. We illustrate the different steps of the processing chain, including: Going through a pivot semantic representation of the utterances (serialized in RDF), as well as setting up and performing the extraction and construction process of the target ontology. Finally, we show how the target ontology can be exploited, in an industrial context, with an example of automatic verification of system requirements.