Analyse d'exigences systèmes dans le projet UNseL

Aurélien Lamercerie, David Rouquet, Valérie Bellynck, Christian Boitet, Vincent Berment

Analyse d'exigences systèmes dans le projet UNSEL

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

* G-INP/LIG/GETALP, ** Tétras-Libre, *** UGA/LIG/GETALP, **** CS Group

Résumé. Cet article décrit l'application d'une méthode d'extraction de contenu sémantique dans un contexte industriel, avec pour objectif la vérification automatique d'exigences systèmes rédigées en langue naturelle. L'étape d'extraction utilise une analyse par transduction sémantique, implémentée en s'appuyant sur les standards du Web Sémantique du W3C. Une représentation linguistique des textes, sous forme de graphes UNL (Universal Networking Language), est exploitée pour fournir une structure semi-formelle et indépendante de la langue source. Les outils développés construisent ensuite une ontologie OWL à partir des spécifications du système, exprimées par des énoncés non contraints. Finalement, une vérification automatique des exigences est réalisée à l'aide de règles SPARQL génériques et de raisonneurs logiques. La fin de l'article montre une mise en pratique sur des exigences issues de documents réels.

1 Introduction

Le projet RAPID UNSEL [1], financé par la DGA [2] de 2019 à 2021, vise à fournir des outils pour accompagner la spécification de "systèmes de systèmes" (par exemple, un système de communications sol-air pour un aéroport ou un système de freinage d'urgence). Un tel système est construit à partir d'un cahier des charges, puis de documents de plus en plus détaillés : document de spécifications externes ou internes, document d'analyse générale ou détaillée, documents de programmation. Une spécification est un ensemble structuré d'exigences, qui sont majoritairement des énoncés descriptifs ou prescriptifs en langue naturelle (LN).

Les problèmes liés aux spécifications sont bien connus [3]. Ce sont *l'incohérence, l'incomplétude* et *l'inadéquation*. Ainsi, il arrive qu'il y ait des exigences contradictoires, que l'ensemble des exigences ne contienne pas tout ce qui est nécessaire ou que des exigences ne soient pas comprises par leurs lecteurs. Le coût d'une erreur s'avère parfois considérable. Ces problèmes, dont l'enjeu est important, sont pourtant peu ou mal traités au niveau opérationnel.

Par ailleurs, dans l'idée d'une application à des systèmes critiques, l'usage de représentations à sens garantie s'est ajouté au contrainte du projet. La revue de l'état de l'art, par exemple Kamath et Das (2019), n'a pas permis de mettre en évidence l'existence d'approche satisfaisante pour répondre à ce besoin. Nos travaux portent une approche originale en conséquence.

1. UNseL : Universal Networking system engineering Language
2. Direction Générale de l'Armement, `https://www.defense.gouv.fr/dga`
3. voir par exemple Dick et al. (2017)

Ainsi, la vérification d'anomalies peut s'appuyer sur la construction d'un système formel de type "ontologie métier" ou "ontologie de domaine". Dans cette optique, il est nécessaire de représenter le sens de chaque exigence, ainsi que leurs liens structurels et séquentiels, de façon aussi précise, exacte et complète que possible. Dans le projet UNSEL, nous cherchons à implémenter les ontologies de domaine dans le langage logique OWL [4].

Il est compliqué, voire impossible (Kay (2017)), d'obtenir automatiquement une représentation linguistique non ambiguë, pour chaque exigence, telle que l'on puisse en dériver le sens dans une ontologie. Aucun analyseur disponible ne le fait, pour aucune langue. Une raison majeure justifiant cette observation est que les énoncés en LN sont presque toujours ambigus, même si on utilise un langage restreint. Pour garantir l'adéquation entre une exigence et sa représentation, le projet UNSEL implémente notamment un système de désambiguïsation interactive permettant de voir qu'un passage est ambigu [5].

Une fois que les exigences sont représentées au niveau linguistique de façon correctement désambiguïsée, dans notre cas grâce à des (hyper)graphes UNL, il est nécessaire de les traduire dans l'ontologie considérée. C'est en effet seulement à ce niveau que les calculs de consistance et de complétude peuvent se faire. Pour répondre à cet enjeu, une technique d'extraction de sens, basée sur le concept de transduction sémantique (Lamercerie (2021)), a été adaptée aux besoins du projet.

La suite de cet article détaille la mise en œuvre de ce procédé. Partant de graphes UNL (section 2), le processus d'extraction de contenu (section 3) est appliqué pour la vérification automatique d'un corpus d'exigences systèmes (section 4). Les outils présentés sont disponibles sous licence CeCILL-B dans un dépôt Gitlab dédié [6].

2 Représentation du sens des textes avec UNL

La mise en œuvre d'une approche d'extraction par transduction sémantique requiert une représentation linguistique des textes sous forme de graphe. Dans le cadre du projet UNSEL, celles-ci sont définies avec le langage UNL [7], ce qui permet d'avoir des représentations précises et indépendantes des langues.

2.1 Langage UNL

UNL est un projet, un langage de graphes sémantiques d'énoncés en langue naturelle et un format de documents multilingues. Le projet international UNL est lancé en décembre 1996 [8], avec initialement les 12 langues les plus parlées au monde, à l'initiative de l'Institute of Advanced Studies (IAS) de l'*Université des Nations Unies* [9] (UNU). Il a pour but la construction d'une infrastructure multilingue permettant à chacun d'accéder à des informations dans sa propre langue.

Le *langage UNL* représente le sens d'une phrase sous forme d'un (hyper)-graphe, comme illustré dans la figure 1. L'idée de base est de rendre les graphes sémantiques compréhensibles

4. W3C OWL Working Group (2012)
5. projet LIDIA, Blanchon (1994), projet Eureka Eurolang, Boitet et al. (1995)
6. `https://gitlab.tetras-libre.fr/unl/tenet`
7. UNL Specification 3.3 (2004)
8. Uchida et al. (1996)
9. `https://unu.edu/`

et constructibles par les chercheurs et développeurs du monde entier. En principe, tous les symboles utilisés dans un graphe UNL proviennent de l'anglais. Ils peuvent néanmoins, par un jeu des restrictions sémantiques, dénoter des acceptions n'existant pas en anglais (e.g. tatami, ou certains niveaux de politesse).

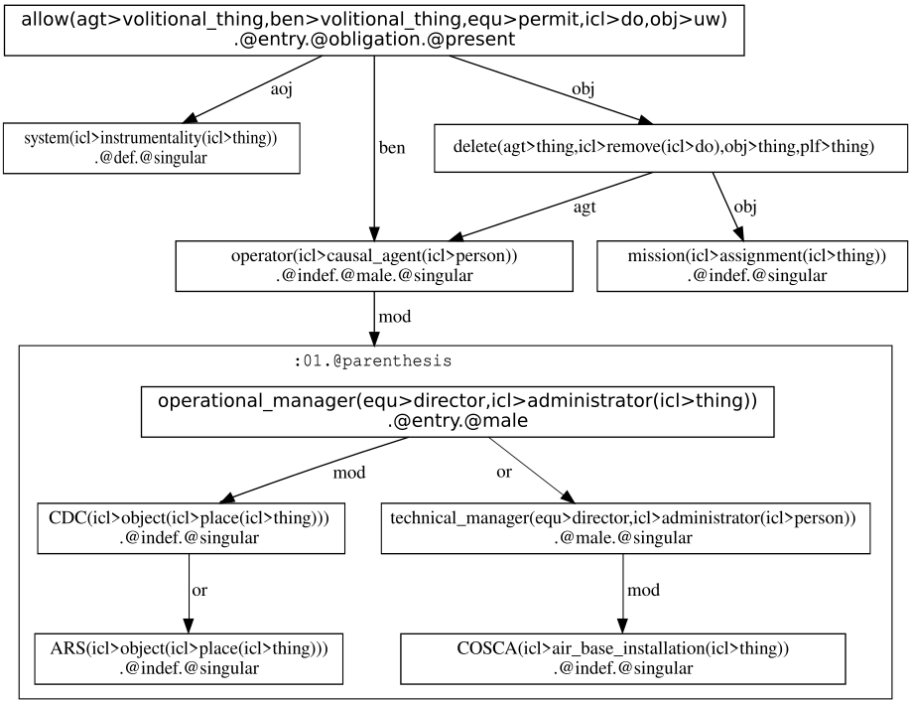

FIG. 1 – *Un graphe UNL possible pour l'exigence "Le système doit permettre à un opérateur (gestionnaire opérationnel d'un CDC ou d'un ARS ou gestionnaire technique d'un COSCA) de supprimer une mission".*

On convient aussi qu'un graphe UNL quelconque doit être la structure abstraite d'un énoncé anglais équivalent à celui d'origine. C'est important parce que les relations sémantiques portées par les arguments d'un prédicat linguistique anglais (verbe ou déverbal) ne sont assez souvent pas les mêmes que celles portées par les arguments correspondants d'un prédicat français (ou autre), bien que le répertoire des relations sémantiques soit universel. On dit souvent qu'UNL est un *pivot anglo-sémantique*.

Un (hyper-)graphe UNL est formé de nœuds, d'arcs orientés et d'attributs booléens, ces différents éléments portant des informations de natures différentes.

Ainsi, un nœud peut être élémentaire ou composé :

— Un nœud élémentaire contient un UW (*Universal Word*, ou *lexème interlingue*) et des attributs booléens sémantiques ou pragmatiques. Un UW est formé d'un mot-vedette (*headword*) anglais [10], et d'une liste de restrictions, le tout dénotant une acception.

10. ou emprunté par l'anglais à une autre langue, comme *tatami*

— Un nœud composé constitue un sous-graphe, appelé *scope*, représentant une partie de la phrase. Le scope i est constitué de tous les arcs de numéro de scope i et des nœuds qu'ils relient. Un arc ne peut appartenir qu'à un scope, mais un nœud élémentaire peut être commun à plusieurs scopes. Enfin, un scope doit être connexe par arcs, si on néglige l'orientation des arcs.

Sur l'exemple de la figure 1, l'UW `operator(icl>causal_agent(icl>person))` dénote un *operateur*, tandis que le verbe *supprimer* est représenté par l'UW `delete(agt>thing, icl>remove(icl>do), obj>thing, plf>thing)`.

Les arcs définissent des relations entre les noeuds. Ils sont orientés et étiquetés par des relations sémantiques binaires :

— Les relations sont notées par des abréviations à 3 lettres (e.g. agent (`agt`), objet (`obj`), bénéficiaire (`ben`), localisation (`plc`), durée (`dur`), destination (`plt`), possesseur (`pos`), etc.).

— Le scope est identifié par un numéro (:01, :02...). Le scope principal, noté :00, est optionnel.

Les attributs booléens sont préfixés par ".@", et attachés aux nœuds (élémentaires ou composés). Ils apportent des précisions sur les noeuds du graphe, telles que :

— le rôle dans le graphe (e.g. *.@entry* indique le nœud principal d'un scope, d'où on doit partir pour interpréter le graphe) ;

— des informations sémantiques et pragmatiques, par exemple le temps abstrait (*time* par opposition à *tense* ;

— d'autres éléments de communication comme les actes de parole, le niveau de politesse, le nombre (singulier, pluriel, collectif), la détermination (défini, indéfini), etc.

Par exemple, l'utilisation de l'imparfait en français, ou du progressif passé en anglais, pour dénoter une habitude passée, peut être représenté à ce niveau sémantique par l'attribut *.@past.@repetition*.

2.2 Enconversion et sérialisation des graphes UNL

La transformation d'un texte en graphes UNL (un par phrase) s'appelle l'enconversion, l'opération inverse la déconversion. Notre enconvertisseur est constitué d'un analyseur structural (syntaxique et sémantique) développé avec l'environnement Ariane-H [11] et permettant de garantir le sens analysé. En présence d'ambiguïtés, cet analyseur produit tous les sens possibles et pose des questions à l'utilisateur pour obtenir le sens correct. La représentation structurale obtenue est, par construction, une image complète et fidèle du texte analysé. Cette représentation est ensuite convertie en graphes UNL.

Afin d'obtenir une chaîne d'extraction entièrement fondée sur les standards du Web sémantique du W3C, nous utilisons une sérialisation RDF des graphes UNL (Rouquet et al. (2020)). Un convertisseur du format standard UNL vers la sérialisation dite *UNL-RDF* a été développé dans le cadre du projet. Il est disponible sous forme d'exécutable Java [12] et de service Web [13].

11. `https://linguarium.org`
12. `https://gitlab.tetras-libre.fr/unl/unlTools`
13. `https://unl.demo.tetras-libre.fr/`

3 Extraction de contenu par transduction sémantique

Le processus d'extraction de contenu utilise une technique d'analyse par transduction sémantique. On part d'un graphe sémantique (graphe UNL dans le projet), que l'on transforme et enrichit pour aboutir à une nouvelle structure formelle. Cette étape permet la construction automatique d'une ontologie OWL à partir d'une ontologie cadre passée en paramètre. L'implémentation est fondée sur les standards du Web sémantique (RDF, OWL, SPARQL, SHACL) [14].

3.1 Analyse par transduction sémantique.

Le principe d'analyse par transduction sémantique (Lamercerie (2021)) a été validé par une première expérimentation sur des graphes AMR (Abstract Meaning Representation, Banarescu et al. (2013)). Nous proposons d'adapter ce cadre pour une application sur des graphes [15] UNL, en reprenant les notions de filets sémantiques et de schémas de transduction compositionnels.

Filets sémantiques. La définition 1 décrit un objet mathématique s'appliquant sur un graphe étiqueté. Considérant l'ensemble S des sommets d'un graphe, un filet sémantique s'applique à l'ensemble des parties de S. Ce concept permet de caractériser des ensembles de sommets avec des attributs particuliers (type et valeurs).

Définition 1 *Soit $\mathcal{G}$ un graphe étiqueté et S l'ensemble des sommets de G. Soit $\mathcal{T}$ un ensemble de types, et $\mathcal{V}$ un ensemble de valeurs. Un filet sémantique sur $\mathcal{G}$ est une structure $f = (x, \tau, v)$ telle que $x \subseteq S$ est une partie de S, $\tau \in \mathcal{T}$ est un type et $v \subseteq \mathcal{V}$ est un ensemble de valeurs.*

Le type d'un filet définit la nature des valeurs qui lui sont associées, et les opérations qui lui seront appliquées. L'ensemble des types $\mathcal{T}$ est supposé muni d'une relation d'ordre, permettant d'établir plusieurs niveaux d'analyse. Notre implémentation inclut le type *atome*, caractérisant des filets *atomiques* ne couvrant qu'un seul nœud, le type *composite*, définissant des filets *composite* associant un concept à plusieurs noœuds, ou encore le type *list* qui caractérise une liste d'éléments.

La notion de filet sémantique est illustrée par la figure 2, qui montre plusieurs filets reliés à des ensembles de nœuds. Sur ce graphe, les filets F_a et F_b couvrent respectivement le nœud *1* (filet F_a), et les nœuds *2* et *4* (filet F_b), tandis que le filet F_{ab} couvrent les nœuds *1*, *2* et *4*. Nous verrons plus loin que le filet F_{ab} peut être obtenu par composition des filets F_a et F_b. Ces structures sont également typées, et associées à différentes valeurs utiles à leur interprétation et composition.

Schémas de transduction compositionnels (STC). La définition 2 spécifie une structure associant une formule logique et un opérateur de transduction. Elle s'applique aux filets d'un graphe sémantique, et permet l'obtention de nouveaux filets par composition. Dans cette optique, nous considérons les propriétés des filets, dérivées des types et des valeurs. L'ensemble des propriétés considérées est noté $\mathcal{P}$.

14. W3C Standards (2021)

15. Il ne s'agit pas des graphes classiques en théorie des graphes, où on ne peut avoir plus d'un arc allant d'un sommet s_i à un sommet s_j (dans un graphe orienté), ou plus d'un arc reliant un sommet s_i à un sommet s_j (dans un graphe non orienté). Il s'agit plutôt de *réseaux*. Les *graphes de transition* d'automates et les *graphes sémantiques* en sont des exemples.

Définition 2 *Soit P un ensemble de prédicat. Soit $\mathcal{R}_\mathcal{P}$ un ensemble de conjonctions d'expressions $p(f_1, \ldots, f_n)$, avec $p \in P$. Un schéma de transduction compositionnel (STC) est une paire (ϕ, tr) telle que :*

- *ϕ est une formule logique sur $\mathcal{R}_\mathcal{P}$;*
- *tr est une application de $f_i = (x_i, \tau_i, f_i)$ vers $(\bigcup x_i, \psi_\tau(\tau_1, .., \tau_n), \psi_v(v_1, .., v_n))$, où ψ_τ et ψ_v sont deux fonctions retournant respectivement un type et un ensemble de valeurs.*

Comme nous le verrons plus loin, les schémas de transduction se traduisent naturellement sous la forme de requêtes SPARQL-update. La figure 2 donne un exemple d'application d'un schéma donné, noté $s = (\phi, tr)$, composant un filet *atomique* et un filet de type *liste*. La partie requête du schéma est définie par $\phi = atom(f_1) \wedge listOf - atom(f_2) \wedge mod(f_1, f_2)$. Elle permet de sélectionner les filets à composer (le filet F_a de type $atom$ et le filet F_b de type $listOf - atom$, ces deux filets étant liés par la propriété mod). œLe filet résultant F_{ab} est obtenu par application de l'opérateur $tr = (\bigcup x_i, \psi_\tau(f_i), \psi_v(f_i))$ sur les filets sélectionnés. Les fonctions ψ_τ et ψ_v sont définies pour préciser le type (ici, $listOf - concept$) et les valeurs associés au nouveau filet (objets en relation définissant une hiérarchie de concepts).

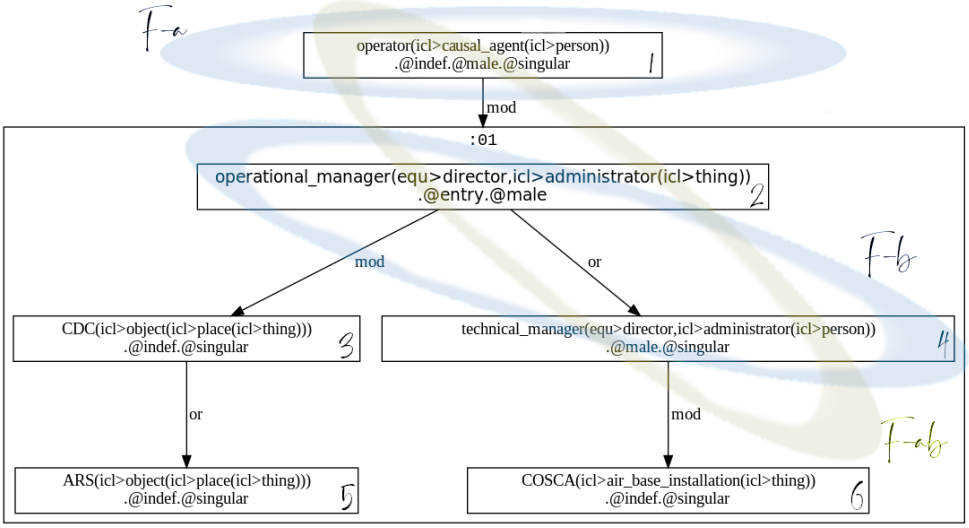

FIG. 2 – *Graphe UNL-RDF portant sur la désignation d'un opérateur, avec quelques filets.*

Application des STC. La mise en œuvre du processus d'analyse implique l'élaboration d'un cadre de calcul organisant l'application des STC. L'objectif est d'enrichir le graphe UNL-RDF en prenant en compte son contenu et sa structure. Nous avons adopté un mode d'exécution sur quatre niveaux : (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. Un exemple d'implémentation est donné en section 3.2, avec quelques précisions complémentaires.

Les STC de niveau 1 s'appliquent sur les sommets du graphe UNL-RDF. Ils permettent d'initialiser le traitement en générant des filets *atomiques*. Comme nous le verrons en section 3.2, l'extraction des éléments atomiques est paramétrée dans une ontologie précisant le résultat attendu (notion d'ontologie *cadre*).

Les STC de niveau 2 suivent un procédé récursif, contrôlé en s'appuyant sur le typage des filets. Cette étape permet de construire des filets *composites*, et d'obtenir ainsi une hiérarchie de concepts dans l'ontologie cible. La figure 2 est une illustration de ce procédé : le filet obtenu désigne une liste d'éléments composites (liste de classes) avec une hiérarchie entre la classe parente *operator* et les classes filles *operational manager*, *technical manager*.

Les STC de niveau 3 opèrent de manière similaire pour détecter des propriétés sur les éléments, atomiques ou composites, et des relations entre ces éléments. Concrètement, tandis que les STC des niveaux précédents produisent des filets rattachées à des classe d'éléments (par exemple, des agents, des messages, des actions), les STC de niveau 3 traduisent les relations entre ces filets en propriétés sur les classes. Les filets produits à ce stade sont de type *property*. Ils sont centrés sur les verbes, traduisant l'attribution d'une propriété ou la qualification d'une action, et rattachés à plusieurs classes d'éléments. Sur l'exemple présenté, la classe d'*operator* est reliée à l'action *delete*, avec pour objet la classe de *message*.

Finalement, les STC de niveau 4 génèrent l'ontologie attendue à partir des filets générés. Le typage des filets oriente directement la construction de nouvelles classes, instances et propriétés pour mettre à jour la structure cible.

3.2 Implémentation.

Cette section présente une implémentation des STC, entièrement basée sur les standards du Web sémantique. Elle est disponible en Open Source sous l'acronyme TENET (*Tool for Extraction using Net Extension by (semantic) Transduction*) [16].

Entrées et paramètres. Le processus d'extraction prend en entrée un ensemble de graphes UNL, dans leur sérialisation RDF. Il prend par ailleurs en paramètre une ontologie cadre, qui définit les grandes classes d'objets visés selon le contexte métier, ainsi que des *graines d'extraction* permettant d'initialiser les STC de niveau 1.

Processus d'extraction à base de STC. Les STC sont implémentés sous la forme de requêtes SPARQL-construct qui s'appliquent aux graphes UNL-RDF. Les requêtes sont intégrées dans un graphe de contraintes (*shapes*) respectant la spécification SHACL-SPARQL [17]. Elles sont ordonnées en niveaux d'application, conformément à ce qui est présenté dans la section 3.1, grâce au mécanisme *sh :order* [18].

Les règles SPARQL sont fortement dépendantes de la structure des graphes sémantiques en entrée, elle même dépendant principalement du formalisme UNL et de la phraséologie du

16. https://gitlab.tetras-libre.fr/unl/tenet
17. https://www.w3.org/TR/shacl/#dfn-shacl-sparql
18. https://www.w3.org/TR/shacl/#order

corpus. Par contre, les règles sont génériques du point de vue du contenu métier des phrases. En effet, ce dernier est paramétré par l'ontologie cadre. Ainsi, des règles développées sur la base de notre corpus, décrivant un système de communication sol-air, restent a priori valables pour des spécifications de métiers très différents, comme un système de freinage d'urgence.

Sorties. La sortie est un ensemble de triplets RDF-OWL enrichissant et instanciant une ontologie cadre passée en paramètre. Il est tout à fait possible que le processus d'extraction vise plusieurs ontologies décrivant des facettes différentes du système.

4 Application pour la vérification d'un corpus d'exigences

Nous décrivons dans cette section l'application de nos méthodes sur un corpus d'exigences système réel fourni par la DGA (*SRSA-IP*). Ce corpus est composé de 367 exigences décrivant un système de communication sol-air. Il n'est pas publiquement accessible. Les premières applications ont été réalisées sur un corpus pilote de 40 exigences, sélectionnées dans SRSA-IP pour leur complexité linguistique et leur représentativité du corpus.

Nous présentons d'abord une extraction et des vérifications réalisées sur l'exigence de la figure 1, puis les résultats obtenus sur le corpus pilote (les graphes UNL n'étant pas encore disponibles pour le reste du corpus).

4.1 Ontologie cadre et règles d'extraction

La figure 3 présente un aperçu de l'ontologie cadre utilisée dans les premières expérimentations. Elle contient essentiellement une dizaine de classes accompagnées de graines d'extraction. Les requêtes SPARQL qui composent les STC de niveau 1 récupèrent les graines dans l'ontologie cadre, pour identifier les éléments atomiques dans les exigences. Actuellement, les graines sont basées sur les restrictions des UW. Par exemple, dans la figure 3, on voit que la classe *Agent* sera initialisée avec toutes les UW portant les restrictions *icl>administrator*, *icl>person* ou *icl>human*. Nous envisageons d'autres méthodes, permettant de définir ces graines à partir d'exemples ou de récupérer des sous-graphes et pas simplement des UW.

Nous avons défini 25 règles SPARQL pour l'extraction. Partant des éléments atomiques identifiés par les graines, les règles parcourent ensuite les (listes de) modificateurs (*mod* en UNL), précisant les éléments atomiques, pour aboutir à une hiérarchie d'éléments composites dans l'ontologie. D'autres règles extraient la liste ouverte des relations que les éléments composites peuvent entretenir entre eux, par exemples les actions des *Agents* sur les *Composants*. Enfin, les classes et relations de l'ontologie sont instanciées. Ce mécanisme permet en particulier d'assurer la traçabilité de ce qui est extrait, par exemple que l'exigence *STB_PHON_300* mentionne un certain opérateur qui peut supprimer une mission particulière.

4.2 Exemple d'extraction et de vérification sur une exigence du corpus

L'application des règles d'extraction SPARQL sur le graphe UNL-RDF de la figure 1 produit les sorties suivantes :

— 289 triplets RDF intermédiaires dans le processus de transduction,

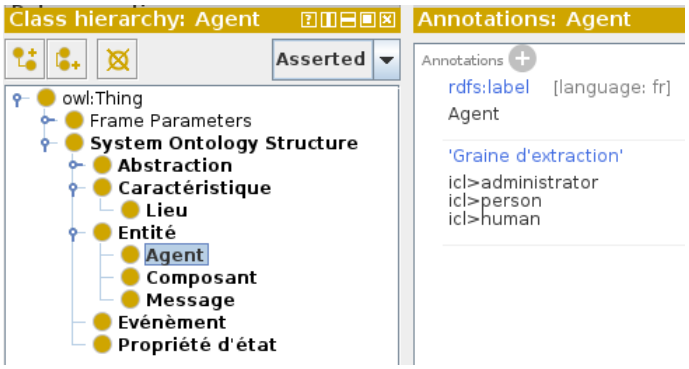

FIG. 3 – *Exemple d'ontologie cadre*

— 119 triplets ajoutés à l'ontologie cadre du système.

La figure 4 illustre les informations extraites :

— en particulier une hiérarchie des agents mentionnés, en haut à gauche (en jaune),

— une classe définie comme union de deux autres, en haut à droite (en jaune),

— une *propriété d'événement* dont le domaine et le but sont précisés, au milieu à droite (en bleu), et

— l'assertion suivante : une instance d'opérateur "*delete*" une instance de mission, en bas (en violet).

Le contenu RDF-OWL extrait permet de réaliser des vérifications, également implémentées sous la forme de règles SPARQL. Des messages d'alerte et des suggestions sont retournées à l'utilisateur après le contrôle des points suivants :

— **Sous-spécifications** dans les exigences, par exemple pour la mention *gestionnaires opérationnel* dans une exigence, parle-t-on bien de toutes les sous-classes comme *gestionnaires opérationnel* 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 *gestionaire 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.

D'autres vérifications de consistance entre les exigences peuvent être réalisées à l'aide de raisonneurs logique génériques. Le rapport Rouquet et al. (2020) détaille un exemple de ce type, dont on donne une version simplifiée ici :

1. *Une voie radio peut prendre deux états : écoute et veille.*
2. *L'opérateur place la voie radio dans l'état trafic.*

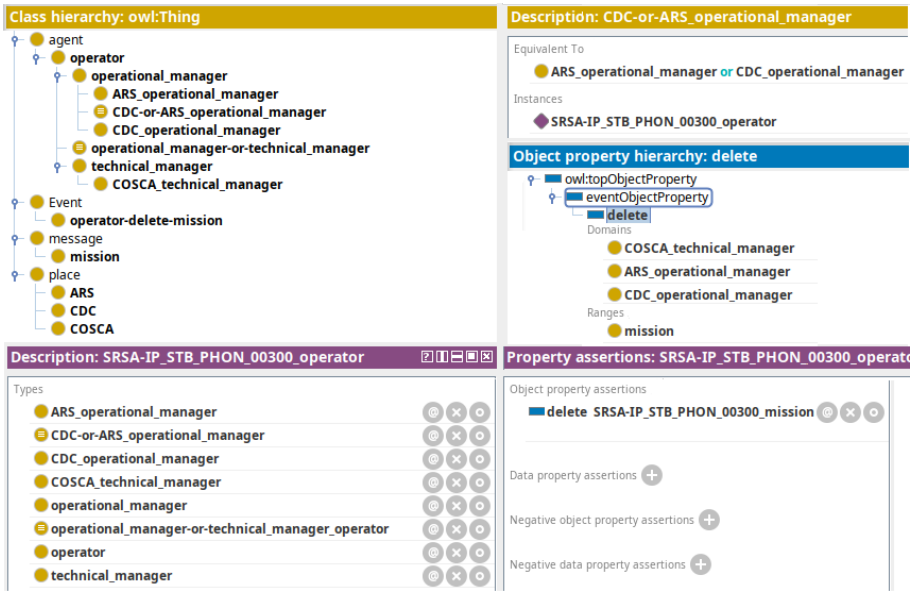

FIG. 4 – *Aperçu du résultat d'extraction sur l'exigence de la figure 1*

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

4.3 Résultats obtenus sur le corpus pilote

L'application des règles d'extraction SPARQL sur les 40 exigences du corpus pilote produit les sorties suivantes :

— 4990 triplets RDF intermédiaires dans le processus de transduction,

— 1930 triplets ajoutés à l'ontologie cadre du système.

Parmi les informations ajoutées à l'ontologie, on trouve en particulier :

— 33 classes organisées en hiérarchie et reliées par des relations ensemblistes dont 12 classes d'agents et 21 composants physiques ou "abstraits",

— 14 propriétés, liant les classes précédentes, dont 5 correspondant à des actions, des événements ou des états du système (*create*, *delete*, *release*, *set_up*, *include*).

Les mesures classiques de précision et de rappel restent à produire sur l'ensemble du corpus, mais les résultats sur le corpus pilote sont plus qu'encourageant. Les informations extraites ont permis de détecter 100% des anomalies pointées manuellement sur le corpus. On note toutefois des aspects pas ou mal pris en compte dans l'extraction et qui laissent une bonne marge de progression, par exemple les modalités déontiques et temporelles.

Malgré ces limites, l'examen du graphe sémantique OWL produit est directement instructif pour un humain et fournit des informations denses et pertinentes pour la compréhension du

système. Par exemple 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 modifier leurs voies radio.

Les résultats des extractions pilotes sont accessibles sur le Gitlab du projet [19].

5 Conclusion

Nous avons présenté, dans cet article, l'application concrète d'une chaîne de traitement globale, partant d'exigences système exprimées en langue naturelles (LN) pour aboutir à une ontologie OWL complète du système décrit par les exigences. Cette chaîne intègre plusieurs étapes : (1) l'enconversion des énoncés LN dans le format standard UNL, (2) la sérialisation RDF des graphes UNL, (3) l'extraction du contenu sémantique pour construire une ontologie OWL du système et (4) la vérification automatique de l'ontologie produite.

L'expérimentation réalisée a permis de montrer qu'il est possible de construire des axiomes OWL significatifs qui supportent un raisonnement non trivial. 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. L'analyse par transduction sémantique forme un procédé simple, traçable et adaptable pour l'interprétation des énoncés et la construction des ontologies visées.

Références

Banarescu, L., C. Bonial, S. Cai, M. Georgescu, K. Griffitt, U. Hermjakob, K. Knight, P. Koehn, M. Palmer, et N. Schneider (2013). Abstract meaning representation for sembanking. In Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, pp. 178–186. Sofia, Bulgaria : Association for Computational Linguistics.

Blanchon, H. (1994). LIDIA-1 : une première maquette vers la TA Interactive "pour tous". Thèse, Université Joseph-Fourier - Grenoble I.

Boitet, C., E. Planas, H. Blanchon, É. Blanc, J.-P. Guilbaud, P. Guillaume, M. Lafourcade, et G. Sérasset (1995). LIDIA-1.2, une maquette de TAO personnelle multicible, utilisant la messagerie usuelle, la désambiguïsation interactive et la rétrotraduction.

Dick, J., E. Hull, et K. Jackson (2017). Requirements Engineering. Springer International Publishing.

Kamath, A. et R. Das (2019). A survey on semantic parsing. In Automated Knowledge Base Construction (AKBC).

Kay, M. (2017). Translation : Linguistic and Philosophical Perspectives, Volume 221 of CSLI Lecture Notes. CSLI Publications.

19. `https://gitlab.tetras-libre.fr/unl/tenet`

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

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).

Uchida, H., M. Zhu, et T. Della Senta (1996). Unl : Universal networking language–an electronic language for communication, understanding, and collaboration. Tokyo : UNU/IAS/UNL Center.

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

W3C OWL Working Group (2012). OWL-2 Overview (http ://www.w3.org/TR/owl2-overview/).

W3C Standards (2021). Semantic web (https ://www.w3.org/standards/semanticweb/).

Summary

This paper presents the application of a semantic content extraction method in an industrial context, with the objective of automatic verification of system requirements written in natural language. The extraction step uses a semantic transduction analysis, implemented using the W3C Semantic Web standards. A linguistic representation of the texts, in the form of UNL (Universal Networking Language) graphs, is exploited to provide a semi-formal structure independent of the source language. The developed tools then build an OWL ontology from the system specifications, expressed by unconstrained statements. Finally, an automatic verification of the requirements is performed using generic SPARQL rules and logical reasoners. The end of the article describes a practical implementation on requirements from real documents.