L'Intelligence Artificielle pour l'accompagnement du raisonnement expert, la solution Khresterion

F. Soualah-Alila[1], M. Ben Ellefi[1], D. Pierre[1]

[1] Pôle R&D Khresterion
59 rue des Petits-Champs, 75001 Paris

Résumé

Les assureurs santé font face à une énorme quantité de données mouvantes : le recours à l'automatisation de leur traitement est désormais indispensable. L'objectif de notre travail est de réduire les délais et les coûts d'accès à l'expertise : la gestion des diagnostics, l'accompagnement à la décision et la rédaction des documents conformes aux contraintes métier et juridiques. Notre approche consiste à proposer aux assureurs un outil, non seulement de gestion de leurs traitements, mais aussi de garantie de conformité de ces derniers. Dans cet article, nous présentons nos travaux sur la modélisation des connaissances du domaine de l'assurance santé et notre méthode de raisonnement automatique permettant de reproduire un raisonnement expert basé sur des règles du métier. Un exemple de scénario de raisonnement sur les expressions de garanties dans un tableau d'assurance santé est détaillé dans cet article.

Mots-clés

Graphe de connaissances, Assurance santé, ontologies, règles, raisonnement, validation, SHACL-SPARQL.

Abstract

Health insurers faces a huge amount of moving data : the need to automate their processes is now essential. The objective of our work is to reduce expertise time and costs : diagnostic management, decision support and drafting documents in accordance with business and legal constraints. Our approach consists on proposing to insurers a tool for managing their processing and guaranteeing their compliance. In this article we present our work on knowledge modeling in the domain of health insurance, as well as our automatic reasoning process to reproduce expert reasoning based on facts and buisness rules. An example of a reasoning scenario based on guarantees expressions applied to a health insurance table is detailed in this paper.

Keywords

Knowledge Graph, Health insurance, ontologies, rules, reasoning, validation, SHACL-SPARQL.

1 Introduction

L'Intelligence Articielle (IA) entretient des échanges fructueux avec les sciences cognitives, d'une part elle fournit de nouveaux repères, points de comparaison pour la compréhension de l'intelligence, d'autre part elle peut s'inspirer de ce que l'on sait du fonctionnement du cerveau et de la façon dont l'homme raisonne, même si rien ne dit que l'IA doit copier l'intelligence humaine dans toutes ses manières de procéder [4]. Dans un domaine métier spécifique, la spécification de la conceptualisation de ce domaine consiste à rendre l'expertise métier compréhensible par le système sous forme d'un graphe de connaissance, connu sous le nom de Knowledge Graph (KG), afin d'assurer une IA basée sur les connaissances (IA symbolique).

Khresterion [1] est spécialiste de deux thématiques importantes en Intelligence Artificielle : la représentation des connaissances et le raisonnement automatique.

Le processus de modélisation des connaissances expertes peut être représenté comme un processus assurant la traduction de la connaissance détenue par un expert vers une forme exploitable par un système informatique et appliqué à un domaine particulier. La formulation de cette connaissance consiste en l'élaboration continue d'un modèle qui se réalise au cours d'cycle de cinq étapes principales que sont la collecte, la formalisation, l'implémentation, la validation et la correction des connaissances (figure 1).

Les technologies du Web Sémantique implémentées en graphe de connaissances offrent une solution pour faciliter l'intégration et l'interopérabilité des données. On parle alors de modélisation en schémas ontologiques permettant de mettre en œuvre des mécanismes de raisonnements.

Dans notre cas, le modèle métier sous-jacent exploité est celui de la construction d'un ensemble d'ontologies, dont des ontologies du domaine de l'assurance santé, permettant la définition d'un vocabulaire adaptable et évolutif. Nos ontologies sont complétées par des règles transcrivant les bonnes pratiques du métier en modèle logique. Nous utilisons à cet effet le langage SHACL-SPARQL [2] qui consiste en une collaboration de deux mécanismes : (i) SHACL [3] (Shapes Constraint Language) qui permet de valider des graphes RDF [4] (Resource Description Framework) avec un ensemble de conditions (ii) et SPARQL [5] (SPARQL Protocol and RDF Query Language) un standard du W3C pour le requêtage avec un noyau de graph pattern matching.

1. https://khresterion.com/
2. https://www.w3.org/2014/data-shapes/wiki/Shacl-sparql
3. https://www.w3.org/TR/shacl/
4. https://www.w3.org/RDF/
5. https://www.w3.org/TR/sparql11-overview/

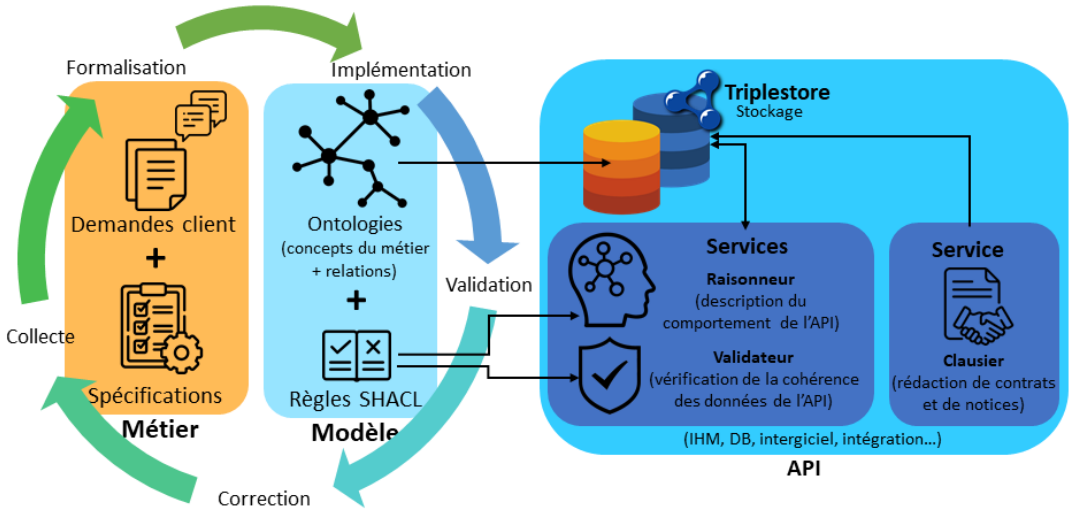

FIGURE 1 – Cycle de vie des modules Khresterion

Le choix de ce mécanisme de raisonnement se justifie par le fait que, parmi nos scénarios de raisonnement, il arrive qu'avec une seule règle le raisonneur doive inférer un gros volume de nouvelles instances (ou individus) à injecter dans le graphe de connaissances. Par exemple, dans notre système, le choix de la date d'effet d'une offre santé déclenche un appel au raisonneur. Le raisonneur infère alors en fonction de cette date de nombreuses instances qui correspondent aux différentes garanties santé de cette offre. La plupart des raisonneurs OWL-DL [6] (Ontology Web Language Description Logics) ne permettent pas la création de nouvelles instances dans la base de connaissances. Une exception est relevée par le langage de règles SWRL [7] (Semantic Web Rule Language) qui permet d'ajouter des fonctions (ou built-ins) dans son formalisme, notamment *swrlx :makeOWLThing*. À notre connaissance aucun raisonneur OWL-DL, à l'exception du raisonneur SWRLTab [8], n'exploite cette fonctionnalité SWRL. Par ailleurs, le raisonneur SWRLTab n'est pas utilisable dans notre cas car il ne permet pas d'ordonner l'exécution des règles, à l'inverse de SHACL avec l'utilisation de *sh :order*.

Dans la version actuelle du langage SHACL, la création des instances n'est pas encore implémentée directement, d'où le besoin d'augmenter ce dernier par du SPARQL CONSTRUCT. Les requêtes SPARQL CONSTRUCT permettent ainsi d'ajouter une flexibilité pour la création de nouvelles URIs et les injecter comme nouvelles instances dans le graphe de connaissances.

Le modèle métier est ensuite intégré dans notre moteur de raisonnement. Ce dernier permet d'exploiter nativement un mécanisme logique de subsomption, de gérer des règles générales et des exceptions pour des cas spécifiques, de proposer des choix multiples et d'explorer des alternatives.

Dans cet article, nous présentons le système de raisonnement Khresterion qui assure la **conformité** des garanties santé dans le cadre d'un contrat d'assurance. Notre moteur de raisonnement permet en plus, et à la différence des systèmes de raisonnements classiques, d'exploiter des formes logiques qui tolèrent la présence de contradictions [5]. Alors que les systèmes experts classiques n'interviennent que sur une logique décisionnelle pure, notre moteur de raisonnement demeure ainsi une assistance à la prise de décision et non un outil de raisonnement à contraintes.

L'article est organisé de la manière suivante : la section 2 introduit le graphe de connaissances de Khresterion et les modes de raisonnements, la section 3 présente une démonstration sous forme d'un scénario détaillé de raisonnement sur des expressions de garanties pour une assurance santé, enfin la section 4 présente une démonstration de règles de validation pour assurer la cohérence et la conformité du système avant de conclure.

2 Le graphe de connaissance de Khresterion

Un des atouts de la technologie Khresterion est d'accéder à toutes les sources existantes, de consolider les données pertinentes et de créer une seule base de connaissances référentielle : le graphe de connaissance de Khresterion, *KGK* (Knowledge Graph Khresterion).

Le Knowledge Graph, littéralement graphe de connaissances en français, est un terme officialisé par Google en 2012. La définition de ce terme reste controversée : un certain nombre de définitions générales ou bien techniques, parfois même contradictoires, ont émergé. L'article [1] liste et analyse les différentes définitions de ce terme. Nous adoptons la définition suivante, qui semble la plus adaptée à notre vision : un graphe de connaissances est une représentation de la connaissance relative à un domaine sous une forme facilement exploitable par la machine. Il est constitué d'entités et de relations entre ces entités, formant un graphe de connaissances stocké dans des bases de données de type graphe. Bien que cette représentation graphique de la connaissance ne soit pas récente, elle a gagné en popularité et est aujourd'hui un élément clé pour des applica-

6. https://www.w3.org/TR/owl2-primer/
7. https://www.w3.org/Submission/SWRL/
8. https://github.com/protegeproject/swrltab

tions d'Intelligence Artificielle liées à la recherche rapide et contextuelle d'information ainsi qu'à la prise de décision.
De cette définition se dégagent ces éléments justifiant notre choix pour ce modèle :

- des données facilement accessibles et compréhensibles car organisées selon des vues métiers,
- des ressources directement exploitables par les outils d'Intelligence Artificielle et de raisonnement logique pour inférer de nouvelles connaissances,
- une grande facilité d'enrichissement du graphe de connaissances même avec des données hétérogènes,
- une interopérabilité des données favorisée par la gestion d'identifiants stables et d'alignement avec des ressources externes,
- un environnement et des standards construits nativement sur les standards du Web.

Le modèle KGK se base essentiellement sur deux couches de l'architecture du Web Sémantique qui sont les couches Ontologie et Règles [2]. La construction de notre référentiel, consiste donc à définir un ensemble comportant ontologies, contraintes et actions adéquates à un usage décrit par des faits.

Ontologies Khresterion :
La première couche du référentiel correspond aux connaissances métier conceptualisées sous la forme d'ontologies. Une ontologie permet d'analyser un domaine de connaissances et de réutiliser ces connaissances. Une ontologie comprend un vocabulaire commun aux experts et une représentation des relations entre ces termes, qui définissent cette connaissance. Elle est utilisable par les humains, comme par les machines [7].
La figure 2 illustre un aperçu des différentes ontologies modélisées par Khresterion pour la représentation des connaissances du domaine de l'assurance santé. Toutes ces ontologies sont la propriété intellectuelle de Khresterion et ont été conçues en collaboration avec des clients assureurs, en nous basant sur des documentations internes et des spécifications fonctionnelles.
Nous distinguons trois niveaux d'ontologies :

- **Une ontologie générique :** elle regroupe des concepts, des relations et des données génériques, indépendants du domaine de l'assurance santé. Ces éléments décrivent des notions universelles ou des concepts généraux et abstraits tels que la modélisation du temps et de l'espace. Ces éléments sont applicables à plusieurs domaines, et dans notre cas exploités par les ontologies de domaine ci-dessous.
- **Des ontologies de domaine** : elles décrivent des connaissances bien spécifiques au domaine de l'assurance santé : description des garanties d'assurances, des établissements d'assurances, des bénéficiaires, des différents documents à éditer sous forme de contrats ou notices, etc. Elles peuvent se décliner en versions spécifiques à un pays afin d'intégrer des définitions locales.
- **Des ontologies d'applications :** elles caractérisent une conceptualisation encore plus spécifique que les ontologies de domaine pour un usage ou un client précis. Nous modélisons par exemple dans ces ontologies les coordonnées des directions commerciales d'un client en particulier ou encore des expressions de garanties particulières à un client, etc.

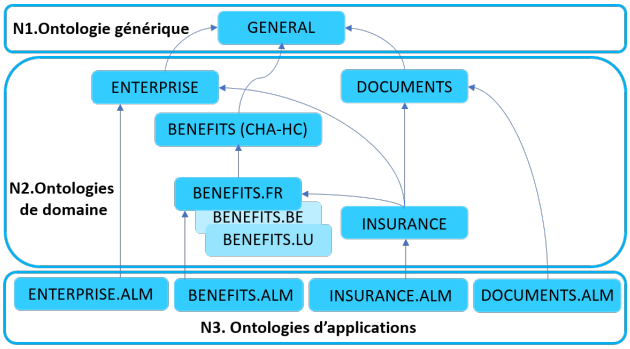

FIGURE 2 – Aperçu des ontologies Khresterion

La gestion de ces multiples ontologies est confiée à un gestionnaire de connaissances, qui regroupe l'ensemble des concepts et relations de ces ontologies sous une même hiérarchie, formant ainsi la TBox (Terminology Box)[3] de notre Knowledge Graph. À la TBox nous associons une population ABox (Assertions Box)[3] décrivant les individus et leurs relations (quel individu appartient à quelle entité, quel individu est lié à quel autre, à travers quelle relation). TBox + ABox sont stockées dans une base de données RDF de type triple-store.

Règles Khresterion :
La deuxième couche du référentiel correspond à un ensemble de règles de cohérence stockées dans un gestionnaire de règles, capable d'exécuter ces règles à la demande afin de gérer la cohérence du domaine.
Les règles capitalisent les connaissances d'un expert, la réglementation et traduisent les contraintes du métier (règles de bonne pratique, de fonctionnement ...). L'une des difficultés consiste à collecter et traduire les textes réglementaires qui n'existent souvent que sous-forme rédigée en un système cohérent et complet de règles formalisées dans un langage compréhensible par la machine.
Nous définissons trois familles de règles :

- **Des règles d'inférences :** elles sont classiquement exprimées sous la forme *Si...Alors...* Ce type de règles de cohérence n'a d'effet que si toutes les prémisses d'une règle sont vraies, c'est à dire si les informations qu'ils représentent sont présentes. Nous utilisons par exemple ces règles pour proposer à l'utilisateur des *expressions de garanties* de manière adaptative et contextuelle.
- **Des règles de calcul :** elles sont utilisées pour calculer certaines informations, qu'elles soient numériques, datées ou bien des chaînes de caractères. Les opérations de calcul sont représentées dans des

règles de cohérence classiques auxquelles sont attachées des formules de calcul. Nous utilisons par exemple ces règles pour calculer *des montants de cotisations sociales*.

— **Des règles de validation :** elles sont utilisées pour vérifier si l'environnement d'exécution contient une information vérifiant une contrainte ou non. Nous utilisons par exemple ces règles pour vérifier si une valeur saisie par l'utilisateur est cohérente ou non dans un certain contexte : vérifier la validité d'un numéro de téléphone, vérifier la cohérence d'une valeur dans une *expression de garanties*, identifier une valeur manquante, etc.

Ces ensembles de règles sont définis en SHACL. Ce nouveau standard recommandé par le W3C, permet de décrire explicitement la forme, *Shape*, du schéma RDF attendue par les machines. SHACL comprend, entre autres, des fonctionnalités permettant d'exprimer des conditions qui limitent le nombre de valeurs qu'une propriété peut avoir, le type de celles-ci, les plages numériques, les modèles de correspondance de chaîne et les combinaisons logiques de certaines contraintes. Seulement, dans la plupart des cas, nos règles expriment des contraintes un peu plus complexes. Nous avons alors rédigé nos règles en SHACL mais en incluant un mécanisme d'extension en SPARQL.
Le gestionnaire de règles est en écoute du système Khresterion lors du changement au niveau du KG. En cas de violation des shapes SHACL-SPARQL, le gestionnaire produit un certain nombre de rapports sous la forme de messages d'informations, de différents niveaux de sévérité, permettant d'accompagner l'utilisateur dans ses prises de décisions. Ces messages, accompagnés de codes couleurs, peuvent être de type conseil (couleur vert), dé-conseil (couleur orange), violation (couleur rouge) ou incomplétude (couleur violet) (figure 3).

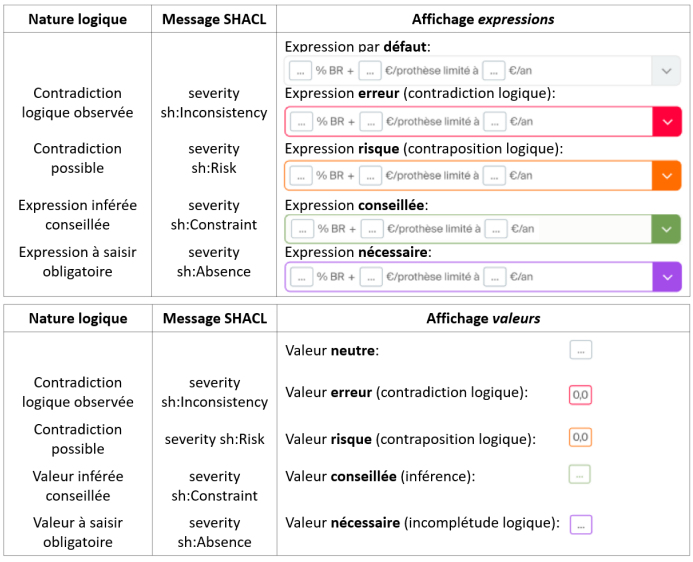

FIGURE 3 – Niveaux de sévérité SHACL-SPARQL

L'utilisation de ces règles contribue efficacement à la qualité des données stockées.
Dans les sections 3 et 4 nous présentons quelques exemples de ces règles appliquées à un scénario de raisonnement sur des expressions de garanties.

3 Scénario de Raisonnement : Cas d'un Tableau de Garantie Santé

Cette section présente une démonstration d'un scénario de raisonnement sur un tableau de garantie santé dans le système Khresterion.
Un *tableau de garanties* d'une assurance santé définit un montant de remboursement pour un poste de soins et un jalon (ou niveau) de remboursement. Par exemple, un poste *Lentilles* est ainsi associé à un certain *remboursement* pour un jalon dit de *base*, mais peut bénéficier d'un niveau de remboursement supérieur pour un niveau *optionnel* ou *surcomplémentaire*.
Les données experts sur des *garanties assurance santé pour les lentilles* considèrent deux types de lentilles : celles qui sont remboursées par la sécurité sociale et celles qui ne le sont pas. Le modèle Khresterion définit dans ce cas : (i) le concept $Lentilles$ pour représenter le type de garantie, et (ii) les deux concepts $Actes\_et\_prestations\_pris\_en\_comptes\_par\_la\_SS$ et $Actes\_et\_prestations\_Non\_pris\_en\_comptes\_par\_la\_SS$ pour qualifier les deux types de remboursement sécurité sociale. Ainsi les lentilles seront représentées dans KGK comme suit :

— Lentilles acceptées par la sécurité sociale (ASS) est une instance RDF de type $Lentilles$ et $Actes\_et\_prestations\_pris\_en\_comptes\_par\_la\_SS$

— Lentilles non acceptées par la sécurité sociale (NASS) est une instance RDF de type $Lentilles$ et $Actes\_et\_prestations\_Non\_pris\_en\_comptes\_par\_la\_SS$

Le modèle Khresterion conceptualise aussi les différents niveaux de garanties (bases, options et surcomplémentaires) et leurs qualifications (responsable, non responsable, facultative, obligatoire, y compris le remboursement sécurité sociale, y compris le régime de base, etc). Par exemple, le niveau de garantie base peut être exprimé en y compris le remboursement de la sécurité sociale (YCRSS), alors que le deuxième niveau de garantie, sa surcomplémentaire, peut être exprimé en y compris le régime de la base (YCRB).
Pour faire le lien entre le poste de garantie (les lentilles dans notre scénario), le niveau de remboursement et l'expression sélectionnée, une instance de type $ReimbursementLevel$ assure la liaison entre ces différents éléments. La figure 4 illustre, par exemple, le cas d'une lentille ASS liée à un paiement. Le paiement la relie à un niveau de garantie de base exprimé en y compris le remboursement sécurité sociale (c.f. l'instance milestone), et la relie aussi à l'expression de remboursement correspondante.
Si un tableau de garanties contient plusieurs niveaux de remboursements (i.e. base + surcomplémentaire), le poste

sera lié à deux instances différentes de paiement dont chacune est liée au niveau de garantie et expression correspondants. Une telle conceptualisation est particulièrement utile pour la modélisation des règles d'inférence sur les expressions possibles par poste et par niveau de remboursement. Il existe, en effet, de nombreuses contraintes, de nature réglementaire ou métier, qui définissent les expressions de remboursement possibles pour un poste et un niveau donnés en fonction de l'expression sélectionnée pour ce même poste et un niveau précédent.

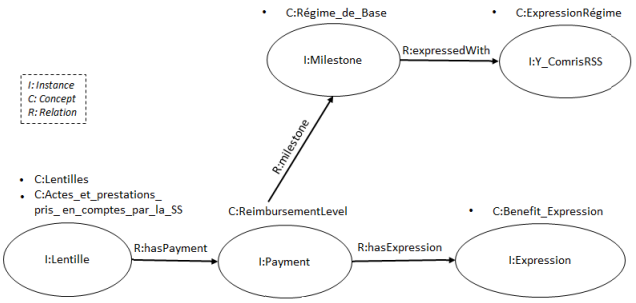

FIGURE 4 – Echantillon du Knowledge Graph Khresterion pour la garantie de base YCRSS au niveau des lentilles ASS

Le système Khresterion propose à l'utilisateur une liste d'expressions conformes selon le poste et le niveau de remboursement : pour les lentilles ASS au niveau de la base YRSS, l'utilisateur peut choisir entre 3 expressions possibles. Le choix de l'expression de base aura une conséquence sur l'inférence du niveau surcomplémentaire correspondant. Par exemple, si l'utilisateur choisit une expression "X % de la BR" dans la base il aura trois expressions proposées par le raisonneur Khresterion dans la surcomplémentaire correspondante : "X % de la BR" , "X % de la BR + un crédit de Y € par année civile" ou "X % de la BR + un crédit de Y % du PMSS par année civile". Ainsi, le système assure par raisonnement la conformité du choix utilisateur par rapport aux règles métiers. Suite au choix de l'utilisateur dans le régime de base, le système rafraîchit son raisonnement pour inférer les expressions possibles dans la surcomplémentaire correspondante tout en respectant la cohérence du KGK. Ce raisonnement est assuré par un ensemble de règles d'inférence SHACL étendu par SPARQL comme expliqué dans la section 2.

4 Scénario de Validation : Cas d'un Tableau de Garanties Santé

En plus des règles d'inférence, pour obtenir la conformité expert, le système Khresterion applique des règles de validation pour vérifier la cohérence des valeurs saisies par l'utilisateur. Par exemple pour les lentilles ASS dans la base YCRSS, le "X % de la BR" doit valider la règle "100 $\leq$ X $\leq$ 1000".
Nous étendons SHACL par du SPARQL pour bénéficier de toute la performance des techniques du graph matching apportée par ce langage de requêtage.

La figure 5 illustre un exemple de la représentation d'une expression dans KGK. Une expression peut avoir un ou plusieurs composants. Chaque composant est constitué d'une unité (exemple Euros), d'une valeur, d'au maximum un opérateur de liaison (exemple, +, -, limité à) et d'au maximum un quantificateur (exemple, crédit, forfait).

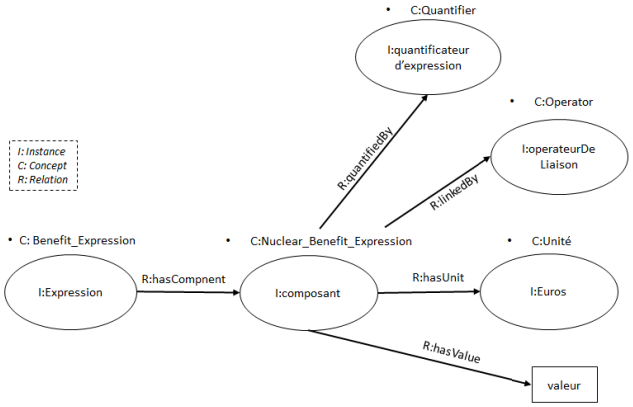

FIGURE 5 – Instance d'une expression dans KGK

Ce niveau de détails dans la modélisation des expressions dans KGK nous permet d'avoir un contrôle total pour assurer la conformité de tous ces composants. La validation SHACL-SPARQL exécutée par le biais du graph matching (par exemple, le parcours du graphe constitué de l'ensemble des noeuds des figure 4 et figure 5) permet de contrôler les valeurs des composants d'une expression d'un poste pour un niveau de garantie spécifique. Par exemple, pour l'expression "X % de la BR" (Base $\leq$ X) des lentilles ASS dans la surcomplémentaire YCRB, si pour le même poste dans le régime de base nous avons "100 % de la BR", alors le raisonneur applique la règle de validation pour vérifir si la valeur du régime surcomplèmentaire est bien supérieure ou égale à la valeur de base.
En plus du contrôle sur les valeurs, le langage SHACL permet de programmer dans les Shapes des messages à afficher à l'utilisateur en exploitant la propriété $sh : severity$. Différents codes couleurs sont attribués selon la sévérité modélisée dans la règle. Exemple, l'expression "80 % de la BR" pour les lentilles ASS dans la base YCRSS va donner une alerte sur l'expression de couleur orange signifiant que l'état de cette expression est déconseillé par rapport à la conformité expert exigeant "100 $\leq$ X". La liste des niveaux de sévérité et des couleurs correspondant est détaillée dans la Section 2.
À la fin de la saisie, l'utilisateur a la possibilité d'éditer son contrat d'assurance. La figure 6 illustre un aperçu de la représentation des expressions de remboursement du poste lentilles dans un contrat. Notons ici que le raisonneur applique des règles de calcul de libellés pour composer les textes des expressions et les textes des contrat d'assurance. Par exemple, des règles de calcul sont appliquées pour la construction du libellé "120 % de la BR + crédit de 400 € par année civile" à partir des différents composants présents dans la figure 5.

Autres dispositifs médicaux optique		
Lentilles acceptées par la SS	100 % de la BR	120 % de la BR + crédit de 400 € par année civile
Lentilles refusées par la SS (y compris lentilles jetables)	Crédit de 300 € par année civile	Crédit de 400 € par année civile
Chirurgie réfractive (Myopie, hypermétropie, astigmatisme, presbytie)	Crédit de 400 € par oeil par année civile	Crédit de 80 % du PMSS par oeil par année civile

FIGURE 6 – Exemple des clauses pour les lentilles dans un contrat

5 Conclusion

Les Knowledge Graphs servent de substrat commun de connaissances au sein d'une organisation ou d'une communauté, permettant la représentation, l'accumulation, la conservation et la diffusion des connaissances au fil du temps [6]. Les Knowledge Graphs ont été appliqués dans une grande variété de cas d'utilisation, allant des applications commerciales - impliquant la recherche sémantique, les systèmes de recommandations, la publicité ciblée, l'automatisation du transport, etc. Dans ce papier, Khresterion spécialiste de solutions IA, présente une démonstration de son système de raisonnement à base de Knowledge Graphs. Un scénario spécifique pour l'accompagnement du raisonnement expert sur les expressions de garantie assurance santé est présenté.
Dans des travaux futurs, nous envisageons d'investiguer l'identification des incohérences entre différents ensembles de règles (pratiques métier spécifiques, pratiques commerciales et réglementations), et de travailler sur un outil de rédaction de règles qui permettra aux experts métiers de définir eux-même les règles avec un langage naturel.

Abréviations

— ASS : accepté par la sécurité sociale
— KGK : Knowledge Graph Khresterion
— NASS : non accepté par la sécurité sociale
— YCRB : y compris le régime de base
— YCRSS : y compris remboursement sécurité sociale

Références

[1] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, J.E. Labra Gayo, S. Kirrane, S. Neumaier, A. Polleres, R. Navigli, A.C. Ngonga Ngomo, S.M. Rashid, A. Rula, L. Schmelzeisen, J. Sequeda, S. Staab, A. Zimmermann, Knowledge Graphs, *arXiv :2003.02320v1*, cs.AI, 2020.

[2] G. Antoniou, C.V. Damásio, B. Grosof, I. Horrocks, M. Kifer, J. Maluszynski, P.F. Patel-Schneider, Combining Rules and Ontologies : A survey, *Technical Report, IST506779/Linköping/I3-D3/D/PU/a1, Linköping University*, Linköping University, 2005.

[3] G. De Giacomo, M. Lenzerini, TBox and ABox Reasoning in Expressive Description Logics, *Proc. of the 5th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR'96)*, Morgan Kaufmann, 1996.

[4] H. Prade, L'intelligence artificielle, mais enfin de quoi s'agit il ? *Les Livrets du Service culture UPS* , Université Paul Sabatier, 2001.

[5] N. da Costa, D. Krause, O. Bueno, Paraconsistent Logics and Paraconsistency, *Philosophy of Logic*, Elsevier, 2007.

[6] N. F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, and J. Taylor. Industry-scale Knowledge Graphs : Lessons and Challenges . *Communications of the ACM, 62(8), 36-43.*,(2019).

[7] N. Guarino, D. Oberle, S. Staab, What is an Ontology ?, *Handbook on Ontologies*, Springer, 1-17, 2009.

[8] T. Eiter, G. Ianni, T. Krennwallner, A. Polleres, Rules and Ontologies for the Semantic Web, *Reasoning Web : 4th International Summer School 2008, Venice, Italy, September 7-11, 2008, Tutorial Lectures*, Springer Berlin Heidelberg, 2008.