REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DEL'ENSEUGNEMENT SUPERIEURE ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE ABOU-BEKR BELKAID – TLEMCEN

MEMOIRE

FACULTE DES SCIENCES – DEPARTEMENT D'INFORMATIQUE

Pour l'obtention du diplôme de :

MASTER EN INFORMATIQUE

Spécialité : Génie Logiciel

Par :

Mlle. BERREZOUG Yamna Dalal

Sur le thème

Annotation sémantique des exigences

Soutenu publiquement le 12 Juillet 2021 à Tlemcen devant le jury composé de :

Mme MALTI Djawida	Université de Tlemcen	Présidente
Mr CHIKH Azeddine	Université de Tlemcen	Encadrant
Mme KHITRI Souad	Université de Tlemcen	Examinatrice

Année Universitaire : 2020 ~ 2021

Dédicaces

Je dédie ce modeste travail, Avec l'expression de ma reconnaissance,

A mon cher père, mon précieux offre du dieu, à qui je dois ma réussite et ma vie. Tout mon respect.

A mon adorable mère, la femme qui a souffert sans me laisser souffrir, qui n'a jamais dit non à mes exigences et qui n'a épargné aucun effort pour me rendre heureuse.

A mes chères sœurs Assia, Sarra, et Wissal qui n'ont pas cessé de m'encourager, conseiller, et soutenir tout au long de mes études. Que Dieu les protège et leurs offre la chance et le bonheur.

A mon adorable nièce Nada qui entre toujours la joie et le bonheur à toute la famille.

Remerciements

Je remercie ALLAH le tout puissant de m'avoir donnée la santé et la volonté d'entamer et de terminer ce mémoire

« El hamdoulillah »

Je tiens à remercier d'abord mon encadreur Mr CHIKH Azeddine pour m'avoir proposé ce thème. Pour la qualité de son encadrement, pour sa rigueur, sa patience, sa disponibilité et ses orientations. Il s'est toujours montre prêt à me prodiguer ses précieux conseils.

Je remercie également les membres du jury Mme MALTI Djawida et Mme KHITRI Souad qui ont bien voulu examiner le présent travail.

Mes remerciements s'adressent aussi à Mr BELABED Amine, enseignant au département d'informatique, pour son aide pratique

Je remercie Mme HAKEM Amel Epse BSESSENOUCI enseignante au département des sciences économiques pour ses encouragements et son soutien moral

Je remercie enfin tous mes enseignants pour leurs générosités et leurs contributions dans notre formation et pour leurs enseignements précieux

Sommaire

Introduction générale 1

Contexte et problématique 1

Objectifs 1

Structure du mémoire 1

Chapitre I : Le Web sémantique 3

I.1 Introduction 4

I.2 Définition du web sémantique 4

I.3 Les principes du web sémantique 5

I.4 Les composants du web sémantique 7

I.4.1 Les ontologies : 7

I.4.1.1 Définitions 7

I.4.1.2 Les principes de développement d'ontologie : 8

I.4.1.3 Les composants d'une ontologie 8

I.4.1.4 Les catégories des ontologies 9

I.4.2 L'annotation sémantique 10

I.4.2.1 Définition d'une annotation classique 10

I.4.2.2 Définition de l'annotation sémantique 11

I.4.2.3 Processus d'annotation sémantique 12

I.4.2.4 Les éléments de bases de l'annotation sémantique 12

I.4.3 Les langages de représentation des données sémantiques : 13

1- **XML** 13

2- **RDF** 13

3- **RDFS** 14

4- **RDFa :** 14

5- **OWL** 14

I.4.4 Langages de requêtage des données sémantiques 15
SPARQL : 15
I.5 Conclusion 15
Chapitre II : Ingénierie des exigences 16
II.1 Introduction 17
II.2 Définitions 17
II.3 Ingénierie des exigences 18
II.4 Processus d'ingénierie des exigences 18
II.5 Spécification des exigences logicielles 19
II.5.1 L'approche « Volere requirement » 20
II.5.1.a Requirement Shell 21
II.5.1.b Modèle de processus Volere 21
II.6 Conclusion 22
Chapitre III : Etude des systèmes existants 23
III.1 Introduction 24
III.2 L'étude d'un système similaire 24
III.3 Conclusion 30
Chapitre IV : Construction des datasets 31
IV.1 Introduction 32
IV.2 La construction de SRS 32
IV.3 La construction de l'ontologie 32
IV.3.1 Motivations 33
IV.4 Construction de document de requêtes 34
IV.5 Conclusion 34
Chapitre V : Conception générale 35
V.1 Introduction 36
V.2 Architecture du système 36
V.3 Conclusion 39

Chapitre VI : Conception détaillée 40
VI.1 Introduction 41
VI.2 La conception des composants 41
VI.2.1 L'ontologie 41
VI.2.1.a Présentation du contexte 41
VI.2.1.b Processus d'affinement de l'ontologie 42
VI.2.1.c Présentation de l'ontologie SWOREPLUS 42
VI.2.2 Le document SRS 47
VI.2.3 Le fichier d'annotation 48
VI.2.4 Le fichier de requêtes SPARQL 49
VI.2.5 L'annotateur sémantique 50
VI.2.6 Editeur d'ontologie 50
VI.2.7 Editeur de texte 50
VI.2.8 Editeur de requêtes SPARQL 51
VI.2.9 Moteur de recherche SPARQL 51
VI.3 Conclusion 53
Chapitre VII : Implémentation 54
VII.1 Introduction 55
VII.2 Environnement de travail 55
VII.2.1 Environnement matériel 55
VII.2.2 Environnement logiciel 55
VII.2.2.a L'environnement de développement d'ontologie 55
VII.2.2.b L'environnement pour l'annotation sémantique 55
VII.2.2.c L'environnement pour SRT et la recherche sémantique 56
VII.3 Réalisation du système 57
VII.3.1 Implémentation de l'ontologie 57
VII.3.2 L'interface de Semantic Requirement Toool 62
VII.3.2.a L'annotation sémantique 63

VII.3.2.b La recherche sémantique 70

VII.4 Conclusion 80

Conclusion générale et perspectives 81

Bibliographie 82

Liste des tableaux

Tableau 1. Comparaison entre les travaux qui utilisent des wikis sémantiques (Ricardo de Almeida Falbon, Carlos E Correa Braga) 29

Tableau 2. Les propriétés d'objet 45

Tableau 3. Les propriétés de donnée 46

Tableau 4. Les restrictions 47

Liste des figures

Figure I.1. La vision du web de Tim Bernard Lee (Tim Bernard Lee, 1989) 6
Figure I.2. L'annotation sémantique (dayra, 2017) 11
Figure I.3. Les couches de L'OWL (Belabed, 2019) 15
Figure II.1. L'utilisation des standards SRS dans différents niveaux (Dörr, 2017) 20
Figure II.2. Requirements Shell (Volere Requirements Specification Template, 2016) 21
Figure III.1. Fragment de l'ontologie des exigences logicielles (Ricardo de Almeida Falbon, Carlos E Correa Braga) 25
Figure III.2.L'analyse d'impact des changements d'exigences (Ricardo de Almeida Falbon, Carlos E Correa Braga) 266
Figure III.3. Matrice de traçabilité Exigences x Cas d'utilisations (Ricardo de Almeida Falbon, Carlos E Correa Braga) 27
Figure III.4. Evaluation de la cohérence des priorités des besoins (Ricardo de Almeida Falbon, Carlos E Correa Braga) 27
Figure III.5. Utilisation de Checklists pour la vérification des exigences besoins 28
Figure IV.1 SWORE Adapté 33
Figure IV.1 SWOREPLUS 33
Figure V.1 Architecture Générale 37
Figure VI.1. La hiérarchie des classes de l'ontologie SWOREPLUS 43
Figure VI.2. Les cas possible pour les valeurs de triplets RDF 49
Figure VI.3. Diagramme de classe 52
Figure VII.1. Le choix du format de l'ontologie 57
Figure VII.2. Le choix d'emplacement de sauvegarde de l'ontologie 57
Figure VII.3. Édition l'URI de l'ontologie 58
Figure VII.4. L'édition des classes de l'ontologie 58
Figure VII.5. L'édition des propriétés d'objet 59
Figure VII.6. L'édition des propriétés de type de donnée 60
Figure VII.7. Édition des restrictions 61
Figure VII.8. visualisation de l'ontologie 62
Figure VII.9. Page d'accueil 63
Figure VII.10. La page d'accueil de GATE 64
Figure VII.11. Chargement de plugin Ontology 65
Figure VII.12. La sélection des plugins Ontology et Ontology Tools pour les chargés 66
Figure VII.13. Les paramètres pour créer un nouvel OWLIM Ontology 67

Figure VII.14 . Annotation sémantique étape1 69
Figure VII.15. Annotation sémantique étape2 70
Figure VII.16. Chargement du fichier 71
Figure VII.17. Les requêtes fréquentes 72
Figure VII.18. Ajouter une nouvelle requête fréquente 73
Figure VII.19. Modifier une requête fréquente 73
Figure VII.20. Supprimer requête fréquente 73
Figure VII.21. L'onglet "Traceability" 75
Figure VII.22. L'onglet "Requirements" 75
Figure VII.23. L'onglet "Conflict" 77
Figure VII.24. L'onglet "Dependies" 77
Figure VII.25. L'onglet "FR associate with NFR" 78
Figure VII.26. NFR no associated 78
Figure VII.27. L'onglet "Advanced Search" 80

Liste des abréviations

HTML : HyperText Markup Language

HTTP : HyperText Transfer Protocol

URL : Uniform Resource Locator

W3C : World Wide Web Consortium

RDF : Resource Description Framework

URI : Uniform Resource Identifier

API : Application Programming Interface

BDD : Base de données

XML :eXtensible Markup Language

<S, P, O> : <Sujet, Prédicat, Objet>

DTD : Document Type Definition

RDFS : RDF Schema

OWL : Ontology Web Language

IRI : Internationalized Resource Identifier

RDFa : Resource Description Framework in Attributes

XHTML : eXtensible HyperText Markup Language

OWL DL : Ontology Web Language Description Logics

SPARQL :SPARQL Protocol And Rdf Query Language

IS : ingénierie des systèmes

IE : Ingénierie des exigences

SRS : Software Requirements Specification

PDF : Portable Document Formatt

LOD : Linked Open Data

SWORE : SoftWiki Ontology for Requirements Engineering

SRT: SemanticRequirements Tool

Introduction générale

Contexte et problématique

Le cout d'erreur dans l'ingénierie des exigences est très élevé. Cependant l'échec dans cette phase dure pendant tout le cycle de vie de développement du logiciel. Dans le pire des cas, à cause des exigences manquantes, incorrectes, ou mal comprises, le projet final ne répond pas aux besoins du client et sera abandonné.

La gestion efficace des documents d'exigences est un facteur important dans l'ingénierie des exigences.

Dans la communauté du web sémantique, les chercheurs préconisent qu'afin de rendre les informations interprétées par les machines, nous devons ajouter des métadonnées basées sur une ontologie au contenu du web. L'acte d'ajouter des métadonnées basées sur une ontologie dans des ressources informatiques est appelée l'annotation sémantique. Evidemment, le rôle de l'ontologie dans cette annotation sémantique est de fournir une base commune pour saisir la signification voulue des connaissances partagées.

Objectifs

L'objectif attendu de ce travail est de proposer une plateforme qui permet de sémantifier les exigences d'un projet donné de logiciel. Cette plateforme se base sur deux dimensions. La première consiste à faire l'annotation sémantique des documents d'exigences selon le Template Volere et conformément à une ontologie relative à l'ingénierie des exigences. La deuxième dimension consiste à faire la recherche sémantique sur les documents d'exigences déjà annotés.

Structure du mémoire

Notre mémoire est structuré selon sept chapitres :

Le chapitre I « le web sémantique » présente les concepts du web sémantique et d'annotation sémantique.

Le chapitre II « l'ingénierie des exigences » présente les principes de base de l'ingénierie des exigences ainsi que le Template Volere

Le chapitre III « Etude des systèmes existants » présente quelques travaux qui utilisent la sémantique dans l'ingénierie des exigences.

Le chapitre IV « Construction des datasets » présente les motivations derrière le choix de notre dataset, à savoir la construction de l'ontologie, la construction du SRS et la construction des requêtes.

Le chapitre V « Conception générale » présente l'architecture générale proposée pour notre système.

Le chapitre VI « Conception détaillée » présente a conception détaillée du chaque composant de notre système.

Le chapitre VII « Implémentation » présente l'implémentation du système à travers ses deux dimensions : l'annotation sémantique et la recherche sémantique.

Chapitre I : Le Web sémantique

I.1 Introduction

Le web classique ou bien le web actuel est considéré comme un ensemble de documents reliés entre eux par des liens hypertextes, représentés par le langage HTML qui donne une mise en forme à ces documents. Il repose sur le protocole HTTP qui permet d'accéder à ces documents grâce à des URL. La structure des ressources dans le web actuel est bien définie, et est conçue pour n'être comprise et traitée que par les humains, et non pas par les machines. Ces limites étaient le point de départ pour étendre le web2.0 vers le web3.0 ou bien ce qu'on appelle le web sémantique, où on doit indexer les informations avec un langage standard pour que les documents soient compréhensibles par les machines et l'information aura un sens bien défini pour que les applications puissent exploiter la sémantique des ressources.

Dans ce chapitre, nous allons présenter la notion du web sémantique, ses principes de base, les langages utilisés, la notion des ontologies et l'annotation sémantique.

I.2 Définition du web sémantique

D'après Tim Bernard Lee, l'inventeur du web et le directeur du W3C, le web sémantique « is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation »

Cela veut dire que le web sémantique est une extension du web actuel dans lequel les informations possèdent un sens bien défini et permettant aux ordinateurs et aux personnes de mieux travailler en coopération

Le web sémantique ou encore le « web de données » peut être considéré comme une couche supplémentaire étroitement imbriquée avec le web classique et possédant les mêmes propriétés (Bizer , Heath , & Berners-Lee, 2011) :

- Le web sémantique est générique et peut contenir tout type de données ;
- Tout le monde peut publier des données sur le web de donnée ;
- Les éditeurs de données ne sont pas limités dans le choix des vocabulaires avec lesquels représenter les données ;
- Les entités sont connectées par les liens RDF, créant un graphe de données global qui couvre les sources de données.

Du point de vue de développement d'application, le web sémantique présente les caractéristiques suivantes :

- Les données sont strictement séparées des aspects de formatage et de représentation ;
- Les données se décrivent elle-même. Si une application utilisant des données liées rencontre des données décrites avec un vocabulaire inconnu, l'application peut déréférencer les URI qui identifient les termes de vocabulaire afin de trouver leur définition ;
- L'utilisation de HTTP comme mécanisme d'accès aux données normalisé et RDF comme modèle de données normalisé simplifie l'accès aux données par rapport aux API web, qui reposent sur des modèles de données et des interfaces d'accès hétérogènes.
- Le web sémantique est ouvert, ce qui signifie que les applications ne sont pas implémentées sur un ensemble fixe de sources de données, mais peuvent découvrir de nouvelles sources de données au moment de l'exécution en suivant les liens RDF.

I.3 Les principes du web sémantique

Le but du web sémantique est d'ajouter un sens à une information dans le web, pour automatiser les services, relier, inférer et découvrir les connaissances qui sont vraisemblables. Nous allons voir dans cette section les principes qui régissent le web sémantique (HACINE GHERBI Ahcine, 2014).

1. Identification par URI : dans le web sémantique, les identifiants sont des URIs (Uniforme Resource Identifier). Toutes les entités, les personnes, les objets du monde réel peuvent être indexés dans le web sémantique par des URIs.

2. Typage des ressources et liens Web : le web est composé de ressources et de liens. Les ressources représentent les pages, et les liens représentent les liens entre ces pages. Dans le web actuel ces ressources et liens ne sont pas typés. Par exemple un homme qui lit article scientifique peut savoir qu'il s'agit d'un article scientifique et non pas un roman. Par contre le machine ne peut pas ne peut pas le savoir. Dans le web sémantique la machine est capable de le savoir. C'est l'un des objectifs du web3.0 de donner un type aux ressources et aux liens du web.

3. Tolérance de l'information partielle : le web sémantique doit continuer son fonctionnement lorsqu'une ressource indexée n'est pas disponible.

4. L'indisponibilité d'une vérité absolue : dans le web n'importe qui peut dire n'importe quoi sur n'importe quel sujet. Ainsi, pour savoir qui a dit quoi, quand et dans quel but, on utilise les signatures numériques. La confiance dans le web est la même que chez les humains, on croit plus à nos proches amis, un peu moins à leurs amis, et moins aux amis de ces derniers.

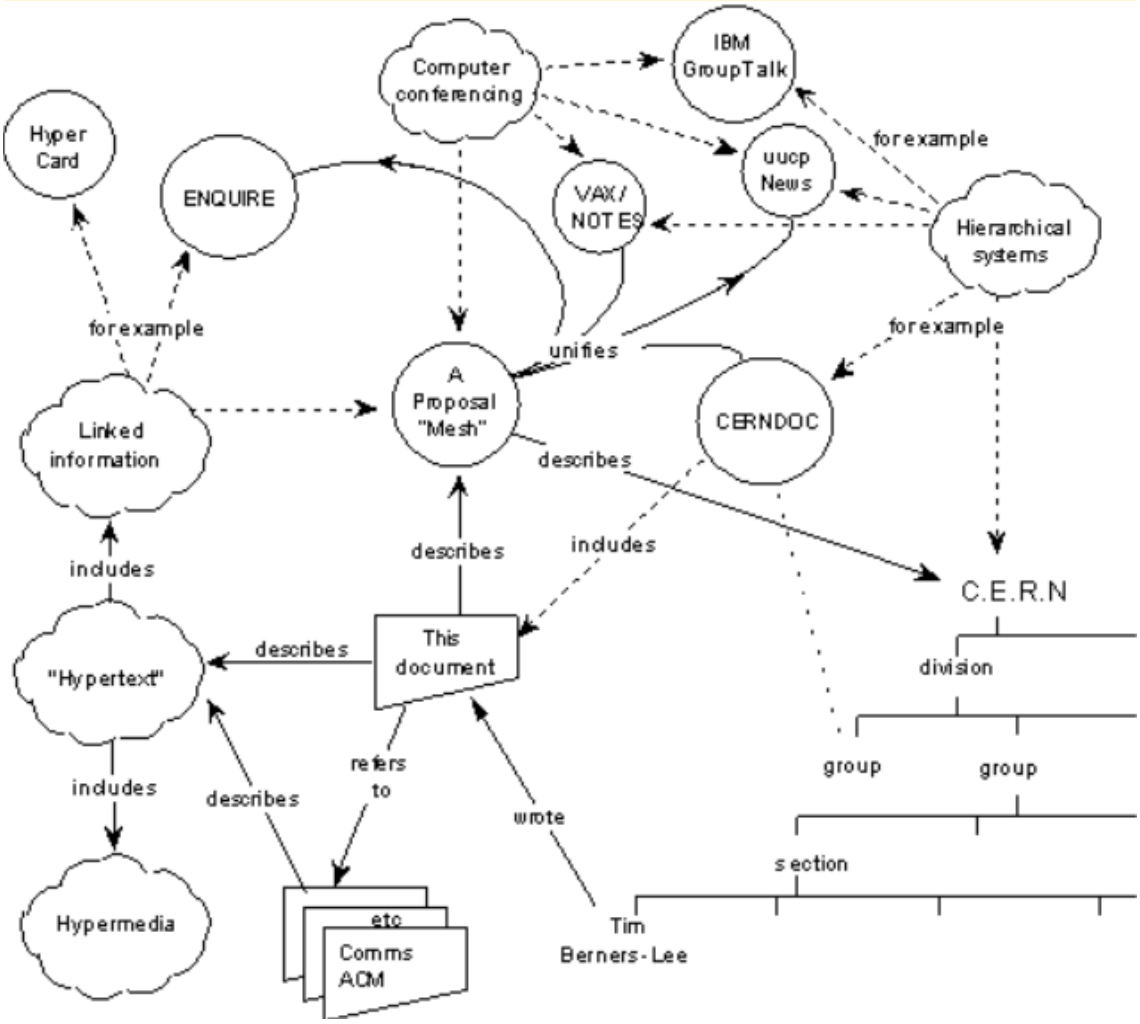


Figure I.1. *La vision du web de Tim Bernard Lee (Tim Bernard Lee, 1989)*

5. Support de l'évolution : dans notre vie quotidienne on voit des concepts semblables définis par différentes personnes ou bien par la même personne à des moments différents. On combine alors les nouveaux concepts avec les anciens sans modifier ces derniers pour bien comprendre. C'est le même principe qui est fourni par le web

sémantique, il fournit des outils communautaires qui peuvent être utilisés pour clarifier les inconsistances

6. Architecture minimaliste : dans le but d'obtenir une adaptation rapide et simple, le web sémantique ne doit pas être compliqué

I.4 Les composants du web sémantique

I.4.1 Les ontologies :

Toute activité humaine spécialisée développe son propre langage ou bien jargon sous forme d'une terminologie. L'absence d'un tel vocabulaire au sein d'une entreprise ou d'une industrie constitue un frein à la collaboration et génère des problèmes de compréhension et des difficultés à partager les connaissances entre les différents acteurs qui font des métiers différents.

Donc le rôle des ontologies est d'améliorer la communication entre les humains, entre les humains et les machines et finalement entre les machines.

I.4.1.1 Définitions

Le terme ontologie est d'origine grecque qui signifie « l'étude de l'existant ». Ce mot a été intégré dans le monde informatique au début des années 1990.

En informatique, une ontologie « est une représentation de propriétés générales de ce qui existe dans un formalisme supportant un traitement rationnel. C'est le résultat d'une formulation exhaustive et rigoureuse de la conceptualisation d'un domaine. » (Fabien Gandon, 2006)

Ainsi, la définition la plus populaire dans le monde informatique est la suivante : « une ontologie est une spécification explicite d'une conceptualisation » (Gruber, 1993). Dans cette définition le terme conceptualisation réfère à un modèle abstrait d'un domaine précis dans le monde réel en identifiant des concepts pertinents décrivant ce domaine, et le mot explicite veut dire que ces concepts utilisés sont définis d'une manière précise et claire. Enfin tout simplement, une ontologie est un ensemble de concepts qui sont liés par des relations.

La notion clé liée aux ontologies est la logique de description. La logique de description est une famille de langage de représentation de connaissance, elle est divisé en trois parties : langage assertionnel (ABOX), langage terminologique (TBOX) et langage de rôle (RBOX) qui sont définis comme suit :

- TBOX : une déclaration TBOX permet de décrire la conceptualisation d'un domaine d'intérêt en définissant des ensembles d'individus en fonction de leurs caractéristiques. Par exemple, pour décrire qu'une femme est une personne féminine on doit écrire la déclaration suivante Femme ≡ Personne ∪ Femelle. (Baader, Calvanese, McGuinness, Patel-Schneider, & Nardi, 2003).
- ABOX : pour associer un fait à un modèle conceptuel ou à une ontologie.il contient des connaissances supplémentaires sur le domaine d'intérêt. Par exemple, Femelle ∪ Personne « Anna » indique que « Anna » est une personne de sexe féminin. (Baader, Calvanese, McGuinness, Patel-Schneider, & Nardi, 2003).
- RBOX : est un ensemble fini d'axiomes d'inclusion de rôles généralisés de la forme R ⊑ S, des axiomes d'équivalence de rôle sous forme R ≡S, des inclusions de rôles complexes sous la forme R1°R2 ≡ S, et déclarations de disjonction de rôles sous la forme Dis(R,S), ou R et S sont des rôles. (Sikos, 2016).
- Pour la relation « co-écrire » pour dire que si Alice et Bob ont on-écrit un article, on déduit la relation « Alice a écrit l'article » et « Bob a écrit l'article », on déclare « écrire ° écrire co-écrire »

I.4.1.2 Les principes de développement d'ontologie :

Les critères suivants permettent de mettre en évidence les aspects importants d'une ontologie (Gruber) :

- La clarté : la définition des concepts doit être complète et non ambigüe ;
- La cohérence : elle doit être vérifiée par les experts de domaine ;
- L'extensibilité : elle doit être facile à étendre ;
- Biais d'encodage minimal : elle doit être indépendante de tout langage d'implémentation ;
- Engagement ontologique minimal : il faut faire ce qu'il faut ni plus ni moins.

I.4.1.3 Les composants d'une ontologie

On distingue les éléments suivants dans une ontologie :

- Concepts/Relations : une ontologie est composée par des concepts ou ce qu'on appelle les classes, ces concepts sont liés entre eux avec les relations. On a 2 types de relations :
  1- Taxonomique : pour dire qu'un sous concept est un concept (pour définir l'héritage)
  2- Non taxonomique : pour dire que concept 1 est relié à concept 2
- Les propriétés (les attributs) : ce sont des restrictions des concepts ou bien des restrictions des relations

- Axiomes : un axiome est une vérité qu'on doit admettre et non pas démontrer
- Les individus : ce sont les instances des concepts

I.4.1.4 Les catégories des ontologies

On peut définir deux typologies d'ontologies :

- Les ontologies basées sur la structure de la conceptualisation
- Les ontologies basées sur le sujet de la conceptualisation

- Les ontologies basées sur la structure de la conceptualisation

Les ontologies basées sur la structure de conceptualisation sont définies par le type de structure de la conceptualisation. Dans cette catégorie on peut poser la question : comment on peut structurer les composants de l'ontologie ?

On distingue trois sous catégories :

1- Terminologique : permet de spécifié les termes utilisés pour représenter les connaissances. (lexiques, glossaires…)
2- Informatique : permet de représenter uniquement les individus d'un domaine sans les liens sémantique. (schéma d'une BDD, RDF)
3- Représentation des connaissances : regroupe les deux types précédents, il permet de modéliser les concepts d'un domaine avec les relations entre ces concepts

- Les ontologies basées sur le sujet de la conceptualisation

Les ontologies basées sur le sujet de la conceptualisation, ils sont définis par le sujet de conceptualisation. Dans cette catégorie on peut poser la question : que ce qu'on doit représenter dans l'ontologie ?

On distingue les quatre sous catégories :

1- Application : permet de représenter une partie ou bien un cas particulier d'un domaine et on ne s'intéresse pas au domaine tout entier.
2- Domaine : permet de représenter un domaine en entier et on ne doit pas traiter les cas particuliers.
3- Générique : permet de définir les concepts qui peuvent être communs entre plusieurs domaines.
4- Représentation : permet de définir la manière avec laquelle on va construire des ontologies de domaine ou des ontologies d'application ou génériques. La différence entre cette catégorie et les trois catégories précédentes est qu'elle ne fait pas référence aux entités du monde réel.

Pour comprendre la différence entre ces catégories, on va voir cet exemple. Si on prend une plateforme pour l'apprentissage en ligne. L'ontologie d'application sera l'ontologie de la plateforme elle-même qui va décrire l'application.

On peut voir une ontologie de E-learning de façon général comme une ontologie de domaine par rapport à cette application car notre plateforme d'apprentissage n'applique que certains principes qui existent dans le domaine de e-learning.

L'ontologie générique ça peut être une ontologie d'enseignement parce qu'il y a plusieurs manières pour enseigner quelque chose, cette ontologie peut représenter des choses qui sont communes entre plusieurs domaines en relations avec l'enseignement

Finalement, les ontologies de représentation ce sont des ontologies qui ne représentent pas des choses qui existent dans le monde réel, elles peuvent décrire la manière avec laquelle on va créer des ontologies génériques, de domaine ou d'application, donc une ontologie de représentation peut être une ontologie de RDF schéma

I.4.2 L'annotation sémantique

Dans notre travail nous nous intéressons aux annotations sémantiques, mais pour comprendre mieux ce principe, il faut d'abord définir l'annotation classique.

I.4.2.1 Définition d'une annotation classique

L'annotation peut prendre plusieurs définitions selon le domaine de recherche.

Selon le Dictionnaire de l'Académie Francaise (9éme édition), le terme annotation est dérivé d'un mot latin « annotare » qui signifier « noter », « annoter ».

Dans (Desmontils, Jaquelin, 2002) l'auteur définit une annotation comme suit : « une *annotation est une information graphique ou textuelle attachée à un document et le plus souvent placée dans ce document* ». D'autre part (Baldonado et al.2000) dans le contexte des recherches sur les Interfaces Homme Machine dit que « *une annotation est un commentaire sur un objet tel que le commentateur veut qu'il soit perceptiblement distinguable de l'objet lui-même et le lecteur l'interprète comme perceptiblement distinguable de l'objet lui-même* ».

D'après (Atilf, 1992), l'annotation est « *l'action d'annoter, et du résultat de cette action* ». Les auteurs de cette définition considèrent que l'action d'annoter est « *accompagner un texte de notes, de remarques, de commentaires* ».

I.4.2.2 Définition de l'annotation sémantique

L'annotation sémantique consiste à associer une sémantique à un document. En se basant sur une ontologie déjà définie, l'annotation permet de lier les classes et les instances d'une ontologie avec le contenu de la ressource annotée. Ce qui illustré dans la figure I.2 :

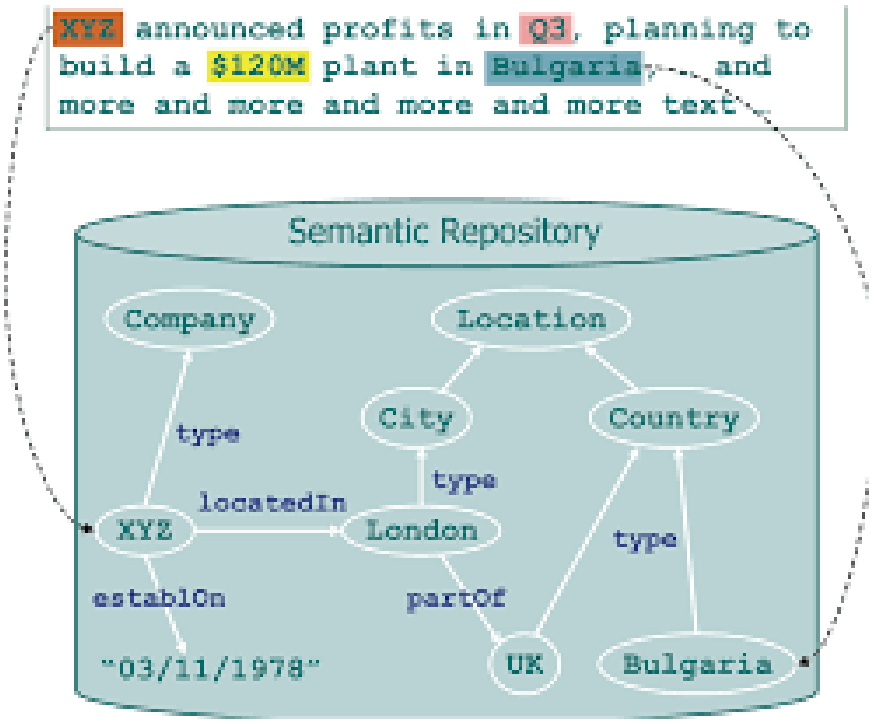


Figure I.2. *L'annotation sémantique (Dayra, 2017)*

Donc, nous pouvons définir l'annotation sémantique comme un ensemble d'instanciations attachées à un document. Ces instances peuvent être : des instances des classes, des instances d'attributs d'une classe, ou encore des instances de relation liées à une ontologie qui décrit le contenu de la ressource annotée.

La différence entre l'annotation sémantique et l'indexation automatique est que l'annotation utilise une ou plusieurs ontologies qui définissent le domaine de référence. En outre, les ressources annotées doivent l'être par des ontologies et non pas par les mots clefs.

La métadonnée est une donnée sur une donnée. Les deux termes annotation et métadonnée peuvent prendre en compte l'ajout d'information à une ressource, mais puisque ces deux termes existent c'est que nous pouvons distinguer entre eux.

La communauté anglophone du web sémantique considère que l'annotation devient des métadonnées dès qu'elles sont stockées (dans un serveur ou dans une base). D'une autre part,

l'annotation est écrite au cours d'un processus d'annotation et elle est située au sein de ressource, par contre une métadonnée est attachée à une ressource en tant que telle sur un document et aura une pertinence à priori (Garlatti et Prié, 2004).

Pour conclure, Nous parlons de l'annotation sémantique, lorsque les métadonnées sont des instances de concepts et de relations entre les concepts qui sont définis précisément, décrits formellement et structurés dans une ontologie (Toussaint et Tenier, 2007).

I.4.2.3 Processus d'annotation sémantique

Le processus d'annotation sémantique de documents en utilisant des ontologies ce base sur trois étapes (Gueraich, 2012) :

- **Repérer** : cette étape peut être faite d'une manière manuel ou automatique, elle consiste à identifier dans un document des références aux concepts de l'ontologie ;
- **Instancier** : elle peut être faite d'une manière manuel ou automatique, consiste à évaluer les attributs des concepts en utilisant des informations présentes dans le document ;
- **Enrichir** : cette étape peut être faite d'une manière manuel, elle consiste à ajouter des informations à l'aide des attributs de concepts qui n'ont pas pu être évalué dans la phase « instancier ».

La première et la deuxième étape ce sont des phases d'insertion de métadonnée, le but de ces deux phases est de localiser l'information déjà existant dans le document. La troisième phase est une phase d'ajout d'information, elle permet d'enrichir le document par des informations qui ne sont pas présente dans le document de façon explicite.

I.4.2.4 Les éléments de bases de l'annotation sémantique

Selon (Garlatti et Prié, 2004), l'annotation sémantique s'appuie sur les standards existants qui forment les technologies du web :

- Le protocole http : pour faire la transmission ;
- HTML et XML : qui représentent des feuilles de style pour représenter des résultats ;
- Le RDF : pour exprimer, échanger les métadonnées sous forme des triplets <S, P, O>.

I.4.3 Les langages de représentation des données sémantiques :

Le partage de l'information entre les applications et les communautés humaines est le rôle clé des ontologies dans la construction du web sémantique. On utilise le XML Schéma et les DTD pour échanger les données, mais le manque de la sémantique en XML ne permet pas de créer de nouveaux vocabulaires. Pour résoudre ces problèmes le RDF et RDFs (RDF Schéma) sont utilisé en associant la sémantique aux identificateurs. L'OWL est un langage conçu pour représenter les connaissances.

1- **XML :** (eXtensible Markup Language) est recommandé par le W3C en 1998. Son objectif était de pallier la sémantique « faible » de HTML car le HTML est plus utiliser pour présenter et non pas pour la description des données, en plus il ne permet plus d'utiliser ses propres balises. Pour ces causes la, le HTML n'est pas utiliser pour exprimer les métadonnées. Par contre le XML est un langage de balisage, il permet de structurer des informations dans un document en créant nos propres balises. Le point fort de l'XML est sa capacité de standardiser une représentation hiérarchique dans un document.

2- **RDF :** est le langage de base du Web sémantique. C'est un langage standardisé, recommandé par le W3C en 1999.

Ce langage n'est utilisé que pour la description des ressources web. Chaque ressource RDF est décrite par des triplets RDF <sujet, prédicat, objet>où le sujet représente la ressource, le prédicat représente une propriété et l'objet représente la valeur de la propriété pour le sujet. D'un autre côté, une ressource RDF est unique car les éléments de triplets RDF utilisent généralement des adresses URI (ou IRI) qui sont des identificateurs pour le triplet. Les triplets RDF forment un graphe RDF appelé base de connaissances

Un document RDF est un ensemble de triplets de la forme <sujet, prédicat, objet> Pour stocker, échanger et modéliser ces documents il existe plusieurs sérialisations, citons les plus populaires :

- RDF/XML : c'est la sérialisation la plus utilisé, elle utilise le format XML pour encoder le graph RDF
- Turtle : elle utilise les préfixes, des raccourcis de notation pour condenser l'écriture
- N-TRIPLET : on échange le triplet tout entier, tout est présenté par un URI, à l'exception des nœuds vides et des valeurs.

3- **RDFS :** le RDF est le modèle de donnée pour décrire les ressources web, et le RDFS est un méta modèle pour décrire ces ressources. C'est un langage de définition de vocabulaire pour le RDF, il permet de définir les classes et les sous-classes, déclarer le type des ressources pour les déclarations RDF et il permet aussi de décrire les relations entre les ressources et les propriétés en formulant des contraintes sur les valeurs associées à une propriété pour donner un sens à ces propriétés.

4- **RDFa :** est une recommandation du W3C, il permet de rajouter des informations ou bien des ressources structurées en RDF dans des documents HTML ou XHTML. Il fournit une expression de données structurées dans le XHTML. L'ajout des informations supplémentaires dans les balises XHTML permet de rajouter les sémantiques aux données qui permettent d'échanger les informations.

5- **OWL :** Web Ontologie Langage est un langage du web sémantique conçu pour représenter des connaissances complexes et riches. Il est utilisé par les machines qui traitent le contenu de l'information en représentant explicitement la signification des termes dans des vocabulaires et les relations entre ces termes. Cette représentation est appelée une ontologie.

L'OWL est construit sur le RDFS en utilisant XML. Il permet de définir les classes en utilisant des opérations ensemblistes (intersection, union, restriction...).Il permet de manipuler les propriétés en ajoutant la notion des propriétés inverses, transitives ou bien des restrictions sur ces propriétés en se basant sur la logique descriptive.

OWL propose trois sous langages conçus pour être utilisés par des utilisateurs et des développeurs spécifiques. La relation entre ces trois sous-langages est une relation de nature hiérarchique, toute ontologie OWL Lite valide est une ontologie OWLDL valide, et toute ontologie OWL DL valide est une ontologie OWL Full valide.

OWL Lite : est le sous langage d'OWL le plus simple. Il prend en charge des utilisateurs qui ont besoin d'une hiérarchie de concepts simple et de fonctionnalités de contraintes simples de cardinalité 0 ou 1

OWL DL : il est plus complexe qu'OWL Lite. Il est destiné aux utilisateurs qui veulent une expressivité maximale en concevant la décidabilité et l'exhaustivité de calcul. Il est fondé sur la logique descriptive. Il garantit la complétude des raisonnements

(toutes les inférences sont calculables) et leur décidabilité (leur calcul se fait en une durée finie)

OWL Full : le sous langage le plus complexe. Cependant il permet le plus haut niveau d'expressivité. Il est destiné aux utilisateurs qui souhaitent avoir une capacité de description de haut niveau mais il ne permet pas de garantir la complétude et la décidabilité des calculs.

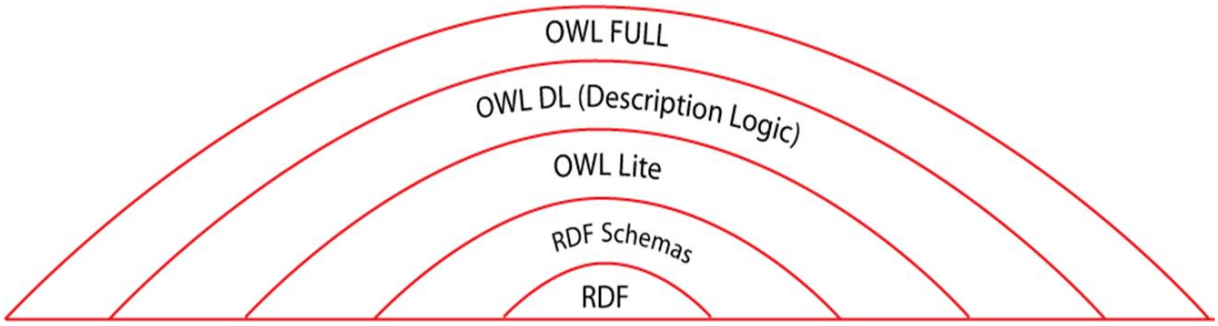


Figure I.3. *Les couches de L'OWL (Belabed, 2019)*

I.4.4 Langages de requêtage des données sémantiques

SPARQL :

« SPARQL Protocol And RDF Query Language » est l'un des trois normes fondamentales du web sémantique avec RDF et OWL, est un langage standard pour faciliter l'interrogation les données de graphes représentés par des triplets RDF. SPARQL c'est (Bernard ESPINASSE, 2019) :

- Un langage de requête pour RDF
- Un protocole : spécification pour émettre et envoyer des requêtes SPARQL (service web) vers des serveurs dédiés et en recevoir les résultats
- Un format XML pour l'affichage des résultats obtenus

I.5 Conclusion

Le travail effectué dans ce chapitre nous a permis de se familiariser avec le vocabulaire utilisé dans le web sémantique et de mieux comprendre ses notions de bases notamment les langages utilisés, la notion des ontologies et l'annotation sémantique. Dans le chapitre suivant nous allons voir « l'ingénierie des exigences ».

Chapitre II : Ingénierie des exigences

II.1 Introduction

Nous présentons dans ce chapitre l'ingénierie des exigences ou bien Requirements engineering dans la littérature anglophone. Mais avant d'entrer dans les détails, nous devons d'abord définir quelques concepts et termes de bases pour la compréhension de ce domaine.

II.2 Définitions

L'ingénierie Système (IS) : selon l'AFIS[1] *« l'ingénierie des systèmes est une démarche méthodologique général qui englobe l'ensemble des activités adéquates pour concevoir, faire évoluer et vérifier un système apportant une solution économique et performante aux besoins d'un client tout en satisfaisant l'ensemble des parties prenantes ».*

Besoin : nous pouvons dire que le besoin c'est la perception de l'utilisateur sur le système. Il s'exprime sous forme de problèmes.

Exigence : selon IEEE 610.12-1990, une exigence est (Pohl,2016) *« une condition ou une capacité qui doit être remplie ou possédée par un système ou un composant de système pour satisfaire un contrat, une norme, une spécification ou d'autres documents formellement imposés »*

Partie-prenante (stakeholder) : c'est toute personne ou une organisation qui peut influencer a un système d'une façon directe ou indirecte

Client : la ou les personnes qui paient le produit et décident généralement (mais pas nécessairement) des exigences. Le client et le fournisseur peuvent être membres de la même organisation (Doe, 2011)

Le fournisseur : la ou les personnes qui produisent le produit pour un client (Doe, 2011)

Le cycle de vie d'un système : c'est les phases dans lesquelles passe le système, commençant par l'émission des besoins jusqu'à le retrait. Ces phases sont : la conception, la

[1] www.afis.fr

réalisation, l'intégration, la production des exemplaires, transfert vers l'exploitation, exploitation et le retrait du service.

II.3 Ingénierie des exigences

Le rôle des exigences dans le développent d'un logiciel est très important. La principale cause de l'échec et de difficultés d'un projet est l'ingénierie des exigences car les défauts dans l'ingénierie des exigences influencent sur toutes les phases de cycle de vie de logiciel. L'objectif d'IE est d'établir une description cohérente, complète et non ambigus des exigences pour un domaine donné en impliquant des parties prenantes.

(Pamela, 1997) a fourni une définition intuitive de l'IE : « *l'ingénierie des exigences est la branche de l'ingénierie logicielle concernée par les objectifs du monde réel, les fonctions et les contraintes des systèmes logiciels. Il s'intéresse également à la relation de ces facteurs avec des spécifications précises du comportement des logiciels, et à leur évolution dans le temps et à travers les familles de logiciel* »

D'une autre part, (Pericles, 1995) a dit que l'ingénierie des exigences « *est le processus systématique d'élaboration d'exigence à travers un processus coopératif itératif d'analyse du problème, documentant les observations résultantes dans une variété de représentations et vérifier l'exactitude de la compréhension acquise* » cette définition ce concentre beaucoup sur le processus de IE

II.4 Processus d'ingénierie des exigences

(Nuseibeh et Easterbrook, 2000) proposent le découpage suivant pour le processus de IE, la première phase est la phase d'élicitation, ensuite la modélisation, l'analyse, la spécification, la validation et enfin la gestion des exigences. Dans les paragraphes suivants nous détaillons chacune de ces phases :

L'**élicitation** des exigences consiste à comprendre le système considéré et collecter ses exigences, les contraintes, les besoins utilisateurs à partir d'une variété de sources. Les parties prenantes impliquées dans ce processus s'accordent sur ces exigences par la négociation de ces derniers.

La **modélisation** est une représentation du problème par des représentations graphiques

L'**analyse** se base sur la compréhension des exigences élicitées pour assurer la meilleure qualité de l'exigence en termes de compréhension, exactitude, et de clarté.

La **spécification** est la phase où l'ingénieur documente et enregistre les exigences pour qu'elles soient utilisables par les stakholders plus précisément par les développeurs qui construisent le système.

La **validation** c'est là où l'ingénieur doit confirmer la qualité des exigences et leur conformité aux besoins utilisateur.

La **gestion** est exécutée tout au long du cycle de vie de l'IE. Cette phase inclut le contrôle de changement, l'évolution des exigences, le contrôle de traçabilité et les versions des exigences et leurs statuts.

II.5 Spécification des exigences logicielles

Une spécification des exigences logicielles (SRS) est une spécification pour un logiciel, programme ou un ensemble de programme particulier qui exécute certaines fonctions dans un environnement spécifique. Le SRS peut être rédigé par un ou plusieurs représentants du fournisseur ou par un ou plusieurs clients ou bien les deux (Doe, 2011). Le SRS décrit ce que le logiciel doit faire et comment il doit fonctionner. C'est une description de l'environnement prévus et de l'objectif de produit logiciel lors de développement.

De toute évidence, pour écrire ce document il existe une manière de rédiger, de structure et de spécifier les exigences Tout document SRS doit être correct, non ambigu, complet, cohérent, consistent, classé pour son importance et/ ou sa stabilité, vérifiable, modifiable, traçable. (Doe, 2011).

Par contre les SRS diffèrent d'une approche à une autre et chaque approche à sa particularité. Il existe des standards pour écrire le document d'exigence, citons ces standards :

- IEEE Standard 1362-1998 Guide for Information Technology-System Definition-Concept of Operations Document ;
- IEEE Standard 830-1998 Recommended Practice for Software Requirements Specifications ;
- Volere Template.

Dans notre travail nous utilisons le Template de Volere pour spécifier les exigences utilisées. La particularité de cette approche par rapport aux autres c'est que ce Template est plus général et peut être utilisé pour la description des problèmes, pour spécifier les exigences utilisateurs et les exigences logicielles. Par contre le standard IEEE 1362-1998 n'est utilisé que pour spécifier les exigences utilisateurs. Finalement, le standard IEEE-830 est utilisé pour spécifier les exigences logicielles.

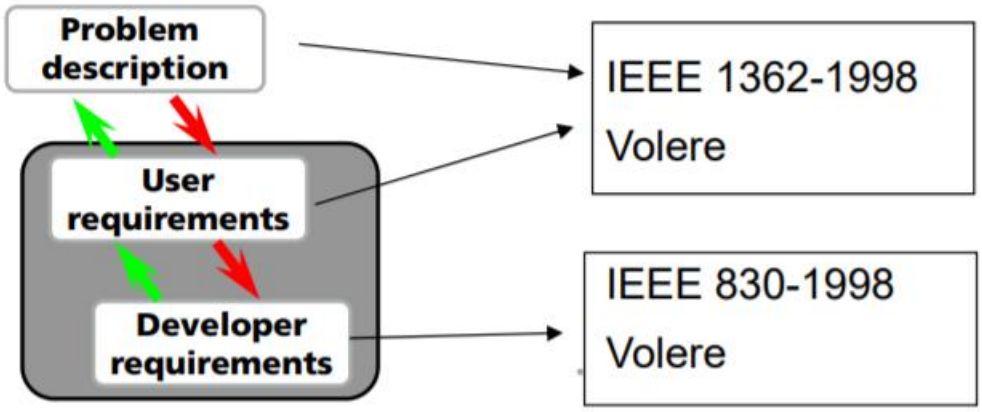


Figure II.1. *L'utilisation des standards SRS dans différents niveaux (Dörr, 2017)*

Dans la section suivante nous détaillons l'approche basée sur le template de Volere Template. (Baouya et al.2019) :

II.5.1 L'approche « Volere requirement »

L'approche Volere [2]est développée par Robertson et James. C'est une approche polyvalente éprouvé et largement utilisée pour l'élicitation des exigences, y compris à la fois le processus d'obtention des exigences ainsi que le format pour les représenter.

L'approche Volere définit 27 sections pour la spécification des exigences. Chacune de ces sections contient une ou plusieurs sous-sections. Au total, Volere définit près de 100 sous-sections pour une spécification complète des exigences.

Parmi celles-ci, 33 sous- sections correspondent à différents types d'exigences. Les autres sous-sections correspondent à d'autres informations auxiliaires (telles que les uses cases).

[2] http://www.volere.co.uk/

Les 33 types d'exigences partagent le même format de représentation, qui est appelé « Shell Exigences ». L'approche Volere définit un modèle de processus pour l'ensemble du processus d'exigences.

II.5.1.a Requirement Shell

La figure II.2 représente le Volere Requirement Shell générique c'est un guide pour écrire les exigences atomiques. Une exigence est constituée de l'ensemble des attributs comme il est présenté dans la figure suivante :

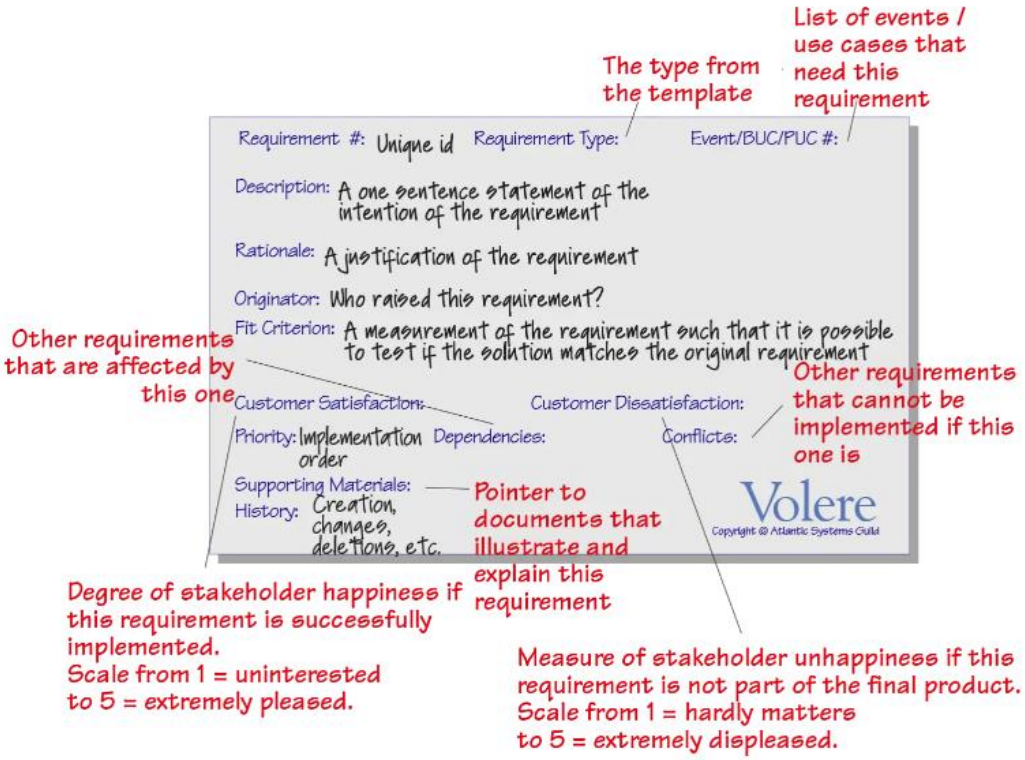


Figure II.2. *Requirements Shell (Volere Requirements Specification Template, 2016)*

Cette approche est globale. Elle peut devenir complexe à mettre en œuvre, pour cette raison il est possible de l'adapter en fonction de la taille du projet, les ressources disponibles et des exigences utilisées.

II.5.1.b Modèle de processus Volere

Selon (Baouya et al, 2019) Le modèle de processus Volere décrit les différentes étapes, acteurs et artefacts qui composent l'exécution de l'approche.

Chaque étape de processus peut être étendu, illustrant que le modèle de processus Volere est plutôt complet et peut devenir complexe à mettre en œuvre. En fonction de la taille du projet, les différentes étapes peuvent être réalisées avec des détails plus ou moins détaillés et plusieurs itérations, ce qui permet d'adapter le processus aux besoins respectifs et aux ressources disponibles. Trois étapes méritent une discussion plus approfondie.

Etape1 : « Prototyper les exigences » il est souvent difficile de définir un ensemble fixe d'exigence pour la mise en œuvre. Surtout pour les produits innovants, c'est parce que (1) les exigences sont souvent difficiles à décrire avec une précision. (2) les exigences sont souvent difficiles à évaluer par rapport à leur mérite relatif. (3) Des exigences qui sont souvent difficiles à équilibrer pour relever ces défis.

Etape 2 : « faire le point sur la spécification » aborde un aspect important que les exigences n'existent pas seuls mais toujours comprises dans le contexte des autres exigences qui ont été définies pour la conception donnée. I.e. que le processus d'exigence ne se termine pas par la création d'exigences individuelles. Une analyse approfondie des exigences tout au long du processus d'exigence et un affinement continu des exigences est primordiale pour obtenir une spécification de haute qualité.

Etape 3 : « la passerelle de la qualité » les exigences peuvent avoir une qualité très variable. Cela comprend à la fois l'exhaustivité de Shell requirement ainsi que la qualité de chaque champ. L'approche Volere recommande une « passerelle qualité » explicite que chaque exigence doit passer avant de pouvoir faire partie de la spécification.

De plus, il existe un schéma spécifique pour formuler les besoins du système : le [composant] doit [faire, fournir, implémenter] [quelque chose] de sorte que [la motivation].

II.6 Conclusion

Le travail effectué dans ce chapitre nous a permis de cerner la notion d'ingénierie d'exigence, ses principes de base et processus de cette ingénierie. Nous sommes aussi arrivés à présenter la spécification des exigences logicielles en particulier selon le template de Volere.

Chapitre III : Etude des systèmes existants

III.1 Introduction

Des nombreux problèmes dans les projets logiciels sont générés dans la phase de l'analyse et la spécification des besoins. Pour cette raison il est nécessaire de fournir un support informatique pour l'ingénierie des exigences pour éviter ces problèmes. Dans ce chapitre nous présentons quelques systèmes qui utilisent l'annotation sémantique dans l'ingénierie des exigences.

Avant de commencer, nous signalons que dans ce chapitre nous nous basons essentiellement sur une seule référence (Ricardo de Almeida Falbo, Carlos Eduardo Correa Braga, 2014).

III.2 L'étude d'un système similaire

Plusieurs outils ont été développés pour prendre en charge l'annotation sémantique tel quel l'infrastructure de gestion des documents sémantiques (IMDS), AktiveDoc, PDFTab, SemanticWord et KIM. Ces outils utilisent des ontologies de domaines pour l'annotation sémantique des documents, et fournissent des fonctionnalités générales pour la gestion des documents sémantiques (annotation, stockage, indexation et récupération de document), par contre aucun de ces outils à l'exception d'IMSD n'a été appliqué pour soutenir l'ingénierie des exigences. Le IMSD permet d'exploiter les éléments de l'ontologie (concepts, relations, propriétés) et de les utiliser pour développer des fonctionnalités spécifiques à un domaine. Dans le système similaire étudié, ils ont appliqués IMSD dans le domaine des exigences logicielles. Ils ont étendu IMSD en introduisant des fonctionnalités spécifiques aux exigences, c'est l'IMSD-Req.

IMSD-Req est une alternative pour fournir un support informatique pour RE, en conservant la culture des organisations qui consiste à utiliser des éditeurs de texte pour documenter les exigences.

IMSD-Req étend IMSD, une plateforme de gestion de documents sémantique à proposition générale, pour inclure des fonctionnalités spécifiques à RE qui visent à répondre

aux opportunités d'amélioration identifiées lors de l'utilisation d'IMSD pour soutenir un processus RE.

Pour répondre à ces opportunités d'amélioration, IMSD-Req explore la conceptualisation fournie par l'ontologie de référence des exigences logicielles SROO illustré dans la figure III.1 afin de fournir des fonctionnalités permettant de prendre en charge l'analyse d'impact des modifications des exigences, évaluation de la cohérence des priorités des exigences, la génération des matrices de traçabilité et de vérification des exigences à l'aide de liste de contrôle. Détaillons ces opportunités dans les paragraphes suivantes ;

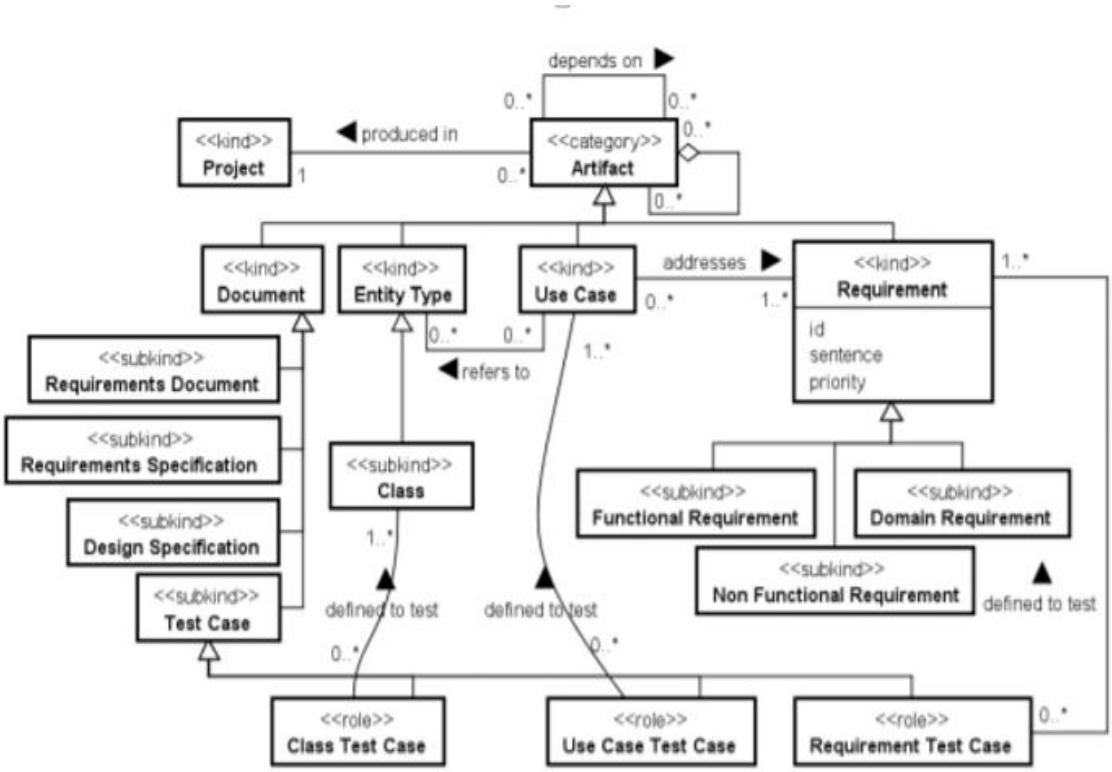


Figure III.1. *Fragment de l'ontologie des exigences logicielles (Ricardo de Almeida Falbon, Carlos E Correa Braga)*

A partir du modèle conceptuel de SRRO illustré à la figure III.1, il est possible de concevoir et coder une ontologie dans OWL qui est utilisée pour annoter sémantiquement les documents d'exigences dans IMSD-Req.

En utilisant IMSD le IMSD-Req introduit de nouvelles fonctionnalités pour

- **Prendre en charge l'analyse d'impact des changements d'exigences** : basée sur la relation « dépend de » et les axiomes qui y sont liés.

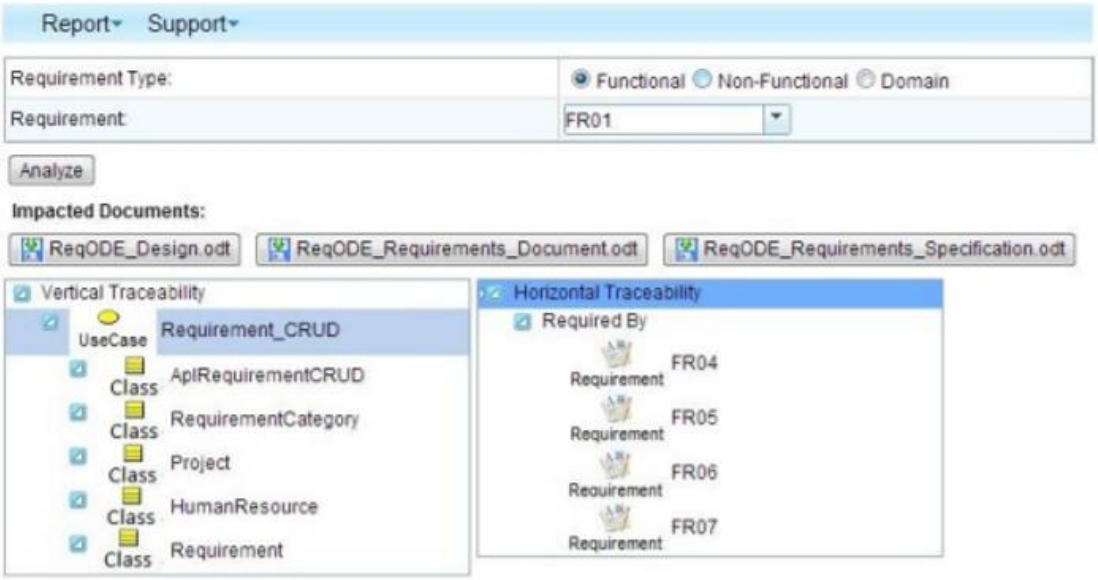


Figure III.2. *L'analyse d'impact des changements d'exigences (Ricardo de Almeida Falbon, Carlos E Correa Braga)*

La figure III.2 représente un exemple qui définit cette fonctionnalité, Sur la base de l'identifiant d'exigence renseigné par l'utilisateur, une liste des documents concernés par une modification de l'exigence correspondante est affichée. Ainsi deux arbres d'analyse d'impact du changement sont présentés : à gauche représente la traçabilité verticale (pour présenter les artefacts directement et indirectement liés à l'exigence), et à droite concerne la traçabilité horizontale.

- **Génération des matrices de traçabilité** : ils ont exploré la relation « dépond de » et les axiomes associés (dans l'ontologie) pour générer des types de matrices de traçabilité. Pour la traçabilité horizontale ils ont généré la matrice Exigence x des matrices de traçabilité des exigences. Et la traçabilité verticale peut être présenté par trois types de matrices : exigences x cas d'utilisation, exigence x classes et exigences x matrices de cas de test. Dans la figure III.3 , ils ont présenté un exemple sur la matrice de traçabilité exigence x cas d'utilisation :

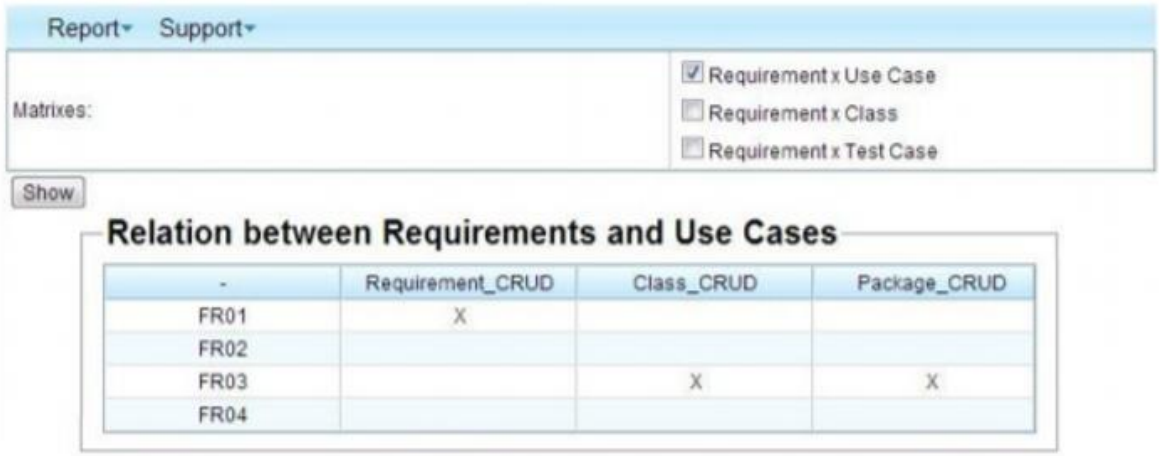


Figure III.3. *Matrice de traçabilité Exigences x Cas d'utilisations (Ricardo de Almeida Falbon, Carlos E Correa Braga)*

- **Evaluation de la cohérence des priorités des besoins** : pour traiter ce cas, ils ont développé une fonctionnalité qui permet à l'utilisateur de hiérarchiser les exigences en tenant compte des priorités des exigences dépendantes. Par exemple si l'utilisateur établit la priorité de l'exigence FR01 comme il est montré dans la Figure III.4, IMSD-Req va récupérer les exigences qui en dépendent de FR01 et leurs priorités. Dans cet exemple l'utilisateur a établi une priorité moyenne à FR01, mais l'une des exigences dépendantes à une priorité élevée. Dans ce cas-là un message d'erreur est affiché, de plus, les valeurs de priorité valides pour ces exigences sont affichées en vert et les non valides en rouge.

Figure III.4*. Evaluation de la cohérence des priorités des besoins (Ricardo de Almeida Falbon, Carlos E Correa Braga)*

- **Vérification des exigences à l'aide de listes de contrôle** : ils ont développé une fonctionnalité qui permet de regrouper et de stocker les requêtes SPARQL à utiliser dans les checklists. Les requêtes doivent être définies pour être utilisées dans les listes

de contrôle. Une fois les requêtes définies, ces questions peuvent être utilisées pour composer plusieurs listes de contrôle. La figure III.5 montre un exemple, c'est une liste de contrôle (les requêtes) pour vérifier les documents de définition des exigences. Lorsque cette liste est appliquée. S'il n'y a pas de problème par exemple tous les cas d'utilisation sont implémentés par une classe, un simple message informatif est affiché, par contre s'il y a un problème avec les exigences par exemple s'il y a des exigences sans priorité établie, les identifiants correspondants à ces exigences sont affichés.

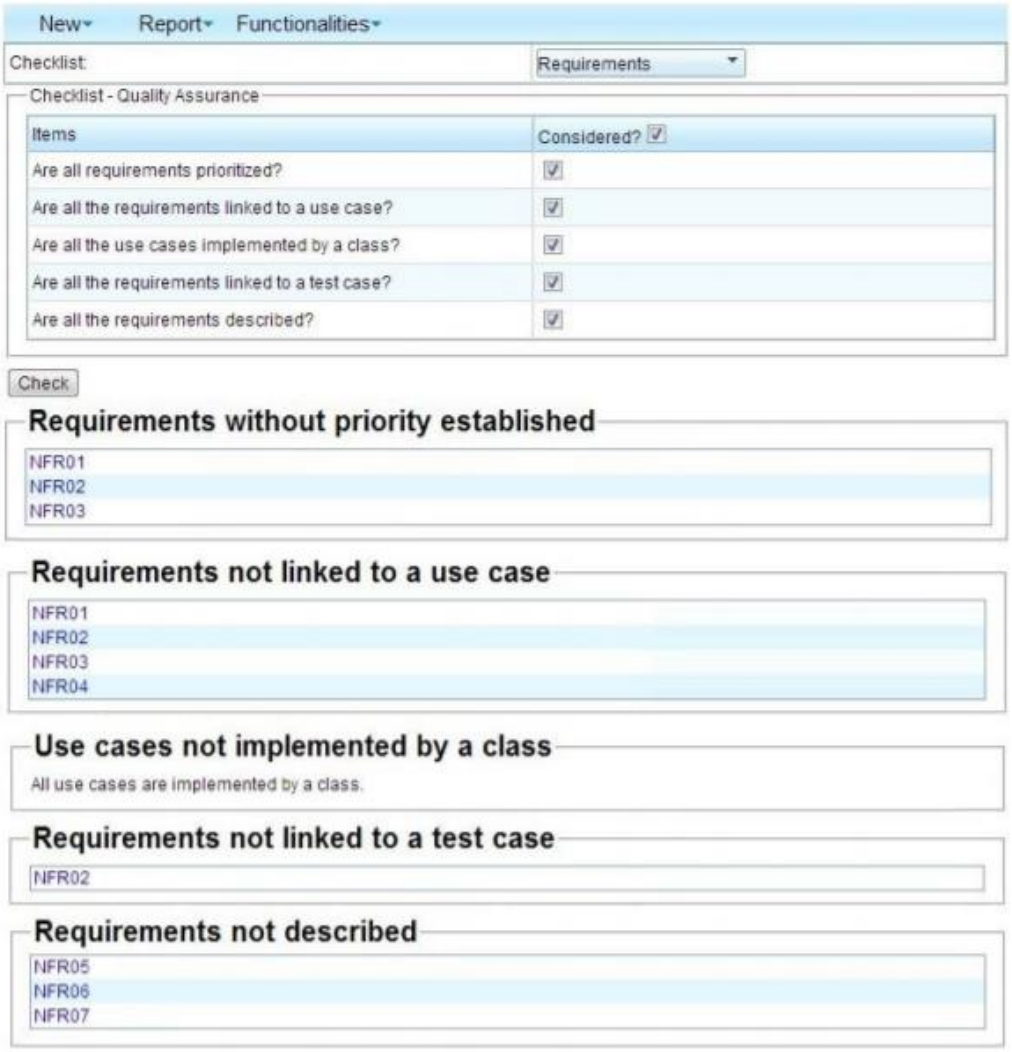


Figure III.5. *Utilisation de Checklists pour la vérification des exigences besoins (Ricardo de Almeida Falbon, Carlos E Correa Braga)*

Concernant la documentation sémantique dans RE, la plupart des travaux utilisent des wikis sémantique. Le wiki sémantique s'est déjà avéré être une plateforme utile pour RE.

Nous discutons dans ce qui suit quelques travaux qui utilisent des wikis sémantiques pour supporter RE, à savoir WikiReq, SOP Wiki, ReqWiki et SoftWiki. Le tableau 1 représente une petite comparaison entre ces travaux.

***Tableau 1**. Comparaison entre les travaux qui utilisent des wikis sémantiques (Ricardo de Almeida Falbon, Carlos E Correa Braga)*

	Basé sur	**Caractéristiques**
WikiReq	Plateforme semantic Mediawiki (SMW)	• A été développé afin d'accompagner les partie prenantes dans l'acquisition d'exigences en terme de concepts définis • Les exigences sont acquises au moyen d'un ensemble de formulaires prédéfinis • Les exigences peuvent être automatiquement transformées en graphiques qui peuvent être utilisés par les développeurs comme entrée de base afin de définir une version formalisée d'une spécification d'exigences • Permettre le débat sur les exigences dans une page à onglet spécifique
SOPWiki	SMW	• Les utilisateurs peuvent ajouter des propriétés aux pages et définir des liens typés entre elles • La possibilité d'exporter le contenu du wiki (ex : les exigences) vers des documents Open Office
ReqWiki	SMW	• Etend la plateforme SMW au moyen d'une ontologie et de services de traitement du langage naturel (NLP) • Destiner à RE basée sur des cas d'utilisation • Son ontologie comprend des concepts tels que les acteurs, les objectifs, les cas d'utilisation, les cas de test, les fonctionnalités et les relations entre eux.

		• Les formulaires sémantiques sont utilisés pour ajouter du contenu au wiki • Annotation automatiquement les spécifications des exigences avec des éléments de l'ontologie • Les requêtes en ligne peuvent être insérées dans les pages wiki
SoftWiki	OntoWiki	• Etend OntoWiki pour prendre en charge l'ingénierie des exigences selon une ontologie pour l'ingénierie des exigences appelée SWORE.

III.3 Conclusion

Le travail effectué dans ce chapitre nous a permis d'avoir une idée sur l'annotation sémantique et l'utilisation de cette notion dans différents systèmes. Avoir une vision globale sur des systèmes existants nous a permis de spécifier la position et le rôle de notre système par rapport aux autres.

Dans le chapitre suivant, nous entamons la première phase de la réalisation de notre système, à savoir la construction des datasets.

Chapitre IV : Construction des datasets

IV.1 Introduction

Dans ce chapitre nous présentons la construction des datasets sur lesquels se base notre système, à savoir le développement de l'ontologie d'ingénierie d'exigence ; le SRS utilisé pour l'annotation et les requêtes SPARQL utilisés. Pour cela, nous présentons d'abord la structure de SRS utilisée pour spécifier les exigences du projet à annoter. Ensuite nous spécifions les motivations derrière la construction de notre ontologie SWOREPLUS qui se base sur l'ontologie SWORE. A la fin nous passons au choix des requêtes.

IV.2 La construction de SRS

L'idée de base de l'annotation sémantique consiste à utiliser un référentiel commun « ontologie » pour enrichir le document à annoter.

Dans notre travail, nous intéressons à annoter sémantiquement des documents SRS écrits en langage naturel (en format Word, PDF).

Comme il est mentionné dans la section **II.5**, il existe plusieurs approches pour écrire un SRS. Dans notre étude nous avons choisi l'approche de Volere pour faire cette tâche (décrire les exigences contenus dans ce document). Car cette approche est globale et largement utilisée pour éliciter les exigences. Cependant il est possible de l'adapter selon les ressources disponibles, la taille et les besoins du projet.

Cette approche contient plusieurs sections pour spécifier les exigences. Parmi celles-ci il y a des sections qui correspondent aux types d'exigences et d'autres sections correspondent à d'autres informations auxiliaires telles que les cas d'utilisations, les stakeholders… (Baouya, et al, 2019).

IV.3 La construction de l'ontologie

L'ontologie est l'un des aspects les plus importants dans l'annotation sémantique. Dans ce paragraphe nous présentons les motivations derrière le développent de notre ontologie.

IV.3.1 Motivations

Selon (Kurt, 2019), pour mettre en place une nouvelle ontologie nous avons quatre choix : (1) utiliser une des ontologies LOD, (2) construire notre propre ontologie, (3) acheter une, (4) ou bien combiner ces solutions. Concernant le premier choix, selon la même personne (Kurt, 2019) pour utiliser des ontologies LOD il faut aligner les schémas des données produits par les applications à cette ontologie et ceci a était écartée car il dépasse la limite de notre travail. L'option d'acheter une a était aussi écartée vu les contraintes de notre projet. Enfin puisque il existe une ontologie publique similaire qui peut répondre à nos besoins il n'y a pas un meilleur choix que d'utiliser un sous-ensemble de cette ontologie en l'adaptant à nos besoins. Pour effectuer cette tâche nous avons utilisé l'ontologie SWORE.

SWORE est une ontologie existante qui définit les concepts de base de l'ingénierie des exigences, elle définit la manière dont ils interdépendants. Pour construire notre ontologie nous avons adapté SWORE avec nos besoins (nous avons éliminé, ajouté, renommé certains composants de l'ontologie et même changé le type de quelque unes). Cette nouvelle ontologie nous l'avons appelé SWORE-PLUS. Nous avons fait cette adaptation car notre travail se base uniquement sur la manipulation des exigences fonctionnelles et non fonctionnelles donc nous devons être plus précis d'une part. D'une autre part, nous utilisons le document SRS basé sur l'approche VOLERE pour faire l'annotation. Cette norme définit un ensemble des caractéristiques des exigences. Pour cette raison nous allons obligé de détailler plus le concept de « Requirement » pour équilibrer et basculer entre les deux documents (SRS et ontologie). La figure IV.1 illustre les éléments que nous avons extraits de SWORE.

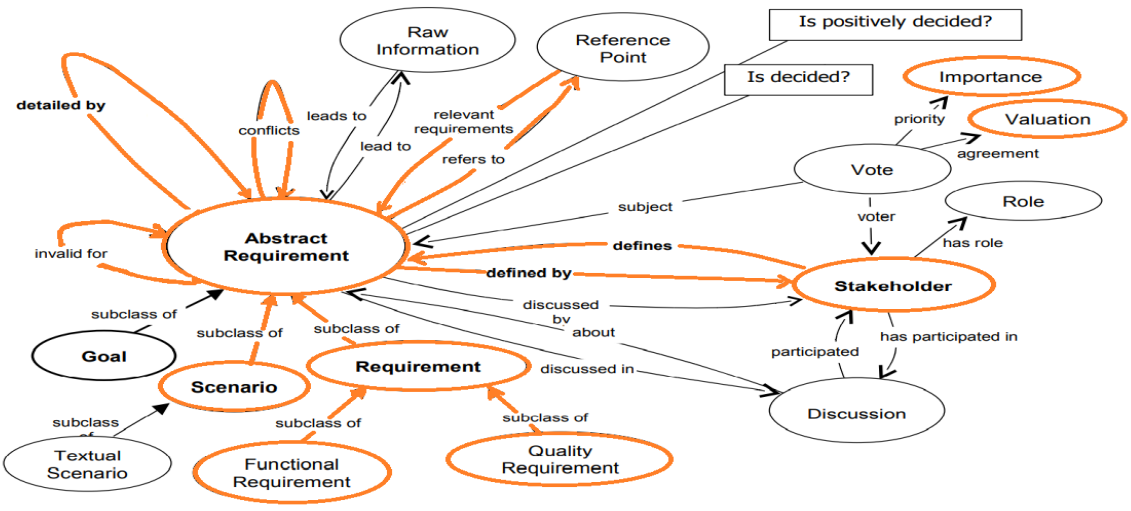


Figure IV.1. *SWORE Adapté*

Apres cette adaptation, la nouvelle ontologie est représenté par la structure suivant

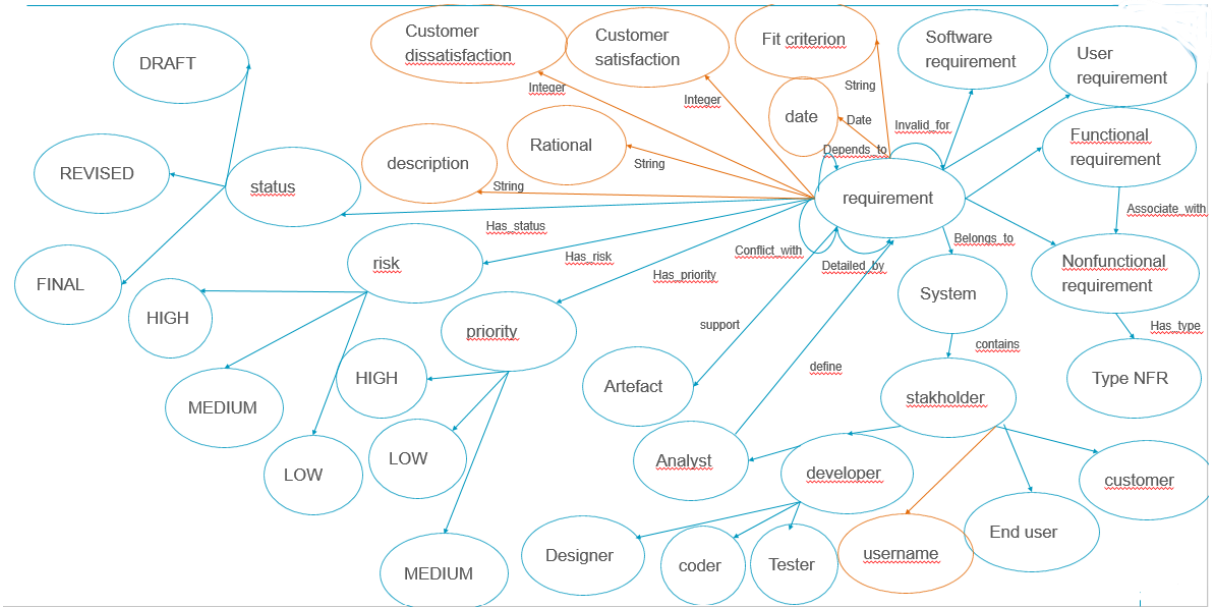


Figure IV.2. *SWOREPLUS*

IV.4 Construction de document de requêtes

Apres l'annotation sémantique, nous nous intéressons à faire la recherche sémantique sur le SRS annoté.

Pour faire la recherche sémantique, nous devons interroger les données représentées par les triplets RDF que nous avons annotés. Pour ce faire, nous utilisons des requêtes SPARQL.

Nous fournissons une liste des requêtes génériques qui sont utiles pour l'ingénieur d'exigences. Evidement cette liste de requête n'est pas censée être complète. Ainsi nous pouvons y rajouter d'autres requêtes. Nous détaillons dans les chapitres suivants ces requêtes et comment elles peuvent être utilisées.

IV.5 Conclusion

Après la construction des datasets en choisissant une norme pour représenter le SRS, le choix de l'ontologie et le choix de présentation des requêtes, nous proposons une architecture du système dans le chapitre suivant.

Chapitre V : Conception générale

V.1 Introduction

Dans ce chapitre nous présentons l'architecture générale du notre système, après nous détaillons chaque composant dans cette architecture.

V.2 Architecture du système

L'architecture du système proposé, intitulé « Semantic Requirements Tool» (SRT), pour faire l'annotation et la recherche sémantique est illustré dans la figure V.1 :

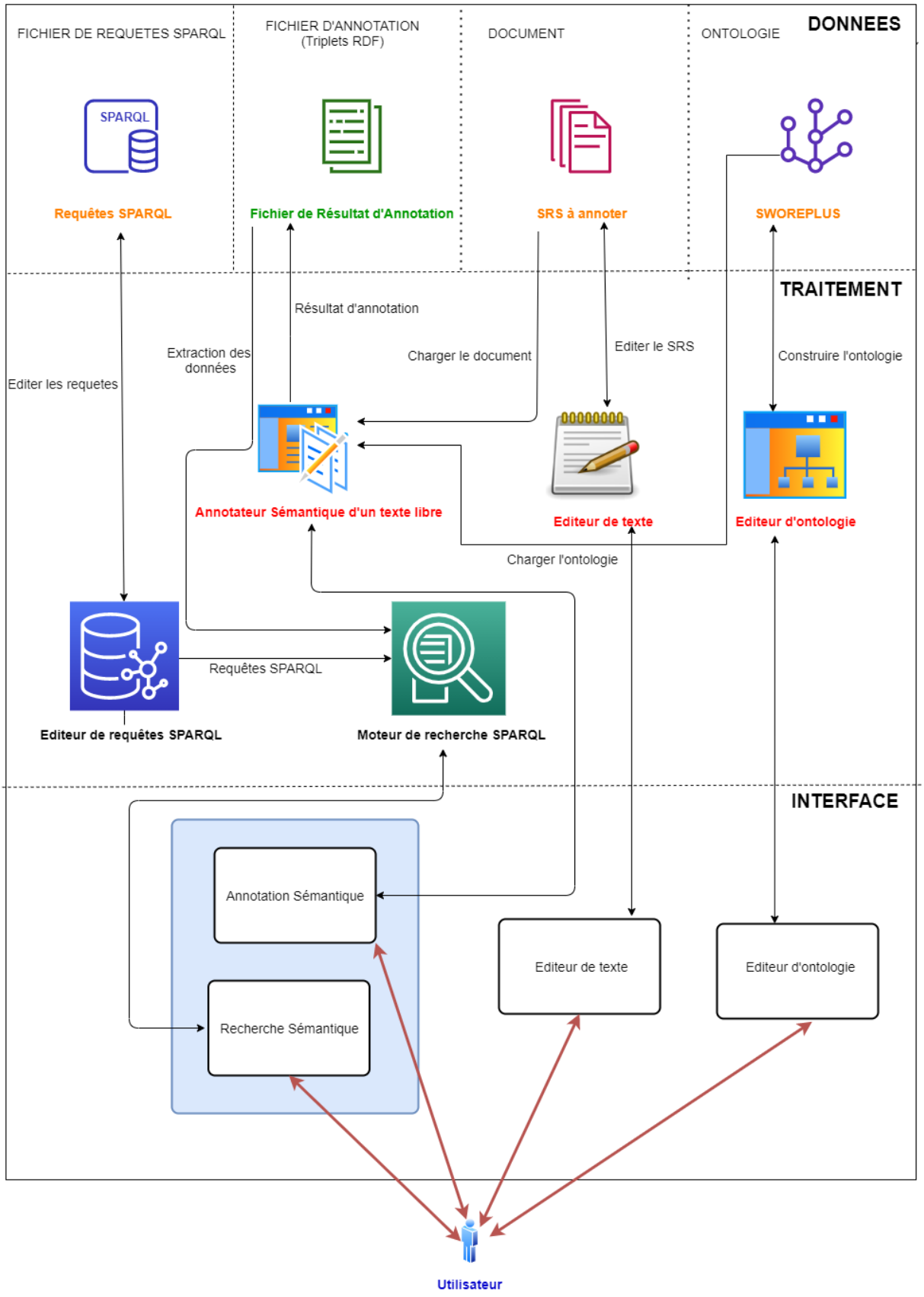


Figure V.1. *Architecture Générale*

Notre système est composé de trois niveaux, niveau de données, niveau de traitement et niveau de résultats.

Données : contient les documents d'entrés et de sortie dans notre système.

Les documents d'entrée sont :

- l'ontologie : représente l'ontologie SWOREPUS qui va être édité par éditeur d'ontologie
- le document SRS à annoté : représente le document SRS que nous allons annoter.
- Fichier de requêtes SPARQL : Le fichier qui contient des requêtes SPARQL

Le document de sortie est :

- Le fichier d'annotation : le ficher RDF qui contient des informations concernant l'annotation sémantique de document SRS en utilisant l'ontologie.

Traitement : contient les composants de traitement à savoir les composants utilisés pour la construction du dataset (éditeur de l'ontologie, éditeur de texte), l'annotation sémantique (annotateur sémantique d'un texte libre) et la recherche sémantique (éditeur de requêtes et moteur de recherche SPARQL).

- Editeur d'ontologie : pour éditer l'ontologie
- Editeur de texte : pour éditer les SRS
- Annotateur sémantique d'un texte libre : représente le sous-système qui permet de faire l'annotation sémantique
- Editeur de requête SPARQL : c'est le composant qui permet d'éditer les requêtes SPARQL
- Moteur de recherche SPARQL : c'est le composant qui permet de faire l'interrogation et la recherche sémantique

Interface : inclut les interfaces d'édition (de l'ontologie et le SRS), et l'interface utilisateur de notre système avec lesquels on interroge l'utilisateur.

L'ontologie est éditée par un éditeur d'ontologie. D'autre part, le SRS est édité par un éditeur de texte. Cette ontologie va être chargée par l'annotateur sémantique avec le document SRS à annoter pour faire l'annotation sémantique.

La sortie de cette annotation sera un fichier RDF qui contient les triplets concernant l'annotation.

Les requêtes SPARQL génériques sont stockées dans un document SPARQL. Ces requêtes vont être éditées via un éditeur de requête.

A la fin, le moteur de recherche SPARQL interroge le fichier d'annotation sémantique en utilisant les requêtes SPARQL pour exécuter ses requêtes et afficher à l'utilisateur les résultats de sa recherche concernant le SRS annoté.

V.3 Conclusion

Le travail effectué dans ce chapitre est le plus important après la réalisation du système lui-même. Nous avons décrit d'une manière schématique les éléments qui construisent notre système, leurs interactions et les relations entre eux.

Dans le chapitre suivant, nous détaillons chaque élément de l'architecture de notre système dans la phase de la conception détaillée.

Chapitre VI : Conception détaillée

VI.1 Introduction

Dans ce chapitre nous détaillons les composants du notre système présenté dans le chapitre précédent afin d'obtenir une forme de notre système prête à être implémentée.

VI.2 La conception des composants

VI.2.1 L'ontologie

VI.2.1.a Présentation du contexte

Une exigence est une attente auquel un produit (ou un service) doit répondre, ou une contrainte qu'il doit satisfaire. Elle peut être une exigence fonctionnelle qui définit une fonctionnalité du système à développer, ou bien une exigence non fonctionnelle qui caractérise une qualité ou une propriété désirer de ce système.

Une exigence peut soit dépendre d'une autre exigence soit être en conflit avec une autre, ou bien être détaillée par une autre exigence.

Certaines exigences fonctionnelles peuvent être associées à une ou plusieurs exigences non fonctionnelles.

D'autre part, chaque exigence a un niveau d'exigence, nous distinguons 2 niveaux : Software Requirements, et User Requirements. Chacun d'eux se concentre sur un aspect différent du problème à résoudre.

Toute exigence appartient à un système. Ce système contient aussi des stakholders qui sont la source principale de l'exigence.

Un stakholder peut être soit un Customer qui demande le système (produit, service), soit un End User qui utilise le système soit enfin un développeur qui produit le système.

Le développeur peut être « Analyst », « Designer », « Coder » ou « Tester ». L'analyste est celui qui définit et vérifie l'exigence.

Une exigence est prise en charge à travers un artefact. Cet artefact est contenu dans un document qui contient un scenario, une liste d'exigences….

VI.2.1.b Processus d'affinement de l'ontologie

Le processus de développent de l'ontologie est non linéaire. Nous ne pouvons pas dissocier les étapes de ce développement. En effet, plusieurs allers-retours sont nécessaires car il n'est pas possible de prédire dès le début qu'un terme va jouer le rôle d'une classe ou d'une propriété. Ce qui implique que plusieurs modifications seront effectuées dans ce sens. Ainsi il n'est pas facile de savoir si les termes collectés sont suffisants pour construire l'ontologie. Alors nous devons soit rajouter des termes si nécessaire ou bien en retirer d'autres qui sont inutiles.

Mais en général, pour représenter l'ontologie, nous devons construire :

- Une liste des concepts : pour représenter les classes ;
- Une liste des attributs : pour représenter les propriétés de données
- Une liste de relations : pour représenter les propriétés d'objet
- Une représentation hiérarchique des concepts

VI.2.1.c Présentation de l'ontologie SWOREPLUS

Le vocabulaire utilisé dans l'ingénierie des exigences nous permet de définir le vocabulaire de l'ontologie. Ce vocabulaire nous permet d'extraire les primitives constituant les concepts de cette ontologie.

a Les classes

- La classification requirement classe les exigences selon leurs types, nous distinguons deux sous classes, les exigences fonctionnelles et les exigences non fonctionnelles.
- La class système définit le système sur lequel se base ce travail.
- La classe stackholder regroupe les parties prenantes que nous pouvons trouver dans un projet logiciel. Nous distinguons trois sous classe dans cette classe : la classe customer, la classe end_user et la classe developper. La classe developper contient elle-même quatres sous classe : analyst, designer, coder, tester.
- La classe artefact pour présenter les artefacts
- La classe document pour présenter les documents utilisés dans le développement du système.

La hiérarchie ci-dessous représente la hiérarchie des classes (figure VI.1) :

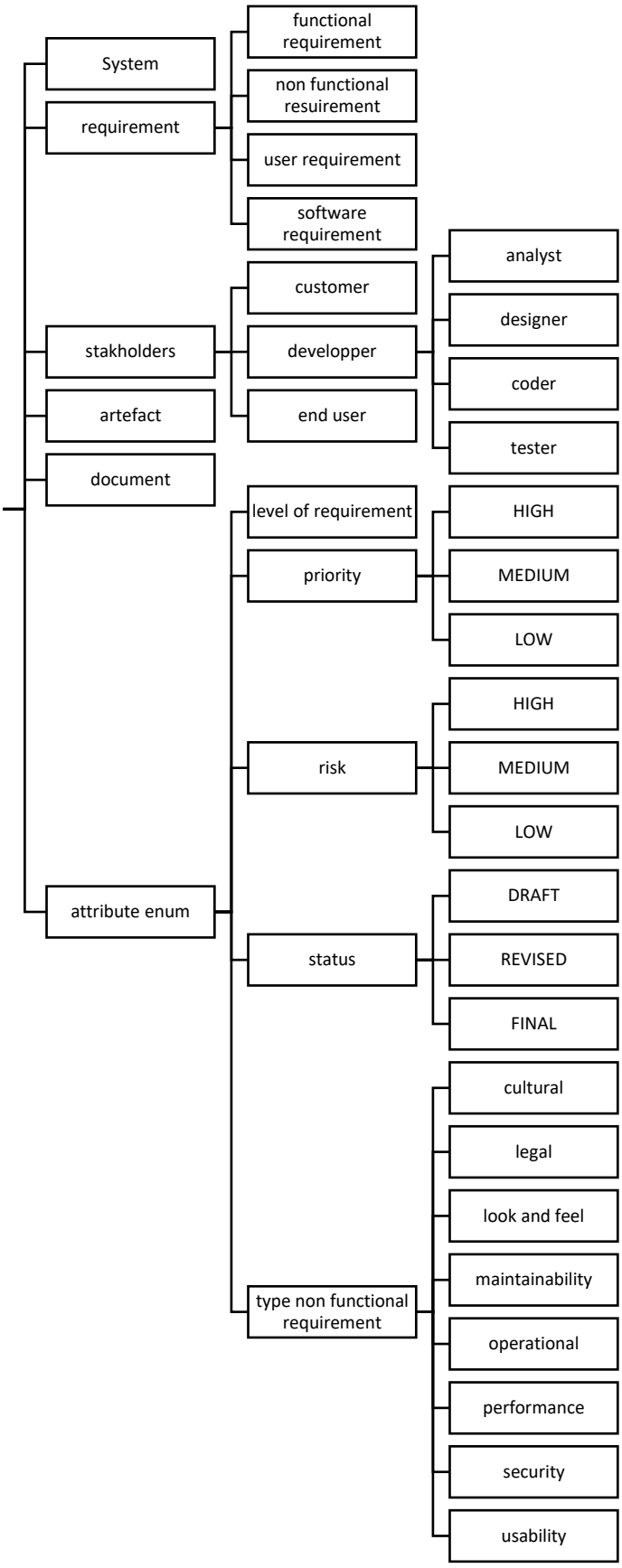


Figure VI.1. *La hiérarchie des classes de l'ontologie SWOREPLUS*

b *Les propriétés des classes*

Les classes seules ne donnent pas assez d'informations. Pour cette raison nous complétons la définition des classes par la définition de la structure interne des concepts.

Avant définir les propriétés d'objet et les propriétés de type de données nous clarifions que pour définir une propriété (d'objet ou type de donnée) nous devons définir leur domaine qui définit les classes dont les instances peuvent être « sujet » au prédicat du type de la propriété définie ; et définir leur « range » qui définit les classes dont les instances peuvent être « objet » au prédicat du type de la propriété définie.

b.a Les propriétés d'objet

Les propriétés d'objet ou les « objects properties » dans la littérature anglophone, sont des propriétés qui relient les instances de classes aux instances d'une autre classe. Le tableau2 montre les propriétés d'objets utilisés dans SWOREPLUS

Tableau 2. *Les propriétés d'objet*

Domain	Object properties	Range
Functional_requirement	Associate_with	Non_functional_requirement
Requirement	belongs_to	System
Analyst	Check	requirement
Requirement	Conflict_with	Requirement
System	Contains	Stackholders
Analyst	Define	Requirement
Requirement	Depends_to	requirement
Requirement	Detailed_by	Requirement
Requirement	Invalid_for	Requirement
Requirement	Support	Artefact
non_functional_requirement	has_type_non_functional_requirement	type_non_functional_requirement
Requirement	has_status	status
Requirement	has_risk	risk
Requirement	has_priority	priority
Requirement	has_level	Level_of_requirement

Les propriétés de type de données (data properties) relient une instance à des valeurs. Dans cette ontologie nous intéressons à représenter les valeurs possibles pour définir les caractéristiques d'une exigence. Le tableau 3 illustre la liste des propriétés de type de données utilisé dans notre ontologie.

Tableau 3. *Les propriétés de donnée*

Domain	Data properties	range
Requirement	Description	String
Requirement	Fit_criterion	String
Requirement	Rational	String
Requirement	Customer_satisfaction	Integer
Requirement	Customer_dissatisfaction	Integer
Requirement	Date	dateTime
Document	Name_document	String
Artefact	Type_artefact	String
Stackholder	Username	String

c Les restrictions

Une restriction est utilisée sur des propriétés (d'objet ou de donnée), elle permet de définir une classe par une contrainte qui porte sur des instances de la classe. Cette contrainte s'exprime par une restriction sur les types des valeurs possibles d'une propriété (Gandon et all, 2015). Le tableau 4 montre la liste des restrictions utilisées dans notre ontologie.

Tableau 4. *Les restrictions*

Les classes	Les restrictions
requirement	has_priority **only** (HIGH **or** LOW **or** MEDIUM)
requirement	has_risk **only** (HIGH **or** LOW **or** MEDIUM)
Requirement	Has_status **only** (DRAFT **or** FINAL **or** REVISED)
Functional_requirement	associate_with **some** non_functional_requirement
non_functional_requirement	has_type_non_functional_requirement **only** (cultural **or** legal **or** look_and_feel **or** operational **or** performance **or** security **or** usability)
software_requirement	has_level **value** software_requirement
user_requirement	has_level **value** user_requirement
analyst	check **some** requirement
analyst	define **some** requirement

VI.2.2 Le document SRS

L'approche Volere contient plusieurs sections pour spécifier les exigences. (Sections d'exigences et d'autres sections telles que les cas d'utilisations, les stakeholders…)

Dans notre travail, nous nous intéressons uniquement aux sections qui correspondent aux exigences fonctionnelles et non fonctionnelles. Les autres sections dépassent la limite de ce travail. Évidemment, nous adoptons Volere selon notre besoin. Pour cette raison, le requirement shell utilisé dans SRT diffère légèrement du générique présenté dans la figure II.2

Le requirement shell dans SRT inclut les attributs suivants (Baouya, et al, 2019). :

Id décrit un identifiant unique pour chaque exigence ;

Description décrit l'exigence ;

Rational définit une justification de l'exigence ;

Fit criterion définit une mesure de l'exigence (telle qu'il est possible de tester si la solution correspond à l'exigence d'origine) ;

Dependies définit la liste des exigences qui dépend de celle-ci ;

Conflicts définit la liste des exigences qui sont en conflit avec celle-ci ;

Type le type de l'exigence si elle est non fonctionnelle ;

Priority définit l'importance de l'exigence. Peut avoir 3 valeurs : HIGH, MEDIUM, LOW.

Risk définit le degré de risque de l'exigence. Elle peut avoir aussi les trois valeurs : HIGH, MEDIUM, LOW ;

Status définit l'état de l'exigence si elle est en brouillon, révisé ou bien validé. Elle peut prend 3 valeurs : DRAFT, REVISED, FINAL.

VI.2.3 Le fichier d'annotation

La sortie de l'annotation sémantique est un fichier qui contient des triplets RDF correspondants. Le sujet dans le triplet représente un texte annoté dans le document. Le prédicat représente la propriété dans l'ontologie. L'objet représente l'instance. Les valeurs possibles pour ces triplets sont illustrées dans la figure VI.2.

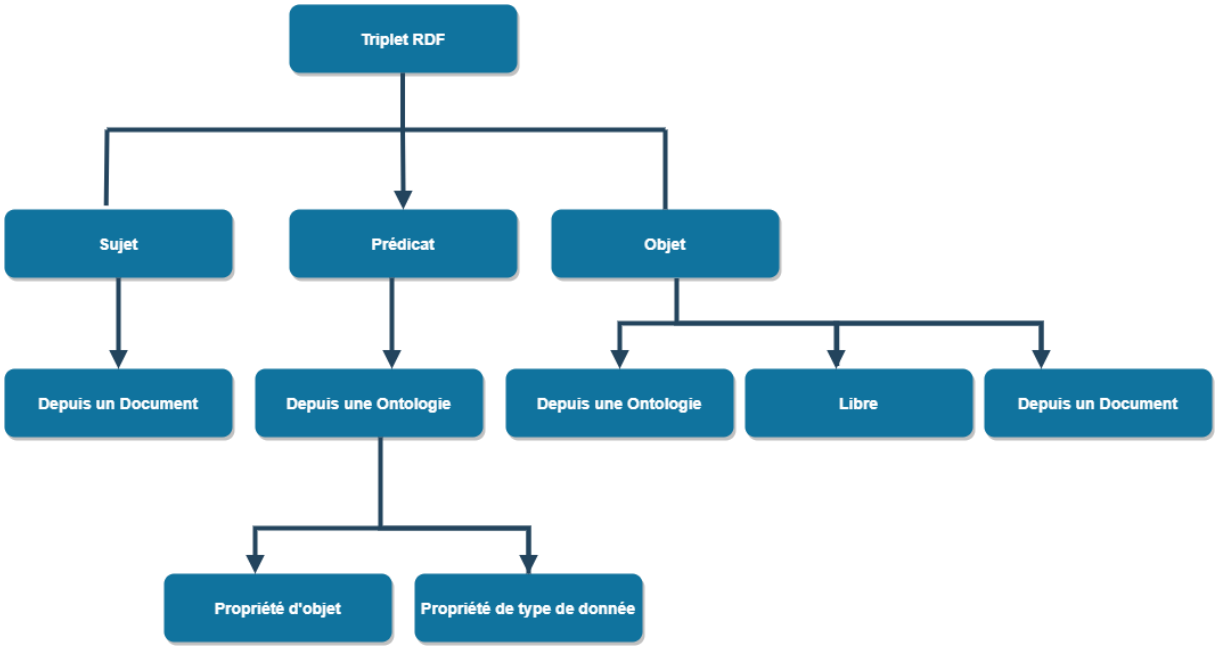


Figure VI.2. *Les cas possibles pour les valeurs de triplets RDF*

VI.2.4 Le fichier de requêtes

Pour faire la recherche sémantique, nous fournissons une liste des requêtes génériques. Nous présentons les requêtes suivantes dont le résultat sera affiché :

- Toutes les exigences fonctionnelles
- Toutes les exigences non fonctionnelles
- Les exigences qui sont en conflit
- Les exigences couteuses (les exigences qui ont un risque élevé ou en conflit)
- Les exigences qui sont dépendantes
- Les exigences importantes
- Les exigences non encore validées
- Les exigences validées
- Les exigences risquées
- La traçabilité d'une exigence
- Les exigences fonctionnelles avec leurs exigences non fonctionnelles associées

- Une recherche avancée (en spécifiant le type d'exigence, sa priorité, son risque, son statut)

Les pluparts de ces requêtes sont capturées dans l'ontologie (par exemple les exigences risquées). Cependant nous proposons autres requêtes qui utilisent le raisonnement (par exemple les exigences coûteuses, la recherche avancée).

Évidement cette liste de requête n'est pas censée être complète, et nous pouvons y ajouter d'autres requêtes.

VI.2.5 L'annotateur sémantique

L'annotation sémantique est l'étape la plus importante dans la réalisation de notre système. Mais l'implémentation d'un système qui permet de faire l'annotation sémantique à partir de zéro dépasse la limite de notre projet à cause de différentes contraintes. Ce qui nous a motivés à intégrer un système prêt qui permet de réaliser cette tâche.

VI.2.6 Editeur d'ontologie

L'ontologie SWOREPLUS est éditée avec l'éditeur d'ontologie, raisonner en utilisant le raisonneur Harmit, et implémentée avec les techniques OWL DL parce que, OWL DL permet d'exprimer des cardinalités multiples et d'une autre part, les autres langages (RDFS, OWL Lite, OWL Full) sont insuffisants ou plus complexe.

Le codage de l'ontologie avec OWL permet de rendre l'ontologie réutilisable grâce à l'utilisation des propriétés de disjonction et d'équivalence entre les concepts et les relations.

VI.2.7 Editeur de texte

Dans notre travail nous nous intéressons à utiliser le SRS écrit en langage naturel, ce qui implique que pour présenter et éditer ce dernier nous utilisons n'importe quel format du fichier SRS (Word, PDF, Excel, …) et n'importe quel éditeur de texte qui soit adéquat.

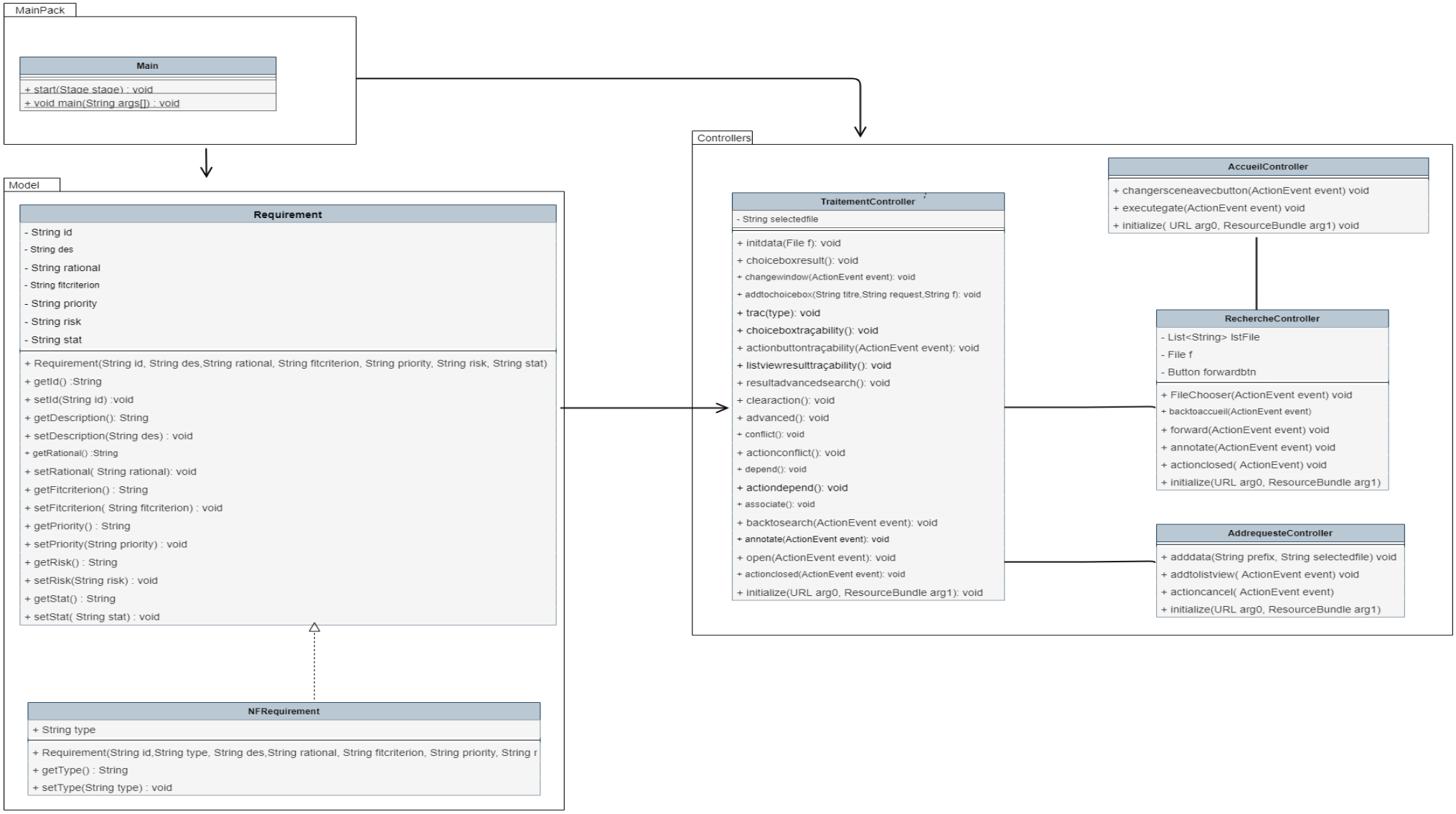
VI.2.8 Editeur de requêtes

Apache Jena est un Framework open source pour la création d'application de web sémantique. Ce Framework est composé de différentes API interagissant entre elles pour traiter les données RDF[3] .Dans notre travail, pour manipuler les requêtes SPARQL, nous avons utilisé cette API pour aider à écrire du code java qui gère SPARQL.

VI.2.9 Moteur de recherche SPARQL

Pour modéliser le moteur de recherche, nous présentons le diagramme de classe suivant (figure VI.3) :

[3] Apache Jena : https://jena.apache.org/



FigureVI.3. *Diagramme de classes*

VI.3 Conclusion

Après la conception, il ne nous reste qu'à implémenter notre système. Dans le chapitre suivant nous présentons cette phase.

Chapitre VII : Implémentation

VII.1 Introduction

L'implémentation est la phase la plus importante. Le choix des outils de développement influence beaucoup sur la flexibilité du système.

Pour réaliser cette tâche ; nous commençons d'abord par la description de l'environnement de travail, ensuite nous passons à la réalisation des composants du notre système.

VII.2 Environnement de travail

VII.2.1 Environnement matériel

Notre environnement matériel est caractérisé par :

- Système d'exploitation : Windows 10 Professionnel
- CPU : Intel® Core™ i5-8250U 1.60GHz
- Mémoire : 8 Go

VII.2.2 Environnement logiciel

L'environnement logiciel est composé des éléments suivants :

VII.2.2.a L'environnement de développement d'ontologie

L'ontologie SWOREPLUS est éditée avec l'éditeur d'ontologie Protégé (version 5.5.0) et implémentée avec les techniques OWL DL parce que, OWL DL permet d'exprimer des cardinalités multiples et d'autre part, les autres langages (RDFS, OWL Lite, OWL Full) sont insuffisants ou plus complexe.

Le codage de l'ontologie avec OWL permet de rendre l'ontologie réutilisable grâce à l'utilisation des propriétés de disjonction et d'équivalence entre les concepts et les relations. Finalement, l'ontologie SWOREPLUS est vérifiée à l'aide de raisonneur HermiT 1.4.3.456.

VII.2.2.b L'environnement pour l'annotation sémantique

Pour faire l'annotation sémantique, nous avons utilisé GATE.

GATE (General Architecture for Text Engineering) est un logiciel libre open source, il est utilisé pour tous les types de taches de calcul impliquant le langage humain. Excelle dans l'analyse de texte de toutes formes et tailles.

GATE offre une gamme des applications. (1) il est développé pour inclure un client de bureau pour les développeurs en utilisant GATE Developer, (2) une application web basée sur un flux de travail en utilisant GATE Teamware, (3) une bibliothèque java en utilisant GATE Embadded.

La solution idéale est d'utiliser le GATE Embadded. Nous aimerons bien utiliser cette API mais à cause de différentes contraintes nous avons écarté cette solution pour présenter notre solution et nous avons intégré GATE Developper dans notre système pour faire l'annotation sémantique.

Pour manipuler et modéliser les ontologies et faire l'annotation sémantique, GATE fournit une API. Cette dernière est livrée avec deux plugins Ontology et Ontology_Tools qui offrent des outils pour éditer et utiliser les ontologies.

Le plugin Ontology : contient une implémentation de l'API de l'ontologie, avant qu'une fonctionnalité basée sur l'ontologie puisse être utilisée, le plugin doit être chargé dans GATE.

Le plugin Ontology_Tools : ce plugin offre « The ontology Annotation Tool » qui permet aux utilisateurs de faire une annotation manuelle de texte avec une ou plusieurs ontologies.

VII.2.2.c L'environnement pour SRT et la recherche sémantique

Pour réaliser cette tache nous avons réalisé notre propre programme en utilisant l'environnement suivant :

- Java 15.0.1 comme langage de programmation pour implémenter notre système car il est le plus adapté
- Eclipse 4.18.0 comme outil de développement
- API Apache Jena 4.0.0 pour manipuler les requêtes SPARQL
- JavaFX 16 et Scene builder 8.5.0 pour réaliser notre interface

VII.3 Réalisation du système

VII.3.1 Implémentation de l'ontologie

Dans ce qui suit nous allons présenter comment nous avons édité cette ontologie à partir du lancement jusqu'à la génération du code OWL.

1- Création d'un nouveau projet

Après le lancement du Protégé nous enregistrons d'abord le projet pour définir le nom de fichier OWL en sortie et le format d'implémentation (figure VII.1, VII.2 respectivement).

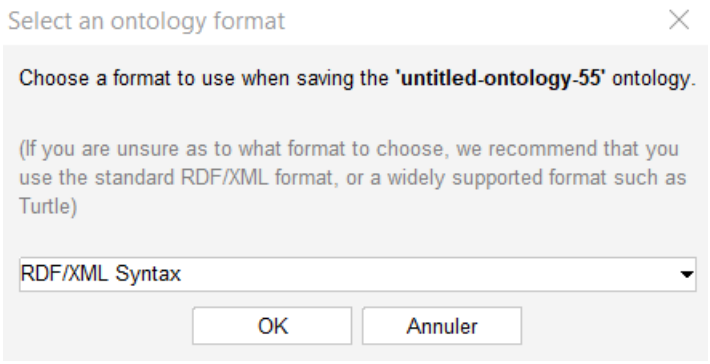
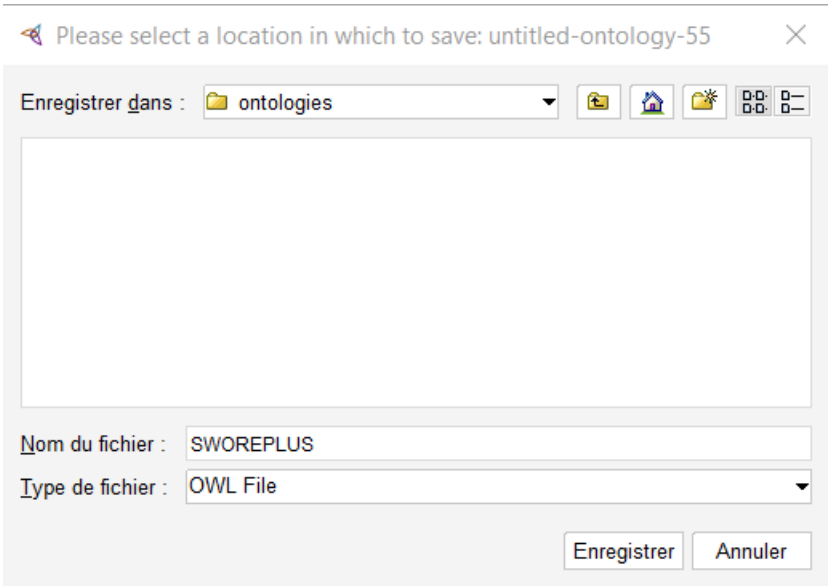
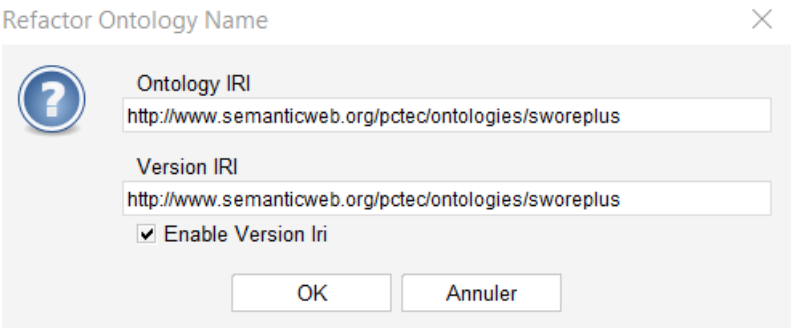


Figure VII.1. *Le choix du format de l'ontologie*



FigureVII.2. *Le choix d'emplacement de sauvegarde de l'ontologie*

Après la sauvegarde du projet, nous allons éditer l'URI de l'ontologie (figure VII.3) :



FigureVII.3. *Édition l'URI de l'ontologie*

Après avoir spécifié les propriétés du projet, nous commençons par la création des classes, et nous définissons les classes disjointes (figure VII.4).

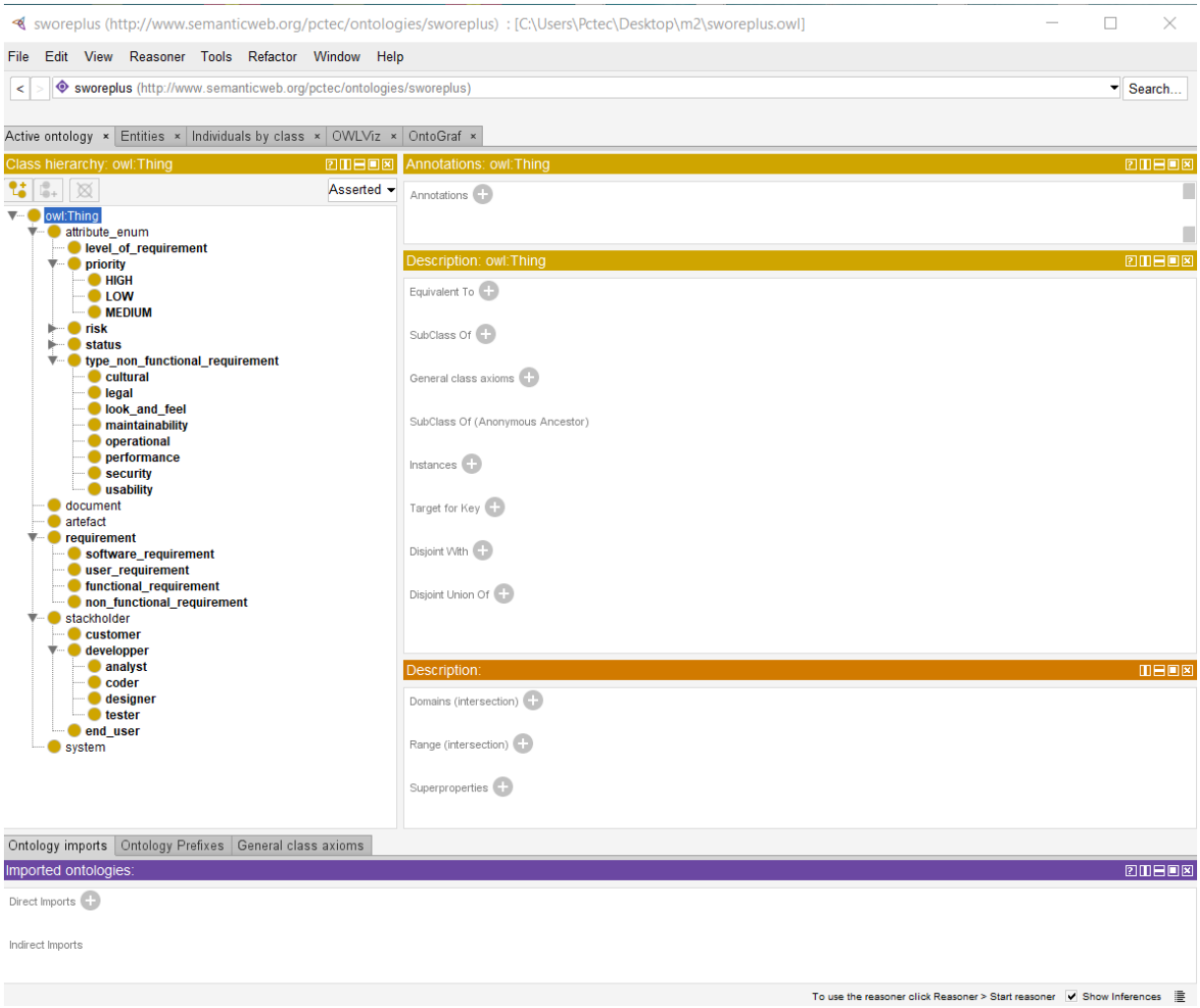


Figure VII.4. *L'édition des classes de l'ontologie*

Maintenant, nous pouvons définir les propriétés et les relations. Nous commençons par les objects properties (figure VII.5).

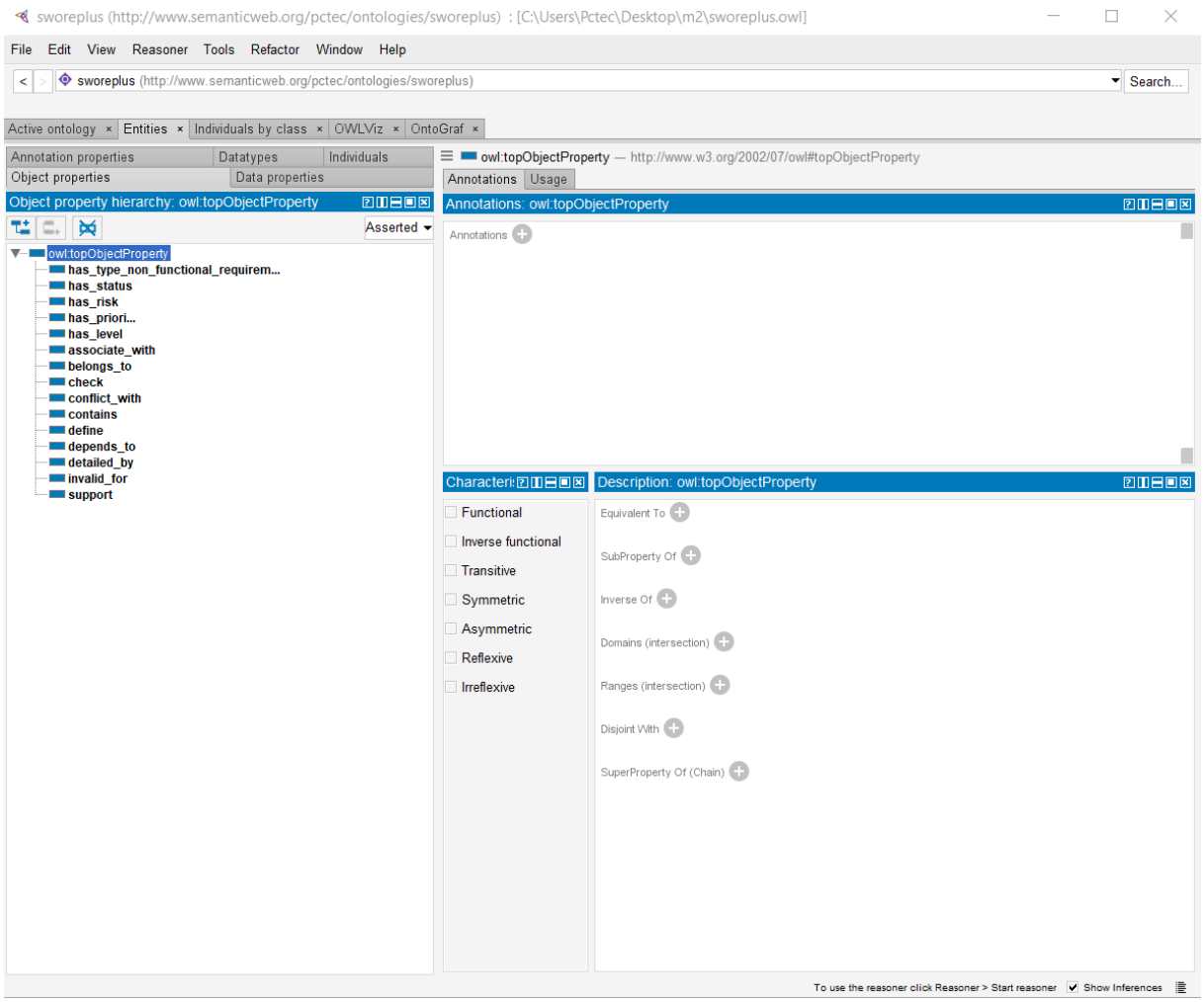


Figure VII.5. *L'édition des propriétés d'objet*

Ensuite nous définissons les data properties (figure VII.6).

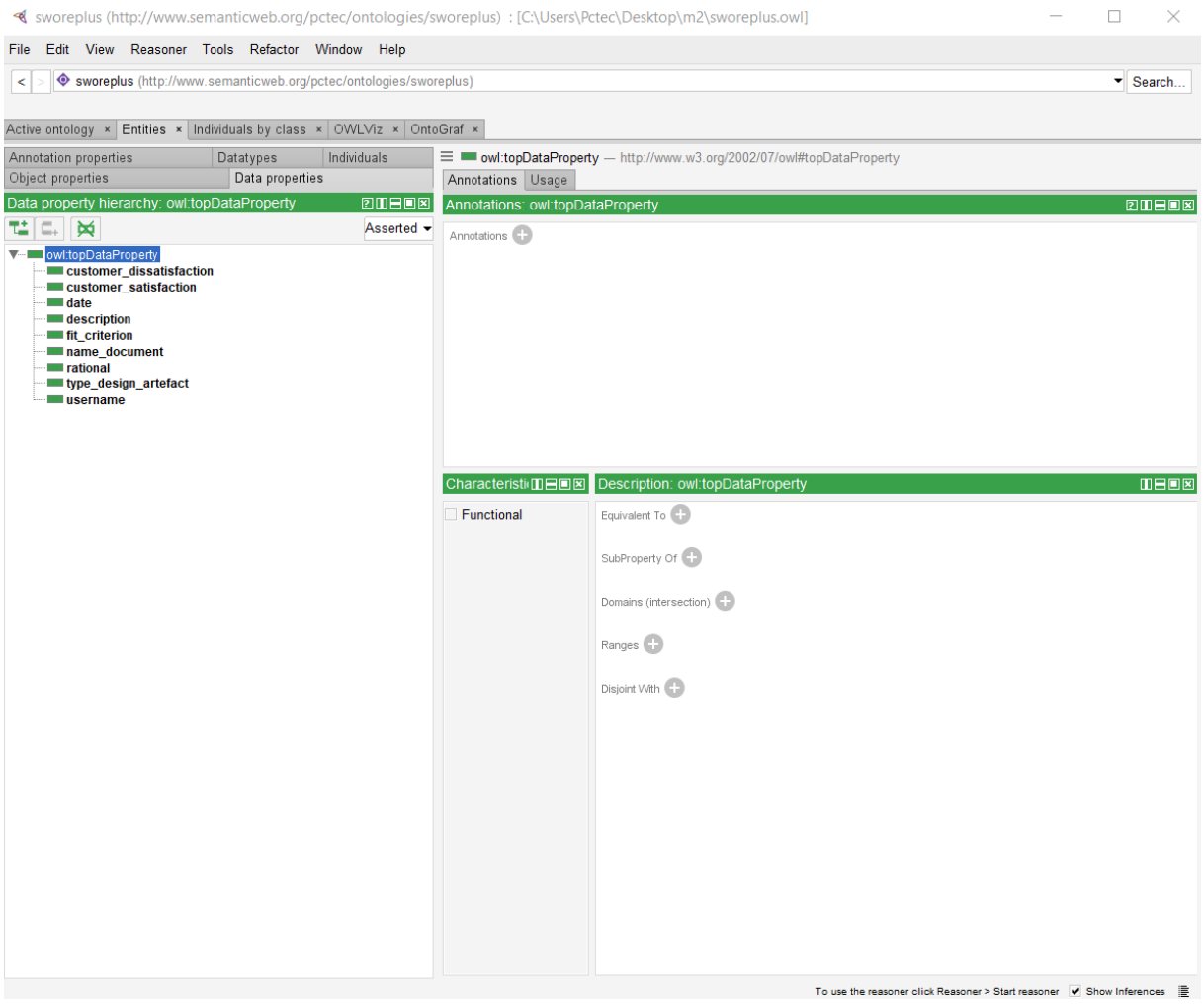
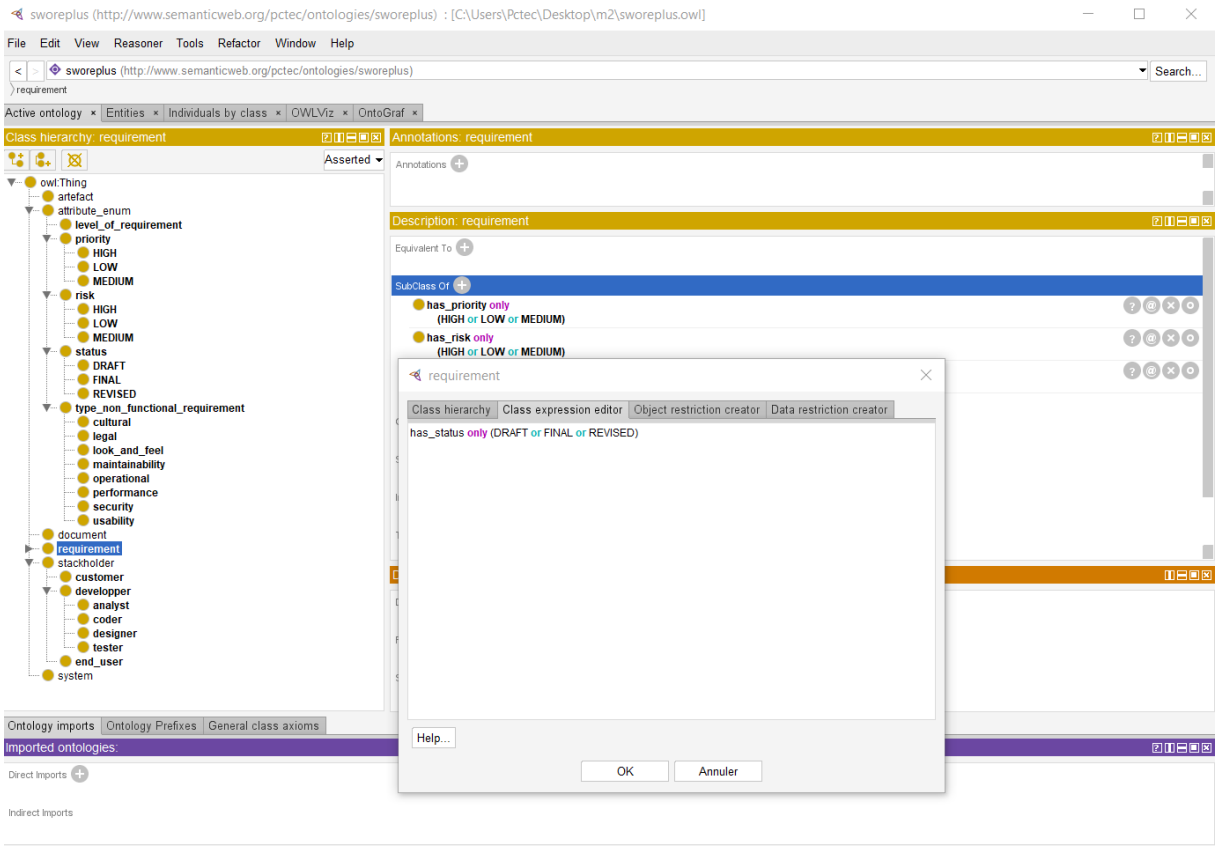


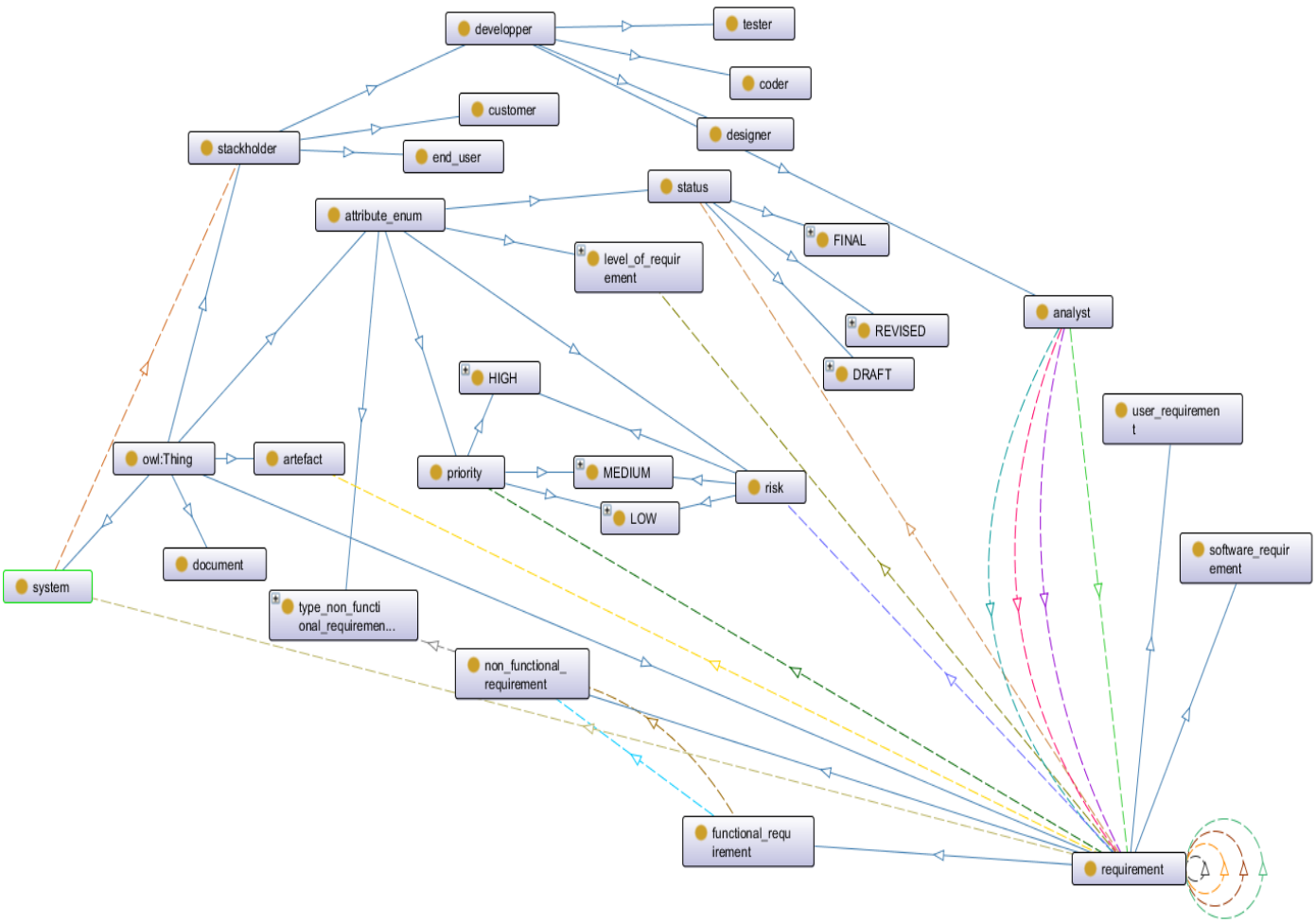
Figure VII.6. *L'édition des propriétés de type de donnée*

Et dans cette étape, nous définissons les restrictions (figure VII.7)



FigureVII.7. *Édition des restrictions*

A la fin nous pouvons visualiser notre ontologie en utilisons le plugin OntoGraf intégré automatiquement dans Protégé. (Figure VII.8).



FigureVII.8. *visualisation de l'ontologie*

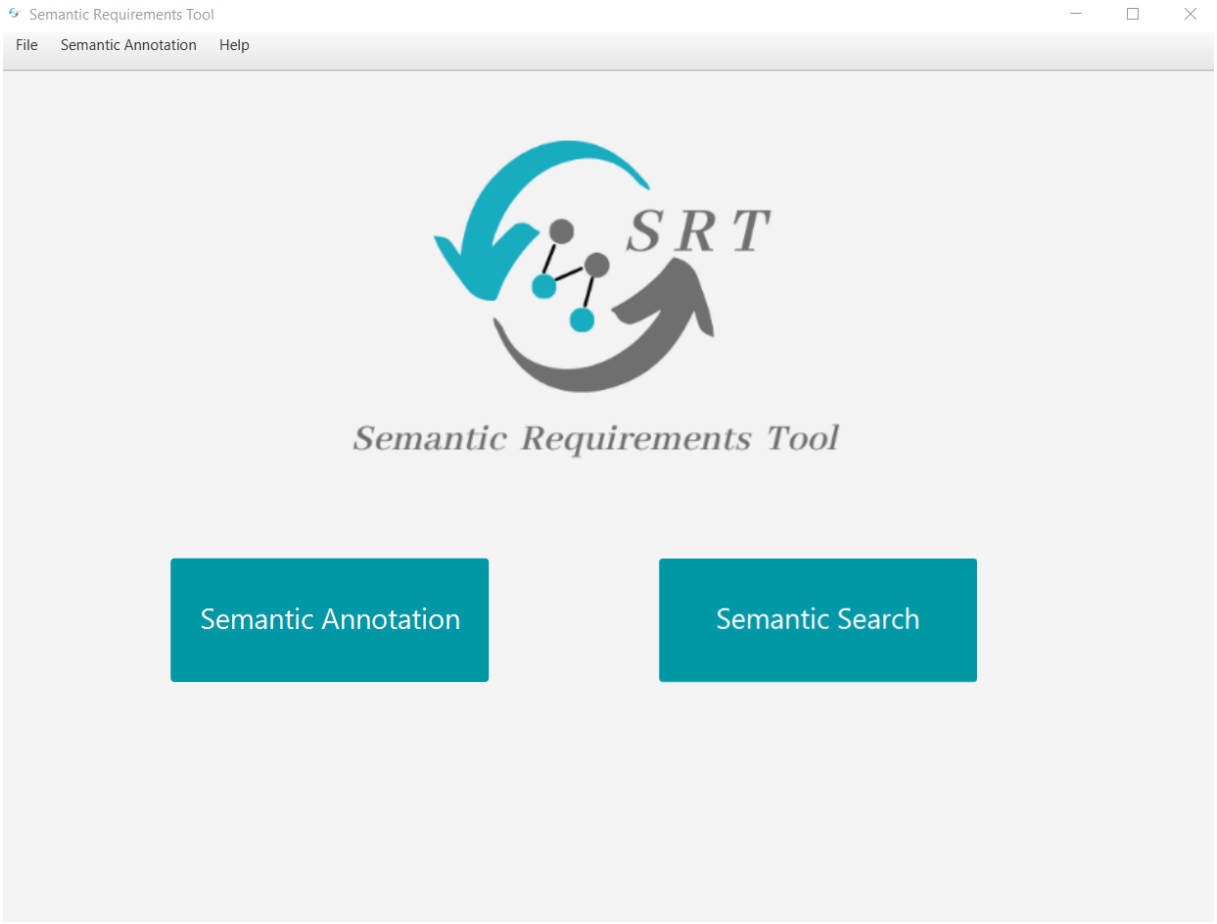
Après la construction du dataset, en choisissant une norme pour représenter le SRS afin de représenter les exigences et la construction de l'ontologie, nous pouvons dire que nous avons les éléments nécessaires pour faire l'annotation sémantique. Dans ce cas n'importe quel document qui suit le Template Volere peut être annoté sémantiquement en utilisant notre ontologie SWOREPLUS dans notre système SRT.

Dans la partie suivante nous montrons comment faire l'annotation sémantique d'un document d'exigence en utilisant l'ontologie SWOREPLUS avec GATE intégrer dans SRT.

VII.3.2 L'interface de Semantic Requirement Toool

Pour faire l'annotation et la recherche sémantique, nous proposons le prototype illustré dans les figures suivantes.

Après le lancement de SRT. La fenêtre d'accueil suivante s'affiche et qui permet de choisir la tâche à réaliser. (Figure VII.9)



FigureVII.9. *Page d'accueil*

Si l'utilisateur a choisi « Semantic Annotation » il va être redirigé vers l'annotateur sémantique GATE pour faire l'annotation sémantique. Par contre s'il a choisi « Semantic Search » il va être redirigé vers une autre fenêtre pour faire la recherche sémantique.

Nous commençons d'abord par l'annotation sémantique.

VII.3.2.a L'annotation sémantique

La fenêtre illustrée par la figure VII.10 apparaitra

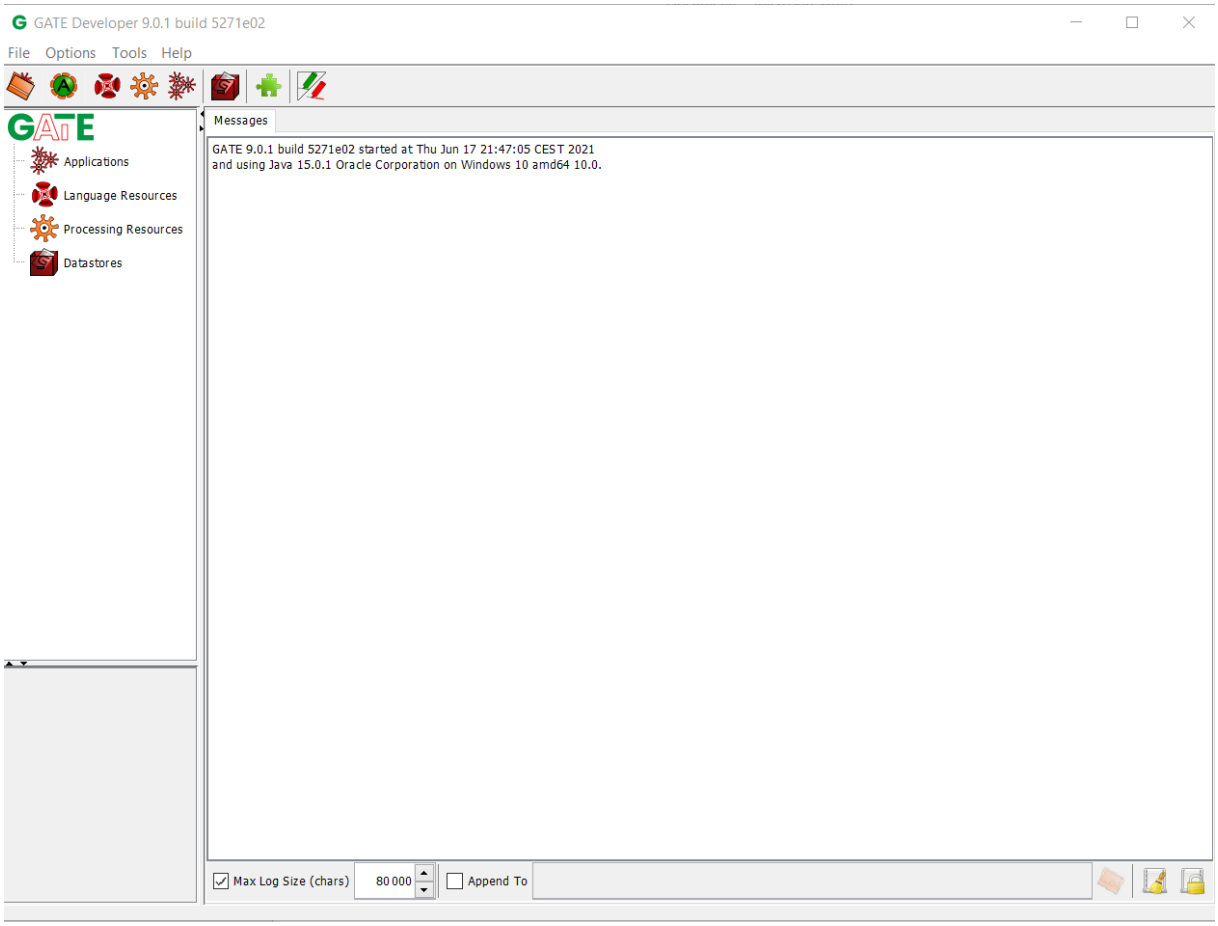


Figure VII.10. *La page d'accueil de GATE*

La deuxième étape consiste à charger les plugins dans GATE. Le plugin Ontology_Tool est disponible dans la liste des plugins disponible dans GATE, et le plugin Ontology contient des bibliothèques qui ne sont pas disponibles dans « Central Maven Repository ».Alors ; nous devons le télécharger séparément. Pour cela nous avons installé la dernière version à partir de Github et nous l'avons dézipé dans le répertoire « gateplugin-Ontology-version ».

La figure VII.11 et VII.12 montrent les étapes de chargement des plugins

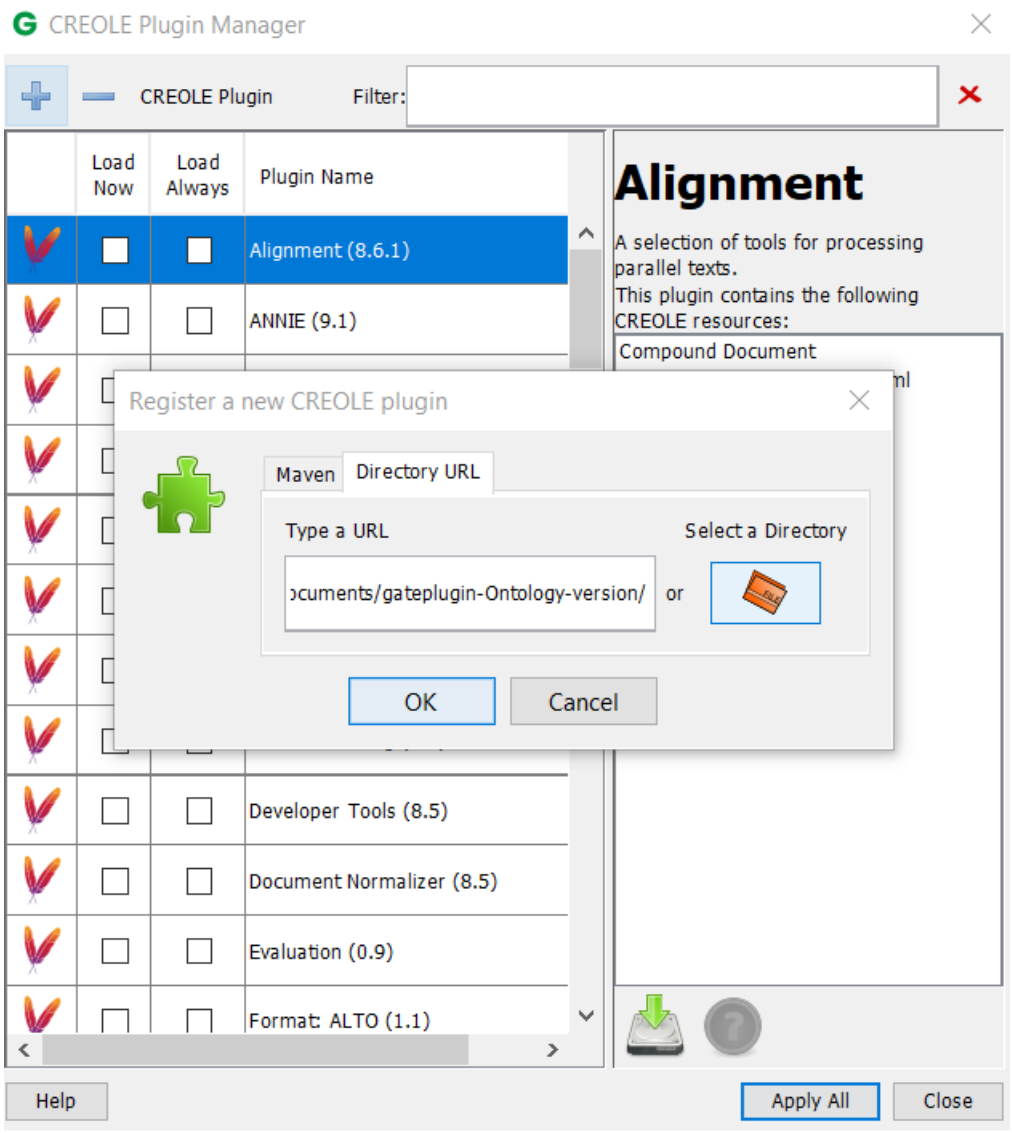


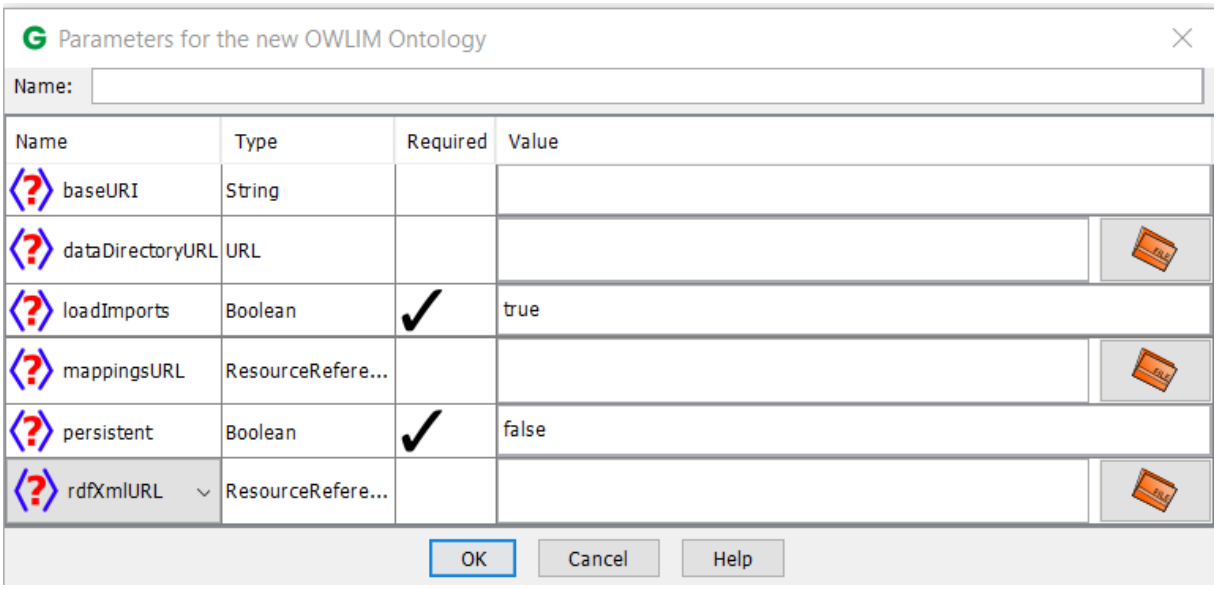
Figure VII.11. *Chargement de plugin Ontology*

FigureVII.12. *La sélection des plugins Ontology et Ontology Tools pour les chargés*

Après le chargement des plugins, trois ressources linguistiques d'ontologie seront ajoutées aux ressources linguistiques

- OWLIMOntology : c'est une ressource linguistique standard qui permet de créer une nouvelle ontologie et éventuellement d'y charger des données d'ontologie
- ConnectSesameOntology permet l'utilisation d'ontologies qui sont déjà stocker dans un référentiel Sesame2 qui sont stocké soit dans un répertoire soit accessible depuis un serveur.
- CreateSesameOntology permet de créer une nouvelle ontologie vide en spécifiant la configuration du référentiel pour la création du référentiel Sesame.

Dans notre travail nous utilisons la ressource linguistique OWLIMOntology car nous n'avons qu'une simple ontologie à utiliser. Pour créer cette ressource, la boite de dialogue illustrée dans la figure VII.13 apparaitra avec un ensemble de paramètres à remplir ou à modifier :



***FigureVII.13**. Les paramètres pour créer un nouvel OWLIM Ontology*

Name : pour spécifier le nom de ressource

BaseURI : URI à utiliser pour résoudre les références URI relatives dans l'ontologie pendant le chargement.

dataDirectoryName : nom de répertoire qui sauvegarde le magasin d'ontologie.

LoadImports : si la valeur est False, toutes les spécifications d'importation d'ontologie trouvées dans l'ontologie chargé sont ignorées. Ce paramètre est ignoré si aucune ontologie n'est chargée lors de la création de la ressource.

mappingURL : URL d'un fichier texte contenant les spécifications des mappages d'importation.

persistent : si false, le répertoire créé à l'intérieur du répertoire de données est supprimé lorsque la ressource de langue est fermée, sinon, ce répertoire est conservé.

rdfXmlUrl : URL spécifiant l'emplacement d'une ontologie au format de sérialisation RDF/XML,N3, turtle, ou N-Triples.

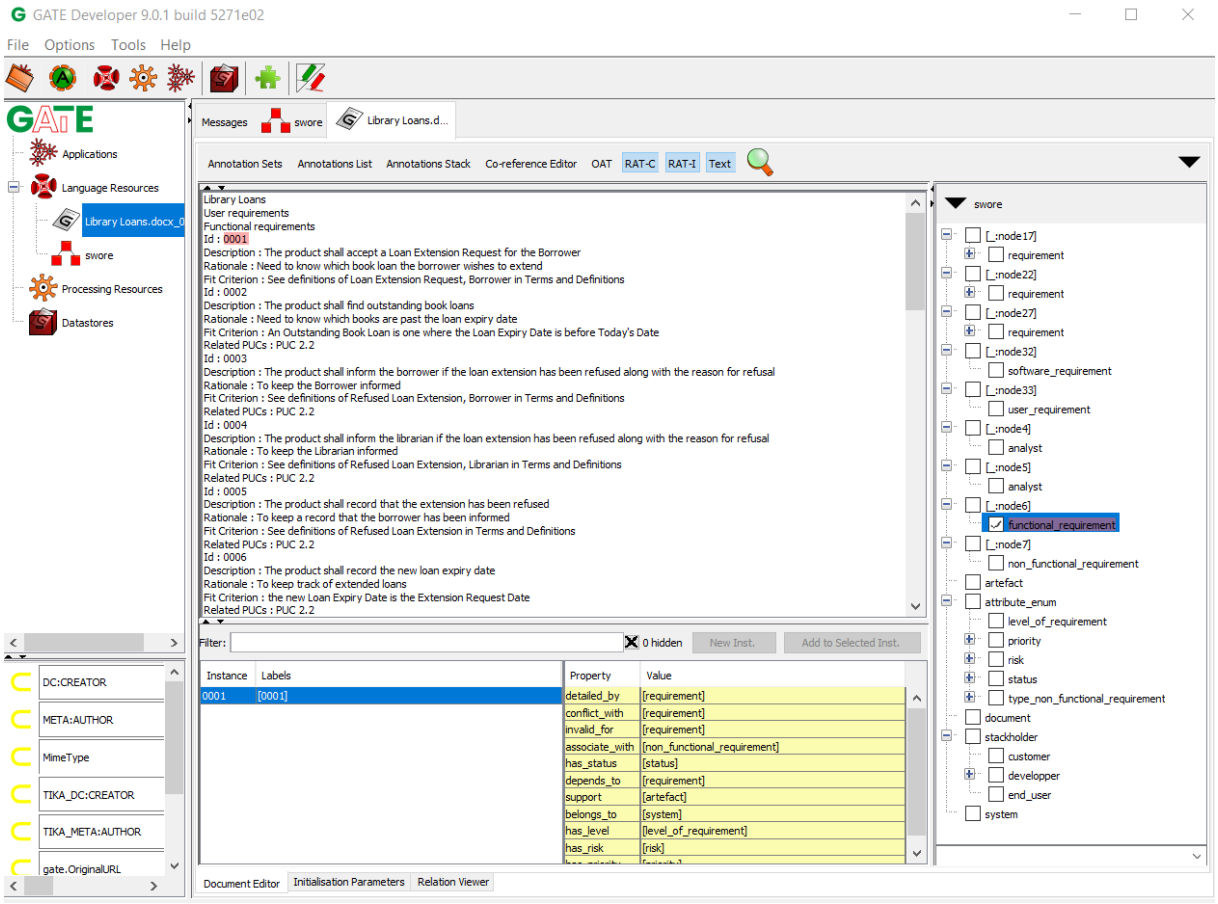
Maintenant, notre ontologie est déjà implémentée en utilisant Protégé. Pour cela, il nous reste que charger l'ontologie en sélectionnant l'option « charger » dans le menu contextuel de la ressource linguistique.

D'autre part, pour annoter sémantiquement un document, nous devons charger ce document. Pour faire ceci, il suffit de charger ce document dans la ressource linguistique, et le document s'affichera dans l'éditeur de document.

L'outil d'annotation RAT-C est un plugin disponible à partir de l'ensemble de plugins Ontology Tools. Il permet d'annoter manuellement un texte sélectionné par rapport à une ontologie.

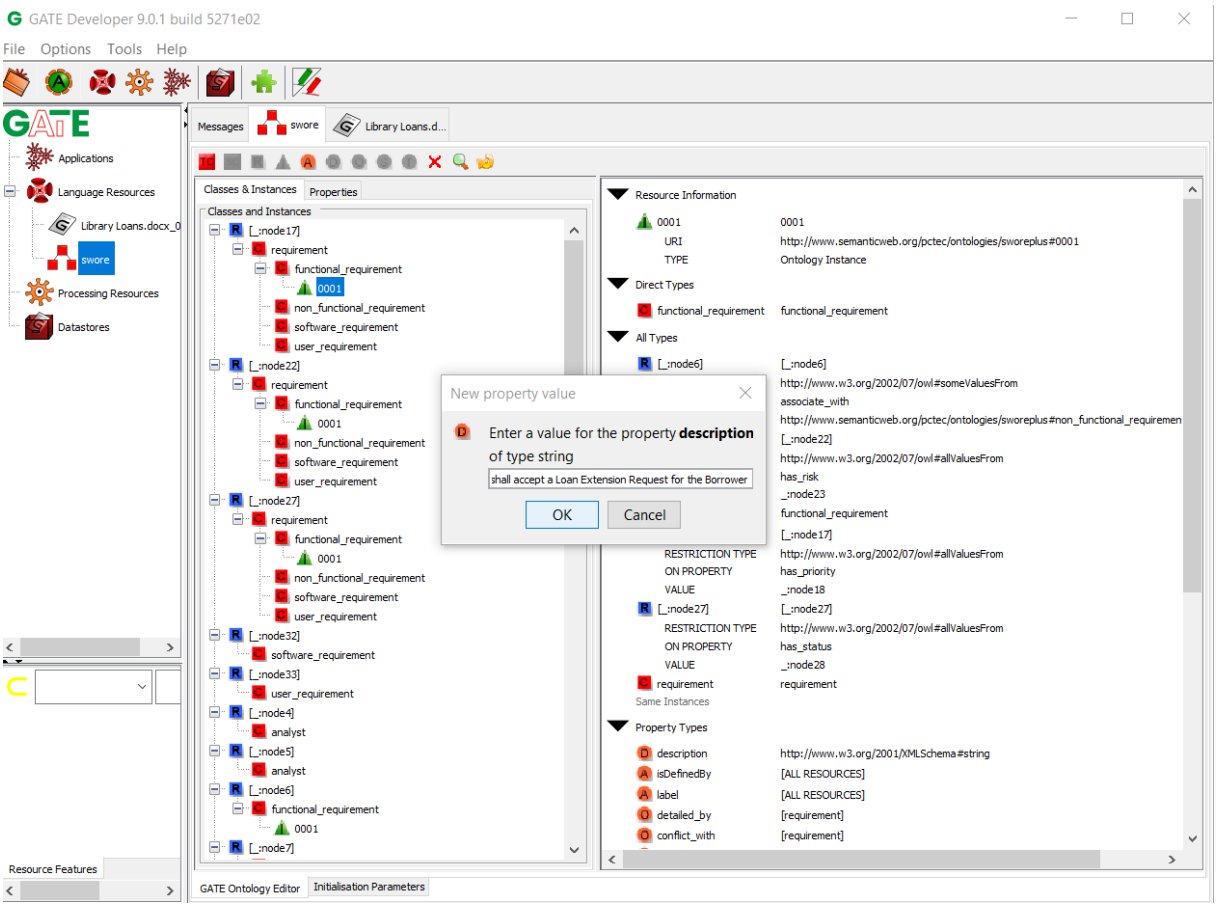
L'ontologie requise doit être sélectionnée à partir d'une liste déroulante d'ontologies disponible.

Pour annoter une classe ; il suffit de sélectionner le texte et cocher la classe à annoter et la liste des associations reliées à cette classe serons afficher en bas de page pour compléter l'annotation. La figure VII.14 illustre l'annotation d'une exigence fonctionnelle dans un document par l'ontologie SWOREPLUS.



FigureVII.14. *Annotation sémantique étape1*

Pour compléter l'annotation nous pouvons aussi définir les propriétés de type de données. Pour se faire, nous devons sélectionner le texte et allons vers l'éditeur de l'ontologie et nous choisissons l'instance et les propriétés correspondantes. (Figure VII.15)



FigureVII.15. *Annotation sémantique étape2*

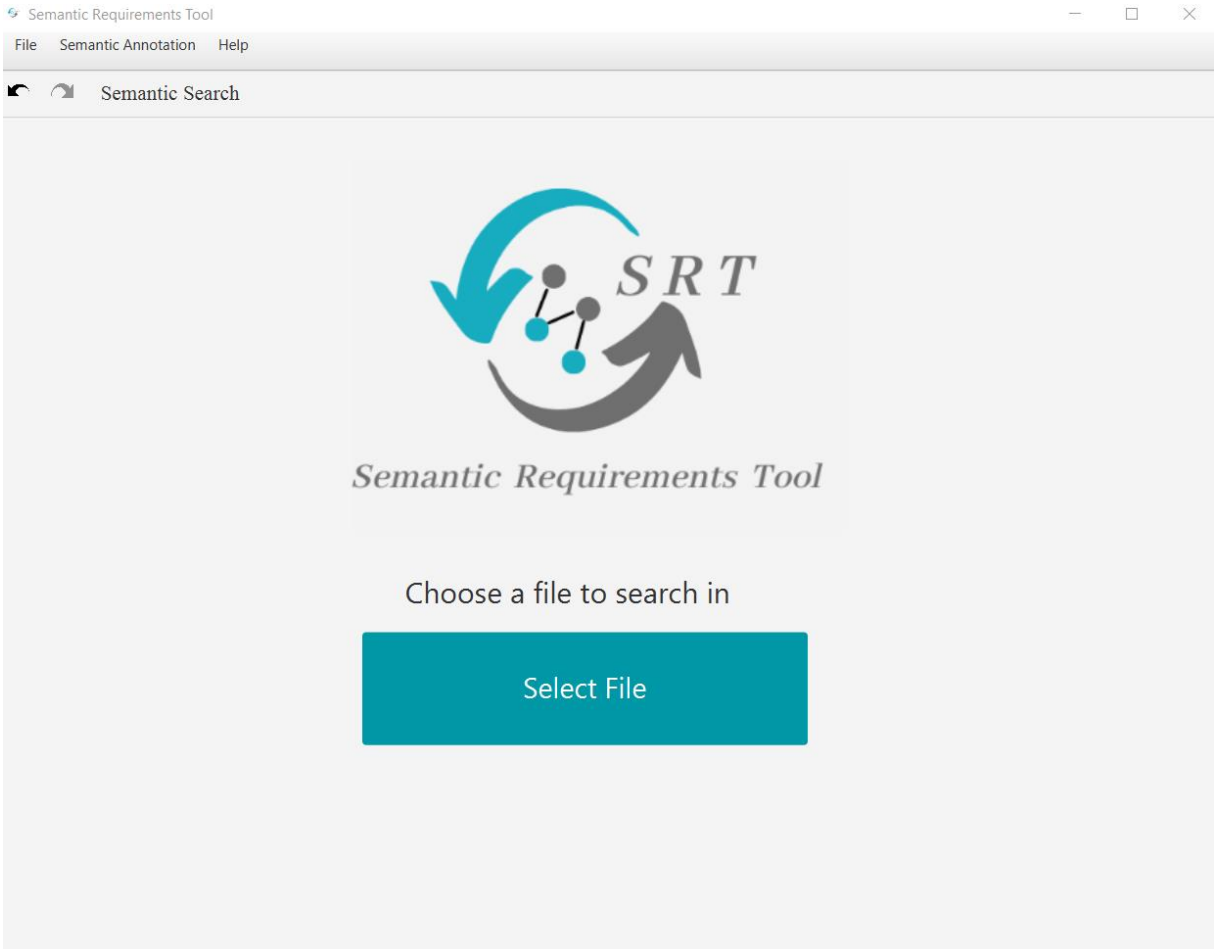
La sortie de cette annotation est un fichier qui contient des triplets RDF correspondant (Nous obtenons ce fichier en enregistrant l'ontologie annotée avec l'extension .rdf).

Nb. Pour la représentation et l'explication du fonctionnement de GATE, nous avons utilisé la référence GATE[4]

VII.3.2.b La recherche sémantique

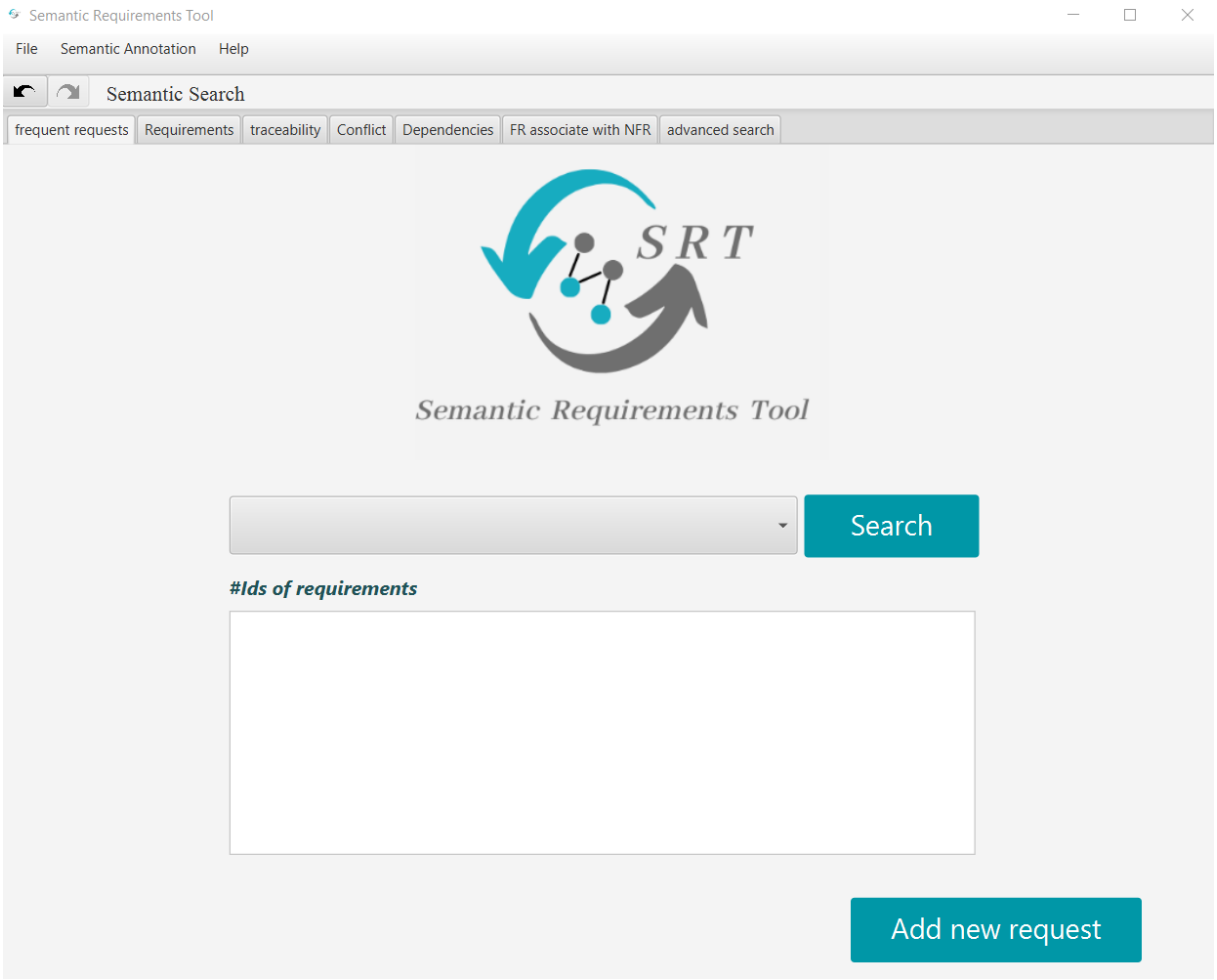
Après l'annotation sémantique, l'utilisateur peut faire la recherche sémantique sur un document annoté Il va être redirigé vers la scène illustré dans la figure VII.16 pour choisir le fichier annoté

[4] GATE, Chapter14 Working with Ontologies : https://gate.ac.uk/releases/gate-5.2.1-build3581-ALL/doc/tao/splitch14.html



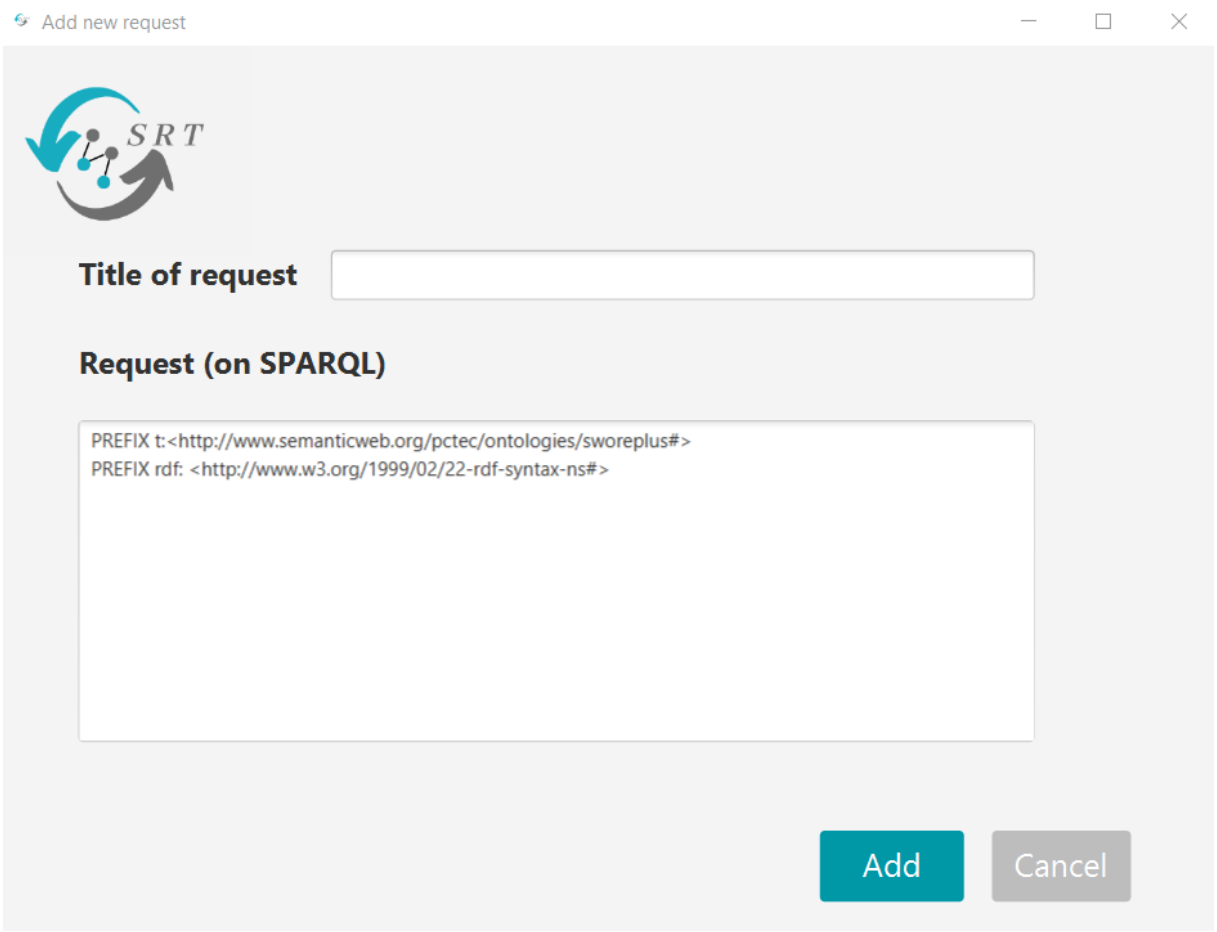
FigureVII.16. *Chargement du fichier*

Après la sélection du fichier, il peut faire sa recherche (figure VII.17).



FigureVII.17. *Les requêtes fréquentes*

Nous pouvons aussi ajouter une nouvelle requête à la liste des requêtes fréquentes en appuyant sur le bouton « add new request » (figure VII.18).



FigureVII.18. *Ajouter une nouvelle requête fréquente*

Les requêtes ajoutées par l'utilisateur, peuvent être modifié ou supprimé. Pour ce faire, il suffit de sélectionner la requête. Les fenêtres illustrés dans les figures VII.19 et VII.20 respectivement représentent ces taches.

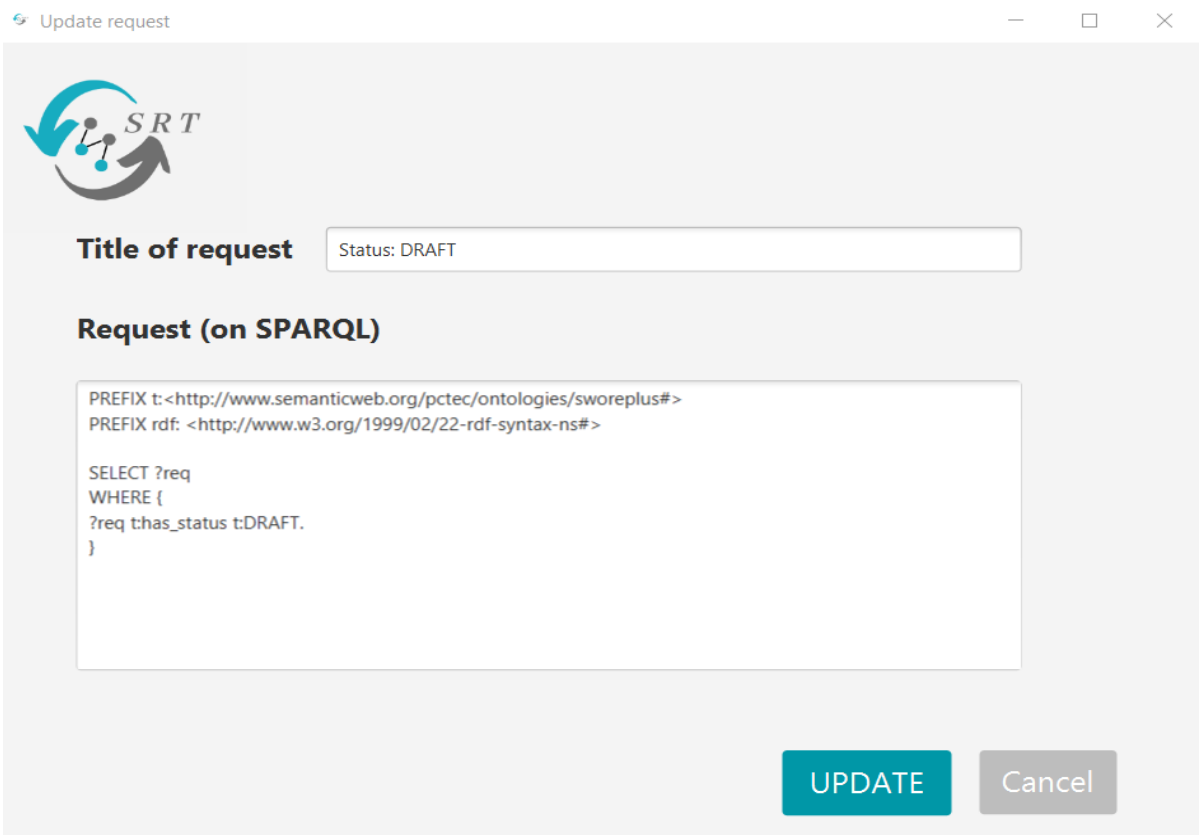


***Figure VII.19**. Modifier une requête fréquente*

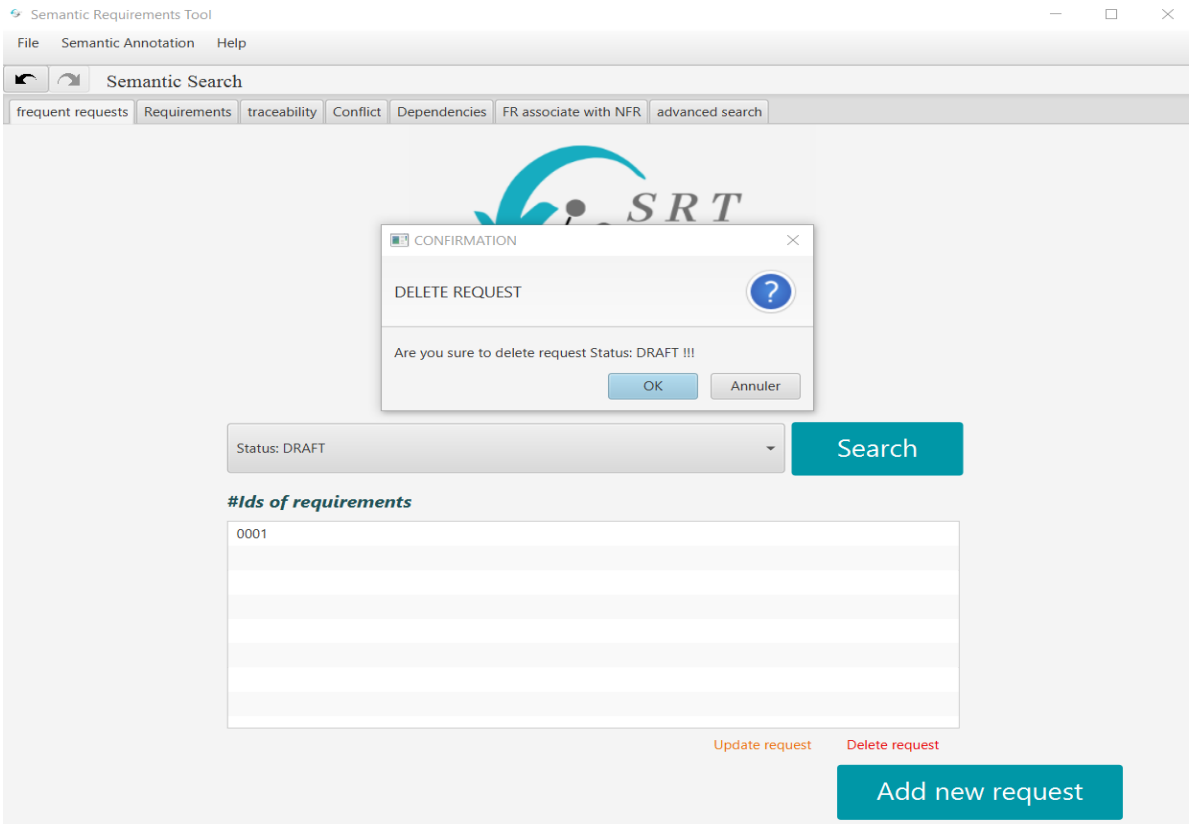


***Figure VII.20**. Supprimer une requête fréquente*

Le deuxième onglet intitulé « Requirement » permet de visualiser la liste des exigences annoté comme il est illustré dans la figure VII.21

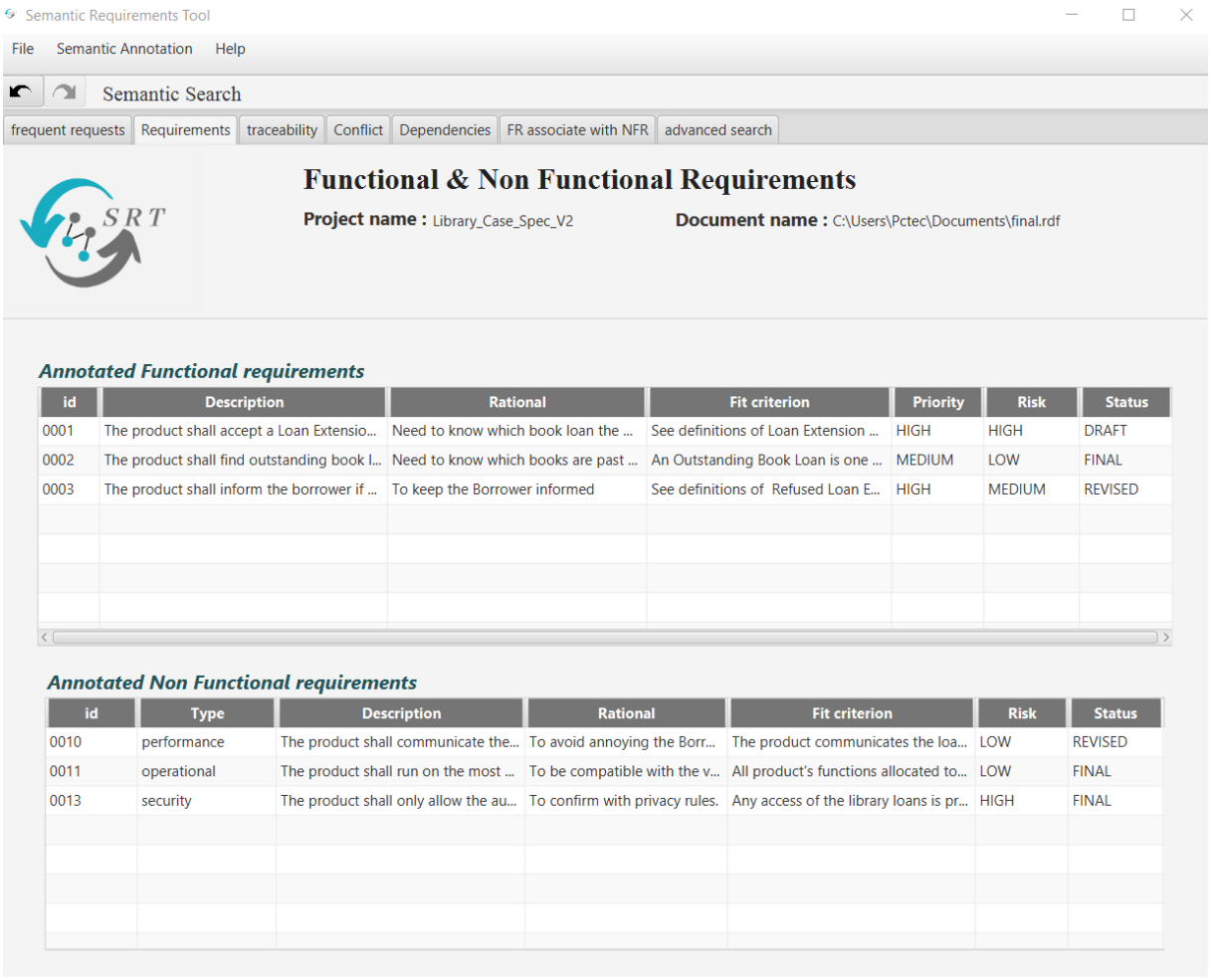


Figure VII.21. *L'onglet "Requirements"*

L'onglet « traceability » permet de visualiser la traçabilité d'une exigence, en choisissant le type d'exigence et son id. la figure VII.22 illustre cette fonctionnalité.

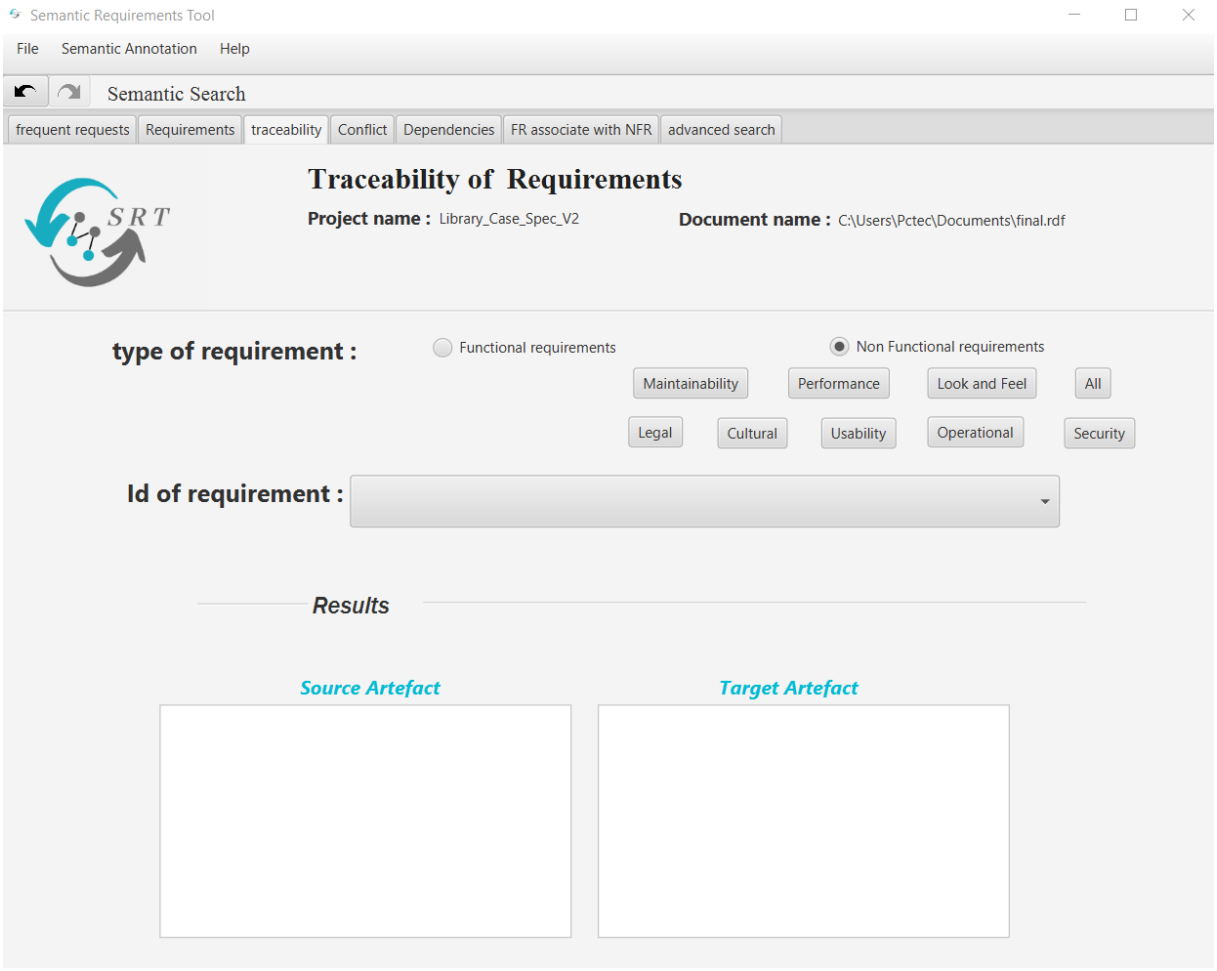


Figure VII.22. *L'onglet "Traceability"*

Le conflit et la dépendance entre les exigences sont des notions importantes dans l'ingénierie des exigences, pour diminuer la complexité et assurer la clarté entre les exigences. Nous présentons les exigences en conflit dans un onglet et les exigences dépendantes dans un autre onglet comme il est illustré dans les figures VII.23 et VII.24 respectivement.

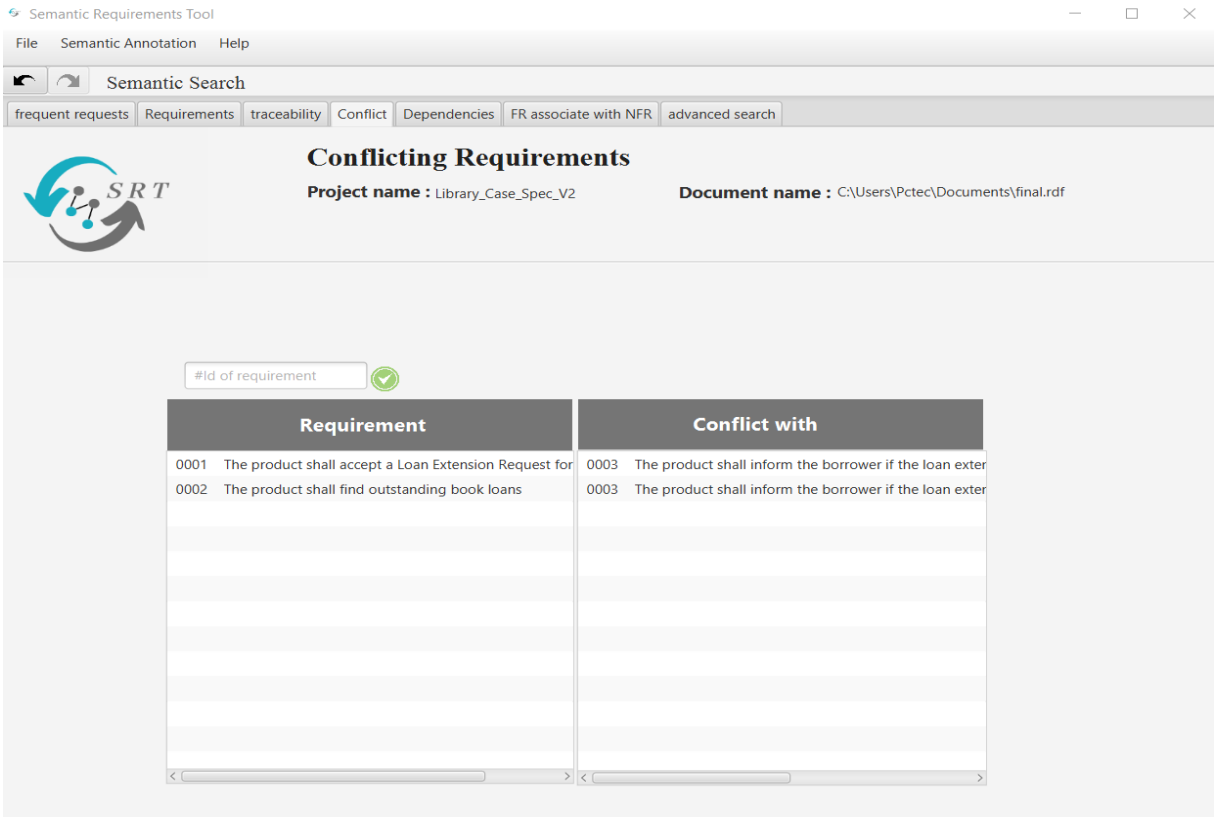


Figure VII.23. *L'onglet "Conflict"*

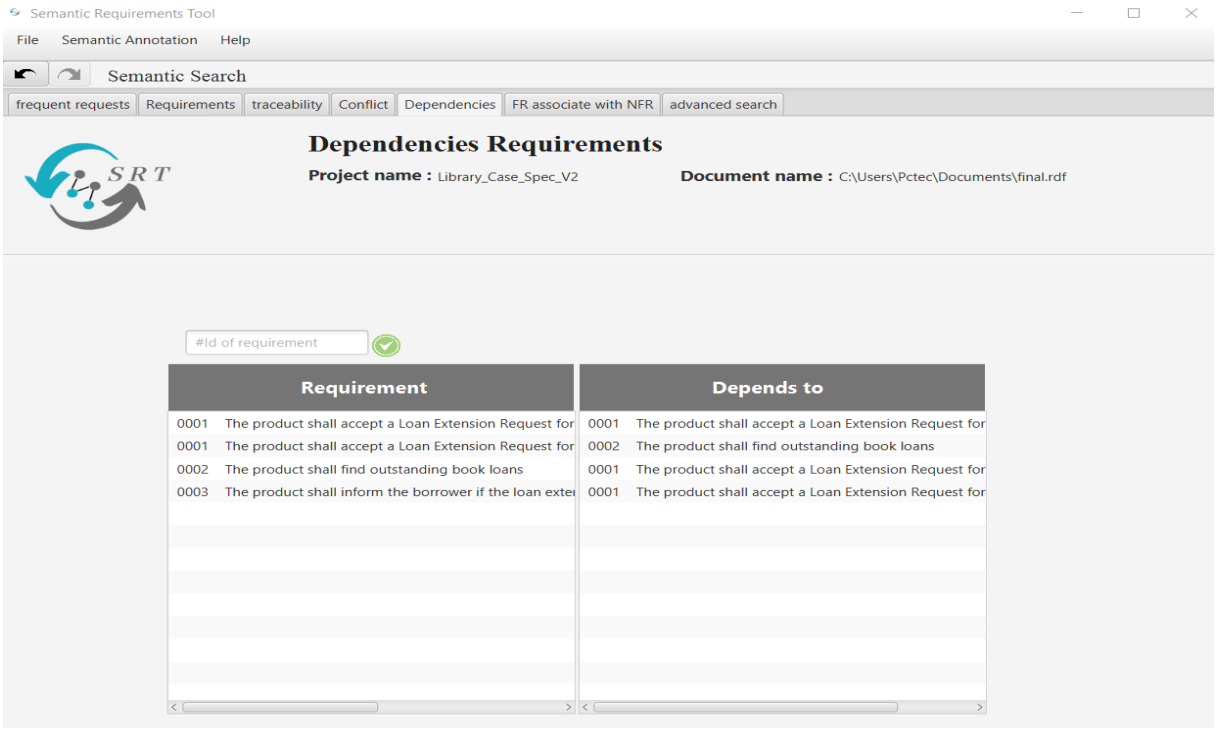


Figure VII.24. *L'onglet "Dependencies"*

Dans l'onglet « FR assosiate with NFR » nous pouvons visualiser les exigences fonctionnelles avec leurs exigences non fonctionnelles associées, comme il est illustré par la capture d'écran illustrée dans la figure VII.25.

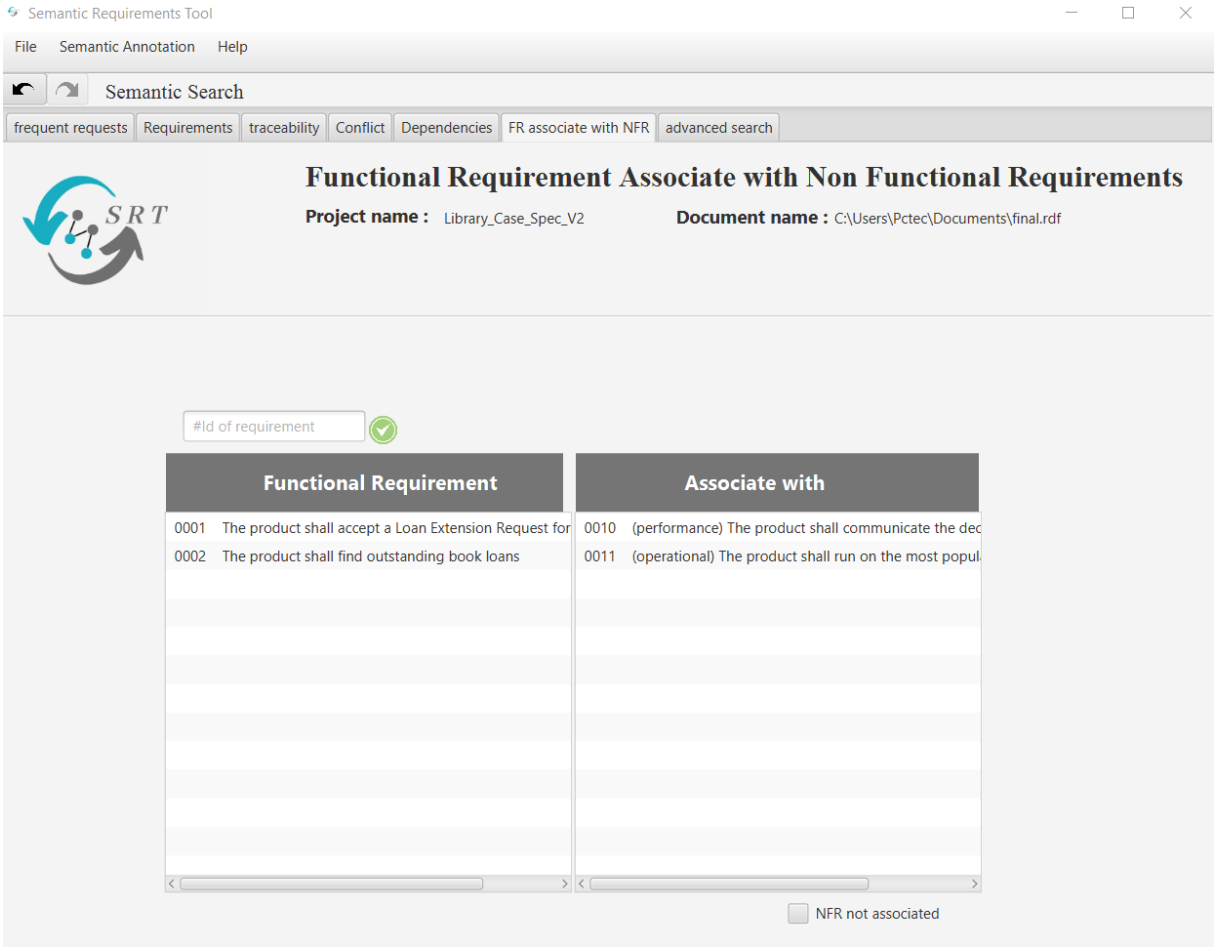


Figure VII.25. *L'onglet "FR associate with NFR"*

En activant le check box "NRF not associated" nous visualisons les exigences non fonctionnelle du système (ie les exigences associé au syteme tout entier et non pas celles qui sont associés à une exigence particuliere) (Figure VII.26).

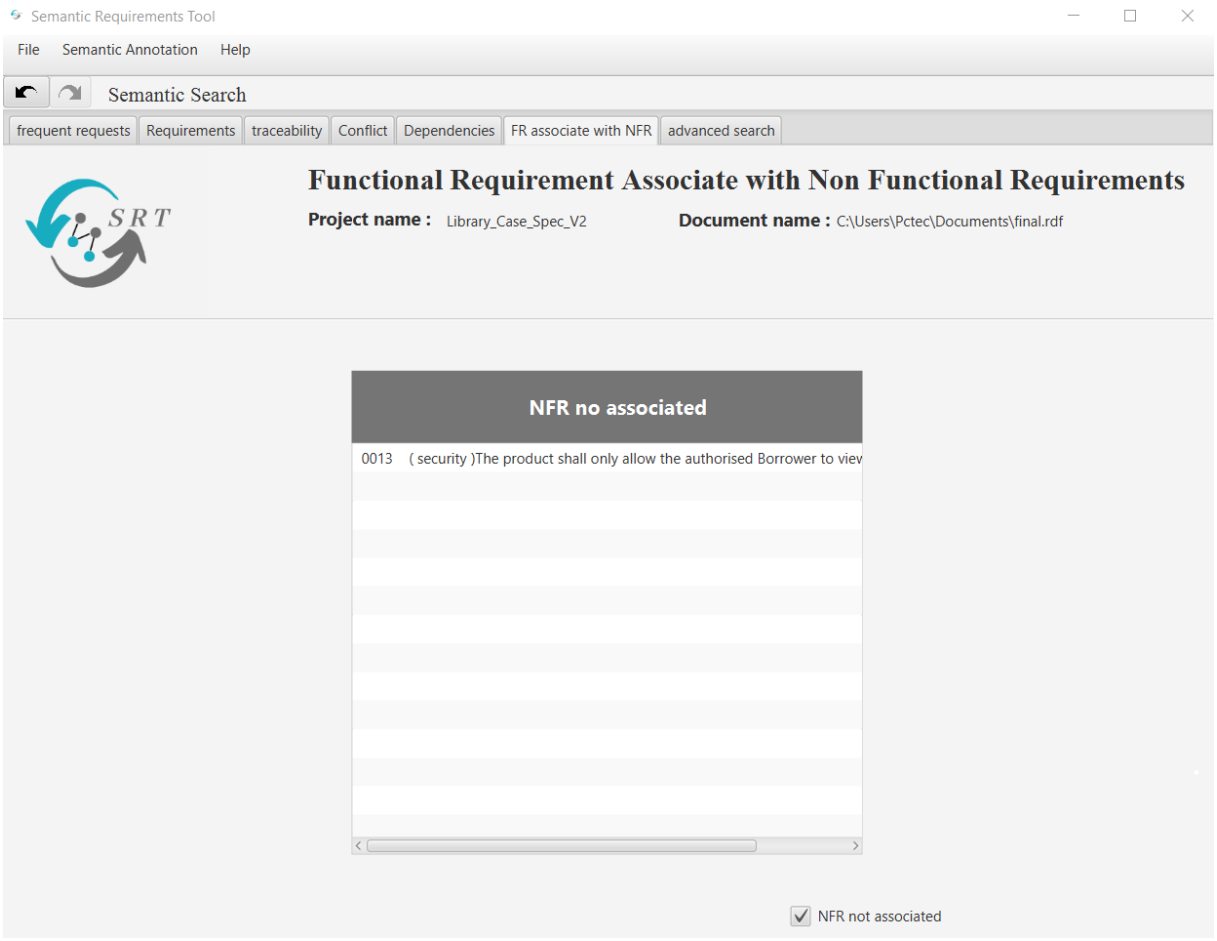


Figure VII.26. *NFR no associated*

Le dernier onglet permet de faire une recherche avancée en choisissant quelques critères sur le type d'exigence (fonctionnelle ou non fonctionnelle) (figure VII.27)

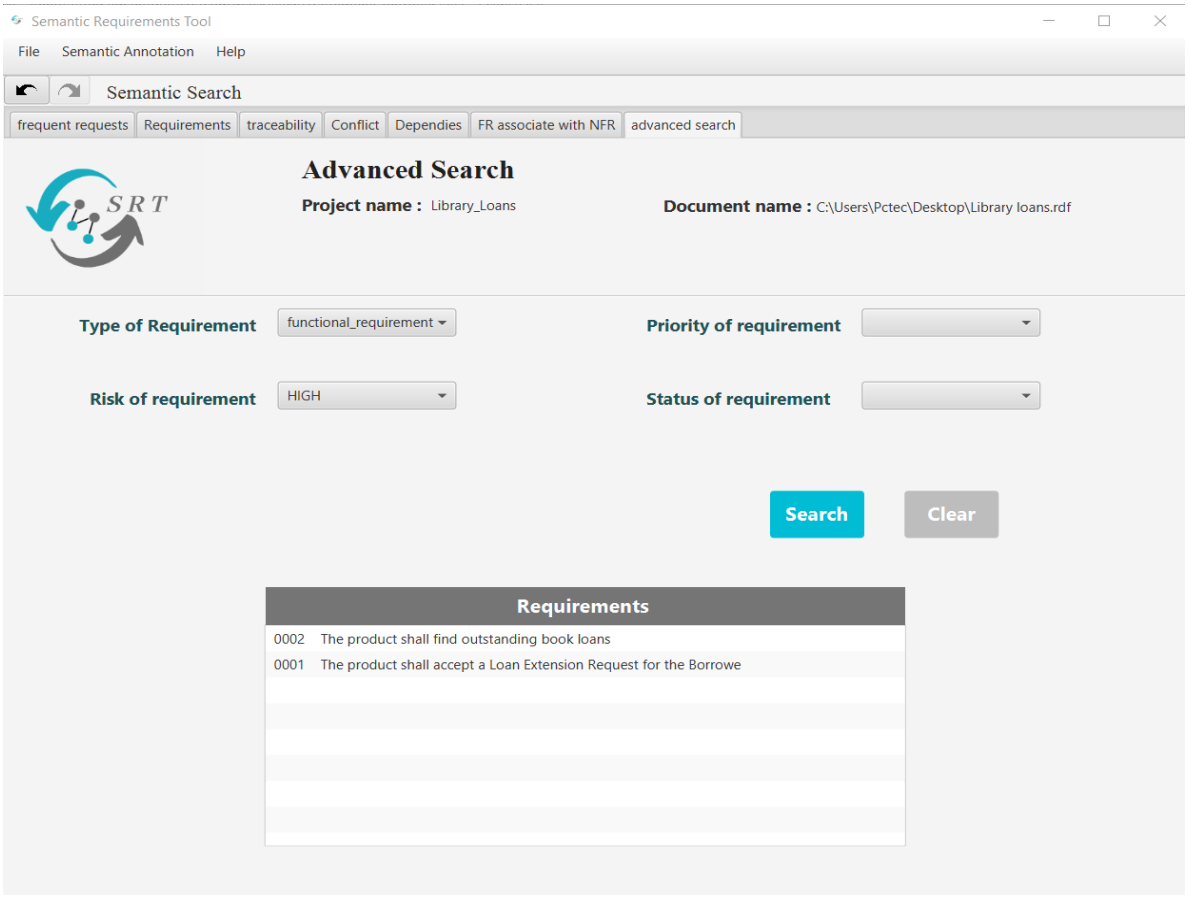


Figure VII.27. *L'onglet "Advanced Search"*

VII.4 Conclusion

Dans ce chapitre, nous avons présenté le prototype proposé. Ce prototype permet de faire l'annotation sémantique d'un SRS qui suit le Template Volere en utilisant une ontologie d'ingénierie d'exigence SWOREPLUS. De plus il permet de faire la recherche sémantique en se basant sur cette annotation.

Conclusion générale et perspectives

Dans ce travail une approche basée sur l'annotation sémantique et la recherche sémantique a été proposé Et ceci en vue de résoudre la complexité de gestion des exigences et automatiser son processus. Il s'agit d'une approche simple et complète dans le sens où elle couvre les points de notre problématique.

Notre approche se caractérise par la modélisation du contenu sémantique des documents à base d'une ontologie d'exigence proposée et commune entre les différents documents d'exigences. L'architecture de l'ontologie construite appelé SWOREPLUS est inspirée de l'ontologie SWORE. Nous l'avons adapté selon nos besoins et selon les composants voulus. Cette approche se caractérise aussi par la modélisation du contenu des documents d'exigences à base d'une spécification des exigences logicielles, en utilisant la norme Volere adapté selon nos besoins.

Les requêtes proposées dans la recherche sémantique sont inspirées des questions les plus utilisées, les plus fréquentes et les plus posées par un ingénieur d'exigence.

L'annotation sémantique proposée est une intégration directe d'un autre outil « GATE » qui offre un module qui permet de faire l'annotation sémantique. Cette solution est proposée juste pour illustrer le prototype de notre application. Nous savons que cette solution n'est pas la meilleure. Cependant à cause de différents contraintes nous ne pouvions pas appliquer la solution idéale qui consiste à intégrer un API qui nous aide à faire l'annotation sémantique par notre propre système. Nous espérons dans un proche avenir à proposer une meilleure solution et une meilleure interface utilisateur pour notre SRT.

Ainsi, nous comptons améliorer la qualité de notre système en proposant un autre module qui permet d'annoter sémantiquement et non pas juste les textes libres, mais aussi les SRS représentés en XML

Bibliographie

Atilf , 1992. ‘‘ Le trésor de la langue francaise’’. Paris, France,

Baader, F., Calvanese, D., McGuinness, D., Patel-Schneider, P., & Nardi, D ,2003. The description logic handbook: Theory, implementation and applications. Cambridge university press

Baldonado. M, Cousins. S, Gwizdka. J. et Paepcke. A. 2000 ‘‘ Notable: At the Intersection of Annotations and Handheld Technologies’’. Bristol, HUC, pp. 100-113.

Belabed Amine, 2019 – cours « Le langage owl (Ontology Web Langage) », université Abou Bakr Belkaid –Tlemcen

Bernard ESPINASSE, 2019 – cours « Introduction à SPARQL » - Aix-Marseille Université disponible sur<https://pageperso.lis-lab.fr/bernard.espinasse/Supports/WEBSEM/4-Cours-SPARQL-BE-4P.pdf>

Bizer , C., Heath , T., & Berners-Lee, T , 2011, Linked Data: The Story so Far. Disponible sur <https://eprints.soton.ac.uk/271285/1/bizer-heath-berners-lee-ijswis-linked-data.pdf >

B. Nuseibeh and S. Easterbrook, 2000, Requirements Engineering : A Roadmap. In Proceedings of International Conference on Software Engineering, ACM Press, Limerick, Ireland,

Dayra, 2017, KIM Platform - L'annotation sémantique disponible sur < https://slidewiki.org/deck/1876-2/plateforme-kim/slide/14344-1/14344-1:6/view?locale=fr>

Desmontils. E, Jacquin. C. 2002 , ‘‘Indexing a Web Site with a Termi nology Oriented Ontology’’. In The Emerging Semantic Web Amsterdam : IOS Press, pp. 181-197.

Fabien Gandon.2006, Ontologies informatiques. *Interstices*, Disponible sur < https://hal.inria.fr/inria-00080851/document> consulté le 15-04-2021

Gandon Fabien, Faron Zucker Catherine, Corby Oliver, 2015, Restriction de propriétés. Disponible sur <https://www.canal-u.tv/video/inria/restriction_de_proprietes.20366>

Gruber, T. R. (1993a). A translation approach to portable ontology specification. Knowledge Acquisition

HACINE GHERBI Ahcine, 2014, mémoire pour l'obtention du mémoire magister, Construction d'une ontologie pour le web sémantique, école doctorale d'information université Ferhat Abbas Sétif disponible sur < https://mmagister.univ-setif.dz/images/facultes/sciences /2014/Magister_HACINEGHERBI_Ahcine2014.pdf> consulté le 10/04/2021

John Doe, 2011, Recommended Practice for Software Requirement Specification (IEEE) disponible sur < https://ieeexplore.ieee.org/document/720574>

Jörg Dörr, 2017, Requirement enginnering, Requirement Specifications & standards

Klaus Pohl, 2016, Requirements Engineering Fundamentals, 2nd Edition: A Study Guide for the Certified Professional for Requirements Engineering Exam - Foundation Level - IREB compliant editeur : Rocky Nook, Inc.,184 pages

Kurt, C. 2019, Ontologies: Borrow, Build or Buy. Forbes. Disponible sur <https://www.forbes.com/sites/cognitiveworld/2019/05/07/ontologies-borrow-build-orbuy/#6b6c3e98fdec>

Marcello Vitali-Rosati, *Michael E. Sinatra*, 2014,PRATIQUES DE L'ÉDITION NUMÉRIQUE, Presses de l'Université de Montréal, MONTREAL,Nombre de pages : 224 p.

Pamela Zave, 1997. Classification of Research Efforts in Requirements Engineering. ACM Computing Surveys, Vol. 29, No. 4,

Pericles Loucopoulos and Vassilios Karakostas, 1995. System Requirements Engineering. McGraw-Hill, Inc., New York, NY, USA.

Pié.Y , 2004. '' Annotations et méta-données pour le web sémantique''. Dans les actes des journées scientifiques « web sémantique et SHS »

Ricardo de Almeida Falbo, Carlos Eduardo Correa Braga, 2014, Semantic documentation in requirements engineering

Salima Bourougaa-Tria (Auteur),2019, Ontologie et Web Sémantique,Munich, GRIN. Disponible sur < https://www.grin.com/document.com/476903> consulté le 12/04/2021

Sikos, L. F. (2016). A novel approach to multimedia ontology engineering for automated reasoning over audiovisual LOD datasets. Asian Conference on Intelligent Information and Database Systems (pp. 3-12). Springer, Berlin, Heidelberg.

Sonia Gueraich, mémoire Une approche basée annotation sémantique de documents pour la gestion d'une mémoire d'entreprise, Université Mentouri Constantine

Tim Bernard Lee, 1989, CERN, disponible sur <https://www.w3.org/History/1989 /proposal.html>

Volere requirements Specification Template, édition 2016, disponible sur <https://www.reqview.com/doc/volere-template> consulté le 29-05-2021>

Yannick Toussaint, Sylvain Tenier, 2017, Annotation sémantique par classification. Ingénierie des connaissances, France, pp.85-96.

Résumé

Dans l'ingénierie des exigences, les documents fournissent une ressource riche pour décrire ce que les parties prenantes savent sur le système. Le contenu de ces documents est conçu pour n'être compréhensible que par les humains. Dans ce travail, nous proposons d'utiliser l'annotation sémantique afin de rendre d'une part ces contenus compréhensibles par les machines. Et d'autre part pour rendre les tâches liées aux recherches, et récupération des données plus simple. Pour ce faire nous proposons une plateforme intitulée SRT « Semantic Requirements Tool » qui permet de faire l'annotation sémantique d'un document d'exigence qui suit le Template Volere en se basant sur une ontologie d'ingénierie d'exigence, appelée « SWOREPLUS ».Ainsi on offre aussi la recherche sémantique sur les documents déjà annotés.

Abstract

In requirements engineering, documents provide a rich resource for describing what stakeholders know about the system to be done. The content of these documents is designed to be only understandable by humans. In this work, we propose to use semantic annotation in order to make this content understandable by machines, and to make easier the tasks related to searching and retrieving data. To do this, we offer a plateform called SRT « Semantic Requirements Tool » which makes it possible to make the semantic annotation of requirements documents that follows the Template Volere based on a requirement engineering ontology, called « SWOREPLUS ».Thus the system allows also to do the semantic search on the already annotated documents.

ملخص

في هندسة المتطلبات ، توفر المستندات مصدرًا غنيًا لوصف ما يعرفه أصحاب المصلحة عن النظام. تم تصميم محتوى هذه الوثائق بحيث يكون مفهوماً من قبل البشر فقط. في هذا العمل ، نقترح استخدام التعليقات التوضيحية الدلالية من أجل جعل هذه المحتويات مفهومة من قبل الآلات من ناحية. ومن ناحية أخرى لتسهيل المهام المتعلقة بالبحث والفهرسة واسترجاع البيانات. للقيام بذلك ، نقدم نظامًا أساسيًا يسمى "أداة المتطلبات الدلالية" SRT والذي يسمح بالتعليق التوضيحي الدلالي لمستند المتطلبات الذي يتبع النموذج VOLERE بناءً على متطلبات الأنطولوجيا الهندسية ، تسمىSWOREPLUS وبالتالي فإننا نقدم أيضًا البحث الدلالي في المستندات المشروحة.

Mots clés: Ingénierie des exigences, Web sémantique, Annotation sémantique, Ontologie, Volere.

Keyswords: Requirements engineering, Semantic web, Semantic annotation, Ontology, Volere.

الكلمات المفتاحية: هندسة المتطلبات ، الويب الدلالي ، الشرح الدلالي ، علم الوجود ، VOLERE.