UNIVERSITE NATIONALE DU VIETNAM, HANOI
INSTITUT FRANCOPHONE INTERNATIONAL

NGUYỄN THỊ QUỲNH

INTÉGRATION ET VISUALISATION DE DONNÉES ISSUES DU PROJET IDEX IDENTITÉS COMPLEXES DE L'UNIVERSITÉ DE STRASBOURG

TÍCH HỢP VÀ HIỂN THỊ CÁC DỮ LIỆU TỪ DỰ ÁN IDEX IDENTITÉS COMPLEXES CỦA TRƯỜNG ĐẠI HỌC STRASBOURG

Spécialité: Réseaux et Système Communicants (RSC)

Code: Programme pilote

RESUME DU MEMOIRE DE FIN D'ETUDES DU MASTER INFORMATIQUE

HANOI – 2016

Travail réalisé à l'Institut Francophone International, Université Nationale du Vietnam, Hanoi

Sous la direction de:

Mme Cecilia ZANNI-MERK

Maître de conférences Hors Classe HDR en Informatique à l'INSA de Strasbourg Responsable\
Adjointe de l'équipe SDC «Sciences de Données et Connaissances du laboratoire ICUBE »

Mme Amira ESSAID-FARHAT

ATER en informatique à l'INSA de Strasbourg

Affiliée à l'ICUBE (équipe SDC), Université de Strasbourg

Affiliée au laboratoire LARODEC

Rapporteur 1:……

Rapporteur 2:……

Le mémoire est soutenu devant le jury à l'Institut Francophone International

le ……..…… 2015 à …… h…….

Le mémoire est accessible:

- au Centre d'Informations et de Bibliothèque, Université Nationale du Vietnam, Hanoi

- à l'Institut Francophone International, Université Nationale du Vietnam, Hanoi

HANOI - 2016

Chapitre 1: Présentation du stage

1.1 Problématique

L'Université de Strasbourg est née en 2009 suite à la fusion de trois universités strasbourgeoises: Louis Pasteur, Marc Bloch et Robert Schuman. Cette fusion a permis à l'Université de Strasbourg de devenir la deuxième plus grande université en France.

Foisonnant de formations, composantes, services, ressources humaines, matérielles et immatérielles, l'Université de Strasbourg est une institution complexe.

L'Université de Strasbourg regroupe une multitude d'entités, de disciplines et de connaissances où les savoirs se côtoient et se relient. Donc ses structures sont très compliquées. À cause de cette complexité, les gens rencontrent les difficultés lorsqu'ils veulent chercher à comprendre un certain composant, même pour ceux qui travaillent dans l'université. Pour résoudre ce problème, il faut avoir un outil qui peut fournir une vue globale pour tous les structures de l'université et une vue spécifique pour chaque composant.

1.2 Description du stage

Ce stage a été réalisé dans le cadre du projet IDEX (initiative d'excellence)[1] "Identités Complexes" de l'Université de Strasbourg. IDEX fait partie des investissements d'avenir dont le but est de créer en France des ensembles pluridisciplinaires d'enseignement supérieur et de recherche qui soient de rang mondial.

L'objectif du projet est de "Rendre lisible la particularité du savoir cultivé, conservé et transmis à l'Université de Strasbourg, à contribuer à l'interdisciplinarité par le graphisme, en rendant visible l'invisible ainsi qu'à élaborer une cartographie du savoir". Ce projet est de grande échelle. C'est pour cette raison, il a été prévu en plusieurs phases. La première phase du projet, intitulé "Recherche – action" a duré une année de Janvier 2015 jusqu'à Janvier 2016), a permis d'élaborer et de réaliser les 3 axes demandés par la présidence et le comité de pilotage et de répondre au Cahier des charges :

- La confection d'un « lexique-graphique »
- L'élaboration d'un « système visuel »
- Création d'une « typographie »

La deuxième phase (Janvier 2016 – Juin 2016) complémentaire d'application a permis de mettre en œuvre le système élaboré et de le développer sur l'université voire sur des institutions partenaires. Il conviendra également d'orienter la recherche sur la

[1] http://www.enseignementsup-recherche.gouv.fr/cid51351/initiatives-d-excellence.html#qu_est_ce_qu_un_campus_d_excellence

cartographie des savoirs, de développer des gabarits de thèses et de mémoires, d'élaborer un outil de génération de signatures institutionnelles (sous réserve des possibilités de développement par la Direction Informatique (DI) ou le service de communication de l'Université), de finaliser la recherche et de promouvoir le nouveau savoir constitué par la publication d'un livret pédagogique et l'organisation d'un colloque international également accompagné d'une publication. Mon stage est dans cette phase. Nous avons participé pour développer une cartographie des savoirs de l'université.

Chapitre 2: Phase d'analyse

2.1 Nouveau langage visuel de l'université de Strasbourg

Pour rendre lisible et intelligible l'université de Strasbourg, il faut construire un nouveau langage visuel qui s'accompagne d'un lexicographe, socle commun de référence destiné à offrir une lecture exhaustive et optimisée de l'ensemble des structures et des savoirs de l'université. Cette interface numérique permet de centraliser l'information tout en assurant une visualisation multiple des données via trois dispositifs complémentaires : le catalogue des structures, le traducteur d'acronymes et la cartographie des savoirs.

Le catalogue des structures a pour vocation de recenser en un même endroit toutes les structures de l'université. Sous forme de liste simple, cette interface permet de nommer et de visualiser les différentes structures de l'université ainsi que d'accéder en un clic aux informations recherchées.

Véritable outil de compréhension du langage universitaire, la fonction principale du traducteur d'acronymes et de logos est de traduire les codes, les abréviations, les sigles, les logos et les acronymes rencontrés à l'université, afin de rendre leurs contenus lisibles.

L'outil de cartographie de l'université permet de visualiser les composants de l'université de Strasbourg et ses relations entre eux. En plus, il visualisera les instances de chaque composant. C'est le système que nous allons développer et décrire en suite.

2.2 Spécification des besoins fonctionnels

Dans notre système, il y a plusieurs types d'acteurs : les personnes qui appartiennent à l'université (des professeurs, des étudiants, des administrateurs, etc.) ou encore toute personne externe qui aime avoir des informations sur l'université. A travers notre système, un acteur est capable de voir toutes les structures de l'Université de Strasbourg ainsi que les différentes relations existantes entre ces structures. Le système permet à l'utilisateur d'avoir des détails pour chaque structure.

L'université est composée de plusieurs structures. Avec notre système, l'utilisateur est capable de visualiser les structures et d'avoir les détails d'une structure donnée.

2.3 Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation sont des diagrammes UML [2]. Ils donnent une vision globale du comportement fonctionnel d'un système logiciel. À travers les diagrammes de cas d'utilisation, nous pourrons savoir : les utilisateurs interagissent avec le système, les cas d'utilisation du système.

Dans notre cas, les acteurs qui interagissent avec le système ont principalement les professeurs, les étudiants, les personnes travaillant à l'université, les personnes externes etc...

Les cas d'utilisations :

- Voir les composants de l'université : L'université est composée de plusieurs composants. Chaque fois de visualisation les structures de l'université, notre système permet l'acteur de choisir un composant. Il visualise les structures sous la forme un arbre avec deux niveaux. Cette forme a une racine qui est en plus haute, des branches et des feuilles qui sont en bas selon son niveau. Chaque fois, nous choisissons un composant qui devient la racine de l'arbre que nous afficherons. D'abord, la racine est en plus haute avec niveau 0. Ensuite, les composants qui ont des relations avec la racine sont en bas de la racine avec niveau 1. En fin, les composants qui ont des relations avec les composants dans niveau 1 sont en plus bas avec niveau 2.
- Voir la liste des instances de chaque composant : Quand l'acteur choisira un composant de l'université, il pourra voir sa liste des instances.
- Voir les relations d'une instance : Quand l'acteur choisira une instance, notre système visualisera cette instance et ses relations entre elle avec les autres instances de l'université.

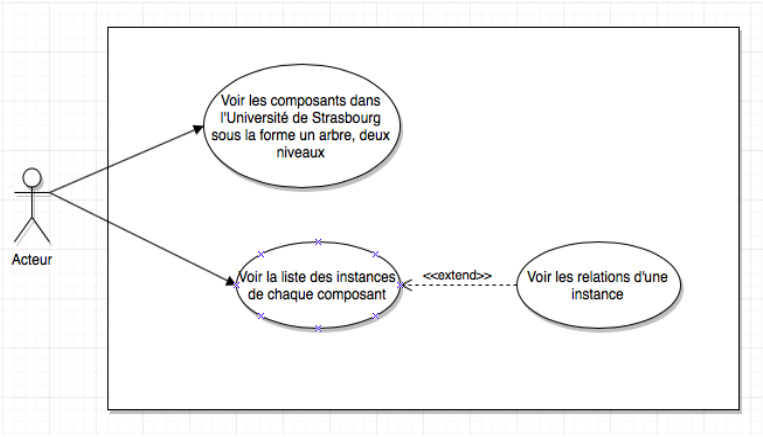

Figure 1: Le diagramme de cas d'utilisation

Chapitre 3: Phase de Conception

3.1 Notions du diagramme de classes

Le diagramme de classes représente une vue statique de la structure interne du système. Il présente les classes du système et les différentes relations entre celles-ci. Dans un diagramme de classes on trouve des classes et des associations entre ces classes.

- Une **classe** est une description d'un ensemble d'objets partageant les mêmes attributs et les mêmes méthodes.
- Une **association** est une relation entre au moins deux classes. Une association peut avoir un nom et une multiplicité.
- La **généralisation** met en relation une classe plus générale à une classe plus spécifique.
- **Agrégation** est une forme spéciale d'association. Elle représente une relation de type "ensemble / élément". Elle est par un "diamant" blanc. Cette association est utilisée quand la durée de vie des composants est indépendante de celles du conteneur.
- La **composition** est une association forte. Un élément ne peut faire partie que d'un seul ensemble. La destruction du conteneur entraîne la destruction des composants.

3.2 Diagramme de classes de l'Université de Strasbourg

L'université de Strasbourg est une institution complexe avec un nombre important d'associations, de campus, d'écoles doctorales, de formations, etc. A l'université de Strasbourg, on peut distinguer principalement deux grandes parties.

- La vie dans les campus qui décrit les différents campus et les différentes facilités offertes par l'université (bibliothèques, associations, musées,...)
- La formation et la recherche au sein de l'université. En effet, on s'intéresse aux:
  - Les différentes facultés, écoles, instituts ainsi qu'aux unités de recherche.
  - Les formations dispensées par l'Université de Strasbourg et les diplômes délivrés.
  - La formation doctorale, notamment les thèses soutenues et les informations associées à ces thèses (doctorant, directeur de thèse, comité de jury, etc…).
  - Les services centraux de l'Université de Strasbourg

Vu la complexité et le nombre assez important de classes et d'associations, nous présentons dans ce qui suit un diagramme de classes relatif à chacune des parties précédemment citée.

La vie dans les campus de l'université

L'Université de Strasbourg contient plusieurs associations qui ont pour local soit un bâtiment sous la direction de l'université ou en dehors de l'université. Chaque association appartient à un domaine bien particulier.

L'université est composée aussi de plusieurs campus et antennes composés de un ou plusieurs bâtiments. L'antenne est un site de l'université n'appartenant pas à un campus. Dans un bâtiment, on peut trouver des associations, des mussés, des bibliothèques, des départements, des équipes de recherche, etc.

L'université a plusieurs partenaires sur le plan national et international. En effet, l'université fait partie du:

- Contrat du site
- Cluster Alsace
- Eucor le campus européen regroupe.
- La Ligue des universités de recherche européennes (LERU)
- Région métropolitaine trinationale du Rhin supérieur.

L'université contient plusieurs des musées et collections qui sont divisées aux les musées universitaires et les collections. L'Université de Strasbourg possède de nombreuses collections scientifiques qui témoignent des activités d'enseignement, de recherche et de diffusion de la connaissance qu'elle a développées au cours de son histoire. Tous les musées et collections sont responsabilité de Jardin des Sciences et des composantes.

Les composantes de formation et les unités de recherche

L'université contient des collégiums. Le collégium est un organe de coordination entre la présidence et les composantes. Chaque collégium a un ou plusieurs domaines. Un collégium a des composantes et des unités de recherche.

- Une unité de recherche peut être rattachée à plusieurs écoles doctorales. Différents types d'unités de recherche existent: unité mixte de recherche, unité de service et de recherche, équipe d'accueil, unité propre de recherche et structure fédérative. Elles sont sous la responsabilité de l'université de Strasbourg, du Centre National de la Recherche Scientifique (CNRS) ou encore sous la responsabilité des deux (l'université et le CNRS). La gestion technique, financière et administrative des unités de recherche est assurée par des unités de service qui peuvent être des unités propres de service ou des unités mixtes de service. Une unité de recherche est composée d'équipes de recherche.
- Une composante peut être un institut universitaire de technologie (IUT), une faculté, une école ou encore un institut. Chacune de ces composantes a des

départements. Une composante est associée aux observatoires. Un observatoire peut être une faculté, une unité mixte de recherche ou encore une école.

La formation à l'Université de Strasbourg et les diplômes délivrés

La formation dispensée par l'Université de Strasbourg se caractérise par une très grande richesse disciplinaire**.** La formation répartie sur 37 IUT, facultés, écoles et instituts couvre plusieurs domaines. Il existe des formations dotées de mentions (Formation - Mention) et dans une mention, l'université peut offrir 0 ou plusieurs parcours (Formation - Mention - Parcours). Les formations peuvent déboucher sur un diplôme. On parle ainsi de formations diplômantes comme elles peuvent être des formations non diplômantes.

Les formations non diplômantes permettent aux étudiants de se former pour améliorer leurs compétences et de s'adapter aux nouvelles technologies. Ce sont généralement des formations basées sur des préparations aux concours comme Préparation aux concours de l'administration et formations juridiques, Préparation aux concours de l'enseignement et formations des enseignants, Première année santé, Concours programme grandes écoles.

Les diplômes délivrés lors des formations diplômantes sont divisés aux diplômes universitaires et diplômes nationaux. Le diplôme universitaire est un diplôme d'établissement (non national) dont l'université de Strasbourg est la seule initiatrice. Dans cette catégorie de diplôme, on trouve le diplôme d'université, le diplôme d'université du secteur santé ou diplôme d'établissement. Quant aux diplômes nationaux, ils sont délivrés le plus souvent au nom du ministère de l'Éducation nationale, de l'Enseignement supérieur et de la Recherche. Les formations de l'enseignement supérieur universitaire (hors certaines spécialités médicales) sont découpées en trois cycles[5].

- Le premier cycle de trois ans est ouvert à tous les titulaires du baccalauréat. Ce cycle permet à l'étudiant d'obtenir une Licence ou une Licence professionnelle. Au cours des deux premières années et dépendant de la composante d'étude, l'étudiant peut obtenir le diplôme d'études universitaires scientifiques et techniques (DEUST), ou encore le diplôme universitaire de technologie (DUT). Ces deux diplômes répondent aux attentes des employeurs et visent l'employabilité immédiate des diplômés à l'issue de la formation.
- Le deuxième cycle permet aux étudiants ayant obtenus leur licence de continuer leurs études supérieures, d'approfondir leurs connaissances et de s'initier à la recherche scientifique. Ce cycle dure deux ans et débouche sur le diplôme de master. Dans une école d'ingénierie et suite à 5 ans d'études supérieurs, l'étudiant obtient son diplôme d'ingénieurs.

- Le troisième cycle correspond à la formation doctorale. Différents diplômes peuvent être délivrés comme le doctorat de la recherche, le doctorat d'exercice, Habilitation à diriger des recherches, Diplôme d'études spécialisées du secteur santé.

La formation doctorale

On s'intéresse dans cette partie à donner le diagramme de classes relatif aux thèses et les différentes informations associées à ces thèses tel que l'étudiant qui a réalisé la thèse, son directeur de thèse, les différents membres du jury, etc…

L'école doctorale participe à la formation des doctorants qui préparent leur thèse de doctorat au sein d'une ou plusieurs unités de recherche. Le jour de la soutenance, le doctorant présente ses travaux de recherche devant un comité composé de deux rapporteurs, un président de jury ainsi que son directeur de thèse et son co-directeur de thèse s'il y en a. Les membres de jury doivent avoir au moins le grade de maître de conférences. Les rapporteurs ainsi que le président de jury peuvent être du personnel de l'université de Strasbourg ou appartenant à une autre université.

Ce diagramme de classes montre aussi les différentes catégories du personnel enseignant dans l'université de Strasbourg. En effet, il existe 3 types d'enseignants:

Enseignant Chercheur (EC), Enseignant Non Chercheur (ENC) et Vacataire. Les EC peuvent être des EC associés ou des EC titre principal. Dans la caétgorie des EC associés, on trouve les Professeurs des universités associés, les Maîtres de Conférences associés et les Attachés temporaires de recherche (ATER). Tandis que dans la catégorie des EC titre principal, on trouve les Professeurs des universités, les Maître de Conférences (avec HDR ou sans HDR).

Chaque enseignant chercheur est attaché à une unité de recherche mais peut éventuellement avoir un rattachement à un autre laboratoire.

L'université a un grand système des Unités des directions et des services qui divisent Services rattachés à la Présidence et à la Direction générale des services, Domaine d'appui aux missions et Domaine de gestion des ressources. Domaine de gestion des ressources fournit des Outils numériques qui peuvent être Services pédagogiques, Gestion vie universitaire, Services d'infrastructures, Gestion finances, etc…

Chapitre 4 : Implémentation et résultats

4.2 État de l'art

4.2.1 Construction des ontologies

4.2.1.1 Théorique

À cause de la complexité des structures de l'Université de Strasbourg : Campus, Antennes, Formations, Composantes, Musées, ... il faut construire une ontologie pour l'université pour la rendre lisible et intelligible. A partir de cette ontologie, toute personne sera capable d'avoir une idée claire sur les différentes structures de l'université.

Dans ce projet, l'utilisation d'une ontologie pour la construction de la cartographie des savoirs a deux avantages: elle permet d'avoir un vocabulaire intégré, compréhensible et partagé par tous les utilisateurs de l'université de Strasbourg et de trouver les incohérences entre les différentes structures de l'université s'il y a.

Une ontologie est composée essentiellement de classes, d'individus, d'attributs et des restrictions sur les attributs. Il existe plusieurs langages de représentation des connaissances, parmi lesquels le langage OWL[2] qui fournit un nombre de constructeurs pour représenter les classes, les individus, etc.

- Les classes décrivent l'ensemble des concepts pour un domaine. Ces concepts sont organisés selon une hiérarchie taxinomique. En effet, une classe peut avoir des sous-classes qui représentent des concepts plus spécifiques que la super-classe (ou classe supérieure).
- Les individus correspondent aux instances d'une classe.
- Les attributs décrivent les relations entre des classes ou entre des individus. Il y a deux types d'attributs: *Object Property* et *Data Property*. *Object Property* est la relation entre deux classes ou deux individus.
- Des restrictions[3] sur les attributs (facettes (appelées parfois restrictions de rôles))

Il n'existe pas qu'un seul mode ou qu'une seule méthodologie « correcte » pour développer des ontologies. Dans le cadre du projet nous avons suivi la méthodologie présentée par Noy [12]. Les étapes principales pour construire une ontologie :

- Étape 1: Définir le domaine et la portée de l'ontologie
- Étape 2 : Envisager une éventuelle réutilisation des ontologies existantes
- Étape 3 : Énumérer les termes importants dans l'ontologie
- Étape 4 : Définir les classes et la hiérarchie des classes

[2] www.w3.org/TR/owl-ref/
[3] http://www.w3.org/TR/owl-ref/#Restriction

- Étape 5 : Définir les propriétés des classes – attributs
- Étape 6 : Définir les facettes des attributs
- Étape 7 : Créer les instances

4.2.1.2 Outils :

Maintenant, il y a plusieurs outils appelés “éditeurs d'ontologie” pour construire une ontologie. Parmi tous ces outils, nous choisissons cinq outils que les gens utilisent plus : *Apollo, OntoStudio, Protégé, Swoop, TopBraid Composer Free Edition*. Nous allons donner une vue d’ensemble de chaque outil pour trouver un outil qui est adapté plus avec notre projet.

4.2.1.2.1 Apollo

Apollo est développé par *The Open University*. Il est une application de modélisation des connaissances conviviale. Chaque ontologie d’*Apollo* contient des classes et ses restrictions comme relations, fonctions, etc… Chaque classe contient des instances qui héritent les restrictions de sa classe. La base de connaissances d’*Apollo* se compose d'une organisation hiérarchique des ontologies. Les ontologies peuvent être héritées d'autres ontologies et elles peuvent les utiliser.

Malheureusement, *Apollo* ne fournit pas une vue graphique, web, l’extraction de l’information, et multi-utilisateurs.

4.2.1.2.2 OntoStudio

OntoStudio est le plus répandu de la modélisation commerciale pour créer et maintenir des ontologies en utilisant des moyens graphiques. Il peut être téléchargé et utilisé gratuitement pendant un mois.

OntoStudio peut importer plusieurs structures, schémas et modèles. Il est basé sur une architecture client / serveur pour supporter muti-utilisateurs. Les ontologies sont gérées dans un serveur central et les clients peuvent accéder et modifier ces ontologies.

4.2.1.2.3 Protégé

Protégé est créé par *Stanford University School of Medicine*. Il est une plate-forme libre, open-source et disponible gratuitement pour télécharger.

Protégé fournit des outils pour construire des modèles de domaine et des applications de la base de connaissances avec des ontologies. Il met en œuvre un ensemble riche de structures connaissances de modélisation et des actions qui prennent en charge la création, la visualisation et la manipulation des ontologies dans différents formats de représentation.

L'avantage significatif de *Protégé* est son extensibilité. *Protégé* permet de construire et de traiter de grandes ontologies de manière efficace. *Protégé* contient le type le plus populaire de plug-ins est plug-ins de l'onglet qui permet d'ajouter des nouveaux plug-ins directement et facilement dans *Protégé*, nous ne devons pas chercher et télécharger le plug-ins ailleurs.

4.2.1.2.4 Swoop

Swoop est un open-source. Il fournit un environnement d'ontologie multiple, par lequel les entités et les relations à travers diverses ontologies peuvent être comparées, éditées et fusionnées de façon transparente. La navigation peut être simple et facile en raison des capacités hyperlien dans son interface. Les utilisateurs peuvent réutiliser les données ontologiques externes simplement par un lien vers l'entité externe, ou ils peuvent importer l'ontologie externe dans *Swoop*.

4.2.1.2.5 TopBraid Composer Free Edition

TopBraid Compositeur qui est développé par *TopQuadrant* est disponible en trois éditions: *Free Edition* (FE), *Standard Edition* (SE), *Maestro Edition* (ME) .

TopBraid Compositeur (FE), un composant de *TopBraid Suite* est un outil de développement pour les ontologies. *TopBraid Compositeur* (FE) peut charger et sauvegarder tous les fichiers *OWL2* dans des formats tels que *RDF / XML* ou tortue.

TopBraid Composer peut être utilisé dans un mode utilisateur unique travaillant avec ontologies stockées sous forme de fichiers ou dans une base de données. Il ne permet pas multi-utilisateurs. *TopBraid Compositeur* (FE) peut télécharger et évaluer la version complète pour une période d'évaluation de 30 jours.

4.2.1.2.6 Comparaison des outils

Une comparaison exhaustive entre les outils présentés ici peut être trouvée dans l'article [1].

Pour résumer, *Apollo, Protégé* et *Swoop* sont open source pour créer et maintenir les ontologies. *OntoStudio* et *TopBraid Composer* (FE) demandent une licence de logiciel. Protégé et *OntoStudio* disposent des outils graphiques d'ontologies. Le *Swoop* est une application basée sur le Web. *Protégé*, *TopBraid Compositeur* (FE), *Swoop* et *OntoStudio* éditeurs fournissent une documentation ontologie, l'ontologie import / export vers différents formats, une vue graphique des ontologies. Les outils *Protégé*, *Swoop*, *OntoStudio*, et *TopBraid* fournissent multi-utilisateurs. Après une étude de ces outils, nous constatons que l'outil Protégé est meilleur pour nous même *OntoStudio* et *TopBraid* sont aussi bien mais ils ne sont pas gratuits. *Apollo* ne supporte pas multi-utilisateurs et il est limité au langage importé et au langage exporté. *Swoop* ne donne pas une vue

graphique et des bibliothèques des ontologies. En plus, *Swoop* ne supporte pas le langage exporté *OWL*.

4.2.2 Visualisation des ontologies

4.2.2.1 Théorique

Ontologies, des ensembles de concepts et ses relations entre eux dans un domaine spécifique, deviennent un utile outil dans les zones des bibliothèques numériques, le web sémantique, et la gestion de l'information personnalisée. La demande de visualiser des ontologies augmente rapidement. Visualisation d'une ontologie est graphiques représentations visuelles des concepts et ses relations.

Visualisation des ontologies ne sont pas facile. Une ontologie est quelque chose de plus qu'une hiérarchie de concepts. Elle contient aussi des autres relations entre des concepts. En outre, chaque concept a des instances qui pourraient être en nombre d'un ou deux à mille. Il est difficile de visualiser toutes les composants de l'ontologie pour que l'utilisateur puisse comprendre facilement.

4.2.2.2 Outils :

Il y a plusieurs outils pour visualiser une ontologie mais chaque outil a des besoins et fonctions différentes. Dans la partie “Construction des ontologies”, nous avons choisi l’outil Protégé pour notre projet. Donc nous étudions quelques outils qui sont des plug-ins de Protégé : *OWLViz, OntoSphere, VOWL*. En plus, nous étudions aussi quelques outils de visualisation des données mais sur un site Web : *WebVOWL, D3.layout.tree.*

4.2.2.2.1 OWLViz

OWLViz peut visualiser l'ontologie sous une forme d'un arbre, la racine est à gauche et les feuilles sont à droite sur l'écran d'ordinateur. Il combine l'édition de l'ontologie avec visualisation. L'utilisateur peut modifier, rechercher et filtrer les classes de l'ontologie *OWLViz*, il ne doit pas déplacer vers l'éditeur de classe *Protégé*. Cependant, *OWLViz* ne supporte pas l'animation graphique, la visualisation sur les instances de concepts. En outre, il ne supporte pas le zoom-in ou zoom-out. *OWLViz* ne peut pas rendre une ontologie qui a dix ou plusieurs niveaux de sous-classes. Pour visualiser une ontologie dans *OWLViz*, un logiciel de plug-in appelé *GraphViz*[4] est nécessaire en plus de la plate-forme *Protégé*. D'une manière générale, *OWLViz* ne convient que pour la visualisation d'une ontologie simple.

[4]http://www.graphviz.org

4.2.2.2.2 OntoSphere

OntoSphere emploie des graphiques 3D. *OntoSphere* peut visualiser une grande ontologie à grande échelle. *OntoSphere* visualise l'ensemble du modèle hiérarchique sous la forme haut-bas ou gauche-droite. Il permet de visualiser l'ensemble des concepts de domaine autour d'un grand domaine hyperbolique, qui est le concept de base. Le problème avec cette technique est que la relation d'héritage et les relations entre les concepts ne peuvent pas être affichées.

4.2.2.2.3 VOWL

VOWL visualise tous les composants définis dans *OWL*. *VOWL* distingue différents composants en représentant les classes dans les cercles alors que tout le reste est présenté sous forme de rectangles. *VOWL* permet à toutes les classes sans liaison directe de se déplacer loin du centre. *VOWL* fournit des fonctions pour que l'utilisateur puisse visualiser les concepts et ses relations facilement comme le zoom, le changement le distance entre des classes. *VOWL* a été mis en œuvre dans deux outils différents à ce jour, qui démontrent son applicabilité: *ProtégéVOWL* a été réalisé sous la forme d'un plug-in Java pour la version de bureau de l'éditeur d'ontologie Protégé, *WebVOWL* est une application autonome basé sur des technologies Web ouvertes.

4.2.2.2.4 WebVOWL

WebVOWL a mise en œuvre des fonctions comme le plugin VOWL pour l'éditeur Protégé. *WebVOWL* existe comme une application autonome et non pas comme un plug-in de Protégé. Il permet de importer une ontologie sous la forme *OWL* ou *JSON* directement sur le web. Nous pouvons exporter l'ontologie sous la forme *JSON* ou *SVG* (une image). *WebVOWL* fournit un certain nombre de filtres qui aident à réduire la taille du graphe de *VOWL*.

4.2.2.2.5 D3.layout.tree

Comme *WebVOWL*, D3.layout.tree est une application basé sur la galerie D3[5]. Il est gratuit à télécharger et à utiliser. Il permet de visualiser les données sous la forme un arbre de gauche à droite. Cet outil permet à l'utilisateur de voir clairement et facilement les structures d'une ontologie. Mais il est adapté seulement à l'ontologie qui n'a que les relations héritées.

4.2.2.2.6 Comparaison des outils

Feature	OWLViz	OntoSphere	VOWL	WebVOWL	D3.layout.tree
Concept	Oui	Non	Oui	Non	Oui

[5]https://github.com/d3/d3/wiki/Gallery

Hiérarchie					
Niveau maximal	10	illimité	illimité	illimité	illimité
Espace	Optimisation limitée, ne peut pas tenir un modèle complexe dans un écran	Bien optimisation, aucun chevaucheme nt	Bien optimisation, mais les étiquettes se chevauchent	Bien optimisation, mais les étiquettes se chevauchent	Bien optimisation, aucun chevauchement
Zoom	Non	Oui	Oui	Oui	Non
Filtrage du concept	Oui	Oui	Non	Oui	Non
Édition du concept	Oui	Oui	Oui	Oui	Non
Système exigences	Oui	Oui	Oui	Non	Non
Sur le web	Non	Non	Non	Oui	Oui
Supporte les relations entre des classes (pas seulement les relations héritées)	Oui	Oui	Oui	Oui	Non

Table 1: Comparaison des outils de visualisation de l'ontologie

Pour résumer, OWLViz, D3.layout.tree visualisent l'ontologie sous la forme d'un arbre. OntoSphere, VOWL et WebVOWL ne visualisent pas sous forme d'un arbre. Tous les outils visualisent les ontologies avec des étiquettes et les nœuds se chevauchent sauf OntoSphere et D3.layout.tree. WebVOWL et D3.layout.tree n'a pas de système exigence pour visualiser une ontologie et ils sont les applications basées sur le web. Dans le cadre de notre projet, nous étions amenés à développer un outil pour que tout le monde puisse utiliser n'importe quand et n'importe où. Malgré les avantages des outils qui sont plug-ins

de Protégé, mais nous devons choisir un outil sur le web. En plus, notre ontologie est très grande et très compliquée, elle contient beaucoup de relations entre les classes qui ne sont pas des relations héritées. C'est pour toutes ces raisons que nous avons opté à utiliser l'outil WebVOWL.

4.3 Implémentation du système

4.3.1 Conception de l'ontologie par l'outil Protégé

Vu les points forts de l'outil *Protégé*, présenté dans la section précédente, nous avons trouvé qu'il est le plus adapté à utiliser. Afin de concevoir l'ontologie de l'Université de Strasbourg, nous avons utilisé la version 4.3 de *Protégé*. Dans notre ontologie, les classes représentent les différentes structures de l'université (campus, association, composante,...); les relations entre les classes peuvent être des *Object property* ou des *Data property*; les individus de chaque classe qui sont des instances de chaque partie de l'université; les relations entre des individus qui concernent avec des relations entre des classes.

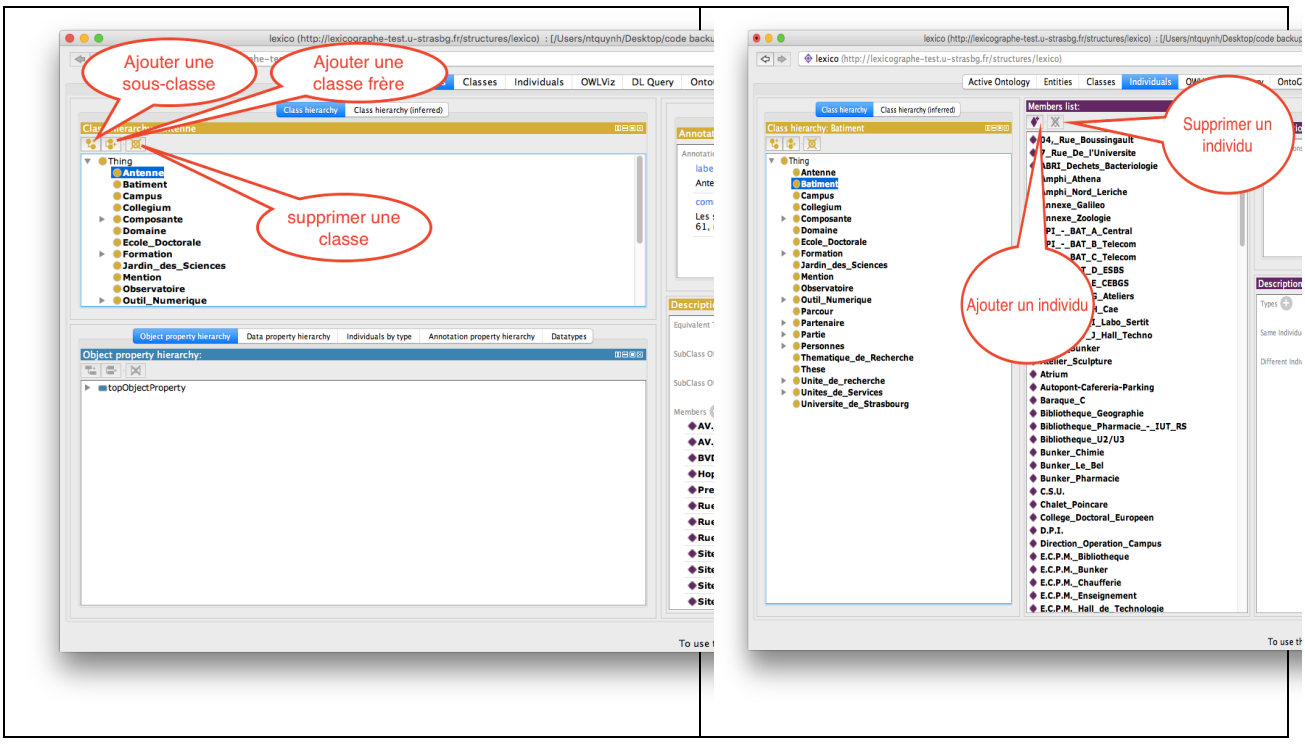

Figure 2: Création d'une nouvelle classe et un nouvel individu dans Protégé

Afin de construire l'ontologie, il faut tout d'abord identifier les classes. La figure 2 illustre l'ajout des classes avec *Protégé*. Dans cette figure, nous cliquons sur la classe Antenne. Un simple clic sur le bouton « Ajouter une classe frère » permet d'ajouter une nouvelle classe qui est meme level avec la classe Antenne (sous classe de *Thing*). Pour

ajouter une classe fille à une classe mère déjà existante, il suffit d'utiliser le bouton « Ajouter une sous-classe ». Dans la figure 2, quand nous créons une nouvelle classe avec le bouton « Ajouter une sous-classe », cette classe est définie comme une classe fille de la classe Antenne. Le bouton « Supprimer une classe » est utilisé pour supprimer une classe si nous choisissons une classe dans l'ontologie et cliquons sur ce bouton.

Dans Protégé, pour créer un nouvel individu pour une classe donnée, il suffit de sélectionner la classe concernée et choisir l'onglet *Individuals*. En effet, comme le montre la figure 2, le bouton « Ajouter un individu » pour créer un nouveau individu de la classe et le bouton « Supprimer un individu » pour supprimer un individu.

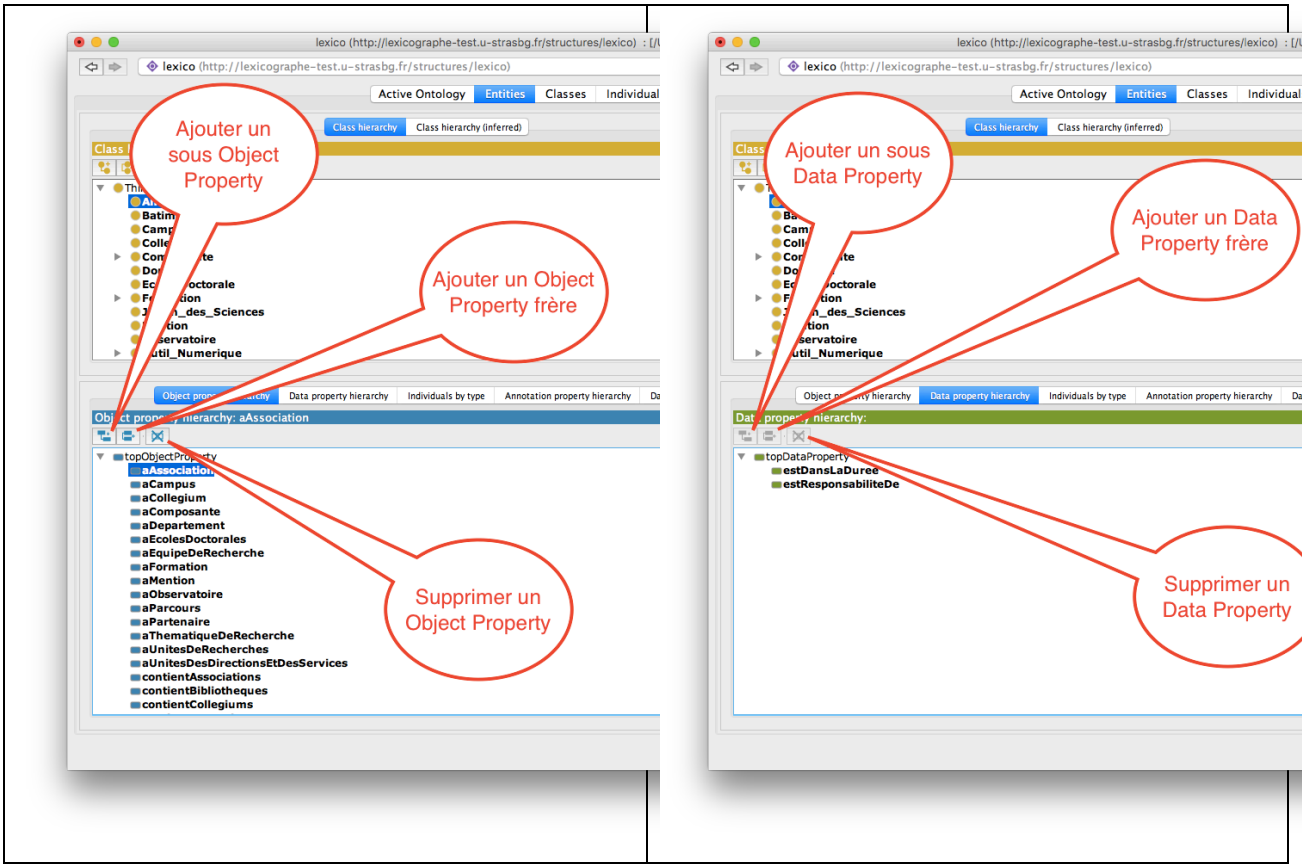

Figure 3: Ajout des Object property et des Data Property

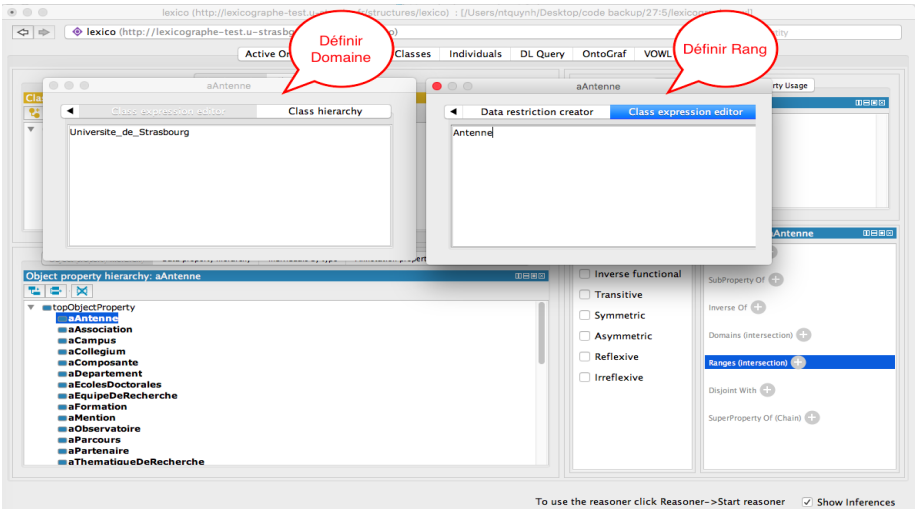

Figure 4: Définition du domaine et du rang des property dans Protégé

Afin de définir des relations entre les classes, il faut créer des Object Property ou des Data Property tout en spécifiant le domaine et le rang de cette relation. Par exemple dans la figure 3, nous avons choisit l'Objet Property aAssociation, nous pourrons créer un nouveau Object Property qui peut être l'enfant (sub property) de l'Object property aAssociation si nous choisissons le bouton « Ajouter un sous Objet Property » ou elle peut être frère (sibling property) de l'Object property aAssociation si nous choisissons le bouton « Ajouter un Objet Property frère ». Si nous choisissons une Object Property et cliquons sur le bouton « Supprimer un Objet Property », Protégé va supprimer cette Object Property. Avec Data Property, Par exemple dans la figure 3, nous avons choisit un Data Property estDansLaDuree, nous pourrons créer un nouveau Data Property qui peut être l'enfant (sub property) de Data Property estDansLaDuree si nous choisissons le bouton « Ajouter un sous Data Property » ou elle peut être frère (sibling property) de l'Data Property estDansLaDuree si nous choisissons le bouton « Ajouter un Data Property frère ». Si nous choisissons un Data Property et cliquons sur le bouton « Supprimer un Data Property », Protégé va supprimer cette Data Property. La relation peut être enrichie par la définition du domaine et du rang comme le montre la figure 4.

L'ajout des relations entre des individus dépend des relations déjà existantes entre leurs classes correspondantes. Dans la figure 5, nous créons la relation estDansCampus entre les deux individus Faculte De Chimie et Campus_Esplanade.

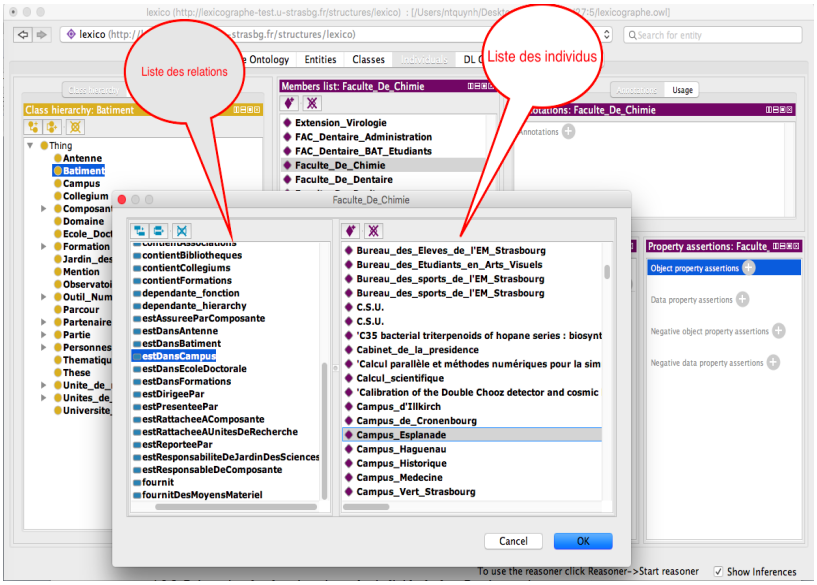

Figure 5: Création une relation entre deux individus dans Protégé

4.3.2 Visualisation de l'ontologie avec *WebVOWL*

Pour visualiser l'ontologie dans une application web, nous avons utilisé l'outil *WebVOWL*. C'est un logiciel open source téléchargeable sur cette adresse (*https://github.com/VisualDataWeb/WebVOWL*).

- Préparation des fichiers de données: La création de l'ontologie sous *Protégé* a généré le fichier *OWL lexicographe.owl*. Pour pouvoir utiliser *WebVOWL*, ce fichier doit être converti en un fichier json. L'outil *OWL2VOWL* [6] est utilisé pour convertir le fichier *lexicographe.owl* en *lexicographe.json*. Ce fichier contient toutes les informations relatives aux classes, aux relations ainsi que l'ensemble des individus de chaque classe. Afin de visualiser les individus, nous avons écrit du code Ruby pour créer un fichier individual.js contenant toutes les informations relatives aux individus
- Déployer le projet *WebVOWL*: Afin de déployer notre projet, nous avons placé le fichier *lexicographe.json* dans le dossier *src/app/data* et le fichier *individual.js* dans le dossier *src/app/js*. Le déploiement se fait par la commande : ***npm run-script release***.
- Afin de visualiser l'ontologie, nous avons eu accès au fichier *index.html* dans le dossier *deploy*.

[6] https://github.com/VisualDataWeb/OWL2VOWL

L'outil *WebVOWL* permet de visualiser une ontologie en entier en affichant toutes les classes et les relations. Ceci ne convient que pour les petites ontologies avec peu de classes et de relations. Malheureusement, l'ontologie de l'Université de Strasbourg contient un nombre important de classes et de relations comme le montre la figure 6.

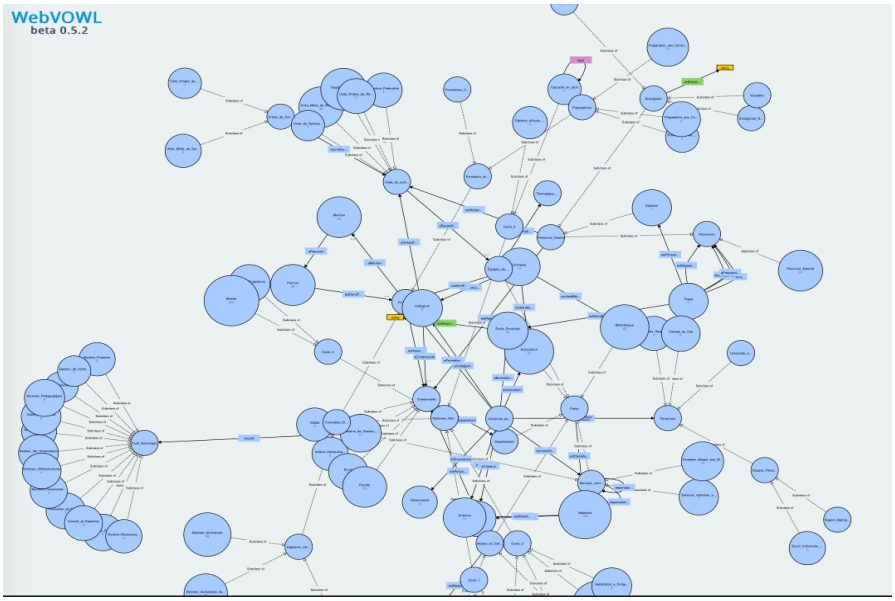

Figure 6: Visualisation de l'ontologie dans WebVOWL

Afin de répondre aux exigences de notre système, à savoir permettre à l'utilisateur de visualiser l'ontologie de l'université de Strasbourg en affichant toutes les structures de l'université, nous avons dû apporter des améliorations au code de *WebVOWL* afin d'afficher l'ontologie sous forme d'un arbre en deux niveaux, marquer le chemin pour accéder à un nœud précédent et enfin pouvoir visualiser tous les individus d'une classe donnée. L'interface du *WebVOWL* est composée principalement de deux parties: une pour l'affichage de l'ontologie (partie gauche) et une pour afficher la liste des individus (partie droite).

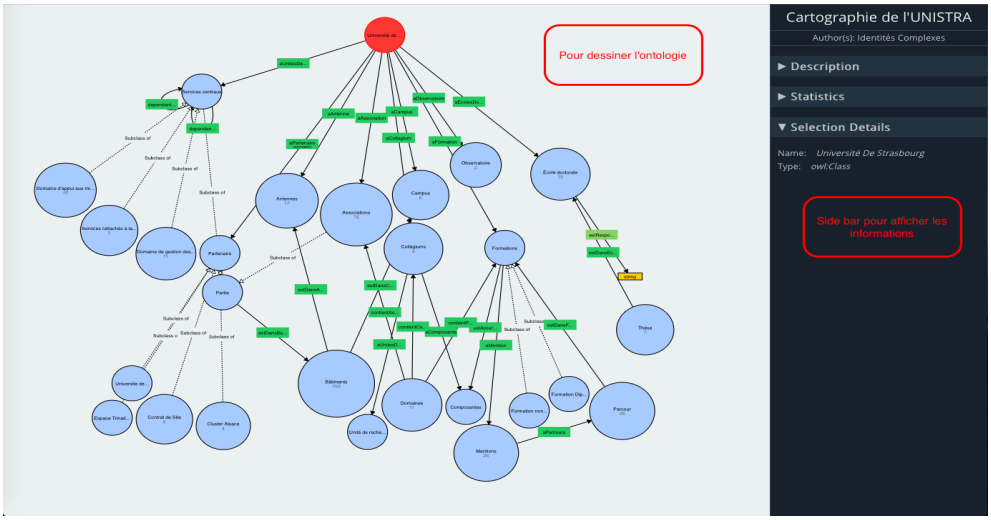

Figure 7: Visualisation de l'ontologie sous forme d'un arbre en deux niveaux (Partie 1)

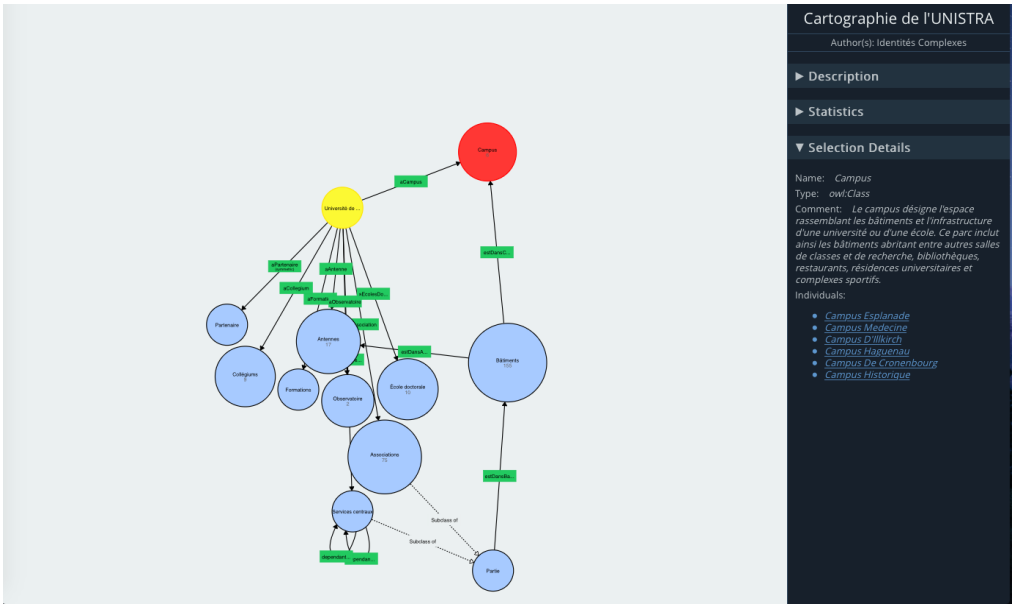

Figure 8: Visualisation de l'ontologie sous forme d'un arbre en deux niveaux (Partie 2)

- Afin que l'ontologie puisse être affichée en deux niveaux, nous avons créé un nouveau fichier *clickToExpand.js*. Chaque fois que l'utilisateur choisit une classe de l'ontologie, des fonctions dans *clickToExpand.js* vont être exécutées pour filtrer toutes les classes de l'ontologie. L'appel de ces fonctions permettra de décider quelles classes afficher. Les classes gardées sont contenues dans un set *connectedNodes*. Par exemple dans notre système, nous allons garder les classes qui sont dans le niveau 1 ou le niveau 2. Afin de visualiser les classes de niveau supérieur à 2, il suffit de sélectionner une classe dans la hiérarchie. Par exemple, l'université de Strasbourg est composée de Campus qui contient un ensemble de Bâtiments qui eux-mêmes sont composés de parties. L'affichage d'une ontologie en deux niveaux permettra d'afficher jusqu'au niveau des Bâtiments. Pour afficher les parties, il suffit de sélectionner les Campus.

Pour visualiser l'ontologie sous forme d'un arbre, nous avons modifié le code dans le fichier *graph.js*.

- Dans le fichier *graph.js*, nous recalculons la position de chaque classe dans *connectedNodes*. Nous considérons les liens qui connectent deux classes dans *connectedNodes*. Chacun de ces liens a un domaine et un rang, nous calculons la différence de niveau entre son domaine et son rang. La classe ayant le niveau le moins élevé sera placé en racine et les autres classes formeront la suite de l'arbre.
- Nous calculons la position de la classe qui est en plus haute (position A). Si cette classe n'est pas la classe que nous venons de cliquer, il faut changer la position

de la classe cliquée est en plus haute (position A). Ceci est pour assurer que la classe cliquée est toujours en plus haute (position A).

- Nous recalculons aussi la position de chaque classe pour que toutes les classes ne se chevauchent pas.

  Par exemple dans la figure 7, la classe choisie (Université de Strasbourg) deviendra la racine de l'arbre avec le niveau 0 et toutes les autres classes qui sont dans *connectedNodes* et liées à Université de Strasbourg sont affichées en deux niveaux. Dans la figure 8, quand nous cliquons sur la classe Campus, la classe Campus devient la racine de l'arbre et le système affiche les classes qui sont dans *connectedNodes* et liées à la classe Campus.

Afin que l'utilisateur puisse faire un retour et afficher la partie de l'ontologie précédemment visualisée, nous avons créé un nouveau fichier focuser.js qui permet de colorier le nœud cliqué en rouge et le nœud précédemment sélectionné en jaune.. Dans la figure 7, la classe Université de Strasbourg représente la racine de l'arbre et est de couleur rouge. Dans la figure 8, nous avons choisi la classe Campus qui devient la racine de l'arbre affiché. Par conséquent le nœud représentant le Campus est en rouge et celui représentant l'Université de Strasbourg est en jaune. Chaque fois que nous sélectionnons une classe, les individus sont affichés dans le sidebar, la partie droite de l'interface graphique. Dans la figure 39, les individus de la classe Campus sont affichés: Campus Esplanade, Campus Medecine, Campus D'Illkirch, Campus Haguenau, Campus De Cronenbourg, Campus Historique.

4.3.3 Visualisation des individus dans l'ontologie par *WebVOWL*

Pour afficher toutes les informations relatives à un individu, nous avons créé un nouveau fichier *tree.js* dans le dossier *src/app/js* qui permet de décrire toutes les informations relatives à chaque individu.

Les individus sont affichés dans le sidebar de l'interface graphique (partie droite) Afin d'avoir de plus amples informations sur un individu bien déterminé, il suffit de sélectionner un individu et un graphe représentant ces informations sera affiché

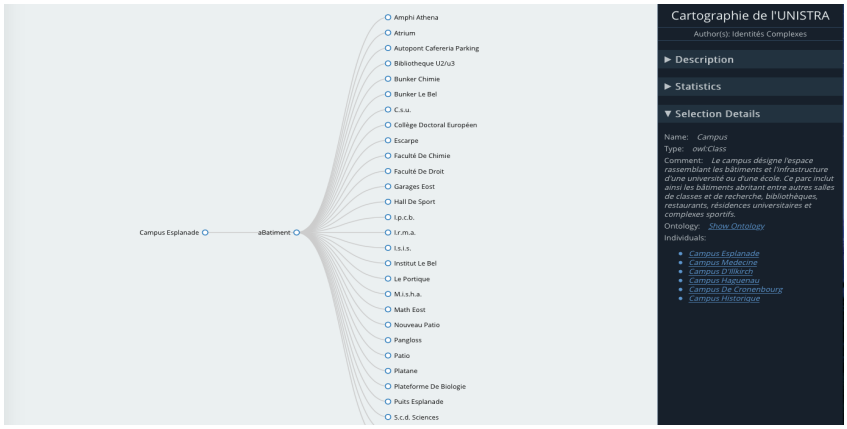

Figure 9: Visualisation de l'individu Campus Esplanade

dans la partie gauche de l'interface graphique. Afin d'afficher l'ontologie, nous avons ajouté un bouton *Show Ontology* permettant de faire un retour un arrière.

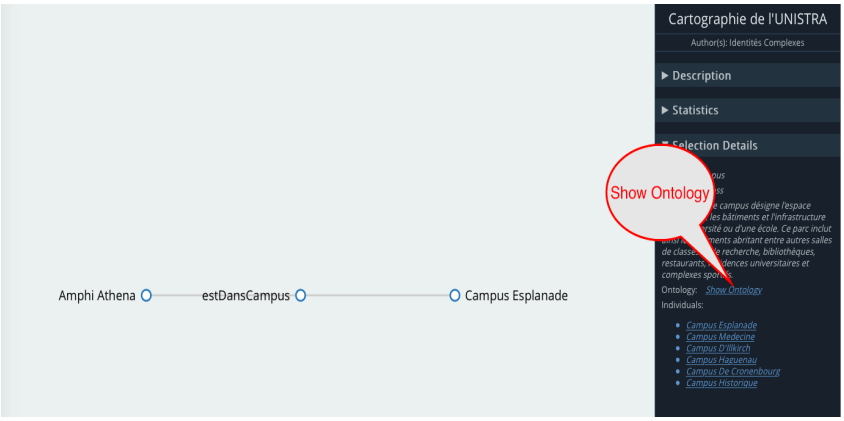

Figure 10: Visualisation de l'individu Amphi Athena

La figure 9 montre le graphe relatif à l'individu Campus Esplanade qui affiche toutes les informations de ce campus, à savoir les bâtiments composant ce campus. A partir de ce graphe, il est possible de sélectionner un des bâtiments et afficher un graphe décrivant le bâtiment sélectionné. A partir du graphe affiché dans la figure 9, nous sélectionnons l'individu Amphi Athena pour afficher son graphe dans la figure 10.Si nous voulons afficher l'ontologie de nouveau, nous devons cliquer sur le bouton *Show Ontology*.

4.3.4 Récupération des données de la base de thèses de l'Université de Strasbourg

Dans le cadre de ce projet, nous étions appelés à part la représentation de la cartographie des savoirs sous forme d'ontologie, à découvrir les liens "cachés" qui ne sont pas représentés dans l'ontologie. En effet, nous nous sommes intéressés à découvrir les liens entre les unités d'enseignement et les unités de recherche en se basant sur les informations extraites des thèses et en particulier des membres de jury. Plus deux enseignants faisaient partie d'un même comité de thèse, plus il y a un lien entre leurs unités d'enseignement ou leurs unités de recherche. Pour cela, à partir du site *http://www.theses.fr* nous avons récupéré 1613 thèses de l'Université de Strasbourg. Ces thèses ont été soutenues entre 2012 et 2015. Les données relatives aux thèses étaient sous forme de texte qui n'est pas visible pour les gens. Pour cela, nous avons pré-traité ces données en distinguant les informations autour d'une thèse: titre, auteur, directeur de thèse, rapporteur, membre de jury, directeur de jury, école doctorale. Par la suite, ces thèses ont été stockées dans une base de données PostgreSQL.

En se fondant sur la base de données créée, nous avons développé un code Ruby pour fouiller les données de thèse et compter le nombre de fois de participation de deux enseignants chercheurs dans un même comité de thèse comme le montre la figure 11. Nous avons opté de sélectionner uniquement les enseignants qui ont participé au moins 3 fois. Ensuite, nous avons cherché pour chaque enseignant son unité d'enseignement et son unité de recherche (figure 12). En se basant sur les informations des figures 11 et 12, nous avons écrit un code Ruby pour trouver le nombre de fois d'apparition de composante ou d'une unité de recherche dans un même comité de thèse comme le montre la figure 13.

Figure 11: Occurrences de personnes

Personne	Composante	Unite de recherche
Mestre Christian	Faculté de Droit, de Sciences Politiques et de Gestion	Centre d'études internationales et européennes (CEIE) - EA 7307
Badariotti Dominique	Faculté de Géographie et d'Aménagement	Laboratoire Image, Ville et Environnement (LIVE) - UMR 7362
Weber Christiane	Faculté de Géographie et d'Aménagement	Laboratoire Image, Ville et Environnement (LIVE) - UMR 7362
Roesz Germain	Faculté des Arts	Approches contemporaines de la création et de la réflexion artistique (ACCRA) - EA 3402
Strickler Yves	Faculté de Droit, de Sciences Politiques et de Gestion	
Wiederkehr Georges	Faculté de Droit, de Sciences Politiques et de Gestion	Centre de Droit Privé Fondamental (CDPF) - EA 1351
Lehmann Yves	Faculté des Lettres	Centre d'Analyse des Rhétoriques Religieuses de l'Antiquité (CARRA) - EA 3094
Pernot Laurent	Faculté des Lettres	Centre d'Analyse des Rhétoriques Religieuses de l'Antiquité (CARRA) - EA 3094
Dumont Paul	Faculté des Langues et des Cultures Etrangères	Groupe d'Etudes Orientales, Slaves et Néo-helléniques (GEO) - EA 1340
Mazzoni Cristiana		Architecture, Morphologie/Morphogenèse Urbaine, Projet (AMUP) - EA 7309
Sauvage Jean-Pierre		Institut de Science et d'Ingénierie Supramoléculaires (ISIS) - UMR 7006
Mir Wais Hosseini	Faculté de Chimie	Chimie de la Matière Complexe, UMR 7140
Rémond Yves	Ecole Européenne de Chimie, Polymères et Matériaux	ICube - Laboratoire des sciences de l'ingénieur, de l'informatique et de l'imagerie (ICube) - UMR 7357
Sock Rudolph	Faculté des Lettres	Linguistique, Langues, Parole (LiLPa) - EA 1339
Vaxelaire Béatrice	Faculté des Lettres	Linguistique, Langues, Parole (LiLPa) - EA 1339
Metzger Marcel	Faculté de Théologie Catholique	Théologie Catholique et Sciences Religieuses (TCSR) - EA 4377
Storck Michel	Faculté de Droit, de Sciences Politiques et de Gestion	Droit religion entreprise et société (DRES) - UMR 7354
Kern Francis	Faculté des Sciences Economiques et de Gestion	Bureau d'Economie Théorique et Appliquée (BETA) - UMR 7522
Ackerer Philippe	Ecole et Observatoire des Sciences de la Terre	Laboratoire d'Hydrologie et de Géochimie de Strasbourg (LHyGeS) - UMR 7517
Younes Anis	Ecole et Observatoire des Sciences de la Terre	Laboratoire d'Hydrologie et de Géochimie de Strasbourg (LHyGeS) - UMR 7517

Figure 12: Composante et unité de recherche pour chaque enseignant chercheur

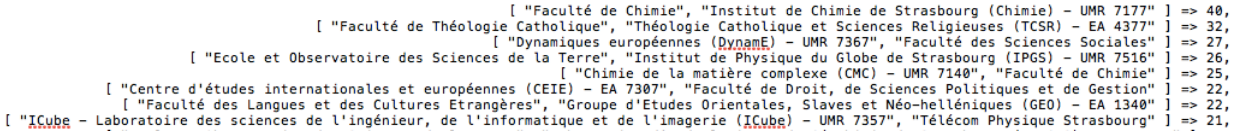

Figure 13: Occurrences de composantes et des unités de recherche

4.3.5 Visualisation les liens entre des composantes et des unités de recherche

Afin de visualiser les liens entre les composantes et les unités de recherche sous forme graphique, nous avons réutilisé le code sur *http://mbostock.github.io/d3/talk/20111116/bundle.html* comme l'illustre la figure 14.

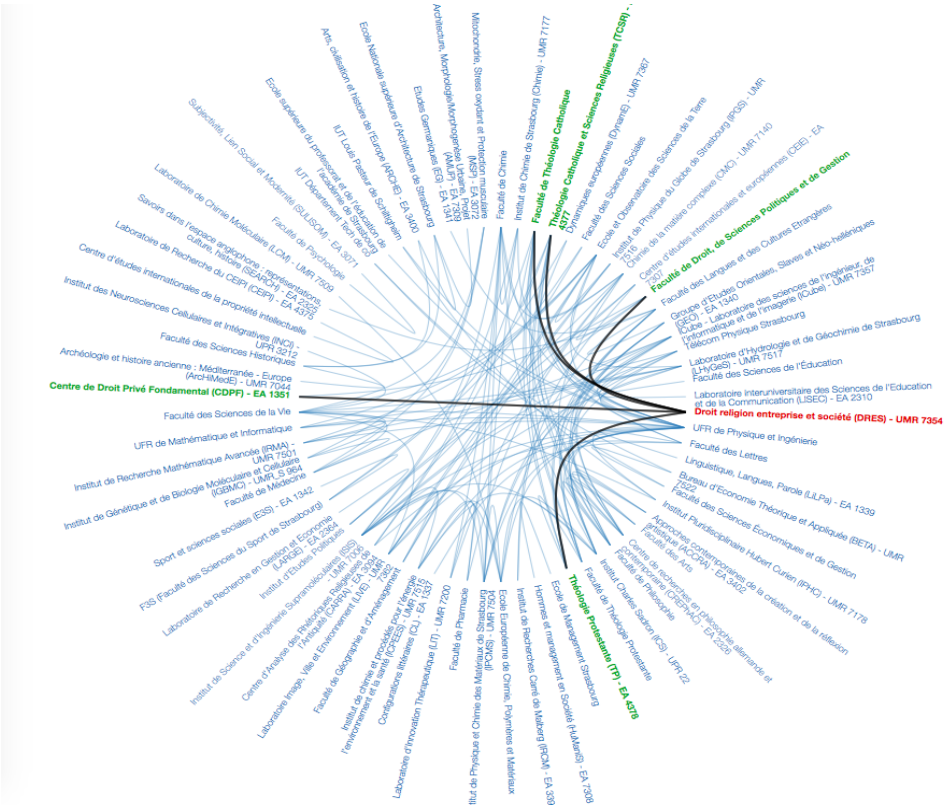

Figure 14: Les liens entre des composantes et des unités de recherche (partie 2)

Une fois que nous sélectionnons une composante ou une unité de recherche, elle sera coloriée en rouge et toutes les composantes ou les unités de recherche qui lui sont reliées sont de couleur verte comme le montre la figure 14. Nous pouvons choisir un lien pour voir la relation entre deux composantes.

Conclusion et Perspectives

Conclusion

L'université de Strasbourg est née suite à la fusion de trois universités strasbourgeoises. Cette fusion a permis à l'Université de Strasbourg de devenir la deuxième plus grande université en France. La pluridisciplinarité ainsi que le nombre croissant des étudiants et des enseignants a rendu l'université une institution complexe et riche. Dans le cadre du projet IDEX "Identités Complexes", nous étions appelés à construire une cartographie du savoir de l'Université de Strasbourg. Le but de cette cartographie était de pouvoir représenter le savoir de l'université et de pouvoir faciliter l'accès à l'information. La cartographie du savoir est basée sur la construction d'une ontologie capable de décrire toutes les structures de l'université. A travers cette ontologie, un utilisateur peut avoir une idée claire sur les formations offertes par l'université, la

recherche au sein des différentes unités de recherche ainsi que la vie dans les campus de l'université, etc.

A part la construction de l'ontologie, nous nous sommes focalisés dans la visualisation de l'ontologie ainsi construite. Pour cela, nous avons utilisé l'outil *WebVOWL* auquel nous avons apporté des améliorations pour qu'il réponde aux exigences et aux besoins de notre système. En particulier, nous avons ajouté des nouvelles fonctions pour qu'il soit possible de :

- Afficher la structure de l'ontologie sous la forme un arbre avec deux niveaux.
- Marquer le chemin pour que l'utilisateur puisse naviguer facilement entre les classes de l'ontologie.
- Visualiser les relations entre des individus.

Au-delà du développement de cet outil pour pouvoir naviguer sur l'ontologie grâce à sa structure, nous avons développé un outil d'apprentissage automatique et visualisation pour enrichir l'ontologie grâce aux liens entre les composantes et les unités de recherche.

Perspectives

Le travail présenté dans le cadre de ce mémoire de stage représente la synthèse de six mois de recherche en informatique. Comme beaucoup de travaux en informatique, ce travail n'est pas une fin. Donc, en perspective des améliorations peuvent être apportées à ce système.

Les structures de l'université de Strasbourg sont très grandes et très complexes. Nous avons eu l'occasion de rencontrer les personnes qui comprennent bien les structures de l'université mais pas toutes les structures. Il reste quelques structures qui n'ont pas été encore considérées, telles que les bibliothèques, les Unités des directions et des services, les musées et les collections.

Malheureusement, les informations associées à ces structures et les individus de chaque classe appartiennent aux différentes composantes, nous n'avons pas eu la possibilité d'accéder tous les bases de données pour les récupérer. En conséquence, notre ontologie n'est pas encore tout à fait complète. Néanmoins, toutes les bases ont été posées pour qu'un autre informaticien puisse prendre le relais et intégrer nos travaux dans un logiciel opérationnel exploitable par le projet dans le futur proche.