République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

UNIVERSITE Dr. TAHAR MOULAY SAIDA

FACULTE : TECHNOLOGIE

DEPARTEMENT : INFORMATIQUE

MEMOIRE

Présentée par
Ait Ahmed Nora
Idris Khodja Asma

Pour l'obtention du diplôme de
MASTER2 « L.M .d » en Informatique
Filière : Informatique
Option : Modélisation Informatique des Connaissances et du Raisonnement

---

Système de Recommandation de Cours à Base d'Ontologie

Soutenue publiquement, le ..../…./ 2018

Devant le jury composé de :

BENYAHIA Kada	**Président**	**MCA**	**Université Dr. Tahar Moulay Saida**
LATRECHE Abdelkrim	**Encadreur**	**MCA**	**Université Dr. Tahar Moulay Saida**
BOUARARA Hadj Ahmed	**Examinateur**	**MCA**	**Université Dr. Tahar Moulay Saida**

Année Universitaire 2017-2018

Résumé

Les systèmes de recommandation apportent une solution au problème de surcharge d'information et sont capables d'estimer l'intérêt d'un utilisateur pour une ressource donnée à partir de certaines informations relatives à d'autres utilisateurs similaires et aux propriétés des ressources. Dans ce mémoire nous avons présenté un système de recommandation de cours à base d'ontologie. Dans notre approche, l'ontologie de cours est intégré dans le processus de calcul de similarité sémantique entre le profil utilisateur et les descripteurs de cours, et qui est combinée avec similarité numérique, ce qui permet d'améliorer la précision de la recommandation et ainsi mieux répondre aux exigences des utilisateurs. Avec les premiers tests notre système donne des résultats encourageants.

Mots-clés *:* Système de recommandation, profil utilisateur, descriptif contenu, préférence, appariement, similarité sémantique, similarité numérique, ontologie.

Summary :

The recommendation systems provide a solution to the problem of information overload and are able to estimate the interest of a user for a given resource based on certain information relating to other similar users and the properties of the resources. In this report we presented an ontology-based course recommendation system. In our approach, the course ontology is integrated in the process of calculating semantic similarity between the user profile and the course descriptors , and which is combined with digital similarity, which improves the accuracy of the recommendation and thus better meet the requirements of users. With the first tests our system gives encouraging results.

Keywords: Recommendation system, user profile, descriptive content, preference, matching, semantic similarity, numerical similarity, ontology.

ملخص

تقدم أنظمة التوصية حلاً لمشكلة التحميل الزائد للمعلومات و تكون قادرة على تقدير مصلحة المستخدم لمورد معين استنادًا إلى معلومات معينة تتعلق بمستخدمين مشابهين آخرين وخصائص الموارد .في هذه الأطروحة ، قدمنا نظامًا لتوصية الدورة التدريبية المستندة إلى علم الوجود في منهجنا ، تم دمج دورة علم الوجود في عملية حساب التشابه الدلالي بين ملف تعريف المستخدم وأوصاف الدورة التدريبية ، والتي يتم دمجها مع التشابه الرقمي ، مما يحسن دقة التوصية وبالتالي تلبية متطلبات المستخدمين بشكل أفضل. مع الاختبارات الأولى يعطي نظامنا نتائج مشجعة .

الكلمات المفتاحية : نظام التوصية ، ملف تعريف المستخدم ، المحتوى الوصفي ، التفضيل ، المطابقة ، التشابه الدلالي ، التشابه العددي ، علم الوجود.

Remerciements

On remercie le bon dieu, qui nous a donné la force, la volonté et le courage pour terminer ce modeste travail.

On tiens à exprimer d'abord toute notre gratitude à notre encadreur Dr. ***Latreche Abdelkrim*** *maitre de conférence à l'université Dr .MOULAY TAHAR de la wilaya de SAIDA, pour son encadrement, ses conseils, ses directives et encouragements.*

Nos remerciement s'adressent à Mrs les juré pour l'intérêt qu'ils ont porté à ce travail en acceptant d'être examinateurs.

C'est un énorme remerciement qu'on s'adresse à nos parents et nos sœurs et frères et nos amis.

Merci à tout le corps professoral et administratif de l'université Dr. Moulay Tahar de la wilaya de Saida

Dédicaces

Je dédie ce travail a ceux qui m'ont encouragé et soutenu ;

A mes chers parents ;

A mes sœurs et frères ;

A mon marie et mes enfants ;

A tous mes proches et à ma belle-famille ;

A notre encadreur « Dr. Latreche Abdelkrim » ;

A mon binôme « Nora Ait Ahmed » ;

A tous le corps professoral et administratif de l'université Dr. Moulay Tahar de la wilaya de Saida ;

A tous nos amis et collègues qu'on a oublié de cité.

Asma

Dédicaces

Je dédie ce travail a ceux qui m'ont encouragé et soutenu ;

A mes chers parents ;

A mes sœurs « Fatima » et « Fadela » et mes frères ;

A notre encadreur « Dr. Latreche Abdelkrim » ;

A mon binôme « Idris Khodja Asma» ;

A tous le corps professoral et administratif de l'université Dr. Moulay Tahar de la wilaya de Saida ;

A tous nos amis et collègues qu'on a oublié de cité.

Nora

Liste des figures

Liste des tableaux

Table des matières

Chapitre 2

Ingénierie Ontologique

Chapitre 3

Conception et Réalisation

Introduction générale

Les systèmes d'informations actuels sont caractérises par leur volume croissant, leur hétérogénéité, et par le fait qu'ils ne sont pas suffisamment adaptes aux besoins des utilisateurs. Au vu de l'état actuel de ces systèmes en termes d'hétérogénéité de domaines, de sources, de représentation et de structuration des informations, l'accès a une information pertinente et adaptée aux utilisateurs est un vrai challenge. Les besoins de l'utilisateur sont difficiles a traiter, d'une part, parce qu'ils ne sont pas formules explicitement et, d'autre part, parce qu'ils sont évolutifs. La popularisation d'internet ainsi que l'explosion des services de recommandation de nos jour ont propulsé la recherche d'information (RI) au premier plan.

Les systèmes de recommandations (SRs) sont des outils puissants aidant les utilisateurs, en ligne, à résoudre le problème de la surcharge d'informations auquel ils sont confrontés aujourd'hui, avec l'avènement d'internet, en leur fournissant personnalisées. Ce sont des systèmes de personnalisation qui présentent aux utilisateurs les contenus les plus pertinents, en utilisant certaines informations concernant leurs préférences passées. Les SRs utilisent des profils représentant des préférences relativement stables des utilisateurs pour calculer des recommandations. Ce calcul se fait par la prédiction des scores qu'un utilisateur est susceptible d'attribuer au contenu. Les stratégies de recommandation se basent, généralement, sur le filtrage collaboratif (FC), le filtrage basé sur le contenu (FBC) ou sur une combinaison de ces deux approches. Le FC recommande des produits en se basant sur la similarité des préférences d'un groupe de clients connus comme étant des voisins. Cela suppose que les utilisateurs ayant des intérêts communs, dans le passé, continueront, probablement, à partager les mêmes goûts dans le futur. Les systèmes FBC utilisent l'appariement entre le profil d'un utilisateur et les descripteurs des contenus pour recommander les produits appropriés. Le calcul des recommandations se fait par la prédiction des scores qu'un utilisateur est susceptible d'attribuer aux contenus. Ce calcul nécessite souvent la présence d'une ontologie. Les ontologies offrent une connaissance partagée sur un domaine qui peut être échangée entre des personnes et des systèmes hétérogènes. Elles ont été définies en intelligence artificielle afin de faciliter le partage des connaissances et leur réutilisation. La définition explicite du concept ontologie soulève un questionnement qui est tout à la fois d'ordre philosophique, épistémologique, cognitif et technique. Une ontologie définit le vocabulaire partagé pour aboutir à une compréhension commune d'un domaine donné.

Dans ce mémoire nous proposons un système de recommandation sémantique de cours basé sur le filtrage collaboratif et sur une ontologie de cours. L'approche adoptée est en général dans le sens où nous pouvant l'appliquer pour recommandation de contenus en général (livre, Url, article, produit, film, chanson, etc.). Nous illustrons son principe pour affiner la recommandation de cours à un utilisateur.

Notre mémoire est organisé comme suit :

Le premier chapitre présente une vue générale sur les systèmes de recommandation ainsi que leurs avantages et inconvénients. Nous définissons d'abord ce qu'est un système de recommandation. Ensuite nous détaillons ses différents types. Enfin, nous passons en revue les différentes mesures de similarité et présentons une classification de ces mesures.

Le deuxième chapitre est dédié à éclaircir la notion de l'ontologie plus particulièrement en ingénierie des connaissances et ses différentes classifications. Nous présentons ensuite les principaux formalismes de représentation de connaissances, les langages de représentation, les outils d'édition et d'interrogation, etc.

Nous exposons notre approche dans le troisième chapitre. Nous commençons d'abord, par définir le modèle de données utilisé ainsi que la conception de notre application via les différents diagrammes UML. Ensuite nous expliquons la méthode adoptée et nous détaillons l'architecture générale du SR adoptant notre nouvelle approche. Nous y montrons, en détail, le fonctionnement des trois modules essentiels pour la résolution contextuelle, à savoir, *Création des profils utilisateurs, Résolution du contexte*, et *appariement profil/contenu*. Enfin, nous présentons les étapes suivies pour la construction de notre ontologie.

Nous clôturons ce mémoire par une conclusion et des perspectives.

Chapitre 1

Les systèmes de recommandation : vue d'ensemble

1.1. Introduction

Avec l'avènement d'internet, nous assistons aujourd'hui à une surcharge d'information. Par exemple, une personne qui désire lire un cours se retrouve devant un volume très grand propositions de cours. Ce qui rend la tache de choix d'un cours très difficile. Les systèmes de recommandation sont apparus pour remédier à ce problème.

Dans ce chapitre, nous commençons par définir ce qu'un système de recommandation. En suite, nous présentons les trois approches de filtrage qui permettent la recommandation. Nous enchainons en présentons les différentes mesures de similarité qui permettent aux systèmes de recommandation de faire l'appariement entre les concepts. En fin, nous terminons en citant les limites et inconvénients des systèmes de recommandation ainsi que quelques principaux travaux dans le domaine.

1.2. Historique

Les systèmes de recommandation ont été utilises afin de faire face au problème de surcharge et de richesse d'informations disponibles notamment a travers le Web ou les e-services. Les systèmes de recommandation visent à proposer à un utilisateur actif une ou des recommandations d'items susceptibles de l'intéresser. Ces recommandations peuvent concerner un article à lire, un livre à commander, un film à regarder, un restaurant à choisir, etc.

« Information Lens System » [1] peut être considère comme le premier système de recommandation. A l'époque, l'approche la plus commune pour le problème de partage d'informations dans l'environnement de messagerie électronique était la liste de distributions basée sur les groupes d'intérêt.

Quelques années plus tard, un certain nombre de systèmes académiques de recommandation ont vu le jour en 1994 et en 1995, tels que le système de recommandation d'articles d'actualités et de films développe par "GroupLens" [2] et le système de recommandation de musique "Ringo" propose par [3]. Ces deux systèmes sont également bases sur le filtrage collaboratif, des

livres [2], des vidéos, des films, des pages Web, des articles de nouvelles Usenet et des liens Internet.

Par la suite, avec l'essor de l'Internet et des applications Web, il y a eu un engouement pour les systèmes de recommandation qui se sont développes dans différents domaines d'applications. Nous pouvons en citer :

- Les systèmes de recommandation de films, tels que : Mobiles et Eachmovie.
- Les systèmes de recommandation de livres (Bookcrossing).
- Les systèmes de recommandation de musique (LastFM6).
- Les systèmes de recommandation d'articles d'actualités.
- Les systèmes de recommandation de blagues.
- Les systèmes de recommandations introduits sur des sites e-commerce (Amazon).
- Les systèmes de recommandation de restaurants.
- Les systèmes de recommandation intègres aux Extranets documentaires (l'Extranet documentaire du Crédit Agricole).
- Les systèmes de recommandations intègres aux moteurs de recherche (le moteur de recherche d'AOL).
- Les systèmes de recommandations implémentent sur des sites de recrutement (Job-Finder).Les systèmes de recommandations de citations bibliographique.

Pour tous les systèmes de recommandation développes jusqu'au nos jours, la collecte de données relatives aux utilisateurs et/ou aux items, représente une phase clé dans le processus de personnalisation. La section qui suit décrit en détails la typologie de données exploitables par les systèmes de recommandation ainsi que les enjeux lies a leur collecte.

1.3. Définition des systèmes de recommandation

Les systèmes de filtrage ou les systèmes de recommandation peuvent être définis de plusieurs façons, vue la diversité des classifications proposées pour ces systèmes, mais il existe une définition générale de Robin Burke [4] qui les définit comme suit :

"*Des systèmes capable de fournir des recommandations personnalisées permettant de guider l'utilisateur vers des ressources intéressantes et utiles au sein d'un espace de données important*".

Un système de filtrage d'information, ou un système de recommandation (recommander system) [5], est un filtre de flux entrant d'information de façon personnalisée pour chaque acteur. Autrement dit, dans un but de personnaliser la recherche d'information dans un domaine d'application particulier, un système de filtrage collecte, sélectionne, classifie et

suggère à l'utilisateur les informations qui répondent vraisemblablement à ses intérêts à long termes.

Les deux entités de base qui apparaissent dans tous les systèmes de recommandations sont l'usager et l'item. L'« usager » est la personne qui utilise un système de recommandation, donne son opinion sur diverses items et reçoit les nouvelles recommandations du système. L'« Item » est le terme général utilise pour designer ce que le système recommande aux usagers.

Les données d'entrée pour un système de recommandation dépendent du type de l'algorithme de filtrage employé. Généralement, elles appartiennent à l'une des catégories suivantes :

- Les estimations :(également appelées les votes), expriment l'opinion des utilisateurs sur les articles (exemple : 1 mauvais a 5 excellent).
- Les données démographiques : se réfèrent a des informations telles que l'âge, le sexe, le pays et l'éducation des utilisateurs. Ce type de données est généralement difficile a obtenir et est normalement collecte explicitement.
- Les données de contenu : qui sont fondées sur une analyse textuelle des documents lies aux éléments évalues par l'utilisateur. Les caractéristiques extraites de cette analyse sont utilisées comme entrées dans l'algorithme de filtrage afin d'en déduire un profil d'utilisateur.

Pour réaliser le filtrage, le système de recommandation (SR) utilise *les profils* représentant des préférences relativement stables des utilisateurs pour calculer des recommandations. Ce calcul se fait par la prédiction des scores qu'un utilisateur est susceptible d'attribuer aux contenus. Le SR adapte ce profil au cours du temps en exploitant au mieux le retour de pertinence que les utilisateurs fournissent sur les informations (documents) reçues. Par exemple, dans la figure 1.1, la fonction de décision du système traite le flux entrant de document pour suggérer à l'utilisateur, en consultant son profil, les documents qu'il préfère. A son tour, l'utilisateur doit fournir des évaluations c'est-à-dire évaluer fréquemment les recommandations, pour que le système comprenne mieux ses besoins en information, et lui fournisse par conséquent de meilleures nouvelles recommandations.

Les trois parties suivantes constituent un système de recommandation :

- **Les producteurs :** Ce sont ceux qui vont permettre de faire les recommandations, ils "fourniront" les données pour.
- **Le module de calcul :** Il s'agit de l'algorithme en lui même. En entrée il y a toutes

les données et la demande et en sortie les différentes recommandations.

- **Le consommateur :** C'est celui qui demande la recommandation.

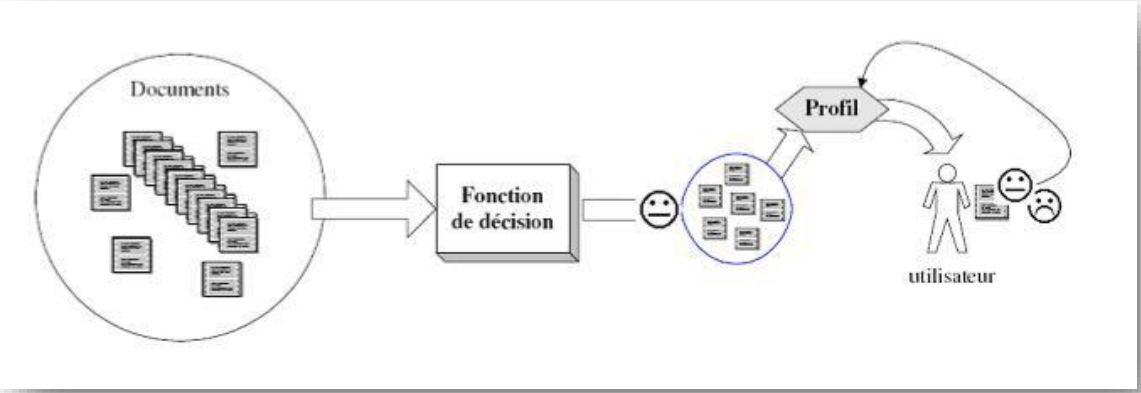


Figure 1.1. Schéma général du filtrage d'information

1.4. Classification des systèmes de recommandation

Il existe plusieurs classifications des systèmes de recommandations (Figure 1.2) :

La classification classique : cette classification de [6] est reconnue par trois types de filtrage ; un filtrage collaboratif(CF), un filtrage base sur le contenu(CBF) et le filtrage hybride.

La Classification de [7] : elle est utilisée dans les systèmes collaboration. Ils proposent une sous-classification qui comprend les techniques hybrides les classer dans les méthodes de collaboration hybrides. [Su et al, 2009] classent filtrage collaboratif en trios categories :

- Approches CF a base de mémoire : pour K-plus proches voisins.
- Approches FC base sur un modèle englobant une variete de techniques telles que: clustering, les réseaux bayesiens, factorisation de matrices, les processus de décision de Markov.
- CF hybride qui combine une technique recommandation CF avec un ou plusieurs autres methodes.

La classification de [8] : c'est une classification en fonction de la source d'information utilisée.

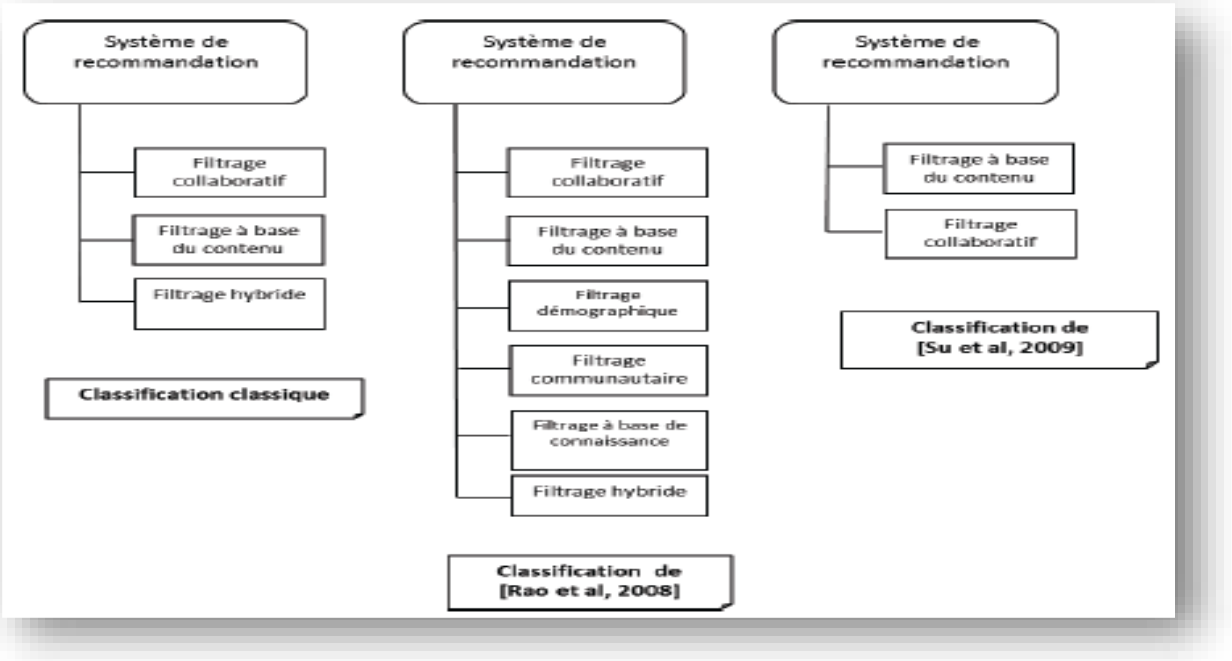


Figure 1.2. Classification principale des systèmes de recommandations.

1.5. Différents types de recommandation

Il existe trois grandes approches de filtrage : basé sur le contenu, collaboratif et hybride. Le filtrage basé sur le contenu compare les nouveaux documents au profil de l'utilisateur, et recommande ceux qui sont les plus proches. Le filtrage collaboratif compare les utilisateurs entre eux sur la base de leurs jugements passés pour créer des communautés, et chaque utilisateur reçoit les documents jugés pertinents par sa communauté. Le filtrage hybride combine le filtrage basé sur le contenu et le filtrage collaboratif pour exploiter au mieux les avantages de chacun.

1.5.1. Recommandation basé sur le contenu

1.5.1.1. Définitions

Le filtrage basé sur le contenu (*Content-based Filtering*) [5], qui est une évolution générale des études sur le filtrage d'information, s'appuie sur le contenu des documents (thèmes abordés) pour les comparer à un profil lui-même constitué de thèmes. Chaque utilisateur du système possède alors un profil qui décrit des centres d'intérêts. Par exemple, le profil peut contenir une liste des thèmes ou préférences que l'utilisateur aime bien ou qu'il n'aime pas. Lors de l'arrivée d'un nouveau document, le système

compare le descriptif du document avec le profil de l'utilisateur pour prédire l'utilité de ce document pour cet utilisateur.

L'avantage des systèmes de filtrage cognitifs, bases contenu est qu'ils permettent d'associer des documents a un profil utilisateur. Notamment, en utilisant des techniques d'indexation et d'intelligence artificielle. L'utilisateur est indépendant des autres ce qui lui permet d'avoir des recommandations même s'il est le seul utilisateur du système. Afin de recommander par exemple des films a un utilisateur, le système analyse les corrélations entre ces films et les films consultés antérieurement par cet utilisateur. Ces corrélations sont évaluées en considérant des attributs comme le titre et le genre. De ce fait, parmi ces films, ceux qui seront recommandes a l'utilisateur, sont les plus similaires (En terme d'attribut) aux films consultes par cet utilisateur.
Cependant, ce type de systèmes présente certaines limitations.

- L'effet "entonnoir" : les besoins de l'utilisateur sont de plus en plus spécifiques, ce qui l'empêche d'avoir une diversité de sujets. Même pire, un nouvel axe de recherche dans un domaine bien précis peut ne pas être pris en compte car il ne fait pas parti du profil explicite de l'utilisateur.
- Filtrage base sur le critère thématique uniquement, absence d'autres facteurs comme la qualité scientifique, le public vise, l'intérêt porte par l'utilisateur, etc.
- Les difficultés a recommander des documents multimédia (images, vidéos, etc.) et ceci a cause de la difficulté a indexer ce type de documents, c'est en fait la même problématique dont souffrent les systèmes de recherche.
- Problème de démarrage a froid : Un nouvel utilisateur du système éprouve des difficultés à exprimer son profil en spécifiant des thèmes qui l'intéressent. Ceci malgré les techniques d'apprentissage ou l'utilisateur fournit des textes exemples.

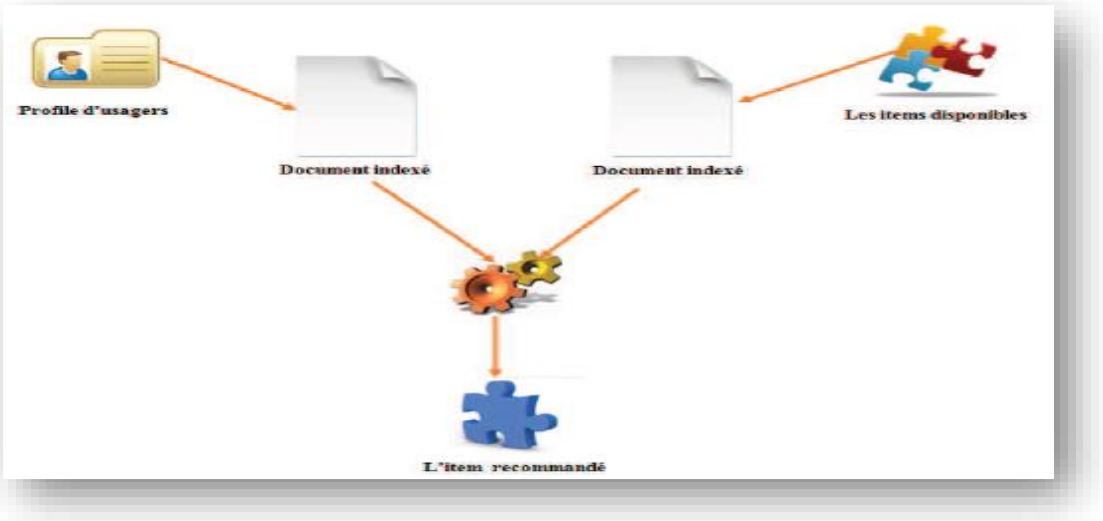


Figure 1.3. Recommandation base sur le contenu.

On distingue deux types de recommandation base sur le contenu : recommandation base sur les mots clefs et recommandation base sur la sémantique.

1.5.1.2. Descripteur d'article et profil utilisateur

L'algorithme de filtrage basé sur le contenu peut réaliser le matching entre un descripteur de contenu (comme par exemple, documents, livre, etc.) et un profile utilisateur et détermine le degré de pertinence de chaque article (ou contenu) pour les utilisateurs potentiels. Si de nombreux articles s'accumulent dans un certain laps de temps, l'algorithme de filtrage de contenu peut ordonner les articles en fonction de leur pertinence pour chacun des utilisateurs potentiels.

Représentation des contenus - le descripteur d'article : Un descripteur d'article se compose d'un ensemble de concept qui peuvent être représentés par une ontologie de domaine. Les concepts qui représentent un élément sont les plus spécialisés dans une branche de la hiérarchie. De toute évidence, un article peut être représenté avec de nombreux concepts de l'ontologie, chaque concept peut apparaître dans n'importe quelle branche de la hiérarchie de l'ontologie et à tout niveau cela dépend du contenu réel de cet article. Il est à noter que le profil peut inclure des concepts frères, c'est-à-dire les fils d'un même concept.

Représentation des utilisateurs - le profil d'utilisateur : Un profil utilisateur basé sur le contenu se compose d'une liste pondérée de concepts de l'ontologie, représentant ses préférences (ses intérêts). De toute évidence, le profil de l'utilisateur peut comporter de nombreux concepts de l'ontologie, chacun figurant dans les différentes branches et différents niveaux de la hiérarchie. Par exemple, le profil de l'utilisateur peut inclure uniquement « sport »,

ou « sport » et « football », ou « football » et « basketball », ou tous les trois - en plus de nombreux autres concepts. Cela signifie qu'un certain concept dans un descripteur d'article peut être comparé avec plus d'un concept équivalent dans le profil de l'utilisateur.

Similarités entre un descripteur d'article et un profil utilisateur : Un descripteur d'article et un profil utilisateur sont semblables à un certain degré si leurs profils comprennent des concepts communs (le même) ou des concepts relatifs, c'est-à-dire des concepts ayant une sorte de relation père- fils. Un descripteur d'article et un profil utilisateur peuvent avoir de nombreux concepts communs ou relatifs; de toute évidence, plus les concepts sont communs ou relatifs, plus forte est leur similitude. Par exemple, si le profil de l'utilisateur inclut « football » et « sport », ce profil est similaire (à un certain degré) à un article qui comprend ces deux concepts, mais il est moins semblable à un article incluant juste « sport », et il est plus semblable à un article, comprenant « sport » et « football ».

1.5.2. Recommandation basé sur le filtrage collaboratif

1.5.2.1. Définition

Le filtrage collaboratif (*Collaborative Filtering* « CF ») a pour principe d'exploiter les évaluations faites par des utilisateurs sur certains documents (contenus), afin de recommander ces mêmes documents à d'autres utilisateurs, et sans qu'il soit nécessaire d'analyser le contenu des documents.

Tous les utilisateurs du système de filtrage collaboratif peuvent tirer profit des évaluations des autres en recevant des recommandations pour lesquelles les utilisateurs les plus proches ont émis un jugement de valeur favorable, et cela sans que le système dispose d'un processus d'extraction du contenu des documents. Grace à son indépendance vis-à-vis de la représentation des données, cette technique peut s'appliquer dans les contextes où le contenu est soit indisponible, soit difficile à analyser, et en particulier elle peut s'utiliser pour tout type de données : texte, image, audio et vidéo.

De plus, l'utilisateur est capable de découvrir divers domaines intéressants, car le principe du filtrage collaboratif ne se fonde absolument pas sur la dimension thématique des profils, et n'est pas soumis à l'effet « entonnoir ».

Un autre avantage du filtrage collaboratif est que les jugements de valeur des utilisateurs intègrent non seulement la dimension thématique mais aussi d'autres facteurs relatifs à la qualité des documents tels que la diversité, la nouveauté, etc.

Le CF souffre de plusieurs gros problèmes. Le problème principal étant le démarrage à froid : c'est le fait qu'un utilisateur doit voter sur beaucoup d'objet avant d'obtenir les recommandations.

1.5.2.2. Processus du filtrage collaboratif

Le processus du filtrage collaboratif suit les étapes données ci-dessous :

1.5.2.2.1. Evaluation des recommandations

Selon le principe de base du filtrage collaboratif, les utilisateurs doivent fournir leurs évaluations sur des documents afin que le système forme les communautés. Evaluer une recommandation peut se faire de façon explicite ou implicite, comme suit **:**

- **Explicite** : L'utilisateur donne une valeur numérique sur une echelle donné (par exemple de 1 à 5, ou de 1 à 10, etc.), ou bien, une valeur qualitative de satisfaction, par exemple, mauvaise, moyenne, bonne et excellente.
- **Implicite** : Le système induit la satisfaction de l'utilisateur à travers ses actions. Par exemple, le système estimera qu'une recommandation supprimée correspond à une évaluation tés mauvaise, alors qu'une recommandation imprimée ou sauvegardée peut être interprétée comme une bonne évaluation.

1.5.2.2.2. Formation des communautés

Le processus de formation des communautés est le noyau d'un système de filtrage collaboratif. Pour chaque utilisateur, le système doit calculer sa communauté, généralement cela se fait par la proximité des évaluations des utilisateurs. Pour ce faire, on peut calculer, dans un premier temps, la proximité entre un utilisateur donné et tous les autres. Ensuite, et afin de créer contrairement la communauté de l'utilisateur, on applique la méthode des voisins les plus proche en utilisant un seuil pour le niveau de proximité ou un seuil pour la taille maximale de la communauté, en raison de sa performance et sa précision.

1.5.2.2.3. Production des recommandations

Dans ce derniers processus, une fois la communauté de l'utilisateur créée, le système prédit l'intérêt qu'un document particulier peut présenter pour l'utilisateur en s'appuyant sur les évaluations que les membres de la communauté ont faites sur ce même document. Lorsque l'intérêt prédit dépasse un certain seuil, le système recommande le document à l'utilisateur.

1.5.2.2.4. Profils et communautés

Ici, nous discutons les profils bases sur l'historique des évaluations des utilisateurs, ainsi que les communautés, qui sont les deux facteurs clés d'SFC. Le problème de la surcharge d'information peut être pallié par la personnalisation de l'accès aux informations, en utilisant des *profils* représentant des intérêts relativement stables des utilisateurs. En d'autres termes les profils des utilisateurs sont utilisés comme des critères persistant dans la recherche d'information

a-) Profil de l'utilisateur :

Le profil utilisateur est composé de prédicats pondérés. Le poids d'un prédicat exprime son intérêt relatif pour l'utilisateur. Il est spécifié par un nombre réel compris entre 0 et 1. Le profil s'enrichit progressivement au fur et à mesure que l'utilisateur évalue des documents reçus. Outre les informations d'identification de base (par exemple, l'identifiant ou des éléments d'état civil), le profil de l'utilisateur peut regrouper des informations très diverses selon les besoins.

Parmi celles-ci, nous pouvons citer :

- Des caractéristiques personnelles pouvant influencer fortement l'interaction (âge, sexe, ….)
- Les intérêts et les préférences générales de l'utilisateur relatives à la tâche à accomplir, qui permettent une adaptation à ses attentes.
- Qualité. Cette dimension contient tous les facteurs reflétant les préférences relatives à la qualité de l'information, comme la disponibilité de données, la concision, le style et la structure du document, etc. Dans cette dimension, nous nous intéressons en particulier à diversité de l'information.
- Sécurité. La dimension de sécurité dans le contexte du filtrage collaboratif, est le niveau de confidentialité concernant tous les autres critères.
- Un historique des interactions avec le service, qui peuvent permettre de modéliser les habitudes comportementales.

b-) Communautés

La notion de communauté dans un système de filtrage collaboratif est définie comme le regroupement des utilisateurs en fonction de l'historique de leurs évaluations, afin que le système calcule des recommandations. Selon cette optique, les profils sont un facteur

interactif, alors que les communautés sont considérées comme un facteur interne du système.

1.5.2.3. Exemple

Dans la figure 1.4, nous schématisons le principe du filtrage collaboratif. On suppose que l'on a des communautés formées par la proximité des évaluations des utilisateurs. Le document *d sera recommandé à l'utilisateur u*, car ce document est apprécié par la communauté G où se l'utilisateur.

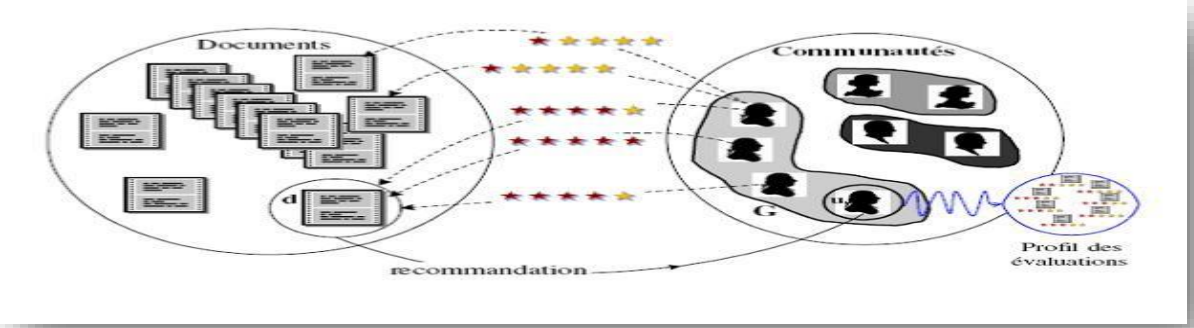


Figure 1.4. Principe général du filtrage collaboratif

1.5.3. Filtrage hybride

Constatant les avantages et inconvénients de chacune des deux approches ci-dessus, on comprend que de nombreux systèmes reposent sur leur combinaison, ce qui en fait des systèmes de filtrage dits « hybrides ». En général, l'hybridation s'effectue en deux phases : (i) appliquer séparément le filtrage collaboratif et autres techniques de filtrage pour générer des recommandations candidates, et (ii) combiner ces ensembles de recommandations préliminaires selon certaines méthodes telles que la pondération, la mixtion, la cascade, la commutation, etc., afin de produire les recommandations finales pour les utilisateurs [9].

Plus généralement, les systèmes hybrides gèrent des profils d'utilisateurs orientes contenu, et la comparaison entre ces profils donne lieu a la formation de communautés d'utilisateurs permettant le filtrage collaboratif. La meilleure description des méthodes hybrides a été faite par [4].Alors, selon Burke on peut distinguer sept façons de combiner les méthodes traditionnelles :

Pondération (Weighted)

Une méthode hybride qui combine la sortie d'approches distinctes, utilisant, par exemple, une combinaison linéaire des scores de chaque technique de recommandation.

Commutation (Switching)

C'est une technique qui permet de faire le choix d'un modèle de recommandation parmi plusieurs, en se basant sur plusieurs critères. La détermination de la technique appropriée dépend de la situation. Le système se doit alors de définir les critères de commutation, ou les cas ou l'utilisation d'une autre technique est recommandée. Ceci permet au système de connaitre les points forts et les points faibles des techniques de recommandation qui le constituent.

Technique mixte (Mixed)

Dans cette approche, le recommandeur ne combine pas, mais augmente la description des ensembles de données, en prenant en considération les estimations des utilisateurs et la description des items. La nouvelle fonction de prédiction doit faire face aux deux types de descriptions et permet d'éviter les problèmes poses par le filtrage collaboratif, a savoir, le démarrage a froid.

Combinaison de caractéristiques (Features combination)

Dans un hybride base sur la combinaison de caractéristiques, les données provenant de techniques collaboratives sont traitées comme une caractéristique, et une approche basée sur le contenu est utilise sur ces données.

Cascade

La cascade implique un processus étape par étape. Dans ce cas, une technique de recommandation est appliquée en premier, produisant un ensemble de candidats potentiels.

Puis, une deuxième technique raffine les résultats obtenus dans la première étape. Cette méthode a pour avantage que si la première technique génère peu de recommandations, ou si ces recommandations sont ordonnées afin de permettre une sélection rapide, la deuxième technique ne sera plus utilisée.

Augmentation de caractéristiques (Feature augmentation)

L'augmentation de caractéristiques est semblable a la cascade, mais dans ce cas-la les résultats obtenus (le classement ou la classification) de la première technique sont utilisées par le deuxième comme une caractéristique ajoutée.

Méta niveau (Meta-level)

Dans un hybride base sur méta niveau, une première technique est utilisée, mais différemment que la précédente méthode (augmentation de caractéristiques), non pas pour produire de nouvelles caractéristiques, mais pour produire un modèle. Et dans la deuxième étape, c'est le

modèle entier qui servira d'entrée pour la deuxième technique [10].

Comme nous l'avons déjà mentionné précédemment, au cœur de la plupart des systèmes de recommandation, nous trouvons un opérateur de matching qui mesure la similarité de deux profils utilisateurs, de deux descripteurs de contenu ou bien la similarité entre un profil utilisateur et un descripteur de contenu. Comme les profils utilisateurs et les descripteurs de contenu sont souvent modélisés avec des vecteurs de mots clés pondérés, seules les mesures vectorielles comme Cosinus et corrélation de Pearson sont utilisées. Or, l'avènement du web sémantique et le développement des ontologies ont mis à notre disposition une panoplie de mesures de similarité sémantiques qui peuvent compléter les mesures vectorielles. Nous avons jugé important de présenter quelques mesures de similarité des plus connues. La section suivante montre une classification de ces mesures.

1.6. Classification des approches de mesure de similarité

Dans cette classification, nous distinguons quatre grandes catégories de mesures de similarité.

1.6.1. Approches basées sur l'espace vectoriel

Dans le domaine de la recherche de l'information, les modèles de l'espace vectoriel sont largement adoptés, on parlera alors de similarité numérique. Ces approches [11] utilisent un vecteur caractéristique, dans un espace dimensionnel, pour représenter chaque objet et calculent la similarité numérique en se basant sur la mesure de cosine ou la corrélation de Pearson. Parmi les approches citées dans la littérature on peut citer :

1.6.4.1. Similarité de Cosine

Cette mesure utilise la représentation vectorielle complète, c'est-à-dire la fréquence des objets (mots). Deux objets (documents) sont similaires si leurs vecteurs sont confondus. Si deux objets ne sont pas similaires, leurs vecteurs forment un angle (X, Y) dont le cosinus représente la valeur de la similarité. La formule est définie par le rapport du produit scalaire des vecteurs x et y et le produit de la norme de x et de y.

$$Sim\,(X,Y) = \cos\,(X,Y) = \frac{X * Y}{\|X\|_2 * \|Y\|_2}$$

La mesure de Cosine [11] quantifie donc la similarité numérique entre les deux vecteurs, comme le cosinus de l'angle entre les deux vecteurs.

1.6.4.2. Similarité de Pearson

La mesure de similarité de Pearson est basée sur le calcul de la corrélation. Pour connaître le coefficient de corrélation liant deux séries X(x1, x2,…,xn) et Y(y1, y2, …,yn), on applique la formule suivante :

$$\text{Sim (x,y)} = r_p = \frac{\sum_{i=1}^{N}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

Si r vaut 0, les deux courbes ne sont pas corrélées et donc ne sont pas similaire. Les deux courbes sont d'autant mieux corrélées que r est loin de 0 (proche de -1 ou 1). Avec:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$$ est la moyenne de X

$$\bar{y} = \frac{1}{N}\sum_{i=1}^{N} y_i$$ est la moyenne de Y

1.6.2. Approches basées sur les arcs

La mesure de similarité la plus intuitive des objets dans une ontologie est leurs distances. Cette similarité est évaluée par la distance qui sépare les objets de l'ontologie. Ces mesures servent de la structure hiérarchique de l'ontologie pour déterminer la similarité sémantique entre les concepts. Le calcul des distances dans l'ontologie est basé sur un graphe de spécialisation des objets.

Parmi les travaux classifiés sous cette catégorie on peut citer :

1.6.2.1. Mesure de Wu & Palmer (1994)

La mesure de similarité de Wu et Palmer [12] est basée sur le principe suivant : Etant donnée une ontologie formée d'un ensemble de nœud et un nœud racine R (Figure 1.5). Soit X et Y deux éléments de l'ontologie dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur les distances (N1 et N2) qui séparent les nœuds X et Y du nœud racine et la distance qui sépare le concept subsumant (CS) de X et de Y du nœud R.

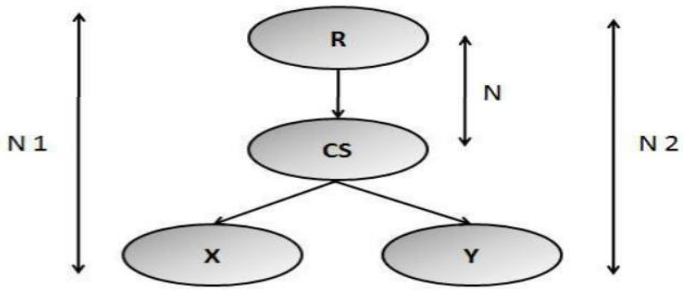


Figure 1.5. Exemple d'un extrait d'une ontologie

La mesure de Wu et Palmer est définie par la formule suivante :

$$\boldsymbol{Sim\,(X,Y) = \frac{2 * N}{N1 + N2}}$$

1.6.1.2. Mesure de Rada et al (1989)

Cette mesure est adoptée dans un réseau sémantique et elle est fondée sur le fait qu'on peut calculer la similarité en se basant sur les liens hiérarchiques «is-a». Pour calculer la similarité de concepts dans une ontologie, on doit calculer le nombre des arcs minimums qui les séparent. Cette mesure [11], est basée sur le calcul de distance entre les nœuds par le chemin le plus court, présente un moyen des plus évidents pour évaluer la similarité sémantique dans une ontologie hiérarchique. Il présente ainsi une mesure utilisant une métrique, distance(c1, c2), qui indique le nombre d'arcs minimum à parcourir pour aller d'un concept c1 à un concept c2.

$$\boldsymbol{Sim(c1, c2) = \frac{1}{1 + distance(c1, c2)}}$$

1.6.3. Approches basées sur les nœuds

Ces approches adoptent une nouvelle mesure en termes de la mesure entropique (Contenu informationnel) de la théorie de l'information. La probabilité P pour l'identification de l'utilisateur d'une classe ou de ses descendants dans un corpus désigne l'information de la classe.

On définit l'entropie d'une classe par la formule suivante : $\boldsymbol{E(c) = -log(Pc))}$
Où P est la probabilité de trouver une instance du concept c. La probabilité d'un concept c est calculée en divisant le nombre des instances de c par nombre total des instances.

En associant les probabilités aux concepts d'ne taxonomie, il est possible d'éviter le manque de fiabilité des distances des arcs. Parmi les travaux, recensés dans la littérature, sous cette catégorie on peut citer :

1.6.3.1. Resnik (1999)

Resnik [13] définit la similarité sémantique entre deux concepts par la quantité d'information qu'ils partagent. Cette information partagée est égale au contenu informationnel (CI) du plus petit généralisant (PPG) (Plus Petit Généralisant est le concept le plus spécifique qui subsume les deux concepts dans l'ontologie). Par exemple, Dans la figure 1.5, le PPG des concepts X et Y est le concept CS.

$$Sim\ (c1, c2) = Max\ [CI\ (PPG\ (c1, c2))] \qquad \text{Où} \qquad CI = -log\ (P\ (c))$$

1.6.3.2. Mesure de Lin (1998)

Lin a défini la similarité de concepts comme le rapport entre la quantité d'informations nécessaires pour indiquer le point commun entre ces deux concepts et les informations nécessaires pour les décrire [14]. Cette mesure est légèrement différente de celle de Resnik :

$$Sim\ (a, b) = \frac{2 * CI\ (PPG(a, b))}{CI\ (a) + CI\ (b)}$$

1.6.2.3. Mesure de Lin (1998)

Lin a défini la similarité de concepts comme le rapport entre la quantité d'informations nécessaires pour indiquer le point commun entre ces deux concepts et les informations nécessaires pour les décrire [14]. Cette mesure est légèrement différente de celle de Resnik :

$$Sim\ (a, b) = \frac{2 * CI\ (PPG(a, b))}{CI\ (a) + CI\ (b)}$$

1.6.4. Approches hybrides

Ces approches sont fondées sur un modèle qui combine entre les approches basées sur les arcs (Distances) en plus du contenu informationnel qui est considéré comme facteur de décision.

1.6.4.1. Mesure de Jiang et Conrath (1997)

Pour remédier au problème présenté au niveau de la mesure de Resnik, Jiang et Conrath [15]

ont apporté une nouvelle formule qui consiste à combiner l'entropie (contenu informationnel) du concept spécifique à ceux des concepts dont on cherche la similarité (combine entre les techniques basées sur les arcs et les techniques basées sur les nœuds qui consistent à compter les arcs afin d'améliorer les résultats par les calculs basés sur les nœuds. Notons que cette formule es définie par l'inverse de la distance sémantique.

$$Sim\,(c1, c2) = \frac{1}{distance(c1, c2)}$$

Sachant que la distance entre c1 et c2 est calculée par la formule suivante :

$$Distance\ (c1, c2) = CI\ (c1) + CI\ (c2) - (2.CI\ (PPG\ (c1, c2)))$$

1.6.4.2. Mesure de Leacock et Chodorow (1998)

Une autre méthode présentée combine la méthode de comptage des arcs et la méthode du contenu informationnel [11]. La mesure proposée par Leacock et Chodorow est basée sur la longueur du plus court chemin entre deux synsets de Wordnet. Les auteurs ont limité leur attention à des liens hiérarchiques «is-a» ainsi que la longueur de chemin par la profondeur globale de la taxonomie. La formule est définie par :

$$Sim\,(X, Y) = -Log\left(\frac{CD(X, Y)}{2 * M}\right)$$

Où M est la longueur du chemin le plus long qui sépare le concept racine, de l'ontologie, du concept le plus en bas (la profondeur globale). On dénote par CD(X, Y) la longueur du chemin le plus court qui sépare X de Y.

1.7. Avantages et inconvénients des systèmes de recommandation

Le tableau 1.1 résume les forces et faiblesses des méthodes traditionnelles utilise par les systèmes de recommandation, en l'occurrence le Filtrage Collaboratif (FC), le Filtrage Démographique (FD), le Filtrage à Base de Contenu (FBC), et le Filtrage à base de données communautaires.

Adaptabilité : Au fur et à mesure que la base de données des évaluations augmente, la recommandation devient plus précise.

Nouvel utilisateur : un nouvel utilisateur qui n'a pas encore accumule suffisamment d'évaluations ne peut pas avoir de recommandations pertinentes.

Nouvel item : un item doit avoir suffisamment d'évaluations pour qu'il soit pris en considération dans le processus de recommandation.

Démarrage à froid : le démarrage a froid est un problème pour les nouveaux utilisateurs qui commencent a jouer avec le système, parce que le système ne dispose pas d'assez d'informations a leur sujet. Si le profil d'utilisateur est vide, il doit consacrer une somme d'efforts à l'aide du système avant d'obtenir une récompense (les recommandations utiles). D'autre part, quand un nouvel item est ajoute a la collection, le système doit avoir suffisamment d'informations pour être en mesure de recommander cet item aux utilisateurs.

Techniques	Avantages	Inconvénients
Filtrage démographique	N'exige aucun historique d'estimations.	• Problème de confidentialité. • Utilisateur avec un goût unique. • Nouvel Item.
Filtrage à base de données communautaire	Adaptabilité : la qualité croit avec le nombre d'amis.	• Nouvel utilisateur. • Nouvel item.
Filtrage à base du contenu	• Pas besoin d'une large communauté d'utilisateurs pour pouvoir effectuer des recommandations. • Une liste de recommandations peut être générée même s'il n'y a qu'un seul utilisateur. • La qualité croit avec le temps. • Pas besoin d'information sur les autres utilisateurs. • Prendre en considération les gouts uniques[37] des utilisateurs.	• L'analyse du contenu est nécessaire pour faire une recommandation. • Problème de recommandation des images et de vidéos en absence de Métadonnées. • Nécessité du profil d'utilisateur.
Filtrage collaboratif	• Ne demande aucune connaissance sur le contenu de l'item ni sa sémantique. • La qualité de la recommandation peut être évaluée. • Plus les nombre d'utilisateurs est grand plus la recommandation est meilleure.	• Démarrage à froid. • Nouvel Item. • Nouvel utilisateur. • Problème de confidentialité. • La complexité : dans les systèmes avec un grand nombre d'items et d'utilisateurs, le calcul croit linéairement.

Table 1.1 – Les avantages et les inconvénients des techniques de recommandations.

1.8. Conclusion

Dans ce chapitre, nous avons d'abord, présenté la notion des systèmes de recommandation, en détaillons les trois approches les plus utilisées, à savoir, l'approche FNC, FC et hybride. Ensuite, nous avons défini la notion de profil utilisateur. Nous avons également passé en revue les différentes classes de mesures de similarité utilisées par les SRs pour faire le matching entre deux profils utilisateur, deux contenus, ou un profil utilisateur et un descripteur de contenu. Enfin, nous avons terminé en citant quelques problèmes rencontrés par les systèmes de recommandation classiques.

Chapitre 2

Ingénierie Ontologique

2.1. Introduction

Durant cette dernière décennie, nous avons remarqué qu'une attention croissante a été concentrée sur l'ingénierie ontologique où l'ontologie est l'objet fondamental sur lequel il faut se pencher. Les ontologies sont largement utilisées et ont prouvé leurs utilités dans de nombreux domaines tels que : l'ingénierie de connaissances, l'intelligence artificielle, la recherche d'information, l'e-commerce et sont au cœur du web sémantique. Cet engouement est motivé par le fait que les ontologies sont un moyen efficace pour la gestion et le partage des connaissances d'un domaine particulier entre personnes et\ou systèmes.

Dans ce chapitre, nous allons présenter en premier lieu la notion d'ontologie et les éléments qui la constituent. Par la suite, nous présenterons ses différentes classifications en se concentrant sur l'objet de conceptualisation et le degré de formalisation. En outre, nous faisons un survol des principaux formalismes de représentation de connaissances à savoir les frames, les graphes conceptuels et les logiques de descriptions. De plus, nous donnons un bref aperçu de quelques outils de développement d'ontologies.

2.2. La notion d'ontologie

Le terme «ontologie» est cependant usité en philosophie depuis le XIX^{ème} siècle. Dans ce domaine, l'ontologie est une étude de l'être en tant qu'être, c'est-à-dire, une étude des propriétés générales de ce qui existe. En informatique, la littérature fournit un tas de définitions du mot ontologie. Ces définitions, dans leur diversité, offrent des points de vues à la fois différents et complémentaires. Dans la section suivante, nous décrivons quelques unes.

2.2.1. Définitions

– Neches et ses collègues [16] ont été les premiers à en proposer une définition :

> *«Une ontologie définit les termes et les relations de base du vocabulaire d'un domaine ainsi que les règles qui indiquent comment combiner les termes et les relations de façon à pouvoir étendre le vocabulaire».*

Cette définition descriptive donne un premier aperçu sur la manière de construire une ontologie, à savoir l'identification des termes de bases d'un domaine et les relations entre ces termes ainsi que les règles pouvant s'appliquer sur ces derniers.

– En 1993, Gruber [17] a proposé une définition à cette notion qui est la plus célèbre couramment citée dans la littérature :

«Une ontologie est une spécification explicite d'une conceptualisation».

Il a introduit la notion de 'conceptualisation' qui réfère à un modèle abstrait d'un certain domaine du monde réel en identifiant les concepts pertinents décrivant ce domaine. Le terme 'explicite' signifie que les concepts utilisés ainsi que les contraintes sur leur emploi, sont réellement définis d'une manière claire et précise.

– En 1997, Borst [18] a modifié légèrement la définition de Gruber en citant qu'une ontologie est définie comme étant :

«Une ontologie est une spécification formelle d'une conceptualisation partagée».

Cette définition précise d'une part, le fait que l'ontologie doit être 'formelle', c'est-à-dire exprimée sous forme d'une logique pouvant être exploitable par une machine. D'autre part, elle doit être 'partagée » dans la mesure où elle doit capturer des connaissances partagées entre différents individus.

– En 1998, Studer et ses collègues [19] ont rassemblé ces deux définitions (celles de Gruber et Borst) dans une seule qui est :

«Une ontologie est une spécification formelle et explicite d'une conceptualisation partagée».

Ils l'expliquent comme suit :

- Spécification explicite signifie que les concepts, les propriétés, les relations, les fonctions, les restrictions et les axiomes de l'ontologie sont définies de façon déclarative ;
- Formelle réfère au fait qu'une ontologie doit être traduite dans un langage interprétable par une machine (machine-readable) ;
- Conceptualisation réfère à un modèle abstrait d'un phénomène du monde en identifiant les concepts appropriés à ce domaine ;
- Partagé réfère au fait qu'une ontologie capture la connaissance consensuelle c'est-à-dire non réservée à quelque individus, mais partagée par un groupe ou une communauté.

– En 1997 Guarino [20] énonce que :

> *«Les ontologies sont des spécifications partielles et formelles d'une conceptualisation commune».*

En 1997, Guarino accentue l'ambigüité du terme conceptualisation qui doit être pris dans son sens intuitif. La spécification des ontologies est partielle, car une conceptualisation ne peut pas toujours être entièrement formalisée dans un cadre logique, du fait d'ambigüités ou du fait qu'aucune représentation de leur sémantique n'existe dans le langage de représentation d'ontologies choisi. «Commune» renvoie à l'idée qu'une ontologie rend compte d'un savoir consensuel, c'est-à-dire qu'elle n'est pas l'objet d'un individu, mais qu'elle est reconnue par un groupe.

Pour conclure cette section, nous pouvons donc affirmer que les définitions du terme ontologie abondent dans la littérature scientifique. Les définitions, dans leur diversité, offrent des points de vues à la fois différents et complémentaires sur un même concept.

2.2.2. Les composantes de l'ontologie

Les ontologies produisent un vocabulaire commun d'un domaine et définissent, de façon plus ou moins formelle, la signification des termes et des relations entre eux. Les connaissances intégrées dans les ontologies sont formalisées en mettant en jeu cinq types de composants : concepts, relations, fonctions, axiomes, instances.

2.2.2.1. Concepts

Ils sont appelés aussi termes ou classes de l'ontologie. Un concept est un constituant de la pensée (un principe, une idée, une notion abstraite) sémantiquement évaluable et communicable. Un concept peut être divisé en trois parties : un terme(ou plusieurs), une notion et un ensemble d'objets.

- Le terme (ou bien label) d'un concept est l'expression linguistique utilisée couramment pour y faire référence.
- La notion désigne ce qui est appelé, au sens de la représentation des connaissances, *l'intension* du concept. Elle contient sa sémantique qui est définie à l'aide de propriétés (relations et attributs), de règles et de contraintes.
- L'ensemble d'objets définis par le concept forme ce qui est appelé *l'extension* du concept. Il s'agit des objets auxquels le concept fait référence, autrement dit, de ses instances.

Selon [Gomez-Pérez ,99], ces concepts peuvent être classifiés selon plusieurs dimensions :

- Niveau d'abstraction (concret ou abstrait).
- Atomicité (élémentaire ou composée).
- Niveau de réalité (réel ou fictif).

En résumé, un concept peut être tout ce qui peut être évoqué et, partant, peut consister en la description d'une tache, d'une fonction, d'une action, d'une stratégie ou d'un processus de raisonnement, etc.

2.2.2.2. Relations

Représentent un type d'interaction, ou bien des associations existant entre les concepts d'un domaine. Formellement, elles sont définies comme étant tout sous ensemble d'un produit de n concepts, c'est-à-dire : $R \subset C1 * C2 * ... Cn$.

Sous-classe-de (Spécialisation, généralisation), partie de (agrégation ou composition), associée-à, instance-de sont des exemples de relations binaires.

Voici quelques relations les plus courantes dans la littérature :

1) *L'équivalence* : une relation R est une relation d'équivalence si et seulement si : R est symétrique, réflexive et transitive. On écrit :

(R est une relation d'équivalence) $\Longleftrightarrow$ *((R symétrique) ^ (R réflexive) ^ (R transitive))*

2) *La cardinalité* : c'est le nombre possible de relations de ce type entre les mêmes concepts (ou instances de concept). Les relations portant une cardinalité représentent souvent des attributs. Exemple : une pièce a au moins une porte, un humain a entre zéro et deux jambes.
3) *L'incompatibilité* : Deux relations sont incompatibles si elles ne peuvent lier les mêmes instances de concepts. Exemples : les relations «être rouge» et «être vert» sont incompatibles.
4) *L'inverse* : Deux relations binaires sont inverses l'une de l'autre si, quand l'une lie deux instances I1 et I2, l'autre lie I2 et I1. Exemple : les relations «a pour père »et «a pour enfant» sont inverses l'une de l'autre.
5) *L'exclusivité* : Deux relations sont exclusives si, quand l'une lie des instances de concepts, l'autre ne lie pas ces instances, et vice-versa. L'exclusivité entraine l'incompatibilité. Exemple : l'appartenance et la non appartenance sont exclusives.

Et bien d'autres relations…

2.2.2.3. Fonctions

Ce sont des cas particuliers de relations dans lesquelles le N^{ème} élément de la relation est défini de manière unique à partir des n-1 premiers. Formellement, les fonctions sont définies ainsi :

$F : C_1 \times C_2 \times \ldots \times C_{n-1} \longrightarrow C_n$

Comme exemple de fonctions binaires, nous avons les fonctions mère de et le carré.

2.2.2.4. Axiomes

Constituent des assertions, acceptées comme vraies, à propos des abstractions du domaine, traduites par l'ontologie. Ils ont pour objectif de représenter des concepts et des relations dans un langage logique permettant de représenter leur sémantique. Ils représentent les intentions des concepts et des relations du domaine et, de manière générale, les connaissances n'ayant pas un caractère strictement terminologique. L'utilisation des axiomes sert à définir le sens des entités, mettre des restrictions sur la valeur des attributs, examiner la conformité des informations spécifiées ou en déduire de nouvelles.

2.2.5. Instances (ou individu)

Elles constituent la définition extensionnelle de l'ontologie. Ils représentent les éléments singuliers véhiculant les connaissances à propos du domaine.

2.3. Classification des ontologies :

Cette section présente les types les plus généralement utilisés d'ontologie. On peut avoir une idée de la connaissance qui est incluse dans chaque type d'ontologie. Les ontologies peuvent être classifiées selon plusieurs dimensions. Parmi celles-ci, nous en examinerons deux :

1) L'objet de conceptualisation ;

2) Le niveau de formalisme de représentation.

2.3.1. L'objet de conceptualisation :

Gomez et ses collègues (Gomez-Pérez, et al, 2004) proposent une classification selon l'objet de conceptualisation des ontologies (le but de leur utilisation) de la façon suivante :

- Ontologie de haut niveau/ de niveau supérieur ;
- Ontologie du domaine ;
- Ontologie de tache ;
- Ontologie d'application.

a) ***Ontologie de haut niveau (top-level or upper-level ontology)*** : décrit des concepts très généraux comme l'espace, le temps, la matière, les objets, les événements, les actions, etc. Ces concepts ne dépendent pas d'un problème ou d'un domaine particulier, et doivent être, du moins en théorie, consensuels à de grandes communautés d'utilisateurs. Des exemples d'ontologies de haut niveau sont Dolce ou Sumo.

b) ***Ontologie de domaine (Domain Ontology)*** : Contrairement aux ontologies de haut niveau, les ontologies de domaine sont plus spécifiques. Elles synthétisent les connaissances spécifiques à un domaine particulier. Elles décrivent le vocabulaire ayant trait à un domaine générique (ex : l'enseignement, la médecine…), notamment en spécialisant les concepts d'une ontologie de haut niveau.

c) ***Ontologie de tache (Task Ontology)*** : Ce type d'ontologie est utilisé pour décrire un vocabulaire relatif à une tache ou une activité générique (faire un diagnostic, planifier une activité…) en spécialisant certains termes des ontologies de haut niveau. Ces ontologies fournissent un ensemble de termes au moyen desquels on peut décrire, au niveau générique, comment résoudre un type de problème.

d) ***Ontologie d'application (Application ontology)*** : Cette ontologie est la plus spécifique, elle contient des concepts dépendants d'un domaine et d'une tache particuliers, qui sont généralement subsumés par des concepts de ces deux ontologies. Ces concepts correspondent souvent aux rôles joués par les entités du domaine lors de l'exécution d'une certaine activité.

2.3.2. Le niveau de formalisme de représentation

Les ontologies peuvent être distinguées en fonction du degré de formalisme utilisé pour les exprimer. Uschold et Grüninger [21] proposent une classification contenant les quatre catégories :

1. ***Informelles*** : ontologie opérationnelles dans un langage naturel (sémantique ouverte).
2. ***Semi-informelles*** : utilisation d'un langage naturel structuré et limité.
3. ***Semi-formelles*** : langage artificiel défini formellement.
4. ***Formelles*** : l'ontologie est exprimée dans un langage artificiel contenant une sémantique formelle, des théorèmes, et des preuves pour vérifier les propriétés telles que la validité et la complétude.

2.4. Les ontologies et le Web Sémantique :

Le Web actuel est essentiellement syntaxique, la structure des ressources étant bien définie, mais leur contenu restant inaccessible aux traitements machines, seuls les humains étant capables de l'interpréter.

Le Web sémantique a alors l'ambition de lever cette difficulté en associant aux ressources du Web des entités ontologiques comme références sémantiques, ce qui permettra aux différents agents logiciels d'accéder et d'exploiter directement le contenu des ressources et de raisonner dessus. Ce référencement sémantique peut aussi résoudre les problèmes d'interprétation des ressources informationnelles provenant des applications hétérogènes et réparties et de permettre ainsi à ces applications d'être intégrées sémantiquement.

2.5. Cycle de vie d'une ontologie

Puisque les ontologies sont destinées à être utilisées comme des composants logiciels dans des systèmes répondant à des objectifs opérationnels différents, leur développement doit s'appuyer sur les mêmes principes que ceux appliqués en génie logiciel. Ainsi, les ontologies doivent être considérées comme des objets techniques évolutifs et possédants un cycle de vie qui nécessite d'être précisé. Dans ce contexte, les activités liées aux ontologies sont, d'une part, des activités de gestion de projet (planification, contrôle, assurance qualité), et d'autre part, des activités de développement (spécification, conceptualisation, formalisation) ; s'y ajoutent des activités transversales de support telles que l'évaluation, la documentation, la gestion de la configuration. Le cycle de vie d'une ontologie comprend une étape initiale de détection et de spécification des besoins qui permet notamment de cerner précisément le domaine de connaissances, une étape de conception qui se subdivise en trois phases, une étape de déploiement et de diffusion, une étape d'utilisation, une étape incontournable, d'évaluation, et enfin, une sixième étape consacrée à l'évolution et à la maintenance du modèle. Après chaque utilisation significative, l'ontologie et les besoins doivent être réévalués et l'ontologie peut être étendue et, si nécessaire, en partie reconstruite. La validation du modèle de connaissances est au centre du processus et se fait de manière itérative. Le processus de construction peut etre intégré au cycle de vie d'une ontologie comme l'indique la Figure 2.1.

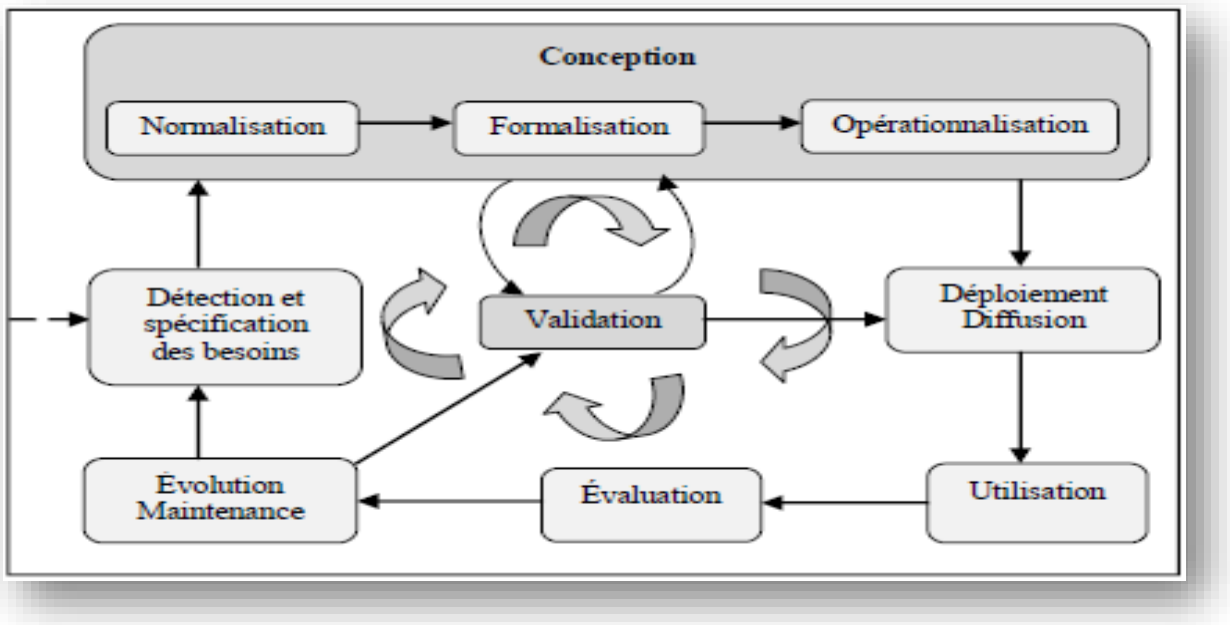


Figure 2.1 : cycle de vie d'une ontologie.

2.6. Différents besoins des ontologies

Les ontologies sont utilisées dans plusieurs domaines, les plus répandus sont :

- Communication.
- Interopérabilité entre les systèmes.
- Ingénierie des systèmes.
- Recherche d'information

2.6.1. Communication

Les humains peuvent communiquer efficacement s'ils ont des connaissances ou des points de vue partagés. Ces connaissances partagées peuvent être obtenues si le domaine est explicitement décrit sans confusion terminologique ou conceptuelle pour être compris de la même façon par tout le monde.

Une ontologie facilite la communication en fournissant une spécification explicite d'un domaine qui représente un modèle normatif. De plus, les ontologies permettent d'assurer la consistante et d'enlever l'ambigüité dans les descriptions des connaissances concernant un domaine spécifique. Finalement, les ontologies peuvent intégrer différents perspectives des utilisateurs.

Quand les utilisateurs (qui ont différentes perspectives d'un domaine) partagent une ontologie, ils ont une perspective standard.

2.6.2. Interopérabilité entre les systèmes

L'interopérabilité implique la possibilité de pouvoir demander et recevoir des services entre des systèmes interopérables. Deux systèmes sont considérés interopérables s'ils vérifient les deux conditions suivantes :

- ✓ Ils opèrent comme une unité afin de réaliser une tache commune.
- ✓ Ils peuvent échanger des messages et des requêtes.

Les ontologies permettent de faciliter l'interopérabilité en intégrant les connaissances concernant différents domaines dont l'objectif est de décrire un domaine unifié ou accomplir une tache commune. Elles permettent aussi d'intégrer les différents vocabulaires concernant certains domaines. Pour ce faire, les ontologies de ces domaines doivent être intégrées par les méthodes d'intégration d'ontologies afin de partager un même vocabulaire.

2.6.3. Ingénierie des systèmes

Le développement des systèmes basé sur les ontologies a donné un profit à l'ingénierie de systèmes qui peut être résumé comme suit :

- Réutilisabilité : l'ontologie encode les informations relatives à un domaine (y compris les composants logiciels) de sorte que le partage et la réutilisation sont possibles.
- Acquisition des connaissances : l'ontologie guide l'acquisition des connaissances.
- Sureté : l'ontologie rend possible l'automatisation du processus de vérification de consistance.
- Spécification : l'ontologie aide le processus d'identification des besoins et la définition des spécifications des systèmes.

2.7. Un Squelette de méthodologie pour construire des ontologies

Le processus de construction d'une ontologie est une collaboration qui réunit des experts du domaine de connaissance des ingénieurs de la connaissance voire les futurs utilisateurs de l'ontologie. Cette collaboration ne peut être fructueuse que si les objectifs du processus ont été clairement définis ainsi que les besoins qui en découlent. La figure ci-dessous représente le processus de construction d'ontologie.

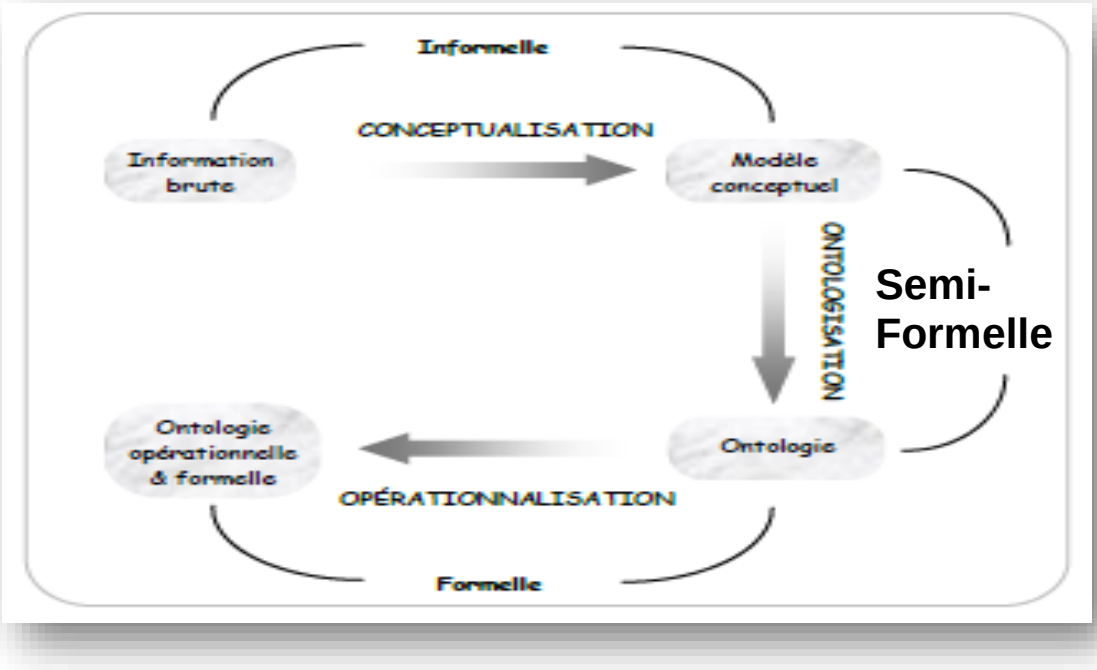


Figure 2.2 : Processus de construction d'ontologies

2.7.1 Evaluation des besoins

Le but visé par la construction d'une ontologie se décline en 3 aspects :

- L'objectif opérationnel : il est indispensable de bien préciser l'objectif opérationnel de l'ontologie en particulier à travers des scénarios d'usage.
- L e domaine de connaissance : il doit être délimité aussi précisément que possible.
- Les utilisateurs : ils doivent être identifiés autant que faire se peut ce qui permet de choisir en accord avec l'objectif opérationnel, le degré de formalisme de l'ontologie et sa granularité.

Une fois le but défini, le processus de construction de l'ontologie peut démarrer, en commençant par la phase de conceptualisation.

2.7.2. Conceptualisation

Cette étape permet d'aboutir à un modèle informel, donc sémantiquement ambiguë et généralement exprimé en langage naturel. Elle consiste, à partir des données brutes à dégager les concepts et les relations entre ces concepts permettant de décrire de manière informelle les entités cognitives du domaine.

L'objectif est d'aboutir à un modèle conceptuel : ce modèle consiste en un ensemble de termes désignant les entités du domaine de connaissances (concept, relation, propriétés des

concepts et des relations¸…) ¸assortis d'informations exprimant leur sémantique. La découverte des connaissances d'un domaine peut s'appuyer a la fois sur l'analyse de documents et sur l'interview d'experts du domaine. Ces activités doivent être raffinées au fur et à mesure que la conceptualisation émerge.

2.7.3. Ontologisation

L'Ontologisation consiste en une formalisation partielle, sans perte d'information, du modèle conceptuel obtenu dans l'étape précédente. Ce qui permet de faciliter sa représentation ultérieure dans un langage complètement formel et opérationnel.

Elle effectue une transcription des connaissances dans un certain formalisme de connaissances, ce formalisme devant être aussi générique que possible mais sémantiquement clair.

Le modèle obtenu est souvent qualifie de semi-formel (car certaines connaissances ne peuvent pas être totalement formalisées).Le caractère semi-formel d'une ontologie lui interdit d'être utilisée telle quelle dans un SBC. En revanche¸ une ontologie¸ contenant toutes les connaissances d'un domaine constitue le support idéal de communication et de partage des connaissances de ce domaine.

2.7.4. Opérationnalisation

Cette étape consiste a formaliser complètement l'ontologie obtenue dans un langage représentation de connaissance formel (i.e. possédant une syntaxe et une sémantique) et opérationnel (i.e. dote de services différentiels permettant de mettre en œuvre des raisonnements) ¸par exemple, le modèle des graphes Conceptuels ou la logique de Descriptions.

On obtient alors une représentation formelle des connaissances du domaine. Ainsi, le caractère formel de l'ontologie permet à une machine, via cette ontologie, de manipuler des connaissances du domaine. La machine doit donc pouvoir utiliser des mécanismes opérant sur les représentations de l'ontologie.

2.8. Quelques méthodologies de construction ontologies

Les méthodologies peuvent porter sur l'ensemble du processus et guider l'otologiste dans toutes les étapes de la construction. Bien qu'aucune méthodologie générale n'ait pour l'instant réussi à s'imposer de nombreux critères de construction d'ontologies ont été

proposés pour des méthodologies. ENTERPRISE¸TOVE et METHONTOLOGY sont les méthodologies les plus représentatives pour construire des ontologies.

2.8.1. TOVE

TOVE (Toronto Virtual Enterprise) développé par l'université de Toronto, cette méthodologie repose sur les expériences de développement d'une entreprise. Elle s'appuie également pour le développement d'une ontologie sur les principales étapes suivantes :

- Capturer des scénarios de motivations : Cette étape consiste à identifier des scénarios qui clarifient le domaine que l'on investit et les différentes applications dans lesquelles l'ontologie sera employée.
- Formuler des questions de compétences informelles : Cette étape consiste à formuler un ensemble de questions (basées sur les scénarios) ¸exprimées en langage naturel, afin de déterminer la portée de l'ontologie. Ces questions et leurs réponses sont utilisées pour extraire les concepts principaux, leurs propriétés et les relations qui existent entre ces concepts.
- Spécifier la terminologie de l'ontologie : Cette étape consiste à représenter les termes (concepts propriétés et relations) ¸identifier dans l'étape précédente en utilisant le formalisme de la logique du premier ordre. Les concepts seront représentés sous forme de constantes ou bien des variables. Par ailleurs, les propriétés et les relations seront représentées par des prédicats.
- Evaluer la complétude de l'ontologie.

2.8.2. ENTERPRISE

Uschold [21 ¸propose le squelette d'une méthode basé sur l'expérience de construction d'ontologie dans le domaine de la gestion des entreprises. La méthode ENTERPRISE repose sur les quatre étapes suivantes :

- Identifier le rôle et la portée de l'ontologie.
- Identifier les concepts et relations fondamentaux et des définitions provisoires de ces éléments. Coder l'ontologie dans un langage adapté.

Intégrer des ontologies existantes. Dans cette étape¸ l'ontologie est réellement construite.

- Evaluer l'ontologie
- Rédiger une documentation et une trace des actions réalisées lors des différentes phases.

Les étapes et sous-taches de la méthode ENTERPRISE sont décrites de façon abstraite. Les techniques utilisées pour les sous –taches ne sont pas précisées (par exemple : comment identifier les concepts fondamentaux? Quel langage utiliser pour représenter l'ontologie?

2.8.3. METHONTOLOGY

La méthodologie de construction d'ontologies « METHONTOLOGY » se situe entre le GL (Génie Logiciel) et l'IC (Ingénierie des connaissances). Elle identifie une séquence d'activités techniques à appliquer pour le développement de l'ontologie .L'approche METHONTOLOGY distingue les étapes suivantes :

2.8.3.1. Spécification

Le développement d'une ontologie commence par la définition du domaine et portée de celle-ci. Cela est basé sur la réponse à certaines questions : Quel est le domaine que l'ontologie va couvrir? A quoi cette ontologie va servir ? A quels types de questions les informations de l'ontologie doivent fournir des réponses? Qui va utiliser et maintenir l'ontologie ?¸etc. Les réponses à ces questions peuvent changer durant le processus de développement de l'ontologie, mais a chaque étape ¸ elles permettant de limiter la portée du modèle. L'une des solutions qui permet de déterminer la portée d'une ontologie consiste à définir ou planifier une liste de questions auxquelles une base de connaissance, basée sur l'ontologie, doit être capable de répondre (competency questions).

2.8.3.2. Conceptualisation

Elle consiste à identifier et à structurer les connaissances du domaine, à partir des sources d'informations. L'acquisition de ces connaissances peut s'appuyer à la fois sur l'analyse de documents et sur l'interview des experts du domaine. Une fois que les concepts sont identifiés par leurs termes, leur sémantique est décrite dans un langage semi-formel (tables et graphes) à travers leurs propriétés› leurs instances connues et les relations qui les lient entre eux.

2.8.3.3. Implémentation

Cette étape consiste à formaliser le modèle conceptuel obtenu dans l'étape précédente par un formalisme de représentation d'ontologie telles que les logiques de description. Puis› à coder l'ontologie dans un langage d'ontologie formel.

2.8.3.4. Maintenance

Cela peut s'agir d'une maintenance corrective ou évolutive de l'ontologie (nouveaux besoins de l'utilisateur), ce qui permet la validation et l'évolution de celle-ci. Cette activité est généralement faite par le constructeur et des experts du domaine. La validation se base sur l'exploitation des services d'inférences associés aux LDS,et qui sont offerts par des raisonneurs.

2.9. Mécanismes (Formalismes) de représentation des connaissances

Représenter des connaissances propres à un domaine consiste à décrire et à coder les éléments de ce domaine pour qu'une machine puisse les manipuler afin de raisonner. Il existe un certain nombre de formalismes de représentation de connaissances ; ceux qui ont été les plus utilisés pour représenter les ontologies sont :

- Les frames ;
- Les graphes conceptuels ;
- Et les logiques de descriptions.

2.9.1. Les Frames

Le modèle des Frames est un classique de l'Intelligence Artificielle, et a été initialement proposé comme langage de représentation d'ontologies par T.GRUBER. Le principe de ce modèle est de décomposer les connaissances en classes (ou frames) qui représentent les concepts du domaine. A un frame est rattaché un certain nombre d'attributs (slots), chaque attribut pouvant prendre ses valeurs parmi un ensemble de facettes (facets). Une autre façon e présenter ces attributs est de les considérer comme des relations binaires entre classes dont le premier argument est appelé domaine (domain) et le deuxième est appelé portée (range).

Des instances des classes, correspondant à l'extension de chaque concept, peuvent être ajoutées, ainsi que des fonctions qui sont des types particuliers de relations liant un ensemble de classes à une valeur calculée à partir des valeurs des attributs des classes. La spécification de propriétés conceptuelles des attributs (ou relations) recourt à des formules de la logique du premier ordre.

2.9.2. Les graphes conceptuels

Le modèle des Graphes Conceptuels (GC), introduit par John F.SOWA en 1984, est un modèle opérationnel de représentation de connaissances, qui appartient à la famille des réseaux sémantiques.

Le formalisme réseaux sémantique est développé par M. Quillian pour représenter la sémantique du langage naturel. Ce formalisme a une représentation de graphe dont les nœuds représentent des concepts et des individus. Ces nœuds sont connectes par des arcs étiquetés. Il y a deux sortes d'arcs : les arcs de propriété qui affectent les propriétés à des concepts ou a des individus et les arcs IS-A qui introduisent les relations hiérarchiques entre des concepts ou entre des individus.

Le modèle des GCs est mathématiquement fondé sur la logique et la théorie des graphes. Cependant, pour raisonner à l'aide des GC, deux approches peuvent être distinguées :

(1) Raisonner à l'aide de la logique en considérant les GCs comme une interface graphique et ;
(2) Considérer les GCs comme un modèle de représentation à part entière disposant de ses propres mécanismes de raisonnement fondés sur la théorie des graphes.

Le modèle des GCs se décompose en deux parties :

- Une partie terminologique dédiée au vocabulaire conceptuel des connaissances à représenter, c'est-à-dire les types de concepts, les types de relations et les instances des types de concepts. Cette partie correspond à la représentation du modèle conceptuel mais intègre également des connaissances sur la hiérarchisation des types de concepts et de relations.
- Une partie assertionnelle dédiée à la représentation des assertions du domaine de connaissances étudié.

2.9.3. Les logiques de descriptions

Les logiques de descriptions LDs (Description Logic DL en anglais), appelées parfois logiques terminologiques, sont des langages formels conçus pour décrire et raisonner sur les connaissances d'un domaine. Elles ont été introduites par Brachman en 1979, par la suite elles ont connues de nombreux développements. Elles sont issues de la logique des prédicats, des frames et des réseaux sémantiques (des correspondances existent entre ces logiques et ces formalismes).

2.9.3.1. Les constructeurs des LDs

Les entités de base qui sont définies et manipulées dans une logique de descriptions sont les concepts et les rôles. Un concept permet de représenter un ensemble d'individus, et un rôle représente une relation binaire entre concepts.

Un concept et un rôle possèdent une description structurée, élaborée à partir d'un certain nombre de constructeurs, les concepts et les rôles peuvent être primitifs ou définis. Les concepts (éventuellement les rôles) primitifs sont comparables à des atomes et servent de base à la construction des concepts définis (éventuellement les rôles). Il existe de nombreux constructeurs permettant de former toute une famille de logiques de description. La table 2.1 décrit les plus importants.

Constructeur	Syntaxe	Sémantique (I : une interprétation)
Universel	$\top$	ΔI (ΔI : ensemble de tous les objets)
Absurde	$\bot$	$\varnothing$
Négation	$\neg C$	$\Delta I \setminus CI$
Conjonction	$C \cap D$	$CI \cap DI$
Disjonction	$C \cup D$	$CI \cup DI$
Restriction universelle	$\forall r.C$	$\{x \in \Delta I \ / \ \forall y, (x,y) \in Ri \Rightarrow y \in CI\}$
Restriction existentielle	$\exists r.C$	$\{x \in \Delta I \ / \exists y, (x,y) \in rI \text{ et } y \in CI\}$
Cardinalité minimum	$(\geq n\ r)$	$\{x \in \Delta I \ / \ \lvert\{y \mid (x,y) \in rI\}\rvert \geq n\}$
Cardinalité maximum	$(\leq n\ r)$	$\{x \in \Delta I / \ \lvert\{y \mid (x,y) \in rI\}\rvert \leq n\}$
Conjonction des rôles	$r \cap s$	$\{(x,y) \in \Delta I x \Delta I / (x,y) \in rI \wedge (x,y) \in sI\}$
Disjonction des rôles	$r \cup s$	$\{(x,y) \in \Delta I x \Delta I / (x,y) \in rI \vee (x,y) \in sI\}$

Table 2.1 : Les constructeurs essentiels d'une logique de description.

2.9.3.2. Les deux niveaux de description

La modélisation des connaissances d'un domaine avec les LDs se réalise en deux niveaux. Le premier, le niveau terminologique ou TBox, décrit les connaissances générales d'un domaine alors que le second, le niveau assertionnel ou ABox, représente une instanciation spécifique.

Une TBox comprend la définition des concepts et des rôles, alors qu'une ABox décrit les individus en les nommant et en spécifiant en terme de concepts et de rôles, des assertions qui portent sur ces individus nommés.

TBox	ABox
Femelle ⊑ ⊤ ⊓ ¬Mâle	Humain(Anne)
Mâle ⊑ ⊤ ⊓ ¬Femelle	Femelle(Anne)
Animal ≡ Mâle ⊔ Femelle	Femme(Sophie)
Humain ⊑ Animal	Humain(Robert)
Femme ≡ Humain ⊓ Femelle	¬Femelle(Robert)
Homme ≡ Humain ⊓ ¬Femelle	Homme(David)
Mère ≡ Femme ⊓ ∃relationParentEnfant	relationParentEnfant(Sophie, Anne)
Père ≡ Homme ⊓ ∃relationParentEnfant	relationParentEnfant(Robert, David)
MèreSansFille ≡ Mère ⊓ ∀relationParentEnfant. ¬Femme	
relationParentEnfant ⊑ $\top_R$	

Table 2.2 : Une base de connaissances composée d'une TBox et d'une ABox

2.9.3.2.1. Le niveau terminologique (TBox)

Les concepts atomiques et rôles atomiques constituent les entités élémentaires d'une TBox.

Les noms débutant par une lettre majuscule désignent les concepts, alors que ceux débutant par une lettre minuscule dénomment les rôles (par exemple : les concepts *Femelle*, *Mâle*, *Homme* et *Femme*, et le rôle *relationParentEnfant*).

2.9.3.2.2. Le niveau assertionnel (ABox)

Une ABox contient un ensemble d'assertions sur les individus. Chaque ABox doit être associée à une TBox, car les assertions s'expriment en terme de concepts et de rôles de la TBox. Une ABox désigne des individus caractérisés par des assertions d'individus nommés.

Une assertion de rôle, de la forme *R (a, b)* indique que pour cette ABox qu'il existe un individu nommé *a* qui est en relation avec un individu nommé *b* par le rôle *R* (défini dans la TBox associée).

2.9.3.3. L'inférence dans les logiques de description

Elle s'effectue au niveau terminologique ou assertionnel. Quatre principaux problèmes se présentent pour chacun de ces deux niveaux que le moteur d'inférence essai de les résoudre.

- L'inférence au niveau terminologique : quatre principaux problèmes d'inférences se présentent au niveau terminologique :
  - Satisfiabilité : Trouver tous les concepts insatisfiables.
  - Subsomption : Calculer la hiérarchie de subsomption ou taxonomie de concepts.

- L'équivalence : Trouver les concepts équivalents.
- La disjonction : Trouver les concepts disjoints.

- L'inférence au niveau assertionnel : le niveau assertionnel comprend quatre principaux problèmes d'inférence :
  - Cohérence de la ABox : les assertions définies restent cohérentes avec la TBox.
  - Vérification d'instance : vérifier que chaque instance respecte la définition de son concept.
  - Vérification de rôle : vérifier si les rôles de l'instance sont correctement utilisés.
  - Le problème de récupération.

2.10. Les outils de développement d'ontologies

Les outils de développement d'ontologies qui existent sur le marché aujourd'hui sont divers et variés. Cet état de choses suscite beaucoup d'interrogations lorsque vient le moment d'en choisir un pour construire une nouvelle ontologie : L'outil offre t-il une assistance au développement ? L'outil dispose t-il d'un moteur d'inférence ? Quels langages d'ontologies l'outil supporte t-il ? L'outil permet-il d'importer/exporter des ontologies ? L'outil offre t-il un support à la réutilisation d'ontologies existantes ? L'outil permet-il de documenter les ontologies construites ? L'outil offre t-il un support graphique à la construction des ontologies ? L'outil est-il stable, convivial, «mature» ? Les réponses à toutes ces questions pourraient s'avérer décisives dans le choix de l'un ou l'autre outil. Dans cette section nous passons en revue les principaux outils disponibles.

2.10.1. Les langages de spécification d'ontologies

Une des principales décisions à prendre dans le procédé de développement d'ontologies consiste à choisir le langage dans lequel l'ontologie sera exprimée et utilisée.

Le boom d'internet à mené à la création des langages d'implémentation des ontologies exploitant les caractéristiques du web. Ils sont connus sous le nom des langages de balisage (markup languages) ou des langages d'ontologie dans le contexte du Web sémantique (web-based ontology language). Certains d'entre eux sont basés sur la syntaxe de XML tels que : XOL (ontology Exchange Language), SHOE (Simple HTML Ontology Extension) qui a été précédemment basé sur le HTML, RDF (Resource Description Framework) et RDF Schéma qui est une extension de RDF. Les deux derniers sont des langages développés par des groupes de travail du World Wide Web Consortium (W3C). La combinaison de RDF et RDF Schéma est connue sous le nom de RDF(S). Par la suite , trois autres langages ont été développés comme

une extension de RFD(S) qui sont basés sur la logique de description : OIL (Ontology Inference Layer), DAML+OIL et OWL qui est le successeur de DAML+OIL.

La Figure 2.2 présente les langages de spécification d'ontologie qui sont récemment développés et les rapports principaux entre eux sous forme d'une pyramide des langages du Web sémantique.

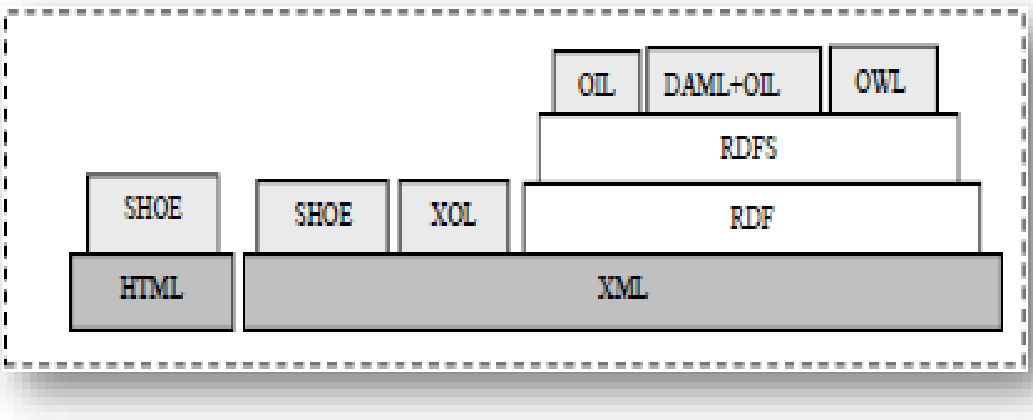


Figure 2.2 : La pyramide des langages d'ontologies basés Web

2.10.1.1. RDF

RDF (Ressource Description Framework) développé et recommandé par le W3C, permet de décrire les ressources du web sémantique qui sont l'élément de base de RDF. Chaque ressource est pourvue d'un identifiant URI (Uniform Resource Identifier).

Tout document RDF est composé d'un ensemble de triplets (sujet, prédicat, objet) ou encore (ressource, propriété, valeur). Un ensemble de tels triplets est appelé un graphe RDF. Ceci peut être illustré par un diagramme composé de nœuds et d'arcs orientés, dans lequel chaque triplet est représenté par un lien nœud-arc-nœud (d'où le terme de 'graphe').

A ce modèle est associée une syntaxe écrite en XML et basée sur les triplets :

- Ressource (Sujet) : une entité d'informations pouvant être référencée par un identificateur. Cet identificateur doit être une URI.
- Propriété (prédicat) : l'attribut ou la relation utilisée pour décrire une ressource.
- Valeur (objet) : la valeur d'une propriété associée à une ressource spécifique.

Exemple : Sami a 23 ans et habite Constantine.

```
<rdf :RDF>
<rdf :Description about='Sami'>
<rdf :Property about='ville'>
    Constantine
</rdf :Property>
<rdf :Property about='age'>
    23
</rdf :Property>
</rdf :Description>
</rdf :RDF>
```

2.10.1.2. RDF(S)

RDFS est un langage permettant de définir des schémas de méta-données. Il définit le sens, les caractéristiques et les relations d'un ensemble de propriété. La principale nouvelle notion est la distinction entre une classe (concept d'une ontologie) et une instance (individu d'une ontologie). Quelques notions définies sont : (rdfs : Class), (rdfs : SubClassOf), (rdfs : domain), (rdfs : range).

Sur l'exemple de Sami, nous définissons le concept de personne, une taxinomie de concepts, et l'instance Sami.

```
<rdf :RDF>
<rdfs :Class rdf :about='Personne'>
<rdfs :subClassOf rdf :resource='Thing'/>
</rdfs :Class>
<rdf :Property about='age'>
<rdfs :domain rdf :resource='Personne'/>
<rdfs :range rdf :resource='xsd :integer'/>
</rdf :Property>
<rdf :Property about='ville'>
<rdfs :domain rdf :resource='Personne'/>
<rdfs :range rdf :resource='xsd :string'/>
</rdf :Property>
</rdf :RDF>
```

```
<Personne rdf :ID='Sami'>
<age rdf :resource='23' />
<ville rdf :resource='Constantine' />
</Personne>
```

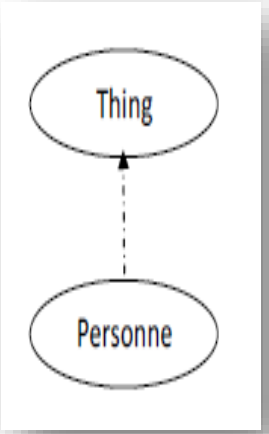


Figure 2.3 : Taxinomie de concept personne

2.10.1.3. DAML+OIL

DAML (DARPA Agent Markup Language) est un langage qui a comme but de fournir les fondations pour la génération suivante du Web sémantique. Comme RDFS, ce langage n'est pas assez expressif relativement aux exigences du Web sémantique, un nouveau langage nommé DAML-ONT a été développé en tant qu'extension de RDF avec les capacités d'un langage de représentation du savoir.

En même temps, nouveau langage nommé OIL a été développé par un groupe des chercheurs pour le même but. Ce langage a une syntaxe basée sur RDF et il est explicitement construit pour sa sémantique puisse être spécifiée à travers une description logique très expressive, la logique de description.

DAML+OIL est la combinaison de ces deux langages. Il hérite des avantages de ces deux langages. En conséquence, DAML+OIL est un langage très expressif et lisible par la machine ainsi que par un être humain avec une syntaxe basée sur RDF.

2.10.1.4. OWL

OWL (Ontology Web Language) est le standard actuellement recommandé par W3C pour représenter les ontologies. C'est une extension du vocabulaire de RDF(S). Il est dérivé du langage d'ontologie DAML+OIL.

OWL se compose de trois sous-langages OWL Lite, OWL DL et OWL Full, qui offrent des capacités d'expression croissantes, chacun est une extension par rapport à son prédécesseur plus simple.

- OWL Lite : répond à des besoins de hiérarchie de classification et de fonctionnalités de contraintes simples de cardinalité 0 ou 1. Une cardinalité 0 ou 1 correspond à des relations fonctionnelles, par exemple, une personne a une adresse. Toutefois, cette personne peut avoir un ou plusieurs prénoms, OWL Lite ne suffit donc pas pour cette situation.
- OWL DL : concerne les utilisateurs qui souhaitent une expressivité maximum couplée à la complétude du calcul (cela signifie que toutes les inférences seront assurées d'être prises en compte) et la décidabilité du système de raisonnement (c'est-à-dire que tous les calculs seront terminés dans un intervalle de temps fini). Ce langage inclut toutes les structures OWL avec certaines restrictions, comme la séparation des types : une classe ne peut pas

aussi être un individu ou une propriété. Il est nommé DL car il correspond à la logique descriptive.

- OWL Full : se destine aux personnes souhaitant une expressivité maximale. Il a l'avantage de la compatibilité complète avec RDF/RDFS, mais l'inconvénient d'avoir un haut niveau de capacité de description, quitte à ne pas garantir la complétude et la décidabilité des calculs liés à l'ontologie.

2.10.2. Les éditeurs d'ontologies

De nombreux éditeurs d'ontologies utilisant des formalismes variés et offrant différentes fonctionnalités ont été développés. Parmi ces outils on trouve : OntoEdit, OILed, Web ode, DOE, Protégé, etc. Voici la description de quelques-uns :

2.10.2.1. OntoEdit

OntoEdit est également un environnement de construction d'ontologies indépendant de tout formalisme. Des outils graphiques dédiés à la visualisation d'ontologies sont inclus dans l'environnement. OntoEdit intègre un serveur destiné à l'édition d'une ontologie par plusieurs utilisateurs. Un contrôle de la cohérence de l'ontologie est assuré à travers la gestion des ordres d'édition.

2.10.2.2. OILed

L'éditeur OILed a été développé en 1991 à l'université de Manchester pour éditer des ontologies dans les langages de représentation OIL, puis DAML+OIL. Il est orienté vers la représentation en logique de description expressive et, à ce titre, fournit tous les éléments d'interface permettant de spécifier des hiérarchies de concepts et de rôles, les restrictions sur les rôles et les instances.

Il peut être connecté à un raisonneur de logique des descriptions tel que FaCT et Racer, capable de tester la satisfiabilité des ontologies construites ou d'expliciter de nouvelles relations de subsomption entre concepts complexes.

2.10.3.3. Protégé

Protégé a été développé par le Stanford Medical Informatics de l'Université de Stanford. Protégé est une plate-forme Open Source autonome, qui fournit un environnement graphique permettant l'édition, la visualisation et le contrôle (vérification des contraintes) d'ontologies. Le modèle de représentation de connaissances de protégé, est issu du modèle des frames. Ce dernier contient des classes (pour modéliser les concepts), des slots (pour modéliser les attributs

des concepts) et des facettes (pour définir les valeurs des propriétés et des contraintes sur ces valeurs), ainsi que des instances des classes. Protégé-OWL introduit la notion de métaclasse, dont les instances sont des classes, ce qui permet de créer son propre modèle de connaissances avant de bâtir une ontologie. De nombreux plug-ins sont disponibles ou peuvent être ajoutés par l'utilisateur.

2.11. Conclusion

A travers ce que nous avons présenté dans ce chapitre, il en ressort que la notion d'ontologie constitue une approche très efficace pour représenter les connaissances. Tout au long de ce chapitre, nous avons essayé d'éclaircir la notion d'ontologie en présentant certaines définitions. Ensuite, nous avons exposé les composantes d'une ontologie, les différentes classifications ainsi que le cycle de vie d'une ontologie et les différentes besoins de cette dernière, ensuite nous avons représenté quelques méthodologies de construction des ontologies, nous avons montré après les principaux formalismes de représentation de connaissances à savoir les frames, les graphes conceptuels et les logiques de descriptions. Et finalement, nous avons présenté les outils nécessaires de développement à savoir les langages de représentation, les outils d'édition et d'interrogation,…, etc.

Chapitre 3

Conception et Réalisation

3.1. Introduction

Ce chapitre est consacré à la partie conception et réalisation de notre application, qui consiste à recommander des cours à un utilisateur en se basant sur une otologie de cours et en combinant deux mesures de similarité : numérique et sémantique. Nous présentons, également l'architecture de notre système, l'ontologie utilisée, l'environnement de travail choisi tout en présentant les langages et les outils utilisés ainsi que les copies d'écran de chaque étape.

3.2. Modèle de données

Dans cette section, nous présentons l'ensemble des définitions relatives aux concepts de base que nous manipulons dans ce chapitre.

Préférence : Une préférence est une formule qui permet d'hiérarchiser un ensemble d'objet par rapport aux intérêts et aux besoins d'un utilisateur. Dans le reste de ce chapitre, nous nous focaliserons sur les préférences quantitatives que nous modélisons comme suit : Une préférence ***pi*** est un couple ***pi (pri, wi)*** où ***pri*** est un prédicat de type <*attribut, opérateur, valeur*> et ***wi*** est un nombre réel compris entre 0 et 1 qui représente le degré d'intérêt de l'utilisateur par rapport au prédicat ***pri***. La valeur 0 reflète la préférence minimale alors que la valeur 1 reflète la préférence maximale.

Exemple : Soit R un schéma de relation modélisant des voitures, R (marque, type, prix, couleur, kilométrage, puissance). Un utilisateur peut définir les préférences suivantes: *P1(<couleur, '=', 'rouge'>, 0.9)*. La préférence *p1*, exprime le fait accorde un score 0.9 aux tuples de la relation R (voiture) pour lesquels l'attribut *couleur* prend la valeur *rouge*.

Profil utilisateur : Dans notre approche, le profil utilisateur se résume à un ensemble de prédicats pondérés. ***P1 : <Cours.categorie, '=', data mining>*** est un exemple de prédicat. Un utilisateur U1 qui a une grande préférence pour les cours de data mining peut attribuer à ce prédicat une pondération (poids) w1=0,9 par exemple. Le profil de l'utilisateur U1 sera donc constitué des paires d'éléments : (P1, w1), (P2, w2), etc.

Descriptif de contenu : Le module Création de Profil Cours a pour tâche de créer le descriptif de cours. Cela correspond, dans notre application, aux catégories associées aux cours. Un cours peut avoir plusieurs catégories.

3.3. Conception de notre application

Pour décrire les différentes étapes de conception de notre application, nous avons opté pour le langage UML.

3.3.1. Diagramme de classe

Le diagramme de classes est considéré comme le plus important de la modélisation orientée objet, il est le seul obligatoire lors d'une telle modélisation. Il représente les classes intervenant dans le système. Le diagramme de classe est une représentation statique des éléments qui composent un système et de leurs relations. Chaque application qui va mettre en œuvre le système sera une instance des différentes classes qui le compose. La figure 3.1 montre le diagramme de classes de notre application.

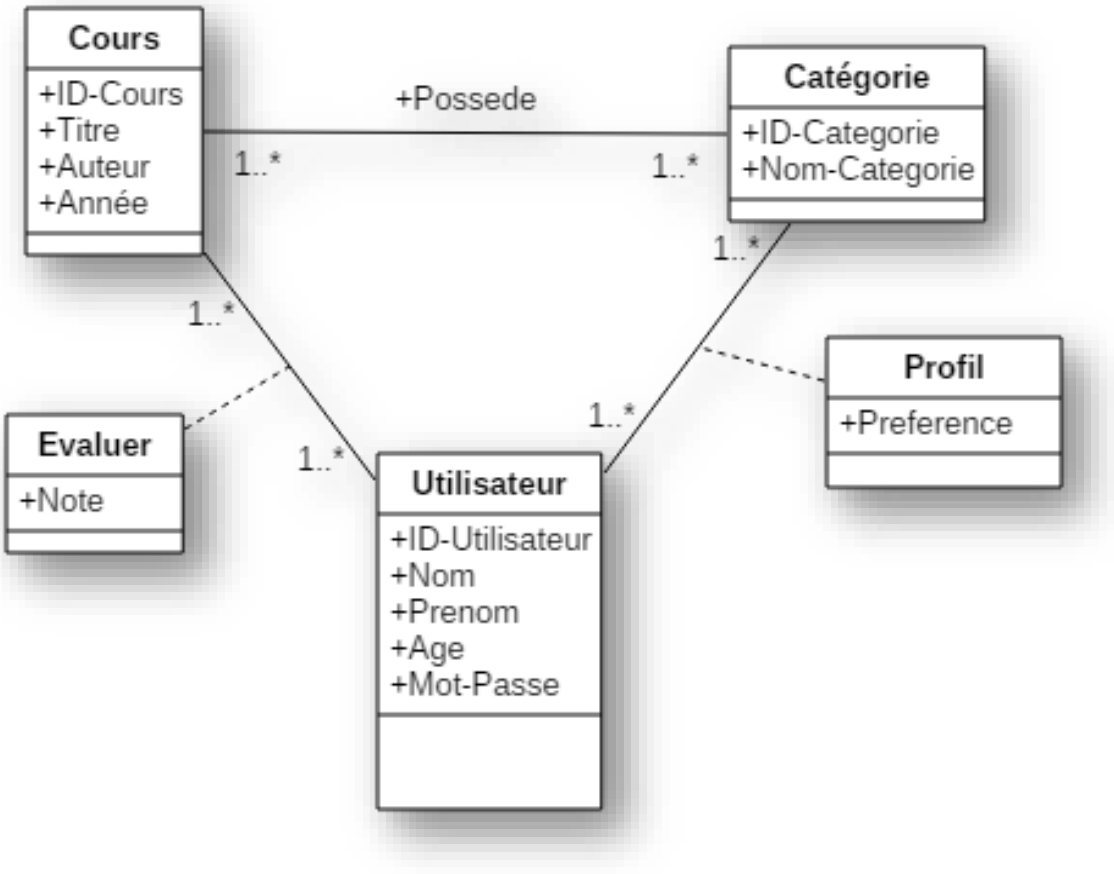


Figure 3.1. Diagramme de classes

3.3.2. Diagramme de séquences

Le diagramme de séquences représente l'une des vues dynamiques les plus importantes d'UML. La figure 3.2 montre les interactions entre les différents objets de notre application.

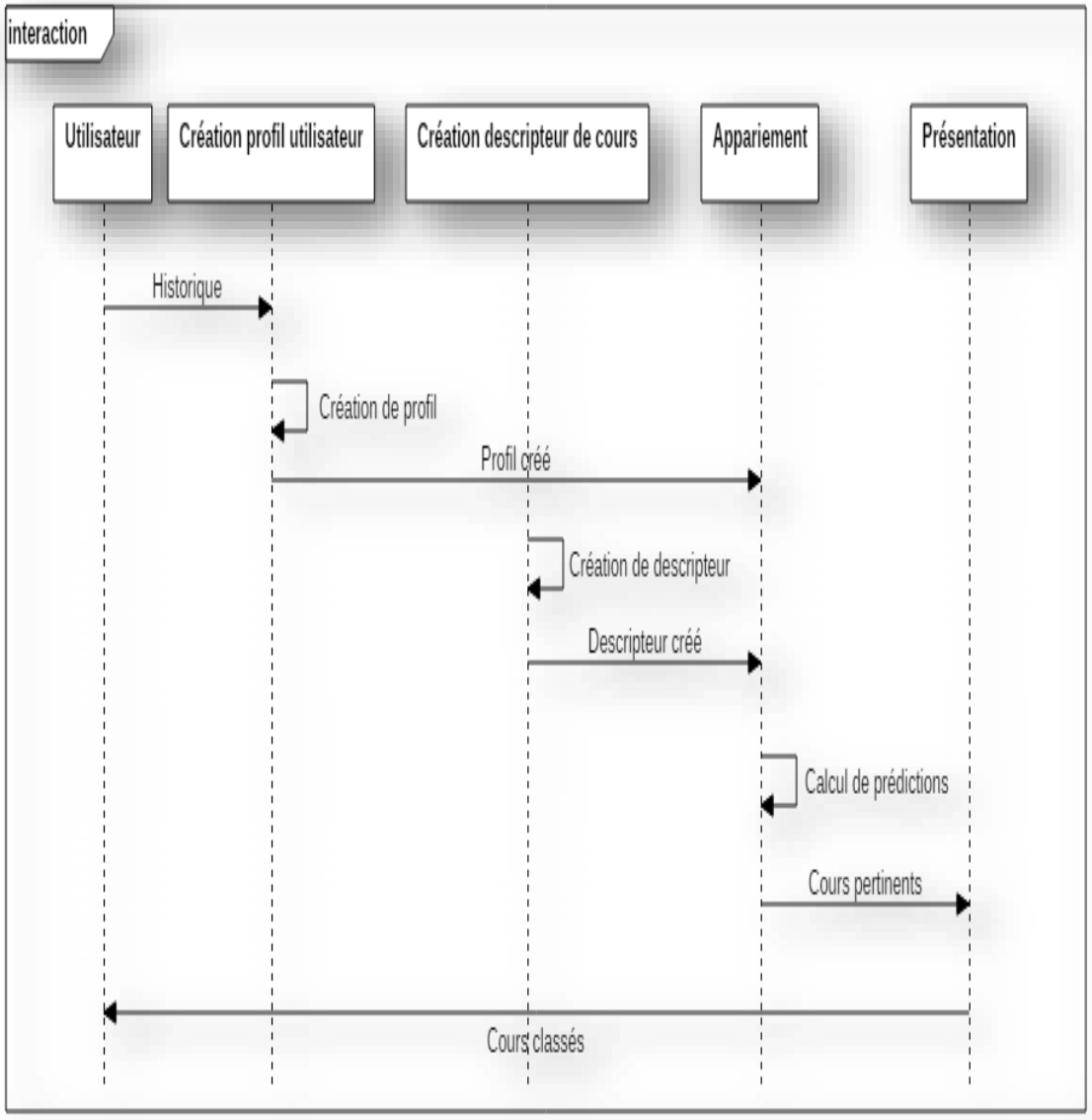


Figure 3.2. Diagramme de séquences

3.3.3. Diagramme de collaboration :

Un diagramme de collaboration montre les relations entre les objets jouant les différents rôles. Toutefois, ce diagramme ne contient aucune notion de temps.

L'utilisateur ait évalué un certain nombre de cours, le module « Création profil utilisateur » construit son profil à partir de son historique et le renvoi au module « Appariement ». Le module «Appariement » reçoit ainsi les descripteurs de cours créés par le module « Création descripteur de cours » est calcule la similarité entre le profil utilisateur et les différents descripteurs de cours, pour ainsi renvoyer les cours pertinents au module « Présentation ». Le module « Présentation » classe les cours par ordre décroissant selon leurs degrés de similarités et les affiche pour les utilisateurs. La figure 3.3 illustre l'échange d'informations en les différents modules.

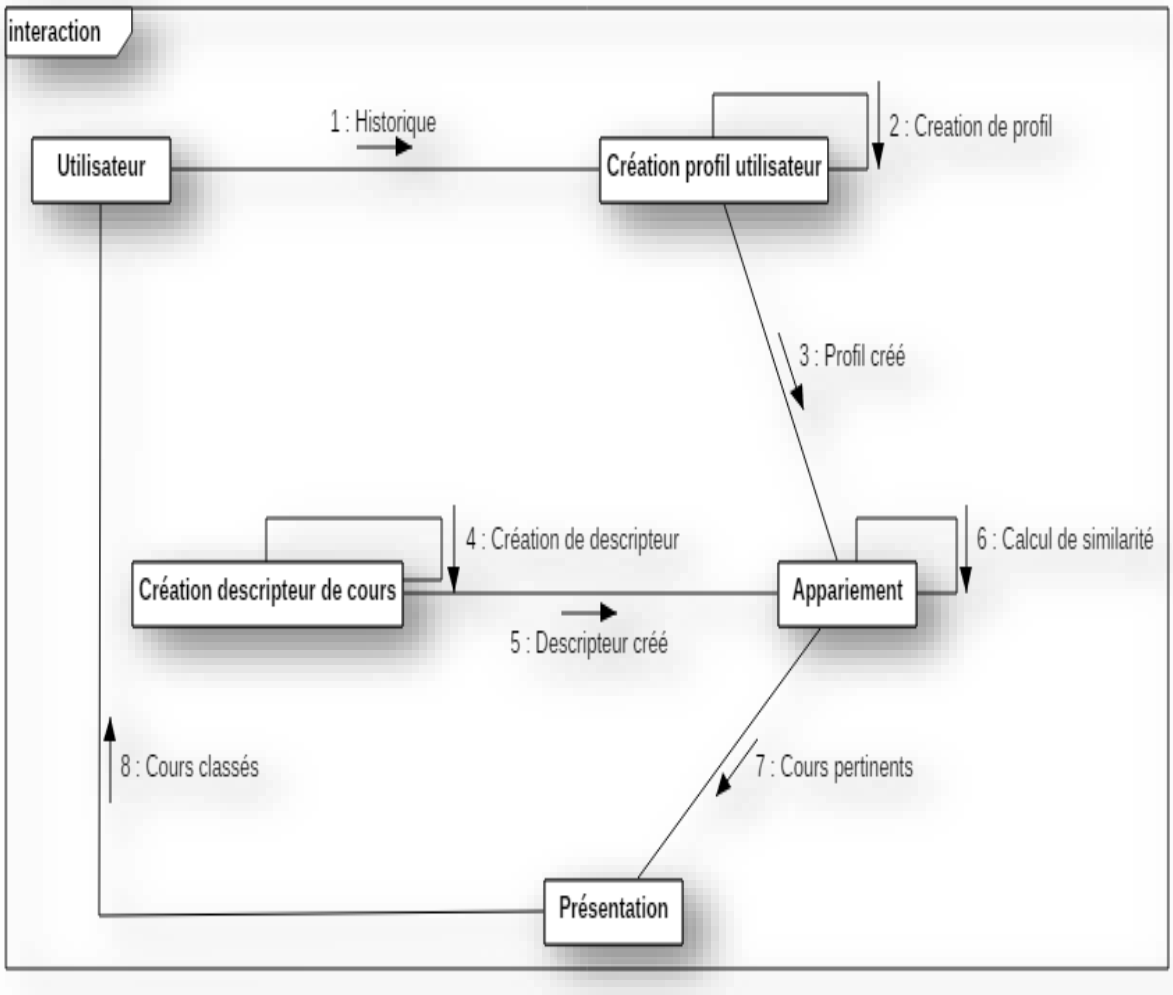


Figure 3.3. Diagramme de collaboration

Dans la section suivante, nous présentons l'architecture générale de notre système de recommandation.

3.4. L'architecture générale de notre système

Comme le montre la figure 3.4, L'architecture globale de notre système de recommandation comporte quatre modules principaux.

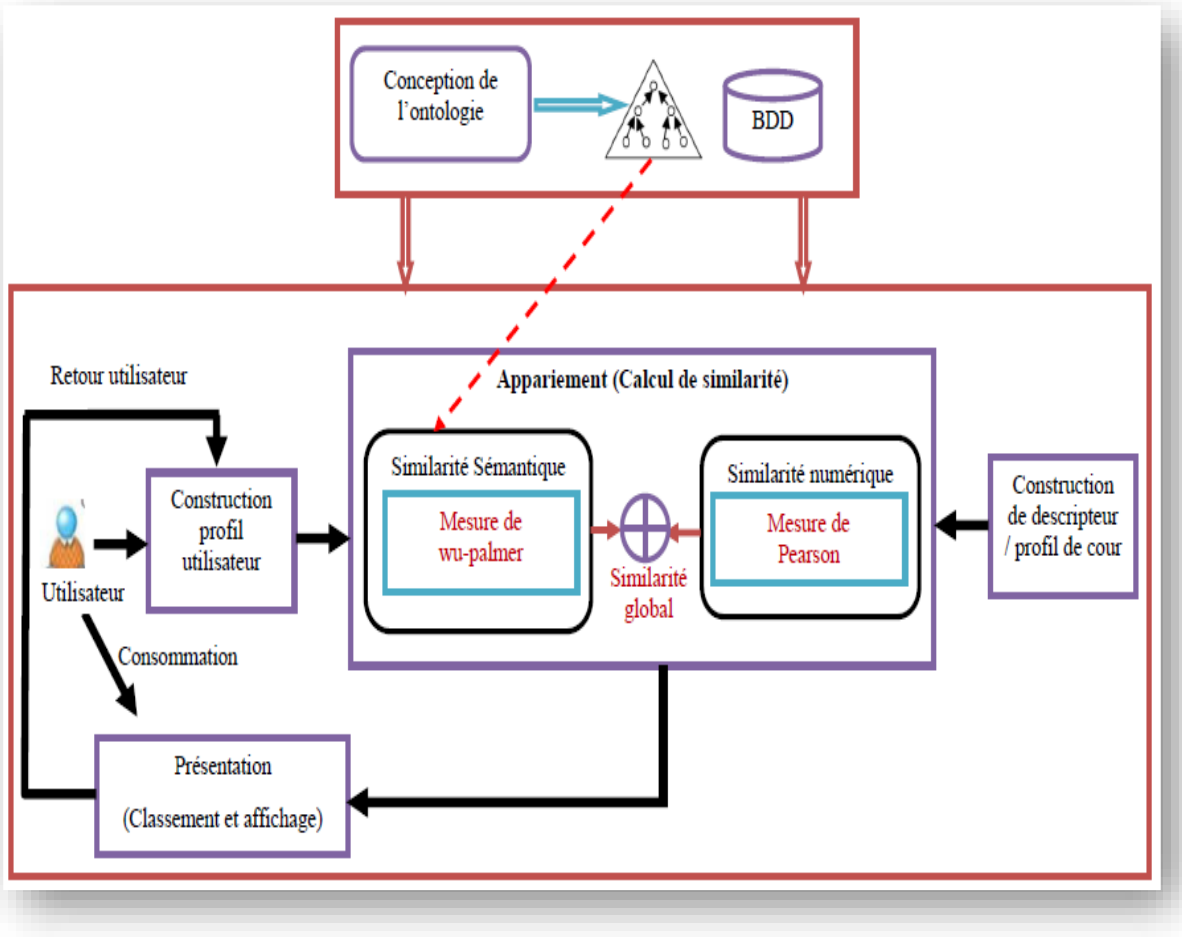


Figure 3.4. L'architecture globale de notre système

3.4.1. Création descripteur/profil de cours

Ce module a pour tâche de créer le descriptif des cours à partir des informations stockées dans la base de données et l'ontologie. Cela correspond, dans notre application, à catégories associées aux cours. Un cours peut avoir plusieurs catégories.

3.4.2. Création profil utilisateur

Ce module a pour tâche de créer les profils utilisateurs. Cela correspond, dans notre application, au calcul de préférence pour chaque utilisateur. Pour simplifier, nous avons supposé, dans notre système, que le profil de l'utilisateur se résume à un ensemble de prédicats pondérés. Un prédicat est de la forme ***<attribut,opérateur de comparaison,valeur>***, où attribut est un élément du profil, et valeur est une valeur appartenant au domaine de l'attribut du profil.

P1 : **<*Cours.categorie, '=', Data_mining*>** est un exemple de prédicat. Un utilisateur U1 qui a une grande préférence pour les cours de ***Data_Mining*** peut attribuer à ce prédicat une pondération (poids) w1=0,9 par exemple. Le profil de l'utilisateur U1 sera donc constitué des paires d'éléments : (P1, w1), (P2, w2), etc.

3.4.3. Appariement profil/contenu

Le processus de base dans un système de recommandation (SR) est celui chargé d'évaluer la pertinence d'un contenu pour un utilisateur donné. Certains SR décident qu'un contenu est pertinent pour un utilisateur si les utilisateurs qui lui ressemblent ont consommé ce contenu et l'ont considéré comme pertinent pour eux. On parle alors de SR basé sur le filtrage collaboratif (CF). D'autres SR prédisent la pertinence d'un contenu pour un utilisateur en calculant la similarité (corrélation) entre le profil utilisateur et le descripteur de contenu. On parle alors de SR basé sur le filtrage à base de contenu (CBF). Dans ce qui suit, nous donnons une description de notre processus d'appariement entre profils et contenu dans le cas des systèmes de recommandation CBF. Comme le montre la figure 3.4, le processus d'appariement prend en entré le profil utilisateur et les descripteurs de cours, il renvoi une liste ordonnée de recommandation. Pour produire cette liste, le processus d'appariement calcul la similarité entre le profil utilisateur et chaque descripteur de cours candidat. Ces derniers sont alors ordonnés dans l'ordre décroissant de leur pertinence à un utilisateur (similarité avec le profil). Les K premiers contenus sont recommandés à l'utilisateur.

L'objectif de ce module est le calcul de la similarité entre les catégories associées à un cours et les catégories de cours préférés par un utilisateur. Par exemple, supposons que nous disposons de deux cours : ***Cours1(Mobile_Development)*** et ***Cours2(Big_Data)***, et qu'il n'y a qu'un seul prédicat **<*Cours.Catégorie, '=', Web_Development*>** dans le profil de l'utilisateur U1. En utilisant une ontologie de cours, et une mesure de similarité sémantique, il faut calculer la similarité entre le profil de U1 et les deux cours, pour décider lequel des deux cours lui recommander. Cela revient (pour cet exemple) à mesurer les similarités : ***sim (Web_Development, Mobile _evelopment)*** et ***sim (Web_Development, Big_Data)***. Dans ce cas, cours1 sera recommandé à U1 car le concept « ***Mobile_Development*** » est plus similaire au concept « ***Web_Development*** » que le concept « ***Big_Data*** ».

Nous avons utilisé deux types de similarité :

a) ***Similarité numérique :*** la mesure de Pearson a été utilisée pour calculer le taux de corrélation entre le profil utilisateur et le c***ours.*** Nous avons utilisé la corrélation de

Pearson entre les deux vecteurs: utilisateur représenté par $X(x_1, x_2, \dots, x_n)$ et cours représenté par $(y_1, y_2, \dots, y_n)$. La formule de Pearson est donnée par :

$$\text{Sim (x,y)} = r_p = \frac{\sum_{i=1}^{N}(x_i - \bar{x}) \cdot (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{N}(x_i - \bar{x})^2} \cdot \sqrt{\sum_{i=1}^{N}(y_i - \bar{y})^2}}$$

b) ***Similarité sémantique :*** nous avons implémenté la mesure de wu-Palmer.

Mesure de Wu & Palmer

La mesure de similarité de Wu et Palmer est basée sur le principe suivant : Etant donnée une ontologie formée d'un ensemble de nœud et un nœud racine R (Figure 3.5). Soit X et Y deux éléments de l'ontologie dont nous allons calculer la similarité. Le principe de calcul de similarité est basé sur les distances (N1 et N2) qui séparent les nœuds X et Y du nœud racine et la distance qui sépare le concept subsumant (CS) de X et de Y du nœud R.

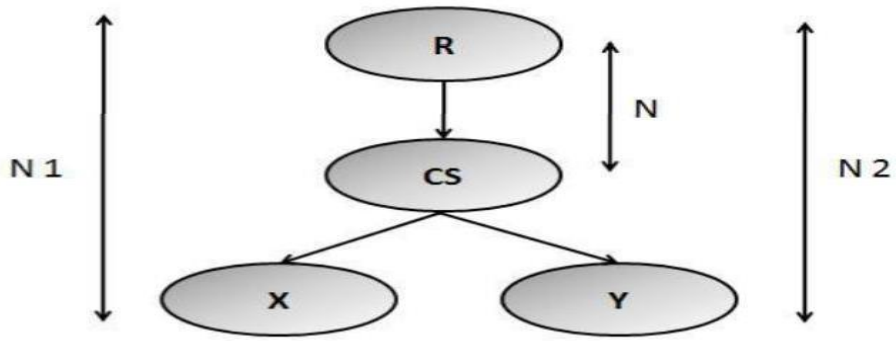


Figure 3.5. Exemple d'un extrait d'une ontologie

La mesure de Wu et Palmer est définie par la formule suivante :

$$\boldsymbol{Sim\,(X,Y) = \frac{2 * N}{N1 + N2}}$$

c) **Similarité globale :** notée $Sim_{globale}$, elle combine les deux similarités : numérique ($\boldsymbol{sim_{préf}}$) et sémantique ($\boldsymbol{sim_{sém}}$) selon la formule suivante:

$$Sim_{globale}(Pu, C) = \alpha \times sim_{sém}(Pu, C) + \beta \times sim_{préf}(\overrightarrow{Pu}, \overrightarrow{C})$$

Avec : $(\alpha + \beta = 1)$.

Nous avons combinés ces deux mesures, afin d'améliorer les résultats de la pertinence

des cours pour l'utilisateur demandeur de recommandations..

3.4.4. Présentation/Classement et affichage

Ce module ce charge de classer les cours par ordre décroissant selon le score de similarité qu'ils ont obtenu par le module d'appariement et les afficher à l'utilisateur.

Consommation : l'utilisateur choisit les cours qui lui sont proposes. Les cours consommés sont notés par l'utilisateur et stockés dans des logs afin d'enrichir son profil ultérieurement.

3.5. Conception de l'ontologie

Dans ce mémoire, nous proposons un système de recommandation ontologique basé sur le FC. Les concepts servant à modéliser les profils utilisateurs et les descripteurs de cours sont organisés dans une ontologie que nous allons créer.

3.5.1. Méthodologie de construction

Pour le développement de notre ontologie, nous avons essayé de suivre les étapes proposées dans [22]. Notre application nécessite une ontologie de cours, qui doit nous permettre de calculer le degré de ressemblance entre des catégories de cours (les catégories préférés par un utilisateur et les catégories associés à un cours) dans le but de recommander une liste de cours à un utilisateur. Par conséquent nous pouvons limiter le domaine de notre ontologie aux catégories de cours, aux propriétés des cours et leurs valeurs.

Pour construire l'ontologie, nous avons adopté les trois étapes ci-dessous :

- Spécifier les termes à collecter et les tâches à effectuer en utilisant cette ontologie. L'ontologie est construite dans l'esprit de fournir les recommandations les plus pertinentes à l'utilisateur r.
- Organiser les termes en utilisant les métas catégories : concepts, relations, attributs, etc.
- Affiner l'ontologie et la structurer selon des principes de modularité et d'organisation hiérarchiques.

3.5.1.1. Définition des classes et de la hiérarchie des classes

Nous avons, dans un premier temps, recensé tous les catégories de cours en s'inspirant des données des sources google livres[1][2] et des plateformes de e-learning et notamment « Class Central »[3]. Nous avons ensuite, classé les catégories de cours en classes et sous-classes formant ainsi une hiérarchie de classes ayant pour racine la classe cours. Ces classes constituent les concepts de notre ontologie. Nous avons retenu un certain nombre de catégorie de cours. Pour établir la hiérarchie des classes, nous avons procédé de haut en bas en commençant par les concepts les plus généraux et en terminant par la spécialisation des concepts. Nous avons, donc, commencé par les classes les plus générales. Ensuite, nous avons affiné chacune de ces classes.

3.5.2. Représentation hiérarchique de l'ontologie conceptuelle

Notre ontologie a été conçue avec l'éditeur « Protégé », distribué en open source par l'université en informatique médicale de Stanford[4]. Il permet, à travers son interface graphique la génération automatique du code OWL correspondant à l'ontologie. Notre ontologie est structurée en hiérarchie de catégories de cours dont la racine est représentée par le concept « Cours ». La figure 3.6, montre un extrait de cette ontologie et le tableau 3.1, monte la signification des abréviations des concepts de cette ontologie. L'ontologie au complet est présentée en annexe.

[1] http://www.openclasseroom.com/

[2] http://classeroom.edu/

[3] https://www.class-central.com/subjects#

[4] http://protege.stanford.edu/

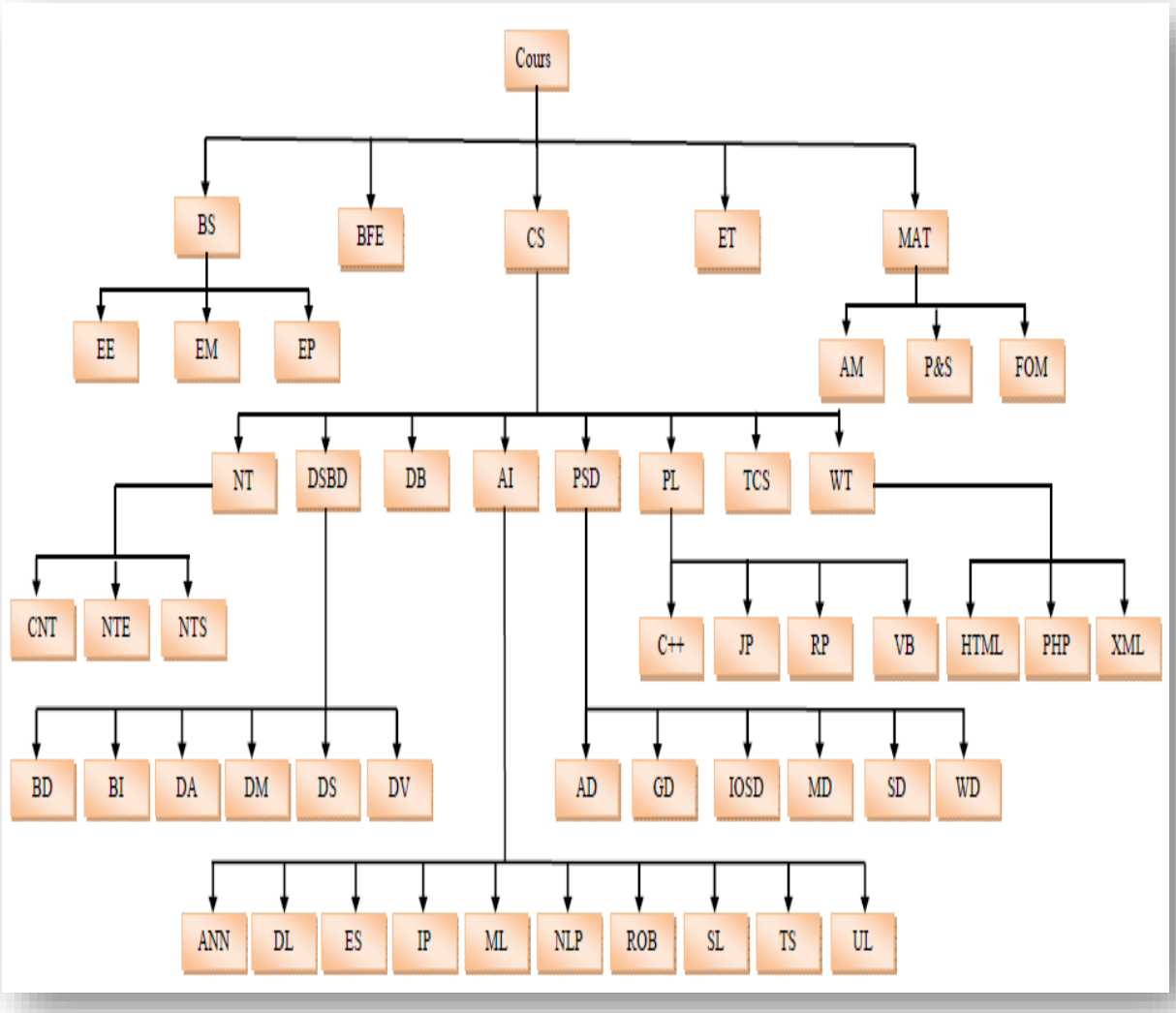


Figure 3.6. Un extrait de la représentation hiérarchique de l'ontologie des cours

Tableau 3.1. Signification des abréviations des concepts de l'ontologie

Abréviation	Signification	Abréviation	Signification
AD	Android Development	IOSD	iOS Development
AI	Artificial Intelligence	IP	Image Processing
AM	Applied Mathematics	JP	Java Programming
ANN	Artificial neural networks	MAT	Mathematics
BD	Big Data	MD	Mobile Development
BFE	Basic Foundations of Engineering	ML	Machine learning
BI	Bioinformatics	NLP	natural language processing
BS	Basic Science	NT	Networks
C++	C++	NTE	Network Engineering
CNT	Computer Networks	NTS	Network Security
CS	Computer Science	PHP	Hypertext Preprocessor
DA	Data Analysis	PL	Programming Language

DB	Database	PSD	Programming & Software Development
DL	Deep learning	ROB	Robotics
DM	Data Mining	RP	R-Programming
DS	Data science	S&P	Statistics & Probability
DSBD	Data Science & Big Data	SD	Software Development
DV	Data Visualization	SL	Supervised learning
EE	Electrical Engineering	TCS	Theoretical Computer Science
EM	Engineering Mathematics	TS	translation systems
EP	Engineering Physics	UL	Unsupervised learning
ES	Expert System	VB	Visual Basic
ET	Education & Teaching	WD	Web Development
FOM	Foundations of Mathematics	WT	Web Technology
GD	Game Development	XML	Extensible Markup Language
HTML	Hyper Text Markup Language		

3.6. Environnement de développement

Nous avons développé notre application sur une machine AMD Atlon 64 bits avec une vitesse de 1.60Ghz, dotée d'une capacité mémoire de 3GB DDR2, sous Windows 7. Notre système est implémenté en Java et Jena[5] API. Pour implémenter notre ontologie, notre choix porte sur OWL qui représente un langage de codification utilisé pour implémenter l'ontologie en OWL, et l'éditeur d'ontologies Protégé pour la création et la mise à jour de cette ontologie. Pour la base de données nous avons utilisé MySql.

3.6.1. NetBeans

NetBeans est un environnement de développement intégré (EDI), placé en open source en juin 2000 sous licence CDDL (Common Dévelopment and Distribution License). En plus de Java, NetBeans permet également de supporter différents autres langages, comme C, C++, JavaScript, XML, et HTML. Il comprend toutes les caractéristiques d'un IDE moderne (éditeur en couleur, projets multi-langage, éditeur graphique d'interfaces et de pages Web).

Conçu en Java, NetBeans est disponible sous Windows, Linux, ou sous une version indépendante des systèmes d'exploitation (requérant une machine virtuelle Java).

NetBeans constitue par ailleurs une plate-forme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plate-forme, il s'enrichit à l'aide de plugins. La page d'accueil de Jena, illustrée ci-dessous.

[5] http://jena.sourceforge.net/

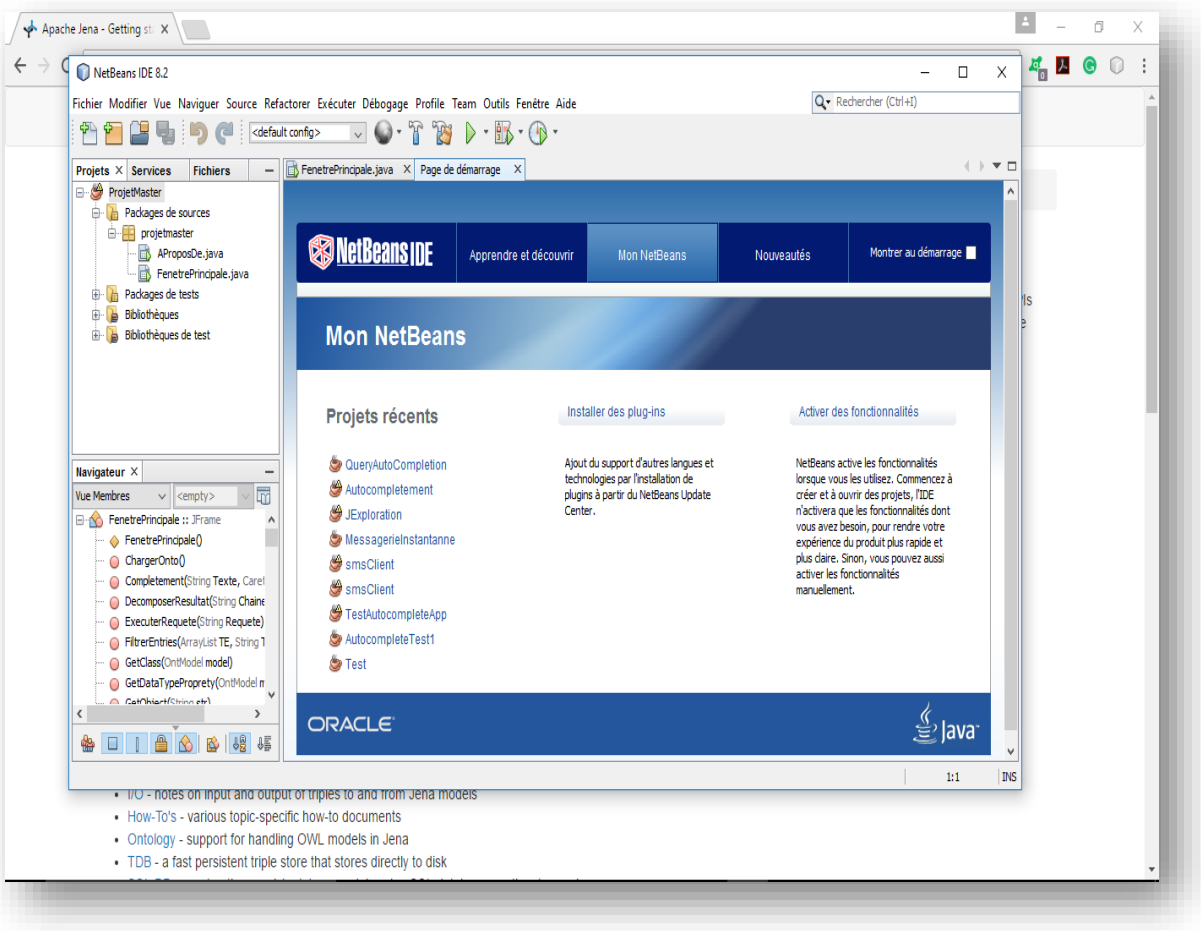


Figure 3.7. Fenêtre principale de NetBeans

3.6.2. Java

Le langage Java est un langage de programmation informatique orienté objet créé par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy. Il fut présenté officiellement en 1995. Selon les développeurs de Sun, Java (qui signifie café en argot américain) est un langage : simple, orienté-objet, distribué, interprété, robuste sécurisé, neutre vis à vis de l'architecture, portable, à haute performance, multi-threaded et dynamique. Le langage Java était à la base un langage pour Internet, pour pouvoir rendre plus dynamiques les pages (tout comme le JavaScript aujourd'hui). Mais le Java a beaucoup évolué et est devenu un langage de programmation très puissant permettant de presque tout faire. Contrairement à la plupart des autres langages (sauf la plateforme .Net), Java met à la disposition du développeur une API très riche lui permettant de faire de très nombreuses choses. Il existe plusieurs IDE (Integrated Development Environement) pour le langage JAVA par exemple Eclips, JBuilder et NetBeans que nous avons utilisé.

3.6.3 . Jena

Jena est un API java open source développé par le laboratoire de Hewlett-Packard permettant la lecture et la manipulation des ontologies décrites en RDFS ou en OWL. La page d'accueil de Jena, illustrée ci-dessous. Jena est un Framework constitué d'un ensemble d'outils implémenté en java :

- une API de programmation pour la gestion des données (RDF, RDFS, DAML+OIL, OWL) des applications de Web sémantique.
- un langage de requêtes qui est une implémentation du langage SPARQL.
- une structure relationnelle pour un stockage persistant des données RDF, RDFS, DAML+OIL et OWL.
- un parseur RDF/XML.
- un moteur d'inférence couplé à autres raisonneurs.
- un outil pour migrer les instances d'un format à un autre.

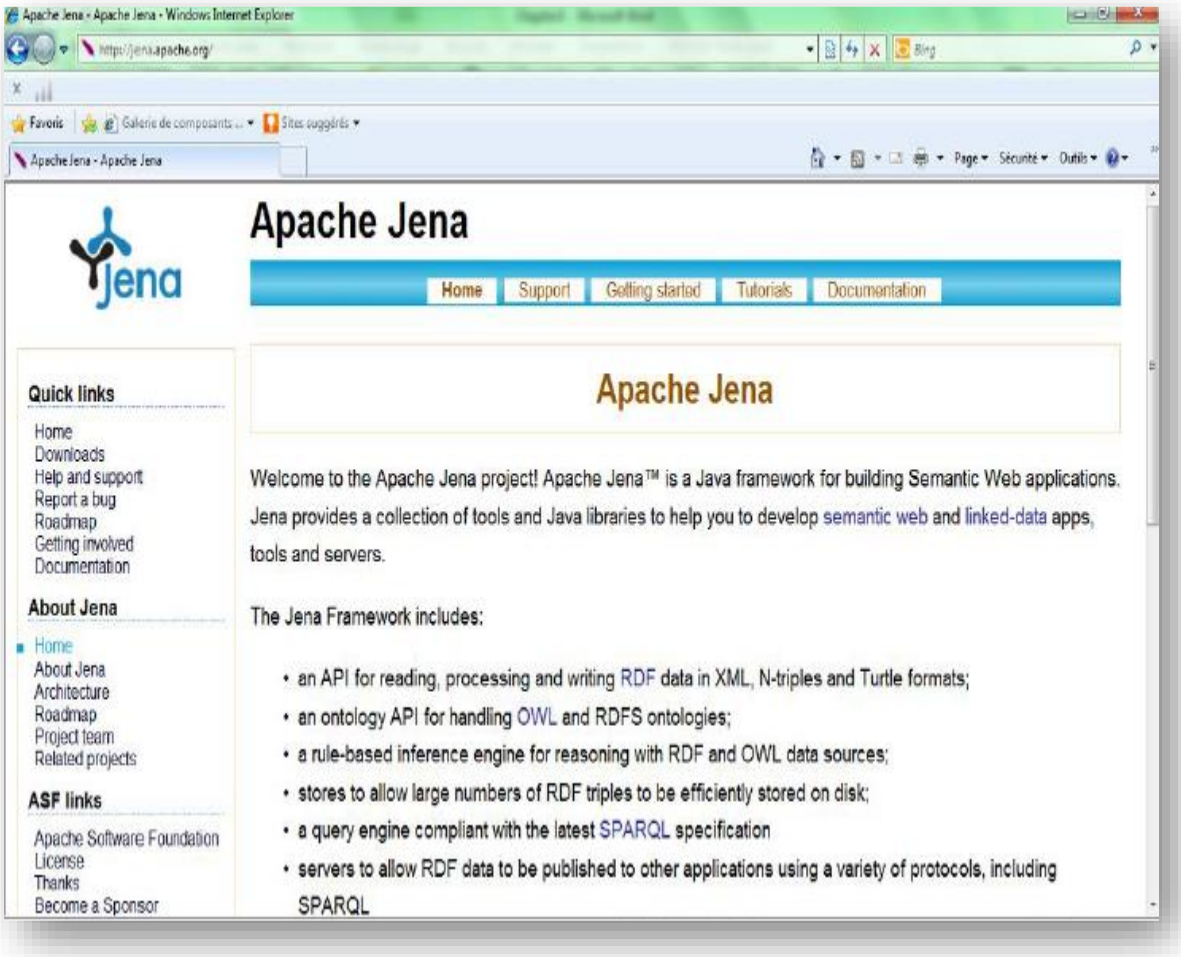


Figure 3.8. Page d'accueil Jena.

3.6.4. Le SGBD MySQL

MySQL est un système de gestion de bases de données relationnelles robuste et rapide. Il est très utilisé dans les projets libres et dans le milieu industriel. MySQL est un SGBDR facile à utiliser qui convient très bien pour la plupart des sites web. La rapidité de développement a été, depuis le début l'objectif principal de ceux qui l'ont écrit. Pour cela ils ont décidé de proposer moins de fonctionnalités, mais son installation et son utilisation sont plus aisées.

.MySQL est donc un serveur multiutilisateur et multithread. Il utilise SQL (Structured Query Language), le langage standard des requêtes des bases de données. MySQL est disponible depuis 1996, mais son développement remonte à 1979. Il s'agit de la base de données open-source la plus employée au monde et elle a reçu le Linux journal Readers' Choice Award à plusieurs reprises.

MySQL possède sur plusieurs avantages :

- des performances élevées ;
- un cout réduit ;
- une simplicité de configuration, et d'apprentissage ;
- sa portabilité ;
- l'accessibilité de son code source ;
- la disponibilité du support

3.6.5. JDBC (Java DataBase Connectivity)

3.6.5.1. Introduction

Les développeurs Java ont constaté qu'ils nécessitent un cadre de travail qui permettra de construire une interface uniforme sur les systèmes de connectivité de base de données d'un tel cadre permettrait au programmeur d'écrire une interface de base de données unique sur de nombreuses plates-formes ce cadre de travail est connu comme l'interface Java Database Connectivity.

3.6.5.2. Définition de JDBC

JDBC est une API fournie avec Java (depuis sa version 1.1) permettant de se connecter à des bases de données, c'est-à-dire que JDBC constitue un ensemble de classes permettant de développer des applications capables de se connecter à des serveurs de bases de données

(SGBD). L'API JDBC a été développée de telle façon à permettre à un programme de se connecter à n'importe quelle base de données en utilisant la même syntaxe, c'est-à-dire que l'API JDBC est indépendante du SGBD.

De plus, JDBC bénéficie des avantages de Java, dont la portabilité du code, ce qui lui vaut en plus d'être indépendant de la base de données d'être indépendant de la plateforme sur laquelle elle s'exécute.

3.6.5.3. Les caractéristiques de JDBC

Les caractéristiques d'internationalisation du pilote JDBC de Microsoft SQL Server incluent ce qui suit :

- Prise en charge d'une version complètement localisée dans les mêmes langues que SQL Server.
- Prise en charge des conversions de langage Java 1.4 pour les données SQL Server sensibles aux paramètres régionaux.
- Prise en charge des langues internationales, indépendamment du système d'exploitation.

3.6.6. Présentation de l'éditeur PROTEGE

PROTEGE est une interface modulaire, développée au Stanford Medical Informatics de l'Université de Stanford, permettant l'édition, la visualisation, le contrôle, l'extraction à partir de sources textuelles, et la fusion semi-automatique d'ontologies. PROTEGE OWL autorise la définition de méta-classes, dont les instances sont des classes, ce qui permet de créer son propre modèle de connaissances avant de bâtir une ontologie. De nombreux plugins sont disponibles ou peuvent être ajoutés par l'utilisateur. La Figure 3.9 présente l'interface de PROTEGE OWL.

L'interface, très bien conçue, et l'architecture logicielle permettant l'insertion de plug-ins pouvant apporter de nouvelles fonctionnalités (par exemple, la possibilité d'importer et d'exporter les ontologies construites dans divers langages opérationnels de représentation tels que OWL ou encore la spécification d'axiomes) ont participé au succès de PROTEGE OWL, qui regroupe une communauté d'utilisateurs très importantes et constitue une référence pour beaucoup d'autres outils.

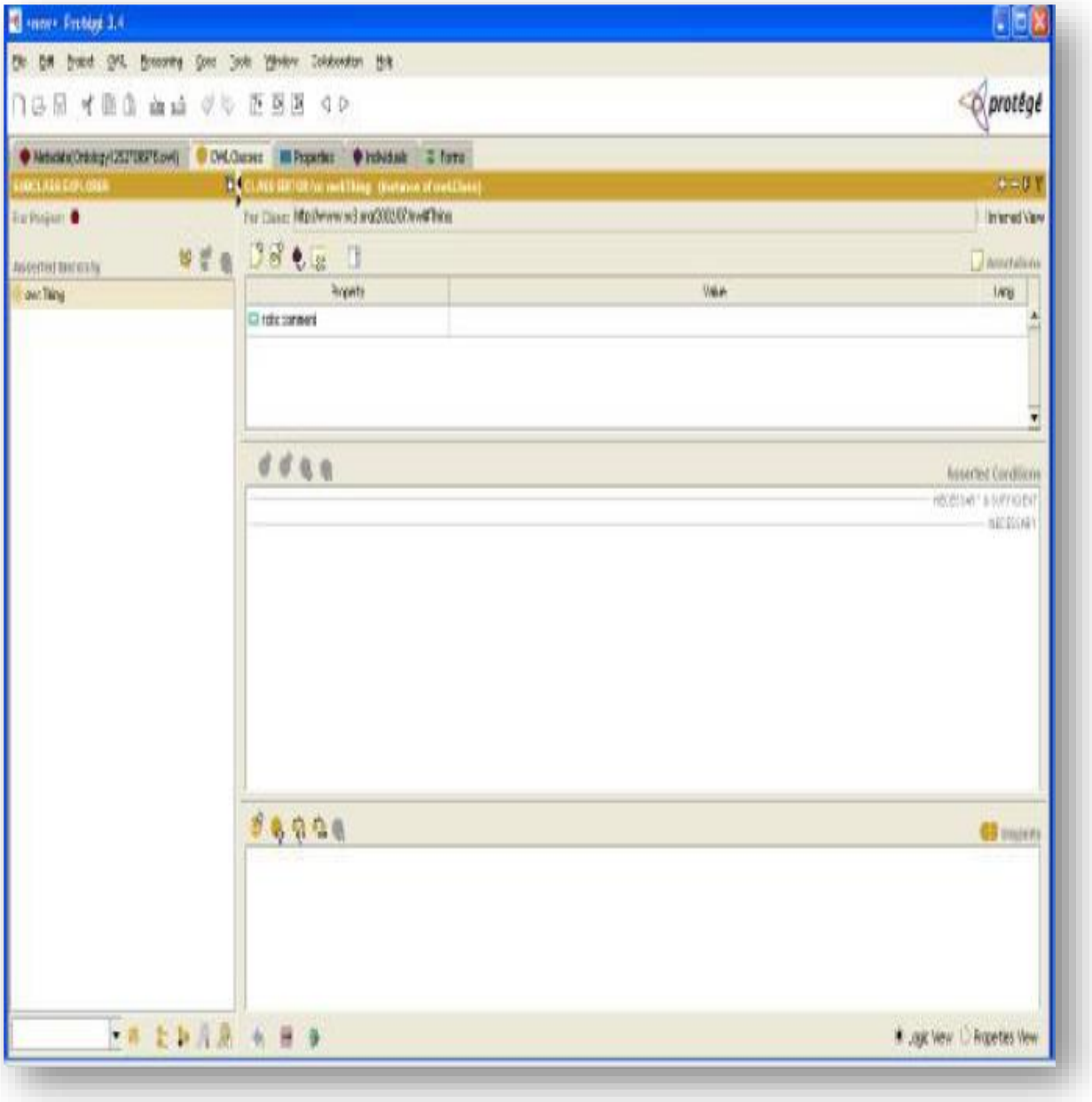


Figure 3.9. Interface de Protégé OWL

3.6.4.1. Définition de la hiérarchie des classes

Nous commencerons tout d'abord par la création des concepts spécifiés dans l'étape de conceptualisation. PROTEGE OWL nous offre également un moyen de construire la hiérarchie de concepts, la Figure 3.10 présente comment-on procède pour créer un concept.

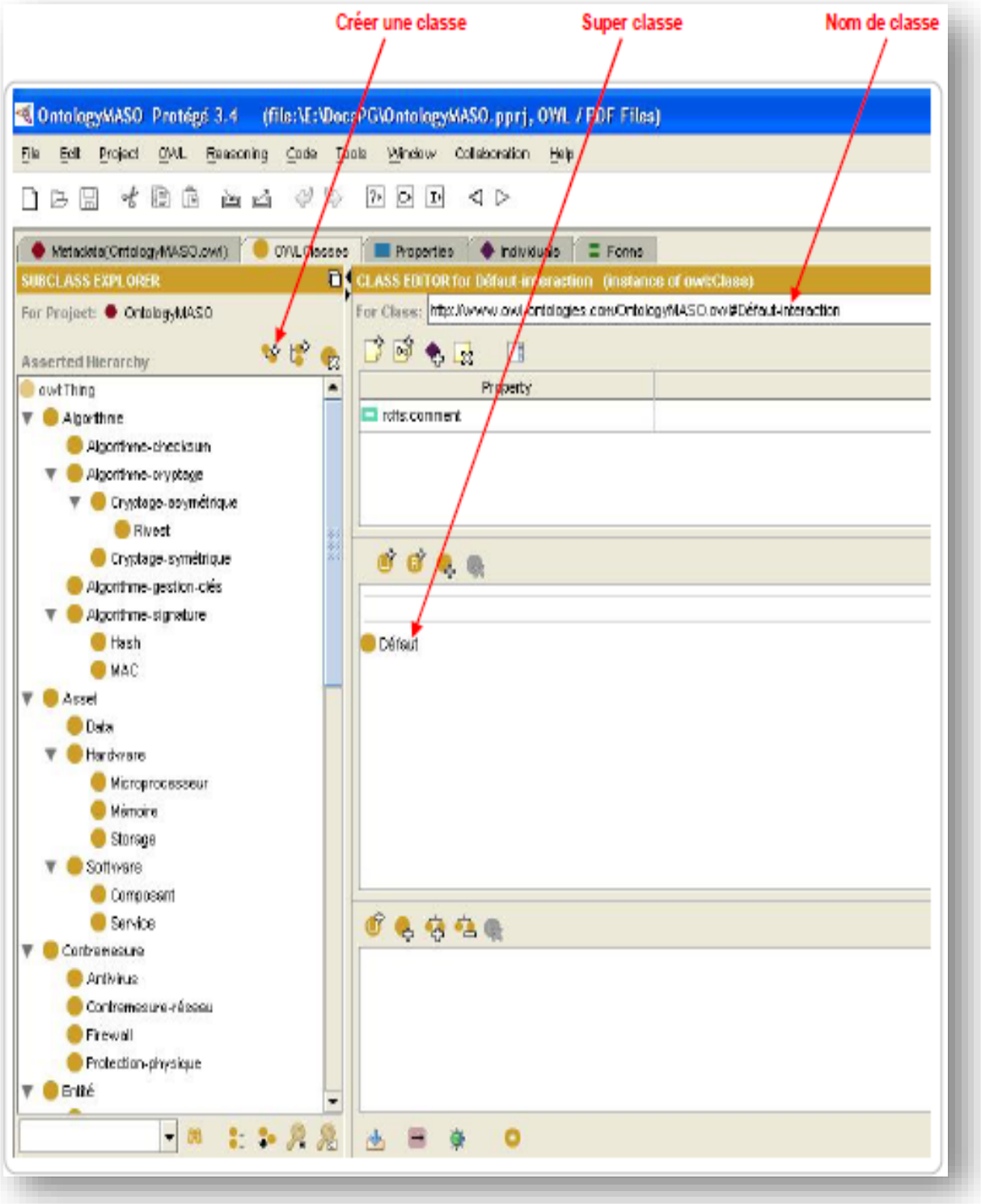


Figure 3.10. Création des classes

La figure ci-dessous montre la r*eprésentation hiérarchique de notre ontologie* sous protégé.

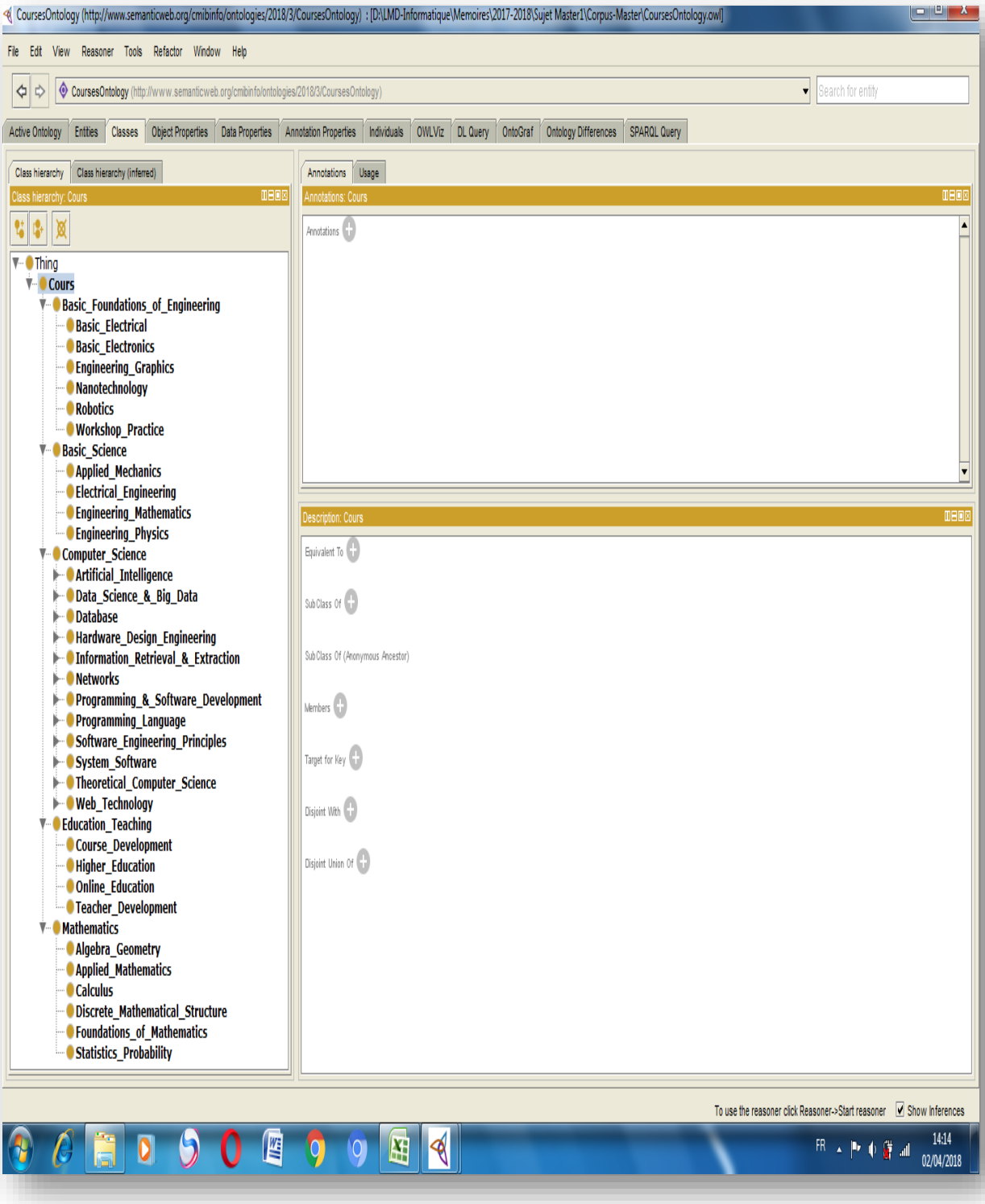


Figure 3.11. Représentation hiérarchique de l'ontologie des cours.

3.6.4.2. Définition des propriétés :

Après avoir construit les classes, nous allons maintenant créer les propriétés pour chacun d'eux, les attributs vont être créés sous PROTEGE OWL par *'dataTypeProperty'* et les relations par *'objectProperty'*. La Figure 3.12 montre les potentialités de PROTEGE pour la création des propriétés.

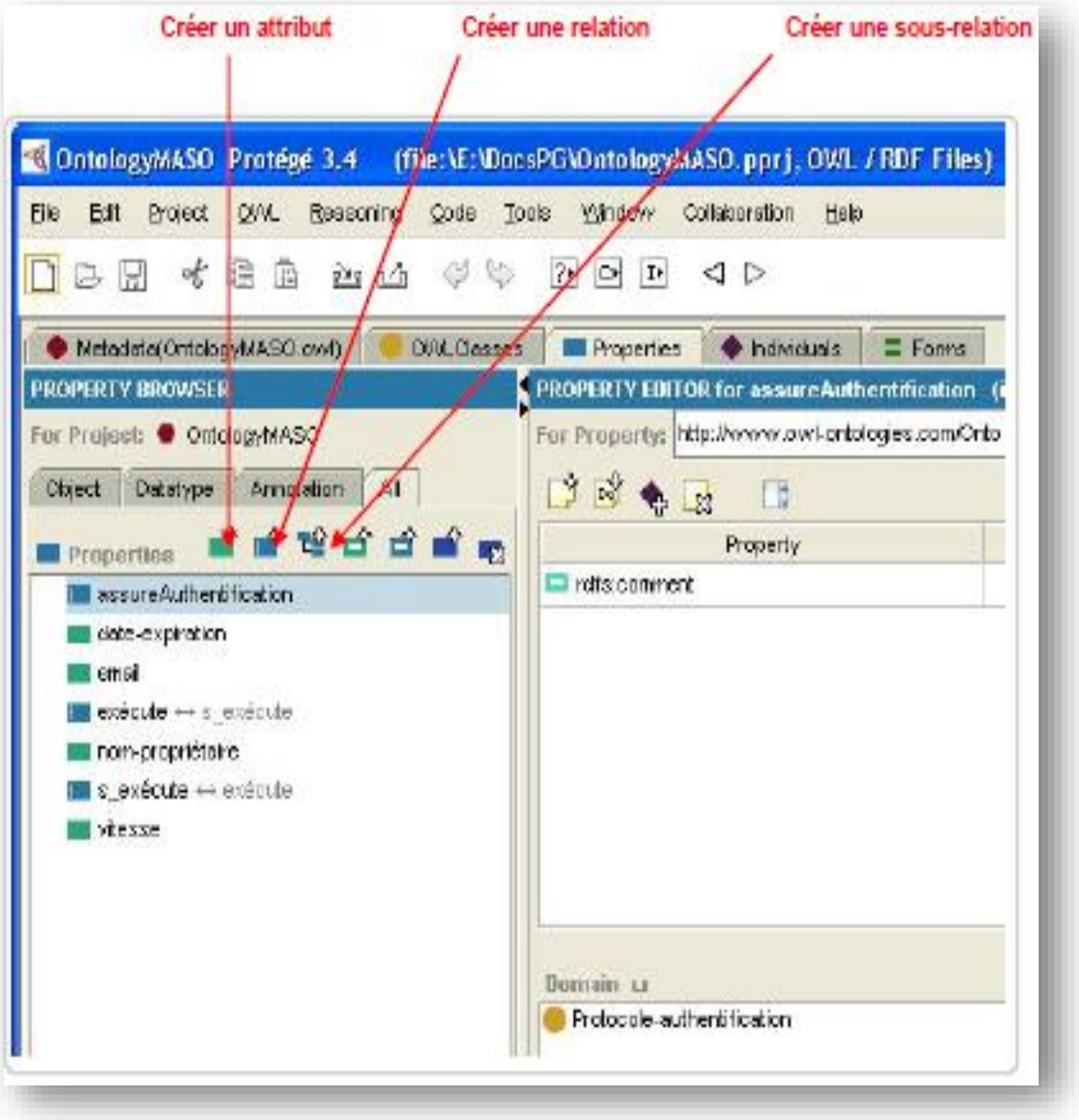


Figure 3.12. Création des propriétés pour une classe

La Figure 3.13 montre comment créer une relation et spécifier son nom, type, domaine et co-domaine.

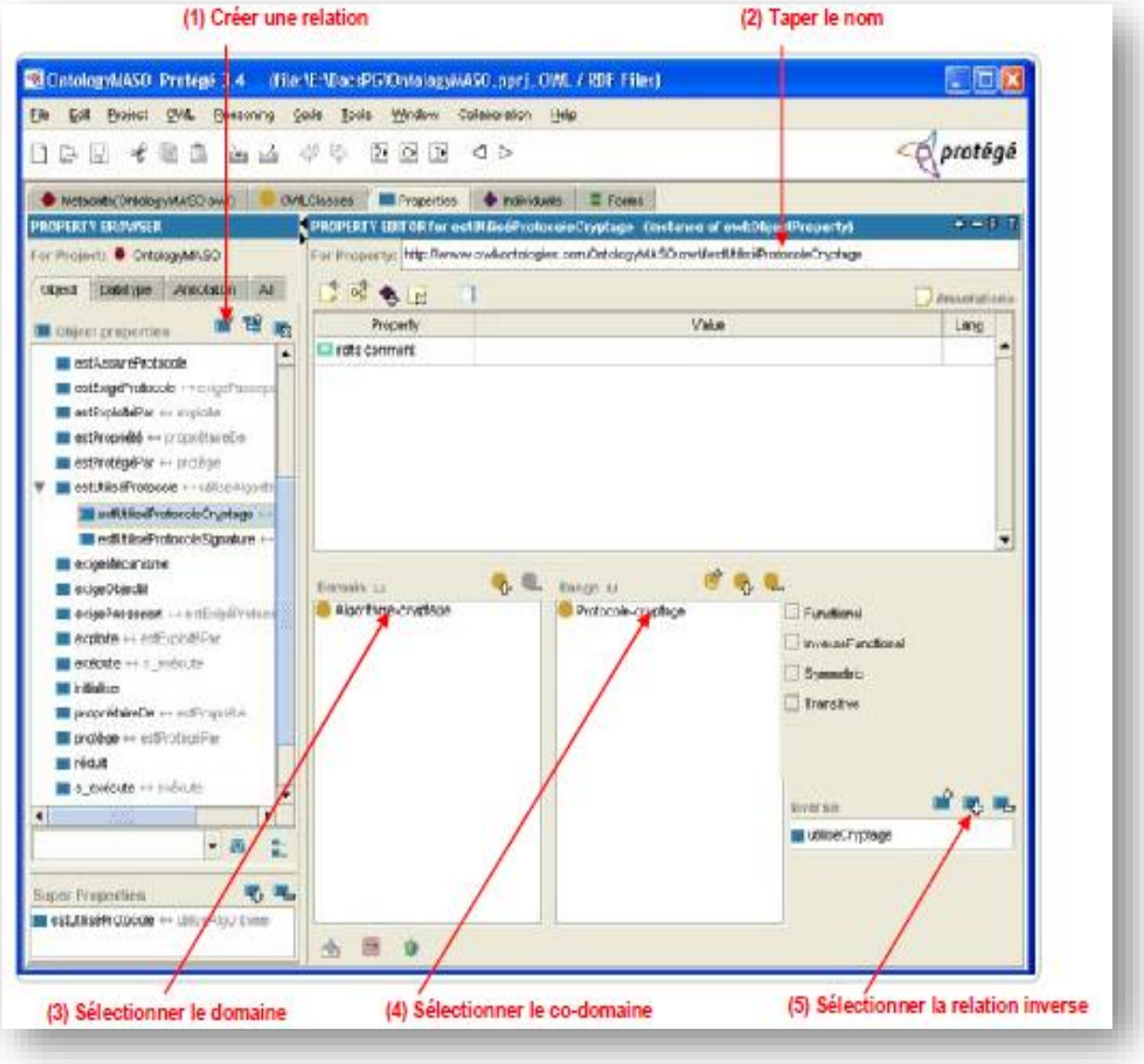


Figure 3.13. Création d'une relation

3.6.4.3. Définition des instances :

La dernière étape consiste à créer les instances des classes dans la hiérarchie. Définir une instance individuelle d'une classe exige

1. choisir une classe
2. créer une instance individuelle de cette classe
3. la renseigner avec les valeurs des attributs.

Par exemple, la Figure 3.14 montre la création d'un individu de la classe Propriétaire.

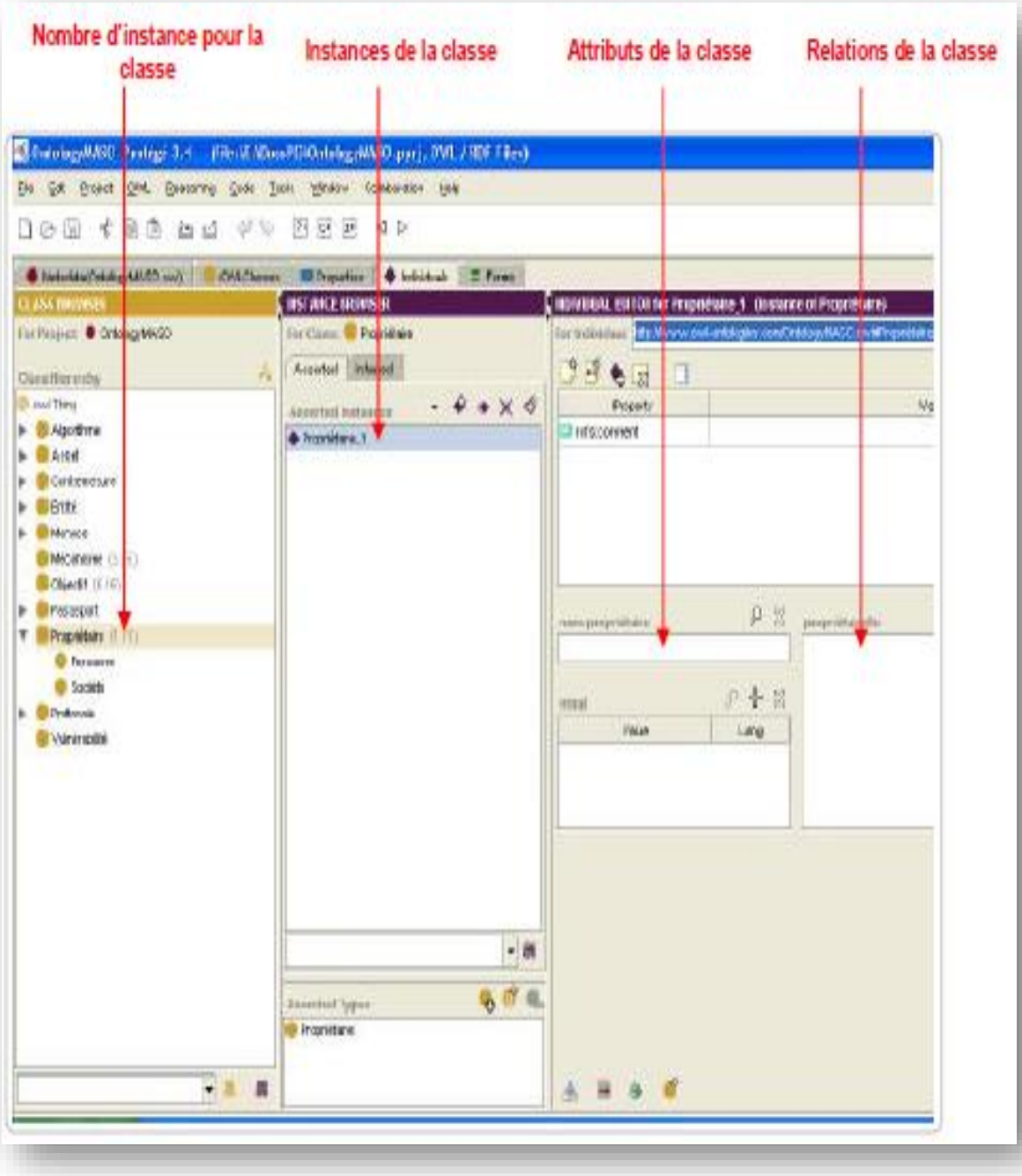


Figure 3.14. Création des instances

3.7. Présentation et évaluation de l'application

Après avoir détaillé la partie conception de notre système, nous consacrons cette partie à sa présentation et son évaluation. Nous commençons d'abord par décrire la base de donnes utilisée, le calcul de prédictions, les différentes copies d'écran de chaque étape de notre système et nous enchaînons en présentant son évaluation.

3.7.1. Construction des tables

Nous présentons dans ce qui suit quelques tables utilisées qui nous ont aidés dans la réalisation et l'évaluation de notre système.

La table **TCours (ID-Cours, Titre, Auteur, Année) :** contient la liste tous les cours.

La table **TCatégories (Id-Cat, Nom-Cat) :** contient la liste de toutes les categorie de cours.

La table **TUtilisateurs (ID-Util, Nom, Prénom, Age, Mot-Passe) :** contient la liste de tous les utilisateurs inscrit dans notre systeme.

La table **TPossede (ID-Cours, ID-Cat) :** contient chacun des cours avec les catégories correspondant.

La table **TEvaluer (ID-Util, ID-Cours, Note, Prédiction)** contient pour chaque utilisateur les cours consultés avec leur notes et leurs prédiction.

La table **TProfil (ID-Util, ID-Cat, Préférence) :** contient tous les utilisateurs et toutes les catégories avec leurs scores ou préferences.

La signification des différents champs des tables utilisées est donnée dans le tableau ci-dessous

Champs	Signification
ID-Util	Identificateur de l'utilisateur
ID-Cours	Identificateur de cours
ID-Cat	Identificateur de categories
Nom-Cat	Nom de catégories de cours
Note	Note donnée par un utilisateur à un cours
Prédiction	Prédiction calculée
Préférence	Préférence ou score d'une catégorie pour un utilisateur

La valeur de la préférence d'une catégorie pour un utilisateur est obtenue par le calcul de la moyenne des notes de cette catégorie. A ce stade de calcul, un changement d'échelle savere nécessaire, car les score (préférence) doivent etre compris entre '0' et '1'. Nous utilisons, pour cela, la formule (X-X0)/(X1-X0), avec X0=1 et X1=5. Par exemple la préférence p pour un utilisateur donné pour une catégorie donné qui est égale à 4,5 sera transformée en (4,4-1)/(5-1)=0,85

3.7.2. Calcul de la prédiction

Dans notre système il faut comparer entre les notes réelles données par l'utilisateur à un cours et la prédiction calculée par le prototype Pour le calcul de la similarité numérique, nous utilisons la mesure de Pearson, et pour le calcule de la similarité sémantique nous utilisons la mesure de wu-palmer. Enfin nous combinons les deux mesure afin d'avoir des meilleurs résultats.

3.7.2.1. Similarité numérique

Pour calculer la prédiction dans cette mesure, nous avons besoin de deux vecteurs, le premier correspond aux catégories de cours à prédire, le deuxième vecteur contient les préférences extraites de la table TProfil. Le tableau 3.2 est une illustration du premier vecteur pour l'utilisateur ***User1*** et le tableau 3.2 est une illustration le deuxième vecteur correspond à la pondération de categoroes associé à un cours ayant l'identificateur ***Cours1***.

Tableau 3.2. Illustrations du vecteur utilisateur pour le calcul de la similarité numérique

User / Catég.	NT	AI	CNT	ML	PSD	WD	….
User1	**0,35**	**0,52**	**0,85**	**0,45**	**0,75**	**0,64**	**….**

Tableau 3.3. Illustration du vecteur Cours pour le calcul de la similarité numérique

Cours / Catég.	NT	AI	CNT	ML	PSD	WD	….
Cours1	**1**	**0**	**1**	**0**	**1**	**0**	**….**

Dans cet exemple la similarité numérique (mesure de Pearson) est donné par : $\boldsymbol{Sim_{num}(User1, Cours1) = 0,329}$

3.7.2.2. Similarité sémantique

Le tableau 3.3 montre exemple de calcul de la similarité sémantique entre un cours dont les catégories sont **AI**, **ML**, **NLP** et un utilisateur qui préfère les catégories **AI** et **WD**. La mesure sémantique utilisée dans cet exemple est la mesure de Wu-Palmer

Tableau 3.4. Illustration d'une matrice pour le calcul de la similarité sémantique

User1 / Cours1	AI	ML	NLP
AI	**1**	**0.57**	**0.57**
WD	**0.40**	**0.33**	**0.33**

Le poids de chaque matrice est calculé par la formule suivante :

$$Sim(V_1, V_2) = \frac{1}{M * N} \sum_{j=1}^{M} \sum_{i=1}^{N} Sim_{ij}$$

Dans cet exemple la similarité sémantique est donné par : $\boldsymbol{Sim_{sem}(User1, Cours1) = 0,533}$

3.2.3 Similarité globale

Dans ce type de similarité, nous avons effectué une combinaison linéaire entre la mesure sémantique de wu-palmer et la mesure numérique de Pearson. En utilisant la formule suivante:

$\mathrm{Sim_{global}} = \alpha * \mathrm{Sim_{Sem}} + \beta * \mathrm{Sim_{Num}}$, Avec : $\alpha + \beta = 1$

D'après nos expérimentations $\alpha = 0{,}75$ et $\beta = 0{,}25$ conduisent à de bonnes recommandations aux utilisateurs.

D'après les exemples précédents la similarité globale est : $\boldsymbol{Sim_{glob}(User1, Cours1) = 0.482}$.

3.8. Interfaces de notre application

Dans cette partie de ce chapitre, nous allons présenter les différentes formes de notre application.

Dans cette section nous nous intéressons seulement la partie de l'application « session utilisateur » qui concerne l'utilisateur de cette application. Par contre pour la partie administration « session administrateur » qui concerne la mise à jours de la base de données, la création de l'ontologie de cours et l'inscription des utilisateurs nous n'avons pas créée d'interfaces qui réalisent ces opérations.

Pour la création et la mise à jour des différentes tables de notre base de données nous pouvons le faire soit à travers l'interface de NetBeans conçue pour se connecter à la DB, mais la plupart des commandes SQL doivent être introduite dans leur entièreté (voir figure 3.15)ou par

PhpMyadmin qui est une interface de gestion de base de données MySQL sur un serveur PHP. PhpMyadmin sert donc à créer très facilement une base de données, et y insérer des informations sans connaissances en SQL qui est pourtant le langage incontournable de la gestion de base de données (voir figure 3.16).

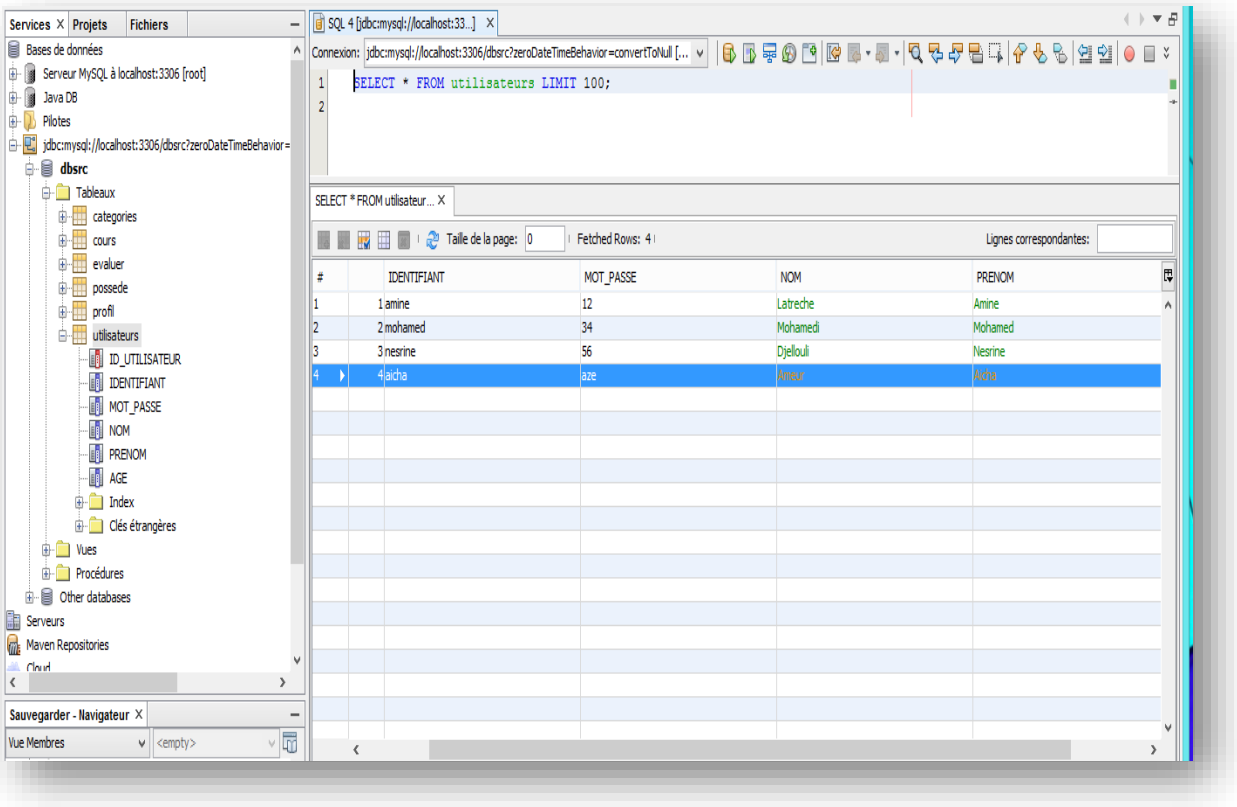


*Figure 3.15. Exemple de creation d'une BD dans*NetBeans

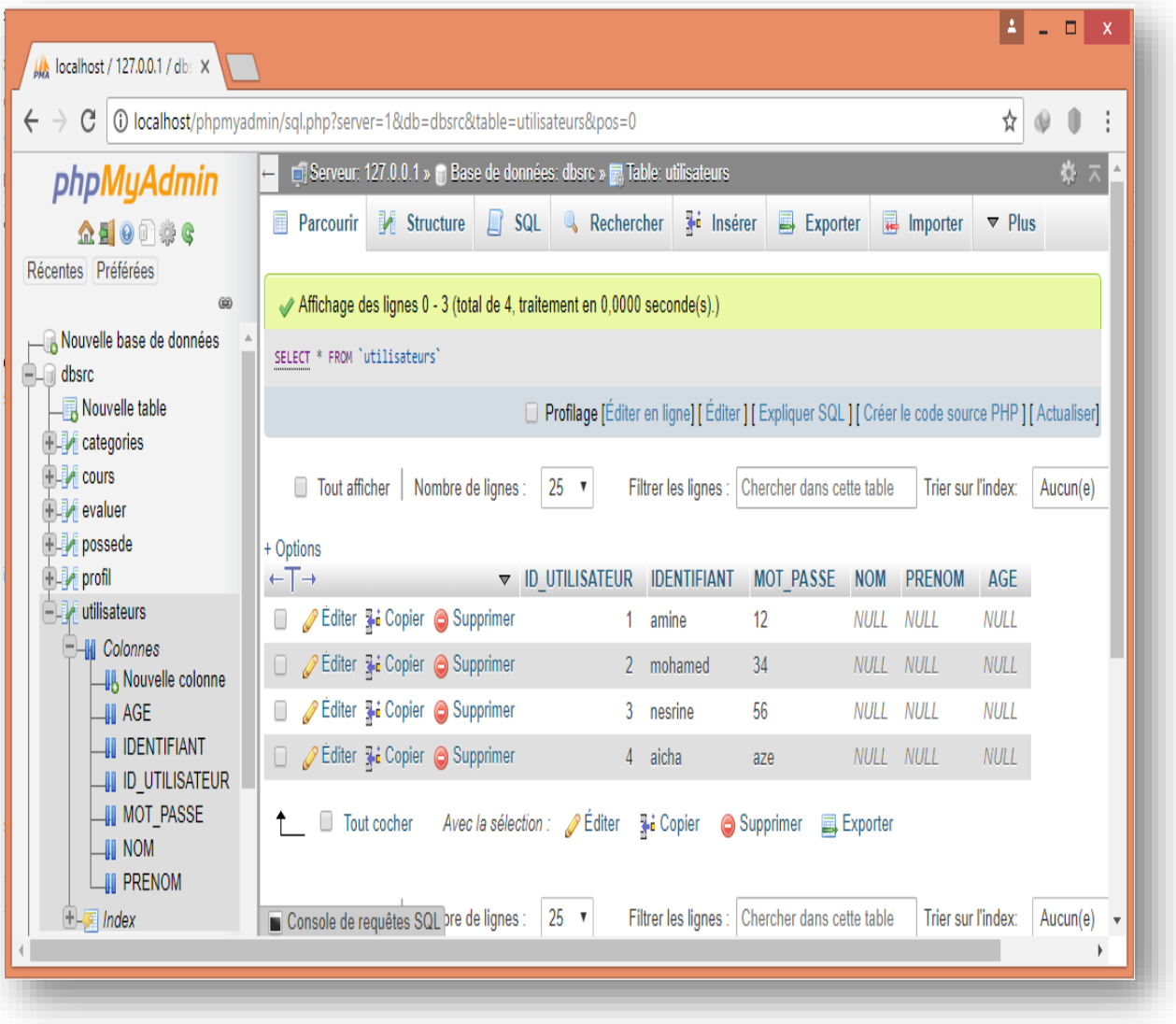


*Figure 3.16. Exemple de création d'une BD dans*phpMyAdmin

Pour la création de notre ontologie de cours nous utilisons tous simplement l'éditeur d'ontologie « Protège » (Voir figure *3.11.*)

Pour la « session utilisateur », la fenêtre de la figure 3.17 s'affiche pour l'authentification de l'utilisateur. Il faut donc entrer l'identifiant ou le pseudonyme ainsi que le mot de passe correspondant

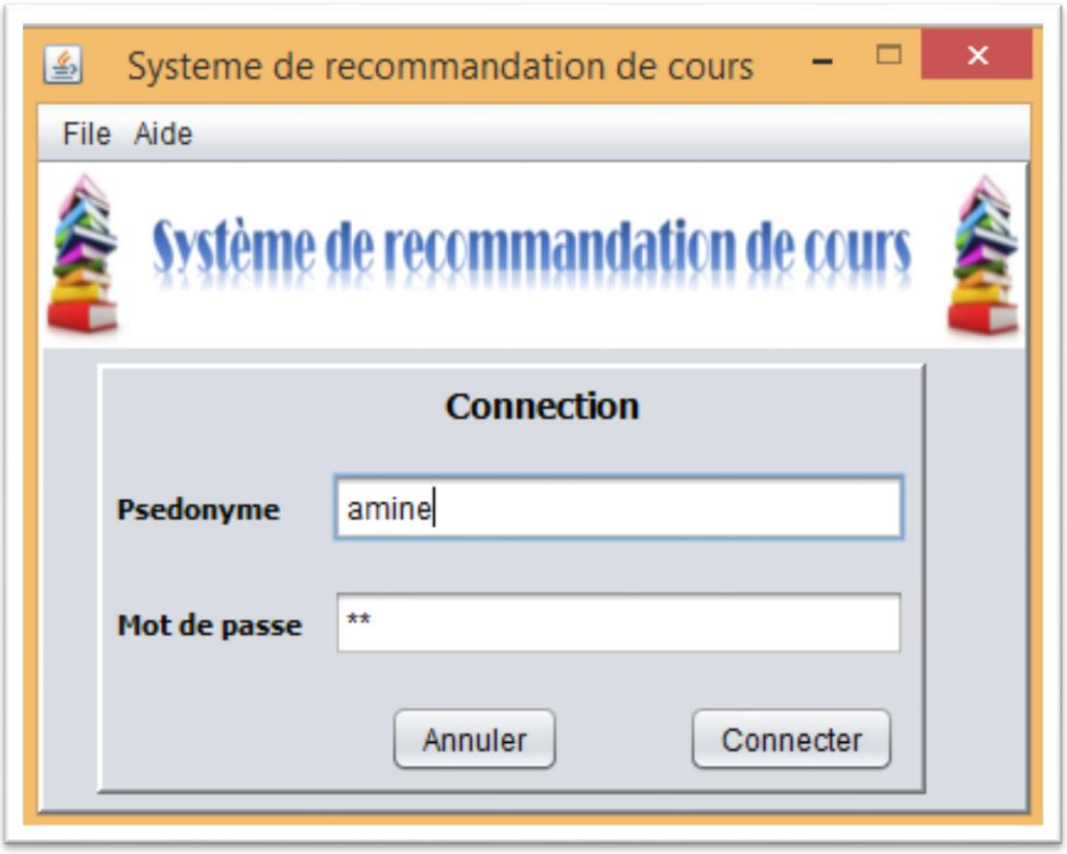


Figure 3.17. Authentification de l'utilisateur

Après l'authentification de l'utilisateur une fenêtre s'affiche (Figure 3.16), qui contient sur la partie gauche la listes des cours les pertinents concernant un utilisateur donné ainsi que les paramètres concernant le calcul de similarité entre le profil utilisateur et descripteur de cours (profil cours). Alors que la partie droite affiche les informations concernant le cours sélectionné, ainsi que la liste des catégories concernant cet utilisateur.

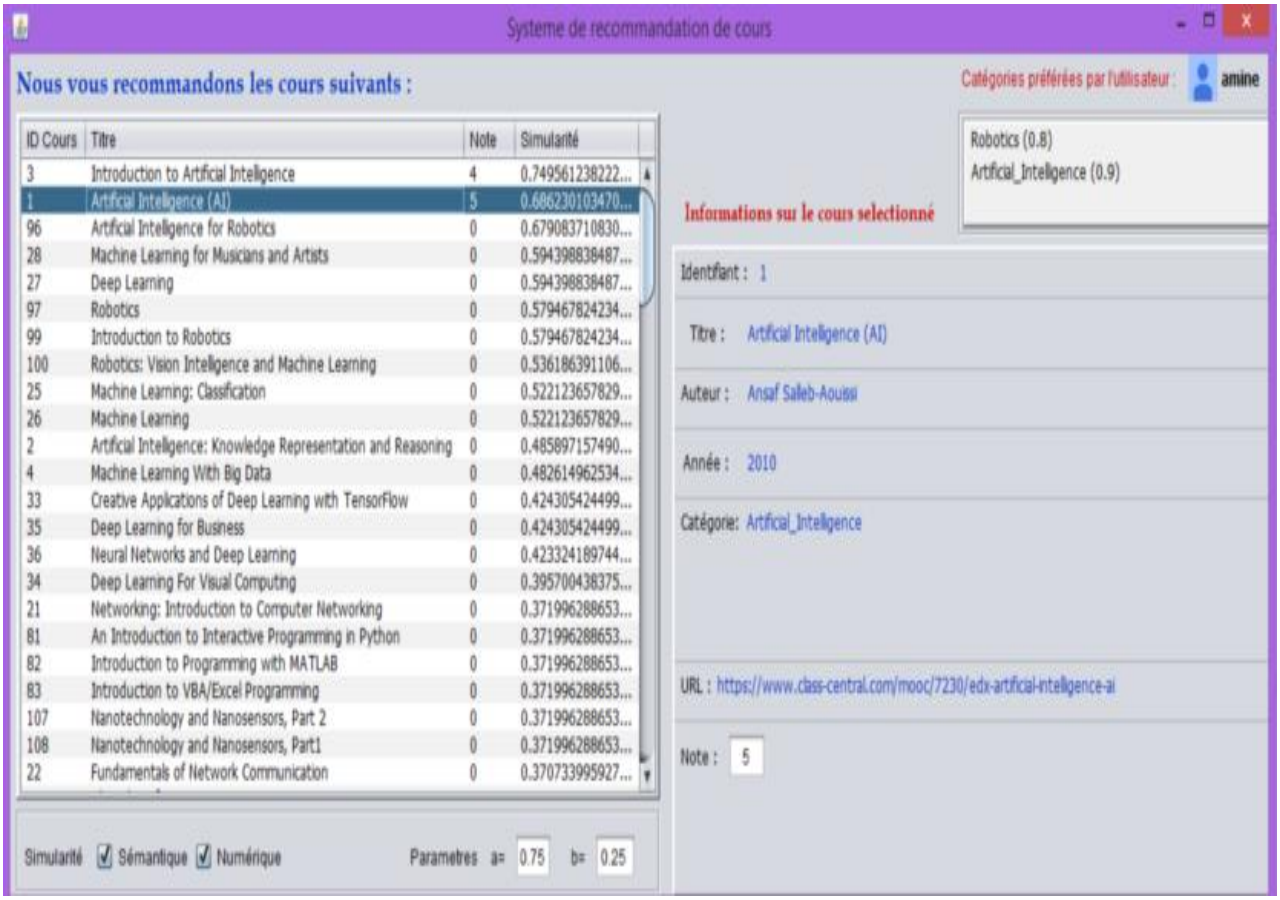


Figure 3.18. Liste des cours pertinents

L'utilisateur peut intervenir sur cette fenêtre de différentes façons :

- Consulter la liste des cours pertinents proposée.
- Noter un ou plusieurs cours sélectionnés.
- Choisir la méthode du calcul de similarité (numérique, sémantique ou la combinaison des deux).
- Modifier les paramètres utilisés dans le calcul de similarité globale.

3.9. Conclusion

Dans ce chapitre, nous avons exposé et présenté les différentes phases suivies pour la conception et la réalisation de notre système de recommandation de cours à base d'ontologie. En effet, nous avons explicité le modèle de données utilisé, la méthode de création des profils utilisateur et les descripteurs de cours. Nous avons détaillé les différents modules constituant notre système de recommandation et nous avons présenté la construction de l'ontologie de cours utilisé dans notre système. Ensuite nous avons présenté les aspects techniques utilisés dans notre travail. Nous avons commencé par présenter les aspects d'implémentation qui sont

utilisés dans notre travail. L'implantation repose essentiellement sur le langage JAVA avec NetBeans comme environnement de développement, Jena API pour exploiter et manipuler l'ontologie, le langage OWL pour la représentation de l'ontologie et l'éditeur Protégé pour la création et la mise à jour de l'ontologie. Ensuite, nous avons présenté et commenter les différentes partie de notre application. Avec les premiers tests notre système a réalisé des résultats encourageant.

Conclusion et perspectives

Les systèmes de recommandation automatique sont devenus, a l'instar des moteurs de recherche, un outil incontournable pour tout site Web focalise sur un certain type d'articles disponibles dans un catalogue riche, que ces articles soient des objets, des produits culturels (livres, films, morceaux de musique, etc.), des éléments d'information (news) ou encore simplement des pages (liens hypertextes). L'objectif de ces systèmes est de sélectionner, dans leur catalogue, les items les plus susceptibles d'intéresser un utilisateur particulier.

Les systèmes de recommandation qui donnent de l'importance aux préférences des utilisateurs, dans le but de leur proposer des ressources à acheter ou à consulter répondent au mieux à leurs besoins. Ils sont devenus des efficaces dans le commerce électronique, la recherche documentaire, le tourisme, etc., en fournissant des suggestions pertinentes au sein d'une grande masse d'informations. L'intégration des ontologies dans les systèmes de recommandation classiques, permet d'améliorer les performances de ces derniers. Nous avons [2]présentés, dans ce mémoire, une nouvelle approche qui permet de recommander de cours à un utilisateur en se basant sur une ontologie. Elle est générale dans le sens où elle peut être appliquée pour la recommandation de contenus en général (livre, Url, article, produit, film, chanson, etc.). Dans cette approche, l'ontologie est utilisé pour réaliser la similarité sémantique entre le profil utilisateur et les descripteurs de cours, et qui est combinée avec similarité numérique, ce qui permet d'améliorer la précision de la recommandation et ainsi mieux répondre aux exigences des utilisateurs. Dans notre travail, nous avons conçu et crée l'ontologie de cours, qui structure sous forme de graphes les différentes relations entre les catégories de cours. Pour ce faire, nous avons eu recours à la méthodologie Methontology. Bien évidemment, nous étions guidés dans notre travail par plusieurs principes largement acceptés par la communauté des ontologistes. Une fois l'ontologie conceptuelle mise au propre, nous avons passé à sa formalisation en nous appuyant sur la logique de description et son opérationnalisation avec l'outil PROTÉGÉ.

Ce travail est loin d'être terminé. Des perspectives sont à envisager, comme :

- Introduire le contexte dans les systèmes de recommandation utilisant le filtrage hybride.

Combiner deux type de contexte à savoir extrinsèque et le contexte intrinsèque afin d''affiner la liste des cours à recommander.

Bibliographie

[1] Malone, T., Brobst, S., Cohen, S., Grant, K., and Turbak, F. (1987). Intelligent information des systemes de partage. In *Communications of the ACM*, volume 30, pages 390–402.

[2] Resnick, P. and Varian, H. (1997). Recommender systems. In Communications of the ACM, volume 40, pages 56–58.

[3] Maes, P. and Shardanand, U. (1995). Social information filtering : algorithms for automating "word of mouth". In the SIGCHI conference on Human factors in computing systems, Denver, Colorado, United States. ACM Press/Addison-Wesley Publishing Co.

[4] Burke, R. (2002). Hybrid recommender systems : Survey and experiments. User Modeling and User-Adapted Interaction, 12(4) :331–370.

[5] A. T. NGUYEN. COCoFil2 : Un nouveau système de filtrage collaboratif basé sur le modèle des espaces de communautés. Thèse. Université joseph Fourier Grenoble I. Novembre 2006.

[6] Adomavicius, G. and Tuzhilin, A. (2005). Toward the next generation of recommender systems : A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, 17(6) :734–749.

[8] Rao, N. and Talwar, V. (2008). Application domain and functional classification of recommender systems a survey. Desidoc journal of library and information technology, 28(3) :17–36.

[9] Nguyen, A. T. (2006). COCoFil2 : Un nouveau système de filtrage collaboratif basé sur le modèle des espaces de communautés. PhD thesis, universite Joseph Fourier-Grenoble I.

[10] Arnautu, O. R. (2012). Mures : Un systeme de recommandation de musique. Master's thesis, La Faculte des arts et des sciences Universite de Montreal.

[11] T. Slimani, B. Ben Yaghlane et K. Mellouli. Une extension de mesure de similarité entre les concepts d'une ontologie. 4rth International Conference: Sciences of Electronic, Technologies of Information and Telecommunications. Mars 2007.

[12] Z. Wu et M. Palmer. Verb semantics and lexical selection. Conference. In Proceedings of the 32nd Annual Meeting of the Associations for Computational Linguistics. Pages 133-138. 1994.

[13] P. Resnik. Semantic Similarity in a Taxonomy: An Information-Based Measure and its Application to Problems of Ambiguity in Natural Language. Journal of Articial Intelligence Research 11. Pages 95-130. 1999.

[14] D. Lin. An Information-Theoretic Definition of similarity. In Proceedings of the Fifteenth International Conference on Machine Learning. Pages 296 - 304. 1998.

[15] J. Jiang et D. Conrath. Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings of the 10th International Conference on Research in Computational Linguistics, Taiwan. 1998.

[16] Neches, R; Fikes, R E; Finin, T; Gruber, T R; Senator, T; Swartout, W R. (1991). Enabling technology for knowledge sharing. AI Magazine , 12 (3), 36–56.

[17] Gruber, T. R. (1993a). A translation approach to portable ontology specification. Knowledge Acquisition , 5 (2), 199–220.

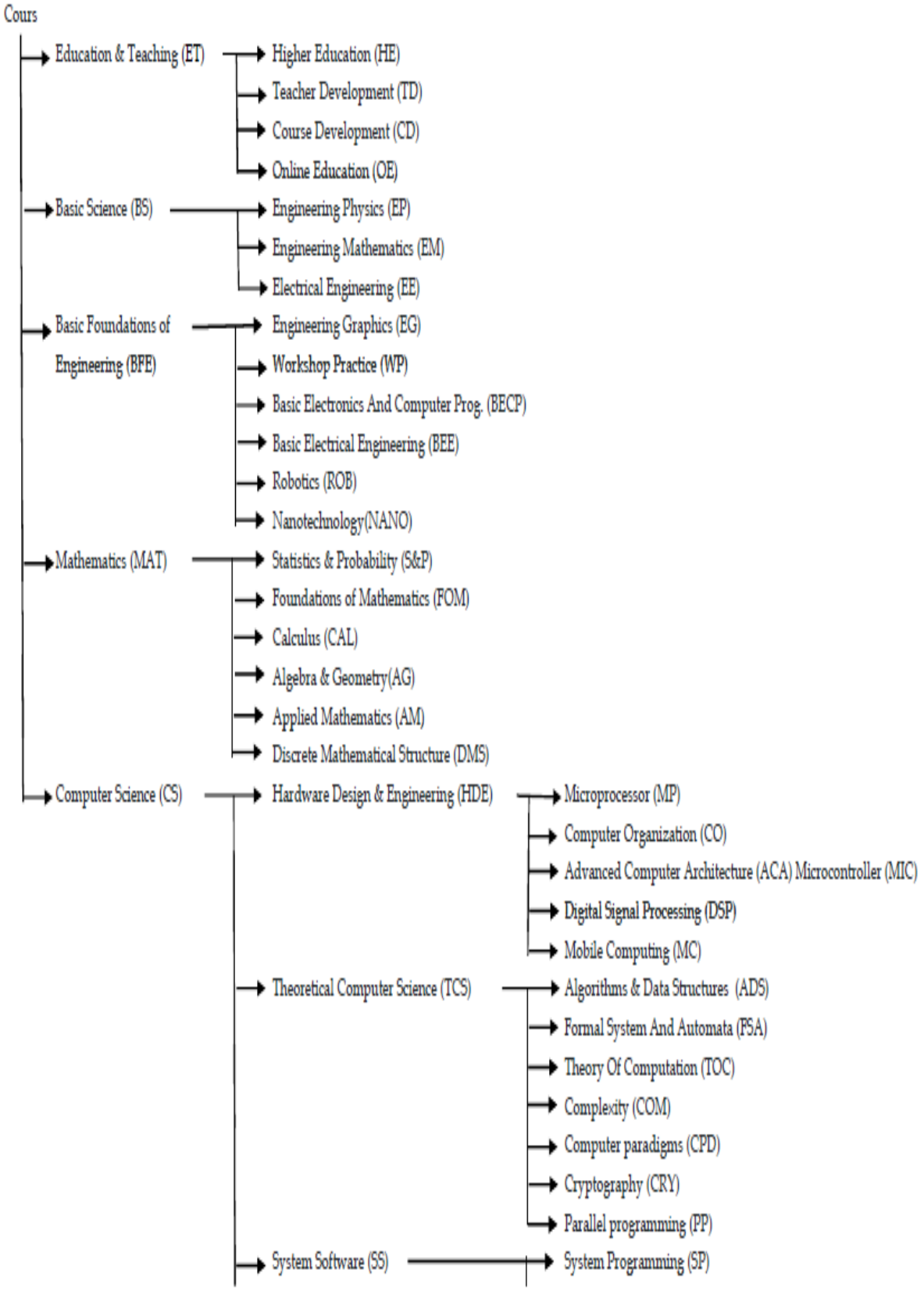
[18] Borst, W. N. (1997). Construction of Engineering Ontologies. Centre for Telematica and Information Technology, University of Tweenty, Enschede, The Netherlands.

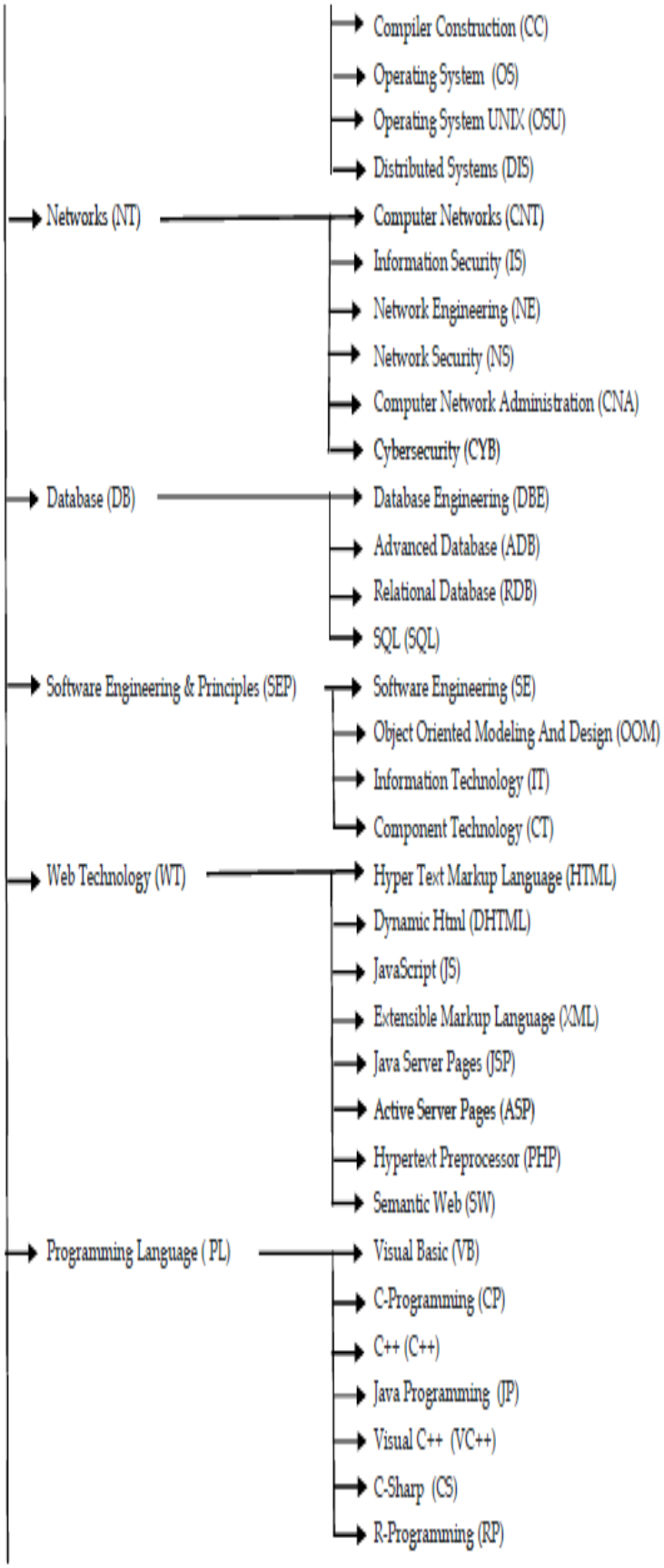
[19] Studer, R., Benjamins, V. R., & Fensel, D. (1998). Knowledge Engineering: Principles and Methods. IEEE Transactions on Data and Knowledge Engineering , 25 (1-2), 161-197.

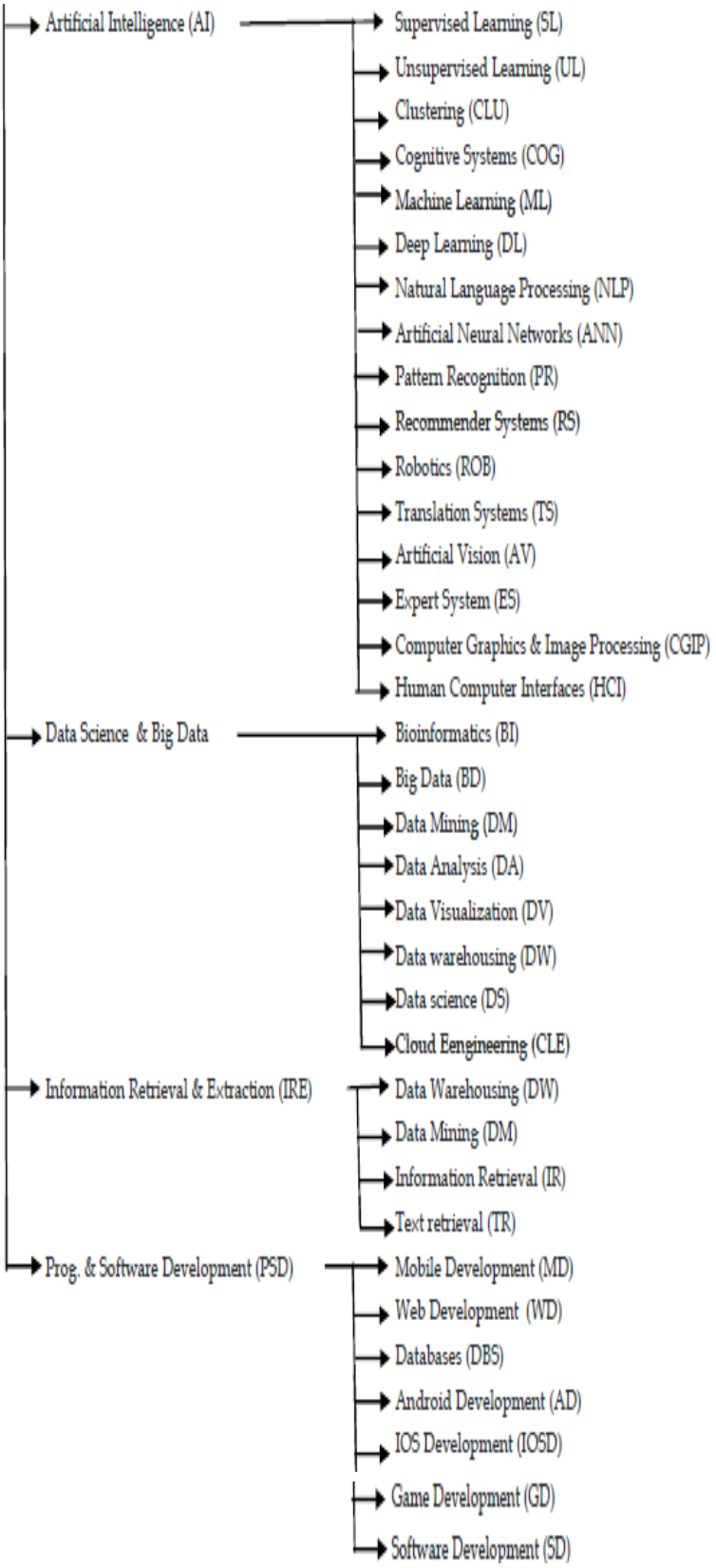
[20] Guarino, N. (1998). Formal Ontology in Information Systems. In N. Guarino (Ed.), 1st International Conference on Formal Ontology in Information Systems (FOIS'98) (pp. 3-15). IOS Press.

[21] Uschold, M., & Grüninger, M. (1996). Ontologies: Principles, Methods and Applications. Knowledge Engineering Review , 11 (2), 93–155.

Annexe A


Cours
Education & Teaching (ET)
Higher Education (HE)
Teacher Development (TD)
Course Development (CD)
Online Education (OE)
Basic Science (BS)
Engineering Physics (EP)
Engineering Mathematics (EM)
Electrical Engineering (EE)
Basic Foundations of Engineering (BFE)
Engineering Graphics (EG)
Workshop Practice (WP)
Basic Electronics And Computer Prog. (BECP)
Basic Electrical Engineering (BEE)
Robotics (ROB)
Nanotechnology(NANO)
Mathematics (MAT)
Statistics & Probability (S&P)
Foundations of Mathematics (FOM)
Calculus (CAL)
Algebra & Geometry(AG)
Applied Mathematics (AM)
Discrete Mathematical Structure (DMS)
Computer Science (CS)
Hardware Design & Engineering (HDE)
Microprocessor (MP)
Computer Organization (CO)
Advanced Computer Architecture (ACA) Microcontroller (MIC)
Digital Signal Processing (DSP)
Mobile Computing (MC)
Theoretical Computer Science (TCS)
Algorithms & Data Structures (ADS)
Formal System And Automata (FSA)
Theory Of Computation (TOC)
Complexity (COM)
Computer paradigms (CPD)
Cryptography (CRY)
Parallel programming (PP)
System Software (SS)
System Programming (SP)


Compiler Construction (CC)
Operating System (OS)
Operating System UNIX (OSU)
Distributed Systems (DIS)
Networks (NT)
Computer Networks (CNT)
Information Security (IS)
Network Engineering (NE)
Network Security (NS)
Computer Network Administration (CNA)
Cybersecurity (CYB)
Database (DB)
Database Engineering (DBE)
Advanced Database (ADB)
Relational Database (RDB)
SQL (SQL)
Software Engineering & Principles (SEP)
Software Engineering (SE)
Object Oriented Modeling And Design (OOM)
Information Technology (IT)
Component Technology (CT)
Web Technology (WT)
Hyper Text Markup Language (HTML)
Dynamic Html (DHTML)
JavaScript (JS)
Extensible Markup Language (XML)
Java Server Pages (JSP)
Active Server Pages (ASP)
Hypertext Preprocessor (PHP)
Semantic Web (SW)
Programming Language (PL)
Visual Basic (VB)
C-Programming (CP)
C++ (C++)
Java Programming (JP)
Visual C++ (VC++)
C-Sharp (CS)
R-Programming (RP)


Artificial Intelligence (AI)
Supervised Learning (SL)
Unsupervised Learning (UL)
Clustering (CLU)
Cognitive Systems (COG)
Machine Learning (ML)
Deep Learning (DL)
Natural Language Processing (NLP)
Artificial Neural Networks (ANN)
Pattern Recognition (PR)
Recommender Systems (RS)
Robotics (ROB)
Translation Systems (TS)
Artificial Vision (AV)
Expert System (ES)
Computer Graphics & Image Processing (CGIP)
Human Computer Interfaces (HCI)
Data Science & Big Data
Bioinformatics (BI)
Big Data (BD)
Data Mining (DM)
Data Analysis (DA)
Data Visualization (DV)
Data warehousing (DW)
Data science (DS)
Cloud Eengineering (CLE)
Information Retrieval & Extraction (IRE)
Data Warehousing (DW)
Data Mining (DM)
Information Retrieval (IR)
Text retrieval (TR)
Prog. & Software Development (PSD)
Mobile Development (MD)
Web Development (WD)
Databases (DBS)
Android Development (AD)
IOS Development (IOSD)
Game Development (GD)
Software Development (SD)