SPIM

Thèse de Doctorat

utbm
université de technologie
Belfort-Montbéliard

école doctorale **sciences pour l'ingénieur et microtechniques**

UNIVERSITÉ DE TECHNOLOGIE BELFORT-MONTBÉLIARD

Un système multi-agents pour la gestion des connaissances hétérogènes et distribuées

INAYA LAHOUD

2013

RAPPORTEURS	▪ DJAMAL BENSLIMANE	Professeur des Universités, Université Claude Bernard, Lyon I
	▪ PIERRE-ALAIN MÜLLER	Professeur des Universités, Université de Haute-Alsace, Mulhouse
EXAMINATEURS	▪ NADA MATTA	Enseignants-Chercheur HDR, Université de Technologie de Troyes
	▪ VINCENT CHEVRIER	Maître de Conférences HDR, Université de Lorraine – Vandoeuvre-Lès-Nancy
	▪ SAMUEL GOMES	Professeur des Universités Université de Technologie de Belfort-Montbéliard
DIRECTEUR DE THESE	▪ VINCENT HILAIRE	Professeur des universités, Université de Technologie de Belfort-Montbéliard
CO-DIRECTEURS	▪ DAVY MONTICOLO	Maître de conférences, Université de Lorraine, Nancy

Remerciements

Cette thèse doit beaucoup aux nombreuses personnes qui m'ont encouragé, soutenu et conforté au long de toutes ces années. Qu'elles trouvent dans ce travail l'expression de mes plus sincères remerciements.

Je remercie mon directeur M. Vincent Hilaire pour avoir accepté encadrer cette thèse et pour son suivi toute la durée de cette thèse.

Mes sentiments chaleureux de gratitude et de remerciements à M. Davy Monticolo, mon encadrant de thèse, qui a fourni une assistance et des conseils précieux tout au long de cette expérience avec beaucoup de patience, de savoirs et d'écoute.

Je renouvelle mes remerciements à mon directeur et mon encadrant pour leur aide précieuse dans la relecture et la correction de ma thèse. La justesse de ses critiques a été très constructive et utile. Je leur exprime ma très profonde gratitude

Je tiens à remercier le laboratoire M3M et spécialement Samuel Gomes pour son confiance en m'engageant dans un contrat de collaboration dans le cadre du projet ADN (Alliance de Données Numériques) et qui était trop utile pour mieux accomplir mes travaux de recherche.

J'aimerais remercier l'équipe MOS du laboratoire M3M (Béatrice Rossez, Pascal Aldinger, Nizar Harb, Salima TIE-BI, Anh Tuan TA, Homam Issa), l'équipe INCIS du laboratoire M3M (Emilie Bertocchi, Christophe Petit, Gaetan Poulain, Valérie Gouy, Alban Rasse, Olivier Dutartre…) et les doctorants du laboratoire SET (Zakaria Hammoudan, Imad Matraji) pour Les conseils qu'ils m'ont apportés durant ces trois années, et les moments inoubliables qui ont marqué nos merveilleuses trois ans passées ensemble.

Enfin, je tiens à dédier cette étape modeste à mes chers parents et à Mouaz Al sayed ahmad qui ont toujours été à mes côtés et m'ont donné de l'énergie et leur soutien tout au long de cette thèse.

A ma Famille

Sommaire

Sommaire des Figures

Liste des tableaux

CHAPITRE 1

INTRODUCTION

Cette introduction a pour objectif de présenter la problématique de cette thèse, le contexte scientifique dans lequel nos travaux ont été menés et la synthèse de notre contribution.

Table de matières

1.1.Contexte général

Le contexte économique actuel impose aux entreprises, en plus du renouvellement continu de leur gamme de produits, (i) de produire des produits de qualité[1], (ii) de baisser les coûts et (iii) de réduire les délais. Le triplet Qualité-Coût-Délai (QCD) est donc devenu la clé du succès. Un des moyens d'optimiser le triplet QCD est d'introduire au sein de l'entreprise une approche de gestion des connaissances. En effet : la Gestion des Connaissances (GC) permet d'améliorer l'ingénierie collaborative innovante par le partage de ressources et la réduction de la part des processus routiniers au bénéfice de la part des processus innovants. Ceci permet à l'entreprise d'avoir une évolution constante de la qualité de ces produits en dégageant du temps sur la part d'ingénierie routinière et en favorisant l'ingénierie innovante.

Par conséquent, un des défis actuel pour les chercheurs et les entreprises, dans une économie de plus en plus mondialisée, est de concevoir un Système de Gestion des Connaissances (SGC). Ce type de système supporte l'approche de GC au sein de l'entreprise. Comme le disent (Johannessen et al., 2001) « la connaissance est devenu la ressource stratégique la plus importante pour l'entreprise ». Les SGC doivent permettre la capitalisation, le partage et la réutilisation des connaissances afin d'aider les acteurs métiers dans le processus de développement de ses produits. Ces objectifs sont éminemment complexes. Toutefois, un système informatique peut contribuer, surtout s'il est doté « d'intelligence », à la satisfaction de ces objectifs.

Pour cela, les entreprises aujourd'hui déploient des approches pour la valorisation de leur capital intellectuel et le développement de leur savoir-faire afin de viser le succès à long terme grâce à la satisfaction des clients ce qui les rend compétitives. Mais gérer son capital savoir ne consiste pas seulement à diffuser les informations par la mise en place de nouvelles technologies informatiques. Selon (Ermine, 2010) « C'est un programme à long terme qui part d'une volonté stratégique, qui passe par une bonne analyse de la nature même du savoir et du savoir-faire de l'entreprise, et qui aboutit à la mise en place d'outils variés et adaptés ».

Pour notre travail, nous avons fait les hypothèses suivantes : nous travaillons dans un environnement industriel où, lors des activités d'ingénierie, les acteurs métier utilisent des

[1] La **Qualité** est défini par la norme ISO 8402 comme un: ensemble de caractéristiques d'une entité qui lui confèrent l'aptitude à satisfaire des besoins exprimés et implicites.

connaissances et savoirs faires propres. Ces acteurs métiers travaillent en collaboration, au cours des projets, pour concevoir, développer et industrialiser les produits. Pour ce faire, ils utilisent des outils métiers. Parmi ces outils métiers on peut citer : des outils logiciels (outils de calcul (Hanselman and Littlefield, 2004), des outils de CAO (Lalit Narayan et al., 2008), outils de gestion de production (Umble et al., 2003)(Pillet et al., 2011), Product Lifecycle Management PLM (Stark, 2011)(Saaksvuori and Immonen, 2008), Product Data Management PDM (Philpotts, 1996) (Tony Liu and William Xu, 2001), etc.). Dans ce contexte, les connaissances possèdent deux caractéristiques importantes : elles sont hétérogènes, car elles proviennent de sources différentes et sont liées à des métiers différents. Elles sont également distribuées à travers le réseau de l'entreprise étendue. En effet, chaque acteur métier utilise ses propres outils d'information sur son poste de travail connecté à l'ensemble du réseau d'entreprise. Ces raisons font qu'il est très difficile de chercher les connaissances manuellement.

La nature hétérogène et distribuée des connaissances pose par ailleurs les problèmes de l'interprétation et de la manipulation.

1.2. Objectifs de nos travaux

Cette thèse se situe donc dans le cadre général de la gestion des connaissances hétérogènes et distribuées dans une entreprise étendue assisté par un SGC.

Le développement de notre SGC s'inscrit dans le cadre du projet ADN (Alliance de Données Numériques) labélisé pôles de compétitivité Systém@tic-Paris et Véhicule du Futur-Alsace/Franche Comté. Les partenaires industriels du projet sont PSA, Faurecia, EADS et DPS (Digital Product Simulation). Ces entreprises ont des domaines métier différents et souhaitent appliquer la même méthodologie pour la gestion des connaissances multi-sources.

Dans ce qui suit, nous esquissons en quelques mots, les enjeux majeurs de notre travail :

Proposer une approche pour la gestion des connaissances hétérogènes et distribuées lors du processus de conception des produits mécaniques.

La nature hétérogène et distribuée des connaissances, hypothèses de ce travail, pose plusieurs sous-problèmes relatifs à leur gestion.

Le premier sous-problème, réside dans la conception d'un SGC capable de s'adapter à des environnements dynamiques et de gérer les différentes sources de connaissances distribuées au sein d'une entreprise étendue. En effet, à cause de la distribution géographique inhérente à l'entreprise étendue, les acteurs métiers ne partagent pas les mêmes environnements de travail, qu'ils soient matériels (bâtiments, sites, bureaux, …) ou logiciels (environnements de travails, ERP, PLM, …). Un SGC, s'il veut s'adapter à cet existant doit être distribué par nature, apte à gérer l'absence de communications directes entre acteurs métiers et autonome pour gérer les connaissances « au fil de l'eau » afin de rendre transparente cette gestion.

Le deuxième sous-problème, pour un SGC est d'identifier les connaissances à capitaliser et réutiliser. Cela revient à définir, tâche généralement réservée à des experts du domaine, ce qui est considéré comme connaissance. Ce sous-problème est crucial car il permet l'émergence du corpus à gérer.

Le troisième sous-problème, consiste pour le SGC et sur la base d'une définition des connaissances d'extraire les données pertinentes et de les annoter afin de les stocker pour les capitaliser. Pour cela, il faut définir un mécanisme qui va permettre de chercher les données au travers de l'entreprise étendue et de traiter ces données afin de les annoter de façon à produire des connaissances pour les acteurs métiers.

Enfin, le quatrième et dernier sous-problème consiste à fournir des moyens aux utilisateurs, acteurs métiers ou experts, pour évaluer la pertinence et la qualité des connaissances capitalisées.

Les travaux décrits dans cette thèse se placent dans la continuité des travaux issus de (Monticolo, 2008) et (Ben Miled, 2011) qui ont défini respectivement, le cadre d'une approche à base de Systèmes Multi-Agents (SMA) pour la gestion des connaissances, pour le premier, et une approche de réutilisation des connaissances pour le second.

Cette thèse tente d'unifier ces travaux et de les étendre dans le cadre de l'entreprise étendue. Le cœur des travaux présentés ci-après repose sur un SMA qui permet d'une part de prendre en compte la nature distribuée et hétérogène de l'entreprise étendue et d'autre part de supporter une gestion « au fil de l'eau » des connaissances.

1.2.1. SMA pour la réutilisation des connaissances

Notre recherche se situe dans la gestion des connaissances hétérogènes et distribuées au sein des entreprises étendues ce qui implique un système très complexe à gérer. Cette

complexité nous a amené à utiliser le paradigme agent qui a montré sa capacité à gérer ce type de problèmes grâce à ses caractéristiques telles que l'interaction durant la résolution de problèmes, la conception de systèmes nécessitant la distribution, l'adaptation aux changements de l'environnement, l'autonomie des entités, l'élaboration, la simulation et/ou la compréhension de systèmes coopératifs ou compétitifs, distribuées, et ouverts pouvant intégrer à la fois des agents humains et/ou artificiels.

Ce système contient un ensemble d'agents qui coopèrent, négocient, coexistent et interagissent afin de résoudre un problème. C'est un réseau faiblement couplé d'agents qui interagissent pour résoudre des problèmes qui sont au-delà des capacités individuelles ou des connaissances de chaque agent (Durfee et al., 1989).

Les agents sont utilisés de plus en plus fréquemment dans le domaine de la GC grâce aux capacités et services exhibés tels que :

- Rechercher, acquérir, analyser, intégrer et archiver les informations provenant des différentes sources hétérogènes.
- Alerter les utilisateurs de nouvelles informations qui peuvent être intéressantes.
- Négocier, acheter et recevoir des informations.
- S'adapter à l'évolution et au changement de l'environnement.

Dans le but de résoudre ce problème de GC nous avons proposé un modèle agent pour la GC hétérogènes et distribuées lors des projets de conception. Ces agents ont pour objectifs d'identifier, capitaliser, extraire, annoter, partager et réutiliser les connaissances, et d'assister les experts métier à créer leurs propres ontologies décrivant le processus de conception d'un produit.

Afin d'évaluer le système et analyser les résultats, nous l'avons expérimenté avec plusieurs experts métiers travaillant dans les entreprises partenaire de notre projet.

1.2.2. Système d'aide à la création d'ontologie pour la définition des connaissances

Nous avons indiqué dans le contexte le problème de gestion des connaissances que les entreprises affrontent aujourd'hui lors du processus de conception d'un produit mécanique. La conception d'un produit nécessite la collaboration entre plusieurs acteurs métiers de

disciplines différentes. Chacun de ces acteurs utilise ces propres expertises et connaissances pour définir le processus de conception de ce produit.

Dans ce but les ontologies constituent un mode de représentation logique des connaissances qui permet de décrire de manière formelle et structurée la connaissance d'un domaine. Les ontologies son très utilisées en gestion des connaissances. En particulier, OWL (Ontology Web Language) (McGuinness and Van Harmelen, 2004) est une des normes recommandé par le W3C pour le Web sémantique. Ses utilisations principales, dans ce cadre, sont de spécifier la norme des vocabulaires conceptuels dans le but de fournir des services pour répondre aux requêtes, de publier des bases de connaissances réutilisables, d'offrir des services pour faciliter l'interopérabilité entre plusieurs systèmes hétérogènes et bases de données, et de raisonner en déduisant de nouveaux faits à partir de ceux qui existent déjà.

Nous proposons dans cette thèse une approche qui permet aux experts métiers de créer leurs propres ontologies de domaine. Chaque domaine, dans ce cadre, se restreint à la description d'un processus métier de conception d'un produit. Chaque ontologie de domaine s'apparente à une taxonomie hiérarchique des concepts, associés à des attributs et reliés entre eux par des relations.

L'approche proposée dans ce cadre est de guider l'expert métier lors de la création d'ontologie de domaine. En effet, la création d'ontologie est une tâche difficile. De plus, un expert métier n'est pas forcément compétent en matière d'ontologies.

1.2.3. Mécanisme d'extraction des connaissances

Les connaissances à extraire se trouvent dans des applications métiers différentes d'où la notion des connaissances hétérogènes et distribuées. Ces applications métiers utilisent des formes de stockage de données différentes : bases de données SQL, fichiers Excel, fichiers XML, fichiers texte...

Il faut trouver un pont pour lier l'ontologie de domaine à ces sources de données. Pour cette raison, nous nous sommes basés sur l'Ingénierie Dirigée par les Modèles (Muller, 2005). Cette technique permet la représentation simplifiée d'un aspect de la réalité pour un objectif donné en passant à un niveau d'abstraction plus élevé : le modèle. Cette technique permet également par le biais de langages de transformation d'exprimer des règles de traduction d'un modèle vers un autre modèle en considérant les concepts propres à chaque modèle définis dans un modèle d'abstraction plus importante : le méta-modèle.

Donc si on nomme notre ontologie de domaine modèle « a » et la source de données modèle « b », il est indispensable de passer au niveau d'abstraction ou méta modèle de ces deux modèles pour les abstraire et pour définir des règles de traduction. Ces règles nous servent à passer du modèle a au modèle b. Cette transformation est exprimée dans la figure ci-dessous (Figure 1).

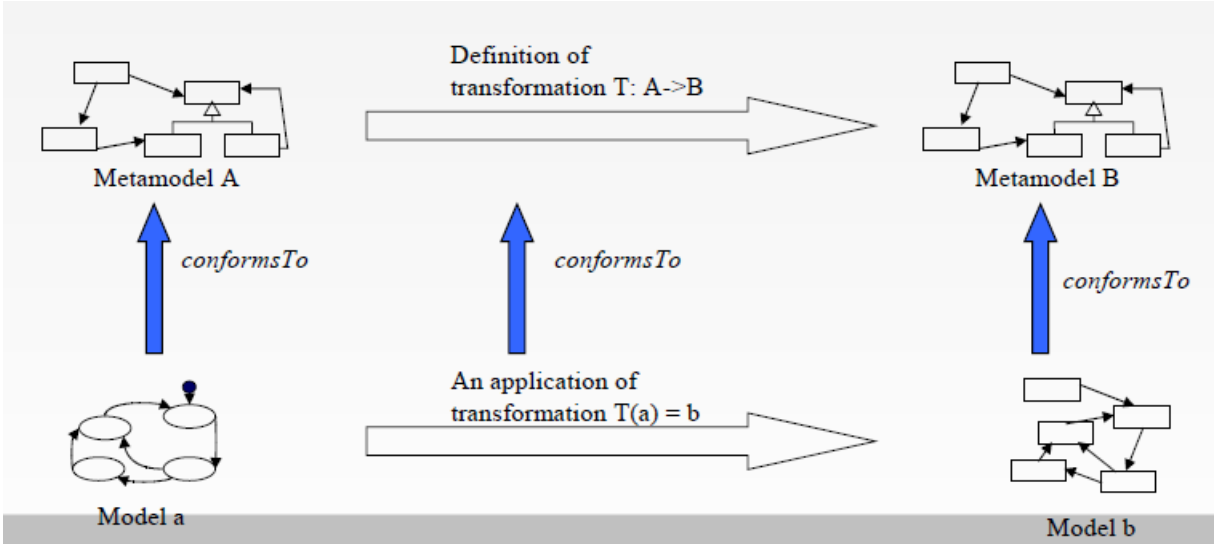


Figure 1: Transformations des modèles

Dans le cadre de notre travail, nous transformons les ontologies en des requêtes et les exécuter sur les applications métiers afin d'extraire les données. Mais ces données extraites n'ont aucune signification toute seules. Par exemple : 20 est une donnée mais nous ne pouvons pas savoir si 20 signifie la température ou l'âge d'une fille ou bien la longueur d'une table, etc…

Pour cela il est judicieux de mettre cette donnée dans un contexte. Cette contextualisation des données s'appelle annotation. Lors de l'annotation d'une donnée ce dernier devient une information. Pour annoter nos données nous avons utilisés l'ontologie « OntoDesign » (Monticolo et al., 2007a) définie pour ce but.

1.2.4. Partage et évaluation des connaissances

Ces informations obtenus par l'annotation seront stocké sous un format structuré dans une mémoire organisationnelle afin qu'elles soient ensuite exploitées pour la recherche sémantique. Nous avons opté pour un wiki sémantique pour stocker et manipuler les connaissances. Ce dernier permet aux acteurs de partager les projets existants.

Dans notre cas industriel nous avons choisi le wiki pour partager les connaissances des experts métiers parce qu'il est simple à utiliser et permet en même temps un accès sécurisé aux connaissances en fonction de leurs rôles dans l'entreprise. Ce wiki va permettre aux acteurs métiers d'accéder aux connaissances des projets pour les réutiliser et aux experts métiers de modifier, d'évaluer et de faire évoluer les connaissances. L'évaluation se fait en associant une note aux connaissances. La note globale de chaque connaissance, évaluée par plusieurs experts, permet d'avoir son indice de maturité. Plus l'indice est élevé plus la connaissance est considérée comme pertinente.

1.3.Plan de la thèse

Après ce bref exposé de nos propositions, cette section présente le plan de cette thèse. La première partie de cette thèse est un tour d'horizon des domaines abordés lors de nos travaux de recherche. Cette partie est composée d'un seul chapitre avec deux grandes sections. Ces derniers constituent un état de l'art où sont analysés les travaux existants sur la gestion des connaissances hétérogènes et distribuées, et les systèmes multi-agents pour la gestion des connaissances.

- Chapitre 2.
  - Section 1 : la gestion des connaissances hétérogènes et distribuées. Nous commençons dans ce chapitre à présenter les termes données, information et connaissances pour mettre en claire les définitions que nous avons adopté pour le long de notre travail. Nous présentons ensuite le domaine de la gestion des connaissances avec ses défis et son importance dans les entreprises aujourd'hui. Nous poursuivons ensuite par la présentation des cycles de vie pour la gestion des connaissances. Nous terminons le chapitre par la présentation des informations multi-sources et les travaux existants dans ce domaine.

  - Section 2 : Les systèmes multi-agents pour la gestion des connaissances. Nous décrivons le paradigme agent suivi de la présentation d'intérêt des SMA dans la GC. Nous présentons par la suite quelques approches agents pour la gestion des connaissances et nous terminons par une synthèse sur ces approches.

La deuxième partie de la thèse décrit le travail de modélisation d'un cycle de gestion des connaissances. Nous avons proposé dans le premier chapitre une méthodologie pour la GC hétérogènes et distribuées composé de quatre étapes. Le deuxième chapitre représente la première étape qui est l'identification et capitalisation des connaissances utilisés lors d'un projet de conception d'un produit en créant des ontologies par les experts métiers spécifiant un vocabulaire et une sémantique représentant les connaissances du domaine. Ensuite notre troisième chapitre présente l'exploitation et l'extraction des connaissances de plusieurs applications métiers. Dans ce chapitre on présente également un système d'annotation des données et de stockage dans une mémoire organisationnelle. Nous terminons par un chapitre qui est dédié à la réutilisation et la diffusion des connaissances.

- Chapitre 3 – Perspective Globale. Ce chapitre présente notre méthodologie globale pour la gestion des connaissances hétérogènes et distribuées en se basant sur un cycle de vie de quatre étapes. Nous présentons ainsi les organisations agents pour chacune de ces étapes en utilisant une méthodologie d'analyse spécifique aux SMA.

- Chapitre 4 – La définition des connaissances et la création d'ontologies. Ce chapitre présente une méthodologie pour l'identification des connaissances en créant des ontologies. Nous présentons notre système d'aide à la création d'ontologies et la méthode pour gérer, notamment, les synonymes dans l'ontologie.

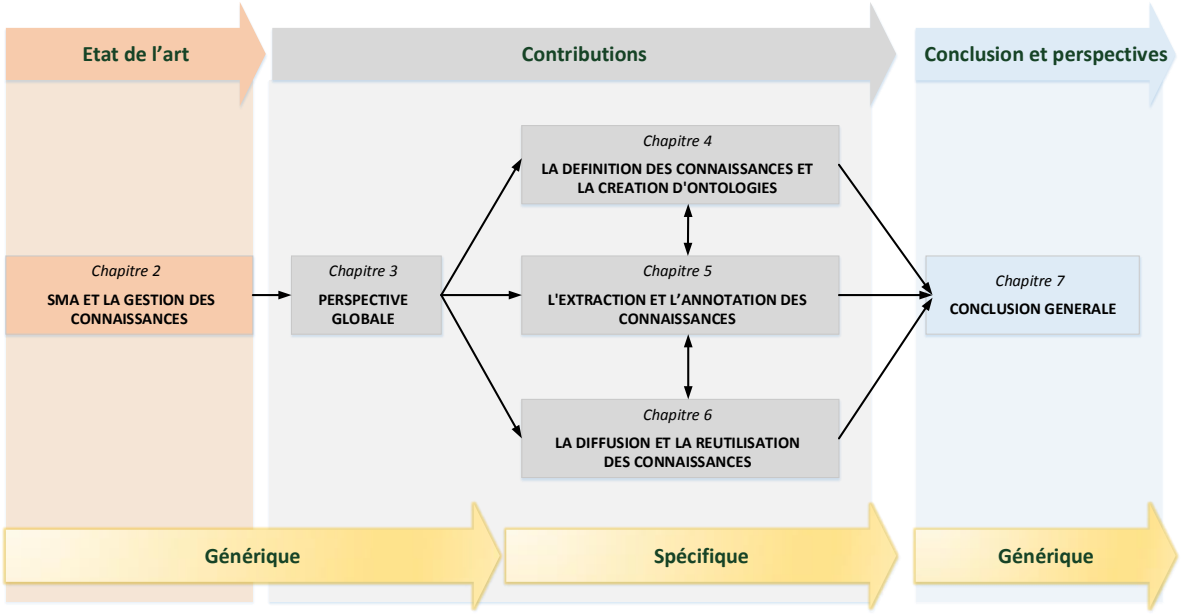
- Chapitre 5 – L'extraction et l'annotation des connaissances. Nous présentons dans ce chapitre notre méthodologie pour la transformation de l'ontologie en une requête exécutable sur une application métier afin d'extraire des données. Nous présentons aussi le système d'annotation des données extraites qui permet de les transformer en informations et enfin de les stocker dans une mémoire organisationnelle.

- Chapitre 6 – La diffusion et la réutilisation des connaissances. Nous décrivons dans ce chapitre le système de partage et de diffusion des connaissances par un wiki sémantique. Ce wiki permet aux utilisateurs, selon leurs rôles, de valider,

faire évoluer, modifier ou supprimer les connaissances, s'ils sont experts, de consulter les projets existants et réutiliser les connaissances, s'ils sont des acteurs.

La troisième partie présente l'apport de la thèse avec les perspectives.

- Chapitre 8 – Conclusion et perspectives. Ce chapitre rappelle notre contribution pour répondre au problème de gestion des connaissances dans les projets de conception mécanique. Il soulève une discussion sur les perspectives de recherches à mener en prolongement de cette thèse.



Le schéma ci-dessus présente le plan de lecture de la thèse du point de vue état de l'art, et méthodologies proposés. Cette thèse expose des informations récentes dans les domaines de la gestion des connaissances et des systèmes multi-agents en proposant un outil de création, d'extraction, d'annotation et de validation des connaissances. Nous espérons que les lecteurs trouveront les informations nécessaires pour qu'ils puissent créer et partager de nouvelles connaissances.

Première Partie

ETAT DE L'ART

CHAPITRE 2

SMA ET GESTION DES CONNAISSANCES

Ce chapitre présente le contexte des domaines de recherche dans lequel se situe ce travail : celui de la gestion des connaissances hétérogènes et distribuées, et des systèmes multi-agents. En premier lieu nous définissons le domaine de la gestion des connaissances, ses caractéristiques et son importance pour les entreprises. L'intérêt est par la suite porté sur les systèmes multi-agents, leurs avantages, ainsi qu'une revue de quelques approches agents dédiées à la gestion des connaissances.

Table de matières

2.1. Introduction

La « richesse intellectuelle» d'une entreprise est en lien avec les connaissances qu'elle possède. Selon (Grundstein, 2004), ces connaissances sont constituées d'éléments tangibles (les bases de données, les procédures, les plans, les modèles, les algorithmes, les documents d'analyse et de synthèse) et d'éléments immatériels, les habiletés, les tours de mains, les « secrets de métiers », les « routines » - logiques d'action individuelles et collectives non écrites (Nelson and Winter, 1990) -, les connaissances de l'historique et les contextes décisionnels.

Toutefois, les connaissances collectives d'une entreprise, qui constituent une de ses ressources essentielles, sont le plus souvent transmises oralement et de manière implicite. Ceci signifie que les connaissances de l'entreprise restent fortement dépendantes des connaissances des personnes et de leur présence dans l'entreprise. Ces connaissances individuelles seront par conséquent difficiles à repérer, exploiter et valoriser par d'autres collaborateurs. Cette problématique pousse les entreprises à chercher des solutions pour capitaliser ces savoirs et savoirs faires. Une des solutions consiste à concevoir un système informatique de gestion des connaissances. Ce système permet à l'entreprise de conserver ses connaissances, les mettre à jour, les évaluer, les partager et de les réutiliser dans des futurs projets. La capitalisation évite de « perdre » des connaissances lors du départ d'experts ou de spécialistes (démission, départ en retraite,…) ou même « d'oublier » des connaissances acquises lors d'anciens projets.

Nous présentons dans ce chapitre le domaine de la gestion des connaissances hétérogènes et distribuées. Puis nous présentons la technologie des agents et quelques approches existantes pour gérer ces connaissances.

2.2.La gestion des connaissances hétérogènes et distribuées

2.2.1. Gestion des connaissances

2.2.1.1. Données, informations et connaissances

Lorsque l'on entre dans le domaine de la gestion des connaissances, les termes « données », « information » et « connaissance » sont fréquemment employés. C'est pour cette raison qu'il est très important de donner une définition pour chacun de ces termes.

Malgré la grande fréquence d'utilisation de ces termes dans le domaine de la gestion de connaissance, il est difficile de trouver une définition faisant consensus.

Les définitions données par Gandon (Gandon, 2002) paraissent les plus proches de notre travail. Gandon a défini la connaissance comme une compréhension d'un sujet obtenue par une expérience ou une étude, l'information comme une connaissance en transit (c.à.d. elle est déjà contextualisé mais pas encore validée dans une expérience), et les données comme l'élément de base de l'information. Par exemple, « 529 » est une donnée mais nous ne savons pas à quoi elle correspondait. Si on met cette donnée dans un contexte comme « la longueur du tube de selle du vélo speedmax AL 8.0 M est de 529 » elle devient une information. Quand un humain valide cette information et précise qu'elle est toujours en actualité, cette information devient une connaissance.

En résumé :

- une donnée est un élément brut livré en dehors de tout contexte
- une information est une donnée contextualisée
- une connaissance est l'interprétation d'une information par un humain

Nous allons par la suite définir le terme « gestion des connaissances » et expliquer son importance pour les entreprises et les défis qu'il affronte dans nos jours.

2.2.1.2. La gestion des connaissances

Comme expliqué dans l'introduction, les entreprises aujourd'hui sont condamnées à développer leur savoir-faire en enrichissant leur capital intellectuel par la création sans cesse des connaissances. Ceci permet d'assurer la survie de l'entreprise.

Le défi aujourd'hui pour les chercheurs et les entreprises dans une économie de plus en plus mondialisée est de développer un système qui permet la capitalisation et la réutilisation de ces connaissances.

Il existe des nombreuses définitions de la gestion des connaissances, parmi eux celle de Nakra (Nakra, 2000), Allee (Allee, 1997), Davenport (Davenport et al., 1998), Alavi et Leidner (Alavi and Leidner, 2001), Jennex (Jennex, 2007) et Malhotra et Galletta (Malhotra and Galletta, 2003) qui ont expliqué que « La gestion des connaissances favorise une approche intégrée qui permet d'identifier, d'intégrer, d'acquérir, de partager et d'évaluer l'ensemble des connaissances d'une entreprise. Ces connaissances peuvent se trouver dans des bases de données, des documents divers, des lignes de conduite et des procédés, ainsi que dans le savoir-faire et l'expérience tacite de chaque membre de l'entreprise».

Notons que la GC est considéré comme une gestion réussie si on peut réutiliser les connaissances afin d'améliorer l'efficacité organisationnelle en fournissant les connaissances appropriées à ceux qui en ont besoin quand elle est nécessaire (Jennex et al., 2007).

Les avantages de la GC sont énumérés par Dalkir en 2005 (Dalkir, 2005) qui a présenté les avantages de la gestion des connaissances pour l'individu, la communauté de pratique et l'organisation comme présenté ci-dessous.

- **Pour l'individu**

  - Aide les gens à faire leur travail et à économiser du temps par de meilleures prises de décisions et résolutions de problèmes ;
  - Crée un sentiment de liens communautaires au sein de l'organisation à mesure que les travailleurs se sentent valorisés pour leur apport ;
  - Augmente le niveau de satisfaction des employés ;
  - Aide les gens à se tenir au courant ;
  - Procure des mises au défi et des occasions d'apporter son aide.

➢ **Pour la communauté**

- Développe des compétences professionnelles ;
- Favorise le mentorat entre pairs ;
- Contribue à un réseautage et une collaboration plus efficaces ;
- Crée un code d'éthique professionnelle que les membres peuvent suivre ;
- Crée un langage commun.

➢ **Pour l'organisation**

- Aide à orienter la stratégie ;
- Règle les problèmes rapidement ;
- Diffuse les pratiques exemplaires ;
- Améliore les connaissances qui font partie des produits et services ;
- Sème en croisé des idées et augmente les occasions d'innover ;
- Permet à l'organisation de mieux devancer la concurrence ;
- Réduit la redondance ;
- Réduit les coûts de recherche et de développement ;
- Améliore les procédés internes de travail ;
- Réduit le nombre d'erreurs ;
- Accroît la diversité des points de vue dans les décisions d'affaires en faisant intervenir les travailleurs ;
- Bâtit une mémoire organisationnelle en conservant le capital intellectuel.

Nous avons vu l'importance de la GC pour l'individu, pour la communauté et pour l'organisation. Pour réaliser un système de gestion des connaissances, il faut prendre en compte le cycle de vie des connaissances. Nous présentons ces cycles de vie dans la section suivante.

2.2.1.3. Le cycle de vie de la connaissance

Alavi and Leidner (Alavi and Leidner, 2001) expliquent que le système de gestion des connaissances (SGC) est un système basé sur les technologies de l'information développé pour soutenir et améliorer les processus organisationnels de la création de connaissances, le

stockage / récupération, le transfert et l'application. Maier (Maier, 2002) élargit le concept de la technologie de l'information pour le SGC en l'appelant système de technologies de l'information et de la communication qui prend en charge les fonctions de création de connaissances, la construction, l'identification, la capture, l'acquisition, la sélection, l'évaluation, l'organisation, la liaison, la structuration, la formalisation, la visualisation, la distribution, la conservation, l'entretien, l'amélioration, l'évolution, l'accès, la recherche et l'application. SGC utilise une variété de technologies conçues pour améliorer le stockage des connaissances, la communication et le transfert des connaissances.

Grundstein a également proposé un cycle de vie de la gestion des connaissances, où, selon lui, "dans toute opération de capitalisation des connaissances, il est important d'identifier les connaissances stratégiques à capitaliser". Son cycle est divisé en quatre volets qui sont: l'identification, la formalisation, mettre en valeur et la mise à jour (Grundstein and Barthès, 1996).

Il existe des nombreuses autres propositions de cycles de vie pour la gestion des connaissances comme par exemple Abecker (Abecker et al., 1998) qui a présenté un cycle en six étapes ; identification, acquisition, développement, diffusion, utilisation, et conservation, et puis Jaspers (Jaspers et al., 1999) qui expose sept étapes : identifier les connaissances explicites, capturer les connaissances tacites et les rendre explicites, les organiser, les maintenir, les diffuser, permettre leur recherche et les réutiliser. D'autre part, Pomian (Pomian, 1996) propose un cycle plus simple en trois étapes : identifier, collecter et réutiliser.

En se basant sur ces cycles de vie, nous avons créé notre propre cycle de vie pour la réalisation de notre système de gestion de connaissances multi-sources. Nous présenterons ce cycle de vie dans le chapitre 3.

2.2.2. La gestion des connaissances multi-sources

Nous travaillons dans un environnement industriel où le développement et la production d'un nouveau produit nécessite la collaboration entre plusieurs entreprises partenaires ce qu'on appelle une entreprise étendue (Tran, 2012).

Par conséquent, l'intérêt industriel des méthodologies et des outils permettant la capitalisation et la gestion des connaissances distribuées et hétérogènes se sont renforcées (Xu

et al., 2011)(Lin et al., 2011), en particulier dans le développement de produits complexes au sein de l'entreprise étendue (Tiedeken et al., 2013). Ces projets complexes impliquent une équipe multidisciplinaire (mécaniciens, automaticiens, concepteurs, ingénieurs et techniciens des méthodes, etc) où les acteurs travaillent en collaboration pour concevoir, développer et industrialiser un produit. Pour ce faire, ils utilisent des outils logiciels spécifiques (outil de calcul, outil de CAO, outil de gestion de production, etc.). Chacun de ces outils génère un ensemble de données. Ces données sont hétérogènes, car elles proviennent de sources différentes. Elles sont également distribuées à travers le réseau de l'entreprise puisque chaque acteur métier utilise ses propres logiciels sur son poste de travail connecté à l'ensemble du réseau d'entreprise. Ceci rend très difficile la recherche manuelle des connaissances (Monticolo et al., 2007b).

La gestion de l'information multi-sources est l'objectif de nombreuses recherches dans tous les domaines. Nous présentons dans cette section quatre méthodes utilisés par les chercheurs pour gérer ce problème des connaissances hétérogènes et distribuées. Ces méthodes sont : l'utilisation d'une ontologie, un modèle d'échange d'objets, une base de données commune, et l'accès direct des bases de connaissances à l'aide d'un langage de requête spécifique.

La première méthode a été utilisée dans des nombreux travaux. Stevens (Stevens et al., 2000) a proposé le système TAMBIS pour donner une solution intégrée au problème de l'hétérogénéité des bases de données biologiques et des outils d'analyse, en se basant sur une base de connaissance biologique commune (une ontologie). De même, Xiuqin Qiu (Qiu and Yue, 2010) a proposé un système basé sur une ontologie pour gérer les informations distribuées et hétérogènes, mais cette fois dans le domaine agricole, afin d'obtenir d'une manière efficace des informations utiles. Pirro (Pirró et al., 2010) a utilisé aussi les ontologies dans le système qu'il a proposé DOKMS afin de créer, organiser, rechercher, diffuser et réutiliser les connaissances multi-sources dans une communauté. Ceci permet d'améliorer la collaboration au sein de la communauté. DOKMS s'appuie également sur un mécanisme de type P2P pour gérer le problème des connaissances distribuées.

Les ontologies ont été proposé également par Nageba (Nageba et al., 2013) comme une base pour un système de gestion des connaissances hétérogènes et distribuées dans le domaine de la santé. Ce système, nommé ONtology Oriented Framework for Pervasive Applications and Services (ONOF-PAS), permet de traiter les informations acquises sur les utilisateurs, les tâches qu'ils accomplissent, la disponibilité et les capacités des ressources nécessaires, et les

organisations qui possèdent ou gèrent ces ressources. Alors le système ONOF-PAS propose, en utilisant les ontologies, les connaissances nécessaires à chaque utilisateur en fonction de sa situation médicale et de ses besoins.

La deuxième méthode a été utilisé par Chawathe (Chawathe et al., 1994). Chawathe a présenté dans le projet TSIMMIS cette nouvelle technique pour gérer les informations multi sources. Pour ce faire, Chawathe a adopté un modèle auto-descriptif d'objets appelé modèle d'échange d'objets où tous les objets et sous-objets ont des étiquettes pour décrire leur signification. En outre, Chawathe a développé un langage de requête pour interroger les objets et obtenir les informations demandées.

Agarwal (Agarwal et al., 2012) a choisi de gérer les informations de la station nucléaire Savannah River, provenant de différentes bases de données et distribuées sur différents ordinateurs, en utilisant la troisième méthode. Il a proposé un système de gestion des connaissances (ODMS) basé sur une conception d'une nouvelle base de données. Cette approche permet de stocker les données de toutes les bases avec un format unique, et permet également la génération automatique de vues qui présentent le même genre d'interfaces pour tous les types de données pertinentes.

Stegmaier (Stegmaier et al., 2013) a utilisé la quatrième méthode pour gérer le même problème de GC. Pour ce faire, Stegmaier a présenté dans son article une architecture, nommé AIR (Architecture for Interoperable Retrieval), qui offre des possibilités réelles de récupération des données stockées dans des sources différentes de multimédia. Pour assurer l'interopérabilité, l'architecture AIR utilise un langage de requête multimédia, le MPEG Query Format (MPQF), et les règles de transformation de JPSearch. Quand le système reçoit une requête du client, AIR évalue et optimise la requête et puis envoie chaque partie de la requête analysé à la base de données respective pour l'avoir exécuté. L'agrégation des résultats de toutes les parties est présentée au client après une élimination des doublons.

Les systèmes de gestion des connaissances hétérogènes et distribuées sont assez complexes à gérer, et requièrent une intervention importante et constante des utilisateurs. Pour pallier à ces inconvénients, une des hypothèses de notre travail est d'utiliser le paradigme des systèmes multi-agents que nous présentons dans la section suivante. En effet, le paradigme agents a prouvé son utilité pour gérer ce type de complexité.

2.3.Les SMA et la gestion des connaissances

2.3.1. Les systèmes multi agents

Les systèmes multi-agents font partis de l'intelligence distribuée (IAD). L'idée de base est de distribuer l'intelligence sur plusieurs agents où chaque agent est associé à un sous problème ou sous-objectif. De la coordination des activités de ces agents émerge le but à atteindre ou objectif initial (résoudre le problème). Ceci a conduit à passer de descriptions de comportements individuels à des comportements collectifs à travers la coopération de plusieurs agents qui sont capables d'associer leurs efforts pour accroître leur intelligence collective. L'IAD s'intéresse donc aux comportements intelligents, résultant de l'activité coopérative de plusieurs entités (agents) (Labidi and Lejouad, 1993). Le concept de coopération est indispensable pour l'IAD tout comme l'interaction entre les agents qui se traduit par la communication et la négociation.

Donc l'IAD introduit le concept du système multi agents qui est un système constitué d'agents, qui interagissent dans et au travers d'un environnement selon certaines relations, et dont les caractéristiques sont : La coopération, la coordination et la communication.

Ferber a donné plus de détails sur la composition du SMA en précisant qu'un système multi-agents doit être composé des éléments suivants (Ferber, 1995):

- Un environnement **E**, c'est-à-dire un espace disposant généralement d'une métrique.
- Un ensemble d'objets **O**. Ces objets sont situés, c'est-à-dire que, pour tout objet, il est possible, à un moment donné, d'associer une position dans E. Ces objets sont passifs, c'est-à-dire qu'ils peuvent être perçus, créés, détruits et modifiés par les agents.
- Un ensemble **A** d'agents, qui sont des objets particuliers (A est contenu dans O), lesquels représentent les entités actives du système.
- Un ensemble **R** de relations qui unissent des objets (et donc des agents) entre eux.
- Un ensemble d'opérateurs **Op**. permettant aux agents de A de percevoir, produire, consommer, transformer et manipuler des objets de O.
- Des opérateurs chargés de représenter l'application de ces opérations et la réaction du monde à cette tentative de modification, que l'on pourrait appeler «les lois de l'univers».

2.3.2. Les agents

Le terme agent est utilisé dans des domaines très différents. On entend parfois des agents de voyage, des agents immobiliers, etc. Aussi bien dans la même communauté un agent peut être un robot, un agent autonome, un agent personnel, etc. Donc il est important de donner les définitions présentes dans le domaine informatique et de préciser lequel on a adopté pour cette thèse.

Selon Ferber un agent est toute entité physique ou virtuelle (Ferber, 1995) qui comprend les caractéristiques suivantes :

- qui est capable d'agir dans un environnement
- qui peut communiquer directement avec d'autres agents
- qui est mue par un ensemble de tendance (sous la forme d'objectifs individuels ou d'une fonction de satisfaction, voire de survie, qu'elle cherche à optimiser)
- qui possède des ressources propres
- qui est capable de percevoir (mais de manière limitée) son environnement
- qui ne dispose que d'une représentation partielle de cet environnement (et éventuellement aucune)
- qui possède des compétences et offre des services
- qui peut éventuellement se « reproduire »
- qui a un comportement qui tend à satisfaire ses objectifs, en tenant compte des ressources et des compétences dont elle dispose, et en fonction de sa perception, de ses représentations et des communications qu'elle reçoit.

Pour que l'agent puisse agir dans son environnement d'une façon autonome, on doit lui fournir alors une base de connaissance qui contient les informations nécessaires pour la réalisation de ses tâches et qui lui permet d'interagir avec les autres agents. On parle de l'autonomie car son comportement dépend au moins partiellement de son expérience (Weiss, 1999).

Il est bien de noter que les agents ne connaissent pas leur environnement global ce qui signifie qu'ils n'ont pas une vision globale. Toutefois, chaque agent possède des compétences

spécifiques et tant que spécialiste dans son domaine et connaît, ou est supposé connaitre, les détails nécessaires pour résoudre les problèmes dans son domaine.

2.3.3. Les approches agents pour la gestion des connaissances

Un système de gestion de connaissance (SGC) est utilisé dans une organisation pour gérer les connaissances explicites (Dieng-Kuntz and Matta, 2002) ou tacites. Ceci signifie que les connaissances sont clairement identifiées dans un document écrit (par exemple, la longueur du tube de selle = 85cm) ou à partir d'un logiciel (outil de CAO, calcul, etc.).

Il y a beaucoup d'efforts dans le développement de SGC comme Abar (Abar et al., 2004) qui présentent la prochaine génération des architectures de SGC et Zhang Xin (Zhang et al., 2006) qui introduit le SGC dans le commerce électronique. Parmi les technologies candidates pour implémenter un SGC les agents et les ontologies sont particulièrement bien adaptées (Fu and Xin, 2008).

En effet, les systèmes de gestion de connaissances sont des systèmes complexes surtout lorsque les connaissances sont distribuées et hétérogènes. D'où la nécessité d'utiliser un système multi-agents pour gérer un SGC. En effet, le SMA offre plusieurs avantages tels que l'interopérabilité avec d'autres systèmes existants, et l'hétérogénéité. Un système multi-agents peut être vu comme une équipe, composé d'un groupe d'agents éventuellement hétérogènes et autonomes, qui partagent un but commun, et travaillent ensembles pour y parvenir.

Le développement d'un SGC basé sur le paradigme agent a été le sujet de recherche de plusieurs travaux comme (Vizcaino et al., 2007) qui a recommandé un système multi-agents aux ingénieurs informatiques pour supporter les différentes étapes de cycle de vie d'un SGC. (Wang and Xu, 2010) a proposé un SGC basé sur les SMA pour aider une organisation dans la résolution de problèmes et dans les activités de prise de décisions. Pour gérer et partager les connaissances de plusieurs applications (de fabrication, bancaire, de médecine, de divertissement..), Sajja (Sajja, 2008) a suivi une modélisation topologique fixe tel qu'un système de gestion des bases de données relationnelles (RDBMS) géré par un SMA. (Zhang et al., 2008) a proposé également un SMA pour gérer les connaissances hétérogènes et distribuées mais dans le cas de réalisation des projets en collaboration via internet. Tandis que (Toledo et al., 2012) a géré ce problème dans le cas de gestion des processus métier en se basant sur la plateforme JaCaMo (Boissier et al., 2013). D'autre part, (Bhat and Wahid, 2012) a proposé un système multi-agents pour l'acquisition, l'évolution, le transfert et la formalisation / stockage des connaissances multi-sources, et en plus un journal d'agents pour enregistrer les activités réalisés par les agents. Ceci permet de savoir quelle connaissance a été

acquise, comment elle a été formalisée, utilisée, partagée, transformée et progressée, afin d'aider les développeurs des SGCs à implémenter ce genre de système.

Alors que de nombreux travaux de recherche se concentrent sur la façon d'élaborer un SGC à l'aide d'agents, Peinl (René Peinl, 2011) présente dans un article une analyse sur l'impact des mesures de gestion des connaissances sur les organisations de l'équipe avec la simulation basée sur les systèmes multi-agents.

En outre, les SMA ont un rôle important dans la prise de décision collective dans la gestion des connaissances comme l'expliquent Brigui-Chtioui (Brigui-Chtioui et al. 2010). Ces auteurs proposent un modèle multi-agents argumentatif à base d'agents médiateurs afin d'automatiser la résolution de conflits entre les décideurs pour identifier les connaissances.

L'approche des SMA pour la gestion des connaissances a également été utilisée dans le domaine industriel, par exemple, pour créer un processus logistique dynamique fondée sur les connaissances du système - RFID (Chow et al., 2006) (Zhang et al., 2010), de même un système de gestion des perturbations et l'atténuation des risques dans les chaînes d'approvisionnement de fabrication (Giannakis and Louis, 2011).

Il existe de nombreuses plateformes qui décrivent la coopération d'agent pour résoudre des problèmes complexes liés à la connaissance comme FRODO (Abecker et al., 2003) , CoMMA (Gandon et al., 2002), Edamok (Bonifacio et al., 2002), KRAFT (Jee and Yang, 2006), MUSETTE (Champin et al., 2003), KATRAS (Monticolo et al., 2009), MAEKMS (Rahman et al., 2012), et les systèmes de (Yang and Chang, 2011) et (FU et al., 2008). Nous présentons par la suite une description de quelques systèmes cités ci-dessus et nous terminons cette section par une synthèse où nous positionnons notre approche de gestion des connaissances.

2.3.3.1. KRAFT (Preece et al., 2000)

La fusion des connaissances à partir de multiples sources distribuées et hétérogènes est l'objectif du projet KRAFT (Knowledge Reuse And Fusion/Transformation). L'architecture KRAFT a été conçu d'une façon à aider les utilisateurs à : (i) trouver des données et des connaissances pertinentes et répond à leurs besoins actuels, (ii) combiner et affiner les données et les connaissances afin de générer les informations dont ils ont besoin, (iii) identifier et exploiter les moteurs de traitement qui sont capable de résoudre leurs problèmes.

En outre, l'architecture KRAFT aidera les fournisseurs et les développeurs de systèmes d'information à (i) mettre leurs ressources à la disposition de la plus nombre d'utilisateurs, (ii)

de s'adapter aux changements produites dans l'information ou le service qu'ils fournissent, tout en minimisant les effets sur les demandes des clients et (iii) garder l'accès des utilisateurs uniquement à leurs ressources.

KRAFT est basé sur une architecture multi-agents. Cette architecture utilise des contraintes comme un format d'échange de connaissances communes, exprimée en termes d'une ontologie commune. Alors pour résoudre le problème des connaissances multi-sources, les connaissances présentes dans les différentes bases locales peuvent être traduites dans le langage de contrainte commune (ontologie), fusionnée avec les connaissances provenant d'autres sources. Lorsque l'utilisateur identifie les connaissances dont il a besoin, KRAFT recherche ces connaissances de la base des connaissances commune et présente à l'utilisateur les connaissances pertinentes qu'il a trouvées afin de les réutiliser. La mise en œuvre de KRAFT est basée sur le standard FIPA avec RDF en tant que langue du contenu. Cette architecture a été réutilisée dans une application de services de réseau, ainsi que dans les systèmes prototypes pour conseiller les étudiants sur les transferts universitaires, et pour conseiller les professionnels de la santé sur les traitements médicamenteux.

2.3.3.2. FRODO (Abecker et al., 2003)

Le projet FRODO ("A Framework for Distributed Organizational Memories") est une plateforme flexible pour la gestion des mémoires organisationnelles distribuées. La mise en œuvre de cette plateforme est basée sur FIPA-compliant agent platform (Bellifemine and Rimassa, 2001).

Une mémoire organisationnelle dans FRODO peut être considérée comme un système de méta-informations avec une intégration étroite dans les processus métiers de l'entreprise. Celui-ci est basé sur des modèles formels appropriés et des ontologies comme base de connaissance pour atteindre une compréhension commune et des capacités de traitement automatique (Abecker et al., 1998).

L'architecture des mémoires organisationnelles dans FRODO comprend autres couches qui sont : i) la couche application, ii) la couche source d'information, iii) et la couche description des connaissances (couche de métadonnées), et iv) la couche d'accès aux connaissances. Les agents dans FRODO résident sur les quatre couches. Un agent FRODO n'est pas seulement décrit par ses connaissances, ses objectifs et ses compétences, mais aussi par ses droits et obligations, formant ainsi des sociétés d'agents.

Grâce à cette architecture d'agents présents dans FRODO, ce système permet de gérer les connaissances hétérogènes et distribuées dans plusieurs mémoires organisationnelles. Pour ce faire, la description des connaissances est maintenue par des ontologies. Ces derniers serviront comme des ponts entre les différents mémoires organisationnels. Le système cherche alors les connaissances, supposées pertinentes pour l'utilisateur, en utilisant les ontologies. Puis il lui propose ces connaissances pour l'assister dans le développement des processus complexes dans le domaine nucléaire,

2.3.3.3. CoMMA (Bergenti et al., 2002)

CoMMA (Corporate Memory Management trough Agent) est un projet européen (Bergenti et al., 2002) qui a comme but le développement d'une application de gestion d'une mémoire d'entreprise. Cette application est basé sur une ontologie formalisée en RDFS (O'CoMMA (Gandon et al., 2001)) afin de faciliter la recherche à l'aide d'un moteur d'inférence CORESE (Corby et al., 2006), et sur la technologie des agents pour gérer la mémoire organisationnelle. CoMMA utilise la plate-forme JADE qui est compatible avec le standard FIPA.

La plateforme CoMMA vise à assister deux scénarios d'application : (i) l'intégration d'un nouveau employé à une organisation et (ii) l'assistance aux activités de veille technologique. Le système ne gère pas directement les documents de la mémoire organisationnelle, mais les annotations des documents référencés leur URI.

CoMMA s'est focalisé sur trois fonctionnalités : (i) améliorer la précision pour rechercher les documents, en utilisant des annotations sémantiques ; (ii) suggérer proactivement des informations en utilisant les organisations et les modèles utilisateurs ; (iii) archiver les nouvelles annotations soumises.

Pour gérer le problème des informations provenant de plusieurs documents, le projet CoMMA a permis de construire une mémoire d'entreprise matérialisée dans une base documentaire annotée par des annotations sémantiques basées sur l'ontologie O'CoMMA. Une société d'agents coopérant et guidés par l'ontologie permettait la recherche d'information dans cette mémoire d'entreprise, l'ajout d'annotations dans la base d'annotations et l'interaction avec les utilisateurs en intégrant CORESE dans un agent « Moteur de recherche».

2.3.3.4. KATRAS (Monticolo et al., 2009)

KATRAS (Knowledge Acquisition Traceability and Reuse by Agents System) est un système de gestion de connaissances qui a pour objectif la capitalisation et la réutilisation semi-automatisés des connaissances tout au long de processus de conception des produits mécaniques. Cette approche est basée sur une mémoire organisationnel, une ontologie et un système multi-agents.

L'ontologie OntoDesign est utilisée pour manipuler les connaissances utilisées en conception et les agents servent à manipuler les connaissances hétérogènes et distribuées au fil de l'eau des projets.

Le SMA dans KATRAS est basé sur une approche organisationnelle afin de modéliser les rôles des humains. Cette modélisation est fondée sur le méta-modèle RIO auquel Monticolo a proposé d'associer les notions de compétence et connaissance. Cette approche de modélisation identifie les rôles des acteurs métier et présente la cartographie des connaissances qu'ils utilisent. Pour manipuler les connaissances hétérogènes et distribuées, KATRAS permet par ses agents d'identifier les connaissances à partir du suivi des actions des rôles des acteurs métier, (ii) d'annoter les connaissances en fonction de la sémantique de l'ontologie OntoDesign et (iii) de fournir une assistance aux acteurs métier afin de réutiliser les connaissances capitalisées et stockées dans le référentiel métier, c'est-à-dire un référentiel où sont stockées toutes les mémoires de projets terminés. Ils répondent aux requêtes des acteurs à partir des connaissances contenues dans le référentiel métier. Le référentiel métier est l'ensemble des mémoires de projet de l'entreprise.

2.3.3.5. Système de FU Rui-xue (FU et al., 2008)

FU Rui-xue a proposé un système de gestion des connaissances pour les petites et moyennes entreprises en basant sur les agents, les ontologies et une base de connaissances. Ce travail était dans le but de (i) traiter chaque étape du cycle de vie de la gestion des connaissances, (ii) de renforcer la capacité d'étendre et de restructurer les connaissances et (iii) de soutenir la capacité de réutilisation de systèmes existants dans les environnements distribuées et hétérogènes. Ce système de gestion des connaissances est composé de deux agences : la première est l'agence de connaissances qui permet d'apporter un soutien aux processus de gestion des connaissances, la deuxième est l'agence d'application pour l'application des connaissances.

Le système de FU Rui-xue gère alors le problème d'hétérogénéité des connaissances en récupérant, filtrant, et fusionnant les informations provenant de sources d'information hétérogènes et distribuées et acquiert des connaissances à l'aide de différentes techniques et outils, tels que les data mining et traitement analytique en ligne (OLAP). Ce système offre à ses utilisateurs (i) une aide en comprenant leurs centres d'intérêts et leurs besoins, (iii) de rechercher les connaissances de la base de connaissances, (iv) d'évaluer les connaissances, et vérifier si la connaissance est ancienne, puis de la supprimer ou non en la comparant avec les nouvelles connaissances, et (v) d'annoter les connaissances acquises et les stocker dans une base de connaissances et (vi) de les diffuser au sein de l'entreprise.

2.3.3.6. Yang 2011 (Yang and Chang, 2011)

Cette étude se concentre sur la façon de collecter des informations en basant sur la coopération des agents intelligents. Yang a proposé alors un système qui permet de renforcer et améliorer la performance de la surveillance du réseau pour constituer un système intelligent de gestion de réseau. Ceci permet de fournir un réseau rapide, pratique et profond pour les utilisateurs. Les résultats expérimentaux ont montré que ce système, basé sur les agents, arrive à reconnaitre précisément les alarmes de défaillance, et en plus réduire le temps de récupération de 61% lors d'un dépannage du réseau.

Pour arriver à ce but Yang a construit une série de diagrammes de suivi avec une ontologie. Cela permet de générer précisément l'intégralité des données du réseau par l'analyse et l'intégration d'un mécanisme d'agents intelligent (Chang & Lu, 2006). Il réduit aussi considérablement le chargement des données depuis le serveur. Le système de Yang permet aussi de mettre ne place des pages web pour répondre aux questions des utilisateurs sur des problèmes de réseau. Alors ce système écoute et récupère le message de l'utilisateur, recherche les données correspondant à sa requête de la base de connaissances depuis le serveur. Cette base dynamique a le format d'une ontologie qui se construit en collectant les données provenant de dispositifs de réseau différentes. Le système présente alors dans les pages web le résultat de la recherche avec un temps minima grâce à la rapidité des agents.

2.3.3.7. MAEKMS(Rahman et al., 2012)

Rahman a proposé une architecture d'un système de gestion des connaissances en utilisant la technologie d'agents. Ce dernier a été utilisé grâce à son efficacité dans la diffusion des connaissances dans les secteurs publics d'une manière facile.

L'architecture est appelée Multi Agent Enterprise Knowledge Management (MAEKM). Elle se compose de trois couches qui sont : la couche utilisateur GUI, la couche agent de Communication et la couche référentiel de connaissances. Ces couches assurent les fonctionnalités suivantes : (i) l'interaction entre le système et l'utilisateur pour extraire, créer, partager, et utiliser les connaissances, (ii) le traitement des connaissances, la recherche, la création et la production des résultats en temps minimum, (iii) le stockage des connaissances dans le référentiel de connaissances, (iv) la réutilisation des connaissances par les agents c'est à dire les agents peuvent réutiliser leurs connaissances personnelles si la requête demandée est très similaire à une ancienne requête.

Ce système a été mis en œuvre pour définir divers services tels que les informations de facturation, services de paiement, les services de traitement des plaintes, et les services clients.

2.3.3.8. Synthèse

Nous présentons dans le tableau ci-dessous (Tableau 1) les systèmes multi-agents dédiés à la gestion des connaissances que nous avons détaillées ci-dessus. Nous avons positionné chaque système en fonction des étapes du cycle de vie de la gestion des connaissances. Nous avons vu dans la section 2.2.1.3 qu'il existe plusieurs cycles de vie de la GC proposé par Alavi, Maier, Grundstein, … Chacun de ces chercheurs présente un cycle avec des étapes différentes de l'autre. Comme les approches agents que nous avons présentés ci-dessus (Kraft, FRODO, COMMA, ..) n'ont pas toutes adopté le même cycle de vie dans leur travail, il est difficile de les comparer. Pour cela, dans notre comparaison nous ne sommes pas limités à un seul cycle de vie. Pour ce faire, nous avons regroupés la plupart des étapes présentes dans la section 2.2.1.3 pour les différents cycles et nous les avons structurées dans le tableau ci-dessous en quatre étapes possédant des sous étapes. Les quatre étapes sont l'identification, la recherche, la diffusion et la réutilisation. La première consiste à définir les connaissances (les besoins) de l'utilisateur. La deuxième consiste à rechercher les connaissances définis et les

présenter à l'utilisateur. La troisième consiste à diffuser et partager les connaissances recherchées aux autres utilisateurs. La quatrième consiste à réutiliser les connaissances recherchées dans des futurs projets.

Si une étape est traitée par l'auteur, nous marquons (x). Nous indiquons aussi le type de stockage de connaissance pour chaque travail de recherche.

Le Tableau 1 montre que tous les auteurs commencent leur système, quel que soit le cycle de vie qu'ils ont adopté, par l'identification et l'acquisition des connaissances à capitaliser. Ensuite certains (CoMMA et KATRAS par exemple) ont formalisé c'est-à-dire structuré et puis organisé les connaissances pour faciliter leur utilisation par la machine. L'utilisation de connaissances identifiées peut être traduite par une fonction de recherche des connaissances d'une base de connaissances. Cette étape a été traitée par tous les auteurs du Tableau 1 mais chacun a choisi sa propre technique. Cette recherche peut faire appel à une transformation des modèles (par exemple FU Rui-xue a traduit la requête en syntaxe ontologique) afin d'extraire des connaissances d'une base de connaissances. Ceux qui traite cette fonctionnalité de recherche récupère le résultat de la recherche, le stocke dans des bases de type mémoire de projet ou mémoire organisationnelle par exemple, et puis présente le résultat de la recherche à l'utilisateur qui a lancé cette recherche. CoMMA, KATRAS, et FU Rui-xue traite la fonctionnalité d'annotation c'est-à-dire de contextualiser les connaissances. Le fait de mettre les connaissances dans un contexte facilite leur réutilisation. KATRAS, et FU Rui-xue voulaient s'assurer que les connaissances dans leur bases sont toujours pertinentes. Pour ce faire, ils ont introduit dans leur système l'évaluation de leurs connaissances par les utilisateurs afin de garantir leur pertinence. D'autres (voir Tableau 1) ont allé plus loin dans leur cycle en permettant à leur utilisateur de faire évoluer leurs connaissances en partant de l'idée que les connaissances évoluent en fonction du temps. Une connaissance pertinente aujourd'hui peut devenir une connaissance non valide dans l'avenir, pour cela ils ont laissé le choix aux utilisateurs de modifier ou supprimer une connaissance. Certains ont choisi de compléter le cycle de vie de la GC en partageant les connaissances. Ceci est un véritable gain du temps ce qui permet aux employés de travailler sur des tâches innovantes au lieu de prendre du temps à chercher des connaissances utilisés dans un ancien projet. Cette tâche est réalisée en permettant à leur système de diffuser les connaissances dans un site web, un forum, un wiki ou autre, dans le but d'une simple consultation ou d'une réutilisation dans le futur.

Tableau 1: Comparaison entre les SMA pour la GC

Approche / Étape	Kraft	Frodo	Comma	KATRAS	FU Rui-xue	Yang	MAEKMS
Identification							
Acquisition	X	X	X	X	X	X	X
Formalisation			X	X			X
Organisation	X			X			X
Recherche							
Extraction	X	X	X		X	X	X
Traduction/transformation	X				X		
Annotation			X	X	X		
Stockage	X	X	X	X	X	X	X
Présentation	X	X	X		X	X	X
Diffusion							
Evaluation				X	X		
Evolution		X		X	X		X
Diffusion	X	X		X	X		X
Réutilisation	X	X		X			X
Stockage base	Base de connaissances	Mémoires organisationnelles	Mémoire de projet	référentiel métier	Base de connaissances	Une base d'ontologies	Référentiel de connaissances

2.4.Conclusion

Dans ce chapitre nous avons réalisé un tour d'horizon du domaine de la gestion des connaissances hétérogènes et distribuées, du rôle que jouent les ontologies et des systèmes multi-agents dans ce domaine de la gestion des connaissances.

Nous avons commencé par une présentation du domaine de la gestion de connaissance et l'intérêt qu'il occupe aujourd'hui dans les entreprises qui souhaitent capitaliser leurs capitaux intellectuels. Pour arriver à ce but, les entreprises doivent concevoir et développer un système de gestion de connaissance. Ce système doit assurer les fonctions nécessaires pour capitaliser les connaissances de l'entreprise. Ces fonctions dépendent du besoin de chaque entreprise. Pour cela, chacune choisit ou crée le cycle de vie de la connaissance qui lui convient. Nous avons présenté dans la première section de ce chapitre quelques cycles de vies existants. Puis nous avons passé à la gestion des connaissances hétérogènes et distribuées.

Nous avons vu qu'il y a beaucoup des travaux et des recherches sur ce sujet dans tous les domaines. Le but de ces travaux est de proposer des systèmes qui permettent de gérer une grande quantité des informations, provenant de plusieurs sources de données et distribuées sur des ordinateurs différents à travers le réseau de l'entreprise ou de la communauté. Ces systèmes offrent aussi à ces utilisateurs de créer, organiser, collecter, rechercher, diffuser, partager et réutiliser les informations et les connaissances depuis les bases respectives. Pour ce faire certains ont utilisé des ontologies pour avoir un vocabulaire commun, d'autres ont créé une nouvelle base de données pour regrouper toutes les données dans un seul format, etc. La plupart des travaux utilisent les ontologies pour résoudre ce genre ce problème. Dans cette thèse nous avons aussi utilisé les ontologies pour gérer le problème de la gestion des connaissances multi-sources. Cette technologie a montré son importance comme un langage de représentation des connaissances hétérogènes et distribuées, et comme un apport sémantique grâce aux contextes qu'elles expriment. Cette sémantique permet aux connaissances d'être compréhension non pas seulement par les humains mais aussi par les machines ce qui améliore la qualité des connaissances retournées aux utilisateurs.

Ensuite nous avons introduit le rôle des systèmes multi-agents pour la gestion des connaissances. Cette technologie a montré aussi sa capacité de gérer des systèmes complexes tels que la gestion des connaissances hétérogènes et distribuées. Ceci est possible grâce aux agents qui sont autonomes, qui coopèrent et communiquent entre eux pour décomposer les problèmes en sous-problèmes et les résoudre plus facilement. Nous avons terminé ce chapitre par une présentation de quelques approches agents existants pour la gestion des connaissances et puis les comparants selon différents critères. Nous avons vu dans le Tableau 1 que les approches agents présentés ne traitent pas intégralement toutes les étapes citées dans ce tableau et par conséquent ne couvrent pas un cycle complet de la gestion des connaissances.

Dans les chapitres qui suivent, nous présentons notre contribution dont le but est de combler ces lacunes.

Deuxième Partie

CONTRIBUTIONS

CHAPITRE 3

PERSPECTIVE GLOBALE

Ce chapitre a pour objectif de présenter un aperçu global du SMA et des hypothèses sous-jacentes. Ces hypothèses et les mécanismes qui en découlent permettent au SMA d'aider les concepteurs dans leurs tâches quotidiennes au travers d'un cycle particulier de gestion des connaissances.

Table de matières

3.1. Introduction

Comme nous l'avons expliqué précédemment, la connaissance, qui provient de différentes applications métiers et qui est utilisée lors des projets de développement de produits, devraient être capitalisée afin de la structurer et la réutiliser pour fournir aux acteurs métiers une forme d'aide à la décision.

Pour ce faire, notre proposition est de définir et construire un SMA qui aura en charge cet objectif. L'intérêt de l'utilisation d'un SMA dans ce cadre est de pouvoir bénéficier des capacités d'autonomie, de réactivité, proactivité et habilité sociale des agents. Ces capacités permettent d'automatiser en partie la gestion des connaissances qui proviennent de différentes applications métiers. De plus, les SMA sont naturellement adaptés à la modélisation et implémentation d'un système distribué, tel que le système d'information de l'entreprise étendue, pris en compte dans cette thèse.

Nous proposons dans ce chapitre de définir l'architecture globale du SGC, réalisé sous la forme d'un SMA, qui assure le cycle de vie des connaissances composé comme le montre la Figure 2. Des processus de : définition, extraction, validation et réutilisation. Nous avons appelé ce SGC OCEAN (Ontology Creator, Extractor & Annotator kNowledge) (Lahoud et al., 2010).

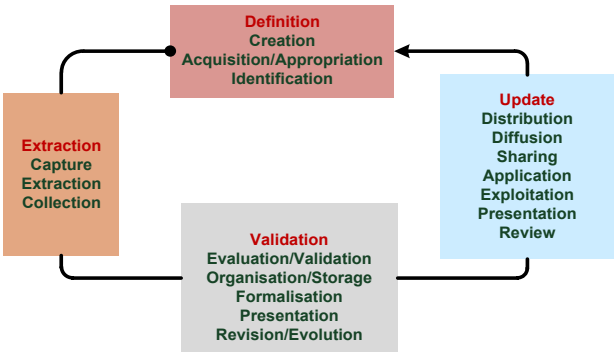


Figure 2: Cycle de vie de la gestion des connaissances

Pour définir OCEAN, nous proposons une approche méthodologique basée sur ASPECS (Cossentino et al., 2010) dédiée à l'analyse, la conception et le déploiement de systèmes complexes. Cette analyse permet de mettre en évidence les objectifs d'un SGC et les principaux mécanismes de son fonctionnement. Parmi les activités d'ASPECS, l'identification des besoins se fait par une approche orientée buts (Bresciani et al., 2004) qui va permettre la modélisation des objectifs d'OCEAN ainsi que les utilisateurs de ce système ainsi que les

parties du SMA en charge des différents objectifs. L'idée à la base de cette méthodologie est, sur la base d'une analyse des besoins, de définir les principaux éléments du problème et la structure organisationnelle du SMA qui va satisfaire les besoins.

Ce chapitre présente tout d'abord dans la section 3.2 les principes généraux d'ASPECS qui seront utilisés dans ce chapitre. La section 3.3 est dédiée à l'analyse du système OCEAN en appliquant les principales activités d'ASPECS. Enfin, la section 3.4 conclut ce chapitre.

3.2.Les principes généraux

ASPECS est un processus d'ingénierie logicielle spécialement adapté aux SMA en vue de déploiement dans le cadre de systèmes complexes. Le processus couvre les phases, depuis l'analyse des besoins jusqu'à la production du code et au déploiement de celui-ci sur une plate-forme spécifique. Les concepts de base de ce processus sont basés sur le méta-modèle CRIO (Gaud, 2007) qui définit les principaux éléments d'analyse, de conception et d'implantation des SMA.

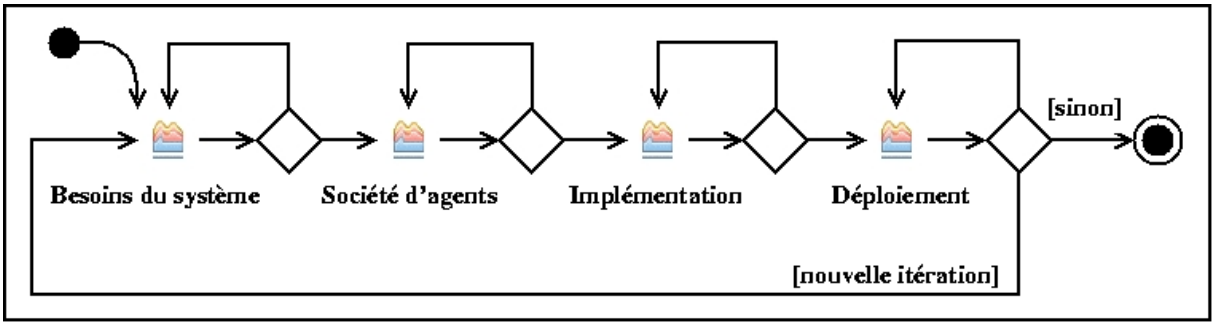


Figure 3: Phases du processus ASPECS

Les phases du processus sont illustrées par la figure ci-dessus. L'analyse des besoins vise à fournir une description organisationnelle du système (décomposition hiérarchique du système). Elle doit également collecter les connaissances disponibles sur le domaine du problème et les organiser au sein d'une ontologie de domaine.

La conception d'une société d'agents a pour but de construire le modèle d'un SMA, dont le comportement global doit être en mesure de fournir une solution au problème décrit dans la phase précédente. Les connaissances sur le système sont affinées et intègrent les éléments spécifiques à la solution proposée.

L'implémentation de la solution décrit l'architecture des agents impliqués dans la solution et doit fournir le code source de l'application.

Le déploiement de la solution constitue la phase finale en charge du déploiement de l'application sur la plate-forme choisie.

Le langage de modélisation adopté est UML. Afin de pleinement satisfaire les objectifs et les besoins spécifiques à l'approche orientée-agents, la sémantique et les notations d'UML ont été étendues, et de nouveaux « profiles » UML ont notamment été introduits.

Chaque phase d'ASPECS est composée d'activités qui s'enchainent selon l'ordre schématisé par la Figure 3.

3.2.1. Description des besoins

ASPECS débute par une description des besoins du domaine. L'objectif consiste à dresser une première description du contexte de l'application et de ses fonctionnalités. Cette activité vise à identifier, classifier et hiérarchiser l'ensemble des buts et des différentes parties prenantes du projet. Elle doit également fournir une première estimation du périmètre de l'application ainsi que de sa taille et de sa complexité. L'analyse des besoins est basée sur une approche orientée buts (Yu, 1997). Cette approche est basée sur l'identification des buts du système, leur décomposition hiérarchique, et les relations de dépendance et décomposition qui existent entre acteurs externes, acteurs internes qui représentent les différents composants du système.

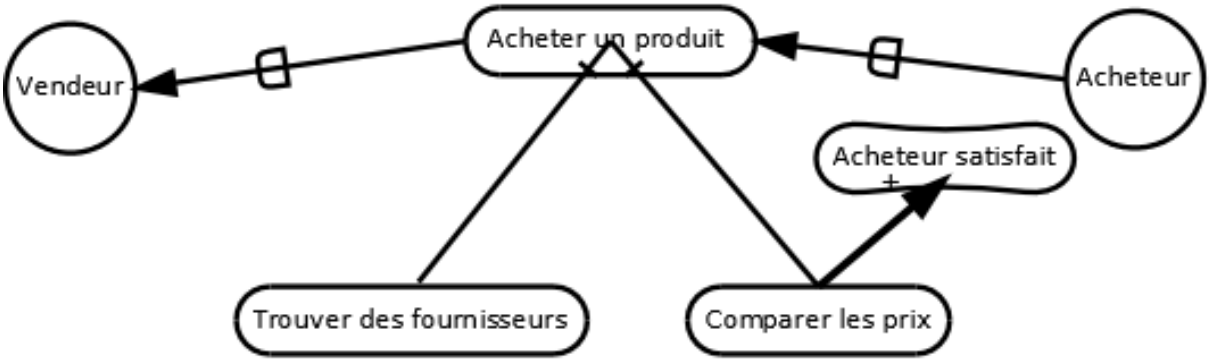


Figure 4: Exemple de description des besoins

La Figure 4 montre un exemple de diagramme illustrant une description des besoins en utilisant une approche orientée but. On distingue l'acteur « *Acheteur* » (représenté par un cercle) qui a pour but « Acheter un produit ». Pour ce but, l'acteur « *Acheteur*» dépend de l'acteur « *Vendeur* ». Ce but est de type hardgoal (présenté par un rectangle à coins arrondis), ce qui signifie qu'il possède un critère clair de satisfaction. Les liens de dépendances sont représentés par des arcs orientés et étiquetés par un D. L'origine de l'arc est l'entité qui

dépend de l'entité destination de l'arc. Par ailleurs, le but « Acheter un produit » peut se décomposer en deux sous-buts : « *Trouver des fournisseurs* » et « *Comparer les prix* ». Le but « *Comparer les prix* » contribue positivement au but « *Client satisfait* ». Ce dernier but est de type Softgoal (présentés sous forme de nuages). Pour ces derniers, il n'existe pas de critères clairs permettant de dire s'ils sont satisfaits ou non.

3.2.2. Description de l'ontologie du problème

Une fois les objectifs de l'application déterminés, la deuxième activité consiste à conceptualiser l'ensemble des connaissances disponibles sur l'application et son contexte sous la forme d'une ontologie de problème. Cette ontologie a pour rôle de fournir une première définition du contexte de l'application, du vocabulaire spécifique au domaine et des relations entre concepts afin de pouvoir les manipuler par la suite et guider le choix des organisations en charge de la réalisation des objectifs du système. Elle vise également à approfondir la compréhension du problème, en complétant l'analyse des besoins. L'ontologie est décrite à l'aide d'un diagramme de classes avec les stéréotypes suivants : *concepts*, *actions* et *prédicats*.

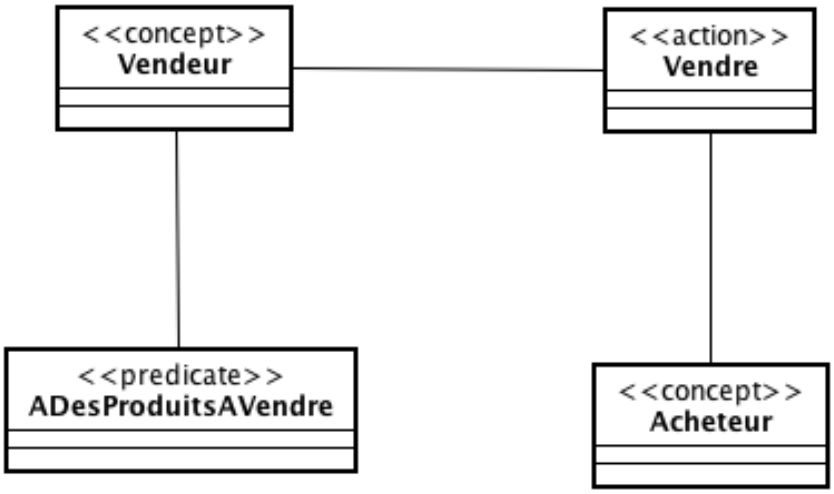


Figure 5: Exemple de fragment de l'Ontologie du problème

Dans l'exemple présenté ci-dessus (Figure 5), on distingue deux concepts : Vendeur et Acheteur. Un prédicat est associé au concept Vendeur et indique si un Vendeur a des produits à vendre. Enfin une action Vendre conceptualise le fait qu'un Vendeur vend un ou des produits à un Acheteur.

3.2.3. Identification des organisations

L'identification des organisations doit établir une première décomposition organisationnelle du système et définir les objectifs de chaque organisation. Chacun des buts, identifiés dans la première activité, se voit associer une organisation incarnant le comportement global en charge de le satisfaire ou de le réaliser. Le contexte de chacune de ces organisations est défini par un sous-ensemble des concepts de l'ontologie du problème. Le résultat de cette activité est ainsi incarné par un ensemble de triplets associant des concepts de l'ontologie, un ou plusieurs buts et une organisation. Les organisations, ainsi identifiées, sont directement ajoutées au diagramme de buts, sous la forme de paquetages stéréotypés englobant les buts que ces organisations sont en charge de satisfaire. Cette étape de l'activité d'identification des organisations permet de fixer les objectifs d'un premier ensemble d'organisations.

ASPECS se basant sur un processus itératif, cet ensemble d'organisations peut ensuite être complété au fur et à mesure des itérations, afin de déterminer la hiérarchie organisationnelle représentant le système. Cette décomposition hiérarchique du système se poursuit jusqu'à un niveau où la complexité des comportements (rôles) est suffisamment faible pour être exécutée par des entités considérées comme atomiques et facilement implémentables. Les liens de composition (comportementale et hiérarchique) entre les organisations sont décrits par des contraintes dans les diagrammes de classe.

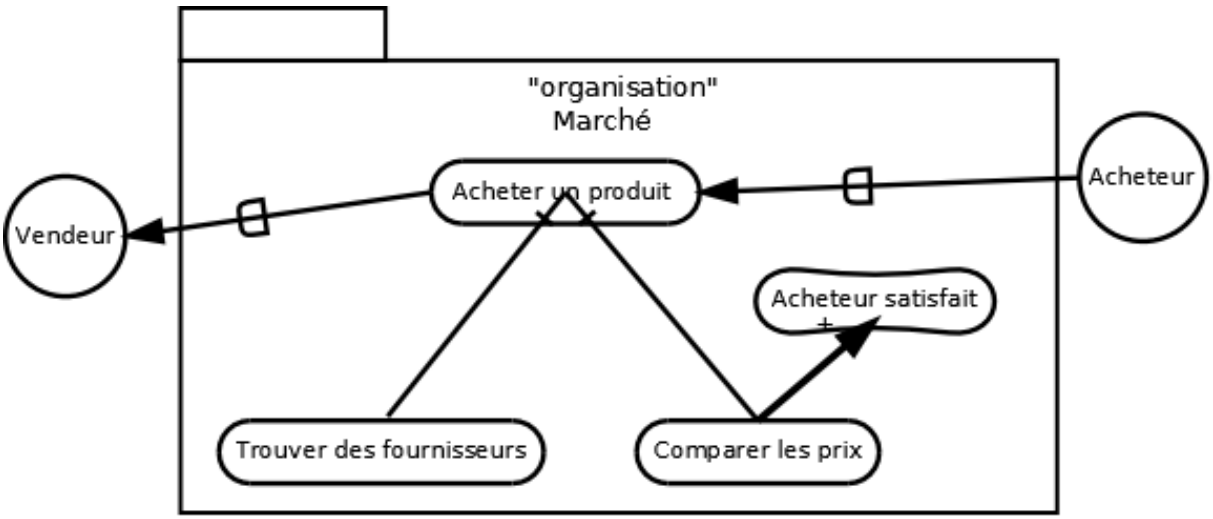


Figure 6: Exemple d'organisation

Dans l'exemple présenté ci-dessus (Figure 6), on a identifié l'organisation Marché pour la résolution des différents buts. Le paquetage représentant l'organisation englobe les buts dont elle a la charge.

3.2.4. Identification des interactions et des rôles

Le contexte et les objectifs de chaque organisation sont désormais connus. L'identification des interactions et des rôles vise à décomposer le comportement global incarné par une organisation en un ensemble de rôles en interaction. D'après Dieng (DIENG et al., 1999): *Un rôle est une abstraction d'un comportement dans un contexte précis et confère un statut dans l'organisation. Le rôle donne à l'entité qu'il interprète le droit d'exercer ces capacités*'. Un rôle interagit avec les autres rôles de l'organisation afin d'accomplir leurs tâches.

Cette activité doit décrire les responsabilités de chaque rôle dans la satisfaction des besoins associés à leurs organisations respectives. Chaque rôle est associé à un ensemble de concepts dans l'ontologie, généralement une sous-partie de ceux associés à son organisation. Grâce aux Boundary Roles, le périmètre de l'application et la frontière entre le système et son environnement sont ici affinés. Les rôles et les interactions composant chaque organisation sont ajoutés à leurs diagrammes de classe. Un rôle est représenté par une classe stéréotypée, et une interaction entre deux rôles est représentée par une association entre les classes des rôles correspondants.

Pour chaque niveau de la hiérarchie organisationnelle, la complexité des rôles identifiés est étudiée. Si celle-ci est considérée comme trop importante pour être gérée par une entité atomique, une nouvelle organisation de niveau d'abstraction inférieur devra alors être identifiée pour répartir la complexité du rôle. Cette identification engendre une nouvelle itération avec l'activité d'identification des organisations.

L'objectif consiste à décomposer les tâches, considérées comme trop complexes au niveau n de la hiérarchie, en un ensemble de tâches plus simples, dont la résolution est répartie entre les rôles du niveau n-1. Chacun de ces rôles de niveau n-1 dispose ainsi d'une complexité inférieure au rôle de niveau n. Les tâches de niveau n sont en fait résolues de manière collaborative par un ensemble de rôles de niveau n-1. Le système est ainsi successivement décomposé niveau par niveau, jusqu'à atteindre un niveau où la complexité des tâches à effectuer est considérée comme suffisamment faible, pour être gérée par des

entités atomiques faciles à implanter. Les contributions entre chaque niveau seront détaillées ultérieurement. Le processus de décomposition d'un système dans ASPECS est basé sur la définition récursive du concept d'organisation. Selon le niveau d'abstraction considéré, une organisation peut être vue, soit comme un comportement unitaire, soit comme un ensemble de comportements en interaction. Cette dualité permet de briser la complexité d'un comportement de niveau n en la répartissant parmi un ensemble de comportements en interaction au niveau n-1.

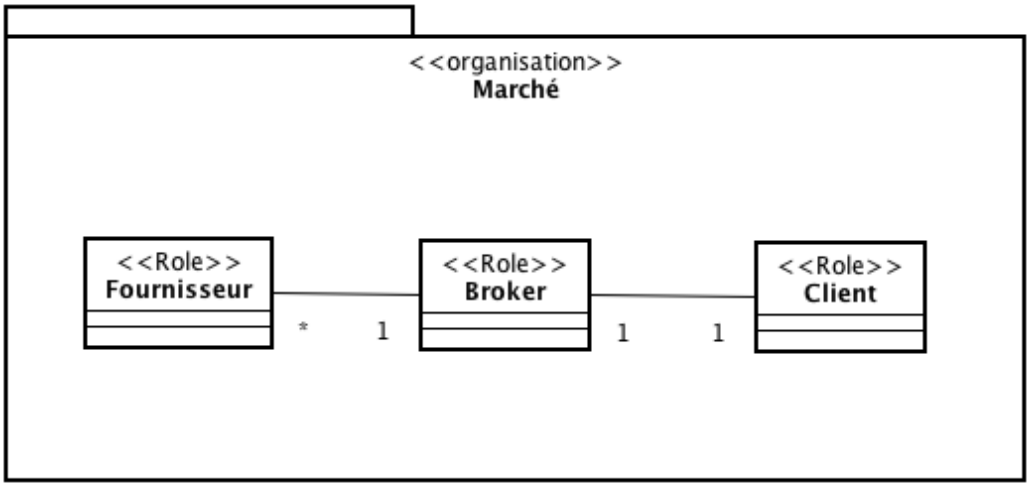


Figure 7: Description des interactions et des rôles dans l'organisation Marché

Dans l'exemple présenté ci-dessus (Figure 7), on distingue l'organisation «Marché », et les rôles qui la composent. Ces rôles sont : « *Client*», « *Broker* » et « *Fournisseur* ». Chaque « *Client*» s'adresse à un « *Broker* » qui va chercher parmi les « *Fournisseur* » connus celui proposant un prix satisfaisant.

3.3.Analyse du problème

3.3.1. Identification des besoins

OCEAN est un système de gestion des connaissances qui a pour but principal de gérer les connaissances multi-sources lors du processus de conception de produits. Dans ce cadre, l'analyse des principaux travaux sur la gestion des connaissances met en avant quatre processus principaux (Figure 2) : la définition des connaissances, l'extraction des connaissances, la validation des connaissances et la réutilisation des connaissances. On peut assimiler ces processus à des buts globaux, de type soft-goal que tout SGC doit atteindre (au

moins en partie). En effet, il semble difficile de définir des critères précis qui permettent d'établir si ces buts sont atteints ou non.

Ces quatre soft goals contribuent donc positivement à la réalisation du soft goal du système de gestion des connaissances « gérer les connaissances ». De plus, ils sont liés par la décomposition ET (Figure 8) car c'est par leur combinaison que le SGC peut tendre au but global.

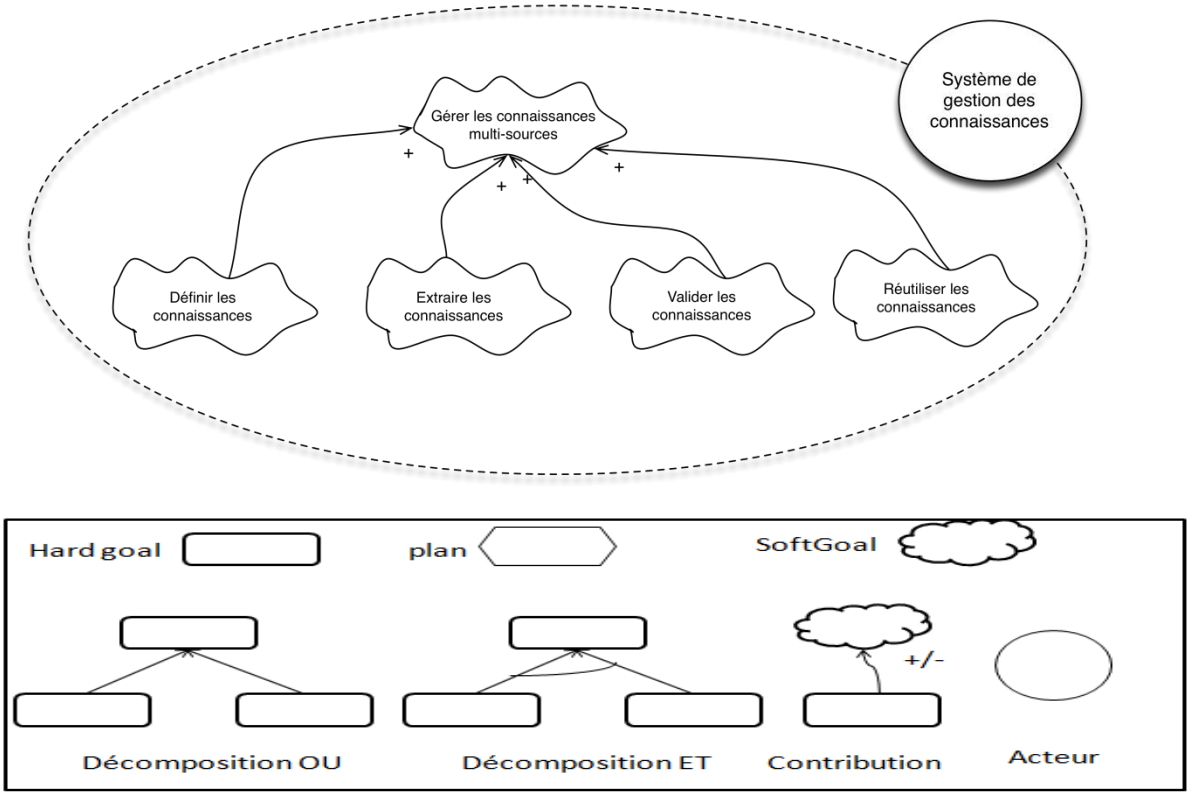


Figure 8: Architecture globale de la GC

3.3.2. Ontologie du problème

L'objectif global de la description de l'ontologie du problème est de fournir une vue conceptuelle d'ensemble du problème en cours d'étude. Cette activité approfondit la compréhension du problème avec une description des concepts qui composent le domaine du problème. L'ontologie du problème est modélisée en utilisant un diagramme de classe où les concepts, les attributs et les actions sont identifiés par des stéréotypes spécifiques (FIPA ACL, 2002). Ces stéréotypes sont :

- «concept» : pour désigner une entité du domaine,
- «action» : pour désigner une transformation d'un concept,
- «predicate» : pour désigner un prédicat relatif à un ensemble de concepts.

Le diagramme UML de la Figure 9 présente notre proposition d'ontologie relative au domaine d'intérêt, à savoir notre système OCEAN.

L'hypothèse à la base de cette ontologie est que l'entreprise étendue est engagé dans des projets visant à produire en sortie des produits. Chaque projet a un produit spécifique en sortie. Ces deux éléments sont représentés par les classes *Projet* et *Produit*. Un *Produit* peut être suffisamment complexe pour être décomposable en sous-unités ou sous-produit. Cette relation est matérialisée par le lien de composition avec l'élément *SousProduit.*

A chaque *Projet* est affecté un ensemble de rôles. Chaque rôle représente le comportement d'un acteur métier. Un rôle permet de déterminer le comportement d'un acteur durant l'activité au travers de ses interactions, ses compétences et des connaissances qu'il utilise et/ou qu'il partage (Monticolo, 2008). Un acteur peut jouer plusieurs rôles et un rôle peut être joué par plusieurs acteurs. La classe *Rôle* représente ce concept qui est lié par le prédicat *ContribueA* au concept *SousProduit*. Ce prédicat permet de spécifier les liens de contribution d'un rôle vis-à-vis d'une partie du *Produit.* Les délivrables des *Rôles* sont matérialisés par la production de *Donnée*. Ce dernier concept représente le résultat du travail que les acteurs métiers effectuent afin de définir les *SousProduits*. L'hypothèse faite dans ce travail est que ces *Données* sont produites à l'aide d'outils logiciels spécifiques. Chaque *OutilLogiciel* est lié à certains *Rôles* qui les utilisent. Ce lien est conceptualisé par l'action *GénérerSousProduit.*

Chaque *Rôle* est également associé à une ontologie produite par un expert du domaine et qui permet de définir les connaissances pertinentes pour le contexte du *Rôle* concerné. Ce contexte représente l'implication du *Rôle* dans le projet et plus spécifiquement les données que le *Rôle* doit produire pour définir les *SousProduits.* Cette ontologie qui est une version simplifiée des ontologies classiques est définie par un ensemble de concepts et de relations : respectivement les classes *GCOntologie*, *GCConcept* et *GCRelation.* Le lien de représentation entre une *GCOntologie* et une *Donnée* est spécifié par le prédicat *Représente.*

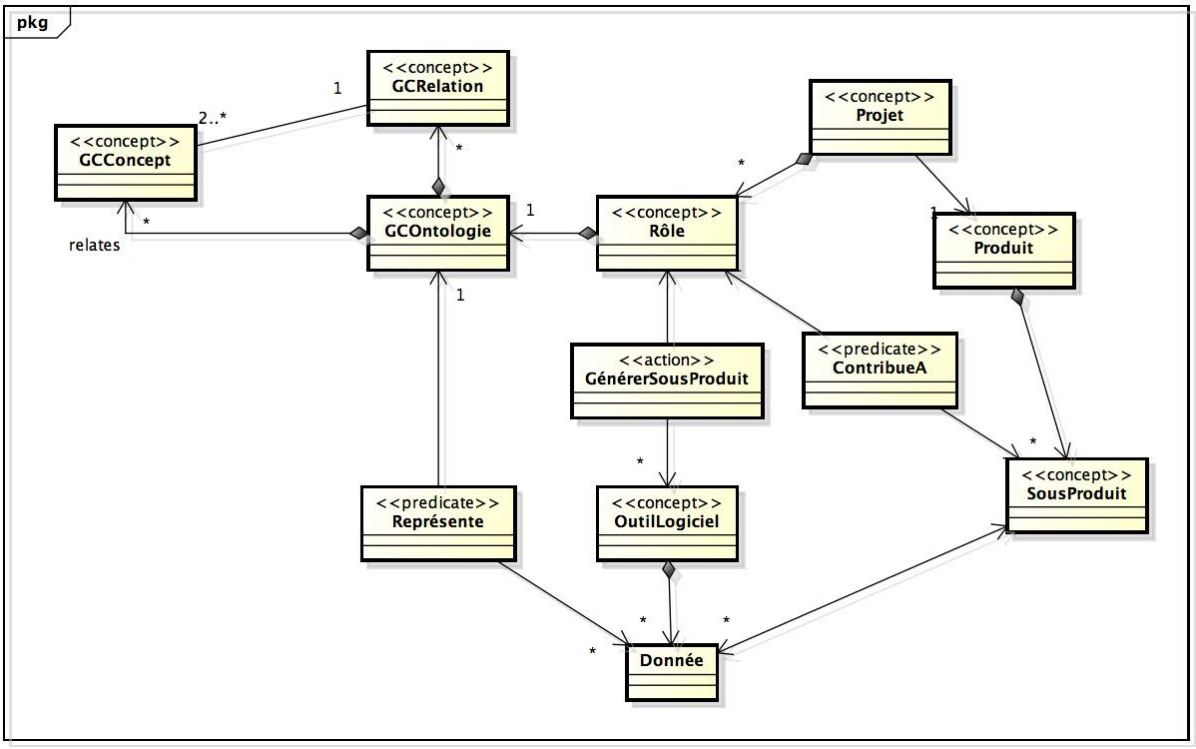


Figure 9: Notre ontologie relative au domaine d'intérêt

3.3.3. Identification des Organisations

Dans l'activité d'identification des organisations, chaque but est affecté à une organisation qui va avoir pour objectif, par interaction des rôles qui vont composer l'organisation, de satisfaire ce but. Les buts identifiés, Figure 8, sont affectés à des organisations selon les relations décrites dans la Figure 10. Une organisation globale, appelée OCEAN, représente les interactions pour le SMA complet. Cette organisation se décompose en quatre sous-organisations.

La première, *Définition*, a pour objectif de satisfaire le but de définir les connaissances qui vont être extraites pour être réutilisées. Pour cela, une ontologie doit être définie par l'organisation afin de représenter un ensemble de *Donnée* liées à un *Rôle* métier et un *OutilLogiciel*. Ce but, avec les hypothèses que nous avons adopté, est raffiné par une contribution positive de trois sous-buts de type ET. Ces sous-buts sont l'identification des connaissances, la détection des similitudes et la validation. Cette sous-organisation est le sujet du chapitre 4.

La deuxième sous-organisation d'OCEAN, *Extraction*, a pour objectif, à partir des ontologies définies par l'organisation *Définition* de fournir des connaissances extraites des

projets existants aux concepteurs. Cette extraction, qui fera le sujet du chapitre 5, est basée sur la transformation des ontologies définissant les connaissances vers un langage d'interrogation pour un ensemble de *Donnée* produit par un *OutilLogiciel.* Les résultats de cette interrogation sont annotés et ensuite proposer à l'utilisateur. Transformation, interrogation et annotation sont les trois sous-buts de '*Extraire les connaissances'* et sont également à la charge de l'organisation *Extraction.*

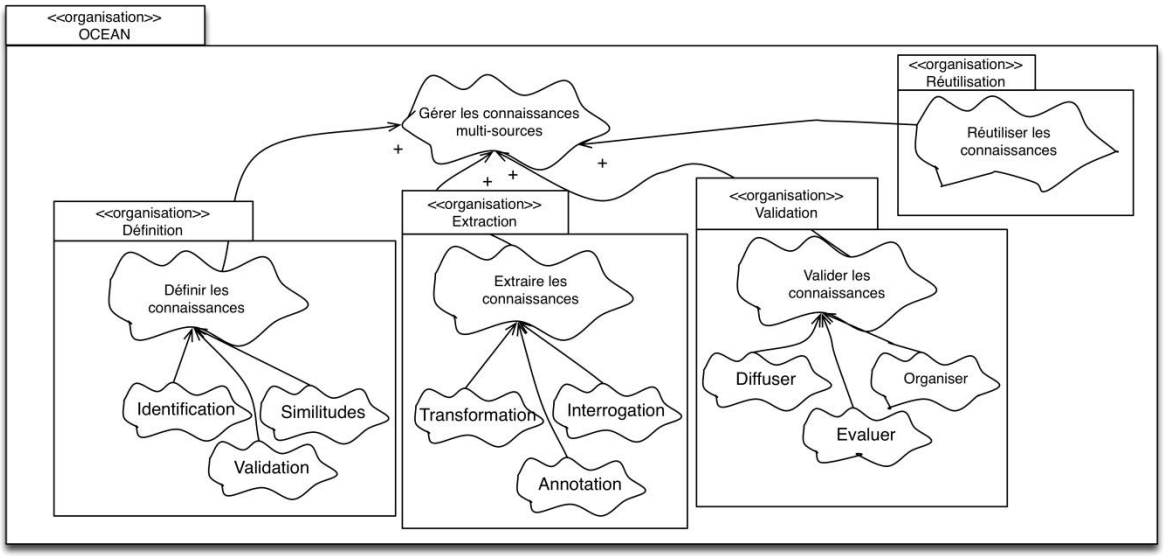


Figure 10: L'organisation globale" OCEAN"

La troisième sous-organisation, *Validation*, a pour objectif de satisfaire le but *Valider les connaissances*. Nous détaillons cette sous-organisation dans le chapitre 6. Cette sous-organisation diffuse les connaissances annotées issues de l'organisation *Extraction* et les met en forme. Ces connaissances formatées vont permettre par le biais de recherche manuelle et d'évaluation de la part des concepteurs d'évaluer les connaissances.

La dernière organisation, *Réutilisation*, n'est pas détaillée dans cette thèse car nous nous basons sur les travaux issus de (Ben Miled, 2011), (Ben Miled et al., 2008) et (Ben Miled et al., 2009) qui définissent une approche à base d'agents dont les hypothèses sous-jacentes sont similaires à celles adoptées dans ce travail. Les travaux cités supposent toutefois l'existence d'une mémoire organisationnelle qui contient les connaissances déjà capitalisées. Cette mémoire organisationnelle est le résultat des trois premières organisations (*Définition*, *Extraction* et *Validation*). Les deux premières organisations consistent à construire et enrichir

cette mémoire organisationnelle par des connaissances. La troisième organisation, *Validation*, s'appuie sur cette mémoire pour d'une part diffuser les connaissances aux concepteurs et d'autre part les valider. La quatrième organisation, *Réutilisation*, permet de réutiliser les connaissances de cette mémoire organisationnelle lors de projets.

3.3.4. Identification des Rôles et Interactions

L'organisation *Définition* (Figure 11) est composée de trois rôles. Le premier rôle *Expert* représente les experts du domaine en charge de la définition et de la manipulation de l'ontologie métier. Cette ontologie est représentée par le rôle *GCOntologie.*

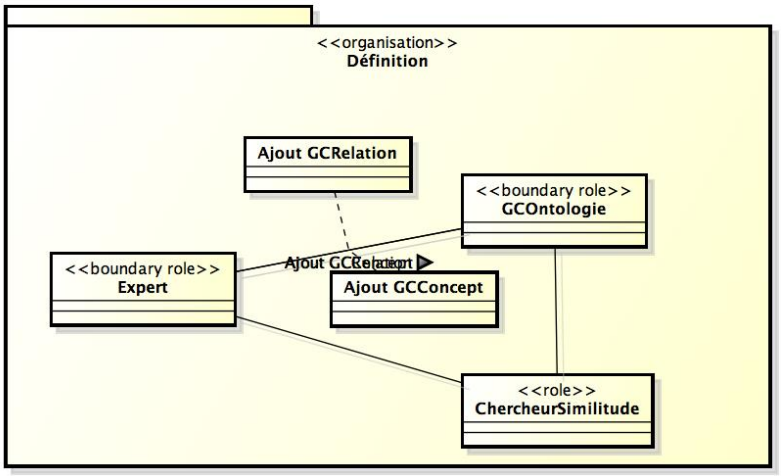


Figure 11: Sous-organisation "définition"

Le rôle *Expert* ajoute des concepts et des relations à l'ontologie *GCOntologie.* A chaque nouvel ajout ou modification, le rôle *ChercheurSimilitude* déclenche un mécanisme de recherche de concepts ou relations similaires par synonymie, et propose les résultats à l'*Expert* pour validation.

L'organisation *Extraction* (Figure 12) est composée de quatre rôles. Le premier rôle est *GCOntologie* et représente l'ontologie métier qui définit les connaissances pour un ensemble de *Donnée*. Cette définition des connaissances est manipulée par le rôle *Transformer* afin de produire une requête exprimée dans un langage d'interrogation adapté pour un *OutilLogiciel*. Cette requête est exécutée par le rôle *Interroger* qui reçoit les résultats en provenance de l'ensemble de Donnée correspondant à l'*OutilLogiciel*.

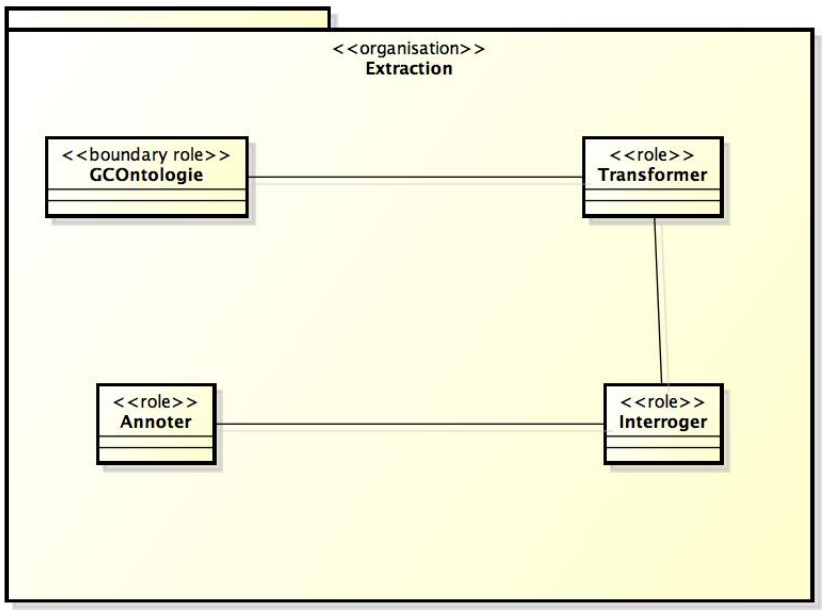


Figure 12: Sous-organisation "extraction"

Le rôle *Annoter* a pour comportement d'annoter les résultats de l'interrogation en fonction du contexte défini par l'*OutilLogiciel,* le *Rôle,* le *Projet* et le *Produit.*

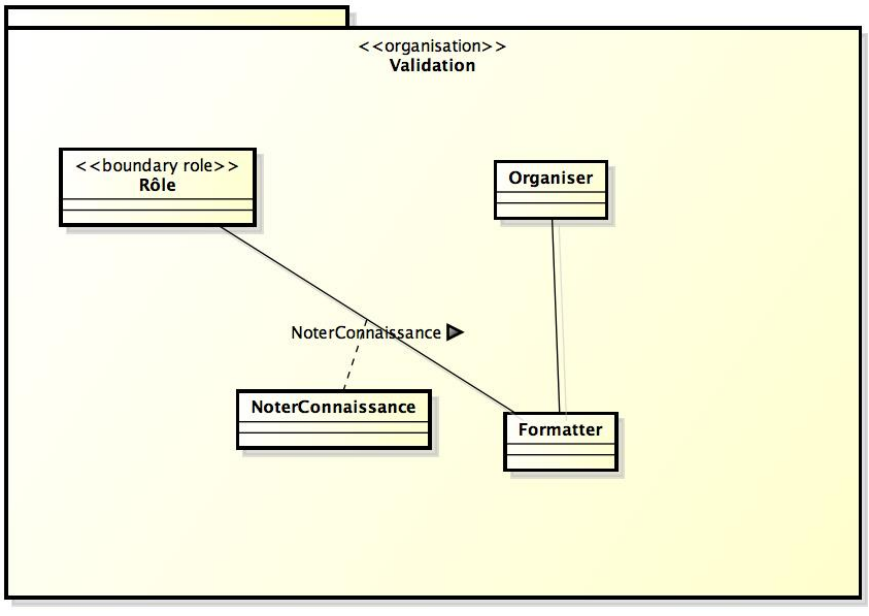


Figure 13: Sous-organisation "validation"

L'organisation Validation (Figure 13) est composée de trois rôles. Le premier rôle, *Organiser,* traite les connaissances annotées et opère un tri sur ces connaissances afin de les structurer. Le rôle *Formater* présente les connaissances au travers d'une interface graphique et permet de faire des recherches manuelles (par l'entrée de mots-clés). Enfin, le rôle *Rôle*

représente un concepteur qui utilise OCEAN et qui peut donner une évaluation des connaissances présentées. En fonction de l'ensemble des évaluations un algorithme de ranking permet de trier les connaissances.

3.4.Conclusion

Ce chapitre présente un aperçu global de notre approche pour la gestion des connaissances à base de SMA en appliquant une méthodologie orientée agent d'analyse et de développement, ASPECS. Les résultats d'activités d'ASPECS, sont :

- une définition des besoins d'un SGC appelé OCEAN sous forme d'arborescence de buts,
- une ontologie de domaine qui conceptualise nos hypothèses,
- un ensemble d'organisations composées de rôles en interactions, dont le comportement global vise à la satisfaction des buts du système.

Les buts sont assignés à ces organisations pour former l'architecture globale d'OCEAN reposant sur nos hypothèses, c'est-à-dire, l'existence de plusieurs applications métiers utilisés par des acteurs dans le cadre d'un même projet de conception de produit. Chaque type d'acteur ou *Rôle* est associé à une ontologie de domaine qui définit les connaissances pertinentes relatives à ce domaine et qui sert de base à la création de transformations pour l'extraction des connaissances issues des données des applications métiers. La définition de ces ontologies est faite par des experts du domaine avec l'aide de notre système OCEAN. Les transformations, la validation et la réutilisation se font de manière automatique ou manuelle à l'aide d'OCEAN.

Cet aperçu global, en particulier les organisations et les rôles qui les composent vont être raffinés et détaillés dans les chapitres qui suivent pour traiter en détails les contenus et principes de chaque partie de ce système.

CHAPITRE 4

LA DEFINITION DES CONNAISSANCES ET LA CREATION D'ONTOLOGIES

Ce chapitre décrit l'approche que nous avons définie pour identifier les connaissances issues des différents modèles métier réalisés lors des projets de développement de produit. L'idée sous-jacente est d'assister les experts métier à formaliser leurs connaissances en créant des ontologies de domaine. Ces ontologies sont créées dans le système OCEAN (Ontology Creator, Extractor & Annotator kNowledge).

Table de matières

4.1. Introduction

L'objectif de ce chapitre est de traiter le problème de la définition et la représentation des connaissances. Le processus de définition des connaissances, consiste à formaliser, structurer, classer, regrouper les connaissances que l'on souhaite capitaliser (Dieng et al., 2005). Les connaissances définies peuvent être représentées à l'aide de concepts. Dans cette thèse, nous utilisons les ontologies pour spécifier les connaissances et définir une sémantique particulière pour décrire leurs relations (Uschold and Gruninger, 1996). Cette technique nous permet de définir, pour un domaine et un problème donnés, la signature fonctionnelle et relationnelle d'un langage formel de représentation et la sémantique associée (Bachimont, 2000).

Un système multi-agent peut exploiter et manipuler les connaissances décrites dans une ontologie. Il peut les transmettre et éventuellement les expliquer (De Azevedo and Barthès, 1997). Toutefois, le développement de systèmes experts a montré que la difficulté ne résidait pas dans la manipulation des connaissances une fois représentées, mais dans la construction du modèle de connaissances (ontologie). Cette idée a été confirmée par Motta (MOTTA et al., 2000) qui considère que la conception d'une ontologie est une tâche difficile même pour un spécialiste du domaine. Pour aider les experts métier, nous avons développé, un module d'aide à la création d'ontologies dans OCEAN pour faciliter l'acquisition des connaissances. Ce module vise à aider les experts métier à formaliser les connaissances du domaine et à préciser les connaissances souhaitées pour la capitalisation. Ces connaissances du domaine sont complètement distribuées dans le réseau d'information de l'entreprise. De plus, les connaissances sont, par nature, hétérogènes puisqu'elles sont issues de différentes sources d'information comme les logiciels, les rapports techniques, les comptes rendus de réunion, … L'utilisation d'agents est donc adapté à résoudre ce problème complexe qui est la capitalisation de connaissances hétérogènes et distribuées.

Dans ce chapitre, nous présentons, dans un premier temps, la méthodologie de création des ontologies dans OCEAN et le rôle des agents pour gérer ce module. Puis, nous décrivons notre module d'aide en soulignant l'intérêt d'avoir un tel module pour aider les experts métier à créer leur ontologie. Nous terminons ce chapitre par une présentation de notre technique de sauvegarde des ontologies créées par les experts.

4.2.La définition des connaissances et la création des ontologies

Dans ce chapitre, nous allons présenter comment le système multi agents OCEAN assiste les experts métier à : formaliser leurs connaissances du domaine, les organiser et les représenter dans une ontologie. En effet, les acteurs métier peuvent définir, à la fois, leur propre vocabulaire et une sémantique propre pour décrire les connaissances qu'ils souhaitent capitaliser et réutiliser. La pertinence de la structure de l'ontologie conditionne l'efficacité de l'exploitation des connaissances. En effet, les ontologies créées vont être utilisées par les agents pour réaliser l'extraction des connaissances dans les applications métier, leur diffusion et leur réutilisation.

4.2.1. La définition des connaissances

Notre approche de définition des connaissances se base sur la création d'ontologies. Ces dernières décrivent les concepts utilisés par les acteurs métiers pour réaliser un projet et/ou concevoir un produit. Toutefois, le module de définition des connaissances est dédié aux experts, c'est-à-dire aux acteurs qui ont une expertise suffisante pour décrire leurs domaines des connaissances.

Une ontologie définit un vocabulaire et une sémantique permettant de décrire des concepts, leurs attributs et leurs relations (Bachimont, 2000). La difficulté, pour l'expert, est de retranscrire son domaine de connaissances. Pour ce faire, il doit se poser plusieurs questions du type : qu'est-ce que je cherche exactement ? Est-ce qu'il y a des règles à respecter ? Comment choisir les bons mots avec la bonne syntaxe pour chaque concept ou chaque attribut ? Une formulation claire et précise pour chaque élément de l'ontologie permet de filtrer la requête créée par l'expert métier et par conséquent obtenir des résultats plus pertinents (Noy and mcguinness, 2001).

4.2.2. La création des ontologies

Notre système multi agents OCEAN est composé de quatre modules (chapitre 3, Figure 2). Le premier module consiste à définir les connaissances nécessaires pour le projet par la création d'une ontologie, c'est-à-dire un ensemble de concepts et de relations permettant aux

acteurs métier de spécifier leurs domaines de connaissances. Nous présentons dans cette section les entités utilisées dans OCEAN pour la création d'une ontologie (Figure 14).

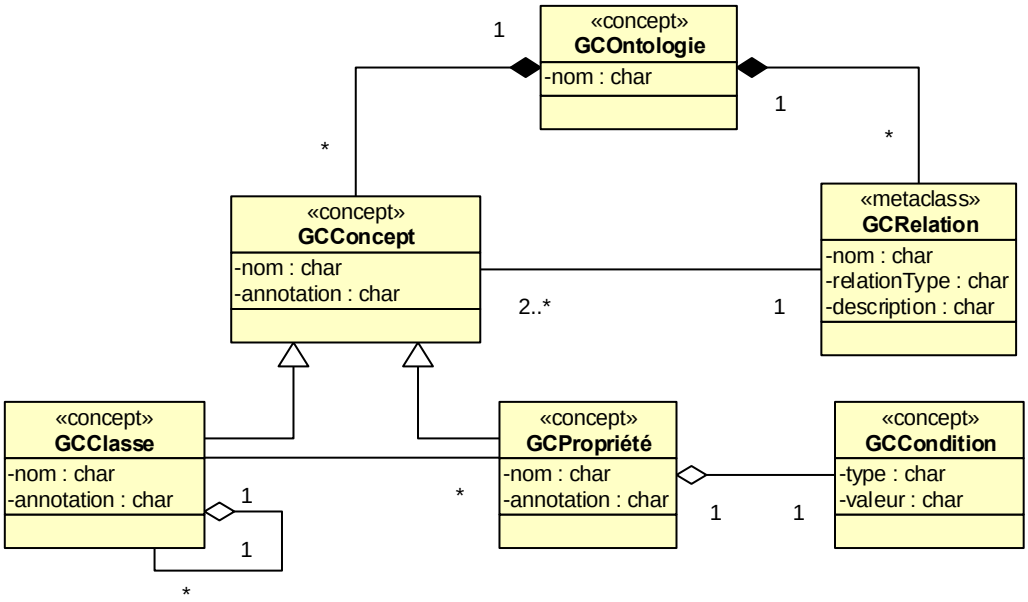


Figure 14: Le méta-modèle d'une ontologie

Nous avons présenté, dans le chapitre 3, notre ontologie du problème en utilisant un diagramme de classe. La Figure 14 détaille une partie de ce diagramme (GCOntologie, GCConcept et GCRelation). Cette figure montre les entités constituant une ontologie de domaine dans notre système OCEAN. Ce méta-modèle sera utilisé principalement par les agents pour transformer une ontologie en langage de requête (au chapitre 5). Nous rappelons que le préfixe « GC » dans la Figure 14 correspond à la gestion des connaissances. Ceci nous permet de créer une identité spécifique pour chaque concept dédié à la gestion des connaissances.

Une GCOntologie est définie par:

- Un ensemble de concepts « GCConcept » et des relations « GCRelation ».
- Les GCConcepts sont de deux types. Ils peuvent être, soit des classes « GCClasse », soit des propriétés « GCPropriété ». Dans le cas où le GCConcept est une GCClasse, celle-ci peut avoir un nom et une annotation décrite en langage naturel.

  Par exemple, on crée une GCClasse avec le nom « table » et une annotation « c'est une table basse avec un coffre de rangement au milieu ». L'annotation est alors vue comme une description du produit à concevoir en langage naturel.

- Chaque GCClasse est composée d'une ou de plusieurs sous-classes qui elles aussi sont des GCClasses. Prenant l'exemple de la table, la classe table est composée de la sous-classe « surface » et de la sous-classe « pied ».
- Une GCClasse classe peut avoir aussi des GCPropriétés. Une GCPropriété est caractérisée par un nom, et une annotation. La propriété peut décrire une simple propriété ou contenir une condition. Dans le cas d'une condition, il faut spécifier son type (condition de différence, d'égalité ou bien d'inclusion) et sa valeur.

  Par exemple, nous considérons que l'expert métier a créé la classe « pied » et les propriétés « longueur », « largeur », « matière » et « forme », qui appartiennent à cette classe. Prenant la propriété « longueur » définie par l'expert, elle a comme nom « longueur » et comme annotation « est la taille du pied ». Cette propriété est une propriété simple comme elle ne possède qu'un nom et une annotation.

  Tandis que si l'on prend la propriété « largeur » définie aussi par l'expert métier, elle a le nom « largeur », l'annotation « est la dimension perpendiculaire à la longueur », et en plus une condition de type « égalité ». L'expert a mentionné que la valeur de cette condition est « 4 ». Ceci signifie que la largeur de la table doit être égale à 4. Cette propriété est alors une propriété conditionnée.

- Les GCConcepts sont reliés entre eux par des relations « GCRelation ». Chaque GCRelation constitue la liaison entre deux concepts déjà définis dans OCEAN. ces relations sont caractérisées par un nom, un type et une description. Les relations peuvent être des relations de tout type et pas seulement des relations d'héritage. Par exemple, la classe « table » est reliée avec la classe « pied » par la relation « est_composé_de ».

  Chaque relation peut avoir une relation inverse, par exemple la classe « pied » est reliée à la classe « table » par la relation « est_élément_de ».

Ainsi, l'expert métier peut créer son ontologie en définissant les éléments présents dans la Figure 14. Les propriétés (simples ou conditionnées) définies par l'expert métier seront les éléments clés de recherche dans OCEAN. Autrement dit, pour concevoir une table, l'expert métier doit définir une ontologie de conception d'une table. Cette ontologie peut contenir des classes et des sous-classes qui correspondent aux éléments de la table, et ensuite les propriétés qui sont les paramètres de la table. L'expert métier définit alors une « surface » comme une

classe et une « forme » comme une propriété de cette classe. De plus, une contrainte supplémentaire définit que cette propriété doit être égale à « rectangulaire ». De cette façon, l'expert métier spécifie un espace de domaine qui ne concerne que des tables ayant une forme rectangulaire. Cet exemple confirme notre idée que les propriétés jouent un rôle très important dans la création d'ontologie de domaine et qu'elles permettent d'affiner les connaissances recherchées.

4.3. Notre système d'aide à la création d'ontologie

4.3.1. Motivation

Nous avons expliqué que la première étape du processus de gestion des connaissances consiste à identifier les connaissances nécessaires pour le projet par l'expert métier. Ces connaissances sont formalisées pour construire une ontologie. Cependant, la création des concepts dans notre système se fait en langage naturel. La grande problématique de l'utilisation du langage naturel est la diversité du sens que peut avoir chaque mot. Pour résoudre ce problème, les entreprises créent un vocabulaire de domaine commun pour ses employés. Mais nous sommes dans le cas d'une entreprise étendue, et par conséquent chaque entreprise a son propre vocabulaire qui n'est pas forcément diffusé ou accepté par les autres membres de l'entreprise. Il est donc difficile de déterminer quel est le vocabulaire à utiliser pour créer l'ontologie. C'est dans ce sens qu'OCEAN va pouvoir apporter une assistance aux experts métier et leur permettre de choisir un vocabulaire adapté.

4.3.2. Scénario

Pour faire face aux problèmes d'ambiguïtés sémantiques dus aux sens des concepts créés dans l'ontologie, nous proposons une approche pour aider l'expert métier à chaque étape de la création de l'ontologie. Comme le montre la (Figure 15), la création d'ontologie est soutenue par un processus de six étapes. Ce processus est assuré par trois agents. Le premier est l'agent « expert » qui interagit avec les actions des experts métiers. L'agent « GCOntologie » qui se charge de la création, le sauvegarde et la manipulation de l'ontologie. Et l'agent « RechercheSimilitude » qui s'occupe de la recherche des connaissances similaire sémantiquement à l'ontologie en cours de création. Les agents assurent alors la création de l'ontologie avec le système d'aide en suivant les étapes suivantes :

• Étape 1: avant de commencer la création d'une ontologie l'expert métier cible une application métier. Cette étape permet de préciser sur quelle application et par conséquent dans quel type de base de données, ou de format de stockage, les agents vont chercher les connaissances.

• Étape 2: créer une ontologie dans OCEAN en lui donnant un nom. Cette ontologie est stockée dans la base des ontologies. La création et la sauvegarde de l'ontologie sont garanties par l'agent «GCOntologie ». Cet agent récupère le nom de l'ontologie proposé par l'expert et crée l'ontologie correspondante dans la base des ontologies. De manière usuelle, une ontologie correspond à chaque application métier.

• Étape 3: créer un concept. L'expert métier peut créer autant de concepts qu'il le souhaite pour représenter son domaine de connaissances. Comme pour la création de l'ontologie, cette action sera assurée par l'agent « GCOntologie ». Un concept décrit une connaissance. Par exemple, « Frameset » (cadre) est un concept dans le domaine de connaissances du vélo.

• Étape 4: Lors de la création d'un concept, l'agent « rechercheSimilitude » ajoute des concepts similaires temporairement. Ces concepts sont recherchés dans la base lexicale WordNet (Stark and Riesenfeld, 1998). WordNet est une base de données lexicale en anglais. Noms, verbes, adjectifs et adverbes sont regroupés en ensemble de synonymes cognitifs (synsets), chacun exprimant un concept distinct. Les Synsets sont reliés entre eux par des relations sémantiques, conceptuelles et lexicales.

L'agent « rechercheSimilitude » interroge la base WordNet (4.1) et effectue quatre recherches basées sur :

- La synonymie (mots qui ont des significations similaires, par exemple heureux et content),
- L'hyperonymie (il se réfère à une relation hiérarchique entre les mots. Par exemple, le mobilier est un hyperonyme de chaise puisque chaque chaise est un meuble (mais pas vice-versa)),
- L'hyponymie (est à l'opposé de hyperonymie. Par exemple chapeau est un hyponyme de coiffure),
- La méronymie (se réfère à une partie du concept). Par exemple, le papier est un méronymie de livre, car le papier est une partie d'un livre. Pour cette raison, notre système utilise la base de données lexicale "WordNet".

Les résultats obtenus par l'agent (4.2) est une liste des vocabulaires sémantiquement similaire à l'ontologie ou au concept créé par l'expert métier (ListSim).

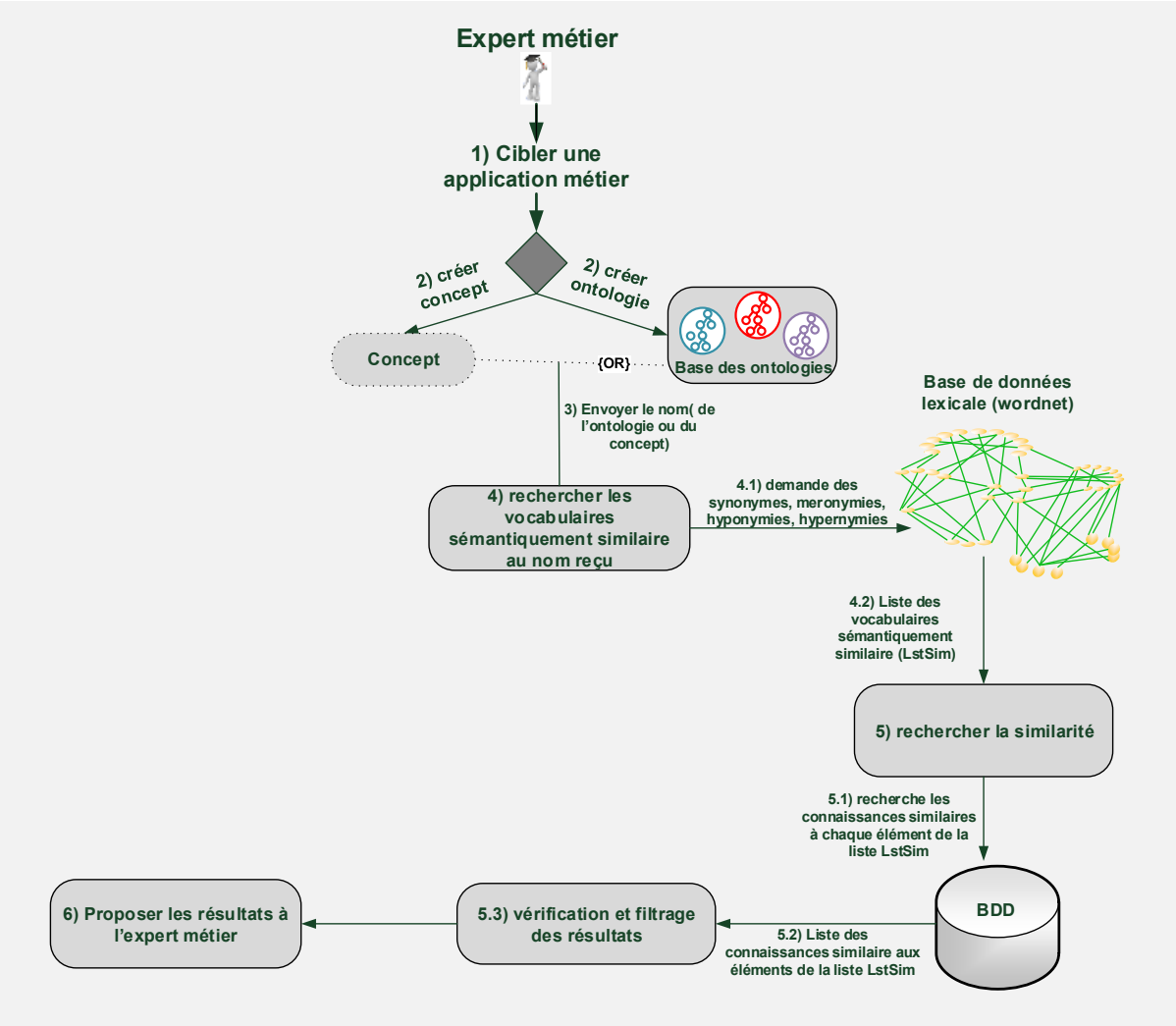


Figure 15: Système d'aide des experts métier dans la création des ontologies

• Étape 5: L'agent « rechercheSimilitude » cherche ensuite la similarité entre les vocabulaires fournis par WordNet (ListSim) et les données de la base de données de l'application métier ciblé par l'expert métier dans l'étape 1. Cette étape consiste à rechercher des connaissances similaires sémantiquement à l'ontologie (ou concept) en cours de création par l'expert métier. Si l'agent trouve des connaissances similaires il les propose à l'expert. Pour ce faire :

5.1) L'agent « rechercheSimilitude » prend chaque élément fourni par WordNet et construit une requête pour rechercher les données similaires à cet élément dans la base de données comme suit :

```
Select columns
From tables
where {each column like '%name of the element provided by WordNet% '}
```

Puis cet agent exécute la requête dans la base de données de l'application métier et obtient comme résultats des données similaires (5.2).

5.3) L'agent filtre les résultats en comparant les enfants, les parents, les attributs et les relations de ce concept existant dans la base. Par exemple si l'expert crée une ontologie de conception d'un vélo où il a défini qu'elle est composé d'un concept « roue » et que ce dernier est composé par les concepts moyeux, jantes et rayons. Ceci signifie que l'agent filtre les résultats en sélectionnant juste les vélos qui ont comme architecture de roue : moyeu, jantes et rayons.

• Étape 6 : enfin l'agent « expert » présente les résultats à l'expert métier. L'expert peut valider ou refuser la similitude entre le concept qu'il a entré et le nom du concept utilisé dans la base de données.

Pour avoir plus de détails sur le fonctionnement de ce module de création des ontologies, nous allons décrire les échanges d'information dans le cas de création de concept d'ontologie dans un diagramme de séquence. Ce diagramme permet de visualiser l'enchaînement des actions dans le temps et montrer les interactions entre les trois agents (« expert », « rechercheSimilitude », et « GCOntologie ») et les objets extérieurs du système (la base de données et la base lexicale WordNet). L'agent communique avec les autres objets par le biais de messages.

Le diagramme de séquence (Figure 16) montre les interactions entre les trois agents et les objets extérieurs du système dans le cas de création d'un concept dans la plate-forme OCEAN. Dans le cas de création d'ontologie, le diagramme de séquence est presque le même que la Figure 16 sauf que nous ne réalisons pas l'étape (5.3) dans ce cas. En plus ce processus se fait une fois pour chaque création d'ontologie.

D'après la Figure 16 nous pouvons voir que le processus se déclenche lorsque l'expert métier cible une application métier duquel l'agent doit chercher les connaissances. Puis l'expert crée un nouveau concept dans son ontologie. L'agent « expert » écoute les actions de l'expert métier. Lorsqu'il détecte une demande de création de concept, il appelle l'agent « GCOntologie » pour ajouter le concept à l'ontologie et la sauvegarder. Lorsque cet agent

accomplit sa mission, il déclenche l'agent « rechercheSimilitude » pour rechercher les vocabulaires similaires (synonymes, hypernymes ...) au nom du concept créée à partir des recherches faites en interrogeant la base lexicale WordNet. Les vocabulaires retournés sont utilisées pour rechercher les connaissances qui leur ressemblent à partir de la base de données de l'application métier ciblé.

Si l'agent « rechercheSimilitude » trouve des résultats dans la base de données, il réalise un filtre (étape 5.3) en comparant les parents, les enfants, etc. Ce filtrage sera présent dans la requête formulé par l'agent afin d'avoir des résultats plus pertinents (Figure 16). Les résultats retournés à l'agent « expert » seront également proposé à l'expert métier dans l'interface d'OCEAN. Ces propositions correspondent aux connaissances existantes dans la base de données qui sont similaires au concept créé. L'expert métier peut valider ou refuser ces propositions.

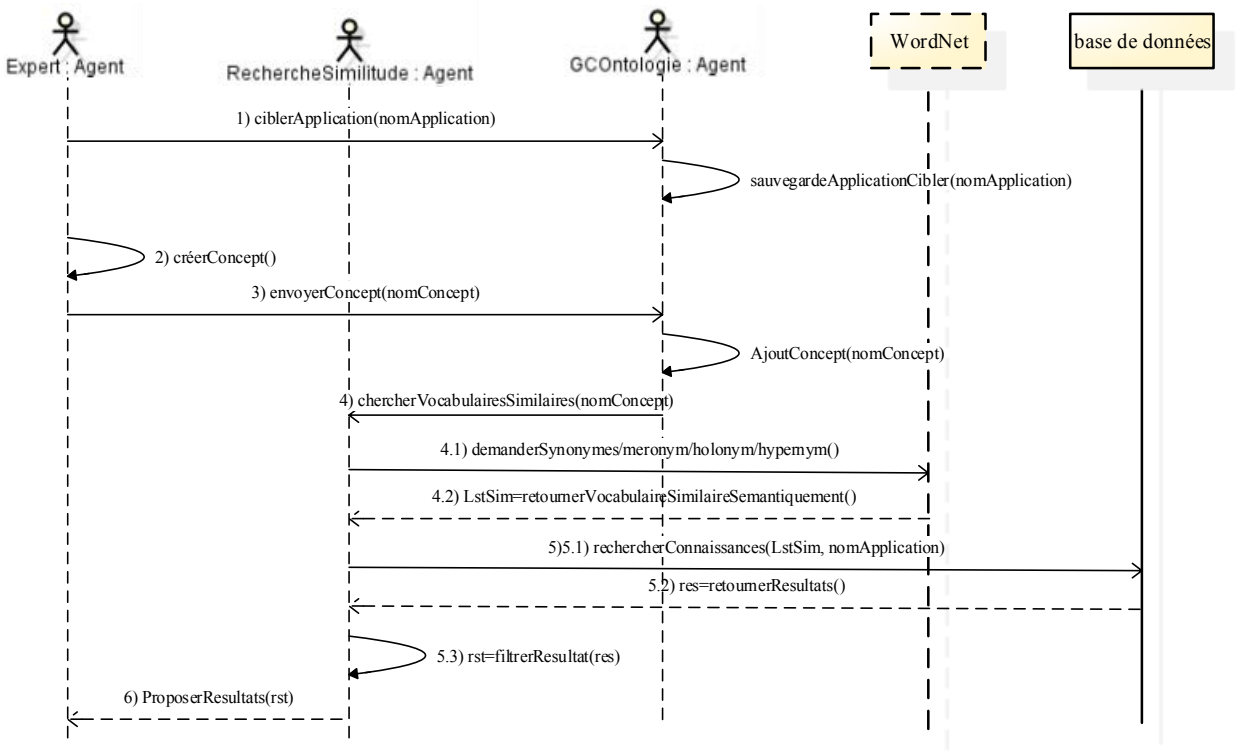


Figure 16: Processus de recherche des projets similaires (cas création concept)

Ce processus d'aide à la création des concepts se déclenche à chaque création d'un nouveau concept dans la même ontologie.

4.3.3. Exemple : cas de construction d'une ontologie d'un moteur mécanique

Nous présentons ici un exemple simple d'ontologie construite par un expert métier dans la plate-forme OCEAN. La base de connaissances décrit les éléments d'un moteur mécanique. (Figure 17). Les concepts en rectangle vert sont des classes, tandis que les propriétés sont représentées par des ellipses bleues. L'expert métier a la possibilité de créer une ou plusieurs propriétés pour chaque classe comme la montre la figure ci-dessous. Par exemple la classe « piston » a neufs propriétés (outside diameter hole, segment number ..). La relation « classe - propriété » est représentée par la flèche titrée « has ». Les classes sont, de leur côté, reliées par des relations définies par l'expert métier. Par exemple la classe « engine » est reliée à la classe « power train » par la relation « is_composed_by ».

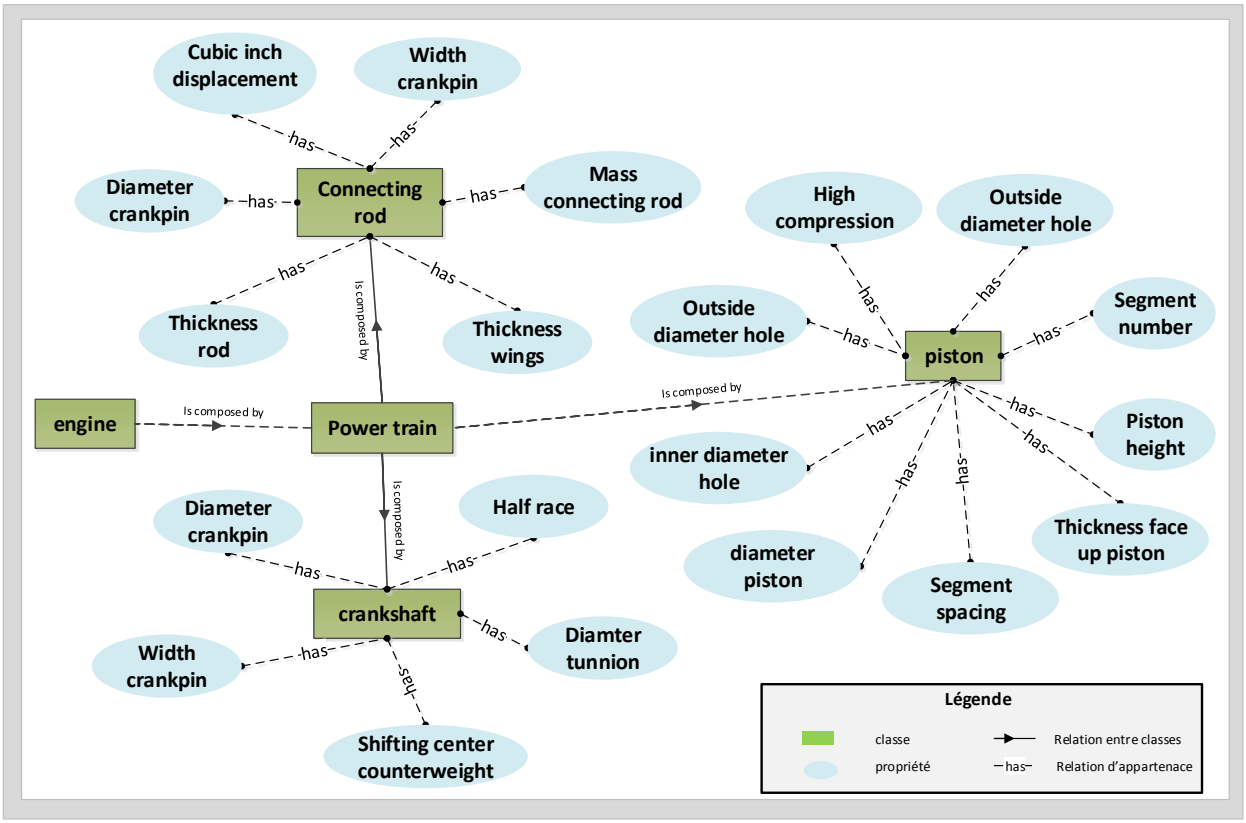


Figure 17: Une ontologie d'un moteur mécanique

Simultanément, les agents se déclenchent automatiquement pour enrichir l'ontologie. Puis ils recherchent des synonymes pour chaque concept (classe) de l'ontologie, comme expliqué dans la section précédente. Les concepts représentés par un hexagone orange sont des synonymes proposés par les agents à partir de WordNet (Figure 18). Les hyponymies sont représentées par des pentagones jaunes, les hypernymies par des étoiles roses, les meronimies par des carrés gris. Nous trouvons aussi les concepts classes, propriétés et les relations entre

eux, comme expliqués dans la Figure 17. Il faut noter que les relations entre les classes sont définies par l'expert métier dans OCEAN, par contre les relations classes-propriétés sont définies toujours par les agents avec la relation d'appartenance « has ».

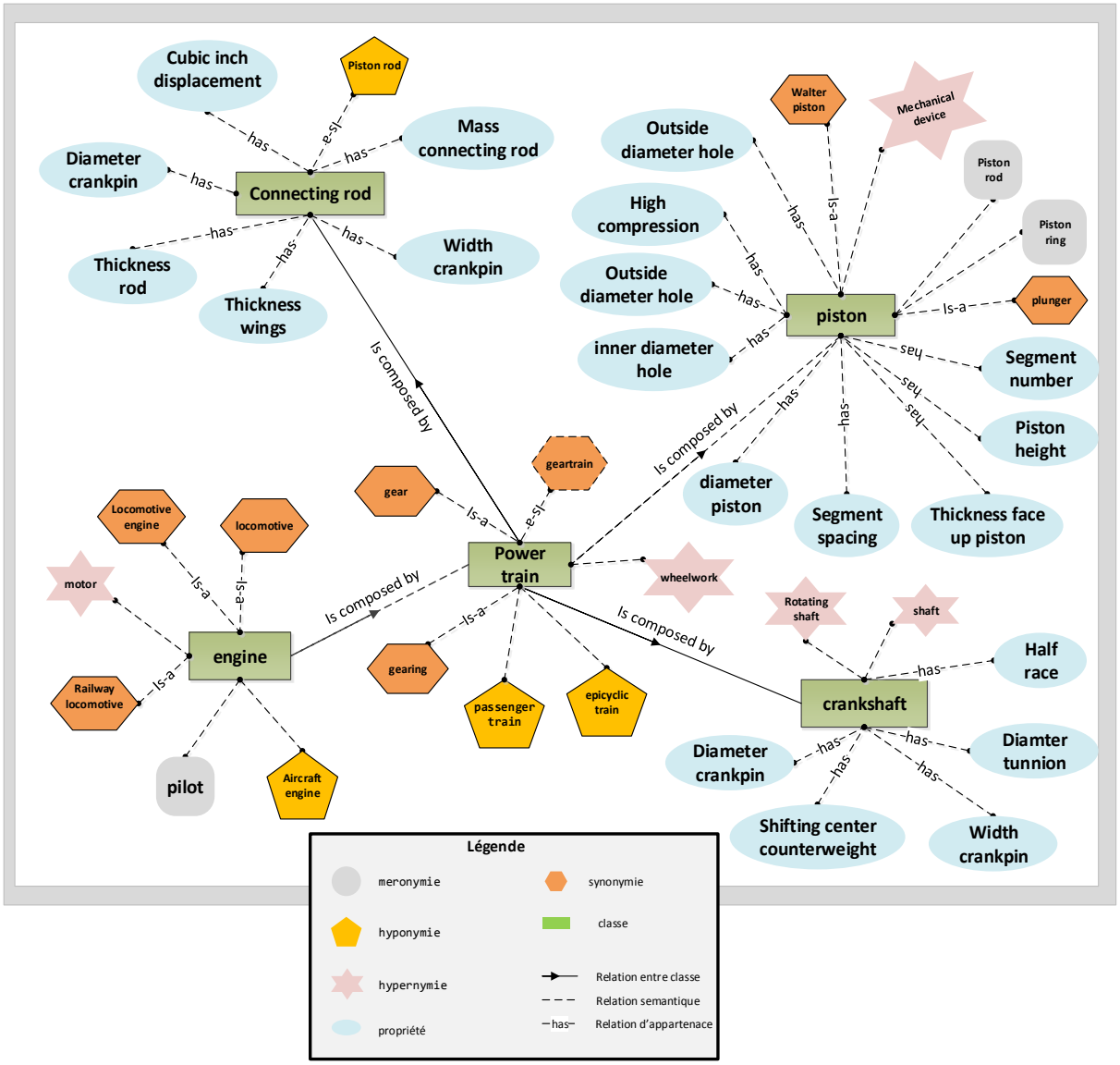


Figure 18: Proposition de nouveaux concepts par les agents

Chaque concept de l'ontologie enrichie (Figure 18) sera utilisé pour rechercher des connaissances similaires dans la base de données existante. Le résultat de cette recherche sera enregistré et envoyé à l'expert en le présentant dans la plateforme OCEAN.

La (Figure 19) montre que l'expert métier a validé les connaissances similaires proposées par l'agent pour deux concepts. Le concept « moteur » défini par l'expert, par exemple, a été remplacé après sa validation par le concept «moteur de la voiture en usine x». Ce dernier était un des concepts proposé à l'expert parce qu'il est similaire au concept moteur

dans la base de données. De même, le concept "power train" créé par l'expert métier a été remplacé par "gear train". Ces changements ne sont pas le fait d'OCEAN seul. En effet, c'est l'expert métier qui a toujours la charge de ces créations, mais il est assisté par le système OCEAN. La (Figure 19) présente l'ontologie finale qui est une « ontologie adaptée », car elle représente l'ontologie originale adaptée avec les résultats de l'assistance qui sont guidés par les bases de données des projets et les agents de l'organisation « Définition ».

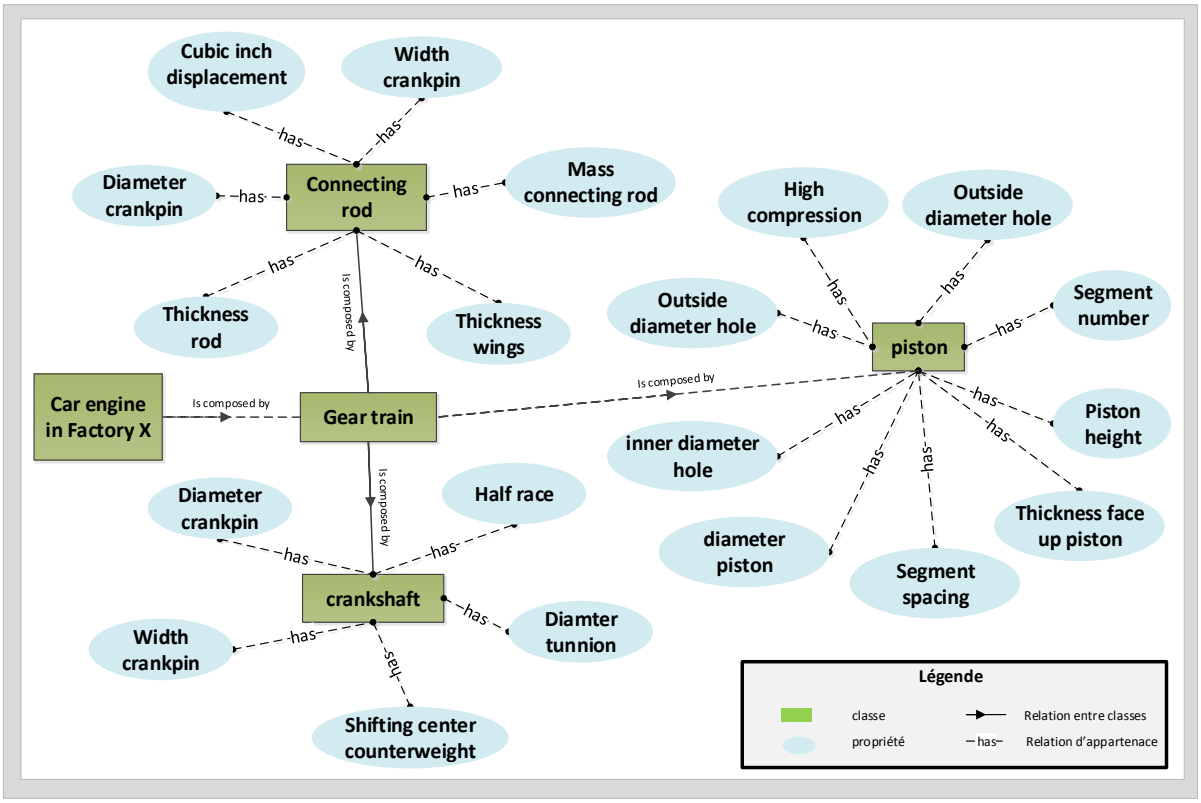


Figure 19: Exemple ontologie adapté à la base de données

4.4.Le sauvegarde des ontologies

Les ontologies créées par les experts métier sont stockées dans des fichiers de format OWL dans une base des connaissances conceptuelles. Cette sauvegarde sera réalisée par l'agent « GCOntologie ». L'agent génère les fichiers en respectant les recommandations du W3C.

Le fichier OWL comporte des classes, des DatatypeProperty, et des ObjectProperty (McGuinness and Van Harmelen, 2004).

Dans la plateforme OCEAN, la création des concepts par les experts correspond à des classes dans le fichier OWL. La création des attributs correspond à des DatatypeProperty et la création des relations entre les concepts (classes) correspond à des ObjectProperty.

La Figure 20 présente l'exemple de la classe « Frame » construite par l'agent. Cette classe est une sous-classe de la classe « speedmax AL 8.0 ».

```
<owl:Class rdf:about="#frame">
<rdfs:comment>this is the frame of the bicycle speedmax AL 8.0</rdfs:comment>
<rdfs:subClassOf rdf:resource="#speedmax AL 8.0"/>
</owl:Class>
```

Figure 20: Classe générée en OWL

Les fichiers OWL comprennent aussi des datatypeProperty et des restrictions sur ces propriétés. La Figure 21 présente une création de la propriété « Seat tube length » (longueur du tube de la selle) avec un domain « frame » et une chaine de caractères comme range (McGuinness and Van Harmelen, 2004) « Range » permet de spécifier le type de la propriété, tandis que « domain » permet ainsi de spécifier quelle est la classe à laquelle nous pouvons affecter telle ou telle propriété.

Dans notre exemple, la propriété contient aussi une restriction qui est précisée par la syntaxe « hasValue ». Cette restriction sur la propriété exige que la longueur de la selle doive être égale à 12. Alors, seuls les vélos qui ont comme longueur de tube de selle =12 seront retournés à l'utilisateur comme résultat.

```
<owl:DatatypeProperty rdf:ID="Seat_tube_length">
<rdfs:domain rdf:resource="#frame"/>
<rdfs:range rdf:resource="&xsd;string"/>
</owl:DatatypeProperty>

<owl:Class rdf:ID="frame">
<rdfs:comment> the seat tube length of frame must be egal to 12 </rdfs:comment>
<rdfs:subClassOf>
<owl:Restriction>
<owl:onProperty rdf:resource="#Seat_tube_length"/>
<owl:hasValue rdf:resource="12"/></owl:Restriction>
</rdfs:subClassOf>
</owl:Class>
```

Figure 21: Propriété avec une restriction d'égalité générée en OWL

La deuxième figure (Figure 22) représente aussi une datatypeProperty avec une restriction. Mais cette condition est désignée par un « datarange » avec une liste d'éléments.

OWL a fourni un nouveau concept pour définir une plage de valeurs de données, à savoir un type de données énumérées. Ce format du type de données utilise le concept d'OWL : oneOf. Dans le cas d'un type de données énumérées, le sujet de owl: oneOf est un nœud vide de la classe owl: DataRange et l'objet est une liste de littéraux.

Dans cette figure, nous pouvons constater que la propriété « reach » (portée du cadre) du classe « frame » doit avoir une valeur de la liste [430, 455, 33, 11].

```
<owl:DatatypeProperty rdf:ID="reach">
<rdfs:domain rdf:resource="#frame"/>
<rdfs:range>
<owl:DataRange>
<owl:oneOf>
<rdf:List>
<rdf:first rdf:datatype="&xsd;string">430</rdf:first>
<rdf:rest>
<rdf:List>
<rdf:first rdf:datatype="&xsd;string">455</rdf:first>
<rdf:rest>
<rdf:List>
<rdf:first rdf:datatype="&xsd;string">33</rdf:first>
<rdf:rest>
<rdf:List>
<rdf:first rdf:datatype="&xsd;string">11</rdf:first>
<rdf:rest rdf:resource="&rdf;nil"/>
```

Figure 22: Propriété avec une restriction DataRange généré en OWL

Nous avons montré deux exemples de propriétés avec des restrictions, mais nous pouvons avoir dans nos fichiers des propriétés définies sans restriction ou des restrictions autres que l'égalité et l'énumération comme la différence (differentFrom une valeur).

Jusque-là, nous avons présenté comment sont créées les classes. Toutefois, ces classes peuvent être reliées par des relations autres que l'héritage. Ces relations seront également définies par les experts métier. La (Figure 23) montre un exemple d'une telle relation qui signifie que le vélo a un cadre. Chaque relation peut avoir aussi une relation inverse. Ceci se traduit par la relation « belongsTo » qui explique que le cadre correspond à un vélo.

```
<owl:ObjectProperty rdf:ID="hasFrameset">
<rdfs:comment> this bike has frameset </rdfs:comment>
<rdfs:domain rdf:resource="#Bike"/>
<rdfs:range rdf:resource="#Frameset"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:ID="belongsTo">
  <owl:inverseOf rdf:resource="#hasFrameset" />
</owl:ObjectProperty>
```

Figure 23: Relation généré en OWL

4.5.Conclusion

Dans ce chapitre, nous avons introduit le premier module de notre système de gestion des connaissances multi-sources. Celui-ci est composé de deux étapes principales qui sont la définition, l'identification des connaissances par l'expert métier, et la création d'une ontologie.

La définition des connaissances est une tâche primordiale avant la création de l'ontologie. L'expert métier identifie les connaissances dont il a besoin pour décrire son problème avant de les formaliser dans notre SGC.

Cette définition des connaissances se fait en langage naturel. Ceci entraine des problèmes dus à la synonymie des mots définis par l'expert métier. Pour cette raison, nous avons introduit, dans notre système, une approche pour aider les experts métiers à bien définir leurs connaissances afin d'obtenir un bon résultat à la fin. Cette approche a été présentée dans la section 4.3 et nous avons décrit le scenario de son fonctionnement. Les agents assurent le bon fonctionnement du système.

Le système que nous avons présenté dans ce chapitre permet aux experts métiers de définir les connaissances à capitaliser afin de les réutiliser pour fournir une aide à la décision pour les acteurs métiers.

Nous allons expliquer, dans le chapitre suivant, comment nous allons utiliser les ontologies de domaines créées par les experts métiers pour extraire des données des applications métiers. Cette extraction servira à créer ou à enrichir la mémoire organisationnelle du système dans le but de réutiliser les connaissances par la suite dans un système de wiki sémantique.

CHAPITRE 5

L'EXTRACTION ET L'ANNOTATION DES CONNAISSANCES

Ce chapitre présente les méthodologies utilisées pour extraire les connaissances issues des différentes applications métiers distribuées sur le réseau de l'entreprise étendue. Ces connaissances sont annotées et sauvegardées dans une mémoire organisationnelle. Les fonctionnalités d'extraction, d'annotation et de sauvegarde sont réalisées dans le système OCEAN en utilisant le paradigme agent.

Table de matières

5.1. Introduction

Le processus de conception de produits mécaniques nécessite fréquemment l'utilisation de plusieurs outils de calcul, outils de CAO, outils de gestion de production, PLM (Product Lifecycle Management), PDM (Product Data Management), etc. Ces outils portent sur des aspects spécifiques de la conception des produits. Bien souvent, ces outils sont répartis sur différents sites géographiques à travers l'ensemble du réseau de l'entreprise étendue.

Le but de ce chapitre est de proposer une approche, basée sur les agents, permettant l'extraction de connaissances issues des outils logiciels de l'entreprise étendue. Cette approche repose sur la création de base de connaissances par les experts métier. Ces bases de connaissances sont construites à l'aide d'ontologies. Ces dernières spécifient les concepts et leurs relations qui conceptualisent la connaissance spécifique d'une application métier. Ces ontologies sont ensuite utilisées pour créer des requêtes afin de rechercher des données stockées dans différentes sources d'information de l'entreprise (base de données, fichiers textes, fichiers xml, etc.). Les données sont ensuite annotées afin de devenir des informations. Une information étant une donnée associée à un contexte particulier. Ces informations sont ensuite proposées aux acteurs métier afin que ces derniers les valident et les évaluent. Les informations évaluées deviennent alors des connaissances.

L'extraction des données opérées par OCEAN dans les différentes applications métiers sont réalisées par le biais des services web (Alonso et al., 2010). De plus, nous utilisons Les techniques de l'Ingénierie Dirigée par les Modèles (IDM) (Jézéquel et al., 2012) (Muller, 2005) pour transformer des ontologies en langage de requête. Cette approche offre une modularité et une réutilisabilité des modèles (Baez, 2005).

Ce chapitre est organisé comme suit : Après une présentation générale de la partie du Système Multi-Agents dédiée à l'extraction des connaissances de plusieurs applications métier, nous détaillons le mécanisme pour différents types d'application métier. La section 5.3.2 détaille le cas d'une application métier basée sur SQL server comme format de stockage des données. Nous introduisons notre méthodologie de l'extraction de ce type de base et puis nous présentons nos règles de transformation de l'ontologie en langage de requête SQL qui est conforme à cette base de données.

Dans la section 5.3.3, nous détaillons la transformation d'une ontologie en une requête exécutable sur des fichiers Excel. Les fichiers Excel peuvent être des sources de données mais ils ne sont pas bien structurés. Alors pour extraire des données d'un fichier Excel de plusieurs feuilles, il faut créer une macro avec un code complexe.

Pour cette raison nous avons décidé d'utiliser une méthode qui est très utilisée aujourd'hui pour transformer les fichiers comme Excel et XML en RDF.

Ceci nous permet de constituer une base des fichiers structurés, qui contiennent de la sémantique et sur laquelle nous pouvons appliquer des requêtes SPARQL et extraire les données facilement.

Pour cette raison nous détaillons dans la section 5.3.3 notre méthodologie pour transformer les fichiers Excel en fichiers RDF et puis transformer l'ontologie en requête SPARQL (Prud'Hommeaux and Seaborne, 2008) afin de l'exécuter sur ces fichiers RDF et extraire des données.

Finalement nous présentons notre approche d'annotation des données résultantes de l'extraction des applications métiers. Cette annotation est basée sur l'ontologie «OntoDesign » (Monticolo et al., 2007a).

5.2.Des agents dédiés à l'extraction des connaissances

5.2.1. Méthodologie globale

Notre second objectif est de concevoir et développer le deuxième module de notre système OCEAN, qui permet d'extraire des connaissances de plusieurs applications métiers. L'approche globale de cette extraction peut être résumée par la figure ci-dessous (Figure 24).

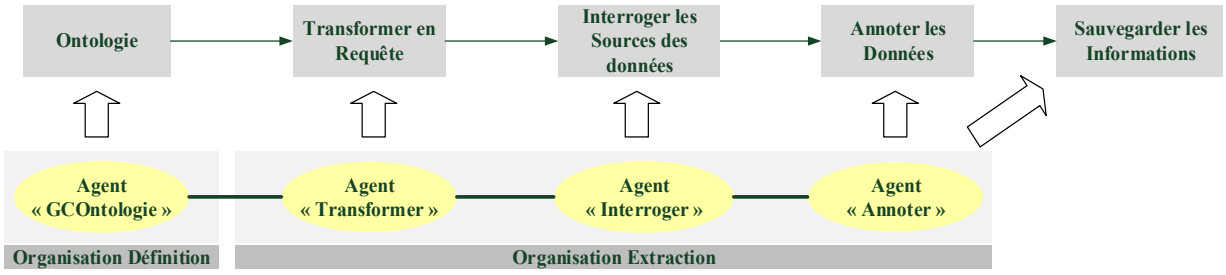


Figure 24: architecture globale du processus d'extraction

Nous avons identifié pour l'organisation « Extraction » trois agents qui sont : agents « transformer », agent « interroger » et agent « annoter ». Tout d'abord, Nous rappelons que l'agent GCOntologie permet de manipuler et représenter une ontologie métier. Cette ontologie correspond à l'ontologie créée par l'expert métier dans la première étape du cycle de vie de notre système de gestion de connaissance OCEAN (voire chapitre 3, section 3.1). L'agent « GCOntologie » de l'organisation « Définition » communique avec l'agent « transformer »

de l'organisation « Extraction ». L'agent GCOntologie lit alors l'ontologie et récupère ses éléments afin de les envoyer à l'agent « Transformer ». Ce dernier récupère les éléments de l'ontologie et les transforme en une série de requêtes exprimées dans un langage permettant de communiquer avec les applications métiers et d'extraire les connaissances. Ces requêtes sont exécutées par l'agent « Interroger ». Cet agent reçoit les données résultat de l'exécution des requêtes. Afin de pouvoir réutiliser ces données, l'agent « annoter » définit le contexte dans lequel la donnée a été capitalisée. La création de ce contexte est à la charge de cet agent. Les données annotées deviennent alors des informations. Les annotations sont construites à partir des concepts définis dans l'ontologie OntoDesign. Les informations sont sauvegardées par le même agent dans des fichiers RDF pour construire une mémoire organisationnelle.

5.2.2. Approche détaillée pour l'extraction des connaissances

Nous avons expliqué notre démarche globale pour extraire les données des applications métiers. Ces dernières peuvent provenir de tous types d'applications, comme des outils CAO, des outils PLM, des logiciels de gestion, etc. Nous avons expérimenté dans notre travail deux types de sources d'informations différentes: Des bases de données SQL server et des fichiers EXCEL.

La Figure 25 explique notre approche détaillée d'extraction pour les deux types de sources d'information:

Cas 1: Extraction à partir d'une base de données : soit l'ontologie "O1" qui décrit les concepts de l'application PLM (étape 1). Cette application a comme format de stockage des données SQL server. Après la manipulation d'O1 par l'agent GCOntologie, ce dernier l'envoie à l'agent « transformer » pour le transformer en une requête SQL (étape 2). Cette requête sera traitée par un service web (étape 3) qui se connecte à la base de données de l'application PLM (étape 4) et l'exécute. Cette étape est réalisée par l'agent « interroger ». Le résultat de l'exécution est un ensemble de données. Ces données (étape 5) sont ensuite annotées par l'agent « annoter » pour devenir des informations, et stockées dans des fichiers RDF venant enrichir une mémoire organisationnelle.

Cas 2: Extraction à partir d'un fichier Excel : les étapes sont les mêmes que pour le Cas 1, mais cette fois l'ontologie OC sera transformée en langage de requête SPARQL exécutable sur les fichiers RDF (résultat de la transformation Excel -> RDF) (étape 2) et la connexion se

fera avec une application métier de type CAO (étape 4). Nous avons testé le cas 2 sur l'outil de conception CATIA (Carman and Tigwell, 1998).

La réalisation de l'extraction des connaissances dans notre système se fait à l'aide de techniques de l'IDM et des services web.

L'ingénierie dirigée par les modèles nous permet de réaliser des transformations entre deux modèles différents (Abderraouf, 2012) (Roser and Bauer, 2006); dans notre cas d'une ontologie vers un langage de requête.

D'autre part, nous utilisons également des services web (SW) qui ont pour avantage d'assurer l'interopérabilité entre les applications métiers et notre système OCEAN en vue de rendre l'accès aux informations dynamique (Authosserre-Cavarero et al., 2012). Ces SW nous permettent d'automatiser l'accès aux sources d'informations ce qui est nécessaire dans le cas d'une entreprise utilisant de multiples sources d'information. Nous avons donc développé un service web par application métier.

Nous avons utilisés le service web pour se connecter aux bases de données des applications métiers qui sont distribuées sur le réseau de l'entreprise (Figure 25). L'expert métier cible une base de données lors de création de son ontologie comme expliqué dans le chapitre 4. Alors l'agent « interroger » utilise le nom de la base de données ciblé afin de rechercher automatiquement le protocole de connexion à cette base et l'envoyer au service web correspondant. Le SW se connecte donc à la base à distance dans le but d'exécuter la requête (résultat de la transformation de l'ontologie en une requête) et extraire des données.

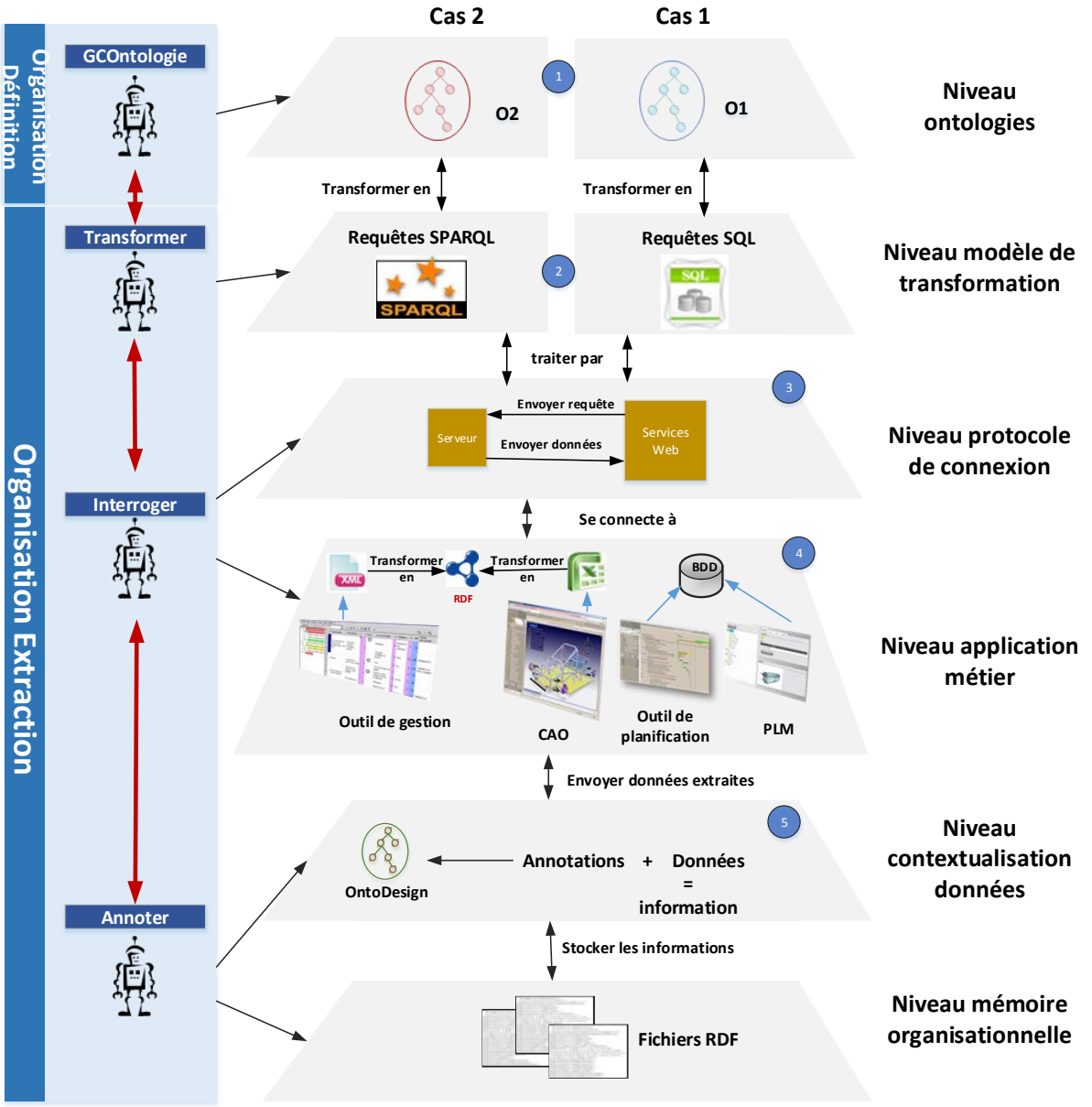


Figure 25: l'approche d'extraction des connaissances pour deux types de données

La section suivante décrit le deux cas d'extraction de connaissances ; celle d'une base SQL server et d'un fichier Excel. Pour réaliser cette extraction nous présentons notre méthodologie pour résoudre la problématique de la transformation d'une ontologie en langage de requête. Nous décrivons ainsi les règles que nous avons appliquées pour ces transformations et des exemples illustrant notre méthodologie.

5.3.L'extraction des connaissances

5.3.1. La méthodologie globale d'extraction

Le module d'extraction de connaissances multi-sources utilise les ontologies définies par les experts métiers et précise les connaissances relatives à un produit mécanique.

Afin d'obtenir des informations à partir des différents outils métiers, les ontologies (source) créées par les experts métiers sont transformées en requêtes (cible). Cette cible correspond à une requête SQL dans le cas 1 et à une requête SPARQL dans le cas 2 (Figure 26).

Dans IDM, le principe de base est que tout est un modèle (Kleppe et al., 2003). La Figure 26 montre l'idée de base de notre approche. Le premier niveau correspond à une instance de modèle.

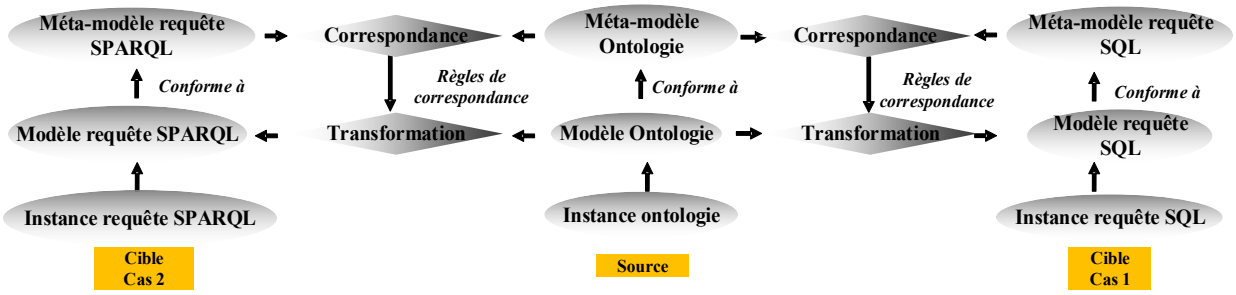


Figure 26: approche de transformation entre modèle source et modèle destination

Le deuxième niveau est la structure d'un modèle. Le niveau trois correspond au méta-modèle.

Pour illustrer ces trois niveaux de modélisation on considère l'exemple d'une ontologie très simplifiée relative aux vélos (Figure 27):

L'expert métier définit son ontologie de vélo (instance) qui comprend comme concepts (un VTT Rockrider 5.0, et un cadre). Dans cette ontologie il a défini le type de cadre (X-control 310L) et la matière de composition (acier). Cette ontologie de vélo correspond au premier niveau qui est représenté par « Instance ontologie » dans la Figure 26.

Nous passons ensuite au deuxième niveau de l'approche. Dans ce niveau nous avons le modèle de l'ontologie vélo. Ce dernier correspond à la structure de l'ontologie vélo. Dans ce modèle nous avons le concept « vélo » avec la propriété « nom ». Tout vélo a un cadre. Le cadre a un type et une matière de composition.

Pour transformer l'ontologie de vélo défini par l'expert métier en langage de requête, nous devons passer à un niveau plus générique qui est le niveau trois de l'approche (le méta-modèle). Ce niveau nous permet de faire la liaison avec d'autres méta-modèles pour trouver des ponts de passage de l'un à l'autre. Le méta-modèle de l'ontologie de vélo est le méta-modèle de l'ontologie en générale quel que soit le contenu. Ce méta-modèle est composé de classes qui peuvent avoir des sous classes. Chaque classe peut avoir aussi des propriétés.

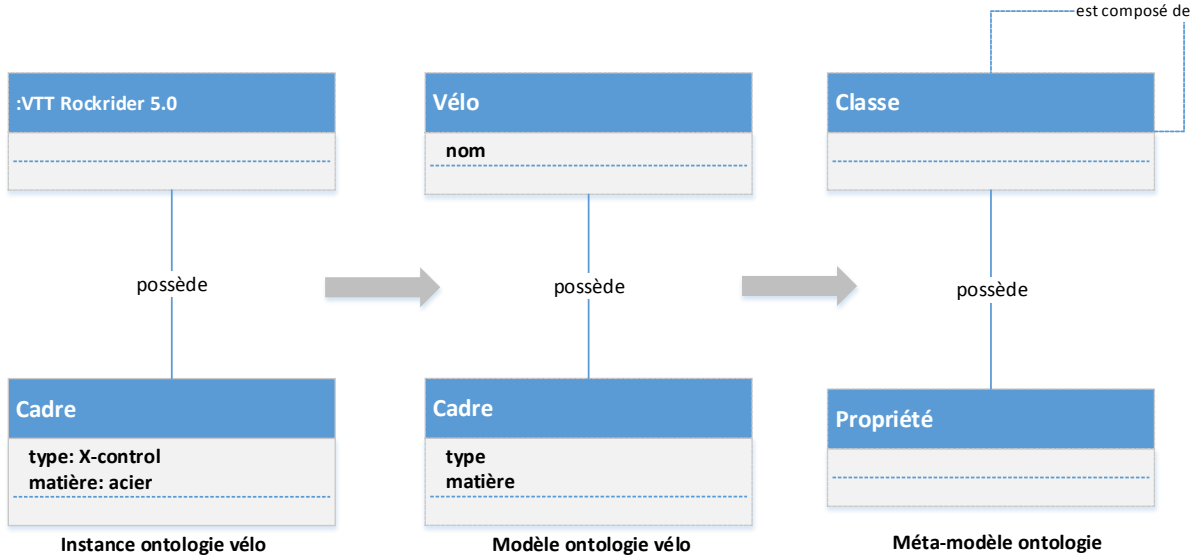


Figure 27: exemple de l'approche IDM sur une ontologie de vélo

Afin de définir la transformation entre le modèle source (ontologie) et le modèle cible (requête de type SQL ou SPARQL), nous devons d'abord définir le méta-modèle de l'ontologie (chapitre 4, figure 1) et le méta-modèle de requête de type SQL (Figure 28) ou SPARQL (Figure 32). Une fois ces méta-modèles définis, nous pouvons définir une transformation qui prend en entrée une instance d'ontologie et qui produit en sortie une instance de requête. Cette transformation est basée sur un ensemble de règles appelées règles de correspondances qui spécifient comment lier les éléments des différents méta-modèles.

Nous détaillons la transformation de l'ontologie en requête SQL dans la section suivante et en requête SPARQL dans la section 5.3.3.

5.3.2. L'extraction des connaissances d'une base SQL Server

La transformation entre les modèles a été le sujet de recherche dans plusieurs domaines tels que la biologie (Dhombres et al., 2012)(Leclercq et al., 2012), les télécoms (Chiprianov, 2012)(Nascimento et al., 2012)(Jan et al., 2012), la pédagogie (Caron, 2007), etc.. Dans le domaine de la gestion des connaissances, cette méthodologie est utilisée pour construire une

ontologie à partir d'une base de données relationnelle (Santoso et al., 2011)(Chen et al., 2012)(Trinkunas and Vasilecas, 2007), ou l'inverse (Souza et al., 2012)(Fabkam et al., 2009)(Astrova et al., 2007). Nous nous sommes inspirés de ces travaux pour réaliser la transformation d'une ontologie vers une requête SQL en créant nos propres règles. Mais avant de décrire nos règles nous présentons le méta-modèle de l'ontologie et celui de la requête SQL afin de définir le pont de passage de l'un à l'autre. Ce pont correspond aux règles de correspondances ou de transformation.

Nous avons présenté le méta-modèle simplifié d'une ontologie dans le chapitre 4 (section 4.2.2). Ce dernier est composé de plusieurs concepts qui sont liés par des relations. Les concepts sont définis par un groupe de classes, de sous-classes et des propriétés. Chaque classe peut avoir 0 ou plusieurs propriétés qui peuvent contenir des conditions.

De même, une base de données relationnelle est considérée comme la mise en œuvre d'un méta-modèle d'une requête SQL (Figure 28). Ce modèle inclut des constructions pour spécifier le "select", "from", "where", "group by", "order by", "tables", "colonnes", et "conditions".

Le méta-modèle d'une requête SQL est défini en utilisant la syntaxe de requête SQL suivante (Lans, 2006):

Select [distinct] {liste des colonnes}
From {tables}
[**Where** conditions]
[**Group by** {liste des colonnes} [**having** condition]]
[**Order by** {liste des colonnes} [**asc** | **desc**]]

La syntaxe de la condition est déterminée par trois éléments:

- Le côté gauche de la condition
- La partie droite de la condition
- L'opérateur qui relie les deux parties

L'opérateur peut être: =, =,>,> =, <, <=, between, and, in (liste), like, is nul ...

Le côté droit de la condition peut correspondre à: une colonne, constante, liste, ou une sous-requête.

Après avoir définis les méta-modèles d'ontologie et de requête SQL, nous définissons les règles de transformation de l'ontologie en requête SQL.

Cette transformation est assurée par l'agent « transformer ». Cet agent communique avec l'agent « GCOntologie » pour échanger le contenu de l'ontologie (les classes, les propriétés et les relations). L'agent « transformer » récupère le contenu de l'ontologie et

assure sa mission de transformer une ontologie en requête SQL, en appliquant les règles suivantes :

R1 le nom d'une ontologie est le nom d'un projet

R2 une classe est une condition sur le nom d'un produit ou celui d'un produit élément

R3 si une classe a une ou plusieurs DatatypeProperty alors nous commençons à construire les conditions

R4 Construisez les conditions :

L'agentExtract analyse le "range" de chaque DatatypeProperty

• si l'attribut est "positiveInteger" alors il y a la condition **check> 0**

• si l'attribut est "DataRange" qui comporte une liste alors la condition est : vérifier l'attribut ***in [value1, value2 ...]***

• s'il existe une restriction sur une DatatypeProperty et que ce n'est pas une restriction de cardinalité comme la restriction « HasValue » alors cette restriction sera une condition dans SQL avec la syntaxe : **attribut = valeur**

• « Inverse functional property » est la contrainte "Disctinct"

• « Required property » signifie vérifier que l'attribut n'est pas nul (non 0 et non ="")

R5 sauvegardez les conditions dans la liste X.

R6 il faut mentionner que nous avons une table qui s'appelle "Matching table" qui stocke toutes les tables, colonnes, base de données et les variables correspondantes. Par exemple, la variable '$ParametersName' correspond à la colonne 'nom_variable' dans la table 'variables' dans la base de données 'ACSP'. La même variable '$ParametersName' peut correspondre aussi à la colonne 'nom_parametre' dans la table « parametres » dans la base de données 'KrossRoad'.

Comme nous avons la liste totale des tables alors nous pouvons rechercher la liste des tables utilisées dans l'ontologie et les sauvegardé dans la liste Y

R7 recherchez les clés primaires et secondaires des tables de la liste Y pour avoir les relations entre eux et les sauvegardés dans une liste des relations.

R8 nous recherchons toujours dans la requête les variables : $ParametersName, $ParametersValue, $ProjectName, $ProjectDesciption, etc... Alors nous cherchons dans la

table "matching" l'équivalence de ces variables dans la base de données et nous les sauvegardons dans la liste Z.

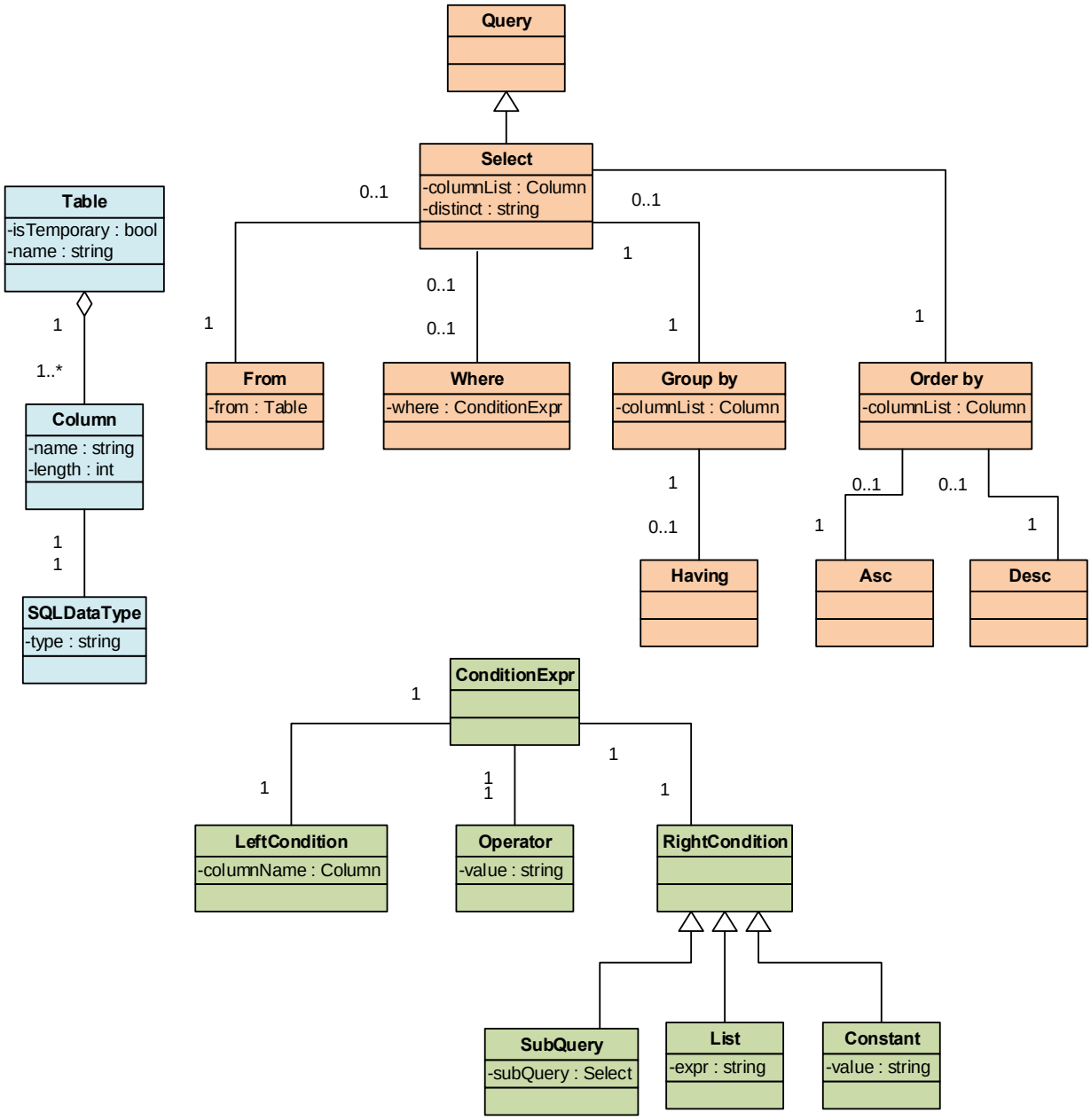


Figure 28: un méta modèle simplifié d'une requête SQL

R9 construisons la requête :

• **Select** toutes les colonnes présentes dans la liste Z précédé par la première lettre de sa table, ex.: p.nom, et séparé par des virgules

• **From** toutes les tables sauvegardées dans la liste Y suivi par la première lettre de la table comme un renommage de la table

• **Where** toutes les relations entre les tables sauvegardées dans la liste des relations et toutes les conditions dans la liste X. les conditions et les relations sont séparées par des « and ».

R10 nous répétons ces règles pour chaque classe dans l'ontologie qui a des datatypeproperties et nous ajoutons le mot « UNION » à la fin de chaque requête pour avoir un seul résultat à la fin.

Pour illustrer les règles citées ci-dessus, nous présentons un exemple qui repose sur une ontologie créée par un expert métier et qui concerne la conception de fauteuils de bureau de modèle « Chief Point 10 ». Figure 29 représente un extrait de cette ontologie.

L'expert métier peut créer son ontologie en spécifiant des concepts généraux ou même des instances pour chaque concept de l'ontologie. Les instances sont dans ce cas considérées comme des conditions. Prenant l'exemple de la Figure 29, l'expert métier définit dans son ontologie : la marque, la gamme des fauteuils de bureau, le mécanisme du fauteuil, le garnissage, la densité, la hauteur et la largeur de l'assise. Toutes ces informations doivent correspondre au modèle « Chief Point 10». De plus ce fauteuil doit pouvoir être modulable. Cette condition est définie par l'instance « fonction modulable ». Nous pouvons voir aussi que le fauteuil a une structure. L'expert précise pour ce concept que ces fauteuils sont en cuir uniquement.

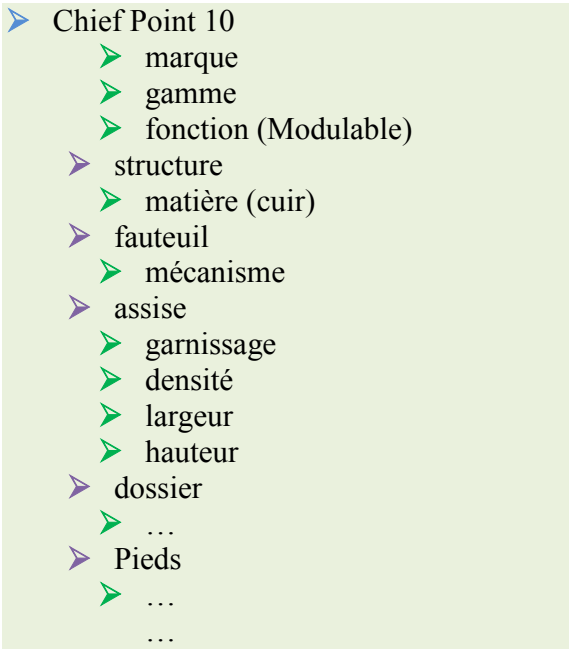


Figure 29: un extrait de l'ontologie du fauteuil de bureau

Les concepts de cette ontologie sont reliés par des relations de type : est composé de, est relié à, possède, etc.

Les éléments constituant cette ontologie sont lus et récupérer par l'agent « GCOntologie ». Ces éléments seront envoyés à l'agent « transformer » afin de transformer l'ontologie en requête SQL en appliquant les règles présentées ci-dessus. Le résultat de la transformation est une requête comme le montre la Figure 30. Cette requête est composée de plusieurs requêtes reliées par des « union ».

```
select v.nom_variable, v.valeur_variable, pc.id_projet, pc.resume_projet
from solveur_variables v, produit_element pe, produit p, projet_de_conception pc
where v.id_produit = pe.n_produit
and v.id_element = pe.n_element
and pe.n_produit = p.id_produit
and p.id_projet like '%fauteuil_bureau%'
and p.nom_produit like '%Chief_Point_10%'
and v.nom_variable like  '%marque%'
Union…
select v.nom_variable, v.valeur_variable, pc.id_projet, pc.resume_projet
from solveur_variables v, produit_element pe, produit p, projet_de_conception pc
where v.id_produit = pe.n_produit
and v.id_element = pe.n_element
and pe.n_produit = p.id_produit
and p.id_projet like '%fauteuil_bureau%'
and p.nom_produit like '%Chief_Point_10%'
and pe.nom_pe like '%structure%'
and v.nom_variable like  '%matiere%'
and v.valeur_variable = 'cuir'
UNION…
```

Figure 30: extraite de la requête SQL généré

Cette requête sera exécutée sur la base de données correspondante par l'agent « interroger » en utilisant un service web. Nous rappelons que la base de données ciblés est défini par l'expert métier lors de sa création de l'ontologie (voire chapitre 4, section 5.3.2). Le résultat de cette requête sera annoté par l'agent « annoter » selon une méthodologie que nous allons présenter dans la section 5.4. Ces informations seront par la suite sauvegardées dans des fichiers de format RDF et stockées dans la mémoire organisationnelle.

5.3.3. L'extraction des connaissances d'un fichier EXCEL

Comme nous l'avons expliqué précédemment, Les entreprises industrielles stockent leurs données dans des logiciels spécifiques comme une base de données relationnelle (SQL server, MySQL, ..), ou dans des fichiers de type Excel, XML, PDF, Texte ou d'autres types spécifiques.

Nous avons étudié dans la section précédente le cas d'une base de données relationnelle comme SQL server. Mais certains outils de CAO utilisés en domaine industriel ont comme format de sauvegarde des données des fichiers Excel tel que l'outil CATIA avec lequel nous avons réalisé nos tests.

Nous expliquons dans cette section notre méthodologie pour extraire des données des fichiers Excel ce qui représente le cas 2 de notre approche globale (Figure 25).

Pour ce faire nous avons choisi de transformer les fichiers comme Excel et XML en RDF comme expliqué dans l'introduction de ce chapitre.

5.3.3.1. La transformation des fichiers Excel en fichiers RDF

Pour réaliser la transformation excel-RDF, nous spécifions le fichier d'entrée (fichier Excel), le fichier de sortie (fichier RDF) et le fichier de correspondance dans lequel nous précisons les cellules d'Excel à transformer en langage RDF. Donc pour automatiser cette tâche nous avons posé des règles pour identifier automatiquement les cellules à transférer en RDF.

Ces règles sont basées sur l'analyse des fichiers Excel où nous avons constaté que ces fichiers contiennent un entête, des concepts et des données. L'entête correspond à des propriétés pour chaque concept par exemple le poids, la masse, une description, la quantité, la référence, etc. Alors que le concept est vu comme une pièce ou un produit. L'instanciation d'une propriété pour un concept donnée constitue une donnée. Donc les règles identifiés sont les suivantes :

R1 Le concept « entête » devient une classe

R2 Le concept « concept » devient une classe

R3 Le concept « donnée » devient une classe

R4 Les relations dans le fichier excel deviennent des Object properties dans le fichier rdf

R5 L'identifiant du « concept » devient une dataTypeProperty de type string

R6 Le nom de l'entête devient une dataTypeProperty de type string

R7 La Valeur devient une dataTypeProperty de type string

R8 Et voici comment on génère les individus dans le RDF :

- Chaque colonne de l'entête devient une instance de la classe entête
- Chaque ligne de la liste des concepts est une instance de la classe concept
- Chaque cellule de la liste des données est une donnée qui a comme caractéristique :
  - Un id individu (nom du concept_nom du fichier rdf_nom de l'attribut)

- Un label (description en langage naturel de l'individu)
- Une valeur (la valeur de la donnée dans la cellule)
- Le nom de l'entête en relation (grâce à l'Object properties hasAttribut)
- Le nom du concept en relation (grâce à l'Object properties hasIdentifiant)

En appliquant ces règles nous pouvons générer des fichiers RDF qui contiennent des classes, des datatypeproperties, des objectproperties, et des individus. Nous présentons dans la figure suivante un exemple d'un concept et d'une donnée généré en langage RDF à partir d'un fichier Excel.

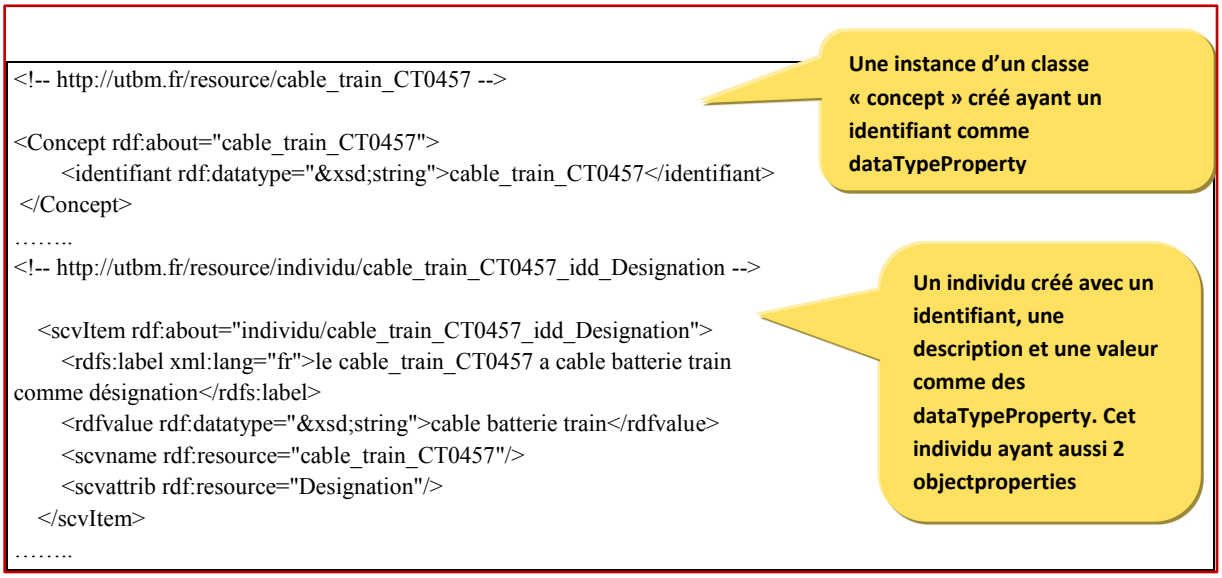


Figure 31: un extrait d'un fichier RDF généré à partir d'Excel

Cette figure montre une instance d'une classe créée. Cette classe est de type « concept » et possède la propriété identifiant qui lui donne une identité unique. La deuxième instance correspond à un individu. Ce dernier est identifié par un nom composé automatiquement en fonction de son contenu (exemple : individu/cable_train_CT0457_idd_Designation). L'individu comprend aussi un label qui décrit ce qu'il représente et une valeur. La relation entre un individu et une classe est présenté par la relation « scvname » qui fait référence à l'identifiant de la classe défini ci-dessus. De même pour le relier à un attribut nous définissons la relation « scvattrib » qui fait référence à une instance de la classe « Attribut ».

5.3.3.2. La transformation d'une ontologie en requête SPARQL

Nous avons vu dans la section précédente comment nous transformons les fichiers Excel en fichier RDF afin d'extraire les données plus facilement. Alors pour extraire les données du fichier RDF nous utilisons le langage de requête SPARQL.

Pour transformer une ontologie en requête SPARQL il faut raisonner au niveau des méta-modèles de l'ontologie et du langage de requête SPARQL pour trouver une transformation.

La transformation d'une ontologie en requête SPARQL repose alors sur une liste de correspondances. Le méta-modèle de l'ontologie est celui présenté par la figure 14. Quant à la requête SPARQL, elle est composée de la clause SELECT et WHERE (Prud'hommeaux and Seaborne, 2008).

Select {liste des ressources}
Where {liste des conditions sous forme de triplets}

Son méta-modèle est décrit dans la figure 33.

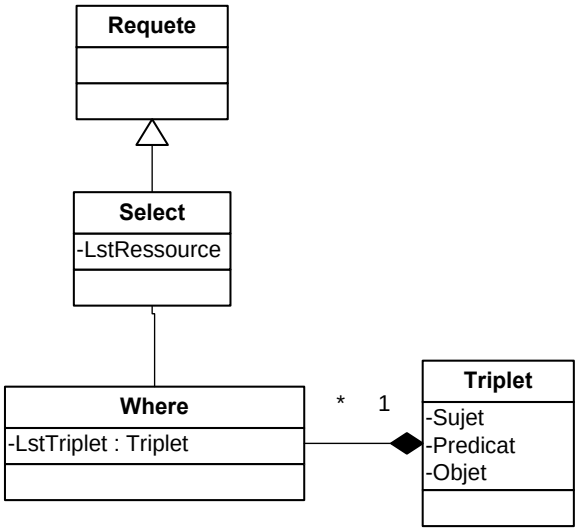


Figure 32: le méta-modèle d'une requête SPARQL

En analysant les deux méta-modèles (celui de l'ontologie et du langage de requête SPARQL), nous avons déterminé les règles suivantes :

1. Une classe de l'ontologie devient un triplé avec un sujet-prédicat-objet. Comme l'objet dans le triplé peut contenir des expressions de filtrage (condition), nous considérons le nom de la classe créé comme une expression de filtrage des résultats de la recherche et nous le mettons dans l'objet du triplé.
   Prenant le triplé:

   ?a0 <http://utbm.fr/resource/identifiant> ?id .filter(regex(str(?id), "roue_R475", "i")).

Dans cet exemple nous pouvons voir que le nom de la classe «roue_R475» est utilisé comme un objet dans le triplé et que cet objet contient un filtrage afin de recevoir juste les résultats ayant roue_R475 comme identifiant.

2. Pour chaque propriété définie par l'utilisateur il faut ajouter trois lignes de triplets et une ligne de condition (filtrage) :
   - Une première ligne, pour préciser que le parent de la propriété est le même que le parent des autres propriétés,
   - Une deuxième ligne, pour préciser qu'il a une valeur,
   - Une troisième ligne, pour préciser le nom de l'attribut,
   - La quatrième ligne est la condition (filtrage) si elle existe. Celle-ci est représentée par une expression de filtrage. Comme nous avons expliqué dans le chapitre 4 section 4.2.2, l'ontologie créé par l'expert métier comprend des classes, des propriétés et des relations et que les propriétés peuvent être simples ou conditionnées. Alors dans le cas où la propriété créé est simple nous n'ajoutons pas la quatrième ligne, dans le cas contraire nous ajoutons la ligne exprimant la condition définie par l'expert métier dans son ontologie. Prenant l'exemple suivant :

```
?a5 <http://utbm.fr/resource/scvname> ?id.
?a5 <http://utbm.fr/resource/rdfvalue> ?a6.
?a5 <http://utbm.fr/resource/scvattrib> ?a7.
        filter ((regex(str(?a7), "designation", "i"))&& (regex(str(?a6), "voiture", "i"))) .
```

La première ligne explique qu'on cherche les propriétés qui ont comme identifiant le même que celui défini par l'expert métier (voir exemple complet Figure 34).

La deuxième ligne signifie que nous cherchons les valeurs des propriétés.

La troisième ligne signifie qu'on cherche les noms des propriétés.

La quatrième ligne est une ligne de filtrage. Cette ligne signifie que nous ne cherchons pas tous les noms et les valeurs des propriétés mais seulement ceux qui contiennent dans leur nom le mot « désignation » et dans leur valeur le mot « voiture ».

Alors les restrictions dans le fichier owl deviennent des expressions de filtrage dans la requête. Voici quelque exemple :

- Si la restriction sur la propriété « est diffèrent de » une valeur alors elle sera traduite par :
  Filter ((regex (str (?variable), nom_propriété, i)) && (?cond != valeur_propriété))
- Si la restriction sur la propriété « est égale de » une valeur alors elle sera traduite par :
  Filter ((regex (str (?variable), nom_propriété, i)) && (regex (str (?cond), valeur_propriété, i))
- Si la restriction sur la propriété une « datarange » de plusieurs valeurs alors elle sera traduite par :

Filter ((regex (star (?variable), nom_propriété, i)) && (?cond= val1 || ?cond= val2 || ?cond=val3 || ?cond= val4))

Comme dans le cas de la transformation d'une ontologie en requête SQL, l'agent « GCOntologie » lit l'ontologie et identifie ses éléments (classes, propriétés et relations). Puis, il l'envoie à l'agent « transformer » pour produire une requête SPARQL exécutable sur les fichiers RDF (ces fichiers sont le résultat de la transformation des fichiers Excel en RDF). Le résultat de cette interrogation sera annoté en utilisant l'ontologie « OntoDesign » et sauvegardé dans la mémoire organisationnelle par l'agent « annoter ».

Nous présentons un exemple d'extraction des données d'un fichier Excel pour illustrer les règles ci-dessus. La Figure 33 montre un extrait d'une ontologie qui décrit la composition d'un produit mécanique à rechercher.

Cette ontologie cherche les produits ayant un identifiant roue_R475 et qui ont comme référence une des valeurs (1 ou 2 ou 3 ou 4 ou 5 ou 6). Ces produits doivent avoir un type différent de 4x4 et une désignation qui comprend le mot voiture.

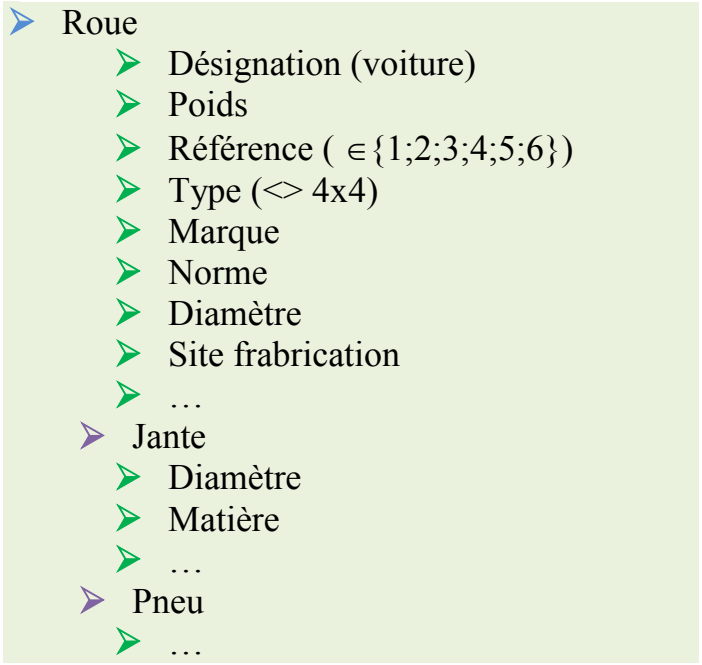


Figure 33: extrait d'une ontologie d'un produit

Quand l'agent « transformer » applique les règles citées ci-dessus sur cette ontologie, il retourne une requête SPARQL (Figure 34) à l'agent « interroger » pour que ce dernier l'exécute sur les fichiers RDF par le biais d'un service web.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT *
WHERE {
?a0 rdf:type <http://utbm.fr/resource/Concept>
?a0 <http://utbm.fr/resource/identifiant> ?id .filter(regex(str(?id), "roue_R475", "i")).
?a1 rdf:type <http://utbm.fr/resource/scvItem> .
?a1 <http://utbm.fr/resource/scvname> ?id .
?a1  <http://utbm.fr/resource/rdfvalue> ?a3 .
?a1  <http://utbm.fr/resource/scvattrib> ?a4 .
          filter ((regex(str(?a4), "reference", "i")) && (?a3="1.0" || ?a3="2.0" || ?a3="3.0" ||
?a3="4.0" || ?a3="5.0" || ?a3="6.0")) .
?a5 <http://utbm.fr/resource/scvname> ?id.
?a5  <http://utbm.fr/resource/rdfvalue> ?a6.
?a5  <http://utbm.fr/resource/scvattrib> ?a7.
          filter ((regex(str(?a7), "designation", "i"))&& (regex(str(?a6), "voiture", "i"))) .
?a8 <http://utbm.fr/resource/scvname> ?id.
?a8  <http://utbm.fr/resource/rdfvalue> ?a9.
?a8  <http://utbm.fr/resource/scvattrib> ?a10.
          filter ((regex(str(?a10), "type", "i"))&& (?a9!= "4x4")) .

          ….
          ….
}
```

Figure 34: extrait d'une requête SPARQL généré

Nous avons vu dans les sections précédentes que quelle que soit l'application métier qu'on utilise nous avons des données extraites comme résultat. Ces données résultantes de l'exécution des requêtes seront formatées et annotées. Suite à l'annotation, les données deviennent des informations puisqu'elles sont mises dans un contexte. Ceci est un facteur important pour la réutilisation. Nous présentons dans la section suivante notre approche pour annoter les données.

5.4.Annotation des données

Notre approche d'annotation des données est basée sur l'ontologie OntoDesign. Les annotations sont réalisés au format RDF (Ressource Description Framework) ce qui permettent de décrire les informations identifiées. L'agent « annoter » utilise alors les concepts de cette ontologie pour spécifier pour chaque donnée le contexte dans lequel elle a été identifiée. Ce contexte est ainsi défini par le triplet Projet-Acteur-Rôle. La Figure 35 présente un extrait de l'ontologie OntoDesign (Projet, Rôle et Acteur) en spécifiant les relations entre ces différents concepts.

L'agent « annoter » consulte l'ontologie OntoDesign pour associer aux données extraites le projet dans lequel elles étaient créées et pour ce projet l'acteur qui l'a réalisé, son rôle dans le projet, de quelle application métier a été extrait ce projet, les domaines d'application du projet et la maturité du projet. Les projets sont mesurés par leur maturité, alors plus l'indice de maturité est élevé plus le projet est considéré comme pertinent pour l'acteur métier. Cet indice de maturité est calculé en divisons le note globale, mises par les acteurs métiers (voire chapitre 6), par le nombre de votants pour ce projet. Cette mesure de maturité est nulle avant le premier passage du projet dans le système d'évaluation des connaissances (chapitre 6).

La relation d'héritage entre les projets est représenté par la relation « sous classe de » ce qui permet de retourner aux acteurs métiers des projets liés directement au indirectement à sa recherche.

Les informations présente dans OntoDesign (projet, date début, date fin, rôle, ..) sont extraites de l'application métier ciblé par l'expert métier et qui a été utilisé pour extraire les données défini dans l'ontologie métier.

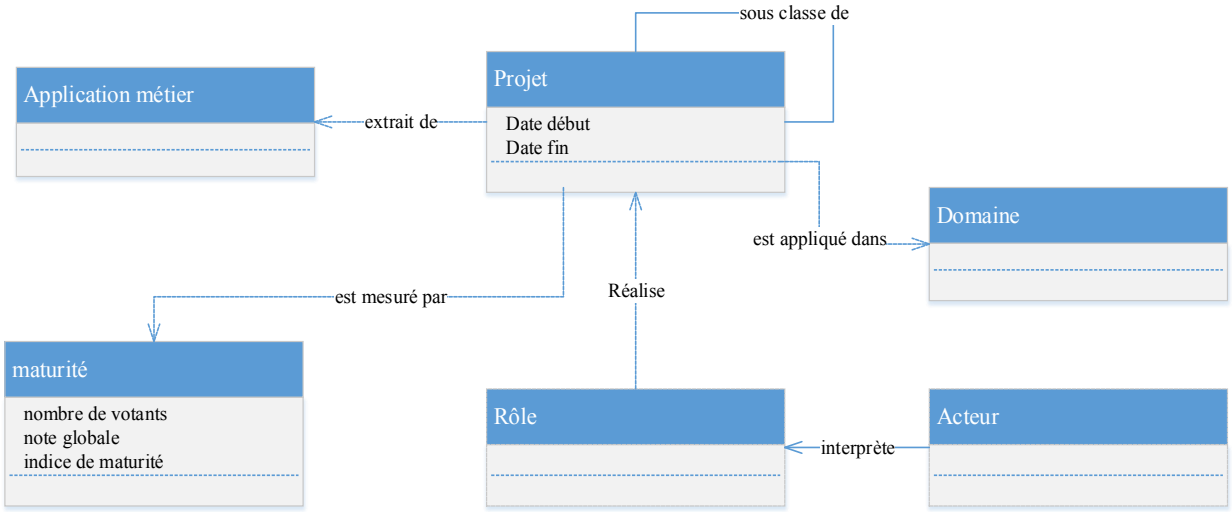


Figure 35: extrait de l'ontologie OntoDesign pour l'annotation des connaissances

Les données annotées seront sauvegardées dans un fichier de type RDF et stockées dans la mémoire organisationnelle. Cette mémoire sera la base d'un wiki sémantique (voire chapitre 6) où les acteurs métiers peuvent rechercher des projets déjà réalisés et réutiliser les connaissances existantes.

La Figure 36 montre un extrait d'un fichier RDF. La première annotation "rdf: Description rdf: about =" https://utbm.acsp.fr/ fauteuil_bureau_chief_point_10" spécifie

l'application métier duquel nous avons extrait les informations (ici l'outil s'appelle ACSP), et le nom du produit (ici fauteuil_bureau_chief_point_10).

Après la présentation de l'application métier et du projet, les agents introduisent les autres informations présentes dans l'ontologie OntoDesign. Par exemple on voit ici que le projet a été créé par ilahoud qui a comme rôle un concepteur. En plus on sait maintenant que le projet a débuté en septembre 2011 et a fini en octobre 2011.

```
<rdf:RDF
   xmlns:fauteuilBureau="http://OCEAN/ActivitePropriete/"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
  <rdf:Description rdf:about=
"http://utbm.acsp.fr/fauteuil_bureau_chief_point_10/">
<FauteuilBureau:RoleCreateurConnaissance>Concepteur
</FauteuilBureau:RoleCreateurConnaissance>
<fauteuilBureau:CreerPar>ILahoud</FauteuilBureau:CreerPar>
<FauteuilBureau:DateDebut>16/09/2011</FauteuilBureau:DateDebut>
<FauteuilBureau:DateFin>02/10/2011</FauteuilBureau:DateFin>
….
<FauteuilBureau:marque> TOPSTAR </FauteuilBureau:marque>
<FauteuilBureau:gamme> Fauteuil de direction </FauteuilBureau:gamme>
<FauteuilBureau:fonction> Modulable </FauteuilBureau:fonction>
<FauteuilBureau:matiereStructure>cuir</FauteuilBureau:matiereStructure>
<FauteuilBureau:mecanismeFauteuil>Système à verrin
 </FauteuilBureau: mecanismeFauteuil>
<FauteuilBureau: GarnissageAssise> Mousse de polyuréthane
</FauteuilBureau: GarnissageAssise>
<FauteuilBureau:densiteAssise> 24 kg/m3
 </FauteuilBureau:densiteAssise>
<FauteuilBureau:largeurAssise> 49 cm
</FauteuilBureau: largeurAssise>
<FauteuilBureau:hauteurAssise> 42 cm </FauteuilBureau: hauteurAssise>
. …..
……..
</rdf:Description>
</rdf:RDF>
```

Les annotations ajoutées suivant l'ontologie décrite dans la figure 12

Les données typées résultat de l'extraction d'une application métier

Figure 36: un extrait du fichier RDF résultat de l'extraction

Nous voyons par la suite toutes les informations demandées par l'expert métier dans son ontologie de départ comme la marque et la gamme du fauteuil de bureau avec les bonnes valeurs.

5.5.Conclusion

Le but de chapitre était de présenter le deuxième module de notre système de gestion des connaissances hétérogènes et distribuées OCEAN. Ce module utilise des ontologies formelles pour extraire des informations à partir d'applications métier dans une première étape. Puis dans une deuxième étape est d'annoter ces informations pour construire une base de connaissances au format RDF.

Mais pour atteindre cet objectif, nous sommes confrontés à un problème de la transformation du modèle d'ontologie au modèle de requête (SQL ou SPARQL) tel que décrit dans le présent chapitre. C'est pourquoi nous avons expliqué la raison pour laquelle nous faisons une transformation entre ces différents modèles.

Nous avons présenté dans ce chapitre deux cas de transformation : la transformation d'une ontologie en requête SQL et d'une ontologie en requête SPARQL, la façon dont nous l'avons fait et des exemples de ces transformations.

Les données résultantes de l'extraction dans les deux cas présentés sont annotées et sauvegardées dans une MO. Cette base sera considérée comme la base du wiki sémantique que nous allons le présenter dans le chapitre suivant. Ce wiki servira à valider et à faire évoluer les informations et les connaissances et permet aux acteurs métiers de réutiliser les connaissances existants.

Nous avons présenté également pendant ce chapitre les agents qui assurent le fonctionnement de ce module et les rôles qu'ils occupent.

CHAPITRE 6

LA DIFFUSION ET LA REUTILISATION DES CONNAISSANCES

Ce chapitre présente notre approche pour diffuser les connaissances auprès des acteurs métiers au sein de l'entreprise étendue. Cette approche est basée sur un wiki sémantique qui sert de point d'accès à la mémoire organisationnelle présentée au chapitre précédent. Le principe sous-jacent est, d'une part, de permettre le partage des connaissances afin de faciliter le travail individuel. D'autre part, cette approche réutilise des travaux antérieurs qui permettent de réutiliser de manière automatique les connaissances capitalisées.

Table de matières

6.1. Introduction

Nous avons présenté, dans les chapitres précédents, notre approche pour définir, spécifier et extraire les connaissances. Les éléments du SGC OCEAN détaillés dans ces chapitres permettent :

- d'assister les experts métiers lors de la création d'ontologies pour l'identification des connaissances,
- d'extraire les données des applications métiers,
- d'annoter et de sauvegarder ces données dans une mémoire organisationnelle.

L'ensemble de ces activités vise essentiellement à rendre possible la diffusion et la réutilisation des connaissances. En d'autres termes, l'objectif est de rendre ces connaissances accessibles aux acteurs métiers.

Nous complétons ainsi le cycle de vie de notre système de gestion de connaissances OCEAN, en décrivant le troisième et le quatrième objectif qui consistent en la diffusion et la réutilisation des connaissances. En effet, une fois stockées, l'objectif d'un SGC est de diffuser les connaissances. Cette diffusion représente un échange, ou transfert, de connaissances entre une source et un destinataire. Cet échange peut se faire de manière indirecte grâce à l'aide de technologies de l'information et de la communication. De plus, la diffusion qui naturellement se fait dans des cercles limités d'accointances au sein d'une entreprise classique est rendue d'autant plus délicate dans le cadre d'une entreprise étendue. Un SGC se doit donc de permettre l'accès et le filtrage des connaissances afin de rendre cette diffusion efficace. Notre approche pour la diffusion des connaissances repose sur la conception, la réalisation et l'utilisation d'un Wiki sémantique qui permet la validation, le partage, la réutilisation et l'évaluation des connaissances stockées dans une mémoire organisationnelle résultant des techniques d'extraction présentées au chapitre précédent.

De manière générale, un wiki est une application web qui permet la création collaborative d'informations et l'édition de contenu hypertexte.

Schaffert (Schaffert, 2006) énumère les spécifications d'un système de wiki:

- Il permet l'édition via un navigateur web;
- Il a une syntaxe wiki simplifiée utilisable par tous les utilisateurs;
- Il gère un mécanisme de restauration ce qui signifie qu'il est capable de versionner les variations de contenu ;

- Son accès est libre, tout le monde peut écrire sur le wiki ;
- Il gère l'édition collaborative : si quelqu'un créer un article, tout le monde peut étendre cet article ;
- Toutes les pages du wiki sont liées les unes aux autres à l'aide des liens hypertexte ;
- Des recherches sur le contenu de toutes les pages stockées sont possibles ;

Au-delà de ces caractéristiques communes aux systèmes de type wiki, les wikis sémantiques possèdent une structure sémantique caractérisant les contenus entrés par les utilisateurs. Cette structure sémantique, lisible et manipulable par un ordinateur, permet d'améliorer les possibilités de navigation, d'interrogation, de partage et de réutilisation des connaissances.

A la vue de ces éléments, les wikis peuvent être considérés comme de bons candidats pour la gestion des connaissances collaborative basée sur les technologies Web2.0 (Richards, 2009). En effet, de nombreux travaux de recherche (Schaffert, 2006)(Vrandecic and Krötzsch, 2006) proposent des wikis pour l'échange de connaissances.

Ce chapitre, décrit notre approche pour diffuser les connaissances en présentant l'architecture du Wiki sémantique avec ses fonctionnalités et son mode de fonctionnement. Nous rappelons que nous n'avons pas abordé dans cette thèse le quatrième objectif du cycle de vie qui est la réutilisation des connaissances. Pour cet aspect nous avons réutilisé la technique décrite dans les travaux de Ben Miled (Ben Miled et al., 2008) (Ben Miled et al., 2009) pour réutiliser les connaissances capitalisées et par conséquent compléter notre cycle de gestion des connaissances.

6.2.Wiki sémantique pour la diffusion des connaissances

6.2.1. Les hypothèses

Nous avons évoqué dans les chapitres précédents que notre travail se situe dans le cas d'une entreprise étendue. Cette entreprise est confrontée au problème de management d'une grande diversité de connaissances. La question qui se pose est celle de la diffusion entre les différentes entreprises industrielles qui la constituent, et entre les employés de chaque entreprise. Alors, pour faire circuler les connaissances au sein de cette entreprise étendue nous

proposons un wiki sémantique. Ce dernier a prouvé son capacité de manipuler et partager les connaissances d'une façon simple et rapide.

Depuis 2004, avec le développement de Platypus Wiki (Campanini et al., 2004), de nombreux wikis sémantiques ont été créés : SweetWiki (Buffa et al., 2008), MaknaWiki (Dello et al., 2006), IkeWiki (Schaffert, 2006), OntoWiki (Auer et al., 2006), Shawn (Aumüller, 2005), Rise (Decker et al., 2005), Semantic MediaWiki (Krötzsch et al., 2007), WikSar (Aumüller and Auer, 2005)). Ces wiki sémantiques offrent la possibilité de stocker en RDF et modifier ce contenu. L'ensemble des wikis sémantiques cités ci-dessus sont construits selon le modèle « wikitology » (Decker et al., 2005) c'est-à-dire que les pages du wiki sont considérés comme des concepts et les liens (dans le contenu de la page et entre les pages du wiki) sont considérés comme des relations ou des attributs. Le wiki peut être ainsi considéré comme une ontologie composée des concepts, des attributs et des relations, autrement dit, il compose l'ontologie.

Notre approche repose sur une hypothèse différente, inspirée de WikiDesign (Monticolo and Gomes, 2011). Ce dernier wiki sémantique, comme SweetWiki et IkeWiki, utilise une ontologie pour l'édition du contenu des pages et des métadonnées.

Néanmoins, à la différence de ces deux wikis sémantiques, WikiDesign a été conçu : (i) pour être associé à un système de gestion des connaissances et (ii) pour faciliter l'évaluation et la création de nouvelles connaissances à partir des mémoires de projets existants. WikiDesign utilise une ontologie de domaine (OntoDesign) pour éditer ses pages.

Nous présentons dans la section suivante l'architecture de notre wiki sémantique qui inclut un SMA.

6.2.2. L'architecture du wiki

Le wiki sémantique est basé sur le contenu de la mémoire organisationnelle. Cette mémoire organisationnelle est construite à la fin de la deuxième étape du cycle de vie de la gestion des connaissances qui est l'extraction des connaissances. Nous avons expliqué dans le chapitre précédant comment le SMA contribue à l'objectif d'extraction des connaissances.

Le fait de contextualiser les données extraites par les agents les transforme en information. Ces informations sont stockées dans un fichier de type RDF. L'ensemble de ces fichiers RDF constitue la mémoire organisationnelle. Il faut noter que les fichiers RDF ne

contiennent que des informations. Ces informations se transforment en connaissances lorsqu'elles sont interprétées par un humain comme expliqué dans le chapitre 2. Dans notre cas, les informations stockées dans les fichiers RDF deviennent des connaissances après un passage dans le wiki sémantique et validation par un acteur métier.

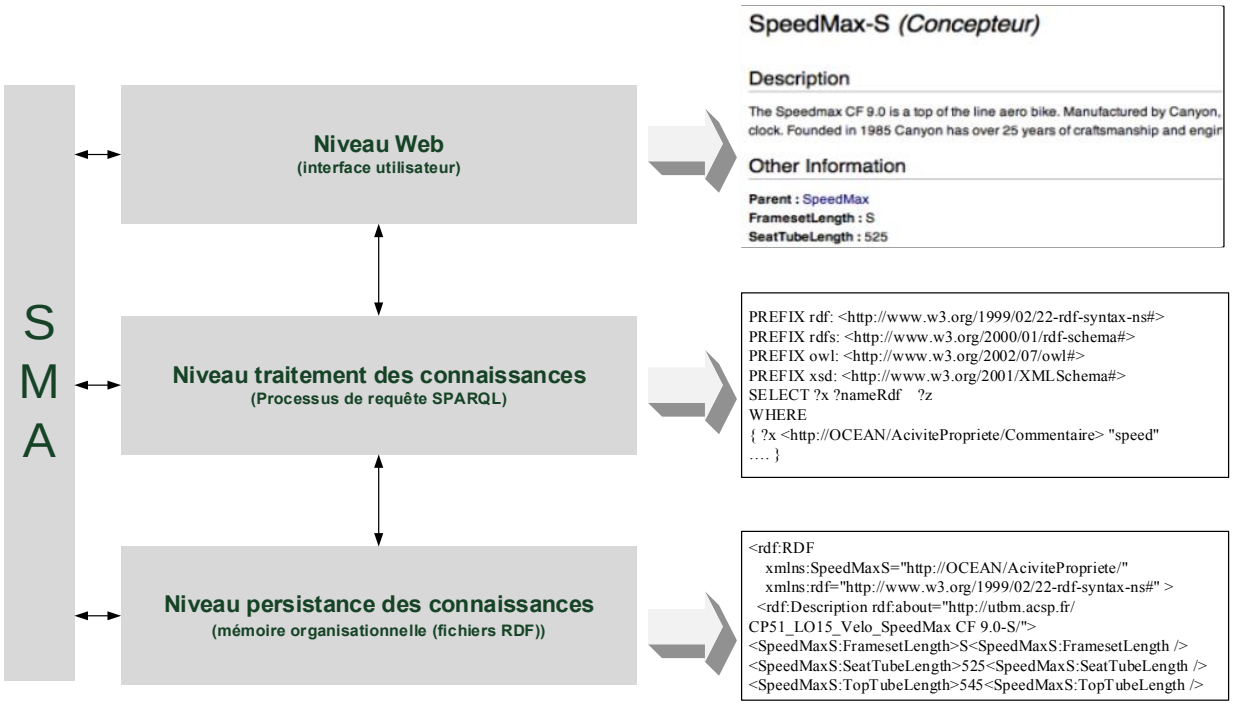


Figure 37: architecture globale du wiki sémantique

La Figure 37 montre les trois niveaux constituant notre wiki sémantique qui sont : le niveau web, le niveau traitement des connaissances et le niveau persistance des connaissances. Les trois niveaux sont gérés par le système multi-agents.

Notre wiki sémantique permet de diffuser les connaissances de la mémoire organisationnelle. Pour accéder à ces connaissances, l'acteur métier se connecte au wiki et lance une recherche (requête) selon ses besoins. Cette requête est traitée par les agents qui se chargent de rechercher dans la mémoire organisationnelle, par le biais des requêtes SPARQL, les fichiers RDF correspondants. Le résultat de la recherche sera présenté dans l'interface du wiki comme illustré dans la Figure 37.

Une recherche produit ainsi en résultat un ensemble de fichiers RDF à partir de la mémoire organisationnelle. Nous considérons dans la suite de ce chapitre qu'un article wiki correspond à un fichier RDF de la mémoire organisationnelle. Comme nous travaillons dans le domaine industriel, chaque article contient un projet décrivant la conception d'un produit mécanique tel que la conception d'un moteur d'une voiture précise ou bien la conception

d'une chaise de bureau (voire chapitre 5 pour un exemple complet du contenu d'un fichier RDF).

Dans la section suivante, nous détaillons les différentes fonctionnalités de ce wiki. Pour chaque fonctionnalité nous pouvons voir le mécanisme de son fonctionnement et l'agent qui l'assure.

6.3.Les fonctionnalités du wiki

Notre wiki sémantique a pour objectif principal de diffuser les connaissances au sein de l'entreprise en permettant aux acteurs métiers la navigation et la recherche sémantique dans les articles du wiki. Ce wiki assure alors les fonctionnalités suivantes : la recherche sémantique, la classification, la représentation, la modification et l'évaluation des connaissances (Figure 38).

La première fonctionnalité « recherche sémantique », cf. Figure 38, sert à rechercher les projets existants dans la mémoire organisationnelle suite à la demande d'un acteur métier. Grimes énumère, cf. (Grimes, 2010), onze approches qui lient la sémantique à la recherche. Mais avant de présenter l'approche de recherche de notre wiki il faut tout d'abord définir les formes de recherches que nous utilisons.

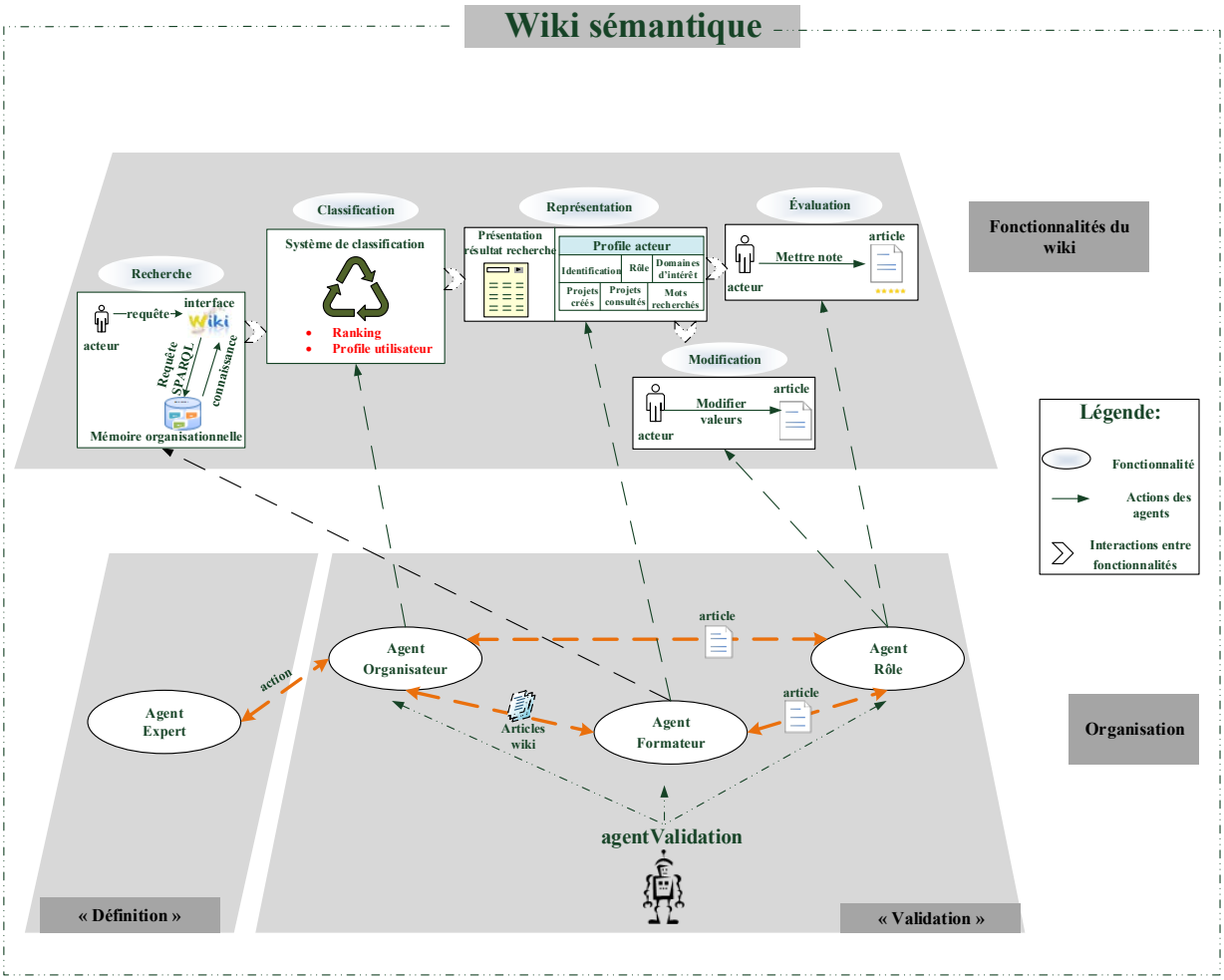


Figure 38: Les fonctionnalités du Wiki sémantique

Guha (Guha et al., 2003) différencie deux formes de recherche: la navigation et la recherche. Dans la première, l'acteur métier utilise le moteur de recherche du wiki comme un outil de navigation pour trouver l'article ciblé. Tandis que, dans la deuxième forme, la recherche sémantique, l'acteur métier peut fournir au moteur de recherche du wiki une phrase qui désigne un objet sur lequel l'acteur métier tente de recueillir de l'information. Dans ce cas l'acteur ne désigne pas un article particulier mais au contraire il essaie d'avoir un certain nombre d'article qui lui donne l'information qu'il cherche.

La navigation « classique » dans notre wiki s'effectue en cliquant sur un projet. Ce clic a pour effet d'ouvrir un article wiki lié à ce projet. Dans la page projet sont présentées toutes les connaissances liées à ce projet.

Pour réaliser la fonctionnalité « recherche sémantique », tel que montré dans la Figure 38, l'agent « expert » de l'organisation « Définition » écoute l'interface du wiki sémantique.

Quand cet agent détecte une demande de recherche d'un acteur métier il déclenche l'agent « formateur » de l'organisation « Validation ». Ce dernier permet de formuler des requêtes pour exploiter les connaissances de la mémoire organisationnelle avec le langage SPARQL. Les requêtes correspondent aux termes entrés par l'acteur métier dans le wiki. Comme la montre la figure ci-dessus, l'agent « formateur » effectue une recherche sémantique qui retourne des connaissances reliées directement ou sémantiquement à la requête de l'acteur métier. Nous détaillons cette fonctionnalité dans la section (6.3.1).

Les résultats de la recherche effectuée par l'agent « formateur » ne sont pas retournés directement à l'acteur mais ils sont envoyés à l'agent « organisateur ». Dans ce cas nous passons à la deuxième fonctionnalité de la figure ci-dessus et qui est la classification. Alors l'agent organisateur traite les résultats qu'il a reçus et opère un tri sur eux en appliquant deux méthodes (section 6.3.2). Ceci permet de structurer les résultats et de les présenter à l'acteur par le plus pertinent pour lui. La première méthode consiste à classer les résultats (articles wiki) par le nombre croissant de vote positif que possède chacun d'eux. Ce système de vote est assuré par la fonctionnalité « évaluation ». La deuxième permet d'ordonner les résultats selon le profil de l'acteur métier.

La troisième fonctionnalité « présentation » consiste en deux tâches assurées par l'agent « formateur ». Cet agent assure, d'une part, la présentation des résultats de recherche (résultat des deux premières fonctionnalités), classifié par l'agent « organisateur », à l'acteur métier. D'autre part, l'agent « formateur » permet à l'acteur métier de gérer son profil. Dans ce profil, il peut définir les différents rôles qu'il a dans le projet ou ses domaines professionnels. Selon ces informations, notre wiki filtre les résultats de la requête et présente ceux qui sont liés à son profil. Nous présentons cette fonctionnalité dans la section 6.3.3

Une fois les résultats (articles wiki) sont présentés dans le wiki par l'agent « formateur », l'acteur a la possibilité de les modifier ou les évaluer. Ces deux fonctionnalités sont alors la « modification » et « l'évaluation ». La Figure 38 montre les deux possibilités par les flèches présentes entre les trois fonctionnalités. Ces deux fonctionnalités seront assurées par l'agent « rôle ».

La fonctionnalité évaluation des articles se traduit par le fait d'attribuer une note entre 1 et 5. La note 1 correspond à une mauvaise note et 5 à une bonne note. Plus l'article possède de bonnes notes, plus il est considéré comme un article pertinent. Nous allons ainsi expliquer comment notre wiki garantit la fiabilité des connaissances grâce à ce processus d'évaluation (section 6.3.4).

En navigant dans les articles du wiki, l'acteur peut consulter les connaissances que possède chaque article. L'acteur a également la possibilité de modifier les valeurs liées aux connaissances s'il a les droits appropriés. Nous expliquons les deux fonctionnalités évaluation et modification des articles du wiki dans la section 6.3.4.

Toutes les connaissances à l'intérieur des articles du notre Wiki sont annotées en fonction de l'ontologie selon le mécanisme détaillé au chapitre 5. Nous décrivons en détail dans la section suivante les différentes fonctionnalités que nous venons de décrire brièvement.

6.3.1. Recherche sémantique

Notre wiki dispose d'un moteur de recherche sémantique pour interroger et raisonner sur la mémoire organisationnelle. La recherche sémantique sert à améliorer la précision de la recherche en comprenant l'objectif de recherche et la signification textuelle de la requête de l'acteur métier tel qu'il l'a écrit dans notre wiki sémantique. Le but de cette recherche est d'avoir des résultats plus pertinents pour l'acteur métier. Par exemple si vous demanderez la requête « A quelle vitesse court un jaguar », le système de wiki doit comprendre que vous ne parlez pas de la marque de voiture, et saura analyser les termes de la question pour vous donner la réponse.

Notre idée pour la recherche sémantique était de réaliser des croisements d'informations entre les différents articles du wiki et de le cumuler avec les informations personnelles des acteurs métiers. Ce croisement se fait en liant les articles par les annotations (le lien de parenté par exemple). Les informations personnelles seront définies dans le profil utilisateur dans le wiki. La liaison avec les informations personnelles permet de donner des résultats proche des centres d'intérêts de l'acteur ce qui limite le nombre de résultats non pertinents.

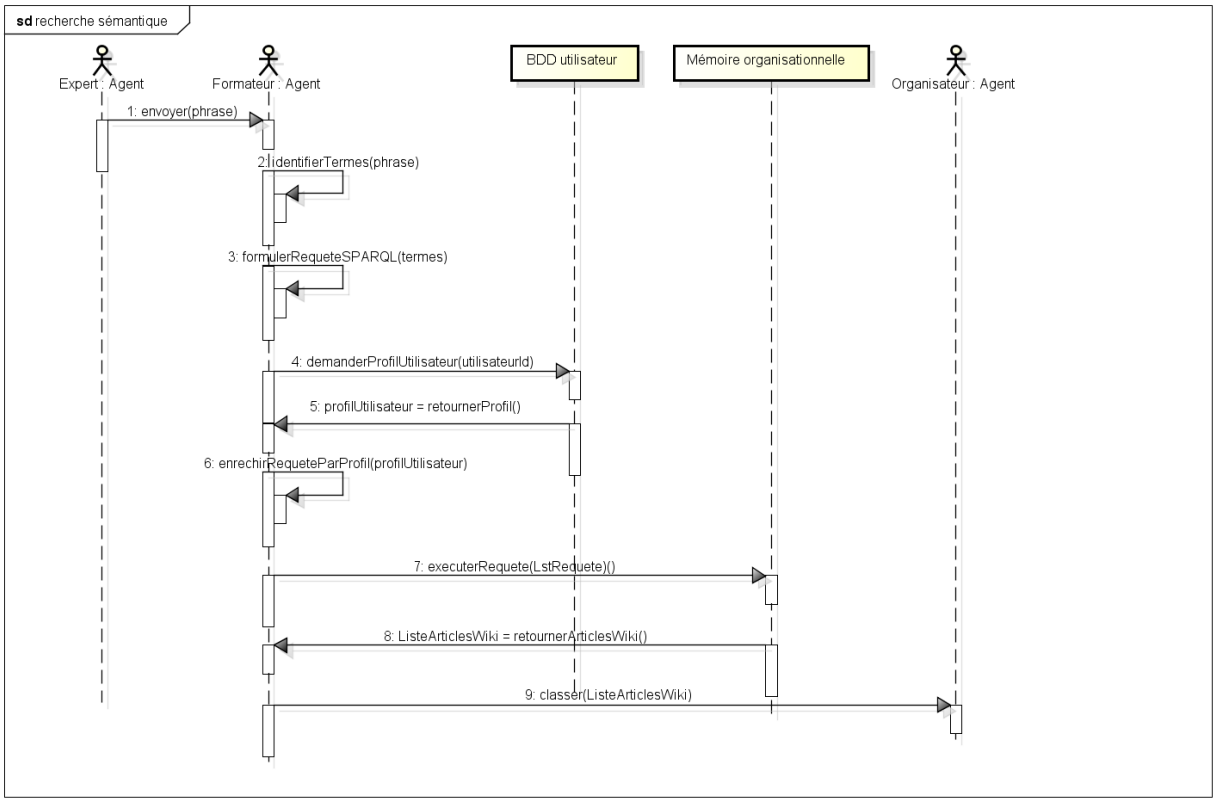


Figure 39: Diagramme de séquence de la fonctionnalité recherche sémantique

Nous présentons en détails les interactions entre les agents et les objets (mémoire organisationnelle et BDD utilisateur) dans un diagramme de séquence (Figure 39).

Pour effectuer cette recherche sémantique, l'agent « expert » de l'organisation « Définition » monitore l'interface du wiki sémantique et réagit à chaque introduction des termes (requête) dans l'interface. Cette réaction consiste à communiquer avec l'agent Formateur, responsable de la recherche, et à lui envoyer les termes introduits (fonction 1). L'agent formateur identifie les termes dans la requête de l'acteur métier (fonction 2) et formule des requêtes de type SPARQL (fonction 3). Pour enrichir ces requêtes, l'agent formateur demande le profile utilisateur de l'acteur métier connecté depuis la base des données des utilisateurs (fonction 4). Ce dernier lui retourne toutes les informations sur l'acteur demandé (fonction 5). L'agent formateur récupère ces informations sur le profil et enrichit les requêtes SPARQL par les informations personnelles de l'acteur métier (fonction 6) afin d'avoir des réponses plus spécifiques et pertinentes. Ces requêtes seront utilisées pour rechercher les articles wiki exprimés en langage RDF et stockés dans la mémoire organisationnelle (fonction 7). Le processeur de requêtes utilise l'API Jena (McBride, 2002)(Rajagopal, 2005). Jena permet de charger les modèles ontologiques au format OWL ou RDFS et gère le langage SPARQL. Ce dernier est basé sur une combinaison booléenne de

triplets qui peuvent être contraints par des expressions évaluables. SPARQL est récemment proposée comme une recommandation du W3C pour l'interrogation des fichiers RDF (Prud'hommeaux and Seaborne, 2008). Le résultat de l'interrogation sur cette mémoire est une liste des articles wiki répondant sémantiquement à la requête de l'acteur métier (fonction 8). Le résultat de la recherche sémantique est une liste des articles wiki. Mais cette liste est retournée sans aucun classement. Alors pour présenter les résultats d'une façon structurée à l'acteur nous appliquons une approche de classification. Cette approche consiste à envoyer la liste des articles wiki par l'agent formateur à l'agent organisateur afin de les classifier (fonction 9). Cette fonction est le sujet de la deuxième fonctionnalité du wiki. Nous expliquons en détail cette fonctionnalité dans la section suivante.

6.3.2. Classification des connaissances

Pour rendre la présentation des résultats d'une requête plus pertinents à l'acteur métier nous présentons une approche de classification pour les afficher par le plus important pour lui (Figure 41). Dans cette approche, nous utilisons deux méthodes de classification : le profil de l'utilisateur et l'évaluation des articles wiki.

La première consiste à ordonner les résultats en appliquant différents types de filtres tel que 1) le rôle de l'utilisateur, 2) les domaines d'intérêt de l'utilisateur défini dans son profil, 3) les projets qu'il consulte, 4) les projets qu'il crée, et 5) l'historique de ses recherches dans le wiki.

La deuxième méthode consiste à trier les résultats par son indice de maturité. Cet indice correspond au nombre des évaluations positives que la page de wiki possède. Nous obtenons cet indice en divisant la note globale (rate) par le nombre de votants (nbRaters).

Plus l'indice de maturité est grand plus il apparait en début de la liste des résultats. Nous expliquons la démarche pour évaluer une page dans la fonctionnalité évaluation (section 6.3.4).

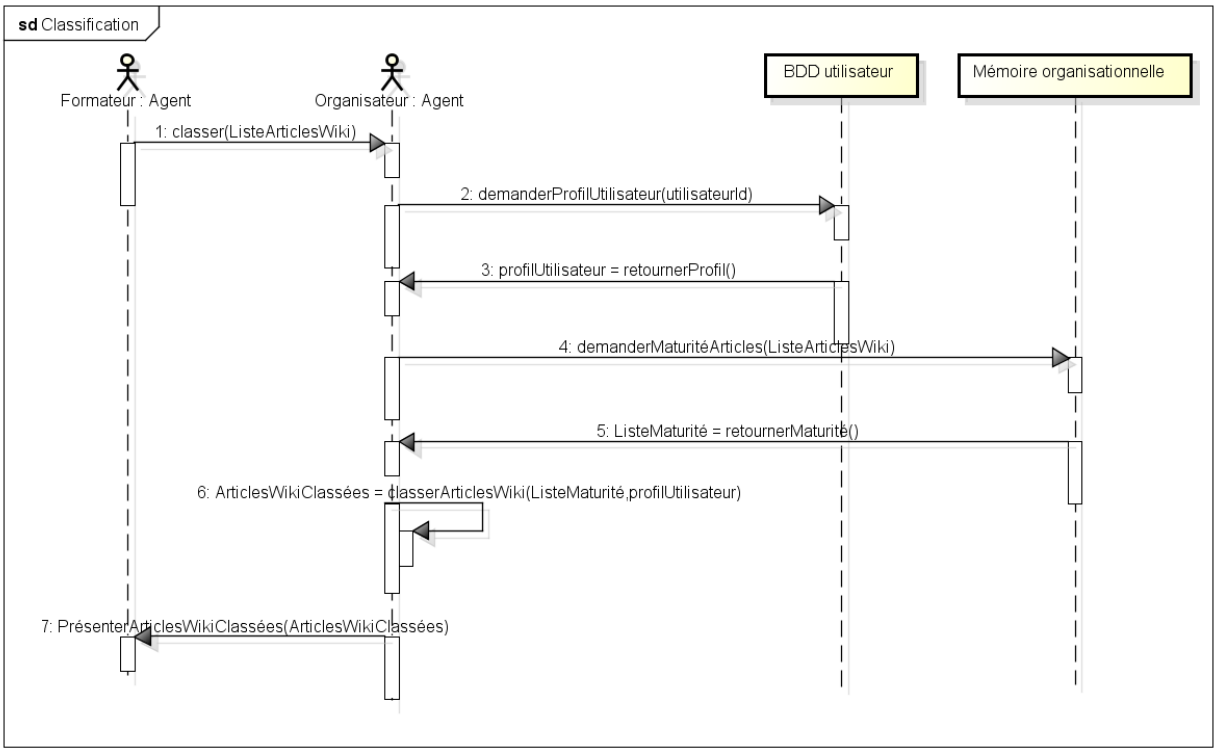


Figure 40: Diagramme de séquence de la fonctionnalité classification des connaissances

Nous présentons dans la Figure 40 les interactions entre les agents (formateur et organisateur) d'une part et avec les objets (base de données utilisateur et la mémoire organisationnelle) pour assurer cette fonctionnalité.

Quand l'agent formateur cherche les articles wiki correspondants à la requête de l'acteur, il les envoie à l'agent organisateur pour les classer (fonction 1). Dans ce but, ce dernier demande le profil de cet acteur métier pour connaitre ses centres d'intérêts, son rôle, etc. (fonction 2). Quand l'agent organisateur reçoit le profil de l'acteur (fonction 3), il lance une deuxième demande mais auprès de la mémoire organisationnelle pour connaitre l'indice de maturité de chaque article wiki reçu par l'agent formateur (fonction 4). Une fois que l'agent organisateur possède la liste des maturités pour les articles wiki (fonction 5) et le profil de l'acteur métier il classe alors les articles selon ces deux critères et retourne par conséquent une liste des articles wiki classés (fonction 6). Cette liste des articles classés est envoyée par l'agent organisateur à l'agent formateur pour que ce dernier la présente à l'acteur métier.

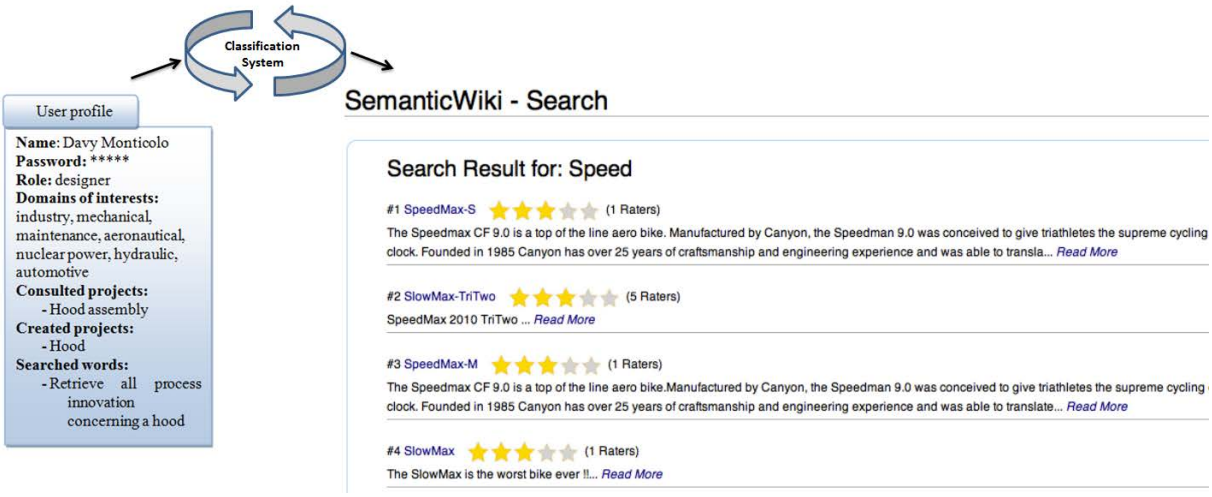


Figure 41: L'approche de classification des projets résultat de la requête

La Figure 41 montre un exemple du mécanisme de classification dans lequel l'utilisateur "Davy" occupe le rôle "concepteur". Cet utilisateur s'intéresse aux domaines de « l'industrie, mécanique, maintenance, aéronautique, nucléaire, hydraulique, automobile", donc quand il cherche dans le système le mot «speed» par exemple le système de classification trie les résultats par le domaine mécanique, car il apparaît aussi dans les domaines de la restauration, de la sécurité, les sports, etc.

6.3.3. Représentation

Cette fonctionnalité repose sur deux tâches effectuées par l'agent « formateur ». Cet agent s'occupe de la présentation des résultats de la recherche sémantique à l'acteur métier. Une vue sur la page de recherche est illustrée par la Figure 42. L'acteur métier introduit une phrase dans la zone de recherche et l'agent lui retourne une liste des articles wiki affichée dans la même page. Cette interface lui permet alors de rentrer un ou plusieurs mots pour rechercher dans la mémoire organisationnelle ses besoins. Le résultat de la recherche sera présenté par ordre décroissant de pertinence pour l'acteur en fonction de son profil et ensuite par maturité décroissante. Chaque résultat correspond à un article wiki et dispose d'une évaluation en fonction de sa maturité (nombre d'étoiles décrivant le nombre d'évaluations) et de son pourcentage d'évaluation positives.

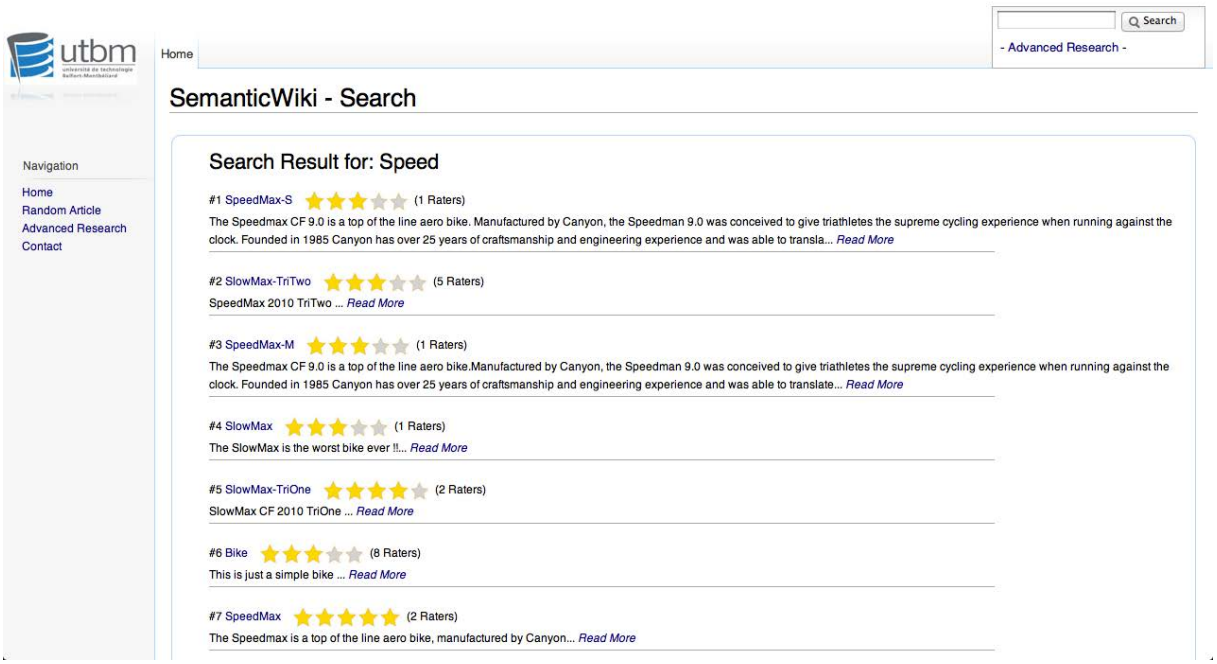


Figure 42: L'interface du Wiki

L'agent formateur permet aussi à l'acteur métier de gérer son profil en définissant son rôle, son mot de passe et les domaines de travail qui l'intéressent. L'agent formateur sauvegarde automatiquement pour chaque acteur les articles wiki qu'il a créé, les articles qu'il a consultés et les requêtes (termes ou phrases) qu'il a déjà demandées comme un historique pour filtrer les résultats de ses prochaines requêtes. Ceci permet de limiter les résultats de recherche mais aussi de rendre les articles les plus riches en informations pour lui.

6.3.4. Evaluation et/ou modification des connaissances

Nous avons expliqué dans la section précédente que l'agent présente les résultats de la recherche sous forme d'articles wiki classés selon deux méthodes. L'acteur métier peut toutefois naviguer dans les articles wiki. Lorsque l'acteur clique sur un article l'agent « rôle » le dirige vers une autre page en affichant les connaissances de cet article. Notre wiki permet à l'acteur deux choix : soit évaluer la page, soit la modifier.

La modification dans notre wiki correspond aux changements des valeurs des connaissances présentes dans l'article par l'acteur métier. La Figure 43 montre les étapes de réalisation de la fonctionnalité modification. Le processus de modification commence lorsque l'agent expert de l'organisation « Définition » détecte une action de modification faite par l'acteur métier. Alors cet agent envoie une demande de modification à l'agent rôle (fonction 1). Ce dernier se charge de chercher le fichier RDF à modifier dans la mémoire

organisationnelle en fonction de son URI (Uniform ressource Identifier) qui est unique pour chaque fichier dans la mémoire (fonction 2). Quand il reçoit le fichier en question (fonction 3) il met à jour les valeurs des connaissances modifiées par l'acteur métier et sauvegarde le fichier RDF (fonction 4). Lorsque l'agent rôle rempli sa mission il envoie à l'agent formateur le fichier RDF modifié pour le présenter avec les nouvelles valeurs à l'acteur métier (fonction 5).

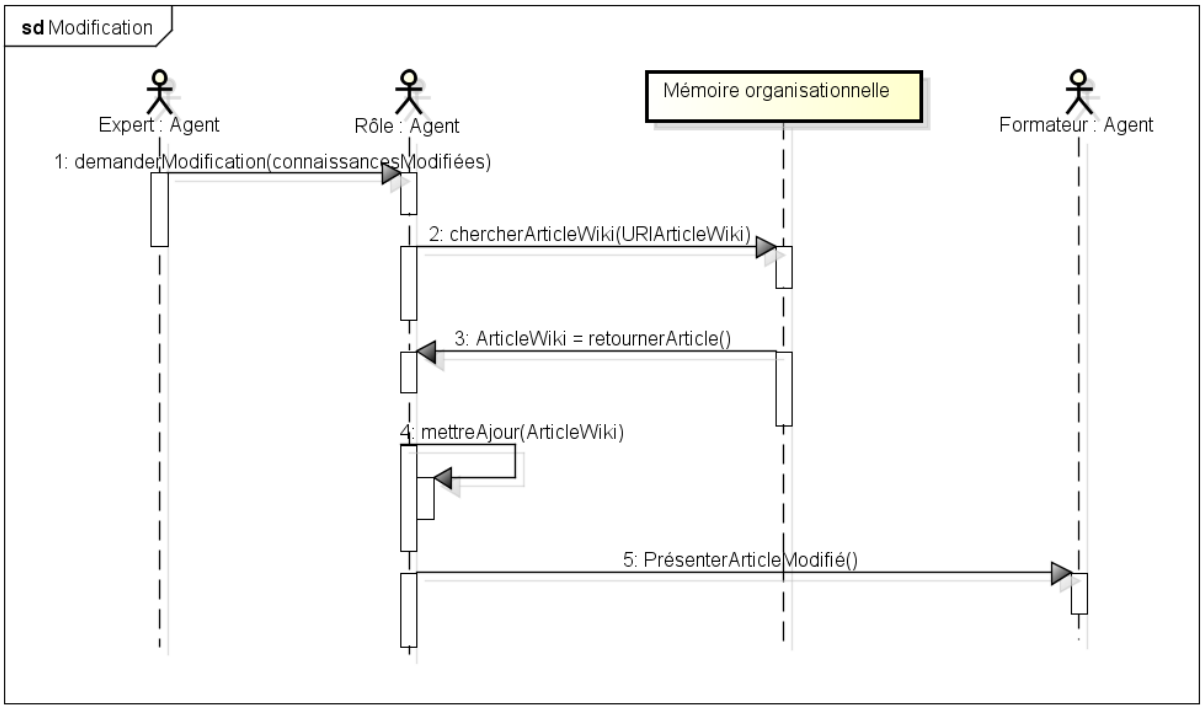


Figure 43: Diagramme de séquence de la fonctionnalité modification des connaissances

La Figure 44 présente un article wiki sur la conception d'un vélo. Cette page affiche comme le montre la figure ci-dessous les connaissances relatives à cet article avec les valeurs que l'acteur métier peut les modifier s'il possède les droits appropriés. Dans le cas où l'acteur souhaite alors modifier une ou plusieurs valeurs présentes dans cet article le processus de modification expliqués dans la Figure 43 se déclenche.

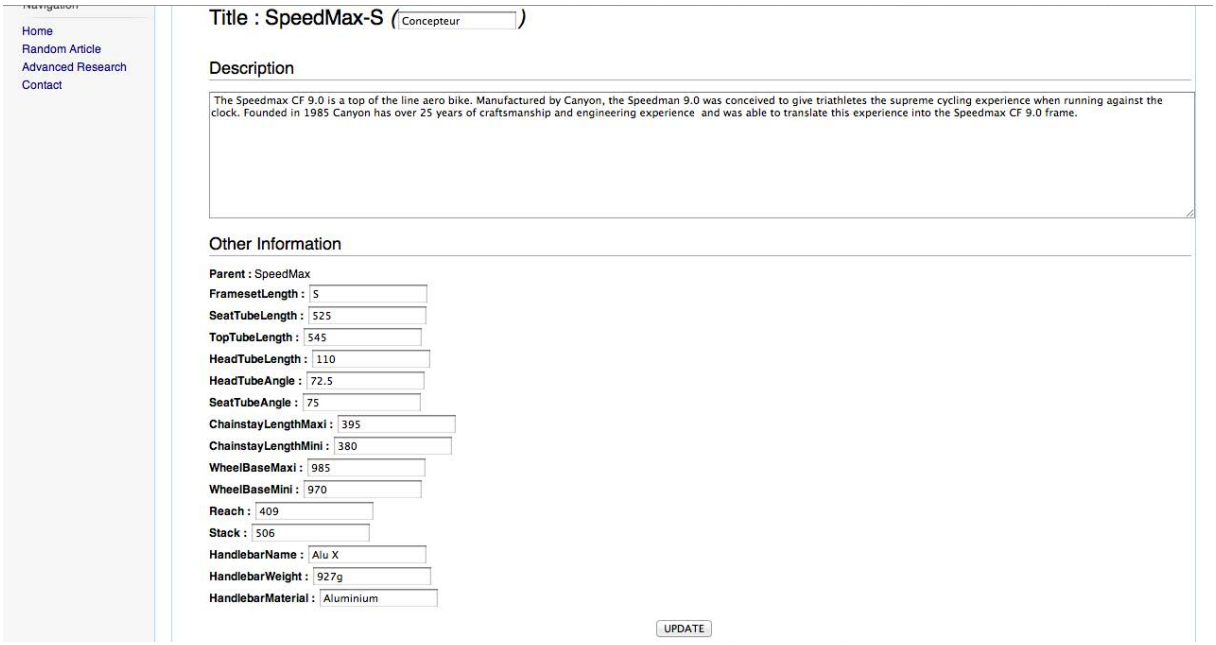


Figure 44: interface de modification des connaissances

A l'intérieur du Wiki, la connaissance est soumise à un processus d'évaluation (Figure 46). Cette évaluation est faite par les acteurs métiers. Dans ce cas, l'acteur donne une note entre 1 et 5 pour chaque connaissance. Cette note signifie que la connaissance est vue comme pertinente si elle est proche de 5 et le contraire si elle est proche de 1.

Le processus d'évaluation des connaissances, comme pour le cas de modification, se déclenche lorsque l'agent expert détecte une action d'évaluation faite par l'acteur métier. Dans ce cas l'agent expert prend la note de vote mise par l'acteur et l'envoie à l'agent rôle (fonction 1). Ce dernier recherche l'article wiki en question de la mémoire organisationnelle et le charge (fonction 2 et 3). L'agent rôle réalise alors une série de modification, premièrement il met à jour la note globale de cet article wiki en ajoutant la note mise par l'acteur métier (fonction 4). Puis il incrémente le nombre de votant sur cet article (fonction 5) et enfin il calcule l'indice de maturité de cet article en divisant la note globale mise à jour sur le nombre de votant (fonction 6). Lorsque l'agent rôle finit les modifications il communique avec l'agent formateur pour que ce dernier présente de nouveau l'article wiki en question avec son nouvel indice de maturité (fonction 7).

Dans le processus d'évaluation, nous considérons que l'acteur métier approuve un article, lorsqu'il l'assigne une évaluation positive. De plus, quand il refuse, l'article obtient une évaluation négative. Ainsi, la connaissance qui vient d'être créé a 100% d'évaluation positive.

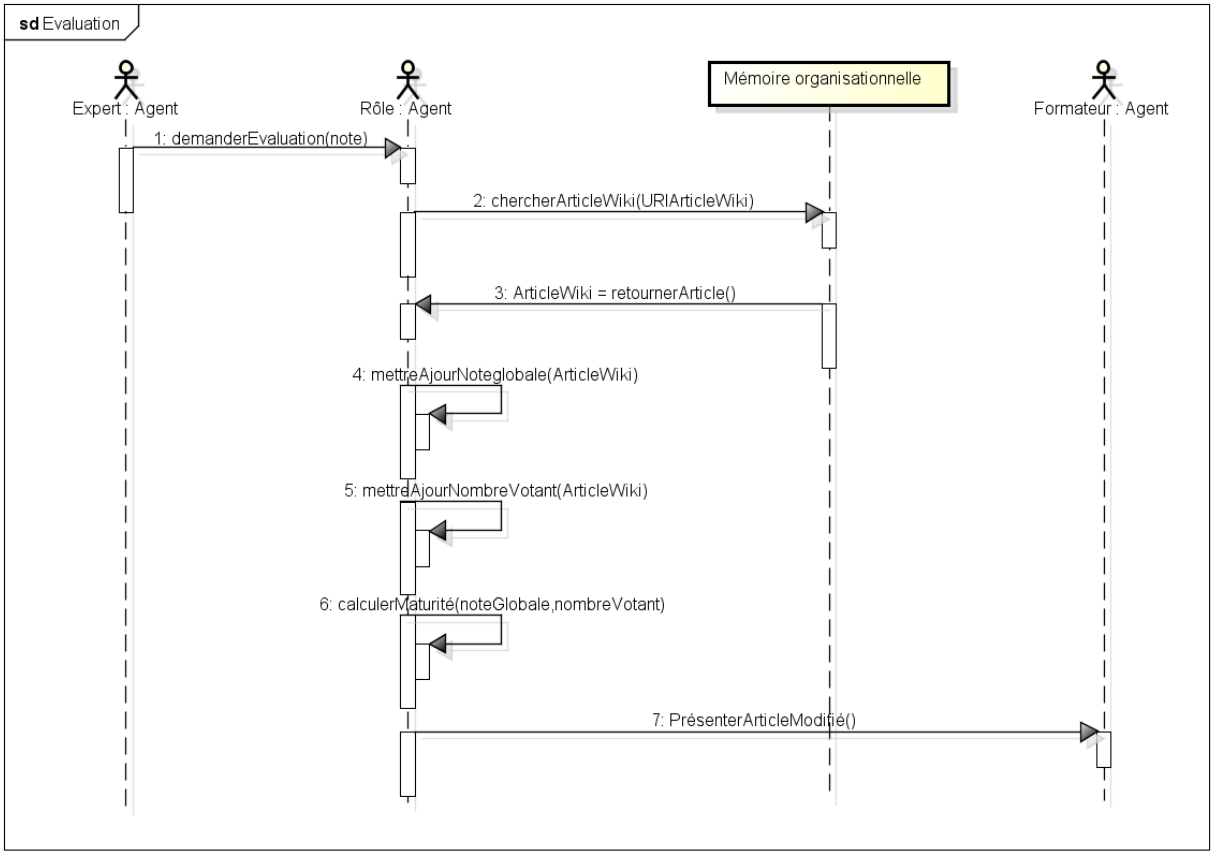


Figure 45: Diagramme de séquence de la fonctionnalité évaluation des connaissances

Au fur et à mesure des évaluations attribuées par les acteurs métier, le pourcentage (indice de maturité) peut diminuer ou augmenter. Si l'indice est inférieur à 25% d'évaluation positive, la connaissance sera considérée comme une connaissance non utile et sera proposée pour suppression à un expert.

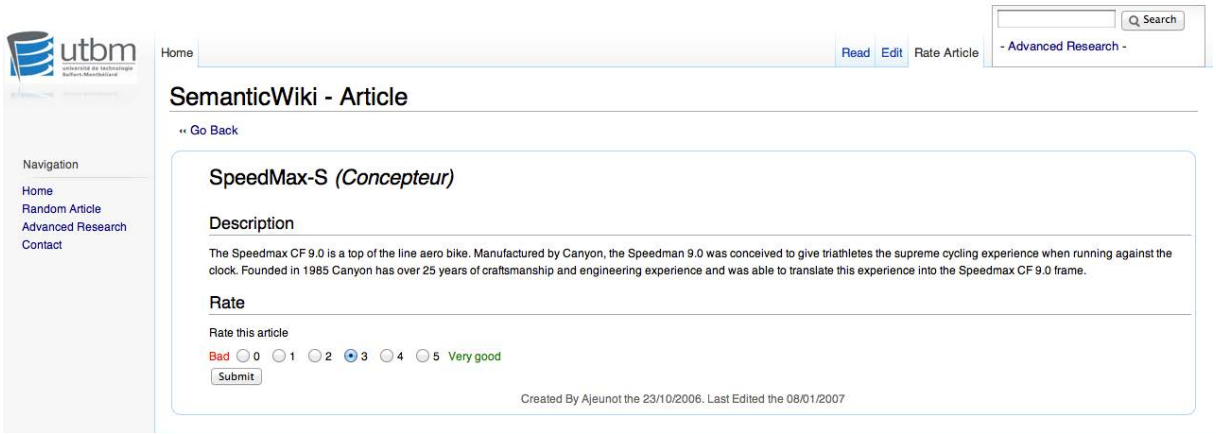


Figure 46: interface d'évaluation des connaissances

6.4.Conclusion

Dans ce chapitre, nous avons présenté le troisième et le quatrième module du cycle de vie de notre système OCEAN qui sont, respectivement, la diffusion et la réutilisation des connaissances. Nous avons utilisé un wiki sémantique basé sur un SMA pour diffuser les connaissances. L'architecture de ce wiki et les fonctionnalités qu'il assure ont été expliquées dans ce chapitre. Ce wiki permet d'évaluer, de modifier, et de partager les nouvelles connaissances.

Notre Wiki est particulièrement utile chez les groupes d'utilisateurs ou communautés qui travaillent en collaboration afin de créer de nouvelles connaissances pour des projets mécaniques et d'entretenir ces connaissances. Il offre également un moyen convivial pour construire et maintenir des vocabulaires communs aux employés de l'entreprise.

L'approche de la réutilisation des connaissances a été adoptée des travaux de Ben-Miled qui consiste à capitaliser et réutiliser des connaissances au fil de l'eau suivant une approche organisationnelle.

En effet, ce système permet d'assister l'acteur métier lors du processus de conception de produit mécanique en lui proposant principalement trois types d'aide; l'assistance automatique, le système d'alerte et le transfert des connaissances.

L'assistance automatique peut être réalisée soit par le PUSH/PULL où le système propose les connaissances utiles à la réalisation de l'activité courante soit par l'assistance automatique spécifique qui évolue au fur et à mesure que l'acteur métier établit ses requêtes de recherche des connaissances. Le système d'alerte permet d'éviter des erreurs et réaliser le produit avec moins d'efforts et en un temps plus court. Enfin, le transfert des connaissances facilite la collaboration entre les différents acteurs métiers en envoyant par mail aux acteurs partageant les mêmes connaissances celles qui sont pertinentes dans la mémoire organisationnelle.

En adoptant cette méthodologie de la réutilisation des connaissances capitalisées nous complétons le cycle de vie de la gestion de connaissances.

CHAPITRE 7

CONCLUSION GÉNÉRALE

Nous revenons dans la conclusion sur le bilan des travaux réalisés dans cette thèse. En particulier, nous dégageons une synthèse des propriétés mises en avant lors de l'approche proposée. Cette lecture nous permet alors de dégager des points d'approfondissement et d'amélioration de l'ensemble du travail présenté. Ces perspectives sont présentées dans la seconde partie de ce chapitre.

Table de matières

7.1. Bilan du travail réalisé

Nous avons abordé tout au long de cette thèse la problématique de la gestion des connaissances hétérogènes et distribuées au sein d'une entreprise étendue. Ce problème a été identifié dans le domaine industriel dans lequel nous travaillons. Pour concevoir et développer un produit les acteurs métiers utilisent leurs connaissances propres et des outils métiers différents. Les connaissances peuvent donc provenir de plusieurs sources et peuvent être distribuées à travers le réseau de l'entreprise étendue.

Dans cette thèse, nous avons conçu et développé un système pour résoudre ce problème en nous basant sur les ontologies et les agents. Ce système s'appelle OCEAN (Ontology Creator, Extractor, and Annotator kNowledge). Ce système assure les quatre étapes du cycle de vie de la gestion de connaissance que nous avons présentées dans le chapitre 3 qui sont : la définition, l'extraction, la validation et la réutilisation des connaissances.

Nous avons présenté dans le chapitre 2 les trois domaines qui constituent la thèse : la gestion des connaissances, les ontologies et les systèmes multi-agents. Particulièrement, nous avons exploré en profondeur la gestion des connaissances multi sources, sa définition, ses avantages pour les employés, pour les entreprises, pour la communauté scientifique, et les cycles de vie existants associés.

Nous nous sommes ensuite consacrés à la définition du système multi-agents pour la gestion des connaissances. Le système multi-agents a prouvé son utilité dans la gestion des systèmes complexes. Cette nécessité résulte du fait que le SMA est composé de plusieurs agents qui fonctionnent en même temps et qui partagent le même environnement. Ces agents grâce à leur autonomie, leur coopération, et la communication entre eux, sont capables de décomposer un problème complexe en plusieurs sous problèmes afin de les résoudre plus facilement et rapidement. Puis nous avons présenté la relation entre les systèmes multi agents et les ontologies puisque ces derniers fournissent aux agents les connaissances nécessaires pour qu'ils exercent leurs tâches. Enfin, nous avons terminé ce chapitre par une présentation de quelques approches agents pour la gestion des connaissances. Nous avons comparés ces approches avec celle que nous avons proposé en fonction des étapes du cycle de vie de la gestion de connaissance qu'elles assurent.

Dans le chapitre 3 nous avons introduit notre approche générale pour gérer les connaissances hétérogènes et distribuées (OCEAN). L'objectif d'OCEAN est de permettre la capitalisation et la réutilisation des connaissances multi-sources. Ce système repose sur un cycle de vie de quatre étapes. Notre système assure la définition, l'extraction, la validation et la réutilisation des connaissances. Chacune de ces étapes est gérée par une organisation composée d'agents garantissant son accomplissement. Les trois premières étapes ont été respectivement l'objet des chapitres 4, 5 et 6. Dans cette thèse nous n'avons pas abordé en détail la quatrième étape qui la réutilisation des connaissances puisque nous avons adopté les travaux de Ben-Miled (Ben Miled, 2011) qui traitent cette problématique.

Dans le chapitre 4 nous avons détaillé la première étape du cycle de vie de la gestion des connaissances qui est la définition des connaissances. Dans cette thèse nous avons utilisé les ontologies pour définir et représenter les connaissances afin de les capitaliser. Mais la définition des connaissances est une tâche difficile même pour les experts du domaine. Dans le but d'aider ces experts à identifier les connaissances dont ils ont besoin, nous avons présenté une méthodologie pour aider les experts dans la création de leurs ontologies. Cette méthodologie repose sur trois agents qui sont l'agent GCOntologie, l'agent RechercheSimilitude, et l'agent Expert.

L'agent GCOntologie permet de créer l'ontologie définie par l'expert métier. Cette ontologie est enrichie par l'agent RechercheSimilitude en ajoutant les concepts de synonymies, de méronymies, d'hyponymies et d'hyperonymies à partir de la base lexicale WordNet. Puis cet agent utilise les concepts de l'ontologie créée pour rechercher la similarité avec les données présentes dans la base de l'application métier et retourne une ontologie adaptée. Cette ontologie correspond à l'ontologie originale créée par l'expert adaptée avec les résultats de l'assistance qui sont guidés la base de données de l'application métier. Cette ontologie adaptée est proposée à l'expert métier par l'agent Expert afin de l'accepter ou de la refuser. Les ontologies créées par les experts métier sont stockées dans des fichiers de format OWL dans une base des connaissances conceptuelles par l'agent GCOntologie.

Nous avons ensuite proposé dans le chapitre 5 notre méthodologie pour l'extraction des connaissances de plusieurs applications métiers. Pour illustrer cette méthodologie, nous avons présenté dans ce chapitre deux études de cas. La première consistait à extraire d'une base de données de type SQL Server et la deuxième d'un fichier Excel. Nous avons utilisé des services web pour interagir de manière distribuée.

Dans les deux cas, la transformation de l'ontologie en langage de requête est réalisée par l'agent Transformer. L'exécution de ces requêtes sur la base correspondante est assurée par l'agent Interroger. Pour donner du sens aux données, résultats de cette interrogation, nous les avons contextualisé c'est-à-dire les annoter. Ceci permet leur réutilisation. Cette annotation garantit le passage des données à ce qu'on appelle des informations. Nous avons sauvegardés ces informations dans des fichiers de type RDF. L'ensemble des fichiers RDF construit notre mémoire organisationnelle. Les fonctions d'annotation et de sauvegarde sont assurées par l'agent Annoter.

Dans le chapitre 6 nous avons présenté la troisième étape du cycle de vie de la gestion des connaissances qui est la validation. Dans ce chapitre nous avons proposé un wiki sémantique basé sur la mémoire organisationnelle présenté dans le chapitre 5. Les informations présentes dans les fichiers RDF, qui composent la mémoire organisationnelle, deviennent des connaissances après leur premier passage dans le wiki et la validation d'un acteur métier.

Notre wiki sémantique offre cinq fonctionnalités, qui sont : la recherche sémantique, la classification des résultats, la présentation, la modification et l'évaluation de connaissances, assurées par trois agents (agent formateur, agent organisateur et agent rôle).

L'agent formateur récupère la requête lancer par l'acteur métier et réalise une recherche dans la mémoire organisationnelle. Les résultats de la recherche sont des articles wiki. Ces derniers sont classifiés, par l'agent organisateur, selon le profil de l'acteur métier et l'indice de maturité de chaque article. Les articles classifiés sont présentés à l'acteur métier par l'agent formateur. Ce dernier permet aussi aux acteurs de gérer leur profil. En plus notre wiki offre aux acteurs la possibilité de modifier les valeurs d'un article wiki ou l'évaluer en lui donnant une note entre 1 et 5 (5 étant la plus importante). Un article ayant une note globale inférieure à 25% d'évaluation positif est présenté à un expert afin de le supprimer de la mémoire.

Toutefois, le cadre global que nous cherchions à construire avec ce travail n'est pas encore complet : des approfondissements restent à faire. La section suivante détaille ces perspectives.

7.2.Perspectives et approfondissements

Nous définissons un axe d'approfondissement à long terme pour la suite de ce travail.

Dans cet axe, nous intéressons à deux points. Le premier consiste à améliorer notre système multi-agents, et la deuxième à fusionner plusieurs ontologies existantes.

7.2.1. Enrichir les fonctionnalités du système multi-agents

Le système OCEAN que nous avons présenté dans cette thèse permet de définir, d'extraire, de valider et de réutiliser les connaissances en se basant sur un système multi-agents. Nous avons expliqué les différentes organisations d'agents que nous avons identifiés dans le chapitre 3. Mais nos agents travaillent successivement c'est-à-dire quand le premier agent, déclenché par le système, accomplit sa mission il déclenche l'agent suivant et ainsi de suite.

Nos prochains travaux se concentrent alors sur l'amélioration de ce système d'agents en le rendant plus actif.

La première amélioration consiste à ce que les agents, même si l'acteur métier n'a pas demandé d'informations, puissent proposer à ces acteurs des informations qui leur semblent pertinentes.

La deuxième amélioration, qui est plus importante, consiste à enrichir l'approche organisationnelle de notre système multi-agents de façon à ajouter des nouveaux concepts au modèle utilisé. L'ajout de mécanismes de raisonnement, de règles ou de mécanismes d'apprentissage peut être pertinent dans bien des situations de conception.

7.2.2. Fusion des ontologies.

La première fonctionnalité de notre système OCEAN consiste à définir les connaissances par un expert métier dans une ontologie. Chaque ontologie correspond à une application métier.

La fonctionnalité telle qu'elle est maintenant, est un peu limitée puisque l'expert métier ne peut pas fusionner plusieurs ontologies déjà créées. Nous allons travailler prochainement à développer cette possibilité. Ceci est utile si l'expert métier souhaite bénéficier des connaissances définies dans des ontologies déjà créées par le même expert ou d'autres expert métier et permet, potentiellement, d'éviter les redondances dans les ontologies.

BIBLIOGRAPHIE

Abar, S., Abe, T., Kinoshita, T., 2004. A next generation knowledge management system architecture, in: 18th International Conference on Advanced Information Networking and Applications, 2004. AINA 2004. Presented at the 18th International Conference on Advanced Information Networking and Applications, 2004. AINA 2004, pp. 191 – 195 Vol.2.

Abderraouf, B., 2012. Contribution à la mise-en-œuvre d'un moteur d'exécution de modèles UML pour la simulation d'applications temporisées et concurrentes. Supélec.

Abecker, A., Bernardi, A., Elst, L. van, 2003. Agent Technology For Distributed Organizational Memories: The Frodo Project.

Abecker, A., Bernardi, A., Hinkelmann, K., Kuhn, O., Sintek, M., 1998. Toward a technology for organizational memories. IEEE Intelligent Systems and their Applications 13, 40 – 48.

Agarwal, D., Wiedmer, A., Faybishenko, B., Hunt, J., Kushner, G., Romosan, A., Shoshani, A., Whiteside, T., 2012. A Methodology for Management of Heterogeneous Site Characterization and Modeling Data, in: The XIX International Conference on Computational Methods in Water Resources (CMWR 2012). Urbana-Champaign, IL.

Alavi, M., Leidner, D.E., 2001. Review: Knowledge management and knowledge management systems: conceptual foundations and research issues. MIS Q. 25, 107–136.

Allee, V., 1997. 12 Principles of Knowledge Management. Training and Development 51, 71–74.

Alonso, G., Casati, F., Kuno, H., Machiraju, V., 2010. Web Services: Concepts, Architectures and Applications, 1st ed. Springer Publishing Company, Incorporated.

Astrova, I., Korda, N., Kalja, A., 2007. Storing OWL Ontologies in SQL Relational Databases, in: World Academy of Science, Engineering and Technology 29.

Auer, S., Dietzold, S., Riechert, T., 2006. OntoWiki–A tool for social, semantic collaboration, in: The Semantic Web-ISWC 2006. Springer, pp. 736–749.

Aumüller, D., 2005. SHAWN: Structure helps a wiki navigate, in: Proceedings of the BTW-Workshop WebDB Meets IR. Karlsruhe, Germany.

Aumüller, D., Auer, S., 2005. Towards a semantic wiki experience–desktop integration and interactivity in WikSAR, in: Semantic Desktop Workshop. Galway, Ireland, pp. 2005–2012.

Authosserre-Cavarero, A., Bertrand, F., Fornarino, M.B.-, Collet, P., Dubois, H., Ducasse, S., Dupuy-Chessa, S., Faron-Zucker, C., Faucher, C., Lafaye, J.-Y., Lahire, P., Goaer, O.L., Montagnat, J., Pinna-Dery, A.-M., 2012. Interopérabilité des systèmes d'information : approches dirigées par les modèles. Presented at the Inforsid.

Bachimont, B., 2000. Engagement sémantique et engagement ontologique: conception et réalisation d'ontologies en ingénierie des connaissances. Ingénierie des connaissances: évolutions récentes et nouveaux défis 305–323.

Baez, G.E.V., 2005. Développement d'Applications à Grande Echelle par Composition de Méta-Modèles. Université Joseph-Fourier - Grenoble I.

Bellifemine, F., Rimassa, G., 2001. Developing multi-agent systems with a FIPA-compliant agent framework. Softw. Pract. Exper. 31, 103–128.

Ben Miled, A., 2011. Vers un système de réutilisation des connaissances en ingénierie de conception (Thèse de doctorat). Université de Technologie de Belfort-Montbeliard.

Ben Miled, A., Hilaire, V., Monticolo, D., Koukam, A., 2008. Reusing Knowledge by Multi Agent System and Ontology, in: IEEE International Conference on Signal Image Technology and Internet Based Systems, 2008. SITIS '08. Presented at the IEEE International Conference on Signal Image Technology and Internet Based Systems, 2008. SITIS '08, pp. 678–683.

Ben Miled, A., Monticolo, D., Hilaire, V., Koukam, A., 2009. A multi-agent based approach for knowledge transfer in product design contexts, in: International Conference on Computers Industrial Engineering, 2009. CIE 2009. Presented at the International Conference on Computers Industrial Engineering, 2009. CIE 2009, pp. 1353–1358.

Bergenti, F., Poggi, A., Rimassa, G., Turci, P., 2002. CoMMA: a multi-agent system for corporate memory management, in: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems: Part 3, AAMAS '02. ACM, New York, NY, USA, pp. 1039–1040.

Bhat, S., Wahid, A., 2012. Log Agent for Knowledge Management Based on Multi Agent System. ijetae 2, 296–299.

Boissier, O., Bordini, R.H., Hübner, J.F., Ricci, A., Santi, A., 2013. JaCaMo Project [WWW Document]. URL http://jacamo.sourceforge.net

Bonifacio, M., Bouquet, P., Traverso, P., 2002. Enabling Distributed Knowledge Management: Managerial and Technological Implications.

Bresciani, P., Perini, A., Giorgini, P., Giunchiglia, F., Mylopoulos, J., 2004. Tropos: An Agent-Oriented Software Development Methodology. Autonomous Agents and Multi-Agent Systems 8, 203–236.

Brigui-Chtioui, I., Saad, I., 2010. A Multiagent Approach for Collective Decision Making in Knowledge Management. Group Decision and Negotiation 20, 19–37.

Buffa, M., Gandon, F., Ereteo, G., Sander, P., Faron, C., 2008. SweetWiki: A semantic wiki. Web Semantics: Science, Services and Agents on the World Wide Web 6, 84–97.

Campanini, S.E., Castagna, P., Tazzoli, R., 2004. Platypus wiki: a semantic wiki wiki web. Semantic Web Applications and Perspectives, Proceedings of 1st Italian Semantic Web Workshop.

Carman, P., Tigwell, P., 1998. Catia Reference Guide, 2nd Revised edition. ed. OnWord Press,U.S.

Caron, P.-A., 2007. Ingénierie dirigée par les modèles pour la construction de dispositifs pédagogiques sur des plateformes de formation. Université des Sciences et Technologie de Lille - Lille I.

Champin, P., Prié, Y., Mille, A., 2003. Musette: Modeling USEs and Tasks for Tracing Experience, in: Workshop 5 "From Structured Cases to Unstructured Problem Solving Episodes For Experience-Based Assistance", ICCBR'03. Trondheim, pp. 279–286.

Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J., Widom, J., 1994. The TSIMMIS Project: Integration of Heterogeneous Information Sources. Presented at the 16th Meeting of the Information Processing Society of Japan.

Chen, A., Liu, L., Shang, J., 2012. A Hybrid Strategy to Construct Scientific Instrument Ontology from Relational Database Model, in: 2012 International Conference on Computer Distributed Control and Intelligent Environmental Monitoring (CDCIEM). Presented at the 2012 International Conference on Computer Distributed Control and Intelligent Environmental Monitoring (CDCIEM), pp. 25–33.

Chiprianov, V., 2012. Collaborative Construction of Telecommunications Services. An Enterprise Architecture and Model Driven Engineering Method. Télécom Bretagne, Université de Bretagne-Sud.

Chow, K.H., Choy, K.L., Lee, W.B., 2006. On the design of a real-time knowledge-based system for managing logistics operations. Intelligent Systems in Accounting, Finance & Management 14, 3–25.

Corby, O., Dieng-Kuntz, R., Gandon, F., Faron-Zucker, C., 2006. Searching the semantic Web: approximate query processing based on ontologies. IEEE Intelligent Systems 21, 20–27.

Cossentino, M., Gaud, N., Hilaire, V., Galland, S., Koukam, A., 2010. ASPECS: an agent-oriented software process for engineering complex systems. Autonomous Agents and Multi-Agent Systems 20, 260–304.

Dalkir, K., 2005. Knowledge Management In Theory and Practice. Butterworth-Heinemann.

Davenport, T.H., Long, D.W.D., Beers, M.C., 1998. Successful Knowledge Management Projects. Sloan management review 39, 43–57.

De Azevedo, H.J.S., Barthès, J.-P., 1997. Contribution à la modélisation des connaissances à l'aide des systèmes multi-agents (Thèse). Université de Compiègne, Compiègne.

Decker, B., Ras, E., Rech, J., Klein, B., Hoecht, C., 2005. Self-organized reuse of software engineering knowledge supported by semantic wikis, in: Proceedings of the Workshop on Semantic Web Enabled Software Engineering (SWESE). Galway, Ireland.

Dello, K., Simperl, E.P.B., Tolksdorf, R., 2006. Creating and using semantic web information with makna, in: First Workshop on Semantic Wikis, From Wiki to Semantics. Budva, Montenegro, p. 1.

Dhombres, F., Vandenbussche, P.-Y., Rath, A., Hanaeur, M., Olry, A., Urbero, B., Choquet, R., Charlet, J., 2012. Projet OrphaOnto - Première étape de l'ontologisation des bases de connaissances d'Orphanet, in: Actes de IC2011. Chambéry, France, pp. 573–588.

Dieng, R., Corby, O., Gandon, F., Giboin, A., Collectif, 2005. Knowledge management : Méthodes et outils pour la gestion des connaissances, 3e édition. ed. Dunod.

DIENG, R., CORBY, O., GIBOIN, A., RIBIÈRE, M., 1999. Methods and tools for corporate knowledge management. International Journal of Human-Computer Studies 51, 567–598.

Dieng-Kuntz, R., Matta, N., 2002. Knowledge Management and Organizational Memories. Springer.

Durfee, E.H., Lesser, V.R., Corkill, D.D., 1989. Trends in cooperative distributed problem solving. IEEE Transactions on Knowledge and Data Engineering 1, 63 –83.

Ermine, J.-L., 2010. Knowledge Crash and Knowledge Management. International Journal of Knowledge and Systems Science 1, 79–95.

Fabkam, C., Bellatreche, L., Dehainsala, H., Ait Ameur, Y., Pierra, G., 2009. SISRO, conception de bases de données à partir d'ontologies de domaine. Techniques et sciences informatiques 28, 1233–1261.

Ferber, J., 1995. Les Systèmes multi-agents: Vers une intelligence collective. Dunod.

FIPA ACL, 2002. FIPA ACL Message Structure Specification (Standard No. SC00037J), Foundation For intelligent Physical Agents.

Fu, R., Xin, Z., 2008. Research on Electronic Commerce KMS Based on Agent and Ontology, in: Proceedings of the First International Workshop on Knowledge Discovery and Data Mining, WKDD '08. IEEE Computer Society, Washington, DC, USA, pp. 190–195.

FU, R., YUE, X., SONG, M., XIN, Z., 2008. An architecture of knowledge management system based on agent and ontology. The Journal of China Universities of Posts and Telecommunications 15, 126–130.

Gandon, F., 2002. Distributed Artificial Intelligence and Knowledge Management: Ontologies and Multi-agent Systems for a Corporate Semantic Web (Thèse).

Gandon, F., Berthelot, L., Dieng-Kuntz, R., 2002. A multi-agent platform for a corporate semantic web, in: Proceedings of the First International Joint Conference on

Autonomous Agents and Multiagent Systems: Part 3, AAMAS '02. ACM, New York, NY, USA, pp. 1025–1032.

Gandon, F., G, F., Dieng-Kuntz, R., 2001. Ontologie pour un système multi-agents dédié à une mémoire d'entreprise, in: Ingénierie Des Connaissances. Grenoble, pp. 1–20.

Gaud, N.A., 2007. Systèmes multi--agents holoniques: de l'analyse à l'implantation (Thèse de doctorat). Université de technologie de Belfort-Montbéliard, France.

Giannakis, M., Louis, M., 2011. A multi-agent based framework for supply chain risk management. Journal of Purchasing and Supply Management 17, 23–31.

Grimes, S., 2010. Breakthrough analysis: two+ nine types of semantic search. InformationWeek.

Grundstein, M., 2004. De la capitalisation des connaissances au management des connaissances dans l'entreprise, in: Management Des Connaissances En Entreprise. Lavoisier, Paris, pp. 25–54.

Grundstein, M., Barthès, J.P., 1996. An industrial view of the process of capitalizing knowledge. Knowledge Management: Organization, Competence and Methodology.

Guha, R., McCool, R., Miller, E., 2003. Semantic search, in: Proceedings of the 12th International Conference on World Wide Web, WWW '03. ACM, New York, NY, USA, pp. 700–709.

Hanselman, D.C., Littlefield, B.L., 2004. Mastering MATLAB 7, 1st ed. Prentice Hall.

Jan, M., Jouvray, C., Kordon, F., Kung, A., Lalande, J., Loiret, F., Navas, J., Pautet, L., Pulou, J., Radermacher, A., Seinturier, L., 2012. Flex-eWare: a flexible model driven solution for designing and implementing embedded distributed systems. Software: Practice and Experience 42, 1467–1494.

Jaspers, R., Uschold, M., Kitzmiller, T., 1999. ntegrated Framework for Knowledge Management. Matta Reimer, Electronic, Proceedings of IJCAI'99 Workshop on Knowledge Management and Organizational Memories.

Jee, K., Yang, J.-J., 2006. Knowledge Description Model for MAS Utilizing Distributed Ontology Repositories, in: Shi, Z.-Z., Sadananda, R. (Eds.), Agent Computing and Multi-Agent Systems, Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 773–780.

Jennex, M.E., 2007. Knowledge Management in Modern Organizations. Idea Group Inc (IGI).

Jennex, M.E., Smolnik, S., Croasdell, D., 2007. Towards Defining Knowledge Management Success, in: 40th Annual Hawaii International Conference on System Sciences, 2007. HICSS 2007. Presented at the 40th Annual Hawaii International Conference on System Sciences, 2007. HICSS 2007, p. 193c.

Jézéquel, J.-M., Combemale, B., Vojtisek, D., 2012. Ingénierie Dirigée par les Modèles : des concepts a la pratique. Ellipses Marketing.

Johannessen, J.-A., Olaisen, J., Olsen, B., 2001. Mismanagement of tacit knowledge: the importance of tacit knowledge, the danger of information technology, and what to do about it. International Journal of Information Management 21, 3–20.

Kleppe, A., Warmer, J., Bast, W., 2003. MDA Explained: The Model Driven Architecture?: Practice and Promise, 1st ed. Addison Wesley.

Krötzsch, M., Vrande\vcić, D., Völkel, M., Haller, H., Studer, R., 2007. Semantic wikipedia. Web Semantics: Science, Services and Agents on the World Wide Web 5, 251–261.

Labidi, S., Lejouad, W., 1993. De l'intelligence artificielle distribuee aux systemes multi-agents. Rapport de recherche n°2004, INRIA.

Lahoud, I., Monticolo, D., Gomes, S., 2010. OCEAN: A Semantic Web Service to Extract Knowledge in E-Groupwares, in: 2010 Sixth International Conference on Signal-Image Technology and Internet-Based Systems (SITIS). Presented at the 2010 Sixth International Conference on Signal-Image Technology and Internet-Based Systems (SITIS), pp. 354 –362.

Lalit Narayan, K., Rao Mallikarjuna, K., Sarcar, M.M.., 2008. Computer Aided Design And Manufacturing. PHI Learning Pvt. Ltd.

Lans, R.F. van der, 2006. Introduction to SQL: Mastering the Relational Database Language, 4th ed. Addison-Wesley Professional.

Leclercq, E., Savonnet, M., Naubourg, P., 2012. Traitement des variabilités métier dans les Systèmes d'Information biologiques, in: XXXeme Congrès INFORSID. Montpellier, France, pp. 173–188.

Lin, L.F., Zhang, W.Y., Lou, Y.C., Chu, C.Y., Cai, M., 2011. Developing manufacturing ontologies for knowledge reuse in distributed manufacturing environment. International Journal of Production Research 49, 343–359.

Maier, R., 2002. Knowledge Management Systems: Information and Communication Technologies for Knowledge Management. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Malhotra, Y., Galletta, D., 2003. A Multidimensional Commitment Model of Volitional Systems Adoption and Usage Behavior. J. Manage. Inf. Syst. 22, 117–151.

McBride, B., 2002. Jena: a semantic Web toolkit. IEEE Internet Computing 6, 55–59.

McGuinness, D.L., Van Harmelen, F., 2004. OWL web ontology language overview. W3C recommendation 10, 10.

Monticolo, D., 2008. Une approche organisationnelle pour la conception d'un système de gestion des connaissances fondé sur le paradigme agent. (Thèse de doctorat). Université de Technologie de Belfort-Montbeliard.

Monticolo, D., Gomes, S., 2011. WikiDesign:Collaborative Knowledge Evaluation with a Semantic Wiki. International Journal of e-Collaboration 7, 31–42.

Monticolo, D., Hilaire, V., Gomes, S., Koukam, A., 2007a. An approach to manage Knowledge based on multi-agents System using a Ontology, in: 19th International Conference on System Research, Informatics & Cybernetics (InterSymp 2007). Presented at the Symposium on Representation of Context in Software, Baden-Baden, p. 11p.

Monticolo, D., Hilaire, V., Koukam, A., Gomes, S., 2007b. A multi agent model to support the knowledge management process inside professional activities, in: 2nd International Conference on Digital Information Management, 2007. ICDIM '07. Presented at the 2nd International Conference on Digital Information Management, 2007. ICDIM '07, pp. 799 –804.

Monticolo, D., Hilaire, V., Koukam, A., Gomes, S., 2009. KATRAS : un système multi-agents pour la gestion des connaissances lors des projets de conception mécanique, in: Journées Francophones Sur Les Systèmes Multi-Agents, JFSMA. Lyon, France, pp. 219–230.

MOTTA, E., BUCKINGHAM SHUM, S., DOMINGUE, J., 2000. Ontology-driven document enrichment: principles, tools and applications. International Journal of Human-Computer Studies 52, 1071–1109.

Muller, P.-A., 2005. Model Transformation, in: Model Driven Engineering for Distributed Real-time Embedded Systems. S. Gérard, J.- P. Babau, J. Champeau.

Nageba, E., Rubel, P., Fayn, J., 2013. Towards an intelligent exploitation of heterogeneous and distributed resources in cooperative environments of eHealth. IRBM 34, 79–85.

Nakra, P., 2000. Knowledge management: The magic is in the culture! Competitive Intelligence Review 11, 53–60.

Nascimento, F.A.M. do, Oliveira, M.F.S., Wagner, F.R., 2012. A model-driven engineering framework for embedded systems design. Innovations Syst Softw Eng 8, 19–33.

Nelson, R.R., Winter, S.G., 1990. Evolutionary Theory of Economic Change, Reprint. ed. The Belknap Press.

Noy, N., mcguinness, D., 2001. Ontology Development 101: A Guide to Creating Your First Ontology. Stanford University Knowledge Systems Laboratory Technical Report KSL-01-05.

Philpotts, M., 1996. An introduction to the concepts, benefits and terminology of product data management. Industrial Management & Data Systems 96, 11–17.

Pillet, M., Martin-Bonnefous, C., Bonnefous, P., Courtois, A., 2011. Gestion de production : Les fondamentaux et les bonnes pratiques, 5e édition. ed. Editions d'Organisation.

Pirró, G., Mastroianni, C., Talia, D., 2010. A framework for distributed knowledge management: Design and implementation. Future Generation Computer Systems 26, 38–49.

Pomian, J., 1996. Mémoire d'entreprise: Techniques et outils de la gestion du savoir. Sapientia.

Preece, A., Hui, K., Gray, A., Marti, P., Bench-Capon, T., Jones, D., Cui, Z., 2000. The KRAFT architecture for knowledge fusion and transformation. Knowledge-Based Systems 13, 113–120.

Prud'hommeaux, E., Seaborne, A., 2008. SPARQL Query Language for RDF. W3C Recommendation, January 2008.

Prud'Hommeaux, E., Seaborne, A., 2008. SPARQL query language for RDF. W3C recommendation 15.

Qiu, X., Yue, J., 2010. Ontology based distributed agricultural knowledge management, in: 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD). Presented at the 2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 2858 –2861.

Rahman, S.A., Yadav, D., Agerwal, P., Bisth, P.S., 2012. Multiagent Knowledge Management Architecture. Journal of Software Engineering and Application 5, 33–40.

Rajagopal, H., 2005. JENA: A Java API for Ontology Management. IBM Corporation, Colorado Software Summit.

René Peinl, R.M., 2011. SimKnowledge – Analyzing Impact of Knowledge Management Measures on Team Organizations with Multi Agent-Based Simulation. Information System Frontiers 13, 621–636.

Richards, D., 2009. A social software/Web 2.0 approach to collaborative knowledge engineering. Inf. Sci. 179, 2515–2523.

Roser, S., Bauer, B., 2006. Ontology-Based Model Transformation, in: Bruel, J.-M. (Ed.), Satellite Events at the MoDELS 2005 Conference, Lecture Notes in Computer Science. Springer Berlin / Heidelberg, pp. 355–356.

Saaksvuori, A., Immonen, A., 2008. Product Lifecycle Management. Springer.

Sajja, P.S., 2008. Multi-Agent System for Knowledge-Based Access to Distributed Databases. Interdisciplinary Journal of Information, Knowledge and Management, Vol. 3.

Santoso, H.A., Haw, S.-C., Abdul-Mehdi, Z.T., 2011. Ontology extraction from relational database: Concept hierarchy as background knowledge. Knowledge-Based Systems 24, 457–464.

Schaffert, S., 2006. IkeWiki: A Semantic Wiki for Collaborative Knowledge Management, in: 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2006. WETICE '06. Presented at the 15th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2006. WETICE '06, pp. 388–396.

Souza, E.F., Oliveira, L.E., Falbo, R.A., Vijaykumar, L., 2012. Using Ontologies to Build a Database to Obtain Strategic Information in Decision Making., in: Malucelli, A., Bax, M. (Eds.), ONTOBRAS-MOST, CEUR Workshop Proceedings. CEUR-WS.org, pp. 200–205.

Stark, J., 2011. Product Lifecycle Management, in: Product Lifecycle Management, Decision Engineering. Springer London, pp. 1–16.

Stark, M.M., Riesenfeld, R.F., 1998. WordNet: An Electronic Lexical Database, in: Proceedings of 11th Eurographics Workshop on Rendering. MIT Press.

Stegmaier, F., Döller, M., Kosch, H., Hutter, A., Riegel, T., 2013. AIR: Architecture for Interoperable Retrieval on Distributed and Heterogeneous Multimedia Repositories, in: Adami, N., Cavallaro, A., Leonardi, R., Migliorati, P. (Eds.), Analysis, Retrieval and Delivery of Multimedia Content, Lecture Notes in Electrical Engineering. Springer New York, pp. 149–163.

Stevens, R.D., Baker, P.G., Bechhofer, S., Ng, G., Jacoby, A., Paton, N., Goble, C.A., Brass, A., 2000. TAMBIS: transparent access to multiple bioinformatics information sources. Bioinformatics 16, 184–186.

Tiedeken, J., Herbst, J., Reichert, M., 2013. Managing Complex Data for Electrical/Electronic Components: Challenges and Requirements. Presented at the BTW'13 Workshophs, 15th Conf on Database Systems, Technology, and Web (BTW'13), Koellen-Verlag, Magdeburg, pp. 141–150.

Toledo, C.M., Bordini, R.H., Chiotti, O., Galli, M.R., 2012. Developing a Knowledge Management Multi-Agent System Using JaCaMo, in: Dennis, L., Boissier, O., Bordini, R.H. (Eds.), Programming Multi-Agent Systems. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 41–57.

Tony Liu, D., William Xu, X., 2001. A review of web-based product data management systems. Computers in Industry 44, 251–262.

Tran, S., 2012. Le pilotage des entreprises étendues : le rôle du SI dans le dispositif de gestion. Finance Contrôle Stratégie 15, 43–63.

Trinkunas, J., Vasilecas, O., 2007. Building ontologies from relational databases using reverse engineering methods, in: Proceedings of the 2007 International Conference on Computer Systems and Technologies, CompSysTech '07. ACM, New York, NY, USA, pp. 13:1–13:6.

Umble, E.J., Haft, R.R., Umble, M.M., 2003. Enterprise resource planning: Implementation procedures and critical success factors. European Journal of Operational Research 146, 241–257.

Uschold, M., Gruninger, M., 1996. Ontologies: principles, methods and applications. The Knowledge Engineering Review 11, 93–136.

Vizcaino, A., Soto, J.P., Portillo, J., Piattini, M., 2007. A Multi-agent Model to Develop Knowledge Management Systems, in: 40th Annual Hawaii International Conference on System Sciences, 2007. HICSS 2007. Presented at the 40th Annual Hawaii International Conference on System Sciences, 2007. HICSS 2007, p. 203b.

Vrandecic, D., Krötzsch, M., 2006. Reusing ontological background knowledge in semantic wikis, in: Proceedings of the 1st Workshop on Semantic Wikis, From Wiki To Semantics, AIFB, ESWC2006. Budva, Montenegro.

Wang, X., Xu, F., 2010. Study on knowledge management system based on MAS, in: 2010 2nd International Conference on Networking and Digital Society (ICNDS). Presented at the 2010 2nd International Conference on Networking and Digital Society (ICNDS), pp. 395 –398.

Weiss, G. (Ed.), 1999. Multiagent systems: a modern approach to distributed artificial intelligence. MIT Press, Cambridge, MA, USA.

Xu, J., Houssin, R., Caillaud, E., Gardoni, M., 2011. Fostering continuous innovation in design with an integrated knowledge management approach. Computers in Industry 62, 423–436.

Yang, S.-Y., Chang, Y.-Y., 2011. An active and intelligent network management system with ontology-based and multi-agent techniques. Expert Systems with Applications 38, 10320–10342.

Yu, E.S.K., 1997. Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering, in: Proceedings of the 3rd IEEE International Symposium on Requirements Engineering, RE '97. IEEE Computer Society, Washington, DC, USA, p. 226.

Zhang, C., Tang, D., Liu, Y., You, J., 2008. A Multi-agent Architecture for Knowledge Management System, in: Fifth International Conference on Fuzzy Systems and Knowledge Discovery, 2008. FSKD '08. Presented at the Fifth International Conference on Fuzzy Systems and Knowledge Discovery, 2008. FSKD '08, pp. 433–437.

Zhang, X., Liu, W., Jin, F., 2006. Intelligent Agent for Knowledge Managemnt in E-Commerce, in: 2006 1st International Symposium on Pervasive Computing and Applications. Presented at the 2006 1st International Symposium on Pervasive Computing and Applications, pp. 455 –460.

Zhang, Y., Huang, G.Q., Qu, T., Ho, O., 2010. Agent-based workflow management for RFID-enabled real-time reconfigurable manufacturing. Int. J. Comput. Integr. Manuf. 23, 101–112.

Publications

Revues :

1. Inaya Lahoud, Davy Monticolo, Vincent Hilaire, “Mapping the semantic web to SQL query to extract knowledge”, *Journal of E-technology*, volume 3, issue 2, Pages: 69-86, May 2012

Chapitres de livre :

1. Inaya Lahoud, Davy Monticolo, Vincent Hilaire, Samuel Gomes, “a Semantic Wiki to support knowledge sharing in innovation activities”, *Edited book : IAENG Transactions on Engineering Technologies - Special Issue of the International MultiConference of Engineers and Computer Scientists 2012,* Chapitre 16, Edition: Yang, G.-C.; Ao, S.-I.; Huang, X.; Castillo, O., Lecture Notes in Electrical Engineering, Vol. 186, 368p. Springer 2013

2. Inaya Lahoud, Davy Monticolo, Vincent Hilaire, Samuel Gomes, “a multi-agent platform to manage distributed and heterogeneous knowledge by using semantic web”, *edited book: IAENG Transactions on Electrical Engineering Volume 1 - Special Issue of the International MultiConference of Engineers and Computer Scientists 2012, en cours de publication*

Conférences internationales :

1. Lahoud, I., Monticolo, D., Gomes, S. 2010. OCEAN: A Semantic Web Service to Extract Knowledge in E-Groupwares. *2010 Sixth International Conference on Signal-Image Technology and Internet-Based Systems (SITIS),* pp.354-362, 15-18 Dec. 2010, Kuala Lumput, Malaysia
   DOI=10.1109/SITIS.2010.64

2. Lebouteiller M., Lahoud I., Lebaal N., Gomes S. 2011. Vers une ingénierie hautement productive basée sur des méthodes et outils d’extraction de connaissances. *9e Congrès International de Génie Industriel*, octobre 2011, Canada

3. Monticolo D., Lahoud I., Camargo M., Morel L. 2011. SemKnow, une architecture basée sur les Web Services Sémantiques pour la gestion des connaissances multi-sources lors des projets de développement de produit. *9e Congrès International de Génie Industriel*, octobre 2011, Canada

4. Davy Monticolo, Inaya Lahoud, Eric Bonjour, Frédéric Demoly , “SemKnow: A Multi-Agent Platform to Manage Distributed Knowledge by using Ontologies”,

Proceedings of the International MultiConference of Engineers and computer scientists, The IAENG International Conference on Artificial Intelligence and Applications (ICAIA'12), vol 1, Pages 58-62*,* 14-16 March 2012, Hong Kong

5. Davy Monticolo, Laure Morel, Vincent Boly, Inaya lahoud, "Wiki-I: A Semantic Wiki to Support the Ideas Development and Knowledge Sharing in Innovation Activities", *Proceedings of the International MultiConference of Engineers and computer scientists, The IAENG International Conference on Industrial Engineering (ICINDE'12), vol 2, Pages* 1307-1311*,* 14-16 March 2012, Hong Kong

6. Inaya Lahoud, Davy Monticolo, Vincent Hilaire, Samuel Gomes, "A metamodeling and transformation approach for knowledge extraction", *The Fourth International Conference on Networked Digital Technologies (NDT'2012),* 24-26 april 2012, Dubai, UAE

7. Inaya Lahoud ,Davy Monticolo, Vincent Hilaire, Samuel Gomes, Eric Bonjour, "A multi-sources knowledge management system", *14th IFAC Symposium on Information Control Problems in Manufacturing (INCOM'12),* 23-25 May 2012, Bucharest, Romania

8. Monticolo, Davy; Lahoud, Inaya; Bonjour, Eric. "Distributed knowledge extracted by a mas using ontology alignment methods", *International Conference on Computer & Information Science (ICCIS)*, vol 1, Pages 386 - 391, 12-14 June 2012, Kuala Lumpur, Malaysia

Présentations et posters :

1. Inaya Lahoud, Davy Monticolo, Vincent Hilaire, Poster : « Gestion des connaissances hétérogènes et distribuées au sein d'une entreprise étendue », *1ères journées des jeunes chercheurs de l'UTBM (ingédoc 2011)*, 1 et 2 décembre 2011, Sévenans, France

2. Inaya lahoud, Davy Monticolo, Vincent Hilaire. Présentation : "Un système multi-agent pour la gestion de connaissances hétérogènes et distribuées », *15ème journées STP du Gdr MACS*, 17 et 18 novembre 2011, tarbes, France

3. École d'été et présentation d'un poster: "managing heterogeneous and distributed knowledge in extended enterprises" au SSSW'12, The 9th Summer School on Ontology Engineering and the Semantic Web, 8 – 14 Juillet 2012. Cercedilla, near Madrid (Spain)

Organisation Workshop :

1. Organisation du workshop international KARE 2011 (*Knowledge Acquisition, Reuse and Evaluation*) avec Davy Monticolo dans la *septième conférence internationale*

Signal Image Technology & Internet Based Systems (SITIS), du 28 Novembre à 1 Décembre 2011, Dijon, France

Un système multi-agents pour la gestion des connaissances hétérogènes et distribuées

Résumé

La gestion des connaissances permet d'identifier et de capitaliser les savoirs faires de l'entreprise afin de les organiser et de les diffuser. Cette thèse propose un système de gestion des connaissances hétérogènes et distribuées, appelé OCEAN. Basé sur les ontologies et sur un système multi-agents, OCEAN a pour but de résoudre le problème de la capitalisation et de réutilisation des connaissances provenant de plusieurs sources différentes, afin d'aider les acteurs métiers dans le processus de développement de produits mécaniques. Le système OCEAN repose sur un cycle de vie de quatre étapes Ce cycle de vie possède les phases : d'identification, d'extraction, de validation et se termine par la réutilisation des connaissances. Chaque phase constitue l'objectif d'une organisation d'agents.

L'identification dans le système OCEAN consiste à définir les connaissances par un expert métier sous la forme d'une ontologie. Les ontologies sont utilisées dans notre système pour représenter les connaissances définis d'une façon structurée et formelle afin d'être compréhensible par les machines. L'extraction des connaissances dans OCEAN est réalisée par les agents de manière automatique à l'aide des ontologies créées par les experts métiers. Les agents interagissent avec les différentes applications métiers via des services web. Le résultat de cette phase est stocké dans une mémoire organisationnelle. La validation des connaissances consiste à permettre aux acteurs métiers de valider les connaissances de la mémoire organisationnelle dans un wiki sémantique. Ce wiki permet de présenter les connaissances de la mémoire organisationnelle aux acteurs pour les réutiliser, les évaluer et les faire évoluer. La réutilisation des connaissances dans OCEAN est inspirée de travaux antérieurs intégrés au sein d'OCEAN. Les quatre phases du cycle de vie des connaissances traitées dans cette thèse nous ont permis de réaliser un système apte à gérer les connaissances hétérogènes et distribuées dans une entreprise étendue.

Mots-clés : gestion des connaissances, ontologies, systèmes multi-agents, wiki sémantique, services web

Multi-agents system for heterogeneous and distributed knowledge management

Abstract

Among the goals of Knowledge Management we can cite the identification and capitalization of the know-how of companies in order to organize and disseminate them. This thesis proposes a heterogeneous and distributed knowledge management system, called OCEAN. Based on ontologies and multi-agents system, OCEAN aims to solve the problem of capitalization and reuse of multi-sources knowledge in order to assist business actors in the development process of mechanical products. The OCEAN system is based on a knowledge life cycle composed by four steps. This knowledge life cycle begins with the identification then extraction, validation and finishes with knowledge reuse. Each step is the goal of an organization of agents.

The identification in OCEAN system consists in the definition of knowledge by a business expert with an ontology. Ontologies are used in our system to represent the knowledge, defined by the business expert, in a structured and formal way in order to be understandable by machines. Agents according to the ontology defined by business experts realize knowledge extraction in OCEAN automatically. Agents interact with professional softwares via web services. The result of this extraction is stored in an organizational memory (OM). Validation of knowledge in OCEAN relies on business actors that validate the knowledge of the OM in a semantic wiki. This wiki allows also the presentation of this knowledge to business actors in order to reuse, evaluate or evolve it. Previous works, integrated within OCEAN, inspires the knowledge reuse step. The four steps lifecycle discussed in this thesis has enabled us to achieve a system that can manage heterogeneous and distributed knowledge in an extended enterprise.

Keywords: knowledge management, ontologies, multi-agent systems, semantic wiki, web services

SPIM

■ École doctorale SPIM - Université de Technologie Belfort-Montbéliard
F - 90010 Belfort Cedex ■ tél. +33 (0)3 84 58 31 39
■ ed-spim@univ-fcomte.fr ■ www.ed-spim.univ-fcomte.fr

utbm
université de technologie
Belfort-Montbéliard